Getting Started

with Informix Dynamic Server.2000

Version 9.2
September 1999
Part No. 000-6207
Published by Informix® Press
Informix Corporation
4100 Bohannon Drive
Menlo Park, CA 94025-1032

© 1999 Informix Corporation. All rights reserved. The following are trademarks of Informix Corporation or its affiliates, one or more of which may be registered in the United States or other jurisdictions:

Answers OnLine™; C-ISAM®; Client SDK™; DataBlade®; Data Director™; Decision Frontier™; Dynamic Scalable Architecture™; Dynamic Server™; Dynamic Server™; Dynamic Server™; Dynamic Server™; Dynamic Server™; Dynamic Server™; Dynamic Server™ with Advanced Decision Support Option™; Dynamic Server™ with Extended Parallel Option™; Dynamic Server™ with MetaCube®; Dynamic Server™ with Universal Data Option™; Dynamic Server™ with Web Integration Option™; Dynamic Server™, Developer Edition™; Dynamic Server™, Workgroup Edition™; Dynamic Virtual Machine™; Enterprise Decision Server™; Formation™; Formation Architect™; Formation Flow Engine™; Gold Mine Data Access®, IIF.2000™; i.Reach™; i.Sell™; Illustra®; Informix®; Informix® 4GL; Informix® Inquire®; Informix® Internet Foundation.2000™; InformixLink®; Informix® Red Brick® Decision Server™; Informix Session Proxy™; Informix® Vista™; InfoShelf™; Interforum™; i-Spy™; Mediazation™; MetaCube®; NewEra™; On-Line Dynamic Server™; OnLine/Secure Dynamic Server™; OpenCase®; Orca®; PaVER®; Red Brick® and Design; Red Brick® Data Mine™; Red Brick® Mine Builder™; Red Brick® DecisionScape™; Red Brick® Ready™; Red Brick Systems®; Regency Support®; Rely on Red Brick®; RISQL®; Solution Design™; STARIndex™; STARJoin™; SuperView®; TARGETIndex™; TARGETJoin™; The Data Warehouse Company®; The one with the smartest data wins.™; The world is being digitized. We’re indexing it.™; Universal Data Warehouse Blueprint™; Universal Database Components™; Universal Web Connect™; ViewPoint®; Visionary™; Web Integration Suite™. The Informix logo is registered with the United States Patent and Trademark Office. The DataBlade logo is registered with the United States Patent and Trademark Office.

Documentation Team: Linda Briscoe, Erin Cizina, Evelyn Eldridge, Signe Haugen, Judith Sherwood

GOVERNMENT LICENSE RIGHTS

Software and documentation acquired by or for the US Government are provided with rights as follows:
(1) if for civilian agency use, with rights as restricted by vendor’s standard license, as prescribed in FAR 12.212;
(2) if for Dept. of Defense use, with rights as restricted by vendor’s standard license, unless superseded by a negotiated vendor license, as prescribed in DFARS 227.7202. Any whole or partial reproduction of software or documentation marked with this legend must reproduce this legend.
## Table of Contents

### Introduction
- In This Introduction .................................................. 3
- About This Manual ...................................................... 3
- Types of Users .............................................................. 3
- Software Dependencies .................................................. 4
- Assumptions About Your Locale ....................................... 4
- Demonstration Databases ............................................... 5
- New Features ............................................................... 5
- Documentation Conventions .......................................... 5
  - Typographical Conventions ......................................... 6
  - Icon Conventions .................................................... 7
- Additional Documentation ............................................. 8
  - On-Line Manuals .................................................... 9
  - Printed Manuals .................................................... 9
  - On-Line Help .......................................................... 9
  - Error Message Documentation ..................................... 9
- Documentation Notes, Release Notes, Machine Notes .......... 10
- Related Reading .......................................................... 11
- Compliance with Industry Standards ................................. 13
- Informix Welcomes Your Comments ................................ 13

### Chapter 1 Informix Dynamic Server Architecture
- In This Chapter .......................................................... 1-3
- Overview of Database Management Systems ....................... 1-3
  - Relational Database Management Systems ....................... 1-5
  - Object-Relational Database Management Systems ............... 1-5
- What Is Informix Dynamic Server? ................................ 1-6
Dynamic Scalable Architecture ............................. 1-6
The Shared-Memory Component ............................. 1-7
The Disk Component ........................................... 1-8
The Virtual Processor Component ............................. 1-9
Architectural Elements of Dynamic Server ............... 1-10
Client/Server Architecture ..................................... 1-10
Scalability ...................................................... 1-13
High Performance .............................................. 1-14
Fault Tolerance and High Availability ....................... 1-17
Multimedia Support ............................................. 1-21
Dynamic System Administration ............................. 1-22
Distributed Databases and Transactions ..................... 1-22
Database Server Security ....................................... 1-24
Graphical Administration Tools ............................. 1-25

Chapter 2 Informix Dynamic Server Features
In This Chapter ..................................................... 2-3
RDBMS Features .................................................. 2-3
Structured Query Language (SQL) ............................ 2-3
Stored Procedure Language (SPL) ............................ 2-4
System Catalog .................................................... 2-5
Data Types ....................................................... 2-6
Data-Management Features ..................................... 2-7
Data Replication .................................................... 2-7
Cost-Based Optimizer .............................................. 2-9
High-Performance Loader ....................................... 2-9
Database Server Utilities ....................................... 2-10
Data-Recovery and Data-Restore Features ..................... 2-12
The ontape Utility .................................................. 2-13
On-Line Archive Recovery System (ON-Archive) ............ 2-13
On-Line Backup and Restore (ON-Bar) ....................... 2-14
Informix Storage Manager ...................................... 2-15
Tools That You Cannot Use for Recovery ....................... 2-15
Database Support .................................................. 2-16
ANSI-Compliant Databases ..................................... 2-16
Distributed Databases .......................................... 2-17
Databases Distributed Across Multiple Vendor Servers ........ 2-18
Dimensional Databases ......................................... 2-19
Object-Relational Databases ................................... 2-20
## Table of Contents

- Access Methods ........................................... 2-21  
  - Primary Access Methods ................................. 2-21  
  - Secondary Access Methods .............................. 2-21  
  - Writing Access Methods ................................. 2-22  
- Application Types ......................................... 2-24  
  - OLTP Applications ...................................... 2-24  
  - DSS Applications ....................................... 2-25  
- Global Language Support ................................. 2-25  
- Other Database Server Features .......................... 2-26  
  - Support for Long Identifiers and User Names ........ 2-26  
  - Shared-Statement Caching .............................. 2-26  
  - Locking .................................................. 2-27  
  - Isolation Level ......................................... 2-28  
  - Transaction Logging .................................... 2-28  
  - Rolling Back Statements in a Transaction ............ 2-28  
  - Role Separation ........................................ 2-29  
  - The Information Schema ................................. 2-29  
- Extending Dynamic Server .................................. 2-30  
  - Extended Data Types .................................... 2-30  
  - Simple and Smart Large Objects ....................... 2-32  
  - User-Defined Routines and DataBlade Modules ........ 2-33  
  - User-Defined Virtual Processors ...................... 2-36  
  - User-Defined Access Methods ......................... 2-36  

### Chapter 3  Getting Up and Running with the Database Server

- In This Chapter ........................................... 3-3  
- Basic Tasks for Getting Started ......................... 3-3  
- Tools and Client Products ............................... 3-4  
  - Tools .................................................... 3-5  
  - Client SDK Products ................................... 3-6  
  - Compatibility of Client and Server Programs ........ 3-10  
- Installing Informix Products .............................. 3-10  
- Configuring the Database Server Environment .......... 3-11  
  - Required Environment Variables ....................... 3-12  
  - Connectivity Information ............................. 3-14  
  - Preparing the ONCONFIG Configuration File ........... 3-16  
  - Allocating Disk Space .................................. 3-17  
  - Initializing Disk Space ................................ 3-18

Table of Contents  v
## Introduction

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>In This Introduction</td>
<td>3</td>
</tr>
<tr>
<td>About This Manual</td>
<td>3</td>
</tr>
<tr>
<td>Types of Users</td>
<td>3</td>
</tr>
<tr>
<td>Software Dependencies</td>
<td>4</td>
</tr>
<tr>
<td>Assumptions About Your Locale</td>
<td>4</td>
</tr>
<tr>
<td>Demonstration Databases</td>
<td>5</td>
</tr>
<tr>
<td>New Features</td>
<td>5</td>
</tr>
<tr>
<td>Documentation Conventions</td>
<td>5</td>
</tr>
<tr>
<td>Typographical Conventions</td>
<td>6</td>
</tr>
<tr>
<td>Icon Conventions</td>
<td>7</td>
</tr>
<tr>
<td>Comment Icons</td>
<td>7</td>
</tr>
<tr>
<td>Cross-Reference Icons</td>
<td>7</td>
</tr>
<tr>
<td>Feature, Product, and Platform Icons</td>
<td>8</td>
</tr>
<tr>
<td>Additional Documentation</td>
<td>8</td>
</tr>
<tr>
<td>On-Line Manuals</td>
<td>9</td>
</tr>
<tr>
<td>Printed Manuals</td>
<td>9</td>
</tr>
<tr>
<td>On-Line Help</td>
<td>9</td>
</tr>
<tr>
<td>Error Message Documentation</td>
<td>9</td>
</tr>
<tr>
<td>Documentation Notes, Release Notes, Machine Notes</td>
<td>10</td>
</tr>
<tr>
<td>Related Reading</td>
<td>11</td>
</tr>
<tr>
<td>UNIX Manuals</td>
<td>12</td>
</tr>
<tr>
<td>Windows NT Manuals</td>
<td>12</td>
</tr>
<tr>
<td>Compliance with Industry Standards</td>
<td>13</td>
</tr>
<tr>
<td>Informix Welcomes Your Comments</td>
<td>13</td>
</tr>
</tbody>
</table>
In This Introduction

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

About This Manual

This manual provides an overview of how to work efficiently with Informix Dynamic Server 2000. It outlines Dynamic Server architecture, introduces the major features, and discusses associated Informix client and application programming interface (API) products and manuals. It also summarizes the basic tasks that are required for you to get up and running with the database server and contains information to help you use the documentation for Dynamic Server, Version 9.2.

Types of Users

This manual is written for the following users:

- Database server administrators
- Database server operators
- Database administrators
- Programmers in the following categories
  - Application developers
  - DataBlade-module developers
  - Authors of user-defined routines
- Database users
This manual assumes 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 are unfamiliar with relational database concepts or SQL, review the *Informix Guide to SQL: Tutorial* and the *Informix Guide to Database Design and Implementation*. For additional titles, see Chapter 4, "Using the Documentation."

**Software Dependencies**

This manual assumes that you are using Informix Dynamic Server 2000, Version 9.2, as your database server.

**Assumptions About Your Locale**

Informix products can support many languages, cultures, and code sets. All culture-specific information is brought together in a single environment, called a Global Language Support (GLS) locale.

This manual assumes that you use the U.S. 8859-1 English locale as the default locale. The default is **en_us.8859-1** (ISO 8859-1) on UNIX platforms or **en_us.1252** (Microsoft 1252) for Windows NT environments. This locale supports U.S. English format conventions for dates, times, and currency, and also supports the ISO 8859-1 or Microsoft 1252 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 information on how to specify a nondefault locale, additional syntax, and other considerations related to GLS locales, see the *Informix Guide to GLS Functionality*. 
Demonstration Databases

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

- The stores_demo database illustrates a relational schema with information about a fictitious wholesale sporting-goods distributor. Many examples in 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 DB-Access User’s Manual. For descriptions of the databases and their contents, see the Informix Guide to SQL: Reference.

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

New Features

For a comprehensive list of new database server features, see the release notes.

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.

The following conventions are discussed:

- Typographical conventions
- Icon conventions
### Typographical Conventions

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

<table>
<thead>
<tr>
<th>Convention</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>KEYWORD</td>
<td>All primary elements in a programming language statement (keywords) appear in uppercase letters in a serif font.</td>
</tr>
<tr>
<td><strong>italics</strong></td>
<td>Within text, new terms and emphasized words appear in italics.</td>
</tr>
<tr>
<td><strong>italics</strong></td>
<td>Within syntax and code examples, variable values that you are to specify appear in italics.</td>
</tr>
<tr>
<td><strong>boldface</strong></td>
<td>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.</td>
</tr>
<tr>
<td><strong>monospace</strong></td>
<td>Information that the product displays and information that you enter appear in a monospace typeface.</td>
</tr>
<tr>
<td>KEYSORKE</td>
<td>Keys that you are to press appear in uppercase letters in a sans serif font.</td>
</tr>
<tr>
<td>♦</td>
<td>This symbol indicates the end of one or more product- or platform-specific paragraphs.</td>
</tr>
<tr>
<td>→</td>
<td>This symbol indicates a menu item. For example, “Choose Tools→Options” means choose the Options item from the Tools menu.</td>
</tr>
</tbody>
</table>

**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.

<table>
<thead>
<tr>
<th>Icon</th>
<th>Label</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="icon.png" alt="Warning" /></td>
<td><strong>Warning:</strong></td>
<td>Identifies paragraphs that contain vital instructions, cautions, or critical information</td>
</tr>
<tr>
<td><img src="icon.png" alt="Important" /></td>
<td><strong>Important:</strong></td>
<td>Identifies paragraphs that contain significant information about the feature or operation that is being described</td>
</tr>
<tr>
<td><img src="icon.png" alt="Tip" /></td>
<td><strong>Tip:</strong></td>
<td>Identifies paragraphs that offer additional details or shortcuts for the functionality that is being described</td>
</tr>
</tbody>
</table>

Cross-Reference Icons

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

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="icon.png" alt="Cross-Reference" /></td>
<td>Identifies paragraphs that contain cross-references to other Informix manuals that provide additional information on a topic</td>
</tr>
</tbody>
</table>
Feature, Product, and Platform Icons

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

<table>
<thead>
<tr>
<th>Icon</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>GLS</td>
<td>Identifies information that relates to the Informix Global Language Support (GLS) feature</td>
</tr>
<tr>
<td>UNIX</td>
<td>Identifies information that is specific to UNIX platforms</td>
</tr>
<tr>
<td>WIN NT</td>
<td>Identifies information that is specific to the Windows NT environment</td>
</tr>
</tbody>
</table>

These icons can apply to an entire section or to one or more paragraphs within a section. If an icon appears next to a section heading, the information that applies to the indicated feature, product, or platform ends at the next heading at the same or higher level. A ♦ symbol indicates the end of feature-, product-, or platform-specific information that appears within one or more paragraphs within a section.

Additional Documentation

For additional information, you might want to refer to the following types of documentation:

- On-line manuals
- Printed manuals
- On-line help
- Error message documentation
- Documentation notes, release notes, and machine notes
- Related reading

For an overview of the manuals in the Dynamic Server documentation set, see Chapter 4, “Using the Documentation.”
On-Line Manuals

An Answers OnLine CD that contains Informix manuals in electronic format is provided with your Informix products. You can install the documentation or access it directly from the CD. For information about how to install, read, and print on-line manuals, see the installation insert that accompanies Answers OnLine.

Informix on-line manuals are also available on the following Web site:

www.informix.com/answers

Printed Manuals

To order printed manuals, call 1-800-331-1763 or send email to moreinfo@informix.com. Please provide the following information when you place your order:

- The documentation that you need
- The quantity that you need
- Your name, address, and telephone number

On-Line Help

Informix provides on-line help with each graphical user interface (GUI) that displays information about those interfaces and the functions that they perform. Use the help facilities that each GUI provides to display the on-line help.

Error Message Documentation

Informix software products provide ASCII files that contain all of the Informix error messages and their corrective actions.
To read error messages and corrective actions on UNIX, use one of the following utilities.

<table>
<thead>
<tr>
<th>Utility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>finderr</td>
<td>Displays error messages on line</td>
</tr>
<tr>
<td>rofferr</td>
<td>Formats error messages for printing</td>
</tr>
</tbody>
</table>

To read error messages and corrective actions in Windows environments, use the *Informix Find Error* utility. To display this utility, choose *Start* → *Programs* → *Informix* from the Task Bar.

Instructions for using the preceding utilities are available in Answers OnLine. Answers OnLine also provides a listing of error messages and corrective actions in HTML format.

**Documentation Notes, Release Notes, Machine Notes**

In addition to printed documentation, the following sections describe the online files that supplement the information in this manual. Please examine these files before you begin using your database server. They contain vital information about application and performance issues.

On UNIX platforms, the following on-line files appear in the `$INFORMIXDIR/release/en_us/0333` directory.

<table>
<thead>
<tr>
<th>On-Line File</th>
<th>Purpose</th>
</tr>
</thead>
<tbody>
<tr>
<td>STARTDOC_9.2</td>
<td>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.</td>
</tr>
<tr>
<td>SERVERS_9.2</td>
<td>The release notes file describes feature differences from earlier versions of Informix products and how these differences might affect current products. This file also contains information about any known problems and their workarounds.</td>
</tr>
<tr>
<td>IDS_9.2</td>
<td>The machine notes file describes any special actions that you must take to configure and use Informix products on your computer. Machine notes are named for the product described.</td>
</tr>
</tbody>
</table>

♦
The following items appear in the Informix folder. To display this folder, choose Start→Programs→Informix from the Task Bar.

### Program Group Item

<table>
<thead>
<tr>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Documentation Notes</strong></td>
</tr>
<tr>
<td><strong>Release Notes</strong></td>
</tr>
</tbody>
</table>

The machine notes do not apply to Windows environments.

### Related Reading

The following publications provide additional information about the topics that this manual discusses.

To learn more about Structured Query Language, consider the following books:


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

- *An Introduction to Database Systems* by C. J. Date (Addison-Wesley Publishing, 1995)
Related Reading

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


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

- *The Data Warehouse Toolkit* by Ralph Kimball (John Wiley & Sons, Inc., 1998)

**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 before you read this manual.

The following texts provide a good introduction to UNIX systems:


**Windows NT Manuals**

This manual assumes that you are familiar with your computer operating system. If you have limited Windows NT experience, consult your operating-system manual or a good introductory text before you read this manual.
Compliance with Industry Standards

The following texts provide an introduction to Windows NT:

- *Using Windows NT Workstation 3.51* by Paul Sanna (Que, 1996)
- *NT Server Management and Control* by Kenneth L. Spencer (Prentice-Hall, 1995)
- *Windows NT Network Programming* by Ralph Davis (Addison-Wesley, 1996)

Compliance with Industry Standards

The American National Standards Institute (ANSI) has established a set of industry standards for SQL. 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.

Informix Welcomes Your Comments

Let us know what you like or dislike about our manuals. To help us with future versions of our manuals, we want to know about any corrections or clarifications that you would find useful. Include the following information:

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

Send electronic mail to us at the following address:

    doc@informix.com

The `doc` alias is reserved exclusively for reporting errors and omissions in our documentation.

We appreciate your suggestions.
Informix Dynamic Server Architecture

In This Chapter ........................................... 1-3
Overview of Database Management Systems ........ 1-3
  Relational Database Management Systems .......... 1-5
  Object-Relational Database Management Systems ...... 1-5
What Is Informix Dynamic Server? ..................... 1-6
Dynamic Scalable Architecture ......................... 1-6
  The Shared-Memory Component ........................ 1-7
  The Disk Component ..................................... 1-8
  The Virtual Processor Component ...................... 1-9
Architectural Elements of Dynamic Server .............. 1-10
  Client/Server Architecture ............................. 1-10
    Supported Connection Types ......................... 1-11
    Supported Network Connections ..................... 1-12
    Supported Client/Server Configurations ............. 1-13
  Scalability .............................................. 1-13
  High Performance .................................... 1-14
    Raw (Unbuffered) Disk Management .................. 1-14
    Memory Management .................................. 1-15
    Dynamic Thread Allocation ........................... 1-15
    Fragmentation ......................................... 1-16
    Parallelization ...................................... 1-17
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fault Tolerance and High Availability</td>
<td>1-17</td>
</tr>
<tr>
<td>Storage Space Backups</td>
<td>1-18</td>
</tr>
<tr>
<td>Logical-Log Backups</td>
<td>1-18</td>
</tr>
<tr>
<td>Fast Recovery</td>
<td>1-18</td>
</tr>
<tr>
<td>Mirroring</td>
<td>1-19</td>
</tr>
<tr>
<td>High-Availability Data Replication</td>
<td>1-20</td>
</tr>
<tr>
<td>Point-in-Time Recovery</td>
<td>1-21</td>
</tr>
<tr>
<td>Multimedia Support</td>
<td>1-21</td>
</tr>
<tr>
<td>Dynamic System Administration</td>
<td>1-22</td>
</tr>
<tr>
<td>Distributed Databases and Transactions</td>
<td>1-22</td>
</tr>
<tr>
<td>Two-Phase Commit</td>
<td>1-23</td>
</tr>
<tr>
<td>Heterogeneous Commit</td>
<td>1-23</td>
</tr>
<tr>
<td>Database Server Security</td>
<td>1-24</td>
</tr>
<tr>
<td>Graphical Administration Tools</td>
<td>1-25</td>
</tr>
</tbody>
</table>
In This Chapter

This chapter introduces relational database management systems and Informix Dynamic Server 2000 architecture. For information about specific features, see Chapter 2, “Informix Dynamic Server Features.” For information about supported Informix products and client applications, see “Tools and Client Products” on page 3-4.

For complete details of Dynamic Server architecture, see the Administrator’s Guide and Administrator’s Reference.

Overview of Database Management Systems

Four fundamentally different technologies are available to manage data:

- File systems
- Relational database management systems (RDBMS)
- Object-oriented database management system (OODBMS)
- Object-relational database management systems (ORDBMS)

Each system has its own strengths that make it appropriate for particular classes of data-management problems.
Overview of Database Management Systems

Figure 1-1 illustrates the four types of database management systems.

Although file systems, video servers, and object-oriented database management systems provide solutions to business applications that do not require query capabilities, this manual concentrates on relational database management systems and object-relational database management systems.

An RDBMS or ORDBMS includes all the software you need to create and maintain a relational database, which consists of tables that are made up of rows and columns. An Informix RDBMS includes the following components:

- A database server
- A database
- One or more client applications

For information about relational and object-relational database design and management, see the Informix Guide to Database Design and Implementation and the Informix Guide to SQL: Tutorial. For information about supported data types, see the Informix Guide to SQL: Reference. To work with the sample relational and object-relational demonstration databases, see the DB-Access User’s Manual.
Relational Database Management Systems

Relational database management systems focus 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

Some relational database management systems provide limited support for complex data, which is stored in simple large objects (TEXT and BYTE data types). However, simple large objects cannot be indexed, searched, or manipulated within the database server.

Object-Relational Database Management Systems

Object-relational database management systems (ORDBMS) combine relational and object-oriented capabilities. They provide advanced support for non alphanumeric data (objects), which is stored in smart large objects (BLOB and CLOB data types).

An ORDBMS can manage complex data as objects, which provides very high-performance query and transaction capability. In addition to providing support for alphanumeric data such as character strings, integers, decimal, and date, an ORDBMS lets you create user-defined data types (opaque and distinct types) and complex data types (composite of existing data types) that provide greater flexibility in the types of data that the database server can store and manipulate.

Dynamic Server starts with the building blocks of a RDBMS and extends relational database technology to support data types that can handle 2D and 3D images, sound, video, electronic documents, HTML pages, time-series data, and spatial data.
What Is Informix Dynamic Server?

Dynamic Server is a database server. A database server is a software package that manages access to one or more databases for one or more client applications.

Specifically, Dynamic Server is a multithreaded object-relational database server that manages data that is stored in rows and columns. It employs a single processor or symmetric multiprocessing (SMP) systems and dynamic scalable architecture to deliver database scalability, manageability and performance.

Dynamic Server can be used for on-line transaction processing (OLTP), packaged applications, modest data-mart and data-warehousing applications, and web solutions. In addition, you can use the object-relational features of the database server, which include user-defined data types, user-defined routines, type and table inheritance, access methods, and virtual processors, to extend the data-management capabilities of the database server.

Dynamic Scalable Architecture

Informix database servers implement an advanced architecture that Informix calls dynamic scalable architecture (DSA). DSA provides distinct performance advantages for both single-processor and symmetric multiprocessor computers. For example:

- A small number of database server processes can service a large number of client application processes, with the following benefits:
  - Reduced operating-system overhead (fewer processes to run)
  - Reduced overall memory requirements
  - Reduced contention for resources within the DBMS
- DSA provides more control over setting priorities and scheduling database tasks than the operating system does.
Informix database server architecture consists of the following main components:

- Shared memory
- Disk
- Virtual processor

For more information on the terms and concepts discussed in this section, see the *Administrator’s Guide* and the *Administrator’s 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
- To provide a fast communications channel for local client applications that use IPC communication

Shared memory lets the database server reduce overall memory uses because the participating processes—in this case, virtual processors—do not need to maintain individual copies of the data that is in shared memory.

Shared memory reduces disk I/O because buffers, which are managed as a common pool, are flushed on a database server-wide basis instead of on a per-process basis. Furthermore, a virtual processor can often avoid reading data from disk because the data is already in shared memory as a result of an earlier read operation. The reduction in disk I/O reduces execution time.

Shared memory provides the fastest method of interprocess communication because processes read and write messages at the speed of memory transfers.
The database server uses two types of shared memory:

- Resident
  Caches data from the disk for faster access
- Virtual
  Maintains and controls the resources required by the virtual processors

### The Disk Component

A disk is a collection of one or more units of disk space assigned to the database server. All the data in the databases and all the system information that is necessary to maintain the database server resides within the disk component.

The database server uses the following types of disk access to store data:

- Buffered
- Unbuffered

The database server uses the following physical units to manage disk space:

- Chunk
- Page
- Blobpage
- Extent

Overlying the physical units of storage space, the database server supports the following logical units that are associated with database management:

- Dbspace
- Blobspace
- Extspace
- Database
- Table
- Tblspace
The database server maintains the following additional disk-space storage structures to ensure physical and logical data consistency:

- Logical log
- Physical log
- Reserved pages

The UNIX operating system provides two distinct 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** is unformatted space where the database server manages the physical organization of the data. Raw files are more reliable than cooked files because the database server guarantees that committed data is stored on disk.
- **Cooked disk space** refers to regular operating-system files. It is space that has already been organized and that the UNIX operating system manages. Cooked files are easier to allocate than raw disk space.

---

The **Virtual Processor Component**

The central component of Informix DSA is the *virtual processor*, which is a database server process that the operating system schedules for execution on the central processing unit (CPU). Database server processes are called virtual processors because they function similarly to 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 client applications.

A thread is a task for a virtual processor in the same way that the virtual processor is a task for the CPU. How the database server processes a thread depends on the operating system. Virtual processors are *multithreaded processes* because they run multiple concurrent threads.
Architectural Elements of Dynamic Server

This section describes the following architectural elements of Dynamic Server:

- Client/server architecture (page 1-10)
- Scalability (page 1-12)
- High performance (page 1-13)
- Fault tolerance and high availability (page 1-17)
- Multimedia support (page 1-21)
- Dynamic system administration (page 1-22)
- Distributed databases and transactions (page 1-22)
- Database server security (page 1-24)
- Graphical administration tools (page 1-24)

Client/Server Architecture

Informix client applications and database servers conform to a model of software design called client/server. Client/server functionality handles all the connections between the client application and the database server. You can put a client on one computer and the database server on another or the same computer. A network protocol governs how data is transferred between database servers and between a client and a database server.

A client is an application program that a user runs to request or modify information from a database by issuing Structured Query Language (SQL) statements. For example, the following Informix tools are client programs:

- DB-Access
- Enterprise Replication
- High Performance Loader (HPL)
Informix clients are briefly described in “Tools and Client Products” on page 3-4.

You can also write your own client programs with Informix ESQL/C, C++, Java, and so on, using an application-programming interface (API) that is included with products such as INFORMIX-4GL and the Informix Client Software Developer’s Kit. For details, see your API documentation.

Dynamic Server accepts requests for data from client applications, accesses the requested information from its databases, and returns the results to the client. Database-access activities include coordination of concurrent requests from multiple clients, read and write operations, and enforcement of physical and logical data consistency.

When a client connects to the database server, the client transparently accesses sqlhosts connectivity information that is contained in a file on UNIX or in the Windows NT registry. (See “Connectivity Information on UNIX” on page 3-14.) The communications tools that are part of all Informix products handle the tasks of locating and attaching to the database servers. To a client application, a database on a networked computer appears no different than a database on the computer where the application resides.

For information about how to define sqlhosts connectivity information, see the Administrator’s Guide. For information on the SQL statements that the database server supports, see the Informix Guide to SQL: Syntax. For detailed instruction in how to use an SQL API, see the appropriate programmer’s manual, as listed in Chapter 4, “Using the Documentation.”

**Supported Connection Types**

A connection is a logical association between two database servers, or between a client application 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. Client applications can establish multiple connections to the database server to access more than one database on behalf of a single user.
Dynamic Server supports the following types of connections to communicate between client applications and database servers.

<table>
<thead>
<tr>
<th>Connection Type</th>
<th>UNIX</th>
<th>NT</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Local</td>
<td>Network</td>
</tr>
<tr>
<td>Sockets</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TLI (transport layer interface)</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Shared memory</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Stream pipe</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Named pipe</td>
<td></td>
<td>✓</td>
</tr>
</tbody>
</table>

For more information about supported connectivity protocols for your operating system, read the machine notes file on UNIX. For information on how to set up local and network connections for Informix database servers and client applications, see the Administrator’s Guide.

**Supported Network Connections**

To establish a network connection between a client on one computer and a database server on another computer, you must use a combination of a network interface and a network protocol.

Dynamic Server supports the following types of interface/protocol combinations for network connections.

<table>
<thead>
<tr>
<th>Interface</th>
<th>Network Protocol</th>
<th>Windows NT</th>
<th>UNIX</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sockets</td>
<td>TCP/IP</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>TLI (transport layer interface)</td>
<td>TCP/IP</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>TLI (transport layer interface)</td>
<td>IPX/SPX</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

For more information about the network connections that the database server supports, see the Administrator’s Guide.
Supported Client/Server Configurations

Although a typical client/server configuration places a client application on one computer and a database server on another, a client application and database server can also reside on the same computer. The client/server environment is often a multiuser environment, with several clients accessing the same database server.

Each type of client/server configuration has its advantages and disadvantages. Considerations include ease of use, data availability, distributed transaction processing, shared memory, response time, and so on.

For more information about the client/server configurations that the database server supports, see the Administrator’s Guide.

Scalability

Dynamic Server lets you scale resources in relation to the demands that applications place on the database server. Dynamic scalable architecture provides the following performance advantages for both single-processor and multiprocessor platforms:

- DSA allows a small number of database server processes to service a large number of client application processes, producing the following benefits:
  - Reduced operating-system overhead (fewer processes to run)
  - Reduced overall memory requirements
  - Reduced contention for resources within the RDBMS

- DSA provides more control over setting priorities and scheduling database tasks than the operating system does.

Dynamic Server employs single-processor or symmetric multiprocessor computer systems. In an SMP computer system, multiple CPUs or processors all run a single copy of the operating system, sharing memory and communicating with each other as necessary.

For detailed information on dynamic scalable architecture, see the Administrator’s Guide. For information on performance tuning, see your Performance Guide.
High Performance

Dynamic Server achieves high performance through the following mechanisms:

- Raw (unbuffered) disk management
- Memory management
- Dynamic thread allocation
- Fragmentation
- Parallelization

Raw (Unbuffered) Disk Management

Dynamic Server can use both file-system disk space and raw disk space in UNIX and Windows NT environments. When the database server uses raw disk space, it performs its own disk management by using raw devices. By storing tables on one or more raw devices instead of in a standard operating-system file system, the database server can manage the physical organization of data and minimize disk I/O. Doing so results in three performance advantages:

- No restrictions because of operating-system limits on the number of tables that can be accessed concurrently
- Optimization of table access by guaranteeing that rows are stored contiguously
- Elimination of operating-system I/O overhead by performing direct data transfer between disk and shared memory

If these issues are not a primary concern, you can configure the database server to use regular operating-system files to store data. In this case, Dynamic Server manages the file contents, but the operating system manages the I/O.

For information on disk management, see the *Administrator’s Guide*. 


**Memory Management**

Dynamic Server provides two main 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. After one application reads data from a table, other applications can access whatever data is already in memory. Disk access, and the corresponding degradation in performance, might not occur because 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. 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 managing the various aspects of memory to increase performance, see your *Performance Guide*.

**Dynamic Thread Allocation**

Dynamic Server supports multiple client applications using a relatively low number of processes called *virtual processors*. A virtual processor is a multi-threaded process that can serve multiple clients and, where necessary, run multiple threads to work in parallel for a single query. In this way, the database server provides a flexible architecture that is well suited for both online transaction processing (OLTP) and decision-support system (DSS) applications. For more information on OLTP and DSS, see “OLTP Applications” on page 2-24 and “DSS Applications” on page 2-25.

For information on how virtual processors can dynamically allocate threads, see the *Administrator’s Guide*.
High Performance

**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.

The database server stores the rows in separate database spaces (dbspaces) that you specify in a fragmentation strategy. A dbspace is a logical collection of one or more database server chunks. Chunks represent specific regions of disk space.

Dynamic Server supports the following distribution 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.

Table fragmentation can improve:

- single-user response time.
- concurrency.
- availability.
- backups and restores.

For information on the different fragmentation strategies Dynamic Server supports, see the *Informix Guide to Database Design and Implementation*. For information about how to formulate a fragmentation strategy to enhance database performance, see your *Performance Guide*. For information about the CREATE TABLE and CREATE INDEX statements, see the *Informix Guide to SQL: Syntax*. 
Parallelization

Dynamic 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 are initiated by decision-support applications. 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 Informix Guide to SQL: Reference.

Fault Tolerance and High Availability

Dynamic Server uses the following logging and recovery mechanisms to protect data integrity and consistency in the event of an operating-system or media failure:

- Storage space backups
- Logical-log backups
- Fast recovery
- Mirroring
- High-availability data replication
- Point-in-time recovery

For information on fault tolerance and high availability, see the Administrator’s Guide.

Dynamic Server also includes Enterprise Replication, which allows you to replicate information from one database table to a database table on another database server. However, Enterprise Replication is explicitly designed to allow sporadic connections between database servers, which makes it unsuitable for protecting data integrity.
Fault Tolerance and High Availability

**Storage Space Backups**

The database server lets you back up the data that it manages. After a media (hardware) failure, if critical data was not damaged (and the database server remains on-line), you can restore only the data that was on the failed media, leaving other data available during the restore.

For information about backing up data, see the *Archive and Backup Guide* if you use ON-Archive, or the *Backup and Restore Guide* if you use ON-Bar.

**Logical-Log Backups**

The database server also lets you store changes to the database server and data since the backup was performed. The changes are stored in logical-log files.

You can create backup tapes and logical-log backup tapes while users are accessing the database server. You can also use on-line archiving to create incremental backups. Incremental backups let you back up only data that has changed since the last backup, which reduces the amount of time that a backup would otherwise require.

For information about logical and physical logs, see the *Administrator’s Guide*. Logical-log record formats are discussed in the *Administrator’s Reference*.

**Fast Recovery**

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, that implies that the database server was shut down in a controlled fashion. If the physical log is not empty, Dynamic Server automatically performs an operation called *fast recovery*. 
Fault Tolerance and High Availability

Fast recovery automatically restores Dynamic Server databases to a state of physical and logical consistency after a system failure that might have left one or more transactions uncommitted. During fast recovery, the database server uses its logical log and physical log to perform the following operations:

- Restore the databases to their state at the last checkpoint
- Roll forward all committed transactions since the last checkpoint
- Roll back any uncommitted transactions

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 eliminates data losses that result from media failures. If mirrored data becomes unavailable for any reason, the mirror of the data is made available immediately and transparently to users.

Mirroring is a strategy that pairs a *primary chunk* of one defined dbspace 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.

Ideally, you should mirror all of your data. However, if cost is a problem, you should select certain critical chunks to mirror. For example:

- The root chunk contains critical information and should always be mirrored.
- Logical-log files contain critical information and should always be mirrored.
- Frequently queried data should be mirrored, if possible, because mirroring improves performance.
The operating system or hardware that you use might provide an alternative form of mirroring. If you consider a mirroring option provided by your operating system instead of by the database server, compare the implementation of both options before you decide which to use. Operating-system mirroring options that do not use parallel mirror writes and split reads (reads from both the primary disk and the mirror disk) might provide inferior performance to mirroring by the database server.

Database server mirroring and operating-system mirroring run independently and can run at the same time. For example, you might have both database server data and non-database server data on a single disk drive. You could use operating-system mirroring to mirror the non-database server data and database server mirroring to mirror the database server data.

Logical-volume managers and hardware mirroring are other alternative mirroring solutions. Saving data to more than two disks with logical-volume managers gives you added protection from media failure, but the additional writes have a performance cost. Hardware mirroring such as RAID (redundant array of independent disks) has the advantage of requiring less disk space to store the same amount of data than does Informix database server mirroring, but it is slower for write operations.

For information about mirroring, see the *Administrator’s Guide*.

**High-Availability Data Replication**

If your organization requires a high degree of availability, you can replicate Dynamic Server and its databases, running simultaneously on a second computer. Replication of the database server and its databases provides a backup system in case of a catastrophic failure. If one site experiences a disaster, you can immediately direct applications to use the second database server in the pair.

Running data replication also allows you to balance read-only applications (such as DSS applications) across both database servers in the data-replication pair.

For information about high-availability data replication, see the *Administrator’s Guide*. 
**Point-in-Time Recovery**

Use point-in-time recovery after a catastrophic event to restore the data in a database to a specific time, perhaps immediately preceding the catastrophic event. A point-in-time recovery can undo mistakes, such as dropping a table, that might not be fixable otherwise.

When you restore the database server to a specific time, some transactions might be lost even though they are included in an existing logical-log backup. Because you can only restore the database server to the last known global point of consistency across all database servers, you cannot retrieve transactions that were committed after the specified recovery point.

For information about point-in-time recovery and data restoration, see the *Archive and Backup Guide* if you use ON-Archive, or the *Backup and Restore Guide* if you use ON-Bar.

**Multimedia Support**

Dynamic Server supports simple large objects (TEXT and BYTE data types) and smart large objects (BLOB and CLOB data types) that place no practical limit on the size of the stored data item or object. This type of data is stored either with other database data or in specially designated portions of the disk called blobspaces (for TEXT and BYTE) and sbspaces (for BLOB and CLOB).

For information about supported data types and how to use them, see the *Informix Guide to SQL: Reference* and the *Informix Guide to Database Design and Implementation*. For information about storing TEXT and BYTE data on write-once-read-many (WORM) optical devices with Dynamic Server, see the *Guide to the Optical Subsystem*. For information about additional multimedia data types such as image, audio, video, and user defined, see the related DataBlade module documentation.
Dynamic System Administration

Distributed databases require dynamic system administration tools that monitor and fine tune the following system parameters:

- CPU and memory utilization
- Asynchronous I/O
- Decision support
- Available disk space
- Efficient partitioning schemes

The database server provides various administration tools to help you perform these tasks. Tools include the system-monitoring interface (SMI), memory grant manager, and onperf utility.

For information on SMI, see the Administrator’s Reference. For information on the memory grant manager and the onperf utility, see your Performance Guide.

Distributed Databases and Transactions

Dynamic Server lets you query (and update) more than one database across multiple database servers within a single transaction. This ability is called distributed database processing.

Distributed databases are useful because operations that access databases are often distributed into separate pieces, either organizationally, geographically, or both. Distributed databases provide the following advantages:

- Local data can be kept where it is more easily maintained and is more frequently used.
- Data from remote sites is available to all users.

Dynamic Server lets you perform transactions on data from databases that reside on different database servers connected across a network. The database server supports two multiphase protocols, two-phase commit and heterogeneous commit, to process transactions that span multiple database servers.
**Two-Phase Commit**

A two-phase commit protocol ensures that transactions are uniformly committed or rolled back across the multiple database servers. This protocol governs the order in which a two-phase commit transaction is performed and provides an automatic recovery mechanism in case a system or media failure occurs during execution of the transaction.

A database server automatically uses the two-phase commit protocol for any transaction that performs modifications to data on more than one database server. The database server uses logical-log records to implement the two-phase commit protocol. You can use these logical-log records to detect heuristic decisions and, if necessary, to help you perform a manual recovery.

**Heterogeneous Commit**

In the context of Informix database servers, the term *heterogeneous environment* refers to a group of database servers in which at least one is not an Informix database server. Heterogeneous commit is a database server feature that ensures the all-or-nothing basis of distributed transactions in a heterogeneous environment.

Unlike the two-phase commit protocol, the heterogeneous commit protocol supports the participation of a non-Informix participant. The non-Informix participant, called a gateway participant, must communicate with the coordinator through an Informix Enterprise Gateway product.

For information about two-phase commit and heterogeneous commit protocols and concepts and the use of logical-log records in distributed databases and transactions, see the *Administrator’s Guide*. For information about a specific Informix Enterprise Gateway product, see the appropriate *Enterprise Gateway User Manual*. 
Database Server Security

Dynamic Server provides the following security features:

- Database-level security
- Table-level security
- Role creation

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.
- Combine the GRANT and CREATE VIEW statements to precisely control the parts of a table that a user can modify.
- 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 on a database server-wide basis. 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 act as evidence of database server abuse.

Informix database servers follow UNIX security requirements for making connections. Thus, the UNIX system administrator might need to make modifications to the `/etc/passwd`, `/etc/hosts`, `~/.rhosts`, and other related files.

For information about database and table privileges, and controlling access to databases, see the Informix Guide to Database Design and Implementation. For the syntax and description of SQL statements, see the Informix Guide to SQL: Syntax. For information on auditing, see the Trusted Facility Manual.
Graphical Administration Tools

Dynamic Server supports the following graphical administration tools:

- **onperf** performance utility
  
  The **onperf** utility displays database server activity. The **onperf** utility allows you to monitor most of the same metrics that **onstat** reports. ♦

- Windows NT interfaces for administering Dynamic Server ♦

- Web-based interfaces for administering Dynamic Server

For information about **onperf**, see your *Performance Guide*. For information about Windows NT interfaces and Web-based interfaces that you can use to administer Dynamic Server on both UNIX and Windows NT platforms, see the documentation notes that are described in “Documentation Notes, Release Notes, Machine Notes” on page -10 of the introduction.
Informix Dynamic Server Features

In This Chapter ........................................ 2-3
RDBMS Features .................................... 2-3
  Structured Query Language (SQL) ........... 2-3
  Stored Procedure Language (SPL) ............ 2-4
System Catalog ..................................... 2-5
Data Types .......................................... 2-6

Data-Management Features ................. 2-7
  Data Replication .................................. 2-7
    High-Availability Data Replication ........... 2-8
    Enterprise Replication ....................... 2-8
  Cost-Based Optimizer ......................... 2-9
  High-Performance Loader ..................... 2-9
Database Server Utilities .................... 2-10

Data-Recovery and Data-Restore Features 2-12
  The ontape Utility ................................ 2-13
  On-Line Archive Recovery System (ON-Archive) 2-13
  On-Line Backup and Restore (ON-Bar) ......... 2-14
Informix Storage Manager .................... 2-15
Tools That You Cannot Use for Recovery .... 2-15

Database Support .................................. 2-16
  ANSI-Compliant Databases ..................... 2-16
  Distributed Databases ......................... 2-17
  Databases Distributed Across Multiple Vendor Servers 2-18
  Dimensional Databases ....................... 2-19
  Object-Relational Databases .................. 2-20
    Extensibility .................................. 2-20
    Inheritance .................................... 2-20
Access Methods ........................................ 2-21
Primary Access Methods .................................. 2-21
Secondary Access Methods .................................. 2-21
  Generic B-Tree Indexes .................................. 2-22
  R-Tree Indexes ........................................ 2-22
Writing Access Methods .................................. 2-22
  R-Tree Access Methods .................................. 2-23
  User-Defined Primary Access Methods .................. 2-23
  User-Defined Secondary Access Methods .................. 2-24
Application Types ....................................... 2-24
  OLTP Applications ..................................... 2-24
  DSS Applications .................................... 2-25
Global Language Support .................................. 2-25
Other Database Server Features ......................... 2-26
  Support for Long Identifiers and User Names ............ 2-26
  Shared-Statement Caching ............................... 2-26
  Locking ............................................... 2-27
  Isolation Level ....................................... 2-28
  Transaction Logging .................................... 2-28
  Rolling Back Statements in a Transaction ................ 2-28
  Role Separation ...................................... 2-29
  The Information Schema ................................ 2-29
Extending Dynamic Server ................................ 2-30
  Extended Data Types ................................... 2-30
    User-Defined Data Types ............................... 2-30
    Complex Data Types .................................. 2-31
  Simple and Smart Large Objects ......................... 2-32
    Simple Large Objects ................................ 2-32
    Smart Large Objects and Sbspaces ..................... 2-33
  User-Defined Routines and DataBlade Modules ........... 2-33
    User-Defined Routines ............................... 2-34
    DataBlade Modules ................................... 2-35
    User-Defined Aggregates .............................. 2-36
  User-Defined Virtual Processors ....................... 2-36
    User-Defined Access Methods ......................... 2-36
In This Chapter

This chapter provides an overview of relational database management features and touches on the significant features of Dynamic Server.

For more information about the database server features discussed in this chapter, see the Administrator’s Guide and Administrator’s Reference. For a glossary of terms that are used in this and other Informix manuals, see the Informix Guide to SQL: Reference.

RDBMS Features

As outlined in Chapter 1, “Informix Dynamic Server Architecture,” an Informix ORDBMS consists of a database server, a database, and one or more client applications. This chapter discusses the database and database server components. For information on client applications, see “Tools and Client Products” on page 3-4.

Structured Query Language (SQL)

Dynamic Server works with relational and object-relational databases. Use Informix SQL to direct all operations on a database. SQL is composed of statements, each of which begins with one or two keywords that specify a function.

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.
ESQL/C is an SQL API that lets you embed SQL statements directly into a C program. DB-Access lets you execute SQL statements interactively.

The client application interacts with you, prepares and formats data, and sends data requests to the database server. The database server interprets the requests, manages the database, and returns data to the client application.

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.

For information about database management, see the *Informix Guide to Database Design and Implementation*. For information about how to create and use SQL, see the *Informix Guide to SQL: Tutorial* and the *Informix Guide to SQL: Syntax*. For information about embedded SQL, see the *Informix ESQL/C Programmer’s Manual*. For information about using SQL interactively and practicing SQL statements with the demonstration databases, see the *DB-Access User’s Manual* and the *Informix ESQL/C Programmer’s Manual*.

**Stored Procedure Language (SPL)**

Informix Stored Procedure Language (SPL) is an extension to SQL that provides flow control such as looping and branching. You can use SPL statements to write user-defined routines and store them in the database. SPL statements are available with products such as Informix ESQL/C and DB-Access.

An SPL routine is parsed, optimized, and stored in the system catalog tables in executable format. An SPL routine can execute routines written in C or Java. External routines can execute SPL routines.

You can use SPL routines to perform any task that you can perform in SQL and to expand what you can accomplish with SQL alone. SPL routines can improve database performance because they are parsed and optimized when they are created rather than at runtime, and because SPL is native to the database. SPL routines can also simplify writing applications and limit or monitor access to data. They also can reduce traffic between a client application and the database server and reduce program complexity.
System Catalog

The system catalog consists of tables that 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 an element in the database. You can query the system catalog tables as you would query any other table in the database.

System catalog tables track the following objects:

- Tables and constraints
- Views
- Triggers
- Authorized users and privileges
- User-defined routines
- Data types
- Casts
- Access methods and operator classes
- User-defined routines
- Inheritance relationships

For information about the structure, contents, and use of the system catalog, see the Informix Guide to SQL: Reference.
Data Types

Every column in a table in an object-relational database is assigned a data type, which precisely defines the kinds of values that you can store in that column. Dynamic Server supports the built-in and extended data types that Figure 2-1 shows.

The database server uses casts to perform data conversion between values of two different data types. For more information on extensibility, see “Extending Dynamic Server” on page 2-29.

For a description of the data types and data type conversions that Informix database servers support, see the Informix Guide to SQL: Reference. For information about choosing data types for your relational or object-relational database, see the Informix Guide to Database Design and Implementation. For information on how to extend existing data types, create new casts, and define new data types for an Dynamic Server database, see Extending Informix Dynamic Server 2000 and Informix Guide to Database Design and Implementation.
Data-Management Features

Dynamic Server includes the following data management tools and utilities:

- Data replication
- Cost-based optimizer
- High-Performance Loader
- Administration utilities

Data Replication

Data replication is the process of representing database objects at more than one distinct site. For example, one way to replicate data is to copy a database to a database server installed on a different computer. This copy allows reports to access the data without disturbing client applications that use the original database.

Dynamic Server supports two kinds of data replication:

- High-availability data replication
- Enterprise Replication

Advantages of data replication are as follows:

- Clients at the site to which the data is replicated experience improved performance because those clients can access data locally rather than connecting to a remote database server over a network.
- Clients at all sites experience improved availability of replicated data. If the local copy of the replicated data is unavailable, clients can still access the remote copy of the data.

These advantages do not come without a cost. Data replication obviously requires more storage, and updating replicated data can take more processing time than updating a single object.
Data Replication

High-Availability Data Replication

As introduced in "High-Availability Data Replication" on page 1-20, Informix high-availability data replication provides a way to maintain an exact copy of an entire database server instance that applications can access quickly in the event of a catastrophic failure. All data managed by one database server is dynamically updated on another database server, often at a separate geographical location.

For more information about high-availability data replication, see the Administrator’s Guide.

Enterprise Replication

Enterprise Replication provides a cost-effective, efficient means to replicate data at the table level throughout an open-systems enterprise. Use Enterprise Replication to replicate a table or subset of a table (such as a set of rows or columns) to other databases in your enterprise.

Enterprise Replication is a client/server application that lets you replicate data asynchronously between an unlimited number of database servers and between heterogeneous hardware platforms throughout your enterprise, including UNIX and Windows NT. A graphical user interface (GUI) helps you define, monitor, and control your replication system.

The key advantages of Enterprise Replication are as follows:

- **High performance**
  
  Log-based transaction capture and a parallel distribution mechanism ensure that the replication system does not burden the database server and that it uses networks efficiently.

- **Consistent information delivery**
  
  Enterprise Replication stores transactions and implements conflict resolution to maintain consistency and ordering of the data.

- **Flexible architecture**
  
  Enterprise Replication supports a full range of ownership models to meet a wide spectrum of business and application requirements.

- **Centralized administration**
  
  A single integrated point of administration lets the DBA graphically configure and monitor the replicated system.
Cost-Based Optimizer

Before a query is executed, the query optimizer formulates an execution plan based on the lowest resource-use cost 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 determines whether one or multiple indexes should be used. If the query involves multiple tables (or multiple database servers), the optimizer determines how each table should be queried and the order in which the tables should be evaluated. The optimizer assesses each query plan on a cost basis and selects the plan with the lowest execution cost.

For more information on the cost-based optimizer, see your Performance 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.
The HPL has the following components:

- The ipload utility (a Motif GUI) ♦
- The onpload utility
- The onpload database

The HPL provides the following key features:

- The HPL supports ASCII, multibyte, COBOL, and binary input data. You can add custom drivers to support other data types.
- The HPL can load and unload data that has a different GLS locale than that of the database server.
- The client/server architecture of the HPL lets you use a graphical user interface on any computer on your network.
- You can load data from or unload data to files, tapes, or application pipes, or to any combination of these three device types.
- The HPL provides context-sensitive on-line help.
- Any database server on your network can access the onpload database, which allows centralized management of your load and unload controls.

For complete details of how to access and use the HPL, including tutorial examples that take you through the process of loading and unloading data, see the Guide to the High-Performance Loader.

Database Server Utilities

Dynamic Server includes utilities that let you perform administrative tasks. The following table lists the supported utilities.

<table>
<thead>
<tr>
<th>Utility</th>
<th>Use</th>
<th>Where Described</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbexport</td>
<td>Unload a database into text files</td>
<td>Informix Migration Guide</td>
</tr>
<tr>
<td>dbimport</td>
<td>Create and populate a database from text files</td>
<td>Informix Migration Guide</td>
</tr>
<tr>
<td>dbload</td>
<td>Load data into databases or tables</td>
<td>Informix Migration Guide</td>
</tr>
</tbody>
</table>

(1 of 2)
### Database Server Utilities

<table>
<thead>
<tr>
<th>Utility</th>
<th>Use</th>
<th>Where Described</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbschema</td>
<td>Create a file that contains the SQL statements needed to replicate a specified table, view, or database, or view the Information Schema</td>
<td>Informix Migration Guide</td>
</tr>
<tr>
<td>onaudit</td>
<td>Manage audit masks and auditing configurations</td>
<td>Trusted Facility Manual</td>
</tr>
<tr>
<td>oncheck</td>
<td>Check specified disk structures for inconsistencies, repair inconsistent index structures, and display information about disk structures</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>ondblog</td>
<td>Change the logging mode</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>oninit</td>
<td>Bring the database server on-line</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onload</td>
<td>Load data that was created with onunload into the database server</td>
<td>Informix Migration Guide</td>
</tr>
<tr>
<td>onlog</td>
<td>Display the contents of logical-log files</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onmode</td>
<td>Change the database server operating mode and perform various other operations on shared memory, sessions, transactions, parameters, and segments</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onparams</td>
<td>Modify the configuration of logical logs or physical logs</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onperf</td>
<td>Monitor database server performance</td>
<td>Performance Guide</td>
</tr>
<tr>
<td>onshowaudit</td>
<td>Extract information from an audit trail</td>
<td>Trusted Facility Manual</td>
</tr>
<tr>
<td>onspaces</td>
<td>Modify blobspaces or dbspaces</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onstat</td>
<td>Monitor the operation of the database server</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>ontape</td>
<td>Log, archive, and restore data as outlined in “The ontape Utility” on page 2-12</td>
<td>Administrator’s Reference</td>
</tr>
<tr>
<td>onunload</td>
<td>Unload data from the database server</td>
<td>Informix Migration Guide</td>
</tr>
</tbody>
</table>

(2 of 2)
Data-Recovery and Data-Restore Features

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

Data-Recovery and Data-Restore Features

Data might be lost or corrupted for reasons that range from a program error to a disk failure to a disaster that damages the facility in which your computer resides. Dynamic Server includes the following features that help you recover and restore data:

- The ontape utility
- ON-Archive
- ON-Bar
- Informix Storage Manager

To use these features, you must understand the following terms:

- An archive is a copy of one or more database server dbspaces (database spaces) and any supporting data that you might need to restore them.
- 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 archives.
- A restore re-creates data, particularly dbspaces, from an archive and backed-up logical-log files. You must restore the data in two operations:
  - A physical restore of dbspace or blobspace data from an archive
  - A logical restore that accesses a logical-log backup to re-create in the restored dbspaces any transactions that were generated after the archive

For more information about these and other terms, see the Administrator’s Guide and the Glossary in the Informix Guide to SQL: Reference. For a list of differences between ontape and ON-Archive, see the Archive and Backup Guide.
The ontape Utility

The ontape utility options of your database server perform the following tasks:

- Archive data that the database server manages
- Change database-logging status
- Back up logical-log files
- Start continuous logical-log file backups
- Restore data from an archive tape
- Use data replication

The ontape utility is interactive and, when necessary, prompts you to mount new tapes or to provide input.

On UNIX platforms, you must be logged in as user root or user informix to execute ontape. On Windows NT platforms, you must be a member of the Informix-Admin group.

**Important:** Do not start ontape in background mode (that is, using the UNIX ampersand (&) operator on the command line), because you might miss prompts and delay an operation.

For information about the ontape utility, see the Administrator’s Reference.

On-Line Archive Recovery System (ON-Archive)

The ON-Archive recovery system performs the same archive and recovery tasks as the ontape utility and provides the following additional features:

- Scheduling and tracking of archives and backups
- Tape reliability and storage options
- Multiple ways of securing data and access to ON-Archive
- Use of multiple tape drives simultaneously for archives and restores
- Unattended operations
- Disaster provision
ON-Archive has the following main components:

- Data to be archived, backed up, or restored
- ON-Archive commands and requests
- ON-Archive programs
- ON-Archive catalog
- Devices to write to and read from
- Media to store the archive or backup data

**Important:** The archive tapes and logical-log backups produced by `ontape` and ON-Archive are not compatible. You cannot create an archive or logical-log backup with `ontape` and restore it with ON-Archive or vice versa.

For information about ON-Archive, see the *Archive and Backup Guide*. For a detailed example of how to use ON-Archive, see the *ON-Archive Quick Start Guide*.

**On-Line Backup and Restore (ON-Bar)**

ON-Bar provides backup and restore capability for all dbspaces and logical logs. To prevent performance problems caused by overfull logical-log space, you can have logical logs backed up automatically as soon as they fill. You can schedule backups of dbspaces appropriately as insurance against system failure.

ON-Bar uses the X/Open Backup Services Application Programmer’s Interface (XBSA). It connects to either the Informix Storage Manager or another XBSA-compliant storage manager that provides device and storage management functionality.

To recover data, you first restore the backup copy of the data and then restore the logical logs to bring data as close as possible to the most recent state. You can activate the `archecker` utility from ON-Bar to check the validity and completeness of the objects that ON-Bar backs up before you use the objects to restore dbspaces.

For complete information about ON-Bar, see the *Backup and Restore Guide*. 
Informix Storage Manager

The Informix Storage Manager (ISM) provides data-storage management services for your Informix database server. It resides on the same computer as ON-Bar and the Informix 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 manages storage devices and storage volumes.

ISM has the following components:

- The ISM server, which backup and recovery information between storage devices and ON-Bar
- The ISM administrator program, which configures and manages the ISM server, storage media, and storage devices
- The ISM catalog, which maintains up-to-date records of the backup operations that have been performed and the media on which the backed-up data is stored

ISM works with ON-Bar to perform the following kinds of tasks:

- Provide complete media management for all ON-Bar backups and restores
- Track the location of all backup data and move backup data through a managed life cycle
- Provide complete disaster-recovery protection for a database server

For information about how to configure and use ISM, see the Informix Storage Manager Administrator’s Guide.

Tools That You Cannot Use for Recovery

You cannot use data-migration utilities such as onunload or dbexport as a substitute for a Dynamic Server archive. None of the data-migration utilities are coordinated with the information stored in the logical-log files and, unlike archives, they do not save system-overhead information that is important to the database server.
Database Support

For information about data-migration utilities, see the Informix Migration Guide.

Database Support

Dynamic Server supports the following types of databases:

- ANSI compliant
- Distributed
- Distributed on multiple vendor servers
- Dimensional (data warehouse)
- Object-relational

ANSI-Compliant Databases

Dynamic Server supports ANSI-compliant databases. An ANSI-compliant database enforces ANSI requirements, such as implicit transactions and required ownership, that are not enforced in databases that are not ANSI compliant.

You must decide whether you want any of the databases to be ANSI compliant before you connect to a database server. ANSI-compliant databases and databases that are not ANSI-compliant differ in the following areas:

- Transaction processing
- Transaction logging
- Owner naming
- Privileges on and access to objects
- Default isolation level
- Character and decimal data types
- Escape characters
- Cursor behavior
- SQLCODE of the SQL Communications Area (SQLCA)
Distributed Databases

Dynamic Server provides distributed database processing capabilities that let you query and update more than one database across multiple database servers within a single transaction. A transaction is a collection of SQL statements that is treated as a single unit of work.

A distributed database has information in several databases. The data can be maintained by a variety of database servers and located on computers that use different operating systems and communication networks. Figure 2-2 shows the configuration of a distributed database, which includes databases maintained by multiple Informix database servers.

For information about ANSI-compliant databases, see the Informix Guide to Database Design and Implementation and the Informix Guide to SQL set of manuals.

- Valid SQL statements
- Synonym behavior
- Data recovery
- Portability

Figure 2-2
Configuration of a Distributed Database with Multiple Instances of Database Servers
Databases Distributed Across Multiple Vendor Servers

Distributed databases are useful because operations that use databases are often distributed into separate pieces, either organizationally, geographically, or both. A distributed database provides the following advantages:

- Local data can be kept where it is most easily maintained and most frequently used.
- Data from remote sites is available to all users.
- Duplicate copies can be maintained in case one database server fails.

A distributed database system has the following disadvantages:

- Management of the distributed database server is more involved than management of a single-host system.
- Network access is often slower than local access.

For information on transaction concepts, see the Informix Guide to SQL: Tutorial.

Databases Distributed Across Multiple Vendor Servers

TP/XA is a library of functions that lets an Informix database server act as a resource manager in a distributed transaction-processing (DTP) environment. The TP/XA library is installed as part of Informix ESQL/C to facilitate communication between a third-party transaction manager and Informix Dynamic Server.

You can use Informix database servers with the TP/XA library to create global transactions that span multiple computers and even multiple XA-compliant database servers from different vendors. TP/XA lets you use database-management systems from multiple vendors to store and access your data.

You might want to use the TP/XA feature when your database environment has the following characteristics:

- Data is distributed across multivendor databases.
- Transactions include Informix and non-Informix data.

The TP/XA library and DTP environment adhere to the XA interface specification developed by the X/Open Company to support large-scale, high-performance, on-line transaction-processing applications.

For more information, see the TP/XA Programmer’s Manual.
Dynamic Server supports the concept of data warehousing. This typically involves a dimensional database that contains large stores of historical data.

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 point in time.

A data warehouse integrates and transforms the data that it retrieves before it is loaded into the warehouse. A primary advantage of a data warehouse is that it provides easy access to, and analysis of, vast stores of information.

A data-warehousing environment can store data in one of the following forms:

- **Data warehouse**
  A data warehouse is a database that is optimized for data retrieval. Data is not stored at the transaction level; some level of data is summarized. A data warehouse provides a decision-support environment in which you can evaluate the performance of an entire enterprise over time.

- **Data mart**
  A data mart is a subset of a data warehouse that is stored in a smaller database. It is oriented toward a specific purpose or subject rather than enterprise-wide strategic planning. It can contain operational data, summarized data, spatial data, or metadata.

- **Operational data store**
  An operational data store is a subject-oriented system for decision making that is optimized for looking up one or two records at a time. It is a hybrid form of data warehouse that contains timely, current, integrated information that typically is of a higher level granularity than the transaction. This data can serve as the common source of data for data warehouses.

- **Repository**
  A repository is a system that combines multiple data sources into one normalized database. The records in a repository are updated frequently. Data stored in a repository is operational rather than historical and might be used for specific decision-support queries.
Object-Relational Databases

For details of how to plan a database, build and implement a traditional relational database or dimensional database, and work with data types and fragmentation, see the *Informix Guide to Database Design and Implementation*. See also the *DB-Access User’s Manual*.

**Object-Relational Databases**

Dynamic Server is an object-relational database server that combines object-oriented and relational capabilities yet still represents all data in the form of tables with rows and columns. In addition to providing support for alphanumeric data such as character strings, integers, decimals, and dates, the database server provides the object-oriented capabilities of extensibility and inheritance.

**Extensibility**

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. You can create UDRs in SPL, Java, and the C programming language. For more information about the extensible features that Dynamic Server supports, see “Extending Dynamic Server” on page 2-29.

Informix 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). A DataBlade might also provide new types of access to large text documents, including phrase matching, fuzzy searches, and synonym matching.

**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.
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.

Primary Access Methods

A primary access method is a set of database server functions that access and manipulate data. It is a set of routines that perform operations such as inserting, deleting, updating, and scanning data in a table. 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 Informix Guide to SQL: Syntax and Extending Informix Dynamic Server 2000.

Secondary Access Methods

A secondary access method is a set of routines that implement an index. 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).
  You can create a functional index on the resulting values of a function on one or more columns. The function must be a user-defined function. When you create a functional index, the database server computes the index-key values by executing the function on each column value.

For more information on B-trees and functional indexes, see your Performance Guide.

R-Tree Indexes

The R-tree indexing structure supports spatial data. Dynamic Server provides the framework for R-trees, but does not provide the routines required to support R-trees.

Writing Access Methods

You can write routines that provide R-tree indexing and custom primary and secondary access methods.
**R-Tree Access Methods**

An R-tree index uses a concept called 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 *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. An extspace is a description of an external location. It provides a way to associate a logical name with a location string such as a pathname or network gateway. You can specify an external space as the storage space for a table for which you create a primary access method.

You can access any of 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

Because you can implement primary access methods as client applications, they typically do not require any direct interaction with low-level database server modules. Thus, they are relatively simpler to develop and support than new storage or indexing strategies. Primary access methods can unify all of the heterogeneous data that is distributed throughout an organization under a single object-relational system.

For information on extspaces and user-defined access methods, see the *Administrator’s Guide* and *Administrator’s Reference*. For information on how to create primary access methods, contact your Informix sales representative for the *Virtual-Table Interface Programmer’s Manual*. 
Application Types

User-Defined Secondary Access Methods

In many cases, index data is stored outside the Informix data space. However, you can build an access method to access 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, contact your Informix sales representative for the Virtual-Index Interface Programmer’s Manual.

Application Types

Two main classes of applications operate on data in a relational 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. These operations typically involve quick, indexed access to a small number of rows. An order-entry system is a typical example of an OLTP application. OLTP applications are often multiuser applications with acceptable response times measured in fractions of seconds.

OLTP applications have the following characteristics:

- Simple transactions that involve small amounts of data
- Indexed access to data
- Many users
- Frequent requests
- 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 managing decision-support systems, see your Performance Guide.

Global Language Support

Informix products include the Global Language Support (GLS) feature. The GLS feature lets the database server handle different languages, cultural conventions, and code sets through the use of 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.

GLS provides support for the following language-related items:

- Collation order of characters
- Definition of uppercase and lowercase conventions
- Non-ASCII characters, including multibyte characters
- Culture-specific formatting for numeric, monetary, date, and time values

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 Informix product.

For complete information about the GLS feature, see the Informix Guide to GLS Functionality. For information on how to program Informix ESQL/C and DataBlade client applications with Informix GLS, see the Informix GLS Programmer’s Manual.
Other Database Server Features

This section outlines a number of other features that Informix database servers share. Most of the features discussed involve the use of SQL statements.

Support for Long Identifiers and User Names

Dynamic Server supports long identifiers and long user names:

- **An identifier** is a sequence of letters, digits, and underscores (_) that represents the name of a database, table, column, cursor, function, index, synonym, alias, view, prepared object, constraint, or procedure name. An identifier can include the dollar sign ($) as any but the first character of the identifier.
  
  The supported identifier length is extended from 18 bytes to 128 bytes (VARCHAR(128)).

- **A user name** is the Informix, login, or client user ID that identifies a user account on a database server or network.
  
  The supported user name length is extended from 8 bytes to 32 bytes (CHAR(32)).

**Tip:** In languages such as English, one character is 1 byte. However, in some languages, such as Chinese, one character can be several bytes.

Set the `IFX_LONGID` environment variable to indicate whether a particular client is capable of handling long identifiers.

For information on long identifiers, see the *Informix Guide to SQL: Syntax*, *Informix Guide to SQL: Tutorial*, and *Informix Guide to SQL: Reference*.

Shared-Statement Caching

The shared-statement cache feature reduces database server memory consumption by increasing the sharing among different user sessions of data structures that represent identical SQL statements. Query plans for SQL statements are cached for subsequent use by other user sessions.
The shared-statement cache feature lets you:

- Dramatically reduce memory consumption by the database server because sharable data structures associated with an SQL statement do not require per-session memory allocation.
- Speed up query processing because subsequent uses of SQL statements that are cached bypass the optimization stage.

The `STMT_CACHE` environment variable lets you control the use of the statement cache on a session. The SQL statement `SET STATEMENT CACHE` lets you control the use of cache within an application.

For information about performance considerations and how to monitor shared-statement caches, see your *Performance Guide*. For information on environment variables, see the *Informix Guide to SQL: Reference*.

**Locking**

*Locking* is the process of temporarily limiting access to an object (database, table, page, or row) to prevent conflicting interactions among concurrent processes. When you write applications, be aware of how the database server handles the following locking issues:

- Database locking
- Table locking
- Page and row locking
- Lock mode
- Shared locks
- Waiting for locks
- Byte range locking (for smart large objects)

For information on the SQL statements that you use for locking, see the *Informix Guide to SQL: Syntax*. For general locking information, see the *Informix Guide to SQL: Tutorial*. For performance considerations, see your *Performance Guide*. 
Isolation Level

Isolation level refers to the level of independence among multiple users when they attempt to read the same data from the database. The isolation level lets you specify whether or not you are willing to read data that might be changed while you (that is, the SQL query) are reading it.

For information about isolation levels, see the Informix Guide to SQL: Syntax and the Informix Guide to SQL: Tutorial. For performance considerations for the different isolation levels, see your Performance Guide.

Transaction Logging

The database server supports buffered logging and lets you switch between buffered and unbuffered logging with the SET LOG statement. Buffered logging holds transactions in memory until the buffer is full, regardless of when the transaction is committed or rolled back. You can also choose to log or not to log data.

For information on the SET LOG statement, see the Informix Guide to SQL: Syntax. For information about buffered and unbuffered logging, see the Administrator’s Guide.

Rolling Back Statements in a Transaction

A roll back is a process that reverses an action or series of actions on a database. If you have not yet issued a statement that commits a transaction, you undo any changes that occurred since the beginning of that transaction. You can roll back any statement.

For a general discussion of transactions, see the Informix Guide to SQL: Tutorial.
Role Separation

Role separation is an option that lets users perform different administrative tasks. Role separation is based on the principle of separation of duties, which can reduce security risks by providing a checks-and-balances mechanism in the system. For example, the person who determines what to audit should be different from the person who monitors the audit trail, and both of these users should be different from the person who is responsible for the operations of the database server.

For information on role separation, see the Trusted Facility Manual and the Informix Guide to SQL: Syntax.

The Information Schema

The dbschema (database schema) utility provides information about all the tables, views, and columns of the specified database.

Use dbschema to display:

- the SQL statements (schema) that would be needed to rebuild the database.
- the distribution information that is stored for one or more tables in the database.
- information on user-defined and complex data types and row types
- the schema for the Information Schema views.

For information about DB-Access, see the DB-Access User’s Manual. For information on the dbschema utility, see the Informix Migration Guide. For information about the Information Schema views, see the Informix Guide to SQL: Reference.
Extending Dynamic Server

Dynamic Server supports the following types of data and extensibility:

- Built-in data types
- User-defined data types (opaque and distinct data types)
- Complex data types (row types and collection types)
- Type and table inheritance
- User-defined casts
- Storage of and access to simple and smart large objects
- User-defined routines (in SPL, C, and Java)
- DataBlade modules (pre-packaged data types and supporting routines)
- User-defined access methods
- User-defined virtual processors

Extended Data Types

Dynamic Server supports user-defined and complex data types.

For information on how to create opaque data types and distinct data types, see Extending Informix Dynamic Server 2000. For information on how to create and use complex types and user-defined casts, see the Informix Guide to Database Design and Implementation.

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.

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. Routines that support a new opaque type pass the contents of the structure to the database server to store in the database.
Extended Data Types

A *distinct type* has the same storage representation as an existing opaque type but is distinguished by its name and cannot be substituted for the source data type. For example, you can create a new distinct type `decnum` with the same representation as `real`. All routines that operate on `real` values have database server-simulated counterparts for `decnum` values. However, `decnum` values and `real` values cannot be added, subtracted, or compared with one another without explicitly *casting* one value to the type of the other. A cast function converts one data type to another 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. An important difference between complex types and user-defined types is that you can access and manipulate the individual components of a complex data type with SQL statements.

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 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.
Simple and Smart Large Objects

Dynamic Server supports inheritance for named row types and typed tables, which allows a type or a table to acquire the properties of another type or table. Inheritance allows for incremental modification, so that a type or table can inherit a general set of properties and add properties that are specific to itself:

- Type inheritance applies to named row types only. You can use inheritance to group named row types into a type hierarchy.
- Tables that are defined on named row types support table inheritance. Table inheritance is the property that allows a table to inherit the behavior (constraints, storage options, triggers, and so on) in the table hierarchy.

Simple and Smart Large Objects

Relational databases provide the ability to store and retrieve large chunks of binary or text data. 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 a very large amount of data. The database server supports simple large objects and smart large objects for storing large data objects in a database.

For more information on simple and smart large objects, see the Informix Guide to SQL: Reference and Informix Guide to SQL: Tutorial.

Simple Large Objects

Simple large objects (TEXT and BYTE data types) have a theoretical size limit of $2^{31}$ bytes and a practical limit that is determined by your disk capacity.

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 and Sbspaces

Smart large objects (CLOB and BLOB data types) allow client applications to seek, read from, and write to segments of the object. You can control the logging characteristic of smart large objects independently from the logging characteristics of the database. Smart large objects obey transaction isolation modes.

You can use smart large objects to store user-defined data types such as video and audio clips, pictures, large text documents, and spatial objects such as drawings and maps. Dynamic Server stores large objects in sbspaces. Sbspaces are collections of database server storage spaces that exclusively store smart large objects.

Informix ESQL/C provides client applications with the Smart Large Object application-programmer’s interface (API), which lets an application access a smart large object much like an operating-system file. The API allows you to access any portion of a smart large object.

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. For information on how to access a simple large object or a smart large object from a client application, see the Informix ESQL/C Programmer’s Manual. For information on how to calculate space and tune sbspaces, see your Performance Guide.

User-Defined Routines and DataBlade Modules

You can use one of the following methods to extend Dynamic Server data-management capabilities:

- Develop your own user-defined routines using Stored Procedure Language (SPL).
- Use features that let you add data types, supporting routines, and access methods.
- Add an Informix or third-party DataBlade module, which is a pre-packaged custom data type and all the required supporting routines.
- Create your own DataBlade module with the DataBlade Developers Kit.
User-Defined Routines and DataBlade Modules

User-Defined Routines

A user-defined routine (UDR) is a routine that you can invoke within an SQL statement or another UDR. You create the UDR in a language that the database server supports and then register it in the system catalog tables so that the database server can access it.

UDRs provide benefits in the areas of performance, optimization, and configuration management. You can write user-defined routines to accomplish the following tasks:

- Encapsulate multiple SQL statements
- Extend functions on built-in data types
- Support new data types
- Create triggered actions for multiple applications
- Restrict who can read data, change data, or create objects
- Manipulate large objects
- Facilitate interactive multimedia publication

When you write a UDR in a language other than SPL, the UDR is called an external routine. The database server supports the following types of UDRs:

- SPL routines written in a language that is internal to the database server
- External routines written in C or Java

An external routine that is written in the C language is called a C UDR. The C UDR uses the server-side implementation of the DataBlade API to communicate with the database server. SPL routines can execute routines written in C or other external languages, and external routines can execute SPL routines.

For information on implementing user-defined routines, see Extending Informix Dynamic Server 2000.
**DataBlade Modules**

A DataBlade module is a collection of database objects and routines that extends the database server by adding new functionality. DataBlade modules are standard software modules that plug into the database.

You can obtain DataBlade modules from Informix and from third-party vendors. You can create your own DataBlade modules with the DataBlade Developers Kit, which is a separate Informix product that is not included with Dynamic Server. For more information, refer to your *DataBlade Developers Kit User’s Guide*.

A DataBlade module lets the database server provide the same level of support for extended data types that it provides for built-in data types. Think of a DataBlade module as an object-oriented package (similar to a C++ class) that encapsulates specialized data types such as video, audio, photographs, documents, two-dimensional spatial objects, and so on. Figure 2-3 shows the major components of a DataBlade module.

For information on how to work with and create your own DataBlade modules, see the *DataBlade API Programmer’s Manual* and the *DataBlade Developers Kit User’s Guide*. 
User-Defined Virtual Processors

User-Defined Aggregates

A user-defined aggregate (UDA) extends the database by performing aggregation over user-defined types (UDTs) and user-defined routines (UDRs). The UDA feature lets you perform the following tasks:

- Overload existing built-in aggregate functions (AVG, COUNT, MAX, MIN, RANGE, STDEV, SUM, and VARIANCE) to support UDTs.
- Write UDRs in C, SPL, and Java that implement user-defined aggregate functions.

The UDA feature lets you re-use existing client applications.

For the SQL syntax to create and drop UDAs, see the Informix Guide to SQL: Syntax. For detailed information on UDAs, UDRs and UDTs, see Extending Informix Dynamic Server 2000.

User-Defined Virtual Processors

To isolate the effects of user-defined routines that programmers might develop, you can define a separate virtual processor class in which to execute the user-defined routines. You can designate a user-defined class of virtual processors to run DataBlade or external routines.

For information on standard virtual processors, see the Administrator’s Guide. For information on user-defined virtual processors, see Extending Informix Dynamic Server 2000.

User-Defined Access Methods

You can create new access methods, if necessary, to support new data types or to include in database information that is stored external to Dynamic Server. For more information, see “Access Methods” on page 2-20.

For information on creating a new access method, refer to the documentation for your custom access methods and to Extending Informix Dynamic Server 2000.
Getting Up and Running with the Database Server

In This Chapter ........................................... 3-3
Basic Tasks for Getting Started ....................... 3-3
Tools and Client Products .............................. 3-4
   Tools ..................................................... 3-5
       DB-Access ........................................ 3-5
       DB/Cockpit ....................................... 3-5
       Informix SNMP Subagent ....................... 3-5
       Optical Subsystem ............................... 3-6
   Client SDK Products ................................. 3-6
       DataBlade API. ................................. 3-7
       Informix ODBC Driver .......................... 3-7
       Informix ESQL/C ............................... 3-8
       Informix GLS .................................... 3-9
       Informix JDBC Driver ......................... 3-9
       TP/XA ........................................... 3-10
   Compatibility of Client and Server Programs .... 3-10
Installing Informix Products ............................ 3-10
Configuring the Database Server Environment ....... 3-11
   Required Environment Variables .................. 3-12
       Other Useful Environment Variables .......... 3-13
   Setting UNIX Environment Variables .............. 3-13
   Setting Windows NT Environment Variables ..... 3-14
Connectivity Information .............................. 3-14
   Connectivity Information on UNIX ............... 3-14
   Connectivity Information on Windows NT ........ 3-16
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Preparing the ONCONFIG Configuration File</td>
<td>3-16</td>
</tr>
<tr>
<td>Creating an ONCONFIG File on UNIX</td>
<td>3-16</td>
</tr>
<tr>
<td>Creating an ONCONFIG File on Windows NT</td>
<td>3-17</td>
</tr>
<tr>
<td>Allocating Disk Space</td>
<td>3-17</td>
</tr>
<tr>
<td>Initializing Disk Space</td>
<td>3-18</td>
</tr>
<tr>
<td>Configuring the SQL API Environment</td>
<td>3-19</td>
</tr>
<tr>
<td>Creating a Database</td>
<td>3-19</td>
</tr>
<tr>
<td>Making the Initial Connection to a Database Server</td>
<td>3-20</td>
</tr>
<tr>
<td>Creating a Demonstration Database</td>
<td>3-20</td>
</tr>
<tr>
<td>Obtaining Error Message Information</td>
<td>3-21</td>
</tr>
<tr>
<td>Migrating to Dynamic Server</td>
<td>3-21</td>
</tr>
<tr>
<td>Performing Standard Administrative Tasks</td>
<td>3-21</td>
</tr>
<tr>
<td>Monitoring Performance</td>
<td>3-22</td>
</tr>
<tr>
<td>Using Operating-System Tools</td>
<td>3-23</td>
</tr>
<tr>
<td>Using Dynamic Server Utilities</td>
<td>3-24</td>
</tr>
</tbody>
</table>
In This Chapter

This chapter provides information to help you get the database server up and running.

Basic Tasks for Getting Started

When you begin working with a new database server, you need to perform the following tasks:

- Review available tools (page 3-4)
- Install Informix products (page 3-9)
- Configure the environment
  - Set required environment variables (page 3-11)
  - Prepare connectivity files (page 3-13)
  - Prepare the configuration file (page 3-14)
  - Allocate and initialize disk space (page 3-15)
- Configure a client (SQL API) environment (page 3-17)
- Choose a database type (page 3-17)
- Connect to the database server (page 3-17)
- Create the demonstration database (page 3-18)
- Obtain error message information (page 3-18)
- Migrate data from an earlier version of the database server (if needed) (page 3-19)
- Perform administrative tasks (page 3-19)

A final section touches on monitoring the performance of Dynamic Server.
Tools and Client Products

This section discusses the Informix tools and client products that will help you manage the database server and create client applications:

- **Tools**
  - DB-Access
  - DB/Cockpit
  - Informix SNMP subagent
  - Optical Subsystem

- **SQL API and client products**
  - DataBlade API
  - DataBlade Developers Kit
  - INFORMIX-CLI
  - Informix ESQL/C
  - Informix GLS API
  - Informix JDBC Driver
  - TP/XA

Other Informix features such as the High-Performance Loader (HPL), Informix Storage Manager (ISM), and database server utilities, are discussed in Chapter 2. Dynamic Server also might support additional development tools that are not described in this manual.

For information about specific versions of (or platforms for) products that Dynamic Server supports, see the release notes described in “Documentation Notes, Release Notes, Machine Notes” on page 10 of the Introduction. See also the manual for the client product or SQL API that you want to use, or the Informix Client Software Developer’s Kit or DataBlade Developers Kit for the specific group of products.
Tools

Dynamic Server supports the tools that are discussed in this section.

**DB-Access**

DB-Access is a client that is included with Informix database servers. It lets you connect to the database server and access, modify, and retrieve information. Use DB-Access to access and manipulate the data in a database and to perform a variety of data-management tasks such as organizing, storing, retrieving, and viewing data.

For more information, see the *DB-Access User’s Manual*.

**DB/Cockpit**

The DB/Cockpit utility is a graphical tool that lets Dynamic Server administrators manage the database server, maintain its integrity, and improve Dynamic Server performance. DB/Cockpit implements a client/server architecture that lets administrators monitor many Dynamic Server system parameters that other utilities report. It combines an *onprobe* server that collects data and an *oncockpit* client program that provides a viewer for user interaction.

For more information, see the *DB/Cockpit User’s Manual*.

**Informix SNMP Subagent**

Simple Network Management Protocol (SNMP) is a published, open standard for network management. The Informix SNMP subagent provides information about Informix database servers to SNMP-compliant applications. Network and database administrators can use the information to database servers.

For more information, see the *Informix SNMP Subagent Guide*. 
Client SDK Products

Optical Subsystem

The optical storage subsystem supports the storage of 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 storage subsystem includes a specific set of SQL statements that support the storage and retrieval of data to and from the optical storage subsystem.

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

Optical media offer the following advantages over conventional magnetic disks for storing data:

- Mass storage capacity (on the order of terabytes)
- Mountable and/or unmountable storage units
- Low cost per bit of storage
- Long media life
- High data stability

For more information, see the Guide to the Optical Subsystem.

Client SDK Products

The Informix Client Software Developer’s Kit (Client SDK) package includes several application-programming interfaces (APIs) that developers can use to write applications for Informix database servers in ESQL, C, and Java. Informix Connect contains the runtime libraries of the APIs in the Client SDK.

For information about available client products that Dynamic Server supports, see the release notes, such as CLIENTS_9.2, described in “Documentation Notes, Release Notes, Machine Notes” on page 10 of the Introduction.
DataBlade API

The DataBlade API is a C-language application programming interface that is provided with Dynamic Server. Experienced C programmers can use DataBlade API functions in DataBlade modules to develop client and database server applications that access data stored in a Dynamic Server database.

With the DataBlade API, you can:

- manage client and database server connections.
- send SQL command strings.
- process query results.
- build a saveset (a mechanism to fetch multiple rows from the database introductory program memory for processing).
- manage events and errors.
- insert plain and internationalized traces.
- create database server functions.

The DataBlade API contains public data structures, public functions, and header files for DataBlade module, ESQL/C, GLS, and so on. Most DataBlade API functions behave identically in client and database server programs.

For more information, see the DataBlade API Programmer’s Manual.

Informix OBDC Driver

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 implement ODBC functionality. The Informix ODBC Driver API lets you access an Informix database and interact with an Informix database server.
Informix ODBC Driver consists of the following components:

- Informix ODBC libraries, which provide the functions and values for the API
- A driver manager, which provides an interface between an Informix ODBC application and the Informix ODBC Driver and checks parameters and transitions
- The Informix ODBC Driver, which provides an interface between a data source and a driver manager or an application

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

**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.

Informix ESQL/C consists of the following components:

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

For more information, see the *Informix ESQL/C Programmer’s Manual*.
Informix GLS

The Informix GLS library contains APIs that let programmers develop internationalized Informix ESQL/C and DataBlade module client applications. Informix GLS accesses GLS locales to handle different languages, cultural conventions, and code sets. For information on the GLS feature, see “Global Language Support” on page 2-25.

Informix GLS provides procedures, macros, and functions to:

- process single-byte and multibyte characters and strings.
  - String-processing operations include string traversal, concatenation, copying, and character searching.
  - Character-processing operations include character classification, case conversion, code-set conversion, and character comparison.
- convert date, time, monetary, and number values from and to locale-specific data formats.

The Informix GLS library supports conversion of a locale-specific string to its internal database representation and formatting of an internal database representation to a locale-specific string.

For more information, see the Informix GLS Programmer’s Manual. In addition, Informix GLS provides on-line reference pages as HTML documentation 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.

Informix JDBC Driver

Informix JDBC Driver lets Java programmers access Informix databases and interact with an Informix database server by using JDBC-compliant Java applications or applets. Programmers can create client applications that use standard JDBC to connect to Dynamic Server, query and retrieve metadata from a database or column, and handle errors. The 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 Informix JDBC Driver Programmer’s Guide.
Compatibility of Client and Server Programs

**TP/XA**

The TP/XA library is a part of Informix ESQL/C that facilitates communication between a third-party transaction manager and an Informix database server for the purpose of distributed transaction processing (DTP) in a multi-vendor database setting. The TP/XA library lets the database server operate as a database management system within a Resource Manager of an X/Open distributed transaction processing environment that conforms to XA specifications.

For more information, see the *TP/XA Programmer’s Manual*.

Compatibility of Client and Server Programs

You can execute a user-defined routine from an SQL statement or from a client application. Any function that does not require interactive input from the client application can be written as a UDR. However, you must balance the UDR load between the client and the database server to achieve optimal performance.

Installing Informix Products

You can install Informix database servers, client products, and DataBlade modules on UNIX or in a Windows environment. To install a database server or other Informix product, you must prepare the UNIX or Windows environment, load the product files supplied by Informix onto your computer, and run an installation script (UNIX) or follow instructions in the windowing environment (such as Windows NT) to correctly set up the product files.

For information about platform-specific requirements, refer to the on-line machine notes for your UNIX platform. For information about installing a database server or other Informix product in a UNIX or Windows environment, see the *Installation Guide* for the product or environment.
If you install more than one Informix product on the same computer, you do not have to install them in a specific order. You can install clients first or the database server first. However, if you install more than one Informix product, you must complete all installation procedures for one product before you start to install the next product. Do not load the files from another Informix product onto your computer until you complete the current installation.

If you want to preserve product files of earlier versions, you must create separate directories for each version of your Informix products. Set the INFORMIXDIR environment variable to the appropriate directory name for the version that you want to access.

For information about the INFORMIXDIR environment variable, see the Administrator’s Guide and the Informix Guide to SQL: Reference. For information on memory and disk space requirements for installation as well as the product installation steps, see your Installation Guide.

Configuring the Database Server Environment

After you install the database server, you must configure the database server environment. Configuration refers to setting specific parameters that affect data processing in the client/server environment.

The following list identifies the basic configuration requirements:

- Set environment variables
- Prepare the sqlhosts information in the file or registry
- Prepare the ONCONFIG file
- Allocate disk space

Before you can start configuring the database server, you must configure the operating system appropriately. You might need the assistance of the system administrator for this task. For information about operating-system configuration dependencies, refer to the machine notes described in “Documentation Notes, Release Notes, Machine Notes” on page 10 of the Introduction.

For a discussion of configuration requirements, see the Administrator’s Guide. For information about configuration parameters and files that Informix Dynamic Server uses, see the Administrator’s Reference.
Required Environment Variables

The following table shows the minimum environment variables that Informix database servers use. You must set the environment variables described in this section before an Informix client application can connect with the database server.

<table>
<thead>
<tr>
<th>Environment Variable</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>INFORMIXDIR</td>
<td>The INFORMIXDIR environment variable must be set to the full pathname of the directory where the Informix product has been installed.</td>
</tr>
<tr>
<td>INFORMIXSERVER</td>
<td>The INFORMIXSERVER environment variable specifies the name of the default database server, as defined in the ONCONFIG configuration file.</td>
</tr>
<tr>
<td>PATH</td>
<td>The PATH environment variable must include the full pathname of the directory where the executable files for the database server reside.</td>
</tr>
<tr>
<td>ONCONFIG</td>
<td>The ONCONFIG environment variable specifies the name of the active ONCONFIG configuration file that the database server uses. If the ONCONFIG environment variable is not set, the database server uses values from the file $INFORMIXDIR/etc/onconfig on UNIX or %INFORMIXDIR%\etc\onconfig on Windows NT.</td>
</tr>
<tr>
<td>TERM</td>
<td>The TERM environment variable specifies the type of terminal interface. If the TERM environment variable has not been set on your computer system, you might need the assistance of your UNIX system administrator because this variable is system and terminal dependent.</td>
</tr>
</tbody>
</table>

For detailed information about how these and other environment variables affect the database server, see the Informix Guide to SQL: Reference and the Administrator’s Guide.
Required Environment Variables

Other Useful Environment Variables

The following environment variables let you work with the Global Language Support (GLS) feature:

- CLIENT_LOCALE
- DB_LOCALE
- SERVER_LOCALE

The following environment variables let you work with the client:

- INFORMIXTERM
- TERMININFO ♦

For detailed information about GLS environment variables, see the Informix Guide to GLS Functionality. For information on how setting certain environment variables can affect performance, see your Performance Guide.

Setting UNIX Environment Variables

Set UNIX environment variables in one of the following ways:

- At the system prompt
- In an environment-configuration file ($INFORMIXDIR/etc/informix.rc)
- In your .profile or .login file

You can check the validity of environment variable settings with the chkenv utility.

To override environment variables that have been set automatically, use a private environment-variable file (~/.informix) or assign new values to environment variables individually.

Tip: Informix recommends that you set the environment variables in the appropriate start-up file for your shell.
**Setting Windows NT Environment Variables**

Depending on the application, you can set Windows NT environment variables in one of the following ways:

- At the system prompt
- In the Windows NT registry
- In an environment-configuration file (%INFORMIXDIR%/etc/informix.rc)
- In a batch file
- With a system applet

The installation procedure prepares a %INFORMIXSERVER%.cmd file that sets the environment variables to their correct values. This file is stored in the %INFORMIXDIR% directory. You must execute INFORMIXSERVER%.cmd before you can use any of the command-line utilities.

**Tip:** Informix recommends that you set the environment variables in the appropriate start-up file for your windowing environment.

**Connectivity Information**

The UNIX connectivity files and the Windows NT registry contain information that allows client/server communication. The database server administrator manages connectivity information. The system (or network) administrator manages the network-configuration and network-security files.

**Connectivity Information on UNIX**

On UNIX, the $INFORMIXDIR/etc/sqlhosts file contains connectivity information.
Connectivity Information

*sqlhosts* file

An *sqlhosts* file must exist on each computer that has either an Informix client application or database server. The *sqlhosts* file specifies the location of each database server in your environment and the interface and/or protocol combinations for a connection to that database server. A client application uses information in the *sqlhosts* file to establish a connection to a database server. The *sqlhosts* file must contain an entry for each type of connection to each database server on the network.

For information about the **INFORMIXSQLHOSTS** environment variable, see the *Informix Guide to SQL: Reference*. For detailed information about how to prepare the *sqlhosts* file, refer to the *Administrator’s Guide*.

Network-Configuration Files

In addition to the *sqlhosts* files, TCP/IP connections require entries in the */etc/hosts* and */etc/services* files. IPX/SPX connections also require auxiliary files. For IPX/SPX, the names of the auxiliary files depend on the hardware vendor.

The network-configuration files are described in the *Administrator’s Reference* and in operating-system manuals.

Network-Security Files

Informix database servers follow UNIX security requirements for making connections. Thus, the UNIX system administrator might need to make modifications to the */etc/passwd*, */etc/hosts*, */.rhosts*, and other related files.

The network-security files are described in the *Administrator’s Reference* and in operating-system manuals.
Preparing the ONCONFIG Configuration File

Connectivity Information on Windows NT

In Windows NT environments, the HKEY_LOCAL_MACHINE registry contains the connectivity information. Informix suggests that you not edit the HKEY_LOCAL_MACHINE registry. The database server installation procedure prepares the registry information. For information about how to use a Windows NT interface to modify connectivity information, see the documentation notes that are described in “Documentation Notes, Release Notes, Machine Notes” on page -10 of the introduction.

Preparing the ONCONFIG Configuration File

The ONCONFIG configuration file contains parameters that describe database server environment. You customize an ONCONFIG file to describe a specific environment and can have several ONCONFIG files to describe different environments, such as learning, production, and development.

For information about how to prepare the ONCONFIG file and configure Dynamic Server as well as a summary of configuration parameters, see the Administrator’s Guide. For a discussion of configuration parameters and example configuration files, see the Administrator’s Reference. For information on how configuration parameter settings can affect performance, see your Performance Guide.

Creating an ONCONFIG File on UNIX

To prepare the ONCONFIG configuration file, use a text editor and complete the following steps:

1. Make a copy of $INFORMIXDIR/etc/onconfig.std (the template file) and store the new file in $INFORMIXDIR/etc. You can name the file according to the requirements of your operating system.
2. Edit your new ONCONFIG configuration file and modify the configuration parameters that you decide to change.
3. Set your ONCONFIG environment variable to the name of your new ONCONFIG configuration file.
Creating an ONCONFIG File on Windows NT

On Windows NT, a new configuration file is created and initialized when you install the Informix software. You can also use the Instance Manager to create a new configuration file.

You can also use a text editor to create a new configuration file, as shown in the following steps:

1. Make a copy of the %INFORMIXDIR%\etc\onconfig.std file. Choose an appropriate name for the new file.
2. Edit the new file and change the configuration parameters.
3. Set your ONCONFIG environment variable to the name of your new ONCONFIG file.
4. Update the %INFORMIXSERVER%.cmd file to reflect the new ONCONFIG value.

Allocating Disk Space

Appropriate disk configuration is the most important task for obtaining optimum performance. Disk I/O is the longest portion of the response time for an SQL operation. Dynamic Server offers parallel access to multiple disks on a computer.

Before you allocate the disk space, study the information about disk space in your operating-system administration guide.

Follow these steps to allocate disks for the database server:

1. Configure a raw device or create an unbuffered file for each disk. To achieve better performance, UNIX platforms use raw disk devices and Windows NT platforms use unbuffered NTFS files to bypass the buffering of disk I/O that the operating system normally performs.
2. Create standard device names or filenames. Informix recommends that you use symbolic links to assign abbreviated standard device names for each raw disk device.
Initiating Disk Space

3. Set permissions, ownership, and group for each raw device or unbuffered file.

On Windows NT, a member of the Informix-Admin group must own the files. On UNIX, the owner and group must be informix, and the permissions must be set to read and write for both user and group (but not for others).

For more information, see “The Disk Component” on page 1-8 and “Raw (Unbuffered) Disk Management” on page 1-14.

For information about how to allocate disk space, see the Administrator's Guide.

Initializing Disk Space

In most cases, you initialize disk space just once for a database server, when you bring the database server on-line for the first time. When the database server is in on-line mode, you can connect with it and perform all database activities.

To bring the database server on-line, enter oninit. ♦

On Windows NT, the database server runs as a service. Use the Service control application to bring the database server on-line. ♦

Warning: When you initialize disk space, all of the existing data in the database server is destroyed. Initialize disk space only when you are starting a new database server.

For information about how to initialize disk space, see the Administrator's Guide.
Configuring the SQL API Environment

After you install an SQL API or client product, you can configure the product environment. You must set the following environment variables on the computer on which you have installed the SQL API or client product before you can compile, link, and run the product:

- INFORMIXDIR
- INFORMIXSERVER
- PATH

If your client application uses a nondefault client locale or accesses a database that has a nondefault database locale, you must set the CLIENT_LOCALE and DB_LOCALE environment variables.

For information about environment variables that affect SQL API programs, see the Informix Guide to SQL: Reference. For information about GLS environment variables and locales, see the Informix Guide to GLS Functionality.

Creating a Database

When the database server is on-line, you can connect client applications and SQL APIs to it and begin to create databases. Before you create any database, you must decide whether you want the database to be ANSI compliant. For information, see “ANSI-Compliant Databases” on page 2-16.

For information about ANSI-compliant databases, see the Informix Guide to Database Design and Implementation. For information about ANSI syntax in SQL statements, see the Informix Guide to SQL: Syntax.
Making the Initial Connection to a Database Server

Before you can access information in a database, the client application must connect to the database server environment. To connect to and disconnect from a database server, you can issue SQL statements from DB-Access or an SQL API such as Informix ESQL/C or Informix ODBC Driver. You also can use DB-Access menu and screen options to execute an existing command file or connect to a listed database server.


Creating a Demonstration Database

DB-Access includes scripts that create and populate the demonstration databases that are available with Informix products:

- The stores_demo database is a relational database that works with all Informix database servers. Create it with the dbaccessdemo script.
- The superstores_demo database is an object-relational database that is available only with Dynamic Server. Create it with the dbaccessdemo_ud script.

Informix ESQL/C includes an esqldemo script that copies ESQL/C demonstration programs into your current directory.

For an explanation of how to create and populate the demonstration databases, refer to the DB-Access User’s Manual. For a description of the structure and contents of the stores_demo database, see the Informix Guide to SQL: Reference. For information on how to work with the superstores_demo database, see the Informix Guide to Database Design and Implementation. For information on the permissions required to use the command files, see the Administrator’s Guide.
Obtaining Error Message Information

Informix software products provide text files that contain all the Informix error messages and their corrective actions. To read the error messages in the text file, Informix provides scripts that let you display error messages on the screen (finderr) or print formatted error messages (rofferr).

To read error messages and corrective actions under Windows NT, use the Informix Find Error utility. To display this utility, choose
Start→Programs→Informix from the Task Bar.

For information on the error message files and a detailed description of the finderr and rofferr scripts, see the Introduction to Informix Error Messages in Answers OnLine.

Migrating to Dynamic Server

If you migrate to Dynamic Server, Version 9.2, from an earlier version of the database server, start with the information provided in the Informix Migration Guide.

Performing Standard Administrative Tasks

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.
- Make arrangements for backup management.
  When you plan your data dbspace and logical-log backup schedule, consider the availability of backup devices to manage the data and the availability of 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.
Monitoring Performance

For information about these and other administrative tasks, see the Administrator's Guide and Administrator's Reference. For information about backup management and schedules, see your Backup and Restore Guide. For information about managing backup devices and media, see the Informix Storage Manager Administrator's Guide. For information about the High-Performance Loader, see the Guide to the High-Performance Loader.

Monitoring Performance

Once Dynamic Server is up and running, you might, as database server administrator, be responsible for maintaining the optimum performance of the database server and database applications. To accomplish this goal, you must:

- establish performance goals.
- identify symptoms of performance problems.
- measure the following attributes:
  - throughput
  - response time
  - cost per transaction
  - resource utilization
- evaluate results and make appropriate adjustments.
- tune configurations.

For information on performance-tuning issues and methods that are relevant to daily database server administration and query execution, see your Performance Guide.
Using Operating-System Tools

The database server relies on the operating system of the host computer to provide access to system resources such as the CPU, memory, and various unbuffered disk I/O interfaces and files. Each operating system has its own set of utilities for reporting how system resources are used. Different operating-system implementations might have monitoring utilities with the same name but different options and informational displays.

For information on how to monitor your operating-system resources, consult the reference manual or your system administration guide.

The following table shows typical UNIX operating-system resource-monitor utilities.

<table>
<thead>
<tr>
<th>UNIX Utility</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>vmstat</td>
<td>Displays virtual-memory statistics.</td>
</tr>
<tr>
<td>iostat</td>
<td>Displays I/O utilization statistics.</td>
</tr>
<tr>
<td>sar</td>
<td>Displays a variety of resource statistics.</td>
</tr>
<tr>
<td>ps</td>
<td>Displays active process information.</td>
</tr>
</tbody>
</table>

To capture the status of system resources at regular intervals, use scheduling tools that are available with your host operating system (for example, cron) as part of your performance monitoring system.

The Windows NT operating-system supplies a Performance Monitor (perfmon.exe) that can monitor resources such as processor, memory, cache, threads, and processes. The Performance Monitor also provides charts, alerts, report capabilities, and the ability to save information to log files for later analysis.
Using Dynamic Server Utilities

Dynamic Server provides utilities to capture information about your configuration and performance. It also provides the system-monitoring interface (SMI) for monitoring performance from within an application.

You can use the following utilities to gather information to identify and manage your high-impact activities or to adjust your database server or operating-system configuration:

- DB-Access
- DB/Cockpit
- oncheck
- onlog
- onperf
- onstat

For information about SQL scripts and DB-Access, see the DB-Access User’s Manual. For information about DB/Cockpit, see the DB/Cockpit User’s Manual. For information about oncheck, onlog, onstat and the SMI tables, see the Administrator’s Reference. For information about onperf, see your Performance Guide.
Using the Documentation

In This Chapter .............................................. 4-3
Dynamic Server Users .................................. 4-3
  Dynamic Server Administrators and Operators. 4-4
  Database Administrators .............................. 4-4
  Programmers and Application Developers ....... 4-4
  Database Users ........................................ 4-5
Task-Documentation Matrix ............................. 4-5
The Informix Dynamic Server Documentation Set . 4-16
Client Manuals for This Release ....................... 4-21
In This Chapter

This chapter provides an overview of Dynamic Server users and the tasks that each type of user is likely to perform. It includes a matrix of those tasks and where to find information throughout the documentation set.

This chapter also contains an alphabetical list of the manuals that are provided with the database server software and a list of the documentation for some related Informix client products.

*Tip:* For current information about the products that the database server supports, see the notes described in “Documentation Notes, Release Notes, Machine Notes” on page 10 of the Introduction.

Dynamic Server Users

The following major groups use Dynamic Server:

- Dynamic Server administrators and operators
- Database administrators
- Programmers
  - Application developers
  - Authors of user-defined routines
  - DataBlade module developers
- Database users
Dynamic Server Administrators and Operators

The Dynamic Server administrator is responsible for the installation, maintenance, administration, and operation of the entire database server that might manage many individual databases.

The Dynamic Server operator is responsible for backing up and restoring databases and for carrying out similar routine tasks that are associated with database server administration.

In Figure 4-1 on page 4-6, look for the term ADMIN in the Role column to identify Dynamic Server administrator tasks and the manuals that will help you complete your tasks. Operators will perform a subset of these tasks.

Database Administrators

A database administrator (DBA) is primarily responsible for creating and managing access control for databases. You use SQL statements to grant and revoke privileges to ensure that the correct individuals are able to perform the actions they need to and that untrained or unscrupulous users are kept from performing potentially damaging or inappropriate resource-intensive activities.

In Figure 4-1 on page 4-6, look for the term DBA in the Role column to identify database administrator tasks and the manuals that will help you complete your tasks.

Programmers and Application Developers

Programmers and application developers develop applications, DataBlade modules, and user-defined routines. The Informix Guide to SQL: Reference offers a number of possibilities for data management, multimedia, isolation levels, and so on. The database server integrates information objects such as scanned and digitized images, voice, graphs, facsimiles, and word-processing documents into an SQL-based relational database.

In Figure 4-1 on page 4-6, look for the term DEV in the Role column to identify application developer tasks and the manuals that will help you complete your tasks.
Database Users

Database users access, insert, update, and manage information in databases with SQL, which is often embedded in a client application.

In Figure 4-1 on page 4-6, look for the term USER in the Role column to identify database user tasks and the manuals that will help you complete your tasks.

Task-Documentation Matrix

The Dynamic Server task-documentation matrix in Figure 4-1 on page 4-6 provides a quick reference to the documentation that is available with Dynamic Server. The matrix includes the following three columns:

- **If You Want To.** A task you might want to perform.
- **Manual.** The primary book that contains information to help you perform the task.
- **Role.** The person most likely to perform the task. (ALL indicates that all types of users can benefit from the documentation listed.)

Use the matrix to associate specific tasks with the manuals that will help you perform the tasks. For a list of the Dynamic Server documentation, see “The Informix Dynamic Server Documentation Set” on page 4-16 and “Client Manuals for This Release” on page 4-21.
### Task-Documentation Matrix

**Figure 4-1**

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Install Dynamic Server in different environments</td>
<td><em>Installation Guide for Informix Dynamic Server 2000 on UNIX</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td></td>
<td><em>Installation Guide for Informix Dynamic Server 2000 on Windows NT</em></td>
<td></td>
</tr>
<tr>
<td>Understand Dynamic Server architecture</td>
<td><em>Administrator's Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Learn administrator tasks and tools</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understand client/server communications and multiple residency</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configure the database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Initialize the database server and manage database server operating modes</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identify sources of information for monitoring an Informix database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage virtual processors, shared memory, and disk space</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage database-logging status, logical-log files, and the physical log</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform fast recovery</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform mirroring operations and consistency checking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use high-availability data replication</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understand multiphase commit protocols</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Recover manually from a failed two-phase commit</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(1 of 10)
## Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Use the SMI tables of the <strong>sysmaster</strong> database to monitor the database server</td>
<td><em>Administrator’s Reference</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Use ON-Monitor to monitor the database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use utilities such as <strong>oncheck</strong>, <strong>ondblog</strong>, <strong>onlog</strong>, and <strong>onstat</strong> to perform administrative tasks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use the <strong>onmode</strong> -I option to collect diagnostic information</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Locate information on the configuration parameters</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with the <strong>oninit</strong>, <strong>onparams</strong>, <strong>onspaces</strong>, and <strong>ontape</strong> utilities</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interpret logical-log records and message-log messages</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understand database server disk structures and storage</td>
<td></td>
<td></td>
</tr>
<tr>
<td>See a list of the files that the database server uses</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Back up the logical log</td>
<td><em>Archive and Backup Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Configure, use, and administer ON-Archive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Configure and use the <strong>ontape</strong> utility</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create an archive</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Restore data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Refer to examples to set up, back up, and restore data for ON-Archive</td>
<td><em>ON-Archive Quick Start Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Configure and use the ON-Bar backup and restore system</td>
<td><em>Backup and Restore Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Copy your data and logical logs as insurance against lost or corrupted data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use the <strong>archecker</strong> utility to verify backed-up objects before you use them to restore data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Restore data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connect your Informix database server to storage devices for backup and restore operations</td>
<td><em>Informix Storage Manager Administrator’s Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Manage backup media and storage devices for all ON-Bar backups and restores</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Track the location of all backup data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move backup data through a managed life cycle</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Provide disaster recovery for a database server instance</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(2 of 10)
### Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Migrate to Dynamic Server from an earlier version of an Informix database server</td>
<td><a href="#">Informix Migration Guide</a></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Move data between different physical equipment (computer and storage devices) and different operating systems</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move data between database servers that have different language support</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with these utilities: <a href="#">dbexport</a>, <a href="#">dbimport</a>, <a href="#">dbload</a>, <a href="#">dbschema</a>, <a href="#">onload</a>, <a href="#">onunload</a>, <a href="#">onmode -b</a></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Revert from Dynamic Server to an earlier version of an Informix database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitor system resources (CPU, memory, disk) that are critical to performance</td>
<td><a href="#">Performance Guide</a></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Identify the database activities that affect critical resources</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identify and monitor queries that are critical to performance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Improve the performance of a query</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use utilities such as <a href="#">onperf</a> for performance monitoring and tuning</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Eliminate database server performance bottlenecks</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Balance the load on system resources</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjust the database server configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Adjust the arrangement of data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Allocate resources for DSS queries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create and manage indexes to speed up retrieval of data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use secondary access methods such as B-trees</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Control the placement and size of tables and table extents</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Control the placement and size of fragments (not in Workgroup and Developer Editions)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Manage resources for parallel database queries (PDQ) (not in Workgroup and Developer Editions)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Monitor and track locking</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn when and how to use optimizer directives</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If You Want To:</td>
<td>Manual</td>
<td>Role</td>
</tr>
<tr>
<td>------------------------------------------------------------------------------</td>
<td>--------------------------------------------------</td>
<td>-------</td>
</tr>
<tr>
<td>Use the <code>ipload</code> and <code>onload</code> utilities to load or unload large quantities of data to or from an Informix database</td>
<td><em>Guide to the High-Performance Loader</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Use the graphical user interface that prepares the <code>onload</code> database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Move data to a different computer or configuration</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Alter the schema of a table</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Detect unusual user actions and unwanted activities and identify the perpetrators</td>
<td><em>Trusted Facility Manual</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Detect unauthorized access attempts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assess potential security compromises</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use the secure-auditing utilities (onaudit, onshowaudit) to set up, administer, and interpret audit trails</td>
<td></td>
<td></td>
</tr>
<tr>
<td>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)</td>
<td><em>Guide to the Optical Subsystem</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Use the set of SQL statements that support the storage and retrieval of data to and from the optical storage subsystem</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use the Informix SNMP subagent to extract information from an Informix database server and pass that information to a network manager</td>
<td><em>Informix SNMP Subagent Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Work with the R-tree secondary access method</td>
<td><em>Informix R-tree Index User’s Guide</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Design, define, monitor, and control your Enterprise Replication system</td>
<td><em>Guide to Informix Enterprise Replication</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Gather information about database server status and activity with the DB/Cockpit utility</td>
<td><em>DB/Cockpit User’s Manual</em></td>
<td>ADMIN</td>
</tr>
<tr>
<td>Use a graphical interface to monitor database servers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with the tables in the <code>sysmaster</code> database</td>
<td><em>Administrator’s Reference</em></td>
<td>DBA</td>
</tr>
<tr>
<td>Understand the impact of database design and use on performance</td>
<td><em>Performance Guide</em></td>
<td>DBA</td>
</tr>
</tbody>
</table>

(4 of 10)
### Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Design data models and implement relational, object-relational, and dimensional databases for Informix database servers</td>
<td>Informix Guide to Database Design and Implementation</td>
<td>DBA</td>
</tr>
<tr>
<td>Understand data-warehousing concepts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Identify and define data and data objects for Informix databases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use supported data types and SQL to implement a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extend a database with complex data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Extend a database with user-defined casts</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understand type and table inheritance</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grant and limit access to a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Define a fragmentation strategy or distribution schema</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform data-definition tasks such as specifying the data types for table columns</td>
<td>DB-Access User’s Manual</td>
<td>DBA</td>
</tr>
<tr>
<td>Perform data-management tasks such as storing, viewing, and changing table data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connect to one or more database servers and transfer data</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verify database server status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create databases and run ad hoc queries</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execute SQL statements and SPL routines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display system catalog tables and the Information Schema</td>
<td></td>
<td></td>
</tr>
<tr>
<td>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)</td>
<td>Guide to the Optical Subsystem</td>
<td>DBA</td>
</tr>
<tr>
<td>Use the set of SQL statements that support the storage and retrieval of data to and from the optical storage subsystem</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Generate Information Schema views</td>
<td>Informix Guide to SQL: Reference</td>
<td>DBA</td>
</tr>
<tr>
<td>Use the system catalog tables to track objects</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assign data types to columns</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create databases and manage access</td>
<td>Informix Guide to SQL: Syntax</td>
<td>DBA</td>
</tr>
<tr>
<td>Compose correct SQL statements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write procedures with SPL and store them in a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>If You Want To:</td>
<td>Manual</td>
<td>Role</td>
</tr>
<tr>
<td>----------------</td>
<td>--------</td>
<td>------</td>
</tr>
<tr>
<td>Query and modify data in a relational database</td>
<td>Informix Guide to SQL: Tutorial</td>
<td>DBA</td>
</tr>
<tr>
<td>Work with user-defined and system-defined casts on extended data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create and use SQL statements and SPL routines</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Execute and debug SQL statements and SPL routines</td>
<td>DB-Access User’s Manual</td>
<td>DEV</td>
</tr>
<tr>
<td>Create databases, run ad hoc queries, test database applications that you intend to store for use in a production environment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Access, modify, and retrieve information form Informix database servers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Verify database server status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connect to one or more databases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display system catalog tables and the Information Schema</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Assign data types to columns</td>
<td>Informix Guide to Database Design and Implementation</td>
<td>DEV</td>
</tr>
<tr>
<td>Use supported data types in programs</td>
<td>Informix Guide to SQL: Reference</td>
<td></td>
</tr>
<tr>
<td>Use the system catalog tables to track objects</td>
<td>Informix Guide to SQL: Syntax</td>
<td></td>
</tr>
<tr>
<td>Find a description of the tables in the stores_demo or superstores_demo database</td>
<td>Informix Guide to SQL: Tutorial</td>
<td></td>
</tr>
<tr>
<td>Compose correct SQL statements</td>
<td>Informix Guide to SQL: Syntax</td>
<td>DEV</td>
</tr>
<tr>
<td>Write procedures with SPL and store them in a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform a parallel database query (PDQ)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use a primary access method</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(6 of 10)
### Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Compose SELECT statements</td>
<td><em>Informix Guide to SQL: Tutorial</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Query and modify data in a relational database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use embedded SQL in programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Program in a multiuser environment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create and use routines with SPL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create and use triggers</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with user-defined and system-defined casts on extended data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use Informix ODBC Driver to access Informix relational databases with SQL</td>
<td><em>Informix ODBC Driver Programmer’s Manual</em></td>
<td>DEV</td>
</tr>
<tr>
<td>and interact with an Informix database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create custom applications with the Informix ODBC API</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Embed SQL statements directly into C programs</td>
<td><em>Informix ESQL/C Programmer’s Manual</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Create new data types and user-defined routines</td>
<td><em>Extending Informix Dynamic Server 2000</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Create new data types and user-defined routines using Java</td>
<td><em>Creating UDRs in Java</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Use the GLS features that let Informix SQL APIs and database servers</td>
<td><em>Informix Guide to GLS Functionality</em></td>
<td>DEV</td>
</tr>
<tr>
<td>handle different languages, cultural conventions, and code sets</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internationalize DataBlade modules and ESQL/C programs with Informix GLS</td>
<td><em>Informix GLS Programmer’s Manual</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Work with the TP/XA library in an X/Open distributed transaction-processing</td>
<td><em>TP/XA Programmer’s Manual</em></td>
<td>DEV</td>
</tr>
<tr>
<td>(DTP) environment</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Develop applications for a third-party transaction manager and an Informix</td>
<td></td>
<td></td>
</tr>
<tr>
<td>database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn how GLS affects database server migration</td>
<td><em>Informix Migration Guide</em></td>
<td>DEV</td>
</tr>
</tbody>
</table>

(7 of 10)
### Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Define new data types or extend the functionality of existing data types for a Dynamic Server database</td>
<td><em>Extending Informix Dynamic Server 2000</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Extend operations on data types, create new casts, extend operator classes for secondary access methods, and write opaque data types for your database or DataBlade programs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create application-specific SPL or external routines for application end-users</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create and register a user-defined routine (UDR) to invoke within an SQL statement or another routine</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use DataBlade API functions in DataBlade modules to develop server and client applications that access data stored in a Dynamic Server database</td>
<td><em>DataBlade API Programmer’s Manual</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Write server routines and client LIBMI applications that use smart large objects and complex and extended data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use Java to create client applications or applets that run against Dynamic Server</td>
<td><em>Informix JDBC Driver Programmer’s Guide</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Install and load the Informix JDBC Driver</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use standard JDBC to connect to a database or database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use standard JDBC to send queries, retrieve results, get database and column metadata, and handle errors</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn how standard Java data types map to Informix database server data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use the object-oriented C++ programming language to create database client applications for Informix database servers</td>
<td><em>INFORMIX-Object Interface for C++ Programmer’s Guide</em></td>
<td>DEV</td>
</tr>
<tr>
<td>Use Object Interface for C++ to create value objects that let C++ client applications support DataBlade module data types</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(8 of 10)
### Task-Documentation Matrix

<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Work with the Object Interface for Java, a library of Java classes and interfaces</td>
<td><a href="#">INFORMIX-Object Interface for Java Programmer’s Guide</a></td>
<td>DEV</td>
</tr>
<tr>
<td>Create enterprise client/server applications that connect directly to an Informix database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create applets that run in remote HTML browsers that connect back to an Informix database server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Create Object Interface for Java objects to handle new data types in DataBlade modules that work with Dynamic Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with the R-tree secondary access method</td>
<td><a href="#">Informix R-tree Index User’s Guide</a></td>
<td>DEV</td>
</tr>
<tr>
<td>Develop a primary or secondary access method to obtain data that is external to the database server</td>
<td><a href="#">Virtual-Index Interface Programmer’s Manual</a></td>
<td>DEV</td>
</tr>
<tr>
<td></td>
<td><a href="#">Virtual-Table Interface Programmer’s Manual</a></td>
<td></td>
</tr>
<tr>
<td>Set environment variables</td>
<td><a href="#">Informix Guide to SQL: Reference</a></td>
<td>USER</td>
</tr>
<tr>
<td>Use supported data types</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Understand the Information Schema</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn the structure and contents of the tables in the stores_demo database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Learn the structure of the superstores_demo database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compose correct SQL statements</td>
<td><a href="#">Informix Guide to SQL: Syntax</a></td>
<td>USER</td>
</tr>
<tr>
<td>Learn the categories of SQL statements</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Write procedures with SPL and store them in a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Compose basic and advanced SELECT statements</td>
<td><a href="#">Informix Guide to SQL: Tutorial</a></td>
<td>USER</td>
</tr>
<tr>
<td>Query and modify data in a relational database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Use embedded SQL in programs</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

(9 of 10)
<table>
<thead>
<tr>
<th>If You Want To:</th>
<th>Manual</th>
<th>Role</th>
</tr>
</thead>
<tbody>
<tr>
<td>Invoke the DB-Access utility</td>
<td><strong>DB-Access User’s Manual</strong></td>
<td>USER</td>
</tr>
<tr>
<td>Use menus, screens, SQL statements, and SPL routines to view, access, retrieve, store, and modify data in a database</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Connect to or create one or more databases and transfer data between a database and external text files</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Display information about a database and verify database server status</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Perform ad hoc queries that you execute once or infrequently</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Practice the statements and examples provided with the demonstration databases</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Install client products that the database server supports, such as Informix Connect and products in the Informix Client Software Developer’s Kit</td>
<td><strong>Informix Client Products Installation Guide for UNIX</strong>&lt;br&gt;<strong>Informix Client Products Installation Guide for Microsoft Windows Environments</strong></td>
<td>ALL</td>
</tr>
<tr>
<td>Set the appropriate environment variables for your Informix product</td>
<td><strong>Informix Guide to SQL: Reference</strong></td>
<td>ALL</td>
</tr>
<tr>
<td>Acquaint yourself with terms used in Informix database server manuals</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Work with relational (<strong>stores_demo</strong>) and object-relational (<strong>superstores_demo</strong>) demonstration databases that Informix products provide</td>
<td><strong>DB-Access User’s Manual</strong></td>
<td>ALL</td>
</tr>
<tr>
<td>Find corrective actions to error messages</td>
<td><strong>Informix Error Messages</strong></td>
<td>ALL</td>
</tr>
</tbody>
</table>

(10 of 10)
The Informix Dynamic Server Documentation Set

This section summarizes the documentation that is available with Dynamic Server. Figure 4-2 lists manuals alphabetically.

<table>
<thead>
<tr>
<th>Book Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><em>Administrator’s Guide for Informix Dynamic Server 2000</em></td>
<td>This user guide for system and database server administrators discusses the concepts, procedures, and syntax for managing Dynamic Server. It is intended to help you understand, configure, and use the database server.</td>
</tr>
<tr>
<td><em>Informix Administrator’s Reference</em></td>
<td>This reference manual for system and database server administrators provides the syntax and reference information for the Dynamic Server utilities, <code>sysmaster</code> tables and SMI, disk structures, and configuration parameters. It also discusses ON-Monitor options, files that the database server uses, message-log messages, trapping errors, and how to read logical log records.</td>
</tr>
<tr>
<td><em>Archive and Backup Guide for Informix Dynamic Server 2000</em></td>
<td>This guide describes the components, features, and tasks that make up the recovery system provided with the database server. It provides detailed information on the ON-Archive and <code>ontape</code> facilities.</td>
</tr>
<tr>
<td><em>Informix Backup and Restore Guide</em></td>
<td>This user guide and reference manual explains the concepts and methods required to back up and restore data with the ON-Bar backup and restore system. It includes information on the <code>archecker</code> utility.</td>
</tr>
<tr>
<td><em>Creating UDRs in Java</em></td>
<td>This guide explains how to use Java to extend existing data types and define new data types for a database on Dynamic Server.</td>
</tr>
</tbody>
</table>

(1 of 5)
<table>
<thead>
<tr>
<th>Book Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DB-Access User’s Manual</strong></td>
<td>This guide describes how to use the DB-Access utility to access, modify, and retrieve information from Informix database servers. It provides information on how to use the interface for SQL statements and SPL routines to perform data-definition and data-management tasks. The guide includes the SQL command files and UDRs for the demonstration databases that are shipped with Informix database servers.</td>
</tr>
<tr>
<td><strong>DB/Cockpit User’s Manual</strong></td>
<td>This manual discusses the features that make up the DB/Cockpit utility, a graphical tool designed for monitoring Dynamic Server and maintaining its integrity. It includes information about the onprobe and oncockpit components.</td>
</tr>
<tr>
<td><strong>Extending Informix Dynamic Server 2000</strong></td>
<td>This guide explains how to extend existing data types and define new data types for a database 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, and write opaque data types. It also explains how to define your own functions and procedures for use in a Dynamic Server database. It describes common considerations for SPL routines and external routines and tells how to invoke user-defined routines (UDRs).</td>
</tr>
<tr>
<td><strong>Getting Started with Informix Dynamic Server 2000</strong></td>
<td>This guide provides an overview of the relational database management system, summarizes important features of Informix products, and provides information to help you use the documentation that is included with Dynamic Server products.</td>
</tr>
<tr>
<td><strong>Guide to Informix Enterprise Replication</strong></td>
<td>This guide contains information to help you understand the concepts of data replication, design your own Enterprise Replication client/server application system, install Enterprise Replication, and administer and manage data replication throughout your open-systems enterprise.</td>
</tr>
<tr>
<td><strong>Guide to the High-Performance Loader</strong></td>
<td>This guide describes how to use the High-Performance Loader (HPL) to efficiently load an unload large quantities of data to or from an Informix database.</td>
</tr>
<tr>
<td>Book Title</td>
<td>Description</td>
</tr>
<tr>
<td>----------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Guide to the Optical Subsystem</strong></td>
<td>This guide describes how to use the Optical Subsystem configuration of Dynamic Server to store TEXT and BYTE data (simple large objects) on optical platters.</td>
</tr>
<tr>
<td><strong>Informix Error Messages</strong></td>
<td>This Answers OnLine product provides a complete list of Informix-specific error messages and describes their corrective actions for current Informix products and earlier Informix products that are still supported. It also describes how to use the <code>finderr</code> and <code>rofferr</code> scripts to read error messages on-line. Use the UNIX <code>finderr</code> and <code>rofferr</code> utilities or the Windows NT Informix Find Error utility to locate the latest information on error messages.</td>
</tr>
<tr>
<td><strong>Informix Guide to Database Design and Implementation</strong></td>
<td>This guide documents how to design and implement databases for Informix database servers. It describes the fundamental ideas and terminology for planning, implementing, and using a relational, object-relational, or dimensional database management system. The guide discusses tasks usually performed by the DBA and addresses data definition with Informix databases. It describes how to design different data models and use the supported data types and SQL to implement a database.</td>
</tr>
<tr>
<td><strong>Informix Guide to GLS Functionality</strong></td>
<td>This manual describes the Global Language Support (GLS) feature that is available in Informix products. The GLS feature allows Informix client products and Informix database servers to handle different languages, cultural conventions, and code sets. This manual describes only the language-related topics that are unique to GLS.</td>
</tr>
<tr>
<td><strong>Informix Guide to SQL: Reference</strong></td>
<td>This manual describes the Informix system catalog tables, the data types supported by Informix database servers, and Informix and other common environment variables that you might need to set. It also contains a description of the structure of the <code>stores_demo</code> and <code>superstores_demo</code> databases and a glossary of terms used in Informix manuals.</td>
</tr>
<tr>
<td><strong>Informix Guide to SQL: Syntax</strong></td>
<td>This manual contains the complete syntax descriptions for Structured Query Language (SQL) and Stored Procedure Language (SPL) statements.</td>
</tr>
<tr>
<td>Book Title</td>
<td>Description</td>
</tr>
<tr>
<td>------------------------------------------------</td>
<td>-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Informix Guide to SQL: Tutorial</strong></td>
<td>This tutorial provides instructions for using basic and advanced 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, and create and use triggers. The guide also discusses how to use implicit and explicit user-defined casts and system-defined casts for extended data types.</td>
</tr>
<tr>
<td><strong>Informix Migration Guide</strong></td>
<td>This manual describes the tasks that you perform when you move data from one location to another and when you migrate existing Informix databases to various Informix database servers. It discusses such database server utilities as <code>dbexport</code>, <code>dbimport</code>, <code>dbload</code>, <code>dbschema</code>, <code>onload</code>, <code>onunload</code>, and <code>onmode -b</code>.</td>
</tr>
<tr>
<td><strong>Informix R-tree Index User's Guide</strong></td>
<td>This guide contains detailed information about the R-tree index structure and secondary access method.</td>
</tr>
<tr>
<td><strong>Informix SNMP Subagent Guide</strong></td>
<td>This manual introduces Simple Network Management Protocol (SNMP) and describes the subagent that provides information about Informix database servers to network-management tools. The guide also documents the Management Information Bases (MIBs) that specify the information that the <code>onSNMP</code> program provides to the network-management tools. It includes a glossary of terms used in the guide.</td>
</tr>
<tr>
<td><strong>Informix Storage Manager Administrator's Guide</strong></td>
<td>This guide describes how to connect your Informix database server to storage volumes on storage devices for ON-Bar backup and restore operations, and how to manage backup media with the Informix Storage Manager.</td>
</tr>
<tr>
<td><strong>Installation Guide for Informix Dynamic Server 2000 on UNIX</strong></td>
<td>This guide contains instructions for installing Dynamic Server on computers that run the UNIX operating system. It also describes common installation problems and indicates how to solve them.</td>
</tr>
<tr>
<td><strong>Installation Guide for Informix Dynamic Server 2000 on Windows NT</strong></td>
<td>This guide contains instructions for installing Dynamic Server on computers that run on Windows NT. It also discusses how to start, stop, and remove the database server and lists common installation error messages and warnings.</td>
</tr>
</tbody>
</table>
## The Informix Dynamic Server Documentation Set

<table>
<thead>
<tr>
<th>Book Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>ON-Archive Quick Start Guide</td>
<td>This guide presents a complete example of setting up, backing up, and restoring data for a system. It introduces you to the concepts, terminology, and basic tasks involved in using ON-Archive as your recovery system. It includes a section on troubleshooting common problems and a scripted sample session.</td>
</tr>
<tr>
<td>Performance Guide for Informix Dynamic Server 2000</td>
<td>This manual explains how to configure and operate Dynamic Server to improve overall system performance as well as the performance of SQL queries.</td>
</tr>
<tr>
<td>Trusted Facility Manual for Informix Dynamic Server 2000</td>
<td>This guide describes the secure auditing facility and includes information on how to set up and administer audit trails, extract and interpret audit records, and use utilities and SQL statements for audit analysis.</td>
</tr>
<tr>
<td>Virtual-Index Interface Programmer’s Manual *</td>
<td>This manual defines a secondary access method and explains how to create one with the Virtual-Index Interface Application Program Interface (VII API). A virtual index accesses data from a source that the database server does not manage or specific data inside of large objects. The manual describes the syntax that the database server uses to invoke your C-language library, as well as the supplied API function calls and data structures. For information on how to obtain this document, contact your Informix representative.</td>
</tr>
<tr>
<td>Virtual-Table Interface Programmer’s Manual *</td>
<td>This manual defines a primary access method and explains how to create one with the Virtual-Table Interface Application Program Interface (VTI API). A virtual table is dynamically created from a source that the database server does not manage or specific data inside of large objects. The manual describes the syntax that the database server uses to invoke your C-language library, as well as the supplied API function calls and data structures. For information on how to obtain this document, contact your Informix representative.</td>
</tr>
</tbody>
</table>

* Important: The Virtual-Table Interface Programmer’s Manual and the Virtual-Index Interface Programmer’s Manual are specifically for customers and partners developing alternative access methods for Dynamic Server. Informix continues to modify the interfaces described in these manuals. Customers and partners who use this interface should work with an Informix representative to ensure that they continue to receive the latest information and that they are prepared to change their access method.
Client Manuals for This Release

This section summarizes the Informix client and Client SDK manuals that you can use when you work with Dynamic Server, Version 9.2. Figure 4-3 lists manuals for Informix client products alphabetically.

<table>
<thead>
<tr>
<th>Book Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>DataBlade API Programmer’s Manual</strong></td>
<td>This manual contains information about the DataBlade API, the C-language application programming interface that is provided with Dynamic Server. It describes how to use the DataBlade API to develop client and server applications that access data stored in a Dynamic Server database.</td>
</tr>
<tr>
<td><strong>Informix ODBC Driver Programmer’s Manual</strong></td>
<td>This user guide and reference manual discusses the features that make up the Informix implementation of the Microsoft Open Database Connectivity (ODBC) interface. It explains how to use the Informix ODBC libraries and driver manager to access Informix databases, manipulate the data in your program, interact with the database server, and check for errors. It includes lists of error codes and supported data types, and a comparison of Informix ODBC Driver and embedded SQL.</td>
</tr>
<tr>
<td><strong>Informix Client Products Installation Guide for Microsoft Windows Environments</strong></td>
<td>This guide describes how to install Informix client products on computers that run in Windows environments and discusses how to solve common installation problems.</td>
</tr>
<tr>
<td><strong>Informix Client Products Installation Guide for UNIX</strong></td>
<td>This guide explains how to install Informix client products such as Informix Connect and the Informix Client Software Developer’s Kit on computers that run the UNIX operating system. It also discusses how to solve common installation problems.</td>
</tr>
<tr>
<td><strong>Informix ESQL/C Programmer’s Manual</strong></td>
<td>This manual explains how to use Informix ESQL/C, the Informix implementation of embedded Structured Query Language for C, to create client applications with database-management capabilities. It is a complete guide to the features of ESQL/C that let programmers interact with the database server, access databases, manipulate the data in programs, and check for errors.</td>
</tr>
</tbody>
</table>
### Client Manuals for This Release

<table>
<thead>
<tr>
<th>Book Title</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Informix GLS Programmer’s Manual</strong></td>
<td>This manual describes the Informix GLS application-programming interface that is available in Informix products. which gives ESQL/C programmers and DataBlade module developers the ability to write or change programs to work with different locales.</td>
</tr>
<tr>
<td><strong>Informix JDBC Driver Programmer’s Guide</strong></td>
<td>This guide provides useful information for programmers who create Java applications or applets that use JDBC to connect to Informix databases with the Informix JDBC Driver, type 4. It describes how to install and load the driver, use standard JDBC, get metadata from a database, and map between standard Java and Informix data types. It includes troubleshooting advice and tips.</td>
</tr>
<tr>
<td><strong>INFORMIX-Object Interface for C++ Programmer’s Guide</strong></td>
<td>This guide describes how C++ and DataBlade developers can develop Informix client applications with the object-oriented C++ programming language. It describes how the INFORMIX-Object Interface for C++ encapsulates Informix database server features into a class hierarchy and extensible object library.</td>
</tr>
<tr>
<td><strong>TP/XA Programmer’s Manual</strong></td>
<td>This guide describes how to use the TP/XA library of Informix ESQL/C to program in an X/Open DTP environment when data is distributed across multivendor or non-Informix databases.</td>
</tr>
</tbody>
</table>
Index

A
Access method
definition 2-21
primary 2-23
secondary 2-21
Administrative tasks
role separation for 2-29
standard, listed 3-21
Aggregate, user-defined 2-36
Allocating disk space 3-17
ANSI compliance Intro-13, 2-16
Architecture 1-5
client/server 1-11
distributed databases 1-22
dynamic thread allocation 1-15
fault tolerance and high
availability 1-17
high-performance 1-14
memory management 1-15
parallelization 1-17
raw disk management 1-14
scalability 1-13
Archive, and backups 1-18
Asynchronous data replication 2-7
Auditing 1-24

Bounding box 2-23
B-tree index 2-22
Buffered logging, description
of 2-28
BYTE data type 2-32

C
Cache, shared-statement 2-26
chkenv utility 3-13
Client
configuring environment for 3-19
description of 1-10
improving performance and
availability 2-7
supported products 3-4
Client application
description of 2-4
types of 2-24
Client SDK products
DataBlade API 3-7
Informix JDBC Driver 3-9
INFORMIX-CLI 3-7
INFORMIX-ESQL/C 3-8
INFORMIX-GLS 3-9
supported 3-6
TP/XA 3-10
Client/server
architecture, description of 1-10
configurations 1-13
definition of 1-10
CLIENT_LOCALE environment
variable 3-19
Code set, ISO 8859-1 Intro-4
<table>
<thead>
<tr>
<th>Collection type</th>
<th>Column data types</th>
<th>CPU, relationship to virtual processor</th>
<th>Commit</th>
<th>Onload for high-performance loader</th>
<th>Cross-reference icons</th>
</tr>
</thead>
<tbody>
<tr>
<td>2-31</td>
<td>2-6</td>
<td>1-9</td>
<td>Intro-7</td>
<td>2-10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>information on</td>
<td></td>
<td></td>
<td>3-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2-29</td>
<td></td>
<td></td>
<td>stores_demo 3-20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>replication of</td>
<td></td>
<td></td>
<td>superstores_demo 3-20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2-8</td>
<td></td>
<td></td>
<td>supported 2-16</td>
<td></td>
</tr>
<tr>
<td></td>
<td>retrieving from</td>
<td></td>
<td></td>
<td>Database management system</td>
<td></td>
</tr>
<tr>
<td></td>
<td>a database 2-3</td>
<td></td>
<td></td>
<td>object-relational 1-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>storage of large</td>
<td></td>
<td></td>
<td>overview of 1-3</td>
<td></td>
</tr>
<tr>
<td></td>
<td>objects 2-32</td>
<td></td>
<td></td>
<td>relational 1-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with built-in</td>
<td></td>
<td></td>
<td>Database server administrator</td>
<td></td>
</tr>
<tr>
<td></td>
<td>data type 2-22</td>
<td></td>
<td></td>
<td>description of 4-4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Comment icons</td>
<td></td>
<td></td>
<td>client/server architecture 1-10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Intro-7</td>
<td></td>
<td></td>
<td>configuring 3-11</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Commit</td>
<td></td>
<td></td>
<td>connecting with an SQL API 3-20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>heterogeneous 1-23</td>
<td></td>
<td></td>
<td>data storage 1-9</td>
<td></td>
</tr>
<tr>
<td></td>
<td>two-phase 1-23</td>
<td></td>
<td></td>
<td>description of 2-4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Communication.</td>
<td></td>
<td></td>
<td>distributed database</td>
<td></td>
</tr>
<tr>
<td></td>
<td>See Connectivity.</td>
<td></td>
<td></td>
<td>processing 2-17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compatibility of</td>
<td></td>
<td></td>
<td>extending 2-20, 2-30</td>
<td></td>
</tr>
<tr>
<td></td>
<td>client and server programs 3-10</td>
<td></td>
<td></td>
<td>fault-tolerant features 1-17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Complex data</td>
<td></td>
<td></td>
<td>high performance of 1-14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>types 2-31</td>
<td></td>
<td></td>
<td>initial connection to 3-20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Compliance</td>
<td></td>
<td></td>
<td>ON-Bar backup and restore 2-14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>with industry</td>
<td></td>
<td></td>
<td>parallel database query</td>
<td></td>
</tr>
<tr>
<td></td>
<td>standards Intro-13</td>
<td></td>
<td></td>
<td>feature 1-17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Configuration</td>
<td></td>
<td></td>
<td>raw-disk management 1-14</td>
<td></td>
</tr>
<tr>
<td></td>
<td>database server</td>
<td></td>
<td></td>
<td>scalability of 1-13</td>
<td></td>
</tr>
<tr>
<td></td>
<td>environment 3-11</td>
<td></td>
<td></td>
<td>security 1-24</td>
<td></td>
</tr>
<tr>
<td></td>
<td>files for network 3-15</td>
<td></td>
<td></td>
<td>system catalog for 2-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ONCONFIG file 3-12</td>
<td></td>
<td></td>
<td>utilities, listed 2-10</td>
<td></td>
</tr>
<tr>
<td></td>
<td>requirements 3-11</td>
<td></td>
<td></td>
<td>Database server feature</td>
<td></td>
</tr>
<tr>
<td></td>
<td>SQL API environment 3-19</td>
<td></td>
<td></td>
<td>isolation level 2-28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Connection</td>
<td></td>
<td></td>
<td>locking 2-27</td>
<td></td>
</tr>
<tr>
<td></td>
<td>database vs.</td>
<td></td>
<td></td>
<td>role separation 2-29</td>
<td></td>
</tr>
<tr>
<td></td>
<td>network 1-11</td>
<td></td>
<td></td>
<td>rolling back transactions 2-28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>defined 1-11</td>
<td></td>
<td></td>
<td>transaction logging 2-28</td>
<td></td>
</tr>
<tr>
<td></td>
<td>multiplexed 1-11</td>
<td></td>
<td></td>
<td>DataBlade modules</td>
<td></td>
</tr>
<tr>
<td></td>
<td>network 1-12</td>
<td></td>
<td></td>
<td>described 2-35</td>
<td></td>
</tr>
<tr>
<td></td>
<td>network, description of 1-12</td>
<td></td>
<td></td>
<td>mentioned 2-20, 2-33</td>
<td></td>
</tr>
<tr>
<td></td>
<td>to database</td>
<td></td>
<td></td>
<td>DB-Access utility Intro-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>server, initial 3-20</td>
<td></td>
<td></td>
<td>creating the demonstration</td>
<td></td>
</tr>
<tr>
<td>Connectivity</td>
<td>client/server 1-10</td>
<td></td>
<td></td>
<td>database 3-20</td>
<td></td>
</tr>
<tr>
<td></td>
<td>files, overview of 3-14</td>
<td></td>
<td></td>
<td>described 3-5</td>
<td></td>
</tr>
<tr>
<td></td>
<td>information on UNIX 3-14</td>
<td></td>
<td></td>
<td>documentation 4-15, 4-17</td>
<td></td>
</tr>
<tr>
<td></td>
<td>information on Windows NT 3-16</td>
<td></td>
<td></td>
<td>dbexport utility 2-10, 2-15, 4-8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>ODBC standard 3-7</td>
<td></td>
<td></td>
<td>dbimport utility 2-10, 4-8</td>
<td></td>
</tr>
<tr>
<td></td>
<td>sqhosts file 3-15</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>supported protocols 1-11</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Contact information Intro-13</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Conventions,</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>documentation Intro-5</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cooked disk space, description of 1-9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Cost-based optimizer 2-9</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data manipulation with DB-Access 3-5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>storage in data warehouse environment 2-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data mart, described 2-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data migration tools, and database server recovery 2-15</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data recovery mechanisms</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>data replication 1-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>mirroring 1-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data replication advantages of 2-7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>defined 2-7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>description of 1-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>enterprise 2-8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>high-availability 2-8</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>two types 2-7</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data storage, for database servers 1-9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data types 2-31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BLOB, mentioned 2-33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>BYTE, mentioned 2-32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>CLOB, mentioned 2-33</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>complex, row type 2-31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>distinct 2-31</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>TEXT, mentioned 2-32</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>user defined 2-30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Data warehouse, described 2-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>Database ANSI compliant 2-16</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>creating 3-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>demonstration 3-20</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>dimensional 2-19</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>distributed 2-17</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>distributed across multiple servers 2-18</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>object-relational 2-20</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
dbload utility 2-10, 4-8
DBMS. See Database management system.
dbschema utility 2-11, 2-29, 4-8
Dbspaces
  archiving 2-12
  restoring 2-12
DB/locale environment variable 3-19
Decision-support query applications 2-25
Default locale Intro-4
Definitions
  data mart 2-19
  data warehouse 2-19
Demonstration database
  creating 3-20
  overview of 3-20
Demonstration databases Intro-5
Dependencies, software Intro-4
Disk access, for data storage 1-8
Disk I/O, role of shared memory in reducing 1-7
Disk space
  allocating 3-17
  cooked 1-9
  initializing 3-18
  raw 1-9
  types of physical units 1-8
Distinct type, description of 2-31
Distributed database
  advantages of 2-18
  and transactions 1-22
  description of 2-17
  disadvantages of 2-17
  multiple vendor servers 2-18
Distributed transaction-processing environment 2-18
Documentation
  for other Informix products, listed 4-21
  for specific tasks 4-5
  listed 4-16
Documentation notes Intro-10
Documentation set, listed 4-16
Documentation, types of
  documentation notes Intro-10
  error message files Intro-9
  machine notes Intro-10
  on-line help Intro-9
  on-line manuals Intro-9
  printed manuals Intro-9
  related reading Intro-11
  release notes Intro-10
Driver manager, described 3-8
DSA. See Dynamic scalable architecture.
DSS applications 2-25
DTP environment. See Distributed transaction-processing environment.
Dynamic scalable architecture
  advantages 1-6
  description of 1-6
  extending 1-5
  performance advantages 1-13
  virtual processor component 1-9

E
Enterprise Replication,
described 2-8
Environment variable
  CLIENT_LOCALE 3-13, 3-19
  DB_LOCAL 3-19
  for client 3-13
  how to set 3-13, 3-14
  INFORMIXDIR 3-11, 3-12, 3-19
  INFORMIXSERVER 3-12, 3-19
  INFORMIXTERM 3-13
  ONCONFIG 3-12
  PATH 3-12, 3-19
  required 3-12
  SERVER_LOCALE 3-13
  setting UNIX 3-13
  TERM 3-12
  TERMINFO 3-13
  useful 3-13
Environment variables
  Intro-6
  setting Windows NT 3-14
  en_us.8859-1 locale Intro-4
Error message files Intro-9
Error messages, getting information 3-21
Expression-based
  fragmentation 1-16
Extending the database server 2-20, 2-30
External routine 2-34
External spaces 2-23
Extspaces 2-23

F
Failure, system, and ON-Bar 2-14
Fast recovery, description of 1-18
Fault tolerance
  data replication 1-20, 2-8
  fast recovery 1-18
  mirroring 1-19
Feature icons Intro-8
Features
  data management 2-7
  data recovery 2-12
  data restore 2-12
  other database server 2-26
  RDBMS 2-3
Features of this product, new Intro-5
Field, description of 2-31
File
  autoexec.bat 3-14
  back up with ontape 2-13
  configuration 3-13
  connectivity 3-14
  cooked 1-9
  create with dbschema 2-11
  database server security 1-24
  display contents with onlog 2-11
  error message 3-21
  for dbexport 2-10
  hosts 3-15
  HPL data 2-10
  index.htm 3-9
  logical log 1-18
  logical-log 2-12
  mirroring 1-19
  network configuration 3-15
  network security 3-15
  ONCONFIG 3-12
  ONCONFIG, preparing 3-16
  onconfig.std 3-16
  operating system 1-9
passwd 3-15
perfmon.exe 3-23
raw 1-9
services 3-15
sqlhosts 1-11, 3-14
UNIX operating system 1-9
%INFORMIXSERVER%.cmd 3-1
Find Error utility Intro-10
finderr
script 3-21
utility Intro-10
Fragmentation
described 1-16
expression-based 1-16
round-robin 1-16

G
Generic B-trees 2-22
Getting error message
information 3-21
Global Language Support
(GLS) Intro-4
described 2-25
GLS library 2-25
locales 2-25
using nondefault locales 3-19
GLS. See Global Language Support.
Graphical administration
tools 1-25

H
Hash joins 2-9
Heterogeneous commit protocol,
described 1-23
High-availability data replication,
described 2-8
High-Performance Loader
described 2-9
utilities for 2-10
HKEY_LOCAL_MACHINE
registry 3-16
hosts file 3-15
HPL. See High-Performance
Loader.
Inheritance 2-20
Initializing disk space 3-18
Installing Informix products 3-10
Interfaces
DB-Access 3-5
DB/Cockpit 3-5
Informix SNMP subagent 3-5
Optical Subsystem 3-6
Interprocess communication 1-7
IPC. See Interprocess
communication.
upload utility 4-9
IPX/SPX. See Network.
ISM. See Informix Storage Manager.
ISO 8859-1 code set Intro-4
Isolation level
definition of 2-28

J
Join, methods 2-9

L
Library
GLS 2-25
INFORMIX-GLS 3-9
ODBC 3-8
TP/XA 2-18
Locale Intro-4
default Intro-4
en_us.8859-1 Intro-4
GLS 2-25
Locking, definition of 2-27
Logging. See Transaction logging.
Logical log, purpose of 1-18
Logical units of storage, list of 1-8
Logical volume manager as
mirroring alternative 1-20
Logical-log files, description
definition of 2-12
Long identifiers 2-26
M
Machine notes Intro-10
Manuals
for additional Informix products, listed 4-21
listed 4-16
See also Documentation.
Media failure, restoring data 1-18
Memory management
buffering transactions 1-15
dynamically sharing memory 1-15
Message file for error messages Intro-9
Method, access 2-21
Migration, information on 3-21
Mirroring
description of 1-19
logical volume manager alternative 1-20
RAID alternative 1-20
Module, DataBlade 2-35
Monitoring tools
database server utilities 3-24
UNIX 3-23
Windows NT 3-23
Multiplexed connection, description of 1-11
Multithreaded processes, description of 1-9

N
Named pipe connection, mentioned 1-12
Named row type, definition of 2-31
Name, long user 2-26
Network
configuration files 3-15
connection information 3-15
management with SNMP 3-5
security files 3-15
Network connection 1-12
New features of this product Intro-5

O
Object-oriented database management system 1-3
Object-relational database 2-20
Object-relational database management system 1-3, 1-5
OLTP applications, description of 2-24
ON-Archive
description of 2-13
ontape compatibility 2-14
onaudit utility 2-11, 4-9
ON-Bar, described 2-14
oncheck utility 2-11, 4-7
oncockpit 3-5
ONCONFIG
configuration file 3-12
configuration file, description of 3-16
configuration file, for different environments 3-16
environment variable 3-12
ondblog utility 2-11
oninit utility 2-11, 4-7
On-line help Intro-9
On-line manuals Intro-9
on-line mode 3-18
onload utility 2-11
onlog utility 2-11, 4-7
onmode utility 2-11, 4-7, 4-8
onparams utility 2-11, 4-7
onperf utility 2-11, 4-8
described 1-25
onpload utility 2-10, 4-9
onprobe 3-5
onshowaudit utility 2-11, 4-9
ontape utility 2-11, 4-7
ontape utility 2-11, 4-7
ontape utility 2-11, 4-7
compatibility with ON-Archive 2-14
described 2-13
listed 2-11
onunload utility 2-11, 2-15
Operating system
and scalability 1-13
managing I/O 1-14
raw and cooked disk space 1-9
thread processing 1-9
UNIX files 1-9
using tools 3-23
Operational data store, described 2-19
Optical media
advantages of 3-6
platter 3-6
Optical Subsystem
described 3-6
Optimizer, cost-based 2-9

P
Parallel database query 1-17
Parallel processing,
mentioned 1-15
Parallelization, described 1-17
passwd file 3-15
PATH environment variable 3-12,
3-19
PDQ. See Parallel database query.
Performance
advantages of raw-disk management 1-14
and DB/Cockpit 3-5
and shared memory 1-7
capturing data 3-23
data replication 2-7
Enterprise Replication 2-8
fragmentation 1-16
High-Performance Loader 2-9
memory management 1-15
monitoring 3-22
onperf 1-25
parallelization 1-17
Performance Monitor 3-23
scalability 1-13
SPL routines 2-4
tuning mechanisms 1-14
UDRs 2-34
Physical units of storage, list of 1-8
Platform icons Intro-8
Point-in-time recovery, described 1-21
Preparring, the ONCONFIG file 3-16
Primary access method 2-23
Printed manuals Intro-9, 4-16
Processes, compared to threads 1-9
Product installation 3-10
Product icons Intro-8
Products, supported 3-4
Program group Documentation notes Intro-11
Release notes Intro-11
Protocol. See Network.

Query and cost-based optimizer 2-9
definition of 2-3
distributed 1-22
language, structured 2-3
on system catalog 2-5
parallel database 1-17

Relational database
and DB-Access 3-5
and SQL 2-3
application types 2-24
components of 1-4, 2-3
data types 2-6
demonstration 3-20
multithreaded server 1-6
object-oriented 2-20
system catalog 2-5
Relational database management system
described 1-5
illustrated 1-4
object, described 1-5
Release notes Intro-10
Repository, described 2-19
Requirements ANSI 2-16
basic configuration 3-11
UNIX security 1-24
Resource manager, in DTP environment 2-18
Restore
description of 2-12
ON-Bar 2-14
point-in-time 1-21
.rhosts file 3-15
rofferr script 3-21
rofferr utility Intro-10
Role separation, description of 2-29
Roll back
defined 2-28
mentioned 1-19
Roll forward, mentioned 1-19
Round-robin fragmentation 1-16
Routine, external 2-34
Row type
named 2-31
unnamed 2-31
R-tree access method 2-23

S
Salvaging logical-log files 2-12
Sbspace, description of 2-33
Schema, Information 2-29
Script
.dbaaccess 3-20
dbaaccess ud 3-20
esqldemo 3-20
finderr 3-21
installation 3-10
rofferr 3-21
SDK products supported 3-6
Security
database server 1-24
files for network 3-15
role separation 2-29
with stored procedures 1-24
Server. See Database server.
services file 3-15
SET LOG statement 2-28
Setting environment variables 3-13, 3-14
Shared data 1-7
Shared memory
dynamic, management of 1-15
for interprocess communication 1-7
performance 1-7
purposes of 1-7
Shared-statement cache 2-26
Simple large object, definition of 2-32
Smart large object 2-33
API, mentioned 2-33
description of 2-33
sbspaces, and 2-33
SMI. See System-monitoring interface.
SMP. See Symmetric multiprocessing.
SNMP, described 3-5
Software dependencies Intro-4
SQL API, configuring environment for 3-19

R
RAID as mirroring alternative 1-20
Raw disk
management 1-14
space 1-9
RDBMS. See Relational database management system.
Recovery
ON-Bar 2-14
point-in-time 1-21
tools you cannot use 2-15
Registry
HKEY_LOCAL_MACHINE 3-16
setting environment variables in 3-14
.sqlhosts information 1-11
Related reading Intro-11
SQL statement for database server security 1-24
shared caching 2-26
sqlhosts file
description of 3-15
SQL. See Structured Query Language.
Stored procedure language, relation to SQL 2-4
Stored procedure, and security 1-24
stores_demo database Intro-5
stores_demo database, overview of 3-20
Structured Query Language and client applications 1-10
and initial connection to database server 3-20
relation to SPL 2-4
superstores Intro-5
superstores_demo database Intro-5
superstores_demo database, overview of 3-20
Support
buffered logging 2-28
client/server configurations 1-13
connection types 1-11
data replication 2-7
data types 1-4, 2-6
data warehouse 2-19
database 2-16
global language feature 2-25
high-performance loader 2-10
interface and client product 3-4
logical units 1-8
manuals for products 4-21
multimedia 1-21
network connection 1-12
physical units 1-8
TEXT and BYTE on optical platter 3-6
utilities 2-10
Symmetric multiprocessing and dynamic scalable
architecture 1-6
description of 1-13
Synchronous data replication 2-7
System catalog, description of 2-5
System failure, and ON-Bar 2-14
System requirements
database Intro-4
software Intro-4
System-monitoring interface
and administration 1-22
and performance 3-24

T
Table
access methods 2-21
fragmentation 1-16
information on 2-29
inheritance 2-32
locking 2-27
replication 2-8
security level 1-24
system catalog 2-5
Table fragmentation. See Fragmentation.
Task-documentation matrix 4-5
Tasks
administrative 3-21
basic, listed 3-3
documentation for 4-5
for application developers 4-4
for database administrators 4-4
for database server administrators 4-4
for database server operators 4-4
for database users 4-5
for programmers 4-4
TERM environment variable 3-12
TEXT data type 2-32
Thread
description of 1-9
dynamic allocation of 1-15
for client applications 1-9
Tip icons Intro-7
Tool
data management 2-7
DB/Cockpit 3-5
dynamic system administration 1-22
graphical administration 1-25
performance monitoring 3-23
SMI 1-22
TP/XA
described 2-18
library 2-18
when to use 2-18
Transaction
definition of 2-17
distributed 2-17
rolling back 2-28
Transaction logging
buffered 2-28
effect on database server type 2-28
using SET LOG statement 2-28
Tree
B 2-22
R 2-22

U
UNIX operating system
default locale for Intro-4
files 1-9
security requirements 1-24
setting environment variables 3-13
Unnamed row type, definition of 2-31
User name, long 2-26
User-defined access methods 2-36
User-defined aggregate (UDA) 2-36
User-defined data types 2-30
User-defined routine (UDR) and UDA 2-36
User-defined routine (UDR) benefits 2-34
description of 2-34
developing 3-10
Users, types of Intro-3, 4-3
Utilities
command-line, and setenv.cmd 3-14
data management 2-7
for auditing 4-20
for migration 4-19
listed 2-10
not to use for recovery 2-15
UNIX 3-23
Utility
chkenv 3-13
DB-Access 3-5, 3-20, 4-15, 4-17
dbexport 2-10, 2-15, 4-8
dbinport 2-10, 4-8
dbload 2-10, 4-8
dbschema 2-11, 2-29, 4-8
DB/Cockpit 3-5
Informix Find Error 3-21
iostat 3-23
ipload 2-10, 4-9
onaudit 2-11, 4-9
oncheck 2-11, 4-7
ondblog 2-11
oninit 2-11, 4-7
onload 2-11
onlog 2-11, 4-7
onmode 2-11, 4-7, 4-8
onparams 2-11, 4-7
onperf 2-11, 4-8
onload 2-10, 4-9
onshowaudit 2-11, 4-9
onspaces 2-11, 4-7
onstat 2-11, 4-7
ontape 2-11, 2-13, 4-7
onunload 2-11, 2-15
ps 3-23
sar 3-23
vmstat 3-23

W
Warning icons Intro-7
Windows NT
default locale for Intro-4
Performance Monitor 3-23
setting environment variables 3-14

X
XA interface standard, adherence to 2-18
X/Open compliance level Intro-13

V
View
information on 2-29
Information Schema 2-11
Virtual processor
description of 1-9
dynamic thread allocation 1-15
shared-memory component 1-7
user-defined 2-36