CREATE DATABASE LINK

Purpose

Use the CREATE DATABASE LINK statement to create a database link. A **database link** is a schema object in one database that enables you to access objects on another database. The other database need not be an Oracle Database system. However, to access non-Oracle systems you must use Oracle Heterogeneous Services.

After you have created a database link, you can use it to refer to tables and views on the other database. In SQL statements, you can refer to a table or view on the other database by appending *@dblink* to the table or view name. You can query a table or view on the other database with the SELECT statement. You can also access remote tables and views using any INSERT, UPDATE, DELETE, or LOCK TABLE statement.

See Also:

- *Oracle Database Application Developer's Guide - Fundamentals* for information about accessing remote tables or views with PL/SQL functions, procedures, packages, and datatypes
- *Oracle Database Administrator's Guide* for information on distributed database systems
- *Oracle Database Reference* for descriptions of existing database links in the ALL_DB_LINKS, DBA_DB_LINKS, and USER_DB_LINKS data dictionary views and for information on monitoring the performance of existing links through the V$DBLINK dynamic performance view
- DROP DATABASE LINK on page 17-57 for information on dropping existing database links
- INSERT on page 18-51, UPDATE on page 19-59, DELETE on page 17-43, and LOCK TABLE on page 18-68 for using links in DML operations

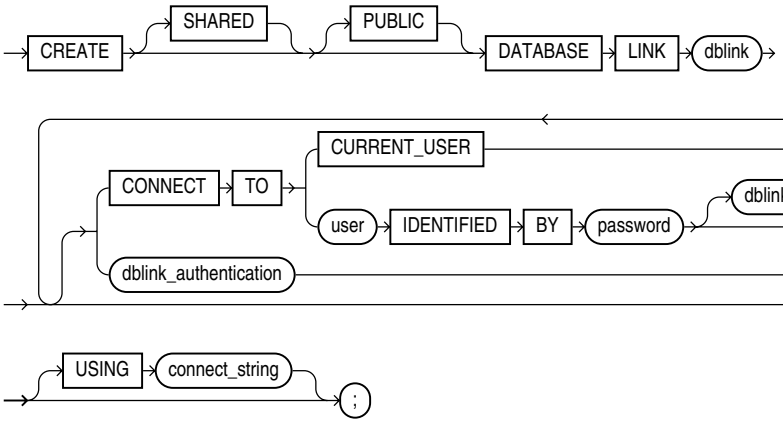
Prerequisites

To create a private database link, you must have the CREATE DATABASE LINK system privilege. To create a public database link, you must have the CREATE PUBLIC DATABASE LINK system privilege. Also, you must have the CREATE SESSION system privilege on the remote Oracle database.

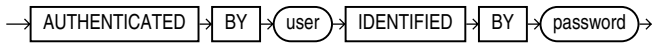
Oracle Net must be installed on both the local and remote Oracle databases.

Syntax

***create_database_link*::=**



***dblink_authentication*::=**



Keyword and Parameters

SHARED

Specify `SHARED` to use a single network connection to create a public database link that can be shared among multiple users. If you specify `SHARED`, you must also specify the *`dblink_authentication`* clause.

> **See Also:** *Oracle Database Administrator's Guide* for more information about shared database links

PUBLIC

Specify `PUBLIC` to create a public database link available to all users. If you omit this clause, the database link is private and is available only to you.

> **See Also:** "Defining a Public Database Link: Example" on page 14-34

dblink

Specify the complete or partial name of the database link. If you specify only the database name, then Oracle Database implicitly appends the database domain of the local database.

Use only ASCII characters for *`dblink`*. Multibyte characters are not supported. The database link name is case insensitive and is stored in uppercase ASCII characters. If you specify the database name as a quoted identifier, then the quotation marks are silently ignored.

If the value of the `GLOBAL_NAMES` initialization parameter is `TRUE`, then the database link must have the same name as the database to which it connects. If the value of `GLOBAL_NAMES` is `FALSE`, and if you have changed the global name of the database, then you can specify the global name.

The maximum number of database links that can be open in one session or one instance of a Real Application Clusters configuration depends on the value of the OPEN_LINKS and OPEN_LINKS_PER_INSTANCE initialization parameters.

Restriction on Creating Database Links You cannot create a database link in another user's schema, and you cannot qualify *dblink* with the name of a schema. Periods are permitted in names of database links, so Oracle Database interprets the entire name, such as ralph.linktosales, as the name of a database link in your schema rather than as a database link named linktosales in the schema ralph.)

See Also:

- "Referring to Objects in Remote Databases" on page 2-104 for guidelines for naming database links
- *Oracle Database Reference* for information on the GLOBAL_NAMES, OPEN_LINKS, and OPEN_LINKS_PER_INSTANCE initialization parameters
- "RENAME GLOBAL_NAME Clause" on page 10-39 (an ALTER DATABASE clause) for information on changing the database global name

CONNECT TO Clause

The CONNECT TO clause lets you enable a connection to the remote database. You can specify this clause and the *dblink_authentication* clause only if you are creating a shared database link.

CURRENT_USER Clause

Specify CURRENT_USER to create a **current user database link**. The current user must be a global user with a valid account on the remote database.

If the database link is used directly, that is, not from within a stored object, then the current user is the same as the connected user.

When executing a stored object (such as a procedure, view, or trigger) that initiates a database link, CURRENT_USER is the username that owns the stored object, and not the username that called the object. For example, if the database link appears inside procedure scott.p (created by scott), and user jane calls procedure scott.p, the current user is scott.

However, if the stored object is an invoker-rights function, procedure, or package, the invoker's authorization ID is used to connect as a remote user. For example, if the privileged database link appears inside procedure scott.p (an invoker-rights procedure created by scott), and user Jane calls procedure scott.p, then CURRENT_USER is jane and the procedure executes with Jane's privileges.

See Also:

- CREATE FUNCTION on page 14-48 for more information on invoker-rights functions
- "Defining a CURRENT_USER Database Link: Example" on page 14-35

user IDENTIFIED BY *password*

Specify the username and password used to connect to the remote database using a **fixed user database link**. If you omit this clause, the database link uses the username

and password of each user who is connected to the database. This is called a **connected user database link**.

> **See Also:** "Defining a Fixed-User Database Link: Example" on page 14-34

dblink_authentication

Specify the username and password on the target instance. This clause authenticates the user to the remote server and is required for security. The specified username and password must be a valid username and password on the remote instance. The username and password are used only for authentication. No other operations are performed on behalf of this user.

You must specify this clause when you specify the `SHARED` clause. You cannot specify this clause unless you specify the `SHARED` clause.

USING '*connect string*'

Specify the service name of a remote database. If you specify only the database name, then Oracle Database implicitly appends the database domain to the connect string to create a complete service name. Therefore, if the database domain of the remote database is different from that of the current database, then you must specify the complete service name.

> **See Also:** *Oracle Database Administrator's Guide* for information on specifying remote databases

Examples

The examples that follow assume two databases, one with the database name `local` and the other with the database name `remote`. The examples use the Oracle Database domain. Your database domain will be different.

Defining a Public Database Link: Example The following statement defines a shared public database link named `remote` that refers to the database specified by the service name `remote`:

```
CREATE PUBLIC DATABASE LINK remote
   USING 'remote';
```

This database link allows user `hr` on the `local` database to update a table on the `remote` database (assuming `hr` has appropriate privileges):

```
UPDATE employees@remote
   SET salary=salary*1.1
   WHERE last_name = 'Baer';
```

Defining a Fixed-User Database Link: Example In the following statement, user `hr` on the `remote` database defines a fixed-user database link named `local` to the `hr` schema on the `local` database:

```
CREATE DATABASE LINK local
   CONNECT TO hr IDENTIFIED BY hr
   USING 'local';
```

After this database link is created, `hr` can query tables in the schema `hr` on the `local` database in this manner:

```
SELECT * FROM employees@local;
```

User hr can also use DML statements to modify data on the local database:

```
INSERT INTO employees@local
   (employee_id, last_name, email, hire_date, job_id)
   VALUES (999, 'Claus', 'sclaus@oracle.com', SYSDATE, 'SH_CLERK');

UPDATE jobs@local SET min_salary = 3000
   WHERE job_id = 'SH_CLERK';

DELETE FROM employees@local
   WHERE employee_id = 999;
```

Using this fixed database link, user hr on the remote database can also access tables owned by other users on the same database. This statement assumes that user hr has SELECT privileges on the oe.customers table. The statement connects to the user hr on the local database and then queries the oe.customers table:

```
SELECT * FROM oe.customers@local;
```

Defining a CURRENT_USER Database Link: Example The following statement defines a current-user database link to the remote database, using the entire service name as the link name:

```
CREATE DATABASE LINK remote.us.oracle.com
   CONNECT TO CURRENT_USER
   USING 'remote';
```

The user who issues this statement must be a global user registered with the LDAP directory service.

You can create a synonym to hide the fact that a particular table is on the remote database. The following statement causes all future references to emp_table to access the employees table owned by hr on the remote database:

```
CREATE SYNONYM emp_table
   FOR oe.employees@remote.us.oracle.com;
```