IBM Informix Dynamic Server Getting Started Guide

Version 9.4
March 2003
Part No. CT1T1NA

Table of Contents

Introduction

In This Introduction

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

About This Manual

Use this manual to get started with IBM Informix Dynamic Server or IBM Informix Dynamic Server with J/Foundation, Version 9.4. This manual describes the products bundled with Dynamic Server, the new features in Version 9.4, 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 manual and the intended audience.

Types of Users

This manual 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 manual 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," and to "Related Reading" on page 13 for a list of supplementary titles.

Software Dependencies

This manual is written with the assumption that you are using Dynamic Server, Version 9.4, 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 manual 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 manuals 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.

New Features in Version 9.4

For a comprehensive list of new features in IBM Informix Dynamic Server, Version 9.4, see Chapter 2, "Using New Features in Dynamic Server."

Organizational Changes to This Manual Since 9.2

Getting Started Guide has been reorganized as follows:

- Chapter 1 describes the products bundled with Dynamic Server and other related products
- Chapter 2 describes the new features in Version 9.4, 9.3, and 9.21.
- Chapter 3 describes the major features in Dynamic Server:
  - Dynamic scalable architecture, data storage, and shared memory
  - High performance
  - Backup, recovery, and high availability
  - Relational and object-relational database features
  - Data types supported
  - Access methods
- Chapter 4 describes the tasks for installing, administering, tuning, and troubleshooting Dynamic Server.
- Chapter 5 describes the tasks for designing, maintaining, and extending databases, and developing application programs.
- Chapter 6 describes the books in the IBM Informix documentation set.
- Appendix A is a quick reference to command-line utilities.
- A Notices appendix describes IBM products, features, and services.
- An index directs you to areas of particular interest.

Documentation Conventions

This section describes the conventions that this manual uses. These conventions make it easier to gather information from this and other volumes in the documentation set.

Typographical Conventions

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

Convention	Meaning
KEYWORD	All primary elements in a programming language statement (keywords) appear in uppercase letters in a serif font.
italics ***italics*** *`italics`*	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 ***boldface***	Names of program entities (such as classes, events, and tables), environment variables, file and pathnames, and interface elements (such as icons, menu items, and buttons) appear in boldface.
`monospace` *`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 the end of product- or platform-specific information.
→	This symbol indicates a menu item. For example, "Choose **Tools→Options**" means choose the **Options** item from the **Tools** menu.

Tip: *When you are instructed to "enter" characters or to "execute" a command, immediately press* RETURN *after the entry. When you are instructed to "type" the text or to "press" other keys, no* RETURN *is required.*

Icon Conventions

Throughout the documentation, you will find text that is identified by several different types of icons. This section describes these icons.

Comment Icons

Comment icons identify three types of information, as the following table describes. This information always appears in italics.

Icon	Label	Description
⚠	***Warning:***	Identifies paragraphs that contain vital instructions, cautions, or critical information
⇨	***Important:***	Identifies paragraphs that contain significant information about the feature or operation that is being described
💡	***Tip:***	Identifies paragraphs that offer additional details or shortcuts for the functionality that is being described

Cross-Reference Icons

Cross-reference icons indicate paragraphs that show where you can find more information about a topic.

Icon	Description
📚	Identifies paragraphs that contain cross-references to other IBM Informix manuals that provide additional information on a topic

Feature, Product, and Platform Icons

Feature, product, and platform icons identify paragraphs that contain feature-specific, product-specific, or platform-specific information.

Icon	Description
E/C	Identifies information that is specific to IBM Informix ESQL/C
GLS	Identifies information that relates to the IBM Informix Global Language Support (GLS) feature
Java	Identifies information that is specific to UDRs written in Java
UNIX	Identifies information that is specific to the UNIX operating system
UNIX/Linux	Identifies information that is specific to the UNIX and Linux operating systems
Windows	Identifies information that applies to all Windows environments
WIN 2000	Identifies information that is specific to the Windows 2000 environment
Win32	Identifies an API that provides 32-bit support for Windows 95 and NT

These icons can apply to a row in a table, one or more paragraphs, or an entire section. A ♦ symbol indicates the end of the feature-specific, product-specific, or platform-specific information.

Additional Documentation

IBM Informix Dynamic Server documentation is provided in a variety of formats:

- **Online manuals.** You can obtain online manuals at the IBM Informix Online Documentation site at http://www.ibm.com/software/data/informix/pubs/library/.
- **Online help.** This facility provides context-sensitive help, an error message reference, language syntax, and more.
- **Documentation notes and release notes.** Documentation notes, which contain additions and corrections to the manuals, and release notes are located in the directory where the product is installed.

  Please examine these files because they contain vital information about application and performance issues. The following table describes these files.

UNIX

On UNIX platforms, the following online files appear in the **$INFORMIXDIR/release/en_us/0333** directory.

Online File	Purpose
ids_start_docnotes_9.40.html	The documentation notes file for your version of this manual describes topics that are not covered in the manual or that were modified since publication.
ids_unix_release_notes_9.40.html **ids_unix_release_notes_9.40.txt**	The release notes file describes feature differences from earlier versions of IBM Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.
ids_machine_notes_9.40.txt	The machine notes file describes any special actions that you must take to configure and use IBM Informix products on your computer. Machine notes are named for the product described.

♦

Windows

The following items appear in the **Informix** folder. To display this folder, choose **Start→Programs→Informix Dynamic Server → Documentation Notes or Release Notes** from the task bar.

Program Group Item	Description
Documentation Notes	This item includes additions or corrections to manuals with information about features that might not be covered in the manuals or that have been modified since publication.
Release Notes	This item describes feature differences from earlier versions of IBM Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.

Machine notes do not apply to Windows platforms. ♦

- IBM Informix software products provide ASCII files that contain all of the error messages and their corrective actions. For a detailed description of these error messages, refer to *IBM Informix Error Messages* in the IBM Informix Online Documentation site at http://www.ibm.com/software/data/informix/pubs/library/.

  UNIX

  To read the error messages on UNIX, you can use the **finderr** command to display the error messages online. ♦

  WIN NT

  To read error messages and corrective actions on Windows, use the **Informix Error Messages** utility. To display this utility, choose **Start→Programs→Informix** from the task bar. ♦

Related Reading

The following publications provide related information about the topics discussed in this manual. To learn more about Structured Query Language, consider the following books:

- *A Guide to the SQL Standard* by C. J. Date with H. Darwen (Addison-Wesley Publishing, 1997)
- *SAMS Teach Yourself SQL in 10 Minutes* by Ben Forta (SAMS, 1999)
- *Understanding the New SQL: A Complete Guide* by J. Melton and A. Simon (Morgan Kaufmann Publishers, 1993)

For additional technical information on database management, consult the following books:

- *An Introduction to Database Systems* by C. J. Date (Addison-Wesley Publishing, 1999)
- *Object-Relational Database Development: A Plumber's Guide* by Paul Brown (Prentice-Hall, 2001)
- *Object-Relational DBMSs: The Next Great Wave* by Michael Stonebraker with Dorothy Moore (Morgan Kaufmann Publishers, Inc., 1996)
- *Transaction Processing: Concepts and Techniques* by Jim Gray and Andreas Reuter (Morgan Kaufmann Publishers, Inc., 1993)

To learn more about fundamental concepts and approaches to database design, consult the following books:

- *Database Modeling and Design*, *The Entity-Relationship Approach* by Toby J. Teorey (Morgan Kauffman Publishers, Inc., 1998)
- *Handbook of Relational Database Design* by Candace C. Fleming and Barbara von Halle (Addison-Wesley Publishing Company, 1988)

To learn more about database design for data warehousing, consider the following books:

- *Building the Data Warehouse* by W. H. Inmon (John Wiley & Sons, Inc., 1996)
- *The Data Warehouse Lifecycle Toolkit* by Ralph Kimball (John Wiley & Sons, Inc., 1998)

UNIX

UNIX Manuals

This manual assumes that you are familiar with your computer operating system. If you have limited UNIX system experience, consult your operating-system manual or a good introductory text. The following texts provide information on UNIX systems:

- *Learning the UNIX Operating System* by G. Todino, J. Strang, and J. Peek (O'Reilly & Associates, 1997)
- *A Practical Guide to Solaris* by M. Sobell (Addison-Wesley, 1999)
- *A Practical Guide to the UNIX System* by M. Sobell (Benjamin/Cummings Publishing, 1994)
- *UNIX for Programmers and Users: A Complete Guide* by Graham Glass (Prentice-Hall, 1993)
- *UNIX System Administration Handbook* by Evi Nemeth (Prentice-Hall, 2000)

WIN 2000

Windows 2000 Manuals

The following texts provide information on Windows 2000:

- *Windows 2000 Server Management and Control* by K. L. Spencer and M. Goncalves (Prentice-Hall, 2000)
- *The Ultimate Windows 2000 System Administration Guide* by G. Robert Williams and Mark Walla (Addison-Wesley, 2000)

UNIX/Linux

Linux Manuals

The following texts provide information on Linux:

- *A Practical Guide to Linux* by Mark Sobell (Addison-Wesley, 1997)
- *Linux Administration: A Beginner's Guide* by Steve Shah (McGraw-Hill, 2000)

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for 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 Informix database servers comply with the SQL-92 Intermediate and Full Level and X/Open SQL CAE (common applications environment) standards.

IBM Welcomes Your Comments

To help us with future versions of our manuals, let us know about any corrections or clarifications that you would find useful. Include the following information:

- The name and version of your manual
- Any comments that you have about the manual
- Your name, address, and phone number

Send electronic mail to us at the following address:

docinf@us.ibm.com

This address is reserved for reporting errors and omissions in our documentation. For immediate help with a technical problem, contact Customer Services.

Chapter

1

Introducing Dynamic Server and Client Products

In This Chapter

This chapter provides an overview of IBM Informix Dynamic Server, Version 9.4, IBM Informix Client Software Developer's Kit, and related products. You can order IBM Informix Dynamic Server or IBM Informix Dynamic Server with J/Foundation. For a list of the manuals and description of each product, see "The IBM Informix Documentation Set" on page 6-3.

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, or Windows with online transaction processing (OLTP), data marts, data warehouses, and e-business applications.

Java

IBM Informix Dynamic Server with J/Foundation

If you have Dynamic Server only, you can use IBM Informix JDBC Driver to compile and run Java client programs, but you cannot write *user-defined routines* (UDRs) in Java. A UDR is a routine that an SQL statement, user-defined function, or user-defined procedure can invoke.

To create and use UDRs written in Java, you must install IBM Informix Dynamic Server with J/Foundation and the Java Development Kit (JDK) provided by Sun Microsystems.

J/Foundation includes Java classes, methods, and interfaces that allow you to access databases from within the database server rather than from a client application. If you have Dynamic Server with J/Foundation, you can write UDRs in Java and compile and run both Java client and server-side programs.

For more information, see *J/Foundation Developer's Guide* and *IBM Informix JDBC Driver Programmer's Guide*.

Installation and Migration

For information on how to install the database server products, see the *IBM Informix Dynamic Server Installation Guide for UNIX and Linux* or the *IBM Informix Dynamic Server Installation Guide for Microsoft Windows*.

If you migrate to Dynamic Server, Version 9.4, 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 Developer's Kit
- IBM Informix Connect
- IBM Informix Server Administrator
- JDBC 2.2 with ESQL J 1.01 or JDBC 1.50
- Server Studio 2.00

Dynamic Server with J/Foundation contains these products as well as the J/Foundation product.

BladeManager

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

For more information, see the *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, 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*.

DataBlade Developers Kit

The DataBlade Developer's 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 *DataBlade Developer's Kit User's Guide*.

DB-Access

DB-Access is a client tool that is included with the database server. DB-Access lets you connect to the database server and access, modify, and retrieve database data. To use DB-Access, type **dbaccess** from the command line.

For more information, see the *IBM Informix DB-Access User's Guide*.

High-Performance Loader

The High-Performance Loader (HPL) lets you efficiently load and unload large quantities of data to or from an Informix database. Use the HPL to exchange data with tapes, data files, and programs and to convert data from these sources into a format compatible with an Informix database. The HPL also lets you manipulate and filter the data as you perform load and unload operations.

Use the **ipload** graphical user interface (GUI) to load and unload data, create, modify, and delete HPL objects on UNIX.

Use the **onpladm** utility to perform the same tasks from the command line on either UNIX or Windows platforms. The ipload and **onpladm** utilities are included with the database server.

For information on how to use the HPL, including tutorial examples, see the *IBM Informix High-Performance Loader User's Guide*. For information on other migration utilities, such as **dbexport** and **dbload**, see the *IBM Informix Migration Guide*.

IBM Informix Connect

IBM Informix Connect allows applications that run on client computers to access the database server.

IBM Informix Server Administrator (ISA)

IBM Informix Server Administrator (ISA) is a browser-based tool that provides web-based system administration for Informix database servers. ISA provides access to database server command-line functions and presents the output in an easy-to-read format.

For information on how to install ISA, see your *Installation Guide*. For information on how to use ISA, see the ISA online help and onscreen instructions.

IBM Informix SNMP Subagent

Simple Network Management Protocol (SNMP) is a published, open standard for network management that is included with the database server. The IBM Informix SNMP subagent provides information about Informix database servers to SNMP-compliant applications.

For more information, see the *IBM Informix SNMP Subagent Guide*.

Optical Subsystem

The Optical Subsystem is included with the database server and stores *simple large objects* (TEXT and BYTE data) on optical platters known as WORM optical media. The optical media are the removable optical platters that contain data. The Optical Subsystem includes a specific set of SQL statements that allow you to store and retrieve data stored on optical disks.

The Optical Subsystem does not store character large object (CLOB) and binary large object (BLOB) data types, also known as *smart large objects*.

For more information, see the *IBM Informix Optical Subsystem Guide*.

Related IBM Informix Products

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

Client SDK Products

The IBM Informix Client Software Developer's 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. 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*.

E/C

IBM Informix ESQL/C

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:

- ESQL/C libraries of C functions for accessing the database server
- 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
- Windows ESQL client-interface *dynamic link libraries* (DLLs), which let an ESQL/C application run in 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 1-9.

GLS

IBM Informix GLS

Global Language Support (GLS) allows you to create databases that use the diacritics, collating sequence, and monetary and time conventions of the language that you select. The IBM Informix GLS library contains APIs that let programmers develop internationalized ESQL/C and DataBlade module client applications. IBM Informix GLS is separately orderable, but the GLS libraries are shipped with the database server and Client SDK.

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 GLS provides an HTML reference that you can access with a web browser. The URL must include the full pathname of the directory that your **INFORMIXDIR** environment variable designates:
$INFORMIXDIR/doc/gls_api/en_us/0333/index.htm on UNIX or
%INFORMIXDIR%\doc\gls_api\en_us\04e4\index.htm on Windows.

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 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*.

Windows

IBM Informix OLE DB Provider

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*.

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 *TP/XA Programmer's Manual*.

UNIX

IBM Informix MaxConnect

IBM Informix MaxConnect is a networking product for Informix database servers on UNIX. Two protocols for multiplexing connections, **ontliimc** and **onsocimc**, are available for MaxConnect users. 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. MaxConnect increases system scalability to many thousands of connections and saves system resources, reducing response times and CPU requirements. You can install MaxConnect on the client application server, on a dedicated server, or on the database server computer.

For more information, see the *IBM Informix MaxConnect*.

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 Office Connect User's Guide*.

Server Studio JE

Server Studio JE, Version 2.0.JC1, is a stand-alone, Java-based Integrated Development Environment (IDE) for the 7.3x and 9.x database servers. Server Studio contains the following modules:

- Database Object Explorer
- Properties Inspector
- SQL Editor
- Table Editor

These modules are provided free of charge. Additional modules are provided with the distribution of Server Studio JE on a try and buy basis. Contact Advanced Global Systems LTD (AGS) to obtain a licence for the additional modules: www.agsltd.com.

Server Studio replaces IBM Informix Database Administrator.

DataBlade Modules

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

- Excalibur Image DataBlade module
- Excalibur Text Search DataBlade module
- IBM Informix Geodetic DataBlade module
- IBM Informix Spatial DataBlade module
- IBM Informix TimeSeries DataBlade module
- Verity Text Search DataBlade module
- IBM Informix Video Foundation DataBlade module
- IBM Informix Web DataBlade module

For a brief description of each of these, see "DataBlade Manuals" on page 6-9.

Related IBM Products

The following IBM products are among those that you can use with IBM Informix Dynamic Server:

- IBM WebSphere Application Server

  Controls interactions with enterprise information systems, including IBM Informix Dynamic Server.

- IBM DB2 Web Query Tool

  Connects users directly to multiple enterprise databases, including IBM Informix Dynamic Server.

- IBM DB2 Relational Connect

  Queries and retrieves information from IBM Informix Dynamic Server and other database servers.

- Tivoli TME 10 NetView

  Acts as an SNMP Network Manager with IBM Informix Dynamic Server.

GLS

Global Language Support

The Global Language Support (GLS) feature lets the database server handle different languages, cultural conventions, and code sets using different locales. A GLS locale is an environment that has defined conventions for a particular language or culture. See "Assumptions About Your Locale" on page 4 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.

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

Chapter

2

Using New Features in Dynamic Server

In This Chapter

This chapter describes the new features in Dynamic Server, Version 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 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
- Changed or New URLs

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 the openSSL 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 2-11.

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 Dynamic Server 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 2-18)
- The **onspaces** utility
- The **ontape** utility (see "Full Use of Storage Media and Increased File Size Limit" on page 2-26)
- 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 support by the following utilities:

- The **ontape** utility (see "Full Use of Storage Media and Increased File Size Limit" on page 2-26)
- 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 2-25.

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 *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 2-13.

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, MULTIST, 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_CIPHER 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 re-negotiated.

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 2-19.

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 2-14, 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 2-14," 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 (SVLs) 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 was 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 1024 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 2-11 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 2-18.

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 GB1839-2000 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	Manual
Return the value of the current database server locale.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_get_db_locale( )** function.	*IBM Informix DataBlade API Function Reference*

New mi_get_transaction_id( ) Function

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

Task	Manual
Return the ID of the current transaction.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_get_transaction_id( )** function.	*IBM Informix DataBlade API Function Reference*

New mi_realloc( ) function

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

Task	Manual
Change the size of an existing memory block.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_realloc( )** function.	*IBM Informix DataBlade API Function Reference*

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	Manual
Determine whether the current user stack has the specified amount of free space.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_stack_limit( )** function.	*IBM Informix DataBlade API Function Reference*

New mi_system( ) Function

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

Task	Manual
Use the **mi_system( )** function.	*IBM Informix DataBlade API Function Reference*

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 and Linux*.

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 and Linux*.

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 and Linux* and the *IBM Informix Dynamic Server Installation Guide for Microsoft Windows*.

Changed or New URLs

The URLs for IBM Informix products have changed. The URLs that are listed in new IBM Informix Dynamic Server manuals have been updated for Version 9.4. You will be able to find the same or updated information as before at the following Web sites:

The home page for the IBM Informix product family:

http://www.ibm.com/software/data/informix

The IBM Informix Online Documentation site (the former Answers Online):

http://www.ibm.com/software/data/informix/pubs/library/

IBM Informix information resource for online documentation:

http://www.informix.com/informix/resource

The former Informix Developer Network Web site, now the IBM Informix Developer Zone:

www.ibm.com/software/data/developer/informix

The former Informix TechInfo Center, now IBM Software Online Support:

www.ibm.com/software/data/informix/support

The IBM Informix Developer Zone Systems Management Corner:

http://www7b.software.ibm.com/dmdd/zones/informix/corner_sm.html

Porting and Certification for IBM Informix database servers, including product compatibility:

http://www.ibm.com/software/data/informix/pubs/smv/index.html

For information about the IBM Informix DataBlade modules:

http://www.ibm.com/software/data/informix/blades/

In addition, to report problems or to provide comments about IBM Informix user documentation, you can contact IBM Informix Information Development group by electronic mail at:

docinf@us.ibm.com

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/Linux

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 and Linux*.

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 *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 *Administrator's Reference*.

Windows

Database Server Administration Utilities

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 *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 *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	Manual
Use the new memory durations.	*IBM Informix DataBlade API Programmer's Guide*
Use memory durations in functions that have a duration argument.	*IBM Informix DataBlade API Function Reference*
Use **mi_dalloc( )** to specify a memory duration.	*IBM Informix Virtual-Table Interface Programmer's Guide*
Allocate user-data memory with PER_STMT_EXEC memory duration.	*IBM Informix Virtual-Table Interface Programmer's Guide*
Display information on the **PRP.*sessionid.threadid*** and **EXE.*sessionid.threadid*** pools.	*Administrator's Reference* (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	Manual
Pass a NULL connection to an **mi_lo*( )** function.	*IBM Informix DataBlade API Programmer's Guide*
Specify a NULL connection in a call to an **mi_lo*( )** routine that takes a connection descriptor.	*IBM Informix DataBlade API Function Reference*

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	Manual
Determine the cardinality of a collection.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_collection_card( )** function.	*IBM Informix DataBlade API Function Reference*
Use LIST, MULTISET, and SET data types.	*IBM Informix Guide to SQL: Reference*

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	Manual
Access client files one buffer at a time.	*IBM Informix DataBlade API Programmer's Guide*
Open a file on the client by passing the MI_O_CLIENT_FILE flag to **mi_file_open( )**.	*IBM Informix DataBlade API Function Reference*

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 DataBlades

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	Manual
Determine the state of a transaction.	*IBM Informix DataBlade API Programmer's Guide*
Use the **mi_transaction_state( )** function.	*IBM Informix DataBlade API Function Reference*

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 *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

For more information on which user-defined types are not supported, see the release notes file that is distributed with Dynamic Server. For the pathname of the release notes file, refer to "Additional Documentation" on page 10 in the Introduction.

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	Manual
Prepare tables for conflict resolution.	*IBM Informix Dynamic Server Enterprise Replication Guide*
Add or drop shadow columns: ■ ALTER TABLE . . . ADD CRCOLS ■ ALTER TABLE . . . DROP CRCOLS	*IBM Informix Guide to SQL: Syntax*
Understand the performance advantages of in-place alters and when they occur.	*Performance Guide*

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	Manual
Use multirepresentational data types.	*IBM Informix User-Defined Routines and Data Types Developer's Guide*
Use the **DeepCopy** function in a UDR.	*IBM Informix DataBlade API Programmer's Guide* *IBM Informix User-Defined Routines and Data Types Developer's Guide*

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	Manual
Use temporary sbspaces and smart large objects.	*Administrator's Guide*
Use **onspaces** to create temporary sbspaces.	*Administrator's Reference*
Improve temporary space utilization.	*Performance Guide*

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	Manual
Monitor the metadata and user-data areas.	*Administrator's Guide*
Read about sbspace structure.	*Administrator's Reference*
Estimate the size of the metadata area and improve space utilization.	*Performance Guide*

Java

J/Foundation Enhancements

If Dynamic Server with J/Foundation is installed, 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 2-48.

JVM 1.3 Support in J/Foundation

Dynamic Server with J/Foundation 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	Manual
Configure lock mode.	*Performance Guide*
Use the DEF_TABLE_LOCKMODE configuration parameter.	*Administrator's Reference*
Use the LOCK MODE clause in the ALTER TABLE or CREATE TABLE statement.	*IBM Informix Guide to SQL: Syntax*

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 *Performance Guide* and the *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	Manual
Improve performance of queries and use optimizer directives.	*Performance Guide*
Use SET EXPLAIN and optimizer directives.	*IBM Informix Guide to SQL: Syntax*

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	Manual
Use dynamically allocated logical logs.	*Administrator's Guide*
Use the **onparams** and **onstat -l** commands, and the DYNAMIC_LOGS, LTXHWM, and LTXEHWM parameters.	*Administrator's Reference*

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*.

Features from Dynamic Server, Version 9.21

These features were introduced in IBM Informix Dynamic Server, 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	Manual
Use the ANSI join syntax.	*IBM Informix Guide to SQL: Syntax*
Use join filters and post-join filters, and interpret SET EXPLAIN output for ANSI joins.	*Performance Guide*

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 cannot have indexes or referential constraints but can be updated. You can create either a standard or raw table and change tables from one type to another.

Task	Manual
Use nonlogging tables.	*Administrator's Guide*
Load and unload nonlogging tables. Lock nonlogging tables.	*Performance Guide*
Specify the logging type in the ALTER TABLE and CREATE TABLE statements.	*IBM Informix Guide to SQL: Syntax*

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 the **onpladm.htm** file included with the database server:

```
$INFORMIXDIR/release/en_us/0333/onpladm/index.html
```

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 *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	**Manual**
Learn about the SQL statement cache	*Performance Guide*
Use qualifying and identical statements; also learn about the memory limits and key-only cache entries.	*IBM Informix Guide to SQL: Syntax*
Configure the SQL statement cache: ■ STMT_CACHE_HITS ■ STMT_CACHE_NOLIMIT ■ STMT_CACHE_NUMPOOL ■ STMT_CACHE_SIZE	*Administrator's Reference*

(1 of 2)

Task	Manual
Display SQL statement cache statistics: ■ **onstat -g ssc** ■ **onstat -g ssc all** ■ **onstat -g ssc pool**	*Administrator's Reference*
Configure the SQL statement cache on the fly: ■ **onmode -W STMT_CACHE_HITS** ■ **onmode -W STMT_CACHE_NOLIMIT** ■ **onmode -W STMT_CACHE_SIZE**	*Administrator's Reference*
Understand the performance advantages of the SQL statement cache: ■ Use the SQL statement cache ■ Enable the SQL statement cache ■ Configure the SQL statement cache ■ Monitor the SQL statement cache	*Performance Guide*

(2 of 2)

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

Java Features in 9.21

The following Java features were introduced in Version 9.21.

JVM 1.2 Support in J/Foundation

Dynamic Server with J/Foundation 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 with J/Foundation.

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 with J/Foundation 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 with J/Foundation supports the following GLS features:

- **CLIENT_LOCALE**, **DB_LOCALE**, **GL_DATE**, **GL_DATETIME**, **DBTIME**, and **DB_CENTURY** 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 with J/Foundation 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 with J/Foundation 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 manuals.

Task	Manual
Use JVM 1.2. Use JCBC 2.0 features. Write UDRs and DataBlade modules in Java.	*J/Foundation Developer's Guide*
Specify Java environment variables.	*J/Foundation Developer's Guide* *IBM Informix Guide to SQL: Reference*
Specify Java configuration parameters.	*J/Foundation Developer's Guide* *IBM Informix Dynamic Server Administrator's Reference*
Set GLS environment variables. Use the connection properties.	*J/Foundation Developer's Guide* *IBM Informix GLS User's Guide*
Use the **update_jars.sql** script.	*IBM Informix Guide to SQL: Syntax*

MaxConnect Support

IBM Informix MaxConnect enables IBM Informix Dynamic Server to support greatly increased numbers of client connections. 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: *MaxConnect and the "IBM Informix MaxConnect" ship separately from IBM Informix Dynamic Server, 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 MaxConnect with two new network protocols: **ontliimc** and **onsocimc**.

- New utility options to monitor MaxConnect
  - **onstat -g imc**
  - **imcadmin**
  - **ISA options**
- New environment variables for MaxConnect
  - **IMCADMIN**
  - **IMCCONFIG**
  - **IMCSERVER**

For more information about installing, configuring, monitoring, and tuning MaxConnect, see the *IBM Informix MaxConnect*.

Chapter

3 Using Existing Dynamic Server Features

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 3-21.

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 *Performance Guide*. For information on how to use Dynamic Server, see the *Administrator's Guide* and the *Administrator's Reference*. For a glossary of terms that are used in IBM Informix manuals, 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.

UNIX

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. ♦

Windows

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 blobspace
Sbpage	The physical unit of disk storage to store smart large objects in an sbspace
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
Dbspace	Stores databases, tables, logical-log files, the physical log, and internal data
Blobspace	Stores simple large objects (TEXT and BYTE data)
Sbspace	Stores smart large objects (CLOB and BLOB data)

(1 of 2)

Logical Storage Unit	Description
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
Tblspace	Contains the disk space allocated to a given table or fragment

(2 of 2)

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 *before image*)

For information about storage spaces and logical and physical logs, see the *Administrator's Guide*. Logical-log record formats are discussed in the *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)
- 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" on page 1-10.

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	✔	✔	✔	✔
TLI (TCP/IP)		✔	✔	✔
TLI (IPX/SPX)		✔	✔	✔

Connection Type	Windows	UNIX	Local	Network
Shared memory		✔	✔	
Stream pipe		✔	✔	
Named pipe	✔		✔	

For information about client/server configurations that the database server supports, see the *Administrator's Guide*. For instructions on how to use client applications, see the appropriate programmer's manual, as listed in Chapter 6, "Using the Documentation."

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 *Performance Guide* and "SQL Enhancements" on page 2-43. For information on transaction logging, see the *Administrator's Guide*.

Using NFS-Mounted Directories

An IBM Informix storage space can reside on an NFS-mounted directory only if the vendor of that NFS device is certified by IBM Informix. For information about NFS products that you can use to mount a storage space for an IBM Informix database server, see the IBM Informix product family Web site at http://www.ibm.com/software/data/informix.

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 *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 *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 *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 Data Replication (HDR)
- 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* re-creates 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 *Backup and Restore Guide*, "The onbar Utility" on page A-10, and "The ontape Utility" on page A-25.

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* and "The ism Utility" on page A-8.

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 *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 *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 Data Replication

High-Availability Data Replication (HDR) provides synchronous data replication for Dynamic Server. HDR allows you to replicate database data running simultaneously on a second computer. If one site experiences a disaster, you can immediately direct applications to use the second database server in the data-replication pair.

For information about HDR, see the *Administrator's Guide*.

HDR 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 HDR.

For more information, see the *IBM Informix Dynamic Server Enterprise Replication Guide* and "The cdr Utility" on page A-4.

Types of Data That You Can Replicate

Figure 3-1 on page 3-14 shows the types of data that you can replicate with HDR or Enterprise Replication.

Figure 3-1
Data Types that HDR and ER Replicate

Data Type	HDR 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

(1 of 2)

Data Type	**HDR Support**	**ER Support**
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*.

(2 of 2)

Database Server Security

The databases and tables that the database server manages enforce access based on a set of database and table privileges. You can use the following SQL statements to manage these privileges:

- Use the GRANT and REVOKE statements to give or deny access to a database or specific tables and to control the kinds of database uses.
- Use the CREATE PROCEDURE statement to write and compile a stored procedure that controls and monitors access to tables.
- Use the CREATE VIEW statement to prepare a restricted or modified view of the data.
- Use the CREATE ROLE statement to set up classifications with privileges on database objects granted to a specific role.

Dynamic Server also lets you audit database events. Auditing allows you to track which users performed specific actions to particular objects at distinct times. You can use this information to monitor database activity for suspicious use, deter unscrupulous users, or even as evidence of database server abuse.

For information about database and table privileges and controlling access to databases, see the *IBM Informix Database Design and Implementation Guide*. For the syntax and description of SQL statements, see the *IBM Informix Guide to SQL: Syntax*. For information on auditing, see the *Trusted Facility 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 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 *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 the following objects:

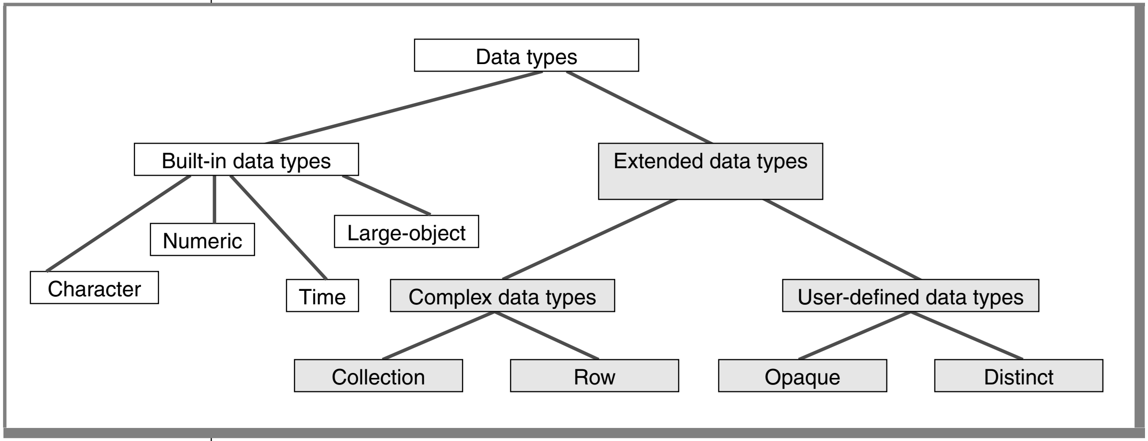
- Tables, views, sequences, and synonyms
- Constraints and indexes
- Triggers
- Authorized users and privileges
- User-defined routines
- Data types and casts
- Aggregates and modifiers
- Access methods and operator classes
- Inheritance relationships

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 3-2 shows.

Figure 3-2
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.*

Figure 3-3 describes the data types that you can define for a database.

Figure 3-3
Data Types

Data Type	Explanation	Examples
Built-in data types	Fundamental data types that cannot be broken into smaller pieces Serve as building blocks for other data types.	■ BLOB ■ BOOLEAN ■ BYTE ■ CHAR(*n*) ■ CHARACTERVARYING(*m,r*) ■ CLOB ■ DATE ■ DATETIME ■ DECIMAL or NUMERIC(*p,s*) ■ DOUBLE PRECISION ■ FLOAT ■ INTEGER ■ INTERVAL ■ LVARCHAR(*m*) ■ MONEY(*p,s*) ■ NCHAR(*n*) ■ NVARCHAR(*m,r*) ■ REAL or SMALLFLOAT ■ SERIAL ■ SERIAL8 ■ SMALLINT ■ TEXT ■ VARCHAR(*n,r*)
Complex data types	Combination of other data types An SQL statement can access individual components within the complex type.	

(1 of 2)

Data Type	Explanation	Examples
Collection types	Complex data types Include groups of elements of the same data type, which can be any built-in or complex data type.	■ SET ■ LIST ■ MULTISET
Row types	Complex data types Include groups of related data *fields* of any data type that form a template for a record.	Named row type Unnamed row type
User-defined data types	Include distinct and opaque types	
Distinct data types	Have the same internal structure as existing data types They have distinct names and functions that make them different from the source type.	`CREATE DISTINCT TYPE birthday AS DATE` The data type is **birthday**.
Opaque data types	User-defined types The internal structure is not known to the database server.	`CREATE OPAQUE TYPE fixlen_typ(INTERNAL-LENGTH=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: Excalibur Image DataBlade module Excalibur Text Search DataBlade module IBM Informix Geodetic DataBlade module IBM Informix TimeSeries DataBlade module IBM Informix Video Foundation DataBlade module IBM Informix Web DataBlade module Unicode DataBlade module Verity Text Search DataBlade module

(2 of 2)

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 *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 3-31)

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 sbspace* 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 ESQL/C programming interface.

For information on sbspaces, see the *Administrator's Guide*. For information on how to create an sbspace, see the discussion of **onspaces** in the *Administrator's Reference* and "The onspaces Utility" on page A-19. For information on how to calculate space and tune sbspaces, see your *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 ESQL/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 *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 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 DataBlade Developer's 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 *DataBlade Developer's 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 Databases and Queries

Dynamic Server supports distributed queries across multiple databases and multiple database servers for transactions that involve only built-in data types. 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 *Administrator's Guide*. For information about a specific IBM Informix Enterprise Gateway product, see the appropriate *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 *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 onspaces Utility" on page A-19, the *Administrator's Guide*, and *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 sbspace. 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

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.
- **Manual**. This column lists the book that contains information to help you perform the task.

Database Server Users

Figure 4-1 shows the major groups of database server users.

Figure 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	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.

(1 of 2)

User	Duties
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.

(2 of 2)

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).

Figure 4-2 on page 4-5 describes the planning, installation, and configuration tasks.

Figure 4-2
Planning, Installation, and Configuration Tasks

If You Want To	Manual
Learn about the new features in Dynamic Server.	Chapter 2, "Using New Features in Dynamic Server"
Acquaint yourself with terms used in IBM Informix manuals.	*IBM Informix Guide to SQL: Reference*
Interpret error messages.	*IBM Informix Error Messages* or **finderr** utility
Plan a database server installation.	*Administrator's Guide*
Plan and configure: ■ Operating system ■ Hardware and system software upgrades ■ Network capacity ■ Integration with other vendor products and applications ■ Disk and storage media	System documentation
Determine optimal memory configuration.	*Performance Guide*
Determine optimal disk layout and striping.	System documentation
Install Dynamic Server on UNIX or Linux: ■ Standard installation ■ Silent installation ■ Private installation	*IBM Informix Dynamic Server Installation Guide for UNIX and Linux*
Install Dynamic Server on Windows: ■ Standard installation ■ Silent installation ■ Multiple residency ■ Cluster installation	*IBM Informix Dynamic Server Installation Guide for Microsoft Windows*
Initialize the database server. Manage database server operating modes.	*Administrator's Guide*

(1 of 3)

If You Want To	Manual
Test the database server connection with DB-Access.	*IBM Informix DB-Access User's Guide*
Install and configure client applications.	*IBM Informix Client Products Installation Guide*
Install and configure DataBlade modules (optional).	*DataBlade Module Installation and Registration Guide*
Install and configure MaxConnect (optional).	*IBM Informix MaxConnect*
Configure the database server manually: ■ 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.	*Administrator's Guide*
Use **Server Setup** in IBM Informix Server Administrator to configure the database server.	ISA online help
Create dbspaces, blobspaces, and sbspaces. Resolve incorrect chunk permissions and ownership. Design and implement logical and physical logs. Implement mirroring.	*Administrator's Guide*
Configure the ON-Bar or **ontape** backup and restore system.	*IBM Informix Backup and Restore Guide*
Set up IBM Informix Storage Manager. Set up the storage volumes and storage devices.	*IBM Informix Storage Manager Administrator's Guide*
Set up a third-party storage manager (optional).	Your storage-manager documentation

(2 of 3)

If You Want To	Manual
Design and set up the Enterprise Replication system.	*IBM Informix Dynamic Server Enterprise Replication Guide*
Design and set up High-Availability Data Replication (HDR).	*Administrator's Guide*
Prepare the old database server version for migration. 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.	*IBM Informix Migration Guide*

(3 of 3)

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.

Figure 4-3 lists the administration tasks and where to find information on those tasks.

Figure 4-3
Administration Tasks

If You Want To	Manual
Monitor an Informix database server. 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.	*Administrator's Guide*
Use ISA to administer and monitor the database server.	ISA online help
Use the following utilities to perform administrative tasks: ■ **oncheck** ■ **ondblog** ■ **oninit** ■ **onlog** ■ **onmode** ■ ON-Monitor ■ **onparams** ■ **onspaces** ■ **onstat**	*Administrator's Reference*
Locate information on the configuration parameters. Use the SMI tables of the **sysmaster** database to monitor the database server.	*Administrator's Reference*

(1 of 3)

If You Want To	Manual
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.	*Administrator's Reference*
Use the ON-Bar or **ontape** utility. Back up and restore storage spaces and logical logs. Use the **archecker** utility to verify backed-up data. Perform an external backup and restore. Use ON-Bar or external backup with HDR.	*Backup and Restore Guide*
Use ISA to administer and monitor the database server.	ISA online help
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.	*IBM Informix Storage Manager Administrator's Guide*
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. Alter the schema of a table.	*IBM Informix High-Performance Loader User's Guide*
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.	*Trusted Facility Guide*

(2 of 3)

If You Want To	Manual
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.	*IBM Informix Optical Subsystem Guide*
Use the IBM Informix SNMP subagent to extract information from an Informix database server and pass that information to a network manager.	*IBM Informix SNMP Subagent Guide*
Design, define, monitor, and control your Enterprise Replication system.	*IBM Informix Dynamic Server Enterprise Replication Guide*
Set up locales for different languages, cultural conventions, and code sets.	*IBM Informix GLS User's Guide*

(3 of 3)

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.

Figure 4-4 lists performance-related tasks and where to find information on those tasks.

Figure 4-4
Performance Tuning Tasks

If You Want To	Manual
Use different types of tables (STANDARD, RAW, TEMP).	*Administrator's Guide*
Use the **onstat -g** utilities to monitor database server performance.	*Administrator's Reference*
Improve backup and restore performance.	*Backup and Restore Guide*
Query the system catalog tables.	*IBM Informix Guide to SQL: Reference*

(1 of 2)

If You Want To	Manual
Adjust the database server configuration.	*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 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.	

(2 of 2)

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.

Figure 4-5 describes the diagnostic tools available for troubleshooting database operations and the database server.

Figure 4-5
Troubleshooting Tasks

If You Want To	Manual
Use the **onstat -g** utilities to diagnose database server problems. 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.	*Administrator's Reference*
Use the **archecker** utility to verify and diagnose problems with backups. Find corrective actions to ON-Bar return codes.	*Backup and Restore Guide*
Find corrective actions to numbered error messages and ON-Bar messages.	*IBM Informix Error Messages*
Fix data replication problems.	*IBM Informix Dynamic Server Enterprise Replication Guide*

Chapter

5 Designing, Maintaining, and Extending the Database

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

Figure 5-1 lists the tasks for designing, developing, and extending the database.

Figure 5-1
Database Tasks

If You Want To	Manual
Work with the tables in the **sysmaster** database.	*Administrator's Reference*
Place tables on disk.	*Performance Guide*
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.	*IBM Informix Error Messages*

(1 of 3)

If You Want To	Manual
Design a database (choose whether to implement the relational, object-relational, or dimensional database model). Build a relational or object-relational database: ■ 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.	*IBM Informix Database Design and Implementation Guide*
Invoke the DB-Access utility. 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.	*IBM Informix DB-Access User's Guide*
Learn how GLS affects database server migration. Load and unload data. Display the database schema with **dbschema**.	*IBM Informix Migration Guide*

(2 of 3)

If You Want To	**Manual**
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.	*IBM Informix Guide to SQL: Reference*
Create databases and manage access. 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.	*IBM Informix Guide to SQL: Syntax*
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.	*IBM Informix Guide to SQL: Tutorial*
Assign data types to columns.	*IBM Informix Database Design and Implementation Guide* *IBM Informix Guide to SQL: Reference* *IBM Informix Guide to SQL: Syntax* *IBM Informix Guide to SQL: Tutorial*
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.	*IBM Informix Optical Subsystem Guide*

(3 of 3)

Developing Application Programs that Access the Database

Figure 5-2 lists the tasks for developing, compiling, and running client application programs and DataBlade modules that access database server data.

Figure 5-2
Application Development Tasks

If You Want To	Manual
Test database applications that you intend to store for use in a production environment.	*IBM Informix DB-Access User's Guide*
Write procedures with SPL and store them in a database. Use a primary access method.	*IBM Informix Guide to SQL: Syntax*
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.	*IBM Informix Guide to SQL: Tutorial*
Use IBM Informix ODBC Driver to access relational databases with SQL. Create custom applications with IBM Informix ODBC API functions.	*IBM Informix ODBC Driver Programmer's Manual*
Embed SQL statements directly into C programs.	*IBM Informix ESQL/C Programmer's Manual*
Create new data types and user-defined routines using Java.	*J/Foundation Developer's Guide*
Use the GLS features that let Informix SQL APIs and database servers handle different languages, cultural conventions, and code sets.	*IBM Informix GLS User's Guide*
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.	*TP/XA Programmer's Manual*

(1 of 3)

If You Want To	Manual
Create new data types and user-defined routines using C. 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.	*IBM Informix User-Defined Routines and Data Types Developer's Guide*
Use DataBlade API functions to develop server and client applications that access data stored in a Dynamic Server database. Write server routines and client LIBMI applications that use smart large objects and complex and extended data types.	*IBM Informix DataBlade API Programmer's Guide*
Use DataBlade API functions. Use ESQL/C functions with the DataBlade API.	*IBM Informix DataBlade API Function Reference*
Use Java to create client applications or applets that run against Dynamic Server. 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.	*IBM Informix JDBC Driver Programmer's Guide*
Use the object-oriented C++ programming language to create database client applications for Informix database servers. Use Object Interface for C++ to create value objects that let C++ client applications support DataBlade module data types.	*IBM Informix Object Interface for C++ Programmer's Guide*
Work with the R-tree secondary access method.	*IBM Informix R-Tree Index User's Guide*

(2 of 3)

If You Want To	Manual
Develop a secondary access method with the Virtual-Index Interface (VII) to create new types of indexes. Use functions in the VII library.	*IBM Informix Virtual-Index Interface Programmer's Guide*
Develop a primary access method with the Virtual-Table Interface (VTI) so that users can access external data. Use functions in the VTI library.	*IBM Informix Virtual-Table Interface Programmer's Guide*
Develop applications using DataBlade modules.	"DataBlade Manuals" on page 6-9

(3 of 3)

Chapter

Using the Documentation

6

In This Chapter

This chapter contains an alphabetical list of the IBM Informix manuals provided:

- "IBM Informix Dynamic Server Manuals" on page 6-3
- "Client SDK and Connectivity Manuals" on page 6-7
- "DataBlade Manuals" on page 6-9

The IBM Informix Documentation Set

This set describes all the manuals available with the database server, client products, tools, and DataBlade modules.

IBM Informix Dynamic Server Manuals

Figure 6-1 summarizes the documentation that is available with Dynamic Server.

Figure 6-1
Database Server Manuals

Book Title	**Description**
IBM Informix Backup and Restore Guide	This manual 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.
IBM Informix Database Design and Implementation Guide	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.
IBM Informix DataBlade API Function Reference	This manual describes the API functions.
IBM Informix DataBlade API Programmer's Guide	This manual 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.
IBM Informix DB-Access User's Guide	This guide describes how to use the DB-Access utility to access, modify, and retrieve information from Informix database servers.
IBM Informix Dynamic Server Administrator's Guide	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 manual name is *Administrator's Guide*.
IBM Informix Dynamic Server Administrator's Reference	This reference manual 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.
IBM Informix Dynamic Server Getting Started Guide	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.

(1 of 4)

Book Title	**Description**
IBM Informix Dynamic Server Enterprise Replication Guide	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.
IBM Informix Dynamic Server Installation Guide for UNIX and Linux	This guide contains instructions for installing Dynamic Server on UNIX and Linux. It also describes how to solve common installation problems.
IBM Informix Dynamic Server Installation Guide for Microsoft Windows	This guide contains instructions for installing Dynamic Server on Windows.
IBM Informix Dynamic Server Performance Guide	This guide explains how to configure and operate Dynamic Server to achieve optimum performance and optimize SQL queries. The short manual name is *Performance Guide*.
IBM Informix Error Messages	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.
IBM Informix GLS User's Guide	This manual describes the Global Language Support (GLS), which allows IBM Informix client products and database servers to handle different languages, cultural conventions, and code sets.
IBM Informix Guide to SQL: Reference	This manual describes the Informix system catalog tables, data types, environment variables, the **stores_demo** and **superstores_demo** databases. It also contains a glossary.
IBM Informix Guide to SQL: Syntax	This manual contains the complete syntax descriptions of all Informix (SQL) and Stored Procedure Language (SPL) statements, and functions.
IBM Informix Guide to SQL: Tutorial	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.

(2 of 4)

Book Title	Description
IBM Informix High-Performance Loader User's Guide	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.
IBM Informix Migration Guide	This manual 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**.
IBM Informix Optical Subsystem Guide	This guide describes how to use the Optical Subsystem, a utility that supports the storage of TEXT and BYTE data on optical disks.
IBM Informix R-Tree Index User's Guide	This guide describes the R-tree secondary access method and how to create an R-tree index on user-defined types.
IBM Informix SNMP Subagent Guide	This manual 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.
IBM Informix Storage Manager Administrator's Guide	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.
IBM Informix Trusted Facility Guide	This guide describes the secure auditing facility and includes information on how to set up and administer audit trails, and extract and interpret audit records.
IBM Informix User-Defined Routines and Data Types Developer's Guide	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.

(3 of 4)

Book Title	Description
IBM Informix Virtual-Index Interface Programmer's Guide	This manual explains how to use the Virtual-Index Interface (VII), typically in a DataBlade module, to create a secondary access method. A *virtual* index accesses data from a source outside the database server or specific data inside large objects. The manual describes the syntax, API function calls, and data structures.
IBM Informix Virtual-Table Interface Programmer's Guide	This manual explains how to create a primary access method with the Virtual-Table Interface (VTI). A *virtual* table is dynamically created from a source outside the database server or specific data inside of large objects. The manual describes the syntax, API function calls, and data structures.
J/Foundation Developer's Guide	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 UDRs that access Dynamic Server with J/Foundation.

(4 of 4)

Client SDK and Connectivity Manuals

Figure 6-2 lists the IBM Informix Client SDK and connectivity manuals that you can use when you work with Dynamic Server, Version 9.4.

Figure 6-2
Client Manuals for Dynamic Server

Book Title	**Description**
IBM Informix Client Products Installation Guide	This guide contains instructions for installing the IBM Informix Client Software Developer's Kit and IBM Informix Connect on Linux, UNIX, and Windows.
IBM Informix Embedded SQLJ User's Guide	This manual 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.
IBM Informix ESQL/C Programmer's Manual	This manual explains how to use ESQL/C, the Informix implementation of embedded SQL for C, to create client applications with database-management capabilities.
IBM Informix JDBC Driver Programmer's Guide	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.
IBM Informix MaxConnect	This manual describes how to install, administer, and tune performance for MaxConnect. MaxConnect enables the database server to support nearly unlimited numbers of client connections and reduces response times and CPU usage.
IBM Informix Object Interface for C++ Programmer's Guide	This guide describes how C++ and DataBlade developers can develop IBM Informix client applications with the C++ programming language.
IBM Informix *Object Translator*	This online help describes how to use the IBM Informix Object Translator, a client-side tool, to integrate object-based applications and XML documents into the database.
IBM Informix ODBC Driver Programmer's Manual	This manual 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.

(1 of 2)

Book Title	Description
IBM Informix OLE DB Provider Programmer's Guide	This manual 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.
IBM Office Connect User's Guide	This manual describes how to populate Excel worksheets with data from Informix databases.
TP/XA Programmer's Manual	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.

(2 of 2)

DataBlade Manuals

Figure 6-3 lists the manuals that you can use when you develop or use DataBlade modules and web-based applications with Dynamic Server or Dynamic Server with J/Foundation, Version 9.4.

Figure 6-3
DataBlade and Tools Manuals

Book Title	Description
DataBlade Developer's Kit User's Guide	This guide describes how to develop and package DataBlade modules using BladeSmith and BladePack.
DataBlade Module Development Overview	This manual provides an overview of DataBlade module development.
DataBlade Module Installation and Registration Guide	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.
Excalibur Image DataBlade Module User's Guide	This manual explains how to use the Excalibur Image DataBlade module to store, process, and retrieve images for customers who want to incorporate images into their database systems.

(1 of 2)

Book Title	Description
Excalibur Text Search DataBlade Module User's Guide	This manual explains how to perform text searches and retrieval using the Excalibur Text Search DataBlade module.
IBM Informix Data Director for Web Tutorial	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.
IBM Informix Data Director for Web User's Guide	This manual 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.
IBM Informix Geodetic DataBlade Module User's Guide	This manual explains how to use the IBM Informix Geodetic DataBlade module to store and use spatio-temporal data such as maps.
IBM Informix Large Object Locator DataBlade Module User's Guide	This guide explains how to use the IBM Informix Large Object Locator DataBlade module to locate large objects that are stored outside the database. IBM Informix Large Object Locator DataBlade module is included with the database server.
IBM Informix Spatial DataBlade Module User's Guide	This guide explains how to use the IBM Informix Spatial DataBlade module to store, manipulate, index, and analyze multidimensional spatial data.
IBM Informix TimeSeries DataBlade Module User's Guide	This manual explains how to use the IBM Informix TimeSeries DataBlade module to store and manage time-stamped data such as stock reports.
IBM Informix Video Foundation DataBlade Module User's Guide	This manual describes how to use the IBM Informix Video Foundation DataBlade module to store video technology in a media management system.
IBM Informix Web DataBlade Module Administrator's Guide	This manual describes how to administer web applications that use the IBM Informix Web DataBlade module to dynamically retrieve data from Informix databases.
IBM Informix Web DataBlade Module Application Developer's Guide	This manual explains how to use the IBM Informix Web DataBlade module to develop web applications that dynamically retrieve data from Informix databases.
Verity Text Search DataBlade Module User's Guide	This manual explains how to use the Verity Text Search DataBlade to search and retrieve text using SQL statements.

(2 of 2)

Database Server Utilities

Database Server Utilities

Dynamic Server includes the following utilities that let you perform administrative tasks and capture information about configuration and performance. Figure A-1 describes each utility. The subsequent tables describe the options for the **cdr**, **dbexport**, **dbimport**, **dbload**, **dbschema**, **onaudit**, **onbar**, **oncheck**, **ondblog**, **oninit**, **onload**, **onlog**, **onmode**, **onparams**, **onpladm**, **onspaces**, **onstat**, **ontape**, and **onunload** utilities.

Figure A-1

Utility	Description	Where Described
cdr	Control Enterprise Replication operations.	*IBM Informix Dynamic Server Enterprise Replication Guide*
dbexport	Unload a database into text files for later import into another database and create a schema file.	*IBM Informix Migration Guide*
dbimport	Create and populate a database from text files. Use the schema file with dbimport to re-create the database schema.	*IBM Informix Migration Guide*
dbload	Load data into databases or tables.	*IBM Informix Migration Guide*
dbschema	Create a file that contains the SQL statements needed to replicate a specified table, view, or database, or view the information schema.	*IBM Informix Migration Guide*
imcadmin	Start or stop MaxConnect, or gather statistics on MaxConnect.	*IBM Informix MaxConnect*
ISA	Perform various administrative tasks using the IBM Informix Server Administrator (ISA).	ISA online help
ism	Manage IBM Informix Storage Manager, storage devices, and media volumes.	*IBM Informix Storage Manager Administrator's Guide*
onaudit	Manage audit masks and auditing configurations.	*Trusted Facility Guide*
onbar	Back up and restore storage spaces and logical logs.	*Backup and Restore Guide*
oncheck	Check specified disk structures for inconsistencies, repair inconsistent index structures, and display information about disk structures.	*Administrator's Reference*
ondblog	Change the logging mode.	*Administrator's Reference*
oninit	Bring the database server online.	*Administrator's Reference*
onload	Load data that was created with **onunload** into the database server.	*IBM Informix Migration Guide*

(1 of 2)

Utility	Description	Where Described
onlog	Display the contents of logical-log files.	*Administrator's Reference*
onmode	Change the database server operating mode and perform various other operations on shared memory, sessions, transactions, parameters, and segments.	*Administrator's Reference*
ON-Monitor	Perform administrative tasks using the ON-Monitor menus.	*Administrator's Reference*
onparams	Modify the configuration of logical logs or physical logs.	*Administrator's Reference*
onperf	Monitor database server performance (create graphs, query trees, show status and metrics).	*Performance Guide*
onpladm	Write scripts and create files that automate data load and unload jobs.	*IBM Informix High-Performance Loader User's Guide*
onshowaudit	Extract information from an audit trail.	*Trusted Facility Guide*
onspaces	Modify dbspaces, blobspaces, sbspaces, or extspaces.	*Administrator's Reference*
onstat	Monitor the operation of the database server.	*Administrator's Reference*
onstat -g	Monitor and debug the database server.	*Administrator's Reference* *Performance Guide*
ontape	Log, back up, and restore data.	*Backup and Restore Guide*
onunload	Unload data from the database server.	*IBM Informix Migration Guide*
Server Setup	Configure the database server, storage spaces, network connectivity, and J/Foundation.	ISA online help
Server Studio JE	Explore database servers, execute SQL statements and stored procedures (SPLs), and display the results.	Server Studio online help

(2 of 2)

The cdr Utility

cdr Option	Function
change replicate	Change an existing replicate by adding or deleting one or more participants.
change replicateset	Change a replicate set by adding or deleting replicates.
connect server	Reestablish a connection to a database server that has been disconnected with a **cdr disconnect server** command.
define replicate	Define a replicate in the global catalog.
define replicateset	Define a replicate set.
define server	Define a database server group and all its members for Enterprise Replication.
delete replicate	Delete a replicate from the global catalog.
delete replicateset	Delete a replicate set.
delete server	Delete a database server from the global catalog.
disconnect server	Stop a server connection.
error	Manage the error table and display errors.
finderr	View Enterprise Replication error messages.
list replicate	Display information about the replicates on the current server.
list replicateset	Display information about the replicate sets on the current server.
list server	Display a list of the Enterprise Replication servers that are visible to the current server.
modify replicate	Modify replicate attributes.
modify replicateset	Modify all the replicates in a replicate set.
modify server	Modify the Enterprise Replication attributes of a database server.
remove	Remove Enterprise Replication from an HDR server.

(1 of 2)

cdr Option	Function
resume replicate	Resume delivery of replication data.
resume replicateset	Resume delivery of replication data for all the replicates in a replicate set.
resume server	Resume delivery of replication data to a suspended database server.
start	Start Enterprise Replication processing.
start replicate	Start the capture and transmittal of replication transactions.
start replicateset	Start the capture and transmittal of replication transactions for all the replicates in a replicate set.
stop	Stop Enterprise Replication processing.
stop replicate	Stop the capture and transmittal of replication transactions.
stop replicateset	Stop the capture and transmittal of replication transactions for all the replicates in a replicate set.
suspend replicate	Suspend delivery of replication data.
suspend replicateset	Suspend delivery of replication data for all the replicates in a replicate set.
suspend server	Suspend delivery of replication data to a database server from specific or all database servers.

(2 of 2)

The dbexport Utility

dbexport Option	Function
-b	Specify, in kilobytes, the block size of the tape device.
-c	Complete exporting unless a fatal error occurs.
-d	Export simple-large-object descriptors only.
-f	Specify the pathname of the schema file.
-o	Specify the directory that holds the data files and schema file that **dbexport** creates.
-q	Suppress the display of error messages, warnings, and SQL data-definition statements.
-s	Specify, in kilobytes, the amount of data stored on tape.
-ss	Generate database server-specific syntax for all tables.
-t	Specify the pathname of the tape device for the data and schema files.
-V	Display product version.
-X	Recognize HEX binary data in character fields.

The dbimport Utility

dbimport Option	Function
-b	Specify, in kilobytes, the block size of the tape device.
-c	Complete importing unless a fatal error occurs.
-f	Specify the pathname of the schema file to use to create the database.
-i	Specify the directory that holds the data and schema files that **dbimport** uses to create and load the new database.

(1 of 2)

dbimport Option	Function
-q	Suppress the display of error messages, warnings, and SQL data-definition statements.
-s	Specify, in kilobytes, the amount of data stored on tape.
-ss	Generate database server-specific syntax for all tables.
-t	Specify the pathname of the tape device that holds the input files.
-V	Display product version.
-X	Recognize HEX binary data in character fields.

(2 of 2)

The dbload Utility

dbload Option	Function
-c	Specify the filename of the **dbload** command file.
-d	Specify the name of the database to receive data.
-e	Specify the number of bad rows that **dbload** reads before terminating.
-i	Specify the number of rows to ignore in the input file.
-k	Lock the tables listed in the command file during the load.
-l	Specify the filename of the error log file.
-n	Specify the commit interval in number of rows.
-p	Prompt for instructions if the number of bad rows exceeds the limit.
-r	Prevent **dbload** from locking the tables during a load.
-s	Check the syntax of the statements in the command file.
-V	Display product version.
-X	Recognize HEX binary data in character fields.

The dbschema Utility

dbschema Option	Function
all	Display all tables included in display of data distributions.
-d	Include database name.
-f	Display stored procedures.
-hd	Display histograms of the data distribution values.
-ss	Generate database server-specific syntax.
-u	Include definitions of user-defined types.
-ui	Include definitions and inheritance of user-defined types.
-V	Display product version.

The ism Utility

ism Option	Function
ism_add	Add administrative users and storage devices.
ism_catalog	Re-create entries in, or recover the ISM catalog, create a new bootstrap, or to find the ISM server bootstrap.
ism_chk.pl	Collect information about ISM, ON-Bar, and the database server.
ism_clone	Clone a storage volume or save set.
ism_config	Configure the ISM server properties and change storage-volume parameters.
ism_op	Label, mount, and unmount storage volumes.
ism_rm	Remove an administrative user or storage device from the ISM server, or remove a storage volume from the ISM catalog.
ism_show	View information about the ISM administrators, media volumes, and storage devices.

(1 of 2)

ism Option	Function
ism_shutdown	Shut down the ISM server.
ism_startup	Start the ISM server.
ism_watch	Monitor the ISM server.

(2 of 2)

The onaudit Utility

onaudit Option	Function
-a	Create audit masks.
-c	Show the audit configuration.
-d	Delete audit masks.
-e	Set the error mode.
-f	Name a file to add audit masks.
-l	Specify the audit mode.
-m	Modify audit masks.
-n	Start a new audit file.
-o	Display audit masks.
-p	Name directory for audit files.
-s	Specify maximum size of audit files.

The onbar Utility

onbar Option	Function
-b	Back up storage spaces (level-0, 1, or 2) and logical logs.
-b -l	Back up the logical logs.
-b -l -c	Back up the current log also.
-b -l -C	Start continuous logical-log backup.
-b -l -s	Salvage the logical logs.
-r	Perform a cold or warm restore.
-r rename	Rename specified chunks to new pathnames and offsets.
-r -l	Perform a logical restore.
-r -n	Perform a point-in-log restore.
-r -p	Perform a physical-only restore.
-r -t	Perform a point-in-time restore.
-RESTART	Restart a failed backup.
-v	Verify a backup.

The oncheck Utility

oncheck Option	Function
-cc, **-pc**	Check and display system catalog tables.
-cd, **-cD**	Check pages.
-ce, **-pe**	Check and display chunk free list.
-ci, **-cI**	Check index node links.
-cr, **-pr**	Check and display reserved pages.

(1 of 2)

oncheck Option	**Function**
-cR, **-pR**	Check and display reserved pages, physical-log, and logical-log pages.
-cs, **-ps**	Check and display sbspace information.
-cS, **-pS**	Check and display more detailed sbspace and metadata information.
-n	Do not repair index. (Use with the **-ci**, **-cI**, **-pk**, **-pK**, **-pl**, **-pL** options.)
-pd, **-pD**	Display rows in hexadecimal format.
-pk, **-pK**	Check and display index node links and key values.
-pl, **-pL**	Check and display index leaf-nodes and key values.
-pB	Display blobspace blobpage statistics.
-pp, **-pP**	Display the contents of a logical page.
-pt, **-pT**	Display tblspace information for a particular table or fragment.
-q	Suppress all messages.
-u	Send special arguments to an access method.
-x	Turn on locking while checking an index.
-y	Repair indexes when errors are detected.

(2 of 2)

The ondblog Utility

ondblog Option	Function
ansi	Change database logging to be ANSI compliant.
buf	Set logging mode to buffered mode.
cancel	Cancel the logging mode change request before the next level-0 backup occurs.
nolog	Set logging mode so that no transactions are logged.
unbuf	Set logging mode to unbuffered mode.

The oninit Utility

oninit Option	Function
none	Initialize shared memory; go to online mode.
-i	Initialize disk and shared memory; go to online mode.
-is	Initialize disk and shared memory; go to quiescent mode.
-p	Do not reclaim temporary tables.
-r	Start the database server in recovery mode. Switch the mode for an HDR pair.
-s	Initialize shared memory; go to quiescent mode.

The onload Utility

onload Option	Function
database	Specify the database name.
owner.	Specify the table owner.
table	Specify the table name.
-b	Specify, in kilobytes, the block size of the tape device.
-l	Read the values for tape device, block size, and tape size from LTAPEDEV, LTAPEBLK, and LTAPESIZE.
-s	Specify, in kilobytes, the amount of data to store on tape.
-t	Specify the pathname of the file on disk or tape device where the input tape is mounted.

The onlog Utility

onlog Option	Function
-b	Display logical-log records for blobspace blobpages.
-l	Display long listing of logical-log records.
-n	Display log records specified by *uniqid*.
-t	Display logical-log records for tablespace.
-u	Display logical-log records for user.
-x	Display logical-log records for transaction.

The onmode Utility

onmode Option	Function
-a	Add a shared-memory segment.
-b	Revert database server disk structures.
-c	Force a full checkpoint.
-c block	Block the database server from transactions.
-c fuzzy	Force a fuzzy checkpoint.
-c unblock	Unblock the database server so transactions can resume.
-d	Set data-replication types for HDR.
-e ENABLE	Enable the SQL statement cache.
-e FLUSH	Flush the SQL statement cache.
-e OFF	Turn off the SQL statement cache.
-e ON	Turn on the SQL statement cache.
-k	Go to offline mode.
-l	Switch to the next logical-log file.
-m	Go from quiescent to online mode.
-n	End forced residency.
-p	Add or remove virtual processors.
-r	Start forced residency.
-s	Go to quiescent gracefully.
-u	Go to quiescent immediately.
-z	Kill a database server session.
-BC	Enable large chunk mode.
-C	Control the B-tree scanner.

(1 of 2)

onmode Option	Function
-D	Change MAX_PDQPRIORITY.
-F	Free unused memory segments.
-M	Change DS_TOTAL_MEMORY.
-O	Override ONDBSPACEDOWN.
-Q	Change DS_MAX_QUERIES.
-R	Regenerate **.infos.*dbservername*** file.
-S	Change DS_MAX_SCANS.
-W STMT_CACHE_HITS	Specify the number of hits before a statement is fully inserted into the SQL statement cache.
-W STMT_CACHE_NOLIMIT	Control insertion of statements into the SQL statement cache when it has reached the size limit.
-W STMT_CACHE_SIZE	Specify the size of the SQL statement cache.
-Y	Dynamically set the SET EXPLAIN statement.
-Z	Kill a distributed transaction.

(2 of 2)

ON-Monitor Utility

Menu Options	Function
Profile	Display database server performance statistics.
Userthreads	Display the status of active user threads.
Spaces	Display status information about database server storage spaces and chunks.
Databases	Display the name, owner, and logging mode of the first 100 databases.
Logs	Display status information about the physical-log buffer, the physical log, the logical-log buffer, and the logical-log files.
Archive	Display a list of all backup tapes and logical-log files that you require to restore data using **ontape**.
data-Replication	Display High-Availability Data-Replication status and configuration.
Output	Store the output of other status information in a specified file.
Configuration	Copy the current database server configuration to a file.

The onparams Utility

onparams Option	Function
-a	Add logical-log file to the end of the list.
-i	Use with the **-a** option to add a logical-log file after the current log.
-d	Drop logical-log file.
-p	Change physical-log size or location.

The onpladm Utility

onpladm Option	Function
list project	List the names of the projects in the **onpload** database.
create project	Create a new empty project.
delete project	Delete a project.
run project	Run all load or unload jobs in a project.
list job	List all the jobs in the **onpload** database.
create job, delete job, run job	Create, delete, or run a load or unload job.
list map	List all the maps in the **onpload** database.
create map, delete map, modify object	Create, delete, or modify a load or unload map.
list format	List all the formats in the **onpload** database.
create object, delete format, modify object	Create, delete, or modify a format.
list filter	List all the filters in the **onpload** database.
create object, delete filter, modify object	Create, delete, or modify a filter.
list query	List all the query objects in the **onpload** database.
create object, delete query, modify object	Create, delete, or modify a query object.
list device	List all the device arrays in the **onpload** database.
create object, delete device, modify object	Create, delete, or modify a device array.

(1 of 2)

onpladm Option	Function
list machine	List all the machine types in the **onpload** database.
create object, delete machine, modify object	Create, delete, or modify a machine type object.
list defaults	List the default attribute values for a database server.
configure defaults	Configure default attribute values for a database server.

(2 of 2)

The onspaces Utility

onspaces Option	Function
-a	Add a chunk to a storage space.
-c	Create a storage space. Suboptions: **-b** blobspace **-d** dbspace **-S** sbspace **-t** temporary dbspace or sbspace **-x** extspace
-ch	Change sbspace attributes.
-cl	Clean up stray references to smart large object.
-d	Drop a storage space or a chunk.
-f	Specify DATASKIP parameter.
-m	Start mirroring.
-r	End mirroring.
-s	Change status of a mirrored chunk.

The onstat Utility

onstat Option	Prints
-	Output header
--	Listing of all **onstat** options
-a	Summary of user-oriented options
-b	Buffers in use
-B	All buffers

(1 of 3)

onstat Option	Prints
-c	Configuration file information
-C	B-tree cleaner requests
-d	Storage space chunks, general information
-D	Storage space chunks, page reads and writes
-f	DATASKIP information
-F	Write type statistics
-g	Monitoring information
-G	Global transaction IDs
-h	Buffer hash chain information
-i	Interactive mode
-j	Prints the interactive status of the active **onpload** process
-k	Locks held
-K	Byte-range locks
-l	Logging information
-m	Database server message log
-o	Shared-memory segment
-O	Optical Subsystem memory cache and staging-area blobspaces
-p	Database server profile of activity
-P	Partition number and buffer pool pages that belong to the partition
-r	Repetition of the **onstat** command
-R	LRU queues
-s	Latches
-t	Tablespaces, active
-T	Tablespaces, all

(2 of 3)

onstat Option	Prints
-u	User threads and transactions
-x	Transaction information
-X	Buffers (includes addresses of waiting threads)
-z	Resetting of statistic counts

(3 of 3)

The onstat -g Utility

onstat -g Option	Prints
act	All active threads
afr pool	Allocated memory fragments for a pool
all	All multithreading information
ath	All threads
cac agg	User-defined aggregates that are in the cache
cac stmt	Contents of the SQL statement cache
cat	Enterprise Replication servers, replicates, and replicate sets
con	Conditions with waiters
ddr	Status of the Enterprise Replication database log reader
dic	Tables cached in the shared-memory dictionary
dis	Status of database servers
dll	Loaded dynamic libraries
dri	Data-replication information for HDR
dsc	Data-distribution cache information
dss	Data sync threads used in Enterprise Replication
dtc	Statistics about the delete table cleaner used in Enterprise Replication
env	Prints the values of environment variables the database server currently uses.
ffr	Free fragments for a pool of shared memory
glo	Global multithreading information
grp	Statistics about the Enterprise Replication grouper
imc	MaxConnect instances

(1 of 4)

onstat -g Option	Prints
ioa	Combined information from **-g ioq**, **-g iov**, and **-g iob**
iob	Big-buffer use by I/O virtual processor class
iof	Asynchronous I/O (AIO) statistics by chunk or file
iog	AIO global information
ioq	I/O queues statistics
iov	Asynchronous I/O statistics by virtual processor
lmx	All locked mutexes
lsc	Light scans
mem	Statistics for a memory pool
mgm	Memory Grant Manager resource information
nbm	Block bitmap for the nonresident segments
nif	Network interface statistics for Enterprise Replication
nsc	Shared memory status by client
nsd	Network shared-memory data for poll threads
nss	Network shared-memory status by session
nta	Combined network statistics from **-g ntd**, **-g ntm**, **-g ntt**, and **-g ntu**
ntd	Network statistics by server
ntm	Network mail statistics
ntt	Network user times
ntu	Network user statistics
pos	**.infos.*dbservername*** file
ppf	Partition profile for partition
prc	Stored-procedure cache information

(2 of 4)

onstat -g Option	Prints
qst	Queue statistics
que	Statistics for the high-level queue interface in Enterprise Replication
rbm	Block bitmap for the resident segment
rcv	Statistics about the receive manager in Enterprise Replication
rea	Ready threads
rep	Queued events for the Enterprise Replication schedule manager
rqm	Low-level queues for Enterprise Replication
sch	Semaphore operations, spins, and busy waits for each virtual processor
seg	Shared-memory-segment statistics
ses	Session information by session ID
sle	Sleeping threads
smb	Sbspace usage
spi	Statistics on spin locks
sql	SQL information by session ID
ssc	Contents of the SQL statement cache (same as **-g cac stmt**)
ssc all	*Key-only* cache entries and the fully cached statements
ssc pool	Memory pool usage for the SQL statement cache
stk	Dump of stack of threads specified by thread ID
stm	Memory that prepared SQL statements use
sts	Maximum and current stack use per thread
tpf	Thread profile for TID
ufr	Allocated fragments by use

(3 of 4)

onstat -g Option	Prints
wai	Waiting threads
wmx	Mutexes with waiters
wst	Wait statistics
xmf	Communication between coservers

(4 of 4)

The ontape Utility

ontape Option	Function
-a	Back up the logical logs.
-c	Start continuous logical-log backup.
-r	Perform a cold or warm restore.
-r rename	Rename specified chunks to new pathnames and offsets.
-p	Perform a physical restore to set up HDR.
-s	Back up storage spaces (level-0, 1, or 2).

The onunload Utility

onunload Option	Function
database	Specify the name of a database.
owner.	Specify the table owner.
table	Specify the table name.
-b	Specify, in kilobytes, the block size of the tape device.
-l	Read the values for tape device, block size, and tape size from LTAPEDEV, LTAPEBLK, and LTAPESIZE.
-s	Specify, in kilobytes, the amount of data to store on tape.
-t	Specify the pathname of the file on disk or tape device where the input tape is mounted.

Tip: *All Informix command-line utilities support the **-V** option that displays the software version and serial numbers.*

Notices

Trademarks

Index