

## DBMS\_AUDIT\_MGMT

The `DBMS_AUDIT_MGMT` package provides subprograms to manage audit trail records. These subprograms enable audit administrators to manage the audit trail. In a mixed-mode environment, these audit trails comprise the database, operating system (OS), and XML audit trails. In a unified auditing environment, this comprises the unified audit trail.

This chapter contains the following topics:

- [DBMS\\_AUDIT\\_MGMT Overview](#)
- [DBMS\\_AUDIT\\_MGMT Deprecated and Desupported Subprograms](#)
- [DBMS\\_AUDIT\\_MGMT Security Model](#)
- [DBMS\\_AUDIT\\_MGMT Constants](#)
- [DBMS\\_AUDIT\\_MGMT Views](#)
- [Subprogram Groups](#)
- [Summary of DBMS\\_AUDIT\\_MGMT Subprograms](#)

### See Also:

- *Oracle Database Security Guide* regarding verifying security access with auditing
- *Oracle Database Upgrade Guide* regarding migrating to unified auditing

## DBMS\_AUDIT\_MGMT Overview

Database auditing helps meet your database security and compliance requirements. In a mixed mode environment, audit records are written to database tables, operating system (OS) files, or XML files depending on the `AUDIT_TRAIL` initialization parameter setting. If you have upgraded to unified auditing, then the audit records are written to the unified audit trail.

### Note:

Traditional auditing is desupported in Oracle Database 23ai. Oracle recommends that you use unified auditing instead. Any current traditional audit settings that you have will still be honored, but you cannot create new traditional audit settings. You can delete existing traditional audit settings. See *Oracle Database Security Guide* for more information.

In a mixed mode environment, when `AUDIT_TRAIL` is set to `DB`, database records are written to the `AUD$` table. In a unified auditing environment, audit records are written to a read-only table in the `AUDSYS` schema. The contents of this table are available from the `UNIFIED_AUDIT_TRAIL` data dictionary view. When `AUDIT_TRAIL` is set to `OS`, audit records are written to operating

system files. When `AUDIT_TRAIL` is set to XML, audit records are written to operating system files in XML format.

With Unified Auditing facility, all audit records are written to the unified audit trail in a uniform format and are made available through the `UNIFIED_AUDIT_TRAIL` views.

It is important to manage your audit records properly in order to ensure efficient performance and disk space management. The `DBMS_AUDIT_MGMT` subprograms enable you to efficiently manage your audit trail records.

If you have not yet migrated to unified auditing, the `DBMS_AUDIT_MGMT` package provides a subprogram that allows you to move the database audit trail tables out of the `SYSTEM` tablespace. This improves overall database performance by reducing the load on the `SYSTEM` tablespace. It also enables you to dedicate an optimized tablespace for audit records.

For a mixed mode environment, the `DBMS_AUDIT_MGMT` subprograms also enable you to manage your operating system and XML audit files. You can define properties like the maximum size and age of an audit file. This enables you to keep the file sizes of OS and XML audit files in check.

The `DBMS_AUDIT_MGMT` subprograms enable you to perform cleanup operations on all audit trail types. Audit trails can be cleaned based on the last archive timestamp value. The last archive timestamp represents the timestamp of the most recent audit record that was securely archived.

The `DBMS_AUDIT_MGMT` package provides a subprogram that enables audit administrators to set the last archive timestamp for archived audit records. This subprogram can also be used by external archival systems to set the last archive timestamp.

The `DBMS_AUDIT_MGMT` subprograms also enable you to configure jobs that periodically delete audit trail records. The frequency with which these jobs should run can be controlled by the audit administrator.



#### See Also:

*Oracle Database Security Guide* for a detailed description of unified auditing

## DBMS\_AUDIT\_MGMT Deprecated and Desupported Subprograms

Oracle recommends that you do not use deprecated subprograms in new applications. Support for deprecated features is for backward compatibility only.

The following have been deprecated from the `DBMS_AUDIT_MGMT` package starting in Oracle Database 12c release 12.2:

- `DBMS_AUDIT_MGMT.FLUSH_UNIFIED_AUDIT_TRAIL` procedure (desupported starting in Oracle Database 23.4)
- `AUDIT_TRAIL_WRITE` mode of the `AUDIT_TRAIL_PROPERTY` parameter of the `DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY` procedure

These are no longer necessary because audit records now bypass the common logging infrastructure queues and are directly written to a new internal relational table.

## DBMS\_AUDIT\_MGMT Security Model

All DBMS\_AUDIT\_MGMT subprograms require the user to have EXECUTE privilege over the DBMS\_AUDIT\_MGMT package. The SYSDBA and AUDIT\_ADMIN roles have EXECUTE privileges on the package by default.

Oracle strongly recommends that only audit administrators should have the EXECUTE privilege on the DBMS\_AUDIT\_MGMT package and be granted the AUDIT\_ADMIN role.

Executions of the DBMS\_AUDIT\_MGMT subprograms are always audited.

## DBMS\_AUDIT\_MGMT Constants

The DBMS\_AUDIT\_MGMT package defines several constants that can be used for specifying parameter values.

These constants shown in the following tables:

- [Table 32-1](#)
- [Table 32-2](#)
- [Table 32-3](#)

Audit trails can be classified based on whether audit records are written to database tables, operating system files, or XML files. The following table lists the audit trail type constants.

**Table 32-1 DBMS\_AUDIT\_MGMT Constants - Audit Trail Types**

Constant	Type	Description
AUDIT_TRAIL_ALL	PLS_INTEGER	All audit trail types. This includes the standard database audit trail (SYS.AUD\$, SYS.FGA_LOG\$ and unified audit trail tables), operating system (OS) audit trail, and XML audit trail.
AUDIT_TRAIL_AUD_STD	PLS_INTEGER	Standard database audit records in the SYS.AUD\$ table
AUDIT_TRAIL_DB_STD	PLS_INTEGER	Both standard audit (SYS.AUD\$) and FGA audit(SYS.FGA_LOG\$) records
AUDIT_TRAIL_FGA_STD	PLS_INTEGER	Standard database fine-grained auditing (FGA) records in the SYS.FGA_LOG\$ table
AUDIT_TRAIL_FILES	PLS_INTEGER	Both operating system (OS) and XML audit trails
AUDIT_TRAIL_OS	PLS_INTEGER	Operating system audit trail. This refers to the audit records stored in operating system files.
AUDIT_TRAIL_UNIFIED	PLS_INTEGER	Unified audit trail. In unified auditing, all audit records are written to the unified audit trail and are made available through the unified audit trail views, such as UNIFIED_AUDIT_TRAIL.
AUDIT_TRAIL_UNIFIED_FILES	PLS_INTEGER	Operating system spillover files in each database (primary or standby); used in archive or purge operations
AUDIT_TRAIL_UNIFIED_TABLE	PLS_INTEGER	Records from the AUDSYS.AUD\$UNIFIED table; used in archive or purge operations

**Table 32-1 (Cont.) DBMS\_AUDIT\_MGMT Constants - Audit Trail Types**

Constant	Type	Description
AUDIT_TRAIL_XML	PLS_INTEGER	XML audit trail. This refers to the audit records stored in XML files.

Audit trail properties determine the audit configuration settings. The following table lists the constants related to audit trail properties.

**Table 32-2 DBMS\_AUDIT\_MGMT Constants - Audit Trail Properties**

Constant	Type	Description
AUDIT_TRAIL_WRITE_MODE	PLS_INTEGER	<p><b>Note:</b> AUDIT_TRAIL_WRITE_MODE has been deprecated starting in Oracle Database 12c release 12.2. It is retained only for backwards compatibility, and has no effect on Oracle Database versions 12.2 and higher.</p> <p>A value of AUDIT_TRAIL_IMMEDIATE_WRITE indicates that the audit record must be immediately persisted and not to be queued. By contrast, AUDIT_TRAIL_QUEUED_WRITE indicates that the audit record can be queued and persisting can be done according the database's flushing strategy.</p> <p>See Also <i>Oracle Database Security Guide</i></p>
CLEAN_UP_INTERVAL	PLS_INTEGER	Interval, in hours, after which the cleanup procedure is called to clear audit records in the specified audit trail
DB_DELETE_BATCH_SIZE	PLS_INTEGER	Specifies the batch size to be used for deleting audit records in database audit tables. The audit records are deleted in batches of size equal to DB_DELETE_BATCH_SIZE.
FILE_DELETE_BATCH_SIZE	PLS_INTEGER	Specifies the batch size to be used for deleting audit files in the audit directory. The audit files are deleted in batches of size equal to FILE_DELETE_BATCH_SIZE.
OS_FILE_MAX_AGE	PLS_INTEGER	Specifies the maximum number of days for which an operating system (OS) or XML audit file can be kept open before a new audit file gets created
OS_FILE_MAX_SIZE	PLS_INTEGER	Specifies the maximum size, in kilobytes (KB), to which an operating system (OS) or XML audit file can grow before a new file is opened

The audit trail purge job cleans the audit trail. The following table lists the constants related to purge job status values.

**Table 32-3 DBMS\_AUDIT\_MGMT Constants - Purge Job Status**

Constant	Type	Description
PURGE_JOB_DISABLE	PLS_INTEGER	Disables a purge job
PURGE_JOB_ENABLE	PLS_INTEGER	Enables a purge job

## DBMS\_AUDIT\_MGMT Views

DBMS\_AUDIT\_MGMT views are used to display DBMS\_AUDIT\_MGMT configuration and cleanup events.

These views listed in the following table.

**Table 32-4 Views used by DBMS\_AUDIT\_MGMT**

View	Description
DBA_AUDIT_MGMT_CLEAN_EVENTS	Displays the cleanup event history
DBA_AUDIT_MGMT_CLEANUP_JOBS	Displays the currently configured audit trail purge jobs
DBA_AUDIT_MGMT_CONFIG_PARAMS	Displays the currently configured audit trail properties
DBA_AUDIT_MGMT_LAST_ARCH_TS	Displays the last archive timestamps set for the audit trails



### See Also:

*Oracle Database Reference* for more information on these views

## DBMS\_AUDIT\_MGMT Subprogram Groups

The DBMS\_AUDIT\_MGMT package subprograms can be grouped into two categories: Audit Trail Management Subprograms and Audit Trail Cleanup Subprograms.

- [Audit Trail Management Subprograms](#)
- [Audit Trail Cleanup Subprograms](#)

## DBMS\_AUDIT\_MGMT Audit Trail Management Subprograms

Audit trail management subprograms enable you to manage audit trail properties.

**Table 32-5 Audit Trail Management Subprograms**

Subprogram	Description
<a href="#">ALTER_PARTITION_INTERVAL Procedure</a>	Changes the unified audit internal relational table's partition interval
<a href="#">CLEAR_AUDIT_TRAIL_PROPERTY Procedure</a>	Clears the value for the audit trail property that you specify
<a href="#">GET_AUDIT_TRAIL_PROPERTY_VALUE Function</a>	Returns the property value set by the <a href="#">SET_AUDIT_TRAIL_PROPERTY Procedure</a>
<a href="#">GET_LAST_ARCHIVE_TIMESTAMP Function</a>	Returns the timestamp set by the <a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a> in that database instance
<a href="#">LOAD_UNIFIED_AUDIT_FILES Procedure</a>	Loads the data from the spillover OS audit files in a unified audit trail into the designated unified audit trail tablespace

**Table 32-5 (Cont.) Audit Trail Management Subprograms**

Subprogram	Description
<a href="#">SET_AUDIT_TRAIL_LOCATION Procedure</a>	Moves the audit trail tables from their current tablespace to a user-specified tablespace
<a href="#">SET_AUDIT_TRAIL_PROPERTY Procedure</a>	Sets an audit trail property for the audit trail type that you specify
<a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a>	Sets a timestamp indicating when the audit records or files were last archived
<a href="#">TRANSFER_UNIFIED_AUDIT_RECORDS Procedure</a>	Transfers audit records from the common logging infrastructure (CLI) swap table to the AUDSYS.AUD\$UNIFIED relational table

The [Summary of DBMS\\_AUDIT\\_MGMT Subprograms](#) contains a complete listing of all subprograms in the package.

## DBMS\_AUDIT\_MGMT Audit Trail Cleanup Subprograms

Audit trail cleanup subprograms help you perform cleanup related operations on the audit trail records.

**Table 32-6 Audit Trail Cleanup Subprograms**

Subprogram	Description
<a href="#">CLEAN_AUDIT_TRAIL Procedure</a>	Deletes audit trail records or files that have been archived
<a href="#">CLEAR_LAST_ARCHIVE_TIMESTAMP Procedure</a>	Clears the timestamp set by the <a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a>
<a href="#">CREATE_PURGE_JOB Procedure</a>	Creates a purge job for periodically deleting the audit trail records or files
<a href="#">DEINIT_CLEANUP Procedure</a>	Undoes the setup and initialization performed by the <a href="#">INIT_CLEANUP Procedure</a>
<a href="#">DROP_OLD_UNIFIED_AUDIT_TABLES Procedure</a>	Drops old unified audit tables following the cloning of a pluggable database (PDB)
<a href="#">DROP_PURGE_JOB Procedure</a>	Drops the purge job created using the <a href="#">CREATE_PURGE_JOB Procedure</a>
<a href="#">INIT_CLEANUP Procedure</a>	Sets up the audit management infrastructure and sets a default cleanup interval for audit trail records or files
<a href="#">IS_CLEANUP_INITIALIZED Function</a>	Checks to see if the <a href="#">INIT_CLEANUP Procedure</a> has been run for an audit trail type
<a href="#">SET_PURGE_JOB_INTERVAL Procedure</a>	Sets the interval at which the <a href="#">CLEAN_AUDIT_TRAIL Procedure</a> is called for the purge job that you specify
<a href="#">SET_PURGE_JOB_STATUS Procedure</a>	Enables or disables the purge job that you specify

The [Summary of DBMS\\_AUDIT\\_MGMT Subprograms](#) contains a complete listing of all subprograms in the package.

## Summary of DBMS\_AUDIT\_MGMT Subprograms

This table lists and describes the subprograms of the DBMS\_AUDIT\_MGMT package

**Table 32-7 DBMS\_AUDIT\_MGMT Package Subprograms**

Subprogram	Description
<a href="#">ALTER_PARTITION_INTERVAL Procedure</a>	Changes the unified audit internal relational table's partition interval
<a href="#">CLEAN_AUDIT_TRAIL Procedure</a>	Deletes audit trail records that have been archived
<a href="#">CLEAR_AUDIT_TRAIL_PROPERTY Procedure</a>	Clears the value for the audit trail property that you specify
<a href="#">CLEAR_LAST_ARCHIVE_TIMESTAMP Procedure</a>	Clears the timestamp set by the <a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a>
<a href="#">CREATE_PURGE_JOB Procedure</a>	Creates a purge job for periodically deleting the audit trail records
<a href="#">DEINIT_CLEANUP Procedure</a>	Undoes the setup and initialization performed by the <a href="#">INIT_CLEANUP Procedure</a>
<a href="#">DROP_OLD_UNIFIED_AUDIT_TABLES Procedure</a>	Drops old unified audit tables following the cloning of a pluggable database (PDB)
<a href="#">DROP_PURGE_JOB Procedure</a>	Drops the purge job created using the <a href="#">CREATE_PURGE_JOB Procedure</a>
<a href="#">GET_AUDIT_COMMIT_DELAY Function</a>	Returns the audit commit delay time as the number of seconds. This is the maximum time that it takes to COMMIT an audit record to the database audit trail.
<a href="#">GET_AUDIT_TRAIL_PROPERTY_VALUE Function</a>	Returns the property value set by the <a href="#">SET_AUDIT_TRAIL_PROPERTY Procedure</a>
<a href="#">GET_LAST_ARCHIVE_TIMESTAMP Function</a>	Returns the timestamp set by the <a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a> in that database instance
<a href="#">INIT_CLEANUP Procedure</a>	Sets up the audit management infrastructure and sets a default cleanup interval for audit trail records
<a href="#">IS_CLEANUP_INITIALIZED Function</a>	Checks to see if the <a href="#">INIT_CLEANUP Procedure</a> has been run for an audit trail type
<a href="#">LOAD_UNIFIED_AUDIT_FILES Procedure</a>	Loads the data from the spillover OS audit files in a unified audit trail into the designated unified audit trail tablespace
<a href="#">SET_AUDIT_TRAIL_LOCATION Procedure</a>	Moves the audit trail tables from their current tablespace to a user-specified tablespace
<a href="#">SET_AUDIT_TRAIL_PROPERTY Procedure</a>	Sets the audit trail properties for the audit trail type that you specify
<a href="#">SET_LAST_ARCHIVE_TIMESTAMP Procedure</a>	Sets a timestamp indicating when the audit records were last archived
<a href="#">SET_PURGE_JOB_INTERVAL Procedure</a>	Sets the interval at which the <a href="#">CLEAN_AUDIT_TRAIL Procedure</a> is called for the purge job that you specify
<a href="#">SET_PURGE_JOB_STATUS Procedure</a>	Enables or disables the purge job that you specify
<a href="#">TRANSFER_UNIFIED_AUDIT_RECORDS Procedure</a>	Transfers audit records from the common logging infrastructure (CLI) swap table to the AUDSYS.AUD\$UNIFIED relational table

## ALTER\_PARTITION\_INTERVAL Procedure

This procedure changes the unified audit internal relational table's partition interval.

### Syntax

```
DBMS_AUDIT_MGMT.ALTER_PARTITION_INTERVAL(
    interval_number          IN BINARY_INTEGER,
    interval_frequency       IN VARCHAR2);
```

### Parameters

**Table 32-8 ALTER\_PARTITION\_INTERVAL Procedure Parameters**

Parameter	Description
interval_number	Sets how often the database creates partitions for the unified audit internal relational table. For example, to specify that the partition is created every two days, you must enter 2.
interval_frequency	Sets the frequency for the value that was set in the interval_number setting. For example, for a partition to be created every two days, with interval_number set to 2, you must set interval_frequency to DAY.  Supported values are YEAR, MONTH, and DAY. The default is DAY.

### Usage Notes

- The interval frequency that you choose depends on the rate of audit records that are generated in your database. The default setting is for one month. If you have a high audit record rate and are using the default, then too many audit records may be generated in the same partition. In this case, you should change the interval frequency to a more frequent interval, such as one month or one day. If the audit record rate generation is not so high, then you may want to keep it at the default of one month.

### Example

The following example sets the partition interval to occur every two months.

```
BEGIN
    DBMS_AUDIT_MGMT.ALTER_PARTITION_INTERVAL(
        interval_number      => 2,
        interval_frequency    => 'MONTH');
END;
```

## CLEAN\_AUDIT\_TRAIL Procedure

This procedure deletes audit trail records.

The `CLEAN_AUDIT_TRAIL` procedure is usually called after the [SET\\_LAST\\_ARCHIVE\\_TIMESTAMP Procedure](#) has been used to set the last archived timestamp for the audit records.

### Syntax

```
DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(
    audit_trail_type          IN PLS_INTEGER,
    use_last_arch_timestamp   IN BOOLEAN DEFAULT TRUE,
```



```

container          IN PLS_INTEGER DEFAULT CONTAINER_CURRENT,
database_id        IN NUMBER DEFAULT NULL,
container_guid     IN VARCHAR2 DEFAULT NULL);

```

## Parameters

**Table 32-9 CLEAN\_AUDIT\_TRAIL Procedure Parameters**

Parameter	Description
audit_trail_type	The audit trail type for which the cleanup operation needs to be performed. Audit trail types are listed in <a href="#">Table 32-1</a> .
use_last_arch_timestamp	Specifies whether the last archived timestamp should be used for deciding on the records that should be deleted. A value of <code>TRUE</code> indicates that only audit records created before the last archive timestamp should be deleted. A value of <code>FALSE</code> indicates that all audit records should be deleted. The default value is <code>TRUE</code> . Oracle recommends using this value, as this helps guard against inadvertent deletion of records.
container	Values: <code>CONTAINER_CURRENT</code> for the connected pluggable database (PDB) or <code>CONTAINER_ALL</code> for all pluggable databases (PDBs). When <code>CONTAINER</code> is set to <code>CONTAINER_ALL</code> , this purges the audit trail in all the PDBs, otherwise it only purges from the connected PDB.
database_id	Database ID (DBID) of the audit records to cleanup
container_guid	Container GUID of the audit records to cleanup <b>Note:</b> This parameter has been deprecated but is currently retained for backward compatibility.

## Usage Notes

The following usage notes apply:

- When cleaning up operating system (OS) or XML audit files, only files in the current audit directory, specified by the `AUDIT_FILE_DEST` parameter, are cleaned up.
- For Windows platforms, no cleanup is performed when the `audit_trail_type` parameter is set to `DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS`. This is because operating system (OS) audit records on Windows are written to the Windows Event Viewer.
- For Unix platforms, no cleanup is performed for cases where the operating system (OS) audit records are written to the syslog. When the `audit_trail_type` parameter is set to `DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS`, it removes only the `*.aud` files under the directory specified by the `AUDIT_FILE_DEST` initialization parameter.

### See Also:

"AUDIT\_SYSLOG\_LEVEL" in the *Oracle Database Reference*

- When the `audit_trail_type` parameter is set to `DBMS_AUDIT_MGMT.AUDIT_TRAIL_XML`, this procedure only removes XML audit files (`*.xml`) from the current audit directory.

Oracle database maintains a book-keeping file (`adx_${ORACLE_SID}.txt`) for the XML audit files. This file is not removed by the cleanup procedure.

- If the cleanup of the unified audit trail is performed when the `use_last_arch_timestamp` parameter is set to `TRUE`:
  - If you set the `database_id` value for the cleanup operation, then this value is used with the last archive timestamp while `CLEAN_AUDIT_TRAIL` runs. While cleaning up the unified audit records that are present in the database tables, if the `HIGH_VALUE` of an audit table partition is less than last archive timestamp, then those partitions will be dropped directly and for remaining audit table partitions, records will be deleted based on the `database_id` value along with last archive timestamp. However, for the unified audit records that are present during the cleanup of spillover operating system audit files, the `database_id` value is ignored. Cleanup for operating system audit files is based on the last archive timestamp only. If you want to have the `database_id` value used for the cleanup operation of unified audit trail records that are present in the spillover operating system audit files, then load the contents of these files into database tables by using the `DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILES` procedure before you run `DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL`.
  - If you do not set the `database_id` value when you invoke the `CLEAN_AUDIT_TRAIL` procedure, then Oracle Database purges unified audit records based on the last archive timestamp value irrespective of `database_id` values and irrespective of the location (that is, database tables or spillover operating system audit files) where the unified audit records reside. During cleanup, last archive timestamp value of current database ID of the container will be used.
  - The last archive timestamp (LAT) should be set in the individual PDBs. Each individual PDB may have a different LAT. If this is not done, there is a risk that uncollected audit records will be deleted.
- If the cleanup of the unified audit trail is performed when the `use_last_arch_timestamp` parameter is set to `FALSE`:
  - If you set the `database_id` value for the cleanup operation, then this value is used while `CLEAN_AUDIT_TRAIL` cleans up the unified audit trail records that are present in database tables. However, the `database_id` value is not used for the cleanup of unified audit trail records that are present in spillover operating system files. If want the `database_id` value to be used in the `CLEAN_AUDIT_TRAIL` operation of unified audit records that are present in spillover operating system audit files, then load the contents of these files to the database tables by using the `DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILES` procedure before you run `DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL`.
  - If you do not set the `database_id` value when you invoke the `CLEAN_AUDIT_TRAIL` procedure, then Oracle Database purges all unified audit records irrespective of `database_id` values and irrespective of the location (that is, database tables or spillover operating system audit files) where the unified audit records reside.
- `CLEAN_AUDIT_TRAIL` procedure expects that the last archive timestamp set via `SET_LAST_ARCHIVE_TIMESTAMP` has been set using `SYS_EXTRACT_UTC` if it want to refer to `SYSTIMESTAMP`. Reference to data in Unified Audit trail is done in UTC timeformat.
- If the PDB database is read only then clean up of that database audit record does not take place and returns without performing a clean up job on `AUDSYS.AUD$UNIFIED` table.
- If timestamp of database is ahead of current system timestamp then it will return an error.
- In a multitenant setup, if one of the PDB has timestamp ahead of the current system timestamp (in UTC format) then records for that PDB are not cleaned where as the other database do clean up their audit trail records leaving the last activity of "clean up" in the audit trail.

- The `SET_LAST_ARCHIVE_TIMESTAMP` procedure and `CLEAN_AUDIT_TRAIL` procedure should be not be executed in the same transaction block. Otherwise, the results are usually unpredictable.
- Make sure that the transaction for `SET_LAST_ARCHIVE_TIMESTAMP` is completed before you call the `CLEAN_AUDIT_TRAIL` procedure. To have predictable results in terms of cleaning records until the time provided in `SET_LAST_ARCHIVE_TIMESTAMP( UTC format )`, make sure that `SET_LAST_ARCHIVE_TIMESTAMP` and `CLEAN_AUDIT_TRAIL` procedures are not called in the same transaction. It is recommended to `commit` before you call the `CLEAN_AUDIT_TRAIL` procedure.
- Run the `CLEAN_AUDIT_TRAIL` procedure when `USE_LAST_ARCH_TIMESTAMP = TRUE` uses the last archived timestamp that was set for the current `DB_UNIQUE_NAME` column value.

### Examples

The following example calls the `CLEAN_AUDIT_TRAIL` procedure to clean up the operating system (OS) audit trail records that were updated before the last archive timestamp.

```
BEGIN
DBMS_AUDIT_MGMT.CLEAN_AUDIT_TRAIL(
    audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
    use_last_arch_timestamp => TRUE);
END;
```

## CLEAR\_AUDIT\_TRAIL\_PROPERTY Procedure

This procedure clears the value for the specified audit trail property.

Audit trail properties are set using the [SET\\_AUDIT\\_TRAIL\\_PROPERTY Procedure](#).

The `CLEAR_AUDIT_TRAIL_PROPERTY` procedure can optionally reset the property value to its default value through the `use_default_values` parameter.

### Syntax

```
DBMS_AUDIT_MGMT.CLEAR_AUDIT_TRAIL_PROPERTY(
    audit_trail_type      IN PLS_INTEGER,
    audit_trail_property  IN PLS_INTEGER,
    use_default_values    IN BOOLEAN DEFAULT FALSE) ;
```

### Parameters

**Table 32-10** `CLEAR_AUDIT_TRAIL_PROPERTY` Procedure Parameters

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the property needs to be cleared. Audit trail types are listed in <a href="#">Table 32-1</a>
<code>audit_trail_property</code>	The audit trail property whose value needs to be cleared. You cannot clear the value for the <code>CLEANUP_INTERVAL</code> property. Audit trail properties are listed in <a href="#">Table 32-2</a>
<code>use_default_values</code>	Specifies whether the default value of the <code>audit_trail_property</code> should be used in place of the cleared value. A value of <code>TRUE</code> causes the default value of the parameter to be used. A value of <code>FALSE</code> causes the <code>audit_trail_property</code> to have no value. The default value for this parameter is <code>FALSE</code> .

## Usage Notes

The following usage notes apply:

- You can use this procedure to clear the value for an audit trail property that you do not wish to use. For example, if you do not want a restriction on the operating system audit file size, then you can use this procedure to reset the `OS_FILE_MAX_SIZE` property.  
  
You can also use this procedure to reset an audit trail property to its default value. You need to set `use_default_values` to `TRUE` when invoking the procedure.
- The `DB_DELETE_BATCH_SIZE` property needs to be individually cleared for the `AUDIT_TRAIL_AUD_STD` and `AUDIT_TRAIL_FGA_STD` audit trail types. You cannot clear this property collectively using the `AUDIT_TRAIL_DB_STD` and `AUDIT_TRAIL_ALL` audit trail types.
- If you clear the value of the `DB_DELETE_BATCH_SIZE` property with `use_default_value` set to `FALSE`, the default value of `DB_DELETE_BATCH_SIZE` is still assumed. This is because audit records are always deleted in batches.
- The `FILE_DELETE_BATCH_SIZE` property needs to be individually cleared for the `AUDIT_TRAIL_OS` and `AUDIT_TRAIL_XML` audit trail types. You cannot clear this property collectively using the `AUDIT_TRAIL_FILES` and `AUDIT_TRAIL_ALL` audit trail types.
- If you clear the value of the `FILE_DELETE_BATCH_SIZE` property with `use_default_value` set to `FALSE`, the default value of `FILE_DELETE_BATCH_SIZE` is still assumed. This is because audit files are always deleted in batches.
- You cannot clear the value for the `CLEANUP_INTERVAL` property.
- You cannot clear the value for the `AUDIT_TRAIL_WRITE_MODE` property.

## Examples

The following example calls the `CLEAR_AUDIT_TRAIL_PROPERTY` procedure to clear the value for the audit trail property, `OS_FILE_MAX_SIZE`. The procedure uses a value of `FALSE` for the `USE_DEFAULT_VALUES` parameter. This means that there will be no maximum size threshold for operating system (OS) audit files.

```
BEGIN
DBMS_AUDIT_MGMT.CLEAR_AUDIT_TRAIL_PROPERTY (
    AUDIT_TRAIL_TYPE      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
    AUDIT_TRAIL_PROPERTY  => DBMS_AUDIT_MGMT.OS_FILE_MAX_SIZE,
    USE_DEFAULT_VALUES    => FALSE );
END;
```

## CLEAR\_LAST\_ARCHIVE\_TIMESTAMP Procedure

This procedure clears the timestamp set by the `SET_LAST_ARCHIVE_TIMESTAMP` Procedure.

### Syntax

```
DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMP (
    audit_trail_type      IN PLS_INTEGER,
    rac_instance_number  IN PLS_INTEGER DEFAULT NULL,
    container             IN PLS_INTEGER DEFAULT CONTAINER_CURRENT,
    database_id          IN NUMBER DEFAULT NULL,
    container_guid        IN VARCHAR2 DEFAULT NULL);
```

## Parameters

**Table 32-11 CLEAR\_LAST\_ARCHIVE\_TIMESTAMP Procedure Parameters**

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the timestamp needs to be cleared. Audit trail types are listed in <a href="#">Table 32-1</a> .
<code>rac_instance_number</code>	The instance number for the Oracle Real Application Clusters (Oracle RAC) instance. The default value is <code>NULL</code> . The <code>rac_instance_number</code> is not relevant for single instance databases. You can find the instance number by issuing the <code>SHOW PARAMETER INSTANCE_NUMBER</code> command in SQL*Plus.
<code>container</code>	Values: <code>CONTAINER_CURRENT</code> for the connected pluggable database (PDB) or <code>CONTAINER_ALL</code> for all pluggable databases (PDBs). When <code>CONTAINER</code> is set to <code>CONTAINER_ALL</code> , this clears the last archive timestamp from all the PDBs, otherwise it clears from only the connected PDB.
<code>database_id</code>	Database ID (DBID) of the audit records to cleanup
<code>container_guid</code>	Container GUID of the audit records to cleanup <b>Note:</b> This parameter has been deprecated but is currently retained for backward compatibility.

## Usage Notes

The following usage notes apply:

- The timestamp for only one `audit_trail_type` can be cleared at a time.
- The following are invalid `audit_trail_type` values for this procedure and cannot be used:
  - `AUDIT_TRAIL_ALL`
  - `AUDIT_TRAIL_DB_STD`
  - `AUDIT_TRAIL_FILES`

## Examples

The following example calls the `CLEAR_LAST_ARCHIVE_TIMESTAMP` procedure to clear the timestamp value for the operating system (OS) audit trail type.

```
BEGIN
DBMS_AUDIT_MGMT.CLEAR_LAST_ARCHIVE_TIMESTAMP(
  audit_trail_type => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
  rac_instance_number => 1);
END;
```

## Related Topics

- [SET\\_LAST\\_ARCHIVE\\_TIMESTAMP Procedure](#)  
This procedure sets a timestamp indicating when the audit records were last archived. The audit administrator provides the timestamp to be attached to the audit records.

## CREATE\_PURGE\_JOB Procedure

This procedure creates a purge job for periodically deleting the audit trail records.

This procedure carries out the cleanup operation at intervals specified by the user. It calls the [CLEAN\\_AUDIT\\_TRAIL Procedure](#) to perform the cleanup operation.

The [SET\\_PURGE\\_JOB\\_INTERVAL Procedure](#) is used to modify the frequency of the purge job.

The [SET\\_PURGE\\_JOB\\_STATUS Procedure](#) is used to enable or disable the purge job.

The [DROP\\_PURGE\\_JOB Procedure](#) is used to drop a purge job created with the CREATE\_PURGE\_JOB procedure.

### Syntax

```
DBMS_AUDIT_MGMT.CREATE_PURGE_JOB (
    audit_trail_type          IN PLS_INTEGER,
    audit_trail_purge_interval IN PLS_INTEGER,
    audit_trail_purge_name    IN VARCHAR2,
    use_last_arch_timestamp   IN BOOLEAN DEFAULT TRUE,
    container                 IN PLS_INTEGER DEFAULT CONTAINER_CURRENT) ;
```

### Parameters

**Table 32-12 CREATE\_PURGE\_JOB Procedure Parameters**

Parameter	Description
audit_trail_type	The audit trail type for which the purge job needs to be created. Audit trail types are listed in <a href="#">Table 32-1</a> .
audit_trail_purge_interval	The interval, in hours, at which the clean up procedure is called. A lower value means that the cleanup is performed more often.
audit_trail_purge_name	A name to identify the purge job.
use_last_arch_timestamp	Specifies whether the last archived timestamp should be used for deciding on the records that should be deleted. A value of <b>TRUE</b> indicates that only audit records created before the last archive timestamp should be deleted. A value of <b>FALSE</b> indicates that all audit records should be deleted. The default value is <b>TRUE</b> .
container	Values: <b>CONTAINER_CURRENT</b> for the connected pluggable database (PDB) or <b>CONTAINER_ALL</b> for all pluggable databases (PDBs). When <b>CONTAINER</b> is set to <b>CONTAINER_ALL</b> , it creates one job in the Root PDB and the invocation of this job will invoke cleanup in all the PDBs.

### Usage Notes

Use this procedure to schedule the [CLEAN\\_AUDIT\\_TRAIL Procedure](#) for your audit trail records.

If the `use_last_arch_timestamp` parameter is set to `TRUE`, the last archive timestamp (LAT) should be set in the individual PDBs. Each individual PDB may have a different LAT. If this is not done, there is a risk that uncollected audit records will be deleted.

### Examples

The following example calls the `CREATE_PURGE_JOB` procedure to create a cleanup job called `CLEANUP`, for all audit trail types. It sets the `audit_trail_purge_interval` parameter to 100. This means that the cleanup job is invoked every 100 hours. It also sets the `use_last_arch_timestamp` parameter value to `TRUE`. This means that all audit records older than the last archive timestamp are deleted.

```
BEGIN
DBMS_AUDIT_MGMT.CREATE_PURGE_JOB (
  audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALL,
  audit_trail_purge_interval => 100 /* hours */,
  audit_trail_purge_name  => 'CLEANUP',
  use_last_arch_timestamp => TRUE);
END;
```

## DEINIT\_CLEANUP Procedure

This procedure undoes the setup and initialization performed by the `INIT_CLEANUP` procedure. This procedure is deprecated starting in Oracle Database 23ai. Traditional auditing packages and functions are deprecated in Oracle Database 23ai. The `DEINIT_CLEANUP` procedure clears the value of the `default_cleanup_interval` parameter. However, when used for audit tables, it does not move the audit trail tables back to their original tablespace.

### Syntax

```
DBMS_AUDIT_MGMT.DEINIT_CLEANUP (
  audit_trail_type IN PLS_INTEGER,
  container        IN PLS_INTEGER DEFAULT CONTAINER_CURRENT);
```

### Parameters

**Table 32-13 DEINIT\_CLEANUP Procedure Parameters**

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the procedure needs to be called. Audit trail types are listed in <a href="#">Table 32-1</a>
<code>container</code>	Values: <code>CONTAINER_CURRENT</code> for the connected pluggable database (PDB) or <code>CONTAINER_ALL</code> for all pluggable databases (PDBs). When <code>CONTAINER</code> is set to <code>CONTAINER_ALL</code> , this de-initializes the audit trail from cleanup in all the pluggable databases, otherwise it de-initializes the audit trail from cleanup in the connected PDB only.

### Usage Notes

With the desupport of traditional auditing, the PL/SQL packages and functions associated with traditional auditing are deprecated. This deprecation includes the packages and functions `INIT_CLEANUP`, `DEINIT_CLEANUP`, and `IS_CLEANUP_INITIALIZED`. While these packages or functions continue to operate in Oracle Database 23ai, you can neither add to or modify traditional auditing configurations.

You cannot run this procedure for `AUDIT_TRAIL_UNIFIED`. Doing so it will raise `ORA-46250` : Invalid value for argument 'AUDIT\_TRAIL\_TYPE'

### Examples

The following example clears the `default_cleanup_interval` parameter setting for the standard database audit trail:

```
BEGIN
DBMS_AUDIT_MGMT.DEINIT_CLEANUP(
  AUDIT_TRAIL_TYPE => DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD);
END;
```

### Related Topics

- [INIT\\_CLEANUP Procedure](#)

## DROP\_OLD\_UNIFIED\_AUDIT\_TABLES Procedure

This procedure drops old unified audit tables following the cloning of a pluggable database (PDB).

### Syntax

```
DBMS_AUDIT_MGMT.DROP_OLD_UNIFIED_AUDIT_TABLES (
  container_guid    IN VARCHAR2) ;
```

### Parameters

**Table 32-14 DROP\_OLD\_UNIFIED\_AUDIT\_TABLES Procedure Parameters**

Parameter	Description
<code>container_guid</code>	Container GUID of the old unified audit tables

### Usage Notes

When a pluggable database gets cloned, the unified audit tables get newly created in the new pluggable database. To drop the old unified audit tables, use the `DROP_OLD_UNIFIED_AUDIT_TABLES` by specifying the old GUID of the PDB from which the clone was created. You can query the historical GUIDs from the `DBA_PDB_HISTORY` view for the given PDB.

Only use the `DBMS_AUDIT_MGMT.DROP_OLD_UNIFIED_AUDIT_TABLES` procedure if the database was upgraded from Oracle Database release 12.1 or earlier. If the database was upgraded from a later release (including release 12.2), then an `ORA-55906: Secure file log [id: 0 name: ORA$AUDIT_NEXTGEN_LOG] does not exist` error will appear. This is because starting with release 12.2, the common logging infrastructure tables that this procedure searches are no longer being created. Instead, the `AUD$UNIFIED` relational table stores the unified audit records.

### Examples

```
BEGIN
  DBMS_AUDIT_MGMT.DROP_OLD_UNIFIED_AUDIT_TABLES (
    container_guid => 'E4721865A9321CB5E043EFA9E80A2D77');
END;
```



## DROP\_PURGE\_JOB Procedure

This procedure drops the purge job created using the CREATE\_PURGE\_JOB Procedure. The name of the purge job is passed as an argument.

### Syntax

```
DBMS_AUDIT_MGMT.DROP_PURGE_JOB(  
    audit_trail_purge_name    IN VARCHAR2) ;
```

### Parameters

**Table 32-15 DROP\_PURGE\_JOB Procedure Parameters**

Parameter	Description
audit_trail_purge_name	The name of the purge job which is being deleted. This is the purge job name that you specified with the <a href="#">CREATE_PURGE_JOB Procedure</a> .

### Examples

The following example calls the DROP\_PURGE\_JOB procedure to drop the purge job called CLEANUP.

```
BEGIN  
DBMS_AUDIT_MGMT.DROP_PURGE_JOB(  
    AUDIT_TRAIL_PURGE_NAME => 'CLEANUP');  
END;
```

### Related Topics

- [CREATE\\_PURGE\\_JOB Procedure](#)  
This procedure creates a purge job for periodically deleting the audit trail records.

## GET\_AUDIT\_COMMIT\_DELAY Function

This function returns the audit commit delay time as the number of seconds. audit commit delay time is the maximum time that it takes to COMMIT an audit record to the database audit trail. If it takes more time to COMMIT an audit record than defined by the audit commit delay time, then a copy of the audit record is written to the operating system (OS) audit trail.

The audit commit delay time value is useful when determining the last archive timestamp for database audit records.

### Syntax

```
DBMS_AUDIT_MGMT.GET_AUDIT_COMMIT_DELAY  
    RETURN NUMBER;
```

## GET\_AUDIT\_TRAIL\_PROPERTY\_VALUE Function

This procedure returns the property value set by the SET\_AUDIT\_TRAIL\_PROPERTY Procedure.

### Syntax

```
DBMS_AUDIT_MGMT.GET_AUDIT_TRAIL_PROPERTY_VALUE (  
    audit_trail_type          IN PLS_INTEGER,  
    audit_trail_property      IN PLS_INTEGER)  
RETURN NUMBER;
```

### Parameters

**Table 32-16** GET\_AUDIT\_TRAIL\_PROPERTY\_VALUE Function Parameters

Parameter	Description
audit_trail_type	The audit trail type for the timestamp to be retrieved. Audit trail types are listed in <a href="#">Table 32-1</a> .
audit_trail_property	The audit trail property that is being queried. Audit trail properties are listed in <a href="#">Table 32-2</a> .

### Return Values

If the property value is cached in SGA memory, this function will return the value set by the [SET\\_AUDIT\\_TRAIL\\_PROPERTY Procedure](#). Else it will return NULL.

The GET\_AUDIT\_TRAIL\_PROPERTY\_VALUE function may return an ORA-46250 error if the audit trail property value has been set to DBMS\_AUDIT\_MGMT.CLEAN\_UP\_INTERVAL. To find the cleanup interval of the purge job, query SYS.DAM\_CLEANUP\_JOBS\$.

### Examples

The following example prints the property value of OS\_FILE\_MAX\_AGE set by the [SET\\_AUDIT\\_TRAIL\\_PROPERTY Procedure](#).

```
SET AUDIT_TRAIL_PROPERTY.  
SET SERVEROUTPUT ON  
DECLARE  
    OS_MAX_AGE_VAL NUMBER;  
BEGIN  
    OS_MAX_AGE_VAL := DBMS_AUDIT_MGMT.GET_AUDIT_TRAIL_PROPERTY_VALUE (  
        audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,  
        audit_trail_property  => DBMS_AUDIT_MGMT.OS_FILE_MAX_AGE);  
    IF OS_MAX_AGE_VAL is not NULL THEN  
        DBMS_OUTPUT.PUT_LINE('The Maximum Age configured for OS Audit files is: ' ||  
                               OS_MAX_AGE_VAL);  
    END IF;  
END;
```

### Related Topics

- [SET\\_AUDIT\\_TRAIL\\_PROPERTY Procedure](#)  
This procedure sets an audit trail property for the audit trail type that is specified.

## GET\_LAST\_ARCHIVE\_TIMESTAMP Function

This procedure returns the timestamp set by the `SET_LAST_ARCHIVE_TIMESTAMP` procedure in that database instance.

### Syntax

```
DBMS_AUDIT_MGMT.GET_LAST_ARCHIVE_TIMESTAMP(  
    audit_trail_type    IN PLS_INTEGER)  
RETURN TIMESTAMP;
```

### Parameters

**Table 32-17** GET\_LAST\_ARCHIVE\_TIMESTAMP Function Parameters

Parameter	Description
audit_trail_type	The audit trail type for the timestamp to be retrieved. Supported Audit trail types for this procedure are <code>AUDIT_TRAIL_OS</code> , <code>AUDIT_TRAIL_XML</code> and <code>AUDIT_TRAIL_UNIFIED</code> . All Audit trail types are listed in <a href="#">Table 32-1</a> .

### Return Values

The `GET_LAST_ARCHIVE_TIMESTAMP` function returns last archive timestamp.

### Usage Notes

- Alternatively, you can query the `DBA_AUDIT_MGMT_LAST_ARCH_TS` data dictionary view to check the last archived timestamp for a database that is in read-write mode.
- The `DBA_AUDIT_MGMT_LAST_ARCH_TS` data dictionary view can contain multiple last archive timestamps with different `DB_UNIQUE_NAME` column values. `GET_LAST_ARCHIVE_TIMESTAMP` will return the last archive timestamp where the `DB_UNIQUE_NAME` column value is equal to the current `DB_UNIQUE_NAME` value.

### Examples

The following example prints the timestamp set by the `SET_LAST_ARCHIVE_TIMESTAMP` procedure on a `READ ONLY` database.

```
SET SERVEROUTPUT ON  
DECLARE  
    LAT_TS TIMESTAMP;  
BEGIN  
    LAT_TS := DBMS_AUDIT_MGMT.GET_LAST_ARCHIVE_TIMESTAMP(  
        audit_trail_type => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS);  
    IF LAT_TS is not NULL THEN  
        DBMS_OUTPUT.PUT_LINE('The Last Archive Timestamp is: ' || to_char(LAT_TS));  
    END IF;  
END;
```

### Related Topics

- [SET\\_LAST\\_ARCHIVE\\_TIMESTAMP Procedure](#)  
This procedure sets a timestamp indicating when the audit records were last archived. The audit administrator provides the timestamp to be attached to the audit records.

## INIT\_CLEANUP Procedure

This procedure sets up the audit management infrastructure and a default cleanup interval for the audit trail records. This procedure is deprecated starting in Oracle Database 23ai. Traditional auditing packages and functions are deprecated in Oracle Database 23ai. If the audit trail tables are in the `SYSTEM` tablespace, then the procedure moves them to the `SYSAUX` tablespace. If you are using unified auditing, you do not need to run this procedure because the unified audit trail tables are in the `SYSAUX` tablespace by default. If you are not using unified auditing, refer to *Oracle Database Upgrade Guide* for documentation which references an environment without unified auditing.

Moving the audit trail tables out of the `SYSTEM` tablespace enhances overall database performance. The `INIT_CLEANUP` procedure moves the audit trail tables to the `SYSAUX` tablespace. If the [SET\\_AUDIT\\_TRAIL\\_LOCATION Procedure](#) has already moved the audit tables elsewhere, then no tables are moved.

The [SET\\_AUDIT\\_TRAIL\\_LOCATION Procedure](#) enables you to specify an alternate target tablespace for the database audit tables.

The `INIT_CLEANUP` procedure is currently not relevant for the `AUDIT_TRAIL_OS`, `AUDIT_TRAIL_XML`, and `AUDIT_TRAIL_FILES` audit trail types. No preliminary set up is required for these audit trail types.



### See Also:

[Table 32-1](#) for a list of all audit trail types

This procedure also sets a default cleanup interval for the audit trail records.

### Syntax

```
DBMS_AUDIT_MGMT.INIT_CLEANUP(
    audit_trail_type          IN PLS_INTEGER,
    default_cleanup_interval  IN PLS_INTEGER
    container                 IN PLS_INTEGER DEFAULT CONTAINER_CURRENT);
```

### Parameters

**Table 32-18** INIT\_CLEANUP Procedure Parameters

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the clean up operation needs to be initialized. Audit trail types are listed in <a href="#">Table 32-1</a> except <code>AUDIT_TRAIL_UNIFIED</code>
<code>default_cleanup_interval</code>	The default time interval, in hours, after which the cleanup procedure should be called. The minimum value is 1 and the maximum is 999.

**Table 32-18 (Cont.) INIT\_CLEANUP Procedure Parameters**

Parameter	Description
container	Values: CONTAINER_CURRENT for the connected pluggable database (PDB) or CONTAINER_ALL for all Open and Available pluggable databases (PDBs). When CONTAINER is set to CONTAINER_ALL, this initializes the audit trails for clean up in all the Open and Available pluggable databases, otherwise this initializes the audit trail in the connected PDB only. When you add a new PDB you need to initialize the audit trails for clean up in the new PDB using the CONTAINER_CURRENT option.

### Usage Notes

With the desupport of traditional auditing, the PL/SQL packages and functions associated with traditional auditing are deprecated. This deprecation includes the packages and functions `INIT_CLEANUP`, `DEINIT_CLEANUP`, and `IS_CLEANUP_INITIALIZED`. While these packages or functions continue to operate in Oracle Database 23ai, you can neither add to or modify traditional auditing configurations.

The following usage notes apply:

- This procedure may involve data movement across tablespaces. This can be a resource intensive operation especially if your database audit trail tables are already populated. Oracle recommends that you invoke the procedure during non-peak hours.
- You should ensure that the `SYSAUX` tablespace, into which the audit trail tables are being moved, has sufficient space to accommodate the audit trail tables. You should also optimize the `SYSAUX` tablespace for frequent write operations.
- You can change the `default_cleanup_interval` later using the [SET\\_AUDIT\\_TRAIL\\_PROPERTY Procedure](#).
- If you do not wish to move the audit trail tables to the `SYSAUX` tablespace, then you should use the `DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION` procedure to move the audit trail tables to another tablespace before calling the `INIT_CLEANUP` procedure.
- Invoking this procedure with `AUDIT_TRAIL_UNIFIED` results in `ORA-46250`. It requires no initializations for cleanup since it is cleanup-ready by default.



### See Also:

["SET\\_AUDIT\\_TRAIL\\_LOCATION Procedure"](#)

### Examples

The following example calls the `INIT_CLEANUP` procedure to set a `default_cleanup_interval` of 12 hours for all audit trail types:

```
BEGIN
DBMS_AUDIT_MGMT.INIT_CLEANUP(
    audit_trail_type => DBMS_AUDIT_MGMT.AUDIT_TRAIL_ALL,
    default_cleanup_interval => 12 /* hours */);
END;
```

**See Also:**

[Table 32-1](#) for a list of all audit trail types

## IS\_CLEANUP\_INITIALIZED Function

This function checks to see if the `INIT_CLEANUP` procedure has been run for an audit trail type. This procedure is deprecated starting in Oracle Database 23ai. Traditional auditing packages and functions are deprecated in Oracle Database 23ai.

With the desupport of traditional auditing, the PL/SQL packages and functions associated with traditional auditing are deprecated. This deprecation includes the packages and functions `INIT_CLEANUP`, `DEINIT_CLEANUP`, and `IS_CLEANUP_INITIALIZED`. While these packages or functions continue to operate in Oracle Database 23ai, you can neither add to or modify traditional auditing configurations.

The `IS_CLEANUP_INITIALIZED` function returns `TRUE` if the procedure has already been run for the audit trail type. It returns `FALSE` if the procedure has not been run for the audit trail type.

This function is currently not relevant for the `AUDIT_TRAIL_OS`, `AUDIT_TRAIL_XML`, and `AUDIT_TRAIL_FILES` audit trail types. The function always returns `TRUE` for these audit trail types. No preliminary set up is required for these audit trail types.

**See Also:**

[Table 32-1](#) for a list of all audit trail types

### Syntax

```
DBMS_AUDIT_MGMT.IS_CLEANUP_INITIALIZED(  
    audit_trail_type IN PLS_INTEGER  
    container        IN PLS_INTEGER DEFAULT CONTAINER_CURRENT)  
RETURN BOOLEAN;
```

### Parameters

**Table 32-19 IS\_CLEANUP\_INITIALIZED Function Parameters**

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the function needs to be called. Note that this does not apply to <code>AUDIT_TRAIL_UNIFIED</code> . Audit trail types are listed in <a href="#">Table 32-1</a>
<code>container</code>	Values: <code>CONTAINER_CURRENT</code> for the connected pluggable database (PDB) or <code>CONTAINER_ALL</code> for all pluggable databases (PDBs). <ul style="list-style-type: none"> <li>When <code>CONTAINER</code> is set to <code>CONTAINER_ALL</code>, this function returns the initialization status of all the pluggable databases. The function returns <code>FALSE</code> even if one of the PDBs is not initialized.</li> <li>When <code>CONTAINER</code> is set to <code>CONTAINER_CURRENT</code>, this returns the initialization status of the connected PDB.</li> </ul>

## Examples

The following example checks to see if the standard database audit trail type has been initialized for cleanup operation. If the audit trail type has not been initialized, then it calls the [INIT\\_CLEANUP Procedure](#) to initialize the audit trail type.

```
BEGIN
  IF
    NOT DBMS_AUDIT_MGMT.IS_CLEANUP_INITIALIZED(DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD)
  THEN
    DBMS_AUDIT_MGMT.INIT_CLEANUP(
      audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
      default_cleanup_interval => 12 /* hours */);
  END IF;
END;
```

## Related Topics

- [INIT\\_CLEANUP Procedure](#)

# LOAD\_UNIFIED\_AUDIT\_FILES Procedure

This procedure loads the data from the spillover OS audit files in a unified audit trail into the designated unified audit trail tablespace.



### See Also:

*Oracle Database Security Guide* for information about moving the OS audit trail records into the unified audit trail

## Syntax

```
DBMS_AUDIT_MGMT.LOAD_UNIFIED_AUDIT_FILES (
  container      IN BINARY_INTEGER,
  load_batch_size IN PLS_INTEGER);
```

## Parameters

**Table 32-20** LOAD\_UNIFIED\_AUDIT\_FILES Procedure Parameters

Parameter	Description
container	<p>Values: CONTAINER_CURRENT for the connected pluggable database (PDB) or CONTAINER_ALL for all pluggable databases (PDBs).</p> <ul style="list-style-type: none"><li>• CONTAINER_CURRENT - loads the unified audit files from \$ORACLE_BASE/audit/\$ORACLE_SID OS directory to the tables in only current PDB</li><li>• CONTAINER_ALL - loads the unified audit files from \$ORACLE_BASE/audit/\$ORACLE_SID OS directory to the tables in the respective PDBs, but for all the active PDBs</li></ul>

**Table 32-20 (Cont.) LOAD\_UNIFIED\_AUDIT\_FILES Procedure Parameters**

Parameter	Description
<code>load_batch_size</code>	Specifies the number of spillover OS audit files to be loaded into the designated unified audit trail tablespace. The <code>load_batch_size</code> parameter can have a minimum value of 1 and maximum value of 32767. The default value is equivalent to <code>DEFAULT_FILE_LOAD_BATCH_SIZE</code> .

#### Usage Notes

- Ensure that you set the audit data designated tablespace to be online before you run the `LOAD_UNIFIED_AUDIT_FILES` procedure.

## SET\_AUDIT\_TRAIL\_LOCATION Procedure

This procedure moves the audit trail tables from their current tablespace to a user-specified tablespace.

The `SET_AUDIT_TRAIL_LOCATION` procedure is not relevant for the `AUDIT_TRAIL_OS`, `AUDIT_TRAIL_XML`, and `AUDIT_TRAIL_FILES` audit trail types. The `AUDIT_FILE_DEST` initialization parameter is the only way you can specify the destination directory for these audit trail types.



#### See Also:

- [Table 32-1](#) for a list of all audit trail types
- `AUDIT_FILE_DEST` in the *Oracle Database Reference*

#### Syntax

```
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION (
    audit_trail_type          IN PLS_INTEGER,
    audit_trail_location_value IN VARCHAR2) ;
```

#### Parameters

**Table 32-21 SET\_AUDIT\_TRAIL\_LOCATION Procedure Parameters**

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the audit trail location needs to be set. Audit trail types are listed in <a href="#">Table 32-1</a>
<code>audit_trail_location_value</code>	Target location or tablespace for the audit trail records

#### Usage Notes

The following usage notes apply:

- Before changing the audit trail location, check the block size of the new location that you plan to use. The block size must be the same size as that of the current audit trail location.



Otherwise, an `ORA-14520: Tablespace string block size [string] does not match existing object block size [string]` error appears. To check the block size of a tablespace, query the `USER_TABLESPACES` or `DBA_TABLESPACES` data dictionary view. See *Oracle Database VLDB and Partitioning Guide* for more information about partition restrictions for multiple block sizes.

- By default, audit trail records are written to the `SYSAUX` tablespace. You can change the tablespace location to a user tablespace using this procedure. However the designated tablespace must be an automatic storage space management (ASSM) tablespace.
- This procedure is valid for the following `audit_trail_type` values:
  - `AUDIT_TRAIL_AUD_STD`
  - `AUDIT_TRAIL_FGA_STD`
  - `AUDIT_TRAIL_DB_STD`
  - `AUDIT_TRAIL_UNIFIED`
- For the `audit_trail_type` values `AUDIT_TRAIL_AUD_STD`, `AUDIT_TRAIL_FGA_STD`, and `AUDIT_TRAIL_DB_STD`, you should ensure that the target tablespace, into which the audit trail tables are being moved, has sufficient space to accommodate the audit trail tables. You should also optimize the target tablespace for frequent write operations.
- This procedure involves data movement across tablespaces. For the `audit_trail_type` values `AUDIT_TRAIL_AUD_STD`, `AUDIT_TRAIL_FGA_STD`, and `AUDIT_TRAIL_DB_STD`, this can be a resource intensive operation especially if your database audit trail tables are already populated. Oracle recommends that you invoke the procedure during non-peak hours.
- You optionally can specify an encrypted tablespace for the audit trail location.
- When `AUDIT_TRAIL_TYPE` is `AUDIT_TRAIL_UNIFIED`, this procedure sets the tablespace for newer audit records in the unified audit trail but does not move the older audit records. Thus, it is not resource intensive for the unified audit trail.
- The `UNIFIED_AUDIT_TRAIL` data dictionary view is built on top of an internal relational table. This table is an interval partitioned table (irrespective of database editions) with a default interval of 1 day. This setting means that when you execute the `DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION` procedure, only newly created partitions of the internal table are created in the new tablespace that is set as part of this procedure. Existing partitions of this table remain in the earlier tablespace (`SYSAUX` is the default tablespace for this internal table). If you want to change this table's partition interval, then use the `DBMS_AUDIT_MGMT.ALTER_PARTITION_INTERVAL` procedure.
- As a best practice to improve performance and the availability of audit records, enable the `AUTOEXTEND` option on the audit tablespace.
- If the tablespace that is specified for the `audit_trail_location_value` parameter has a block size that is different from the current tablespace, then this procedure for `AUDIT_TRAIL_UNIFIED` `audit_trail_type` fails with an `ORA-14520: Tablespace string block size [string] does not match existing object block size [string]` error. This is due to the restrictions of multiple block sizes for partition tables.

## Examples

The following example moves the database audit trail tables, `AUD$` and `FGA_LOG$`, from the current tablespace to a user-created tablespace called `RECORDS`:

```
BEGIN
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_LOCATION(
    audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_DB_STD,
```

```
        audit_trail_location_value => 'RECORDS');
END;
```

## SET\_AUDIT\_TRAIL\_PROPERTY Procedure

This procedure sets an audit trail property for the audit trail type that is specified.

The procedure sets the properties `OS_FILE_MAX_SIZE`, `OS_FILE_MAX_AGE`, and `FILE_DELETE_BATCH_SIZE` for operating system (OS), XML, and unified audit trail types. The `OS_FILE_MAX_SIZE` and `OS_FILE_MAX_AGE` properties determine the maximum size and age of an audit trail file before a new audit trail file is created. The `OS_FILE_MAX_SIZE` and `OS_FILE_MAX_AGE` properties determine the maximum size and age of an audit trail file before a new audit trail file gets created. The `FILE_DELETE_BATCH_SIZE` property specifies the number of audit trail files that are deleted in one batch.

The procedure sets the properties `DB_DELETE_BATCH_SIZE` and `CLEANUP_INTERVAL` for the database audit trail type. `DB_DELETE_BATCH_SIZE` specifies the batch size in which records get deleted from audit trail tables. This ensures that if a cleanup operation gets interrupted midway, the process does not need to start afresh the next time it is invoked. This is because all batches before the last processed batch are already committed.

The `CLEANUP_INTERVAL` specifies the frequency, in hours, with which the cleanup procedure is called.

### Syntax

```
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY(
    audit_trail_type          IN PLS_INTEGER,
    audit_trail_property      IN PLS_INTEGER,
    audit_trail_property_value IN PLS_INTEGER) ;
```

### Parameters

**Table 32-22 SET\_AUDIT\_TRAIL\_PROPERTY Procedure Parameters**

Parameter	Description
<code>audit_trail_type</code>	The audit trail type for which the property needs to be set. Audit trail types are listed in <a href="#">Table 32-1</a>
<code>audit_trail_property</code>	The audit trail property that is being set. Audit trail properties are listed in <a href="#">Table 32-2</a>

**Table 32-22 (Cont.) SET\_AUDIT\_TRAIL\_PROPERTY Procedure Parameters**

Parameter	Description
audit_trail_property_value	<p>The value of the property specified using <code>audit_trail_property</code>. The following are valid values for audit trail properties:</p> <ul style="list-style-type: none"> <li>• <code>OS_FILE_MAX_SIZE</code> can have a minimum value of 1 and maximum value of 2000000. The default value is 10000. <code>OS_FILE_MAX_SIZE</code> is measured in kilobytes (KB).</li> <li>• <code>OS_FILE_MAX_AGE</code> can have a minimum value of 1 and a maximum value of 497. The default value is 5. <code>OS_FILE_MAX_AGE</code> is measured in days.</li> <li>• <code>DB_DELETE_BATCH_SIZE</code> can have a minimum value of 100 and a maximum value of 1000000. The default value is 10000. <code>DB_DELETE_BATCH_SIZE</code> is measured as the number of audit records that are deleted in one batch.</li> <li>• <code>FILE_DELETE_BATCH_SIZE</code> can have a minimum value of 100 and a maximum value of 1000000. The default value is 1000. <code>FILE_DELETE_BATCH_SIZE</code> is measured as the number of audit files that are deleted in one batch.</li> <li>• <code>CLEANUP_INTERVAL</code> can have a minimum value of 1 and a maximum value of 999. The default value is set using the <a href="#">INIT_CLEANUP Procedure</a>. <code>CLEANUP_INTERVAL</code> is measured in hours.</li> </ul>

### Usage Notes

The following usage notes apply:

- The audit trail properties for which you do not explicitly set values use their default values.
- If you have set both the `OS_FILE_MAX_SIZE` and `OS_FILE_MAX_AGE` properties for an operating system (OS), XML or unified audit trail type, then a new audit trail file is created depending on which of these two limits is reached first

For example, let us take a scenario where `OS_FILE_MAX_SIZE` is 10000 and `OS_FILE_MAX_AGE` is 5. If the operating system audit file is already more than 5 days old and has a size of 9000 KB, then a new audit file is opened. This is because one of the limits has been reached.

- You can set the `OS_FILE_MAX_SIZE` and `OS_FILE_MAX_AGE` properties for the unified audit trail type by using `AUDIT_TRAIL_UNIFIED` or `AUDIT_TRAIL_UNIFIED_FILES`. Note that `AUDIT_TRAIL_UNIFIED` is deprecated
- The `DB_DELETE_BATCH_SIZE` property needs to be individually set for the `AUDIT_TRAIL_AUD_STD` and `AUDIT_TRAIL_FGA_STD` audit trail types. You cannot set this property collectively using the `AUDIT_TRAIL_DB_STD` and `AUDIT_TRAIL_ALL` audit trail types.
- The `DB_DELETE_BATCH_SIZE` property enables you to control the number of audit records that are deleted in one batch. Setting a large value for this parameter requires increased allocation for the undo log space.
- The `FILE_DELETE_BATCH_SIZE` property needs to be individually set for the `AUDIT_TRAIL_OS` and `AUDIT_TRAIL_XML` audit trail types. You cannot set this property collectively using the `AUDIT_TRAIL_FILES` and `AUDIT_TRAIL_ALL` audit trail types.

- The `FILE_DELETE_BATCH_SIZE` property enables you to control the number of audit files that are deleted in one batch. Setting a very large value may engage the `GEN0` background process for a long time.
- In Oracle Database Standard Edition, you can only associate the tablespace for unified auditing once. You should perform this association before you generate any audit records for the unified audit trail. The default tablespace is `SYSAUX`. After you have associated the tablespace, you cannot modify it on the Standard Edition because the partitioning feature is not supported in the Standard Edition.

### Examples

The following example calls the `SET_AUDIT_TRAIL_PROPERTY` procedure to set the `OS_FILE_MAX_SIZE` property for the operating system (OS) audit trail. It sets this property value to 102400. This means that a new audit file gets created every time the current audit file size reaches 100 MB.

```
BEGIN
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY(
    audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
    audit_trail_property  => DBMS_AUDIT_MGMT.OS_FILE_MAX_SIZE,
    audit_trail_property_value => 102400 /* 100MB*/ );
END;
```

The following example calls the `SET_AUDIT_TRAIL_PROPERTY` procedure to set the `OS_FILE_MAX_AGE` property for the operating system (OS) audit trail. It sets this property value to 5. This means that a new audit file gets created every sixth day.

```
BEGIN
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY(
    audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
    audit_trail_property  => DBMS_AUDIT_MGMT.OS_FILE_MAX_AGE,
    audit_trail_property_value => 5 /* days */);
END;
```

The following example calls the `SET_AUDIT_TRAIL_PROPERTY` procedure to set the `DB_DELETE_BATCH_SIZE` property for the `AUDIT_TRAIL_AUD_STD` audit trail. It sets this property value to 100000. This means that during a cleanup operation, audit records are deleted from the `SYS.AUD$` table in batches of size 100000.

```
BEGIN
DBMS_AUDIT_MGMT.SET_AUDIT_TRAIL_PROPERTY(
    audit_trail_type      => DBMS_AUDIT_MGMT.AUDIT_TRAIL_AUD_STD,
    audit_trail_property  => DBMS_AUDIT_MGMT.DB_DELETE_BATCH_SIZE,
    audit_trail_property_value => 100000 /* delete batch size */);
END;
```

## SET\_LAST\_ARCHIVE\_TIMESTAMP Procedure

This procedure sets a timestamp indicating when the audit records were last archived. The audit administrator provides the timestamp to be attached to the audit records.

The [CLEAN\\_AUDIT\\_TRAIL Procedure](#) uses this timestamp to decide on the audit records to be deleted.


### Syntax

```
DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP(
    audit_trail_type      IN PLS_INTEGER,
    last_archive_time     IN TIMESTAMP,
```

```
rac_instance_number    IN PLS_INTEGER DEFAULT NULL,
container              IN PLS_INTEGER DEFAULT CONTAINER_CURRENT,
database_id            IN NUMBER DEFAULT NULL,
container_guid         IN VARCHAR2 DEFAULT NULL);
```

## Parameters

**Table 32-23 SET\_LAST\_ARCHIVE\_TIMESTAMP Procedure Parameters**

Parameter	Description
audit_trail_type	The audit trail type for which the timestamp needs to be set. Audit trail types are listed in <a href="#">Table 32-1</a> .
last_archive_time	The <code>TIMESTAMP</code> value based on which the audit records or files should be deleted. This indicates the last time when the audit records or files were archived.
	<div>  <b>Note:</b>  ORA-46250 error is thrown if any future date or <code>TIMESTAMP</code> value is given. </div>
rac_instance_number	<p>The instance number for the Oracle Real Application Clusters (Oracle RAC) instance. The default value is <code>NULL</code>.</p> <p>The <code>rac_instance_number</code> parameter is not accepted when <code>AUDIT_TRAIL_TYPE</code> has values:</p> <ul style="list-style-type: none"> <li><code>audit_trail_aud_std</code></li> <li><code>audit_trail_db_std</code></li> <li><code>audit_trail_fga_std</code></li> <li><code>audit_trail_unified</code></li> </ul> <p>Set <code>rac_instance_number</code> to 1 for a single-instance database.</p>
container	Values: <code>CONTAINER_CURRENT</code> for the connected pluggable database (PDB) or <code>CONTAINER_ALL</code> for all pluggable databases (PDBs). When <code>CONTAINER</code> is set to <code>CONTAINER_ALL</code> , this sets the value for last archive timestamp in all the pluggable databases, otherwise it sets the value in the connected PDB only.
database_id	Database ID (DBID) of the audit records to cleanup
container_guid	Container GUID of the audit records to cleanup <b>Note:</b> This parameter has been deprecated but is currently retained for backward compatibility.

## Usage Notes

The following usage notes apply:

- The `last_archive_time` must be specified in Coordinated Universal Time (UTC) when the audit trail types are `AUDIT_TRAIL_AUD_STD`, `AUDIT_TRAIL_FGA_STD`, or `AUDIT_TRAIL_UNIFIED`. This is because the database audit trails store the timestamps in UTC. UTC is also known as Greenwich Mean Time (GMT).
- The `last_archive_time` must be specified as the local time zone time when the audit trail types are `AUDIT_TRAIL_OS` or `AUDIT_TRAIL_XML`. The time zone must be the time zone of the machine where the OS or XML audit files were created. This is because the operating

system audit files are cleaned based on the audit file's Last Modification Timestamp property. The Last Modification Timestamp property value is stored in the local time zone of the machine.

- When you use an Oracle Real Application Clusters (Oracle RAC) database, Oracle recommends that you use the Network Time Protocol (NTP) to synchronize individual Oracle RAC nodes.
- The `SET_LAST_ARCHIVE_TIMESTAMP` procedure and `CLEAN_AUDIT_TRAIL` procedure should be not be executed in the same transaction block. Otherwise, the results are usually unpredictable.
- If this timestamp set to a future date, an error is returned.
- Make sure that the transaction for `SET_LAST_ARCHIVE_TIMESTAMP` is completed before you call the `CLEAN_AUDIT_TRAIL` procedure. To have predictable results in terms of cleaning records until the time provided in `SET_LAST_ARCHIVE_TIMESTAMP( UTC format )`, make sure that `SET_LAST_ARCHIVE_TIMESTAMP` and `CLEAN_AUDIT_TRAIL` procedures are not called in the same transaction. It is recommended to `commit` before you call the `CLEAN_AUDIT_TRAIL` procedure.
- When you invoke `SET_LAST_ARCHIVE_TIMESTAMP` using database links, be cognizant that there could be time difference between the source and the target databases.
- When you set the last archive timestamp on a read-write database, the `DB_UNIQUE_NAME` column of the `DBA_AUDIT_MGMT_LAST_ARCH_TS` data dictionary view is also populated. You cannot configure the `DB_UNIQUE_NAME` value, so you cannot set it by using the `SET_LAST_ARCHIVE_TIMESTAMP` procedure.
- For every configuration of the last archive timestamp, Oracle Database cleans entries from the `DBA_AUDIT_MGMT_LAST_ARCH_TS` data dictionary view where the `DB_UNIQUE_NAME` column value is not equal to the current `DB_UNIQUE_NAME` value. This avoids having to reuse the old last archive timestamp when multiple switch-over operations take place and keeps in check the growing rows of the `DBA_AUDIT_MGMT_LAST_ARCH_TS` view.

### Examples

The following example calls the `SET_LAST_ARCHIVE_TIMESTAMP` procedure to set the last archive timestamp for the operating system (OS) audit trail type on Oracle RAC instance 1. It uses the `TO_TIMESTAMP` function to convert a character string into a timestamp value.

A subsequent call to the [CLEAN\\_AUDIT\\_TRAIL Procedure](#), with `use_last_arch_timestamp` set to `TRUE`, will delete all those OS audit files from the current `AUDIT_FILE_DEST` directory that were modified before 10-Sep-2012 14:10:10.0.

```
BEGIN
DBMS_AUDIT_MGMT.SET_LAST_ARCHIVE_TIMESTAMP(
  audit_trail_type    => DBMS_AUDIT_MGMT.AUDIT_TRAIL_OS,
  last_archive_time    => TO_TIMESTAMP('12-SEP-0714:10:10.0', 'DD-MON-RRHH24:MI:SS.FF'),
  rac_instance_number => 1);
END;
```

## SET\_PURGE\_JOB\_INTERVAL Procedure

This procedure sets the interval at which the `CLEAN_AUDIT_TRAIL` Procedure is called for the purge job specified.

The purge job must have already been created using the [CREATE\\_PURGE\\_JOB Procedure](#).

## Syntax

```
DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVAL(  
    audit_trail_purge_name      IN VARCHAR2,  
    audit_trail_interval_value  IN PLS_INTEGER) ;
```

## Parameters

**Table 32-24 SET\_PURGE\_JOB\_INTERVAL Procedure Parameters**

Parameter	Description
audit_trail_purge_name	The name of the purge job for which the interval is being set. This is the purge job name that you specified with the <a href="#">CREATE_PURGE_JOB Procedure</a> .
audit_trail_interval_value	The interval, in hours, at which the clean up procedure should be called. This value modifies the audit_trail_purge_interval parameter set using the <a href="#">CREATE_PURGE_JOB Procedure</a>

## Usage Notes

Use this procedure to modify the audit\_trail\_purge\_interval parameter set using the [CREATE\\_PURGE\\_JOB Procedure](#).

## Examples

The following example calls the SET\_PURGE\_JOB\_INTERVAL procedure to change the frequency at which the purge job called CLEANUP is invoked. The new interval is set to 24 hours.

```
BEGIN  
DBMS_AUDIT_MGMT.SET_PURGE_JOB_INTERVAL(  
    AUDIT_TRAIL_PURGE_NAME      => 'CLEANUP',  
    AUDIT_TRAIL_INTERVAL_VALUE  => 24 );  
END;
```

## Related Topics

- [CLEAN\\_AUDIT\\_TRAIL Procedure](#)  
This procedure deletes audit trail records.

# SET\_PURGE\_JOB\_STATUS Procedure

This procedure enables or disables the specified purge job.

The purge job must have already been created using the [CREATE\\_PURGE\\_JOB Procedure](#).

## Syntax

```
DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUS(  
    audit_trail_purge_name      IN VARCHAR2,  
    audit_trail_status_value    IN PLS_INTEGER) ;
```

## Parameters

**Table 32-25 SET\_PURGE\_JOB\_STATUS Procedure Parameters**

Parameter	Description
audit_trail_purge_name	The name of the purge job for which the status is being set. This is the purge job name that you specified with the <a href="#">CREATE_PURGE_JOB Procedure</a> .
audit_trail_status_value	One of the values specified in <a href="#">Table 32-3</a> . The value PURGE_JOB_ENABLE enables the specified purge job. The value PURGE_JOB_DISABLE disables the specified purge job.

## Examples

The following example calls the SET\_PURGE\_JOB\_STATUS procedure to enable the CLEANUP purge job.

```
BEGIN
DBMS_AUDIT_MGMT.SET_PURGE_JOB_STATUS (
  audit_trail_purge_name => 'CLEANUP',
  audit_trail_status_value => DBMS_AUDIT_MGMT.PURGE_JOB_ENABLE);
END;
```

# TRANSFER\_UNIFIED\_AUDIT\_RECORDS Procedure

This procedure transfers unified audit records that were in a pre-upgraded Oracle database to an internal relational table that is designed to improve read performance.

In the pre-upgraded Oracle database, these records resided in the common logging infrastructure (CLI) SGA back-end tables.

## Syntax

```
DBMS_AUDIT_MGMT.TRANSFER_UNIFIED_AUDIT_RECORDS (
  container_guid          IN VARCHAR2    DEFAULT NULL);
```

## Parameters

**Table 32-26 TRANSFER\_UNIFIED\_AUDIT\_RECORDS Procedure Parameters**

Parameter	Description
container_guid	The GUID of the container of the associated CLI back-end table. This back-end table contains the audit records from the pre-upgraded Oracle database. If you omit this setting, then the GUID of the current container is used.

## Usage Notes

- It is not mandatory to run DBMS\_AUDIT\_MGMT.TRANSFER\_UNIFIED\_AUDIT\_RECORDS after an upgrade, but for better read performance of the unified audit trail, Oracle highly recommends that you run this procedure.
- The DBMS\_AUDIT\_MGMT.TRANSFER\_UNIFIED\_AUDIT\_RECORDS is designed to be a one-time operation, to be performed after you upgrade from Oracle Database 12c release 12.1.



- There is one CLI back-end table per GUID of the container. You can find the GUIDs for containers by querying the `PDB_GUID` column of the `DBA_PDB_HISTORY` data dictionary view. Execute the `DBMS_AUDIT_MGMT.TRANSFER_UNIFIED_AUDIT_RECORDS` procedure by passing each of these GUIDs one by one to ensure that you move the unified audit records from these CLI back-end tables to the `AUDSYS.AUD$UNIFIED` table.
- In a multitenant environment, you must run the `DBMS_AUDIT_MGMT.TRANSFER_UNIFIED_AUDIT_RECORDS` procedure only in the container to which the transfer operation applies, whether it is the root or an individual PDB. You cannot run this procedure in the root, for example, to transfer audit records in a PDB.
- If you have a high rate of audit record generation and your database supports partitioning, then you may want to use the `DBMS_AUDIT_MGMT.ALTER_PARTITION_INTERVAL` procedure to alter the partition interval setting for the internal relational table. See [ALTER\\_PARTITION\\_INTERVAL Procedure](#) for more information.