82
DBMS_DST

The DBMS_DST package provides an interface to apply the Daylight Saving Time (DST) patch to the Timestamp with Time Zone datatype.

This chapter contains the following topics:

- Overview
- Security Model
- Views
- Summary of DBMS_DST Subprograms

> **See Also:**
>
> - *Oracle Database Globalization Support Guide*
> - *Oracle Database Reference*

DBMS_DST Overview

The transition period during which Daylight Saving Time comes into effect, or stops being in effect, has the potential for problems, such as data loss, when handling timestamps with time zone data. The `DBMS_DST` package enables working with these transitions in the context of a set of rules.

DBMS_DST Security Model

The `DBMS_DST` package is an invoker's rights package.

> **See Also:**
>
> *Oracle Database PL/SQL Language Reference* for more information about using Invoker Rights or Definer Rights

The execute privilege on the package is granted to the `EXECUTE_CATALOG_ROLE` role. This role is normally granted to selected users to allow `EXECUTE` privileges for packages and procedures in the data dictionary.

The user that invokes the package must have the following privileges:

- `CREATE ANY TABLE`
- `ALTER ANY TABLE`

- DROP ANY TABLE
- SELECT ANY TABLE
- LOCK ANY TABLE
- ALTER ANY INDEX
- ALTER ANY TRIGGER
- UPDATE ANY TABLE
- EXECUTE ANY TYPE

DBMS_DST Views

The DBMS_DST package uses views to display table information.

These views are shown in the following table. They are further described in the *Oracle Database Reference*:

Table 82-1 Views used by DBMS_DST

View	Description
DBA_TSTZ_TABLES	Displays information about all tables in the database, which have columns defined on TIMESTAMP WITH TIME ZONE datatypes or object types containing attributes of TIMESTAMP WITH TIME ZONE datatypes. Its columns are the same as those in ALL_TSTZ_TABLES
USER_TSTZ_TABLES	Displays information about the tables owned by the current user, which have columns defined on TIMESTAMP WITH TIME ZONE datatypes or object types containing attributes of TIMESTAMP WITH TIME ZONE datatypes. Its columns (except for OWNER) are the same as those in ALL_TSTZ_TABLES.
ALL_TSTZ_TABLES	Displays information about the tables accessible to the current user, which have columns defined on TIMESTAMP WITH TIME ZONE datatypes or object types containing attributes of TIMESTAMP WITH TIME ZONE datatypes

Summary of DBMS_DST Subprograms

This table lists and describes the DBMS_DST package subprograms.

Table 82-2 DBMS_DST Package Subprograms

Subprogram	Description
BEGIN_PREPARE Procedure	Starts a prepare window
BEGIN_UPGRADE Procedure	Starts an upgrade window
CREATE_AFFECTED_TABLE Procedure	Creates a table that has the schema shown in the comments for the FIND_AFFECTED_TABLES Procedure
CREATE_ERROR_TABLE Procedure	Creates a log error table

Table 82-2 (Cont.) DBMS_DST Package Subprograms

Subprogram	Description
CREATE_TRIGGER_TABLE Procedure	Creates a table that is used to record active triggers disabled before performing upgrade on the table, having not been enabled due to fatal failure during the upgrading process
END_PREPARE Procedure	Ends a prepare window
END_UPGRADE Procedure	Ends an upgrade window
FIND_AFFECTED_TABLES Procedure	Finds all the tables that have affected TSTZ data due to the new timezone version
UPGRADE_DATABASE Procedure	Upgrades all tables in the database that have one or more columns defined on the TSTZ type, or an ADT containing the TSTZ type
UPGRADE_SCHEMA Procedure	Upgrades tables in a specified list of schemas that has one or more columns defined on the TSTZ type, or an ADT containing the TSTZ type
UPGRADE_TABLE Procedure	Upgrades a specified list of tables that has one or more columns defined on the TSTZ type or an ADT containing the TSTZ type

BEGIN_PREPARE Procedure

This procedure starts a prepare window. Once a prepare window is started successfully, the database property 'DST_UPGRADE_STATE' is set to 'PREPARE', and the database property 'SECONDARY_TT_VERSION' is set to a new timezone version.

The prepare window lets a DBA investigate data affected by the upgrade, and so judge when it is optimal to perform the upgrade. The prepare window can overlap normal database operation.

Syntax

```
DBMS_DST.BEGIN_PREPARE (
   new_version               IN  BINARY_INTEGER);
```

Parameters

Table 82-3 BEGIN_PREPARE Procedure Parameters

Parameter	Description
new_version	New timezone version to which the database is to be prepared to upgrade

BEGIN_UPGRADE Procedure

This procedure starts an upgrade window.

Syntax

```
DBMS_DST.BEGIN_UPGRADE (
   new_version               IN  BINARY_INTEGER,
   error_on_overlap_time     IN  BOOLEAN := FALSE,
   error_on_nonexisting_time IN  BOOLEAN := FALSE);
```

Parameters

Table 82-4 BEGIN_UPGRADE Procedure Parameters

Parameter	Description
new_version	New timezone version to which the database is to be upgraded
error_on_overlap_time	Boolean flag indicating whether to report errors on the 'overlap' time semantic conversion error. The default is FALSE. For more information about boundary cases, see *Oracle Database SQL Language Reference*.
error_on_nonexisting_time	Boolean flag indicating whether to report errors on the 'non-existing' time semantic conversion error. The default value is FALSE.

CREATE_AFFECTED_TABLE Procedure

This procedure creates a table that has the schema shown in the comments for the FIND_AFFECTED_TABLES Procedure.

Syntax

```
DBMS_DST.CREATE_AFFECTED_TABLE (
   table_name      IN  VARCHAR2);
```

Parameters

Table 82-5 CREATE_AFFECTED_TABLE Procedure Parameters

Parameter	Description
table_name	Name of the table created

Usage Notes

This procedures takes a table_name without schema qualification, creating a table within the current user schema.

Related Topics

- FIND_AFFECTED_TABLES Procedure
  This procedure finds all the tables which have affected TSTZ data due to the new timezone version.

CREATE_ERROR_TABLE Procedure

This procedure creates a log error table.

The table has the following schema:

```
CREATE TABLE dst$error_table(
   table_owner     VARCHAR2(30),
   table_name      VARCHAR2(30),
   column_name     VARCHAR2(4000),
   rid             ROWID,
   error_number    NUMBER)
```

Syntax

```
DBMS_DST.CREATE_ERROR_TABLE (
   table_name      IN  VARCHAR2);
```

Parameters

Table 82-6 CREATE_ERROR_TABLE Procedure Parameters

Parameter	Description
table_name	Name of the table created

Usage Notes

- This procedures takes a `table_name` without schema qualification, creating a table within the current user schema.
- The error number is found when upgrading time zone file and timestamp with time zone data. For more information about error handling when upgrading time zone file and timestamp with time zone data, see Oracle Database Globalization Support Guide

CREATE_TRIGGER_TABLE Procedure

This procedure creates a table to record active triggers that are disabled before performing upgrade on the table, having not been enabled due to fatal failure during the upgrading process.

The table that has the following schema.

```
CREATE TABLE dst_trigger_table (
trigger_owner  VARCHAR2(30),
trigger_name   VARCHAR2(30));
```

Syntax

```
DBMS_DST.CREATE_TRIGGER_TABLE (
   table_name     IN    VARCHAR2);
```

Parameters

Table 82-7 CREATE_TRIGGER_TABLE Procedure Parameters

Parameter	Description
table_name	Name of table to be created

Usage Notes

This procedures takes a `table_name` without schema qualification, creating a table within the current user schema.

END_PREPARE Procedure

This procedure ends a prepare window.

Syntax

```
DBMS_DST.END_PREPARE;
```

END_UPGRADE Procedure

This procedure ends an upgrade window. An upgraded window is ended if all the affected user tables have been upgraded. Otherwise, the `OUT` parameter `num_of_failures` indicates how many tables have not been converted.

Syntax

```
DBMS_DST.END_UPGRADE (
  num_of_failures    OUT  BINARY_INTEGER);
```

Parameters

Table 82-8 END_UPGRADE Procedure Parameters

Parameter	Description
`num_of_failures`	Number of tables that fail to complete

FIND_AFFECTED_TABLES Procedure

This procedure finds all the tables which have affected TSTZ data due to the new timezone version.

This procedure can only be invoked during a prepare window. The tables which have affected TSTZ data are recorded into a table indicated by parameter `affected_tables`. If semantic errors must be logged, they are recorded into a table indicated by parameter `log_errors_table`.

Syntax

```
DBMS_DST.FIND_AFFECTED_TABLES (
   affected_tables             IN  VARCHAR2 =: 'sys.dst$affected_tables',
   log_errors                  IN  BOOLEAN := FALSE,
   log_errors_table            IN  VARCHAR2 =: 'sys.dst$error_table',
   parallel                    IN  BOOLEAN := FALSE);
```

Parameters

Table 82-9 FIND_AFFECTED_TABLES Procedure Parameters

Parameter	Description

affected_tables	Name of table with the following schema:
	`CREATE TABLE dst$affected_tables (`
	`table_owner VARCHAR2(30),`
	`table_name VARCHAR2(30),`
	`column_name VARCHAR2(4000),`
	`row_count NUMBER,`
	`error_count NUMBER)`
	The table can be created with the CREATE_AFFECTED_TABLE Procedure.

log_errors	Boolean flag indicating whether to log errors during upgrade. If FALSE, no error is logged into the log_errors_table after aborting conversion of the current table. If TRUE, the error is logged to the log_errors_table. The default is FALSE.

log_errors_table	Table name with the following schema:
	`CREATE TABLE dst$error_table (`
	`table_owner VARCHAR2(30),`
	`table_name VARCHAR2(30),`
	`column_name VARCHAR2(4000),`
	`rid ROWID,`
	`error_number NUMBER)`
	The table can be created with the CREATE_ERROR_TABLE Procedure. The rid column records the rowids of the offending rows, and the error_number column records the corresponding error number.

parallel	Boolean flag indicating whether to find the affected tables using parallel queries or serial queries. The default is FALSE.

UPGRADE_DATABASE Procedure

This procedure upgrades all tables in the database, which have one or more columns defined on the TSTZ type or an ADT containing the TSTZ type.

This procedure can only be invoked after an upgrade window has been started. Each table is upgraded in an atomic transaction. Note that, a base table and its materialized view log table are upgraded in an atomic transaction.

Syntax

```
DBMS_DST.UPGRADE_DATABASE  (
   num_of_failures             OUT BINARY_INTEGER,
   upgrade_data                IN  BOOLEAN := TRUE,
   parallel                    IN  BOOLEAN := FALSE,
   continue_after_errors       IN  BOOLEAN := TRUE,
   log_errors                  IN  BOOLEAN := FALSE,
   log_errors_table            IN  VARCHAR2 =: 'sys.dst$error_table' ,
   error_on_overlap_time       IN  BOOLEAN := FALSE,
```

```
  error_on_nonexisting_time   IN  BOOLEAN := FALSE,
  log_triggers_table          IN  VARCHAR2 := 'sys.dst$trigger_table');
```

Parameters

Table 82-10 UPGRADE_DATABASE Procedure Parameters

Parameter	Description
num_of_failures	Number of tables that fail to complete
upgrade_data	Boolean flag indicating whether to convert TSTZ data using the new Time Zone patch File (TRUE), or to leave it unconverted (FALSE).The default is TRUE.
parallel	Boolean flag indicating whether to convert tables using PDML (Parallel DML) or Serial DML.The default is FALSE.
continue_after_errors	Boolean flag indicating whether to continue after upgrade fails on the current table. The default is TRUE.
log_errors	Boolean flag indicating whether to log errors during upgrade. If FALSE, no error is logged into the log_errors_table after aborting conversion of the current table. If TRUE, errors are logged to the log_errors_table. The default is FALSE.

log_errors_table	Table name with the following schema:
	`CREATE TABLE dst$error_table (`
	`table_owner VARCHAR2(30),`
	`table_name VARCHAR2(30),`
	`column_name VARCHAR2(4000),`
	`rid ROWID,`
	`error_number NUMBER)`
	The table can be created with the CREATE_ERROR_TABLE Procedure. The rid column records the rowids of the offending rows, and the error_number column records the corresponding error number.

error_on_overlap_time	Boolean flag indicating whether to report errors on the 'overlap' time semantic conversion error. The default is TRUE.
error_on_nonexisting_time	Boolean flag indicating whether to report errors on the 'non-existing' time semantic conversion error. The default is TRUE.
log_triggers_table	Table to log triggers which are disabled before upgrade, having not been enabled due to a fatal failure when performing an upgrade

UPGRADE_SCHEMA Procedure

This procedure upgrades tables in a specified list of schemas that have one or more columns defined on the TSTZ type, or an ADT containing the TSTZ type.

This procedure can be invoked only after an upgrade window has been started. Each table is upgraded in an atomic transaction. Note that a base table and its materialized view log table are upgraded in an atomic transaction.

Syntax

```
DBMS_DST.UPGRADE_SCHEMA (
   num_of_failures             OUT BINARY_INTEGER,
   schema_list                 IN  VARCHAR2,
   upgrade_data                IN  BOOLEAN := TRUE,
   parallel                    IN  BOOLEAN := FALSE,
   continue_after_errors       IN  BOOLEAN := TRUE,
   log_errors                  IN  BOOLEAN := FALSE,
   log_errors_table            IN  VARCHAR2 =: 'sys.dst$error_table' ,
   error_on_overlap_time       IN  BOOLEAN := FALSE,
   error_on_nonexisting_time   IN  BOOLEAN := FALSE,
   log_triggers_table          IN  VARCHAR2 := 'sys.dst$trigger_table');
```

Parameters

Table 82-11 UPGRADE_SCHEMA Procedure Parameters

Parameter	Description
num_of_failures	Number of tables that fail to complete
schema_list	Schema name list (comma separated strings)
upgrade_data	Boolean flag indicating whether to convert TSTZ data using the new Time Zone patch File (TRUE) or to leave unconverted (FALSE).The default is TRUE.
parallel	Boolean flag indicating whether to convert tables using PDML (Parallel DML) or Serial DML.The default is FALSE.
continue_after_errors	Boolean flag indicating whether to continue after upgrade fails on the current table.The default is TRUE.
log_errors	Boolean flag indicating whether to log errors during upgrade. If FALSE, no error is logged into the log_errors_table after aborting conversion of the current table. If TRUE, the error is logged to the log_errors_table. The default is FALSE.

log_errors_table	Table name with the following schema:
	`CREATE TABLE dst$error_table (`
	`table_owner VARCHAR2(30),`
	`table_name VARCHAR2(30),`
	`column_name VARCHAR2(4000),`
	`rid ROWID,`
	`error_number NUMBER)`
	The table can be created with the CREATE_ERROR_TABLE Procedure. The rid column records the rowids of the offending rows, and the error_number column records the corresponding error number.

error_on_overlap_time	Boolean flag indicating whether to report errors on the 'overlap' time semantic conversion error. The default is TRUE.
error_on_nonexisting_time	Boolean flag indicating whether to report errors on the 'non-existing' time semantic conversion error. The default is TRUE.

Table 82-11 (Cont.) UPGRADE_SCHEMA Procedure Parameters

Parameter	Description
log_triggers_table	Table to log triggers that are disabled before upgrade, having not been enabled due to a fatal failure when performing an upgrade

UPGRADE_TABLE Procedure

This procedure upgrades a specified list of tables that have one or more columns defined on the TSTZ type, or an ADT containing the TSTZ type.

Syntax

```
DBMS_DST.UPGRADE_TABLE  (
   num_of_failures             OUT BINARY_INTEGER,
   table_list                  IN  VARCHAR2,
   upgrade_data                IN  BOOLEAN := TRUE,
   parallel                    IN  BOOLEAN := FALSE,
   continue_after_errors       IN  BOOLEAN := TRUE,
   log_errors                  IN  BOOLEAN := FALSE,
   log_errors_table            IN  VARCHAR2 =: 'sys.dst$error_table' ,
   error_on_overlap_time       IN  BOOLEAN := FALSE,
   error_on_nonexisting_time   IN  BOOLEAN := FALSE,
   log_triggers_table          IN  VARCHAR2 := 'sys.dst$trigger_table',
   atomic_upgrade              IN  BOOLEAN := FALSE);
```

Parameters

Table 82-12 UPGRADE_TABLE Procedure Parameters

Parameter	Description
num_of_failures	Number of tables that fail to complete
table_list	Table name list (comma separated strings)
upgrade_data	Boolean flag indicating whether to convert TSTZ data using the new Time Zone patch File (TRUE), or to leave unconverted (FALSE). The default is TRUE.
parallel	Boolean flag indicating whether to convert tables using PDML (Parallel DML), or Serial DML. The default is FALSE.
continue_after_errors	Boolean flag indicating whether to continue after upgrade fails on the current table. The default is TRUE.
log_errors	Boolean flag indicating whether to log errors during upgrade. If FALSE, no error is logged into the log_errors_table after aborting conversion of the current table. If TRUE, the error is logged to the log_errors_table. The default is FALSE.

Table 82-12 (Cont.) UPGRADE_TABLE Procedure Parameters

Parameter	Description

log_errors_table	Table name with the following schema:
	`CREATE TABLE dst$error_table (`
	`table_owner VARCHAR2(30),`
	`table_name VARCHAR2(30),`
	`column_name VARCHAR2(4000),`
	`rid ROWID,`
	`error_number NUMBER)`
	The table can be created with the CREATE_ERROR_TABLE Procedure. The rid parameter records the rowids of the offending rows and the corresponding error number.

error_on_overlap_time	Boolean flag indicating whether to report errors on the 'overlap' time semantic conversion error. The default is TRUE.
error_on_nonexisting_time	Boolean flag indicating whether to report errors on the 'non-existing' time semantic conversion error. The default is TRUE.
log_triggers_table	Table to log triggers that are disabled before upgrade, having not been enabled due to a fatal failure when performing an upgrade
atomic_upgrade	Boolean flag indicating whether to convert the listed tables atomically (in a single transaction). If FALSE, each table is converted in its own transaction. The default is FALSE.

Usage Notes

This procedure can only be invoked after an upgrade window has been started. The table list has to satisfy the following partial ordering:

1. If a base table has a materialized view log table, the log table must be the next item in the list.
2. If the container table for a materialized view appears in the list, the materialized view's 'non-upgraded' base tables and log tables must appear in the table list and before the container table.

A base table and its materialized view log table need to be upgraded in an atomic transaction by specifying atomic_upgrade to TRUE.