

# **Oracle® Database Gateway**

for Sybase User's Guide

11g Release 1 (11.1)

**B31048-01**

May 2007

Beta Draft

Oracle Database Gateway for Sybase User's Guide, 11g Release 1 (11.1)

B31048-01

Copyright © 2002, 2007, Oracle. All rights reserved.

Primary Author: Amitai Sela

Contributing Author: Laurel Hale, Cynthia Kibbe, Kishan Peyetti, Maitreyee Chaliha

Contributor: Orit Curiel, Jacco Draaijer, Vira Goorah

The Programs (which include both the software and documentation) contain proprietary information; they are provided under a license agreement containing restrictions on use and disclosure and are also protected by copyright, patent, and other intellectual and industrial property laws. Reverse engineering, disassembly, or decompilation of the Programs, except to the extent required to obtain interoperability with other independently created software or as specified by law, is prohibited.

The information contained in this document is subject to change without notice. If you find any problems in the documentation, please report them to us in writing. This document is not warranted to be error-free. Except as may be expressly permitted in your license agreement for these Programs, no part of these Programs may be reproduced or transmitted in any form or by any means, electronic or mechanical, for any purpose.

If the Programs are delivered to the United States Government or anyone licensing or using the Programs on behalf of the United States Government, the following notice is applicable:

U.S. GOVERNMENT RIGHTS Programs, software, databases, and related documentation and technical data delivered to U.S. Government customers are "commercial computer software" or "commercial technical data" pursuant to the applicable Federal Acquisition Regulation and agency-specific supplemental regulations. As such, use, duplication, disclosure, modification, and adaptation of the Programs, including documentation and technical data, shall be subject to the licensing restrictions set forth in the applicable Oracle license agreement, and, to the extent applicable, the additional rights set forth in FAR 52.227-19, Commercial Computer Software--Restricted Rights (June 1987). Oracle USA, Inc., 500 Oracle Parkway, Redwood City, CA 94065.

The Programs are not intended for use in any nuclear, aviation, mass transit, medical, or other inherently dangerous applications. It shall be the licensee's responsibility to take all appropriate fail-safe, backup, redundancy and other measures to ensure the safe use of such applications if the Programs are used for such purposes, and we disclaim liability for any damages caused by such use of the Programs.

Oracle, JD Edwards, PeopleSoft, and Siebel are registered trademarks of Oracle Corporation and/or its affiliates. Other names may be trademarks of their respective owners.

The Programs may provide links to Web sites and access to content, products, and services from third parties. Oracle is not responsible for the availability of, or any content provided on, third-party Web sites. You bear all risks associated with the use of such content. If you choose to purchase any products or services from a third party, the relationship is directly between you and the third party. Oracle is not responsible for: (a) the quality of third-party products or services; or (b) fulfilling any of the terms of the agreement with the third party, including delivery of products or services and warranty obligations related to purchased products or services. Oracle is not responsible for any loss or damage of any sort that you may incur from dealing with any third party.

Alpha and Beta Draft documentation are considered to be in prerelease status. This documentation is intended for demonstration and preliminary use only. We expect that you may encounter some errors, ranging from typographical errors to data inaccuracies. This documentation is subject to change without notice, and it may not be specific to the hardware on which you are using the software. Please be advised that prerelease documentation is not warranted in any manner, for any purpose, and we will not be responsible for any loss, costs, or damages incurred due to the use of this documentation.

---

---

# Contents

<b>Preface</b> .....	ix
Audience.....	ix
Documentation Accessibility .....	ix
Related Documents .....	x
Conventions .....	x
 <b>1 Introduction</b>	
Overview.....	1-1
Heterogeneous Services Technology .....	1-2
Oracle Database Gateways.....	1-2
 <b>2 Sybase Gateway Features and Restrictions</b>	
Using the Pass-Through Feature .....	2-1
Executing Stored Procedures and Functions.....	2-2
Return Values and Stored Procedures .....	2-2
Result Sets and Stored Procedures .....	2-3
OCI Program Fetching from Result Sets in Sequential Mode .....	2-4
PL/SQL Program Fetching from Result Sets in Sequential Mode.....	2-5
Database Compatibility Issues for Sybase.....	2-6
Chained Mode .....	2-6
Column Definitions.....	2-6
Naming Rules .....	2-6
Rules for Naming Objects .....	2-7
Case Sensitivity .....	2-7
Data Types.....	2-7
Binary Literal Notation .....	2-8
Data Type Conversion.....	2-8
Queries.....	2-8
Row Selection .....	2-8
Empty Strings .....	2-8
Empty Bind Variables.....	2-9
Locking .....	2-9
Known Restrictions .....	2-9
Transactional Integrity .....	2-10

Transaction Capability .....	2-10
COMMIT or ROLLBACK in PL/SQL Cursor Loops Closes Open Cursors .....	2-10
Stored Procedures .....	2-10
Pass-Through Feature.....	2-11
Sybase NCHAR and NVARCHAR Data Types .....	2-11
SQL Syntax .....	2-11
WHERE CURRENT OF Clause.....	2-11
CONNECT BY Clause .....	2-11
ROWID .....	2-11
Subqueries in INSERT Statement .....	2-12
EXPLAIN PLAN Statement.....	2-12
Callback Support.....	2-12
SQL*Plus COPY Command with Lowercase Table Names.....	2-12
Database Links.....	2-12
<b>Known Problems .....</b>	<b>2-13</b>
FLOAT Data Type .....	2-13
VARBINARY Data Type .....	2-13
Encrypted Format Login .....	2-13
Date Arithmetic .....	2-13
Sybase IMAGE and TEXT Data Type.....	2-13
String Functions.....	2-14
Schema Names and PL/SQL.....	2-14
Data Dictionary Views and PL/SQL.....	2-14
Stored Procedures .....	2-14

### 3 Case Studies

<b>Case Descriptions .....</b>	<b>3-1</b>
<b>Distrubution Media Contents.....</b>	<b>3-2</b>
<b>Demonstration Files.....</b>	<b>3-2</b>
<b>Demonstration Requirements.....</b>	<b>3-2</b>
<b>Creating Demonstration Tables.....</b>	<b>3-2</b>
Demonstration Table Definitions.....	3-3
Demonstration Table Contents .....	3-4
<b>Case 1: Simple Queries .....</b>	<b>3-5</b>
<b>Case 2: A More Complex Query .....</b>	<b>3-5</b>
<b>Case 3: Joining Sybase Tables .....</b>	<b>3-5</b>
<b>Case 4: Write Capabilities .....</b>	<b>3-5</b>
DELETE Statement.....	3-5
UPDATE Statement .....	3-5
INSERT Statement.....	3-5
<b>Case 5: Data Dictionary Query .....</b>	<b>3-6</b>
<b>Case 6: The Pass-Through Feature .....</b>	<b>3-6</b>
UPDATE Statement .....	3-6
SELECT Statement .....	3-6
<b>Case 7: Executing Stored Procedures .....</b>	<b>3-6</b>

## **A Data Type Conversion**

Data Type Conversion .....	A-1
----------------------------	-----

## **B Supported SQL Syntax and Functions**

Supported SQL Statements .....	B-1
DELETE .....	B-1
INSERT .....	B-1
SELECT .....	B-1
UPDATE .....	B-2
Oracle Functions .....	B-2
Functions Not Supported by Sybase .....	B-2
Functions Supported by Sybase .....	B-2
Arithmetic Operators .....	B-2
Comparison Operators .....	B-3
Group Functions .....	B-3
String Functions .....	B-3
Other Functions .....	B-3

## **C Data Dictionary**

Data Dictionary Support .....	C-1
Sybase System Tables .....	C-1
Accessing the Gateway Data Dictionary .....	C-1
Direct Queries to Sybase Tables .....	C-2
Supported Views and Tables .....	C-2
Data Dictionary Mapping .....	C-3
Default Column Values .....	C-4
Gateway Data Dictionary Descriptions .....	C-4

## **D Initialization Parameters**

Initialization Parameter File Syntax .....	D-1
Oracle Database Gateway for Sybase Initialization Parameters .....	D-2
Initialization Parameter Description .....	D-3
HS_CALL_NAME .....	D-3
HS_DB_DOMAIN .....	D-3
HS_DB_INTERNAL_NAME .....	D-4
HS_DB_NAME .....	D-4
HS_DESCRIBE_CACHE_HWM .....	D-4
HS_LANGUAGE .....	D-5
Character Sets .....	D-5
Language .....	D-5
Territory .....	D-5
HS_LONG_PIECE_TRANSFER_SIZE .....	D-6
HS_OPEN_CURSORS .....	D-6
HS_RPC_FETCH_REBLOCKING .....	D-6
HS_RPC_FETCH_SIZE .....	D-7

HS_TIME_ZONE .....	D-7
HS_TRANSACTION_MODEL .....	D-7
IFILE .....	D-8
HS_FDS_CONNECT_INFO .....	D-8
HS_FDS_DEFAULT_OWNER .....	D-8
HS_FDS_PROC_IS_FUNC.....	D-9
HS_FDS_RECOVERY_ACCOUNT .....	D-9
HS_FDS_RECOVERY_PWD.....	D-9
HS_FDS_RESULTSET_SUPPORT .....	D-10
HS_FDS_TRACE_LEVEL.....	D-10
HS_FDS_TRANSACTION_LOG .....	D-10
HS_FDS_FETCH_ROWS.....	D-11

## Index

## List of Tables

A-1	<i>Data Type Conversions</i> .....	A-1
C-1	Oracle Data Dictionary View Names and Sybase Equivalents .....	C-3
C-2	ALL_CATALOG .....	C-4
C-3	ALL_COL_COMMENTS .....	C-4
C-4	ALL_CONS_COLUMNS .....	C-5
C-5	ALL_CONSTRAINTS.....	C-5
C-6	ALL_IND_COLUMNS.....	C-5
C-7	ALL_INDEXES .....	C-6
C-8	ALL_OBJECTS.....	C-7
C-9	ALL_TAB_COLUMNS.....	C-7
C-10	ALL_TAB_COMMENTS.....	C-8
C-11	ALL_TABLES .....	C-8
C-12	ALL_USERS .....	C-10
C-13	ALL_VIEWS.....	C-10
C-14	DBA_CATALOG.....	C-10
C-15	DBA_COL_COMMENTS.....	C-10
C-16	DBA_OBJECTS .....	C-11
C-17	DBA_TAB_COLUMNS.....	C-11
C-18	DBA_TAB_COMMENTS.....	C-12
C-19	DBA_TABLES.....	C-12
C-20	DICT_COLUMNS.....	C-13
C-21	DICTIONARY .....	C-13
C-22	DUAL.....	C-14
C-23	TABLE_PRIVILEGES .....	C-14
C-24	USER_CATALOG.....	C-14
C-25	USER_COL_COMMENTS .....	C-14
C-26	USER_CONS_COLUMNS .....	C-14
C-27	USER_CONSTRAINTS .....	C-15
C-28	USER_IND_COLUMNS.....	C-15
C-29	USER_INDEXES.....	C-15
C-30	USER_OBJECTS .....	C-17
C-31	USER_TAB_COLUMNS .....	C-17
C-32	USER_TAB_COMMENTS .....	C-18
C-33	USER_TABLES .....	C-18
C-34	USER_USERS.....	C-19
C-35	USER_VIEWS .....	C-20





---

---

# Preface

This manual describes the Oracle Database Gateway for Sybase, which enables Oracle client applications to access Sybase data through Structured Query Language (SQL). The gateway, with the Oracle database, creates the appearance that all data resides on a local Oracle database, even though the data can be widely distributed.

This preface covers the following topics:

- [Audience](#)
- [Documentation Accessibility](#)
- [Related Documents](#)
- [Conventions](#)

## Audience

This manual is intended for Oracle database administrators who perform the following tasks:

- Installing and configuring the Oracle Database Gateway for Sybase
- Diagnosing gateway errors
- Using the gateway to access Sybase data

---

---

**Note:** You should understand the fundamentals of Oracle Database Gateways and the UNIX based platform before using this guide to install or administer 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.

## **Related Documents**

For more information, see these Oracle resources:

- *Oracle Database New Features Guide*
- *Oracle Call Interface Programmer's Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database Advanced Application Developer's Guide*
- *Oracle Database Concepts*
- *Oracle Database Performance Tuning Guide*
- *Oracle Database Error Messages*
- *Oracle Database Globalization Support Guide*
- *Oracle Database Reference*
- *Oracle Database SQL Language Reference*
- *Oracle Database Net Services Administrator's Guide*
- *SQL\*Plus User's Guide and Reference*
- *Oracle Database Heterogeneous Connectivity Administrator's Guide*
- *Oracle Database Security Guide*

Many of the examples in this book use the sample schemas of the seed database, which is installed by default when you install Oracle. Refer to *Oracle Database Sample Schemas* for information on how these schemas were created and how you can use them yourself.

## **Conventions**

This section describes the conventions used in the text and code examples of this documentation set. It describes:

- [Conventions in Text](#)
- [Conventions in Code Examples](#)

## Conventions in Text

We use various conventions in text to help you more quickly identify special terms. The following table describes those conventions and provides examples of their use.

Convention	Meaning	Example
<b>Bold</b>	Bold typeface indicates terms that are defined in the text or terms that appear in a glossary, or both.	When you specify this clause, you create an <b>index-organized table</b> .
<i>Italics</i>	Italic typeface indicates book titles or emphasis.	<i>Oracle Database Concepts</i> Ensure that the recovery catalog and target database do <i>not</i> reside on the same disk.
UPPERCASE monospace (fixed-width) font	Uppercase monospace typeface indicates elements supplied by the system. Such elements include parameters, privileges, data types, RMAN keywords, SQL keywords, SQL*Plus or utility commands, packages and methods, as well as system-supplied column names, database objects and structures, usernames, and roles.	You can specify this clause only for a NUMBER column. You can back up the database by using the BACKUP command. Query the TABLE_NAME column in the USER_TABLES data dictionary view. Use the DBMS_STATS.GENERATE_STATS procedure.
lowercase monospace (fixed-width) font	Lowercase monospace typeface indicates executables, filenames, directory names, and sample user-supplied elements. Such elements include computer and database names, net service names, and connect identifiers, as well as user-supplied database objects and structures, column names, packages and classes, usernames and roles, program units, and parameter values.  <b>Note:</b> Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	Enter sqlplus to open SQL*Plus. The password is specified in the orapwd file. Back up the datafiles and control files in the /disk1/oracle/dbs directory. The department_id, department_name, and location_id columns are in the hr.departments table. Set the QUERY_REWRITE_ENABLED initialization parameter to true. Connect as oe user. The JRepUtil class implements these methods.
<i>lowercase italic monospace (fixed-width) font</i>	Lowercase italic monospace font represents placeholders or variables.	You can specify the <i>parallel_clause</i> . Run <i>Uold_release</i> .SQL where <i>old_release</i> refers to the release you installed prior to upgrading.

## Conventions in Code Examples

Code examples illustrate SQL, PL/SQL, SQL\*Plus, or other command-line statements. They are displayed in a monospace (fixed-width) font and separated from normal text as shown in this example:

```
SELECT username FROM dba_users WHERE username = 'MIGRATE';
```

The following table describes typographic conventions used in code examples and provides examples of their use.

Convention	Meaning	Example
[ ]	Brackets enclose one or more optional items. Do not enter the brackets.	DECIMAL ( <i>digits</i> [ , <i>precision</i> ])
{ }	Braces enclose two or more items, one of which is required. Do not enter the braces.	{ENABLE   DISABLE}

Convention	Meaning	Example
	A vertical bar represents a choice of two or more options within brackets or braces. Enter one of the options. Do not enter the vertical bar.	{ENABLE   DISABLE} [COMPRESS   NOCOMPRESS]
...	Horizontal ellipsis points indicate either: <ul style="list-style-type: none"> <li>■ That we have omitted parts of the code that are not directly related to the example</li> <li>■ That you can repeat a portion of the code</li> </ul>	CREATE TABLE ... AS subquery;  SELECT col1, col2, ... , coln FROM employees;
. . . .	Vertical ellipsis points indicate that we have omitted several lines of code not directly related to the example.	SQL> SELECT NAME FROM V\$DATAFILE; NAME ----- /fs1/dbs/tbs_01.dbf /fs1/dbs/tbs_02.dbf . . . /fs1/dbs/tbs_09.dbf 9 rows selected.
Other notation	You must enter symbols other than brackets, braces, vertical bars, and ellipsis points as shown.	acctbal NUMBER(11,2); acct      CONSTANT NUMBER(4) := 3;
<i>Italics</i>	Italicized text indicates placeholders or variables for which you must supply particular values.	CONNECT SYSTEM/ <i>system_password</i> DB_NAME = <i>database_name</i>
UPPERCASE	Uppercase typeface indicates elements supplied by the system. We show these terms in uppercase in order to distinguish them from terms you define. Unless terms appear in brackets, enter them in the order and with the spelling shown. However, because these terms are not case sensitive, you can enter them in lowercase.	SELECT last_name, employee_id FROM employees; SELECT * FROM USER_TABLES; DROP TABLE hr.employees;
lowercase	Lowercase typeface indicates programmatic elements that you supply. For example, lowercase indicates names of tables, columns, or files.  <b>Note:</b> Some programmatic elements use a mixture of UPPERCASE and lowercase. Enter these elements as shown.	SELECT last_name, employee_id FROM employees; sqlplus hr/hr CREATE USER mjones IDENTIFIED BY ty3MU9;

---

# Introduction

This chapter introduces the challenge faced by organizations when running several different database systems. It briefly covers Heterogeneous Services, the technology that the Oracle Database Gateway for Sybase is based on.

To get a good understanding of generic gateway technology, Heterogeneous Services, Generic Connectivity, and how Oracle Database Gateways fit in the picture, reading the *Oracle Database Heterogeneous Connectivity Administrator's Guide* first is highly recommended.

This chapter contains the following sections:

- [Overview](#)
- [Heterogeneous Services Technology](#)
- [Oracle Database Gateways](#)

## Overview

Heterogeneous data access is a problem that affects a lot of companies. A lot of companies run several different database systems. Each of these systems stores data and has a set of applications that run against it. Consolidation of this data in one database system is often hard - in large part because many of the applications that run against one database may not have an equivalent that runs against another. Until such time as migration to one consolidated database system is made feasible, it is necessary for the various heterogeneous database systems to interoperate.

Oracle Database Gateways provide the ability to transparently access data residing in a non-Oracle system from an Oracle environment. This transparency eliminates the need for application developers to customize their applications to access data from different non-Oracle systems, thus decreasing development efforts and increasing the mobility of the application. Applications can be developed using a consistent Oracle interface for both Oracle and Sybase.

Gateway technology is composed of two parts: a component that has the generic technology to connect to a non-Oracle system, which is common to all the non-Oracle systems, called Heterogeneous Services, and a component that is specific to the non-Oracle system that the gateway connects to. Heterogeneous Services, in conjunction with the Oracle Database Gateway agent, enables transparent access to non-Oracle systems from an Oracle environment.

## Heterogeneous Services Technology

Heterogeneous Services provides the generic technology for connecting to non-Oracle systems. As an integrated component of the database, Heterogeneous Services can exploit features of the database, such as the powerful SQL parsing and distributed optimization capabilities.

Heterogeneous Services extend the Oracle SQL engine to recognize the SQL and procedural capabilities of the remote non-Oracle system and the mappings required to obtain necessary data dictionary information. Heterogeneous Services provides two types of translations: the ability to translate Oracle SQL into the proper dialect of the non-Oracle system as well as data dictionary translations which displays the metadata of the non-Oracle system in the local format. For situations where no translations are available, native SQL can be issued to the non-Oracle system using the pass-through feature of Heterogeneous Services.

Heterogeneous Services also maintains the transaction coordination between Oracle and the remote non-Oracle system, such as providing the two-phase commit protocol to ensure distributed transaction integrity, even for non-Oracle systems that do not natively support two-phase commit.

**See Also:** *Oracle Database Heterogeneous Connectivity Administrator's Guide* for more information about Heterogeneous Services.

## Oracle Database Gateways

The capabilities, SQL mappings, data type conversions, and interface to the remote non-Oracle system are contained in the gateway. The gateway interacts with Heterogeneous Services to provide the transparent connectivity between Oracle and non-Oracle systems.

The gateway must be installed on a machine running either the Sybase database or the Sybase client. This machine can be the same machine as the Oracle database or on the same machine as the Sybase database or on a third machine as a standalone. Each configuration has its advantages and disadvantages. The issues to consider when determining where to install the gateway are network traffic, operating system platform availability, hardware resources and storage.

---

## Sybase Gateway Features and Restrictions

After the gateway is installed and configured, you can use the gateway to access Sybase data, pass Sybase commands from applications to the Sybase database, perform distributed queries, and copy data.

This chapter contains the following sections:

- [Using the Pass-Through Feature](#)
- [Executing Stored Procedures and Functions](#)
- [Database Compatibility Issues for Sybase](#)
- [Known Restrictions](#)
- [Known Problems](#)

### Using the Pass-Through Feature

The gateway can pass Sybase commands or statements from the application to the Sybase database using the DBMS\_HS\_PASSTHROUGH package.

Use the DBMS\_HS\_PASSTHROUGH package in a PL/SQL block to specify the statement to be passed to the Sybase database, as follows:

```
DECLARE
    num_rows INTEGER;
BEGIN
    num_rows := DBMS_HS_PASSTHROUGH.EXECUTE_IMMEDIATE@SYBS('command');
END;
/
```

Where *command* cannot be one of the following:

- BEGIN TRANSACTION
- COMMIT
- ROLLBACK
- SAVE
- SHUTDOWN
- Sybase tool commands

The DBMS\_HS\_PASSTHROUGH package supports passing bind values and executing SELECT statements.

---

---

**Note:** TRUNCATE cannot be used in a pass-through statement.

---

---

**See Also:** *Oracle Database PL/SQL Packages and Types Reference* and Chapter 3, *Features of Oracle Database Gateways and Generic Connectivity*, of *Oracle Database Heterogeneous Connectivity Administrator's Guide* for more information about the DBMS\_HS\_PASSTHROUGH package.

## Executing Stored Procedures and Functions

Using the procedural feature, the gateway can execute stored procedures that are defined in the Sybase database. It is not necessary to relink the gateway or define the procedure to the gateway, but the procedure's access privileges must permit access by the gateway.

**See Also:** *Oracle Database Heterogeneous Connectivity Administrator's Guide* for more information about executing stored procedures.

Standard PL/SQL statements are used to execute a stored procedure.

## Return Values and Stored Procedures

By default, all stored procedures and functions do not return a return value to the user. To enable return values, set the HS\_FDS\_PROC\_IS\_FUNC parameter in the initialization parameter file.

**See Also:** [Appendix D, "Initialization Parameters"](#) for information about both editing the initialization parameter file and the HS\_FDS\_PROC\_IS\_FUNC parameter.

---

---

**Note:** If you set the HS\_FDS\_PROC\_IS\_FUNC gateway initialization parameter, you must change the syntax of the procedure execute statement for all existing stored procedures.

---

---

In the following example, the employee name JOHN SMYTHE is passed to the Sybase stored procedure REVISE\_SALARY. The stored procedure retrieves the salary value from the Sybase database to calculate a new yearly salary for JOHN SMYTHE. The revised salary returned in RESULT is used to update EMP in a table of an Oracle database:

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

The procedural feature automatically converts non-Oracle data types to and from PL/SQL data types.



## Result Sets and Stored Procedures

The Oracle Database Gateway for Sybase provides support for stored procedures which return result sets.

By default, all stored procedures and functions do not return a result set to the user. To enable result sets, set the `HS_FDS_RESULTSET_SUPPORT` parameter in the initialization parameter file.

**See Also:** [Appendix D, "Initialization Parameters"](#) for information about both editing the initialization parameter file and the `HS_FDS_RESULTSET_SUPPORT` parameter. For further information about Oracle support for result sets in non-Oracle databases see *Oracle Database Heterogeneous Connectivity Administrator's Guide*.

---

---

**Note:** If you set the `HS_FDS_RESULTSET_SUPPORT` gateway initialization parameter, you must change the syntax of the procedure execute statement for all existing stored procedures or errors will occur.

---

---

When accessing stored procedures with result sets through the Oracle Database Gateway for Sybase, you must work in the sequential mode of Heterogeneous Services.

The Oracle Database Gateway for Sybase returns the following information to Heterogeneous Services during procedure description:

- All the input arguments of the remote stored procedure
- None of the output arguments
- One out argument of type ref cursor (corresponding to the first result set returned by the stored procedure)

Client programs have to use the virtual package function `dbms_hs_result_set.get_next_result_set` to get the ref cursor for subsequent result sets. The last result set returned is the out argument from the procedure.

The limitations of accessing result sets are the following:

- Result sets returned by a remote stored procedure have to be retrieved in the order in which they were placed on the wire
- On execution of a stored procedure, all result sets returned by a previously executed stored procedure will be closed (regardless of whether the data has been completely

In the following example, the Sybase stored procedure is executed to fetch the contents of the `emp` and `dept` tables from Sybase:

```
create procedure REFCURPROC (@arg1 varchar(255), @arg2 varchar(255) output)
as
select @arg2 = @arg1
select * from EMP
select * from DEPT
go
```

This stored procedure assigns the input parameter `arg1` to the output parameter `arg2`, opens the query `SELECT * FROM EMP` in ref cursor `rc1`, and opens the query `SELECT * FROM DEPT` in ref cursor `rc2`.

---

---

**Note:** Chained mode must be set before creating the stored procedure. Issue the following command in Sybase: `set chained on`

---

---

## OCI Program Fetching from Result Sets in Sequential Mode

The following example shows OCI program fetching from result sets in sequential mode:

```
OCIEnv *ENVH;
OCISvcCtx *SVCH;
OCIStmt *STMH;
OCIError *ERRH;
OCIBind *BNDH[3];
OraText arg1[20];
OraText arg2[255];
OCIResult *rset;
OCIStmt *rstmt;
ub2 rcode[3];
ub2 rlens[3];
sb2 inds[3];
OraText *stmt = (OraText *) "begin refcurproc@SYBS(:1,:2,:3); end;";
OraText *n_rs_stm = (OraText *)
    "begin :ret := DBMS_HS_RESULT_SET.GET_NEXT_RESULT_SET@SYBS; end;";

/* Prepare procedure call statement */

/* Handle Initialization code skipped */
OCIStmtPrepare(STMH, ERRH, stmt, strlen(stmt), OCI_NTV_SYNTAX, OCI_DEFAULT);

/* Bind procedure arguments */
inds[0] = 0;
strcpy((char *) arg1, "Hello World");
rlens[0] = strlen(arg1);
OCIBindByPos(STMH, &BNDH[0], ERRH, 1, (dvoid *) arg1, 20, SQLT_CHR,
    (dvoid *) &(inds[0]), &(rlens[0]), &(rcode[0]), 0, (ub4 *) 0,
    OCI_DEFAULT);
inds[1] = -1;
OCIBindByPos(STMH, &BNDH[1], ERRH, 1, (dvoid *) arg2, 20, SQLT_CHR,
    (dvoid *) &(inds[1]), &(rlens[1]), &(rcode[1]), 0, (ub4 *) 0,
    OCI_DEFAULT);

inds[2] = 0;
rlens[2] = 0;
OCIDescriptorAlloc(ENVH, (dvoid **) &rset, OCI_DTYPE_RSET, 0, (dvoid **) 0);
OCIBindByPos(STMH, &BNDH[2], ERRH, 2, (dvoid *) rset, 0, SQLT_RSET,
    (dvoid *) &(inds[2]), &(rlens[2]), &(rcode[2]),
    0, (ub4 *) 0, OCI_DEFAULT);

/* Execute procedure */
OCIStmtExecute(SVCH, STMH, ERRH, 1, 0, (CONST OCISnapshot *) 0,
    (OCISnapshot *) 0, OCI_DEFAULT);

/* Convert result set to statement handle */
OCIResultSetToStmt(rset, ERRH);
rstmt = (OCIStmt *) rset;

/* After this the user can fetch from rstmt */
/* Issue get_next_result_set call to get handle to next_result set */
```

```

/* Prepare Get next result set procedure call */

OCISstmtPrepare(STMH, ERRH, n_rs_stm, strlen(n_rs_stm), OCI_NTV_SYNTAX,
                OCI_DEFAULT);

/* Bind return value */
OCIBindByPos(STMH, &BNDH[1], ERRH, 1, (dvoid *) rset, 0, SQLT_RSET,
              (dvoid *) &(inds[1]), &(rlens[1]), &(rcode[1]),
              0, (ub4 *) 0, OCI_DEFAULT);

/* Execute statement to get next result set*/
OCISstmtExecute(SVCH, STMH, ERRH, 1, 0, (CONST OCISnapshot *) 0,
                (OCISnapshot *) 0, OCI_DEFAULT);

/* Convert next result set to statement handle */
OCIResultSetToStmnt(rset, ERRH);
rstmt = (OCISstmt *) rset;

/* Now rstmt will point to the second result set returned by the
remote stored procedure */

/* Repeat execution of get_next_result_set to get the output arguments */

```

### PL/SQL Program Fetching from Result Sets in Sequential Mode

Assume that the table `loc_emp` is a local table exactly like the Sybase `emp` table. The same assumption applies for `loc_dept`. `outargs` is a table with columns corresponding to the out arguments of the Sybase stored procedure.

```

declare
    rc1 rcpackage.rctype;
    rec1 loc_emp%rowtype;
    rc2 rcpackage.rctype;
    rec2 loc_dept%rowtype;
    rc3 rcpackage.rctype;
    rec3 outargs%rowtype;
    out_arg varchar2(255);

begin

    -- Execute procedure
    out_arg := null;
    refcurproc@SYBS('Hello World', out_arg, rc1);

    -- Fetch 20 rows from the remote emp table and insert them into loc_emp
    for i in 1 .. 20 loop
        fetch rc1 into rec1;
        insert into loc_emp (rec1.empno, rec1.ename, rec1.job,
                           rec1.mgr, rec1.hiredate, rec1.sal, rec1.comm, rec1.deptno);
    end loop;

    -- Close ref cursor
    close rc1;

    -- Get the next result set returned by the stored procedure
    rc2 := dbms_hs_result_set.get_next_result_set@SYBS;

    -- Fetch 5 rows from the remote dept table and insert them into loc_dept
    for i in 1 .. 5 loop
        fetch rc2 into rec2;
    end loop;

```

```
        insert into loc_dept values (rec2.deptno, rec2.dname, rec2.loc);
    end loop;

    --Close ref cursor
    close rc2;

    -- Get the output arguments from the remote stored procedure
    -- Since we are in sequential mode, they will be returned in the
    -- form of a result set
    rc3 := dbms_hs_result_set.get_next_result_set@SYBS;

    -- Fetch them and insert them into the outarguments table
    fetch rc3 into rec3;
    insert into outargs (rec3.outarg, rec3.retval);

    -- Close ref cursor
    close rc3;

end;
/
```

## Database Compatibility Issues for Sybase

Sybase and Oracle databases function differently in some areas, causing compatibility problems. The following compatibility issues are described in this section:

- [Chained Mode](#)
- [Column Definitions](#)
- [Naming Rules](#)
- [Data Types](#)
- [Queries](#)
- [Locking](#)

### Chained Mode

The gateway supports the ANSI-standard chained mode. Sybase stored procedures must be written for this mode. Running in chained mode allows the gateway to extend the Oracle two-phase commit protection to transactions updating Oracle and Sybase databases.

### Column Definitions

By default, a Sybase table column cannot contain null values unless `NULL` is specified in the column definition. In compliance with the ANSI standard, the Sybase database option "allow nulls by default" can be set to true to change the default column definition to `NULL`.

For an Oracle table, null values are allowed in a column unless `NOT NULL` is specified in the column definition.

### Naming Rules

Naming rule issues include the following:

- [Rules for Naming Objects](#)

## ■ Case Sensitivity

### Rules for Naming Objects

Oracle and Sybase use different database object naming rules. For example, the maximum number of characters allowed for each object name can be different. Also, the use of single and double quotation marks, case sensitivity, and the use of alphanumeric characters can all be different.

**See Also:** *Oracle Database Reference* and Sybase documentation.

### Case Sensitivity

The Oracle database defaults to uppercase unless you surround identifiers with double quote characters. For example, to refer to the Sybase table called emp, enter the name with double quote characters, as follows:

```
SQL> SELECT * FROM "emp"@SYBS;
```

However, to refer to the Sybase table called emp owned by Scott from an Oracle application, enter the following:

```
SQL> SELECT * FROM "Scott"."emp"@SYBS;
```

If the Sybase table called emp is owned by SCOTT, a table owner name in uppercase letters, you can enter the owner name without double quote characters, as follows:

```
SQL> SELECT * FROM SCOTT."emp"@SYBS;
```

Or

```
SQL> SELECT * FROM scott."emp"@SYBS;
```

Oracle Corporation recommends that you surround all Sybase object names with double quote characters and use the exact letter case for the object names as they appear in the Sybase data dictionary. This convention is not required when referring to the supported Oracle data dictionary tables or views listed in [Appendix C, "Data Dictionary"](#).

If existing applications cannot be changed according to these conventions, create views in Oracle to associate Sybase names to the correct letter case. For example, to refer to the Sybase table emp from an existing Oracle application by using only uppercase names, define the following view:

```
SQL> CREATE VIEW EMP (EMPNO, ENAME, SAL, HIREDATE)
      AS SELECT "empno", "ename", "sal", "hiredate"
      FROM "emp"@SYBS;
```

With this view, the application can issue statements such as the following:

```
SQL> SELECT EMPNO, ENAME FROM EMP;
```

Using views is a workaround solution that duplicates data dictionary information originating in the Sybase data dictionary. You must be prepared to update the Oracle view definitions whenever the data definitions for the corresponding tables are changed in the Sybase database.

## Data Types

Data type issues include the following:

### ■ Binary Literal Notation

- [Data Type Conversion](#)

### Binary Literal Notation

Oracle SQL uses hexadecimal digits surrounded by single quotes to express literal values being compared or inserted into columns defined as data type RAW.

This notation is not converted to syntax compatible with the Sybase VARBINARY and BINARY data types (a 0x followed by hexadecimal digits, surrounded by single quotes).

For example, the following statement is not supported:

```
SQL> INSERT INTO BINARY_TAB@SYBS VALUES ('0xff')
```

Where BINARY\_TAB contains a column of data type VARBINARY or BINARY. Use bind variables when inserting into or updating VARBINARY and BINARY data types.

### Data Type Conversion

Sybase does not support implicit date conversions. Such conversions must be explicit.

For example, the gateway issues an error for the following SELECT statement:

```
SELECT DATE_COL FROM TEST@
SYBS
WHERE DATE_COL = "1-JAN-2001";
```

To avoid problems with implicit conversions, add explicit conversions, as in the following:

```
SELECT DATE_COL FROM TEST@
SYBS
WHERE DATE_COL = TO_DATE("1-JAN-2001")
```

**See Also:** [Appendix A, "Data Type Conversion"](#) for more information about restrictions on data types.

## Queries

Query issues include the following:

- [Row Selection](#)
- [Empty Strings](#)
- [Empty Bind Variables](#)

### Row Selection

Sybase evaluates a query condition for all selected rows before returning any of the rows. If there is an error in the evaluation process for one or more rows, no rows are returned even though the remaining rows satisfy the condition.

Oracle evaluates the query condition row-by-row and returns a row when the evaluation is successful. Rows are returned until a row fails the evaluation.

### Empty Strings

Oracle processes an empty string in a SQL statement as a null value. Sybase processes an empty string as an empty string.

Comparing to an empty string

The Gateway passes literal empty strings to the Sybase database without any conversion. If you intended an empty string to represent a null value, Sybase does not process the statement that way; it uses the empty string.

You can avoid this problem by using `NULL` or `IS NULL` in the SQL statement instead of the empty string syntax, as in the following example:

```
SELECT * from "emp"@SYBS where "ename" IS NULL;
```

Selecting an empty string

For `VARCHAR` columns, the gateway returns an empty string to the Oracle database as `NULL` value.

For `CHAR` columns, the gateway returns the full size of the column with each character as empty space (' ').

### Empty Bind Variables

For `VARCHAR` bind variables, the gateway passes empty bind variables to the Sybase database as a `NULL` value.

## Locking

The locking model for an Sybase database differs significantly from the Oracle model. The gateway depends on the underlying Sybase behavior, so Oracle applications that access Sybase through the gateway can be affected by the following possible scenarios:

- Read access might block write access
- Write access might block read access
- Statement-level read consistency is not guaranteed

**See Also:** Sybase documentation for information about the Sybase locking model.

## Known Restrictions

If you encounter incompatibility problems not listed in this section or in "[Known Problems](#)" on page 2-13, please contact Oracle Support Services. The following section describes the known restrictions and includes suggestions for dealing with them when possible:

- [Transactional Integrity](#)
- [Transaction Capability](#)
- [COMMIT or ROLLBACK in PL/SQL Cursor Loops Closes Open Cursors](#)
- [Stored Procedures](#)
- [Pass-Through Feature](#)
- [Sybase NCHAR and NVARCHAR Data Types](#)
- [SQL Syntax](#)
- [SQL\\*Plus COPY Command with Lowercase Table Names](#)
- [Database Links](#)

The following restriction also applies:

- When negative numbers are used as the second parameter in a SUBSTR function, incorrect results are returned. This is due to incompatibility between the Oracle SUBSTR function and the equivalent in Sybase.

---

**Note:** If you have any questions or concerns about the restrictions, contact Oracle Support Services.

---

## Transactional Integrity

The gateway cannot guarantee transactional integrity in the following cases:

- When a statement that is processed by the gateway causes an implicit commit in the target database
- When the target database is configured to work in autocommit mode

---

**Note:** Oracle corporation strongly recommends the following:

- If you know that executing a particular statement causes an implicit commit in the target database, then ensure that this statement is executed in its own transaction.
  - Do not configure the target database to work in autocommit mode.
- 

## Transaction Capability

The gateway does not support savepoints. If a distributed update transaction is under way involving the gateway and a user attempts to create a savepoint, the following error occurs:

```
ORA-02070: database dblink does not support savepoint in this context
```

By default, the gateway is configured as COMMIT\_CONFIRM and it is always the commit point site when the Sybase database is updated by the transaction.

## COMMIT or ROLLBACK in PL/SQL Cursor Loops Closes Open Cursors

Any COMMIT or ROLLBACK issued in a PL/SQL cursor loop closes all open cursors, which can result in the following error:

```
ORA-1002: fetch out of sequence
```

To prevent this error, move the COMMIT or ROLLBACK statement outside the cursor loop.

## Stored Procedures

Changes issued through stored procedures that embed commits or rollbacks cannot be controlled by the Oracle transaction manager or Oracle COMMIT or ROLLBACK commands.

When accessing stored procedures with result sets through the Oracle Database Gateway for Sybase, you must work in the sequential mode of Heterogeneous Services.

When accessing stored procedures with multiple result sets through the Oracle Database Gateway for Sybase, you must read all the result sets before continuing.



Output parameters of stored procedures must be initialized to a `NULL` value.

You cannot use `RETURN_VALUE` nor `RETURNVALUE` as the names for any input or output parameters of stored procedures.

Oracle Database Gateway for Sybase does not support output parameters or stored procedures with output parameters, inside a pass through query.

## Pass-Through Feature

DDL statements executed by Sybase using the gateway pass-through feature might fail if they are in a multi-statement transaction. Set the Sybase option "ddl in tran" to allow DDL statements in a transaction.

Oracle Corporation recommends that you place a DDL statement in its own transaction when executing such a statement with the pass-through feature. An explicit `COMMIT` must be issued after the DDL statement.

If the SQL statements being passed through the gateway result in an implicit commit at the Sybase database, the Oracle transaction manager is unaware of the commit and an Oracle `ROLLBACK` command cannot be used to roll back the transaction.

## Sybase NCHAR and NVARCHAR Data Types

The gateway cannot select a column defined with a Sybase `NCHAR` or `NVARCHAR` data type.

## SQL Syntax

This section lists restrictions on the following SQL syntax:

- [WHERE CURRENT OF Clause](#)
- [CONNECT BY Clause](#)
- [ROWID](#)
- [Subqueries in INSERT Statement](#)
- [EXPLAIN PLAN Statement](#)
- [Callback Support](#)

**See Also:** [Appendix B, "Supported SQL Syntax and Functions"](#) for more information about restrictions on SQL syntax.

### WHERE CURRENT OF Clause

`UPDATE` and `DELETE` statements with the `WHERE CURRENT OF` clause are not supported by the gateway because they rely on the Oracle `ROWID` implementation. To update or delete a specific row through the gateway, a condition style `WHERE` clause must be used.

### CONNECT BY Clause

The gateway does not support the `CONNECT BY` clause in a `SELECT` statement.

### ROWID

The Oracle `ROWID` implementation is not supported.

### Subqueries in INSERT Statement

Subqueries of INSERT statements cannot use multiple aliases for the same table. For example, the following statement is not supported:

```
SQL> INSERT INTO "emp_target"@SYBS
      SELECT a."empno" FROM "emp_source"@SYBS a,
      "emp_source"@SYBS b WHERE b."empno"=9999
```

### EXPLAIN PLAN Statement

The EXPLAIN PLAN statement is not supported.

### Callback Support

SQL statements that require the gateway to callback to Oracle database would not be supported.

The following categories of SQL statements will result in a callback:

- Any DML with a sub-select, which refers to a table in Oracle database. For example:  

```
INSERT INTO emp@non_oracle SELECT * FROM oracle_emp;
```
- Any DELETE, INSERT, UPDATE or "SELECT... FOR UPDATE..." SQL statement containing SQL functions or statements that need to be executed at the originating Oracle database.

These SQL functions include USER, USERENV, and SYSDATE, and the SQL statements are in selects of data from the originating Oracle database. For example:

```
DELETE FROM emp@non_oracle WHERE hiredate > SYSDATE;
```

```
SELECT ename FROM tkhoemp@non_oracle
WHERE hiredate IN (SELECT hiredate FROM tkhoemp)
FOR UPDATE OF empno;
```

- Any SQL statement that involves a table in Oracle database, and a LONG or LOB column in a remote table. For example:

```
SELECT a.long1, b.empno FROM scott.table@non_oracle a, emp b
WHERE a.id=b.empno;
```

```
SELECT a.long1, b.dummy FROM table_non@non_oracle a, dual b;
```

where a.long1 is a LONG column.

### SQL\*Plus COPY Command with Lowercase Table Names

You need to use double quotes to wrap around lower case table names.

For example:

```
copy from tkhouser/tkhouser@inst1 insert loc_tkhodept using select * from
"tkhodept"@holink2;
```

### Database Links

The gateway is not multithreaded and cannot support shared database links. Each gateway session spawns a separate gateway process and connections cannot be shared.

## Known Problems

This section describes known problems and includes suggestions for correcting them when possible. If you have any questions or concerns about the problems, contact Oracle Support Services. A current list of problems is available online. Contact your local Oracle Corporation office for information about accessing the list.

The following known problems are described in this section:

- [FLOAT Data Type](#)
- [VARBINARY Data Type](#)
- [Encrypted Format Login](#)
- [Date Arithmetic](#)
- [Sybase IMAGE and TEXT Data Type](#)
- [String Functions](#)
- [Schema Names and PL/SQL](#)
- [Data Dictionary Views and PL/SQL](#)
- [Stored Procedures](#)

### FLOAT Data Type

The `FLOAT` data type precision is (7).

### VARBINARY Data Type

The `VARBINARY` data type is reported as `BINARY`.

### Encrypted Format Login

Oracle database no longer supports the initialization parameter `DBLINK_ENCRYPT_LOGIN`. Up to version 7.3, this parameter's default `TRUE` value prevented the password for the login user ID from being sent over the network (in the clear). Later versions automatically encrypt the password.

### Date Arithmetic

The following SQL expressions do not function correctly with the gateway:

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

Statements with the preceding expressions are sent to the Sybase database without any translation. Since Sybase does not support these date arithmetic functions, the statements return an error.

### Sybase IMAGE and TEXT Data Type

The following restrictions apply when using `IMAGE` and `TEXT` data types:

- An unsupported SQL function cannot be used in a SQL statement that accesses a column defined as Sybase data type `IMAGE` or `TEXT`.

- You cannot use SQL\*Plus to select data from a column defined as Sybase data type `IMAGE` or `TEXT` when the data is greater than 80 characters in length. Oracle Corporation recommends using Pro\*C or Oracle Call Interface to access such data in a Sybase database.
- `IMAGE` and `TEXT` data types must be `NULLABLE` for `INSERT` or `UPDATE` to work.
- A table including a `IMAGE` or `TEXT` column must have a unique index defined on the table or the table must have a separate column that serves as a primary key.
- `IMAGE` and `TEXT` data cannot be read through pass-through queries.

The gateway does not support the PL/SQL function `COLUMN_VALUE_LONG` of the `DBMS_SQL` package.

**See Also:** [Appendix B, "Supported SQL Syntax and Functions"](#) for more information about restrictions on SQL syntax.

## String Functions

If you concatenate numeric literals using the `||` operator when using the gateway to query a Sybase database, the result is an arithmetic addition. For example, the result of the following statement is 18:

```
SQL> SELECT 9 || 9 FROM DUAL@SYBS;
```

The result is 99 when using Oracle to query an Oracle database.

## Schema Names and PL/SQL

If you do not prefix a Sybase database object with its schema name in a SQL statement within a PL/SQL block, the following error message occurs:

```
ORA-6550 PLS-201 Identifier table_name must be declared.
```

Change the SQL statement to include the schema name of the object.

## Data Dictionary Views and PL/SQL

You cannot refer to data dictionary views in SQL statements that are inside a PL/SQL block.

## Stored Procedures

Stored procedures with output parameters defined with a `CHAR` data type return output parameters with `VARCHAR` data types.

Stored procedures with input or output parameters defined with `NUMERIC` or `DECIMAL` data types that are passed with either a `NULL` value or no value for input will return output values without the fractional part. To prevent this from happening, always specify an input parameter value, even if it is not used.

Return values of stored procedures which return result sets are incorrect.

---

## Case Studies

The following case studies for Sybase demonstrate some of the features of the Oracle Database Gateway. You can verify that the gateway is installed and operating correctly by using the demonstration files included on the distribution media.

The demonstration files are automatically copied to disk when the gateway is installed.

This chapter contains the following sections:

- [Case Descriptions](#)
- [Distribution Media Contents](#)
- [Demonstration Files](#)
- [Demonstration Requirements](#)
- [Creating Demonstration Tables](#)
- [Case 1: Simple Queries](#)
- [Case 2: A More Complex Query](#)
- [Case 3: Joining Sybase Tables](#)
- [Case 4: Write Capabilities](#)
- [Case 5: Data Dictionary Query](#)
- [Case 6: The Pass-Through Feature](#)
- [Case 7: Executing Stored Procedures](#)

### Case Descriptions

The cases illustrate:

- A simple query (Case 1)
- A more complex query (Case 2)
- Joining Sybase tables (Case 3)
- Write capabilities (Case 4)
- A data dictionary query (Case 5)
- The pass-through feature (Case 6)
- Executing stored procedures (Case 7)

## Distrubution Media Contents

The distribution media contains the following:

- Demonstration files
- One SQL script file that creates the demonstration tables and stored procedures in the Sybase database
- One SQL script file that drops the demonstration tables and stored procedures from the Sybase database

## Demonstration Files

After a successful gateway installation, use the demonstration files stored in the directory `$ORACLE_HOME/tg4sybs/demo`, where `$ORACLE_HOME` is the directory under which the gateway is installed. The directory contains the following demonstration files:

Demonstration Files	Demonstration Files
<code>bldsybs.sql</code>	<code>case4c.sql</code>
<code>case1.sql</code>	<code>case5.sql</code>
<code>case2.sql</code>	<code>case6a.sql</code>
<code>case3.sql</code>	<code>case6b.sql</code>
<code>case4a.sql</code>	<code>case7.sql</code>
<code>case4b.sql</code>	<code>dropsybs.sql</code>

## Demonstration Requirements

The case studies assume these requirements have been met:

- The gateway demonstration tables and stored procedures are installed in the Sybase database
- The Oracle database has an account named `SCOTT` with a password of `TIGER`
- The Oracle database has a database link called `GTWLINK` (set up as public or private to the user `SCOTT`) which connects the gateway to a Sybase database as `SCOTT` with password `TIGER2`.

For example, you can create the database link as follows:

```
SQL> CREATE DATABASE LINK GTWLINK CONNECT TO SCOTT
      2 IDENTIFIED BY TIGER2 USING 'GTWSID';
```

- Oracle Net Services is configured correctly and running
- The Sybase environment variable, `SYBASE`, is set correctly

## Creating Demonstration Tables

The case studies are based on the `GTW_EMP`, `GTW_DEPT`, and `GTW_SALGRADE` tables and the stored procedures `InsertDept` and `GetDept`. If the demonstration tables and stored procedures have not been created in the Sybase database, use the `bldsybs.sql` script to create them. Enter the following:

```
$ isql -USCOTT -PTIGER2 -ibldsybs.sql
```

The script creates the demonstration tables and stored procedures in the Sybase database accordingly:

```
CREATE TABLE GTW_EMP (
EMPNO      SMALLINT NOT NULL
ENAME      VARCHAR(10),
JOB        VARCHAR(9),
MGR        SMALLINT,
HIREDATE   DATETIME,
SAL        NUMERIC(7,2),
COMM       NUMERIC(7,2),
DEPTNO     SMALLINT)
go

CREATE TABLE GTW_DEPT (
DEPTNO     SMALLINT NOT NULL,
DNAME      VARCHAR(14),
LOC        VARCHAR(13))
go

CREATE TABLE GTW_SALGRADE (
GRADE      MONEY,
LOSAL      NUMERIC(9,4),
HISAL      NUMERIC(9,4))
go

SET CHAINED ON
go

DROP PROCEDURE InsertDept
go

CREATE PROCEDURE InsertDept (@dno INTEGER,
                             @dname VARCHAR(14), @loc VARCHAR(13))
AS INSERT INTO GTW_DEPT VALUES (@dno, @dname, @loc)
go

DROP PROCEDURE GetDept
go

CREATE PROCEDURE GetDept (@dno INTEGER, @dname VARCHAR(14) OUTPUT)
AS SELECT @dname=DNAME FROM GTW_DEPT WHERE DEPTNO=@dno
go
```

## Demonstration Table Definitions

The table definitions are listed in the following table using information retrieved by the SQL\*PLUS DESCRIBE command:

### GTW\_EMP

Name	Null?	Type
EMPNO	NOT NULL	NUMBER(5)
ENAME		VARCHAR2(10)
JOB		VARCHAR2(9)
MGR		NUMBER(5)
HIREDATE		DATE
SAL		NUMBER(7,2)
COMM		NUMBER(7,2)
DEPTNO		NUMBER(5)

### GTW\_DEPT

Name	Null?	Type
DEPTNO	NOT NULL	NUMBER (5)
DNAME		VARCHAR2 (14)
LOC		VARCHAR2 (13)

### GTW\_SALGRADE

Name	Null?	Type
GRADE		NUMBER (19, 4)
LOSAL		NUMBER (9, 4)
HISAL		NUMBER (9, 4)

## Demonstration Table Contents

The contents of the Sybase tables are:

### GTW\_EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7369	SMITH	CLERK	7902	17-DEC-80	800		20
7499	ALLEN	SALESMAN	7698	20-FEB-81	1600	300	30
7521	WARD	SALESMAN	7698	22-FEB-81	1250	500	30
7566	JONES	MANAGER	7839	02-APR-81	2975		20
7654	MARTIN	SALESMAN	7698	28-SEP-81	1250	1400	30
7698	BLAKE	MANAGER	7839	01-MAY-81	2850		30
7782	CLARK	MANAGER	7839	09-JUN-81	2450		10
7788	SCOTT	ANALYST	7566	09-DEC-82	3000		20
7839	KING	PRESIDENT		17-NOV-81	5000		10
7844	TURNER	SALESMAN	7698	08-SEP-81	1500	0	30
7876	ADAMS	CLERK	7788	12-JAN-83	1100		20
7900	JAMES	CLERK	7698	03-DEC-81	950		30
7902	FORD	ANALYST	7566	03-DEC-81	3000		20
7934	MILLER	CLERK	7782	23-JAN-82	1300		10

### GTW\_DEPT

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

### GTW\_SALGRADE

GRADE	LOSAL	HISAL
1	700	1200
2	1201	1400
3	1401	2000
4	2001	3000
5	3001	9999



## Case 1: Simple Queries

Case 1 demonstrates the following:

- A simple query
- A simple query retrieving full date information

The first query retrieves all the data from `GTW_DEPT` and confirms that the gateway is working correctly. The second query retrieves all the data from `GTW_EMP` including the time portion of the hire date because the default date format was set to `DD-MON-YY HH24:MM:SS` for the session by an `ALTER SESSION` command.

## Case 2: A More Complex Query

Case 2 demonstrates the following:

- The functions `SUM(expression)` and `NVL(expr1, expr2)` in the `SELECT` list
- The `GROUP BY` and `HAVING` clauses

This query retrieves the departments from `GTW_EMP` whose total monthly expenses are higher than \$10,000.

## Case 3: Joining Sybase Tables

Case 3 demonstrates the following:

- Joins between Sybase tables
- Subselects

The query retrieves information from three Sybase tables and relates the employees to their department name and salary grade, but only for those employees earning more than the average salary.

## Case 4: Write Capabilities

Case 4 is split into three cases and demonstrates the following:

- [DELETE Statement](#)
- [UPDATE Statement](#)
- [INSERT Statement](#)

### DELETE Statement

Case 4a demonstrates bind values and subselect. All employees in department 20 and one employee, `WARD`, in department 30 are deleted.

### UPDATE Statement

Case 4b provides an example of a simple `UPDATE` statement. In this example, employees are given a \$100 a month salary increase.

### INSERT Statement

Case 4c is an example of a simple insert statement that does not provide information for all columns.

## Case 5: Data Dictionary Query

Case 5 demonstrates data dictionary mapping. It retrieves all the tables and views that exist in the Sybase database that begin with "GTW".

## Case 6: The Pass-Through Feature

Case 6 demonstrates the gateway pass-through feature which allows an application to send commands or statements to Sybase.

This case demonstrates:

- A pass-through UPDATE statement using bind variables
- A pass-through SELECT statement

### UPDATE Statement

Case 6a provides an example of a pass-through UPDATE statement with bind variables. In this example, the salary for EMPNO 7934 is set to 4000.

### SELECT Statement

Case 6b provides an example of a pass-through SELECT statement. The data that is returned from the SELECT statement is inserted into a local table at the Oracle database.

## Case 7: Executing Stored Procedures

Case 7 demonstrates the gateway executing a stored procedure in the Sybase database.

Before running the script, make sure that chained mode is set on, by issuing the following command in Sybase:

```
set chained on
```

## Data Type Conversion

This appendix contains the following section:

- [Data Type Conversion](#)

### Data Type Conversion

The gateway converts Sybase data types to Oracle data types as follows:

**Table A-1 Data Type Conversions**

Sybase	Oracle	Comment
BINARY	RAW	-
BIT	NUMBER (3)	-
CHAR	CHAR	-
DATETIME	DATE	-
DECIMAL	NUMBER (18)	-
DOUBLE PRECISION	FLOAT (53)	-
FLOAT	FLOAT (53)	-
IMAGE	LONG RAW	-
INT	NUMBER (10)	NUMBER range is -2,147,483,647 to 2,147,483,647
MONEY	NUMBER (p [, s])	-
NCHAR	Not supported	-
NUMERIC	NUMBER (p [, s])	-
NVARCHAR	Not supported	-
REAL	FLOAT (24)	-
SMALLDATETIME	DATE	-
SMALLINT	NUMBER (5)	NUMBER range is -32,767 to 32,767
SMALL MONEY	NUMBER (p [, s])	-
TEXT	LONG	-
TINYINT	NUMBER (3)	-
UNICHAR	NCHAR	-
UNIVARCHAR	NVARCHAR	-

**Table A-1 (Cont.) Data Type Conversions**

<b>Sybase</b>	<b>Oracle</b>	<b>Comment</b>
VARBINARY	RAW	-
VARCHAR	VARCHAR2	-

---

## Supported SQL Syntax and Functions

---

This appendix contains the following sections:

- [Supported SQL Statements](#)
- [Oracle Functions](#)

### Supported SQL Statements

With a few exceptions, the gateway provides full support for Oracle `DELETE`, `INSERT`, `SELECT`, and `UPDATE` statements.

The gateway does not support Oracle data definition language (DDL) statements. No form of the Oracle `ALTER`, `CREATE`, `DROP`, `GRANT`, or `TRUNCATE` statements can be used. Instead, for `ALTER`, `CREATE`, `DROP`, and `GRANT` statements, use the pass-through feature of the gateway if you need to use DDL statements against the Sybase database.

---

**Note:** `TRUNCATE` cannot be used in a pass-through statement.

---

**See Also:** *Oracle Database SQL Language Reference* for detailed descriptions of keywords, parameters, and options and the Sybase documentation for details of executing SQL statements in a transaction.

#### DELETE

The `DELETE` statement is fully supported. However, only Oracle functions supported by Sybase can be used.

**See Also:** ["Functions Supported by Sybase"](#) on page B-2 for a list of supported functions.

#### INSERT

The `INSERT` statement is fully supported. However, only Oracle functions supported by Sybase can be used.

**See Also:** ["Functions Supported by Sybase"](#) on page B-2 for a list of supported functions.

#### SELECT

The `SELECT` statement is fully supported, with these exceptions:

- `CONNECT BY condition`
- `NOWAIT`
- `START WITH condition`
- `WHERE CURRENT OF`

## UPDATE

The `UPDATE` statement is fully supported. However, only Oracle functions supported by Sybase can be used. Also, you cannot have SQL statements in the subquery that refer to the same table name in the outer query. Subqueries are not supported in the `SET` clause.

**See Also:** ["Functions Supported by Sybase"](#) on page B-2 for a list of supported functions.

## Oracle Functions

All functions are evaluated by the Sybase database after the gateway has converted them to Sybase SQL.

### Functions Not Supported by Sybase

Oracle SQL functions with no equivalent function in Sybase are not supported in `DELETE`, `INSERT`, or `UPDATE` statements, but are evaluated by the Oracle database if the statement is a `SELECT` statement. That is, the Oracle database performs post-processing of `SELECT` statements sent to the gateway.

If an unsupported function is used in a `DELETE`, `INSERT`, or `UPDATE`, statement, the following Oracle error occurs:

ORA-02070: database *db\_link\_name* does not support *function* in this context

### Functions Supported by Sybase

The gateway translates the following Oracle database functions in SQL statements to their equivalent Sybase functions:

- [Arithmetic Operators](#)
- [Comparison Operators](#)
- [Group Functions](#)
- [String Functions](#)
- [Other Functions](#)

#### Arithmetic Operators

Oracle	Sybase
+	+
-	-
*	*
/	/

## Comparison Operators

Oracle	Sybase
=	=
>	>
<	<
>=	>=
<=	<=
<>, !=, ^=	<>, !=
IS NOT NULL	IS NOT NULL
IS NULL	IS NULL

## Group Functions

Oracle	Sybase
AVG	AVG
COUNT	COUNT
MAX	MAX
MIN	MIN
SUM	SUM

## String Functions

Oracle	Sybase
	+ ( <i>expression1</i> + <i>expression2</i> )
ASCII	ASCII
CHR	CHAR
INSTR	CHARINDEX
LENGTH	CHAR_LENGTH
LOWER	LOWER
SUBSTR (second argument cannot be a negative number)	SUBSTRING
UPPER	UPPER

## Other Functions

Oracle	Sybase
ABS	ABS
CEIL	CEILING
COS	COS
EXP	EXP

Oracle	Sybase
FLOOR	FLOOR
LN	LOG
LOG	LOG10
MOD	%
NOT NVL	IS NOT NULL
NVL	IS NULL
POWER	POWER
ROUND	ROUND
SIN	SIN
SQRT	SQRT
TAN	TAN



---

## Data Dictionary

The Oracle Database Gateway for Sybase translates a query that refers to an Oracle database data dictionary table into a query that retrieves the data from Sybase system tables. You perform queries on data dictionary tables over the database link in the same way you query data dictionary tables in the Oracle database. The gateway data dictionary is similar to the Oracle database data dictionary in appearance and use.

This appendix contains the following sections:

- [Data Dictionary Support](#)
- [Data Dictionary Mapping](#)
- [Gateway Data Dictionary Descriptions](#)

### Data Dictionary Support

The following paragraphs describe the Oracle Database Gateway for Sybase data dictionary support.

#### Sybase System Tables

Sybase data dictionary information is stored in the Sybase database as Sybase system tables. All Sybase system tables have names prefixed with "sys". The Sybase system tables define the structure of a database. When you change data definitions, Sybase reads and modifies the Sybase system tables to add information about the user tables.

#### Accessing the Gateway Data Dictionary

Accessing a gateway data dictionary table or view is identical to accessing a data dictionary in an Oracle database. You issue a SQL `SELECT` statement specifying a database link. The Oracle database data dictionary view and column names are used to access the gateway data dictionary in an Oracle database. Synonyms of supported views are also acceptable. For example, the following statement queries the data dictionary table `ALL_CATALOG` to retrieve all table names in the Sybase database:

```
SQL> SELECT * FROM "ALL_
CATALOG
"@SYBS;
```

When a data dictionary access query is issued, the gateway:

1. Maps the requested table, view, or synonym to one or more Sybase system table names. The gateway translates all data dictionary column names to their corresponding Sybase column names within the query. If the mapping involves one Sybase system table, the gateway translates the requested table name to its

corresponding Sybase system table name within the query. If the mapping involves multiple Sybase system tables, the gateway constructs a join in the query using the translated Sybase system table names.

2. Sends the translated query to Sybase.
3. Might convert the retrieved Sybase data to give it the appearance of the Oracle database data dictionary table.
4. Passes the data dictionary information from the translated Sybase system table to the Oracle database.

---

**Note:** The values returned when querying the gateway data dictionary might not be the same as the ones returned by the Oracle SQL\*Plus DESCRIBE command.

---

## Direct Queries to Sybase Tables

Queries issued directly to individual Sybase system tables are allowed but they return different results because the Sybase system table column names differ from those of the data dictionary view. Also, certain columns in an Sybase system table cannot be used in data dictionary processing.

## Supported Views and Tables

The gateway supports the following views and tables:

Supported Views and Tables	Supported Views and Tables
ALL_CATALOG	ALL_COL_COMMENTS
ALL_CONS_COLUMNS	ALL_CONSTRAINTS
ALL_IND_COLUMNS	ALL_INDEXES
ALL_OBJECTS	ALL_TAB_COLUMNS
ALL_TAB_COMMENTS	ALL_TABLES
ALL_USERS	ALL_VIEWS
DBA_CATALOG	DBA_COL_COMMENTS
DBA_OBJECTS	DBA_TABLES
DBA_TAB_COLUMNS	DBA_TAB_COMMENTS
DICT_COLUMNS	DICTIONARY
DUAL	TABLE_PRIVILEGES
USER_CATALOG	USER_COL_COMMENTS
USER_CONS_COLUMNS	USER_CONSTRAINTS
USER_IND_COLUMNS	USER_INDEXES
USER_OBJECTS	USER_TAB_COLUMNS
USER_TAB_COMMENTS	USER_TABLES
USER_USER	USER_VIEWS

No other Oracle database data dictionary tables or views are supported. If you use a view not on the list, you receive the Oracle database error code for no more rows available.

Queries through the gateway of any data dictionary table or view beginning with ALL\_ can return rows from the Sybase database even when access privileges for those Sybase objects have not been granted. When querying an Oracle database with the Oracle data dictionary, rows are returned only for those objects you are permitted to access.

## Data Dictionary Mapping

The tables in this section list Oracle data dictionary view names and the equivalent Sybase system tables used. A plus sign (+) indicates that a join operation is involved.

**Table C-1 Oracle Data Dictionary View Names and Sybase Equivalents**

View Name	Sybase System Table Name
ALL_CATALOG	sysusers + sysobjects
ALL_COL_COMMENTS	sysusers + sysobjects + syscolumns
ALL_CONS_COLUMNS	sp_pkeys + sp_fkeys
ALL_CONSTRAINTS	sysconstraints + sysusers + sysobjects
ALL_IND_COLUMNS	sysusers + sysindexes + syscolumns
ALL_INDEXES	sysusers + sysindexes + sysobjects
ALL_OBJECTS	sysusers + sysobjects + sysindexes
ALL_TAB_COLUMNS	sysusers + sysobjects + syscolumns
ALL_TAB_COMMENTS	sysusers + sysobjects
ALL_TABLES	sysusers + sysobjects
ALL_USERS	sysusers
ALL_VIEWS	sysusers + sysobjects + syscomments
DBA_CATALOG	sysusers + sysobjects
DBA_COL_COMMENTS	sysusers + sysobjects + syscolumns
DBA_OBJECTS	sysusers + sysobjects + sysindexes
DBA_TABLES	sysusers + sysobjects
DBA_TAB_COLUMNS	sysusers + sysobjects + syscolumns
DBA_TAB_COMMENTS	sysusers + sysobjects
DICT_COLUMNS	sysobjects + syscolumns
DICTIONARY	sysobjects
DUAL	(Defined in the Gateway)
TABLE_PRIVILEGES	sysprotects + sysusers + sysobjects
USER_CATALOG	sysusers + sysobjects
USER_COL_COMMENTS	sysusers + sysobjects + syscolumns
USER_CONS_COLUMNS	sp_pkeys + sp_fkeys

**Table C–1 (Cont.) Oracle Data Dictionary View Names and Sybase Equivalents**

View Name	Sybase System Table Name
USER_CONSTRAINTS	sysconstraints + sysusers + sysobjects
USER_IND_COLUMNS	sysusers + sysindexes + syscolumns
USER_INDEXES	sysusers + sysindexes + sysobjects
USER_OBJECTS	sysusers + sysobjects + sysindexes
USER_TAB_COLUMNS	sysusers + sysobjects + syscolumns
USER_TAB_COMMENTS	sysusers + sysobjects
USER_TABLES	sysusers + sysobjects
USER_USERS	sysusers
USER_VIEWS	sysusers + sysobjects + syscomments

## Default Column Values

There is a minor difference between the gateway data dictionary and a typical Oracle database data dictionary. The Oracle database columns that are missing in an Sybase system table are filled with zeros, spaces, null values, not-applicable values (N.A.), or default values, depending on the column type.

## Gateway Data Dictionary Descriptions

The gateway data dictionary tables and views provide the following information:

- Name, data type, and width of each column
- The contents of columns with fixed values

They are described here with information retrieved by an Oracle SQL\*Plus DESCRIBE command. The values in the Null? column might differ from the Oracle database data dictionary tables and views. Any default value is shown to the right of an item, but this is not information returned by DESCRIBE.

**Table C–2 ALL\_CATALOG**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"

**Table C–3 ALL\_COL\_COMMENTS**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	NOT NULL	VARCHAR2 (30)	-
COMMENTS	NOT NULL	CHAR (1)	" "

**Table C-4 ALL\_CONS\_COLUMNS**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
CONSTRAINT_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (8192)	-
POSITION	-	FLOAT (49)	-

**Table C-5 ALL\_CONSTRAINTS**

Name	Null?	Type	Value
OWNER	-	VARCHAR2 (30)	-
CONSTRAINT_NAME	-	VARCHAR2 (30)	-
CONSTRAINT_TYPE	-	VARCHAR2 (1)	"R" or "P"
TABLE_NAME	-	VARCHAR2 (30)	-
SEARCH_CONDITION	-	VARCHAR2 (1)	NULL
R_OWNER	-	VARCHAR2 (30)	-
R_CONSTRAINT_NAME	-	VARCHAR2 (30)	-
DELETE_RULE	-	VARCHAR2 (9)	"NO ACTION" or " "
STATUS	-	VARCHAR2 (7)	"ENABLED"
DEFERRABLE	-	VARCHAR2 (1)	NULL
DEFERRED	-	VARCHAR2 (1)	NULL
VALIDATED	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
BAD	-	VARCHAR2 (1)	NULL
RELY	-	VARCHAR2 (1)	NULL
LAST_CHANGE	-	DATE	-

**Table C-6 ALL\_IND\_COLUMNS**

Name	Null?	Type	Value
INDEX_OWNER	NOT NULL	VARCHAR2 (30)	-
INDEX_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (8192)	-
COLUMN_POSITION	NOT NULL	FLOAT (49)	-
COLUMN_LENGTH	NOT NULL	FLOAT (49)	-
DESCEND	-	VARCHAR (4)	"DESC" or "ASC"

**Table C-7 ALL\_INDEXES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
OWNER	NOT NULL	VARCHAR2 (30)	-
INDEX_NAME	-	VARCHAR2 (30)	-
INDEX_TYPE	-	VARCHAR2 (1)	NULL
TABLE_OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (7)	"TABLE" or "CLUSTER"
UNIQUENESS	-	VARCHAR2 (1)	NULL
COMPRESSION	-	VARCHAR2 (1)	NULL
PREFIX_LENGTH	-	NUMBER	0
TABLESPACE_NAME	-	VARCHAR2 (1)	NULL
INI_TRANS	-	NUMBER	0
MAX_TRANS	-	NUMBER	0
INITIAL_EXTENT	-	NUMBER	0
NEXT_EXTENT	-	NUMBER	0
MIN_EXTENTS	-	NUMBER	0
MAX_EXTENTS	-	NUMBER	0
PCT_INCREASE	-	NUMBER	0
PCT_THRESHOLD	-	NUMBER	0
INCLUDE_COLUMN	-	NUMBER	0
FREELISTS	-	NUMBER	0
FREELIST_GROUPS	-	NUMBER	0
PCT_FREE	-	NUMBER	0
LOGGING	-	VARCHAR2 (1)	NULL
BLEVEL	-	NUMBER	0
LEAF_BLOCKS	-	NUMBER	0
DISTINCT_KEYS	-	NUMBER	0
AVG_LEAF_BLOCKS_PER_KEY	-	NUMBER	0
AVG_DATA_BLOCKS_PER_KEY	-	NUMBER	0
CLUSTERING_FACTOR	-	NUMBER	0
STATUS	-	VARCHAR2 (1)	NULL
NUM_ROWS	-	NUMBER	0
SAMPLE_SIZE	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL
DEGREE	-	VARCHAR2 (1)	NULL
INSTANCES	-	VARCHAR2 (1)	NULL
PARTITIONED	-	VARCHAR2 (1)	NULL

**Table C-7 (Cont.) ALL\_INDEXES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
TEMPORARY	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL
BUFFER_POOL	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
DURATION	-	VARCHAR2 (1)	NULL
PCT_DIRECT_ACCESS	-	NUMBER	0
ITYP_OWNER	-	VARCHAR2 (1)	NULL
ITYP_NAME	-	VARCHAR2 (1)	NULL
PARAMETERS	-	VARCHAR2 (1)	NULL
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
DOMIDX_STATUS	-	VARCHAR2 (1)	NULL
DOMIDX_OPSTATUS	-	VARCHAR2 (1)	NULL
FUNCIDX_STATUS	-	VARCHAR2 (1)	NULL

**Table C-8 ALL\_OBJECTS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
OWNER	-	VARCHAR2 (30)	-
OBJECT_NAME	-	VARCHAR2 (30)	-
SUBOBJECT_NAME	-	VARCHAR2 (1)	NULL
OBJECT_ID	-	NUMBER	-
DATA_OBJECT_ID	-	NUMBER	0
OBJECT_TYPE	-	VARCHAR2 (9)	"TABLE" or "VIEW" or "PROCEDURE"
CREATED	-	DATE	-
LAST_DDL_TIME	-	DATE	-
TIMESTAMP	-	VARCHAR2 (1)	NULL
STATUS	-	VARCHAR2 (5)	"VALID"
TEMPORARY	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL

**Table C-9 ALL\_TAB\_COLUMNS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
OWNER	-	VARCHAR2 (30)	-
TABLE_NAME	-	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (30)	-

**Table C-9 (Cont.) ALL\_TAB\_COLUMNS**

Name	Null?	Type	Value
DATA_TYPE	-	VARCHAR2 (8)	-
DATA_TYPE_MOD	-	VARCHAR2 (1)	NULL
DATA_TYPE_OWNER	-	VARCHAR2 (1)	NULL
DATA_LENGTH	-	NUMBER	-
DATA_PRECISION	-	NUMBER	-
DATA_SCALE	-	NUMBER	-
NULLABLE	-	VARCHAR2 (1)	"Y" or "N"
COLUMN_ID	-	NUMBER	-
DEFAULT_LENGTH	-	NUMBER	0
DATA_DEFAULT	-	VARCHAR2 (1)	NULL
NUM_DISTINCT	-	NUMBER	0
LOW_VALUE	-	NUMBER	0
HIGH_VALUE	-	NUMBER	0
DENSITY	-	NUMBER	0
NUM_NULLS	-	NUMBER	0
NUM_BUCKETS	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL
SAMPLE_SIZE	-	NUMBER	0
CHARACTER_SET_NAME	-	VARCHAR2 (1)	NULL
CHAR_COL_DECL_LENGTH	-	NUMBER	0
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
AVG_COL_LEN	-	NUMBER	0

**Table C-10 ALL\_TAB\_COMMENTS**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"
COMMENTS	-	VARCHAR2 (1)	NULL

**Table C-11 ALL\_TABLES**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLESPACE_NAME	NOT NULL	CHAR (1)	" "



**Table C-11 (Cont.) ALL\_TABLES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
CLUSTER_NAME	NOT NULL	CHAR(1)	" "
IOT_NAME	NOT NULL	CHAR(1)	" "
PCT_FREE	NOT NULL	NUMBER(10)	0
PCT_USED	NOT NULL	NUMBER(10)	0
INI_TRANS	NOT NULL	NUMBER(10)	0
MAX_TRANS	NOT NULL	NUMBER(10)	0
INITIAL_EXTENT	NOT NULL	NUMBER(10)	0
NEXT_EXTENT	NOT NULL	NUMBER(10)	0
MIN_EXTENTS	NOT NULL	NUMBER(10)	0
MAX_EXTENTS	NOT NULL	NUMBER(10)	0
PCT_INCREASE	NOT NULL	NUMBER(10)	0
FREELISTS	NOT NULL	NUMBER(10)	0
FREELIST_GROUPS	NOT NULL	NUMBER(10)	0
LOGGING	NOT NULL	CHAR(1)	" "
BACKED_UP	NOT NULL	CHAR(1)	" "
NUM_ROWS	NOT NULL	NUMBER(10)	0
BLOCKS	NOT NULL	NUMBER(10)	0
EMPTY_BLOCKS	NOT NULL	NUMBER(10)	0
AVG_SPACE	NOT NULL	NUMBER(10)	0
CHAIN_CNT	NOT NULL	NUMBER(10)	0
AVG_ROW_LEN	NOT NULL	NUMBER(10)	0
AVG_SPACE_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0
NUM_FREELIST_BLOCKS	NOT NULL	NUMBER(10)	0
DEGREE	NOT NULL	CHAR(1)	" "
INSTANCES	NOT NULL	CHAR(1)	" "
CACHE	NOT NULL	CHAR(1)	" "
TABLE_LOCK	NOT NULL	CHAR(1)	" "
SAMPLE_SIZE	NOT NULL	NUMBER(10)	0
LAST_ANALYZED	-	DATE	NULL
PARTITIONED	NOT NULL	CHAR(1)	" "
IOT_TYPE	NOT NULL	CHAR(1)	" "
TEMPORARY	NOT NULL	CHAR(1)	" "
SECONDARY	NOT NULL	CHAR(1)	" "
NESTED	NOT NULL	CHAR(1)	" "
BUFFER_POOL	NOT NULL	CHAR(1)	" "
ROW_MOVEMENT	NOT NULL	CHAR(1)	" "
GLOBAL_STATS	NOT NULL	CHAR(1)	" "

**Table C-11 (Cont.) ALL\_TABLES**

Name	Null?	Type	Value
USER_STATS	NOT NULL	CHAR (1)	" "
DURATION	NOT NULL	CHAR (1)	" "
SKIP_CORRUPT	NOT NULL	CHAR (1)	" "
MONITORING	NOT NULL	CHAR (1)	" "

**Table C-12 ALL\_USERS**

Name	Null?	Type	Value
USERNAME	NOT NULL	VARCHAR2 (30)	-
USER_ID	NOT NULL	NUMBER (10)	-
CREATED	-	DATE	SYSDATE

**Table C-13 ALL\_VIEWS**

Name	Null?	Type	Value
OWNER	NOT NULL	CHAR (30)	-
VIEW_NAME	NOT NULL	CHAR (30)	-
TEXT_LENGTH	NOT NULL	NUMBER (10)	0
TEXT		VARCHAR2 (32512 )	-
TYPE_TEXT_LENGTH	NOT NULL	NUMBER (10)	0
TYPE_TEXT	NOT NULL	CHAR (1)	" "
OID_TEXT_LENGTH	NOT NULL	NUMBER (10)	0
OID_TEXT	NOT NULL	CHAR (1)	" "
VIEW_TYPE_OWNER	NOT NULL	CHAR (1)	" "
VIEW_TYPE	NOT NULL	CHAR (1)	" "

**Table C-14 DBA\_CATALOG**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"

**Table C-15 DBA\_COL\_COMMENTS**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	NOT NULL	VARCHAR2 (30)	-
COMMENTS	NOT NULL	CHAR (1)	" "

**Table C-16 DBA\_OBJECTS**

Name	Null?	Type	Value
OWNER	-	VARCHAR2 (30)	-
OBJECT_NAME	-	VARCHAR2 (30)	-
SUBOBJECT_NAME	-	VARCHAR2 (1)	NULL
OBJECT_ID	-	NUMBER	-
DATA_OBJECT_ID	-	NUMBER	0
OBJECT_TYPE	-	VARCHAR2 (9)	"TABLE" or "VIEW" or "PROCEDURE"
CREATED	-	DATE	-
LAST_DDL_TIME	-	DATE	-
TIMESTAMP	-	VARCHAR2 (1)	NULL
STATUS	-	VARCHAR2 (5)	NULL
TEMPORARY	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL

**Table C-17 DBA\_TAB\_COLUMNS**

Name	Null?	Type	Value
OWNER	-	VARCHAR2 (30)	-
TABLE_NAME	-	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (30)	-
DATA_TYPE	-	VARCHAR2 (8)	-
DATA_TYPE_MOD	-	VARCHAR2 (1)	NULL
DATA_TYPE_OWNER	-	VARCHAR2 (1)	NULL
DATA_LENGTH	-	NUMBER	-
DATA_PRECISION	-	NUMBER	-
DATA_SCALE	-	NUMBER	-
NULLABLE	-	VARCHAR2 (1)	"Y" or "N"
COLUMN_ID	-	NUMBER	-
DEFAULT_LENGTH	-	NUMBER	0
DATA_DEFAULT	-	VARCHAR2 (1)	NULL
NUM_DISTINCT	-	NUMBER	0
LOW_VALUE	-	NUMBER	0
HIGH_VALUE	-	NUMBER	0
DENSITY	-	NUMBER	0
NUM_NULLS	-	NUMBER	0
NUM_BUCKETS	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL

**Table C-17 (Cont.) DBA\_TAB\_COLUMNS**

Name	Null?	Type	Value
SAMPLE_SIZE	-	NUMBER	0
CHARACTER_SET_NAME	-	VARCHAR2 (1)	NULL
CHAR_COL_DEC_LENGTH	-	NUMBER	0
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
AVG_COL_LEN	-	NUMBER	0

**Table C-18 DBA\_TAB\_COMMENTS**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"
COMMENTS	-	VARCHAR2 (1)	NULL

**Table C-19 DBA\_TABLES**

Name	Null?	Type	Value
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLESPACE_NAME	NOT NULL	CHAR (1)	" "
CLUSTER_NAME	NOT NULL	CHAR (1)	" "
IOT_NAME	NOT NULL	CHAR (1)	" "
PCT_FREE	NOT NULL	NUMBER (10)	0
PCT_USED	NOT NULL	NUMBER (10)	0
INI_TRANS	NOT NULL	NUMBER (10)	0
MAX_TRANS	NOT NULL	NUMBER (10)	0
INITIAL_EXTENT	NOT NULL	NUMBER (10)	0
NEXT_EXTENT	NOT NULL	NUMBER (10)	0
MIN_EXTENTS	NOT NULL	NUMBER (10)	0
MAX_EXTENTS	NOT NULL	NUMBER (10)	0
PCT_INCREASE	NOT NULL	NUMBER (10)	0
FREELISTS	NOT NULL	NUMBER (10)	0
FREELIST_GROUPS	NOT NULL	NUMBER (10)	0
LOGGING	NOT NULL	CHAR (1)	" "
BACKED_UP	NOT NULL	CHAR (1)	" "
NUM_ROWS	NOT NULL	NUMBER (10)	0
BLOCKS	NOT NULL	NUMBER (10)	0

**Table C-19 (Cont.) DBA\_TABLES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
EMPTY_BLOCKS	NOT NULL	NUMBER (10)	0
AVG_SPACE	NOT NULL	NUMBER (10)	0
CHAIN_CNT	NOT NULL	NUMBER (10)	0
AVG_ROW_LEN	NOT NULL	NUMBER (10)	0
AVG_SPACE_FREELIST_BLOCKS	NOT NULL	NUMBER (10)	0
NUM_FREELIST_BLOCKS	NOT NULL	NUMBER (10)	0
DEGREE	NOT NULL	CHAR (1)	" "
INSTANCES	NOT NULL	CHAR (1)	" "
CACHE	NOT NULL	CHAR (1)	" "
TABLE_LOCK	NOT NULL	CHAR (1)	" "
SAMPLE_SIZE	NOT NULL	NUMBER (10)	0
LAST_ANALYZED	-	DATE	NULL
PARTITIONED	NOT NULL	CHAR (1)	" "
IOT_TYPE	NOT NULL	CHAR (1)	" "
TEMPORARY	NOT NULL	CHAR (1)	" "
SECONDARY	NOT NULL	CHAR (1)	" "
NESTED	NOT NULL	CHAR (1)	" "
BUFFER_POOL	NOT NULL	CHAR (1)	" "
ROW_MOVEMENT	NOT NULL	CHAR (1)	" "
GLOBAL_STATS	NOT NULL	CHAR (1)	" "
USER_STATS	NOT NULL	CHAR (1)	" "
DURATION	NOT NULL	CHAR (1)	" "
SKIP_CORRUPT	NOT NULL	CHAR (1)	" "
MONITORING	NOT NULL	CHAR (1)	" "

**Table C-20 DICT\_COLUMNS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	NOT NULL	VARCHAR2 (30)	-
COMMENTS	NOT NULL	CHAR (1)	" "

**Table C-21 DICTIONARY**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COMMENTS	NOT NULL	CHAR (1)	" "

**Table C-22 DUAL**

Name	Null?	Type	Value
DUMMY	NOT NULL	VARCHAR2 (1)	"X"

**Table C-23 TABLE\_PRIVILEGES**

Name	Null?	Type	Value
GRANTEE	NOT NULL	VARCHAR2 (30)	-
OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
GRANTOR	NOT NULL	VARCHAR2 (30)	-
SELECT_PRIV	-	VARCHAR2 (1)	"Y"
INSERT_PRIV	-	VARCHAR2 (1)	"A"
DELETE_PRIV	-	VARCHAR2 (1)	"Y"
UPDATE_PRIV	-	VARCHAR2 (1)	"A"
REFERENCES_PRIV	-	VARCHAR2 (1)	"A"
ALTER_PRIV	-	VARCHAR2 (1)	"Y"
INDEX_PRIV	-	VARCHAR2 (1)	"Y"
CREATED	NOT NULL	DATE	-

**Table C-24 USER\_CATALOG**

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"

**Table C-25 USER\_COL\_COMMENTS**

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	NOT NULL	VARCHAR2 (30)	-
COMMENTS	-	VARCHAR2 (1)	NULL

**Table C-26 USER\_CONS\_COLUMNS**

Name	Null?	Type	Value
OWNER	-	VARCHAR2 (30)	-
CONSTRAINT_NAME	-	VARCHAR2 (30)	-
TABLE_NAME	-	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (4000)	-
POSITION	-	NUMBER	-

**Table C-27 USER\_CONSTRAINTS**

Name	Null?	Type	Value
OWNER	-	VARCHAR2 (30)	-
CONSTRAINT_NAME	-	VARCHAR2 (30)	-
CONSTRAINT_TYPE	-	VARCHAR2 (1)	"R" or "P"
TABLE_NAME	-	VARCHAR2 (30)	-
SEARCH_CONDITION	-	VARCHAR2 (1)	NULL
R_OWNER	-	VARCHAR2 (30)	-
R_CONSTRAINT_NAME	-	VARCHAR2 (30)	-
DELETE_RULE	-	VARCHAR2 (9)	"NOACTION" or " "
STATUS	-	VARCHAR2 (7)	"ENABLED"
DEFERRABLE	-	VARCHAR2 (1)	NULL
DEFERRED	-	VARCHAR2 (1)	NULL
VALIDATED	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
BAD	-	VARCHAR2 (1)	NULL
RELY	-	VARCHAR2 (1)	NULL
LAST_CHANGE	-	DATE	-

**Table C-28 USER\_IND\_COLUMNS**

Name	Null?	Type	Value
INDEX_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (8192 )	-
COLUMN_POSITION	NOT NULL	FLOAT (49)	-
COLUMN_LENGTH	NOT NULL	FLOAT (49)	-
DESCEND	-	VARCHAR (4)	"DESC" or "ASC"

**Table C-29 USER\_INDEXES**

Name	Null?	Type	Value
INDEX_NAME	-	VARCHAR2 (30)	-
INDEX_TYPE	-	VARCHAR2 (1)	NULL
TABLE_OWNER	NOT NULL	VARCHAR2 (30)	-
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (7)	"TABLE" or "CLUSTER"
UNIQUENESS	-	VARCHAR2 (1)	NULL
COMPRESSION	-	VARCHAR2 (1)	NULL

**Table C-29 (Cont.) USER\_INDEXES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
PREFIX_LENGTH	-	NUMBER	0
TABLESPACE_NAME	-	VARCHAR2 (1)	NULL
INI_TRANS	-	NUMBER	0
MAX_TRANS	-	NUMBER	0
INITIAL_EXTENT	-	NUMBER	0
NEXT_EXTENT	-	NUMBER	0
MIN_EXTENTS	-	NUMBER	0
MAX_EXTENTS	-	NUMBER	0
PCT_INCREASE	-	NUMBER	0
PCT_THRESHOLD	-	NUMBER	0
INCLUDE_COLUMN	-	NUMBER	0
FREELISTS	-	NUMBER	0
FREELIST_GROUPS	-	NUMBER	0
PCT_FREE	-	NUMBER	0
LOGGING	-	VARCHAR2 (1)	NULL
BLEVEL	-	NUMBER	0
LEAF_BLOCKS	-	NUMBER	0
DISTINCT_KEYS	-	NUMBER	0
AVG_LEAF_BLOCKS_PER_KEY	-	NUMBER	0
AVG_DATA_BLOCKS_PER_KEY	-	NUMBER	0
CLUSTERING_FACTOR	-	NUMBER	0
STATUS	-	VARCHAR2 (1)	NULL
NUM_ROWS	-	NUMBER	0
SAMPLE_SIZE	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL
DEGREE	-	VARCHAR2 (1)	NULL
INSTANCES	-	VARCHAR2 (1)	NULL
PARTITIONED	-	VARCHAR2 (1)	NULL
TEMPORARY	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL
BUFFER_POOL	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
DURATION	-	VARHCAR2 (1)	NULL
PCT_DIRECT_ACCESS	-	NUMBER	0
ITYP_OWNER	-	VARCHAR2 (1)	NULL
ITYP_NAME	-	VARCHAR2 (1)	NULL



**Table C-29 (Cont.) USER\_INDEXES**

Name	Null?	Type	Value
PARAMETERS	-	VARCHAR2 (1)	NULL
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
DOMIDX_STATUS	-	VARCHAR2 (1)	NULL
DOMIDX_OPSTATUS	-	VARCHAR2 (1)	NULL
FUNCIDX_STATUS	-	VARCHAR2 (1)	NULL

**Table C-30 USER\_OBJECTS**

Name	Null?	Type	Value
OBJECT_NAME	-	VARCHAR2 (30)	-
SUBOBJECT_NAME	-	VARCHAR2 (1)	NULL
OBJECT_ID	-	NUMBER	-
DATA_OBJECT_ID	-	NUMBER	0
OBJECT_TYPE	-	VARCHAR2 (9)	"TABLE" or "VIEW" or "PROCEDURE"
CREATED	-	DATE	-
LAST_DDL_TIME	-	DATE	-
TIMESTAMP	-	VARCHAR2 (1)	NULL
STATUS	-	VARCHAR2 (5)	"VALID"
TEMPORARY	-	VARCHAR2 (1)	NULL
GENERATED	-	VARCHAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL

**Table C-31 USER\_TAB\_COLUMNS**

Name	Null?	Type	Value
TABLE_NAME	-	VARCHAR2 (30)	-
COLUMN_NAME	-	VARCHAR2 (30)	-
DATA_TYPE	-	VARCHAR2 (8)	-
DATA_TYPE_MOD	-	VARCHAR2 (1)	NULL
DATA_TYPE_OWNER	-	VARCHAR2 (1)	NULL
DATA_LENGTH	-	NUMBER	-
DATA_PRECISION	-	NUMBER	-
DATA_SCALE	-	NUMBER	-
NULLABLE	-	VARCHAR2 (1)	"Y" or "N"
COLUMN_ID	-	NUMBER	-
DEFAULT_LENGTH	-	NUMBER	0
DATA_DEFAULT	-	VARCHAR2 (1)	NULL
NUM_DISTINCT	-	NUMBER	0

**Table C-31 (Cont.) USER\_TAB\_COLUMNS**

Name	Null?	Type	Value
LOW_VALUE	-	NUMBER	0
HIGH_VALUE	-	NUMBER	0
DENSITY	-	NUMBER	0
NUM_NULLS	-	NUMBER	0
NUM_BUCKETS	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL
SAMPLE_SIZE	-	NUMBER	0
CHARACTER_SET_NAME	-	VARCHAR2 (1)	NULL
CHAR_COL_DECL_LENGTH	-	NUMBER	0
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
AVG_COL_LEN	-	NUMBER	0

**Table C-32 USER\_TAB\_COMMENTS**

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLE_TYPE	-	VARCHAR2 (5)	"TABLE" or "VIEW"
COMMENTS	-	VARCHAR2 (1)	NULL

**Table C-33 USER\_TABLES**

Name	Null?	Type	Value
TABLE_NAME	NOT NULL	VARCHAR2 (30)	-
TABLESPACE_NAME	-	VARCHAR2 (1)	NULL
CLUSTER_NAME	-	VARCHAR2 (1)	NULL
IOT_NAME	-	VARCHAR2 (1)	NULL
PCT_FREE	-	NUMBER	0
PCT_USED	-	NUMBER	0
INI_TRANS	-	NUMBER	0
MAX_TRANS	-	NUMBER	0
INITIAL_EXTENT	-	NUMBER	0
NEXT_EXTENT	-	NUMBER	0
MIN_EXTENTS	-	NUMBER	0
MAX_EXTENTS	-	NUMBER	0
PCT_INCREASE	-	NUMBER	0
FREELISTS	-	NUMBER	0
FREELIST_GROUPS	-	NUMBER	0

**Table C-33 (Cont.) USER\_TABLES**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
LOGGING	-	VARCHAR2 (1)	NULL
BACKED_UP	-	VARCHAR2 (1)	NULL
NUM_ROWS	-	NUMBER	0
BLOCKS	-	NUMBER	0
EMPTY_BLOCKS	-	NUMBER	0
AVG_SPACE	-	NUMBER	0
CHAIN_CNT	-	NUMBER	0
AVG_ROW_LEN	-	NUMBER	0
AVG_SPACE_FREELIST_BLOCKS	-	NUMBER	0
NUM_FREELIST_BLOCKS	-	NUMBER	0
DEGREE	-	VARCHAR2 (1)	NULL
INSTANCES	-	VARCHAR2 (1)	NULL
CACHE	-	VARCHAR2 (1)	NULL
TABLE_LOCK	-	VARCHAR2 (1)	NULL
SAMPLE_SIZE	-	NUMBER	0
LAST_ANALYZED	-	DATE	NULL
PARTITIONED	-	VARCHAR2 (1)	NULL
IOT_TYPE	-	VARCHAR2 (1)	NULL
TEMPORARY	-	VARHCAR2 (1)	NULL
SECONDARY	-	VARCHAR2 (1)	NULL
NESTED	-	VARCHAR2 (1)	NULL
BUFFER_POOL	-	VARCHAR2 (1)	NULL
ROW_MOVEMENT	-	VARCHAR2 (1)	NULL
GLOBAL_STATS	-	VARCHAR2 (1)	NULL
USER_STATS	-	VARCHAR2 (1)	NULL
DURATION	-	VARCHAR2 (1)	NULL
SKIP_CORRUPT	-	VARCHAR2 (1)	NULL
MONITORING	-	VARCHAR2 (1)	NULL

**Table C-34 USER\_USERS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
USERNAME	NOT NULL	VARCHAR2 (30)	-
USER_ID	NOT NULL	NUMBER (5)	-
ACCOUNT_STATUS	-	VARCHAR2 (4)	"OPEN"
LOCK_DATE	-	DATE	NULL
EXPIRY_DATE	-	DATE	NULL
DEFAULT_TABLESPACE	-	VARCHAR2 (1)	NULL

**Table C–34 (Cont.) USER\_USERS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
TEMPORARY_TABLESPACE	-	VARCHAR2 (1)	NULL
CREATED	-	DATE	SYSDATE
INITIAL_RSRC_CONSUMER_GROUP	-	VARCHAR2 (1)	NULL
EXTERNAL_NAME	-	VARCHAR2 (1)	NULL

**Table C–35 USER\_VIEWS**

<b>Name</b>	<b>Null?</b>	<b>Type</b>	<b>Value</b>
VIEW_NAME	NOT NULL	VARCHAR2 (30)	-
TEXT_LENGTH	-	NUMBER	0
TEXT	-	VARCHAR2 (255)	-
TYPE_TEXT_LENGTH	-	NUMBER	0
TYPE_TEXT	-	VARCHAR2 (1)	NULL
OID_TEXT_LENGTH	-	NUMBER	0
OID_TEXT	-	VARCHAR2 (1)	NULL
VIEW_TYPE_OWNER	-	VARCHAR2 (1)	NULL
VIEW_TYPE	-	VARCHAR2 (1)	NULL

---

# Initialization Parameters

The Oracle database initialization parameters in the `init.ora` file are distinct from gateway initialization parameters. Set the gateway parameters in the initialization parameter file using an agent-specific mechanism, or set them in the Oracle data dictionary using the `DBMS_HS` package. The gateway initialization parameter file must be available when the gateway is started.

This appendix contains a list of the gateway initialization parameters that can be set for each gateway and their description. It also describes the initialization parameter file syntax. It includes the following sections:

- [Initialization Parameter File Syntax](#)
- [Oracle Database Gateway for Sybase Initialization Parameters](#)
- [Initialization Parameter Descriptions](#)

## Initialization Parameter File Syntax

The syntax for the initialization parameter file is as follows:

1. The file is a sequence of commands.
2. Each command should start on a separate line.
3. End of line is considered a command terminator (unless escaped with a backslash).
4. If there is a syntax error in an initialization parameter file, none of the settings take effect.
5. Set the parameter values as follows:

```
[SET] [PRIVATE] parameter=value
```

Where:

*parameter* is an initialization parameter name. It is a string of characters starting with a letter and consisting of letters, digits and underscores. Initialization parameter names are case sensitive.

*value* is the initialization parameter value. It is case-sensitive. An initialization parameter value is either:

- a. A string of characters that does not contain any backslashes, white space or double quotation marks ("")
- b. A quoted string beginning with a double quotation mark and ending with a double quotation mark. The following can be used inside a quoted string:

- \* backslash (\) is the escape character
- \* \n inserts a new line
- \* \t inserts a tab
- \* \" inserts a double quotation mark
- \* \\ inserts a backslash

A backslash at the end of the line continues the string on the next line. If a backslash precedes any other character then the backslash is ignored.

For example, to enable tracing for an agent, set the HS\_FDS\_TRACE\_LEVEL initialization parameter as follows:

```
HS_FDS_TRACE_LEVEL=ON
```

SET and PRIVATE are optional keywords. You cannot use either as an initialization parameter name. Most parameters are needed only as initialization parameters, so you usually do not need to use the SET or PRIVATE keywords. If you do not specify either SET or PRIVATE, the parameter is used only as an initialization parameter for the agent.

SET specifies that, in addition to being used as an initialization parameter, the parameter value is set as an environment variable for the agent process. Use SET for parameter values that the drivers or non-Oracle system need as environment variables.

PRIVATE specifies that the initialization parameter should be private to the agent and should not be uploaded to the Oracle database. Most initialization parameters should not be private. If, however, you are storing sensitive information like a password in the initialization parameter file, then you may not want it uploaded to the server because the initialization parameters and values are not encrypted when uploaded. Making the initialization parameters private prevents the upload from happening and they do not appear in dynamic performance views. Use PRIVATE for the initialization parameters only if the parameter value includes sensitive information such as a username or password.

SET PRIVATE specifies that the parameter value is set as an environment variable for the agent process and is also private (not transferred to the Oracle database, not appearing in dynamic performance views or graphical user interfaces).

## Oracle Database Gateway for Sybase Initialization Parameters

This section lists all the initialization file parameters that can be set for the Oracle Database Gateway for Sybase. They are as follows:

- [HS\\_CALL\\_NAME](#)
- [HS\\_DB\\_DOMAIN](#)
- [HS\\_DB\\_INTERNAL\\_NAME](#)
- [HS\\_DB\\_NAME](#)
- [HS\\_DESCRIBE\\_CACHE\\_HWM](#)
- [HS\\_LANGUAGE](#)
- [HS\\_LONG\\_PIECE\\_TRANSFER\\_SIZE](#)
- [HS\\_OPEN\\_CURSORS](#)
- [HS\\_RPC\\_FETCH\\_REBLOCKING](#)

- [HS\\_RPC\\_FETCH\\_SIZE](#)
- [HS\\_TIME\\_ZONE](#)
- [HS\\_TRANSACTION\\_MODEL](#)
- [IFILE](#)
- [HS\\_FDS\\_CONNECT\\_INFO](#)
- [HS\\_FDS\\_DEFAULT\\_OWNER](#)
- [HS\\_FDS\\_PROC\\_IS\\_FUNC](#)
- [HS\\_FDS\\_RECOVERY\\_ACCOUNT](#)
- [HS\\_FDS\\_RECOVERY\\_PWD](#)
- [HS\\_FDS\\_RESULTSET\\_SUPPORT](#)
- [HS\\_FDS\\_TRACE\\_LEVEL](#)
- [HS\\_FDS\\_TRANSACTION\\_LOG](#)
- [HS\\_FDS\\_FETCH\\_ROWS](#)

## Initialization Parameter Description

The following sections describe all the initialization file parameters that can be set for gateways.

### HS\_CALL\_NAME

Property	Description
Default value	None
Range of values	Not applicable

Specifies the remote functions that can be referenced in SQL statements. The value is a list of remote functions and their owners, separated by semicolons, in the following format:

*owner\_name.function\_name*

For example:

`owner1.A1;owner2.A2;owner3.A3`

If an owner name is not specified for a remote function, the default owner name becomes the user name used to connect to the remote database (specified when the Heterogeneous Services database link is created or taken from user session if not specified in the DB link).

The entries for the owner names and the function names are case-sensitive.

### HS\_DB\_DOMAIN

Property	Description
Default value	WORLD

Property	Description
Range of values	1 to 199 characters

Specifies a unique network sub-address for a non-Oracle system. The HS\_DB\_DOMAIN initialization parameter is similar to the DB\_DOMAIN initialization parameter, described in the *Oracle Database Reference*. The HS\_DB\_DOMAIN initialization parameter is required if you use the Oracle Names server. The HS\_DB\_NAME and HS\_DB\_DOMAIN initialization parameters define the global name of the non-Oracle system.

---

---

**Note:** The HS\_DB\_NAME and HS\_DB\_DOMAIN initialization parameters must combine to form a unique address in a cooperative server environment.

---

---

## HS\_DB\_INTERNAL\_NAME

Property	Description
Default value	01010101
Range of values	1 to 16 hexadecimal characters

Specifies a unique hexadecimal number identifying the instance to which the Heterogeneous Services agent is connected. This parameter's value is used as part of a transaction ID when global name services are activated. Specifying a nonunique number can cause problems when two-phase commit recovery actions are necessary for a transaction.

## HS\_DB\_NAME

Property	Description
Default value	HO
Range of values	1 to 8 characters

Specifies a unique alphanumeric name for the data store given to the non-Oracle system. This name identifies the non-Oracle system within the cooperative server environment. The HS\_DB\_NAME and HS\_DB\_DOMAIN initialization parameters define the global name of the non-Oracle system.

## HS\_DESCRIBE\_CACHE\_HWM

Property	Description
Default value	100
Range of values	1 to 4000

Specifies the maximum number of entries in the describe cache used by Heterogeneous Services. This limit is known as the describe cache high water mark. The cache contains descriptions of the mapped tables that Heterogeneous Services reuses so that it does not have to re-access the non-Oracle data store.



If you are accessing many mapped tables, increase the high water mark to improve performance. Increasing the high water mark improves performance at the cost of memory usage.

## HS\_LANGUAGE

Property	Description
Default value	System-specific
Range of values	Any valid language name (up to 255 characters)

Provides Heterogeneous Services with character set, language, and territory information of the non-Oracle data source. The value must use the following format:

*language[\_territory.character\_set]*

---

**Note:** The globalization support initialization parameters affect error messages, the data for the SQL Service, and parameters in distributed external procedures.

---

### Character Sets

Ideally, the character sets of the Oracle database server and the non-Oracle data source are the same. If they are not the same, Heterogeneous Services attempts to translate the character set of the non-Oracle data source to the Oracle database character set, and back again. The translation can degrade performance. In some cases, Heterogeneous Services cannot translate a character from one character set to another.

---

**Note:** The specified character set must be a superset of the operating system character set on the platform where the agent is installed.

---

### Language

The language component of the HS\_LANGUAGE initialization parameter determines:

- Day and month names of dates
- AD, BC, PM, and AM symbols for date and time
- Default sorting mechanism

Note that Oracle does not determine the language for error messages for the generic Heterogeneous Services messages (ORA-25000 through ORA-28000). These are controlled by the session settings in the Oracle database server.

---

**Note:** Use the HS\_NLS\_DATE\_LANGUAGE initialization parameter to set the day and month names, and the AD, BC, PM, and AM symbols for dates and time independently from the language.

---

### Territory

The territory clause specifies the conventions for day and week numbering, default date format, decimal character and group separator, and ISO and local currency

symbols. Note that the level of globalization support between the Oracle database server and the non-Oracle data source depends on how the gateway is implemented.

## HS\_LONG\_PIECE\_TRANSFER\_SIZE

Property	Description
Default value	64 KB
Range of values	Any value up to 2 GB

Sets the size of the piece of `LONG` data being transferred. A smaller piece size means less memory requirement, but more round-trips to fetch all the data. A larger piece size means fewer round-trips, but more of a memory requirement to store the intermediate pieces internally. Thus, the initialization parameter can be used to tune a system for the best performance, with the best trade-off between round-trips and memory requirements, and network latency.

## HS\_OPEN\_CURSORS

Property	Description
Default value	50
Range of values	1 to the value of Oracle's <code>OPEN_CURSORS</code> initialization parameter

Defines the maximum number of cursors that can be open on one connection to a non-Oracle system instance.

The value never exceeds the number of open cursors in the Oracle database server. Therefore, setting the same value as the `OPEN_CURSORS` initialization parameter in the Oracle database server is recommended.

## HS\_RPC\_FETCH\_REBLOCKING

Property	Description
Default value	ON
Range of values	OFF or ON

Controls whether Heterogeneous Services attempts to optimize performance of data transfer between the Oracle database server and the Heterogeneous Services agent connected to the non-Oracle data store.

The following values are possible:

- `OFF` disables reblocking of fetched data so that data is immediately sent from agent to server.
- `ON` enables reblocking, which means that data fetched from the non-Oracle system is buffered in the agent and is not sent to the Oracle database server until the amount of fetched data is equal or higher than the value of `HS_RPC_FETCH_SIZE` initialization parameter. However, any buffered data is returned immediately when a fetch indicates that no more data exists or when the non-Oracle system reports an error.

## HS\_RPC\_FETCH\_SIZE

Property	Description
Default value	50000
Range of values	1 to 10000000

Tunes internal data buffering to optimize the data transfer rate between the server and the agent process.

Increasing the value can reduce the number of network round-trips needed to transfer a given amount of data, but also tends to increase data bandwidth and to reduce response time or *latency* as measured between issuing a query and completion of all fetches for the query. Nevertheless, increasing the fetch size can increase latency for the initial fetch results of a query, because the first fetch results are not transmitted until additional data is available.

## HS\_TIME\_ZONE

Property	Description
Default value for '[+ -]hh:mm'	Derived from the NLS_TERRITORY initialization parameter
Range of values for '[+ -]hh:mm'	Any valid datetime format mask

Specifies the default local time zone displacement for the current SQL session. The format mask, [+|-]hh:mm, is specified to indicate the hours and minutes before or after UTC (Coordinated Universal Time—formerly Greenwich Mean Time). For example:

```
HS_TIME_ZONE = [+ | -] hh:mm
```

## HS\_TRANSACTION\_MODEL

Property	Description
Default Value	COMMIT_CONFIRM
Range of Values	COMMIT_CONFIRM, READ_ONLY, SINGLE_SITE

Specifies the type of transaction model that is used when the non-Oracle database is updated by a transaction.

The following values are possible:

- COMMIT\_CONFIRM provides read and write access to the non-Oracle database and allows the gateway to be part of a distributed update. To use the commit-confirm model, the following items must be created in the non-Oracle database:
  - Transaction log table. The default table name is HS\_TRANSACTION\_LOG. A different name can be set using the HS\_FDS\_TRANSACTION\_LOG parameter. The transaction log table must be granted SELECT, DELETE, and INSERT privileges set to public.

- Recovery account. The account name is assigned with the HS\_FDS\_RECOVERY\_ACCOUNT parameter.
- Recovery account password. The password is assigned with the HS\_FDS\_RECOVERY\_PWD parameter.
- READ\_ONLY provides read access to the non-Oracle database.
- SINGLE\_SITE provides read and write access to the non-Oracle database. However, the gateway cannot participate in distributed updates.

## IFILE

Property	Description
Default value	None
Range of values	Valid parameter file names

Use the IFILE initialization parameter to embed another initialization file within the current initialization file. The value should be an absolute path and should not contain environment variables. The three levels of nesting limit does not apply.

**See Also:** *Oracle Database Reference*

## HS\_FDS\_CONNECT\_INFO

Property	Description
Default Value	None
Range of Values	Not applicable

HS\_FDS\_CONNECT\_INFO which describes the connection to the non-Oracle system.

The default initialization parameter file already has an entry for this parameter. The syntax for HS\_FDS\_CONNECT\_INFO for the gateway is as follows:

`HS_FDS_CONNECT_INFO=host_name:port_number/database_name`

where, *host\_name* is the host name or IP address of the machine hosting the Sybase database, *port\_number* is the port number of the Sybase database server, and *database\_name* is the Sybase database name.

## HS\_FDS\_DEFAULT\_OWNER

Property	Description
Default Value	None
Range of Values	Not applicable

The name of the table owner that is used for the non-Oracle database tables if an owner is not specified in the SQL statements.

---

**Note:** If this parameter is not specified and the owner is not explicitly specified in the SQL statement, then the user name of the Oracle user or the user name specified when creating the database link is used.

---

## HS\_FDS\_PROC\_IS\_FUNC

Property	Description
Default Value	FALSE
Range of Values	TRUE, FALSE

Enables return values from functions. By default, all stored procedures and functions do not return a return value to the user.

---

**Note:** If you set this initialization parameter, you must change the syntax of the procedure execute statement for all existing stored procedures to handle return values.

---

## HS\_FDS\_RECOVERY\_ACCOUNT

Property	Description
Default Value	RECOVER
Range of values	Any valid user ID

Specifies the name of the recovery account used for the commit-confirm transaction model. An account with user name and password must be set up at the non-Oracle system. For more information about the commit-confirm model, see the HS\_TRANSACTION\_MODEL parameter.

The name of the recovery account is case-sensitive.

## HS\_FDS\_RECOVERY\_PWD

Property	Description
Default Value	RECOVER
Range of values	Any valid password

Specifies the password of the recovery account used for the commit-confirm transaction model set up at the non-Oracle system. For more information about the commit-confirm model, see the HS\_TRANSACTION\_MODEL parameter.

The name of the password of the recovery account is case-sensitive.

## HS\_FDS\_RESULTSET\_SUPPORT

Property	Description
Default Value	FALSE
Range of Values	TRUE, FALSE

Enables result sets to be returned from stored procedures. By default, all stored procedures do not return a result set to the user.

---

**Note:** If you set this initialization parameter, you must do the following:

- Change the syntax of the procedure execute statement for all existing stored procedures, to handle result sets
  - Work in the sequential mode of Heterogeneous Services
- 

## HS\_FDS\_TRACE\_LEVEL

Property	Description
Default Value	OFF
Range of values	OFF, ON, DEBUG

Specifies whether error tracing is turned on or off for gateway connectivity.

The following values are valid:

- OFF disables the tracing of error messages.
- ON enables the tracing of error messages that occur when you encounter problems. The results are written by default to a gateway log file in LOG directory where the gateway is installed.
- DEBUG enables the tracing of detailed error messages that can be used for debugging.

## HS\_FDS\_TRANSACTION\_LOG

Property	Description
Default Value	HS_TRANSACTION_LOG
Range of Values	Any valid table name

Specifies the name of the table created in the non-Oracle system for logging transactions. For more information about the transaction model, see the HS\_TRANSACTION\_MODEL parameter.

## HS\_FDS\_FETCH\_ROWS

Property	Description
Default Value	100
Range of Values	Any integer between 1 and 1000
Syntax	HS_FDS_FETCH_ROWS= <i>num</i>

HS\_FDS\_FETCH\_ROWS specifies the fetch array size. This is the number of rows to be fetched from the non-Oracle database and to return to Oracle database at one time. This parameter will be affected by the HS\_RPC\_FETCH\_SIZE and HS\_RPC\_FETCH\_REBLOCKING parameters.





---

---

# Index

## A

---

ALTER statement, B-1  
Arithmetic operators, B-2

## B

---

BINARY data type, A-1  
BIT data type, A-1

## C

---

Case rules, 2-7  
Case studies, 3-1  
Chained mode, 2-6  
CHAR data type, A-1  
character sets  
    Heterogeneous Services, D-5  
COMMIT  
    restrictions, 2-10  
Commit point site, 2-10  
Comparison operators, B-3  
CONCAT operator, 2-14  
CONNECT BY clause, 2-11  
COPY command, 2-12  
CREATE statement, B-1  
Cursor loops  
    restrictions, 2-10

## D

---

Data definition language, B-1  
Data dictionary  
    views, C-2  
Data type  
    BINARY, A-1  
    BIT, A-1  
    CHAR, A-1  
    conversion, 2-8  
    DATE, A-1  
    DATETIME, A-1  
    DECIMAL, A-1  
    DOUBLE PRECISION, A-1  
    FLOAT, A-1  
        FLOAT data type, 2-13  
    IMAGE, A-1  
    INTEGER, A-1

LONG, A-1  
LONG RAW, A-1  
MONEY, A-1  
NCHAR, A-1  
NUMBER, A-1  
NUMERIC, A-1  
NVARCHAR, A-1  
RAW, A-1  
REAL, A-1  
SMALL DATETIME, A-1  
SMALL MONEY, A-1  
SMALLINT, A-1  
TEXT, A-1  
TINYINT, A-1  
VARBINARY, 2-13, A-2  
VARCHAR, A-2  
VARCHAR2, A-2  
DATE data type, A-1  
DATETIME data type, A-1  
DDL statement, 2-11  
DECIMAL data type, A-1  
DELETE statement, 3-5, B-1, B-2  
Demo build SQL script, 3-2  
Demonstration files, 3-2  
Demonstration tables, 3-2  
Demonstration tables build SQL script, 3-2  
describe cache high water mark  
    definition, D-4  
DOUBLE PRECISION data type, A-1  
DROP statement, B-1

## E

---

Encrypted format login, 2-13  
Error messages  
    error tracing, D-10  
Errors  
    ORA-02070, 2-10  
Executing Stored Procedures, 3-6

## F

---

fetch array size, with HS\_FDS\_FETCH\_ROWS, D-11  
FLOAT data type, A-1  
Functions in SQL, 2-2

## G

---

### Gateway

- case studies, 3-1
- data dictionary tables, C-1
- pass-through feature, 2-1
- supported functions, B-1
- supported SQL syntax, B-1

### globalization support

- Heterogeneous Services, D-5

### GRANT statement, B-1

### Group functions, B-3

## H

---

### Heterogeneous Services

- defining maximum number of open cursors, D-6
- optimizing data transfer, D-6
- setting global name, D-4
- specifying cache high water mark, D-4
- tuning internal data buffering, D-7
- tuning LONG data transfer, D-6

### Hexadecimal notation, 2-8

### HS\_CALL\_NAME initialization parameter, D-3

### HS\_DB\_NAME initialization parameter, D-4

### HS\_DESCRIBE\_CACHE\_HWM initialization parameter, D-4

### HS\_FDS\_CONNECT\_INFO, D-8

### HS\_FDS\_DEFAULT\_OWNER initialization parameter, D-8

### HS\_FDS\_FETCH\_ROWS parameter, D-11

### HS\_FDS\_PROC\_IS\_FUNC initialization parameter, D-9

### HS\_FDS\_RESULTSET\_SUPPORT initialization parameter, D-10

### HS\_FDS\_TRACE\_LEVEL initialization parameter, D-10

- enabling agent tracing, D-2

### HS\_FDS\_TRANSACTION\_LOG initialization parameter, D-10

### HS\_LANGUAGE initialization parameter, D-5

### HS\_LONG\_PIECE\_TRANSFER\_SIZE initialization parameter, D-6

### HS\_OPEN\_CURSORS initialization parameter, D-6

### HS\_RPC\_FETCH\_REBLOCKING initialization parameter, D-6

### HS\_RPC\_FETCH\_SIZE initialization parameter, D-7

### HS\_TIME\_ZONE initialization parameter, D-7

## I

---

### IFILE initialization parameter, D-8

### IMAGE data type, A-1

### Initialization parameter file

- customizing, D-1

### INSERT statement, 2-12, 3-5, B-1, B-2

### INTEGER data type, A-1

## K

---

### Known restrictions, 2-9

## L

---

### Locking, database, 2-9

### LONG data type, A-1

### LONG RAW data type, A-1

## M

---

### MONEY data type, A-1

## N

---

### NCHAR data type, A-1

### NULL values, 2-6

### NUMBER data type, A-1

### NUMERIC data type, A-1

### NVARCHAR data type, A-1

### NVL function, 3-5

## O

---

### Objects, naming rules, 2-7

### ORA-02070, 2-10

## P

---

### parameters

#### gateway initialization file

##### HS\_FDS\_FETCH\_ROWS, D-11

### Passing commands to database, 2-11

### Pass-Through Feature, 3-6

### Pass-through feature, 2-11

### PL/SQL, 2-14

## R

---

### RAW data type, A-1

### REAL data type, A-1

### remote functions

#### referenced in SQL statements, D-3

### ROLLBACK

#### restrictions, 2-10

### ROWID, 2-11

## S

---

### savepoint support, 2-10

### SELECT statement, 3-6, B-1, C-1

### SMALL DATETIME data type, A-1

### SMALLINT data type, A-1

### Stored procedures, 2-10, 2-14

#### running in chained mode, 2-6

### Stored procedures in SQL Server, 2-2

### String functions, B-3

### SUM function, 3-5

## T

---

### TEXT data type, A-1

### TINYINT data type, A-1

### Transaction modes, 2-6

### transactional capability, 2-10

transactional integrity, 2-10  
TRUNCATE statement, B-1  
Two-phase commit, 2-10

## **U**

---

UPDATE statement, 3-5, 3-6, B-2

## **V**

---

VARBINARY data type, 2-13, A-2  
VARCHAR data type, A-2  
VARCHAR2 data type, A-2

## **W**

---

WHERE CURRENT OF clause, 2-11

