221

DBMS_WORKLOAD_REPLAY

The DBMS_WORKLOAD_REPLAY package provides a interface to replay a workload capture.

This chapter contains the following topics:

> **See Also:**
>
> *Oracle Database Testing Guide* for more information about database replay

DBMS_WORKLOAD_REPLAY Overview

The DBMS_WORKLOAD_REPLAY package provides an interface to replay a workload capture that was originally created by way of the DBMS_WORKLOAD_CAPTURE package.

Typically, the DBMS_WORKLOAD_CAPTURE package is used in the production system to capture a production workload, and the DBMS_WORKLOAD_REPLAY package is subsequently used in a test system to replay the captured production workload for testing purposes.

Related Topics

- DBMS_WORKLOAD_CAPTURE
  The DBMS_WORKLOAD_CAPTURE package configures the Workload Capture system and produce the workload capture data.

DBMS_WORKLOAD_REPLAY Security Model

The security model describes the privileges needed for using DBMS_WORKLOAD_REPLAY.

The following code sample shows the minimum set of privileges required to:

- Create directory objects
- Operate the interface provided by the DBMS_WORKLOAD_CAPTURE and DBMS_WORKLOAD_REPLAY packages
- Act as a replay client user (wrc someuser/somepassword or wrc USER=someuser PASSWORD=somepassword)

```
DROP USER rom1 CASCADE;
CREATE USER rom1 IDENTIFIED BY rom1;

GRANT EXECUTE ON DBMS_WORKLOAD_CAPTURE TO rom1;
GRANT EXECUTE ON DBMS_WORKLOAD_REPLAY TO rom1;
```

```
GRANT CREATE SESSION TO rom1;
GRANT CREATE ANY DIRECTORY TO rom1;
GRANT SELECT_CATALOG_ROLE TO rom1;
GRANT BECOME USER TO rom1;
```

Appropriate OS permissions are required to access and manipulate files and directories on both the capture and replay system. The Oracle process(es) and the OS user performing the capture or replay must be able to access and manipulate at least one common directory accessible from the host where the instance is running.

The replay client is a multithreaded program (an executable named `wrc` located in the `$ORACLE_HOME/bin` directory) where each thread submits a workload from a captured session. The OS user performing the replay must be able to execute `wrc` on hosts that are used for the replay clients and be able to access the file system appropriately to be able to copy the capture to the replay clients' hosts if required.

Summary of DBMS_WORKLOAD_REPLAY Subprograms

This table lists the `DBMS_WORKLOAD_REPLAY` package subprograms in alphabetical order.

Table 221-1 DBMS_WORKLOAD_REPLAY Package Subprograms

Subprogram	Description
ADD_CAPTURE Function	Adds the given capture to the current schedule
ADD_FILTER Procedure	Adds a filter to replay only a subset of the captured workload
ADD_SCHEDULE_ORDERING Function	Adds a schedule order between two captures
ASSIGN_GROUP_TO_INSTANCE Procedure	Modifies the view `DBA_WORKLOAD_GROUP_ASSIGNMENTS`
BEGIN_REPLAY_SCHEDULE Procedure	Initiates the creation of a reusable replay schedule
CALIBRATE Function	Operates on a processed workload capture directory to estimate the number of hosts and workload replay clients needed to faithfully replay the given workload
CANCEL_REPLAY Procedure	Cancels the workload replay in progress
COMPARE_PERIOD_REPORT Procedure	Generates a report comparing a replay to its capture or to another replay of the same capture
COMPARE_SQLSET_REPORT Function	Generates a report comparing a sqlset captured during replay to one captured during workload capture or to one captured during another replay of the same capture
CREATE_FILTER_SET Procedure	Uses the replay filters added to create a set of filters to use against the replay in `replay_dir`
DELETE_FILTER Procedure	Deletes the named filter
DELETE_REPLAY_INFO Procedure	Deletes the rows in `DBA_WORKLOAD_REPLAYS` that corresponds to the given workload replay ID
END_REPLAY_SCHEDULE Procedure	Wraps up the creation of the current schedule
EXPORT_AWR Procedure	Exports the Automatic Workload Repository (AWR) snapshots associated with a given replay ID
GENERATE_CAPTURE_SUBSET Procedure	Creates a new capture from an existing workload capture

Table 221-1 (Cont.) DBMS_WORKLOAD_REPLAY Package Subprograms

Subprogram	Description
GET_DIVERGING_STATEMENT Function	Exports the Automatic Workload Repository (AWR) snapshots associated with a given replay ID
GET_REPLAY_DIRECTORY Function	Returns the current replay directory set by the SET_REPLAY_DIRECTORY Procedure.
GET_REPLAY_INFO Function	Retrieves information about the workload capture and the history of all the workload replay attempts from the related directory
GET_REPLAY_TIMEOUT Procedure	Retrieves the replay timeout setting
IMPORT_AWR Function	Imports the Automatic Workload Repository (AWR) snapshots associated with a given replay ID
INITIALIZE_CONSOLIDATED_REPLAY Procedure	Puts the database state in `INIT` for a multiple-capture replay
INITIALIZE_REPLAY Procedure	Initializes replay, and loads specific data produced during processing into the database
IS_REPLAY_PAUSED Function	Reports whether the replay is currently paused
LOAD_LONG_SQLTEXT Procedure	Loads the captured SQL statements that are longer than 1000 characters to the `DBA_WORKLOAD_LONG_SQLTEXT` view
PAUSE_REPLAY Procedure	Pauses the in-progress workload replay
POPULATE_DIVERGENCE Procedure	Precomputes the divergence information for the given call, stream, or the whole replay so that the GET_DIVERGING_STATEMENT Function returns as quickly as possible for the precomputed calls
PREPARE_CONSOLIDATED_REPLAY Procedure	Puts the database in a special "Prepare" mode for a multiple-capture replay
PREPARE_REPLAY Procedure	Puts the database in a special "Prepare" mode
PROCESS_CAPTURE Procedure	Processes the workload capture found in `capture_dir` in place
REMAP_CONNECTION Procedure	Remaps the captured connection to a new one so that the user sessions can connect to the database in a desired way during workload replay
REMOVE_CAPTURE Procedure	Removes the given capture from the current schedule
REMOVE_SCHEDULE_ORDERING Procedure	Removes an existing schedule order from the current replay schedule
REPORT Function	Generates a report on the given workload replay
RESUME_REPLAY Procedure	Resumes a paused workload replay
REUSE_REPLAY_FILTER_SET Procedure	Reuses filters in the specified filter set as if each were added using the ADD_SCHEDULE_ORDERING Function
SET_ADVANCED_PARAMETER Procedure	Sets an advanced parameter for replay besides the ones used with the PREPARE_REPLAY Procedure
SET_REPLAY_DIRECTORY Procedure	Sets a directory that contains multiple workload captures as the current replay directory
SET_REPLAY_TIMEOUT Procedure	Sets the replay timeout setting
SET_SQL_MAPPING Procedure	Specifies SQL statements to be skipped or replaced during a database replay operation

Table 221-1 (Cont.) DBMS_WORKLOAD_REPLAY Package Subprograms

Subprogram	Description
SET_USER_MAPPING Procedure	Sets a new schema or user name to be used during replay instead of the captured user
START_CONSOLIDATED_REPLAY Procedure	Starts the replay of a multiple-capture capture
START_REPLAY Procedure	Starts the workload replay
USE_FILTER_SET Procedure	Uses the given filter set that has been created by calling the CREATE_FILTER_SET Procedure to filter the current replay

ADD_CAPTURE Function

This function adds the given capture to the current schedule. The directory has to be a valid capture processed in the current database's version. It returns a unique ID that identifies this capture within this schedule.

Syntax

```
DBMS_WORKLOAD_REPLAY.ADD_CAPTURE (
   capture_dir_name      IN    VARCHAR2,
   start_delay_seconds   IN    NUMBER  DEFAULT 0,
   stop_replay           IN    BOOLEAN FALSE,
   take_begin_snapshot   IN    BOOLEAN TRUE,
   take_end_snapshot     IN    BOOLEAN TRUE,
   query_only            IN    BOOLEAN DEFAULT FALSE)
 RETURN NUMBER;

DBMS_WORKLOAD_REPLAY.ADD_CAPTURE (
   capture_dir_name      IN    VARCHAR2,
   start_delay_seconds   IN    NUMBER  DEFAULT 0,
   stop_replay           IN    BOOLEAN FALSE,
   take_begin_snapshot   IN    BOOLEAN TRUE,
   take_end_snapshot     IN    BOOLEAN TRUE,
   query_only            IN    VARCHAR2 DEFAULT 'N')
 RETURN NUMBER;
```

Parameters

Table 221-2 ADD_CAPTURE Function Parameters

Parameter	Description
capture_dir_name	Name of the OS directory containing the capture under the replay top-level directory
start_delay_seconds	Delay time in seconds before the replay of this capture starts
stop_replay	Stop the replay after it finishes
take_begin_snapshot	Take an AWR snapshot when the replay of this capture starts
take_end_snapshot	Take an AWR snapshot when the replay of this capture finishes
query_only	Replay only the read-only queries of this workload capture

Usage Notes

The SET_REPLAY_DIRECTORY Procedure must have already been called.

ADD_FILTER Procedure

This procedure adds a filter to replay only a subset of the captured workload.

The procedure adds a new filter that is used in the next replay filter set created using the CREATE_FILTER_SET Procedure. This filter will be considered an "INCLUSION" or "EXCLUSION" filter depending on the argument passed to CREATE_FILTER_SET when creating the filter set.

Syntax

```
DBMS_WORKLOAD_REPLAY.ADD_FILTER (
   fname           IN VARCHAR2,
   fattribute      IN VARCHAR2,
   fvalue          IN VARCHAR2);

DBMS_WORKLOAD_REPLAY.ADD_FILTER (
   fname           IN VARCHAR2,
   fattribute      IN VARCHAR2,
   fvalue          IN NUMBER);
```

Parameters

Table 221-3 ADD_FILTER Procedure Parameters

Parameter	Description
fname	(Mandatory) Name of the filter. Can be used to delete the filter later if it is not required.
fattribute	(Mandatory) Specifies the attribute on which the filter is defined as one of the following values of type STRING : • USER • MODULE • ACTION • PROGRAM • SERVICE • CONNECTION_STRING
fvalue	(Mandatory) Specifies the value to which the given 'attribute' must be equal to for the filter to be considered active. Wildcards such as '%' are acceptable for all attributes that are of type STRING. Currently all the listed values of fattribute are of type STRING. INSTANCE_NUMBER is a NUMBER attribute. It is currently only supported for capture.

ADD_SCHEDULE_ORDERING Function

This function adds a schedule order between two captures.

Together, schedule_capture_id and waitfor_capture_id form a schedule ordering that previously added by the ADD_SCHEDULE_ORDERING Function. The order is that replay of capture indicated by schedule_capture_id will not start unless the replay of capture indicated by waiting_for_capture_id finishes.

Syntax

```
DBMS_WORKLOAD_REPLAY.ADD_SCHEDULE_ORDERING (
   schedule_capture_id     IN VARCHAR2,
   waitfor_capture_id      IN VARCHAR2)
  RETURN NUMBER;
```

Parameters

Table 221-4 ADD_SCHEDULE_ORDERING Function Parameters

Parameter	Description
schedule_capture_id	Points to a capture that has been added to the current replay schedule. According to the new schedule ordering added by this subprogram, its replay will not start until the replay of another capture specified by waitfor_capture_id runs to completion.
waitfor_capture_id	Points to a capture that has been added to the current replay schedule. According to the new schedule ordering added by this subprogram, the replay of capture specified by schedule_capture_id will not start until the replay of this capture runs to completion.

Return Values

Returns a non-zero error code if the constraint cannot be added

Usage Notes

The two captures must have already been added to the replay schedule.

ASSIGN_GROUP_TO_INSTANCE Procedure

This procedure modifies the view DBA_WORKLOAD_GROUP_ASSIGNMENTS.

Syntax

```
DBMS_WORKLOAD_REPLAY.ASSIGN_GROUP_TO_INSTANCE (
   group_id          IN INTEGER,
   instance_number   IN INTEGER);
```

Parameters

Table 221-5 ASSIGN_GROUP_TO_INSTANCE Procedure Parameters

Parameter	Description
group_id	The identifier of the specified group of capture files
instance_number	The number used for instance registration. It is equivalent to the INSTANCE_NUMBER column in V$INSTANCE.

> **See Also:**
>
> - `DBA_WORKLOAD_GROUP_ASSIGNMENTS` in *Oracle Database Reference*
> - `V$INSTANCE` in *Oracle Database Reference*

BEGIN_REPLAY_SCHEDULE Procedure

This procedure initiates the creation of a reusable replay schedule.

Syntax

```
DBMS_WORKLOAD_REPLAY.BEGIN_REPLAY_SCHEDULE  (
   replay_dir_obj      IN     VARCHAR2,
   schedule_name       IN     VARCHAR2);
```

Parameters

Table 221-6 BEGIN_REPLAY_SCHEDULE Procedure Parameters

Parameter	Description
`replay_dir_obj`	Directory object that points to the replay directory that contains all the capture directories involved in the schedule
`schedule_name`	Name of the schedule to be replayed

Usage Notes

- Only one schedule can be in creation mode at a time. Calling the subprogram again before `end_replay_schedule` will raise an error.
- Prerequisites:
  - The workload capture was already processed using the PROCESS_CAPTURE Procedure in the same database version.
  - The user must have copied the capture directory appropriately.
  - The database is not in replay mode.
  - The SET_REPLAY_DIRECTORY Procedure has already been called.

CALIBRATE Function

This function operates on a processed workload capture directory to estimate the number of hosts and workload replay clients needed to faithfully replay the given workload. This function returns the results as an XML `CLOB`.

Syntax

```
DBMS_WORKLOAD_REPLAY.CALIBRATE (
   capture_dir          IN VARCHAR2,
   process_per_cpu      IN BINARY_INTEGER DEFAULT 4,
   threads_per_process  IN BINARY_INTEGER DEFAULT 50)
  RETURN CLOB;
```

Parameters

Table 221-7 CALIBRATE Function Parameters

Parameter	Description
capture_dir	Name of the directory object that points to the (case sensitive) OS directory that contains processed capture data
process_per_cpu	Maximum number of processes allowed for each CPU (default is 4)
threads_per_process	Maximum number of threads allowed for each process (default is 50)

Return Values

Returns a CLOB formatted as XML that contains:

- Information about the capture
- Current database version
- Input parameters to this function
- Number of CPUs and replay clients needed to replay the given workload
- Information about the sessions captured (total number and maximum concurrency)

Usage Notes

- Prerequisite: The input workload capture was already processed using the PROCESS_CAPTURE Procedure in the same database version.
- This procedure will return the same results as the workload replay client in calibrate mode, which can be run as follows.

```
$ wrc mode=calibrate replaydir=
```

CANCEL_REPLAY Procedure

This procedure cancels workload replay in progress. All the external replay clients (WRC) will automatically be notified to stop issuing the captured workload and exit.

Syntax

```
DBMS_WORKLOAD_REPLAY.CANCEL_REPLAY (
   error_msg    IN VARCHAR2 DEFAULT NULL);
```

Parameters

Table 221-8 CANCEL_REPLAY Procedure Parameters

Parameter	Description
error_msg	An optional reason for cancelling the replay can be passed which is recorded into DBA_WORKLOAD_REPLAYS.ERROR_MESSAGE.

Usage Notes

Prerequisite: A call to the INITIALIZE_REPLAY Procedure, or PREPARE_REPLAY Procedure, or START_REPLAY Procedure was already issued.

COMPARE_PERIOD_REPORT Procedure

This procedure generates a report comparing a replay to its capture or to another replay of the same capture.

Syntax

```
DBMS_WORKLOAD_REPLAY.COMPARE_PERIOD_REPORT (
   replay_id1   IN   NUMBER,
   replay_id2   IN   NUMBER,
   format       IN   VARCHAR2,
   result       OUT  CLOB );
```

Parameters

Table 221-9 COMPARE_PERIOD_REPORT Procedure Parameters

Parameter	Description
replay_id1	First ID of the workload replay whose report is requested
replay_id2	Second ID of the workload replay whose report is requested. If this is NULL, then the comparison is done with the capture.
format	Specifies the report format. Valid values are DBMS_WORKLOAD_CAPTURE.TYPE_HTML and DBMS_WORKLOAD_CAPTURE.TYPE_XML.
result	Output of the report (CLOB)

COMPARE_SQLSET_REPORT Function

This procedure generates a report comparing a sqlset captured during replay to one captured during workload capture or to one captured during another replay of the same capture.

Syntax

```
DBMS_WORKLOAD_REPLAY.COMPARE_SQLSET_REPORT (
   replay_id1    IN NUMBER,
   replay_id2    IN NUMBER,
   format        IN VARCHAR2,
   r_level       IN VARCHAR2 DEFAULT 'ALL',
   r_sections    IN VARCHAR2 DEFAULT 'ALL',
   result        OUT CLOB )
  RETURN VARCHAR2;
```

Parameters

Table 221-10 COMPARE_SQLSET_REPORT Function Parameters

Parameter	Description
replay_id1	First ID of the workload replay after a change
replay_id2	Second ID of the workload replay before a change. If this is NULL, then the comparison is done with the capture.

Table 221-10 (Cont.) COMPARE_SQLSET_REPORT Function Parameters

Parameter	Description
format	Specifies the report format. Valid values are DBMS_WORKLOAD_CAPTURE.TYPE_HTML, DBMS_WORKLOAD_CAPTURE.TYPE_XML and DBMS_WORKLOAD_CAPTURE.TYPE_TEXT.
r_level	See level parameter in the REPORT_ANALYSIS_TASK Function in the DBMS_SQLPA package
r_sections	See section parameter in the REPORT_ANALYSIS_TASK Function in the DBMS_SQLPA package
result	Output of the report (CLOB)

CREATE_FILTER_SET Procedure

This procedure creates a new filter set for the replays at replay_dir.

It includes all the replay filters that have already been added by the ADD_FILTER Procedure. After the procedure has completed and replay initiated, the newly-created filter set can be used to filter the replay in replay_dir by calling the USE_FILTER_SET Procedure.

Syntax

```
DBMS_WORKLOAD_REPLAY.CREATE_FILTER_SET(
   replay_dir       IN  VARCHAR2,
   filter_set       IN  VARCHAR2,
   default_action   IN  VARCHAR2 DEFAULT 'INCLUDE');
```

Parameters

Table 221-11 CREATE_FILTER_SET Procedure Parameters

Parameter	Description
replay_dir	Object directory of the replay to be filtered
filter_set	Name of the filter set to create (to use in USE_FILTER_SET Procedure)
default_action	Can be either INCLUDE or EXCLUDE. Determines whether, by default, every captured call must be replayed or not. Also determines whether the workload filters specified must be considered as INCLUSION filters or EXCLUSION filters.) If it is INCLUDE, then by default all captured calls are replayed, except for the part of the workload defined by the filters. In this case, all the filters that were specified using the ADD_SCHEDULE_ORDERING Function are treated as EXCLUSION filters, and will determine the workload that will not be replayed. If it is EXCLUDE, then by default no captured call to the database is replayed, except for the part of the workload defined by the filters. In this case, all the filters that were specified using he ADD_SCHEDULE_ORDERING Function are treated as INCLUSION filters, and will determine the workload that is replayed. Default: INCLUDE and all the filters specified are assumed to be EXCLUSION filters

Usage Notes

This operation must be invoked when no replay is initialized, prepared, or in progress.

DELETE_FILTER Procedure

This procedure deletes the named filter.

Syntax

```
DBMS_WORKLOAD_REPLAY.DELETE_FILTER(
   fname    IN  VARCHAR2);
```

Parameters

Table 221-12 DELETE_FILTER Procedure Parameters

Parameter	Description
fname	(Mandatory) Name of the filter that must be deleted

Usage Notes

The DELETE_FILTER Procedure only affects filters that have not been used by any previous capture. Consequently, filters can be deleted only if they have been added using the ADD_FILTER Procedures after any capture has been completed. Filters that have been added using ADD_FILTER before a START_CAPTURE and FINISH_CAPTURE cannot be deleted anymore using this subprogram.

DELETE_REPLAY_INFO Procedure

This procedure deletes the rows in DBA_WORKLOAD_REPLAYS that correspond to the given workload replay ID.

Syntax

```
DBMS_WORKLOAD_REPLAY.DELETE_REPLAY_INFO (
   replay_id    IN  NUMBER);
```

Parameters

Table 221-13 DELETE_REPLAY_INFO Procedure Parameters

Parameter	Description
replay_id	(Mandatory) ID of the workload replay that must be deleted. Corresponds to DBA_WORKLOAD_REPLAYS.ID

END_REPLAY_SCHEDULE Procedure

This procedure wraps up the creation of the current schedule. The schedule is now saved and associated with the replay directory and can be used for a replay.

Syntax

```
DBMS_WORKLOAD_REPLAY.END_REPLAY_SCHEDULE;
```

Usage Notes

The BEGIN_REPLAY_SCHEDULE Procedure must have already been called.

EXPORT_AWR Procedure

This procedure exports the AWR snapshots associated with a stipulated replay ID.

Syntax

```
DBMS_WORKLOAD_REPLAY.EXPORT_AWR (
   replay_id    IN  NUMBER);
```

Parameters

Table 221-14 EXPORT_AWR Function Parameters

Parameter	Description
replay_id	(Mandatory) ID of the replay whose AWR snapshots are to be exported

Usage Notes

- At the end of each replay, the corresponding AWR snapshots are automatically exported. Consequently, there is no need to do this manually after a workload replay is complete, unless the automatic EXPORT_AWR invocation failed.
- This procedure will work only if the corresponding workload replay was performed in the current database (meaning that the corresponding row in DBA_WORKLOAD_REPLAYS was not created by calling the GET_REPLAY_INFO Function) and the AWR snapshots that correspond to that replay time period are still available.

GENERATE_CAPTURE_SUBSET Procedure

This procedure creates a new capture from an existing workload capture.

Syntax

```
DBMS_WORKLOAD_REPLAY.GENERATE_CAPTURE_SUBSET (
   input_capture_dir          IN   VARCHAR2,
   output_capture_dir         IN   VARCHAR2,
   new_capture_name           IN   VARCHAR2,
   begin_time                 IN   NUMBER,
   begin_include_incomplete   IN   BOOLEAN DEFAULT TRUE,
   end_time                   IN   NUMBER,
   end_include_incomplete     IN   BOOLEAN DEFAULT FALSE,
   parallel_level             IN   NUMBER DEFAULT NULL);
```

Parameters

Table 221-15 GENERATE_CAPTURE_SUBSET Procedure Parameters

Parameter	Description
input_capture_dir	(Mandatory) Name of the directory object that points to an existing workload capture
output_capture_dir	(Mandatory) Name of the directory object that points to the new capture
new_capture_name	(Mandatory) Name of new capture
begin_time	Start of the time range - time offset in seconds from the start of a workload capture
begin_include_incomplete	Column to include incomplete calls caused by begin_time
end_time	End of the time range - time offset in seconds from the start of a workload capture. If end_time is zero or end_time is less or equal than begin_time, the time range is invalid. The new capture will use the whole duration of the input capture.
end_include_incomplete	Column to include incomplete calls caused by end_time
parallel_level	Number of Oracle processes used to process the input captures in a parallel fashion. The NULL default value will auto-compute the parallelism level based on number of CPUs, whereas a value of 1 will enforce serial execution.

GET_DIVERGING_STATEMENT Function

This function retrieves information about a diverging call, including the statement text, the SQL ID, and the binds. If the replay of a recorded user call has data or error divergence, it is a diverging call.

Syntax

```
DBMS_WORKLOAD_REPLAY.GET_DIVERGING_STATEMENT (
   replay_id    IN NUMBER,
   stream_id    IN NUMBER,
   call_counter IN NUMBER)
  RETURN CLOB;
```

Parameters

Table 221-16 GET_DIVERGING_STATEMENT Function Parameters

Parameter	Description
replay_id	ID of the replay in which that call diverged
stream_id	Stream ID of the diverging call
call_counter	Call counter of the diverging call

Usage Notes

- Returns a CLOB formatted as XML that contains:
  - SQL ID

  - SQL Text
  - Bind information: position, name and value
- This function will silently invoke the POPULATE_DIVERGENCE Procedure to read the information from the capture files. Therefore, if divergence has not been populated, then the first call to this function for a particular diverging call might take longer, especially in very large captures.

GET_REPLAY_DIRECTORY Function

This function returns the current replay directory set by the SET_REPLAY_DIRECTORY Procedure. It returns `NULL` if no replay directory has been set.

Syntax

```
DBMS_WORKLOAD_REPLAY.GET_REPLAY_DIRECTORY
   RETURN VARCHAR2;
```

Related Topics

- SET_REPLAY_DIRECTORY Procedure
  This procedure sets a directory that contains multiple workload captures as the current replay directory.

GET_REPLAY_INFO Function

This function retrieves information about the workload capture and the history of all the workload replay attempts from the stipulated directory.

Syntax

```
DBMS_WORKLOAD_REPLAY.GET_REPLAY_INFO (
   replay_dir    IN VARCHAR2,
   load_details  IN BOOLEAN DFAULT FALSE)
 RETURN NUMBER;
```

Parameters

Table 221-17 GET_REPLAY_INFO Function Parameters

Parameter	Description
`replay_dir`	(Mandatory) Name of the workload replay directory object (case sensitive).
`load_details`	Load the divergence and tracked commits data. The default value is `FALSE`.

Return Values

The procedure returns the `CAPTURE_ID`, which can be associated with both `DBA_WORKLOAD_CAPTURES.ID` and `DBA_WORKLOAD_REPLAYS.CAPTURE_ID` to access the imported information.

Usage Notes

- The procedure first imports a row into DBA_WORKLOAD_CAPTURES which will contain information about the capture. It then imports a row for every replay attempt retrieved from the given replay directory into DBA_WORKLOAD_REPLAYS.
- The procedure will not insert new rows to DBA_WORKLOAD_CAPTURES and DBA_WORKLOAD_REPLAYS if these views already contain rows describing the capture and replay history present in the given directory.

GET_REPLAY_TIMEOUT Procedure

This procedure gets the replay timeout setting.

Syntax

```
DBMS_WORKLOAD_REPLAY.GET_REPLAY_TIMEOUT (
   enabled       OUT  BOOLEAN,
   min_delay     OUT  NUMBER,
   max_delay     OUT  NUMBER,
   delay_factor  OUT  NUMBER);
```

Parameters

Table 221-18 GET_REPLAY_TIMEOUT Procedure Parameters

Parameter	Description
enabled	TRUE if the timeout action is enabled, FALSE otherwise.
min_delay	Lower bound of call delay in minutes. The replay action is activated only when the delay is equal to or more than min_delay.
max_delay	Upper bound of call delay in minutes. The timeout action throws ORA-15569 when the delay is more than max_delay.
delay_factor	Factor for the call delay that is between min_delay and max_delay. The timeout action throws ORA-15569 when the current replay elapsed time is more than the product of capture elapsed time and delay_factor.

Usage Notes

This procedure can be called anytime during replay.

IMPORT_AWR Function

This procedure imports the AWR snapshots from a given replay.

Syntax

```
DBMS_WORKLOAD_REPLAY.IMPORT_AWR (
   replay_id       IN   NUMBER,
   staging_schema  IN   VARCHAR2)
  RETURN NUMBER;
```

Parameters

Table 221-19 IMPORT_AWR Function Parameters

Parameter	Description
replay_id	(Mandatory) ID of the replay whose AWR snapshots must be imported
staging_schema	(Mandatory) Name of a valid schema in the current database which can be used as a staging area while importing the AWR snapshots from the replay directory to the SYS AWR schema. The SYS schema is not a valid input.

Return Values

Returns the new randomly generated database ID that was used to import the AWR snapshots. The same value can be found in the AWR_DBID column in the DBA_WORKLOAD_REPLAYS view.

Usage Notes

- This procedure will work provided those AWR snapshots were exported earlier from the original replay system using the EXPORT_AWR Procedure.
- IMPORT_AWR will fail if the staging_schema provided as input contains any tables with the same name as any of the AWR tables, such as WRM$_SNAPSHOT or WRH$_PARAMETER. Drop any such tables in the staging_schema before invoking IMPORT_AWR.

INITIALIZE_CONSOLIDATED_REPLAY Procedure

This procedure puts the database state in INIT for a multiple-capture replay.

It uses the replay_dir which has already been defined by the SET_REPLAY_DIRECTORY Procedure, pointing to a directory that contains all the capture directories involved in the schedule. It reads data about schedule schedule_name from the directory, and loads required connection data into the replay system.

Syntax

```
DBMS_WORKLOAD_REPLAY.INITIALIZE_CONSOLIDATED_REPLAY (
   replay_name        IN    VARCHAR2,
   schedule_name      IN    VARCHAR2,
   plsql_mode         IN    VARCHAR2 DEFAULT 'TOP_LEVEL');
```

Parameters

Table 221-20 INITIALIZE_CONSOLIDATED_REPLAY Procedure Parameters

Parameter	Description
replay_name	(Mandatory) Name of the workload replay. Every replay of a processed workload capture can be given a name.
schedule_name	Name of the schedule to be replayed. It must have been created through the BEGIN_REPLAY_SCHEDULE Procedure for the replay directory replay_dir.

Table 221-20 (Cont.) INITIALIZE_CONSOLIDATED_REPLAY Procedure Parameters

Parameter	Description
plsql_mode	Specifies the replay options for PL/SQL calls: • TOP_LEVEL — only top-level PL/SQL calls are replayed • EXTENDED — SQL executed from PL/SQL or top-level SQL PL/SQL if there is no SQL recorded inside the PL/SQL are replayed. All captures must have been done in 'EXTENDED' PL/SQL mode.

Usage Notes

Prerequisites:

- Workload capture was already processed using the PROCESS_CAPTURE Procedure in the same database version.
- Database state has been logically restored to what it was at the beginning of the original workload capture.
- The SET_REPLAY_DIRECTORY Procedure has been called.

INITIALIZE_REPLAY Procedure

This procedure puts the database state in INIT for REPLAY mode, and loads data into the replay system that is required before preparing for the replay (by executing the PAUSE_REPLAY Procedure).

Syntax

```
DBMS_WORKLOAD_REPLAY.INITIALIZE_REPLAY (
   replay_name     IN  VARCHAR2,
   replay_dir      IN  VARCHAR2,
   plsql_mode      IN  VARCHAR2 DEFAULT 'TOP_LEVEL',
   rac_inst_list   IN  VARCHAR2 DEFAULT NULL);
```

Parameters

Table 221-21 INITIALIZE_REPLAY Procedure Parameters

Parameter	Description
replay_name	(Mandatory) Name of the workload replay. Every replay of a processed workload capture can be given a name.
replay_dir	Name of the directory object that points to the OS directory (case sensitive) that contains processed capture data
plsql_mode	Specifies the replay options for PL/SQL calls: • TOP_LEVEL — only top-level PL/SQL calls are replayed • EXTENDED — SQL executed from PL/SQL or top-level SQL PL/SQL if there is no SQL recorded inside the PL/SQL are replayed. All captures must have been done in 'EXTENDED' PL/SQL mode.

Table 221-21 (Cont.) INITIALIZE_REPLAY Procedure Parameters

Parameter	Description
rac_inst_list	Specifies a list of Oracle Real Application Clusters (Oracle RAC) instances that will be used for replay. The parameter is a string of instance numbers that are separated by commas. For example: `rac_inst_list='1,3,5'`

Usage Notes

- Prerequisites:
  - Workload capture was already processed using the PROCESS_CAPTURE Procedure in the same database version.
  - Database state has been logically restored to what it was at the beginning of the original workload capture.
- The subprogram loads data into the replay system that is required before preparing for the replay by calling the PAUSE_REPLAY Procedure.

  For instance, during capture the user may record the connection string each session used to connect to the server. The INITIALIZE_REPLAY Procedure loads this data and allows the user to re-map the recorded connection string to new connection strings or service points.

  Elaborating on the example described in the PROCESS_CAPTURE Procedure, the user could invoke the following:

  ```
  DBMS_WORKLOAD_REPLAY.INITIALIZE_REPLAY('replay foo #1', 'rec_dir');
  ```

  This command will load up the connection map and by default will set all replay time connection strings to be equal to `NULL`. A `NULL` replay time connection string means that the workload replay clients (WRCs) will connect to the default host as determined by the replay client's runtime environment settings. The user can change a particular connection string to a new one (or a new service point) for replay by using the REMAP_CONNECTION Procedure.
- For encrypted capture, the `INITIALIZE_REPLAY` Procedure relies on Oracle wallet. The identifier is `oracle.rat.database_replay.encryption` (case-sensitive).

Related Topics

- PAUSE_REPLAY Procedure
  This procedure pauses the in-progress workload replay.
- INITIALIZE_REPLAY Procedure
  This procedure puts the database state in `INIT` for `REPLAY` mode, and loads data into the replay system that is required before preparing for the replay (by executing the PAUSE_REPLAY Procedure).
- PROCESS_CAPTURE Procedure
  This procedure processes the workload capture found in `capture_dir` in place.
- REMAP_CONNECTION Procedure
  This procedure remaps the captured connection to a new one so that the user sessions can connect to the database in a desired way during workload replay.

IS_REPLAY_PAUSED Function

This function reports whether the replay is currently paused.

Syntax

```
DBMS_WORKLOAD_REPLAY.IS_REPLAY_PAUSED
   RETURN BOOLEAN;
```

Return Values

Returns `TRUE` if the PAUSE_REPLAY Procedure has been called successfully and the RESUME_REPLAY Procedure has not been called yet.

Usage Notes

A call to the START_REPLAY Procedure must have already been issued as a prerequisite.

LOAD_LONG_SQLTEXT Procedure

This procedure loads the captured SQL statements that are longer than 1000 characters to the `DBA_WORKLOAD_LONG_SQLTEXT` view.

Syntax

```
DBMS_WORKLOAD_REPLAY.LOAD_LONG_SQLTEXT (
   capture_id     IN  NUMBER);
```

Parameters

Table 221-22 LOAD_LONG_SQLTEXT Procedure Parameters

Parameter	Description
capture_id	Internal key for the workload capture

> **Note:**
>
> This procedure is available starting with Oracle Database Release 18c.

> **See Also:**
>
> `DBA_WORKLOAD_LONG_SQLTEXT` in *Oracle Database Reference*

PAUSE_REPLAY Procedure

This procedure pauses the in-progress workload replay.

All subsequent user calls from the replay clients will be stalled until either a call to the RESUME_REPLAY Procedure is issued or the replay is cancelled.

Syntax

```
DBMS_WORKLOAD_REPLAY.PAUSE_REPLAY;
```

Usage Notes

- Prerequisite: A call to the START_REPLAY Procedure must have already been issued.
- User calls that were already in-progress when this procedure was invoked are allowed to run to completion. Only subsequent user calls, when issued, are paused.

POPULATE_DIVERGENCE Procedure

This procedure precomputes the divergence information for the given call, stream, or the whole replay so that the GET_DIVERGING_STATEMENT Function returns as quickly as possible for the precomputed calls.

Syntax

```
DBMS_WORKLOAD_REPLAY.POPULATE_DIVERGENCE (
   replay_id    IN   NUMBER,
   stream_id    IN   NUMBER DEFAULT NULL,
   call_counter IN   NUMBER DEFAULT NULL);
```

Parameters

Table 221-23 POPULATE_DIVERGENCE Procedure Parameters

Parameter	Description
replay_id	ID of the replay
stream_id	Stream ID of the diverging call. If NULL is provided, then divergence information is precomputed for all diverging calls in the given replay.
call_counter	Call counter of the diverging call. If NULL is provided, then divergence information is precomputed for all diverging calls in the given stream.

Related Topics

- GET_DIVERGING_STATEMENT Function
  This function retrieves information about a diverging call, including the statement text, the SQL ID, and the binds. If the replay of a recorded user call has data or error divergence, it is a diverging call.

PREPARE_CONSOLIDATED_REPLAY Procedure

Similar to the PREPARE_REPLAY Procedure, this procedure puts the database in a special "Prepare" mode for a multiple-capture replay. The difference is that this subprogram should be used only for consolidated replays.

Syntax

```
DBMS_WORKLOAD_REPLAY.PREPARE_CONSOLIDATED_REPLAY (
   synchronization           IN BOOLEAN,
   connect_time_scale        IN NUMBER    DEFAULT 100,
   think_time_scale          IN NUMBER    DEFAULT 100,
   think_time_auto_correct   IN BOOLEAN   DEFAULT TRUE,
   capture_sts               IN BOOLEAN   DEFAULT FALSE,
```

```
   sts_cap_interval          IN NUMBER    DEFAULT 300);

DBMS_WORKLOAD_REPLAY.PREPARE_CONSOLIDATED_REPLAY (
   synchronization           IN VARCHAR2  DEFAULT 'SCN',
   connect_time_scale        IN NUMBER    DEFAULT 100,
   think_time_scale          IN NUMBER    DEFAULT 100,
   think_time_auto_correct   IN BOOLEAN   DEFAULT TRUE,
   capture_sts               IN BOOLEAN   DEFAULT FALSE,
   sts_cap_interval          IN NUMBER    DEFAULT 300);
```

Parameters

Table 221-24 PREPARE_CONSOLIDATED_REPLAY Procedure Parameters

Parameter	Description
`synchronization`	Sets the synchronization mode for replay: • `'TIME'` — The synchronization will be based on the time the action took place during capture (clock-based time). • `'SCN'` — The synchronization will be based on the capture-time commits; the commit order will be preserved during replay. This is the default mode. • `'OBJECT_ID'` — Every replayed action will be executed only *after* the relevant commits have finished execution. The relevant commits are those that were issued before the given action in the captured workload and that modified at least one of the database objects the given action is referencing (either implicitly or explicitly). This synchronization mode makes sure that any replay action will see the same data that the action saw during capture, but allows greater concurrency for the actions that do not touch the same objects/tables. This value is deprecated. For compatibility, an overloaded version of this procedure uses BOOLEAN for this parameter: • TRUE means `'SCN'` • FALSE means `'TIME'`
`connect_time_scale`	Scales the time elapsed between the instant the workload capture was started and the session connects with the given value. The input is interpreted as a % value. Can potentially be used to increase or decrease the number of concurrent users during the workload replay. `DEFAULT VALUE` is 100. See "Example 221-1".
`think_time_scale`	Scales the time elapsed between two successive user calls "Example 221-1"from the same session. The input is interpreted as a % value. Can potentially be used to increase or decrease the number of concurrent users during the workload replay. `DEFAULT VALUE` is 100. See "Example 221-2".
`think_time_auto_correct`	Auto corrects the think time between calls appropriately when user calls takes longer to complete during replay than during the original capture. `DEFAULT` is `TRUE` which is to reduce think time if replay goes slower than capture. See "Example 221-3".

Table 221-24 (Cont.) PREPARE_CONSOLIDATED_REPLAY Procedure Parameters

Parameter	Description
capture_sts	If this parameter is TRUE, then a SQL tuning set capture is also started in parallel with workload replay. The resulting SQL tuning set can be exported using the EXPORT_AWR Procedure along with the Automatic Workload Repository (AWR) data. Currently, parallel SQL tuning set (STS) capture is not supported in an Oracle RAC environment. So, this parameter has no effect in that context. The calling user must have the appropriate privileges ('ADMINISTER SQL TUNING SET'). The default value is FALSE.
sts_cap_interval	Specifies the capture interval of the SQL set capture from the cursor cache in seconds. The default value is 300.

Usage Notes

A consolidated replay replays multiple captures in one replay. Each capture records different system change number (SCN) values. For this reason SCN-based sync is not supported for consolidated replays. Consolidated replays only support non-sync mode and the Object-ID based synchronization, and SCN-based synchronization is currently not supported.

Related Topics

- PREPARE_REPLAY Procedure
  This procedure puts the database state in PREPARE FOR REPLAY mode.

PREPARE_REPLAY Procedure

This procedure puts the database state in PREPARE FOR REPLAY mode.

Syntax

```
DBMS_WORKLOAD_REPLAY.PREPARE_REPLAY (
   synchronization           IN BOOLEAN,
   connect_time_scale        IN NUMBER    DEFAULT 100,
   think_time_scale          IN NUMBER    DEFAULT 100,
   think_time_auto_correct   IN BOOLEAN   DEFAULT TRUE,
   scale_up_multiplier       IN NUMBER    DEFAULT 1,
   capture_sts               IN BOOLEAN   DEFAULT FALSE,
   sts_cap_interval          IN NUMBER    DEFAULT 300,
   rac_mode                  IN NUMBER    DEFAULT GLOBAL_SYNC,
   query_only                IN BOOLEAN   DEFAULT FALSE);

DBMS_WORKLOAD_REPLAY.PREPARE_REPLAY (
   synchronization           IN VARCHAR2  DEFAULT 'SCN',
   connect_time_scale        IN NUMBER    DEFAULT 100,
   think_time_scale          IN NUMBER    DEFAULT 100,
   think_time_auto_correct   IN BOOLEAN   DEFAULT TRUE,
   scale_up_multiplier       IN NUMBER    DEFAULT 1,
   capture_sts               IN BOOLEAN   DEFAULT FALSE,
   sts_cap_interval          IN NUMBER    DEFAULT 300),
   rac_mode                  IN NUMBER    DEFAULT GLOBAL_SYNC,
   query_only                IN BOOLEAN   DEFAULT FALSE);
```

Parameters

Table 221-25 PREPARE_REPLAY Procedure Parameters

Parameter	Description
synchronization	Sets the synchronization mode for replay: • 'TIME' — The synchronization will be based on the time the action took place during capture (clock-based time). • 'SCN' — The synchronization will be based on the capture-time commits; the commit order will be preserved during replay. This is the default mode. • 'OBJECT_ID' — Every replayed action will be executed only *after* the relevant commits have finished execution. The relevant commits are those that were issued before the given action in the captured workload and that modified at least one of the database objects the given action is referencing (either implicitly or explicitly). This synchronization mode makes sure that any replay action will see the same data that the action saw during capture, but allows greater concurrency for the actions that do not touch the same objects/tables. This value is deprecated. For compatibility, an overloaded version of this procedure uses BOOLEAN for this parameter: • TRUE means 'SCN' • FALSE means 'TIME'
connect_time_scale	Scales the time elapsed between the instant the workload capture was started and the session connects with the given value. The input is interpreted as a % value. Can potentially be used to increase or decrease the number of concurrent users during the workload replay. DEFAULT VALUE is 100. See "Example 221-1".
think_time_scale	Scales the time elapsed between two successive user calls from the same session. The input is interpreted as a % value. Can potentially be used to increase or decrease the number of concurrent users during the workload replay. DEFAULT VALUE is 100. See "Example 221-2".
think_time_auto_correct	Auto corrects the think time between calls appropriately when a user call takes longer to complete during replay than during the original capture. DEFAULT is TRUE which is to reduce think time if replay goes slower than capture. See "Example 221-3".
scale_up_multiplier	Defines the number of times the query workload is scaled up during replay. Each captured session is replayed concurrently as many times as the value of the scale_up_multiplier. However, only one of the sessions in each set of identical replay sessions executes both queries and updates. The remaining sessions only execute queries.
capture_sts	If this parameter is TRUE, then a SQL tuning set capture is also started in parallel with workload replay. The resulting SQL tuning set can be exported using the EXPORT_AWR Procedure along with the AWR data. Currently, parallel SQL tuning set (STS) capture is not supported in an Oracle RAC environment. So, this parameter has no effect in that context. The calling user must have the appropriate privileges ('ADMINISTER SQL TUNING SET'). The default value is FALSE.
sts_cap_interval	Specifies the capture interval of the SQL set capture from the cursor cache in seconds. The default value is 300.

Table 221-25 (Cont.) PREPARE_REPLAY Procedure Parameters

Parameter	Description
rac_mode	Specifies replay options in an Oracle Real Application Cluster (Oracle RAC) environment. This parameter accepts the following values: • GLOBAL_SYNC — Synchronization across all Oracle RAC instances. This is the default value. • PER_INSTANCE_CLIENT — Synchronization is global and each WRC client handles part of the workload that is sent to only one instance. • PER_INSTANCE_SYNC — Synchronization is local (within each instance only) and each WRC client handles part of the workload that is sent to only one instance
query_only	Replays only the read-only queries of the workload capture. The default value is FALSE.

Usage Notes

- Prerequisites:
  - The database has been initialized for replay using the INITIALIZE_REPLAY Procedure.
  - Any capture time connection strings that require remapping have been already done using the REMAP_CONNECTION Procedure.
- One or more external replay clients (WRC) can be started once the PREPARE_REPLAY procedure has been executed.
- With regard to scale_up_multiplier:
  - One replay session (base session) of each set of identical sessions will replay every call from the capture as usual.
  - The remaining sessions (scale-up sessions) will only replay calls that are read-only. Thus, DDL, DML, and PL/SQL calls that modified the database is skipped. SELECT FOR UPDATE statements are also skipped.
  - Read-only calls from the scale-up are synchronized appropriately and obey the timings defined by think_time_scale, connect_time_scale, and think_time_auto_correct. Also, the queries are made to wait for the appropriate commits.
  - No replay data or error divergence records are generated for the scale-up sessions.
  - All base or scale-up sessions that replay the same capture file will connect from the same workload replay client.

Example 221-1 Application of the connect_time_scale Parameter

If the following was observed during the original workload capture:

```
12:00 : Capture was started
12:10 : First session connect  (10m after)
12:30 : Second session connect (30m after)
12:42 : Third session connect  (42m after)
```

If the connect_time_scale is 50, then the session connects will happen as follows:

```
12:00 : Replay was started with 50% connect time scale
12:05 : First session connect  ( 5m after)
```

```
12:15 : Second session connect (15m after)
12:21 : Third session connect  (21m after)
```

If the connect_time_scale is 200, then the session connects will happen as follows:

```
12:00 : Replay was started with 200% connect time scale
12:20 : First session connect  (20m after)
13:00 : Second session connect (60m after)
13:24 : Third session connect  (84m after)
```

Example 221-2 Application of the think_time_scale Parameter

If the following was observed during the original workload capture:

```
12:00 : User SCOTT connects
12:10 : First user call issued (10m after completion of prevcall)
12:14 : First user call completes in 4mins
12:30 : Second user call issued (16m after completion of prevcall)
12:40 : Second user call completes in 10m
12:42 : Third user call issued ( 2m after completion of prevcall)
12:50 : Third user call completes in 8m
```

If the think_time_scale is 50 during the workload replay, then the user calls will look something like below:

```
12:00 : User SCOTT connects
12:05 : First user call issued 5 mins (50% of 10m) after the completion of
        previous call
12:10 : First user call completes in 5m (takes a minute longer)
12:18 : Second user call issued 8 mins (50% of 16m) after the completion of prev
        call
12:25 : Second user call completes in 7m (takes 3 minutes less)
12:26 : Third user call issued 1 min  (50% of 2m) after the completion of prev
        call
12:35 : Third user call completes in 9m (takes a minute longer)
```

Example 221-3 Application of the think_time_auto_correct Parameter

If the following was observed during the original workload capture:

```
12:00 : User SCOTT connects
12:10 : First user call issued (10m after completion of prevcall)
12:14 : First user call completes in 4m
12:30 : Second user call issued (16m after completion of prevcall)
12:40 : Second user call completes in 10m
12:42 : Third user call issued ( 2m after completion of prevcall)
12:50 : Third user call completes in 8m
```

If the think_time_scale is 100 and the think_time_auto_correct is TRUE during the workload replay, then the user calls will look something like below:

```
12:00 : User SCOTT connects
12:10 : First user call issued 10 mins after the completion of prev call
12:15 : First user call completes in 5m (takes 1 minute longer)
12:30 : Second user call issued 15 mins (16m minus the extra time of 1m the prev
        call took) after the completion of prev call
12:44 : Second user call completes in 14m (takes 4 minutes longer)
12:44 : Third user call issued immediately (2m minus the extra time of 4m the prev
        call took) after the completion of prev call
12:52 : Third user call completes in 8m
```

PROCESS_CAPTURE Procedure

This procedure processes the workload capture found in `capture_dir` in place.

Syntax

```
DBMS_WORKLOAD_REPLAY.PROCESS_CAPTURE (
   capture_dir            IN   VARCHAR2,
   parallel_level         IN   NUMBER DEFAULT NULL,
   synchronization        IN   VARCHAR2 DEFAULT 'SCN',
   plsql_mode             IN   VARCHAR2 DEFAULT 'TOP_LEVEL');
```

Parameters

Table 221-26 PROCESS_CAPTURE Procedure Parameters

Parameter	Description
`capture_dir`	(Mandatory) Name of the workload capture directory object (case sensitive). The directory object must point to a valid OS directory that has the appropriate permissions. New files are added to this directory.
`parallel_level`	Number of Oracle processes used to process the capture in parallel. The `NULL` default value will auto-compute the parallelism level, whereas a value of 1 will enforce serial execution.
`synchronization`	Determines the synchronization mode that the user will be able to use for replay: • `'TIME'` — When `'TIME'` is selected, the replay can use `'TIME'` synchronization mode only. When `'TIME'` synchronization mode is used for replay, the synchronization will be based on the time the action took place during capture (clock-based time). • `'SCN'` — When `'SCN'` is selected, the replay can use the `'TIME'` or `'SCN'` synchronization mode. This is the default. When `'SCN'` synchronization mode is used for replay, the synchronization will be based on the capture-time commits; the commit order will be preserved during replay. This is the default mode. • `'OBJECT_ID'` — When `'OBJECT_ID'` is selected, replay can use the `'TIME'`, `'SCN'`, or `'OBJECT_ID'` synchronization mode. When `'OBJECT_ID'` synchronization mode is used for replay, every replayed action will be executed only *after* the relevant commits have finished execution. The relevant commits are those that were issued before the given action in the captured workload and that modified at least one of the database objects the given action is referencing (either implicitly or explicitly). This synchronization mode makes sure that any replay action will see the same data that the action saw during capture, but allows greater concurrency for the actions that do not touch the same objects/tables. This synchronization mode is deprecated.

Table 221-26 (Cont.) PROCESS_CAPTURE Procedure Parameters

Parameter	Description
plsql_mode	Specifies the processing mode for PL/SQL: • 'TOP_LEVEL' — metadata is generated for top-level PL/SQL calls only; 'TOP_LEVEL' will be the only option for replay. • 'EXTENDED' — metadata is generated for both top-level PL/SQL calls and the SQL called from PL/SQL. A new directory ppe_X.X.X.X (where the Xs represent the current Oracle version) is created under the capture root directory. Capture must have been done with this same value for the plsql_mode parameter. Replay can use either 'TOP_LEVEL' or 'EXTENDED'.

Usage Notes

- This subprogram analyzes the workload capture found in the capture_dir and creates new workload replay specific metadata files that are required to replay the given workload capture. It only creates new files and does not modify any files that were originally created during the workload capture. Therefore, this procedure can be run multiple times on the same capture directory, such as when the procedure encounters unexpected errors or is cancelled by the user.
- Once this procedure runs successfully, the capture_dir can be used as input to the INITIALIZE_REPLAY Procedure in order to replay the captured workload present in capture_dir.
- Before a workload capture can be replayed in a particular database version, the capture must be processed using PROCESS_CAPTURE in the same database version. Once created, a processed workload capture can be used to replay the captured workload multiple times in the same database version.

  For example, suppose workload "foo" was captured in rec_dir in Oracle database version 10.2.0.5. In order to replay the workload "foo" in version 11.1.0.1 the workload must be processed in version 11.1.0.1. The following procedure must be executed in an 11.1.0.1 database in order to process the capture directory rec_dir:

  ```
  DBMS_WORKLOAD_REPLAY.PROCESS_CAPTURE('rec_dir');
  ```

  Now, rec_dir contains a valid 11.1.0.1 processed workload capture that can be used to replay the workload "foo" in 11.1.0.1 databases as many times as required.
- For encrypted capture, the PROCESS_CAPTURE procedure relies on Oracle wallet. The identifier is oracle.rat.database_replay.encryption (case-sensitive).

REMAP_CONNECTION Procedure

This procedure remaps the captured connection to a new one so that the user sessions can connect to the database in a desired way during workload replay.

Syntax

```
DBMS_WORKLOAD_REPLAY.REMAP_CONNECTION (
   connection_id         IN  NUMBER,
   replay_connection     IN  VARCHAR2);

DBMS_WORKLOAD_REPLAY.REMAP_CONNECTION (
   capture_number        IN  VARCHAR2,
```

```
   connection_id         IN  NUMBER,
   replay_connection     IN  VARCHAR2);
```

Parameters

Table 221-27 REMAP_CONNECTION Procedure Parameters

Parameter	Description
capture_number	Pointing to a capture of the current replay schedule
connection_id	ID of the connection to be remapped. Corresponds to DBA_WORKLOAD_CONNECTION_MAP.CONN_ID.
replay_connection	New connection string to be used during replay

Usage Notes

- Prior to calling REMAP_CONNECTION all replay connection strings are set to NULL by default. If a replay_connection is NULL, then the replay sessions will connect as determined by the replay client's runtime environment. For example, if the environment variable TNS_ADMIN is defined and the user does not call the REMAP_CONNECTION Procedure, then the wrc executable will connect to the server specified in the tnsnames.ora file pointed to by TNS_ADMIN.
- A valid replay_connection must specify a connect identifier or a service point. See the *Oracle Database Net Services Reference* for ways to specify connect identifiers (such as net service names, database service names, and net service aliases) and naming methods that can be used to resolve a connect identifier to a connect descriptor.
- An error is returned if no row matches the given connection_id.
- Use the DBA_WORKLOAD_CONNECTION_MAP view to review all the connection strings that are used by the subsequent workload replay, and also to examine connection string remappings used for previous workload replays.

REMOVE_CAPTURE Procedure

This procedure removes the given capture from the current schedule.

Syntax

```
DBMS_WORKLOAD_REPLAY.REMOVE_CAPTURE (
   schedule_capture_number    IN     NUMBER);
```

Parameters

Table 221-28 REMOVE_CAPTURE Procedure Parameters

Parameter	Description
schedule_capture_number	Unique ID that identifies this capture within this schedule

REMOVE_SCHEDULE_ORDERING Procedure

This procedure removes an existing schedule order from the current replay schedule.

Together, `schedule_capture_id` and `waitfor_capture_id` form a schedule ordering that previously added by the ADD_SCHEDULE_ORDERING Function (`schedule_capture_id`, `waitfor_capture_id`). The order is that replay of capture indicated by `schedule_capture_id` will not start unless the replay of capture indicated by `waiting_for_capture_id` finishes.

Syntax

```
DBMS_WORKLOAD_REPLAY.REMOVE_SCHEDULE_ORDERING (
   schedule_capture_id     IN         NUMBER,
   waitfor_capture_id      IN         NUMBER);
```

Parameters

Table 221-29 REMOVE_SCHEDULE_ORDERING Procedure Parameters

Parameter	Description
`schedule_capture_id`	Points to a capture that has been added to the current replay schedule (see procedure description).
`waitfor_capture_id`	Points to a capture that has been added to the current replay schedule.

Usage Notes

Prerequisites:

- The BEGIN_REPLAY_SCHEDULE Procedure must have been called.
- The replay schedule order should have already been added using the ADD_SCHEDULE_ORDERING Function.

REPORT Function

This function generates a report on the stipulated workload replay.

Syntax

```
DBMS_WORKLOAD_REPLAY.REPORT (
   replay_id          IN NUMBER,
   format             IN VARCHAR2)
  RETURN CLOB;
```

Parameters

Table 221-30 REPORT Function Parameters

Parameter	Description
`replay_id`	(Mandatory) Specifies the ID of the workload replay whose report is requested.

Table 221-30 (Cont.) REPORT Function Parameters

Parameter	Description
format	(Mandatory) Specifies the report format. Valid values: • HTML - Generates the HTML version of the report • XML - Generates the XML version of the report • TEXT - Generates the text version of the report

Return Values

The report body in the desired format returned as a CLOB

RESUME_REPLAY Procedure

This procedure resumes a paused workload replay.

Syntax

```
DBMS_WORKLOAD_REPLAY.RESUME_REPLAY;
```

Usage Notes

Prerequisite: A call to the PAUSE_REPLAY Procedure must have already been issued.

REUSE_REPLAY_FILTER_SET Procedure

This procedure reuses filters in the specified filter set as if each were added using the ADD_SCHEDULE_ORDERING Function.

Each call adds one filter set, which is a collection of individual filters on various attributes. Also, a new filter rule can be added, and an existing filter can be deleted before invoking the CREATE_FILTER_SET Procedure to create a new filter set.

Syntax

```
DBMS_WORKLOAD_REPLAY.REUSE_REPLAY_FILTER_SET(
   replay_dir  IN VARCHAR2,
   filter_set  IN VARCHAR2);
```

Parameters

Table 221-31 REUSE_REPLAY_FILTER_SET Procedure Parameters

Parameter	Description
replay_dir	Capture ID of the existing filter set with which it is associated
filter_set	Name of the filter set to be reused

Related Topics

- ADD_SCHEDULE_ORDERING Function
  This function adds a schedule order between two captures.

SET_ADVANCED_PARAMETER Procedure

This procedure sets an advanced parameter for replay besides the ones used with the PREPARE_REPLAY Procedure.

The advanced parameters control aspects of the replay that are more specialized. The advanced parameters are reset to their default values after the replay has finished.

Syntax

```
DBMS_WORKLOAD_REPLAY.SET_ADVANCED_PARAMETER(
   pname    IN   VARCHAR2,
   pvalue   IN   VARCHAR2);

DBMS_WORKLOAD_REPLAY.SET_ADVANCED_PARAMETER(
   pname    IN   VARCHAR2,
   pvalue   IN   NUMBER);

DBMS_WORKLOAD_REPLAY.SET_ADVANCED_PARAMETER(
   pname    IN   VARCHAR2,
   pvalue   IN   BOOLEAN);
```

Parameters

Table 221-32 SET_ADVANCED_PARAMETER Procedure Parameters

Parameter	Description
pname	Name of the parameter (case insensitive)
pvalue	Value of the parameter

Usage Notes

The current parameters and values that can be used are:

```
'DO_NO_WAIT_COMMITS': (default: FALSE)
```

This parameter controls whether the `COMMIT` issued by replay sessions is `NOWAIT`. The default value for this parameter is `FALSE`. In this case all the `COMMIT`s are issued with the mode they were captured (`wait`, `no-wait`, `batch`, `no-batch`). If the parameter is set to `TRUE`, then all `COMMIT`s are issued in `no-wait` mode. This is useful in cases where the replay is becoming noticeably slow because of a high volume of concurrent `COMMIT`s. Setting the parameter to `TRUE` will significantly decrease the waits on the `'log file sync'` event during the replay with respect to capture.

Related Topics

- PREPARE_REPLAY Procedure
  This procedure puts the database state in `PREPARE FOR REPLAY` mode.

SET_REPLAY_DIRECTORY Procedure

This procedure sets a directory that contains multiple workload captures as the current replay directory.

Syntax

```
DBMS_WORKLOAD_REPLAY.SET_REPLAY_DIRECTORY (
   replay_dir    IN    VARCHAR2);
```

Parameters

Table 221-33 SET_REPLAY_DIRECTORY Procedure Parameters

Parameter	Description
replay_dir	Name of the OS directory containing the captures for a workload consolidation

SET_REPLAY_TIMEOUT Procedure

This procedure sets the replay timeout setting. The purpose is to abort user calls that might make the replay much slower or even cause a replay hang.

Syntax

```
DBMS_WORKLOAD_REPLAY.SET_REPLAY_TIMEOUT (
   enabled       OUT  BOOLEAN DEFAULT TRUE,
   min_delay     OUT  NUMBER DEFAULT 10,
   max_delay     OUT  NUMBER DEFAULT 120,
   delay_factor  OUT  NUMBER DEFAULT 8);
```

Parameters

Table 221-34 SET_REPLAY_TIMEOUT Procedure Parameters

Parameter	Description
enabled	TRUE to enable the timeout action, and FALSE to disable.
min_delay	Lower bound of call delay in minutes. The replay action is activated only when the delay is equal to or more than min_delay. Default = 10.
max_delay	Upper bound of call delay in minutes. The timeout action throws ORA-15569 when the delay is more than max_delay. Default = 120.
delay_factor	Factor for the call delay that is between min_delay and max_delay. The timeout action throws ORA-15569 when the current replay elapsed time is more than the product of capture elapsed time and delay_factor. Default = 8.

Usage Notes

- This procedure can be called anytime during replay.
- Call delay is defined as the difference between replay and capture if replay elapsed time is longer than call elapsed time.

- Once a replay timeout action is enabled, a user call will exit with `ORA-15569` if it has been delayed more than the condition specified by the replay action. The call and its error are reported as error divergence.
- Replay timeout operates as follows:
  - The timeout action has no effect if it is not enabled.
  - If the call delay in minutes is less than a lower bound specified by parameter `min_delay`, then the timeout action is non-operational.
  - If the delay in minutes is more than a upper bound specified by parameter `max_delay`, the timeout action will abort the user call and throw `ORA-15569`.
  - For delay that is between the lower bound and upper bound, the user call will abort with `ORA-15569` only when the current replay elapsed time is more than the product of capture elapsed time and parameter `delay_factor`.

SET_SQL_MAPPING Procedure

This procedure specifies SQL statements to be skipped or replaced during a database replay operation.

Syntax

```
PROCEDURE SET_SQL_MAPPING (
   schedule_cap_id      IN NUMBER,
   sql_id               IN VARCHAR2,
   operation            IN VARCHAR2,
   replacement_sql_text IN VARCHAR2 DEFAULT NULL);

PROCEDURE SET_SQL_MAPPING (
   sql_id               IN VARCHAR2,
   operation            IN VARCHAR2,
   replacement_sql_text IN VARCHAR2 DEFAULT NULL);
```

Parameters

Table 221-35 SET_SQL_MAPPING Procedure Parameters

Parameter	Description
`schedule_cap_id`	ID of a capture in the schedule
`sql_id`	SQL identifier of the SQL statement at the time of capture
`operation`	Directs that one of the following actions be performed for the specified statement during database replay: • `'SKIP'` — Skip the SQL statement identified by `sql_id` • `'REPLACE'` — Replace this SQL statement identified by `sql_id` with the SQL statement in the `replacement_sql_text` parameter

Usage Notes

- `replacement_sql_text`: When `'SKIP'` is specified for the `operation` parameter, this parameter is NULL. When `'REPLACE'` is specified for the `operation` parameter, this parameter's value is the SQL statement to be used.
- `schedule_cap_id` is used for consolidated replay.

SET_USER_MAPPING Procedure

This procedure sets a new schema or user name to be used during replay instead of the captured user.

Syntax

```
DBMS_WORKLOAD_REPLAY.SET_USER_MAPPING (
   schedule_cap_id      IN NUMBER,
   capture_user         IN VARCHAR2,
   replay_user          IN VARCHAR2);

DBMS_WORKLOAD_REPLAY.SET_USER_MAPPING (
   capture_user         IN VARCHAR2,
   replay_user          IN VARCHAR2);
```

Parameters

Table 221-36 SET_USER_MAPPING Procedure Parameters

Parameter	Description
schedule_cap_id	ID of the a capture in the schedule
capture_user	User name during the time of the workload capture
replay_user	User name to which captured user is remapped during replay.

Usage Notes

- A `schedule_cap_id` of `NULL` is used for regular non-consolidate replay.
- The replay must be initialized but not prepared in order to use this subprogram.
- If `replay_user` is set to `NULL`, then the mapping is disabled.
- After multiple calls with the same `capture_user`, the last call always takes effect.
- To list all the mappings that will be in effect during the subsequent replay execute the following:

  ```
  SELECT * FROM DBA_WORKLOAD_ACTIVE_USER_MAP
  ```

- The overloaded version without the `schedule_cap_id` calls the one with the `schedule_cap_id` argument by passing in `NULL`.
- Mappings are stored in a table made public through the view `DBA_WORKLOAD_USER_MAP`. To remove old mappings execute

  ```
  DELETE * FROM DBA_WORKLOAD_USER_MAP
  ```

START_CONSOLIDATED_REPLAY Procedure

This procedure starts the replay of a multiple-capture capture. It should be used only for consolidated replays.

Syntax

```
DBMS_WORKLOAD_REPLAY.START_CONSOLIDATED_REPLAY;
```

Usage Notes

Prerequisites:

- The call to the PREPARE_REPLAY Procedure was already issued.
- A sufficient number of external replay clients (WRC) that can faithfully replay the captured workload already started. The status of such external replay clients can be monitored using `V$WORKLOAD_REPLAY_CLIENTS`.

START_REPLAY Procedure

This procedure starts the workload replay.

All the external replay clients (WRC) that are currently connected to the replay database will automatically be notified, and those replay clients (WRC) will begin issuing the captured workload. It should only be used for consolidated replays.

Syntax

```
DBMS_WORKLOAD_REPLAY.START_REPLAY;
```

Usage Notes

- Prerequisites:
  - The call to the PREPARE_REPLAY Procedure was already issued.
  - A sufficient number of external replay clients (WRC) that can faithfully replay the captured workload already started. The status of such external replay clients can be monitored using `V$WORKLOAD_REPLAY_CLIENTS`.
- Use the WRC's `CALIBRATE` mode to determine the number of replay clients that might be required to faithfully replay the captured workload. For example:

  ```
  $ wrc mode=calibrate replaydir=.
  ```

USE_FILTER_SET Procedure

This procedure applies a filter set to a capture in the current replay schedule.

The filter set must have been created by calling the CREATE_FILTER_SET Procedure.

Syntax

```
DBMS_WORKLOAD_REPLAY.USE_FILTER_SET(
   capture_number   IN    VARCHAR2,
   filter_set       IN    VARCHAR2);

DBMS_WORKLOAD_REPLAY.USE_FILTER_SET(
   filter_set       IN    VARCHAR2);
```

Parameters

Table 221-37 USE_FILTER_SET Procedure Parameters

Parameter	Description
`capture_number`	Pointing to a capture of the current replay schedule

Table 221-37 (Cont.) USE_FILTER_SET Procedure Parameters

Parameter	Description
`filter_set`	Name of the filter set

Usage Notes

The filter set must have been created by calling the CREATE_FILTER_SET Procedure.