Oracle® Database Gateway for DB2/400

Installation and User's Guide

10*g* Release 2 (10.2) for IBM iSeries OS/400

B16222-02

August 2007

ORACLE®

Oracle Database Gateway for DB2/400 Installation and User's Guide, 10*g* Release 2 (10.2) for IBM iSeries OS/400

B16222-02

Contents

2 Release Information

3 System Requirements

4 Installing the Gateway

5 Oracle Net

6   Configuring the Gateway

7   Using the Gateway

8 Administering the Gateway

9 Developing Applications

10   Error Messages, Diagnosis, and Reporting

A   Oracle Net Files and Members

B   Globalization Support

C   Data Dictionary Views

Index

Preface

Read this guide if you are responsible for performing tasks such as:

- installing and configuring the gateway

- administering the gateway

- using the gateway

Understand the fundamentals of database gateways and OS/400 before using this guide for installation or system administration.

Intended Audience

This guide is intended for anyone installing, configuring and administering the gateway, as well as for developers writing applications that access remote host databases through the gateway.

Documentation Accessibility

Our goal is to make Oracle products, services, and supporting documentation accessible, with good usability, to the disabled community. To that end, our documentation includes features that make information available to users of assistive technology. This documentation is available in HTML format, and contains markup to facilitate access by the disabled community. Accessibility standards will continue to evolve over time, and Oracle is actively engaged with other market-leading technology vendors to address technical obstacles so that our documentation can be accessible to all of our customers. For more information, visit the Oracle Accessibility Program Web site at

http://www.oracle.com/accessibility/

Accessibility of Code Examples in Documentation

Screen readers may not always correctly read the code examples in this document. The conventions for writing code require that closing braces should appear on an otherwise empty line; however, some screen readers may not always read a line of text that consists solely of a bracket or brace.

Accessibility of Links to External Web Sites in Documentation

This documentation may contain links to Web sites of other companies or organizations that Oracle does not own or control. Oracle neither evaluates nor makes any representations regarding the accessibility of these Web sites.

TTY Access to Oracle Support Services

Oracle provides dedicated Text Telephone (TTY) access to Oracle Support Services within the United States of America 24 hours a day, seven days a week. For TTY support, call 800.446.2398.

Product Name

The complete name for this product is Oracle Database Gateway for DB2/400, also abbreviated DG4DB2/400 in this document.

Platform Name

In this manual, the platform is called OS/400. The terms AS/400 and OS/400 both refer to the IBM iSeries computers and operating system, which are referred to by IBM as IBM eServer iSeries servers, and include the IBM i5.

Conventions

Examples of input and output to the system are shown in a special font:

```
WRKCFGSTS *DEV device_name
```

Punctuation, such as commas and quotes, must be entered as shown unless otherwise specified. All output is shown as it actually appears. For input, the following conventions apply:

UPPERCASE Uppercase words, such as `WRKCFGSTS`, indicate that a word or phrase must be entered exactly as spelled.

Italic code font Lowercase or uppercase entries in Italic font, such as `device_name`, indicate that a word or phrase must be substituted, such as the actual device name.

< > Angle brackets indicate that the enclosed arguments are required, and that at least one of the arguments must be entered. Do not enter the brackets themselves.

[ ] Square brackets indicate that the enclosed arguments are optional. Do not enter the brackets themselves

| Vertical lines separate choices.

... Ellipses indicate that the preceding item can be repeated. You can enter an arbitrary number of similar items.

Commands, reserved words, and keywords appear in uppercase in both examples and text. A file ID can appear with both uppercase and lowercase text. When portions of a file ID appear in *italic case*, this indicates that those portions in *italic case* can vary. Reserved words and keywords must always be entered as is and have reserved meanings within Oracle applications.

Related Documents

Oracle Books:

For more information, refer to the following Oracle documentations:

- *Oracle Database Advanced Security Administrator's Guide*

- *Oracle C++ Call Interface Programmer's Guide*
- *Oracle Call Interface Programmer's Guide*
- *Oracle Database Advanced Replication*
- *Oracle Database Application Developer's Guide - Fundamentals*
- *Oracle Database Application Developer's Guide - Large Objects*
- *Oracle Database Application Developer's Guide - Object-Relational Features*
- *Oracle Database Backup and Recovery Advanced User's Guide*
- *Oracle Database Backup and Recovery Basics*
- *Oracle Database Administrator's Guide*
- *Oracle Database Error Messages*
- *Oracle Database Globalization Support Guide*
- *Oracle Database New Features*
- *Oracle Database Performance Tuning Guide*
- *Oracle Database Reference*
- *Oracle Database Security Guide*
- *Oracle Database SQL Quick Reference*
- *Oracle Database SQL Reference*
- *Oracle Database Heterogeneous Connectivity Administrator's Guide*
- *Oracle Database Net Services Administrator's Guide*
- *Oracle Database Net Services Reference*
- *Oracle Database Backup and Recovery Quick Start Guide*
- *Oracle Database Backup and Recovery Reference*
- *Oracle Database PL/SQL Packages and Types Reference*
- *Oracle Database PL/SQL User's Guide and Reference*
- *Oracle Secure Backup Reference*
- *Oracle Secure Backup Administrator's Guide*
- *Oracle Secure Backup Installation Guide*
- *SQL*Plus Quick Reference*
- *SQL*Plus User's Guide and Reference*

IBM Books:

Refer to the IBM documents for your platform and operating system.

Storage Measurements

Storage measurements use the following abbreviations:

- KB, for kilobyte, which equals 1024 bytes
- MB, for megabyte, which equals 1, 048, 576 bytes

- GB, for gigabyte, which equals 1, 073, 741, 824 bytes

1

Introduction

This chapter presents information about the architecture, uses, and features of the Oracle Database Gateway for DB2/400. It contains the following sections:

- Introduction on page 1-1
- 10g Release 2 Gateways on page 1-2
- Two-Phase Commit and Multisite Transactions on page 1-6
- Site Autonomy on page 1-6
- Migration and Coexistence on page 1-6
- Security on page 1-6
- Gateway Architecture on page 1-7
- How the Gateway Works on page 1-7

1.1 Introduction

In today's global economy, information is a company's most valuable resource. Whether you need to analyze new markets, tailor your products to meet local demands, increase your ability to handle complex customer information, or streamline operations, your company requires fast access to current and complete information.

Company growth and diversification often result in geographically scattered data that is processed with a collage of applications that may be using incompatible networks, platforms, and storage formats. Diverse application standards and storage formats can make integration of information difficult. For many years, Oracle has offered integration technologies to overcome these technical barriers. Database Gateways simplify complex systems and remove obstacles to information flow, thereby providing your company the opportunity to focus on business.

1.1.1 Protection of Current Investment

Oracle Database Gateway for DB2/400 gives your company the ability to develop its information systems without forfeiting your investments in current data and applications. The gateway gives you access to your Oracle and DB2/400 data with a single set of Oracle applications while you continue to use existing IBM applications to access the IBM data. You can also use more productive database tools and move to a distributed database technology without giving up access to your current data.

If you choose to migrate to Oracle Database technology and productivity, then the gateway allows you to control the pace of your migration. As you transfer applications

from your previous technology to the Oracle Database, you can use the gateway to move the DB2/400 data into Oracle databases.

The Oracle Database Gateway for DB2/400 enables you to:

- Integrate heterogeneous database management systems so that they appear as a single homogeneous database system.

- Read and write data from Oracle applications to data in DB2/400 in addition to reading and writing data from Oracle applications to data in any Oracle server (DB2/400 Large objects can be read only at this time).

1.2 10*g* Release 2 Gateways

The Oracle Database 10*g* provides the foundation for the next generation of the Gateways, which deliver enhanced integration capabilities by exploiting Oracle Heterogeneous Services. Heterogeneous Services is a component of the Oracle Database 10*g* server. The Oracle Database 10*g* server provides the common architecture for future generations of the gateways. For detailed information about Oracle Heterogeneous Services, refer to *Oracle Database Heterogeneous Connectivity Administrator's Guide*.

Version 10*g* Release 2 gateways are closely integrated with the Oracle Database 10*g* server, and they enable improved performance and enhanced functionality while still providing transparent integration of Oracle data and non-Oracle data. For example, connection initialization information is available in the local Oracle Database 10*g* server, reducing the number of round trips and the amount of data that is sent over the network. Structured Query Language (SQL) execution is also faster, because statements issued by an application are parsed and translated once and can then be reused by multiple applications.

Version 10*g* Release 2 gateways leverage the enhancements in the Oracle Database 10*g* server, and you can quickly extend those benefits to your non-Oracle data.

1.2.1 Advantages of the Gateway

Oracle Database Gateway for DB2/400 enables Oracle client applications to access DB2/400 through SQL. The gateway and Oracle Database 10*g* server together create the appearance that all data resides on a local Oracle Database 10*g* server, though data might be widely distributed. If data is moved from a DB2/400 database to an Oracle Database, then no changes in client application design or function are needed. The gateway handles all differences in data types and SQL functions between the application and the database.

Oracle Database Gateway for DB2/400 gives you the power to integrate your heterogeneous systems into a single, seamless environment, enabling you to make full use of existing hardware and applications throughout your corporate wide environment. You can eliminate the need to rewrite applications for each configuration and avoid the tedious, error-prone process of manual data transfer. Together with the Oracle world-class tools, networking, and data server technology, the Oracle Database Gateway for DB2/400 sets the standard for seamless, enterprise wide information access.

1.2.1.1 Transparency at All Levels

By using the Oracle Database Gateway for DB2/400, you can achieve transparency at every level within your enterprise.

- Location transparency:

Users can access tables by name, without having to understand the physical location of the tables.

- Network transparency:

  The gateways exploit the Oracle Net technology to allow users to access data across multiple networks without concern for the network architecture or protocols. Multiple protocols are supported.

- Operating system transparency:

  You can access data that is stored under multiple operating systems without being aware of the different operating systems that hold the data.

- Data storage transparency:

  Data can be accessed regardless of the database or file format.

- Access method transparency:

  You can use a single dialect of SQL for any data store, thereby eliminating the need to code for database-specific access methods or SQL implementations.

1.2.1.2 Extended Database Services

The following are some of the more sophisticated Oracle Database 10*g* server services that are available through the gateway.

- SQL functionality:

  Your application can access all your data using Oracle SQL, which is rich in features. Advanced Oracle Database 10*g* server functionalities, such as outer joins, are available even if the target data stores do not support them in a native environment. The manner in which the gateways are integrated with the Oracle Database 10*g* server ensures that the latest features of each database release are always available immediately to the gateway.

- Distributed capabilities:

  Heterogeneous data can be integrated seamlessly because Oracle Database distributed capabilities, such as `JOIN` and `UNION`, can be applied against non-Oracle data without any special programming or mapping.

- Distributed query optimization:

  The Oracle Database 10*g* server can use its advanced query optimization techniques to ensure that SQL statements are executed efficiently against any of your data. The data distribution and storage characteristics of local and remote data are considered equally.

- Two-phase commit protection:

  The Oracle two-phase commit mechanism provides consistency across data stores by ensuring that a transaction that spans data stores is still treated as a single unit of work. Changes are not committed, nor permanently stored, in any data store unless the changes can be committed in all data stores that are affected.

- Stored procedures and database triggers:

  The same Oracle stored procedures and database triggers can be used to access all of your data, ensuring uniform enforcement of your business rules across the enterprise.

1.2.1.3 Extended Advanced Networking, Internet, and Intranet Support

The gateway integration with the Oracle Database 10*g* server extends to non-Oracle data the benefits of the Internet and Oracle Net and the Oracle client/server and server/server connectivity software. These powerful features include:

- Application server support:

    Any Internet or intranet application that can access data in an Oracle database can also incorporate information from data stores that are accessible through the gateways. Web browsers can connect to the Oracle Database using any application server product that supports Oracle software.

- Implicit protocol conversion:

    Oracle and Oracle Net can work together as a protocol converter, allowing applications to transparently access other data stores on platforms that do not support the client network protocol. For example, an application can use SPX/IPX to communicate with an Oracle Database 10*g* server, which can use TCP/IP to communicate with the gateway and another data store on a non-SPX/IPX platform.

- Advanced Security:

    Non-Oracle data can be protected from unauthorized access or tampering during transmission to the client. This is done by using the hardware-independent and protocol-independent encryption and `CHECKSUM` services of the Advanced Security.

- Wireless communication:

    Oracle Mobile Agents, an Oracle industry-leading mobile technology, enables wireless communication to Oracle Database 10*g* server or any databases that are accessible through the gateways. This gives field personnel direct access to enterprise data from mobile laptop commuters.

1.2.1.4 Dynamic Dictionary Mapping

Before an application can access any information, the application must be told the structure of the data, such as the names of columns of a table and their lengths. Many other products require administrators to manually define this information in a separate data dictionary stored in a hub. Applications then access information using the hub dictionary instead of the native dictionaries of each database. This approach requires a great deal of manual configuration and maintenance on your part. Administrators must update the data dictionary in the hub whenever the structure of a remote table is changed.

Inefficient duplication is eliminated with Oracle Database Gateway for DB2/400. The simple setup of the gateway does not require any additional mapping. The gateway uses the existing native dictionaries of each database. Your applications access data using the dictionaries that are designed specifically for each database, which means that no redundant dictionary need ever be created or maintained.

1.2.2 SQL

Oracle Database Gateway for DB2/400 eases the application development and maintenance by allowing you to access any data using a uniform set of SQL. Changes to the location, storage characteristics, or table structure do not require any changes to your applications. ANSI and ISO standard SQL are supported, along with powerful Oracle extensions.

1.2.2.1 Passthrough and Native DB2/400 SQL

Native DB2/400 SQL can be passed through the gateway for running directly against DB2/400. This enables applications to send statements, such as a DB2/400 CREATE TABLE, to the gateway for execution on a target DB2/400 system.

1.2.2.1.1 Data Definition Language Oracle applications can create tables in target data stores by using native data definition language (DDL) statements.

1.2.2.1.2 Data Control Language You can issue native data control language (DCL) statements from an Oracle environment, allowing central administration of user privileges and access levels for heterogeneous data stores. Refer to *Oracle Database Heterogeneous Connectivity Administrator's Guide* for more information about DCL.

1.2.3 Stored Procedures and Functions

The gateway enables you to exploit both Oracle and non-Oracle stored procedures and functions, leveraging your investments in a distributed, multi-database environment. Oracle stored procedures and functions can access and update multiple data stores easily, with no special coding for heterogeneous data access.

1.2.3.1 Oracle Stored Procedures and Functions

Oracle stored procedures and functions enable you to access and update DB2/400 data using centralized business rules that are stored in the Oracle Database 10*g* server. The use of Oracle stored procedures and functions can increase the database performance by minimizing network traffic. For example, instead of sending individual SQL statements across the network, an application can send a single EXECUTE command to begin an entire PL/SQL routine.

1.2.3.2 Native DB2/400 Stored Procedures

The gateway can execute DB2/400 stored procedures using standard Oracle PL/SQL. The Oracle application executes the DB2/400 stored procedure as if it were an Oracle remote procedure.

> **Note:** If you are going to use DB2/400 stored procedures, then you must journal the ORACLE2PC table.

1.2.3.3 Native DB2/400 Stored Functions (User Defined Functions)

The gateway can execute DB2/400 User Defined Functions using standard Oracle PL/SQL. In addition, a User Defined Function may be executed within a SQL statement. In both cases, the Oracle application executes the DB2/400 User Defined Function as if it were a remote Oracle stored function. When the User Defined Function is executed as part of a SELECT statement, it may not cause any database changes on the remote (DB2/400) database.

1.2.4 Languages

Any application or tool that supports the Oracle Database 10*g* server can access over 30 different data sources through the Oracle gateways. A wide variety of open system tools from Oracle and from third-party vendors can be used, even if the data is stored in legacy, proprietary formats. Hundreds of tools are supported, including ad-hoc query tools, Web browsers, turnkey applications, and application development tools.

1.2.5 SQL*Plus

Use SQL*Plus for moving data between the databases. This product gives you the ability to copy data from your department databases to corporate Oracle databases.

1.2.6 Oracle Database 10*g* Server Technology and Tools

The gateway is integrated into the Oracle Database 10*g* server technology, which provides global query optimization, transaction coordination for multi-site transactions, support for all Oracle Net configurations, and so on. Tools and applications that support the Oracle Database 10*g* server can be used to access heterogeneous data through the gateway.

1.3 Two-Phase Commit and Multisite Transactions

The gateway can participate as a partner in multisite transactions and two-phase commit. How this occurs depends on the capabilities of the underlying data source, meaning that the gateway can be implemented as any one of the following:

- a full two-phase commit partner
- a commit point site
- a single-site update partner
- a read-only partner

The deciding factors for the implementation of the gateway are the locking and transaction-handling capabilities of your target database.

Oracle Database Gateway for DB2/400, by default, is configured as a commit point site (that is, commit-confirm protocol). Optionally, you can configure the gateway as read-only if you choose to enforce read-only capability through the gateway. Other protocols are not supported. Refer to "Read-Only Gateway" on page 6-11 for more information.

1.4 Site Autonomy

All Oracle Database 10*g* server products, including gateways, supply site autonomy. For example, administration of a data source remains the responsibility of the original system administrator. Site autonomy also functions so that gateway products do not override the security methods of the data source or of the operating environment.

1.5 Migration and Coexistence

The integration of a data source through the gateway requires no changes to be made to applications at the data source. As a result of this, the Oracle Database 10*g* server technology is nonintrusive, providing coexistence and an easy migration path.

1.6 Security

The gateway does not bypass existing security mechanisms. Gateway security coexists with the security mechanisms that are already used in the operating environment of the data source.

Functionally, gateway security is identical to that of an Oracle Database 10*g* server, as described in the *Oracle Database Administrator's Guide*. Oracle Database security is mapped to the data dictionary of the data source.

1.7 Gateway Architecture

The gateway architecture consists of four main components as shown in Figure 1–1.

1. Client:

   The client is an Oracle application or tool.

2. Oracle Database 10*g* server:

   The Oracle Database 10*g* server is an Oracle instance.

3. Oracle Database Gateway for DB2/400:

   The Oracle Database Gateway for DB2/400 must be installed on an AS/400 system and is responsible for issuing the dynamic SQL calls to DB2/400. The gateway can access files in an OS/400 SQL Collection or files that are externally described to DB2/400.

   The gateway is not started as an Oracle instance and has no continuously running background processes. The gateway is started either by using the OS/400 STRSBS command or by using the ORAMON command. Individual gateway tasks run in a subsystem, as does the LISTENER task for TCP/IP.

4. DB2/400 Server:

   The DB2/400 database is the database that is being accessed by the gateway.

Multiple Oracle Database 10*g* servers can access the same gateway. A single gateway installation can be configured to access one, and only one, DB2/400 server, because there is only one instance of a DB2/400 database for each AS/400 system. Multiple gateways can be installed on a single AS/400 system.

Figure 1–1   Architecture of the Oracle Database Gateway for DB2/400
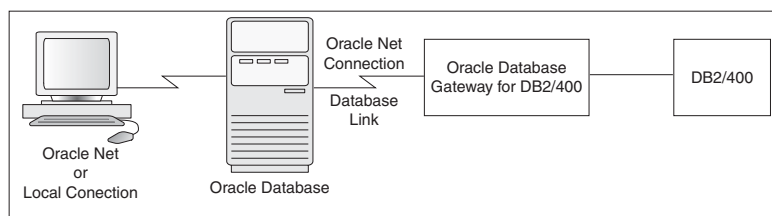
1.8 How the Gateway Works

The gateway has no database functions of its own. Instead, it provides an interface by which the Oracle Database 10*g* server can direct SQL operations to a DB2/400 database.

The gateway that is supporting the DB2/400 server is identified to the Oracle Database 10*g* server by using a database link. The database link is the same construct that is used to identify other Oracle Database 10*g* server databases.

Tables on the DB2/400 server are referenced in SQL as:

```
table_name@dblink_name
```

or, preferably, as:

```
library.table_name@dblink_name
```

If you create synonyms or views in the Oracle Database, then you can refer to tables on the DB2/400 server by using simple names as though the tables were local to the Oracle Database 10*g* server.

When the Oracle Database 10*g* server encounters a reference to a table on the DB2/400 server, the applicable portion of the SQL statement is sent to the gateway for processing. Any host variables that are associated with the SQL statement are bound to the gateway and, therefore, to the DB2/400 server.

The gateway is responsible for sending these SQL statements to the DB2/400 server. The DB2/400 server is responsible for executing the SQL statements and for detecting errors and returning responses.

1.8.1 Gateway Changes to the AS/400 System

Installing the gateway on the AS/400 system produces the following changes to the system:

1. A library is created, with the instance name as given in the first installation panel. (For an example, refer to Example 4–1, "Install Oracle Database Gateway Panel" on page 4-4). Almost all of the objects needed for this gateway instance will be found in this library, with the exception of some objects dealing with service programs and the items mentioned in parts 3 and 4 of this list.

    The gateway can also be installed into a previously created DB2/400 SQL COLLECTION that contains objects of type *FILE (physical and logical files), a journal of type *JRN, a journal receiver of type *JRNRCV, and a data dictionary object of type *DTADCT. The collection name will be the name of the library in which these objects are created by DB2/400.

2. A user profile is created. The user profile has the same name as the library that is created when you install the gateway.

3. A subdirectory entry is created in the /home directory in the Integrated File System. The subdirectory entry name will be the same as the instance name. So, if ORACLE is the instance name, then a subdirectory node will be created at /home/ORACLE. Further nodes will be created below this node. Agent trace files will appear (if requested) in a subdirectory of the created node. An initialization file will also appear in a subdirectory of this node.

4. A library named ORASRVLIB is created. If the installation software finds that it must install the service programs that are found on the installation medium, then a library with a name such as ORASRV*nnnn* (where *nnnn* is a number) will be created. The relevant service programs and a file that is used for Globalization Support transactions will be placed in that library. Additional files or members within files may be added to the ORASRVLIB library.

5. A message queue with the name of the installation library is created in the QUSRSYS library. This is as a result of using the CRTUSRPRF command to create the user profile that corresponds to the installation library.

6. A library named ORATEMP will be created to hold intermediate installation objects during the install.

1.8.2 SQL Differences

Not all SQL implementations are the same. The Oracle Database 10*g* server may support a larger set of built-in functions than the databases that are currently being accessed through the gateway. Or, the semantics of some functions in the DB2/400 server may not agree with an identically named function in the Oracle Database 10*g*

server. The SUBSTR function is an example of this, because the second and third arguments of SUBSTR in an Oracle Database 10*g* server may be negative. In DB2/400, the second and third arguments of the SUBSTR function cannot be negative. In any case, the Oracle Database 10*g* server and the gateway work together to convert the SQL to a form that is compatible with the specific DB2/400 server.

During this conversion, an Oracle Database 10*g* server function can be converted to a function that is recognizable to the specific DB2/400 server. For example, the Oracle Database 10*g* server NVL function is converted to the DB2/400 VALUE function.

Alternatively, the Oracle integrating server withholds functions that are not executable by the DB2/400 server, and performs those functions on the Oracle server side after the rows are fetched from the DB2/400 server. This is called post processing because it occurs after the rows are fetched from DB2/400. This post processing generally applies to SELECT statements.

The SUBSTR function, when used with two or three parameters, is an example of a function that is always post processed on the Oracle server side since the semantics of the SUBSTR function with two or thee parameters differs between Oracle 10*g* and DB2/400. The Oracle integrating server and the gateway cannot perform this kind of manipulation on UPDATE, INSERT, or DELETE statements because doing so changes transaction semantics.

1.8.3 Heterogeneous Services Architecture

This release of the Oracle Database Gateway for DB2/400 uses the Oracle Heterogeneous Services component within the Oracle Database 10*g* server. For detailed information about heterogeneous services, refer to the *Oracle Database Heterogeneous Connectivity Administrator's Guide*.

2

Release Information

This chapter describes the changes and corrected problems in this release. It contains the following sections:

- Product Set on page 2-1
- Changes and Enhancements on page 2-1
- Known Problems on page 2-2
- Known Restrictions on page 2-3

2.1 Product Set

The production components that are included on the product CD-ROM are:

- Oracle Database Gateway for DB2/400, 10*g* Release 2 (10.2.0.1.0)
- Oracle Net, Release 10.2.0.1.0

2.2 Changes and Enhancements

The following changes and enhancements have been made to the Oracle Database Gateway for DB2/400 for this current release:

New DB2 Data Types Supported

The DB2/400 large-object data types, namely binary large objects (`BLOB`s), character large objects (`CLOB`s), and graphic large objects (`DBCLOB`s) are now supported. However, these data types can only be used in `SELECT` statements. At the current time, data cannot be `INSERT`ed into columns with these data types. In addition the previous support in 10.2.0.1.0 for large objects via the use of locators is no longer available.

DB2 stored functions (UDFs) Supported

DB2/400 User Defined Functions are now supported. DB2/400 UDFs must be made known to Oracle before their use, however. The IFS file `/home/myinst/rdbms/admin/hs_call_names.ora` contains a line for each UDF.

Savepoints Supported

Intermediate savepoints can now be set. Data can be rolled back to these savepoints.

2.2.1 Bugs

Bugs fixed in this release of the Oracle Database Gateway for DB2/400 are as follows:

Bug 4959570: const `CHAR` fields in DD Views sometimes coming back as `RAW`s

Bug 4865825: Gateway listener hangs

Bug 4566044: `SQL0804` when calling DB2/400 stored procedure with `DECIMAL IN` parameter

Bug 4473636: `NLS_LANG CZECH.CZECH REPUBLIC.EE8EBCDIC870` not accepted by GTW

Bug 4414487: New DataDictionary Definitions for Streams replication (to accompany fix for generic bug 3653933.

Bug 4338891: Column lengths get increased three times and padded with blanks in Oracle UTF8 DB

Bug 4333999: `CRTORAGTWI` fails with `CPFA0A9` copying timezone.dat

Bug number 4307761: Dead connection detection is not working when client connection is broken

Bug number 4093610: CHAR data padded with multibyte blank

Bug number 4085764: `MCH0601 F/QC2STRMI/STRSTR/7 T/HOADB2/HOAParS/26` during `INSERT`

Bug number 4019083: `MCH0601`: space offset &2 or &9 is outside current limit for object.

Bug number 3985615: `ORA-0004` homtiap_initAgentProcess returns non-zero return code `1804` for `CCSID 930`

Bug number 3975301: `SQL0901` on second and subsequent calls to stored procedure with no parameters (check IBM APAR SE18413)

Bug number 3907455: `ORA-01017` on logon under `V5R3M0`

Bug number 3803899: Problems when cloning a new instance or when running a cloned instance.

Bug number 3755039: After upgrading to OS/400 V5R3, TG4DB2400 returns `ORA-01017`.

Bug number 3699834: When duplicating data via SQL*Plus `COPY` command, the length of copied data is incorrect.

Bug number 3695959: `GRAPHIC(127)` in DB2 is incorrectly mapped to `VARCHAR2` in ORACLE DB

Bug number 3693493: "`CREATE VIEW AS SELECT ...`" incorrectly changed the length of columns.

Bug number 3629826: `CCSID 930/5026` Gateway trace not displayed in `WRKTRCLOG`

2.3 Known Problems

The problems documented in the following section are specific to the Oracle Database Gateway for DB2/400 and are known to exist in this release of the product. These problems are currently being addressed by Oracle. Refer to the respective bugs for the current status of each problem. If you have any questions or concerns about these problems, then please contact Oracle Support Services.

A current list of problems is available online. Contact your local Oracle office for information about accessing this online information.

2.4 Known Restrictions

The restrictions documented in this section are known to exist for the products in this release of the Oracle Database Gateway for DB2/400. Refer to Chapter 9, "Developing Applications" for information about limitations when developing your applications.

2.4.1 General Restrictions

The following sections describes general restrictions when using the gateway.

2.4.1.1 SQL Development Kit, Product 5722-ST1

As described in Bug 3616425, the Gateway needs to have product 5722-ST1 (SQL Development Kit) installed on the AS/400. This is a software prerequisite if one Gateway instance is cloned from another.

2.4.1.2 Selecting CHAR Column Returns Width Doubled

Selecting CHAR/VARCHAR column returns its width doubled. To correct the problem reported in bug number 2451535, create the data area named ONLS_MINNB and set its value to 2, as in the following command:

```
CRTDTAARA instlib/ONLS_MINNB TYPE(*CHAR) LEN(10) value('2')
```

where *instlib* is the installation library.

2.4.1.3 Coexistence with Previous Version Gateway: Data Dictionary Tables

After the data dictionary tables are installed by a 10*g* Release 2 Gateway, *do not* reinstall the data dictionary tables from a previous version gateway, or you will not receive the benefits of the full capabilities of 10*g* Release 2.

2.4.1.4 Oracle Fast Refresh Snapshots

Oracle fast refresh snapshots are not supported between the gateway and the Oracle Database 10*g*. However, Oracle complete refresh snapshots are supported between the gateway and the Oracle Database 10*g* server.

2.4.1.5 Oracle SQL Command INSERT

When copying data from an Oracle Database 10*g* server to a DB2/400 server, the Oracle SQL command INSERT is not supported. The SQL*Plus COPY command must be used. Refer to "SQL*Plus COPY Command" on page 7-9, for more information.

2.4.1.6 Verify the QCRTAUT Parameter Setting Before Installation

Before installing the gateway, verify that the QCRTAUT parameter is set to allow installation (not set to *EXCLUDE). Refer to "Installation Steps" on page 4-2 for information about verifying this value.

2.4.1.7 Date Arithmetic

Date arithmetic is not supported for SQL statements. For example, the following SQL expressions *do not* work:

```
date + number
number + date
date - number
date1 - date2
```

Refer to Chapter 9, "Developing Applications" for additional information.

2.4.1.8 Julian Dates Support

You must use the J option on the `TO_DATE` and `TO_CHAR` functions to receive valid Julian dates.

2.4.1.9 GRAPHIC Constants in SQL Commands

`GRAPHIC` constants in SQL statements that are valid in one character set may not be valid as G type or N type `GRAPHIC` constants in DB2/400. For example, graphic constants in character set `KO16KCSC5601` may contain both single-width and double-width characters. `GRAPHIC` constants in DB2/400 can contain only double-width characters.

> **Caution — Potential Loss of Data:**   Oracle strongly recommends that you do not directly use graphic constants in SQL statements. If you do use graphic constants, then you should ensure that all of the characters within a graphic constant are double-width characters.
>
> Oracle recommends that you use bind variables if you wish to use such constants.
>
> The gateway checks the graphic constants found in `INSERT` statements and can force the single-width characters to become double-width characters, if requested. With graphic data resulting from a `SELECT`, the gateway can force those double-width characters back to single-width characters.
>
> The ability to do this results in a data integrity exposure because the gateway cannot determine whether a given double-width character from a DB2/400 column was `INSERT`ed as a single-width character that was forced to become a double-width character or was entered as a true double-width character.

2.4.2 Owners of DB2/400 Components

The following sections describe restrictions related to DB2/400 components.

2.4.2.1 DD Basic Tables and Views

The owner of Data Dictionary (DD) basic tables and views is `OTGDB2`. This cannot be changed.

2.4.2.2 Binary Literal Notation

Oracle SQL uses hexadecimal digits surrounded by single quotation marks to express literal values being compared or inserted into columns defined as `RAW`. Currently, this is not converted to DB2/400 syntax (an X followed by quoted hexadecimal digits) when the SQL destination is the gateway. You must use bind variables to compare or insert into a DB2/400 server column that is defined with the `FOR BIT DATA` option.

2.4.2.3 Programmatic Limitations

Gateway design requires that all host variables in a SQL operation be bound before a describe function is performed. This is required because of a limitation in the Oracle Call Interface (OCI), which requires that all OCI bind calls for a given statement must be completed before an OCI describe call is made.

2.4.2.4 Columns Defined with RAW Data

When you select RAW data into character bind variables, the CHAR column must be two times the size of the RAW data. Selecting RAW data into character bind variables causes implicit RAW-to-HEX conversion. If the character bind variable column is too small, then the SELECT statement fails.

2.4.2.5 GLOBAL_NAMES Initialization Parameter

If GLOBAL_NAMES is set to TRUE in the Oracle integrating server INIT.ORA file, then, to be able to connect to the gateway, you must specify the value of the Heterogeneous Services (HS) initialization parameter, HS_DB_DOMAIN, in the "Change Oracle Gateway Initialization Parameters" panel to exactly match the value of the Oracle server DB_DOMAIN parameter. Refer to Chapter 6, "Configuring the Gateway" for more information.

2.4.2.6 Precompiler Limitations

The SQLCHECK option must be set to NONE when precompiling programs with Oracle Precompilers.

2.4.2.7 Some SQL Functions Post processed

The Oracle Database 10*g* server provides more functionality for some SQL functions compared to similarly named DB2/400 SQL functions. As a result, these SQL functions are not passed through from the Oracle Database 10*g* server to DB2/400. They are, instead, post processed. SUBSTR is an example of such a function because the semantics of the SUBSTR function in Oracle as compared to that in DB/400 show slight, but potentially problematic, differences. Some other SQL functions have different semantics in Oracle SQL and DB2/400 SQL. For details, refer to "SQL Functions" on page 9-20 in Chapter 9, "Developing Applications".

2.4.3 DB2/400 SQL Limitations

The following sections describe restrictions when using SQL.

2.4.3.1 Oracle Bind Variables

Oracle bind variables become DB2/400 SQL parameter markers when used with the gateway. The bind variables are therefore subject to the same restrictions as DB2/400 SQL parameter markers.

For example, the following statements are *not* allowed:

```
WHERE :x IS NULL
WHERE :x = :y
```

2.4.3.2 CONNECT BY Is Not Supported

Oracle Database Gateway for DB2/400 does not support CONNECT BY in SELECT statements.

2.4.3.3 Oracle Incremental Materialized View Refresh

Incremental Materialized View refresh is not supported between the gateway and the Oracle Database 10*g* server. However, Oracle complete Materialized View is supported between the gateway and the Oracle Database 10*g* server.

2.4.3.4 LONG Data Types

Any DB2/400 CHAR or VARCHAR column with a length greater than 4000 bytes is considered an Oracle LONG data type by the gateway.

When using a SELECT statement in SQL*Plus for any DB2/400 GRAPHIC column that is defined with a length greater than 2000 characters, or for any DB2/400 CHAR or VARCHAR column that is defined with a length greater than 4000 characters, the column will be seen by the Oracle database as a LONG column. In SQL*Plus, the default buffer length for a retrieved LONG column is 80 characters. For this reason, if you wish to retrieve a DB2/400 column that will be interpreted as an Oracle LONG column, then you must set the SQL*Plus buffer length correctly. If the buffer size is not sufficient to contain the column value, then you will receive the following error message:

```
ORA-28528: Heterogeneous Services datatype conversion error
```

To work around this problem, reset the size of the buffer using the SET LONG command in SQL*PLUS or Server Manager.

In addition, if you use a program to retrieve a LONG column on the Oracle client, then you must retrieve the entire LONG column in one call to the Oracle client. You cannot retrieve the LONG column in sections through the gateway.

3

System Requirements

This chapter describes the OS/400 hardware and software requirements for the Oracle Database Gateway for DB2/400. It contains the following sections:

- Hardware Requirements on page 3-1
- Software Requirements on page 3-2
- Documentation Requirements on page 3-2
- Distribution Kit on page 3-2

3.1 Hardware Requirements

The following sections describe the hardware requirements for installing and running the gateway.

3.1.1 Processor

The gateway requires any AS/400 processor that supports a version of OS/400 that is currently supported by IBM.

3.1.2 Memory

For most installations, a minimum of 7 MB of virtual memory on the AS/400 side is recommended to support each active user of the Oracle Database Gateway for DB2/400.

The total virtual memory requirement for each concurrent use of the gateway depends on the following factors:

- number of concurrent TCP/IP connections opened by each user
- number of data items being transferred between the gateway and the remote transaction program
- additional factors, such as configured network buffer size

3.1.3 CD-ROM Drive

A CD-ROM drive is required.

3.1.4 Disk Space

Permanent disk space of about 225 MB is required for installation of this product. Installation of further (cloned) instances of the product at the same level use

approximately 76 MB each. The actual installation may required an additional 150 MB of temporary disk space.

These numbers are for initial conditions. The disk space can be reduced by running the `REMOVEOBS` procedure (refer to). After running `REMOVEOBS`, each gateway instance will use 70 MB, and the common objects (service programs and the National Language files) will use approximately 236 MB.

3.2 Software Requirements

The system software configuration that is described in the following requirements is supported by Oracle as long as the underlying system software products are supported by their respective software vendors. Verify the latest support status with your system software vendors.

3.2.1 Operating System Requirements

This gateway requires any current version of OS/400 that is supported (with maintenance) by IBM.

The `RUNSQLSTM` command that is found in 5722-ST1 "SQL Development Kit" is needed when creating one instance from another (cloning an instance). It is not needed for an initial installation from an Oracle distribution medium (CD-ROM or file).

3.2.2 Oracle Integrating Server

The Oracle server that is to act as the Oracle integrating server requires the Oracle Database 10*g* server or Oracle9*i* server. The Oracle integrating server should be running with the latest released patch set for Oracle Database 10*g* server, Oracle9*i* server, or Oracle8*i* server (release 8.1.7 or later).

3.3 Documentation Requirements

In addition to the manuals supporting the Oracle Database Gateway for DB2/400, Oracle Net and Oracle Database 10*g* server manuals are recommended. You also need the appropriate OS/400 documentation for your system.

3.4 Distribution Kit

Before installing the gateway, verify that you have the correct CD-ROM and proper documentation:

- one Oracle Database Gateway for DB2/400 product CD-ROM

- appropriate Oracle documentation for installing, administering, and using the gateway: *Oracle Database Gateway for DB2/400 Installation and User's Guide for IBM iSeries OS/400*

- Oracle Database 10*g* server, Oracle9*i* server, or Oracle8*i* server (release 8.1.7 or later)

4

Installing the Gateway

Before you begin installation, ensure that you have the required hardware and software described in Chapter 3, "System Requirements". This chapter contains the following sections:

- Gateway Coexistence on page 4-1

- Checklists on page 4-1

- Preinstallation Steps on page 4-6

- Installation Steps on page 4-6

- Postinstallation Steps on page 4-6

4.1 Gateway Coexistence

Oracle Database Gateway for DB2/400 Release 10.2.0.1.0 can coexist with any previous version of the product. There are no known conflicts. Oracle Corporation does not support upgrading a previous version gateway to a version 10 gateway. You must reinstall the gateway. The various instances of the gateway on a computer have no knowledge of each other, other than through the DB2/400 files against which the gateways are executed, and through the data dictionary tables.

> **Caution — Potential Loss of Data:** After the data dictionary tables are installed by a release 10.2 Gateway, do not reinstall the data dictionary tables from an earlier release of the gateway. If this is done by mistake, then you can reinstall the version 10.2 data dictionary table by using the `CRTORADDB` command that is described in Chapter 6, "Configuring the Gateway".

4.2 Checklists

The following sections list the various checklists required while installing the gateway.

4.2.1 Preinstallation Checklist

- Step 1: Log On

- Step 2: Verify System Value for the QCRTAUT Parameter

4.2.2 Installation Checklist

- Step 1: Mount the CD-ROM

- Step 2: Start the Installation Process

- Step 3: Set Required Parameters

- Step 4: Verify Installation

- Step 5: Finish the Installation

- Step 6: Start the Gateway Subsystem

4.2.3 Postinstallation Checklist

- Step 1: Verify the Gateway Version

- Step 2: Verify Journaling

- Step 3: Journal the ORACLE2PC File

- Step 4: Configure Oracle Net

- Step 5: Remove Observability

- Step 6: Raise the Gateway Priority Level

4.3 Preinstallation Steps

Before installing the gateway, you must perform the following steps:

4.3.1 Step 1: Log On

Log on with a user profile that has the following special authorities: `*SECADM`, `*JOBCTL`, `*ALLOBJ`, and `*SYSCFG`. User profile `QSECOFR` (as distributed by IBM) has these authorities.

4.3.2 Step 2: Verify System Value for the QCRTAUT Parameter

Ensure that the system value for the `QCRTAUT` parameter is not `*EXCLUDE`. To find the current system value, enter the following statement from the command line:

```
WRKSYSVAL QCRTAUT
```

If the system value is `*EXCLUDE`, then change it to any other less restrictive value in order to allow installation of the gateway, and remember to change `QRCTAUT` back to `*EXCLUDE` after installing the gateway.

4.4 Installation Steps

The panels below (beginning with Example 4–1, "Install Oracle Database Gateway Panel") are shown as they appear when they are first displayed. Any values on the panel are default values that are provided by Oracle. Default values are used unless you change them before going on to the next panel.

The installation procedure defines an OS/400 user ID that is identical to the name of the install library. This user ID is known as the gateway ID. The installation procedure also defines that ID with a user class of `*USER`, which has `*JOBCTL` authority. This ID is given authority over the entire library. To do that, the commands executed during installation are:

```
GRTOBJAUT OBJ(QSYS/gateway-id) OBJTYPE(*LIB) USER(gateway_id) AUT(*ALL)
GRTOBJAUT OBJ(gateway-id/*ALL) OBJTYPE(*ALL) USER(gateway_id) AUT(*ALL)
```

where `gateway-id` is both the gateway user ID and the library name.

The security changes that are in the following six lines of example code may be made to allow the gateway to adopt the security authority of the "connect to" user ID. The GRTOBJAUT should be done only if the referenced user (in this case, *PUBLIC) does not already have the indicated authority to the object.

```
GRTOBJAUT OBJ(QSYS/QWTSETP) OBJTYPE(*PGM) USER(*PUBLIC) AUT(*USE)
GRTOBJAUT OBJ(QSYS/QSYGETPH) OBJTYPE(*PGM) USER(*PUBLIC) AUT(*USE)
GRTOBJAUT OBJ(QSYS/QSYRLSPH) OBJTYPE(*PGM) USER(*PUBLIC) AUT(*USE)
```

You can use the EDTOBJAUT command to check the authorities of the three objects. For example, to check the current authority of the QWTSETP program in the QSYS library, use the following command:

```
EDTOBJAUT OBJ(QSYS/QWTSETP) OBJTYPE(*PGM)
```

If the *PUBLIC line shows *USE, then no change is needed.

4.4.1 Step 1: Mount the CD-ROM

Insert the CD-ROM into the AS/400 CD-ROM drive.

4.4.2 Step 2: Start the Installation Process

Enter the following command:

```
LODRUN drive
```

where `drive` is the name of the CD-ROM drive where you mounted the product CD-ROM. The panel in Example 4–1, "Install Oracle Database Gateway Panel" is displayed.

4.4.3 Step 3: Set Required Parameters

The following parameter must be set: Instance Name

4.4.3.1 Panel

Install Oracle Database Gateway.

4.4.3.2 Action

You must fill in the `Instance name`. Use the default name of `ORACLE` or enter a name from one to six characters long.

The installation procedure either creates a new library, uses an existing (but empty) library, or uses an existing library that has been created using the DB2/400 SQL command CREATE COLLECTION. If the library does not fit into this scenario, then it will not be used. Refer to Example 4–1, "Install Oracle Database Gateway Panel". The library must contain no objects other than those objects that were created by the CREATE COLLECTION command.

The preferred method is to use STRSQL to create a DB2/400 collection and then to install the gateway into the library that is associated with that collection. Objects that are created later in the collection will be automatically journaled.

Example 4–1   Install Oracle Database Gateway Panel

```
            Install Oracle Database Gateway at V10.2.0.1.0

Type Choices, press Enter.

Instance name............  ORACLE              Name (up to six characters)




===>
F1=Help   F4=Prompt   F9=Retrieve   F10=Additional parameters   F12=Cancel

(C) COPYRIGHT ORACLE CORPORATION, 1994, 2006
```

After entering a name, press Enter. The panel in Example 4–2, "Install Oracle Database Gateway Panel, With Name Choices" appears.

Example 4–2   Install Oracle Database Gateway Panel, With Name Choices

```
            Install Oracle Database Gateway at V10.2.0.1.0
                                                    System: AS400A
Type Choices, press Enter.

Instance name....................  ORACLE        Name (up to six characters)
Instance password................  ORACLE        1-10 characters
Prestart jobs....................  *YES          *YES, *NO
TCP/IP port number...............  1521          1024-65534
Recovery user profile............  ORACLE        Name
Recovery user password...........  ORACLE        1-10 characters
Auxiliary storage pool id.........  1            1-16
Install Data Dictionary Support...  *YES         *YES, *NO


===>

F1=Help   F4=Prompt   F9=Retrieve   F12=Cancel
```

You may change the choices:

Table 4–1   Name Choices on Install Panel

Choice Name	Description
Instance name	is a name from one to six characters long. The default is ORACLE.
Instance password	is the password for the user ID, the Instance User Profile, (same as OS/400 User Profile) that is created during installation. The Instance User Profile (user ID) has the same value as the instance name.

Table 4–1   (Cont.) Name Choices on Install Panel

Choice Name	Description
Prestart jobs	prestarts the server jobs. Use *YES to prestart the server jobs, or *NO to prestart NO jobs. The server jobs run the Gateway executable. You can change how many jobs are prestarted by changing the value of the initial number of TCP/IP jobs parameter by using the CHGORAPJE command. For more information, refer to "CHGORAPJE, Change Prestarted Job Parameters" on page 6-6.
TCP/IP port number	is 1521, which is the default. If port 1521 is unavailable, then enter a different port number. Each instance requires a unique port number. Each instance has its own LISTENER.
Recovery user profile	is a profile name that you enter, or use the default of the gateway instance name.
Recovery user password	is a password for the recovery user profile. The default is the gateway instance name. Following installation, this password must be kept in synchronization with the password for the Recovery User Profile, which is changed by using the AS/400 CHGUSRPRF command. Refer to "CHGRECOPRF, Change Recovery Profile Parameters" on page 6-9
Auxiliary storage pool id	uses the default of 1, or you can enter another numeric ID.
Install Data Dictionary Support	turns on data dictionary installation. The default is *YES. Installation of the data dictionary view support lengthens the installation process. Only one Database Gateway data dictionary is created for the entire AS/400 computer. All Database Gateway instances use the same data dictionary. The data dictionary is created in the library OTGDB2.

After filling in the choices, press Enter to continue.

4.4.4 Step 4: Verify Installation

The following steps verify installation of the gateway.

4.4.4.1 Panel

Verify Oracle Database Installation.

4.4.4.2 Action

Enter ***YES** and press Enter to continue the installation process.

Example 4–3   Verify Oracle Install Panel

```
              Verify Oracle Install at V10.2.0.1.0
                                              System: AS400A



Verify that you want to install the product.  Once this process has started, it
will create a library and other objects with the name of the instance
specified below.  Enter *YES to verify this operation before it is started.

New instance name .........   ORACLE
```

```
enter *YES to verify.......   *NO     *NO, *YES




===>                                                             Bottom

F1=Help   F4=Prompt   F9=Retrieve   F12=Cancel
```

4.4.5 Step 5: Finish the Installation

If you entered `*YES,` then a message is displayed on the screen approximately
5 minutes to 15 minutes later to inform you that the installation is finished. Check the
job log for error messages by using the following command:

```
DSPJOBLOG
```

4.4.6 Step 6: Start the Gateway Subsystem

Enter the following command:

```
STRSBS instance_name/instance_name
```

where `instance_name` is the name that is used in "Step 3: Set Required
Parameters" on page 4-3.

Installation of the gateway is complete.

4.5 Postinstallation Steps

If you have more than one instance of the gateway, then repeat postinstallation steps
for each instance. Note that you need to install data dictionary support only once.
Refer to the table of installation choices in "Step 3: Set Required Parameters" for more
information on data dictionary support. Oracle recommends performing the following
postinstallation steps:

4.5.1 Journaling

Journaling is the process of recording changes that are made to files on the AS/400 in
order to ensure transactional consistency. Every table on the AS/400 that is going to be
changed by the gateway must be journaled. Changes to a file are recorded by a
`JOURNAL` in a `JOURNAL RECEIVER`. Journaling is not required for gateways that are
used for read-only purposes.

By default, the gateway runs under commitment control change (`*CHG`). This means
that:

- AS/400 files that are not changed by SQL operations do not need to be journaled.

- Users can see pending changes that have been input by others before the changes
  have been committed.

You can change the isolation level from the default setting, `*CHG`, to cursor stability,
*CS. With cursor stability:

- Users cannot see current changes until the changes have been committed.

- All files that are changed by SQL operations must be journaled.

For additional information about changing the isolation level, refer to "CHGORATUN, Change Initialization Parameters" on page 6-6.

For information about the `*RR` isolation level setting and for more information about `*CS` and `*CHG` isolation level settings, refer to the Isolation Level section of the IBM reference for DB2/400.

> **Note:** When creating a SQL collection, DB2/400 creates a journal and a journal receiver within that collection. Any files (tables) that are created within that collection are automatically journaled. For this reason, it is strongly recommended that you create a SQL collection before installing the Gateway and then install the Gateway into that SQL collection.

4.5.2 Step 1: Verify the Gateway Version

To verify installation and find out what version of the gateway is running, use the following command:

```
DSPPGM <instance_name>/GTW
```

where:

instance_name is the name that is given to the gateway when it is installed.

GTW is the name of the gateway program object (the gateway executable program).

The Text Description line resulting from the `DSPPGM` command informs you of the gateway version.

You can also find out what version of the gateway is running by looking in the job logs for a `RUNORAGxxx` job. Usually the job name is `RUNORAGTTP`.

4.5.3 Step 2: Verify Journaling

Journal all AS/400 files that are changed through the gateway. To start journaling for a file, use the `STRJRNPF` command. To use the `STRJRNPF` command, you must have previously created a journal and a journal receiver, as illustrated in the following example. Note that if you installed the gateway into a library that was created as a SQL collection, then the journal and journal receiver already exist. You do not need to explicitly journal files that are created in a library that is part of a DB2/400 SQL collection. You can journal many files to the same pair of journal and journal receiver.

```
CRTJRNRCV JRNRCV(instance/QSQJRNRCV)
CRTJRN JRN(instance/QSQJRN) JRNRCV(instance/QSQJRNRCV)
STRJRNPF FILE(library/file) JRN(instance/QSQJRN)
```

where:

instance is the name that is given to the gateway when it is installed.

library is the name of the library where the journal is located.

file is the name of the file that you are journaling.

> **Note:** If you attempt to change a file that is not journaled, then you may receive an ORA-28500 error (connection from ORACLE to a non-Oracle system returned this message), followed by a second error message that is generated by the target system or gateway (usually a `SQL 7008:filename in library not valid for this operation)`. In this case, enter `CRTJRNRCV` to create the journal receiver, and then enter `CRTJRN` to create the journal. Then, start journaling the specified file by using the `STRJRNPF` command.
>
> Refer to the Isolation Level section of the IBM reference for DB2/400 for more information about journaling and its parameters.

4.5.4 Step 3: Journal the ORACLE2PC File

This step is required only if:

- you are planning to issue a distributed update between the Oracle Database 10*g* server and the gateway

- any OS/400 stored procedures are executed through the gateway

- any OS/400 User Defined Function executed anywhere other than in a SQL `SELECT` statement. User Defined Functions executed in a SQL `SELECT` statement cannot cause any changes to the DB2/400 data.

Journal the `ORACLE2PC` physical file, which records two-phase commit transactions. You must journal this file before issuing a distributed transaction to the AS/400. To journal the file, enter:

```
STRJRNPF FILE(instance_name/ORACLE2PC) JRN(library/journal)
```

where:

instance_name is the name that is given to the gateway when it is installed.

library is the name of the library where the journal is located.

journal is the name of the journal.

4.5.5 Step 4: Configure Oracle Net

Configure Oracle Net on the Oracle Database 10*g* server. Refer to Chapter 5, "Oracle Net" for more information.

4.5.6 Step 5: Remove Observability

The gateway is shipped with IBM observability. Removing observability reduces gateway program object size by approximately 60 percent. If disk space is a concern and if you have no plans to upgrade to a higher version of OS/400, then Oracle recommends that you remove observability from the gateway.

To remove observability, perform the following steps:

1. Stop the gateway by entering:

   ```
   ENDSBS instance_name *IMMED
   ```

   where *instance_name* is the name that is given to the gateway when it is installed.

2. Use the SBMJOB command to submit a batch job using the QSECOFR ID. The SBMJOB syntax is:

```
SBMJOB CMD(CALL instance_name/REMOVEOBS PARM(instance_name))
```

4.5.7 Step 6: Raise the Gateway Priority Level

The gateway subsystem and listener default to an AS/400 priority level of 50 during the installation process. In many cases, a priority level of 50 is conducive to performing the functions that are provided by the gateway. However, if the AS/400 is running at a high CPU capacity with several jobs competing for CPU allocation, then you might need to raise the priority of the gateway subsystem and listener to meet your performance requirements.

> **Note:**   The gateway runs as a batch job. Raising the priority can impact the performance of other processes. Raising the priority of the gateway to a priority higher than 20 is not recommended. (Lower numbers represent higher priority levels.)

5

Oracle Net

Oracle Net is an Oracle product providing distributed database and processing capabilities. Generally, Oracle Net for AS/400 supports network communications between Oracle applications, Oracle servers, and Oracle gateways across different AS/400 systems or foreign operating systems. For product-specific information, refer to the *Oracle Database Net Services Administrator's Guide* and the *Oracle Database Net Services Reference*.

This chapter presents information about the Oracle Net architecture and how to configure and use Oracle Net for AS/400. It contains the following sections:

- Overview of Oracle Net on page 5-1

- Connecting to the AS/400 through TCP/IP on page 5-4

- Checklists for Configuring Oracle Net on page 5-5

- Configuring Oracle Net for TCP/IP AS/400 on page 5-6

- TCP/IP Connection Problems on page 5-8

- Resolving Connection Problems on page 5-8

- Troubleshooting TCP/IP Configuration on page 5-10

5.1 Overview of Oracle Net

Oracle Net is a required Oracle product supporting network communications between Oracle applications, Oracle servers, and Oracle gateways across different CPUs and operating systems. It also supports communication across different Oracle Databases and CPUs providing distributed database and distributed processing capabilities.

Oracle Net also allows applications to connect to multiple Oracle servers or gateways across a network, selecting from a variety of communications protocols and application program interfaces (APIs) to establish a distributed processing and distributed database environment.

A communications protocol is a set of implemented standards or rules governing data transmission across a network. An API is a set of subroutines providing a programming interface for application processes to the network environment.

5.1.1 Distributed Processing

Dividing processing between a front-end computer that is running an application and a back-end computer that is used by the application is known as distributed processing. Oracle Net enables an Oracle tool or application to connect to a remote computer containing an Oracle server or Oracle gateway.

5.1.2 Distributed Database

Several databases that are linked through a network and that appear as a single logical database are known as a distributed database. An Oracle tool running on a client computer or on an Oracle server running on a host computer can share and obtain information that is retrieved from other remote Oracle servers or Oracle gateways. Regardless of the number of database information sources, you might be aware of only one logical database.

5.1.3 Terminology for Oracle Net

The following terms are used to explain the architecture of Oracle Net for AS/400:

host: is the computer on which the database resides and the computer that runs the Oracle gateway. On AS/400, only a gateway can be running. The Oracle Database does not run on the AS/400.

client (task):   is the application using an Oracle Net driver to communicate with the Oracle Database server or gateway. A server is also considered to be a client if it initiates a connection with another Oracle Database server, or with an Oracle gateway.

protocol: is a set of standards or rules governing the operation of a communication link.

driver: is the part of Oracle Net that supports a given network protocol or communication method.

network: is a configuration of devices and software that are connected in order to interchange information.

5.1.4 Oracle Net for AS/400 Architecture

Oracle Net connections are established on the AS/400 through a listener. A listener receives incoming connections from Oracle Net clients and starts or transfers to a job on the AS/400 system.

On the AS/400, the Oracle Net TNS listener is used for TCP/IP.

Figure 5–1 shows communication between a client and the AS/400. Oracle Net on the AS/400 can accept TCP/IP connections.

Figure 5–1   Oracle Net Communication to the OS/400
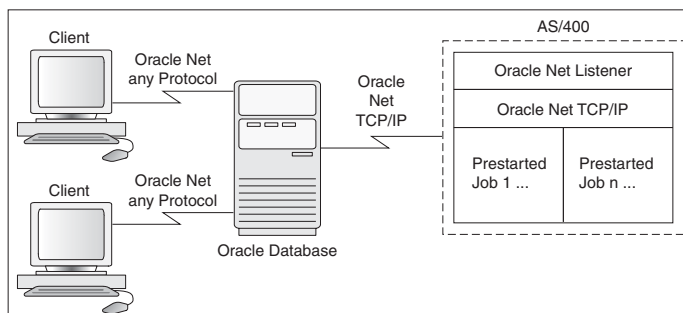
After the listener accepts the client connection, a batch job is started on the AS/400, or (more usually) a prestarted AS/400 batch job receives control. The batch job executes the gateway and sends a response back to the client. The response is data or a message. Each batch job is responsible for a client connection. Only one batch job is

executed for a single connection, and a single connection is associated with only one batch job.

To reduce the waiting time for connecting to gateway jobs, a set of prestarted gateway batch jobs are associated with a TNS listener. The number of prestarted jobs and the point at which additional jobs are started can be modified by the system administrator.

5.1.5 File Name Structure

Oracle Net refers to files in the following format:

```
basename.extension
```

where *basename* is the base portion of the name, and *extension* is the second part of the name.

An example of this form is SQLNET.ORA.

On the AS/400, the Oracle Net parameter files are installed in the gateway instance library using the following file name mapping:

- The base portion of the file name is mapped to a member name in an AS/400 file.

- The extension portion of the file name is mapped to an AS/400 file name.

For example, the SQLNET.ORA parameter file is mapped to member SQLNET in the ORA file that is located in the gateway instance library. Member names are referred to as:

```
file(member_name)
```

Some of the parameter values in ORA(SQLNET) can be changed with the CHGORANET command. For more information, refer to "CHGORANET, Change Network Parameters" on page 6-5.

For a list of the network files and members that are provided with the gateway, refer to Appendix A, "Oracle Net Files and Members".

5.1.5.1 Trace Files for Job Members

To enable tracing for the listener or server, use the CHGORANET command to change the values of the Listener trace level and Server trace level parameters. Possible trace level values are:

- ***OFF** (the default) provides no trace information. Tracing is not enabled.

- ***USER** provides the least detailed trace data.

- ***ADMIN** provides more detailed trace data.

- **16** provides comprehensive trace data.

> **Note:** Follow the advice of Oracle Support Services for setting trace levels.

Because trace files are large, you should use the least detailed trace level possible. For more information about changing the trace level parameters, refer to "CHGORANET, Change Network Parameters" on page 6-5.

When tracing is enabled, each job produces a trace file member in the TRC file with a member name in the following format:

SERV*xxxxxx* (for server jobs) or LIST*xxxxxx* (for listener jobs), where *xxxxxx* is the job number.

5.2 Connecting to the AS/400 through TCP/IP

Perform the following steps to configure your Oracle Database server computers for TCP/IP. Refer to the networking documentation for your Oracle Database server computer for more information about configuring it.

1. Step 1: Add a TCP/IP Connect Descriptor to tnsnames.ora

2. Step 2: Specify SID Names

5.2.1 Step 1: Add a TCP/IP Connect Descriptor to tnsnames.ora

Use the TCP/IP connect descriptor in the tnsnames.ora file on the Oracle Database 10*g* server:

```
connect_name = (DESCRIPTION=
                   (ADDRESS=(PROTOCOL=TCP)
                     (PORT=1521)
                     (HOST=host_name)
                   )
                   (CONNECT_DATA=(SID=csi_name))
                 (HS=)
                 )
```

where:

connect_name is the name of the connection. This name must be unique within the tnsnames.ora file. The *connect_name* corresponds to the value that is found with the USING keyword on a CREATE DATABASE LINK command.

TCP is the TCP protocol that is used for TCP/IP connections.

1521 is the default port number that is used by the Oracle Net listener on the AS/400. Change this value if you have changed the listener port number.

host_name is the TCP/IP host name of your AS/400 computer. It may be a DNS name, such as myas400.us.mycompany.com or an IP dotted value such as 10.2.3.7.

csi_name is the name of the communication side information (*CSI) file that is used by the listener to start the server. Usually, the *csi_name* is ORAGATE. Refer to "Step 2: Specify SID Names" on page 5-4 for more SID names.

HS= This keyword is mandatory. The right-hand side of the HS keyword is not needed.

5.2.2 Step 2: Specify SID Names

Oracle servers that are using TCP/IP to connect to a gateway must specify a system identifier (SID) name in the TNSNAMES.ORA file that is used to connect to the AS/400. The SID name is used on the AS/400 to start the server. The listener uses the SID name to reference a *CSI file. The *CSI file uses a TP_NAME to start the required CL program in the gateway server. Several SID names are configured in the shipped system and are listed in the following table.

Set the SID to ORAGATE in order to use a prestarted job. ORAGATE would give the best response time. Setting the SID to other values should be done in special situations as directed by Oracle support personnel.

Table 5–1   SID Names That Are configured in the Shipped System

SID	*CSI	TP_NAME	CL Program
ORAGATE	ORAGATE	ORAGWTPJ	RUNORAGTTP
ORAGTPJ	ORAGTPJ	ORAGWTPJ	RUNORAGTTP
ORAGTEJ	ORAGTEJ	ORAGWTEJ	RUNORAGTTE
ORAGTTC	ORAGTTC	ORAGWTTC	RUNORAGTTT

The following table provides the name and usage of each SID:

Table 5–2   SID Names and Their Usage

SID	Usage
ORAGATE	uses one of the prestarted TCP/IP jobs.
ORAGTPJ	is an alias for ORAGATE.
ORAGTEJ	starts a new gateway job.
ORAGTTC	starts a gateway job and a CPIC trace. It is also used for debugging the connection between the listener and the server.

Refer to "Viewing Jobs" on page 7-4 for instructions on viewing jobs and a description of the jobs that typically exist.

5.3 Checklists for Configuring Oracle Net

The following sections include checklists for configuring Oracle Net and for connecting to an Oracle server.

5.3.1 Checklist for Configuring Oracle Net for TCP/IP

- Step 1: Configure the Port Number
- Step 2: Define the Host Name
- Step 3: Verify the Host Name
- Step 4: Verify that the Listener is Started

5.3.2 Connecting Through TCP/IP (on the Oracle Server) Checklist

- Step 1: Add a TCP/IP Connect Descriptor to tnsnames.ora
- Step 2: Specify SID Names

5.3.3 Preliminary Step: Define a Physical Line

This step is required only if no physical connection currently exists between the Oracle Database 10*g* server and the AS/400.

To define the physical connection, use the CRTLINETH command to create an ethernet connection.

When you use the CRTLINETH command, you can set the AUTOCREATE CONTROLLER parameter to *YES (the AS/400 system default is *NO). This enables the AS/400 auto configuration feature.

If a line is already defined, then you can use the DSPLIND *line_name* command to display the line description parameters.

These line description parameters are used to configure the network on the computer where the gateway resides.

5.4 Configuring Oracle Net for TCP/IP AS/400

When you install the gateway, you are prompted for a listener port, through which TCP/IP connections will be established. By default, a listener job starts when the gateway subsystem starts. The listener monitors the TCP/IP port that you specified and accepts incoming connections directed to that port. After a connection is made, the listener transfers control to a server job by using the Oracle Net bequeath mechanism.

Internal process communication (IPC) between the listener and the server job is achieved through AS/400 local APPC devices. The listener then redirects the client connection to a randomly generated port that is assigned to the server job. This process is the Oracle Net inherit mechanism.

For more information about Oracle Net, refer to the *Oracle Database Net Services Administrator's Guide* and the *Oracle Database Net Services Reference*.

Figure 5–2 demonstrates a TCP/IP connection flow.

Figure 5–2 TCP/IP Connection Flow
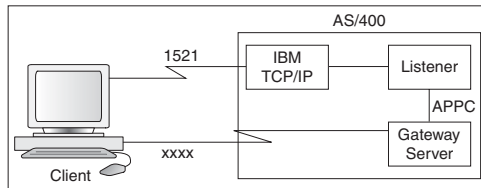
> **Note:**   You may need to refer to the IBM Information Center for information about TCP/IP concepts and terminology, setting up TCP/IP, and using TCP/IP. Search for "tcp/ip setup" in the Information Center. Correct configuration of TCP/IP on the AS/400 ensures that the gateway functions properly.

5.4.1 Configuring for TCP/IP

Perform the following steps to configure your TCP/IP connection between the Oracle Database 10*g* server and the AS/400:

Step 1: Configure the Port Number
Step 2: Define the Host Name
Step 3: Verify the Host Name
Step 4: Verify that the Listener is Started

5.4.2 Step 1: Configure the Port Number

If the gateway is already assigned to port 1521 or to another available port number, then go to "Step 2: Define the Host Name".

The gateway listener must be assigned a dedicated port number on which to listen for incoming connection requests. The default port number is 1521. This number can be changed during installation if you know that port 1521 is already used or if you prefer to use a different port number for any other reason. The port number may also be changed after installation by using the CHGORANET command.

You can find the values of port numbers that are currently active by using the AS/400 command NETSTAT *CNN. Each running instance requires a unique listener port number. Two gateway instances that are active at the same time cannot use the same listener port number. For more information, refer to "CHGORANET, Change Network Parameters" on page 6-5.

5.4.3 Step 2: Define the Host Name

To define the host name, perform the following steps:

1. Run the AS/400 CFGTCP command.

2. Select option 12, Change TCP/IP domain information. The panel in Example 5–1, "Change TCP/IP Domain (CHGTCPDMN)" is displayed:

Example 5–1   Change TCP/IP Domain (CHGTCPDMN)

```
                       Change TCP/IP Domain (CHGTCPDMN)

Type Choices, press Enter.

Host name...                        'AS400A'
Domain name...                      'US.ORACLE.COM'
Domain search list...              *DFT
Host name search priority...       *REMOTE   *REMOTE,  *LOCAL,  *SAME
Domain name server
  Internet address...              '140.24.88.144'
                                   '140.24.244.29'




                                                                 Bottom

F3=Exit  F4=Prompt  F5=Refresh  F10=Additional parameters  F12=Cancel
F11=How to use this dispay  F24=More Keys
```

1. Enter your domain name if the Domain name field is empty. Enter your host name if the Host name field is empty.

    Press Enter to save your changes and return to the option list panel. (The changes take effect after you restart the gateway.)

    The host name and the domain name combine to form the qualified host name. For example, AS400A combines with US.ORACLE.COM to form AS400A.US.ORACLE.COM as the qualified host name.

5.4.4 Step 3: Verify the Host Name

Verify that your host name is in your host name table by using the following AS/400 command:

```
PING host_name.domain_name
```

You can also use a PING command on the client-side Oracle Database server to verify that it can communicate with the OS/400.

> **Note:** The PING LOOPBACK command does not verify the host name.

5.4.5 Step 4: Verify that the Listener is Started

Use the Oracle Gateway Monitor to verify that the listener is started. If it is necessary to start the listener, then use the Oracle Gateway Monitor to do so, or start the listener manually with the following command:

```
STRORALSN
```

You can use the NETSTAT *CNN command to verify that the port that is assigned to the LISTENER is, in fact, being used by the LISTENER job in a given gateway instance. For more information about using the Oracle Gateway Monitor, refer to Chapter 8, "Administering the Gateway".

5.5 TCP/IP Connection Problems

If you cannot access AS/400 data by using Oracle Net after configuring TCP/IP, then check to see whether the AS/400 is reachable by pinging the AS/400 from an Oracle Database server host from the operating system prompt, or use a similar command from the shell prompt:

```
PING host_name.domain_name
```

where *host_name* is the name that identifies the AS/400.

If you cannot resolve your connection problem, then refer to "Message and Error Code Processing" on page 10-1 for more information.

5.6 Resolving Connection Problems

You may experience the following error types: AS/400 data is not reachable, errors ORA-28509, ORA-28511, ORA-28500, ORA-12154, or a hanging condition.

5.6.1 AS/400 Connection Problems

The following suggestions may resolve your connection problems:

5.6.1.1 The AS/400 is not reachable:

Review your Oracle Net configuration.

5.6.1.2 The AS/400 is reachable, but you cannot access AS/400 data:

Verify that the definition for the host name (from "Step 2: Define the Host Name" on page 5-7 in the "Configuring for TCP/IP" section) contains the fully qualified name.

Without a fully qualified `host_name.domain_name,` Oracle Net cannot resolve the address, and you will be unable to access AS/400 data.

5.6.2 ORA-28509

The following are some causes of the error.

5.6.2.1 `HS=` is missing from your tnsnames.ora file:

HS= is a parameter that you must specify in the tnsnames.ora file, or you will not be able to use version 10 of the gateway. The correct syntax is shown at "Connecting to the AS/400 through TCP/IP" on page 5-4. If HS= is missing from the entry in your tnsnames.ora file, then some symptoms to look for are a hanging connection and errors ORA-2068 and ORA-3114. Search Metalink for Note 136294.1, which provides more details about this topic.

5.6.2.2 `HS=` Not in Correct Place:

HS= is specified in tnsnames.ora, but it is not specified in the correct place, or the number of parentheses is uneven (an odd number due to missing or extra parentheses).

HS= needs to be set up outside the CONNECT_DATA specification. If you misplace HS=, or if you do not have the correct parenthesis specification, then you can get error ORA-28509.

Other symptoms (same as when HS= is missing from your tnsnames.ora file) are hanging connection and errors ORA-2068 and ORA-3114.

5.6.2.3 TCP/IP Configuration:

TCP/IP configuration under AS/400 ("Option 10 of CFGTCP: Work with TCP/IP Host Table Entries" on page 5-10) is not configured as anticipated by the gateway listener.

The gateway listener requires `host_name.domain_name` in the TCP/IP host table entries. If you do not specify this, then you will not be able to get any response back from the gateway. You will be able to see the connection in the AS/400, but no data can be transferred.

5.6.2.4 Gateway Listener is Not Up:

By default, the gateway listener is automatically started when you start the gateway instance. You can manually start the listener by executing the command: STRORALSN.

5.6.3 ORA-28511

The following are some causes of the error.

5.6.3.1 Gateway or Listener is Not Up:

Check to see that the gateway is up and running correctly and is properly configured. Check to see if the listener is up.

5.6.3.2 Set Fully Qualified `host.domain` Name

Ensure that a fully qualified `host.domain` name has been set on the AS/400. Use CFGTCP to do this. Use Option 10 on page 5-10 and Option 12 on page 5-10 to set a fully qualified `host.domain` name. Use ping to verify that the `host.domain` name matches the IP address of the host name.

5.6.4 ORA-28500

Check any accompanying message from the AS/400 to determine what the problem might be, or check the AS/400 job log (`RUNORAGTTP` job log). For example, message SQL-7008 indicates that the file is not journaled.

5.6.5 ORA-12154

This error indicates that a missing or incorrect tnsnames file has been specified, or that an incorrect spelling or service name has been specified in the tnsnames file. This is generally an Oracle Net issue, not a gateway issue.

5.7 Troubleshooting TCP/IP Configuration

You may encounter errors during the test phase of the gateway installation even though you are able to issue a `PING`. If you do encounter such errors, then please check the panels that were used to define the host name. Specifically, check the values presented when using option 12 of the `CFGTCP` command.

5.7.1 Option 10 of CFGTCP: Work with TCP/IP Host Table Entries

Table 5–3   TCP/IP Host Table Entries

Option	Internet Address	Host Name
	10.10.5.40	AS400 AS400.US.ORACLE.COM
	127.0.0.1	LOOPBACK LOCALHOST

Enter Internet addresses and host names. Host names in the "Work with TCP/IP Host Table Entries" panel must include both the unqualified and the qualified names (the one that includes the domain name).

Option 10 depends upon Option 12. Option 12 defines your host name as well as your domain name. Under Option 10, you must enter your host name and your `hostname.domain_name`. This may seem redundant, but if the gateway is not configured correctly for TCP/IP, then the gateway will not make a connection, regardless of the output from the `PING` command. Any changes to `CFGTCP` will require you to restart the gateway.

TCP configuration is documented in "Configuring Oracle Net for TCP/IP AS/400" on page 5-6 in this chapter.

5.7.2 Option 12 of CFGTCP: Change TCP/IP Domain (CHGTCPDMN)

Table 5–4   Change TCP/IP Domain

Option	Host name	Domain name
	'AS400A'	'US.ORACLE.COM'

The host name and the domain name combine to form the qualified host name. For example, `AS400A` (as illustrated in Option 12) combines with "US.ORACLE.COM" to form the qualified host name AS400A.US.ORACLE.COM.

6

Configuring the Gateway

After installing the gateway, you can run gateway commands and change gateway parameters.

This chapter contains the following sections:

- Gateway Commands on page 6-1
- Setting Optional Parameters on page 6-10

6.1 Gateway Commands

All gateway parameters are changed with gateway commands, which are accessed through a menu system. These commands and their menus are described in this section.

The gateway comes with commands to do the following tasks:

- copy (clone) the gateway
- change the most common gateway parameters
- change the level of tracing and debugging

All commands can be used after a gateway is installed. The following table summarizes each command and its purpose. The Menu Choice is used when executing the `CMDORAGTW` menu.

Table 6–1   Command Summary

Menu Choice	Command Name	Purpose
1	CRTORAGTWI	creates a copy of an installed gateway version.
		You can copy a version as many times as needed.
		After making a copy, use the other gateway commands to change the parameters of the new copy.
2	CHGORANET	changes the values of network parameters.
3	CHGORAPJE	changes the parameter values of prestarted jobs.
4	CHGORATUN	changes the gateway initialization parameters.
5	CHGGTWDBG	sets or changes values for debugging parameters
		(Use only under the guidance of a representative from Oracle Support Services.)
6	CRTORADDB	submits a batch job to create Oracle data dictionary views based on the DB2/400 system catalog.

Table 6–1   (Cont.) Command Summary

Menu Choice	Command Name	Purpose
7	CHGRECOPRF	changes the User Profile name, or the User Profile password, or both, for the User Profile that is associated with transaction recovery. For password precautions, refer to "CHGRECOPRF, Change Recovery Profile Parameters" on page 6-9.

6.1.1 Running the Commands

All gateway commands can be accessed through a main menu. To invoke the main menu, enter:

```
ADDLIBLE  instance_name
GO CMDORAGTW
```

where: *instance_name* is the name given to the gateway when it was installed.

The main menu panel that is illustrated in Example 6–1, "CMDORAGTW Oracle Commands Panel" is displayed:

Example 6–1   CMDORAGTW Oracle Commands Panel

```
CMDORAGTW                    Oracle Commands
                                                        System:  AS400A
Select one of the following:

 1. Create instance
 2. Change network settings
 3. Change prestart job settings
 4. Change gateway initialization settings
 5. Change debugging options
 6. Change Oracle Data Dictionary objects (batch)
 7. Change Recovery Profile Parameters


Selection or command
===>

F3=Exit  F4=Prompt  F9=Retrieve  F12=Cancel
```

Enter the choice number (1, 2, 3, 4, 5, 6, or 7) or the corresponding command name:

- CRTORAGTWI

- CHGORANET

- CHGORAPJE

- CHGORATUN

- CHGGTWDBG

- CRTORADDB

■ CHGRECOPRF

After you have made your selection, press Enter to continue. The panel for the command is displayed. Command panels are described under the individual commands.

> **Note:** For more information about these choices, move the cursor to the value on the panel and press PF1.

6.1.2 CRTORAGTWI, Copy the Gateway

You can have as many copies of a gateway instance on your system as you want. After you have installed a release 10 gateway, if you need another instance of the gateway, before you issue the CRTORAGTWI command, then you must:

■ Shut down the instance you are about to copy. Use the ENDSBS command with the instance name as the operand to perform the shutdown operation.

■ Ensure that you log on with a user profile that has the *SECADM,*JOBCTL, *ALLOBJ, and *SYSCFG special authorities. The user profile QSECOFR as distributed by IBM has these authorities.

■ Ensure that a library or collection with the same name as the instance name to be created does not already exist. If it does exist, then it must be empty or must have only those objects that would result from a SQL CREATE COLLECTION command.

After entering 1 at the main menu panel, or CRTORAGTWI, the panel in Example 6–2, "Create Oracle Database Gateway Panel" is displayed.

Example 6–2   Create Oracle Database Gateway Panel

```
                  Create Oracle Database Gateway V10.2.0.1.0
                                                   System:   AS400A
 Type choices, press Enter.

  Existing instance name ....       ORACLE        Name (up to six characters)
  Instance name to create ....                    Name (up to six characters)




 ===>

 F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

Enter the name of the new instance to be created and press Enter to continue. The panel in Example 6–3, "Create Oracle Database Gateway Panel, with New Values" appears.

Example 6–3   Create Oracle Database Gateway Panel, with New Values

```
                  Create Oracle Database Gateway V10.2.0.1.0
```

```
                                                  System:  AS400A
Type choices, press Enter.

 Existing instance name .......    ORACLE       Name (up to six characters)
 Instance name to create ......    ORANEW       Name (up to six characters)
 Instance password ............    ORANEW       1-10 characters
 Recovery user profile ........    ORANEW       Name
 Recovery user password .......    ORANEW       1-10 characters
 Prestart jobs ................    *YES         *YES, *NO
 TCP/IP port number ...........    1521         1024-65534
 Auxiliary storage pool id ....    1            1-16
 Install Data Dictionary
   Support ....................    *NO          *YES, *NO


===>

F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

The first two entries are set from the previous screen and cannot be changed:

`Existing instance name` is a name from one character to six characters long. The default is whatever is in the `ORA_HOME` data area. If you `ADDLIBLE` `instance_name,` then the value in the `ORA_HOME` data area will be the existing instance name.

`Instance name to create` is a name from one character to six characters long.

You must fill in the following choices:

`Instance password` is the password for the user ID that will be created for the new user profile. The user ID has the same name as the newly-created instance name. You should change this password for security reasons.

`Recovery user profile` is a profile name that you enter, or you can use the default of the newly created gateway instance name.

`Recovery user password` is a password for the recovery user profile. The default is the newly created gateway instance name. You should change this for security reasons. Use the OS/400 `CHGUSRPRF` command to make the change. Then, use the `CHGRECOPRF` command (refer to "CHGRECOPRF, Change Recovery Profile Parameters" on page 6-9).

`Prestart jobs` prestarts the TCP/IP jobs. By default, two TCP/IP jobs are prestarted. Use *YES to prestart the TCP/IP server jobs. Use `*NO` to prestart `NO` TCP/IP server jobs.

You can change how many jobs are prestarted by changing the value of the initial number of TCP/IP jobs parameter with the `CHGORAPJE` command. For more information, refer to "CHGORAPJE, Change Prestarted Job Parameters" on page 6-6.

`TCP/IP port number` will be the same as the port number of the cloned instance. You should enter a different port number because each instance requires a unique port number.

`Auxiliary storage pool id` uses the default of 1, or you can enter another ID if you have additional auxiliary storage pools defined.

Install Data Dictionary Support Enter *YES if you wish to reinstall the Data Dictionary support. The default is *NO. If you are copying (cloning) a gateway, then the Data Dictionary was most likely installed when the copied (cloned) gateway itself was installed. In that case, you do not need to reinstall the Data Dictionary.

6.1.3 CHGORANET, Change Network Parameters

After entering 2 at the main menu panel or by entering the CHGORANET command, enter the appropriate instance name and press Enter. The panel in Example 6–4, "Change Oracle Network Parameters Panel" appears. Except when you are changing the value of the TCP/IP port number, you should use CHGORANET only under the guidance of a representative from Oracle Support Services. Enter the new values and press Enter to continue. The new values do not take effect until you shut down and restart the gateway that was specified in the Existing instance name parameter of the command.

Example 6–4   Change Oracle Network Parameters Panel

```
                        Change Oracle Database Gateway
                                                          System:  AS400A
Type choices, press Enter.

 Existing instance name .......    ORACLE       Name
 TCP/IP port number ...........    1521         1024-65534
 Client trace level ...........    *OFF         *OFF, *USER, *ADMIN, 16
 Listener trace level .........    *OFF         *OFF, *USER, *ADMIN, 16
 Server trace level ...........    *OFF         *OFF, *USER, *ADMIN, 16
 Regenerate files .............    *NO          *YES, *NO




===>

F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

6.1.3.1 Changing the ORA(SQLNET_CHG) File

Some network parameters are documented in the ORA(SQLNET) file but are not displayed on the CHGORANET panel. These parameters cannot be changed directly by editing the ORA(SQLNET) file. They must be changed in the ORA(SQLNET_CHG) file:

1. Use an OS/400 editor (e.g., SEU) to change parameter values in ORA(SQLNET_CHG).

2. On the OS/400 command line, use the CHGORANET command, or use the GO CMDORAGTW command and specify option 2.

3. Enter the instance name of the gateway, and press Enter.

4. Specify *YES on the "Regenerate Files" line, and press Enter.

The ORA(SQLNET) file now reflects the values that are specified in the ORA(SQLNET_CHG) file.

6.1.3.2 Changing the ORA(LISTEN_CHG) File

Some network parameters are documented in the ORA(LISTENER) file but not displayed on the CHGORANET panel. These parameters cannot be changed directly by editing the ORA(LISTENER) file. They must be changed using the ORA(LISTEN_CHG) file:

1. Use an OS/400 editor (e.g., SEU) to change parameter values in ORA(LISTEN_CHG).

2. On the OS/400 command line, use the CHGORANET command, or use the GO CMDORAGTW command and specify option 2.

3. Enter the instance name of the gateway, and press Enter.

4. Specify *YES on the "Regenerate Files" line, and press Enter.

The ORA(LISTENER) file now reflects the values that are specified in the ORA(LISTEN_CHG) file.

6.1.4 CHGORAPJE, Change Prestarted Job Parameters

On the OS/400 command line, use the CHGORAPJE command, or use the GO CMDORAGTW command, specify option 3, and press Enter. The panel in Example 6–5, "Change Oracle Prestart Parameters Panel" appears. Enter the new values and press Enter to continue.

Example 6–5   Change Oracle Prestart Parameters Panel

```
                     Change Oracle Prestart Parameters
                                                    System:  AS400A
Type choices for prestart jobs, press Enter.

 Existing instance name ............    ORACLE         Name
 Start TCP/IP jobs .................    *YES           *SAME, *YES, *NO
 Initial number of TCP/IP jobs......    2              1-1000, *SAME
 TCP/IP threshold ..................    1              1-1000, *SAME
 Additional number of TCP/IP jobs ..    2              0-999, *SAME
 Maximum number of TCP/IP jobs .....    *NOMAX         *SAME, *NOMAX
 Start TCP/IP listener .............    *YES           *SAME, *YES, *NO



===>

F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

For information about what values you should enter for these parameters, move the cursor to the value on the panel and press PF1.

The new values do not take effect until you shut down and restart the gateway that is specified in the Existing instance name parameter of the command.

6.1.5 CHGORATUN, Change Initialization Parameters

On the OS/400 command line, use the CHGORATUN command, or use the GO CMDORAGTW command and specify option 4. Then press Enter. The panel in Example 6–6, "Change Oracle Gateway Initialization Parameters (first page of panel)"

appears. CHGORATUN uses two panels. Refer to Example 6–6, "Change Oracle Gateway Initialization Parameters (first page of panel)" and Example 6–7, "Change Oracle Gateway Initialization Parameters (second page of panel)".

> **Note:** CHGORATUN can be run only while you are signed on as the user profile corresponding to the instance ID, or as a user profile that has *SECADM and *ALLOBJ special authorities. (QSECOFR as distributed by IBM has such authorities).

For information about what values you should enter for these parameters, move the cursor to the value on the panel and press PF1.

Enter the new values on the panels. Before entering values for the DATABASE DOMAIN, GATEWAY NATIONAL LANGUAGE, RPC FETCH REBLOCKING, and RPC FETCH SIZE gateway initialization parameters, read the following notes:

Table 6–2   Initialization Parameter Notes

Parameter	Notes	Default
DATABASE DOMAIN	If the value for the GLOBAL_NAME parameter is set to TRUE in the Oracle database INIT.ORA file, then the value that you enter in this field must exactly match that specified for the DB_DOMAIN parameter in the Oracle database INIT.ORA file.	WORLD
Gateway Language	The GATEWAY NATIONAL LANGUAGE line no longer appears in the CHGORATUN screen, but NLS_NCHAR actually still does exist for other reasons.	AMERICAN_AMERICA. WE8EBCDIC37
RPC FETCH REBLOCKING and RPC FETCH SIZE	If the RPC FETCH REBLOCKING parameter is set to *YES (the default), then the block size of the buffer for SELECT statements is determined by the value of the RPC FETCH SIZE parameter. The recommended value for Oracle Database Gateway for DB2/400 is 40,000. The RPC FETCH SIZE parameter defines the maximum number of bytes that are sent with each fetch between the gateway and the Oracle database. Each fetch block may contain multiple rows of data.	*YES 40 000
V4 GRAPHIC compatibility mode and UCS-2 support	Before deciding on a value for this parameter, refer to "DB2/400 GRAPHIC Support" on page 9-16 for more information.	*NO

Example 6–6   Change Oracle Gateway Initialization Parameters (first page of panel)

```
_____
             Change Oracle Gateway Initialization Parameters
                                              System:  AS400A
Type choices, press Enter.

Existing instance name .........   ORACLE           Name
Database Domain ................   WORLD
Database Name...................   ORACLE
Array block size................   100              0-32767
Gateway language ...............   american_america.we8ebcdic37
                                              Language ID (NLS_LANG)
V4 Graphic and UCS-2
  Compatibility mode ...........   *NO              V4 Graphic and UCS-2
```

```
                                            Compatibility mode
                                            (ORAGRAPH4)
Maximum Date ...................             ORA_MAX_DATE
Option for CCSID=65535 fields    *BITDATA    *BITDATA, *CHARDATA
User Profile CCSID .............  *SYSVAL    *SAME, *SYSVAL, *HEX, CCSID
                                              value

===>                                                     More...


F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

Example 6–7   Change Oracle Gateway Initialization Parameters (second page of panel)

```
            Change Oracle Gateway Initialization Parameters


                                                System:  AS400A
Type choices, press Enter.

Change Isolation Level........  *CHG          *CHG, *CS, *RR
Set gateway for READ-ONLY.....  *NO           *YES, *NO
Maximum Number Cursors........  200           50-200
RPC Fetch Reblocking .........  *YES          *YES, *NO
RPC Fetch Size ...............  40000         4000-50000




====>                                                    Bottom

F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

The new values do not take effect until you shut down and restart the gateway that is specified in the `Existing instance name` parameter of the command.

6.1.6 CHGGTWDBG, Change Debugging Parameters

This command is used for isolating the cause of a suspected gateway problem. Use CHGGTWDBG only under the guidance of a representative from Oracle Support Services.

On the OS/400 command line, use the CHGGTWDBG command, or use the GO CMDORAGTW command and specify option 5, then press Enter. Enter the appropriate instance name and press Enter. The panel in Example 6–8, "Change Oracle Gateway Debugging Option Panel" appears. For more information about the value choices, move the cursor to the value on the panel and press PF1.

Example 6–8   Change Oracle Gateway Debugging Option Panel

```
            Change Oracle Gateway Debugging Options
                                        System:  AS400A
```

```
Type choices for debugging options, press Enter.

Existing instance name ..........    ORACLE          Name
Use gateway debugging version ....   *NORMAL         *NORMAL, *DEBUG
Gateway pause during job start ...   *NORMAL         *NORMAL, *PAUSE
Gateway show GETENV messages .....   *NORMAL         *NORMAL, *YES
Gateway continue after error .....   *NORMAL         *NORMAL, *CONTINUE
Gateway hang on error ...........    *NORMAL         *NORMAL, *HANG
Gateway hang time in minutes .....   *DAY            *NORMAL, 1-10080, *HOUR
                                                     *DAY, *WEEK
Gateway trace level .............    0               0-255
Listener pause during job start ..   *NORMAL         *NORMAL, *PAUSE
Listener show GETENV messages ....   *NORMAL         *NORMAL, *YES
Listener continue after error ....   *NORMAL         *NORMAL, *CONTINUE
Listener hang on error ...........   *NORMAL         *NORMAL, *HANG
Listener hang time in minutes ....   *DAY            *NORMAL, 1-10080, *HOUR
                                                     *DAY, *WEEK


===>                                                            Bottom

F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

After entering the new values, press Enter to continue. The new values do not take effect until you shut down and restart the gateway specified in the `Existing instance name` parameter of the command.

6.1.7 CRTORADDB, Create Data Dictionary Views

This command submits a batch job to create Oracle data dictionary views of the system catalog. If you are going to run an application such as Oracle Developer, then you need to create data dictionary views. Therefore, if you create these views when originally installing the gateway, then you should not need to create them again.

On the OS/400 command line, use the `CRTORADDB` command, or use the `GO CMDORAGTW` command and specify option 6. Then press Enter. Enter the appropriate instance name and press Enter. A batch job is submitted. No additional panel appears.

The Oracle Data Dictionary views of the system catalog are used by all Oracle Gateway instances on that AS/400. One copy of these views is on each AS/400.

6.1.8 CHGRECOPRF, Change Recovery Profile Parameters

This command changes the OS/400 User Profile name or password, or both, for the User Profile name or password that the gateway uses when directed to perform transaction recovery by the Oracle server. Transaction recovery is necessary when any failure occurs during a distributed transaction. The User Profile name and password are created by the OS/400 `CRTUSRPRF` command or are changed by the OS/400 `CHGUSRPRF` command.

Example 6–9   Change Recovery Profile Parameters Panel

```
                    Change Oracle Recovery Profile Parameters
                                                      System:   AS400A
Type choices, press Enter.
```

```
Existing instance name . . . .    ORACLE        Name
Recovery Profile Name  . . . .    ORACLE        Profile Name
Recovery Profile Password  . .                  Password
Confirmation of Recovery
  Profile Password . . . . . .                  Password




F1=Help  F4=Prompt  F9=Retrieve  F12=Cancel
```

Enter the User Profile name in the Recovery Profile Name field, or leave it as it is. Enter the password for the User Profile name on the next two lines. The password is stored in an encoded form. Because this password is no longer visible as plain text, extra care may be required to keep the recovery password synchronized between the AS/400 computer and the gateway. Refer to the following note.

> **Caution:**   You must keep the gateway values for recovery user profile and password synchronized with the OS/400 values for recovery user profile and password. The gateway recovery user ID and password must be valid to the AS/400 at all times. If the recovery password is not valid (if it expires, for example), then when an in-doubt transaction occurs, the Oracle Database alert log will most likely show an ORA-1017 (invalid user ID or password) error. Use the CHGRECOPRF command to change the recovery profile parameters (including recovery user ID and password).

6.2 Setting Optional Parameters

You can change the values of optional gateway parameters after the product is installed by using the gateway commands. Three commonly changed parameters are:

- `Array block size` in the `BLOCKSIZE` data area. Refer to "Retrieving Data" for more information.

- `Default character conversion` in the `ORARAW` data area. Refer to "Data Conversion" for more information.

- `Set gateway for READ-ONLY` for configuring the gateway with read-only capabilities. Refer to "Read-Only Gateway" for more information.

6.2.1 Retrieving Data

The gateway can retrieve multiple rows from a table or view with a single fetch. The gateway uses the `BLOCKSIZE` data area to determine the number of rows to retrieve. These conditions apply for the `BLOCKSIZE` data area:

- If the `BLOCKSIZE` data area is set to 0, then no block retrieval is performed. This is similar to setting `BLOCKSIZE` to `1`.

- If the `BLOCKSIZE` data area is set to `n`, then the gateway retrieves `n` rows from DB2/400 in a single fetch, where `n` is a value from `1` to `32767`.

- If the `BLOCKSIZE` data area does not exist, then the gateway will retrieve one row per fetch from DB2/400.

> **Note:** For performance reasons, Oracle recommends that the
> `BLOCKSIZE` data area be set between `10` and `100`.

The gateway uses a default value of 100 for the `BLOCKSIZE` data area. The
`BLOCKSIZE` value is used for substitution for host variable `N` in a SQL `FETCH`
statement such as:

```
FETCH CN for :N ROWS...
```

To change this default value, use the gateway command `CHGORATUN`. After displaying
the panel for `CHGORATUN`, enter a new value for `Array block size.` Refer to
"CHGORATUN, Change Initialization Parameters" on page 6-6 for information about
changing the setting for `Array block size.`

6.2.2 Data Conversion

The default coded character set identifier (system value `QCCSID`) for the AS/400
system is `65535`. This CCSID value indicates to the gateway that character data in a
column with such a CCSID is not to be converted and is to be treated as bit data.

The line entitled "Option for CCSID=65535 fields" on the "Change Oracle Gateways
parameters" panel (use the `CHGORATUN` command) specifies how the gateway is to
handle the "For Bit Data" and "CCSID=65535" fields. If the specification is `*BITDATA`,
then the fields are treated as binary data and no translation occurs. If `*CHARDATA` is
specified, then the fields are treated as if they were in the character set ID in which the
gateway runs. When using the `CHGORATUN` command, the "User Profile CCSID" line
specifies the character set ID in which the gateway runs. The `ORARAW` data area is used
to hold the data conversion specification.

6.2.3 Read-Only Gateway

The gateway can be configured with read-only capabilities. The read-only option may
provide improved performance and security, based on your configuration and
parameter selections. The `READONLY` data area controls whether the gateway is
enabled in this mode. The default setting for the read-only feature is `*NO`. You can
change the value of this environment parameter using the `CHGORATUN` command.

If you enable the read-only feature by changing the setting to `*YES`, then only queries
(`SELECT` statements) are allowed to DB2/400. The capabilities which control whether
updates are allowed through the gateway are not enabled. These capabilities include
insert, update, delete, and stored procedure support (pass through SQL, DB2/400
stored procedures, but not DB2/400 User Defined Functions). Statements attempting
to modify records at the gateway site are rejected.

Oracle recommends that you do not routinely switch between settings of the read-only
parameter. If you need both update and read-only functionality, then you should
install two separate instances of the gateway with different read-only settings.

Refer to "CHGORATUN, Change Initialization Parameters" on page 6-6 for
information about changing the `READONLY` setting.

7

Using the Gateway

After installing the gateway, you can administer database links, access the gateway, access AS/400 file members, perform distributed queries, and copy data between the Oracle database and the AS/400.

This chapter contains the following sections:

- Database Link Behavior on page 7-1
- Using DB2/400 Cursors on page 7-3
- Using the Synonym Feature on page 7-3
- Accessing the Gateway on page 7-4
- Accessing OS/400 File Members on page 7-5
- Performing Distributed Queries on page 7-6
- Replicating in a Heterogeneous Environment on page 7-8
- Copying Data from Oracle Server to DB2/400 Server on page 7-8
- Copying Data to Oracle Server from DB2/400 Server on page 7-9

7.1 Database Link Behavior

A connection to the gateway is established through a database link when it is first used in a gateway session or transaction. In this context, connection refers to the connection between the Oracle database and the gateway. The connection remains established until the session ends. Another session or user can access the same database link but will get a different connection to the gateway and DB2/400 database.

Connections to the DB2/400 database might be limited by factors that include memory, gateway parameters, or DB2/400 server resources.

The database and application administrators of a distributed database system are responsible for managing the necessary database links defining paths to the gateway.

Database links are discussed in detail in the *Oracle Database Administrator's Guide*. Information for using database links with the gateway is given here in this manual.

7.1.1 Creating Database Links

To create a database link in your Oracle Database and to define a path to the gateway, use the `CREATE DATABASE LINK` statement. The `CONNECT TO` clause specifies the remote user ID and password to use when creating a session in the gateway. If you do not specify a user ID and password in the `CONNECT TO` clause, then the Oracle logon user ID and password are used.

The USING clause specifies a `tnsnames.ora` connect descriptor.

7.1.2 Creating Database Links Using Oracle Net

Oracle Net is required. The following syntax creates a database link to access information in the DB2/400 database using Oracle Net:

```
CREATE DATABASE LINK dblink
       CONNECT TO userid IDENTIFIED BY password
       USING 'tns_name_entry';
```

where:

dblink   is the complete database link name (such as gateway).

userid   is the user ID that is used to establish a session in the remote AS/400 system. It must be authorized to any table or file on the DB2/400 server that is referenced in the SQL commands. The user ID cannot be longer than ten characters.

password   is the password that is used to establish a session in the remote database. This must be a valid OS/400 server password. The password cannot be longer than ten characters.

tns_name_entry   specifies the Oracle Net TNS connect descriptor that is used to identify the gateway subsystem.This is identical to the CONNECT_NAME that is found in entries in the `tnsnames.ora` file. Refer to "Step 1: Add a TCP/IP Connect Descriptor to tnsnames.ora" on page 5-4.

7.1.3 Closing Database Links

After being used, a database link remains open for the duration of the gateway session. If you want to close a database link during a session, then you can do so with the ALTER SESSION CLOSE DATABASE LINK dblink statement.

7.1.4 Accessing Data through Database Links

DB2/400 tables, views, and synonyms that are available to the user ID that is specified in the CONNECT TO clause can be accessed with the following syntax:

```
SELECT * FROM table@gateway
```
or
```
SELECT * FROM user.table@gateway
```

The CONNECT TO user ID provides implicit qualification for unqualified tables. For example:

```
SELECT * FROM EMP@gateway
```

resolves to SCOTT.EMP on DB2/400 if the CONNECT TO user is SCOTT. If no CONNECT TO statement is defined with the database link, then the Oracle user ID that is using the database link is used as the implicit qualifier.

Oracle strongly recommends that your table or view specifications (or both) always be qualified with the table or view owner, as in the following example:

```
SELECT * from user.table@gateway.
```

7.1.5 Dropping Database Links

You can drop a database link with the DROP DATABASE LINK statement. For example, to drop the public database link named dblink, enter the following statement:

```
DROP PUBLIC DATABASE LINK dblink;
```

Do not drop a database link if it might be required to resolve an in-doubt distributed transaction. Refer to the *Oracle Database Administrator's Guide* for additional information about dropping database links.

7.1.6 Examining Available Database Links

The data dictionary of each Oracle Database stores the definitions of all the database links in that database. The USER_DB_LINKS data dictionary view shows the database links that are defined for a specific Oracle Database user. The ALL_DB_LINKS data dictionary views show all defined database links, both public and private. The user has access to all these views. The DBA_DB_LINKS dictionary view, which is accessible only to users with DBA authorization, shows all database links that are defined in the Oracle Database instance.

7.1.7 Limiting the Number of Active Database Links

You can limit the number of connections from a user process to remote databases with the INIT.ORA parameter OPEN_LINKS. This parameter controls the number of remote connections any single user process can use concurrently with a single SQL statement. Refer to the *Oracle Database Administrator's Guide* for additional information about limiting the number of active database links.

7.2 Using DB2/400 Cursors

The maximum number of DB2/400 cursors that the gateway can open per Oracle Database session is 200. Although the gateway can open 200 cursors, other Oracle Database or DB2/400 limits might affect how many cursors can actually be opened for a specific application.

Use the CHGORATUN command to change the maximum number of cursors.

7.3 Using the Synonym Feature

You can provide complete data, location, and network transparency by using the synonym feature of the Oracle Database 10*g* server. When a synonym is defined, you do not need to know the underlying table or network protocol that is being used. A synonym can be public, which means that all users can make reference to the synonym. A synonym can also be defined as private, which means that every user must have a synonym defined to access the underlying DB2/400 table. Refer to the Oracle Database 10*g* server documentation for details about the synonym feature.

The following statement creates a system wide synonym named EMPDB2 in the Oracle Database for the SCOTT.EMP file in the DB2/400 server:

```
CREATE PUBLIC SYNONYM EMPDB2 FOR SCOTT.EMP@gateway
```

Only those users with database administrator authority can create public synonyms. You can use a similar statement to create a private synonym if you do not have database administrator authority:

```
CREATE SYNONYM EMPDB2 FOR SCOTT.EMP@gateway
```

7.4 Accessing the Gateway

To access the gateway, complete the following steps on the Oracle Database 10*g* server:

1. Log in to the Oracle Database 10*g* server.

2. Create a database link to the AS/400 database with the following syntax:

    ```
    CREATE DATABASE LINK AS400
       CONNECT TO userid IDENTIFIED BY password
       USING 'as400'
    ```

3. Retrieve data from the AS/400 database using one of the following methods:

    ■   If the CONNECT TO clause of the database link has specified ORACLE as the user ID, then this query retrieves data from the EMP file in the ORACLE library, using the name ORACLE as the AS/400 user profile:

        ```
        SELECT * FROM EMP@AS400
        ```

    ■   This query retrieves the EMP file in the library or collection CORPDATA, using the name ORACLE as the AS/400 user profile. The ORACLE OS/400 user profile must have the appropriate AS/400 privileges to access the CORPDATA.EMP file:

        ```
        SELECT * FROM CORPDATA.EMP@AS400
        ```

        The following messages are displayed if insufficient privileges were granted to Oracle user:

        ```
        ERROR at line 1:
        ORA-28500:  Connection from Oracle to non-Oracle system returned this
        message
        SQL0551:    Not authorized to object EMP in CORPDATA type *FILE
        ORA-02063:  preceding 2 lines from AS400
        ```

4. Update data on the AS/400.

5. Before attempting to update a file on the AS/400, ensure that the file is currently being journaled. If the file is not journaled, then the following messages are displayed:

    ```
    ORA-28500:  Connection from Oracle to non-Oracle system returned this message
    SQL7008:    EMP in CORPDATA not valid for operation.
    ORA-02063:  preceding 2 lines from AS400
    ```

7.4.1 Viewing Jobs

Use the WRKACTJOB command to view all jobs. Use the WRKACTJOB SBS(*myinst*) command to see only the jobs supporting your gateway instance; substitute your gateway instance name for *myinst*. If the gateway subsystem is active, then you will see the LISTENER job in SELW (select wait) status on the WRKACTJOB panel under the gateway subsystem. When the listener attempts to "hand-off" a new connection to a pre-started job, its status is CPCW (wait for completion of CPI communications call).

From the WRKACTJOB panel, press PF14 to see all of the RUNORAGT* jobs. Two RUNORAGTTP jobs are usually present, both in PSRW (pre-start, or ready to run) status.

If an active gateway connection exists, then the `RUNORAGT*` jobs can be in either `RUN` or `TIMW` (time wait) status.

7.5 Accessing OS/400 File Members

An OS/400 physical file can have multiple members. However, OS/400 does not allow you to directly access a physical file member through SQL. For example, if you have an OS/400 file named `EMP` in the library `CORPDATA` with members `MBR1`, `MBR2`, and `MBR3`, then issuing the following SQL statement results in an error:

```
SELECT * FROM CORPDATA.EMP.MBR1
```

If your OS/400 file has multiple members, then you can use OS/400 logical files to point to some or all of the members. To access a file member, create a logical file over the physical file members. In the `physical file data members` parameter (`DTAMBRS`) of the `CRTLF` command, specify which members you want to access. Using the previous example, if you wanted to access members `MBR1` and `MBR2` of `EMP`, then you would enter the following command:

```
CRTLF FILE((CORPDATA/EMPLF) DTAMBRS((CORPDATA/EMP (MBR1 MBR2))) +
SRCFILE(myinst/QDDSSRC) SRCMBR(EMPLF)
```

where member `EMPLF` in file `myinst`/QDDSSRC is as follows

```
A*  ACCESS TWO DIFFERENT MBRS IN AN EMP *FILE USING A LOGICAL
A*  FILE (THE TWO MEMBERS ARE NAMED EXTERNALLY ON CRTLF COMMAND)
A          R EMPTABLE                    PFILE(CORPDATA/EMP)
A          K EMPNO
```

This creates a logical file, `EMPLF`, that points to `MBR1` and `MBR2` of the `EMP` file. The SQL statement:

```
SELECT * FROM CORPDATA.EMPLF
```

will return all of the rows that are contained in `MBR1` and `MBR2`.

You can also specify `DTAMBRS(*ALL)` to include all members of a physical file in a logical file.

Note: To insert into a specific member, you must create a logical file for that member.

7.5.1 Accessing Flat Files

Flat files are OS/400 files that are created by a facility other than DB2/400, for example the `CRTPF` and `CRTSRCPF` commands can create such files. All flat files have an `externally described` attribute. The `externally described` attribute refers to whether or not a file is defined by an external template. All DB2/400 tables are externally described. You can access externally described flat files through the gateway in the same way that you access DB2/400 tables. Files in OS/400 that are created with the `CRTPF` command (which specifies DDS source) are externally described. Files in OS/400 that are created with the `CRTPF` command and without specifying DDS source are not externally described. The latter type of file can be accessed by the gateway. However, only one column will be available, the column name will be the same as the member name, and the data will be described as `RAW(n)`, where `n` is the record length that was used in the `CRTPF` command. Files in the integrated file system (IFS) are not accessible to the gateway because they are not accessible to DB2/400.

The syntax for accessing flat files is:

```
library.file
```

where:

library is the library name.

file is the file member (flat file) name.

If flat files have multiple members, then you need to create a view for each member or combination of members.

7.6 Performing Distributed Queries

The gateway technology can execute distributed queries that join with data from the Oracle Database 10*g* server and DB2/400, and any other data store for which Oracle provides a gateway. These complex operations can be completely invisible to the users requesting the data.

7.6.1 Example of a Distributed Query

7.6.1.1 SQL Command, Example 1

Assume that dblink DB2 points to a database gateway for DB2 on z/OS, and assume that dblink ORACLESERVR points to a remote Oracle Database instance, and assume that dblink AS400 points to an instance of the Oracle Database Gateway for DB2/400. Then the following example joins data between those three database servers.

```
SELECT O.CUSTNAME, P.PROJNO, E.ENAME, SUM(E.RATE*P.HOURS)
FROM ORDERS@DB2 O, EMP@ORACLESRVR E, PROJECTS@AS400 P
WHERE O.PROJNO = P.PROJNO
AND P.EMPNO = E.EMPNO
GROUP BY O.CUSTNAME, P.PROJNO, E.ENAME
```

Through a combination of views and synonyms, the process of distributed queries may become invisible to the user. For example:

```
CREATE SYNONYM ORDERS FOR ORDERS@DB2
CREATE SYNONYM PROJECTS FOR PROJECTS@AS400
CREATE SYNONYM EMP FOR EMP@ORACLESRVR
CREATE VIEW DETAILS (CUSTNAME,PROJNO,ENAME,SPEND) AS
  SELECT O.CUSTNAME, P.PROJNO, E.ENAME, SUM(E.RATE*P.HOURS)
  FROM ORDERS O, EMP E, PROJECTS P
  WHERE O.PROJNO = P.PROJNO AND P.EMPNO = E.EMPNO
  GROUP BY O.CUSTNAME, P.PROJNO, E.ENAME
```

7.6.1.2 SQL Command, Example 2

Using the following SQL statement, a user can retrieve information from the three data stores in Example 1 using a single command, as follows:

```
SELECT * FROM DETAILS;
```

The results from the SQL command in example 2 are the same as those from the SQL command in Example 1, and might look like the following:

Table 7–1   Example Results from SQL Distributed Queries

CUSTNAME	PROJNO	ENAME	SPEND
ABC Co.	1	Jones	400
ABC Co.	1	Smith	180
XYZ Inc.	2	Jones	400
XYZ Inc.	2	Smith	180

7.6.2 Two-Phase Commit Processing

For a database to fully participate in a two-phase commit transaction, it must support both a prepare phase and a commit phase for committing transactions. The prepare phase ensures that all participating nodes that are referenced in a distributed transaction are prepared to commit or to abort the transaction, regardless of intervening failures.

The Oracle database supports two-phase commit transactions. So, any number of Oracle Database servers can participate in a distributed two-phase commit transaction. The prepare phase is performed when a COMMIT is issued at the end of a distributed transaction.

> **Note:**   The prepare phase occurs automatically when an application COMMIT is executed. No other action is necessary.

DB2/400 does not support a prepare phase for committing a transaction. Consequently, it does not support two-phase commit transactions. Therefore, the two-phase commit protocol is limited when the gateway participates in a distributed transaction. In this case, the gateway becomes the commit point site of the distributed transaction. Because the gateway is configured as commit and confirm, it is always the commit point site, regardless of the COMMIT_POINT_STRENGTH setting of any of the participating Oracle databases. The gateway commits the local AS/400 unit of work after verifying that all Oracle databases in the transaction have successfully committed their work.

Because the gateway must drive the distributed transaction, only one gateway can participate in an Oracle two-phase commit transaction.

Two-phase commit transactions are recorded in the ORACLE2PC physical file. You must journal this file before issuing a distributed transaction to the AS/400. Refer to "Step 3: Journal the ORACLE2PC File" on page 4-8 for more information.

For additional information about the two-phase commit process, refer to the *Oracle Database Administrator's Guide*.

7.6.2.1 Recovering Failed Transactions

If a two-phase commit transaction fails because the database connection is lost, then the pending transaction is stored on the Oracle Database. Every time a user attempts to login to the gateway via s specific database link, the Oracle Server checks to see if there are any pending (probably failed) transactions that had previously used that same database link. If so, Oracle suspends the current gateway login and instead tells the Gateway to login in using the Recovery User Profile and Password and to perform some recovery operations. After these recovery operations are complete, Oracle will then continue with the normal gateway login (which had been suspended).

During the recovery process, the server uses the recovery user ID and password to login to the system and resolve pending transactions. After recovery is complete, the gateway executes the user's login request. This recovery process is transparent to the user.

The recovery user ID and password are initially set in the panel shown in Example 4–2, "Install Oracle Database Gateway Panel, With Name Choices". If you need to change these values, then you must run the CHGRECOPRF command.

7.7 Replicating in a Heterogeneous Environment

Oracle Database Gateway for DB2/400 provides a number of options for replicating Oracle data and non-Oracle data throughout the enterprise.

7.7.1 Oracle Database Server Triggers

When updates are made to the Oracle database, synchronous copies of Oracle data and non-Oracle data can be maintained automatically by using Oracle database triggers.

7.7.2 Oracle Materialized View

Oracle Database Gateway for DB2/400 can use the Oracle Materialized View (snapshot) feature to automatically replicate non-Oracle data into the Oracle database. This complete refresh capability of Oracle Materialized View can be used to propagate a complete copy or a subset of the non-Oracle data into the Oracle database at user-defined intervals.

7.8 Copying Data from Oracle Server to DB2/400 Server

Data can be copied from the Oracle Database 10*g* server to the DB2/400 server by two methods:

- Triggers
- SQL*Plus COPY command

7.8.1 Triggers

When updates are made to the Oracle Database, synchronous copies of Oracle data and non-Oracle data can be maintained automatically by using Oracle Database triggers.

For example, suppose that you have an Oracle table ORA_EMP that contains fields ENAME and EMPNO. Suppose that you also have a table called DB2_EMP, which is a copy of ORA_EMP and which resides on DB2/400. You want all changes made to the Oracle ENAME field to be reflected immediately in your DB2_EMP table on DB2/400. In the following scenario, an Oracle database trigger can be developed to run every time an update is made to the ENAME field in your Oracle ORA_EMP table:

```
CREATE OR REPLACE trigger EMP_TRIGGER
 after update of ename on SCOTT.ORA_EMP
 for each row
BEGIN
     UPDATE SCOTT.DB2_EMP@tg4db2
     SET ENAME = :NEW.ENAME
     WHERE EMPNO = :NEW.EMPNO;
END;
```

where *tg4db2* is the name of the database link that is used to access the gateway.

7.8.2 SQL*Plus COPY Command

The SQL*Plus COPY command copies data from the Oracle Database to the DB2/400 server. The SQL command INSERT is not supported as a way to copy tables from the Oracle server to the gateway. The command:

```
INSERT INTO gateway_table@gateway SELECT * FROM oracle_table;
```

displays the following message:

```
ORA-2025: All tables in the SQL statement must be at the remote database.
```

Use the following SQL*Plus syntax to copy data from your local Oracle Database to the DB2/400 server:

```
COPY FROM username/password@ORACLESRVR -
INSERT destination_table@gateway -
USING query;
```

The next example selects all rows from the local Oracle EMP table and inserts them into the EMP table on the DB2/400 server:

```
COPY FROM SCOTT/TIGER@ORACLESRVR -
INSERT SCOTT.EMP@gateway -
USING SELECT * FROM EMP;
```

> **Note:**   Although the SQL*Plus COPY command supports the APPEND, CREATE, INSERT, and REPLACE options, INSERT is the only option that is supported when copying to the DB2/400 server. For more information about the COPY command, refer to the *SQL*Plus User's Guide and Reference*.

7.9 Copying Data to Oracle Server from DB2/400 Server

Use one of the following options to copy data from the DB2/400 server to the Oracle Database:

- Use the CREATE TABLE command to copy data from the DB2/400 server to the Oracle database. To create a table on your local database and to insert rows from a DB2/400 table, use:

  ```
  CREATE TABLE table_name AS query;
  ```

  The next example creates the table EMP in the local Oracle database and inserts the rows from the EMP table on the DB2/400 server:

  ```
  CREATE TABLE EMP AS SELECT * FROM SCOTT.EMP@gateway;
  ```

- Use the INSERT command to copy data from the DB2/400 server to the Oracle Database:

  ```
  INSERT INTO oracle_table SELECT * FROM db2table@gateway;
  ```

  The following example selects all rows from the EMP table on the DB2/400 server and inserts them into the local Oracle EMP table:

```
INSERT INTO EMP SELECT * FROM SCOTT.EMP@gateway;
```

- Use the CREATE MATERIALIZED VIEW command to automatically and asynchronously copy DB2/400 server data into the Oracle database. The complete refresh capability can be used to propagate a complete copy or a subset. For more information about creating materialized views, refer to the *Oracle Database SQL Reference*. To create a copy:

```
CREATE MATERIALIZED VIEW empdb2
  PCTFREE 5 PCTUSED 60
  TABLESPACE users
  STORAGE (INITIAL 50K NEXT 50K)
  REFRESH COMPLETE NEXT SYSDATE + 1
  WITH ROWID
  AS
    SELECT * FROM SCOTT.EMP@gateway;
```

The following example creates a materialized view of data that is refreshed every day after the first refresh. If you require only a subset of the AS/400 data, then a WHERE clause is added, as in the following example:

```
CREATE MATERIALIZED VIEW empdb2
  PCTFREE 5 PCTUSED 60
  TABLESPACE users
  STORAGE (INITIAL 50K NEXT 50K)
  REFRESH COMPLETE NEXT SYSDATE + 1
  WITH ROWID
  AS
    SELECT * FROM SCOTT.EMP@gateway
  WHERE deptno=20;
```

- Use the SQL*Plus COPY command to copy data from the DB2/400 server to the Oracle Database:

```
COPY FROM username/password@tns_alias -
INSERT destination_table -
USING query;
```

where *username* or *password,* or both, are valid at the Oracle Database that is identified by *tns_alias,* and where *query* refers to a remote table that is accessed through a gateway database link.

The following example selects all rows from the EMP table in DB2/400 and inserts them into the local Oracle EMP table:

```
COPY FROM SCOTT/TIGER@ORACLE_SERVER -
INSERT EMP -
USING SELECT * FROM SCOTT.EMP@gateway;
```

8

Administering the Gateway

Read this chapter to understand how to administer your gateway either by using the Oracle Gateway Monitor or by entering commands at the command line prompt. The following sections are included:

- Overview of the Oracle Gateway Monitor on page 8-1
- Starting the Oracle Gateway Monitor on page 8-1
- Starting and Stopping the Gateway on page 8-3
- Starting the TCP/IP Listener on page 8-4
- Displaying the Gateway Call Stack on page 8-4
- Working with Gateway Job Locks on page 8-4
- Displaying Open Files on page 8-4
- Gateway Security on page 8-4
- Streams Replication on page 8-5

8.1 Overview of the Oracle Gateway Monitor

The Oracle Gateway Monitor is a facility that is provided with the Oracle Database Gateway for DB2/400. This facility permits a single interface to perform the following functions:

- start and stop an Oracle gateway instance
- work with Oracle gateway jobs
- start the TCP/IP listener job
- display the gateway call stack
- work with the gateway job locks
- display the open files of the gateway

8.2 Starting the Oracle Gateway Monitor

To execute the Oracle gateway monitor, you must log on to the gateway instance user ID and have the gateway instance library included in your library list. To start the Oracle gateway monitor, enter the following command:

```
ORAMON
```

The main menu for the Oracle Gateway Monitor is displayed as shown in
Example 8–1, "Monitor Oracle Subsystems".

Example 8–1 Monitor Oracle Subsystems

```
                        Monitor Oracle Subsystems          System:  AS400A


Position to _____     Update frequency  *OFF


  Type Options, press Enter.
     2=Start   4=End   5=Work with   7=Listen   10=Display call stack
     11=Work with locks   14=Display open files

                                            Job    Aux    Aux    CPU
  Opt      Name               Type    Status  Number  I/O    KB    Secs
  -        ORACLE             SBS     ACTV             0      0     .0
  -          LISTENER         LIS     SELW    015826   564   3386   2.8
  -          RUNORAGTTP       PJ      PSRW    015823   124   3725   1.6
  -          RUNORAGTTP       PJ      PSRW    015822   138   3737   1.5


F1=Help   F2=Change list   F3=Exit   F5=Refresh   F8=Start auto update
```

The columns in the panel in Example 8–1, "Monitor Oracle Subsystems" represent the
following:

Opt
is used to perform various functions on subsystems and jobs. For example, option 2
starts a gateway subsystem. Enter an option number next to the subsystem or job
name, and then press Enter. You can select options for more than one subsystem or job
at a time. The option numbers are listed in the panel of Example 8–1.

Name
displays the names of Oracle subsystems and jobs.

Type
is the type of entry for this line:

- **ASJ –** automatically started job

- **LIS –** listener

- **PJ –** pre started job waiting for start request

- **SBS –** subsystem

- **SRV –** server job

- **UNK –** unknown

Status
is the status of the subsystem or job that is associated with the line.

For subsystems (type **SBS**):

- **ACTV –** active subsystem

- **ERROR –** error while determining subsystem status

- **INACTV –** inactive subsystem

- **NOAUTH –** user has no authorization to start or stop subsystem

For jobs, refer to the Help information for the Status column of the WRKACTJOB panel.

Job Number
is the OS/400 job number for job-related lines. It is blank for subsystem lines.

Aux I/O
is the total number of auxiliary storage I/O operations that were performed.

Aux KB
is the current amount of auxiliary storage occupied by the job, in kilobytes (units of 1024 bytes).

CPU Secs
is the total CPU time used, in seconds.

To display specific gateway subsystems and jobs, you must enter the subsystem into the Oracle gateway monitor subsystem by pressing PF2. This displays a panel where you can enter the name of the Oracle gateway subsystems. The panel in Example 8–1, "Monitor Oracle Subsystems" shows a single gateway subsystem with a single listener and two prestarted gateway jobs.

To work with a specific gateway subsystem or job, you must enter an option number in the *Opt* field next to the subsystem or job that is displayed on the main panel.

In Example 8–1, "Monitor Oracle Subsystems", the *Update frequency* field is set to *OFF. This means that no updates of the information are displayed until you press PF5. If you want the screen to be updated periodically, then you need to press PF8. The default update refresh period is 30 seconds, but you can set it to a value between 30 seconds and 600 seconds.

If help is needed at any time, then pressing PF1 on any of the panels will display more information about a specific panel.

8.3 Starting and Stopping the Gateway

Entering a 2 in the Option field on the main panel allows you to start a gateway subsystem. ACTV is displayed in the STATUS field of a subsystem (SBS) when the gateway subsystem is started. To start the gateway subsystem, you must be authorized to use the STRSBS command.

To stop the gateway subsystem or a job in a gateway subsystem, move the cursor to the Opt field that is next to the subsystem or job that you want to shut down, and enter a 4. This ends the specified gateway subsystem or job. When the gateway subsystem has shut down, INACTV is displayed in the field next to the subsystem or job that ended. To shut down the gateway, you must be authorized to use the ENDSBS command.

8.3.1 Working with Oracle Gateway Jobs

To work with gateway jobs, specify option 5 next to the job. This displays the OS/400 Work with Job menu to work with a specific gateway job.

8.3.2 Starting and Stopping the Gateway Using the Command Line

An instance of the Oracle Database Gateway for DB2/400 runs within an OS/400 subsystem. All jobs relating to that instance run within the subsystem. You use the STRSBS command to start a subsystem. For example, to start an instance of a gateway called ORACLE, you enter the following command:

```
STRSBS ORACLE/ORACLE
```

where the first *ORACLE* is a library name, and the second *ORACLE* is the name of a subsystem.

To stop a gateway, you must end the subsystem by using the ENDSBS command. For example, to stop a gateway called ORACLE, to immediately end all jobs within the ORACLE subsystem, and to stop the subsystem itself, you enter the following command:

```
ENDSBS ORACLE OPTION(*IMMED)
```

8.4 Starting the TCP/IP Listener

Specify option 7 in the Opt field that is next to the gateway subsystem name if you want to start the gateway listener. The gateway listener is identified by LIS in the TYPE field that is displayed on the main menu.

8.5 Displaying the Gateway Call Stack

Displaying the gateway call stack is not needed for most installations but might be requested by Oracle Support Services during problem isolation and determination. Entering option 10 in the Opt field displays a call stack for a specific job.

8.6 Working with Gateway Job Locks

Specify option 11 in the Opt field next to a specific job in order to display and work with object locks that are held by the gateway. This option displays the OS/400 Work with Job Locks panel.

8.7 Displaying Open Files

Specify option 14 in the Opt field next to a specific job to show the files that are open for that specific job. This option displays the OS/400 Display Open Files panel.

8.8 Gateway Security

The Oracle user ID and password are passed over the database link to the gateway to authorize gateway users to DB2/400 objects. If the CONNECT TO clause is specified when creating the database link, then the user ID and password that are sent to the gateway are those that are specified in this clause. If the CONNECT TO clause is omitted from the database link specification, then the Oracle user ID and password of the task that is using the database link are passed to the gateway for authorization.

> **Caution:** You must keep the gateway recovery user ID and password synchronized with the OS/400 User Profile and password. The gateway recovery user ID and password must be valid to OS/400 at all times. If, when transaction recovery is being attempted by Oracle, the password for the recovery user ID is not valid or if the recovery user ID itself is invalid (it may have expired, for example), an ORA-1017 error (invalid user ID or password) will be inserted into the Oracle Database alert log. The CHGRECOPRF command is used to change the recovery profile parameters (including recovery user ID and password).
>
> For more information about using CHGRECOPRF, refer to

8.9 Streams Replication

The Oracle Database Gateway for DB2/400 and Heterogeneous Services now support replication to DB2/400 by using Oracle Streams. Oracle Streams is a rule-based process that allows changes to an Oracle Database table to be captured and applied to an equivalent DB2/400 table, based on user-written rules.

> **Note:** Replication from an Oracle Database to DB2/400 is supported. Replication from DB2/400 to an Oracle Database is *not* supported at this time.

Before setting up your Streams replication environment, ensure that archivelog is enabled.

> **Note:** Streams will *not* work if archivelog is not enabled.

An example of a simple table replication follows:

First you should grant the necessary authorizations to your Streams admin user ID.

```
CONNECT SYS/SYS_PASSWORD AS SYSDBA

GRANT CONNECT, RESOURCE, SELECT_CATALOG_ROLE
  TO strmadmin IDENTIFIED BY strmadminpw;

GRANT EXECUTE ON DBMS_APPLY_ADM TO strmadmin;
GRANT EXECUTE ON DBMS_AQADM TO strmadmin;
GRANT EXECUTE ON DBMS_CAPTURE_ADM TO strmadmin;
GRANT EXECUTE ON DBMS_FLASHBACK TO strmadmin;
GRANT EXECUTE ON DBMS_PROPAGATION_ADM TO strmadmin;
GRANT EXECUTE ON DBMS_STREAMS_ADM TO strmadmin;

BEGIN
  DBMS_RULE_ADM.GRANT_SYSTEM_PRIVILEGE(
  privilege => DBMS_RULE_ADM.CREATE_RULE_SET_OBJ,
  grantee => 'strmadmin',
  grant_option => FALSE);
END;
/

BEGIN
```

```
      DBMS_RULE_ADM.GRANT_SYSTEM_PRIVILEGE(
      privilege => DBMS_RULE_ADM.CREATE_RULE_OBJ,
      grantee => 'strmadmin',
      grant_option => FALSE);
END;
/
```

Then, set up the Streams queue and the database link that the apply process will use.

```
CONNECT strmadmin/strmadminpw

EXEC DBMS_STREAMS_ADM.SET_UP_QUEUE();

DROP DATABASE LINK strmdblink.your.domain.com;

CREATE DATABASE LINK strmdblink.your.domain.com
  CONNECT TO userid IDENTIFIED BY password
     USING 'tnsnames_entry';
```

Next, create the capture and apply processes and define the replication rules.

```
CONNECT SYS/SYS_PASSWORD AS SYSDBA

ALTER SYSTEM ARCHIVE LOG CURRENT;

CONNECT strmadmin/strmadminpw


--- ----------------------------------------------------------------
--- Stop the capture process if it's already active.
--- ----------------------------------------------------------------
BEGIN
  DBMS_CAPTURE_ADM.STOP_CAPTURE(
     capture_name => 'db2400_capture');
END;
/

--- ----------------------------------------------------------------
--- Stop the apply process if it's already active.
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.STOP_APPLY(
     apply_name => 'apply_2_db2400');
END;
/

--- ----------------------------------------------------------------
--- Define the capture rule, this one captures changes to scott.emp
--- ----------------------------------------------------------------
BEGIN
  DBMS_STREAMS_ADM.ADD_SCHEMA_RULES(
     schema_name => 'scott',
     streams_type => 'capture',
     streams_name => 'db2400_capture',
     queue_name => 'strmadmin.streams_queue',
     include_dml => true,
     include_ddl => true);
END;
/

--- ----------------------------------------------------------------
--- Set the capture instantiation level
```

```
--- ----------------------------------------------------------------
DECLARE
  iscn NUMBER; -- Variable to hold instantiation SCN value
BEGIN
  iscn := DBMS_FLASHBACK.GET_SYSTEM_CHANGE_NUMBER();

  DBMS_APPLY_ADM.SET_TABLE_INSTANTIATION_SCN(
    source_object_name => 'scott.emp',
    source_database_name => 'ORAv92',
    instantiation_scn => iscn,
    apply_database_link => 'strmdblink.your.domain.com');
END;
/


--- ----------------------------------------------------------------
--- Drop the apply process if it already exists.
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.DROP_APPLY(
    apply_name => 'apply_2_db2400');
END;
/


--- ----------------------------------------------------------------
--- Create the apply process
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.CREATE_APPLY(
    queue_name => 'strmadmin.streams_queue',
    apply_name => 'apply_2_db2400',
    apply_database_link => 'strmdblink.your.domain.com',
    apply_captured => true);
END;
/


--- ----------------------------------------------------------------
--- Create the apply rule
--- ----------------------------------------------------------------
BEGIN
  DBMS_STREAMS_ADM.ADD_TABLE_RULES(
    table_name => 'scott.emp',
    streams_type => 'apply',
    streams_name => 'apply_2_db2400',
    queue_name => 'strmadmin.streams_queue',
    include_dml => true,
    include_ddl => false,
    source_database => 'ORAv92');
END;
/


--- ----------------------------------------------------------------
--- Turn on tracing for the apply process (be careful, this
--- generates alot of output).
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.SET_PARAMETER(
    apply_name => 'apply_2_db2400',
    parameter => 'trace_level',
    value => 127 );
END;
```

```
/

--- ----------------------------------------------------------------
--- Turn off disable_on_error for the apply process
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.SET_PARAMETER(
    apply_name => 'apply_2_db2400',
    parameter => 'disable_on_error',
    value => 'n');
END;
/


--- ----------------------------------------------------------------
--- Start the apply process.
--- ----------------------------------------------------------------
BEGIN
  DBMS_APPLY_ADM.START_APPLY(
    apply_name => 'apply_2_db2400');
END;
/


--- ----------------------------------------------------------------
--- Start the capture process.
--- ----------------------------------------------------------------
BEGIN
  DBMS_CAPTURE_ADM.START_CAPTURE(
    capture_name => 'db2_capture');
END;
/
```

For detailed information about Oracle streams replication, refer to Oracle Streams
Concepts and Administration.

9

Developing Applications

An application that is written to access information in an Oracle DB can be made to access similar information in DB2/400 by simply qualifying the table, view, stored procedure, or function name with a database link. Using a database link, the access can be made transparent by using synonyms or views of the DB2/400 tables. However, fundamental SQL, data type, and semantic differences exist between the Oracle server and DB2/400. Read this chapter to learn these differences and to find information about developing applications.

To develop applications that access OS/400 data through the gateway, you need to understand the topics in this chapter. The following sections are included:

- Gateway Appearance to Application Programs on page 9-1

- Array Processing on page 9-2

- Using Oracle Stored Procedures with the Gateway on page 9-4

- Using DB2/400 Stored Procedures with the Gateway on page 9-5

- Passing DB2/400 SQL Statements Through the Gateway on page 9-12

- Converting DB2/400 Data Types to Oracle Data Types on page 9-14

- SQL Functions on page 9-20

- Oracle Server SQL Construct Processing on page 9-21

- Oracle Server and DB2/400 Differences on page 9-22

- Oracle Data Dictionary Emulation in a DB2/400 Server on page 9-23

- Oracle Developer Forms Compatibility on page 9-24

9.1 Gateway Appearance to Application Programs

An application that is written to access information in a DB2/400 database interfaces with an Oracle database. When developing applications, remember the following:

- You must define the object in the DB2/400 database to the Oracle application by use of a database link that is defined at the Oracle DB. Your application specifies DB2/400 objects (for example, tables) by qualifying those object names with a database link. For example, suppose you have defined a database link and named it DB2400. And suppose you have a table named EMP in the local Oracle database and a similarly defined EMPS table in DB2/400. And finally, suppose you want to join the two tables to extract some information. The following SQL statement retrieves data from both the Oracle database and DB2/400:

```
SELECT EMP.EMPNO, EMPS.SALARY FROM EMP, EMPS@DB2400
```

```
WHERE EMP.EMPNO = EMPS.EMPNO;
```

Alternatively, you can define a synonym or a view referencing the DB2/400 server table and access the information without the database link qualification.

For example:

```
 CREATE SYNONYM EMPS FOR EMPS@DB2400;
SELECT EMP.EMPNO, EMPS.SALARY FROM EMP, EMPS
      WHERE EMP.EMPNO = EMPS.EMPNO;
```

- You can perform reads and writes of data to a defined DB2/400 database. SELECT, INSERT, UPDATE, and DELETE are all valid operations.

- A single transaction *cannot* write to more than one DB2/400 database. A stored procedure call through DB2/400 counts as a write. A call to a user-defined function within a SELECT, UPDATE, DELETE, or INSERT does *not* count as a write.

- Single SQL statements, using a JOIN, can refer to tables in multiple Oracle databases, multiple DB2/400 databases, or both.

9.1.1 Connecting to DB2/400 Databases with Database Links

A connection to the gateway is established through a database link when that link is first used in an Oracle database session. The connection remains active until the session ends. Another session or user can access the same database link but will get a separate connection to DB2/400.

Connections to DB2/400 might be limited by factors such as memory, gateway parameters, or OS/400 server resources.

9.1.2 Read and Write Access to DB2/400 Databases

You can perform read and write functions to a DB2/400 database. SELECT, INSERT, UPDATE, and DELETE are all valid operations.

A single transaction can use an UPDATE statement to change multiple Oracle databases. Only one DB2/400 database can be updated within a transaction. If a stored procedure call is directed to DB2/400, then no updates or stored procedure calls can be directed to any other DB2/400 databases during the transaction. Note that a user-defined function is treated differently as Oracle assumes a user-defined function does *not* write to DB2/400.

A single transaction can use a SELECT statement to retrieve data from multiple Oracle databases and multiple DB2/400 databases.

The gateway is set up with commitment control set to *CHG. Refer to the IBM reference for IBM DB2 for OS/400 DB2/400 SQL for more information regarding OS/400 commitment control.

9.1.3 Accessing Tables in Multiple Databases

Single SQL statements, using JOINs, can refer to tables in multiple Oracle Databases or multiple DB2/400 databases.

9.2 Array Processing

When evaluating and tuning your gateway configuration, you may achieve performance gains by using the Oracle Array Processing interface. An array is a

collection of data items, called elements, that are associated with a single variable. With arrays, you can use a single SQL statement to manipulate an entire collection of data items. For example, suppose you want to insert information regarding 100 employees into the EMP table on DB2/400. Without arrays, your program must do 100 individual INSERTs, one for each employee. With arrays, only one INSERT is necessary.

The use of array processing reduces network calls, which may save elapsed time and CPU cycles. In addition, when using INSERT for multiple rows, DB2/400 processing is optimized by retaining the original SQL statement for repeated execution.

The array size is a factor in transferring data between the client application (for example, SQL*Plus) and the Oracle integrating server when using UPDATE, DELETE, and INSERT statements, in that it allows one UPDATE, DELETE, or INSERT statement to be used with a series of values. For more information about array processing usage and implementation in the Oracle application, refer to *Oracle Database SQL Reference*.

Figure 9–1   Use of Array Size Definition in the DB2/400 Architecture for UPDATE, DELETE, and INSERT
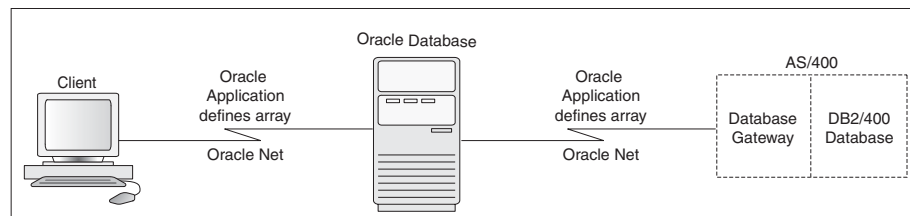
> **Note:**   For performance reasons, Oracle recommends setting the initial Oracle application array size between 10 and 100.

9.2.1 Fetch Reblocking

The array size of the application for SELECT is effective between the application and the Oracle Database 10*g* server. However, the block size of the buffer between the Oracle integrating server and the gateway is governed by two initialization parameters, RPC FETCH SIZE and RPC FETCH REBLOCKING. These gateway initialization parameters correspond to the Heterogeneous Services initialization parameters HS_RPC_FETCH_SIZE and HS_RPC_FETCH_REBLOCKING, respectively. Figure 9–2 illustrates this. Refer to *Oracle Database Administrator's Guide* for more information.

With release 10 of the gateway, the Oracle Database 10*g* server supports fetch reblocking with the RPC FETCH REBLOCKING parameter. When the value of this parameter is set to *YES (the default), the block size of the buffer for SELECT statements is determined by the RPC FETCH SIZE value. The RPC FETCH SIZE parameter defines the number of bytes sent with each buffer from the gateway to the Oracle Database 10*g* server. The buffer might contain one or more qualifying rows from DB2/400. This feature can provide significant performance enhancements, depending on your application design, installation type, and workload. The RPC FETCH REBLOCKING and RPC FETCH SIZE parameters may be changed by using the CHGORATUN command. Refer to "CHGORATUN, Change Initialization Parameters" on page 6-6 for more information.

The array size between the client and the Oracle Database 10*g* server is still determined by the Oracle application. To ensure optimal performance, Oracle

Corporation recommends that you set your Oracle application array size equal to the setting of the gateway ARRAY BLOCK SIZE parameter. For more information about the gateway ARRAY BLOCK SIZE parameter, refer to "Setting Optional Parameters" on page 6-10. For more information about array processing usage and implementation in your Oracle application, refer to *Oracle Database SQL Reference*.

Figure 9–2   Array Size Definition in the DB2/400 Architecture for SELECT
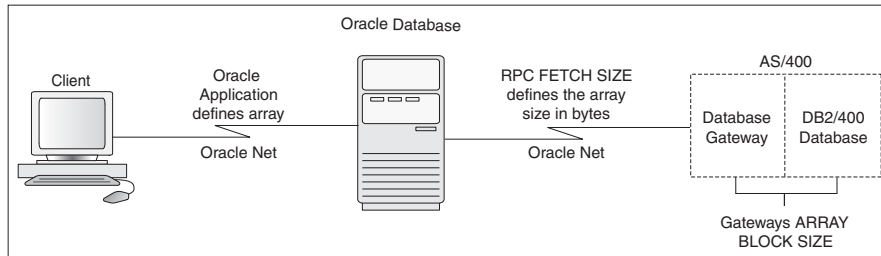
9.3 Using Oracle Stored Procedures with the Gateway

The gateway stored procedure support is an extension of Oracle stored procedures. An Oracle stored procedure is a schema object that logically groups a set of SQL and other PL/SQL programming language statements together to perform a specific task. Oracle stored procedures are stored in the database for continual use. Applications use standard Oracle PL/SQL to invoke stored procedures. There are no other ways to invoke a stored procedure.

Oracle stored procedures can be located in a local instance of Oracle and a remote instance. Figure 9–3 shows two stored procedures. oraproc1 is a procedure stored in the ORA1 Oracle instance, while oraproc2 is a procedure stored in the ORA2 Oracle instance.

Figure 9–3   Calling Oracle Stored Procedures in a Distributed Oracle Environment
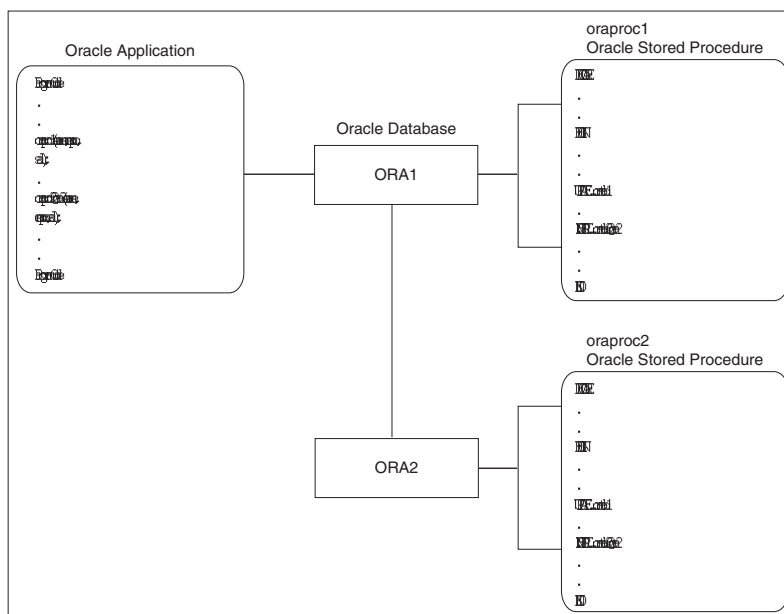
If the application needs to maintain location transparency, then a synonym can be created:

```
CREATE SYNONYM ORAPROC2 FOR oraproc2@ora2;
```

where `oraproc2` is the procedure that is stored in `ORA2`.

After this synonym is created, the application no longer needs to use the database link specification for invoking the stored procedure at the remote Oracle instance.

In Figure 9–3, the `INSERT` statement in `oraproc1` is used to access a table in the `ORA2` instance. In the same way, Oracle stored procedures can be used to access DB2/400 tables through the gateway.

In Figure 9–4, `empproc` is an Oracle stored procedure that subsequently accesses data in DB2 by using the Oracle Database Gateway for DB2/400.

Figure 9–4   Using Oracle Stored Procedures with DB2/400
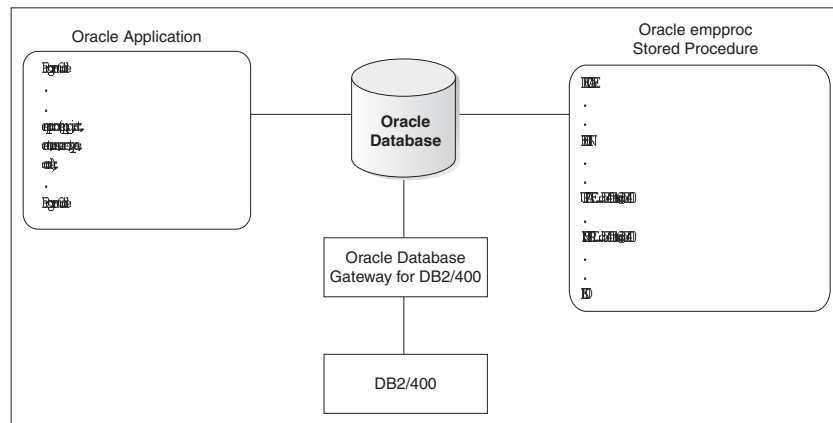
As with the Oracle gateway, standard PL/SQL is used to create and execute the procedure. There is no difference in the gateways except that the stored procedure is accessing DB2/400 instead of an Oracle Database.

Gateway two-phase commit processing also applies to updates to DB2/400 that are being made within an Oracle stored procedure. This means that the stored procedure can update a single instance of DB2/400 while also updating any number of Oracle Databases within a single transaction.

A call to a DB2/400 stored procedure counts as a write because, as far as the Oracle Database Gateway for DB2/400 is concerned, the gateway cannot tell if the stored procedure is actually changing any object in OS/400. So, you cannot have a transaction that calls two DB2/400 stored procedures via two separate database links. Correspondingly, during a transaction, you cannot call a DB2/400 stored procedure through one database link and also do an `INSERT`, `UPDATE`, or `DELETE` relative to DB2/400 using another database link (even if the two database links resolve to the same OS/400 system).

9.4 Using DB2/400 Stored Procedures with the Gateway

The procedural feature of the gateway enables execution of native DB2/400 stored procedures. In other words, the stored procedure is no longer defined in the Oracle Database but instead is defined to DB2/400. Again, standard Oracle PL/SQL is used by the Oracle application to execute the DB2/400 stored procedure.

The gateway does not require special definitions in order to invoke the DB2/400 stored procedure. After the stored procedure is defined to DB2/400, the gateway will be able to use the existing DB2/400 definition to execute the procedure.

In Figure 9–5, an Oracle application calls the `empproc` stored procedure that is defined to DB2/400.

Figure 9–5   Executing DB2/400 Stored Procedures
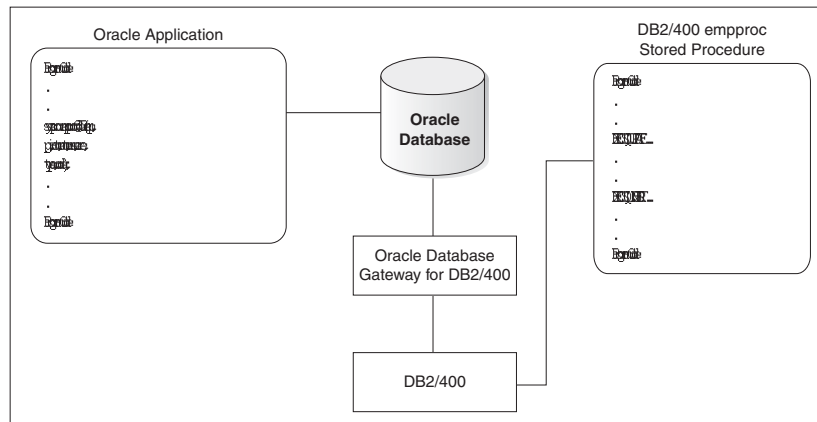
From the perspective of the application, executing the DB2/400 stored procedure is no different than invoking a stored procedure at a remote Oracle Database instance.

9.4.1 Executing DB2/400 Stored Procedures from Applications

In order for an application to invoke a DB2/400 stored procedure, it is first necessary to create the DB2/400 stored procedure on the DB2/400 system by using the procedures that are documented in the IBM reference for DB2/400.

After the stored procedure is defined to DB2/400, the gateway is able to access that stored procedure by using a standard PL/SQL call. For example, an employee name, `JOHN SMYTHE`, is passed to the DB2/400 stored procedure `REVISE_SALARY`. The DB2/400 stored procedure retrieves the salary value from the DB2/400 database to calculate a new yearly salary for `JOHN SMYTHE`.

The revised salary returned in `RESULT` is used to update `SAL` in the `EMP` table of an Oracle Database as follows:

```
DECLARE
  INPUT VARCHAR2(15);
  RESULT NUMBER(8,2);
BEGIN
  INPUT := 'JOHN SMYTHE';
  MYLIB.REVISE_SALARY@DB2400(INPUT, RESULT);
  UPDATE EMP SET SAL = RESULT WHERE ENAME := INPUT;
END;
```

When the gateway receives a call to execute a DB2/400 stored procedure, it first does a lookup of the procedure in the `QSYS2/SYSPROCS` and `QSYS2/SYSPARMS` DB2/400 system tables to determine the following:

1. The stored procedure to execute

The gateway retrieves information from the `ROUTINE_SCHEMA` and `ROUTINE_NAME` columns of the `QSYS2/SYSPROCS` table in order to locate the actual DB2/400 stored procedure code to be invoked by the gateway. Multiple DB2/400 stored procedures can have the same `ROUTINE_NAME`, but they must be uniquely identified by the combination of the `ROUTINE_SCHEMA` and `ROUTINE_NAME` columns of the `QSYS2/SYSPROCS` table.

The gateway always receives a qualified stored procedure name from the Oracle database and executes the DB2/400 stored procedure by using this qualified name. If a qualified name is not explicitly identified by the application, then the user ID that is passed over the database link is used to match the value in `ROUTINE_SCHEMA`. To ensure that you execute the correct DB2/400 stored procedure, Oracle recommends that you use fully qualified stored procedure names when invoking DB2/400 stored procedures from Oracle PL/SQL.

2. The parameter list of the stored procedure

   When a DB2/400 stored procedure is defined, the attributes of each entry in the argument list for the stored procedure are placed in the `QSYS2/SYSPARMS` table. The gateway uses this information to convert the data type of each argument to its corresponding PL/SQL data type. Refer to "Converting DB2/400 Data Types to Oracle Data Types" on page 9-14 for gateway data type conversion rules.

After the lookup of the description of the DB2/400 stored procedure is complete, the gateway uses this information to execute the DB2/400 stored procedure.

9.4.2 Considerations for Using Stored Procedures with DB2/400

Following is a list of special considerations for using the procedure feature and DB2/400 stored procedures with the Oracle Database Gateway for DB2/400:

- PL/SQL records cannot be passed as parameters when invoking a DB2/400 stored procedure.

- The gateway executes the DB2/400 `CALL` statement to execute DB2/400 stored procedures. Commitment control (of database changes that are made with the DB2/400 stored procedure) is managed by DB2/400. For more information regarding commitment control, refer to the IBM reference for DB2/400.

- The gateway supports both `GENERAL` and `GENERAL WITH NULLS` linkage conventions of DB2/400 stored procedures.

    - The `GENERAL` linkage convention means that the parameters that are passed to and from DB2/400 stored procedures cannot be null.

    - The `GENERAL WITH NULLS` linkage convention means that parameters passed to and from DB2/400 stored procedures can be null when they are passed using indicator variables.

  Oracle Call Interface (OCI) or embedded PL/SQL can be used in host programs to operate on indicator variables. For information about writing DB2/400 stored procedures to use indicator variables, refer to the IBM reference for DB2/400.

- To ensure commitment control integrity between the gateway and the stored procedure program, you must create the ILE `*PGM` object containing the DB2/400 stored procedure by using the activation group attribute `*CALLER`. If you do not use the `*CALLER` attribute, then stored procedures execute in a different activation group than the gateway. If the activation group attribute was given as `*NEW`, then a forced `COMMIT` may occur when the activation group ends (that is, when the program object that is defined by the stored procedure ends). For more information about activation groups, refer to the IBM document for ILE concepts.

- If the stored procedure does not change any file or any DB2/400 table on the AS/400, then the program object may be an OPM program object. Otherwise, the program object must be an ILE program object. Failure to adhere to this may cause data to become unsynchronized in the case of a `ROLLBACK` or `COMMIT` that either may be planned by you or may be forced because of some failure.

9.5 Using Oracle Stored Functions with the Gateway

The gateway stored function support is an extension of Oracle stored functions. An Oracle stored function is a schema object that logically groups a set of SQL and other PL/SQL programming language statements together to perform a specific task and to return a value to the invoker of that function. Applications may use standard Oracle PL/SQL to invoke a stored function; an Oracle stored function may also be invoked as part of an Oracle SELECT, INSERT, DELETE, or UPDATE statement.

Similar to Oracle stored procedures, an Oracle stored function can be located in a local instance of Oracle and in a remote instance. Figure 9–6 shows two stored functions, orafunc1 is a function stored in the ORA1 Oracle instance, while orafunc2 is a procedure stored in the ORA2 Oracle instance.

Figure 9–6   Calling Oracle Stored Function in a Distributed Oracle Environment
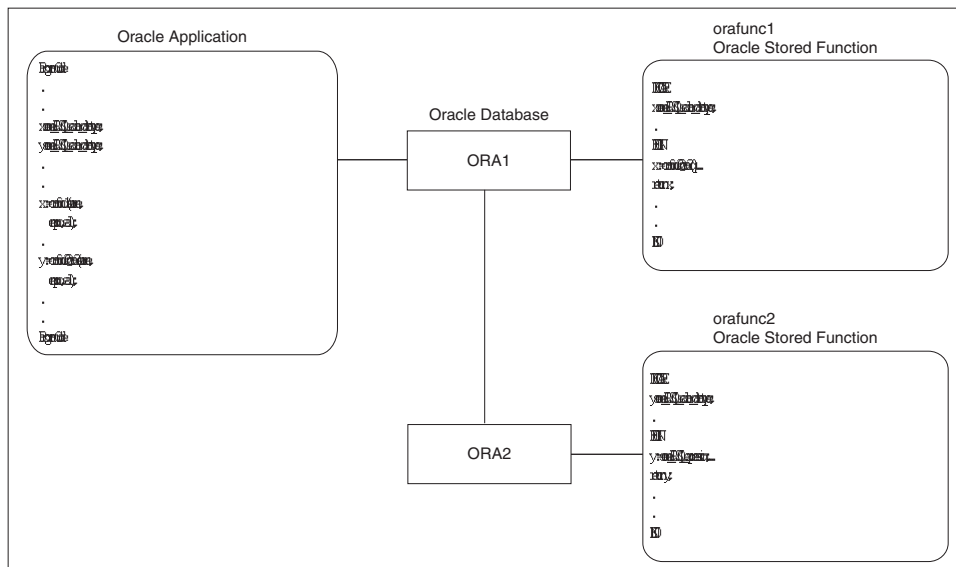
If the application needs to maintain location transparency, then a synonym can be created:

```
CREATE SYNONYM ORAFUNC2 FOR orafunc2@ora2;
```

where orafunc2 is the function that is stored in ORA2.

After this synonym is created, the application no longer needs to use the database link specification for invoking the stored function at the remote Oracle Database.

In Figure 9–6, the reference to function orafunc2 is used to call that same function in the ORA2 instance. In the same way, Oracle can be used to access DB2/400 user-defined functions through the gateway.

In Figure 9–7, empfunc is an Oracle stored function that subsequently accesses data in DB2/400 using the gateway.

Figure 9–7   Using Oracle Stored Functions with DB2/400
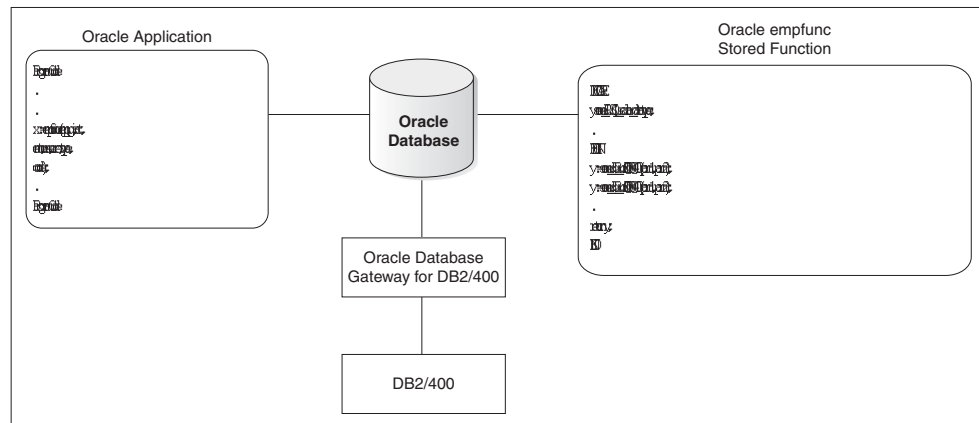
As with the Oracle database, standard PL/SQL is used to create the function. Standard PL/SQL may be used to execute the function or the function may be executed with a reference in a SQL command such as SELECT, UPDATE, INSERT, or DELETE. There is no difference in the gateway case except that the stored function is accessing DB2/400 rather than the Oracle database.

Oracle assumes that the underlying user defined function on the DB2/400 does not change any data on the AS/400 side if the stored function is referenced in a SQL command. One explicitly declares that a stored function does *not* change the AS/400 data by specifying that function name in an IFS file named /HOME/*MYINST*/RDBMS/ADMIN/CALL_NAMES.ORA, where, *MYINST* is your instance name. If you attempt to use an AS/400 stored function (DB2/400 user defined function) in a SQL command without specifying that name in the above IFS file, then you will get the Oracle error 'ORA-06571: function xxxxx does not guarantee not to update database'

Since it is assumed that AS/400 stored functions do *not* change the database, gateway two-phase commit processing does not result from calls made to these stored functions. To effect gateway two-phase commit processing, changes must be made via a stored procedure or via an INSERT, DELETE, or UPDATE SQL command directed to an AS/400 object via the gateway.

> **Note:**   The parameters to a DB2/400 user defined function are all of mode IN. INOUT or OUT parameters are not allowed.

9.6 Using DB2/400 Stored Functions with the Gateway

The procedural feature of the gateway enables execution of DB2/400 native stored functions (also called user defined functions, or UDFs). In other words, the stored function are no longer defined in the Oracle Database, but instead are defined to DB2/400. Oracle PL/SQL may be used to directly call these functions within a PL/SQL block. Or the functions may be called as part of a SELECT, INSERT, DELETE, or UPDATE SQL command.

If the stored function is referenced within a SQL command, then the user must specify its name in the IFS file /HOME/*MYINST*/RDBMS/ADMIN/CALL_NAMES.ORA, where, *MYINST* is your instance name. If you attempt to use an AS/400 stored function (DB2/400 user defined function) in a SQL command without specifying that name in the above IFS file, then you will get the Oracle error 'ORA-06571: function xxxxx does

not guarantee not to update database'. After changing this file, it is recommended that you bring down and then re-start the instance.

Other than the declaration of the function name in the above IFS file, the gateway does not require any other special definitions in order to invoke the DB2/400 stored function.

In Figure 9–8, an Oracle application calls the `empfunc` stored function that is defined in DB2/400.

Figure 9–8 Executing DB2/400 Stored Functions
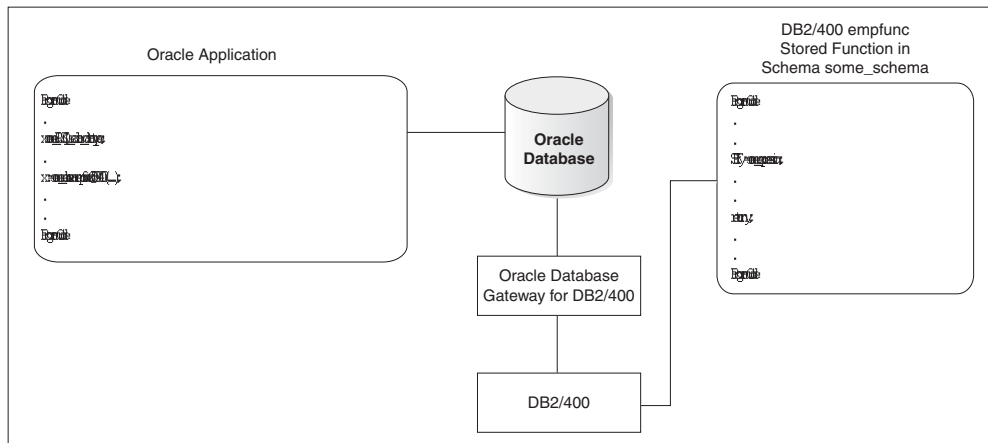
From the perspective of the application, executing the DB2/400 stored function is no different that invoking a stored function at a remote Oracle instance.

9.6.1 Executing DB2/400 Stored Functions from Applications

In order for an application to invoke a DB2/400 stored function, it is first necessary to create the DB2/400 stored function on the DB2/400 system by using the procedures that are documented in the IBM reference for DB2/400.

Next, if you are going to reference that function in a SQL DML statement (such as `SELECT`, `INSERT`, `DELETE`, or `UPDATE`), then you *must* specify the referenced name in the IFS file /HOME/*MYINST*/RDBMS/ADMIN/CALL_NAMES.ORA, where, *MYINST* is your instance name. If you attempt to use an AS/400 stored function (DB2/400 user defined function) in a SQL command without specifying that name in the above IFS file, then you will get the Oracle error 'ORA-06571: function xxxxx does not guarantee not to update database'.

After doing the above, the gateway is able to access the stored function using a standard PL/SQL call syntax or via a reference in a SQL DML statement. For example, suppose our DB2/400 stored procedure gives us what should be a revised-salary given a specific employee name. The employee name, `JOHN SMYTHE`, is passed to the DB2/400 stored function `REVISE_SALARY_FUNC`. The DB2/400 stored function retrieves the salary value from the DB2/400 database and calculates a new revised salary for our specific employee. Since `REVISE_SALARY_FUNC` is a function, it returns this salary value as the return value of the function.

The revised salary returned in `RESULT` is used to update the `SAL` column in the `EMP` table of the Oracle Database as follows:

```
DECLARE
   INPUT VARCHAR2(15);
   RESULT NUMBER(8,2);
```

```
BEGIN
  INPUT := 'JOHN SMYTHE';
  RESULT := MYLIB.REVISE_SALARY_FUNC@DB2400(INPUT);
  UPDATE EMP SET SAL = REUSLT WHERE ENAME := INPUT;
END;
```

> **Note:** The call to the `REVISE_SALARY_FUNC` function may also be done in a `SELECT` statement. That `SELECT` statement would be:
>
> ```
> SELECT MYLIB.REVISE_SALARY_FUNC@DB2400(INPUT) INTO
> RESULT FROM OTGDB2.DUAL@DB2400;
> ```

When the gateway receives a call to execute a DB2/400 stored function, it first does a lookup of the function name in the `QSYS2/SYSFUNCS` and `QSYS2/SYSPARMS` DB2/400 system tables to determine the following:

- The stored function to execute

  The gateway retrieves information from the `ROUTINE_SCHEMA` and `ROUTINE_NAME` columns of the `QSYS2/SYSFUNCS` table in order to locate the actual DB2/400 stored function code that is to be invoked by the gateway. Multiple DB2/400 stored functions can have the same `ROUTINE_NAME`, but they must be uniquely identified by the combination of the `ROUTINE_SCHEMA` and `ROUTINE_NAME` columns of the `QSYS2/SYSFUNCS` table.

  The gateway always receives a qualified stored function name from the Oracle Database server and executes the DB2/400 stored function by using this qualified name. If a qualified name is not explicitly identified by the application, the user ID that is passed over the database link is used to match the value in `ROUTINE_SCHEMA`. To ensure that you execute the correct DB2/400 stored function, Oracle recommends that you use fully qualified stored function names when invoking DB2/400 stored functions from Oracle PL/SQL and from SQL DML commands.

- The parameter list of the stored functions

  When a DB2/400 stored function is defined, the attributes of each entry in the argument list for that stored function are placed in the `QSYS2/SYSPARMS` table. The gateway uses this information to convert the data type of each argument to its corresponding PL/SQL data type. Refer to "Converting DB2/400 Data types to Oracle Data types" on page 9-11 for gateway data type conversion rules.

After the lookup of the description of the DB2/400 stored function is complete, the gateway uses this information to execute the DB2/400 stored function.

9.6.2 Considerations for Using Stored Functions with DB2/400

Following is a list of special considerations for use the procedural feature for stored functions with the Oracle Database Gateway for DB2/400:

- PL/SQL records cannot be passed as parameters when invoking a DB2/400 stored function.

- No matter the context in which the original function call was done, that is either as a PL/SQL function call or as a function call in a SQL DML statement, the gateway executes the function as part of a `SELECT`. That is, it does:

  ```
  SELECT func_call(func_args, ...) FROM OTGDB2.DUAL;
  ```

- The gateway supports both `GENERAL` and `GENERAL WITH NULLS` linkage conventions of DB2/400 stored functions.
  - The `GENERAL` linkage convention means that the parameters that are passed to DB2/400 stored functions cannot be `NULL`.
  - The `GENERAL WITH NULLS` linkage convention means that parameters passed to DB2/400 stored functions can be `NULL` when they are passed using indicator variables.

  Oracle Call Interface (OCI) or embedded PL/SQL can be used in host programs to operate on indicator variables. See the appropriate DB2/400 IBM references on how to detect `NULL`s passed as arguments to a DB2/400 user defined functions.

9.7 Passing DB2/400 SQL Statements Through the Gateway

The passthrough SQL feature allows an application developer to send a SQL statement directly to DB2/400 without the statement being interpreted by the Oracle database. DB2/400 SQL statements that are supported by the gateway through the passthrough facility are limited to non queries (`INSERT`, `UPDATE`, `DELETE`, and DDL statements) and cannot contain bind variables. The gateway can run native DB2/400 SQL statements using the `DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` function.

`DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` is a function that is built into the gateway. This function takes one input argument and returns the number of rows that are affected by the SQL statement. For DDL statements, the function returns zero.

`DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` is a reserved name of the gateway and is used specifically for running native DB2/400 SQL.

This release of Oracle Database Gateway for DB2/400 enables retrieval of result sets from queries that are issued with passthrough. The syntax is different from the `DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` function. Refer to "Retrieving Results Sets Through Passthrough" on page 9-13 for more information.

9.7.1 Using the DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE Function

To execute a passthrough SQL statement by using the `DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` function, use the following syntax:

```
number_of_rows = DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@dblink ('DB2/400_sql');
```

where:

number_of_rows is a variable that is assigned the number of rows that are affected by the passthrough SQL execution. For DDL statements, a zero is returned for the number of rows affected.

dblink is the name of the database link that is used to access the gateway.

DB2/400_sql is a valid DB2/400 nonquery SQL statement (except `CONNECT`, `COMMIT`, or `ROLLBACK`). DB2/400 SQL statements that cannot be dynamically prepared are rejected by DB2/400. The SQL statement that is passed by the `DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE` function must be a character string. For more information regarding DB2/400 SQL statements, refer to the IBM reference for DB2/400.

9.7.2 Examples

Following are examples of the DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE function.

Refer to the next section, "Using Passthrough Statements to Create Tables", for more information.

1. Insert a row into a DB2/400 table using DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE:

```
declare
  num_rows NUMBER;
begin
  num_rows := DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@dblink
  ('INSERT INTO SCOTT.DEPT VALUES ( 10, ''PURCHASING'',
  ''PHOENIX'')');
end;
/
```

2. Create a DB2/400 table using DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE:

```
declare
  num_rows NUMBER;
begin
  num_rows := DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@dblink
  ('CREATE TABLE MYTABLE (COL1 INTEGER, COL2 INTEGER, COL3 CHAR(14),
  COL4 VARCHAR(13))');
end;
/
```

9.7.3 Using Passthrough Statements to Create Tables

If you use DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE to create a DB2/400 table, then one of the following is required for journaling:

- a QSQJRN journal in the target library

- a target library that was created as a collection

  (This requirement applies if the DB2/400 isolation level is set to *CHG, which is the default.)

If neither of the above is true, then you will receive an error message that the table has been created but cannot be journaled. In this case, you can disregard the error message and journal the file manually. For more information on journaling, refer to "Step 2: Verify Journaling" on page 4-7.

9.7.4 Retrieving Results Sets Through Passthrough

Oracle Database Gateway for DB2/400 provides a facility to retrieve results sets from a SELECT SQL statement that is issued through passthrough. In the example below, the values for a single field are selected from all the rows of the foreign table PT_TABLE, and these values are inserted into the local Oracle Database table PT_TABLE_LOCAL. Refer to *Oracle Database Administrator's Guide* for additional information.

9.7.4.1 Example

```
DECLARE
```

```
                      CRS binary_integer;
                      RET binary_integer;
                      VAL VARCHAR2(10)
                 BEGIN
                      CRS:=DBMS_HS_PASSTHROUGH.OPEN_CURSOR@gtwlink;
                      DBMS_HS_PASSTHROUGH.PARSE@gtwlink(CRS,'SELECT NAME FROM PT_TABLE');
                 BEGIN
                      RET:=0;
                      WHILE (TRUE)
                      LOOP
                        RET:=DBMS_HS_PASSTHROUGH.FETCH_ROW@gtwlink (CRS,FALSE);
                        DBMS_HS_PASSTHROUGH.GET_VALUES@gtwlink (CRS,1,VAL);
                        INSERT INTO PT_TABLE_LOCAL VALUES (VAL);
                      END LOOP;
                      EXCEPTION
                        WHEN NO_DATA_FOUND THEN
                          BEGIN
                            DBMS_OUTPUT.PUT_LINE('END OF FETCH');
                            DBMS_HS_PASSTHROUGH.CLOSE_CURSOR@gtwlink(CRS);
                          END;
                        END;
                 END;
                 /
```

9.8 Converting DB2/400 Data Types to Oracle Data Types

To move data between applications and the underlying database, the gateway binds
data values from a host variable or literal to a data type that is understood by the
underlying database.

Oracle tools expect Oracle data types. Consequently, the gateway maps values from
DB2/400 servers into appropriate Oracle data types before passing these values back
to the application or Oracle tool. The following table lists the data type mapping and
restrictions:

Table 9–1   Data Type Mapping and Restrictions

DB2/400	Oracle	Criteria	Comments
CHAR(N)	CHAR(N)	$1 \le N \le 255$	Maximum length of a DB2/400 CHAR column is 32766 characters.
	VARCHAR2(N)	$255 < N \le 4000$	
	LONG	$4000 < N$	
VARCHAR(N)	VARCHAR2(N)	$N \le 4000$	Maximum length of a DB2/400 VARCHAR column is 32740 characters
	LONG	$4000 < N \le 32740$	
LONG VARCHAR(N)	VARCHAR2(N)	$N \le 4000$	Maximum length of a DB2/400 VARCHAR column is 32740 characters.
	LONG	$4000 < N \le 32740$	
VARCHAR(N) for Bit Data	RAW(N)	$1 \le N \le 255$	Maximum length of a DB2/400 VARCHAR column is 32740 characters.
	LONG RAW(N)	$255 < N \le 32740$	

Table 9–1   (Cont.) Data Type Mapping and Restrictions

DB2/400	Oracle	Criteria	Comments
DATE	DATE	Refer to"Performing Date and Time Operations" on page 9-18.	Refer to "Performing Date and Time Operations" on page 9-18.
TIME	CHAR(8)		
ROWID	RAW(40)		
# TIMESTAMP	CHAR(26)		See note below table
# TIMESTAMP	TIMESTAMP		Based on ORATIMSTMP environment variable. See note below table
BLOB	LONG RAW		
CLOB	LONG		
DBCLOB	LONG		
* GRAPHIC(N)	CHAR(2*N) VARCHAR2(2*N)	$1 \leq N \leq 127$ $127 < N \leq 16370$	Maximum length of a DB2/400 GRAPHIC column is 16370 graphic characters
* VARGRAPHIC(N)	VARCHAR2(2*N) LONG(2*N)	$1 \leq N \leq 2000$ $2000 < N$	Maximum length of a DB2/400 VARGRAPHIC column is 16370 graphic characters
* LONG VARGRAPHIC(N)	VARCHAR2(2*N) LONG(2*N)	$1 \leq N \leq 2000$ $2000 < N$	Maximum length of a DB2/400 VARGRAPHIC column is 16370 graphic characters
FLOAT(N)	FLOAT(21)	$1 \leq N \leq 21$	
FLOAT(N)	FLOAT(53)	$22 \leq N \leq 53$	
DECIMAL(P,S)	NUMBER(P,S)		Packed decimal in OS/400
NUMERIC(P,S)	NUMBER(P,S)		Zoned decimal in OS/400
BIGINT	NUMBER(19)		
INTEGER	NUMBER(10)		
SMALLINT	NUMBER(5)		

 # Table Note: TIMESTAMP appears in two rows in the table. The data type that is used depends on the value in the ORATIMSTMP DataArea. If the DataArea is missing, then the CHAR(26) value is used. If the DataArea exists and has value ON, then the Oracle TIMESTAMP data type is used.

* Table Note: If the column CCSID is 13488 (UCS-2), and if the gateway `NLS_LANG` specifies a single-byte character set, then replace `2*N` in the Oracle Database column with `N`, replace `127` with `255`, and replace `2000` with `4000`. If the column `CCSID` is `13488`, and if the Oracle Database is expecting single-byte data, then `(VAR)GRAPHIC`s are treated as `(VAR)CHAR`s.

9.8.1 DB2/400 GRAPHIC Support

Oracle database no longer supports `NCHAR` or `NVARCHAR2` data types. With Oracle Database 10*g* server, the Oracle Database Gateway for DB2/400 supports double-byte data types (`GRAPHIC` and `VARGRAPHIC` in DB2/400 terms) in a manner similar to the manner in which they were supported in version 4 of the Oracle Database Gateway for DB2/400. That is to say, in Oracle tools and applications, they are supported as `CHAR` and `VARCHAR2` data types of the appropriate size.

To support DB2/400 columns of `GRAPHIC` or `VARGRAPHIC` type, you must set `NLS_LANG` to a language in which the character set part corresponds to the CCSID of the `(VAR)GRAPHIC` column. This way, if the column is `CCSID 933`, then the character set part of `NLS_LANG` must be set to `KO16DBCS` (refer to Appendix B, "Globalization Support"). In addition, you must specify on the `CHGORATUN` screen that you want V4 graphics compatibility, and the gateway job should be running in the CCSID of the `GRAPHIC`s columns. The job `CCSID` is also set on the `CHGORATUN` panel.

9.8.2 For DBCS Users

The Oracle Database Gateway for DB2/400 allows you to use DBCS (Double Byte Character Set) data under limited conditions, which include a risk for data integrity.

9.8.2.1 Double-Byte Character Support

This version of the Oracle Database Gateway for DB2/400 allows you to access and change columns in DB2/400 that are designated as `VARGRAPHIC` or `GRAPHIC`. Because of the IBM definition of a `(VAR)GRAPHIC` column, you can insert *only* double-width characters into `(VAR)GRAPHIC` columns — in other words, the string to be inserted must consist entirely of DBCS characters. Attempts to insert a mixed-byte string into a `(VAR)GRAPHIC` column will result in an error. For example, attempting to insert a value represented by a mixed-byte string constant into a `(VAR)GRAPHIC` column will result in a SQL0105 error: "Mixed or Graphic String Constant Not Valid."

An ASCII client might attempt to use the following `INSERT` statement:

```
INSERT INTO mytable.graphcol@tg4db2400 values('AxxB')
```

where `"A"` and `"B"` are the normal ASCII single-width characters and `"xx"` designates a double-width character in one of the Far Eastern ASCII-based character sets such as `JA16SJIS`, `ZHS16CGB213280`, `ZHT16BIG5`, or `KO16KSC5601`. There is no unambiguous way in which to insert such data into a `(VAR)GRAPHIC` column. The problem exists because no DBCS character corresponds to each of the single-byte characters.

This version of the Oracle Database Gateway for DB2/400 provides a workaround, but the workaround carries a risk for data integrity. If you desire to choose this path, then you must realize that if you attempt to `INSERT` a string with a single-byte character, then that string may be returned to you entirely as a DBCS string in a subsequent `SELECT` as a double-byte character. The rules are as follows:

1. With no options specified, the user must ensure that any strings to be `INSERT`ed into `(VAR)GRAPHIC` columns are in their "wide" form on the ASCII computer. The

string can*not* contain any single-byte characters. This means that when converting from the ASCII-based string to a DBCS string, the conversion does not result in a DBCS string with a `SHIFT-OUT` or `SHIFT-IN` character at any other position than the first character position for the `SHIFT-OUT` character or the last character position for the `SHIFT-IN` character. The same is true for values that are represented by constants or for values that are represented by bind-variables.

2. Specifying the `FORCE_SB` option causes all single-byte characters to be translated to their corresponding double-byte values on input — through either an `INSERT` or a bind-variable. The entire string is then `INSERT`ed into the `(VAR)GRAPHIC` column. On output, that is to say, on a `SELECT`, each `(VAR)GRAPHIC` column will be examined for double-byte characters that correspond to single-byte characters. Each such character is replaced by its corresponding single-byte character.

This is where the data integrity problem arises. Because single-byte characters have been forced to their corresponding double-byte characters on input, there is no way to know if a double-byte character actually came from the translation of an ASCII wide character or from the process of forcing single-byte characters to the corresponding double-byte characters.

As an example, use the previous `INSERT` statement, which is:

```
INSERT INTO mytable.graphcol@tg4db2400 values('AxxB')
```

If you use the `FORCE_SB` option, then the resultant DBCS value in the DB2/400 column may look like the following:

```
wAyywB
```

where `"wA"` is the DBCS correspondence (usually 0x42C1) for the single byte `'A'`, `"wB"` is the DBCS correspondence (usually 0x42C2) for the single byte `'B'`, and `"yy"` is the DBCS character corresponding to the ASCII-based, double-width `'xx'` character.

When `SELECT`ing from this column, on the client, you would get exactly what you `INSERT`ed, that is, `'AxxB'` which is good. But, what if the DBCS character that is represented in the DB2/400 column by `"wa"` had actually been `INSERT`ed into the column through a valid ASCII representation for a wide A? With the `FORCE_SB` option, you get a single-byte A on the client. This may not be exactly what you wanted.

In the end, it is you, the customer, who must decide whether this option is valuable to you. You must decide if you can accept the possible problems that can arise.

The `ORADBMBOPT` Data Area (or Environment variable) controls this feature. If no `ORADBMBOPT` Data Area (or Environment variable) is present, then no forcing of single-byte character to double-byte character will take place when transferring data into DB2/400 columns, and no forcing of double-byte characters to the single-byte correspondences will take place when transferring data from DB2/400 columns. Placing the string "FORCE_SB" into `ORADBMBOPT` turns the feature `ON`.

Note that you should *not* use PassThrough to execute any SQL commands that contain graphic constants, unless those constants conform fully with DB2/400 graphic constants. DB2/400 graphic constants start with G' (G apostrophe) or N' (N apostrophe) and end with an apostrophe (') character. The first character after the G' or N' must be a `SHIFT-OUT` character, and the character preceding the apostrophe (') at the end of the constant must be a `SHIFT-IN` character. Between the `SHIFT-OUT` and `SHIFT-IN` characters, an even number of bytes must be present, with each byte pair making up a single DBCS character.

9.8.3 Performing Character String Operations

The gateway performs all character string comparisons, concatenations, and sorts by using the data type of the referenced columns (for example, fixed or variable length). The gateway determines the validity of character string values that are passed by applications and automatically converts character strings from one data type to another. It also automatically converts between character strings and dates when necessary.

In contrast, the Oracle Database 10*g* server processes character strings as variable length character strings. The Oracle database performs character string comparisons, concatenations, and sorts by using variable length character string representations. The Oracle database never pads or truncates character strings. The Oracle database also automatically converts among numbers, character strings, and dates when necessary.

9.8.4 Converting Character String Data Types

The gateway binds character string data values from host variables as fixed length character strings. The bind length is the length of the character string data value. This conversion is performed on every bind.

DB2/400 performs automatic conversions to the character string data type (for example, fixed length and variable length) of the destination column. You have no control over the conversion. The conversion might be independent of the data type of the destination column in the database. For example:

```
UPDATE EMP@AS400 SET ENAME = 'BLAKE' WHERE EMPNO = 7654;
```

If `ENAME` is a `CHAR(9)`, fixed-length column in the `ENAME` table, then the update inserts 'BLAKE' as the last name of employee number `7654`. `BLAKE` is padded with four trailing spaces by the host database to make it conform to the fixed length definition of the column.

The gateway returns data bound in the manner that is requested by the application. For character strings, the format is fixed length string.

The DB2/400 `VARCHAR` data type can be from 1 to 32740 bytes in length. This data type is converted to an Oracle `VARCHAR2` data type if it is between 1 and 4000 characters in length. If it is between 4001 and 32740 characters in length, then it is converted to an Oracle `LONG` data type.

The Oracle `LONG` data type can be from 1 byte to 2 GB in length. Because the DB2/400 `VARCHAR` data type can be no longer than 32740 bytes, you receive an error message if you attempt to insert data longer than 32740 bytes into a DB2/400 `VARCHAR` or `LONG VARCHAR` data type.

9.8.5 Performing Date and Time Operations

The implementation of date and time data differs significantly in DB2/400 and the Oracle database. The Oracle database has a single date data type, `DATE`, that can contain both calendar date and time-of-day information. DB2/400 supports the following date and time data types:

`DATE` is the calendar date only.

`TIME` is the time of day only.

`TIMESTAMP` is a numerical value that combines a calendar date and time-of-day with microsecond resolution.

There is no mechanism that translates the IBM `TIME` and `TIMESTAMP` data to Oracle `DATE` data. An application must process `TIME` data types in the Oracle `CHAR` format with a length of 8 bytes. An application must process the `TIMESTAMP` data type in the Oracle `CHAR` format with a length of 26 bytes.

An application reads `TIME` and `TIMESTAMP` columns as character strings and converts portions or subsets of the string to perform numerical operations. `TIME` and `TIMESTAMP` values can be sent to DB2/400 as character literals or bind variables of the appropriate length and format.

Oracle `DATE` data types and IBM `DATE` data types are mapped to each other. If an IBM `DATE` is queried, then it is converted to an Oracle `DATE` with a zero (midnight) time of day. If an Oracle `DATE` is processed against an IBM `DATE` column, then the date value is converted to the IBM `DATE` format and any time value is discarded. DB2/400 columns of `DATE` or `TIME` data type are assumed to be in ISO format. For `DATE`, the column format is "yyyy-mm-dd", and for `TIME` columns, the format is "hh.mm.ss". TIME is in a 24-hour format.

Character representations of dates are different in Oracle format and DB2/400 format. When an Oracle application SQL statement contains a date literal or conveys a date through a character bind variable, the gateway must convert the date to a DB2/400 compatible format.

The gateway does not automatically recognize when a character value is going to be processed against an IBM `DATE` column. Applications are required to distinguish character date values by enclosing them with the Oracle `TO_DATE( )` function notation.

For example, if `EMP` is a synonym or view that is accessing data in DB2/400, then instead of using the following SQL statement:

```
SELECT * FROM EMP@AS400 WHERE HIREDATE = '03-MAR-81'
```

you must use the following syntax:

```
SELECT * FROM EMP@AS400 WHERE HIREDATE = TO_DATE('03-MAR-81')
```

In a programmatic interface program using a character bind variable for the qualifying date value, you must use the following SQL statement:

```
SELECT * FROM EMP@AS400 WHERE HIREDATE = TO_DATE(:1)
```

This SQL notation does not affect SQL statement semantics when the statement is executed against an Oracle table. The statement remains the same across Oracle and DB2/400 data stores.

The `DATE` requirement does not apply to input bind variables in Oracle date seven byte binary format. The gateway recognizes such values to be dates.

> **Note:** Special considerations exist for using date or time data types in stored procedures with DB2/400. Refer to "Known Problems" on page 2-2 for more information.

9.8.6 Dates in the 21st Century

Oracle recommends that you set the Oracle Database 10*g* server default `NLS_DATE_FORMAT` parameter to a format including a four-digit year. Use the `TO_DATE` function to enter dates in the twenty-first century. You can use any date format which includes a four-character field for the year. For example,

TO_DATE('2008-07-23', 'YYYY-MM-DD') can be used in any SELECT, INSERT, UPDATE, or DELETE statement.

9.8.7 Performing Numeric Data Type Operations

DB2/400 performs automatic conversions to the numeric data type of the destination column such as BIGINT, INTEGER, SMALLINT, PACKED DECIMAL, and NUMERIC. You have no control over the data type conversion, and the conversion is independent of the data type of the destination column in the database.

For example, if PRICE is an integer column of the PRODUCT table in DB2/400, then the update shown in this example inaccurately sets the price of an ice cream cone to $1.00, because DB2/400 automatically converts a double-precision floating point to an integer:

```
UPDATE PRODUCT@AS400
SET PRICE = 1.50
WHERE PRODUCT_NAME = 'ICE CREAM CONE    ';
```

Because PRICE is an integer (in this example), the DB2/400 server automatically converts the decimal data value of 1.50 to 1.

The gateway returns values that are bound in the manner that is requested by the Oracle application. For numbers, the format is typically an Oracle number, limited to a certain number of digits of precision by the application.

9.9 SQL Functions

One of the most important features of the Oracle Database Gateway product family is the ability to provide SQL transparency to the user and to the application programmer. Foreign data store SQL functions can be categorized into three areas (compatible, translated, and compensated):

1. Compatible

   Compatible SQL functions have the same meaning and results on both Oracle and foreign data store. Compatible SQL functions include both column and scalar functions.

   Column Functions

   - AVG
   - COUNT (*) only
   - COUNT (DISTINCT expression)
   - MAX
   - MIN
   - STDDEV (one argument)
   - SUM

   Scalar Functions

   - ABS
   - COS
   - COSH
   - EXP

- `LENGTH (char) only`

- `LN`

- `MOD`

- `POWER`

- `SIN`

- `SINH`

- `SQRT`

- `TAN`

- `TANH`

- `UPPER`

- `VARIANCE (one argument)`

2. Translated

   Translated SQL functions provide the same functionality, but are referenced by different names at the Oracle server and the foreign data store.

   Translated SQL functions include:

   Table 9–2   Some Translated SQL Functions

Oracle	DB2/400
`NVL`	`VALUE`
`LOWER`	`TRANSLATE`
`||` operator	`CONCAT`

   > **Note:**   Use the passthrough feature for native DB2/400 SQL statements when using the `VALUE`, `TRANSLATE`, or `CONCAT` functions. Refer to "Passing DB2/400 SQL Statements Through the Gateway" on page 9-12 for additional information.

3. Compensated

   Compensated SQL functions are advanced SQL functions that are supported by the Oracle Database and that cannot be expressed or recognized by the foreign data store.

   SQL compensation in the gateways enriches the semantics of the native SQL of a remote data source, such as DB2/400. This important feature of the gateway allows application developers and users to leverage the advanced features of the Oracle Database.

   Oracle SQL functions that are not listed in the compatible or translated lists includer earlier are compensated SQL functions.

9.10 Oracle Server SQL Construct Processing

Special circumstances and restrictions for processing.

9.10.1 SELECT without the FOR UPDATE Clause

A `SELECT` without the `FOR UPDATE` clause can be handled in one of three ways:

- If the entire `WHERE` clause of the `SELECT` statement is acceptable syntax for DB2/400, then it is given to DB2/400 to perform.

- If part, but not all, of the `WHERE` clause of the `SELECT` statement uses features that are not available in DB2/400, then the `WHERE` clause is split between the DB2/400 system and the Oracle database.

   The portion of the `WHERE` clause that is acceptable for DB2/400 is sent to DB2/400. The Oracle database post processes the results of the DB2/400 `SELECT` and applies the Oracle-specific `WHERE` clause elements. This results in DB2/400 doing as much of the `WHERE` clause as possible.

- If the entire `WHERE` clause is not acceptable for DB2/400, then an unqualified `SELECT` (without the `WHERE` clause) is sent for DB2/400, and the Oracle database post processes the entire `WHERE` clause.

The Oracle database post processes `SELECT` statements without the `FOR UPDATE` clause. Most Oracle `SELECT` statements are supported. One exception is the `CONNECT BY` clause.

9.10.2 SELECT FOR UPDATE, INSERT, and DELETE Clauses

DB2/400 must process the entire `SELECT FOR UPDATE`, `INSERT`, and `DELETE` clauses. The Oracle server cannot post process these clauses. Only SQL that is a common subset of Oracle and DB2/400 SQL can be used with these statements.

The following rules exist for the use of `SELECT FOR UPDATE`, `INSERT`, and `DELETE` clauses:

- Only Oracle syntax that is also valid for DB2/400 can be used. For DB2/400 SQL syntax, refer to the IBM reference for DB2/400.

- The following Oracle functions are supported with all options:

  - `AVG`

  - `MAX`

  - `MIN`

  - `SUM`

  - `TO_DATE`

- The `NOWAIT` option of the `FOR UPDATE` clause of the `SELECT` statement is not supported.

- Although DB2/400 requires a list of column names in the `FOR UPDATE` clause, the unqualified Oracle syntax `FOR UPDATE` with no column names is accepted. The gateway derives the column list from the `SELECT` result column list.

9.11 Oracle Server and DB2/400 Differences

Differing functions and operations between the two databases are explained below.

9.11.1 Oracle Bind Variables

Oracle bind variables become DB2/400 parameter markers when used with the gateway. Therefore, the bind variables are subject to the same restrictions as DB2/400 parameter markers. For example, the following statement fragments are not allowed:

```
WHERE :x IS NULL
WHERE :x = :y
```

For more information about DB2/400 parameter marker restrictions, refer to the IBM reference for DB2/400.

9.11.2 Performing Zoned Decimal Operations

A zoned decimal field is described as packed decimal on an Oracle database. However, an Oracle application such as a Pro*C program can insert any supported Oracle numeric data type into a zoned decimal column. The gateway converts this number into the most suitable data type. Data can be retrieved from DB2/400 into any Oracle data type provided it does not result in loss of information.

9.12 Oracle Data Dictionary Emulation in a DB2/400 Server

The gateway can optionally augment DB2/400 database catalogs with data dictionary views that are modeled after the Oracle Database data dictionary. These views are based on the dictionary tables in DB2/400, presenting the catalog information in views familiar to Oracle Database users.

The views that are created during the installation of the gateway automatically limit the data dictionary information that is presented to each user, based on the privileges of that user.

9.12.1 Using the Gateway Data Dictionary

The gateway data dictionary views provide you with an Oracle-like interface to the contents and use of the DB2/400 data dictionary. Some of these views are required by Oracle products.

You can query the gateway data dictionary views to see the objects in DB2/400 and to determine the authorized users of those objects.

Refer to Appendix C, "Data Dictionary Views" for descriptions of DB2/400 catalog views.

9.12.2 DB2/400 Special Registers

You can access DB2/400 special registers by using the gateway. During installation of the gateway, a DB2/400 view is created in order to enable access to special registers. For example, to find out the primary authorization ID that is being used by the gateway, the following command from your application:

```
SELECT CURRENT_USER FROM OTGDB2.OTGREGISTER@DB2400
```

where *OTGDB2* is the default qualifier of the OTGREGISTER view, and *DB2400* is the name of a database link to the gateway.

9.13 Oracle Developer Forms Compatibility

You can use Oracle Developer Forms version 4 or higher to build applications that are stored in DB2/400. When you connect to a non-Oracle data source with a gateway product, you should be aware of four transaction processing options in Oracle Developer Forms. These options include two block properties and two form module properties. When designing the form, set these options as described in the following table:

Table 9–3   Transaction Processing Options in Oracle Developer Forms

Option	Set Value To
Key Mode block property	Anything except Unique_key
Locking Mode block property	Delayed
Cursor Mode form module	Close_at_commit
Savepoint Mode form module	Off

You should also ensure that, at item and block level, the primary key is set to `TRUE`.

For more information, refer to your Oracle Forms documentation.

10

Error Messages, Diagnosis, and Reporting

This chapter discusses error messages that are generated by Oracle Database Gateway for DB2/400, the diagnosis of suspected Oracle errors, and the requirements for documenting these errors to Oracle Support Services.

This chapter contains the following sections:

- Message and Error Code Processing on page 10-1
- Contacting Oracle Support Services on page 10-3
- Error Categories on page 10-3

10.1 Message and Error Code Processing

The gateway architecture includes a number of separate components. Any of these components can detect and report an error condition while processing a SQL statement that refers to one or more DB2/400 database tables. An error condition can be complex, involving error codes and supporting data from multiple components. In all cases, the application receives a single Oracle error code on which to act.

When possible, an error code from DB2/400 is converted to the Oracle error code that is associated with the same logical condition.

Error code mapping is provided to support application designs that test for and act upon specific error conditions. The set of mapped errors is limited to those errors that are associated with conditions that are common to most relational databases.

10.1.1 Mapping DB2/400 Error Messages to Oracle Error Messages

DB2/400 error messages (SQLSTATE codes) are mapped to Oracle database error codes. Notice that multiple DB2/400 SQLSTATE codes can refer to the same Oracle database error code.

Table 10–1 SQLSTATE Codes Mapped to Oracle Server Error Codes

Description	SQLSTATE Code	Oracle Server Error Code
No rows selected	02000	0
Unique index constraint violated	23505	ORA-0001
Table or view does not exist	52004 or 42704	ORA-00942
Object name too long. Therefore, object does not exist	54003 or 42622	ORA-00942

Table 10–1 (Cont.) SQLSTATE Codes Mapped to Oracle Server Error Codes

Description	SQLSTATE Code	Oracle Server Error Code
Insufficient privileges	42501	ORA-01031
Divisor is equal to zero	01519 or 01564	ORA-01476

10.1.2 Interpreting Gateway Message Formats

Error messages are generally accompanied by additional message text, beyond the text associated with the Oracle message number. The additional text includes details about the error.

Most gateway messages exceed the 70-character message area in the Oracle SQLCA. Use SQLGLM or OCIErrorGet in the programmatic and Oracle Call Interfaces that you use with the gateway to view the entire message. Refer to the *Oracle Database PL/SQL User's Guide and Reference* for information about SQLGLM and the *Oracle Call Interface Programmer's Guide* for information about OCIErrorGet.

Gateway messages use the following format:

```
ORA-nnnnn error_message_text
DB2/400_error_messages
ORA-02063: preceding n lines from dblink
```

for example:

```
select * from scott.dummy@DB2link
ERROR at line 1:
ORA-00942 = table or view does not exist
SQL0204: DUMMY in SCOTT type * FILE not found
ORA-02063 preceding 2 lines from dblink
```

where:

nnnnn is an Oracle error number. If *nnnnn* is between 28500 and 28559, then the message is from the gateway. If it is not in this range, then it is a mapped error message. In the example above, the value for *nnnnn* is **00942.**

error_message_text is the text of the message that is associated with the error. In the example above, the value for *error_message_text* is **table or view does not exist.**

DB2/400_error_messages are additional messages that are generated by DB2/400. In the example above, the value for *DB2/400_error_messages* is **SQL0204 : DUMMY in SCOTT type *FILE not found**.

n is the total number of DB2/400_error_messages lines that are referenced above the code entry line. In the example above, the value for *n* is **2.**

dblink is the name of the database link that is used to access the gateway. In the example above, the value for *dblink* is **DB2link.**

10.1.3 Diagnosing Errors Detected by the Oracle Database 10*g* Server

If an error is detected by the Oracle Database 10*g* server, then the DB2/400 error messages do not occur. For example, if the gateway cannot be accessed because of an Oracle Net or gateway installation problem, then the DB2/400 error message is not present in the received error message.

Another example of Oracle database error messages without DB2/400 error messages occurs when an INSERT statement attempts to insert data into a table but does not

include values for all of the columns in the table. This SQL statement causes an error message as follows:

```
SQL> INSERT INTO EMP@AS400 VALUES(9999);
ERROR at line 1:
ORA-00947: NOT ENOUGH VALUES
```

The ORA-00947 message is not accompanied by DB2/400 error messages because the error is detected by the Oracle database. The Oracle database obtains a description of the DB2/400 table before sending the INSERT statement to the gateway for processing. This allows the server to detect when the INSERT statement is invalid.

10.2 Contacting Oracle Support Services

To maximize the effectiveness of your inquiries, gather the following information before contacting Oracle Support Services:

- OS/400 release number
- gateway release number (found in the JobLog for a Gateway task)
- Oracle Net release number (found in the JobLog for a Gateway task)
- Oracle database release number and platform

10.3 Error Categories

Use the following error categories to describe the error:

- documentation errors
- incorrect output
- Oracle external error
- abnormal termination
- program loop
- performance
- missing functionality

10.3.1 Documentation Errors

When reporting documentation errors, you are asked to provide the following information:

- document name
- document part number
- date of publication
- page number

Describe the error in detail. Documentation errors can include erroneous documentation and omission of required information.

10.3.2 Incorrect Output

In general, an incorrect output error occurs whenever an Oracle utility produces a result that differs from written Oracle product documentation. When describing errors

of incorrect output, you need to describe, in detail, the operation of the function in error. Be prepared to describe your understanding of the proper function, the specific Oracle documentation that describes the proper operation of the function, and a detailed description of the incorrect operation.

If you think you have found a software bug, then be prepared to answer the following questions:

- Does the problem occur in more than one Oracle tool? (Examples of Oracle tools are SQL*Plus and Oracle Developer.)

- What are the exact SQL statements that are used to reproduce the problem?

- What are the full version numbers of the AS/400 system, Oracle Database, Oracle gateway, and related Oracle software?

- What is the problem, and how is it reproduced?

10.3.3 Oracle External Error

Oracle error messages are produced whenever an Oracle gateway, server, tool, or DB2/400 system detects an error condition. Depending on the circumstances, error messages might be fatal or nonfatal to the utility or server.

Be prepared to identify the exact error message and message number that are received and the complete circumstances surrounding the error.

10.3.4 Performance

AS/400 system performance is determined by many factors, most of which are not within the control of Oracle. Considerations such as system load, I/O topology, network topology use, and DB2/400 resource availability and use, make the documentation of performance errors difficult.

Provide detailed information about the state of your environment when reporting an error. Specific documentation might include:

- CPU type and memory configuration

- database topology

- I/O topology

- network topology

- system workload by type

- Oracle workload characterization

- query execution plans

- DB2/400 resource information

10.3.5 Missing Functionality

Enhancement requests can be opened with Oracle Support Services to request the inclusion of functions and features that are not currently available with Oracle products. When opening an enhancement request, describe the specific feature or function to be added to the product, and provide a business case to justify the enhancement.

A

Oracle Net Files and Members

This appendix lists and defines the member names in the `LOG`, `ORA`, and `TRC` gateway library files

This appendix contains the following sections:

- File Members on page A-1
- Agent Trace Files on page A-3

A.1 File Members

Members are listed for the following file types:

- LOG File Members
- ORA File Members
- TRC File Members

A.1.1 LOG File Members

The `LOG` file contains the `LISTENER` member, which is the default log output member for the Oracle Net TCP/IP listener.

To display this member on the AS/400, use the following command:

```
WRKMBRPDM instance_name/LOG
```

where *instance_name* is the name that is used for the gateway when it is installed.

In the LOG file, which is in the instance library, the `LISTENER` member records information about the startup of the listener and also records information about every connection attempt that is made to the associated instance. The file can get quite large over time, and you should therefore occasionally clear out the `LISTENER` member entries by using the `CLRPFM FILE` command. To issue the command, you need to sign on to your iSeries computer with either the `QSECOFR` ID or the instance ID and then enter the following command:

```
CLRPFM FILE(instance_name/LOG) MBR(LISTENER)
```

The `LISTENER` member can also be displayed using the `WRKTRCLOG` command. The gateway instance library should be in the library list when you use the `WRKTRCLOG` command.

A.1.2 ORA File Members

The `ORA` file contains the following members:

A.1.2.1 LISTENER

Member `LISTENER` contains the parameter values for the listener. The values in this file should not be changed directly with an AS/400 editor. The most commonly used parameters can be changed with the `CHGORANET` command. All other values must be changed in the `ORA(LISTENER_CHG)` file. Refer to "CHGORANET, Change Network Parameters" on page 6-5 for more information.

A.1.2.2 LISTENER_CHG

Member `LISTENER_CHG` is used to make changes to the `ORA(LISTENER)` file. Refer to "CHGORANET, Change Network Parameters" on page 6-5 for more information.

A.1.2.3 LISTENLOCK

Member `LISTENLOCK` is the `ORA(LISTENLOCK)` member. It is used to ensure that only one instance of the `RUNORALSN` program is running. The `RUNORALSN` program runs the TCP/IP listener. Do not modify this file.

A.1.2.4 SQLNET

Member SQLNET contains the network parameter values for Oracle Net. The values in this file should not be changed directly with an AS/400 editor. The most commonly used parameters can be changed with the `CHGORANET` command. All other values must be changed in the `ORA(SQLNET_CHG)` file. Refer to "CHGORANET, Change Network Parameters" on page 6-5 for more information.

A.1.2.5 SQLNET_CHG

Member SQLNET_CHG is used to make changes to the `ORA(SQLNET)` file. Refer to "CHGORANET, Change Network Parameters" on page 6-5 for more information.

A.1.2.6 Displaying File Members

To display these members on the AS/400, use the following command:

```
WRKMBRPDM instance_name/ORA
```

where `instance_name` is the name that is used for the gateway when it is installed. You may also use the command `WRKTRCLOG`, but you must first make sure that the instance library is in the library list before using this command.

A.1.3 TRC File Members

The TRC file contains the following members, where `xxxxxx` is numeric (the job number):

- `LISTxxxxxx` is the default trace output file for the Oracle Net TCP/IP listener, where `xxxxxx` is the job number of the listener.

- `SERVxxxxxx` is the Oracle Net trace for server job number `xxxxxx`.

To display these members on the AS/400, use the following AS/400 command:

```
WRKMBRPDM instance_name/TRC
```

where *instance_name* is the name that is used for the gateway when it is installed. Alternatively, you may use the command `WRKTRCLOG`, but you must first make sure that the instance library is in the library list before using this command.

A.1.4 ORA files in the Integrated File System (IFS)

There is a `.ORA` file, `CALL_NAMES.ORA`, in the IFS. This file is used in conjunction with DB2/400 SQL User Defined Functions (UDF) that are to be called through the gateway. Oracle must be aware of all the DB2/400 UDFs. An attempt to use a UDF that does not appear in this file will result in an ORA-06571 error. The file is located at /home/myinst/rdbms/admin/call_names.ora where 'myinst' is the name of your gateway instance. The file may be edited with the EDTF system editor. The names of the UDFs are found in this file, at most one name in each line. The name may be qualified by the name of the library in which a UDF resides, e.g., `"MYORAINST.MYFUNC"`.

A.2 Agent Trace Files

Agent trace files are now located within the integrated file system in the directory /home/*myinst*/rdbms/log, where *myinst* is the instance name. The file names are of the form *myinst_agt_nnnnnn.trc,* where *myinst* is the instance name, and *nnnnnn* is the job number. To display these files, use the `WRKTRCLOG` command, but ensure that the instance library is in the library list before doing so.

B

Globalization Support

This appendix documents the Globalization Support information for the Oracle Database Gateway for DB2/400. For more information about using Globalization Support, refer to *Oracle Database Application Developer's Guide - Fundamentals*.

This appendix contains the following sections:

- Overview of Globalization Support on page B-1
- Gateway Configuration on page B-1
- DB2/400 Coded Character Set Considerations on page B-2
- Oracle Server and Client Configuration on page B-4
- Message Availability on page B-5
- DB2/400 GRAPHIC Support on page B-5

B.1 Overview of Globalization Support

Globalization Support is a technology that enables Oracle applications to interact with users in their native language, using the conventions of that language for displaying data. The Oracle Globalization Support architecture is datadriven, enabling support for specific languages and character encoding schemes to be added without requiring any changes in source code.

There are a number of settings in the gateway, DB2/400, Oracle server, and the client that affect Globalization Support processing. In order for translations to take place correctly, character settings of these components must be compatible. Each character in one encoding scheme must have a matching character in another encoding scheme.

B.2 Gateway Configuration

After the gateway is installed, you must use the CHGORATUN command if you need to change language settings.

B.2.1 CHGORATUN Language Specification

The CHGORATUN command allows you to change the language parameter that defines the character set that is used for the gateway. The language parameter that is entered with this command specifies the conventions, such as language used for gateway messages, names of days and months, symbols for AD, BC, AM, and PM, and the default language sorting mechanism.

The syntax for specifying the language parameter is:

language[*_territory.character_set*]

where:

language is any valid language documented in Table B–2, " Supported Languages and Territories" on page B-3.

territory is optional and defaults to AMERICA. Valid values are documented in Table B–2, " Supported Languages and Territories" on page B-3.

character_set is optional and defaults to WE8EBCDIC37. Valid values are documented in Table B–1, " Character Sets Supported by Oracle Database Gateway for DB2/400" on page B-2.

The default setting is:

AMERICAN_AMERICA.WE8EBCDIC37

To change this setting, use the Gateway language field on the CHGORATUN main menu panel. The gateway must be shut down and restarted before the new parameter takes effect. For more information about the CHGORATUN command, refer to "CHGORATUN, Change Initialization Parameters" on page 6-6.

B.3 DB2/400 Coded Character Set Considerations

If the coded character set identifier (CCSID) of the AS/400 data field differs from 65535, then the Oracle language parameters must correspond to the CCSID of the AS/400 data field that is being accessed. For example, if the CCSID of the data field is 280 for Italy, then the Oracle character set must be set to I8EBCDIC280. The exception to this is for columns with a CCSID of 13488 (UCS-2). The data in these columns will always be converted to the character set that is determined by NLS_LANG. Contact your DBA or refer to the IBM manual for AS/400 Globalization Support for additional information about AS/400 CCSID codes.

B.3.1 Supported Character Sets

Oracle Database Gateway for DB2/400 supports the following languages and values for **character_set.**

Note that the character sets that are marked with an asterisk (*) are the Euro versions of the immediately preceding character set.

Table B–1   Character Sets Supported by Oracle Database Gateway for DB2/400

Description	Character Set	OS/400 CCSID
Austrian/German	D8EBCDIC273	273
Austrian/German (Euro)	D8EBCDIC1141*	1141
Traditional Chinese	ZHT16DBCS	937
Simplified Chinese	ZHS16DBCS	935
Danish/Norwegian	DK8EBCDIC277	277
Danish/Norwegian (Euro)	DK8E8CDIC1142*	1142
Eastern European	EE8EBCDIC870	870
Finnish/Swedish	S8EBCDIC278	278
Finnish/Swedish (Euro)	S8EBCDIC1143*	1143
French/France	F8EBCDIC297	297

Table B–1   (Cont.) Character Sets Supported by Oracle Database Gateway for DB2/400

Description	Character Set	OS/400 CCSID
French/France (Euro)	F8EBCDIC1147*	1147
German/Germany	D8EBCDIC273	273
German/Germany (Euro)	D8EBCDIC1141*	1141
Greek	EL8EBCDIC875	875
Hebrew	IW8EBCDIC424	424
Italian	I8EBCDIC280	280
Italian (Euro)	I8EBCDIC1144*	1144
Japanese	JA16DBCS	939, 5035
Japanese	JA16EBCDIC930	930, 5026
Korean	KO16DBCS	933
Spanish	WE8EBCDIC284	284
Thai	TH8TISEBCDIC	838
Turkish	TR8EBCDIC1026	1026
United States/Canada	WE8EBCDIC37	37
United States/Canada (Euro)	WE8EBCDIC1140*	1140
United States/Canada	WE8EBCDIC37C	
Western European	WE8EBCDIC500	500
Western European (Euro)	WE8EBCDIC1148*	1148
Western European	WE8EBCDIC500C	
Western European (Euro)	WE8EBCDIC1148C*	
United Kingdom	WE8EBCDIC285	285
United Kingdom (Euro)	WE8EBCDICIC1146*	1146

B.3.2 Supported Languages and Territories

Oracle Database Gateway for DB2/400 supports the following language and territory combinations:

Table B–2   Supported Languages and Territories

Language	Territory
American	America
Brazilian Portuguese	Brazil
Canadian French	Canada
Czech	Czech Republic
Danish	Denmark
Dutch	Netherlands
Finnish	Finland
French	France
German	Germany

Table B–2   (Cont.) Supported Languages and Territories

Language	Territory
Greek	Greece
Hungarian	Hungary
Icelandic	Iceland
Italian	Italy
Japanese	Japan
Mexican Spanish	Mexico
Norwegian	Norway
Polish	Poland
Portuguese	Portugal
Simplified Chinese	China
Slovak	Slovakia
Spanish	Spain
Swedish	Sweden
Traditional Chinese	Taiwan
Turkish	Turkey

B.4 Oracle Server and Client Configuration

A number of Globalization Support parameters control Globalization Support processing between the Oracle database and the client. You can set language-dependent behavior defaults for the server and set language-dependent behavior for the client to override these defaults. For a complete description of Globalization Support parameters, refer to the Globalization Support chapter in the *Oracle Database Globalization Support Guide*. These parameters do not affect gateway processing. However, you must ensure that the character set is compatible with the character set that you specify on the gateway and DB2/400. In other words, each character in one encoding scheme must have a matching character in another encoding scheme.

> **Caution:**   Character sets *must* be compatible for successful data transfer. Make sure that you know which character sets are being used and that they are compatible.

When you create your Oracle Database, the character set that is used to store data is specified by the CHARACTER SET parameter. After the database is created, the database character set cannot be changed unless you re-create the database.

The Oracle database default value for CHARACTER SET is platform dependent and version dependent. The value needs to be compatible with the CHARACTER SET that is used by the gateway. To check the character set of an existing database, issue the following command in SQL*Plus:

```
SELECT userenv('language') FROM DUAL
```

Normally, WE8ISO8859P1 is the default CHARACTER SET for Western European languages (including English) on non-EBCDIC platforms. Other languages may get a different default character set, for example, KO16KSC5601 for Korean.

B.5 Message Availability

Availability of the supported language message modules depends on which modules are installed in the Oracle product set that is running on the server. If you do not have message modules installed for a particular language set, then specifying that language with a language parameter results in no messages being displayed for that module in the requested language. Only a generalized (and rather uninformative) message will be provided.

B.6 DB2/400 GRAPHIC Support

When converting DB2/400 data types to Oracle data types, if support for DB2/400 GRAPHIC data types (GRAPHIC, VARGRAPHIC, or LONG VARGRAPHIC) is required, then special consideration must be given to the selection of the NLS_LANG character set. Refer to "DB2/400 GRAPHIC Support" on page 9-16 for more information.

B.7 Character Set Expansions

If you use a CREATE TABLE x AS SELECT command in Oracle and the source-table of the CREATE TABLE is a table in DB2/400, you may be surprised as to the width of the character-type columns of the new table in the Oracle DB. This column "expansion" is due to the fact that all codepoints in the CCSID of the DB2/400 column need to be representable in the Oracle column. Problems like this occur when the CHARACTER SET of the Oracle DB is something like UTF8. In this case, no matter what CCSID the DB2/400 column has, there is going to be a character set expansion. This is because there are at least 192 actual code-points in an SBCS code-page on the AS/400, but there are only 94 usable codepoints when that character is to be represented as a single-byte UTF8 character.

Here is an example:

The Japanese Yen character appears at codepoint 0xB2 in the code page represent by CCSID 37 (which corresponds to Oracle Characterset WE8EBCDIC37). But the codepoint for this character is 0xC2A5 in UTF8. The cent-sign appears at codepoint 0x4A in the code page represented by CCSID 37, but is represented by the codepoint 0xC2A2 in UTF8. So even though *your* data in DB2/400 does *not* contain either of a Yen or Cent-Sign symbol, the general case must hold that to represent either of these single byte SBCS characters in the Oracle DB, one needs two bytes. Now it actually turns out that there is one, and only one, character in the CCSID 37 codepage that cannot be represented by ANY sequence in the UTF8 characterset. That SBCS character is 0xFF and that is represented by the UTF8 sequence 0xEFBFBD. So, if one does a CREATE TABLE x AS SELECT, then one is *always* going to get at least a triple expansion of the columns. That is, a column represented by a CHAR(5) in DB2/400 and where that column has a default SBCS CCSID of perhaps 37, the corresponding column in the Oracle DB will be a CHAR(15).

C

Data Dictionary Views

This appendix includes the gateway data dictionary views that are accessible to all users of an Oracle database. Most views can be accessed by any user with SELECT privileges for DB2/400 catalog tables.

If a dictionary item is described with "N/A" in this appendix, then that means that the item is not available for the gateway. Such items default to zero for numeric columns, or to blanks or some fixed value for (VAR)CHAR columns.

This appendix contains the following sections:

The column names that are listed for the following views are the column names that the Oracle database presents to the client, and they may actually differ from the underlying views in DB2/400.

C.1 ALL_CATALOG

All tables, views, synonyms, and sequences that are accessible to the user.

Column Name	Description
OWNER	Schema of the object
TABLE_NAME	Name of the object
TABLE_TYPE	Type of object

C.2 ALL_COL_COMMENTS

Comments on columns of accessible tables and views.

Column Name	Description
OWNER	Schema of the object
TABLE_NAME	Object name
COLUMN_NAME	Column name
COMMENTS	Comments on the column

C.3 ALL_CONS_COLUMNS

Information about columns in constraint definitions for all users. Similar to [USER_CONS_COLUMNS](#) on page C-11.

Column Name	Description
OWNER	Schema of the object
CONSTRAINT_NAME	Name associated with the constraint definition
TABLE_NAME	Object name
COLUMN_NAME	Column name
POSITION	Original position of the column in the definition

C.4 ALL_CONSTRAINTS

Constraint definitions on accessible tables.

Column Name	Description
OWNER	Owner of the constraint definition
CONSTRAINT_NAME	Name associated with the constraint definition
CONSTRAINT_TYPE	Type of constraint definition
TABLE_NAME	Name associated with the table with the constraint definition
SEARCH_CONDITION	Text of the search condition for the table check
R_OWNER	Owner of the table used in the referential constraint
R_CONSTRAINT_NAME	Name of the unique constraint definition for the referenced table
DELETE_RULE	Delete rule for the referential constraint
STATUS	Status of the constraint
DEFERRABLE	N/A
DEFERRED	N/A
VALIDATED	N/A
GENERATED	N/A
BAD	N/A
RELY	N/A
LAST_CHANGE	N/A
INDEX_OWNER	N/A
INDEX_NAME	N/A
INVALID	N/A
VIEW_RELATED	N/A

C.5 ALL_IND_COLUMNS

Columns of the indexes on the accessible tables.

Column Name	Description
INDEX_OWNER	Owner of the index
INDEX_NAME	Name of the index
TABLE_OWNER	Table owner
TABLE_NAME	Table name
COLUMN_NAME	Column name
COLUMN_POSITION	Position of column within the index
COLUMN_LENGTH	Indexed length of column
CHAR_LENGTH	N/A
DESCEND	ASC for Ascending key, DESC for Descending key

C.6 ALL_INDEXES

Description of indexes on tables that are accessible to the user.

Column Name	Description
OWNER	Owner of the index
INDEX_NAME	Name of the index
INDEX_TYPE	Type of the Index
TABLE_OWNER	Owner of the indexed object
TABLE_NAME	Name of the indexed object
TABLE_TYPE	Type of indexed object
UNIQUENESS	Uniqueness status of the index
COMPRESSION	N/A
PREFIX_LENGTH	N/A
TABLESPACE_NAME	Name of the table space containing the index
INI_TRANS	N/A
MAX_TRANS	N/A
INITIAL_EXTENT	N/A
NEXT_EXTENT	N/A
MIN_EXTENTS	N/A
MAX_EXTENTS	N/A
PCT_INCREASE	N/A
PCT_THRESHOLD	N/A
INCLUDE_COLUMN	N/A
FREELISTS	N/A
FREELIST_GROUPS	N/A
PCT_FREE	N/A
LOGGING	N/A
BLEVEL	Depth of the index from its root block to its leaf blocks A depth of one indicates that the root block and the leaf block are the same.
LEAF_BLOCKS	Number of leaf blocks in the index
DISTINCT_KEYS	Number of distinct indexed values For indexes enforcing UNIQUE and PRIMARY KEY constraints, this value is the same as the number of rows in the table.
AVG_LEAF_BLOCKS_PER_KEY	N/A
AVG_DATA_BLOCKS_PER_KEY	N/A
CLUSTERING_FACTOR	N/A
STATUS	State of the index: VALID
NUM_ROWS	N/A

Column Name	Description
SAMPLE_SIZE	N/A
LAST_ANALYZED	N/A
DEGREE	N/A
INSTANCES	N/A
PARTITIONED	N/A
TEMPORARY	N/A
GENERATED	N/A
SECONDARY	N/A
BUFFER_POOL	N/A
USER_STATS	N/A
DURATION	N/A
PCT_DIRECT_ACCESS	N/A
ITYP_OWNER	N/A
ITYP_NAME	N/A
PARAMETERS	N/A
GLOBAL_STATS	N/A
DOMIDX_STATUS	N/A
DOMIDX_OPSTATUS	N/A
FUNCIDX_STATUS	N/A
JOIN_INDEX	N/A
IOT_REDUNDANT_PKEY	N/A
DROPPED	N/A

C.7 ALL_OBJECTS

Objects that are accessible to the user.

Column Name	Description
OWNER	Owner of the object
OBJECT_NAME	Name of object
SUBOBJECT_NAME	N/A
OBJECT_ID	Object number of the object
DATA_OBJECT_ID	N/A
OBJECT_TYPE	Type of object
CREATED	N/A
LAST_DDL_TIME	N/A
TIMESTAMP	N/A
STATUS	State of the object
TEMPORARY	N/A

Column Name	Description
GENERATED	N/A
SECONDARY	N/A

C.8 ALL_SYNONYMS

All synonyms that are accessible to the user.

Column Name	Description
OWNER	Owner of the synonym
SYNONYM_NAME	Name of the synonym
TABLE_OWNER	Owner of the object referenced by the synonym
TABLE_NAME	Name of the object referenced by the synonym
DB_LINK	N/A

C.9 ALL_TAB_COLUMNS

Columns of all tables, views, and clusters that are accessible to the user.

Column Name	Description
OWNER	Owner of the table or view
TABLE_NAME	Table or view name
COLUMN_NAME	Column name
DATA_TYPE	Data type of the column
DATA_TYPE_MOD	N/A
DATA_TYPE_OWNER	N/A
DATA_LENGTH	Maximum length of the column in bytes
DATA_PRECISION	N/A
DATA_SCALE	Digits to the right of decimal point in a number
NULLABLE	Asks if the column allow null values Value is n if there is a NOT NULL constraint on the column or if the column is part of a PRIMARY key.
COLUMN_ID	Sequence number of the column as created
DEFAULT_LENGTH	N/A
DATA_DEFAULT	N/A
NUM_DISTINCT	Number of distinct values in each column of the table
LOW_VALUE	Second lowest value for tables with more than three rows These statistics are expressed in hexadecimal notation for the internal representation of the first 32 bytes of the values.

Column Name	Description
HIGH_VALUE	Second highest value for tables with more than three rows
	These statistics are expressed in hexadecimal notation for the internal representation of the first 32 bytes of the values.
DENSITY	N/A
NUM_NULLS	N/A
NUM_BUCKETS	N/A
LAST_ANALYZED	N/A
SAMPLE_SIZE	N/A
CHARACTER_SET_NAME	N/A
CHAR_COL_DECL_LENGTH	N/A
GLOBAL_STATS	N/A
USER_STATS	N/A
AVG_COL_LEN	N/A
CHAR_LENGTH	N/A
CHAR_USED	N/A
V80_FMT_IMAGE	N/A
DATA_UPGRADED	N/A
HISTOGRAM	N/A

C.10 ALL_TAB_COMMENTS

Comments on tables and views that are accessible to the user.

Column Name	Description
OWNER	Owner of the object
TABLE_NAME	Name of the object
TABLE_TYPE	Type of object
COMMENTS	Comments on the object

C.11 ALL_TABLES

Description of tables that are accessible to the user.

Column Name	Description
OWNER	Owner of the table
TABLE_NAME	Name of the table
TABLESPACE_NAME	Name of the table space containing the table
CLUSTER_NAME	N/A
IOT_NAME	N/A
PCT_FREE	N/A

Column Name	Description
PCT_USED	N/A
INI_TRANS	N/A
MAX_TRANS	N/A
INITIAL_EXTENT	N/A
NEXT_EXTENT	N/A
MIN_EXTENTS	N/A
MAX_EXTENTS	N/A
PCT_INCREASE	N/A
FREELISTS	N/A
FREELIST_GROUPS	N/A
LOGGING	N/A
BACKED_UP	N/A
NUM_ROWS	Number of rows in the table
BLOCKS	N/A
EMPTY_BLOCKS	N/A
AVG_SPACE	N/A
CHAIN_CNT	N/A
AVG_ROW_LEN	Average length of a row in the table in bytes
AVG_SPACE_FREELIST_BLOCKS	N/A
NUM_FREELIST_BLOCKS	N/A
DEGREE	N/A
INSTANCES	N/A
CACHE	N/A
TABLE_LOCK	N/A
SAMPLE_SIZE	N/A
LAST_ANALYZED	N/A
PARTITIONED	N/A
IOT_TYPE	N/A
TEMPORARY	N/A
SECONDARY	N/A
NESTED	N/A
BUFFER_POOL	N/A
ROW_MOVEMENT	N/A
GLOBAL_STATS	N/A
USER_STATS	N/A
DURATION	N/A
SKIP_CORRUPT	N/A
MONITORING	N/A

Column Name	Description
CLUSTER_OWNER	N/A
DEPENDENCIES	N/A
COMPRESSION	N/A
DROPPED	N/A

C.12 ALL_USERS

Information about all users of the database.

Column Name	Description
USERNAME	Name of the user
USER_ID	N/A
CREATED	N/A

C.13 ALL_VIEWS

Text of views that are accessible to the user.

Column Name	Description
OWNER	Owner of the view
VIEW_NAME	Name of the view
TEXT_LENGTH	Length of the view text
TEXT	View text
	Only the first row of text is returned, even if multiple rows exist.
TYPE_TEXT_LENGTH	N/A
TYPE_TEXT	N/A
OID_TEXT_LENGTH	N/A
OID_TEXT	N/A
VIEW_TYPE_OWNER	N/A
VIEW_TYPE	N/A
SUPERVIEW_NAME	N/A

C.14 COLUMN_PRIVILEGES

Grants on columns for which the user is the grantor, grantee, or owner, or PUBLIC is the grantee.

Column Name	Description
GRANTEE	Name of the user to whom access is granted
OWNER	User name of the owner of the object
TABLE_NAME	Name of the object

Column Name	Description
COLUMN_NAME	Name of the column
GRANTOR	Name of the user who performed the grant
INSERT_PRIV	Permission to insert into the column
UPDATE_PRIV	Permission to update the column
REFERENCES_PRIV	Permission to reference the column
CREATED	Time stamp for the grant

C.15 DICTIONARY

Information about the tables in the Data Dictionary.

Column Name	Description
TABLE_NAME	Name of the table in the dictionary
COMMENTS	Comments about this dictionary entry

C.16 DUAL

Information about the all-purpose single-row DUAL table.

Column Name	Description
DUMMY	A dummy column (has value X)

C.17 OTGREGISTER

DB2/400 Special Registers

Column Name	Description
CURRENT_USER	Primary authorization id
CURRENT_DATE	Current date
CURRENT_TIME	Current time
CURRENT_TIMESTAMP	Current time stamp
CURRENT_TIMEZONE	Current time zone
CURRENT_SERVER	Host name of the current server
"current_user"	Primary authorization id
"current_date"	Current date
"current_time"	Current time
"current_timestamp"	Current time stamp
"current_timezone"	Current time zone
"current_server"	Host name of the current server

C.18 TABLE_PRIVILEGES

Grants on objects for which the user is the grantor, grantee, or owner, or PUBLIC is the grantee.

Column Name	Description
GRANTEE	Name of the user to whom access is granted
OWNER	Owner of the object
TABLE_NAME	Name of the object
GRANTOR	Name of the user who performed the grant
SELECT_PRIV	Permission to select from an object
INSERT_PRIV	Permission to insert into an object
DELETE_PRIV	Permission to delete from an object
UPDATE_PRIV	Permission to update an object
REFERENCES_PRIV	N/A
ALTER_PRIV	Permission to alter an object
INDEX_PRIV	Permission to create or drop an index on an object
CREATED	Time stamp for the grant

C.19 USER_CATALOG

Tables, views, synonyms, and sequences that are owned by the user.

Column Name	Description
TABLE_NAME	Name of the object
TABLE_TYPE	Type of object

C.20 USER_COL_COMMENTS

Comments on columns of user tables and views.

Column Name	Description
TABLE_NAME	Object name
COLUMN_NAME	Column name
COMMENTS	Comments on the column

C.21 USER_CONS_COLUMNS

Information about columns in constraint definitions that are owned by the user.

Column Name	Description
OWNER	Owner of the constraint definition
CONSTRAINT_NAME	Name associated with the constraint definition
TABLE_NAME	Name associated with the table with the constraint definition
COLUMN_NAME	Name associated with the column specified in the constraint definition
POSITION	Original position of the column in the definition

C.22 USER_CONSTRAINTS

Constraint definitions on user tables.

Column Name	Description
OWNER	Owner of the constraint definition
CONSTRAINT_NAME	Name associated with the constraint definition
CONSTRAINT_TYPE	Type of constraint definition
TABLE_NAME	Name associated with the table with the constraint definition
SEARCH_CONDITION	Text of the search condition for the table check
R_OWNER	Owner of the table used in the referential constraint
R_CONSTRAINT_NAME	Name of the unique constraint definition for the referenced table
DELETE_RULE	Delete rule for the referential constraint
STATUS	Status of the constraint
DEFERRABLE	N/A
DEFERRED	N/A
VALIDATED	N/A
GENERATED	N/A
BAD	N/A
RELY	N/A
LAST_CHANGE	N/A
INDEX_OWNER	N/A
INDEX_NAME	N/A
INVALID	N/A
VIEW_RELATED	N/A

C.23 USER_INDEXES

Description of the user's own indexes.

Column Name	Description
INDEX_NAME	Name of the index

Column Name	Description
INDEX_TYPE	Type of Index
TABLE_OWNER	Owner of the indexed object
TABLE_NAME	Name of the indexed object
TABLE_TYPE	Type of the indexed object
UNIQUENESS	Uniqueness status of the index
COMPRESSION	N/A
PREFIX_LENGTH	N/A
TABLESPACE_NAME	Name of the table space containing the index
INI_TRANS	N/A
MAX_TRANS	N/A
INITIAL_EXTENT	N/A
NEXT_EXTENT	N/A
MIN_EXTENTS	N/A
MAX_EXTENTS	N/A
PCT_INCREASE	N/A
PCT_THRESHOLD	N/A
INCLUDE_COLUMN	N/A
FREELISTS	N/A
FREELIST_GROUPS	N/A
PCT_FREE	N/A
LOGGING	N/A
BLEVEL	Depth of the index from its root block to its leaf blocks A depth of one indicates that the root block and the leaf block are the same.
LEAF_BLOCKS	Number of leaf blocks in the index
DISTINCT_KEYS	Number of distinct indexed values For indexes enforcing UNIQUE and PRIMARY KEY constraints, this value is the same as the number of rows in the table.
AVG_LEAF_BLOCKS_PER_KEY	N/A
AVG_DATA_BLOCKS_PER_KEY	N/A
CLUSTERING_FACTOR	N/A
STATUS	State of the indexes: VALID
NUM_ROWS	N/A
SAMPLE_SIZE	N/A
LAST_ANALYZED	N/A
DEGREE	N/A
INSTANCES	N/A
PARTITIONED	N/A

Column Name	Description
TEMPORARY	N/A
GENERATED	N/A
SECONDARY	N/A
BUFFER_POOL	N/A
USER_STATS	N/A
DURATION	N/A
PCT_DIRECT_ACCESS	N/A
ITYP_OWNER	N/A
ITYP_NAME	N/A
PARAMETERS	N/A
GLOBAL_STATS	N/A
DOMIDX_STATUS	N/A
DOMIDX_OPSTATUS	N/A
FUNCIDX_STATUS	N/A
JOIN_INDEX	N/A
IOT_REDUNDANT_PKEY	N/A
DROPPED	N/A

C.24 USER_OBJECTS

Objects that are owned by the user.

Column Name	Description
OBJECT_NAME	Name of object
SUBOBJECT_NAME	N/A
OBJECT_ID	Object number of the object
DATA_OBJECT_ID	N/A
OBJECT_TYPE	Type of object
CREATED	N/A
LAST_DDL_TIME	N/A
TIMESTAMP	N/A
STATUS	State of the object: VALID
TEMPORARY	N/A
GENERATED	N/A
SECONDARY	N/A

C.25 USER_SYNONYMS

The user's private synonyms.

Column Name	Description
SYNONYM_NAME	Name of the synonym
TABLE_OWNER	Owner of the object referenced by the synonym
TABLE_NAME	Name of the object referenced by the synonym
DB_LINK	N/A

C.26 USER_TAB_COLUMNS

Columns of user tables, views, and clusters.

Column Name	Description
TABLE_NAME	Table, view, or cluster name
COLUMN_NAME	Column name
DATA_TYPE	Data type of the column
DATA_TYPE_MOD	N/A
DATA_TYPE_OWNER	N/A
DATA_LENGTH	Maximum length of the column in bytes
DATA_PRECISION	N/A
DATA_SCALE	Digits to the right of decimal point in a number
NULLABLE	Asks if the column allow null values
	Value is n if there is a NOT NULL constraint on the column or if the column is part of a PRIMARY key.
COLUMN_ID	Sequence number of the column as created
DEFAULT_LENGTH	N/A
DATA_DEFAULT	N/A
NUM_DISTINCT	Number of distinct values in each column of the table
LOW_VALUE	Second lowest value for tables with more than three rows
	These statistics are expressed in hexadecimal notation for the internal representation of the first 32 bytes of the values.
HIGH_VALUE	Second highest value for tables with more than three rows
	These statistics are expressed in hexadecimal notation for the internal representation of the first 32 bytes of the values.
DENSITY	N/A
NUM_NULLS	N/A
NUM_BUCKETS	N/A
LAST_ANALYZED	N/A
SAMPLE_SIZE	N/A
CHARACTER_SET_NAME	N/A

Column Name	Description
CHAR_COL_DECL_LENGTH	N/A
GLOBAL_STATS	N/A
USER_STATS	N/A
AVG_COL_LEN	N/A
CHAR_LENGTH	N/A
CHAR_USED	N/A
V80_FMT_IMAGE	N/A
DATA_UPGRADED	N/A
HISTOGRAM	N/A

C.27 USER_TAB_COMMENTS

Comments on the tables and views that are owned by the user.

Column Name	Description
TABLE_NAME	Name of the object
TABLE_TYPE	Type of object
COMMENTS	Comments on the object

C.28 USER_TABLES

Description of the user's own tables.

Column Name	Description
TABLE_NAME	Name of the table
TABLESPACE_NAME	Name of the table space containing the table
CLUSTER_NAME	N/A
IOT_NAME	N/A
PCT_FREE	N/A
PCT_USED	N/A
INI_TRANS	N/A
MAX_TRANS	N/A
INITIAL_EXTENT	N/A
NEXT_EXTENT	N/A
MIN_EXTENTS	N/A
MAX_EXTENTS	N/A
PCT_INCREASE	N/A
FREELISTS	N/A
FREELIST_GROUPS	N/A
LOGGING	N/A

Column Name	Description
BACKED_UP	N/A
NUM_ROWS	Number of rows in the table
BLOCKS	N/A
EMPTY_BLOCKS	N/A
AVG_SPACE	N/A
CHAIN_CNT	N/A
AVG_ROW_LEN	Average length of a row in the table in bytes
AVG_SPACE_FREELIST_BLOCKS	N/A
NUM_FREELIST_BLOCKS	N/A
DEGREE	N/A
INSTANCES	N/A
CACHE	N/A
TABLE_LOCK	N/A
SAMPLE_SIZE	N/A
LAST_ANALYZED	N/A
PARTITIONED	N/A
IOT_TYPE	N/A
TEMPORARY	N/A
SECONDARY	N/A
NESTED	N/A
BUFFER_POOL	N/A
ROW_MOVEMENT	N/A
GLOBAL_STATS	N/A
USER_STATS	N/A
DURATION	N/A
SKIP_CORRUPT	N/A
MONITORING	N/A
CLUSTER_OWNER	N/A
DEPENDENCIES	N/A
COMPRESSION	N/A
DROPPED	N/A

C.29 USER_USERS

Information about the current user.

Column Name	Description
USERNAME	Name of the user
USER_ID	N/A

Column Name	Description
ACCOUNT_STATUS	N/A
LOCK_DATE	N/A
EXPIRY_DATE	N/A
DEFAULT_TABLESPACE	N/A
TEMPORARY_TABLESPACE	N/A
CREATED	N/A
INITIAL_RSRC_CONSUMER_GROUP	N/A
EXTERNAL_NAME	N/A

C.30 USER_VIEWS

Text of views that are owned by the user.

Column Name	Description
VIEW_NAME	Name of the view
TEXT_LENGTH	Length of the view text
TEXT	First line of view text
TYPE_TEXT_LENGTH	N/A
TYPE_TEXT	N/A
OID_TEXT_LENGTH	N/A
OID_TEXT	N/A
VIEW_TYPE_OWNER	N/A
VIEW_TYPE	N/A
SUPERVIEW_NAME	N/A

Index

D