181
DBMS_SHARDING_DIRECTORY

This package provides procedures to manage an Oracle Globally Distributed Database sharded database created with directory-based data distribution.

This chapter contains the following topics:

DBMS_SHARDING_DIRECTORY Overview

PL/SQL package `DBMS_SHARDING_DIRECTORY` provides support operations on the directory table for sharded databases using the directory-based data distribution method.

This package includes procedures to:

- Add a key to a partition entry
- Remove a key mapping entry
- Flag a key for partition split

The APIs in this package can be invoked on the shard catalog.

The directory table is automatically created during root table creation. The definition of the directory table is `shard user schema.root_table$SDIR`.

When adding and removing keys there are APIs that include commit and those that do not. Unless the commit versions of the APIs are used, the directory content is not propagated to the shards until commit is issued explicitly.

> **See Also:**
>
> Directory-Based Sharding in *Oracle Globally Distributed Database Guide* for a detailed description of sharded database directory-based data distribution.

DBMS_SHARDING_DIRECTORY Security Model

All `DBMS_SHARDING_DIRECTORY` subprograms require the user to have `EXECUTE` privilege over the `DBMS_SHARDING_DIRECTORY` package.

Only the schema owner of the root table is allowed to execute the procedures in this package, and they can only execute it on the shard catalog.

DBMS_SHARDING_DIRECTORY Public Constants

There are public constants defined for use with the `DBMS_SHARDING_DIRECTORY.setAssignmentRule` procedure for key-to-partition assignment rules.

All such constants are defined as part of the `DBMS_SHARDING_DIRECTORY` package. Any references to these constants must be prefixed by `DBMS_SHARDING_DIRECTORY.` and followed by the symbols in the following lists

- `NONE constant number :=0;` -- turn off rule-based assignment
- `LAST_PARTITION constant number := 1;` -- rule for assigning key only to the last added partition
- `ROUNT_ROBIN constant number :=2;` -- rule for assigning key to partition by round robin
- `RANDOM constant number :=3;` -- rule for assigning key to partition randomly
- `CUSTOM constant number :=4;` -- TBD

Summary of DBMS_SHARDING_DIRECTORY Subprograms

This table lists and describes the subprograms of the `DBMS_SHARDING_DIRECTORY` package

Table 181-1 DBMS_SHARDING_DIRECTORY Package Subprograms

Subprogram	Description
addKeyToPartition Procedure	This procedure allows you to add a new key to the directory with the specified partition name.
addKeyToPartitionCommit Procedure	This procedure allows you to add a new key to the directory with the specified partition name, and performs a commit at the end.
flagKeyForSplit Procedure	This procedure allows you to mark a key in the directory for split, to be performed later.
removeKey Procedure	This procedure allows you to remove a key from the directory.
removeKeyCommit Procedure	This procedure allows you to remove a key from the directory, and performs a commit at the end.
setAssignmentRule	This procedure allows you to indicate an automatic key-to-partition assignment rule for subsequent new key inserts into the root table.

addKeyToPartition Procedure

This procedure allows you to add a new key to the directory with the specified partition name.

Syntax

```
DBMS_SHARDING_DIRECTORY.addKeyToPartition(
   (schema_name    IN varchar2,
    root_table     IN varchar2,
    partition_name IN  varchar2,
    key … );
```

Parameters

Table 181-2 addKeyToPartition Procedure Parameters

Parameter	Description
schema_name	Root table schema name.
partition_name	Name of the partition.
root_table	Root table name.
key	Shard key column values.

Usage Notes

Note that the key column value needs to be in the same order as specified in the CREATE TABLE statement with the correct types. The procedure can only succeed if the provided key does not yet exist in the directory.

addKeyToPartitionCommit Procedure

This procedure allows you to add a new key to the directory with the specified partition name.

Syntax

```
DBMS_SHARDING_DIRECTORY.addKeyToPartitionCommit(
   (schema_name    IN varchar2,
    root_table     IN varchar2,
    partition_name IN  varchar2,
    key … );
```

Parameters

Table 181-3 addKeyToPartitionCommit Procedure Parameters

Parameter	Description
schema_name	Root table schema name.
partition_name	Name of the partition.
root_table	Root table name.
key	Shard key column values.

Usage Notes

The addKeyToPartitionCommit procedure is exactly the same as the addKeyToPartition procedure with the same parameters, except that it performs a commit automatically at the end.

Note that the key column value needs to be in the same order as specified in the CREATE TABLE statement with the correct types. The procedure can only succeed if the provided key does not yet exist in the directory.

flagKeyForSplit Procedure

This procedure allows you to mark a key in the directory for split, to be performed later.

Syntax

```
DBMS_SHARDING_DIRECTORY.flagKeyForSplit(
   (schema_name    IN varchar2,
    root_table     IN varchar2,
    key … );
```

Parameters

Table 181-4 flagKeyForSplit Procedure Parameters

Parameter	Description
schema_name	Root table schema name.
root_table	Root table name.
key	Shard key column values.

Usage Notes

Note that the `key` column values need to be in the same order as specified in the `CREATE TABLE` statement with the correct types. The procedure can only succeed if the provided key exists in the directory.

A subsequent `ALTER TABLE SPLIT PARTITION` operation will go through all the keys that have been marked for split in the directory and split the corresponding data out into the new partition.

removeKey Procedure

This procedure allows you to remove a key from the directory.

Syntax

```
DBMS_SHARDING_DIRECTORY.removeKey(
   (schema_name    IN varchar2,
    root_table     IN varchar2,
    key … );
```

Parameters

Table 181-5 removeKey Procedure Parameters

Parameter	Description
schema_name	Root table schema name.
root_table	Root table name.
key	Shard key column values.

Usage Notes

Note that the `key` column values need to be in the same order as specified in the `CREATE TABLE` statement with the correct types. The procedure can only succeed if the provided key exists in the directory, and there are no tables (either root table or child tables) with rows still referencing the key.

removeKeyCommit Procedure

This procedure allows you to remove a key from the directory.

Syntax

```
DBMS_SHARDING_DIRECTORY.removeKeyCommit(
   (schema_name     IN varchar2,
    root_table      IN varchar2,
    key … );
```

Parameters

Table 181-6 removeKeyCommit Procedure Parameters

Parameter	Description
`schema_name`	Root table schema name.
`root_table`	Root table name.
`key`	Shard key column values.

Usage Notes

The `removeKeyCommit` procedure is exactly the same as the `removeKey` procedure with the same parameters, except that it performs a commit automatically at the end.

Note that the `key` column values need to be in the same order as specified in the `CREATE TABLE` statement with the correct types. The procedure can only succeed if the provided key exists in the directory, and there are no tables (either root table or child tables) with rows still referencing the key.

setAssignmentRule

This procedure allows you to indicate an automatic key-to-partition assignment rule for subsequent new key inserts into the root table.

It will be in effect across different sessions and regardless of system restart until another call to this procedure is made with a different `rule_id` value, or with `NONE`, meaning automatic assignment should be turned off.

Rule ID values are defined in DBMS_SHARDING_DIRECTORY Public Constants.

Syntax

```
DBMS_SHARDING_DIRECTORY.setAssignmentRule(
   (schema_name     IN varchar2,
    root_table      IN varchar2,
    rule_id         IN number);
```

Parameters

Table 181-7 setAssignmentRule Procedure Parameters

Parameter	Description
`schema_name`	Root table schema name.
`root_table`	Root table name.
`rule_id`	Rule ID value, as defined in DBMS_SHARDING_DIRECTORY Public Constants.