Oracle® Database Gateway for Sybase
User's Guide

18c
E84093-01
February 2018

ORACLE®

Oracle Database Gateway for Sybase User's Guide, 18c

E84093-01

Contents

3   Case Studies

A   Data Type Conversion

B   Supported SQL Syntax and Functions

C   Data Dictionary

D Initialization Parameters

Index

List of Tables

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

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at http://www.oracle.com/pls/topic/lookup?ctx=acc&id=docacc.

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=info or visit http://www.oracle.com/pls/topic/lookup?ctx=acc&id=trs if you are hearing impaired.

Related Documents

For more information, see the following documents:

- *Oracle Database New Features Guide*
- *Oracle Call Interface Programmer's Guide*
- *Oracle Database Administrator's Guide*
- *Oracle Database Development 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 User'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

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction to the Oracle Database Gateway for Sybase

Oracle Database Gateways provide the ability to transparently access data residing in a non-Oracle system from an Oracle environment. The following sections 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, and how Oracle Database Gateways fit in the picture, refer to the *Oracle Database Heterogeneous Connectivity User's Guide* first.

Topics:

- Overview of Oracle Database Gateways
- About Heterogeneous Services Technology
- Oracle Database Gateways

Overview of Oracle Database Gateways

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 that 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.

About 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 that 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 User's Guide* for more information about Heterogeneous Services.

About Oracle Database Gateway for Sybase

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 can be installed on any machine independent of the Oracle or non-Oracle database. It 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.

2

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.

Topics:

- Remote Insert Rowsource
- Using the Pass-Through Feature
- Executing Stored Procedures and Functions
- CHAR Semantics
- Multi-byte Character Sets Ratio Suppression
- IPv6 Support
- Gateway Session IDLE Timeout
- Database Compatibility Issues for Sybase
- Known Restrictions
- Known Problems

Remote Insert Rowsource

A remote insert rowsource feature allows remote insert requiring local Oracle data to work through the Oracle database and Oracle Database Gateway. This functionality requires that the Oracle database and the Oracle Database Gateway to be version 12.2 or later.

By Oracle Database design, some distributed statement must be executed at the database link site. But in certain circumstances, there is data needed to execute these queries that must be fetched from the originating Oracle Database. Under homogeneous connections, the remote Oracle database would call back the source Oracle database for such data. But in heterogeneous connections, this is not viable, because this means that the Foreign Data Store would have to query call back functions, or data, that can only be provided by the Oracle instance that issued the query. In general, these kinds of statements are not something that can be supported through the Oracle Database Gateway.

The following categories of SQL statements results in a callback:

- Any DML with a sub-select, which refers to a table in Oracle database.
- Any `DELETE`, `INSERT`, `UPDATE` or `"SELECT... FOR UPDATE..."` SQL statement containing SQL functions or statements that needs to be executed at the originating Oracle database.

  These SQL functions include `USER`, `USERENV`, and `SYSDATE`; and involve the selection of data from the originating Oracle database.

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

An example of a remote `INSERT` statement that can work through the remote insert rowsource feature is as follows:

```
INSERT INTO gateway_table@gateway_link select * from local_table;
```

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, of *Oracle Database Heterogeneous Connectivity User'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.

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

The gateway supports stored procedures in three mutually exclusive modes:

- Normal mode: Have access to IN/OUT arguments only
- Return value mode: Have a return value for all stored procedures
- Resultset mode: Out values are available as last result set

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:**
>
> 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:**
>
> 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 User'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 will be 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 */

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

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

```
create or replace package rcpackage is
  type RCTYPE is ref cursor;
end rcpackage;
/

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;
                        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;
                    /
```

CHAR Semantics

This feature allows the gateway to optionally run in CHAR Semantics mode. Rather than always describing Sybase CHAR columns as CHAR(n BYTE), this feature describes them as CHAR(n CHAR) and VARCHAR(n CHAR). The concept is similar to Oracle database CHAR Semantics. You need to specify HS_NLS_LENGTH_SEMANTICS=CHAR gateway parameter to activate this option. Refer to Initialization Parameters for more detail.

Multi-byte Character Sets Ratio Suppression

This feature optionally suppresses the ratio expansion from Sybase database to Oracle database involving multi-byte character set. By default, Oracle gateways assume the worst ratio to prevent data being truncated or insufficient buffer size situation. However, if you have specific knowledge of your Sybase database and do not want the expansion to occur, you can specify HS_KEEP_REMOTE_COLUMN_SIZE parameter to suppress the expansion. Refer to Initialization Parameters for more detail.

IPv6 Support

Besides full IPv6 support between Oracle databases and the gateway, IPv6 is also supported between this gateway and Sybase database. Refer to the `HS_FDS_CONNECT_INFO` parameter in Initialization Parameters for more detail.

Gateway Session IDLE Timeout

You can optionally choose to terminate long idle gateway sessions automatically with the gateway parameter `HS_IDLE_TIMEOUT`. Specifically, when a gateway session is idle for more than the specified time limit, the gateway session is terminated with any pending update rolled back.

Database Compatibility Issues for Sybase

Sybase and Oracle databases function differently in some areas, causing compatibility problems such as follows:

- Chained Mode
- Column Definitions
- Naming Rules
- Data Types
- Queries
- Locking
- Sybase Identifiers Length Limit

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 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 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:**
>
> 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.

When comparing 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 a 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.

Sybase Identifiers Length Limit

By default, the gateway will always quote identifiers. However, certain Sybase releases have a limit of 30 characters for identifiers such as table or column names and quotes are considered part of the names when checking against this limit. Therefore, when quotes are used, you can only specify 28 characters. In order to support the maximum length limit in those Sybase releases, you need to specify `HS_FDS_QUOTE_IDENTIFIER=FALSE` in your gateway initialization parameter file. Setting this initialization parameter will cause the gateway to send identifiers without quotes. However, it has the side effect of precluding the use of identifiers that contain dots (.) or spaces, and the identifiers will follow the case sensitivity of the Sybase database being used.

Known Restrictions

If you encounter incompatibility problems not listed in this section or in "Known Problems", 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 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.

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 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

Restrictions on SQL syntax are listed as follows:

- WHERE CURRENT OF Clause
- CONNECT BY Clause
- ROWID
- Subqueries in INSERT Statement
- EXPLAIN PLAN Statement

> ✎ **See Also:**
>
> 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.

SQL*Plus COPY Command with Lowercase Table Names

You need to use double quotes to wrap around lowercase 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 office for information about accessing the list.

The known problems are as follows:

- 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

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 + numbernumber + datedate - numberdate1 - 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 recommends using Pro*C or Oracle Call Interface to access such data in a Sybase database.

- `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:**
>
> 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

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

3

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.

Topics:

- Case Descriptions
- Distribution Media Contents
- Accessing the 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)

Distribution 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

Accessing the Demonstration Files

After a successful gateway installation, use the demonstration files stored in the directory `$ORACLE_HOME/dg4sybs/demo`, where `$ORACLE_HOME` is the directory under which the gateway is installed. The directory contains the following demonstration files:

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

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`) that 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 that 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
```

A

Data Type Conversion

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

Table A-1   Data Type Mapping and Restrictions

Sybase	Oracle	Comment	If Oracle uses large varchar (32k)
`BIGINT`	`NUMBER(19)`		
`BINARY`	`RAW`	-	
`BIT`	`NUMBER(3)`	-	
`CHAR`	`CHAR`	-	
`DATE`	`DATE`		
`DATETIME`	`DATE`	-	
`DECIMAL`	`NUMBER(p[,s])`	-	
`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])`	-	
`NUMERIC`	`NUMBER(p[,s])`	-	
`REAL`	`FLOAT(24)`	-	
`SMALLDATETIME`	`DATE`	-	
`SMALLINT`	`NUMBER(5)`	`NUMBER` range is `-32,767` to `32,767`	
`SMALLMONEY`	`NUMBER(p[,s])`	-	
`SYSNAME`	`VARCHAR2`		
`TEXT`	`LONG`	-	
`TIME`	`CHAR(15)`		
`TIMESTAMP`	`RAW`		
`TINYINT`	`NUMBER(3)`	-	
`UNICHAR`	`NCHAR`	-	
`UNSIGNED BIGINT`	`NUMBER(20)`		
`UNSIGNED INT`	`NUMBER(10)`		
`UNSIGNED SMALLINT`	`NUMBER(5)`		
`UNITEXT`	`LONG`	if Oracle DB Character Set = Unicode. Otherwise, it is not supported	
`UNIVARCHAR`	`NVARCHAR`	-	
`VARBINARY`	`RAW`	-	

Table A-1   (Cont.) Data Type Mapping and Restrictions

Sybase	Oracle	Comment	If Oracle uses large varchar (32k)
VARCHAR	VARCHAR2	N < = 4000	N <= 32767

B

Supported SQL Syntax and Functions

The following topics describe supported SQL Syntax and Functions:

- 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" 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" 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" 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.

Topics:

- Functions Not Supported by Sybase
- Functions Supported by Sybase

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
||	+ (*expression1* + *expression2*)
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
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

C

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.

Topics:

- Sybase System Tables
- Accessing the Gateway Data Dictionary
- Direct Queries to Sybase Tables
- Supported Views and Tables
- Data Dictionary Mapping
- Gateway Data Dictionary Descriptions

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_USERS	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 returns 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
USER_CONSTRAINTS	sysconstraints + sysusers + sysobjects
USER_IND_COLUMNS	sysusers + sysindexes + syscolumns
USER_INDEXES	sysusers + sysindexes + sysobjects
USER_OBJECTS	sysusers + sysobjects + sysindexes

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

View Name	Sybase System Table Name
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.

ALL_CATALOG

Table C-2   ALL_CATALOG

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"

ALL_COL_COMMENTS

Table C-3   ALL_COL_COMMENTS

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-

Table C-3   (Cont.) ALL_COL_COMMENTS

Name	Type	Value
COLUMN_NAME	VARCHAR2(30)	-
COMMENTS	CHAR(1)	" "

ALL_CONS_COLUMNS

Table C-4   ALL_CONS_COLUMNS

Name	Type	Value
OWNER	VARCHAR2(30)	-
CONSTRAINT_NAME	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(8192)	-
POSITION	FLOAT(49)	-

ALL_CONSTRAINTS

Table C-5   ALL_CONSTRAINTS

Name	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	-

ALL_IND_COLUMNS

Table C-6   ALL_IND_COLUMNS

Name	Type	Value
INDEX_OWNER	VARCHAR2(30)	-
INDEX_NAME	VARCHAR2(30)	-
TABLE_OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(8192)	-
COLUMN_POSITION	FLOAT(49)	-
COLUMN_LENGTH	FLOAT(49)	-
DESCEND	VARCHAR(4)	"DESC" or "ASC"

ALL_INDEXES

Table C-7   ALL_INDEXES

Name	Type	Value
OWNER	VARCHAR2(30)	-
INDEX_NAME	VARCHAR2(30)	-
INDEX_TYPE	VARCHAR2(1)	NULL
TABLE_OWNER	VARCHAR2(30)	-
TABLE_NAME	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

Table C-7 (Cont.) ALL_INDEXES

Name	Type	Value
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	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

ALL_OBJECTS

Table C-8 ALL_OBJECTS

Name	Type	Value
OWNER	VARCHAR2(30)	-

Table C-8   (Cont.) ALL_OBJECTS

Name	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

ALL_TAB_COLUMNS

Table C-9   ALL_TAB_COLUMNS

Name	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

Table C-9 (Cont.) ALL_TAB_COLUMNS

Name	Type	Value
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

ALL_TAB_COMMENTS

Table C-10 ALL_TAB_COMMENTS

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"
COMMENTS	VARCHAR2(1)	NULL

ALL_TABLES

Table C-11 ALL_TABLES

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLESPACE_NAME	CHAR(1)	" "
CLUSTER_NAME	CHAR(1)	" "
IOT_NAME	CHAR(1)	" "
PCT_FREE	NUMBER(10)	0
PCT_USED	NUMBER(10)	0
INI_TRANS	NUMBER(10)	0
MAX_TRANS	NUMBER(10)	0
INITIAL_EXTENT	NUMBER(10)	0
NEXT_EXTENT	NUMBER(10)	0
MIN_EXTENTS	NUMBER(10)	0
MAX_EXTENTS	NUMBER(10)	0

Table C-11   (Cont.) ALL_TABLES

Name	Type	Value
PCT_INCREASE	NUMBER(10)	0
FREELISTS	NUMBER(10)	0
FREELIST_GROUPS	NUMBER(10)	0
LOGGING	CHAR(1)	" "
BACKED_UP	CHAR(1)	" "
NUM_ROWS	NUMBER(10)	0
BLOCKS	NUMBER(10)	0
EMPTY_BLOCKS	NUMBER(10)	0
AVG_SPACE	NUMBER(10)	0
CHAIN_CNT	NUMBER(10)	0
AVG_ROW_LEN	NUMBER(10)	0
AVG_SPACE_FREELIST_BLOCKS	NUMBER(10)	0
NUM_FREELIST_BLOCKS	NUMBER(10)	0
DEGREE	CHAR(1)	" "
INSTANCES	CHAR(1)	" "
CACHE	CHAR(1)	" "
TABLE_LOCK	CHAR(1)	" "
SAMPLE_SIZE	NUMBER(10)	0
LAST_ANALYZED	DATE	NULL
PARTITIONED	CHAR(1)	" "
IOT_TYPE	CHAR(1)	" "
TEMPORARY	CHAR(1)	" "
SECONDARY	CHAR(1)	" "
NESTED	CHAR(1)	" "
BUFFER_POOL	CHAR(1)	" "
ROW_MOVEMENT	CHAR(1)	" "
GLOBAL_STATS	CHAR(1)	" "
USER_STATS	CHAR(1)	" "
DURATION	CHAR(1)	" "
SKIP_CORRUPT	CHAR(1)	" "
MONITORING	CHAR(1)	" "

ALL_USERS

Table C-12   ALL_USERS

Name	Type	Value
USERNAME	VARCHAR2(30)	-
USER_ID	NUMBER(10)	-
CREATED	DATE	SYSDATE

ALL_VIEWS

Table C-13   ALL_VIEWS

Name	Type	Value
OWNER	CHAR(30)	-
VIEW_NAME	CHAR(30)	-
TEXT_LENGTH	NUMBER(10)	0
TEXT	VARCHAR2(32512)	-
TYPE_TEXT_LENGTH	NUMBER(10)	0
TYPE_TEXT	CHAR(1)	" "
OID_TEXT_LENGTH	NUMBER(10)	0
OID_TEXT	CHAR(1)	" "
VIEW_TYPE_OWNER	CHAR(1)	" "
VIEW_TYPE	CHAR(1)	" "

DBA_CATALOG

Table C-14   DBA_CATALOG

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"

DBA_COL_COMMENTS

Table C-15   DBA_COL_COMMENTS

Name	Type	Value
OWNER	VARCHAR2(30)	-

Table C-15   (Cont.) DBA_COL_COMMENTS

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(30)	-
COMMENTS	CHAR(1)	" "

DBA_OBJECTS

Table C-16   DBA_OBJECTS

Name	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

DBA_TAB_COLUMNS

Table C-17   DBA_TAB_COLUMNS

Name	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	-

Table C-17   (Cont.) DBA_TAB_COLUMNS

Name	Type	Value
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_DEC_LENGTH	NUMBER	0
GLOBAL_STATS	VARCHAR2(1)	NULL
USER_STATS	VARCHAR2(1)	NULL
AVG_COL_LEN	NUMBER	0

DBA_TAB_COMMENTS

Table C-18   DBA_TAB_COMMENTS

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"
COMMENTS	VARCHAR2(1)	NULL

DBA_TABLES

Table C-19   DBA_TABLES

Name	Type	Value
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
TABLESPACE_NAME	CHAR(1)	" "

Table C-19   (Cont.) DBA_TABLES

Name	Type	Value
CLUSTER_NAME	CHAR(1)	" "
IOT_NAME	CHAR(1)	" "
PCT_FREE	NUMBER(10)	0
PCT_USED	NUMBER(10)	0
INI_TRANS	NUMBER(10)	0
MAX_TRANS	NUMBER(10)	0
INITIAL_EXTENT	NUMBER(10)	0
NEXT_EXTENT	NUMBER(10)	0
MIN_EXTENTS	NUMBER(10)	0
MAX_EXTENTS	NUMBER(10)	0
PCT_INCREASE	NUMBER(10)	0
FREELISTS	NUMBER(10)	0
FREELIST_GROUPS	NUMBER(10)	0
LOGGING	CHAR(1)	" "
BACKED_UP	CHAR(1)	" "
NUM_ROWS	NUMBER(10)	0
BLOCKS	NUMBER(10)	0
EMPTY_BLOCKS	NUMBER(10)	0
AVG_SPACE	NUMBER(10)	0
CHAIN_CNT	NUMBER(10)	0
AVG_ROW_LEN	NUMBER(10)	0
AVG_SPACE_FREELIST_BLOCKS	NUMBER(10)	0
NUM_FREELIST_BLOCKS	NUMBER(10)	0
DEGREE	CHAR(1)	" "
INSTANCES	CHAR(1)	" "
CACHE	CHAR(1)	" "
TABLE_LOCK	CHAR(1)	" "
SAMPLE_SIZE	NUMBER(10)	0
LAST_ANALYZED	DATE	NULL
PARTITIONED	CHAR(1)	" "
IOT_TYPE	CHAR(1)	" "
TEMPORARY	CHAR(1)	" "
SECONDARY	CHAR(1)	" "
NESTED	CHAR(1)	" "
BUFFER_POOL	CHAR(1)	" "
ROW_MOVEMENT	CHAR(1)	" "
GLOBAL_STATS	CHAR(1)	" "

Table C-19   (Cont.) DBA_TABLES

Name	Type	Value
USER_STATS	CHAR(1)	" "
DURATION	CHAR(1)	" "
SKIP_CORRUPT	CHAR(1)	" "
MONITORING	CHAR(1)	" "

DICT_COLUMNS

Table C-20   DICT_COLUMNS

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(30)	-
COMMENTS	CHAR(1)	" "

DICTIONARY

Table C-21   DICTIONARY

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
COMMENTS	CHAR(1)	" "

DUAL

Table C-22   DUAL

Name	Type	Value
DUMMY	VARCHAR2(1)	"X"

TABLE_PRIVILEGES

Table C-23   TABLE_PRIVILEGES

Name	Type	Value
GRANTEE	VARCHAR2(30)	-
OWNER	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-

Table C-23   (Cont.) TABLE_PRIVILEGES

Name	Type	Value
GRANTOR	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	DATE	-

USER_CATALOG

Table C-24   USER_CATALOG

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"

USER_COL_COMMENTS

Table C-25   USER_COL_COMMENTS

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(30)	-
COMMENTS	VARCHAR2(1)	NULL

USER_CONS_COLUMNS

Table C-26   USER_CONS_COLUMNS

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

USER_CONSTRAINTS

Table C-27   USER_CONSTRAINTS

Name	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	-

USER_IND_COLUMNS

Table C-28   USER_IND_COLUMNS

Name	Type	Value
INDEX_NAME	VARCHAR2(30)	-
TABLE_NAME	VARCHAR2(30)	-
COLUMN_NAME	VARCHAR2(8192)	-
COLUMN_POSITION	FLOAT(49)	-
COLUMN_LENGTH	FLOAT(49)	-
DESCEND	VARCHAR(4)	"DESC" or "ASC"

USER_INDEXES

Table C-29   USER_INDEXES

Name	Type	Value
INDEX_NAME	VARCHAR2(30)	-

ORACLE

Table C-29   (Cont.) USER_INDEXES

Name	Type	Value
INDEX_TYPE	VARCHAR2(1)	NULL
TABLE_OWNER	VARCHAR2(30)	-
TABLE_NAME	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
TEMPORARY	VARCHAR2(1)	NULL
GENERATED	VARCHAR2(1)	NULL

Table C-29   (Cont.) USER_INDEXES

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

USER_OBJECTS

Table C-30   USER_OBJECTS

Name	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

ORACLE

USER_TAB_COLUMNS

Table C-31   USER_TAB_COLUMNS

Name	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
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

USER_TAB_COMMENTS

Table C-32   USER_TAB_COMMENTS

Name	Type	Value
TABLE_NAME	VARCHAR2(30)	-
TABLE_TYPE	VARCHAR2(5)	"TABLE" or "VIEW"
COMMENTS	VARCHAR2(1)	NULL

USER_TABLES

Table C-33   USER_TABLES

Name	Type	Value
TABLE_NAME	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
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

Table C-33   (Cont.) USER_TABLES

Name	Type	Value
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

USER_USERS

Table C-34   USER_USERS

Name	Type	Value
USERNAME	VARCHAR2(30)	-
USER_ID	NUMBER(5)	-
ACCOUNT_STATUS	VARCHAR2(4)	"OPEN"
LOCK_DATE	DATE	NULL
EXPIRY_DATE	DATE	NULL
DEFAULT_TABLESPACE	VARCHAR2(1)	NULL
TEMPORARY_TABLESPACE	VARCHAR2(1)	NULL
CREATED	DATE	SYSDATE
INITIAL_RSRC_CONSUMER_GROUP	VARCHAR2(1)	NULL
EXTERNAL_NAME	VARCHAR2(1)	NULL

USER_VIEWS

Table C-35   USER_VIEWS

Name	Type	Value
VIEW_NAME	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

Table C-35   (Cont.) USER_VIEWS

Name	Type	Value
VIEW_TYPE_OWNER	VARCHAR2(1)	NULL
VIEW_TYPE	VARCHAR2(1)	NULL

D
Initialization Parameters

The Oracle database initialization parameters in the `init.ora` file are distinct from the 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.

The following topics contain a list of the gateway initialization parameters that can be set for each gateway and their description. The topics also describe the initialization parameter file syntax.

- Initialization Parameter File Syntax
- Oracle Database Gateway for Sybase Initialization Parameters

Initialization Parameter File Syntax

The syntax for the initialization parameter file is as follows:

- The file is a sequence of commands.
- Each command should start on a separate line.
- End of line is considered a command terminator (unless escaped with a backslash).
- If there is a syntax error in an initialization parameter file, none of the settings take effect.
- 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 string of characters that does not contain any backslashes, white space or double quotation marks (")

    – 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 user name 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

The initialization file parameters that can be set for the Oracle Database Gateway for Sybase 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_ENCRYPT_SESSION
- HS_FDS_VALIDATE_SERVER_CERT
- HS_FDS_TRUSTSTORE_FILE
- HS_FDS_TRUSTSTORE_PASSWORD
- HS_FDS_TRANSACTION_ISOLATION
- HS_FDS_TIMESTAMP_MAPPING
- HS_FDS_DATE_MAPPING
- HS_FDS_CONNECT_INFO
- 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
- HS_FDS_QUOTE_IDENTIFIER
- HS_IDLE_TIMEOUT
- HS_NLS_LENGTH_SEMANTICS
- HS_KEEP_REMOTE_COLUMN_SIZE
- HS_FDS_REMOTE_DB_CHARSET
- HS_FDS_SUPPORT_STATISTICS
- HS_FDS_RSET_RETURN_ROWCOUNT
- HS_FDS_SQLLEN_INTERPRETATION
- HS_FDS_REPORT_REAL_AS_DOUBLE
- HS_FDS_ARRAY_EXEC

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
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

Property	Description
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 and the non-Oracle data source are the same. In almost all cases, HS_LANGUAGE should be set exactly the same as Oracle database character set for optimal character set mapping and performance. 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

ORACLE

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.

As more Oracle databases and non-Oracle databases use Unicode as database character sets, it is preferable to also run the gateway in Unicode character set. To do so, you must set `HS_LANGUAGE=AL32UTF8`. However, when the gateway runs on Windows, the Microsoft ODBC Driver Manager interface can exchange data only in the double-byte character set, UCS2. This results in extra ratio expansion of described buffer and column sizes. Refer to HS_FDS_REMOTE_DB_CHARSET for instruction on how to adjust to correct sizes.

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.

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 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 or response time.

HS_OPEN_CURSORS

Property	Description
Default value	50
Range of values	1 to the value of Oracle's OPEN_CURSORS 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. Therefore, setting the same value as the OPEN_CURSORS initialization parameter in the Oracle database 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 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 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 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`, `READ_ONLY_AUTOCOMMIT`, `SINGLE_SITE_AUTOCOMMIT`

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.

- `READ_ONLY_AUTOCOMMIT` provides read only access to the non-Oracle database that does not use logging.

- `SINGLE_SITE_AUTOCOMMIT` provides read and write access to the non-Oracle database without logging. The gateway cannot participate in distributed updates. Moreover, any update to the non-Oracle database is committed immediately.

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 do not apply.

> ✎ **See Also:**
>
> *Oracle Database Reference*

HS_FDS_VALIDATE_SERVER_CERT

Property	Description
Default Value	`ENABLED`
Range of values	`{ENABLED|DISABLED}`
Syntax	`HS_FDS_VALIDATE_SERVER_CERT = {ENABLED|DISABLED}`

Specifies whether the driver validates the certificate that is sent by the database server when SSL encryption is enabled through `HS_FDS_ENCRYPT_SESSION`. When using SSL server authentication, any certificate sent by the server must be issued by a trusted Certificate Authority. Valid values are:

- `ENABLED` : the gateway validates the certificate that is sent by the database server. Any certificate from the server must be issued by a trusted Certificate Authority in the truststore file. The truststore information is specified using the `HS_FDS_TRUSTSTORE_FILE` and `HS_FDS_TRUSTSTORE_PASSWORD` initialization parameters.

- `DISABLED` : the gateway does not validate the certificate that is sent by the database server.

HS_FDS_ENCRYPT_SESSION

Property	Description
Default Value	`NONE`
Range of values	`{NONE|SSL|NOTRUST_SSL}`
Syntax	`HS_FDS_ENCRYPT_SESSION = {NONE|SSL|NOTRUST_SSL}`

Specifies the way the session to Sybase is encrypted. Valid values are:

- `NONE` : data transmitted between the gateway and Sybase is not encrypted. (default).

ORACLE

- `SSL` : data transmitted between the gateway and Sybase is encrypted using SSL.

- `NOTRUST_SSL`: This option is equivalent to the SSL setting, with initialization parameter `HS_FDS_VALIDATE_SERVER_CERT = DISABLED`

HS_FDS_TRUSTSTORE_FILE

Property	Description
Default Value	none
Range of values	*path to truststore file*
Syntax	`HS_FDS_TRUSTSTORE_FILE =` *path to truststore file*

Specifies the path that specifies the location of the truststore file. The truststore file contains a list of the valid Certificate Authorities (CAs) that are trusted by the client machine for SSL server authentication.

HS_FDS_TRUSTSTORE_PASSWORD

Property	Description
Default Value	none
Range of values	*password*
Syntax	`HS_FDS_TRUSTSTORE_PASSWORD=` *password*

Specifies the password required to access the truststore.

HS_FDS_TRANSACTION_ISOLATION

Property	Description
Default Value	`READ_COMMITTED`
Range of Values	`{READ_UNCOMMITTED|READ_COMMITTED|REPEATABLE_READ|SERIALIZABLE}`
Syntax	`HS_FDS_ISOLATION_LEVEL={{READ_UNCOMMITTED|READ_COMMITTED|REPEATABLE_READ|SERIALIZABLE}`

`HS_FDS_TRANSACTION_ISOLATION` specifies the isolation level that is used for the transaction that the gateway opens on the non-Oracle database.

The isolation levels of `READ_UNCOMMITTED`, `READ_COMMITTED`, `REPEATABLE_READ`, and `SERIALIZABLE` are the four isolation levels defined in the SQL standard and adopted by both ANSI and ISO/IEC. For additional information regarding them, see *Oracle Database Concepts*.

Use caution when specifying an isolation level lower than the Oracle transaction isolation level being used, as the gateway transaction will have different Preventable Read Phenomena from what will occur in the Oracle database transaction.

HS_FDS_TIMESTAMP_MAPPING

Property	Description
Default Value	DATE
Range of Values	CHAR|DATE|TIMESTAMP
Syntax	HS_FDS_TIMESTAMP_MAPPING={CHAR|DATE|TIMESTAMP}

If set to CHAR , then non-Oracle target timestamp would be mapped to CHAR(26). If set to DATE (default), then non-Oracle target timestamp would be mapped to Oracle DATE. If set to TIMESTAMP, then non-Oracle target timestamp would be mapped to Oracle TIMESTAMP.

HS_FDS_DATE_MAPPING

Property	Description
Default Value	DATE
Range of Values	DATE|CHAR
Syntax	HS_FDS_DATE_MAPPING={DATE|CHAR}

If set to CHAR, then non-oracle target date would be mapped to CHAR(10). If set to DATE, then non-Oracle target date would be mapped to Oracle date.

HS_FDS_CONNECT_INFO

Property	Description
Default Value	None
Range of Values	Not applicable

HS_FDS_CONNECT_INFO that 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.

This release supports IPv6 format, so you can enter IPv6 format in place of hostname, but you need to wrap square brackets around the IPv6 specification.

For example,

HS_FDS_CONNECT_INFO=[2001:0db8:20c:f1ff:fec6:38af]:port_number/…

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=`*num*

`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.

HS_FDS_QUOTE_IDENTIFIER

Property	Description
Default Value	`TRUE` for Sybase
Range of Values	`TRUE` | `FALSE`
Syntax	`HS_FDS_QUOTE_IDENTIFIER = { TRUE	FALSE }`

`HS_FDS_QUOTE_IDENTIFIER` overrides the target's ability to support quote identifier depending on the value provided. The default value is `TRUE` for Sybase.

HS_IDLE_TIMEOUT

Property	Description
Default Value	0 (no timeout)
Range of Values	0-9999 (minutes)
Syntax	`HS_IDLE_TIMEOUT=`*num*

This feature is only available for Oracle Net TCP protocol. When there is no activity for a connected gateway session for this specified time period, the gateway session would be terminated automatically with pending update (if any) rolled back.

HS_NLS_LENGTH_SEMANTICS

Property	Description
Default Value	`BYTE`
Range of Values	`BYTE` | `CHAR`
Syntax	`HS_NLS_LENGTH_SEMANTICS = { BYTE	CHAR }`

This release of gateway has Character Semantics functionality equivalent to the Oracle Database Character Semantics, that is, `NLS_LENGTH_SEMANTICS`. When `HS_NLS_LENGTH_SEMANTICS` is set to `CHAR`, the `(VAR)CHAR` columns of Sybase database are to be interpreted as having `CHAR` semantics. The only situation the gateway does not

honor the `HS_NLS_LENGTH_SEMANTICS=CHAR` setting is when both Oracle database and the gateway are on the same multi-byte character set.

HS_KEEP_REMOTE_COLUMN_SIZE

Property	Description
Default Value	`OFF`
Range of Values	`OFF` | `LOCAL` | `REMOTE` | `ALL`
Syntax	`HS_KEEP_REMOTE_COLUMN_SIZE = OFF` | `LOCAL` |`REMOTE` | `ALL`
Parameter type	String

`HS_KEEP_REMOTE_COLUMN_SIZE` specifies whether to suppress ratio expansion when computing the length of `(VAR)CHAR` datatypes during data conversion from the non-Oracle database to the gateway, and then to the Oracle database. When it is set to `REMOTE`, the expansion is suppressed between the non-Oracle database and the gateway. When it is set to `LOCAL`, the expansion is suppressed between the gateway and the Oracle database. When it is set to `ALL`, the expansion is suppressed from the non-Oracle database to the Oracle database.

When the parameter is set, the expansion is suppressed when reporting the remote column size, calculating the implicit resulting buffer size, and instantiating in the local Oracle database. This has effect only for remote column size from non-Oracle database to Oracle database. If the gateway runs on Windows and `HS_LANGUAGE=AL32UTF8`, then you must not specify this parameter, as it would influence other ratio related parameter operation. It has no effect for calculating ratio for data moving from Oracle database to non-Oracle database through gateway during `INSERT`, `UPDATE`, or `DELETE`.

HS_FDS_REMOTE_DB_CHARSET

Property	Description
Default Value	None
Range of values	Not applicable
Syntax	`HS_FDS_REMOTE_DB_CHARSET`

This parameter is valid only when `HS_LANGUAGE` is set to `AL32UTF8` and the gateway runs on Windows. As more Oracle databases and non-Oracle databases use Unicode as database character sets, it is preferable to also run the gateway in Unicode character set. To do so, you must set `HS_LANGUAGE=AL32UTF8`. However, when the gateway runs on Windows, the Microsoft ODBC Driver Manager interface can exchange data only in the double-byte character set, UCS2. This results in extra ratio expansion of described buffer and column sizes. To compensate, the gateway can re-adjust the column size if `HS_FDS_REMOTE_DB_CHARSET` is set to the corresponding non-Oracle database character set. For example, `HS_FDS_REMOTE_DB_CHARSET=KO16KSC5601`.

HS_FDS_SUPPORT_STATISTICS

Property	Description
Default Value	`TRUE`
Range of values	`{TRUE|FALSE}`
Syntax	`HS_FDS_SUPPORT_STATISTICS= {TRUE|FALSE}`

We gather statistics from the non-Oracle database by default. You can choose to disable the gathering of remote database statistics by setting the `HS_FDS_SUPPORT_STATISTICS` parameter to `FALSE`.

HS_FDS_RSET_RETURN_ROWCOUNT

Property	Description
Default Value	`FALSE`
Range of values	`{TRUE|FALSE}`
Syntax	`HS_FDS_RSET_RETURN_ROWCOUNT= {TRUE|FALSE}`

When set to `TRUE`, the gateway returns the row counts of DML statements that are executed inside a stored procedure. The row count is returned as a single row, single column result set of type signed integer.

When set to `FALSE`, the gateway skips the row counts of DML statements that are executed inside a stored procedure. This is the default behavior, and it is the behavior of 11.1 and older gateways.

HS_FDS_SQLLEN_INTERPRETATION

Property	Description
Default Value	`64`
Range of values	`{64|32}`
Syntax	`HS_FDS_SQLLEN_INTERPRETATION= {64|32}`

This parameter is only valid for 64 bit platforms. ODBC standard specifies `SQLLEN` (of internal ODBC construct) being 64 bit on 64 bit platforms, but some ODBC driver managers and drivers violate this convention, and implement it as 32 bit. In order for the gateway to compensate their behavior, you need to specify `HS_FDS_SQLLEN_INTERPRETATION=32` if you use these types of driver managers and driver.

HS_FDS_REPORT_REAL_AS_DOUBLE

Property	Description
Default Value	`FALSE`
Range of Values	`TRUE, FALSE`

Enables Oracle Database Gateway for Sybase treat SINGLE FLOAT PRECISION fields as DOUBLE FLOAT PRECISION fields.

HS_FDS_ARRAY_EXEC

Property	Description
Default Value	TRUE
Range of values	{TRUE	FALSE}
Syntax	HS_FDS_ARRAY_EXEC= {TRUE	FALSE}

If set to TRUE, the gateway will use array operations for insert, update, delete statements containing binds against the remote data source. The array size is determined by the value of the HS_FDS_FETCH_ROWS init parameter.

If set to FALSE, the gateway will not use array operations for insert, update, and delete statements. Instead, a single statement will be issued for every value.

Index

G

Gateway
    case studies, *3-1*
    data dictionary tables, *C-1*
    pass-through feature, *2-2*
    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-7
    optimizing data transfer, *D-7*
    setting global name, *D-5*
    specifying cache high water mark, *D-5*
    tuning internal data buffering, *D-7*
    tuning LONG data transfer, *D-6*
Hexadecimal notation, *2-10*
HS_CALL_NAME initialization parameter, *D-3*
HS_DB_NAME initialization parameter, *D-5*
HS_DESCRIBE_CACHE_HWM initialization
        parameter, *D-5*
HS_FDS_CONNECT_INFO, *D-11*
HS_FDS_FETCH_ROWS parameter, *D-14*
HS_FDS_PROC_IS_FUNC initialization
        parameter, *D-12*
HS_FDS_RECOVERY_PWD initialization
        parameter, *D-16*
HS_FDS_RESULTSET_SUPPORT initialization
        parameter, *D-13*
HS_FDS_TRACE_LEVEL initialization
        parameter, *D-13*
    enabling agent tracing, *D-2*
HS_FDS_TRANSACTION_ISOLATIONparamete
        r, *D-10*
HS_FDS_TRANSACTION_LOG initialization
        parameter, *D-13*
HS_IDLE_TIMEOUT initialization parameter,
        D-14
HS_KEEP_REMOTE_COLUMN_SIZE
        initialization parameter, *D-15*
HS_LANGUAGE initialization parameter, *D-5*
HS_LONG_PIECE_TRANSFER_SIZE
        initialization parameter, *D-6*
HS_NLS_LENGTH_SEMANTICS initialization
        parameter, *D-14*
HS_OPEN_CURSORS initialization parameter,
        D-7

HS_RPC_FETCH_REBLOCKING initialization
        parameter, *D-7*
HS_RPC_FETCH_SIZE initialization parameter,
        D-7
HS_TIME_ZONE initialization parameter, *D-8*

I

IFILE initialization parameter, *D-9*
IMAGE data type, *A-1*
Initialization parameter file
    customizing, *D-1*
INSERT statement, *2-14*, *3-5*, *B-1*, *B-2*
INTEGER data type, *A-1*
isolation level,
        HS_FDS_TRANSACTION_ISOLATION,
        D-10

K

Known restrictions, *2-12*

L

Locking, database, *2-11*
LONG data type, *A-1*

M

MONEY data type, *A-1*

N

NULL values, *2-8*
NUMERIC data type, *A-1*

O

Objects, naming rules, *2-9*
ORA-02070, *2-13*

P

parameters
    gateway initialization file
        HS_FDS_FETCH_ROWS, *D-14*
        HS_FDS_TRANSACTION_ISOLATION,
            D-10
Pass-through feature, *2-13*
Pass-Through Feature, *3-6*
Passing commands to database, *2-14*
PL/SQL, *2-16*