198

DBMS_STORAGE_MAP

With the `DBMS_STORAGE_MAP` package, you can communicate with the Oracle background process FMON to invoke mapping operations that populate mapping views. FMON communicates with operating and storage system vendor-supplied mapping libraries.

This chapter contains the following topics:

DBMS_STORAGE_MAP Overview

This terminology and descriptions will help you understand the `DBMS_STORAGE_MAP` API.

- Mapping libraries

  Mapping libraries help you map the components of I/O processing stack elements. Examples of I/O processing components include files, logical volumes, and storage array I/O targets. The mapping libraries are identified in `filemap.ora`.

- Mapping files

  A mapping file is a mapping structure that describes a file. It provides a set of attributes, including file size, number of extents that the file is composed of, and file type.

- Mapping elements and sub-elements

  A mapping element is the abstract mapping structure that describes a storage component within the I/O stack. Examples of elements include mirrors, stripes, partitions, raid5, concatenated elements, and disks—structures that are the mapping building blocks. A mapping sub-element describes the link between an element and the next elements in the I/O mapping stack

- Mapping file extents

  A mapping file extent describes a contiguous chunk of blocks residing on one element. This includes the device offset, the extent size, the file offset, the type (data or parity), and the name of the element where the extent resides. In the case of a raw device or volume, the file is composed of only one file extent component. A mapping file extent is different from Oracle extents.

> **See Also:**
>
> - *Oracle Database Administrator's Guide*for more information
> - *Oracle Database Reference*for `V$MAP` views, including `V$MAP_FILE`, `V$MAP_ELEMENT`, `V$MAP_SUBELEMENT`, `V$MAP_FILE_EXTENT`

DBMS_STORAGE_MAP Operational Notes

Invoking the `MAP_ELEMENT`, `MAP_FILE`, and `MAP_ALL` functions when mapping information already exists will refresh the mapping, if configuration IDs are supported. If configuration IDs are not supported, invoking these functions again will rebuild the mapping.

> **See Also:**
>
> *Oracle Database Administrator's Guide*for a discussion of the configuration ID, an attribute of the element or file that is changed.

Summary of DBMS_STORAGE_MAP Subprograms

This table lists the `DBMS_STORAGE_MAP` subprograms and briefly describes them.

Table 198-1 DBMS_STORAGE_MAP Package Subprograms

Subprogram	Description
DROP_ALL Function	Drops all mapping information in the shared memory of the instance
DROP_ELEMENT Function	Drops the mapping information for the element defined by `elemname`
DROP_FILE Function	Drops the file mapping information defined by `filename`
LOCK_MAP Procedure	Locks the mapping information in the shared memory of the instance
MAP_ALL Function	Builds the entire mapping information for all types of Oracle files (except archive logs), including all directed acyclic graph (DAG) elements
MAP_ELEMENT Function	Builds mapping information for the element identified by `elemname`
MAP_FILE Function	Builds mapping information for the file identified by `filename`
MAP_OBJECT Function	Builds the mapping information for the Oracle object identified by the object name, owner, and type
RESTORE Function	Loads the entire mapping information from the data dictionary into the shared memory of the instance
SAVE Function	Saves information needed to regenerate the entire mapping into the data dictionary
UNLOCK_MAP Procedure	Unlocks the mapping information in the shared memory of the instance.

DROP_ALL Function

This function drops all mapping information in the shared memory of the instance.

Syntax

```
DBMS_STORAGE_MAP.DROP_ALL(
      dictionary_update IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-2 DROP_ALL Function Parameters

Parameter	Description
dictionary_update	If TRUE, mapping information in the data dictionary is updated to reflect the changes. The default value is TRUE; dictionary_update is an overloaded argument.

DROP_ELEMENT Function

This function drops the mapping information for the element defined by elemname.

Syntax

```
DBMS_STORAGE_MAP.DROP_ELEMENT(
   elemname          IN VARCHAR2,
   cascade           IN BOOLEAN,
   dictionary_update IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-3 DROP_ELEMENT Function Parameters

Parameter	Description
elemname	The element for which mapping information is dropped.
cascade	If TRUE, then DROP_ELEMENT is invoked recursively on all elements of the DAG defined by elemname, if possible.
dictionary_update	If TRUE, mapping information in the data dictionary is updated to reflect the changes. The default value is TRUE; dictionary_update is an overloaded argument.

DROP_FILE Function

This function drops the file mapping information defined by filename.

Syntax

```
DBMS_STORAGE_MAP.DROP_FILE(
   filename          IN VARCHAR2,
   cascade           IN BOOLEAN,
   dictionary_update IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-4 DROP_FILE Function Parameters

Parameter	Description
filename	The file for which file mapping information is dropped.
cascade	If TRUE, then the mapping DAGs for the elements where the file resides are also dropped, if possible.

Table 198-4 (Cont.) DROP_FILE Function Parameters

Parameter	Description
dictionary_update	If TRUE, mapping information in the data dictionary is updated to reflect the changes. The default value is TRUE; dictionary_update is an overloaded argument.

LOCK_MAP Procedure

This procedure locks the mapping information in the shared memory of the instance.

This is useful when you need a consistent snapshot of the V$MAP tables. Without locking the mapping information, V$MAP_ELEMENT and V$MAP_SUBELEMENT, for example, may be inconsistent.

Syntax

```
DBMS_STORAGE_MAP.LOCK_MAP;
```

MAP_ALL Function

This function builds the entire mapping information for all types of Oracle files (except archive logs), including all directed acyclic graph (DAG) elements. It obtains the latest mapping information because it explicitly synchronizes all mapping libraries.

Syntax

```
DBMS_STORAGE_MAP.MAP_ALL(
   max_num_fileext   IN NUMBER DEFAULT 100,
   dictionary_update IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-5 MAP_ALL Function Parameters

Parameter	Description
max_num_fileext	Defines the maximum number of file extents to be mapped. This limits the amount of memory used when mapping file extents. The default value is 100; max_num_fileextent is an overloaded argument.
dictionary_update	If TRUE, mapping information in the data dictionary is updated to reflect the changes. The default value is TRUE; dictionary_update is an overloaded argument.

Usage Notes

You must explicitly call MAP_ALL in a cold startup scenario.

MAP_ELEMENT Function

This function builds mapping information for the element identified by elemname. It may not obtain the latest mapping information if the element being mapped, or any one of the elements

within its I/O stack (if `cascade` is `TRUE`), is owned by a library that must be explicitly synchronized.

Syntax

```
DBMS_STORAGE_MAP.MAP_ELEMENT(
   elemname          IN VARCHAR2,
   cascade           IN BOOLEAN,
   dictionary_update IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-6 MAP_ELEMENT Function Parameters

Parameter	Description
`elemname`	The element for which mapping information is built.
`cascade`	If `TRUE`, all elements within the elemname I/O stack DAG are mapped.
`dictionary_update`	If `TRUE`, mapping information in the data dictionary is updated to reflect the changes. The default value is `TRUE`; `dictionary_update` is an overloaded argument.

MAP_FILE Function

This function builds mapping information for the file identified by `filename`. Use this function if the mapping of one particular file has changed. The Oracle database server does not have to rebuild the entire mapping.

Syntax

```
DBMS_STORAGE_MAP.MAP_FILE(
   filename           IN VARCHAR2,
   filetype           IN VARCHAR2,
   cascade            IN BOOLEAN,
   max_num_fileextent IN NUMBER DEFAULT 100,
   dictionary_update  IN BOOLEAN DEFAULT TRUE);
```

Parameters

Table 198-7 MAP_FILE Function Parameters

Parameter	Description
`filename`	The file for which mapping information is built.
`filetype`	Defines the type of the file to be mapped. It can be `"DATAFILE"`, `"SPFILE"`, `"TEMPFILE"`, `"CONTROLFILE"`, `"LOGFILE"`, or `"ARCHIVEFILE"`.
`cascade`	Should be `TRUE` only if a storage reconfiguration occurred. For all other instances, such as file resizing (either through an `ALTER SYSTEM` command or DML operations on extended files), `cascade` can be set to `FALSE` because the mapping changes are limited to the file extents only. If `TRUE`, mapping DAGs are also built for the elements where the file resides.

Table 198-7 (Cont.) MAP_FILE Function Parameters

Parameter	Description
max_num_fileextent	Defines the maximum number of file extents to be mapped. This limits the amount of memory used when mapping file extents. The default value is 100; max_num_fileextent is an overloaded argument.
dictionary_update	If TRUE, mapping information in the data dictionary is updated to reflect the changes. The default value is TRUE; dictionary_update is an overloaded argument.

Usage Notes

This function may not obtain the latest mapping information if the file being mapped, or any one of the elements within its I/O stack (if cascade is TRUE), is owned by a library that must be explicitly synchronized.

MAP_OBJECT Function

This function builds the mapping information for the Oracle object identified by the object name, owner, and type.

Syntax

```
DBMS_STORAGE_MAP.MAP_OBJECT(
   objname  IN  VARCHAR2,
   owner    IN  VARCHAR2,
   objtype  IN  VARCHAR2);
```

Parameters

Table 198-8 MAP_OBJECT Function Parameters

Parameter	Description
objname	The name of the object.
owner	The owner of the object.
objtype	The type of the object.

RESTORE Function

This function loads the entire mapping information from the data dictionary into the shared memory of the instance.

You can invoke RESTORE only after a SAVE operation. You must explicitly call RESTORE in a warm startup scenario.

Syntax

```
DBMS_STORAGE_MAP.RESTORE;
```

SAVE Function

This function saves information needed to regenerate the entire mapping into the data dictionary.

Syntax

```
DBMS_STORAGE_MAP.SAVE;
```

UNLOCK_MAP Procedure

This procedure unlocks the mapping information in the shared memory of the instance.

Syntax

```
DBMS_STORAGE_MAP.UNLOCK_MAP;
```