17

SQL Statements: DROP LIBRARY to DROP SYNONYM

This chapter contains the following SQL statements:

DROP LIBRARY

Purpose

Use the `DROP LIBRARY` statement to remove an external procedure library from the database.

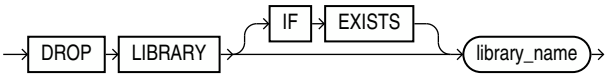
> **See Also:**
>
> CREATE LIBRARY for information on creating a library

Prerequisites

You must have the `DROP ANY LIBRARY` system privilege.

Syntax

***drop_library*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

library_name

Specify the name of the external procedure library being dropped.

Examples

Dropping a Library: Example

The following statement drops the `ext_lib` library:

```
DROP LIBRARY ext_lib;
```

DROP LOCKDOWN PROFILE

Purpose

Use the `DROP LOCKDOWN PROFILE` statement to remove a PDB lockdown profile from the database. A PDB that was assigned the dropped profile will continue to be assigned the profile, but will not be subject to the restrictions imposed by the dropped profile.

If the `PDB_LOCKDOWN` initialization parameter for a CDB, an application root, or a PDB has the value of the dropped lockdown profile, then the restrictions imposed by the dropped profile will be disabled when you drop it. However, the value of the `PDB_LOCKDOWN` initialization parameter will remain until you explicitly unset it.

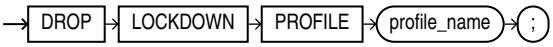
> **See Also:**
>
> - CREATE LOCKDOWN PROFILE and ALTER LOCKDOWN PROFILE
> - *Oracle Database Security Guide* for more information on PDB lockdown profiles

Prerequisites

- You must issue this statement from the CDB Root or the Application Root.
- You must have the `DROP LOCKDOWN PROFILE` system privilege in the container where you mean to issue the statement.

Syntax

***drop_lockdown_profile*::=**



Semantics

profile_name

Specify the name of the PDB lockdown profile to be dropped.

You can find the names of existing PDB lockdown profiles by querying the `DBA_LOCKDOWN_PROFILES` data dictionary view.

> **See Also:**
>
> *Oracle Database Reference* for more information on the `DBA_LOCKDOWN_PROFILES` data dictionary view and the `PDB_LOCKDOWN` initialization parameter

Example

The following statement drops PDB lockdown profile `hr_prof`:

```
DROP LOCKDOWN PROFILE hr_prof;
```

DROP MATERIALIZED VIEW

Purpose

Use the `DROP MATERIALIZED VIEW` statement to remove an existing materialized view from the database.

When you drop a materialized view, Oracle Database does not place it in the recycle bin. Therefore, you cannot subsequently either purge or undrop the materialized view.

> **Note:**
>
> The keyword `SNAPSHOT` is supported in place of `MATERIALIZED VIEW` for backward compatibility.

> **See Also:**
>
> - CREATE MATERIALIZED VIEW for more information on the various types of materialized views
> - ALTER MATERIALIZED VIEW for information on modifying a materialized view
> - *Oracle Database Administrator's Guide* for information on materialized views in a replication environment
> - *Oracle Database Data Warehousing Guide* for information on materialized views in a data warehousing environment

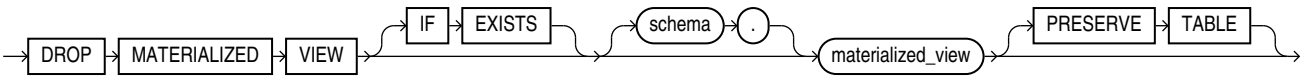
Prerequisites

The materialized view must be in your own schema or you must have the `DROP ANY MATERIALIZED VIEW` system privilege. You must also have the privileges to drop the internal table, views, and index that the database uses to maintain the materialized view data.

> **See Also:**
>
> DROP TABLE , DROP VIEW , and DROP INDEX for information on privileges required to drop objects that the database uses to maintain the materialized view

Syntax

***drop_materialized_view*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement.`

schema

Specify the schema containing the materialized view. If you omit *`schema`*, then Oracle Database assumes the materialized view is in your own schema.

materialized_view

Specify the name of the existing materialized view to be dropped.

- If you drop a simple materialized view that is the least recently refreshed materialized view of a master table, then the database automatically purges from the master table materialized view log only the rows needed to refresh the dropped materialized view.

- If you drop a materialized view that was created on a prebuilt table, then the database drops the materialized view, and the prebuilt table reverts to its identity as a table.
- When you drop a master table, the database does not automatically drop materialized views based on the table. However, the database returns an error when it tries to refresh a materialized view based on a master table that has been dropped.
- If you drop a materialized view, then any compiled requests that were rewritten to use the materialized view will be invalidated and recompiled automatically. If the materialized view was prebuilt on a table, then the table is not dropped, but it can no longer be maintained by the materialized view refresh mechanism.

PRESERVE TABLE Clause

This clause lets you retain the materialized view container table and its contents after the materialized view object is dropped. The resulting table has the same name as the dropped materialized view.

Oracle Database removes all metadata associated with the materialized view. However, indexes created on the container table automatically during creation of the materialized view are preserved, with one exception: the index created during the creation of a rowid materialized view is dropped. Also, if the materialized view has any nested table columns, then the storage tables for those columns are preserved, along with their metadata.

Restriction on the PRESERVE TABLE Clause

This clause is not valid for materialized views that have been imported from releases earlier than Oracle9*i*, when these objects were called "snapshots".

Examples

Dropping a Materialized View: Examples

The following statement drops the materialized view `emp_data` in the sample schema `hr`:

```
DROP MATERIALIZED VIEW emp_data;
```

The following statement drops the `sales_by_month_by_state` materialized view and the underlying table of the materialized view, unless the underlying table was registered in the `CREATE MATERIALIZED VIEW` statement with the `ON PREBUILT TABLE` clause:

```
DROP MATERIALIZED VIEW sales_by_month_by_state;
```

DROP MATERIALIZED VIEW LOG

Purpose

Use the `DROP MATERIALIZED VIEW LOG` statement to remove a materialized view log from the database.

> **Note:**
>
> The keyword `SNAPSHOT` is supported in place of `MATERIALIZED VIEW` for backward compatibility.

> **See Also:**
>
> - CREATE MATERIALIZED VIEW and ALTER MATERIALIZED VIEW for more information on materialized views
> - CREATE MATERIALIZED VIEW LOG for information on materialized view logs
> - *Oracle Database Administrator's Guide* for information on materialized views in a replication environment
> - *Oracle Database Data Warehousing Guide* for information on materialized views in a data warehousing environment

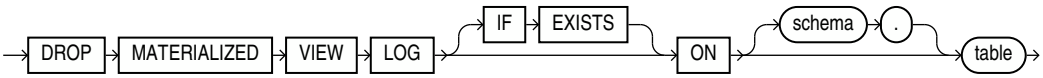
Prerequisites

To drop a materialized view log, you must have the privileges needed to drop a table.

> **See Also:**
>
> DROP TABLE

Syntax

***drop_materialized_view_log*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the materialized view log and its master table. If you omit *`schema`*, then Oracle Database assumes the materialized view log and master table are in your own schema.

table

Specify the name of the master table associated with the materialized view log to be dropped.

After you drop a materialized view log that was created `FOR FAST REFRESH`, some materialized views based on the materialized view log master table can no longer be fast refreshed. These materialized views include rowid materialized views, primary key materialized views, and subquery materialized views.

> **See Also:**
>
> *Oracle Database Data Warehousing Guide* for a description of these types of materialized views

After you drop a materialized view log that was created `FOR SYNCHRONOUS REFRESH` (a staging log), the materialized views based on the staging log master table can no longer be synchronous refreshed.

Examples

Dropping a Materialized View Log: Example

The following statement drops the materialized view log on the `oe.customers` master table:

```
DROP MATERIALIZED VIEW LOG ON customers;
```

DROP MATERIALIZED ZONEMAP

Purpose

Use the `DROP MATERIALIZED ZONEMAP` statement to remove an existing zone map from the database.

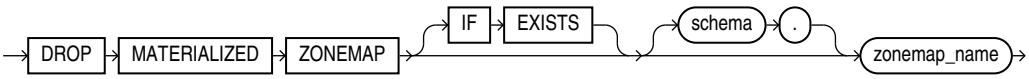
Prerequisites

The zone map must be in your own schema or you must have the `DROP ANY MATERIALIZED VIEW` system privilege. You must also have the privileges to drop the internal table and indexes that the database uses to maintain the zone map data.

> **See Also:**
>
> DROP TABLE and DROP INDEX for information on privileges required to drop objects that the database uses to maintain the zone map

Syntax

***drop_materialized_zonemap*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the zone map. If you omit *schema*, then Oracle Database assumes the zone map is in your own schema.

zonemap_name

Specify the name of the existing zone map to be dropped.

Example

Dropping a Zone Map: Examples

The following statement drops the zone map `sales_zmap`:

```
DROP MATERIALIZED ZONEMAP sales_zmap;
```

DROP MLE ENV

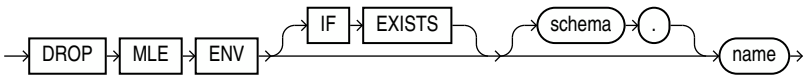
Purpose

Drop an exisiting environment with `DROP MLE ENV`.

Prerequisites

You must have the `DROP ANY MLE` privilege to drop an environment in schemas other than your own. No privilege is needed to drop an environment in your own schema.

Syntax



Semantics

Specify `IF EXISTS` to drop an existing MLE environment.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

See Also:

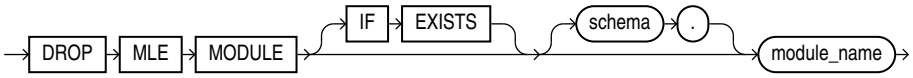
- CREATE MLE ENV
- ALTER MLE ENV
- CREATE MLE MODULE

DROP MLE MODULE

Purpose

You can drop a previously deployed MLE module using `DROP MLE MODULE` .

Syntax



Semantics

Specify `IF EXISTS` to drop an existing MLE module.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement.`

`schema` Specify the schema containing the MLE module. If you do not specify the schema, then Oracle Database assumes that the module is in your own schema.

`module_name` refers to the module name.

> **See Also:**
>
> - CREATE MLE MODULE
> - ALTER MLE MODULE

DROP OPERATOR

Purpose

Use the `DROP OPERATOR` statement to drop a user-defined operator.

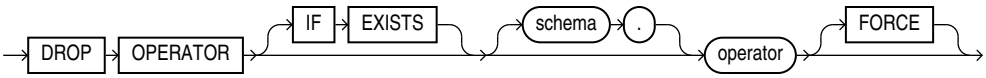
> **See Also:**
>
> - CREATE OPERATOR and ALTER OPERATOR for information on creating and modifying operators
> - "User-Defined Operators " and *Oracle Database Data Cartridge Developer's Guide* for more information on operators in general
> - ALTER INDEXTYPE for information on dropping an operator of a user-defined indextype

Prerequisites

The operator must be in your schema or you must have the `DROP ANY OPERATOR` system privilege.

Syntax

***drop_operator*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the operator. If you omit *`schema`*, then Oracle Database assumes the operator is in your own schema.

operator

Specify the name of the operator to be dropped.

FORCE

Specify `FORCE` to drop the operator even if it is currently being referenced by one or more schema objects, such as indextypes, packages, functions, procedures, and so on. The database marks any such dependent objects `INVALID`. Without `FORCE`, you cannot drop an operator if any schema objects reference it.

Examples

Dropping a User-Defined Operator: Example

The following statement drops the operator `eq_op`:

```
DROP OPERATOR eq_op;
```

Because the `FORCE` clause is not specified, this operation will fail if any of the bindings of this operator are referenced by an indextype.

DROP OUTLINE

Purpose

> **Note:**
>
> - Stored outlines are deprecated. They are still supported for backward compatibility. However, Oracle recommends that you use SQL plan management instead. SQL plan management creates SQL plan baselines, which offer superior SQL performance stability compared with stored outlines.
> - You can migrate existing stored outlines to SQL plan baselines by using the `MIGRATE_STORED_OUTLINE` function of the `DBMS_SPM` package or Enterprise Manager Cloud Control. When the migration is complete, the stored outlines are marked as migrated and can be removed. You can drop all migrated stored outlines on your system by using the `DROP_MIGRATED_STORED_OUTLINE` function of the `DBMS_SPM` package.
> - **See Also:** *Oracle Database SQL Tuning Guide* for more information about SQL plan management and *Oracle Database PL/SQL Packages and Types Reference* for information about the `DBMS_SPM` package

Use the `DROP OUTLINE` statement to drop a stored outline.

> **See Also:**
>
> CREATE OUTLINE for information on creating an outline

Prerequisites

To drop an outline, you must have the `DROP ANY OUTLINE` system privilege.

Syntax

***drop_outline*::=**

Semantics

outline

Specify the name of the outline to be dropped.

After the outline is dropped, if the SQL statement for which the stored outline was created is compiled, then the optimizer generates a new execution plan without the influence of the outline.

Examples

Dropping an Outline: Example

The following statement drops the stored outline called `salaries`.

```
DROP OUTLINE salaries;
```

DROP PACKAGE

Purpose

Packages are defined using PL/SQL. Refer to *Oracle Database PL/SQL Language Reference* for complete information on creating, altering, and dropping packages.

Use the `DROP PACKAGE` statement to remove a stored package from the database. This statement drops the body and specification of a package.

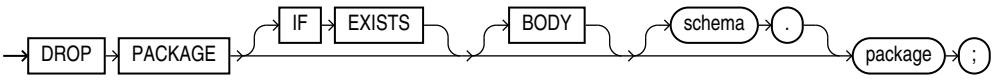
> **Note:**
>
> Do not use this statement to remove a single object from a package. Instead, re-create the package without the object using the `CREATE PACKAGE` and `CREATE PACKAGE BODY` statements with the `OR REPLACE` clause.

Prerequisites

The package must be in your own schema or you must have the `DROP ANY PROCEDURE` system privilege.

Syntax

***drop_package*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

BODY

Specify `BODY` to drop only the body of the package. If you omit this clause, then Oracle Database drops both the body and specification of the package.

When you drop only the body of a package but not its specification, the database does not invalidate dependent objects. However, you cannot call one of the procedures or stored functions declared in the package specification until you re-create the package body.

schema

Specify the schema containing the package. If you omit *schema*, then the database assumes the package is in your own schema.

package

Specify the name of the package to be dropped.

Oracle Database invalidates any local objects that depend on the package specification. If you subsequently reference one of these objects, then the database tries to recompile the object and returns an error if you have not re-created the dropped package.

If any statistics types are associated with the package, then the database disassociates the statistics types with the `FORCE` clause and drops any user-defined statistics collected with the statistics types.

> **See Also:**
>
> ASSOCIATE STATISTICS and DISASSOCIATE STATISTICS

Examples

Dropping a Package: Example

The following statement drops the specification and body of the `emp_mgmt` package, invalidating all objects that depend on the specification. See *Oracle Database PL/SQL Language Reference* for the example that creates this package.

```
DROP PACKAGE emp_mgmt;
```

DROP PLUGGABLE DATABASE

Purpose

Use the `DROP PLUGGABLE DATABASE` statement to drop a pluggable database (PDB). The PDB can be a traditional PDB, an application container, an application seed, or an application PDB.

When you drop a PDB, the control file of the multitenant container database (CDB) is modified to remove all references to the dropped PDB and its data files. Archived logs and backups associated with the dropped PDB are not deleted. You can delete them using Oracle Recovery Manager (RMAN), or you can retain them in case you subsequently want to perform point-in-time recovery of the PDB.

> **Caution:**
>
> You cannot roll back a `DROP PLUGGABLE DATABASE` statement.

Prerequisites

You must be connected to a CDB.

To drop a traditional PDB or an application container, the current container must be the root, you must be authenticated AS SYSDBA or AS SYSOPER, and the SYSDBA or SYSOPER privilege must be either granted to you commonly, or granted to you locally in the root and locally in traditional PDB or application container you want to drop. The application container must be empty, that is, it must not contain an application seed or any application PDBs.

To drop an application seed, the current container must be the root or the application root, you must be authenticated AS SYSDBA or AS SYSOPER, and the SYSDBA or SYSOPER privilege must be either granted to you commonly, or granted to you locally in the root or application root.

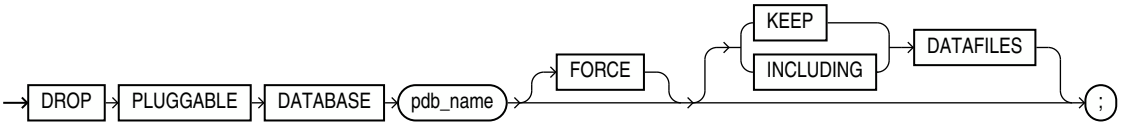
To drop an application PDB, the current container must be the root or the application root, you must be authenticated AS SYSDBA or AS SYSOPER, and the SYSDBA or SYSOPER privilege must be either granted to you commonly, or granted to you locally in the root or application root, and locally in the application PDB you want to drop.

To specify KEEP DATAFILES (the default), the PDB you want to drop must be unplugged.

To specify INCLUDING DATAFILES, the PDB you want to drop must be in mounted mode or it must be unplugged.

Syntax

***drop_pluggable_database*::=**



Semantics

pdb_name

Specify the name of the PDB you want to drop. You cannot drop the seed (PDB$SEED). However, you can drop an application seed.

FORCE

Use FORCE to drop an orphaned application root container.

FORCE requires the following condition: the APP_ROOT_CLONE must be closed, and the APP_CDB must be open.

To close the APP_ROOT_CLONE, you must set the variable _ORACLE_SCRIPT" to true using ALTER SESSION.

Keeping APP_CDB open follow the instructions to close the APP_ROOT_CLONE:

```
ALTER SESSION SET _ORACLE_SCRIPT"=true ;
ALTER PLUGGABLE DATABASE APP_ROOT_CLONE CLOSE;
DROP PLUGGABLE DATABASE APP_ROOT_CLONE FORCE INCLUDING DATAFILES;
```

> **See Also:**
>
> *Removing a PDB*

KEEP DATAFILES

Specify `KEEP DATAFILES` to retain the data files associated with the PDB after the PDB is dropped. The temp file for the PDB is deleted because it is no longer needed. This is the default.

Keeping data files may be useful in scenarios where a PDB that is unplugged from one CDB is plugged into another CDB, with both CDBs sharing storage devices.

INCLUDING DATAFILES

Specify `INCLUDING DATAFILES` to delete the data files associated with the PDB being dropped. The temp file for the PDB is also deleted.

Restriction on Dropping SNAPSHOT COPY PDBs

If a PDB was created with the `SNAPSHOT COPY` clause, then you must specify `INCLUDING DATAFILES` when you drop the PDB.

Examples

Dropping a PDB: Example

The following statement drops the PDB `pdb1` and its associated data files:

```
DROP PLUGGABLE DATABASE pdb1
  INCLUDING DATAFILES;
```

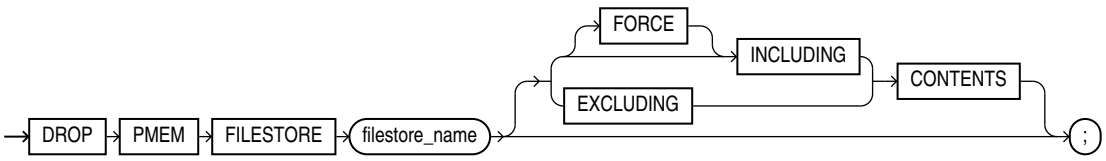
DROP PMEM FILESTORE

Purpose

You can drop a PMEM file store with this command.

Syntax

***drop_pmem_filestore*::=**



Semantics

INCLUDING CONTENTS

Specify `INCLUDING CONTENTS` to confirm that Oracle should remove all the files in the PMEM file store.

EXCLUDING CONTENTS

Specify `EXCLUDING CONTENTS` to ensure that Oracle drops the PMEM file store only when the file store is empty.

FORCE

Specify `FORCE` along with `INCLUDING CONTENTS` if you suspect that the file store is corrupt.

Note that this option does not check if the file store has content in it prior to deleting it.

If you specify neither `INCLUDING CONTENTS` nor `EXCLUDING CONTENTS`, you must ensure that the file store is empty. `EXCLUDING CONTENTS` is the default behavior.

Example

```
DROP PMEM FILESTORE cloud_db_1 EXCLUDING CONTENTS
```

DROP PROCEDURE

Purpose

Procedures are defined using PL/SQL. Refer to *Oracle Database PL/SQL Language Reference* for complete information on creating, altering, and dropping procedures.

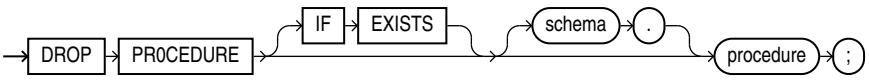
Use the `DROP PROCEDURE` statement to remove a standalone stored procedure from the database. Do not use this statement to remove a procedure that is part of a package. Instead, either drop the entire package using the `DROP PACKAGE` statement, or redefine the package without the procedure using the `CREATE PACKAGE` statement with the `OR REPLACE` clause.

Prerequisites

The procedure must be in your own schema or you must have the `DROP ANY PROCEDURE` system privilege.

Syntax

***drop_procedure*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the procedure. If you omit *`schema`*, then Oracle Database assumes the procedure is in your own schema.

procedure

Specify the name of the procedure to be dropped.

When you drop a procedure, Oracle Database invalidates any local objects that depend upon the dropped procedure. If you subsequently reference one of these objects, then the database

tries to recompile the object and returns an error message if you have not re-created the dropped procedure.

Examples

Dropping a Procedure: Example

The following statement drops the procedure `remove_emp` owned by the user `hr` and invalidates all objects that depend upon `remove_emp`:

```
DROP PROCEDURE hr.remove_emp;
```

DROP PROFILE

Purpose

Use the `DROP PROFILE` statement to remove a profile from the database. You can drop any profile except the `DEFAULT` profile.

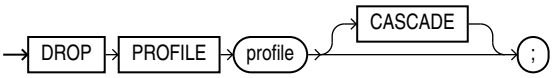
> **See Also:**
>
> CREATE PROFILE and ALTER PROFILE on creating and modifying a profile

Prerequisites

You must have the `DROP PROFILE` system privilege.

Syntax

***drop_profile*::=**



Semantics

profile

Specify the name of the profile to be dropped.

CASCADE

Specify `CASCADE` to deassign the profile from any users to whom it is assigned. Oracle Database automatically assigns the `DEFAULT` profile to such users. You must specify this clause to drop a profile that is currently assigned to users.

Examples

Dropping a Profile: Example

The following statement drops the profile `app_user`, which was created in "Creating a Profile: Example". Oracle Database drops the profile `app_user` and assigns the `DEFAULT` profile to any users currently assigned the `app_user` profile:

```
DROP PROFILE app_user CASCADE;
```

DROP PROPERTY GRAPH

Purpose

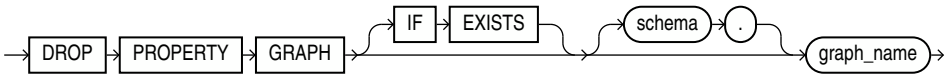
You can drop property graphs with `DROP PROPERTY GRAPH`.

Prerequistes

Like tables and views, you can drop a property graph in your own schema. To drop a property graph in any schema except `SYS` and `AUDSYS`, you must have the `DROP ANY PROPERTY GRAPH` privilege.

Syntax

***drop_property_graph*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing property graph.

If you specify `IF NOT EXISTS` with `DROP`, the command fails with the error message: `Incorrect IF EXISTS clause for ALTER/DROP statement`.

DROP RESTORE POINT

Purpose

Use the `DROP RESTORE POINT` statement to remove a normal restore point or a guaranteed restore point from the database.

- You need not drop normal restore points. The database automatically drops the oldest restore points when necessary, as described in the semantics for *restore_point*. However, you can drop a normal restore point if you want to reuse the name.
- Guaranteed restore points are not dropped automatically. Therefore, if you want to remove a guaranteed restore point from the database, then you must do so explicitly using this statement.

> **See Also:**
>
> CREATE RESTORE POINT , FLASHBACK DATABASE, and FLASHBACK TABLE for information on creating and using restore points

Prerequisites

To drop a normal restore point, you must have the `SELECT ANY DICTIONARY`, `FLASHBACK ANY TABLE`, `SYSBACKUP`, or `SYSDG` system privilege.

To drop a guaranteed restore point, you must fulfill *one* of the following conditions:

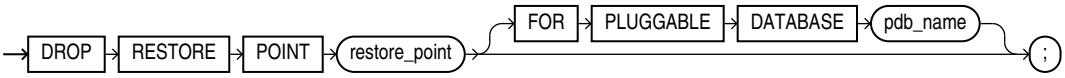
- You must connect `AS SYSDBA`, or `AS SYSBACKUP`, or `AS SYSDG`.
- You must have been granted the `SYSDBA` privilege, and be using a multitenant database.
- You must be running as user `SYS`, and be using a a multitenant database.

You can drop a restore point when connected to a multitenant container database (CDB) as follows:

- To drop a normal CDB restore point, the current container must be the root and you must have the `SELECT ANY DICTIONARY` or `FLASHBACK ANY TABLE` system privilege, either granted commonly or granted locally in the root, or the `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege granted commonly.
- To drop a guaranteed CDB restore point, the current container must be the root and you must have the `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege granted commonly.
- To drop a normal PDB restore point, the current container must be the root and you must have the `SELECT ANY DICTIONARY`, `FLASHBACK ANY TABLE`, `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege, granted commonly, or the current container must be the PDB in which you want to create the restore point and you must have the `SELECT ANY DICTIONARY`, `FLASHBACK ANY TABLE`, `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege, granted commonly or granted locally in that PDB.
- To drop a guaranteed PDB restore point, the current container must be the root and you must have the `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege, granted commonly, or the current container must be the PDB in which you want to create the restore point and you must have the `SYSDBA`, `SYSBACKUP`, or `SYSDG` system privilege, granted commonly or granted locally in that PDB.

Syntax

***drop_restore_point*::=**



Semantics

restore_point

Specify the name of the restore point you want to drop.

FOR PLUGGABLE DATABASE

This clause enables you to drop a PDB restore point when you are connected to the root. For *pdb_name*, specify the name of the PDB that contains the restore point you want to drop.

If you are connected to the PDB from which you want to drop the restore point, then it is not necessary to specify this clause. However, if you specify this clause, then you must specify the name of the PDB to which you are connected.

Examples

Dropping a Restore Point: Example

The following example drops the `good_data` restore point, which was created in "Creating and Using a Restore Point: Example":

```
DROP RESTORE POINT good_data;
```

DROP ROLE

Purpose

Use the `DROP ROLE` statement to remove a role from the database. When you drop a role, Oracle Database revokes it from all users and roles to whom it has been granted and removes it from the database. User sessions in which the role is already enabled are not affected. However, no new user session can enable the role after it is dropped.

> **See Also:**
>
> - CREATE ROLE and ALTER ROLE for information on creating roles and changing the authorization needed to enable a role
> - SET ROLE for information on disabling roles for the current session

Prerequisites

You must have been granted the role with the `ADMIN OPTION` or you must have the `DROP ANY ROLE` system privilege.

Syntax

***drop_role*::=**

Semantics

role

Specify the name of the role to be dropped.

Examples

Dropping a Role: Example

To drop the role `dw_manager`, which was created in "Creating a Role: Example", issue the following statement:

```
DROP ROLE dw_manager;
```

DROP ROLLBACK SEGMENT

Purpose

Use the `DROP ROLLBACK SEGMENT` to remove a rollback segment from the database. When you drop a rollback segment, all space allocated to the rollback segment returns to the tablespace.

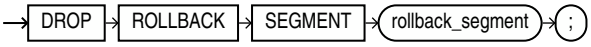
> **Note:**
>
> If your database is running in automatic undo mode, then this is the only valid operation on rollback segments. In that mode, you cannot create or alter a rollback segment.

Prerequisites

You must have the `DROP ROLLBACK SEGMENT` system privilege, and the rollback segment must be offline.

Syntax

***drop_rollback_segment*::=**



Semantics

rollback_segment

Specify the name the rollback segment to be dropped.

Restrictions on Dropping Rollback Segments

This statement is subject to the following restrictions:

- You can drop a rollback segment only if it is offline. To determine whether a rollback segment is offline, query the data dictionary view `DBA_ROLLBACK_SEGS`. Offline rollback segments have the value `AVAILABLE` in the `STATUS` column. You can take a rollback segment offline with the `OFFLINE` clause of the `ALTER ROLLBACK SEGMENT` statement.
- You cannot drop the `SYSTEM` rollback segment.

Examples

Dropping a Rollback Segment: Example

The following syntax drops the rollback segment created in "Creating a Rollback Segment: Example":

```
DROP ROLLBACK SEGMENT rbs_one;
```

DROP SEQUENCE

Purpose

Use the `DROP SEQUENCE` statement to remove a sequence from the database.

You can also use this statement to restart a sequence by dropping and then re-creating it. For example, if you have a sequence with a current value of 150 and you would like to restart the sequence with a value of 27, then you can drop the sequence and then re-create it with the same name and a `START WITH` value of 27.

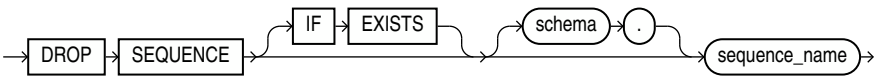
> **See Also:**
>
> CREATE SEQUENCE and ALTER SEQUENCE for more information on creating and modifying a sequence

Prerequisites

The sequence must be in your own schema or you must have the `DROP ANY SEQUENCE` system privilege.

Syntax

***drop_sequence*::=**



Semantics

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the sequence. If you omit *`schema`*, then Oracle Database assumes the sequence is in your own schema.

sequence_name

Specify the name of the sequence to be dropped.

Examples

Dropping a Sequence: Example

The following statement drops the sequence `customers_seq` owned by the user `oe`, which was created in "Creating a Sequence: Example". To issue this statement, you must either be connected as user `oe` or have the `DROP ANY SEQUENCE` system privilege:

```
DROP SEQUENCE oe.customers_seq;
```

DROP SYNONYM

Purpose

Use the `DROP SYNONYM` statement to remove a synonym from the database or to change the definition of a synonym by dropping and re-creating it.

> **See Also:**
>
> CREATE SYNONYM for more information on synonyms

Prerequisites

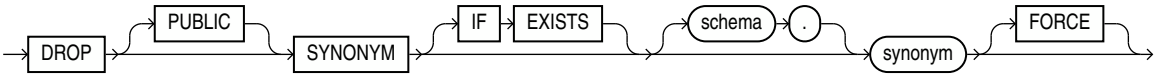
To drop a private synonym, either the synonym must be in your own schema or you must have the `DROP ANY SYNONYM` system privilege.

To drop a `PUBLIC` synonym, you must have the `DROP PUBLIC SYNONYM` system privilege.

Syntax

***drop_synonym*::=**

drop_synonym::=



Semantics

PUBLIC

You must specify `PUBLIC` to drop a public synonym. You cannot specify *`schema`* if you have specified `PUBLIC`.

IF EXISTS

Specify `IF EXISTS` to drop an existing object.

Specifying `IF NOT EXISTS` with `DROP` results in `ORA-11544: Incorrect IF EXISTS clause for ALTER/DROP statement`.

schema

Specify the schema containing the synonym. If you omit *`schema`*, then Oracle Database assumes the synonym is in your own schema.

synonym

Specify the name of the synonym to be dropped.

If you drop a synonym for the master table of a materialized view, and if the defining query of the materialized view specified the synonym rather than the actual table name, then Oracle Database marks the materialized view unusable.

If an object type synonym has any dependent tables or user-defined types, then you cannot drop the synonym unless you also specify `FORCE`.

FORCE

Specify `FORCE` to drop the synonym even if it has dependent tables or user-defined types.

> **Note:**
>
> Oracle does not recommend that you specify `FORCE` to drop object type synonyms with dependencies. This operation can result in invalidation of other user-defined types or marking `UNUSED` the table columns that depend on the synonym. For information about type dependencies, see *Oracle Database Object-Relational Developer's Guide*.

Examples

Dropping a Synonym: Example

To drop the public synonym named `customers`, which was created in "Oracle Database Resolution of Synonyms: Example", issue the following statement:

```
DROP PUBLIC SYNONYM customers;
```