



IBM Informix Dynamic Server Administrator's Reference



IBM Informix Dynamic Server Administrator's Reference

Note

Before using this information and the product it supports, read the information in "Notices" on page H-1.

This edition replaces SC23-7749-01.

This document contains proprietary information of IBM. It is provided under a license agreement and is protected by copyright law. The information contained in this publication does not include any product warranties, and any statements provided in this manual should not be interpreted as such.

When you send information to IBM, you grant IBM a nonexclusive right to use or distribute the information in any way it believes appropriate without incurring any obligation to you.

© **Copyright International Business Machines Corporation 1996, 2008.**

US Government Users Restricted Rights – Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM Corp.

Contents

Introduction	xix
In This Introduction	xix
About This Publication	xix
Types of Users	xix
Software Dependencies	xix
Assumptions About Your Locale	xx
Demonstration Database	xx
What's New in Administrator's Reference for Dynamic Server, Version 11.50	xx
Documentation Conventions	xxiii
Typographical Conventions	xxiii
Feature, Product, and Platform Markup	xxiv
Example Code Conventions	xxiv
Additional Documentation	xxiv
Compliance with Industry Standards	xxv
Syntax Diagrams	xxv
How to Read a Command-Line Syntax Diagram	xxvi
Keywords and Punctuation	xxvii
Identifiers and Names	xxvii
How to Provide Documentation Feedback	xxvii

Part 1. Configuring and Monitoring Dynamic Server

Chapter 1. Configuration Parameters	1-1
ONCONFIG File Conventions	1-1
Format of ONCONFIG File	1-1
ONCONFIG File Templates	1-1
Displaying ONCONFIG Settings	1-2
Summary of Configuration Parameters	1-3
Parameter Attributes	1-9
Using a Utility to Change a Parameter Value	1-10
Environment Variables	1-10
Archecker Configuration Parameters	1-10
ADMIN_MODE_USERS	1-11
ADMIN_USER_MODE_WITH_DBSA	1-12
ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)	1-12
ALARMPROGRAM	1-12
ALLOW_NEWLINE	1-13
ALRM_ALL_EVENTS	1-13
AUTO_AIOVPS	1-14
AUTO_CKPTS.	1-14
AUTO_LRU_TUNING	1-15
AUTO_REPREPARE	1-15
BLOCKTIMEOUT	1-16
BTSCANNER	1-16
BUFFERPOOL.	1-17
CKPTINTVL	1-21
CLEANERS.	1-22
CONSOLE	1-22
DATASKIP	1-22
DBCREATE_PERMISSION.	1-23
DB_LIBRARY_PATH.	1-24
DBSERVERALIASES.	1-24
DBSERVERNAME	1-25
DBSPACETEMP	1-26
Using Hash Join Overflow and DBSPACETEMP	1-28

DD_HASHMAX	1-28
DD_HASHSIZE	1-29
DEADLOCK_TIMEOUT	1-29
DEF_TABLE_LOCKMODE	1-30
DIRECT_IO (UNIX)	1-31
DIRECTIVES	1-31
DISABLE_B162428_XA_FIX	1-32
DRDA_COMMBUFSIZE	1-32
DRAUTO	1-33
DRIDXAUTO	1-34
DRINTERVAL	1-34
DRLOSTFOUND	1-35
DRTIMEOUT	1-35
DS_HASHSIZE	1-36
DS_MAX_QUERIES	1-36
DS_MAX_SCANS	1-37
DS_NONPDQ_QUERY_MEM	1-38
DS_POOLSIZ	1-38
DS_TOTAL_MEMORY	1-39
Algorithm for DS_TOTAL_MEMORY	1-40
DUMPCNT (UNIX)	1-40
DUMPCORE (UNIX)	1-41
DUMPDIR	1-41
DUMPGCORE (UNIX)	1-41
DUMPSHMEM (UNIX)	1-42
DYNAMIC_LOGS	1-43
ENCRYPT_CIPHERS	1-44
ENCRYPT_HDR	1-45
ENCRYPT_MAC	1-46
ENCRYPT_MACFILE	1-46
ENCRYPT_SMX	1-47
ENCRYPT_SWITCH	1-48
Enterprise Replication Configuration Parameters	1-48
EXPLAIN_STAT	1-50
EXT_DIRECTIVES	1-50
EXTSHMADD	1-51
FAILOVER_CALLBACK	1-51
FASTPOLL	1-51
FILLFACTOR	1-52
HA_ALIAS	1-52
HETERO_COMMIT	1-52
IFX_EXTEND_ROLE	1-53
IFX_FOLDVIEW	1-53
ISM_DATA_POOL and ISM_LOG_POOL	1-54
Java Configuration Parameters	1-54
LIMITNUMSESSIONS	1-55
LISTEN_TIMEOUT	1-56
LOCKS	1-56
LOGBUFF	1-57
LOGFILES	1-58
LOG_INDEX_BUILDS	1-58
LOGSIZE	1-59
LOGSIZE for Smart Large Objects	1-60
LTAPEBLK	1-60
LTAPEDEV	1-61
LTAPESIZE	1-61
LTXEHWM	1-62
LTXHWM	1-62
MAX_FILL_DATA_PAGES	1-63
MAX_INCOMPLETE_CONNECTIONS	1-64
MAX_PDQPRIORITY	1-65

MaxConnect Configuration Parameters	1-65
MIRROR	1-66
MIRROROFFSET	1-67
MSG_DATE	1-67
MSGPATH	1-68
MULTIPROCESSOR	1-68
NETTYPE	1-69
Protocol	1-70
Number of Poll Threads	1-70
Number of Connections	1-71
Class of Virtual Processor	1-71
Default Values	1-71
Multiplexed Connections	1-72
IBM Informix MaxConnect.	1-72
OFF_RECVRY_THREADS	1-72
ON_RECVRY_THREADS	1-72
ON-Bar Configuration Parameters	1-73
ONDBSPACEDOWN	1-74
Database Server Behavior When ONDBSPACEDOWN Does Not Apply.	1-74
ONLIDX_MAXMEM.	1-74
OPCACHEMAX (UNIX)	1-75
OPTCOMPIND	1-76
OPT_GOAL	1-76
PC_HASHSIZE	1-77
PC_POOLSIZE.	1-78
PHYSBUFF	1-78
PHYSFILE	1-79
PLOG_OVERFLOW_PATH	1-79
QSTATS	1-80
RA_PAGES	1-80
RA_THRESHOLD	1-80
UPDATABLE_SECONDARY	1-81
RESIDENT	1-82
RESTARTABLE_RESTORE	1-82
ROOTNAME	1-83
ROOTOFFSET	1-84
ROOTPATH	1-84
ROOTSIZE	1-85
RTO_SERVER_RESTART	1-85
SBSPACENAME	1-86
SBSPACETEMP	1-87
SDS_ENABLE	1-87
SDS_PAGING	1-88
SDS_TEMPDBS	1-88
SDS_TIMEOUT	1-89
SECURITY_LOCALCONNECTION.	1-90
SERVENUM	1-90
SHMADD	1-90
SHMBASE	1-91
SHMNOACCESS	1-92
SHMTOTAL	1-92
SHMVIRT_ALLOCSEG	1-93
SHMVIRT_SIZE.	1-93
SINGLE_CPU_VP	1-94
User-Defined VP Classes and SINGLE_CPU_VP	1-95
SQLTRACE	1-95
SSL_KEYSTORE_LABEL	1-96
STACKSIZE.	1-96
STAGEBLOB	1-97
STMT_CACHE	1-97
STMT_CACHE_HITS	1-98

STMT_CACHE_NOLIMIT	1-99
STMT_CACHE_NUMPOOL	1-99
STMT_CACHE_SIZE	1-100
STORAGE_FULL_ALARM	1-100
SYSALARMPROGRAM	1-101
SYSBSPACENAME	1-101
TAPEBLK	1-103
TAPEDEV	1-103
TAPESIZE	1-104
TBLSPACE_STATS	1-105
TBLTBLFIRST	1-105
TBLTBLNEXT	1-105
TEMPTAB_NOLOG	1-106
TXTIMEOUT	1-106
UNSECURE_ONSTAT	1-107
USELASTCOMMITTED	1-107
USEOSTIME	1-108
VP_MEMORY_CACHE_KB	1-109
VPCLASS	1-109
WSTATS	1-114

Chapter 2. The sysmaster Database 2-1

The sysmaster Database	2-1
The buildsmi Script	2-1
The bldutil.sh Script	2-2
The System-Monitoring Interface	2-2
Understanding the SMI Tables	2-2
Accessing SMI Tables	2-3
The System-Monitoring Interface Tables	2-4
The sysutils Tables	2-6
sysadinfo	2-7
sysaudit	2-7
syschkio	2-8
syscheckpoint	2-8
syschunks	2-9
syscsm	2-10
syscsmsla	2-10
syscsmstab	2-11
sysconfig	2-11
sysdatabases	2-11
sysdblocale	2-12
sysdbspaces	2-12
sysdri	2-13
sysdual	2-14
sysenv	2-14
sysenvses	2-14
sysextents	2-14
sysextspaces	2-14
sysha_latency	2-15
sysha_type	2-16
sysha_workload	2-16
sysipl	2-17
syslocks	2-17
syslogs	2-18
sysmgminfo	2-18
sysnetclienttype	2-19
sysnetglobal	2-19
sysnetworkio	2-20
sysonlinelog	2-21
sysprofile	2-21
sysproxyagents	2-22

Time Stamps	4-26
Database and Table Creation: What Happens on Disk.	4-27
Database Creation	4-27
Table Creation	4-27

Chapter 5. Interpreting Logical-Log Records 5-1

About Logical-Log Records	5-1
Transactions That Drop a Table or Index	5-1
Transactions That Are Rolled Back	5-1
Checkpoints with Active Transactions	5-2
Distributed Transactions	5-2
Logical-Log Record Structure	5-2
Logical-Log Record Header	5-2
Logical-Log Record Types and Additional Columns.	5-3
Log Record Types for Smart Large Objects	5-14

Part 2. Administrative Utilities

Chapter 6. Overview of Utilities 6-1

Complete List of Utilities	6-1
Obtaining Utility Version Information	6-1
Multibyte Characters (GLS)	6-2
OpenAdmin Tool for IDS	6-2
IBM Informix Server Administrator	6-3
Server Studio	6-3

Chapter 7. The oncheck Utility 7-1

oncheck Check-and-Repair	7-1
What Does Each Option Do?	7-1
Using the -y Option to Perform Repairs.	7-2
Repairing Indexes in Sbspaces and External Spaces	7-2
Locking and oncheck	7-3
Oncheck Syntax.	7-3
oncheck -cc and -pc: Check system catalog tables.	7-8
oncheck -cd and -cD: Check pages.	7-8
oncheck -ce, -pe: Check the chunk-free list.	7-9
oncheck -ci and -cI: Check index node links	7-10
oncheck -cr and -cR: Check reserved pages	7-11
oncheck -cs, -cS, -ps, -pS: Check and display sbspaces	7-12
oncheck -pB: Display blobspace statistics	7-12
oncheck -pd and pD: Display rows in hexadecimal format	7-12
oncheck -pk, -pK, -pl, -pL: Display index information.	7-14
oncheck -pp and -pP: Display the contents of a logical page	7-15
oncheck -pr and pR: Display reserved-page information	7-17
oncheck -pt and -pT: Display tblspaces for a table or fragment.	7-18
Turn On Locking with -x	7-19
Send Special Arguments to the Access Method with -u	7-19
Return Codes on Exit	7-19

Chapter 8. The ondblog Utility 8-1

ondblog: Change Logging Mode	8-1
ondblog Syntax	8-1

Chapter 9. The oninit Utility 9-1

oninit: Initialize the Database Server.	9-1
Initializing the Server in Administrative Mode with the -j Option	9-1
Initializing the Server in Wait Mode with the -w Option	9-2
oninit Syntax	9-2
Initialize Shared Memory Only	9-3

Initialize Disk Space and Shared Memory 9-4

Chapter 10. The onlog Utility 10-1

Chapter 11. The onmode Utility 11-1

onmode Syntax	11-2
onmode -a: Add a shared-memory segment	11-3
onmode -BC: Allow large chunk mode.	11-3
onmode -c: Force a checkpoint	11-4
onmode -C: Control the B-tree scanner.	11-4
onmode -d: Set data-replication types	11-5
Using the -d standard Option.	11-6
Using the -d primary dbservername Option	11-6
Using the -d secondary dbservername Option	11-6
onmode -d: Set High Availability server characteristics	11-7
onmode -d: Replicate an index with data-replication	11-8
onmode -D, -M, -Q, -S: Change decision-support parameters	11-10
onmode -e: Change usage of the SQL statement cache	11-11
onmode -F: Free unused memory segments	11-11
onmode -k, -m, -s, -u, -j: Change database server mode.	11-12
Taking the Database Server to Offline Mode with the -k Option	11-13
Bringing the Database Server Online with the -m Option	11-13
Shutting Down the Database Server Gracefully with the -s Option	11-13
Shutting Down the Database Server Immediately with the -u Option	11-13
Changing the Database Server to Administration Mode with the -j Option	11-14
Changing Database Server Mode with ON-Monitor (UNIX)	11-14
onmode -l: Switch the logical-log file	11-15
onmode -n, -r: Change shared-memory residency.	11-15
onmode -O: Override ONDBSPACEDOWN WAIT mode	11-15
onmode -p: Add or remove virtual processors	11-16
Adding and Dropping Virtual Processors	11-17
Dropping Virtual Processors Automatically	11-18
Monitoring Poll Threads with onstat	11-18
onmode -R: Regenerate .infos File	11-18
onmode -W: Change settings for the SQL statement cache	11-19
SQL Statement Cache Examples	11-19
onmode -wf, -wm: Dynamically change certain configuration parameters.	11-20
onmode -wm: Change LRU tuning status	11-21
onmode -Y: Dynamically change SET EXPLAIN	11-22
onmode -z: Kill a database server session	11-23
onmode -Z: Kill a distributed transaction	11-23

Chapter 12. The ON-Monitor Utility 12-1

Using ON-Monitor (UNIX)	12-1
Navigating ON-Monitor and Using Help	12-1
Executing Shell Commands Within ON-Monitor	12-2
ON-Monitor Screen Options	12-2
Setting Configuration Parameters in ON-Monitor	12-3

Chapter 13. The onparams Utility 13-1

onparams Syntax	13-1
onparams -a -d <i>dbspace</i> : Add a logical-log file	13-2
onparams -d -l <i>lognum</i> : Drop a logical-log file	13-2
onparams -p: Change physical-log parameters	13-3
Backing Up After You Change the Physical-Log Size or Location	13-4
Changing the Size of the Physical Log and Using Non-Default Page Sizes	13-4
Using a Text Editor to Change the Physical-Log Size or Location	13-4
onparams -b: Add a new buffer pool	13-4
Examples of onparams Commands	13-6

Chapter 14. The onspaces Utility. 14-1

onspaces Syntax	14-1
Administrative API <i>command</i> string.	14-2
onspaces -a: Add a chunk to a dbspace or blobspace	14-3
onspaces -a: Add a chunk to an sbspace	14-4
onspaces -c -b: Create a blobspace	14-5
onspaces -c -d: Create a dbspace.	14-7
Creating a Temporary Dbspace with the -t Option	14-10
Specifying First and Next Extent Size for the tblspace tblspace	14-10
Specifying a Non-Default Page Size with the Same Size as the Buffer Pool	14-10
onspaces -c -S: Create an sbspace	14-11
Creating a Temporary Sbspace with the -t Option	14-12
Creating an Sbspace with the -Df option.	14-12
Changing the -Df Settings	14-15
Using the onspaces -g Option	14-16
onspaces -c -x: Create an extspace	14-16
onspaces -ch: Change sbspace default specifications	14-17
onspaces -cl: Clean up stray smart large objects in sbspaces	14-17
onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace	14-18
onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace.	14-20
onspaces -f: Specify DATASKIP parameter	14-21
onspaces -m: Start mirroring.	14-21
Using a File to Specify Chunk-Location Information with the -f Option	14-23
onspaces -r: Stop mirroring	14-23
onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace	14-24
Renaming a Dbspace, Blobspace, Sbspace, or Extspace when Enterprise Replication Is Active	14-25
Performing an Archive after Renaming a Space	14-25
onspaces -s: Change status of a mirrored chunk	14-25

Chapter 15. The onstat Utility 15-1

Monitor the Database Server Status.	15-2
onstat Syntax	15-2
Interactive Execution	15-6
Continuous onstat Execution	15-6
onstat: onstat -pu equivalent	15-6
onstat -: Print output header	15-6
Logs Full Subheader.	15-7
onstat --: Print onstat options and functions	15-7
Running onstat Commands on a Shared Memory Dump File	15-8
onstat -a: Print onstat -cuskbtdlp	15-9
onstat -b: Print buffer information for buffers in use	15-9
onstat -B: Print all buffer information.	15-10
onstat -c: Print ONCONFIG file contents	15-12
onstat -C: Print B-tree scanner information	15-13
onstat -d: Print chunk information.	15-19
Using onstat -d with Sbspaces	15-22
Using onstat -d with Blobspaces	15-22
onstat -D: Print page-read and page-write information	15-22
onstat -f: Print dbspace information affected by dataskip	15-23
onstat -F: Print counts	15-23
onstat -g Monitoring Options	15-25
onstat -g act: Print active threads	15-30
onstat -g afr: Print allocated memory fragments	15-31
onstat -g all: Print output from all onstat -g options	15-31
onstat -g ath: Print information about all threads	15-31
onstat -g buf: Print buffer pool profile information	15-32
onstat -g cat: Print ER global catalog information.	15-34
onstat -g cdr config: Print ER settings	15-36
onstat -g ckp: Print checkpoint history and configuration recommendations	15-38
onstat -g cmsm: Print Connection Manager information	15-40
onstat -g con: Print condition and thread information	15-41

onstat -g cpu: Print runtime statistics	15-42
onstat -g dbc: Print dbScheduler and dbWorker thread statistics	15-43
onstat -g ddr: Print ER database log reader status	15-45
onstat -g dic: Print table information	15-46
onstat -g dis: Print database server information	15-47
onstat -g dll: Print dynamic libraries list	15-48
onstat -g dmp: Print raw memory	15-49
onstat -g dri: Print High-Availability Cluster information	15-50
onstat -g dsc: Print distribution cache information	15-51
onstat -g dss: Print ER environment data	15-52
onstat -g dtc: Print delete table cleaner statistics	15-52
onstat -g env: Print environment variable values	15-53
onstat -g ffr: Print free fragments	15-55
onstat -g glo: Print global multithreading information	15-56
onstat -g grp: Print ER grouper statistics.	15-57
onstat -g his: Print SQLTRACE information	15-62
onstat -g ioa: Print combined onstat -g information	15-64
onstat -g iob: Print big buffer use summary	15-65
onstat -g iof: Print asynchronous I/O statistics	15-66
onstat -g iog: Print AIO global information	15-67
onstat -g ioq: Print I/O queue information	15-67
onstat -g ipl: Print index page logging status information	15-69
onstat -g iov: Print AIO VP statistics	15-69
onstat -g lap: Print light appends status information	15-71
onstat -g lmx: Print all locked mutexes	15-71
onstat -g lsc: Print active light scan status	15-72
onstat -g mem: Print pool memory statistics	15-73
onstat -g mgm: Print MGM resource information	15-74
onstat -g nbm: Print a block bit map	15-77
onstat -g nif: Print ER network interface statistics.	15-78
onstat -g nsc <i>client_id</i> : Print current shared memory connection information	15-79
onstat -g nsd: Print poll threads shared-memory data	15-81
onstat -g nss: Print shared memory network connections status	15-82
onstat -g ntd: Print network statistics	15-83
onstat -g ntm: Print network mail statistics	15-83
onstat -g ntt: Print network user times	15-84
onstat -g ntu: Print network user statistics	15-84
onstat -g opn: Print open partitions	15-85
onstat -g pos: Print file values	15-87
onstat -g ppf: Print partition profiles	15-88
onstat -g prc: Print sessions using UDR or SPL routine	15-89
onstat -g proxy: Print Proxy Distributor information.	15-90
onstat -g que: Prints ER queue statistics	15-95
onstat -g qst: Print wait options for mutex and condition queues	15-96
onstat -g rbm: Print a block map of shared memory	15-97
onstat -g rcv: Print ER receive manager statistics	15-98
onstat -g rea: Print ready threads	15-101
onstat -g rep: Print ER schedule manager events	15-101
onstat -g rqm: Print low-level queue statistics	15-102
onstat -g rss: Print RS secondary server information	15-104
onstat -g rwm: Print read and write mutexes	15-107
onstat -g sch: Print VP information	15-107
onstat -g sds: Print SD secondary server information	15-108
onstat -g seg: Print shared memory segment statistics	15-112
onstat -g ses: Print session-related information	15-113
onstat -g sle: Print all sleeping threads	15-117
onstat -g smb: Print sbspaces information	15-118
onstat -g smx: Print multiplexer group information	15-119
onstat -g spi: Print spin locks with long spins	15-121
onstat -g sql: Print SQL-related session information.	15-122
onstat -g src: Patterns in shared memory	15-124

onstat -g ssc: Print SQL statement occurrences	15-124
onstat -g stk <i>tid</i> : Print thread stack	15-126
onstat -g stm: Print SQL statement memory usage	15-126
onstat -g stq: Print queue information	15-127
onstat -g sts: Print stack usage per thread	15-127
onstat -g sync: Print ER synchronization status	15-128
onstat -g tpf <i>tid</i> : Print thread profiles	15-130
onstat -g ufr: Print memory pool fragments	15-130
onstat -g vpcache: Print CPU VP memory block cache statistics.	15-132
onstat -g wai: Print wait queue thread list	15-133
onstat -g wmx: Print all mutexes with waiters	15-134
onstat -g wst: Print wait statistics for threads	15-134
onstat -G: Print TP/XA transaction information	15-136
onstat -h: Print buffer header hash chain information	15-138
onstat -i: Initiate interactive mode	15-139
onstat -j: Provide onpload status information	15-139
onstat -k: Print active lock information	15-141
onstat -l: Print physical and logical log information.	15-142
onstat -m: Print recent system message log information	15-145
onstat -o: Output shared memory contents to a file.	15-146
onstat -O: Print optical subsystem information	15-146
onstat -p: Print profile counts	15-148
onstat -P: Print partition information	15-151
onstat -r: Repeatedly print selected statistics	15-152
onstat -R: Print LRU, FLRU, and MLRU queue information	15-155
onstat -s: Print latch information	15-157
onstat -t and -T: Print tblspace information	15-159
onstat -u: Print user activity profile	15-161
onstat -x: Print database server transaction information	15-163
Determining the Position of a Logical-Log Record	15-164
Determining the Mode of a Global Transaction	15-165
onstat -X: Print thread information	15-165
onstat -z: Clear statistics.	15-167
Return Codes on Exit.	15-167

Chapter 16. The oncmsm Utility 16-1

Chapter 17. The onpassword Utility 17-1

Part 3. Appendixes

Appendix A. Files That the Database Server Uses. A-1

Database Server Files.	A-1
Descriptions of Files	A-3
af.xxx	A-3
ac_msg.log	A-3
ac_config.std	A-4
bar_act.log	A-4
bldutil.process_id	A-4
buildsmi.out (UNIX) or buildsmi_out (Windows)	A-4
concdr.sh.	A-4
.conf.dbservername	A-4
core	A-5
Emergency Boot Files for ON-Bar	A-5
gcore.xxx (UNIX)	A-5
illlsra.xx.	A-5
~/informix	A-5
informix.rc (UNIX)	A-6
INFORMIXTMP	A-6
.inf.servicename	A-6

.infos.dbservername	A-6
.infxdirs	A-6
InstallServer.log (Windows)	A-6
ISM Catalog.	A-6
ISM Logs.	A-7
ISMversion	A-7
JVM_vpid	A-7
JVPLOG	A-7
.jvpprops.	A-7
Message Log	A-7
onconfig.std.	A-8
The ONCONFIG File.	A-8
onconfig	A-8
oncfg_servername.servernum	A-8
onsnmp.servername	A-9
onsrvapd.log	A-9
revcdr.sh	A-9
shmem.xxx (UNIX)	A-9
sm_versions.std	A-9
snmpd.log	A-9
sqlhosts	A-9
VP.servername.nnx	A-10
xbsa.messages.	A-10
Appendix B. Trapping Errors	B-1
Collecting Diagnostics using onmode -I.	B-1
Syntax.	B-1
Creating Tracepoints	B-1
Appendix C. Event Alarms	C-1
Using ALARMPROGRAM to Capture Events.	C-1
Setting ALRM_ALL_EVENTS	C-1
Writing Your Own Alarm Script	C-1
Customizing the ALARMPROGRAM scripts	C-1
Precautions for Foreground Operations in Alarm Scripts	C-2
Interpreting Error Messages	C-2
Event-Alarm Parameters	C-3
Event Severity	C-3
Event Alarms on Dynamic Server	C-4
RS Secondary Server Event Alarms	C-5
SD Secondary Server Event Alarms	C-6
Appendix D. Discontinued Configuration Parameters	D-1
AFF_NPROCS (Discontinued).	D-1
AFF_SPROC (Discontinued)	D-2
BUFFERS (Discontinued)	D-3
FAST_RESTART_CKPT_FUZZYLOG (Discontinued)	D-3
FAST_RESTART_PHYSLOG (Discontinued)	D-4
JDKVERSION (Discontinued)	D-5
LBU_PRESERVE (Discontinued)	D-5
LOGSMAX (Discontinued)	D-5
LRU_MAX_DIRTY (Discontinued)	D-5
LRU_MIN_DIRTY (Discontinued)	D-6
LRUS (Discontinued).	D-6
NOAGE (Discontinued)	D-7
NUMAIOVPS (Discontinued)	D-7
NUMCPUVPS (Discontinued)	D-8
PHYSDBS (Discontinued)	D-9
Appendix E. Error Messages	E-1

How the Messages Are Ordered in This Chapter	E-1
How to View These Messages	E-1
Message Categories	E-2
Messages: A-B	E-2
Aborting Long Transaction: <i>tx Oxn</i>	E-2
Affinitied VP <i>mm</i> to phys proc <i>nn</i>	E-2
Affinity not enabled for this server.	E-3
Assert Failed: Error from SBSpace cleanup thread.	E-3
Assert Failed: Short description of what failed Who: Description of user/session/thread running at the time	
Result: State of the affected database server entity Action: What action the database administrator should take	
See Also: DUMPDIR/af.uniqid containing more diagnostics..	E-3
Begin re-creating indexes deferred during recovery..	E-3
Building 'sysmaster' database requires ~ <i>mm</i> pages of logical log. Currently there are <i>nn</i> pages available.	
Prepare to back up your logs soon.	E-4
Building 'sysmaster' database...	E-4
Messages: C	E-4
Cannot Allocate Physical-log File, <i>mm</i> wanted, <i>nn</i> available..	E-4
Cannot alter a table which has associated violations table.	E-4
Cannot change to mode..	E-4
Cannot Commit Partially Complete Transactions.	E-5
Cannot create a user-defined VP class with 'SINGLE_CPU_VP' non-zero.	E-5
Cannot create violations/diagnostics table.	E-5
Cannot insert from the violations table to the target table.	E-6
Cannot modify/drop a violations/diagnostics table.	E-6
Cannot Open Dbspace <i>mmm</i>	E-6
Cannot Open Logical Log.	E-7
Cannot Open Mirror Chunk <i>pathname</i> , errno = <i>nn</i>	E-7
Cannot Open Primary Chunk <i>pathname</i> , errno = <i>mmm</i>	E-7
Cannot Open Primary Chunk <i>chunkname</i>	E-7
Cannot open sysams in database <i>name</i> , iserrno <i>number</i>	E-7
Cannot open sysdistrib in database <i>name</i> , iserrno <i>number</i>	E-8
Cannot open <i>system_table</i> in database <i>name</i> , iserrno <i>number</i>	E-8
Cannot open systrigbody in database <i>name</i> , iserrno <i>number</i>	E-8
Cannot open systriggers in database <i>name</i> , iserrno <i>number</i>	E-8
Cannot open sysxdtypes in database <i>name</i> , iserrno <i>number</i>	E-8
Cannot Perform Checkpoint, shut system down..	E-8
Cannot Restore to Checkpoint..	E-9
Cannot Rollback Incomplete Transactions.	E-9
Cannot update pagezero.	E-9
Cannot update syscasts in database <i>name</i> . Iserrno <i>number</i>	E-9
Can't affinity VP <i>mm</i> to phys proc <i>nn</i>	E-10
Changing the sbspace minimum extent value: old value <i>value1</i> , new value <i>value2</i>	E-10
Checkpoint blocked by down space, waiting for override or shutdown.	E-10
Checkpoint Completed: duration was <i>n</i> seconds.	E-10
Checkpoint Page Write Error.	E-10
Checkpoint Record Not Found in Logical Log.	E-11
Chunk <i>chunkname</i> added to space <i>spacename</i>	E-11
Chunk <i>chunkname</i> dropped from space <i>spacename</i>	E-11
Chunk <i>number nn pathname</i> -- Offline.	E-11
Chunk <i>number nn pathname</i> -- Online.	E-11
The chunk <i>pathname</i> must have READ/WRITE permissions for owner and group.	E-12
The chunk <i>pathname</i> must have <i>owner-ID</i> and <i>group-ID</i> set to informix.	E-12
The chunk <i>pathname</i> will not fit in the space specified.	E-12
Cleaning stray LOs in sbspace <i>sbspacename</i>	E-12
Completed re-creating indexes.	E-12
Configuration has been grown to handle up to <i>integer</i> chunks.	E-13
Configuration has been grown to handle up to <i>integer</i> dbslices.	E-13
Configuration has been grown to handle up to <i>integer</i> dbspaces.	E-13
Continuing Long Transaction (for COMMIT): <i>tx Oxn</i>	E-13
Could not disable priority aging: errno = <i>number</i>	E-13
Could not fork a virtual processor: errno = <i>number</i>	E-14

Create_vp: cannot allocate memory.	E-14
Messages: D-E-F	E-14
Dataskip is OFF for all dbspaces.	E-14
Dataskip is ON for all dbspaces..	E-14
Dataskip is ON for dbspaces: <i>dbspacelist</i>	E-14
Dataskip will be turned {ON OFF} for <i>dbspacename</i>	E-15
DBSERVERALIASES exceeded the maximum limit of 32.	E-15
DBSPACETEMP internal list not initialized, using default.	E-15
The DBspace/BLOBspace <i>spacename</i> is now mirrored.	E-15
The DBspace/BLOBspace <i>spacename</i> is no longer mirrored.	E-15
Dbspace <i>dbspacename</i> for Physical-log File not found.	E-16
<i>devname</i> : write failed, file system is full.	E-16
Dropping temporary tblspace <i>0xn</i> , recovering <i>nn</i> pages.	E-16
Dynamically allocated new shared memory segment (size <i>nnnn</i>).	E-16
ERROR: NO "wait for" locks in Critical Section.	E-16
Error building sysmaster database. See <i>outfile</i>	E-17
Error in dropping system defined type.	E-17
Error in renaming systdist.	E-17
Error removing sysdistrib row for <i>tabid = tabid</i> , <i>colid = colid</i> in database <i>name</i> . <i>iserrno = number</i>	E-17
Error writing <i>pathname</i> <i>errno = number</i>	E-17
Error writing shmем to file <i>filename</i> (<i>error</i>). Unable to create output file <i>filename</i> <i>errno=mm</i> .Error writing <i>filename</i> <i>errno=nn</i>	E-18
Fail to extend physical log space.	E-18
Fatal error initializing CWD string. Check permissions on current working directory. Group <i>groupname</i> must have at least execute permission on ' <i>.</i> '.	E-18
The following tables have outstanding old version data pages due to an In-Place Alter Table. Perform UPDATE <i>tablename</i> SET <i>column = column</i> WHERE 1=1; to clear these pages from the following tables.	E-18
Fragments <i>dbspacename1 dbspacename2</i> of table <i>tablename</i> set to non-resident.	E-19
Forced-resident shared memory not available.	E-19
Freed <i>mm</i> shared-memory segment(s) <i>number</i> bytes.	E-19
Messages: G-H-I	E-19
gcore <i>pid</i> ; mv <i>core.pid</i> dir/ <i>core.pid</i> .ABORT.	E-19
I/O <i>function</i> chunk <i>mm</i> , <i>pagenum nn</i> , <i>pagecnt aa</i> --> <i>errno = bb</i>	E-20
I/O error, <i>primary/mirror</i> Chunk <i>pathname</i> -- Offline (<i>sanity</i>).	E-20
Informix <i>database_server</i> Initialized - Complete Disk Initialized.	E-20
Informix <i>database_server</i> Initialized - Shared Memory Initialized.	E-20
Informix <i>database_server</i> Stopped.	E-21
ERROR: Insufficient available disk in the root dbspace to increase the entire Configuration save area.	E-21
Insufficient available disk in the root dbspace for the CM save area. Increase the size of the root dbspace in the ONCONFIG file and reinitialize the server.	E-21
Internal overflow of <i>shmids</i> , increase system max shared memory segment size.	E-22
Messages: J-K-L-M	E-22
Listener-thread err = <i>error_number: error_message</i>	E-22
Lock table overflow - user id <i>mm</i> session id <i>nn</i>	E-22
Logical-log File not found..	E-22
Logical Log <i>nn</i> Complete.	E-22
Logical logging <i>vbrerror</i> for <i>type:subtype</i> in (<i>failed_system</i>)..	E-23
Log Record: log = <i>ll</i> , pos = <i>0xn</i> , type = <i>type:subtype(snum)</i> , trans = <i>xx</i>	E-23
Log record (<i>type:subtype</i>) at log <i>nn</i> , <i>0xn</i> was not undone..	E-23
Log record (<i>type:subtype</i>) failed, partnum <i>pnum</i> row <i>rid</i> <i>iserrno num</i>	E-24
Log record (<i>type:subtype</i>) in log <i>nn</i> , offset <i>0xn</i> was not rolled back.	E-24
Logical Recovery allocating <i>nn</i> worker threads <i>thread_type</i>	E-24
Logical Recovery Started.	E-25
Maximum server connections <i>number</i>	E-25
Memory allocation error.	E-25
Mirror Chunk <i>chunkname</i> added to space <i>spacename</i> . Perform manual recovery.	E-25
Mixed transaction result. (<i>pid=nn</i> user= <i>userid</i>).	E-26
mt_shm_free_pool: pool <i>0xn</i> has blocks still used (id <i>nn</i>).	E-26
mt_shm_init: can't create <i>resident/virtual</i> segment.	E-26
mt_shm_remove: WARNING: may not have removed all/correct segments.	E-26
Messages: N-O-P	E-27

Newly specified value of <i>value</i> for the pagesize in the configuration file does not match older value of <i>value</i> . Using the older value.	E-27
Not enough main memory.	E-27
Not enough logical-log files, Increase LOGFILES.	E-27
Not enough physical procs for affinity.	E-27
The number of configured CPU poll threads exceeds NUMCPUVPS.	E-28
onconfig parameter <i>parameter</i> modified from <i>old_value</i> to <i>new_value</i>	E-28
oninit: Cannot have SINGLE_CPU_VP non-zero and number of CPU VPs greater than 1.	E-28
oninit: Cannot have SINGLE_CPU_VP non-zero and user-defined VP classes.	E-28
oninit: Cannot mix VPCLASS cpu and NUMCPUVPS, AFF_SPROC, AFF_NPROCS, or NOAGE parameters.	E-29
oninit: Cannot mix VPCLASS aio and NUMAIOVPS parameters.	E-29
oninit: Fatal error in initializing ASF with 'ASF_INIT_DATA' flags asfcode = '25507'.	E-29
oninit: invalid or missing name for Subsystem Staging Blobspace.	E-29
Cannot alter a table which has associated violations table.	E-30
oninit: Too many VPCLASS parameters specified.	E-30
oninit: VPCLASS <i>classname</i> bad affinity specification.	E-30
oninit: VPCLASS <i>classname</i> duplicate class <i>name</i>	E-30
oninit: VPCLASS <i>classname</i> illegal option.	E-30
oninit: VPCLASS <i>classname</i> maximum number of VPs is out of the range 0-10000.	E-31
oninit: VPCLASS <i>classname</i> name is too long. Maximum length is <i>maxlength</i>	E-31
oninit: VPCLASS <i>classname</i> number of VPs is greater than the maximum specified.	E-31
oninit: VPCLASS <i>classname</i> number of VPs is out of the range 0-10000.	E-31
onmode: VPCLASS <i>classname</i> name is too long. Maximum length is <i>maxlength</i>	E-32
Optical Subsystem is running.	E-32
Optical Subsystem is not running.	E-32
Optical Subsystem STARTUP Error.	E-32
Online Mode.	E-32
onspaces: unable to reset dataskip.	E-33
Open transaction detected when changing log versions.	E-33
Out of message shared memory.	E-33
Out of resident shared memory.	E-33
Out of virtual shared memory.	E-33
PANIC: Attempting to bring system down.	E-34
Participant site <i>database_server</i> heuristically rolled back.	E-34
Physical recovery complete: <i>number</i> pages examined, <i>number</i> pages restored.	E-34
Physical recovery started at page (<i>chunk:offset</i>).	E-34
Portions of partition partnum of table tablename in database dbname were not logged. This partition cannot be rolled forward.	E-35
Possible mixed transaction result.	E-35
Prepared participant site <i>server_name</i> did not respond.	E-35
Prepared participant site <i>server_name</i> not responding.	E-35
Messages: Q-R-S	E-36
Quiescent Mode.	E-36
Read failed. Table <i>name</i> , Database <i>name</i> , iserrno = <i>number</i>	E-36
Recovery Mode.	E-36
Recreating index: ' <i>dbname:"owner".tablename-idxname</i> '.	E-36
Rollforward of log record failed, iserrno = <i>nn</i>	E-36
Root chunk is full and no additional pages could be allocated to chunk descriptor page.	E-37
scan_logundo: subsys <i>ss</i> , type <i>tt</i> , iserrno <i>ee</i>	E-37
Session completed abnormally. Committing <i>tx</i> id <i>0xm</i> , flags <i>0xn</i>	E-37
Session completed abnormally. Rolling back <i>tx</i> id <i>0xm</i> , flags <i>0xn</i>	E-37
semctl: errno = <i>nn</i>	E-38
semget: errno = <i>nn</i>	E-38
shmat: <i>some_string</i> os_errno: <i>os_err_text</i>	E-38
shmctl: errno = <i>nn</i>	E-38
shmdt: errno = <i>nn</i>	E-38
shmem sent to <i>filename</i>	E-39
shmget: <i>some_str</i> os_errno: key <i>shnkey</i> : <i>some_string</i>	E-39
Shutdown (onmode -k) or override (onmode -O).	E-39
Shutdown Mode.	E-39
Space <i>spacename</i> added.	E-40

Space <i>spacename</i> dropped.	E-40
Space <i>spacename</i> -- Recovery Begins(<i>addr</i>).	E-40
Space <i>spacename</i> -- Recovery Complete(<i>addr</i>).	E-40
Space <i>spacename</i> -- Recovery Failed(<i>addr</i>).	E-41
sysmaster database built successfully.	E-41
Successfully extend physical log space.	E-41
Messages: T-U-V	E-41
This ddl operation is not allowed due to deferred constraints pending on this table and dependent tables.	E-41
This type of space does not accept log files.	E-42
TIMER VP: Could not redirect I/O in initialization, errno = <i>nn</i>	E-42
Too Many Active Transactions.	E-42
Too many violations.	E-42
Transaction Not Found..	E-43
Transaction heuristically rolled back.	E-43
Transaction table overflow - user id <i>nn</i> , process id <i>nn</i>	E-43
Unable to create output file <i>filename</i> errno = <i>nn</i>	E-43
Unable to extend <i>nn</i> reserved pages for <i>purpose</i> in root chunk.	E-43
Unable to initiate communications with the Optical Subsystem.	E-44
Unable to start SQL engine.	E-44
Unable to open tblspace <i>nn</i> , iserrno = <i>nn</i>	E-44
The value of pagesize <i>pagesize</i> specified in the config file is not a valid pagesize. Use 2048, 4096 or 8192 as the value for PAGESIZE in the onconfig file and restart the server.	E-44
Violations table is not started for the target table.	E-45
Violations table reversion test completed successfully.	E-45
Violations table reversion test failed.	E-45
Violations table reversion test start..	E-45
Violations tables still exist..	E-45
Virtual processor limit exceeded.	E-46
VPCLASS <i>classname</i> name is too long. Maximum length is <i>maxlength</i>	E-46
VPCLASS <i>classname</i> duplicate class name.	E-46
VPCLASS <i>classname</i> Not enough physical procs for affinity..	E-46
Messages: W-X-Y-Z	E-47
WARNING: aio_wait: errno = <i>nn</i>	E-47
WARNING: Buffer pool size may cause database server to get into a locked state. Recommended minimum buffer pool size is <i>num</i> times maximum concurrent user threads..	E-47
warning: Chunk time stamps are invalid..	E-47
Warning: <i>name_old</i> is a deprecated onconfig parameter. Use <i>name_new</i> instead. See the release notes and the Informix Administrator's Reference for more information.	E-47
Warning: <i>name_old</i> is a deprecated onconfig parameter. Use <i>name_new</i> instead.	E-48
Warning: Unable to allocate requested big buffer of size <i>nn</i>	E-48
You are turning off smart large object logging.	E-48
Messages: Symbols	E-48
HH:MM:SS Informix database server Version R.VV.PPPPP Software Serial Number RDS#XXXXXXXXX.. . . .	E-48
<i>argument</i> : invalid argument.	E-49
<i>function_name</i> : cannot allocate memory.	E-49
Conversion/Reversion Messages	E-49
Messages: A-C.	E-49
Messages: D-F.	E-52
Messages: I-P	E-54
Messages: R-W	E-58
Conversion and Reversion Messages for Enterprise Replication	E-61
CDR reversion test completed successfully.	E-62
CDR reversion test failed; for details look in \$INFORMIXDIR/etc/revtestcdr.out.	E-62
Enterprise Replication is not ready for conversion. The Control and TRG send queues should be empty for conversion/reversion to proceed.	E-62
Enterprise Replication is not ready for conversion. The syscdr database should NOT contain old-style group definitions for conversion to succeed.	E-63
Enterprise Replication should be in a stopped state for conversion/reversion to proceed..	E-63
Reversion of 'syscdr' failed; for details look in \$INFORMIXDIR/etc/revcdr.out..	E-63
Starting CDR reversion test....	E-63
Starting 'syscdr' conversion....	E-64

Starting 'syscdr' reversion...	E-64
'syscdr' conversion completed successfully.	E-64
'syscdr' conversion failed. For details, look in \$INFORMIXDIR/etc/concdr.out.	E-64
Syscdr should NOT contain new replicate sets for reversion to succeed.	E-65
Syscdr should not contain replicates defined with the --floatieee option for reversion to succeed.	E-65
Dynamic Log Messages.	E-65
Dynamically added log file <i>logid</i> to DBspace <i>dbspace_number</i> .	E-65
Log file <i>logid</i> added to DBspace <i>dbspace_number</i> .	E-65
Log file number <i>logid</i> has been dropped from DBspace <i>dbspace_number</i> ..	E-66
Log file <i>logid</i> has been pre-dropped.	E-66
Pre-dropped log file number <i>logid</i> has been deleted from DBspace <i>dbspace_number</i> ..	E-66
ALERT: Because the oldest logical log (<i>logid</i>) contains records from an open transaction (<i>transaction_address</i>), the server is attempting to dynamically add a log file. But there is no space available. Please add a DBspace or chunk. Then complete the transaction as soon as possible.	E-66
ALERT: The oldest logical log (<i>logid</i>) contains records from an open transaction (<i>transaction_address</i>). Logical logging will remain blocked until a log file is added. Add the log file with the onparams -a command, using the -i (insert) option, as in: onparams -a -d <i>dbspace</i> -s <i>size</i> -i. Then complete the transaction as soon as possible..	E-67
Log file <i>logid</i> has been pre-dropped. It will be deleted from the log list and its space can be reused once you take level-0 archives of all BLOBspaces, Smart BLOBspaces and non-temporary DBspaces.	E-67
Sbspace Metadata Messages	E-67
Allocated <i>number</i> pages to Metadata from chunk <i>number</i> .	E-67
Allocated <i>number</i> pages to Userdata from chunk <i>number</i> .	E-67
Freeing reserved space from chunk <i>number</i> to Metadata..	E-68
Freeing reserved space from chunk <i>number</i> to Userdata..	E-68
Truncate Table Messages	E-68
The table cannot be truncated if it has an open cursor or dirty readers..	E-68
The table cannot be truncated. It has at least one non-empty child table with referential constraints.	E-68

Appendix F. Limits in IBM Informix Dynamic Server F-1

Limitations on UNIX Operating Systems	F-1
System-Level Parameter Limits (UNIX)	F-1
Table-Level Parameter Limits (UNIX)	F-1
Access Capabilities (UNIX)	F-2
IBM Informix Dynamic Server System Defaults (UNIX)	F-2
ON-Monitor Statistics (UNIX)	F-3
Limitations on Windows Operating Systems	F-3
System-Level Parameter Limits (Windows)	F-3
Table-Level Parameter Limits (Windows)	F-3
Access Capabilities (Windows).	F-4
IBM Informix Dynamic Server System Defaults (Windows)	F-4

Appendix G. Accessibility G-1

Accessibility features for IBM Informix Dynamic Server	G-1
Accessibility Features	G-1
Keyboard Navigation	G-1
Related Accessibility Information.	G-1
IBM and Accessibility	G-1
Dotted Decimal Syntax Diagrams	G-1

Notices H-1

Trademarks	H-3
------------	-----

Index X-1

Introduction

In This Introduction

This introduction provides an overview of the information in this publication and describes the conventions it uses.

About This Publication

This publication provides reference material for IBM® Informix® Dynamic Server (IDS). It contains comprehensive descriptions of configuration parameters, the system-monitoring interface (SMI) tables in the **sysmaster** database, the syntax of database server utilities such as **onmode** and **onstat**, logical-log records, disk structures, event alarms, and unnumbered error messages. This publication has two companion volumes, the *IBM Informix Administrator's Guide* and the *IBM Informix Performance Guide*.

This section discusses the intended audience for this publication and the associated software products that you must have to use the administrative utilities.

Types of Users

This publication is written for the following users:

- Database administrators
- System administrators
- Performance engineers

This publication is written with the assumption that you have the following background:

- A working knowledge of your computer, your operating system, and the utilities that your operating system provides
- Some experience working with relational databases or exposure to database concepts
- Some experience with database server administration, operating-system administration, or network administration

If you have limited experience with relational databases, SQL, or your operating system, refer to the *IBM Informix Dynamic Server Getting Started Guide* for your database server for a list of supplementary titles.

Software Dependencies

This publication is written with the assumption that you are using IBM Informix Dynamic Server (IDS) or IBM Informix Dynamic Server with J/Foundation, Version 11.50, as your database server.

Assumptions About Your Locale

IBM Informix products can support many languages, cultures, and code sets. All the information related to character set, collation, and representation of numeric data, currency, date, and time is brought together in a single environment, called a Global Language Support (GLS) locale.

The examples in this publication are written with the assumption that you are using the default locale, **en_us.8859-1**. This locale supports U.S. English format conventions for date, time, and currency. In addition, this locale supports the ISO 8859-1 code set, which includes the ASCII code set plus many 8-bit characters such as é, è, and ñ.

If you plan to use nondefault characters in your data or your SQL identifiers, or if you want to conform to the nondefault collation rules of character data, you need to specify the appropriate nondefault locale.

For instructions on how to specify a nondefault locale, additional syntax, and other considerations related to GLS locales, see the *IBM Informix GLS User's Guide*.

Demonstration Database

The DB–Access utility, which is provided with your Informix database server products, includes one or more of the following demonstration databases:

- The **stores_demo** database illustrates a relational schema with information about a fictitious wholesale sporting-goods distributor. Many examples in IBM Informix publications are based on the **stores_demo** database.
- The **superstores_demo** database illustrates an object-relational schema. The **superstores_demo** database contains examples of extended data types, type and table inheritance, and user-defined routines.

For information about how to create and populate the demonstration databases, see the *IBM Informix DB–Access User's Guide*. For descriptions of the databases and their contents, see the *IBM Informix Guide to SQL: Reference*.

The scripts that you use to install the demonstration databases reside in the **\$INFORMIXDIR/bin** directory on UNIX® and in the **%INFORMIXDIR%\bin** directory on Windows®.

What's New in Administrator's Reference for Dynamic Server, Version 11.50

For a comprehensive list of new features for this release, see the *IBM Informix Dynamic Server Getting Started Guide*. The following changes and enhancements are relevant to this publication.

Table 1. What's New in IBM Informix Dynamic Server Administrator's Reference for Version 11.50.xC3

Overview	Reference
<p>Dynamically Updating the LTXEHWM, LTXHWM, and DYNAMIC_LOGS Configuration Parameters.</p> <p>You can now dynamically update the value of the LTXEHWM, LTXHWM, and DYNAMIC_LOGS configuration parameters by using the onmode -wf or onmode -wm command. The onmode -wf command changes the value in the ONCONFIG file. The onmode -wm command changes the value for the current session.</p>	<p>"onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20, "DYNAMIC_LOGS" on page 1-43, "LTXEHWM" on page 1-62, and "LTXHWM" on page 1-62,</p>

Table 2. What's New in IBM Informix Dynamic Server Administrator's Reference for Version 11.50.xC2

Overview	Reference
<p>Controlling I/O of B-Tree Indexes with Compression Levels</p> <p>B-tree scanners can now compress indexes by merging two partially used index pages if the data on those pages totals a set level (low, medium, or high). You can specify the index compression level by modifying the value of the compression field of the BTSCANNER configuration parameter option, by running an onmode -C compression value command, or by running a SQL API function with a SET INDEX COMPRESSION command.</p>	<p>"BTSCANNER" on page 1-16 and "onmode -C: Control the B-tree scanner" on page 11-4</p>
<p>Limiting the Number of Sessions That Can Connect to Dynamic Server</p> <p>You can now limit the number of sessions that can connect to the server. You do this by setting the LIMITNUMSESSIONS configuration parameter to the maximum number of sessions that you want connected to the database server. Optionally, you can also specify whether you want the server to print messages when the number of sessions approaches a specified maximum number. You can use onmode -wm and onmode -wf commands to turn this configuration parameter on or off or change the value of the configuration parameter.</p>	<p>"LIMITNUMSESSIONS" on page 1-55</p>

Table 3. What's New in IBM Informix Dynamic Server Administrator's Reference for Version 11.50.xC1

Overview	Reference
<p>Configuration parameter to use for Secure Sockets Layer (SSL) support</p> <p>The SSL_KEYSTORE_LABEL configuration parameter specifies the label of the server digital certificate used in the keystore database that stores SSL keys and digital certificates.</p>	<p>"SSL_KEYSTORE_LABEL" on page 1-96</p>
<p>Update Data Support on High-Availability Cluster Secondary Servers</p> <p>You can configure secondary servers so that client applications can send them transactions that update data. You enable secondary server updates with the UPDATABLE_SECONDARY configuration parameter.</p>	<p>"UPDATABLE_SECONDARY" on page 1-81</p>

Table 3. What's New in IBM Informix Dynamic Server Administrator's Reference for Version 11.50.xC1 (continued)

Overview	Reference
<p>Enhanced Connection Management for High-Availability Clusters</p> <p>The new Connection Manager dynamically routes client application connection requests to the most appropriate server in a high-availability cluster. Connection Manager connects to each of the servers in the cluster and gathers statistics about the type of server, unused workload capacity, and the current state of the server. From this information, the Connection Manager redirects the connection to the appropriate server. In addition, Connection Manager Arbitrator provides automatic failover logic for high-availability clusters. Using a configuration file, you specify which secondary server takes over if the primary server fails.</p>	<p>Chapter 16, "The oncmsm Utility," on page 16-1</p>
<p>New utility to encrypt and decrypt password files</p> <p>The onpassword utility is used to encrypt and decrypt password files. Password files are used by Enterprise Replication and Connection Manager.</p>	<p>Chapter 17, "The onpassword Utility," on page 17-1</p>
<p>Improved onconfig.std file and new default values for configuration parameters</p> <p>The onconfig.std file is easier to read because the comments and the parameters are listed separately and grouped by functional areas. Most supported configuration parameters are now included in the file. Deprecated configuration parameters were removed. Some configuration parameters that specify sizes now have higher values. Some configuration parameters that specify file locations now have more secure default locations under the \$INFORMIXDIR directory.</p>	<p>Chapter 1, "Configuration Parameters," on page 1-1</p>
<p>Enhanced shared-memory dump file size control</p> <p>By using the new options for the DUMPSHMEM configuration parameter and the onstat utility, you can control how much memory is written to a dump file. These options exclude the buffer pool in the resident memory, which can result in a much smaller file.</p>	<p>"DUMPSHMEM (UNIX)" on page 1-42</p> <p>"onstat -o: Output shared memory contents to a file" on page 15-146</p>
<p>Enhanced startup script customization</p> <p>You can customize startup scripts and automate startup with the new -w option for the oninit utility. The -w option forces the server to wait until it successfully initializes before returning a shell prompt.</p>	<p>"Initializing the Server in Wait Mode with the -w Option" on page 9-2</p>
<p>Viewing Data Server client session information</p> <p>A new sysmaster database table and new onstat -g ses fields display Data Server client session ID, session application name, and a session value.</p>	<p>"sysseappinfo" on page 2-29</p> <p>"onstat -g ses: Print session-related information" on page 15-113</p>
<p>Alarms for full storage spaces</p> <p>The new STORAGE_FULL_ALARM configuration parameter sets the threshold level and time interval for alarms that are raised when storage spaces or partitions become full.</p>	<p>"STORAGE_FULL_ALARM" on page 1-100</p>

Table 3. What's New in IBM Informix Dynamic Server Administrator's Reference for Version 11.50.xC1 (continued)

Overview	Reference
Dates for online log messagesThe new MSG_DATE configuration parameter inserts a date at the beginning of messages printed to the online log.	"MSG_DATE" on page 1-67
The new HA_ALIAS configuration parameter represents the name by which the server is known within a high-availability cluster.	"HA_ALIAS" on page 1-52

Documentation Conventions

This section describes the following conventions, which are used in the product documentation for IBM Informix Dynamic Server:

- Typographical conventions
- Feature, product, and platform conventions
- Syntax diagrams
- Command-line conventions
- Example code conventions

Typographical Conventions

This publication uses the following conventions to introduce new terms, illustrate screen displays, describe command syntax, and so forth.

Convention	Meaning
KEYWORD	Keywords of SQL, SPL, and some other programming languages appear in uppercase letters in a serif font.
<i>italics</i>	Within text, new terms and emphasized words appear in italics. Within syntax and code examples, variable values that you are to specify appear in italics.
boldface	Names of program entities (such as classes, events, and tables), environment variables, file names, path names, and interface elements (such as icons, menu items, and buttons) appear in boldface.
monospace	Information that the product displays and information that you enter appear in a monospace typeface.
KEYSTROKE	Keys that you are to press appear in uppercase letters in a sans serif font.
>	This symbol indicates a menu item. For example, "Choose Tools > Options " means choose the Options item from the Tools menu.

Technical changes to the text are indicated by special characters depending on the format of the documentation:

HTML documentation

New or changed information is surrounded by blue \gg and \ll characters.

PDF documentation

A plus sign (+) is shown to the left of the current changes. A vertical bar (|) is shown to the left of changes made in earlier shipments.

Feature, Product, and Platform Markup

Feature, product, and platform markup identifies paragraphs that contain feature-specific, product-specific, or platform-specific information. Some examples of this markup follow:

Dynamic Server only: Identifies information that is specific to the Windows operating system

Windows only: Identifies information that is specific to the Windows operating system

This markup can apply to one or more paragraphs within a section. When an entire section applies to a particular product or platform, this is noted as part of the heading text, for example:

Table Sorting (Windows)

Example Code Conventions

Examples of SQL code occur throughout this publication. Except as noted, the code is not specific to any single IBM Informix application development tool.

If only SQL statements are listed in the example, they are not delimited by semicolons. For instance, you might see the code in the following example:

```
CONNECT TO stores_demo
...

DELETE FROM customer
  WHERE customer_num = 121
...

COMMIT WORK
DISCONNECT CURRENT
```

To use this SQL code for a specific product, you must apply the syntax rules for that product. For example, if you are using an SQL API, you must use EXEC SQL at the start of each statement and a semicolon (or other appropriate delimiter) at the end of the statement. If you are using DB–Access, you must delimit multiple statements with semicolons.

Tip: Ellipsis points in a code example indicate that more code would be added in a full application, but it is not necessary to show it to describe the concept being discussed.

For detailed directions on using SQL statements for a particular application development tool or SQL API, see the documentation for your product.

Additional Documentation

You can view, search, and print all of the product documentation from the IBM Informix Dynamic Server information center on the Web at <http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp>.

For additional documentation about IBM Informix Dynamic Server and related products, including release notes, machine notes, and documentation notes, go to the online product library page at <http://www.ibm.com/software/data/informix/pubs/library/>. Alternatively, you can access or install the product documentation from the Quick Start CD that is shipped with the product.

Compliance with Industry Standards

The American National Standards Institute (ANSI) and the International Organization of Standardization (ISO) have jointly established a set of industry standards for the Structured Query Language (SQL). IBM Informix SQL-based products are fully compliant with SQL-92 Entry Level (published as ANSI X3.135-1992), which is identical to ISO 9075:1992. In addition, many features of IBM Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL Common Applications Environment (CAE) standards.

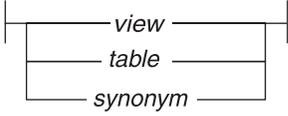
Syntax Diagrams

This guide uses syntax diagrams built with the following components to describe the syntax for statements and all commands other than system-level commands.

Table 4. Syntax Diagram Components

Component represented in PDF	Component represented in HTML	Meaning
	>>-----	Statement begins.
	----->	Statement continues on next line.
	>-----	Statement continues from previous line.
	-----><	Statement ends.
	-----SELECT-----	Required item.
	--+-----+-- '-----LOCAL-----'	Optional item.
	---+-----ALL-----+--- +---DISTINCT-----+ '---UNIQUE-----'	Required item with choice. One and only one item must be present.
	---+-----+--- +---FOR UPDATE-----+ '---FOR READ ONLY---'	Optional items with choice are shown below the main line, one of which you might specify.
	.---NEXT-----. ---+-----+--- +---PRIOR-----+ '---PREVIOUS-----'	The values below the main line are optional, one of which you might specify. If you do not specify an item, the value above the line will be used as the default.
	.-----,-----. v ---+-----+--- +---index_name-----+ '---table_name-----'	Optional items. Several items are allowed; a comma must precede each repetition.
	>>-- Table Reference --<<	Reference to a syntax segment.

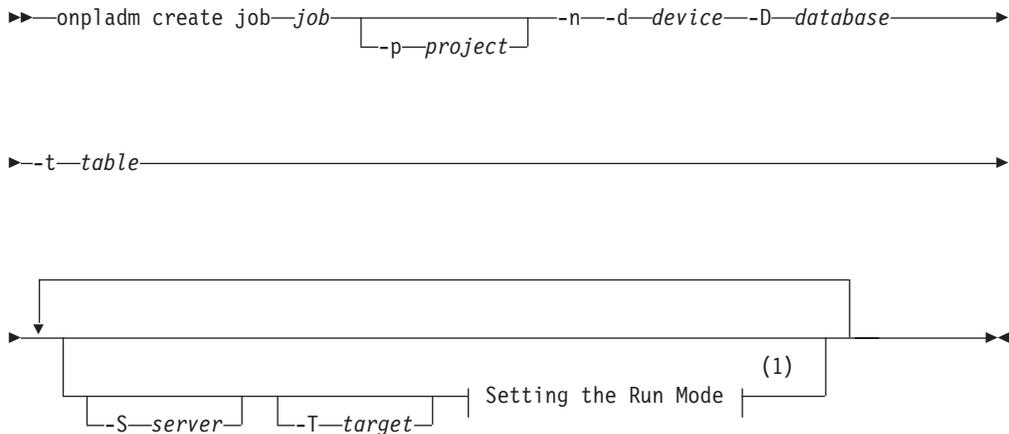
Table 4. Syntax Diagram Components (continued)

Component represented in PDF	Component represented in HTML	Meaning
<p>Table Reference</p> 	<p>Table Reference</p> <pre> ---+-----view-----+--- +-----table-----+ '-----synonym-----' </pre>	Syntax segment.

How to Read a Command-Line Syntax Diagram

The following command-line syntax diagram uses some of the elements listed in the table in Syntax Diagrams.

Creating a No-Conversion Job

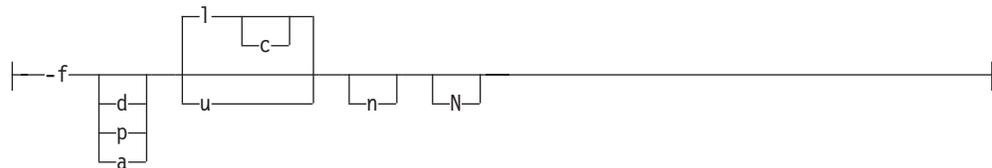


Notes:

- 1 See page Z-1

The second line in this diagram has a segment named “Setting the Run Mode,” which according to the diagram footnote, is on page Z-1. If this was an actual cross-reference, you would find this segment in on the first page of Appendix Z. Instead, this segment is shown in the following segment diagram. Notice that the diagram uses segment start and end components.

Setting the Run Mode:



To see how to construct a command correctly, start at the top left of the main diagram. Follow the diagram to the right, including the elements that you want. The elements in this diagram are case sensitive because they illustrate utility syntax. Other types of syntax, such as SQL, are not case sensitive.

The Creating a No-Conversion Job diagram illustrates the following steps:

1. Type **onpladm create job** and then the name of the job.
2. Optionally, type **-p** and then the name of the project.
3. Type the following required elements:
 - **-n**
 - **-d** and the name of the device
 - **-D** and the name of the database
 - **-t** and the name of the table
4. Optionally, you can choose one or more of the following elements and repeat them an arbitrary number of times:
 - **-S** and the server name
 - **-T** and the target server name
 - The run mode. To set the run mode, follow the Setting the Run Mode segment diagram to type **-f**, optionally type **d**, **p**, or **a**, and then optionally type **l** or **u**.
5. Follow the diagram to the terminator.

Keywords and Punctuation

Keywords are words reserved for statements and all commands except system-level commands. When a keyword appears in a syntax diagram, it is shown in uppercase letters. When you use a keyword in a command, you can write it in uppercase or lowercase letters, but you must spell the keyword exactly as it appears in the syntax diagram.

You must also use any punctuation in your statements and commands exactly as shown in the syntax diagrams.

Identifiers and Names

Variables serve as placeholders for identifiers and names in the syntax diagrams and examples. You can replace a variable with an arbitrary name, identifier, or literal, depending on the context. Variables are also used to represent complex syntax elements that are expanded in additional syntax diagrams. When a variable appears in a syntax diagram, an example, or text, it is shown in *lowercase italic*.

The following syntax diagram uses variables to illustrate the general form of a simple SELECT statement.

►►—SELECT—*column_name*—FROM—*table_name*—►►

When you write a SELECT statement of this form, you replace the variables *column_name* and *table_name* with the name of a specific column and table.

How to Provide Documentation Feedback

You are encouraged to send your comments about IBM Informix user documentation by using one of the following methods:

- Send e-mail to docinf@us.ibm.com.

- Go to the Information Center at <http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp> and open the topic that you want to comment on. Click the feedback link at the bottom of the page, fill out the form, and submit your feedback.

Feedback from both methods is monitored by those who maintain the user documentation of Dynamic Server. The feedback methods are reserved for reporting errors and omissions in our documentation. For immediate help with a technical problem, contact IBM Technical Support. For instructions, see the IBM Informix Technical Support Web site at <http://www.ibm.com/planetwide/>.

We appreciate your suggestions.

Part 1. Configuring and Monitoring Dynamic Server

Chapter 1. Configuration Parameters

In This Chapter

This chapter describes the **ONCONFIG** file conventions, lists the configuration parameters in the **ONCONFIG** file, and provides a short discussion of each parameter.

ONCONFIG File Conventions

The **ONCONFIG** environment variable specifies the file that contains the configuration parameters. This file is also called the *ONCONFIG file*. The database server uses the **ONCONFIG** file during initialization.

Format of ONCONFIG File

In the **ONCONFIG** file, each parameter is on a separate line. The file can also contain blank lines and comment lines that start with a **#** symbol. The following line shows the syntax for a parameter line:

```
PARAMETER_NAME    parameter_value
```

Descriptions of parameters and their possible values are surrounded by comment symbols.

Parameters and their values in the **ONCONFIG** file are case sensitive. The parameter names are always uppercase. If the value entry is described with uppercase letters, you must use uppercase (for example, the CPU value of the **NETTYPE** parameter). You must put white space (tabs, spaces, or both) between the parameter name, parameter value, and optional comment. Do not use any tabs or spaces within a parameter value.

Tip:

If you use a utility like **grep** on the **onconfig.std**, include a new-line character when you search for configuration parameters names so that you do not return the parameter description. For example, the following command returns both the description and the configuration parameter:

```
grep "MSGPATH" onconfig.std
# MSGPATH      - The name of the IDS message log file
MSGPATH $INFORMIXDIR/tmp/online.log
```

Whereas, this command returns only the configuration parameter:

```
grep "^MSGPATH" onconfig.std
MSGPATH $INFORMIXDIR/tmp/online.log
```

Restriction: The maximum line limit of the **ONCONFIG** file is 512 bytes. Lines that exceed this limit are truncated and might cause configuration problems.

ONCONFIG File Templates

The database server provides a template for a configuration file that contains initial values for many of the **ONCONFIG** parameters.

IBM Informix Dynamic Server (IDS) provides **onconfig.std** as a template configuration file that you can copy and tailor to your specific configuration.

If you omit a parameter value in your copy of the configuration file, the database server either uses default values in **onconfig.std** or calculates values based on other parameter values. For information on the order of files in which the database server looks for configuration values during initialization, refer to the chapter on initializing the database server in the *IBM Informix Administrator's Guide*.

Warning: Do not modify or delete **onconfig.std**, which is a template and not a functional configuration.

The following table lists the locations of the **ONCONFIG** and **onconfig.std** files.

Operating System	ONCONFIG File	Template File
UNIX	<code>\$INFORMIXDIR/etc/\$ONCONFIG</code>	<code>\$INFORMIXDIR/etc/onconfig.std</code>
Windows	<code>%INFORMIXDIR%\etc\%ONCONFIG%</code>	<code>%INFORMIXDIR%\etc\onconfig.std</code>

To prepare the ONCONFIG file:

1. Copy the **onconfig.std** template file.
2. Modify the *copy* of the template file.
3. Set the **ONCONFIG** environment variable to the name of the copy of the pertinent template file. If you do not set **ONCONFIG**, the default filename is **onconfig**.

For more details on why you might want to modify the default configuration parameters, refer to the chapter on configuring the database server in the *IBM Informix Administrator's Guide*.

Important: Print out a copy of the **onconfig.std** file to see the latest default values for the configuration parameters and recommended settings.

Displaying ONCONFIG Settings

When the database server restarts, it reads the ONCONFIG file. To view the ONCONFIG settings, use one of the following tools:

- A text editor
- IBM Informix Server Administrator (ISA)
- **oncheck -pr**

The information under `PAGE_CONFIG` lists the configuration parameter settings at restart. For more information, see “oncheck -pr and pR: Display reserved-page information” on page 7-17.

- **.infos.dbservername**

If you set the **ONCONFIG** environment variable to the name of a different ONCONFIG file while the database server is online, the **.infos.dbservername** file contains the current settings. For more information, see “.infos.dbservername” on page A-6 and “The ONCONFIG File” on page A-8.

For more information about the **ONCONFIG** environment variable, see the *IBM Informix Guide to SQL: Reference*.

Summary of Configuration Parameters

The following table contains a list of configuration parameters. The list also shows any related environment variables. For information on the discontinued configuration parameters, see Appendix D, “Discontinued Configuration Parameters,” on page D-1.

Configuration Parameter	Related Environment Variable	Reference
AC_DEBUG	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_IXBAR	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_LTAPEBLOCK	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_LTAPEDEV	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_MSGPATH	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_SCHEMA	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_STORAGE	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_TAPEBLOCK	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_TAPEDEV	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_TIMEOUT	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
AC_VERBOSE	AC_CONFIG	“Archecker Configuration Parameters” on page 1-10
ADMIN_MODE_USERS		“ADMIN_MODE_USERS” on page 1-11
ADMIN_USER_MODE_WITH_DBSA		“ADMIN_USER_MODE_WITH_DBSA” on page 1-12
ADTERR		“ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)” on page 1-12
ADTMODE		“ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)” on page 1-12
ADTPATH		“ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)” on page 1-12
ADTSIZE		“ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)” on page 1-12
AFCRASH		“Java Configuration Parameters” on page 1-54
ALARMPROGRAM		“ALARMPROGRAM” on page 1-12
ALLOW_NEWLINE		“ALLOW_NEWLINE” on page 1-13
ALRM_ALL_EVENTS		“ALRM_ALL_EVENTS” on page 1-13
AUTO_AIOVPS		“AUTO_AIOVPS” on page 1-14
AUTO_CKPTS		“AUTO_CKPTS” on page 1-14
AUTO_LRU_TUNING		“AUTO_LRU_TUNING” on page 1-15

Configuration Parameter	Related Environment Variable	Reference
AUTO_REPREPARE		" AUTO_REPREPARE" on page 1-15
BAR_ACT_LOG		" ON-Bar Configuration Parameters" on page 1-73
BAR_BSALIB_PATH		" ON-Bar Configuration Parameters" on page 1-73
BAR_DEBUG		" ON-Bar Configuration Parameters" on page 1-73
BAR_DEBUG_LOG		" ON-Bar Configuration Parameters" on page 1-73
BAR_HISTORY		" ON-Bar Configuration Parameters" on page 1-73
BAR_MAX_BACKUP		" ON-Bar Configuration Parameters" on page 1-73
BAR_NB_XPORT_COUNT		" ON-Bar Configuration Parameters" on page 1-73
BAR_PERFORMANCE		" ON-Bar Configuration Parameters" on page 1-73
BAR_PROGRESS_FREQ		" ON-Bar Configuration Parameters" on page 1-73
BAR_RETRY		" ON-Bar Configuration Parameters" on page 1-73
BAR_XFER_BUF_SIZE		" ON-Bar Configuration Parameters" on page 1-73
BLOCKTIMEOUT		"BLOCKTIMEOUT" on page 1-16
BTSCANNER		"BTSCANNER" on page 1-16
BUFFERPOOL		"BUFFERPOOL" on page 1-17
CDR_DBSPACE		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_DSLOCKWAIT		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_ENV		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_EVALTHREADS		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_MAX_DYNAMIC_LOGS		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_NIFCOMPRESS		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_QDATA_SBSpace		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_QHDR_DBSPACE		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_QUEUEMEM		"Enterprise Replication Configuration Parameters" on page 1-48
CDR_SERIAL		"Enterprise Replication Configuration Parameters" on page 1-48

Configuration Parameter	Related Environment Variable	Reference
CDR_SUPPRESS_ATSRISWARN		"Enterprise Replication Configuration Parameters" on page 1-48
CKPTINTVL		"CKPTINTVL" on page 1-21
CLEANERS		"CLEANERS" on page 1-22
CONSOLE		"CONSOLE" on page 1-22
DATASKIP		" DATASKIP" on page 1-22
DB_LIBRARY_PATH		" DB_LIBRARY_PATH" on page 1-24
DBSERVERALIASES		"DBSERVERALIASES" on page 1-24
DBSERVERNAME	INFORMIXSERVER	"DBSERVERNAME" on page 1-25
DBSPACETEMP	DBSPACETEMP	"DBSPACETEMP" on page 1-26
DD_HASHMAX		" DD_HASHMAX" on page 1-28
DD_HASHSIZE		" DD_HASHSIZE" on page 1-29
DEADLOCK_TIMEOUT		"DEADLOCK_TIMEOUT" on page 1-29
DEF_TABLE_LOCKMODE	IFX_DEF_TABLE_LOCKMODE	" DEF_TABLE_LOCKMODE" on page 1-30
DIRECT_IO		" DIRECT_IO (UNIX)" on page 1-31
DIRECTIVES	IFX_DIRECTIVES	" DIRECTIVES" on page 1-31
DISABLE_B162428_XA_FIX	IFX_XASTDCOMPLIANCE_XAEND	" DISABLE_B162428_XA_FIX" on page 1-32
DRAUTO		" DRAUTO" on page 1-33
DRDA_COMMBUFFSIZE		" DRDA_COMMBUFFSIZE" on page 1-32
DRIDXAUTO		" DRIDXAUTO" on page 1-34
DRINTERVAL		"DRINTERVAL" on page 1-34
DRLOSTFOUND		"DRLOSTFOUND" on page 1-35
DRTIMEOUT		"DRTIMEOUT" on page 1-35
DS_HASHSIZE		"DS_HASHSIZE" on page 1-36
DS_MAX_QUERIES		"DS_MAX_QUERIES" on page 1-36
DS_MAX_SCANS		"DS_MAX_SCANS" on page 1-37
DS_NONPDQ_QUERY_MEM		"DS_NONPDQ_QUERY_MEM " on page 1-38
DS_POOLSIZE		" DS_POOLSIZE" on page 1-38
DS_TOTAL_MEMORY		" DS_TOTAL_MEMORY" on page 1-39
DUMPCNT		"DUMPCNT (UNIX)" on page 1-40
DUMPCORE		"DUMPCORE (UNIX)" on page 1-41
DUMPDIR		"DUMPDIR" on page 1-41
DUMPGCORE		"DUMPGCORE (UNIX)" on page 1-41
DUMPSHMEM		"DUMPSHMEM (UNIX)" on page 1-42
DYNAMIC_LOGS		" DYNAMIC_LOGS" on page 1-43
ENCRYPT_CDR		"Enterprise Replication Configuration Parameters" on page 1-48

Configuration Parameter	Related Environment Variable	Reference
ENCRYPT_CIPHERS		"ENCRYPT_CIPHERS" on page 1-44 "Enterprise Replication Configuration Parameters" on page 1-48
ENCRYPT_HDR		" ENCRYPT_HDR" on page 1-45
ENCRYPT_MAC		"ENCRYPT_MAC" on page 1-46 "Enterprise Replication Configuration Parameters" on page 1-48
ENCRYPT_MACFILE		"ENCRYPT_MACFILE" on page 1-46 "Enterprise Replication Configuration Parameters" on page 1-48
ENCRYPT_SMX		"ENCRYPT_SMX" on page 1-47
ENCRYPT_SWITCH		"ENCRYPT_SWITCH" on page 1-48 "Enterprise Replication Configuration Parameters" on page 1-48
EXPLAIN_STAT		" EXPLAIN_STAT" on page 1-50
EXT_DIRECTIVES	IFX_EXTDIRECTIVES	" EXT_DIRECTIVES" on page 1-50
EXTSHMADD		" EXTSHMADD" on page 1-51
FAILOVER_CALLBACK		"FAILOVER_CALLBACK" on page 1-51
FASTPOLL		"FASTPOLL" on page 1-51
FILLFACTOR		"FILLFACTOR" on page 1-52
HA_ALIAS		"HA_ALIAS" on page 1-52
HETERO_COMMIT		"HETERO_COMMIT" on page 1-52
IFX_EXTEND_ROLE		"IFX_EXTEND_ROLE" on page 1-53
IMCLOG		"MaxConnect Configuration Parameters" on page 1-65
IMCTRANSPTS		"MaxConnect Configuration Parameters" on page 1-65
IMCWORKERDELAY		"MaxConnect Configuration Parameters" on page 1-65
IMCWORKERTHREADS		"MaxConnect Configuration Parameters" on page 1-65
ISM_DATA_POOL		" ISM_DATA_POOL and ISM_LOG_POOL" on page 1-54
ISM_LOG_POOL		" ISM_DATA_POOL and ISM_LOG_POOL" on page 1-54
JVMTHREAD		" Java Configuration Parameters" on page 1-54
JVPCLASSPATH		" Java Configuration Parameters" on page 1-54
JVPDEBUG		" Java Configuration Parameters" on page 1-54
JVPHOME		" Java Configuration Parameters" on page 1-54

Configuration Parameter	Related Environment Variable	Reference
JVPJAVAHOME		"Java Configuration Parameters" on page 1-54
JVPJAVALIB		"Java Configuration Parameters" on page 1-54
JVPJAVAVM		"Java Configuration Parameters" on page 1-54
JVPLOGFILE		"Java Configuration Parameters" on page 1-54
JVPPROFILE		"Java Configuration Parameters" on page 1-54
LIMITNUMSESSIONS		"LIMITNUMSESSIONS" on page 1-55
LISTEN_TIMEOUT		"LISTEN_TIMEOUT" on page 1-56
LOCKS		"LOCKS" on page 1-56
LOGBUFF		"LOGBUFF" on page 1-57
LOGFILES		"LOGFILES" on page 1-58
LOG_INDEX_BUILDS		"LOG_INDEX_BUILDS" on page 1-58
LOGSIZE		"LOGSIZE" on page 1-59
LTAPEBLK		"LTAPEBLK" on page 1-60
LTAPEDEV		"LTAPEDEV" on page 1-61
LTAPESIZE		"LTAPESIZE" on page 1-61
LTXEHWM		"LTXEHWM" on page 1-62
LTXHWM		"LTXHWM" on page 1-62
MAX_FILL_DATA_PAGES		"MAX_FILL_DATA_PAGES" on page 1-63
MAX_INCOMPLETE_CONNECTIONS		"MAX_INCOMPLETE_CONNECTIONS" on page 1-64
MAX_PDQPRIORITY		"MAX_PDQPRIORITY" on page 1-65
MIRROR		"MIRROR" on page 1-66
MIRROROFFSET		"MIRROROFFSET" on page 1-67
MIRRORPATH		"MIRRORPATH" on page 1-67
MSG_DATE		"MSG_DATE" on page 1-67
MSGPATH		"MSGPATH" on page 1-68
MULTIPROCESSOR		"MULTIPROCESSOR" on page 1-68
NETTYPE		"NETTYPE" on page 1-69
OFF_RECVRY_THREADS		"OFF_RECVRY_THREADS" on page 1-72
ON_RECVRY_THREADS		"ON_RECVRY_THREADS" on page 1-72
ONDBSPACEDOWN		"ONDBSPACEDOWN" on page 1-74
ONLIDX_MAXMEM		"ONLIDX_MAXMEM" on page 1-74
OPCACHEMAX	INFORMIXOPCACHE	"OPCACHEMAX (UNIX)" on page 1-75
OPTCOMPIND	OPTCOMPIND	"OPTCOMPIND" on page 1-76
OPT_GOAL	OPT_GOAL	"OPT_GOAL" on page 1-76
PC_HASHSIZE		"PC_HASHSIZE" on page 1-77
PC_POOLSIZE		"PC_POOLSIZE" on page 1-78

Configuration Parameter	Related Environment Variable	Reference
PHYSBUFF		"PHYSBUFF" on page 1-78
PHYSFILE		"PHYSFILE" on page 1-79
QSTATS		"QSTATS" on page 1-80
RA_PAGES		"RA_PAGES" on page 1-80
RA_THRESHOLD		"RA_THRESHOLD" on page 1-80
REDIRECTED_WRITES		"UPDATABLE_SECONDARY" on page 1-81
RESIDENT		"RESIDENT" on page 1-82
RESTARTABLE_RESTORE		"RESTARTABLE_RESTORE" on page 1-82
ROOTNAME		"ROOTNAME" on page 1-83
ROOTOFFSET		"ROOTOFFSET" on page 1-84
ROOTPATH		"ROOTPATH" on page 1-84
ROOTSIZE		"ROOTSIZE" on page 1-85
RTO_SERVER_RESTART		"RTO_SERVER_RESTART" on page 1-85
SBSPACENAME		"SBSPACENAME" on page 1-86
SBSPACETEMP		"SBSPACETEMP" on page 1-87
SDS_ENABLE		"SDS_ENABLE" on page 1-87
SDS_PAGING		"SDS_PAGING" on page 1-88
SDS_TEMPDBS		"SDS_TEMPDBS" on page 1-88
SDS_TIMEOUT		"SDS_TIMEOUT" on page 1-89
SECURITY_LOCALCONNECTION		"SECURITY_LOCALCONNECTION" on page 1-90
SERVERNUM		"SERVERNUM" on page 1-90
SHMADD		"SHMADD" on page 1-90
SHMBASE		"SHMBASE" on page 1-91
SHMNOACCESS		"SHMNOACCESS" on page 1-92
SHMTOTAL		"SHMTOTAL" on page 1-92
SHMVIRT_ALLOCSEG		"SHMVIRT_ALLOCSEG" on page 1-93
SHMVIRT_SIZE		"SHMVIRT_SIZE" on page 1-93
SINGLE_CPU_VP		"SINGLE_CPU_VP" on page 1-94
SQLTRACE		"SQLTRACE" on page 1-95
SSL_KEYSTORE_LABEL		"SSL_KEYSTORE_LABEL" on page 1-96
STACKSIZE	INFORMIXSTACKSIZE	"STACKSIZE" on page 1-96
STAGEBLOB		"STAGEBLOB" on page 1-97
STMT_CACHE	STMT_CACHE	"STMT_CACHE" on page 1-97
STMT_CACHE_HITS		"STMT_CACHE_HITS" on page 1-98
STMT_CACHE_NOLIMIT		"STMT_CACHE_NOLIMIT" on page 1-99
STMT_CACHE_NUMPOOL		"STMT_CACHE_NUMPOOL" on page 1-99
STMT_CACHE_SIZE		"STMT_CACHE_SIZE" on page 1-100
STORAGE_FULL_ALARM		"STORAGE_FULL_ALARM" on page 1-100
SYSALARMPROGRAM		"SYSALARMPROGRAM" on page 1-101

Configuration Parameter	Related Environment Variable	Reference
SYSSBSPACENAME		"SYSSBSPACENAME" on page 1-101
TAPEBLK		" TAPEBLK" on page 1-103
TAPEDEV		" TAPEDEV" on page 1-103
TAPESIZE		" TAPESIZE" on page 1-104
TBLSPACE_STATS		" TBLSPACE_STATS" on page 1-105
TBLTBLFIRST		" TBLTBLFIRST" on page 1-105
TBLTBLNEXT		" TBLTBLNEXT" on page 1-105
TEMPTAB_NOLOG		" TEMPTAB_NOLOG" on page 1-106
TXTIMEOUT		"TXTIMEOUT" on page 1-106
USELASTCOMMITTED		"USELASTCOMMITTED" on page 1-107
USEOSTIME		"USEOSTIME" on page 1-108
UNSECURE_ONSTAT		"UNSECURE_ONSTAT" on page 1-107
VP_MEMORY_CACHE_KB		"VP_MEMORY_CACHE_KB" on page 1-109
VPCLASS		"VPCLASS" on page 1-109
WSTATS		"WSTATS" on page 1-114

Parameter Attributes

This topic describes one or more of the following attributes (if relevant) for each parameter.

Attribute

Description

onconfig.std value

The default value that appears in the **onconfig.std** file. The database server uses these default values for all configurations.

if not present

The value that the database server supplies if the parameter is missing from your ONCONFIG file. If this value is present in **onconfig.std**, the database server uses the **onconfig.std** value. If this value is not present in **onconfig.std**, the database server calculates the value based on other values in **onconfig.std**.

units The units in which the parameter is expressed

separators

The separators that can be used when the parameter value has several parts. Do *not* use white space within a parameter value.

range of values

The valid values for this parameter

takes effect

The time at which a change to the value of the parameter affects the operation of the database server. *Disk is initialized* means to reinitialize the database server.

utilities

The database server utilities that you can use to change the value of the parameter

refer to Cross-reference to further discussion

Using a Utility to Change a Parameter Value

Use one of these utilities to change the value of a configuration parameter. The *utilities* section for each configuration parameter lists the specific utilities to use.

Tool Description

ON-Monitor(UNIX)

You can use ON-Monitor to change certain parameter values. In ON-Monitor, some of the responses are Y/N (yes/no). When those responses are recorded in the ONCONFIG file, Y becomes 1, and N becomes 0.

ISA To use IBM Informix Server Administrator (ISA) to change parameter values, select **Configuration > ONCONFIG**.

Command-line utility

The *utilities* section lists one or more command-line utilities that you can use to change a parameter value.

Text editor

You can use a text editor to modify the ONCONFIG file.

Environment Variables

If you set the environment variable on the database server, it applies to all sessions. If you set the environment variable in the client environment, it applies to the current session and overrides the equivalent configuration parameter (if any). For a complete list of environment variables, and how to set them, see the *IBM Informix Guide to SQL: Reference*.

Note: The INFORMIXDIR environment variable must always be set.

Archecker Configuration Parameters

The **ac_config.std** template contains the default **archecker** configuration parameters. Usually, you would not change these parameters. However, if you need to change these parameters, copy the **ac_config.std** template to the AC_CONFIG file. (The **AC_CONFIG** environment variable specifies the location of the AC_CONFIG file.) The **archecker** utility uses these parameters when it verifies a backup or performs a table-level restore. For information on these parameters, see the *IBM Informix Backup and Restore Guide*.

Configuration Parameter

Description

AC_DEBUG

Prints debugging messages in the **archecker** message log.

AC_IXBAR

Specifies the pathname to the IXBAR file.

AC_LTAPEBLOCK

Specifies the **ontape** block size for reading logical logs.

AC_LTAPEDEV

Specifies the local device name used by **ontape** for reading logical logs.

AC_MSGPATH

Specifies the location of the **archecker** message file.

AC_SCHEMA

Specifies the pathname to the **archecker schema** command.

AC_STORAGE

Specifies the location of the temporary files that **archecker** builds.

AC_TAPEBLOCK

Specifies the tape block size in kilobytes.

AC_TAPEDEV

Specifies the device name used by the **ontape** utility.

AC_TIMEOUT

Specifies the timeout value for ON-Bar and **archecker** processes if one of them exits prematurely.

AC_VERBOSE

Specifies either verbose or quiet mode for **archecker** messages.

ADMIN_MODE_USERS

onconfig.std *value*

None

range of values

comma-separated user names, such as: Karin,Sarah,Andrew, up to a string of 127 characters

takes effect

When the database server is shut down and restarted

utilities

oninit -U, **onmode -j -U**, **onmode -wm**, and **onmode -wf**

refer to

- “Initialize Shared Memory Only” on page 9-3
- “Changing the Database Server to Administration Mode with the -j Option” on page 11-14
- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- “ ADMIN_USER_MODE_WITH_DBSA” on page 1-12

ADMIN_MODE_USERS specifies which users (in addition to user **informix** and DBSA group users) can access the database server in administration mode. The list of users is preserved indefinitely, and any or all of the users can be removed by using **onmode -wm** or **onmode -wf** .

To enable access to the database server in administration mode for the users specified in ADMIN_MODE_USERS, the ADMIN_USER_MODE_WITH_DBSA configuration parameter needs to be set to 1.

Use **onmode -j -U** to allow any or more users to access the database server in administration mode when the database is running.

ADMIN_USER_MODE_WITH_DBSA

onconfig.std *value*
None

if not present
0

takes effect
When the database server is shut down and restarted

range of values
0 = only user **informix** can connect 1 = user **informix**, users who have the **informix** group included in their group list (UNIX), DBSA group users, and administration users listed in ADMIN_MODE_USERS can connect

refer to “ ADMIN_MODE_USERS” on page 1-11

ADMIN_USER_MODE_WITH_DBSA specifies whether the users specified in ADMIN_MODE_USERS (in addition to user **informix** and DBSA group users) can connect to the database server while it is in administration mode.

ADTERR, ADTMODE, ADTPATH, and ADTSIZE (UNIX)

ADTERR, ADTMODE, ADTPATH, and ADTSIZE are configuration parameters for auditing. For information on these parameters, see the *IBM Informix Security Guide*.

ALARMPROGRAM

onconfig.std *value*
On UNIX: **\$INFORMIXDIR/etc/alarmprogram.sh** On Windows:
\$INFORMIXDIR\etc\alarmprogram.bat

if not present
On UNIX: **\$INFORMIXDIR/etc/alarmprogram.sh** On Windows:
\$INFORMIXDIR\etc\alarmprogram.bat

range of values
Full pathname

takes effect
When the database server is shut down and restarted

refer to

- “Writing Your Own Alarm Script” on page C-1
- *IBM Informix Backup and Restore Guide*

Use the ALARMPROGRAM parameter to display event alarms. The following sample scripts are provided.

Script Name	Platform	Description
log_full.sh	UNIX	To back up logical logs automatically when the database server issues a log-full event alarm, set ALARMPROGRAM to log_full.sh or log_full.bat .
log_full.bat	Windows	
no_log.sh	UNIX	To disable automatic logical-log backups, set ALARMPROGRAM to no_log.sh or no_log.bat .
no_log.bat	Windows	

Script Name	Platform	Description
alarmprogram.sh	UNIX	Handles event alarms and controls logical-log backups. Modify alarmprogram.sh or alarmprogram.bat and set ALARMPROGRAM to the full pathname of alarmprogram.sh or alarmprogram.bat . See “ Customizing the ALARMPROGRAM scripts” on page C-1.
alarmprogram.bat	Windows	

Important: Backup media should always be available for automatic log backups.

You can set the ALRM_ALL_EVENTS configuration parameter to specify whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only for specified noteworthy events (events greater than severity 1).

Instead of using the supplied scripts, you can write your own shell script, batch file, or binary program to execute events. Set ALARMPROGRAM to the full pathname of this file. The database server executes this script when noteworthy events occur. These events include database, table, index, or simple-large-object failure; all logs are full; internal subsystem failure; initialization failure; and long transactions. You can have the events noted in an email or pagermail message.

ALLOW_NEWLINE

onconfig.std *value*

0

range of values

0 = Disallow the newline character in quoted strings for all sessions.

1 = Allow the newline character in quoted strings for all sessions.

takes effect

When the database server is shut down and restarted

refer to

- Quoted strings in the *IBM Informix Guide to SQL: Syntax*
- Newline characters in quoted strings in the *IBM Informix ESQL/C Programmer's Manual*

You can specify that you want the database server to allow the newline character (**\n**) in a quoted string either for all sessions or for a specific session. A session is the duration of a client connection to the database server.

To allow or disallow newline characters in quoted strings for all sessions, set the ALLOW_NEWLINE parameter in the ONCONFIG file. To allow all remote sessions in a distributed query to support embedded newline characters, specify ALLOW_NEWLINE in their ONCONFIG files.

To allow or disallow a newline character in a quoted string for a particular session when ALLOW_NEWLINE is not set, you must execute the **ifx_allow_newline(boolean)** user-defined routine (UDR).

ALRM_ALL_EVENTS

onconfig.std *value*

0

takes effect

When the database server is shut down and restarted

range of values

0, 1

ALRM_ALL_EVENTS specifies whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only for noteworthy events. If ALRM_ALL_EVENTS is set to 1, it will trigger the ALARMPROGRAM and it will display all event alarms.

AUTO_AIOVPS

onconfig.std *value*

1

takes effect

When the database server is shut down and restarted

range of values

0= Off

1 = On

utilities

onmode -wf or **onmode -wm**

refer to "onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20.

AUTO_AIOVPS enables the database server to automatically increase the number of AIO VPs and page cleaner threads when the database server detects that the I/O workload has outpaced the performance of the existing AIO VPs. You can dynamically enable or disable the automatic increase of AIO VPs and page cleaner threads by using **onmode -wm** or **onmode -wf**.

The VPCLASS **ai** configuration parameter controls the number of AIO VPs. If the VPCLASS **ai** parameter is not set in the ONCONFIG file, the initial number of AIO VPs the database server starts when AUTO_AIOVPS is enabled is equal to the number of AIO chunks. The maximum number of AIO VPs the database server can start if VPCLASS **ai** is not set is 128.

AUTO_CKPTS

onconfig.std *value*

1

takes effect

When the database server is shut down and restarted

range of values

0= Off 1 = On

utilities

onmode -wf or **onmode -wm**

refer to "onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20.

AUTO_CKPTS allows the server to trigger checkpoints more frequently to avoid transaction blocking. You can dynamically enable or disable automatic checkpoints by using **onmode -wm** or **onmode -wf**.

AUTO_LRU_TUNING

onconfig.std *value*

1

takes effect

When the database server is shut down and restarted

range of values

0= Off 1 = On

utilities

onmode -wf or **onmode -wm**

refer to The following:

- “onmode -wm: Change LRU tuning status” on page 11-21
- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20

AUTO_LRU_TUNING enables automatic LRU tuning. You can dynamically enable or disable automatic LRU tuning by using **onmode -wm** or **onmode -wf**.

AUTO_REPREPARE

onconfig.std *value*

1

takes effect

When the database server is shut down and restarted

range of values

0= Disables the automatic reparation of prepared objects after the schema of a directly or an indirectly referenced table is modified. Also disables the automatic reoptimization of SPL routines after the schema of an indirectly referenced table is modified. 1 = Enables the automatic reparation and automatic reoptimization feature

refer to PREPARE, UPDATE STATISTICS, and SET ENVIRONMENT in *IBM Informix Guide to SQL: Syntax*

AUTO_REPREPARE controls whether a Dynamic Server feature is in effect that automatically re-optimizes SPL routines and re-prepares prepared objects after the schema of a table referenced by the SPL routine or by the prepared object has been changed.

When AUTO_REPREPARE is disabled, if DDL statements such as CREATE INDEX, DROP INDEX, DROP COLUMN, and RENAME COLUMN are executed, users of prepared objects that reference the modified tables or SPL routines that reference the modified tables indirectly receive -710 errors the next time they execute the prepared object or the SPL routine. Enabling AUTO_REPREPARE can avoid many -710 errors and can reduce the number of reprepare and reoptimize operations that users must perform manually after the schema of a table is modified.

BLOCKTIMEOUT

onconfig.std *value*

3600

units Seconds

takes effect

When the database server is shut down and restarted

BLOCKTIMEOUT specifies the number of seconds that a thread or database server will hang. After the timeout, the thread or database server will either continue processing or fail.

BTSCANNER

onconfig.std *value*

num=1,threshold=5000,rangesize=-1,alice=6,compression=default

syntax

BTSCANNER [num=scanner_threads][,threshold=dirty_hits][,rangesize=size]
[,alice=mode][,compression=level]

range of values

num = The number of B-tree scanner threads to start at system startup. The default is 1.

threshold = The number of dirty hits (committed deleted index items) an index must encounter before the index is placed on the hot list for cleaning. Systems updated frequently should increase this value by a factor of 10x or 100x. The default is 5000.

rangesize = The size, in kilobytes, an index or index fragment must exceed before the index is cleaned with range scanning. To allow small indexes to be scanned by the leaf scan method set rangesize to 100. The default is OFF (-1).

alice = The mode for adaptive linear index cleaning (alice) scanning. For small- to medium-sized systems with few or no indexes above 1 gigabyte, set alice to a mode of 6 or 7. For systems with large indexes, set alice to a higher mode. The initial system-wide alice mode determines the initial size of the bitmaps that track the deleted index entries. Valid values range from 0 to 12. The default is OFF (0).

compression = The level at which two partially used index pages are merged. The pages are merged if the data on those pages totals a set level. Valid values for the level are low, med (medium), high, or default. The system default value is med.

takes effect

When the database server is initialized. You can adjust these B-tree scanner settings with the **onmode -C** command while the database server is online.

refer to

- “onmode -C: Control the B-tree scanner” on page 11-4.
- Configuring B-Tree Scanner Information to Improve Transaction Processing, in your *IBM Informix Performance Guide*.

The BTSCANNER configuration parameter sets the B-tree scanner.

The B-tree scanner improves transaction processing for logged databases when rows are deleted from a table with indexes. The B-tree scanner threads remove deleted index entries and rebalance the index nodes. The B-tree scanner automatically determines which index items are to be deleted.

After all indexes above the threshold are cleaned, indexes below the threshold are added to the hot list. The default threshold is 500.

BUFFERPOOL

onconfig.std values

Operating systems with 2K default page size:

```
BUFFERPOOL default,buffers=10000,lrus=8,lru_min_dirty=50.00,  
lru_max_dirty=60.50  
BUFFERPOOL size=2k,buffers=50000,lrus=8,lru_min_dirty=50,  
lru_max_dirty=60
```

Operating systems with 4K default page size:

```
BUFFERPOOL default,buffers=10000,lrus=8,lru_min_dirty=50.00,  
lru_max_dirty=60.50  
BUFFERPOOL size=4k,buffers=10000,lrus=8,lru_min_dirty=50,  
lru_max_dirty=60
```

syntax BUFFERPOOL default,buffers=*num_buffers*,lrus=*num_lrus*,
lru_min_dirty=*percent_min*,lru_max_dirty=*percent_max_dirty*

BUFFERPOOL
size=*sizek*,buffers=*num_buffers*,lrus=*num_lrus*,lru_min_dirty=*percent_min*
,lru_max_dirty=*percent_max_dirty*

takes effect

When the database server is shut down and restarted

utilities

onparams -b (See “onparams -b: Add a new buffer pool” on page 13-4.)

onspaces (See “Specifying a Non-Default Page Size with the Same Size as the Buffer Pool” on page 14-10. ON-Monitor (See Figure 12-7 on page 12-6.)

refer to “onspaces -c -d: Create a dbspace” on page 14-7 *The IBM Informix Dynamic Server Administrator’s Guide*

The BUFFERPOOL configuration parameter specifies the default values for buffers and LRU queues in a buffer pool for both the default page size buffer pool and for any non-default pages size buffer pools.

Note: Information that was specified with the BUFFERS, LRUS, LRU_MAX_DIRTY, and LRU_MIN_DIRTY configuration parameters prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter.

The BUFFERPOOL configuration parameter consists of two lines in the **onconfig.std** file, as shown in this example for a platform with a default page size of 2K:

```
BUFFERPOOL default,buffers=10000,lrus=8,lru_min_dirty=50.00,lru_max_dirty=60.50  
BUFFERPOOL size=2k,buffers=50000,lrus=8,lru_min_dirty=50,lru_max_dirty=60
```

The top line specifies the default values that are used if you create a dbspace with a page size that does not already have a corresponding buffer pool created at start up. The line below the default line specifies the database server’s default values for a buffer pool, which are based on the database server’s default page size. When

you add a dbspace with a different page size with the **onspaces** utility or when you add a new buffer pool with the **onparams** utility, a new line is appended to the BUFFERPOOL configuration parameter in the ONCONFIG file. The page size for each buffer pool must be a multiple of the system's default page size. Below is an example of the BUFFERPOOL lines where a third line has been appended:

```
BUFFERPOOL default,buffers=10000,lrus=8,lru_min_dirty=50.00,lru_max_dirty=60.50
BUFFERPOOL size=2k,buffers=50000,lrus=8,lru_min_dirty=50,lru_max_dirty=60
BUFFERPOOL size=6k,buffers=3000,lrus=8,lru_min_dirty=50,lru_max_dirty=60
```

The order of precedence for the BUFFERPOOL configuration parameter settings is:

1. The BUFFERPOOL size line, for example:
BUFFERPOOL size=2k,buffers=50000,lrus=8,lru_min_dirty=50,lru_max_dirty=60
2. Any deprecated parameters in the ONCONFIG file:
 - BUFFERS
 - LRUS
 - LRU_MAX_DIRTY
 - LRU_MIN_DIRTY

For more information about deprecated configuration parameters, see Appendix D, "Discontinued Configuration Parameters," on page D-1.

3. The BUFFERPOOL default line, for example:
BUFFERPOOL default,buffers=10000,lrus=8,lru_min_dirty=50.00,lru_max_dirty=60.50
4. Database server defaults.

When you use **onspaces** to create a new dbspace with a new page size, the database server takes the values of **buffers**, **lrus**, **lru_min_dirty** and **lru_max_dirty** from BUFFERPOOL default line unless there already is a BUFFERPOOL entry for that page size.

You can use the **onparams** utility when the database server is in online, quiescent, or in administration mode to add a new buffer pool with a different page size. There must be one buffer pool for each page size used by the dbspaces and all dbspaces using that page size must use the single buffer pool with that page size. When you use the **onparams** utility to add a buffer pool or when you add a dbspace with a different page size with the **onspaces** utility, the information you specify is automatically appended to the ONCONFIG file and new values are specified using the BUFFERPOOL keyword. You cannot change the values by editing the **onconfig.std** file. If you need to resize or delete an existing buffer pool, you must restart the database server and then run **onparams** again.

Buffer pools that are added while the database server is running go into virtual memory, not into resident memory. Only those buffer pool entries that are specified in the ONCONFIG file at startup go into resident memory, depending on the availability of the memory you are using.

The fields in the BUFFERPOOL lines are not case sensitive (so you can specify **lrus** or **Lrus** or **LRUS**) and the fields can appear in any order.

For more information on buffer pools, including information on resizing and deleting buffer pools, see *IBM Informix Dynamic Server Administrator's Guide*.

The lrus Field

```
onconfig.std value
lrus=8
```

syntax lrus=num_lrus

units Number of LRU queues

range of values

32-bit platforms: 1 through 128 64-bit platforms: 1 through 512

The **lrus** field specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool. You can tune the value of **lrus**, in combination with the **lru_min_dirty** and **lru_max_dirty** fields, to control how frequently the shared-memory buffers are flushed to disk.

Setting **lrus** too high might result in excessive page-cleaner activity.

The buffers Field

onconfig.std value

: buffers=10000

syntax buffers=num_buffers

units Number of buffers. Each buffer is the size of the operating system page.

range of values

For 32-bit platform on UNIX: with page size equal to 2048 bytes: 100 through 1,843,200 buffers (1843200 = 1800 * 1024)

with page size equal to 4096 bytes: 100 through 921,600 buffers (921,600 = ((1800 * 1024)/4096) * 2048)

For 32-bit platform on Windows: 100 through 524,288 buffers (524,288 = 512 * 1024)

For 64-bit platforms: 100 through $2^{31}-1$ buffers (For the actual value for your 64-bit platform, see your machine notes. The maximum number of buffers on Solaris is 536,870,912.)

The **buffers** value specifies the maximum number of shared-memory buffers that the database server user threads have available for disk I/O on behalf of client applications. Therefore, the number of buffers that the database server requires depends on the applications. For example, if the database server accesses 15 percent of the application data 90 percent of the time, you need to allocate enough buffers to hold that 15 percent. Increasing the number of buffers can improve system performance.

Recommendation: Set the buffer space before you calculate other shared-memory parameters. On systems with a large amount of physical memory (4 GB or more), buffer space can be as much as 90 percent of physical memory.

If you also want to perform read-ahead, increase the value of **buffers**. After you have configured all other shared-memory parameters, if you find that you can afford to increase the size of shared memory, increase the value of **buffers** until buffer space reaches the recommended 25 percent maximum.

If your databases contain smart large objects, you need to consider them when you calculate the value for **buffers**, because smart large objects are stored in the default page size buffer pool. If your applications frequently access smart large objects that are 2 kilobytes or 4 kilobytes in size, use the buffer pool to keep them in memory longer.

Use the following formula to increase the value of **buffers**:

$$\text{Additional_BUFFERS} = \text{numcur_open_lo} * (\text{lo_userdata} / \text{pagesize})$$

numcur_open_lo

is the number of concurrently opened smart large objects that you can obtain from the **onstat -g smb fdd** option.

lo_userdata

is the number of bytes of smart-large-object data that you want to buffer.

pagesize

is the page size in bytes for the database server.

As a general rule, try to have enough buffers to hold two smart-large-object pages for each concurrently open smart large object. (The additional page is available for read-ahead purposes).

If the system uses lightweight I/O (as set by the access-mode constant **LO_NOBUFFER**), the system allocates the buffers from shared memory and does not store the smart large objects in the buffer pool. For information on access-mode flags and constants, see the chapter on “Working with Smart Large Objects of the Universal Data Option” in the *IBM Informix ESQ/L/C Programmer’s Manual*.

The **lru_min_dirty** Field

onconfig.std *value*

lru_min_dirty=50.00

syntax lru_min_dirty=*percent_min*

units Percent

range of values

0 through 100 (fractional values are allowed)

The **lru_min_dirty** field specifies the percentage of modified pages in the LRU queues at which page cleaning is no longer mandatory. Page cleaners might continue cleaning beyond this point under some circumstances. If a field is specified out of the range of values, then the default of 80.00 percent is set.

The **lru_max_dirty** Field

onconfig.std *value*

lru_max_dirty=60.50

syntax lru_max_dirty=*percent_max*

units Percent

range of values

0 through 100 (fractional values are allowed)

The **lru_max_dirty** field specifies the percentage of modified pages in the LRU queues at which the queue is cleaned. If a field is specified out of the range of values, then the default of 60.00 percent is set.

The **size** Field

onconfig.std *value*

2K default page size: size=2k 4K default page size: size=4k

syntax size=size
units Kilobytes
range of values
2 through 16

The **size** field specifies the page size for the particular BUFFERPOOL line. The k is optional.

System Page Size

The system page size is the default page size and is platform-dependent on Dynamic Server.

You can use the following utilities to display the system page size.

Utility Description

onstat -b

Displays the system page size, given as buffer size on the last line of the output

oncheck -pr

Checks the root-dbspace reserved pages and displays the system page size in the first section of its output

ON-Monitor(UNIX)

Displays the system page size under the **Parameters > Initialize** option.
Displays system page size under the **Parameters > Shared-Memory** option, which does not require the database server to be running.

CKPTINTVL

onconfig.std *value*
300

units Seconds

range of values
Any value greater than or equal to 0

takes effect

When the database server is shut down and restarted.

RTO_SERVER_RESTART and CKPTINTVL are mutually exclusive. If the RTO_SERVER_RESTART configuration parameter is enabled, it will trigger checkpoints and CKPTINTVL values are ignored. Otherwise, CKPTINTVL values are used to trigger checkpoints.

refer to

- Checkpoints, in the shared-memory and fast-recovery chapters of the *IBM Informix Administrator's Guide*
- Your *IBM Informix Performance Guide*

CKPTINTVL specifies the frequency, expressed in seconds, at which the database server checks to determine whether a checkpoint is needed. When a checkpoint occurs, all pages in the shared-memory buffer pool are written to disk.

If you set CKPTINTVL to an interval that is too short, the system spends too much time performing checkpoints, and the performance of other work suffers. If you set CKPTINTVL to an interval that is too long, fast recovery might take too long.

In practice, 30 seconds is the smallest interval that the database server checks. If you specify a checkpoint interval of 0, the database server does not check if the checkpoint interval has elapsed. However, the database server still performs checkpoints. Other conditions, such as the physical log becoming 75 percent full, also cause the database server to perform checkpoints.

CLEANERS

onconfig.std *value*
8

units Number of page-cleaner threads

range of values
1 through 128

takes effect
When the database server is shut down and restarted.

utilities
onstat -F (see “onstat -F: Print counts” on page 15-23.)

refer to How the database server flushes data to disk, in the shared-memory chapter of the *IBM Informix Administrator's Guide*

CLEANERS specifies the number of page-cleaner threads available during the database server operation. By default, the database server always runs one page-cleaner thread. A general guideline is one page cleaner per disk drive. The value specified has no effect on the size of shared memory.

Based on the server work load, the server automatically attempts to optimize AIO VPs and page-cleaner threads and adjust the number of AIO VPs and page-cleaner threads upward when needed. Automatic AIO VP and page-cleaner thread tuning can be disabled using the environmental variable IFX_NO_AIOVP_TUNING or the **onmode -wm** utility option.

CONSOLE

onconfig.std *value*
On UNIX: **\$INFORMIXDIR/tmp/online.con** On Windows: **online.con**

range of values
Pathname

takes effect
When the database server is shut down and restarted

refer to The system console in the chapter on database server administration in the *IBM Informix Administrator's Guide*

CONSOLE specifies the pathname and the filename for console messages.

DATASKIP

syntax DATASKIP *state* [*dbspace1 dbspace2 ...*]

The *state* entry is required. If *state* is ON, at least one *dbspace* entry is required.

onconfig.std *value*
None

if not present
OFF

separators
Space

range of values
ALL = Skip all unavailable fragments. OFF = Turn off DATASKIP. ON = Skip some unavailable fragments.

utilities
onspaces -f (see “ onspaces -f: Specify DATASKIP parameter” on page 14-21.) **onstat -f** (see “onstat -D: Print page-read and page-write information” on page 15-22.)

refer to

- “ onspaces -f: Specify DATASKIP parameter” on page 14-21
- Your *IBM Informix Performance Guide*

DATASKIP lets you avoid points of media failure. This capability can result in higher availability for your data. To instruct the database server to skip some or all unavailable fragments, set this parameter. Whenever the database server skips over a *dbspace* during query processing, a warning is returned.

ESQL/C:

The previously reserved SQLCA warning flag **sqlwarn.sqlwarn7** is set to W for IBM Informix ESQL/C.

Use the following syntax in the parameter line:

```
DATASKIP OFF  
DATASKIP ON dbspace1 dbspace2...  
DATASKIP ALL
```

Use the **-f** option of the **onspaces** utility to alter the value of the DATASKIP parameter at runtime.

An application can use the SQL statement SET DATASKIP to override the DATASKIP value that the ONCONFIG parameter or **onspaces** sets. If the application then executes the SQL statement SET DATASKIP DEFAULT, the DATASKIP value for that session returns to whatever value is currently set for the database server.

DBCREATE_PERMISSION

syntax DBCREATE_PERMISSION *value*

onconfig.std *value*
#DBCREATE_PERMISSION informix (suggested value, but not set)

units usernames

separator
comma

takes effect

When the database server is shut down and restarted

DBCREATE_PERMISSION restricts the permission to create databases to the specified user. You can include multiple copies of DBCREATE_PERMISSION in the ONCONFIG file to give additional users permission to create databases.

The informix user always has permission to create databases. To restrict the ability to create databases to the informix user, add the following line to the ONCONFIG file:

```
DBCREATE_PERMISSION informix
```

DB_LIBRARY_PATH

syntax DB_LIBRARY_PATH *value*

onconfig.std *value*

Not set

if not present

The database server can load external modules from any location

range of values

List of path names (up to 512 bytes)

separators

Comma

takes effect

When the database server is shut down and restarted

The DB_LIBRARY_PATH configuration parameter specifies a comma-separated list of valid directory prefix locations from which the database server can load external modules, such as DataBlade[®] Modules. You can also include server environment variables, such as \$INFORMIXDIR.

You must specify paths for external modules exactly as they are registered with Dynamic Server. Relative paths or paths that include double periods (..) are not valid. External modules in the file systems that are not specified by this parameter cannot be loaded. This list is scanned prior to loading C language modules.

If you set this parameter, you must also include the string \$INFORMIXDIR/extend as part of the value. If the string \$INFORMIXDIR/extend is not included in DB_LIBRARY_PATH, IBM-supplied DataBlade Modules, the BladeManager, Large Object Locator DataBlade module functions, and DataBlade modules that you created with the DataBlade Developers Kit will not load.

DBSERVERALIASES

onconfig.std *value*

None

if not present

None

separators

Comma

range of values

Up to 128 lowercase characters for each dbserver alias. Up to 32 values separated by commas. The value for DBSERVERALIASES follows the same rules as the DBSERVERNAME parameter (see “DBSERVERNAME”).

takes effect

When the database server is shut down and restarted. In addition, you might need to update the **sqlhosts** file or registry of each database server.

Informix MaxConnect users

To use Informix MaxConnect with more than one communication protocol, specify additional dbservernames in the DBSERVERALIASES parameter in the ONCONFIG file. The value of the **INFORMIXSERVER** environment variable on the client must match either the DBSERVERNAME or one of the entries of the DBSERVERALIASES parameter.

refer to The following topics in the chapter on client/server communications in the *IBM Informix Administrator's Guide*:

- ONCONFIG parameters for connectivity
- Using multiple connection types

DBSERVERALIASES specifies a list of alternative dbservernames and lets you assign multiple aliases to a database server, so each entry in the **sqlhosts** file or registry can have a unique name.

You can use DBSERVERALIASES to specify alternative dbservernames and aliases for both Secure Sockets Layer (SSL) and non-SSL connection protocols.

If Dynamic Server supports more than one communication protocol (for example, both an IPC mechanism and the TCP network protocol), you must describe each valid connection to the database server with an entry in the **sqlhosts** file or registry. For example, suppose you have a server that has the DBSERVERNAME `sanfrancisco` and you have a DBSERVERALIASES entry called `menlo` for an SSL connection. You must define information for both `sanfrancisco` and `menlo` servers in the **sqlhosts** file or registry.

If the database server needs to support both the SQLI and Distributed Relational Database Architecture™ (DRDA®) protocols, you must assign an alias to the DRDA database server and add an entry in the **sqlhosts** file.

Important: You can specify up to 32 DBSERVERALIASES for a database server. If you attempt to define more than 32 DBSERVERALIASES, a warning message displays twice on the console. If you attempt to specify the DBSERVERALIASES all on one line, and the line exceeds 512 bytes, the excess bytes are truncated.

For each alternate name listed in DBSERVERALIASES, the database server starts an additional listener thread. If you have many client applications connecting to the database server, you can distribute the connection requests between several listener threads and reduce connection time. To take advantage of the alternate connections, instruct some of your client applications to use a **CONNECT TO dbserveralias** statement instead of **CONNECT TO dbservername**.

DBSERVERNAME

onconfig.std *value*
None

if not present

On UNIX: *hostname* On Windows: *ol_hostname* (The *hostname* variable is the name of the host computer.)

range of values

Up to 128 lowercase characters

DBSERVERNAME must begin with a letter and can include any printable character except the following characters:

- Uppercase characters
- A field delimiter (space or tab)
- A newline character
- A comment character
- A hyphen, minus, or @ character

takes effect

When the database server is shut down and restarted. The **sqlhosts** file or registry of each database server that communicates with this database server might need to be updated. In addition, the **INFORMIXSERVER** environment variable for all users might need to be changed.

Informix MaxConnect users

The value of the **INFORMIXSERVER** environment variable on the client must match either the DBSERVERNAME or one of the entries of the DBSERVERALIASES parameter.

refer to DBSERVERNAME configuration parameter in the chapter on client/server communications in the *IBM Informix Administrator's Guide*

When you install the database server, specify the *dbservername*. DBSERVERNAME specifies a unique name associated with this specific occurrence of the database server. The value of DBSERVERNAME is called the *dbservername*. Each *dbservername* is associated with a communication protocol in the **sqlhosts** file or registry. If the database server uses multiple communication protocols, additional values for *dbservername* must be defined with the DBSERVERALIASES configuration parameter.

Client applications use *dbservername* in the **INFORMIXSERVER** environment variable and in SQL statements such as CONNECT and DATABASE, which establish a connection to a database server.

Important: To avoid conflict with other instances of Informix database servers on the same computer or node, it is recommended that you use DBSERVERNAME to assign a *dbservername* explicitly.

DBSPACETEMP

onconfig.std *value*

None

if not present

ROOTNAME

separators

Comma or colon (no white space)

range of values

The list of dbspaces can contain standard dbspaces, temporary dbspaces, or

both. Use a colon or comma to separate the dbspaces in your list. The length of the list cannot exceed 254 characters.

takes effect

When the database server is shut down and restarted

environment variable **DBSPACETEMP**

Specifies dbspaces that the database server uses to store temporary tables for a particular session. If **DBSPACETEMP** is not set, the default location is the root dbspace.

utilities

onspaces -t (see “Creating a Temporary Dbspace with the -t Option” on page 14-10.) **onstat -d flags** field (see “onstat -d: Print chunk information” on page 15-19.)

refer to

- What is a temporary table, in the chapter on data storage in the *IBM Informix Administrator's Guide*
- *IBM Informix Guide to SQL: Reference*
- The order of precedence that the database server uses when it creates implicit sort files, in the *IBM Informix Performance Guide*
- The order of precedence of the default locations where the database server stores logged and unlogged temporary tables in the *IBM Informix Guide to SQL: Reference*.

DBSPACETEMP specifies a list of dbspaces that the database server uses to globally manage the storage of temporary tables. DBSPACETEMP improves performance by enabling the database server to spread out I/O for temporary tables efficiently across multiple disks. The database server also uses temporary dbspaces during backups to store the before-images of data that are overwritten while the backup is occurring.

DBSPACETEMP can contain dbspaces with a non-default page size, but all of the dbspaces in the DBSPACETEMP list must have the same page size. For more information about dbspaces in non-default buffer pools, see “BUFFERPOOL” on page 1-17.

If a client application needs to specify an alternative list of dbspaces to use for its temporary-table locations, the client can use the **DBSPACETEMP** environment variable to list them. The database server uses the storage locations that the **DBSPACETEMP** environment variable specifies only when you use the HIGH option of UPDATE STATISTICS.

Important: The dbspaces that you list in the DBSPACETEMP configuration parameter must consist of chunks that are allocated as raw UNIX devices. On Windows, you can create temporary dbspaces in NTFS files.

If both standard and temporary dbspaces are listed in the DBSPACETEMP configuration parameter or environment variable, the following rules apply:

- Sort, backup, implicit, and nonlogging explicit temporary tables are created in temporary dbspaces if adequate space exists.
- Explicit temporary tables created without the WITH NO LOG option are created in standard (rather than temporary) dbspaces.

When you create a temporary dbspace with ISA or with the **onspaces** utility, the database server does not use the newly created temporary dbspace until you perform the following steps.

To enable the database server to use the new temporary dbspace:

1. Add the name of a new temporary dbspace to your list of temporary dbspaces in the DBSPACETEMP configuration parameter, the **DBSPACETEMP** environment variable, or both.
2. Restart the database server with the **oninit** command (UNIX) or restart the database server service (Windows).

If you use the **DBSPACETEMP** environment variable to create a temporary dbspace in a user session, the change takes effect immediately and overrides the **DBSPACETEMP** value in the **ONCONFIG** file.

Using Hash Join Overflow and DBSPACETEMP

Dynamic Server uses an operating-system directory or file to direct any overflow that results from the following database operations if you do not set the **DBSPACETEMP** environment variable or **DBSPACETEMP** configuration parameter. You can specify the operating-system directory or file in the following ways:

- SELECT statement with GROUP BY clause
- SELECT statement with ORDER BY clause
- Hash-join operation
- Nested-loop join operation
- Index builds

If you do not set the **DBSPACETEMP** environment variable or **DBSPACETEMP** configuration parameter, the database server directs any overflow that results from the preceding operations to the operating-system directory or file that you specify in one of the following variables:

•

UNIX Only: The operating-system directory or directories that the **PSORT_DBTEMP** environment variable specifies, if it is set. If **PSORT_DBTEMP** is not set, the database server writes sort files to the operating-system file space in the **tmp** directory.

•

Windows Only: The directory specified in **TEMP** or **TMP** in the User Environment Variables window in **Control Panel > System**.

DD_HASHMAX

onconfig.std *value*

10

units Maximum number of tables in a hash bucket

range of values

Positive integers

takes effect

When the database server is shut down and restarted

utilities

Use a text editor to modify the configuration file.

refer to

- Configuration effects on memory, in your *IBM Informix Performance Guide*
- “ DD_HASHSIZE”

DD_HASHMAX specifies the maximum number of tables in each hash bucket in the data-dictionary cache. A *hash bucket* is the unit of storage (typically a page) whose address is computed by the hash function. A hash bucket contains several records.

For example, if DD_HASHMAX is 10 and DD_HASHSIZE is 100, you can store information about 1000 tables in the data-dictionary cache, and each hash bucket can have a maximum of 10 tables.

DD_HASHSIZE

onconfig.std *value*

31

units Number of hash buckets or lists

range of values

Any positive integer; a prime number is recommended

takes effect

When the database server is shut down and restarted

utilities

Use a text editor to modify the configuration file.

refer to

- Configuration effects on memory, in your *IBM Informix Performance Guide*
- “ DD_HASHMAX” on page 1-28

DD_HASHSIZE specifies the number of hash buckets or lists in the data-dictionary cache.

DEADLOCK_TIMEOUT

onconfig.std *value*

60

units Seconds

range of values

Positive integers

takes effect

When the database server is shut down and restarted

utilities

onstat -p dltouts field (See “onstat -p: Print profile counts” on page 15-148.)

refer to Configuration parameters used in two-phase commits, in the chapter on multiphase commit protocols in the *IBM Informix Administrator's Guide*

DEADLOCK_TIMEOUT specifies the maximum number of seconds that a database server thread can wait to acquire a lock. Use this parameter only for distributed queries that involve a remote database server. Do not use this parameter for nondistributed queries.

DEF_TABLE_LOCKMODE

onconfig.std *value*
PAGE

if not present
PAGE

range of values
PAGE = sets lock mode to page for new tables ROW = sets lock mode to row for new tables

takes effect
When the database server is shut down and restarted

environment variable
IFX_DEF_TABLE_LOCKMODE

refer to

- Environment variables in the *IBM Informix Guide to SQL: Reference*
- Setting lock modes, in the *IBM Informix Guide to SQL: Tutorial*
- Configuring lock mode, in the *IBM Informix Performance Guide*

If DEF_TABLE_LOCKMODE = ROW, it sets the lock mode to row for every newly created table for all sessions that are connected to logging or nonlogging databases. This parameter has no effect on the lock mode for existing tables.

If DEF_TABLE_LOCKMODE is set to PAGE, the USELASTCOMMITTED configuration parameter and COMMITTED READ LAST COMMITTED option of the SET ISOLATION statement cannot enable access to the most recently committed data in tables on which uncommitted transactions hold exclusive locks, unless the tables were explicitly created or altered to have ROW as their locking granularity.

The rules of precedence for setting the lock mode are as follows.

Precedence

Command

1 (highest)

CREATE TABLE or ALTER TABLE statements that use the LOCK MODE clause

2 **IFX_DEF_TABLE_LOCKMODE** environment variable set on the client side

3 **IFX_DEF_TABLE_LOCKMODE** environment variable set on the server side

4 DEF_TABLE_LOCKMODE value in ONCONFIG file

5 (lowest)

Default behavior (page-level locking)

DIRECT_IO (UNIX)

onconfig.std *value*

0

range of values

0 = direct I/O is not used 1 = direct I/O is used if available

takes effect

When the database server is shut down and restarted

refer to

- Direct I/O information in the *IBM Informix Performance Guide*
- Direct I/O information in the *IBM Informix Administrator's Guide*
- "AUTO_AIOVPS" on page 1-14
- "NUMAIOVPS (Discontinued)" on page D-7

DIRECT_IO controls the use of direct I/O for cooked files used for dbspace chunks.

If you use direct I/O for cooked files used for dbspace chunks and the DIRECT_IO configuration parameter is enabled, you might be able to reduce the number of AIO virtual processors.

DIRECT_IO is not used for temporary dbspaces and can only be used for dbspace chunks whose file systems support direct I/O for the page size. If direct I/O is used for a dbspace chunk, KAIO (kernel asynchronous I/O) is used if supported by the file system. However, KAIO is not used if the environment variable KAIOOFF is set. If DIRECT_IO is enabled and KAIO is also used, the number of AIO virtual processors can also be reduced. If DIRECT_IO is enabled and KAIO is not used, the number of AIO virtual processors should not be reduced.

Note:

Windows platforms do not recognize the value of the DIRECT I/O configuration parameter, and direct I/O is used for dbspace chunks on Windows platforms regardless of the value of the DIRECT_I/O configuration parameter.

DIRECTIVES

onconfig.std *value*

1

range of values

0 optimizer directives disabled

1 optimizer directives enabled

takes effect

When the database server is shut down and restarted

environment variable

IFX_DIRECTIVES

refer to

- Environment variables in the *IBM Informix Guide to SQL: Reference*
- SQL directives, in the *IBM Informix Guide to SQL: Syntax*
- Performance impact of directives, in your *IBM Informix Performance Guide*

The **DIRECTIVES** parameter enables or disables the use of SQL directives. SQL directives allow you to specify behavior for the query optimizer in developing query plans for **SELECT**, **UPDATE**, and **DELETE** statements.

Set **DIRECTIVES** to 1, which is the default value, to enable the database server to process directives. Set **DIRECTIVES** to 0 to disable the database server from processing directives. Client programs also can set the **IFX_DIRECTIVES** environment variable to **ON** or **OFF** to enable or disable processing of directives by the database server. The setting of the **IFX_DIRECTIVES** environment variable overrides the setting of the **DIRECTIVES** configuration parameter. If you do not set the **IFX_DIRECTIVES** environment variable, all sessions for a client inherit the database server configuration for processing SQL directives.

DISABLE_B162428_XA_FIX

onconfig.std *value*

Not in **onconfig.std**

units Integer

range of values

0 = (Default) Frees transactions only when an `xa_rollback` is called
1 = Frees transactions if transaction rollback for other than an `xa_rollback`

takes effect

When the database server is shut down and restarted

refer to *IBM Informix Guide to SQL: Reference*

Set **DISABLE_B162428_XA_FIX** to 1 to immediately free all global transactions after a transaction rollback, which is the default for Dynamic Server 9.40 and earlier versions. The default behavior for Dynamic Server 10.0 is to free global transactions after an `xa_rollback` is called, and this behavior is required to conform to the XA state table that a transaction can be freed only after `xa_rollback` is called. Setting **DISABLE_B162428_XA_FIX** to 1 ensures that applications written for the earlier version of Dynamic server work properly.

You can override the **DISABLE_B162428_XA_FIX** configuration parameter for a client session with the **IFX_XASTDCOMPLIANCE_XAEND** environment variable. Setting **IFX_XASTDCOMPLIANCE_XAEND** to 1 will free transactions only when an `xa_rollback` is called. Setting **IFX_XASTDCOMPLIANCE_XAEND** to 0 will free transactions if the transaction rollback is for other than an `xa_rollback`.

DRDA_COMMBUFSIZE

onconfig.std *value*

Not in **onconfig.std**

if not present

32K

range of values

Minimum = 4 Kilobytes
Maximum = 2 Megabytes

takes effect

When shared memory is initialized

refer to *Setting the Size of the DRDA Communications Buffer in the IBM Informix Administrator's Guide.*

DRDA_COMMBUFFSIZE specifies the size of the DRDA communications buffer. When a DRDA session is established, the session is allocated a communication buffer equal to the current buffer size. If the buffer size is subsequently changed, existing connections are not affected, but new DRDA connections use the new size. IDS silently resets values greater than 2 Megabyte to 2 Megabytes and resets values less than 4 Kilobytes to the 32 Kilobyte default value.

Users may specify the DRDA_COMMBUFFSIZE value in either MB or KB by adding either 'M' or 'K' to the value. The letter is case-insensitive, and the default is kilobytes. For example, a one megabyte buffer can be specified in any of these ways:

- DRDA_COMMBUFFSIZE 1M
- DRDA_COMMBUFFSIZE 1m
- DRDA_COMMBUFFSIZE 1024K
- DRDA_COMMBUFFSIZE 1024k
- DRDA_COMMBUFFSIZE 1024

DRAUTO

onconfig.std *value*

0

range of values

0 signifies OFF = Do not automatically switch the server type in the high-availability cluster environment.

1 signifies RETAIN_TYPE = Automatically switch secondary to standard when the primary server fails. Switch back to secondary when restarting the high-availability cluster.

2 signifies REVERSE_TYPE = Automatically switch secondary to standard when the primary server fails. Switch to primary (and switch original primary to secondary) when restarting the high-availability cluster.

3 signifies that the server requires verification from Connection Manager Arbitrator.

takes effect

When shared memory is initialized

utilities

ON-Monitor > Parameters > data-Replication > Auto **onstat** (See "onstat -g dri: Print High-Availability Cluster information" on page 15-50.)

DRAUTO determines how a secondary database server reacts to a primary server failure. This parameter should have the same value on both the primary and any secondary servers.

If DRAUTO is set to OFF, the secondary database server remains a secondary database server in read-only mode when a failure of the primary server occurs.

If DRAUTO is set to either RETAIN_TYPE or REVERSE_TYPE, the secondary database server switches to type standard automatically when an HDR failure is detected. If DRAUTO is set to RETAIN_TYPE, the original secondary database server switches back to type secondary when the HDR connection is restored. If DRAUTO is set to REVERSE_TYPE, the original secondary database server switches to type primary when the HDR connection is restored, and the original primary switches to type secondary.

Setting DRAUTO to 3 prevents the possibility of having multiple primary servers within a high-availability cluster. If an attempt is made to bring a server on line as a primary server and DRAUTO=3, then Connection Manager Arbitrator will verify that there are no other active primary servers in the cluster. If another primary server is active, then Connection Manager Arbitrator will reject the request. The server will be unavailable until the Connection Manager Arbitrator is enabled.

Use this parameter carefully. A network failure (that is, when the primary database server does not really fail, but the secondary database server perceives network slowness as an HDR failure) can cause the two database servers to become out of synch.

DRIDXAUTO

onconfig.std *value*
0

range of values
0= Off 1 = On

utilities

onstat (See “onstat -g dri: Print High-Availability Cluster information” on page 15-50.)

takes effect

When the database server is shut down and restarted

Specifies whether the primary High-Availability Data Replication (HDR) server automatically starts index replication if the secondary HDR server detects a corrupted index. To enable automatic index replication, set the value of the DRIDXAUTO configuration parameter to 1. You can alter the value of DRIDXAUTO for a running server instance without restarting the instance using the **onmode -d idxauto** command. However, the **onmode -d idxauto** command will not change the value of the DRIDXAUTO parameter in the ONCONFIG file. For more information, see “onmode -d: Replicate an index with data-replication” on page 11-8.

DRINTERVAL

onconfig.std *value*
30

units Seconds

range of values
-1, 0, and positive integer values

takes effect

When the database server is shut down and restarted

utilities

onstat (See “onstat -g dri: Print High-Availability Cluster information” on page 15-50.)

refer to When log records are sent, in the chapter on High-Availability Data Replication in the *IBM Informix Administrator's Guide*

DRINTERVAL specifies the maximum interval in seconds between flushing of the high-availability data-replication buffer. To update synchronously, set the parameter to -1.

DRLOSTFOUND

onconfig.std *value*

On UNIX: **\$INFORMIXDIR/etc/dr.lostfound**

On Windows: **\$INFORMIXDIR\tmp**

range of values

Pathname

takes effect

When the database server is shut down and restarted

utilities

onstat (See “onstat -g dri: Print High-Availability Cluster information” on page 15-50.)

refer to Lost-and-found transactions, in the chapter on High-Availability Data Replication in the *IBM Informix Administrator's Guide*

DRLOSTFOUND specifies the pathname to the **dr.lostfound.timestamp** file. This file contains transactions committed on the primary database server but not committed on the secondary database server when the primary database server experiences a failure. The file is created with a time stamp appended to the filename so that the database server does not overwrite another lost-and-found file if one already exists.

This parameter is not applicable if updating between the primary and secondary database servers occurs synchronously (that is, if DRINTERVAL is set to -1).

DRTIMEOUT

onconfig.std *value*

30

units Seconds

range of values

Positive integers

takes effect

When the database server is shut down and restarted

utilities

onstat (See “onstat -g dri: Print High-Availability Cluster information” on page 15-50.)

refer to How High-Availability Data Replication failures are detected, in the chapter on High-Availability Data Replication in the *IBM Informix Administrator's Guide*

DRTIMEOUT applies only to high-availability data-replication pairs. This value specifies the length of time, in seconds, that a database server in a high-availability data-replication pair waits for a transfer acknowledgment from the other database server in the pair. Use the following formula to calculate DRTIMEOUT:

$DRTIMEOUT = wait_time / 4$

In this formula, *wait_time* is the length of time, in seconds, that a database server in a high-availability data-replication pair must wait before it assumes that a high-availability data-replication failure occurred.

For example, suppose you determine that *wait_time* for your system is 160 seconds. Use the preceding formula to set DRTIMEOUT as follows:

DRTIMEOUT = 160 seconds / 4 = 40 seconds

DS_HASHSIZE

onconfig.std *value*

31

units Number of hash buckets or lists

range of values

Any positive integer; a prime number is recommended

takes effect

When the database server is shut down and restarted

refer to

- *IBM Informix Performance Guide* for how to monitor and tune the data-distribution cache
- “DS_POOLSIZ” on page 1-38

The DS_HASHSIZE parameter specifies the number of hash buckets in the data-distribution cache that the database server uses to store and access column statistics that the UPDATE STATISTICS statement generates in the MEDIUM or HIGH mode.

Use DS_HASHSIZE and DS_POOLSIZ to improve performance of frequently executed queries in a multiuser environment.

For information on configuration parameters for UDR cache, see “PC_HASHSIZE” on page 1-77 and “PC_POOLSIZ” on page 1-78.

DS_MAX_QUERIES

onconfig.std *value*

None

if not present

num_cpu_vps * 2 * 128

units Number of queries

range of values

Minimum = 1 Maximum = 8,388,608 (8 megabytes)

utilities

onmode -Q (see “onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10.) **onstat -g mgm** (See “onstat -g mgm: Print MGM resource information ” on page 15-74.)

refer to

- “Specifying the Number of CPU VPs” on page 1-113
- Parallel database query in your *IBM Informix Performance Guide*

DS_MAX_QUERIES is the maximum number of PDQ queries that can run concurrently. The Memory Grant Manager (MGM) reserves memory for a query based on the following formula:

$$\text{memory_reserved} = \text{DS_TOTAL_MEMORY} * \frac{(\text{PDQ-priority} / 100) * (\text{MAX_PDQPRIORITY} / 100)}$$

The value of PDQPRIORITY is specified in either the **PDQPRIORITY** environment variable or the SQL statement SET PDQPRIORITY.

DS_MAX_SCANS

onconfig.std *value*

1048576 or (1024 * 1024)

units Number of PDQ scan threads

range of values

10 through (1024 * 1024)

utilities

onmode -S (see “ onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10.) **onstat -g mgm** (See “onstat -g mgm: Print MGM resource information ” on page 15-74.)

refer to Parallel database query in your *IBM Informix Performance Guide*

DS_MAX_SCANS limits the number of PDQ scan threads that the database server can execute concurrently. When a user issues a query, the database server apportions some number of scan threads, depending on the following values:

- The value of PDQ priority (set by the environment variable **PDQPRIORITY** or the SQL statement SET PDQPRIORITY)
- The ceiling that you set with DS_MAX_SCANS
- The factor that you set with MAX_PDQPRIORITY
- The number of fragments in the table to scan (*nfrags* in the formula)

The Memory Grant Manager (MGM) tries to reserve scan threads for a query according to the following formula:

$$\text{reserved_threads} = \min (nfrags, (\text{DS_MAX_SCANS} * \frac{\text{PDQPRIORITY} / 100 * (\text{MAX_PDQPRIORITY} / 100)}{100}))$$

If the DS_MAX_SCANS part of the formula is greater than or equal to the number of fragments in the table to scan, the query is held in the ready queue until as many scan threads are available as there are table fragments. Once underway, the query executes quickly because threads are scanning fragments in parallel.

For example, if *nfrags* equals 24, DS_MAX_SCANS equals 90, **PDQPRIORITY** equals 50, and MAX_PDQPRIORITY equals 60, the query does not begin execution until *nfrags* scan threads are available. Scanning takes place in parallel.

If the DS_MAX_SCANS formula falls below the number of fragments, the query might begin execution sooner, but the query takes longer to execute because some threads scan fragments serially.

If you reduce DS_MAX_SCANS to 40 in the previous example, the query needs fewer resources (12 scan threads) to begin execution, but each thread needs to scan two fragments serially. Execution takes longer.

DS_NONPDQ_QUERY_MEM

onconfig.std

128

units Kilobytes

range of values

From 128 Kilobytes to 25 percent of the value of DS_TOTAL_MEMORY

takes effect

When the database server is initialized

utilities

onstat -g mgm (See “onstat -g mgm: Print MGM resource information ” on page 15-74.) **onmode ON-Monitor**

Use the DS_NONPDQ_QUERY_MEM configuration parameter to increase the amount of memory that is available for a query that is not a Parallel Database Query (PDQ). (You can only use this parameter if PDQ priority is set to zero.) If you specify a value for the DS_NONPDQ_QUERY_MEM parameter, determine and adjust the value based on the number and size of table rows.

Tip: Set the value to generally not exceed the largest available temporary dbspace size.

The DS_NONPDQ_QUERY_MEM value is calculated during database server initialization based on the calculated DS_TOTAL_MEMORY value. If during the processing of the DS_NONPDQ_QUERY_MEM, the database server changes the value that you set, the server sends a message in this format:

DS_NONPDQ_QUERY_MEM recalculated and changed from *old_value* Kb to *new_value* Kb.

In the message, *old_value* represents the value that you assigned to DS_NONPDQ_QUERY_MEM in the user configuration file, and *new_value* represents the value determined by the database server.

The value for DS_NONPDQ_QUERY_MEM can be changed using the **onmode -wf** option or superseded for a session with the **onmode -wm** option. For more information about **onmode**, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

DS_POOLSIZE

onconfig.std *value*

127

default value

127

units Maximum number of entries in the data-distribution cache

range of values

Any positive value from 127 to *x*, where *x* is dependent upon the shared memory configuration and available shared memory for the server instance.

takes effect

When the database server is shut down and restarted

refer to

- *IBM Informix Performance Guide* for how to monitor and tune the data-distribution cache
- “DS_HASHSIZE” on page 1-36

The DS_POOLSIZE parameter specifies the maximum number of entries in each hash bucket in the data-distribution cache that the database server uses to store and access column statistics that the UPDATE STATISTICS statement generates in the MEDIUM or HIGH mode.

Use DS_HASHSIZE and DS_POOLSIZE to improve performance of frequently executed queries in a multi-user environment.

For information on configuration parameters for UDR cache, see “PC_HASHSIZE” on page 1-77 and “PC_POOLSIZE” on page 1-78.

DS_TOTAL_MEMORY

onconfig.std *value*

None

if not present

If SHMTOTAL=0 and DS_MAX_QUERIES is set, DS_TOTAL_MEMORY = DS_MAX_QUERIES * 128.

If SHMTOTAL=0 and DS_MAX_QUERIES is not set, DS_TOTAL_MEMORY = num_cpu_vps * 2 * 128.

units Kilobytes

range of values

If DS_MAX_QUERY is set, the minimum value is DS_MAX_QUERY * 128.

If DS_MAX_QUERY is not set, the minimum value is num_cpu_vps * 2 * 128.

Maximum value for 32-bit platform: 2 gigabytes Maximum value for 64-bit platform: 4 gigabytes

utilities

onmode -M (see “ onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10.) **onstat -g mgm** (See “onstat -g mgm: Print MGM resource information ” on page 15-74.)

refer to

- Your *IBM Informix Performance Guide* for the algorithms
- “ SHMTOTAL” on page 1-92
- “SHMVIRTSIZE” on page 1-93
- “Specifying the Number of CPU VPs” on page 1-113
- The maximum memory available on your platform, in the machine notes

DS_TOTAL_MEMORY specifies the amount of memory available for PDQ queries. It should be smaller than the computer physical memory, minus fixed overhead such as operating-system size and buffer-pool size.

Do not confuse DS_TOTAL_MEMORY with the configuration parameters SHMTOTAL and SHMVIRTSIZE. SHMTOTAL specifies all the memory for the

database server (total of the resident, virtual, and message portions of memory). SHMVIRTSIZE specifies the size of the virtual portion. DS_TOTAL_MEMORY is part of SHMVIRTSIZE.

For OLTP applications, set DS_TOTAL_MEMORY to between 20 and 50 percent of the value of SHMTOTAL in kilobytes.

For applications that involve large decision-support (DSS) queries, increase the value of DS_TOTAL_MEMORY to between 50 and 80 percent of SHMTOTAL. If you use your database server for DSS queries exclusively, set this parameter to 90 and 100 percent of SHMTOTAL.

Set the DS_TOTAL_MEMORY configuration parameter to any value not greater than the quantity (SHMVIRTSIZE - 10 megabytes).

Algorithm for DS_TOTAL_MEMORY

The database server derives a value for DS_TOTAL_MEMORY when you do not set DS_TOTAL_MEMORY, or if you set it to an inappropriate value. For information on the algorithms, see configuration effects on memory utilization in your *IBM Informix Dynamic Server Performance Guide*.

DUMPCNT (UNIX)

onconfig.std *value*

1

if not present

1

units Number of assertion failures

range of values

Positive integers

takes effect

When the database server is shut down and restarted

utilities

onmode -wf or **onmode -wm**

refer to

- Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator's Guide*
- "onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20
- "DUMPDIR" on page 1-41
- "DUMPSHMEM (UNIX)" on page 1-42

DUMPCNT specifies the number of assertion failures for which one database server thread dumps shared memory or generates a core file by calling **gcore**. An assertion is a test of some condition or expression with the expectation that the outcome is true. For example, the following statement illustrates the concept of an assertion failure:

```
if (a != b)
    assert_fail("a != b");
```

DUMPCORE (UNIX)

onconfig.std *value*
0

range of values

0 = Do not dump core image. 1 = Dump core image.

takes effect

When the database server is shut down and restarted

refer to Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator's Guide*

DUMPCORE controls whether assertion failures cause a virtual processor to dump a core image. The core file is left in the directory from which the database server was last invoked. (The DUMPDIR parameter has no impact on the location of the core file.)

Warning: When *DUMPCORE* is set to 1, an assertion failure causes a virtual processor to dump a core image, which in turn causes the database server to abort. Set *DUMPCORE* only for debugging purposes in a controlled environment.

DUMPDIR

onconfig.std *value*

On UNIX: **\$INFORMIXDIR/tmp** On Windows: **\$INFORMIXDIR\tmp**

if not present

\$INFORMIXDIR/tmp

range of values

Any directory to which user **informix** has write access

takes effect

When the database server is shut down and restarted

refer to

- Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator's Guide*
- "DUMPCNT (UNIX)" on page 1-40
- "DUMPSHMEM (UNIX)" on page 1-42

DUMPDIR specifies a directory in which the database server dumps shared memory, **gcore** files, or messages from a failed assertion. Because shared memory can be large, set DUMPDIR to a file system with a significant amount of space. The directory to which DUMPDIR is set must exist for the server to start.

DUMPGCORE (UNIX)

onconfig.std *value*
0

range of values

0 = Do not dump **gcore**. 1 = Dump **gcore**.

takes effect

When the database server is shut down and restarted

refer to Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator's Guide*

DUMPGCORE is used with operating systems that support **gcore**. If you set DUMPGCORE, but your operating system does not support **gcore**, messages in the database server message log indicate that an attempt was made to dump a core image, but the database server cannot find the expected file. (If your operating system does not support **gcore**, set DUMPCORE instead.)

If DUMPGCORE is set, the database server calls **gcore** whenever a virtual processor encounters an assertion failure. The **gcore** utility directs the virtual processor to dump a core image to the **core.pid.cnt** file in the directory that DUMPDIR specifies and continue processing.

The **pid** value is the process identification number of the virtual processor. The **cnt** value is incremented each time that this process encounters an assertion failure. The **cnt** value can range from 1 to the value of DUMPCNT. After that, no more core files are created. If the virtual processor continues to encounter assertion failures, errors are reported to the message log (and perhaps to the application), but no further diagnostic information is saved.

DUMPSHMEM (UNIX)

onconfig.std *value*
1

range of values

0 = Do not create a shared memory dump
1 = Create a shared memory dump of all the shared memory that the database uses
2 = Create a shared memory dump that excludes the buffer pool in the resident memory

takes effect

When the database server is shut down and restarted

utilities

onmode -wf or **onmode -wm**

refer to

- Collecting diagnostic information in the chapter on consistency checking in the *IBM Informix Administrator's Guide*
- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- “onstat -o: Output shared memory contents to a file” on page 15-146
- “Running onstat Commands on a Shared Memory Dump File” on page 15-8
- “DUMPCNT (UNIX)” on page 1-40
- “DUMPDIR” on page 1-41

DUMPSHMEM indicates whether a shared memory dump is created on an assertion failure and how much memory is written to the **shmem.pid.cnt** file in the directory specified by the DUMPDIR configuration parameter. If DUMPSHMEM is set to 1, all the shared memory that the database server uses is dumped, which can result in a large file. When space is limited, set DUMPSHMEM to 2 because this setting creates a smaller shared-memory dump file.

The *pid* value is the process identification number for the virtual processor. The *cnt* value increments each time that this virtual processor encounters an assertion failure. The *cnt* value can range from 1 to the value of the DUMPCNT configuration parameter. After the value of DUMPCNT is reached, no more files are created. If the database server continues to detect inconsistencies, errors are reported to the message log (and perhaps to the application), but no further diagnostic information is saved.

DYNAMIC_LOGS

onconfig.std *value*
2

if not present
2 (Default)

range of values

0 = Turn off dynamic-log allocation.

1 = Set off the “log file required” alarm and pause to allow manual addition of a logical-log file. You can add a log file immediately after the current log file or to the end of the log file list.

2 = Turn on dynamic-log allocation. When the database server dynamically adds a log file, it sets off the “dynamically added log file” alarm.

takes effect

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

utilities

“onparams -a -d *dbspace*: Add a logical-log file” on page 13-2

refer to

- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- “LTXEHW” on page 1-62
- “LTXHWM” on page 1-62
- Logical logs in the *IBM Informix Administrator’s Guide*

If DYNAMIC_LOGS is 2, the database server automatically allocates a new log file when the next active log file contains an open transaction. Dynamic-log allocation prevents long transaction rollbacks from hanging the system.

If you want to choose the size and location of the new logical-log file, set DYNAMIC_LOGS to 1. Use the **onparams -a** command with the size (**-s**), location (**-d *dbspace***), and **-i** options to add a log file after the current log file.

Even when DYNAMIC_LOGS is turned off, you do not have the same risks as in previous database server versions. In Version 9.3 and later, if the database server hangs from a long transaction rollback, you can shut down the database server, set DYNAMIC_LOGS to 1 or 2, and then restart the database server.

Important: If you are using *Enterprise Replication* with dynamic log allocation, set *LTXEHWM* to no higher than 70.

ENCRYPT_CIPHERS

onconfig.std *value*

None

syntax ENCRYPT_CIPHERS all|allbut:<list of ciphers and modes>|cipher:mode{,cipher:mode ...}

- all

Specifies to include all available ciphers and modes, except ECB mode. For example: ENCRYPT_CIPHERS all

- allbut:<list of ciphers and modes>

Specifies to include all ciphers and modes except the ones in the list. Separate ciphers or modes with a comma. For example: ENCRYPT_CIPHERS allbut:<cbc,bf>

- cipher:mode

Specifies the ciphers and modes. Separate cipher-mode pairs with a comma. For example: ENCRYPT_CIPHERS des3:cbc,des3:ofb

takes effect

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

refer to

- “ENCRYPT_HDR” on page 1-45
- “ENCRYPT_MAC” on page 1-46
- “ENCRYPT_MACFILE” on page 1-46
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_CIPHERS configuration parameter defines all ciphers and modes that can be used by the current database session. ENCRYPT_CIPHERS is used for Enterprise Replication and High-Availability Data Replication only.

The cipher list for **allbut** can include unique, abbreviated entries. For example, **bf** can represent bf-1, bf-2, and bf-3; however, if the abbreviation is the name of an actual cipher, then only that cipher is eliminated. Therefore, **des** eliminates only the des cipher, but **de** eliminates des, des3, and desx.

Important: The encryption cipher and mode used is randomly chosen among the ciphers common between the two servers. It is strongly recommended that you do not specify specific ciphers. For security reasons, all ciphers should be allowed. If a specific cipher is discovered to have a weakness, then that cipher can be eliminated by using the **allbut** option.

The following ciphers are supported. For an updated list, see the Release Notes.

Cipher	Description
des	DES (64-bit key)
des3	Triple DES
desx	Extended DES (128-bit key)

Cipher	Description
aes	AES 128bit key
aes192	AES 192bit key
bf-1	Blow Fish (64-bit key)
bf-2	Blow Fish (128-bit key)
bf-3	Blow Fish (192-bit key)
aes128	AES 128bit key
aes256	AES 256bit key

The following modes are supported.

- ecb** Electronic Code Book (ECB)
- cbc** Cipher Block Chaining
- cfb** Cipher Feedback
- ofb** Output Feedback

All ciphers support all modes, except the **desx** cipher, which only supports the **cbc** mode.

Because **cdb** mode is considered weak, it is only included if specifically requested. It is not included in the **all** or the **allbut** list.

ENCRYPT_HDR

onconfig.std *value*

None

range of values

0 = Disables HDR encryption

1 = Enables HDR encryption

takes effect

when the server is initialized

refer to

- “ENCRYPT_CIPHERS” on page 1-44
- “ENCRYPT_MAC” on page 1-46
- “ENCRYPT_MACFILE” on page 1-46
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

ENCRYPT_HDR enables or disables HDR encryption. Enabling HDR encryption provides a secure method for transferring data from one server to another in an HDR pair. HDR encryption works in conjunction with Enterprise Replication (ER) encryption. However, it is not necessary to have ER encryption enabled for HDR encryption. HDR encryption works whether ER encryption is enabled or not. HDR and ER share the same encryption configuration parameters: ENCRYPT_CIPHERS, ENCRYPT_MAC, ENCRYPT_MACFILE and ENCRYPT_SWITCH.

ENCRYPT_MAC

onconfig.std *value*
None

range of values

One or more of the following options, separated by commas:

off = does not use MAC generation

low = uses XOR folding on all messages

medium = uses SHA1 MAC generation for all messages greater than 20 bytes long and XOR folding on smaller messages

high = uses SHA1 MAC generation on all messages

For example: ENCRYPT_MAC medium,high

takes effect

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

refer to

- “ENCRYPT_CIPHERS” on page 1-44
- “ ENCRYPT_HDR” on page 1-45
- “ENCRYPT_MACFILE”
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_MAC configuration parameter controls the level of message authentication code (MAC) generation and is used for Enterprise Replication and High-Availability Data Replication only.

The level is prioritized to the highest value. For example, if one node has a level of **high** and **medium** enabled and the other node has only **low** enabled, then the connection attempt fails. Use the **off** entry between servers only when a secure network connection is guaranteed.

ENCRYPT_MACFILE

onconfig.std *value*
None

units pathnames, up to 1536 bytes in length

range of values

One or more full path and filenames separated by commas, and the optional **builtin** keyword. For example: ENCRYPT_MACFILE /usr/local/bin/mac1.dat, /usr/local/bin/mac2.dat,builtin

takes effect

For HDR: when the database server is shut down and restarted

For Enterprise Replication: when Enterprise Replication is started

refer to

- “ENCRYPT_CIPHERS” on page 1-44

- “ ENCRYPT_HDR” on page 1-45
- “ENCRYPT_MAC” on page 1-46
- “ENCRYPT_SWITCH” on page 1-48
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_MACFILE configuration parameter specifies a list of the full path names of MAC key files and is used for Enterprise Replication and High-Availability Data Replication only.

To specify the built-in key, use the keyword **builtin**. Using the **builtin** option provides limited message verification (some validation of the received message and determination that it appears to have come from a Dynamic Server client or server). The strongest verification is done by a site-generated MAC key file.

To generate a MAC key file:

1. Execute the following command from the command line:

GenMacKey -o filename

The *filename* is the name of the MAC key file.

2. Update the ENCRYPT_MACFILE configuration parameter on participating servers to include the location of the new MAC key file.
3. Distribute the new MAC key file.

Each of the entries for the ENCRYPT_MACFILE configuration parameter is prioritized and negotiated at connect time. The prioritization for the MAC key files is based on their creation time by the **GenMacKey** utility. The **builtin** option has the lowest priority. Because the MAC key files are negotiated, you should periodically change the keys.

ENCRYPT_SMX

onconfig.std *value*

None

range of values

0, 1, 2

takes effect

When the server is restarted

refer to

- Server Multiplexer group (SMX) Connections in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_SMX configuration parameter sets the level of encryption for high-availability secondary server configurations.

Value	Explanation
0	Off. Do not encrypt.
1	On. Encrypt where possible. Encrypt SMX transactions when the database server being connected to also supports encryption.

Value	Explanation
2	On. Always encrypt. Only connections to encrypted database servers are allowed.

ENCRYPT_SWITCH

onconfig.std *value*
None

syntax ENCRYPT_SWITCH *cipher_switch_time*, *key_switch_time*

- *cipher_switch_time* specifies the minutes between cipher renegotiation
- *key_switch_time* specifies the minutes between secret key renegotiation

units minutes

range of values
positive integers

takes effect
For HDR: when the database server is shut down and restarted
For Enterprise Replication: when Enterprise Replication is started

refer to

- “ENCRYPT_CIPHERS” on page 1-44
- “ ENCRYPT_HDR” on page 1-45
- “ENCRYPT_MAC” on page 1-46
- “ENCRYPT_MACFILE” on page 1-46
- HDR Encryption Options in the *IBM Informix Administrator’s Guide*
- Using High-Availability Data Replication in the *IBM Informix Dynamic Server Enterprise Replication Guide*

The ENCRYPT_SWITCH configuration parameter defines the frequency at which ciphers or secret keys are renegotiated. The longer the secret key and encryption cipher remains in use, the more likely the encryption rules might be broken by an attacker. To avoid this, cryptologists recommend changing the secret keys on long-term connections. The default time that this renegotiation occurs is once an hour. ENCRYPT_SWITCH is used for Enterprise Replication and High-Availability Data Replication only.

Enterprise Replication Configuration Parameters

The following configuration parameters apply to Enterprise Replication. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

In addition, the ENCRYPT_CIPHERS, ENCRYPT_MAC, ENCRYPT_MACFILE and ENCRYPT_SWITCH configuration parameters apply to high-availability data replication (HDR). For more information, see the specific configuration parameter entries in this book.

Configuration Parameter
Description

CDR_DBSPACE
Specifies the dbspace where the **syscdr** database is created.

CDR_DSLOCKWAIT

Specifies the number of seconds that the Datasync (data synchronization) component waits for database locks to be released.

CDR_ENV

Sets the Enterprise Replication environment variables **CDR_LOGDELTA**, **CDR_PERFLOG**, **CRD_ROUTER**, or **CDR_RMSCALEFACT**.

CDR_EVALTHREADS

Specifies the number of grouper evaluator threads to create when Enterprise Replication starts and enables parallelism.

CDR_MAX_DYNAMIC_LOGS

Specifies the number of dynamic log file requests that Enterprise Replication can make in one server session.

CDR_NIFCOMPRESS

Specifies the level of compression that the database server uses before sending data from the source database server to the target database server.

CDR_QDATA_SBSpace

Specifies the list of up to 32 names of sbspaces that Enterprise Replication uses to store spooled transaction row data.

CDR_QHDR_DBSPACE

Specifies the location of the dbspace that Enterprise Replication uses to store the transaction record headers spooled from the send and receive queues.

CDR_QUEUEMEM

Specifies the maximum amount of memory that is used for the send and receive queues.

CDR_SERIAL

Controls generating values for SERIAL and SERIAL8 columns in tables defined for replication. Use this parameter to generate SERIAL column primary keys.

CDR_SUPPRESS_ATSRISWARN

Specifies the Datasync error and warning code numbers to be suppressed in the ATS and RIS files.

ENCRYPT_CDR

Specifies the level of Enterprise Replication encryption.

ENCRYPT_CIPHERS

Specifies the ciphers to use for Enterprise Replication encryption.

ENCRYPT_MAC

Specifies the level of message authentication coding to use with Enterprise Replication encryption.

ENCRYPT_MACFILE

Specifies the message authentication coding key files to use with Enterprise Replication encryption.

ENCRYPT_SWITCH

Defines the frequency at which ciphers and secret keys are re-negotiated for Enterprise Replication encryption.

EXPLAIN_STAT

onconfig.std *value*

1

range of values

0 to 1

takes effect

When the database server is shut down and restarted

refer to

- “onmode -Y: Dynamically change SET EXPLAIN” on page 11-22
- SET EXPLAIN, in the *IBM Informix Guide to SQL: Syntax*
- Query Plan Report, in the *IBM Informix Performance Guide*

Value	Description
0	Disables the display of query statistics
1	Enables the display of query statistics

The EXPLAIN_STAT configuration parameter enables or disables the inclusion of a Query Statistics section in the explain output file. You can generate the output file by using either the SET EXPLAIN statement of SQL or the **onmode -Y sessionid** command. When enabled, the Query Statistics section shows the Query Plan’s estimated number of rows, and the actual number of returned rows.

EXT_DIRECTIVES

onconfig.std *value*

0

range of values

0, 1, 2

takes effect

When the database server is shut down and restarted

environment variable

IFX_EXTDIRECTIVES

refer to

- Environment variables and information about the **sysdirectives** system catalog table, in the *IBM Informix Guide to SQL: Reference*
- SQL directives, in the *IBM Informix Guide to SQL: Syntax*
- Using external optimizer directives, in the *IBM Informix Performance Guide*

The EXT_DIRECTIVES configuration parameter enables or disables the use of external SQL directives. Enable external directives by using the EXT_DIRECTIVES configuration parameter in combination with the client-side **IFX_EXTDIRECTIVES** environment variable as follows:

Value	Explanation
0 (default)	Off. The directive cannot be enabled even if IFX_EXTDIRECTIVES is on.
1	On. The directive can be enabled for a session if IFX_EXTDIRECTIVES is on.

Value	Explanation
2	On. The directive can be used even if IFX_EXTDIRECTIVES is not set.

The setting of the **IFX_EXTDIRECTIVES** environment variable overrides the setting of the **EXT_DIRECTIVES** configuration parameter. If you do not set the **IFX_EXTDIRECTIVES** environment variable, all sessions for a client inherit the database server configuration for processing external directives.

EXTSHMADD

onconfig.std *value*
8192

range of values
1024 through 524,288

units Kilobytes

takes effect
When the database server is shut down and restarted

utilities
onstat -g seg

EXTSHMADD specifies the size of extension virtual segments that you add. Other virtual segment additions are based on the size that is specified in the SHMADD configuration parameter.

FAILOVER_CALLBACK

onconfig.std *value*
None

default value
None

range of values
Pathname

takes effect
When the database server is shut down and restarted

The database server executes the script specified by **FAILOVER_CALLBACK** when a database server transitions from a secondary server to a primary or standard server. Set **FAILOVER_CALLBACK** to the full pathname of the script.

FASTPOLL

onconfig.std *value*
1

range of values
0 = Disables fast polling. 1 = Enables fast polling.

takes effect
When the database server is shut down and restarted

Use the FASTPOLL configuration parameter to enable or disable fast polling of your network. FASTPOLL is a platform-specific configuration parameter.

FILLFACTOR

onconfig.std *value*
90

units Percent

range of values
1 through 100

takes effect

When the index is built. Existing indexes are not changed. To use the new value, the indexes must be rebuilt.

refer to "Structure of B-Tree Index Pages" on page 4-16

FILLFACTOR specifies the degree of index-page fullness. A low value provides room for growth in the index. A high value compacts the index. If an index is full (100 percent), any new inserts result in splitting nodes. You can also set the FILLFACTOR as an option on the CREATE INDEX statement. The setting on the CREATE INDEX statement overrides the ONCONFIG file value.

HA_ALIAS

onconfig.std *value*
None

takes effect

When the database server is shut down and restarted

refer to

- "onmode -wf, -wm: Dynamically change certain configuration parameters" on page 11-20

The value specified by HA_ALIAS must be one of the values specified in the DBSERVERNAME or DBSERVERALIASES configuration parameters, and must refer to a server whose connection type is a TCP network protocol.

The value specified by HA_ALIAS is optional and represents the name by which the server is known within a high-availability cluster. Valid values are the same as for DBSERVERNAME, except that the number of listener threads is not supported.

When a secondary server connects to a primary server, the secondary server sends the name of a network alias that can be used in case of failover. The setting of HA_ALIAS is used to describe which network alias will be sent.

Using the **onmode -wf** or **onmode -wm** commands, the parameter will take effect immediately.

HETERO_COMMIT

onconfig.std *value*
0

range of values

1 = Enable heterogeneous commit. 0 = Disable heterogeneous commit.

takes effect

When the database server is shut down and restarted

refer to

- Heterogeneous commit protocol, in the chapter on multiphase commit protocols in the *IBM Informix Administrator's Guide*
- *IBM Informix Enterprise Gateway Manager User Manual*

The HETERO_COMMIT configuration parameter specifies whether or not the database server is prepared to participate with IBM Informix Gateway products in heterogeneous commit transactions. Setting HETERO_COMMIT to 1 allows a single transaction to update *one* non-Informix database (accessed with any of the Gateway products) and one or more Informix databases.

If HETERO_COMMIT is 0, a single transaction can update databases as follows:

- One or more Informix databases and no non-Informix databases
- One non-Informix database and no Informix databases

You can read data from any number of Informix and non-Informix databases, regardless of the setting of HETERO_COMMIT.

IFX_EXTEND_ROLE

onconfig.std *value*

1

range of values

1 or 0n (default) = Enables the EXTEND role so that administrators can grant privileges to a user to create or drop a UDR that has the EXTERNAL clause.

0 or 0ff = Disables the EXTEND role so that any user can register an external routine.

refer to Information on security for external routines in the *IBM Informix Security Guide*

Your database system administrator (DBSA), by default user **informix**, can use the IFX_EXTEND_ROLE parameter in the **onconfig** file to control which users are authorized to register DataBlade modules or external user-defined routines (UDRs).

IFX_FOLDVIEW

onconfig.std *value*

0

range of values

0 or 0ff (default) = Disables view folding

1 or 0n = Enables view folding

refer to Information on security for external routines in the *IBM Informix Administrator's Guide*

IFX_FOLDVIEW enables or disables view folding. For certain situations where a view is involved in a query, view folding can significantly improve the performance of the query. In these cases, views are folded into a parent query instead of the query results being put into a temporary table.

The following types of queries can take advantage of view folding:

- Views that contain a UNION ALL clause and the parent query has a regular join, Informix join, ANSI join, or an ORDER BY clause
- Views with multiple table joins where the main query contains Informix or ANSI type outer joins

A temporary table is created and view folding is not performed for the following types of queries that perform a UNION ALL operation involving a view:

- The view has one of the following clauses: AGGREGATE, GROUP BY, ORDER BY, UNION, DISTINCT, or OUTER JOIN (either Informix or ANSI type).
- The parent query has a UNION or UNION ALL clause.

ISM_DATA_POOL and ISM_LOG_POOL

The ISM_DATA_POOL and ISM_LOG_POOL parameters control where IBM Informix Storage Manager stores backed-up data and logical logs. For information on these parameters, see the *IBM Informix Backup and Restore Guide* or the *IBM Informix Storage Manager Administrator's Guide*.

Java Configuration Parameters

The following configuration parameters allow you to use J/Foundation, which incorporates an embedded Java™ virtual machine on the database server. For more information on these parameters, see *J/Foundation Developer's Guide*.

Configuration Parameter Description

AFCRASH

When the 0x10 bit is on for AFCRASH, all the messages that the Java Virtual Machine generates are logged into the **JVM_vpid** file, where *vpid* is the process ID of the Java virtual processor. This file is stored in the directory where the JVPLOG file is stored.

JVPDEBUG

When set to 1, writes tracing messages to the JVPLOG file

JVPHOME

Directory where the classes of the IBM Informix JDBC Driver are installed

JVPLOGFILE

Absolute pathname for your Java VP log files

JVPPROFILE

Absolute pathname for the Java VP properties file

JVPJAVAVM

Libraries to use for the Java Virtual Machine (JVM)

JVPJAVAHOME

Directory where the Java Runtime Environment (JRE) for the database server is installed

JVMTHREAD

Thread package (green or native) to use for the JVM

JVPJVALIB

Path from **JVPJAVAHOME** to the location of the Java VM libraries

JVPCLASSPATH

Initial Java class path setting

VPCLASS jvp

Number of Java virtual processors that the database server should start.
(See “VPCLASS” on page 1-109.)

LIMITNUMSESSIONS

onconfig.std *value*

None

syntax LIMITNUMSESSIONS *maximum_number_of_sessions,print_warning*

separators

Comma

range of values

maximum_number_of_sessions = 0 to 2,097,152 (2*1024*1024). The default is 0.

print_warning = 0 (off) or 1 (on). The default for this optional value is 0.

takes effect

When the database server is shut down and restarted

utilities

onmode -wf or **onmode -wm**

The LIMITNUMSESSIONS configuration parameter defines the maximum number of sessions that you want connected to Dynamic Server. If you specify a maximum number, you can also specify whether you want Dynamic Server to print messages to the **online.log** file when the number of sessions approaches the maximum number.

If the print_warning is set to 1, a warning is triggered when the number of sessions is greater than or equal to 95% of the maximum_number_of_sessions value. If print_warning is set to zero, or if it is not set, no warning is issued. No new user sessions can be opened after the maximum_number_of_sessions limit is reached.

The DBSA can begin new sessions, even if the LIMITNUMSESSIONS configuration parameter is enabled and sessions are being restricted because of this limit.

Distributed queries against a server are counted against the limit.

If the maximum_number_of_sessions value for the LIMITNUMSESSIONS configuration parameter is set to 0, or if it is not set, there is no limit to the number of sessions that can connect to the server.

The following example specifies that you want a maximum of 100 sessions to connect to the server and you want to print a warning message when the number of connected sessions approaches 100.

```
LIMITNUMSESSIONS 100,1
```

The settings in this example cause a warning to be printed when more than 94 sessions are concurrently connected. Only a member of the DBSA group can start a new session when 100 sessions are already connected.

Use **onmode -wf** or **onmode -wm**, or the equivalent SQL Admin API ONMODE commands, to dynamically increase or temporarily disable the LIMITNUMSESSIONS setting. Use this configuration parameter to allow administrative utilities to run if the database server is reaching the the maximum_number_of_sessions limit.

If the LIMITNUMSESSIONS configuration parameter is enabled and sessions are restricted because of this limit, DBSA users can still initiate new sessions.

The LIMITNUMSESSIONS configuration parameter is not intended to be used as a means to adhere to license agreements.

LISTEN_TIMEOUT

onconfig.std *value*
60

Units Seconds

takes effect

When the database server is stopped and restarted

utilities

onmode -wf onmode-wm

refer to IBM Informix Security Guide

LISTEN_TIMEOUT specifies the number of seconds the server waits for a connection. It can be set to a lower number to guard against faulty connection requests that might indicate a Denial of Service attack. See also information about the MAX_INCOMPLETE_CONNECTIONS configuration parameter on page “MAX_INCOMPLETE_CONNECTIONS” on page 1-64.

Depending on the machine capability of holding the threads (in number), you can configure MAX_INCOMPLETE_CONNECTIONS to a higher value and depending on the network traffic, you can set LISTEN_TIMEOUT to a lower value to reduce the chance that an attack can reach the maximum limit.

Both the LISTEN_TIMEOUT and the MAX_INCOMPLETE_CONNECTIONS configuration parameters can be changed using the **onmode -wf** option or superseded for a session with the **onmode -wm** option. For more information about **onmode**, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

LOCKS

onconfig.std *value*
20000

units Number of locks in the internal lock table

range of values

2,000 through 8,000,000 for 32-bit database servers
2,000 through 500,000,000 for 64-bit database servers

takes effect

When the database server is shut down and restarted

utilities

onstat -k (see “onstat -k: Print active lock information” on page 15-141.)

refer to

- The memory and locking chapters in your *IBM Informix Performance Guide*
- The shared memory chapter in the *IBM Informix Administrator's Guide*

LOCKS specifies the initial size of the lock table. The lock table holds an entry for each lock. If the number of locks allocated exceeds the value of LOCKS, the database server increases the size of the lock table. The lock table can be increased a maximum of 99 times.

The database server increases the size of the lock table by attempting to double the lock table on each increase. However, the amount added during each increase is limited to a maximum value. For 32-bit platforms, a maximum of 100,000 locks can be added during each increase. Therefore, the total maximum locks allowed for 32-bit platforms is 8,000,000 (maximum number of starting locks) + (99 (maximum number of dynamic lock table extensions) × 100,000 (maximum number of locks added per lock table extension)). For 64-bit platforms, a maximum of 1,000,000 locks can be added during each increase. Therefore, the total maximum locks allowed is 500,000,000 (maximum number of starting locks) + (99 (maximum number of dynamic lock table extensions) × 1,000,000 (maximum number of locks added per lock table extension)).

With the initial lock table stored in resident memory and each additional lock stored in virtual memory, locks can become a resource drain if you have a limited amount of shared memory. The amount of storage occupied by a single lock ranges from 100 to 200 bytes, depending on the word size and platform.

Tip: When you drop a database, a lock is acquired and held on each table in the database until the database is dropped. For more information on the DROP DATABASE statement, see the *IBM Informix Guide to SQL: Syntax*.

LOGBUFF

onconfig.std *value*

64

units Kilobytes

range of values

32 kilobytes through (32767 * page size / 1024) kilobytes

takes effect

When the database server is shut down and restarted

utilities

onstat -l buffer field, second section. See “onstat -l: Print physical and logical log information” on page 15-142.

refer to Logical-log buffer, in the shared-memory chapter of the *IBM Informix Administrator's Guide*

LOGBUFF specifies the size in kilobytes for the three logical-log buffers in shared memory. Triple buffering permits user threads to write to the active buffer while one of the other buffers is being flushed to disk. If flushing is not complete by the time the active buffer fills, the user thread begins writing to the third buffer.

If you are using RTO_SERVER_RESTART, a LOGBUFF value of 256 kilobytes is recommended. If the LOGBUFF value is less than 256 kilobytes, a warning

message displays when you restart the server. Otherwise, set LOGBUFF to 32 kilobytes for standard workloads or 64 kilobytes for heavy workloads. Choose a value for LOGBUFF that is evenly divisible by the page size. If the value of LOGBUFF is not evenly divisible by the page size, the database server rounds down the size to the nearest value that is evenly divisible by the page size.

If you log user data in smart large objects, increase the size of the log buffer to make the system more efficient. The database server logs only the portion of a smart-large-object page that changed.

Important: The database server uses the LOGBUFF parameter to set the size of internal buffers that are used during recovery. If you set LOGBUFF too high, the database server can run out of memory and shut down during recovery.

To set the system page size, use one of the utilities listed in “System Page Size” on page 1-21.

LOGFILES

onconfig.std *value*

6

if not present

6

units Number of logical-log files

range of values

3 through 32,767 (integers only)

takes effect

During disk initialization and when you add a new log file. You add a new log with one of the following utilities.

utilities

onparams (see “In This Chapter” on page 13-1)

refer to The following topics in the *IBM Informix Administrator’s Guide*:

- Size of logical-log files, in the chapter on the logical log
- Adding or dropping a logical-log file, in the chapter on managing the logical log

LOGFILES specifies the number of logical-log files that the database server creates during disk initialization. To change the number of logical-log files, add or drop logical-log files.

If you use ISA or **onparams** to add or drop log files, the database server automatically updates LOGFILES.

LOG_INDEX_BUILDS

onconfig.std *value*

None

if not present

0 (disabled)

range of values

0, 1 (0 = disable, 1 = enable)

takes effect

When the database server is stopped and restarted

utilities

onmode -wf onmode -wm

refer to The following topics in the *IBM Informix Administrator's Reference*:

- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- Chapter 11, “The onmode Utility,” on page 11-1

LOG_INDEX_BUILDS is used to enable or disable index page logging. If LOG_INDEX_BUILDS is enabled, logical log file space consumption will increase, depending on the size of the indexes. This might lead to logical log file backups being required more frequently. Messages are written to the online.log file when index page logging status changes.

Using **onmode -wm** enables or disables index page logging for the current session only, and does not affect the setting in the onconfig file. If the server is stopped and restarted, the setting in the onconfig file determines whether index page logging is enabled. Therefore, enabling index page logging using **onmode -wm** is not recommended when using RS secondary servers; instead, use **onmode -wf** to update the onconfig file, so that index page logging is enabled after restarting the server. Index page logging is a requirement when using RS secondary servers.

LOGSIZE

onconfig.std *value*

10000

if not present

10000

units Kilobytes

range of values

Minimum = 200 Maximum = (ROOTSIZE - PHYSFILE - 512 - (63 * ((pagesize)/1024))) / LOGFILES

The *pagesize* value is platform dependent.

takes effect

When the database server is shut down and restarted. The size of log files added after shared memory is initialized reflects the new value, but the size of existing log files does not change.

utilities

onparams See “ onparams -p: Change physical-log parameters” on page 13-3.

refer to The following topics in the *IBM Informix Administrator's Guide*:

- Size of the logical log and logging smart large objects, in the chapter on the logical log
- Changes to LOGSIZE or LOGFILES, in the chapter on managing logical logs
- “LTXHWM” on page 1-62

LOGSIZE specifies the size that is used when logical-log files are created. It does not change the size of existing logical-log files. The total logical-log size is LOGSIZE * LOGFILES.

To verify the page size that the database server uses on your platform, use one of the utilities listed in “System Page Size” on page 1-21.

LOGSIZE for Smart Large Objects

If you declare logging for a smart-large-object column, you must ensure that the logical log is considerably larger than the amount of data logged during inserts or updates.

Important: The database server cannot back up open transactions. If many transactions are active, the total logging activity should not force open transactions to the log backup files. For example, if your log size is 1000 kilobytes and the high-watermark is 60 percent, do not use more than 600 kilobytes of the logical log for the smart-large-object updates. The database server starts rolling back the transaction when it reaches the high-watermark of 600 kilobytes.

LTAPEBLK

onconfig.std *value*

UNIX: 32Windows: 16

units Kilobytes

range of values

Values greater than (*page size*/1024)

To obtain the page size, see the commands listed in “System Page Size” on page 1-21

takes effect

For **ontape**: When you execute **ontape**

For **onload** and **onunload**: When the database server is shut down and restarted

refer to

- Using **ontape**, in the *IBM Informix Backup and Restore Guide*
- Using **onload** and **onunload**, in the *IBM Informix Migration Guide*
- “TAPEBLK” on page 1-103

LTAPEBLK specifies the block size of the device to which the logical logs are backed up when you use **ontape** for dbspace backups. LTAPEBLK also specifies the block size for the device to which data is loaded or unloaded when you use the -l option of **onload** or **onunload**. If you are using **onload** or **onunload**, you can specify a different block size at the command line.

Specify LTAPEBLK as the largest block size permitted by your tape device. The database server does not check the tape device when you specify the block size. Verify that the LTAPEDEV tape device can read the block size that you specify. If not, you might not be able to read from the tape.

UNIX Only:

The UNIX **dd** utility can verify that the LTAPEDEV tape device can read the block size. It is available with most UNIX systems.

LTAPEDEV

onconfig.std *value*

On UNIX: **/dev/tapedev** On Windows: **NUL**

if not present

On UNIX: **/dev/null** On Windows: **NUL**

takes effect

For **ontape**: when the database server is shut down and restarted, if set to **/dev/null** on UNIX or **nul** on Windows. When you execute **ontape**, if set to a tape device.

For **onload** and **onunload**: when the database server is shut down and restarted

refer to

- How to set and change the LTAPEDEV value for **ontape** and how LTAPEDEV affects ON-Bar, in the *IBM Informix Backup and Restore Guide*
- Using **onload** or **onunload**, in the *IBM Informix Migration Guide*
- “TAPEDEV” on page 1-103

LTAPEDEV specifies the device or directory file system to which the logical logs are backed up when you use **ontape** for backups.

LTAPEDEV also specifies the device to which data is loaded or unloaded when you use the **-l** option of **onload** or **onunload**. If you are using LTAPEDEV to specify a device for **onunload** or **onload**, the same information for TAPEDEV is relevant for LTAPEDEV.

Warning: Do not set LTAPEDEV to **/dev/null** or **nul** when you use ON-Bar to back up logical logs.

LTAPESIZE

onconfig.std *value*

0

units Kilobytes

range of values

0 through 2,097,151

takes effect

For **ontape**: when you execute **ontape**

For **onload** and **onunload**: when the database server is shut down and restarted

refer to

- Using **ontape**, in the *IBM Informix Backup and Restore Guide*
- Using **onload** or **onunload**, in the *IBM Informix Migration Guide*
- “TAPESIZE” on page 1-104

LTAPESIZE specifies the maximum tape size of the device to which the logical logs are backed up when you use **ontape** for backups. LTAPESIZE also specifies the

maximum tape size of the device to which data is loaded or unloaded when you use the **-I** option of **onload** or **onunload**. If you are using **onload** or **onunload**, you can specify a different tape size on the command line. If you want to use the full capacity of a tape, set **LTAPESIZE** to 0.

LTXEHWM

onconfig.std *value*
80

if not present

90 (if **DYNAMIC_LOGS** is set to 1 or 2) 60 (if **DYNAMIC_LOGS** is set to 0)

units Percent

range of values

LTXHWM through 100

takes effect

When the database server is shut down and restarted

refer to

- “**DYNAMIC_LOGS**” on page 1-43
- “**LTXHWM**”
- “**onmode -wf, -wm: Dynamically change certain configuration parameters**” on page 11-20
- Setting high-watermarks for rolling back long transactions, in the chapter on managing logical logs in the *IBM Informix Administrator's Guide*

A *transaction is long* if it is not committed or rolled back when it reaches the long-transaction high-watermark. **LTXEHWM** specifies the *long-transaction, exclusive-access, high-watermark*. When the logical-log space reaches the **LTXEHWM** threshold, the long transaction currently being rolled back is given *exclusive* access to the logical log.

If your system runs out of log space before the rollback completes, lower the **LTXEHWM** value.

If you do not want too many logical logs to be added, **LTXEHWM** should be set to a smaller value (around 60). If dynamic logging is turned off (**DYNAMIC_LOGS** = 0), **LTXEHWM** should be set lower (around 50) to avoid running out of logical space.

Tip: To allow users to continue to access the logical logs, even during a long transaction rollback, set **LTXEHWM** to 100. Set **DYNAMIC_LOGS** to 1 or 2 so that the database server can add a sufficient number of log files to prevent long transactions from hanging and to allow long transactions to roll back.

LTXHWM

onconfig.std *value*
70

if not present

80 (if **DYNAMIC_LOGS** is set to 1 or 2) 50 (if **DYNAMIC_LOGS** is set to 0)

units Percent

range of values
1 through 100

takes effect
When the database server is shut down and restarted

refer to

- “ DYNAMIC_LOGS” on page 1-43
- “LTXEHWM” on page 1-62
- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- Setting high-watermarks for rolling back long transactions, in the chapter on managing logical logs in the *IBM Informix Administrator’s Guide*

LTXHWM specifies the long-transaction high-watermark. The *long-transaction high-watermark* is the percentage of available log space that, when filled, triggers the database server to check for a long transaction. When the logical-log space reaches the LTXHWM threshold, the database server starts rolling back the transaction. If you decrease the LTXHWM value, increase the size or number of log files to make rollbacks less likely.

If DYNAMIC_LOGS is set to 1 or 2, the database server can add a sufficient number of log files to complete the transactions or to prevent rollbacks from hanging when you have long transactions.

If you do not want too many logical logs to be added, LTXHWM should be set to a smaller value (around 60). If dynamic logging is turned off (DYNAMIC_LOGS = 0), LTXHWM should be set lower (around 50) to avoid running out of logical space.

Warning: If you set both LTXHWM and LTXEHWM to 100, long transactions are never aborted. Although you can use this configuration to your advantage, you should set LTXHWM to below 100 for normal database server operations.

If you set LTXHWM to 100, the database server issues a warning message:
LTXHWM is set to 100%. This long transaction high water mark will never be reached. Transactions will not be aborted automatically by the server, regardless of their length.

If the transaction hangs, follow the instructions for recovering from a long transaction hang, in the chapter on managing logical-log files in the *IBM Informix Administrator’s Guide*.

MAX_FILL_DATA_PAGES

onconfig.std *value*
0

units Integer

range of values
0 or 1

takes effect
When the database server is stopped and restarted

refer to Reducing disk space through additional rows per page in tables with variable-length rows in the *IBM Informix Performance Guide*

Set the MAX_FILL_DATA_PAGES value to 1 to allow more rows to be inserted per page in tables that have variable-length rows. This setting can reduce disk space, make more efficient use of the buffer pool, and reduce table scan times.

If MAX_FILL_DATA_PAGES is enabled, the server will add a new row to a recently modified page with existing rows if adding the row leaves at least 10 percent of the page free for future expansion of all the rows in the page. If MAX_FILL_DATA_PAGES is not set, the server will add the row only if there is sufficient room on the page to allow the new row to grow to its maximum length.

A possible disadvantage of enabling MAX_FILL_DATA_PAGES and allowing more variable-length rows per page is that the server might store rows in a different physical order. Also, as the page fills, updates made to the variable-length columns in a row could cause the row to expand so it no longer completely fits on the page. This causes the server to split the row onto two pages, increasing the access time for the row.

To take advantage of this setting, existing tables with variable-length rows must be reloaded or existing pages must be modified, followed by further inserts.

MAX_INCOMPLETE_CONNECTIONS

onconfig.std *value*
1024

units Number of listener threads

takes effect

When the database server is stopped and restarted

utilities

onmode -wf onmode-wm

refer to *IBM Informix Security Guide*

Use MAX_INCOMPLETE_CONNECTIONS to specify the maximum number of incomplete connections in a session. After this number is reached, an error message is written in the online message log stating that the server might be under a Denial of Service attack. See also information about the LISTEN_TIMEOUT configuration parameter on page “LISTEN_TIMEOUT” on page 1-56.

Depending on the machine capability of holding the threads (in number), you can configure MAX_INCOMPLETE_CONNECTIONS to a higher value and depending on the network traffic, you can set LISTEN_TIMEOUT to a lower value to reduce the chance that an attack can reach the maximum limit.

Both the MAX_INCOMPLETE_CONNECTIONS and the LISTEN_TIMEOUT configuration parameters can be changed using the **onmode -wf** option or superseded for a session with the **onmode -wm** option. For more information about **onmode**, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

MAX_PDQPRIORITY

onconfig.std *value*
100

if not present
100

range of values
0 through 100

takes effect
On all user sessions

utilities
onmode -D onstat -g mgm (See “onstat -g mgm: Print MGM resource information ” on page 15-74.)

refer to

- The chapter on using PDQ, in the *IBM Informix Performance Guide*
- “ onmode -D, -M, -Q, -S: Change decision-support parameters” on page 11-10

MAX_PDQPRIORITY limits the PDQ resources that the database server can allocate to any one DSS query. MAX_PDQPRIORITY is a factor that is used to scale the value of PDQ priority set by users. For example, suppose that the database administrator sets MAX_PDQPRIORITY to 80. If a user sets the PDQPRIORITY environment variable to 50 and then issues a query, the database server silently processes the query with a PDQ priority of 40.

You can use the **onmode** utility to change the value of MAX_PDQPRIORITY while the database server is online.

In Dynamic Server, PDQ resources include memory, CPU, disk I/O, and scan threads. MAX_PDQPRIORITY lets the database administrator run decision support concurrently with OLTP, without a deterioration of OLTP performance. However, if MAX_PDQPRIORITY is too low, the performance of decision- support queries can degrade.

You can set MAX_PDQPRIORITY to one of the following values.

Value Database Server Action

- | | |
|-----|---|
| 0 | Turns off PDQ. DSS queries use no parallelism. |
| 1 | Fetches data from fragmented tables in parallel (parallel scans) but uses no other form of parallelism. |
| 100 | Uses all available resources for processing queries in parallel. |

number

An integer between 0 and 100. Sets the percentage of the user-requested PDQ resources actually allocated to the query.

MaxConnect Configuration Parameters

Before you start IBM Informix MaxConnect, you need to specify the following configuration parameters in the IMCCONFIG file. This file contains both start-time and runtime parameters.

Configuration Parameter
Description

IMCLOG

Specifies the pathname of the Informix MaxConnect log file.

IMCTransports

Specifies the number of TCP network connections (transports) between Informix MaxConnect and the database server.

IMCWORKERDELAY

Determines the time that worker threads wait to accumulate packets before they perform an aggregated send.

IMCWORKERTHREADS

Specifies the number of worker threads for Informix MaxConnect.

Informix MaxConnect uses the following environment variables. For more information, see the section on the configuration file in the *IBM Informix MaxConnect User's Guide*:

- **INFORMIXDIR**
- **INFORMIXSERVER**
- **INFORMIXSQLHOSTS**
- **IMCADMIN**
- **IMCCONFIG**
- **IMCSERVER**

MIRROR

onconfig.std *value*
0

range of values

0 = disable mirroring 1 = enable mirroring

takes effect

When the database server is shut down and restarted

utilities

onstat -d *flags* field (see “onstat -d: Print chunk information” on page 15-19.)

refer to The following topics in the *IBM Informix Administrator's Guide*:

- Mirroring critical data in the chapter on where is data stored
- Enabling mirroring in the chapter on using mirroring

The MIRROR parameter indicates whether mirroring is enabled for the database server. It is recommended that you mirror the root dbspaces and the critical data as part of initialization. Otherwise, leave mirroring disabled. If you later decide to add mirroring, you can edit your configuration file to change the parameter value.

You do not have to set the MIRROR configuration parameter to the same value on both database servers in the high-availability data-replication pair. You can enable or disable mirroring on either the primary or the secondary database server independently. Do not set the MIRROR configuration parameter to 1 unless you are using mirroring.

MIRROROFFSET

onconfig.std *value*
0

units Kilobytes

range of values

Any value greater than or equal to 0

takes effect

When the database server is shut down and restarted

refer to Mirroring the root dbspace during initialization, in the chapter on using mirroring in the *IBM Informix Administrator's Guide*

In Dynamic Server, MIRROROFFSET specifies the offset into the disk partition or into the device to reach the chunk that serves as the mirror for the initial chunk of the root dbspace.

MIRRORPATH

onconfig.std *value*

On UNIX: `$INFORMIXDIR/tmp/demo_on.root_mirror` On Windows: None

range of values

65 or fewer characters

takes effect

When the database server is shut down and restarted

refer to The following material in the *IBM Informix Administrator's Guide*:

- Mirroring the root dbspace during initialization, in the chapter on using mirroring
- Using links, in the chapter on managing disk space

MIRRORPATH specifies the full pathname of the mirrored chunk for the initial chunk of the root dbspace. MIRRORPATH should be a link to the chunk pathname of the actual mirrored chunk for the same reasons that ROOTPATH is specified as a link. Similarly, select a short pathname for the mirrored chunk.

Setting Permissions (UNIX)

You must set the permissions of the file that MIRRORPATH specifies to 660. The owner and group must both be **informix**.

If you use raw disk space for your mirror chunk on a UNIX platform, it is recommended that you define MIRRORPATH as a pathname that is a link to the initial chunk of the mirror dbspace, instead of entering the actual device name for the initial chunk.

MSG_DATE

onconfig.std *value*

Not in the onconfig.std template file

range of values

0 = OFF (the default)
1 = ON

takes effect

When the database server is shut down and restarted

utilities

onmode -wf or **onmode -wm**

refer to “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

MSG_DATE inserts a date in the MM/DD/YY format to the beginning of each message printed to the online log.

In the following example MSG_DATE is set to 1 (ON).

```
04/10/08 10:26:08 Value of MSG_DATE has been changed to 1.  
04/10/08 10:27:35 Value of MSG_DATE has been changed to 1.
```

MSGPATH

onconfig.std *value*

On UNIX: **\$INFORMIXDIR/tmp/online.log** On Windows: **online.log**

range of values

Pathname

takes effect

When the database server is shut down and restarted

utilities

onstat -m to view the message log (For more information, see “onstat -l: Print physical and logical log information” on page 15-142.)

refer to Message log, in the chapter on overview of database server administration in the *IBM Informix Administrator's Guide*

MSGPATH specifies the full pathname of the message-log file. The database server writes status messages and diagnostic messages to this file during operation.

If the file that MSGPATH specifies does not exist, the database server creates the file in the specified directory. If the directory that MSGPATH specifies does not exist, the database server sends the messages to the system console.

If the file that MSGPATH specifies does exist, the database server opens it and appends messages to it as they occur.

MULTIPROCESSOR

onconfig.std *value*

0

if not present

Platform dependent

range of values

0 = No multiprocessor 1 = Multiprocessor available

takes effect

When the database server is shut down and restarted

refer to CPU virtual processors, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*

If MULTIPROCESSOR is set to 0, the parameters that set processor affinity are ignored. MULTIPROCESSOR specifies whether the database server performs locking in a manner that is suitable for a single-processor computer or a multiprocessor computer.

NETTYPE

syntax NETTYPE protocol,poll_threads,connections,VP_class Specify the *protocol* as **iihpp** where:

ii=[ipc|soc|tli] **ppp**=[shm|str|tcp|spx|imc|ssl]

The *protocol* value is required. You cannot use any white space in the fields, but you can omit trailing commas.

onconfig.std values

On UNIX: ipcshm,1,50,CPU On Windows: None

if not present

protocol:

On UNIX: **protocol** field from the **sqlhosts** file (with or without the database server prefix of **on**, **ol**, or **dr**)

On Windows: onsoctcp

number of poll_threads: 1 *number of connections*: 50 *VP_class*: NET for DBSERVERALIASES; CPU for DBSERVERNAME

separators

Commas

range of values

number of poll_threads:

On UNIX: If *VP_class* is NET, a value greater than or equal to 1 If *VP_class* is CPU, 1 through num_cpu_vps

On Windows: Any value greater than or equal to 1

number of connections: 1 through 32,767

VP_class: CPU = CPU VPs (on UNIX) NET = Network VPs

takes effect

When the database server is shut down and restarted

utilities

onstat -g nsc (see “onstat -g Monitoring Options” on page 15-25.) **onstat -g nss onstat -g nta**

refer to The following sections in the *IBM Informix Administrator's Guide*:

- Connecting to IBM data server clients
- Allocating poll threads for an interface/protocol combination
- Setting the size of the DRDA communications buffer
- Displaying DRDA thread and session information
- Network protocol entry, in the chapter on client/server communications
- Multiplexed connections, in the chapter on client/server communications
- Network virtual processors, in the chapter on virtual processors
- Should poll threads run on CPU or network virtual processors, in the chapter on virtual processors

- Monitoring and tuning the number of poll threads and connections, in the *IBM Informix Performance Guide*

Configuring Informix MaxConnect in *IBM Informix MaxConnect User's Guide*

Configuring a server instance for Secure Sockets Layer (SSL) connections, in the *IBM Informix Security Guide*

The **NETTYPE** parameter usually provides tuning options for the protocols that **dbservername** entries define in the **sqlhosts** file or registry.

Each **dbservername** entry in the **sqlhosts** file or registry is defined on either the **DBSERVERNAME** parameter or the **DBSERVERALIASES** parameter in the **ONCONFIG** file.

The **NETTYPE** configuration parameter describes a network connection as follows:

- The protocol (or type of connection)
- The number of poll threads assigned to manage the connection
- The expected number of concurrent connections
- The class of virtual processor that will run the poll threads

You can specify a **NETTYPE** parameter for each protocol that you want the database server to use. The following example illustrates **NETTYPE** parameters for two types of connections to the database server: a shared memory connection for local clients, and a network connection that uses sockets:

```
NETTYPE ipcshm,3,,CPU
NETTYPE soctcp,,20,NET
```

The **NETTYPE** parameter for the shared-memory connection (**ipcshm**) specifies three poll threads to run in CPU virtual processors. The number of connections is not specified, so it is set to 50. For **ipcshm**, the number of poll threads correspond to the number of memory segments.

The **NETTYPE** parameter for the sockets connection (**soctcp**) specifies that only 20 simultaneous connections are expected for this protocol and that one poll thread (because the number of poll threads is not specified) will run in a network virtual processor (in this case, **NET**).

Protocol

The protocol entry is the same as the **nettype** field in the **sqlhosts** file or registry, except that the database server prefix of **on**, **ol** or **dr** is optional. The first three characters of the protocol entry specify the interface type, and the last three characters specify the IPC mechanism or the network protocol.

Number of Poll Threads

This field specifies the number of poll threads for a specific protocol. The default value of *poll_threads* is 1.

If your database server has a large number of connections, you might be able to improve performance by increasing the number of poll threads. In general, each poll thread can handle approximately 200 to 250 connections.

Number of Connections

This field specifies the maximum number of connections per poll thread that can use this protocol at the same time. The default value of *connections* is 50. If only a few connections will be using a protocol concurrently, you might save memory by explicitly setting the estimated number of connections.

Use this formula to calculate the maximum number of connections expected. For shared memory (**ipcshm**), double the number of connections.

$\text{connections} = \text{max_connections} / \text{poll threads}$

UNIX Only:

The following example specifies 3 poll threads and 20 connections for a total of 60 shared-memory connections.

```
NETTYPE ipcshm,3,20,CPU
```

For all net types other than **ipcshm**, the poll threads dynamically reallocate resources to support more connections, as needed. Avoid setting the value for the number of concurrent connections to much higher than you expect. Otherwise, you might waste system resources.

Windows Only:

On Windows, the number of connections per poll thread is used for **ipcshm** connections. Other protocols ignore this value. Use NET virtual processors to run the poll threads.

Class of Virtual Processor

You can set the *VP_class* entry to specify either CPU or NET. However, the combined number of poll threads defined with the CPU VP class for all net types cannot exceed the maximum number of CPU VPS. You should carefully distinguish between poll threads for network connections and poll threads for shared memory connections, which should run one per CPU virtual processor. TCP connections should only be in network virtual processors, and you should only have the minimum needed to maintain responsiveness. Shared memory connections should only be in CPU virtual processors and should run in every CPU virtual processor.

Note: If you use the VP classes *tli*, *shm*, *str*, or *soc* in the settings for the VPCLASS configuration parameter, you must use the class of virtual processor class NET for the NETTYPE configurator parameter. For more information on the VPCLASS configuration parameter, see “VPCLASS” on page 1-109.

For more advice on whether to run the poll threads on CPU or NET virtual processors, refer to the chapter on virtual processors in the *IBM Informix Administrator's Guide*.

Default Values

It is recommended that you use NETTYPE to configure each of your connections. However, if you do not use NETTYPE, the database server uses the default values to create a single poll thread for the protocol. If the *dbservername* is defined by *DBSERVERNAME*, by default the poll thread is run by the CPU class. If the *dbservername* is defined by *DBSERVERALIASES*, the default VP class is NET.

Multiplexed Connections

To enable the database server to use multiplexed connections on UNIX, you must include a special `NETTYPE` parameter with the value `sqlmux`, as in the following example:

```
NETTYPE sqlmux
```

`NETTYPE sqlmux` does not need to be present in the `ONCONFIG` file. For more information on enabling multiplexed connections, see the *IBM Informix Administrator's Guide*.

IBM Informix MaxConnect

If you are using IBM Informix MaxConnect, see the *IBM Informix MaxConnect User's Guide* for how to specify the fields in the `NETTYPE` parameter. The `ontliimc` and `onsocimc` protocols use TCP/IP to communicate with Informix MaxConnect. You can use these protocols to either connect Informix MaxConnect or the application clients to the database server.

OFF_RECVRY_THREADS

`onconfig.std` value
10

units Number of recovery threads that run in parallel

range of values
Positive integers

takes effect
When the database server is shut down and restarted

refer to

- *IBM Informix Backup and Restore Guide*
- *IBM Informix Performance Guide*

`OFF_RECVRY_THREADS` is the number of recovery threads used in logical recovery when the database server is offline (during a cold restore). This number of threads is also used to roll forward logical-log records in fast recovery.

Before you perform a cold restore, you can set the value of this parameter to approximately the number of tables that have a large number of transactions against them in the logical log. For single-processor computers or nodes, more than 30 to 40 threads is probably too many, because the overhead of thread management offsets the increase in parallel processing.

ON_RECVRY_THREADS

`onconfig.std` value
1

units Number of recovery threads that run in parallel

range of values
Positive integers

takes effect
When the database server is shut down and restarted

refer to

- *IBM Informix Backup and Restore Guide*
- *IBM Informix Performance Guide*

ON_RECOVERY_THREADS is the maximum number of recovery threads that the database server uses for logical recovery when the database server is online (during a warm restore).

You can tune ON_RECOVERY_THREADS to the number of tables that are likely to be recovered, because the logical-log records that are processed during recovery are assigned threads by table number. The maximum degree of parallel processing occurs when the number of recovery threads matches the number of tables being recovered.

To improve the performance of warm restores, increase the number of fast-recovery threads with the ON_RECOVERY_THREADS parameter.

ON-Bar Configuration Parameters

The following table lists the configuration parameters that apply exclusively to the ON-Bar backup and restore utility. For more information on these parameters, see the *IBM Informix Backup and Restore Guide*.

Configuration Parameter	Description
BAR_ACT_LOG	Specifies the location of the ON-Bar activity log file.
BAR_BSA_PATH	Specifies the pathname and filename of the XBSA shared library for the storage manager.
BAR_DEBUG	Specifies the level of debugging messages in the ON-Bar activity log.
BAR_DEBUG_LOG	Specifies the location of the ON-Bar debug log.
BAR_HISTORY	Specifies whether the sysutils database maintains a backup history.
BAR_MAX_BACKUP	Specifies the maximum number of parallel backup processes that are allowed for each ON-Bar command.
BAR_NB_XPORT_COUNT	Specifies the number of shared-memory data buffers for each backup or restore process.
BAR_PERFORMANCE	Specifies the level of performance statistics to print to the ON-Bar activity log.
BAR_PROGRESS_FREQ	Specifies in minutes how frequently the backup or restore progress messages display in the activity log.
BAR_RETRY	Specifies how many times ON-Bar should retry a backup or restore operation.
BAR_XFER_BUF_SIZE	Specifies the size in pages of the buffers.
ISM_DATA_POOL	Specifies the volume pool that you use for backing up storage spaces.
ISM_LOG_POOL	Specifies the volume pool that you use for backing up logical logs.

Backup and restore can also be performed using SQL API *command* equivalents. For more information see *IBM Informix Guide to SQL: Syntax* and *IBM Informix Administrator's Guide*

ONDBSPACEDOWN

ONDBSPACEDOWN defines the action that the database server takes when any disabling event occurs on a primary chunk within a noncritical dbspace.

onconfig.std *value*
2

range of values
0, 1, 2

refer to Monitoring the database server for disabling I/O errors, in the chapter on consistency checking in the *IBM Informix Administrator's Guide*

The following values are valid for this parameter:

Value	Description
-------	-------------

- | | |
|---|---|
| 0 | The database server marks the dbspace as offline and continues. |
| 1 | The database server aborts. |
| 2 | The database server writes the status of the chunk to the logs and waits for user input. If you set this option, but you want the database server to mark a disabled dbspace as down and continue processing, use onmode -O to override this ONDBSPACEDOWN setting. See “onmode -O: Override ONDBSPACEDOWN WAIT mode” on page 11-15. |

Database Server Behavior When ONDBSPACEDOWN Does Not Apply

The database server will not come online if a chunk within any **critical** dbspace (for example, rootdbs or logsdbs) is missing.

The value of ONDBSPACEDOWN has no effect on temporary dbspaces. For temporary dbspaces, the database server continues processing regardless of the ONDBSPACEDOWN setting. If a temporary dbspace requires fixing, you should drop and recreate it.

For a non-primary chunk within a noncritical dbspace, the behavior of the database server depends on the transaction status of the chunk when the disabling event occurs:

- **No transaction:** If no transactions are detected against that chunk, the chunk is individually marked as down. In this case, subsequent attempts to write to that chunk fail, rolling back the associated transaction. You can safely put the chunk back and then use the **onspaces -s** utility to mark the chunk as back online.
- **Transaction detected:** If there are transactions to roll forward or back, then the database server aborts with an appropriate fast recovery error. In this case, you should put the chunk back and restart the database server.

ONLIDX_MAXMEM

onconfig.std *value*
5120

units Kilobytes

range of values

16 through 4294967295

takes effect

When the database server is shut down and restarted

utilities

onmode -wf onmode-wm

The ONLIDX_MAXMEM configuration parameter limits the amount of memory that is allocated to a single *preimage* pool and a single *updater* log pool. The preimage and updater log pools, **pimage_partnum** and **ulog_partnum**, are shared memory pools that are created when a CREATE INDEX ONLINE statement is executed. The pools are freed when the execution of the statement is completed.

If you specify a value for this parameter and then create a table, add rows to the table, and start to execute a CREATE INDEX ONLINE statement on a column, you can also perform other operations on the column, such as running UPDATE STATISTICS HIGH, without having memory problems.

The ONLIDX_MAXMEM configuration parameter can be changed using the **onmode -wf** option or superseded for a session with the **onmode -wm** option. For more information about **onmode**, see “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20.

OPCACHEMAX (UNIX)

onconfig.std *value*

0

if not present

128

units Kilobytes

range of values

0 through (4 * 1024 * 1024)

takes effect

When the Optical Subsystem needs more memory

utilities

onstat -O (For more information, see “onstat -o: Output shared memory contents to a file” on page 15-146.)

refer to

- *IBM Informix Optical Subsystem Guide*
- **INFORMIXOPCACHE** environment variable, in the *IBM Informix Guide to SQL: Reference*

OPCACHEMAX specifies the size of the memory cache for the Optical Subsystem. The database server stores pieces of TEXT or BYTE data in the memory cache before it delivers them to the subsystem. Use this parameter only if you use the Optical Subsystem.

The **INFORMIXOPCACHE** environment variable lets the client restrict the size of the optical cache that it uses.

OPTCOMPIND

onconfig.std *value*
2

range of values

0 = When appropriate indexes exist for each ordered pair of tables, the optimizer chooses index scans (nested-loop joins), without consideration of the cost, over table scans (hash joins). This value ensures compatibility with previous versions of the database server.

1 = The optimizer uses costs to determine an execution path if the isolation level is not Repeatable Read. Otherwise, the optimizer chooses index scans (it behaves as it does for the value 0). This setting is recommended for optimal performance.

2 = The optimizer uses cost to determine an execution path for any isolation level. Index scans are not given preference over table scans; the optimizer bases its decision purely on cost. This value is the default if the variable is not set.

refer to

- Your *IBM Informix Performance Guide*
- **OPTCOMPIND** environment variable, in the *IBM Informix Guide to SQL: Reference*
- SET ENVIRONMENT OPTCOMPIND statement to dynamically change the value of the OPTCOMPIND configuration parameter for a session, in the *IBM Informix Guide to SQL: Syntax*

OPTCOMPIND helps the optimizer choose an appropriate query plan for your application.

Tip: You can think of the name of the variable as arising from “OPTimizer COMPare (the cost of using) INDexes (with other methods).”

Because of the nature of *hash joins*, an application with isolation mode set to Repeatable Read might *temporarily* lock all records in tables that are involved in the join (even those records that fail to qualify the join) for each ordered set of tables. This situation leads to higher contention among connections. Conversely, nested-loop joins lock fewer records but provide inferior performance when the database server retrieves a large number of rows. Thus, both join methods offer advantages and disadvantages. A client application can also influence the optimizer in its choice of a join method.

OPT_GOAL

onconfig.std *value*
-1

range of values

0 or -1

takes effect

When the database server is shut down and restarted

refer to The following manuals:

- ALL_ROWS and FIRST_ROWS directives and on the SET OPTIMIZATION statement, in the *IBM Informix Guide to SQL: Reference*

- **OPT_GOAL** environment variable, in the *IBM Informix Guide to SQL: Syntax*
- Performance issues associated with setting an optimization goal, in the *IBM Informix Performance Guide*

The **OPT_GOAL** parameter enables you to specify one of the following optimization goals for queries:

- Optimize for **FIRST ROWS**
- Optimize for **ALL ROWS**

A value of 0 sets the optimization goal to **FIRST_ROWS**. A value of -1 sets the optimization goal to **ALL_ROWS**, which is the default.

When you set the optimization goal to optimize for **FIRST ROWS**, you specify that you want the database server to optimize queries for perceived response time. In other words, users of interactive applications perceive response time as the time that it takes to display data on the screen. Setting the optimization goal to **FIRST ROWS** configures the database server to return the first rows of data that satisfy the query.

When you set the optimization goal to optimize for **ALL ROWS**, you specify that you want the database server to optimize for the total execution time of the query. Making **ALL ROWS** the optimization goal instructs the database server to process the total query as quickly as possible, regardless of how long it takes to return the first rows to the application.

You can specify the optimization goal in one of four ways:

- By query (**SELECT** statement)
Use the **ALL_ROWS** and **FIRST_ROWS** directives.
- By session
Use the **SET OPTIMIZATION** statement.
- By environment
Set the **OPT_GOAL** environment variable.
- By database server
Set the **OPT_GOAL** configuration parameter.

To determine the optimization goal, the database server examines the settings in the order shown. The first setting encountered determines the optimization goal. For example, if a query includes the **ALL_ROWS** directive but the **OPT_GOAL** configuration parameter is set to **FIRST_ROWS**, the database server optimizes for **ALL_ROWS**, as the query specifies.

PC_HASHSIZE

onconfig.std *value*
31

range of values

Any positive integer, a prime number is recommended.

takes effect

When the database server is shut down and restarted

refer to Your *IBM Informix Performance Guide*

Use PC_HASHSIZE to specify the number of hash buckets in the caches that the database server uses.

PC_HASHSIZE applies to UDR cache only. For information on configuration parameters for other types of cache, see the “DS_POOLSIZ” on page 1-38 and the “DS_HASHSIZE” on page 1-36.

PC_POOLSIZ

onconfig.std *value*
127

default value
127

range of values
Any positive value from 127 to x , where x is dependent upon the shared memory configuration and available shared memory for the server instance.

takes effect
When the database server is shut down and restarted

refer to Your IBM Informix Performance Guide

PC_POOLSIZ specifies the maximum number of UDRs stored in the UDR cache.

For information on configuration parameters for other types of cache, see the “DS_POOLSIZ” on page 1-38 and the “DS_HASHSIZE” on page 1-36.

PHYSBUFF

onconfig.std *value*
128

units Kilobytes

range of values
4 kilobytes through $(32767 * \text{page size} / 1024)$ kilobytes.

takes effect
When the database server is shut down and restarted

utilities
onstat -l *buffer* field, first section (For more information, see “onstat -l: Print physical and logical log information” on page 15-142.)

refer to Physical-log buffer, in the shared-memory chapter of the IBM Informix Administrator’s Guide

PHYSBUFF specifies the size in kilobytes of the two physical-log buffers in shared memory. Double buffering permits user threads to write to the active physical-log buffer while the other buffer is being flushed to the physical log on disk. The value of the PHYSBUFF parameter determines how frequently the database server needs to flush the physical-log buffer to the physical-log file. If RTO_SERVER_RESTART is enabled, use the 512 kilobyte default value for PHYSBUFF; if the PHYSBUFF value is less than 512 kilobytes, a warning message displays when you restart the server.

A write to the physical-log buffer is exactly one page in length. Choose a value for PHYSBUFF that is evenly divisible by the page size. If the value of PHYSBUFF is not evenly divisible by the page size, the database server rounds down the size to the nearest value that is evenly divisible by the page size.

The user-data portion of a smart large object does not pass through the physical-log buffers.

The system page size is platform-dependent on Dynamic Server. To obtain the system page size, use the commands listed in the table in “System Page Size” on page 1-21.

PHYSFILE

onconfig.std *value*
50000

if not present
200

units Kilobytes

range of values
200 or more

takes effect
When the database server is initialized

utilities

onparams -p

refer to The following topics in the *IBM Informix Administrator's Guide*:

- Sizing the physical log, in the chapter on the physical log
- Changing the physical-log location and size, in the chapter on managing the physical log

PHYSFILE specifies the size of the physical log. PHYSFILE can be changed dynamically with the **onparams** utility. Restarting the server is not required for the changes to take effect.

When RTO_SERVER_RESTART is enabled, ensure that the PHYSFILE value is equal to at least 110% of the buffer pool size. A warning message prints to the message log when the PHYSFILE value is changed to less than 110% of all of the buffer pools, when the server is restarted, or when a new buffer pool is added.

PLOG_OVERFLOW_PATH

onconfig.std *value*
On UNIX: \$INFORMIXDIR/tmp On Windows: None

if not present
\$INFORMIXDIR/tmp

takes effect
When the database server is brought up (shared memory is initialized)

refer to Your *IBM Informix Administrator's Guide*

The PLOG_OVERFLOW_PATH parameter specifies the location of the file that is used during fast recovery if the physical log file overflows. The file is

plog_extend.servernum and by default located in \$INFORMIXDIR/tmp. Use the full pathname to specify a different location for the file with the PLOG_OVERFLOW_PATH parameter.

QSTATS

onconfig.std *value*

0

range of values

0 = Disable queue statistics 1 = Enable queue statistics

takes effect

When the database server is shut down and restarted

utilities

onstat -g qst

refer to

- “onstat -g qst: Print wait options for mutex and condition queues ” on page 15-96

QSTATS specifies the ability of **onstat -g qst** to print queue statistics.

RA_PAGES

onconfig.std *value*

64

if not present

4 if MULTIPROCESSOR is 0; 8 if MULTIPROCESSOR is 1

units Number of data pages

range of values

RA_THRESHOLD through BUFFERPOOL

takes effect

When the database server is shut down and restarted

refer to

- Configuring the database server to read ahead, in the shared-memory chapter of the *IBM Informix Administrator's Guide*
- Calculating RA_PAGES and RA_THRESHOLD, in your *IBM Informix Performance Guide*

RA_PAGES specifies the number of disk pages to attempt to read ahead during sequential scans of data records. Read-ahead can greatly speed up database processing by compensating for the slowness of I/O processing relative to the speed of CPU processing.

This parameter works with the RA_THRESHOLD parameter. Specifying values that are too large can result in excessive buffer-caching activity.

RA_THRESHOLD

onconfig.std *value*

16

if not present

RA_PAGES/2

units Number of data pages

range of values

0 through (RA_PAGES - 1)

takes effect

When the database server is shut down and restarted

refer to

- Configuring the database server to read ahead, in the shared-memory chapter of the *IBM Informix Administrator's Guide*
- Calculating RA_PAGES and RA_THRESHOLD, in your *IBM Informix Performance Guide*

RA_THRESHOLD is used with RA_PAGES when the database server reads during sequential scans of data records. RA_THRESHOLD specifies the read-ahead threshold; that is, the number of unprocessed data pages in memory that signals the database server to perform the next read-ahead.

If the value of RA_THRESHOLD is greater than the value of RA_PAGES, RA_THRESHOLD has a value of RA_PAGES/2.

Specifying values that are too large for RA_PAGES and RA_THRESHOLD can result in excessive buffer-caching activity.

UPDATABLE_SECONDARY

onconfig.std *value*

0

if not present

0

units Number of network connections between a given secondary server and its primary server

range of values

Subject to system resources (see text below)

takes effect

When the secondary database server is shut down and restarted

refer to

- Data Replication Overview chapter of the *IBM Informix Administrator's Guide*
- Server Multiplexer Group (SMX) Connections chapter of the *IBM Informix Administrator's Guide*

To enable client applications to perform update, insert, and delete operations on a high-availability secondary server, set the UPDATABLE_SECONDARY configuration parameter to the number of connections to establish between the primary and secondary servers. To set the secondary server as read-only, which is the default behavior, set UPDATABLE_SECONDARY to 0.

RESIDENT

onconfig.std *value*
0

range of values

-1 to 99 0 = off 1 = lock the resident segment only n = lock the resident segment and the next *n*-1 virtual segments -1 = lock all resident and virtual segments 99 = lock the resident segment and the next 98 virtual segments

Certain platforms have different values. For information, see your machine notes.

if not present
0

takes effect

When the database server is shut down and restarted

utilities

onmode -r (see “ onmode -n, -r: Change shared-memory residency” on page 11-15)

refer to The following topics in the *IBM Informix Administrator's Guide* for a discussion of residency:

- Resident portion of shared memory, in the shared-memory chapter
- Setting database server shared-memory configuration parameters, in the chapter on managing shared memory

The RESIDENT parameter specifies whether resident and virtual segments of shared memory remain resident in operating-system physical memory.

Some systems allow you to specify that the resident portion of shared memory must stay (be resident) in memory at all times. If your operating system supports forced residency, you can specify that resident and virtual segments of shared memory not be swapped to disk.

Warning: Before you decide to enforce residency, verify that the amount of physical memory available is sufficient to execute all required operating-system and application processes. If insufficient memory is available, a system hang could result that requires a reboot.

RESTARTABLE_RESTORE

onconfig.std *value*
ON

if not present
ON

range of values

OFF = restartable restore is disabled ON = restartable restore is enabled

takes effect

When the database server is shut down and restarted

refer to *IBM Informix Backup and Restore Guide*

If you set `RESTARTABLE_RESTORE` to `ON`, you enable the database server to restart a failed physical or cold logical restore at the point at which the failure occurred. To perform a restartable restore with `ON-Bar`, use the **onbar -RESTART** command.

Increase the size of your physical log if you plan to use restartable restore. For more information, see “`PHYSFILE`” on page 1-79. Although a restartable restore slows down the logical restore if many logs need to be restored, you save a lot of time from not having to repeat the entire restore.

Important: If the database server fails during a warm logical restore, you must repeat the entire restore. If the database server is still running, use **onbar -r -l** to complete the restore.

If you do a cold restore on systems that are not identical, you can assign new pathnames to chunks, and you can rename devices for critical chunks during the restore. You must perform a level-0 archive after the rename and restore operation completes. For details, see the *IBM Informix Backup and Restore Guide*

The database server uses physical recovery and logical recovery to restore data as follows:

- **Physical recovery.** The database server writes data pages from the backup media to disk. This action leaves the storage spaces consistent to the point at which it was originally backed up. However, the backup times for each storage space are usually different. A restartable restore is restartable to the level of a storage space. If only some chunks of a storage space are restored when the restore fails, the entire storage space needs to be recovered again when you restart the restore.
- **Logical recovery.** The database server replays logical-log records on media to bring all the storage spaces up to date. At the end of logical recovery, all storage spaces are consistent to the same point.

ROOTNAME

onconfig.std *value*
rootdbs

units A dbspace

range of values

Up to 128 bytes. `ROOTNAME` must begin with a letter or underscore and must contain only letters, numbers, underscores, or \$ characters.

takes effect

When disk is initialized (destroys all data)

refer to Allocating disk space, in the chapter on managing disk space in the *IBM Informix Administrator's Guide*

`ROOTNAME` specifies a name for the root dbspace for this database server configuration.

The name must be unique among all dbspaces that the database server manages. It is recommended that you select a name that is easily recognizable as the root dbspace.

ROOTOFFSET

onconfig.std *value*

0

units Kilobytes

range of values

Any value greater than or equal to 0

takes effect

When disk is initialized (destroys all data)

refer to Allocating raw disk space on UNIX, in the chapter on managing disk space in the *IBM Informix Administrator's Guide*

ROOTOFFSET specifies the offset into an allocation of disk space (file, disk partition, or device) at which the initial chunk of the root dbspace begins.

UNIX Only:

On some UNIX platforms, it is not valid to set ROOTOFFSET to 0. When this parameter is set incorrectly, you must reinitialize disk space and reload data to resume proper operation of the database server. Before you configure the database server, always check your machine notes file for information about proper settings.

ROOTPATH

onconfig.std *value*

On UNIX: `$INFORMIXDIR/tmp/demo_on.rootdbs` On Windows: None

range of values

Pathname

takes effect

When disk is initialized (destroys all data)

refer to The following material in the chapter on managing disk space in the *IBM Informix Administrator's Guide*

- Allocating disk space
- Creating links for raw devices

ROOTPATH specifies the full pathname, including the device or filename, of the initial chunk of the root dbspace. ROOTPATH is stored in the reserved pages as a chunk name.

On UNIX, you must set the permissions of the file that ROOTPATH specifies to 660, and the owner and group must both be **informix**. On Windows, a member of the **Informix-Admin** group must own the file that ROOTPATH specifies.

UNIX Only:

If you use unbuffered disk space for your initial chunk on UNIX, it is recommended that you define ROOTPATH as a pathname that is a link to the initial chunk of the root dbspace instead of entering the actual device name for the initial chunk.

ROOTSIZE

onconfig.std *value*
200000

if not present
0

units Kilobytes

range of values
50,000 through maximum capacity of the storage device

takes effect
When disk is initialized (destroys all data)

refer to Calculating the size of the root dbspace, in the chapter on where is data stored in the *IBM Informix Administrator's Guide*

ROOTSIZE specifies the size of the initial chunk of the root dbspace, expressed in kilobytes. The size that you select depends on your immediate plans for your database server.

To change ROOTSIZE after you initialize the database server, completely unload and reload your data.

RTO_SERVER_RESTART

onconfig.std *value*
0 (disabled)

units seconds

range of values
0 = disabled
60 to 1800

takes effect
When the database server is stopped and restarted. When the RTO_SERVER_RESTART value is changed by using **onmode -wm** or **onmode -wf**.

utilities
oncheck -pronmode -wf or **onmode -wmonparams**

refer to

- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- Nonblocking checkpoints information in *IBM Informix Administrator's Guide*
- Effects of configuration on I/O activity in *IBM Informix Performance Guide*

RTO_SERVER_RESTART enables you to use recovery time objective (RTO) standards to set the amount of time, in seconds, that Dynamic Server has to recover from a problem after you restart Dynamic Server and bring the server into online or quiescent mode.

SBSPACENAME

onconfig.std *value*

None

if not present

0

range of values

Up to 128 bytes. SBSPACENAME must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or \$ characters.

takes effect

When shared memory is reinitialized

utilities

onspaces -c -S

refer to

- Using **onspaces** to “onspaces -c -S: Create an sbspace” on page 14-11
- “SBSPACETEMP” on page 1-87
- “SYSSBSPACENAME” on page 1-101
- “Sbspace Structure” on page 4-23
- What is an sbspace, in the chapter on data storage in the *IBM Informix Administrator’s Guide*
- Altering sbspace characteristics, in the chapter on managing data on disk in the *IBM Informix Administrator’s Guide*
- Assigning a smart large object to an sbspace, in the section on the CREATE TABLE and ALTER TABLE statements, in the *IBM Informix Guide to SQL: Syntax*
- Creating an sbspace for Enterprise Replication usage, in the *IBM Informix Dynamic Server Enterprise Replication Guide*
- Using multirepresentational data, in the *IBM Informix DataBlade API Programmer’s Guide*

SBSPACENAME specifies the name of the default sbspace. If your database tables include smart-large-object columns that do not explicitly specify a storage space, that data is stored in the sbspace that SBSPACENAME specifies. The default sbspace is also used by the built-in encryption and decryption functions to store BLOB or CLOB values. If DECRYPT_BINARY or an encryption function cannot find an sbspace in which to store a BLOB or CLOB argument or returned value, the function fails with the following error message:

```
Fatal error in server row processing - SQL error -9810 ISAM error -12053
```

If you see this error message after you invoke an encryption or decryption function that has a CLOB or BLOB argument, configure a default sbspace using the SBSPACENAME configuration parameter, and then repeat the function call.

You must create the default sbspace with the **onspaces -c -S** utility before you can use it. The database server validates the name of the default sbspace when one of the following occurs:

- You specify the default sbspace as the storage option for a CLOB or BLOB column in the PUT clause of the CREATE TABLE or ALTER TABLE statement.
- The database server attempts to write a smart large object to the default sbspace when no sbspace was specified for the column.

- You store multirepresentational data in the default sbspace.

JAVA Language Support:

If you are using IBM Informix Dynamic Server with J/Foundation, you must provide a smart large object where the database server can store the Java archive (JAR) files. These JAR files contain your Java user-defined routines (UDRs). It is suggested that when you use Java UDRs, you create separate sbspaces for storing smart large objects.

Warning: When you use Enterprise Replication, you must set the `CDR_QDATA_SBSPACE` parameter and create the sbspace before you define the replication server.

SBSPACETEMP

onconfig.std *value*
None

if not present

Temporary smart large objects are stored in a standard sbspace.

range of values

Up to 128 bytes. SBSPACETEMP must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or \$ characters.

takes effect

When shared memory is reinitialized

utilities

onspaces

refer to

- “onspaces -c -S: Create an sbspace” on page 14-11
- “SBSPACENAME” on page 1-86
- Temporary sbspaces, in the chapter on data storage in the *IBM Informix Administrator’s Guide*
- Creating a temporary sbspace, in the chapter on managing disk space in the *IBM Informix Administrator’s Guide*
- Using temporary smart large objects, in the *IBM Informix DataBlade API Programmer’s Guide*

SBSPACETEMP specifies the name of the default temporary sbspace for storing temporary smart large objects without metadata or user-data logging. If you store temporary smart large objects in a standard sbspace, the metadata is logged.

SDS_ENABLE

onconfig.std *value*
None

if not present

0

range of values

0, 1 0 = disable, 1 = enable

takes effect

When database server is shut down and restarted.

refer to

- *Using Shared Disk Secondary Servers in the IBM Informix Dynamic Server Administrator's Guide.*

SDS_ENABLE enables SD secondary server functionality. You must set SDS_ENABLE to 1 (enable) on the SD secondary server to enable SD secondary server functionality.

SDS_ENABLE is set to 1 (enabled) automatically when you run the following command:

```
onmode -d set SDS primary
```

SDS_ENABLE is set to 0 (disabled) when you run the following command:

```
onmode -d clear SDS primary
```

To prevent data corruption, you cannot use the **oninit -i** or **oninit -iy** command to initialize disk space on a server if SDS_ENABLE is set to 1 (enabled). To initialize an SD secondary server, initialize only the shared memory by using **oninit** with no parameters. To initialize a primary server to which one or more SD secondary servers are attached, and whose disk has never been initialized, set SDS_ENABLE to 0 and initialize the server memory and disk using **oninit -i**. To initialize a primary server to which SD secondary servers are attached, and whose disk is already initialized, set SDS_ENABLE to 1 and initialize shared memory only using **oninit** with no parameters. See Chapter 9, "The oninit Utility," on page 9-1.

SDS_PAGING

onconfig.std *value*

None

takes effect

When SD secondary server is started

refer to

- *Using Shared Disk Secondary Servers in the IBM Informix Dynamic Server Administrator's Guide.*

SDS_PAGING specifies the location of two files that will act as buffer paging files. Set SDS_PAGING to ensure that the SD secondary server starts. The size of the buffer paging files can be as much as two times the size of the value specified in the PHYSFILE configuration parameter (see "PHYSFILE" on page 1-79).

SDS_TEMPDBS

onconfig.std *value*

None

if not present

Error message displayed: "Can not initialize SDS - Missing or invalid entry for SDS_TEMPDBS." SD secondary server will not start.

units pagesize, offset, and size specified in KB. All temporary SDS dbspaces must be the same page size.

range of values

0 on the primary database server, 1 to 16 dbspace entries on an SD secondary server

takes effect

When the SD secondary server is started

refer to

- *Using Shared Disk Secondary Servers* in the *IBM Informix Dynamic Server Administrator's Guide*.

SDS_TEMPDBS specifies information that the SD secondary server uses to dynamically create temporary dbspaces. These temporary dbspaces are created (or initialized if the dbspaces existed previously) when the SD secondary server starts. The temporary dbspaces are used for creating temporary tables. There must be at least one occurrence of SDS_TEMPDBS in the ONCONFIG file of the SD secondary server for the SD secondary server to start.

Specify up to 16 SD secondary dbspaces in the ONCONFIG file by using multiple occurrences of SDS_TEMPDBS. For each occurrence of SDS_TEMPDBS in the ONCONFIG file, *dbspaceName* must be unique and the combination of *path*, *offset*, and *size* must not cause any overlap with existing chunks or among SDS_TEMPDBS spaces. In addition, *dbspaceName* must be unique among all existing dbspaces, blobspaces, and sbspaces, including those (possibly disabled) temp spaces inherited from a primary server.

SDS_TEMPDBS is specified only on the SD secondary server.

SDS_TIMEOUT

onconfig.std *value*

20

if not present

10

range of values

2 to 2147483647

units seconds

takes effect

When shared disk functionality is enabled on the primary server

refer to

- The **onmode -wf** command
- *Using Shared Disk Secondary Servers* in the *IBM Informix Dynamic Server Administrator's Guide*.

SDS_TIMEOUT specifies the amount of time in seconds that the primary server will wait for a log position acknowledgement to be sent from the SD secondary server. If there is no log position acknowledgement received from the SD secondary server in the specified amount of time, the primary server will be disconnected from the SD secondary server and continue. After waiting for the SDS_TIMEOUT number of seconds, the primary server will start removing SD secondary servers if page flushing has timed out while waiting for an SD secondary server.

SECURITY_LOCALCONNECTION

onconfig.std *value*
None

range of values

0, 1, 2 0 = no security checking occurs 1 = Dynamic Server checks whether the ID of the user who is running the program matches the ID of the user who is trying to connect to the database. 2 = same as 1, plus Dynamic Server retrieves the peer port number from the network API and verifies that the connection is coming from the client program. You can only specify 2 if your system has SOCTCP or IPCSTR network protocols.

takes effect

When the database server is shut down and restarted

refer to Role of the SERVERNUM configuration parameter, in the multiple-residency chapter of the *IBM Informix Administrator's Guide*

SECURITY_LOCALCONNECTION lets you verify security on local connections by verifying that the ID of the local user who is running a program is the same ID of the user who is trying to access the database.

SERVERNUM

onconfig.std *value*
0

range of values

0 through 255

takes effect

When the database server is shut down and restarted

refer to Role of the SERVERNUM configuration parameter, in the multiple-residency chapter of the *IBM Informix Administrator's Guide*

SERVERNUM specifies a relative location in shared memory. The value that you choose must be unique for each database server on your local computer. The value does not need to be unique on your network. Because the value 0 is included in the **onconfig.std** file, it is suggested that you choose a value other than 0 to avoid inadvertent duplication of SERVERNUM.

SHMADD

onconfig.std *value*
8192

range of values

32-bit platforms: 1024 through 524288 64-bit platforms: 1024 through 4294967296

units Kilobytes

takes effect

When the database server is shut down and restarted

utilities

onstat -g seg (Use the **onstat -g seg** command to display the number of

shared-memory segments that the database server is currently using. For more information, see “onstat -g seg: Print shared memory segment statistics” on page 15-112)

refer to The following material in the *IBM Informix Administrator's Guide*:

- Virtual portion of shared memory, in the shared-memory chapter
- Monitoring shared-memory segments with **onstat -g seg**, in the managing memory chapter

SHMADD specifies the size of a segment that is dynamically added to the virtual portion of shared memory.

It is more efficient to add memory in large segments, but wasteful if the added memory is not used. Also, the operating system might require you to add memory in a few large segments rather than many small segments.

The following table contains recommendations for setting the initial value of SHMADD.

Amount of Physical Memory	Recommended SHMADD Value
Less than 256 megabytes	8192
Between 256 megabytes and 512 megabytes	16,384
Greater than 512 megabytes	32,768

SHMBASE

onconfig.std *value*

Platform dependent

units Address

range of values

Positive integers

takes effect

When the database server is shut down and restarted

utilities

To see the shared-memory segment addresses, use the **onstat -g seg** command.

refer to Setting operating-system shared-memory configuration parameters, in the chapter on managing shared memory in the *IBM Informix Administrator's Guide*

SHMBASE specifies the base address where shared memory is attached to the memory space of a virtual processor. The addresses of the shared-memory segments start at the SHMBASE value and grow until the upper-bound limit, which is platform specific.

Do not change the value of SHMBASE. The **onconfig.std** value for SHMBASE depends on the platform and whether the processor is 32-bit or 64-bit. For information on which SHMBASE value to use, see the machine notes.

SHMNOACCESS

onconfig.std *value*

On UNIX: None

On Windows: #SHMNOACCESS 0x70000000-0x7FFFFFFF, and this value is commented out in the onconfig.std template file.

separators

Comma

range of values

From one to ten address ranges

takes effect

When the database server is shut down and restarted

The SHMNOACCESS configuration parameter specifies a virtual memory address range to **not** use to attach shared memory. SHMNOACCESS is used to avoid specific range process addresses, which in turn avoids conflicts with operating system libraries.

Each address in each range must start in hexadecimal format. Each address in a range must be separated by a hyphen and each range must be separated by a comma, as the following example shows:

```
SHMNOACCESS 0x70000000-0x75000000,  
0x7A000000-0x80000000
```

SHMTOTAL

onconfig.std *value*

0

units Kilobytes

range of values

0 (= no specific limit) or any integer greater than or equal to 1

takes effect

When the database server is shut down and restarted

refer to How much shared memory the database server needs, in the shared-memory chapter of the *IBM Informix Administrator's Guide*

SHMTOTAL specifies the total amount of shared memory (resident, virtual, communications, and virtual extension portions) to be used by the database server for all memory allocations. The **onconfig.std** value of 0 implies that no limit on memory allocation is stipulated.

SHMTOTAL enables you to limit the demand for memory that the database server can place on your system. However, applications might fail if the database server requires more memory than the limit imposed by SHMTOTAL. When this situation occurs, the database server writes the following message in the message log:

```
size of resident + virtual segments xx + yy > zz total allowed by  
configuration parameter SHMTOTAL
```

This message includes the following values.

Value	Description
-------	-------------

xx	Current size of resident segments
----	-----------------------------------

yy Current size of virtual segments
zz Total shared memory required

UNIX Only:

Set the operating-system parameters for maximum shared-memory segment size, typically SHMMAX, SHMSIZE, or SHMALL, to the total size that your database server configuration requires. For information on the amount of shared memory that your operating system allows, see the machine notes.

SHMVIRT_ALLOCSEG

onconfig.std *value*
0,3

units First parameter: a decimal number indicating the percentage of memory used before a segment is added OR the whole number of kilobytes remaining in the server when a segment is added
Second parameter: alarm level

range of values

First parameter: 0 (the default) OR any percentage from .40 to .99 OR a whole number of kilobytes from 256 to 1,000,000 = Disabled

Second parameter: Alarm level values from 1 to 5 where: 1 = Not noteworthy
2 = Information
3 = Attention
4 = Emergency
5 = Fatal

takes effect

When the database server is shut down and restarted

refer to

- “ Event Severity” on page C-3
- Configuration Parameters That Affect Memory Utilization in the *IBM Informix Performance Guide*

SHMVIRT_ALLOCSEG specifies a threshold at which Dynamic Server should allocate a new memory segment and the alarm level activated if the server cannot allocate the new memory segment, thus ensuring that the server never runs out of memory. Once the alarm level is activated, it will repeat every thirty minutes if a new memory segment cannot be allocated.

SHMVIRT_SIZE

onconfig.std *value*
32656

if not present

If SHMADD is present: SHMADD
If SHMADD is not present: 8

units Kilobytes

range of values

32-bit platforms: Positive integer with a maximum value of 2 gigabytes

64-bit platforms: Positive integer with a maximum value of 4 terabytes

The maximum value might be less on some platforms due to operating-system limitations. For the actual maximum value for your UNIX platform, see the machine notes.

takes effect

When the database server is shut down and restarted

utilities

onstat -g seg (see “onstat -g seg: Print shared memory segment statistics” on page 15-112)

refer to

- Virtual portion of shared memory, in the shared-memory chapter of the *IBM Informix Administrator's Guide*
- Chapter on configuration effects on memory utilization, in your *IBM Informix Performance Guide*

SHMVIRTSIZE specifies the initial size of a virtual shared-memory segment. Use the following algorithm to determine the size of the virtual portion of shared memory:

$$\text{shmvirtualsize} = \text{fixed overhead} + \text{shared structures} + (\text{mncs} * \text{private structures}) + \text{other buffers}$$

This algorithm includes the following values.

Value	Description
-------	-------------

fixed_overhead	Global pool + thread pool after booting (partially dependent on the number of virtual processors)
-----------------------	---

shared_structures	AIO vectors + sort memory + dbspace backup buffers + dictionary size + size of stored-procedure cache + histogram pool + other pools (See the onstat display.)
--------------------------	---

mncs	Maximum number of concurrent sessions
-------------	---------------------------------------

private_structures	Stack (generally 32 kilobytes but dependent on recursion in SPL routines and triggers) + heap (about 30 kilobytes) + session-control-block structures
---------------------------	---

other_buffers	private buffers allocated for features such as lightweight I/O operations for smart large objects (about 180 kilobytes per user).
----------------------	---

If messages in the message file indicate that the database server is adding segments to the virtual portion of shared memory for you, add the amount that these messages indicate to the value of SHMVIRTSIZE. It is recommended that you initially create a virtual portion of shared memory of a size that is more than sufficient for your daily processing, if possible.

Use the **onstat -g seg** command to determine peak usage and lower the value of SHMVIRTSIZE accordingly.

SINGLE_CPU_VP

onconfig.std *value*

0

range of values

0 = running with multiple CPU VPs Any nonzero value = running with one CPU VP

takes effect

When the database server is shut down and restarted

refer to Running on a single-processor computer, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*

SINGLE_CPU_VP specifies whether or not the database server is running with only one CPU virtual processor.

Setting SINGLE_CPU_VP to nonzero allows the database server to use optimized code based on the knowledge that only one CPU virtual processor is running. It enables the database server to bypass many of the mutex calls that it must use when it runs multiple CPU virtual processors.

It is strongly recommended that you set this parameter when the database server will run only one CPU virtual processor. Depending on the application and workload, setting this parameter can improve performance by up to 10 percent.

If you set SINGLE_CPU_VP to nonzero and try to add a CPU virtual processor, you receive one of the following messages:

```
onmode: failed when trying to change the number of classname VPs by n.  
onmode: failed when trying to change the number of cpu virtual processors by n.
```

If you set SINGLE_CPU_VP to nonzero and then attempt to bring up the database server with VPCLASS *cpu,num* set to a value greater than 1, you receive the following error message, and the database server initialization fails:

```
Cannot have SINGLE_CPU_VP non-zero and CPU VPs greater than 1.
```

User-Defined VP Classes and SINGLE_CPU_VP

Important: Dynamic Server treats user-defined virtual-processor classes as if they were CPU virtual processors. Thus, if you set *SINGLE_CPU_VP* to nonzero, you cannot create any user-defined virtual-processor classes.

If you set this parameter to nonzero and then attempt to bring up the database server with the VPCLASS *cpu* value for *num* set to a value greater than 1, you receive the following error message, and the database server initialization fails:

```
Cannot have SINGLE_CPU_VP non-zero and CPU VPs greater than 1.
```

If you set this parameter to nonzero and then attempt to bring up the database server with a user-defined VPCLASS, you receive the following error message, and the database server initialization fails:

```
oninit: Cannot have SINGLE_CPU_VP non-zero and user-defined VP classes
```

SQLTRACE

syntax

```
SQLTRACE [level=low|high],  
[ntraces=number of traces],  
[size=size of each trace buffer]  
, [mode=global|userid]
```

onconfig.std *value*

```
#SQLTRACE level=low,ntraces=1000,size=2,mode=global (suggested value,  
but not set)
```

range of values

- level** Amount of information traced.
- ntraces** Number of SQL statements to trace before reusing the resources. The range is 500 to 2147483647.
- size** Maximum size of variable length data to be stored. The range is 1 Kilobyte to 100 Kilobytes.
- mode** Type of tracing performed. `global`= all users; `userid` = specific user.

takes effect

When the database server is shut down and restarted

utilities

onstat -g his

refer to SQL tracing information in the *IBM Informix Administrator's Guide*

The SQLTRACE parameter controls the startup environment of the SQL Trace facility.

SSL_KEYSTORE_LABEL

onconfig.std *value*
None

range of values

Up to 512 characters for the label of the Dynamic Server certificate used in Secure Sockets Layer (SSL) protocol communications

takes effect

When the database server is shut down and restarted

refer to Secure Sockets Layer protocol information in the *IBM Informix Security Guide*

The SSL_KEYSTORE_LABEL configuration parameter specifies the label of the server digital certificate used in the keystore database, a protected database that stores SSL keys and digital certificates. The default value is name of the label for the default SSL certificate that is stored in the IDS keystore in the `INFORMIXDIR/ssl/servername.kdb` directory.

For information on configuration parameters that you need to set on clients, see the *IBM Informix Security Guide*.

STACKSIZE

onconfig.std *value*
32 for 32-bit database servers 64 for 64-bit database servers

units Kilobytes

range of values

32 through limit determined by the database server configuration and the amount of memory available

takes effect

When the database server is shut down and restarted

refer to

- Stacks, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*
- CREATE FUNCTION statement, in the *IBM Informix Guide to SQL: Syntax*

The STACKSIZE parameter specifies the stack size for the database server user threads. The value of STACKSIZE does not have an upper limit, but setting a value that is too large wastes virtual memory space and can cause swap-space problems.

For 32-bit platforms, the default STACKSIZE value of 32 kilobytes is sufficient for nonrecursive database activity. For 64-bit platforms, the recommended STACKSIZE value is 64 kilobytes. When the database server performs recursive database tasks, as in some SPL routines, for example, it checks for the possibility of stack-size overflow and automatically expands the stack.

User threads execute user-defined routines. To increase the stack size for a particular routine, use the **stack** modifier on the CREATE FUNCTION statement.

Warning: Setting the value of *STACKSIZE* too low can cause stack overflow, the result of which is undefined but usually undesirable.

STAGEBLOB

onconfig.std *value*
None

range of values

Up to 128 bytes. STAGEBLOB must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or \$ characters.

takes effect

When the database server is shut down and restarted

refer to *IBM Informix Optical Subsystem Guide*

Use this parameter only if you are storing TEXT or BYTE data on optical storage with the Optical Subsystem. This parameter has no effect on ordinary blobspaces or sbspaces.

STAGEBLOB is the blobspace name for the area where the Optical Subsystem stages TEXT and BYTE data that is destined for storage on optical disk.

STMT_CACHE

onconfig.std *value*
0

if not present

0

range of values

0, 1, or 2

takes effect

When the database server is shut down and restarted

utilities

onmode -e

refer to

- “onmode -e: Change usage of the SQL statement cache” on page 11-11
- Improving query performance, in the *IBM Informix Performance Guide*

STMT_CACHE determines whether the database server uses the SQL statement cache. You can enable the SQL statement cache in one of two modes:

- Always use the SQL statement cache unless a user explicitly specifies not to use it. Set the STMT_CACHE configuration parameter to 2 or **onmode -e ON**.
- Use the SQL statement cache only when a user explicitly specifies to use it. Set the STMT_CACHE configuration parameter to 1 or **onmode -e ENABLE**.

The following table describes the possible values.

Possible Value

Meaning

- | | |
|---|--|
| 0 | SQL statement cache not used (equivalent to onmode -e OFF). |
| 1 | SQL statement cache enabled, but user sessions do not use the cache. Users use the cache only if they set the environment variable STMT_CACHE to 1 or execute the SQL statement SET STATEMENT CACHE ON. |
| 2 | SQL statement cache turned on. All statements are cached. To turn off statement caching, set the environment variable STMT_CACHE to 0 or execute the SQL statement SET STATEMENT CACHE OFF. |

STMT_CACHE_HITS

onconfig.std *value*

0

if not present

0

units Integer

range of values

Any value greater than or equal to 0

takes effect

When the database server is shut down and restarted

utilities

onmode -W STMT_CACHE_HITS onstat (See “onstat -g ssc: Print SQL statement occurrences ” on page 15-124.)

refer to

- “onmode -W: Change settings for the SQL statement cache” on page 11-19
- Improving query performance, in the *IBM Informix Performance Guide*

STMT_CACHE_HITS specifies the number of hits (references) to a statement before it is fully inserted in the SQL statement cache. The following table describes the possible values.

Value Meaning

- | | |
|----|---|
| 0 | Fully insert all qualified statements in the SQL statement cache. |
| >0 | The first time a user issues a unique statement, the database server inserts a <i>key-only</i> entry in the cache that identifies the statement. Subsequent |

identical statements increment the hit count of the *key-only* cache entry. When the hit count of the *key-only* cache entry reaches the specified number of hits, the database server fully inserts the statement in the cache. Set *hits* to 1 or more to exclude ad hoc queries from entering the cache.

STMT_CACHE_NOLIMIT

onconfig.std *value*
0

if not present
1

range of values
0 or 1

takes effect
When the database server is shut down and restarted

utilities
onmode -W STMT_CACHE_NOLIMITonstat (See “onstat -g ssc: Print SQL statement occurrences ” on page 15-124.)

refer to

- “onmode -W: Change settings for the SQL statement cache” on page 11-19
- Improving query performance, in the *IBM Informix Performance Guide*

STMT_CACHE_NOLIMIT controls whether to insert qualified statements into the SQL statement cache. The following table describes the possible values.

Value Meaning

- | | |
|---|--|
| 0 | Prevents statements from being inserted in the cache. The cache can grow beyond the size limit if most of the statements in the cache are currently in use, because the cache cleaning cannot catch up with the insert rate. If you are concerned about memory usage, turn off STMT_CACHE_NOLIMIT to prevent the database server from allocating a large amount of memory for the cache. |
| 1 | Always insert statements in the SQL statement cache regardless of the cache size. |

STMT_CACHE_NUMPOOL

onconfig.std *value*
1

if not present
1

units Positive integer

range of values
1 to 256

takes effect
When the database server is shut down and restarted

refer to Improving query performance, in the *IBM Informix Performance Guide*

STMT_CACHE_NUMPOOL specifies the number of memory pools for the SQL statement cache. To obtain information about these memory pools, use **onstat -g ssc pool**.

Because the database server does not insert not all statements that allocate memory from the memory pools in the cache, the cache size might be smaller than the total size of the memory pools.

STMT_CACHE_SIZE

onconfig.std *value*
512

default size of SQL statement cache
512 kilobytes (524288 bytes)

units Kilobytes

range of values
Positive integer

takes effect
When the database server is shut down and restarted

utilities
onstat -g ssc (Maxsize field)

refer to Improving query performance, in the *IBM Informix Performance Guide*

The STMT_CACHE_SIZE configuration parameter specifies the size of the SQL statement caches in kilobytes. The new cache size takes effect the next time a statement is added to a cache.

STORAGE_FULL_ALARM

onconfig.std *value*
Not in **onconfig.std**

default value
600 3

units *seconds severity_level*

range of values
time_interval alarm_severity
time_interval = 0 (off) or a positive integer indicating the number of seconds between notifications
alarm_severity = 0 (no alarms) or 1 through 5

takes effect
When the database server is shut down and restarted

utilities
None

refer to The section on how to monitor storage spaces is in the chapter on managing disk space in the *IBM Informix Administrator's Guide*.

Use the STORAGE_FULL_ALARM configuration parameter to configure the frequency and severity of messages and alarms when storage spaces become full. When a storage space, such as a dbspace, sbspace, blobspace, or tblspace, or a

partition becomes full, an alarm is raised and a message is sent to the online message log. You can specify the number of seconds between notifications with the first value of this parameter. You can specify the threshold of the severity of the alarm with the second value of this parameter. For more information on alarm levels, see “Event Severity” on page C-3. You can prevent alarms when storage spaces become full by setting this parameter to 0.

Regardless of the value of STORAGE_FULL_ALARM, messages are sent to the online message log when storage spaces or partitions become full.

SYSSALARMPROGRAM

onconfig.std *value*

On UNIX: `$INFORMIXDIR/etc/evidence.sh` On Windows: Not set.
(Commented out.) Listed as `$INFORMIXDIR\etc\evidence.bat`

range of values

Pathname

takes effect

When the database server is shut down and restarted

utilities

None

refer to None

Set SYSSALARMPROGRAM to the full pathname of the **evidence.sh** script. The database server executes **evidence.sh** when a database server failure occurs. Technical Support uses the output from the **evidence.sh** script to diagnose the cause of a database server failure.

On Windows, you must enable command extensions for **evidence.bat** to successfully complete. You can enable and disable the extensions for the Command Prompt you are working in by issuing the following commands:

- Enable: `cmd /x`
- Disable: `cmd /y`

You can also enable and disable command extensions from the Windows XP registry:

Table 1-1. Enabling command extensions from the Windows registry

Attribute	Value
Hive	HKEY_CURRENT_USER
Key	Software\Microsoft\Command Processor
Name	EnableExtensions
Type	REG_DWORD
Values	0 (disable), 1 (enable)

SYSSBSPACENAME

onconfig.std *value*

None

if not present

0

range of values

Up to 128 bytes. SYSSBSPACENAME must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or \$ characters.

takes effect

When disk is initialized (destroys all data)

utilities

onspaces

refer to

- “onspaces -c -S: Create an sbspace” on page 14-11
- “Sbspace Structure” on page 4-23
- Updating statistics, in the chapter on individual query performance in your *IBM Informix Performance Guide*
- Sbspace characteristics, in the chapter on configuration effects on I/O in your *IBM Informix Performance Guide*
- Writing user-defined statistics, in the performance chapter in *IBM Informix User-Defined Routines and Data Types Developer's Guide*
- Providing statistics data for a column, in the *IBM Informix DataBlade API Programmer's Guide*
- “SBSPACENAME” on page 1-86 (specifies the name of the default sbspace)

SYSSBSPACENAME specifies the name of the sbspace in which the database server stores statistics that the UPDATE STATISTICS statement collects for certain user-defined data types. Normally, the database server stores statistics in the **sysdistrib** system catalog table.

Because the data distributions for user-defined data types can be large, you have the option to store them in an sbspace instead of in the **sysdistrib** system catalog table. If you store the data distributions in an sbspace, use DataBlade API or Informix ESQL/C functions to examine the statistics.

Even though you specify an sbspace with the SYSSBSPACENAME parameter, you must create the sbspace with the **-c -S** option of the **onspaces** utility before you can use it. The database server validates the name of this sbspace when one of the following occurs:

- The database server attempts to write data distributions of the multirepresentational type to SYSSBSPACENAME when it executes the UPDATE STATISTICS statement with the MEDIUM or HIGH keywords.
- The database server attempts to delete data distributions of the multirepresentational type to SYSSBSPACENAME when it executes the UPDATE STATISTICS statement with the DROP DISTRIBUTIONS keywords.

Although you can store smart large objects in the sbspace specified in SYSSBSPACENAME, keeping the distribution statistics and smart large objects in separate sbspaces is recommended because:

- You avoid disk contention when queries are accessing smart large objects and the optimizer is using the distributions to determine a query plan.
- Disk space takes longer to fill up when each sbspace is used for a different purpose.

TAPEBLK

onconfig.std *value*

On UNIX: 32 On Windows: 16

units Kilobytes

range of values

Values greater than pagesize/1024

To obtain the page size, see the commands listed in “System Page Size” on page 1-21

takes effect

For **ontape**: when you execute **ontape**For **onload** and **onunload**: when the database server is shut down and restarted

refer to

- Using **onload** and **onunload**, in the *IBM Informix Migration Guide*
- Using **ontape**, in the *IBM Informix Backup and Restore Guide*
- “LTAPEBLK” on page 1-60

TAPEBLK specifies the block size of the device to which **ontape** writes during a storage-space backup. TAPEBLK also specifies the default block size of the device to which data is loaded or unloaded when you use the **onload** or **onunload** utilities. If you are using **onload** or **onunload**, you can specify a different block size on the command line.

The database server does not check the tape device when you specify the block size. Verify that the TAPEBLK tape device can read the block size that you specify. If not, you might not be able to read from the tape.

TAPEDEV

onconfig.std *value*

On UNIX: /dev/tapedev On Windows: \\.\TAPE0

if not present

On UNIX: /dev/null

units Pathname

takes effect

For **ontape**: when you execute **ontape**For **onload** and **onunload**: when the database server is shut down and restarted

refer to

- Using **onload** and **onunload**, in the *IBM Informix Migration Guide*
- Using **ontape**, in the *IBM Informix Backup and Restore Guide*
- “LTAPEDEV” on page 1-61

TAPEDEV specifies the device or directory file system to which **ontape** backs up storage spaces.

TAPEDEV also specifies the default device to which data is loaded or unloaded when you use the **onload** or **onunload** utilities. In Dynamic Server 10.0 and later, you can set TAPEDEV to STDIO to direct back up and restore operations to standard I/O instead of to a device.

If you change the tape device, verify that TAPEBLK and TAPESIZE are correct for the new device.

Using Symbolic Links and a Remote Device (UNIX)

TAPEDEV can be a symbolic link, enabling you to switch between tape devices without changing the pathname that TAPEDEV specifies.

Use the following syntax to specify a tape device attached to another host computer:

```
host_machine_name:tape_device_pathname
```

The following example specifies a tape device on the host computer **kyoto**:

```
kyoto:/dev/rmt01
```

Rewinding Tape Devices Before Opening and on Closing

The tape device that TAPEDEV specifies must perform a rewind before it opens and when it closes. The database server requires this action because of a series of checks that it performs before it writes to a tape.

When the database server attempts to write to any tape other than the first tape in a multivolume dbspace or logical-log backup, the database server first reads the tape header to make sure that the tape is available for use. Then the device is closed and reopened. The database server assumes the tape was rewound when it closed, and the database server begins to write.

Whenever the database server attempts to read a tape, it first reads the header and looks for the correct information. The database server does not find the correct header information at the start of the tape if the tape device did not rewind when it closed during the write process.

TAPESIZE

onconfig.std *value*

0

units Kilobytes

range of values

0 through 2,097,151

takes effect

For **ontape**: when you execute **ontape**For **onload** and **onunload**: when the database server is shut down and restarted

refer to

- Using **onload** and **onunload**, in the *IBM Informix Migration Guide*
- Using **ontape**, in the *IBM Informix Backup and Restore Guide*
- "LTAPESIZE" on page 1-61

Note: Tape size is irrelevant if TAPEDEV is set to STDIO.

The TAPESIZE parameter specifies the size of the device to which **ontape** backs up storage spaces. TAPESIZE also specifies the size of the default device to which data is loaded or unloaded when you use **onload** or **onunload**. If you are using **onload**

or **onunload**, you can specify a different tape size on the command line. If you want to use the full physical capacity of a tape, set TAPESIZE to 0.

TBLSPACE_STATS

onconfig.std *value*
1

if not present
1

units Integer

range of values
0 or 1

takes effect

When the database server is shut down and restarted

The TBLSPACE_STATS configuration parameter turns on and off the collection of tblspace statistics. Use **onstat -g ppf** to list tblspace statistics.

To turn off the collection of tblspace statistics, set TBLSPACE_STATS to 0. When TBLSPACE_STATS is set to 0, **onstat -g ppf** displays "partition profiles disabled." To turn on the collection of tblspace statistics, set TBLSPACE_STATS to 1.

TBLTBLFIRST

onconfig.std *value*
0

units Kilobytes in multiples of page size

range of values

From the equivalent of 250 pages specified in kilobytes to the size of the first chunk minus the space needed for any system objects.

takes effect

When the database server is initialized

Specifies the first extent size of tblspace **tblspace** in the root dbspace. You might want to specify first and next extent sizes to reduce the number of tblspace **tblspace** extents and reduce the frequency of situations when you need to place the tblspace **tblspace** extents in non-primary chunks. (A primary chunk is the initial chunk in a dbspace.) For more information, see specifying first and next extent size in the chapter on managing dbspaces in the *IBM Informix Administrator's Guide*.

You can use **oncheck -pt** and **oncheck -pT** to show the first and next extent sizes of a tblspace **tblspace**. For more information about the **oncheck** utility, see "oncheck -pt and -pT: Display tblspaces for a table or fragment" on page 7-18.

If you want to configure the first extent for a non-root dbspace, see information about the **onspaces** utility in Chapter 14, "The onspaces Utility," on page 14-1.

TBLTBLNEXT

onconfig.std *value*
0

units Kilobytes

range of values

From equivalent of 4 pages specified in kilobytes to the maximum chunk size minus three pages

takes effect

When the database server is initialized

Specifies the next extent size of tblspace **tblspace** in the root dbspace.

If there is not enough space for a next extent in the primary chunk, the extent is allocated from another chunk. If the specified space is not available, the closest available space is allocated. For more information on configuring extent sizes in tblspace **tblspace**, see “TBLTBLFIRST” on page 1-105.

TEMPTAB_NOLOG

onconfig.std *value*

0

takes effect

When the database server is shut down and restarted

range of values

0 = Enable logical logging on temporary table operations 1 = Disable logical logging on temporary table operations

utilities

onmode -wf

refer to “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20

Use the TEMPTAB_NOLOG parameter to disable logging on temporary tables. This parameter can improve performance in application programs, especially in a data replication environment with HDR secondary, RS secondary, or SD secondary servers because it prevents Dynamic Server from transferring temporary tables over the network. The setting can be updated dynamically with the **onmode -wf** utility.

If you enable this setting, be aware that because no data is logged when using temporary tables, rolling back a transaction on a temporary table will no longer undo the work in the temporary table.

TXTIMEOUT

onconfig.std *value*

300

units Seconds

range of values

Positive integers

takes effect

When the database server is shut down and restarted

refer to How the two-phase commit protocol handles failures, in the chapter on multiphase commit protocols in the *IBM Informix Administrator's Guide*

TXTIMEOUT specifies the amount of time that a participant in a two-phase commit waits before it initiates participant recovery.

This parameter is used only for distributed queries that involve a remote database server. Nondistributed queries do not use this parameter.

UNSECURE_ONSTAT

onconfig.std *value*
None

possible values

1 = All users can run **onstat** commands to view running SQL statements

takes effect

When the database server is shut down and restarted

refer to *IBM Informix Administrator's Guide*

The onstat commands that show the SQL statement text that is executing on a session are by default normally restricted to DBSA users. To remove this restriction, set the UNSECURE_ONSTAT configuration parameter to 1. The **onstat** commands that show SQL statements include **onstat -g his**, **onstat -g ses**, **onstat -g stm**, **onstat -g ssc**, and **onstat -g sql**.

USELASTCOMMITTED

onconfig.std *value*
None

range of values

None = No isolation level identified

'Committed Read'

= All transactions from a Committed Read isolation level

'Dirty Read'

= All transactions from a Dirty Read isolation level

All = Both Committed Read and Dirty Read isolation levels

takes effect

When the database server is stopped and restarted.

utilities

onmode -wf or **-wm**

refer to

- “onmode -wf, -wm: Dynamically change certain configuration parameters” on page 11-20
- SET ISOLATION statement information in *IBM Informix Guide to SQL: Syntax*
- Isolation level information in *IBM Informix Performance Guide*

USELASTCOMMITTED specifies the isolation level for which the LAST COMMITTED feature of the COMMITTED READ isolation level is implicitly in effect. The LAST COMMITTED feature can reduce the risk of locking conflicts between concurrent transactions on tables that have exclusive row locks.

USELASTCOMMITTED can also enable LAST COMMITTED semantics for READ COMMITTED and READ UNCOMMITTED isolation levels of the SET TRANSACTION statement.

USELASTCOMMITTED only works with tables that have been created or altered to have ROW as their locking granularity. Tables created without any explicit lock mode setting will use the default setting in DEF_TABLE_LOCKMODE. If DEF_TABLE_LOCKMODE is set to PAGE, the USELASTCOMMITTED configuration parameter cannot enable access to the most recently committed data in tables on which uncommitted transactions hold exclusive locks, unless the tables were explicitly altered to have ROW level of locking granularity. For more details, please refer to “ DEF_TABLE_LOCKMODE” on page 1-30.

The USELASTCOMMITTED parameter is also valid on SD (Shared Disk) secondary database servers. The following table shows valid values for USELASTCOMMITTED on SD secondary servers and their descriptions.

Table 1-2. Valid Secondary Server USELASTCOMMITTED Values

USELASTCOMMITTED value	Description
None	COMMITTED READ LAST COMMITTED is not the default isolation level for sessions
Committed Read	COMMITTED READ LAST COMMITTED is the default isolation level for all sessions with Committed Read isolation
Dirty Read	COMMITTED READ LAST COMMITTED is the default isolation level for all sessions with Dirty Read isolation
All	COMMITTED READ LAST COMMITTED is the default isolation level for all sessions with Committed Read or Dirty Read isolation

You can dynamically change the values of USELASTCOMMITTED through the SET ISOLATION statement or by using **onmode -wf** or **onmode -wm**.

USEOSTIME

onconfig.std *value*
0

range of values
0 = Off 1 = On

takes effect
During initialization

refer to

- Your *IBM Informix Performance Guide*
- Using the CURRENT function to return a datetime value, in the *IBM Informix Guide to SQL: Syntax*

Setting USEOSTIME to 1 specifies that the database server is to use subsecond precision when it obtains the current time from the operating system for SQL statements. The following example shows subseconds in a datetime value:

2001-09-29 12:50:04.612

If subsecond precision is not needed, the database server retrieves the current time from the operating system once per second, making the precision of time for client applications one second. If you set USEOSTIME to 0, the current function returns a zero (.000) for the year to fraction field.

When the host computer for the database server has a clock with subsecond precision, applications that depend on subsecond accuracy for their SQL statements should set USEOSTIME to 1.

Systems that run with USEOSTIME set to nonzero notice a performance degradation of up to 4 to 5 percent compared to running with USEOSTIME turned off.

This setting does not affect any calls regarding the time from application programs to Informix embedded-language library functions.

VP_MEMORY_CACHE_KB

onconfig.std *value*
0

range of values

0 = Off 800 kilobytes per VP to the total kilobytes of the private caches in all VPs, not to exceed 40% of the memory limit as specified in the SHMTOTAL configuration parameter

takes effect

During initialization

utilities

onmode -wf or **-wm onstat -g vpcache**

refer to the CPU Virtual Processor Memory Caches section of the *IBM Informix Performance Guide*

The VP_MEMORY_CACHE_KB parameter enables the database server to access the private memory blocks of your CPU VP.

Dynamic Server uses the VP private memory cache when a thread needs to allocate whole memory blocks. Dynamic Server does not use the private memory cache for buffers in the buffer pool or for any pages read from disk. For example, if you set the VP_MEMORY_CACHE_KB configuration parameter to 1000, the sum of all memory blocks in the cache will be no greater than 1000 kilobytes in size.

VP_MEMORY_CACHE_KB can be changed using the **onmode -wf** or **-wm** options; the changes take effect when the server is restarted. If the **onmode** options are set to 0, the memory caches are emptied. The **onstat -g vpcache** option returns information about CPU VP memory block cache statistics.

VPCLASS

onconfig.std *values*

VPCLASS cpu,num=1,noage

#VPCLASS aio,num=1 (suggested value, but not set)

#VPCLASS jvp,num=1 (suggested value, but not set)

syntax *classname, options*

The *classname* variable is required. VPCLASS has several option fields that can appear in any order, separated by commas. You cannot use any white space in the fields. VPCLASS has the following options:

```
num=num_VPsmax=max_VPsaff=affinity noage noyield
```

For more information about using these options, refer to the individual discussions later in this section.

range of values

Up to 128 bytes. VPCLASS must be unique, begin with a letter or underscore, and contain only digits, letters, underscores, or \$ characters.

takes effect

When the database server is shut down and restarted

utilities

onmode -p (to add or delete VP classes)

refer to

- Specifying user-defined classes of virtual processors, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*
- Specifying a nonyielding user-defined virtual processor (noyield option), in the chapter on virtual processors in the *IBM Informix Administrator's Guide*
- Using **onmode -p** in "onmode -p: Add or remove virtual processors" on page 11-16
- *IBM Informix User-Defined Routines and Data Types Developer's Guide*
- *J/Foundation Developer's Guide*

The VPCLASS parameter allows you to designate a class of virtual processors (VPs), create a user-defined VP, and specify the following information for it:

- The number of virtual processors that the database server should start initially
- The maximum number of virtual processors allowed for this class
- The assignment of virtual processors to CPUs if processor affinity is available
- The disabling of priority aging by the operating system if the operating system implements priority aging

You can put several VPCLASS parameter definitions in your ONCONFIG file. Each VPCLASS parameter describes one class of virtual processors. Put each definition on a separate line, as in the following example:

```
VPCLASS cpu,num=8,aff=0-7,noage
VPCLASS new,num=0
```

Default Values for the VPCLASS Options

The following table shows the defaults and value ranges for the VPCLASS parameter options.

VPCLASS option	Class	Default Value	Range of Values
<i>aio,num</i>	AIO	Not set in onconfig.std . When VPCLASS aio is not set, the number of AIO VPs is determined by the setting of the AUTO_AIOVPS configuration parameter and is limited to 128: <ul style="list-style-type: none"> • If AUTO_AIOVPS is set to 1 (on), the number of AIO VPs initially started is equal to the number of AIO chunks. • If AUTO_AIOVPS is set to 0 (off), the number of AIO VPs started is equal to the greater of 6 or twice the number of AIO chunks. 	1 to 10,000
<i>cpu,num</i>	CPU	1 if MULTIPROCESSOR is 0, 2 otherwise.	1 to 10,000
<i>jvp</i>	JVP	Not set in onconfig.std .	0 to 10,000
<i>num</i>	All other classes	1	1 to 10,000
<i>max_VPs</i>	All	Unlimited.	1 to 10,000
<i>affinity</i>	All	VPs are assigned to available processors in round-robin fashion.	Integers from 0 to (number of CPUs -1)
<i>noage</i>	All	Priority aging is in effect.	noage or omitted
<i>noyield</i>	User defined	Threads will yield.	noyield or omitted

Interaction of VPCLASS with Other Configuration Parameters

Using the VPCLASS parameter instead of the AFF_SPROC, AFF_NPROCS, NOAGE, NUMCPUVPS, and NUMAIOVPS parameters is required. If you use VPCLASS, you must explicitly remove other parameters from your ONCONFIG file:

- If you use VPCLASS **cpu**, remove NUMCPUVPS, AFF_SPROC, AFF_NPROCS, and NOAGE.
- If you use VPCLASS *user-defined*, remove SINGLE_CPU_VP.
- If you use VPCLASS **aio**, remove NUMAIOVPS.

VPCLASS Name

The first item in the VPCLASS parameter provides the name of the virtual-processor class that you are describing. The VPCLASS name is not case sensitive.

You can define new virtual-processor classes for user-defined routines or DataBlade modules, or you can set values for a predefined virtual-processor class. The following virtual-processor classes are predefined by the database server and have specific functions:

```

adm      lio      shm
adt      msc      soc
cpu      ntk      str
jvp      opt      tli
kio      aio      pio
encrypt

```

For VP classes `tli`, `shm`, `str`, and `soc`, you must set `NETTYPE` configuration parameter's `VP_class` field to `NET`.

For example, if the `VPCLASS` parameter is set as:

```

VPCLASS shm,num=1
VPCLASS tli,num=1

```

then the `NETTYPE` parameter should be set as follows:

```

NETTYPE ipcshm,,3,NET
NETTYPE ipctli,,3,NET

```

For more information on the `NETTYPE` configuration parameter, see “`NETTYPE`” on page 1-69.

The following example specifies that the database server should start three virtual processors of the `CPU` class:

```

VPCLASS cpu,num=3

```

The `JVP` option of the `VPCLASS` configuration parameter sets the number of Java virtual processors. This parameter is required when you use the IBM Informix JDBC Driver. On UNIX, you must define multiple Java virtual processors to execute Java user-defined routines in parallel.

Creating a User-Defined Class

The `VPCLASS` configuration parameter also allows you to create a class of user-defined virtual processors (VPs). A user-defined class of VPs can run ill-behaved user-defined routines (UDRs).

Warning: Execution of an ill-behaved routine in the *CPU VP* can cause serious interference with the operation of the database server. In addition, the routine itself might not produce correct results.

For more information on ill-behaved UDRs, see user-defined classes of virtual processors, in the chapter on virtual processors in the *IBM Informix Administrator's Guide*.

You might want to describe a user-defined class of virtual processors to run `DataBlade` or user-defined routines. The following example creates the user-defined class **new**, for which the database server starts three virtual processors initially:

```

VPCLASS new,num=3

```

At a later time, you can use **onmode -p** to add virtual processors to the class. The following command adds three virtual processors to the **new** class:

```

onmode -p +3 new

```

Tip: When you create a user-defined routine or function, you use the `CLASS` parameter of the `CREATE FUNCTION` statement to assign it to a class of virtual processors. You must ensure that the name of the user-defined class agrees with

the name that you assigned in the CREATE FUNCTION statement. If you try to use a function that refers to a user-defined class, that class must exist and have virtual processors assigned to it. If the class does not have any virtual processors, you receive an SQL error.

For more information on how to assign a user-defined routine to either CPU or user-defined classes of virtual processors, refer to *IBM Informix User-Defined Routines and Data Types Developer's Guide*. For more information on the syntax of the CREATE FUNCTION or CREATE PROCEDURE statement, refer to the *IBM Informix Guide to SQL: Syntax*.

Using the **noyield** Option

By default, the VPCLASS parameter defines a yielding VP class, which allows the C UDR to yield to other threads that need access to the user-defined VP class. A UDR can perform blocking I/O calls if it executes in a yielding user-defined VP. However, it must still yield for other threads to have access to the VP.

You can also define nonyielding user-defined VPs with the **noyield** option of VPCLASS. The **noyield** option specifies creation of a nonyielding user-defined VP class. A nonyielding user-defined VP class executes a user-defined routine in a way that gives the routine exclusive use of the virtual-processor class. In other words, user-defined routines that use a **noyield** virtual-processor class run serially. They never yield the VP to another thread.

You do not need to specify more than one VP in a nonyielding user-defined VP class, because the UDR runs on a single VP until it completes and any additional virtual processors would be idle.

Important: If your UDR uses global variables, only one VP in the user-defined virtual-processor class should be nonyielding.

The following example specifies a user-defined class of virtual processors called **new_noyield**, which runs in no-yield mode:

```
VPCLASS new_noyield,noyield,num=1
```

The **noyield** option applies only to user-defined VP classes. The database server ignores **noyield** if it is part of a VPCLASS parameter that defines a predefined VP class such as CPU, AIO, and so on.

Using the **num** Option

The *num* option sets the number of virtual processors of the specified class that the database server should start during initialization.

On a single-processor computer, allocate only one CPU virtual processor. On a multiprocessor computer, do not allocate more CPU and user-defined virtual processors, combined, than there are CPUs on the computer.

Use the following syntax to specify the number of virtual processors:

```
num=num_VPs
```

Specifying the Number of CPU VPs

For example, the following parameter specifies that the database server should start four virtual processors for the **cpu** class:

```
VPCLASS cpu,num=4
```

At a later time, you can use the **onmode -p** command to add virtual processors for the class.

Using the max_VPs Option

The *max_VPs* option specifies the maximum number of virtual processors that the database server can start for the class.

Use the following syntax to specify the number of virtual processors:

```
max=max_VPs
```

The value can be any integer greater than 0. If you omit the *max_VPs* option, the number is unlimited.

Using the affinity Option

On multiprocessor computers that support *processor affinity*, the affinity option specifies the CPUs to which the database server binds virtual processors.

The affinity option has the following two forms:

```
aff=processor_number  
aff=start_range,end_range
```

In the first form, the database server binds all virtual processors in the class to the CPU numbered *processor_number*. (On a multiprocessor system, the operating system numbers the CPUs from 0 to (number of CPUs-1)). In the second form, the database server assigns the virtual processors of the class to processors in the range *start_range* to *end_range*, inclusive. The value *end_range* must be larger than *start_range*, and all values must be less than the total number of available CPUs.

For example, if your platform has eight CPUs, your ONCONFIG file might include the following VPCLASS entries:

```
VPCLASS    first,aff=3  
VPCLASS    second,num=3,aff=5-7  
VPCLASS    cpu,num=8,aff=0-7,noage
```

For more information about using processor affinity, refer to the chapter on virtual processors in the *IBM Informix Administrator's Guide*.

WSTATS

onconfig.std *value*
0

range of values

0 = Disable wait statistics 1 = Enable wait statistics

takes effect

When the database server is shut down and restarted

utilities

onstat -g wst

refer to

- “onstat -g wst: Print wait statistics for threads ” on page 15-134

WSTATS specifies the ability of **onstat -g wst** to print wait statistics for threads within the system. You should expect a small performance impact due to the cost of gathering statistical information and enabling WSTATS for production systems is not recommended.

Chapter 2. The sysmaster Database

In This Chapter

This chapter describes the **sysmaster** database and contains reference information for the *system-monitoring interface* (SMI). It provides information on the following topics:

- What is the **sysmaster** database
- How to use SMI tables
- Descriptions of the SMI tables
- A map of the documented SMI tables

For information about the ON-Bar tables, see the *IBM Informix Backup and Restore Guide*.

The sysmaster Database

The database server creates and maintains the **sysmaster** database. It is analogous to the system catalog for databases, which is described in the *IBM Informix Guide to SQL: Reference*. Just as a system catalog for every database managed by the database server keeps track of objects and privileges in the database, a **sysmaster** database for every database server keeps track of information about the database server.

The **sysmaster** database contains the *system-monitoring interface* (SMI) tables. The SMI tables provide information about the state of the database server. You can query these tables to identify processing bottlenecks, determine resource usage, track session or database server activity, and so on. This chapter describes these tables, which are slightly different from ordinary tables.

Warning: The database server relies on information in the **sysmaster** database. Do not change any of the tables in **sysmaster** or any of the data within the tables. Such changes could cause unpredictable and debilitating results.

The database server creates the **sysmaster** database when it initializes disk space. The database server creates the database with unbuffered logging. You cannot drop the database or any of the tables in it, and you cannot turn logging off.

As user **informix** on UNIX or a member of the **Informix-Admin** group on Windows, you can create SPL routines in the **sysmaster** database. (You can also create triggers on tables within **sysmaster**, but the database server never executes those triggers.)

Joins of multiple tables in **sysmaster** might return inconsistent results because the database server does not lock the tables during a join. You can join **sysmaster** tables with tables in other databases. However, to join **sysmaster** tables with tables in a nonlogging database, first make the nonlogging database the current database.

The buildsmi Script

When you bring the database server up for the first time, it runs a script called **buildsmi**, which is in the **etc** directory. This script builds the database and tables

that support SMI. The database server requires approximately 1750 free pages of logical-log space to build the **sysmaster** database.

If you receive an error message that directs you to run the **buildsmi** script, a problem probably occurred while the database server was building the SMI database, tables, and views. When you use **buildsmi**, the existing **sysmaster** database is dropped and then re-created.

This script must be run as user **informix** on UNIX, or as a member of the **Informix-Admin** group on Windows, after ensuring that no connections to the **sysmaster** database are made during the build of the database. For example, if a Scheduler task is running when the **buildsmi** script commences, the script fails when the Scheduler attempts to access any of the **sysmaster** tables.

Errors issued while the **buildsmi** script runs are written (on UNIX) to the file **/tmp/buildsmi.out**, or on Windows to the file **%INFORMIXDIR%\etc\buildsmi_out.%INFORMIXSERVER%**, where **%INFORMIXSERVER%** is the name of the Dynamic Server instance.

The **bdutil.sh** Script

When you initialize the database server for the first time, it runs a script called **bdutil.sh** on UNIX or **bdutil.bat** on Windows. This script builds the **sysutils** database. If it fails, the database server creates an output file in the **tmp** directory. The output file is **bdutil.process_id** on UNIX and **bdutil.out** on Windows. The messages in this output file reflect errors that occurred during the script execution.

The System-Monitoring Interface

This section describes the SMI tables and how you access them to monitor the database server operation.

Understanding the SMI Tables

The SMI (system-monitoring interface) consists of tables and pseudo-tables that the database server maintains automatically. While the SMI tables appear to the user as tables, they are not recorded on disk as normal tables are. Instead, the database server constructs the tables in memory, on demand, based on information in shared memory at that instant. When you query an SMI table, the database server reads information from these shared-memory structures. Because the database server continually updates the data in shared memory, the information that SMI provides lets you examine the current state of your database server.

The SMI tables provide information about the following topics:

- Auditing
- Checkpoints
- Chunk I/O
- Chunks
- Database-logging status
- Dbspaces
- Disk usage
- Environment variables
- Extents
- Locks

- Networks
- SQL statement cache statistics
- SQL tracing
- System profiling
- Tables
- User profiling
- Virtual-processor CPU usage

The data in the SMI tables changes dynamically as users access and modify databases that the database server manages.

Accessing SMI Tables

Any user can use SQL SELECT statements to query an SMI table, but standard users cannot execute statements other than SELECT. Attempts to do so result in permission errors. The administrator can execute SQL statements other than SELECT, but the results of such statements are unpredictable.

Dynamic Server provides the **sysadinfo** and **sysaudit** tables. Only user **informix** on UNIX or members of the **Informix-Admin** group on Windows can query **sysadinfo** and **sysaudit**.

You cannot use **dbschema** or **dbexport** on any of the tables in the **sysmaster** database. If you do, the database server generates the following error message:

```
Database has pseudo tables - can't build schema
```

SELECT Statements

You can use SELECT statements on SMI tables wherever you can use SELECT against ordinary tables (from DB–Access, in an SPL routine, with Informix ESQ/C, and so on) with one restriction: you cannot (meaningfully) reference **rowid** when you query SMI tables. SELECT statements that use **rowid** do not return an error, but the results are unpredictable.

All standard SQL syntax, including joins between tables, sorting of output, and so on, works with SMI tables. For example, if you want to join an SMI table with a non-SMI table, name the SMI table with the following standard syntax:

```
sysmaster[@dbservername]:[owner.]tablename
```

Triggers and Event Alarms

Triggers based on changes to SMI tables never run. Although you can define triggers on SMI tables, triggers are activated only when an INSERT, UPDATE, or DELETE statement occurs on a table. The updates to the SMI data occur within the database server, without the use of SQL, so a trigger on an SMI table is never activated, even though the data returned by a SELECT statement indicates that it should be.

To create an event alarm, query for a particular condition at predefined intervals, and execute an SPL routine if the necessary conditions for the alarm are met.

SPL and SMI Tables

You can access SMI tables from within a SPL routine. When you reference SMI tables, use the same syntax that you use to reference a standard table.

Locking and SMI Tables

The information in the SMI tables changes based on the database server activity. However, the database server does not update the information using SQL statements. When you use SMI tables with an isolation level that locks objects, it prevents other users from accessing the object, but it does not prevent the data from changing. In this sense, all the SMI tables have a permanent Dirty Read isolation level.

The System-Monitoring Interface Tables

The database server supports the following SMI tables.

Table	Description	Reference
<code>sysadinfo</code>	Auditing configuration information	page “ <code>sysadinfo</code> ” on page 2-7
<code>sysaudit</code>	Auditing event masks	page “ <code>sysadinfo</code> ” on page 2-7
<code>syscheckpoint</code>	Checkpoint information	page “ <code>syscheckpoint</code> ” on page 2-8
<code>syschkio</code>	Chunk I/O statistics	page “ <code>syschkio</code> ” on page 2-8
<code>syschunks</code>	Chunk information	page “ <code>syschunks</code> ” on page 2-9
<code>syscmsmsla</code>	Connection Manager information	page “ <code>syscmsmsla</code> ” on page 2-10
<code>syscmsmtab</code>	Connection Manager information	page “ <code>syscmsmtab</code> ” on page 2-11
<code>sysconfig</code>	Configuration information	page “ <code>sysconfig</code> ” on page 2-11
<code>sysdatabases</code>	Database information	page “ <code>sysdatabases</code> ” on page 2-11
<code>sysdbslocale</code>	Locale information	page “ <code>sysdbslocale</code> ” on page 2-12
<code>sysdbspaces</code>	Dbpace information	page “ <code>sysextents</code> ” on page 2-14
<code>sysdri</code>	Data-replication information	page “ <code>sysdri</code> ” on page 2-13
<code>sysdual</code>	Is a single-row table	page “ <code>sysdual</code> ” on page 2-14
<code>sysenv</code>	Online server’s startup environment	page “ <code>sysenv</code> ” on page 2-14
<code>sysenvses</code>	Session-level environment variable	page “ <code>sysenvses</code> ” on page 2-14
<code>sysextents</code>	Extent-allocation information	page “ <code>sysextents</code> ” on page 2-14
<code>sysextspaces</code>	External spaces information	page “ <code>sysextspaces</code> ” on page 2-14
<code>syssha_latency</code>	Secondary server latency statistics	page “ <code>syssha_latency</code> ” on page 2-15

Table	Description	Reference
syssha_type	Information about connected servers	page " syssha_type" on page 2-16
syssha_workload	Secondary server workload statistics	page " syssha_workload" on page 2-16
sysipl	Index page logging information	page " sysipl" on page 2-17
syslocks	Active locks information	page " syslocks" on page 2-17
syslogs	Logical-log file information	page " syslogs" on page 2-18
sysmgminfo	Memory Grant Manager/Parallel Data Query information	page " sysmgminfo" on page 2-18
sysnetclienttype	Client type network activity	page " sysnetclienttype" on page 2-19
sysnetglobal	Global network information	page " sysnetglobal" on page 2-19
sysnetworkio	Network I/O	page " sysnetworkio" on page 2-20
sysonlineelog	Online log information	page " sysonlineelog" on page 2-21
sysprofile	System-profile information	page " sysprofile" on page 2-21
sysproxyagents	information about all the proxy agent threads	page "sysproxyagents" on page 2-22
sysproxydistributors	Proxy distributor information	page "sysproxydistributors" on page 2-23
sysproxysessions	Information about sessions using redirected-write functionality	page " sysproxysessions" on page 2-23
sysproxytxnops	Information about transactions operations executing via each proxy distributor	page "sysproxytxnops" on page 2-24
sysproxytxns	Information about all of the current transactions executing via each proxy distributor	page "sysproxytxns" on page 2-24
sysptprof	Table information	page " sysptprof" on page 2-25
sysrepevtreg	Post events to Connection Manager and to the OpenAdmin tool	page " sysrepevtreg" on page 2-25
sysrepstats	Post events to Connection Manager and to the OpenAdmin tool	page " sysrepstats" on page 2-26
sysrsslog	RS secondary server information	page " sysrsslog" on page 2-28
syssscb1st	Memory by user	page " syssscb1st" on page 2-29
sysssesprof	Counts of various user actions	page " sysssesprof" on page 2-29

Table	Description	Reference
sysseappinfo	Distributed Relational Database Architecture (DRDA) client-session information.	page "sysseappinfo" on page 2-29
sysessions	Description of each user connected	page "sysessions" on page 2-30
sysmx	SMX (server multiplexer group) connection information	page "sysmx" on page 2-31
sysmxses	SMX (server multiplexer group) session information	page "sysmxses" on page 2-32
sysqltrace	SQL statement information	page "sysqltrace" on page 2-32
sysqltrace_info	SQL profile trace system information	page "sysqltrace_info" on page 2-33
sysqltrace_iter	SQL statement iterators	page "sysqltrace_iter" on page 2-33
sysrcrss	RS secondary server statistics	page "sysrcrss" on page 2-34
sysrcsds	SD secondary server statistics	page "sysrcsds" on page 2-34
systabnames	Database, owner, and table name for the tblspace tblspace	page "systabnames" on page 2-35
systhreads	Wait statistics	page "systhreads" on page 2-35
sysstrgrss	RS secondary server statistics	page "sysstrgrss" on page 2-35
sysstrgsds	SD secondary server statistics	page "sysstrgsds" on page 2-36
sysvpprof	User and system CPU used by each virtual processor	page "sysvpprof" on page 2-36

Many other tables in the **sysmaster** database are part of the system-monitoring interface but are not documented. Their schemas and column content can change from version to version. The **flags_text** table now contains more rows. To view the new rows you must first drop and then recreate the **sysmaster** database.

The sysutils Tables

ON-Bar uses the following tables in the **sysutils** database. For more information, see the *IBM Informix Backup and Restore Guide*.

Table Description

bar_action

Lists all backup and restore actions that are attempted against an object, except during a cold restore. Use the information in this table to track backup and restore history.

bar_instance

Writes a record to this table for each successful backup. ON-Bar might later use the information for a restore operation.

bar_object

Describes each backup object. This table provides a list of all storage spaces and logical logs from each database server for which at least one backup attempt was made.

bar_server

Lists the database servers in an installation. This table is used to ensure that backup objects are returned to their proper places during a restore.

sysadinfo

The **sysadinfo** table contains information about the auditing configuration for the database server. For more information, see your *IBM Informix Security Guide*. You must be user **informix** or user **root** on UNIX or a member of the **Informix-Admin** group on Windows to retrieve information from the **sysadinfo** table.

Column	Type	Description
adtmode	integer	Controls the level of auditing.
adterr	integer	Specifies how the database server behaves when it encounters an error while it writes an audit record.
adtsize	integer	Maximum size of an audit file
adtpath	char(256)	Directory where audit files are written
adtfile	integer	Number of the audit file

sysaudit

For each defined audit mask (that is, for each *username*), the **sysaudit** table contains flags that represent the database events that generate audit records. The **success** and **failure** columns represent the bitmasks that compose the audit masks. If a bit is set in both the **success** and **failure** columns, the corresponding event generates an audit record whether or not the event succeeded.

You must be user **informix** or user **root** on UNIX or a member of the **Informix-Admin** group on Windows to retrieve information from the **sysaudit** table.

Use the **onaudit** utility to list or modify an audit mask. For information about **onaudit** and auditing, see your *IBM Informix Security Guide*.

Column	Type	Description
username	char(32)	Name of the mask
succ1	integer	Bitmask of the audit mask for success
succ2	integer	Bitmask of the audit mask for success
succ3	integer	Bitmask of the audit mask for success
succ4	integer	Bitmask of the audit mask for success
succ5	integer	Bitmask of the audit mask for success
fail1	integer	Bitmask of the audit mask for failure
fail2	integer	Bitmask of the audit mask for failure
fail3	integer	Bitmask of the audit mask for failure
fail4	integer	Bitmask of the audit mask for failure

Column	Type	Description
fail5	integer	Bitmask of the audit mask for failure

syschkio

The **syschkio** table provides I/O statistics for individual chunks that the database server manages.

Column	Type	Description
chunknum	smallint	Chunk number
reads	integer	Number of physical reads
pagesread	integer	Number of pages read
writes	integer	Number of physical writes
pageswritten	integer	Number of pages written
mreads	integer	Number of physical reads (mirror)
mpagesread	integer	Number of pages read (mirror)
mwrites	integer	Number of physical writes (mirror)
mpageswritten	integer	Number of pages written (mirror)

syscheckpoint

The **syscheckpoint** table provides information and statistics about checkpoints.

Column	Type	Description
interval	integer	Number of checkpoints since the server was started
type	char(12)	Hard or Interval
caller	char(10)	Caller of the checkpoint
clock_time	integer	Time of day the checkpoint occurred
crit_time	float	Time spent waiting for the critical section to be released
flush_time	float	Time spent flushing pages to disk
cp_time	float	Duration from checkpoint pending until checkpoint done
n_dirty_buffs	integer	Number of dirty buffers
plogs_per_sec	integer	Number of physical log pages processed in a second
llogs_per_sec	integer	Number of logical log pages processed in a second
dskflush_per_sec	integer	Number of buffer pool pages flushed in a second
ckpt_logid	integer	Unique id of the logical log at the checkpoint
ckpt_logpos	integer	Position of the logical log at the checkpoint
physused	integer	Number of pages used in the physical log
logused	integer	Number of pages used in the logical log
n_crit_waits	integer	Number of users who had to wait to enter a critical section
tot_crit_wait	float	Duration spent waiting for all users waiting at the checkpoint critical section block
longest_crit_wait	float	Longest critical section wait

Column	Type	Description
block_time	float	Duration of the checkpoint that blocked the system

syschunks

The **syschunks** table describes each of the chunks that the database server manages. In the **flags** and **mflags** columns, each bit position represents a separate flag. Thus, it might be easier to read values in the **flags** and **mflags** columns if the values are returned using the HEX function.

Column	Type	Description		
chcknum	smallint	Chunk number		
dbspace	smallint	Dbspace number		
nxchcknum	smallint	Number of the next chunk in this dbspace		
chcksize	integer	Number of pages in this chunk (in units of system default page size)		
offset	integer	Page offset of the chunk in its device or path		
pagesize	integer	Page size (in bytes)		
nfree	integer	Number of free pages in the chunk (in units of system default page size)		
is_offline	integer	1 If the chunk is offline, 0 if not		
is_recovering	integer	1 If the chunk is being recovered, 0 if not		
is_blobchunk	integer	1 If the chunk is in a blobspace, 0 if not		
is_sbchunk	integer	1 If the chunk is a sbpace, 0 if not		
is_inconsistent	integer	1 If the chunk is undergoing logical restore, 0 if not		
flags	smallint	Flags	Hexadecimal	Meaning
		16	0x0010	Chunk is a mirrored chunk
		32	0x0020	Chunk is in offline mode
		64	0x0040	Chunk is in online mode
		128	0x0080	Chunk is in recovery mode
		256	0x0100	Chunk has just been mirrored
		512	0x0200	Chunk is part of a blobspace
		1024	0x0400	Chunk is being dropped
		2048	0x0800	Chunk is part of an optical stageblob
		4096	0x1000	Chunk is inconsistent
		16384	0x4000	Chunk contains temporary log space
32768	0x8000	Chunk was added during roll forward		
fname	char(256)	Pathname for the file or device of this chunk		
mdsize	integer	Size in pages of the metadata area of a chunk that belongs to a smart blobspace. If the chunk does not belong to a smart blobspace, the column stores -1.		

Column	Type	Description
mfname	char(256)	Pathname for the file or device of the mirrored chunk, if any
moffset	integer	Page offset of the mirrored chunk
mis_offline	integer	1 If mirror is offline, 0 if not
mis_recovering	integer	1 If mirror is being recovered, 0 if not
mflags	smallint	Mirrored chunk flags; values and meanings are the same as the flags column.
udfree	integer	Free space in pages within the user data area of a chunk that belongs to a smart blobspace. If the chunk does not belong to a smart blobspace, the column stores -1.
udsize	integer	Size in pages of the user data area of a chunk that belongs to a smart blobspace. If the chunk does not belong to a smart blobspace, the column stores -1.

syscsm

The **syscsm** table is a view of the **syscsmstab** and **syscsmsla** tables. It contains Connection Manager service level agreement (SLA) information. The table is updated once every five seconds.

Column	Type	Description
sid	integer	Connection Manager session ID
name	char(128)	Connection Manager name
host	char(256)	Host name
foc	char(128)	Failover configuration (FOC)
flag	integer	Arbitrator flag. A value of 1 indicates that the Connection Manager Arbitrator is active. A value of 0 indicates that the Arbitrator is inactive.
sla_id	integer	Connection Manager service level agreement (SLA) ID
sla_name	char(128)	SLA name
sla_define	char(128)	SLA define
connections	integer	Number of connections made through Connection Manager

syscsmsla

The **syscsmsla** table contains Connection Manager service level agreement (SLA) information. The table is updated once every five seconds.

Column	Type	Description
address	int8	CMSLA internal address
sid	integer	Connection Manager session ID
sla_id	integer	Connection Manager service level agreement (SLA) ID
sla_name	char(128)	SLA name
sla_define	char(128)	SLA define

Column	Type	Description
connections	integer	Number of connections made through Connection Manager

syscsmtab

The **syscsmtab** table contains Connection Manager information.

Column	Type	Description
address	int8	Connection Manager internal address
sid	integer	Connection Manager session ID
name	char(128)	Connection Manager name
host	char(256)	Host name
foc	char(128)	Failover configuration (FOC)
flag	integer	Arbitrator flag. A value of 1 indicates that the Connection Manager Arbitrator is active. A value of 0 indicates that the Arbitrator is inactive.

sysconfig

The **sysconfig** table describes the effective, original, and default values of the configuration parameters. For more information about the ONCONFIG file and the configuration parameters, see Chapter 1, "Configuration Parameters," on page 1-1.

Column	Type	Description
cf_id	integer	Unique numeric identifier
cf_name	char(128)	Configuration parameter name
cf_flags	integer	Reserved for future use
cf_original	char(256)	Value in the ONCONFIG file at boot time
cf_effective	char(256)	Value currently in use
cf_default	char(256)	Value provided by the database server if no value is specified in the ONCONFIG file

sysdatabases

The **sysdatabases** table describes each database that the database server manages.

Column	Type	Description
name	char(128)	Database name
partnum	integer	The partition number (tblspace identifier) for the systables table for the database
owner	char(32)	User ID of the creator of the database
created	date	Date created
is_logging	integer	1 If logging is active, 0 if not
is_buff_log	integer	1 If buffered logging, 0 if not
is_ansi	integer	1 If ANSI-compliant, 0 if not

Column	Type	Description	
is_nls	integer	1 If GLS-enabled, 0 if not	
flags	smallint	Logging flags (hex values)	
		0	No logging
		1	Unbuffered logging
		2	Buffered logging
		4	ANSI-compliant database
		8	Read-only database
		10	GLS database
		20	Checking of the logging mode of syscdr database bypassed
		100	Changed status to buffered logging
		200	Changed status to unbuffered logging
		400	Changed status to ANSI compliant
800	Database logging turned off		
1000	Long ID support enabled		

sysdbslocale

The **sysdbslocale** table lists the locale of each database that the database server manages.

Column	Type	Description
db_s_dbname	char(128)	Database name
db_s_collate	char(32)	The locale of the database

sysdbspaces

The **sysdbspaces** table describes each of the dbspaces that the database server manages. In the **flags** column, each bit position represents a separate flag. Thus, it might be easier to read values in the **flags** column if the values are returned using the HEX function.

Column	Type	Description
db_snum	smallint	Dbspace number
name	char(128)	Dbspace name
owner	char(32)	User ID of owner of the dbspace
fchunk	smallint	Number of the first chunk in the dbspace
nchunks	smallint	Number of chunks in the dbspace
pagesize	integer	Page size
is_mirrored	integer	1 If dbspace is mirrored, 0 if not
is_blobspace	integer	1 If the dbspace is a blobspace, 0 if not
is_sbspace	integer	1 If the dbspace is a sbspace, 0 if not
is_temp	integer	1 If the dbspace is a temporary dbspace, 0 if not

Column	Type	Description		
		Flags	Hexadecimal	Meaning
flags	smallint	1	0x0001	Dbospace has no mirror
		2	0x0002	Dbospace uses mirroring
		4	0x0004	Dbospace mirroring is disabled
		8	0x0008	Dbospace is newly mirrored
		16	0x0010	Space is a blobospace
		32	0x0020	Blobospace is on removable media
		64	0x0040	Blobospace is on optical media
		128	0x0080	Blobospace has been dropped.
		256	0x0100	Blobospace is an optical stageblob
		512	0x0200	Space is being recovered
		1024	0x0400	Space has been physically recovered
		2048	0x0800	Space is in logical recovery
		32768	0x8000	Space is an sbospace

sysdri

The **sysdri** table provides information about the High-Availability Data-Replication status of the database server.

Column	Type	Description
type	char(50)	High-Availability Data Replication type Possible values: <ul style="list-style-type: none"> • primary • secondary • standard • not initialized
state	char(50)	State of High-Availability Data Replication Possible values: <ul style="list-style-type: none"> • off • on • connecting • failure • read-only
name	char(128)	The name of the other database server in the High-Availability Data-Replication pair
intvl	integer	The High-Availability Data-Replication interval
timeout	integer	The High-Availability Data-Replication timeout value for this database server
lostfound	char(256)	The pathname to the lost-and-found file

sysdual

The **sysdual** table returns exactly one column and one row.

Column	Type	Description
dummy	char(1)	Dummy columns returning "X"

sysenv

The **sysenv** table displays the startup environment settings of the database server.

Column	Type	Description
env_id	integer	Identifier variable number
env_name	char(128)	Environment variable name
env_value	char(512)	Environment variable value

sysenvses

The **sysenvses** table displays the environment variable at the session level.

Column	Type	Description
envses_sid	integer	Session id
envses_id	integer	Identifier variable number
envses_name	char(128)	Session environment variable name
envses_value	char(512)	Session environment variable value

sysextents

The **sysextents** table provides information about extent allocation.

Column	Type	Description
dbsname	char(128)	Database name
tablename	char(128)	Table name
chunk	integer	Chunk number
offset	integer	Number of pages into the chunk where the extent begins
size	integer	Size of the extent, in pages

sysextspaces

The **sysextspaces** table provides information about external spaces. Indexes for the **id** column and the **name** column allow only unique values.

Column	Type	Description
id	integer	External space ID
name	char(128)	External space name

Column	Type	Description
owner	char(32)	External space owner
flags	integer	External space flags (reserved for future use)
refcnt	integer	External space reference count.
locsize	integer	Size of external space location, in bytes
location	char (256)	Location of external space

sysha_latency

The `sysha_latency` table provides a history of the amount of time that it took to apply a log record on any of the secondary nodes. This table contains a history of the last 20 samplings performed for a particular secondary server.

Column	Type	Description
lt_secondary	char(128)	Name of secondary server
lt_time_last_update	integer	Time at which log record was last updated
lt_lagtime_1	float	Amount of time required to apply log record for the most recent five-second interval
lt_lagtime_2	float	Amount of time required to apply log record for the second most recent five-second interval
lt_lagtime_3	float	Amount of time required to apply log record for the third most recent five-second interval
lt_lagtime_4	float	Amount of time required to apply log record for the fourth most recent five-second interval
lt_lagtime_5	float	Amount of time required to apply log record for the fifth most recent five-second interval
lt_lagtime_6	float	Amount of time required to apply log record for the sixth most recent five-second interval
lt_lagtime_7	float	Amount of time required to apply log record for the seventh most recent five-second interval
lt_lagtime_8	float	Amount of time required to apply log record for the eighth most recent five-second interval
lt_lagtime_9	float	Amount of time required to apply log record for the ninth most recent five-second interval
lt_lagtime_10	float	Amount of time required to apply log record for the tenth most recent five-second interval
lt_lagtime_11	float	Amount of time required to apply log record for the eleventh most recent five-second interval
lt_lagtime_12	float	Amount of time required to apply log record for the twelfth most recent five-second interval
lt_lagtime_13	float	Amount of time required to apply log record for the thirteenth most recent five-second interval
lt_lagtime_14	float	Amount of time required to apply log record for the fourteenth most recent five-second interval
lt_lagtime_15	float	Amount of time required to apply log record for the fifteenth most recent five-second interval
lt_lagtime_16	float	Amount of time required to apply log record for the sixteenth most recent five-second interval

Column	Type	Description
lt_lagtime_17	float	Amount of time required to apply log record for the seventeenth most recent five-second interval
lt_lagtime_18	float	Amount of time required to apply log record for the eighteenth most recent five-second interval
lt_lagtime_19	float	Amount of time required to apply log record for the nineteenth most recent five-second interval
lt_lagtime_20	float	Amount of time required to apply log record for the twentieth most recent five-second interval

syssha_type

The **syssha_type** table is a single row table that is used to describe the type of server that is connected.

Column	Type	Description
ha_type	integer	Server type (see table below)
ha_primary	char(128)	Server name (see table below)

Table 2-1. Descriptions for the values in the **syssha_type** table

Value of <i>ha_type</i>	Value of <i>ha_primary</i>	Description
0	NULL	Not part of a high-availability environment
1	<primary server name>	Primary server
2	<primary server name>	HDR secondary server
3	<primary server name>	SD secondary server
4	<primary server name>	RS secondary server

syssha_workload

The **syssha_workload** table contains workload statistics on each of the secondary servers.

Column	Type	Description
wl_secondary	char(128)	Name of secondary server
wl_time_last_update	integer	Time at which workload last updated
wl_type	char(12)	This row contains the ready queue size, user CPU time, and system CPU time
wl_workload_1	float	Most recent workload activity
wl_workload_2	float	Second most recent workload activity
wl_workload_3	float	Third most recent workload activity
wl_workload_4	float	Fourth most recent workload activity
wl_workload_5	float	Fifth most recent workload activity
wl_workload_6	float	Sixth most recent workload activity
wl_workload_7	float	Seventh most recent workload activity

Column	Type	Description
wl_workload_8	float	Eighth most recent workload activity
wl_workload_9	float	Ninth most recent workload activity
wl_workload_10	float	Tenth most recent workload activity
wl_workload_11	float	Eleventh most recent workload activity
wl_workload_12	float	Twelfth most recent workload activity
wl_workload_13	float	Thirteenth most recent workload activity
wl_workload_14	float	Fourteenth most recent workload activity
wl_workload_15	float	Fifteenth most recent workload activity
wl_workload_16	float	Sixteenth most recent workload activity
wl_workload_17	float	Seventeenth most recent workload activity
wl_workload_18	float	Eighteenth most recent workload activity
wl_workload_19	float	Nineteenth most recent workload activity
wl_workload_20	float	Twentieth most recent workload activity

sysipl

The **sysipl** table provides information about the status of index page logging at the primary server.

Column	Type	Description
ipl_status	integer	Index page logging status
ipl_time	integer	Time at which index page logging was enabled

syslocks

The **syslocks** table provides information about all the currently active locks in the database server.

Column	Type	Description	
dbname	char(128)	Database name	
tablename	char(128)	Table name	
rowidlk	integer	Real rowid, if it is an index key lock	
keynum	smallint	Key number of index key lock	
type	char(4)	Type of lock	
		B	Byte lock
		IS	Intent shared lock
		S	Shared lock
		XS	Shared key value held by a repeatable reader
		U	Update lock
		IX	Intent exclusive lock
		SIX	Shared intent exclusive lock
		X	Exclusive lock
XR	Exclusive key value held by a repeatable reader		

Column	Type	Description
owner	integer	Session ID of the lock owner
waiter	integer	Session ID of the user waiting for the lock. If more than one user is waiting, only the first session ID appears.

syslogs

The **syslogs** table provides information about space use in logical-log files. In the **flags** column, each bit position represents a separate flag. For example, for a log file, the **flags** column can have flags set for both current log file and temporary log file. Thus, it might be easier to read values in the **flags** column if the values are returned using the HEX function.

Column	Type	Description		
number	smallint	Logical-log file number		
uniqid	integer	Log-file ID		
size	integer	Number of pages in the log file		
used	integer	Number of pages used in the log file		
is_used	integer	1 If file is used, 0 if not		
is_current	integer	1 If file is the current file, 0 if not		
is_backed_up	integer	1 If file has been backed up, 0 if not		
is_new	integer	1 If the log has been added since the last level-0 dbspace backup, 0 if not		
is_archived	integer	1 If file has been placed on the backup tape, 0 if not		
is_temp	integer	1 If the file is flagged as a temporary log file, 0 if not		
flags	smallint	Flags	Hexadecimal	Meaning
		1	0x01	Log file is in use
		2	0x02	File is current log file
		4	0x04	Log file has been backed up
		8	0x08	File is newly added log file
		16	0x10	Log file has been written to dbspace backup media
		32	0x20	Log is a temporary log file

sysmgminfo

The **sysmgminfo** table provides an overview of the Memory Grant Manager (MGM) and Parallel Data Query (PDQ) information.

Column	Type	Description
max_query	integer	Maximum number of active queries allowed
total_mem	integer	Total MGM memory
avail_mem	integer	Free MGM memory
total_seq	integer	Total number of sequential scans
avail_seq	integer	Unused sequential scans

Column	Type	Description
active	integer	Number of active MGM queries
ready	integer	Number of ready MGM queries
min_free_mem	integer	Minimum free MGM memory
avg_free_mem	float	Average free MGM memory
std_free_mem	float	Standard free MGM memory
min_free_seq	integer	Minimum free MGM sequential scans
avg_free_seq	float	Average free MGM sequential scans
std_free seq	float	Standard free MGM sequential scans
max_active	integer	Maximum active MGM SQL operations
cnt_active	integer	Number of active MGM SQL operations
avg_active	float	Average active MGM SQL operations
std_active	float	Standard active MGM SQL operations
max_ready	integer	Maximum ready MGM SQL operations
cnt_ready	integer	Number of ready MGM SQL operations
avg_ready	float	Average ready MGM SQL operations
std_ready	float	Standard ready MGM SQL operations

sysnetclienttype

The **sysnetclienttype** table provides an overview of the network activity for each client type.

Column	Type	Description
nc_cons_allowed	integer	Whether or not connections are allowed
nc_accepted	integer	Number of connections that were accepted
nc_rejected	integer	Number of network connections that were rejected
nc_reads	int8	Number of network reads for this client type
nc_writes	int8	Number of network writes for this client type
nc_name	char(18)	Name of the client type

sysnetglobal

The **sysnetglobal** table provides an overview of the system network.

Column	Type	Description
ng_reads	int8	Number of network reads
ng_writes	int8	Number of network writes
ng_connects	int8	Number of network connections
ng_his_read_count	int8	Number of network reads by users who have disconnected ng_his_read_bytes
ng_his_read_bytes	int8	Data transferred to the server by users who have disconnected

Column	Type	Description
ng_his_write_count	int8	Number of network writes by users who have disconnected
ng_his_write_bytes	int8	Data transferred to the client by users who have disconnected
ng_num_netscbs	integer	Number of network subscribers
ng_max_netscbs	integer	Maximum number of network subscribers
ng_free_thres	integer	Threshold for the maximum number of freed buffers in the buffer list
ng_free_cnt	integer	Number of times the ng_free_thres limit has been reached
ng_wait_thres	integer	Threshold for the maximum number of buffers that can be held in the buffer list for one connection
ng_wait_cnt	integer	Number of times the ng_wait_thres limit has been reached
ng_pvt_thres	integer	Threshold for the maximum number of freed buffers in the private buffer queue
ng_netbuf_size	integer	Size of the transport network buffers
ng_buf_alloc	integer	Number of network buffers allocated
ng_buf_alloc_max	integer	Maximum value of allocated network buffers
ng_netscb_id	integer	Next netscb id

sysnetworkio

The `sysnetglobal` table provides an overview of the system network.

Column	Type	Description
net_id	integer	Netscb id
net_sid	integer	Session id
net_netscb	int8	Netscb prt
net_client_type	integer	Client type Int
net_client_name	char(12)	Client protocol name
net_read_cnt	int8	Number of network reads
net_write_cnt	int8	Number of network writes
net_open_time	integer	Time this session connected
net_last_read	integer	Time of the last read from the network
net_last_write	integer	Time of the last write from the network
net_stage	integer	Connect / Disconnect / Receive
net_options	integer	Options from sqlhosts
net_protocol	integer	Protocol
net_type	char(10)	Type of network protocol
net_server_fd	integer	Server fd
net_poll_thread	integer	Poll thread

sysonlinelog

The **sysonlinelog** table provides a view of the information stored in the online.log file.

Column	Type	Description
offset	int8	File offset
next_offset	int8	Offset to the next message
line	char(4096)	Single line of text from the file

sysprofile

The **sysprofile** table contains profile information about the database server.

Column	Type	Description
name	char(13)	Name of profiled event. (See table that follows for a list of possible events.)
value	integer	Value of profiled event. (See table that follows for a list of possible events.)

The following table lists the events that, together with a corresponding value, make up the rows of the **sysprofile** table.

Events Profiled in sysprofile	Description
dskreads	Number of actual reads from disk
bufreads	Number of reads from shared memory
dskwrites	Actual number of writes to disk
bufwrites	Number of writes to shared memory
isamtot	Total number of calls
isopens	isopen calls
isstarts	isstart calls
isreads	isread calls
iswrites	iswrite calls
isrewrites	isrewrite calls
isdeletes	isdelete calls
iscommits	iscommit calls
isrollbacks	isrollback calls
ovlock	Overflow lock table
ovuser	Overflow user table
ovtrans	Overflow transaction table
latchwts	Latch request waits
bufwts	Buffer waits
lockreqs	Lock requests
lockwts	Lock waits

Events Profiled in sysprofile	Description
ckptwts	Checkpoint waits
deadlks	Deadlocks
lktouts	Deadlock time-outs
numckpts	Number checkpoints
plgpagewrites	Physical-log pages written
plgwrites	Physical-log writes
llgrecs	Logical-log records
llgpagewrites	Logical-log writes
llgwrites	Logical-log pages written
pagreads	Page reads
pagwrites	Page writes
flushes	Buffer-pool flushes
compress	Page compresses
fgwrites	Foreground writes
lruwrites	Least-recently used (LRU) writes
chunkwrites	Writes during a checkpoint
btradata	Read-ahead data pages read through index leaf node
btraidx	Read-ahead data pages read through index branch or root node
dpra	Data pages read into memory with read-ahead feature
rapgs_used	Read-ahead data pages that user used
seqscans	Sequential scans
totalsorts	Total sorts
memsorts	Sorts that fit in memory
disksorts	Sorts that did not fit in memory
maxsortspace	Maximum disk space used by a sort

sysproxyagents

The **sysproxyagents** table contains information about all proxy agent threads. Proxy agent threads run on the primary server and accept requests from secondary servers to process DML operations. The primary server also contains a proxy distributor that handles secondary server updates. Secondary servers determine how many instances of the proxy distributor to create based on the UPDATABLE_SECONDARY setting in the secondary server's ONCONFIG file.

Column	Type	Description
tid	integer	Transaction ID of the proxy agent thread running on the primary server. This ID is created by the proxy distributor to handle work from the secondary server session.
flags	integer	Flags of the proxy agent thread.
proxy_id	integer	ID of the proxy distributor on behalf of the currently executing proxy agent thread (TID).

Column	Type	Description
source_session_id	integer	ID of the user's session on the secondary server.
proxy_txn_id	integer	Number of the current transaction. These numbers are unique to the proxy distributor.
current_seq	integer	The sequence number of the current operation in the current transaction.
sqlerrno	integer	Error number of any SQL error (or 0 on success)
iserrno	integer	Error number of any ISAM/RSAM error (or 0 on success)

sysproxydistributors

The **sysproxydistributors** table contains information about the proxy distributors.

On the primary server, this table contains information about all of the proxy distributors in a high-availability cluster. On a secondary server, this table contains information about only those proxy distributors that are assigned to process updates to the secondary server.

Column	Type	Description
node_name	char	Name of the secondary server as it is known by the primary server (for example, INFORMIXSERVER, HA_ALIAS, and so on).
proxy_id	integer	ID of the proxy distributor. These IDs are unique within a high-availability cluster.
transaction_count	integer	Number of transactions currently being processed by the proxy distributor.
hot_row_total	integer	Total number of hot rows ever handled by the proxy distributor.

sysproxysessions

The **sysproxysessions** table contains information about each of the sessions that are using redirected-write functionality. This table is only valid on the secondary server.

Column	Type	Description
session_id	integer	ID of a user's session on the secondary server.
proxy_id	integer	ID of the proxy distributor on behalf of which the proxy agent thread (TID) is running
proxy_tid	integer	Transaction ID of the proxy agent thread running on the primary server. This ID is created by the proxy distributor to handle work from the secondary server session.
proxy_txn_id	integer	Number of the current transaction. These numbers are unique to the proxy distributor.
current_seq	integer	The sequence number of the current operation in the current transaction.

Column	Type	Description
pending_ops	integer	The number of operations buffered on the secondary server that have not yet been sent to the primary server.
reference_count	integer	Indicates the number of threads (for example, sqlexec, sync reply, recovery, and so on) that are using the information for this transaction. When reference_count equals 0, the transaction processing has completed (either successfully or unsuccessfully).

sysproxytnops

The **sysproxytnops** table contains information about each of the transactions executing via each proxy distributor.

On the primary server, this table contains information about all of the proxy distributors in the high-availability cluster. On a secondary server, this table only contains information about the proxy distributors used to process updates to the secondary server.

Column	Type	Description
proxy_id	integer	ID of the proxy distributor. These IDs are unique within a high-availability cluster.
proxy_txn_id	integer	Number of the transaction. These numbers are unique to the proxy distributor.
sequence_number	integer	The number of the operation.
operation_type	integer	The type of operation to be performed; Insert, Update, Delete, or Other.
rowidn	integer	The ID of the row on which to apply the operation.
table	char	The full table name, trimmed to fit a reasonable length. Format: <i>database:owner.tablename</i>
sqlerrno	integer	Error number of any SQL error (or 0 on success)

sysproxytxns

The **sysproxytxns** table contains information about all of the current transactions executing via each proxy distributor.

On the primary server, this table contains information about each of the proxy distributors in the high-availability cluster. On a secondary server, this table only contains information about the proxy distributors used to process updates to the secondary server.

Column	Type	Description
proxy_id	integer	ID of the proxy distributor. These IDs are unique within a high-availability cluster.
proxy_txn_id	integer	Number of the transaction. These numbers are unique to the proxy distributor.

Column	Type	Description
reference_count	integer	Indicates the number of threads (for example, sqlxexec, sync reply, recovery, and so on) that are using the information for this transaction. When the count becomes 0 this indicates the transaction processing is complete. (either successfully or unsuccessfully).
pending_ops	integer	On the primary server: the number of operations received from the secondary server that have not yet been processed. On the secondary server, the number of operations buffered on the secondary server that have not yet been sent to the primary server.
proxy_sid	integer	Proxy Session ID

sysptprof

The **sysptprof** table lists information about a tblspace. Tblspaces correspond to tables.

Profile information for a table is available only when a table is open. When the last user who has a table open closes it, the tblspace in shared memory is freed, and any profile statistics are lost.

Column	Type	Description
dbname	char(128)	Database name
tablename	char(128)	Table name
partnum	integer	Partition (tblspace) number
lockreqs	integer	Number of lock requests
lockwts	integer	Number of lock waits
deadlks	integer	Number of deadlocks
lktouts	integer	Number of lock timeouts
isreads	integer	Number of isreads
iswrites	integer	Number of iswrites
isrewrites	integer	Number of isrewrites
isdeletes	integer	Number of isdeletes
bufreads	integer	Number of buffer reads
bufwrites	integer	Number of buffer writes
seqscans	integer	Number of sequential scans
pagreads	integer	Number of page reads
pagwrites	integer	Number of page writes

sysrepevtreg

Connection Manager, the OpenAdmin tool, or any client application can register for a pre-defined set of events using the **sysrepevtreg** pseudo table. After registering events through the **sysrepevtreg** pseudo table, Connection Manager, the OpenAdmin tool, or any client application can receive event data by querying the table.

Column	Type	Description
evt_bitmap	integer	Event class ID bitmap
evt_timeout	integer	Maximum time in seconds that the client can wait for event data. Valid timeout values are: <ul style="list-style-type: none"> • 0; no wait (default) • -1; wait forever • <i>n</i> (where <i>n</i> > 0) wait <i>n</i> seconds
evt_hwm	integer	Pending event list high-water mark
evt_info	char(256)	Event information (Not yet implemented)

sysrepstats

The **sysrepstats** table can be used to post events to Connection Manager and to the OpenAdmin tool. Connection Manager, the OpenAdmin tool, and client applications can communicate with each other by posting events to the **sysrepstats** pseudo table.

Column	Type	Description
repstats_type	integer	Event class ID
repstats_subtype	integer	Sub event ID
repstats_time	integer	Time at which event was initiated
repstats_ver	integer	Version number of event data
repstats_desc	lvarchar	Event data

User interface for sysrepstats and sysrepevtreg tables

Client applications can post events to Connection Manager or to other clients by inserting event information into the **sysrepstats** pseudo table. Posting events to the table provides the ability for programs such as the OpenAdmin tool to communicate with Connection Manager. By posting events to the **sysrepstats**, you can issue control messages to Connection Manager without having to directly connect to Connection Manager itself.

When Connection Manager registers that it wishes to receive events, it passes a bitmap of the event types that it wants to receive. As events are received, they are posted to the thread that placed the request. Client applications can register events using the sysmaster pseudo table **sysrepevtreg**, and receive event data by issuing select or fetch statements against the **sysrepstats** pseudo table.

The following table lists each event class, its bit value, and a description of the event class.

Event class name	Bit value	Description
REPEVT_CLUST_CHG	0x1	Event class for High-Availability cluster changes
REPEVT_CLUST_PERFSTAT	0x2	Event class for workload statistics for the server nodes in a High-Availability cluster
REPEVT_CLUST_LATSTAT	0x4	Event class for replication latency information for server nodes in a High-Availability cluster

Event class name	Bit value	Description
REPEVT_CM_ADM	0x8	Connection Manager administration commands
REPEVT_SRV_ADM	0x10	Event class for server mode changes
REPEVT_ER_ADM	0x20	Event class for events related to Enterprise Replication (ER)
REPEVT_CLIENT	0x40	User-defined client event

The following table lists sub-events for the event class REPEVT_CLUST_CHG:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_CLUST_ADD	1	Adding new node to a High-Availability cluster	Only at primary server in a High-Availability cluster
REPEVT_SUB_CLUST_DROP	2	Dropping a node from a High-Availability cluster	Only at primary server in a High-Availability cluster
REPEVT_SUB_CLUST_CON	3	High-Availability secondary node connected to primary server	Only at primary server in a High-Availability cluster
REPEVT_SUB_CLUST_DIS	4	High-Availability secondary node disconnected from primary server	Only at primary server in a High-Availability cluster
REPEVT_SUB_CLUST_NEWPRIM	5	High-Availability primary node changed	Only at secondary servers in a High-Availability cluster
REPEVT_SUB_CLUST_DROFF	6	HDR secondary node disconnected from primary server	HDR primary and secondary servers
REPEVT_SUB_CLUST_DRON	7	HDR secondary node connected to primary server	HDR primary and secondary servers

The following table lists sub-events for the event class REPEVT_CLUST_PERFSTAT:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_LOCAL_PERFSTAT	1	Work load statistics for local server	All servers in a High-Availability cluster
REPEVT_SUB_REMOTE_PERFSTAT	2	Work load statistics for High-Availability secondary servers	Only at the primary server in a High-Availability cluster

The following table lists sub-events for the event class REPEVT_CLUST_LATSTAT:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_LOCAL_LATSTAT	1	Replication latency statistics for secondary servers in a High-Availability cluster	Only at the primary server in a High-Availability cluster

The following table lists sub-events for the event class REPEVT_CM_ADM:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_CM_ADM_REQ	1	Command request	All IDS server instances
REPEVT_SUB_CM_ADM_ACK	2	Command response	All IDS server instances
REPEVT_SUB_CM_REG	3	Connection Manager registered with server	All IDS server instances
REPEVT_SUB_CM_DEREG	4	Connection Manager de-registered with server	All IDS server instances
REPEVT_SUB_CM_FATAL	5	Connection Manager terminated without de-registering with server	All IDS server instances

The following table lists sub-events for the event class REPEVT_SRV_ADM:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_SRV_BLK	1	Server blocked due to DDRBLOCK	All IDS server instances
REPEVT_SUB_SRV_UBLK	2	Server unblocked; DDRBLOCK removed	All IDS server instances

The following table lists sub-events for the event class REPEVT_ER_ADM:

Sub-event name	Value	Description	Event available at:
REPEVT_SUB_ER_SPOOL_FULL	1	ER blocked while waiting for space to be added in either the queue data sbspace or dbspace, or in the grouper paging sbspace.	Enterprise Replication server nodes

sysrsslog

The **sysrsslog** table captures information about RS secondary servers at the primary server.

Column	Type	Description
server_name	char(128)	Server name

Column	Type	Description
from_cache	integer	Total pages read from log buffer cache
from_disk	integer	Total pages read from disk
logpages_tossed	integer	Total number of log pages not written to log buffer cache

sysscblst

These columns of the **sysscblst** table provide information about session memory amounts.

Column	Type	Description
memtotal	integer	Total memory available
memused	integer	Total memory used

sys sesappinfo

The **sys sesappinfo** table in the **sysmaster** displays information on Distributed Relational Database Architecture (DRDA) client sessions. The **sys sesappinfo** table has the following columns.

Column	Type	Explanation
sesapp_sid	INTEGER	Client session ID
sesapp_name	CHAR(128)	Session application name
sesapp_value	CHAR(512)	Session value

sys sesprof

The **sys sesprof** table lists cumulative counts of the number of occurrences of user actions such as writes, deletes, or commits.

Column	Type	Description
sid	integer	Session ID
lockreqs	integer	Number of locks requested
locksheld	integer	Number of locks currently held
lockwts	integer	Number of times waited for a lock
deadlks	integer	Number of deadlocks detected
lktouts	smallint	Number of deadlock timeouts
logrecs	integer	Number of logical-log records written
isreads	integer	Number of reads
iswrites	integer	Number of writes
isrewrites	integer	Number of rewrites
isdeletes	integer	Number of deletes
iscommits	integer	Number of commits
isrollbacks	integer	Number of rollbacks

Column	Type	Description
longtxs	integer	Number of long transactions
bufreads	integer	Number of buffer reads
bufwrites	integer	Number of buffer writes
seqscans	integer	Number of sequential scans
pagreads	integer	Number of page reads
pagwrites	integer	Number of page writes
total_sorts	integer	Number of total sorts
dsksorts	integer	Number of sorts that did not fit in memory
max_sortdiskspace	integer	Maximum space used by a sort
logspused	integer	Number of bytes of logical-log space used by current transaction of session
maxlogsp	integer	Maximum number of bytes of logical-log space ever used by the session

sysessions

The **sysessions** table provides general information on each user connected to the database server. In the **state** column, each bit position represents a separate flag. Thus, it might be easier to read values in the **state** column if the values are returned using the HEX function.

Column	Type	Description
sid	integer	Session ID
username	char(32)	User ID
uid	smallint	User ID number
pid	integer	Process ID of the client
hostname	char(16)	Hostname of client
tty	char(16)	Name of the user's stderr file
connected	integer	Time that user connected to the database server
feprogram	char(16)	Reserved for future use
pooladdr	integer	Session pool address
is_wlatch	integer	1 If the primary thread for the session is waiting for a latch
is_wlock	integer	1 If the primary thread for the session is waiting for a lock
is_wbuff	integer	1 If the primary thread for the session is waiting for a buffer
is_wckpt	integer	1 If the primary thread for the session is waiting for a checkpoint
is_wlogbuf	integer	1 If the primary thread for the session is waiting for a log buffer
is_wtrans	integer	1 If the primary thread for the session is waiting for a transaction
is_monitor	integer	1 If the session is a special monitoring process
is_incrit	integer	1 If the primary thread for the session is in a critical section

Column	Type	Description		
state	integer	Flags	Hexadecimal	Meaning
		1	0x00000001	User structure in use
		2	0x00000002	Waiting for a latch
		4	0x00000004	Waiting for a lock
		8	0x00000008	Waiting for a buffer
		16	0x00000010	Waiting for a checkpoint
		32	0x00000020	In a read call
		64	0x00000040	Writing logical-log file to backup tape
		128	0x00000080	ON-Monitor (UNIX)
		256	0x00000100	In a critical section
		512	0x00000200	Special daemon
		1024	0x00000400	Archiving
		2048	0x00000800	Clean up dead processes
		4096	0x00001000	Waiting for write of log buffer
		8192	0x00002000	Special buffer-flushing thread
		16384	0x00004000	Remote database server
		32768	0x00008000	Deadlock timeout used to set RS_timeout
		65536	0x00010000	Regular lock timeout
		262144	0x00040000	Waiting for a transaction
524288	0x00080000	Primary thread for a session		
1048576	0x00100000	Thread for building indexes		
2097152	0x00200000	B-tree cleaner thread		

sysasmx

The `sysasmx` table provides SMX (server multiplexer group) connection information.

Column	Type	Description
address	int8	SMX pipe address
name	char(128)	Target server name
encryption_status	char(20)	Enabled or disabled
buffers_sent	integer	Number of buffers sent
buffers_rcv	integer	Number of buffers received
bytes_sent	int8	Number of bytes sent
bytes_rcv	int8	Number of bytes received
reads	integer	Number of read calls
writes	integer	Number of write calls
retries	integer	Number of write call retries

sysmxses

The `sysmxses` table provides SMX (server multiplexer group) session information.

Column	Type	Description
<code>name</code>	char(128)	Target server name
<code>address</code>	int8	SMX session address
<code>client_type</code>	char(20)	SMX client type
<code>reads</code>	integer	Number of read calls
<code>writes</code>	integer	Number of write calls

syssqltrace

The `syssqltrace` table provides detailed information about a single SQL statement.

Column	Type	Description
<code>sql_id</code>	int8	Unique SQL execution ID
<code>sql_address</code>	int8	Address of the statement in the code block
<code>sql_sid</code>	int	Database session ID of the user running the SQL statement
<code>sql_uid</code>	int	User ID of the statement running the SQL
<code>sql_stmnttype</code>	int	Statement type
<code>sql_stmntname</code>	char(40)	Statement type displayed as a word
<code>sql_finishtime</code>	int	Time this statement completed (UNIX)
<code>sql_begintxtime</code>	int	Time this transaction started
<code>sql_runtime</code>	float	Statement execution time
<code>sql_pgreads</code>	int	Number of disk reads for this SQL statement
<code>sql_bfreads</code>	int	Number of buffer reads for this SQL statement
<code>sql_rdcache</code>	float	Percentage of time the page was read from the buffer pool
<code>sql_bfidxreads</code>	int	Number of index page buffer reads
<code>sql_pgwrites</code>	int	Number of pages written to disk
<code>sql_bfwrites</code>	int	Number of pages modified and returned to the buffer pool
<code>sql_wrcache</code>	float	Percentage of time a page was written to the buffer pool but not to disk
<code>sql_lockreq</code>	int	Total number of locks required by this SQL statement
<code>sql_lockwaits</code>	int	Number of times the SQL statement waited on locks
<code>sql_lockwtime</code>	float	Time the system waited for locks during SQL statement
<code>sql_logspace</code>	int	Amount of space the SQL statement used in the logical log
<code>sql_sorttotal</code>	int	Number of sorts that ran for the statement
<code>sql_sortdisk</code>	int	Number of sorts that ran on disk
<code>sql_sortmem</code>	int	Number of sorts that ran in memory
<code>sql_executions</code>	int	Number of times the SQL statement ran

Column	Type	Description
sql_totalltime	float	Total amount of time spent running the statement
sql_avgtime	float	Average amount of time spent running the statement
sql_maxtime	float	Maximum amount of time spent executing the SQL statement
sql_numioawaits	int	Number of times an I/O operation had to wait
sql_avgioawaits	float	Average amount of time that the SQL statement had to wait
sql_totaliowaits	float	Total amount of time that the SQL statement had to wait for I/O. This excludes any asynchronous I/O.
sql_rowspersec	float	Average number of rows (per second) produced
sql_estcost	int	Cost associated with the SQL statement
sql_estrows	int	Estimated number of rows returned for the SQL statement as predicted by the optimizer
sql_actualrows	int	Number of rows returned for the SQL statement
sql_sqlerror	int	SQL error number
sql_isamerror	int	RSAM/ISAM error number
sql_isollevel	int	Isolation level of the SQL statement.
sql_sqlmemory	int	Number of bytes needed to execute the SQL statement
sql_numiterators	int	Number of iterators used by the statement
sql_database	char(128)	Database name
sql_numtables	int	Number of tables used in executing the SQL statement
sql_tablelist	char(4096)	List of table names directly referenced in the SQL statement. If the SQL statement fires triggers that execute statements against other tables, the other tables are not listed.
sql_statement	char(1600)	SQL statement that ran

sysssltrace_info

The `sysssltrace_info` table describes information about the SQL profile trace system.

Column	Type	Description
flags	integer	SQL trace flags
ntraces	integer	Number of items to trace
tracesize	integer	Size of the text to store for each SQL trace item
duration	integer	Trace buffer (in seconds)
sqlseen	int8	Number of SQL items traced since start or resizing
starttime	integer	Time tracing was enabled
memoryused	int8	Number of bytes of memory used by SQL tracing

sysssltrace_iter

The `sysssltrace_iter` table lists the SQL statement iterators.

Column	Type	Description
sql_id	int8	SQL execution ID
sql_address	int8	Address of the SQL statement block
sql_itr_address	int8	Address of the iterator
sql_itr_id	int	Iterator ID
sql_itr_left	int	Iterator ID to the left
sql_itr_right	int	Iterator ID to the right
sql_itr_cost	int	Iterator cost
sql_itr_estrows	int	Iterator estimated rows
sql_itr_numrows	int	Iterator actual rows processed
sql_itr_type	int	Iterator type
sql_itr_misc	int	Iterator miscellaneous flags
sql_it_info	char(256)	Iterator miscellaneous flags displayed as text

syssrcrs

The `syssrcrs` table provides RS secondary server related statistics at the primary server.

Column	Type	Description
address	int8	RS secondary server control block address
server_name	char(128)	Database server name
server_status	char(20)	Quiescent, active, or inactive
connection_status	char(20)	Connected or disconnected
log_transmission_status	char(20)	Active or blocked
next_page_tosend_log_uniq	integer	Unique log ID of next page to send
next_page_tosend_log_page	integer	Page number of next page to send
seq_tosend	integer	Sequence ID of last buffer sent
last_seq_acked	integer	Sequence ID of last buffer acknowledged

syssrcsds

The `syssrcsds` table provides SD secondary server related statistics at the primary server.

Column	Type	Description
address	int8	SD secondary server control block address
source_server	char(128)	Primary database server name
connection_status	char(20)	Connected or disconnected
last_received_log_uniq	integer	Unique log ID of last log page received
last_received_log_page	integer	Page number of last log page received
next_lpgtoread_log_uniq	integer	Unique log ID of next log page to read
next_lpgtoread_log_page	integer	Page number of next log page to read
last_acked_lsn_uniq	integer	Unique log ID of last LSN acknowledged

Column	Type	Description
last_acked_lsn_pos	integer	Log position of last LSN acknowledged
last_seq_received	integer	Sequence ID of last buffer received
last_seq_acked	integer	Sequence ID of last buffer acknowledged
cur_pagingfile	char(640)	Current paging file name
cur_pagingfile_size	int8	Current paging file size
old_pagingfile	char(640)	Old paging file name
old_pagingfile_size	int8	Old paging file size

systabnames

The **systabnames** table describes each table that the database server manages.

Column	Type	Description
partnum	integer	tblspace identifier
dbname	char(128)	Database name
owner	char(32)	User ID of owner
tablename	char(128)	Table name
collate	char(32)	Collation associated with a database that supports GLS

systhreads

The **systhreads** table describes information about waiting threads.

Column	Type	Description
wreason	integer	Reason the thread is waiting
wait_reason	char(18)	Reason the thread is waiting

systmgrss

The **systmgrss** table provides RS secondary server related statistics at the RS secondary server.

Column	Type	Description
address	int8	RS secondary server control block address
source_server	char(128)	Source server serving the RS secondary server
connection_status	char(20)	Connected or disconnected
last_received_log_uniq	integer	Unique log ID of last log page received
last_received_log_page	integer	Page number of last log page received
last_seq_received	integer	Sequence ID of last buffer received
last_seq_acked	integer	Sequence ID of last buffer acknowledged

sysstrgsds

The **sysstrgsds** table provides SD secondary server related statistics at the SD secondary server.

Column	Type	Description
address	int8	SD secondary server control block address
source_server	char(128)	Source server serving the SD secondary server
connection_status	char(20)	Connected or disconnected
last_received_log_uniq	integer	Unique log ID of last log page received
last_received_log_page	integer	Page number of last log page received
next_lptoread_log_uniq	integer	Unique log ID of next log page to read
next_lptoread_log_page	integer	Page number of next log page to read
last_acked_lsn_uniq	integer	Unique log ID of last LSN acknowledged
last_acked_lsn_pos	integer	Log position of last LSN acknowledged
last_seq_received	integer	Sequence ID of last buffer received
last_seq_acked	integer	Sequence ID of last buffer acknowledged
cur_pagingfile	char(640)	Current [®] paging file name
cur_pagingfile_size	int8	Current paging file size
old_pagingfile	char(640)	Old paging file name
old_pagingfile_size	int8	Old paging file size

sysvpprof

The **sysvpprof** table lists user and system CPU time for each virtual processor.

Column	Type	Description
vpid	integer	Virtual processor ID
class	char(50)	Type of virtual processor: <ul style="list-style-type: none">• cpu• adm• lio• pio• aio• tli• soc• str• shm• opt• msc• adt
usercpu	float	Number of microseconds of user time
syscpu	float	Number of microseconds of system time

The SMI Tables Map

Figure 2-1 on page 2-38 displays the columns in some of the SMI tables.

sysadinfo
adtmode
adterr
adtsize
adtpath
adtfle

sysaudit
username
succ1
succ2
succ3
succ4
succ5
fail1
fail2
fail3
fail4
fail5

syschkio
chunknum
reads
pagesread
writes
pageswritten
mreads
mpagesread
mwrites
mpageswritten

syschunks
chknun
dbsnum
nxchknun
chksize
offset
nfree
ls_offline
is_recovering
is_blobchunk
is_sbchunk
is_inconsistent
flags
fname
mfname
moffset
mis_offline
mis_recovering
mflags

sysconfig
cf_id
cf_name
cf_flags
cf_originals
cf_effective
cf_default

sysdatabases
name
partnum
owner
created
is_logging
is_buff_log
is_ansi
is_nls
flags

sysdbslocale
dbs_dbsname
dbs_collate

sysdbspaces
dbsnum
name
owner
fchunk
nchunks
is_mirrored
is_blobspace
is_sbpace
is_temp
flags

sysdri
type
state
name
intvl
timeout
lostfound

sysextents
dbsname
tabname
chunk
offset
size

sysextspaces
id
name
owner
flags
refcnt
locsize
location

syslocks
dbsname
tabname
rowidlk
keynum
type
owner
waiter

syslogs
number
uniqid
size
used
is_used
is_current
is_backed_up
is_new
is_archived
is_temp
flags

sysprofile
name
value

sysptprof
dbsname
tabname
partnum
lockreqs
lockwts
deadlks
lktouts
isreads
iswrites
isrewrites
isdeletes
bufreads
bufwrites
seqscans
pagereads
pagewrites

sysstesprof
sid
lockreqs
locksheld
lockwts
deadlks
lktouts
logrecs
isreads
iswrites
isrewrites
isdeletes
iscommits
isrollbacks
longtxs
bufreads
bufwrites
seqscans
pagereads
pagewrites

sysessions
sid
username
uid
pid
hostname
tty
connected
feprogram
pooladdr
is_wlatch
is_wlock
is_wbuff
is_wckpt
is_wlogbuf
is_wtrans
is_monitor
is_incrit
stata

Information from onstat in the SMI Tables

To obtain information provided by the **onstat** utility, you can use SQL to query appropriate SMI tables. The following table indicates which SMI tables to query to obtain the information provided by a given **onstat** option. For descriptions of the **onstat** options, see “Monitor the Database Server Status” on page 15-2.

onstat Option	SMI Tables to Query	onstat Fields <i>Not</i> in SMI Tables
-d	sysdbspaces syschunks	address bpages
-D	sysdbspaces syschkio	
-F	sysprofile	address flusher snoozer state data
-g ath	systhreads	
-g dri	sysdri	Last DR CKPT (id/pg)
-g glo	sysvpprof	Listing of virtual processors by class
-g ipl	sysipl	
-g rss	sysrsslog systgrss syssrcrs	
-g his	syssqltracing	
-g sds	syssrcsds systrgsds	
-g smx	sysmx	
-g smx ses	sysmxses	
-k	syslocks	address lklist tblsnum
-l	syslogs sysprofile	All physical-log fields (except numpages and numwrits) All logical-log buffer fields (except numrecs, numpages, and numwrits) address begin % used
-p	sysprofile	
-u	sysessions sysseprof	address wait nreads nwrites

Chapter 3. The sysadmin Database

In This Chapter

This chapter describes the **sysadmin** database and contains reference information about the tables in the database.

The sysadmin Database

The Scheduler is an administrative tool that enables the database server to execute database functions and procedures at predefined times or as determined internally by the server. The Scheduler is defined and driven by tasks, and the **sysadmin** database contains the six tables which contain and organize Scheduler task information. By default, only user **informix** is granted access to the **sysadmin** database; other users may be granted access to **sysadmin**. For detailed information about the Scheduler, see the *IBM Informix Administrator's Guide*.

Because several important database server components use it, you should not drop or alter the **sysadmin** database. You can, however, move the **sysadmin** database from its default root dbspace location if the root dbspace does not have enough space for storing task properties and command history information. For instructions on moving the **sysadmin** database, see the procedure for using the RESET SYSADMIN SQL Administration API command in the *IBM Informix Administrator's Guide*.

The PH_TASK Table

The **PH_TASK** table contains information about how and when each task will be executed.

Column	Type	Description
tk_id	serial	Sequential task ID
tk_name	char(36)	Task name. A unique index on this column ensures that no two names are the same.
tk_description	lvarchar	Description about this task
tk_type	char(18)	Type of task: <ul style="list-style-type: none">• TASK: Executes a task which does not collect data• SENSOR: A task which collects data• STARTUP SENSOR: Runs only when the server starts• STARTUP MONITOR: Runs only when the server starts
tk_sequence	integer	Current data collection number System updated; do not modify
tk_owner	integer	Owner's thread ID System updated; do not modify

Column	Type	Description
tk_result_table	varchar	Result table name Note: The tk_result_table column is used only by sensors and the content matches the table created in tk_create. When the tk_delete interval is exceeded, data is deleted from tk_result_table.
tk_create	lvarchar	The CREATE TABLE statement to execute Note: The tk_create column is used only by sensors. and as necessary, is created to contain any data a sensor might store.
tk_execute	lvarchar	The SQL object to execute
tk_delete	interval day(2) to second	Deletes data older than this interval
tk_start_time	datetime hour to second	Starting time of this task
tk_stop_time	datetime hour to second	Time of day this task should stop running.
tk_frequency	interval day(2) to second	How often this task runs
tk_next_execution	datetime year to second	Next time this task should be executed
tk_attributes	integer	FlagsSystem updated; do not modify
tk_group	varchar(128)	Group Name references ph_group(group_name)
tk_exec_num	integer	Number of times to execute this taskSystem updated; do not modify
tk_exec_time	integer	Total time spent executing this taskSystem updated; do not modify
tk_enable	boolean	Whether or not the task is enabledIf the value of tk_enabled equals FALSE, the task is not scheduled for execution
tk_priority	integer	Job priority, on a scale of 0- 5. If there are several jobs to execute simultaneously, the job with the highest priority executes first. The default is 0.

The PH_RUN Table

The **PH_RUN** table contains information about how and when each Scheduler task ran.

Column	Type	Description
run_id	serial	Sequential ID generated during execution
run_task_id	integer	ID of the Scheduler task executed out of the PH_TASK table
run_task_seq	integer	Data Collector sequence number
run_retcode	integer	Return code or SQLcode from the UDR or SQL statement

Column	Type	Description
run_time	datetime year to second	Time this Scheduler task was executed
run_duration	float	Time it took to execute this job (in seconds)
run_ztime	integer	Time onstat -z was last run
run_btime	integer	Time when server started
run_mvertime	integer	Time the task was executed

The PH_GROUP Table

The **PH_GROUP** table contains information about the Scheduler group names.

Column	Type	Description
group_id	serial	Group ID
group_name	varchar(128)	Unique name of the group
group_description	lvarchar	Description of the group

The PH_ALERT Table

The **PH_ALERT** table contains information for the Scheduler about error, warning, or informational messages.

Column	Type	Description
ID	serial	Alert ID
alert_run_id	integer	Invocation of a Scheduler task that created the alert
alert_task_seq	integer	Identifies which invocation of a Scheduler task created the alert
alert_type	char(8)	Informational, warning, or error
alert_color	char(15)	Green, yellow, or red. For more information about the alerts, see Table x below.
alert_time	datetime year to second	Time the alert was generated
alert_state	char(15)	Indicates which state the object is in currently: NEW The alert was newly added and no other action has occurred on this alert. IGNORED The alert was acknowledged by the DBA and no action was taken. ACKNOWLEDGED The alert has been acknowledged by the DBA. ADDRESSED The alert has been addressed by the DBA.
alert_state_changed	datetime year to second	The last time the state was changed

Column	Type	Description
alert_object_type	char(15)	The type of object: <ul style="list-style-type: none"> • SERVER • DATABASE • TABLE • INDEX • DBSPACE • CHUNK • USER • SQL_STATEMENT • MISC
alert_object_name	varchar(255)	The name of the object described above
alert_message	lvarchar	Message
alert_action	lvarchar	Corrective Action. This is an SQL script which can be executed by the user or tool or it will be NULL if no action is available. This script must comply with all multi-statement prepare rules.
alert_action_dbs	lvarchar(256)	Name of the database to use when executing the alert_action

This table defines the alert colors for the three different types of messages.

	Green	Yellow	Red
Informative	A status message indicating a component's operation status	An important status message	A status message that requires action.
Warning	A warning from the database that was automatically addressed	A future event that needs to be addressed	A predicted failure is imminent. Action is necessary now.
Error	A failure in a component corrected itself	A failure in a component corrected itself but might need DBA action	A failure in a component requires DBA action.

The PH_THRESHOLD Table

The **PH_THRESHOLD** table contains information about thresholds for Scheduler tasks.

Column	Type	Description
ID	integer	Alert ID
task_name	varchar	Scheduler task name associated with the threshold
Name	char	Name of the threshold

Column	Type	Description
Value	lvarchar	Value of the threshold
Value_Type	char	The data type of the value column: <ul style="list-style-type: none"> • STRING • NUMERIC • NUMERIC, MAX, MIN
Description	lvarchar	Description of the threshold

The Results Table

The Results table contains historical data about Scheduler task execution.

Column	Type	Description
ID	integer	Required Column and Name This column links to the PH_RUN table.
USER COLUMNS	any	User Column

The command_history Table

The **command_history** table contains a list of all commands that the Administration API ran. This table, which is in the **sysadmin** database, also shows the results of the commands.

The **command_history** table shows if an administrative task was executed through an **admin()** or **task()** function and displays information about the user who executed the command, the time the command was executed, the command, and the message returned when the database server completed running the command.

Table 3-1. Example Showing command_history Table Information

Column	Data Type	Description
cmd_number	serial	Unique ID for each row
cmd_exec_time	datetime year-to-second	Time the command started
cmd_user	vvarchar	User who executed the command
cmd_hostname	vvarchar	Name of the host computer from which the command was executed
cmd_executed	vvarchar	The command that was executed
cmd_ret_status	integer	Return code
cmd_ret_msg	lvarchar	Return message

The following table shows sample commands and the associated results in a **command_history** table.

Table 3-2. Example of some Information in a `command_history` Table

Command Executed	Sample Returned Messages
set sql tracing on	SQL tracing on with 1000 buffers of 2024 bytes.
create dbspace	Space 'space12' added.
checkpoint	Checkpoint completed.
add log	Added 3 logical logs to dbspace logdbs.

To display the command history, run this SQL statement:

```
SELECT * from command_history
```

Tasks in the `command_history` table are automatically removed after a fixed period of time. You can modify this time period by changing information in the `COMMAND HISTORY RETENTION` row in the `ph_threshold` table. The `COMMAND HISTORY RETENTION` parameter sets the length of time rows should remain in the `command_history` table.

You can use SQL commands like `delete` or `truncate table` to manually remove data from this table.

Chapter 4. Disk Structures and Storage

In This Chapter

The database server achieves its high performance by managing its own I/O. The database server manages storage, search, and retrieval. As the database server stores data, it creates the structures it needs to search for and retrieve the data later. The database server disk structures also store and track control information needed to manage logging and backups. Database server structures contain all the information needed to ensure data consistency, both physical and logical.

Before you read this chapter, familiarize yourself with the disk-space terms and definitions in the chapter on where data is stored in the *IBM Informix Administrator's Guide*.

This chapter discusses the following topics related to disk data structures:

- Dbspace structure and storage
- Storage of simple large objects
- Sbspace structure
- Time stamps
- Database and table creation: what happens on disk

Dbspace Structure and Storage

This section explores the disk structures and storage techniques that the database server uses to store data in a dbspace.

Structure of the Root Dbspace

The ROOTNAME, ROOTOFFSET, ROOTPATH, and ROOTSIZE configuration parameters specify the size and location of the initial chunk of the root dbspace. If the root dbspace is mirrored, the MIRROROFFSET and MIRRORPATH configuration parameters specify the mirror-chunk location. For more information about these parameters, see Chapter 1, "Configuration Parameters," on page 1-1

As part of disk-space initialization, the database server initializes the following structures in the initial chunk of the root dbspace:

- Twelve reserved pages
- The first chunk free-list page
- The tblspace tblspace
- The database tblspace
- The physical log
- The logical-log files
- **oncheck -pe**

For more information, see "oncheck -ce, -pe: Check the chunk-free list" on page 7-9.

Reserved Pages

The first 12 pages of the initial chunk of the root dbspace are reserved pages. Each reserved page contains specific control and tracking information used by the database server.

To obtain a listing of the contents of your reserved pages, execute the command **oncheck -pr**. To also list information about the physical-log and logical-log pages, including the active physical-log pages, execute **oncheck -pR**.

The following example shows **oncheck -pr** output for interval checkpoints:

Time of checkpoint	10/25/2005 17:05:20
Checkpoint Interval	1234

Figure 4-1. *oncheck -pr* Output

For examples of PAGE_CONFIG reserved page and logical-log file **oncheck -pr** output, see *IBM Informix Administrator's Guide*.

Structure of a Regular Dbspace

After disk-space initialization, you can add new dbspaces. When you create a dbspace, you assign at least one chunk (either raw or cooked disk space) to the dbspace. This chunk is referred to as the initial chunk of the dbspace. Figure 4-2 illustrates the structure of the initial chunk of a regular (nonroot) dbspace.

When the dbspace is first created, it contains the following structures:

- Two reserved pages
- The first chunk free-list page in the chunk
- The tblspace **tblspace** for this dbspace
- Unused pages

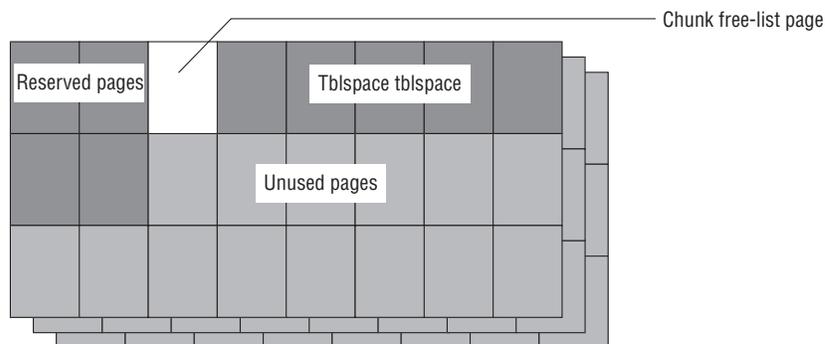


Figure 4-2. *Initial Chunk of Regular Dbspace*

Structure of an Additional Dbspace Chunk

You can create a dbspace that contains more than one chunk. The initial chunk in a dbspace contains the tblspace **tblspace** for the dbspace. Additional chunks do not. When an additional chunk is first created, it contains the following structures:

- Two reserved pages
- The first chunk free-list page
- Unused pages

Figure 4-3 illustrates the structure of all additional chunks in a dbspace. (The structure also applies to additional chunks in the root dbspace.)

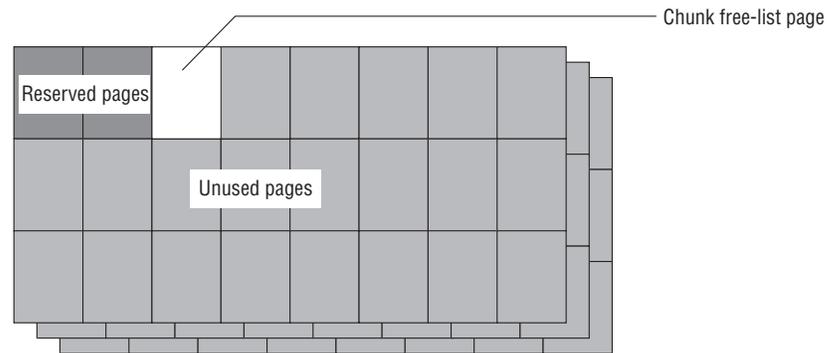


Figure 4-3. Additional Dbspace Chunk

Structure of a Mirror Chunk

Each mirror chunk must be the same size as its primary chunk. When a mirror chunk is created, the database server writes the contents of the primary chunk to the mirror chunk immediately.

The mirror chunk contains the same control structures as the primary chunk. Mirrors of blob space, sbspace, or dbspace chunks contain the same physical contents as their primary counterpart after the database server brings them online.

Figure 4-4 illustrates the mirror-chunk structure as it appears after the chunk is created.

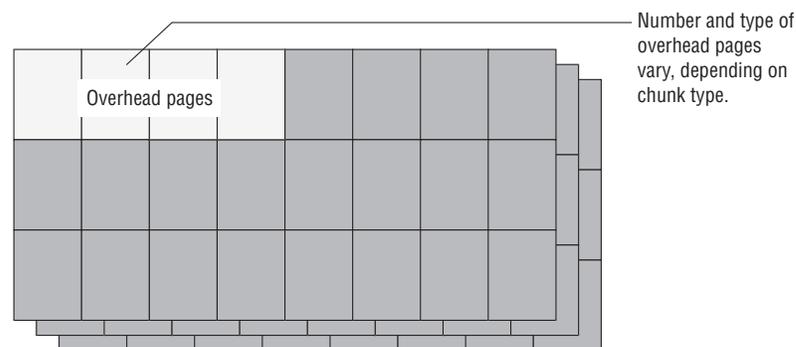


Figure 4-4. Mirror-Chunk Structure

The mirror-chunk structure always shows no free space because all of its space is reserved for mirroring. For more information, see the chapter on what is mirroring in the *IBM Informix Administrator's Guide*.

Structure of the Chunk Free-List Page

In every chunk, the page that follows the last reserved page is the first of one or more chunk free-list pages that tracks available space in the chunk. For a non-root chunk, the initial length of the free space is equal to the size of the chunk minus three pages. If an additional chunk free-list page is needed to accommodate new entries, a new chunk free-list page is created in one of the free pages in the chunk. Figure 4-5 on page 4-4 illustrates the location of the free-list page.

Use **oncheck -pe** to obtain the physical layout of pages in the chunk. For more information, see “oncheck -ce, -pe: Check the chunk-free list” on page 7-9.



Figure 4-5. Free-List Page

Structure of the Tblspace Tblspace

Each dbspace contains a tblspace called the *tblspace tblspace* that describes all tblspaces in the dbspace. When the database server creates a tblspace, it places an entry in the *tblspace tblspace* that describes the characteristics of the newly created tblspace. You cannot drop or move a chunk containing a *tblspace tblspace*.

A dbspace can have a maximum number of 2**20 tblspaces.

The default size of the first and next extents depends on whether the dbspace is the root dbspace or not, as shown in the following table.

Table 4-1. Default sizes for each extent and type of dbspace

Type of dbspace	Default Size of First Extent	Default Size of Next Extents
Root	<ul style="list-style-type: none"> • 500 KB for a 2 kilobyte page system • 1000 KB for a 4 kilobyte page system 	<ul style="list-style-type: none"> • 100 KB for a 2 kilobyte page system • 200 KB for a 4 kilobyte page system
Non-root	<ul style="list-style-type: none"> • 100 KB for a 2 kilobyte page system • 200 KB for a 4 kilobyte page system 	<ul style="list-style-type: none"> • 100 KB for a 2 kilobyte page system • 200 KB for a 4 kilobyte page system

You can specify a non-default size for the first and next extents for a *tblspace tblspace* in the following ways:

- For the root dbspace, set the TBLTBLFIRST and TBLTBLNEXT configuration parameters.
- For non-root dbspaces, use the **onspaces** utility **-ef** and **-en** options when you create a dbspace.

Tblspace Tblspace Entries

To display information on the *tblspace tblspace*, use the **oncheck -pt** command. For more information, see “oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18.

Component	Description
Page header	24 bytes, standard page-header information
Page-ending time stamp	4 bytes
Tblspace header	68 bytes, general <i>tblspace tblspace</i> information

Component	Description
Column information	Each special column in the table is tracked with an 12-byte entry. (A special column is defined as a VARCHAR, BYTE, or TEXT data type.)
Tblspace name	80 bytes, <i>database.owner.tablename</i>
Index information	Each index on the table contains a 20-byte header that contains general information about the index, followed by a 4-byte entry for each column component of the index
Extent information	Each extent allocated to this tblspace is tracked with a 12-byte entry

Tblspace Numbers

Each tblspace that is described in the `tblspace` table receives a tblspace number. This tblspace number is the same value that is stored as the **partnum** field in the **sysables** system catalog table and as the **partn** field in the **sysfragments** system catalog table.

The following SQL query retrieves the **partnum** for every table in the database (these can be located in several different dbspaces) and displays it with the table name and the hexadecimal representation of **partnum**:

```
SELECT tabname, tabid, partnum, HEX(partnum) hex_tblspace_name FROM systables
```

If the output includes a row with a table name but a **partnum** of 0, this table consists of two or more table fragments, each located in its own tblspace. For example, Figure 4-6 shows a table called **account** that has **partnum** 0.

tabname	tabid	partnum	hex_tblspace_name
sysfragments	25	1048611	0x00100023
branch	100	1048612	0x00100024
teller	101	1048613	0x00100025
account	102	0	0x00000000
history	103	1048615	0x00100027
results	104	1048616	0x00100028

Figure 4-6. Output from `systables` Query with `partnum` Values

To obtain the actual tblspace numbers for the fragments that make up the table, you must query the **sysfragments** table for the same database. Figure 4-7 shows that the **account** table from Figure 4-6 has three table fragments and three index fragments.

tabid	fragtype	partn	hex_tblspace_name
102	T	1048614	0x00100026
102	T	2097154	0x00200002
102	T	3145730	0x00300002
102	I	1048617	0x00100029
102	I	2097155	0x00200003
102	I	3145731	0x00300003

Figure 4-7. Output from `sysfragments` Table with `partn` Values

Tblspace Number Elements

The first page in a tblspace is logical page 0. (Physical page numbers refer to the address of the page in the chunk.) The root space tblspace **tblspace** is always contained in the first dbspace and on logical page 1 within the tblspace **tblspace**. (The bitmap page is page 0.)

Tblspace Tblspace Size

These tblspace **tblspace** pages are allocated as an extent when the dbspace is initialized. If the database server attempts to create a table, but the tblspace **tblspace** is full, the database server allocates a next extent to the tblspace.

When a table is removed from the dbspace, its corresponding entry in the tblspace **tblspace** is deleted.

Tblspace Tblspace Bitmap Page

The first page of the tblspace **tblspace**, like the first page of any initial extent, is a bitmap that describes the page fullness of the following pages. Each page that follows has an entry on the bitmap page. If needed, additional bitmap pages are located throughout the contiguous space allocated for the tblspace, arranged so that each bitmap describes only the pages that follow it, until the next bitmap or the end of the dbspace. Bitmap pages fall at distinct intervals within tblspaces pages. Each bitmap page describes a fixed number of pages that follow it.

Structure of the Database Tblspace

The database tblspace appears only in the initial chunk of the root dbspace. The database tblspace contains one entry for each database managed by the database server. Figure 4-8 illustrates the location of the database tblspace.

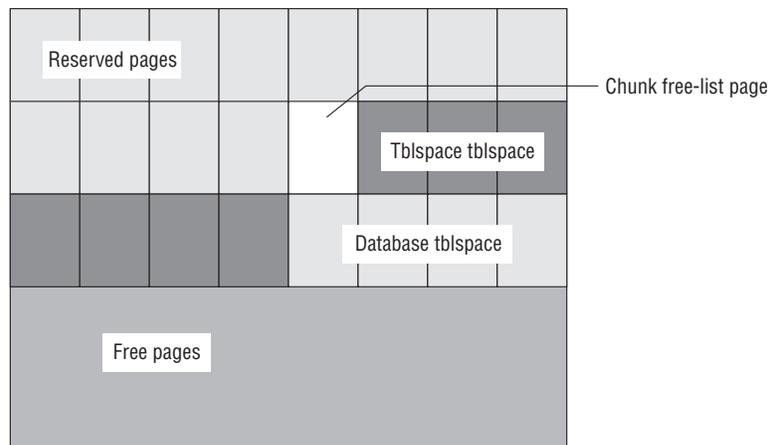


Figure 4-8. Database Tblspace Location in Initial Chunk of Root Dbspace

Database Tblspace Number

The tblspace number of the database tblspace is always 0x100002. This tblspace number appears in an **onstat -t** listing if the database tblspace is active.

Database Tblspace Entries

Each database tblspace entry includes the following five components:

- Database name
- Database owner
- Date and time that the database was created
- The tblspace number of the **systables** system catalog table for this database
- Flags that indicate logging mode

The database tblspace includes a unique index on the database name to ensure that every database is uniquely named. For any database, the **systables** table describes each permanent table in the database. Therefore, the database tblspace only points to the detailed database information located elsewhere.

When the root dbspace is initialized, the database tblspace first extent is allocated. The initial-extent size and the next-extent size for the database tblspace are four pages. You cannot modify these values.

Structure and Allocation of an Extent

This section covers the following topics:

- Extent structure
- Next-extent allocation

Extent Structure

An extent is a collection of contiguous pages within a dbspace. Every permanent database table has two extent sizes associated with it. The initial-extent size is the number of kilobytes allocated to the table when it is first created. The next-extent size is the number of kilobytes allocated to the table when the initial extent, and every extent thereafter, becomes full.

Blobspaces do not use extents.

For specific instructions on how to specify and calculate the size of an extent, see your *IBM Informix Performance Guide*.

Extent Size:

The minimum size of an extent is four pages. The default size of an extent is eight pages. The maximum size of an extent is 2^{31} pages, equivalent to the maximum chunk size. If the chunk is smaller than the maximum size, the maximum extent size depends on the contiguous space available in the chunk.

Tbspaces that hold *index fragments* follow different rules for extent size. The database server bases the extent size for these tablespaces on the extent size for the corresponding table fragment. The database server uses the ratio of the row size to index key size to assign an appropriate extent size for the index tblspace (see the sections on estimating index page size and fragmenting table indexes in the *IBM Informix Performance Guide*).

Page Types Within a Table Extent:

Within the extent, individual pages contain different types of data. Extent pages for a table can be separated into the following categories:

- Data pages

Data pages contain the data rows for the table.

- Bitmap pages

Bitmap pages contain control information that monitors the fullness of every page in the extent.

- Blobpages

Blobpages contain TEXT and BYTE data that is stored with the data rows in the dbspace. TEXT and BYTE data that resides in a blobspace is stored in blobpages, a structure that is completely different than the structure of a dbspace blobpage.

- Free pages

Free pages are pages in the extent that are allocated for tblspace use, but whose function has not yet been defined. Free pages can be used to store any kind of information: data, including TEXT or BYTE data types; index; or bitmap.

Figure 4-9 illustrates the possible structure of a nonfragmented table with an initial-extent size of 8 pages and a next-extent size of 16 pages.

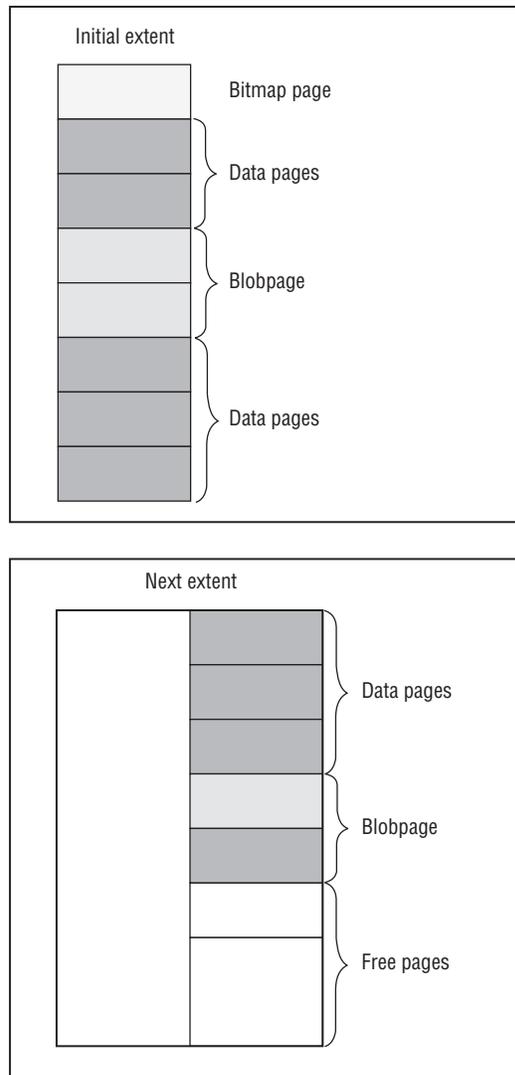


Figure 4-9. Extent Structure of a Table

Page Types Within an Index Extent:

The database server stores index pages into different tablespaces than the table with which it is associated. Within the extent, individual index pages contain different types of data. Index pages can be separated into the following categories:

- Index pages (root, branch, and leaf pages)
Index pages contain the index information for the table.
- Bitmap pages
Bitmap pages contain control information that monitors the fullness of every page in the extent.
- Free pages
Free pages are pages in the extent that are allocated for tablespace use, but whose function has not yet been defined. Free pages can be used to store any kind of information: data, index, TEXT or BYTE data, or bitmap.

All indexes are detached unless you explicitly specify attached indexes.

Important: An extent that is allocated for a table fragment does not contain index pages. Index pages for a fragmented table always reside in a separate tablespace. For more information, see fragmenting table indexes in the chapter on table fragmentation and PDQ in the *IBM Informix Administrator's Guide*.

Figure 4-10 on page 4-10 illustrates the extent structure of an index.

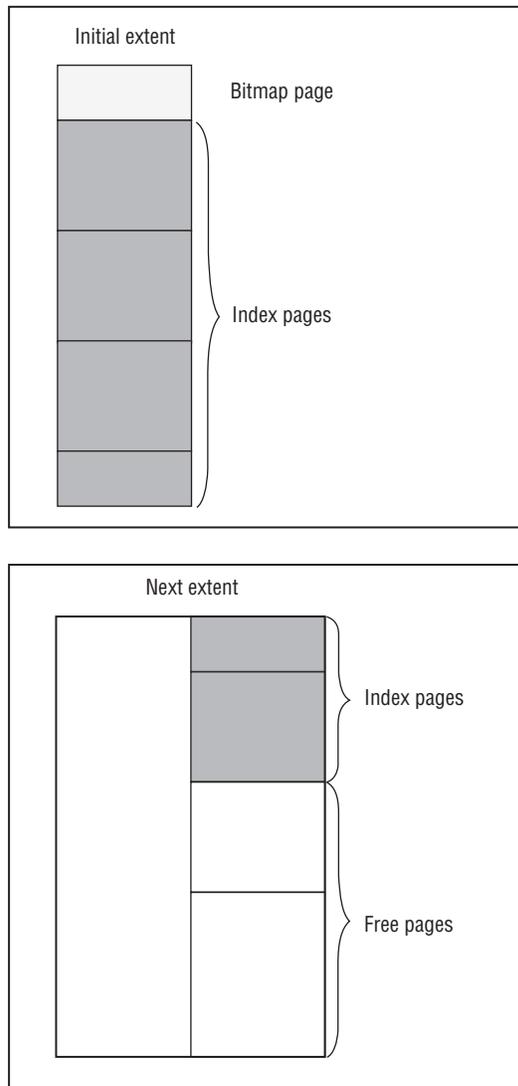


Figure 4-10. Extent Structure of an Index

Next-Extent Allocation

After the initial extent fills, the database server attempts to allocate another extent of contiguous disk space. The procedure that the database server follows is referred to as next-extent allocation.

Extents for a `tblspace` are tracked as one component of the `tblspace` **tblspace** information for the table. The maximum number of extents allocated for any `tblspace` is application and machine dependent because it varies with the amount of space available on the `tblspace` **tblspace** entry.

Next-Extent Size:

The number of kilobytes that the database server allocates for a next extent is, in general, equal to the size of a next extent, as specified in the SQL statement `CREATE TABLE`. However, the actual size of the next-extent allocation might deviate from the specified size because the allocation procedure takes into account the following three factors:

- Number of existing extents for this tblspace
- Availability of contiguous space in the chunk and dbspace
- Location of existing tblspace extents

The effect of each of these factors on next-extent allocation is explained in the paragraphs that follow and in Figure 4-11 on page 4-12.

Extent Size Doubling:

If a permanent table or user-defined temporary table already has 16 extents allocated, the database server automatically doubles the size for subsequent allocations. This doubling occurs every 16 extents. For example, if you create a table with NEXT SIZE equal to 20 kilobytes, the database server allocates the first 16 extents at a size of 20 kilobytes each. The database server allocates extents 17 to 32 at 40 kilobytes each, extents 33 to 48 at 80 kilobytes each, and so on.

The extent size doubling is allowed only if the total number of pages allocated so far is at least 16 times the current next extent size. This is a precautionary measure to limit the exponential doubling of next extent size, if the system has many small holes of pages less than the next extent size, thereby creating a greater number of small extents.

For system-created temporary tables, the next-extent size begins to double after 4 extents have been added.

Lack of Contiguous Space:

If the database server cannot find available contiguous space in the first chunk equal to the size specified for the next extent, it extends the search to the next chunk in the dbspace. Extents are not allowed to span chunks.

If the database server cannot find adequate contiguous space anywhere in the dbspace, it allocates to the table the largest available amount of contiguous space. (The minimum allocation is four pages. The default value is eight pages.) No error message is returned if an allocation is possible, even when the amount of space allocated is less than the requested amount.

Merge of Extents for the Same Table:

If the disk space allocated for a next extent is physically contiguous with disk space already allocated to the same table, the database server allocates the disk space but does not consider the new allocation as a separate extent. Instead, the database server extends the size of the existing contiguous extent. Thereafter, all disk-space reports reflect the allocation as an extension of the existing extent. That is, the number of extents reported is always the number of physically distinct extents, not the number of times a next extent has been allocated plus one (the initial extent). Figure 4-11 on page 4-12 illustrates extent-allocation strategies.

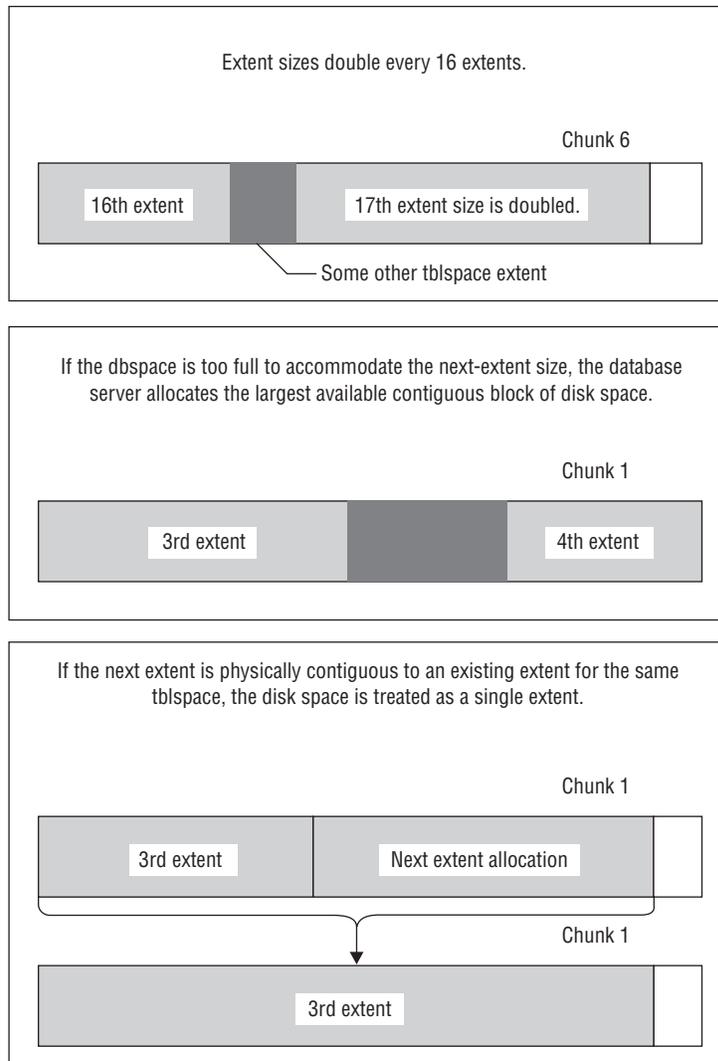


Figure 4-11. Next-Extent Allocation Strategies

After disk space is allocated to a tblspace as part of an extent, the space remains dedicated to that tblspace even if the data contained in it is deleted. For alternative methods of reclaiming this empty disk space, see your *IBM Informix Performance Guide*.

Structure and Storage of a Dbspace Page

The basic unit of database server I/O is a page. Page size might vary among computers.

In Dynamic Server, the page size depends on the operating system.

Rows in Nonfragmented Tables

The database server can store rows that are longer than a page. The database server also supports the VARCHAR data type, which results in rows of varying length. As a result, rows do not conform to a single format.

Rows within a table are not necessarily the same length if the table contains one or more columns of type VARCHAR. In addition, the length of a row in such a table might change when an end user modifies data contained in the VARCHAR column.

The length of a row can be greater than a page.

TEXT and BYTE data is not stored within the data row. Instead, the data row contains a 56-byte descriptor that points to the location of the data. The descriptor can point to a dbspace page.

The descriptor can point to a blobpage. If you are using the Optical Subsystem, the descriptor can also point to an optical-storage subsystem.

For instructions about how to estimate the length of fixed-length and variable-length data rows, see your *IBM Informix Performance Guide*.

Definition of Rowid: Informix uses two different types of rowids to identify data in tables:

- *Serial rowid*

These rowids are fields in a table and are assigned to tables created with the WITH ROWID option.

- *Internal rowid*

The database server identifies each data row in a table with a unique internal rowid. This rowid identifies the location of the row within the dbspace.

To obtain the internal rowids for a table, use the **oncheck -pD** option. For more information, see “oncheck -cd and-cD: Check pages” on page 7-8.

In a nonfragmented table, the term *rowid* refers to a unique 4-byte integer that defines the physical location of the row in the table. The page that contains the first byte of the data row is the page that is specified by the rowid. This page is called the data row *home page*.

Fragmented tables can also have rowids, but they are implemented in a different way. For more information on this topic, see “Rows in Fragmented Tables.”

Use of Rowids: Every data row in a nonfragmented table is uniquely identified by an unchanging rowid. When you create an index for a nonfragmented table, the rowid is stored in the index pages associated with the table to which the data row belongs. When the database server requires a data row, it searches the index to find the key value and uses the corresponding rowid to locate the requested row. If the table is not indexed, the database server might sequentially read all the rows in the table.

Eventually, a row might outgrow its original storage location. If this occurs, a *forward pointer* to the new location of the data row is left at the position defined by the rowid. The forward pointer is itself a rowid that defines the page and the location on the page where the data row is now stored.

Rows in Fragmented Tables

Unlike rows in a nonfragmented table, the database server does *not* assign a rowid to rows in fragmented tables. If you want to access data by rowid, you must explicitly create a rowid column as described in your *IBM Informix Performance*

Guide. If user applications attempt to reference a rowid in a fragmented table that does not contain a rowid that you explicitly created, the database server returns an appropriate error code to the application.

Access to Data in Fragmented Tables with Rowid: From the viewpoint of an application, the functionality of a rowid column in a fragmented table is identical to the rowid of a nonfragmented table. However, unlike the rowid of a nonfragmented table, the database server uses an index to map the rowid to a physical location.

When the database server accesses a row in a fragmented table using the rowid column, it uses this index to look up the physical address of the row before it attempts to access the row. For a nonfragmented table, the database server uses direct physical access without an index lookup. As a consequence, accessing a row in a fragmented table using rowid takes slightly longer than accessing a row using rowid in a nonfragmented table. You should also expect a small performance impact on the processing of inserts and deletes due to the cost of maintaining the rowid index for fragmented tables.

Primary-key access can lead to significantly improved performance in many situations, particularly when access is in parallel.

Recommendations on Use of Rowid

It is recommended that application developers use primary keys as a method of access rather than rowids. Because primary keys are defined in the ANSI specification of SQL, using them to access data makes your applications more portable.

For a complete description on how to define and use primary keys to access data, see the *IBM Informix Guide to SQL: Reference* and the *IBM Informix Guide to SQL: Tutorial*.

Data-Row Format and Storage

The variable length of a data row has the following consequences for row storage:

- A page might contain one or more whole rows.
- A page might contain portions of one or more rows.
- A page might contain a combination of whole rows and partial rows.
- An updated row might increase in size and become too long to return to its original storage location in a row.

The following paragraphs describe the guidelines that the database server follows during data storage.

Storage of Row:

To minimize retrieval time, rows are not broken across page boundaries unnecessarily. Rows that are shorter than a page are always stored as whole rows. A page is considered *full* when the count of free bytes is less than the number of bytes needed to store a row of maximum size.

Location of Rows:

When the database server receives a row that is longer than a page, the row is stored in as many whole pages as required. The database server then stores the trailing portion in less than a full page.

The page that contains the first byte of the row is the row home page. The number of the home page becomes the logical page number contained in the rowid. Each full page that follows the home page is referred to as a big-remainder page. If the trailing portion of the row is less than a full page, it is stored on a remainder page.

After the database server creates a remainder page to accommodate a long row, it can use the remaining space in this page to store other rows.

Figure 4-12 illustrates the concepts of home page, big-remainder page, and remainder page.

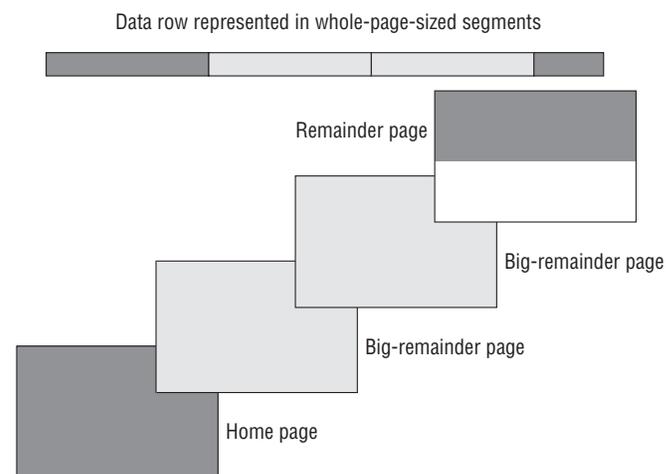


Figure 4-12. Remainder Pages

Page Compression:

Over time, the free space on a page can become fragmented. When the database server attempts to store data, it first checks row length against the number of free bytes on a page to determine if the row fits. If adequate space is available, the database server checks if the page contains adequate contiguous free space to hold the row (or row portion). If the free space is not contiguous, the database server calls for page compression.

Structure of Fragmented Tables

Although table fragmentation is transparent to applications, as database server administrator you should be aware of how the database server allocates disk space for table fragments and how the database server identifies rows in those fragments.

Each table fragment has its own *tblspace* with a unique *tblspace_id* or *fragment_id*. Figure 4-13 on page 4-16 shows the disk allocation for a fragmented table that resides in different partitions of the same *dbspace*.

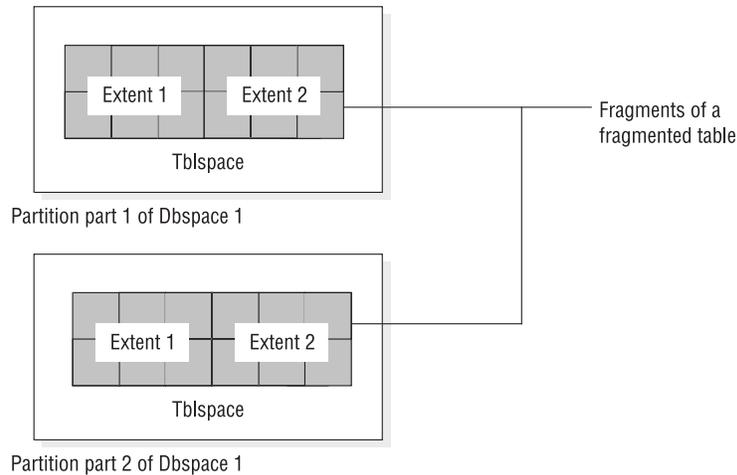


Figure 4-13. Disk Structures for a Fragmented Table

Attached Indexes

With an attached index, the index and data are fragmented in the same way. You can decide whether to store the index pages with the corresponding data pages in the same dbospace or store them in separate dbospaces. For information on choosing a fragmentation strategy, see the *IBM Informix Performance Guide*.

Detached Indexes

For detached indexes, the table fragment and index fragment are stored in tblspaces in separate dbospaces.

Structure of B-Tree Index Pages

This section provides general information about the structure of B-tree index pages. It is designed as an overview for the interested reader. For more information on B-tree indexes, see your *IBM Informix Performance Guide*.

Definition of B-Tree Terms

The database server uses a B-tree structure to organize index information. Figure 4-14 on page 4-17 shows that a fully developed B-tree index is composed of the following three different types of index pages or nodes:

- One *root node*
A root node contains node pointers to branch nodes.
- Two or more *branch nodes*
A branch node contains pointers to leaf nodes or other branch nodes.
- Many *leaf nodes*
A leaf node contains index items and horizontal pointers to other leaf nodes.

Each node serves a different function. The following sections describe each node and the role that it plays in indexing.

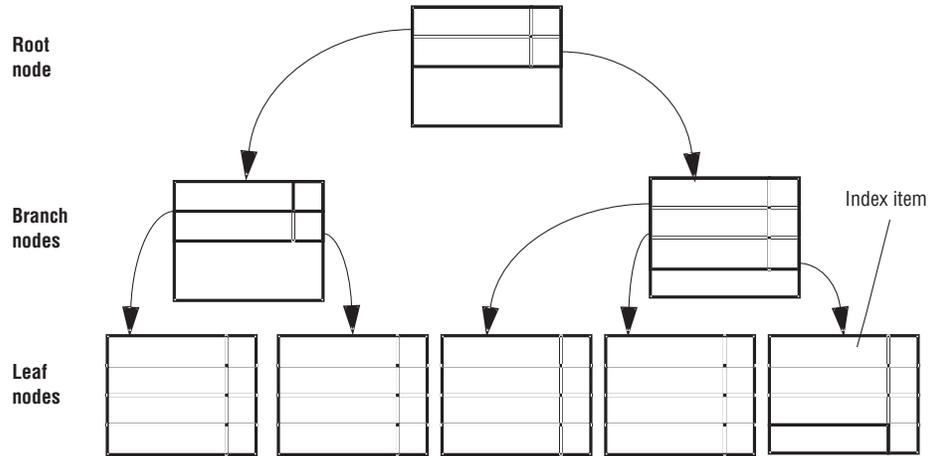


Figure 4-14. Full B-Tree Structure

Index Items

The fundamental unit of an index is the *index item*. An index item contains a key value that represents the value of the indexed column for a particular row. An index item also contains rowid information that the database server uses to locate the row in a data page.

Nodes

A node is an index page that stores a group of index items. For the three types of nodes, see “Definition of B-Tree Terms” on page 4-16.

Logical Storage of Indexes

This section presents an overview of how the database server creates and fills an index.

Creation of Root and Leaf Nodes: When you create an index for an empty table, the database server allocates a single index page. This page represents the root node and remains empty until you insert data in the table.

At first, the root node functions in the same way as a leaf node. For each row that you insert into the table, the database server creates and inserts an index item in the root node. Figure 4-15 illustrates how a root node appears before it fills.

Root node 1	
Albertson	rowid information
Baxter	rowid information
Beatty	rowid information
Currie	rowid information
Keyes	rowid information
Lawson	rowid information
Mueller	rowid information
Wallach	rowid information

Figure 4-15. Root Node

When the root node becomes full of index items, the database server splits the root node by performing the following steps:

- Creates two leaf nodes

- Moves approximately half of the root-node entries to each of the newly created leaf nodes
- Puts pointers to leaf nodes in the root node

As you add new rows to a table, the database server adds index items to the leaf nodes. When a leaf node fills, the database server creates a new leaf node, moves part of the contents of the full index node to the new node, and adds a node pointer to the new leaf node in the root node.

For example, suppose that leaf node 3 in Figure 4-16 becomes full. When this situation occurs, the database server adds yet another leaf node. The database server moves part of the records from leaf node 3 to the new leaf node, as Figure 4-16 shows.

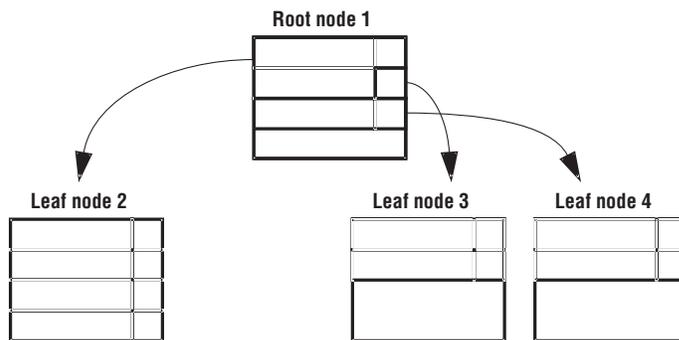


Figure 4-16. Leaf Node 4 Created After Leaf Node 3 Fills

Creation of Branch Nodes: Eventually, as you add rows to the table, the database server fills the root node with node pointers to all the existing leaf nodes. When the database server splits yet another leaf node, and the root node has no room for an additional node pointer, the following process occurs.

The database server splits the root node and divides its contents among two newly created branch nodes. As index items are added, more and more leaf nodes are split, causing the database server to add more branch nodes. Eventually, the root node fills with pointers to these branch nodes. When this situation occurs, the database server splits the root node again. The database server then creates yet another branch level between the root node and the lower branch level. This process results in a four-level tree, with one root node, two branch levels, and one leaf level. The B-tree structure can continue to grow in this way to a maximum of 20 levels.

Branch nodes can point either to other branch nodes below them (for large indexes of four levels or more) or to leaf nodes. In Figure 4-17 on page 4-19, the branch node points to leaf nodes only. The first item in the left branch node contains the same key value as the largest item in the leftmost leaf node and a node pointer to it. The second item contains the largest item in the next leaf node and a node pointer to it. The third item in the branch node contains only a pointer to the next higher leaf node. Depending on the index growth, this third item can contain the actual key value in addition to the pointer at a later point during the lifespan of the index.

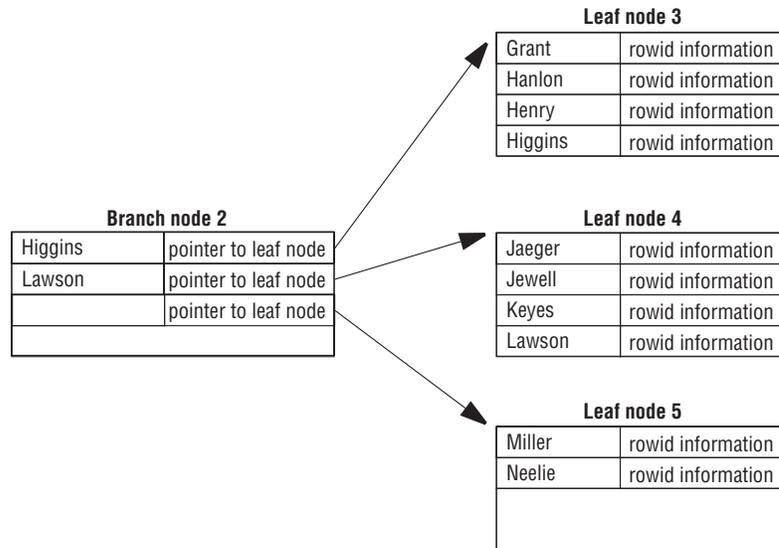


Figure 4-17. Typical Contents of a Branch Node

Duplicate Key Values: Duplicate key values occur when the value of an indexed column is identical for multiple rows. For example, suppose that the third and fourth leaf nodes of a B-tree structure contain the key value Smith. Suppose further that this value is duplicated six times, as Figure 4-18 illustrates.

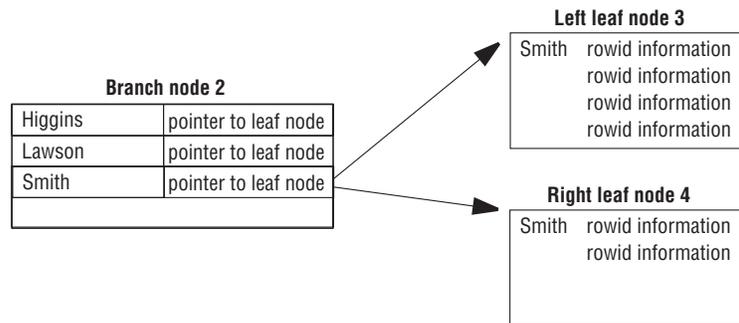


Figure 4-18. Leaf Nodes 3 and 4

The first item on the third leaf page contains the duplicate key value, Smith, and the rowid information for the first physical row in the table that contains the duplicate key value. To conserve space, the second item does not repeat the key value Smith but instead contains just the rowid information. This process continues throughout the page; no other key values are on the leaf, only rowid information.

The first item on the fourth leaf page again contains the duplicated key value and rowid information. Subsequent items contain only rowid information.

Now consider the branch node. The third item in the branch node contains the same key value and rowid as the largest item in the third leaf node and a node pointer to it. The fourth item would contain only a node pointer to the fourth leaf node, thus saving the space of an additional duplicate key value.

Key-Value Locking: To increase concurrency, the database server supports *key-value* locking in the B-tree index. Key-value locking locks only the value of the key instead of the physical location in the B-tree index.

One of the most important uses for key-value locking is to assure that a unique key remains unique through the end of the transaction that deleted it. Without this protection mechanism, user A might delete a unique key within a transaction, and user B might insert a row with the same key before the transaction commits. This scenario makes rollback by user A impossible. Key-value locking prevents user B from inserting the row until the end of user A's transaction.

Adjacent Key Locking: With Repeatable Read isolation level, the database server is required to protect the *read set*. The read set consists of the rows that meet the filters in the WHERE clause of the query. To guarantee that the rows do not change, the database server obtains a lock on the index item that is adjacent to the right-most item of the read set.

Freed Index Pages: When the database server physically removes an index item from a node and frees an index page, the freed page is reused.

Filling Indexes: When you create an index, you can specify how densely or sparsely filled you want the index. The index fill factor is a percentage of each index page that will be filled during the index build. Use the FILLFACTOR option of the CREATE INDEX statement or the FILLFACTOR configuration parameter to set the fill factor. This option is particularly useful for indexes that you do not expect to grow after they are built. For additional information about the FILLFACTOR option of the CREATE INDEX statement, see the *IBM Informix Guide to SQL: Syntax*.

Calculating the Length of Index Items: For data types other than VARCHAR, the length of an index item is calculated by adding the length of the key value plus 5 bytes for each rowid information associated with the key value.

The key values in an index are typically of fixed length. If an index holds the value of one or more columns of the VARCHAR data type, the length of the key value is at least the sum of the length-plus-one of each VARCHAR value in the key.

In Dynamic Server, the maximum length of a key value is 390 bytes. The combined size of VARCHAR columns that make up a key must be less than 390, minus an additional byte for each VARCHAR column. For example, the key length of the index that the database server builds for the following statements equals 390, or $((255+1) + (133+1))$:

```
CREATE TABLE T1 (c1 varchar(255, 10), c2 varchar(133, 10));
CREATE INDEX I1 on T1(c1, c2);
```

Functional Indexes

A *functional index* is one in which all keys derive from the results of a function. If you have a column of pictures, for example, and a function to identify the predominant color, you can create an index on the result of the function. Such an index would enable you to quickly retrieve all pictures having the same predominant color, without re-executing the function.

A functional index uses the same B-tree structure as any other B-tree index. The only difference is that the determining function is applied during an insert or an update whenever the column that is the argument to the function changes. For more information on the nature of functional indexes, refer to your *IBM Informix Performance Guide*.

To create a functional index, use the CREATE FUNCTION and CREATE INDEX statements. For more information on these statements, refer to the *IBM Informix Guide to SQL: Syntax*.

Structure of R-Tree Index Pages

An index structure that relies on one-dimensional ordering of key values does not work for spatial data; for example, two dimensional geometric shapes such as circles, squares, and triangles. Efficient retrieval of spatial data, such as the data used in geographic information systems (GIS) and computer-aided design (CAD) applications, requires an access method that handles multidimensional data. The database server implements an R-tree index to access spatial data efficiently. For information about the structure of index pages, refer to the *IBM Informix R-Tree Index User's Guide*.

Storage of Simple Large Objects

This section explains the structures and storage techniques that the database server uses to store simple large objects (TEXT or BYTE data).

Structure of a Blobpage

When you create a blobpage, you can specify the effective size of the data pages, which are called blobpages. The blobpage size for the blobpage is specified when the blobpage is created. Blobpage size must be a multiple of page size. (For information on determining database server page size, see the chapter on managing disk space in the *IBM Informix Administrator's Guide*.) All blobpages within a blobpage are the same size, but the size of the blobpage can vary between blobpages. Blobpage size can be greater than the page size because data stored in a blobpage is never written to the page-sized buffers in shared memory.

The advantage of customizing the blobpage size is storage efficiency. Within a blobpage, TEXT and BYTE data is stored in one or more blobpages, but simple large objects do not share blobpages. Storage is most efficient when the TEXT or BYTE data is equal to or slightly smaller than the blobpage size.

The blobpage free-map pages and bitmap pages are the size specified as a database server page, which enables them to be read into shared memory and to be logged.

When the blobpage is first created, it contains the following structures:

- Blobpage free-map pages
- The blobpage bitmap that tracks the free-map pages
- Unused blobpages

Structure of a Dbspace Blobpage

TEXT or BYTE data that is stored in the dbspace is stored in a blobpage. The structure of a dbspace blobpage is similar to the structure of a dbspace data page. The only difference is an extra 12 bytes that can be stored along with the TEXT or BYTE data in the data area.

Simple large objects can share dbspace blobpages if more than one simple large object can fit on a single page, or if more than one trailing portion of a simple large object can fit on a single page.

For a discussion of how to estimate the number of dbspace blobpages needed for a specific table, see your *IBM Informix Performance Guide*.

Each segment of TEXT or BYTE data stored in a dbspace page might be preceded by up to 12 bytes of information that does not appear on any other dbspace page. These extra bytes are overhead.

Simple-Large-Object Storage and the Descriptor

Data rows that include TEXT or BYTE data do not include the data in the row itself. Instead, the data row contains a 56-byte descriptor with a forward pointer (rowid) to the location where the first segment of data is stored.

The descriptor can point to one of the following items:

- A page (if the data is stored in a dbspace)
- A blobpage (if the data is stored in a blobspace)
- An optical platter (if you are using the Optical Subsystem)

Creation of Simple Large Objects

When a row that contains TEXT or BYTE data is to be inserted, the simple large objects are created first. After the simple large objects are written to disk (or optical medium), the row is updated with the descriptor and inserted.

Deletion or Insertion of Simple Large Objects

The database server cannot modify simple large objects. It can only insert or delete them. Deleting a simple large object means that the database server frees the space consumed by the deleted object for reuse.

When TEXT or BYTE data is updated, a new simple large object is created, and the data row is updated with the new blob descriptor. The old image of the row contains the descriptor that points to the obsolete value for the simple large object. The space consumed by the obsolete simple large object is freed for reuse after the update is committed. Simple large objects are automatically deleted if the rows that contain their blob descriptors are deleted. (Blobpages that stored a deleted simple large object are not available for reuse until the logical log that contains the original INSERT record for the deleted simple large object is backed up. For more information, see backing up logical-log files to free blobpages in the chapter on what is the logical log in the *IBM Informix Administrator's Guide*.)

Size Limits for Simple Large Objects

The largest simple large object that the blob descriptor can accommodate is $(2^{31} - 1)$, or about 2 gigabytes.

Blobspace Page Types

Every blobspace chunk contains three types of pages:

- A blobspace free-map page
- A bitmap page
- Blobpages

Blobspace Free-Map Page

The blobspace free-map page identifies unused blobpages so that the database server can allocate them as part of simple-large-object creation. When a blobpage is allocated, the free-map entry for that page is updated. All entries for a single simple large object are linked.

A blobspace free-map page is the size of one database server page. Each entry on a free-map page is 8 bytes, stored as two 32-bit words, as follows:

- The first bit in the first word specifies whether the blobpage is free or used.
- The next 31 bits in the first word identify the logical-log file that was current when this blobpage was written. (This information is needed for logging TEXT or BYTE data.)
- The second word contains the tblspace number associated with the simple large object stored on this page.

The number of entries that can fit on a free-map page depends on the page size of your computer. The number of free-map pages in a blobspace chunk depends on the number of blobpages in the chunk.

Blobspace Bitmap Page

The blobspace bitmap page tracks the fullness and number of blobspace free-map pages in the chunk. Each blobspace bitmap page is capable of tracking a quantity of free-map pages that represent more than 4,000,000 blobpages. Each blobspace bitmap page is the size of one page.

Blobpage

The blobpage contains the TEXT or BYTE data. Blobpage size is specified by the database server administrator who creates the blobspace. Blobpage size is specified as a multiple of the page size.

Structure of a Blobspace Blobpage

The storage strategy used to store simple large objects in a blobspace differs from the dbspace storage strategy. The database server does not combine whole simple large objects or portions of a simple large object on a single blobspace blobpage. For example, if blobspace blobpages are 24 kilobytes each, a simple large object that is 26 kilobytes is stored on two 24-kilobyte pages. The extra 22 kilobytes of space remains unused.

The structure of a blobpage includes a blobpage header, the TEXT or BYTE data, and a page-ending time stamp. The blobpage header includes, among other information, the page-header time stamp and the blob time stamp associated with the forward pointer in the data row. If a simple large object is stored on more than one blobpage, a forward pointer to the next blobpage and another blob time stamp are also included in the blobpage header.

Sbospace Structure

An sbospace is similar to a blobspace except that it holds smart large objects.

When an sbspace is created in a database, it contains an sbspace descriptor. Each sbspace chunk contains the following structures:

- Sbspace chunk descriptors
- Chunk free-page list
- An sbspace metadata area (up to one for each chunk)
- Reserved data areas (up to two for each chunk)
- User-data areas (up to two for each chunk)

For best performance, it is recommended that the metadata area be located in the middle of the sbspace. The database server automatically places the metadata area in the correct location. However, to specify the location of the metadata area, specify the **-Mo** flag in the **onspaces** command.

If you do not specify the size of the metadata area in the **-Ms** flag of the **onspaces** command, the database server uses the value of `AVG_LO_SIZE` (defaults to 8 kilobytes) to calculate the size of the metadata area. For more information, see “Creating an Sbspace with the **-Df** option” on page 14-12.

Normally, you can let the system calculate the metadata size for you. If you want to estimate the size of the metadata area, see the chapter on table performance considerations in the *IBM Informix Performance Guide*.

Figure 4-19 illustrates the chunk structure of an sbspace as it appears immediately after the sbspace is created. Each reserved area can be allocated to either the user-data or metadata area. Reserved areas are always within the user-data area of the chunk.

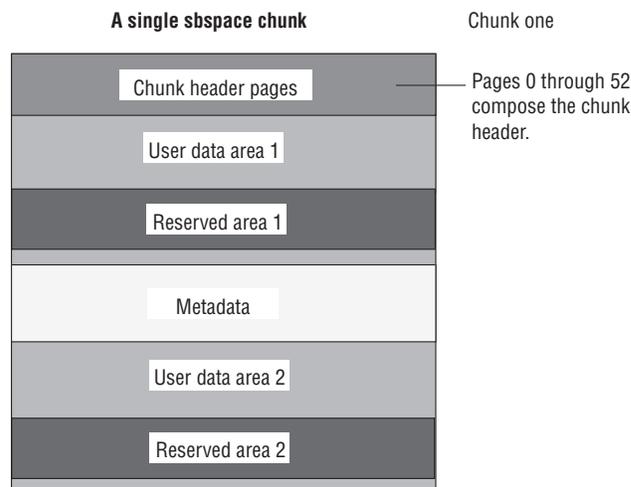


Figure 4-19. A Single Sbspace Chunk

Because the chunk in Figure 4-19 is the first in the sbspace, it contains an sbspace descriptor. The chunk descriptor `tblspace` in **chunk one** contains information about chunk one and all chunks added to the sbspace thereafter.

Structure of the Metadata Area

As with the chunk header pages, four areas are exclusive to the first chunk in a sbspace: the sbspace descriptor `tblspace`, the chunk adjunct `tblspace`, and the level-1 and level-2 archive `tblspaces`. The `tblspace` header section contains a

tblspace header for each of these tblspaces (notably excluding the tblspace **tblspace**). Figure 4-20 shows the layout of the metadata in the single-chunk sbspace.

Structure of the metadata area for a single-chunk sbspace

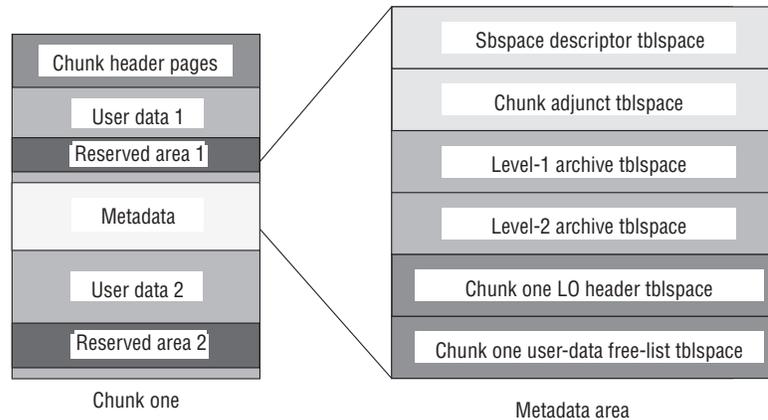


Figure 4-20. Structure of the Metadata Area for a Single-Chunk Sbspace

When you specify the sbspace name in the **oncheck -ps** option, you can display the number of pages allocated and used for each tblspace in the metadata area.

The following describes how the metadata area grows:

- The sbspace descriptor tblspace does not grow.
- The chunk adjunct tblspace grows as chunks are added.
- The LO header tblspace grows as chunks are added.
- The tblspace for user-data free list grows if free spaces in the chunk are heavily fragmented.

Sbpage Structure

Each sbpage is composed of three elements: an sbpage header, the actual user data itself, and an sbpage trailer. Figure 4-21 on page 4-26 shows the structure of an sbpage. The sbpage header consists of the standard page header. The sbpage trailer is used to detect an incomplete write on the page and to detect page corruption.

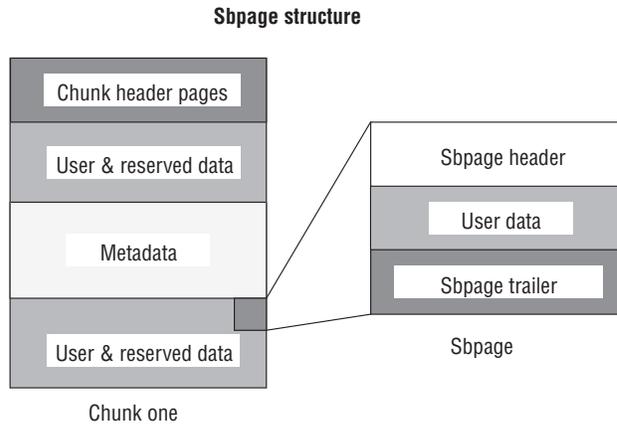


Figure 4-21. Sbpage Structure

Multiple Chunk Sbspace

Figure 4-22 illustrates a possible configuration for a three-chunk sbspace. In this example, **chunk two** contains no metadata of its own. Metadata information for **chunk two** is stored in the metadata area of **chunk one**.

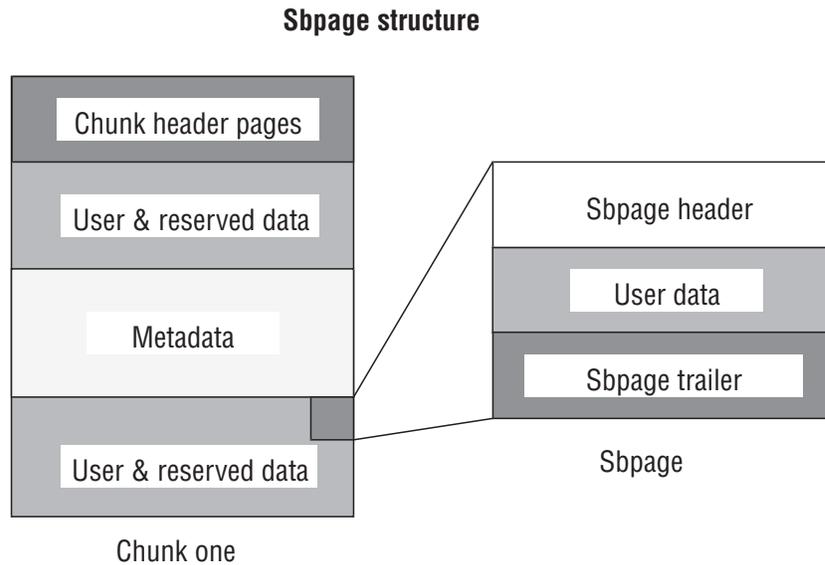


Figure 4-22. Multiple-Chunk Sbspace Structure

The user-data area in **chunk one** of the example is actually optional. **Chunk one** could contain metadata for all other chunks in the sbspace.

Time Stamps

The database server uses a time stamp to identify a time when an event occurred relative to other events of the same kind. The time stamp is not a literal time that refers to a specific hour, minute, or second. It is a 4-byte integer that the database server assigns sequentially.

Database and Table Creation: What Happens on Disk

This section explains how the database server stores data related to the creation of a database or table and allocates the disk structures that are necessary to store your data.

Database Creation

After the root dbspace exists, users can create a database. The paragraphs that follow describe the major events that occur on disk when the database server adds a new database.

Disk-Space Allocation for System Catalog Tables

The database server searches the chunk free-list pages in the dbspace, looking for free space in which to create the system catalog tables. For each system catalog table, in turn, the database server allocates eight contiguous pages, the size of the initial extent of each system catalog table. The tables are created individually and do not necessarily reside next to each other in the dbspace. They can be located in different chunks. As adequate space is found for the initial extent of each table, the pages are allocated, and the associated chunk free-list page is updated.

Tracking of System Catalog Tables

The database server tracks newly created databases in the database tblspace, which resides in the root dbspace. An entry describing the database is added to the database tblspace in the root dbspace. (See “Structure of the Database Tblspace” on page 4-6.) For each system catalog table, the database server adds a one-page entry to the tblspace **tblspace** in the dbspace where the database was built. (See “Structure of the Tblspace Tblspace” on page 4-4.) Figure 4-23 illustrates the relationship between the database tblspace entry and the location of the **systables** system catalog table for the database.

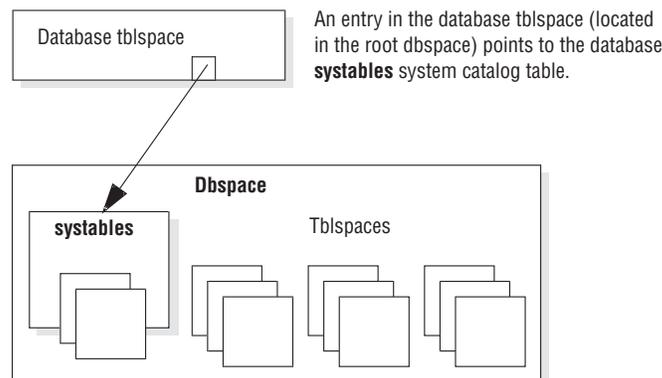


Figure 4-23. New Databases

For instructions on how to list your databases after you create them, see monitoring databases in the chapter on managing database-logging status in the *IBM Informix Administrator's Guide*.

Table Creation

After the root dbspace exists, and a database has been created, users with the necessary SQL privileges can create a database table. When users create a table, the database server allocates disk space for the table in units called extents (see what is an extent in the chapter on where data is stored in the *IBM Informix Administrator's*

Guide). The paragraphs that follow describe the major events that occur when the database server creates a table and allocates the initial extent of disk space.

Disk-Space Allocation

The database server searches the chunk free-list pages in the dbspace for contiguous free space equal to the initial extent size for the table. When adequate space is found, the pages are allocated, and the associated chunk free-list page is updated.

If the database server cannot find adequate contiguous space anywhere in the dbspace, it allocates to the table the largest available amount of contiguous space. No error message is returned if an allocation is possible, even when the amount of space allocated is less than the requested amount. If the minimum extent size cannot be allocated, an error is returned. (Extents cannot span two chunks.)

Entry in the Tblspace Tblspace

The database server adds a one-page entry for this table to the tblspace **tblspace** in this dbspace. The tblspace number assigned to this table is derived from the logical page number in the tblspace **tblspace** where the table is described. See “Tblspace Numbers” on page 4-5.

The tblspace number indicates the dbspace where the tblspace is located. Tblspace extents can be located in any of the dbspace chunks.

If you must know exactly where the tblspace extents are located, execute the **oncheck -pe** command for a listing of the dbspace layout by chunk.

Entries in the System Catalog Tables

The table itself is fully described in entries stored in the system catalog tables for the database. Each table is assigned a table identification number or *tabid*. The *tabid* value of the first user-defined table object in a database is always 100. (The object whose *tabid* = 100 might also be a view, synonym, or a sequence.) For a complete discussion of the system catalog, see the *IBM Informix Guide to SQL: Reference*.

A table can be located in a dbspace that is different than the dbspace that contains the database. The tblspace itself is the sum of allocated extents, not a single, contiguous allocation of space. The database server tracks tblspaces independently of the database.

Creation of a Temporary Table

The tasks involved in creating temporary tables are similar to the tasks that the database server performs when it adds a new permanent table. The key difference is that temporary tables do not receive an entry in the system catalog for the database. For more information, see the section defining a temporary table, in the chapter on where data is stored in the *IBM Informix Administrator's Guide*.

Chapter 5. Interpreting Logical-Log Records

In This Chapter

To display the logical-log records that the logical-log files contain, use the **onlog** utility.

This chapter provides the following information:

- Brief guidance on reading logical-log records
- A listing of the different logical-log record types

In general, you do not need to read and interpret your logical-log files. However, **onlog** output is useful in debugging situations. For example, you might want to use **onlog** to track a specific transaction or to see what changes the database server made to a specific tblspace. You can also use **onlog** to investigate the cause of an error that occurs during a rollforward. For more information, see “onlog: Display Logical-Log Contents” on page 10-1

About Logical-Log Records

Most SQL statements generate multiple logical-log records. Interpreting logical-log records is more complicated when the database server records the following events in the logical log:

- A transaction that drops a table or index
- A transaction that rolls back
- A checkpoint in which transactions are still active
- A distributed transaction

The following sections discuss the logical-log records for these events.

Transactions That Drop a Table or Index

Once the database server drops a table or index from a database, it cannot roll back that drop operation. If a transaction contains a DROP TABLE or DROP INDEX statement, the database server handles this transaction as follows:

1. The database server completes all the other parts of the transaction and writes the relevant logical-log records.
2. The database server writes a BEGCOM record to the logical log and the records associated with the DROP TABLE or DROP INDEX (DINDEX, for example).
3. The database server writes a COMMIT record.

If the transaction is terminated unexpectedly after the database server writes the BEGCOM record to the logical log, the database server rolls *forward* this transaction during recovery because it cannot roll back the drop operation.

Transactions That Are Rolled Back

When a rollback occurs, the database server generates a compensation-log record (CLR) for each record in the logical log that is rolled back. The database server uses the CLRs if a system failure takes place *during a rollback*. The CLRs provide

the database server with information on how far the rollback progressed before the failure occurred. In other words, the database server uses the CLR to log the rollback.

If a CLR contains the phrase includes next record, the next log record that is printed is included within the CLR log record as the compensating operation. Otherwise, you must assume that the compensating operation is the logical undo of the log record to which the **link** field of the CLR points.

Checkpoints with Active Transactions

If any transactions are active at the time of a checkpoint, checkpoint records include subentries that describe each of the active transactions using the following columns:

- Log begin (decimal format)
- Transaction ID (decimal format)
- Unique log number (decimal format)
- Log position (hexadecimal format)
- User name

Distributed Transactions

When distributed transactions (transactions that span multiple database servers) generate log records, they are slightly different than nondistributed transactions. You might need to read and interpret them to determine the state of the transaction on both database servers if a failure occurs as a transaction was committing.

The following log records are involved in distributed transactions:

- BEGPREP
- ENDTRANS
- HEURTX
- PREPARE
- TABLOCKS

For more information about this type of logical-log record, see the material on two-phase commit and logical-log records in the *IBM Informix Administrator's Guide*.

If you are performing distributed transactions with TP/XA, the database server uses an XAPREPARE record instead of a PREPARE record.

Logical-Log Record Structure

Each logical-log record has *header* information. Depending on the record type, additional columns of information also appear in the output, as explained in "Logical-Log Record Types and Additional Columns" on page 5-3.

Logical-Log Record Header

Table 5-1 on page 5-3 contains sample output to illustrate the header columns that display for a logical-log record.

Table 5-1. Sample Output from onlog

addr	len	type	xid	id	link
2c018	32	BEGIN	6	3	0
2c038	140	HDELETE	6	0	2c018
2c0c4	64	DELITEM	6	0	2c038
2c104	40	DELITEM	6	0	2c0c4
2c12c	72	HDELETE	6	0	2c104
2c174	44	DELITEM	6	0	2c12c
2c1a0	72	HDELETE	6	0	2c174
2c1e8	44	DELITEM	6	0	2c1a0
2c214	64	HDELETE	6	0	2c1e8
2c254	56	DELITEM	6	0	2c214
2c28c	48	DELITEM	6	0	2c254
2c2bc	24	PERASE	6	0	2c28c
2c2d4	20	BEGCOM	6	0	2c2bc
2c2e8	24	ERASE	6	0	2c2d4
2c300	28	CHFREE	6	0	2c2e8
2c31c	24	COMMIT	6	0	2c300

Table 5-2 defines the contents of each header column.

Table 5-2. Definition of onlog Header Columns

Header Field	Contents	Format
addr	Log-record address (log position)	Hexadecimal
len	Record length in bytes	Decimal
type	Record-type name	ASCII
xid	Transaction number	Decimal
id	Logical-log number	Decimal
link	Link to the previous record in the transaction	Hexadecimal

Logical-Log Record Types and Additional Columns

In addition to the six header columns that display for every record, some record types display additional columns of information. The information that appears varies, depending on record type. Table 5-3 on page 5-4 lists all the record types and their additional columns.

The **Action** column indicates the type of database server action that generated the log entry. The **Additional Columns** and **Format** columns describe what information appears for each record type in addition to the header described in “Logical-Log Record Header” on page 5-2.

Table 5-3. Logical-Log Record Types

Record Type	Action	Additional Columns	Format
ADDCHK	Add chunk.	chunk number	Decimal
		chunk name	ASCII
ADDDBS	Add dbspace.	dbspace name	ASCII
ADDITEM	Add item to index.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		logical page	Decimal
		key number	Decimal
		key length	Decimal
ADDLOG	Add log.	log number	Decimal
		log size (pages)	Decimal
		pageno	Hexadecimal
ALLOCGENPG	Allocate a generic page.	tblspace ID	Decimal
		rowid	Decimal
		slot flags and length	Decimal
		page version if delete	Decimal
		flags, vimage record	Decimal
		rowid for previous	Decimal
		data	ASCII
ALTERDONE	Alter of fragment complete.	tblspace ID	Hexadecimal
		physical page number previous page	Hexadecimal
		logical page number	Decimal
		version of alter	Decimal
ALTSPCOLSNEW	Changed columns in an alter table.	number of columns	Decimal
		special column list	array
ALTSPCOLSOLD	Changed columns in an alter table.	number of columns	Decimal
		special column list	array
BADIDX	Bad index	tblspace ID	Hexadecimal
BEGCOM	Begin commit.	(None)	(None)
BEGIN	Begin work.	date	Decimal
		time	Decimal
		SID	Decimal
		user	ASCII
BEGPREP	Written by the coordinator database server to record the start of the two-phase commit protocol.	flags	Decimal (Value is 0 in a distributed transaction.)
		number of participants	Decimal
BEGWORK	Begin a transaction.	begin transaction time	Decimal
		user ID	Decimal
		process ID	Decimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
BFRMAP	Simple-large-object free-map change.	tblspace ID	Hexadecimal
		bpageno	Hexadecimal
		status	USED/FREE
		log ID	Decimal
		prev page	Hexadecimal
BLDCL	Build tblspace.	tblspace ID	Hexadecimal
		fextsize	Decimal
		nextsize	Decimal
		row size	Decimal
		ncolumns	Decimal
		table name	ASCII
BMAPFULL	Bitmap modified to prepare for alter.	tblspace ID	Hexadecimal
		bitmap page num	Decimal
BMAP2TO4	2-bit bitmap altered to two 4-bit bitmaps.	tblspace ID	Hexadecimal
		2-bit bitmap page number	Decimal
		flags	Decimal
BSPADD	Add blobspace.	blobspace name	ASCII
BTCPYBCK	Copy back child key to parent.	tblspace ID	Hexadecimal
		parent logical page	Decimal
		child logical page	Decimal
		slot	Decimal
		rowoff	Decimal
		key number	Decimal
BTMERGE	Merge B-tree nodes.	tblspace ID	Hexadecimal
		parent logical page	Decimal
		left logical page	Decimal
		right logical page	Decimal
		left slot	Decimal
		left rowoff	Decimal
		right slot	Decimal
		right rowoff	Decimal
		key number	Decimal
BTSHUFFL	Shuffle B-tree nodes.	tblspace ID	Hexadecimal
		parent logical page	Decimal
		left logical page	Decimal
		right logical page	Decimal
		left slot	Decimal
		left rowoff	Decimal
		key number	Decimal
		flags	Hexadecimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
BTSPLIT	Split B-tree node.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		parent logical page	Decimal
		left logical page	Decimal
		right logical page	Decimal
		infinity logical page	Decimal
		rootleft logical page	Decimal
		midsplit	Decimal
		key number	Decimal
	key length	Decimal	
CDINDEX	Create detached index.	database name	ASCII
		owner	ASCII
		table name	ASCII
		index name	ASCII
CDR	<p>Captures the set of table columns modified by an update statement such as a <i>bitvector</i>. This log record allows Enterprise Replication to capture only the changed data to avoid transmitting the unchanged columns to a target site.</p> <p>In the example, the first six columns of the table are unchanged (6 leftmost bits in the bitvector are 0), the seventh and eighth columns have been updated (seventh and eighth bits are 1), and so on. The onlog output displays as many bits of bitvector as fit in a single line of the output. To see the entire bitvector displayed in hexadecimal, use the onlog -l command.</p>	name of CDR record	ASCII
		partition number	Hexadecimal
		bitvector	Binary
<p>Sample onlog output for CDR log record:</p> <pre>adr len type xid id link name partno bitvector 40 36 CDR 14 0 18 UPDCOLS 10009a 000000110100110100</pre>			
CHALLOC	Chunk extent allocation.	pageno	Hexadecimal
		size	Hexadecimal
CHCOMBINE	Chunk extent combine.	pageno	Hexadecimal
CHFREE	Chunk extent free.	pageno	Hexadecimal
		size	Hexadecimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
CHKADJUP	Update chunk adjunct on disk. The database server writes this record when it moves space from the reserved area to the metadata or user-data area or when the user adds an sbpace chunk.	chunk number	Integer
		ud1_start_page	Integer
		ud1_size	Integer
		md_start_page	Integer
		md_size	Integer
		ud2_start_page	Integer
		ud2_size	Integer
CHPHYLOG	Change physical-log location.	flags	Hexadecimal
		pageno	Hexadecimal
		size in kilobytes	Hexadecimal
CHRESERV	Reserve extent for metadata stealing. This record is written when you add an sbpace chunk.	dbspace name	ASCII
		chunk number	Integer
		page number	Integer
CHSPLIT	Chunk extent split.	length	Integer
		pageno	Hexadecimal
CINDEX	Create index.	index descriptor	ASCII
		tblspace ID	Hexadecimal
		low rowid	Decimal
		high rowid	Decimal
COARSELOCK	Coarse-grain locking	tblspace ID	Hexadecimal
		old coarse-locking flag value	Decimal
		new coarse-locking flag value	Decimal
CKPOINT	Checkpoint.	max users	Decimal
		number of active transactions	Decimal
CLR	Compensation-log record; created during a rollback.	(None)	(None)
CLUSIDX	Create clustered index.	tblspace ID	Hexadecimal
		key number	Decimal
COLREPAI	Adjust BYTE, TEXT, or VARCHAR column.	tblspace ID	Hexadecimal
		number of columns adjusted	Decimal
COMMIT	Commit work.	date	Decimal
		time	Decimal
COMTAB	Compact slot table on a page.	logical page number	Decimal
		number slots moved	Decimal
		compressed slot pairs	ASCII
COMWORK	End a transaction and commit work.	end transaction time	Decimal
		begin transaction time	Decimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
DELETE	Delete before-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
DELITEM	Delete item from index.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		logical page	Decimal
		key number	Decimal
		key length	Decimal
DERASE	Drop tblspace in down dbspace.	tblspace number	Hexadecimal
		table lock number	Decimal
DINDEX	Drop index.	tblspace ID	Hexadecimal
		key number	Decimal
DRPBSP	Drop blobspace.	blobspace name	ASCII
DRPCHK	Drop chunk.	chunk number	Decimal
		chunk name	ASCII
DRPDBS	Drop dbspace.	dbspace name	ASCII
DRPLOG	Drop log.	log number	Decimal
		log size (pages)	Decimal
		pageno	Hexadecimal
ENDTRANS	<p>Written by both the coordinator and participant database servers to record the end of the transaction. ENDTRANS instructs the database server to remove the transaction entry from its shared-memory transaction table and close the transaction.</p> <p>In the coordinator logical log, each BEGPREP that results in a committed transaction is paired with an ENDTRANS record. If the final decision of the coordinator is to roll back the transaction, no ENDTRANS record is written.</p> <p>In the participant logical log, each ENDTRANS record is paired with a corresponding HEURTX record.</p>	(None)	(None)
ERASE	Drop tblspace.	tblspace ID	Hexadecimal
FREE_RE	Allocate extent from reserve extent to metadata or user-data area of an sbspace chunk.	chunk number	Integer
		page number	Integer
		length	Integer
		flag	Hexadecimal
HDELETE	Delete home row.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
HEURTX	Written by a participant database server to record a heuristic decision to roll back the transaction. It should be associated with a standard ROLLBACK record indicating that the transaction was rolled back.	flag	Hexadecimal (Value is always 1.)
HINSERT	Home row insert.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
HUPAFT	Home row update, after-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
HUPBEF	Home row update, before-image. In addition, the flag field of the HUPBEF header may include the following values: LM_PREVLSN Confirms that an LSN exists. LM_FIRSTUPD Confirms that this is the first update for this rowID by this transaction.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
		LSN (optional)	Decimal
HUPDATE	If the home row update before-images and after-images can both fit into a single page, the database server writes a single HUPDATE record. In addition, the flag field of the HUPDATE log may include the following values: LM_PREVLSN Confirms that an LSN exists. LM_FIRSTUPD Confirms that this is the first update for this rowID by this transaction.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		forward ptr rowid	Hexadecimal
		old slotlen	Decimal
		new slotlen	Decimal
		number of pieces	Decimal
		LSN (optional)	Decimal
IDXFLAGS	Index flags.	tblspace ID	Hexadecimal
		key number	Hexadecimal
INSERT	Insert after-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
ISOSPCOMMIT	Log an isolated save-point commit.	end transaction time	Decimal
		begin transaction time	Decimal
LCKLVL	Locking mode (page or row).	tblspace ID	Hexadecimal
		old lockmode	Hexadecimal
		new lockmode	Hexadecimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
LG_ADDBPOOL	Add a buffer pool online.	page size in bytes	Decimal
		number of buffers in the pool	Decimal
		number of lru queues	Decimal
		percent of lru_max_dirty	Decimal
		percent of lru_min_dirty	Decimal
PTRUNCATE	Identifies an intention to truncate a table. The partitions are marked to be dropped or reused, according to the specified command option.	tblspace ID	Hexadecimal
TRUNCATE	TRUNCATE has freed the extents and the transaction will be committed.	tblspace ID	Hexadecimal
MVIDXND	Index node moved to allow for 2-bit to 4-bit bitmap conversion.	tblspace ID	Hexadecimal
		old page number	Decimal
		new page number	Decimal
		parent page number	Decimal
		parent slot number	Decimal
		parent slot offset	Decimal
		key number	Decimal
PBDELETE	Delete tblspace blobpage.	bpageno	Hexadecimal
		status	USED/FREE
		unique ID	Decimal
PBINSERT	Insert tblspace blobpage.	bpageno	Hexadecimal
		tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
		pbrowid	Hexadecimal
PDINDEX	Predrop index.	tblspace ID	Hexadecimal
PGALTER	Page altered in place.	tblspace ID	Hexadecimal
		physical page number	Hexadecimal
PGMODE	Page mode modified in bitmap.	tblspace ID	Hexadecimal
		logical page number	Decimal
		old mode	Hexadecimal
		new mode	Hexadecimal
PERASE	Preerase old file. Mark a table that is to be dropped. The database server frees the space on the commit.	tblspace ID	Hexadecimal
PNGPALIGN8	Use the pages in this tblspace as generic pages.	None	
PNLOCKID	Change tblspaces lockid.	tblspace ID	Hexadecimal
		old lock ID	Hexadecimal
		new lock ID	Hexadecimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
PNSIZES	Set tblspace extent sizes.	tblspace ID	Hexadecimal
		fextsize	Decimal
		nextsize	Decimal
PREPARE	Written by a participant database server to record the ability of the participant to commit the transaction, if so instructed.	DBSERVERNAME of coordinator	ASCII
PTADESC	Add alter description information.	tblspace ID	Hexadecimal
		physical page number of previous page	Hexadecimal
		logical page number	Decimal
		number of columns added	Decimal
PTALTER	Alter of fragment begun.	tblspace ID	Hexadecimal
		physical page number previous page	Hexadecimal
		logical page number	Decimal
		alter desc page number	Decimal
		num columns added	Decimal
		version of alter	Decimal
		added rowsize	Decimal
PTALTNEWKEYD	Update key descriptors in a tblspace header after an alter table command.	bytes in key descriptor	Decimal
		data in key descriptor	ASCII
PTALTOLDKEYD	Update key descriptors after an alter table command.	bytes in key descriptor	Decimal
		data in key descriptor	ASCII
PTCOLUMN	Add special columns to fragment.	tblspace ID	Hexadecimal
		number of columns	Decimal
PTEXTEND	Tblspace extend.	tblspace ID	Hexadecimal
		last logical page	Decimal
		first physical page	Hexadecimal
PTRENAME	Rename table.	tblspace ID	Hexadecimal
		old table name	ASCII
		new table name	ASCII
RDELETE	Remainder page delete.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
		hrowid (optional)	Decimal
		poffset (optional)	Decimal
RENDBS	Rename dbspace.	new dbspace name	ASCII
REVERT	Logs the reversion of a database space to a database space of an earlier version.	type of reversion event	Decimal
		arg1	Decimal
		arg2	Decimal
		arg3	Decimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
RINSERT	Remainder page insert.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
		hrowid (optional)	Decimal
		poffset (optional)	Decimal
ROLLBACK	Rollback work.	date	Decimal
		time	Decimal
ROLWORK	End a transaction and roll back work.	end transaction time	Decimal
		begin transaction time	Decimal
RSVEXTEND	Logs the extension to the reserved pages.	number of pages	Decimal
		physical page number of extent	Hexadecimal
RTREE	Logs inserts and deletions for R-tree index pages. (Other operations on R-tree indexes are physically logged.) The record subtypes are: <ul style="list-style-type: none"> • LEAFINS - insert item in a leaf page • LEAFDEL - delete item from leaf page 	record subtype	ASCII
		[index page rowid	Hexadecimal
		tuple length	Decimal
		base table rowid	Decimal
		base table fragid	Decimal
		delete flag]	Decimal
RUPAFT	Remainder page update, after-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
RUPBEF	Remainder page update, before-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		slotlen	Decimal
		hrowid (optional)	Decimal
		poffset (optional)	Decimal
RUPDATE	If the remainder page update before-images and after-images can both fit into a single page, the database server writes a single RUPDATE record.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
		forward ptr rowid	Hexadecimal
		old slotlen	Decimal
		new slotlen	Decimal
		number of pieces	Decimal
		hrowid (optional)	Decimal
		poffset (optional)	Decimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
SBLOB	<p>Indicates a subsystem log record for a smart large object.</p> <p>The various record subtypes are:</p> <ul style="list-style-type: none"> CHALLOC CHCOMBINE CHFREE CHSPLIT CREATE DELETES EXTEND HDRUPD PDELETE PTRUNC REFCOUNT UDINSERT UDINSERT_LT UDUPAFT UDUPAFT_LT UDUPAFT UDUPAFT_LT UDWRITE UDWRITE_LT 	<p>Varies</p> <p>For more information, see “Log Record Types for Smart Large Objects” on page 5-14.</p>	Varies
SYNC	Written to a logical-log file if that log file is empty and administrator instructs the database server to switch to next log file.	(None)	(None)
TABLOCKS	Written by either a coordinator or a participant database server. It is associated with either a BEGPREP or a PREPARE record and contains a list of the locked tablespaces (by tablespace number) held by the transaction. (In a distributed transaction, transactions are shown as the owners of locks.)	number of locks	Decimal
		tablespace number	Hexadecimal
UDINSERT	Append new user data.	chunk	Decimal
		page within chunk	Hexadecimal
		offset within page	Hexadecimal
		data length	Hexadecimal
UDUPAFT	Update user data after-image if a UDWRITE is too expensive.	chunk	Decimal
		page within chunk	Hexadecimal
		offset within page	Hexadecimal
		data length	Hexadecimal
UDUPBEF	Update user-data before-image if a UDWRITE is too expensive.	chunk	Decimal
		page within chunk	Hexadecimal
		offset within page	Hexadecimal
		data length	Hexadecimal

Table 5-3. Logical-Log Record Types (continued)

Record Type	Action	Additional Columns	Format
UDWRITE	Update user data (difference image).	chunk	Decimal
		page within chunk	Hexadecimal
		offset within chunk	Hexadecimal
		length before write	Hexadecimal
		length after write	Hexadecimal
UNDO	Header record to a series of transactions to be rolled back.	count	Decimal
UNDOBLDC	This record is written if a CREATE TABLE statement should be rolled back but cannot be because the relevant chunk is down. When the log file is replayed, the table will be dropped.	tblspace number	Hexadecimal
UNIQID	Logged when a new serial value is assigned to a row.	tblspace ID	Hexadecimal
		unique ID	Decimal
UPDAFT	Update after-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
UPDBEF	Update before-image.	tblspace ID	Hexadecimal
		rowid	Hexadecimal
XAPREPARE	Participant can commit this XA transaction.	(None)	(None)

Log Record Types for Smart Large Objects

All smart-large-object log records are the SBLOB type. Each smart-large-object log record contains six header columns, described in “Logical-Log Record Header” on page 5-2; the record subtype; and additional information. The information that appears varies, depending on record subtype.

Table 5-4 lists all the smart-large-object record types. The **Subtype** column describes the smart-large-object record type. The **Action** column indicates the type of database server action that generated the log entry. The **Additional Columns** and **Format** columns describe what information appears for each record type.

Table 5-4. Record Subtypes for Smart Large Objects

Record Subtype	Action	Additional Columns	Format
CHALLOC	Allocate chunk extent.	extent [chk, page, len]	Decimal
		flags	Hexadecimal
CHCOMBINE	Combine two pages in the user-data extent list.	chunk number	Decimal
		first page	Decimal
		second page	Decimal
CHFREE	Frees chunk extent.	extent [chk, page, len]	Decimal
CHSPLIT	Split a page in the user-data extent list.	chunk number	Decimal
		UDFET page to split	Decimal

Table 5-4. Record Subtypes for Smart Large Objects (continued)

Record Subtype	Action	Additional Columns	Format
CREATE	Create smart large object.	smart-large-object ID [sbs, chk, page, oid]	Decimal
		number of extents in lomaphdr	Decimal
DELETE	Delete a smart large object at commit.	smart-large-object ID [sbs, chk, page, oid]	Decimal
EXTEND	Add extent to an extent list of a smart large object.	smart-large-object ID [sbs, chk, page, oid]	Decimal
		extent [chk, page, len]	Decimal
		lomaph overflow page number	Decimal
HDRUPD	Update smart-large-object header page.	smart-large-object ID [sbs, chk, page, oid]	Decimal
		old EOF offset	String
		new EOF offset	String
		old times	Decimal
		new times	Decimal
PDELETE	Queue a smart large object for deletion at commit.	smart-large-object ID [sbs, chk, page, oid]	Decimal
PTRUNC	Queue a smart large object for truncation at commit.	smart-large-object ID [sbs, chk, page, oid]	Decimal
		old offset	String
		new offset	String
REFCOUNT	Increment or decrement the reference count of a smart large object.	smart-large-object ID [sbs, chk, page, oid]	Decimal
		1 if increment; 0 if decrement	Decimal
UDINSERT,	Append new user data.	chunk	Decimal
UDINSERT_LT		page within chunk	Decimal
		offset within page	Decimal
		data length	Decimal
UDUPAFT,	Update user-data after-image if a UDWRITE is too expensive.	chunk	Decimal
UDUPAFT_LT		page within chunk	Decimal
		offset within page	Decimal
		data length	Decimal
UDUPBEF,	Update user-data beforeimage if a UDWRITE is too expensive.	chunk	Decimal
UDUPBEF_LT		page within chunk	Decimal
		offset within page	Decimal
		data length	Decimal
UDWRITE,	Update user data (difference image).	chunk	Decimal
UDWRITE_LT		page within chunk	Decimal
		offset within page	Decimal
		length before write	Decimal
		length after write	Decimal
		number of different image pieces	Decimal

For an example of smart-large-object records in **onlog** output, see smart-large-object log records in the chapter on what is the logical log in the *IBM Informix Administrator's Guide*.

Figure 5-1 shows an example of smart-large-object records in **onlog** output. The first two records show that an extent was freed. The next group of records, flanked by BEGIN and COMMIT, shows the allocation of storage and creation of the smart large objects.

addr	len	type	xid	id	link	subtype	specific-info
4e8428	40	SBLOB	8	0	4e7400	CHFREE	(2,53,421)
4e8450	40	SBLOB	8	0	4e8428	CHFREE	(2,579,421)
c8018	40	BEGIN	8	3	0	07/13/98	10:23:04 34 informix
c8040	264	SBLOB	8	0	c8018	CREATE	[2,2,1,900350517] 10
c8148	44	SBLOB	8	0	c8040	CHALLOC	(2,53,8) 0x1
c8174	68	SBLOB	8	0	c8148	EXTEND	[2,2,1,900350517] (2,53,8) -1
c81b8	264	SBLOB	8	0	c8174	CREATE	[2,2,2,900350518] 10
c82c0	44	SBLOB	8	0	c81b8	CHALLOC	(2,61,1) 0x1
c82ec	68	SBLOB	8	0	c82c0	EXTEND	[2,2,2,900350518] (2,61,1) -1
c8330	56	SBLOB	8	0	c82ec	REFCOUNT	[2,2,1,900350517] 1
c8368	56	SBLOB	8	0	c8330	REFCOUNT	[2,2,2,900350518] 1
c83a0	36	COMMIT	8	0	c8368	07/13/98	10:23:05
c83c4	40	BEGIN	8	3	0	07/13/98	10:23:05 34 informix
c83ec	264	SBLOB	8	0	c83c4	CREATE	[2,2,3,900350519] 10
c84f4	44	SBLOB	8	0	c83ec	CHALLOC	(2,62,1) 0x1
c8520	68	SBLOB	8	0	c84f4	EXTEND	[2,2,3,900350519] (2,62,1) -1
c8564	56	SBLOB	8	0	c8520	REFCOUNT	[2,2,3,900350519] 1
c859c	36	COMMIT	8	0	c8564	07/13/98	10:23:05

Figure 5-1. Smart-Large-Object Records in onlog Output

Part 2. Administrative Utilities

Chapter 6. Overview of Utilities

This chapter provides reference material for the Informix database server utilities. These utilities allow you to perform administrative tasks directly from the command line. For a complete listing of utilities, see your *IBM Informix Dynamic Server Getting Started Guide*.

You can use the following utilities:

- IBM Informix Server Administrator (ISA)
- ON-Bar
- **oncheck**
- **ondblog**
- **oninit**
- **onlog**
- **onmode**
- ON-Monitor
- **onparams**
- **onspaces**
- **onstat**
- **ontape**
- OpenAdmin Tool for IDS

The database server must be online before you execute a utility, with the following exceptions:

- **oninit**
- Some **onlog** options
- Some **oncheck** options

Note: When using utilities, do not use the UNIX command CTRL-C to send an interrupt signal to a process because it might produce an error.

Complete List of Utilities

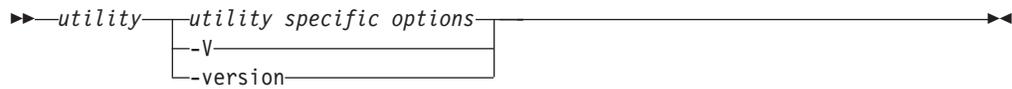
The appendix in your *IBM Informix Dynamic Server Getting Started Guide* contains a quick reference to all utilities and their options.

Obtaining Utility Version Information

Many Informix command-line utilities allow you to obtain version information using **-V** and **-version** options. You use the **-V** and **-version** options primarily for debugging. When a Technical Support representative asks for the version number, you can use the **-V** and **-version** options to find the information.

The **-V** option displays the software version number and the serial number.

The **-version** option extends the **-V** option to display additional information on the build operation system, build number, and build date.



The **-V** and **-version** options cannot be used with any other utility options. For example, the **onstat -version** command might display the following output.

```
onstat -version
```

```

Program:          onstat
Build Version:    11.50.FC1
Build Host:       connla
Build OS:         SunOS 5.6
Build Number:     009
Build Date:       Sat Aug 11 03:38:27 CDT 2008
GLS Version:      glslib-4.50.xc2
  
```

The **onstat -V** command might display the following information:

```

IBM Informix Dynamic Server Version 11.50.FC1   Software Serial Number
RDS#N0000000
  
```

Syntax of Utility-Specific Options

The following syntax diagram illustrates the **-V** and **-version** options

Multibyte Characters (GLS)

The database server utilities support multibyte command-line arguments. For a complete list of the utilities that support multibyte command-line arguments, see the *IBM Informix GLS User's Guide*.

OpenAdmin Tool for IDS

OpenAdmin Tool for IDS is a PHP-based Web browser administration tool that can administer multiple database server instances using a single installation on a Web server. You access the Web server through any browser to administer all your database servers. You can perform the following tasks with OpenAdmin:

- Add new connections to servers and server groups
- View server information on a map
- Customize the help system to add your own content to help topics
- Configure the display to change the sort order of reports by clicking on column headings
- Manage dbspaces, chunks, and recovery logs
- Monitor performance statistics, including recent SQL statements, combine graphs and reports to create custom reports
- View the online message log and the ON-Bar activity log
- Execute ad hoc or saved SQL statements
- View database, table, and column information
- Monitor high-availability clusters: HDR servers, Shared Disk secondary servers, and Remote Standalone secondary servers
- View information about executed SQL Administration API commands

OpenAdmin is an open-source program that you can download from this Web site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd.

IBM Informix Server Administrator

IBM Informix Server Administrator (ISA) allows a DBA to manage Informix database servers by executing Informix commands from any web browser. You do not need to be familiar with the syntax and format of database server commands. ISA presents the command output in an easy-to-read format.

ISA is available for download at <http://www.ibm.com/software/data/informix/downloads.html>.

With ISA, you can perform these database server administrative tasks:

- Change configuration parameters temporarily or permanently.
- Use **Server Setup** to configure or reconfigure the database server.
- Change the database server mode.
- Modify connectivity information in the **sqlhosts** file.
- Check dbspaces, blobspaces, and sbspaces.
- Manage logical logs and physical logs.
- Examine and modify memory usage.
- Read the message log.
- Back up and restore dbspaces, blobspaces, and sbspaces.
- Run various **onstat** commands to monitor performance.
- Enter SQL statements and examine database schemas.
- Add and remove chunks, dbspaces, blobspaces, sbspaces.
- Examine and manage user sessions.
- Examine and manage virtual processors (VPs).
- Use the High-Performance Loader (HPL), **dbimport**, and **dbexport**.
- Manage Enterprise Replication.
- Manage a Informix MaxConnect server.
- Set up primary and secondary database servers for High-Availability Data Replication.
- Use the following utilities: **dbaccess**, **dbschema**, **onbar**, **oncheck**, **ondblog**, **oninit**, **onlog**, **onmode**, **onparams**, **onspaces**, **onstat**, **onpladm**.
- Enter any Informix utility, UNIX shell command, or Windows command.

Server Studio

Server Studio by AGS is a CD included in the IBM Informix Dynamic Server bundle. It provides a collection of tools for DBAs and developers for performing common database tasks. Server Studio contains the following modules:

- Object Explorer
- Schema Editor
- SQL Editor

Server Studio offers several additional modules that are available on a 30-day trial basis. For more information, visit the AGS Web site at <http://www.agsltd.com>.

Chapter 7. The oncheck Utility

In This Chapter

This chapter shows you how to use the **oncheck** options and functionalities.

oncheck Check-and-Repair

The **oncheck** utility can repair the following types of disk structures:

- Partition page statistics
- Bitmap pages
- Partition blobpages
- Blobspace blobpages
- Indexes
- Sbspace pages
- Metadata partitions for sbspaces

If **oncheck** detects inconsistencies in other structures, messages alert you to these inconsistencies, but **oncheck** cannot resolve the problem. For more information, see the chapter on consistency checking in the *IBM Informix Administrator's Guide* and Chapter 4, "Disk Structures and Storage," on page 4-1

What Does Each Option Do?

As Table 7-1 shows, the **oncheck** options fall into three categories: check, repair, and display. The display or print options (those prefixed with the letter **p**) are identical in function to the **-c** options, except that the **-p** options display additional information about the data that is being checked as the **oncheck** utility executes. You cannot combine **oncheck** option flags except as the following paragraphs describe.

In general, the **-c** options check for consistency and display a message on the screen only if they find an error or inconsistency.

Any user can execute the check options. On UNIX platforms, you must be user **informix** or **root** to display database data or initiate repair options. On Windows, you must be a member of the **Informix-Admin** group to display database data or initiate repair options.

Table 7-1 associates **oncheck** options with their function. It also shows the Administrative API *command* strings that are equivalent to the **oncheck -c** options.

Table 7-1. *oncheck* Options and Their Function

Object	Check	Administrative API <i>command</i> string	Repair	Display
Blobspace simple large objects				-pB
System catalog tables	-cc			-pc

Table 7-1. *oncheck* Options and Their Function (continued)

Object	Check	Administrative API command string	Repair	Display
Data rows, no simple large objects or smart large objects	-cd			-pd
Data rows, simple large objects but no smart large objects	-cD			-pD
Table with a user-defined access method	-cd, -cD	CHECK DATA		
Chunks and extents	-ce	CHECK EXTENTS		-pe
Index (key values)	-ci, -cix		-ci -y -pk -y, -pkx -y	-pk
Index (keys plus rowids)	-cI, -cIx		-cI -y -pK -y, -pKx -y	-pK
Index with a user-defined access method	-ci, -cI			
Index (leaf key values)			-pl -y, -plx -y	-pl
Index (leaf keys plus rowids)			-pL -y, -pLx -y	-pL
Pages (by table or fragment)				-pp
Pages (by chunk)				-pP
Root reserved pages	-cr, -cR			-pr, -pR
Metadata for smart large objects	-cs, -cS			-ps, -pS
Space usage (by table or fragment)		CHECK PARTITION PRINT PARTITION		-pt
Space usage (by table, with indexes)				-pT

Using the -y Option to Perform Repairs

Use the **-y** option to instruct **oncheck** to perform repairs automatically, as the following examples show:

```
oncheck -cd -y
oncheck -cD -y
oncheck -ci -y
oncheck -cI -y
```

If you do not use the **-y** option, **oncheck** prompts you when it encounters an inconsistency and allows you to request a repair. If you specify option **-n**, **oncheck** does not prompt you because this option instructs **oncheck** to not perform repairs.

Repairing Indexes in Sbspaces and External Spaces

The **oncheck** utility can repair an index in an sbspace or external space if the index is created using an access method that supports the **oncheck -y** option. Although the **oncheck** utility does not repair fragmented indexes, user-defined access

methods can repair them. For more information about the **oncheck** options that access methods support, see the *IBM Informix DataBlade API Programmer's Guide* or the *IBM Informix Virtual-Index Interface Programmer's Guide*.

Locking and oncheck

The **oncheck** utility places a shared lock on a table during the following operations, so no other users can perform updates, inserts, or deletes until the check has completed:

- When it checks data
- When it checks indexes (with **-ci**, **-cI**, **-pk**, **-pK**, **-pl**, or **-pL**) and the table uses page locking
- When you specify the **-x** option with **-ci**, **-cI**, **-pk**, **-pK**, **-pl**, or **-pL** and the table uses row locking

If the table does not use page locking, the database server does not place a shared lock on the table when you check an index with the **oncheck -ci**, **-cI**, **-pk**, **-pK**, **-pl**, or **-pL** options. When no shared lock is on the table during an index check, other users can update rows during the check.

By not placing a shared lock on tables using row locks during index checks, the **oncheck** utility cannot be as accurate in the index check. For absolute assurance of a complete index check, you can execute **oncheck** with the **-x** option. With the **-x** option, **oncheck** places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed.

For more information about the **-x** option, refer to “Turn On Locking with **-x**” on page 7-19. For information on shared locks and intent shared locks, see the *IBM Informix Performance Guide*.

The **oncheck** utility places a shared lock on system catalog tables when they are checked. It places an exclusive lock on a table when it executes .

Oncheck Syntax

Element	Purpose	Key Considerations
-ce	Checks each chunk-free list and corresponding free space and each tblspace extent. Also checks smart-large-object extents and sbspace metadata	The oncheck process verifies that the extents on disk correspond to the current control information that describes them. See “oncheck -ce, -pe: Check the chunk-free list” on page 7-9. For background information, see “Next-Extent Allocation” on page 4-10.
-ci	Checks the ordering of key values and the consistency of horizontal and vertical node links for all indexes associated with the specified table Also checks indexes that use a user-defined access method	See “oncheck -ci and -cI: Check index node links” on page 7-10.
-cI	Same as -ci but also checks that the key value tied to a rowid in an index is the same as the key value in the row	See “oncheck -ci and -cI: Check index node links” on page 7-10.
-cr	Checks each of the root dbspace reserved pages for several conditions	See “oncheck -cr and -cR: Check reserved pages” on page 7-11.
-cR	Checks the root dbspace reserved pages, physical-log pages, and logical-log pages	See “oncheck -cr and -cR: Check reserved pages” on page 7-11
-cs	Checks smart large object and sbspace metadata for an sbspace	See “oncheck -cs, -cS, -ps, -pS: Check and display sbspaces” on page 7-12.
-cS	Checks smart large object and sbspace metadata for an sbspace as well as extents	See “oncheck -cs, -cS, -ps, -pS: Check and display sbspaces” on page 7-12.
sbspace	Indicates optional sbspace name If not supplied, all sbspaces are checked.	None.
-n	Indicates that no index repair should be performed, even if errors are detected	Use with the index repair options (-ci , -cI , -pk , -pK , -pl , and -pL).
-pB	Displays statistics that describe the average fullness of blobpage blobpages in a specified table	These statistics provide a measure of storage efficiency for individual simple large objects in a database or table. If a table or fragment is not specified, statistics are displayed for the entire database. See “oncheck -pB: Display blobpage statistics” on page 7-12. For information about optimizing blobpage size, see the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-pc	Same as -cc but also displays the system catalog information as it checks the system catalog tables, including extent use for each table	None.
-pd	Displays rows in hexadecimal format	See “oncheck -pd and pD: Display rows in hexadecimal format” on page 7-12.

Element	Purpose	Key Considerations
-pD	Displays rows in hexadecimal format and simple-large-object values stored in the tblspace or header information for smart large objects stored in an sbspace sbpage and simple large objects stored in a blobospace blobpage	See “oncheck -pd and pD: Display rows in hexadecimal format” on page 7-12.
-pe	Same as -ce but also displays the chunk and tblspace extent information as it checks the chunk free list, the corresponding free space, and each tblspace extent	See “oncheck -ce, -pe: Check the chunk-free list” on page 7-9.
-pk	Same as -ci but also displays the key values for all indexes on the specified table as it checks them	See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-14.
-pK	Same as -ci but also displays the key values and rowids as it checks them	See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-14.
-pl	Same as -ci but also displays the key values. Only leaf-node index pages are checked	See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-14.
-pL	Same as -ci but also displays the key values and rowids for leaf-node index pages only	See “oncheck -pk, -pK, -pl, -pL: Display index information” on page 7-14.
-pp	Displays contents of a logical page	See “oncheck -pp and -pP: Display the contents of a logical page” on page 7-15.
-pP	Same as -pp but requires a chunk number and logical page number or internal rowid as input	See “oncheck -pp and -pP: Display the contents of a logical page” on page 7-15.
-pr	Same as -cr but also displays the reserved-page information as it checks the reserved pages	See “oncheck -pr and pR: Display reserved-page information” on page 7-17.
-pR	Same as -cr but also displays the information for the reserved pages, physical-log pages, and logical-log pages	See “oncheck -pr and pR: Display reserved-page information” on page 7-17.
-ps	Checks and displays smart-large-object and sbspace metadata for an sbspace	See “oncheck -cs, -cS, -ps, -pS: Check and display sbspaces” on page 7-12.
-pS	Checks and displays smart-large-object and sbspace metadata. Lists extents and header information for individual smart large objects	See “oncheck -cs, -cS, -ps, -pS: Check and display sbspaces” on page 7-12.
-pt	Displays tblspace information for a table or fragment	See “oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18.
-pT	Same as -pt but also displays index-specific information and page-allocation information by page type (for dbspaces)	See “oncheck -pt and -pT: Display tblspaces for a table or fragment” on page 7-18.
-q	Suppresses all checking and validation message	None.
-x	Places a shared lock on the table when you check and print an index	Use with the -ci , -ci , -pk , -pK , -pl , or -pL options. For complete information, see “Turn On Locking with -x” on page 7-19.

Element	Purpose	Key Considerations
-y	Repairs indexes when errors are detected	None.
-V	Displays the software version number and the serial number	See “Obtaining Utility Version Information” on page 6-1.
-version	Displays the build version, host, OS, number and date, as well as the GLS version	See “Obtaining Utility Version Information” on page 6-1.
<i>chunknum</i>	Specifies a decimal value that you use to indicate a particular chunk	Value must be an unsigned integer greater than 0. Chunk must exist. Execute the -pe option to learn which chunk numbers are associated with specific dbspaces, blobspaces or sbspaces.
<i>database</i>	Specifies the name of a database that you want to check for consistency	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
db1	Specifies the local database that contains a data type that you want to check	Optionally specify the local database server name using the format db1@server1 .
db2	Specifies the remote database that contains a data type that you want to check	Optionally specify the remote database server name using the format db2@server2 .
<i>frag_dbs</i>	Specifies the name of a dbspace that contains a fragment you want to check for consistency	Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>index_name</i>	Specifies the name of the index that you want to check for consistency	Index must exist on table and in database specified. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>logical pagenum</i>	Specifies an integer value that you use to indicate a particular page in a tblspace	Value must be an unsigned integer between 0 and 16,777,215, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.
<i>object</i>	Specifies the name of the DataBlade, cast, operator class, user-defined data type, or UDR that you want to check	If you do not specify an object name, the database server compares all objects of the same type with the same name and owner.
<i>owner</i>	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Owner Name segment; for more information, see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>pagenum</i>	Indicates the page number of the sbspace metadata portion to check and display	None.
<i>partnum</i>	Identifies the sbspace metadata partition to check and display	None.
<i>rowid</i>	Identifies the rowid of the row whose contents you want to display. The rowid is displayed as part of oncheck -pD output	Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.
<i>sbspace</i>	Specifies the name of the sbspace that you want to check for consistency	None.

Element	Purpose	Key Considerations
<i>server</i>	Specifies the database server name	If you omit the database server name, oncheck uses the name that INFORMIXSERVER specifies.
<i>table</i>	Specifies the name of the table that you want to check for consistency	Table exists when you execute the utility. Syntax must conform to the Table Name segment; for more information, see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>tblspacenum</i>	Identifies the tblspace whose contents you want to display	Value must be an unsigned integer between 0 and 208,666,624, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.

oncheck -cc and -pc: Check system catalog tables

Syntax:

```

▶▶ oncheck [-cc] [-pc] database

```

The **-cc** option checks all system catalog tables for the specified database. If you do not specify a database, it checks all system catalog tables for all databases.

The **-pc** option performs the same checks on system catalog tables and also displays the system catalog information, including the physical address, type of locking used, row size, number of keys, extent use, the number of pages allocated and used, tblspace partnum, and index use for each table.

Before you execute **oncheck -cc** or **oncheck -pc**, execute the SQL statement UPDATE STATISTICS to ensure that an accurate check occurs. To check a table, **oncheck** compares each system catalog table to its corresponding entry in the tblspace. For more information about the tblspace, see “Structure of the Tblspace Tblspace” on page 4-4.)

oncheck -cd and -cD: Check pages

Syntax:

```

▶▶ oncheck [-cd] [-cD] database [:owner.] table [,frag_dbs] [,frag_part]

```

The **-cd** option reads all pages, excluding blobpages and sbpages, from the tblspace for the specified database, table, fragment, or multiple fragments (fragparts), and checks each page for consistency. It checks entries in the bitmap page against the pages to verify mapping.

The **-cD** option performs the same checks as **oncheck -cd**, and also checks the header of each blobpage for consistency. The **oncheck -cD** option does not compare the beginning time stamps stored in the header with ending time stamps stored at the end of a blobpage. Use the **oncheck -cD -y** option to clean up orphaned simple large objects in blobspaces, which may occur after a rollback across several log files.

If the database contains fragmented tables, but no fragment is specified, **oncheck -cd** checks all fragments in the table. If you do not specify a table, it checks all tables in the database. By comparison, the **-pd** option displays a hexadecimal dump of specified pages but does not check for consistency.

For both the **-cd** and **-cD** options, the **oncheck** utility locks each table as it checks the indexes for the table. To repair the pages, specify **oncheck -cd -y** or **oncheck -cD -y**.

If tables are fragmented on multiple partitions in the same dbspace, the **oncheck -cd** and **oncheck -cD** commands show the partition names. The following example shows typical output for a table that has fragments in multiple partitions in the same dbspace:

```
TBLspace data check for multipart:informix.t1
      Table fragment partition part_1 in DBspace dbs1
      Table fragment partition part_2 in DBspace dbs1
      Table fragment partition part_3 in DBspace dbs1
      Table fragment partition part_4 in DBspace dbs1
      Table fragment partition part_5 in DBspace dbs1
```

The following example checks the data rows, including simple large objects and smart large objects, in the **catalog** table:

```
oncheck -cD superstores_demo:catalog
```

If **oncheck** finds an inconsistency, it displays a message similar to the following one:

```
BAD PAGE 2:28: pg_addr 2:28 != bp-> bf_pagenum 2:69
```

The physical address 2:28 represents page 28 of chunk number 2. If **oncheck** finds no inconsistencies, it displays a header similar to the following one for each table that it checks:

```
TBLSPACE data check for stores_demo:informix.customer
```

If you specify a single fragment, **oncheck** displays a single header for that fragment. The **oncheck** utility displays a header similar to the following one for fragmented tables, one per fragment:

```
TBLspace data check for stores_demo:informix.tab1
      Table fragment in DBspace db1
```

If an index that uses an access method provided by a DataBlade module cannot find the access method, you receive the following message:

```
-9845 Access method access_method_name does not exist in database.
Ensure that the DataBlade installation was successful.
```

To monitor blob space blob pages, refer to the **oncheck -pB** information at “oncheck -pB: Display blob space statistics” on page 7-12).

Use CHECK DATA as the Administrative API *command* string for **oncheck -cd**.

oncheck -ce, -pe: Check the chunk-free list

Syntax:

```
▶▶ oncheck [ -ce ] [ -pe ] ▶▶
```

The **-ce** option checks each chunk-free list and corresponding free space and each tblspace extent. For more information, refer to “Next-Extent Allocation” on page 4-10 and “Structure of the Chunk Free-List Page” on page 4-3, respectively. The **oncheck** process verifies that the extents on disk correspond to the current control information that describes them.

The **-pe** option performs the same checks and also displays the chunk and tblspace extent information during the check. The **-ce** and **-pe** options also check blobspaces, smart-large-object extents, and user-data and metadata information in sbspace chunks.

For information about using **oncheck -ce** and **-pe**, see managing disk space in the *IBM Informix Administrator's Guide*.

Use CHECK EXTENTS as the Administrative API *command* string for **oncheck -ce**.

oncheck -ci and -cI: Check index node links

Syntax:

▶▶ oncheck [-ci] [-cI] ▶▶

The **-ci** option checks the ordering of key values and the consistency of horizontal and vertical node links for all indexes associated with the specified table. For more information about indexes, see “Structure of B-Tree Index Pages” on page 4-16.)

The **-cI** option performs the same checks as **-ci** and it also checks that the key value tied to a rowid in an index is the same as the key value in the row. The **-cI** option does not cross-check data on a functional index.

If you do not specify an index, the option checks all indexes. If you do not specify a table, the option checks all tables in the database.

The same **-ci** repair options are available with **-cI**. If **oncheck -ci** or **oncheck -cI** detects inconsistencies, it prompts you for confirmation to repair the problem index. If you specify the **-y** (yes) option, indexes are automatically repaired. If you specify the **-n** (no) option, the problem is reported but not repaired; no prompting occurs.

If **oncheck** does not find inconsistencies, the following message appears:
validating indexes.....

The message displays the names of the indexes that **oncheck** is checking.

Note: Using **oncheck** to rebuild indexes can be time consuming. Processing is usually faster if you use the SQL statements DROP INDEX and CREATE INDEX to drop and re-create the index.

The following example checks all indexes on the **customer** table:

```
oncheck -cI -n stores_demo:customer
```

The following example checks the index **zip_ix** on the **customer** table:

```
oncheck -cI -n stores_demo:customer#zip_ix
```

If indexes are fragmented on multiple partitions in the same dbspace, the **oncheck -ci** and **oncheck -cI** commands show the partition names. The following example show typical output for an index that has fragments in multiple partitions in the same dbspace:

```
Validating indexes for multipart:informix.t1...
Index idx_t1
Index  fragment partition part_1 in DBspace dbs1
Index  fragment partition part_2 in DBspace dbs1
Index  fragment partition part_3 in DBspace dbs1
Index  fragment partition part_4 in DBspace dbs1
Index  fragment partition part_5 in DBspace dbs1
```

By default, the database server does not place a shared lock on the table when you check an index with the **oncheck -ci** or **-cI** options unless the table uses page locking. For absolute assurance of a complete index check, you can execute **oncheck -ci** or **oncheck -cI** with the **-x** option. With the **-x** option, **oncheck** places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed. For more information on using the **-x** option, “Turn On Locking with **-x**” on page 7-19.

When you execute **oncheck** on an external index, the user-defined access method is responsible for checking and repairing an index. If an index that employs a user-defined access method cannot find the access method, the database server reports an error. The **oncheck** utility does not repair inconsistencies in external indexes. You should not use **oncheck -cI** on a table that contains more than one type of index.

Important: If you are using the Verity Text Search DataBlade Module, the **-cI** option performs an index merge instead of the usual operations.

oncheck -cr and -cR: Check reserved pages

Syntax:

```
▶▶ oncheck [ -cr ] [ -cR ] ▶▶
```

The **-cr** option checks each of the root dbspace reserved pages as follows:

- It validates the contents of the ONCONFIG file with the PAGE_CONFIG reserved page.
- It ensures that all chunks can be opened, that chunks do not overlap, and that chunk sizes are correct.

The **-cR** option performs the same checking and validation, and also checks all logical-log and physical-log pages for consistency. The **-cr** option is considerably faster because it does not check the log-file pages.

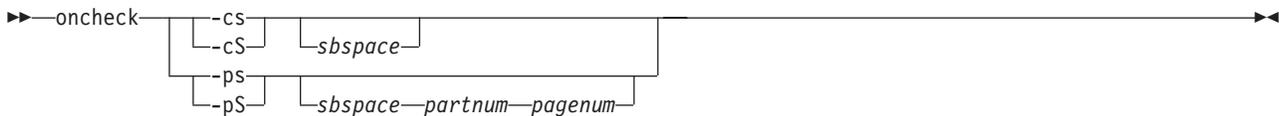
If you have changed the value of a configuration parameter (either through ISA, **onparams**, **onmonitor**, **onspaces**, or by editing the configuration file), but you have not yet reinitialized shared memory, **oncheck -cr** and **oncheck -cR** detect the inconsistency and return an error message.

If **oncheck -cr** does not display any error messages after you execute it, you can assume that all three items in the preceding list were checked successfully.

For more information on reserved pages, see “Reserved Pages” on page 4-2.

oncheck -cs, -cS, -ps, -pS: Check and display sbspaces

Syntax:



The **-cs** option checks sbspaces. The **-ps** option checks sbspaces and extents.

The **-cS** option validates and displays metadata for an sbspace.

The **-ps** option checks sbspaces and extents. If you do not specify the sbspace name, these options check all sbspaces.

The **-pS** option validates and displays metadata for an sbspace and also lists extents and header information for smart large objects.

If you do not specify the sbspace name, all sbspaces will be checked. The following example checks and displays metadata for **test_sbspace**:

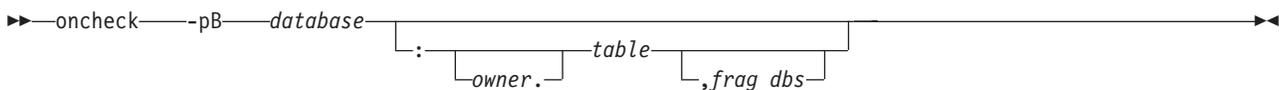
```
oncheck -ps test_sbspace
```

If you specify **rootdbs** as the sbspace name with the **-cs** or **-ps** options, **oncheck** checks the root dbspace.

For more information about using the **-cs**, **-cS**, **-ps**, and **-pS** options, see the *IBM Informix Administrator's Guide*.

oncheck -pB: Display blobspace statistics

Syntax:

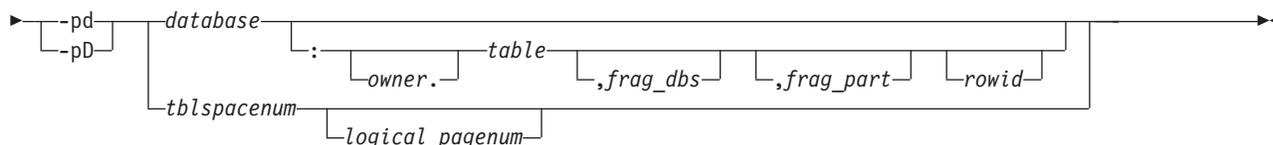


The **-pB** option displays statistics that describe the average fullness of blobpage blobpages in a specified table. These statistics provide a measure of storage efficiency for individual simple large objects in a database or table. If you do not specify a table or fragment, the option displays statistics for the entire database. For more information, see optimizing blobpage blobpage size in the chapter on managing disk space in the *IBM Informix Administrator's Guide*.

oncheck -pd and pD: Display rows in hexadecimal format

Syntax:





The **-pd** option takes a database, a table, a fragment, a fragment partition (fragpart), and a specific rowid or tblspace number and logical page number as input. In every case, **-pd** prints page-header information and displays the specified rows for the database object (database, table, fragment, internal rowid, or page number) that you specify in hexadecimal and ASCII format. No checks for consistency are performed.

Element	Purpose	Key Considerations
<i>database</i>	Specifies the name of a database that you want to check for consistency	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>frag_dbs</i>	Specifies the name of a dbspace that contains a fragment you want to check for consistency	Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>frag_part</i>	Specifies the fragment partition	For fragmented tables or an index that use expression-based or round-robin distribution schemes, you can create multiple partitions, which are collections of pages for a table or index, within a single dbspace. This partition is referred to as a <i>fragment partition</i> or <i>fragpart</i> .
<i>logical pagenum</i>	Specifies an integer value that you use to indicate a particular page in a tblspace	Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier. Value must be an unsigned integer between 0 and 16,777,215, inclusive.
<i>owner</i>	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>rowid</i>	Identifies the rowid of the row whose contents you want to display. The rowid is displayed as part of oncheck -pD output	Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.
<i>table</i>	Specifies the name of the table that you want to check for consistency	Table exists when you execute the utility. Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>tblspaceum</i>	Identifies the tblspace whose contents you want to display	Value must be an unsigned integer between 0 and 208,666,624, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.

If you specify an internal rowid (expressed as a hexadecimal value), the rowid maps to a particular page, and all rows from that page are printed.

If you specify a logical page number (expressed as a decimal), all the rows of the tblspace number with the logical page number are printed.

Element	Purpose	Key Considerations
<i>index_name</i>	Specifies the name of the index that you want to check for consistency	Index must exist on table and in database specified. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>owner</i>	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>table</i>	Specifies the name of the table that you want to check for consistency	Table exists when you execute the utility. Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
-x	Places a shared lock on the table when you check and print an index	For complete information, see "Turn On Locking with -x" on page 7-19.

If any of the **oncheck** options detect inconsistencies, you are prompted for confirmation to repair the problem index. If you specify the **-y** (yes) option, indexes are automatically repaired. If you specify the **-n** (no) option, the problem is reported but not repaired; no prompting occurs.

The following example displays information about all indexes on the **customer** table:

```
oncheck -pl -n stores_demo:customer
```

The following example displays information about the index **zip_ix**, which was created on the **customer** table:

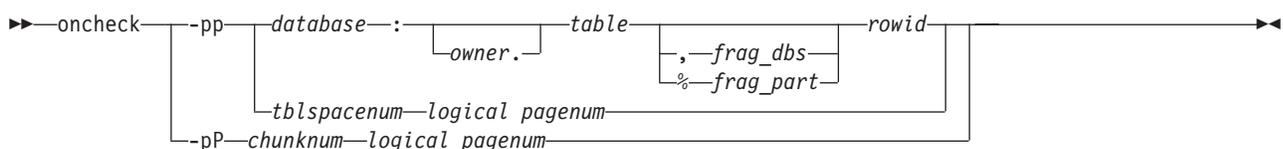
```
oncheck -pl -n stores_demo:customer#zip_ix
```

By default, the database server does not place a shared lock on the table when you check an index with the **oncheck -pk**, **-pK**, **-pl**, or **-pL** options unless the table uses page locking. For absolute assurance of a complete index check, you can execute **oncheck -pk**, **oncheck -pK**, **oncheck -pl**, or **oncheck -pL** with the **-x** option. With the **-x** option, **oncheck** places a shared lock on the table, and no other users can perform updates, inserts, or deletes until the check has completed. For more information on using the **-x** option, "Turn On Locking with -x" on page 7-19.

For more information on **oncheck -ci**, see "oncheck -ci and -cI: Check index node links" on page 7-10. For more information index pages, see "Structure of B-Tree Index Pages" on page 4-16.

oncheck -pp and -pP: Display the contents of a logical page

Syntax:



Element	Purpose	Key Considerations
<i>database</i>	Specifies the name of a database that you want to check for consistency	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>chunknum</i>	Specifies a decimal value that you use to indicate a particular chunk	Value must be an unsigned integer greater than 0. Chunk must exist.
<i>frag_dbs</i>	Specifies the name of a dbspace that contains a fragment you want to check for consistency	Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>frag_part</i>	Specifies the partition name of the fragment to be checked. This is useful in cases where more than one fragment of a table was created in the same dbspace.	For fragmented tables or an index that use expression-based or round-robin distribution schemes, you can create multiple partitions, which are collections of pages for a table or index, within a single dbspace. This partition is referred to as a <i>fragment partition</i> or <i>fragpart</i> .
<i>logical pagenum</i>	Specifies an integer value that you use to indicate a particular page in a tblspace	Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier. Value must be an unsigned integer between 0 and 16,777,215, inclusive.
<i>owner</i>	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>rowid</i>	Identifies the rowid of the row whose contents you want to display. The rowid is displayed as part of oncheck -pD output	Value must be an unsigned integer between 0 and 4,277,659,295, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.
<i>table</i>	Specifies the name of the table that you want to check for consistency	Table exists when you execute the utility. Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>tblspacenum</i>	Identifies the tblspace whose contents you want to display	Value must be an unsigned integer between 0 and 208,666,624, inclusive. Value can be expressed as an unsigned integer or hexadecimal that begins with 0x identifier.

The **-pp** option has the following syntax variations:

Invocation	Explanation
oncheck -pp tblspc lpn <pages>	Displays the contents of a logical page using a tblspace number and logical page number. You can also specify an optional parameter specifying the number of pages to be printed.
oncheck -pp tblspc lpn -h	Displays only the header of a logical page using a tblspace number and logical page number.
oncheck -pp database:table rowid	Displays the contents of a logical page using a database name, table name, and an Informix internal rowid. You can obtain this internal rowid with the oncheck -pD command. This internal rowid is not the serial rowid that is assigned in tables created with the CREATE TABLE tablename WITH ROWIDS statement. For more information, see "Definition of Rowid" on page 4-13

The page contents appear in ASCII format. The display also includes the number of slot-table entries on the page. The following example shows different invocations of the **oncheck -pp** command:

```
oncheck -pp stores_demo:orders 0x211 # database:owner.table, # fragment rowid
oncheck -pp stores_demo:informix.customer,frag_dbspcel 0x211
oncheck -pp 0x100000a 25 # specify the tblspace number and # logical page number
```

The **-pP** option provides the following syntax variations:

Invocation	Explanation
oncheck -pP chunk# offset pages	Displays the contents of a logical page using a chunk number and an offset. You can also specify an optional parameter specifying the number of pages to be printed.
oncheck -pP chunk# offset -h	Displays only the header of a logical page using a chunk number and an offset.

Note: The output for chunk page displays both the start and the length fields in decimal format.

The following example shows typical output using the **onstat -pP** command:

```
oncheck -pP 1 5 2
addr      stamp      nslots      flag      type      frptr      frcnt      next      prev
stamp    100005      250181      2         1000      ROOTRSV    320        1716     0
0        250181      slot      ptr      len      flg
...
addr      stamp      nslots      flag      type      frptr      frcnt      next      prev
stamp    100005      6          250182    2         1000      ROOTRSV    128       1908     0
250182   slot      ptr      len      flg      1         24         56         0
2        80        48       0
```

oncheck -pr and pR: Display reserved-page information

Syntax:

```
▶▶ oncheck [-pr] [ -pR ] ▶▶
```

The **-pr** option performs the same checks as **oncheck -cr** and displays the reserved-page information.

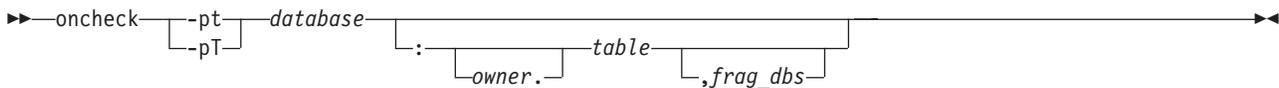
The **-pR** option performs the same checks as **oncheck -cR**, displays the reserved-page information, and displays detailed information about logical-log and physical-log pages, marking the start and end of the active physical-log pages.

If you have changed the value of a configuration parameter (either through ISA or by editing the configuration file), but you have not yet reinitialized shared memory, **oncheck -pr** and **oncheck -pR** detect the inconsistency and return an error message.

For a listing and explanation of **oncheck -pr** output, see “Reserved Pages” on page 4-2. For a description of the **-cr** option, see “oncheck -cr and -cR: Check reserved pages” on page 7-11.

oncheck -pt and -pT: Display tblspaces for a table or fragment

Syntax:



The **-pt** option prints a tblspace report for a given table or fragment whose name and database you specify when you execute **oncheck** at the command line. The report contains general allocation information including the maximum row size, the number of keys, the number of extents, their sizes, the pages allocated and used per extent, the current serial value, and the date that the table was created. The **-pt** output prints the pagesize of the tblspace, the number of pages (allocated, used and data) in terms of logical pages. The **Extents** fields list the physical address for the tblspace **tblspace** entry for the table and the address of the first page of the first extent. The extent list shows the number of logical as well as physical pages in every extent. If you do not specify a table, the option displays this information for all tables in the database.

The **-pT** option prints the same information as the **-pt** option. In addition, the **-pT** option displays index-specific information and page-allocation information by page type (for dbspaces).

Element	Purpose	Key Considerations
<i>database</i>	Specifies the name of a database that you want to check for consistency	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>frag_dbs</i>	Specifies the name of a dbspace that contains a fragment you want to check for consistency	Dbspace must exist and contain the fragment that you want to check for consistency. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>owner</i>	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Owner Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
<i>table</i>	Specifies the name of the table that you want to check for consistency	Table exists when you execute the utility. Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .

Output for both **-pt** and **-pT** contains listings for **Number of pages used**. The value shown in the output for this field is never decremented because the disk space allocated to a tblspace as part of an extent remains dedicated to that extent even after you free space by deleting rows. For an accurate count of the number of pages currently used, refer to the detailed information on tblspace use (organized by page type) that the **-pT** option provides.

The following example shows an example of output of the **oncheck -pt** command:

TBLspace Report for testdb:tab1

```

Physical Address      2:10
Creation date        10/07/2004 17:01:16
TBLspace Flags       801      Page Locking
                                TBLspace use 4 bit bit-maps
  
```

```

Maximum row size          14
Number of special columns 0
Number of keys            0
Number of extents        1
Current serial value      1
Pagesize (k)             4
First extent size        4
Next extent size         4
Number of pages allocated 340
Number of pages used     337
Number of data pages     336
Number of rows           75806
Partition partnum        2097154
Partition lockid         2097154

```

```

Extents
      Logical Page      Physical Page      Size Physical Pages
              0              2:106          340          680

```

For more examples of using **oncheck -pt** and **-pT**, see managing disk space in the *IBM Informix Administrator's Guide* and the *IBM Informix Performance Guide*.

Turn On Locking with -x

The **-x** option can be appended to the **-ci**, **-cl**, **-pk**, **-pK**, **-pl**, and **-pL** options to place a shared lock on affected tables. While the table is locked, no other users can perform inserts, updates, and deletions while **oncheck** checks or prints the index. Without the **-x** option for tables with row locking, **oncheck** only places an IS (intent shared) lock on the table, which prevents actions such as dropping the table or the indexes during the check.

For example, the following sample command instructs **oncheck** to lock indexes for the **customer** table while it validates the order of key values, validates horizontal links, and ensures that no node appears twice in the index:

```
oncheck -cix stores_demo:customer
```

When you specify option **-x**, **oncheck** locks indexes for tables that use row locking. If **oncheck** detects page-lock mode, it displays a warning message and places a shared lock on the table regardless.

Send Special Arguments to the Access Method with -u

You can use the **-u** option to send special arguments to the access method. The possible arguments depend on the access method. For example, the R-tree access method supports the **display** option, as the following example shows:

```
oncheck -pl -u "display"
```

Use commas to separate multiple arguments in the argument string.

For information on valid arguments for your access method, refer to the user manual for your access method.

Return Codes on Exit

The **oncheck** utility returns the following codes on exit.

GLS failures:-1
Invalid serial/key:2
Onconfig access error:2
Invalid onconfig settings:2
Invalid arguments to oncheck:2
Error connecting database server:1
error detected by oncheck:2
no errors detected by oncheck:0

Windows only:

Not properly installed:1
Authentication error:2

Chapter 8. The ondblog Utility

In This Chapter

This chapter shows you how to use the ondblog utility.

ondblog: Change Logging Mode

The **ondblog** utility lets you change the logging mode for one or more databases. Alternatively, you can change the logging mode by using the **ALTER LOGMODE** Administrative API *command* string. For information on using Administrative API commands, see *IBM Informix Guide to SQL: Syntax*.

The **ondblog** utility logs its output in the BAR_ACT_LOG file.

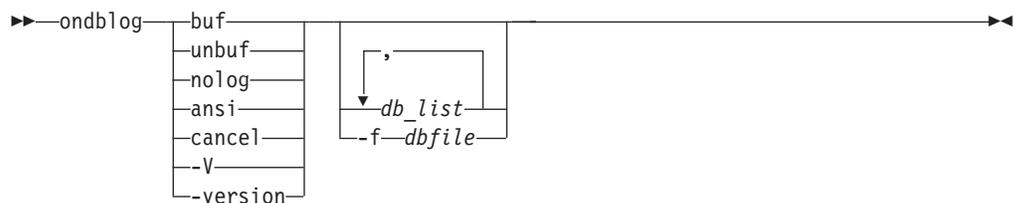
If you turn on transaction logging for a database, you must create a level-0 backup of all of the storage spaces that contain data in the database before the change takes effect.

For more information and examples of logging modes, see the following topics in the chapter on managing database-logging status in the *IBM Informix Administrator's Guide*:

- Modifying the database-logging status
- Modifying table-logging status

You cannot use the ondblog utility on High-Availability Data Replication (HDR) secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

ondblog Syntax



Element	Purpose	Key Considerations
buf	Sets the logging mode so that transaction information is written to a buffer before it is written to a logical log	None.
unbuf	Sets the logging mode so that data is not written to a buffer before it is written to a logical log	None.
nolog	Sets the logging mode so that no database transactions are logged	None.
ansi	Changes database logging to be ANSI compliant	Once you create or convert a database to ANSI mode, you cannot change it back to any of the other logging modes.

Element	Purpose	Key Considerations
cancel	Cancels the logging-mode change request before the next level-0 backup occurs	None.
-f <i>dbfile</i>	Changes the logging status of the databases that are listed (one per line) in the text file whose pathname is given by <i>dbfile</i>	This command is useful if the list of databases is long or used often.
<i>db_list</i>	Names a space-delimited list of databases whose logging status is to be changed	If you do not specify anything, all databases that the database server manages are modified.

Chapter 9. The oninit Utility

oninit: Initialize the Database Server

Run the **oninit** utility from the command line to initialize database server shared memory and to bring the database server online. If you use the **oninit -i** option, you can also initialize disk space.

Before you initialize the database server, set the **INFORMIXSERVER** environment variable to the database server name that you chose when you set the configuration parameter **DBSERVERNAME**. The **INFORMIXSERVER** environment variable is not required for initialization. However, if the **INFORMIXSERVER** environment variable is not set, the database server does not build the **sysmaster** tables. Also, the DB–Access utility requires the **INFORMIXSERVER** environment variable to be set.

Prerequisite:

On UNIX, you must be logged in as user **root** or **informix** to run **oninit**. User **informix** should be the only member of the group **informix**.

On Windows, IDS runs as a Windows service. Users with permissions can start and stop IDS through one of these methods:

- Control Panel Administrative Tools
- `net start` and `net stop` commands

Use the **starts** utility to pass command line arguments to **oninit**. For example, to start IDS in single-user mode use the following command:

```
%INFORMIXDIR%\bin\starts %INFORMIXSERVER% -j
```

For information about what happens during initialization, see the chapter on initializing the database server in the *IBM Informix Administrator's Guide*.

Initializing the Server in Administrative Mode with the -j Option

To initialize the database server in administration mode, use the **oninit -j** option. This is an administrator-only mode you can use to perform maintenance operations including those that require running SQL or DDL commands. You can use the **-j** flag with other **oninit** flags, except the **-s** flag. When in administration mode, the system only accepts connection requests from the **informix** user. The server makes an entry in the online log whenever it enters or exits administration mode.

The **oninit** utility does not have a functionally equivalent Administrative API *command* string. You cannot initialize the database server through an Administrative API *command* string.

Initializing the Server in Wait Mode with the -w Option

You can use the **oninit -w** command in customized startup scripts and to automate startup. The **-w** flag forces the server to wait until it successfully initializes before it returns to the shell prompt with a return code of 0. If the server fails to initialize within the timeout, the server returns a 1 to the shell prompt and writes a warning message in the `online.log`.

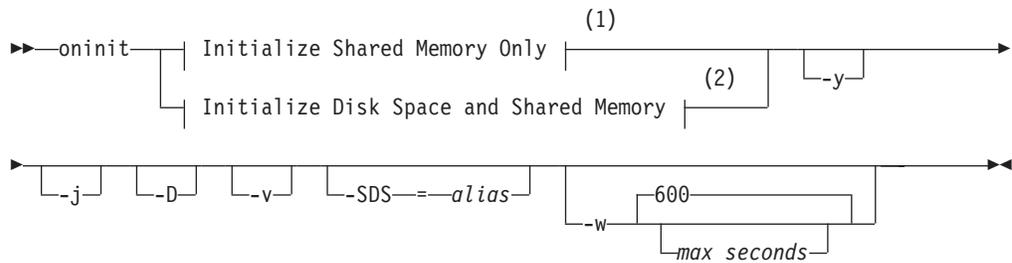
The default timeout is 600 seconds (10 minutes), which you can modify to any integer value.

In this example, if the server fails to initialize within 60 seconds, it returns a code of 1 to the prompt:

```
oninit -w 60
```

To determine the reason for the server failing to initialize, check the `online.log`. You can also try increasing the timeout value. When using **oninit -w** in a script, check the mode of the server with the **onstat - (Print output header)** command.

oninit Syntax



Notes:

- 1 see “Initialize Shared Memory Only” on page 9-3
- 2 see “Initialize Disk Space and Shared Memory” on page 9-4

Element	Purpose	Key Considerations
-y	Causes the database server to automatically respond yes to all prompts.	The -y flag bypasses the verification prompts.
-j	Initializes the server in administration mode.	The -j flag can be combined with other oninit flags, except the quiescent mode (-s) flag.
-D	Prevents both Enterprise Replication and HDR from initializing on the server instance.	None.
-v	Causes the server to print additional messages to standard output at the time of server initialization.	The -v flag can be combined with the -i flag.
-SDS=alias	Defines the SDS primary server alias.	When both the primary server and all of the SDS servers are down, the -SDS=alias flag is used to bring up the designated SDS server as the primary server. The -SDS=alias flag cannot be combined with the -i flag.

The following sequence of commands shuts down and restarts the database server in quiescent mode:

```
onmode -ky
oninit -s
```

Initialize Disk Space and Shared Memory

Warning: When you initialize disk space, the initialization destroys all data that your database server currently manages.

The database server must be offline when you initialize disk space.

Syntax:

```
oninit [-i] [-s]
```

Element	Purpose	Key Considerations
-i	Causes the database server to initialize disk space and shared memory. Leaves the database server in online mode after it initializes disk space.	None.
-s	When used with -i , causes the database server to be left in quiescent mode after disk initialization.	None.

When Dynamic Server 10.0 or later is first initialized with the **oninit -iyv** command, by default it comes online with large chunk mode fully enabled. Reversion is not possible. For more information about allowing large chunk mode, see “onmode -BC: Allow large chunk mode” on page 11-3.

Chapter 10. The onlog Utility

onlog: Display Logical-Log Contents

The **onlog** utility displays the contents of a logical-log file, either on disk or on backup.

The **onlog** output is useful in debugging situations when you want to track a specific transaction or see what changes have been made to a specific tblspace. (For information about interpreting the logical-log file contents, see Chapter 5, “Interpreting Logical-Log Records,” on page 5-1)

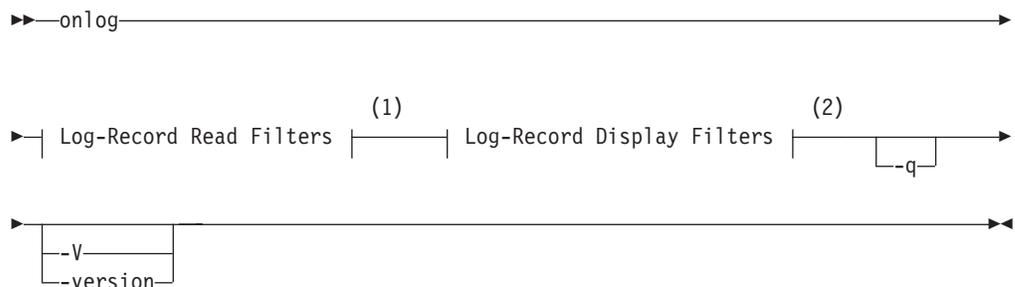
Any user can run all of the **onlog** options except the **-l** option. Only user **informix** on UNIX or a member of the **Informix-Admin** group on Windows can run the **-l** option.

If the database server is in offline mode when you execute **onlog**, only the files on disk are read. If the database server is in quiescent or online mode, **onlog** also reads the logical-log records stored in the logical-log buffers in shared memory (after all records on disk have been read).

When the database server reads a logical-log file with status U from disk while in online mode, the database server denies all access to the logical-log files, effectively stopping database activity for all sessions. (For more information, see “onstat -l: Print physical and logical log information” on page 15-142.) For this reason, it is recommended that you wait until the files have been backed up and then read the contents of the logical-log files from backup.

Note: The **onlog** utility does not have a functionally equivalent Administrative API *command* string.

onlog Syntax



Notes:

- 1 see “Log-Record Read Filters” on page 10-2
- 2 see Chapter 10, “The onlog Utility”

Element	Purpose	Key Considerations
-q	Suppresses the initial header and the one-line header that appears every 18 records by default	None.
-V	Displays the software version number and the serial number	See "Obtaining Utility Version Information" on page 6-1.
-version	Displays the build version, host, OS, number and date, as well as the GLS version	See "Obtaining Utility Version Information" on page 6-1.

You direct **onlog** to read the following portions of the logical log as it searches for records to display:

- Records stored on disk
- Records stored on backup media
- Records from the specified logical-log file

By default, **onlog** displays the logical-log record header, which describes the transaction number and the record type. The record type identifies the type of operation performed.

In addition to the header, you can use the read filters to direct **onlog** to display the following information:

- Logical-log record header and data (including copies of simple large objects stored in a dbspace or tblspace)
- Copies of blobpages from blobspaces
They are copied from the logical-log backup only. They are not available from disk.

You can display every logical-log record header, or you can specify output based on the following criteria:

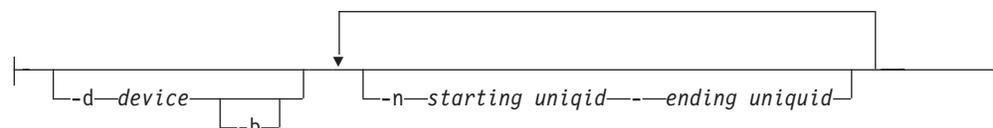
- Records associated with a specific table
- Records initiated by a specific user
- Records associated with a specific transaction

If **onlog** detects an error in the log file, such as an unrecognizable log type, it displays the entire log page in hexadecimal format and terminates.

Log-Record Read Filters

The **onlog** utility uses the pathnames that are stored in the root dbspace reserved pages to locate the logical-log files. If you use ON-Bar to back up the logical logs, **onlog** asks the storage manager to retrieve the appropriate logical-log records from the backup media.

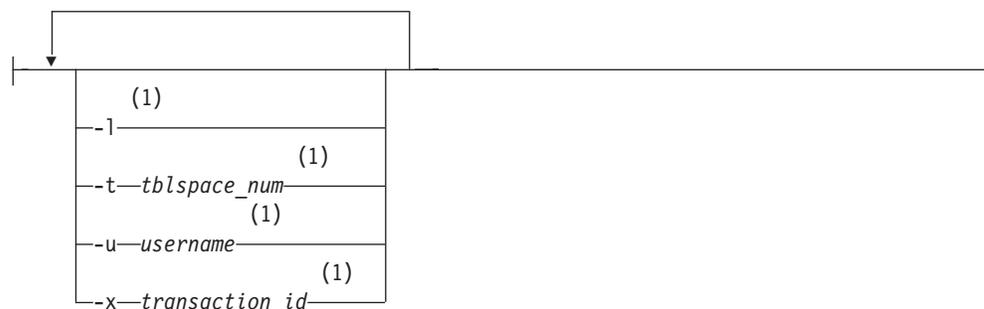
Syntax:



Element	Purpose	Key Considerations
-b	Displays logical-log records associated with blobspace blobpages	The database server stores these records on the logical-log backup media as part of blobspace logging.
-d device	Names the pathname of the storage device where the desired logical-log backup is mounted	<p>If you use ontape, the device that you name must be the same as the pathname of the device assigned to the configuration parameter LTAPEDEV. If the -d option is not used, onlog reads the logical-log files stored on disk, starting with the logical-log file with the lowest <i>logid</i>.</p> <p>If you use ON-Bar to back up logical logs, use the onbar -P command to view the contents of a logical-log file. See the <i>IBM Informix Backup and Restore Guide</i>.</p> <p>For pathname syntax, see your operating-system documentation.</p>
-n starting uniqid-ending uniqid	Directs onlog to read all the logical-log records contained in the log file that you specified from <i>starting uniqid</i> to the <i>ending uniqid</i> .	<p>The <i>starting uniqid</i> and the <i>ending uniqid</i> are the unique ID numbers of the logical log. To determine the <i>uniqid</i> of a particular logical-log file, use the onstat -l command.</p> <p>If you do not use the -n option, onlog reads all the logical-log files that are available (either on disk or on tape).</p> <p>For information about the onstat utility, see “Monitor the Database Server Status” on page 15-2.</p>

Log-Record Display Filters

Syntax::



Notes:

- 1 Only one occurrence of this item allowed

Element	Purpose	Key Considerations
-l	Displays the long listing of the logical-log record.	The long listing of a log record includes a complex hexadecimal and ASCII dump of the entire log record. The listing is not intended for casual use.
-t tblspace_num	Displays records associated with the tblspace that you specify.	<p>Unsigned integer. Number, greater than 0, must be in the partnum column of the systables system catalog table.</p> <p>Specify this value as either an integer or hexadecimal value. (If you do not use a 0x prefix, the value is interpreted as an integer.) To determine the tblspace number of a particular tblspace, query the systables system catalog table as described in “Tblspace Numbers” on page 4-5.</p>

Element	Purpose	Key Considerations
-u <i>username</i>	Displays records for a specific user.	User name must be an existing login name. User name must conform to operating-system-specific rules for login name.
-x <i>transaction_id</i>	Displays only records associated with the transaction that you specify.	Value must be an unsigned integer between 0 and TRANSACTIONS - 1, inclusive. You should need to use the -x option only in the unlikely case that an error is generated during a rollforward. When this situation occurs, the database server sends a message to the message log that includes the transaction ID of the offending transaction. You can use this transaction ID with the -x option of onlog to investigate the cause of the error.

If you do not specify any options, **onlog** displays a short listing of all the records in the log. You can combine options with any other options to produce more selective filters. For example, if you use both the **-u** and **-x** options, **onlog** displays only the activities that the specified user initiated during the specified transaction. If you use both the **-u** and **-t** options, **onlog** displays only the activities initiated by the specified user and associated with the specified tblspace.

Chapter 11. The onmode Utility

In This Chapter

This chapter shows how to use the **onmode** options. If you do not use any options, the database server returns a usage statement.

On UNIX, you must be user **root** or user **informix** to execute **onmode**. On Windows, you must be a member of the **Informix-Admin** group.

The following **onmode** options have equivalent Administrative API *command* strings:

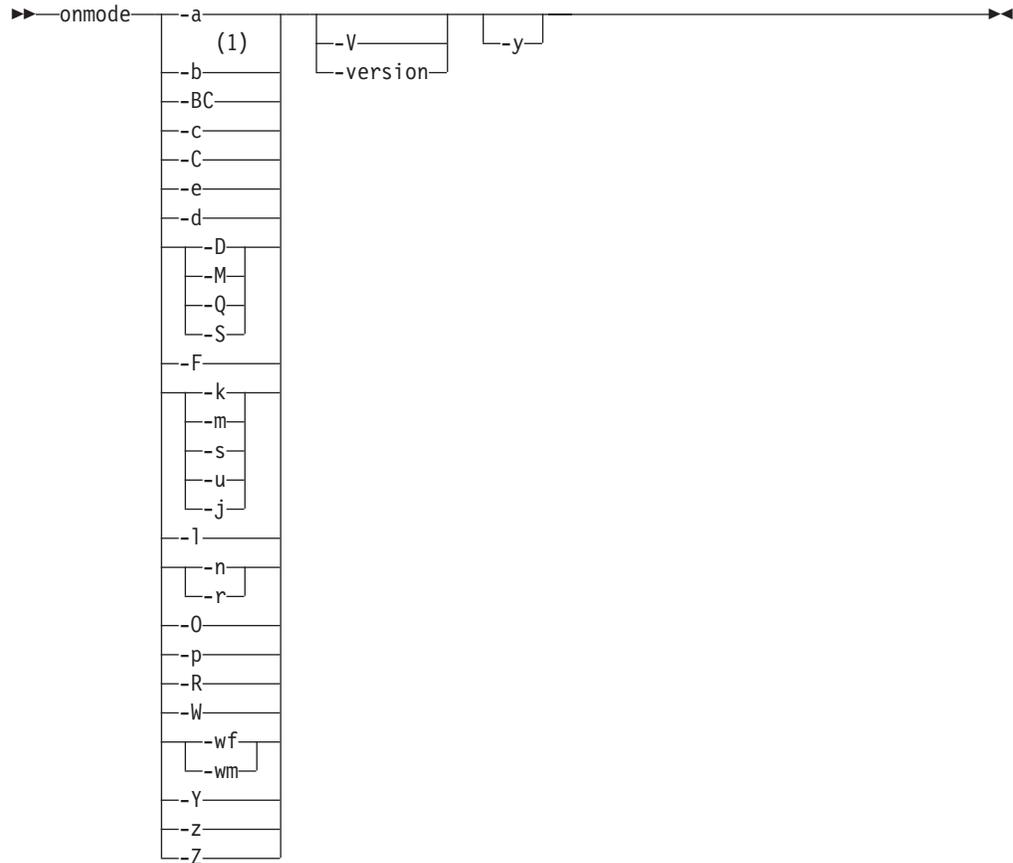
Table 11-1. **onmode** options

onmode option	Administrative API <i>command</i> string
onmode -a	ADD MEMORY
onmode -a <i>seg_size</i>	ONMODE
onmode -BC {1 2}	ONMODE
onmode -c	CHECKPOINT
onmode -c { block unblock }	ONMODE
onmode -C	ONMODE
onmode -d	ONMODE
onmode -D	ONMODE
onmode -e <i>keyword</i>	ONMODE
onmode -F	ONMODE
onmode -j	ONMODE
onmode -ku	ONMODE SHUTDOWN IMMEDIATE
onmode -l	ONMODE
onmode -m	ONMODE
onmode -M <i>kilobytes</i>	ONMODE
onmode -n	ONMODE
onmode -O	ONMODE
onmode -p { + - # } <i>class</i>	ONMODE
onmode -Q	ONMODE
onmode -r	ONMODE
onmode -R	ONMODE
onmode -S	ONMODE
onmode -W	ONMODE
onmode -wf	ONMODE
onmode -wm	ONMODE
onmode -Y { 0 1 }	ONMODE
onmode -z <i>session_id</i>	ONMODE

Table 11-1. **onmode** options (continued)

onmode option	Administrative API command string
onmode -Z address	ONMODE

onmode Syntax



Notes:

- 1 See the *IBM Informix Migration Guide* for onstat —b description.

Element	Purpose	Key Considerations
-y	Causes the database server to automatically respond yes to all prompts	None.
-V	Displays the software version number and the serial number	See "Obtaining Utility Version Information" on page 6-1.
-version	Displays the build version, host, OS, number and date, as well as the GLS version	See "Obtaining Utility Version Information" on page 6-1.

onmode -a: Add a shared-memory segment

Syntax:

►► onmode -a seg_size ◀◀

Element	Purpose	Key Considerations
-a seg_size	Allows you to add a new virtual shared-memory segment. Size is specified in kilobytes	Restrictions: The value of <i>seg_size</i> must be a positive integer. It must not exceed the operating-system limit on the size of shared-memory segments.

Ordinarily, you do not need to add segments to the virtual portion of shared memory because the database server automatically adds segments as they are needed. However, as segments are added, the database server might reach the operating-system limit for the maximum number of segments before it acquires the memory that it needs. This situation typically occurs when SHMADD is set so small that the database server exhausts the number of available segments before it acquires the memory that it needs for some operation.

If you manually add a segment that is larger than the segment specified by SHMADD, you can avoid exhausting the operating-system limit for segments but still meet the need that the database server has for additional memory.

Use ADD MEMORY as the Administrative API *command* string for **onmode -a**. Use ONMODE as the Administrative API *command* string for **onmode -a seg_size**.

onmode -BC: Allow large chunk mode

Syntax:

►► onmode -BC 1
-BC 2 ◀◀

Element	Purpose	Key Considerations
-BC 1	Enables support of large chunks, large offsets that are greater than 2 GB, and allows up to 32,768 chunks per dbspace.	This option allows large chunks to be created. Reversion without dropping the dbspace is possible if no chunks are larger than 2 GB. Dbspaces and blobspaces without chunks greater than 2 GB remain in the old format. After a chunk larger than 2 GB is added to a dbspace or blobspace then all chunks added or altered in that dbspace or blobspace are in the new format. See your <i>IBM Informix Administrator's Guide</i> .
-BC 2	Allows large-chunk-only mode for all dbspaces.	Reversion is not possible. Enables the 9.4 large chunk feature for all dbspaces and blobspaces, Any chunk or offset added or modified has the new format. Existing chunks that you do not alter remain in the old format. See your <i>IBM Informix Administrator's Guide</i>

The **onmode -BC** (backward-compatible) commands are useful if you have converted from Dynamic Server 9.40 (small chunk mode) to Dynamic Server 10.0

or later. When Dynamic Server 10.0 or later is first initialized (with the **oninit -iyv** command), by default it comes online with large chunk mode already fully enabled. Reversion is not possible. In the case of a newly initialized instance of Dynamic Server 10.0 or later, the **onmode -BC** commands will return an error.

Use ONMODE as the Administrative API *command* string for **onmode -BC**.

Note: After executing the **onmode -BC** command, perform a complete system level-0 backup.

onmode -c: Force a checkpoint

Syntax:

```

▶▶ onmode -c [block | unblock]

```

Element	Purpose	Key Considerations
-c	Forces a checkpoint that flushes the buffers to disk	You can use the -c option to force a sync checkpoint if the most recent checkpoint record in the logical log was preventing the logical-log file from being freed (status U-B-L).
block	Blocks the database server from any transactions	While the database server is blocked, users can access it in read-only mode. Use this option to perform an external backup on Dynamic Server. For more information, see the <i>IBM Informix Backup and Restore Guide</i> .
unblock	Unblocks the database server	When the database server is unblocked, data transactions and normal database server operations can resume. Use this option after you complete an external backup on Dynamic Server. For more information, see the <i>IBM Informix Backup and Restore Guide</i> .

Use CHECKPOINT as the Administrative API *command* command for **onmode -c**. Use ONMODE as the Administrative API *command* string for **onmode -c block** or **onmode -c unblock**.

onmode -C: Control the B-tree scanner

Syntax:

```

▶▶ onmode -C (yielding syntax)
    -start
    -count
    -stop count
    -kill count
    -threshold size
    -duration num
    -rangesize size
    -alice mode
    -compression value

```

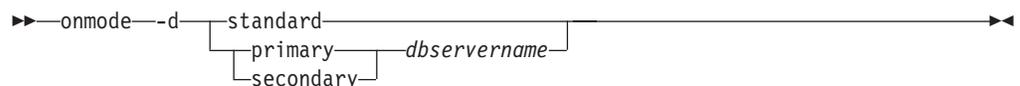
Element	Purpose	Key Considerations
-C	Controls the B-tree scanner for cleaning indexes of deleted items	There is no limit to the number of threads that can run at one time. However, there is a limit of 128 threads that can be started at one time. If, for example, you wanted 150 threads to run, you could execute two commands: onmode -C 100 and onmode -C 50 .
start count	Starts additional B-tree scanner threads.	If <i>count</i> is not specified, a <i>count</i> of 1 is assumed. There is no limit on the number of scanner threads that can be specified.
stop count kill count	Stops B-tree scanner threads.	If <i>count</i> is not specified, a <i>count</i> of 1 is assumed. Stopping all index scanners prevents all index cleaning. Either of these commands stop the B-tree scanner.
threshold size	Sets the minimum number of deleted items an index must encounter before an index is placed on the hot list.	Once all indexes above the threshold have been cleaned and there is no other work for the B-tree scanner to do, the indexes below the threshold are added to the hot list.
duration num	The number of seconds that the hot list is valid.	After this number of seconds expires, the hot list will be rebuilt by the next available B-tree scanner thread, even if unprocessed items are on the list. Scanners currently processing requests are not interrupted.
rangesize size	Determines the size of an index before index range cleaning is enabled.	A size of -1 can be used to disable range scanning.
alice num	Sets the system's alice mode.	Valid <i>num</i> values range from 0 (OFF) to 12.
compression value	For a database server instance, modifies the level at which two partially used index pages are merged. The pages are merged if the data on those pages totals a set level.	Valid values for the level are low, med (medium), high, and default. The system default value is med.

The B-tree scanner has statistical information which tracks index efficiency and how much extra work the index currently places on the server. Based on the amount of extra work the index has accomplished because of committed deleted index items, the B-tree scanner develops an ordered list of indexes that have caused the server to do extra work. This list is called the hot list. The index causing the highest amount of extra work is cleaned first and the rest of the indexes are cleaned in descending order. The DBA can allocate cleaning threads dynamically, thus allowing for configurable workloads.

Use ONMODE as the Administrative API *command* string for **onmode -C**.

onmode -d: Set data-replication types

Syntax:



Element	Purpose	Key Considerations
-d	Used to set the High-Availability Data-Replication type, either standard, primary, or secondary, as the following sections describe	You can use the -d primary and -d secondary options only when the database server is in quiescent mode. You can use the -d standard option when the database server is in quiescent, online, or read-only mode.
<i>dbservername</i>	Identifies the database server name of the primary or secondary database server	<p>The <i>dbservername</i> argument must correspond to the DBSERVERNAME parameter in the ONCONFIG file of the intended secondary database server. It should <i>not</i> correspond to one of the database servers that the DBSERVERALIASES parameter specifies.</p> <p>The <i>dbservername</i> argument of the other database server in the data-replication pair and the type of a database server (standard, primary, or secondary) is preserved after reinitialization of shared memory.</p> <p>For more information, see <i>range of values</i> for the DBSERVERNAME configuration parameter in “DBSERVERNAME” on page 1-25.</p>

Using the -d standard Option

The **-d standard** option drops the connection between database servers in a data replication pair (if one exists) and sets the database server type of the current database server to standard. This option does not change the mode or type of the other database server in the pair.

Using the -d primary dbservername Option

The **-d primary *dbservername*** option sets the database server type to primary and attempts to connect with the database server that *dbservername* specifies. If the connection is successful, data replication is turned on. The primary database server goes into online mode, and the secondary database server goes into read-only mode. If the connection is not successful, the database server comes to on-line mode, but data replication is not turned on.

Using the -d secondary dbservername Option

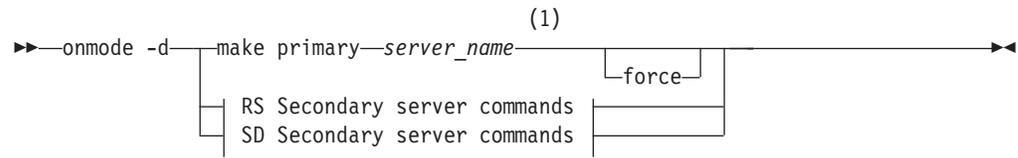
The **-d secondary *dbservername*** option sets the database server type to secondary and attempts to connect with the database server that *dbservername* specifies. If the connection is successful, data replication is turned on. The primary database server goes online, and the secondary database server goes into read-only mode. If the connection is not successful, the database server comes to read-only mode, but data replication is not turned on.

Setting data replication characteristics can also be performed using SQL Administration API *command* equivalents. For more information see *IBM Informix Guide to SQL: Syntax* and *IBM Informix Administrator's Guide*.

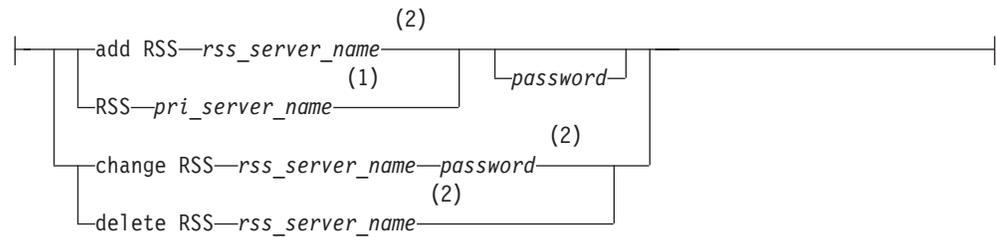
For other **onstat -d** information, see “onmode -d: Set High Availability server characteristics” on page 11-7 and “onmode -d: Replicate an index with data-replication” on page 11-8.

onmode -d: Set High Availability server characteristics

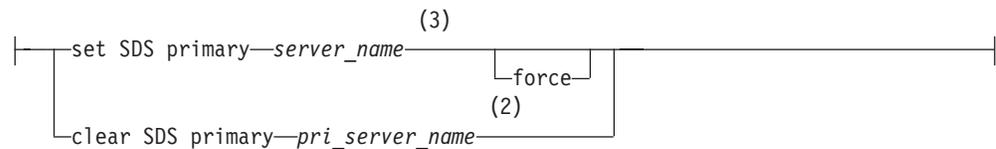
Syntax:



RS Secondary server commands:



SD Secondary server commands:



Notes:

- 1 Run on secondary server only.
- 2 Run on primary server only.
- 3 Run on primary server or secondary server.

Element	Purpose	Key Considerations
-d	Used to create, modify, or delete secondary servers in high-availability configurations	
add RSS	Adds an RS secondary server	This command should be run on the primary database server.
<i>rss_server_name</i>	Identifies the RS secondary database server name	The <i>servername</i> argument can be DBSERVERNAME, DBSERVERALIASES and ER group name. For more information, see <i>range of values</i> for DBSERVERNAME configuration parameter in "DBSERVERNAME" on page 1-25 and DBSERVERALIASES in "DBSERVERALIASES" on page 1-24.
<i>password</i>	Specifies the secondary server password	The password is used only during the first connection attempt. After the primary and secondary server have connected, the password cannot be changed.

Element	Purpose	Key Considerations
-d	Specifies how indexes are replicated to a High-Availability Data-Replication (HDR) secondary server when an index on the secondary server becomes corrupt	You can use the onmode -d idxauto and -d index commands while the server is in online mode.
idxauto	Enables automatic index replication when an index on a secondary server becomes corrupt	Use onmode -d idxauto to overwrite the value of the DRIDXAUTO configuration parameter within a session. For more information on DRIDXAUTO, see “DRIDXAUTO” on page 1-34. For more information on replicating indexes, see the chapter on using HDR in the <i>IBM Informix Administrator’s Guide</i> .
index	Replicates an index from a primary to a secondary server	If you detect a corrupt index on a secondary server, use the onmode -d index command to start replication of the index from the primary to the secondary server.
database	Specifies the database containing the index to replicate	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
index	Specifies the name of the index to replicate	Index must exist on table and in database specified. Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
owner	Specifies the owner of a table	You must specify the current owner of the table. Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .
table	Specifies the name of the table on which the index is based	Syntax must conform to the Table Name segment; see <i>IBM Informix Guide to SQL: Syntax</i> .

The **-d idxauto** and the **-d index** options provide methods to replicate an index to a secondary server containing a corrupted index. The base table will be locked during the transfer of an index. The alternative to using these options is to drop and rebuild the corrupt index on the primary server.

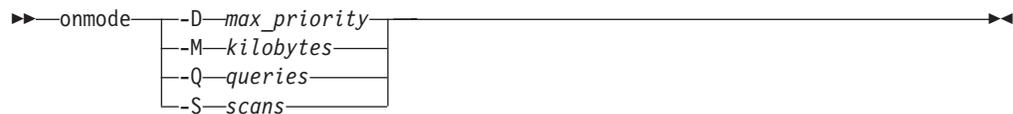
In the case of a fragmented index with one corrupt fragment, the **-d idxauto** option only transfers the single affected fragment, whereas the **-d index** option transfers the whole index.

Setting data replication characteristics can also be performed using SQL Administration API *command* equivalents. For more information see *IBM Informix Guide to SQL: Syntax* and *IBM Informix Administrator’s Guide*.

For other **onmode -d** information, see “onmode -d: Set High Availability server characteristics” on page 11-7 and “onmode -d: Set data-replication types” on page 11-5.

onmode -D, -M, -Q, -S: Change decision-support parameters

Syntax:



Element	Purpose	Key Considerations
-D max_priority	Changes the value of MAX_PDQPRIORITY	<p>This value must be an unsigned integer between 0 and 100.</p> <p>Specify <i>max_priority</i> as a factor to temper user requests for PDQ resources.</p> <p>For information on parameters used for controlling PDQ, see “MAX_PDQPRIORITY” on page 1-65 and the <i>IBM Informix Performance Guide</i>.</p>
-M kilobytes	Changes the value of DS_TOTAL_MEMORY	<p>This value has a platform-dependent upper limit. The value for 32-bit systems must be an unsigned integer between 128 * DS_MAX_QUERIES and 1,048,576. On 64-bit systems, the limit is generally higher and varies with the operating system. On HP 9000 platforms, for example, the maximum value is 4,294,967,296.</p> <p>Specify <i>kilobytes</i> for the maximum amount of memory available for parallel queries.</p> <p>For more information, see “DS_TOTAL_MEMORY” on page 1-39 and the <i>IBM Informix Performance Guide</i>.</p>
-Q queries	Changes the value of DS_MAX_QUERIES	<p>This value must be an unsigned integer between 1 and 8,388,608.</p> <p>Specify <i>queries</i> for the maximum number of concurrently executing parallel queries.</p> <p>For information on parameters used for controlling PDQ, see “DS_MAX_QUERIES” on page 1-36 and the <i>IBM Informix Performance Guide</i>.</p>
-S scans	Changes the value of DS_MAX_SCANS	<p>This value must be an unsigned integer between 10 and 1,048,576.</p> <p>Specify <i>scans</i> for the maximum number of concurrently executing parallel scans.</p> <p>For information on parameters used for controlling PDQ, see “DS_MAX_SCANS” on page 1-37 and the <i>IBM Informix Performance Guide</i>.</p>

These options allow you to change configuration parameters while the database server is online. The new values affect only the current instance of the database server; the values are not recorded in the ONCONFIG file. If you shut down and restart the database server, the values of the parameters revert to the values in the ONCONFIG file. For more information about these configuration parameters, see Chapter 1, “Configuration Parameters,” on page 1-1.

To check the current values for the MAX_PDQPRIORITY, DS_TOTAL_MEMORY, DS_MAX_SCANS, DS_MAX_QUERIES, and the DS_NONPDQ_QUERY_MEM configuration parameters, use **onstat -g mgm**. See “onstat -g mgm: Print MGM resource information ” on page 15-74.

Use ONMODE as the Administrative API *command* string for **onmode -D**, **onmode -M**, **onmode -Q**, or **onmode -S**.

onmode -e: Change usage of the SQL statement cache

Syntax:

►► onmode -e mode ◀◀

Element	Purpose	Key Considerations
onmode -e ENABLE	Enables the SQL statement cache. For more information, see the material on improving query performance in the <i>IBM Informix Performance Guide</i> .	User sessions use the cache only when they perform either of the following actions: <ul style="list-style-type: none"> Set the environment variable STMT_CACHE to 1 Execute the SQL statement SET STATEMENT CACHE ON
onmode -e FLUSH	Flushes the statements that are not in use from the SQL statement cache	The onstat -g ssc ref_cnt field shows 0.
onmode -e OFF	Turns off the SQL statement cache	No statements are cached.
onmode -e ON	Turns on the SQL statement cache	All statements are cached unless the user turns it off with one of the following actions: <ul style="list-style-type: none"> Set the environment variable STMT_CACHE to 0 Execute the SQL statement SET STATEMENT CACHE OFF

The **onmode -e** changes are in effect for the current database server session only. When you restart the database server, it uses the default STMT_CACHE parameter value in the ONCONFIG file.

Use ONMODE as the Administrative API *command* string for **onmode -e**.

onmode -F: Free unused memory segments

Syntax:

►► onmode -F ◀◀

Element	Purpose	Key Considerations
-F	Frees unused memory segments	None.

When you execute **onmode -F**, the memory manager examines each memory pool for unused memory. When the memory manager locates blocks of unused memory,

it immediately frees the memory. After the memory manager checks each memory pool, it begins checking memory segments and frees any that the database server no longer needs.

It is recommended that you run **onmode -F** from an operating-system scheduling facility regularly and after the database server performs any function that creates additional memory segments, including large index builds, sorts, or backups.

Running **onmode -F** causes a significant degradation of performance for any users that are active when you execute the utility. Although the execution time is brief (1 to 2 seconds), degradation for a single-user database server can reach 100 percent. Systems with multiple CPU virtual processors experience proportionately less degradation.

To confirm that **onmode** freed unused memory, check your message log. If the memory manager frees one or more segments, it displays a message that indicates how many segments and bytes of memory were freed.

Use ONMODE as the Administrative API *command* string for **onmode -F**.

onmode -k, -m, -s, -u, -j: Change database server mode

Syntax:



Element	Purpose	Key Considerations
-k	Takes the database server to offline mode and removes shared memory	To reinitialize shared memory, shut down and restart the database server. "Taking the Database Server to Offline Mode with the -k Option" on page 11-13.
-m	Takes the database server from quiescent or administration mode to online mode	See "Bringing the Database Server Online with the -m Option" on page 11-13.
-s	Shuts down the database server gracefully	Users who are using the database server are allowed to finish before the database server comes to quiescent mode, but no new connections are allowed. When all processing is finished, -s takes the database server to quiescent mode. The -s option leaves shared memory intact. See "Shutting Down the Database Server Gracefully with the -s Option" on page 11-13.
-u	Shuts down the database server immediately	This option brings the database server to quiescent mode without waiting for users to finish their sessions. Their current transactions are rolled back, and their sessions are terminated. See "Shutting Down the Database Server Immediately with the -u Option" on page 11-13.

Element	Purpose	Key Considerations
-j	Puts the database server into administration mode	This option brings the database server to administration mode, allowing the informix user all functions including the issuance of SQL and DDL commands. The -j -U option enables the DBSA to designate specific users (in addition to the informix user) to access the database server. See your <i>IBM Informix Administrator's Guide</i> .

The following sections describe the options that take the database server from one mode to another.

Taking the Database Server to Offline Mode with the -k Option

The **onmode -k** option takes the database server to offline mode and removes database server shared memory.

A prompt asks for confirmation. Another prompt asks for confirmation to kill user threads before the database server comes offline. If you want to eliminate these prompts, execute the **-y** option with the **-s** option.

This option does not kill all client sessions. Use the **-u** option to avoid hanging client sessions or virtual server processes.

Important: When you use the **onmode -k** command to shut down the database server, utilities that are waiting for a user response might not terminate. For example, **ontape** might be waiting for another tape, **onstat -i** might be waiting for a user response, or **onspaces** might be waiting for **y** or **n** to continue. If this problem occurs, use **onmode -uk** or **-uky** instead to roll back work before removing shared memory. For more information, see the descriptions of other options on this page.

Bringing the Database Server Online with the -m Option

The **-m** option brings the database server online from quiescent mode.

Shutting Down the Database Server Gracefully with the -s Option

The **-s** option causes a graceful shutdown. Users who are using the database server are allowed to finish before the database server comes to quiescent mode, but no new connections are allowed. When all processing is finished, **-s** takes the database server to quiescent mode. The **-s** option leaves shared memory intact.

A prompt asks for confirmation. If you want to eliminate this prompt, execute the **-y** option with the **-s** option.

Shutting Down the Database Server Immediately with the -u Option

The **-u** option causes immediate shutdown. This option brings the database server to quiescent mode without waiting for users to finish their sessions. Their current transactions are rolled back, and their sessions are terminated.

A prompt asks for confirmation. Another prompt asks for confirmation to kill user threads before the database server comes to quiescent mode. If you want to eliminate these prompts, execute the `-y` option with the `-s` option.

Changing the Database Server to Administration Mode with the `-j` Option

The `-j` option puts the database server into the administration mode and allows only the DBSA group and the user **informix** to connect to the server. The `-j` option allows a DBSA to have the server in a fully functional mode to perform maintenance.

The `-j -U` option enables the DBSA to grant individual users access to the database server in administration mode. Once connected, these individual users can execute any SQL or DDL command. When the server is changed to administration mode, all sessions for users other than user **informix**, the DBSA group users, and those identified in the **onmode -j -U** command lose their database server connection.

The following example enables three individual users to connect to the database server and have database server access until the database server mode changes to offline, quiescent or online mode:

```
onmode -j -U karin,sarah,andrew
```

Access for individual users can also be removed by executing **onmode -j -U** and removing their name from the new list of names in the command. For example, in the following commands, the first command grants only Karin access, the second command grants Karin and Sarah access, and the third command grants only Sarah access (and removes access from Karin).

```
onmode -j -U karin
onmode -j -U karin,sarah
onmode -j -U sarah
```

To allow user **informix** and the DBSA group user to retain their database server access in administration mode and remove all single users from accessing the database server, use the following command:

```
onmode -j -U ' '
```

For information on designating single users in administration mode using a configuration parameter, see “ADMIN_MODE_USERS” on page 1-11

Changing Database Server Mode with ON-Monitor (UNIX)

You can also use ON-Monitor options to change the database server mode. The following table shows ON-Monitor options that are equivalent to the **onmode** options.

onmode Option	ON-Monitor Option
<code>-k</code>	Take-Offline
<code>-m</code>	On-Line
<code>-s</code>	Graceful-Shutdown
<code>-u</code>	Immediate-Shutdown
<code>-j</code>	Administration Mode

onmode -l: Switch the logical-log file

Syntax:

►► onmode -l ◀◀

Element	Purpose	Key Considerations
-l	Switches the current logical-log file to the next logical-log file	You must use onmode to switch to the next logical-log file. For information on switching to the next logical-log file, see the chapter on managing logical-log files in the <i>IBM Informix Administrator's Guide</i> .

Use ONMODE as the Administrative API *command* string for **onmode -l**.

onmode -n, -r: Change shared-memory residency

Syntax:

►► onmode [-n] [-r] ◀◀

Element	Purpose	Key Considerations
-n	Ends forced residency of the resident portion of shared memory	This command does not affect the value of RESIDENT, the forced-residency parameter in the ONCONFIG file.
-r	Starts forced residency of the resident portion of shared memory	This command does not affect the value of RESIDENT, the forced-memory parameter in the ONCONFIG file.

Important: Set the RESIDENT parameter to 1 before you use the **onmode -r** or **-n** options.

For information on using the forced-residency parameter to turn residency on or off for the next time that you restart the database server, see the chapter on managing shared memory in the *IBM Informix Administrator's Guide*.

Use ONMODE as the Administrative API *command* string for **onmode -n** or **onmode -r**.

onmode -O: Override ONDBSPACEDOWN WAIT mode

Syntax:

►► onmode -O ◀◀

Element	Purpose	Key Considerations
-O	Overrides the WAIT mode of the ONDBSPACEDOWN configuration parameter	None.

Use the **onmode -O** option only in the following circumstances:

- ONDBSPACEDOWN is set to WAIT.
- A disabling I/O error occurs that causes the database server to block all updating threads.
- You cannot or do not want to correct the problem that caused the disabling I/O error.
- You want the database server to mark the disabled dbspace as down and continue processing.

When you execute this option, the database server marks the dbspace responsible for the disabling I/O error as down, completes a checkpoint, and releases blocked threads. Then **onmode** prompts you with the following message:

```
This will render any dbspaces which have incurred disabling I/O errors unusable
and require them to be restored from an archive.
Do you wish to continue?(y/n)
```

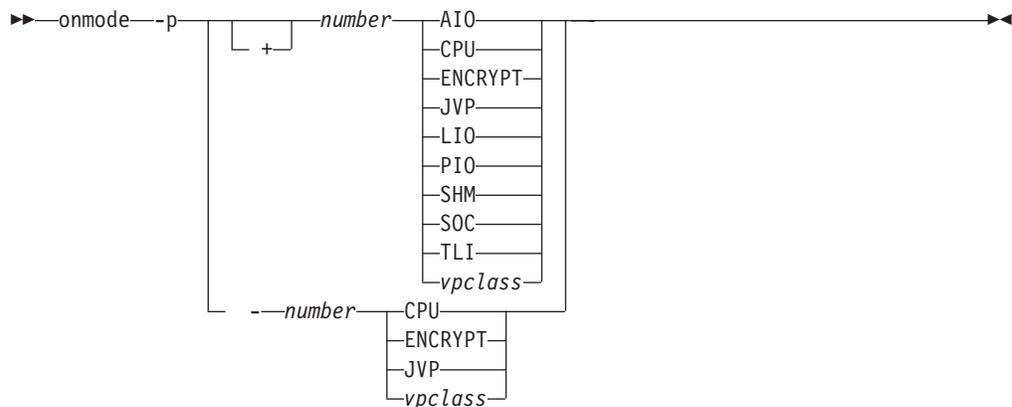
If **onmode** does not find any disabling I/O errors on noncritical dbspaces when you run the **-O** option, it notifies you with the following message:

```
There have been no disabling I/O errors on any noncritical dbspaces.
```

Use ONMODE as the Administrative API *command* string for **onmode -O**.

onmode -p: Add or remove virtual processors

Syntax:



Element	Purpose	Key Considerations
-p number	<p>Adds or removes virtual processors. The <i>number</i> argument indicates the number of virtual processors to add or remove</p> <p>If this value is a negative integer, processors are removed. If this value is a positive integer, processors are added.</p>	<p>You can use the -p option only when the database server is in online mode, and you can add to only one class of virtual processors at a time.</p> <p>For more details, see “Adding and Dropping Virtual Processors” on page 11-17.</p> <p>If you are removing virtual processors, the maximum cannot exceed the actual number of processors of the specified type. If you are adding virtual processors, the maximum number depends on the operating system.</p> <p>For more information, see the chapter on using virtual processors in the <i>IBM Informix Administrator’s Guide</i>.</p>

Element	Purpose	Key Considerations
AIO	Performs nonlogging disk I/O to cooked disk spaces	Also performs nonlogging I/O to raw disk spaces if kernel asynchronous I/O (KAIO) is not used.
CPU	Runs all session threads and some system threads	It is recommended that the number of CPU VPs not be greater than the number of physical processors. If KAIO is used, performs I/O to raw disk spaces, including I/O to physical and logical logs. Runs thread for KAIO where available or a single poll thread. The database server uses the number of CPU VPs to allocate resources for parallel database queries (PDQ). If you drop CPU VPs, your queries will run significantly slower. The Reinit field of the onstat -g mgm output displays information on the number of queries that are waiting for running queries to complete after an onmode -p command. Also see the <i>IBM Informix Performance Guide</i> .
ENCRYPT	Executes column-level encryption and decryption routines	Specify more ENCRYPT virtual processors if you have multiple encrypted columns.
JVP	Executes Java user-defined routines in the Java Virtual Machine (JVM)	Specify more JVPs if you are running many Java UDRs.
LIO	Writes to the logical-log files if they are in cooked disk space	Use two LIO virtual processors only if the logical logs are in mirrored dbspaces. The database server allows a maximum of two LIO virtual processors.
PIO	Writes to the physical log if it is in cooked disk space	Use two PIO virtual processors only if the physical log is in a mirrored dbspace. The database server allows a maximum of two PIO virtual processors.
SHM	Performs shared-memory communication	You can use the SHM virtual processor even if the database server is not configured for shared-memory communication.
SOC	Uses sockets to perform network communications	You can use the SOC virtual processor only if the database server is configured for network connections through sockets.
STR	Performs stream pipe connections	
TLI	Uses the Transport Layer Interface (TLI) to perform network communication	You can use the TLI virtual processor only if the database server is configured for network connections through TLI.
<i>vpclass</i>	Names a user-defined virtual processor class	Use the VPCLASS parameter in the ONCONFIG to define the user-defined virtual-processor class. Specify more user-defined virtual processors if you are running many UDRs. On Windows, you can have only one user-defined virtual processor class at a time. Omit the <i>number</i> parameter in the onmode -p vpclass command. For more information on extension classes, see “VPCLASS” on page 1-109.

Adding and Dropping Virtual Processors

The following rules about adding or dropping virtual processors apply:

- You can add but not drop virtual processors of the AIO, PIO, LIO, TLI, SHM, SOC, and STR classes.
- You cannot add or drop virtual processors of the OPT, ADM, ADT, and MSC classes. The database server adds them automatically.
- You can add or drop virtual processors of the CPU, ENCRYPT, JVP, and user-defined (*vpclass*) classes.
-

Windows Only: You can add a virtual processor of any class, but you cannot drop virtual processors.

Dropping Virtual Processors Automatically

Table 11-2 shows the virtual processors that the database server starts automatically. You cannot add or drop these virtual processors with the **onmode -p** command. To drop these virtual processors, shut down and restart the database server.

Table 11-2. Virtual-Processor Classes That the Database Server Starts Automatically

Virtual-Processor Class	Description
ADM	Performs administrative functions
ADT	Runs auditing processes The database server starts one virtual processor in the audit class when you turn on audit mode by setting the ADTMODE parameter in the ONCONFIG file.
MSC	Services requests for system calls that require a large stack The database server starts this virtual processor automatically.
OPT	Performs I/O to the optical disk The database server starts one OPT virtual processor when you use the Optical Subsystem.

Monitoring Poll Threads with onstat

While the database server is online, you cannot drop a CPU virtual processor that is running a poll thread. To identify poll threads that run on CPU virtual processors, use the following command:

```
onstat -g ath | grep 'cpu.*poll'
```

The following **onstat -g ath** output shows two CPU virtual processors with poll threads. In this situation, you cannot drop to fewer than two CPU virtual processors.

```
tid tcb      rstcb prty  status  vp-class  name
8   a362b90  0     2     running 1cpu    tlitcpoll
9   a36e8e0  0     2     cond wait arrived 3cpu
```

For more information on the types of virtual processors, see the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*.

Use ONMODE as the Administrative API *command* string for **onmode -p**.

onmode -R: Regenerate .infos File

Syntax:

```
▶▶ onmode --R ◀◀
```

Element	Purpose	Key Considerations
-R	Re-creates the <code>.infos.dbservername</code> file	Before you use the -R option, set the INFORMIXSERVER environment variable to match the DBSERVERNAME parameter from the ONCONFIG file. Do not use the -R option if INFORMIXSERVER is one of the DBSERVERALIAS names. For more information, see “.infos.dbservername” on page A-6.

The database server uses information from the `.infos.dbservername` file when it accesses utilities. The database server creates and manages this file, and you should never need to do anything to the file. However, if `.infos.dbservername` is accidentally deleted, you must either recreate the file or shut down and restart the database server.

Use **ONMODE** as the Administrative API *command* string for **onmode -R**.

onmode -W: Change settings for the SQL statement cache

Syntax:

```

>> onmode -W STMT_CACHE_HITS hits
                |
                | STMT_CACHE_NOLIMIT value
  
```

Element	Purpose	Key Considerations
STMT_CACHE_HITS <i>hits</i>	Specifies the number of hits (references) to a statement before it is fully inserted in the SQL statement cache. Set <i>hits</i> to 1 or more to exclude ad hoc queries from entering the cache.	You can only increase or reset the value of STMT_CACHE_HITS . The new value displays in the #hits field of the onstat-gssc output. If <i>hits</i> = 0, the database server inserts all qualified statements and its memory structures in the cache. If <i>hits</i> > 0 and the number of times the SQL statement has been executed is less than STMT_CACHE_HITS , the database server inserts <i>key-only</i> entries in the cache. It inserts qualified statements in the cache after the specified number of hits have been made to the statement. ONCONFIG Parameter: STMT_CACHE_HITS
STMT_CACHE_NOLIMIT <i>value</i>	Controls whether statements are inserted in the SQL statement cache.	If <i>value</i> = 0, the database server inserts no statements in the cache. If <i>value</i> = 1, the database server always inserts statements in the cache. If none of the queries are shared, turn off STMT_CACHE_NOLIMIT to prevent the database server from allocating a large amount of memory for the cache. ONCONFIG Parameter: STMT_CACHE_NOLIMIT

SQL Statement Cache Examples

The following are examples of **onmode -W** commands for changing SQL statement cache (SSC) settings. The changes are in effect for the current database server session only and do not change the **ONCONFIG** values. When you restart the

database server, it uses the default SSC settings, if not specified in the ONCONFIG file, or the ONCONFIG settings. To make the changes permanent, set the appropriate configuration parameter.

```
onmode -W STMT_CACHE_HITS 2 # number of hits before statement is
# inserted into SSC
onmode -W STMT_CACHE_NOLIMIT 1 # always insert statements into
# the cache
```

Use ONMODE as the Administrative API *command* string for **onmode -W**.

onmode -wf, -wm: Dynamically change certain configuration parameters

Syntax:

```
▶▶ onmode [ -wf config_param=value | -wm config_param=value ] ▶▶
```

Use **onmode -wf** or **onmode -wm** to dynamically change certain configuration parameters.

Element	Purpose	Key Considerations
-wf	Updates the value of the specified configuration parameter in the ONCONFIG file.	None.
-wm	Dynamically sets the value of the specified configuration parameter for the current session.	None.

Element	Purpose	Key Considerations
<i>config_param=value</i>	<p>Specifies the configuration parameter and its new value. The following configuration parameters can be specified:</p> <ul style="list-style-type: none"> • ADMIN_MODE_USERS • AUTO_AIOVPS • AUTO_CKPTS • DS_MAX_QUERIES • DS_MAX_SCANS • DS_NONPDQ_QUERY_MEM • DS_TOTAL_MEMORY • DUMPCNT • DUMPSHMEM • DYNAMIC_LOGS • EXPLAIN_STAT • HA_ALIAS • LIMITNUMSESSIONS • LISTEN_TIMEOUT • LOG_INDEX_BUILDS • LTXEHWM • LTXHWM • MAX_INCOMPLETE_CONNECTIONS • MAX_PDQPRIORITY • MSG_DATE • ONLIDX_MAXMEM • RESIDENT • RTO_SERVER_RESTART • SBSPACENAME • SBSPACETEMP • SDS_TIMEOUT • TEMPTAB_NOLOG • USELASTCOMMITTED • VP_MEMORY_CACHE_KB 	See "Summary of Configuration Parameters" on page 1-3.

Use ONMODE as the Administrative API *command* string for **onmode -wf** or **onmode -wm**.

onmode -wm: Change LRU tuning status

Syntax:

```

▶▶ onmode -wm AUTO_LRU_TUNING [0|1], [min|max|min, max]

```

Use **onmode -wm** to change LRU tuning status.

Element	Purpose	Key Considerations
-wm	Dynamically sets the value of the specified configuration parameter for the current session.	None.
0	Turns off automatic LRU tuning for the current session.	None.
1	Turns on automatic LRU tuning for the current session.	None.
min	Sets the LRU flushing parameters to a minimum	LRU flushing parameters can be set when LRU_TUNING is either on or off.
max	Sets the LRU flushing parameters to a maximum	LRU flushing parameters can be set when LRU_TUNING is either on or off.
min, max	Sets the LRU flushing parameters to a minimum and a maximum.	LRU flushing parameters can be set when LRU_TUNING is either on or off. If a <i>min</i> value is greater or equal to the <i>max</i> value for a specific buffer pool, the <i>min</i> value is ignored and no error message is returned.

Use ONMODE as the Administrative API *command* string for **onmode -wm**.

onmode -Y: Dynamically change SET EXPLAIN

Syntax:

Element	Purpose	Key Considerations
-Y	Dynamically change the value of the SET EXPLAIN statement.	None.
<i>sessionid</i>	Identifies the specific session.	None.

You can use the SET EXPLAIN statement to display the query plan of the optimizer, an estimate of the number of rows returned, and the relative cost of the query. When you use the **onmode -Y** command to turn on SET EXPLAIN, the output is displayed in the **sqexplain.out.sessionid** file. If an **sqexplain.out** file already exists, the database server reads that file until an administrator turns off the SET EXPLAIN for the session.

The **onmode -Y** command dynamically changes the value of the SET EXPLAIN statement for an individual session. The following invocations are valid with this command:

Invocation	Explanation
onmode -Y sessionid 2	Turns SET EXPLAIN on for <i>sessionid</i> and displays the query plan only
onmode -Y sessionid 1	Turns SET EXPLAIN on for <i>sessionid</i>
onmode -Y sessionid 0	Turns SET EXPLAIN off for <i>sessionid</i>

For more information on using the SET EXPLAIN statement, see the *IBM Informix Guide to SQL: Syntax*. For more information on interpreting the `sqexplain.out` file to improve query performance, see the *IBM Informix Performance Guide*.

Use ONMODE as the Administrative API *command* string for **onmode -Y**.

onmode -z: Kill a database server session

Syntax:

►► onmode -z sid ◀◀

Element	Purpose	Key Considerations
-z sid	Kills the session that you specify in <i>sid</i>	This value must be an unsigned integer greater than 0 and must be the session identification number of a currently running session.

To use the **-z** option, first obtain the session identification (*sessid*) with **onstat -u**, then execute **onmode -z**, substituting the session identification number for *sid*.

When you use **onmode -z**, the database server attempts to kill the specified session. If the database server is successful, it frees any resources that the session holds. If the database server cannot free the resources, it does not kill the session.

If the session does not exit the section or release the latch, the database server administrator can take the database server offline, as described in “Taking the Database Server to Offline Mode with the -k Option” on page 11-13, to close all sessions.

Use ONMODE as the Administrative API *command* string for **onmode -z**.

onmode -Z: Kill a distributed transaction

Syntax:

►► onmode -Z address ◀◀

Element	Purpose	Key Considerations
-Z address	Kills a distributed transaction associated with the shared-memory address <i>address</i>	<p>This argument must be the address of an ongoing distributed transaction that has exceeded the amount of time that TXTIMEOUT specifies. The address must conform to the operating-system-specific rules for addressing shared-memory. (The address is available from onstat -x output.)</p> <p>This option is not valid until the amount of time that the ONCONFIG parameter TXTIMEOUT specifies has been exceeded. The -Z option should rarely be used and only by an administrator of a database server involved in distributed transactions.</p> <p>For information on initiating independent actions in a two-phase commit protocol, see the chapter on multiphase commit protocols in the <i>IBM Informix Administrator's Guide</i>.</p>

Distributed transactions provide the ability to query data on different database servers.

Warning: If applications are performing distributed transactions, killing one of the distributed transactions can leave your client/server database system in an inconsistent state. Try to avoid this situation.

Use ONMODE as the Administrative API *command* string for **onmode -Z**.

Chapter 12. The ON-Monitor Utility

Using ON-Monitor (UNIX)

Use the ON-Monitor utility to perform various administrative tasks. This section provides a quick reference for the ON-Monitor screens. To start ON-Monitor, execute the following command from the operating-system prompt:

```
onmonitor
```

If you are logged in as user **informix** or user **root**, the main menu appears. All users other than **informix** and **root** have access only to the Status menu.

The ON-Monitor main menu displays the following menus:

- **Status** menu
- **Parameters** menu
- **Dbspaces** menu
- **Mode** menu
- **Force-Ckpt** menu
- **Archive** menu
- **Logical-Logs** menu
- **Exit** option

These menus are shown on the following pages (Table 12-1 on page 12-2 through Table 12-7 on page 12-3).

To obtain ON-Monitor version information, execute the **-V** or **-version** command from the operating-system prompt. For complete information on version information, see "Obtaining Utility Version Information" on page 6-1

You cannot use ON-Monitor on High-Availability Data Replication (HDR) secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

Navigating ON-Monitor and Using Help

All menus and screens in ON-Monitor function in the same way. For menus, use the arrow keys or SPACEBAR to scroll to the option that you want to execute and press RETURN, or press the first capitalized letter of the option (usually the first letter). When you move from one option to the next by pressing SPACEBAR or an arrow key, the option explanation (line 2 of the menu) changes.

If you want general instructions for a specific screen, press CTRL-W. If you need help to determine what you should enter in a field on the screen, use the TAB key to highlight the field and press CTRL-F or F2.

Some of the menus display ellipses (...) on the far right or left side. The ellipses indicate that you can move in the direction of the dots, using the arrow keys or SPACEBAR, to view other options.

Executing Shell Commands Within ON-Monitor

To execute a shell command from within ON-Monitor, type an exclamation point (!) followed by the command. For example, to list the files in the current directory, type the following command:

```
!ls
```

ON-Monitor Screen Options

Table 12-1. Status Menu

Menu	Description
Profile	Displays database server performance statistics
Userthreads	Displays the status of active user threads
Spaces	Displays status information about database server storage spaces and chunks
Databases	Displays the name, owner, and logging mode of the 100 first databases
Logs	Displays status information about the physical-log buffer, the physical log, the logical-log buffer, and the logical-log files
Archive	Displays a list of all backup tapes and logical-log files that you require to restore data using ontape
data-Replication	Displays High-Availability Data-Replication (HDR) status and configuration
Output	Stores the output of other status information in a specified file
Configuration	Copies the current database server configuration to a file

Table 12-2. Parameters Menu

Menu	Description
Initialize	Initializes database server disk space or modifies disk-space parameters
Shared-Memory	Initializes database server shared memory or modifies shared-memory parameters
perFormance	Specifies the number of virtual processors for each VP class
data-Replication	Specifies the HDR parameters
diaGnostics	Specifies values for the diagnostics parameters
pdQ	Changes parameters for parallel database queries
Add-Log	Adds a logical-log file to a dbspace
Drop-Log	Drops a logical-log file from a dbspace
Physical-Log	Changes the size or the location of the database server physical log

Table 12-3. Dbspaces Menu

Menu	Description
Create	Creates a dbspace
BLOBSpace	Creates a blobspace
Mirror	Adds mirroring to an existing storage space or ends mirroring for a storage space
Drop	Drops a storage space from the database server configuration

Table 12-3. Dbspaces Menu (continued)

Menu	Description
Info	Displays the identification number, location, and fullness of each chunk assigned to a storage space
Add_chunk	Adds a chunk to a storage space
datasKip	Changes the database parameter
Status	Changes the status of a chunk in a mirrored pair

Table 12-4. Mode Menu

Menu	Description
Startup	Initializes shared memory and takes the database server to quiescent mode
On-Line	Takes the database server from quiescent to online mode
Graceful-Shutdown	Takes the database server from online to quiescent mode so users can complete work
Immediate-Shutdown	Takes the database server from online to quiescent mode in 10 seconds
Take-Offline	Detaches shared memory and immediately takes the database server to offline mode
Add-Proc	Adds virtual processors
Drop-Proc	Drops virtual processors
Decision-support	Sets decision-support parameters dynamically
Administration	Tells the server to change into administration mode

Table 12-5. Force-Ckpt Menu

Menu	Description
Force-Ckpt	Displays the time of the most-recent checkpoint or forces the database server to execute a checkpoint

Table 12-6. Archive Menu

Menu	Description
Tape-Parameters	Modifies the ontape parameters for the backup tape device

Table 12-7. Logical Logs Menu

Menu	Description
Databases	Modifies the logging status of a database
Tape-Parameters	Modifies the ontape parameters for the logical-log backup tape device

Setting Configuration Parameters in ON-Monitor

Figure 12-1 on page 12-4 shows which ONCONFIG parameters correspond to the Initialization screen.

DISK PARAMETERS			
Page Size	[2] Kbytes		Mirror {MIRROR}
Tape Dev.	{TAPEDEV}		
Block Size	{TAPEBLK}	Total Tape Size	{TAPESIZE}
Log Tape Dev.	{LTAPEDEV}		
Block Size	{LTAPEBLK}	Total Tape Size	{LTAPESIZE}
Stage Blob	{STAGEBLOB}		
Root Name	{ROOTNAME}	Root Size	{ROOTSIZE}
Primary Path	{ROOTPATH}		
		Root Offset	{ROOTOFFSET}
Mirror Path	{MIRRORPATH}		
		Mirror Offset	{MIRROROFFSET}
Phy. Log Size	{PHYSFILE}	Log. Log Size	{LOGSIZE}
		Number of Logical Logs	{LOGFILES}

Figure 12-1. Initialization Screen with Parameter Names

Note: If you change the value of the ROOTSIZE configuration parameter after the database server is initialized, the change will not be effective until the database server is reinitialized, a process which destroys all data on the disk.

Figure 12-2 shows which ONCONFIG parameters correspond to the Shared-Memory screen.

SHARED MEMORY PARAMETERS			
Server Number	{SERVERNUM}	Server Name	{DBSERVERNAME}
Server Aliases	{DBSERVERALIASES}		
DbSPACE Temp	{DBSPACE TEMP}		
Deadlock Timeout	{DEADLOCK_TIMEOUT}	DbSPACE Down Option	{ONDBSPACEDOWN}
Forced Residency	{RESIDENCY}	Number of Page Cleaners	{CLEANERS}
Non Res. SegSize (K)	{SHMVIRT SIZE}	Stack Size (K)	{STACKSIZE}
Heterogeneous Commit	{HETERO_COMMIT}	Optical Cache Size (K)	{OPCACHEMAX}
Physical Log Buffer Size	{PHYSBUFF}	Transaction Timeout	{TXTIMEOUT}
Logical Log Buffer Size	{LOGBUFF}	Index Page Fill Factor	{FILLFACTOR}
Max # of Locks	{LOCKS}	Add SegSize	{SHMADD}
Max # of Buffers	{BUFFERS}	Total Memory	{SHMTOTAL}
Resident Shared Memory size	[] Kbytes	Page Size	[2] Kbytes

Figure 12-2. Shared-Memory Screen with Parameter Names

Note: Although Dynamic Server can support a shared memory segment that is larger than 4 gigabytes, ON-Monitor does not support a shared memory segment that is larger than 4 gigabytes. Therefore, the ON-Monitor screen cannot hold values that are larger than 4 gigabytes.

Figure 12-3 on page 12-5 shows which ONCONFIG parameters correspond to the Performance Tuning screen.

PERFORMANCE TUNING PARAMETERS			
Multiprocessor Machine	{MULTIPROCESSOR}	LRU Max Dirty	{LRU_MAX_DIRTY}
Num Procs to Affinity	{VPCLASS aff}	LRU Min Dirty	{LRU_MIN_DIRTY}
Proc num to start with	{VPCLASS num}	Checkpoint Interval	{CKPTINTVL}
CPU VPs	{VPCLASS cpu}	Num of Read Ahead Pages	{RA_PAGES}
AIO VPs	{VPCLASS aio}	Read Ahead Threshold	{RA_THRESHOLD}
Single CPU VP	{SINGLE_CPU_VP}	NETTYPE settings:	
Use OS Time	{USE_OS_TIME}	Protocol Threads	Users VP-class
Disable Priority Aging	{VPCLASS noage}	[] [] [] []	
Offline Recovery Threads	{OFF_RECVRY_THREADS}		
Online Recovery Threads	{ON_RECVRY_THREADS}		
Num of LRU queues	{LRUS}		

Figure 12-3. Performance Screen with Parameter Names

Figure 12-4 shows which ONCONFIG parameters correspond to the Data Replication screen.

DATA REPLICATION PARAMETERS	
Interval	{DRINTERVAL}
Timeout	{DRTIMEOUT}
Lost & Found	{DRLOSTFOUND}

Figure 12-4. Data-Replication Screen with Parameter Names

Figure 12-5 shows which ONCONFIG parameters correspond to the Diagnostics screen.

DIAGNOSTIC PARAMETERS	
Message Log	{MSGPATH}
Console Msgs.	{CONSOLE}
Alarm Program	{ALARMPROGRAM}
Dump Shared Memory	{DUMPSHMEM}
Dump Gcore	{DUMPGCORE}
Dump Core	{DUMPCORE}
Dump Count	{DUMPCNT}
Dump Directory	{DUMPDIR}

Figure 12-5. Diagnostics Screen with Parameter Names

Figure 12-6 shows which ONCONFIG parameters correspond to the PDQ screen.

PARALLEL DATABASE QUERIES PARAMETERS	
Max PDQ Priority	{MAX_PDQPRIORITY}
Decision Support Queries	{DS_MAX_QUERIES}
Decision Support Memory (Kbytes)	{DS_TOTAL_MEMORY}
Maximum Decision Support Scans	{DS_MAX_SCANS}
Dataskip	{DATASKIP}
Optimizer Hint	{OPTCOMPIND}
Non PDQ Memory	{DS_NONPDQ_QUERY_MEM}

Figure 12-6. PDQ Screen with Parameter Names

Figure 12-7 on page 12-6 shows the ON-Monitor screen for creating a dbspace.

```

CREATE DBSPACE
DbSPACE Name [          ] Mirror [ ] Temp [ ]
Page Size [ 2] Kbytes
PRIMARY CHUNK INFORMATION:
Full Pathname [          ]
Offset [          0] Kbytes      Size [          0] Kbytes
MIRROR CHUNK INFORMATION:
Full Pathname [          ]
Offset [          0] Kbytes

```

Figure 12-7. Create Dbspace Screen

Note: All tables, indexes, and other allocations within the dbspace use pages of the specified page size. The value for Page Size must be a multiple of the page size of the root dbspace.

Chapter 13. The onparams Utility

In This Chapter

This chapter shows you how to use the following **onparams** options:

- “ **onparams -a -d *dbspace***: Add a logical-log file” on page 13-2
- “ **onparams -d -l *lognum***: Drop a logical-log file” on page 13-2
- “ **onparams -p**: Change physical-log parameters” on page 13-3
- “**onparams -b**: Add a new buffer pool” on page 13-4

Any **onparams** command fails if a storage-space backup is in progress. If you do not use any options, **onparams** returns a usage statement.

In addition to using **onparams** to manage your database server, you can also use Administrative API commands. Table 13-1 identifies the **onparams** options and the Administrative API *command* string.

Table 13-1. **onparams** Options

Function	onparams Command	Database Server Mode	Administrative API <i>command</i> string
Add a logical-log file	onparams -a -d <i>dbspace</i> [-i]	Online, quiescent, or fast-recovery mode	ADD LOG
Drop a logical-log file	onparams -d -l <i>lognum</i>	Online, quiescent, or fast-recovery mode	DROP LOG
Change the size or location of the physical log	onparams -p	Quiescent mode only	ALTER PLOG
Add a new buffer pool	onparams -b	Online, quiescent, or administration mode	ADD BUFFERPOOL

On UNIX, you must be logged in as user **root** or user **informix** to execute **onparams**. Only user **informix** is allowed to execute the Administrative API *command* strings.

On Windows, you must be a member of the **Informix-Admin** group to execute **onparams**.

You cannot use the **onparams** utility on High-Availability Data Replication (HDR) secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

onparams Syntax

```
▶▶ onparams ▶▶  
-a -d dbspace  
-d -l lognum  
-p  
-b  
-v  
-version
```

Element	Purpose	Key Considerations
-V	Displays the software version number and the serial number	See "Obtaining Utility Version Information" on page 6-1.
-version	Displays the build version, host, OS, number and date, as well as the GLS version	See "Obtaining Utility Version Information" on page 6-1.

onparams -a -d *dbspace*: Add a logical-log file

Syntax:

```

▶▶ onparams --a --d dbspace [ -s size ] [ -i ]

```

Element	Purpose	Key Considerations
-a -d <i>dbspace</i>	Adds a logical-log file to the end of the log-file list to the specified <i>dbspace</i>	<p>You can add a log file to a <i>dbspace</i> only if the database server has adequate contiguous space. The newly added log files have a status of A and are immediately available for use. You can add a log file during a backup. You can have a maximum of 32,767 logical-log files. Use onstat -l to view the status of your logical-log files. It is recommended that you take a level-0 backup of the root <i>dbspace</i> and the <i>dbspace</i> that contains the log file as soon as possible.</p> <p>You cannot add a log file to a <i>blob</i>space or <i>sb</i>space.</p> <p>Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i>.</p>
-i	Inserts the logical-log file after the current log file	Use this option when the Log File Required alarm prompts you to add a logical-log file.
-s <i>size</i>	Specifies a size in kilobytes for the new logical-log file	<p>This value must be an unsigned integer greater than or equal to 200 kilobytes.</p> <p>If you do not specify a size with the -s option, the size of the log file is taken from the value of the LOGSIZE parameter in the ONCONFIG file when database server disk space was initialized.</p> <p>For information on changing LOGSIZE, see the chapter on managing logical-log files in the <i>IBM Informix Administrator's Guide</i>.</p>

Use ADD CHUNK as the Administrative API *command* string for **onparams -a -d *dbspace* [-i]**.

onparams -d -l *lognum*: Drop a logical-log file

Syntax:

```

▶▶ onparams --d --l lognum [ -y ]

```

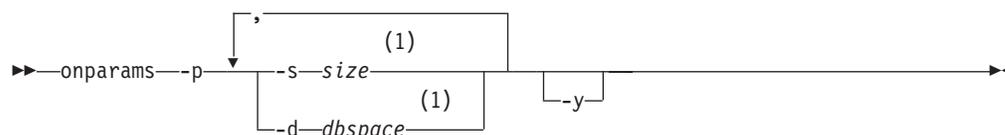
Element	Purpose	Key Considerations
-d -l lognum	Allows you to drop a logical-log file specified by the log file number	<p>Restrictions: This value must be an unsigned integer greater than or equal to 0.</p> <p>The database server requires a minimum of three logical-log files at all times. You cannot drop a log file if the database server is configured for three logical-log files. Drop log files one at a time.</p> <p>Additional Information: You can obtain the <i>lognum</i> from the number field of onstat -l. The sequence of <i>lognum</i> might be out of order.</p> <p>You can drop a log file immediately that has a status of newly Added (A). If you drop a log file that has a status of Used (U) or Free (F), the database server marks it as Deleted (D) and drops it when you take a level-0 backup of all the dbspaces.</p>
-y	Causes the database server to automatically respond yes to all prompts	None.

Use DROP LOG as the Administrative API *command* string for **onparams -d -l lognum**.

When you move logical-log files to another dbspace, use the **onparams** commands to add and drop logical-log files. See moving a logical-log file, in the chapter on managing logical-log files in the *IBM Informix Administrator's Guide*.

onparams -p: Change physical-log parameters

Syntax:



Notes:

- 1 Only one occurrence of this item is allowed

Element	Purpose	Key Considerations
-p	Changes the location or size of the physical log	You can use onparams -p with -s , -d , or both. The database server must be online; the server does not need to be restarted for the changes take effect.
-d dbspace	Changes the location of the physical log to the specified <i>dbspace</i>	<p>The space allocated for the physical log must be contiguous.</p> <p>Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i>.</p>
-s size	Changes the size (in kilobytes) of the physical log	<p>This value must be an unsigned integer greater than or equal to 200 kilobytes.</p> <p>Warning: If you move the log to a dbspace without adequate contiguous space or increase the log size beyond the available contiguous space, the operation will fail and the physical log will not change.</p>

Element	Purpose	Key Considerations
-y	Causes the database server to automatically respond yes to all prompts	None.

Backing Up After You Change the Physical-Log Size or Location

Changes to the physical log do not take effect until you restart the database server. To restart the database server immediately, execute the **onparams** command with the **-y** option.

Create a level-0 backup of the root dbspace immediately after you restart the database server. This backup is critical for proper recovery of the database server.

Changing the Size of the Physical Log and Using Non-Default Page Sizes

If you use non-default page sizes, you might need to increase the size of your physical log. If you perform many updates to non-default pages you might need a 150 to 200 percent increase of the physical log size. Some experimentation might be needed to tune the physical log. You can adjust the size of the physical log as necessary according to how frequently the filling of the physical log triggers checkpoints.

Using a Text Editor to Change the Physical-Log Size or Location

Use ALTER PLOG as the Administrative API *command* string for **onparams -p**.

onparams -b: Add a new buffer pool

Syntax:

```

▶▶ onparams -b -g size [-n-number] [-r-number] [-x-percentage]
▶ [-m-percentage]

```

Element	Purpose	Key Considerations
-b	Creates a new buffer pool	You can add a new buffer pool while the database server is running. For more information on buffer pools, see the description of the configuration parameter "BUFFERPOOL" on page 1-17 and the information on buffer pools in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
-g <i>size</i>	Specifies the size in kilobytes of the buffer pages to create	Each dbspace you create with a non-default page size must have a corresponding buffer pool with the corresponding page size. If you create a dbspace with a page size that has no buffer pool, the system will automatically create a buffer pool using the fields in the default line of the BUFFERPOOL parameter. The size of the buffer pages must be between 2 and 16 kilobytes and it must be a multiple of the default page size.
-m <i>percent</i>	Specifies the percentage of modified pages in the LRU queues at which page cleaning is no longer mandatory	Fractional values are allowed. If you do not specify this option, the percentage used is the value of the <i>lru_min_dirty</i> field as set in the default line of the BUFFERPOOL configuration parameter. For the range of values , see “The lru_min_dirty Field” on page 1-20.
-n <i>number</i>	Specifies the number of buffers in the buffer pool	If you do not specify this option, the number used is the value of <i>buffers</i> as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see “The buffers Field” on page 1-19.
-r <i>number</i>	Specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool	If you do not include this option, the number of LRU queues allocated is equal to the value of <i>lrus</i> as set in the default line of the BUFFERPOOL configuration parameter. For the range of values, see “The lrus Field” on page 1-18.
-x <i>percent</i>	Specifies the default percentage of modified pages in the LRU queues at which the queue is cleaned	Fractional values are allowed. If you do not specify this option, the percentage used is the value of <i>lru_max_dirty</i> as set in the default line of the BUFFERPOOL configuration parameter. For the range of values , see “The lru_max_dirty Field” on page 1-20.

Create a buffer pool that corresponds to the page size of the dbspace. It is recommended that you do this before you create the dbspace. You cannot reduce or increase the number of buffers in an existing buffer pool while the database server is running. You also cannot drop a buffer pool while the database server is running. You can, however, add new buffer pools with a new size while the database server is running.

Buffer pools added with the **onparams** utility are put into virtual memory, not into resident memory. Upon restart, buffer pool entries will go into resident memory depending on the amount of memory that is available.

When you add a new buffer pool with the **onparams** utility or when you add a dbspace with a different page size (with the **onspaces** utility), the settings for the BUFFERPOOL configuration parameter in the ONCONFIG file are rewritten to reflect the new entry.

Use ADD BUFFERPOOL as the Administrative API *command* string for **onparams** **-b**.

Examples of onparams Commands

The following are examples of **onparams** commands:

```
onparams -a -d rootdbs -s 1000 # adds a 1000-KB log file to rootdbs
onparams -a -d rootdbs -i      # inserts the log file after the current log
onparams -d -l 7               # drops log 7
onparams -p -d dbpace1 -s 3000 # resizes and moves physical-log to dbpace1
onparams -b -g 6 -n 3000 -r 2 -x 2.0 -m 1.0 # adds 3000 buffers of size
6K bytes each with 2 LRUS with maximum dirty of 2% and minimum dirty of 1%
```

Chapter 14. The onspaces Utility

In This Chapter

This chapter shows you how to use the following **onspaces** options:

- “ onspaces -a: Add a chunk to a dbspace or blobspace” on page 14-3
- “ onspaces -a: Add a chunk to an sbspace” on page 14-4
- “ onspaces -c -b: Create a blobspace” on page 14-5
- “ onspaces -c -d: Create a dbspace” on page 14-7
- “ onspaces -c -S: Create an sbspace” on page 14-11
- “ onspaces -c -x: Create an extspace” on page 14-16
- “ onspaces -ch: Change sbspace default specifications” on page 14-17
- “ onspaces -cl: Clean up stray smart large objects in sbspaces ” on page 14-17
- “ onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace” on page 14-18
- “ onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace” on page 14-20
- “ onspaces -f: Specify DATASKIP parameter” on page 14-21
- “ onspaces -m: Start mirroring” on page 14-21
- “ onspaces -r: Stop mirroring” on page 14-23
- “ onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace” on page 14-24
- “ onspaces -s: Change status of a mirrored chunk” on page 14-25

When you use **onspaces** or ISA to manage a storage space, the database server updates information about the space in the **oncfg_servername.servernum** file. For more information on the **oncfg*** file, refer to Appendix A, “Files That the Database Server Uses,” on page A-1.

You can specify a maximum of 2047 chunks for a storage space, and a maximum of 2047 storage spaces on the database server system. The storage spaces can be any combination of dbspaces, blobspaces, and sbspaces.

On UNIX, you must be logged in as user **root** or user **informix** to execute **onspaces**. On Windows, you must be a member of the **Informix-Admin** group.

You cannot use the **onspaces** utility on High-Availability Data Replication (HDR) secondary servers, remote standalone (RS) secondary servers, or shared disk (SD) secondary servers.

onspaces Syntax

▶▶ onspaces	-a	
	-c -b	
	-c -d	
	-c -x	
	-c -S	
	-ch	
	-cl	
	-d	
	-f	
	-m	
	-r	
	-ren	
	-s	
	-V	
	-version	

Element	Purpose	Key Considerations
-V	Displays the software version number and the serial number	See "Obtaining Utility Version Information" on page 6-1
-version	Displays the build version, host, OS, number and date, as well as the GLS version	See "Obtaining Utility Version Information" on page 6-1

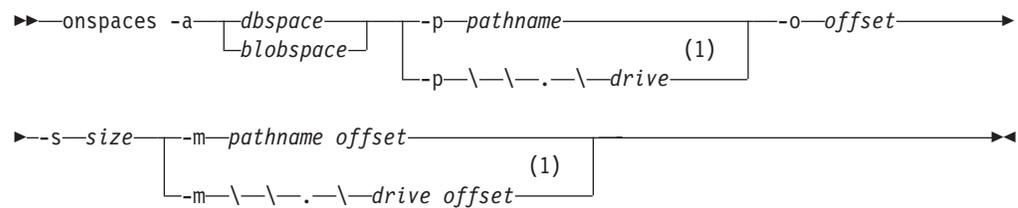
Administrative API *command string*

The following **onspaces** options have equivalent Administrative API command strings:

onspaces option	Administrative API <i>command string</i>
onspaces -a	ADD CHUNK CREATE CHUNK
onspaces -c -b	CREATE BLOBSPACE
onspaces -c -d	CREATE DBSPACE
onspaces -c -d -t	CREATE TEMPDBSPACE
onspaces -c -S	CREATE SBSPACE
onspaces -cl	CLEAN SBSPACE
onspaces -d	DROP BLOBSPACE DROP CHUNKDROP DBSPACEDROP LOGDROP SBSPACEDROP TEMPDBSPACE
onspaces -d -l	DROP LOG
onspaces -Df	SET SBSPACE ACESSTIME ONSET SBSPACE ACESSTIME OFFSET SBSPACE AVG_LO_SIZESET SBSPACE LOGGING ONSET SBSPACE LOGGING OFF
onspaces -f	SET DATASKIP ONSET DATASKIP OFF
onspaces -m	START MIRRORING
onspaces -r	STOP MIRRORING
onspaces -ren	RENAME SPACE
onspaces -s	SET CHUNK ONLINESET CHUNK OFFLINE

onspaces -a: Add a chunk to a dbspace or blobspace

Syntax:



Notes:

- 1 Windows only

Use **onspaces -a** to add a chunk to a dbspace or blobspace.

Element	Purpose	Key Considerations
-a	Indicates that a chunk is to be added	A dbspace, blobspace, or sbspace can contain up to 32,766 chunks.
<i>drive</i>	Specifies the Windows drive to allocate as unbuffered disk space. The format can be either <code>\\.\<drive></code> , where <i>drive</i> is the drive letter assigned to a disk partition, or <code>\\.\PhysicalDrive<number></code> , where <i>PhysicalDrive</i> is a constant value and <i>number</i> is the physical drive number.	For more information on allocating unbuffered disk space, see allocating raw disk space on Windows in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> . Example: <code>\\.\F:</code> For pathname syntax, see your operating-system documentation.
-m pathname offset	Specifies an optional pathname and offset to the chunk that mirrors the new chunk. Also see the entries for <i>pathname</i> and <i>offset</i> in this table.	For more information, see adding a chunk to a dbspace and adding a chunk to a blobspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-o offset	After the -a option, <i>offset</i> indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace or dbspace.	Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-p pathname	Indicates the disk partition or unbuffered device of the initial chunk of the blobspace or dbspace that you are adding. The chunk must be an existing unbuffered device or buffered file.	The chunk name can be up to 128 bytes. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): <code>/dev/rdisk/c0t3d0s4</code> UNIX example (buffered device): <code>/ix/ids9.2/db1chunk</code> Windows example: <code>c:\Ifmxdta\o1_icecream\mychunk1.dat</code> For pathname syntax, see your operating-system documentation.

Element	Purpose	Key Considerations
-s size	Indicates, in kilobytes, the size of the new blobspace or dbspace chunk	Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes.
blobspace	Names the blobspace to which you are adding a chunk	See adding a chunk to a blobspace in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> . Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> .
dbspace	Names the dbspace to which you are adding a chunk	See adding a chunk to a dbspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> . Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> .

Use **ADD CHUNK** or **CREATE CHUNK** as the Administrative API *command* string for **onspaces -a**.

onspaces -a: Add a chunk to an sbospace

Syntax:

```

▶▶ onspaces -a—blobspace—-p—pathname—-o—offset—-s—size————▶
▶
┌-m—pathname offset┐ ┌-Ms—mdsize┐ ┌-Mo—mdoffset┐ ┌-U┐

```

Use **onspaces -a** to add a chunk to an sbospace.

Element	Purpose	Key Considerations
-a	Indicates that a chunk is to be added	An sbospace can contain up to 32,766 chunks.
-m pathname offset	Specifies an optional pathname and offset to the chunk that mirrors the new chunk Also see the entries for <i>pathname</i> and <i>offset</i> in this table.	For background information, see adding a chunk to an sbospace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-Mo mdooffset	Indicates, in kilobytes, the offset into the disk partition or into the device where metadata should be stored	Value can be an integer between 0 and the chunk size. You cannot specify an offset that causes the end of the metadata space to be past the end of the chunk. For background information, see sizing sbospace metadata, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-Ms mdsiz	Specifies the size, in kilobytes, of the metadata area allocated in the initial chunk. The remainder is user-data space	Value can be an integer between 0 and the chunk size. For background information, see sizing sbospace metadata, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
-o <i>offset</i>	After the -a option, <i>offset</i> indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the new blobspace or dbspace.	Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-p <i>pathname</i>	Indicates the disk partition or unbuffered device of the initial chunk of the sbspace that you are creating The chunk must be an existing unbuffered device or buffered file.	The chunk name can be up to 128 bytes. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.
-U	Specifies that the entire chunk should be used to store user data	The -M and -U options are mutually exclusive. For background information, see adding a chunk to an sbspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-s <i>size</i>	Indicates, in kilobytes, the size of the new sbspace chunk	Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes.
<i>sbspace</i>	Names the sbspace to which you are adding a chunk	See adding a chunk to an sbspace in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> . Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> .

Use ADD CHUNK or CREATE CHUNK as the Administrative API *command* string for **onspaces -a**.

onspaces -c -b: Create a blobspace

Syntax:

```

▶▶ onspaces -c -b blobspace -g pageunit -p pathname -o offset -s size
      └─p \ \ \ . \ drive (1)
      └─p \ \ \ . \ drive

```



```

▶ ┌─m pathname offset
  │ (1)
  └─m \ \ \ . \ drive offset

```

Notes:

- 1 Windows Only

Use **onspaces -c -b** to create a blobspace.

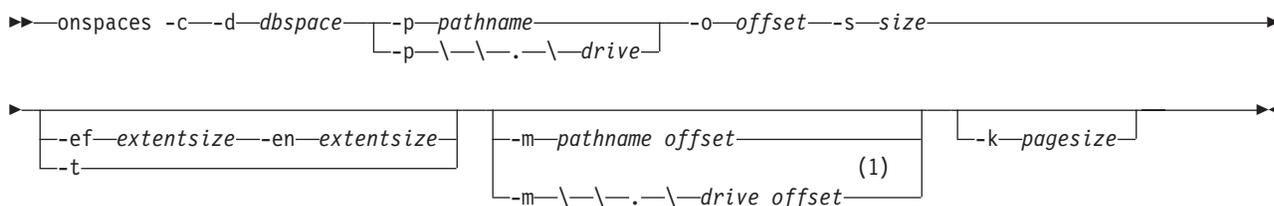
Element	Purpose	Key Considerations
-b blobspace	Names the blobspace to be created	<p>The blobspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the \$ character.</p> <p>For more information, see creating a blobspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>. The syntax must conform to the Identifier segment. For more information, see the <i>IBM Informix Guide to SQL: Syntax</i>.</p>
-c	<p>Creates a dbspace, blobspace, sbspace, or extspace</p> <p>You can create up to 2047 storage spaces of any type.</p>	<p>After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one.</p> <p>For more information, see creating a dbspace, blobspace, or extspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
drive	<p>Specifies the Windows drive to allocate as unbuffered disk space</p> <p>The format can be either <code>\\.\<drive></code>, where <i>drive</i> is the drive letter assigned to a disk partition, or <code>\\.\PhysicalDrive<number></code>, where <i>PhysicalDrive</i> is a constant value and <i>number</i> is the physical drive number.</p>	<p>For information on allocating unbuffered disk space, see allocating unbuffered disk space on Windows in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>. Examples:</p> <pre>\\.\F: \\.\PhysicalDrive2</pre> <p>For pathname syntax, see your operating-system documentation.</p>
-g pageunit	Specifies the blobspace blobpage size in terms of <i>page_unit</i> , the number of disk pages per blobpage	<p>Unsigned integer. Value must be greater than 0.</p> <p>For more information, see blobpage size considerations, in the chapter on I/O Activity in the <i>IBM Informix Performance Guide</i>.</p>
-m pathname offset	<p>Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new blobspace or dbspace</p> <p>Also see the entries for -p pathname and -o offset in this table.</p>	For more information, see creating a dbspace or a blobspace in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-o offset	Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace, dbspace, or sbspace	<p>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform.</p> <p>For more information, see allocating raw disk space, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>

Element	Purpose	Key Considerations
-p pathname	Indicates the disk partition or device of the initial chunk of the blobspace or dbspace that you are creating	The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): /dev/rdisk/c0t3d0s4 UNIX example (buffered device): /ix/ids9.2/db1chunk Windows example:c:\lfmxdata\ol_icecream\mychunk1.dat For pathname syntax, see your operating-system documentation.
-s size	Indicates, in kilobytes, the size of the initial chunk of the new blobspace or dbspace	Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 terabytes, depending on the platform.

Use CREATE BLOBSPACE as the Administrative API *command* string for **onspaces -c -b**.

onspaces -c -d: Create a dbspace

Syntax:



Notes:

- 1 Windows Only

Use **onspaces -c -d** to create a dbspace or a temporary dbspace.

Element	Purpose	Key Considerations
-c	Creates a dbspace You can create up to 2047 storage spaces of any type.	After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one. For more information, see creating a dbspace in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
<i>drive</i>	<p>Specifies the Windows drive to allocate as unbuffered disk space</p> <p>The format can be either <code>\\.\<drive></code>, where <i>drive</i> is the drive letter assigned to a disk partition, or <code>\\.\PhysicalDrive<number></code>, where <i>PhysicalDrive</i> is a constant value and <i>number</i> is the physical drive number.</p>	<p>For information on allocating unbuffered disk space, see allocating unbuffered disk space on Windows in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p> <p>Examples: <code>\\.\F:</code> <code>\\.\PhysicalDrive2</code></p> <p>For pathname syntax, see your operating-system documentation.</p>
-d <i>dbspace</i>	Names the dbspace to be created	<p>The dbspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the \$ character.</p> <p>For more information, see creating a dbspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>. The syntax must conform to the Identifier segment. For more information, see the <i>IBM Informix Guide to SQL: Syntax</i>.</p>
-en <i>extentsize</i>	Indicates, in kilobytes, the size of the first extent for the tblspace	<p>The minimum, and default, size of the first extent for the tblspace of a non-root dbspace is equivalent to 50 dbspace pages, specified in K. For example: 100 KB for a 2 KB page size dbspace, 200 KB for a 4 KB page size dbspace, 400 KB for an 8 KB page size dbspace.</p> <p>The maximum size of a tblspace extent is 1048575 pages minus the space needed for any system objects. On a 2 KB pagesize system this would evaluate to approximately 2 GB.</p> <p>For more information, see specifying first and next extent size in the chapter on managing dbspaces in the <i>IBM Informix Administrator's Guide</i>.</p>
-ef <i>extentsize</i>	Indicates, in kilobytes, the size of the next extents in the tblspace	<p>The minimum size of the next extents for the tblspace of a non-root dbspace is equivalent to 4 dbspace pages, specified in K. For example: 8 KB for a 2 KB page size dbspace, 16 KB for a 4 KB page size dbspace, 32 KB for an 8 KB page size dbspace.</p> <p>The default size for a next extent is 50 dbspace pages.</p> <p>The maximum size of a tblspace extent is 1048572 pages. On a 2 KB pagesize system this would evaluate to approximately 2 GB.</p> <p>If there is not enough space for a next extent in the primary chunk, the extent is allocated from another chunk. If the specified space is not available, the closest available space is allocated.</p> <p>For more information, see specifying first and next extent size in the chapter on managing dbspaces in the <i>IBM Informix Administrator's Guide</i>.</p>

Element	Purpose	Key Considerations
-k <i>pagesize</i>	<p>Indicates in kilobytes, the non-default page size for the new dbspace.</p> <p>For systems with sufficient storage, performance advantages of a larger page size can include the following:</p> <ul style="list-style-type: none"> • Reduced depth of B-tree indexes, even for smaller index keys • You can group on the same page long rows that currently span multiple pages of the default page size • Checkpoint time is typically reduced with larger pages • You can define a different page size for temporary tables, so that they have a separate buffer pool. 	<p>The page size must be between 2KB and 16KB and must be a multiple of the default page size. For example, if the default page size is 2KB, then <i>pagesize</i> can be 2, 4, 6, 8, 10, 12, 14, or 16. If the default page size is 4KB (Windows), then <i>pagesize</i> can be 4, 8, 12, or 16.</p> <p>For more information, see creating a dbspace with a non-default page size in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-m <i>pathname offset</i>	<p>Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new dbspace</p> <p>Also see the entries for -p <i>pathname</i> and -o <i>offset</i> in this table.</p>	<p>For more information, see creating a dbspace in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-o <i>offset</i>	<p>Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new dbspace</p>	<p>Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The offset must be a multiple of the page size. The maximum offset is 2 or 4 gigabytes, depending on the platform.</p> <p>For more information, see allocating raw disk space, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-p <i>pathname</i>	<p>Indicates the disk partition or device of the initial chunk of the dbspace that you are creating</p>	<p>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. UNIX example (unbuffered device): <code>/dev/rdisk/c0t3d0s4</code> UNIX example (buffered device): <code>/ix/ids9.2/db1chunkWindows</code> example: <code>c:\Ifmxdata\ol_icecream\mychunk1.dat</code></p> <p>For pathname syntax, see your operating-system documentation.</p>
-s <i>size</i>	<p>Indicates, in kilobytes, the size of the initial chunk of the new dbspace</p>	<p>Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size.</p> <p>The maximum chunk size is 2 or 4 terabytes, depending on the platform.</p>

Element	Purpose	Key Considerations
-t	Creates a temporary dbspace for storage of temporary tables	<p>You cannot mirror a temporary dbspace. You cannot specify the first and next extent sizes for the tblspace tblspace of a temporary dbspace.</p> <p>For more information, see temporary dbspaces, in the chapter on data storage, and creating a temporary dbspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>

Use CREATE DBSPACE as the Administrative API *command* string for **onspaces -c -d**.

Creating a Temporary Dbspace with the -t Option

When you create a temporary dbspace with **onspaces**, the database server uses the newly created temporary dbspace, after you perform the following steps:

1. Add the name of the new temporary dbspace to your list of temporary dbspaces in the DBSPACETEMP configuration parameter, the DBSPACETEMP environment variable, or both.
2. Restart the database server.

Use CREATE TEMPDBSPACE as the Administrative API *command* string for **onspaces -c -d -t**.

Specifying First and Next Extent Size for the tblspace tblspace

You cannot specify the first and next extent of a temporary dbspace. The extent size for temporary dbspaces is 100 kilobytes for a 2 kilobyte page system or 200 kilobytes for a 4 kilobyte page system.

To specify the first and next extent sizes of a root **tblspace** **tblspace**, use the TBLTBLFIRST and TBLTBLNEXT configuration parameters before you create the root dbspace the first time you start the database server.

Specifying a Non-Default Page Size with the Same Size as the Buffer Pool

When you create a dbspace with a non-default page size, you must also create a buffer pool specific to that page size. It is recommended that you create the buffer pool before you create the dbspace. Use the **onparams** utility to create a buffer pool. For more information, see “onparams -b: Add a new buffer pool” on page 13-4.

When you add a dbspace with a different page size with the **onspaces** utility or you add a new buffer pool (with the **onparams** utility), a new BUFFERPOOL line is appended in the BUFFERPOOL configuration parameter in the ONCONFIG file to reflect the new entry and it is rewritten to disk.

Notes:

1. You cannot change the page size of a dbspace after you create it.
2. You cannot store logical or physical logs in a dbspace that is not the default platform page size.

- If a dbspace is created when a buffer pool with that page size does not exist, Dynamic Server creates a buffer pool using the values of the fields of the default line of the BUFFERPOOL parameter. You cannot have multiple buffer pools with the same page size.

onspaces -c -S: Create an sbspace

Syntax:

```

▶ onspaces -c -S sbspace [-t] -p pathname -o offset -s size
▶ [-m pathname offset] [-Ms mdsize] [-Mo mdoffset]
▶ [-Df default list]

```

Use **onspaces -c -S** to create a sbspace or a temporary sbspace.

Element	Purpose	Key Considerations
-S sbspace	Names the sbspace to be created	The sbspace name must be unique and must not exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the \$ character. Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> .
-c	Creates an sbspace You can create up to 32767 storage spaces of any type.	None.
-m pathname offset	Specifies an optional pathname and offset to the chunk that mirrors the initial chunk of the new sbspace Also see the entries for -p pathname and -o offset in this table.	For more information, see sbspaces in the chapter on data storage, and creating an sbspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-Mo mdoffset	Indicates, in kilobytes, the offset into the disk partition or into the device where metadata will be stored.	Restrictions: Value can be an integer between 0 and the chunk size. You cannot specify an offset that causes the end of the metadata space to be past the end of the chunk. References: For more information, see sizing sbspace metadata, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-Ms mdsize	Specifies the size, in kilobytes, of the metadata area allocated in the initial chunk The remainder is user-data space.	Restrictions: Value can be an integer between 0 and the chunk size.

Element	Purpose	Key Considerations
-o <i>offset</i>	Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the sbspace	<p>Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 4 terabytes for systems with a two-kilobyte page size and 8 terabytes for systems with a four-kilobyte page size.</p> <p>References: For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-p <i>pathname</i>	Indicates the disk partition or unbuffered device of the initial chunk of the sbspace	<p>The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server.</p> <p>References: For pathname syntax, see your operating-system documentation.</p>
-s <i>size</i>	Indicates, in kilobytes, the size of the initial chunk of the new sbspace	<p>Restrictions: Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size.</p> <p>The maximum chunk size is 2 or 4 gigabytes, depending on the platform.</p>
-t	Creates a temporary sbspace for storage of temporary smart large objects. You can specify the size and offset of the metadata area	<p>Restrictions: You cannot mirror a temporary sbspace. You can specify any -Df option, except the LOGGING=ON option, which has no effect.</p> <p>References: For more information, see "Creating a Temporary Sbspace with the -t Option."</p>
-Df <i>default list</i>	Lists default specifications for smart large objects stored in the sbspace	<p>Restrictions: Tags are separated by commas. If a tag is not present, system defaults take precedence. The list must be enclosed in double quotation marks (") on the command line.</p> <p>References: For a list of tags and their parameters, see Table 14-1 on page 14-13.</p>

Creating a Temporary Sbspace with the -t Option

This example creates a temporary sbspace of 1000 kilobytes:

```
onspaces -c -S tempsbsp -t -p ./tempsbsp -o 0 -s 1000
```

You can optionally specify the name of the temporary sbspace in the SBSPACETEMP configuration parameter. Restart the database server so that it can use the temporary sbspace.

Creating an Sbspace with the -Df option

When you create an sbspace with the optional **-Df** option, you can specify several default specifications that affect the behavior of the smart large objects stored in the sbspace. The default specifications must be expressed as a list separated by

commas. The list need not contain all of the tags. The list of tags must be enclosed in double quotation marks (“”). The table in Table 14-1 describes the tags and their default values.

The four levels of inheritance for sbspace characteristics are system, sbspace, column, and smart large objects. For more information, see smart large objects in the chapter on where data is stored in the *IBM Informix Administrator's Guide*.

Table 14-1. -Df Default Specifications

Tag	Values	Default	Description
ACCESSTIME	ON or OFF	OFF	<p>When set to ON, the database server tracks the time of access to all smart large objects stored in the sbspace.</p> <p>For information about altering storage characteristics of smart large objects, see the <i>IBM Informix DataBlade API Programmer's Guide</i>.</p>
AVG_LO_SIZE	Windows: 4 to 2**31 UNIX: 2 to 2**31	8	<p>Specifies the average size, in kilobytes, of the smart large object stored in the sbspace</p> <p>The database server uses this value to calculate the size of the metadata area. Do not specify AVG_LO_SIZE and -Ms together. You can specify AVG_LO_SIZE and the metadata offset (-Mo) together.</p> <p>If the size of the smart large object exceeds 2**31, specify 2**31. If the size of the smart large object is less than 2 on UNIX or less than 4 in Windows, specify 2 or 4.</p> <p>Error 131 is returned if you run out of space in the metadata and reserved areas in the sbspace. To allocate additional chunks to the sbspace that consist of metadata area only, use the -Ms option instead.</p> <p>For more information, see creating smart large objects, in the chapter on managing data on disk in the <i>IBM Informix Administrator's Guide</i>.</p>
BUFFERING	ON or OFF	ON	<p>Specifies the buffering mode of smart large objects stored in the sbspace</p> <p>If set to ON, the database server uses the buffer pool in the resident portion of shared memory for smart-large-object I/O operations. If set to OFF, the database server uses light I/O buffers in the virtual portion of shared memory (lightweight I/O operations).</p> <p>BUFFERING = OFF is incompatible with LOCK_MODE = RANGE and creates a conflict</p> <p>For more information, see lightweight I/O, in the chapter on configuration effects on memory in the <i>IBM Informix Performance Guide</i>.</p>

Table 14-1. -Df Default Specifications (continued)

Tag	Values	Default	Description
LOCK_MODE	RANGE or BLOB	BLOB	<p>Specifies the locking mode of smart large objects stored in the sbspace</p> <p>If set to RANGE, only a range of bytes in the smart large object is locked. If set to BLOB, the entire smart large object is locked.</p> <p>LOCK_MODE = RANGE is incompatible with BUFFERING = OFF and creates a conflict.</p> <p>For more information, see smart large objects, in the chapter on locking in the <i>IBM Informix Performance Guide</i>.</p>
LOGGING	ON or OFF	OFF	<p>Specifies the logging status of smart large objects stored in the sbspace</p> <p>If set to ON, the database server logs changes to the user data area of the sbspace. When you turn on logging for an sbspace, take a level-0 backup of the sbspace.</p> <p>When you turn off logging, the following message displays: You are turning off smart large object logging.</p> <p>For more information, see smart large objects, in the chapters on data storage and logging in the <i>IBM Informix Administrator's Guide</i>. For information about onspaces -ch messages, see Appendix E, "Error Messages," on page E-1.</p>
EXTENT_SIZE	4 to 2**31	None	<p>Specifies the size, in kilobytes, of the first allocation of disk space for smart large objects stored in the sbspace when you create the table</p> <p>Let the system select the EXTENT_SIZE value. To reduce the number of extents in a smart large object, use mi_lo_specset_estbytes (DataBlade API) or ifx_lo_specset_estbytes (Informix ESQ/C) to hint to the system the total size of the smart large object. The system attempts to allocate a single extent for the smart large object.</p> <p>For more information, see smart large objects, in the chapter on where data is stored in the <i>IBM Informix Administrator's Guide</i>. For information about altering storage characteristics of smart large objects, see the <i>IBM Informix DataBlade API Programmer's Guide</i> or the <i>IBM Informix ESQ/C Programmer's Manual</i>.</p>
MIN_EXT_SIZE	2 to 2**31	Windows: 4UNIX: 2	<p>Specifies the minimum amount of space, in kilobytes, to allocate for each smart large object</p> <p>The following message displays: Changing the sbspace minimum extent size: old value <i>value1</i> new value <i>value2</i>.</p> <p>For information about tuning this value, see smart large objects, in the chapter on configuration effects on I/O utilization in the <i>IBM Informix Performance Guide</i>. For information about onspaces -ch messages, see Appendix E, "Error Messages," on page E-1.</p>

Table 14-1. -Df Default Specifications (continued)

Tag	Values	Default	Description
NEXT_SIZE	4 to 2**31	None	<p>Specifies the extent size, in kilobytes, of the next allocation of disk space for smart large objects when the initial extent in the sbsp becomes full. Let the system select the NEXT_SIZE value. To reduce the number of extents in a smart large object, use mi_lo_specset_estbytes or ifx_lo_specset_estbytes to hint to the system the total size of the smart large object. The system attempts to allocate a single extent for the smart large object.</p> <p>For more information, see smart large objects, in the chapter on where data is stored in the <i>IBM Informix Administrator's Guide</i>. For information about obtaining the size of smart large objects, see the <i>IBM Informix DataBlade API Programmer's Guide</i> or the <i>IBM Informix ESQ/C Programmer's Manual</i>.</p>

This example creates a 20-megabyte mirrored sbsp, **eg_sbsp**, with the following specifications:

- An offset of 500 kilobytes for the primary and mirror chunks
- An offset of 200 kilobytes for the metadata area
- An average expected smart-large-object size of 32 kilobytes
- Log changes to the smart large objects in the user-data area of the sbsp

UNIX Only:

```
% onspaces -c -S eg_sbsp -p /dev/raw_dev1 -o 500 -s 20000
-m /dev/raw_dev2 500 -Mo 200 -Df "AVG_LO_SIZE=32,LOGGING=ON"
```

Changing the -Df Settings

As the database server administrator, you can override or change the **-Df** default settings in one of the following ways:

- To change the default settings for an sbsp, use the **onspaces -ch** option. For more information, refer to “onspaces -ch: Change sbsp default specifications” on page 14-17.
- To override the following **-Df** default settings for a specific table, use the SQL statements CREATE TABLE or ALTER TABLE:
 - LOGGING
 - ACESSTIME
 - EXTENT_SIZE
 - NEXT_SIZE

For more information on the ALTER TABLE and CREATE TABLE statements, see the *IBM Informix Guide to SQL: Syntax*.

The programmer can override these **-Df** default settings with DataBlade API and Informix ESQ/C functions. For information about altering storage characteristics of smart large objects, see the *IBM Informix DataBlade API Programmer's Guide* and the *IBM Informix ESQ/C Programmer's Manual*.

Using the onspaces -g Option

The **onspaces -g** option is not used for sbspaces. The database server uses a different method to determine the number of pages to transfer in an I/O operation for sbspaces than for blobspaces. The database server can automatically determine the block size to transfer in an I/O operation for smart large objects. For more information, see sbspace extent sizes in the chapter on I/O activity in your *IBM Informix Performance Guide*.

Use CREATE SBSPACE as the Administrative API *command* string for **onspaces -c -S**.

onspaces -c -x: Create an extspace

Syntax:

►► onspaces -c -x *extspace* [-l *location*] [-o *offset*] [-s *size*] ◀◀

Use **onspaces -c -x** to create an extspace.

Element	Purpose	Key Considerations
-c	Creates a dbspace, blobspace, sbspace, or extspace You can create up to 2047 storage spaces of any type.	After you create a storage space, you must back up both this storage space and the root dbspace. If you create a storage space with the same name as a deleted storage space, perform another level-0 backup to ensure that future restores do not confuse the new storage space with the old one. For more information, see creating a dbspace, blobspace, or extspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-l <i>location</i>	Specifies the location of the extspace The access method determines the format of this string.	Restrictions: String. Value must not be longer than 255 bytes. For more information, see creating an extspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-o <i>offset</i>	Indicates, in kilobytes, the offset into the disk partition or into the device to reach the initial chunk of the new blobspace, dbspace, or sbspace	Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 2 or 4 gigabytes, depending on the platform. For more information, see allocating raw disk space, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-s <i>size</i>	Indicates, in kilobytes, the size of the initial chunk of the new blobspace or dbspace	Restrictions: Unsigned integer. The size must be equal to or greater than 1000 kilobytes and a multiple of the page size. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum chunk size is 2 or 4 terabytes, depending on the platform.

Element	Purpose	Key Considerations
<i>-x extspace</i>	Names the extspace to be created	<p>Restrictions: Extspace names can be up to 128 bytes. They must be unique, begin with a letter or underscore, and contain only letters, digits, underscores, or \$ characters.</p> <p>For more information, see <i>extspaces</i>, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>

onspaces -ch: Change sbspace default specifications

Syntax:

►► onspaces -ch *sbspace* *-Df default list* ◀◀

Use **onspaces -ch** to change the default specifications of a sbspace.

Element	Purpose	Key Considerations
-ch	Indicates that one or more sbspace default specifications are to be changed	None.
<i>sbspace</i>	Names the sbspace for which to change the default specifications	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see <i>changing default specifications of an sbspace with onspaces</i> in the <i>IBM Informix Performance Guide</i> .
-Df default list	Lists new default specifications for smart large objects stored in the sbspace	<p>Tags are separated by commas. If a tag is not present, system defaults take precedence. The list must be enclosed in double quotation marks (") on the command line.</p> <p>For a list of tags and their parameters, see Table 14-1 on page 14-13.</p>

You can change any of the **-Df** tags with the **onspaces -ch** option. The database server applies the change to each smart large object that was created prior to changing the default specification.

For example, to turn off logging for the sbspace that you created in "Creating an Sbspace with the -Df option" on page 14-12, use the following command:

```
onspaces -ch eg_sbsp -Df "LOGGING=OFF"
```

Note: After you turn on logging for an sbspace, take a level-0 backup of the sbspace to create a point from which to recover.

onspaces -cl: Clean up stray smart large objects in sbspaces

Syntax:

►► onspaces -cl *sbspace* ◀◀

Use **onspaces -cl** to clean up stray smart large objects in sbspaces.

Syntax:

```
onspaces -cl sbspace
```

Element	Purpose	Key Considerations
-cl	Cleans up stray smart large objects in an sbspace	To find any stray smart large objects, use the oncheck -pS command when no users are connected to the database server. The smart large objects with a reference count of 0 are stray objects.
<i>sbspace</i>	Names the sbspace to be cleaned up	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> .

During normal operation, no unreferenced (stray) smart large objects should exist. When you delete a smart large object, the space is released. If the database server fails or runs out of system memory while you are deleting a smart large object, the smart large object might remain as a stray object.

The following is an example of the **onspaces -cl** command:

```
onspaces -cl myspace
```

The best way to find the reference count for a smart large object is to call the **mi_lo_stat** or **ifx_lo_stat** functions from a C program. Although the **mi_lo_increfcount** and **mi_lo_decrefcount** functions return the reference count, they increment or decrement the reference count. For more information on these functions, see the *IBM Informix DataBlade API Function Reference*.

Use CLEAN SBSPACE as the Administrative API *command* string for **onspaces -cl**.

onspaces -d: Drop a chunk in a dbspace, blobspace, or sbspace

Syntax:

```
onspaces -d [dbspace | blobspace | sbspace] -p pathname -o offset [-y] [-f]
```

Use **onspaces -d** to drop a chunk in a dbspace, blobspace, or sbspace.

Use DROP CHUNK as the Administrative API *command* string for **onspaces -d**.

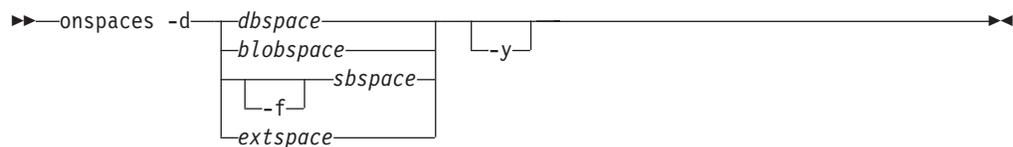
Element	Purpose	Key Considerations
-d	Drops a chunk	<p>You can drop a chunk from a dbspace, temporary dbspace, or sbspace when the database server is online or quiescent. For more information, see the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p> <p>You can drop a chunk from a blobspace only when the database server is in quiescent mode.</p>

Element	Purpose	Key Considerations
-f	Drops an sbspace chunk that contains user data but <i>no</i> metadata. If the chunk contains metadata for the sbspace, you must drop the entire sbspace.	Use the -f option with sbspaces only. If you omit the -f option, you cannot drop an sbspace that contains data. For more information, see dropping a chunk from an sbspace with onspaces , in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-o offset	Indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the dbspace, blobspace, or sbspace that you are dropping	Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-p pathname	Indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that you are dropping	The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.
-y	Causes the database server to automatically respond yes to all prompts	None.
blobpace	Names the blobspace from which the chunk is dropped	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see dropping a chunk from a blobspace, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
dbspace	Names the dbspace from which the chunk is dropped	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see dropping a chunk from a dbspace with onspaces , in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
sbspace	Names the sbspace from which the chunk is dropped	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see dropping a chunk from a sbspace with onspaces , in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .

Important: You must specify a pathname to indicate to the database server that you are dropping a chunk.

onspaces -d: Drop a blobspace, dbspace, extspace, or sbspace

Syntax:



Use **onspaces -d** to drop a dbspace, blobspace, sbspace, or extspace.

Use DROP BLOBSPACE, DROP DBSPACE, DROP SBSPACE, or DROP TEMPDBSPACE as the Administrative API *command* strings for **onspaces -d**.

Element	Purpose	Key Considerations
-d	Indicates that a dbspace, blobspace, sbspace, or extspace is to be dropped	<p>You can drop a dbspace, blobspace, sbspace, or extspace while the database server is online or in quiescent mode. After you drop a storage space, you must back it up to ensure that the sysutils database and the reserved pages are up-to-date.</p> <p>Execute oncheck -pe to verify that no table is currently storing data in the dbspace, blobspace, or sbspace.</p> <p>For more information, see dropping a storage space, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-y	Causes the database server to automatically respond yes to all prompts	None.
-f	Drops an sbspace that contains user data and metadata	<p>You must use the -f (force) option to drop an sbspace that contains data.</p> <p>Restriction: Use the -f option with sbspaces only.</p> <p>Warning: If you use the -f option, the tables in the database server might have dead pointers to the smart large objects that were deleted with this option.</p> <p>For more information, see dropping a chunk from an sbspace with onspaces, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
<i>blobspace</i>	Names the blobspace to be dropped	Before you drop a blobspace, drop all tables that include a TEXT or BYTE column that references the blobspace.
<i>dbspace</i>	Names the dbspace to be dropped	Before you drop a dbspace, drop all databases and tables that you previously created in the dbspace.
<i>extspace</i>	Names the extspace to be dropped	You cannot drop an extspace if it is associated with an existing table or index.
<i>sbspace</i>	Names the sbspace to be dropped	Before you drop an sbspace, drop all tables that include a BLOB or CLOB column that references the sbspace.

Important: Do not specify a pathname when you drop these storage spaces.

onspaces -f: Specify DATASKIP parameter

Syntax:

```

▶▶ onspaces -f {OFF | ON} [dbspace-list] [-y]

```

Use **onspaces -f** to specify the value of the DATASKIP configuration parameter. Use SET DATASKIP ON and SET DATASKIP OFF as the Administrative API *command* strings **onspaces -f**.

Element	Purpose	Key Considerations
-f	Indicates to the database server that you want to change the DATASKIP default for specified dbspaces or all dbspaces	All changes in the DATASKIP status are recorded in the message log.
-y	Causes the database server to automatically respond yes to all prompts	None.
<i>dbspace-list</i>	Specifies the name of one or more dbspaces for which DATASKIP will be turned ON or OFF	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see "DATASKIP" on page 1-26 and the <i>IBM Informix Performance Guide</i> .
OFF	Turns off DATASKIP	If you use OFF without <i>dbspace-list</i> , DATASKIP is turned off for all fragments. If you use OFF with <i>dbspace-list</i> , only the specified fragments are set with DATASKIP off.
ON	Turns on DATASKIP	If you use ON without <i>dbspace-list</i> , DATASKIP is turned on for all fragments. If you use ON with <i>dbspace-list</i> , only the specified fragments are set with DATASKIP on.

The **onspaces** utility lets you specify DATASKIP on a dbspace level or across all dbspaces.

onspaces -m: Start mirroring

Syntax:

```

▶▶ onspaces -m {dbspace | blobspace | sbpace}

```

```

▶▶ {
    -p pathname -o offset -m pathname offset
    -f filename
} [-y]

```

Use **onspaces -m** to start mirroring for a dbspace, blobspace, or sbpace. Use START MIRRORING as the Administrative API *command* string for **onspaces -m**.

Element	Purpose	Key Considerations
-f filename	Indicates that chunk-location information is in a file named <i>filename</i>	The file must be a buffered file that already exists. The pathname must conform to the operating-system-specific rules for pathnames. For more information, see “Using a File to Specify Chunk-Location Information with the -f Option” on page 14-23.
-m	Adds mirroring for an existing dbspace, blobspace, or sbspace	User-data chunks in a mirrored sbspace need not be mirrored. The mirrored chunks should be on a different disk. You must mirror all the chunks at the same time.
-m pathname offset	The second time that <i>pathname</i> occurs in the syntax diagram, it indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that performs the mirroring. The second time <i>offset</i> appears in the syntax diagram, it indicates the offset to reach the mirrored chunk of the newly mirrored dbspace, blobspace, or sbspace. Also see the entries for <i>pathname</i> and <i>offset</i> in this table.	None.
-o offset	The first time that <i>offset</i> occurs in the syntax diagram, it indicates, in kilobytes, the offset into the disk partition or into the unbuffered device to reach the initial chunk of the newly mirrored dbspace, blobspace, or sbspace.	Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The maximum offset is 4 terabytes. For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-p pathname	The first time <i>pathname</i> occurs in the syntax diagram, it indicates the disk partition or unbuffered device of the initial chunk of the dbspace, blobspace, or sbspace that you want to mirror.	The chunk must be an existing unbuffered device or buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server. For pathname syntax, see your operating-system documentation.
-y	Causes the database server to automatically respond yes to all prompts	None.
blobspace	Names the blobspace that you want to mirror	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .
dbspace	Names the dbspace that you want to mirror	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
<i>sbspace</i>	Names the sbspace that you want to mirror	Syntax must conform to the Identifier segment; see <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .

Using a File to Specify Chunk-Location Information with the -f Option

You can create a file that contains the chunk-location information. Then, when you execute **onspaces**, use the **-f** option to indicate to the database server that this information is in a file whose name you specify in *filename*.

The contents of the file should conform to the following format, with options separated by spaces and each set of primary and mirror chunks on separate lines:

```
primary_chunk_path offset mirror_chunk_path offset
```

If the dbspace that you are mirroring contains multiple chunks, you must specify a mirror chunk for each of the primary chunks in the dbspace that you want to mirror. For an example that enables mirroring for a multichunk dbspace, see starting mirroring for unmirrored dbspaces with **onspaces** in the chapter on using mirroring in the *IBM Informix Administrator's Guide*.

onspaces -r: Stop mirroring

Syntax:

```

▶▶ onspaces -r [dbspace] [-blobspace] [-sbspace] [-y]

```

Use **onspaces -r** to end mirroring for a dbspace, blobspace, or sbspace. Use STOP MIRRORING as the Administrative API *command* string for **onspaces -r**.

Element	Purpose	Key Considerations
-r	Indicates to the database server that mirroring should be ended for an existing dbspace, blobspace, or sbspace	For background information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .
-y	Causes the database server to respond yes to all prompts automatically	None.
<i>blobspace</i>	Names the blobspace for which you want to end mirroring	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .
<i>dbspace</i>	Names the dbspace for which you want to end mirroring.	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
<i>sbspace</i>	Names the sbspace for which you want to end mirroring	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see the chapter on using mirroring in the <i>IBM Informix Administrator's Guide</i> .

onspaces -ren: Rename a dbspace, blobspace, sbspace, or extspace

Syntax:

```

▶▶ onspaces -ren dbspace | blobspace | sbspace | extspace -n name

```

Use **onspaces -ren** to rename a dbspace, blobspace, sbspace, or extspace. Use **RENAME SPACE** as the Administrative API *command* string for **onspaces -ren**.

Element	Purpose	Key Considerations
-ren	Causes the database server to rename the specified blobspace, dbspace, extspace, or sbspace	Restrictions: You can rename a blobspace, dbspace, extspace, or sbspace when the database server is in quiescent mode. For more information, see the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
-n name	Specifies the new name for the blobspace, dbspace, extspace, or sbspace	Restrictions: The blobspace, dbspace, external space, or sbspace name must be unique and cannot exceed 128 bytes. It must begin with a letter or underscore and must contain only letters, numbers, underscores, or the \$ character. For more information, see the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> . The syntax must conform to the Identifier segment. For more information, see the <i>IBM Informix Guide to SQL: Syntax</i> .
<i>blobspace</i>	Names the blobspace to be renamed	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see renaming spaces, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
<i>dbspace</i>	Names the dbspace to be renamed	Restrictions: You cannot rename a critical dbspace, such as the root dbspace or a dbspace that contains physical logs. Additional Information: If you rename dbspaces that are included in the DATASKIP list, update the DATASKIP configuration parameter with the new names using the onspaces -f command. Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see renaming spaces, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .
<i>extspace</i>	Names the extspace to be renamed	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see renaming spaces, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .

Element	Purpose	Key Considerations
<i>sbspace</i>	Names the sbspace to be renamed	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see renaming spaces, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i> .

Renaming a Dbspace, Blobspace, Sbspace, or Extspace when Enterprise Replication Is Active

You can rename a space (dbspace, blobspace, sbspace, or extspace) when Enterprise Replication is active. When you put the database server into quiescent mode to rename the space, Enterprise Replication will be disconnected. You can then rename the space. The servers will resynchronize after you put the database server into online mode. If you want to rename the same space on another server, you must put that server into quiescent mode and rename the space separately. No enforced relationship is propagated between renamed spaces on different ER servers; the same tables can be in different spaces.

If the Enterprise Replication server also participates in High-Availability Data Replication (HDR), you can rename the dbspace on the primary server and it will be automatically propagate to the secondary server. (The secondary server cannot participate in Enterprise Replication.)

Performing an Archive after Renaming a Space

After renaming any space (except extspaces or temporary spaces), perform a level-0 archive of the renamed space and the root dbspace. This will ensure that you can restore the spaces to a state including or following the rename dbspace operation. It is also necessary prior to performing any other type of archive.

onspaces -s: Change status of a mirrored chunk

Syntax:

```

▶▶ onspaces -s dbspace -p-pathname -o-offset -D _0 _y
-s blobspace sbspace

```

Use **onspaces -s** to change the status of a mirrored chunk in a dbspace, a non-primary chunk within a noncritical dbspace, a blobspace or an sbspace. Use SET CHUNK ONLINE or SET CHUNK OFFLINE as the Administrative API *command* string for **onspaces -s**.

Element	Purpose	Key Considerations
-D	Indicates that you want to take the chunk down	None.

Element	Purpose	Key Considerations
-o <i>offset</i>	Indicates, in kilobytes, the offset into the disk partition or unbuffered device to reach the chunk	<p>Restrictions: Unsigned integer. The starting offset must be equal to or greater than 0. The starting offset plus the chunk size cannot exceed the maximum chunk size. The offset must be a multiple of the page size.</p> <p>The maximum offset is 4 terabytes.</p> <p>For more information, see allocating raw disk space on UNIX, in the chapter on managing disk space in the <i>IBM Informix Administrator's Guide</i>.</p>
-O	Indicates that you want to restore the chunk and bring it online	None.
-p <i>pathname</i>	Indicates the disk partition or unbuffered device of the chunk	<p>The chunk can be an unbuffered device or a buffered file. When you specify a pathname, you can use either a full pathname or a relative pathname. However, if you use a relative pathname, it must be relative to the directory that was the current directory when you initialized the database server.</p> <p>For pathname syntax, see your operating-system documentation.</p>
-s	Indicates that you want to change the status of a chunk	<p>Restrictions: You can only change the status of a chunk in a mirrored pair or a non-primary chunk within a noncritical dbspace.</p> <p>For more information, see changing the mirror status in the <i>IBM Informix Administrator's Guide</i>.</p>
-y	Causes the database server to respond yes to all prompts automatically	None.
<i>blob</i> space	Names the blob space whose status you want to change	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see changing the mirror status in the <i>IBM Informix Administrator's Guide</i> .
<i>db</i> space	Names the db space whose status you want to change	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For more information, see changing the mirror status in the <i>IBM Informix Administrator's Guide</i> .
<i>sb</i> space	Names the sb space whose status you want to change	Syntax must conform to the Identifier segment; see the <i>IBM Informix Guide to SQL: Syntax</i> . For background information, see changing the mirror status in the <i>IBM Informix Administrator's Guide</i> .

Chapter 15. The onstat Utility

In This Chapter

This chapter shows you how to use the following **onstat** options:

- “onstat -: Print output header” on page 15-6
- “onstat --: Print onstat options and functions” on page 15-7
- “Running onstat Commands on a Shared Memory Dump File” on page 15-8
- “onstat -a: Print onstat -cuskbtdlp” on page 15-9
- “onstat -b: Print buffer information for buffers in use” on page 15-9
- “onstat -B: Print all buffer information” on page 15-10
- “onstat -c: Print ONCONFIG file contents” on page 15-12
- “onstat -C: Print B-tree scanner information” on page 15-13
- “onstat -d: Print chunk information” on page 15-19
- “onstat -D: Print page-read and page-write information” on page 15-22
- “onstat -F: Print counts” on page 15-23
- “onstat -g Monitoring Options” on page 15-25
- “onstat -G: Print TP/XA transaction information” on page 15-136
- “onstat -h: Print buffer header hash chain information” on page 15-138
- “onstat -j: Provide onpload status information ” on page 15-139
- “onstat -k: Print active lock information” on page 15-141
- “onstat -l: Print physical and logical log information” on page 15-142
- “onstat -o: Output shared memory contents to a file” on page 15-146
- “onstat -p: Print profile counts” on page 15-148
- “onstat -P: Print partition information” on page 15-151
- “onstat -r: Repeatedly print selected statistics” on page 15-152
- “onstat -t and -T: Print tblspace information” on page 15-159
- “onstat -u: Print user activity profile” on page 15-161
- “onstat -x: Print database server transaction information” on page 15-163
- “onstat -X: Print thread information” on page 15-165
- “onstat -z: Clear statistics” on page 15-167

The **onstat** utility reads shared-memory structures and provides statistics about the database server at the time that the command executes. The *system-monitoring interface* also provides information about the database server. For information on the system-monitoring interface, see Chapter 2, “The sysmaster Database.”

You can combine multiple **onstat** option flags in a single command. The contents of shared memory might change as the **onstat** output displays. The **onstat** utility does not place any locks on shared memory, so running the utility does not affect performance.

Monitor the Database Server Status

One useful feature of `onstat` output is the heading that indicates the database server status. Whenever the database server is blocked, `onstat` displays the following line after the banner line:

Blocked: *reason*

The variable *reason* can take one of the following values.

Reason

Description

CKPT Checkpoint

LONGTX
Long transaction

ARCHIVE
Ongoing archive

MEDIA_FAILURE
Media failure

HANG_SYSTEM
Database server failure

DBS_DROP
Dropping a dbspace

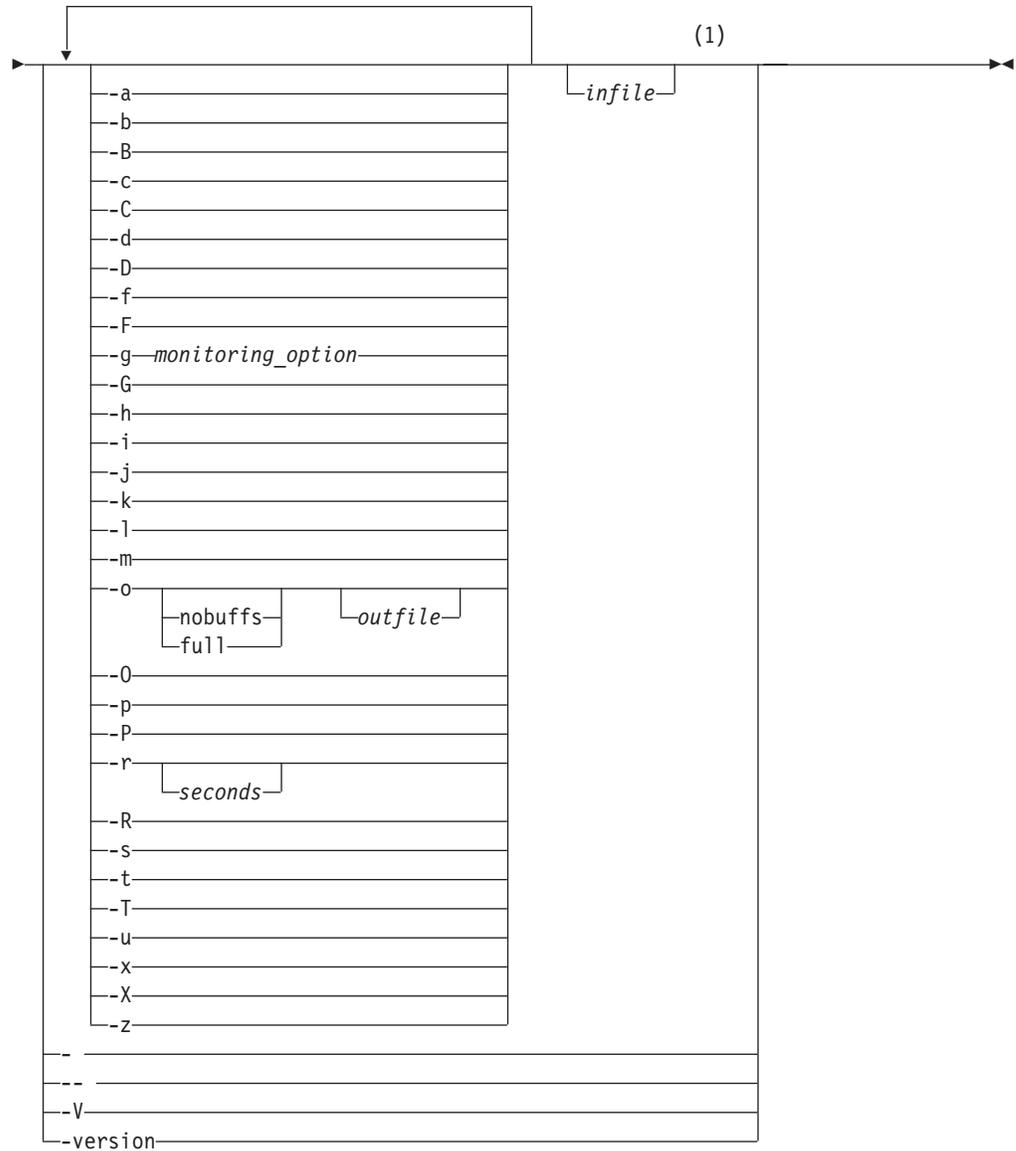
DDR Discrete high-availability data replication

LBU Logs full high-watermark

ADMINISTRATION
Database is in administration mode

onstat Syntax

►► onstat `--pu` ►►



Notes:

- 1 Only one occurrence of each item is allowed. More than one option can be specified on a single **onstat** command invocation.

Element	Purpose	Key Considerations
-	Displays the output header only.	See “onstat -: Print output header” on page 15-6.
--	Displays a listing of all onstat options and their functions	See “onstat --: Print onstat options and functions” on page 15-7. This option cannot be combined with any other onstat option.
-a	Interpreted as onstat -cuskbtdlp . Displays output in that order.	See “onstat -a: Print onstat -cuskbtdlp” on page 15-9.
-b	Displays information about buffers currently in use, including number of resident pages in the buffer pool	See “onstat -b: Print buffer information for buffers in use” on page 15-9.

Element	Purpose	Key Considerations
-B	Obtains information about all database server buffers, not just buffers currently in use.	See “onstat -B: Print all buffer information” on page 15-10.
-c	Displays the ONCONFIG file: <ul style="list-style-type: none"> • \$INFORMIXDIR/etc/\$ONCONFIG for UNIX • %INFORMIXDIR%\etc\ %ONCONFIG% for Windows 	See “onstat -c: Print ONCONFIG file contents” on page 15-12.
-C	Prints B-tree scanner information	See “onstat -C: Print B-tree scanner information” on page 15-13.
-d	Displays information for chunks in each storage space	See “onstat -d: Print chunk information” on page 15-19.
-D	Displays page-read and page-write information for the first 50 chunks in each dbspace	See “onstat -D: Print page-read and page-write information” on page 15-22.
-f	Lists the dbspaces currently affected by the DATASKIP feature	See “onstat -f: Print dbspace information affected by dataskip” on page 15-23.
-F	Displays a count for each type of write that flushes pages to disk	See “onstat -F: Print counts” on page 15-23.
-g option	Prints monitoring option	See “onstat -g Monitoring Options” on page 15-25.
-G	Prints global transaction IDs	See “onstat -G: Print TP/XA transaction information” on page 15-136.
-h	Provides information on the buffer header hash chains	See “onstat -h: Print buffer header hash chain information” on page 15-138.
-i	Puts the onstat utility into interactive mode	See “onstat -i: Initiate interactive mode” on page 15-139.
-j	Prints the interactive status of the active onpload process	See “onstat -j: Provide onpload status information ” on page 15-139.
-k	Displays information about active locks	See “onstat -k: Print active lock information” on page 15-141.
-l	Displays information about physical and logical logs, including page addresses	See “onstat -l: Print physical and logical log information” on page 15-142.
-m	Displays the 20 most recent lines of the database server message log	Output from this option lists the full pathname of the message-log file and the 20 file entries. A date-and-time header separates the entries for each day. A time stamp prefaces single entries within each day. The name of the message log is specified as MSGPATH in the ONCONFIG file. See “onstat -m: Print recent system message log information” on page 15-145.
-o	Saves a copy of the shared-memory segments to <i>outfile</i>	See “onstat -o: Output shared memory contents to a file” on page 15-146.
-O	Displays information about the Optical Subsystem memory cache and staging-area blob space	See “onstat -O: Print optical subsystem information” on page 15-146.
-p	Displays profile counts.	See “onstat -p: Print profile counts” on page 15-148.

Element	Purpose	Key Considerations
-P	Displays for all partitions the partition number and the break-up of the buffer-pool pages that belong to the partition	See “onstat -P: Print partition information” on page 15-151.
-pu	If you invoke onstat without any options, the command is interpreted as onstat -pu (-p option and -u option). Displays profile counts and prints a profile of user activity	See “onstat -p: Print profile counts” on page 15-148 and “onstat -u: Print user activity profile” on page 15-161.
-r seconds	Repeats the accompanying onstat options after a wait time specified in <i>seconds</i> between each execution	See “onstat -r: Repeatedly print selected statistics” on page 15-152.
-R	Displays detailed information about the LRU queues, FLRU queues, and MLRU queues	See “onstat -R: Print LRU, FLRU, and MLRU queue information” on page 15-155.
-s	Displays general latch information	See “onstat -s: Print latch information” on page 15-157.
-t	Displays tbspace information, including residency state, for active tbspaces	See “onstat -t and -T: Print tbspace information” on page 15-159.
-T	Displays tbspace information for all tbspaces	See “onstat -t and -T: Print tbspace information” on page 15-159.
-u	Prints a profile of user activity	See “onstat -u: Print user activity profile” on page 15-161.
-V	Displays the software version number and the serial number. This option cannot be combined with any other onstat option.	See “Obtaining Utility Version Information” on page 6-1.
-version	Displays the build version, host, OS, number and date, as well as the GLS version. This option cannot be combined with any other onstat option.	See “Obtaining Utility Version Information” on page 6-1.
-x	Displays information about transactions	See “onstat -x: Print database server transaction information” on page 15-163.
-X	Obtains precise information about the threads that are sharing and waiting for buffers	See “onstat -X: Print thread information” on page 15-165.
-z	Sets the profile counts to 0	See “onstat -z: Clear statistics” on page 15-167.
<i>infile</i>	Specifies a source file for the onstat command	This file must include a previously stored shared-memory segment that you created with the onstat -o command. For instructions on how to create the <i>infile</i> with onstat -o , see “onstat -o: Output shared memory contents to a file” on page 15-146. For information about running onstat on the source file, see “Running onstat Commands on a Shared Memory Dump File” on page 15-8.

Interactive Execution

To put the **onstat** utility in interactive mode, use the **-i** option. Interactive mode allows you to enter multiple options, one after the other, without exiting the program. For information on using interactive mode, see “onstat -i: Initiate interactive mode” on page 15-139.

Continuous onstat Execution

Use the **onstat -r** option combined with other **onstat** options to cause the other options to execute repeatedly at a specified interval. For information, see “onstat -r: Repeatedly print selected statistics” on page 15-152.

onstat: onstat -pu equivalent

Syntax:

▶▶ onstat —————▶▶

If you invoke **onstat** without any options, the command is interpreted as **onstat -pu** (**-p** option and **-u** option).

onstat -: Print output header

Syntax:

▶▶ onstat - —————▶▶

All **onstat** output includes a header. The **onstat -** option displays only the output header and is useful for checking the database server mode. The header takes the following form:

Version--*Mode (Type)*--(*Checkpoint*)--Up *Uptime*--*Sh_mem* Kbytes

Version

Is the product name and version number

Mode

Is the current operating mode.

(*Type*)

If the database server uses High-Availability Data Replication, indicates whether the type is primary or secondary

If the database server is not involved in data replication, this field does not appear. If the type is primary, the value P appears. If the type is secondary, the value S appears.

(*Checkpoint*)

Is a checkpoint flag

If it is set, the header might display two other fields after the mode if the timing is appropriate:

(CKPT REQ)

Indicates that a user thread has requested a checkpoint

(CKPT INP)

Indicates that a checkpoint is in progress. During the checkpoint,

access is limited to read only. The database server cannot write or update data until the checkpoint ends

Uptime

Indicates how long the database server has been running

Sh_mem

Is the size of database server shared memory, expressed in kilobytes

A sample header for the database server follows:

Dynamic Server Version 11.50.UC1--On-Line--Up 15:11:41--9216 Kbytes

Logs Full Subheader

If the database server is blocked, the **onstat** header output includes an extra line that reads as follows:

Blocked: *reason(s)*

The reason can be one or more of the following.

Reason

Explanation

CKPT Checkpoint

LONGTX

Long transaction

ARCHIVE

Ongoing storage-space backup

MEDIA_FAILURE

Media failure

HANG_SYSTEM

Database server failure

DBS_DROP

Dropping a dbspace

DDR Discrete data replication

LBU Logs full high-watermark

ADMINISTRATION

Database is in administration mode, with limited users, including user **informix**

onstat --: Print onstat options and functions

Syntax:

▶▶ onstat -- ◀◀

The **--** option displays a listing of all **onstat** options and their functions. You cannot combine this option with any other flag.

Running onstat Commands on a Shared Memory Dump File

Syntax:

```
▶▶ onstat options infile ▶▶
```

You can run **onstat** commands against a shared memory dump file. The shared memory dump can be produced explicitly by using the **onstat -o** command. If the DUMPSHMEM configuration parameter is set to 1 or set to 2, the dump file is created automatically at the time of an assertion failure.

When using the command line, enter the source file as the final argument. The following example prints information about all threads for the shared memory dump contained in the file named `onstat.out`, rather than attempting to attach to the shared memory of a running server.

```
onstat -g ath onstat.out
```

For instructions on how to create the memory dump file with **onstat -o**, see “onstat -o: Output shared memory contents to a file” on page 15-146.

Running onstat Commands on a Shared Memory Dump File Interactively

Use **onstat -i** (interactive mode) to run more than one **onstat** command against a dump file. Interactive mode can save time because the file is read only once. In command-line mode, each command reads the file.

The following example reads the shared memory dump file and enters interactive mode. Other **onstat** commands can be executed against the dump file in the normal interactive fashion.

```
onstat -i source_file
```

For information about interactive mode, see “onstat -i: Initiate interactive mode” on page 15-139.

Running onstat Commands on a Shared Memory Dump File Created Without a Buffer Pool

Certain **onstat** commands have different output when you run them on a dump file created without the buffer pool (created with **onstat -o nobuffs** or with the DUMPSHMEM configuration parameter set to 2).

If you run **onstat -B** on a dump file created without the buffer pool, the output will display 0 in the `memaddr`, `nslots`, and `pgflgs` columns.

If you run **onstat -g seg** on a dump file created without the buffer pool, the output will show both the original and `nobuffs` resident segment size.

If you run **onstat -P** on a shared-memory dump file that does not have the buffer pool, the output is:

```
Nobuffs dumpfile -- this information is not available
```

onstat -a: Print onstat -cuskbtdlp

Syntax:

▶▶ onstat -a ▶▶

The **-a** option is interpreted as **onstat -cuskbtdlp**, and output is displayed in that order. For an explanation of each option, refer to the appropriate flag in the following sections.

onstat -b: Print buffer information for buffers in use

Syntax:

▶▶ onstat -b ▶▶

The **onstat -b** option displays information about buffers currently in use, including the total number of resident pages in the buffer pool. For information about displaying information about all buffers, use “onstat -B: Print all buffer information” on page 15-10.

The maximum number of buffers available is specified in the **buffers** field in the BUFFERPOOL configuration parameter in the ONCONFIG file.

The **onstat -b** option also provides summary information about the number of modified buffers, the total number of resident pages in the buffer pool, the total number of buffers available, the number of hash buckets available, and the size of the buffer in bytes (the page size).

123 modified, 23 resident, 2000 total, 2048 hash buckets, 2048 buffer size.

Example Output

Following is sample output from the **onstat -b** command. For a description of the output, see “onstat -B: Print all buffer information” on page 15-10.

```
IBM Informix Dynamic Server Version 11.50.UC2 -- On-Line -- Up 00:01:39 -- 1075308 Kbytes
```

```
Buffer pool page size: 2048
```

address	userthread	flgs	pagenum	memaddr	nslots	pgflgs	xflgs	owner	waitlist
44454970	0	84	1:30563	4472f000	18	801	80	ffffffffffffffff	0
4445d418	0	84	1:30562	447b1800	18	801	80	ffffffffffffffff	45d654e0
44468b60	0	84	1:30567	4485e000	18	801	80	ffffffffffffffff	0
44476ec0	0	84	1:30565	44934000	18	801	80	ffffffffffffffff	0
444875b8	0	84	1:30564	44a2b800	18	801	80	ffffffffffffffff	0
4449dc50	0	84	1:30566	44b7d000	18	801	80	ffffffffffffffff	0
444d0700	0	c23	1:34245	44e78000	18	801	10	0	0
444d1800	0	c23	1:34253	44e88000	18	801	10	0	0
444d2900	0	c23	1:34261	44e98000	18	801	10	0	0
444d3a00	0	c23	1:34269	44ea8000	18	801	10	0	0
444d4b00	0	c23	1:34277	44eb8000	18	801	10	0	0
444d5c00	0	c23	1:34285	44ec8000	18	801	10	0	0
444d6c78	0	84	1:30568	44ed7800	18	801	80	ffffffffffffffff	0
444d6d00	0	c23	1:34293	44ed8000	18	801	10	0	0
444d7e00	0	c23	1:34301	44ee8000	18	801	10	0	0
444d8f00	0	c23	1:34309	44ef8000	18	801	10	0	0
444da000	0	c23	1:34317	44f08000	18	801	10	0	0
444db100	0	c23	1:34325	44f18000	18	801	10	0	0
444dc200	0	c23	1:34333	44f28000	18	801	10	0	0
444dca80	0	c23	1:36184	44f30000	18	801	10	0	0
444dd300	0	c23	1:34341	44f38000	18	801	10	0	0
444ddb80	0	c23	1:34346	44f40000	18	801	10	0	0
444ed288	0	84	1:30569	45028800	18	801	80	ffffffffffffffff	0

4472 modified, 5000 total, 8192 hash buckets, 2048 buffer size

```
Buffer pool page size: 8192
```

```
0 modified, 1000 total, 1024 hash buckets, 8192 buffer size
```

Figure 15-1. `onstat -b` Output

onstat -B: Print all buffer information

Syntax:

```
▶▶ onstat -B ◀◀
```

The **onstat -B** option displays information about all buffers. Both **onstat -B** and **onstat -b** display the similar information except that **onstat -b** only displays buffers that are currently being accessed whereas **onstat -B** displays information for all the buffers.

For information about running **onstat -B** on a dump file created without the buffer pool, see “Running onstat Commands on a Shared Memory Dump File” on page 15-8.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC2 -- On-Line -- Up 00:01:39 -- 1075308 Kbytes
```

```
Buffer pool page size: 2048
```

address	userthread	flgs	pagenum	memaddr	nslots	pgflgs	xflgs	owner	waitlist
44454970	0	84	1:30563	4472f000	18	801	80	ffffffffffffffff	0
4445d418	0	84	1:30562	447b1800	18	801	80	ffffffffffffffff	45d654e0
44468b60	0	84	1:30567	4485e000	18	801	80	ffffffffffffffff	0
44476ec0	0	84	1:30565	44934000	18	801	80	ffffffffffffffff	0
444875b8	0	84	1:30564	44a2b800	18	801	80	ffffffffffffffff	0
4449dc50	0	84	1:30566	44b7d000	18	801	80	ffffffffffffffff	0
444d0700	0	c23	1:34245	44e78000	18	801	10	0	0
444d1800	0	c23	1:34253	44e88000	18	801	10	0	0
444d2900	0	c23	1:34261	44e98000	18	801	10	0	0
444d3a00	0	c23	1:34269	44ea8000	18	801	10	0	0
444d4b00	0	c23	1:34277	44eb8000	18	801	10	0	0
444d5c00	0	c23	1:34285	44ec8000	18	801	10	0	0
444d6c78	0	84	1:30568	44ed7800	18	801	80	ffffffffffffffff	0
444d6d00	0	c23	1:34293	44ed8000	18	801	10	0	0
444d7e00	0	c23	1:34301	44ee8000	18	801	10	0	0
444d8f00	0	c23	1:34309	44ef8000	18	801	10	0	0
444da000	0	c23	1:34317	44f08000	18	801	10	0	0
444db100	0	c23	1:34325	44f18000	18	801	10	0	0
444dc200	0	c23	1:34333	44f28000	18	801	10	0	0
444dca80	0	c23	1:36184	44f30000	18	801	10	0	0
444dd300	0	c23	1:34341	44f38000	18	801	10	0	0
444ddb80	0	c23	1:34346	44f40000	18	801	10	0	0
444ed288	0	84	1:30569	45028800	18	801	80	ffffffffffffffff	0

4472 modified, 5000 total, 8192 hash buckets, 2048 buffer size

```
Buffer pool page size: 8192
```

```
0 modified, 1000 total, 1024 hash buckets, 8192 buffer size
```

Figure 15-2. onstat -B Output

Output Description

Buffer pool page size

the size of the buffer pool pages in bytes

address the address of the buffer header in the buffer table

userthread

the address of the most recent user thread to access the buffer table. Many user threads might be reading the same buffer concurrently.

flgs Uses the following flag bits to describe the buffer:

0x01 Modified data

0x02 Data

0x04 LRU

0x08 Error

pagenum

the physical page number on the disk

memaddr

the buffer memory address

nslots the number of slot-table entries in the page

This field indicates the number of rows (or portions of a row) that are stored on the page.

pgflgs Uses the following values, alone or in combination, to describe the page type:

- 1 Data page
- 2 Tblspace page
- 4 Free-list page
- 8 Chunk free-list page
- 9 Remainder data page
- b Partition resident blobpage
- c Blobspace resident blobpage
- d Blob chunk free-list bit page
- e Blob chunk blob map page
- 10 B-tree node page
- 20 B-tree root-node page
- 40 B-tree branch-node page
- 80 B-tree leaf-node page
- 100 Logical-log page
- 200 Last page of logical log
- 400 Sync page of logical log
- 800 Physical log
- 1000 Reserved root page
- 2000 No physical log required
- 8000 B-tree leaf with default flags

xflgs Uses the following flag bits to describe buffer access:

- 0x10 share lock
- 0x80 exclusive lock

owner the user thread that set the *xflgs* buffer flag

waitlist

the address of the first user thread that is waiting for access to this buffer

For a complete list of all threads waiting for the buffer, refer to “onstat -X: Print thread information” on page 15-165.

onstat -c: Print ONCONFIG file contents

Syntax:

▶▶ onstat -c ◀◀

Use the **onstat -c** option to display the contents of the ONCONFIG file. The database server first checks if you have assigned a value to the environment variable **ONCONFIG**. You can use the **onstat -c** option with the database server in any mode, including offline.

UNIX Only:

On UNIX, if you have set **ONCONFIG**, **onstat -c** displays the contents of the **\$INFORMIXDIR/etc/\$ONCONFIG** file. If not, by default, **onstat -c** displays the contents of **\$INFORMIXDIR/etc/onconfig**.

Windows Only:

On Windows, if you have set **ONCONFIG**, **onstat -c** displays the contents of the **%INFORMIXDIR%\etc\%ONCONFIG%** file. If not, by default, **onstat -c** displays the contents of **%INFORMIXDIR%\etc\onconfig**.

onstat -C: Print B-tree scanner information

Syntax:



Use the **-C** option to print the information about the B-tree scanner subsystem and each B-tree scanner thread. The following options are available with the **onstat -C** command and can be combined:

- prof* Prints the profile information for the system and each B-tree scanner thread. This is the default option.
- hot* Prints the hot list index key in the order to be cleaned
- part* Prints all partitions with index statistics
- clean* Prints information about all the partitions that were cleaned or need to be cleaned
- range* Prints the savings in pages processed by using index range scanning
- map** Displays the current bitmaps for each index being cleaned by the alice cleaning method
- alice** Displays the efficiency of the alice cleaning method option
- all* Prints all **onstat -C** options

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line --
Up 10 days 02:18:06 -- 48080 Kbytes

Btree Cleaner Info
BT scanner profile Information
=====
Active Threads                                1
Global Commands                               2000000   Building hot list
Number of partition scans                     11003
Main Block                                    0xc000000003c9dc68
BTC Admin                                      0xc0000000024bc208

BTS info          id  Prio  Partnum  Key  Cmd  Yield N
0xc000000003c9dee8  0  High  0x00000000  0  40  77
  Number of leaves pages scanned              77
  Number of leaves with deleted items          6
  Time spent cleaning (sec)                   0
  Number of index compresses                  0
  Number of deleted items                     113
  Number of index range scans                 0
  Number of index leaf scans                 0
  Number of index alice scans                2

```

Figure 15-3. onstat -C Prof

Output Description

Id BTSCANNER ID

Prio Current priority of BTSCANNER

Partnum

The partition number for the index this thread is currently working on

Cmd Command this thread is processing currently

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line -- Up 10 days 02:19:56 -- 48080 Kbytes

Btree Cleaner Info

Index Hot List
=====
  Current Item      5   List Created      15:29:47
  List Size         4   List expires in   0 sec
  Hit Threshold     500  Range Scan Threshold -1

Partnum  Key  Hits
0x00100191  1   14 *
0x00A00022  1   13 *
0x00100191  2    8 *
0x00100150  2    7 *

```

Figure 15-4. onstat -C hot

Output Description

Partnum

The partition number for an index

Key Index Key
Hits The current value of the Hit counter
 * Indicates that this partition has been cleaned during this hot list duration

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line -- Up 10 days 02:20
:34 -- 48080 Kbytes

Btree Cleaner Info

Index Statistics
=====
Partnum Key      Positions  Compress   Split
0x00100002 1          146        0           0
0x00100004 1           4          0           0
0x00100004 2          13         0           0
0x00100005 1           1          0           0
0x00100005 2           0          0           0
0x00100006 1           1          0           0
0x00100006 2           0          0           0
0x00100007 2           1          0           0
0x00100008 2           1          0           0
0x0010000a 1           0          0           0
0x0010000e 3           1          0           0
0x00100011 1           1          0           0
0x00100013 2           2          0           0

```

Figure 15-5. onstat -C part

Output Description

Partnum

The partition number for an index

Key Index Key

Positions

Number of times index has been read

Compress

Number of pages which have been compressed

Split

Number of splits that have occurred

C Indicates partition is busy being cleaned

N Index partition no longer eligible for cleaning

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line --
Up 10 days 02:21                               :50 -- 48080 Kbytes

```

Btree Cleaner Info

Index Cleaned Statistics

=====

Partnum	Key	Dirty Hits	Clean Time	Pg Examined	Items Del	Pages/Sec
0x00100013	2	2	0	0	0	0.00
0x0010008b	3	1	0	0	0	0.00
0x001000c7	1	2	0	0	0	0.00
0x00100150	2	7	0	0	0	0.00
0x0010016f	2	2	0	0	0	0.00
0x00100191	1	14	0	0	0	0.00
0x00100191	2	8	0	0	0	0.00
0x00a00011	2	6	0	0	0	0.00
0x00a00013	1	0	0	24	0	24.00
0x00a00019	1	0	0	470	225	470.00
0x00a00022	1	13	0	0	0	0.00
0x00a00022	2	5	0	0	0	0.00

Figure 15-6. onstat -C clean

Output Description

Partnum

The partition number for an index

Key Index Key

Dirty Hits

Number of times a dirty page has been scanned

Clean Time

Total time spent, in seconds

Pg Examined

Number of pages examined by btscanner thread

Items Del

Number of items removed from this index

Pages/Sec

Number of pages examined per second

C Indicates partition is busy being cleaned

N index partition is no longer eligible for cleaning

Example Output

```

BM Informix Dynamic Server Version 11.50.FC1    -- On-Line --
Up 10 days 02:23:40 -- 48080 Kbytes

Btree Cleaner Info

Cleaning Range Statistics
=====
Partnum  Key          Low          High          Size          Saving
0x001001bc 2          36           69           96           65.6 %
0x001001be 1          16           20           48           91.7 %
0x001001cd 1          8            21           32           59.4 %
0x001001cd 2          24           25           32           96.9 %

```

Figure 15-7. onstat -C range

Output Description

- Partnum* The partition number
- Key* Index Key
- Low* Low boundary for range scan
- High* High boundary for index scan
- Size* Size of index in pages
- Saving* Percentage of time saved versus a full scan
- C** Indicates partition is busy being cleaned
- N** Index partition is no longer eligible for cleaning

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line --
Up 10 days 02:25:05 -- 48080 Kbytes

Btree Cleaner Info

ALICE Bitmap of Deleted Index Items
=====
Partnum  Key          Map
0x00100013 2 0000: 80000000 00000000
0x0010008b 3 0000: 80000000 00000000
0x001000c7 1 0000: 80000000 00000000
0x00100150 2 0000: 80000000 00000000
0x0010016f 2 0000: 80000000 00000000
0x00100191 1 0000: 80000000 00000000
0x00100191 2 0000: 80000000 00000000
0x00a00011 2 0000: 80000000 00000000
0x00a00013 1 0000: 00000000 00000000
0x00a00019 1 0000: 00000000 00000000
0x00a00022 1 0000: 80000000 00000000
0x00a00022 2 0000: 80000000 00000000

```

Figure 15-8. onstat -C map

Output Description

- Partnum* The partition number

Key Index Key
Map Alice bitmap

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1    -- On-Line --
Up 10 days 02:24:24 -- 48080 Kbytes

Btree Cleaner Info

ALICE Cleaning Statistics
=====

System ALICE Info: Mode =    6, Eff =    30 %, Adj =    5

Partnum    Mode BM_Sz    Used_Pg    Examined    Dirty_Pg    # I/O    Found    Eff    Adj
0x00100013    6    64        97            0            0        0        0    0.0 %    0
0x0010008b    6    64        5            0            0        0        0    0.0 %    0
0x001000c7    6    64        2            0            0        0        0    0.0 %    0
0x00100150    6    64        91           0            0        0        0    0.0 %    0
0x0010016f    6    64        91           0            0        0        0    0.0 %    0
0x00100191    6    64        26           0            0        0        0    0.0 %    0
0x00100191    6    64        26           0            0        0        0    0.0 %    0
0x001001bc    0    0        91           0            0        0        0    0.0 %    0
0x001001cd    0    0        26           0            0        0        0    0.0 %    0
0x001001cd    0    0        26           0            0        0        0    0.0 %    0
0x00a00011    6    64        91           0            0        0        0    0.0 %    0
0x00a00013    6    64        25           24           3        3        1    33.3 %    1
0x00a00019    6    64       470           470           3        3        2    66.7 %    1
0x00a00022    6    64        26           0            0        0        0    0.0 %    0
0x00a00022    6    64        26           0            0        0        0    0.0 %    0

```

Figure 15-9. onstat -C alice

Output Description

Partnum

The partition number for an index

Mode

The alice mode for the current partition

BM_Sz

The size allocated for the bitmap

Used_Pg

The size of the index in pages (used)

Dirty_Pg

Number of dirty pages

I/O

Number of pages read

Found

Number of dirty pages found in reads

Eff

How efficient was the bitmap

Adj

Number of times the alice efficiency level for the partition was insufficient and was adjusted

onstat -d: Print chunk information

Syntax:

```
▶▶ onstat -d ◀◀
```

Use the **-d** option to display information for chunks in each storage space. You can interpret output from this option as follows.

Example Output

```
Dbspaces
address number flags fchunk nchunks pgsz flags owner name
ad357e8 1 0x60001 1 1 2048 N B informix rootdbs
b62a5b0 2 0x60001 2 1 4096 N B informix dbbsp1
2 active, 2047 maximum

Chunks
address chunk/dbs offset page Rd page Wr pathname
ad35948 1 1 0 493 5803 /local0/engines/ol_tuxedo/ifmxdata/rootdbs
b62a710 2 2 0 4 20 /local0/engines/ol_tuxedo/ifmxdata/dbbsp1
2 active, 32766 maximum

NOTE: The values in the "page Rd" and "page Wr" columns for DBspace chunks are
displayed in terms of system base page size.

Expanded chunk capacity mode: always
```

Figure 15-10. onstat -d Output

Output Description

The first section of the display describes the storage spaces:

address Is the address of the storage space in the shared-memory space table

number

Is the unique ID number of the storage space assigned at creation

flags Uses the following hexadecimal values to describe each storage space:

0x00000000

Mirror not allowed and dbspace is unmirrored

0x00000001

Mirror is allowed and dbspace is unmirrored

0x00000002

Mirror is allowed and dbspace is mirrored

0x00000004

Down

0x00000008

Newly mirrored

0x00000010

Blobspace

0x00000020
 Blobospace on removable media

0x00000040
 Blobospace is on optical media

0x00000080
 Blobospace is dropped

0x00000100
 Blobospace is the optical STAGEBLOB

0x00000200
 Space is being recovered

0x00000400
 Space is fully recovered

0x00000800
 Logical log is being recovered

0x00001000
 Table in dbspace is dropped

0x00002000
 Temporary dbspace

0x00004000
 Blobospace is being backed up

0x00008000
 Sbspace

0x0000a001
 Temporary sbspace

0x00010000
 Physical or logical log changed

0x00020000
 Dbspace or chunk tables have changed

0x20002
 Dbspace or chunk tables have changed and dbspace is mirrored

0x60001
 Dbspace has large chunks and is unmirrored. Any changes result in changes on rootdbspace

fchunk Is the ID number of the first chunk

nchunks

Is the number of chunks in the storage space

pgsize Is the size of the dbspace pages in bytes

flags Uses the following letter codes to describe each storage space:

Position 1:

M Mirrored

N Not mirrored

Position 2:

X Newly mirrored

- P Physically recovered, waiting for P -- logical recovery
- L Being logically recovered
- R Being recovered

Position 3:

- B Blobspace
- S Sbspace
- T Temporary Dbspace
- U Temporary Sbspace
- W Temporary Dbspace on Primary (this flag is shown on SD secondary servers only)

Position 4:

- B Dbspace has large chunks greater than 2 GB

- owner* Is the owner of the storage space
- name* Is the name of the storage space

In the line immediately following the storage-space list, **active** refers to the current number of storage spaces in the database server instance including the rootdbs and **maximum** refers to total *allowable* spaces for this database server instance.

The second section of the **onstat -d** output describes the chunks:

address

Is the address of the chunk

chk/dbs

Is the chunk number and the associated space number

offset Is the offset into the file or raw device in pages

size Is the size of the chunk in terms of the page size of the dbspace to which it belongs.

free Is the number of free pages in the chunk in terms of the page size of the dbspace to which it belongs.

For a blobspace, a tilde indicates an approximate number of free blobpages.

For an sbspace, indicates the number of free pages of user data space and total user data space.

bpages

Is the size of the chunk in blobpages

Blobpages can be larger than disk pages; therefore, the **bpages** value can be less than the **size** value.

For an sbspace, is the size of the chunk in sbpages

flags Provides the chunk status information as follows:

Position 1:

- P Primary
- M Mirror

Position 2:

- N Renamed and either Down or Inconsistent
- O Online
- D Down
- X Newly mirrored
- I Inconsistent

Position 3:

- Dbspace
- B Blobspace
- S Sbspace

Position 4:

- B Has large chunks greater than 2 GB

pathname

Is the pathname of the physical device

In the line immediately following the chunk list, **active** displays the number of active chunks (including the root chunk) and **maximum** displays the total number of chunks.

For information about page reads and page writes, refer to “onstat -D: Print page-read and page-write information.”

Using onstat -d with Sbspaces

For information about using **onstat -d** to determine the size of sbspaces, user-data areas, and metadata areas, see monitoring sbspaces in the *IBM Informix Administrator's Guide*.

Using onstat -d with Blobspaces

If you issue the **onstat -d** command on an instance with blobspace chunks, the database server displays the following message:

NOTE: For BLOB chunks, the number of free pages shown is out of date.
Run 'onstat -d update' for current stats.

To obtain the current statistics for blobspace chunks, issue the **onstat -d update** command. The **onstat** utility updates shared memory with an accurate count of free pages for each blobspace chunk. The database server displays the following message:

Waiting for server to update BLOB chunk statistics ...

onstat -D: Print page-read and page-write information

Syntax:

▶▶ onstat -D ◀◀

Use the **-D** option to display page-read and page-write information for the first 50 chunks in each space.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:59:42 -- 34816 Kbytes

Dbspaces
address number flags fchunk nchunks pgsz flags owner name
a40d7d8 1 0x1 1 1 2048 N informix rootdbs
1 active, 2047 maximum

Chunks
address chunk/dbs offset page Rd page Wr pathname
a40d928 1 1 0 0 0 /work/11.1/dbspaces/stardbs3
1 active, 2047 maximum

Expanded chunk capacity mode: disabled
```

Figure 15-11. *onstat -D* Output

Output Description

The output of **onstat -D** is almost identical to the output of **onstat -d**. The following columns are unique to **onstat -D**. For information on the other output columns see “*onstat -d: Print chunk information*” on page 15-19.

page Rd

Is the number of pages read

page Wr

Is the number of pages written

onstat -f: Print dbspace information affected by dataskip

Syntax:

►► onstat -f ◀◀

Use the **-f** option to list the dbspaces that the dataskip feature currently affects. The **-f** option lists both the dbspaces that were set with the DATASKIP configuration parameter and the **-f** option of **onspaces**. When you execute **onstat -f**, the database server displays one of the following three outputs:

- Dataskip is OFF for all dbspaces.
- Dataskip is ON for all dbspaces.
- Dataskip is ON for the following dbspaces:
dbspace1 dbspace2...

onstat -F: Print counts

Syntax:

►► onstat -F ◀◀

Use the **-F** option to display a count for each type of write that flushes pages to disk.

Example Output

```

IBM Informix Dynamic Server Version 11.50.F

Fg Writes      LRU Writes      Chunk Writes
0              330             7631

address flusher state data # LRU  Chunk  Wakeups  Idle Time
c7c8850 0      I      0      9      29     16116   16093.557
      states: Exit Idle Chunk Lru

```

Figure 15-12. *onstat -F* Output

Output Description

You can interpret output from this option as follows:

Fg Writes

Is the number of times that a foreground write occurred

LRU Writes

Is the number of times that an LRU write occurred

Chunk Writes

Is the number of times that a chunk write occurred

address Is the address of the user structure assigned to this page-cleaner thread

flusher Is the page-cleaner number

state Uses the following codes to indicate the current page-cleaner activity:

- C** Chunk write
- E** Exit
- I** Cleaner is idle
- L** LRU queue

The exit code indicates either that the database server is performing a shutdown or that a page cleaner did not return from its write in a specific amount of time. When an operation fails to complete within the allotted time, this situation is known as a time-out condition. The database server does not know what happened to the cleaner, so it is marked as `exit`. In either case, the cleaner thread eventually exits.

data Provides additional information in concert with the **state** field

If **state** is **C**, **data** is the chunk number to which the page cleaner is writing buffers. If **state** is **L**, **data** is the LRU queue from which the page cleaner is writing. The **data** value is displayed as a decimal, followed by an equal sign, and repeated as a hexadecimal.

#LRU Corresponds to the *onstat -g ath* thread ID output

Chunk

Number of chunks cleaned

Wakeups

Number of times the flusher thread was awoken

Idle Time

Time in seconds the flusher thread has been idle

onstat -g Monitoring Options

The following **onstat -g** options are provided for support and debugging only. You can include only one of these options per **onstat -g** command. For more information, see your *IBM Informix Performance Guide*.

onstat -g Option	Topic or Function
-g act	Prints active threads. See "onstat -g act: Print active threads" on page 15-30.
-g afr	Prints allocated memory fragments for a specified session or shared-memory pool. Each session is allocated a pool of shared memory. To obtain the pool name, see the -mem option. See "onstat -g afr: Print allocated memory fragments" on page 15-31.
-g all	Prints output from all onstat -g options. See "onstat -g all: Print output from all onstat -g options" on page 15-31.
-g ath	Prints all threads. The sqlmain threads represent client sessions. The rstcb value corresponds to the user field of the onstat -u command. See "onstat -g ath: Print information about all threads" on page 15-31. For information on using onstat -g ath to print Enterprise Replication threads, see the <i>IBM Informix Dynamic Server Enterprise Replication Guide</i> .
-g buf	Prints profile information for each buffer pool. See "onstat -g buf: Print buffer pool profile information " on page 15-32.
-g cat	Prints information from the Enterprise Replication global catalog. The global catalog contains a summary of information about the defined servers, replicates, and replicate sets on each of the servers within the enterprise. See "onstat -g cat: Print ER global catalog information" on page 15-34.
-g cac agg	Prints the definitions for user-defined aggregates that are currently in the cache.
-g cac stmt	Prints the contents of the SQL statement cache. Prints the same output as the -g ssc statement.
-g cdr	Prints the settings of Enterprise Replication configuration parameters and environment variables. See "onstat -g cdr config: Print ER settings" on page 15-36.
-g ckp	Prints the checkpoint history and displays configuration recommendations if a suboptimal configuration is detected. See "onstat -g ckp: Print checkpoint history and configuration recommendations" on page 15-38.
-g cmsm	Prints Connection Manager daemon instances and displays the number of connections each daemon has processed. See "onstat -g cmsm: Print Connection Manager information" on page 15-40.
-g con	Prints conditions with waiters. See "onstat -g con: Print condition and thread information " on page 15-41.
-g cpu	Prints information about runtime statistics for all the threads running in the server. See "onstat -g cpu: Print runtime statistics" on page 15-42.
-g dbc	Prints information for database scheduler and the SQL Administration API. See "onstat -g dbc: Print dbScheduler and dbWorker thread statistics" on page 15-43.

onstat -g Option	Topic or Function
-g ddr	Prints the status of the Enterprise Replication database log reader. If log reading is blocked, data might not be replicated until the problem is resolved. See “onstat -g ddr: Print ER database log reader status” on page 15-45.
-g dic	Prints one line of information for each table cached in the shared-memory dictionary. If given a specific table name as a parameter, prints internal SQL information for that table. For more information, see your <i>IBM Informix Performance Guide</i> . For sample output, see “onstat -g dic: Print table information” on page 15-46.
-g dis	Prints a list of database servers and their status, and information about each database server, INFORMIXDIR , sqlhosts file, ONCONFIG file, and hostname. See “onstat -g dis: Print database server information” on page 15-47.
-g dll	Prints a list of dynamic libraries that have been loaded. See “onstat -g dis: Print database server information” on page 15-47.
-g dmp	Prints raw memory at a given address for a number of given bytes. See “onstat -g dmp: Print raw memory ” on page 15-49.
-g dri	Prints data-replication information. See monitoring High-Availability Data-Replication status in the <i>IBM Informix Administrator's Guide</i> . See “onstat -g dri: Print High-Availability Cluster information” on page 15-50.
-g dsc	Prints data-distribution cache information. See “onstat -g dsc: Print distribution cache information ” on page 15-51.
-g dss	Prints detailed statistical information about the activity of individual data sync threads and about user-defined data types. See “onstat -g dss: Print ER environment data” on page 15-52.
-g dtc	Prints statistics about the delete table cleaner which removes rows from the delete table when they are no longer needed. See “onstat -g dtc: Print delete table cleaner statistics” on page 15-52.
-g env	Prints the values of environment variables the database server currently uses. See “onstat -g env: Print environment variable values” on page 15-53.
-g ffr	Prints free fragments for a pool of shared memory. See “onstat -g ffr: Print free fragments ” on page 15-55.
-g glo	Prints global multithreading information. This information includes CPU use information about the virtual processors, the total number of sessions, and other multithreading global counters. On Windows, the virtual processors are operating system threads. The values displayed under the ‘pid’ field are thread IDs not process IDs. (Windows). See “onstat -g glo: Print global multithreading information ” on page 15-56.
-g grp	Prints statistics about the Enterprise Replication grouper. The grouper evaluates the log records, rebuilds the individual log records into the original transaction, packages the transaction, and queues the transaction for transmission. See “onstat -g grp: Print ER grouper statistics” on page 15-57.
-g his	Prints information about the SQLTrace configuration parameter. See “onstat -g his: Print SQLTRACE information ” on page 15-62.
-g imc	Prints information about Informix MaxConnect instances that are connected to the database server. If Informix MaxConnect is not connected to the database server, this command displays No MaxConnect servers are connected.

onstat -g Option	Topic or Function
-g ioa	Prints combined information from -g ioq and -g iov . See “onstat -g ioa: Print combined onstat -g information ” on page 15-64.
-g iob	Prints the big buffer usage summary. See “onstat -g iob: Print big buffer use summary ” on page 15-65.
-g iof	Prints asynchronous I/O statistics by chunk or file. This option is similar to the -D option, except it also displays information on nonchunk, temporary, and sort-work files. See “onstat -g iof: Print asynchronous I/O statistics ” on page 15-66.
-g iog	Prints AIO global information. See “onstat -g iog: Print AIO global information” on page 15-67.
-g ioq	Prints pending I/O operations for the <i>queue name</i> . If given the <i>gfd</i> or <i>kaio</i> queue name, a queue for each CPU VP is displayed. If <i>queue name</i> is omitted, I/O statistics for all queues are displayed. See “onstat -g ioq: Print I/O queue information ” on page 15-67.
-g iov	Prints asynchronous I/O statistics by virtual processor. See “onstat -g iov: Print AIO VP statistics ” on page 15-69.
-g ipl	Prints index page logging status. See “onstat -g ipl: Print index page logging status information ” on page 15-69.
-g lap	Prints information on the status of light appends. See “onstat -g lap: Print light appends status information ” on page 15-71.
-g lmx	Prints all locked mutexes. See “onstat -g lmx: Print all locked mutexes ” on page 15-71.
-g lsc	Prints information about light scans. See “onstat -g lsc: Print active light scan status ” on page 15-72.
-g mem	<p>Prints statistics for a memory pool. Also prints the pool name, type of shared memory segment that contains the pool, the address of the pool, the total size of the pool, the number of bytes of free memory that it contains, and the number of free and allocated fragments in the pool. If no argument is provided, displays information about all pools. The block pools are listed in a separate section after the main pool list.</p> <p>You also can use ISA to obtain detailed information about a memory pool. If you run an SQL query that allocates memory from the PER_STMT_EXEC and PER_STMT_PREP memory duration pools, onstat -g mem displays information on the PRP.sessionid.threadid pool and the EXE.sessionid.threadid pool.</p> <p>See “onstat -g mem: Print pool memory statistics ” on page 15-73. For more information, see the <i>IBM Informix DataBlade API Programmer's Guide</i>.</p>
-g mgm	Prints Memory Grant Manager resource information. See “onstat -g mgm: Print MGM resource information ” on page 15-74.
-g nbm	Prints block bit map for the nonresident segments, one bit per 8-kilobyte block. Bit set indicates block free. See “onstat -g nbm: Print a block bit map ” on page 15-77.
-g nif	Prints statistics about the network interface. This command is useful to determine why data is not replicating. See “onstat -g nif: Print ER network interface statistics” on page 15-78.
-g nsc	Prints shared-memory status by <i>client id</i> . If <i>client id</i> is omitted, all client status areas are displayed. This command prints the same status data as the nss command. See “onstat -g nsc <i>client_id</i> : Print current shared memory connection information ” on page 15-79.

onstat -g Option	Topic or Function
-g nsd	Prints network shared-memory data for poll threads. See “onstat -g nsd: Print poll threads shared-memory data ” on page 15-81.
-g nss	Prints network shared-memory status by <i>session id</i> . If <i>session id</i> is omitted, all session status areas are displayed. This command prints the same status data as the nsc command. See “onstat -g nss: Print shared memory network connections status ” on page 15-82.
-g nta	Prints combined network statistics from -g ntd , -g ntm , -g ntt , and -g ntu . If Informix MaxConnect is installed, this command prints statistics that you can use to tune Informix MaxConnect performance.
-g ntd	Prints network statistics by service. See “onstat -g ntd: Print network statistics” on page 15-83.
-g ntm	Prints network mail statistics. See “onstat -g ntm: Print network mail statistics” on page 15-83.
-g ntt	Prints network user times. See “onstat -g ntt: Print network user times” on page 15-84.
-g ntu	Prints network user statistics. See “onstat -g ntu: Print network user statistics” on page 15-84.
-g opn	Prints open partitions. See “onstat -g opn: Print open partitions” on page 15-85.
-g pos	Prints \$INFORMIXDIR/etc/ .infos.DBSERVERNAME file for UNIX or the %INFORMIXDIR%\etc\ .infos.DBSERVERNAME file for Windows. See “onstat -g pos: Print file values ” on page 15-87.
-g ppf	Prints partition profile for the specified partition number or prints profiles for all partitions. See “onstat -g ppf: Print partition profiles” on page 15-88.
-g prc	Prints information about SPL routine cache. See “onstat -g prc: Print sessions using UDR or SPL routine ” on page 15-89.
-g proxy	Prints proxy distributor information. See “onstat -g proxy: Print Proxy Distributor information” on page 15-90.
-g qst	Prints queue wait statistics. See “onstat -g qst: Print wait options for mutex and condition queues ” on page 15-96.
-g que	Prints statistics for the high-level queue interface (which are common to all the queues of the Enterprise Replication Queue Manager. See “onstat -g que: Prints ER queue statistics ” on page 15-95.
-g rbm	Prints block bit map for the resident segment (communication message area). See “onstat -g rbm: Print a block map of shared memory ” on page 15-97.
-g rcv	Prints statistics about the receive manager, which is a set of service routines between the receive queues and data sync. See “onstat -g rcv: Print ER receive manager statistics” on page 15-98.
-g rea	Prints ready threads. See “onstat -g rea: Print ready threads” on page 15-101.
-g rep	Prints events that are in the queue for the schedule manager. See “onstat -g rep: Print ER schedule manager events ” on page 15-101.
-g rqm	Prints statistics and contents of the low-level queues (each individual queue) managed by the Reliable Queue Manager (RQM). See “onstat -g rqm: Print low-level queue statistics” on page 15-102.

onstat -g Option	Topic or Function
-g rss	Prints remote standalone secondary (RSS) server information. See “onstat -g rss: Print RS secondary server information” on page 15-104.
-g rwm	Prints read/write mutexes. See “onstat -g rwm: Print read and write mutexes ” on page 15-107.
-g sch	Prints the number of semaphore operations, spins, and busy waits for each virtual processor. On Windows, the virtual processors are operating system threads. The values displayed under the ‘pid’ field are thread IDs not process IDs. (Windows) See “onstat -g sch: Print VP information” on page 15-107.
-g sds	Prints shared disk secondary (SDS) server information. See “onstat -g sds: Print SD secondary server information ” on page 15-108.
-g seg	Prints shared-memory-segment statistics. This option shows the number and size of shared-memory segments that the database server is currently using. See “onstat -g seg: Print shared memory segment statistics” on page 15-112.
-g ses	Prints session information by <i>sessionid</i> . If <i>sessionid</i> is missing, a one-line summary of each session prints. See “onstat -g ses: Print session-related information” on page 15-113.
-g sle	Prints all sleeping threads. See “onstat -g sle: Print all sleeping threads ” on page 15-117.
-g smb	Prints detailed information about sbspaces. See “onstat -g smb: Print sbspaces information ” on page 15-118.
-g smx	Displays server multiplexer group connections information. See “onstat -g smx: Print multiplexer group information ” on page 15-119.
-g spi	Prints spin locks with long spins. See “onstat -g spi: Print spin locks with long spins” on page 15-121.
-g sql	Prints SQL information by <i>session id</i> . If <i>session id</i> is omitted, a one-line summary for each session prints. See “onstat -g sql: Print SQL-related session information” on page 15-122.
-g src	Searches for patterns in shared memory. See “onstat -g src: Patterns in shared memory” on page 15-124.
-g ssc	Monitors the number of times that the database server reads the SQL statement in the cache. See example output, see “onstat -g ssc: Print SQL statement occurrences ” on page 15-124. Displays the same output as onstat -g cac stmt . For more information, see improving query performance in the <i>IBM Informix Performance Guide</i> .
-g ssc all	Reports the <i>key-only</i> cache entries as well as the fully cached statements. If the value in the hits column is less than the STMT_CACHE_HITS value, that entry is a <i>key-only</i> cache entry. For more information, see memory utilization in the <i>IBM Informix Performance Guide</i> .
-g ssc pool	Reports usage of all memory pools for the SQL statement cache. The output displays information on the name, class, address, and total size of the memory pools. For more information, see improving query performance in the <i>IBM Informix Performance Guide</i> .

onstat -g Option	Topic or Function
-g stk	Prints the stack of a specified thread or prints stacks for <i>all</i> threads. This option is not supported on all platforms and is not always accurate. See “onstat -g stk <i>tid</i> : Print thread stack ” on page 15-126.
-g stm	Prints the memory that each prepared SQL statement uses. See “onstat -g stm: Print SQL statement memory usage ” on page 15-126. For more information, see memory utilization and improving query performance in the <i>IBM Informix Performance Guide</i> .
-g stq	Prints queue stream information. See “onstat -g stq: Print queue information ” on page 15-127.
-g sts	Prints maximum and current stack use per thread. See “onstat -g sts: Print stack usage per thread ” on page 15-127.
-g sync	Prints which sync is active. See “onstat -g sync: Print ER synchronization status ” on page 15-128.
-g tpf	Prints thread profile for a specific thread ID. See “onstat -g tpf <i>tid</i> : Print thread profiles ” on page 15-130.
-g ufr	Prints allocated fragments by use. See “onstat -g ufr: Print memory pool fragments ” on page 15-130.
-g vpcache	Prints information about CPU VP memory block cache statistics. See “onstat -g vpcache: Print CPU VP memory block cache statistics” on page 15-132.
-g wai	Prints waiting threads; all threads waiting on mutex or condition, or yielding. See “onstat -g wai: Print wait queue thread list ” on page 15-133.
-g wmx	Prints all mutexes with waiters. See “onstat -g wmx: Print all mutexes with waiters ” on page 15-134.
-g wst	Prints wait statistics for threads. See “onstat -g wst: Print wait statistics for threads ” on page 15-134.

onstat -g act: Print active threads

Syntax:

►► onstat — -g — act ◀◀

The **onstat -g act** option prints active threads.

Following is sample output from the **onstat -g act** command. For a description of the output, see “onstat -g ath: Print information about all threads” on page 15-31.

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 18:47:42
-- 101376 Kbytes
Running threads:
tid  tcb      rstcb  prty  status  vp-class  name
2    b3132d8    0     1    running *2adm    adminthd
40   c5384d0    0     1    running *1cpu    tlitcpoll

```

Figure 15-13. onstat -g act Output

onstat -g afr: Print allocated memory fragments

Syntax:

```
▶▶ onstat -g afr ◀◀
```

The **onstat -g afr** option prints allocated memory fragments for a specified session or shared-memory pool. Each session is allocated a pool of shared memory.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 18:47:42
-- 43008 Kbytes
Allocations for pool name dfm_pool:
addr          size      memid
10ac8c000     192      overhead
10ac8d000     24352    dfm
```

Figure 15-14. *onstat -g afr* Output

Output Description

addr Memory address of the pool fragment

size Size, in bytes, of the pool fragment

memid

Memory ID of the pool fragment

onstat -g all: Print output from all onstat -g options

Syntax:

```
▶▶ onstat -g all ◀◀
```

The **onstat -g all** option prints output from all of the **onstat -g** options and is primarily used by IBM Support to gather diagnostic information. For normal administrative purposes, only the individual options should be invoked.

onstat -g ath: Print information about all threads

Syntax:

```
▶▶ onstat -g ath ◀◀
```

The **onstat -g ath** option prints information about all threads.

Example Output

```

Threads:
tid    tcb          rstcb      prty status          vp-class  name
2      10bbf36a8    0          1    IO Idle          3lio     lio vp 0
3      10bc12218    0          1    IO Idle          4pio     pio vp 0
4      10bc31218    0          1    running          5aio     aio vp 0
5      10bc50218    0          1    IO Idle          6msc     msc vp 0
6      10bc7f218    0          1    running          7aio     aio vp 1
7      10bc9e540    10b231028  1    sleeping secs: 1 1cpu     main_loop()
8      10bc12548    0          1    running          1cpu     tlitcpoll
9      10bc317f0    0          1    sleeping forever 1cpu     tlitcplst
10     10bc50438    10b231780  1    IO Wait          1cpu     flush_sub(0)
11     10bc7f740    0          1    IO Idle          8aio     aio vp 2
12     10bc7fa00    0          1    IO Idle          9aio     aio vp 3
13     10bd56218    0          1    IO Idle          10aio    aio vp 4
14     10bd75218    0          1    IO Idle          11aio    aio vp 5
15     10bd94548    10b231ed8  1    sleeping forever 1cpu     aslogflush
16     10bc7fd00    10b232630  1    sleeping secs: 34 1cpu     btscanner 0
32     10c738ad8    10b233c38  1    sleeping secs: 1 1cpu     onmode_mon
50     10c0db710    10b232d88  1    IO Wait          1cpu     sqlxec

```

Figure 15-15. `onstat -g ath` Output

Output Description

tid Thread ID
tcb Thread control block access
rstcb RSAM thread control block access
prty Thread priority
status Thread status
vp-class
 Virtual processor class
name Thread name

onstat -g buf: Print buffer pool profile information

Syntax:

▶▶ `onstat -g buf` ◀◀

The `onstat -g buf` option prints profile information for each buffer pool.

Example Output

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:00:25 -- 1075788 Kbytes

Profile

Buffer pool page size: 2048
dskreads  pagreads  bufreads  %cached  dskwrits  pagwrits  bufwrits  %cached
2065      2067      274619   99.25   4418      36043     81649     94.59
bufwrits_sinceckpt  bufwaits  ovbuff  flushes
14850              0        0        6

Fg Writes      LRU Writes      Avg. LRU Time  Chunk Writes
0              0              nan            2909

Buffer pool page size: 8192
dskreads  pagreads  bufreads  %cached  dskwrits  pagwrits  bufwrits  %cached
0          0          0          0.00    0          0          0          0.00
bufwrits_sinceckpt  bufwaits  ovbuff  flushes
0              0        0        0

Fg Writes      LRU Writes      Avg. LRU Time  Chunk Writes
0              0              nan            0
```

Figure 15-16. `onstat -g buf` Output

Output Description

Buffer pool page size

Number of bytes in a page in the buffer pool being profiled

dskreads

Number of disk read operations performed to bring pages into this buffer pool. Each read operation reads one or more pages.

pagreads

Number of pages read from disk to this buffer pool

bufreads

Number of times a memory image for a page was read from this buffer pool

%cached

Percentage of page reads for this buffer pool that were satisfied by a cached page image (rather than having to perform a disk read). Computed as $(\text{bufreads} - \text{dskreads}) / \text{bufreads} \times 100$. Higher percentages indicate better caching performance.

dskwrits

Number of disk write operations performed to write changed pages from this buffer pool back to disk. Each write operation writes one or more pages.

pagwrits

Number of pages written to disk from this buffer pool

bufwrits

Number of times a memory image of a page was written to in this buffer pool

%cached

Percentage of page writes for this buffer pool that were satisfied by a cached page image (rather than having to perform a disk write). Computed as $(\text{bufwrits} - \text{dskwrits}) / \text{bufwrits} \times 100$.

bufwrits_sinceckpt

Number of times a memory image of a page was written to in this buffer pool since the last checkpoint

bufwaits

Number of times a thread had to wait for a lock on a buffer in this buffer pool. Higher numbers indicate more contention among multiple threads for mutually incompatible locks on the same pages.

ovbuff

Number of times a changed buffer from this buffer pool was written to disk specifically to create a free buffer to read another requested page. If the *ovbuff* value is high, it may indicate that the buffer pool is not large enough to hold the working set of pages needed by the applications using this buffer pool, which may lead to performance degradation from I/O thrashing.

flushes

Number of times the server performed a mass flush of all dirty buffers in the buffer pool. This can occur for a variety of reasons, including as part of checkpoint processing or if the buffer pool is running out of clean buffers despite normal LRU cleaning activity.

Fg Writes

Number of changed buffers from this buffer pool that were written to disk by a non-I/O flusher thread that was accessing the buffer. This number is a superset of *ovbuff*. In addition to the writes to service page faults counted by *ovbuff*, this value also includes foreground writes that are done by certain operations to maintain the consistency of database logs and reserved pages in order to guarantee correctness of crash recovery in special cases.

LRU Writes

Number of changed buffers from this buffer pool that were written to disk by an LRU cleaner thread. LRU cleaners are activated if the buffer pool exceeds its *LRU_MAX_DIRTY* threshold or if foreground writes occur due to buffer pool overflows.

Avg. LRU Time

Average amount of time taken by an LRU cleaner thread to clean a single LRU chain.

Chunk Writes

Number of changed buffers that were written to disk by a chunk cleaning operation. Chunk cleaning writes out all changed buffers of a given chunk that are in the buffer pool. This is done in a variety of special circumstances that need to clean a large number of buffers quickly, such as checkpoint processing and fast recovery.

onstat -g cat: Print ER global catalog information

Syntax:

▶▶ onstat -g cat ▶▶

The **onstat -g cat** option prints information from the Enterprise Replication global catalog. The global catalog contains a summary of information about the defined servers, replicates, and replicate sets on each of the servers within the enterprise. If a replicated table is undergoing an alter operation, the **onstat -g cat** command shows that it is in alter mode. For example, use this command to determine:

- How many servers and how many replicates are configured

- Which table matches a given replicate
- Whether a server is a root or leaf server
- The current bitmap mask for a given server. You can use the bitmap mask with the output from the **onstat -g rqm** command to determine which server Enterprise Replication is waiting on for an acknowledgement.

The **onstat -g cat** command has the following formats:

```
onstat -g cat
onstat -g cat scope
onstat -g cat replname
```

The following table describes *replname* and *scope*.

Modifier	Description
<i>replname</i>	The name of a replicate
<i>scope</i>	One of the following values: servers —Print information on servers only repls —Print information on replicates only full —Print expanded information for both replicate servers and replicates

Example Output

This sample output from the **onstat -g cat repls** command shows that the table **tab** is in alter mode. The replicate **rep1** is defined on this table, its replicate ID is 6553601. For more information on the replicate attributes that this command displays, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:01:39 -- 28672 Kbytes
GLOBAL-CATALOG CACHE STATISTICS
REPLICATES
-----
Parsed statements:
  Id 6553601 table tab
  Id 6553602 table tab12
Inuse databases: test(2)
Name: rep1, Id: 6553601 State: ACTIVE Flags: 0x800000 ALTERMODE
      use 0 lastexec Wed Dec 31 18:00:00 1969
Local Participant: test:nagaraju.tab
Attributes: TXN scope, Enable ATS, Enable RIS, all columns
            sent in updates
Conflict resolution: [TIMESTAMP]
Column Mapping: ON, columns INORDER, offset 8, uncomp_len 12
Column Name Verification: ON
No Replicated UDT Columns
Name: rep12, Id: 6553602 State: ACTIVE Flags: 0x800000 use 0
      lastexec Wed Dec 31 18:00:00 1969
Local Participant: test:nagaraju.tab12
Attributes: TXN scope, Enable ATS, Enable RIS, all columns
            sent in updates
Conflict resolution: [TIMESTAMP]
Column Mapping: ON, columns INORDER, offset 8, uncomp_len 2064
Column Name Verification: ON
No Replicated UDT Columns
```

Figure 15-17. *onstat -g cat repls* Output


```

CDR_LOGDELTA:
  CDR_LOGDELTA configuration setting:          0
CDR_PERFLOG:
  CDR_PERFLOG configuration setting:           0
CDR_ROUTER:
  CDR_ROUTER configuration setting:            0
CDR_RMSCALEFACT:
  CDR_RMSCALEFACT configuration setting:       0
CDRSITES_731:
  CDRSITES_731 configuration setting:          [None configured]
CDRSITES_92X:
  CDRSITES_92X configuration setting:          [None configured]
CDRSITES_10X:
  CDRSITES_10X configuration setting:          [None configured]

```

The following sample output of the **onstat -g cdr config** command shows the settings of all Enterprise Replication configuration parameters and CDR_ENV environment variables:

```

onstat -g cdr config

IBM Informix Dynamic Server Version 11.50.UC1      -- On-Line -- Up 00:08:05
CDR_DBSPACE:
  CDR_DBSPACE configuration setting:              rootdbs
CDR_DSLOCKWAIT:
  CDR_DSLOCKWAIT configuration setting:           5
CDR_EVALTHREADS:
  CDR_EVALTHREADS configuration setting:          1, 2
CDR_MAX_DYNAMIC_LOGS:
  CDR_MAX_DYNAMIC_LOGS configuration setting:     0
CDR_NIFCOMPRESS:
  CDR_NIFCOMPRESS configuration setting:          0
CDR_QDATA_SBSpace:
  CDR_QDATA_SBSpace configuration setting:        cdrsbsp
CDR_QHDR_DBSPACE:
  CDR_QHDR_DBSPACE configuration setting:         rootdbs
CDR_QUEUEMEM:
  CDR_QUEUEMEM configuration setting:            4096
CDR_SERIAL:
  CDR_SERIAL configuration setting:              0, 0
CDR_SUPPRESS_ATSRISWARN:
  CDR_SUPPRESS_ATSRISWARN configuration setting:  [None suppressed]
ENCRYPT_CDR:
  ENCRYPT_CDR configuration setting:              0
ENCRYPT_CIPHERS:
  ENCRYPT_CIPHERS configuration setting:          [None configured]
ENCRYPT_MAC:
  ENCRYPT_MAC configuration setting:              [None configured]
ENCRYPT_MACFILE:
  ENCRYPT_MACFILE configuration setting:          [None configured]
ENCRYPT_SWITCH:
  ENCRYPT_SWITCH configuration setting:           0,0
CDR_ENV environment variable settings:
  CDR_LOGDELTA:
    CDR_LOGDELTA configuration setting:          0
  CDR_PERFLOG:
    CDR_PERFLOG configuration setting:           0
  CDR_ROUTER:
    CDR_ROUTER configuration setting:            0
  CDR_RMSCALEFACT:
    CDR_RMSCALEFACT configuration setting:       0
  CDRSITES_731:
    CDRSITES_731 configuration setting:          [None configured]
  CDRSITES_92X:
    CDRSITES_92X configuration setting:          [None configured]
  CDRSITES_10X:
    CDRSITES_10X configuration setting:          [None configured]

```

onstat -g ckp: Print checkpoint history and configuration recommendations

Syntax:

▶▶ onstat -g ckp ◀◀

The **onstat -g ckp** option prints checkpoint history and displays configuration recommendations if a suboptimal configuration is detected.

Example Output

```

IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:01:20 -- 299368 Kbytes
Auto Checkpoints=On  RTO_SERVER_RESTART=60 seconds  Estimated recovery time 7 seconds

Interval  Clock      Trigger  LSN      Total  Flush  Block  #      Ckpt  Wait  Long  #Dirty
          Time          LSN      Time    Time   Time   Waits  Time  Time  Time  Buffers
1         18:41:36 Startup  1:f8     0.0    0.0    0.0    0      0.0  0.0  0.0    4
2         18:41:49 Admin   1:11c12cc 0.3    0.2    0.0    1      0.0  0.0  0.0  2884
3         18:42:21 Llog    8:188    2.3    2.0    2.0    1      0.0  2.0  2.0  14438
4         18:42:44*User  10:19c018 0.0    0.0    0.0    1      0.0  0.0  0.0   39
5         18:46:21 RTO    13:188   54.8   54.2   0.0    30     0.6  0.4  0.6  68232

          Physical Log  Logical Log
          Total  Avg  Total  Avg
Dskflu   /Sec  Pages /Sec  Pages /Sec
4         3      0     1      0
2884     1966  163   4549  379
7388     318   10   65442 2181
39        536   21   20412  816
1259     210757 1033  150118 735

Max Plog  Max Llog  Max Dskflush  Avg Dskflush  Avg Dirty  Blocked
pages/sec pages/sec  Time          pages/sec     pages/sec   Time
8796     6581     54           43975         2314        0
    
```

Figure 15-18. onstat -g ckp Output

Output Description

Auto Checkpoints

Indicates if the AUTO_CKPTS configuration parameter is on or off

RTO_SERVER_RESTART

Displays the RTO time in seconds. Zero (0) means that RTO is off.

Estimated recovery time ## seconds

Indicates the estimated recovery time if the data server stops responding. This value only appears if RTO_SERVER_RESTART is active.

Interval

Checkpoint interval ID

Clock Time

Clock time when checkpoint occurred.

Trigger

Event that triggered the checkpoint. An asterisk (*) indicates that the checkpoint requested was a transaction-blocking checkpoint. Events

include Admin, Startup, CKPTINTVL, LongTX, Recovery, Backup, Plog, Llog, Misc, RTO, CDR, Pload, Conv/Rev, Reorg, HDR, User, and Lightscan

LSN Logical log position where checkpoint is recorded

Total Time

Total checkpoint duration in seconds from request time to checkpoint completion

Flush Time

Time, in seconds, to flush bufferpools

Block Time

Individual transaction blocking time, in seconds, for that particular checkpoint

Waits

Number of transactions blocked waiting for checkpoint

Ckpt Time

Time, in seconds, for all transactions to recognize a requested checkpoint

Wait Time

Average time, in seconds, transactions waited for checkpoint

Long Time

Longest amount of time, in seconds, a transaction waited for checkpoint

Dirty Buffers

Number of dirty buffers flushed to disk during checkpoint

Dskflu/sec

Number of buffers flushed per second

Physical Log Total Pages

Total number of pages physically logged during checkpoint interval

Physical Log Avg/Sec

Average rate of physical log activity during checkpoint interval

Logical Log Total Pages

Total number of pages logically logged during checkpoint interval

Logical Log Avg/Sec

Average rate of logical log activity during checkpoint interval

Max Plog pages/sec

Maximum rate of physical log activity during checkpoint interval

Max Llog pages/sec

Maximum rate of logical log activity during checkpoint interval

Max Dskflush Time

Maximum time, in seconds, to flush bufferpools to disk

Avg Dskflush pages/sec

Average rate bufferpools are flushed to disk

Avg Dirty pages/sec

Average rate of dirty pages between checkpoints

Blocked Time

Longest blocked time, in seconds, since last time the data server was started

If the IDS data server detects a configuration that is less than optimal, a performance advisory message with tuning recommendations appears below the checkpoint history. This performance advisory message also appears in the message log. Following are examples of performance advisory messages:

Physical log is too small for bufferpool size. System performance may be less than optimal.
Increase physical log size to at least %ldKb

Physical log is too small for optimal performance.
Increase the physical log size to at least \$ldKb.

Logical log space is too small for optimal performance.
Increase the total size of the logical log space to at least %ld Kb.

Transaction blocking has taken place. The physical log is too small.
Please increase the size of the physical log to %ldKb

Transaction blocking has taken place. The logical log space is too small.
Please increase the size of the logical log space to %ldKb

onstat -g cmsm: Print Connection Manager information

Syntax:

▶▶ onstat -g cmsm ◀◀

The **onstat -g cmsm** option prints Connection Manager information, such as the number of Connection Manager daemons attached to the current database instance, as well as the number of connections each daemon has processed.

Example Output

```
onstat -g cmsm
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 12:52:50 -- 325632 Kbytes
CM name  host      sla      define      foc      flag  connections
cm1      bia      oltp_cm1 primary      SDS+HDR+RSS,0  1      5
cm1      bia      report_cm1 (SDS+RSS)  SDS+HDR+RSS,0  1      16
```

Figure 15-19. onstat -g cmsm Output with One Connection Manager Instance

```
onstat -g cmsm
IBM Informix Dynamic Server Version 11.50.F      -- Read-Only (SDS) -- Up 00:08:38 -- 325632 Kbytes
CM name  host      sla      define      foc      flag  connections
cm1      bia      oltp_cm1 primary      SDS+HDR+RSS,0  1      5
cm1      bia      report_cm1 (SDS+RSS)  SDS+HDR+RSS,0  1      16
cm2      argo     oltp_cm2 primary      SDS+HDR+RSS,0  0      0
cm2      argo     report_cm2 SDS+HDR      SDS+HDR+RSS,0  0      1
```

Figure 15-20. onstat -g cmsm Output with Two Connection Manager Instances

Output Description

CM name

Connection Manager instance name

host Host name

sla Service level agreement name

define Service level agreement definition. Can include server types (Primary, SDS, RSS, HDR) or server names

foc Failover configuration

flag Failover arbitrator indicator. The value of the flag field is derived from the following bit values:

0 The Connection Manager automatic failover logic is not enabled

1 The Connection Manager automatic failover logic is enabled

2 The Connection Manager has detected an active primary server in the cluster

4 The Connection Manager has detected that the primary server is not responding

8 Automatic failover is disabled; no failover processing will be performed by the Connection Manager.

The flag value can be a cumulative value of the bit numbers. For example, a flag value of 10 is a combination of the bits 2 and 8.

For information about Connection Manager failover arbitration, see the *IBM Informix Administrator's Guide*.

connections

The number of connections that are processed with the service level agreement

onstat -g con: Print condition and thread information

Syntax:

▶ onstat -g con ◀

The **onstat -g con** option prints information on conditions and the threads that are waiting for them.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 18:47:42
-- 101376 Kbytes
Conditions with waiters:
cid      addr          name          waiter  waittime
271      c63d930      netnorm      1511    6550
```

Figure 15-21. onstat -g con Output

Output Description

cid Condition identifier

addr Condition control block address

name Name of condition the thread is waiting on

waiter ID of thread waiting on condition

waittime

Time, in seconds, thread has been waiting on this condition

onstat -g cpu: Print runtime statistics

Syntax:

▶▶ onstat -g cpu ▶▶

The **onstat -g cpu** command prints information about runtime statistics for all the threads running in the server.

Example Output

```
onstat -g cpu
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 12:52:50 -- 325632 Kbytes
Thread CPU Info:
tid  name          vp      Last Run          CPU Time    #scheds    status
2    lio vp 0        3lio*  07/18 08:35:35   0.0000     1      IO Idle
3    pio vp 0        4pio*  07/18 08:35:36   0.0102     2      IO Idle
4    aio vp 0        5aio*  07/18 08:35:47   0.6876    68      IO Idle
5    msc vp 0        6msc*  07/18 11:47:24   0.0935    14      IO Idle
6    main_loop()    1cpu*  07/18 15:02:43   2.9365   23350   sleeping secs: 1
7    soctcppoll     7soc*  07/18 08:35:40   0.1150     1      running
8    soctcpio      8soc*  07/18 08:35:40   0.0037     1      running
9    soctcplst     1cpu*  07/18 11:47:24   0.1106    10      sleeping forever
10   soctcplst     1cpu*  07/18 08:35:40   0.0103     6      sleeping forever
11   flush_sub(0)  1cpu*  07/18 15:02:43   0.0403   23252   sleeping secs: 1
12   flush_sub(1)  1cpu*  07/18 15:02:43   0.0423   23169   sleeping secs: 1
13   flush_sub(2)  1cpu*  07/18 15:02:43   0.0470   23169   sleeping secs: 1
14   flush_sub(3)  1cpu*  07/18 15:02:43   0.0407   23169   sleeping secs: 1
15   flush_sub(4)  1cpu*  07/18 15:02:43   0.0307   23169   sleeping secs: 1
16   flush_sub(5)  1cpu*  07/18 15:02:43   0.0323   23169   sleeping secs: 1
17   flush_sub(6)  1cpu*  07/18 15:02:43   0.0299   23169   sleeping secs: 1
18   flush_sub(7)  1cpu*  07/18 15:02:43   0.0314   23169   sleeping secs: 1
19   kaio         1cpu*  07/18 14:56:42   1.4560  2375587 IO Idle
20   aslogflush   1cpu*  07/18 15:02:43   0.0657   23166   sleeping secs: 1
21   btscanner_0  1cpu*  07/18 15:00:53   0.0484    784    sleeping secs: 61
37   onmode_mon   1cpu*  07/18 15:02:43   0.3467   23165   sleeping secs: 1
43   dbScheduler  1cpu*  07/18 14:58:14   1.6613    320    sleeping secs: 31
44   dbWorker1    1cpu*  07/18 13:48:10   0.4264    399    sleeping forever
45   dbWorker2    1cpu*  07/18 14:48:11   1.9346   2936    sleeping forever
94   bf_priosweep() 1cpu*  07/18 15:01:42   0.0431    77     cond wait bp_cond
```

Figure 15-22. onstat -g cpu Output

Output Description

tid Thread ID

name Thread name

vp ID of the virtual processor in which the thread is running

Last Run

Timestamp when the thread last ran

CPU Time

Time taken until now by the thread

#schedules

Number of times the thread was scheduled to run

status State of the thread. Following are the possible status values:

- cond wait
- IO Idle
- join wait
- mutex wait
- ready
- sleeping
- terminated
- running
- yield

onstat -g dbc: Print dbScheduler and dbWorker thread statistics

Syntax:

▶▶ onstat -g dbc ◀◀

The **onstat -g dbc** option prints statistics about dbScheduler and dbWorker threads, which are part of the database scheduler and the SQL Administration API.

Example Output

```

IBM Informix Dynamic Server Version 11.50.FC1 -- On-Line -- Up 00:01:20 -- 299368 Kbytes
Worker Thread(0) 46fa6f10
=====
Task: 47430c18
Task Name: mon_config_startup
Task ID: 3
Task Type: STARTUP SENSOR
Last Error
  Number -310
  Message Table (informix.mon_onconfig) already exists in database.
  Time 09/11/2007 11:41
  Task Name mon_config_startup
Task Execution: onconfig_save_diffs

WORKER PROFILE
  Total Jobs Executed 10
  Sensors Executed 8
  Tasks Executed 2
  Purge Requests 8
  Rows Purged 0

Worker Thread(1) 46fa6f80
=====
Task: 4729fc18
Task Name: mon_sysenv
Task ID: 4
Task Type: STARTUP SENSOR
Task Execution: insert into mon_sysenv select 1, env_name, env_value FROM sysmaster:sysenv

WORKER PROFILE
  Total Jobs Executed 3
  Sensors Executed 2
  Tasks Executed 1
  Purge Requests 2
  Rows Purged 0

Scheduler Thread 46fa6f80
=====
Run Queue
  Empty
Run Queue Size 0
Next Task 7
Next Task Waittime 57

```

Figure 15-23. onstat -g dbc Output

Output Description

Worker Thread

Address of the worker thread in shared memory

Task Name of the last executed task

Task ID

tk_id from the **sysadmin:ph_task** table for this task

Task Type

Type of the task

Last Error

Error number, error message, time (in seconds), and task name from the last execution of this task

Task Execution

SQL statement or SPL procedure or routine executed as part of the task

WORKER PROFILE

The dbWorker thread profile data shows the total jobs executed, number of sensors executed, number of tasks executed, number of purge requests, and the number of rows purged from the resultant table for the task

Scheduler Thread

Address of the scheduler thread in shared memory

Run Queue

The tk_id for the next scheduled task. If no task is scheduled, the value is Empty.

Run Queue Size

The number of tasks that are waiting to be executed by the dbWorker thread

Next Task

The next task that will be scheduled to be executed

Next Task Waittime

The number of seconds before the Next Task will be scheduled for execution

onstat -g ddr: Print ER database log reader status

Syntax:

▶▶ onstat -g ddr ◀◀

The **onstat -g ddr** option prints the status of the Enterprise Replication database log reader. The **ddr**, or **ddr_snoopy**, is an internal component of Enterprise Replication that reads the log buffers and passes information to the grouper.

You can use the information from the **onstat -g ddr** command to monitor *replay position* in the log file and ensure replay position is never overwritten (which can cause loss of data). The replay position is the point from where, if a system failure occurs, Enterprise Replication starts re-reading the log information into the log update buffers. All the transactions generated before this position at all the target servers have been applied by Enterprise Replication or safely stored in stable queue space.

The **onstat -g ddr** output shows you a snapshot of the replay position, the *snoopy position*, and the *current position*. The snoopy position identifies the position of the **ddr_snoopy** thread in the logical logs. The **ddr_snoopy** has read the log records up until this point. The current position is the position where the server has written its last logical log record.

The *log needs* position is based on replay position and is set at a certain distance from replay position, for example, at seventy percent of the log file. The remainder of the circular log file comprises the DDR BLOCK zone. As messages are acknowledged or stored in the stable queue, the replay position, and hence also the log needs position, should advance. If you notice that replay position is not advancing, this can mean that the stable queue is full or a remote server is down.

If log reading is blocked, data might not be replicated until the problem is resolved. If the block is not resolved, the database server might overwrite the read (**ddr_snoopy**) position, which means that data will not be replicated. If this occurs, you must manually resynchronize the source and target databases.

For servers of Version 9.4, and later, you can enable dynamic log creation by setting the `CDR_MAX_DYNAMIC_LOGS` configuration parameter in the `ONCONFIG` file. If the current position reaches the log needs position, instead of going into a blocked state, Enterprise Replication automatically adds another log file. If this option is set, the **onstat -g ddr** command prints the number of dynamic log requests made. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Example Output

The following sample output from the **onstat ddr** command shows the replay position, snoopy position, and current position highlighted.

```

DDR -- Running
# Event Snoopy   Snoopy  Replay  Replay  Current  Current
Buffers ID    Position ID      Position ID      Position
528      24    165018  24      6a018   24      166000

Log Pages Snooped:      From      From      Tossed
                       Cache     Disk (LBC full)
                       247      111      0

Total dynamic log requests: 0
DDR events queue

Type  TX id  Partnum  Row id

```

Figure 15-24. `onstat -g ddr` Output

onstat -g dic: Print table information

Syntax:

►►—onstat— -g—dic—◄◄

Without any parameters the **onstat -g dic** option prints one line of information for each table cached in the shared-memory dictionary. If a table name is specified, prints internal SQL information for that table.

For more information see the *Performance Guide*.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 18:47:42
-- 101376 Kbytes
Dictionary Cache: Number of lists: 31, Maximum list size: 10
list#  size  refcnt  dirty?  heapptr          table name
-----
  1     3     1     no     14b5d890        wbe@oninit_shm:informix.t0010url
           1     no     14cbb820        wbe@oninit_shm:informix.t9051themeval
           0     no     14b63c20        wbe@oninit_shm:informix.t0060hits

  2     2     0     no     14b97420        wbe@oninit_shm:informix.t0120import
           1     no     14b6c820        wbe@oninit_shm:informix.t9110domain

  3     3     0     no     14bce020        wbe@oninit_shm:informix.t0150url
           0     no     14d3d820        contact@oninit_shm:informix.wbtags
           0     no     14c87420        wbe@oninit_shm:informix.wbtags

  4     1     0     no     14b7a420        drug@oninit_shm:viagra.product      .....
Total number of dictionary entries: 36

```

Figure 15-25. `onstat -g dic` Output

Output Description

list# Data dictionary hash chain ID

size Number of entries in this hash

refcnt Number of SQL statements currently referencing one of the cache entries.

dirty? Whether the entry has been modified since last written to disk.

heapptr

Address for the heap used to store this table

table name

Name of table in cache

onstat -g dis: Print database server information

Syntax:

▶▶ onstat -g dis ◀◀

Prints a list of database servers and their status, and information about each database server, **INFORMIXDIR**, **sqlhosts** file, **ONCONFIG** file, and hostname. You can use this option with the database server in any mode, including offline.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 18:47:42
-- 101376 Kbytes
There are 2 servers found
Server      : ol_tuxedo
Server Number : 53
Server Type  : IDS
Server Status : Up
Server Version: IBM Informix Dynamic Server Version 11.50.UC1
Shared Memory : 0xa000000
INFORMIXDIR  : /local1/engines/ol_tuxedo/dist
ONCONFIG     : /local1/engines/ol_tuxedo/dist/etc/onconfig.ol_tuxedo
SQLHOSTS    : /local1/engines/ol_tuxedo/dist/etc/sqlhosts
Host        : avocet

Server      : ol_9next
Server Number : 0
Server Type  : IDS
Server Status : Down
Server Version:
Shared Memory : 0
INFORMIXDIR  : /local1/engines/ol_9next/dist
ONCONFIG     :
SQLHOSTS    :
Host        :

```

Figure 15-26. `onstat -g dis` Output

Output Description

Server Server name

Server Number
Number of the server.

Server Type
Type of server

Server Status
Up means that the server is online, Down means that the server is offline

Server Version
Version of the server

Shared Memory
Location of the shared memory address

INFORMIXDIR
Location of the `$INFORMIXDIR/` directory on UNIX and in the `%INFORMIXDIR%\` directory on Windows.

ONCONFIG
Location of the ONCONFIG file

SQLHOSTS
Location of the `sqlhosts` file

Host Host name of the server

onstat -g dll: Print dynamic libraries list

Syntax:

```
▶▶ onstat -g dll ◀◀
```

Prints a list of dynamic libraries that have been loaded.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 18:47:42 -- 101376 Kbytes
Datablades:
addr      slot  vp  baseaddr  filename
140090fc  2     1   fe64d4e0  MYPATH/informix/extend/web.xxxxxx/web.bld
141c70fc  2     2   fe7cd4e0
141ca0fc  3     3   fe7cd4e0
```

Figure 15-27. *onstat -g dll* Output

Output Description

addr DLL address

slot Slot number entry in the library table

vp Virtual processor ID

baseaddr
Virtual processor base address

filename
DLL filename

onstat -g dmp: Print raw memory

Syntax:

```
▶▶ onstat -g dmp ◀◀
```

The **onstat -g dmp** option prints raw memory at a given address for a number of given bytes. Each address and length must be within the allocated memory shown from **onstat -g seg** output.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:03:58 -- 1058816 Kbytes

address          bytes in mem
000000010a000000: e0b70001 0000120e ffffffff 00000000  .....
000000010a000010: 00000000 00000000 00000000 00000000  .....
000000010a000020: 00000000 00000000 00000000 00000000  .....
000000010a000030: 00000000 00000000 00000000 00000000  .....
000000010a000040: 00000000 00000000 00000000 00000000  .....
000000010a000050: 00000000 00000000 00000000 00000000  .....
000000010a000060: 00000000
```

Figure 15-28. *onstat -g dmp* Output

Output Description

address

Memory address of the raw memory

bytes in mem

Hexadecimal and ASCII representations of the memory contents

onstat -g dri: Print High-Availability Cluster information

Syntax:

▶ onstat -g dri ◀

The **onstat -g dri** option prints information about high-availability cluster information on the current server.

Example Output

Data Replication:				
Type	State	Paired server	Last DR CKPT (id/pg)	Supports Proxy Writes
primary	on	B_151162	554 / 558	Y
DRINTERVAL	30			
DRTIMEOUT	30			
DRAUTO	0			
DRLOSTFOUND	/vobs/tristarm/sqldist/etc/dr.lostfound			
DRIDXAUTO	0			
ENCRYPT_HDR	0			

Figure 15-29. onstat -g dri Output

Output Description

Type Current type of server: primary, secondary, or standard

State on or off

Paired server

Name of the primary or secondary server that this server is paired with

Last DR CKPT

Last checkpoint ID and page

Supports Proxy Writes

Displays whether the server is currently configured to allow secondary server updates. **Y** = supports secondary server updates, **N** = does not support secondary server updates.

The second section lists the values of the following configuration parameters in the ONCONFIG file:

- DRINTERVAL
- DRTIMEOUT
- DRAUTO
- DRLOSTFOUND
- DRIDXAUTO
- ENCRYPT_HDR

onstat -g dsc: Print distribution cache information

Syntax:

```
▶▶ onstat -g dsc ◀◀
```

Prints a list of distribution cache information.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:54:52
-- 101376 Kbytes
Distribution Cache:
  Number of lists           : 31
  PC_POOLSIZE               : 50
  Number of entries         : 0
  Number of entries in use  : 0
Distribution Cache Entries:
list# id ref_cnt dropped? heap_ptr      distribution name
-----
Distribution Cache is empty.
```

Figure 15-30. onstat -g dsc Output

Output Description

The first section of output describes the distribution cache.

Number of lists

Number of lists in the distribution cache

PC_POOLSIZE

Number of entries that can be cached at one time

Number of entries

Number of entries in the distribution cache

Number of entries in use

Number of entries being used

The second section of output describes the distribution cache entries.

list# Distribution cache hash chain ID

id Number of hash entries

ref_cnt

Number of statements referencing a cache entry

dropped?

Whether this entry has been dropped since being added to the cache

heap_ptr

Heap address used to store this entry

distribution name

The name of the distribution in the cache

onstat -g dss: Print ER environment data

Syntax:

```
▶▶ onstat -g dss [modifier] ▶▶
```

The **onstat -g dss** command prints detailed statistical information about the activity of individual data sync threads in an Enterprise Replication environment. The data sync thread applies the transaction on the target server. Statistics include the number of applied transactions and failures and when the last transaction from a source was applied.

The **onstat -g dss** command has the following formats:

```
onstat -g dss
onstat -g dss modifier
```

The following table describes the values for *modifier*.

Modifier	Action
UDR	Prints summary information about any UDR invocations by the data sync threads.
UDRx	Prints expanded information (including a summary of error information) about any UDR invocations by the data sync threads. The ProcId column lists the UDR procedure ID.

Example Output

In the following example, only one data sync thread is currently processing the replicated data. It has applied a total of one replicated transaction and the transaction was applied at 2004/09/13 18:13:10. The Processed Time field shows the time when the last transaction was processed by this data sync thread.

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:00:28 -- 28672 Kbytes
DS thread statistic
cmtTime Tx Tx Tx Last Tx
Name < local Committed Aborted Processed Processed Time
-----
CDRD_1 0 1 0 1 (1095117190) 2004/09/13 18:13:10
Tables (0.0%):
Databases: test
CDR_DSLOCKWAIT = 1
CDR_DSCLOSEINTERVAL = 60
```

Figure 15-31. *onstat -g dss* Output

onstat -g dtc: Print delete table cleaner statistics

Syntax:

```
▶▶ onstat -g dtc ▶▶
```

The **onstat -g dtc** command prints statistics about the delete table cleaner. The delete table cleaner removes rows from the delete table when they are no longer needed.

The `-g dtc` option is used primarily as a debugging tool and by IBM Support.

Example Output

In the following example, the thread name of the delete table cleaner is **CDRDTCleaner**. The total number of rows deleted is **1**. The last activity on this thread occurred at 2006/09/13 18:47:19. The delete table for replicate **rep1** was last cleaned at 2006/09/13 18:28:25.

```

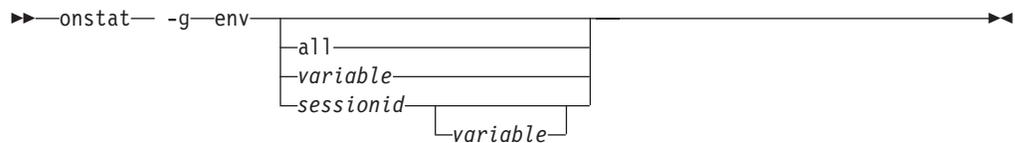
IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 00:59:15 -- 28672 Kbytes
-- Delete Table Cleanup Status as of (1095119368) 2006/09/13 18:49:28
  thread          = 49 <CDRDTCleaner>
  rows deleted    = 1
  lock timeouts  = 0
  cleanup interval = 300
  list size       = 3
  last activity   = (1095119239) 2006/09/13 18:47:19
Id      Database      Server      Last Cleanup Time
  Replicate
=====
000001  test              (1095118105) 2004/09/13 18:28:25
         rep1        g_bombay     (1095118105) 2006 /09/1318:28:25
         rep1        g_delhi      (1095118105) 2006 /09/13 18:28:25
000002  test              <never cleaned>

```

Figure 15-32. `onstat -g dtc` Output

onstat -g env: Print environment variable values

Syntax:



The `onstat -g env` option displays the values of environment variables the database server currently uses. You can specify one of the following invocations.

Invocation	Explanation
<code>onstat -g env</code>	Displays the settings of environment variables when the database server was started Does not display environment variables that have not been set explicitly.
<code>onstat -g env all</code>	Displays the settings used by all sessions This display is the same as the output of <code>onstat -g env</code> and <code>onstat -g env sessionid</code> iteratively on all current sessions.
<code>onstat -g env variable</code>	Displays the default value of the specified environment variable This <code>variable</code> argument eliminates the need to pipe the output to <code>grep</code> (or some other utility) to locate an environment variable among many that might be set.

Invocation	Explanation
onstat -g env <i>sessionid</i>	Displays the settings that a specific session uses. This display includes the following values: <ul style="list-style-type: none"> • Set in the environment of the session • Assigned by the database server, as onstat -g env displays
onstat -g env <i>sessionid variable</i>	Displays the value of the specified environment variable that the specified session uses The <i>sessionid</i> and <i>variable</i> arguments eliminate the need to pipe the output to grep (or some other utility) to locate an environment variable among many that might be set.

The **onstat -g env** option displays the current setting of an environment variable and the complete list of values each time the variable was set in the environment. For example, if PDQPRIORITY is set to 10 in the **.informix.rc** file and set to 55 in the shell environment, **onstat -g env** displays both values.

However, if you change the PDQPRIORITY with the **onmode -q pdqpriority *sessionid*** option, **onstat -g env** does not display the new value for the session. The **onstat -g env** option displays only the values of environment variables set in the environment. It does not display values modified while the session is running.

You might want to display the values of environment variables in the following situations:

- The database server instance has been up for months, and you cannot remember the setting of an environment variable (such as the server locale setting **SERVER_LOCALE**).
- You want to display the complete list of values for an environment variable to identify when an environment variable has been set in multiple places.
- Environment files on disk might have changed or been lost in the interim.
- A support engineer wants to know settings of specific environment variables.

The following figure shows the output for the **onstat -g env** option.

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 4 days 17:08:43
-- 45056 Kbytes

Variable          Value [values-list]
DBDATE            DMY4/
DBDELIMITER       |
DBPATH            .
DBPRINT           lp -s
DBTEMP            /tmp
INFORMIXDIR       /build2/9.30/tristarm/sqldist
                  [/build2/9.30/tristarm/sqldist]
                  [/usr/informix]
INFORMIXSERVER    parata930
INFORMIXTERM      termcap
LANG              C
LC_COLLATE        C
LC_CTYPE          C
LC_MONETARY       C
LC_NUMERIC        C
LC_TIME           C
LD_LIBRARY_PATH   /usr/openwin/lib:/lib:/usr/lib
LKNOTIFY          yes
LOCKDOWN          no
NODEFDAC          no
NON_M6_ATTRS_OK  1
PATH              /build2/9.30/tristarm/sqldist/bin:::
                  /root/bin:/opt/SUNWspro/bin:/usr/ccs/bin:
                  /usr/openwin/bin:/usr/sbin:/usr/bin:/usr
                  /local/binSERVER_LOCALE      en_US.819
SHELL             /bin/ksh
SINGLELEVEL        no
SUBQCACHESZ       10
TBCONFIG          onconfig
TERM              xterm
                  [xterm]
                  [dumb]
TERMCAP           /etc/termcap
TZ                GB

```

Figure 15-33. `onstat -g env` Output

onstat -g ffr: Print free fragments

Syntax:

```

▶▶ onstat -g ffr name-session id ▶▶

```

The `onstat -g ffr` option prints free fragments for a pool of shared memory.

Example Output

```
Free list for pool name dfm_pool:
addr      size
10ac92f20 224
10ac8c0c0 3904
```

Figure 15-34. `onstat -g ffr` Output

Output Description

addr Pool fragment address
size Fragment size, in bytes

onstat -g glo: Print global multithreading information

Syntax:

▶▶ `onstat -g glo` ◀◀

The `onstat -g glo` option prints global multithreading information.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1    -- On-Line -- Up 11 days 02:12:45 -- 144676 Kbytes

MT global info:
sessions threads  vps      lngspins
0          28      11      0

          sched calls      thread switches  yield 0   yield n   yield forever
total:    19577737            16256991        3688561  11340793  280099
per sec:  0                  0                0         0         0

Virtual processor summary:
class    vps      usercpu  syscpu   total
cpu      1        107.40   8.09     115.49
aio      6         4.19    94.42    98.61
lio      1         0.68    3.54     4.22
pio      1         0.03    0.33     0.36
adm      1         0.00    0.05     0.05
msc      1         0.00    0.01     0.01
total    11        112.30  106.44   218.74

Individual virtual processors:
vp  pid   class  usercpu  syscpu  total  Thread  Eff
1   5738  cpu    107.40   8.09    115.49  246.14  46%
2   5739  adm     0.00    0.05    0.05    0.00    0%
3   5740  lio     0.68    3.54    4.22    36.44   11%
4   5741  pio     0.03    0.33    0.36    7.50    4%
5   5742  aio     2.03    51.92   53.95   358.09  15%
```

Figure 15-35. `onstat -g glo` Output

Output Description

Virtual Processor Summary

class The type of virtual processor.
vps The number of instances of this class of VP.

usercpu

The total user time, in seconds, that this class of VP has spent running on the CPU.

syscpu

The total system time, in seconds, this class of VP has spent running on the CPU.

total The total number of virtual processors, user time, and system time.

Individual Virtual Processors

vp The virtual processor number.

pid The Process ID of this oninit process.

class The type of virtual processor.

usercpu

The total user time, in seconds, that this class of VP has spent running on the CPU.

syscpu

The total system time, in seconds, that this class of VP has spent running on the CPU.

total The total number of virtual processors, user time, and system time.

Thread

The total time the threads ran on this VP.

Eff Efficiency. The ratio of total CPU time (usercpu+syscpu) to total time threads that ran on this VP. The total CPU time is the usercpu + syscpu.

onstat -g grp: Print ER grouper statistics**Syntax:**

```
▶▶ onstat -g grp modifier ▶▶
```

The **onstat -g grp** command prints statistics about the Enterprise Replication grouper. The grouper evaluates the log records, rebuilds the individual log records into the original transaction, packages the transaction, and queues the transaction for transmission.

The **-g grp** option is used primarily as a debugging tool and by Technical Support.

The **onstat -g grp** command has the following formats:

```
onstat -g grp
onstat -g grp modifier
```

The following table describes the values for *modifier*.

Modifier	Action
A	Prints all the information printed by the G, T, P, E, R, and S modifiers
E	Prints grouper evaluator statistics
Ex	Prints grouper evaluator statistics, expands user-defined routine (UDR) environments

G	Prints grouper general statistics
L	Prints grouper global list
Lx	Prints grouper global list, expands open transactions
M	Prints grouper compression statistics
Mz	Clears grouper compression statistics
P	Prints grouper table partition statistics
pager	Prints grouper paging statistics
R	Prints grouper replicate statistics
S	Prints grouper serial list head (The serial list head is the first transaction in the list, that is, the next transaction that will be placed in the send queue.)
Sl	Prints grouper serial list (The serial list is the list of transactions, in chronological order.)
Sx	Prints grouper serial list, expands open transactions
T	Prints grouper transaction statistics
UDR	Prints summary information about any UDR invocations by the grouper threads
UDRx	Prints expanded information (including a summary of error information) about any UDR invocations by the grouper threads The ProcId column lists the UDR procedure ID.

Example Output

This section contains sample output from various **onstat -g grp** *modifier* commands. The following sample shows output for the **onstat -g grp** command.

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:47:07
-- 28672 Kbytes
Grouper at 0xb014018:
Last Idle Time: (1095122236) 2004/09/13 19:37:16
RSAM interface ring buffer size: 528
RSAM interface ring buffer pending entries: 0
Eval thread interface ring buffer size: 48
Eval thread interface ring buffer pending entries: 0
Log update buffers in use: 0
Max log update buffers used at once: 5
Log update buffer memory in use: 0
Max log update buffer memory used at once: 320
Updates from Log: 16
Log update links allocated: 512
Blob links allocated: 0
Conflict Resolution Blocks Allocated: 0
Memory pool cache: Empty
Last Tx to Queuer began : (1095118105) 2004/09/13 18:28:25
Last Tx to Queuer ended : (1095118105) 2004/09/13 18:28:25
Last Tx to Queuer log ID, position: 12,23
Open Tx: 0
Serial Tx: 0
Tx not sent: 0
Tx sent to Queuer: 2
Tx returned from Queuer: 2
Events sent to Queuer: 7
Events returned from Queuer: 7
Total rows sent to Queuer: 2
Open Tx array size: 1024
Table 'tab' at 0xae8ebb0 [ CDRShadow ]
Table 'tab12' at 0xae445e0 [ CDRShadow ]
```

```

Grouped Table Partitions:
Slot 312...
'tab' 1048888
Slot 770...
'tab12' 3145730
Slot 1026...
'tab12' 4194306
Repl links on global free list: 2
Evaluators: 3
  Evaluator at 0xb03d030 ID 0 [Idle:Idle] Protection:unused
    Eval iteration: 1264
    Updates evaluated: 0
    Repl links on local free list: 256
    UDR environment table at 0xb03d080
      Number of environments:      0
      Table memory limit   :      25165
      Table memory used    :          0
      SAPI memory limit    :     131072
      SAPI memory used     :          0
      Count failed UDR calls:      0
  Evaluator at 0xb03d0d8 ID 1 [Idle:Idle] Protection:unused
    Eval iteration: 1265
    Updates evaluated: 2
    Repl links on local free list: 254
    UDR environment table at 0xb03d128
      Number of environments:      0
      Table memory limit   :      25165
      Table memory used    :          0
      SAPI memory limit    :     131072
      SAPI memory used     :          0
      Count failed UDR calls:      0
  Evaluator at 0xb03d180 ID 2 [Idle:Idle] Protection:unused
    Eval iteration: 1266
    Updates evaluated: 4
    Repl links on local free list: 256
    UDR environment table at 0xb03d1d0
      Number of environments:      0
      Table memory limit   :      25165
      Table memory used    :          0
      SAPI memory limit    :     131072
      SAPI memory used     :          0
      Count failed UDR calls:      0
    Total Free Repl links 768
Replication Group 6553601 at 0xb0a8360
  Replication at 0xb0a82b0 6553601:6553601 (tab) [ NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
      CDR Shadow: offset 0, size 8
      In Order: offset 8, size 10
Replication Group 6553602 at 0xb0a8480
  Replication at 0xb0a83d0 6553602:6553602 (tab12) [ Ignore Stopped
NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
      CDR Shadow: offset 0, size 8
      In Order: offset 8, size 16

```

The following example shows output for the **onstat -g grp E** command. The field **Evaluators: 4** indicates that there are four evaluation threads configured for the system.

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 02:07:10 -- 36864 Kbytes
Repl links on global free list: 0 Evaluators: 4
  Evaluator at 0xba71840 ID 0 [Idle:Idle] Protection: unused
    Eval iteration: 1007
    Updates evaluated: 0
    Repl links on local free list: 256
    UDR environment table at 0xba71890
      Number of environments:      0
      Table memory limit   :      16777
      Table memory used    :      0
      SAPI memory limit    :     131072
      SAPI memory used     :      0
      Count failed UDR calls:      0
  Evaluator at 0xba718f0 ID 1 [Idle:Idle] Protection: unused
    Eval iteration: 1007
    Updates evaluated: 0
    Repl links on local free list: 256
    UDR environment table at 0xba71940
      Number of environments:      0
      Table memory limit   :      16777
      Table memory used    :      0
      SAPI memory limit    :     131072
      SAPI memory used     :      0
      Count failed UDR calls:      0
  Evaluator at 0xba8c260 ID 2 [Idle:Idle] Protection: unused
    Eval iteration: 1007
    Updates evaluated: 0
    Repl links on local free list: 256
    UDR environment table at 0xba8c2b0
      Number of environments:      0
      Table memory limit   :      16777
      Table memory used    :      0
      SAPI memory limit    :     131072
      SAPI memory used     :      0
      Count failed UDR calls:      0
  Evaluator at 0xbaac2a0 ID 3 [Idle:Idle] Protection: unused
    Eval iteration: 1007
    Updates evaluated: 0
    Repl links on local free list: 256
    UDR environment table at 0xbaac2f0
      Number of environments:      0
      Table memory limit   :      16777
      Table memory used    :      0
      SAPI memory limit    :     131072
      SAPI memory used     :      0
      Count failed UDR calls:      0
Total Free Repl links 1024

```

Figure 15-36. `onstat -g grp E` Output

The following example shows output for the `onstat -g grp G` command.

```

IBM Informix Dynamic Server Version 11.50.UC1    -- On-Line -- Up 02:08:56 -- 36864 Kbytes
Grouper at 0xb8ab020:
Last Idle Time: (1095115397) 2004/09/13 17:43:17
RSAM interface ring buffer size: 1040
RSAM interface ring buffer pending entries: 0
Eval thread interface ring buffer size: 64
Eval thread interface ring buffer pending entries: 0
Log update buffers in use: 0
Max log update buffers used at once: 1
Log update buffer memory in use: 0
Max log update buffer memory used at once: 64
Updates from Log: 1
Log update links allocated: 512
Blob links allocated: 0
Conflict Resolution Blocks Allocated: 0
Memory pool cache: Empty

```

Figure 15-37. `onstat -g grp G` Output

The following example shows output for the `onstat -g grp P` command. In the example, the grouper is evaluating rows for the **account**, **teller** and **customer** tables.

```

IBM Informix Dynamic Server Version 11.50.UC1    -- On-Line -- Up 02:11:39 -- 36864 Kbytes
Table 'teller' at 0xb851480 [ CDRShadow VarChars ]
Table 'account' at 0xb7faad8 [ CDRShadow VarChars VarUDTs Floats Blobs]
Table 'customer' at 0xbbe67a8 [ CDRShadow VarChars VarUDTs]
Grouper Table Partitions:
Slot 387...
  'account' 1048707
Slot 389...
  'teller' 1048709
Slot 394...
  'customer' 1048714

```

Figure 15-38. `onstat -g grp P` Output

The following example shows output for the `onstat -g grp pager` command. The sample output shows the grouper large transaction evaluation statistics.

```

IBM Informix Dynamic Server Version 11.50.UC1    -- On-Line -- Up 00:20:42 -- 28672 Kbytes
Grouper Pager statistics:
Number of active big transactions: 0
Total number of big transactions processed: 0
Spool size of the biggest transaction processed: 0 Bytes

```

Figure 15-39. `onstat -g grp pager` Output

The following example shows output for the `onstat -g grp R` command. In this example, the grouper is configured to evaluate rows for replicates with IDs **6553601** and **6553602** (you can use the `onstat -g cat repls` command to obtain the replicate names). The **Ignore** attribute of replicate ID **6553602** shows that the grouper is currently not evaluating rows for this replicate. This can happen if the replicate state is not **ACTIVE**. You can obtain the replicate state using the `onstat -g cat repls` command.

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 00:04:47 -- 28672 Kbytes
Replication Group 6553601 at 0xb0a8360
  Replication at 0xb0a82b0 6553601:6553601 (tab) [ NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
    CDR Shadow: offset 0, size 8
    In Order: offset 8, size 10
Replication Group 6553602 at 0xb0a8480
  Replication at 0xb0a83d0 6553602:6553602 (tab12) [ Ignore Stopped NotifyDS FullRowOn ]
    Column Information [ CDRShadow VarUDTs InOrder Same ]
    CDR Shadow: offset 0, size 8
    In Order: offset 8, size 16

```

Figure 15-40. `onstat -g grp R` Output

The following example shows output for the `onstat -g grp T` command. In this example, the grouper evaluated and queued 1 transaction to the send queue. The **Tx sent to Queuer** field shows the total number of transactions evaluated and queued to the send queue for propagating to all the replicate participants. The **Total rows sent to Queuer** field shows the total number of rows queued to the send queue for propagating to all the replicate participants.

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 00:14:51 -- 28672 Kbytes
Last Tx to Queuer began : (1095116676) 2004/09/13 18:04:36
Last Tx to Queuer ended : (1095116676) 2004/09/13 18:04:36
Last Tx to Queuer log ID, position: 5,3236032
Open Tx: 0
Serial Tx: 0
Tx not sent: 0
Tx sent to Queuer: 1
Tx returned from Queuer: 0
Events sent to Queuer: 0
Events returned from Queuer: 0
Total rows sent to Queuer: 1
Open Tx array size: 1024

```

Figure 15-41. `onstat -g grp T` Output

onstat -g his: Print SQLTRACE information

Syntax:

```

▶▶ onstat -g his ◀◀

```

The `onstat -g his` option prints information about the SQLTRACE configuration parameter. By default, only the DBSA can view `onstat -g his` syssqltrace information. However, when `UNSECURE_ONSTAT = 1` all users can view this information.

Example Output

Statement Statistics:						
Page Read	Buffer Read	Read % Cache	Buffer IDX Read	Page Write	Buffer Write	Write % Cache
1285	19444	93.39	5359	810	17046	95.25
Lock Requests	Lock Waits	LK Wait Time (S)	Log Space	Num Sorts	Disk Sorts	Memory Sorts
10603	0	0.0000	60.4 KB	0	0	0
Total Executions	Total Time (S)	Avg Time (S)	Max Time (S)	Avg IO Wait	I/O Wait Time (S)	Avg Rows Per Sec
1	30.8660	30.8660	30.8660	0.0141	29.2329	169.8959
Estimated Cost	Estimated Rows	Actual Rows	SQL Error	ISAM Error	Isolation Level	SQL Memory
102	1376	5244	0	0	CR	32608

Figure 15-42. onstat -g his Output

Output Description

Page Read

Number of pages that have been read from disk

Buffer Reads

Number of times a page has been read from the buffer pool and not read from disk

Read % Cache

Percentage of times the page should be read from the buffer pool

Buffer IDX Read

Number of buffer reads for index pages

Page Write

Number of pages written to disk

Buffer Write

Number of pages modified and sent back to the buffer pool

Write % Cache

Percentage of time that a page was written to the buffer pool but not to disk

Lock Requests

Total number of locks required by this statement

Lock Waits

Number of times this SQL statement waited on locks

LK Wait Time

Time spent waiting for locks during this SQL statement in seconds

Log Space

Num Sorts

Total number of sorts used to execute the statement

Disk Sorts

Number of sorts for this SQL statement that were executed on disk

Memory Sorts

Total Executions

Total number of times this statement has been executed or the number of times this cursor has been re-used

Total Time

Total time executing this statement in seconds

Avg Time

Average time this state takes to execute in seconds

Max Time

Total time to run the SQL statement in seconds, excluding any time taken by the application

LK Wait Time

Amount of time the statement waited for application locks

Avg IO Wait

Amount of time the statement waited for I/O, excluding any asynchronous I/O.

Avg Rows Per Sec

Average number of rows a second produced by this statement

Estimated Cost

Cost associated with the SQL statement

Estimated Rows

Estimated number of rows returned, as estimated by the optimizer for the statement

Actual Rows

Number of rows returned for this statement

SQL Error

The SQL error number

ISAM Error

The RSAM/ISAM error number

Isolation Level

Isolation level this statement was run with

SQL Memory

Number of bytes this SQL statement requires

onstat -g ioa: Print combined onstat -g information

Syntax:

▶▶ onstat -g ioa ◀◀

The **onstat -g ioa** option prints combined information from **-g ioq**, **-g iov**, and **-g iob**.

Example Output

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:55:13 -- 101376 Kbytes

AIO global info:

7 aio classes

4 open files

64 max global files

32768 max files from setrlimit

AIO I/O queues:

q name/id	len	maxlen	totalops	dskread	dskwrite	dskcopy
fifo 0	0	0	0	0	0	0
adt 0	0	0	0	0	0	0
msc 0	0	0	0	0	0	0
aio 0	0	0	0	0	0	0
pio 0	0	0	0	0	0	0
lio 0	0	0	0	0	0	0
gfd 3	0	1	607	0	607	0

AIO I/O vps:

class/vp	s	io/s	totalops	dskread	dskwrite	dskcopy	wakeups	io/wup	polltries
pollfound kaio_pend									
fifo 0	i	0.0	0	0	0	0	0	0.0	0
0		0							
fifo 1	i	0.0	0	0	0	0	0	0.0	0
0		0							
msc 0	i	0.0	0	0	0	0	0	0.0	0
0		0							
aio 0	i	0.3	607	0	607	0	607	1.0	0
0		0							

AIO global files:

gfd	pathname	totalops	dskread	dskwrite	io/s
3	rootdbs.1	607	0	607	0.3

AIO big buffer usage summary:

class	reads						writes		
	pages	ops	pgs/op	holes	hl-ops	hls/op	pages	ops	pgs/op
fifo	0	0	0.00	0	0	0.00	0	0	0.00
kio	0	0	0.00	0	0	0.00	0	0	0.00
adt	0	0	0.00	0	0	0.00	0	0	0.00
msc	0	0	0.00	0	0	0.00	0	0	0.00
aio	0	0	0.00	0	0	0.00	607	607	1.00
pio	0	0	0.00	0	0	0.00	0	0	0.00
lio	0	0	0.00	0	0	0.00	0	0	0.00

Figure 15-43. onstat -g ioa Output

Output Description

For a description of each output column, see the **-g ioq**, **-g iov**, and **-g iob** options.

onstat -g iob: Print big buffer use summary

Syntax:

▶▶ onstat -g iob ◀◀

The **onstat -g iob** option prints a summary of big buffer use.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:55:13 -- 101376 Kbytes
AIO big buffer usage summary:
class      reads
           pages  ops  pgs/op  holes  hl-ops  hls/op      writes
           pages  ops  pgs/op  holes  hl-ops  hls/op  pages  ops  pgs/op
fifo       0      0   0.00    0      0      0.00     0      0   0.00
kio        0      0   0.00    0      0      0.00     0      0   0.00
adt        0      0   0.00    0      0      0.00     0      0   0.00
msc        0      0   0.00    0      0      0.00     0      0   0.00
aio        0      0   0.00    0      0      0.00    607    607  1.00
pio        0      0   0.00    0      0      0.00     0      0   0.00
lio        0      0   0.00    0      0      0.00     0      0   0.00

```

Figure 15-44. `onstat -g iob` Output

onstat -g iof: Print asynchronous I/O statistics

Syntax:

▶▶ `onstat -g iof` ◀◀

The `onstat -g iof` option prints asynchronous I/O statistics by chunk or file. This option is similar to the `-D` option, except that information on nonchunk files is also displayed. It includes information about temporary files and sort-work files.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:55:32 -- 101376 Kbytes
AIO global files:
gfd pathname      totalops  dskread  dskwrite  io/s
3 rootdbs.1      613      0        613      0.3

```

Figure 15-45. `onstat -g iof` Output

Output Description

gfd Global file descriptor number for this chunk

pathname
The pathname of the chunk

totalops
Total number of read and write operations that have occurred against the chunk

dskread
Number of disk read that have occurred against the chunk

dskwrite
Number of disk writes that have occurred against the chunk

io/s Number of I/Os per second

onstat -g iog: Print AIO global information

Syntax:

```
▶▶ onstat -g iog ◀◀
```

The **onstat -g iog** option prints AIO global information.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:55:42 -- 101376 Kbytes
AIO global info:
  8 aio classes
  5 open files
 64 max global files
```

Figure 15-46. *onstat -g iog* Output

onstat -g ioq: Print I/O queue information

Syntax:

```
▶▶ onstat -g ioq [queue_name] ◀◀
```

The **onstat -g ioq** option shows statistics about the number and types of operations performed by I/O queues. If a *queue_name* is given then only queues with that name are shown. If no *queue_name* is given then information is given for all queues.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 01:00:54 -- 109568 Kbytes

AIO I/O queues:
q name/id      len maxlen totalops  dskread dskwrite  dskcopy
sqli_dbg      0      0      0          0         0         0
fifo          0      0      0          0         0         0
adt           0      0      0          0         0         0
msc           0      0      1          537        0         0
aio           0      0      3          6537       238       5777
pio           0      0      2          1103        0       1102
lio           0      0      2          11795       0       11794
gfd           3      0     17          17489      1526     15963
gfd           4      0     17          18347      2384     15963
gfd           5      0     16           220         41        179
gfd           6      0      4           4           0          4
gfd           7      0      4           4           0          4
gfd           8      0      4           4           0          4
gfd           9      0      9           54          24         30
gfd          10      0     16           149         40        109
gfd          11      0     16           621        128        493
gfd          12      0     16          1953       1146        807
gfd          13      0     16           409         71         338
gfd          14      0     16           378         60         318

```

Figure 15-47. onstat -g ioq Output

Output Description

q name/id

The name and number of the I/O queue. The name indicates what type of queue it is. The number is used to tell queues of the same name apart.

Here is a list of the possible queue names and what each type of queue handles:

sqli_dbg

Handles I/O for IBM Technical Support's SQL Interface Debugging feature

fifo Handles I/O for FIFO VPs

adt Handles auditing I/O

msc Handles miscellaneous I/O

aio Handles IBM Informix asynchronous I/O

kio Handles kernel AIO

pio Handles physical logging I/O

lio Handles logical logging I/O

gfd Global File Descriptor - Each primary and mirror chunk is given a separate global file descriptor. Individual gfd queues are used depending on whether kaio is on and the associated chunk is cooked or raw.

len The number of pending I/O requests in the queue

maxlen

The largest number of I/O requests that have been in the queue at the same time

totalops

The total number of I/O operations that have been completed for the queue

dskread

Total number of completed read operations for the queue

dskwrite

Total number of completed write operations for the queue

dskcopy

Total number of completed copy operations for the queue

onstat -g ipl: Print index page logging status information

Syntax:

▶▶ onstat -g ipl ◀◀

The **onstat -g ipl** option shows index page logging status information.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:20:55 -- 46080 Kbytes
Index page logging status: Enabled
Index page logging was enabled at: 2006/12/20 16:01:02

```

Figure 15-48. onstat -g ipl Output

Output Description**Index page logging status**

Status of index page logging: Enabled or Disabled.

Index page logging was enabled at

The date and time at which index page logging was enabled.

onstat -g iov: Print AIO VP statistics

Syntax:

▶▶ onstat -g iov ◀◀

The **onstat -g iov** option shows asynchronous I/O statistics for each virtual processor.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:56:26 -- 101376 Kbytes
AIO I/O vps:
class/vp s io/s totalops dskread dskwrite dskcopy wakeups io/wup errors
fifo 0 i 0.0 0 0 0 0 0 0.0 0
fifo 1 i 0.0 0 0 0 0 0 0.0 0
msc 0 i 0.0 0 0 0 0 0 0.0 0
aio 0 s 0.3 628 0 628 0 628 1.0 0

```

Figure 15-49. `onstat -g iov` Output

Output Description

class The class of the virtual processor.

vp The ID number of the virtual processor within its class.

s Current status of the AIO virtual processor

f Fork

i Idle

s Search

b Busy

o Open

c Close

io/s The average I/O speed (measured in operations per second) for the virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

totalops

Total number of I/O operations performed by this virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

dskread

Total number of read operations performed by this virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

dskwrite

Total number of write operations performed by this virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

dskcopy

Total number of copy operations performed by this virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

wakeups

For AIO VPs, the number of times the virtual processor has gone idle since the time the database server started or since `onstat -z` was last run, whichever happened last.

io/wup

For AIO VPs, the average number of I/O operations performed per wake-up by this virtual processor since the time the database server started or since `onstat -z` was last run, whichever happened last.

errors Total number of KAIO out of resource errors.

onstat -g lap: Print light appends status information

Syntax:

```
▶▶ onstat -g lap ◀◀
```

The **onstat -g lap** option prints information on the status of light appends occurring in the system.

Example Output

```
$>onstat -g lap
IBM Informix Dynamic Server Version 11.50.UC2 -- On-Line -- Up 00:01:24 -- 26624 Kbytes

Light Append Info
session id address cur_ppage la_npused la_ndata la_nrows bufcnt
31 b60a5e8 ffbff494 2938 2937 93990 4
```

Figure 15-50. *onstat -g lap* Output

Output Description

Session id (decimal)

Session ID performing the light append operation

address (hexadecimal)

Address of the light append buffer

cur_ppage (hexadecimal)

Current physical page address

la_npused (decimal)

Number of pages allocated

la_ndata (decimal)

Number of data pages appended

la_nrows (decimal)

Number of rows appended

bufcnt (decimal)

Number of light append buffers

onstat -g lmx: Print all locked mutexes

Syntax:

```
▶▶ onstat -g lmx ◀◀
```

The **onstat -g lmx** option prints all locked mutexes.

Example Output

```

Locked mutexes:
mid      addr          name          holder  lkcnt  waiter  waittime
Number of mutexes on VP free lists: 49

```

Figure 15-51. `onstat -g lmx` Output

Output Description

mid Internal mutex identifier
addr Address of locked mutex
name Name of the mutex
holder Session ID of the thread holding the mutex
lkcnt Number of waiters for this mutex
waiter List of addresses waiting for this mutex
waittime
Amount of time this thread has been waiting

`onstat -g lsc`: Print active light scan status

Syntax:

```
▶▶ onstat -g lsc ◀◀
```

The `onstat -g lsc` option displays the status of any currently active light scans.

Example Output

```

IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:08:42 -- 1067288 Kbytes

Light Scan Info
descriptor  address          next_lpage  next_ppage      ppage_left  bufcnt  look_aside
3           474b74b0        4a0        7e2c80         416         1       N

```

Figure 15-52. `onstat -g lsc` Output

Output Description

descriptor (decimal)
Light scan ID

address (hex)
Memory address of the light scan descriptor

next_lpage (hex)
Next logical page address to scan

next_ppage (hex)
Next physical page address to scan

ppage_left (decimal)
Number of physical pages left to scan in the current extent

#bufcnt (decimal)
Number of light scan buffers used for this light scan

#look_aside (char)

Whether look aside is needed for this light scan (Y = yes, N = no). Look asides occur when a thread needs to examine the buffer pool for existing pages to obtain the latest image of a page being light scanned.

onstat -g mem: Print pool memory statistics

Syntax:

```
▶▶ onstat -g mem [pool name—session id]
```

The **onstat -g mem** option prints memory statistics for a pool. Session pools are named with the session number. If no argument is provided, information about all pools is displayed.

Example Output

```
Pool Summary:
name      class addr      totalsize freesize #allocfrag #freefrag
resident  R    10a001028  2420736  7960     2           2
res-buff  R    10a250028  8269824  7960     2           2
global    V    10aac0028  9351168  32648   650        11
...
...
...
onmode_mon V    10b983028  20480    2752    108         1
13         V    10bd5d028  16384    5200     12         2
Blkpool Summary:
name      class addr      size      #blks      pre-hint  szavail|
global    V    10aac8920   0         0          0         0
xmf_msc_pl V    10ac84ca0  954368    73        0         0
```

Figure 15-53. onstat -g mem Output

Output Description

Pool Summary

name Pool name

class Shared memory segment type where pool is created

addr Pool memory address

totalsize
Pool size, in bytes

freesize
Free memory in pool

#allocfrag
Allocated fragments in pool

#freefrag
Free fragments in pool

Blkpool Summary

name Pool name

class Shared memory segment type where pool is created
addr Pool memory address
size Pool size, in bytes
#blks Number of blocks in pool

onstat -g mgm: Print MGM resource information

Syntax:

▶▶ onstat -g mgm ◀◀

The **onstat -g mgm** option prints Memory Grant Manager (MGM) resource information. You can use the **onstat -g mgm** option to monitor how MGM coordinates memory use and scan threads. This **onstat** option reads shared-memory structures and provides statistics that are accurate at the instant that the command executes.

The **onstat -g mgm** output displays a unit of memory called a *quantum*. The *memory quantum* represents a unit of memory, as follows:

memory quantum = DS_TOTAL_MEMORY / DS_MAX_QUERIES

The following calculation shows the memory quantum for the values that Figure 15-54 on page 15-75 displays:

memory quantum = 4000 kilobytes / 31
= 129 kilobytes

The *scan thread quantum* is always equal to 1.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:00:51 -- 21504 Kbytes

Memory Grant Manager (MGM)
-----
MAX_PDQPRIORITY: 100
DS_MAX_QUERIES: 31
DS_MAX_SCANS: 1048576
DS_NONPDQ_QUERY_MEM: 128 KB
DS_TOTAL_MEMORY: 4000 KB

Queries:  Active      Ready  Maximum
          0          0      31

Memory:   Total      Free  Quantum
(KB)     4000     4000   128

Scans:    Total      Free  Quantum
          1048576  1048576  1

Load Control:  (Memory)      (Scans) (Priority) (Max Queries) (Reinit)
              Gate 1      Gate 2   Gate 3     Gate 4     Gate 5
(Queue Length) 0          0       0         0         0

Active Queries: None
Ready Queries:  None
Free Resource   Average #      Minimum #
-----
Memory         0.0 +- 0.0     500
Scans          0.0 +- 0.0    1048576

Queries        Average #      Maximum #      Total #
-----
Active         0.0 +- 0.0     0              0
Ready         0.0 +- 0.0     0              0

Resource/Lock Cycle Prevention count: 0

```

Figure 15-54. `onstat -g mgm` Output

Output Description

The first portion of the output shows the values of the PDQ configuration parameters.

The second portion of the output describes MGM internal control information. It includes four groups of information. The first group is **Queries**:

Active Number of PDQ queries that are currently executing

Ready Number of user queries ready to run but whose execution the database server deferred for load-control reason

Maximum

Maximum number of queries that the database server permits to be active. Reflects current value of the `DS_MAX_QUERIES` configuration parameter

The next group is **Memory**:

Total Kilobytes of memory available for use by PDQ queries (`DS_TOTAL_MEMORY` specifies this value.)

Free Kilobytes of memory for PDQ queries not currently in use

Quantum

Kilobytes of memory in a memory quantum

The next group is **Scans**:

- Total* The total number of scan threads as specified by the DS_MAX_SCANS configuration parameter
- Free* Number of scan threads currently available for decision-support queries
- Quantum*
The number of scan threads in a scan-thread quantum

The last group in this portion of the output describes **MGM Load Control**:

- Memory*
Number of queries that are waiting for memory
- Scans* Number of queries that are waiting for scans
- Priority*
Number of queries that are waiting for queries with higher PDQ priority to run
- Max Queries*
Number of queries that are waiting for a query slot
- Reinit* Number of queries that are waiting for running queries to complete after an **onmode -M** or **-Q** command

The next portion of the output, **Active Queries**, describes the MGM active and ready queues. This portion of the output shows the number of queries waiting at each gate:

- Session* The session ID for the session that initiated the query
- Query* Address of the internal control block associated with the query
- Priority*
PDQ priority assigned to the query
- Thread* Thread that registered the query with MGM
- Memory*
Memory currently granted to the query or memory reserved for the query (Unit is MGM pages, which is 8 kilobytes.)
- Scans* Number of scan threads currently used by the query or number of scan threads allocated to the query
- Gate* Gate number at which query is waiting

The next portion of the output, **Free Resource**, provides statistics for MGM free resources. The numbers in this portion and in the final portion reflect statistics since system initialization or the last **onmode -Q**, **-M**, or **-S** command. This portion of the output contains the following information:

- Average*
Average amount of memory and number of scans
- Minimum*
Minimum available memory and number of scans

The next portion of the output, **Queries**, provides statistics concerning MGM queries:

- Average*
Average active and ready queue length

Maximum

Maximum active and ready queue length

Total

Total active and ready queue length

Resource/Lock Cycle Prevention count

Number of times the system immediately activated a query to avoid a potential deadlock. (The database server can detect when some of the queries in its queue might create a deadlock situation if the queries are not run immediately.)

onstat -g nbm: Print a block bit map

Syntax:

▶▶ onstat -g nbm ◀◀

The **onstat -g nbm** option shows the block bit map for the nonresident segments. Each bit of the bitmap represents a 4 KB block. If the block is used then the bit is set to 1. If it is free the bit is set to 0. The bitmap is shown as a series of hexadecimal numbers. The bits, and therefore the blocks, are numbered starting at 0 so the first block is block 0, the second is block 1, and so on.

Example Output

This example shows the bitmap for the segment of virtual memory at 0x10CC00000. The bitmap itself is at 0x10CC00290. All 1792 blocks of the segment are free except for block 0 and block 1023.

```

Block bitmap for virtual segment address 0x10cc00000:
address = 0x10cc00290, size(bits) = 1792
used = 1, largest_free = -1
 0:8000000000000000 0000000000000000 0000000000000000 0000000000000000
256:0000000000000000 0000000000000000 0000000000000000 0000000000000000
512:0000000000000000 0000000000000000 0000000000000000 0000000000000000
768:0000000000000000 0000000000000000 0000000000000000 0000000000000001
1024:0000000000000000 0000000000000000 0000000000000000 0000000000000000
1280:0000000000000000 0000000000000000 0000000000000000 0000000000000000
1536:0000000000000000 0000000000000000 0000000000000000 0000000000000000

```

Figure 15-55. onstat -g nbm Output

Output Description

address

The starting address of the bitmap.

size

The number of bits in the bitmap. This is also the number of 4 KB blocks in the memory segment.

used

The total number of bits in the bitmap that are set to 1. This is also the number of 4 KB blocks that are in use in the memory segment.

largest free

If this is a value other than -1 it is the largest number of consecutive bits that are free, which is also the number of 4 KB blocks in the largest contiguous set of blocks in the memory segment.

A value of -1 means that the largest free space has not been calculated. The database server only calculates the largest free space if it tries to allocate a set of blocks starting at the *lastalloc* block but there is not enough free space. The value is set to -1 again as soon as another block is allocated in the segment.

onstat -g nif: Print ER network interface statistics

Syntax:

```
▶▶ onstat -g nif modifier ▶▶
```

The **onstat -g nif** command prints statistics about the network interface for Enterprise Replication. The output shows which sites are connected and provides a summary of the number of bytes sent and received by each site. This can help you determine that a site is in a hung state, if it is not sending or receiving bytes.

The **-g nif** option is used primarily as a debugging tool and by Technical Support.

The **onstat -g nif** command has the following formats:

```
onstat -g nif
onstat -g nif modifier
```

The following table describes the values for *modifier*.

Modifier	Action
all	Prints the sum and the sites
sites	Prints the NIF site context blocks
serverid	Prints information about the replication server whose groupID is serverID
sum	Prints the sum of the number of buffers sent and received for each site

Example Output

The following example shows output for the **onstat -g nif** command. In this example, the local server is connected to the server group **g_bombay** and its CDR ID is **200**. The connection status is set to running. The server group **g_bombay** NIF version is **7**. The local server has sent three messages to the server **g_bombay** and it has received two messages from **g_bombay**.

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:02:34 -- 28672 Kbytes
NIF anchor Block: af01610
      nifGState      RUN
      RetryTimeout   300

CDR connections:
  Id   Name      State   Version   Sent   Received
-----
  200  g_bombay    RUN     7         3     2
```

Figure 15-56. onstat -g nif Output

onstat -g nsc *client_id*: Print current shared memory connection information

Syntax:

```
▶▶ onstat -g nsc [client_id] ▶▶
```

If no *client_id* is provided, information about all current shared memory connections to the database server is given. If a *client_id* is provided then this command gives more detailed information about the shared memory connection with that ID.

Example Output

This is output of **onstat -g nsc** with no *client_id*. It shows that there is only one user currently connecting to the database server through shared memory. That connection has an ID of 0.

```
IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 6 days

 clientid  clientPID    state #serverbufs #clientbufs  #rdwrts
         0         6031 Connected         4         4         12
```

Figure 15-57. onstat -g nsc Output

This example shows output from running the command using a *client_id* of 0.

```
Network Shared Memory Status for Client: 0

 clientid  clientPID    state #serverbufs #clientbufs  #rdwrts
         0         18949 Connected         4         4         447048

 needbuf   segid      semid      semnum    be_semid    be_semnum
         0         1303      851969         0         851969         10

 be_curread  be_curwrite  fe_curread  fe_curwrite
         -1          1           0           2

 be_nextread be_nextwrite fe_nextread fe_nextwrite
         2          2           4           3

 readyqueue
 -1  -1  -1  -1  -1  -1  -1  -1  -1  -1  -1

 Server Buffers
 i: bufid  status  offset  fe_addr
 0:  4     inuse  4474   804474
 1:  5     inuse  4888   804888
 2:  6     avail  4c9c   804c9c
 3:  7     avail  50b0   8050b0
 4: -1     free   0       0
 5: -1     free   0       0

 Client Buffers
 bufid  status  offset  fe_addr
 0:  0     avail  3424   803424
 1:  1     avail  3838   803838
 2:  2     inuse  3c4c   803c4c
 3:  3     avail  4060   804060
 4: -1     free   0       0
 5: -1     free   0       0
```

Figure 15-58. onstat -g nsc with client id Output

Output Description

clientid

Server assigned ID

clientPID

Client process ID

state State of connection

Connected

The client has established a connection with the server.

Con1 The server has successfully set up a connection with the client, but the client has not yet been notified of it.

Waiting

The server is in the process of setting up a connection with the client.

Reject Client connection has been rejected by the server, normally because the server is shutting down or not yet in on-line mode.

Closed

Server has closed the connection with the client. Client might not be aware of the fact yet.

Not connected

Server is initializing internal structures for the connection.

Unknown

Connection has been closed and the client is aware of the fact. Server is cleaning up internal structures.

#serverbufs

Database server buffers currently allocated

#clientbufs

Client buffers currently allocated

#rdwrts

The total number of reads and writes performed through this connection since it was created.

The following items are only in the output if you run **onstat -g nsc** with a *client_id*:

needbuf

Indicates if server is waiting for a buffer to be freed

0 False

1 True

segid Shared memory segment ID

semid Semaphore ID

semnum

Semaphore number in the semaphore ID

be_semid

Backend semaphore ID

be_semnum

Backend semaphore number in the semaphore ID

be_curread
ID of backend buffer being read

be_curwrite
ID of backend buffer being written

fe_curread
ID of frontend buffer being read

fe_currwrite
ID of frontend buffer being written

be_nextread
ID of next backend buffer to be read

be_nextwrite
ID of next backend buffer to be written

fe_nextread
ID of next frontend buffer to be read

fe_nextwrite
ID of next frontend buffer to be written

readyqueue
Queue of the shared memory buffer ids

Buffers

i Internal location key of message buffer

bufid Message buffer ID

status Status of message buffer

offset Offset of memory buffer in shared memory segments

fe_addr
Frontend address of message buffer

onstat -g nsd: Print poll threads shared-memory data

Syntax:

▶▶ onstat -g nsd ◀◀

The **onstat -g nsd** option prints shared-memory data for poll threads.

Example Output

```

Network Shared Memory Data for Poll Thread: 0
Free Message Buffer Bitmap
(bitmap address = 10b9eef80, bitmap size 480)
000000010b9eef80:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
000000010b9eefa0:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
Free Message Buffer Status Bitmap
(bitmap address = 10ca0a9b0, bitmap size 50)
000000010ca0a9b0:ffffffff ffffff
Message Buffer Table
bufid  clientid      addr
Message Buffer Status Table
clientid      netscb addr          addr          offset

```

Figure 15-59. `onstat -g nsd` Output

onstat -g nss: Print shared memory network connections status

Syntax:

```

▶▶ onstat -g nss [sessionid] ▶▶

```

The `onstat -g nss sessionid` option displays the status of shared memory network connections. If no `sessionid` is provided, a one-line summary for each shared memory connection is listed.

Example Output

clientid	clientPID	state	#serverbufs	#clientbufs	#rdwrts
1	14018	Connected	4	4	331
0	12398	Connected	4	4	294
2	14036	Connected	4	4	59

Figure 15-60. `onstat -g nss` Output

Output Description

clientid (decimal)

Server assigned value for lookups

clientPID (decimal)

Client process ID

state (string)

Current state of the connection.

- Connected
- Con1
- Waiting
- Reject
- Badvers
- Closed
- Not connected

- Unknown

#serverbufs (dec)

Number of database server buffers currently allocated

#clientbufs (dec)

Number of client buffers currently allocated

#rdwrts (dec)

Total number of buffers in use

onstat -g ntd: Print network statistics

Syntax:

▶▶ onstat -g ntd ◀◀

The **onstat -g ntd** option prints network statistics by service.

Example Output

```

IBM Informix Dynamic Server Version 11.50.TC1  -- On-Line -- Up 04:27:28 -- 78208 Kbytes
global network information:
#netscb connects      read      write q-limits  q-exceed alloc/max
  4/   5      11         0    3546 3549/ 10   10/   0    0/   0

Client Type   Calls   Accepted   Rejected      Read      Write
sqlexec      yes      11         0           3531     3540
srvinfx      yes       0         0            0         0
ospace       yes       0         0            4         9
onlog        yes       0         0            0         0
onparam      yes       0         0            0         0
oncheck      yes       0         0            0         0
onload       yes       0         0            0         0
onunload     yes       0         0            0         0
onmonitor    yes       0         0            0         0
dr_accept    yes       0         0            0         0
cdraccept    no        0         0            0         0
ontape       yes       0         0            0         0
srvstat      yes       0         0            0         0
asfecho      yes       0         0            0         0
listener     yes       0         0           11         0
crsamexec    yes       0         0            0         0
onutil       yes       0         0            0         0
drdaexec     yes       0         0            0         0
smx          yes       0         0            0         0
safe         yes       0         0            0         0
Totals              11         0           3546     3549

```

Figure 15-61. onstat -g ntd Output

onstat -g ntm: Print network mail statistics

Syntax:

▶▶ onstat -g ntm ◀◀

The **onstat -g ntm** option prints network mail statistics.

Example Output

```
IBM Informix Dynamic Server Version 11.50.TC1-- On-Line -- Up 04:28:13 -- 78208 Kbytes

global network information:
#netscb connects    read    write  q-limits  q-exceed alloc/max
 4/ 5      11      0    3546 3549/ 10  10/ 0    0/ 0

Network mailbox information:
box  netscb thread name    max received  in box  max in box full signal
 5  f07e8b0 soctcppoll    10      24      0        1      0    yes
 6  f0b6ad8 soctcplst     10       0      0         0      0    no
 7  f0e8b18 soctcplst     10       0      0         0      0    no
```

Figure 15-62. **onstat -g ntm** Output

onstat -g ntt: Print network user times

Syntax:

►► onstat -g ntt ◀◀

The **onstat -g ntt** option prints network user times.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:57:44 -- 101376 Kbytes

global network information:
#netscb connects    read    write  q-limits  q-exceed alloc/max
 3/ 3      0      0      0    135/ 10  0/ 0    2/ 0

Individual thread network information (times):
netscb thread name sid  open    read    write    address
c76ea28 ontape      61  14:34:48 14:34:50 14:34:50
c63e548 tlitcplst     4   14:30:43 14:34:48  server.ibm.com|5006|tlitcp
c631028 tlitcppoll     3   14:32:32
```

Figure 15-63. **onstat -g ntt** output

onstat -g ntu: Print network user statistics

Syntax:

►► onstat -g ntu ◀◀

The **onstat -g ntu** option prints network user statistics.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:57:53 -- 101376 Kbytes
global network information:
#netscb connects      read      write q-limits  q-exceed alloc/max
 4/ 5      11      0      3546 3549/ 10  10/ 0  0/ 0

Individual thread network information (basic):
netscb type  thread name      sid  fd poll  reads  writes q-nrm q-exp
f0e8b18 soctcp soctcplst      5 1736 5      0      0   10   1
f0b6ad8 soctcp soctcplst      4 1764 5     11      0   10   1
f095d60 soctcp soctcpio      3  0  0      0      0   10   1
f07e8b0 soctcp soctcppoll     2  0  5      0      0   10   1

```

Figure 15-64. `onstat -g ntu` Output

onstat -g opn: Print open partitions

Syntax:

```

▶▶ onstat -g opn [thread_id]

```

The `onstat -g opn` option prints a list of the partitions (tables/indexes), by thread ID, that are currently open in the system. The `onstat -g opn thread_id` option restricts the list to the specified ID.

Example Output

```

IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 17:22:49 -- 1067276 Kbytes
tid  rstcb          isfd  op_mode  op_flags  partnum  ucount  ocount  lockmode
38  0x00000000460db7b0 0    0x00000400 0x00000397 0x001000af 2      2      0
38  0x00000000460db7b0 1    0x00000002 0x00000117 0x001000af 2      2      0
38  0x00000000460db7b0 2    0x00000440 0x00000797 0x0010010c 2      0      0
38  0x00000000460db7b0 3    0x00000400 0x00000407 0x0010010a 2      0      0
38  0x00000000460db7b0 4    0x00000400 0x00000407 0x0010010a 2      0      0
38  0x00000000460db7b0 5    0x00000002 0x00000003 0x00100003 2      2      0
38  0x00000000460db7b0 6    0x00000400 0x00000397 0x00100003 2      2      0
38  0x00000000460db7b0 7    0x00000400 0x00000413 0x0010010f 2      0      0
38  0x00000000460db7b0 8    0x00000440 0x00000797 0x0010010c 2      0      0
38  0x00000000460db7b0 9    0x00000402 0x00000403 0x0010010f 2      0      0
38  0x00000000460db7b0 10   0x00000442 0x00000403 0x00100111 1      0      0
38  0x00000000460db7b0 11   0x00000442 0x00000403 0x00100110 1      0      0
38  0x00000000460db7b0 12   0x00000442 0x00000403 0x00100112 1      0      0
38  0x00000000460db7b0 15   0x00000400 0x00000407 0x00000006 1      0      0
38  0x00000000460db7b0 16   0x00000400 0x00000413 0x00100119 1      0      0

36  0x00000000460dbf98 0    0x00000400 0x00000397 0x001000af 2      2      0
36  0x00000000460dbf98 1    0x00000002 0x00000003 0x001000af 2      2      0
36  0x00000000460dbf98 3    0x00000402 0x00000407 0x0010010a 2      0      0
36  0x00000000460dbf98 4    0x00000400 0x00000413 0x0010010a 2      0      0
36  0x00000000460dbf98 6    0x00000442 0x00000797 0x0010010c 1      0      0

37  0x00000000460dc780 0    0x00000400 0x00000397 0x001000af 2      2      0
37  0x00000000460dc780 1    0x00000002 0x00000117 0x001000af 2      2      0
37  0x00000000460dc780 2    0x00000400 0x00000407 0x0010010a 2      0      0
37  0x00000000460dc780 3    0x00000440 0x00000797 0x0010010c 2      0      0
37  0x00000000460dc780 4    0x00000400 0x00000413 0x0010010f 2      0      0
37  0x00000000460dc780 5    0x00000400 0x00000407 0x0010010a 2      0      0
37  0x00000000460dc780 6    0x00000440 0x00000797 0x0010010c 2      0      0
37  0x00000000460dc780 7    0x00000400 0x00000397 0x00100003 2      2      0
37  0x00000000460dc780 8    0x00000002 0x00000003 0x00100003 2      2      0
37  0x00000000460dc780 9    0x00000442 0x00000403 0x00100111 1      0      0
37  0x00000000460dc780 10   0x00000442 0x00000403 0x00100110 1      0      0
37  0x00000000460dc780 11   0x00000402 0x00000403 0x0010010f 2      0      0
37  0x00000000460dc780 12   0x00000400 0x00000413 0x00100119 1      0      0
37  0x00000000460dc780 13   0x00000442 0x00000403 0x00100112 1      0      0
37  0x00000000460dc780 14   0x00000400 0x00000407 0x00000006 1      0      0

```

Figure 15-65. *onstat -g opn* Output

Output Description

tid (decimal)

Thread ID currently accessing the partition resource (table/index)

rstcb (hex)

In-memory address of the RSAM thread control block for this thread

isfd (decimal)

ISAM file descriptor associated with the open partition

op_mode (hex)

Current status of the partition lock mode using a combination of the following hexadecimal values:

- 0x000000 Open for input only
- 0x000001 Open for output only
- 0x000002 Open for input and output
- 0x000004 System catalog
- 0x000008 No logical logging
- 0x000010 Open if not already opened for alter
- 0x000020 Open all fragments data and index
- 0x000040 Do not allocate a blob descriptor
- 0x000080 Open for alter

- 0x000100 Open all data fragments
- 0x000200 Automatic record lock
- 0x000400 Manual record lock
- 0x000800 Exclusive ISAM file lock
- 0x001000 Ignore dataskip - data cannot be ignored
- 0x002000 Dropping partition - delay file open
- 0x004000 Do not drop blob space blobs when table dropped
(alter fragment)
- 0x010000 Open table for DDL operations
- 0x040000 Do not assert fail if this partnum does not exist
- 0x080000 Include fragments of subtables
- 0x100000 Table created under supertable
- 0x400000 Blob in use by CDR

op_flags (hex)

Current status of the partition using a combination of the following hexadecimal values:

- 0x0001 Open data structure is in use
- 0x0002 Current position exists
- 0x0004 Current record has been read
- 0x0008 Duplicate created or read
- 0x0010 Skip current record on reverse read
- 0x0020 Shared blob information
- 0x0040 Partition opened for rollback
- 0x0080 Stop key has been set
- 0x0100 No index related read aheads
- 0x0200 isstart called for current stop key
- 0x0400 Pseudo-closed
- 0x0800 Real partition opened for SMI query
- 0x1000 Read ahead of parent node is done
- 0x2000 UDR keys loaded
- 0x4000 Open is for a pseudo table
- 0x8000 End of file encountered when positioning in table

partnum (hex)

Partition number for the open resource (table/index)

ucount (decimal)

Number of user threads currently accessing this partition

ocount (decimal)

Number of times this partition was opened

lockmode (decimal)

Type of lock being held using one of the following coded values:

- 0 No locks
- 1 Byte lock
- 2 Intent shared lock
- 3 Shared lock
- 4 Shared lock by repeatable read (only on items)
- 5 Update lock
- 6 Update lock by repeatable read (only on items)
- 7 Intent exclusive lock
- 8 Shared, intent exclusive lock
- 9 Exclusive lock
- 10 Exclusive lock by repeatable read (only on items)
- 11 Inserter's repeatable read test lock

onstat -g pos: Print file values

Syntax:

► onstat -g pos ◀

The **onstat -g pos** option prints the values for the **\$INFORMIXDIR/etc/infos.DBSERVERNAME** file.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:58:04 -- 101376 Kbytes
 1  7  0  infos ver/size 6 520
 2  1  0  snum= 101 shm=52665801 shmb=000000000a000000 cosvr=1 gpid=2599 qa10_1
 3  4  0  onconfig path /work/xps/sqlldist/etc/onconfig.xps
 4  5  0  host qa10-1
 5  6  0  oninit ver IBM Informix Extended Parallel Server Version 8.50.FN145
 6  8  0  infos sqlhosts: /work/xps/sqlldist/etc/sqlhosts
 7 12  0  del
 8 13  0  del
 9  2 4001 shm id=4001 key=0x52665801 (1382438913) addr=0x a000000 size=19918848 R
10  3  1  sema 1
11 11  0  MRI: addr = 0xb4110e8 version = 0x10001
12  2  2  shm id=2 key=0xab00bf7c (-1426014340) addr=0x 200000000 size=16777216 R
13  2  3  shm id=3 key=0x52665802 (1382438914) addr=0x bb00000 size=9437184 V
14  2  5  shm id=5 key=0x52665803 (1382438915) addr=0x c400000 size=8388608 V
15  2  7  shm id=7 key=0x52665804 (1382438916) addr=0x cc00000 size=32505856 V
16  2  8  shm id=8 key=0x52665805 (1382438917) addr=0x eb00000 size=8388608 V
17  2  9  shm id=9 key=0x52665806 (1382438918) addr=0x f300000 size=8388608 V
18  2 10  shm id=10 key=0x52665807 (1382438919) addr=0x fb00000 size=8388608 V
```

Figure 15-66. **onstat -g pos** Output

onstat -g ppf: Print partition profiles

Syntax:

```
▶▶ onstat -g ppf partition_number ▶▶
```

The **onstat -g ppf *partition_number*** option prints the partition profile for the specified partition number. The **onstat -g ppf 0** option prints profiles for all partitions. If the **TBLSPACE_STATS** configuration parameter is set to 0, then the **onstat -g ppf** command displays: Partition profiles disabled.

For more information on the **onstat -g ppf** option, see the *IBM Informix Performance Guide*.

Example Output

Partition profiles												
partnum	lkrqs	lkwts	dlks	touts	isrd	iswrt	isrwt	isdel	bfrd	bfwrt	seqsc	rhitratio
0x100001	0	0	0	0	0	0	0	0	0	0	0	0
0x100002	1506	0	0	0	416	4	0	4	1282	20	0	97
0x100003	15	0	0	0	5	0	0	0	20	0	0	75
0x1000a5	0	0	0	0	0	0	0	0	12	0	0	67
0x1000e3	4	0	0	0	1	0	0	0	4	0	0	25
0x200001	0	0	0	0	0	0	0	0	0	0	0	0
0x300001	0	0	0	0	0	0	0	0	0	0	0	0
0x400001	0	0	0	0	0	0	0	0	0	0	0	0

Figure 15-67. `onstat -g ppf` Output

Output Description

partnum

Partition number

lkrqs Lock requests

lkwts Lock waits

dlks Deadlocks

touts Remote deadlock timeout

isrd Number of reads

iswrt Number of rewrites

isdel Deletes

bfrd Number of buffer reads in pages

bfwrt Number of buffer writes in pages

seqsc Sequential scans

rhitratio

Ratio of disk read to buffer read

onstat -g prc: Print sessions using UDR or SPL routine

Syntax:

▶▶ `onstat -g prc` ◀◀

The `onstat -g prc` option prints the number of sessions currently using the UDR or SPL routine.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:59:00 -- 101376 Kbytes
Stored Procedure Cache:
  Number of lists      : 31
  PC_POOLSIZe        : 50
  Number of entries    : 0
  Number of inuse entries : 0
Stored Procedure Cache Entries:
list# id ref_cnt  dropped?  heap_ptr  procedure name
-----
Stored Procedure Cache is empty.

```

Figure 15-68. `onstat -g prc` Output

onstat -g proxy: Print Proxy Distributor information

Syntax:

```

▶▶ onstat -g proxy [all] [proxy_id] [proxy_transaction_id] [sequence_number]

```

The `onstat -g proxy` command prints information about proxy distributors. The output of the `onstat -g proxy` command differs slightly depending on whether the command is run on a primary server or on a secondary server.

Invocation	Explanation
<code>onstat -g proxy</code>	Displays proxy distributor information
<code>onstat -g proxy all</code>	When run on the primary server, displays information about proxy distributors and proxy agent threads. When run on the secondary server, displays information about all sessions currently performing updates to secondary servers.
<code>onstat -g proxy proxy_id proxy_transaction_id sequence_number</code>	This option is valid only on secondary servers. Displays detailed information about the current work being performed by a given proxy distributor. The <code>proxy_transaction_id</code> and <code>sequence_number</code> are optional parameters. When supplied, the first number is considered the <code>proxy_transaction_id</code> , and the second is interpreted as the <code>sequence_number</code> . If the supplied <code>proxy_transaction_id</code> or <code>sequence_number</code> do not exist, the command output is the same as the output for <code>onstat -g proxy</code> .

Example Output

```

onstat -g proxy

IBM Informix Dynamic Server Version 11.50.U      -- On-Line -- Up 00:07:55 -- 46392 Kbytes
Secondary      Proxy      Transaction Hot Row
Node           ID           Count       Total
serv2          392          2           112
serv2          393          2           150

```

Figure 15-69. onstat -g proxy Output (run from primary server)

Output Description

Secondary Node

Name of the secondary server as it is known by the primary server.

Proxy ID

ID of the proxy distributor. Proxy IDs are unique within a high-availability cluster.

Transaction Count

The number of transactions currently being processed by the proxy distributor.

Hot Row Total

Total number of hot rows ever handled by the proxy distributor.

Example Output

```

$ onstat -g proxy

IBM Informix Dynamic Server Version 11.50.U      -- Updatable (SDS) -- Up 00:38:30 -- 62776 Kbytes
Primary      Proxy      Transaction Hot Row
Node         ID           Count       Total
serv1        392          2           112
serv1        393          2           150

```

Figure 15-70. onstat -g proxy (run from secondary server)

Output Description

Primary Node

Name of the primary server.

Proxy ID

ID of the proxy distributor. Proxy IDs are unique within a high-availability cluster.

Transaction Count

The number of transactions currently being processed by the proxy distributor.

Hot Row Total

Total number of hot rows ever handled by the proxy distributor.

Example Output

```

$ onstat -g proxy all

IBM Informix Dynamic Server Version 11.50.U      -- On-Line -- Up 00:08:28 -- 46392 Kbytes
Secondary      Proxy      Transaction  Hot Row
Node           ID          Count        Total
serv2          392         2            1
serv2          393         2            0

TID      Flags      Proxy Source  Proxy  Current  sqlerrno iserrno
         ID          ID  SessID TxnID   Seq
63      0x00000024 392  22     1      5      0      0
64      0x00000024 392  19     2      5      0      0
62      0x00000024 393  23     1      5      0      0
65      0x00000024 393  21     2      5      0      0

```

Figure 15-71. *onstat -g proxy all* Output (run from primary server)

Output Description

Secondary Node

Name of the secondary server as it is known by the primary server.

Proxy ID

ID of the proxy distributor. Proxy IDs are unique within a high-availability cluster.

Transaction Count

The number of transactions currently being processed by the proxy distributor.

Hot Row Total

Total number of hot rows ever handled by the proxy distributor.

TID ID of the proxy agent thread running on the primary server. This ID is created by the proxy distributor to handle work from the session on the secondary server.

Flags Flags of the proxy agent thread.

Proxy ID

The ID of the proxy distributor on behalf of which the proxy agent thread (TID) is running.

Source SessID

The ID of the user's session on the secondary server.

Proxy TxnID

The number of the current transaction. These numbers are unique to the proxy distributor.

Current Seq

The sequence number of the current operation in the current transaction.

sqlerrno

The error number of any SQL error (or 0 if no errors).

iserrno The error number of any ISAM or RSAM error (or 0 if no errors).

Example Output

```

$ onstat -g proxy all

IBM Informix Dynamic Server Version 11.50.U      -- Updatable (SDS) -- Up 00:13:34 -- 62776 Kbytes
Primary      Proxy      Transaction      Hot Row
Node         ID          Count           Total
serv1       3466        0               1
serv1       3465        1               0

Session      Proxy      Proxy      Proxy      Current      Pending      Reference
ID           ID         TID        TxnID      Seq          Ops          Count
19          3465      67         1          23           0            1

```

Figure 15-72. `onstat -g proxy all` Output (run from secondary server)

Output Description

Primary Node

Name of the primary server.

Proxy ID

ID of the proxy distributor. Proxy IDs are unique within a high-availability cluster.

Transaction Count

The number of transactions currently being processed by the proxy distributor.

Hot Row Total

Total number of hot rows ever handled by the proxy distributor.

Session

The session ID

Proxy ID

The ID of the proxy distributor on behalf of which the proxy agent thread (TID) is running.

Proxy TID

Transaction ID of the proxy agent thread running on the primary server. This ID is created by the proxy distributor to handle work from the secondary server session.

Proxy TxnID

The number of the current transaction. These numbers are unique to the proxy distributor.

Current Seq

The sequence number of the current operation in the current transaction.

Pending Ops

The number of operations buffered on the secondary server that have not yet been sent to the primary server.

Reference Count

Indicates the number of threads that are using the information for the current transaction. When the count becomes 0, the transaction processing is complete (either successfully or unsuccessfully).

Example Output

```

$onstat -g proxy 944

IBM Informix Dynamic Server Version 11.50.U      -- Updatable (SDS) -- Up 00:18:48 -- 62776 Kbytes
Proxy      Reference  Pending  ProxySID
TxnID      Count      Ops      3
19         1          0

```

Figure 15-73. *onstat -g proxy proxy_id* Output (run from secondary server)

Output Description

Proxy TxnID

The number of the current transaction. These numbers are unique to the proxy distributor.

Reference Count

Indicates the number of threads that are using the information for the current transaction. When the count becomes 0, the transaction processing is complete (either successfully or unsuccessfully).

Pending Ops

The number of operations buffered on the secondary server that have not yet been sent to the primary server.

Proxy SID

Proxy session ID.

Example Output

```

$onstat -g proxy 944 19

IBM Informix Dynamic Server Version 11.50.U      -- Updatable (SDS) -- Up 00:18:48 -- 62776 Kbytes
Sequence  Operation  rowid  Table      sqlerrno
Number    Type       Name
1         Update    10     db1:informix.products  0
2         Insert    10     db1:informix.products  0

```

Figure 15-74. *onstat -g proxy proxy_id proxy_transaction_id* Output (run from secondary server)

Output Description

Sequence Number

The number of the operation.

Operation Type

The type of operation to be performed. One of: Insert, Update, Delete, Other.

rowid The row ID of the row in which to apply the operation.

Table Name

The full table name, trimmed to fit a reasonable length. Format: database.owner.tablename

sqlerrno

The error number of any SQL error (or 0 if no errors).

Example Output

```

$onstat -g proxy 944 19 1

IBM Informix Dynamic Server Version 11.50.U      -- Updatable (SDS) -- Up 00:18:48 -- 62776 Kbytes
Sequence Operation rowid Table          sqlerrno
Number   Type
1        Update    10    db1:informix.products  0

```

Figure 15-75. `onstat -g proxy proxy_id proxy_transaction_id sequence_number` Output (run from secondary server)

Output Description

Sequence Number

The number of the operation.

Operation Type

The type of operation to be performed. One of: Insert, Update, Delete, Other.

rowid The row ID of the row in which to apply the operation.

Table Name

The full table name, trimmed to fit a reasonable length. Format: database.owner.tablename

sqlerrno

The error number of any SQL error (or 0 if no errors).

onstat -g que: Prints ER queue statistics

Syntax:

```
▶▶ onstat -g que ◀◀
```

The `onstat -g que` command prints statistics that are common to all queues in Enterprise Replication. The queuer manages the logical aspects of the queue. The RQM (reliable queue manager) manages the physical queue.

The `-g que` option is used primarily as a debugging tool and by Technical Support.

Example Output

In the following example, **Element high water mark** shows the maximum size of the transaction buffer header data (metadata) allowed in memory, shown in kilobytes. **Data high water mark** shows the maximum size of transactions for user data allowed in memory, shown in kilobytes.

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:40:28 -- 28672 Kbytes
CDR Queuer Statistics:
  Queuer state      : 2
  Local server     : 100
  Element high water mark : 131072
  Data high water mark : 131072
  # of times txns split : 0
  Total # of split txns : 0
  allowed log delta : 30
  maximum delta detected : 4
  Control Key      : 0/00000007
  Synchronization Key : 0/00000003
Replay Table:
  Replay Posn (Disk value): 12/00000018 (12/00000018)
  Replay save interval   : 10
  Replay updates        : 10
  Replay # saves        : 17
  Replay last save time  : (1095118157) 2004/09/13 18:29:17
Send Handles
  Server ID          : 200
  Send state,count   : 0,0
  RQM hdl for trg_send: Traverse handle (0xaf8e018) for thread CDRACK_0 at Head_of_Q,
  Flags: None
  RQM hdl for control_send: Traverse handle (0xaf74018)
  for thread CDRACK_0 at Head_of_Q, Flags: None
  RQM hdl for sync_send: Traverse handle (0xadc6018) for thread CDRACK_0 at Head_of_Q,
  Flags: None
  Server ID          : 200
  Send state,count   : 0,0
  RQM hdl for trg_send: Traverse handle (0xac8b018) for thread CDRACK_1 at Head_of_Q,
  Flags: None
  RQM hdl for control_send: Traverse handle (0xb1ce018) for thread CDRACK_1 at Head_of_Q,
  Flags: None
  RQM hdl for sync_send: Traverse handle (0xadc5018) for thread CDRACK_1 at Head_of_Q,
  Flags: None
  Server ID          : 200
  Send state,count   : 0,0
  RQM hdl for trg_send: Traverse handle (0xaea71d8) for thread CDRNsA200 at Head_of_Q,
  Flags: None
  RQM hdl for ack_send: Traverse handle (0xae8c1d8) for thread CDRNsA200 at Head_of_Q,
  Flags: None
  RQM hdl for control_send: Traverse handle (0xae9e1d8) for thread CDRNsA200 at Head_of_Q,
  Flags: None

```

Figure 15-76. *onstat -g que* Output

onstat -g qst: Print wait options for mutex and condition queues

Syntax:

►► onstat -g qst ◀◀

The **onstat -g qst** option displays wait statistics for mutex queues and condition queues (queues of waiters for a mutex or a condition). The QSTATS configuration parameter must be set to 1 to enable statistics collection. For more information, see "QSTATS" on page 1-80.

Example Output

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 02:11:18 -- 1067288 Kbytes
Mutex Queue Statistics
name      nwaits  avg_time max_time avgq maxq nservs  avg_time
ddh chai 1      1354863 1354863 1    1    56      1690

Condition Queue Statistics
name      nwaits  avg_time max_time avgq maxq nservs  avg_time
arrived  1      110008  110008  1    1    0        0
logbf0   21      642    4431   1    2    0        0
logbf1   15      475    2519   1    2    0        0
logbf2   19      596    3274   1    2    0        0
bp_cond  1        0      0       1    1    0        0
```

Figure 15-77. `onstat -g qst` Output

Output Description

name (string)

Name of the mutex or condition resource being waited for

nwaits (decimal)

Number of times this resource was waited for

avg_time (decimal)

Average time spent waiting (in microseconds)

max_time (decimal)

Maximum time spent waiting (in microseconds)

avgq (decimal)

Average length of the queue

maxq (decimal)

Maximum length of the queue

nservs (decimal)

Number of times this resource was acquired

avg_time (decimal, microsecond)

Average time the resource was held per acquisition (in microseconds)

onstat -g rbm: Print a block map of shared memory

Syntax:

►► onstat -g rbm ◀◀

The `onstat -g rbm` option prints a block map for the resident segment of shared memory.

Example Output

```

Block bitmap for resident segment address 0x44000000:
address = 0x440003bc, size(bits) = 3035
used = 3031, largest_free = 4

  0:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
256:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
512:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
768:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
1024:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
1280:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
1536:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
1792:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
2048:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
2304:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
2560:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff
2816:ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffff ffffffe00

```

Figure 15-78. `onstat -g rbm` Output

Output Description

Header

address (hex)

In-memory starting address of the used/free blocks in the segment

size (bits)

Number of bits in the block bitmap; each bit represents one block

used (blocks)

Used blocks in the bitmap

largest_free (blocks)

Largest run of free blocks

Data

Bit number (decimal): data (hex)

Bit number followed by 32 bytes of data (hex)

onstat -g rcv: Print ER receive manager statistics

Syntax:

```

▶▶ onstat -g rcv [server_id] [full] ◀◀

```

The `onstat -g rcv` command prints statistics about the receive manager in Enterprise Replication. The receive manager is a set of service routines between the receive queues and data sync.

The `onstat -g rcv` command has the following formats:

```

onstat -g rcv
onstat -g rcv server_id
onstat -g rcv full

```

The `server_id` modifier causes the command to print only those output messages received from the replication server whose group ID is `server_id`. The **full** modifier causes the command to print all statistics.

The **onstat -g rcv** command includes the Receive Manager global section. In this section, the following fields have the meanings shown:

Field	Description
cdrRM_DSParallelPL	Shows the current level of Apply Parallelism, 0 (zero) being the highest
cdrRM_DSNumLockTimeout cdrRM_DSNumLockRB cdrRM_DSNumDeadLocks	Indicate the number of collisions between various apply threads
cdrRM_acksinList	Shows acknowledgements that have been received but not yet processed

The **onstat -g rcv** command includes the Receive Parallelism Statistics section, a summary of the data sync threads by source server.

Field	Description
Server	Source server ID
Tot.Txn.	Total number of transactions applied from this source server
Pending	Number of current transactions in the pending list for this source server
Active	Number of current transactions currently being applied from this source server
MaxPnd	Maximum number of transactions in the pending list queue
MaxAct	Maximum number of transaction in the active list queue
AvgPnd	Average depth of the pending list queue
AvgAct	Average depth of the active list queue
CommitRt	Commit rate of transaction from this source server based on transactions per second

The Statistics by Source section of the **onstat -g rcv** command shows the following information for each source server. For each replicate ID:

- The number of transactions applied from the source servers
- The number of inserts, deletes, and updates within the applied transactions
- The timestamp of the most recently applied transaction on the target server
- The timestamp of the commit on the source server for the most recently applied transaction

The **-g rcv** option is used primarily as a debugging tool and by Technical Support. If you suspect that acknowledgement messages are not being applied, you can use this option to check.

Example Output

The following example shows output for the **onstat -g rcv full** command.

```

Receive Manager global block 0D452018
  cdrRM_inst_ct:          5
  cdrRM_State:           00000000
  cdrRM_numSleepers:     3
  cdrRM_DsCreated:       3
  cdrRM_MinDSThreads:    1
  cdrRM_MaxDSThreads:    4
  cdrRM_DSBlock:         0
  cdrRM_DSParallelPL:    0
  cdrRM_DSFailRate:      0.000000
  cdrRM_DSNumRun:        35
  cdrRM_DSNumLockTimeout 0
  cdrRM_DSNumLockRB:     0
  cdrRM_DSNumDeadLocks: 0
  cdrRM_DSNumPCommits:   0
  cdrRM_ACKwaiting:      0
  cdrRM_totSleep:        77
  cdrRM_Sleeptime:       153
  cdrRM_Workload:         0
  cdrRM_optscale:         4
  cdrRM_MinFloatThreads: 2
  cdrRM_MaxFloatThreads: 7
  cdrRM_AckThreadCount:  2
  cdrRM_AckWaiters:      2
  cdrRM_AckCreateStamp: Wed Sep 08 11:47:49 2004
  cdrRM_DSCreateStamp:  Wed Sep 08 14:16:35 2004
  cdrRM_acksInList:      0
  cdrRM_BlobErrorBufs:   0

Receive Parallelism Statistics
Srvr Tot.Txn. Pnding Active MaxPnd MaxAct AvgPnd AvgAct CommitRt
  1   35      0      0    21     3   7.00   1.63   0.00
  5    3      0      0     1     1   1.00   1.00   0.02
  6    6      0      0     1     1   1.00   1.00   0.21
Tot Pending:0 Tot Active:0 Avg Pending:5.77 Avg Active:1.50
Commit Rate:0.01

Time Spent In RM Parallel Pipeline Levels
Lev. TimeInSec Pcnt.
  0    17405 100.00%
  1      0  0.00%
  2      0  0.00%

Statistics by Source
Server 1
Repl Txn Ins Del Upd Last Target Apply Last Source Commit
65541 23  0  1 616 2004/09/08 14:20:15 2004/09/08 14:20:15
65542 11  0  0 253 2004/09/08 14:19:33 2004/09/08 14:19:33
65545  1  0  67  0 2004/09/08 14:20:37 2004/09/08 14:20:37
Server 5
Repl Txn Ins Del Upd Last Target Apply Last Source Commit
65541  3  0  0  81 2004/09/08 16:36:10 2004/09/08 16:36:09
Server 6
Repl Txn Ins Del Upd Last Target Apply Last Source Commit
65548  6  0  0  42 2004/09/08 16:37:59 2004/09/08 16:37:58

```

Figure 15-79. onstat -g rcv Output

onstat -g rea: Print ready threads

Syntax:

```
▶▶ onstat -g rea
```

The **onstat -g rea** option prints ready threads.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 01:59:34 -- 101376 Kbytes
```

```
Ready threads:
```

tid	tcb	rstcb	prty	status	vp-class	name
6	536a38	406464	4	ready	3cpu	main_loop()
28	60cfe8	40a124	4	ready	1cpu	onmode_mon
33	672a20	409dc4	2	ready	3cpu	sqlexec

Figure 15-80. *onstat -g rea* Output

onstat -g rep: Print ER schedule manager events

Syntax:

```
▶▶ onstat -g rep
```

```
└─ replname ─┘
```

The **onstat -g rep** command prints events that are in the queue for the schedule manager for Enterprise Replication. The **-g rep** option is used primarily as a debugging tool and by Technical Support.

The **onstat -g rep** command has the following formats:

```
onstat -g rep
onstat -g rep replname
```

The *repl_name* modifier limits the output to those events originated by the replicate named *repl_name*.

Example Output

The following example shows sample output for the **onstat -g rep** command.

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:30:10 -- 28672 Kbytes
```

```
Schedule manager Cb: add7e18 State: 0x8100 <CDRINIT,CDRRUNNING>
```

```
Event      Thread      When
=====
CDRDS      CDREvent    00:00:20
```

Figure 15-81. *onstat -g rep* Output

onstat -g rqm: Print low-level queue statistics

Syntax:

```
onstat -g rqm modifier
```

The **onstat -g rqm** command prints statistics and contents of the low-level queues (send queue, receive queue, ack send queue, sync send queue, and control send queue) managed by the Reliable Queue Manager (RQM) in Enterprise Replication. The RQM manages the insertion and removal of items to and from the various queues. The RQM also manages spooling of the in-memory portions of the queue to and from disk. The **-g rqm** option displays the contents of the queue, size of the transactions in the queue, how much of the queue is in memory and on disk, the location of various handles to the queue, and the contents of the various progress tables. You can choose to print information for all queues or for just one queue by using one of the modifiers described below.

If a queue is empty, no information is printed for that queue.

The **onstat -g rqm** command has the following formats:

```
onstat -g rqm  
onstat -g rqm modifier
```

The following table describes the values for *modifier*.

Modifier	Action
ACKQ	Prints the ack send queue
BRIEF	Prints a brief summary of the number of transactions in each of the queues and the replication servers for which the data is queued. Use this modifier to quickly identify sites where a problem exists. If large amounts of data are queued for a single server, then that server is probably down or off the network.
CNTRLQ	Prints the control send queue
FULL	Prints full information about every in-memory transaction for every queue
RECVQ	Prints the receive queue
SBSACES	Prints information about the sbspaces configured for ER
SENDQ	Prints the send queue
SYNCQ	Prints the sync send queue
VERBOSE	Prints all the buffer headers in memory

When you specify a modifier to select a specific queue, the command prints all the statistics for that queue and information about the first and last in-memory transactions for that queue.

The other modifiers of the **onstat -g rqm** command are used primarily as a debugging tool and by Technical Support.

The output for the SENDQ modifier contains the following sections:

- RQM Statistics for Queue—a summary of current and historical information for the queue. This includes the number of transactions in the queue, how many are

spooled, how many bytes they are using, some maximum statistics, and the high water marks that will trigger stably storing transactions in the **syscdr** tables.

- **First Txn**—information about the first transaction in the queue. To check if the queue is draining, you can run **onstat -g rqm** several times and see if the first transaction's RQM key is changing. The RQM key has the following format: *Server_ID/Commit_unique_logID/Commit_log_position/Sequence*. If it is not draining, the target server may be offline or some other problem is occurring. The **NeedAck** field shows from which server the transaction is waiting for an acknowledgement. You can use this bitmap mask with the output from the **onstat -g cat** command to determine the name of the server which server Enterprise Replication is waiting on for an acknowledgement.
- **Last Txn**—information about the last transaction in the queue
- **Traverse handle**—lists the handles used for threads
- **Progress table**—provides information about the progress of each replicate under the headers: **Server**, **Group**, **Bytes Queued**, **Acked**, and **Sent**. The **Group** field shows the replicate ID. The **Acked** field shows what has been acknowledged. The **Sent** field shows which entries are now in transit. Both the **Acked** and the **Sent** field show the RQM key, which has the following format: *Server_ID/Commit_unique_logID/Commit_log_position/Sequence*.

Example Output

The following example shows output for the **onstat -g rqm SENDQ** command.

```
RQM Statistics for Queue (0x0D3DF018) trg_send
Transaction Spool Name: trg_send_stxn
Insert Stamp: 35/0
Flags: SEND_Q, SPOOLED, PROGRESS_TABLE, NEED_ACK
Txns in queue:          35
Log Events in queue:    0
Txns in memory:         35
Txns in spool only:     0
Txns spooled:           0
Unspooled bytes:        176206
Size of Data in queue:  176206 Bytes
Real memory in use:     176206 Bytes
Pending Txn Buffers:    0
Pending Txn Data:       0 Bytes
Max Real memory data used: 176206 (2457600) Bytes
Max Real memory hdrs used 65988 (2457600) Bytes
Total data queued:      176206 Bytes
Total Txns queued:      35
Total Txns spooled:     0
Total Txns restored:    0
Total Txns recovered:   0
Spool Rows read:        0
Total Txns deleted:     0
Total Txns duplicated:  0
Total Txn Lookups:      363

First Txn (0x0D60C018) Key: 1/9/0x000d4bb0/0x00000000
Txn Stamp: 1/0, Reference Count: 0.
Txn Flags: Notify
Txn Commit Time: (1094670993) 2004/09/08 14:16:33
Txn Size in Queue: 5908
First Buf's (0x0D31C9E8) Queue Flags: Resident
First Buf's Buffer Flags: TRG, Stream
NeedAck: Waiting for Acks from <[0004]>
No open handles on txn.

Last Txn (0x0D93A098) Key: 1/9/0x00138ad8/0x00000000
Txn Stamp: 35/0, Reference Count: 0.
Txn Flags: Notify
```

```

Txn Commit Time: (1094671237) 2004/09/08 14:20:37
Txn Size in Queue: 6298
First Buf's (0x0D92FFA0) Queue Flags: Resident
First Buf's Buffer Flags: TRG, Stream
NeedAck: Waiting for Acks from <[0004]>
Traverse handle (0x0D045018) for thread CDRNsA3 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D08E018) for thread CDRNsA4 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D523018) for thread CDRNsA5 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D0D9018) for thread CDRNsA6 at txn (0x0D93A098)
End_of_Q,Flags: None

Traverse handle (0x0D4041D8) for thread CDRNsA2 at Head_of_Q,Flags: None
Traverse handle (0x0D3F01D8) for thread CDRNrA2 at Head_of_Q, Flags: None
Traverse handle (0x0D045018) for thread CDRNsA3 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D31C018) for thread CDRNrA3 at Head_of_Q, Flags: None
Traverse handle (0x0D08E018) for thread CDRNsA4 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D4C8018) for thread CDRNrA4 at Head_of_Q, Flags: None
Traverse handle (0x0D523018) for thread CDRNsA5 at txn (0x0D93A098)
End_of_Q,Flags: None
Traverse handle (0x0D57F018) for thread CDRNrA5 at Head_of_Q, Flags: None
Traverse handle (0x0D0D9018) for thread CDRNsA6 at txn (0x0D93A098)
End_of_Q,Flags: None

```

Server	Group	Bytes Queued	Acked	Sent
6	0x10009	0	1/9/138ad8/0	- 1/9/138ad8/0
5	0x10009	0	1/9/138ad8/0	- 1/9/138ad8/0
4	0x10009	0	1/9/138ad8/0	- 1/9/138ad8/0
3	0x10009	0	1/9/138ad8/0	- 1/9/138ad8/0
2	0x10009	4154	ffffff/ffffff/ffffff/ffffff	- 1//138ad8/0
6	0x10006	0	1/9/12d8f8/0	- 1//12d8f8/0
5	0x10006	0	1/9/12d8f8/0	- 1//12d8f8/0
4	0x10006	0	1/9/12d8f8/0	- 1/9/12d8f8/0
3	0x10006	0	1/9/12d8f8/0	- 1/9/12d8f8/0
2	0x10006	31625	ffffff/ffffff/ffffff/ffffff	- 1/9/12d8f8/0

onstat -g rss: Print RS secondary server information

Syntax:

```

▶▶ onstat -g rss [ -verbose ] [ -log ] [ server_name ]

```

The **onstat -g rss** command prints RS secondary server information. The output of the **onstat -g rss** command differs slightly depending on whether the command is run on the primary server or on the RS secondary server.

Invocation	Explanation
onstat -g rss	Displays brief RS secondary server information
onstat -g rss verbose	Displays detailed RS secondary server information
onstat -g rss log	Displays log information. This command is only applicable when run on the primary server.

Invocation	Explanation
<code>onstat -g rss server_name</code>	Displays information about a specific RS secondary server. This command is only applicable when run on the primary server.

Example Output

```

Local server type: Primary
Index page logging status: Enabled
Index page logging was enabled at: 2007/02/20 18:10:01
Number of RSS servers: 3

RSS Server information:

RSS Srv      RSS Srv      Connection      Next LPG to send      Supports
name         status        status           (log id,page)         Proxy Writes
cdr_ol_nag_1_c1  Active      Connected        7,899                  Y
cdr_ol_nag_1_c2  Active      Connected        7,899                  Y
cdr_ol_nag_1_c3  Active      Connected        7,899                  Y

```

Figure 15-82. `onstat -g rss` Output (run on primary server)

Output Description

Local server type

Primary or RSS (remote standalone secondary) server type

Index page logging status

Displays whether index page logging is enabled or disabled between primary server and secondary server

Index page logging was enabled at

Date and time that index page logging was enabled

Number of RSS servers

Number of RS secondary servers connected to the primary server

RSS Srv name

Name of RS secondary server

RSS Srv status

Displays whether RS secondary server is active or not

Connection status

Connection status of RS secondary server

Next LPG to send (log id, page)

LPG log ID and page

Supports Proxy Writes

Displays whether the server is currently configured to allow updates to secondary servers. **Y** = supports updates to secondary servers, **N** = does not support updates to secondary servers.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- Read-Only (RSS) -- Up 00:05:18 -- 55296 Kbytes

Local server type: RSS
Server Status : Active
Source server name: cdr_ol_nag_1
Connection status: Connected
Last log page received(log id,page): 7,877

```

Figure 15-83. *onstat -g rss* Output (run on RS secondary server)

Output Description

Local server type

Primary or RSS (remote standalone secondary) server type

Server Status

Displays whether RS secondary server is active

Source server name

Name of the primary server

Connection status

Connection status of RS secondary server

Last log page received (log id,page)

Most recent log ID and page received

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- On-Line -- Up 00:08:57 -- 47104 Kbytes

Log Pages Snooped:
RSS Srv      From      From      Tossed
name         Cache     Disk      (LBC full)
cdr_ol_nag_1_c1  1368     1331     0
cdr_ol_nag_1_c2  1357     1342     0
cdr_ol_nag_1_c3  1356     1343     0

```

Figure 15-84. *onstat -g rss log* Output (run on primary server)

Output Description

Log Pages Snooped

Statistics for each RS secondary server

RSS Srv name

RS secondary server name

From Cache

From cache number

From Disk

Log from disk

Tossed (LBC full)

Number of log pages discarded as a result of the LBC becoming full

onstat -g rwm: Print read and write mutexes

Syntax:

▶▶ onstat -g rwm ◀◀

The **onstat -g rwm** command prints read and write mutexes.

Example Output

```
MUTEX  NAME      write/read/wait  tcb list
<address> <name>      first mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
<address> <name>      second mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
....
....
....
<address> <name>      last mutex
  Writer  ticket = <ticket address>  tcb=<thread address> <thread name>
  Readers ticket = <ticket address>  tcb=<thread address> <thread name>
  Waiters ticket = <ticket address>  tcb=<thread address> <thread name>
```

Figure 15-85. onstat -g rwm Output

Output Description

tcb List of thread addresses

Writer List of write threads

Readers
List of read threads

Waiters
List of waiting threads

ticket Address of ticket acquired by the thread

onstat -g sch: Print VP information

Syntax:

▶▶ onstat -g sch ◀◀

The **onstat -g sch** option prints the number of semaphore operations, spins, and busy waits for each virtual processor.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 02:00:03 -- 101376 Kbytes
VP Scheduler Statistics:
vp  pid  class  semops  busy waits  spins/wait
1   3284  cpu    23997   0           0
2   1340  adm    0       0           0
3   4624  lio    2       0           0
4   3320  pio    2       0           0
5   6076  aio    7710   0           0
6   4580  msc    46     0           0
7   3428  soc    7       0           0
8   2308  soc    1       0           0

Thread Migration Statistics:
vp  pid  class  steal-at  steal-sc  idlvp-at  idlvp-sc  inl-polls  Q-ln
1   3284  cpu    0         0         0         0         0         0
2   1340  adm    0         0         0         0         0         0
3   4624  lio    0         0         0         0         0         0
4   3320  pio    0         0         0         0         0         0
5   6076  aio    0         0         0         0         0         0
6   4580  msc    0         0         0         0         0         0
7   3428  soc    0         0         0         0         0         0
8   2308  soc    0         0         0         0         0         0

```

Figure 15-86. `onstat -g sch` Output

onstat -g sds: Print SD secondary server information

Syntax:

```

▶▶ onstat -g sds [verbose server_name]

```

The `onstat -g sds` command prints SD secondary server information. The output of the `onstat -g sds` command differs slightly depending on whether the command is run on the primary server or on the SD secondary server.

Invocation	Explanation
<code>onstat -g sds</code>	Displays brief SD secondary server information
<code>onstat -g sds verbose</code>	Displays detailed SD secondary server information
<code>onstat -g sds server_name</code>	Displays information about a specific SD secondary server. When <code>server_name</code> is specified, the command must be issued from the primary server.

Example Output

```

Local server type: Primary
Number of SDS servers:1

SDS server information

SDS srv      SDS srv      Connection      Last LPG sent  Supports
name         status       status          (log id,page)  Proxy Writes
C_151162     Active       Connected       554,4998       Y

```

Figure 15-87. *onstat -g sds* Output (run from primary server)

Output Description

- Local server type*
Primary or SDS (shared disk secondary) server type
- Number of SDS servers*
Number of SD secondary servers connected to the primary server
- SDS Srv name*
Name of SD secondary server
- SDS Srv status*
Displays whether SD secondary server is active
- Connection status*
Displays whether SD secondary server is connected
- Last LPG sent (log id, page)*
Most recent LPG log ID and page
- Supports Proxy Writes*
Displays whether the server is currently configured to allow updates to secondary servers. **Y** = supports updates to secondary servers, **N** = does not support updates to secondary servers.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- On-Line -- Up 00:06:17 -- 38912 Kbytes

Number of SDS servers:2
Updater node alias name: cdr_ol_nag_1

SDS server control block: 0xb3cd8d0
server name: cdr_ol_nag_1_sdc1
server type: SDS
server status: Active
connection status: Connected
Last log page sent(log id,page):7,884
Last log page flushed(log id,page):7,884
Last LSN acked (log id,pos):7,3621272
Sequence number of next buffer to send: 176
Sequence number of last buffer acked: 0
Time of last ack:2007/02/20 21:04:13
Total LSNs posted:0
Total LSNs sent:0
Total page flushes posted:0
Total page flushes sent:0
Supports Proxy Writes: Y

SDS server control block: 0xc09bbd8
server name: cdr_ol_nag_1_sdc2
server type: SDS
server status: Active
connection status: Connected
Last log page sent(log id,page):7,884
Last log page flushed(log id,page):7,884
Last LSN acked (log id,pos):7,3621272
Sequence number of next buffer to send: 173
Sequence number of last buffer acked: 0
Time of last ack:2007/02/20 21:04:13
Total LSNs posted:0
Total LSNs sent:0
Total page flushes posted:0
Total page flushes sent:0
Supports Proxy Writes: Y

```

Figure 15-88. *onstat -g sds verbose* Output (run from primary server)

Output Description

Number of SDS servers

Number of attached SDS (shared disk secondary) servers

Updater node alias name

Name of primary server

SDS server control block

SD secondary server control block

server types

Server type

server status

Active or inactive

connection status

Status of connection between primary and secondary server

Last log page sent (log id, page)

Log ID and page of most recent log page sent

Last log page flushed (log id, page)

Log ID and page of the most recent log page flushed

Last LSN acked (log id, pos)
Most recent LSN (log position) acknowledged

Sequence number of next buffer to send
Sequence number of next buffer to send

Sequence number of next buffer acked
Sequence number of next buffer acknowledged

Time of last ack
Date and time of last log acknowledgement

Total LSNs posted
Total number of log position reports

Total LSNs sent
Total number of log position reports sent

Total page flushes posted
Total page flushes posted

Total page flushes sent
Total page flushes sent

Supports Proxy Writes
Displays whether the server is currently configured to allow updates to secondary servers. **Y** = supports updates to secondary servers, **N** = does not support updates to secondary servers.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- Read-Only (SDS) -- Up 00:03:17 -- 47104 Kbytes
SDS server control block: 0xb299880
Local server type: SDS
Server Status : Active
Source server name: cdr_ol_nag_1
Connection status: Connected
Last log page received(log id,page): 7,884
Next log page to read(log id,page):7,885
Last LSN acked (log id,pos):7,3621272
Sequence number of last buffer received: 0
Sequence number of last buffer acked: 0
Current paging file:/work1/nagaraju/dbspaces/page_cdr_ol_nag_1_sdc1_
Current paging file size:2048
Old paging file:/work1/nagaraju/dbspaces/page_cdr_ol_nag_1_sdc1_
Old paging file size:10240

```

Figure 15-89. *onstat -g sds* verbose Output (run from SD secondary server)

Output Description

SDS server control block
SD secondary server control block

Local server type
Primary or SDS (shared disk secondary) server type

Server status
Displays whether SD secondary server is active

Source server name
Displays name of primary server

Connection status
Displays whether SD secondary server is connected

Last log page received (log id, page)
Most recent log page received

Next log page to read (log id,page)
Next log page in sequence to read

Last LSN acked (log id,pos)
Most recent LSN acknowledged

Sequence number of last buffer received
Sequence number of last buffer received

Sequence number of last buffer acked
Sequence number of last buffer acknowledged

Current paging file
Name of current paging file

Current paging file size
Size of current paging file

Old paging file
Name of previous paging file

Old paging file size
Size of previous paging file

onstat -g seg: Print shared memory segment statistics

Syntax:

►► onstat — -g — seg —————►►

The **onstat -g seg** option prints shared-memory segment statistics. This option shows how many segments are attached and their sizes.

For information about running **onstat -g seg** on a dump file created without the buffer pool, see “Running onstat Commands on a Shared Memory Dump File” on page 15-8.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 02:00:13 -- 101376 Kbytes
Segment Summary:
id      key          addr      size      ovhd      class blkused blkfree
4001    1382438913  a000000  19918848  1760      R      4820    43
(shared) 1382438913  b2ff000  8392704   928       V      2049    0
3       1382438914  bb00000  9437184   952       V      2304    0
5       1382438915  c400000  8388608   920       V      1724    324
7       1382438916  cc00000  32505856  1656      V      7936    0
8       1382438917  eb00000  8388608   920       V      282     1766
9       1382438918  f300000  8388608   920       V      393     1655
10      1382438919  fb00000  8388608   920       V      393     1655
Total:  -        -        103809024 -        -        19901   5443
(* segment locked in memory)

```

Figure 15-90. `onstat -g seg` Output

onstat -g ses: Print session-related information

Syntax:

```

▶▶ onstat -g ses [sessionid] ◀◀

```

The `onstat -g ses` option prints session-related information. By default, only the DBSA can view `onstat -g ses` `syssqltrace` information. However, when `UNSECURE_ONSTAT = 1` all users can view this information. You can specify one of the following invocations.

Invocation

Explanation

`onstat -g ses`

Displays a one-line summary for each session

`onstat -g ses sessionid`

Displays information for a specific session

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 7 days 18:43:13 --
38912 Kbytes

session
id      user      tty      pid      hostname  #RSAM  total  used  dynamic
                 memory  memory  explain
24      informix -        0        -         0      12288  7936  off
23      informix -      17602   carson    1      57344  48968  off
3       informix -        0        -         0      12288  9168  off
2       informix -        0        -         0      12288  7936  off

```

Figure 15-91. `onstat -g ses` Output

```

IBM Informix Dynamic Server Version 11.50.UC1  -- On-Line -- Up 2 days 19:42:
11 -- 38912 Kbytes

session
id      user      tty      pid      hostname #RSAM  total  used  dynamic
16      sitaramv  1       18523    carson   1      81920  71720 off

tid     name      rstcb    flags    curstk   status
35      sqlxec    a7ed9f4 Y--P--- 5488     cond wait(netnorm)

Memory pools  count 1
name        class addr      totalsize freesize #allocfrag #freefrag
16          V      afea020  81920    10200    119       13

name        free      used      name        free      used
overhead    0         1648     scb          0         96
opentable   0         1768     filetable    0         336
log         0         21880    temprec      0         16200
keys        0         680      ralloc       0         5120
gentcb      0         1208     ostcb        0         2528
sqscb       0         13216    sql          0         40
rdahead     0         184      hashfiletab  0         280
osenv       0         1920     sqtcb        0         2024
fragman     0         208      udr          0         312
xatm        0         2072

sqscb info
scb      sqscb  optofc  pdqpriority  sqlstats  optcompind  directives
adff580 af93018 0        0            0          2            1

Sess SQL      Current      Iso Lock      SQL  ISAM F.E.
Id  Stmt type  Database     Lvl Mode      ERR  ERR  Vers Explain
16  -        xabasicdb   RR  Not Wait    0    0    9.03 Off

Last parsed SQL statement :
EXECUTE FUNCTION xa2pc_mi_unregister("xads_t2_i2")

Xadatasources participated in this session :
Xadatasource name                                RMID      Active
xabasicdb@atmo110:sitaramv.xads_t3_i1           6         YES
xabasicdb@atmo110:sitaramv.xads_t2_i1           4         YES
xabasicdb@atmo110:sitaramv.xads_t1_i3           3         YES
xabasicdb@atmo110:sitaramv.xads_t1_i2           2         YES
xabasicdb@atmo110:sitaramv.xads_t1_i1           1         YES
xabasicdb@atmo110:sitaramv.xads_t2_i2           5         NO

DRDA client info
  Userid:
  Wrkstnname: nemea
  Applname: db2jcc_application
  Acctng: JCC03510nemea
  Programid:
  Autocommit:
  Packagepath:

```

Figure 15-92. onstat -g ses sessionid Output

You can interpret the output from this option as follows:

Session section

Session id

The session ID

user The username who started the session

tty The tty associated with the front-end for this session

pid The process ID associated with the front-end for this session

hostname
The hostname from which this session has connected

#RSAM threads
The number of RSAM thread allocated for this session

total memory
The amount of memory allocated for this session

used memory
The amount of memory actually used by this session

dynamic explain
Generate explain output of the sql statements of the session (on or off)

Threads section

tid The thread ID

name The name of the thread

rstcb RSAM control block

flags Describes the status of the thread using the following codes:

Position 1

B Waiting on a buffer

C Waiting on a checkpoint

G Waiting on a logical-log buffer write

L Waiting on a lock

S Waiting on a mutex

T Waiting on a transaction

X Waiting on a transaction cleanup

Y Waiting on a condition

Position 2

*** An asterisk in this position means that the thread encountered an I/O failure in the middle of a transaction

Position 3

A Archive thread

B Begin work

P Begin Prepare or Prepared work

X XA prepared

C Committing or committed

R Aborting or aborted

H Heuristically aborted or heuristically rolling back

Position 4

P Primary thread

Position 5

R Reading

X Critical section

Position 6

R Recovery thread

Position 7

M Monitor thread

D Daemon thread

C Cleaner

F Flusher

B B-tree scanner

curstk Current stack size

status Current thread status

Memory pools header section. The information is repeated for each session pool.

name Name of pool

class Class of the memory where the pool is allocated from. R is for Resident, V is for Virtual, and M is for Message

addr Address of the pool structure

totalsize

Total size of the memory acquired by the pool in bytes

freesize Number of bytes free in the pool

#allocfrag

Number of allocated memory fragments in the pool

#freefrag

Number of free fragments in the pool

The memory pool section

name Name of a component which has allocated memory from the pool

free Number of bytes freed

used Number of bytes allocated

The sqscb information section

scb The session control block. This is the address of the main session structure in shared memory.

sqscb SQL level control block of the session

optofc The current value of the **OPTOFC** environment variable or onconfig setting.

pdqpriority

The current value of the **PDQPRIORITY** environment variable or onconfig setting.

sqlstats The current value of the **SQLSTATS** environment variable or onconfig setting.

optcompind

The current value of the **OPTCOMPIND** environment variable or onconfig setting.

directives

The current value of the **DIRECTIVES** environment variable or onconfig setting.

The Last parsed SQL statement section has the same information as the **onstat -g sql** option. See “onstat -g sql: Print SQL-related session information” on page 15-122.

Xdatasources participated in this session section shows information about the XA data sources that are available during the session, their resource manager identifiers, and whether they are currently active.

Xdatasource name

The XA data source that participated in the session

RMID The identifier of the resource manager for the corresponding XA data source

Active Whether the XA data source is still active

The **DRDA client info** section shows information on Distributed Relational Database Architecture (DRDA) connections to clients.

Userid

User ID of the client user

Wrkstnname

Name of the client workstation

Applname

Name of the client application, for example db2jcc_application

Acctng

Accounting string from the client, for example JCC03510nemea

Programid

Client program identifier (not used by Dynamic Server)

Autocommit

Default transaction autocommit mode for Dynamic Server data sources

Packagepath

Client package path (not used by Dynamic Server)

onstat -g sle: Print all sleeping threads

Syntax:

▶▶ onstat -g sle ▶▶

The **onstat -g sle** option prints all sleeping threads.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 02:00:27 -- 101376 Kbytes
Current Admin VP sleep period: 10 millisecs
Sleeping threads with timeouts: 21 threads
  tid v_proc      rstcb      name      time
   49   1      b3b13a8    onmode_mon 0.02
    5   1           0    Cosvr Avail Mgr 0.05
   42   1      b3ad028    main_loop() 0.08
    9   3      b3ad6e8      xtm_svcc 0.64
   14   5           0    mgmt_thd_5 0.65
   13   4           0    mgmt_thd_4 0.65
    4   1           0    mgmt_thd_1 0.65
    6   3           0      dfm_svc 0.98
   33  13           0    mgmt_thd_13 1.54
   27  10           0    mgmt_thd_10 1.54
   21   7           0    mgmt_thd_7 1.54
   12   3           0    mgmt_thd_3 1.76
   29  11           0    mgmt_thd_11 1.76
   23   8           0    mgmt_thd_8 2.08
   31  12           0    mgmt_thd_12 2.08
   35  14           0    mgmt_thd_14 2.98
   19   6           0    mgmt_thd_6 3.00
   25   9           0    mgmt_thd_9 3.00
   37   3           0      sch_rgm 3.48
   44   5      b3af8a8    btscanner 0 7.31
   46   3      b3b0628    bum_sched 41.26
```

Figure 15-93. `onstat -g sle` Output

onstat -g smb: Print sbspaces information

Syntax:

```
▶▶ onstat -g smb c fdd lod s ◀◀
```

The `onstat -g smb` option prints detailed information about sbspaces:

Invocation

Explanation

`onstat -g smb c`

lists all the chunks in the sbspace

`onstat -g smb fdd`

lists the smart-large-object file descriptors

`onstat -g smb lod`

lists the smart-large-object headers in the header table

`onstat -g smb s`

lists the sbspace attributes (owner, name, page size, `-Df` flag settings).
Fields with a value of 0 or -1 were not initialized during sbspace creation.

The `onstat -g smb c` command displays for each sbspace chunk the following:

- chunk number and sbspace name
- chunk size and pathname
- total user data pages and free user data pages
- location and number of pages in each user-data and metadata areas

Use the **onstat -g smb c** option to monitor the amount of free space in each sbspace chunk, and the size in pages of the user-data, metadata, and reserved areas. In the following example, chunk 2 of sbspace1 has 2253 used pages (usr pgs) and 2245 free pages (free pg). For the first user-data area Ud1, the starting page offset is 53 and the number of pages is 1126. For the metadata area Md, the starting page offset is 1179 and the number of pages is 194. For the reserved data Ud2, the starting page offset is 1373 and the number of pages is 1127.

Chunk Summary:

```

sbnm 2 chunk 2
chunk:  address  flags  offset  size  orig fr  usr pgs  free pg
        303cf2a8  F-----  0       2500  2253    2253    2245
        path: /usr11/myname/sbspace1

        start pg  npages
Ud1   :   53     1126
Md    :  1179    194
Ud2   :  1373   1127

```

The **onstat -g smb s** command displays the storage attributes for all sbspaces in the system:

- sbspace name, flags, owner
- logging status
- average smart-large-object size
- first extent size, next extent size, and minimum extent size
- maximum I/O access time
- lock mode

For more information on the **onstat -g smb** option, see the *Performance Guide*.

onstat -g smx: Print multiplexer group information

Syntax:

```

▶▶ onstat -g smx [ses]

```

The **onstat -g smx** command prints server multiplexer group information for servers using SMX.

Invocation	Explanation
onstat -g smx	Displays SMX connection statistics
onstat -g smx ses	Displays SMX session statistics

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- On-Line (Prim) -- Up 00:08:06 -- 47104 Kbytes

SMX connection statistics:
SMX control block: 0x10b01c028

Peer server name: serv1_c1
SMX connection address: 0x10c2570d0
Encryption status: Enabled
Total bytes sent: 2758764
Total bytes received: 1608
Total buffers sent: 756
Total buffers received: 36
Total write calls: 95
Total read calls: 36
Total retries for write call: 1

```

Figure 15-94. `onstat -g smx` Output

Output Description

SMX control block

SMX control block

Peer server name

Displays the name of the peer server

SMX connection address

Displays the address of the SMX connection

Encryption status

Displays whether encryption is enabled or disabled

Total bytes sent

Displays the total number of bytes sent

Total bytes received

Displays the total number of bytes received

Total buffers sent

Displays the total number of buffers sent

Total buffers received

Displays the total number of buffers received

Total write calls

Displays the total number of write calls

Total read calls

Displays the total number of read calls

Total retries for write call

Displays the total number of retries for write call

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1      -- On-Line -- Up 00:25:36 -- 248832 Kbytes

SMX session statistics:
SMX control block: 0x17c69028

Peer      SMX session      client      reads      writes
name     address             type
delhi_sec 19022050    smx Clone Send      6          183

```

Figure 15-95. `onstat -g smx ses` Output

Output Description

SMX control block

SMX control block

Peer name

Displays the name of the peer server

SMX session address

SMX session address

Client type

Displays type of secondary server

reads Displays the total number of session reads

writes Displays the total number of session writes

onstat -g spi: Print spin locks with long spins

Syntax:

▶▶ onstat -g spi ◀◀

The **onstat -g spi** option prints spin locks with long spins.

Many resources in the server are accessed by two or more threads. In some of these accesses (such as updating a shared value), the server must guarantee that only one thread is accessing the resource at a time. A spin lock is the mechanism used to provide this mutually exclusive access for some resources. With this type of lock, a thread that did not succeed in acquiring the lock on the first try (because another thread was holding it) repeatedly attempts to acquire the lock until it succeeds.

The overhead cost of a spin lock is small, and spin locks are normally used for resources that require mutual exclusion for short periods of time. However, if a spin lock becomes highly contended, the loop-and-retry mechanism can become expensive.

The **onstat -g spi** option is helpful for identifying performance bottlenecks that are caused by highly contended spin locks. This option lists spin locks with waits, those spin locks for which a thread was not successful in acquiring the lock on its first attempt and thus had to loop and re-attempt.

Example Output

```

IBM Informix Dynamic Server Version 11.50      -- On-Line -- Up 04:13:15 -- 1067288 Kbytes

Spin locks with waits:

Num Waits   Num Loops   Avg Loop/Wait   Name
114         117675     1032.24         lockfr3
 87         256461     2947.83         fast mutex, lockhash[832]
 1          11         11.00          fast mutex, 1:bhash[16668]
 4          51831     12957.75        fast mutex, 1:lru-4
 1          490       490.00         fast mutex, 1:bf[994850] 0xe00002 0x14eb32000

```

Figure 15-96. `onstat -g spi` Output

Output description

Num Waits (decimal)

Total number of times a thread waited for this spin lock.

Num Loops (decimal)

Total number of attempts before a thread successfully acquired the spin lock.

Avg Loop/Wait (floating point)

Average number of attempts needed to acquire the spin lock. Computed as Num Loops / Num Waits.

Name (string)

Uses the following codes to name the spin lock

lockfr The lock free list. The number after **lockfr** is the index into the lock free list array.

lockhash[] The lock hash bucket. The field inside the brackets is the index into the lock hash bucket array.

:bhash [] The buffer hash bucket. The field before the colon is the buffer pool index; the field inside the brackets after **bhash** is the index into the buffer hash bucket array.

:lru- The LRU latch. The field before the colon is the buffer pool index; the field after **lru-** identifies the buffer chain pairs that are being used.

:bf[] The buffer latch. The field before the colon is the buffer pool index; the field inside the brackets after **bf** is the position of buffer in the buffer array. The next two fields are the partition number and the page header address in memory for the buffer in hex form.

onstat -g sql: Print SQL-related session information

Syntax:

►► `onstat -g sql—sessionid` ◄◄

The **onstat -g sql** option prints SQL-related information about a session. By default, only the DBSA can view **onstat -g sql** `syssqltrace` information. However, when `UNSECURE_ONSTAT = 1` all users can view this information. You can specify one of the following invocations.

Invocation

Explanation

onstat -g sql

Displays a one line summary for each session

onstat -g sql *sessionid*

Displays SQL information for a specific session

Note: Encrypted passwords and password hint parameters in encryption functions are not shown. Figure 15-97 displays an encrypted password in the Last parsed SQL statement field.

```
onstat -g sql 22
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:07:38 -- 19456 Kbytes
Sess SQL Current Iso Lock SQL ISAM F.E. Current
Id Stmt type Database Lvl Mode ERR ERR Vers Explain Role
22 - test CR Not Wait 0 0 9.03 Off hr
Last parsed SQL statement :
select id, name, decrypt_char(ssn, 'XXXXXXXXXX') from emp
```

Figure 15-97. *onstat -g sql* Output

Output description:

Sess id The session identifier

SQL Stmt type

The type of SQL statement

Current Database

Name of the current database of the session

ISO Lvl

Isolation level

DR Dirty Read

CR Committed Read

CS Cursor Stability

DRU Dirty Read, Retain Update Locks

CRU Committed Read, Retain Update Locks

CSU Cursor Stability, Retain Update Locks

LC Committed Read, Last Committed

RR Repeatable Read

NL Database Without Transactions

Lock mode

Lock mode of the current session

SQL Error

SQL error number encountered by the current statement

ISAM Error

ISAM error number encountered by the current statement

F.E. Version

The version of the SQLI protocol used by the client program

Explain

SET EXPLAIN setting

Current Role

Role of the current user

onstat -g src: Patterns in shared memory

Syntax:

▶▶ onstat -g src <pattern> <mask> ◀◀

The **onstat -g src** option searches for patterns in shared memory.

Example Output

The following example shows output for the **onstat -g src <pattern> <mask>** command where *pattern* = 0x123 and *mask* = 0xffff.

```
Search Summary:
addr           contents
00000000ad17a50: 01090000 00000000 00000000 00000123  ....#
00000000ad7dec0: 00000001 014e3a0c 00000000 0ade0123  ....N:  ....#
```

Figure 15-98. onstat -g src Output

Output description

addr (hexidecimal)

Address in shared memory where search pattern is found

contents (hexidecimal)

Contents of memory at given address

onstat -g ssc: Print SQL statement occurrences

Syntax:

▶▶ onstat -g ssc ◀◀

The **onstat -g ssc** command monitors the number of times that the database server reads the SQL statement in the cache. By default, only the DBSA can view **onstat -g ssc** syssqltrace information. However, when UNSECURE_ONSTAT = 1 all users can view this information.

Example Output

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:08:26 -- 29696 Kbytes

Statement Cache Summary:

```
#lrus  currsz  maxsz  Poolsize #hits nolimit
4      117640 524288 139264  0      1
```

Statement Cache Entries:

```
lru hash ref_cnt hits flag heap_ptr      database      user
-----
0 262    0   7  -F aad8038      sscsi007      admin
INSERT INTO ssc1 ( t1_char , t1_short , t1_key , t1_float , t1_smallfloat
, t1_decimal , t1_serial ) VALUES ( ? , ? , ? , ? , ? , ? , ? )
0 127    0   9  -F b321438      sscsi007      admin
INSERT INTO ssc2 ( t2_char , t2_key , t2_short ) VALUES ( ? , ? , ? )
1 134    0  15  -F aae0c38      sscsi007      admin
SELECT t1_char , t1_short , t1_key , t1_float , t1_smallfloat ,
t1_decimal , t1_serial FROM ssc1 WHERE t1_key = ?
1 143    0   3  -F b322c38      sscsi007      admin
INSERT INTO ssc1 ( t1_char , t1_key , t1_short ) SELECT t2_char , t2_key
+ ? , t2_short FROM ssc2
2 93     0   7  -F aae9838      sscsi007      admin
DELETE FROM ssc1 WHERE t1_key = ?
2 276    0   7  -F aaefc38      sscsi007      admin
SELECT count ( * ) FROM ssc1
2 240    1   7  -F b332838      sscsi007      admin
SELECT COUNT ( * ) FROM ssc1 WHERE t1_char = ? AND t1_key = ? AND
t1_short = ?
3 31     0   7  -F aaec038      sscsi007      admin
SELECT count ( * ) FROM ssc1 WHERE t1_key = ?
3 45     0   1  -F b31e438      sscsi007      admin
DELETE FROM ssc1
3 116    0   0  -F b362038      sscsi007      admin
SELECT COUNT ( * ) FROM ssc1
Total number of entries: 10.
```

Figure 15-99. onstat -g ssc Output

Output Description

Statement Cache Summary section

#lrus Number of least recently used queues (LRUS)

currsz Current cache size

maxsz Limit on total cache memory

Poolsize Total pool size

#hits The number of hits before insertion. This number equals the value of the STMT_CACHE_HITS configuration parameter

nolimit The value of the STMT_CACHE_NOLIMIT configuration parameter

The Statement Cache Entries section shows the entries that are fully inserted into the cache.

lru The index of lru queue to which the cache entry belongs

hash Hash values of cached entry

ref_count Number of threads referencing the statement

- hits* Number of times a statement matches a statement in the cache. The match can be for a key-only or fully cached entry.
- flag* Cache entry flag -F indicates the statement is fully cached -D indicates the statement is dropped
- heap_ptr*
 Address of memory heap for cache entry

onstat -g stk *tid*: Print thread stack

Syntax:

▶▶ onstat -g stk *tid* ◀◀

The **onstat -g stk *tid*** option prints the stack of the thread specified by thread ID.

Example Output

```
Stack for thread: 2 adminthd
base: 0x000000010aad5028
len: 33280
pc: 0x00000001002821e8
tos: 0x000000010aad621
state: running
vp: 2

0x1002821e8 oninit :: yield_processor + 0x260 sp=0x10aadce20(0x10ac834d0, 0x0, 0x1,
0x100000000, 0xc8a000, 0x100c8a000)
0x100274e38 oninit :: wake_periodic + 0xdc sp=0x10aadced0 delta_sp=176(0x41b0, 0xc7a024bc,
0x0, 0x41c4, 0x10aacf598, 0x90)
0x100274fcc oninit :: admin_thread + 0x108 sp=0x10aadcf80 delta_sp=176(0x0, 0x2328,
0xd26c00, 0x5, 0xc8a000, 0x156c)
0x1002484ec oninit :: startup + 0xd8 sp=0x10aadd050 delta_sp=208(0xa, 0x10aad47d0,
0x10aad47d0, 0x100db1988, 0xd1dc00, 0x1)
```

Figure 15-100. onstat -g stk Output

onstat -g stm: Print SQL statement memory usage

Syntax:

▶▶ onstat -g stm ◀◀

The **onstat -g stm** command displays the memory that each prepared SQL statement uses. By default, only the DBSA can view **onstat -g stm** syssqltrace information. However, when UNSECURE_ONSTAT = 1 all users can view this information. To display the memory for only one session, specify the session ID in the **onstat -g stm** option.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:26:46 -- 29696 Kbytes
session 65 -----
sdblock heapsz statement ('*' = Open cursor)
aad8028 16544 SELECT COUNT ( * ) FROM ssc1 WHERE t1_char = ?
AND t1_key = ? AND t1_short = ?

```

Figure 15-101. *onstat -g stm* Output

Output Description

sdblock Address of the statement descriptor block

heapsz Size of the statement memory heap

statement
Query text

onstat -g stq: Print queue information

Syntax:

```

▶▶ onstat -g stq session ▶▶

```

The **onstat -g stq session** option prints queue information. To view queue information for a particular session specify the *session* option. To view queue information for all sessions, do not specify the *session* option.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:18:26 -- 6
7584 Kbytes

Stream Queue: (session 25 cnt 4) 0:db12400 1:db18400 2:dcf0400 3:dcf6400
Full Queue: (cnt 2 waiters 0) 0:0 1:db12400
Empty Queue: (cnt 0 waiters 0)

```

Figure 15-102. *onstat -g stq* Output

Output Description

session Session id

cnt Number of stream queue buffers

waiters Number of threads waiting for the stream queue buffer

onstat -g sts: Print stack usage per thread

Syntax:

```

▶▶ onstat -g sts ▶▶

```

The **onstat -g sts** option prints maximum and current stack use per thread.

Example Output

```
Stack usage:
```

TID	Total	Max		Current		Thread Name
		bytes	%	bytes	%	
2	32768	3124	9	3079	9	adminthd
3	32768	2870	8	2871	8	childthd
5	32768	14871	45	2871	8	Cosvr Avail Mgr
6	32768	2870	8	2871	8	dfm_svc
7	131072	3190	2	3191	2	xmf_svc
9	32768	3126	9	3127	9	xtm_svcc
10	32768	3580	10	3335	10	xtm_svcp
11	32768	3238	9	3239	9	cfgmgr_svc
12	32768	6484	19	2871	8	lio vp 0
14	32768	6484	19	2871	8	pio vp 0
16	32768	6484	19	2871	8	aio vp 0
18	131072	10391	7	2871	2	msc vp 0
20	32768	4964	15	2871	8	fifo vp 0
22	32768	4964	15	2871	8	fifo vp 1
24	32768	6028	18	2871	8	aio vp 1
26	32768	5444	16	2951	9	dfmxml_svc
27	32768	2886	8	2887	8	sch_svc
28	32768	7812	23	5015	15	rqm_svc
29	32768	7140	21	3079	9	sm_poll
30	32768	11828	36	6439	19	sm_listen
31	32768	2870	8	2871	8	sm_discon
32	32768	14487	44	4055	12	main_loop()
33	32768	4272	13	2903	8	flush_sub(0)
34	32768	2902	8	2903	8	flush_sub(1)
35	32768	2870	8	2871	8	btscanner 0
36	32768	3238	9	3239	9	aslogflush
37	32768	3055	9	2887	8	bum_local
38	32768	3238	9	3239	9	bum_rcv
39	32768	4902	14	4903	14	onmode_mon
42	32768	4964	15	2871	8	lio vp 1
44	32768	5136	15	2871	8	pio vp 1

Figure 15-103. `onstat -g sts` Output

`onstat -g sync`: Print ER synchronization status

Syntax:

```
▶▶ onstat -g sync ◀◀
```

The `onstat -g sync` command displays the synchronization status when Enterprise Replication is used. The `onstat -g sync` option is used primarily as a debugging tool and by IBM Support.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:10:16 -- 44084 Kbytes
Prim  Sync  St.  Shadow Flag Stat Block  EndBlk
Repl  Source  Repl          Num    Num
655361 20    0    1310729 2    0    592    600

```

Figure 15-104. *onstat -g sync* Output

Output Description

Prim Repl

Replicate number of the replicate being synchronized

Sync Source

Source server of the sync

St Sync replicate state

Shadow Repl

The shadow replicate used to perform the sync

Flag Internal flags:

- 0x02 = external sync
- 0x04 = shutdown request has been issued
- 0x08 = abort has occurred
- 0x010 = a replicate stop has been requested
- 0x020 = shadow or primary replicate has been deleted

Stat Resync job state

Block num

Last block applied on targets (on source always 0)

EndBlock Num

Last block in resync process. Marks the end of the sync scan on the target. A value of -2 indicates that the scan is still in progress, and the highest block number is not yet known.

Additional fields for forwarded rows:

ServID Server where forwarded row originated

fwdLog ID

Originator's log ID of the forwarded row

fwdLog POS

Originator's log position of the forwarded row

endLog ID

Operation switches back to normal at this point

endLog POS

Operation switches back to normal at this log position

complete flag

Set to 1 after normal processing resumes for the originating source.

onstat -g tpf *tid*: Print thread profiles

Syntax:

▶▶ onstat -g tpf *tid* ◀◀

Prints thread profile for *tid*; 0 prints profiles for all threads.

Example Output

```
onstat -g tpf 945
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:21:39 -- 29696 Kbytes
Thread profiles
tid lkreqs lkw dl to lgrs isrd iswr isrw isdl isct isrb lx bfr bfw lsus lsmx seq
945 1969 0 0 0 6181 1782 2069 13 0 0 0 0 16183 7348 743580 0 6
```

Figure 15-105. onstat -g tpf Output

Output Description

<i>tid</i>	Thread ID
<i>lkreqs</i>	Lock requests
<i>lkw</i>	Lock waits
<i>dl</i>	Deadlocks
<i>to</i>	Remote deadlock timeout
<i>lgrs</i>	Log records
<i>isrd</i>	Number of reads
<i>iswr</i>	Number of writes
<i>isrw</i>	Number of rewrites
<i>isdl</i>	Number of deletes
<i>isct</i>	Number of commits
<i>isrb</i>	Number of rollbacks
<i>lx</i>	Long transactions
<i>bfr</i>	Buffer reads
<i>bfw</i>	Buffer writes
<i>lsus</i>	Log space currently used
<i>lsmx</i>	Max log space used
<i>seq</i>	Sequence scans

onstat -g ufr: Print memory pool fragments

Syntax:

▶▶ onstat -g ufr ◀◀

The **onstat -g ufr** option displays a list of the fragments that are currently in use in the specified memory pool. It requires an additional argument specifying either a pool name or session ID whose pool is to be displayed. Use **onstat -g mem** to identify the pool name and **onstat -g ses** to identify the session ID.

Memory pools are broken into fragments for various uses. With the **onstat -g ufr** command it is possible to see a list of these fragments showing their respective sizes in bytes and the type of information they contain. The information provided is generally used by IBM Support to assist in the analysis of a reported problem.

Example Output

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:56:39 -- 1067288 Kbytes

Memory usage for pool name btscanner_0:
size      memid
3256      overhead
144       scb
552       opentable
552       hashfiletab
2904      ostcb
1584      gentcb
12096     log
1912     sqtcb
```

Figure 15-106. *onstat -g ufr* Output for pool name *btscanner_0*

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:57:27 -- 1067288 Kbytes

Memory usage for pool name 6:
size      memid
3256      overhead
144       scb
2968      ostcb
18896     sqscb
3312      opentable
72        sql
808       filetable
352       fragman
552       hashfiletab
1584      gentcb
12096     log
2960      sqtcb
2928      osend
720       keys
224       rdahead
16248     temprec
```

Figure 15-107. *onstat -g ufr* Output for session ID 6

Output Description

size (decimal)

Size of the fragment in bytes

memid (string)

Name assigned to this fragment

onstat -g vpcache: Print CPU VP memory block cache statistics

Syntax:

```
▶▶ onstat -g vpcache ▶▶
```

The **onstat -g vpcache** option returns information about CPU VP memory block cache statistics.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 00:00:38 -- 18432 Kbytes
```

```
CPU VP memory block cache statistics - 4096 byte blocks
```

```
Number of 4096 byte memory blocks requested for each CPU VP:250
```

vpid	pid	Blocks held	Hit percentage	Free cache
1	7889	193	77.4 %	21.9 %

```
Current VP total allocations from cache: 0
```

size	cur blks	alloc	miss	free	drain
1	30	13	4	43	0
2	12	3	0	9	0
3	42	7	0	21	0
4	4	0	0	1	0
5	0	0	0	0	0
6	0	0	0	0	0
7	0	0	0	0	0
8	8	0	0	1	0
9	63	0	0	7	0
10	0	0	0	0	0
11	0	0	0	0	0
12	0	0	0	0	0
13	0	0	0	0	0
14	0	0	0	0	0
15	0	0	0	0	0
16	0	0	0	0	0
17	34	1	3	3	0
18	0	0	0	0	0
19	0	0	0	0	0
20	0	0	0	0	0
21	0	0	0	0	0
22	0	0	0	0	0
23	0	0	0	0	0
24	0	0	0	0	0
25	0	0	0	0	0
26	0	0	0	0	0
27	0	0	0	0	0
28	0	0	0	0	0
29	0	0	0	0	0
30	0	0	0	0	0
31	0	0	0	0	0
32	0	0	0	0	0

Figure 15-108. onstat -g vpcache Output

Output Description

You can interpret output from `onstat -g vpcache` as follows:

- size** Is the size of the memory blocks in 4096 byte blocks
- cur blks**
Is the current number of 4096 blocks, a multiple of *size*
- alloc** Is the number of times a requestor received a block of this size
- miss** Is the number of times a block was requested but none were available
- free** Is the number of times a memory block was placed into the cache
- drain** Is the number of times an aged block was forced out to make room for another block

onstat -g wai: Print wait queue thread list

Syntax:

```
▶▶ onstat -g wai ◀◀
```

The `onstat -g wai` option displays a list of the threads in the system that are currently in the wait queue and not currently executing. The output is sorted by thread ID.

Example Output

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 02:26:15 -- 1067288 Kbytes

Waiting threads:
tid   tcb           rstcb      prty status                vp-class  name
 2    46b1ea40      0          1    IO Idle                5lio     lio vp 0
 3    46b3dc58      0          1    IO Idle                6pio     pio vp 0
 4    46b5dc58      0          1    IO Idle                7aio     aio vp 0
 5    46b7cc58      0          1    IO Idle                8msc     msc vp 0
 6    46b1ed10      460f5028   1    sleeping secs: 1       3cpu     main_loop()
 9    46d0d6e0      0          1    sleeping forever      1cpu     soctcplst
10    46d70b48      0          1    sleeping forever      3cpu     sm_listen
11    46e5d9a0      0          1    sleeping secs: 1       3cpu     sm_discon
12    46e5dc70      460f5820   1    sleeping secs: 1       3cpu     flush_sub(0)
13    46e8a5a8      460f6018   1    sleeping secs: 1       3cpu     aslogflush
14    46fe8148      460f6810   1    sleeping secs: 41      3cpu     btscanner_0
15    46fe84a8      0          1    IO Idle                10aio    aio vp 1
16    46fe8778      460f7008   1    sleeping secs: 1       1cpu     onmode_mon
36    47531960      460f7ff8   1    sleeping secs: 253     3cpu     dbScheduler
37    47531c30      460f87f0   1    sleeping forever      4cpu     dbWorker1
38    47491028      460f7800   1    sleeping forever      4cpu     dbWorker2
```

Figure 15-109. `onstat -g wai` Output

Output Description

tid (decimal)

Thread ID

tcb (hex)

In-memory address of the thread control block

rstcb (hex)

In-memory address of the RSAM thread control block

prty (decimal)

Thread priority. Higher numbers represent higher priorities

status (string)

Current status of the thread

vp-class (decimal and string)

Virtual processor integer ID of the VP on which the thread last ran, concatenated with the name of the VP class upon which the thread runs

name (string)

Name of the thread

onstat -g wmx: Print all mutexes with waiters

Syntax:

```
▶▶ onstat -g wmx ◀◀
```

The **onstat -g wmx** option prints all mutexes with waiters.

Example Output

```
Mutexes with waiters:
mid      addr          name          holder  lkcnt  waiter  waittime
```

Figure 15-110. *onstat -g wmx* Output

onstat -g wst: Print wait statistics for threads

Syntax:

```
▶▶ onstat -g wst ◀◀
```

The **onstat -g wst** option prints wait statistics for threads within the system. The WSTATS configuration parameter must be set to 1 to enable wait statistics collection. For more information, see “WSTATS” on page 1-114.

Example Output

The following sample output is from the command **onstat -g wst**:

```
IBM Informix Dynamic Server Version 11.50.F -- On-Line -- Up 01:47:11
-- 1067288 Kbytes
```

name	tid	state	n	avg(us)	max(us)
lio vp 0 2		ready	128	17	454819
lio vp 0 2		run	127	405	441
lio vp 0 2		IO Idle	126	643203	727160
pio vp 0 3		yield 0	1	38	38
pio vp 0 3		ready	2	22	43
pio vp 0 3		run	1	16	16

aio vp 0 4	yield 0	1	37	37
aio vp 0 4	yield time	2	747701	984462
aio vp 0 4	ready	230	129	5284
aio vp 0 4	run	229	145	10045
aio vp 0 4	IO Idle	226	45823	941363
msc vp 0 5	yield 0	1	38	38
msc vp 0 5	ready	5	280	1273
msc vp 0 5	run	4	178	429
msc vp 0 5	IO Idle	3	896605	1.0s
main_loo 6	IO Wait	26	10274	12113
main_loo 6	yield time	6416	1.0s	1.0s
main_loo 6	yield forever	4	97377	105682
main_loo 6	ready	6450	23	31864
main_loo 6	run	6436	5	3500
soctcpo 7	yield forever	1027128	3	1.0s
soctcpo 7	other cond	1	110728	110728
soctcpo 7	ready	2	177208	1.3s
soctcpo 7	run	1027127	118377	1.0s
sm_poll 8	yield 0	1	61	61
sm_poll 8	yield time	1	887246	887246
sm_poll 8	ready	3	30	69
sm_poll 8	run	1	30	30
soctcpls 9	IO Wait	5	781	1580
soctcpls 9	ready	7	14	54
soctcpls 9	run	5	267	695
sm_liste 10	IO Wait	8	168	718
sm_liste 10	ready	9	93	629
sm_liste 10	run	8	99	561
sm_disco 11	yield time	6417	1.0s	1.0s
sm_disco 11	ready	6418	38	38860
sm_disco 11	run	6417	2	7
flush_su 12	yield time	6417	1.0s	1.1s
flush_su 12	ready	6418	38	38901
flush_su 12	run	6417	2	7
aslogflu 13	yield time	6416	1.0s	1.0s
aslogflu 13	ready	6418	33	38824
aslogflu 13	run	6417	2	8
btscanne 14	yield 0	1	7	7
btscanne 14	yield time	72	498264	623090
btscanne 14	ready	222	765502	1.0s
btscanne 14	run	73	123	653
onmode_m 25	yield time	6414	1.0s	1.0s
onmode_m 25	ready	6416	29	38816
onmode_m 25	run	6414	4	19
aio vp 1 30	yield 0	1	37	37
aio vp 1 30	ready	143	11	278
aio vp 1 30	run	142	11	142
aio vp 1 30	IO Idle	141	45023	779089
bf_prios 31	ready	1	0	0
dbSchedu 32	yield bufwait	11	35	158
dbSchedu 32	IO Wait	151	134	4231
dbSchedu 32	yield 0	74	211	455

dbSchedu	32	yield time	50	109362	368997
dbSchedu	32	logio cond	13	1865	3304
dbSchedu	32	ready	323	310	4728
dbSchedu	32	run	298	203	922
dbWorker	33	yield bufwait	12	126	749
dbWorker	33	IO Wait	170	326	6492
dbWorker	33	yield 0	79	198	4012
dbWorker	33	yield forever	17	484	733
dbWorker	33	logio cond	19	796	3305
dbWorker	33	ready	330	196	4114
dbWorker	33	run	298	12821	18228
dbWorker	34	yield bufwait	8	2008	4314
dbWorker	34	IO Wait	82	4397	6747
dbWorker	34	yield 0	66	208	2411
dbWorker	34	yield forever	18	64320	728046
dbWorker	34	logio cond	18	1208	4658
dbWorker	34	other mutex	1	591	591
dbWorker	34	ready	203	389	3682
dbWorker	34	run	193	378	3566

Output Description

name (string)

Thread name

tid (decimal)

Thread ID

state (string)

State the thread waited in for this line of output. A single thread may have multiple lines of output if it has waited in more than one different state.

n (decimal)

Number of times the thread waited in this state

avg(us) (floating point)

Average user time the thread spent waiting in this state per wait occurrence. Time is in microseconds; an *s* after the value indicates user time in seconds.

max(us) (floating point)

Maximum user time the thread spent waiting in this state for a single wait occurrence. Time is in microseconds; an *s* after the value indicates user time in seconds.

onstat -G: Print TP/XA transaction information

Syntax:

▶▶ onstat -G ◀◀

Use the **-G** option to display information about global transactions generated through TP/XA. For more information on TP/XA, see the *IBM Informix TP/XA Programmer's Manual*.

Example Output

Figure 15-111 on page 15-137 shows an example of **onstat -G** output:

```

Global Transaction Identifiers
address  flags  isol  timeout  fID      gtl  bql  data
ae35e34  -LR-G  COMMIT  0        4478019  16  48  438709F23076254C80F33A62B
AF4CF763C1BCFFAD7AE0243AA5CE243FA5381C903AA9F52A1546044992C5A7BC03582E77999EFBA7
25D3D40BDAF37404D9DAFF1
ae3623c  AL--G  COMMIT  0        4478019  16  48  438709F23076254C80F33A62B
AF4CF763C1BCFFAD7AE0243AA5CE243FA5381C903AA9F52A1546044992C5A7BC03582E77999EFBA7
25D3D40BDAF37404D9DA000
  2 active, 128 total

```

Figure 15-111. `onstat -G` Output

For a tightly coupled transaction, all branches will share the same transaction address shown in the address column.

Output Description

address

Transaction address

flags

The flag codes for position 1 (current transaction state):

- A** User thread attached to the transaction
- S** TP/XA suspended transaction
- C** TP/XA waiting for rollback

The flag codes for position 2 (transaction mode):

- T** Tightly-coupled mode (MTS)
- L** Loosely-coupled mode (default mode)

The flag codes for position 3 (transaction stage):

- B** Begin work
- P** Distributed query prepared for commit
- X** TP/XA prepared for commit
- C** Committing or committed
- R** Rolling back or rolled back
- H** Heuristically rolling back or rolled back

The flag code for position 4:

- X** XA DataSource global transaction

The flag codes for position 5 (type of transaction):

- G** Global transaction
- C** Distributed query coordinator
- S** Distributed query subordinate
- B** Both distributed query coordinator and subordinate

isol Transaction isolation level

timeout

Transaction lock timeout

fID Format ID

gtl Global transaction ID length
bql Branch qualifier length
data Transaction-specific data

onstat -h: Print buffer header hash chain information

Syntax:

▶▶ onstat -h ◀◀

The **onstat -h** option provides information on the buffer header hash chains (sometimes called "hash buckets") used to access pages in each buffer pool, in the form of a numeric histogram of chain lengths, plus summary information for each buffer pool. All numeric values in the output are decimal. Shorter hash chains enable requested buffers to be located more quickly by the server, because on average it will need to check fewer buffer headers on a target chain to find the target buffer.

The page size of the buffer pool in bytes is shown as a header to the output for each buffer pool. The histogram and summary information are then presented for that buffer pool.

Example Output

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 00:00:14 -- 1071740 Kbytes
Buffer pool page size: 2048
buffer hash chain length histogram
# of chains      of len
   3423           0
   4546           1
    223           2
   8192 total chains
   4992 hashed buffs
   5000 total buffs
Buffer pool page size: 4096
buffer hash chain length histogram
# of chains      of len
    707           0
    315           1
     2            2
   1024 total chains
    319 hashed buffs
   1000 total buffs
```

Figure 15-112. onstat -h Output

Output Description

You can interpret output from this option as follows:

Histogram Information on Hash Chains

The histogram information has a row for each buffer hash chain length that presently exists in the system. Each row has two columns:

of chains

Number of hash chains of the given length

of len Length of these chains

Summary Information Per Buffer Pool

total chains

Number of hash chains that exist for this buffer pool

hashed buffs

Number of buffer headers currently hashed into the hash chains for this buffer pool

total buffs

Total number of buffers in this buffer pool

onstat -i: Initiate interactive mode

Syntax:

```
▶▶ onstat -i [r seconds] [rz seconds] ▶▶
```

Use the **-i** option to put **onstat** in interactive mode. In interactive mode, you can enter multiple **onstat** options per session, but only one at a time. An **onstat** prompt appears and allows you to enter an option.

In interactive mode, do not precede the option with a dash.

Two additional options, **r seconds** and **rz seconds**, are available in interactive mode. The **r seconds** option is similar to the current **onstat -r seconds** option, which repeatedly generates a display. If an administrator executes **r seconds** at the interactive-mode prompt, the prompt changes to reflect the specified interval in seconds and reappears, waiting for the next command. In the following example, the display generated by the next command repeats every three seconds:

```
onstat> r 3  
onstat[3]>
```

The **rz seconds** option enables you to repeat the next command as specified and set all profile counters to 0 between each execution.

To terminate interactive mode, press CTRL-d.

To terminate a repeating sequence, press CTRL-c.

onstat -j: Provide onpload status information

Syntax:

```
▶▶ onstat -j ▶▶
```

The **-j** option of the **onstat** utility provides special information about the status of an **onpload** job. The **-j** option provides an interactive mode that is analogous to **onstat -i**.

When **onpload** starts, it writes a series of messages to **stdout** or to a log file. The following lines show a typical **onpload** log file:

```
Mon Jul 23 16:11:30 2007

SHMBASE      0x4400000
CLIENTNUM    0x49010000
Session ID 1

Load Database -> cnv001
Load Table   -> cnv001a
Load File    -> testrec.dat
Record Mapping -> cnv001a

Database Load Completed -- Processed 50 Records
Records Inserted-> 50
Detected Errors--> 0
Engine Rejected--> 0

Mon Jul 23 16:11:37 2007
```

The two lines that start with **SHMBASE** and **CLIENTNUM** provide the information that you need to locate shared memory for an instance of **onpload**. The **oninit** process has similar values stored in the **\$ONCONFIG** file. When you use **onstat** to gather information about the **oninit** process, **onstat** uses information from **\$INFORMIXDIR/etc/\$ONCONFIG** to locate shared memory. When you use **onstat** to gather information about **onpload**, you must give **onstat** the name of a file that contains **SHMBASE** and **CLIENTNUM** information.

Typically the file that contains the **SHMBASE** and **CLIENTNUM** information is the log file. For example, if the **onpload** log file is **/tmp/cnv001a.log**, you can enter the following command:

```
onstat -j /tmp/cnv001a.log
```

The previous command causes **onstat** to attach to **onpload** shared memory and to enter interactive mode. You can then enter **?** or any other pseudo request to see a usage message displayed. An example follows:

```
onstat> ?
Interactive Mode: One command per line, and - are optional.
  -rz   repeat option every n seconds (default: 5) and
        zero profile counts
MT COMMANDS:
all    Print all MT information
ath    Print all threads
wai    Print waiting threads
act    Print active threads
rea    Print ready threads
sle    Print all sleeping threads
spi    print spin locks with long spins
sch    print VP scheduler statistics
lmx    Print all locked mutexes
wmx    Print all mutexes with waiters
con    Print conditions with waiters
stk <tid>  Dump the stack of a specified thread
glo    Print MT global information
mem <pool name|session id>  print pool statistics.
seg    Print memory segment statistics.
rbm    print block map for resident segment
nbm    print block map for non-resident segments
```

```

afr <pool name|session id> Print allocated pool fragments.
ffr <pool name|session id> Print free pool fragments.
ufr <pool name|session id> Print pool usage breakdown
ioy Print disk IO statistics by vp
iof Print disk IO statistics by chunk/file
ioq Print disk IO statistics by queue
iog Print AIO global information
iob Print big buffer usage by IO VP class
sts Print max and current stack sizes
qst print queue statistics
wst print thread wait statistics
jal Print all Pload information
jct Print Pload control table
jpa Print Pload program arguments
jta Print Pload thread array
jmq Print Pload message queues, jms for summary only
onstat>

```

Most of the options are the same as those that you use to gather information about Dynamic Server, with the following exceptions:

```

jal Print all Pload information
jct Print Pload control table
jpa Print Pload program arguments
jta Print Pload thread array
jmq Print Pload message queues, jms for summary only

```

These options apply only to **onpload**. You can use **onstat -j** to check the status of a thread, locate the VP and its PID, and then attach a debugger to a particular thread. The options for **onstat** that do not apply to **onpload** are not available (for example, `-g ses`).

onstat -k: Print active lock information

Syntax:

```
▶▶ onstat -k ◀◀
```

Use the `-k` option to display information about active locks.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:55:17 -- 15360 Kbytes

Locks
address  wtlst  owner  lklist  type  tblsnum  rowid  key#/bsiz
a095f78  0      a4d9e68  0      HDR+S  100002  203    0
1 active, 2000 total, 2048 hash buckets, 0 lock table overflows

```

Figure 15-113. `onstat -k` Output

Output Description

You can interpret output from this option as follows:

address Is the address of the lock in the lock table

If a user thread is waiting for this lock, the address of the lock appears in the **wait** field of the **onstat -u** (users) output.

wtlist Is the first entry in the list of user threads that is waiting for the lock, if there is one

owner Is the shared-memory address of the thread that is holding the lock
This address corresponds to the address in the **address** field of **onstat -u** (users) output. When the *owner* value is displayed in parenthesis, it represents the shared memory address of a transaction structure. This scenario is possible only when a lock is allocated for a global transaction. This address corresponds to the address field of the output for **onstat -G**.

lklist Is the next lock in a linked list of locks held by the owner just listed

type Uses the following codes to indicate the type of lock:

HDR	Header
B	Bytes
S	Shared
X	Exclusive
I	Intent
U	Update
IX	Intent-exclusive
IS	Intent-shared
SIX	Shared, intent-exclusive

tblsnum Is the tblspace number of the locked resource. If the number is less than 10000, it indicates Enterprise Replication pseudo locks.

rowid Is the row identification number

The rowid provides the following lock information:

- If the rowid equals zero, the lock is a table lock.
- If the rowid ends in two zeros, the lock is a page lock.
- If the rowid is six digits or fewer and does not end in zero, the lock is probably a row lock.
- If the rowid is more than six digits, the lock is probably an index key-value lock.

key#/bsiz

Is the index key number, or the number of bytes locked for a VARCHAR lock

If this field contains 'K-' followed by a value, it is a key lock. The value identifies which index is being locked. For example, K-1 indicates a lock on the first index defined for the table.

The maximum number of locks available is specified as LOCKS in the ONCONFIG file.

onstat -l: Print physical and logical log information

Syntax:

Use the **-l** option to display information about physical and logical logs.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:55:32 -- 15360 Kbytes

Physical Logging
Buffer bufused  bufsize  numpages numwrits pages/io
P-1  0           16       716      55       13.02
      phybegin      physize  phypos   phyused  %used
      1:263         500     270     0        0.00

Logical Logging
Buffer bufused  bufsize  numrecs  numpages numwrits  recs/pages  pages/io
L-3  0           16       42169   2872    1043     14.7      2.8
      Subsystem    numrecs  Log Space used
      OLDRSAM      42169   4436496

address  number  flags   uniqid  begin          size  used  %used
a517f70  1       U-B----  1       1:763          500  500  100.00
a517fb0  2       U-B----  2       1:1263         500  500  100.00
a40daf0  3       U-B----  3       1:1763         500  500  100.00
a40db30  4       U-B----  4       1:2263         500  500  100.00
a40db70  5       U-B----  5       1:2763         500  500  100.00
a40dbb0  6       U---C-L  6       1:3263         500  372  74.40
a40dbf0  7       A-----  0       1:3763         500  0    0.00
a40dc30  8       A-----  0       1:4263         500  0    0.00
8 active, 8 total
    
```

Figure 15-114. onstat -l Output

Output Description

You can interpret output from this option as follows. The first section of the display describes the physical-log configuration:

buffer Is the number of the physical-log buffer

bufused

Is the number of pages of the physical-log buffer that are used

bufsize Is the size of each physical-log buffer in pages

numpages

Is the number of pages written to the physical log

numwrits

Is the number of writes to disk

pages/io

Is calculated as $numpages/numwrits$

This value indicates how effectively physical-log writes are being buffered.

phybegin

Is the physical page number of the beginning of the log

physize Is the size of the physical log in pages

phypos Is the current position in the log where the next log-record write is to occur

phyused
Is the number of pages used in the log

%used Is the percent of pages used

The second section of the **onstat -l** display describes the logical-log configuration:

buffer Is the number of the logical-log buffer

bufused
Is the number of pages used in the logical-log buffer

bufsize Is the size of each logical-log buffer in pages

numrecs
Is the number of records written

numpages
Is the number of pages written

numwrits
Is the number of writes to the logical log

recs/pages
Is calculated as $\text{numrecs}/\text{numpages}$

You cannot affect this value. Different types of operations generate different types (and sizes) of records.

pages/io
is calculated as $\text{numpages}/\text{numwrits}$

You can affect this value by changing the size of the logical-log buffer (specified as LOGBUFF in the ONCONFIG file) or by changing the logging mode of the database (from buffered to unbuffered, or vice versa).

The following fields are repeated for each logical-log file:

address Is the address of the log-file descriptor

number
Is logid number for the logical-log file

The logid numbers might be out of sequence because either the database server or administrator can insert a log file in-line.

flags Provides the status of each log as follows:

A Newly added (and ready to use)

B Backed up

C Current logical-log file

D Marked for deletion

To drop the log file and free its space for reuse, you must perform a level-0 backup of all storage spaces

F Free, available for use

L The most recent checkpoint record

U Used

uniqid Is the unique ID number of the log
begin Is the beginning page of the log file
size Is the size of the log in pages
used Is the number of pages used
%used Is the percent of pages used
active Is the number of active logical logs
total Is the total number of logical logs

The database server uses *temporary logical logs* during a warm restore because the permanent logs are not available then. The following fields are repeated for each temporary logical-log file:

address Is the address of the log-file descriptor

number

Is logid number for the logical-log file

flags Provides the status of each log as follows:

B Backed up
C Current logical-log file
F Free, available for use
U Used

uniqid Is the unique ID number of the log

begin Is the beginning page of the log file

size Is the size of the log in pages

used Is the number of pages used

%used Is the percent of pages used

active Is the number of active temporary logical logs

onstat -m: Print recent system message log information

Syntax:

▶▶ onstat -m ◀◀

Use the **-m** option to display the 20 most-recent lines of the system message log. You can use the **onstat -m** option with the database server in any mode, including offline.

Output from this option lists the full pathname of the message-log file and the 20 file entries. A date-and-time header separates the entries for each day. A time stamp prefaces single entries within each day. The name of the message log is specified as MSGPATH in the **ONCONFIG** file.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:55:41 -- 15360 Kbytes
Message Log File: /work/11.50/dbspaces/star3.log
11:26:33 Checkpoint Completed: duration was 0 seconds.
11:26:33 Checkpoint loguniq 1, logpos 0x23c408, timestamp: 0x2cc2 Interval: 9

```

Figure 15-115. `onstat -m` Output

onstat -o: Output shared memory contents to a file

Syntax:

```

▶▶ onstat -o [nobuffs] [full] outfile

```

Use the **onstat -o** option to write the contents of shared memory to a specified file for later analysis. If you do not specify an output file, a file named **onstat.out** is created in the current directory.

If you use the **nobuffs** option, the shared memory dump excludes the buffer pool in the resident segment of shared memory, resulting in a smaller dump file.

If you use the **full** option, the file created by **onstat -o** is the same size as the shared memory segments for the Informix Dynamic Server instance. You must have enough room in the file system to handle the output.

If you do not specify either the **nobuffs** or the **full** option, the output is controlled by the database server DUMPSHMEM configuration parameter setting:

- If DUMPSHMEM is set to 0 or to 1, **onstat -o** writes a full shared-memory dump file.
- If DUMPSHMEM is set to 2, **onstat -o** writes a **nobuffs** shared-memory dump file that excludes the buffer pool in the resident segment.

By executing additional **onstat** commands against the file, you can gather information from a previously saved shared memory dump. The *outfile* that you create with **onstat -o** is the *infile* that you can use as a source file to run the additional **onstat** commands. For more information, see “Running onstat Commands on a Shared Memory Dump File” on page 15-8.

onstat -O: Print optical subsystem information

Syntax:

```

▶▶ onstat -O

```

Use the **-O** option of the **onstat** utility to display information about the Optical Subsystem memory cache and staging-area blobspace. You can interpret output from this option as follows. The totals shown in the display accumulate from session to session. The database server resets the totals to 0 only when you execute **onstat -z**.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 --Online-- Up 00:45:18 -- 11656 Kbytes

Optical StageBlob Cache
System Cache Totals:
Size Alloc. Avail.          Number Kbytes System Blob Totals:
500   500    0              1     20      3     1500

User Cache Totals:
SID  User  Size          Number Kbytes User Blob Totals:
94   doug  250           1     20      1     300
95   beth  500           0     0       2    1200
```

Figure 15-116. *onstat -O* Output

Output Description

The first section of the display provides the following information on system-cache totals:

size Is the size that the OPCACHEMAX configuration parameter specifies

alloc Is the number of 1-kilobyte allocations to the cache

avail Describes how much of **alloc** (in kilobytes) is not used

number

Is the number of simple large objects that the database server successfully put in the cache without overflowing

kbytes Is the number of kilobytes of TEXT or BYTE data that the database server put in the cache without overflowing

number

Is the number of simple large objects that the database server wrote to the staging-area blob space

kbytes Is the number of kilobytes of TEXT or BYTE data that the database server wrote to the staging-area blob space

Although the **size** output indicates the amount of memory that is specified in the configuration parameter OPCACHEMAX, the database server does not allocate memory to OPCACHEMAX until necessary. Therefore, the **alloc** output reflects only the number of 1-kilobyte allocations of the largest simple large object that has been processed. When the values in the **alloc** and **avail** output are equal to each other, the cache is empty.

The second section of the display describes the following user-cache totals information:

SID Is the session ID for the user

user Is the user ID of the client

size Is the size specified in the **INFORMIXOPCACHE** environment variable, if it is set

If you do not set the **INFORMIXOPCACHE** environment variable, the database server uses the size that you specify in the configuration parameter OPCACHEMAX.

number

Is the number of simple large objects that the database server put into cache without overflowing

kbytes

Is the number of kilobytes of TEXT or BYTE data that the database server put in the cache without overflowing

number

Is the number of simple large objects that the database server wrote to the staging-area blobspace

kbytes

Is the number of kilobytes of TEXT or BYTE data that the database server wrote to the staging-area blobspace

The last line of the display lists the total number of sessions that are using the cache.

onstat -p: Print profile counts

Syntax:

▶▶ onstat -p ◀◀

Use the **-p** option to display profile counts either since you started the database server or since you ran **onstat** with the **-z** option.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:56:40 -- 15360 Kbytes

Profile
dskreads pagreads bufreads %cached dskwrits pagwrits bufwrits %cached
939      943      143905  99.35  3925     10816   46919  91.63

isamtot  open    start   read    write   rewrite delete  commit  rollbk
100055  15851  16112  24632  13343  1342   1392   905    0

gp_read  gp_write gp_rewrt gp_del  gp_alloc gp_free  gp_curs
0        0        0        0       0        0        0

ovlock   ovuserth ovbuff   usercpu  syscpu   numckpts flushes
0        0        0        12.00   2.69    9        101

bufwaits lokwaits lockreqs deadlks  dltouts  ckpwaits compress seqscans
8        0        26894   0        0        1        1247   478

ixda-RA  idx-RA   da-RA    RA-pgsused lchwaits
5        0        10       15        23
```

Figure 15-117. onstat -p Output

Output Description

The first portion of the display describes reads and writes.

Reads and writes are tabulated in three categories: from disk, from buffers, and number of pages (read or written).

The first **%cached** field is a measure of the number of reads from buffers compared to reads from disk. The second **%cached** field is a measure of the number of writes to buffers compared to writes to disk.

The database server buffers information and writes to the disk in pages. For this reason, the number of disk writes displayed as `dskwrits` is usually less than the number of writes that an individual user executes:

dskreads

Is the number of actual reads from disk

pagreads

Is the number of pages read

bufreads

Is the number of reads from shared memory

%cached

Is the percent of reads cached, calculated as follows:

$100 * (\text{bufreads} - \text{dskreads}) / \text{bufreads}$

If `bufreads` exceeds the maximum integer (or long) value, its internal representation becomes a negative number, but the value appears as 0.0.

dskwrits

Is the actual number of physical writes to disk

This number includes the writes for the physical and logical logs reported in **onstat -l**.

pagwrits

Is the number of pages written

bufwrits

Is the number of writes to shared memory

%cached

Is the percent of writes cached, calculated as follows:

$100 * (\text{bufwrits} - \text{dskwrits}) / \text{bufwrits}$

If `dskwrits` exceeds `bufwrits`, the value appears as 0.0. The next portion of the **-p** display tabulates the number of times different ISAM calls were executed. The calls occur at the lowest level of operation and do not necessarily correspond one-to-one with SQL statement execution. A single query might generate multiple ISAM calls. These statistics are gathered across the database server and cannot be used to monitor activity on a single database unless only one database is active or only one database exists:

isamtot Is the total number of calls

open Increments when a `tblspace` is opened

start Increments the pointer within an index

read Increments when the read function is called

write Increments with each write call

rewrite Increments when an update occurs

delete Increments when a row is deleted

commit Increments each time that an **iscommit()** call is made

No one-to-one correspondence exists between this value and the number of explicit COMMIT WORK statements that are executed.

rollbk Increments when a transaction is rolled back

The next portion of the **-p** display provides information on generic pages. The Generic Page Manager provides an API for Dynamic Server to manage nonstandard pages in the database server buffer pool. The following table describes the Generic Page Manager fields in the **onstat -p** output.

gp_read The number of generic page reads

gp_write The number of generic page writes

gp_rewrt The number of generic page updates

gp_del The number of generic page deletes

gp_alloc The number of generic page allocations

gp_free The number of generic pages freed and returned to tblspaces

gp_curs The number of cursors used against generic pages

The next portion of the **-p** display tracks the number of times that a resource was requested when none was available:

ovlock Is the number of times that the database server attempted to allocate locks more than 15 times

For more information, see "LOCKS" on page 1-56.

ovuserthread Is the number of times that a user attempted to exceed the maximum number of user threads

ovbuff Is the number of times that the database server could not find a free shared-memory buffer

When no buffers are free, the database server writes a dirty buffer to disk and then tries to find a free buffer.

usercpu Is the total user CPU time that all user threads use, expressed in seconds
This entry is updated every 15 seconds.

syscpu Is the total system CPU time that all user threads use, expressed in seconds
This entry is updated every 15 seconds.

numckpts Is the number of checkpoints since the boot time

flushes Is the number of times that the buffer pool has been flushed to the disk

The next portion of the **-p** display contains miscellaneous information, as follows:

bufwaits Increments each time that a user thread must wait for a buffer

lokwaits Increments each time that a user thread must wait for a lock

lockreqs Increments each time that a lock is requested

deadlks Increments each time that a potential deadlock is detected and prevented

dltouts Increments each time that the distributed deadlock time-out value is exceeded while a user thread is waiting for a lock

ckpwaits Is the number of checkpoint waits

compress Increments each time that a data page is compressed

seqscans Increments for each sequential scan

The last portion of the **-p** display contains the following information:

ixda-RA Is the count of read-aheads that go from index leaves to data pages

idx-RA Is the count of read-aheads that traverse index leaves

da-RA Is the count of data-path-only scans

RA-pgsused Indicates the number of pages used that the database server read ahead
 If this number is significantly less than the total number of pages read ahead, the read-ahead parameters might be set too high.

lchwaits Stores the number of times that a thread was required to wait for a shared-memory latch
 A large number of latch waits typically results from a high volume of processing activity in which the database server is logging most of the transactions.

onstat -P: Print partition information

Syntax:

▶▶ onstat -P ◀◀

Use the **-P** option to display the partition number and the pages in the buffer pool for all partitions.

For information about running **onstat -P** on a dump file created without the buffer pool, see “Running onstat Commands on a Shared Memory Dump File” on page 15-8.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 18:44:15 -- 34816 Kbytes
Buffer pool page size: 2048
partnum total    btree    data    other    dirty
0         36      1         8       27       0
1048577  2         0         0        2       0
1048578  4         1         1        2       0
1048579  23        10        12       1       0
1048580  68        31        36       1       0
4194309  3         0         1        2       0

Totals: 3000    786    1779    435    0
Percentages:
Data 59.30
Btree 26.20
Other 14.50

Buffer pool page size: 8192
partnum total    btree    data    other    dirty
0         999     0         0       999     0
5242881  1         0         0        1       0

Totals: 1000    0         0       1000    0
Percentages:
Data 0.00
Btree 0.00
Other 100.00

```

Figure 15-118. `onstat -P` Output

Output Description

Buffer pool page size

Is the size of the buffer pool pages in bytes

partnum

Is the partition number

total

Is the total number of partitions

btree

Is the number of B-tree pages in the partition

data

Is the number of data pages in the partition

other

Is the number of other pages in the partition

resident

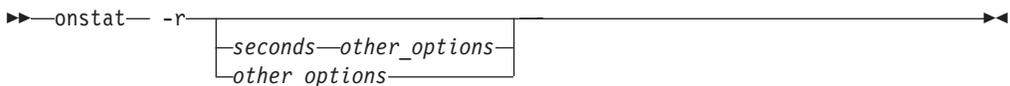
Is the number of resident pages in the partition

dirty

Is the number of dirty pages in the partition

onstat -r: Repeatedly print selected statistics

Syntax:



The **onstat -r** option causes the other options specified in the command to be repeated at specified intervals. Use **onstat -r [seconds] [other_options]** to specify the seconds to repeat the other option. Use **onstat -r [other_options]** to have the option repeat every five seconds, which allows the other options to be

concatenated with the `-r` flag, as in this example: `onstat -rFh`. The `onstat -r` option can be used in both command mode and interactive mode, and can be useful for repeating command output to monitor system resource utilization.

Example Output 1: execute 'onstat' every five seconds

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:05:25 -- 1067288 Kbytes
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:05:30 -- 1067288 Kbytes
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:05:35 -- 1067288 Kbytes
```

Figure 15-119. `onstat -r` Output

Example Output 2: execute 'onstat' every ten seconds

```
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:06:58 -- 1067288 Kbytes
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:07:08 -- 1067288 Kbytes
IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:07:18 -- 1067288 Kbytes
```

Figure 15-120. `onstat -r 10` Output

Example Output 3: execute 'onstat -h' every one second

```
IBM Informix Dynamic Server Version 11.50.F -- On-Line -- Up 20:10:28 -- 1067288 Kbytes
```

```
Buffer pool page size: 2048
```

```
buffer hash chain length histogram
```

# of chains	of len
3841	0
3767	1
522	2
62	3
8192	total chains
4351	hashed buffs
5000	total buffs

```
IBM Informix Dynamic Server Version 11.50.F -- On-Line -- Up 20:10:29 -- 1067288 Kbytes
```

```
Buffer pool page size: 2048
```

```
buffer hash chain length histogram
```

# of chains	of len
4020	0
3392	1
735	2
43	3
2	4
8192	total chains
4172	hashed buffs
5000	total buffs

Figure 15-121. `onstat -r 1 -h` Output

Example Output 4: execute 'onstat -Fh' every five seconds

```

IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:12:13 -- 1067288 Kbytes

Fg Writes      LRU Writes      Chunk Writes
0              0                21

address        flusher  state  data  # LRU  Chunk  Wakeups  Idle Tim
460e6820      0        I    0     0     2     5        9.820
      states: Exit Idle Chunk Lru

Buffer pool page size: 2048

buffer hash chain length histogram
# of chains      of len
  6342           0
  1850           1
  8192 total chains
  1850 hashed buffs
  5000 total buffs

IBM Informix Dynamic Server Version 11.50.F      -- On-Line -- Up 20:12:18 -- 1067288 Kbytes

Fg Writes      LRU Writes      Chunk Writes
0              0                21

address        flusher  state  data  # LRU  Chunk  Wakeups  Idle Tim
460e6820      0        I    0     0     2    10       22.755
      states: Exit Idle Chunk Lru

Buffer pool page size: 2048

buffer hash chain length histogram
# of chains      of len
  4396           0
  3796           1
  8192 total chains
  3796 hashed buffs
  5000 total buffs

```

Figure 15-122. **onstat -rFh** Output

onstat -R: Print LRU, FLRU, and MLRU queue information

Syntax:

►► onstat -R ◀◀

Use the **-R** option to display detailed information about the LRU queues, FLRU queues, and MLRU queues. For an in-depth discussion of the three types of queues, see LRU queues in the shared-memory chapter of the *IBM Informix Administrator's Guide*.

For each queue, **onstat -R** lists the number of buffers in the queue and the number and percentage of buffers that have been modified.

Example Output

```

IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 18:46:59 -- 34816 Kbytes
  Buffer pool page size: 2048
    8 buffer LRU queue pairs
# f/m pair total % of length priority levels
0 f 375 100.0% 375 375 0
1 m 0 0.0% 0 0 0
2 f 375 100.0% 375 375 0
3 m 0 0.0% 0 0 0
4 f 375 100.0% 375 375 0
5 m 0 0.0% 0 0 0
6 F 375 100.0% 375 375 0
7 m 0 0.0% 0 0 0
8 f 375 100.0% 375 375 0
9 m 0 0.0% 0 0 0
10 f 375 100.0% 375 375 0
11 m 0 0.0% 0 0 0
12 f 375 100.0% 375 375 0
13 m 0 0.0% 0 0 0
14 f 375 100.0% 375 375 0
15 m 0 0.0% 0 0 0
0 dirty, 3000 queued, 3000 total, 4096 hash buckets, 2048 buffer size
start clean at 60.000% (of pair total) dirty, or 226 buffs dirty, stop at
50.000%
  Buffer pool page size: 8192
    4 buffer LRU queue pairs
# f/m pair total % of length priority levels
0 F 250 100.0% 250 250 0
1 m 0 0.0% 0 0 0
2 f 250 100.0% 250 250 0
3 m 0 0.0% 0 0 0
4 f 250 100.0% 250 250 0
5 m 0 0.0% 0 0 0
6 f 250 100.0% 250 250 0
7 m 0 0.0% 0 0 0
0 dirty, 1000 queued, 1000 total, 1024 hash buckets, 8192 buffer size
start clean at 60.000% (of pair total) dirty, or 150 buffs dirty, stop at
50.000%

```

Figure 15-123. *onstat -R* Output

Output Description

You can interpret output from this option as follows:

Buffer pool page size

Is the page size of the buffer pool in bytes

Shows the queue number

Each LRU queue is composed of two subqueues: an FLRU queue and a MLRU queue. (For a definition of FLRU and MLRU queues, see LRU queues in the shared-memory chapter of the *IBM Informix Administrator's Guide*.) Thus, queues 0 and 1 belong to the first LRU queue, queues 2 and 3 belong to the second LRU queue, and so on.

f/m Identifies queue type

This field has four possible values:

f Free LRU queue

In this context, free means not modified. Although nearly all the buffers in an LRU queue are available for use, the database server attempts to use buffers from the FLRU queue rather than the MLRU queue. (A modified buffer must be written to disk before the database server can use the buffer.)

F Free LRU with fewest elements

The database server uses this estimate to determine where to put unmodified (free) buffers next.

m MLRU queue

M MLRU queue that a flusher is cleaning

length Tracks the length of the queue measured in buffers

% of Shows the percent of LRU queue that this subqueue composes

For example, suppose that an LRU queue has 50 buffers, with 30 of those buffers in the MLRU queue and 20 in the FLRU queue. The *% of* column would list percents of 60.00 and 40.00, respectively.

pair total

Provides the total number of buffers in this LRU queue

priority levels

Displays the priority levels: LOW, MED_LOW, MED_HIGH, HIGH

The **-R** option also lists the priority levels.

Summary information follows the individual LRU queue information. You can interpret the summary information as follows:

dirty Is the total number of buffers that have been modified in all LRU queues

queued Is the total number of buffers in LRU queues

total Is the total number of buffers

hash buckets

Is the number of hash buckets

buffer size

Is the size of each buffer

start clean

Is the value of LRU_MAX_DIRTY

stop at Is the value of LRU_MIN_DIRTY

priority downgrades

Is the number of LRU queues downgraded to a lower priority.

priority upgrades

Is the number of LRU queues upgraded to a higher priority.

onstat -s: Print latch information

Syntax:

▶▶ onstat -s ◀◀

Use the `-s` option to display general latch information.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:57:17 -- 15360 Kbytes

Latches with lock or userthread set
name      address  lock wait userthread
```

Figure 15-124. `onstat -s` Output

Output Description

You can interpret output from this option as follows:

name Identifies the resource that the latch controls with the following abbreviations:

archive

Storage-space backup

bf Buffers

bh Hash buffers

chunks

Chunk table

ckpt Checkpoints

dbspace

DbSPACE table

flushctl

Page-flusher control

flushr Page cleaners

locks Lock table

loglog Logical log

LRU LRU queues

physb1

First physical-log buffer

physb2

Second physical-log buffer

physlog

Physical log

pt Tblspace tblspace

tblsps Tblspace table

users User table

address Is the address of the latch

This address appears in the `-u` (users) output wait field if a thread is waiting for the latch.

lock Indicates if the latch is locked and set

The codes that indicate the lock status (1 or 0) are computer dependent.

wait Indicates if any user thread is waiting for the latch

userthread

Is the shared-memory address of any user thread that is waiting for a latch

Instead this field contains the thread-control block address, which all threads have. You can compare this address with the user addresses in the **onstat -u** output to obtain the user-process identification number.

To obtain the **rstcb** address from the **tcb** address, examine the output of the **onstat -g ath** option, which lists both addresses for each user thread.

onstat -t and -T: Print tblspace information

Syntax:

► onstat [-t] [-T] ◀

Use the **-t** option to display tblspace information for active tblspaces, including whether tblspaces are memory resident. Use the **-T** option to display the total number of tblspaces.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:58:08 -- 15360 Kbytes

Tblspaces
 n address flgs ucnt tblnum physaddr npages nused npdata nrows nextns
62 a40dc70 0 1 100001 1:14 250 250 0 0 1
195 ac843e0 0 1 1000df 1:236 16 9 4 53 2
2 active, 221 total
```

Figure 15-125. **onstat -t** Output

Output Description

You can interpret output from this option as follows:

n Is a counter of open tblspaces

address Is the address of the tblspace in the shared-memory tblspace table

flgs Uses the following flag bits to describe the flag:

0x00000001

Partition structure is being initialized

0x00000002

Partition was modified. The modified pages have not been flushed to disk.

0x00000004

Partition is being dropped

0x00000008

Partition is for a pseudo table

- 0x00000010 Partition is being altered in an ADD INDEX or DROP INDEX operation
- 0x00000020 Partition is being altered in an ALTER TABLE operation
- 0x00000080 Partition is being dropped while the dbspace is down
- 0x00000100 Simple large objects in blobspaces are not deleted when the table is dropped
- 0x00000200 Partition alter page count is updated
- 0x00000400 Pages have been altered to the latest database schema
- 0x00000800 System temp table
- 0x00001000 User temp table
- 0x00004000 Index operations are deferred during recovery
- 0x00008000 Partition is being truncated
- 0x00010000 Partition is partially truncated

ucnt Is the usage count, which indicates the number of user threads currently accessing the tblspace

tblnum Is the tblspace number expressed as a hexadecimal value
The integer equivalent appears as the **partnum** value in the **systables** system catalog table.

physaddr Is the physical address (on disk) of the tblspace

npages Is the number of pages allocated to the tblspace

nused Is the number of used pages in the tblspace

npdata Is the number of data pages used

nrows Is the number of data rows used

nextns Is the number of noncontiguous extents allocated

This number is not the same as the number of times that a next extent has been allocated.

The **-t** option also lists the number of active tblspaces and the total number of tblspaces.

onstat -u: Print user activity profile

Syntax:

►► onstat -u ◀◀

Use the **-u** option to print a profile of user activity.

Example Output

```
Userthreads
address flags  sessid  user    tty    wait    tout locks nreads nwrites
a4d8018 ---P--D 1      informix -      0      0      0      58      4595
a4d8628 ---P--F 0      informix -      0      0      0      0      2734
a4d8c38 ---P--- 5      informix -      0      0      0      0      1
a4d9248 ---P--B 6      informix -      0      0      0      40      0
a4d9858 ---P--D 7      informix -      0      0      0      0      0
a4d9e68 Y--P--- 21     niraj   -      a65e5a8 0      1      0      0
6 active, 128 total, 7 maximum concurrent
```

Figure 15-126. onstat -u Output

Output Description

The **-u** option provides the following output for each user thread.

address Is the shared-memory address of the user thread (in the user table)

Compare this address with the addresses displayed in the **-s** output (latches); the **-b**, **-B**, and **-X** output (buffers); and the **-k** output (locks) to learn what resources this thread is holding or waiting for.

flags Provides the status of the session.

The flag codes for position 1:

- B** Waiting for a buffer
- C** Waiting for a checkpoint
- G** Waiting for a write of the logical-log buffer
- L** Waiting for a lock
- S** Waiting for mutex
- T** Waiting for a transaction
- Y** Waiting for condition
- X** Waiting for a transaction cleanup (rollback)

DEFUNCT

The thread has incurred a serious assertion failure, and has been suspended to allow other threads to continue their work.

The flag code for position 2:

- *** Transaction active during an I/O failure

The flag code for position 3:

- A** A dbspace backup thread

For other values that appear here, see the third position of flag codes for the `-x` option.

The flag code for position 4:

P Primary thread for a session

The flag codes for position 5:

R Reading

X Thread in critical section

The flag codes for position 7:

B A B-tree cleaner thread

C Terminated user thread waiting for cleanup

D A daemon thread

F A page-cleaner thread

M Special ON-Monitor thread (UNIX)

sessid Is the session identification number

During operations such as parallel sorting and parallel index building, a session might have many user threads associated with it. For this reason, the session ID identifies each unique session.

user Is the user login name (derived from the operating system)

tty Indicates the tty that the user is using (derived from the operating system)

This field is blank on Windows.

wait If the user thread is waiting for a specific latch, lock, mutex, or condition, this field displays the address of the resource. Use this address to map to information provided in the `-s` (latch) or `-k` (lock) output. If the wait is for a persistent condition, run a `grep` for the address in the `onstat -a` output.

tout Is the number of seconds left in the current wait

If the value is 0, the user thread is not waiting for a latch or lock. If the value is -1, the user thread is in an indefinite wait.

locks Is the number of locks that the user thread is holding

(The `-k` output should include a listing for each lock held.)

nreads Is the number of disk reads that the user thread has executed

nwrites

Is the number of write calls that the user thread has executed

All write calls are writes to the shared-memory buffer cache.

The last line of `onstat -u` output displays the maximum number of concurrent user threads that were allocated since you initialized the database server. For example, the last line of a sample `onstat -u` output is as follows:

```
4 active, 128 total, 17 maximum concurrent
```

The last part of the line, 17 maximum concurrent, indicates that the maximum number of user threads that were running concurrently since you initialized the database server is 17.

The output also indicates the number of active users and the maximum number of users allowed.

onstat -x: Print database server transaction information

Syntax:

```
▶▶ onstat -x ◀◀
```

Use the `-x` option to display transaction information on the database server. The transaction information is required only in the following situations:

- X/Open environment
- Database server participation in distributed queries
- Database server uses the Microsoft® Transaction Server (MTS) transaction manager

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 03:58:41 -- 15360 Kbytes

Transactions
address  flags  userthread  locks  beginlg  curlog  logposit  isol  retrys  coord
a509018  A----  a4d8018    0      0        6      0x17304c  COMMIT  0
a5091e8  A----  a4d8628    0      0        0      0x0      COMMIT  0
a5093b8  A----  a4d8c38    0      0        0      0x0      COMMIT  0
a509588  A----  a4d9248    0      0        0      0x0      COMMIT  0
a509758  A----  a4d9858    0      0        0      0x0      COMMIT  0
a509928  A---S  a4d9e68    1      0        0      0x0      COMMIT  0      xps_qa
6 active, 128 total, 8 maximum concurrent
```

Figure 15-127. `onstat -x` Output

Output Description

You can interpret output from `onstat -x` as follows:

address

Is the shared-memory address of the transaction structure

flags The flag codes for position 1 (current transaction state):

A User thread attached to the transaction

S TP/XA suspended transaction

C TP/XA waiting for rollback

The flag codes for position 2 (transaction mode):

T Tightly-coupled mode (MTS)

L Loosely-coupled mode (default mode)

The flag codes for position 3 (transaction stage):

B Begin work

P Distributed query prepared for commit

X TP/XA prepared for commit

- C Committing or committed
- R Rolling back or rolled back
- H Heuristically rolling back or rolled back

The flag code for position 4:

- X XA transaction

The flag codes for position 5 (type of transaction):

- G Global transaction
- C Distributed query coordinator
- S Distributed query subordinate
- B Both distributed query coordinator and subordinate

userthread

Is the thread that owns the transaction (**rstcb** address)

locks Is the number of locks that the transaction holds

beginlg

Is the log in which the BEGIN WORK record was logged

curlog Is the current log that the transaction is writing to

logposit

Is the log position

The format of a 4-byte log position is 0xPPPPBBB, where P is the page offset in the log and BBB is the byte offset in the page. The *logposit* can refer to a maximum of 0x100000 (or 1048576) pages in a log file.

For example, a record on the first page of log 12, at a byte offset of 24 would have a log position of 0x18 (page 0, byte offset 18). For more information, see "Determining the Position of a Logical-Log Record."

isol Is the isolation level.

retrys Are the attempts to start a recovery thread for the distributed query

coord Is the name of the transaction coordinator when the subordinate is executing the transaction

This field tells you which database server is coordinating the two-phase commit.

The last line of the **onstat -x** output indicates that 8 is the maximum number of concurrent transactions since you initialized the database server.

8 active, 128 total, 8 maximum concurrent

Determining the Position of a Logical-Log Record

The **curlog** and **logposit** fields provide the exact position of a logical-log record. If a transaction is not rolling back, **curlog** and **logposit** describe the position of the most recently written log record. When a transaction is rolling back, these fields describe the position of the most recently "undone" log record. As the transaction rolls back, the **curlog** and **logposit** values decrease. In a long transaction, the rate at which the **logposit** and **beginlg** values converge can help you estimate how much longer the rollback is going to take.

For an **onstat -x** example, see monitoring a global transaction in the chapter on multiphase commit protocols in the *IBM Informix Administrator's Guide*.

Determining the Mode of a Global Transaction

The **onstat -x** utility is useful for determining whether a global transaction is executing in loosely-coupled or tightly-coupled mode. The second position of the flags column displays the flags for global transactions. The T flag indicates tightly-coupled mode and the L flag indicates loosely-coupled mode.

Loosely-coupled mode means that the different database servers coordinate transactions but do not share locks. Each branch in a global transaction has a separate transaction XID. The records from all branches display as separate transactions in the logical log.

Tightly-coupled mode means that the different database servers coordinate transactions and share resources such as locking and logging. In a global transaction, all branches that access the same database share the same transaction XID. Log records for branches with the same XID appear under the same session ID. MTS uses tightly-coupled mode.

onstat -X: Print thread information

Syntax:

►► onstat -X ◀◀

Use the **-X** option to obtain precise information about the threads that are waiting for buffers. For each buffer in use, the **-X** option displays general buffer information that is also available with either the **-b** or **-B** option. For more information, refer to **onstat -b** in “onstat -b: Print buffer information for buffers in use” on page 15-9.

Example Output

```
IBM Informix Dynamic Server Version 11.50.UC1 -- On-Line -- Up 18:47:42 -- 34816 Kbytes
  Buffers (Access)
address owner  flags pagenum      memaddr  nslots pgflgs scout  waiter
  Buffer pool page size: 2048
  0 modified, 3000 total, 4096 hash buckets, 2048 buffer size
  Buffer pool page size: 8192
  0 modified, 1000 total, 1024 hash buckets, 8192 buffer size
```

Figure 15-128. onstat -X Output

Output Description

The **onstat -X** option has a **waiter** field to list all user threads that are waiting for the buffer, whereas the **onstat -b** and **-B** options contain a **waitlist** field that displays the address of the first user thread that is waiting for the buffer. The maximum number of shared buffers is specified in the **buffers** field in the BUFFERPOOL configuration parameter in the ONCONFIG file.

Buffer pool page size
is the size of the buffer pool pages in bytes

address

Is the address of the buffer header in the buffer table

flags Flags identifying the current status of the buffer page:

- 0x01 Modified Data
- 0x02 Data
- 0x04 LRU
- 0x08 Error
- 0x20 LRU AIO write in progress
- 0x40 Chunk write in progress
- 0x80 Buffer is/will be result of read-ahead
- 0x100 Cleaner assigned to LRU
- 0x200 Buffer should avoid bf_check calls
- 0x400 Do log flush before writing page
- 0x800 Buffer has been 'buff' -checked
- 0x8000 Buffer has been pinned

pagenum

Is the physical page number on the disk

memaddr

Is the buffer memory address

nslots Is the number of slot-table entries in the page

This field indicates the number of rows (or portions of a row) that are stored on the page.

pgflgs Uses the following values, alone or in combination, to describe the page type:

- 1 Data page
- 2 Tblspace page
- 4 Free-list page
- 8 Chunk free-list page
- 9 Remainder data page
- b Partition resident blobpage
- c Blobspace resident blobpage
- d Blob chunk free-list bit page
- e Blob chunk blob map page
- 10 B-tree node page
- 20 B-tree root-node page
- 40 B-tree branch-node page
- 80 B-tree leaf-node page
- 100 Logical-log page
- 200 Last page of logical log
- 400 Sync page of logical log
- 800 Physical log
- 1000 Reserved root page
- 2000 No physical log required
- 8000 B-tree leaf with default flags

scount Displays the number of threads that are waiting for the buffer
waiter Lists the addresses of all user threads that are waiting for the buffer

onstat -z: Clear statistics

Syntax:

▶▶ onstat -z ◀◀

Use the **-z** option to clear database server statistics, including statistics that relate to Enterprise Replication, and set the profile counts to 0.

If you use the **-z** option to reset and monitor the count of some fields, be aware that profile counts are incremented for all activity that occurs in any database that the database server manages. Any user can reset the profile counts and thus interfere with monitoring that another user is conducting.

Return Codes on Exit

The **onstat** utility returns the following codes on exit.

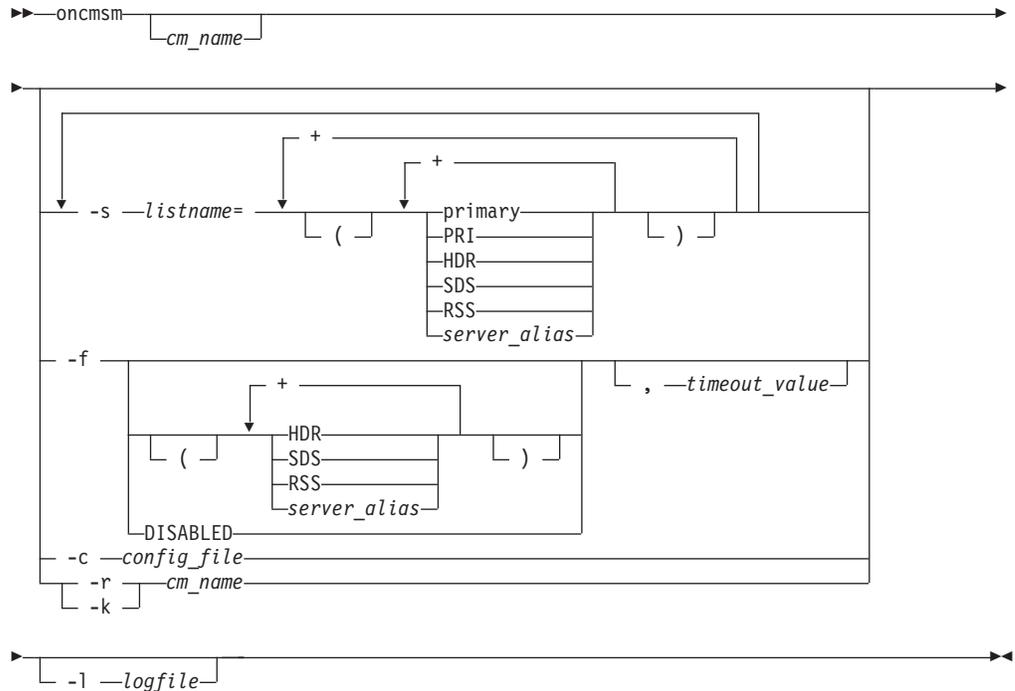
GLS failures: -1
Failed to attach shared memory: -1
Failed to attach shared memory when running 'onstat -': 255
All other errors detected by onstat: 1
No errors detected by onstat: 0
Administration mode: 7

Chapter 16. The oncmsm Utility

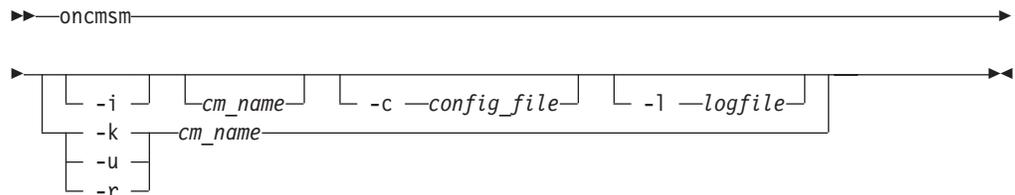
The **oncmsm** utility is used to configure and start the Connection Manager.

Syntax

UNIX syntax diagram:



Windows syntax diagram:



Element	Purpose	Key Considerations
-i	Use the -i option to install oncmsm as a Windows service.	This option is valid for Windows platforms only.
-u	Use the -u option to uninstall oncmsm as a Windows service.	This option is valid for Windows platforms only.
<i>cm_name</i>	Name of the Connection Manager instance.	
<i>server_alias</i>	Used to specify a server alias name.	

Element	Purpose	Key Considerations
primary	Specifies the primary server.	Use PRI or primary to specify the primary server. Both have the same meaning.
PRI	Specifies the primary server.	
HDR	Specifies a HDR (High-Availability Data Replication) secondary server.	
SDS	Specifies a SD (Shared Disk) secondary server.	
RSS	Specifies a RS (Remote Standalone) secondary server.	
-s listname=	Used to identify the Service Level Agreement (SLA) of a given listener port.	The -s option is valid for UNIX systems only.
-f failover_configuration	Specifies the secondary servers for failover.	If <i>failover_configuration</i> does not contain any valid servers during failover, then a message is logged in the configuration manager's log file. See Table 16-1 for valid <i>failover_configuration</i> options. The -f option is valid for UNIX systems only.
<i>timeout_value</i>	Specifies the time (in seconds) to wait before performing failover.	If the Connection Manager is not able to determine within the time specified by <i>timeout_value</i> that the current primary server is active, then it will perform the failover. If the Connection Manager receives an indication that the primary server is back online within the time period specified by <i>timeout_value</i> , then no failover attempt will be performed.
-c config_file	Identifies an optional configuration file that will contain service level agreements and failover options.	See "Configuration File Format" on page 16-3 for additional information.
-k cm_name	Use this option to shut down a specific instance of the Connection Manager.	
-l logfile	Used to specify the name of a log file.	
-r cm_name	Use this option to reload Connection Manager settings.	Use the -r option to reload Connection Manager settings without stopping and restarting the Connection Manager.

The following table describes the valid options for configuring failover options.

Table 16-1. Valid Failover Configuration Parameters

Failover configuration parameter	Description
SDS	SD (Shared Disk) secondary server
HDR	HDR (High-Availability Data Replication) secondary server
RSS	RS (Remote Standalone) secondary server
<i>alias</i>	Alias name of secondary server
DISABLED	Disables the failover functionality of the Connection Manager.

Usage

The **oncmsm** utility starts the Connection Manager, which is used to manage and redirect client connection requests based on service level agreements configured by the system administrator. The Connection Manager is a daemon program that accepts a connection request from a client connection and then returns a redirected connection to the client. The Connection Manager connects to one or more of the servers in a high-availability cluster and gathers statistics from each server, such as the server type, the unused workload capacity, and the current state of the server. From this information, the Connection Manager is able to redirect the client connection to the appropriate server.

When the Connection Manager must choose among multiple servers to connect with, it decides which server to connect to based on the free CPU cycles on each of the servers. For example, if CPU utilization of one of the servers is at 10 percent and another is at 50 percent, then the server with 10 percent utilization will be used to resolve the connection request.

The Connection Manager is configured as a normal server in the **sqlhosts** file, enabling you to configure a group of Connection Manager daemons as a unit, which allows the ability to provide failover of the connection managers.

Execute the **oncmsm** utility from the command line to initialize the Connection Manager. For more information and examples of using the **oncmsm** utility, refer to the *IBM Informix Administrator's Guide*.

Note: If the Connection Manager failover feature is enabled, and the primary server fails, do not attempt to manually restart the failed primary server until failover processing has completed.

UNIX Only: Only user **informix** can execute **oncmsm**. If user **root** or a member of the DBSA group is granted privileges to connect to the sysadmin database, then user **root** or that member of the DBSA group can also invoke **oncmsm**.

Windows Only:

- You must be a member of the **Informix-Admin** group to execute **oncmsm**. If user **administrator** or a member of the DBSA group is granted privileges to connect to the sysadmin database, then user **administrator** or that member of the DBSA group can also invoke **oncmsm**.
- You must first install **oncmsm** as a service. Two additional options can be passed on the **oncmsm** command line. Use the **-i** option to install **oncmsm** as a Windows service, and **-u** to uninstall the service. Once **oncmsm** is installed as a Windows service, it can be started from either a command line by running **oncmsm** or it can be started from the **Services** program (click **Start**, click **Control Panel**, double-click **Administrative Tools**, and then double-click **Services**).

Configuration File Format

The format of the configuration file is as follows:

```
NAME      Connection Manager instance name
SLA       name=value
SLA       name=value
DEBUG     value (0 = no debug, 1=debug)
LOGFILE   path to log file
FOC       failover_configuration,timeout_value
```

In the configuration file above, *value* can refer to a server name, a server type, or a server list.

The default configuration file name and location is:

```
$INFORMIXDIR/etc/cmsm.cfg
```

To start the Connection Manager using a configuration file other than the default file:

```
oncmsm -c /path and file name of config file
```

Alternatively, if `$INFORMIXDIR/etc/cmsm.cfg` already exists, then Connection Manager can be started by:

```
oncmsm
```

Service Level Agreement Examples

In each of the following examples, it is possible to use specific server alias names or generic server types. For example, specifying the keyword **RSS** indicates a generic server type, which means that the Connection Manager will evaluate every RS secondary server in the cluster. You can also refer to a specific server by specifying its alias, such as *sds1*, or *rss1*.

The first example starts the Connection Manager with two service level agreements. The first SLA directs client connection requests such as connect to @oltp to the primary server. The second SLA directs connection requests such as connect to @report to either the HDR secondary server or to the SD secondary server, depending on which server has the lowest CPU utilization.

```
oncmsm -s oltp=primary -s report=HDR+SDS
```

The order of the entries after the `-s` option determines the order that the Connection Manager will attempt to resolve the request. In the next example, the Connection Manager first attempts to resolve a client request of connect to @secondaryNodes by connecting to one of the SD secondary servers. If no SD secondary servers exist, then the Connection Manager will attempt to direct the connection to the HDR secondary server. If the HDR secondary server is inoperative, then the Connection Manager will attempt to connect to the primary server.

```
oncmsm -s secondaryNodes=SDS+HDR+primary
```

Connection types can be grouped by using parentheses. The next example causes the Connection Manager to evaluate the SD secondary servers and only connect to the HDR secondary if the SD secondary servers are not available.

```
oncmsm -s secondary=SDS+HDR
```

Adding parentheses indicates that the connection could be routed to an SD secondary server or to the HDR secondary server based on which server has the lowest CPU utilization.

```
oncmsm -s secondary=(SDS+HDR)
```

Failover Configuration Examples

These examples show failover configurations used in configuration files. It is also possible to specify a failover configuration from the **oncmsm** command line. If you

specify both a configuration file with a failover configuration and a failover configuration on the command line, the failover configuration in the configuration file takes precedence.

The following examples show several ways to configure Connection Manager to promote one of the secondary servers in a high-availability cluster configuration to the primary server in the event the original primary server encounters a problem.

In the first example, if the Connection Manager detects that the primary server is off line, then failover processing starts immediately because the timeout value is set to zero. The Connection Manager first attempts to convert the best available SD secondary server to the primary server; if no SD secondary servers are on line, it attempts to convert the HDR secondary into the primary server; if that is not possible, it attempts to convert the most suitable RS secondary server to the primary server. If `-f` is not specified in the command line, or if the failover configuration (FOC) is not specified in the configuration file, the Connection Manager will attempt to convert the RS secondary server to the primary server.

```
FOC SDS+HDR+RSS,0
```

In the next example, setting the timeout value to 10 causes the Connection Manager to wait 10 seconds for the primary server to come back on line before starting failover processing. If the primary server does not come back on line within the specified time, then the Connection Manager attempts to convert any SD secondary server, any HDR secondary server, or any RS secondary server to the primary server.

```
FOC (SDS+HDR+RSS),10
```

To specify a group of specific servers that can become the primary server, use alias names. In the next example, the Connection Manager immediately performs a failover because the **timeout_value** is not specified and is assumed to be 0. The Connection Manager attempts to fail over to the server with alias **hdrs1**; if **hdrs1** is not available, then it attempts to fail over to **sds1** if possible, and so on, trying **rss1**, **sds2**, and finally **rss2**.

```
FOC hdrs1+sds1+rss1+sds2+rss2
```

In the next example, if the Connection Manager is not able to determine within 20 seconds that the current primary server is on line, then failover processing is started. The Connection Manager first attempts to convert the server with alias **sds1** to the primary server if possible; if not, then it attempts to convert the server with alias **hdrs1** to the primary server if possible; if not then the best SD secondary server is converted to the primary. Note that this failover configuration will not fail over to RS secondary servers because none are specified.

```
FOC sds1+hdrs1+SDS,20
```


Chapter 17. The onpassword Utility

Use the **onpassword** utility to encrypt or decrypt a password file so that the Connection Manager can securely connect to each of the servers in a cluster.

Syntax

```

▶▶ onpassword -k access_key [-e plaintext_file | -d output_filename]

```

Element	Purpose	Key Considerations
-k <i>access_key</i>	Specifies the key used to encrypt a plain text password file. The same key must be specified in order to decrypt the file.	The length of the <i>access_key</i> cannot exceed 24 characters. The <i>access_key</i> is required in order to decrypt and edit the file. If the administrator loses or forgets the access key, decrypting and editing the password file will not be possible. In such cases, the recommended solution is to re-encrypt the plain text password file. Also see "Password Key" on page 17-2.
-e <i>plaintext_file</i>	Encrypt the specified ASCII plain text file containing user IDs, password, and server names. When the -e option is used, the output file is always written to \$INFORMIXDIR/etc/passwd_file.	See "Password File Structure" on page 17-2.
-d <i>output_filename</i>	Decrypts the specified encrypted password file. When -d option is used, the file is written to the specified location.	See "Password File Structure" on page 17-2.

Usage

The **onpassword** utility is used to encrypt or decrypt password files. A password file contains user IDs, passwords, and server information that are required for connecting to Informix servers in high-availability clusters. The password file is used by the Connection Manager. Only users logged in as **informix** have permission to run the **onpassword** utility.

On both UNIX and Windows systems, an encrypted password file is required in order to configure the Connection Manager to run in an untrusted network environment. In certain situations, an encrypted password file is also required for trusted network environments. An encrypted password file is required when a local system account attempts to connect to a server within the cluster, or when the user ID does not exist on one or more servers within the cluster. The password file is required because a local system account or a Windows-only account cannot directly connect to a remote server. In this case, the Connection Manager uses the encrypted password file to provide the correct system-level access for the request. The password file should contain suitable ID and password combinations that allow the Connection Manager to connect to any of the servers within the cluster.

You decrypt and modify the password file if you add or remove servers from your cluster, change the passwords, or change your secret key.

The encrypted password file cannot be copied from one platform to another. For example, password files created on a UNIX platform will not work on a Windows platform.

Password File Structure

The input file (*plaintext_file*) is an ASCII text file with the following structure:

```
ServerName_1 AlternateServer_1 UserName_1 Password_1  
ServerName_2 AlternateServer_2 UserName_2 Password_2
```

... and so on ...

The file contains the server names, user names, and passwords that must be specified in order to connect to the appropriate server. **AlternateServer** specifies the name of an alternate server to connect with in case the connection cannot be made to **ServerName**. For example, **AlternateServer** name is used when **ServerName** is located on a secure port (using the `s=6` option in the `sqlhosts` file).

Password Key

The password key can be any sequence of numbers or letters up to 24 characters long. To include a space in the key, enclose the entire key in quotation marks.

Examples

The first example encrypts a file named `passwords.txt` using a key named `SecretKey` and places the encrypted file in `$INFORMIXDIR/etc/passwd_file`:

```
onpassword -k SecretKey -e ./passwords.txt
```

The next example decrypts the `$INFORMIXDIR/etc/passwd_file` and places the unencrypted file in `passwords.txt`:

```
onpassword -k SecretKey -d ./passwords.txt
```

You must specify the key used to encrypt the file; in this case, `SecretKey`.

Part 3. Appendixes

Appendix A. Files That the Database Server Uses

This appendix provides brief summaries of the files that you use when you configure and use the database server. It also includes descriptions of files (and one directory) created and used internally by the database server. For many of these files, your only responsibilities are to recognize that those files are legitimate and refrain from deleting them.

Pathnames that appear in the following format indicate files that reside on UNIX: **/directory/filename**. Pathnames that appear in the following format indicate files that reside on Windows: **\directory\filename**.

In some cases, environment variables are used to specify the initial pathname of a file. On UNIX, references to environment variables begin with a dollar sign: **\$INFORMIXDIR**. On Windows, references to environment variables begin and end with percent signs: **%INFORMIXDIR%**.

Database Server Files

Table A-1 lists the database server files and the directories in which they reside.

Table A-1. List of Files That the Database Server Uses

Filename	Directory	Purpose	Created
af.xxx	Specified by DUMPPDIR configuration parameter	Assertion-failure information	By the database server
ac_msg.log	/tmp, %INFORMIXDIR%\etc	archecker message log (for Technical Support)	By the database server
ac_config.std	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Template for archecker-parameter values	By the database server
bar_act.log	/tmp, %INFORMIXDIR%\etc	ON-Bar activity log	By ON-Bar
bldutil.process_id	/tmp, \tmp	Error messages about the sysutils database appear in this file	By the database server
buildsmi.out (UNIX) buildsmi_out (Windows)	/tmp, %INFORMIXDIR%\etc (UNIX) %INFORMIXDIR%\etc\ buildsmi_out. %INFORMIXSERVER% (Windows)	Error messages about SMI database	By the database server
concdr.sh	\$INFORMIXDIR /etc/conv, %INFORMIXDIR%\etc\conv	Converts the syscdr database to Version 10.0 format	By the database server
.conf.dbservername		The onsnmp utility uses this file to obtain the database server configuration	By the database server
core	Directory from which the database server was invoked	Core dump	By the database server

Table A-1. List of Files That the Database Server Uses (continued)

Filename	Directory	Purpose	Created
Emergency boot files (For filenames, see “ Emergency Boot Files for ON-Bar” on page A-5).	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Used in a cold restore	By ON-Bar
gcore (UNIX)	Specified by DUMPDIR configuration parameter	Assertion failure information	By the database server
illsrta.xx	\$INFORMIXDIR/lib, %INFORMIXDIR%\lib	Shared libraries for the database server and some utilities	By install procedure
.informix (UNIX)	User’s home directory	Set personal environment variables	By the user
informix.rc (UNIX)	\$INFORMIXDIR/etc	Set default environment variables for all users	By the database administrator
INFORMIXTMP	/INFORMIXTMP, \%INFORMIXDIR%.	Temporary directory for internal files	By the database server
.inf.servicename	/INFORMIXTMP,drive:\ INFORMIXTMP	Connection information	By the database server
.infos.dbservername	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Connection information	By the database server
.infxdirs	/INFORMIXTMP,drive:\ INFORMIXTMP	Database server discovery file that onsnmp uses	By the database server
InstallServer.log (Windows)	C:\temp	Database server installation log	By the database server
ISM catalog	\$INFORMIXDIR/ism, %ISMDIR%	Records saved backup objects and storage volumes that IBM Informix Storage Manager (ISM) uses	By ISM
ISM logs	\$INFORMIXDIR/ism/logs, %ISMDIR%\logs	Operator alert messages, backend status, additional ISM information	By ISM
ISMversion	\$INFORMIXDIR/ism, %ISMDIR%	ISM version	During installation
JVM_vpid	Specified by JVPLOG configuration parameter	Messages that the Java virtual machine generates	By the Java virtual machine
JVPLOG	Specified by JVPLOG configuration parameter	Messages from the Java virtual processor	By the database server
.jvpprops	Specified by JVPPROFILE configuration parameter	Template for Java VP properties	During installation
Message log	Specified by MSGPATH configuration parameter	Error messages and status information	By the database server
The ONCONFIG file	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Configuration information	By the database administrator
onconfig	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Default ONCONFIG file (optional)	By the database server administrator
onconfig.std	\$INFORMIXDIR/etc	Template for configuration- parameter values	During installation
oncfg_servicename. servernum	\$INFORMIXDIR/etc, %INFORMIXDIR%\etc	Configuration information for whole-system restores	By the database server

Table A-1. List of Files That the Database Server Uses (continued)

Filename	Directory	Purpose	Created
<code>onsnmp.servername</code>	<code>/tmp, \tmp</code>	Log file that the onsnmp subagent uses	By onsnmp
<code>onsrvapd.log</code>	<code>/tmp, \tmp</code>	Log file for the database server daemon onsrvapd	By onsnmp
<code>revcdr.sh</code>	<code>\$INFORMIXDIR /etc/conv, %INFORMIXDIR% \etc\conv</code>	Reverts the syscdr database to an earlier format	By the database server
<code>servicename.exp</code>	<code>/INFORMIXTMP, drive:\INFORMIXTMP</code>	Connection information	By the database server
<code>servicename.str</code>	<code>/INFORMIXTMP, drive:\INFORMIXTMP</code>	Connection information	By the database server
<code>shmem.xxx</code> (UNIX)	Specified by DUMPPDIR configuration parameter	Assertion-failure information	By the database server
<code>sm_versions.std</code>	<code>\$INFORMIXDIR/etc, %INFORMIXDIR%\etc</code>	Identifies storage manager in use	During installation
<code>snmpd.log</code>	<code>/tmp, \tmp</code>	Log file for the SNMP master agent, snmpdm	By onsnmp
<code>sqlhosts</code> (UNIX)	<code>\$INFORMIXDIR/etc</code>	Connection information; contained in the registry on Windows	During installation; modified by the database server administrator
<code>VP.servername.nnx</code>	<code>/INFORMIXTMP, drive:\INFORMIXTMP</code>	Connection information	By the database server
<code>xbsa.messages</code>	<code>\$INFORMIXDIR /ism/applogs, %ISMDIR%\applogs</code>	XBSA library call information	By ISM

Descriptions of Files

This section provides short descriptions of the files listed in Table A-1 on page A-1.

af.xxx

The database server writes information about an assertion failure to the **af.xxx** file. The file is stored in the directory that the DUMPPDIR configuration parameter specifies. For more information, see the information on monitoring for data inconsistency in your *IBM Informix Administrator's Guide*.

ac_msg.log

When you use **archecker** with ON-Bar to verify a backup, it writes brief status and error messages to the ON-Bar activity log and writes detailed status and error messages to the **archecker** message log (**ac_msg.log**). Technical Support uses the **archecker** message log to diagnose problems with backups and restores.

You specify the location of the **archecker** message log with the AC_MSGPATH configuration parameter. For more information, see the *IBM Informix Backup and Restore Guide*.

ac_config.std

The **ac_config.std** file contains the default **archecker** (archive checking) utility parameters. To use the template, copy it into another file, and modify the values. For a comprehensive list of the **archecker** parameters and how to use **archecker** with ON-Bar, see the *IBM Informix Backup and Restore Guide*.

bar_act.log

As ON-Bar backs up and restores data, it writes progress messages, warnings, and error messages to the ON-Bar activity log (**bar_act.log**). You specify the location of the ON-Bar activity log with the **BAR_ACT_LOG** configuration parameter. For more information, see the *IBM Informix Backup and Restore Guide*.

bldutil.process_id

If the database server cannot build the sysutils database, it creates the **bldutil.<process_id>** file which contains the error messages. The **process_id** value is the process ID of the **bldutil.sh** program. To access this output file, specify **\${RESFILE}**.

buildsmi.out (UNIX) or buildsmi_out (Windows)

If the database server cannot build the **sysmaster** database, it places a message in the message log that refers you to the output file of the **buildsmi** script, whose filename and pathname are platform-dependent.

On UNIX, the file specification is

```
/tmp/buildsmi.out
```

On Windows, the file specification is

```
%INFORMIXDIR%\etc\buildsmi_out.%INFORMIXSERVER%
```

Here **%INFORMIXSERVER%** is the name of the Dynamic Server instance. This file provides information about why the build failed. For information about the **sysmaster** database, refer to Chapter 2, "The sysmaster Database."

concdr.sh

To convert the **syscdr** database from 7.31, 9.20, 9.21, 9.3 or 9.4 to 10.0 format, run the **concdr.sh** script on UNIX or the **concdr.bat** script on Windows. For details, see the *IBM Informix Migration Guide*.

.conf.dbservername

The **.conf.dbservername** file is created when you initialize the database server. The **onsnmp** utility queries this file to find out the configuration status of the database server. Do not delete this file.

The **.conf.dbservername** file contains information on shared memory and configuration that allows shared-memory clients to connect to the database server when they use utilities such as **onstat** or **onmode**.

core

The **core** file contains a core dump caused by an assertion failure. The database server writes this file to the directory from which the database server was invoked. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the *IBM Informix Administrator's Guide*.

Emergency Boot Files for ON-Bar

The ON-Bar emergency boot files contain the information needed to perform a cold restore, and are updated after every backup. For details, see the *IBM Informix Backup and Restore Guide*.

The filename for the Dynamic Server emergency boot file is **ixbar_hostname.servernum**.

gcore.xxx (UNIX)

The database server writes information about an assertion failure to the **gcore.xxx** file. The file is stored in the directory specified by the DUMPDIR configuration parameter. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the *IBM Informix Administrator's Guide*.

illsrta.xx

The **illsrta.xx** files are shared libraries that the database server and some database server utilities use. The shared libraries, if supported on your platform, are installed in **\$INFORMIXDIR/lib** or **%INFORMIXDIR%\lib**.

The naming convention of the Informix shared library filename is as follows:

illsrta.xx

<i>lll</i>	library class (for example, asf or smd)
<i>s</i>	library subclass (d=DSA; s=standard)
<i>rr</i>	major release number (for example, 07 or 08)
<i>a</i>	library version ID (for example, a or b)
<i>xx</i>	shared-library filename extension (for example, so)

UNIX Only:

Symbolic links to these files are automatically created in **/usr/lib** when the products are installed on your computer. The symbolic links to the shared libraries in **/usr/lib** are automatically created by the product installation procedures. However, if your **\$INFORMIXDIR** is not installed using the standard installation method (for example, your **\$INFORMIXDIR** is NFS-mounted from another computer), you or your system administrator might need to create manually the symbolic links of the shared libraries in **/usr/lib**.

~/.informix

The **~/.informix** file is the *private-environment file*. Users can create this file and store it in their home directory. The *IBM Informix Guide to SQL: Reference* discusses the environment-configuration files.

informix.rc (UNIX)

The `/informix.rc` file is the *environment-configuration file*. You can use it to set environment variables for all users of IBM Informix products. The *IBM Informix Guide to SQL: Reference* discusses the environment-configuration files.

INFORMIXTMP

The **INFORMIXTMP** directory is an *internal database server directory*. During initialization, the database server creates this directory (if it does not exist yet) for storing internal files that must be local and relatively safe from deletion. The **onsnmp** utility uses the files in the **INFORMIXTMP** directory.

.inf.servicename

The database server creates the **.inf.servicename** file if any **DBSERVERNAME** or **DBSERVERALIASES** uses a shared-memory connection type. The database server removes the file when you take the database server offline. The name of this file is derived from the **servicename** field of the **sqlhosts** file or registry.

The database server keeps information about client/server connections in this file. You do not use the **.inf.servicename** file directly. You only need to recognize that it is a legitimate file when it appears in the **INFORMIXTMP** directory.

If this file is accidentally deleted, you must restart the database server.

.infos.dbservername

The database server creates the **.infos.dbservername** file when you initialize shared memory and removes the file when you take the database server offline. This file resides in **\$INFORMIXDIR/etc** or **%INFORMIXDIR%\etc**. The name of this file is derived from the **DBSERVERNAME** parameter in the **ONCONFIG** configuration file.

The **.infos.dbservername** file contains information on shared memory and configuration that allows shared-memory clients to connect to the database server when they use utilities such as **onstat** or **onmode**. Do not delete this file.

.infxdirs

The database server maintains an **.infxdirs** file in the **INFORMIXTMP** directory. This file contains a line for every **INFORMIXDIR** from which a database server has been launched. If you remove the **.infxdirs** file, **onsnmp** cannot discover any database servers until the next time you restart the database server. Each time you restart the database server, it re-creates the **.infxdirs** file.

InstallServer.log (Windows)

The database server creates the **InstallServer.log** during installation.

ISM Catalog

ISM creates the ISM catalog during the **ism_startup** initialization. The ISM catalog records information about backup and restore save sets and about storage volumes

that the storage manager uses. The ISM catalog records are stored in the **mm**, **index**, and **res** files in the **\$INFORMIXDIR/ism** or **%ISMDIR%\ism** directory. For more information, see the *IBM Informix Storage Manager Administrator's Guide*.

ISM Logs

ISM creates several logs during ON-Bar backup and restore operations. The message window in the ISM Administrator GUI displays messages from these logs.

Log	Description
daemon.log	ISM backend status
messages	Operator alert messages
summary	Additional ISM information

For more information, see the *IBM Informix Storage Manager Administrator's Guide*.

ISMversion

The **ISMversion** file, which is installed with the database server, identifies the ISM version. Do not edit this file.

JVM_vpid

When the 0x10 bit is on for AFCRASH or the **AFDEBUG** environment variable is on, all the messages that the Java virtual machine generates are logged into the **JVM_vpid** file, where **vpid** is the process ID of the Java virtual processor. For more information, see *J/Foundation Developer's Guide*.

JVPLOG

When JVPDEBUG is set to 1, the database server writes tracing messages to the **JVPLOG** file. You can adjust the tracing level. On UNIX, you can have multiple **JVPLOG** files, one for each JVP virtual processor. On Windows, only one **JVPLOG** file exists. To obtain the JVP IDs, use the **onstat -g glo** command. For more information, see *J/Foundation Developer's Guide*.

.jvpprops

The **.jvpprops** file sets the Java virtual processor properties. Copy the **.jvpprops.template** to a new file named **.jvpprops**, and modify the values. For more information, see *J/Foundation Developer's Guide*.

Message Log

The database server writes status and error information to the message-log file. You specify the filename and location of the message log with the **MSGPATH** configuration parameter. For more information, refer to "MSGPATH" on page 1-68.

onconfig.std

The **onconfig.std** file serves as the template for creating the ONCONFIG configuration file. To use the template, copy it to another file and modify the values.

Important: Do not modify or delete **onconfig.std**. The database server uses values listed in this file when those values are missing from the ONCONFIG file.

For a comprehensive list of the ONCONFIG parameters, see Chapter 1, “Configuration Parameters,” on page 1-1.

The ONCONFIG File

The *current configuration file* is the `%INFORMIXDIR%\etc\%ONCONFIG%` or `$INFORMIXDIR/etc/$ONCONFIG` file. The database server uses the ONCONFIG file during initialization.

If you start the database server with `oninit` and do not explicitly set the **ONCONFIG** environment variable, the database server looks for configuration values in the **onconfig.std** file. If no **onconfig.std** file exists, the database server returns the following error message:

```
WARNING: Cannot access configuration file $INFORMIXDIR/etc/$ONCONFIG.
```

For more information on the order of files where the database server looks for configuration values during initialization, refer the material on initializing the database server in the *IBM Informix Administrator's Guide*.

For more information on setting up your ONCONFIG file, refer to the materials on installing and configuring the database server in the *IBM Informix Administrator's Guide*.

onconfig

The **onconfig** file is an optional file that you create in the `$INFORMIXDIR/etc` or `%INFORMIXDIR%\etc` directory. The **onconfig** file is the default configuration file if the **ONCONFIG** environment variable is not set. For more information, refer to processing the configuration file in the *IBM Informix Administrator's Guide*.

To create the **onconfig** file, you can copy **onconfig.std** or one of your customized configuration files. For more information on setting up your **ONCONFIG** file, refer to installing and configuring the database server in the *IBM Informix Administrator's Guide*.

oncfg_servername.servernum

The database server creates the **oncfg_servername.servernum** file in the `$INFORMIXDIR/etc` or `%INFORMIXDIR%\etc` directory when you initialize disk space. The database server updates the file every time that you add or delete a dbspace, a logical-log file, or a chunk. The database server uses the **oncfg_servername.servernum** file when it salvages logical-log files during a whole-system restore. The database server derives the name of this file from the values of the **DBSERVERNAME** and **SERVERNUM** parameters in the ONCONFIG configuration file.

The database server uses the **oncfg_servername.servernum** files, so do not delete them. For more information, refer to creating the **oncfg_servername.servernum** file in the *IBM Informix Administrator's Guide* and the *IBM Informix Backup and Restore Guide*.

onsnmp.servername

The **onsnmp** subagent uses this log file. For more information, see the *IBM Informix SNMP Subagent Guide*.

This log file is called **onsnmp.servername** on Dynamic Server.

onsrvapd.log

The **onsrvapd** daemon uses this log file. For more information, see the *IBM Informix SNMP Subagent Guide*.

revcdr.sh

To revert the **syscdr** database from 10.0 to 9.4, 9.3, 7.31, 9.20, or 9.21 format, run the **revcdr.sh** script on UNIX or the **revcdr.bat** script on Windows. For details, see the *IBM Informix Migration Guide*.

shmem.xxx (UNIX)

The database server writes information about an assertion failure to the **shmem.xxx** file. The file is stored in the directory that the DUMPPDIR configuration parameter specifies. For more information on monitoring for data inconsistency, see the chapter on consistency checking in the *IBM Informix Administrator's Guide*.

sm_versions.std

The **sm_versions.std** file is a template for the **sm_versions** file that you create. The **sm_versions** file contains a line identifying the current storage-manager version.

The storage manager uses the data in the **sm_versions** file (no **.std** suffix). To update the storage-manager version, edit the **sm_versions** file and then run the **ism_startup** command. For more information, see the *IBM Informix Backup and Restore Guide*.

snmpd.log

The SNMP master agent, **snmpdm** uses this log file. For more information, see the *IBM Informix SNMP Subagent Guide*.

sqlhosts

UNIX Only:

The **sqlhosts** file is the *connectivity file* on UNIX platforms. It contains information that lets an IBM Informix client connect to an IBM Informix database server. For more information on the **sqlhosts** file, see client/server communications in the *IBM Informix Administrator's Guide*.

Windows Only:

On Windows, the connectivity information is in the **HKEY_LOCAL_MACHINE\SOFTWARE\INFORMIX\SQLHOSTS** key in the Windows registry.

VP.servername.nnx

The database server creates the **VP.servername.nnx** file, if needed, when you initialize shared memory. The name of this file comes from **DBSERVERNAME** or **DBSERVERALIASES** in the **ONCONFIG** file, the VP number (**nn**), and an internal identifier (**x**).

The database server keeps information about client/server connections in the **VP.servername.nnx** file. You do not use the file directly. You only need to recognize that it is a legitimate file.

If this file is accidentally deleted, you must restart the database server.

xbsa.messages

The **xbsa.messages** log contains XBSA library call information. ON-Bar and ISM use XBSA to communicate with each other. Technical Support would use the **xbsa.messages** log to diagnose problems with ON-Bar and ISM communications.

Appendix B. Trapping Errors

Occasionally, a series of events causes the database server to return unexpected error codes. If you do not have the appropriate diagnostic tools in place when these events occur, it might be difficult for you to determine the cause of these errors. This section discusses the following diagnostic tools:

- **onmode -I**
- tracepoints

Collecting Diagnostics using onmode -I

To help collect additional diagnostics, you can use **onmode -I** to instruct the database server to perform the diagnostics collection procedures that the *IBM Informix Administrator's Guide* describes. To use **onmode -I** when you encounter an error number, supply the *iserrno* and an optional session ID. The **-I** option is just one of many **onmode** options. For more information about **onmode**, see "In This Chapter " on page 11-1.

Syntax



Element	Purpose	Key Considerations
-I iserrno	Error number of the error for which you want to collect diagnostic information	None.
sid	Session ID of the session for which you want to collect diagnostic information	None.

Creating Tracepoints

Tracepoints are useful in debugging user-defined routines written in C. You can create a user-defined tracepoint to send special information about the current execution state of a user-defined routine.

Each tracepoint has the following parts:

- A *trace class* groups related tracepoints together so that they can be turned on or off at the same time.
You can either use the built-in trace class called **_myErrors** or create your own. To create your own trace class, you insert rows into the **systraceclasses** system catalog table.
- A *trace message* is the text that the database server sends to the tracing-output file.
You can store internationalized trace messages in the **systracemsgs** system catalog table.
- A *tracepoint threshold* determines when the tracepoint executes.

By default, the database server puts all trace messages in the trace-output file in the **tmp** directory with the following filename:

`session_num.trc`

For more information on tracing user-defined routines, see the *IBM Informix DataBlade API Programmer's Guide*.

Appendix C. Event Alarms

The database server provides a mechanism for automatically triggering administrative actions based on an event that occurs in the database server environment. This mechanism is the event-alarm feature. Events can be informative (for example, Backup Complete) or can indicate an error condition that requires your attention (for example, Unable to Allocate Memory).

Using ALARMPROGRAM to Capture Events

On UNIX, use the **alarmprogram.sh** and on Windows, use the **alarmprogram.bat** shell script, for handling event alarms and starting automatic log backups. For the setup instructions, see “ALARMPROGRAM” on page 1-12.

To automate logical-log backups only, two ready-made scripts are provided: **log_full.[sh | bat]** and **no_log.[sh | bat]**. Set ALARMPROGRAM to the full pathname of the script. For information, see “ALARMPROGRAM” on page 1-12.

Setting ALRM_ALL_EVENTS

You can set ALRM_ALL_EVENTS to specify whether ALARMPROGRAM runs for all events that are logged in the MSGPATH or only specified noteworthy events (events greater than severity 1).

Writing Your Own Alarm Script

Alternatively, you can write your own shell script, batch file, or binary program that contains the event-alarm parameters. When an event occurs, the database server invokes this executable file and passes it the event-alarm parameters (see Table C-1 on page C-3). For example, your script can use the **class_id** and **class_msg** parameters to take administrative action when a table failure occurs. Set ALARMPROGRAM to the full pathname of this executable file.

Customizing the ALARMPROGRAM scripts

Follow these steps to customize the **alarmprogram.[sh | bat]** script. You can use **alarmprogram.[sh | bat]** instead of **log_full.[sh | bat]** to automate log backups.

To customize the ALARMPROGRAM scripts:

1. Change the value of ADMINMAIL to the email address of the database server administrator.
2. Change the value of PAGERMAIL to the pager service email address.
3. Set the value of the parameter MAILUTILITY with **/usr/bin/mail** for UNIX and **\$INFORMIXDIR/bin/ntmail.exe** for Windows.
4. To automatically back up logical logs as they fill, change BACKUP to yes. To stop automatic log backups, change BACKUP to any value other than yes.
5. In the ONCONFIG file, set ALARMPROGRAM to the full pathname of **alarmprogram.[sh | bat]**.
6. Restart the database server.

Alarms with a severity of 1 or 2 do not write any messages to the message log nor send email. Alarms with severity of 3 or greater send email to the database administrator. Alarms with severity of 4 and 5 also notify a pager via email.

Precautions for Foreground Operations in Alarm Scripts

To ensure continuous server availability, do not run certain foreground operations in an alarm script.

When the server invokes an alarm script, the server sometimes waits for the script to complete before proceeding. For example:

- When an alarm is invoked because of a fatal error, the server waits for the script to finish writing information to the error log.
- In certain situations when Enterprise Replication runs out of disk space, the server waits for a DBA to add more disk space before proceeding with a script.

Because the server might need to wait for the alarm program script to finish, do not run the following operations in the foreground in an alarm script:

- An onmode command that forces user connections off the server such as `onmode -u` or `onmode -yuk`. These kinds of onmode commands can cause a deadlock between the server and the alarm script because the server might wait for the alarm script to complete while the alarm script that executed the onmode command waits for the user sessions to shut down, and one of those sessions is running the alarm script itself.
- Operations that might take a long time to complete or that have a highly variable run time. Operations that take a long time to complete can cause the server to appear as if it is not responding while the operation is running.

If you need to run the above operations in an alarm script, run them in the background using one of the following operating system utilities:

On UNIX: Use the `nohup` utility. For example, `nohup onmode -yuk &` instructs `nohup` to continue running the command even if its parent terminates and the ampersand, `&`, runs the command in the background so it will not block execution of the alarm program script itself.

On Windows: Use the `start` utility with the `/B` flag. For example, `start /B onmode -yuk`.

Interpreting Error Messages

Some of the events that the database server reports to the message log cause it to invoke the alarm program. The class messages indicate the events that the database server reports.

The database server reports a nonzero exit code in the message log. In the alarm program, set the `EXIT_STATUS` variable to 0 for successful completion and to another number for a failure.

For example, if a thread attempts to acquire a lock, but the maximum number of locks that `LOCKS` specifies has already been reached, the database server writes the following message to the message log:

```

10:37:22 Checkpoint Completed: duration was 0 seconds.
10:51:08 Lock table overflow - user id 30032, rstcb 10132264
10:51:10 Lock table overflow - user id 30032, rstcb 10132264
10:51:12 Checkpoint Completed: duration was 1 seconds.

```

When the database server invokes **alarmprogram.[sh | bat]** or your alarm program, it generates a message that describes the severity and class of the event. If the severity is greater than 2, the message takes the following format:

```

Reasonably severe server event:
Severity: 3
Class ID: 21
Class msg: Database server resource overflow: 'Locks'.
Specific msg: Lock table overflow - user id 30032, rstcb 10132264
See Also:                # optional message
The following message appears at the end of each e-mailed message:
This e-mail was generated by the server ALARMPROGRAM script on
  servername
because something untoward just happened to eventname.

```

Event-Alarm Parameters

Table C-1 lists the event-alarm parameters.

Table C-1. Event-Alarm Parameters

Parameter	Meaning	Type
severity	Event severity (See Table C-2 for values.)	integer
class_id	Event class ID (See Table C-3 on page C-4 for values.)	integer
class_msg	Event class message (See Table C-3 on page C-4 for messages.)	string
specific_msg	Event specific messages	string
see_also	Event see-also file	string

Event Severity

The first parameter passed to the alarm program is the event-severity code. All events reported to the message log have one of the severity codes listed in Table C-2. Message-log events that have severity 1 do not cause the database server to invoke the alarm program unless the **ALRM_ALL_EVENTS** configuration parameter, supported by Dynamic Server, Version 10.0 or later, is enabled.

Table C-2. Event-Severity Codes

Severity	Description
1	Not noteworthy. The event (for example, date change in the message log) is not reported to the alarm program unless ALRM_ALL_EVENTS is enabled.
2	Information. No error has occurred, but some routine event completed successfully (for example, checkpoint or log backup completed).
3	Attention. This event does not compromise data or prevent the use of the system; however, it warrants attention (for example, one chunk of a mirrored pair goes down). Sends e-mail to the system administrator.
4	Emergency. Something unexpected occurred that might compromise data or access to data (assertion failure, or oncheck reports data corrupt). Take action immediately. Pages the system administrator.

Table C-2. Event-Severity Codes (continued)

Severity	Description
5	Fatal. Something unexpected occurred and caused the database server to fail. Pages the system administrator.

Event Class ID

An event class ID is an integer that the database server substitutes as the second parameter in your alarm program. Each event class ID is associated with one of the events that causes the database server to run your alarm program.

Class Message

A class message is the text of the message that the database server substitutes for the third parameter of your alarm program when an event causes the database server to run your alarm program. The class messages are different for Dynamic Server and Extended Parallel Server.

Specific Messages

The database server substitutes additional information for the fourth parameter of your alarm program. In general, the text of this message is that of the message written to the message log for the event.

See Also Paths

For some events, the database server writes additional information to a file when the event occurs. The pathname in this context refers to the pathname of the file where the database server writes the additional information.

Event Alarms on Dynamic Server

Table C-3 shows the class IDs and class messages for alarms on Dynamic Server. The first column lists the class IDs that identify each alarm and the second column lists the class messages. For more information about setting the ALARMPROGRAM parameter, which controls alarms, see “ALARMPROGRAM” on page 1-12.

Table C-3. Event Alarms on Dynamic Server

Class ID	Class Message
1	Table failure: <i>'dbsname:"owner".tabname'</i>
2	Index failure: <i>'dbsname:"owner".tabname-idxname'</i>
3	Blob failure: <i>'dbsname:"owner".tabname'</i>
4	Chunk is offline, mirror is active: <i>chunk number</i>
5	Dbospace is offline: <i>'dbospace name'</i>
6	Internal subsystem failure: <i>'message'</i>
7	Database server initialization failure
8	Physical restore failure
9	Physical recovery failure
10	Logical recovery failure

Table C-3. Event Alarms on Dynamic Server (continued)

Class ID	Class Message
11	Cannot open chunk: ' <i>pathname</i> '
12	Cannot open dbspace: ' <i>dbspace name</i> '
13	Performance improvement possible
14	Database failure. ' <i>database name</i> '
15	High-Availability Data-Replication failure
16	Backup completed: ' <i>dbspace list</i> '
17	Backup aborted: ' <i>dbspace list</i> '
18	Log backup completed: <i>log number</i>
19	Log backup aborted: <i>log number</i>
20	Logical logs are full—backup is needed
21	Database server resource overflow: ' <i>resource name</i> '
22	Long transaction detected
23	Logical log ' <i>number</i> ' complete
24	Unable to allocate memory
25	Internal subsystem initialized: ' <i>message</i> ' (starts the optical subsystem)
26	Dynamically added log file logid
27	Log file required
28	No space for log file
29	Chunk (storage) failure
29	Data capacity
29	Logical log capacity
29	Maximum locks
29	Maximum capacity
29	Maximum sessions

RS Secondary Server Event Alarms

Monitor the following event alarms that RS secondary servers trigger (see Table C-4). When each alarm is triggered, a message is written to the message log. For more information, see the chapters on event alarms and configuration parameters in the *IBM Informix Administrator's Reference*.

Table C-4. Event Alarms for RS Secondary Servers

Class ID	Severity	Class Message	Message
40	ALRM_ATTENTION	RSS <i>servername</i> log replay position is falling too far behind RSS source	This message displays when the log replay position for the RS secondary server is falling too far behind the primary. If this trend continues, the primary might not have a logical log file available when the RS secondary server requests it.
40	ALRM_ATTENTION	RSS <i>servername</i> added	The RS secondary server with name <i>servername</i> was added to the cluster.
40	ALRM_ATTENTION	RSS <i>servername</i> deleted	The RS secondary server with name <i>servername</i> was removed from the cluster.

Table C-4. Event Alarms for RS Secondary Servers (continued)

Class ID	Severity	Class Message	Message
40	ALRM_ATTENTION	HA password for RSS <i>servername</i> changed	The password to configure the RS secondary server with name <i>servername</i> was changed.
40	ALRM_ATTENTION	RSS <i>servername</i> is not acknowledging log transmissions	The RS secondary server with name <i>servername</i> is not acknowledging log transmissions. The primary server will not send any more logical log pages until an acknowledgement is received.
40	ALRM_ATTENTION	Error receiving a buffer from RSS <i>servername</i> shutting down	The primary server experienced an error receiving a message from the RS secondary server with name <i>servername</i> .

SD Secondary Server Event Alarms

Monitor the following event alarms that SD secondary servers trigger (see Table C-5). When each alarm is triggered, a message is written to the message log. For more information, see the chapters on event alarms and configuration parameters in the *IBM Informix Administrator's Reference*.

Table C-5. Event Alarms for SD Secondary Servers

Class ID	Severity	Class Message	Message
41	ALRM_ATTENTION	ERROR: Removing SDS node <i>servername</i> has timed out - removing	The SD secondary server was removed due to a time-out situation. For more information, see the SDS_TIMEOUT configuration parameter in <i>IBM Informix Administrator's Reference</i>

Appendix D. Discontinued Configuration Parameters

This section lists the discontinued and obsolete configuration parameters for Dynamic Server.

Table D-1 summarizes the discontinued parameters. Although these parameters are still supported, it is recommended that you do not use them. Remove these parameters from the ONCONFIG file before using the VPCLASS parameter.

Table D-1. Discontinued Configuration Parameters

Configuration Parameter	Reference
AFF_NPROCS	page "AFF_NPROCS (Discontinued)"
AFF_SPROC	page "AFF_SPROC (Discontinued)" on page D-2
BUFFERS	page "BUFFERS (Discontinued)" on page D-3
FAST_RESTART_CKPT_FUZZYLOG	page "FAST_RESTART_CKPT_FUZZYLOG (Discontinued)" on page D-3
FAST_RESTART_PHYSLOG	page "FAST_RESTART_PHYSLOG (Discontinued)" on page D-4
LRU_MAX_DIRTY	page "LRU_MAX_DIRTY (Discontinued)" on page D-5
LRU_MIN_DIRTY	page "LRU_MIN_DIRTY (Discontinued)" on page D-6
LRUS	page "LRUS (Discontinued)" on page D-6
NOAGE	page "NOAGE (Discontinued)" on page D-7
NUMAIOVPS	page "NUMAIOVPS (Discontinued)" on page D-7
NUMCPUVPS	page "NUMCPUVPS (Discontinued)" on page D-8
PHYSDBS	page "PHYSDBS (Discontinued)" on page D-9

Table D-2 summarizes the configuration parameters that are no longer supported.

Table D-2. Obsolete Configuration Parameters

Configuration Parameter	Reference
JDKVERSION	page "JDKVERSION (Discontinued)" on page D-5
LBU_PRESERVE	page "LBU_PRESERVE (Discontinued)" on page D-5
LOGSMAX	page "LOGSMAX (Discontinued)" on page D-5

AFF_NPROCS (Discontinued)

`onconfig.std value`
0

range of values

0 through number of CPUs in the computer

takes effect

When the database server shuts down and restarts

refer to

- Virtual-processor classes, in the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*
- "AFF_SPROC (Discontinued)"
- "VPCLASS" on page 1-109

On multiprocessor computers that support *processor affinity*, AFF_NPROCS specifies the number of CPUs to which the database server can bind CPU virtual processors. Binding a CPU virtual processor to a CPU causes the virtual processor to run exclusively on that CPU. The database server assigns CPU virtual processors to CPUs in serial fashion, starting with the processor number that AFF_SPROC specifies.

If you specify more CPU virtual processors than there are processors, the database server starts over again at the beginning. For example, if you set AFF_NPROCS to 3 and AFF_SPROCS to 5, the database server assigns two CPU virtual processors to processor 5, two CPU virtual processors to processor 6, and one CPU virtual processor to processor 7.

Important: Use VPCLASS instead of AFF_NPROCS to specify the number of CPUs. You cannot use both AFF_NPROCS and VPCLASS *cpu* in the same ONCONFIG file.

AFF_SPROC (Discontinued)

onconfig.std value

0

units CPU number

range of values

0 through (AFF_NPROCS - NUMCPUVPS + 1)

takes effect

When the database server shuts down and restarts

refer to

- Virtual-processor classes, in the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*
- "AFF_NPROCS (Discontinued)" on page D-1
- "VPCLASS" on page 1-109

On multiprocessor computers that support *processor affinity*, AFF_SPROC specifies the CPU, starting with 0, on which the database server starts binding CPU virtual processors to CPUs. The AFF_NPROCS parameter specifies the number of CPUs that the database server will use. The NUMCPUVPS parameter specifies the number of CPU virtual processors to be started, and the AFF_SPROC parameter specifies the CPU on which the first virtual processor is to start. For example, if you assign eight CPUs (AFF_NPROCS = 8), and set NUMCPUVPS to 3 and AFF_SPROC to 5, the database server binds CPU virtual processors to the fifth, sixth, and seventh CPUs.

Important: Use VPCLASS instead of AFF_SPROC to specify processor affinity. You cannot use both AFF_SPROC and VPCLASS *cpu* in the same ONCONFIG file.

BUFFERS (Discontinued)

onconfig.std *value*

UNIX: 5000Windows : 2000

units Number of buffers

range of values

For 32-bit platform on UNIX: with page size equal to 2048 bytes: 100 through 1,843,200 buffers (1843200 = 1800 * 1024)

with page size equal to 4096 bytes: 100 through 921,600 buffers (921,600 = ((1800 * 1024)/4096) * 2048)

For 32-bit platform on Windows: 100 through 524,288 buffers (524,288 = 512 * 1024)

For 64-bit platforms: 100 through $2^{31}-1$ buffers (For the actual value for your 64-bit platform, see your machine notes. The maximum number of buffers on Solaris is 536,870,912.)

takes effect

When the database server is shut down and restarted

utilities

onstat -b or **-B** (See “onstat -b: Print buffer information for buffers in use” on page 15-9.)

refer to

- Shared-memory buffer pool in the shared-memory chapter of the *IBM Informix Administrator's Guide*
- “RA_PAGES” on page 1-80
- “RA_THRESHOLD” on page 1-80
- Your *IBM Informix Performance Guide*

Note: Information that was specified with the BUFFERS configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see “BUFFERPOOL” on page 1-17.

BUFFERS specifies the maximum number of shared-memory buffers that the database server user threads have available for disk I/O on behalf of client applications. Therefore, the number of buffers that the database server requires depends on the applications. For example, if the database server accesses 15 percent of the application data 90 percent of the time, you need to allocate enough buffers to hold that 15 percent. Increasing the number of buffers can improve system performance.

In general, buffer space should range from 20 to 25 percent of physical memory. It is recommended that you calculate all other shared-memory parameters after you set buffer space (BUFFERS **system_page_size*) to 20 percent of physical memory.

FAST_RESTART_CKPT_FUZZYLOG (Discontinued)

onconfig.std *value*

The FAST_RESTART_CKPT_FUZZYLOG parameter does not need to be in the **onconfig.std** file.

range of values

0 (default) = Disable the flushing of dirty fuzzy pages to the physical log at checkpoint.

1 = Enable the flushing of dirty fuzzy pages to the physical log at checkpoint.

takes effect

At the checkpoint that occurs after the parameter is enabled. If the total number of unflushed, dirty fuzzy pages exceeds 20 percent of the total physical log space, the pages will not be written to the physical log. If server fails after this checkpoint, crash recovery receives no performance benefit.

refer to Information on fast recovery and alternative fast restart recovery options for fuzzy operations in the *IBM Informix Administrator's Guide*.

The FAST_RESTART_CKPT_FUZZYLOG parameter and the FAST_RESTART_PHYSLOG parameter enable the database server to perform physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time. You can use either parameter or both when using fuzzy checkpoints.

The database server must be online when you enable the FAST_RESTART_CKPT_FUZZYLOG parameter.

FAST_RESTART_PHYSLOG (Discontinued)

onconfig.std *value*

The FAST_RESTART_PHYSLOG parameter does not need to be in the **onconfig.std** file.

range of values

0 (default) = Disable physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery.

1 = Enable physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time.

takes effect

Immediately. If the total number of unflushed, fuzzy dirty pages exceeds 20 percent of the total physical log space, the pages will not be written to the physical log. However, if the database server fails before the next checkpoint performs, maximum fast-recovery performance does not occur because the database server did not log all of the fuzzy updates in the checkpoint intervals.

refer to Information on fast recovery and alternative fast restart recovery options for fuzzy operations in the *IBM Informix Administrator's Guide*

The FAST_RESTART_PHYSLOG parameter and the FAST_RESTART_CKPT_FUZZYLOG parameter enable the database server to perform physical logging on fuzzy checkpoints during the roll-forward (log replay) phase of recovery, thus decreasing recovery time. You can use either parameter or both when using fuzzy checkpoints.

Only use the FAST_RESTART_PHYSLOG parameter if the buffer pool is at least 25 percent larger than the physical buffer size. The buffer pool must be large enough

to hold the physical log, log pages, and other pages read during recovery. If the buffer pool is not configured correctly, fast recovery performance is compromised.

The extra physical logging that occurs when the database server uses the `FAST_RESTART_PHYSLOG` parameter affects runtime performance. If you do not want to sacrifice runtime performance or if you do not want to increase the buffer size, use the `FAST_RESTART_CKPT_FUZZYLOG` parameter to reduce some recovery time.

After enabling the `FAST_RESTART_PHYSLOG` parameter by setting it to 1, you can initiate fast recovery using the `oninit` utility. Simply execute `oninit` without any options.

The database server must be online when you enable the `FAST_RESTART_PHYSLOG` parameter.

JDKVERSION (Discontinued)

onconfig.std *value*
1.4

range of values

For this release, the valid values are 1.4, 1.3, and 1.2.

takes effect

When shared memory is initialized

JDKVERSION is the major version of the JDK or JRE release. That is, the version number does not include *x* when the version is JDK 1.4.*x*.

This parameter is required if the number of JVPs (set in `VPCLASS JVP`) is greater than 0.

LBU_PRESERVE (Discontinued)

Dynamic Server no longer supports the `LBU_PRESERVE` parameter, which reserves the last logical log for ON-Archive use. ON-Archive, which has been discontinued, was the only utility that required free log space to back up a logical log.

LOGSMAX (Discontinued)

Dynamic Server no longer supports the `LOGSMAX` parameter.

`LOGSMAX` specifies the maximum number of logical-log files for a database server instance. The database server requires at least three logical-log files for operation. The maximum number of logical logs is 32,767. The `LOGSMAX` value must be equal to or less than the highest log file number.

LRU_MAX_DIRTY (Discontinued)

onconfig.std *value*
60.00

units Percent

range of values

0 through 100 (fractional values are allowed)

takes effect

When the database server is shut down and restarted

refer to The following topics in the shared-memory chapter of the *IBM Informix Dynamic Server Administrator's Guide*

- LRU queues
- Limiting the number of pages added to the MLRU queues

Note: Information that was specified with the LRU_MAX_DIRTY configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see "BUFFERPOOL" on page 1-17.

LRU_MAX_DIRTY specifies the percentage of modified pages in the LRU queues at which the queue is cleaned. If a parameter is specified out of the range of values, then the default of 60.00 percent is set.

LRU_MIN_DIRTY (Discontinued)

onconfig.std *value*

50.00

units Percent

range of values

0 through 100 (fractional values are allowed)

takes effect

When the database server is shut down and restarted

refer to The following topics in the shared-memory chapter of the *IBM Informix Dynamic Server Administrator's Guide*:

- LRU queues
- When MLRU cleaning ends

Note: Information that was specified with the LRU_MIN_DIRTY configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see "BUFFERPOOL" on page 1-17.

LRU_MIN_DIRTY specifies the percentage of modified pages in the LRU queues at which page cleaning is no longer mandatory. Page cleaners might continue cleaning beyond this point under some circumstances. If a parameter is specified out of the range of values, then the default of 50.00 percent is set.

LRUS (Discontinued)

onconfig.std *value*

8

if not present

If MULTIPROCESSOR is set: MAX(4, num_cpu_vps) If MULTIPROCESSOR is not set: 4

units Number of LRU queues

range of values

1 through 128

takes effect

When the database server is shut down and restarted

utilities

onstat -R (see “onstat -R: Print LRU, FLRU, and MLRU queue information” on page 15-155.)

refer to

- LRU queues, in the shared-memory chapter of the *IBM Informix Dynamic Server Administrator's Guide*
- Chapter on configuration effects on memory, in your *IBM Informix Dynamic Server Performance Guide*

Note: Information that was specified with the LRUS configuration parameter prior to Version 10.0 is now specified using the BUFFERPOOL configuration parameter. For more information, see “BUFFERPOOL” on page 1-17.

LRUS specifies the number of LRU (least-recently-used) queues in the shared-memory buffer pool. You can tune the value of LRUS, in combination with the LRU_MIN_DIRTY and LRU_MAX_DIRTY parameters, to control how frequently the shared-memory buffers are flushed to disk.

Setting LRUS too high might result in excessive page-cleaner activity.

NOAGE (Discontinued)

onconfig.std value

0

range of values

0 = Use priority aging. 1 = Disable priority aging.

takes effect

When the database server shuts down and restarts

refer to

- Preventing priority aging, in the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*
- “VPCLASS” on page 1-109

Some operating systems lower the priority of processes as the processes run over a long period of time. NOAGE, when set to 1, disables *priority aging* of CPU virtual processors by the operating system. When NOAGE is set to the default of 0, the operating system might lower the priority of CPU virtual processors, as well as other processes, as they accumulate processing time. If your operating system supports priority aging, it is recommended that you set NOAGE to 1.

Important: It is recommended that you specify priority aging with the VPCLASS parameter instead of the NOAGE parameter. You cannot use both NOAGE and VPCLASS *cpu* in the same ONCONFIG file.

NUMAIOVPS (Discontinued)

onconfig.std value

None

if not present

If AUTO_AIOVPS is set to 1 (on), the number of AIO VPs initially started is equal to the number of AIO chunks, up to a maximum of 128.

If AUTO_AIOVPS is set to 0 (off), the number of AIO VPs started is equal to the greater of 6 or twice the number of AIO chunks, up to a maximum of 128.

units Number of AIO VPs

range of values

Integer between the value of 1 and 10,000, inclusive

takes effect

When the database server shuts down and restarts

utilities

onmode -p in “onmode -p: Add or remove virtual processors” on page 11-16

refer to

- Asynchronous I/O, in the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*
- “VPCLASS” on page 1-109

NUMAIOVPS specifies the number of virtual processors of the AIO class to run. Unless kernel asynchronous I/O is implemented, the AIO virtual processors perform all the database server disk I/O, other than I/O to the log files.

Important: It is recommended that you specify the number of AIO VPs with VPCLASSaio instead of NUMAIOVPS. You cannot use both NUMAIOVPS and VPCLASS aio in the same ONCONFIG file.

UNIX Only:

If your platform has kernel-asynchronous I/O (KAIO) turned on, the database server uses AIO virtual processors to perform I/O only to cooked chunks. The database server uses KAIO to perform all I/O to raw disk space and to the physical and logical logs. For details, see the machine notes.

NUMCPUVPS (Discontinued)

onconfig.std value

1

units Number of CPU VPs

range of values

1 through the number of CPUs

takes effect

When the database server shuts down and restarts

utilities

onmode -p in “onmode -p: Add or remove virtual processors” on page 11-16

refer to

- CPU virtual processors, in the chapter on virtual processors and threads in the *IBM Informix Administrator's Guide*
- “VPCLASS” on page 1-109

NUMCPUVPS specifies the number of virtual processors of the CPU class to run. CPU virtual processors run all threads that start as the result of a connection by a client application, as well as internal threads. In general, allocate only one CPU virtual processor on a single-processor computer or node. On a multiprocessor computer or node, do not allocate more CPU virtual processors than there are CPUs.

Important: It is recommended that you specify the number of CPU virtual processors with `VPCLASS cpu` instead of `NUMCPUVPS`. You cannot use both `NUMCPUVPS` and `VPCLASS cpu` in the same `ONCONFIG` file.

On UNIX, use the `onmode -p -1 CPU` command to decrease the number of CPU VPs. On Windows, you can add a CPU VP, but you cannot subtract it.

PHYSDBS (Discontinued)

onconfig.std *value*
rootdbs

if not present

The dbspace that ROOTNAME specifies

units A dbspace

range of values

Up to 128 bytes. PHYSDBS must be unique, begin with a letter or underscore, and contain only letters, numbers, underscores, or \$ characters.

takes effect

When the database server is initialized

refer to

- “ onparams -p: Change physical-log parameters” on page 13-3
- Where the physical log is located, in the chapter on what is the physical log in the *IBM Informix Administrator's Guide*
- Changing the physical-log location and size, in the chapter on managing the physical log in the *IBM Informix Administrator's Guide*

PHYSDBS specifies the name of the dbspace that contains the physical log. To reduce disk contention, you can move the physical log to a dbspace other than the root dbspace.

When you initialize disk space (`oninit -i`), the PHYSDBS value must be equal to the ROOTDBS value.

Appendix E. Error Messages

This chapter lists nonnumbered messages that are printed in the database server message log and provides corrective actions.

For information on numbered messages and the unnumbered ON-Bar messages, search for the message text in the error messages file, which is located in the subdirectory for your locale under the `$INFORMIXDIR/msg` directory. You can also search *IBM Informix Error Messages* in English at the IBM Informix Information Center at <http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp>.

Some of the messages included below might require you to contact Technical Support staff. Such messages are rarely, if ever, seen at customer locations.

For information on what the message log is, see installing and configuring the database server in the *IBM Informix Administrator's Guide*. For information on specifying the path to the message file, see "MSGPATH" on page 1-68.

How the Messages Are Ordered in This Chapter

Database server message-log messages are arranged in this chapter in alphabetical order, sorted with the following additional rules:

- The time stamp that precedes each message is ignored.
- Letter case is ignored in alphabetization.
- Spaces are ignored.
- Quotation marks are ignored.
- Leading ellipses are ignored.
- The word *the* is ignored if it is the first word in the message.
- Messages that begin with numbers or punctuation symbols appear toward the end of the list in a special section labeled "Messages: Symbols" on page E-48.
- Certain related messages are grouped together, as follows:
 - "Conversion/Reversion Messages" on page E-49
 - "Conversion and Reversion Messages for Enterprise Replication" on page E-61
 - "Dynamic Log Messages" on page E-65
 - "Sbpace Metadata Messages" on page E-67
 - "Truncate Table Messages" on page E-68

A cause and suggested corrective action for a message or group of messages follow the message text.

How to View These Messages

Use one of the following methods to view these messages:

- Online message log
 - To see the messages displayed as they occur, use the `tail -f online.log` command.
- `onstat -m` command

For more information, see “onstat -l: Print physical and logical log information” on page 15-142.

- IBM Informix Server Administrator (ISA)
For more information, see the ISA online help.

To see the error number associated with these unnumbered messages, view the **logmessage** table in the **sysmaster** database:

```
SELECT * FROM logmessage;
```

Message Categories

Four general categories of unnumbered messages exist, although some messages fall into more than one category:

- Routine information
- Assertion-failed messages
- Administrative action needed
- Unrecoverable error detected

Technical Support uses the assertion-failed messages to assist in troubleshooting and diagnostics. The information that they report often falls into the category of *unexpected events* that might or might not develop into problems caught by other error codes. Moreover, the messages are terse and often extremely technical. They might report on one or two isolated statistics without providing an overall picture of what is happening. This information can suggest to technical support possible research paths.

Messages: A-B

Aborting Long Transaction: *tx Oxn*.

Cause

The transaction spans the log space specified by transaction high-watermark (LTXHWM), and the offending long transaction is rolling back.

Action

No additional action is needed. The address of the transaction structure in shared memory is displayed as a hexadecimal value.

Affinitied VP *mm* to phys proc *nn*.

Cause

The database server successfully bound a CPU virtual processor to a physical processor.

Action

None required.

Affinity not enabled for this server.

Cause

You tried to bind your CPU virtual processors to physical processors, but the database server that you are running does not support process affinity.

Action

Set `AFF_NPROCS` to 0, or remove the affinity setting from `VPCLASS`.

Assert Failed: Error from SBSpace cleanup thread.

Cause

The sbspace cleanup thread encountered an error while cleaning up stray smart large objects.

Action

See the action suggested in the message log file.

Most of the time, running `onspaces -cl sbspacename` on the failed sbspace succeeds in cleaning up any stray smart large objects. If you encounter an unrecoverable error, contact Technical Support.

Assert Failed: Short description of what failed Who: Description of user/session/thread running at the time Result: State of the affected database server entity Action: What action the database administrator should take See Also: DUMPDIR/af.uniqid containing more diagnostics.

Cause

This message indicates an internal error.

Action

The `af.uniqid` file in the directory specified by the `ONCONFIG` parameter `DUMPDIR` contains a copy of the assertion-failure message that was sent to the message log, as well as the contents of the current, relevant structures and/or data buffers. The information included in this message is intended for Technical Support.

Begin re-creating indexes deferred during recovery.

Cause

During recovery, indexes to be created are deferred until after recovery completes. This message indicates that the database server deferred re-creating indexes and that it is now creating the indexes. During the time that the database server re-creates the indexes, it locks the affected tables with a shared lock.

Action

None required.

Building 'sysmaster' database requires ~mm pages of logical log. Currently there are nn pages available. Prepare to back up your logs soon.

Cause

You do not currently have the approximate amount of free log space necessary to complete a build of the sysmaster database.

Action

Back up your logs.

Building 'sysmaster' database...

Cause

The database server is building the sysmaster database.

Action

None required.

Messages: C

Cannot Allocate Physical-log File, mm wanted, nn available.

Cause

The database server attempted to initialize shared memory with a physical-log size that exceeds the amount of contiguous space available in the dbspace (specified as PHYSDBS in ONCONFIG). Both quantities of space, wanted and available, are expressed as kilobytes.

Action

You must either reduce the size of the physical log (specified as PHYSFILE in ONCONFIG) or change the location of the physical log to a dbspace that contains adequate contiguous space to accommodate the physical log.

Cannot alter a table which has associated violations table.

Cause

The user tried to add, drop, or modify a column in a table that has a violations table associated with it.

Action

Do not change the columns in the user table.

Cannot change to mode.

Cause

Some error during fast or full recovery has prevented the system from changing to online or quiescent mode.

Action

See previous messages in the log file for information.

Cannot Commit Partially Complete Transactions.

Cause

Transactions that drop tables or indexes do not perform the drop until a COMMIT statement is processed (with a few exceptions). In these cases, a *beginning commit* log record is written, followed by the usual commit log record. If the database server fails in between the two, the fast recovery process attempts to complete the commit the next time that you initialize the database server.

If this completion of the commit fails, the database server generates the preceding message.

Action

To determine if you need to take action, examine the logical log as described in Chapter 5, "Interpreting Logical-Log Records," on page 5-1.

Cannot create a user-defined VP class with 'SINGLE_CPU_VP' non-zero.

Cause

SINGLE_CPU_VP is set to nonzero, and **onmode** was used to create a user-defined VP class.

Action

If user-defined VP classes are necessary, stop the database server, change SINGLE_CPU_VP to zero, and restart the database server.

Cannot create violations/diagnostics table.

Cause

The user issued a START VIOLATIONS TABLE statement for a target table. The database server cannot create the violations table for this target table. Any of the following situations might be the reason for this failure:

- The target table already has a violations table.
- You specified an invalid name for the violations table in the START VIOLATIONS TABLE statement. For example, if you omit the USING clause from the statement and if the number of characters in the target table plus four characters is longer than the maximum identifier length, the generated names of the violations table exceed the maximum identifier length.
- You specified a name for the violations table in the START VIOLATIONS TABLE statement that match the names of existing tables in the database.
- The target table contains columns with the names **informix_tupleid**, **informix_optype**, or **informix_reowner**. Because these column names duplicate the **informix_tupleid**, **informix_optype**, or **informix_reowner** columns in the violations table, the database server cannot create the violations table.
- The target table is a temporary table.
- The target table is serving as a violations table for some other table.

- The target table is a system catalog table.

Action

To resolve this error, perform one of the following actions:

- If the violations table name was invalid, specify a unique name for the violations table in the USING clause of the START VIOLATIONS TABLE statement.
- If the target table contains columns with the names **informix_tupleid**, **informix_optype**, or **informix_reowner**, rename them to something else.
- Choose a permanent target table that is not a system catalog table or a violations table for some other table.

Cannot insert from the violations table to the target table.

Cause

The user has issued a statement that attempts to insert rows from the violations table into the target table. For example, the user enters the following invalid statement:

```
INSERT INTO mytable SELECT * FROM mytable_vio;
```

Also, if the target table has filtering-mode constraints, you receive this error. Extended Parallel Server does not support filtering-mode constraints.

Action

To recover from this error, perform the following actions:

- Do not use filtering constraints.
- Stop the violations table.
- Insert rows from the violations table into a temporary table, and then insert rows from the temporary table into the target table.

Cannot modify/drop a violations/diagnostics table.

Cause

The user has tried to alter or drop a table that is serving as a violations table for another table.

Action

Do not alter or drop the violations table.

Cannot Open Dbspace *nnn*.

Cause

The database server is unable to access the specified dbspace. This message indicates a problem opening the tblspace or corruption in the initial chunk of the dbspace.

Action

Verify that the device or devices that make up the chunks of this dbspace are functioning properly and that you assigned them the correct operating-system permissions (rw-rw----). You might be required to perform a data restore.

Cannot Open Logical Log.

Cause

The database server is unable to access the logical-log files. Because the database server cannot operate without access to the logical log, you must resolve this problem.

Action

Verify that the chunk device where the logical-log files reside is functioning and has the correct operating-system permissions (rw-rw----).

Cannot Open Mirror Chunk *pathname*, **errno** = *nn*.

Cause

The database server cannot open the mirrored chunk of a mirrored pair. The chunk *pathname* and the operating-system error are returned.

Action

For more information about corrective actions, see your operating-system documentation.

Cannot Open Primary Chunk *pathname*, **errno** = *nnn*.

Cause

The primary chunk of a mirrored pair cannot be opened. The chunk *pathname* and the operating-system error are returned.

Action

For more information about corrective actions, see your operating-system documentation.

Cannot Open Primary Chunk *chunkname*.

Cause

The *initial* chunk of the dbspace cannot be opened.

Action

Verify that the chunk device is running properly and has the correct operating-system permissions (rw-rw----).

Cannot open sysams in database *name*, **iserrno** *number*.

Cause

An error occurred when the database server opened the **sysams** system table.

Action

Note the error *number* and contact Technical Support.

Cannot open sysdistrib in database *name*, iserrno *number*.

Cause

An error occurred when the database server accessed the `sysdistrib` system table.

Action

Note the error *number* and contact Technical Support.

Cannot open *system_table* in database *name*, iserrno *number*.

Cause

An error occurred when the database server opened the specified system table.

Action

Note the error *number* and contact Technical Support.

Cannot open sysstrigbody in database *name*, iserrno *number*.

Cause

An error occurred when the database server accessed the `sysstrigbody` system table.

Action

Note the error *number* and contact Technical Support.

Cannot open sysstriggers in database *name*, iserrno *number*.

Cause

An error occurred when the database server accessed the `sysstriggers` system table.

Action

Note the error *number* and contact Technical Support.

Cannot open sysxdtypes in database *name*, iserrno *number*.

Cause

An error occurred while accessing the `sysxdtypes` system table.

Action

Note the error *number* and contact Technical Support.

Cannot Perform Checkpoint, shut system down.

Cause

A thread that is attempting to restore a mirrored chunk has requested a checkpoint, but the checkpoint cannot be performed.

Action

Shut down the database server.

Cannot Restore to Checkpoint.

Cause

The database server is unable to recover the physical log and thus unable to perform fast recovery.

Action

If the database server does not come online, perform a data restore from dbspace backup.

Cannot Rollback Incomplete Transactions.

Cause

Within the fast-recovery or data-restore procedure, the logical-log records are first rolled forward. Then, open transactions that have not committed are rolled back. An open transaction could fail during the rollback, leaving some of the modifications from the open transaction in place. This error does not prevent the database server from moving to quiescent or online mode, but it might indicate an inconsistent database.

Action

To determine if any action is needed, use the onlog utility to examine the logical log.

Cannot update pagezero.

Cause

A failure occurred while the database server was trying to rewrite a reserved page during the reversion process.

Action

See previous messages in the log file for information, or contact Technical Support.

Cannot update syscasts in database *name*. Iserrno *number*.

Cause

An internal error occurred while inserting data into the `syscasts` system table.

Action

Contact Technical Support..

Can't affinity VP *mm* to phys proc *nn*.

Cause

The database server supports process affinity, but the system call to bind the virtual processor to a physical processor failed.

Action

See your operating-system documentation.

Changing the sbspace minimum extent value: old value *value1*, new value *value2*.

Cause

This informational message occurs when you issue the following command:
`onspaces -ch sbspace -Df "MIN_EXT_SIZE=value1" -y`

Action

None. For more information, see "onspaces -ch: Change sbspace default specifications" on page 14-17.

Checkpoint blocked by down space, waiting for override or shutdown.

Cause

A dbspace has gone down during a checkpoint interval. The database server is configured to wait for an override when this situation occurs.

Action

Either shut down the database server or issue an **onmode -O** command to override the down dbspace. For more information on the **onmode** utility, see "In This Chapter " on page 11-1.

Checkpoint Completed: duration was *n* seconds.

Cause

A checkpoint completed successfully.

Action

None required.

Checkpoint Page Write Error.

Cause

The database server detected an error in an attempt to write checkpoint information to disk.

Action

For additional assistance in resolving this situation, contact Technical Support.

Checkpoint Record Not Found in Logical Log.

Cause

The logical log or the chunk that contains the logical log is corrupted. The database server cannot initialize.

Action

Perform a data restore from dbspace backup.

Chunk *chunkname* added to space *spacename*.

Cause

The variables in this message have the following values:

chunkname

is the name of the chunk that the database server administrator is adding.

spacename

is the name of the storage space to which the database server administrator is adding the chunk.

Action

None required.

Chunk *chunkname* dropped from space *spacename*.

Cause

The database server administrator dropped chunk *chunkname* from space *spacename*.

Action

None required.

Chunk *number nn pathname* -- Offline.

Cause

The indicated chunk in a mirrored pair has been marked with status D and taken offline. The other chunk in the mirrored pair is operating successfully.

Action

Take steps now to repair the chunk device and restore the chunk. The chunk *number* and chunk device *pathname* are displayed.

Chunk *number nn pathname* -- Online.

Cause

The indicated chunk in a mirrored pair has been recovered and is online (marked with status 0). The chunk *number* and chunk device *pathname* are displayed.

Action

None required.

The chunk *pathname* must have READ/WRITE permissions for owner and group.

Cause

The chunk *pathname* does not have the correct owner and group permissions.

Action

Make sure that you assigned the correct permissions (-rw-rw---) to the device on which the chunk is located.

The chunk *pathname* must have *owner-ID* and *group-ID* set to informix.

Cause

The chunk *chunkname* does not have the correct owner and group ID.

Action

Make sure the device on which the chunk is located has the ownership. On UNIX, both owner and group should be **informix**. On Windows, the owner must be a member of the **Informix-Admin** group.

The chunk *pathname* will not fit in the space specified.

Cause

The chunk *pathname* does not fit in the space that you specified.

Action

Choose a smaller size for the chunk, or free space where the chunk is to be created.

Cleaning stray LOs in sbspace *sbspacename*.

Cause

The database server administrator is running **onspaces -cl sbspacename**.

Action

None required.

Completed re-creating indexes.

Cause

The database server finished re-creating the deferred indexes.

Action

None required.

Configuration has been grown to handle up to *integer* chunks.

Cause

The database server administrator increased the number of chunks to the specified value by changing CONFIGSIZE or setting MAX_CHUNKS to a higher value.

Action

None required. The change was successful.

Configuration has been grown to handle up to *integer* dbslices.

Cause

The database server administrator increased the number of dbslices to the specified value by changing CONFIGSIZE or setting MAX_DBSLICES to a higher value.

Action

None required. The change was successful.

Configuration has been grown to handle up to *integer* dbspaces.

Cause

The database server administrator increased the number of dbspaces to the specified value by changing CONFIGSIZE or setting MAX_DBSPACES to a higher value.

Action

None required. The change was successful.

Continuing Long Transaction (for COMMIT): *tx 0xn*.

Cause

The logical log has filled beyond the long-transaction high-watermark (LTXHWM), but the offending long transaction is in the process of committing. In this case, the transaction is permitted to continue writing to the logical log and is not rolled back. The address of the transaction structure in shared memory is displayed as hexadecimal value *tx 0xn*.

Action

None required.

Could not disable priority aging: errno = *number*.

Cause

An operating-system call failed while it was trying to disable priority aging for the CPU virtual processor. The system error *number* associated with the failure is returned.

Action

See your operating-system documentation.

Could not fork a virtual processor: errno = *number*.

Cause

The fork of a virtual processor failed. The database server returns the operating-system error *number* associated with the failure.

Action

For information on determining the maximum number of processes available per user and for the system as a whole, refer to your operating-system documentation.

Create_vp: cannot allocate memory.

Cause

The database server cannot allocate new shared memory.

Action

The database server administrator must make more shared memory available. This situation might require increasing SHMTOTAL or reconfiguring the operating system. This message is usually accompanied by other messages that give additional information.

Messages: D-E-F

Dataskip is OFF for all dbspaces.

Cause

Informational.

Action

None required.

Dataskip is ON for all dbspaces.

Cause

Informational.

Action

None required.

Dataskip is ON for dbspaces: *dbspacelist*.

Cause

Informational; DATASKIP is ON for the specified dbspaces.

Action

None required.

Dataskip will be turned {ON|OFF} for *dbspacename*.**Cause**

Informational; DATASKIP is ON or OFF for the specified dbspace.

Action

None required.

DBSERVERALIASES exceeded the maximum limit of 32**Cause**

The limit of 32 aliases was reached.

Action

Nothing. Only the first 32 will be used.

DBSPACETEMP internal list not initialized, using default.**Cause**

An error occurred while initializing a user-specified DBSPACETEMP list. Typically this condition is due to a memory-allocation failure.

Action

Check for accompanying error messages.

The DBspace/BLOBspace *spacename* is now mirrored.**Cause**

You successfully added mirroring to the indicated storage space.

Action

None required.

The DBspace/BLOBspace *spacename* is no longer mirrored.**Cause**

You have ended mirroring for the indicated storage space.

Action

None required.

Dbospace *dbspacename* for Physical-log File not found.

Cause

The dbospace *dbspacename* specified by the PHYSDBS configuration parameter does not exist. As a consequence, the database server cannot complete initialization.

Action

Use a dbospace known to exist.

***devname*: write failed, file system is full.**

Cause

Because the file system *devname* is full, the write failed.

Action

Free some space in *devname*.

Dropping temporary tblspace *0xn*, recovering *nn* pages.

Cause

During shared-memory initialization, the database server routinely searches for temporary tables that are left without proper cleanup. If the database server finds a temporary table, it drops the table and recovers the space. The database server located the specified temporary tblspace and dropped it. The value *0xn* is the hexadecimal representation of the tblspace number.

Action

None required.

Dynamically allocated new shared memory segment (size *nnnn*).

Cause

This status message informs you that the database server successfully allocated a new shared-memory segment of size *nnnn*.

Action

None required.

ERROR: NO "wait for" locks in Critical Section.

Cause

The database server does not permit a thread to own locks that might have to wait while that thread is within a critical section. Any such lock request is denied, and an ISAM error message is returned to the user.

Action

The error reported is an internal error. Contact IBM Informix Technical Support.

Error building sysmaster database. See *outfile*.

Cause

Errors were encountered in building the sysmaster database. The file *outfile* contains the result of running the script *buildsmi*.

Action

See the file *outfile*.

Error in dropping system defined type.

Cause

An internal error occurred while updating either the **sysxdtypes**, **sysctddesc**, or **sysxdtpeauth** system table.

Action

Contact Technical Support.

Error in renaming systdist.

Cause

An internal error occurred while trying to find and rename the **Informix.systdist** SPL routine.

Action

Contact Technical Support.

Error removing sysdistrib row for *tabid* = *tabid*, *colid* = *colid* in database *name*. *iserrno* = *number*

Cause

An error occurred while updating the **sysdistrib** system table.

Action

Note the error *number* and contact Technical Support.

Error writing *pathname* *errno* = *number*.

Cause

The operating system cannot write to *pathname*. *Number* is the number of the operating-system error that was returned.

Action

Investigate the cause of the operating-system error. Usually it means that no space is available for the file. It might also mean that the directory does not exist or that no write permissions exist.

Error writing shmemp to file *filename* (error). Unable to create output file *filename* errno=*mm*.Error writing *filename* errno=*nn*.

Cause

The database server detected an error in an attempt to write shared memory to *filename*. The first message is followed by one of the next two. Either the attempt failed because the output file could not be created or because the contents of shared memory could not be written. The error refers to the operating-system error that prompted the attempted write of shared memory to a file. The value of *nn* is the operating-system error.

Action

See your operating-system documentation.

Fail to extend physical log space.

Cause

The attempt to extend the physical log space failed. Either the path does not exist or the permissions are incorrect.

Action

Use a path that exists. Check permissions on the current working directory. You or the system administrator must give your group execute permission on the current working directory. After your group has been given permission, retry the operation that generated this message.

Fatal error initializing CWD string. Check permissions on current working directory. Group *groupname* must have at least execute permission on '.'.

Cause

Group *groupname* does not have execute permission for the current working directory.

Action

Check permissions on the current working directory. You or the system administrator must give your group execute permission on the current working directory. After your group has been given permission, retry the operation that generated this message.

The following tables have outstanding old version data pages due to an In-Place Alter Table. Perform UPDATE *tablename* SET *column* = *column* WHERE 1=1; to clear these pages from the following tables.

Cause

Reversion to a previous version of the database server has been attempted while an in-place ALTER TABLE is in progress. The previous versions of the database server cannot handle tables that have multiple schemas of rows in them.

Action

Force any in-place alters to complete by updating the rows in the affected tables before you attempt to revert to a previous version of the database server. To do this, create a dummy update in which a column in the table is set to its own value, forcing the row to be updated to the latest schema in the process without actually changing column values. Rows are always altered to the latest schema, so a single pass through the table that updates all rows completes all outstanding in-place alters.

Fragments *dbspacename1 dbspacename2* of table *tablename* set to non-resident.

Cause

The specified fragments of *tablename* either have been set to nonresident by the SET TABLE statement.

Action

None required.

Forced-resident shared memory not available.

Cause

The database server port for your computer does not support forced-resident shared memory.

Action

None required.

Freed *mm* shared-memory segment(s) *number* bytes.

Cause

The database server sends this message to the message log after you run the **-F** option of the **onmode** utility to free unused memory. The message informs you of the number of segments and bytes that the database server successfully freed.

Action

None required.

Messages: G-H-I

gcore *pid*; mv *core.pid* *dir/core.pid*.ABORT.

Cause

This status message during a database server failure provides the name and place of each core file associated with the virtual processors.

Action

None required.

I/O *function* chunk *mm*, pagenum *nn*, pagecnt *aa* --> errno = *bb*.

Cause

An operating-system error occurred during an attempt to access data from disk space. The operating-system function that failed is defined by *function*. The chunk number and physical address of the page where the error occurred are displayed as integers. The *pagecnt* value refers to the number of pages that the thread was attempting to read or write. If an *errno* value is displayed, it is the number of the operating-system error and might explain the failure. If *function* is specified as *bad request*, some unexpected event caused the I/O attempt on an invalid chunk or page.

Action

If the chunk status changes to D, or down, restore the chunk from its mirror or repair the chunk. Otherwise, perform a data restore.

I/O error, *primary/mirror* Chunk *pathname* -- Offline (*sanity*).

Cause

The database server detected an I/O error on a primary or mirror chunk with *pathname*. The chunk was taken offline.

Action

Check that the device on which the chunk was stored is functioning as intended.

Deleted Indexes idx1 and idx 2 error message

Informix *database_server* Initialized - Complete Disk Initialized.

Cause

Disk space and shared memory have been initialized. Any databases that existed on the disk before the initialization are now inaccessible.

Action

None required.

Informix *database_server* Initialized - Shared Memory Initialized.

Cause

Shared memory has been initialized.

Action

None required.

Informix *database_server* Stopped.

Cause

The database server has moved from quiescent mode to offline mode. The database server is offline.

Action

None required.

ERROR: Insufficient available disk in the root dbspace to increase the entire Configuration save area.

Cause

The user attempted to increase the number of storage objects to a specific value by changing CONFIGSIZE or setting MAX_DBSPACES, MAX_DBSLICES, or MAX_CHUNKS to a higher value, but the database server did not have enough rootspace for the increased number of storage objects. A storage object might be a dbspace, dbslice, or chunk.

Action

Increase the size of the root dbspace or reset CONFIGSIZE, MAX_DBSPACES, MAX_DBSLICES, or MAX_CHUNKS to a lower value and restart the database server. For example, if you set MAX_CHUNKS to 32,768, but the root dbspace did not have enough space, set MAX_CHUNKS to a lower value.

Insufficient available disk in the root dbspace for the CM save area. Increase the size of the root dbspace in the ONCONFIG file and reinitialize the server.

Cause

The cause might be one of the following:

- The user attempted to increase the number of storage objects to a specific value by changing CONFIGSIZE or setting MAX_DBSPACES, MAX_DBSLICES, or MAX_CHUNKS to a higher value, but the database server did not have enough rootspace for the increased number of storage objects. A storage object might be a dbspace, dbslice, or chunk.
- The user converted to a database server version that requires slightly more rootspace, but it is not available (this case is unlikely).

Action

Take one of the following actions:

- Increase the size of the root dbspace or reset CONFIGSIZE, MAX_DBSPACES, MAX_DBSLICES, or MAX_CHUNKS to a lower value and restart the database server. For example, if you set MAX_DBSPACES to 32,768 but the root dbspace did not have enough space, set MAX_DBSPACES to a lower value.
- Increase the size of the root dbspace and reinitialize the database server.

Internal overflow of shmid's, increase system max shared memory segment size.

Cause

The database server was initializing shared memory when it ran out of internal storage for the shared-memory IDs associated with this segment.

Action

Increase the value of your maximum kernel shared-memory segment size, usually SHMMAX. For more information, see your operating-system documentation.

Messages: J-K-L-M

Listener-thread err = *error_number*: *error_message*.

Cause

A listener thread has encountered an error. This message displays the error number and message text.

Action

For the cause and corrective action, see the IBM Informix Information Center at <http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp>.

Lock table overflow - user id *mm* session id *nn*.

Cause

A thread attempted to acquire a lock when no locks were available. The user ID and session ID are displayed.

Action

Increase the LOCKS configuration parameter, and initialize shared memory.

Logical-log File not found.

Cause

The checkpoint record in the root dbspace reserved page is corrupted.

Action

Perform a data restore from dbspace backup.

Logical Log *nn* Complete.

Cause

The logical-log file identified by log-ID number *nn* is full. The database server automatically switches to the next logical-log file in the sequence.

Action

None required.

Logical logging *vberror* for *type:subtype* in (*failed_system*).

Cause

Logging failed. The log record that caused the error is identified as follows:

type Is the logical-log record type.

subtype
Is the logging subsystem.

failed_system
Is the name of an internal function that indicates what system failed to log.

Action

Contact Technical Support.

Log Record: **log = ll, pos = 0xn, type = type:subtype(snum), trans = xx**

Cause

The database server detected an error during the rollforward portion of fast recovery or logical-log restore.

The log record that caused the error is identified as follows:

ll Is the logical-log ID where the record is stored.

0xn Is the hexadecimal address position within the log.

type Is the logical-log record type.

subtype
Is the logging subsystem.

snum Is the subsystem number.

xx Is the transaction number that appears in the logical log.

Action

Contact Technical Support.

Log record (*type:subtype*) at log *nn, 0xn* was not undone.

Cause

A log undo failed because a log is corrupt.

The log record that caused the error is identified as follows:

type Is the logical-log record type.

subtype
Is the logging subsystem.

nn Is the logical-log ID where the record is stored.

0xn Is the hexadecimal address position within the log.

Action

To determine if any action is needed, use the onlog utility to examine the logical log. Contact Technical Support.

Log record (*type:subtype*) failed, partnum *pnum* row *rid* iserrno *num*.

Cause

A logging failure occurred.

The log record that caused the error is identified as follows:

type Is the logical-log record type.

subtype
Is the logging subsystem.

pnum Is the part number.

rid Is the row ID.

num Is the iserror number.

Action

Contact Technical Support.

Log record (*type:subtype*) in log *nn*, offset *0xn* was not rolled back.

Cause

A log undo failed because a log is corrupt.

The log record that caused the error is identified as follows:

type Is the logical-log record type.

subtype
Is the logging subsystem.

log Is the logical-log ID where the record is stored.

offset Is the hexadecimal address position within the log.

Action

To determine if any action is needed, use the onlog utility to examine the logical log. Contact Technical Support.

Logical Recovery allocating *nn* worker threads *thread_type*.

Cause

The database server determined the number of worker threads that will be used for parallel recovery. The variable *thread_type* can assume the values ON_RECVRY_THREADS or OFF_RECVRY_THREADS.

Action

This status message requires no action. If you want a different number of worker threads allocated for parallel recovery, change the value of the ONCONFIG configuration parameter ON_RECVRY_THREADS or OFF_RECVRY_THREADS.

Logical Recovery Started.

Cause

Logical recovery began.

Action

This status message requires no action.

Maximum server connections *number*.

Cause

Outputs with each checkpoint message to indicate the maximum number of concurrent connections to the database server since the last restart.

Action

This message helps the customer track license usage to determine when more licenses need to be purchased. For assistance, Contact Technical Support.

Memory allocation error.

Cause

The database server ran out of shared memory.

Action

Take one of the following actions:

1. Increase swap space on the computer.
2. Check kernel shared-memory parameters for limits on shared memory.
3. Decrease the size of the memory allocated, with the **buffers** field in the BUFFERPOOL configuration parameter.
4. Increase the virtual-memory size (SHMVIRTSIZE), the size of the added segments, (SHMADD), or your total shared-memory size (SHMTOTAL).

Mirror Chunk *chunkname* added to space *spacename*. Perform manual recovery.

Cause

Fast recovery, full recovery, or an HDR secondary has recovered the add of a mirror chunk. It does not perform automatic mirror recovery, however. The administrator must do this.

Action

Use either the **onspaces** utility or ON-Monitor to attempt to recover the mirror chunks.

Mixed transaction result. (*pid=nn user=userid*).

Cause

You receive this message only when more than one database server is involved in a transaction. This message indicates that a database server, after preparing a transaction for commit, heuristically rolled back the transaction, and the global transaction completed inconsistently. The *pid* value is the user-process identification number of the coordinator process. The value of *user* is the user ID associated with the coordinator process.

Action

See the information on recovering manually from failed two-phase commit in your *IBM Informix Administrator's Guide*.

mt_shm_free_pool: pool *Oxn* has blocks still used (id *nn*).

Cause

An internal error occurred during a pool deallocation because blocks are still associated with the pool.

Action

Contact Technical Support.

mt_shm_init: can't create *resident/virtual* segment.

Cause

The causes for the failure to create the resident or virtual segment are as follows: (1) the segment size is less than the minimum segment size; (2) the segment size is larger than the maximum segment size; (3) allocating another segment would exceed the allowable total shared-memory size; or (4) a failure occurred while the database server was trying to allocate the segment.

Action

If you suspect that this error was generated because of item 1 or 2 in the preceding paragraph, Contact Technical Support. To correct item 3, increase the SHMTOTAL value in your ONCONFIG configuration file. For additional information about errors generated because of item 4, see your logical-log file.

mt_shm_remove: WARNING: may not have removed all/correct segments.

Cause

When the operating system tried to remove the shared-memory segments associated with the database server, the last segment did not equal the last segment registered internally. This situation is probably due to the unexpected failure of the database server.

Action

Remove any segments that were not cleaned up.

Messages: N-O-P

Newly specified value of *value* for the pagesize in the configuration file does not match older value of *value*. Using the older value.

Cause

This message displays upon database server restart. The PAGESIZE value changed in the ONCONFIG file after the database server was initialized.

Action

The database server uses the older PAGESIZE value.

Not enough main memory.

Cause

The database server detected an error in an attempt to acquire more memory space from the operating system.

Action

For more information about shared-memory configuration and management, refer to your operating-system documentation.

Not enough logical-log files, Increase LOGFILES.

Cause

During a data restore, the value of the LOGFILES configuration must always be greater than or equal to the total number of logical-log files. At some point during the restore, the number of logical-log files exceeded the value of LOGFILES.

Action

Increase the value of LOGFILES in ONCONFIG.

Not enough physical procs for affinity.

Cause

The ONCONFIG parameters AFF_NPROCS and AFF_SPROC are not correctly set. AFF_SPROC plus AFF_NPROCS is greater than the number of physical processors on your computer or node.

Action

Reset AFF_NPROCS and AFF_SPROC, such that the value AFF_SPROC plus value of AFF_NPROCS is less than or equal to the number of physical processors.

The number of configured CPU poll threads exceeds NUMCPUVPS.

Cause

The number of in-line poll threads that you specified in the ONCONFIG configuration file exceeds the number of CPU virtual processors.

Action

Reduce the number of in-line poll threads to be less than or equal to the number of CPU virtual processors.

onconfig parameter *parameter* modified from *old_value* to *new_value*.

Cause

When the database server shared memory is reinitialized, this message documents any changes that occurred since the last initialization.

Action

None required.

oninit: Cannot have SINGLE_CPU_VP non-zero and number of CPU VPs greater than 1.

Cause

The ONCONFIG file contains VPCLASS cpu with a num= value greater than 1 and a nonzero value for SINGLE_CPU_VP. SINGLE_CPU_VP must be 0 (or omitted) when there are more than 1 CPU VPs.

Action

Correct the ONCONFIG file and restart the database server.

oninit: Cannot have SINGLE_CPU_VP non-zero and user-defined VP classes.

Cause

The ONCONFIG file contains a user-defined VPCLASS as well as a nonzero value for SINGLE_CPU_VP. SINGLE_CPU_VP must be 0 (or omitted) when the ONCONFIG file contains a user-defined VPCLASS.

Action

Correct the ONCONFIG file and restart the database server.

oninit: Cannot mix VPCLASS cpu and NUMCPUVPS, AFF_SPROC, AFF_NPROCS, or NOAGE parameters.

Cause

The ONCONFIG file contains both VPCLASS cpu and one or more of the other listed parameters. It cannot contain both.

Action

Correct the ONCONFIG file and restart the database server.

oninit: Cannot mix VPCLASS aio and NUMAIOVPS parameters.

Cause

The ONCONFIG file contains both VPCLASS aio and NUMAIOVPS. It cannot contain both.

Action

Correct the ONCONFIG file and restart the database server.

oninit: Fatal error in initializing ASF with 'ASF_INIT_DATA' flags asfcode = '25507'.

Cause

The **nettype** value specified in the **sqlhosts** file or registry for the database server is invalid or unsupported, or the **servicename** specified in the **sqlhosts** file or registry for the database server is invalid.

Action

Check the **nettype** and **servicename** values in the **sqlhosts** file or registry for each DBSERVERNAME and for the DBSERVERALIASES. Check the **nettype** value in each NETTYPE parameter in the ONCONFIG file.

oninit: invalid or missing name for Subsystem Staging Blobspace.

Cause

You set the configuration parameter STAGEBLOB to a blobspace that does not exist.

Action

Use the **-d** option of **onspaces** to create the blobspace specified in STAGEBLOB, and restart the database server.

Cannot alter a table which has associated violations table.

Cause

The user tried to add, drop, or modify a column in a table that has a violations table associated with it.

Action

Do not change the columns in the user table.

oninit: Too many VPCLASS parameters specified.

Cause

Too many VPCLASS parameter lines have been specified in the ONCONFIG file.

Action

Reduce the number of VPCLASS lines, if possible. If not possible, contact Technical Support.

oninit: VPCLASS *classname* bad affinity specification.

Cause

The affinity specification for the VPCLASS line is incorrect. Affinity is specified as a range:

For m , use processor m .

For m to n , use processors in the range m to n inclusive, where $m \leq n$, $m \geq 0$, and $n \geq 0$.

Action

Correct the VPCLASS parameter in the ONCONFIG file and restart the database server.

oninit: VPCLASS *classname* duplicate class *name*.

Cause

The VPCLASS *classname* in the ONCONFIG file has a duplicate name. VP class names must be unique.

Action

Correct the duplicate name and restart the database server.

oninit: VPCLASS *classname* illegal option.

Cause

One of the fields in the VPCLASS *classname* parameter is illegal.

Action

Correct the parameter in the ONCONFIG file and restart the database server.

oninit: VPCLASS *classname* maximum number of VPs is out of the range 0-10000.

Cause

The maximum number of VPs specified by a VPCLASS parameter line must be in the range 1 to 10,000.

Action

Correct the value and restart the database server.

oninit: VPCLASS *classname* name is too long. Maximum length is *maxlength*.

Cause

The length of the name field in VPCLASS *classname* is too long.

Action

Choose a shorter class name, correct the ONCONFIG file, and restart the database server.

oninit: VPCLASS *classname* number of VPs is greater than the maximum specified.

Cause

The initial number of VPs specified by a VPCLASS parameter is greater than the maximum specified by the same VPCLASS parameter.

Action

Correct the VPCLASS parameter and restart the database server.

oninit: VPCLASS *classname* number of VPs is out of the range 0-10000.

Cause

The initial number of VPs specified by a VPCLASS parameter line must be in the range 1 to 10,000.

Action

Correct the value and restart the database server.

onmode: VPCLASS *classname* name is too long. Maximum length is *maxlength*.

Cause

The name of a dynamically added VP class that **onmode -p** specifies is too long.

Action

Choose a shorter name, and retry the **onmode -p** command.

Optical Subsystem is running.

Cause

You set the value of the STAGEBLOB parameter in the configuration file, and the database server is communicating properly with the optical-storage subsystem.

Action

No action is required.

Optical Subsystem is not running.

Cause

You set the value of the STAGEBLOB parameter in the configuration file, but the database server cannot detect the existence of the optical-storage subsystem.

Action

Check that the optical subsystem is online.

Optical Subsystem STARTUP Error.

Cause

The database server detects that the optical-storage subsystem is running, but the database server cannot communicate with it properly.

Action

Check your optical subsystem for errors.

Online Mode.

Cause

The database server is in online mode. Users can access all databases

Action

This status message requires no action.

onspaces: unable to reset dataskip.

Cause

This error message comes from the **onspaces** utility. For some reason, the utility cannot change the specification of DATASKIP (ON or OFF) across all dbspaces in the database server instance.

Action

You are unlikely to receive this message. If the error persists after you restart the database server, Contact Technical Support.

Open transaction detected when changing log versions.

Cause

The database server detected an open transaction while it was trying to convert the data from a previous version of the database server.

Action

Conversion is not allowed unless the last record in the log is a checkpoint. You must restore the previous version of the database server, force a checkpoint, and then retry conversion.

Out of message shared memory.

Cause

The database server could not allocate more memory for the specified segment.

Action

For additional information, see the log file.

Out of resident shared memory.

Cause

The database server could not allocate more memory for the specified segment.

Action

For additional information, see the log file.

Out of virtual shared memory.

Cause

The database server could not allocate more memory for the specified segment.

Action

For additional information, see the log file.

PANIC: Attempting to bring system down.

Cause

A fatal database server error occurred.

Action

See the error that caused the panic and attempt the corrective action suggested by the error message. For additional information that might explain the failure, refer also to other messages in the message-log file.

Participant site *database_server* heuristically rolled back.

Cause

A remote site rolled back a transaction after it reached the prepared-for-commit phase.

Action

You might need to roll back the transaction on other sites and then restart it.

Physical recovery complete: *number pages examined*, *number pages restored*.

Cause

This message displays during fast recovery. The *number of pages examined* indicates the number of page images that exist in the physical log. The *number of pages restored* indicates the actual number of pages that are restored from the physical log. The number of pages restored is always less than or equal to the number examined.

The database server might physically log a page image multiple times between checkpoints. Physical recovery restores only the first logged page image.

If a page stays in the memory buffer pool, the database server physically logs it once per checkpoint, and stores one page image in the physical log. If the buffer pool is too small, a page that is being updated many times might get forced out of the buffer pool to disk and then brought back into memory for the next update. Each time the page is brought into memory, it is physically logged again, resulting in duplicate page images in the physical log.

Action

If the *number of pages examined* is much larger than the *number of pages restored*, increase the size of the buffer pool to reduce the number of duplicate before-images. For more information, see the *IBM Informix Performance Guide*.

Physical recovery started at page (*chunk:offset*).

Cause

This message displays during fast recovery. *Chunk* is the number of the chunk that contains the physical log. *Offset* is the page offset of the start of the physical log entries. Physical recovery begins restoring pages from that point.

Action

No action required. For information on fast recovery, see the *IBM Informix Administrator's Guide*.

Portions of partition partnum of table tablename in database dbname were not logged. This partition cannot be rolled forward.

Cause

Light appends occurred to the operational table since the last backup.

Action

If you want full access to data in this table, you need to alter the table to raw and then to the desired table type. This alter operation removes inconsistencies in the table that resulted from replaying non-logged operations such as light appends.

Possible mixed transaction result.

Cause

This message indicates that error -716 has been returned. Associated with this message is a list of the database servers where the result of a transaction is unknown.

Action

For information on determining if a transaction was implemented inconsistently, see the *IBM Informix Administrator's Guide*.

Prepared participant site *server_name* did not respond.

Cause

Too many attempts were made to contact remote site *server_name*. After several timeout intervals were met, the site was determined to be down.

Action

Verify that the remote site is online and that it is correctly configured for distributed transactions. Once the remote site is ready, reinitiate the transaction.

Prepared participant site *server_name* not responding.

Cause

The database server is attempting to contact remote site *server_name*. For some unknown reason, the database server cannot contact the remote site.

Action

Verify that the remote site is online and that it is correctly configured for distributed transactions.

Messages: Q-R-S

Quiescent Mode.

Cause

The database server has entered quiescent mode from some other state. On UNIX, only users logged in as **informix** or as **root** can interact with the database server. On Windows, only members of the **Informix-Admin** group can interact with the database server. No user can access a database.

Action

None required.

Read failed. Table *name*, Database *name*, **iserrno = number**

Cause

An error occurred reading the specified system table.

Action

Note the error number and contact Technical Support.

Recovery Mode.

Cause

The database server entered the recovery mode. No user can access a database until recovery is complete.

Action

None required.

Recreating index: '*dbname:"owner".tablename-idxname*'.

Cause

This message indicates which index is currently being re-created.

Action

None required.

Rollforward of log record failed, **iserrno = nn**.

Cause

The message appears if, during fast recovery or a data restore, the database server cannot roll forward a specific logical-log record. The database server might be able to change to quiescent or online mode, but some inconsistency could result. For further information, see the message that immediately precedes this one. The *iserrno* value is the error number.

Action

Contact IBM Informix Technical Support.

Root chunk is full and no additional pages could be allocated to chunk descriptor page.

Cause

The root chunk is full.

Action

To free space in the root chunk, take one of the following actions:

- Drop and re-create the **sysmaster** database.
- Move user tables from the root dbspace to another dbspace.
- Refragment tables.

scan_logundo: subsys *ss*, type *tt*, iserrno *ee*.

Cause

A log undo failed because log type *tt* is corrupt.

The variables in this message have the following values:

ss Is the subsystem name.

tt Is the logical-log record type.

ee Is the iserror number.

Action

Examine the logical log with the onlog utility to determine if any action is needed.
Contact Technical Support.

Session completed abnormally. Committing *tx id 0xm*, flags *0xn*.

Cause

Abnormal session completion occurs only when the database server is attempting to commit a transaction that has no current owner, and the transaction develops into a long transaction. The database server forked a thread to complete the commit.

Action

None required.

Session completed abnormally. Rolling back *tx id 0xm*, flags *0xn*.

Cause

Abnormal session completion occurs only when the database server is attempting to commit a distributed transaction that has no current owner, and the transaction

develops into a long transaction. The database server forked a thread that rolled back the transaction.

Action

None required.

semctl: errno = nn.

Cause

When the database server initialized a semaphore, an error occurred. The operating-system error is returned.

Action

See your operating-system documentation.

semget: errno = nn.

Cause

An allocation of a semaphore set failed. The operating-system error is returned.

Action

See your operating-system documentation.

shmat: some_string os_errno: os_err_text.

Cause

An attempt to attach to a shared-memory segment failed. The system error number and the suggested corrective action are returned.

Action

Review the corrective action (if given), and determine if it is reasonable to try. For more information, refer to your operating-system documentation.

shmctl: errno = nn.

Cause

An error occurred while the database server tried to remove or lock a shared-memory segment. The operating-system error number is returned.

Action

See your operating-system documentation.

shmdt: errno = nn.

Cause

An error occurred while the database server was trying to detach from a shared-memory segment. The operating-system error number is returned.

Action

See your operating-system documentation.

shmem sent to *filename*.

Cause

The database server wrote a copy of shared memory to the specified file as a consequence of an assertion failure.

Action

None.

shmget: *some_str* os_errno: key *shmkey*: *some_string*.

Cause

Either the creation of a shared-memory segment failed, or an attempt to get the shared-memory ID associated with a certain key failed. The system error number and the suggested corrective action are returned.

Action

Consult your operating-system documentation.

Shutdown (onmode -k) or override (onmode -O).

Cause

A dbspace has gone down during a checkpoint interval. The database server is configured to wait for an override when this situation occurs.

When the checkpoint actually happens, the following message appears: Checkpoint blocked by down space, waiting for override or shutdown.

Action

Either shut down the database server or issue an **onmode -O** command to override the down dbspace. For more information on the **onmode** utility, see "In This Chapter " on page 11-1.

Shutdown Mode.

Cause

The database server is in the process of moving from online mode to quiescent mode.

Action

None required.

Space *spacename* added.

Cause

The database server administrator added a new storage space *spacename* to the database server.

Action

None required.

Space *spacename* dropped.

Cause

The database server administrator dropped a storage space *spacename* from the database server.

Action

None required.

Space *spacename* -- Recovery Begins(*addr*).

Cause

This informational message indicates that the database server is attempting to recover the storage space.

The variables in this message have the following values:

spacename

Is the name of the storage space that the database server is recovering.

addr

Is the address of the control block.

Action

None required.

Space *spacename* -- Recovery Complete(*addr*).

Cause

This informational message indicates that the database server recovered the storage space.

The variables in this message have the following values:

spacename

Is the name of the storage space that the database server has recovered.

addr

Is the address of the control block.

Action

None required.

Space *spacename* -- Recovery Failed(*addr*).

Cause

This informational message indicates that the database server was unable to recover the storage space.

The variables in this message have the following values:

spacename

Is the name of the storage space that the database server failed to recover.

addr

Is the address of the control block.

Action

None required.

sysmaster database built successfully.

Cause

The database server successfully built the sysmaster database.

Action

None required.

Successfully extend physical log space

Cause

The physical log space was successfully extended to the file *plog_extend.servernum* under the designated path.

Action

None required.

Messages: T-U-V

This ddl operation is not allowed due to deferred constraints pending on this table and dependent tables.

Cause

This error is returned when you attempt to start a violations table and constraints are in deferred mode.

Note: No error is returned if you start a violations table and then later set the constraints to deferred. However, the violations get undone immediately rather than written into the deferred constraint buffer. For more information, see the *IBM Informix Guide to SQL: Syntax*.

Action

If you would like to start a violations table, you must either change the constraint mode to immediate or commit the transaction.

This type of space does not accept log files.

Cause

Adding a logical-log file to a blobspace or sbspace is not allowed.

Action

Add the logical-log file to a dbspace. For more information, see “ onparams -a -d *dbspace*: Add a logical-log file” on page 13-2.

TIMER VP: Could not redirect I/O in initialization, errno = nn.

Cause

The operating system could not open the null device or duplicate the file descriptor associated with the opening of that device. The system error number is returned.

Action

See your operating-system documentation.

Too Many Active Transactions.

Cause

During a data restore, there were too many active transactions. At some point during the restore, the number of active transactions exceeded 32 kilobytes.

Action

None.

Too many violations.

Cause

The number of violations in the diagnostics table exceeds the limit that is specified in the MAX VIOLATIONS clause of the START VIOLATIONS TABLE statement. When a single statement on the target table (such as an INSERT or UPDATE statement) inserts more records into the violations table than the limit that is specified by the MAX VIOLATIONS clause, this error is returned to the user who issued the statement on the target table.

This MAX VIOLATIONS limit applies to each coserver. For example, if you reach the MAX VIOLATIONS limit on coserver 2, you can continue to issue statements that violate rows on other coservers until you reach the MAX VIOLATIONS limit.

Action

To resolve this error, perform one of the following actions:

- Omit the MAX VIOLATIONS clause in the START VIOLATIONS TABLE statement when you start a violations table. Here, you are specifying no limit to the number of rows in the violations table.
- Set MAX VIOLATIONS to a high value.

Transaction Not Found.

Cause

The logical log is corrupt. This situation can occur when a new transaction is started, but the first logical-log record for the transaction is not a BEGWORK record.

Action

Contact Technical Support.

Transaction heuristically rolled back.

Cause

A heuristic decision occurred to roll back a transaction after it completed the first phase of a two-phase commit.

Action

None required.

Transaction table overflow - user id *nn*, process id *nn*.

Cause

A thread attempted to allocate an entry in the transaction table when no entries in the shared-memory table were available. The user ID and process ID of the requesting thread are displayed.

Action

Try again later.

Unable to create output file *filename* errno = *nn*.

Cause

The operating system cannot create output file *filename*. The *errno* is the number of the operating-system error returned.

Action

Verify that the directory exists and has write permissions.

Unable to extend *nn* reserved pages for *purpose* in root chunk.

Cause

The operating system cannot extend to *nn* reserved pages for *purpose* in root chunk. (The value *purpose* can be either Checkpoint/Log, DBSpace, Chunk, or Mirror Chunk.)

Action

Reduce the ONCONFIG parameter for the resource cited; bring the database server up and free some space in the primary root chunk. Then reattempt the same operation.

Unable to initiate communications with the Optical Subsystem.

Cause

The optical driver supplied by the optical-drive vendor has indicated that the drive is not accessible.

Action

Check driver installation and cabling between the computer and the drive.

Unable to start SQL engine.

Cause

The database server encountered an out-of-memory condition.

Action

No action is necessary.

Unable to open tblspace *nn*, iserrno = *nn*.

Cause

The database server cannot open the specified tblspace. (The value *nn* is the hexadecimal representation of the tblspace number.)

Action

See the ISAM error message number *nn*, which should explain why the tblspace cannot be accessed. The error message appears in *IBM Informix Error Messages* at the IBM Informix Online Documentation site at: <http://www.ibm.com/software/data/developer/informix>.

The value of pagesize *pagesize* specified in the config file is not a valid pagesize. Use 2048, 4096 or 8192 as the value for PAGESIZE in the onconfig file and restart the server.

Cause

This message displays upon disk initialization. The value of PAGESIZE that was specified in the ONCONFIG file is not a valid value.

Action

Restart the database server with a valid PAGESIZE value.

Violations table is not started for the target table.

Cause

If you issue a STOP VIOLATIONS TABLE statement for which no violations table is started, you receive this message.

Action

To recover from this error, you must start a violations table for the target table.

Violations table reversion test completed successfully.

Cause

This message is recorded in the **logmessage** table in the **sysmaster** database when the **revtestviolations.sh** script has completed successfully (no open violations tables were found).

Action

No action is necessary. For more information on **revtestviolations.sh**, see the *IBM Informix Migration Guide*.

Violations table reversion test failed.

Cause

When the database server finds an open violations table, it reports errors 16992 and 16993 in the **logmessage** table in the **sysmaster** database and aborts the reversion process.

Action

When this message appears, you must issue the STOP VIOLATIONS TABLE FOR *table_name* command for each open violations table. After you close all open violations tables, you can restart the reversion process.

Violations table reversion test start.

Cause

This message is recorded in the **logmessage** table in the **sysmaster** database when the **revtestviolations.sh** script is executed.

Action

No action is necessary. For more information on **revtestviolations.sh**, see the *IBM Informix Migration Guide*.

Violations tables still exist.

Cause

This message is recorded in the **logmessage** table in the **sysmaster** database when an open violations table is found.

Action

When this message appears, you must issue the STOP VIOLATIONS TABLE FOR *table_name* command for each open violations table. After you close all open violations tables, you can restart the reversion process.

Virtual processor limit exceeded.

Cause

You configured the database server with more than the maximum number of virtual processors allowed (1000).

Action

To reduce the number of virtual processors, decrease the values of VPCLASS, NUMCPUVPS, NUMAIOVPS, or NETTYPE in your ONCONFIG file.

VPCLASS *classname* name is too long. Maximum length is *maxlength*.

Cause

This message indicates an internal error.

Action

Contact Technical Support.

VPCLASS *classname* duplicate class name.

Cause

This message indicates an internal error.

Action

Contact Technical Support.

VPCLASS *classname* Not enough physical procs for affinity.

Cause

The physical processors in the affinity specification for the VP class *classname* do not exist or are offline. The problem might be with the VPCLASS parameter for cpu class VPs or with the AFF_SPROC and AFF_NPROCS parameters.

Action

Make sure the named processors are online. Correct the affinity specification for the named VP class. Restart the database server.

Messages: W-X-Y-Z

WARNING: aio_wait: errno = *nn*.

Cause

While the database server was waiting for an I/O request to complete, it generated error number *nn* on an operation that it was attempting to execute.

Action

Contact Technical Support for assistance.

WARNING: Buffer pool size may cause database server to get into a locked state. Recommended minimum buffer pool size is *num* times maximum concurrent user threads.

Cause

There are not enough buffers in the buffer pool. The database server could use all available buffers and cause a deadlock to occur.

Action

Change the **buffers** field in the BUFFERPOOL parameter in the ONCONFIG file to the number that this message recommends. For more information on the BUFFERPOOL parameter, see "BUFFERPOOL" on page 1-17..

warning: Chunk time stamps are invalid.

Cause

A sanity check is performed on chunks when they are first opened at system initialization. The chunk specified did not pass the check and will be brought offline.

Action

Restore the chunk from a dbspace backup or its mirror.

Warning: *name_old* is a deprecated onconfig parameter. Use *name_new* instead. See the release notes and the Informix Administrator's Reference for more information.

Cause

A deprecated ONCONFIG parameter was used. This message displays the first time that you use a deprecated parameter. The shorter form of the message displays thereafter.

Action

Use the suggested alternative ONCONFIG parameter.

Warning: *name_old* is a deprecated onconfig parameter. Use *name_new* instead.

Cause

A deprecated ONCONFIG parameter was used.

Action

Use the suggested alternative ONCONFIG parameter.

Warning: Unable to allocate requested big buffer of size *nn*.

Cause

The internal memory allocation for a big buffer failed.

Action

Increase either virtual memory size (SHMVIRTSIZE), the size of the added segments (SHMADD), or your total shared-memory size (SHMTOTAL).

You are turning off smart large object logging.

Cause

These changes will become the new sbspace default values. Changes have been made to the sbspace. The onspaces utility will read and update 100 smart large objects at a time and commit each block of 100 smart large objects as a single transaction. This utility might take a long time to complete.

Action

This informational message occurs when you issue the following command:

```
onspaces -ch sbspace -Df "LOGGING=OFF" -y
```

For more information, see "onspaces -ch: Change sbspace default specifications" on page 14-17.

Messages: Symbols

***HH:MM:SS* Informix database server Version *R.VV.PPPPP*
Software Serial Number *RDS#XXXXXXXX*.**

Cause

This message indicate the start-up of the database server, after the initialization of shared memory.

Action

No action is required.

***argument:* invalid argument.**

Cause

This internal error indicates that an invalid argument was passed to an internal routine.

Action

Contact Technical Support.

***function_name:* cannot allocate memory.**

Cause

The database server cannot allocate memory from internal shared-memory pool.

Action

Increase either virtual-memory size (SHMVIRTSIZE), the size of the added segments (SHMADD), or your total shared-memory size (SHMTOTAL).

Conversion/Reversion Messages

These messages might display during database server conversion or reversion.

Messages: A-C

Cannot revert constraint with id *id* (in syschecks).

Cause

The database has a constraint that was defined in a version more recent than the one to which you are reverting.

Action

Drop the specified constraint and retry reversion.

Cannot revert new fragment expression for index *index*, tabid *id*.

Cause

The index fragmentation was defined in a version more recent than the one to which you are reverting.

Action

Drop the problem index-fragmentation scheme and retry reversion.

Cannot revert new table fragment expression for *table* with id *id*.

Cause

The fragmentation of this table was defined in a version more recent than the one to which you are reverting.

Action

Drop the problem table fragmentation scheme and retry reversion.

Cannot update page zero.**Cause**

Attempt to write page zero failed.

Action

Contact Technical Support.

Checking database *name* for revertibility.**Cause**

Indicates that start of the reversion checks on the specified database.

Action

None required.

Conversion of pre 7.3 in-place alter started *status*.**Cause**

The database server is converting data structures for in-place alters to the new format.

Action

None required.

Conversion of pre 9.2 database tablespaces *status*.**Cause**

The database server is converting tablespaces to the new format.

Action

None required.

The conversion of the database *name* has failed.**Cause**

Indicates that the conversion of the specified database has failed.

Action

Connect to the database. This action triggers conversion of the database. If it fails, the relevant error message appears. Contact Technical Support.

Converting database *name*...**Cause**

This message appears at the start of conversion of each database in the system.

Action

None required.

Converting in-place alters to new format.**Cause**

The database server is converting data structures for in-place alters to the new format.

Action

None required.

Converting 'onpload' database...**Cause**

Printed in `online.log` at the beginning of `onpload` conversion.

Action

None required.

Converting partition header from version 7.x.**Cause**

The database server is converting the partition header page to the new format that contains the chunk number and offset.

This message is optional verbose output that is logged only if you start `oninit` with the `-v` flag.

Action

None required.

Converting partition header page *address*.**Cause**

The database server is converting the partition header page to the new format that contains the chunk number and page offset.

This message is optional verbose output that is logged only if you start `oninit` with the `-v` flag.

Action

None required.

Converting partition header pages *status*.**Cause**

This message tracks the progress of the conversion of the partition header pages. The status is identified as follows:

- started
- succeeded

- FAILED

Action

If the status is started or succeeded, no action is required.

If conversion of the partition header pages failed, restart the database server. It will attempt to continue converting where it left off in the restartable conversion phase. If this action fails, diagnose the problem, restore from tape, fix the problem, and retry conversion.

Converting partition keys to 9.2.**Cause**

The database server is converting the partition keys to the Version 9.2 format.

This message is optional verbose output that is logged only if you start **oninit** with the **-v** flag.

Action

None required.

Converting partition name for *databasename:tablename*.**Cause**

The database server is converting the partition name for the *databasename:tablename*.

This message is optional verbose output that is logged only if you start **oninit** with the **-v** flag.

Action

None required.

Messages: D-F**The database *name* has been converted successfully.****Cause**

Indicates successful completion of the conversion of the specified database.

Action

None required.

Database *name* is not revertible...**Cause**

The database has failed one of the reversion checks and is not revertible.

Action

Take action to correct the error displayed as a separate message.

Database *name* is revertible...**Cause**

The database has passed all reversion checks and is revertible to the specified version.

Action

None required.

Database *name*: Must drop trigger (id = *id_number*).**Cause**

The database contains a trigger that was created in a version more recent than the one to which you are converting.

Action

Drop the trigger with the specified trigger identification number and then attempt reversion.

Database *name* SUCCESSFULLY reverted...**Cause**

Indicates the success of reversion of the specified database.

Action

None required.

... dropping sysmaster database.**Cause**

The database server is dropping sysmaster database during the reversion process.

Action

No action is required.

The dummy updates failed while converting database *name*. This may imply data corruption in the database. If so, restore the original database with the tape backup. For more information, see *output_file*.

Cause

During conversion of a database from a version earlier than Version 9.2, dummy update statements are run against the system tables in the database being converted. This message indicates failure in running one of these update statements.

Action

To retry the dummy updates, run the dummy update script for your old database server version. For instructions, refer to the *IBM Informix Migration Guide*.

If data corruption occurred, restore the original database with the tape backup. For more information, see the *IBM Informix Backup and Restore Guide*.

The dummy updates succeeded while converting database *name*.
Cause

During conversion of a database from a version earlier than Version 9.2, dummy update statements are run against the system tables in the database being converted. This message indicates successful completion of these updates.

Action

None required.

Error in slow altering a system table.
Cause

An internal error occurred while performing reversion.

Action

Contact Technical Support.

External conversion aborted due to incompatible sysmaster database.
Cause

The **sysmaster** database was not converted to the current database server version. A current **sysmaster** database is needed for external conversion to complete.

Action

Drop the **sysmaster** database and reboot the database server. It will build a new **sysmaster** database and relaunch external conversion automatically.

Messages: I-P

Internal server error.
Cause

An unexpected error occurred during database reversion.

Action

Contact Technical Support.

Must drop long identifiers in table *name* in database *name*
Cause

Identifiers greater than 18 bytes in length are not supported in the database server version to which you are reverting.

Action

Make sure that all long identifiers in the system are either dropped or renamed before you attempt reversion.

Must drop new database (*name*) before attempting reversion.
Iserrno *error_number*

Cause

The system contains a database that was created in a more recent version of the database server.

Action

Drop the new database and attempt reversion.

Must drop new user defined statistics in database *name*, iserrno *number*

Cause

Some distributions in the **sysdistrib** system table use user-defined statistics. This feature is not supported in the version to which you are reverting.

Action

Ensure that no user-defined statistics are present or used in the system and then attempt reversion.

ON-Bar conversion completed successfully.

Cause

ON-Bar conversion completed successfully.

Action

None.

ON-Bar conversion failed see /tmp/bar_conv.out.

Cause

ON-Bar conversion failed.

Action

For failure details, see /tmp/bar_conv.out.

ON-Bar conversion start:

Cause

ON-Bar conversion script is now running.

Action

None.

ON-Bar reversion completed successfully.

Cause

ON-Bar reversion was completed successfully.

Action

None.

ON-Bar reversion failed see /tmp/bar_rev.out.

Cause

ON-Bar reversion failed.

Action

For failure details, see /tmp/bar_rev.out.

ON-Bar reversion start:

Cause

ON-Bar reversion script is now running.

Action

None.

ON-Bar reversion test completed successfully.

Cause

ON-Bar reversion test was completed successfully.

Action

None.

ON-Bar reversion test start:

Cause

ON-Bar reversion test script is now running.

Action

None.

'onpload' conversion completed successfully.

Cause

Displayed in `online.log` at the successful completion of `onpload` conversion.

Action

None required.

**'onpload' conversion failed. For details, look in
\$INFORMIXDIR/etc/conpload.out.**

Cause

Conversion of the `onpload` database failed.

Action

Find out the cause of failure from `$INFORMIXDIR/etc/conpload.out`. Fix the problem before you reattempt conversion.

...'onpload' reversion completed successfully.

Cause

Printed in `online.log` at the successful completion of reversion.

Action

None required.

...'onpload' reversion failed. For details, look in \$INFORMIXDIR/etc/revpload.out.

Cause

Reversion of the `onpload` database failed.

Action

Find the cause of failure in `$INFORMIXDIR/etc/revpload.out`. Fix the problem before you reattempt reversion.

'onpload' reversion test completed successfully.

Cause

Printed in `online.log` if the `onpload` database is revertible.

Action

None required.

'onpload' reversion test start:

Cause

Printed in `online.log` at the beginning of `onpload` reversion testing.

Action

None required.

The pload database contains load/unload jobs referring to long table names, column names, or database names. These jobs will not work as expected until they are redefined.

Cause

Printed during `onpload` reversion testing if the `onpload` database contains references to long table names, column names, or database names. But the reversion will complete.

Action

Redefine the load and unload jobs in the `onpload` database that have references to long identifiers.

Messages: R-W

...reverting 'onpload' database.

Cause

Printed in `online.log` at the beginning of `onpload` reversion.

Action

None required.

Reverting partition header from version 9.2.

Cause

The database server is reverting the partition header page to the old format that contains the physical address.

This message is optional verbose output that is logged only if you start `oninit` with the `-v` flag.

Action

None required.

Reverting partition header page *address*.

Cause

The database server is reverting the partition header page to the old format that contains the physical address.

This message is optional verbose output that is logged only if you start `oninit` with the `-v` flag.

Action

None required.

Reverting partition header pages *status*.

Cause

The database server is reverting the partition header pages to the old format. The status is identified as follows:

- started
- succeeded
- FAILED

Action

If reversion of the partition header pages started or succeeded, no action is required. If reversion of the partition header pages failed, restore from a tape backup, diagnose and fix the problem, and retry conversion.

Reverting partition keys to pre 9.2.

Cause

The database server is reverting the partition keys to the pre-Version 9.2 format.

This message is optional verbose output that is logged only if you start **oninit** with the **-v** flag.

Action

None required.

Reverting partition *name* for *databasename:tablename*.

Cause

The database server is reverting the partition name for *databasename:tablename*.

This message is optional verbose output that is logged only if you start **oninit** with the **-v** flag.

Action

None required.

... reverting reserved pages.

Cause

The database server is reverting reserved pages.

Action

No action is required.

... reverting tables that underwent In-Place Alter.

Cause

The database server is reverting tables that underwent in-place alter.

Action

No action is required.

R-tree error message conversion completed successfully.

Cause

R-tree error message conversion was completed successfully.

Action

None required

R-tree error message conversion failed. (See /tmp/contree.out or %TMP%\contree.out)

Cause

R-tree error message conversion failed.

Action

See `/tmp/conR-tree.out` and `/tmp/R-tree.databases`.

R-tree error message conversion started.**Cause**

R-tree error message conversion script is now running.

Action

None required.

Reversion cancelled.**Cause**

The reversion process was cancelled because of errors encountered.

Action

Correct the cause of the errors, and restart reversion.

Reversion complete. Install IBM Informix database server *version* before restarting.**Cause**

The reversion process was completed successfully.

Action

You must install the older database version.

Reversion of database *name* FAILED**Cause**

Indicates the failure of reversion of the specified database.

Action

None required.

...reverting 'syscdr' database.**Cause**

Printed in `online.log` at the beginning of Enterprise Replication reversion.

Action

None required.

...starting reversion of database *name*.**Cause**

Indicates the start of actual reversion of the specified database.

Action

None required.

There is a semi-detached index in this table, which cannot be reverted. Drop this index, and retry reversion.

Cause

A semi-detached index on this table cannot be reverted.

Action

To see the list of all semi-detached indexes, refer to the database server message log. These indexes cannot be reverted. To continue reversion, drop these semi-detached indexes and retry reversion. If needed, you will need to re-create these indexes after reversion is complete.

Unable to read reserved page *chunk:offset - reserved_page*.

Cause

Both disk pages in a given reserved page pair are bad. On the disk page, *chunk* represents the chunk number and *offset* represents the page offset for the chunk.

Action

Contact Technical Support.

WARNING: Target server version must have a certified Storage Manager installed after conversion/reversion and before bringing up server.

Cause

ON-Bar is being converted or reverted. The user must ensure that a storage manager, certified with the target database server version, is installed.

Action

None.

Conversion and Reversion Messages for Enterprise Replication

Use the **concdr.sh** script on UNIX or the **concdr.bat** script on Windows to convert Enterprise Replication and the **syscdr** database to Version 10.0. Use the **revcdr.sh** script on UNIX or the **revcdr.bat** script on Windows to revert Enterprise Replication and the **syscdr** database to an earlier version. These scripts write conversion and reversion messages for Enterprise Replication to the following locations:

- Output of the **concdr.sh** or **concdr.bat** script, which is standard output by default
- **concdr.out** file
- Output of the **revcdr.sh** or **revcdr.bat** script, which is standard output by default
- **revcdr.out** file
- **revtestcdr.out** file

You can find the **concdr.out**, **revcdr.out**, and **revtestcdr.out** files in **\$INFORMIXDIR/etc** on UNIX or **%INFORMIXDIR%\etc** on Windows. For more information on converting and reverting Enterprise Replication, see the *IBM Informix Migration Guide*.

CDR reversion test completed successfully.

Cause

The **syscdr** database is revertible.

Action

None required.

Prints the output of the **revcdr.sh** or **revcdr.bat** script to standard output.

CDR reversion test failed; for details look in \$INFORMIXDIR/etc/revtestcdr.out.

Cause

Enterprise Replication is not revertible.

Action

For more information, look at the messages in **revtestcdr.out**. Fix the reported problem before you attempt reversion.

Prints the output of the **revcdr.sh** or **revcdr.bat** script to standard output.

Enterprise Replication is not ready for conversion. The Control and TRG send queues should be empty for conversion/reversion to proceed.

Cause

There are elements in the control and Transaction Send Queue (also called TRG) send queues. The database server sends replicated data to the TRG queue before sending it to the target system.

Action

Wait for these queues to empty before you attempt either conversion or reversion. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Prints this message to **concdr.out** during conversion or to **revcdr.out** during reversion.

Enterprise Replication is not ready for conversion. The syscdr database should NOT contain old-style group definitions for conversion to succeed.

Cause

The `syscdr` database *should not* contain old-style group definitions for conversion to succeed.

Action

Use the `cdr delete group` command to delete the old-style groups before attempting conversion. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Prints this message to `concdr.out`.

Enterprise Replication should be in a stopped state for conversion/reversion to proceed.

Cause

Enterprise Replication should be in a stopped state for conversion or reversion to proceed.

Action

Stop Enterprise Replication. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Prints this message to `concdr.out` during conversion or to `revcdr.out` during reversion.

Reversion of 'syscdr' failed; for details look in \$INFORMIXDIR/etc/revcdr.out.

Cause

The reversion of the `syscdr` database failed.

Action

Find the cause of failure in the `revcdr.out` file, then fix the problem before you attempt reversion.

Prints the output of the `revcdr.sh` or `revcdr.bat` script to standard output.

Starting CDR reversion test...

Cause

This message displays at the beginning of Enterprise Replication reversion testing.

Prints the output of the `revcdr.sh` or `revcdr.bat` script to standard output.

Action

None required.

Starting 'syscdr' conversion...

Cause

This message displays when you run the **concdr.sh** or **concdr.bat** script to convert the **syscdr** database to Version 10.0.

Action

None required.

Prints the output of the **concdr.sh** or **concdr.bat** script to standard output.

Starting 'syscdr' reversion...

Cause

This message displays when you run the **revcdr.sh** or **revcdr.bat** script to revert the **syscdr** database to an earlier version.

Action

None required.

Prints the output of the **revcdr.sh** or **revcdr.bat** script to standard output.

'syscdr' conversion completed successfully.

Cause

This message displays after you complete converting Enterprise Replication and the **syscdr** database to Version 10.0.

Action

None required.

Prints the output of the **concdr.sh** or **concdr.bat** script to standard output.

'syscdr' conversion failed. For details, look in \$INFORMIXDIR/etc/concdr.out.

Cause

Conversion of the **syscdr** database failed.

Action

If conversion fails, resolve the problem reported in **concdr.out**. Restore the **syscdr** database from backup and reattempt conversion.

Prints the output of the **concdr.sh** or **concdr.bat** script to standard output.

Syscdr should NOT contain new replicate sets for reversion to succeed.

Cause

The new replicate sets in the `syscdr` database are not compatible with older versions.

Action

Use the `cdr delete replicateset` command to delete the replicate sets. Then rerun the `revcdr.sh` or `revcdr.bat` script to reattempt reversion.

Prints this message to `revtestcdr.out`.

Syscdr should not contain replicates defined with the --floatiee option for reversion to succeed.

Cause

Replicates have been defined with the `--floatiee` option. You cannot revert these replicates to the older version.

Action

Use the `cdr delete replicateset` command to delete replicates defined with the `--floatiee` option, then reattempt reversion.

Prints this message to `revtestcdr.out`.

Dynamic Log Messages

Dynamically added log file *logid* to DBspace *dbspace_number*.

Cause

The next active log file contains records of an open transaction. Whenever the database server adds a log dynamically, it logs this message. Example: Dynamically added log file 38 to DBspace 5.

Action

Complete the transaction as soon as possible.

Log file *logid* added to DBspace *dbspace_number*.

Cause

Whenever the administrator adds a log file manually, the database server logs this message. Example: Log file 97 added to DBspace 2.

Action

None required.

Log file number *logid* has been dropped from DBspace *dbspace_number*.

Cause

When you drop a newly-added log file, the database server logs this message.
Example: Log file number 204 has been dropped from DBspace 17.

Action

None required.

Log file *logid* has been pre-dropped.

Cause

When you drop a used log file, it is marked as deleted (status **D**) and cannot be used again. After you perform a level-0 backup, the database server drops this log file and can reuse the space. Example: Log file 12 has been pre-dropped.

Action

To delete the log file, perform a level-0 backup of all storage spaces.

Pre-dropped log file number *logid* has been deleted from DBspace *dbspace_number*.

Cause

After a backup, the database server deletes a pre-dropped log file and logs this message. Example: Pre-dropped log file number 12 has been deleted from DBspace 3.

Action

None required.

ALERT: Because the oldest logical log (*logid*) contains records from an open transaction (*transaction_address*), the server is attempting to dynamically add a log file. But there is no space available. Please add a DBspace or chunk. Then complete the transaction as soon as possible.

Cause

If the database server is unable to dynamically add a log file because the instance is out of space, it logs this message.

Action

Add a dbspace or chunk to an existing dbspace. Then complete the transaction as soon as possible.

ALERT: The oldest logical log (*logid*) contains records from an open transaction (*transaction_address*). Logical logging will remain blocked until a log file is added. Add the log file with the onparams `-a` command, using the `-i` (insert) option, as in: `onparams -a -d dbspace -s size -i`. Then complete the transaction as soon as possible.

Cause

If the DYNAMIC_LOGS parameter is set to 1, the database server prompts the administrator to add log files manually when they are needed.

Action

Use the `onparams -a` command with the `-i` option to add the log file after the current log file. Then complete the transaction as soon as possible.

Log file *logid* has been pre-dropped. It will be deleted from the log list and its space can be reused once you take level-0 archives of all BLOBspaces, Smart BLOBspaces and non-temporary DBspaces.

Cause

When you drop a used log file, it is marked as deleted (status **D**) and cannot be used again, and `onparams` prints this message.

Action

To delete the log file, perform a level-0 backup of all storage spaces.

Sbospace Metadata Messages

Allocated *number* pages to Metadata from chunk *number*.

Cause

The database server freed the specified number of pages from the reserved area and moved them to the metadata area of chunk *number*.

Action

None required.

Allocated *number* pages to Userdata from chunk *number*.

Cause

The database server freed the specified number of pages from the reserved area and moved them to the user-data area of chunk *number*.

Action

None required.

Freeing reserved space from chunk *number* to Metadata.

Cause

The metadata area in chunk *number* is full. The database server is trying to free space from the reserved area to the metadata area.

Action

None required.

Freeing reserved space from chunk *number* to Userdata.

Cause

The user-data area in chunk *number* is full. The database server is trying to free space from the reserved area to the user-data area.

Action

None required.

Truncate Table Messages

The table cannot be truncated if it has an open cursor or dirty readers.

Cause

You must have exclusive access to the table.

Action

Wait for dirty readers to complete or close all the open cursors and reissue the TRUNCATE TABLE command.

The table cannot be truncated. It has at least one non-empty child table with referential constraints.

Cause

You cannot truncate a table if it has child tables with referential constraints and at least one row.

Action

Empty the child tables before you truncate this table.

Appendix F. Limits in IBM Informix Dynamic Server

The following sections list selected capacity limits and system defaults for IBM Informix Dynamic Server.

Limitations on UNIX Operating Systems

System-Level Parameter Limits (UNIX)

System-Level Parameters	Maximum Capacity per Computer System
IBM Informix Dynamic Server systems per computer (Dependent on available system resources)	255
Maximum number of accessible remote sites	Machine specific
Maximum virtual shared memory segment (SHMVIRTSIZE)	2GB (32-bit platforms) or 4TB (64-bit platforms)
Maximum address space	1.7GB if boot.ini file not modified to 3GB 2.7GB if boot.ini file is modified to 3GB

Table-Level Parameter Limits (UNIX)

Table-Level Parameters (based on 2K page size)	Maximum Capacity per Table
Data rows per fragment	4,277,659,295
Data pages per fragment	16,775,134
Data bytes per fragment (excludes Smart Large Objects (BLOB, CLOB) and Simple Large Objects (BYTE, TEXT) created in Blobspaces)	33,818,671,136
Binary Large Object BLOB/CLOB pages	4*2*40
Binary Large Objects TEXT/BYTE bytes	4*2*40
Row length	32,767
Number of columns	32K
Key parts per index	16
Columns per functional index	102 (for C UDRs) 341 (for SPL or Java UDRs)
Maximum bytes per index key (for a given page size):	2K page size = 387 4K page size = 796 8K page size = 1615 12K page size = 2435 16K page size = 3254
Maximum size of an SQL statement	64K

Access Capabilities (UNIX)

Access Capabilities	Maximum Capacity per System
Maximum databases per Dynamic Server system	21 million
Maximum tables per Dynamic Server system	477,102,080
Maximum active users per Dynamic Server (minus the minimum number of system threads)	32K user threads
Maximum active users per database and table (also limited by the number of available locks, a tunable parameter)	32K user threads
Maximum number of open tables per Dynamic Server system	Dynamic allocation
Maximum number of open tables per user and join	Dynamic allocation
Maximum locks per Dynamic Server system and database	Dynamic allocation
Maximum number of page cleaners	128
Maximum number of partitions per dbspace	4K page size: 1048445, 2K page size: 1048314 (based on 4-bit bitmaps)
Maximum number of recursive synonym mappings	16
Maximum number of tables locked with LOCK TABLE per user	32
Maximum number of cursors per user	Machine specific
Maximum Enterprise Replication transaction size	4 TB
Maximum dbspace size	4 TB with 2K page size 8 TB with 4K page size
Maximum sbspace size	4 TB with 2K page size 8 TB with 4K page size
Maximum chunk size	4 TB
Maximum number of chunks	32,766
Maximum number of 2K pages per chunk	2 billion
Maximum number of open Simple Large Objects (applies only to TEXT and BYTE data types)	20
Maximum number of B-tree levels	20
Maximum amount of decision support memory	Machine specific
Maximum size of a Dynamic Server instance	8 PB
Utility support for large files	17 billion GB
Maximum number of storage spaces (dbspaces, blobspaces, sbspaces, or extspaces)	2047

IBM Informix Dynamic Server System Defaults (UNIX)

Table lock mode	Page
Initial extent size	8 pages
Next extent size	8 pages
Read-only isolation level (with database transactions)	Committed Read

Read-only isolation level (ANSI-compliant database)	Repeatable Read
---	-----------------

ON-Monitor Statistics (UNIX)

Number of displayed user threads	1000
Number of displayed chunks	1000
Number of displayed dbspaces	1000
Number of displayed databases	1000
Number of displayed logical logs	1000

Limitations on Windows Operating Systems

System-Level Parameter Limits (Windows)

System-Level Parameters	Maximum Capacity per Computer System
IBM Informix Dynamic Server systems per computer (Dependent on available system resources)	255
Maximum number of accessible remote sites	Machine specific
Maximum virtual shared memory segment (SHMVIRTSIZE)	2GB (32-bit platforms) or 4TB (64-bit platforms)
Maximum address space	1.7GB if boot.ini file not modified to 3GB 2.7GB if boot.ini file is modified to 3GB

Table-Level Parameter Limits (Windows)

Table-Level Parameters (based on 2K page size)	Maximum Capacity per Table
Data rows per fragment	4,277,659,295
Data pages per fragment	16,775,134
Data bytes per fragment (excludes Smart Large Objects (BLOB, CLOB) and Simple Large Objects (BYTE, TEXT) created in Blobspaces)	33,818,671,136
Binary Large Object BLOB/CLOB pages	4 TB
Binary Large Objects TEXT/BYTE bytes	4 TB
Row length	32,767
Number of columns	32 K
Key parts per index	16
Columns per functional index	102 (for C UDRs) 341 (for SPL or Java UDRs)
Maximum bytes per index key (for a given page size):	2K page size = 387 4K page size = 796 8K page size = 1615 12K page size = 2435 16K page size = 3254

Table-Level Parameters (based on 2K page size)	Maximum Capacity per Table
Maximum size of an SQL statement	64 K

Access Capabilities (Windows)

Access Capabilities	Maximum Capacity per System
Maximum databases per Dynamic Server system	21 million
Maximum tables per Dynamic Server system	477,102,080
Maximum active users per Dynamic Server (minus the minimum number of system threads)	32K user threads
Maximum active users per database and table (also limited by the number of available locks, a tunable parameter)	32K user threads
Maximum number of open tables per Dynamic Server system	Dynamic allocation
Maximum number of open tables per user and join	Dynamic allocation
Maximum locks per Dynamic Server system and database	Dynamic allocation
Maximum number of page cleaners	128
Maximum number of recursive synonym mappings	16
Maximum number of tables locked with LOCK TABLE per user	32
Maximum number of cursors per user	Machine specific
Maximum Enterprise Replication transaction size	4 TB
Maximum dbspace size	8 TB
Maximum sbspace size	8 TB
Maximum chunk size	4 TB
Maximum number of chunks	32,766
Maximum number of 2K pages per chunk	2 billion
Maximum number of open Simple Large Objects (applies only to TEXT and BYTE data types)	20
Maximum number of B-tree levels	20
Maximum amount of decision support memory	Machine specific
Maximum size of a Dynamic Server instance	8 PB
Utility support for large files	17 billion GB
Maximum number of storage spaces (dbspaces, blobspaces, sbspaces, or extspaces)	2047
Maximum number of partitions per dbspace	4K page size: 1048445, 2K page size: 1048314 (based on 4-bit bitmaps)

IBM Informix Dynamic Server System Defaults (Windows)

Table lock mode	Page
Initial extent size	8 pages

Next extent size	8 pages
Read-only isolation level (with database transactions)	Committed Read
Read-only isolation level (ANSI-compliant database)	Repeatable Read

Appendix G. Accessibility

IBM strives to provide products with usable access for everyone, regardless of age or ability.

Accessibility features for IBM Informix Dynamic Server

Accessibility features help a user who has a physical disability, such as restricted mobility or limited vision, to use information technology products successfully.

Accessibility Features

The following list includes the major accessibility features in IBM Informix Dynamic Server. These features support:

- Keyboard-only operation.
- Interfaces that are commonly used by screen readers.
- The attachment of alternative input and output devices.

Tip: The IBM Informix Dynamic Server Information Center and its related publications are accessibility-enabled for the IBM Home Page Reader. You can operate all features using the keyboard instead of the mouse.

Keyboard Navigation

This product uses standard Microsoft Windows navigation keys.

Related Accessibility Information

IBM is committed to making our documentation accessible to persons with disabilities. Our publications are available in HTML format so that they can be accessed with assistive technology such as screen reader software. The syntax diagrams in our publications are available in dotted decimal format. For more information about the dotted decimal format, go to "Dotted Decimal Syntax Diagrams."

You can view the publications for IBM Informix Dynamic Server in Adobe Portable Document Format (PDF) using the Adobe Acrobat Reader.

IBM and Accessibility

See the *IBM Accessibility Center* at <http://www.ibm.com/able> for more information about the commitment that IBM has to accessibility.

Dotted Decimal Syntax Diagrams

The syntax diagrams in our publications are available in dotted decimal format, which is an accessible format that is available only if you are using a screen reader.

In dotted decimal format, each syntax element is written on a separate line. If two or more syntax elements are always present together (or always absent together), the elements can appear on the same line, because they can be considered as a single compound syntax element.

Each line starts with a dotted decimal number; for example, 3 or 3.1 or 3.1.1. To hear these numbers correctly, make sure that your screen reader is set to read punctuation. All syntax elements that have the same dotted decimal number (for example, all syntax elements that have the number 3.1) are mutually exclusive alternatives. If you hear the lines 3.1 USERID and 3.1 SYSTEMID, your syntax can include either USERID or SYSTEMID, but not both.

The dotted decimal numbering level denotes the level of nesting. For example, if a syntax element with dotted decimal number 3 is followed by a series of syntax elements with dotted decimal number 3.1, all the syntax elements numbered 3.1 are subordinate to the syntax element numbered 3.

Certain words and symbols are used next to the dotted decimal numbers to add information about the syntax elements. Occasionally, these words and symbols might occur at the beginning of the element itself. For ease of identification, if the word or symbol is a part of the syntax element, the word or symbol is preceded by the backslash (\) character. The * symbol can be used next to a dotted decimal number to indicate that the syntax element repeats. For example, syntax element *FILE with dotted decimal number 3 is read as 3 * FILE. Format 3* FILE indicates that syntax element FILE repeats. Format 3* * FILE indicates that syntax element * FILE repeats.

Characters such as commas, which are used to separate a string of syntax elements, are shown in the syntax just before the items they separate. These characters can appear on the same line as each item, or on a separate line with the same dotted decimal number as the relevant items. The line can also show another symbol that provides information about the syntax elements. For example, the lines 5.1*, 5.1 LASTRUN, and 5.1 DELETE mean that if you use more than one of the LASTRUN and DELETE syntax elements, the elements must be separated by a comma. If no separator is given, assume that you use a blank to separate each syntax element.

If a syntax element is preceded by the % symbol, this identifies a reference that is defined elsewhere. The string following the % symbol is the name of a syntax fragment rather than a literal. For example, the line 2.1 %OP1 means that you should refer to a separate syntax fragment OP1.

The following words and symbols are used next to the dotted decimal numbers:

- ? Specifies an optional syntax element. A dotted decimal number followed by the ? symbol indicates that all the syntax elements with a corresponding dotted decimal number, and any subordinate syntax elements, are optional. If there is only one syntax element with a dotted decimal number, the ? symbol is displayed on the same line as the syntax element (for example, 5? NOTIFY). If there is more than one syntax element with a dotted decimal number, the ? symbol is displayed on a line by itself, followed by the syntax elements that are optional. For example, if you hear the lines 5 ?, 5 NOTIFY, and 5 UPDATE, you know that syntax elements NOTIFY and UPDATE are optional; that is, you can choose one or none of them. The ? symbol is equivalent to a bypass line in a railroad diagram.
- ! Specifies a default syntax element. A dotted decimal number followed by the ! symbol and a syntax element indicates that the syntax element is the default option for all syntax elements that share the same dotted decimal number. Only one of the syntax elements that share the same dotted decimal number can specify a ! symbol. For example, if you hear the lines

2? FILE, 2.1! (KEEP), and 2.1 (DELETE), you know that (KEEP) is the default option for the FILE keyword. In this example, if you include the FILE keyword but do not specify an option, default option KEEP is applied. A default option also applies to the next higher dotted decimal number. In this example, if the FILE keyword is omitted, default FILE(KEEP) is used. However, if you hear the lines 2? FILE, 2.1, 2.1.1! (KEEP), and 2.1.1 (DELETE), the default option KEEP only applies to the next higher dotted decimal number, 2.1 (which does not have an associated keyword), and does not apply to 2? FILE. Nothing is used if the keyword FILE is omitted.

- * Specifies a syntax element that can be repeated zero or more times. A dotted decimal number followed by the * symbol indicates that this syntax element can be used zero or more times; that is, it is optional and can be repeated. For example, if you hear the line 5.1* data-area, you know that you can include more than one data area or you can include none. If you hear the lines 3*, 3 HOST, and 3 STATE, you know that you can include HOST, STATE, both together, or nothing.

Notes:

1. If a dotted decimal number has an asterisk (*) next to it and there is only one item with that dotted decimal number, you can repeat that same item more than once.
2. If a dotted decimal number has an asterisk next to it and several items have that dotted decimal number, you can use more than one item from the list, but you cannot use the items more than once each. In the previous example, you could write HOST STATE, but you could not write HOST HOST.
3. The * symbol is equivalent to a loop-back line in a railroad syntax diagram.

- + Specifies a syntax element that must be included one or more times. A dotted decimal number followed by the + symbol indicates that this syntax element must be included one or more times. For example, if you hear the line 6.1+ data-area, you must include at least one data area. If you hear the lines 2+, 2 HOST, and 2 STATE, you know that you must include HOST, STATE, or both. As for the * symbol, you can only repeat a particular item if it is the only item with that dotted decimal number. The + symbol, like the * symbol, is equivalent to a loop-back line in a railroad syntax diagram.

Notices

This information was developed for products and services offered in the U.S.A.

IBM may not offer the products, services, or features discussed in this document in other countries. Consult your local IBM representative for information on the products and services currently available in your area. Any reference to an IBM product, program, or service is not intended to state or imply that only that IBM product, program, or service may be used. Any functionally equivalent product, program, or service that does not infringe any IBM intellectual property right may be used instead. However, it is the user's responsibility to evaluate and verify the operation of any non-IBM product, program, or service.

IBM may have patents or pending patent applications covering subject matter described in this document. The furnishing of this document does not grant you any license to these patents. You can send license inquiries, in writing, to:

IBM Director of Licensing
IBM Corporation
North Castle Drive
Armonk, NY 10504-1785
U.S.A.

For license inquiries regarding double-byte (DBCS) information, contact the IBM Intellectual Property Department in your country or send inquiries, in writing, to:

IBM World Trade Asia Corporation
Licensing
2-31 Roppongi 3-chome, Minato-ku
Tokyo 106-0032, Japan

The following paragraph does not apply to the United Kingdom or any other country where such provisions are inconsistent with local law: INTERNATIONAL BUSINESS MACHINES CORPORATION PROVIDES THIS PUBLICATION "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF NON-INFRINGEMENT, MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE. Some states do not allow disclaimer of express or implied warranties in certain transactions, therefore, this statement may not apply to you.

This information could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein; these changes will be incorporated in new editions of the publication. IBM may make improvements and/or changes in the product(s) and/or the program(s) described in this publication at any time without notice.

Any references in this information to non-IBM Web sites are provided for convenience only and do not in any manner serve as an endorsement of those Web sites. The materials at those Web sites are not part of the materials for this IBM product and use of those Web sites is at your own risk.

IBM may use or distribute any of the information you supply in any way it believes appropriate without incurring any obligation to you.

Licenses of this program who wish to have information about it for the purpose of enabling: (i) the exchange of information between independently created programs and other programs (including this one) and (ii) the mutual use of the information which has been exchanged, should contact:

IBM Corporation
J46A/G4
555 Bailey Avenue
San Jose, CA 95141-1003
U.S.A.

Such information may be available, subject to appropriate terms and conditions, including in some cases, payment of a fee.

The licensed program described in this document and all licensed material available for it are provided by IBM under terms of the IBM Customer Agreement, IBM International Program License Agreement or any equivalent agreement between us.

Any performance data contained herein was determined in a controlled environment. Therefore, the results obtained in other operating environments may vary significantly. Some measurements may have been made on development-level systems and there is no guarantee that these measurements will be the same on generally available systems. Furthermore, some measurements may have been estimated through extrapolation. Actual results may vary. Users of this document should verify the applicable data for their specific environment.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

All statements regarding IBM's future direction or intent are subject to change or withdrawal without notice, and represent goals and objectives only.

All IBM prices shown are IBM's suggested retail prices, are current and are subject to change without notice. Dealer prices may vary.

This information is for planning purposes only. The information herein is subject to change before the products described become available.

This information contains examples of data and reports used in daily business operations. To illustrate them as completely as possible, the examples include the names of individuals, companies, brands, and products. All of these names are fictitious and any similarity to the names and addresses used by an actual business enterprise is entirely coincidental.

COPYRIGHT LICENSE:

This information contains sample application programs in source language, which illustrate programming techniques on various operating platforms. You may copy, modify, and distribute these sample programs in any form without payment to IBM, for the purposes of developing, using, marketing or distributing application programs conforming to the application programming interface for the operating

platform for which the sample programs are written. These examples have not been thoroughly tested under all conditions. IBM, therefore, cannot guarantee or imply reliability, serviceability, or function of these programs.

Each copy or any portion of these sample programs or any derivative work, must include a copyright notice as follows:

© your company name) (year). Portions of this code are derived from IBM Corp. Sample Programs.

© Copyright IBM Corp. _enter the year or years_. All rights reserved.

If you are viewing this information softcopy, the photographs and color illustrations may not appear.

Trademarks

IBM, the IBM logo, and [ibm.com](http://www.ibm.com) are trademarks or registered trademarks of International Business Machines Corporation in the United States, other countries, or both. These and other IBM trademarked terms are marked on their first occurrence in this information with the appropriate symbol (® or ™), indicating US registered or common law trademarks owned by IBM at the time this information was published. Such trademarks may also be registered or common law trademarks in other countries. A current list of IBM trademarks is available on the Web at <http://www.ibm.com/legal/copytrade.shtml>.

Adobe, Acrobat, Portable Document Format (PDF), PostScript, and all Adobe-based trademarks are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, other countries, or both.

Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

Java and all Java-based trademarks are trademarks of Sun Microsystems, Inc. in the United States, other countries, or both.

Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.

Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.

UNIX is a registered trademark of The Open Group in the United States and other countries.

Other company, product, or service names may be trademarks or service marks of others.

Index

Special characters

- V option 6-1
- version option 6-1
- .conf.dbservername file A-4
- .inf.servicename file A-6
- .informix file A-5
- .infos.dbservername file
 - defined A-6
 - regenerating 11-18
- .infxdirs file A-6
- jvpprops file 1-54, A-7

Numerics

- 64-bit addressing
 - buffer pool 1-19
 - memory 1-90

A

- AC_CONFIG environment variable 1-10
- ac_config.std file 1-10, A-4
- AC_DEBUG configuration parameter 1-10
- AC_IXBAR configuration parameter 1-10
- AC_LTAPEBLOCK configuration parameter 1-10
- AC_LTAPEDEV configuration parameter 1-10
- ac_msg.log file A-3
- AC_MSGPATH configuration parameter 1-10, 1-11
- AC_SCHEMA configuration parameter 1-11
- AC_STORAGE configuration parameter 1-10, 1-11
- AC_TAPEBLOCK configuration parameter 1-11
- AC_TAPEDEV configuration parameter 1-11
- AC_TIMEOUT configuration parameter 1-11
- AC_VERBOSE configuration parameter 1-11
- accessibility G-1
 - keyboard G-1
 - shortcut keys G-1
- Accessibility G-1
 - dotted decimal format of syntax diagrams G-1
 - syntax diagrams, reading in a screen reader G-1
- ACCESSTIME tag 14-13
- ADDCHK logical-log record 5-4
- ADDDBS logical-log record 5-4
- Adding
 - CPU virtual processors 11-16, 11-17
- ADDITEM logical-log record 5-4
- ADDLOG logical-log record 5-4
- ADMIN_MODE_USERS
 - changing dynamically 11-20
- ADMIN_MODE_USERS configuration parameter 1-11
- ADMIN_USER_MODE_WITH_DBSA configuration parameter 1-12
- Administration mode 11-14
- Administrative API command
 - ALTER LOGMODE 8-1
- ADTERR configuration parameter 1-12
- ADTMODE configuration parameter 1-12
- ADTPATH configuration parameter 1-12
- ADTSIZE configuration parameter 1-12
- af.xxx file A-3

- AFCRASH configuration parameter 1-54, A-7
- AFDEBUG environment variable A-7
- AFF_NPROCS configuration parameter
 - defined D-2
 - VPCLASS D-2
- AFF_SPROC configuration parameter
 - defined D-2
 - VPCLASS D-3
- alarm scripts
 - foreground operations C-2
- ALARMPROGRAM configuration parameter 1-12, C-1
- Allocating unbuffered disk space 14-6, 14-8
- ALLOCGENPG log record 5-4
- ALLOW_NEWLINE configuration parameter 1-13
- ALRM_ALL_EVENTS configuration parameter 1-13
- ALTER LOGMODE
 - Administrative API command 8-1
- ALTERDONE log record 5-4
- Alternate dbservername 1-25
- ALTSPCOLSNEW log record 5-4
- ALTSPCOLSOLD log record 5-4
- archecker utility
 - configuration parameters 1-10
- Archiving
 - after renaming spaces 14-25
- Assertion failure
 - af.xxx file A-3
 - DUMPCNT configuration parameter 1-40
 - DUMPCORE configuration parameter 1-41
 - DUMPSHMEM configuration parameter 1-42
 - gcore file A-5
 - shmem.xxx file A-9
- Assertion-failed messages E-2
- Asynchronous I/O
 - cooked chunks D-8
- Attribute
 - configuration parameter 1-9
- Audit records
 - configuration parameters 1-12
 - sysadinfo table 2-7
 - sysaudit table 2-7
- AUTO_AIOVPS
 - changing dynamically 11-20
- AUTO_AIOVPS configuration parameter 1-14
- AUTO_CKPTS
 - changing dynamically 11-20
- AUTO_CKPTS configuration parameter 1-14
- AUTO_LRU_TUNING
 - changing dynamically 11-21
- AUTO_LRU_TUNING configuration parameter 1-15
- AUTO_REPREPARE configuration parameter 1-15
- AVG_LO_SIZE tag 14-13

B

- B-tree
 - functional index 4-20
 - key-value locking 4-19
 - structure 4-16
- B-tree scanner 1-16, 11-4

- Backups
 - adding log files 13-2
 - after creating a storage space 14-6, 14-7, 14-16
 - automatic log 1-12, C-1
 - boot file A-5
 - changing physical log 13-4
 - displaying contents 10-1
 - dropping log files 13-3
 - external 11-4
 - ixbar file A-5
- BADIDX logical-log record 5-4
- BAR_ACT_LOG configuration parameter 1-73
- BAR_ACT_LOG file 8-1
- bar_act.log file A-4
- bar_action table 2-6
- BAR_BSALIB_PATH configuration parameter 1-73
- BAR_DEBUG configuration parameter 1-73
- BAR_DEBUG_LOG configuration parameter 1-73
- BAR_HISTORY configuration parameter 1-73
- bar_instance table 2-6
- BAR_MAX_BACKUP configuration parameter 1-73
- BAR_NB_XPORT_COUNT configuration parameter 1-73
- bar_object table 2-7
- BAR_PERFORMANCE configuration parameter 1-73
- BAR_PROGRESS_FREQ configuration parameter 1-73
- BAR_RETRY configuration parameter 1-73
- bar_server table 2-7
- BAR_XFER_BUF_SIZE configuration parameter 1-73
- BEGCOM logical-log record 5-4
- BEGIN logical-log record 5-4
- beginlg field 15-164
- BEGPREP logical-log record 5-4
- BEGWORK logical-log record 5-4
- BFRMAP logical-log record 5-5
- Big-remainder page 4-15
- Binding CPU virtual processors D-2
- Bitmap page
 - blobspace 4-23
 - tblspace tblspace 4-6
- bitvector field 5-6
- BLDCL logical-log record 5-5
- bldutil.sh script 2-2, A-4
- Blobpage
 - average fullness statistics 7-5, 7-12
 - size, blobpage 4-21
 - size, specifying 4-21, 14-6
 - structure
 - and storage 4-21, 4-23
 - dbspace blobpage 4-21
- Blobspaces
 - adding chunk 14-3
 - bit-map page 4-23
 - blobpage structure 4-23
 - blobspace mirror chunk, structure 4-3
 - blobspace structure 4-21
 - creating 14-6
 - dropping blobspaces 14-20
 - dropping chunk 14-18
 - free-map page
 - defined 4-23
 - location in blobspace 4-21
 - role in blobpage logging 4-23
 - tracked by bitmap 4-23
 - maximum number 14-6, 14-7, 14-16
 - mirroring, ending 14-23
 - mirroring, starting 14-21
 - naming conventions 14-6
- Blobspaces (*continued*)
 - page types 4-22
 - restriction
 - dropping 14-20
 - simple-large-object storage 4-22
- Blocking
 - database server 11-4
- BLOCKTIMEOUT configuration parameter 1-16
- BMAP2TO4 logical-log record 5-5
- BMAPFULL logical-log record 5-5
- BSPADD logical-log record 5-5
- BTCPYBCK logical-log record 5-5
- BTMERGE logical-log record 5-5
- BTSCANNER configuration parameter 1-16
- BTSHUFFL logical-log record 5-5
- BTSPLIT logical-log record 5-6
- Buffer pools 14-10
 - 64-bit addressing 1-19
 - adding a pool 13-4
 - creating a pool 13-4
 - smart large objects 1-19
- Buffered
 - disk space examples 14-3
- BUFFERING tag
 - in -Df option 14-13
- BUFFERPOOL configuration parameter 1-17
- Buffers
 - access-level flag bits 15-10
 - page-type codes 15-12, 15-166
- BUFFERS discontinued configuration parameter D-3
- buffers field in the BUFFERPOOL configuration parameter 1-19
- buildsmi script
 - error log message E-17
 - failure A-4
 - initializing database server 2-1
- buildsmi_out file A-4
- buildsmi.out file A-4

C

- Cache
 - SQL statements, printing 15-29
- CDR logical-log record 5-6
- CDR_DBSPACE configuration parameter 1-48
- CDR_DSLOCKWAIT configuration parameter 1-49
- CDR_ENV configuration parameter 1-49
- CDR_EVALTHREADS configuration parameter 1-49
- CDR_MAX_DYNAMIC_LOGS configuration parameter 1-49
- CDR_NIFCOMPRESS configuration parameter 1-49
- CDR_QDATA_SBSPACE configuration parameter 1-49
- CDR_QHDR_DBSPACE configuration parameter 1-49
- CDR_QUEUEMEM configuration parameter 1-49
- CDR_SERIAL configuration parameter 1-49
- CDR_SUPPRESS_ATSRISWARN configuration parameter 1-49
- CHALLOC
 - logical-log record 5-6
 - record subtype (SBLOB) 5-14
- Change
 - physical log
 - size and location 13-3
 - text editor 13-4
- Changing
 - ONCONFIG in ISA 1-10
 - sbspace attributes 14-17
- changing to 11-14

- CHCOMBINE
 - logical-log record 5-6
 - record subtype, SBLOB 5-14
- Checkpoint
 - CKPTINTVL configuration parameter 1-21
 - disabling I/O errors 1-74
 - statistics 2-8
 - warm restores performance 1-73
- CHFREE
 - logical-log record 5-6
 - record subtype, SBLOB 5-14
- CHKADJUP log record 5-7
- CHPHYLOG log record 5-7
- CHRESERV logical-log record 5-7
- CHSPLIT
 - logical-log record 5-7
 - record subtype, SBLOB 5-14
- Chunks
 - changing mirroring status 14-25
 - checking for overlap 7-11
 - free list, checking with oncheck 7-5, 7-10
 - free-list page 4-2, 4-3
 - initial chunk of dbspace 4-1
 - initial mirror offset 1-67
 - maximum number 14-3
 - monitoring 2-8
 - structure
 - additional dbspace chunk 4-2
 - initial dbspace chunk 4-2
 - mirror chunk 4-3
 - using a symbolic link for the pathname 1-67, 1-84
- CINDEX logical-log record 5-7
- Cipher
 - encryption 1-44
- CKPOINT logical-log record 5-7
- CKPTINTVL configuration parameter 1-21
- CLEANERS configuration parameter 1-22
- Client
 - killing sessions (onmode -z) 11-23
 - results of connection D-9
 - specifying dbservername 1-26
 - USEOSTIME configuration parameter 1-109
- CLR logical-log record 5-7
- CLUSIDX logical-log record 5-7
- COARSELOCK log record 5-7
- Cold restore
 - number of recovery threads 1-72
- COLREPAI logical-log record 5-7
- command_history table 3-5
- Commands
 - onstat -g cat 15-34
 - onstat -g ddr 15-45
 - onstat -g dss 15-52
 - onstat -g dtc 15-52
 - onstat -g grp 15-57
 - onstat -g nif 15-78
 - onstat -g que 15-95
 - onstat -g rcv 15-98
 - onstat -g rep 15-101
 - onstat -g rqm 15-102
 - onstat utility 15-102
- COMMIT logical-log record 5-7
- Compactness of index page 1-52
- COMTAB logical-log record 5-7
- COMWORK log record 5-7
- concdr.out file E-61
- concdr.sh script A-1, A-4, E-61
- Configurable page size 1-17
- Configuration file
 - displaying settings 1-2
 - format 1-1
 - preparing 1-2
 - processing A-8
 - warning about modifying onconfig.std 1-2, A-8
- Configuration parameter
 - ENCRYPT_MAC 1-46
 - ENCRYPT_MACFILE 1-46
- configuration parameter settings 1-16
- configuration parameters
 - WSTATS 1-114
- Configuration parameters
 - AC_DEBUG 1-10
 - AC_IXBAR 1-10
 - AC_LTAPBLOCK 1-10
 - AC_LTAPDEV 1-10
 - AC_MSGPATH 1-11
 - AC_SCHEMA 1-11
 - AC_STORAGE 1-11
 - AC_TAPEBLOCK 1-11
 - AC_TAPEDEV 1-11
 - AC_TIMEOUT 1-11
 - AC_VERBOSE 1-11
 - ADMIN_MODE_USERS 1-11
 - ADMIN_USER_MODE_WITH_DBSA 1-12
 - ADTERR 1-12
 - ADTMODE 1-12
 - ADTPATH 1-12
 - ADTSIZE 1-12
 - AFCRASH 1-54, A-7
 - AFF_NPROCS D-2
 - AFF_SPROC D-2
 - ALARMPROGRAM 1-12, C-1
 - ALLOW_NEWLINE 1-13
 - ALRM_ALL_EVENTS 1-13
 - attributes 1-9
 - AUTO_AIOVPS 1-14
 - AUTO_CKPTS 1-14
 - AUTO_LRU_TUNING 1-15
 - AUTO_REPREPARE 1-15
 - BAR_ACT_LOG 1-73
 - BAR_BSALIB_PATH 1-73
 - BAR_DEBUG 1-73
 - BAR_DEBUG_LOG 1-73
 - BAR_HISTORY 1-73
 - BAR_MAX_BACKUP 1-73
 - BAR_NB_XPORT_COUNT 1-73
 - BAR_PERFORMANCE 1-73
 - BAR_PROGRESS_FREQ 1-73
 - BAR_RETRY 1-73
 - BAR_XFER_BUF_SIZE 1-73
 - BLOCKTIMEOUT 1-16
 - BTSCANNER 1-16
 - BUFFERPOOL 1-17
 - CDR_DBSPACE 1-48
 - CDR_DSLOCKWAIT 1-49
 - CDR_ENV 1-49
 - CDR_EVALTHREADS 1-49
 - CDR_MAX_DYNAMIC_LOGS 1-49
 - CDR_NIFCOMPRESS 1-49
 - CDR_QDATA_SBSpace 1-49
 - CDR_QHDR_DBSPACE 1-49
 - CDR_QUEUEMEM 1-49
 - CDR_SERIAL 1-49
 - CDR_SUPPRESS_ATRISWARN 1-49

Configuration parameters *(continued)*

changing a value 1-10
 CKPTINTVL 1-21
 CLEANERS 1-22
 CONSOLE 1-22
 current default values 1-1
 DATASKIP 1-22
 DB_LIBRARY_PATH 1-24
 DBCREATE_PERMISSION 1-23
 DBSERVERALIASES 1-25, 11-6
 DBSERVERNAME 1-26, 11-6
 DBSPACETEMP 1-26
 DD_HASHMAX 1-28
 DD_HASHSIZE 1-29
 DEADLOCK_TIMEOUT 1-30
 DEF_TABLE_LOCKMODE 1-30
 DIRECT_IO 1-31
 DIRECTIVES 1-31
 DISABLE_B162428_XA_FIX 1-32
 DRAUTO 1-33
 DRDA_COMMBUFFSIZE 1-32
 DRIDXAUTO 1-34
 DRINTERVAL 1-34
 DRLOSTFOUND 1-35
 DRTIMEOUT 1-35
 DS_HASHSIZE 1-36, 1-39
 DS_MAX_QUERIES 1-36, 11-10
 DS_MAX_SCANS 1-37, 11-10
 DS_NONPDQ_QUERY_MEM 1-38
 DS_POOLSIZ 1-38
 DS_TOTAL_MEMORY 1-39, 11-10
 DUMPCNT 1-40
 DUMPCORE 1-41
 DUMPDIR 1-41
 DUMPGCORE 1-42
 DUMPSHMEM 1-42
 DYNAMIC_LOGS 1-43
 ENCRYPT_CDR 1-49
 ENCRYPT_CIPHERS 1-44
 ENCRYPT_HDR 1-45
 ENCRYPT_MACFILE 1-49
 ENCRYPT_SMX 1-47
 ENCRYPT_SWITCH 1-48, 1-49
 EXPLAIN_STAT 1-50
 EXT_DIRECTIVES 1-50
 EXTSHMADD 1-51
 FAILOVER_CALLBACK 1-51
 FAST_RESTART_CKPT_FUZZYLOG D-3
 FAST_RESTART_PHYSLOG D-4
 FASTPOLL 1-52
 FILLFACTOR 1-52, 4-20
 HA_ALIAS 1-52
 HETERO_COMMIT 1-53
 IFX_EXTEND_ROLE 1-53
 IFX_FOLDVIEW 1-53
 IMCLOG 1-66
 IMCTRANSPTS 1-66
 IMCWORKERDELAY 1-66
 IMCWORKERTHREADS 1-66
 ISM_DATA_POOL 1-54, 1-73
 ISM_LOG_POOL 1-54, 1-73
 JDKVERSION D-5
 JVMTHREAD 1-54
 JVPCLASSPATH 1-55
 JVPDEBUG 1-54, A-7
 JVPHOME 1-54
 JVPJAVAHOME 1-54

Configuration parameters *(continued)*

JVPJAVALIB 1-54
 JVPJAVAVM 1-54
 JVPLOGFILE 1-54, A-7
 JVPPROPFILE 1-54
 LIMITNUMSESSIONS 1-55
 list of 1-3
 LISTEN_TIMEOUT 1-56
 LOCKS 1-57
 LOG_INDEX_BUILDS 1-59
 LOGBUFF 1-57
 LOGFILES 1-58
 LOGSIZE 1-60
 LTAPEBLK 1-60
 LTAPEDEV 1-61
 LTAPESIZE 1-61
 LTXEHW 1-62
 LTXHWM 1-63
 MAX_FILL_DATA_PAGES 1-63
 MAX_INCOMPLETE_CONNECTIONS 1-64
 MAX_PDQPRIORITY 1-65, 11-10
 MIRROR 1-66
 MIRROROFFSET 1-67
 MIRRORPATH 1-67
 MSG_DATE 1-68
 MSGPATH 1-68
 MULTIPROCESSOR 1-69
 NETTYPE 1-70
 NUMAIOVPS D-8
 NUMCPUVPS D-9
 OFF_RECVRY_THREADS 1-72
 ON_RECVRY_THREADS 1-73
 ON-Bar utility, types of 1-73
 ONDBSPACEDOWN 1-74, 11-15
 ONLIDX_MAXMEM 1-75
 OPCACHEMAX 1-75
 OPT_GOAL 1-76
 OPTCOMPIND 1-76
 PC_HASHSIZE 1-78
 PC_POOLSIZ 1-78
 PHYSBUFF 1-78
 PHYSDBS D-9
 PHYSFILE 1-79
 PLOG_OVERFLOW_PATH 1-79
 QSTATS 1-80
 RA_PAGES 1-80
 RA_THRESHOLD 1-81
 RESIDENT 1-82, 11-15
 RESTARTABLE_RESTORE 1-82
 ROOTOFFSET 1-84
 ROOTPATH 1-83, 1-84
 ROOTSIZE 1-85
 RTO_SERVER_RESTART 1-85
 SBSPACENAME 1-86, 1-102
 SBSPACETEMP 1-87, 14-12
 SDS_ENABLE 1-88
 SDS_PAGING 1-88
 SDS_TEMPDBS 1-89
 SDS_TIMEOUT 1-89
 SECURITY_LOCALCONNECTION 1-90
 SERVERNUM 1-90
 setting decision-support with onmode 11-10
 SHMADD 1-90
 SHMBASE 1-91
 SHMNOACCESS 1-92
 SHMTOTAL 1-92
 SHMVIRT_ALLOCSEG 1-93

Configuration parameters (*continued*)

- SHMVIRTSIZE 1-94
- SINGLE_CPU_VP 1-95
- SQLTRACE 1-95
- SSL_KEYSTORE_LABEL 1-96
- STACKSIZE 1-97
- STAGEBLOB 1-97
- STMT_CACHE 1-97
- STMT_CACHE_HITS 1-98, 11-19
- STMT_CACHE_NOLIMIT 1-99
- STMT_CACHE_NUMPOOL 1-99
- STMT_CACHE_SIZE 1-100
- STORAGE_FULL_ALARM 1-100
- summary 1-3, 1-9
- SYSALARMPROGRAM 1-101
- SYSSBSPACENAME 1-102
- TAPEBLK 1-103
- TAPEDEV 1-103
- TAPESIZE 1-104
- TBLSPACE_STATS 1-105
- TBLTBLFIRST 1-105
- TBLTBLNEXT 1-105
- TEMPTAB_NOLOG 1-106
- TXTIMEOUT 1-107, 11-23
- UNSECURE_ONSTAT 1-107
- UPDATABLE_SECONDARY 1-81
- use
 - Data-replication screen 12-3
 - Initialization screen 12-3
 - PDQ screen 12-3
- USELASTCOMMITTED 1-107
- USEOSTIME 1-108
- viewing Enterprise Replication settings 15-36
- VP_MEMORY_CACHE_KB 1-109
- VPCLASS 1-55, 1-110, 11-17

Connection Manager

- configuring 16-1
- failover processing 16-1
- oncmism 16-1
- tables
 - syscmism 2-10
 - syscmismsla 2-10
 - syscmsmtab 2-11

CONSOLE configuration parameter 1-22

controlling with onmode -C 11-4

Conversion messages

- database server E-49, E-61
- Enterprise Replication E-61, E-65

Core dump

- contained in core file A-5
- DUMPCORE parameter 1-41

core.pid.cnt file 1-42

CPU

- time tabulated 15-150

CPU virtual processor

- binding D-2
- SINGLE_CPU_VP parameter 1-95

Create Dbspace with On-Monitor 12-3

CREATE FUNCTION statement 1-112

CREATE INDEX statement

- using FILLFACTOR 1-52

CREATE record subtype (SBLOB) 5-15

Creating

- buffer pool 13-4
- curllog field 15-164

D

daemon.log A-7

Data distributions

- sbspaces 1-102

Data pages

- number to read ahead 1-80
- oncheck -cd 7-8
- oncheck -cD 7-8
- oncheck -cd and -cD 7-4

Data replication

- ENCRYPT_CIPHERS 1-44
- ENCRYPT_HDR 1-45
- ENCRYPT_MAC 1-46
- ENCRYPT_MACFILE 1-46
- ENCRYPT_SMX 1-47
- ENCRYPT_SWITCH 1-48
- flush interval 1-34
- information in sysdri table 2-13
- lost-and-found file 1-35
- wait time for response 1-35

Data row

- big-remainder page 4-15
- forward pointer 4-13
- home page 4-13, 4-15
- locating the row 4-13
- rowid 4-13
- storage strategies 4-12
- storing data on a page 4-14
- TEXT and BYTE data descriptor 4-22

Data-dictionary cache 1-29

Data-distribution cache

- specifying
 - entries 1-38
 - hash buckets 1-36

Database servers

- blocking 11-4
- bringing online from quiescent mode 11-12, 11-13
- name 1-25
- parallel database query 11-17
- quiescent mode 11-12, 11-13
- remote 2-31
- restarting 1-9
- shutting down 11-12, 11-13
- unblocking 11-4

Database tblspace

- entries 4-7
- location in root dbspace 4-1, 4-6
- relation to systable 4-27
- structure and function 4-6
- tblspace number 4-6

Databases

- effect of creation 4-27
- locale, in sysdbslocale table 2-12
- owner, in sysmaster database 2-11
- sysdatabases table 2-11

DATASKIP configuration parameter

- defined 1-22
- using onspaces -f 14-21

DB_LIBRARY_PATH configuration parameter 1-24

DBC_CREATE_PERMISSION configuration parameter 1-23

DBSERVERALIASES configuration parameter

- defined 1-25
- using onmode -d 11-6

DBSERVERNAME configuration parameter

- defined 1-26
- using onmode -d 11-6

- dbspaces
 - adding chunk 14-3
 - blobpage structure 4-21
 - creating
 - with onspaces 14-8
 - dropping
 - chunk 14-18
 - with onspaces 14-20
 - ending mirroring 14-23
 - list of structures contained 4-2
 - maximum number 14-6, 14-7, 14-16
 - modifying with onspaces 14-21
 - monitoring with SMI 2-12
 - naming conventions 14-8
 - root name 1-83
 - simple-large-object storage 4-22
 - starting mirroring 14-21
 - storage 4-1
 - structure
 - additional dbspace chunk 4-2
 - chunk free-list page 4-3
 - dbspace 4-1, 4-2
 - mirror chunk 4-3
 - nonroot dbspace 4-2
 - tblspace tblspace 4-4
- DBSPACETEMP configuration parameter 1-26
- DD_HASHMAX configuration parameter 1-28
- DD_HASHSIZE configuration parameter 1-29
- Deadlock 1-30
- DEADLOCK_TIMEOUT configuration parameter 1-30
- Decision-support queries
 - DS_MAX_QUERIES configuration parameter 1-36
 - DS_TOTAL_MEMORY configuration parameter 1-39
 - gate information 15-76
 - gate numbers 15-76
 - MAX_PDQPRIORITY configuration parameter 1-65
 - setting parameters with onmode 11-10
- DEF_TABLE_LOCKMODE configuration parameter 1-30
- Default configuration file 1-2, A-8
- DELETE
 - logical-log record 5-8
 - record subtype, SBLOB 5-15
- DELITEM logical-log record 5-8
- DERASE logical-log record 5-8
- Descriptor, TEXT and BYTE data 4-22
- Diagnostic
 - using onmode B-1
- DINDEX logical-log record 5-8
- DIRECT_IO configuration parameter 1-31
- DIRECTIVES configuration parameter 1-31
- Disabilities, visual
 - reading syntax diagrams G-1
- disability G-1
- DISABLE_B162428_XA_FIX configuration parameter 1-32
- Disabling SQL statement cache 11-11
- Disk I/O
 - buffers 1-19
 - PDQ resources 1-65
- Disk page
 - page compression 4-15
 - storing data on a page 4-14
 - structure
 - blobpage blobpage 4-8
 - dbspace page 4-12
 - types of pages in an extent 4-7, 4-9
- Disk space
 - allocating
 - for system catalogs 4-27
 - when a database is created 4-27
 - when a table is created 4-28
 - chunk free-list page 4-3
 - initializing (oninit -i) 9-1
 - list of structures 4-1
 - maximum chunk size 14-3, 14-6, 14-9, 14-12, 14-16
 - page compression 4-15
 - tracking available space in
 - blobpage 4-23
 - chunk 4-3
- Distributed transactions
 - killing 11-23
- Dotted decimal format of syntax diagrams G-1
- dr.lostfound file 1-35
- DRAUTO configuration parameter 1-33
- DRDA_COMMBUFFSIZE 1-32
- DRIDXAUTO configuration parameter 1-34
- DRINTERVAL configuration parameter 1-34
- DRLOSTFOUND configuration parameter 1-35
- DROP DISTRIBUTIONS keyword 1-102
- DRPBSP logical-log record 5-8
- DRPCHK logical-log record 5-8
- DRPDBS logical-log record 5-8
- DRPLOG logical-log record 5-8
- DRTIMEOUT configuration parameter 1-35
- DS_HASHSIZE configuration parameter 1-36, 1-39
- DS_MAX_QUERIES
 - changing dynamically 11-20
- DS_MAX_QUERIES configuration parameter
 - changing value 11-10
 - defined 1-36
- DS_MAX_SCANS
 - changing dynamically 11-20
- DS_MAX_SCANS configuration parameter
 - changing value 11-10
 - defined 1-37
- DS_NONPDQ_QUERY_MEM
 - changing dynamically 11-20
- DS_NONPDQ_QUERY_MEM configuration parameter 1-38
- DS_POOLSIZE configuration parameter 1-38
- DS_TOTAL_MEMORY
 - changing dynamically 11-20
- DS_TOTAL_MEMORY configuration parameter
 - changing value 11-10
 - defined 1-39
- DUMPCNT configuration parameter 1-40
- DUMPCORE configuration parameter 1-41
- DUMPDIR configuration parameter
 - af.xxx assertion failure file A-3
 - defined 1-41
 - gcore file A-5
 - shmem file A-9
- DUMPGCORE configuration parameter 1-42
- DUMPSHMEM configuration parameter 1-42
- Dynamic log messages E-65, E-67
- DYNAMIC_LOGS configuration parameter 1-43

E

- Emergency boot file A-5
- Enabling SQL statement cache 11-11
- ENCRYPT_CDR configuration parameter 1-49
- ENCRYPT_CIPHERS configuration parameter 1-44, 1-49
- ENCRYPT_HDR configuration parameter 1-45

ENCRYPT_MAC configuration parameter 1-46, 1-49
 ENCRYPT_MACFILE configuration parameter 1-46, 1-49
 ENCRYPT_SMX configuration parameter 1-47
 ENCRYPT_SWITCH configuration parameter 1-48, 1-49
 Encrypting or decrypting files 17-1
 Encryption

- cipher renegotiation 1-48
- high-availability data replication 1-45
- MAC files, specifying 1-46
- message authentication code generation 1-46
- specifying ciphers and modes 1-44

 ENDTRANS logical-log record 5-8
 Enterprise Replication

- CDR log record 5-6
- configuration parameters 1-48
- messages E-61, E-65
- renaming spaces 14-25

 Environment configuration file A-6
 Environment variables

- AC_CONFIG 1-10
- AFDEBUG A-7
- defined 1-10
- IFX_DEF_TABLE_LOCKMODE 1-30
- IFX_DIRECTIVES 1-31, 1-50
- IFX_XASTDCOMPLIANCE_XAEND 1-32
- IMCADMIN 1-66
- IMCCONFIG 1-66
- IMCSERVER 1-66
- INFORMIXDIR 1-66
- INFORMIXOPCACHE 1-75
- INFORMIXSERVER 1-26, 1-66, 11-19
- INFORMIXSQLHOSTS 1-66
- ONCONFIG
 - defined A-8
 - onstat -c 15-12
 - setting 1-2
- OPTCOMPIND 1-76
- SERVER_LOCALE 15-53
- STMT_CACHE 1-98, 11-11
- viewing Enterprise Replication settings 15-36

 ERASE logical-log record 5-8
 Event alarm

- ALARMPROGRAM parameter 1-12, C-1
- automatic log backup 1-12, C-1
- class ID parameter C-4
- class message parameter C-4
- creating 2-3
- defined C-1
- event severity codes C-3
- exit code C-2
- RS secondary server C-5
- SD secondary server C-6
- using ex_alarm.sh C-1
- writing your own script C-1

 Event severity codes C-3
 ex_alarm.sh script 1-13, C-1
 Exclusive-access, high-water mark 1-62
 EXIT_STATUS exit code C-2
 Exit, return codes on 15-167
 EXPLAIN_STAT

- changing dynamically 11-20

 EXPLAIN_STAT configuration parameter 1-50
 EXT_DIRECTIVES configuration parameter 1-50
 EXTEND record subtype, SBLOB 5-15
 EXTENT_SIZE tag 14-14
 Extents

- automatic doubling of size 4-11

Extents (*continued*)

- default size 4-7
- disk page types 4-7, 4-9
- merging 4-11
- next-extent, allocating 4-10, 4-11
- procedure for allocating 4-10
- size
 - index fragments 4-7
 - initial extent 4-7
 - next extent 4-10
- structure 4-7
- sysextents table 2-14

 External backup

- commands 11-4

 EXTSHMADD configuration parameter 1-51
 Extspace

- creating 14-17
- dropping 14-20
- naming conventions 14-17
- specifying location 14-16
- sysextspaces table 2-14

F

Failover processing 16-1
 FAILOVER_CALLBACK configuration parameter

- defined 1-51

 FAST_RESTART_CKPT_FUZZYLOG configuration

- parameter D-3

 FAST_RESTART_PHYSLOG configuration parameter D-4
 FASTPOLL configuration parameter

- defined 1-52

 Files

- .conf.dbservername A-4
- .inf.servicename A-6
- .informix A-5
- .infos.dbservername A-6
- .infxdirs A-6
- javpprops 1-54, A-7
- ac_config.std A-4
- ac_msg.log A-3
- archecker configuration file A-4
- bar_act.log A-4
- buildsmi_out A-4
- buildsmi.out A-4
- core.pid.cnt 1-42, A-5
- default configuration file A-8
- dr.lostfound 1-35
- gcore 1-41, A-5
- informix.rc environment file A-6
- INFORMIXTMP directory A-6
- ISM logs A-7
- ISMVersion A-7
- JVM_vpid 1-54, A-7
- JVPLOG 1-54, A-7
- oncfg* A-8
- ONCONFIG A-8
- private environment file A-5
- shmem.pid.cnt 1-42
- shmem.xxx A-9
- sm_versions A-9
- summary of database server files A-1
- VP.servicename.nnx A-10
- xbsa.messages A-10

 FILLFACTOR configuration parameter

- control how indexes fill 4-20
- defined 1-52

- Flushing
 - data-replication buffer 1-34
 - SQL statement cache 11-11
- Force option, onspaces 14-20
- Forced residency
 - starting and ending with onmode 11-15
- Formula
 - quantum of memory 15-74
- Forward pointer
 - blob space blobpage 4-23
 - db space storage of simple large objects 4-22
 - defined 4-13
- Fragment
 - index 4-7
 - internal structure of tables 4-15
 - rowids 4-13
 - table, using primary keys 4-14
 - turning DATASKIP ON or OFF for 14-21
 - warning returned when skipped during query 1-23
- FREE_RE logical-log record 5-8
- Free-map page, blob space 4-23
- Freeing
 - blob pages 15-22
 - unused memory segments 11-11
- Functional index 4-20

G

- Gateway transactions 1-53
- gcore
 - file A-5
 - utility 1-40, 1-42
- General Page Manager 15-150
- Global transactions
 - using onstat -G 15-136
 - using onstat -x 15-165

H

- HA_ALIAS
 - changing dynamically 11-20
- HA_ALIAS configuration parameter
 - defined 1-52
- Hash buckets
 - data-dictionary cache 1-29
 - data-distribution cache
 - specifying hash buckets 1-36
- HDELETE logical-log record 5-8
- HDRUPD record subtype, SBLOB 5-15
- HETERO_COMMIT configuration parameter 1-53
- Heterogeneous-commit transactions 1-53
- HEURTX logical-log record 5-9
- High-water mark, transaction 1-63
- HINSERT logical-log record 5-9
- Home page 4-13, 4-15
- HUPAFT logical-log record 5-9
- HUPBEF logical-log record 5-9
- HUPDATE logical-log record 5-9

I

- I/O
 - lightweight 14-13
- IBM Informix Server Administrator
 - adding or dropping logs 1-58, 1-59
 - defined 6-3

- IBM Informix Server Administrator (*continued*)
 - setting ONCONFIG parameters 1-2, 1-10
- IBM Informix STAR queries 15-163
- IBM Informix Storage Manager
 - catalog A-6
 - ISMversion file A-7
 - logs A-7
 - sm_versions file A-9
- Identifier, defined 1-26
- IDXFLAGS logical-log record 5-9
- ifx_allow_newline() routine 1-13
- IFX_DEF_TABLE_LOCKMODE environment variable 1-30
- IFX_DIRECTIVES environment variable 1-31, 1-50
- IFX_EXTEND_ROLE configuration parameter 1-53
- IFX_FOLDVIEW configuration parameter 1-53
- ifx_lo_specset_estbytes function 14-14, 14-15
- ifx_lo_stat function 14-18
- IFX_XASTDCOMPLIANCE_XAEND environment variable 1-32
- illsrta.xx file A-5
- IMCADMIN environment variable 1-66
- IMCCONFIG environment variable 1-66
- IMCLOG configuration parameter 1-66
- IMCSERVER environment variable 1-66
- IMCTRANSSPORTS configuration parameter 1-66
- IMCWORKERDELAY configuration parameter 1-66
- IMCWORKERTHREADS configuration parameter 1-66
- Index
 - branch node 4-16
 - configuration 4-20
 - duplicate key values 4-19
 - functional 4-20
 - how created and filled 4-17
 - item described 4-17
 - key value locking 4-19
 - leaf node 4-16
 - reuse of freed pages 4-20
 - root node 4-16
 - structure of B-tree 4-16
- Index item
 - calculating the length of 4-20
 - defined 4-17
- Index page
 - compactness 1-52
 - creation of first 4-17
 - effect of creation 4-17
 - structure 4-16
- Information
 - SQL profile 2-33
- informix.rc environment file A-6
- INFORMIXDIR environment variable 1-66
- INFORMIXOPCACHE environment variable 1-75
- INFORMIXSERVER environment variable 1-26, 1-66, 11-19
- INFORMIXSQLHOSTS environment variable 1-66
- INFORMIXTMP directory A-6
- Initializing
 - disk structures 1-9, 4-1
 - shared memory 1-9
- INSERT logical-log record 5-9
- InstallServer.log file A-6
- Interrupt signal 6-1
- Interval
 - checkpoint 1-21
- ISA. 1-58
- ISAM calls tabulated 15-149
- ISM_DATA_POOL configuration parameter 1-54, 1-73
- ISM_LOG_POOL configuration parameter 1-54, 1-73

ism_startup command A-9
ISM.
See IBM Informix Storage Manager. A-6
ISMversion file A-7
ISOSPCOMMIT log record 5-9

J

Java configuration parameters 1-54
Java virtual processor 1-55
JDBC configuration parameter 1-54
JDKVERSION discontinued configuration parameter D-5
JDKVERSION parameter D-5
JVM_vpid file 1-54, A-7
JVMTHREAD configuration parameter 1-54
JVPCLASSPATH configuration parameter 1-55
JVPDEBUG configuration parameter 1-54, A-7
JVPHOME configuration parameter 1-54
JVPJAVAHOME configuration parameter 1-54
JVPJAVALIB configuration parameter 1-54
JVPJAVAVM configuration parameter 1-54
JVPLOG file 1-54, A-7
JVPLOGFILE configuration parameter 1-54, A-7
JVPPROFILE configuration parameter 1-54

K

Key value
checking order with oncheck 7-5, 7-10
duplicates 4-19
locking 4-19
Key-only
inserting entries 1-98, 11-19
Killing a session 11-23

L

Large chunk mode 11-3
Latch
displaying with onstat -s 15-5, 15-158
identifying the resource controlled 15-158
LBU_PRESERVE discontinued configuration parameter D-5
LCKLVL logical-log record 5-9
LG_ADDBPPOOL logical-log record 5-10
LG_CDINDEX log record 5-6
Licensed users, maximum allowed E-25
Lightweight I/O 14-13
LIMITNUMSESSIONS
changing dynamically 11-20
LIMITNUMSESSIONS configuration parameter 1-55
Limits
SQL statement cache size 1-99, 11-19
virtual processors 11-16
Linking, name of root dbspace 1-84
LISTEN_TIMEOUT
changing dynamically 11-20
LISTEN_TIMEOUT configuration parameter 1-56
Locales xx
Location, extspace 14-16
Lock
buffer-access-level flag bits 15-10
information in syslocks table 2-17
key-value 4-19
maximum time to acquire 1-30
monitoring with onstat -k 15-4, 15-141
multiprocessor 1-69

Lock (*continued*)
oncheck options 7-1
type codes 15-142
Lock mode, page or row 1-30
LOCK_MODE tag 14-14
LOCKS configuration parameter 1-57
Log position 15-164
log_full scripts 1-12, C-1
LOG_INDEX_BUILDS configuration parameter 1-59
LOGBUFF configuration parameter 1-57
LOGFILES configuration parameter 1-58
Logging
blobspace free-map page 4-23
flags for mode 4-7
LOGGING tag 14-14
Logical log
adding files 13-2
backup
alarm triggered 1-12, C-1
checking consistency 7-11
dropping files 13-2
file
created during initialization 1-58
displaying contents 10-1
log position 15-164
moving 13-3
reading the log file 10-1
size 1-60
switching with onmode 11-15
files
maximum number 13-2
minimum number 13-3
in root dbspace 4-1
log position 15-164
maximum size 15-164
monitoring with SMI 2-18
onparams
adding files 13-2
dropping files 13-2
record
additional columns 5-3
checkpoint 5-2
displaying 10-1
distributed transactions 5-2
DROP TABLE operation 5-1
generated by rollback 5-1
header columns 5-2
types 5-3, 5-14
specifying file size 13-2
Logical page contents
displaying with oncheck 7-15
Logical recovery, number of threads 1-73
Logical-log buffer and LOGBUFF configuration parameter 1-57
logmessage table E-2
logposit field 15-164
LOGSIZE configuration parameter 1-60
LOGSMAX discontinued configuration parameter D-5
Long transaction
high-water mark 1-63
LTXEHWM 1-62
LTXHWM 1-63
Loosely-coupled mode 15-165
LRU
changing dynamically 11-21
LRU queues
displaying with onstat -R 15-5, 15-155

- LRU queues (*continued*)
 - FLRU queues 15-155
 - MLRU queues 15-155
 - modified pages, percentage 1-20
- LRU_MAX_DIRTY discontinued configuration parameter D-6
- lru_max_dirty field in the BUFFERPOOL configuration parameter 1-20
- LRU_MIN_DIRTY discontinued configuration parameter D-6
- lru_min_dirty field in the BUFFERPOOL configuration parameter 1-20
- LRUS discontinued configuration parameter D-7
- lrus field in the BUFFERPOOL configuration parameter 1-19
- LTAPEBLK configuration parameter 1-60
- LTAPEDEV configuration parameter 1-61
- LTAPESIZE configuration parameter 1-61
- LTXEHWM configuration parameter 1-62
- LTXHWM configuration parameter 1-63

M

- MAX_FILL_DATA_PAGES configuration parameter 1-63
- MAX_INCOMPLETE_CONNECTIONS
 - changing dynamically 11-20
- MAX_INCOMPLETE_CONNECTIONS configuration parameter 1-64
- MAX_PDQPRIORITY
 - changing dynamically 11-20
- MAX_PDQPRIORITY configuration parameter
 - changing value 11-10
 - defined 1-65
- MaxConnect
 - configuration parameters 1-65
 - DBSERVERALIASES configuration parameter 1-25
 - DBSERVERNAME configuration parameter 1-26
 - environment variables 1-66
 - NETTYPE configuration parameter 1-72
 - network statistics 15-28
 - onstat -g imc 15-26
 - onstat -g nta 15-28
- Maximum number
 - chunks 14-3
 - storage spaces 14-6, 14-7, 14-11, 14-16
- Maximum user connections E-25
- Memory
 - freeing unused segments 11-11
 - pools, SQL statement cache 1-99
 - quantum allocated by MGM 15-74, 15-75, 15-76
 - specifying size of 1-75
- Memory Grant Manager
 - monitoring resources 15-74
- Memory information
 - SMI tables 2-19, 2-29
- Message authentication code files 1-46
- Message log
 - alphabetical listing of messages E-1
 - categories of messages E-2
 - defined A-7, E-1
 - displaying with onstat -m 15-4, 15-145
 - event alarms C-2
 - location 1-68
 - viewing messages E-1
- Messages
 - A-B E-2, E-4
 - assertion-failed E-2
 - C E-4, E-14
 - changing sbspace minimum extent size 14-14
 - conversion and reversion E-49, E-61

- Messages (*continued*)
 - D-E-F E-14, E-19
 - dynamic log E-65, E-67
 - Enterprise Replication E-61, E-65
 - G-H-I E-19, E-22
 - in-place ALTER TABLE E-18
 - J-K-L-M E-22, E-27
 - N-O-P E-27, E-36
 - onspaces E-33
 - Q-R-S E-36, E-41
 - sbspace metadata E-67, E-68
 - symbols E-48, E-49
 - T-U-V E-41, E-47
 - truncate table E-68
 - turning off smart-large-object logging 14-14
 - W-X-Y-Z E-47, E-48

Metadata

- area, structure 4-24
- checking with oncheck 7-1
- creating 14-11
- messages E-67, E-68
- size 14-11, 14-13
- specifying offset 4-24, 14-11
- specifying size 14-11
- temporary sbspace 1-87

MGM

- information 2-18
- mi_lo_decrefcount() function 14-18
- mi_lo_increfcount() function 14-18
- mi_lo_specset_estbytes() function 14-14, 14-15
- mi_lo_stat function() 14-18
- Microsoft Transaction Server
 - defined 15-163
 - onstat -x output 15-165
- MIN_EXT_SIZE tag 14-14
- Mirror chunk, structure 4-3
- MIRROR configuration parameter 1-66
- Mirroring
 - changing chunk status 14-25
 - enable flag 1-66
 - initial chunk 1-67
 - starting 14-21
 - stopping 14-23
- MIRROROFFSET configuration parameter 1-67
- MIRRORPATH configuration parameter 1-67

Modes

- encryption 1-44

Modified pages

- specifying percentage
 - LRU queue 13-5

Monitoring

- display environment variables 15-53
- distributed queries 15-163
- licensed user connections E-25
- MGM resources 15-74

- Moving logical-log files 13-3

MSG_DATE

- changing dynamically 11-20
- MSG_DATE configuration parameter 1-68
- MSGPATH configuration parameter 1-68

Multiprocessor computer

- AFF_SPROC configuration parameter D-2
- processor affinity 1-114
- MULTIPROCESSOR configuration parameter 1-69
- MVIDXND logical-log record 5-10

N

Name

- blobspace 14-6
- dbspace 14-8
- extspace 14-17
- sbspaces 14-11
- NETTYPE configuration parameter
 - defined 1-70
 - tuning example 1-70
- Network information
 - SMI tables 2-19, 2-20
- Newline character, quoted strings 1-13
- NEXT_SIZE tag 14-15
- Next-extent
 - allocation 4-10
 - allocation strategy 4-11
 - doubling of size 4-11
 - initial size 4-7
 - nonfragmented table 4-8
- no_log scripts 1-12
- NOAGE discontinued configuration parameter D-7
- Node, index
 - branch
 - creating 4-18
 - defined 4-16
 - what points to 4-18
 - checking horizontal and vertical nodes 7-5, 7-10
 - defined 4-17
 - leaf
 - contents 4-18
 - defined 4-16
 - pointer 4-18
 - root node
 - creating 4-17
 - defined 4-16
 - when fills 4-17
 - types 4-16
- Non-default page size
 - physical log 13-4
- NUMAIOVPS configuration parameter
 - defined D-8
- Number of page-cleaner threads 1-22
- NUMCPUVPS configuration parameter
 - defined D-9

O

- Object Explorer 6-3
- OFF_RECOVERY_THREADS configuration parameter 1-72
- Offset
 - mirrored chunk 14-11
 - size 14-3, 14-6, 14-9, 14-12, 14-16
- ON_RECOVERY_THREADS configuration parameter 1-73
- ON-Bar utility
 - activity log A-4
 - configuration parameters 1-73
 - emergency boot files A-5
 - sm_versions file A-9
 - system tables 2-6
 - xbsa.messages log A-10
- On-Monitor Create Dbsapce name 12-3
- ON-Monitor utility
 - Archive menu options 12-3
 - changing
 - database server mode 11-14
 - parameter values 1-10
- ON-Monitor utility (*continued*)
 - Dbspaces menu options 12-2
 - Diagnostics screen 12-3
 - displaying system page size 1-21
 - executing shell commands 12-2
 - Force-Ckpt menu options 12-3
 - help 12-1
 - Initialization screen 12-3
 - Logical-Logs menu options 12-3
 - Mode menu options 12-3
 - navigating 12-1
 - Parameters menu options 12-2
 - Performance screen 12-3
 - Shared-memory screen 12-3
 - Status menu options 12-2
 - using 12-1
- oncfg file
 - and onspaces 14-1
 - defined A-8
- oncheck utility
 - check-and-repair 7-1
 - defined 7-1
 - display reserved, physical-log, and logical-log pages 4-2
 - list of functions 7-1
 - locking 7-3
 - option descriptions 7-3
 - options
 - cc 7-8
 - cd 7-8
 - cD 7-8
 - ce 7-5, 7-10
 - ci 7-10
 - cl 7-10
 - cr 7-11
 - cR 7-11
 - cs 7-12
 - cS 7-12
 - n 7-2, 7-5
 - pB 7-12
 - pc 7-8
 - pd 7-13
 - pD 7-13
 - pe 4-1, 7-10
 - pk 7-14
 - pK 7-14
 - pl 7-14
 - pL 7-14
 - pp 7-15
 - pP 7-15
 - pr 1-2, 4-2, 7-17
 - pR 4-2, 7-17
 - ps 7-12
 - pS 7-12
 - pt 7-18
 - pT 7-18
 - u 7-19
 - x 7-19
 - y 7-2, 7-7
 - overview of functionality 7-1
 - suppressing messages 7-6
 - syntax 7-3
- oncmstm utility
 - managing high-availability servers 16-1
- ONCONFIG configuration file
 - changing parameter values 1-10
 - changing with ISA 1-10
 - defined A-8

ONCONFIG configuration file (*continued*)

- displaying 1-2, 15-4, 15-12
- format 1-1
- preparing 1-2
- templates 1-1
- white space 1-1

ONCONFIG environment variable

- ONCONFIG file A-8
- setting 1-2
- using onstat -c 15-12

onconfig.std file

- default value 1-9
- defined 1-1
- printing 1-1

ondblog utility 8-1

- BAR_ACT_LOG file 8-1
- defined 8-1

ONDBSPACEDOWN configuration parameter

- defined 1-74
- overriding WAIT mode 11-15

oninit utility

- option descriptions 9-4

options

- i 9-4
- j 9-1
- p 9-3
- s 9-3, 9-4
- S 9-3
- U 9-3
- w 9-2
- y 9-2

- starting database server 9-1

- temporary tables 9-3

ONLIDX_MAXMEM

- changing dynamically 11-20

ONLIDX_MAXMEM configuration parameter 1-75

onlog utility

- defined 10-1

filters for logical-log records

- displaying 10-3
- reading 5-6, 10-2

options

- b 10-3
- d 10-3
- l 5-6, 10-3
- n 10-3
- q 10-2
- t 10-3
- u 10-4
- x 10-4

onmode utility

- wf option 1-68
- wm option 1-68

adding

- shared-memory segment 11-3
- virtual processors 11-16

- administration mode 11-14

- blocking the database server 11-4

changing

- database server mode 11-12, 11-14
- DS_MAX_QUERIES 11-10
- DS_MAX_SCANS 11-10
- DS_TOTAL_MEMORY 11-10
- MAX_PDQPRIORITY 11-10
- shared-memory residency 11-15
- SQL statement cache usage 1-98, 11-11, 11-19
- defined 11-1

onmode utility (*continued*)

- forcing a checkpoint 11-4

- freeing memory segments 11-11

killing

- distributed transactions 11-23
- session 11-23

- marking disabled dbspace as down 11-15

options

- a 11-3
- BC 1 11-3
- BC 2 11-3
- c block 11-4
- c unblock 11-4
- d 11-6
- D 11-10
- e 1-98, 11-11
- F 11-11
- I B-1
- j 11-13
- j -U 11-14
- k 11-12, 11-13
- l 11-15
- m 11-12, 11-13
- M 11-10
- n 11-15
- O 11-15
- p 11-16
- Q 11-10
- r 11-15
- R 11-19
- s 11-12, 11-13
- S 11-10
- u 11-12, 11-13
- W 11-19
- Y 11-22
- y confirm action 11-2
- z 11-23
- Z 11-23

PDQ 15-76

- regenerating .infos file 11-19

- removing virtual processors 11-16

setting

- data replication type 11-6

- decision-support parameters 11-10

- Starting or ending forced residency 11-15

- switching logical-log files 11-15

- trapping errors B-1

- unblocking the database server 11-4

onparams utility 13-1

- adding logical-log file 13-2

- backing up changes to physical log 13-4

- changing physical log size and location 13-3

- defined 13-1

- dropping a logical-log file 13-2

- examples 13-6

onpassword utility

- encrypting or decrypting text files 17-1

onpload

- onstat -j utility 15-140

- onsnmp log file A-9

- onsocimc protocol 1-72

onspaces utility

- Df options 14-12

- adding a chunk to

- dbspace or blobspace 14-3

- sbspaces 14-4

- changing chunk status 14-25

onspaces utility (*continued*)
 changing sbspace defaults 14-15, 14-17
 cleaning up sbspaces 14-17
 creating a blobspace 14-5
 creating a temporary sbspace 14-11
 creating an extspace 14-16
 creating an sbspace 14-11
 defined 14-1
 dropping a blobspace, dbspace, extspace, or sbspace 14-20
 dropping a chunk 14-18
 ending mirroring 14-23
 forcing a drop 14-20
 options
 -a 14-3, 14-4
 -b 14-6
 -c 14-6, 14-7, 14-16
 -cl 14-18
 -d 14-18, 14-20
 -Df 14-12
 -f 14-21
 -g 14-6, 14-16
 -l 14-16
 -m 14-22
 -Mo 14-4, 14-11
 -Ms 14-4, 14-11
 -r 14-23
 -s 14-26
 -S 14-11
 -t 14-10, 14-12
 -x 14-17
 specifying DATASKIP 14-21
 starting mirroring 14-21
 onsrvapd.log file A-9
 onstat -g cpu 15-42
 onstat -g options 15-25
 onstat command 15-102
 onstat utility
 - option 15-6
 -- option 15-2, 15-7
 -a option 15-3, 15-9
 -b option 1-21, 15-3, 15-9
 -B option 15-4, 15-9, 15-10
 -c option 15-4, 15-12
 -C option 15-4, 15-13
 -d option 1-27, 15-4, 15-19
 -D option 15-4, 15-23
 -d update option 15-22
 -f option 1-23, 15-4
 -F option 1-22, 15-4, 15-24
 -g act option 15-30
 -g afr option 15-31
 -g all option 15-31
 -g ath option 15-31
 -g buf option 15-32
 -g cac agg option 15-25
 -g cac stmt option 15-25
 -g cat option 15-34
 -g cdr option 15-36
 -g ckp option 15-38
 -g cmsm option 15-40
 -g con option 15-41, 15-101, 15-102
 -g cpu option 15-42
 -g dbc option 15-43
 -g ddr command 15-45
 -g dic option 15-46
 -g dis option 15-47
 -g dll option 15-48
 -g dmp option 15-49
 -g dri option 15-50
 -g dsc option 15-51
 -g dss command 15-52
 -g dtc command 15-52
 -g env option 15-53, 15-54
 -g ffr option 15-55
 -g glo option 15-56
 -g glo options A-7
 -g grp command 15-57
 -g his option 15-62
 -g imc option 15-26
 -g ioa option 15-64
 -g iob option 15-65
 -g iof option 15-66
 -g iog option 15-67
 -g ioq option 15-67
 -g iov option 15-69
 -g ipl option 15-69
 -g lap option 15-71
 -g lmx option 15-71
 -g lsc option 15-72
 -g mem option 15-73
 -g mgm option 11-11, 15-74
 -g nbm option 15-77
 -g nif command 15-78
 -g nsc option 1-69, 15-79
 -g nsd option 15-81
 -g nss option 15-82
 -g nta option 15-28
 -g ntd option 15-83
 -g ntm option 15-83
 -g ntt option 15-84
 -g ntu option 15-84
 -g opn option 15-85
 -g option 15-4, 15-25
 -G option 15-4, 15-136
 -g pos option 15-87
 -g ppf option 1-105, 15-88
 -g prc option 15-89
 -g proxy option 15-90
 -g qst option 15-96
 -g que command 15-95
 -g que option 15-95
 -g rbm option 15-97
 -g rcv command 15-98
 -g rcv option 15-98
 -g rea option 15-101
 -g rep command 15-101
 -g rqm command 15-102
 -g rss option 15-104
 -g rwm option 15-107
 -g sch option 15-107
 -g sds option 15-108
 -g seg option 1-90, 15-112
 -g ses option 15-113
 -g sle option 15-117
 -g smb option 15-118
 -g smx option 15-119
 -g spi option 15-121
 -g sql option 15-122
 -g src option 15-124
 -g ssc all option 15-29
 -g ssc option 1-99, 15-29, 15-124
 -g ssc pool option 1-100, 15-29
 -g stk option 15-126

onstat utility (*continued*)

- g stm option 15-126
- g stq option 15-127
- g sts option 15-127
- g sync option 15-128
- g tpf option 15-130
- g ufr option 15-130
- g vpcache option 15-132
- g wai option 15-133
- g wmx option 15-134
- g wst option 15-134
- h option 15-4, 15-138
- i option 15-4, 15-139
- j option 15-139
- k option 1-56, 15-4, 15-141
- l option 1-57, 13-2, 15-4, 15-143
- m option 1-68, 15-4, 15-145, E-1
- o option 15-4
- O option 1-75, 15-4, 15-146
- options source_file 15-8
- p option 1-29, 15-148
- P option 15-5, 15-151
- pu option 15-2
- r option 15-5, 15-152
- R option 15-5, 15-155
- s option 15-5, 15-158
- t option 15-5, 15-159
- T option 15-159
- u option 15-5, 15-161
- x option 11-23, 15-5, 15-163
- X option 15-5, 15-165
- z option 15-5, 15-167
- defined 15-1
- displaying
 - chunk information 15-19
 - global transactions 15-136
 - ONCONFIG file 15-4
- freeing blobpages 15-22
- header 15-6
- monitoring
 - PDQ 15-74
- no options 15-6
- print file info about B-tree scanner subsystem and thread 15-13
- repeated execution
 - r option 15-5
 - seconds parameter 15-6
- syntax 15-2
- table of options 15-2
- terminating interactive mode or repeating sequence 15-139
- using SMI tables for onstat information 2-39

ontape utility

- LTAPEBLK, uses 1-60, 1-103
- LTAPEDEV, uses 1-61
- LTAPESIZE, uses 1-61
- TAPEDEV, uses 1-103
- TAPESIZE, uses 1-104

ontliimc protocol 1-72

onunload utility

- LTAPEBLK, uses 1-60, 1-103
- LTAPEDEV, uses 1-61
- LTAPESIZE, uses 1-61
- TAPEDEV, uses 1-103
- TAPESIZE, uses 1-104

OPCACHEMAX configuration parameter 1-75

OpenAdmin Tool for IDS 6-2

OPT_GOAL configuration parameter 1-76

OPTCOMPIND

- configuration parameter 1-76
- environment variables 1-76

Optical storage and STAGEBLOB configuration parameter 1-97

Optical Subsystem memory cache 15-4, 15-146, 15-147

Optimizing hash and nested-loop joins 1-76

P

Page

- bitmap page 4-22
- blobspace blobpage 4-22
- blobspace free-map page 4-22
- components of dbspace page 4-12
- compression 4-15
- dbspace blobpage 4-21
- dbspace page types 4-7, 4-9
- definition of full page 4-14
- free page, defined 4-8, 4-9
- page types in extent 4-7, 4-8
- reuse of index page 4-20
- size, shown with onstat -b 15-9
- structure and storage of 4-12

Page compression 4-15

Page header, length of 4-4

PAGE_CONFIG reserved page 1-2, 7-11

Page-cleaner threads

- codes for activity state 15-24
- monitoring activity 15-4, 15-24
- numbers 1-22

Parallel database queries

- gate information 15-76
- MGM resources 15-76
- monitoring resources 15-76
- monitoring resources allocated 15-76

Partnum field in systables 4-5

Passwords, encrypted, not shown in onstat -g sql 15-123

Pathname, specifying 14-7, 14-9, 14-12

PBDELETE logical-log record 5-10

PBINSERT logical-log record 5-10

PC_HASHSIZE configuration parameter 1-78

PC_POOLSIZ configuration parameter 1-78

PDELETE record subtype (SBLOB) 5-15

PDINDEX logical-log record 5-10

PDQ

- CPU VPs 11-17
- DS_MAX_QUERIES configuration parameter 1-36
- DS_MAX_SCANS configuration parameter 1-37
- DS_TOTAL_MEMORY configuration parameter 1-39
- information 2-18
- MAX_PDQPRIORITY configuration parameter 1-65
- PDQPRIORITY configuration parameter 1-65

Pending transaction 15-163

PERASE logical-log record 5-10

PGALTER logical-log record 5-10

PGMODE logical-log record 5-10

PH_ALERT table 3-3

PH_GROUP table 3-3

PH_RUN table 3-2

PH_TASK table 3-1

PH_THRESHOLD table 3-4

PHYSBUFF configuration parameter 1-78

PHYSDBS configuration parameter D-9

PHYSFILE configuration parameter 1-79

- Physical log
 - backing up
 - changes 13-4
 - files 13-4
 - changing
 - size and location 13-3
 - text editor 13-4
 - changing size 13-3
 - checking consistency 7-11
 - root dbspace 4-1
 - size 1-79
 - using non-default page size 13-4
- Physical-log buffer
 - dbspace location D-9
 - size 1-78
- PLOG_OVERFLOW_PATH configuration parameter 1-79
- PNGPALIGN8 log record 5-10
- PNLOCKID logical-log record 5-10
- PNSIZES logical-log record 5-11
- Pools
 - SQL statement cache 15-29
- PREPARE logical-log record 5-11
- Preventing long transactions 1-63
- Primary key, use in fragmented table 4-14
- Printing
 - diagnostics, onmode -I B-1
 - global transactions 15-136
 - MaxConnect information 15-26
 - network statistics 15-28
 - onconfig.std file 1-1
 - SQL statement cache 15-25, 15-29
 - user-defined aggregates 15-25
- Priority aging, of CPU virtual processors D-7
- Private environment file A-5
- Processor affinity
 - AFF_NPROCS configuration parameter D-1
 - AFF_SPROC configuration parameter D-2
 - multiprocessors 1-69
 - set with VPCLASS configuration parameter 1-114
- Processor, locking for multiple or single 1-69
- Profile
 - displaying count, onstat -p 15-4, 15-148
 - monitoring with SMI 2-21
 - setting counts to zero 15-5, 15-167
- Profile, partition 1-105
- PTADDESC logical-log record 5-11
- PTALTER logical-log record 5-11
- PTALTNEWKEYD log record 5-11
- PTALTOLDKEYD log record 5-11
- PTCOLUMN log record 5-11
- PTEXTEND logical-log record 5-11
- PTRENAME log record 5-11
- PTRUNC record subtype 5-15
- PTRUNCATE log record 5-10

Q

- QSTATS configuration parameter 1-80
- Quantum, of memory 15-74, 15-75, 15-76
- Quiescent mode 9-3, 11-12, 11-13

R

- RA_PAGES configuration parameter 1-80
- RA_THRESHOLD configuration parameter 1-81
- Raw disk space
 - UNIX 14-3
 - Windows 14-3, 14-6, 14-8
- RDELETE logical-log record 5-11
- Read-ahead, data pages
 - number 1-80
 - threshold 1-81
- Recovery threads
 - offline 1-72
 - online 1-73
- REFCOUNT record subtype 5-15
- Remainder page, defined 4-15
- Removing
 - CPU virtual processors 11-17
 - stray smart large objects 14-17
 - virtual processors 11-16
- RENDBS logical-log record 5-11
- Reserved area, sbspace 4-24
- Reserved pages
 - checking with oncheck 7-5, 7-11
 - defined 4-2
 - location in root dbspace 4-1
 - viewing contents 4-2
- RESIDENT
 - changing dynamically 11-20
- RESIDENT configuration parameter
 - defined 1-82
 - onmode -r or -n 11-15
- Resident shared memory
 - RESIDENT configuration parameter 1-82
 - turning on and off residency 11-15
- RESTARTABLE_RESTORE configuration parameter 1-82
- Restarting the database server 1-9
- Results table 3-5
- Return codes on exit 15-167
- Reuse of freed index pages 4-20
- revcdr.out file E-61
- revcdr.sh script A-3, A-9, E-61
- Reversion messages
 - database server E-49, E-61
 - Enterprise Replication E-61, E-65
- REVERT logical-log record 5-11
- revtestcdr.out file E-61
- RINSERT logical-log record 5-12
- ROLLBACK logical-log record 5-12
- Rolling back long transactions 1-62
- ROLWORK logical-log record 5-12
- Root dbspace
 - initial chunk 1-84
 - mirroring 1-67
 - specifying ROOTNAME configuration parameter 1-83
 - structure 4-1
 - using a link 1-84
- ROOTNAME configuration parameter
 - defined 1-83
 - use by PHYSDBS D-9
- ROOTOFFSET configuration parameter 1-84
- ROOTPATH configuration parameter
 - defined 1-84
 - specifying as a link 1-67, 1-84
- ROOTSIZE configuration parameter 1-85
- ROWID
 - defined 4-13
 - fragmented table 4-13
 - functions as forward pointer 4-13
 - locking information 15-142
 - stored in index pages 4-13

- Rows
 - data, storage 4-14
 - displaying contents with oncheck 7-13
 - storage location 4-14
- RS secondary server
 - event alarms and messages C-5
- RSVEXTEN logical-log record 5-12
- RTO_SERVER_RESTART
 - changing dynamically 11-20
- RTO_SERVER_RESTART configuration parameter 1-85
- RTREE logical-log record 5-12
- RUPAFT logical-log record 5-12
- RUPBEF logical-log record 5-12
- RUPDATE logical-log record 5-12

S

- SBLOB logical-log record 5-13, 5-14
- Sbpage structure 4-25
- SBSPACENAME configuration parameter 1-86, 1-102
- sbspaces
 - g option 14-16
 - adding a chunk 14-4
 - changing defaults 14-15, 14-17
 - cleaning up references 14-17
 - creating with onspaces 14-11
 - default name 1-86
 - dropping a chunk 14-18
 - dropping a sbspace 14-20
 - ending mirroring 14-23
 - maximum number 14-6, 14-7, 14-11, 14-16
 - metadata area
 - size and offset 14-11, 14-13
 - structure 4-24
 - multiple chunk 4-26
 - naming conventions 14-11
 - onstat -d usage 15-20, 15-22
 - reserved area 4-24
 - sbpage structure 4-25
 - starting mirroring 14-21
 - structure 4-23
 - temporary 1-87, 14-12, 15-20
 - creating 14-12
 - user-defined data statistics 1-102
- SBSPACETEMP configuration parameter 1-87, 14-12
- Scheduler group names
 - PH_GROUP table 3-3
- Scheduler messages
 - PH_ALERT table 3-3
- Scheduler task historical data
 - Results table 3-5
- Scheduler task information
 - PH_RUN table 3-2
 - PH_TASK table 3-1
- Scheduler task thresholds
 - PH_THRESHOLD table 3-4
- Schema Editor 6-3
- Screen reader
 - reading syntax diagrams G-1
- Scripts
 - concdr.sh A-4
 - ex_alarm.sh 1-13, C-1
 - log_full 1-12, C-1
 - no_log 1-12
 - revcdr.sh A-9
- SD secondary server
 - event alarms and messages C-6

- SDS_ENABLE configuration parameter 1-88
- SDS_PAGING configuration parameter 1-88
- SDS_TEMPDBS configuration parameter 1-89
- SDS_TIMEOUT configuration parameter 1-89
- security
 - local 1-90
- SECURITY_LOCALCONNECTION configuration parameter 1-90
- Server Studio 6-3
- SERVERNUM configuration parameter 1-90
- Session information
 - global transactions 15-165
 - setting environment variables 1-10
 - SMI tables 2-25, 2-29
- Sessions
 - limiting them 1-55
- SET EXPLAIN statement
 - setting dynamically 11-22
- SET STATEMENT CACHE statement 1-98, 11-11
- Setting data replication type 11-6
- Shared library files A-5
- Shared memory
 - adding segment with onmode 11-3
 - base address 1-91, 1-92
 - buffer, frequency of flushing 1-19
 - buffer, maximum number 1-19
 - changing
 - decision-support parameters 11-10
 - residency with onmode 11-15
 - dumps 1-41, 1-42
 - examining with SMI 2-2
 - initializing 9-1
 - monitoring 15-1
 - physical-log buffer 1-78
 - resident portion, flag 1-82
 - saving copy of with onstat 15-4
 - segments, dynamically added, size 1-91
 - SERVERNUM configuration parameter 1-90
 - size displayed by onstat 15-6
 - virtual segment, initial size 1-94
- SHMADD configuration parameter
 - 64-bit addressing 1-90
 - defined 1-90
- SHMBASE configuration parameter 1-91
- shmem file
 - DUMPSHMEM configuration parameter 1-42
 - shmem.xxx A-9
- SHMNOACCESS configuration parameter 1-92
- SHMTOTAL configuration parameter 1-92
- SHMVIRT_ALLOCSEG configuration parameter 1-93
- SHMVIRT_SIZE configuration parameter 1-94
- shortcut keys
 - keyboard G-1
- Shutting down the database server 11-12, 11-13
- SINGLE_CPU_VP configuration parameter 1-95
- Size
 - chunk 14-4, 14-5
 - index fragments 4-7
 - metadata 14-11
 - offset 14-6, 14-9, 14-12, 14-16
- sm_versions.std file A-9
- Smart large objects
 - buffer pool 1-19
 - cleaning up references 14-17
 - default name 1-86
 - logging 14-14
 - logical-log records 5-14

- Smart large objects (*continued*)
 - user-defined data statistics 1-102
- snmpd.log file A-9
- Specify
 - modified pages, percentage
 - LRU queue 13-5
- Specifying pathname 14-3
- SQL Editor 6-3
- SQL profile
 - information 2-33
- SQL statement cache
 - enabling the cache 1-98, 11-11
 - flushing the cache 11-11
 - inserting
 - key-only entries 1-98, 11-19
 - qualified statements 1-98, 11-19
 - limiting the cache size 1-99
 - memory pools 1-99
 - printing contents 15-25, 15-29
 - specifying number of hits 1-98, 11-19
 - turning off the cache 11-11
 - turning on the cache 1-98, 11-11
- SQL statements
 - SET STATEMENT CACHE 1-98, 11-11
 - UPDATE STATISTICS 1-102
- SQLCA, warning flag when fragment skipped during query 1-23
- sqlhosts file or registry
 - defined A-9
 - multiple dbservernames 1-25
- sqlmux, multiplexed connections in NETTYPE configuration parameter 1-72
- SQLTRACE configuration parameter 1-95
- SSL_KEYSTORE_LABEL configuration parameter 1-96
- STACKSIZE configuration parameter 1-97
- STAGEBLOB configuration parameter 1-97
- Starting database server with oninit 9-1
- Starting the database server online 11-12, 11-13
- STMT_CACHE configuration parameter 1-97
- STMT_CACHE environment variable 11-11
- STMT_CACHE environment variables 1-98
- STMT_CACHE_HITS configuration parameter 1-98, 11-19
- STMT_CACHE_NOLIMIT configuration parameter 1-99
- STMT_CACHE_NUMPOOL configuration parameter 1-99
- STMT_CACHE_SIZE configuration parameter 1-100
- Storage manager
 - xbsa.messages log A-10
- STORAGE_FULL_ALARM configuration parameter 1-100
- Symbolic links
 - using with shared libraries A-5
 - using with TAPEDEV configuration parameter 1-104
- SYNC logical-log record 5-13
- Syntax diagrams
 - reading in a screen reader G-1
- sysadmin database
 - Results table 3-5
 - tables
 - command_history 3-5
 - PH_ALERT 3-3
 - PH_GROUP 3-3
 - PH_RUN 3-2
 - PH_TASK 3-1
 - PH_THRESHOLD 3-4
- sysadinfo table 2-7
- SYSALARMPROGRAM configuration parameter 1-101
- sysaudit table 2-7
- syscheckpoint table 2-8
- syschkio table 2-8
- syschunks table 2-9
- syscsm table 2-10
- syscsmsla table 2-10
- syscsmstab table 2-11
- sysconfig table 2-11
- sysdatabases table 2-11
- sysdbslocale table 2-12
- sysdbspaces table 2-12
- sysdri table 2-13
- sysdual table 2-14
- sysenv table 2-14
- sysenvses table 2-14
- sysessions table 2-30
- sysextents table 2-14
- sysextspaces table 2-14
- syssha_latency table 2-15
- syssha_type table 2-16
- syssha_workload table 2-16
- sysipl table 2-17
- syslocks table 2-17
- syslogs table 2-18
- sysmaster database
 - defined 2-1
 - failure to build A-4
 - functionality of 2-1
 - initialization 9-1
 - list of topics covered by 2-2
 - SMI tables 2-2
 - space required to build 2-1
 - sysextspaces 2-14
 - types of tables 2-1
 - warning 2-1
 - when created 2-1
- sysmgminfo table 2-18
- sysnetclienttype table 2-19
- sysnetglobal table 2-19
- sysnetworkio table 2-20
- sysonlinelog table 2-21
- sysprofile table 2-21
- sysproxyagents table 2-22
- sysproxydistributors table 2-23
- sysproxysessions table 2-23
- sysproxytxnops table 2-24
- sysproxytxns table 2-24
- sysptprof table 2-25
- sysrepevtreg table 2-25
- sysrepstats table 2-26
- sysrsslog table 2-28
- SYSSBSPACENAME configuration parameter 1-102
- sysscblst table 2-29
- sysessesappinfo table 2-29
- sysessesprof table 2-29
- sysssmx table 2-31
- sysssmxses table 2-32
- sysssqltrace table 2-32
- sysssqltrace_info table 2-33
- sysssqltrace_iter table 2-33
- sysssrcrs table 2-34
- sysssrcsds table 2-34
- sysstabnames table 2-35
- System catalog tables
 - disk space allocation 4-27
 - listing 7-6
 - oncheck -cc 7-8
 - oncheck -pc 7-8
 - sysdistrib 1-102

- temporary sbspaces
 - creating with onspaces 14-11
- Temporary smart large object
 - default sbspaces 1-87
 - SBSPACETEMP configuration parameter 1-87
- Temporary tables
 - DBSPACETEMP configuration parameter 1-26
 - extent size doubling 4-11
 - oninit utility 9-3
 - rules for use 1-27
- TEMPTAB_NOLOG configuration parameter 1-106
- TEXT and BYTE data
 - blob descriptor 4-13, 4-22
 - modifying storage 4-22
 - page descriptor 4-22
 - size limitations 4-22
 - storage on disk 4-21, 4-22
 - updating 4-22
 - when modified 4-22
 - when written 4-22
- Threads
 - onstat -X usage 15-5, 15-165
- Tightly-coupled mode 15-165
- Time stamp
 - blobspace blobpage 4-23
 - defined 4-26
- Time-out condition 15-24
- Trace class B-1
- Trace message B-1
- Tracepoints B-1
- Transaction manager
 - loosely-coupled mode 15-165
 - tightly-coupled mode 15-165
- Transaction Replicate Group E-62
- Transactions
 - heterogeneous commit 1-53
 - kill with onmode -Z 11-23
 - pending 15-163
 - XID 15-165
- Trapping errors with onmode B-1
- TRUNCATE 5-10
- TRUNCATE log record 5-10
- Truncate table messages E-68
- Tuning
 - large number of users 1-70
 - use of NETTYPE configuration parameter 1-70
- Turning on SQL statement cache 11-11
- Two-phase commit protocol, killing distributed transactions 11-23
- TXTIMEOUT configuration parameter
 - defined 1-107
 - onmode utility 11-23
- Unblocking database server 11-4
- Unbuffered disk space
 - UNIX 14-3
 - Windows 14-3, 14-6, 14-8
- UNDO logical-log record 5-14
- UNDOBLDC log record 5-14
- UNIQID logical-log record 5-14
- UNIX
 - buffered disk space 14-3
 - interrupt signal 6-1
 - unbuffered disk space 14-3
 - using onspaces 14-1
- UNSECURE_ONSTAT configuration parameter 1-107
- UPDAFT logical-log record 5-14
- UPDATABLE_SECONDARY configuration parameter 1-81
- UPDATE STATISTICS statement 1-36, 1-38, 1-102
- Updating blobspace statistics 15-22
- UPDBEF logical-log record 5-14
- USELASTCOMMITTED
 - changing dynamically 11-20
- USELASTCOMMITTED configuration parameter 1-107
- USEOSTIME parameter 1-108
- User connections, monitoring E-25
- User session
 - monitoring with SMI 2-30
 - status codes 15-161
- User-defined aggregates, printing definitions 15-25
- User-defined routines, debugging B-1
- User-defined type
 - data distributions 1-102
- Utilities 6-1, 15-25
 - V option 6-1
 - version option 6-1
 - changing parameter values 1-10
 - gcore 1-40, 1-42
 - IBM Informix Server Administrator 6-3
 - ON-Monitor utility 1-10, 1-21, 12-1, 12-4
 - oncheck 7-1, 7-19
 - oncmsm 16-1
 - ondblog utility 8-1
 - oninit 9-1
 - onlog 10-1, 10-4
 - onmode 11-1, 11-22
 - onmode and PDQ 15-76
 - onparams 13-1, 13-6
 - onpassword 17-1
 - onspaces 14-1, 14-21
 - onstat utility 15-1, 15-167
 - g env option 15-53, 15-54
 - g mgm option 15-74
 - g seg option 1-90
 - option 6-1
 - quick reference 6-1

U

- UDINSERT
 - logical-log record 5-13
 - record subtype 5-15
- UDUPAFT
 - logical-log record 5-13
 - record subtype 5-15
- UDUPBEF
 - logical-log record 5-13
 - record subtype 5-15
- UDWRITE
 - logical-log record 5-14
 - record subtype 5-15

V

- VARCHAR data type
 - 4-bit bit map requirement 4-8, 4-9
 - implications for data row storage 4-14
 - indexing considerations 4-20
 - storage considerations 4-12
- Verifying backups 1-10
- Violations table
 - messages E-4, E-44
- Virtual processors
 - adding or removing with onmode 11-16
 - limits 11-16

- Virtual processors (*continued*)
 - number in
 - AIO class D-8
 - CPU class D-9
 - priority aging D-7
 - processor affinity 1-69
- Visual disabilities
 - reading syntax diagrams G-1
- VP_MEMORY_CACHE_KB
 - changing dynamically 11-20
- VP_MEMORY_CACHE_KB configuration parameter
 - defined 1-109
- VP.servername.nnx file A-10
- VPCLASS configuration parameter 1-55
 - AFF_NPROCS D-2
 - AFF_SPROC D-3
 - default values 1-110
 - defined 1-110
 - in ONCONFIG file 1-110
 - NOAGE D-7
 - NUMAIOVPS D-8
 - onmode utility 11-17
 - reserved names 1-111
 - setting maximum VPs 1-114
 - setting number of VPs 1-113
 - setting processor affinity 1-114
 - user-defined classes 1-112

W

- Warnings
 - buildsmi script 2-2
 - when fragment skipped during query processing 1-23
- White space in ONCONFIG file 1-1
- Windows
 - adding or removing virtual processors 11-18
 - buffered disk space 14-3
 - unbuffered disk space 14-3, 14-6, 14-8
 - using onspaces 14-1
- WSTATS configuration parameter 1-114

X

- XAPREPARE logical-log record 5-14
- xbsa.messages log A-10



Printed in USA

SC23-7749-02



Spine information:

IBM Informix **Version 11.50**

IBM Informix Dynamic Server Administrator's Reference

