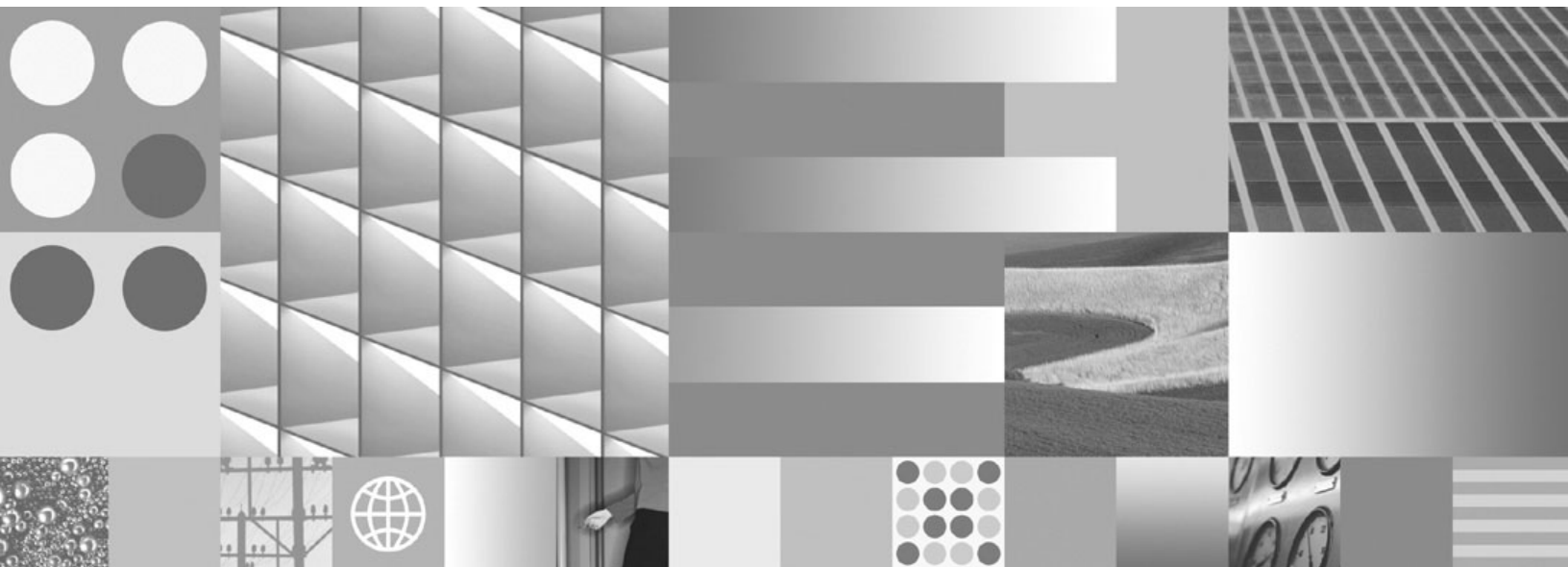


IBM Informix

Version 11.50



**IBM Informix Dynamic Server
Getting Started Guide**

IBM Informix

Version 11.50



IBM Informix Dynamic Server Getting Started Guide

Note:

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

This edition replaces GI11-8342-01.

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 publication 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	ix
In This Introduction	ix
About This Publication	ix
Types of Users	ix
Software Dependencies	x
Assumptions About Your Locale	x
Demonstration Database	x
What's New for Version 11.50	xi
Documentation Conventions	xi
Typographical Conventions	xi
Feature, Product, and Platform Markup	xi
Example Code Conventions	xii
Additional Documentation	xii
Compliance with Industry Standards	xiii
How to Provide Documentation Feedback	xiii

Chapter 1. Introducing Dynamic Server and Client Products 1-1

In This Chapter	1-1
IBM Informix Dynamic Server	1-1
IBM Informix Dynamic Server Edition Comparison	1-2
Installation and Migration	1-2
Products Bundled with the Database Server	1-2
BladeManager	1-2
DataBlade API	1-2
IBM Data Studio	1-3
IBM Informix Client SDK Products	1-3
IBM Informix DataBlade Developers Kit (DBDK)	1-5
IBM Informix JDBC Driver	1-5
IBM Informix Spatial DataBlade Module	1-5
IBM Informix Web DataBlade Module	1-5
International Language Supplement	1-6
Rational Application Developer for WebSphere Software	1-6
Server Studio	1-6
Related IBM Informix Products	1-6
IBM Informix Server Administrator (ISA)	1-6
OpenAdmin Tool for IDS	1-6
IBM Informix MaxConnect (UNIX)	1-7
IBM Office Connect	1-7
IBM Informix Data Director for Web	1-7
DataBlade Modules	1-7
Client Products for Dynamic Server	1-8
JDBC Drivers	1-8
.NET Providers	1-8
PHP Drivers	1-8
Ruby on Rails Adapters	1-9
Other Informix Drivers	1-9
Other Related IBM Products	1-9

Chapter 2. Using New Features in Dynamic Server 2-1

In This Chapter	2-5
+ New Features in Version 11.50.xC3 of IBM Informix Dynamic Server	2-6
+ Performing Unattended Installations on Mac OS X	2-6
+ Setting up Data Source Names on Mac OS X	2-6
+ Using SQL Admin API to Dynamically Update Configuration Parameters	2-6
+ Dynamically Updating the LTXEHW, LTXHW, and DYNAMIC_LOGS Configuration Parameters	2-7

+ Dynamically Updating Enterprise Replication Configuration Parameters in the ONCONFIG File.	2-7
+ Improved Consistency Reporting after Enterprise Replication Repair Operations	2-7
+ Administering and Monitoring Enterprise Replication with SQL Admin API	2-7
+ Improved SQL Tracing with the SQL Admin API	2-8
+ Changing the Size of the First Extent of a Table	2-8
+ Rolling Back SQL Transactions to a Savepoint.	2-8
+ Capturing Transactional Data with the Change Data Capture API	2-9
+ Basic Text Search DataBlade Module Supports High-Availability Clusters	2-9
+ Querying XML Attributes with the Basic Text DataBlade Module	2-9
+ Setting the Frequency of Error Checking for Smart Large Object Transmission	2-9
New Features in Version 11.50.xC2 of IBM Informix Dynamic Server	2-10
Reconfiguring Connection Manager while It's Running	2-10
Multiple Copies of Dynamic Server on the Same Windows Computer	2-10
Enhanced Dynamic Server Installation Application on Mac OS X	2-11
Controlling Memory Use during Enterprise Replication Synchronization	2-11
Obtaining IDS Version Information From the cdr Utility	2-11
Monitoring Enterprise Replication with New SMI Tables.	2-11
Monitoring the Whole Enterprise Replication Domain	2-12
Preventing ATS or RIS File Generation.	2-12
The ISM Administrator Program is Included with Storage Manager on Windows	2-12
Limiting the Number of Sessions That Can Connect to Dynamic Server.	2-12
New Format for Backup Filters	2-13
Enhancements to the OpenAdmin Tool for IDS	2-13
Controlling I/O of B-Tree Indexes with Compression Levels	2-13
Subquery Support in UPDATE and DELETE Statements	2-13
Longer Return Strings from String Manipulation Functions	2-14
Server-Specific Audit Configuration File Functionality	2-14
New Features in Version 11.50.xC1 of IBM Informix Dynamic Server	2-14
SQL Admin API Commands to Configure High-Availability Clusters Added	2-15
Enhanced Connection Management for High-Availability Clusters	2-15
New Options to Troubleshoot High-Availability Clusters.	2-16
Update Data on Secondary Servers	2-16
Support for Transient Types on High-Availability Cluster Secondary Servers	2-16
Enhanced Configuration Options During Installation	2-16
Install as the Local System Account support added (Windows)	2-17
Enhanced Data Server Client Session Information	2-17
Improved onconfig.std and New Default Values for Configuration Parameters	2-17
Enhancements to the OpenAdmin Tool for IDS	2-22
Enhanced Shared-Memory Dump File Size Control	2-22
Enhanced Startup Script Customization	2-23
New Options for Configuring Storage Space Monitoring.	2-23
Updating Table Statistics with AUS Scheduler Tasks	2-23
Control External Directives for a Session	2-23
Improved Query Performance for Large Integers and Serial Data	2-24
Support for Obtaining Explain Output in XML Format Added.	2-24
DRDA Protocol Configuration During Installation Added	2-24
SQL Expressions with the IS [NOT] NULL Predicate	2-24
Determine Data Currency with a Version Column	2-24
Distributed Relational Database Architecture (DRDA) Enhancements	2-25
J/Foundation Upgraded to JRE 5.0	2-25
Improvements to the Basic Text Search DataBlade Module	2-25
New Support for Dynamic SQL Statements in SPL Routines	2-26
New Support for Extensible Stylesheet Language Transformation (XSLT)	2-26
Single Sign-on Support Added	2-26
Support for Encrypting Data by Using Secure Sockets Layer (SSL) Communications Added	2-26
New Features in Version 11.10.xC2 of IBM Informix Dynamic Server	2-27
Prevent Shared Memory Addresses from being used by IDS	2-27
Preview Enterprise Replication ATS or RIS Repair Operations	2-27
New Features in Version 11.10 of IBM Informix Dynamic Server	2-27
Multiple Remote Standalone Secondary Servers.	2-28
Multiple Shared Disk Secondary Servers	2-29

Backup and Restore to Directories with ontape	2-29
Continuous Logical Log Restore	2-29
Encrypted Communications for HDR	2-29
Improved Parallelism during Backup and Restore	2-30
Automatic Ordering of dbspaces during Backup and Restore	2-30
RTO Policy to Manage Server Restart	2-30
Non-blocking Checkpoints.	2-30
Performance Improvements for Enterprise Replication	2-30
ON-Bar Performance Report	2-31
Transform Data during Backup and Restore	2-31
Improved Performance for Cooked Files with Direct I/O on UNIX	2-31
Improved Performance of Online Index Creation	2-31
SQL Administration API	2-32
Schedule Administrative Tasks	2-32
Monitor and Analyze Recent SQL Statements	2-32
Dynamically Change Enterprise Replication Configuration Parameters and Environment Variables	2-33
Dynamically Rename Enterprise Replication Columns, Tables, and Databases.	2-33
Truncate Replicated Tables.	2-33
Improved Statistics Maintenance.	2-33
Installation Improvements on Windows Platforms	2-33
Session Configuration Routines	2-34
Multiple Users for Administration Mode	2-34
PHP-based OpenAdmin Tool for IDS	2-34
Named Parameters in a JDBC CallableStatement	2-34
Index Binary Data Types	2-35
Trigger Enhancements	2-35
Derived tables in the FROM Clause of Queries	2-36
Index Self-Join Query Plans	2-36
Optimizer Directives in ANSI-Compliant Joined Queries.	2-36
Deployment Wizard	2-37
Enhanced Concurrency with Committed Read Isolation	2-37
Enhanced Data Types and UDR Support in Cross-Server Distributed Operations	2-37
XML Publishing	2-38
Index Hierarchical Data.	2-38
Basic Text Search	2-38
Improved Concurrency with Private Memory Caches for Virtual Processors	2-39
Web Feature Service for Geospatial Data	2-39
Support for Data Server Clients with DRDA.	2-39
Statement Labels, GOTO, and LOOP Statements in the SPL language	2-39
New SQL Functions	2-40
Automatic Re-Compilation of Prepared Statements	2-40
Label-Based Access Control	2-41
New Features in Version 10.00 of IBM Informix Dynamic Server	2-42
New Features in Version 10.00.xC4	2-42
New Features in Version 10.00.xC3	2-43
New Features in Version 10.00.xC1	2-45
New Features in Version 9.4	2-56
Security Enhancement	2-56
Database Server Usability Enhancements	2-56
Performance Enhancements	2-59
Enterprise Replication Enhancements	2-59
Extensibility Enhancements	2-61
SQL Enhancements	2-62
GLS Enhancements	2-66
Reliability, Availability, and Supportability Features	2-66
DataBlade API Enhancements.	2-67
High-Performance Loader Enhancements	2-68
Backup and Restore Enhancements	2-68
Installation Enhancements	2-69
New Features in Version 9.3	2-69
UNIX Bundle Installer	2-70

Database Server Usability Enhancements	2-70
DataBlade API Enhancements.	2-71
Enterprise Replication Enhancements	2-72
Extensibility Enhancements	2-75
J/Foundation Enhancements	2-76
Performance Enhancements	2-77
SQL Enhancements	2-78
New Features in Dynamic Server, Version 9.21	2-78
ANSI Join Syntax	2-78
Rename Index Statement	2-79
Nonlogging (Raw) Tables	2-79
onpladm Utility	2-79
The onbar -b -l Command	2-79
9.x DB-Access to 7.x Synonyms	2-79
SQL Statement Cache Improvements	2-79
DataBlade API Features.	2-80
Java Features in 9.21.	2-81
MaxConnect Support	2-82

Chapter 3. Using Existing Dynamic Server Features 3-1

In This Chapter.	3-2
Dynamic Scalable Architecture.	3-2
The Shared-Memory Component	3-2
The Disk Component	3-3
The Virtual Processor Component.	3-4
Client/Server Connections	3-4
High Performance	3-5
Memory Management	3-5
Parallelization	3-6
Query Optimizer	3-6
Fault Tolerance and High Availability	3-6
Backup and Restore	3-6
Fast Recovery	3-8
Mirroring	3-8
Data Replication	3-8
Database Server Security.	3-9
Informix RDBMS Features	3-10
Structured Query Language (SQL)	3-10
Stored Procedure Language (SPL)	3-10
System Catalog Tables	3-11
Data Types	3-11
Application Types	3-13
OLTP Applications	3-13
DSS Applications	3-13
Database Support.	3-14
Relational Databases.	3-14
ANSI-Compliant Databases	3-14
Object-Relational Databases	3-14
Dimensional Databases	3-18
Distributed Queries and Multiphase Transactions	3-19
Access Methods	3-19
Primary Access Methods	3-19
Secondary Access Methods	3-20
User-Defined Primary Access Methods	3-20
User-Defined Secondary Access Methods	3-20

Chapter 4. Installing, Administering, and Tuning the Database Server 4-1

In This Chapter.	4-1
Database Server Users	4-1
Planning, Installing, and Configuring the Database Server	4-1

Administering the Database Server	4-4
Monitoring Performance	4-7
Troubleshooting the Database Server.	4-8
Chapter 5. Designing, Maintaining, and Extending the Database.	5-1
In This Chapter.	5-1
Designing, Developing, and Extending the Database	5-1
Developing Application Programs that Access the Database	5-3
Chapter 6. Using the Documentation	6-1
In This Chapter.	6-1
The IBM Informix Documentation Set	6-1
IBM Informix Dynamic Server Publications	6-1
Client SDK and Connectivity Publications	6-3
DataBlade Publications	6-4
Appendix A. Database Server Utilities	A-1
Appendix B. Accessibility	B-1
Accessibility features for IBM Informix Dynamic Server	B-1
Accessibility Features.	B-1
Keyboard Navigation.	B-1
Related Accessibility Information.	B-1
IBM and Accessibility.	B-1
Notices	C-1
Trademarks	C-3
Index	X-1

Introduction

In This Introduction.	ix
About This Publication.	ix
Types of Users	ix
Software Dependencies	x
Assumptions About Your Locale.	x
Demonstration Database	x
What's New for Version 11.50	xi
Documentation Conventions	xi
Typographical Conventions	xi
Feature, Product, and Platform Markup	xi
Example Code Conventions	xii
Additional Documentation	xii
Compliance with Industry Standards	xiii
How to Provide Documentation Feedback	xiii

In This Introduction

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

About This Publication

Use this publication to get started with IBM® Informix® Dynamic Server, Version 11.50. This publication describes the products bundled with Dynamic Server, an overview of major features in Dynamic Server, and the documentation for Dynamic Server. It also summarizes the basic tasks for using the database server and provides a quick reference to command-line utilities.

This section discusses the organization of the publication and the intended audience.

Types of Users

This publication is written for all Dynamic Server users:

- Database server administrators
- Database administrators
- Performance engineers
- Database users
- Programmers in the following categories
 - Application developers
 - DataBlade® module developers
 - Authors of user-defined routines
- Technical support

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 computer programming
- 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 Chapter 6, “Using the Documentation,” on page 6-1, for a list of supplementary titles.

Software Dependencies

This publication is written with the assumption that you are using Dynamic Server, Version 11.50, as your database server. Check the release notes for specific version compatibility.

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 IBM 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 for Version 11.50

For a comprehensive list of new features in IBM Informix Dynamic Server (IDS), Version 11.50, see Chapter 2, “Using New Features in Dynamic Server,” on page 2-1.

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
- 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 >> and << 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
Identifies information that is specific to IBM Informix Dynamic Server
End of Dynamic Server

Windows Only
Identifies information that is specific to the Windows operating system
End of Windows Only

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.

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.

Chapter 1. Introducing Dynamic Server and Client Products

In This Chapter	1-1
IBM Informix Dynamic Server	1-1
IBM Informix Dynamic Server Edition Comparison	1-2
Installation and Migration	1-2
Products Bundled with the Database Server	1-2
BladeManager	1-2
DataBlade API	1-2
IBM Data Studio	1-3
IBM Informix Client SDK Products	1-3
IBM Informix ESQL/C	1-3
IBM Informix ESQL/J Pre-Processor	1-3
IBM Informix GLS	1-4
IBM Informix Object Interface for C++	1-4
IBM Informix ODBC Driver	1-4
IBM Informix OLE DB Provider (Windows)	1-4
IBM Informix .NET Provider (Windows)	1-4
TP/XA	1-5
IBM Informix DataBlade Developers Kit (DBDK)	1-5
IBM Informix JDBC Driver	1-5
IBM Informix Spatial DataBlade Module	1-5
IBM Informix Web DataBlade Module	1-5
International Language Supplement	1-6
Rational Application Developer for WebSphere Software	1-6
Server Studio	1-6
Related IBM Informix Products	1-6
IBM Informix Server Administrator (ISA)	1-6
OpenAdmin Tool for IDS	1-6
IBM Informix MaxConnect (UNIX)	1-7
IBM Office Connect	1-7
IBM Informix Data Director for Web	1-7
DataBlade Modules	1-7
Client Products for Dynamic Server	1-8
JDBC Drivers	1-8
.NET Providers	1-8
PHP Drivers	1-8
Ruby on Rails Adapters	1-9
Other Informix Drivers	1-9
Other Related IBM Products	1-9

In This Chapter

This chapter provides an overview of IBM Informix Dynamic Server (IDS), Version 11.50, IBM Informix Client Software Development Kit, and related products. For a list of the publications and description of each product, see “The IBM Informix Documentation Set” on page 6-1.

IBM Informix Dynamic Server

A *database server* is a software package that manages access to one or more databases for one or more client applications. Dynamic Server is a fast and scalable database server that manages traditional relational, object-relational, and web-based databases. Dynamic Server supports alphanumeric and rich data, such as graphics, multimedia, geospatial, HTML, and user-defined types. You can use

Dynamic Server on UNIX, Linux, Mac OS X, or Windows with online transaction processing (OLTP), data marts, data warehouses, and e-business applications.

You can write *user-defined routines* (UDRs) in Java™, C, and stored procedure language (SPL). A UDR is a routine that an SQL statement, user-defined function, or user-defined procedure can invoke.

IBM Informix Dynamic Server Edition Comparison

Informix Dynamic Server is available in different editions to fit different business needs. For details about the differences between editions, see the following Web site: <http://www.ibm.com/software/data/informix/ids/ids-ed-choice/>

Installation and Migration

For information on how to install the database server products, see the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X*, the *IBM Informix Dynamic Server Installation Guide for Windows*, or the *Quick Beginnings for IBM Informix Dynamic Server Express Edition*.

If you migrate to Dynamic Server, Version 11.50, from an earlier version of the database server, start with the information provided in the *IBM Informix Migration Guide*.

Products Bundled with the Database Server

In addition, several products are included with the database server. This section discusses the IBM Informix products that help you manage the database server. The following products are bundled with Dynamic Server:

- BladeManager
- DataBlade API
- IBM Data Studio
- IBM Informix Client Software Development Kit
- IBM Informix Connect
- IBM Informix DataBlade Developers Kit (DBDK)
- IBM Informix JDBC Driver
- IBM Informix Spatial DataBlade Module
- IBM Informix Web DataBlade Module
- International Language Supplement
- Rational Application Developer for WebSphere® Software
- Server Studio

BladeManager

Use the BladeManager to register new DataBlade modules in Informix databases. BladeManager runs on client computers.

For more information, see the *IBM Informix DataBlade Module Installation and Registration Guide*.

DataBlade API

The DataBlade API is a C-language application programming interface that is provided with Dynamic Server. Experienced C programmers can use API functions

in DataBlade modules to develop client and database server applications that access data stored in a database. The DataBlade API contains public data structures, public functions, and header files for DataBlade modules, Informix ESQL/C, GLS, and so on.

For more information, see the *IBM Informix DataBlade API Programmer's Guide* and the *IBM Informix DataBlade API Function Reference*.

IBM Data Studio

IBM Data Studio is an integrated data-management environment. Data architects, developers, and database administrators can use IBM Data Studio to collaborate throughout the data-driven application development lifecycle. Learn how IBM Data Studio can help throughout the data lifecycle at: <http://www.ibm.com/software/data/studio/>

The following information center introduces IBM Data Studio and the underlying workbench technology, and explains how to use its integrated development environment to achieve your database development goals: <http://publib.boulder.ibm.com/infocenter/dstudio/v1r1m0/index.jsp>

IBM Informix Client SDK Products

The IBM Informix Client Software Development Kit (Client SDK) includes several application-programming interfaces (APIs) that developers can use to write applications for Informix database servers in ESQL, C, and Java. You can also write Informix ESQL/C applications for the DB2® database server. IBM Informix Connect contains the runtime libraries of the APIs in the Client SDK.

For more information, see your *IBM Informix Client Products Installation Guide*.

IBM Informix ESQL/C

Informix ESQL/C is an SQL application programming interface (API) that lets programmers embed SQL statements directly into a C program to interact with the database server, access databases, manipulate the data in a program, and check for errors.

IBM Informix ESQL/C consists of the following components:

- Informix ESQL/C libraries of C functions for accessing the database server
- Informix ESQL/C header files, which provide definitions for the data structures, constants, and macros
- **esql**, a command that manages the source-code processing to convert a C file that contains SQL statements into an object file
- ESQL client-interface *dynamic link libraries* (DLLs), which let an Informix ESQL/C application run on Windows

For more information, see the *IBM Informix ESQL/C Programmer's Manual*.

IBM Informix ESQL/J Pre-Processor

IBM Informix Embedded SQLJ enables you to embed SQL statements in your Java programs. It consists of the SQLJ translator, which translates SQLJ code into Java code, and a set of Java classes that provide runtime support for SQLJ programs. When you run an SQLJ program, it uses IBM Informix JDBC Driver to connect to an Informix database.

For more information, see the *IBM Informix Embedded SQLJ User's Guide* and "IBM Informix JDBC Driver" on page 1-5.

IBM Informix GLS

The Global Language Support (GLS) feature lets the database server handle different languages, cultural conventions, and code sets using different locales. GLS allows you to create databases that use the diacritics, collating sequence, and monetary and time conventions of the language that you select. A GLS locale is an environment that has defined conventions for a particular language or culture. See "Assumptions About Your Locale" on page x of the Introduction.

With GLS support, the database server does not need to specify how to process culture-specific information directly because this information resides in a GLS locale. When the database server needs culture-specific information, it makes a call to the GLS library. The GLS library, in turn, accesses the GLS locale and returns the information to the IBM Informix product.

IBM Informix GLS provides procedures, macros, and functions to:

- Process single-byte and multibyte characters and strings.
- Convert date, time, monetary, and number values from and to locale-specific data formats.

For more information, see the *IBM Informix GLS User's Guide*.

IBM Informix Object Interface for C++

Use the IBM Informix Object Interface for C++ to develop IBM Informix client applications using the C++ programming language.

For more information, see the *IBM Informix Object Interface for C++ Programmer's Guide*.

IBM Informix ODBC Driver

IBM Informix ODBC Driver is the Informix implementation of the Microsoft® Open Database Connectivity (ODBC) standard. It supports SQL statements with a library of C functions that an application calls to access Informix databases.

For more information, see the *IBM Informix ODBC Driver Programmer's Manual*.

IBM Informix OLE DB Provider (Windows)

IBM Informix OLE DB Provider enables OLE DB applications, such as Active Data Objects (ADO) applications and web pages, to access the database server.

For more information, see the *IBM Informix OLE DB Provider Programmer's Guide*.

IBM Informix .NET Provider (Windows)

The IBM Informix .NET Provider is a .NET assembly that lets .NET applications access and manipulate data in IBM Informix databases. It does this by implementing several interfaces in the Microsoft .NET Framework that are used to access data from a database.

For more information, see the *IBM Informix .NET Provider Reference Guide*.

TP/XA

The TP/XA library facilitates communication between a third-party transaction manager and your database server. TP/XA is supplied with IBM Informix ESQL/C. Use TP/XA for distributed transaction processing in a multivendor database setting.

For more information, see the *IBM Informix TP/XA Programmer's Manual*.

IBM Informix DataBlade Developers Kit (DBDK)

The Informix DataBlade Developers Kit includes the following tools for developing and packaging DataBlade modules:

- BladeSmith (organizes a DataBlade development project)
- DBDK Visual C++ Add-In and Ifx Query (debugs DataBlade modules)
- BladePack (creates a DataBlade package)
- BladeManager (registers and unregisters DataBlade modules)

For more information, see the *DataBlade Module Development Overview* and *IBM Informix DataBlade Developers Kit User's Guide*.

IBM Informix JDBC Driver

IBM Informix JDBC Driver lets Java programmers access Informix databases from within Java applications or applets. Programmers can create client applications that use JDBC to connect to Dynamic Server, query and retrieve data from a database or column, handle errors, and write UDRs. The IBM Informix JDBC Driver is compatible with the JavaSoft JDBC specifications. It maps standard Java data types and Informix database server data types.

For more information, see the *IBM Informix JDBC Driver Programmer's Guide*.

IBM Informix Spatial DataBlade Module

The IBM Informix Spatial DataBlade module embeds a geographic information system (GIS) into your IBM Informix Dynamic Server kernel. The IBM Informix Spatial DataBlade module implements the Open GIS Consortium, Inc. (OpenGIS®, or OGC) SQL3 specification of abstract data types (ADTs). These data types can store spatial data such as the location of a landmark, a street, or a parcel of land. The IBM Informix Spatial DataBlade module also conforms to the OpenGIS Simple Features Specification for SQL Revision 1.1.

The Spatial DataBlade module comes with a sample data disk that contains worldwide location-based data that can be visualized and manipulated using the free IBM Informix Spatial DataBlade bundle. This bundle can be ordered or downloaded free from <http://www.ibm.com/software/data/informix/blades/spatial/>.

IBM Informix Web DataBlade Module

Use the Web DataBlade module to create Web applications that incorporate data retrieved dynamically from the IDS database.

Using the Web DataBlade module, you need not develop a CGI application to dynamically access database data. Instead, you create HTML pages that include Web DataBlade module tags and functions that dynamically execute the SQL statements you specify and format the results. These pages are called Application

Pages (AppPages). The types of data you retrieve can include traditional data types, as well as HTML, image, audio, and video data.

For more information, see the *Informix Web DataBlade Module User's Guide* and the *Informix Web DataBlade Module Application Developer's Guide*.

International Language Supplement

IBM Informix products include a core set of GLS locale files, including the default locale and most locales to support English, Western European, Eastern European, Asian, and African territories. If you do not find a locale to support your language and territory, you can get additional locales in the International Language Supplement (ILS) product. The ILS provides all available GLS locales and code-set conversion files. It also includes error messages to support several languages.

Rational Application Developer for WebSphere Software

Rational® Application Developer is part of the Rational Software Development Platform series of products. The products are all built on Eclipse, which is an open-source platform for creating application development tools. IBM Rational Application Developer for WebSphere Software extends Eclipse with visual construction development. It helps Java developers rapidly design, develop, assemble, test, profile and deploy Java/J2EE, Portal, Web, Web services and SOA applications. You can learn more about this product on the Web at <http://www.ibm.com/software/awdtools/developer/application/index.html>.

Server Studio

Server Studio by AGS (formerly known as Server Studio JE) is a stand-alone, Java-based Integrated Development Environment (IDE) for the 7.3x, 9.x, 10.x, and 11.x database servers. Server Studio contains the following modules:

- Database Object Explorer
- SQL Editor
- Schema Editor

Additional modules are provided with the distribution of Server Studio on a try and buy basis. Contact Advanced Global Systems Ltd. (AGS) to obtain a license for the additional modules at www.agsltd.com.

Related IBM Informix Products

This section discusses the related products that you can use with Dynamic Server. For information on ordering these products, contact your IBM sales representative.

IBM Informix Server Administrator (ISA)

IBM Informix Server Administrator (ISA) is not included with Dynamic Server. ISA is available for download from <http://www.ibm.com/software/data/informix/downloads.html>.

OpenAdmin Tool for IDS

A PHP-based Web browser administration tool, the OpenAdmin Tool for IDS, provides the ability to administer multiple database server instances from single location. Some tasks you can perform with OpenAdmin include: gathering and analyzing performance statistics, monitoring system health, defining and managing automated tasks through the SQL Administration API, creating and displaying performance histograms for analysis and tuning, and monitoring high availability

solutions that include HDR, shared disk secondary servers, and remote standalone secondary servers. You can easily plug in your own extensions to OpenAdmin to create the functionality you need.

OpenAdmin is an open-source program that you can download from https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd.

IBM Informix MaxConnect (UNIX)

IBM Informix MaxConnect is a networking product for Informix database servers on UNIX. Two protocols for multiplexing connections, **ontliimc** and **onsocimc**, are available for Informix MaxConnect users. Informix MaxConnect manages large numbers (from several hundred to tens of thousands) of client/server connections. The ratio of client connections to database connections can be 100:1 or higher. Informix MaxConnect increases system scalability to many thousands of connections and saves system resources, reducing response times and CPU requirements. You can install Informix MaxConnect on the client application server, on a dedicated server, or on the database server computer.

For more information, see the *IBM Informix MaxConnect User's Guide*.

IBM Office Connect

IBM Office Connect enables your Excel worksheets to access, display, and modify data from Informix and other ODBC databases.

For more information, see the *IBM Informix Connect User's Guide*.

IBM Informix Data Director for Web

IBM Informix Data Director for Web provides a model-driven development environment designed explicitly for creating powerful database applications that can grow with your business, addressing both evolving enterprise needs and increasingly diverse technical requirements. Data Director for Web has the following capabilities:

- Automates all of the data access operations of the client application
- Eliminates the task of writing data-access code
- Allows developers to easily incorporate sophisticated functionality without having to be database programming experts
- Helps project teams improve time to market with scalable applications that solve real business problems
- Enables interactive Web sites with Informix Web DataBlade module

DataBlade Modules

DataBlade modules extend the capabilities of Dynamic Server with user-defined objects. Available DataBlade modules include:

- IBM Informix Image Foundation DataBlade module
- IBM Informix Excalibur Text Search DataBlade Module
- IBM Informix Geodetic DataBlade Module
- IBM Informix TimeSeries DataBlade Module
- IBM Informix TimeSeries Real Time Loader DataBlade module
- Informix Video Foundation DataBlade Module
- IBM Informix Web DataBlade Module

For a brief description of each of these, see “DataBlade Publications” on page 6-4.

Client Products for Dynamic Server

For some client APIs, you can choose between two drivers that have different communication protocols:

Informix proprietary protocol

Informix Client SDK and the Informix JDBC driver

Informix clients can connect only to Dynamic Server. They support the full range of Dynamic Server functionality and are compatible with all earlier, supported versions of Dynamic Server.

Distributed Relational Database Architecture™ (DRDA®)

IBM Data Server drivers

Data Server clients can connect to both Dynamic Server and DB2 database servers. Data Server clients have some Dynamic Server-specific functionality restrictions, but they typically provide a larger range of functionality than Informix clients. Data Server clients support Informix Dynamic Server Version 11.10 and later.

JDBC Drivers

You can use either of the following JDBC drivers with Dynamic Server:

IBM Informix JDBC Driver

Included with Dynamic Server on most platforms.

IBM Data Server Driver for JDBC and SQLJ

Available on the Informix download site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd

.NET Providers

You can use either of the following .NET Providers with Dynamic Server:

IBM Informix .NET Provider

The Informix .NET Provider is provided with Client SDK and includes the Visual Studio Add-Ins. The Informix .NET Provider is provided to support existing .NET applications.

IBM Data Server Driver for ODBC, CLI and .NET

Available on the Informix download site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd For IDS, the .NET Provider and Visual Studio Add-Ins are supported. The Data Server .NET Provider provides more functionality than the Informix .NET Provider and is recommended if you are writing new applications.

PHP Drivers

You can use either of the following open source PHP drivers with Dynamic Server:

PHP Driver for Informix

Available from the PHP Web site: http://pecl.php.net/package/PDO_INFORMIX. The PHP Driver for Informix requires Client SDK to be installed on the same computer.

PHP Driver for Data Server clients

Available from the PHP Web site: http://pecl.php.net/package/PDO_IBM. The PHP Driver for Data Server clients requires the IBM Data Server Driver for ODBC and CLI to be installed on the same computer. The IBM Data Server Driver for ODBC and CLI is available on the Informix download site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd.

Ruby on Rails Adapters

You can use either of the following open source Ruby on Rails adapters with Dynamic Server:

Ruby on Rails Adapter for Informix

Available from the Ruby Web site: <http://rubyforge.org/projects/ruby-informix> as a Ruby gem. The Ruby on Rails adapter for Informix requires Client SDK to be installed on the same computer.

Ruby on Rails Adapter for Data Server clients

Available from the Ruby Web site: <http://rubyforge.org/projects/rubyibm> as a Ruby gem. The Ruby on Rails adapter for Data Server clients requires the IBM Data Server Driver for ODBC and CLI to be installed on the same computer. The IBM Data Server Driver for ODBC and CLI is available on the Informix download site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd.

Other Informix Drivers

The following drivers are only available with Client SDK:

- IBM Informix ESQL/C
- IBM Informix GLS
- Informix Object Interface for C++
- IBM Informix ODBC Driver
- IBM Informix OLE DB Provider

Other Related IBM Products

You can use many IBM products with Informix Dynamic Server. The Informix Dynamic Server Interoperability with Other Products Web site shows which versions of other products are certified to work with specific versions of Dynamic Server. As new releases are certified, the Web site is updated: <http://www.ibm.com/software/data/informix/ids/interop/>.

Chapter 2. Using New Features in Dynamic Server

In This Chapter	2-5
+ New Features in Version 11.50.xC3 of IBM Informix Dynamic Server	2-6
+ Performing Unattended Installations on Mac OS X	2-6
+ Setting up Data Source Names on Mac OS X	2-6
+ Using SQL Admin API to Dynamically Update Configuration Parameters	2-6
+ Dynamically Updating the LTXEHWM, LTXHWM, and DYNAMIC_LOGS Configuration Parameters	2-7
+ Dynamically Updating Enterprise Replication Configuration Parameters in the ONCONFIG File.	2-7
+ Improved Consistency Reporting after Enterprise Replication Repair Operations	2-7
+ Administering and Monitoring Enterprise Replication with SQL Admin API	2-7
+ Improved SQL Tracing with the SQL Admin API	2-8
+ Changing the Size of the First Extent of a Table	2-8
+ Rolling Back SQL Transactions to a Savepoint.	2-8
+ Capturing Transactional Data with the Change Data Capture API	2-9
+ Basic Text Search DataBlade Module Supports High-Availability Clusters	2-9
+ Querying XML Attributes with the Basic Text DataBlade Module	2-9
+ Setting the Frequency of Error Checking for Smart Large Object Transmission	2-9
New Features in Version 11.50.xC2 of IBM Informix Dynamic Server	2-10
Reconfiguring Connection Manager while It's Running	2-10
Multiple Copies of Dynamic Server on the Same Windows Computer	2-10
Enhanced Dynamic Server Installation Application on Mac OS X	2-11
Controlling Memory Use during Enterprise Replication Synchronization	2-11
Obtaining IDS Version Information From the cdr Utility	2-11
Monitoring Enterprise Replication with New SMI Tables.	2-11
Monitoring the Whole Enterprise Replication Domain	2-12
Preventing ATS or RIS File Generation.	2-12
The ISM Administrator Program is Included with Storage Manager on Windows	2-12
Limiting the Number of Sessions That Can Connect to Dynamic Server.	2-12
New Format for Backup Filters	2-13
Enhancements to the OpenAdmin Tool for IDS	2-13
Controlling I/O of B-Tree Indexes with Compression Levels	2-13
Subquery Support in UPDATE and DELETE Statements	2-13
Longer Return Strings from String Manipulation Functions	2-14
Server-Specific Audit Configuration File Functionality	2-14
New Features in Version 11.50.xC1 of IBM Informix Dynamic Server	2-14
SQL Admin API Commands to Configure High-Availability Clusters Added	2-15
Enhanced Connection Management for High-Availability Clusters	2-15
New Options to Troubleshoot High-Availability Clusters.	2-16
Update Data on Secondary Servers	2-16
Support for Transient Types on High-Availability Cluster Secondary Servers	2-16
Enhanced Configuration Options During Installation	2-16
Install as the Local System Account support added (Windows)	2-17
Enhanced Data Server Client Session Information	2-17
Improved onconfig.std and New Default Values for Configuration Parameters	2-17
Enhancements to the OpenAdmin Tool for IDS	2-22
Enhanced Shared-Memory Dump File Size Control	2-22
Enhanced Startup Script Customization	2-23
New Options for Configuring Storage Space Monitoring.	2-23
Updating Table Statistics with AUS Scheduler Tasks	2-23
Control External Directives for a Session	2-23
Improved Query Performance for Large Integers and Serial Data	2-24
Support for Obtaining Explain Output in XML Format Added.	2-24
DRDA Protocol Configuration During Installation Added	2-24
SQL Expressions with the IS [NOT] NULL Predicate	2-24
Determine Data Currency with a Version Column	2-24
Distributed Relational Database Architecture (DRDA) Enhancements	2-25

J/Foundation Upgraded to JRE 5.0	2-25
Improvements to the Basic Text Search DataBlade Module	2-25
New Support for Dynamic SQL Statements in SPL Routines	2-26
New Support for Extensible Stylesheet Language Transformation (XSLT)	2-26
Single Sign-on Support Added	2-26
Support for Encrypting Data by Using Secure Sockets Layer (SSL) Communications Added	2-26
New Features in Version 11.10.xC2 of IBM Informix Dynamic Server	2-27
Prevent Shared Memory Addresses from being used by IDS	2-27
Preview Enterprise Replication ATS or RIS Repair Operations	2-27
New Features in Version 11.10 of IBM Informix Dynamic Server	2-27
Multiple Remote Standalone Secondary Servers.	2-28
Multiple Shared Disk Secondary Servers	2-29
Backup and Restore to Directories with ontape	2-29
Continuous Logical Log Restore	2-29
Encrypted Communications for HDR	2-29
Improved Parallelism during Backup and Restore	2-30
Automatic Ordering of dbspaces during Backup and Restore	2-30
RTO Policy to Manage Server Restart	2-30
Non-blocking Checkpoints.	2-30
Performance Improvements for Enterprise Replication	2-30
ON-Bar Performance Report	2-31
Transform Data during Backup and Restore	2-31
Improved Performance for Cooked Files with Direct I/O on UNIX	2-31
Improved Performance of Online Index Creation	2-31
SQL Administration API	2-32
Schedule Administrative Tasks	2-32
Monitor and Analyze Recent SQL Statements	2-32
Dynamically Change Enterprise Replication Configuration Parameters and Environment Variables	2-33
Dynamically Rename Enterprise Replication Columns, Tables, and Databases.	2-33
Truncate Replicated Tables.	2-33
Improved Statistics Maintenance.	2-33
Installation Improvements on Windows Platforms	2-33
Session Configuration Routines	2-34
Multiple Users for Administration Mode	2-34
PHP-based OpenAdmin Tool for IDS	2-34
Named Parameters in a JDBC CallableStatement	2-34
Index Binary Data Types	2-35
Trigger Enhancements	2-35
Derived tables in the FROM Clause of Queries	2-36
Index Self-Join Query Plans	2-36
Optimizer Directives in ANSI-Compliant Joined Queries.	2-36
Deployment Wizard	2-37
Enhanced Concurrency with Committed Read Isolation	2-37
Enhanced Data Types and UDR Support in Cross-Server Distributed Operations	2-37
XML Publishing	2-38
Index Hierarchical Data.	2-38
Basic Text Search	2-38
Improved Concurrency with Private Memory Caches for Virtual Processors	2-39
Web Feature Service for Geospatial Data	2-39
Support for Data Server Clients with DRDA.	2-39
Statement Labels, GOTO, and LOOP Statements in the SPL language	2-39
New SQL Functions	2-40
Automatic Re-Compilation of Prepared Statements	2-40
Label-Based Access Control	2-41
New Features in Version 10.00 of IBM Informix Dynamic Server	2-42
New Features in Version 10.00.xC4	2-42
TRUNCATE Table support.	2-42
Enterprise Replication Direct Synchronization	2-42
Enterprise Replication Consistency Checking	2-42
Enhanced Support for IPv6	2-42
Secure Local Connections	2-42

Secure DataBlade Module Paths	2-42
DB-Access Stops a Process After the First Error	2-43
Informix Interface for Tivoli Storage Manager Supports HP-UX (Itanium)	2-43
New Default Value for IFX_EXTEND_ROLE Configuration Parameter	2-43
New Features in Version 10.00.xC3	2-43
ANSI-Joins in Distributed Queries	2-43
Transaction Support for XA-Compliant External Data Sources	2-44
MQ DataBlade Module	2-44
Restricting Database Creation	2-44
Secure DUMPDIR Configuration Parameter Default Directory	2-44
Table-Level Restore Support for Smart Large Objects	2-44
AES Ciphers for Network Encryption	2-44
Enterprise Replication Statistics Commands	2-44
CSDK Included in Dynamic Server Installation	2-45
Enhanced Support for Retrieving Subsets of Query Results	2-45
Returning Subsets of Query Results as Collection-Derived Tables	2-45
J/Foundation Upgraded to JRE 1.4.2	2-45
Secure ADTPATH Configuration Parameter Default Directory	2-45
UNSECURE_ONSTAT Configuration Parameter	2-45
New Features in Version 10.00.xC1	2-45
Security Enhancements	2-46
Server Usability Enhancements	2-47
Performance Enhancements	2-50
SQL Enhancements	2-51
Enterprise Replication Enhancements	2-52
Backup and Restore Enhancements	2-53
Storage Enhancements	2-54
Extensibility Enhancements	2-54
Installation Enhancements	2-55
New installation program on UNIX and Linux	2-55
Interoperability Enhancements	2-55
New Features in Version 9.4	2-56
Security Enhancement	2-56
Database Server Usability Enhancements	2-56
Increase Size of Chunks, Chunk Offsets, and Number of Allowable Chunks	2-56
Configurable Event Alarms	2-57
Increased Database Server Aliases	2-57
Increased File Size Limit	2-57
Full Use of Storage Media	2-57
Increased Default Values for Tape Block Size Configuration Parameters	2-58
Chunk Reserve Pages in Non-Root Chunks	2-58
Restartable Fast Recovery	2-58
Microsoft Transaction Server/XA Support	2-58
Performance Enhancements	2-59
PDQ is Enabled for Hold Cursors	2-59
Improved Transaction Processing with the B-tree Scanner	2-59
Improved Priority Management for the Buffer Manager	2-59
Spatial Query Costing	2-59
More Precise LRU Maximum and Minimum Settings	2-59
Enterprise Replication Enhancements	2-59
Enterprise Replication Security	2-60
Support for ROW and Collection Data types	2-60
Faster Queue Recovery	2-60
Replication During Queue Recovery	2-60
Large Transactions Support	2-60
Improved Availability with HDR	2-60
Dynamic Log File	2-60
New Commands	2-60
New and Altered Configuration Parameters	2-60
New Environment Variables	2-61
Extensibility Enhancements	2-61

Enhanced HDR Support for Extensibility Features	2-61
Using an Iterator Function in the FROM Clause of a SELECT Statement	2-62
Enhanced CREATE FUNCTION and CREATE PROCEDURE Syntax	2-62
SQL Enhancements	2-62
INSTEAD OF Triggers on Views	2-62
Enhanced SELECT Statement Syntax	2-63
Functional Indexes on More Than 16 Columns	2-63
Enhanced Dynamic Query Support	2-63
Session-Level Non-Default Collation	2-63
LOAD TO and UNLOAD FROM with Large Files	2-64
SET Residency Statements No Longer Needed	2-64
Multiple OUT Parameters	2-64
Sequence Objects	2-65
ANSI Join Syntax	2-65
Unions in Subqueries of SELECT Statements	2-65
LVARCHAR Data Types Greater Than 2048 Bytes	2-65
New SQL Reserved Words	2-65
New Environment Variables	2-66
GLS Enhancements	2-66
Support for Unicode	2-66
Support for Unicode Collation	2-66
Full Support for Chinese GB18030-2000 Locale	2-66
Reliability, Availability, and Supportability Features	2-66
Dynamically Monitor Queries	2-67
Print the Session Control Block Address	2-67
Display Environment Variable Settings	2-67
Print Online Chunk Pages	2-67
Display Stored Procedure Information	2-67
DataBlade API Enhancements	2-67
New mi_get_db_locale() Function	2-67
New mi_get_transaction_id() Function	2-67
New mi_realloc() function	2-67
New mi_stack_limit() Function	2-68
New mi_system() Function	2-68
Enhanced Stream Support	2-68
High-Performance Loader Enhancements	2-68
Full Use of Storage Media	2-68
New Location for the Custom-Code Shared Library File	2-68
Custom-Code Function Input and Output Length	2-68
Backup and Restore Enhancements	2-68
Renaming Chunks During a Cold Restore	2-69
Full Use of Storage Media and Increased File Size Limit	2-69
Installation Enhancements	2-69
No Files Installed in the /usr/lib Directory	2-69
More Recent Client and GLS Files Are Not Overwritten	2-69
Serial Number and Key are No Longer Needed	2-69
New Features in Version 9.3	2-69
UNIX Bundle Installer	2-70
Database Server Usability Enhancements	2-70
Ability to Display the Maximum Number of Connections	2-70
Changes to onconfig.std File	2-70
Database Server Administration Utilities (Windows)	2-70
High-Availability Data Replication Failover Scripts	2-71
DataBlade API Enhancements	2-71
New PER_STMT_EXEC and PER_STMT_PREP Memory Durations	2-71
NULL Connections for mi_lo() Functions	2-71
New mi_collection_card() Function to Obtain Cardinality for Collections	2-71
Access to Files on a Client Computer One Buffer at a Time	2-72
New Callbacks to Handle Transactions	2-72
New Function for Determining the Transaction State for DataBlade Modules	2-72
Enterprise Replication Enhancements	2-72

Replication of Extensible Data Types	2-73
Support Functions for Replication of User-Defined Types	2-73
Performance Enhancements to Enterprise Replication	2-73
SERIAL Column Primary Keys	2-74
Replicate Sets and Exclusive Replicate Sets	2-74
Replicating Only Changed Columns	2-74
Spooling of Replicate Data to Nonlogging Smart Large Objects	2-74
In-Place Alters to Add or Drop Shadow Columns (CRCOLS)	2-75
New onstat Options for Enterprise Replication	2-75
The cdr finderr Utility	2-75
Extensibility Enhancements	2-75
DeepCopy Function for Multirepresentational Data Types	2-75
Nearest Neighbor Queries in R-Trees	2-76
Temporary Sbspaces and Smart Large Objects	2-76
Improved Space Allocation of User Data and Metadata in Sbspaces	2-76
J/Foundation Enhancements	2-76
JVM 1.3 Support in J/Foundation	2-77
Performance Enhancements	2-77
Configurable Default Lock Modes	2-77
The onstat -g stm Option	2-77
The Ability to Display the Query Plan Without Executing the Query	2-77
Dynamic Addition of Logical Logs	2-78
SQL Enhancements	2-78
Optional FROM in the DELETE Statement	2-78
REVOKE AS User	2-78
New Features in Dynamic Server, Version 9.21	2-78
ANSI Join Syntax	2-78
Rename Index Statement	2-79
Nonlogging (Raw) Tables	2-79
onpladm Utility	2-79
The onbar -b -l Command	2-79
9.x DB-Access to 7.x Synonyms	2-79
SQL Statement Cache Improvements	2-79
DataBlade API Features.	2-80
Functions for Controlling the Virtual-Processor Environment	2-80
Functions for Getting Information About a UDR	2-80
Java Features in 9.21.	2-81
JVM 1.2 Support in J/Foundation	2-81
Default Values of Java Configuration Parameters	2-81
JDBC 2.0 Support.	2-81
GLS Support for J/Foundation	2-81
update_jars.sql Script	2-81
Java Runtime Environment Variables	2-81
Partial Support for Variable-Length Opaque-Types.	2-81
References to J/Foundation Features	2-82
MaxConnect Support	2-82

In This Chapter

This chapter describes the new features in Dynamic Server, Version 11.50, 11.10, 10.00, 9.4, 9.3, and 9.21.

Important: See your release notes and documentation notes for the latest information on new features.

+ New Features in Version 11.50.xC3 of IBM Informix Dynamic Server

+ IBM Informix Dynamic Server, Version 11.50.xC3 contains new functionality in the
+ following areas:

- + • Administration
 - + – Performing Unattended Installations on Mac OS X
 - + – Setting up Data Source Names on Mac OS X
 - + – Using SQL Admin API to Dynamically Update Configuration Parameters
 - + – Dynamically Updating the LTXEHWM, LTXHWM, and DYNAMIC_LOGS Configuration Parameters
 - + – Dynamically Updating Enterprise Replication Configuration Parameters in the ONCONFIG File
 - + – Improved Consistency Reporting after Enterprise Replication Repair Operations
 - + – Administering and Monitoring Enterprise Replication with SQL Admin API
 - + – Improved SQL Tracing with the SQL Admin API
- + • Performance
 - + – Changing the Size of the First Extent of a Table
- + • Application Development
 - + – Rolling Back SQL Transactions to a Savepoint
 - + – Server-Specific Audit Configuration File Functionality
 - + – Capturing Transactional Data with the Change Data Capture API
 - + – Basic Text Search DataBlade Module Supports High-Availability Clusters
 - + – Querying XML Attributes with the Basic Text DataBlade Module
 - + – Setting the Frequency of Error Checking for Smart Large Object Transmission

+ Performing Unattended Installations on Mac OS X

+ Now you can install Dynamic Server, Client Software Development Kit (Client
+ SDK), and Informix Connect on Mac OS X by setting installation information in the
+ new **bundle.ini** file, which is provided on the installation media as a template, and
+ running an installation command. In earlier releases, to install these products on
+ Mac OS X, you had to start the installation process and provide installation
+ information interactively.

+ This installation process is documented in the *IBM Informix Dynamic Server*
+ *Installation Guide for UNIX, Linux, and Mac OS X*.

+ Setting up Data Source Names on Mac OS X

+ You can now set up IBM Informix ODBC Driver data source names (DSNs) by
+ using the Apple ODBC Administrator on computers running Mac OS X.

+ This feature is documented in the *IBM Informix ODBC Driver Programmer's Manual*.

+ Using SQL Admin API to Dynamically Update Configuration Parameters

+ You can dynamically set configuration parameters by using the new SQL Admin
+ API SET ONCONFIG commands. These new commands provide the same function
+ as the **onmode -wf** or **onmode -wm** commands. Use the SET ONCONFIG or SET

ONCONFIG PERMANENT commands to change configuration parameters in the ONCONFIG file. Use the SET ONCONFIG MEMORY command to configure parameters only for the current session.

The SET ONCONFIG commands are described in the *IBM Informix Guide to SQL: Syntax*.

Dynamically Updating the LTXEHWM, LTXHWM, and DYNAMIC_LOGS Configuration Parameters

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.

These onmode commands are documented in the *IBM Informix Dynamic Server Administrator's Reference*.

Dynamically Updating Enterprise Replication Configuration Parameters in the ONCONFIG File

Previously, the **cdr add onconfig**, **cdr change onconfig**, and **cdr remove onconfig** commands dynamically updated Enterprise Replication configuration parameters for the current session only. Now these commands create persistent updates to Enterprise Replication configuration parameters in the ONCONFIG file.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Improved Consistency Reporting after Enterprise Replication Repair Operations

Consistency reports generated after repair operations might show temporary inconsistencies between servers until all replicated transactions are completed on the target server. The **-R** option for **cdr check replicate** and **cdr check replicateset** commands now re-checks the repaired rows. If any rows are inconsistent, the check is rerun every five seconds, up to a maximum of 12 times (for a total elapsed time of 60 seconds). If rows are still inconsistent after the 12th check, a description of those inconsistent rows is displayed.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Administering and Monitoring Enterprise Replication with SQL Admin API

Now you can use the SQL Admin API to administer and monitor Enterprise Replication with the same syntax elements that are in the **cdr** command-line syntax. The syntax for using Enterprise Replication **cdr** commands with the ADMIN and TASK routines is described in *IBM Informix Guide to SQL: Syntax*.

The **cdr** command-line syntax is described in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Improved SQL Tracing with the SQL Admin API

You can use these new commands to manage SQL tracing by databases, sessions, and users: SET SQL TRACING DATABASE, SET SQL TRACING SESSION, and SET SQL TRACING USER. Previously, you could only trace SQL for all databases at the server. Now you can control which databases to include in the SQL trace. You can also turn tracing on or off for a specific session, and specify whether you want to trace SQL statements run by specific users. You can also suspend and resume all tracing at the server, without deallocating any resources, by using the SET SQL TRACING SUSPEND and RESUME commands.

All SET SQL TRACING commands are described in the *IBM Informix Guide to SQL: Syntax*.

Changing the Size of the First Extent of a Table

When you create a table, you specify a first extent size based on the eventual estimated size of the table. If the table becomes larger or smaller than that estimate, you might want to change the size of the first extent to avoid either having too many extents or creating extents that are larger than necessary. You can now change the size of the first extent of a table in a dbspace. When you change the size of the first extent, Dynamic Server records the change (in the system catalog and on the partition page), but only makes the actual change when the table is rebuilt or a new partition or fragment is created.

This feature, including the syntax for the new MODIFY EXTENT SIZE clause that you use with the ALTER TABLE statement, is documented in the *IBM Informix Guide to SQL: Syntax*.

Rolling Back SQL Transactions to a Savepoint

You can now declare or reference savepoint objects in SQL statements. A savepoint identifies an arbitrary location within the statements of an SQL transaction. Within its transaction, the savepoint resembles a statement label to which the ROLLBACK statement of SQL can cancel any changes to the database that the transaction produced between the savepoint and the ROLLBACK statement. A client application that declares one or more savepoints within a transaction can implement error-handling logic that rolls back only the portion of the transaction that follows the specified savepoint, rather than cancelling the entire transaction if an exception occurs.

The following new SQL statements are implemented for savepoints:

- The SAVEPOINT statement creates a savepoint within the current SQL transaction.
- The RELEASE SAVEPOINT statement destroys a specified savepoint, as well as any other savepoints that exist between the RELEASE statement and the savepoint that it references.
- The ROLLBACK WORK TO SAVEPOINT statement discards changes to the schema of the database or to its data values by statements that follow the savepoint but the effects of DDL and DML statements that preceded the savepoint persist.

This release of Dynamic Server directly supports the SQL statements that declare or reference savepoint objects in DB-Access and in UDRs written in the SPL, C, or Java languages. Methods that support savepoints are also supported in JDBC applications. Applications written in ESQL/C that use the savepoint statements of SQL are also supported with this release of Client SDK.

This feature is documented in *IBM Informix Guide to SQL: Syntax* and *IBM Informix JDBC Driver Programmer's Guide*.

Capturing Transactional Data with the Change Data Capture API

Use the Change Data Capture (CDC) API with your client applications to capture transactional data from Dynamic Server databases. The API includes a new system database called syscdc, which includes built-in SQL functions that control data capture.

This feature is documented in a new publication, the *IBM Informix Change Data Capture API Programmer's Guide*.

Basic Text Search DataBlade Module Supports High-Availability Clusters

You can now use the Basic Text Search DataBlade module to perform searches on high-availability cluster servers by creating indexes in sbspaces. Previously, the Basic Text Search DataBlade module only supported the creation of indexes in extspaces, and thus could not participate in any queries on high-availability secondary servers and in backup and restore operations.

The Basic Text Search DataBlade module enhancements are documented in the *IBM Informix Database Extensions User's Guide*.

Querying XML Attributes with the Basic Text DataBlade Module

The Basic Text Search DataBlade module now supports searches on XML attributes in a document repository. The new **all_xmlattrs** parameter enables searches on all attributes that are contained in the XML tags or paths in a column that contains an XML document.

This feature is documented in the *IBM Informix Database Extensions User's Guide*.

Setting the Frequency of Error Checking for Smart Large Object Transmission

Use the **IFX_LOB_XFERSIZE** environment variable to specify the number of bytes in a CLOB or BLOB to transfer from a client application to the database server before checking for errors. The error check occurs each time the specified number of bytes is transferred. If an error occurs, the remaining data is not sent and an error is reported. If no error occurs, the file transfer will continue until it finishes. For example, if the value of **IFX_LOB_XFERSIZE** is set to 10485760 bytes (10 MB), error checking will occur after every 10485760 bytes of the CLOB or BLOB are sent. If you do not set this environment variable, the error check occurs after the entire BLOB or CLOB is transferred.

You should adjust the value of **IFX_LOB_XFERSIZE** to suit your environment. Set **IFX_LOB_XFERSIZE** low enough so that transmission errors of large BLOB or CLOB data types are detected early, but not so low that excessive network resources are consumed.

This feature is currently documented in the documentation notes for the *IBM Informix Guide to SQL: Reference*.

New Features in Version 11.50.xC2 of IBM Informix Dynamic Server

IBM Informix Dynamic Server, Version 11.50.xC2 contains new functionality in the following areas:

- High Availability
 - Reconfiguring Connection Manager while it's running
- Administration
 - Multiple copies of Dynamic Server on one Windows® operating system
 - Enhanced Dynamic Server installation application on Mac OS X
 - Limiting memory use during Enterprise Replication synchronization
 - Obtaining IDS version information from the **cdr** utility
 - Monitoring Enterprise Replication with new SMI tables
 - Monitor the whole Enterprise Replication domain
 - Prevent ATS or RIS file generation
 - The ISM Administrator Program is included with Storage Manager on Windows
 - Limiting the number of sessions that can connect to Dynamic Server
 - New format for backup filters
 - Enhancements to the OpenAdmin Tool for IDS
- Performance
 - Controlling I/O of B-Tree Indexes with Compression Levels
- Application Development
 - Subquery support in UPDATE and DELETE statements
 - Longer return strings from string manipulation functions
- Security
 - Server-Specific Audit Configuration File Functionality

Reconfiguring Connection Manager while It's Running

In past releases, you had to stop the Connection Manager to modify how it was configured, and then restart it for the new configuration to take effect. With the new reload option (**-r**), you can reconfigure the Connection Manager without stopping and restarting it.

Connection Manager is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and the *IBM Informix Dynamic Server Administrator's Reference*.

Multiple Copies of Dynamic Server on the Same Windows Computer

In version 11.10, you could not install and run multiple copies of the same version of Dynamic Server on the same Windows computer. Now you can do so by using the graphical user interface or by supplying installation parameters in a file to perform a silent, non-interactive installation. If the installation program detects that the same version of Dynamic Server is already installed, you can choose to install a new copy in another directory or you can choose to modify the existing installation.

This feature is documented in the *IBM Informix Dynamic Server Installation Guide for Windows*.

Enhanced Dynamic Server Installation Application on Mac OS X

The Dynamic Server installation application can automatically tune the kernel settings to values that support a working instance of the database server on your computer. In addition, after setup of a demonstration database server is complete, a terminal icon appears inside the installation directory for easier navigation.

This feature is documented in the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux[®], and Mac OS X*.

Controlling Memory Use during Enterprise Replication Synchronization

Enterprise Replication uses the value of the CDR_QUEUEMEM configuration parameter as the size of the send queue during a synchronization operation. To specify a larger or smaller size for the send queue during a particular synchronization operation, use the **--memadjust** option for the **cdr sync replicate** and **cdr sync replicateset** commands. If you synchronize with the **cdr start replicate**, **cdr start replicateset**, or **cdr realize template** commands, you can specify that the synchronization operation is performed as a foreground process with the new **--foreground** option, and adjust the send queue size with the **--memadjust** option.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Obtaining IDS Version Information From the cdr Utility

In previous releases, you could not obtain product version information by using the cdr utility. You can now use the **cdr -V** command to display the version of IDS that is currently running.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Monitoring Enterprise Replication with New SMI Tables

SMI tables provide a way to obtain information by using SQL commands, either locally or from remote servers. The following new SMI tables contain information about Enterprise Replication that you can use to monitor status and diagnose problems:

syscdr_state

Contains information on whether Enterprise Replication, data capture, data apply, and the network between servers is active.

syscdr_ddr

Contains information about the status of log capture and the proximity or status of transaction blocking (DDRBLOCK) or transaction spooling.

syscdr_nif

Contains information about network connections and the flow of data between Enterprise Replication servers.

syscdr_rcv

Contains information about transactions being applied on target servers and acknowledgements being sent from target servers.

syscdr_atmdir

Contains information about the contents of the ATS directory.

syscdr_risdir

Contains information about the contents of the RIS directory.

syscdr_ats

Contains the first ten lines of the header of each ATS file.

syscdr_ris

Contains the first ten lines of the header of each RIS file.

syscdr_rqmstamp

Contains information about which transaction is being added into each queue.

syscdr_rqmhandle

Contains information about which transaction is being processed in each queue.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Monitoring the Whole Enterprise Replication Domain

You can now monitor the status of all Enterprise Replication servers from any one of those servers with the new **cdr view** command. Specify one or more subcommands, depending on what information you want to monitor. You can automatically rerun the **cdr view** command every specified interval by using the **--repeat** option.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Preventing ATS or RIS File Generation

You can prevent generating ATS or RIS files by setting the ATS or RIS directory to **/dev/null** on UNIX and **NUL** on Windows.

This feature is documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

The ISM Administrator Program is Included with Storage Manager on Windows

You can use the ISM Administrator program to manage your ISM server and storage devices from a graphical interface on Windows platforms. You can monitor the progress of your backup and restore operations, manage backed-up data, and control your storage media and storage devices through the ISM Administrator program.

The ISM Administrator program is documented in the *IBM Informix Storage Manager Administrator's Guide*.

Limiting the Number of Sessions That Can Connect to Dynamic Server

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.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Reference*.

New Format for Backup Filters

Backups have a new format for version 11.50.xC2 if they are performed with the **ontape** command using a filter specified by the **BACKUP_FILTER** configuration parameter. The new backup format is not backwards compatible. Any backups that were made prior to version 11.50.xC2 must be restored with the earlier versions of the filter.

Enhancements to the OpenAdmin Tool for IDS

The OpenAdmin Tool for IDS Version 2.22 has the following enhancements:

- Enhanced security for the SQL Toolbox features
- Enterprise Replication Plug-in for OpenAdmin Tool for IDS, Version 1.1:
 - A Routing Topology page for monitoring Enterprise Replication nodes at a domain level
 - New Node Details pages to view errors and configuration settings for individual ER nodes

The OpenAdmin Tool for IDS has enhanced security for the SQL Toolbox features.

See the OpenAdmin Tool for IDS release notes for more information.

Controlling I/O of B-Tree Indexes with Compression Levels

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 an SQL API function with a **SET INDEX COMPRESSION** command.

This feature is primarily documented in the *IBM Informix Dynamic Server Performance Guide*.

Subquery Support in UPDATE and DELETE Statements

The **FROM** clause of a subquery in the **WHERE** clause of the **DELETE** or **UPDATE** statement can specify as a data source the same table or view that the **FROM** clause of the **DELETE** or **UPDATE** statement specifies. In this version, Dynamic Server supports **DELETE** and **UPDATE** operations with subqueries that reference the same table object if all of the following conditions are true:

- The subquery either returns only a single row, or else has no references to columns in any table that is not listed in its **FROM** clause.
- The subquery is specified in the **WHERE** clause of the outer **DELETE** or **UPDATE** statement, using the Condition with Subquery syntax.

If you call SPL routines within the same subquery, these cannot reference the table that is being modified.

This feature is documented in the *IBM Informix Dynamic Server Guide to SQL: Syntax*.

Longer Return Strings from String Manipulation Functions

The following SQL string-manipulation functions now support operations where the returned string can be up to 32 KB in length:

- LPAD
- RPAD
- REPLACE
- SUBSTR
- SUBSTRING
- TRIM
- LTRIM
- RTRIM
- ENCRYPT_AES
- ENCRYPT_TDES
- DECRYPT_CHAR
- DECRYPT_BINARY
- CONCAT (and the || operator)

In addition, the **CONCAT** function and the || operator now provide native support for the LVARCHAR data type.

This feature is documented in the *IBM Informix Dynamic Server Guide to SQL: Syntax*.

Server-Specific Audit Configuration File Functionality

Each server instance uses its own `adtcfg.servernumber` file if the audit configuration parameters are changed with the **onaudit** utility. The new **-n** option for the **onshowaudit** utility reads the server-specific audit configuration (`adtcfg.servernumber`) file.

New Features in Version 11.50.xC1 of IBM Informix Dynamic Server

IBM Informix Dynamic Server, Version 11.50.xC1 contains new functionality in the following areas:

- High Availability
 - SQL Administration API commands to configure high availability clusters added
 - Enhanced connection management for high-availability clusters
 - Update data on secondary servers
 - New options to troubleshoot high-availability clusters
 - Support for transient types on high-availability cluster secondary servers
- Administration
 - Enhanced configuration options during installation
 - Install as the Local System User ID support added (Windows)
 - Enhancements to the OpenAdmin Tool for IDS
 - Improved onconfig.std and new default values for configuration parameters
 - Enhanced Data Server client session information

- Enhanced shared-memory dump file size control
- Enhanced startup script customization
- New Options for Configuring Storage Space Monitoring
- Updating Table Statistics with AUS Scheduler Tasks
- Performance
 - Control external directives for a session
 - Improved Query Performance for Large Integers and Serial Data
 - Support for Obtaining Explain Output in XML Format Added
- Application Development
 - DRDA Protocol Configuration During Installation Added
 - SQL expressions with the IS [NOT] NULL predicate
 - Determine data currency with a version column
 - Distributed Relational Database Architecture (DRDA) Enhancements
 - J/Foundation Upgraded to JRE 5.0
 - Improvements to the Basic Text Search DataBlade Module
 - New Support for Dynamic SQL Statements in SPL Routines
 - New Support for Extensible Stylesheet Language Transformation (XSLT)
- Security
 - Single Sign-on Support (SSO) Added
 - Support for Encrypting Data by Using Secure Sockets Layer (SSL) Communications Added

SQL Admin API Commands to Configure High-Availability Clusters Added

You can now use SQL Administration API commands to perform the equivalent of **onmode -d** commands to configure high-availability clusters.

SQL Administration API commands are documented in the *Guide to SQL: Syntax*.

Enhanced Connection Management for High-Availability Clusters

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.

Connection Manager is installed with Client SDK. Connection Manager requires that applications connecting to it be compiled with Client SDK 3.50 or Data Server Driver for JDBC and SQLJ.

Connection Manager is primarily documented in the *Administrator's Guide* and the *Administrator's Reference*.

New Options to Troubleshoot High-Availability Clusters

The following new **onstat** commands provide diagnostic information about high-availability clusters:

- **onstat -g dri**
- **onstat -g sds**
- **onstat -g rss**
- **onstat -g proxy**

The following new **sysmaster** tables contain equivalent information as the new **onstat** commands:

- **sysproxyagents**
- **sysproxydistributers**
- **sysproxysessions**
- **sysproxytxnops**
- **sysproxytxns**

These options are documented in the *Administrator's Guide* and *Administrator's Reference*.

Update Data on Secondary Servers

You can configure secondary servers so that client applications can send them transactions that update data. The ability to update secondary servers is not enabled by default. To enable updates to secondary servers, set the **UPDATABLE_SECONDARY** configuration parameter to the number of SMX pipes between the primary and secondary servers. The maximum value of this parameter is two times the number of CPU virtual processors. To set the secondary server as read-only, which is the default behavior, set **UPDATABLE_SECONDARY** to 0.

This feature is documented in the *Administrator's Guide* and the *Administrator's Reference*.

Support for Transient Types on High-Availability Cluster Secondary Servers

Transient unnamed complex data types (ROW, SET, LIST, and MULTiset) can now be used on high-availability cluster secondary servers, whether the secondary servers are read-only or use updatable secondary functionality.

The following types of operations that use transient types are supported on secondary servers:

- SQL queries that explicitly use transient types
- SQL queries that use derived tables, collection subqueries, and XML functions (these statements implicitly use transient types)
- Temporary tables created with the **CREATE TEMP** statement that use transient types

This feature is primarily documented in the *Administrator's Guide*.

Enhanced Configuration Options During Installation

You can use the new Instance Configuration Wizard to automatically create the database server configuration file (ONCONFIG) during installation. Access the wizard in the installation program with either GUI or console installation modes:

- UNIX and Linux: Select to create a demonstration database server and choose to customize the default configuration file
- Windows: Choose custom installation setup, select "Initialize Server," and select the option to enable a default configuration file for your hardware and user needs

Provide the information for the instance that you are installing, such as the number of CPUs, memory, disk space, and estimates of online transactions and query clients. The wizard ensures that your settings are valid, and it calculates values for other server configuration parameters based on your settings. Your custom configuration information is stored in the ONCONFIG file so that when you start the instance after the product is installed, the instance runs with your settings.

This feature is documented in the *IBM Informix Dynamic Server Installation Guide for Microsoft Windows* and the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux*.

Install as the Local System Account support added (Windows)

In past releases, the IDS Windows Service was only allowed to log on as the **informix** user.

Starting with version 11.50, you can install IDS on Windows as the local system account. Select the Local System User option in the installation program. That option provides the same privileges as the **informix** user account; however, it uses an internal account that does not require a password. The local system account is used by the operating system and services running under Windows during the installation of IDS.

You can choose not to create an **informix** user account at all, but if you do so, you will not be able to use Enterprise Replication between IDS on UNIX and Windows platforms.

This feature is documented in the *IBM Informix Dynamic Server Installation Guide for Microsoft Windows*.

Enhanced Data Server Client Session Information

You can use the new **sys sesappinfo** table in the **sysmaster** database to view Data Server client session information. The table shows the client session ID, session application name, and a session value in the **sesapp_sid**, **sesapp_name**, and **sesapp_value** columns.

You can now also display client session information using the **onstat -g ses** command.

The **sys sesappinfo** table and the **onstat -g ses** command are documented in the *Administrator's Reference*.

Improved onconfig.std and New Default Values for Configuration Parameters

The IDS configuration file, **onconfig.std**, has been improved in the following ways:

- 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 have now have higher values.
- Some configuration parameters that specify file locations now have more secure default locations under the \$INFORMIXDIR directory.

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
```

The following table lists the configuration parameters that are added to the `onconfig.std` file and their values.

Table 2-1. Configuration Parameters added to onconfig.std

Configuration Parameter	Value
ADMIN_USER_MODE_WITH_DBSA	none
BTSCANNER	num=1,priority=low,threshold=5000, rangesize=-1,alice=6
BACKUP_FILTER	none
BAR_DEBUG	0
CDR_SUPPRESS_ATSRISWARN	none
DD_HASHMAX	10
DD_HASHSIZE	31
DEF_TABLE_LOCKMODE	page
DS_HASHSIZE	31
DS_POOLSIZE	127
ENCRYPT_CDR	none
ENCRYPT_CIPHERS	none
ENCRYPT_HDR	none
ENCRYPT_MAC	none
ENCRYPT_MACFILE	none
ENCRYPT_SMX	none
ENCRYPT_SWITCH	none
EXT_DIRECTIVES	0
FAILOVER_CALLBACK	none
FASTPOLL	1
HA_ALIAS	none
LOG_INDEX_BUILDS	none
MAX_INCOMPLETE_CONNECTIONS	1024
PC_HASHSIZE	31

Table 2-1. Configuration Parameters added to onconfig.std (continued)

Configuration Parameter	Value
PC_POOLSIZE	127
PLCY_HASHSIZE	127
PLCY_POOLSIZE	31
PLOG_OVERFLOW_PATH	UNIX: \$INFORMIXDIR/tmp Windows: none
RESTORE_FILTER	none
SBSPACE_TEMP	none
SDS_ENABLE	none
SDS_PAGING	none
SDS_TEMPDBS	none
SDS_TIMEOUT	20
SECURITY_LOCALCONNECTION	none
SQLTRACE	Commented out: # SQLTRACE level=low,ntraces=1000,size=2,mode=global
SSL_KEYSTORE_LABEL	none
STMT_CACHE	0
STMT_CACHE_HITS	0
STMT_CACHE_NOLIMIT	0
STMT_CACHE_NUMPOOL	1
STMT_CACHE_SIZE	512
TEMPTAB_NOLOG	0
UNSECURE_ONSTAT	none
UPDATABLE_SECONDARY	0
USRC_HASHSIZE	31
USRC_POOLSIZE	127
VPCLASS	cpu,num=1,noage Commented out: # VPCLASS aio,num=1 Commented out: #VPCLASS jvp,num=1

The following table lists configuration parameters whose value in onconfig.std file has been updated.

Table 2-2. Configuration Parameters Updated in onconfig.std

Configuration Parameter	Previous value	New value
ADMIN_MODE_USERS	1	None
ALARMPROGRAM	UNIX: /usr/informix/etc/ alarmprogram.sh Windows: None	UNIX: \$INFORMIXDIR/etc/ alarmprogram.sh Windows: \$INFORMIXDIR\etc\ alarmprogram.bat

Table 2-2. Configuration Parameters Updated in onconfig.std (continued)

Configuration Parameter	Previous value	New value
BAR_ACT_LOG	/usr/informix/bar_act.log	UNIX: \$INFORMIXDIR/tmp/ bar_act.log Windows: \$INFORMIXDIR\tmp\ bar_act.log
BAR_BSALIB_PATH	UNIX: \$INFORMIXDIR/lib/ libsad001.so Window: libbsa.dll	None
BAR_DEBUG_LOG	UNIX: /usr/informix/bar_dbug.log Windows: bar_dbug.log	UNIX: \$INFORMIXDIR/tmp/ bar_dbug.log Windows: \$INFORMIXDIR\tmp\ bar_dbug.log
BUFFERPOOL	Operating systems with 2K page size: default,buffers=5000,lrus=8, lru_min_dirty=50,lru_max_dirty=60 size=2k,buffers=5000,lrus=8, lru_min_dirty=50,lru_max_dirty=60 Operating systems with 4K page size: default,buffers=1000,lrus=8, lru_min_dirty=50,lru_max_dirty=60 size=4k,buffers=1000,lrus=8, lru_min_dirty=50,lru_max_dirty=60	Operating systems with 2K page size: default,buffers=10000,lrus=8, lru_min_dirty=50.00, lru_max_dirty=60.50 size=2k,buffers=50000,lrus=8, lru_min_dirty=50, lru_max_dirty=60 Operating systems with 4K page size: default,buffers=10000,lrus=8, lru_min_dirty=50.00, lru_max_dirty=60.50 size=4k,buffers=10000,lrus=8, lru_min_dirty=50, lru_max_dirty=60
CLEANERS	1	8
CONSOLE	UNIX: /dev/console Windows: console.log	UNIX: \$INFORMIXDIR/tmp/ online.con Windows: online.con
DB_LIBRARY_PATH	commented out: # DB_LIBRARY_PATH \$INFORMIXDIR/extend	commented out: # DB_LIBRARY_PATH
DRLOSTFOUND	UNIX: /usr/etc/dr.lostfound Windows: \tmp	UNIX: \$INFORMIXDIR/etc/ dr.lostfound Windows: \$INFORMIXDIR\tmp
DUMPDIR	UNIX: /usr/informix/tmp Windows: INFORMIXDIR\tmp	UNIX: \$INFORMIXDIR/tmp Windows: \$INFORMIXDIR\tmp
EXPLAIN_STAT	1	0
LISTEN_TIMEOUT	10	60
LOCKS	2000	20000
LOGBUFF	32	64
LOGSIZE	2000	10000

Table 2-2. Configuration Parameters Updated in onconfig.std (continued)

Configuration Parameter	Previous value	New value
LTAPEDEV	UNIX: /dev/tapedev Windows: \\.\TAPE1	UNIX: /dev/tapedev (same as previous value) Windows: NUL
MIRRORPATH	None	UNIX: \$INFORMIXDIR/tmp/demo_on.root_mirror Windows: none
MSGPATH	UNIX: /usr/informix/online.log Windows: online.log	UNIX: \$INFORMIXDIR/tmp/online.log Windows: online.log
NETTYPE	UNIX: none Windows: onsoctcp,drsoctcp,1,NET	UNIX: ipcshm,1,50,CPU Windows: none
PHYSBUFF	32	128
PHYSFILE	2000	50000
RA_PAGES	None	64
RA_THRESHOLD	None	16
ROOTPATH	UNIX: /dev/online_root Windows: None	UNIX: \$INFORMIXDIR/tmp/demo_on.rootdbs Windows: None
ROOTSIZE	30000	200000
SHMVIRT_ALLOCSEG	0	0,3
SHMVIRTSIZE	8192	32656
SYSALARMPROGRAM	UNIX: /usr/informix/etc/evidence.sh Windows: INFORMIXDIR\etc\evidence.bat	UNIX: \$INFORMIXDIR/etc/evidence.sh Windows: Commented out: # SYSALARMPROGRAM \$INFORMIXDIR\etc\evidence.bat
TAPEBLK	32	UNIX: 32 Windows: 16
TAPESIZE	10240	0

The following configuration parameters were removed from the onconfig.std file:

- AFF_NPROCS
- AFF_SPROC
- NOAGE
- NUMCPUVPS
- PHYSDBS
- JDKVERSION

The changes to configuration parameter default values are documented in the *Administrator's Reference*.

Enhancements to the OpenAdmin Tool for IDS

The OpenAdmin Tool for IDS has the following enhancements:

- Enhanced high availability cluster management (MACH) support:
 - Starting and stopping a remote server in the cluster
 - Encryption for idsd
 - Connection Manager Wizard
 - SD Secondary Disk Setup Wizard
- New security features:
 - Read-only user group support
 - User privileges administration
- New Task Scheduler to create and modify Scheduler tasks
- New server administration tools:
 - Automated update of statistics
 - Dynamic update of configuration parameters
 - System validation
 - Virtual processor administration
- Improved performance monitoring tools:
 - Improved historical performance data tracking
 - New system reports
 - Memory pool statistics
- Improved usability:
 - Menu restructuring and new look and feel
 - New contextual help topics
 - HOWTO.html task guide

See the OpenAdmin Tool release notes for more information.

Enhanced Shared-Memory Dump File Size Control

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.

Use the DUMPSHMEM configuration parameter to automatically create a dump file when an assertion fails. Set DUMPSHMEM to 2 to create a shared memory dump that excludes the buffer pool. You can dynamically change the value of DUMPSHMEM with **onmode -wm** and **onmode -wf**.

Use the new **nobuffs** option for the **onstat -o** command to write the contents of shared memory without the buffer pool to a specified file. If you do not specify a file, the contents are written to a file called **onstat.out** in the current working directory.

If you use **onstat -o** without an option, the **nobuffs** or **full** option is controlled by the DUMPSHMEM configuration parameter setting.

Any **onstat** options, such as **onstat -i** (interactive), that work with a full dump file will work with a dump file that excludes the buffer pool.

This feature is primarily documented in the *Administrator's Reference*.

Enhanced Startup Script Customization

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. The **-w** option provides a return code so that you can check if the system started without incident. If the server fails to initialize within a configurable timeout period, the utility returns a 1 and writes a message to the `online.log` file. A return code of 0 indicates a successful initialization.

The **-w** option works with both disk-space initialization and shared-memory initialization **oninit** options.

In a high-availability environment, you can only use the **oninit -w** command on primary servers; it is not valid on secondary servers.

The **-w** option to the **oninit** utility is documented in the *Administrator's Reference*.

New Options for Configuring Storage Space Monitoring

You can configure how you are notified when a storage space or partition becomes full with the new `STORAGE_FULL_ALARMS` configuration parameter. By default, you will now receive alarms and messages in the online message log when a `dbspace`, `sbspace`, temporary `dbspace`, `blobospace`, or partition becomes full. You can specify the level of the alarms or disable them. You can also specify the time interval between alarms and messages. By default, level 3 alarms are enabled with an interval of 10 minutes.

This feature is documented in the *Administrator's Guide* and the *Administrator's Reference*.

Updating Table Statistics with AUS Scheduler Tasks

The new Auto Update Statistics (AUS) scheduler tasks can improve the performance of the database server by regularly providing the query optimizer with updated table statistics as the basis for efficient query plans.

- Use the **Auto Update Statistics Evaluation** task to generate a prioritized set of `UPDATE STATISTICS` statements. Built-in policies specify criteria for identifying the tables whose column distributions should be recalculated, but you can modify these default policies.
- Use the **Auto Update Statistics Refresh** task to run the statements during a recurring time interval that you set.

This feature is primarily documented in the *IBM Informix Performance Guide*.

Control External Directives for a Session

You now can use the new `EXTDIRECTIVES` session environment option of the `SET ENVIRONMENT` statement to control whether external directives are enabled, disabled, or have default behavior during a session. Specify the default directives behavior with the `EXT_DIRECTIVES` configuration parameter and the client-side `IFX_EXTDIRECTIVES` environment variable.

For more information, see the `SET ENVIRONMENT` syntax diagram and the section on the `EXTDIRECTIVES` option in the *Guide to SQL: Syntax*.

Improved Query Performance for Large Integers and Serial Data

The ISO standard data types BIGINT and BIGSERIAL are provided as alternatives to INT8 and SERIAL8. BIGINT and BIGSERIAL data types can provide better performance than the INT8 and SERIAL8 data types.

This feature is primarily documented in the *Guide to SQL: Reference*.

Support for Obtaining Explain Output in XML Format Added

EXPLAIN_SQL() is a new routine that prepares a query and returns a query plan in XML. The IBM Data Studio Administration Console can use EXPLAIN_SQL to obtain the XML query plan, interpret the XML, and render the plan visually. Vendors can use this routine to obtain explain output in XML format.

For information on using IBM Data Studio, see IBM Data Studio documentation.

DRDA Protocol Configuration During Installation Added

It's easier now than in past releases to set up an instance to use a variety of database clients. When you install IDS 11.50, the installer enables you to configure a database server alias and a port for clients that use the Distributed Relational Database Architecture (DRDA) protocol. By default, those items are configured for you unless you deselect DRDA support. DRDA is for open development of applications that allow access of distributed data.

This feature is documented in the *IBM Informix Dynamic Server Installation Guide for Microsoft Windows* and the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux*.

SQL Expressions with the IS [NOT] NULL Predicate

Now you can use SQL expressions as operands of the IS NULL and IS NOT NULL predicate. Previously, you could only use column name operands with IS NULL and IS NOT NULL, which return a TRUE or FALSE result.

The use of IS NULL or IS NOT NULL with expressions enables you to provide a value for entries that otherwise are not computable because NULL is not a valid numerical value. For more information on using IS NULL or IS NOT NULL with expressions, and for an example of a CASE expression that allows NULL values to be treated as 0 for the purpose of arithmetic computations, see the *Guide to SQL: Syntax*.

Determine Data Currency with a Version Column

You can now add a version column to a table to contain both a checksum and a version number. Use a version column to detect if a row has been updated since it was originally queried. Also, version numbers enable you to detect differences if a row is deleted and another row is re-inserted into a table. To add a version column, use the ALTER TABLE *tablename* ADD VERCOLS statement.

Web applications can use a version column to determine if information about a previously retrieved object is still current. For example, a Web application can display an item to a customer, and then when the customer decides to buy the item, the application checks the version column of the item's row to determine if the information about the item has changed.

A version column might also improve the updatable secondary performance in a high-availability cluster because during collision detection, the primary server only needs to compare the version of the updated row instead of the entire contents of the row.

This feature is documented in the *Guide to SQL: Syntax* and the *Administrator's Guide*.

Distributed Relational Database Architecture (DRDA) Enhancements

DRDA enables communication between applications and database systems on disparate platforms and enables relational data to be distributed among multiple platforms. Enhancements to DRDA functionality include:

- Support for retrieving ISAM errors
- Support for connecting to a database that has a name that is longer than 18 bytes
- Batching of multiple statements into one request
- Support for displaying DRDA session information in the new `sysesappinfo` table in the `sysmaster` database and with the `onstat -g ses` command
- Support for using `IS NULL` or `IS NOT NULL` with expressions
- Support for using the Secure Sockets Layer (SSL) protocol for DRDA communications
- Support for DRDA connections between primary and shared disk secondary servers in high availability clusters

These features are primarily documented in the *Administrator's Guide*.

J/Foundation Upgraded to JRE 5.0

The J/Foundation component of Dynamic Server has been upgraded from JRE 1.4.2 to JRE 5.0 (Java version 1.5.0). This upgrade supports JRE 5.0 and JDK 5.0 for server-based Java routines.

The JRE 5.0 upgrade has the following changes:

- New settings for the `JVPAVM` configuration parameter
- The `JDKVERSION` configuration parameter has been deprecated

Depending on your application requirements, you should test and possibly increase the value for the `JVM_MAX_HEAP_SIZE` environment variable to configure the heap size for the new JVM.

This feature is documented in the *J/Foundation Developer's Guide*.

Improvements to the Basic Text Search DataBlade Module

You have the following new facilities for using **bts** indexes:

- Obtain a list of fields in an index that you can search on with the new `bts_index_fields()` function
- Specify a custom stopword list with a new index parameter instead of using the default stopword list
- Index XML documents with new index parameters

To use these features, you must register this new version of the Basic Text Search DataBlade module in your databases.

These features are documented in the *Database Extensions User's Guide*.

New Support for Dynamic SQL Statements in SPL Routines

In earlier versions of the Informix SPL language, the form of SQL statements in a UDR was known when the SPL routine was written. Users can now specify queries dynamically at runtime in SPL routines, using the EXECUTE IMMEDIATE and PREPARE statements to execute dynamically constructed SQL statements and to define and manage cursors and prepared objects with the DECLARE, OPEN, FETCH, CLOSE, and FREE statements.

For some of these Dynamic SQL statements, the SPL syntax is a subset of the existing ESQL/C syntax, but character expressions that evaluate to SQL statement text are now supported by PREPARE and EXECUTE IMMEDIATE statements in SPL routines. In addition, Boolean conditions in SPL statements can specify SQLCODE as an expression that returns the SQLCA status from the most recently executed SQL statement.

This feature increases the flexibility of UDRs written in the SPL language, and simplifies the migration to IDS of stored procedures and applications written for other database servers that support Dynamic SQL operations. This feature is documented in the *Guide to SQL: Syntax*.

New Support for Extensible Stylesheet Language Transformation (XSLT)

You can now use built-in SQL functions to transform XML documents using extensible stylesheet language transformations (XSLTs). IDS supports the XSLT 1.0 standard for style sheets with the Xalan and Xerces libraries.

This feature is documented in the *XML User's Guide*.

Single Sign-on Support Added

With single sign-on, users authenticate themselves once and the authentication is carried securely over trusted and non-trusted networks. The users provide a valid user ID and password when they log in to a client computer, and they can access the database server and other SSO-enabled services without having to log in again. In the past, users had to log in multiple times.

IDS delivers support for SSO in the Generic Security Services Communications Support Module (GSSCSM). GSSCSM works with Kerberos, which is a network authentication protocol. You must deploy a Kerberos authentication protocol that supports the Generic Security Services Application Programming Interface (GSSAPI) before you can use SSO with IDS. After you enable SSO, you benefit from centralized management of authentication.

This feature is primarily documented in the *Security Guide*.

Support for Encrypting Data by Using Secure Sockets Layer (SSL) Communications Added

You can now configure IDS to use the SSL protocol, which encrypts data in TCP/IP connections between two points over a network. The SSL protocol is an

alternative to the IDS-specific encryption Communication Support Module (CSM) and simple password CSM for CSDK clients. You must use SSL to encrypt data in communications between IDS and DRDA clients.

This feature is primarily documented in the *Security Guide*.

New Features in Version 11.10.xC2 of IBM Informix Dynamic Server

IBM Informix Dynamic Server, Version 11.10.xC2 contains the following new features:

- Prevent Shared Memory Addresses from being used by IDS
- Preview Enterprise Replication ATS or RIS Repair Operations

Prevent Shared Memory Addresses from being used by IDS

You can specify virtual memory address ranges that IDS cannot use to attach shared memory with the SHMNOACCESS configuration parameter. 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 hex 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
```

The SHMNOACCESS configuration parameter is documented in the *Administrator's Reference*.

Preview Enterprise Replication ATS or RIS Repair Operations

You can preview the operations that an ATS or RIS repair operation performs by using the new **--check** option with the **cdr repair ats** or **cdr repair ris** command. The **--check** option specifies that the repair operations are displayed to stderr but not actually performed. For more information, issue the **cdr repair ats -h** or **cdr repair ris -h** command.

This feature is documented in the *Enterprise Replication Guide*.

New Features in Version 11.10 of IBM Informix Dynamic Server

Scalability, High Availability, and Performance

- Multiple Remote Standalone Secondary Servers
- Multiple Shared Disk Secondary Servers
- Backup and Restore to Directories with **ontape**
- Continuous Logical Log Restore
- Encrypted Communications for HDR
- Improved Parallelism during Backup and Restore
- Automatic Ordering of dbspaces during Backup and Restore
- RTO Policy to Manage Server Restart
- Non-blocking Checkpoints
- Performance Improvements for Enterprise Replication
- ON-Bar Performance Report

- Transform Data During Backup and Restore
- Improved Performance for Cooked Files with Direct I/O
- Improved Performance of Online Index Creation

Administration

- SQL Administration API
- Schedule Administrative Tasks
- Monitor and Analyze Recent SQL Statements
- Dynamically Change Enterprise Replication Configuration Parameters and Environment Variables
- Dynamically Rename Enterprise Replication Columns, Tables, and Databases
- Truncate Replicated Tables
- Improved Statistics Maintenance
- Installation Improvements on Windows Platforms
- Session Configuration Routines
- Multiple Users for Administration Mode
- OpenAdmin (a PHP-based admin utility)

Integrated Solutions

- Named Parameters in a JDBC CallableStatement
- Index Binary Data Types
- Trigger Enhancements
- Derived tables in the FROM Clause of Queries
- Index Self-Join Query Plans
- Optimizer Directives in ANSI-Compliant Joined Queries
- Deployment Wizard
- Enhanced Concurrency with Committed Read Isolation
- Enhanced Data Types and UDR Support in Cross-Server Distributed Operations
- XML Publishing
- Index Hierarchical Data
- Basic Text Search
- Improved Concurrency with Private Memory Caches for Virtual Processors
- Web Feature Services for Spatial Data
- Support for Common Clients with DRDA
- Statement Labels, GOTO, and LOOP Statements in the SPL language
- New SQL Functions
- Automatic Re-Compilation of Prepared Statements

Security

- Label-Based Access Control (LBAC)

Multiple Remote Standalone Secondary Servers

The high availability functionality currently available in IDS with HDR is extended with the capability to add multiple remote standalone (RS) secondary servers. This provides new high-availability configuration options that can be combined with HDR. RS secondary servers can be geographically distant from the primary server, serving as remote back up servers in disaster-recovery scenarios. Each RS

secondary server maintains a complete copy of the database, with updates transmitted asynchronously from the primary server over secure network connections. In the event of a failover from the primary server to the HDR server, one of the RS secondary servers can be promoted to become the HDR secondary.

This feature is primarily documented in the *Administrator's Guide* and the *Administrator's Reference*.

Multiple Shared Disk Secondary Servers

IDS now provides multiple server access to a single shared disk for high availability and query workload distribution. The shared disk (SD) solution for secondary servers provides configuration options that can be combined with remote standalone secondary servers and HDR. The primary server has write access to a disk or disk array, while all SD secondary servers have read-only access. An SD secondary server does not maintain a copy of the physical database on its own disk space; rather, it shares disks with the primary server. A single copy of data is shared among the servers, lowering data storage costs. New secondary servers can be added dynamically to share query workload and expand availability options. A secondary server can be promoted to the primary as needed to maintain continuous availability.

This feature is primarily documented in the *Administrator's Guide* and the *Administrator's Reference*.

Backup and Restore to Directories with **ontape**

You can use the **ontape** utility to backup and restore data from file system without interactive prompts. To enable this feature, set the TAPEDEV and LTAPEDEV configuration parameter to a valid directory of a local or remote mounted file system. The **ontape** utility generates file names automatically and performs physical and log backups.

This feature is documented in the *Backup and Restore Guide*.

Continuous Logical Log Restore

This feature lets you perform continuous restore of logical log backups using the **ontape** and ON-Bar utilities. Use continuous log restore to keep a second system (hot backup) available to replace the primary system if the primary system fails. Logical logs backed up on the primary system can be restored on the secondary system as they become available. If the primary system fails, the remaining available logical logs can be restored on the second system, which can then be brought online and function as the new primary system. You can suspend the log restore upon exit from the restore command and let the instance resume it with successive restore commands.

This feature is documented in the *Backup and Restore Guide*.

Encrypted Communications for HDR

You can encrypt communication between an HDR pair, to secure the transmission of data over unsecured networks, including the internet. You use new configuration parameters to enable encryption between the HDR servers and specify encryption options. After you enable encryption, the HDR primary database server encrypts the data before sending it to the secondary database server. The secondary database server decrypts the data. HDR encryption works in

conjunction with Enterprise Replication encryption and operates whether Enterprise Replication encryption is enabled or not.

This feature is documented in the *Administrator's Reference* and *Administrator's Guide*.

Improved Parallelism during Backup and Restore

This feature lets ON-Bar backup and restore a whole system using parallel I/O, which reduces the total time that is required to complete the backup or restore. Parallel whole system backups are still restorable to a consistent state without log backup and log restore.

This feature is documented in the *Backup and Restore Guide*.

Automatic Ordering of dbspaces during Backup and Restore

IDS now makes intelligent decisions regarding the ordering of dbspaces during backup and restore to achieve maximum parallelism, thus reducing the backup and restore time necessary. For example, if the largest dbspace is backed up in parallel to other smaller dbspaces, the complete system backup will take less time. During a restore, dbspaces are restored in the same order in which they were backed up, reducing restore time.

RTO Policy to Manage Server Restart

You can now create a recovery time objective (RTO) policy to set the amount of time, in seconds, that Informix Dynamic Server has to recover from a problem after you restart the server. You do this using a new configuration parameter, `RTO_SERVER_RESTART`. This configuration parameter enables failure recovery to meet the RTO policy by monitoring the workload and triggering checkpoints in a timely manner to ensure failure recovery will meet the policy.

This feature is documented in the *Administrator's Reference*, *Performance Guide*, and *Administrator's Guide*.

Non-blocking Checkpoints

Informix Dynamic Server has replaced its checkpoint algorithm with a virtually non-blocking checkpoint algorithm. Informix Dynamic Server now allows applications to continue to process transactions while checkpoint processing is occurring. Informix Dynamic Server monitors the workload and past checkpoint performance and triggers checkpoints more frequently to avoid running out of critical resources, like the physical or logical log, to make sure transactions do not experience blocking during checkpoint processing. For applications that are sensitive to response times, the old method of using aggressive LRU flushing to reduce checkpoint quiescent times can be changed. LRU flushing can be less aggressive since transaction processing is not blocked during checkpoint processing. Making LRU flushing less aggressive can improve transaction performance.

This feature is documented in the *Administrator's Reference*, *Performance Guide*, and *Administrator's Guide*.

Performance Improvements for Enterprise Replication

Enterprise Replication has increased the degree of parallelism when applying transactions on target servers, resulting in better performance.

ON-Bar Performance Report

This new feature provides a report of ON-Bar backup and restore performance. You can set the frequency, in minutes, of the progress messages with the `BAR_PROGRESS_FREQ` configuration parameter. You can configure the report to contain sub-second timestamps for ON-Bar processing, as well as the transfer rates between ON-Bar and the storage manager, and between ON-Bar and the IDS instance. You can set the level of reporting to be written to the ON-Bar Activity log by using the new `BAR_PERFORMANCE` configuration parameter.

This feature is documented in the *Backup and Restore Guide*.

Transform Data during Backup and Restore

This feature provides the option of specifying external programs, or filters, to transform data during backup and restore with both ON-Bar and ontape. You can use filters for compression or other data transformations. The backup filter reads the data to be backed up, transforms it, and then returns the transformed data to the backup utility. The restore filter receives the restored data from disk, transforms it back to its original state, and then passes the data to the restore utility. For example, if you want to compress your archive data, the backup filter receives the data, compresses it, and then backs up the compressed data. During the restore, the restore filter decompresses the data before it is restored to the database. You specify the filters with two new configuration parameters, `BACKUP_FILTER` and `RESTORE_FILTER`.

This feature is documented in the *Backup and Restore Guide*.

Improved Performance for Cooked Files with Direct I/O on UNIX

You can now improve the performance of cooked files used for dbspace chunks by using direct I/O. Informix Dynamic Server allows you to use either raw devices or cooked files for dbspace chunks. In general, cooked files are slower because of the additional overhead and buffering provided by the file system. Direct I/O bypasses the use of the file system buffers, and therefore is more efficient for reads and writes that go to disk. You specify direct I/O with the new `DIRECT_IO` configuration parameter. If your file system supports direct I/O for the page size used for the dbspace chunk and you use direct I/O, performance for cooked files can approach the performance of raw devices used for dbspace chunks.

This feature is primarily documented in the *Performance Guide* and the *Administrator's Guide*.

Improved Performance of Online Index Creation

The `CREATE INDEX ONLINE` statement now never places an exclusive lock over the source table for the final 10% of the index build.

If concurrent transactions apply changes to the table faster than online index creation can apply those same changes to the index, the server now automatically places a transient shared lock on the table from time to time to reduce the amount of concurrent insert, update, or delete activity in the table to allow the index build to catch up with new changes. Because it is a shared lock, read activity in the table is not affected. Consequently, online index creation no longer results in long transactions or running out of space to store records of the ongoing changes that need to be applied to the index.

SQL Administration API

A new SQL Administration API enables the DBSA to perform administrative tasks remotely by issuing SQL statements. The DBSA can accomplish administrative tasks, which in previous releases required administrative utilities of Informix Dynamic Server, by invoking new built-in **admin()** or **task()** functions with arguments that emulate command-line arguments to the corresponding utility. For example, the following SQL statement, which is equivalent to the **oncheck -ce** command, instructs the database server to check the extents:

```
EXECUTE FUNCTION admin('check extents');
```

The **admin()** and **task()** functions are equivalent, except that their return values have different data types. The return values indicate the result of the operation.

Some options to these functions can also accomplish tasks for which no corresponding utility exists. The effects of calling the **admin()** and **task()** functions with the same argument list are equivalent, but their return values, which indicate the result of the operation, have different data types. Information about the execution of Administration API functions is stored in the **command_history** table of a new **sysadmin** database of the Informix Dynamic Server instance.

This feature is primarily documented in the *Guide to SQL: Syntax, Administrator's Reference*, and *Administrator's Guide*.

Schedule Administrative Tasks

The new Scheduler allows you to manage and run scheduled maintenance, monitoring, and administration tasks at predefined times or as determined internally by the server. You can monitor activities (for example, space management) and create automatic corrective actions. Scheduler functions collect information and monitor and adjust the server, using an SQL-based administrative system and a set of tasks. A set of task properties, which define what needs to be collected or executed, control the Scheduler.

The task properties are stored in the **ph_task** table in the **sysadmin** database. Each row in this table is a separate task and each column is a task property. You can modify task properties, and you can set up new tasks by inserting rows into the table.

This feature is primarily documented in the *Guide to SQL: Syntax, Administrator's Reference*, and *Administrator's Guide*.

Monitor and Analyze Recent SQL Statements

You can now monitor the performance of recently executed SQL statements by configuring SQL statement tracing. This feature provides statistical information about each SQL statement executed on the system. The statistical information is stored in a circular buffer, which the DBA can resize. By default this feature is turned off. The feature can be enabled for all users or just a specific set of users.

This feature is primarily documented in the *Guide to SQL: Syntax, Administrator's Reference*, and *Administrator's Guide*.

Dynamically Change Enterprise Replication Configuration Parameters and Environment Variables

You can add, change, and remove in-memory values for Enterprise Replication configuration parameters and environment variables while the server is running. This feature does not update the onconfig file; the values are only valid for the current Enterprise Replication session.

This feature is documented in the *Enterprise Replication Guide*.

Dynamically Rename Enterprise Replication Columns, Tables, and Databases

You can now rename a replicated column, table, or database while Enterprise Replication is active. Use the RENAME statement to rename a column, table, or database on every participant in the replicate.

This feature is documented in the *Enterprise Replication Guide*.

Truncate Replicated Tables

You can now use the TRUNCATE statement on replicated tables while replication is active. The truncate operation is useful prior to direct synchronization with target tables are significantly inconsistent.

This feature is documented in the *Enterprise Replication Guide*.

Improved Statistics Maintenance

Informix Dynamic Server now automatically collects index statistics, equivalent to the statistics gathered by UPDATE STATISTICS in LOW mode, when you create a B-tree index on a UDT column of an existing table or if you create a functional or VII index on a column of existing table. Statistics that are collected automatically by this feature are available to the query optimizer, removing the necessity of manually running UPDATE STATISTICS. When B-tree indexes are created, column statistics are collected on the first index column, equivalent to what UPDATE STATISTICS generates in HIGH mode, with a resolution is 1% for tables of fewer than a million rows, and 0.5% for larger tables.

You can now view statistics about completed queries in the new Query statistics section in SET EXPLAIN.

In explicit UPDATE STATISTICS operations in MEDIUM mode, a new SAMPLING SIZE option in the Resolution clause can specify the minimum number of rows to sample for column distributions.

You can use the SET EXPLAIN statement to designate a non-default output file to capture the information that normally goes to the "sqexplain.out" file.

This feature is primarily documented in the *Guide to SQL: Syntax, Performance Guide*, and *Administrator's Reference*.

Installation Improvements on Windows Platforms

IBM Informix Dynamic Server installation on Windows platforms is now accomplished using an industry-standard installation tool, which provides a shorter and easier installation using a substantially smaller footprint.

This feature is documented in the *Installation Guide*.

This feature is documented in the *Quick Beginnings for IBM Informix Dynamic Server Express Edition*.

Session Configuration Routines

New built-in SPL procedures enable the Database Administrator to execute SQL and SPL statements automatically when a user connects to or disconnects from the database. These built-in procedures can be useful in setting the session environment or performing tasks like activating a role for users of information management applications whose code cannot easily be modified, or in automating operations that need to be performed after the application terminates.

If the DBA specifies the login ID of a user as the owner of a procedure whose name is **sysdbopen()**, Informix Dynamic Server executes that procedure when the specified user connects to or disconnects from the database. If the DBA specifies PUBLIC as the owner, that routine is automatically executed when a user who is not the owner of any of these built-in session configuration procedures connects to the database. The **sysdbopen()** routine is not invoked, however, when a user who is already connected to a database performs a distributed operation, such as a cross-database or cross-server query, that references an object in another database.

Similarly, another built-in procedure, **user.sysdbclose()** or **public.sysdbclose()**, if no **user.sysdbclose()** is registered in the database for that user, is called automatically when the user closes the connection to the database.

This feature is primarily documented in the *Guide to SQL: Syntax*, *Guide to SQL: Reference*, *Guide to SQL: Tutorial*, and *Administrator's Guide*.

Multiple Users for Administration Mode

The new administration mode enhances and replaces single-user mode as a way to temporarily restrict access to the database server to perform administrative tasks. Single-user mode allowed only the user **informix** or a member of the DBSA group to connect to the database server. The user **informix** or a DBSA can now dynamically give one or more specific users the ability to connect to the database server in administration mode. Administration mode is enabled using a new **onmode** command option, a new **oninit** command option, or the new ADMIN_MODE_USERS configuration parameter.

This feature is primarily documented in the *Administrator's Guide* and *Administrator's Reference*.

PHP-based OpenAdmin Tool for IDS

A new PHP-based Web browser administration tool, the OpenAdmin Tool for IDS, provides the ability to administer multiple database server instances from a single location.

OpenAdmin is an open-source program that you can download from the following Web site: https://www14.software.ibm.com/webapp/iwm/web/preLogin.do?lang=en_US&source=swg-informixfpd.

Named Parameters in a JDBC CallableStatement

A CallableStatement provides a way to call a stored procedure on the server from a Java program. This feature provides support for named parameters in a

CallableStatement, which was introduced in the JDBC 3.0 specification. Using named parameters in a CallableStatement adds the convenience of being able to identify parameters by name instead of by ordinal position. If the stored procedure is unique, you can omit parameters that have default values and you can enter the parameters in any order. Named parameters are especially useful for calling stored procedures that have many arguments and some of those arguments have default values.

This feature is documented in the *JDBC Driver Programmer's Guide*.

Index Binary Data Types

The new Binary UDT DataBlade module provides two new data types allow you to store binary-encoded strings, which can be indexed for quick retrieval. The binaryvar data type is a variable-length opaque type with a maximum length of 255 bytes. The binary18 data type is the same as the binaryvar data type except it holds a fixed value of 18 bytes. As part of a new DataBlade module, these data types come with string manipulation functions to validate the data types and bitwise operation functions that allow you to perform bitwise logical AND, OR, XOR, and NOT comparisons.

This feature is documented in the *Database Extensions User's Guide*.

Trigger Enhancements

Several new features expand the syntax and the functionality of triggers on tables and on views:

- You can now define multiple INSERT, DELETE, UPDATE, and SELECT triggers on a table and multiple INSTEAD OF triggers for the view.
- When a table, view, or column list has multiple triggers for a DML event type, Informix Dynamic Server executes all BEFORE triggered actions before the FOR EACH ROW actions, and executes all FOR EACH ROW actions before the AFTER actions.
- You can create SPL procedures which can refer to applicable OLD and NEW trigger correlated values. In this procedure you can access applicable OLD and NEW values and modify the NEW values: e.g. Using LET statements. From FOR EACH ROW trigger action, you can execute this SPL procedure [syntax: execute procedure foo() with trigger references].
- New Boolean operators (DELETING, INSERTING, SELECTING, and UPDATING) can be used in procedures executed from trigger action statements. These test whether the currently executing triggered action was triggered by the specified type of DML event and return a boolean value. The IF statement of SPL and the CASE expression of SQL can specify these operators as the condition in a trigger routine.

These features make it easier to incorporate Informix Dynamic Server triggers on tables and on views within a heterogeneous information management system where multiple applications need to share the table or view.

You can find examples about using multiple triggers in demo directory: %INFORMIXDIR%\demo\dbaccess\demo_ud\cr_trig.sql.

This feature is primarily documented in the *Guide to SQL: Syntax* and *Guide to SQL: Tutorial*.

Derived tables in the FROM Clause of Queries

The SELECT statement can now include syntax that complies with ISO/IEC 9075:1992, the SQL-92 standard, to specify full select subquery in the FROM clause as a data source for the query. These subqueries are called derived tables or table expressions, can be simple, UNION, or joined subqueries, including OUTER joins, and can include the ORDER BY clause. In addition, AS correlation specifications in the FROM clause can declare temporary names for columns within the query. Informix-extension syntax, such as the FUNCTION keyword with iterator functions or the TABLE (MULTISET (SELECT ...)) keywords for collection-derived tables, can now be replaced in the FROM clause by SQL-92 syntax. This feature expands the capability of Informix Dynamic Server to run without modification queries that are interoperable on other database servers that support industry-standard SQL syntax.

You can find examples about using derived tables in the demo directory:
`%INFORMIXDIR%\demo\dbaccess\demo_ud\sel_sql99.sql`.

This feature is primarily documented in the *Guide to SQL: Syntax* and *Guide to SQL: Tutorial*.

Index Self-Join Query Plans

In earlier Informix Dynamic Server versions, queries of tables with composite indexes perform inefficiently if the ratio of duplicate values to the number of distinct values is much higher for the leading columns than for subsequent columns of the index. A new feature of the query optimizer supports a new type of index scan, called an index self-join path, that uses only subsets of the full range of a composite index. The table is logically joined to itself, and the more selective non-leading index keys are applied as index bound filters to each unique combination of the leading key values. By default, the optimizer considers this type of scan.

The optimizer also supports two new access-method directives, INDEX_SJ and AVOID_INDEX_SJ. The INDEX_SJ directive forces an index self-join path using the specified index, or choosing the least costly index in a list of indexes, even if data distribution statistics are not available for the leading index key columns. The AVOID_INDEX_SJ directive prevents a self-join path for the specified index or indexes. This feature can improve performance for queries of tables on which composite indexes are defined.

This feature is primarily documented in the *Guide to SQL: Syntax* and *Performance Guide*.

Optimizer Directives in ANSI-Compliant Joined Queries

Earlier Informix Dynamic Server versions supported optimizer directives in Informix-extension joined queries, but not in queries that used ANSI/ISO syntax to specify joins. For both inline directives and external directives, this release extends support in ANSI/ISO joined queries to the following classes of optimizer directives:

- Access-method directives (FULL, AVOID_FULL, INDEX, AVOID_INDEX, INDEX_SJ, AVOID_INDEX_SJ)
- Explain-mode directives (EXPLAIN, AVOID_EXECUTE)
- Optimization-goal directives (ALL_ROWS, FIRST_ROWS).

The join-order directive (ORDERED) is supported only in ANSI/ISO-compliant LEFT OUTER joins and INNER joins. Because of ordering requirements for OUTER

joins, in ANSI-compliant joined queries that specify the RIGHT OUTER JOIN or FULL OUTER JOIN keywords, the ORDERED join-order directive is ignored, but it is listed under Directives Not Followed in the sqexplain.out file.

This feature does not support the join-method directives (USE_NL, AVOID_NL, USE_HASH, AVOID_HASH, /BUILD, and /PROBE) in ANSI/ISO joined queries, except in cases where the optimizer rewrites the query so that it no longer uses the ANSI/ISO syntax.

This feature is documented in the *Guide to SQL: Syntax*.

Deployment Wizard

The new Deployment Wizard allows you to perform a custom installation of Informix Dynamic Server by the components and features you do not currently wish to install. This enables database administrators and independent software vendors to minimize the disk space (footprint) required for a custom installation of IDS. All installation methods (console, GUI, and silent) use the wizard to enforce dependencies between components and provide estimated total footprint of selected components prior to actual file loading. You can selectively install or uninstall components at any time. You can use the generated response file to automate future replication of the installation setup in other installation locations.

This feature is documented in the *Installation Guide*.

This feature is documented in the *Quick Beginnings for IBM Informix Dynamic Server Express Edition*.

Enhanced Concurrency with Committed Read Isolation

In Committed Read isolation level, exclusive row-level locks held by other sessions can cause SQL operations to fail when attempting to read data in the locked rows. This release introduces a new LAST COMMITTED keyword option to the SET ISOLATION COMMITTED READ statement to reduce the risk of locking conflicts when attempting to read a table. This new syntax instructs Informix Dynamic Server to return the most recently committed version of the rows, even if another concurrent session holds an exclusive row-level lock.

This behavior can be extended to the Dirty Read, Read Uncommitted, and Read Committed isolation levels by setting the new USELASTCOMMITTED configuration parameter or through new options to the SET ENVIRONMENT statement.

This feature supports B-tree indexes and functional indexes, but not R-tree indexes. It does not support tables that are being accessed by DataBlade modules, tables with columns of collection data types, tables created using a Virtual Table Interface, tables with page-level locking, tables with exclusive table-level locks, unlogged tables, or tables in databases with no transaction logging.

This feature is primarily documented in the *Guide to SQL: Syntax*, *Guide to SQL: Reference*, *Administrator's Reference*, and *Administrator's Guide*.

Enhanced Data Types and UDR Support in Cross-Server Distributed Operations

This release extends support for UDRs in cross-database and cross-server distributed operations to most contexts where a UDR is valid in the local database.

In addition, external routines written in the C or Java languages are now valid in any distributed operation where an SPL routine is valid.

This release also extends the data types that are valid as parameters or return values of cross-server UDRs, which were formerly restricted to non-opaque built-in SQL data types, by supporting these additional data types:

- BOOLEAN
- LVARCHAR
- DISTINCT of non-opaque built-in types
- DISTINCT of BOOLEAN
- DISTINCT of LVARCHAR,
- DISTINCT of the DISTINCT types that are listed above

These data types can be returned by SPL, C, or Java language UDRs that use these data types as parameters or as return values, if the UDRs are defined in all the participating databases. Any implicit or explicit casts defined over these data types must be duplicated across all the participating IDS instances. The DISTINCT data types must have exactly the same data type hierarchy defined in all databases that participate in the distributed query.

This feature does not relax existing restrictions on other opaque and DISTINCT types or on large-object, serial, and collection data types in locally or remotely executed SPL routines or external routines.

This feature is primarily documented in the *Guide to SQL: Syntax*.

XML Publishing

You can now perform XML publishing with Informix Dynamic Server. Built-in functions let you transform results of an SQL query to XML for use in XML applications or in a heterogeneous database environment. Other built-in functions let you use XPATH expressions to extract elements and values from XML documents.

This feature is documented in the *XML User's Guide*.

Index Hierarchical Data

The node data type is part of the new Node DataBlade module which, with its supporting functions, gives you the ability to represent hierarchical data within the relational database. The advantage to this new data type is that it allows for searches within the hierarchy with a single SELECT statement using traditional operators without recursion. Represented as an ordinal number followed by either a single .0 or a set of ordinal numbers separated by dots, the node data type corresponds to a position in a tree structure similar to the way a books table of contents represents chapter, section, and subsection information.

This feature is primarily documented in the *Database Extensions User's Guide*.

Basic Text Search

The Basic Text Search DataBlade module allows you to search words and phrases in an unstructured document repository stored in a column of a table. The column can be a BLOB, CHAR, CLOB, LVARCHAR, NCHAR, NVARCHAR, or VARCHAR data type. Search strategies include single and multiple character wildcard

searches, fuzzy and proximity searches, and AND, OR and NOT Boolean operations. This feature is included with the database server at no extra cost.

This feature is primarily documented in the *Database Extensions User's Guide*.

Improved Concurrency with Private Memory Caches for Virtual Processors

You can now configure a private memory cache for every CPU virtual processor to decrease the time of server memory allocation on large multiprocessor computers by using the `VP_MEMORY_CACHE_KB` configuration parameter.

This feature is documented in the *Performance Guide* and the *Administrator's Reference*.

Web Feature Service for Geospatial Data

The new Web Feature Service DataBlade module implements an Open Geospatial Consortium Web Feature Service (OGC WFS) in Informix Dynamic Server to act as a presentation layer for the Spatial and Geodetic DataBlade modules. The OGC WFS interface allows requests for geographical features across the web using platform-independent calls. The XML-based GML (Geography Markup Language) is used as the encoding for transporting the geographic features.

This feature is compatible with the 8.21 version of the Spatial DataBlade module. See your release notes for the Geodetic DataBlade module for compatibility information.

This feature is primarily documented in the *Database Extensions User's Guide* and the *Spatial DataBlade Module User's Guide*.

Support for Data Server Clients with DRDA

You can use IBM Data Server JDBC Driver and the IBM Data Server .NET Provider to communicate with IDS as well as DB2. IDS now supports DRDA, the communications protocol used by DB2. As a result, application developers can create a solution using this API and enable their customers to deploy on the IBM data server they prefer to use.

DRDA is primarily documented in the *Administrator's Guide*.

Statement Labels, GOTO, and LOOP Statements in the SPL language

The new support for statement labels and the GOTO and LOOP statements provide greater flexibility in iterating and in exiting from statement loops in SPL routines. They also facilitate migration to Informix Dynamic Server of routines written in the procedural languages of other database servers that support GOTO and LOOP statement syntax. This release introduces the new SPL statements and constructs for statement loops in SPL routines:

- In the lexicographic sequence of statements in an SPL routine, a statement label can precede an executable statement, a program block, or a LOOP statement within the same SPL routine to which the GOTO label, END LOOP label, or EXIT label statement can transfer control.
- The GOTO label statement can unconditionally exit from a loop and transfer control to the executable statement or statement block that follows the specified statement label.

- The LOOP statement executes a statement block for an unspecified number of iterations. This can be within a WHILE condition LOOP or a FOR condition LOOP statement, or independent of any FOR or WHILE statement. LOOP statements can also be nested.
- To avoid infinite iterations of LOOP statements, new EXIT and CONTINUE options to the IF statement are valid within LOOP statements.
- The EXIT label WHEN condition statement can terminate a loop that has a label.
- The EXIT label statement can terminate a loop that has a label.

This feature is documented in the *Guide to SQL: Syntax*.

New SQL Functions

IDS now supports the following new built-in SQL functions to perform common mathematical, casting, and bitmap operations, and for manipulating character string, date, and datetime values:

- ADD_MONTHS()
- ASCII()
- BITAND()
- BITANDNOT()
- BITNOT()
- BITOR()
- BITXOR()
- CEIL()
- FLOOR()
- FORMAT_UNITS()
- LAST_DAY()
- LTRIM()
- MONTHS_BETWEEN ()
- NEXT_DAY ()
- NULLIF()
- POWER()
- ROUND()
- RTRIM()
- SYSDATE()
- TO_CHAR()
- TO_NUMBER()
- TRUNC()

These built-in SQL functions can simplify the migration to IDS of applications that have been developed for other database servers.

These functions are documented in the *Guide to SQL: Syntax*.

Automatic Re-Compilation of Prepared Statements

In this release, when you execute a prepared statement, IDS detects changes to underlying objects and re-prepares (recompiles) the statement if needed. After a statement has been prepared (compiled) by the application, the statement objects -- tables, indices, statistics -- can be altered. Some operations like ALTER TABLE require no cursor to be open, while others like CREATE INDEX ONLINE allow

cursors on the table. To account for these changes, the statement has to be re-prepared. In previous releases, when you tried to open a statement that's using an altered object, IDS raised -710 error so the client program can catch the exception, re-prepare the statement before proceeding.

After a DDL operation modifies the schema of a database table, the database server now automatically performs the following actions that previously had to be performed manually before executing any SPL routines or prepared objects that reference the modified table:

- The database server automatically issues the UPDATE STATISTICS statement to recalculate routine statistics for all SPL routines that reference the table.
- The database server automatically issues the PREPARE statement to update any prepared objects that reference the table.

Cursors associated with automatically updated routines or prepared statements can be used in dynamic SQL operations without the user manually issuing the PREPARE or UPDATE STATISTICS statement. For schema changes where automatic recompilation is not desired, the legacy behavior can be restored with the AUTO_REPREPARE configuration parameter or within a session by resetting a new SET ENVIRONMENT option.

This feature is primarily documented in the *Guide to SQL: Syntax* and the *Administrator's Reference*.

Label-Based Access Control

This feature implements Mandatory Access Control, a security requirement that the United States Federal government and public sector agencies in other countries impose on some information management systems that handle sensitive or classified information. This feature provides mechanisms by which a hierarchy of security labels can be defined and assigned to database objects and to users of the database. In a database that implements label-based access control (LBAC), there are two fundamental access rules:

- Users cannot have Read access to database objects that are at a higher security level than the user.
- Users cannot have Write access to database objects that are at a lower security level than the user.

This feature is supported by various enhancements to the SQL syntax of Informix Dynamic Server, and to the security of existing Informix Dynamic Server features, including secure auditing, high-availability data replication, backup and restore features, and administrative utilities:

- The database server administrator can grant a new built-in role, DBSECADM, to a database security administrator.
- The DBSECADM can issue new data definition language statements to create, drop, alter, and rename security policies, security labels, and security label components.
- The DBSECADM can grant exemptions to individual users to bypass LBAC access rules.
- Each untyped permanent database table can receive LBAC protection through a security label.
- A new built-in data type can store row labels in protected tables.
- The system catalog is enhanced to store LBAC information.

This feature is documented in the *Guide to SQL: Syntax*, *Guide to SQL: Reference*, and the *Security Guide*.

New Features in Version 10.00 of IBM Informix Dynamic Server

New Features in Version 10.00.xC4

IBM Informix Dynamic Server, Version 10.00.xC4 contains the following new features:

- TRUNCATE table support
- Enterprise Replication direct synchronization
- Enterprise Replication consistency checking
- Enhanced support for IPv6
- Secure local connections
- Secure DataBlade module paths
- Parallel backup and restore is more efficient
- DB-Access stops a process after the first error
- Informix Interface for TSM supports HP-UX (Itanium®)
- New default value for IFX_EXTEND_ROLE configuration parameter

TRUNCATE Table support

TRUNCATE is a SQL keyword that quickly deletes active rows from a table and the b-tree structures of its indexes. The active rows are deleted without dropping the table or its schema, access privileges, triggers, constraints, and other attributes. With TRUNCATE TABLE, you can de-populate a local table and release (or reuse for the same table) the storage space that formerly held its data rows and b-tree structures. For more information, see the *Guide to SQL: Syntax*.

Enterprise Replication Direct Synchronization

The `cdr sync replicate` and `cdr sync replicateset` commands perform direct synchronization between a reference server and one or more target servers. For more information, see the *Enterprise Replication Guide*.

Enterprise Replication Consistency Checking

The `cdr check replicate` and `cdr check replicateset` commands check for consistency within replicates, and optionally, repair inconsistent rows. For more information, see the *Enterprise Replication Guide*.

Enhanced Support for IPv6

IPv6 support now includes automatic discovery of whether or not a platform supports IPv6.

Secure Local Connections

The SECURITY_LOCALCONNECTION configuration parameter enables a database server administrator (DBSA) to set up security checking for local connections with the same host. For more information, see the *Administrator's Reference*.

Secure DataBlade Module Paths

The DB_LIBRARY_PATH configuration parameter allows you to specify a comma-separated list of valid directory prefix locations from which the database server can load external modules, such as DataBlade modules. You can use DB_LIBRARY_PATH to control the location from which shared objects can be loaded, and it allows you to enforce policies and standards on the formats of the

EXTERNAL NAME clause of the CREATE FUNCTION, CREATE PROCEDURE, and CREATE ROUTINE statements. For more information, see the *Administrator's Reference*.

DB-Access Stops a Process After the First Error

In DB-Access, using the **-a** command-line option will stop a process directly after the first error is encountered. Stopping a process from continuing after the first error ensures greater database consistency. For more information on DB-Access, see the *DB-Access User's Guide*.

Informix Interface for Tivoli Storage Manager Supports HP-UX (Itanium)

You can now use the ON-Bar utility with Tivoli® Storage Manager on the HP-UX (Itanium) platform. The Informix Interface for TSM supports HP-UX.

New Default Value for IFX_EXTEND_ROLE Configuration Parameter

The default value of the IFX_EXTEND_ROLE configuration parameter is changing from 0 to 1. The new default is more restrictive so that only administrators or users who have been granted extend role by administrators can register external routines.

New Features in Version 10.00.xC3

IBM Informix Dynamic Server, Version 10.00.xC3 contains the following new features:

- ANSI-joins in distributed queries
- Transaction support for XA-compliant external data sources
- MQ DataBlade module
- New DBCREATE_PERMISSION configuration parameter to restrict the ability to create databases
- New secure default directory for the DUMPPDIR configuration parameter
- Table-level restore for smart large object columns
- AES cipher support for network encryption
- New Enterprise Replication commands to show statistics information
- Client SDK included in Dynamic Server installation process
- Returning subsets of query results as collection-derived tables
- Ordering subsets of query results in collection-derived tables
- J/Foundation upgrade to JRE 1.4.2
- New default directory for ADTPATH configuration parameter
- New UNSECURE_ONSTAT configuration parameter

ANSI-Joins in Distributed Queries

Distributed queries that use ANSI-compliant JOIN syntax for specifying joined tables, execute more efficiently in Informix Dynamic Server 10.00.xC3 than in earlier releases, this is achieved by sending the remote joined tables as one query instead of multiple queries for each remote table, to each participating database server for operations on local tables of those servers. If you revert from Dynamic Server 10.00.xC3 to an earlier release that does not support this implementation of the ANSI-compliant syntax, such queries might show reduced performance, because the Dynamic Server instance from which the query originates performs the joins locally.

Transaction Support for XA-Compliant External Data Sources

The Dynamic Server Transaction Manager recognizes XA-compliant external data sources, which can participate in two-phase commit transactions. The transaction manager can invoke support routines for each XA-compliant external data source that participates in a distributed transaction at a particular transactional event, such as prepare, commit, or rollback. This interaction conforms to X/Open XA interface standards. For more information, see the *IBM Informix DataBlade API Programmer's Guide* and the *IBM Informix DataBlade API Function Reference*.

MQ DataBlade Module

The MQ DataBlade module provides mechanisms for data exchange between Dynamic Server and queues managed by IBM WebSphere MQ. For more information, see the *IBM Informix Built-In DataBlade Modules User's Guide*.

Restricting Database Creation

You can restrict which users can create Informix databases by setting the DBCREATE_PERMISSION configuration parameter in the ONCONFIG file. If you do not set DBCREATE_PERMISSION, all valid users can create databases. When you set DBCREATE_PERMISSION, only user **informix** and the users whom you specify are allowed to create databases. For more information, see the chapter on security in the *IBM Informix Dynamic Server Administrator's Guide* and the *Administrator's Reference* documentation notes.

Secure DUMPDIR Configuration Parameter Default Directory

The previous default value of the DUMPDIR configuration parameter was the **/tmp** directory. The new default value of the DUMPDIR configuration parameter in the **onconfig.std** file is the more secure **/usr/informix/tmp** on UNIX and **%INFORMIXDIR%\tmp** on Windows. If the DUMPDIR configuration parameter is not in the ONCONFIG file, then the **\$INFORMIXDIR/tmp** directory is used.

Table-Level Restore Support for Smart Large Objects

For Dynamic Server version 10.00.xC1, table-level restore supported all built-in, BOOLEAN, and LVARCHAR data types for physical and logical restore. For Dynamic Server version 10.00.xC3, table-level restore also supports smart large objects for physical restore only (restore from level-0 archive). For more information, see the *Backup and Restore Guide* documentation notes.

AES Ciphers for Network Encryption

The following AES ciphers are supported for network encryption:

aes AES 128bit key
aes128 AES 128bit key
aes192 AES 192bit key
aes256 aes 256bit key

Network encryption is described in the chapter on security in the *IBM Informix Dynamic Server Administrator's Guide*.

Enterprise Replication Statistics Commands

The **cdr stats rqm** command displays information about the reliable queue manager (RQM) queues used for Enterprise Replication. The **cdr stats recv** command displays receiver parallelism statistics and latency statistics by source node. For more information, see the *Enterprise Replication Guide* documentation notes.

CSDK Included in Dynamic Server Installation

You can install IBM Informix CSDK with the Dynamic Server installation program. CSDK version 2.90.xC3 is included as part of the Dynamic Server version 10.00.xC3 installation package. You can choose whether to install CSDK simultaneously with Dynamic Server or separately. For more information, see the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux*.

Enhanced Support for Retrieving Subsets of Query Results

New syntax enhancements support FIRST (and its new keyword synonym LIMIT) in the Projection clause, as well as the new SKIP keyword, which can exclude a specified number of the first qualifying rows before saving the result set as a collection-derived table. You can also use host variables or local SPL variables to specify the SKIP or FIRST parameters of queries in prepared objects. Before applying the FIRST and SKIP clause, the rows in the result set are numbered starting from 1. This feature provides greater flexibility in SELECT and INSERT statements, and in web applications whose pages display disjunct subsets of qualifying rows. For more information, see the *IBM Informix Guide to SQL: Syntax*.

Returning Subsets of Query Results as Collection-Derived Tables

Previous Dynamic Server releases did not support the ORDER BY clause in collection subqueries nor in conjunction with the FIRST keyword in the Projection clause of collection subquery. Syntax enhancements now support the ORDER BY clause in the FROM clause of a SELECT statements that defines a collection-derived tables. If the SELECT statement also includes any of the FIRST, LIMIT, or SKIP keywords in its Projection clause, the ORDER BY clause is applied to qualifying rows before the SKIP and FIRST specifications are applied to the result set. For more information, see the *IBM Informix Guide to SQL: Syntax*.

J/Foundation Upgraded to JRE 1.4.2

The J/Foundation component of Dynamic Server now includes JRE 1.4.2.

Secure ADTPATH Configuration Parameter Default Directory

The default value of the ADTPATH configuration parameter for Dynamic Server 10.00.xC3 and later on UNIX is the secure \$INFORMIXDIR/aaodir directory, instead of the /tmp directory.

UNSECURE_ONSTAT Configuration Parameter

The **onstat** commands that show SQL statement text being executed by a session are normally restricted to DBSA users. This restriction can be removed by setting the UNSECURE_ONSTAT configuration parameter to 1. For example, the **onstat -ses**, **onstat -stm**, **onstat -ssc**, and **onstat -sql** commands show SQL statement text.

onconfig.std value

not set

possible values

1

takes effect

when the database server is shut down and restarted

New Features in Version 10.00.xC1

IBM Informix Dynamic Server, Version 10.00.xC1 contains new features in the following areas:

- Security enhancements
- Server usability enhancements

- Performance enhancements
- SQL enhancements
- Enterprise Replication enhancements
- Backup and restore enhancements
- Storage enhancements
- Extensibility enhancements
- Installation enhancements
- Interoperability enhancements

Security Enhancements

IBM Informix Dynamic Server 10.0 provides significant advances in database server security, encryption, authentication, and availability.

Column-Level Encryption: You can use the new SQL statement, SET ENCRYPTION PASSWORD, to implement column-level encryption to improve the confidentiality of the data. New built-in functions provide methods for encrypting and decrypting data. The system catalog does not identify which columns contain encrypted data, and the same column can include both encrypted and unencrypted values. This security enhancement feature supports data confidentiality and data integrity.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Guide to SQL: Syntax*.

Server Utilities Check for Secure Environment Before Starting: This feature was first available with version 9.40.xC3. Server utilities on UNIX now check if the environment is secure by testing for the following conditions before starting:

- The permissions on **\$INFORMIXDIR** and some directories under it are correct. For each directory, check that the directory exists, is owned by user **informix** and the correct group, and that its permissions do not include write permissions for the group or other users.
- The permissions on the ONCONFIG file are correct. The file must belong to the DBSA group. If the DBSA group is group **informix** (default), then the ONCONFIG file should be owned by user **informix** too; otherwise, the ownership is not constrained. The file must not have write permissions for others.
- The permissions on the **sqlhosts** file are correct. Under a default configuration, the **sqlhosts** file is **\$INFORMIXDIR/etc/sqlhosts**; the owner should be user **informix**, the group should be either the **informix** group or the DBSA group, and there should be no public write permissions. If the file is specified by setting the **INFORMIXSQLHOSTS** environment variable, then the owner and group are not checked, but public write permissions are not permitted.
- The length of both the file specifications **\$INFORMIXDIR/etc/onconfig.std** and **\$INFORMIXDIR/etc/\$ONCONFIG** must each be less than 256 characters.

If the tests for any of these conditions fail, the utilities exit with an error message.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide*.

Restricting Registration of External Routines: The database server administrator (DBSA) can use the new built-in role called EXTEND to specify which users can register UDRs that include the EXTERNAL NAME clause. User-defined routines written in the C or Java languages use shared-object files that are external to the

database server and that could potentially contain harmful code. Similarly, Java UDRs are stored in class or jar files outside the DBMS. The DBSA can use the GRANT statement to confer the EXTEND role on a user (typically the DBA of a local database), or can use REVOKE to withdraw that role from a user. The DBSA can disable this feature by setting to off a new IFX_EXTEND_ROLE configuration parameter. This feature is intended to improve security and to control accessibility.

This feature is primarily documented in *IBM Informix Guide to SQL: Syntax*, *IBM Informix Dynamic Server Administrator's Guide*, and *IBM Informix Dynamic Server Administrator's Reference*.

Preventing Denial-of-Service Attacks: Dynamic Server provides multiple listener threads available to handle connections and imposes limits on the availability of the listener VP for incomplete connections. This feature reduces the risk of a hostile denial-of-service attack by making it more difficult to overwhelm the listener VP that handles connections. The default incomplete connection timeout period is reduced from 60 to 10 seconds. The default maximum number of incomplete connections is 1024.

You can customize this feature with the following two new configuration parameters:

- LISTEN_TIMEOUT. Sets the incomplete connection timeout period.
- MAX_INCOMPLETE_CONNECTION. Restricts the number of incomplete requests for connections.

You can change the value of these configuration parameters with the **onmode** utility while the database server is running.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Server Usability Enhancements

The server usability enhancements include features for ease of administration, scalability, and high availability.

Configuring Page Size: This feature adds support for specifying the page size for a standard or temporary dbspace when you create the dbspace. You might want to specify a non-default page size if you want a longer key length than is available for the default page size. The root dbspace is the default page size. If you want to specify a page size, the size must be an integral multiple of the default page size, not greater than 16 kilobytes.

You can also use the new BUFFERPOOL configuration parameter to create a buffer pool that corresponds to the page size of the dbspace.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Defining Buffer Pools: You can use the new BUFFERPOOL configuration parameter or the **onparams** utility to define a buffer pool for pages that correspond to each unique page size in use by your dbspaces. When you use the BUFFERPOOL configuration parameter or the **onparams** utility to define a buffer pool, you specify information about the buffer pool including its size, the number of LRUS in the buffer pool, the number of buffers in the buffer pool, and lru_min_dirty and lru_max_dirty values.

The BUFFERS, LRUS, LRU_MAX_DIRTY, and LRU_MIN_DIRTY configuration parameters are no longer used. 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 or the onparams utility. Information you enter using the BUFFERPOOL configuration parameter or onparams utility supersedes any information previously specified with the deprecated parameters.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Managing the tblspace `tblspace`: You have enhanced flexibility in managing the `tblspace tblspace`. The `tblspace tblspace` is a set of pages that describe the location and structure of all `tblspaces` in a given `dbspace`. Use the **onspaces** utility to move or drop the chunk containing the `tblspace tblspace`.

You can also specify the size of the first extent and of subsequent extents when `dbspaces` are created:

- Use the TBLTBLFIRST and TBLTBLNEXT configuration parameters to specify the size of the extents in the root `dbspace`. You must set these configuration parameters before creating the root `dbspace`.
- Use the onspaces utility to specify the size of non-root `dbspaces` during creation.

This feature allows you to reduce the number of `tblspace tblspace` extents and reduce the frequency of cases where those extents are placed in non-primary chunks.

These features are primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Administering the Database Server in Single-User Mode: As a Database Administrator, you can use a new single-user mode that is an intermediary mode between quiescent mode and online mode. While the server is in single-user mode, new connections are accepted only for user **informix**. Use this mode to perform any maintenance task, including tasks requiring the execution of SQL and DDL statements, when no other users are connected to the database server. Administrators can also perform all other functions available in online mode. You can set this mode using the **oninit** or **onmode** utilities, or the IBM Informix Server Administrator (ISA).

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Managing Database Permissions Through Default Roles: You can create a default role and assign that role to individual users or to PUBLIC on a per-database level. Each user who is assigned to a default role receives the privileges of that role as well as whatever other privileges are granted to the user individually. The syntax of the GRANT, REVOKE, and SET ROLE statements support this feature. This feature allows client applications to reset permissions (to the default role) of the user running the application, instead of requiring the DBA to reset permissions manually.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Guide to SQL: Syntax*.

Renaming Dbspaces: If you are user **informix** or have DBA privileges and the database server is in quiescent mode, you can rename a previously defined standard dbspace. You might want to rename standard dbspaces if you are reorganizing data in an existing dbspace and see a need to change the dbspace name. The rename dbspace operation only changes the dbspace name; it does not reorganize data. The database server automatically updates the system catalog so that database objects residing in a renamed dbspace are registered with the new name, but subsequent DDL statements referencing that dbspace must specify the new name, rather than the old name.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Specifying Which Events Trigger the Alarm Program: You can use the new alarm configuration parameter, **ALRM_ALL_EVENTS**, to specify whether the event alarm program operates for all events that are logged in the MSGPATH or only specified noteworthy events.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Specifying Shared Memory Size Greater Than 4 GB: You can now specify that segments for shared memory be created as large as your operating system platform or the SHMMAX parameter allows.

NOTE: ON-Monitor does not support Shared Memory Size greater than 4 GB. Therefore, the size specified for SHMVIRSIZE, SHMADD and SHMTOTAL in the "Shared-Memory screen" cannot be greater than 4 GB.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Setting Up HDR with External Backup and Restore: You can set up High-Availability Data Replication using standard ON-Bar or **ontape** commands for external backup and restore. Doing so can significantly reduce initial setup time.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Backup and Restore Guide*.

Replicating Indexes to HDR Secondary Servers: You can replicate an index to a secondary server in an HDR pair that contains a corrupted index. Replicating an index does not require a lock on the table. You can manually replicate an index when you detect a corrupt index with the **onmode -d index** command. Alternatively, you can configure automatic index replication with the **onmode -d idxauto** command or the DRIDXAUTO configuration parameter. This feature increases the availability of the HDR primary server because replicating an index is quicker than dropping and then rebuilding the index on the primary server.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Automating HDR Failover: You can automate switching servers for High-Availability Data Replication if the primary server fails by using the DRAUTO configuration parameter. 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.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Administrator's Reference*.

Determining Version Information: You can now use the new **-version** option with all server utilities to provide detailed information on the build operating system, build number, and build date. The **-version** option provides more information than the existing **-V** option. This feature enables DBAs and IBM Technical Support personnel to track version- and build-machine information, access documents before installing a product, be sure they are ready to install a product, and diagnose problems.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Reference*.

IBM Informix Server Administrator Enhancements: IBM Informix Server Administrator (ISA) supports the following new features:

- **Single-user mode.** See “Administering the Database Server in Single-User Mode” on page 2-48.
- **Enterprise Replication templates.** See “Replicate Templates” on page 2-52.

IPv6 Format for IP Addresses Support: You can use the IPv6 format for IP addresses with Dynamic Server. The IBM Informix JDBC Driver, Version 3.0, supporting the JDK 1.4, is IPv6 aware. That is, the code that parses the connection URL can handle the longer (128-bit mode) IPv6 addresses (as well as IPv4 format). This IP address can be an IPv6 literal.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide*.

Performance Enhancements

The performance enhancements include improved query performance and recovery time. In addition to the topics discussed below, enhancements have been made to improve performance in the following areas:

- XA transactions
- Nested ANSI-compliant left-outer joins
- Subqueries
- Full-outer joins

Allocating Memory for Non-PDQ Queries: This feature was first available in 9.40.xC4. You can specify how much memory is allocated to non-PDQ queries. The default of 128K can be insufficient for queries that specify ORDER BY, GROUP BY, hash joins, or other memory-intensive options. Use the new configuration parameter, DS_QUERY_MEM, to specify more memory than the 128K that is allocated to non-PDQ queries by default.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Guide* and *IBM Informix Dynamic Server Performance Guide*.

Storing and Applying External Optimizer Directives: You can create, save, and reuse external optimizer directives. In previous releases of Dynamic Server, external optimizer directives existed as text strings within DML statements, but were not stored as separate entities. External optimizer directives are useful when it is not feasible to rewrite a query for a short-term solution to a problem, for example, when a query starts to perform poorly. In this release, the DBA (as user **informix**) can create external optimizer directives and apply them to subsequent queries, without changing existing application code. This feature is implemented as a new SQL statement, **SAVE EXTERNAL DIRECTIVES**, to create and register external optimizer directives in a new **sysdirectives** table of the system catalog. Use the new **IFX_EXTDIRECTIVES** environment variable and the **EXT_DIRECTIVES** configuration parameter to enable this feature.

This feature is primarily documented in the *IBM Informix Dynamic Server Performance Guide*, *IBM Informix Dynamic Server Administrator's Reference*, and *IBM Informix Guide to SQL: Reference*.

Storing Multiple Table or Index Fragments in a Single Dbspace: You can store multiple fragments of the same table or index in a single dbspace, reducing the total number of dbspaces needed for a fragmented table. Each fragment is stored in a separate, named partition in the dbspace. Storing multiple table or index fragments in a single dbspace improves query performance over storing each fragmented expression in a different dbspace. This feature improves performance and simplifies management of dbspaces.

This feature is primarily documented in the *IBM Informix Dynamic Server Performance Guide*, *IBM Informix Dynamic Server Administrator's Guide*, and *IBM Informix Dynamic Server Administrator's Reference*.

Recovering Quickly with Fuzzy Checkpoints: You can set two new configuration parameters (**FAST_RESTART_PHYSLOG** and **FAST_RESTART_CKPT_FUZZYLOG**) to reduce the time required for database server recovery. This supports high availability by improving recovery performance when using fuzzy checkpoints.

This feature is primarily documented in the *IBM Informix Dynamic Server Administrator's Reference* and *IBM Informix Dynamic Server Performance Guide*.

Dynamically Setting the OPTCOMPIND Environment Variable: This feature was first available with Version 9.40.UC3. You can use the new **SET ENVIRONMENT OPTCOMPIND** statement to set the value of the **OPTCOMPIND** environment variable dynamically for the current session. For example, you might wish to change the value for different kinds of queries. For a DSS query, you should set the value of **OPTCOMPIND** to 2 or 1, and you should be sure the isolation level is not set to **REPEATABLE READ**. For an OLTP query, you can set the value to 0 or 1 with the isolation level not set to **REPEATABLE READ**. The value that you enter using this statement takes precedence over the current setting specified in the **ONCONFIG** file. The default setting of the **OPTCOMPIND** environment variable is restored when your current session terminates. No other user sessions are affected by **SET ENVIRONMENT OPTCOMPIND** statements that you execute.

This feature is primarily documented in the *IBM Informix Guide to SQL: Syntax* and *IBM Informix Dynamic Server Performance Guide*.

SQL Enhancements

The SQL enhancement improves database availability.

Creating and Dropping Indexes without Locking Tables: The SQL syntax of CREATE INDEX and DROP INDEX now supports the new ONLINE keyword. When you use the ONLINE keyword, DDL operations execute without applying an exclusive lock to the table on which the specified index is defined. If you use this syntax to create an index on a table that other users are accessing, the index is not available until no user is updating the table. After you issue the new syntax to drop an index, no one can reference the index, but concurrent DML operations can use the index until they terminate. Dropping the index is deferred until no user is using the index. This feature maintains the availability of the table within a production environment after an existing index has ceased to be efficient.

This feature is primarily documented in the *IBM Informix Guide to SQL: Syntax*.

Enterprise Replication Enhancements

The Enterprise Replication enhancements ease administration, improve data integrity, and allow additional SQL operations. These enhancements are documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Master Replicates: A master replicate is a replicate that uses a specified participant as a master against which all other participants are tested for consistency.

Creating a replicate as a master replicate provides several advantages:

- Ensures data integrity by verifying that all participants in the replicate have table and replicated column attributes that match the master replicate definition. Verification occurs when the replicate is defined and when the replicated is started, thus avoiding runtime errors.
- Provides automatic table generation on participants that do not already contain the table specified in the master replicate.
- Allows alter operations on the replicated tables. For more information, see “Altering Replicated Tables” on page 2-53.

Replicate Templates: Enterprise Replication has a replicate template option that greatly simplifies setting up the replication system. Replication templates contain schema information about a database, a group of tables, column attributes, and the primary keys that identify rows. You first define the template by specifying schema information and then apply the template on the database servers within the replication domain. If the tables and primary keys do not exist on the server during the realization of the template, then the tables and primary keys are created according to the template definitions. Table attributes are verified during the realization of the template to ensure that proper replication can be performed on that database. Replicates created as part of the realization of a replicate template are created as mastered replicates and grouped as a replicate set.

You can create, realize, view, and delete templates using the Enterprise Replication command line syntax or IBM Informix Server Administrator (ISA).

Performing Synchronization and Repair: Enterprise Replication can perform an initial synchronization on data to be replicated as well as repair a system in which data may have failed to be applied for any reason. Initial synchronization is performed at the startup of a new replicate or when a new participant is added to an existing replicate. Repair jobs reconciles the differences when the normal replication process has failed. Repair processes can be optimized to only compare rows found in ATS or RIS files: if the row still exists in the source, it is re-replicated; if it does not, then it is deleted on the target.

Altering Replicated Tables: Beginning with version 10.00 you can alter a replicated table in any of these ways:

- Add or drop UNIQUE, DISTINCT or FOREIGN KEY constraints
- Alter next extent size
- Alter the locking granularity of a table.
- Add or drop default values
- Add or drop SQL checks
- Alter serial columns
- Add or drop fragments (new in 10.00)
- Attach or detach fragments (new in 10.00)
- Add, modify, or drop columns (new in 10.00)
- Create a clustered index or recluster an existing index (new in 10.00)

Before attaching fragments place the table in alter mode. Alter mode is a new state of a replicated table. In this mode only DDL and select operations are allowed. No insert, update, or delete operations are allowed in alter mode. After attaching the fragments, unset alter mode.

When performing any sort of alter other than attaching fragments, the database server automatically sets alter mode before altering the table and unsets it after the table is altered.

To set/unset the alter mode on a table, use the CDR CLI interface.

ALTER TABLE and ALTER FRAGMENT statements are allowed only on master replicates.

Remastering: An existing replicate can be redefined by “remastering the replicate”. Remastering allows you to change what columns are in a replicate definition without interrupting the replication. Also an existing non-master replicate can be converted to a master replicate using remastering.

Detecting Event Alarms with the Event Alarm Program: Enterprise Replication event alarms are now detected uniquely by the event alarm program. You can specify what actions are triggered by specific Enterprise Replication event alarms.

Suppressing DataSync Warning and Error Messages: You can selectively suppress any DataSync warning or error message that the ATS or RIS files contain with the CDR_SUPPRESS_ATSRISWARN configuration parameter.

Backup and Restore Enhancements

The backup and restore enhancements improve performance and debugging. These enhancements are documented in the *IBM Informix Backup and Restore Guide*.

Performing Point-In-Time Restores of Tables with Archecker: You can recover specific tables from an archive using the **archecker** utility. Tables can be restored up to a specific point in time. This allows you to restore specific pieces of data without having to perform a lengthy restore of the entire archive. Data can be restored without restoring the full database server on another instance. To perform a table level restore, **archecker** supports a command file which uses an SQL-like syntax to specify the source and destination table schemas.

View Logical Logs Backed Up by ON-Bar: You can now view the logical-logs backed up by ON-Bar, similar to using the **onlog** utility to view the logical-logs

backed up by the **ontape** utility. If you need to perform a restore, you can view the old logical-log files that were backed up by ON-Bar to find out the exact problem.

Enhanced Debugging for ON-Bar: You can now change the ON-Bar debugging level while ON-Bar is running. You can save a large amount of time and disk space by setting high debugging levels only when you need them. You set the debugging level with the `BAR_DEBUG` configuration parameter. The value of `BAR_DEBUG` ranges from 0-9, with 0 being no debugging information and 9 being the most detailed debugging information. You can change the value of `BAR_DEBUG` as frequently as you want. Similar to the **onstat -m** command, you can use ON-Bar to print the recent ON-Bar activity from the **onbar** activity log file.

The ontape Utility Can Use Standard I/O: You can now specify that **ontape** uses standard I/O instead of a tape device or disk file. During a backup, **ontape** writes the data to stdout (standard output). During a restore, **ontape** reads data from stdin (standard input). Specifying stdout or stdin allows **ontape** to use pipes (an OS provided buffer mechanism to connect separate programs to a data stream) for archives and restores. Using pipes, the data can be processed by other programs without requiring that the data be saved in files or tape devices. For example, you can use compression to save media space, use cloning to duplicate the archive for safety reasons, or restore the data onto another server instance. This feature is especially efficient for setting up High-Data Availability Replication by restoring the data to the secondary server while skipping the intermediary step of saving the data to a file or disk.

External Backup and Restore Using ontape: You can use the **ontape** utility to perform external backup and restore procedures.

Storage Enhancements

The storage enhancements improve ease of use.

Using Long Identifiers with the High-Performance Loader: The High-Performance Loader utilities **onpload** and **onpladm** include support for long object names up to 128 characters, but the **ipload** GUI does not. If you use long database, table or column names and create jobs using **onpladm**, you cannot run these jobs using **ipload**. For **ipload**, database, table and column names cannot exceed 18 characters.

This feature is documented in the documentation notes for the *IBM Informix High-Performance Loader User's Guide*.

Informix Interface for Tivoli Storage Manager: The Informix Interface for Tivoli Storage Manager (formerly known as Tivoli Data Protection for Informix) is bundled with the Dynamic Server installation. This feature eases installation. You can use Informix Interface for TSM with ON-Bar to store data. Informix Interface for TSM stores Dynamic Server databases and logical logs on the Tivoli Storage Manager.

Extensibility Enhancements

The extensibility enhancements improve distributed transactions, obtaining information from trigger executions, and Java support.

Manipulating Built-in Opaque Data Types In External Tables: You can use built-in opaque data types in remote queries involving databases residing on the same database server. The built-in opaque data types supported by this feature are `BOOLEAN`, `BLOB`, `CLOB`, and `LVARCHAR`. User-defined types that can be cast explicitly to a built-in data type are also supported. You can use these data types in

DML operations and as parameters and returned data types of UDRs between all databases of the same Dynamic Server instance. The target data type of the explicit cast must be a built-in opaque type or other built-in SQL data type, and all the casts and all the UDTs must be defined in all of the databases participating in the query. The following types of operations are supported:

- DML operations such as SELECT, UPDATE, DELETE, and INSERT on tables across databases having built-in opaque data type columns. The table can be a table, view, or synonym.
- DDL operations such as CREATE VIEW and CREATE SYNONYM at the local database on cross-database tables having built-in opaque data type columns. The view or synonym can be created only at the local database.
- Implicit and explicit execution of user-defined routines (written in SPL, C, or Java) with built-in opaque data type parameters and return types. This is applicable for both functions and procedures.

Obtaining Information From Trigger Executions: You can create user-defined routines that are invoked in trigger action statements to obtain information about the triggers, triggering tables, views, statements, and the values of rows involved in the trigger actions. Using the new DataBlade API routines, you can write a general purpose user-defined routine that can you can use to audit any table and any trigger event.

This feature is documented in the *IBM Informix DataBlade API User's Guide*.

Support for JRE Version 1.4: J/Foundation supports JRE Version 1.4 and the JDBC 3.0 specification.

Installation Enhancements

The installation enhancements improve usability. These enhancements are documented in the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux*.

IBM Software Electronic Licensing: Dynamic Server, CSDK, JDBC, and ISA installation programs now include the display and required acceptance of a License Agreement. The License Agreement and License Information files are installed with the product and available for customer review at any time.

New doc Directory Before Installation: The following files are now available in a /doc directory that is available before installing Dynamic Server:

- Release notes
- Machine notes
- Documentation notes
- *IBM Informix Dynamic Server Installation Guide* (in PDF format)

New installation program on UNIX and Linux

You can install Dynamic Server on UNIX or Linux using the new installation application in console, GUI, or silent mode. For detailed instructions, see the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux*.

Interoperability Enhancements

The interoperability enhancements improve communication between Informix and DB2 products.

Running Informix ESQL/C Applications with DB2: You can run Informix ESQL/C applications with DB2 servers and databases. The Informix ESQL/C

product provides new libraries that are called when you use the **esql** command to preprocess your files to work with DB2. Informix ESQL/C runs with DB2 Version 8.2, or later, running on Linux, UNIX, and Windows operating systems.

New Features in Version 9.4

The new features for Dynamic Server, Version 9.4, fall into the following major areas:

- Security Enhancement
- Database Server Usability Enhancements
- Performance Enhancements
- Enterprise Replication Enhancements
- Extensibility Enhancements
- SQL Enhancements
- GLS Enhancements
- Reliability, Availability, and Supportability Features
- DataBlade API Enhancements
- High-Performance Loader Enhancements
- Backup and Restore Enhancements
- Installation Enhancements

The *IBM Informix Migration Guide* lists all new environment variables, configuration parameters, system-monitoring interface (SMI) tables in the **sysmaster** database, system catalog tables, and reserved SQL keywords in Version 9.4.

Security Enhancement

Version 9.4 of Dynamic Server supports encrypting data transmissions over the network using the encryption communication support module (ENCCSM).

This option provides complete data encryption with a standard cryptography library, with many configurable options. A message authentication code (MAC) will be transmitted as part of the encrypted data transmission to ensure data integrity. A MAC is an encrypted message digest.

The encryption algorithms use openssl 0.9.6 as the code base.

Distributed queries can also be encrypted.

For more information on encryption, see the *IBM Informix Dynamic Server Administrator's Guide*.

Enterprise Replication implements encryption with configuration parameters instead of the ENCCSM. For more information, see "Enterprise Replication Security" on page 2-60.

Database Server Usability Enhancements

Version 9.4 of Dynamic Server supports the following usability enhancements.

Increase Size of Chunks, Chunk Offsets, and Number of Allowable Chunks

Chunks and chunk offsets now have a limit of 4 TB (2^{42} bytes) in size. The previous limit was 2 GB (2^{31} bytes). The number of chunks per database server is

now 32,766. The previous limit was 2,047. These features are enabled by setting large chunk mode with the **onmode** utility.

For information on these new limits, see the *IBM Informix Dynamic Server Administrator's Guide*. For information on how to enable large chunk mode, see the *IBM Informix Administrator's Reference*.

Configurable Event Alarms

You can now configure event alarms with a modifiable shell script, **alarmprogram.sh**.

Set the ALARMPROGRAM configuration parameter to alarmprogram.sh, and edit the file to specify the email address of the database administrator, the email address of the pager service, the mail utility, and whether to automatically backup logical logs.

For more information on event alarm parameters, see the *IBM Informix Dynamic Server Administrator's Guide*. For more information on setting event alarms, see the *IBM Informix Dynamic Server Administrator's Reference*.

Increased Database Server Aliases

You can now specify up to 32 database server aliases with the DBSERVERALIASES configuration parameter.

For more information, see the *IBM Informix Dynamic Server Administrator's Reference*.

Increased File Size Limit

The new file size limit is 4 TB. This limit applies to all database server utilities, including the following:

- The UNLOAD and LOAD statements of SQL (see “LOAD TO and UNLOAD FROM with Large Files” on page 2-64)
- The **onspaces** utility
- The **ontape** utility (see “Full Use of Storage Media and Increased File Size Limit” on page 2-69)
- The shared-memory dump file
- The **dbimport** and **dbexport** utilities
- DataBlade API stream support functions

The previous file size limit was 2 GB. (Logical log files, however, must not exceed 1 GB in size for Version 9.4.)

For more information on **dbimport** and **dbexport**, see the *IBM Informix Migration Guide*.

Full Use of Storage Media

Utilities that use storage media backup and restore or loading and unloading data can use the full size of the storage media. This feature is supported by the following utilities:

- The **ontape** utility (see “Full Use of Storage Media and Increased File Size Limit” on page 2-69)
- The **onload** and **onunload** utilities
- The **dbimport** and **dbexport** utilities
- High-Performance Loader utilities: **ipload**, **onpload**, and **onpladm**

Except for the High-Performance Loader utilities, you use this option by setting the tape size to 0. For information about using this feature with HPL utilities, see “High-Performance Loader Enhancements” on page 2-68.

In previous releases, the user was required to specify a non-zero tapesize value when using these utilities, and risked wasting storage space. The previous limit was 2 GB per storage device.

For more information on the **onload**, **onunload**, **dbimport**, and **dbexport** utilities, see the *IBM Informix Migration Guide*.

Increased Default Values for Tape Block Size Configuration Parameters

The default tape block size for the TAPEBLK and LTAPEBLK configuration parameters in the **onconfig.std** file has been increased to 32 kilobytes in Version 9.4. Here TAPEBLK specifies the block size for tapes used in storage-space backups, and LTAPEBLK specifies the block size for tapes used in logical-log backups.

The default value for TAPEBLK and for LTAPEBLK in earlier releases was 16 kilobytes.

For more information, see the *IBM Informix Dynamic Server Administrator's Reference*.

Chunk Reserve Pages in Non-Root Chunks

Chunk reserve pages are stored in the root chunk. In previous releases of Dynamic Server, you could not add chunks if the root chunk was full. However, for Version 9.4, if you add chunks when the root chunk is full, the new chunk metadata is stored in extended chunk reserve pages allocated from non-root chunks in the root dbspace.

For more information, see the *IBM Informix Dynamic Server Administrator's Guide*.

Restartable Fast Recovery

Restartable fast recovery allows physical logging during the roll forward phase to prevent fast recovery failure. If the physical log overflows during fast recovery, the physical log is extended to a disk file, named **plog_extend.server_number**. The location of this file is set by the new PLOG_OVERFLOW_PATH configuration parameter. This file is removed after the first checkpoint during fast recovery.

For more information on fast recovery, see the *IBM Informix Dynamic Server Administrator's Guide*. For more information on the PLOG_OVERFLOW_PATH configuration parameter, see the *IBM Informix Dynamic Server Administrator's Reference*.

Microsoft Transaction Server/XA Support

Transaction managers coordinate distributed queries between Informix and non-Informix databases. Informix supports XA transactions in a tightly-coupled mode, which allows you to use Microsoft Transaction Server (MTS/XA) as a transaction manager. You can use MTS/XA with IBM Informix ODBC Driver.

For information on how to monitor transactions with **onstat -x**, see the *IBM Informix Administrator's Guide* and the *IBM Informix Dynamic Server Performance Guide*. For information on MTS/XA, see the MTS/XA documentation.

Performance Enhancements

The following new features are designed to improve the performance of Dynamic Server.

PDQ is Enabled for Hold Cursors

Cursors created with the WITH HOLD keywords can now be processed in parallel.

For more information on how this feature can affect performance, see the *IBM Informix Dynamic Server Performance Guide*. For more information on the syntax of this feature, see the section on DECLARE in the *IBM Informix Guide to SQL: Syntax*.

Improved Transaction Processing with the B-tree Scanner

The new 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, based on a priority list.

For more information on how this feature can affect performance, see the *IBM Informix Dynamic Server Performance Guide*. For information on how to configure the B-tree scanner with the **onstat -C** command, see the *IBM Informix Dynamic Server Administrator's Reference*.

Improved Priority Management for the Buffer Manager

Buffers are now divided into two classes: HIGH priority for frequently accessed buffers, and LOW priority for infrequently accessed buffers. Priority classification is dynamic, based on observed access frequencies of the buffers. The CPU usage of the buffer manager is reduced, thus improving performance.

For more information, see the *IBM Informix Dynamic Server Performance Guide*.

Spatial Query Costing

You can provide cost and selectivity functions for R-tree indexes to allow the optimizer to accurately choose the appropriate index to use for a particular query.

For more information, see the *IBM Informix R-Tree Index User's Guide*.

More Precise LRU Maximum and Minimum Settings

The LRU_MAX_DIRTY and LRU_MIN_DIRTY configuration parameters can take a FLOAT type value, thereby increasing the precision of buffer clean to two positions to the right of the decimal point.

For more information on how these configuration parameters affect performance, see the *IBM Informix Dynamic Server Performance Guide*. For more information on setting these configuration parameters, see the *IBM Informix Dynamic Server Administrator's Reference*.

Enterprise Replication Enhancements

The following new features enhance the extensibility, usability, or performance of the Enterprise Replication facility of Dynamic Server.

All Enterprise Replication features are documented in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Enterprise Replication Security

Enterprise Replication supports the same levels of network encryption available for client/server communications. Encryption is implemented in Enterprise Replication with the encryption configuration parameters listed in “New and Altered Configuration Parameters” on page 2-60.

Support for ROW and Collection Data types

Enterprise Replication can now replicate the following data types:

- Named and unnamed ROW data types
- Collection data types: LIST, MULTiset, and SET

Faster Queue Recovery

The addition of a table with replicate information to the transaction records and row data tables reduces transaction processing time.

Replication During Queue Recovery

Users can connect to a database server during queue recovery; transactions are added to the queue. However, if the volume of transactions during queue recovery is so large that the logical log is in danger of being overwritten, replication is blocked.

Large Transactions Support

Enterprise Replication automatically spools large transactions to disk instead of holding them in memory. Rows from spooled transactions are paged in and out of memory as needed. Enterprise Replication can replicate transactions up to 4 TB in size.

Improved Availability with HDR

You can use High-Availability Data Replication (HDR) on critical database servers in an Enterprise Replication system to provide identical backup database servers. (Dynamic Server releases earlier than Version 9.4 could support either Enterprise Replication or HDR, but both could not run concurrently.)

Dynamic Log File

Enterprise Replication can request the database server to add a new dynamic log file if replication enters DDRBLOCK mode.

The new CDR_MAX_DYNAMIC_LOGS configuration parameter specifies the number of dynamic log file requests that Enterprise Replication can make in one server session.

New Commands

The new **brief** option for the **cdr list replicate** command displays a summary of participants for all replicates

The new **cdr remove** command removes Enterprise Replication from an HDR server.

New and Altered Configuration Parameters

Enterprise Replication has the following new configuration parameters:

- CDR_DBSPACE specifies the dbspace of the **syscdr** table.
- CDR_ENV sets Enterprise Replication environment variables.
- CDR_MAX_DYNAMIC_LOGS specifies the number of dynamic log file requests that Enterprise Replication can make in one server session.
- ENCRYPT_CDR to enable and set the level of network encryption.

- ENCRYPT_CIPHERS to specify the ciphers to use for encryption.
- ENCRYPT_MAC to specify the level of message authentication coding to use.
- ENCRYPT_MACFILE to specify MAC key files.
- ENCRYPT_SWITCH to define the frequency at which ciphers and secret keys are renegotiated.

The CDR_QDATA_SBSPACE configuration parameter now allows you to specify up to 32 sbspaces for Enterprise Replication to use for storing spooled row data.

The CDR_QDATA_SBFLAGS configuration parameter is deprecated.

New Environment Variables

The **CDR_LOGDELTA** environment variable determines when the send and receive queues are spooled to disk.

The **CDR_PERFLOG** environment variable enables queue tracing.

The **CDR_ROUTER** environment variable disables intermediate acknowledgements of transactions in hierarchical topologies.

The **CDR_RMSCALEFACT** environment variable sets the number of DataSync threads started for each CPU VP.

Extensibility Enhancements

The following new features are designed to improve the extensibility of Dynamic Server.

Enhanced HDR Support for Extensibility Features

High-Availability Data Replication (HDR) now supports replication of the following extended objects:

- All built-in and extended data types.
- User-defined routines.
- R-tree and functional indexes
- TimeSeries DataBlade module

User-defined data types (UDTs) must be logged and must reside in a single database server. Data types with out-of-row data are replicated if the data are stored in an sbspace or in a different table on the same database server.

HDR does not replicate data stored in operating system files nor in persistent (that is, non-temporary) external files. HDR also does not replicate memory objects that are associated with user-defined routines.

To use user-defined data types, user-defined routines, or DataBlade modules with HDR, you must install the user-defined data types, user-defined routines, or DataBlade modules on both the HDR primary and secondary database servers. Register the user-defined data types, user-defined routines, or DataBlade modules only on the HDR primary database server.

For more information see the *IBM Informix Dynamic Server Administrator's Guide*.

Using an Iterator Function in the FROM Clause of a SELECT Statement

An iterator function can now be specified in the FROM clause of the SELECT statement. (An iterator function is a user-defined function that returns to its calling context more than once, each time returning a value.)

You can query the returned result set of an iterator UDR using a virtual table-interface. You can then manipulate the iterator result set in a number of ways, such as by using the WHERE clause to filter the result set; by joining the UDR result set with other table scans; by running GROUP BY, aggregation, and ORDER BY operations; and so on.

For information on writing iterators, see the *IBM Informix User-Defined Routines and Data Types Developer's Guide*. For information on using iterators in the FROM clause of a SELECT statement syntax, see the *IBM Informix Guide to SQL: Syntax*.

Enhanced CREATE FUNCTION and CREATE PROCEDURE Syntax

Several new features improve the functionality of user-defined functions.

Multiple SLVs in the WHERE Clause of SELECT, UPDATE, and INSERT Statements:

Because a user-defined function now can return more than one OUT parameter, DML (data manipulation language) statements that use the returned values from function calls as statement-local variables (SLVs) within queries or subqueries can now support multiple SLVs.

For more information on OUT parameters, see “Multiple OUT Parameters” on page 2-64.

For more information on SLVs, see the *IBM Informix Guide to SQL: Syntax*.

Declaring Names for Returned Values of an SPL UDR: Releases of Dynamic Server earlier than Version 9.4 support user-defined functions written in the SPL language that return one or more values of specified data types. In this release, the RETURNS (or RETURNING clause) of an SPL function can also declare a name for each returned value. This feature can make it easier for SPL functions to pass column headings to SELECT statements.

For more information, see the *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

SQL Enhancements

Besides the enhancements that are described in section “Extensibility Enhancements” on page 2-61, the following additional changes to the IBM Informix dialect of the Structured Query Language (SQL) have been implemented in Version 9.4 of Dynamic Server.

INSTEAD OF Triggers on Views

The CREATE TRIGGER statement has been enhanced to support INSTEAD OF triggers on views. You can define an INSERT, UPDATE, or DELETE event on a specified view that activates the trigger. Rather than directly performing the triggering DML event, the database server executes the Action clause of the INSTEAD OF trigger. This feature provides a mechanism for updating the underlying tables of views that include columns from more than one table; such views were not updatable in earlier releases of Dynamic Server.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

Enhanced SELECT Statement Syntax

The syntax rules for the SELECT statement have been enhanced.

For more information on these features, see the *IBM Informix Guide to SQL: Syntax*.

Ordering by Columns or Expressions Not in Projection List: The ORDER BY clause now can include column names or expressions that do not appear in the select list of the projection clause. The following query, for example, is now valid:

```
SELECT stock_num, manu_code FROM stock ORDER BY unit_price
```

Earlier releases had required that **unit_price** also appear in the Projection clause.

Iterator UDRs in the FROM Clause: As noted in section “Using an Iterator Function in the FROM Clause of a SELECT Statement” on page 2-62, iterator functions are now valid in the FROM clause of the SELECT statement.

Functional Indexes on More Than 16 Columns

Functional indexes are UDRs that accept column names as arguments, and whose return values are specified as index keys in the CREATE INDEX statement. In previous Dynamic Server releases, the number of columns was restricted to no more than 16.

In Version 9.4, however, the number of columns that can be arguments to a functional index is language-dependent. For UDRs written in the C language, a functional index can have up to 102 key parts. A functional index defined in the SPL or Java languages can have up to 341 key parts.

For more information, see the *IBM Informix Guide to SQL: Reference*.

Enhanced Dynamic Query Support

The DESCRIBE statement now recognizes the OUTPUT keyword. The new dynamic SQL statement, DESCRIBE INPUT, can provide information about the retrieved columns and dynamic parameters of prepared DML statements.

For more information on these features, see the *IBM Informix Guide to SQL: Syntax*.

The DESCRIBE INPUT Statement: The DESCRIBE statement in previous releases of Dynamic Server could not provide information about input parameters of the WHERE clause of prepared INSERT or SELECT statements. It could provide limited support for UPDATE parameters if the **IFX_UPDDESC** environment variable were set. In this release, you can specify the INPUT keyword in the DESCRIBE statement to return information about each input parameter of a prepared DML statement, including the data type, identifier, and length (in bytes).

The DESCRIBE OUTPUT Statement: The client system that executed a dynamic SQL application can use the DESCRIBE OUTPUT statement (or simply DESCRIBE, because the OUTPUT keyword is optional) to obtain information about the output parameters of a prepared DML statement. (This is a CSDK feature, but it requires information that the database server did not make available to the client application in releases earlier than Version 9.4.)

Session-Level Non-Default Collation

In previous Dynamic Server releases, the database server sorted NCHAR and NVARCHAR values according to the localized collating sequence of the locale that

the **DB_LOCALE** environment variable specified, if that locale defined a **COLLATION**; otherwise, all sorting operations followed the code set order.

In this release, the new **SET COLLATION** statement can specify the localized collation of another locale. For the rest of the session (or until the next **SET COLLATION** statement in the same session), sorting of **NCHAR** and **NVARCHAR** values ignores the **DB_LOCALE** setting. You can restore the default collating order by issuing the **SET NO COLLATION** statement. This feature enables the database server to use different localized collating orders on **NCHAR** and **NVARCHAR** data sets within a single database, if both collating orders can operate on the same character set.

Database objects (such as indexes, check constraints, and triggers) that perform collation use the collating order that was in effect when the object was created, rather than the order that is in effect at runtime, if these two collating orders are not the same.

For more information on the **SET COLLATION** statement, see the *IBM Informix Guide to SQL: Syntax*. For more information on the **DB_LOCALE** environment variable, see the *IBM Informix GLS User's Guide*. For more information on the **NCHAR** and **NVARCHAR** data types, see the *IBM Informix Guide to SQL: Reference*.

LOAD TO and UNLOAD FROM with Large Files

The **LOAD** and **UNLOAD** statements were previously restricted on most platforms to files no larger than 2 GB for **LOAD** and **UNLOAD** flat-file I/O operations. This restriction has been relaxed to 4 TB in Version 9.4.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

SET Residency Statements No Longer Needed

In Dynamic Server releases earlier than Version 9.4, the **SET TABLE** and **SET INDEX** statements could specify whether one or more fragments of a table or of an index remain in a shared memory buffer, rather than be written to disk. These statements are no longer supported, because this functionality is now provided automatically by the database server. No error is issued, however, when applications include a **SET Residency** statement; the **SET TABLE** or **SET INDEX** statement is simply ignored.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

Multiple OUT Parameters

In a user-defined routine (UDR), an **OUT** parameter corresponds to a value returned through a pointer. Earlier releases of Dynamic Server supported no more than one **OUT** parameter in UDRs, and any **OUT** parameter was required to appear as the last item in the parameter list. Version 9.4 drops these restrictions, supporting multiple **OUT** parameters anywhere in the parameter list of the UDR. This feature provides greater flexibility in defining UDRs, and removes the need to return collection variables in contexts where multiple returned values are required. JDBC client applications can use this feature to create multiple statement-local variables (SLVs) in the **WHERE** clause of a DML statement that invokes the UDR.

For more information on how to use **OUT** parameters in UDRs, see the *IBM Informix User-Defined Routines and Data Types Developer's Guide*. For more information on **OUT** parameter syntax, see the *IBM Informix Guide to SQL: Syntax*.

Sequence Objects

This release introduces new DML statements (CREATE SEQUENCE, ALTER SEQUENCE, RENAME SEQUENCE, DROP SEQUENCE) for sequence generators, database objects that multiple users can access concurrently to generate unique integers in the INT8 range.

The GRANT and REVOKE statements have been enhanced to support access privileges on sequence objects, and the CREATE SYNONYM and DROP SYNONYM statements can now reference synonyms for sequence objects in the local database. Two new operators, CURRVAL and NEXTVAL, can read or increment the value of an existing synonym. The system catalog includes a new **syssequences** table for information about sequence objects. Sequences are an efficient way to generate primary key values.

For more information on sequence object syntax, see the *IBM Informix Guide to SQL: Syntax*.

ANSI Join Syntax

The syntax of the SELECT statement has been enhanced to support the ANSI/ISO syntax for cross joins, right outer joins, and full outer joins. The keywords CROSS, RIGHT, and FULL are now supported in the context of queries that join two or more tables. This feature provides greater compliance with the ANSI standard for SQL.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

Unions in Subqueries of SELECT Statements

The UNION operator is allowed in subqueries of SELECT statements. The elements of a union are SELECT statements that can recursively contain other unions.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

LVARCHAR Data Types Greater Than 2048 Bytes

In previous releases, database columns of the LVARCHAR built-in opaque data type had an upper limit of 2048 bytes. Version 9.4 supports a *size* parameter in the declarations of LVARCHAR columns (or LVARCHAR variables of SPL), where *size* can be up to 32,739 bytes.

For backward compatibility, LVARCHAR objects declared with no *size* parameter can store up to 2048 bytes. This feature increases the storage capacity of the varying-length data types of Dynamic Server.

For more information, see the *IBM Informix Guide to SQL: Reference*.

New SQL Reserved Words

IBM Informix Dynamic Server, Version 9.4, recognizes new SQL keywords that might affect migration of your applications. Although you can use almost any word as an SQL identifier, syntactic ambiguities can occur in contexts where the keyword is also valid. An ambiguous statement might not produce the desired results.

For information about workarounds for such ambiguities, see the *IBM Informix Guide to SQL: Syntax*.

The following SQL keywords are new in Dynamic Server, Version 9.4:

COLLATION	FULL	RESTART
CROSS	INSTEAD	RIGHT

If you are migrating from a Dynamic Server release earlier than Version 9.30, see the release notes to Version 9.30 for words that have been added to the list of SQL keywords since Version 9.21.

For a complete list of SQL keywords, see Appendix A of the *IBM Informix Guide to SQL: Syntax*, Version 9.4.

New Environment Variables

The new **USETABLENAME** environment variable can invalidate the use of synonyms in ALTER TABLE and DROP TABLE statements of SQL.

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

The section “Enterprise Replication Enhancements” on page 2-59 describes additional new environment variables that can affect Enterprise Replication.

GLS Enhancements

Dynamic Server Version 9.4 uses Version 4.0 of the GLS library, which supports important new features for databases that do not use the default locale.

For information on the new collation order feature, see “Session-Level Non-Default Collation” on page 2-63.

All GLS features are documented in the *IBM Informix GLS User’s Guide*.

Support for Unicode

The GLS library now supports the International Components for Unicode (ICU) code points for multilingual data, based on the ICU open source implementation of Unicode. By internally mapping the code set from ICU, rather than loading it from external locale files, this feature enables you to store, retrieve, and display strings from multiple languages within the same database.

Support for Unicode Collation

The GLS library now supports the Unicode Collation® Algorithm that was developed by the Unicode consortium for comparing two Unicode strings. This de facto standard for multinational applications incorporates ICU technology.

Full Support for Chinese GB18030-2000 Locale

The previous release of the GLS library (Version 3.13.xC4) supported the code points within the Basic Multilingual Plane (BMP) of Unicode (code points 0x00 through 0xFFFF). The new version now supports all GB18030-2000 code points, using ICU.

Reliability, Availability, and Supportability Features

The following additional new features are designed to improve the reliability, availability, and supportability of Dynamic Server.

For more information on these features, see the *IBM Informix Dynamic Server Administrator’s Reference*.

Dynamically Monitor Queries

Ability to monitor queries dynamically using the **onmode -Y** command.

Print the Session Control Block Address

Print the session control block address with the **onstat -g ses** command.

Display Environment Variable Settings

Display the current setting and values of environment variables with the **onstat -g env** command.

Print Online Chunk Pages

Ability to specify the number of pages to print, whether to print just the page headers, and to print pages from chunks that are online with the **oncheck** utility.

Display Stored Procedure Information

Display the types and values of host variables in SQL statements, show the stored procedure stack, and show the current SQL statement in a stored procedure using the **onstat -g sql** command.

DataBlade API Enhancements

The following enhancements have been made to functions that are valid within DataBlade API modules.

New **mi_get_db_locale()** Function

Use the **mi_get_db_locale()** function to return the value of the current database server locale.

Task	Publication
Return the value of the current database server locale.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use the mi_get_db_locale() function.	<i>IBM Informix DataBlade API Function Reference</i>

New **mi_get_transaction_id()** Function

Use the **mi_get_transaction_id()** function to return the ID of the current transaction.

Task	Publication
Return the ID of the current transaction.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use the mi_get_transaction_id() function.	<i>IBM Informix DataBlade API Function Reference</i>

New **mi_realloc()** function

Use the **mi_realloc()** function to change the size of an existing memory block.

Task	Publication
Change the size of an existing memory block.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use the mi_realloc() function.	<i>IBM Informix DataBlade API Function Reference</i>

New `mi_stack_limit()` Function

Use the `mi_stack_limit()` function to determine whether the current user stack has the specified amount of free space.

Task	Publication
Determine whether the current user stack has the specified amount of free space.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use the <code>mi_stack_limit()</code> function.	<i>IBM Informix DataBlade API Function Reference</i>

New `mi_system()` Function

Use the `mi_system()` function to execute operating system commands in a separate thread.

Task	Publication
Use the <code>mi_system()</code> function.	<i>IBM Informix DataBlade API Function Reference</i>

Enhanced Stream Support

Version 9.4 provides stream support for greater than 2 GB files.

High-Performance Loader Enhancements

The following enhancements have been made to the High-Performance Loader (HPL).

All new features for HPL are documented in the *IBM Informix High-Performance Loader User's Guide*.

Full Use of Storage Media

HPL utilities can use the full size of the storage media:

- For `ipload`, check the **Write/read to/from tape until end of device** checkbox on the **Load Select Job** or **Unload Select Job** windows.
- For `onpload` or `onpladm`, specify the **-Z** option with the `onpload` or `onpladm run job` commands.

New Location for the Custom-Code Shared Library File

Previously, the custom-code shared library file was installed in the `/usr/lib` directory. Now it is installed in the `$INFORMIXDIR/lib` directory. You can set the location of this file with the new `HPL_DYNAMIC_LIB_PATH` configuration parameter.

Custom-Code Function Input and Output Length

You can now use a different length for data in the input and output arguments of custom-code functions by setting the `HPLAPIVERSION` configuration parameter.

Backup and Restore Enhancements

The following enhancements have been made to the ON-Bar and `ontape` utilities for Dynamic Server, Version 9.4.

All new backup and restore features are documented in the *IBM Informix Backup and Restore Guide*.

Renaming Chunks During a Cold Restore

You can rename chunks by specifying new chunks paths and offsets during a cold restore with ON-Bar and **ontape**. This option is useful if you need to restore storage spaces to a different disk from the one on which the backup was made.

Full Use of Storage Media and Increased File Size Limit

The **ontape** utility can now use the full size of the storage media if the specified tape size is 0. The **ontape** utility can now backup and restore file sizes up to 4 Terabytes.

Installation Enhancements

The following enhancements have been made to the files used in installation and the installation process.

No Files Installed in the `/usr/lib` Directory

Files that were previously installed in the `/usr/lib` directory on UNIX are now installed in `$INFORMIXDIR/lib`. Specifically, the HPL custom-code shared library file and the optical shared library file are no longer installed in `/usr/lib` (see *IBM Informix High-Performance Loader User's Guide* and the *IBM Informix Optical Subsystem Guide*). In addition, SmartDisk is no longer supported.

For more information, see the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X*.

More Recent Client and GLS Files Are Not Overwritten

The installation program on UNIX prompts the user to avoid overwriting existing client or GLS files that are more recent than those included with the database server.

For more information, see the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X*.

Serial Number and Key are No Longer Needed

The installation program no longer prompts for a serial number and key.

This change is reflected in the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X* and the *IBM Informix Dynamic Server Installation Guide for Windows*.

New Features in Version 9.3

The new features for Dynamic Server, Version 9.3, fall into the following major areas:

- Database server usability enhancements
- DataBlade API enhancements
- Enterprise Replication enhancements
- Extensibility enhancements
- Java enhancements
- Performance enhancements
- SQL enhancements

UNIX Bundle Installer

Use the IBM Informix UNIX Bundle Installer to install your IBM Informix products on UNIX or Linux and configure a demo database server that you can customize. The installer filename is **ids_install**.

For the installation instructions, see the *IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X*.

Database Server Usability Enhancements

This release includes new features that make the database server easier to install, use, and manage.

Ability to Display the Maximum Number of Connections

When the database server starts, it checks the number of connections that the license allows and writes a message to the message log.

For information on how to display the maximum number of connections, see the chapter on initializing the database server in the *IBM Informix Dynamic Server Administrator's Guide*. For the messages, see the *IBM Informix Administrator's Reference*.

Changes to onconfig.std File

Use the VPCLASS parameter options for VP type, number, affinity, and noage, to configure virtual processor classes. The VPCLASS configuration parameter has replaced the following ONCONFIG configuration parameters:

- AFF_NPROCS
- AFF_SPROC
- NOAGE
- NUMAIOVPS
- NUMCPUVPS

The following configuration parameters are obsolete:

- LBU_PRESERVE
- LOGSMAX

For more information on configuration parameters and **onconfig.std**, see the *IBM Informix Administrator's Reference*.

Database Server Administration Utilities (Windows)

The following IBM Informix utilities simplify administration of the database server on Windows:

- The **ixpasswd.exe** utility changes the logon password for all services that log on as user **informix**.
- The **ixsu.exe** utility launches a command-line window that runs as the specified user.
- The **ntchname.exe** utility changes the registry entries for Dynamic Server from the old hostname to the new hostname.

For more information on these utilities, see the *IBM Informix Administrator's Guide*.

High-Availability Data Replication Failover Scripts

Use the `hdrmkpri.sh` and `hdrmksec.sh` scripts to switch the roles of database servers in a High-Availability Data Replication (HDR) pair. For more information, see the *IBM Informix Administrator's Guide*.

DataBlade API Enhancements

This release includes the following improvements in the DataBlade API.

New PER_STMT_EXEC and PER_STMT_PREP Memory Durations

When a user-defined routine (UDR) calls a memory-allocation function, the memory exists until the duration assigned to that memory expires. The `PER_STMT_PREP` memory duration lasts for the life of a prepared statement. The `PER_STMT_EXEC` memory duration lasts for the duration of the SQL statement.

Use the `PER_STMT_EXEC` and `PER_STMT_PREP` memory durations instead of the `PER_STATEMENT` memory duration.

Task	Publication
Use the new memory durations.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use memory durations in functions that have a duration argument.	<i>IBM Informix DataBlade API Function Reference</i>
Use <code>mi_dalloc()</code> to specify a memory duration.	<i>IBM Informix Virtual-Table Interface Programmer's Guide</i>
Allocate user-data memory with <code>PER_STMT_EXEC</code> memory duration.	<i>IBM Informix Virtual-Table Interface Programmer's Guide</i>
Display information on the <code>PRP.sessionid.threadid</code> and <code>EXE.sessionid.threadid</code> pools.	<i>IBM Informix Administrator's Reference</i> (see onstat -g mem)

NULL Connections for mi_lo() Functions

The DataBlade API provides a set of `mi_lo*()` functions for handling smart large objects. This feature permits a NULL connection using the same error-handling behavior as with a valid connection. To use `mi_lo*()` functions without a connection, specify the NULL argument.

Task	Publication
Pass a NULL connection to an <code>mi_lo*()</code> function.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Specify a NULL connection in a call to an <code>mi_lo*()</code> routine that takes a connection descriptor.	<i>IBM Informix DataBlade API Function Reference</i>

New mi_collection_card() Function to Obtain Cardinality for Collections

Use the `mi_collection_card()` function in a UDR to return the cardinality of a collection (the number of items in a collection such as LIST, SET, and MULTiset).

Task	Publication
Determine the cardinality of a collection.	<i>IBM Informix DataBlade API Programmer's Guide</i>

Task	Publication
Use the mi_collection_card() function.	<i>IBM Informix DataBlade API Function Reference</i>
Use LIST, MULTISSET, and SET data types.	<i>IBM Informix Guide to SQL: Reference</i>

Access to Files on a Client Computer One Buffer at a Time

The DataBlade API provides a set of **mi_file*()** functions for performing I/O operations on files. Previously, the **mi_file*()** functions transferred the entire file to the client computer but now these functions can transfer the file one buffer at a time.

Task	Publication
Access client files one buffer at a time.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Open a file on the client by passing the MI_O_CLIENT_FILE flag to mi_file_open() .	<i>IBM Informix DataBlade API Function Reference</i>

New Callbacks to Handle Transactions

The database server invokes three new callbacks for transactions:

- The database server invokes a save point callback (MI_EVENT_SAVEPOINT) before committing or rolling back a save point in a transaction.
- The database server calls MI_EVENT_COMMIT_ABORT before committing or rolling back a transaction
- The database server calls MI_EVENT_POST_XACT after committing or rolling back a transaction.

For details, see the *IBM Informix DataBlade API Programmer's Guide*.

New Function for Determining the Transaction State for DataBlade Modules

The **mi_transaction_state()** function returns the current transaction state for a DataBlade module to the caller. The transaction states are none, implicit, or explicit.

Task	Publication
Determine the state of a transaction.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Use the mi_transaction_state() function.	<i>IBM Informix DataBlade API Function Reference</i>

Enterprise Replication Enhancements

Enterprise Replication conversion and reversion is now manual instead of automatic. For instructions, see the *IBM Informix Migration Guide*. For the error messages, see the *IBM Informix Administrator's Reference*.

Dynamic Server, Version 9.3, includes extensibility enhancements, performance improvements, functionality enhancements, and command-line changes for Enterprise Replication.

Replication of Extensible Data Types

Enterprise Replication provides support for replicating the following extensible data types:

- Data stored as smart large objects in sbspaces (CLOB and BLOB data types), CLOB and BLOB columns (explicitly specified in the table schema), and updates to CLOB and BLOB columns (with some restrictions)
- Opaque user-defined types (UDTs)
- Multirepresentational data types, if the required stream support functions exist. For information on writing the required functions, refer to the *IBM Informix DataBlade API Function Reference*.
- IBM Informix Spatial DataBlade Module

Version 9.3 does not include support for replication of the following user-defined types:

- Row types
- Collections
- Lists
- Sets and Multisets

Enterprise Replication allows the following (with some restrictions):

- UDT column references and UDRs in replicate WHERE clauses
- UDTs for primary key columns

For more information, see *IBM Informix Dynamic Server Enterprise Replication Guide*:

- Replicating simple and smart large objects
- Considerations for replicating opaque data types
- UDT support functions

Support Functions for Replication of User-Defined Types

To replicate UDTs, Enterprise Replication requires that the UDT designer provide two support functions: **streamwrite()** and **streamread()**. The **streamwrite()** function converts the UDT column data from the in-server representation to a representation that can be shipped over the network. On the target server, Enterprise Replication calls the **streamread()** function for each UDT column that it transmitted using the **streamwrite()** function.

For more information, see the section on writing opaque-type support functions in the *IBM Informix DataBlade API Programmer's Guide*.

Performance Enhancements to Enterprise Replication

Enterprise Replication includes the following performance improvements to parallel processing:

- Enterprise Replication now applies all replicates (in replicate sets and individually) in parallel by default.
- Enterprise Replication threads now apply transactions from the same source in parallel unless they contain updates to the same row.
- Enterprise Replication threads normally commit on the target in the same order as on the source.
- Enterprise Replication threads can commit out of order on the target if there are no conflicts.
- Enterprise Replication now uses buffered logging to apply transactions.

Improved parallel processing is built in and requires no user configuration or interaction. However, this feature is automatically disabled if you are using page-level locking.

SERIAL Column Primary Keys

The CDR_SERIAL configuration parameter enables control over generating values for serial and SERIAL8 columns in tables defined for replication. This feature is useful for generating serial column primary keys in an Enterprise Replication environment.

For more information, see CDR_SERIAL in the section on configuration parameters in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Replicate Sets and Exclusive Replicate Sets

You can manage replicates individually and as part of a *replicate set*. Put tables in *exclusive replicate sets* to guarantee referential integrity between tables when you use any form of time-based replication.

Warning: Replicate groups are not supported in Version 9.3. Before you migrate to Version 9.3, you must remove any replicate groups.

For more information, see creating and managing replicate sets in the *IBM Informix Dynamic Server Enterprise Replication Guide* and migrating Enterprise Replication data in the *IBM Informix Migration Guide*.

Replicating Only Changed Columns

Enterprise Replication provides the ability to replicate only the changed columns, rather than the entire row.

If only changed columns are replicated, the data for all replicated columns might not be available for spooling to the ATS (Aborted Transaction Spooling) and RIS (Row Information Spooling) files. Therefore, the format for these files has changed.

For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide* regarding:

- Replicating only changed columns
- Aborted Transaction Spooling files
- Row Information Spooling files

Spooling of Replicate Data to Nonlogging Smart Large Objects

Enterprise Replication spools row data in the send and receive queues to an sbspace that you specify in the CDR_QDATA_SBSPACE configuration parameter. You can control logging of these sbspaces.

Enterprise Replication spools transaction records from the send and receive queues to a dbspace that you specify in the cdr_qhdr_dbspace parameter.

For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide* regarding:

- Setting up send and receive queue spool areas
- Defining replication servers
- Specifying CDR_QDATA_SBSPACE and CDR_QHDR_DBSPACE configuration parameters

In-Place Alters to Add or Drop Shadow Columns (CRCOLS)

Enterprise Replication uses shadow columns for conflict resolution. The database server now processes the following ALTER statements for adding and dropping shadow columns as in-place alters in most cases:

```
ALTER TABLE ... ADD CRCOLS
ALTER TABLE ... DROP CRCOLS
```

In-place alters are quick because the database server updates each row in place instead of copying the entire table. The in-place processing of these ALTER statements requires no user action.

Task	Publication
Prepare tables for conflict resolution.	<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>
Add or drop shadow columns: <ul style="list-style-type: none">• ALTER TABLE ... ADD CRCOLS• ALTER TABLE ... DROP CRCOLS	<i>IBM Informix Guide to SQL: Syntax</i>
Understand the performance advantages of in-place alters and when they occur.	<i>IBM Informix Performance Guide</i>

New onstat Options for Enterprise Replication

Use the following **onstat** options to obtain information about replication of user-defined routines (UDRs):

- **onstat -g dss UDR**
- **onstat -g dss UDRx**
- **onstat -g grp UDR**
- **onstat -g grp UDRx**

For details, see the appendix on **onstat** commands in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

The cdr finderr Utility

This release includes updates to the command-line interface to support new features, including a new **cdr finderr** utility which looks up a specific Enterprise Replication error number and displays the corresponding error text.

For more information, see the command-line utility reference in the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Extensibility Enhancements

This release includes the following improvements in the area of extensibility.

DeepCopy Function for Multirepresentational Data Types

Use the **DeepCopy** function for user-defined types with multiple representations, such as images. The **DeepCopy** function copies the user-defined type so that the user can safely allocate both the in-row value and out-of-row data with the default memory duration.

After you register the **DeepCopy** function for the multirepresentational types, the database server automatically invokes **DeepCopy**.

Task	Publication
Use multirepresentational data types.	<i>IBM Informix User-Defined Routines and Data Types Developer's Guide</i>
Use the DeepCopy function in a UDR.	<i>IBM Informix DataBlade API Programmer's Guide</i> <i>IBM Informix User-Defined Routines and Data Types Developer's Guide</i>

Nearest Neighbor Queries in R-Trees

R-tree indexes support nearest-neighbor queries. A *nearest-neighbor query* asks for items in a spatial database that are the nearest to a specific location or object. If you do a nearest-neighbor query on a map of the San Jose area, Santa Clara would be the nearest neighbor, but not San Francisco. Version 9.3 supports *composite R-tree indexes*.

For more information, see the *IBM Informix R-Tree Index User's Guide*.

Temporary Sbspaces and Smart Large Objects

Smart-large-object performance is significantly faster for certain operations. Writes to temporary smart large objects are faster than for standard smart large objects.

Use *temporary smart large objects* to store text, image, or user-defined data that you need temporarily for a user session. You can store temporary smart large objects in a standard sbspace or *temporary sbspace*. If temporary smart large objects are stored in a temporary sbspace, the metadata and user data are not logged.

To specify the default temporary sbspace, use the SBSPACETEMP configuration parameter.

Task	Publication
Use temporary sbspaces and smart large objects.	<i>IBM Informix Administrator's Guide</i>
Use onspaces to create temporary sbspaces.	<i>IBM Informix Administrator's Reference</i>
Improve temporary space utilization.	<i>IBM Informix Performance Guide</i>

Improved Space Allocation of User Data and Metadata in Sbspaces

The database server reserves 40 percent of the user-data space in the sbspace chunk. When the chunk runs out of metadata or user-data space, the database server moves some of the reserved space to the corresponding area. This feature enables the database server to use the space in the sbspace more efficiently.

Task	Publication
Monitor the metadata and user-data areas.	<i>IBM Informix Administrator's Guide</i>
Read about sbspace structure.	<i>IBM Informix Administrator's Reference</i>
Estimate the size of the metadata area and improve space utilization.	<i>IBM Informix Performance Guide</i>

J/Foundation Enhancements

You can create and execute UDRs and applications written in Java. The following improvement for this release is that J/Foundation performance is now faster.

For more information, see *J/Foundation Developer's Guide*, the *IBM Informix JDBC Driver Programmer's Guide*, and "Java Features in 9.21" on page 2-81.

JVM 1.3 Support in J/Foundation

Dynamic Server supports Java 2 and includes the Java Runtime Environment (JRE). The database server supports Version 1.3 of the Java Virtual Machine (JVM) and embeds the hotspot server VM.

Performance Enhancements

This release includes many features that help you monitor and improve performance.

Configurable Default Lock Modes

You can set the default lock mode to page or row for new tables in the following ways:

- LOCK MODE clause in the ALTER TABLE or CREATE TABLE statement
- IFX_DEF_TABLE_LOCKMODE environment variable
- DEF_TABLE_LOCKMODE configuration parameter

Task	Publication
Configure lock mode.	<i>IBM Informix Performance Guide</i>
Use the DEF_TABLE_LOCKMODE configuration parameter.	<i>IBM Informix Administrator's Reference</i>
Use the LOCK MODE clause in the ALTER TABLE or CREATE TABLE statement.	<i>IBM Informix Guide to SQL: Syntax</i>

The onstat -g stm Option

Use the **onstat -g stm** option to display the memory that prepared SQL statements use:

```
onstat -g stm session_id
```

For more information on **onstat -g stm**, see the *IBM Informix Performance Guide* and the *IBM Informix Administrator's Reference*.

The Ability to Display the Query Plan Without Executing the Query

To display the query plan without executing the query, use the SET EXPLAIN ON AVOID_EXECUTE statement or the AVOID_EXECUTE optimizer directive. This option allows you to evaluate the query plan that the optimizer has written to the **sqexplain.out** file.

To use this feature as a directive for a single statement:

```
SELECT --+EXPLAIN AVOID_DIRECTIVE  
* FROM tablename;
```

To use this feature as a SET EXPLAIN keyword for a block of statements:

```
SET EXPLAIN ON AVOID_EXECUTE;
```

Task	Publication
Improve performance of queries and use optimizer directives.	<i>IBM Informix Performance Guide</i>
Use SET EXPLAIN and optimizer directives.	<i>IBM Informix Guide to SQL: Syntax</i>

Dynamic Addition of Logical Logs

The database server automatically adds a logical-log file after the current log file when the next log file contains an open transaction. Dynamic log allocation prevents logs from filling and hanging the system during long-transaction rollbacks. You also can choose whether to add a log file manually after the current log file or at the end of the log file list.

The DYNAMIC_LOGS configuration parameter determines whether the database server allocates new logical-log files dynamically. The LTXHWM and LTXEHWM configuration parameters set high-watermarks for long transactions. If DYNAMIC_LOGS is set to 1 or 2, the default LTXHWM value is 80 percent and LTXEHWM is 90 percent.

The **onstat -l** output also displays information about temporary logical logs.

Task	Publication
Use dynamically allocated logical logs.	<i>IBM Informix Administrator's Guide</i>
Use the onparams and onstat -l commands, and the DYNAMIC_LOGS, LTXHWM, and LTXEHWM parameters.	<i>IBM Informix Administrator's Reference</i>

SQL Enhancements

This release includes several new SQL statements that ease migration from non-Informix databases to Dynamic Server, Version 9.3.

Optional FROM in the DELETE Statement

The DELETE statement no longer requires the FROM keyword. You can use this syntax:

```
DELETE customer WHERE customer_num = 105;
```

For more information, see the *IBM Informix Guide to SQL: Syntax*.

REVOKE AS User

The REVOKE statement allows the owner of a database object to revoke the privileges of other users. REVOKE ... AS and REVOKE FRAGMENT .. AS allow *user2* to revoke the privileges for *user1*. For example, you can revoke privileges from user names such as **informix** that are authorization identifiers but not users that the operating system recognizes. You can use this syntax:

```
REVOKE privilege FROM user1 AS user2;
```

For more information on the REVOKE statement, see the *IBM Informix Guide to SQL: Syntax*.

New Features in Dynamic Server, Version 9.21

These features were introduced in IBM Informix Dynamic Server (IDS), Version 9.21.

ANSI Join Syntax

You begin an ANSI join with the [LEFT] [OUTER] JOIN keywords, use the ON clause to specify the join filter, and use the WHERE clause to specify a post-join filter.

Task	Publication
Use the ANSI join syntax.	<i>IBM Informix Guide to SQL: Syntax</i>
Use join filters and post-join filters, and interpret SET EXPLAIN output for ANSI joins.	<i>IBM Informix Performance Guide</i>

Rename Index Statement

Use the RENAME INDEX statement to change the name of an index.

For more information, see the *IBM Informix Guide to SQL: Syntax*.

Nonlogging (Raw) Tables

You can create nonlogging tables in a logging database on Dynamic Server. These tables are also called raw tables. Raw tables do not support primary constraints, unique constraints, and rollback. However, these tables can be indexed and updated. You can create either a standard or raw table and change tables from one type to another.

Task	Publication
Use nonlogging tables.	<i>IBM Informix Administrator's Guide</i>
Load and unload nonlogging tables. Lock nonlogging tables.	<i>IBM Informix Performance Guide</i>
Specify the logging type in the ALTER TABLE and CREATE TABLE statements.	<i>IBM Informix Guide to SQL: Syntax</i>

onpladm Utility

High-Performance Loader (HPL) includes the command-line utility **onpladm**. You can use the **onpladm** utility to create, modify, describe, list, run, configure, and delete jobs for unloading and loading tables or an entire database. For more information, see *IBM Informix High-Performance Loader User's Guide*.

The onbar -b -l Command

Use the **onbar -b -l** command instead of **onbar -l** to back up the logical logs.

For more information, see the *IBM Informix Backup and Restore Guide*.

9.x DB-Access to 7.x Synonyms

In earlier versions, you could use DB-Access to access synonym names only if the remote database server was Version 9.x. Now you can access synonym names on remote Version 7.x database servers.

SQL Statement Cache Improvements

The database server uses the SQL statement cache (SSC) to store SQL statements that a user executes. When users execute a statement stored in the SQL statement cache, the database server does not parse and optimize the statement again, which improves performance.

In Version 9.21, the SQL statement cache was enhanced to support the following capabilities:

- Insert the statement in the SQL statement cache as a *key-only* entry to track the number of times it has been referenced. After the statement has been referenced a specific number of times, it is inserted fully into the cache.
- Control whether statements enter the SQL statement cache after it exceeds its size limit.
- Define multiple pools for the SQL statement cache.

Task	Publication
Learn about the SQL statement cache	<i>IBM Informix Performance Guide</i>
Use qualifying and identical statements; also learn about the memory limits and key-only cache entries.	<i>IBM Informix Guide to SQL: Syntax</i>
Configure the SQL statement cache: <ul style="list-style-type: none"> • STMT_CACHE_HITS • STMT_CACHE_NOLIMIT • STMT_CACHE_NUMPOOL • STMT_CACHE_SIZE 	<i>IBM Informix Administrator's Reference</i>
Display SQL statement cache statistics: <ul style="list-style-type: none"> • onstat -g ssc • onstat -g ssc all • onstat -g ssc pool 	<i>IBM Informix Administrator's Reference</i>
Configure the SQL statement cache on the fly: <ul style="list-style-type: none"> • onmode -W STMT_CACHE_HITS • onmode -W STMT_CACHE_NOLIMIT 	<i>IBM Informix Administrator's Reference</i>
Understand the performance advantages of the SQL statement cache: <ul style="list-style-type: none"> • Use the SQL statement cache • Enable the SQL statement cache • Configure the SQL statement cache • Monitor the SQL statement cache 	<i>IBM Informix Performance Guide</i>

DataBlade API Features

The following DataBlade API features were introduced in Version 9.21.

Functions for Controlling the Virtual-Processor Environment

The DataBlade API now provides functions that allow you to control the virtual-processor (VP) environment from within a UDR. These new functions provide the ability to:

- Obtain information about a VP.
- Obtain information about a VP class.
- Lock the UDR.
- Change the VP environment.

For more information, see the *IBM Informix DataBlade API Programmer's Guide*.

Functions for Getting Information About a UDR

The DataBlade API now provides functions that obtain additional information about a UDR, including the:

- Name of the UDR (as defined in the **sysprocedures** system catalog table)

- Routine identifier
- Address of the **MI_FPARAM** structure for the UDR

For more information, see the *IBM Informix DataBlade API Programmer's Guide*.

Java Features in 9.21

The following Java features were introduced in Version 9.21.

JVM 1.2 Support in J/Foundation

Dynamic Server supports Version 1.2 of the Java Virtual Machine (JVM).

Default Values of Java Configuration Parameters

The default values of the **JDKVERSION**, **JVPJAVAHOME**, **JVPJAVALIB**, and **JVPJAVAVM** parameters in the **ONCONFIG** file have changed for Dynamic Server.

JDBC 2.0 Support

IBM Informix JDBC driver is bundled with Embedded SQLJ 1.10.1.JC1, a product for embedding SQL statements in Java. Dynamic Server supports the following JDBC 2.0 features:

- Complex data types
- Collections
- Scrollable cursors
- Batch updates
- Interval data types
- Extensions to prepared statement
- Callable statements

GLS Support for J/Foundation

Dynamic Server supports the following GLS features:

- **CLIENT_LOCALE**, **DBCENTURY**, **DB_LOCALE**, **GL_DATE**, **GL_DATETIME**, and **DBTIME**, environment variables
- New connection properties (**NEWLOCALE** and **NEWCODESET**) for mapping a locale or code set in the JDBC driver

update_jars.sql Script

Use the **update_jars.sql** script to update the names of jar files in a database after you rename the database.

Java Runtime Environment Variables

Dynamic Server supports the **JVM_MAX_HEAP_SIZE**, **JAR_TEMP_PATH**, **JAVA_COMPILER**, and **AFDEBUG** environment variables.

Partial Support for Variable-Length Opaque-Types

You can now write UDRs and DataBlade modules in Java.

Dynamic Server supports the following items:

- Variable-length opaque data types
- Data I/O conversion routines:
 - input/output
 - send/receive
 - import/export
 - importbin/exportbin

References to J/Foundation Features

For more information on J/Foundation features, see these publications.

Task	Publication
Use JVM 1.2. Use JCBC 2.0 features. Write UDRs and DataBlade modules in Java.	<i>J/Foundation Developer's Guide</i>
Specify Java environment variables.	<i>J/Foundation Developer's Guide</i> <i>IBM Informix Guide to SQL: Reference</i>
Specify Java configuration parameters.	<i>J/Foundation Developer's Guide</i> <i>IBM Informix Dynamic Server Administrator's Reference</i>
Set GLS environment variables. Use the connection properties.	<i>J/Foundation Developer's Guide</i> <i>IBM Informix GLS User's Guide</i>
Use the update_jars.sql script.	<i>IBM Informix Guide to SQL: Syntax</i>

MaxConnect Support

IBM Informix MaxConnect enables IBM Informix Dynamic Server to support greatly increased numbers of client connections. Informix MaxConnect is a new software tier, introduced between the database server and clients, that transparently funnels multiple client connections onto a smaller number of server connections. The database server is freed from managing thousands of client connections, which results in improved response time and decreased CPU cost for the database server.

Important: Informix MaxConnect and the *IBM Informix MaxConnect User's Guide* ship separately from IBM Informix Dynamic Server (IDS), Version 9.3.

The following features were introduced in Version 9.21 to support the IBM Informix MaxConnect product, which is separately orderable:

- New network protocols
The database server supports Informix MaxConnect with two new network protocols: **ontliimc** and **onsocimc**.
- New utility options to monitor Informix MaxConnect
 - **onstat -g imc**
 - **imcadmin**
 - **ISA options**
- New environment variables for Informix MaxConnect
 - **IMCADMIN**
 - **IMCCONFIG**
 - **IMCSERVER**

For more information about installing, configuring, monitoring, and tuning Informix MaxConnect, see the *IBM Informix MaxConnect User's Guide*.

Chapter 3. Using Existing Dynamic Server Features

In This Chapter	3-2
Dynamic Scalable Architecture	3-2
The Shared-Memory Component	3-2
The Disk Component	3-3
The Virtual Processor Component	3-4
Client/Server Connections	3-4
High Performance	3-5
Memory Management	3-5
Dynamically Sharing Memory	3-5
Buffering Transactions	3-5
Using NFS-Mounted Directories	3-5
Fragmentation	3-5
Parallelization	3-6
Query Optimizer	3-6
Fault Tolerance and High Availability	3-6
Backup and Restore	3-6
ontape Utility	3-7
ON-Bar Utility	3-7
IBM Informix Storage Manager	3-7
The archecker Utility	3-7
Fast Recovery	3-8
Mirroring	3-8
Data Replication	3-8
High-Availability Clusters	3-8
Enterprise Replication	3-9
Types of Data That You Can Replicate	3-9
Database Server Security	3-9
Auditing Database Events	3-10
Informix RDBMS Features	3-10
Structured Query Language (SQL)	3-10
Stored Procedure Language (SPL)	3-10
System Catalog Tables	3-11
Data Types	3-11
Application Types	3-13
OLTP Applications	3-13
DSS Applications	3-13
Database Support	3-14
Relational Databases	3-14
ANSI-Compliant Databases	3-14
Object-Relational Databases	3-14
Simple and Smart Large Objects	3-15
User-Defined Data Types	3-16
Complex Data Types	3-16
User-Defined Routines	3-16
Operator Functions	3-17
User-Defined Casts	3-17
Inheritance	3-17
User-Defined Aggregates	3-17
User-Defined Virtual Processors	3-17
DataBlade Modules	3-18
Dimensional Databases	3-18
Distributed Queries and Multiphase Transactions	3-19
Access Methods	3-19
Primary Access Methods	3-19
Secondary Access Methods	3-20

Generic B-Tree Indexes	3-20
R-Tree Indexes	3-20
User-Defined Primary Access Methods	3-20
User-Defined Secondary Access Methods	3-20

In This Chapter

This chapter provides an overview of Dynamic Server architecture and significant features. Dynamic Server delivers database scalability, manageability, and performance.

Dynamic Scalable Architecture

Dynamic Server is a multithreaded *object-relational database* server that uses symmetric multiprocessor (SMP) and uniprocessor architectures. In an SMP system, multiple CPUs or processors all run a single copy of the operating system, sharing memory and communications.

Dynamic scalable architecture (DSA) allows you to scale resources to varying application loads (from small to huge) and improves performance. Key elements of DSA are the virtual processors that manage central processing, disk I/O, networking, and optical functions in parallel.

Scalability has two aspects: speedup and scaleup. *Speedup* means the ability to add computing hardware and achieve faster performance for a decision support (DSS) query or online transaction processing (OLTP). *Scaleup* means the ability to process a larger workload with a correspondingly larger amount of computing resources in the same time. For more information on DSS and OLTP operations, see “Application Types” on page 3-13.

Informix database server architecture consists of the following main components:

- Shared memory
- Disk
- Virtual processor
- Client/server connections

For more information on database server architecture, see the *IBM Informix Performance Guide*. For information on how to use Dynamic Server, see the *IBM Informix Administrator's Guide* and the *IBM Informix Administrator's Reference*. For a glossary of terms that are used in IBM Informix publications, see the *IBM Informix Guide to SQL: Reference*.

The Shared-Memory Component

Shared memory is an operating-system feature that lets the database server processes and threads share data by sharing access to pools of memory. The database server uses shared memory for the following purposes:

- To reduce memory use and disk I/O
- To perform high-speed communication between processes
- To enable virtual processors and utilities to share data

The database server creates the following portions of shared memory:

- Resident
 - Caches data from the disk for faster access

- Virtual
Maintains and controls the resources required by the virtual processors
- Interprocess (IPC) communications
Provide a fast communications channel for local client applications that use IPC communication on UNIX
- Virtual extension
Enables DataBlade modules and user-defined routines (UDRs) to run in user-defined virtual processors

The Disk Component

The database server uses the physical units of storage to allocate disk space. You define the logical units that the database server uses to store data. All the databases and all the system information that is necessary to maintain the database server reside within the disk component.

On UNIX, the database server stores data in two types of disk space: raw and cooked. The database server allows you to use either type of disk space or a combination of both types.

- *Raw disk space* (also called *unbuffered* disk space) is unformatted space where the database server manages the physical organization of the data.
- *Cooked disk space* (also called *buffered* disk space) refers to regular operating-system files.

On Windows, the database server stores data in two types of disk space:

- New Technology File System (NTFS)
- Logical partition or physical drive

The database server uses the following physical units to manage disk space.

Physical Unit	Description
Chunk	The largest unit of database server data storage
Page	The physical unit of disk storage to read from and write to databases
Blobpage	The physical unit of disk storage to store simple large objects in a blobpage
Sbpage	The physical unit of disk storage to store smart large objects in an sbpage
Extent	A fixed amount of space to contain the data stored in a table

The database server uses the following logical units to manage disk space. Dbspaces, blobspaces, and sbspaces are composed of one or more chunks.

Logical Storage Unit	Description
Dbpace	Stores databases, tables, logical-log files, the physical log, and internal data
Blobspace	Stores simple large objects (TEXT and BYTE data)
Sbpace	Stores smart large objects (CLOB and BLOB data)
Extspace	References the location of external data
Database	Contains tables and indexes
Table	Consists of a row of column headings with zero or more rows of data values

Logical Storage Unit	Description
Tblspace	Contains the disk space allocated to a given table or fragment

The database server maintains the following storage structures to ensure physical and logical data consistency.

Data Consistency	Description
Logical log	A circular file that stores log records of transactions and database server changes
Physical log	A set of disk pages where the database server stores an unmodified copy of the page (called a <i>before image</i>)

For information about storage spaces and logical and physical logs, see the *IBM Informix Administrator's Guide*. Logical-log record formats are discussed in the *IBM Informix Administrator's Reference*.

The Virtual Processor Component

Database server processes are called *virtual processors* because they function like a CPU in a computer. Just as a CPU runs multiple operating-system processes to service multiple users, a virtual processor runs multiple *threads*, or pieces of work, to service multiple SQL client applications. Virtual processors improve database server performance.

Client/Server Connections

You can put a client on one computer and the database server on another or the same computer. A *client* is an application that a user runs to request or modify information from a database by issuing SQL statements. The following IBM Informix tools are client programs:

- DB–Access
- Enterprise Replication
- High Performance Loader (HPL)
- Informix ESQL/C
- IBM Informix JDBC Driver
- ODBC
- DataBlade API

The database administrator specifies the types of connections that the database server supports in the **sqlhosts** file on UNIX or the PROTOCOL field in the SQLHOSTS registry key on Windows.

Use a *network protocol* to connect to and transfer data between database servers, or between a client and a database server. You must establish a *connection* between the client and database server before data transfer can take place and you must maintain it for the duration of the data transfer.

A *multiplexed connection* uses a single network connection between the database server and a client to handle multiple database connections from the client. If you need to manage several hundred or thousands of client connections, consider ordering IBM Informix MaxConnect. For details, see “IBM Informix MaxConnect (UNIX)” on page 1-7.

The database server supports the following types of connections to communicate between client applications and a database server.

Connection Type	Windows	UNIX	Local	Network
Sockets	X	X	X	X
TLI (TCP/IP)		X	X	X
TLI (IPX/SPX)		X	X	X
Shared memory		X	X	
Stream pipe		X	X	
Named pipe	X		X	

For information about client/server configurations that the database server supports, see the *IBM Informix Administrator's Guide*.

High Performance

Dynamic Server achieves high performance through the following mechanisms:

- Memory management
- Fragmentation
- Parallelization
- Query optimization

Memory Management

Dynamic Server provides several options to help you manage memory to optimize performance.

Dynamically Sharing Memory

All applications that use the same database server share data in the memory space of the database server. The database server adds memory dynamically as it needs it. The database server administrator can control the amount of shared memory that is available to the database server.

Buffering Transactions

You can determine how the database server logs transactions. A *transaction* is a collection of SQL statements that is treated as a single unit of work. Your logs can be buffered or unbuffered. Buffered logging holds transactions in memory until the buffer is full, regardless of when the transaction is committed.

For information on how to manage the various aspects of memory to increase performance, see your *IBM Informix Performance Guide* and "SQL Enhancements" on page 2-78. For information on transaction logging, see the *IBM Informix Administrator's Guide*.

Using NFS-Mounted Directories

An Informix storage space can reside on an NFS-mounted file system using regular operating-system files.

Fragmentation

Dynamic Server supports table and index *fragmentation* over multiple disks. Fragmentation lets you group rows within a table according to a distribution scheme. Fragmentation improves performance on large databases.

Dynamic Server supports the following fragmentation schemes:

- *Round-robin fragmentation* places rows one after another in fragments, rotating through the series of fragments to distribute the rows evenly.
- *Expression-based fragmentation* puts rows that contain specified values in the same fragment. You specify a fragmentation expression that defines criteria for assigning a set of rows to each fragment, either as a range rule or some arbitrary rule.

For information on the fragmentation strategies, see the *IBM Informix Database Design and Implementation Guide*. For information about how to create a fragmentation strategy to enhance database performance, see your *IBM Informix Performance Guide*.

Parallelization

The database server can allocate multiple threads to work in parallel on a single query. This feature is known as parallel database query (PDQ).

PDQ can improve performance dramatically when the database server processes queries that DSS applications initiate. PDQ lets the database server distribute the work for one aspect of a query among several processors.

For information on how to implement PDQ and how parallelization can enhance performance, see your *IBM Informix Performance Guide*. For information on the SET PDQPRIORITY environment variable, see the *IBM Informix Guide to SQL: Reference*.

Query Optimizer

The *query optimizer* formulates a query plan to fetch the data rows that are required to process a query. The optimizer evaluates the different ways in which a query might be performed. For example, the optimizer must determine whether indexes should be used. If the query includes a join, the optimizer must determine the join plan (*hash* or *nested loop*) and the order in which tables are evaluated or joined.

For more information on the optimizer, see your *IBM Informix Performance Guide*.

Fault Tolerance and High Availability

Dynamic Server uses the following logging and recovery mechanisms to protect data integrity and consistency if an operating-system or media failure occurs:

- Backup and restore
- Fast recovery
- Mirroring
- High-Availability Clusters
- Enterprise Replication

Backup and Restore

Use the ON-Bar or **ontape** utility to back up your database server data and logical logs as insurance against lost or corrupted data. A program error or disk failure can cause data loss or corruption. If a dbspace, an entire disk, or the database server goes down, use ON-Bar or **ontape** to restore the data from the backup copy. You must use the same utility for both the backup and restore.

The following are basic backup and restore terms:

- A *backup* is a copy of one or more *storage spaces* and the logical logs.
- A *logical-log backup* is a copy to tape or disk of logical-log files that have become full and eligible for backup.

The logical-log files store a record of database server activity that occurs between backups.

- A *restore* recreates data from a backup.
- A *point-in-time restore* allows you to restore the data in a database to a specific time.

A point-in-time restore can undo mistakes, such as dropping a table, that might not be fixable otherwise.

ontape Utility

The **ontape** utility does not require a storage manager. Use **ontape** to perform the following tasks:

- Back up and restore storage spaces and logical logs.
- Change database-logging status.
- Start continuous logical-log backups.
- Use data replication.
- Rename chunks to different pathnames and offsets.

ON-Bar Utility

The ON-Bar utility requires a storage manager such as IBM Informix Storage Manager (ISM). Use ON-Bar to perform the following tasks:

- Back up and restore storage spaces and logical logs.
- Perform point-in-time restores.
- Start continuous logical-log backups.
- Verify a backup with the **archecker** utility.
- Perform external backups and restores.

An *external backup and restore* allows you to copy and physically restore data without using ON-Bar. Then you use ON-Bar for the logical restore.

- Rename chunks to different pathnames and offsets.

For information about backing up data with ON-Bar or **ontape** and the **archecker** utility, see the *IBM Informix Backup and Restore Guide*.

IBM Informix Storage Manager

IBM Informix Storage Manager (ISM) manages data storage for the Informix database server. ISM resides on the same computer as ON-Bar and the database server.

ISM receives backup and restore requests from ON-Bar and directs data to and from storage volumes that are mounted on storage devices. ISM tracks backed-up data through a data life cycle that the database or system administrator determines and also manages storage devices and storage volumes.

For information about ISM, see the *IBM Informix Storage Manager Administrator's Guide*.

The archecker Utility

When you use the **onbar -v** command to verify ON-Bar backups, it calls the **archecker** utility.

Fast Recovery

Fast recovery is an automatic procedure that restores the database server to a consistent state after it goes offline under uncontrolled conditions. Fast recovery also rolls forward all committed transactions since the last checkpoint and rolls back any uncommitted transactions.

When the database server starts up, it checks the *physical log*, which contains pages that have not yet been written to disk. If the physical log is empty, the database server was shut down in a controlled fashion. If the physical log is *not* empty, the database server automatically performs *fast recovery*.

For information about fast recovery, see the *IBM Informix Administrator's Guide*.

Mirroring

When you use disk mirroring, the database server writes each piece of data to two locations. Mirroring is a strategy that pairs a *primary chunk* of one storage space with an equal-sized *mirrored chunk*. Every write to the primary chunk is automatically accompanied by an identical write to the mirrored chunk. If a failure occurs on the primary chunk, mirroring lets you read from and write to the mirrored chunk until you can recover the primary chunk, all without interrupting user access to data.

It is recommended that you mirror the following data:

- Root dbspace
- Dbspaces that contain the physical log and logical-log files
- Frequently queried data

For information about mirroring, see the *IBM Informix Administrator's Guide*.

Data Replication

Data replication generates and manages multiple copies of data at one or more sites, which allows an enterprise to share corporate data throughout its organization. Data replication provides a backup system in case of a catastrophic failure.

High-Availability Clusters

IBM Informix Dynamic Server provides several high-availability cluster configuration options. A high-availability cluster consists of a primary server and one or more secondary servers on which data from the primary server is replicated. Data replication provides a way to duplicate database objects at more than one distinct site. Dynamic Server provides several secondary server configuration options, including:

- High-availability data replication (HDR) secondary server, which provides synchronous data replication for Dynamic Server. Use an HDR secondary server if you require a hot standby. Configuring an HDR secondary server provides a way to maintain a backup copy of the entire database server that applications can access quickly in the event of a catastrophic failure of the primary server.
- Shared-Disk (SD) secondary server is a server that shares disk space with a primary server. The primary server has write access to a disk or disk array, while all SD secondary servers have updatable secondary access. An SD secondary server does not maintain a copy of the physical database on its own disk space; rather, it shares disks with the primary server.

- Remote Standalone (RS) secondary server is a server that is updated asynchronously from the primary server. RS secondary servers can be geographically distant from the primary server, serving as remote backup servers in disaster-recovery scenarios. Each RS secondary server maintains a complete copy of the database, with updates transmitted asynchronously from the primary server over secure network connections.

For information about HDR secondary servers, RS secondary servers, and SD secondary servers, see the *IBM Informix Administrator's Guide*.

Any of the above configurations can be combined with Enterprise Replication. For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Enterprise Replication

Enterprise Replication captures transactions to be replicated throughout the enterprise. On the source database server, Enterprise Replication reads the logical log and transmits each transaction to the target database servers. At each target database server, Enterprise Replication receives and applies each transaction to the appropriate databases and tables. Enterprise Replication can be combined with other data replication solutions.

For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Types of Data That You Can Replicate

Table 3-1 shows the types of data that you can replicate with data replication or Enterprise Replication.

Table 3-1. Data Types that high-availability clusters and ER Replicate

Data Type	Cluster Support	ER Support
Atomic data types such as numeric, character, varying character, time, Boolean	Yes	Yes
Simple large objects in dbspaces	Yes	Yes
Simple large objects in blobspaces	No	Yes
User-defined data types	Yes	Yes ¹
DataBlade types (text, image, video, web, and geodetic)	Yes	Yes
Smart large objects	Yes	Yes ²

Notes:

1. To replicate user-defined data types, the required **streamwrite()** and **streamread()** functions must exist. For information on writing and registering support functions, see the section on writing Enterprise Replication stream support functions in the *IBM Informix DataBlade API Programmer's Guide*.
2. For information on restrictions for replicating smart large objects, see the *IBM Informix Dynamic Server Enterprise Replication Guide*.

Database Server Security

The *IBM Informix Security Guide* contains information that your database administrator needs to know.

Database security features include the following types of tasks:

- Secure server utilities
- Encrypt data across the network

- Encrypt column-level data
- Secure connections
- Control user privileges
- Control user access to data

Auditing Database Events

You can audit database events by tracking activities that users perform on particular objects at distinct times. You can use this information to monitor database activity for suspicious use, detect unauthorized access attempts, assess potential security damage, unscrupulous users, and provide evidence of database server abuse.

For information on auditing, see the *IBM Informix Security Guide*.

Informix RDBMS Features

This section discusses the database components and extensibility features.

Structured Query Language (SQL)

You can use SQL statements to retrieve, insert, update, and delete data from a database. To retrieve data from a database, you perform a query, which is a SELECT statement that specifies the rows and columns to be retrieved from the database.

You can write programs that exchange data with the database server. You can also write programs that take data from any source in any format, prepare it, and insert it into the database.

Use Informix ESQL/C to embed SQL statements directly into a C program. DB–Access lets you execute SQL statements interactively. Use JDBC to embed SQL statements directly into a Java program.

For information about database management, see the *IBM Informix Database Design and Implementation Guide*. For information about how to create and use SQL, see the *IBM Informix Guide to SQL: Tutorial* and the *IBM Informix Guide to SQL: Syntax*. For information about embedded SQL, see the *IBM Informix ESQL/C Programmer's Manual* and *J/Foundation Developer's Guide*. For information about how to use DB–Access, see the *IBM Informix DB–Access User's Guide*.

Stored Procedure Language (SPL)

Informix Stored Procedure Language (SPL) is an extension to SQL that provides flow control such as looping and branching. Consider using SPL procedures and routines for SQL-intensive tasks. An *SPL procedure* is a routine written in SPL and SQL that does not return a value. An *SPL function* is a routine written in SPL and SQL that returns a single value, a value with a complex data type, or multiple values.

You can write user-defined routines in the SPL, C, and Java languages and store them in the database.

For information on how to create and use SPL routines, see the *IBM Informix Guide to SQL: Tutorial*. For the syntax diagrams of SPL statements, see the *IBM Informix Guide to SQL: Syntax*. For performance aspects, see your *IBM Informix Performance Guide*.

System Catalog Tables

Sometimes called the “data dictionary,” the *system catalog* tables describe the structure of the database. The database server automatically generates the system catalog tables when you create a database. Each system catalog table contains specific information about elements in the database.

System catalog tables track database objects such as the following:

- Tables, views, sequences, synonyms, and sequence objects
- Columns, constraints, indexes, and fragments
- Triggers
- Procedures, functions, routines, and associated messages
- Authorized users and privileges
- User-defined routines
- Data types and casts (IDS)
- Aggregate functions (IDS)
- Access methods and operator classes (IDS)
- Inheritance relationships (IDS)
- External optimizer directives (IDS)

For information about the system catalog, see the *IBM Informix Guide to SQL: Reference*.

Data Types

Every column in a table is assigned a data type. The data type precisely defines the values that you can store in that column. Dynamic Server supports the data types that Figure 3-1 shows.

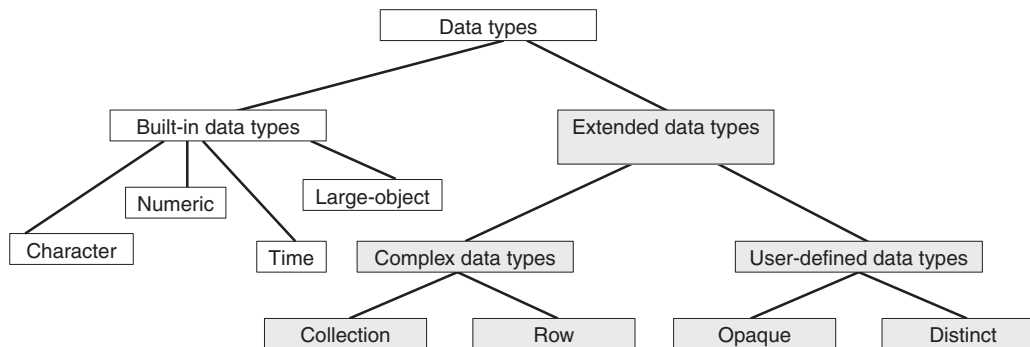


Figure 3-1. Overview of Supported Data Types

For a description of the data types and data type conversions, see the *IBM Informix Guide to SQL: Reference*. For information about how to choose data types for your relational or object-relational database, see the *IBM Informix Database Design and Implementation Guide*. For information on how to extend existing data types, create new casts, and define new data types for a database, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

Table 3-2 describes the data types that you can define for a database.

Table 3-2. Data Types

Data Type	Explanation	Examples
Built-in data types	<p>Fundamental data types that cannot be broken into smaller pieces</p> <p>Serve as building blocks for other data types.</p>	<ul style="list-style-type: none"> • BIGINT • BIGSERIAL • BLOB • BOOLEAN • BYTE • CHAR(<i>n</i>) • CHARACTERVARYING(<i>m,r</i>) • CLOB • DATE • DATETIME • DECIMAL or NUMERIC(<i>p,s</i>) • DOUBLE PRECISION • FLOAT • IDSSECURITYLABEL • INTEGER • INTERVAL • LVARCHAR(<i>m</i>) • MONEY(<i>p,s</i>) • NCHAR(<i>n</i>) • NVARCHAR(<i>m,r</i>) • REAL or SMALLFLOAT • SERIAL • SERIAL8 • SMALLINT • TEXT • VARCHAR(<i>n,r</i>)
Complex data types	<p>Combination of other data types</p> <p>An SQL statement can access individual components within the complex type.</p>	
Collection types	<p>Complex data types</p> <p>Include groups of elements of the same data type, which can be any built-in or complex data type.</p>	<ul style="list-style-type: none"> • SET • LIST • MULTISSET
Row types	<p>Complex data types</p> <p>Include groups of related data <i>fields</i> of any data type that form a template for a record.</p>	<p>Named row type</p> <p>Unnamed row type</p>
User-defined data types	Include distinct and opaque types	
Distinct data types	<p>Have the same internal structure as existing data types</p> <p>They have distinct names and functions that make them different from the source type.</p>	<p>CREATE DISTINCT TYPE birthday AS DATE</p> <p>The data type is birthday.</p>

Table 3-2. Data Types (continued)

Data Type	Explanation	Examples
Opaque data types	User-defined types The internal structure is not known to the database server.	CREATE OPAQUE TYPE fixlen_typ (INTERNALLENGTH=8, CANNOTHASH) The data type is fixlen_typ .
DataBlade data types	New data types from IBM Informix DataBlade modules A DataBlade module is a collection of functions that describe special-purpose data types and all of their support functions. A DataBlade module can contain any or all of the previously described data types.	Examples of DataBlade modules include: IBM Informix <ul style="list-style-type: none"> • Image Foundation DataBlade module • Informix Excalibur Text Search DataBlade Module • IBM Informix Geodetic DataBlade Module • IBM Informix Spatial DataBlade module • IBM Informix TimeSeries DataBlade Module • IBM Informix TimeSeries Real-Time Loader DataBlade module • Informix Video Foundation DataBlade Module • IBM Informix Web DataBlade Module

Application Types

Two main classes of applications operate on data in an Informix database:

- Online transaction processing (OLTP) applications
- Decision-support system (DSS) applications

OLTP Applications

OLTP applications are often used to capture new data or update existing data. An order-entry system is a typical example of an OLTP application.

OLTP applications have the following characteristics:

- Transactions that involve small amounts of data
- Indexed access to data
- Many users
- Frequent queries and updates
- Fast response times

DSS Applications

DSS applications often report on or consolidate data that OLTP operations have captured over time. These applications provide information that is often used for accounting, strategic planning, and decision making. Data in the database is typically queried but not updated during DSS operations. Typical DSS applications include payroll, inventory, and financial reports.

For more information on how to manage DSS systems, see your *IBM Informix Performance Guide*.

Database Support

Dynamic Server supports the following types of databases:

- Relational
- ANSI compliant
- Object-relational
- Dimensional (data warehouse)
- Distributed

Relational Databases

Relational database management systems (RDBMS) are designed for online transaction processing (OLTP), although you can use an RDBMS for DSS processing. An RDBMS focuses on high-speed, short-running queries and transactions on the following types of simple data:

- Integer
- Floating-point number
- Character string, fixed, or variable length
- Date and time, time interval
- Numeric and decimal
- Simple large objects (TEXT and BYTE data)

For information about relational databases, see the *IBM Informix Database Design and Implementation Guide* and the *IBM Informix Guide to SQL: Syntax*.

ANSI-Compliant Databases

You create an ANSI-compliant database when you use the MODE ANSI keywords in the CREATE DATABASE statement. You can use the same SQL statements with both ANSI-compliant databases and non-ANSI-compliant databases. You might want to create an ANSI-compliant database for the following reasons:

- Privileges and access to objects
ANSI rules govern privileges and access to objects such as tables and synonyms.
- Name isolation
The ANSI table-naming scheme allows different users to create tables in a database without having to worry about name conflicts.
- Transaction isolation
- Data recovery
ANSI-compliant databases enforce unbuffered logging and automatic transactions for Dynamic Server.

For information about ANSI-compliant databases, see the *IBM Informix Database Design and Implementation Guide* and the *IBM Informix Guide to SQL: Syntax*.

Object-Relational Databases

Object-relational database management systems (ORDBMS) combine relational and object-oriented capabilities. Choose an object-relational database if you need greater flexibility in the types of data that the database server can store and manipulate. An example of an object-relational database is an online store catalog.

You can extend the capability of the database server by defining new data types and user-defined routines (UDRs) that let you store, access, and manage images, audio, video, large text documents, and so on.

An object-relational database supports the following data types and extensibility:

- Alphanumeric data (such as character strings, integers, decimal, floating point, and date)
- Simple large objects (TEXT and BYTE data types)
- Smart large objects (BLOB and CLOB data types)
- User-defined types (opaque and distinct types)
- Complex data types (composites of existing data types)
- User-defined routines
- Operator functions
- User-defined casts
- User-defined aggregates
- Type and table inheritance
- DataBlade modules
- User-defined virtual processors
- User-defined access methods (see “Access Methods” on page 3-19)

For information about object-relational databases, see the *IBM Informix Database Design and Implementation Guide* and *IBM Informix Guide to SQL: Syntax*. For more information about extending the database server, see *IBM Informix User-Defined Routines and Data Types Developer's Guide* and *J/Foundation Developer's Guide*.

Simple and Smart Large Objects

The database server supports *simple large objects* and *smart large objects* for storing large chunks of binary or text data in a database. A *large object* is a data object that is logically stored in a table column but physically stored independently of the column. Large objects are stored separately from the table because they typically store very large amounts of data.

For more information on simple and smart large objects, see the *IBM Informix Guide to SQL: Reference* and *IBM Informix Guide to SQL: Tutorial*.

Simple Large Objects (TEXT and BYTE Data Types): The database server stores simple large objects in a dbspace or blobspace. Simple large objects do not support random access to the data. When you transfer a simple large object between a client application and the database server, you must transfer the entire BYTE or TEXT value.

Smart Large Objects (CLOB and BLOB Data Types): You can use smart large objects to store user-defined types such as video and audio clips, pictures, large text documents, and spatial objects such as drawings and maps.

The database server stores smart large objects in *sbspaces*. You can control the logging characteristics of smart large objects and sbspaces independently from the logging characteristics of the database. Use a *temporary sbpace* to store *temporary smart large objects* without any logging.

Programmers can use functions similar to UNIX and Windows functions to read, write, and seek smart large objects. Dynamic Server provides the smart-large-object API in the DataBlade API and the Informix ESQL/C programming interface.

For information on sbspaces, see the *IBM Informix Administrator's Guide*. For information on how to create an sbspaces, see the discussion of **onspaces** in the *IBM Informix Administrator's Reference*. For information on how to calculate space and tune sbspaces, see your *IBM Informix Performance Guide*. For information on how to access a simple large object or a smart large object from a client application, see the *IBM Informix ESQ/C Programmer's Manual*. For information on using the DataBlade API with smart large objects, see the *IBM Informix DataBlade API Programmer's Guide*.

User-Defined Data Types

You can create *user-defined data types* (UDTs) to extend the database server and provide greater flexibility in the types of data that you can store and manipulate. User-defined data types can be opaque or distinct.

You create a *distinct data type* with the CREATE DISTINCT TYPE statement. A distinct type has the same internal structure as an existing data type. However, it has a distinct name and therefore distinct functions that make it different from its source type. Once you create the distinct type, you can use it anywhere that other data types are valid.

You create and register an *opaque data type* with the CREATE OPAQUE TYPE statement. An opaque type stores a single value and cannot be divided into components by the database server. It is implemented as a structure and a set of routines that allow the database server to support the data type.

Complex Data Types

A *complex data type* is a composite of existing data types. It can be a named row type, unnamed row type, or collection type. For example, you might create a complex type whose components include built-in types, opaque types, distinct types, or other complex types.

A *collection type* is a group of elements of the same data type. Collection data types let you store and manipulate collections of data within a single row of a table.

A *row type* is a sequence of one or more fields. Each field has a name and a data type. The fields of a row are comparable to the columns of a table, but there are important differences. You cannot define a default value for a field, you cannot define constraints on a field, and you cannot use fields with tables, only with row types. Row types can be named or unnamed:

- A *named row type* is a group of fields that are defined under a single name. A *field* refers to a component of a row type. Once you create a named row type, the name that you assign to the row type represents a unique data type within the database.
- An *unnamed row type* is a group of fields that are defined by their structure. Unlike a named row type, which you can use to define a table, you cannot use an unnamed row type to define a table. Use an unnamed row type to define a column, field, or variable.

User-Defined Routines

A *routine* is a collection of program statements that perform a particular task. A *user-defined routine* (UDR) is a routine that you can define and that can be invoked within an SQL statement or within another UDR. A UDR can either return values or not, as follows:

- A *user-defined function* returns one or more values and therefore can be used in SQL expressions.

- A *user-defined procedure* is a routine that optionally accepts a set of arguments but does not return any values. A procedure cannot be used in SQL expressions because it does not return a value.

The database server supports UDRs written in the following languages:

- *Stored Procedure Language (SPL)*, a language that is internal to the database server
- *External languages*, such as the C or Java languages

For information on implementing user-defined routines, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

Operator Functions

An *operator function* is an SQL-invoked function that has a corresponding operator symbol (such as '=' or '+'). These operator symbols are used within expressions in an SQL statement.

The database server provides operator functions for most built-in data types. You can extend an existing operator to operate on a user-defined data type.

For information on extended operations, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

User-Defined Casts

A *cast* performs a conversion between two data types. The database server provides casts between the built-in data types. For example, when you add an integer value to a decimal value, the database server performs a cast to change the integer into a decimal so that it can perform the addition.

You can write user-defined cast functions to convert between an existing data type and an extended data type that you create.

For information on implementing user-defined casts, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

Inheritance

Inheritance lets you define objects (types and tables) that acquire the properties of other objects and add new properties that are specific to the object that you define.

User-Defined Aggregates

Use a *user-defined aggregate* (UDA) to perform any kind of aggregate computation on a column, such as the average or the count. You can either create a user-defined aggregate or extend an existing aggregate for extended data types.

For the SQL syntax to create and drop UDAs, see the *IBM Informix Guide to SQL: Syntax*. For information on using UDAs, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

User-Defined Virtual Processors

You can designate a user-defined virtual processor to run DataBlade modules or UDRs written in the C language. Designate a Java virtual processor to run UDRs written in the Java language.

For information on virtual processors, see the *IBM Informix Administrator's Guide* and *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

DataBlade Modules

IBM and other vendors package some data types and their access methods into DataBlade modules (shared class libraries) that you can add to the database server to store and access nontraditional data types such as two-dimensional spatial objects (lines, polygons, ellipses, and circles), 3D images, sound, video, electronic documents, HTML pages, and time-series data. A DataBlade module might also provide new types of access to large text documents, including phrase matching, fuzzy searches, and synonym matching.

You can do the following:

- Add an IBM Informix or third-party DataBlade module, which is a pre-packaged custom data type.
- Create your own DataBlade module with the Informix DataBlade Developers Kit.

For information on how to work with and create your own DataBlade modules, see the *IBM Informix DataBlade API Programmer's Guide*, the *IBM Informix DataBlade API Function Reference*, and the *IBM Informix DataBlade Developers Kit User's Guide*.

Dimensional Databases

Dynamic Server supports *data warehouses* and *data marts*. This typically involves a *dimensional* database that contains large stores of historical data. The databases that track your grocery purchases and voting trends in your state are examples of data warehouses.

A dimensional database is optimized for data retrieval and analysis. The data is stored as a series of snapshots, in which each record represents data at a specific time. Existing records in a dimensional database are updated infrequently. This type of informational processing is known as online analytical processing (OLAP) or decision-support processing.

A data-warehousing environment can store data in one of the following forms:

- Data warehouse
A database that is optimized for data retrieval
Data is not stored at the transaction level; some level of data is summarized.
- Data mart
A subset of a data warehouse that is stored in a smaller database and that is oriented toward a specific purpose or subject rather than enterprise-wide strategic planning
A data mart can contain operational data, summarized data, spatial data, or metadata.
- Operational data store
A subject-oriented system that is optimized for looking up one or two records at a time for decision making
An operational data store is a hybrid form of data warehouse that contains timely, current, integrated information. This data can serve as the common source of data for data warehouses.
- Repository
A repository combines multiple data sources into one normalized database
The records in a repository are updated frequently. Data stored in a repository is operational, not historical.

For details on how to plan, build, and implement a dimensional database, see the *IBM Informix Database Design and Implementation Guide*.

Distributed Queries and Multiphase Transactions

Dynamic Server supports distributed queries for transactions that involve only built-in data types, and certain built-in opaque and DISTINCT types, across the following server instances:

- Multiple databases of a single server instance (cross-server distributed queries)
- Multiple database server instances (cross-database distributed queries)

To issue a distributed query, a client application connects to a single database server, called the *local database server*, and specifies a database, called the *local database*. By default, all the database objects that you reference come from the local database.

All other databases are *external databases*. All other database servers are *remote database servers*. A database on a remote database server is an *external remote database*.

When the external database is on the same database server as the local database, you must qualify the object name with the external database name (for example, **salesdb:contacts**). When the external database is on a remote database server, you must qualify the object name with the remote database server name and the external remote database name (**salesdb@distantserver:contacts**).

The database server supports two multiphase protocols, *two-phase commit* and *heterogeneous commit*, to process transactions that span multiple database servers.

For information about using distributed queries, see *IBM Informix User-Defined Routines and Data Types Developer's Guide*. For information about two-phase commit and heterogeneous commit protocols, see the *IBM Informix Administrator's Guide*. For information about a specific IBM Informix Enterprise Gateway product, see the appropriate *IBM Informix Enterprise Gateway User Manual*.

Access Methods

An *access method* is a set of database server functions that a database server uses to access and manipulate a table or an index. Dynamic Server supports *primary* and *secondary access methods*. You can write routines that provide R-tree indexing and custom primary and secondary access methods.

Primary Access Methods

The *primary access method* handles storage and retrieval of a particular data type in a table. If the primary access method does not handle a particular data type, the database server cannot access values of that type. Dynamic Server provides all of the necessary routines for accessing the built-in data types.

For information on how to use primary access methods, see the *IBM Informix Guide to SQL: Syntax* and *IBM Informix User-Defined Routines and Data Types Developer's Guide*.

Secondary Access Methods

The *secondary access method* handles all indexing operations for a particular data type. If the *operator class* of a secondary access method does not handle a particular data type, you cannot build an index on that data type.

Dynamic Server provides two built-in secondary access methods:

- Generic B-trees
- R-trees

Generic B-Tree Indexes

A B-tree index organizes index information. A B-tree index is arranged as a hierarchy of pages. Dynamic Server uses a B-tree index for the following values:

- Columns that contain built-in data types (known as a *traditional B-tree index*)
Built-in data types include CHARACTER, DATETIME, INTEGER, FLOAT, and so forth.
- One-dimensional user-defined data types (known as a *generic B-tree index*)
- Values that a user-defined function returns (known as a *functional index*)

For more information on B-trees and functional indexes, see your *IBM Informix Performance Guide*.

R-Tree Indexes

The *R-tree* indexing structure supports spatial data. An R-tree index uses a *bounding box*, which is a set of coordinates that contains one or more objects and supports spatial data (two-dimensional, three-dimensional, and so on). An object can theoretically belong to more than one bounding box. An R-tree index is useful for searches on multidimensional data.

For information about R-trees, see the *IBM Informix R-Tree Index User's Guide*.

User-Defined Primary Access Methods

Dynamic Server supports *external spaces* (*extspaces*), which are storage spaces that the database server does not manage directly. Use **onspaces -c -x** to specify an external space as the storage space for a table for which you create a primary access method.

You can access the following types of data with a primary access method:

- Database tables from other vendors
- Data stored in sequential files
- Remote data stored across a network

For information on creating extspaces, see the *IBM Informix Administrator's Guide*, and *IBM Informix Administrator's Reference*. For information on how to create primary access methods, see the *IBM Informix Virtual-Table Interface Programmer's Guide*.

User-Defined Secondary Access Methods

In many cases, index data is stored outside the Informix dbspace. However, you can build an access method for data stored as a large object in an sbospace. The database server can use a virtual index transparently to access data in an Informix table. Use this method to create an alternative indexing strategy for specialized data types.

For information on how to create secondary access methods, see the *IBM Informix Virtual-Index Interface Programmer's Guide*.

Chapter 4. Installing, Administering, and Tuning the Database Server

In This Chapter	4-1
Database Server Users	4-1
Planning, Installing, and Configuring the Database Server	4-1
Administering the Database Server	4-4
Monitoring Performance	4-7
Troubleshooting the Database Server.	4-8

In This Chapter

This chapter describes the tasks that a database server administrator is likely to perform and where to find information on those tasks throughout the documentation set. The task matrixes in this book have the following columns:

- **If You Want To.** This column describes a task you might want to perform.
- **Publication.** This column lists the book that contains information to help you perform the task.

Database Server Users

Table 4-1 shows the major groups of database server users.

Table 4-1. Database Server Users

User	Duties
Database administrator (DBA)	A DBA is primarily responsible for creating, managing, and controlling access for databases.
Database server administrator (DBSA)	The database server administrator is responsible for the installation, configuration, maintenance, administration, and operation of the database server that might manage many individual databases.
Performance specialist	The performance specialist optimizes and tunes the database server and query performance.
Programmers and application developers	Programmers and application developers develop applications, DataBlade modules, and user-defined routines in C, C++, or Java.
Operator	The operator is responsible for backing up and restoring databases and for carrying out routine database server administration tasks.
Database user	Database users access, insert, update, and manage information in databases with SQL, which is often embedded in a client application.

Planning, Installing, and Configuring the Database Server

When you begin working with a new database server, you need to perform the following tasks:

- Set up and configure system hardware and software.
- Install the database server and client applications.
- Migrate data from an earlier version of the database server (if needed).

- Configure the environment.
 - Set required environment variables.
 - Prepare connectivity files.
 - Prepare the configuration file.
 - Allocate and initialize disk space.
- Choose a database type.
- Create the demonstration database (optional).

Table 4-2 describes the planning, installation, and configuration tasks.

Table 4-2. Planning, Installation, and Configuration Tasks

If You Want To	Publication
Learn about the new features in Dynamic Server.	Chapter 2, “Using New Features in Dynamic Server,” on page 2-1
Acquaint yourself with terms used in IBM Informix publications.	<i>IBM Informix Glossary</i>
Plan a database server installation.	<i>IBM Informix Administrator’s Guide</i>
Plan to migrate to Dynamic Server from an earlier database server version.	<i>IBM Informix Migration Guide</i>
Plan and configure: <ul style="list-style-type: none"> • Operating system • Hardware and system software upgrades • Network capacity • Integration with other vendor products and applications • Disk and storage media 	<ul style="list-style-type: none"> • <i>IBM Informix Dynamic Server Installation Guide for Windows, IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X, or the Quick Beginnings for IBM Informix Dynamic Server Express Edition;</i> • <i>IBM Informix Administrator’s Guide</i>
Determine optimal memory configuration.	<ul style="list-style-type: none"> • <i>IBM Informix Performance Guide</i>
Determine optimal disk layout and striping.	<ul style="list-style-type: none"> • System documentation • Machine notes
Install Dynamic Server on UNIX or Linux: <ul style="list-style-type: none"> • Typical installation • Silent installation • Custom installation • Modify an installation • Uninstall • Multiple residency 	<ul style="list-style-type: none"> • <i>IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X</i> • <i>Quick Beginnings for IBM Informix Dynamic Server Express Edition</i>

Table 4-2. Planning, Installation, and Configuration Tasks (continued)

If You Want To	Publication
Install Dynamic Server on Windows: <ul style="list-style-type: none"> • Typical installation • Silent installation • Custom installation • Modify an installation • Uninstall • Multiple residency • Cluster installation 	<ul style="list-style-type: none"> • <i>IBM Informix Dynamic Server Installation Guide for Windows</i> • <i>Quick Beginnings for IBM Informix Dynamic Server Express Edition</i>
Initialize the database server. Manage database server operating modes.	<i>IBM Informix Administrator's Guide</i>
Test the database server connection with DB–Access.	<i>IBM Informix DB–Access User's Guide</i>
Install and configure client applications.	<i>IBM Informix Client Products Installation Guide</i>
Install and configure DataBlade modules (optional).	<i>IBM Informix DataBlade Module Installation and Registration Guide</i>
Install and configure Informix MaxConnect (optional).	<i>IBM Informix MaxConnect User's Guide</i>
Configure the database server manually: <ul style="list-style-type: none"> • Set environment variables. • Set ONCONFIG parameters. • Configure J/Foundation (optional). • Configure client/server connectivity. • Set up multiple instances of the database server. • Test the database server configuration. 	<i>IBM Informix Administrator's Guide</i>
Use the OpenAdmin Tool for IDS (OAT) to configure the database server.	OAT online help OAT <i>How do I?</i> document
Use IBM Informix Server Administrator to configure the database server.	ISA online help
Create dbspaces, blobspaces, and sbspaces.	<i>IBM Informix Administrator's Guide</i>
Resolve incorrect chunk permissions and ownership.	
Design and implement logical and physical logs.	
Implement mirroring.	
Configure the ON–Bar or ontape backup and restore system.	<i>IBM Informix Backup and Restore Guide</i>
Set up IBM Informix Storage Manager.	<i>IBM Informix Storage Manager Administrator's Guide</i>
Set up the storage volumes and storage devices.	
Set up a third-party storage manager (optional).	Your storage-manager documentation
Design and set up the Enterprise Replication system.	<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>
Design and set up a High-Availability Data Replication (HDR) system.	<i>IBM Informix Administrator's Guide</i>

Table 4-2. Planning, Installation, and Configuration Tasks (continued)

If You Want To	Publication
Prepare the old database server version for migration.	<i>IBM Informix Migration Guide</i>
Migrate to Dynamic Server from an earlier database server version.	
Move data among different physical equipment (computer and storage devices) and different operating systems.	
Move data among database servers that have different language support.	
Work with these utilities: dbexport , dbimport , dbload , dbschema , onload , onunload , onmode -b .	
Revert from Dynamic Server to an earlier database server version.	
Interpret error messages.	<i>IBM Informix Error Messages</i> or finderr utility

Administering the Database Server

The database server administrator should routinely perform the following tasks after the database server is initialized:

- Prepare the operating system to automatically start and stop the database server when the system is shut down or rebooted.
- Back up and restore storage spaces (dbspaces, blobspaces, and sbspaces) and logical logs.

When you plan your backup schedule, consider the availability of backup devices and operators to perform backups.

- Check that users have set the correct environment variables.
- Review the database server configuration parameters.
- Transfer data that was created on other Informix database servers.

Table 4-3 on page 4-5 lists the administration tasks and where to find information on those tasks.

Table 4-3. Administration Tasks

If You Want To	Publication
Monitor an Informix database server.	<i>IBM Informix Administrator's Guide</i>
Configure client/server connections.	
Manage virtual processors, shared memory, and storage spaces.	
Manage temporary space usage and table extents.	
Manage database-logging status, logical-log files, and the physical log.	
Monitor and manage sbspaces.	
Resolve long transaction problems.	
Perform fast recovery and checkpoints.	
Perform mirroring operations.	
Verify database consistency using oncheck commands.	
Use High-Availability Data Replication.	
Understand two-phase and heterogeneous commit protocols.	
Recover manually from a failed two-phase commit.	
Use OpenAdmin Tool for IDS to administer and monitor the database server.	OpenAdmin Tool online help
Use the following utilities to perform administrative tasks:	<i>IBM Informix Administrator's Reference</i>
• oncheck	
• ondblog	
• oninit	
• onlog	
• onmode	
• ON-Monitor	
• onparams	
• onspaces	
• onstat	
Locate information on the configuration parameters.	<i>IBM Informix Administrator's Reference</i>
Use the SMI tables of the sysmaster database to monitor the database server.	<i>IBM Informix Administrator's Reference</i>
Interpret logical-log records and message-log messages.	
Understand database server disk structures and storage.	
See a list of the files that the database server uses.	
Work with event alarms.	

Table 4-3. Administration Tasks (continued)

If You Want To	Publication
Use the ON-Bar or ontape utility.	<i>IBM Informix Backup and Restore Guide</i>
Back up and restore storage spaces and logical logs.	
Use the archecker utility to verify backed-up data.	
Perform an external backup and restore.	
Perform a table-level restore.	
Set up HDR with external backup and restore.	<i>IBM Informix Storage Manager Administrator's Guide</i>
Connect your database server to storage devices for ON-Bar backup and restore operations.	
Issue ISM commands.	
Manage backup media and storage devices.	
Track the location of all backup data.	
Move backup data through a managed life cycle.	
Provide disaster recovery for a database server instance.	
Perform an imported restore to a database server on another computer.	
Use the ipload , onpladm , and onpload utilities to load or unload large quantities of data to or from an Informix database.	
Use the High-Performance Loader (HPL) GUI.	
Move data to a different computer or configuration.	<i>IBM Informix High-Performance Loader User's Guide</i>
Alter the schema of a table.	
Encrypt network data	
Encrypt column-level data.	
Prevent unauthorized connections to the database server.	
Control access to database objects.	<i>IBM Informix Security Guide</i>
Control access to data.	
Detect unusual user actions and unwanted activities and identify the perpetrators.	
Detect unauthorized access attempts.	
Assess potential security compromises.	
Use the secure-auditing utilities (onaudit , onshowaudit) to set up, administer, and interpret audit trails.	<i>IBM Informix Optical Subsystem Guide</i>
Use the Optical Subsystem interface for an optical storage subsystem to store TEXT and BYTE data (simple large objects) on optical platters (WORM optical media).	
Use SQL statements to store and retrieve data to and from the optical storage subsystem.	

Table 4-3. Administration Tasks (continued)

If You Want To	Publication
Use the IBM Informix SNMP subagent to extract information from an Informix database server and pass that information to a network manager.	<i>IBM Informix SNMP Subagent Guide</i>
Design, define, monitor, and control your Enterprise Replication system.	<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>
Set up locales for different languages, cultural conventions, and code sets.	<i>IBM Informix GLS User's Guide</i>

Monitoring Performance

After the database server is up and running, the database server administrator or performance specialist is responsible for maintaining the optimum performance of the database server and database applications, as follows:

- Monitor system resources that are critical to performance.
- Identify database activities that affect these critical resources.
- Identify and monitor queries that are critical to performance.
- Use the database server utilities for performance monitoring and tuning.
- Optimize query execution.
- Eliminate performance bottlenecks in the following ways:
 - Balance the load on system resources.
 - Adjust the configuration of your database server.
 - Adjust the arrangement of your data.
 - Allocate resources for decision-support queries.
 - Create indexes to speed up retrieval of your data.

Table 4-4 lists performance-related tasks and where to find information on those tasks.

Table 4-4. Performance Tuning Tasks

If You Want To	Publication
Use different types of tables (STANDARD, RAW, TEMP).	<i>IBM Informix Administrator's Guide</i>
Use the onstat -g utilities to monitor database server performance.	<i>IBM Informix Administrator's Reference</i>
Improve backup and restore performance.	<i>IBM Informix Backup and Restore Guide</i>
Query the system catalog tables.	<i>IBM Informix Guide to SQL: Reference</i>
Use OpenAdmin Tool for IDS performance tools.	OpenAdmin Tool online help

Table 4-4. Performance Tuning Tasks (continued)

If You Want To	Publication
Adjust the database server configuration.	IBM Informix Performance Guide
Allocate resources for DSS or OLTP systems.	
Balance the load on system resources.	
Collect performance statistics.	
Control the placement and size of tables and table extents.	
Create and manage indexes to speed up retrieval of data.	
Design and use parallel database queries (PDQ).	
Eliminate database server performance bottlenecks.	
Fragment tables for improved performance.	
Identify and monitor queries that are critical to performance.	
Improve checkpoint performance and manage LRU queues.	
Improve the performance of a query.	
Manage data distributions.	
Monitor critical system resources (CPU, memory, disk, virtual processors).	
Monitor and track locking and isolation levels.	
Optimize the disk layout.	
Tune buffer cache.	
Use case studies to tune performance.	
Use optimizer directives and SET EXPLAIN to optimize query plans.	
Use query drill down.	
Use secondary access methods such as B-trees.	
Use the onperf utility for performance monitoring and tuning.	
Use the SQL statement cache.	
Use UPDATE STATISTICS.	
Write complex SQL statements, including outer joins and subqueries.	

Troubleshooting the Database Server

Usually, the database server runs smoothly, but when something goes wrong or a puzzling error message displays, an array of diagnostic tools is available to help you solve the problem. Technical Support also can assist you in troubleshooting and fixing problems with Dynamic Server.

Table 4-5 on page 4-9 describes the diagnostic tools available for troubleshooting database operations and the database server.

Table 4-5. Troubleshooting Tasks

If You Want To	Publication
Use the onstat -g utilities to diagnose database server problems.	<i>IBM Informix Administrator's Reference</i>
Use the onmode -I option to collect diagnostic information.	
Use event alarms to automatically trigger administrative actions.	
Find corrective actions to unnumbered error messages.	
Collect diagnostic dumps using the DUMP* configuration parameters.	<i>IBM Informix Backup and Restore Guide</i>
Use the archecker utility to verify and diagnose problems with backups. Find corrective actions to ON-Bar return codes.	
Find corrective actions to numbered error messages and ON-Bar messages.	<i>IBM Informix Error Messages</i>
Fix data replication problems.	<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>

Chapter 5. Designing, Maintaining, and Extending the Database

In This Chapter	5-1
Designing, Developing, and Extending the Database	5-1
Developing Application Programs that Access the Database	5-3

In This Chapter

This chapter describes the tasks that database administrators (DBA) and application developers are likely to perform and where to find information on those tasks.

Designing, Developing, and Extending the Database

Table 5-1 lists the tasks for designing, developing, and extending the database.

Table 5-1. Database Tasks

If You Want To	Publication
Work with the tables in the sysmaster database.	<i>IBM Informix Administrator's Reference</i>
Place tables on disk.	<i>IBM Informix Performance Guide</i>
Estimate table size and manage table extents.	
Modify tables (truncate, alter, modify columns, load, attach fragments).	
Denormalize data to improve performance.	
Create and manage B-tree indexes.	
Work with specialized indexes (R-tree and secondary access methods).	
Set appropriate lock modes and monitor locks.	
Design a fragmentation strategy (round-robin or expression-based).	
Fragment indexes and temporary tables.	
Use the WHERE clause and joins to filter queries.	
Find corrective actions to error messages.	<i>IBM Informix Error Messages</i>

Table 5-1. Database Tasks (continued)

If You Want To	Publication
Design a database (choose whether to implement the relational, object-relational, or dimensional database model).	<i>IBM Informix Database Design and Implementation Guide</i>
Build a relational or object-relational database: <ul style="list-style-type: none"> • Define the data objects. • Create an entity-relationship diagram. • Normalize the data. • Create and populate the database. 	
Build and implement a dimensional database for data warehousing.	
Choose data types for the database.	
Set up check constraints and referential constraints.	
Determine the primary keys and foreign keys in tables.	
Extend a database with user-defined casts.	
Understand type and table inheritance.	
Grant and limit access to a database.	
Use views and privileges.	
Define a fragmentation strategy or distribution schema.	
Invoke the DB–Access utility.	<i>IBM Informix DB–Access User’s Guide</i>
Connect to or create one or more databases and transfer data between a database and external text files.	
Display information about databases and verify database server status.	
Perform ad hoc queries that you execute once or infrequently.	
Execute and debug SQL statements and SPL routines. Display system catalog tables and the Information Schema.	
Access, modify, and retrieve information from the database server.	
Use menus, screens, SQL statements, and SPL routines to view, access, retrieve, store, and modify data in a database.	
Work with relational (stores_demo) and object-relational (superstores_demo) demonstration databases.	
Learn how GLS affects database server migration.	<i>IBM Informix Migration Guide</i>
Load and unload data.	
Display the database schema with dbschema .	<i>IBM Informix Guide to SQL: Reference</i>
Use the system catalog tables to track objects.	
Set environment variables.	
Find a description of the tables in the stores_demo or superstores_demo database.	
Look up definitions in the glossary.	

Table 5-1. Database Tasks (continued)

If You Want To	Publication
Create databases and manage access.	<i>IBM Informix Guide to SQL: Syntax</i>
Compose correct SQL statements.	
Learn the categories of SQL statements.	
Use segments such as arguments, expressions, and identifiers.	
Write procedures with SPL and store them in a database.	
Look up reserved words.	<i>IBM Informix Guide to SQL: Tutorial</i>
Learn database concepts.	
Compose basic and advanced SELECT statements.	
Use functions and SPL routines in SQL statements.	
Modify data in a database.	
Set locks.	
Work with casts on extended data types.	
Create and use triggers.	
Use embedded SQL in programs.	
Assign data types to columns.	
	<i>IBM Informix Database Design and Implementation Guide</i> <i>IBM Informix Guide to SQL: Reference</i> <i>IBM Informix Guide to SQL: Syntax</i> <i>IBM Informix Guide to SQL: Tutorial</i>
Use the Optical Subsystem interface for an optical storage subsystem to store TEXT and BYTE data (simple large objects) on optical platters (WORM optical media). Use SQL statements to store and retrieve data to and from the Optical Subsystem.	<i>IBM Informix Optical Subsystem Guide</i>

Developing Application Programs that Access the Database

Table 5-2 lists the tasks for developing, compiling, and running client application programs and DataBlade modules that access database server data.

Table 5-2. Application Development Tasks

If You Want To	Publication
Test database applications that you intend to store for use in a production environment.	<i>IBM Informix DB–Access User's Guide</i>
Write procedures with SPL and store them in a database.	<i>IBM Informix Guide to SQL: Syntax</i>
Use a primary access method.	<i>IBM Informix Guide to SQL: Tutorial</i>
Use embedded SQL in programs.	
Program in a multiuser environment.	
Create and use routines with SPL.	
Work with user-defined and system-defined casts on extended data types.	

Table 5-2. Application Development Tasks (continued)

If You Want To	Publication
Use IBM Informix ODBC Driver to access relational databases with SQL.	<i>IBM Informix ODBC Driver Programmer's Manual</i>
Create custom applications with IBM Informix ODBC API functions.	
Embed SQL statements directly into C programs.	<i>IBM Informix ESQL/C Programmer's Manual</i>
Create new data types and user-defined routines using Java.	<i>J/Foundation Developer's Guide</i>
Use the GLS features that let Informix SQL APIs and database servers handle different languages, cultural conventions, and code sets.	<i>IBM Informix GLS User's Guide</i>
Work with the TP/XA library in an X/Open distributed transaction-processing (DTP) environment. Develop applications for a third-party transaction manager and an Informix database server.	<i>IBM Informix TP/XA Programmer's Manual</i>
Create new data types and user-defined routines using C.	<i>IBM Informix User-Defined Routines and Data Types Developer's Guide</i>
Define new data types or extend the functionality of existing data types.	
Extend operations on data types, create new casts, extend operator classes for secondary access methods, and create opaque data types for your database or DataBlade programs.	
Create application-specific SPL or external routines for application end-users.	
Create and register a user-defined routine (UDR) to invoke within an SQL statement or another routine.	
Use DataBlade API functions to develop server and client applications that access data stored in a Dynamic Server database.	<i>IBM Informix DataBlade API Programmer's Guide</i>
Write server routines and client LIBMI applications that use smart large objects and complex and extended data types.	
Use DataBlade API functions. Use Informix ESQL/C functions with the DataBlade API.	<i>IBM Informix DataBlade API Function Reference</i>
Use Java to create client applications or applets that run against Dynamic Server.	<i>IBM Informix JDBC Driver Programmer's Guide</i>
Install and load the IBM Informix JDBC Driver.	
Use standard JDBC to connect to a database or database server.	
Use standard JDBC to send queries, retrieve results, get database and column metadata, and handle errors.	
Learn how standard Java data types map to Informix data types.	
Store and retrieve XML documents.	
Use the IBM Informix HTTP proxy servlet.	
Debug JDBC API programs.	
Improve query performance in your JDBC applications.	
Use the object-oriented C++ programming language to create database client applications for Informix database servers.	<i>IBM Informix Object Interface for C++ Programmer's Guide</i>
Use Object Interface for C++ to create value objects that let C++ client applications support DataBlade module data types.	
Work with the R-tree secondary access method.	<i>IBM Informix R-Tree Index User's Guide</i>

Table 5-2. Application Development Tasks (continued)

If You Want To	Publication
Develop a secondary access method with the Virtual-Index Interface (VII) to create new types of indexes.	<i>IBM Informix Virtual-Index Interface Programmer's Guide</i>
Use functions in the VII library.	
Develop a primary access method with the Virtual-Table Interface (VTI) so that users can access external data.	<i>IBM Informix Virtual-Table Interface Programmer's Guide</i>
Use functions in the VTI library.	
Develop applications using DataBlade modules.	"DataBlade Publications" on page 6-4
Develop existing .NET applications that access Informix databases using the IBM Informix .NET Provider provided with Client SDK.	<i>IBM Informix .NET Provider Reference Guide</i>
Develop new .NET applications that access Informix or DB2 databases using the IBM Data Server Provider for .NET.	<i>IBM Data Server Provider for .NET Programmer's Guide, Informix Dynamic Server Edition</i>
Develop Java or SQLJ applications that access Informix or DB2 databases using the IBM Data Server Driver for JDBC and SQLJ.	<i>IBM Data Server Driver for JDBC and SQLJ for IDS</i>
Develop PHP applications with the PDO_INFORMIX extension to connect to Informix, or develop PHP applications with the PDO_IBM extension to connect to Informix or DB2 databases.	http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp?topic=/com.ibm.conn.doc/php.htm
Develop applications in the Ruby programming language. Two Ruby Gems are available: the Ruby Informix Gem to access Informix databases and the Ruby Gem for IBM Data Servers to access Informix or DB2 databases.	http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp?topic=/com.ibm.conn.doc/ruby.htm
Learn about and install the Call Level Interface (CLI) component of the IBM Data Server Driver for ODBC and CLI. This driver is a prerequisite for the PDO for Data Server clients for PHP connections and for the Ruby Gem for Data Server clients.	http://publib.boulder.ibm.com/infocenter/idshelp/v115/index.jsp?topic=/com.ibm.conn.doc/installcli.htm

Chapter 6. Using the Documentation

In This Chapter	6-1
The IBM Informix Documentation Set	6-1
IBM Informix Dynamic Server Publications	6-1
Client SDK and Connectivity Publications	6-3
DataBlade Publications	6-4

In This Chapter

This chapter contains an alphabetical list of the IBM Informix publications provided.

The IBM Informix Documentation Set

This set describes all the publications available with the database server, client products, tools, and DataBlade modules.

IBM Informix Dynamic Server Publications

Table 6-1 summarizes the documentation that is available with Dynamic Server.

Table 6-1. Database Server Publications

Book Title	Description
<i>IBM Informix Backup and Restore Guide</i>	This publication explains the concepts and methods required to back up and restore data with the ON-Bar and ontape utilities. It includes information on the archecker utility.
<i>IBM Informix Database Design and Implementation Guide</i>	This guide documents how to design, implement, and manage Informix databases. It includes data models that illustrate different approaches to database design and shows you how to use SQL to implement and manage databases.
<i>IBM Informix DataBlade API Function Reference</i>	This publication describes the API functions.
<i>IBM Informix DataBlade API Programmer's Guide</i>	This publication describes the API, the C-language application programming interface that is provided with Dynamic Server. Use the API to develop client and server applications that access data stored in a Dynamic Server database.
<i>IBM Informix DB–Access User's Guide</i>	This guide describes how to use the DB–Access utility to access, modify, and retrieve information from Informix database servers.
<i>IBM Informix Dynamic Server Administrator's Guide</i>	This user guide for system and database server administrators discusses the concepts and procedures for managing Dynamic Server. It is intended to help you understand, configure, and use the database server. The short publication name is <i>IBM Informix Administrator's Guide</i> .
<i>IBM Informix Administrator's Reference</i>	This reference publication provides the syntax of database server utilities such as onmode and onstat , and comprehensive descriptions of configuration parameters, SMI tables in the sysmaster database, logical-log records, disk structures, files that the database server uses, trapping errors, event alarms, and message-log messages.

Table 6-1. Database Server Publications (continued)

Book Title	Description
<i>IBM Informix Dynamic Server Getting Started Guide</i>	This guide provides an overview of IBM Informix products, summarizes the new features in this release, and provides a roadmap to user tasks in the documentation set for the database server.
<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>	This guide contains information to help you understand the concepts of data replication, design your own Enterprise Replication system, install Enterprise Replication, and administer and manage data replication throughout your enterprise.
<i>IBM Informix Dynamic Server Installation Guide for UNIX, Linux, and Mac OS X</i>	This guide contains instructions for installing Dynamic Server on UNIX, Linux, and Mac OS X. It also describes how to solve common installation problems.
<i>IBM Informix Dynamic Server Installation Guide for Windows</i>	This guide contains instructions for installing Dynamic Server on Windows.
<i>IBM Informix Dynamic Server Performance Guide</i>	This guide explains how to configure and operate Dynamic Server to achieve optimum performance and optimize SQL queries. The short publication name is <i>IBM Informix Performance Guide</i> .
<i>IBM Informix Error Messages</i>	This HTML file in the IBM Informix Online Documentation site includes causes and solutions for numbered error messages that you might receive from IBM Informix products. Use the UNIX finderr utility or the Windows Informix Error Messages utility to locate the latest information on error messages.
<i>IBM Informix GLS User's Guide</i>	This publication describes the Global Language Support (GLS), which allows IBM Informix client products and database servers to handle different languages, cultural conventions, and code sets.
<i>IBM Informix Guide to SQL: Reference</i>	This publication describes the Informix system catalog tables, data types, environment variables, the stores_demo and superstores_demo databases. It also contains a glossary.
<i>IBM Informix Guide to SQL: Syntax</i>	This publication contains the complete syntax descriptions of all Informix (SQL) and Stored Procedure Language (SPL) statements, and functions.
<i>IBM Informix Guide to SQL: Tutorial</i>	This tutorial provides instructions for using SQL to query and modify data in a relational database. It discusses how to embed SQL in programs, create and use stored-procedure language (SPL) routines, create and use triggers, and use casts for extended data types.
<i>IBM Informix High-Performance Loader User's Guide</i>	This guide describes how to use the High-Performance Loader (HPL) to efficiently load and unload large quantities of data to or from an Informix database.
<i>IBM Informix Migration Guide</i>	This publication describes the tasks that you perform when you move data from one location to another and when you migrate existing databases to various Informix database servers. It discusses such database server utilities as dbexport , dbimport , dbload , dbschema , onload , onunload , and onmode -b .
<i>IBM Informix Optical Subsystem Guide</i>	This guide describes how to use the Optical Subsystem, a utility that supports the storage of TEXT and BYTE data on optical disks.
<i>IBM Informix R-Tree Index User's Guide</i>	This guide describes the R-tree secondary access method and how to create an R-tree index on user-defined types.
<i>IBM Informix SNMP Subagent Guide</i>	This publication describes the subagent that allows a Simple Network Management Protocol (SNMP) network manager to monitor the status of Informix database servers. It includes a glossary of terms used in the guide.

Table 6-1. Database Server Publications (continued)

Book Title	Description
<i>IBM Informix Storage Manager Administrator's Guide</i>	This guide describes IBM Informix Storage Manager (ISM). ISM receives backup and restore requests from ON-Bar and directs the data to and from storage volumes that are mounted on storage devices.
<i>IBM Informix Security Guide</i>	This guide describes the security features to secure data and the secure auditing facility.
<i>IBM Informix User-Defined Routines and Data Types Developer's Guide</i>	This guide explains how to define new data types and create UDRs on Dynamic Server. It describes the tasks that you must perform to extend operations on data types, create new casts, extend operator classes for secondary access methods, write opaque data types, and create and register routines.
<i>IBM Informix Virtual-Index Interface Programmer's Guide</i>	This publication explains how to use the Virtual-Index Interface (VII), typically in a DataBlade module, to create a secondary access method. A <i>virtual</i> index accesses data from a source outside the database server or specific data inside large objects. The publication describes the syntax, API function calls, and data structures.
<i>IBM Informix Virtual-Table Interface Programmer's Guide</i>	This publication explains how to create a primary access method with the Virtual-Table Interface (VTI). A <i>virtual</i> table is dynamically created from a source outside the database server or specific data inside of large objects. The publication describes the syntax, API function calls, and data structures.
<i>J/Foundation Developer's Guide</i>	This guide explains how to use J/Foundation to write user-defined routines (UDRs) in the Java language. It describes the library of classes and interfaces that allow programmers to create and execute Java UDRs that access Dynamic Server.

Client SDK and Connectivity Publications

Table 6-2 lists the IBM Informix Client SDK and connectivity publications that you can use when you work with Dynamic Server.

Table 6-2. Client Publications for Dynamic Server

Book Title	Description
<i>IBM Informix Client Products Installation Guide</i>	This guide contains instructions for installing the IBM Informix Client Software Development Kit and IBM Informix Connect on Linux, UNIX, Mac OS X, and Windows.
<i>IBM Informix Embedded SQLJ User's Guide</i>	This publication describes how to use IBM Informix Embedded SQLJ to embed SQL statements in Java programs. When you run a SQLJ program, it uses IBM Informix JDBC Driver to connect to the database.
<i>IBM Informix ESQL/C Programmer's Manual</i>	This publication explains how to use Informix ESQL/C, the Informix implementation of embedded SQL for C, to create client applications with database-management capabilities.
<i>IBM Informix JDBC Driver Programmer's Guide</i>	This guide describes how to install, load, and use IBM Informix JDBC Driver to connect to an Informix database from a Java application or applet. You can use IBM Informix JDBC Driver to write user-defined routines.
<i>IBM Informix MaxConnect User's Guide</i>	This publication describes how to install, administer, and tune performance for Informix MaxConnect. Informix MaxConnect enables the database server to support nearly unlimited numbers of client connections and reduces response times and CPU usage.

Table 6-2. Client Publications for Dynamic Server (continued)

Book Title	Description
<i>IBM Informix Object Interface for C++ Programmer's Guide</i>	This guide describes how C++ and DataBlade developers can develop IBM Informix client applications with the C++ programming language.
<i>IBM Informix ODBC Driver Programmer's Manual</i>	This publication explains how to use the IBM Informix ODBC Driver to access Informix databases and the database server. The IBM Informix ODBC Driver is the IBM Informix implementation of the Microsoft Open Database Connectivity (ODBC) interface.
<i>IBM Informix OLE DB Provider Programmer's Guide</i>	This publication explains how to use IBM Informix OLE DB Provider to enable Active Data Objects applications and web pages, for example, to access the database server.
<i>IBM Informix TP/XA Programmer's Manual</i>	This guide describes how to use the TP/XA library, which facilitates communication between a third-party transaction manager and your database server. TP/XA is supplied with IBM Informix ESQL/C.
<i>IBM Data Server Provider for .NET Programmer's Guide, Informix Dynamic Server Edition</i>	This guide describes how to use the IBM Data Server Provider for .NET to access and manipulate data in IBM Informix databases. The provider, which is based on the Distributed Relational Database Architecture (DRDA) protocol, is also integrated with DB2. This capability means that the common features of the IBM Data Server Provider for .NET allow you to write client applications that can use both DB2 and IDS data servers.
<i>IBM Data Server Driver for JDBC and SQLJ for IDS</i>	This guide describes how to install, load, and use IBM Data Server Driver for JDBC and SQLJ to connect to an Informix database from a Java application or applet. The driver, which is based on the Distributed Relational Database Architecture (DRDA) protocol, is also integrated with DB2. This capability means that the common features of the IBM Data Server Driver for JDBC and SQLJ allow you to write client applications that can use both DB2 and IDS data servers.

DataBlade Publications

Table 6-3 lists the publications that you can use when you develop or use DataBlade modules and web-based applications with Dynamic Server.

Table 6-3. DataBlade and Tools Publications

Book Title	Description
<i>IBM Informix Database Extensions User's Guide</i>	<p>This guide explains how to use the following IBM Informix DataBlade modules that come with IBM Informix Dynamic Server:</p> <ul style="list-style-type: none"> • Large Object Locator, a foundation DataBlade module for large objects management that can be used by other modules that create or store large-object data. • MQ DataBlade module, to allow IBM Informix database applications to communicate with other MQSeries® applications with MQ messaging. • Binary DataBlade Module, which includes binary data types to store binary-encoded strings that can be indexed for quick retrieval. • Basic Text Search DataBlade module, to search words and phrases in an unstructured document repository stored in a column of a table. • Node DataBlade Module for the hierarchical data type, to represent hierarchical data within the relational database. • Web Feature Service DataBlade module, to add an Open Geospatial Consortium (OGC) web feature service as a presentation layer for the Spatial and Geodetic DataBlade modules.
<i>IBM Informix DataBlade Developers Kit User's Guide</i>	This guide describes how to develop and package DataBlade modules using BladeSmith and BladePack.
<i>DataBlade Module Development Overview</i>	This publication provides an overview of DataBlade module development.
<i>IBM Informix DataBlade Module Installation and Registration Guide</i>	This guide explains how to install DataBlade modules and use the BladeManager application to manage DataBlade modules in Informix databases. BladeManager runs on client computers.
<i>Excalibur Text Search DataBlade Module User's Guide</i>	This publication explains how to perform text searches and retrieval using the Informix Excalibur Text Search DataBlade Module.
<i>IBM Informix Data Director for Web Tutorial</i>	This tutorial teaches you how to create a small web site using Data Director for Web. You can choose the exercises that teach what you need to learn or copy the entire web site into a database and use it as an example.
<i>IBM Informix Data Director for Web User's Guide</i>	This publication describes how to use the Data Director for Web with the IBM Informix Web DataBlade Module to develop and manage web sites. Also, see the tutorial.
<i>IBM Informix Geodetic DataBlade Module User's Guide</i>	This publication explains how to use the IBM Informix Geodetic DataBlade Module to store and use spatio-temporal data such as maps.
<i>IBM Informix Image Foundation DataBlade Module User's Guide</i>	This publication explains how the Image Foundation DataBlade module provides a base on which new or specialized image types and image processing technologies can be quickly added or changed.
<i>IBM Informix Spatial DataBlade Module User's Guide</i>	This guide explains how to use the IBM Informix Spatial DataBlade Module to store, manipulate, index, and analyze multidimensional spatial data.
<i>IBM Informix TimeSeries DataBlade Module User's Guide</i>	This publication explains how to use the IBM Informix TimeSeries DataBlade Module to store and manage time-stamped data such as stock reports.

Table 6-3. DataBlade and Tools Publications (continued)

Book Title	Description
<i>IBM Informix TimeSeries Real-Time Loader DataBlade Module User's Guide</i>	This publication describes how to use the IBM Informix TimeSeries Real-Time Loader DataBlade module to load time-stamped data and make the data available to queries in real-time.
<i>IBM Informix Video Foundation DataBlade Module User's Guide</i>	This publication describes how to use the Informix Video Foundation DataBlade Module to store video technology in a media management system.
<i>IBM Informix Web DataBlade Module Administrator's Guide</i>	This publication describes how to administer web applications that use the IBM Informix Web DataBlade Module to dynamically retrieve data from Informix databases.
<i>IBM Informix Web DataBlade Module Application Developer's Guide</i>	This publication explains how to use the IBM Informix Web DataBlade Module to develop web applications that dynamically retrieve data from Informix databases.

Appendix A. Database Server Utilities

Dynamic Server includes the following utilities that let you perform administrative tasks and capture information about configuration and performance. These utilities are described in detail in the relevant publication of the Dynamic Server documentation set, as shown in the final column of the table.

Table A-1. Informix Dynamic Server Utilities

Utility	Description	Where Described
archecker	Verify backups and perform table-level restores	<i>IBM Informix Backup and Restore Guide</i>
cdr	Control Enterprise Replication operations.	<i>IBM Informix Dynamic Server Enterprise Replication Guide</i>
dbexport	Unload a database into text files for later import into another database and create a schema file.	<i>IBM Informix Migration Guide</i>
dbimport	Create and populate a database from text files. Use the schema file with dbimport to recreate the database schema.	<i>IBM Informix Migration Guide</i>
dbload	Load data into databases or tables.	<i>IBM Informix Migration Guide</i>
dbschema	Create a file that contains the SQL statements needed to replicate a specified table, view, or database, or view the information schema.	<i>IBM Informix Migration Guide</i>
imcadmin	Start or stop Informix MaxConnect, or gather statistics on Informix MaxConnect.	<i>IBM Informix MaxConnect User's Guide</i>
ism	Manage IBM Informix Storage Manager, storage devices, and media volumes.	<i>IBM Informix Storage Manager Administrator's Guide</i>
onaudit	Manage audit masks and auditing configurations.	<i>IBM Informix Security Guide</i>
onbar	Back up and restore storage spaces and logical logs.	<i>IBM Informix Backup and Restore Guide</i>
oncheck	Check specified disk structures for inconsistencies, repair inconsistent index structures, and display information about disk structures.	<i>IBM Informix Administrator's Reference</i>
oncmsm	Start the Connection Manager, which manages and redirects client connection requests based on service level agreements configured by the system administrator.	<i>IBM Informix Administrator's Reference</i>
ondblog	Change the logging mode.	<i>IBM Informix Administrator's Reference</i>
oninit	Bring the database server online.	<i>IBM Informix Administrator's Reference</i>
onload	Load data that was created with onunload into the database server.	<i>IBM Informix Migration Guide</i>
onlog	Display the contents of logical-log files.	<i>IBM Informix Administrator's Reference</i>
onmode	Change the database server operating mode and perform various other operations on shared memory, sessions, transactions, parameters, and segments.	<i>IBM Informix Administrator's Reference</i>
ON-Monitor	Perform administrative tasks using the ON-Monitor menus.	<i>IBM Informix Administrator's Reference</i>

Table A-1. Informix Dynamic Server Utilities (continued)

Utility	Description	Where Described
onparams	Modify the configuration of logical logs or physical logs.	<i>IBM Informix Administrator's Reference</i>
onpassword	Encrypts and decrypts password files for Enterprise Replication and Connection Manager.	<i>IBM Informix Administrator's Reference</i>
onperf	Monitor database server performance (create graphs, query trees, show status and metrics).	<i>IBM Informix Performance Guide</i>
onpladm	Write scripts and create files that automate data load and unload jobs.	<i>IBM Informix High-Performance Loader User's Guide</i>
onshowaudit	Extract information from an audit trail.	<i>IBM Informix Security Guide</i>
onspaces	Modify dbspaces, blobspaces, sbspaces, or extspaces.	<i>IBM Informix Administrator's Reference</i>
onstat	Monitor the operation of the database server.	<i>IBM Informix Administrator's Reference</i>
onstat -g	Monitor and debug the database server.	<i>IBM Informix Administrator's Reference</i> <i>IBM Informix Performance Guide</i>
ontape	Log, back up, and restore data.	<i>IBM Informix Backup and Restore Guide</i>
onunload	Unload data from the database server.	<i>IBM Informix Migration Guide</i>

Appendix B. 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.

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.

Notices

IBM may not offer the products, services, or features discussed in this document in all 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 give 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.

Licensees 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 information 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 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. 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 IBM's application programming interfaces.

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 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), and PostScript 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.

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.

Java and all Java-based trademarks and logos are trademarks of Sun Microsystems, Inc. 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

.NET Provider
developing applications 5-5

A

Access method
defined 3-19
primary 3-20, 5-3, 5-5
secondary 3-20, 5-5
accessibility B-1
keyboard B-1
shortcut keys B-1
Active data objects 1-4
Ad hoc query 5-2
Add-In, DBDK Visual C++ 1-5
Administrative tasks 4-4, 4-7
Administrator
database server 4-1
AFDEBUG environment variable 2-81
AFF_NPROCS configuration parameter 2-70
AFF_SPROC configuration parameter 2-70
Aggregate, user-defined 3-17
alarmprogram.sh script 2-57
Alarms
diagnosing problems 4-9
using 4-5
Allocation, dynamic log 2-78
ALTER TABLE statement
in-place 2-75
lock mode 2-77
logging mode 2-79
shadow columns 2-75
Altering tables 5-1
ANSI joins 2-65, 2-78
ANSI-compliant database 3-14
Applets, Java 5-4
Application developer 4-1
archchecker utility 4-9, 6-1, A-1
Architecture
fault tolerance and high availability 3-6
high performance 3-5
memory management 3-5
parallelization 3-6
Arguments 5-3
Attach fragments 5-1
Auditing
defined 3-10, 6-3
onaudit utility A-1
AVOID_EXECUTE directive 2-77

B

B-tree
index 3-20, 4-8, 5-1
scanner 2-59
B-trees 4-8
Backup and restore, external 3-7, 4-6
Backups
IBM Informix Storage Manager 3-7

Backups (*continued*)
improving performance 4-7
logical logs 2-79, 4-6
ON-Bar utility 3-7, A-1
ontape utility A-2
verifying 4-9
Before-image 3-4
BIGINT data type 3-12
BIGSERIAL data type 3-12
BladeManager 1-2, 6-5
BladePack 1-5
BladeSmith 1-5
BLOB data type 2-73, 3-12, 3-15
Blobpage 3-3
Blobspaces
creating 4-3
defined 3-3
onspaces utility A-2
BOOLEAN data type 3-9, 3-12
Bottleneck, performance 4-8
Bounding box 3-20
Buffer cache 4-8
Buffer manager 2-59
Buffered
disk space 3-3
transactions 3-5
Built-in data types
list 3-12
replication 3-9
Bundle Installer 2-70
BYTE data type 3-15

C

C programs 5-4
C++ Object Interface 5-4
Cache
buffer 4-8
enabling SQL statement 2-80
entry fully-inserted 2-80
SQL statements 2-80, 4-8
Callbacks 2-72
Cardinality function 2-71
Case studies 4-8
Casts
user-defined 5-2
using 5-3
cdr finderr utility 2-75
cdr utilities A-1
CDR_DBSPACE configuration parameter 2-60
CDR_ENV configuration parameter 2-60
CDR_LOGDELTA environment variable 2-61
CDR_MAX_DYNAMIC_LOGS configuration parameter 2-60
CDR_PERFLOG environment variable 2-61
CDR_QDATA_SBSpace configuration parameter 2-74
CDR_QHDR_DBSPACE configuration parameter 2-74
CDR_RMSCALEFACT environment variable 2-61
CDR_ROUTER environment variable 2-61
CDR_SERIAL configuration parameter 2-74
CHAR data type 3-12
CHARACTER VARYING data type 3-12

- Check constraints 5-2
- Checkpoint
 - improving performance 4-8
 - using 4-5
- Chinese GB18030-2000 locale 2-66
- Chunks
 - defined 3-3
 - number per database 2-56
 - offset, size limit 2-56
 - permissions and ownership 4-3
 - renaming 2-69
 - reserve pages location 2-58
 - size limit 2-56
- Ciphers 2-61
- Client application
 - defined 3-4
 - installing 4-3
 - types 3-13
- Client SDK products
 - ESQL/J 1-3
 - IBM Informix GLS 1-4
 - IBM Informix-ESQL/C 1-3
 - JDBC Driver 1-5
 - OLE DB Provider 1-4
 - TP/XA 1-5
- CLIENT_LOCALE 2-81
- Client/server
 - architecture, defined 3-4
 - setting up connectivity 4-3, 4-5
- CLOB data type 2-73, 3-12
- Cluster installation 4-3
- Collation
 - changing locale 2-63
 - Unicode algorithm 2-66
- Collection data type 2-73, 3-12, 3-16
 - cardinality 2-71
 - replicating 2-60
- Columns
 - assigning data types 5-3
 - built-in data type 3-20
 - data types, overview 3-11
 - retrieving from a database 3-10
 - storing large objects 3-15
- Commit
 - and abort callback 2-72
 - heterogeneous 3-19, 4-5
 - protocols 4-5
- Communications.
 - See* Connectivity.
- Complex data type 3-12, 3-16
- Compliance
 - ANSI 3-14
- Component, disk 3-3
- Composite R-tree index 2-76
- Configurable lock mode 2-77
- Configuration parameters
 - AFF_NPROCS 2-70
 - AFF_SPROC 2-70
 - CDR_DBSPACE 2-60
 - CDR_ENV 2-60
 - CDR_QDATA_SBSPACE 2-74
 - CDR_SERIAL 2-74
 - DBSERVERALIASES 2-57
 - DEF_TABLE_LOCKMODE 2-77
 - DYNAMIC_LOGS 2-78
 - ENCRYPT_CDR 2-60
 - ENCRYPT_CIPHERS 2-61

- Configuration parameters *(continued)*
 - ENCRYPT_MAC 2-61
 - ENCRYPT_MACFILE 2-61
 - ENCRYPT_SWITCH 2-61
 - HPL_DYNAMIC_LIB_PATH 2-68
 - HPLAPIVERSION 2-68
 - JDKVERSION 2-81
 - JVPJAVAHOME 2-81
 - JVPJAVALIB 2-81
 - JVPJAVAVM 2-81
 - LBU_PRESERVE 2-70
 - list of 2-56, 4-5
 - LOGSMAX 2-70
 - LRU_MAX_DIRTY 2-59
 - LRU_MIN_DIRTY 2-59
 - LTAPEBLK 2-58
 - LTXEHWM 2-78
 - LTXHWM 2-78
 - NOAGE 2-70
 - NUMAIOVPS 2-70
 - NUMCPUVPS 2-70
 - PLOG_OVERFLOW_PATH 2-58
 - SBSPACETEMP 2-76
 - STMT_CACHE_HITS 2-80
 - STMT_CACHE_NOLIMIT 2-80
 - STMT_CACHE_NUMPOOL 2-80
 - STMT_CACHE_SIZE 2-80
 - TAPEBLK 2-58
 - VPCLASS 2-70
- Configuring
 - backup and restore 4-3
 - client/server connections 4-5
 - database server 4-3
 - Enterprise Replication 4-3
 - HDR 4-3
 - ISM 4-3
 - locales 4-7
 - memory 4-2
 - physical and logical logs 4-3
 - SQL statement cache 2-80
 - tasks 4-1, 4-4
- Connections
 - configuring 4-3, 4-5
 - database server 4-3
 - database versus network 3-4
 - defined 3-4
 - display max number 2-70
 - mi_lo functions 2-71
 - multiplexed 3-4
 - properties 2-81
- Connectivity
 - client/server 3-4
 - ODBC standard 1-4
- Consistency, using oncheck 4-5, A-1
- Constraints 5-2
- Continuous log backup 3-7
- Cooked disk space 3-3
- Coordinating database server 3-19
- Coordinating server 3-19
- Cost-based optimizer 3-6
- CPU
 - monitoring 4-8
 - virtual processor 3-4
- CRCOLS
 - adding 2-75
 - dropping 2-75
- CREATE DISTINCT TYPE statement 3-16

- CREATE OPAQUE TYPE statement 3-13
- CREATE TABLE statement
 - lock mode 2-77
 - logging mode 2-79
- Cursor, hold with PDQ 2-59
- Custom installation 4-2, 4-3

D

- Data
 - distribution 4-8
 - I/O conversion routines 2-81
 - ls 5-2
 - storage 3-3
- Data Director For Web 6-5
- Data mart 1-1, 3-18
- Data recovery
 - data replication 3-8
 - mirroring 3-8
- Data replication
 - Enterprise Replication 2-59, 2-61, 2-72, 2-75, 3-9, 4-7
 - fixing problems 4-9
 - High-Availability Data Replication 3-8, 4-5
 - pair 3-8
- Data Server Driver for JDBC and SQLJ for IDS 5-5, 6-4
- Data Server Provider for .NET 5-5
- Data Studio 1-2, 1-3
- Data types
 - assigning to columns 5-3
 - BLOB 2-73, 3-15
 - built-in 3-12
 - BYTE 3-15
 - choosing 5-2
 - CLOB 2-73, 3-15
 - collection 2-73
 - complex 2-73
 - defining 5-4
 - distinct 3-16
 - documentation 6-2
 - geodetic 2-73
 - HTML 2-73
 - LIST 2-71, 2-73
 - multirepresentational 2-73, 2-75
 - MULTISET 2-73
 - opaque 2-73, 2-81, 5-4
 - row type 3-16
 - SET 2-71
 - TEXT 3-15
 - user defined 3-16
- Data warehouse
 - defined 1-1, 3-18
 - designing 5-2
- Database logging status 3-7, 4-5
- Database server aliases, limit 2-57
- Database servers
 - administrator 4-1
 - allocating logs dynamically 2-78
 - auditing users 4-6
 - available data types 3-11
 - client/server architecture 3-4
 - configuring 4-3
 - distributed queries 3-19
 - extending 3-15
 - fault-tolerance 3-6
 - files used 4-5
 - high performance of 3-5
 - initializing 4-3
- Database servers (*continued*)
 - installing 4-2
 - local 3-19
 - migrating 4-2, 4-4
 - monitoring 4-5
 - multiple instances 4-3
 - operating modes 4-3
 - operators 4-1
 - parallel database query 3-6
 - performance specialist 4-1
 - remote 3-19
 - security 3-9
 - system catalogs 3-11
 - transaction manager 5-4
 - unauthorized connections, preventing 4-6
 - users 4-1
 - verifying consistency 4-5
 - Windows utilities 2-70
- Databases
 - administrator 4-1
 - ANSI compliant 3-14
 - controlling access 5-2, 5-3
 - data warehousing 3-18
 - defined 3-3
 - denormalized 5-1
 - designing 5-2, 6-1
 - dimensional 3-18
 - displaying schema 5-2
 - distributed 3-19
 - external 3-19
 - external data, using 5-2
 - external remote 3-19
 - implementing 5-2
 - loading 5-2
 - local 3-19
 - management system
 - object-relational 3-14
 - relational 3-14
 - modifying data 5-3
 - normalized 5-2
 - Object Explorer 1-6
 - object-relational 3-14
 - supported types 3-14
 - tasks 5-1, 5-3
 - users 4-1
- DataBlade
 - Basic Text Search 2-38
 - data replication 3-9
 - defining data types 3-13, 5-4
 - developing applications 5-5
 - documentation 6-4
 - Excalibur Text Search 6-5
 - Geodetic 6-5
 - Image Foundation 6-5
 - installing and registering 4-3, 6-5
 - internationalized applications 1-4
 - modules, defined 3-18
 - TimeSeries 6-5
 - Video Foundation 6-6
 - Web 6-6
- DataBlade API 1-2
 - access files one buffer at a time 2-72
 - defined 1-2
 - large files, stream support for 2-68
 - memory durations 2-71
 - mi_collection_card function 2-71
 - mi_get_db_locale function 2-67

- DataBlade API (*continued*)
 - mi_get_transaction_id function 2-67
 - mi_realloc function 2-67
 - mi_stack_limit function 2-68
 - mi_system 2-68
 - transaction state 2-72
 - UDRs, functions for getting information about 2-80
 - using mi_lo without connection 2-71
 - VP environment, controlling 2-80, 2-82
- DataBlade Developers Kit 1-2, 1-5
 - User's Guide 6-5
- DataBlade modules
 - Development Overview 6-5
 - Installation and Registration Guide 6-5
- DATE data type 3-12
- DATETIME data type 3-12
- DB_LOCALE environment variable 2-81
- DB-Access
 - synonym names 2-79
 - test connection 4-3
 - using 5-2
- DBA
 - See* database administrator
- DBCENTURY environment variable 2-81
- DBDK Visual C++ Add-In 1-5
- dbexport utility 4-4, A-1
- dbimport utility 4-4, A-1
- dbload utility 4-4, A-1
- dbschema utility 4-4, 5-2, A-1
- DBSERVERALIASES configuration parameter 2-57
- dbspaces
 - creating 4-3
 - defined 3-3
 - onspaces utility A-2
- DBTIME environment variable 2-81
- DECIMAL data type 3-12
- Decision-support queries 3-13
- Decision-support system 4-8
- DeepCopy function 2-75
- DEF_TABLE_LOCKMODE configuration parameter 2-77
- Default locale x
- DELETE statements, FROM keyword 2-78
- Demonstration databases 5-2
- Denormalizing data 5-1
- Dependencies, software x
- DESCRIBE INPUT statement 2-63
- DESCRIBE OUTPUT statement 2-63
- Diagnostic information 4-9
- Dimensional database 5-2
- Director For Web, Data 6-5
- Directories, NFS 3-5
- disability B-1
- Disaster recovery 4-6
- Disk component 3-3
- Disk I/O
 - monitoring 4-8
 - reducing 3-2
 - smart large objects 3-15
- Disk space
 - optimize layout 4-8
 - placing tables 5-1
 - planning 4-2
 - raw 3-3
 - types of physical units 3-3
- Disk structure 4-5
- Displaying
 - query plan 2-77

- Displaying (*continued*)
 - SSC statistics 2-80
- Distinct data types
 - creating 3-16
 - defined 3-12
- DISTINCT TYPE statement, CREATE 3-16
- Distributed queries
 - defined 3-19
 - Microsoft Transaction Server 2-58, 2-82
- Distributions, data 4-8
- DLL 1-3
- Documentation set 6-1
- DOUBLE PRECISION data type 3-12
- Drop shadow columns 2-75
- DSA 3-2
- DSS 4-8
- DSS applications 3-13, 4-8
- DTP environment
 - See* Distributed queries.
- Dumps 4-9
- Dynamic link library 1-3
- Dynamic log
 - allocation 2-78
 - file, with Enterprise Replication 2-60
- Dynamic log allocation 2-78
- Dynamic query 2-63
- Dynamic scalable architecture
 - description of 3-2
 - virtual processor component 3-4
- Dynamic Server
 - defined 1-1
 - documentation 6-1
 - installing and migrating 1-2
- DYNAMIC_LOGS configuration parameter 2-78

E

- Embedded SQL
 - C 5-4
 - using in programs 5-3
- Embedded SQLJ
 - defined 1-3
 - JDBC Driver 2-81
- en_us.8859-1 locale x
- Enabling SQL statement cache 2-80
- ENCRYPT_CDR configuration parameter 2-60
- ENCRYPT_CIPHERS configuration parameter 2-61
- ENCRYPT_MAC configuration parameter 2-61
- ENCRYPT_MACFILE configuration parameter 2-61
- ENCRYPT_SWITCH configuration parameter 2-61
- Encryption
 - column-level data 4-6
 - communication support module 2-56
 - data transmissions 2-56
 - network data 4-6
- Enterprise Replication
 - adding shadow columns 2-75
 - cdr finderr utility 2-75
 - cdr utilities A-1
 - collection data types 2-60
 - configuring 4-3
 - defined 3-9
 - documentation 6-2
 - dropping shadow columns 2-75
 - dynamic log file 2-60
 - encryption 2-60
 - exclusive replicate sets 2-74

Enterprise Replication *(continued)*

- fixing problems 4-9
- HDR, using with 2-60
- large transaction support 2-60
- onstat options 2-75
- performance enhancements 2-73
- replicate groups 2-74
- replicate sets 2-74
- replicating
 - changed columns 2-74
 - during queue recovery 2-60
 - user-defined types 2-72
- ROW data types 2-60
- smart large objects 2-72
- spooling replicated data 2-74
- streamread 2-73
- streamwrite 2-73
- using 4-7

Entity-relationship diagram 5-2

Environment variables

- AFDEBUG 2-81
- CDR_LOGDELTA 2-61
- CDR_PERFLOG 2-61
- CDR_RMSCALEFACT 2-61
- CDR_ROUTER 2-61
- CLIENT_LOCALE 2-81
- DB_LOCALE 2-81
- DBCENTURY 2-81
- DBTIME 2-81
- documentation 6-2
- GL_DATE 2-81
- GL_DATETIME 2-81
- IFX_DEF_TABLE_LOCKMODE 2-77
- IMCADMIN 2-82
- IMCONFIG 2-82
- IMCSERVER 2-82
- INFORMIXDIR 1-4
- JAR_TEMP_PATH 2-81
- JAVA_COMPILER 2-81
- JVM_MAX_HEAP_SIZE 2-81
- list 2-56
- setting 4-3, 5-2
- USETABLENAME 2-66

Error messages

- cdr finderr utility 2-75
- corrective actions 4-9, 5-1
- documentation 4-4, 6-2

ESQL/C

- defined 1-3
- international applications 1-4

Event alarm 2-57, 4-5, 4-9

Excalibur Image DataBlade Module User Guide 6-5

Excalibur Text Search DataBlade Module User Guide 6-5

EXE.sessionid.threadid 2-71

Export function 2-81

Exportbin support function 2-81

Expression-based fragmentation 3-6, 5-1, 5-3

Extending database server 3-15

Extensible data types

- defined 5-3
- replicating 2-73

Extents

- defined 3-3
- tables 4-5, 4-8

External

- backup and restore 3-7, 4-6
- data, displaying 5-2

External *(continued)*

- database 3-19
 - remote database 3-19
 - spaces 3-3, 3-20
- ## Extspace 3-3, 3-20

F

Failover

- scripts for HDR 2-71
- scripts for High-Availability Data Replication 2-71

Failure system, and ON-Bar 3-7

Fast recovery 2-58, 3-8, 4-5

Fault tolerance

- data replication 3-8
- fast recovery 3-8
- mirroring 3-8

Field, defined 3-16

Files

- accessing one buffer at a time 2-72
- cooked 3-3
- creating with dbschema A-1
- database server 4-5
- dbexport A-1
- display contents with onlog A-1
- logical-log 3-7
- operating system 3-3
- raw 3-3
- size limit 2-57
- UNIX operating system 3-3

finderr utility 4-4

FLOAT data type 3-12

Foreign key 5-2

Fragmentation

- attaching 5-1
- defined 3-5
- expression-based 3-6, 5-1
- round robin 3-6, 5-1
- tables 4-8
- using 5-2

FROM keyword 2-78

Fully-inserted cache entry 2-80

Functional index

- B-tree 3-20
- column limit 2-63

Functions

- mi_collection_card 2-71
- mi_dalloc 2-71
- mi_file 2-72
- mi_get_db_locale 2-67
- mi_get_transaction_id 2-67
- mi_realloc 2-67
- mi_stack_limit 2-68
- mi_system 2-68
- mi_transaction_state 2-72
- streamread 2-73
- streamwrite 2-73

G

Generic B-tree 3-20

Geodetic

- data type 2-73
- DataBlade 6-5

GL_DATE environment variable 2-81

GL_DATETIME environment variable 2-81

- Global Language Support (GLS)
 - Chinese GB18030-2000 locale 2-66
 - defined x
 - defining locales 4-7
 - GLS library 1-4
 - Java support 2-81
 - migrating 5-2
 - programming 5-4
 - Unicode support 2-66
 - Unicode, collation 2-66
- Global Language Support (GLS)
 - documentation 6-2
- Glossary 4-2, 5-2
- GLS.

See Global Language Support.

- Granting
 - database access 5-2
 - privileges 2-78

H

- Hardware upgrades 4-2
- Hash join 3-6
- HDR.
 - See* High-Availability Data Replication.
- hdrmkpri.sh script 2-71
- hdrmksec.sh script 2-71
- Heterogeneous commit 3-19, 4-5
- High-Availability Data Replication
 - configuring 4-3
 - creating external backups 4-6
 - defined 3-8, 4-5
 - Enterprise Replication, using with 2-60
 - failover scripts 2-71
 - type of data replicated 2-61
- High-Performance Loader
 - custom-code shared library file 2-68
 - documentation 6-2
 - HPL_DYNAMIC_LIB_PATH configuration parameter 2-68
 - HPLAPIVERSION configuration parameter 2-68
 - using 4-6
 - using full capacity of storage media 2-68
- HPL_DYNAMIC_LIB_PATH configuration parameter 2-68
- HPL.
 - See* High-Performance Loader.
- HPLAPIVERSION configuration parameter 2-68
- HTML data type 2-73
- HTTP proxy server 5-4

I

- IBM Data Server Driver for JDBC and SQLJ for IDS 5-5, 6-4
- IBM Data Server Provider for .NET 5-5
- IBM Informix
 - Client Software Development Kit
 - defined 1-3
 - publications 6-3
 - Data Director For Web User's Guide 6-5
 - DB-Access User's Guide 6-1
 - GLS User's Guide 6-2
 - Migration Guide 6-2
 - SNMP Subagent Guide 6-2
 - Web DataBlade Module Administrator's Guide 6-6
- IBM Informix Dynamic Server.
 - See* Database server.
- IBM Informix ESQL/C 1-3

- IBM Informix ESQL/J pre-processor 1-3
- IBM Informix GLS 1-4
- IBM Informix JDBC Driver
 - defined 1-5
 - using 5-4
- IBM Informix MaxConnect 1-7, 6-3
- IBM Informix Object Interface for C++ 1-4
- IBM Informix ODBC Driver 1-4, 2-58, 5-4
- IBM Informix OLE DB provider 1-4
- IBM Informix Server Administrator
 - defined 1-6
 - monitoring MaxConnect 2-82
 - Server Setup 4-3
- IBM Informix SNMP subagent 4-7
- IBM Informix Spatial DataBlade module 2-73, 6-5
- IBM Informix Spatial DataBlade Module 1-5
- IBM Informix Storage Manager
 - defined 3-7
 - documentation 6-3
 - ism utility A-1
 - setting up 4-3
 - using 4-6
- IBM Office Connect 1-7
- Identical statements 2-80
- Identifiers 5-3
- IDSSECURITYLABEL data type 3-12
- IFX_DEF_TABLE_LOCKMODE environment variable 2-77
- Image Foundation DataBlade module 6-5
- IMCADMIN environment variable 2-82
- imcadmin utility 2-82, A-1
- IMCONFIG environment variable 2-82
- IMCSERVER environment variable 2-82
- import function 2-81
- importbin function 2-81
- Imported restore 4-6
- Improving performance 5-1
- In-place alters 2-75
- Index
 - access methods 3-19
 - B-tree 3-20
 - column limit 2-63
 - fragmenting 3-5, 5-1
 - functional 3-20
 - OLTP applications 3-13
 - optimizer-determined 3-6
 - R-Tree 3-20
 - repair A-1
 - using access methods 4-8
- index.htm 1-4
- Information Schema A-1
- Informix Client SDK 1-2
- Informix Connect 1-2
- Informix Spatial DataBlade module 1-2
- Informix Web DataBlade Module 1-2, 1-5
- INFORMIXDIR environment variable 1-4
- INFORMIXDIR/bin directory x
- Inheritance 3-17, 5-2
- Initializing database server 4-3
- Input function 2-81
- Installing
 - client applications 4-3
 - client files 2-69
 - database server 4-2
 - DataBlades 4-3, 6-5
 - GLS files 2-69
 - Informix products 1-2, 2-70
 - MaxConnect 4-3

Installing (*continued*)
no files in /usr/lib directory 2-69
planning for 4-2
serial and key not needed 2-69
tasks 4-1, 4-4

INTEGER data type 3-12
Integration, vendor products 4-2
International Components for Unicode 2-66
International Language Supplement 1-2
Interprocess communications 3-3
INTERVAL data type 3-12
IPC.

See Interprocess communications.

ipload utility 4-6

IPX/SPX

See Network.

ism utility A-1

ISO 8859-1 code set x

Isolation level 4-8

Iterator, in a FROM clause 2-62

ixpasswd.exe utility 2-70

ixsu.exe utility 2-70

J

J/Foundation

9.21 features 2-81

accessing opaque types 2-81

configuring 4-3

connection properties 2-81

documentation 6-3

embedding SQL statements 2-81

runtime environment variables 2-81

send and receive functions 2-81

updating names of jar files 2-81

user-defined routines 5-4

using applications 1-5, 5-4

J/Foundation Developer Guide 6-3

JAR files, renaming 2-81

JAR_TEMP_PATH environment variable 2-81

Java Applets 5-4

Java Runtime Environment 2-77, 2-81

Java Virtual Machine 2-77, 2-81

JAVA_COMPILER environment variable 2-81

Java.

See J/Foundation.

JavaSoft specifications 1-5

JDBC

defined 1-5

Version 2.0 support 2-81

JDKVERSION parameter 2-81

Join

ANSI 2-65, 2-78

methods 3-6

JRE.

See Java Runtime Environment.

JVM_MAX_HEAP_SIZE environment variable 2-81

JVM.

See Java Virtual Machine.

JVPJAVAHOME configuration parameter 2-81

JVPJAVALIB configuration parameter 2-81

JVPJAVAVM configuration parameter 2-81

K

Key-only cache entry 2-80

Keys

non-leading index 2-36

primary and foreign 5-2

secret 2-61

L

Language, types of 3-17

LBU_PRESERVE configuration parameter 2-70

LIBMI applications 5-4

Library

ESQL/C 1-3

GLS feature 1-4

ODBC 1-4

Limiting database access 5-2

Linux

installation applications 2-55

installing database server 1-2, 6-2

LIST data type

defined 3-12

obtaining cardinality 2-71

replication not supported 2-73

Load balancing 4-8

LOAD TO statement 2-64

Loading tables

migration 5-2

modifying 5-1

Local database server 3-19

Locales

Chinese GB18030-2000 2-66

collation, changing 2-63

data formats 1-4

GLS feature x, 1-4

setting up 4-7

Lock mode, configurable 2-77

Locking

new tables 2-77

setting lock mode 5-1, 5-3

UDRs 2-81

Log files

list 2-78

salvaging, logical 3-7

Logging mode A-1

Logging.

See Logical log.

Logical log

backup 2-79, 3-7, 4-6

configuring 4-3

defined 3-4

dynamic allocation 2-78

managing 4-5

onlog utility A-1

onparams utility A-2

records 4-5

Logical partition 3-3

Logical units of storage, list of 3-3

LOGSMAX configuration parameter 2-70

Long transaction 2-78, 4-5

LRU queues 4-8

LRU_MAX_DIRTY configuration parameter 2-59

LRU_MIN_DIRTY configuration parameter 2-59

LTAPBLK configuration parameter 2-58

LTXEHWM configuration parameter 2-78

LTXHWM configuration parameter 2-78

LVARCHAR data type 2-65, 3-9, 3-12

M

- MaxConnect
 - defined 1-7, 2-82
 - documentation 6-3
 - imcadmin utility 2-82, A-1
 - installing 4-3
- Maximum number of connections 2-70
- Memory
 - dynamically sharing memory 3-5
 - managing
 - buffered transactions 3-5
 - shared memory 4-5
 - monitoring 4-8
 - optimal configuration 4-2
- Memory duration 2-71, 2-75
- Messages.
 - See* Error messages.
- Metadata
 - data mart 3-18
 - new chunk 2-58
 - partitioning 2-76
- Methods
 - access 3-19
 - join 3-6
 - primary access 5-5
 - R-tree 5-4
 - secondary access 5-5
- mi_collection_card() function 2-71
- mi_dalloc() function 2-71
- MI_EVENT_COMMIT_ABORT callback 2-72
- MI_EVENT_POST_XACT callback 2-72
- MI_EVENT_SAVEPOINT callback 2-72
- mi_fparam structure 2-81
- mi_get_db_locale() function 2-67
- mi_get_transaction_id() function 2-67
- mi_lo functions 2-71
- mi_realloc() function 2-67
- mi_stack_limit() function 2-68
- mi_system() function 2-68
- mi_transaction_state() function 2-72
- Microsoft Open Database Connectivity 1-4
- Migration
 - database server 1-2, 4-2, 4-4
 - documentation 6-2
 - Enterprise Replication 2-72, 2-74
 - GLS feature 5-2
 - utilities A-1
- Mirroring
 - defined 3-8
 - implementing 4-3
 - performing 4-5
- Mode
 - database server 4-3
 - lock, setting 5-3
- MONEY data type 3-12
- Monitoring
 - database server 4-5
 - locks 5-1
 - MaxConnect 2-82
 - SQL statement cache 2-80
 - system and queries 4-8
 - transactions 2-58
- Moving data 4-4, 4-6
- MTS/XA 2-58, 2-82
- Multibyte character string 1-4
- Multiple
 - OUT parameter 2-64

- Multiple (*continued*)
 - residency 4-2, 4-3
- Multiplexing connections 1-7, 3-4
- Multirepresentational data 2-73, 2-75
- MULTISET data type
 - defined 3-12
 - obtaining cardinality 2-71
 - replication not supported 2-73
- Multiuser environment 5-3

N

- Named
 - pipe connection 3-5
 - return values 2-62
 - row data type 3-12, 3-16
- NCHAR data type 3-12
- Nearest-neighbor, query 2-76
- Network
 - capacity, planning 4-2
 - protocols 2-82
 - SNMP 4-7
- New features 4-2
- New Technology File System 3-3
- NEWCODESET connection property 2-81
- NEWLOCALE connection property 2-81
- NFS directory 3-5
- NOAGE configuration parameter 2-70
- Nonlogging tables 2-79
- Normalized database 5-2
- ntchname utility 2-70
- NTFS 3-3
- Null values 2-71
- NUMAIOVPS configuration parameter 2-70
- NUMCPUVPS configuration parameter 2-70
- NUMERIC data type 3-12
- NVARCHAR data type 3-12

O

- Object Explorer 1-6
- Object Interface for C++ 1-4, 5-4
- Object-relational database 3-14, 5-2
- Objects, data 5-2
- ODBC driver 1-4, 2-58, 5-4
- OLE DB provider 1-4
- OLTP applications 3-13, 4-8
- ON-Bar utility A-1
 - b -l command 2-79
 - configuring 4-3
 - defined 3-7
 - documentation 6-1
 - renaming chunks during restore 2-69
 - return codes 4-9
 - using full capacity of storage media 2-69
- ON-Monitor utility 4-5
- onaudit utility 4-6, A-1
- oncheck utility
 - defined A-1
 - printing chunk pages 2-67
 - verifying consistency 4-5
- oncmsm utility
 - defined A-1
- ONCONFIG file, setting parameters 4-3
- onconfig.std file 2-70
- ondblog utility 4-5, A-1

- oninit utility 4-5, A-1
- Online transaction processing 1-1, 4-8
- onload utility 4-4, A-1
- onlog utility 4-5, A-1
- onmode utility
 - b option 4-4
 - I option 4-9
 - W options 2-80
 - Y option 2-67
 - described A-1
- onparams utility 2-78, 4-5, A-2
- onpassword utility
 - defined A-2
- onperf utility 4-8, A-2
- onpladm utility 2-79, 4-6, A-2
- onpload utility 4-6
- onshowaudit utility 4-6, A-2
- onsocimc protocol 1-7
- onspaces utility 4-5, A-2
- onstat utility
 - g dss UDR option 2-75
 - g dss UDRx option 2-75
 - g env option 2-67
 - g grp UDRx option 2-75
 - g imc option 2-82
 - g mem option 2-71
 - g ses option 2-67
 - g sql option 2-67
 - g ssc option 2-80
 - g stm option 2-77
 - x option 2-58
 - defined A-2
 - diagnosing problems 4-7
 - usage 4-5
- ontape utility
 - configuring 4-3
 - documentation 6-1
 - listed A-2
 - renaming chunks during restore 2-69
 - using full capacity of storage media 2-69
- ontliimc protocol 1-7
- onunload utility 4-4, A-2
- Opaque data types
 - creating 5-4
 - defined 3-13
 - support for replicating 2-73
- OpenAdmin Tool for IDS 1-6, 4-5, 4-7
 - Server Setup 4-3
- Operating modes 4-3
- Operating system
 - configure 4-2
 - raw and cooked disk space 3-3
 - UNIX files 3-3
- Operational data store 3-18
- Operator class, extending 5-4
- Operator, backup 4-1
- Optical Subsystem
 - documentation 6-2
 - using 4-6, 5-3
- Optimizer
 - cost-based 3-6
 - external directives 2-51
 - spacial query costing 2-59
- Optimizer directives
 - AVOID_EXECUTE 2-77
 - using 4-8
- Optimizing performance 6-2

- Outer join 4-8
- Output function 2-81
- Ownership, resolving 4-3

P

- Page 3-3
- Parallel database queries 2-59, 3-6, 4-8
- Parallel processing
 - defined 3-6
 - Enterprise Replication 2-73
- PDQ
 - See Parallel database query.
- PER_STATEMENT memory duration 2-71
- PER_STMT_EXEC memory duration 2-71
- PER_STMT_PREP memory duration 2-71
- Performance
 - B-tree scanner 2-59
 - backup and restore 4-7
 - buffer manager 2-59
 - denormalizing data 5-1
 - fragmentation 3-5
 - LRU settings 2-59
 - memory management 3-2, 3-5
 - monitoring 4-7
 - new improvements 2-73
 - onperf utility A-2
 - optimizing 6-2
 - parallelization 3-6
 - PDQ with hold cursors 2-59
 - queries 4-8, 5-4
 - spatial query costing 2-59
 - specialist 4-1
 - SPL routines 3-10
 - statistics 4-8
 - tuning mechanisms 3-5
- Permissions
 - chunks 4-3
 - resolving 4-3
- PHP
 - developing applications 5-5
- Physical drive 3-3
- Physical log
 - configuring 4-3
 - defined 3-4
 - fast recovery 3-8
 - managing 4-5
 - onparams utility A-2
 - overflow 2-58
- Physical units of storage, list of 3-3
- Planning
 - database design 5-2
 - database server installation 4-2
 - tasks 4-1, 4-4
- PLOG_OVERFLOW_PATH configuration parameter 2-58
- Point-in-time restore 3-7
- Populating databases 5-2
- Post-transaction callback 2-72
- Primary access method 3-20, 5-3, 5-5
- Primary key
 - UDT columns 2-73
 - using in tables 5-2
- Priority management for buffers 2-59
- Private installation 4-2, 4-3
- Privileges 2-78, 5-2
- Processes
 - compared to threads 3-4

- Processes (*continued*)
 - repair 2-52
- Processing ALTER statements 2-75
- Programmers 4-1
- Programming tasks 5-3, 5-5
- Protocol
 - See* Network.
- Protocol for multiplexing connections 1-7
- Proxy server 5-4
- PRP.sessionid.threadid pool 2-71
- Publication
 - listed 6-1

Q

- Qualifying statements 2-80
- Query
 - ad hoc 5-2
 - defined 3-10
 - filtering 5-1
 - improving performance 4-8, 5-4
 - language, structured 3-10
 - monitoring 4-8
 - nearest neighbor 2-76
 - optimizer 3-6
 - parallel database 3-6
 - system catalog 3-11
- query drill down 4-8
- Query plans
 - displaying without executing the query 2-77
 - optimizing 4-8

R

- R-tree index
 - access method 3-20
 - documentation 6-2
 - nearest-neighbor query 2-76
 - query costing 2-59
 - using 5-1, 5-4
- Rational Application Developer for WebSphere Software 1-2, 1-6
- Raw disk space 3-3
- Raw table 2-79, 4-7
- RDBMS.
 - See* Relational Database Management System.
- Read-only scroll cursors 2-81
- REAL data type 3-12
- Records, logical log 4-5
- Recovery.
 - See* Restore.
- Referential constraint 5-2
- Registering DataBlades 6-5
- Relational database
 - application types 3-13
 - components of 3-10
 - data types 3-11
 - designing 5-2
 - system catalog 3-11
- Relational Database Management System 3-14
- Remote database
 - access 2-75
 - server 3-19
- RENAME INDEX statement 2-79
- Renaming
 - chunks 2-69

- Renaming (*continued*)
 - jar files 2-81
- Replicate
 - groups 2-74
 - sets 2-74
- Replicating
 - changed columns 2-74
 - smart large objects 2-73
 - user-defined types 2-73
- Repository, defined 3-18
- Reserved area, sbspaces 2-76
- Reserved SQL keywords 2-56, 5-3
- Residency, multiple 4-2, 4-3
- Restore
 - archecker utility A-1
 - defined 3-6, 3-7
 - external 4-6
 - imported 4-6
 - improving performance 4-7
 - ON-Bar utility 3-7, A-1
 - ontape utility A-2
 - point-in-time 3-7
- Return codes, ON-Bar 4-9
- Reverting database server 4-4
- REVOKE statement 2-78
- Revoking privileges 2-78
- Roll back 2-78
- Root dbspace 3-8
- Round-robin fragmentation 3-6, 5-1
- Routine identifier 2-81
- ROW data types
 - defined 3-12
 - named 3-16
 - replicating 2-60
 - unnamed 3-16
- Ruby
 - developing applications 5-5

S

- Salvaging logical-log files 3-7
- Save point callback 2-72
- Save points 2-72
- Sbpage 3-3
- sbspaces
 - creating 4-3
 - defined 3-3, 3-15
 - managing 4-5
 - onspaces utility A-2
 - reserved space 2-76
 - temporary 2-76, 3-15
- SBSPACETEMP configuration parameter 2-76
- Scaleup 3-2
- Schema 4-6, 5-2
- Schema Editor 1-6
- Scroll cursors 2-81
- Searching text
 - Basic Text Search DataBlade 2-38
 - Excalibur Text Search 6-5
- Secondary-access methods 3-20, 4-8, 5-5
- Secret keys 2-61
- Security
 - access control 4-6
 - authorized users 4-6
 - column-level data encryption 4-6
 - database server 3-9
 - encrypting transmissions 2-56

- Security (*continued*)
 - network data encryption 4-6
 - unauthorized connections, preventing 4-6
- Segment 5-3
- SELECT statements 2-62, 2-65, 5-3
- send function 2-81
- Sequence objects 2-65
- SERIAL or SERIAL8 data type 3-12
- Server Setup 4-3
- Server Studio 1-2, 1-6
- Server.
 - See* Database server.
- SET data type 2-71, 3-12
- SET EXPLAIN statement
 - ANSI joins 2-79
 - AVOID_EXECUTE keyword 2-77
- SET residency statement not needed 2-64
- Shadow columns 2-75
- Shared memory
 - connection 3-5
 - interprocess communication 3-3
 - managing 3-5, 4-5
 - performance 3-2
- shortcut keys
 - keyboard B-1
- Silent installation 4-2, 4-3
- Simple large objects
 - defined 3-15
 - replication 3-9
 - using Optical Subsystem 4-6, 5-3
- Simple Network Management Protocol 6-2
- Single-byte character string 1-4
- SMALLFLOAT data type 3-12
- SMALLINT data type 3-12
- Smart large objects
 - APIs 3-15
 - copying data 2-75
 - defined 3-15
 - I/O properties 3-15
 - replicating 2-72
 - sbspaces 3-15
 - spooling replicated data 2-74
 - temporary 2-76, 3-15
- SmartDisk, not supported 2-69
- SMI tables 4-5
- SMI.
 - See* System-monitoring interface.
- SMP.
 - See* Symmetric multiprocessing.
- SNMP subagent 4-7, 6-2
- Software dependencies x
- Software upgrades 4-2
- Spatial database 2-76
- Spatial DataBlade 2-73
- Spatial DataBlade module 6-5
- Spatial query costing 2-59
- Speedup 3-2
- SPL routines
 - creating UDRs 5-3
 - using 5-2, 5-3
- Spooling replicated data 2-74
- sqexplain.out file 2-77
- SQL Editor 1-6
- SQL reserved words 2-65
- SQL statements
 - ALTER TABLE 2-77, 2-79
 - cache 2-79, 4-8

- SQL statements (*continued*)
 - client applications 3-4
 - composing 5-3
 - CREATE OPAQUE TYPE 3-13
 - CREATE SEQUENCE 2-65
 - CREATE TABLE 2-77, 2-79
 - DELETE 2-78
 - DESCRIBE INPUT 2-63
 - DESCRIBE OUTPUT 2-63
 - displaying memory used 2-77
 - documentation 6-2
 - embedded SQL 5-3
 - INSTEAD OF triggers on views 2-62
 - invoking UDRs 5-4
 - LOAD TO 2-64
 - memory durations 2-71
 - ORDER BY 2-63
 - RENAME INDEX 2-79
 - reserved keywords 2-56
 - REVOKE 2-78
 - SELECT 5-3
 - SET COLLATION 2-63
 - SET EXPLAIN ON AVOID_EXECUTE 2-77
 - SET residency 2-64
 - UNLOAD TO 2-64
 - UPDATE STATISTICS 4-8
- sqlhosts file or registry 3-4
- SQLJ, defined 1-3
- Standard table 4-7
- Statement cache, SQL 2-79
- Statement Local Variables, multiple 2-62
- Statistics 4-8
- STMT_CACHE_HITS configuration parameter 2-80
- STMT_CACHE_NOLIMIT configuration parameter 2-80
- STMT_CACHE_NUMPOOL configuration parameter 2-80
- STMT_CACHE_SIZE configuration parameter 2-80
- Storage manager, third-party 4-3
- Storage media
 - configure 4-3
 - planning 4-2
 - using full capacity of 2-57
- Storage spaces
 - backing up 3-7
 - managing 3-3, 4-5
- Storage volumes 4-3
- stores_demo database x, 5-2
- Storing on optical disk 4-6
- Stream-pipe connection 3-5
- streamread() support function 2-73
- streamwrite() support function 2-73
- Striping, disks 4-2
- Subquery, writing 4-8
- superstores_demo database x, 5-2
- Symmetric multiprocessing 3-2
- Synonym names in DB-Access 2-79
- sysmaster database 4-5, 5-1
- sysprocedures table 2-80
- syssscstat table 2-80
- sysstmtcache table 2-80
- System catalog tables
 - defined 3-11
 - documentation 6-2
 - querying 4-7
 - sysprocedures 2-80
 - tables, list of 2-56
 - using 5-2
- System failure, and ON-Bar 3-7

- System requirements
 - database x
- System-monitoring interface tables
 - list of 2-56
 - syssscstat 2-80
 - sysstmtcache 2-80
 - using 4-5

T

- Table
 - access methods 3-19
 - altering schema 4-6
 - defined 3-3
 - demonstration databases 5-2
 - extents 4-5, 4-8
 - fragmenting 3-5, 5-1
 - inheritance 5-2
 - locking 2-77
 - modifying 5-1
 - nonlogging 2-79
 - placing on disk 5-1
 - RAW 4-7
 - STANDARD 4-7
 - system catalog 3-11
 - TEMP 4-7
- Table fragmentation.
 - See* Fragmentation.
- Tape
 - block size 2-58
 - using the full capacity 2-57
- TAPEBLK configuration parameter 2-58
- Task-documentation matrix 4-1
- Tasks
 - administrative 4-4, 4-7
 - application programming 5-3, 5-5
 - database-related 5-1, 5-3
 - performance 4-7
 - plan, install, and configure 4-1, 4-4
 - troubleshooting 4-8, 4-9
- Tblspace 3-4
- TCP/IP 3-5
- Temp table 4-7
- Temporary
 - dbspaces 4-5
 - sbspaces 2-76, 3-15
 - smart large object 2-76, 3-15
 - table, fragmenting 5-1
- Terms, defined 4-2, 5-2
- TEXT data type 3-15
- Text search
 - Basic, DataBlade 2-38
 - Excalibur Text Search 6-5
- Third-party storage managers 4-3
- Threads 3-4
- TIME data type 3-9
- Time-stamped data 6-5
- Timeseries DataBlade 6-5
- TLI connection 3-5
- TP/XA library 1-5, 5-4
- Transaction state, DataBlades 2-72
- Transactions
 - callbacks 2-72
 - committing and rolling back 2-72
 - defined 3-5
 - distributed 3-19
 - long 2-78, 4-5

- Transactions (*continued*)
 - manager 2-58, 5-4
- Triggers 2-62, 5-3
- Troubleshooting 4-8, 4-9
- Truncating tables 5-1
- Two-phase commit 3-19, 4-5
- Type of tables 4-7
- Typical installation 4-2, 4-3

U

- UDT.
 - See* User-defined type.
- Unbuffered disk space 3-3
- Unicode code points 2-66
- Uninstall 4-2, 4-3
- Union, in subqueries 2-65
- UNIX
 - Bundle Installer 2-70
 - files 3-3
 - installing database server 1-2, 6-2
- UNLOAD TO statement 2-64
- Unnamed row type 3-12, 3-16
- UPDATE STATISTICS statement 4-8
- update_jars.sql script 2-81
- User-defined
 - access methods 3-17
 - aggregate 3-17
 - cast, using 5-2, 5-3
 - functions 3-16
 - procedure 3-17
- User-defined routines
 - aggregates 3-17
 - defined 3-16
 - documentation 6-3
 - Java routines 1-5, 5-4
 - memory durations 2-71
 - multiple OUT parameters 2-64
 - naming returned values 2-62
 - obtaining information about 2-80
 - registering 5-4
 - SPL 5-3
- User-defined type
 - defined 3-16
 - primary key column 2-73
 - remote database access 2-75
 - replication 2-72, 3-9
 - WHERE clause, column reference 2-73
- Users, types of ix, 4-1
- USETABLENAME environment variable 2-66
- Utilities
 - archecker A-1
 - auditing 6-3
 - cdr A-1
 - DB-Access 5-2, 6-1
 - dbexport 4-4, A-1
 - dbimport 4-4, A-1
 - dbload 4-4, A-1
 - dbschema 4-4, 5-2, A-1
 - ipload 4-6
 - ISA 1-6
 - ism A-1
 - ixpasswd 2-70
 - ixsu 2-70
 - migration 6-2
 - ntchname 2-70
 - ON-Monitor utility 4-5

Utilities (*continued*)

- onaudit 4-6, A-1
- onbar A-1
- oncheck 4-5, A-1
- oncmsm A-1
- ondblog utility 4-5, A-1
- oninit 4-5, A-1
- onload 4-4, A-1
- onlog 4-5, A-1
- onmode
 - b option 4-4
 - I option 4-9
 - defined A-1
- onparams 2-78, 4-5, A-2
- onpassword A-2
- onperf 4-8, A-2
- onpladm 4-6, A-2
- onpload 4-6
- onshowaudit 4-6, A-2
- onspaces 4-5, A-2
- onstat -g 4-7
- onstat utility 4-5, A-2
- ontape utility A-2
- onunload 4-4, A-2

V

- VARCHAR data type 3-12
- Variable-length opaque type 2-81
- Varying character data type 3-9
- Verifying backups 4-9
- Video Foundation DataBlade 6-6
- Views 2-62, 5-2
- VII 5-5
- Virtual processors
 - controlling 2-80
 - defined 3-4
 - dynamic scalable architecture 3-2
 - managing 4-5
 - monitoring 4-8
 - shared-memory component 3-2
 - user-defined 3-17
- Virtual-index interface 5-5
- Virtual-table interface 5-5
- Volume, storage 4-3
- VPCLASS configuration parameter 2-70
- VTI 5-5

W

- Web DataBlade module 6-6
- Web pages, accessing with
 - OLE DB 1-4
 - Web DataBlade 6-6
- WHERE clause, UDT column reference 2-73
- Windows
 - installing database server 1-2, 6-2
 - utilities 2-70
- Words, reserved 5-3
- WORM optical media 4-6, 5-3

X

- X/Open environment 5-4
- XML documents 5-4



Printed in USA

GI11-8342-02

