244
DBMS_XSTREAM_ADM

This `DBMS_XSTREAM_ADM` package provides interfaces for streaming database changes between an Oracle database and other systems. XStream enables applications to stream out or stream in database changes.

This chapter contains the following topic:

> **See Also:**
>
> - *Oracle Database XStream Guide*
> - *Oracle Call Interface Programmer's Guide*
> - *Oracle Database XStream Java API Reference*

DBMS_XSTREAM_ADM Overview

The package provides interfaces for configuring outbound servers that stream database changes from an Oracle database to other systems. The package also provides interfaces for configuring inbound servers that stream database changes from other systems to an Oracle database.

In both cases, the database changes are encapsulated in logical change records (LCRs). Also, the other systems can be Oracle systems or a non-Oracle systems, such as non-Oracle databases or file systems.

XStream outbound servers can stream out LCRs from an Oracle database programmatically using C or Java. After receiving the LCRs, the other system can process them in any customized way. For example, the other system can save the contents of the LCRs to a file, send the LCRs to an Oracle database through an XStream inbound server, or generate SQL statements and execute them on any Oracle or non-Oracle databases.

XStream inbound servers accept LCRs from another system and either apply them to an Oracle database or process them in a customized way using apply handlers.

XStream can be used in a multitenant container database (CDB). A CDB is an Oracle database that includes zero, one, or many user-created pluggable databases (PDBs).

> **See Also:**
>
> - *Oracle Database XStream Guide*
> - *Oracle Database Concepts* for more information about CDBs and PDBs

DBMS_XSTREAM_ADM Security Model

To ensure that the user who runs the subprograms in this package has the necessary privileges, configure an XStream administrator and connect as the XStream administrator when using this package.

An administrator must be granted the `DBA` role when the administrator is performing any of the following actions:

- Running the `ADD_OUTBOUND` procedure while connected as a user that is different from the configured connect user for an outbound server
- Running the `ALTER_OUTBOUND` procedure to change the capture user for a capture process or the connect user for an outbound server
- Running the `CREATE_OUTBOUND` procedure, because this procedure creates a capture process
- Running the `ALTER_INBOUND` procedure to change the apply user for an inbound server
- Running the `ADD_INBOUND` procedure while connected as a user that is different from the configured apply user for an inbound server

When the administrator does not need to perform the preceding tasks, the `DBA` role is not required.

> **See Also:**
>
> *Oracle Database XStream Guide, Chapter 4, "XStream Out and Security"* for more information about XStream and security.

DBMS_XSTREAM_ADM Operational Notes

Some subprograms in the `DBMS_APPLY_ADM` package can manage XStream outbound servers, and some subprograms in the `DBMS_APPLY_ADM` package can manage XStream inbound servers.

> **See Also:**
>
> DBMS_APPLY_ADM for details about which subprograms can manage outbound servers and inbound servers

Summary of DBMS_XSTREAM_ADM Subprograms

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

Table 244-1 DBMS_XSTREAM_ADM Package Subprograms

Subprogram	Description
ADD_COLUMN Procedure	Either adds or removes a declarative rule-based transformation which adds a column to a row logical change record (row LCR) that satisfies the specified rule
ADD_GLOBAL_PROPAGATION_RULES Procedure	Either adds global rules to the positive rule set for a propagation, or adds global rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist
ADD_GLOBAL_RULES Procedure	Adds global rules to either the positive or negative rule set of a capture process or apply process, and creates the specified capture process or apply process if it does not exist
ADD_OUTBOUND Procedure	Creates an XStream outbound server that dequeues LCRs from the specified queue
ADD_SCHEMA_PROPAGATION_RULES Procedure	Either adds schema rules to the positive rule set for a propagation, or adds schema rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist
ADD_SCHEMA_RULES Procedure	Adds rules to a rule set of XStream clients.
ADD_SUBSET_OUTBOUND_RULES Procedure	Adds subset rules to an outbound server configuration
ADD_SUBSET_PROPAGATION_RULES Procedure	Adds subset rules to the positive rule set for a propagation, and creates the specified propagation if it does not exist
ADD_SUBSET_RULES Procedure	Adds subset rules to the positive rule set of a capture process or apply process, and creates the specified capture process or apply process if it does not exist
ADD_TABLE_PROPAGATION_RULES Procedure	Either adds table rules to the positive rule set for a propagation, or adds table rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist
ADD_TABLE_RULES Procedure	This procedure adds rules to a rule set of an XStream client.
ALTER_INBOUND Procedure	Modifies an XStream inbound server
ALTER_OUTBOUND Procedure	Modifies an XStream outbound server
CREATE_INBOUND Procedure	Creates an XStream inbound server and its queue
CREATE_OUTBOUND Procedure	Creates an XStream outbound server, queue, and capture process to enable XStream client applications to stream out Oracle database changes encapsulated in LCRs
DELETE_COLUMN Procedure	Either adds or removes a declarative rule-based transformation which deletes a column from a row LCR that satisfies the specified rule
DROP_INBOUND Procedure	Removes an inbound server configuration

Table 244-1 (Cont.) DBMS_XSTREAM_ADM Package Subprograms

Subprogram	Description
DROP_OUTBOUND Procedure	Removes an outbound server configuration
ENABLE_GG_XSTREAM_FOR_STREAMS Procedure	Enables XStream performance optimizations for Oracle Replication components
GET_MESSAGE_TRACKING Function	Returns the tracking label for the current session
GET_TAG Function	Gets the binary tag for all redo entries generated by the current session
IS_GG_XSTREAM_FOR_STREAMS Function	Returns `TRUE` if XStream performance optimizations are enabled for Oracle Replication components, or returns `FALSE` if XStream performance optimizations are disabled for Oracle Replication components
KEEP_COLUMNS Procedure	Either adds or removes a declarative rule-based transformation which keeps a list of columns in a row LCR that satisfies the specified rule
MERGE_STREAMS Procedure	Merges a stream flowing from one capture process with a stream flowing from another capture process
MERGE_STREAMS_JOB Procedure	Determines whether the original capture process and the cloned capture are within the specified merge threshold and, if they are, runs the `MERGE_STREAMS` procedure to merge the two streams
PURGE_SOURCE_CATALOG Procedure	Removes all Oracle Replication data dictionary information at the local database for the specified object
RECOVER_OPERATION Procedure	Provides options for a split and merge operation that stopped because it encountered an error. This procedure either rolls forward the operation, rolls back the operation, or purges all of the metadata about the operation.
REMOVE_QUEUE Procedure	Removes the specified `ANYDATA` queue
REMOVE_RULE Procedure	Removes the specified rule or all rules from the rule set associated with the specified capture process, apply process, or propagation.
REMOVE_SUBSET_OUTBOUND_RULES Procedure	Removes subset rules from an outbound server configuration
REMOVE_XSTREAM_CONFIGURATION Procedure	Removes the XStream configuration at the local database
RENAME_COLUMN Procedure	Either adds or removes a declarative rule-based transformation which renames a column in a row LCR that satisfies the specified rule
RENAME_SCHEMA Procedure	Either adds or removes a declarative rule-based transformation which renames a schema in a row LCR that satisfies the specified rule
RENAME_TABLE Procedure	Either adds or removes a declarative rule-based transformation which renames a table in a row LCR that satisfies the specified rule
SET_MESSAGE_TRACKING Procedure	Sets the tracking label for logical change records (LCRs) produced by the current session
SET_PARAMETER Procedure	Sets a parameter for an outbound server, an inbound server, or an outbound server's capture process

Table 244-1 (Cont.) DBMS_XSTREAM_ADM Package Subprograms

Subprogram	Description
SET_TAG Procedure	Sets the binary tag for all redo entries subsequently generated by the current session
SET_UP_QUEUE Procedure	Creates a queue table and a queue for use with the capture, propagate, and apply functionality of XStream
SPLIT_STREAMS Procedure	Splits one stream flowing from a capture process off from all of the other streams flowing from the capture process
START_OUTBOUND Procedure	Starts an XStream outbound server
STOP_OUTBOUND Procedure	Stops an XStream outbound server

Note:

All subprograms commit unless specified otherwise.

ADD_COLUMN Procedure

This procedure either adds or removes a declarative rule-based transformation which adds a column to a row logical change record (row LCR) that satisfies the specified rule.

For the transformation to be performed when the specified rule evaluates to `TRUE`, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

This procedure is overloaded. The `column_value` and `column_function` parameters are mutually exclusive.

Note:

- `ADD_COLUMN` transformations cannot add columns of the following data types: `BLOB`, `CLOB`, `NCLOB`, `BFILE`, `LONG`, `LONG RAW`, `ROWID`, user-defined types (including object types, `REF`s, varrays, nested tables), and Oracle-supplied types (including any types, XML types, spatial types, and media types).
- Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

See Also:

Oracle Database XStream Guide for more information about declarative rule-based transformations

Syntax

```
DBMS_XSTREAM_ADM.ADD_COLUMN(
   rule_name     IN  VARCHAR2,
   table_name    IN  VARCHAR2,
   column_name   IN  VARCHAR2,
   column_value  IN  ANYDATA,
   value_type    IN  VARCHAR2     DEFAULT 'NEW',
   step_number   IN  NUMBER       DEFAULT 0,
   operation     IN  VARCHAR2     DEFAULT 'ADD');

DBMS_XSTREAM_ADM.ADD_COLUMN(
   rule_name        IN  VARCHAR2,
   table_name       IN  VARCHAR2,
   column_name      IN  VARCHAR2,
   column_function  IN  VARCHAR2,
   value_type       IN  VARCHAR2     DEFAULT 'NEW',
   step_number      IN  NUMBER       DEFAULT 0,
   operation        IN  VARCHAR2     DEFAULT 'ADD');
```

Parameters

Table 244-2 ADD_COLUMN Procedure Parameters

Parameter	Description
rule_name	The name of the rule, specified as [*schema_name.*]*rule_name*. If NULL, then the procedure raises an error. For example, to specify a rule in the hr schema named employees12, enter hr.employees12. If the schema is not specified, then the current user is the default.
table_name	The name of the table to which the column is added in the row LCR, specified as [*schema_name.*]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default.
column_name	The name of the column added to each row LCR that satisfies the rule.
column_value	The value of the added column. Specify the appropriate ANYDATA function for the column datatype and the column value. For example, if the datatype of the column being added is NUMBER and the value is NULL, then specify the ANYDATA.ConvertNumber(NULL) function. This parameter cannot be specified if the column_function parameter is specified.

Table 244-2 (Cont.) ADD_COLUMN Procedure Parameters

Parameter	Description
column_function	Either the 'SYSDATE' or the 'SYSTIMESTAMP' SQL function. The 'SYSDATE' SQL function returns the current date and time. 'SYSDATE' uses the time zone of either the database host system or the database, depending on the setting of the TIME_AT_DBTIMEZONE initialization parameter. The datatype of the returned value is DATE, and the format returned depends on the value of the NLS_DATE_FORMAT initialization parameter. The 'SYSTIMESTAMP' SQL function returns the system date, including fractional seconds. 'SYSTIMESTAMP' uses the time zone of either the database host system or the database, depending on the setting of the TIME_AT_DBTIMEZONE initialization parameter. The return type is TIMESTAMP WITH TIME ZONE. The function executes when the rule evaluates to TRUE. This parameter cannot be specified if the column_value parameter is specified. **See Also:** *Oracle Database Reference* for more information about the TIME_AT_DBTIMEZONE parameter.
value_type	Specify 'NEW' to add the column to the new values in the row LCR. Specify 'OLD' to add the column to the old values in the row LCR.
step_number	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
operation	Specify 'ADD' to add the transformation to the rule. Specify 'REMOVE' to remove the transformation from the rule.

Usage Notes

When 'REMOVE' is specified for the operation parameter, all of the add column declarative rule-based transformations for the specified rule are removed that match the specified table_name, column_name, and step_number parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the ADD_COLUMN procedures when one or more of these parameters is NULL:

table_name	column_name	step_number	Result
NULL	NULL	NULL	Remove all add column transformations for the specified rule.
NULL	NULL	non-NULL	Remove all add column transformations with the specified step_number for the specified rule.
NULL	non-NULL	non-NULL	Remove all add column transformations with the specified column_name and step_number for the specified rule.

table_name	column_name	step_number	Result
non-NULL	NULL	non-NULL	Remove all add column transformations with the specified table_name and step_number for the specified rule.
NULL	non-NULL	NULL	Remove all add column transformations with the specified column_name for the specified rule.
non-NULL	non-NULL	NULL	Remove all add column transformations with the specified table_name and column_name for the specified rule.
non-NULL	NULL	NULL	Remove all add column transformations with the specified table_name for the specified rule.
non-NULL	non-NULL	non-NULL	Remove all add column transformations with the specified table_name, column_name, and step_number for the specified rule.

ADD_GLOBAL_PROPAGATION_RULES Procedure

This procedure either adds global rules to the positive rule set for a propagation, or adds global rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist.

This procedure is overloaded. One version of this procedure contains two OUT parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_GLOBAL_PROPAGATION_RULES(
   streams_name             IN   VARCHAR2  DEFAULT NULL,
   source_queue_name        IN   VARCHAR2,
   destination_queue_name   IN   VARCHAR2,
   include_dml              IN   BOOLEAN   DEFAULT TRUE,
   include_ddl              IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr       IN   BOOLEAN   DEFAULT FALSE,
   source_database          IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name            OUT  VARCHAR2,
   ddl_rule_name            OUT  VARCHAR2,
   inclusion_rule           IN   BOOLEAN   DEFAULT TRUE,
   and_condition            IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue           IN   BOOLEAN   DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_GLOBAL_PROPAGATION_RULES(
   streams_name             IN   VARCHAR2  DEFAULT NULL,
   source_queue_name        IN   VARCHAR2,
   destination_queue_name   IN   VARCHAR2,
   include_dml              IN   BOOLEAN   DEFAULT TRUE,
   include_ddl              IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr       IN   BOOLEAN   DEFAULT FALSE,
   source_database          IN   VARCHAR2  DEFAULT NULL,
   inclusion_rule           IN   BOOLEAN   DEFAULT TRUE,
   and_condition            IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue           IN   BOOLEAN   DEFAULT NULL);
```

Parameters

Table 244-3 ADD_GLOBAL_PROPAGATION_RULES Procedure Parameters

Parameter	Description
streams_name	The name of the propagation. Do not specify an owner. If the specified propagation does not exist, then the procedure creates it automatically. If NULL and a propagation exists for the same source queue and destination queue (including database link), then the procedure uses this propagation. If NULL and no propagation exists for the same source queue and destination queue (including database link), then the procedure creates a propagation automatically with a system-generated name.
source_queue_name	The name of the source queue, specified as [*schema_name.*]*queue_name*. The current database must contain the source queue, and the queue must be ANYDATA type. For example, to specify a source queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default.
destination_queue_name	The name of the destination queue, including a database link, specified as [*schema_name.*]*queue_name*[*@dblink_name*], if the destination queue is in a remote database. The queue must be ANYDATA type. For example, to specify a destination queue named streams_queue in the strmadmin schema and use a database link named dbs2.net, enter strmadmin.streams_queue@dbs2.net for this parameter. If the schema is not specified, then the current user is the default. If the database link is omitted, then the procedure uses the global name of the current database, and the source queue and destination queue must be in the same database. **Note:** Connection qualifiers are not allowed.
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.

Table 244-3 (Cont.) ADD_GLOBAL_PROPAGATION_RULES Procedure Parameters

Parameter	Description
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a logical change record (LCR) has a non-NULL tag. If the rules are added to the positive rule set for the propagation, then an LCR is always considered for propagation, regardless of whether it has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the propagation, then whether an LCR is discarded does not depend on the tag for the LCR. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if an LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for the propagation, then an LCR is considered for propagation only when the LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the propagation, then an LCR can be discarded only if it has a NULL tag. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.
source_database	The global name of the source database. The source database is where the changes originated. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically. Oracle recommends that you specify a source database for propagation rules.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the propagation. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the propagation. In either case, the system creates the rule set if it does not exist.

Table 244-3 (Cont.) ADD_GLOBAL_PROPAGATION_RULES Procedure Parameters

Parameter	Description
and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way: (*system_condition*) AND (*and_condition*) The variable in the specified condition must be :lcr. For example, to specify that the global rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition: :lcr.get_tag() = HEXTORAW(''02'') The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure the procedure only generates the appropriate rule. Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter. **See Also:** Logical Change Record TYPEs
queue_to_queue	If TRUE or NULL, then a new propagation created by this procedure is a queue to queue propagation. A queue-to-queue propagation always has its own propagation job and uses a service for automatic failover when the destination queue is a buffered queue in an Oracle Real Application Clusters (Oracle RAC) database. If FALSE, then a new propagation created by this procedure is a queue-to-dblink propagation. A queue-to-dblink propagation can share a propagation job with other propagations that use the same database link and does not support automatic failover in an Oracle RAC environment. The procedure cannot change the queue to queue property of an exiting propagation. If the specified propagation exists, then the procedure behaves in the following way for each setting: • If TRUE and the specified propagation is not a queue to queue propagation, then the procedure raises an error. • If FALSE and the specified propagation is a queue to queue propagation, then the procedure raises an error. • If NULL, then the procedure does not change the queue to queue property of the propagation.

Usage Notes

This procedure configures propagation using the current user. Only one propagation is allowed between a particular source queue and destination queue.

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the database name with a sequence number appended to it. The sequence number

is used to avoid naming conflicts. If the database name plus the sequence number is too long, then the database name is truncated. A propagation uses the rules for filtering.

Examples

The following is an example of a global rule condition created for DML changes:

```
(:dml.is_null_tag() = 'Y' and :dml.get_source_database_name() = 'DBS1.NET' )
```

ADD_GLOBAL_RULES Procedure

This procedure adds rules to an XStream clients rule set.

It adds rules to a rule set of one of the following types of XStream clients:

- When the `streams_type` parameter is set to `capture`, this procedure adds capture process rules for capturing changes to an entire database.

  This procedure creates the specified capture process if it does not exist.
- When the `streams_type` parameter is set to `apply` and the `streams_name` parameter specifies the name of an apply process, outbound server, or inbound server, this procedure adds apply rules for applying all logical change records (LCRs) it receives. The rules can specify that the LCRs must be from a particular source database.

  This procedure creates an apply process if no apply process, outbound server, or inbound server exists with the specified `streams_name`. This procedure can add rules to an outbound server or inbound server, but it cannot create an outbound server or inbound server.

This procedure is overloaded. One version of this procedure contains two `OUT` parameters, and the other does not.

> **Note:**
>
> If you add global rules to the positive rule set for a capture process, then make sure you add rules to the negative capture process rule set to exclude database objects that are not supported by Oracle Replication. Query the `DBA_XSTREAM_OUT_SUPPORT_MODE` data dictionary view to determine which database objects are not supported by Oracle Replication. If unsupported database objects are not excluded, then capture errors will result.

Syntax

```
DBMS_XSTREAM_ADM.ADD_GLOBAL_RULES(
   streams_type           IN   VARCHAR2,
   streams_name           IN   VARCHAR2  DEFAULT NULL,
   queue_name             IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml            IN   BOOLEAN   DEFAULT TRUE,
   include_ddl            IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr     IN   BOOLEAN   DEFAULT FALSE,
   source_database        IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name          OUT  VARCHAR2,
   ddl_rule_name          OUT  VARCHAR2,
   inclusion_rule         IN   BOOLEAN   DEFAULT TRUE,
   and_condition          IN   VARCHAR2  DEFAULT NULL,
   source_root_name       IN   VARCHAR2  DEFAULT NULL,
   source_container_name  IN   VARCHAR2  DEFAULT NULL);
```

```
DBMS_XSTREAM_ADM.ADD_GLOBAL_RULES(
   streams_type           IN   VARCHAR2,
   streams_name           IN   VARCHAR2  DEFAULT NULL,
   queue_name             IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml            IN   BOOLEAN   DEFAULT TRUE,
   include_ddl            IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr     IN   BOOLEAN   DEFAULT FALSE,
   source_database        IN   VARCHAR2  DEFAULT NULL,
   inclusion_rule         IN   BOOLEAN   DEFAULT TRUE,
   and_condition          IN   VARCHAR2  DEFAULT NULL,
   source_root_name       IN   VARCHAR2  DEFAULT NULL,
   source_container_name  IN   VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-4 ADD_GLOBAL_RULES Procedure Parameters

Parameter	Description
streams_type	The type of XStream client: • Specify capture for a capture process. • Specify apply for an apply process.
streams_name	The name of the capture process or apply process. Do not specify an owner. If NULL, if streams_type is capture, and if one relevant capture process for the queue exists, then the relevant XStream client is used. If no relevant XStream client exists for the queue, then an XStream client is created automatically with a system-generated name. If NULL and multiple XStream clients of the specified streams_type for the queue exist, then the procedure raises an error. If NULL, if streams_type is apply, and if one relevant apply process exists, then the procedure uses the relevant apply process. The relevant apply process is identified in one of the following ways: • If one existing apply process has the source database specified in source_database and uses the queue specified in queue_name, then the procedure uses this apply process. • If source_database is NULL and one existing apply process is using the queue specified in queue_name, then the procedure uses this apply process. If NULL and no relevant apply process exists, then the procedure creates an apply process automatically with a system-generated name. If NULL and multiple relevant apply processes exist, then the procedure raises an error. Each apply process must have a unique name.
queue_name	The name of the local queue, specified as [*schema_name.*]*queue_name*. The current database must contain the queue, and the queue must be ANYDATA type. For example, to specify a queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default. For capture process rules, this is the queue into which a capture process enqueues LCRs. For outbound server rules, this is the queue from which the outbound server dequeues LCRs. For inbound server rules, this is the queue into which an inbound server enqueues error transactions.

Table 244-4 (Cont.) ADD_GLOBAL_RULES Procedure Parameters

Parameter	Description
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a redo entry or LCR has a non-NULL tag. If the rules are added to the positive rule set for the process, then a redo entry is always considered for capture, and an LCR is always considered for apply, regardless of whether the redo entry or LCR has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the process, then whether a redo entry or LCR is discarded does not depend on the tag. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if a redo entry or LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for the process, then a redo entry is considered for capture, and an LCR is considered for apply, only when the redo entry or LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the process, then a redo entry or LCR can be discarded only if it has a NULL tag. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.
source_database	The global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. For capture process rules, specify NULL or the global name of the local database if you are creating a capture process locally at the source database. If you are adding rules to a downstream capture process rule set at a downstream database, then specify the source database of the changes that will be captured. For apply process rules, specify the source database of the changes that will be applied by the apply process. The source database is the database where the changes originated. If an apply process applies captured LCRs, then the apply process can apply LCRs from only one capture process at one source database. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.

Table 244-4 (Cont.) ADD_GLOBAL_RULES Procedure Parameters

Parameter	Description
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the XStream client. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the XStream client. In either case, the system creates the rule set if it does not exist.
and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way: (*system_condition*) AND (*and_condition*) The variable in the specified condition must be :lcr. For example, to specify that the global rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition: :lcr.get_tag() = HEXTORAW(''02'') The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure this procedure only generates the appropriate rule. Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter. **See Also:** Logical Change Record TYPEs
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.

Usage Notes

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the database name with a sequence number appended to it. The sequence number is used to avoid naming conflicts. If the database name plus the sequence number is too long,

then the database name is truncated. A capture process or apply process uses the rules for filtering.

> **See Also:**
>
> - "Operational Notes"
> - "Security Model"

Examples

The following is an example of a global rule condition created for DML changes:

```
(:dml.is_null_tag() = 'Y' and :dml.get_source_database_name() = 'DBS1.NET' )
```

ADD_OUTBOUND Procedure

This procedure creates an XStream outbound server that dequeues LCRs from the specified queue. The outbound server streams out the LCRs to an XStream client application.

This procedure creates neither a capture process nor a queue. To create an outbound server, a capture process, and a queue with one procedure call, use the CREATE_OUTBOUND Procedure.

To create the capture process individually, use one of the following packages:

- `DBMS_XSTREAM_ADM`
- `DBMS_CAPTURE_ADM`

To create a queue individually, use the `SET_UP_QUEUE` procedure in the `DBMS_XSTREAM_ADM` package.

This procedure is overloaded. One `table_names` parameter is type `VARCHAR2` and the other `table_names` parameter is type `DBMS_UTILITY.UNCL_ARRAY`. Also, one `schema_names` parameter is type `VARCHAR2` and the other `schema_names` parameter is type `DBMS_UTILITY.UNCL_ARRAY`. These parameters enable you to enter the lists of tables and schemas in different ways and are mutually exclusive.

> **Note:**
>
> - A client application can create multiple sessions. Each session can attach to only one outbound server, and each outbound server can serve only one session at a time. However, different client application sessions can connect to different outbound servers. See *Oracle Call Interface Programmer's Guide* and *Oracle Database XStream Java API Reference* for information about attaching to an outbound server.
> - This procedure enables the outbound server that it creates.
> - Starting with Oracle Database 11*g* Release 2 (11.2.0.2), the `capture_name`, `start_scn`, and `start_time` parameters are included in this procedure.

Syntax

```
DBMS_XSTREAM_ADM.ADD_OUTBOUND(
   server_name             IN  VARCHAR2,
   queue_name              IN  VARCHAR2   DEFAULT NULL,
   source_database         IN  VARCHAR2   DEFAULT NULL,
   table_names             IN  DBMS_UTILITY.UNCL_ARRAY,
   schema_names            IN  DBMS_UTILITY.UNCL_ARRAY,
   connect_user            IN  VARCHAR2   DEFAULT NULL,
   comment                 IN  VARCHAR2   DEFAULT NULL,
   capture_name            IN  VARCHAR2   DEFAULT NULL,
   start_scn               IN  NUMBER     DEFAULT NULL,
   start_time              IN  TIMESTAMP  DEFAULT NULL,
   include_dml             IN  BOOLEAN    DEFAULT TRUE,
   include_ddl             IN  BOOLEAN    DEFAULT FALSE,
   source_root_name        IN  VARCHAR2   DEFAULT NULL,
   source_container_name   IN  VARCHAR2   DEFAULT NULL,
   lcrid_version           IN  NUMBER     DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_OUTBOUND(
   server_name             IN  VARCHAR2,
   queue_name              IN  VARCHAR2   DEFAULT NULL,
   source_database         IN  VARCHAR2   DEFAULT NULL,
   table_names             IN  VARCHAR2   DEFAULT NULL,
   schema_names            IN  VARCHAR2   DEFAULT NULL,
   connect_user            IN  VARCHAR2   DEFAULT NULL,
   comment                 IN  VARCHAR2   DEFAULT NULL,
   capture_name            IN  VARCHAR2   DEFAULT NULL,
   start_scn               IN  NUMBER     DEFAULT NULL,
   start_time              IN  TIMESTAMP  DEFAULT NULL,
   include_dml             IN  BOOLEAN    DEFAULT TRUE,
   include_ddl             IN  BOOLEAN    DEFAULT FALSE,
   source_root_name        IN  VARCHAR2   DEFAULT NULL,
   source_container_name   IN  VARCHAR2   DEFAULT NULL,
   lcrid_version           IN  NUMBER     DEFAULT NULL);
```

Parameters

Table 244-5 ADD_OUTBOUND Procedure Parameters

Parameter	Description
server_name	The name of the outbound server being created. A NULL specification is not allowed. Do not specify an owner. The specified name must not match the name of an existing outbound server, inbound server, apply process, or messaging client. **Note:** The server_name setting cannot exceed 30 bytes, and it cannot be altered after the outbound server is created.
queue_name	The name of the local queue from which the outbound server dequeues LCRs, specified as [*schema_name.*]*queue_name*. The current database must contain the queue, and the queue must be ANYDATA type. For example, to specify a queue named xstream_queue in the xstrmadmin schema, enter xstrmadmin.xstream_queue for this parameter. If the schema is not specified, then the current user is the default. If NULL, the procedure raises an error.

Table 244-5 (Cont.) ADD_OUTBOUND Procedure Parameters

Parameter	Description
source_database	The global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
table_names	The tables for which data manipulation language (DML) and data definition language (DDL) changes are streamed out to the XStream client application. The tables can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.UNCL_ARRAY, where each element is the name of a table. Specify the first table in position 1. The last position must be NULL. Each table should be specified as [*schema_name*.]*table_name*. For example, you can specify hr.employees. If the schema is not specified, then the current user is the default. **See Also:** "Usage Notes" for more information about this parameter
schema_names	The schemas for which DML and DDL changes are streamed out to the XStream client application. The schemas can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.UNCL_ARRAY, where each element is the name of a schema. Specify the first schema in position 1. The last position must be NULL. **Note:** This procedure does not concatenate the schema_names parameter with the table_names parameter. To specify tables, enter fully qualified table names in the table_names parameter (*schema_name.table_name*). **See Also:** "Usage Notes" for more information about this parameter
connect_user	The user who can attach to the specified outbound server to retrieve the LCR stream. The client application must attach to the outbound server as the specified connect user. See "CREATE_OUTBOUND Procedure" for information about the privileges required by a connect user. If NULL, then the current user is the default.
comment	An optional comment associated with the outbound server.
capture_name	The name of the capture process configured to capture changes for the outbound server. Do not specify an owner. If the specified name matches the name of an existing capture process for another outbound server, then the procedure uses the existing capture process and adds the rules for capturing changes to the database to the positive capture process rule set. If the specified name matches the name of an existing capture process for an apply process, then an error is raised. If the specified name does not match the name of an existing capture process, then an error is raised. If NULL, then the outbound server is created without a capture process.

Table 244-5 (Cont.) ADD_OUTBOUND Procedure Parameters

Parameter	Description
start_scn	A valid system change number (SCN) for the database from which the capture process starts capturing changes. If the capture_name parameter is NULL, then this parameter is ignored. If NULL and the capture_name parameter is non-NULL, then the start SCN of the capture process is not changed. An error is returned if an invalid SCN is specified. The start_scn and start_time parameters are mutually exclusive.
start_time	A valid time from which the capture process starts capturing changes. If the capture_name parameter is NULL, then this parameter is ignored. If NULL and the capture_name parameter is non-NULL, then the start SCN of the capture process is not changed. The start_scn and start_time parameters are mutually exclusive.
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.
lcrid_version	The LCRID version for captured LCRs, either 1 or 2. If 2, then the LCRs are compatible with a database with its compatibility level at 12.2.0 or higher. If 1, then the LCRs are compatible with a database with its compatibility level at 12.1.0 or lower. If NULL, the default, and the database compatibility level is 12.2.0 or higher, then the lcrid_version is set to 2 internally. If the database compatibility level is 12.1.0 or lower, then the lcrid_version is set to 1 internally.

Usage Notes

The following list describes the behavior of the outbound server for various combinations of the table_names and schema_names parameters:

- If both the table_names and schema_names parameters are NULL or empty, then the outbound server streams all DML and DDL changes to the client application.

This procedure is overloaded. The `table_names` and `schema_names` parameters are defaulted to `NULL`. Do not specify `NULL` for both `table_names` and `schema_names` in the same call; otherwise, error `PLS-00307` is returned.

- If both the `table_names` and `schema_names` parameters are specified, then the outbound server streams DML and DDL changes for the specified tables and schemas.
- If the `table_names` parameter is specified and the `schema_names` parameter is `NULL` or empty, then the outbound server streams DML and DDL changes for the specified tables.
- If the `table_names` parameter is `NULL` or empty and the `schema_names` parameter is specified, then the outbound server streams DML and DDL changes for the specified schemas.

For the procedure that uses the `DBMS_UTILITY.UNCL_ARRAY` type for the `table_names` and `schema_names` parameters, both parameters must be specified. To specify only tables, the `schema_names` parameter must be specified and empty. To specify only schemas, the `table_names` parameter must be specified and empty.

> **Note:**
>
> An empty array includes one `NULL` entry.

ADD_SCHEMA_PROPAGATION_RULES Procedure

This procedure either adds schema rules to the positive rule set for a propagation, or adds schema rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist.

This procedure is overloaded. One version of this procedure contains two `OUT` parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_SCHEMA_PROPAGATION_RULES(
   schema_name             IN   VARCHAR2,
   streams_name            IN   VARCHAR2  DEFAULT NULL,
   source_queue_name       IN   VARCHAR2,
   destination_queue_name  IN   VARCHAR2,
   include_dml             IN   BOOLEAN   DEFAULT TRUE,
   include_ddl             IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr      IN   BOOLEAN   DEFAULT FALSE,
   source_database         IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name           OUT  VARCHAR2,
   ddl_rule_name           OUT  VARCHAR2,
   inclusion_rule          IN   BOOLEAN   DEFAULT TRUE,
   and_condition           IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue          IN   BOOLEAN   DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_SCHEMA_PROPAGATION_RULES(
   schema_name             IN   VARCHAR2,
   streams_name            IN   VARCHAR2  DEFAULT NULL,
   source_queue_name       IN   VARCHAR2,
   destination_queue_name  IN   VARCHAR2,
   include_dml             IN   BOOLEAN   DEFAULT TRUE,
   include_ddl             IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr      IN   BOOLEAN   DEFAULT FALSE,
```

```
  source_database         IN   VARCHAR2  DEFAULT NULL,
  inclusion_rule          IN   BOOLEAN   DEFAULT TRUE,
  and_condition           IN   VARCHAR2  DEFAULT NULL,
  queue_to_queue          IN   BOOLEAN   DEFAULT NULL);
```

Parameters

Table 244-6 ADD_SCHEMA_PROPAGATION_RULES Procedure Parameters

Parameter	Description
schema_name	The name of the schema. For example, hr.
streams_name	The name of the propagation. Do not specify an owner. If the specified propagation does not exist, then the procedure creates it automatically. If NULL and a propagation exists for the same source queue and destination queue (including database link), then the procedure uses this propagation. If NULL and no propagation exists for the same source queue and destination queue (including database link), then the procedure creates a propagation automatically with a system-generated name.
source_queue_name	The name of the source queue, specified as [*schema_name*.]*queue_name*. The current database must contain the source queue, and the queue must be ANYDATA type. For example, to specify a source queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default.
destination_queue_name	The name of the destination queue, including a database link, specified as [*schema_name*.]*queue_name*[*@dblink_name*], if the destination queue is in a remote database. The queue must be ANYDATA type. For example, to specify a destination queue named streams_queue in the strmadmin schema and use a database link named dbs2.net, enter strmadmin.streams_queue@dbs2.net for this parameter. If the schema is not specified, then the current user is the default. If the database link is omitted, then the procedure uses the global name of the current database, and the source queue and destination queue must be in the same database. **Note:** Connection qualifiers are not allowed.
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.

Table 244-6 (Cont.) ADD_SCHEMA_PROPAGATION_RULES Procedure Parameters

Parameter	Description
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a logical change record (LCR) has a non-NULL tag. If the rules are added to the positive rule set for the propagation, then an LCR is always considered for propagation, regardless of whether it has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the propagation, then whether an LCR is discarded does not depend on the tag for the LCR. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if an LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for the propagation, then an LCR is considered for propagation only when the LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the propagation, then an LCR can be discarded only if it has a NULL tag. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.
source_database	The global name of the source database. The source database is where the change originated. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically. Oracle recommends that you specify a source database for propagation rules.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the propagation. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the propagation. In either case, the system creates the rule set if it does not exist.

Table 244-6 (Cont.) ADD_SCHEMA_PROPAGATION_RULES Procedure Parameters

Parameter	Description
and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way: (*system_condition*) AND (*and_condition*) The variable in the specified condition must be :lcr. For example, to specify that the schema rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition: :lcr.get_tag() = HEXTORAW(''02'') The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure this procedure only generates the appropriate rule. Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter. **See Also:** Logical Change Record TYPEs
queue_to_queue	If TRUE or NULL, then a new propagation created by this procedure is a queue to queue propagation. A queue-to-queue propagation always has its own propagation job and uses a service for automatic failover when the destination queue is a buffered queue in an Oracle Real Application Clusters (Oracle RAC) database. If FALSE, then a new propagation created by this procedure is a queue-to-dblink propagation. A queue-to-dblink propagation can share a propagation job with other propagations that use the same database link and does not support automatic failover in an Oracle RAC environment. This procedure cannot change the queue to queue property of an exiting propagation. If the specified propagation exists, then the procedure behaves in the following way for each setting: • If TRUE and the specified propagation is not a queue to queue propagation, then the procedure raises an error. • If FALSE and the specified propagation is a queue to queue propagation, then the procedure raises an error. • If NULL, then the procedure does not change the queue to queue property of the propagation.

Usage Notes

This procedure configures propagation using the current user. Only one propagation is allowed between a particular source queue and destination queue.

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the schema name with a sequence number appended to it. The sequence number

is used to avoid naming conflicts. If the schema name plus the sequence number is too long, then the schema name is truncated. A propagation uses the rules for filtering.

Examples

The following is an example of a schema rule condition created for DML changes:

```
((:dml.get_object_owner() = 'HR') and :dml.is_null_tag() = 'Y'
and :dml.get_source_database_name() = 'DBS1.NET' )
```

ADD_SCHEMA_RULES Procedure

This procedures adds rules to a rule set of one of the following types of XStream clients:

- When the `streams_type` parameter is set to `capture`, this procedure adds capture process rules for capturing changes to a specified schema.

  This procedure creates the specified capture process if it does not exist.

- When the `streams_type` parameter is set to `apply` and the `streams_name` parameter specifies the name of an apply process, outbound server, or inbound server, this procedure adds apply rules for applying logical change records (LCRs) that contain changes to a specified schema. The rules can specify that the LCRs must be from a particular source database.

  This procedure creates an apply process if no apply process, outbound server, or inbound server exists with the specified `streams_name`. This procedure can add rules to an outbound server or inbound server, but it cannot create an outbound server or inbound server.

This procedure is overloaded. One version of this procedure contains two `OUT` parameters, and the other does not.

> **Note:**
>
> If you add schema rules to the positive rule set for a capture process, then make sure you add rules to the negative capture process rule set to exclude database objects in the schema that are not supported by Oracle Replication. Query the `DBA_XSTREAM_OUT_SUPPORT_MODE` data dictionary view to determine which database objects are not supported by Oracle Replication. If unsupported database objects are not excluded, then capture errors will result.

Syntax

```
DBMS_XSTREAM_ADM.ADD_SCHEMA_RULES(
   schema_name             IN   VARCHAR2,
   streams_type            IN   VARCHAR2,
   streams_name            IN   VARCHAR2  DEFAULT NULL,
   queue_name              IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml             IN   BOOLEAN   DEFAULT TRUE,
   include_ddl             IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr      IN   BOOLEAN   DEFAULT FALSE,
   source_database         IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name           OUT  VARCHAR2,
   ddl_rule_name           OUT  VARCHAR2,
   inclusion_rule          IN   BOOLEAN   DEFAULT TRUE,
   and_condition           IN   VARCHAR2  DEFAULT NULL,
   source_root_name        IN  VARCHAR2  DEFAULT NULL,
```

```
   source_container_name  IN  VARCHAR2  DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_SCHEMA_RULES(
   schema_name            IN   VARCHAR2,
   streams_type           IN   VARCHAR2,
   streams_name           IN   VARCHAR2  DEFAULT NULL,
   queue_name             IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml            IN   BOOLEAN   DEFAULT TRUE,
   include_ddl            IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr     IN   BOOLEAN   DEFAULT FALSE,
   source_database        IN   VARCHAR2  DEFAULT NULL,
   inclusion_rule         IN   BOOLEAN   DEFAULT TRUE,
   and_condition          IN   VARCHAR2  DEFAULT NULL,
   source_root_name       IN  VARCHAR2  DEFAULT NULL,
   source_container_name  IN  VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-7 ADD_SCHEMA_RULES Procedure Parameters

Parameter	Description
schema_name	The name of the schema. For example, hr. You can specify a schema that does not yet exist, because Oracle Replication does not validate the existence of the schema.
streams_type	The type of XStream client: • Specify capture for a capture process. • Specify apply for an apply process.
streams_name	The name of the capture process or apply process. Do not specify an owner. If NULL, if streams_type is capture, and if one relevant capture process for the queue exists, then the relevant XStream client is used. If no relevant XStream client exists for the queue, then an XStream client is created automatically with a system-generated name. If NULL and multiple XStream clients of the specified streams_type for the queue exist, then the procedure raises an error. If NULL, if streams_type is apply, and if one relevant apply process exists, then the procedure uses the relevant apply process. The relevant apply process is identified in one of the following ways: • If one existing apply process has the source database specified in source_database and uses the queue specified in queue_name, then the procedure uses this apply process. • If source_database is NULL and one existing apply process is using the queue specified in queue_name, then the procedure uses this apply process. If NULL and no relevant apply process exists, then the procedure creates an apply process automatically with a system-generated name. If NULL and multiple relevant apply processes exist, then the procedure raises an error. Each apply process must have a unique name.

Table 244-7 (Cont.) ADD_SCHEMA_RULES Procedure Parameters

Parameter	Description
queue_name	The name of the local queue, specified as [*schema_name.*]*queue_name*. The current database must contain the queue, and the queue must be ANYDATA type. For example, to specify a queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default. For capture process rules, this is the queue into which a capture process enqueues LCRs. For outbound server rules, this is the queue from which the outbound server dequeues LCRs. For inbound server rules, this is the queue into which an inbound server enqueues error transactions.
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a redo entry or LCR has a non-NULL tag. If the rules are added to the positive rule set for the process, then a redo entry is always considered for capture, and an LCR is always considered for apply, regardless of whether the redo entry or LCR has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the process, then whether a redo entry or LCR is discarded does not depend on the tag. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if a redo entry or LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for the process, then a redo entry is considered for capture, and an LCR is considered for apply, only when the redo entry or LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the process, then a redo entry or LCR can be discarded only if it has a NULL tag. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.

Table 244-7 (Cont.) ADD_SCHEMA_RULES Procedure Parameters

Parameter	Description
source_database	The global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. For capture process rules, specify NULL or the global name of the local database if you are creating a capture process locally at the source database. If you are adding rules to a downstream capture process rule set at a downstream database, then specify the source database of the changes that will be captured. For apply process rules, specify the source database of the changes that will be applied by the apply process. The source database is the database where the changes originated. If an apply process applies captured LCRs, then the apply process can apply LCRs from only one capture process at one source database. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the XStream client. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the XStream client. In either case, the system creates the rule set if it does not exist.

and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way:
	`(system_condition) AND (and_condition)`
	The variable in the specified condition must be :lcr. For example, to specify that the schema rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition:
	`:lcr.get_tag() = HEXTORAW(''02'')`
	The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure this procedure only generates the appropriate rule.
	Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter.
	See Also: Logical Change Record TYPEs

Table 244-7 (Cont.) ADD_SCHEMA_RULES Procedure Parameters

Parameter	Description
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.

Usage Notes

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the schema name with a sequence number appended to it. The sequence number is used to avoid naming conflicts. If the schema name plus the sequence number is too long, then the schema name is truncated. A capture process or apply process uses the rules for filtering.

See Also:

- "Operational Notes"
- "Security Model"

Examples

The following is an example of a schema rule condition created for DML changes:

```
((:dml.get_object_owner() = 'HR') and :dml.is_null_tag() = 'Y'
and :dml.get_source_database_name() = 'DBS1.NET' )
```

ADD_SUBSET_OUTBOUND_RULES Procedure

This procedure adds subset rules to an outbound server configuration. Subset rules instruct the outbound server to stream out a subset of the changes to the specified tables. Outbound servers can stream out a subset of both rows and columns.

This procedure is overloaded. One column_list parameter is type VARCHAR2 and the other column_list parameter is type DBMS_UTILITY.LNAME_ARRAY. These parameters enable you to enter the list of columns in different ways and are mutually exclusive.

> **Note:**
>
> This procedure does not add rules to the outbound server's capture process.

Syntax

```
DBMS_XSTREAM_ADM.ADD_SUBSET_OUTBOUND_RULES(
   server_name      IN VARCHAR2,
   table_name       IN VARCHAR2,
   condition        IN VARCHAR2  DEFAULT NULL,
   column_list      IN DBMS_UTILITY.LNAME_ARRAY,
   keep             IN BOOLEAN   DEFAULT TRUE,
   source_database  IN VARCHAR2  DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_SUBSET_OUTBOUND_RULES(
   server_name      IN VARCHAR2,
   table_name       IN VARCHAR2,
   condition        IN VARCHAR2  DEFAULT NULL,
   column_list      IN VARCHAR2  DEFAULT NULL,
   keep             IN BOOLEAN   DEFAULT TRUE,
   source_database  IN VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-8 ADD_SUBSET_OUTBOUND_RULES Procedure Parameters

Parameter	Description
`server_name`	The name of the outbound server to which rules are being added. Specify an existing outbound server. Do not specify an owner.
`table_name`	The name of the table specified as `[schema_name.]object_name`. For example, you can specify `hr.employees`. If the schema is not specified, then the current user is the default. If the outbound server configuration uses a local capture process, then the table must exist at the local source database. If the outbound server configuration uses a downstream capture process, then the table must exist at both the source database and at the downstream capture database. The specified table cannot have any LOB, `LONG`, or `LONG RAW` columns currently or in the future.
`condition`	The subset condition. Specify this condition similar to the way you specify conditions in a `WHERE` clause in SQL. For example, to specify rows in the `hr.employees` table where the `salary` is greater than `4000` and the `job_id` is `SA_MAN`, enter the following as the condition: `' salary > 4000 and job_id = ''SA_MAN'' '` If `NULL`, then the procedure raises an error. **Note:** The quotation marks in the preceding example are all single quotation marks.

Table 244-8 (Cont.) ADD_SUBSET_OUTBOUND_RULES Procedure Parameters

Parameter	Description
column_list	The list of columns either to include in the outbound server configuration or to exclude from the outbound server configuration. Whether the columns are included or excluded depends on the setting for the keep parameter. The columns can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.LNAME_ARRAY, where each element is the name of a column. Specify the first column in position 1. The last position must be NULL. To include or exclude all of the columns in a table, specify each column in the table in the list or array. If NULL, then the procedure raises an error.
keep	If TRUE, then the columns specified in the column_list parameter are kept as part of the outbound server configuration. Therefore, changes to these columns that satisfy the condition in the condition parameter are streamed to the outbound server's client application. If FALSE, then the columns specified in the column_list parameter are excluded from the outbound server configuration. Therefore, changes to these columns are not streamed to the outbound server's client application. **See Also:** "Usage Notes"
source_database	The global name of the container where the specified table_names and schema_names are located. If non-NULL, then a condition is added to the outbound server's rules to filter the LCRs based on the global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically.

Usage Notes

When the keep parameter is set to TRUE, this procedure creates a keep columns declarative rule-based transformation for the columns listed in column_list.

When the keep parameter is set to FALSE, this procedure creates a delete column declarative rule-based transformation for each column listed in column_list.

> **See Also:**
>
> *Oracle Database XStream Guide* for information about declarative rule-based transformations

ADD_SUBSET_PROPAGATION_RULES Procedure

This procedures adds propagation rules that propagate the logical change records (LCRs) related to a subset of the rows in the specified table in a source queue to a destination queue, and creates the specified propagation if it does not exist.

This procedure is overloaded. One version of this procedure contains three `OUT` parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_SUBSET_PROPAGATION_RULES(
   table_name                IN   VARCHAR2,
   dml_condition             IN   VARCHAR2,
   streams_name              IN   VARCHAR2  DEFAULT NULL,
   source_queue_name         IN   VARCHAR2,
   destination_queue_name    IN   VARCHAR2,
   include_tagged_lcr        IN   BOOLEAN   DEFAULT FALSE,
   source_database           IN   VARCHAR2  DEFAULT NULL,
   insert_rule_name          OUT  VARCHAR2,
   update_rule_name          OUT  VARCHAR2,
   delete_rule_name          OUT  VARCHAR2,
   queue_to_queue            IN   BOOLEAN   DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_SUBSET_PROPAGATION_RULES(
   table_name                IN   VARCHAR2,
   dml_condition             IN   VARCHAR2,
   streams_name              IN   VARCHAR2  DEFAULT NULL,
   source_queue_name         IN   VARCHAR2,
   destination_queue_name    IN   VARCHAR2,
   include_tagged_lcr        IN   BOOLEAN   DEFAULT FALSE,
   source_database           IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue            IN   BOOLEAN   DEFAULT NULL);
```

Parameters

Table 244-9 ADD_SUBSET_PROPAGATION_RULES Procedure Parameters

Parameter	Description
`table_name`	The name of the table specified as `[schema_name.]object_name`. For example, `hr.employees`. If the schema is not specified, then the current user is the default. The specified table must exist in the same database as the propagation. Also, the specified table cannot have any LOB, `LONG`, `LONG RAW`, or `XMLType` columns currently or in the future.
`dml_condition`	The subset condition. Specify this condition similar to the way you specify conditions in a `WHERE` clause in SQL. For example, to specify rows in the `hr.employees` table where the `salary` is greater than `4000` and the `job_id` is `SA_MAN`, enter the following as the condition: `' salary > 4000 and job_id = ''SA_MAN'' '` **Note:** The quotation marks in the preceding example are all single quotation marks.

Table 244-9 (Cont.) ADD_SUBSET_PROPAGATION_RULES Procedure Parameters

Parameter	Description
streams_name	The name of the propagation. Do not specify an owner. If the specified propagation does not exist, then the procedure creates it automatically. If NULL and a propagation exists for the same source queue and destination queue (including database link), then the procedure uses this propagation. If NULL and no propagation exists for the same source queue and destination queue (including database link), then the procedure creates a propagation automatically with a system-generated name.
source_queue_name	The name of the source queue, specified as [*schema_name*.]*queue_name*. The current database must contain the source queue, and the queue must be ANYDATA type. For example, to specify a source queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default.
destination_queue_name	The name of the destination queue, including a database link, specified as [*schema_name*.]*queue_name*[*@dblink_name*], if the destination queue is in a remote database. The queue must be ANYDATA type. For example, to specify a destination queue named streams_queue in the strmadmin schema and use a database link named dbs2.net, enter strmadmin.streams_queue@dbs2.net for this parameter. If the schema is not specified, then the current user is the default. If the database link is omitted, then the procedure uses the global name of the current database, and the source queue and destination queue must be in the same database. **Note:** Connection qualifiers are not allowed.
include_tagged_lcr	If TRUE, then an LCR is always considered for propagation, regardless of whether it has a non-NULL tag. This setting is appropriate for a full (for example, standby) copy of a database. If FALSE, then an LCR is considered for propagation only when the LCR contains a NULL tag. A setting of FALSE is often specified in update-anywhere configurations to avoid sending a change back to its source database.
source_database	The global name of the source database. The source database is where the change originated. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically. Oracle recommends that you specify a source database for propagation rules.
insert_rule_name	Contains the system-generated INSERT rule name. This rule handles inserts and updates that must be converted into inserts.

Table 244-9 (Cont.) ADD_SUBSET_PROPAGATION_RULES Procedure Parameters

Parameter	Description
update_rule_name	Contains the system-generated UPDATE rule name. This rule handles updates that remain updates.
delete_rule_name	Contains the system-generated DELETE rule name. This rule handles deletes and updates that must be converted into deletes
queue_to_queue	If TRUE or NULL, then a new propagation created by this procedure is a queue to queue propagation. A queue-to-queue propagation always has its own propagation job and uses a service for automatic failover when the destination queue is a buffered queue in an Oracle Real Application Clusters (Oracle RAC) database. If FALSE, then a new propagation created by this procedure is a queue-to-dblink propagation. A queue-to-dblink propagation can share a propagation job with other propagations that use the same database link and does not support automatic failover in an Oracle RAC environment. This procedure cannot change the queue to queue property of an exiting propagation. If the specified propagation exists, then the procedure behaves in the following way for each setting: • If TRUE and the specified propagation is not a queue to queue propagation, then the procedure raises an error. • If FALSE and the specified propagation is a queue to queue propagation, then the procedure raises an error. • If NULL, then the procedure does not change the queue to queue property of the propagation.

Usage Notes

This procedure configures propagation using the current user. Only one propagation is allowed between a particular source queue and destination queue.

Running this procedure generates three rules for the specified propagation: one for INSERT statements, one for UPDATE statements, and one for DELETE statements. For INSERT and DELETE statements, only row LCRs that satisfy the condition specified for the dml_condition parameter are propagated. For UPDATE statements, the following variations are possible:

- If both the new and old values in a row LCR satisfy the specified dml_condition, then the row LCR is propagated without any changes.
- If neither the new or old values in a row LCR satisfy the specified dml_condition, then the row LCR is not propagated.
- If the old values for a row LCR satisfy the specified dml_condition, but the new values do not, then the update row LCR is converted into a delete row LCR.
- If the new values for a row LCR satisfy the specified dml_condition, but the old values do not, then the update row LCR is converted to an insert row LCR.

When an update is converted into an insert or a delete, it is called row migration.

A propagation uses the rules for filtering. If the propagation does not have a positive rule set, then the procedure creates a positive rule set automatically, and the rules for propagating changes to the table are added to the positive rule set. A subset rule can be added to positive rule set only, not to a negative rule set. Other rules in an existing positive rule set for the

propagation are not affected. Additional rules can be added using either the DBMS_XSTREAM_ADM package or the DBMS_RULE_ADM package.

Rules for INSERT, UPDATE, and DELETE statements are created automatically when you run this procedure, and these rules are given a system-generated rule name. Each rule has a system-generated rule name that consists of the table name with a sequence number appended to it. The sequence number is used to avoid naming conflicts. If the table name plus the sequence number is too long, then the table name is truncated. The ADD_SUBSET_RULES procedure is overloaded, and the system-generated rule names for INSERT, UPDATE, and DELETE statements are returned.

When you create propagation subset rules for a table, you should create an unconditional supplemental log group at the source database with all the columns in the table. Supplemental logging is required if an update must be converted to an insert. The propagation rule must have all the column values to be able to perform this conversion correctly.

> **Note:**
>
> Subset rules should only reside in positive rule sets. You should not add subset rules to negative rule sets. Doing so might have unpredictable results because row migration would not be performed on LCRs that are not discarded by the negative rule set.

Examples

The following is an example of a rule condition created for filtering a row LCR containing an update operation when the dml_condition is region_id = 2, the table_name is hr.regions, and the source_database is dbs1.net:

```
:dml.get_object_owner()='HR' AND :dml.get_object_name()='REGIONS'
AND :dml.is_null_tag()='Y' AND :dml.get_source_database_name()='DBS1.NET'
AND :dml.get_command_type()='UPDATE'
AND (:dml.get_value('NEW','"REGION_ID"') IS NOT NULL)
AND (:dml.get_value('OLD','"REGION_ID"') IS NOT NULL)
AND (:dml.get_value('OLD','"REGION_ID"').AccessNumber()=2)
AND (:dml.get_value('NEW','"REGION_ID"').AccessNumber()=2)
```

ADD_SUBSET_RULES Procedure

This procedure adds rules to an XStream client.

It adds rules to a rule set of one of the following types of XStream clients:

- When the streams_type parameter is set to capture, this procedure adds capture process rules for capturing changes to a subset of rows in a specified table.

  This procedure creates the specified capture process if it does not exist.

- When the streams_type parameter is set to apply and the streams_name parameter specifies the name of an apply process, outbound server, or inbound server, this procedure adds apply rules for applying logical change records (LCRs) that contain changes to a subset of rows in a specified table. The rules can specify that the LCRs must be from a particular source database.

  This procedure creates an apply process if no apply process, outbound server, or inbound server exists with the specified streams_name. This procedure can add rules to an

outbound server or inbound server, but it cannot create an outbound server or inbound server.

This procedure is overloaded. One version of this procedure contains three `OUT` parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_SUBSET_RULES(
   table_name             IN   VARCHAR2,
   dml_condition          IN   VARCHAR2,
   streams_type           IN   VARCHAR2 DEFAULT 'apply',
   streams_name           IN   VARCHAR2 DEFAULT NULL,
   queue_name             IN   VARCHAR2 DEFAULT 'streams_queue',
   include_tagged_lcr     IN   BOOLEAN  DEFAULT FALSE,
   source_database        IN   VARCHAR2 DEFAULT NULL,
   insert_rule_name       OUT  VARCHAR2,
   update_rule_name       OUT  VARCHAR2,
   delete_rule_name       OUT  VARCHAR2,
   source_root_name       IN  VARCHAR2  DEFAULT NULL,
   source_container_name  IN  VARCHAR2  DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_SUBSET_RULES(
   table_name             IN   VARCHAR2,
   dml_condition          IN   VARCHAR2,
   streams_type           IN   VARCHAR2 DEFAULT 'apply',
   streams_name           IN   VARCHAR2 DEFAULT NULL,
   queue_name             IN   VARCHAR2 DEFAULT 'streams_queue',
   include_tagged_lcr     IN   BOOLEAN  DEFAULT FALSE,
   source_database        IN   VARCHAR2 DEFAULT NULL,
   source_root_name       IN  VARCHAR2  DEFAULT NULL,
   source_container_name  IN  VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-10 ADD_SUBSET_RULES Procedure Parameters

Parameter	Description
`table_name`	The name of the table specified as `[schema_name.]object_name`. For example, `hr.employees`. If the schema is not specified, then the current user is the default. The specified table must exist in the same database as the capture process or apply process. Also, the specified table cannot have any LOB, `LONG`, `LONG RAW`, or `XMLType` columns currently or in the future.
`dml_condition`	The subset condition. Specify this condition similar to the way you specify conditions in a `WHERE` clause in SQL. For example, to specify rows in the `hr.employees` table where the `salary` is greater than `4000` and the `job_id` is `SA_MAN`, enter the following as the condition: `' salary > 4000 and job_id = ''SA_MAN'' '` **Note:** The quotation marks in the preceding example are all single quotation marks.
`streams_type`	The type of XStream client: • Specify `capture` for a capture process. • Specify `apply` for an apply process.

Table 244-10 (Cont.) ADD_SUBSET_RULES Procedure Parameters

Parameter	Description
streams_name	The name of the capture process or apply process. Do not specify an owner. If NULL, if streams_type is capture, and if one relevant capture process for the queue exists, then the procedure uses the relevant XStream client. If no relevant XStream client exists for the queue, then the procedure creates an XStream client automatically with a system-generated name. If NULL and multiple XStream clients of the specified streams_type for the queue exist, then the procedure raises an error. If NULL, if streams_type is apply, and if one relevant apply process exists, then the procedure uses the relevant apply process. The relevant apply process is identified in one of the following ways: • If one existing apply process has the source database specified in source_database and uses the queue specified in queue_name, then the procedure uses this apply process. • If source_database is NULL and one existing apply process is using the queue specified in queue_name, then the procedure uses this apply process. If NULL and no relevant apply process exists, then the procedure creates an apply process automatically with a system-generated name. If NULL and multiple relevant apply processes exist, then the procedure raises an error. Each apply process must have a unique name.
queue_name	The name of the local queue, specified as [*schema_name.*]*queue_name*. The current database must contain the queue, and the queue must be ANYDATA type. For example, to specify a queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default. For capture process rules, this is the queue into which a capture process enqueues LCRs. For outbound server rules, this is the queue from which the outbound server dequeues LCRs. For inbound server rules, this is the queue into which an inbound server enqueues error transactions.
include_tagged_lcr	If TRUE, then the XStream client performs its action regardless of the tag: • A redo entry is always considered for capture by a capture process, regardless of whether the redo entry has a non-NULL tag. • An LCR is always considered for apply by an apply process, regardless of whether redo entry or LCR has a non-NULL tag. If FALSE, then an XStream client performs its action only when the tag is NULL: • A redo entry is considered for capture by a capture process only when the redo entry contains a NULL tag. • An LCR is considered for apply by an apply process only if the LCR contains a NULL tag. A setting of FALSE is often specified in update-anywhere configurations to avoid sending a change back to its source database.

Table 244-10 (Cont.) ADD_SUBSET_RULES Procedure Parameters

Parameter	Description
source_database	The global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. For capture process rules, specify NULL or the global name of the local database if you are creating a capture process locally at the source database. If you are adding rules to a downstream capture process rule set at a downstream database, then specify the source database of the changes that will be captured. For apply process rules, specify the source database of the changes that will be applied by the apply process. The source database is the database where the changes originated. If an apply process applies captured LCRs, then the apply process can apply LCRs from only one capture process at one source database. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
insert_rule_name	Contains the system-generated INSERT rule name. This rule handles inserts and updates that must be converted into inserts.
update_rule_name	Contains the system-generated UPDATE rule name. This rule handles updates that remain updates.
delete_rule_name	Contains the system-generated DELETE rule name. This rule handles deletes and updates that must be converted into deletes
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.

Usage Notes

Running this procedure generates three rules for the specified capture process or apply process: one for INSERT statements, one for UPDATE statements, and one for DELETE statements. For INSERT and DELETE statements, only DML changes that satisfy the condition specified for the dml_condition parameter are captured or applied. For UPDATE statements, the following variations are possible:

- If both the new and old values in a DML change satisfy the specified `dml_condition`, then the DML change is captured or applied without any changes.
- If neither the new or old values in a DML change satisfy the specified `dml_condition`, then the DML change is not captured or applied.
- If the old values for a DML change satisfy the specified `dml_condition`, but the new values do not, then the DML change is converted into a delete.
- If the new values for a DML change satisfy the specified `dml_condition`, but the old values do not, then the DML change is converted to an insert.

When an update is converted into an insert or a delete, it is called row migration.

A capture process or apply process uses the rules for filtering. If the XStream client does not have a positive rule set, then this procedure creates a positive rule set automatically, and adds the rules for the table to the positive rule set. A subset rule can be added to positive rule set only, not to a negative rule set. Other rules in an existing rule set for the process are not affected. Additional rules can be added using either the `DBMS_XSTREAM_ADM` package or the `DBMS_RULE_ADM` package.

Rules for `INSERT`, `UPDATE`, and `DELETE` statements are created automatically when you run this procedure, and these rules are given a system-generated rule name. Each rule has a system-generated rule name that consists of the table name with a sequence number appended to it. The sequence number is used to avoid naming conflicts. If the table name plus the sequence number is too long, then the table name is truncated. The `ADD_SUBSET_RULES` procedure is overloaded, and the system-generated rule names for `INSERT`, `UPDATE`, and `DELETE` statements are returned.

> **Note:**
>
> Subset rules should only reside in positive rule sets. You should not add subset rules to negative rule sets. Doing so might have unpredictable results because row migration would not be performed on LCRs that are not discarded by the negative rule set.

Examples

The following is an example of a rule condition created for filtering DML changes containing an update operation when the `dml_condition` is `region_id = 2`, the `table_name` is `hr.regions`, and the `source_database` is `dbs1.net`:

```
:dml.get_object_owner()='HR' AND :dml.get_object_name()='REGIONS'
AND :dml.is_null_tag()='Y' AND :dml.get_source_database_name()='DBS1.NET'
AND :dml.get_command_type()='UPDATE'
AND (:dml.get_value('NEW','"REGION_ID"') IS NOT NULL)
AND (:dml.get_value('OLD','"REGION_ID"') IS NOT NULL)
AND (:dml.get_value('OLD','"REGION_ID"').AccessNumber()=2)
AND (:dml.get_value('NEW','"REGION_ID"').AccessNumber()=2)
```

ADD_TABLE_PROPAGATION_RULES Procedure

This procedures adds table rules to the positive rule set for a propagation, or adds table rules to the negative rule set for a propagation, and creates the specified propagation if it does not exist.

This procedure is overloaded. One version of this procedure contains two `OUT` parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_TABLE_PROPAGATION_RULES(
   table_name              IN   VARCHAR2,
   streams_name            IN   VARCHAR2  DEFAULT NULL,
   source_queue_name       IN   VARCHAR2,
   destination_queue_name  IN   VARCHAR2,
   include_dml             IN   BOOLEAN   DEFAULT TRUE,
   include_ddl             IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr      IN   BOOLEAN   DEFAULT FALSE,
   source_database         IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name           OUT  VARCHAR2,
   ddl_rule_name           OUT  VARCHAR2,
   inclusion_rule          IN   BOOLEAN   DEFAULT TRUE,
   and_condition           IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue          IN   BOOLEAN   DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_TABLE_PROPAGATION_RULES(
   table_name              IN   VARCHAR2,
   streams_name            IN   VARCHAR2  DEFAULT NULL,
   source_queue_name       IN   VARCHAR2,
   destination_queue_name  IN   VARCHAR2,
   include_dml             IN   BOOLEAN   DEFAULT TRUE,
   include_ddl             IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr      IN   BOOLEAN   DEFAULT FALSE,
   source_database         IN   VARCHAR2  DEFAULT NULL,
   inclusion_rule          IN   BOOLEAN   DEFAULT TRUE,
   and_condition           IN   VARCHAR2  DEFAULT NULL,
   queue_to_queue          IN   BOOLEAN   DEFAULT NULL);
```

Parameters

Table 244-11 ADD_TABLE_PROPAGATION_RULES Procedure Parameters

Parameter	Description
`table_name`	The name of the table specified as `[`*`schema_name`*`.]`*`table_name`*. For example, `hr.employees`. If the schema is not specified, then the current user is the default.
`streams_name`	The name of the propagation. Do not specify an owner. If the specified propagation does not exist, then the procedure creates it automatically. If `NULL` and a propagation exists for the same source queue and destination queue (including database link), then the procedure uses this propagation. If `NULL` and no propagation exists for the same source queue and destination queue (including database link), then the procedure creates a propagation automatically with a system-generated name.

Table 244-11 (Cont.) ADD_TABLE_PROPAGATION_RULES Procedure Parameters

Parameter	Description
source_queue_name	The name of the source queue, specified as [*schema_name.*]*queue_name*. The current database must contain the source queue, and the queue must be ANYDATA type. For example, to specify a source queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default.
destination_queue_name	The name of the destination queue, including a database link, specified as [*schema_name.*]*queue_name*[*@dblink_name*], if the destination queue is in a remote database. The queue must be ANYDATA type. For example, to specify a destination queue named streams_queue in the strmadmin schema and use a database link named dbs2.net, enter strmadmin.streams_queue@dbs2.net for this parameter. If the schema is not specified, then the current user is the default. If the database link is omitted, then the procedure uses the global name of the current database, and the source queue and destination queue must be in the same database. **Note:** Connection qualifiers are not allowed.
include_dml	If TRUE, then the procedure creates a rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted. The generated rule evaluates to TRUE for any DDL change that operates on the table or on an object that is part of the table, such as an index or trigger on the table. The rule evaluates to FALSE for any DDL change that either does not refer to the table or refers to the table in a subordinate way. For example, the rule evaluates to FALSE for changes that create synonyms or views based on the table. The rule also evaluates to FALSE for a change to a PL/SQL subprogram that refers to the table.

Table 244-11 (Cont.) ADD_TABLE_PROPAGATION_RULES Procedure Parameters

Parameter	Description
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a logical change record (LCR) has a non-NULL tag. If the rules are added to the positive rule set for the propagation, then an LCR is always considered for propagation, regardless of whether it has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the propagation, then whether an LCR is discarded does not depend on the tag for the LCR. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if an LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for the propagation, then an LCR is considered for propagation only when the LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the propagation, then an LCR can be discarded only if it has a NULL tag. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.
source_database	The global name of the source database. The source database is where the change originated. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically. Oracle recommends that you specify a source database for propagation rules.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the propagation. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the propagation. In either case, the system creates the rule set if it does not exist.

Table 244-11 (Cont.) ADD_TABLE_PROPAGATION_RULES Procedure Parameters

Parameter	Description
and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way: (*system_condition*) AND (*and_condition*) The variable in the specified condition must be :lcr. For example, to specify that the table rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition: :lcr.get_tag() = HEXTORAW(''02'') The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure this procedure only generates the appropriate rule. Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter. **See Also:** Logical Change Record TYPEs
queue_to_queue	If TRUE or NULL, then a new propagation created by this procedure is a queue to queue propagation. A queue-to-queue propagation always has its own propagation job and uses a service for automatic failover when the destination queue is a buffered queue in an Oracle Real Application Clusters (Oracle RAC) database. If FALSE, then a new propagation created by this procedure is a queue-to-dblink propagation. A queue-to-dblink propagation can share a propagation job with other propagations that use the same database link and does not support automatic failover in an Oracle RAC environment. This procedure cannot change the queue to queue property of an exiting propagation. If the specified propagation exists, then the procedure behaves in the following way for each setting: • If TRUE and the specified propagation is not a queue to queue propagation, then the procedure raises an error. • If FALSE and the specified propagation is a queue to queue propagation, then the procedure raises an error. • If NULL, then the procedure does not change the queue to queue property of the propagation.

Usage Notes

This procedure configures propagation using the current user. Only one propagation is allowed between a particular source queue and destination queue.

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the table name with a sequence number appended to it. The sequence number is

used to avoid naming conflicts. If the table name plus the sequence number is too long, then the table name is truncated. A propagation uses the rules for filtering.

Examples

The following is an example of a table rule condition created for filtering DML statements:

```
(((:dml.get_object_owner() = 'HR' and :dml.get_object_name() = 'LOCATIONS'))
and :dml.is_null_tag() = 'Y' and :dml.get_source_database_name() = 'DBS1.NET' )
```

ADD_TABLE_RULES Procedure

This procedure adds rules to an XStream client rule set.

It adds rules to a rule set of one of the following types of XStream clients:

- When the `streams_type` parameter is set to `capture`, this procedure adds capture process rules for capturing changes to a specified table.

  This procedure creates the specified capture process if it does not exist.

- When the `streams_type` parameter is set to `apply` and the `streams_name` parameter specifies the name of an apply process, outbound server, or inbound server, this procedure adds apply rules for applying logical change records (LCRs) that contain changes to a specified table. The rules can specify that the LCRs must be from a particular source database.

  This procedure creates an apply process if no apply process, outbound server, or inbound server exists with the specified `streams_name`. This procedure can add rules to an outbound server or inbound server, but it cannot create an outbound server or inbound server.

This procedure is overloaded. One version of this procedure contains two `OUT` parameters, and the other does not.

Syntax

```
DBMS_XSTREAM_ADM.ADD_TABLE_RULES(
   table_name             IN   VARCHAR2,
   streams_type           IN   VARCHAR2,
   streams_name           IN   VARCHAR2  DEFAULT NULL,
   queue_name             IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml            IN   BOOLEAN   DEFAULT TRUE,
   include_ddl            IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr     IN   BOOLEAN   DEFAULT FALSE,
   source_database        IN   VARCHAR2  DEFAULT NULL,
   dml_rule_name          OUT  VARCHAR2,
   ddl_rule_name          OUT  VARCHAR2,
   inclusion_rule         IN   BOOLEAN   DEFAULT TRUE,
   and_condition          IN   VARCHAR2  DEFAULT NULL,
   source_root_name       IN  VARCHAR2   DEFAULT NULL,
   source_container_name  IN  VARCHAR2   DEFAULT NULL);

DBMS_XSTREAM_ADM.ADD_TABLE_RULES(
   table_name             IN   VARCHAR2,
   streams_type           IN   VARCHAR2,
   streams_name           IN   VARCHAR2  DEFAULT NULL,
   queue_name             IN   VARCHAR2  DEFAULT 'streams_queue',
   include_dml            IN   BOOLEAN   DEFAULT TRUE,
   include_ddl            IN   BOOLEAN   DEFAULT FALSE,
   include_tagged_lcr     IN   BOOLEAN   DEFAULT FALSE,
   source_database        IN   VARCHAR2  DEFAULT NULL,
```

```
  inclusion_rule         IN   BOOLEAN   DEFAULT TRUE,
  and_condition          IN   VARCHAR2  DEFAULT NULL,
  source_root_name       IN  VARCHAR2   DEFAULT NULL,
  source_container_name  IN  VARCHAR2   DEFAULT NULL);
```

Parameters

Table 244-12 ADD_TABLE_RULES Procedure Parameters

Parameter	Description
table_name	The name of the table specified as [*schema_name.*]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default. You can specify a table that does not yet exist, because Oracle Replication does not validate the existence of the table.
streams_type	The type of XStream client: • Specify capture for a capture process. • Specify apply for an apply process.
streams_name	The name of the capture process or apply process. Do not specify an owner. If NULL, if streams_type is capture, and if one relevant capture process for the queue exists, then the procedure uses the relevant XStream client. If no relevant XStream client exists for the queue, then the procedure creates an XStream client automatically with a system-generated name. If NULL and multiple XStream clients of the specified streams_type for the queue exist, then the procedure raises an error. If NULL, if streams_type is apply, and if one relevant apply process exists, then the procedure uses the relevant apply process. The relevant apply process is identified in one of the following ways: • If one existing apply process has the source database specified in source_database and uses the queue specified in queue_name, then the procedure uses this apply process. • If source_database is NULL and one existing apply process is using the queue specified in queue_name, then the procedure uses this apply process. If NULL and no relevant apply process exists, then the procedure creates an apply process automatically with a system-generated name. If NULL and multiple relevant apply processes exist, then the procedure raises an error. Each apply process must have a unique name.
queue_name	The name of the local queue, specified as [*schema_name.*]*queue_name*. The current database must contain the queue, and the queue must be ANYDATA type. For example, to specify a queue named streams_queue in the strmadmin schema, enter strmadmin.streams_queue for this parameter. If the schema is not specified, then the current user is the default. For capture process rules, this is the queue into which a capture process enqueues LCRs. For outbound server rules, this is the queue from which the outbound server dequeues LCRs. For inbound server rules, this is the queue into which an inbound server enqueues error transactions.
include_dml	If TRUE, then the procedure creates a DML rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.

Table 244-12 (Cont.) ADD_TABLE_RULES Procedure Parameters

Parameter	Description
include_ddl	If TRUE, then the procedure creates a DDL rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted. The generated rule evaluates to TRUE for any DDL change that operates on the table or on an object that is part of the table, such as an index or trigger on the table. The rule evaluates to FALSE for any DDL change that either does not refer to the table or refers to the table in a subordinate way. For example, the rule evaluates to FALSE for changes that create synonyms or views based on the table. The rule also evaluates to FALSE for a change to a PL/SQL subprogram that refers to the table.
include_tagged_lcr	If TRUE, then the procedure does not add a condition regarding Oracle Replication tags to the generated rules. Therefore, these rules can evaluate to TRUE regardless of whether a redo entry, session, or LCR has a non-NULL tag. If the rules are added to the positive rule set for the XStream client, then the XStream client performs its action regardless of the tag: • A redo entry is always considered for capture by a capture process, regardless of whether the redo entry has a non-NULL tag. • An LCR is always considered for apply by an apply process, regardless of whether redo entry or LCR has a non-NULL tag. If the rules are added to a positive rule set, then setting this parameter to TRUE is appropriate for a full (for example, standby) copy of a database. If the rules are added to the negative rule set for the XStream client, then whether a database change is discarded does not depend on the tag. If FALSE, then the procedure adds a condition to each generated rule that causes the rule to evaluate to TRUE only if a redo entry, session, or LCR has a NULL Oracle Replication tag. If the rules are added to the positive rule set for an XStream client, then the XStream client performs its action only when the tag is NULL: • A redo entry is considered for capture by a capture process only when the redo entry contains a NULL tag. • An LCR is considered for apply by an apply process only if the LCR contains a NULL tag. If the rules are added to a positive rule set, then setting this parameter to FALSE might be appropriate in update-anywhere configurations to avoid sending a change back to its source database. If the rules are added to the negative rule set for the XStream client, then a database change can be discarded only if it has a NULL tag. A setting of FALSE is often specified in update-anywhere configurations to avoid sending a change back to its source database. Usually, specify TRUE for this parameter if the inclusion_rule parameter is set to FALSE.

Table 244-12 (Cont.) ADD_TABLE_RULES Procedure Parameters

Parameter	Description
source_database	The global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. For capture process rules, specify NULL or the global name of the local database if you are creating a capture process locally at the source database. If you are adding rules to a downstream capture process rule set at a downstream database, then specify the source database of the changes that will be captured. For apply process rules, specify the source database of the changes that will be applied by the apply process. The source database is the database where the changes originated. If an apply process applies captured LCRs, then the apply process can apply LCRs from only one capture process at one source database. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
dml_rule_name	If include_dml is TRUE, then this parameter contains the DML rule name. If include_dml is FALSE, then this parameter contains a NULL.
ddl_rule_name	If include_ddl is TRUE, then this parameter contains the DDL rule name. If include_ddl is FALSE, then this parameter contains a NULL.
inclusion_rule	If inclusion_rule is TRUE, then the procedure adds the rules to the positive rule set for the XStream client. If inclusion_rule is FALSE, then the procedure adds the rules to the negative rule set for the XStream client. In either case, the system creates the rule set if it does not exist.
and_condition	If non-NULL, appends the specified condition to the system-generated rule condition using an AND clause in the following way: (*system_condition*) AND (*and_condition*) The variable in the specified condition must be :lcr. For example, to specify that the table rules generated by the procedure evaluate to TRUE only if the Oracle Replication tag is the hexadecimal equivalent of '02', specify the following condition: :lcr.get_tag() = HEXTORAW(''02'') The :lcr in the specified condition is converted to :dml or :ddl, depending on the rule that is being generated. If you are specifying an LCR member subprogram that is dependent on the LCR type (row or DDL), then make sure this procedure only generates the appropriate rule. Specifically, if you specify an LCR member subprogram that is valid only for row LCRs, then specify TRUE for the include_dml parameter and FALSE for the include_ddl parameter. If you specify an LCR member subprogram that is valid only for DDL LCRs, then specify FALSE for the include_dml parameter and TRUE for the include_ddl parameter. **See Also:** Logical Change Record TYPEs

Table 244-12 (Cont.) ADD_TABLE_RULES Procedure Parameters

Parameter	Description
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.

Usage Notes

This procedure creates DML and DDL rules automatically based on include_dml and include_ddl parameter values, respectively. Each rule has a system-generated rule name that consists of the table name with a sequence number appended to it. The sequence number is used to avoid naming conflicts. If the table name plus the sequence number is too long, then the table name is truncated. A capture process or apply process uses the rules for filtering.

See Also:

- "Operational Notes"
- "Security Model"

Examples

The following is an example of a table rule condition created for DML changes:

```
(((:dml.get_object_owner() = 'HR' and :dml.get_object_name() = 'LOCATIONS'))
and :dml.is_null_tag() = 'Y' and :dml.get_source_database_name() = 'DBS1.NET' )
```

ALTER_INBOUND Procedure

This procedure modifies an XStream inbound server.

Syntax

```
DBMS_XSTREAM_ADM.ALTER_INBOUND(
   server_name IN VARCHAR2,
   apply_user  IN VARCHAR2  DEFAULT NULL,
   comment     IN VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-13 ALTER_INBOUND Procedure Parameters

Parameter	Description
server_name	The name of the inbound server being altered. Specify an existing inbound server. Do not specify an owner.
apply_user	The user who applies all DML and DDL changes that satisfy the inbound server rule sets, who runs user-defined apply handlers, and who runs custom rule-based transformations configured for inbound server rules. The client application must attach to the inbound server as the apply user. Specify a user to change the apply user. In this case, the user who invokes the ALTER_INBOUND procedure must be granted the DBA role. Only the SYS user can set the apply_user to SYS. If NULL, then the apply user is not changed. See "CREATE_INBOUND Procedure" for information about the required privileges for an apply user.
comment	An optional comment associated with the inbound server. If non-NULL, then the specified comment replaces the existing comment. If NULL, then the existing comment is not changed.

ALTER_OUTBOUND Procedure

This procedure modifies an XStream outbound server configuration.

This procedure always alters the specified outbound server. This procedure can also alter the outbound server's capture process when either of the following conditions is met:

- The capture process was created by the CREATE_OUTBOUND procedure in this package.
- The queue used by the capture process was created by the CREATE_OUTBOUND procedure.

To check whether this procedure can alter the outbound server's capture process, query the CAPTURE_NAME column in the ALL_XSTREAM_OUTBOUND view. When the name of the capture process appears in the CAPTURE_NAME column of this view, the ALTER_OUTBOUND procedure can manage the capture process's rules or change the capture user for the capture process. When the CAPTURE_NAME column of this view is NULL, the ALTER_OUTBOUND procedure cannot manage the capture process.

This procedure is overloaded. One table_names parameter is type VARCHAR2 and the other table_names parameter is type DBMS_UTILITY.UNCL_ARRAY. Also, one schema_names parameter is type VARCHAR2 and the other schema_names parameter is type DBMS_UTILITY.UNCL_ARRAY. These parameters enable you to enter the list of tables and schemas in different ways and are mutually exclusive.

> **Note:**
>
> Starting with Oracle Database 11*g* Release 2 (11.2.0.2), the start_scn and start_time parameters are included in this procedure.

Syntax

```
DBMS_XSTREAM_ADM.ALTER_OUTBOUND(
   server_name           IN VARCHAR2,
   table_names           IN DBMS_UTILITY.UNCL_ARRAY,
   schema_names          IN DBMS_UTILITY.UNCL_ARRAY,
   add                   IN BOOLEAN    DEFAULT TRUE,
   capture_user          IN VARCHAR2   DEFAULT NULL,
   connect_user          IN VARCHR2    DEFAULT NULL,
   comment               IN VARCHAR2   DEFAULT NULL,
   inclusion_rule        IN BOOLEAN    DEFAULT TRUE,
   start_scn             IN NUMBER     DEFAULT NULL,
   start_time            IN TIMESTAMP  DEFAULT NULL,
   include_dml           IN BOOLEAN    DEFAULT TRUE,
   include_ddl           IN BOOLEAN    DEFAULT FALSE,
   source_database       IN VARCHAR2   DEFAULT NULL,
   source_container_name IN VARCHAR2   DEFAULT NULL);

DBMS_XSTREAM_ADM.ALTER_OUTBOUND(
   server_name           IN VARCHAR2,
   table_names           IN VARCHAR2   DEFAULT NULL,
   schema_names          IN VARCHAR2   DEFAULT NULL,
   add                   IN BOOLEAN    DEFAULT TRUE,
   capture_user          IN VARCHAR2   DEFAULT NULL,
   connect_user          IN VARCHAR2   DEFAULT NULL,
   comment               IN VARCHAR2   DEFAULT NULL,
   inclusion_rule        IN BOOLEAN    DEFAULT TRUE,
   start_scn             IN NUMBER     DEFAULT NULL,
   start_time            IN TIMESTAMP  DEFAULT NULL,
   include_dml           IN BOOLEAN    DEFAULT TRUE,
   include_ddl           IN BOOLEAN    DEFAULT FALSE,
   source_database       IN VARCHAR2   DEFAULT NULL,
   source_container_name IN VARCHAR2   DEFAULT NULL);
```

Parameters

Table 244-14 ALTER_OUTBOUND Procedure Parameters

Parameter	Description
`server_name`	The name of the outbound server being altered. Specify an existing outbound server. Do not specify an owner.
`table_names`	The tables that are either added to or removed from the XStream Out configuration. Whether the tables are added or removed depends on the setting for the `add` parameter. The tables can be specified in the following ways: • Comma-delimited list of type `VARCHAR2`. • A PL/SQL associative array of type `DBMS_UTILITY.UNCL_ARRAY`, where each element is the name of a table. Specify the first table in position 1. The last position must be `NULL`. Each table should be specified as `[`*`schema_name`*`.]`*`table_name`*. For example, `hr.employees`. If the schema is not specified, then the current user is the default. **See Also:** "Usage Notes" for more information about this parameter

Table 244-14 (Cont.) ALTER_OUTBOUND Procedure Parameters

Parameter	Description
schema_names	The schemas that are either added to or removed from the XStream Out configuration. Whether the schemas are added or removed depends on the setting for the add parameter. The schemas can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.UNCL_ARRAY, where each element is the name of a schema. Specify the first schema in position 1. The last position must be NULL. **Note:** This procedure does not concatenate the schema_names parameter with the table_names parameter. To specify tables, enter fully qualified table names in the table_names parameter (*schema_name.table_name*). **See Also:** "Usage Notes" for more information about this parameter
add	If TRUE, then the procedure adds to the XStream Out configuration the tables specified in the table_names parameter and the schemas specified in the schema_names parameter. If FALSE, then the procedure removes from the XStream Out configuration the tables specified in the table_names parameter and the schemas specified in the schema_names parameter.
capture_user	The user in whose security domain a capture process captures changes that satisfy its rule sets and runs custom rule-based transformations configured for capture process rules. Specify a user to change the capture user. In this case, the user who invokes the ALTER_OUTBOUND procedure must be granted the DBA role. Only the SYS user can set the capture_user to SYS. If NULL, then the capture user is not changed. If you change the capture user, then this procedure grants the new capture user enqueue privilege on the queue used by the capture process and configures the user as a secure queue user. Ensure that the capture user is granted the other required privileges. See "CREATE_OUTBOUND Procedure" for information about the privileges required by a capture user. The capture process is stopped and restarted automatically when you change the value of this parameter. **Note:** If the capture user for a capture process is dropped using DROP USER . . . CASCADE, then the capture process is also dropped automatically.
connect_user	The user who can attach to the specified outbound server to retrieve the change stream. The XStream client application must attach to the outbound server as the specified connect user. Specify a user to change the connect user. In this case, the user who invokes the ALTER_OUTBOUND procedure must be granted the DBA role. Only the SYS user can set the connect_user to SYS. If NULL, then the connect user is not changed. If you change the connect user, then this procedure grants the new connect user dequeue privileges on the queue used by the outbound server and configures the user as a secure queue user. Ensure that the connect user is granted the other required privileges. See "CREATE_OUTBOUND Procedure" for information about the privileges required by a connect user.

Table 244-14 (Cont.) ALTER_OUTBOUND Procedure Parameters

Parameter	Description
comment	An optional comment associated with the outbound server. If non-NULL, then the specified comment replaces the existing comment. If NULL, then the existing comment is not changed.
inclusion_rule	If TRUE and the add parameter is set to TRUE, then the procedure adds rules for the tables specified in the table_names parameter and the schemas specified in the schema_names parameter to the positive rule sets in the XStream Out configuration. When rules for tables and schemas are in positive rule sets, the XStream Out configuration streams DML and DDL changes to the tables and schemas out to the client application. If TRUE and the add parameter is set to FALSE, then the procedure removes rules for the tables specified in the table_names parameter and the schemas specified in the schema_names parameter from the positive rule sets in the XStream Out configuration. If FALSE and the add parameter is set to TRUE, then the procedure adds rules for the tables specified in the table_names parameter and the schemas specified in the schema_names parameter to the negative rule sets in the XStream Out configuration. When rules for tables and schemas are in negative rule sets, the XStream Out configuration does not stream changes to the tables and schemas out to the client application. If FALSE and the add parameter is set to FALSE, then the procedure removes rules for the tables specified in the table_names parameter and the schemas specified in the schema_names parameter from the negative rule sets in the XStream Out configuration.
start_scn	A valid SCN for the database from which the capture process starts capturing changes. To be valid, the SCN value must be greater than or equal to the first SCN for the capture process. If a valid SCN is specified, then the capture process captures changes from the specified SCN when it is restarted. An error is returned if an invalid SCN is specified. If NULL and the start_time parameter is NULL, then the start SCN is not changed. If NULL and the start_time parameter is non-NULL, then the start SCN is changed to match the specified start time. The start_scn and start_time parameters are mutually exclusive. **Note:** If the capture process is enabled, then the ALTER_OUTBOUND procedure automatically stops and restarts the capture process when the start_scn parameter is non-NULL. If the capture process is disabled, then the ALTER_OUTBOUND procedure automatically starts the capture process when the start_scn parameter is non-NULL.

Table 244-14 (Cont.) ALTER_OUTBOUND Procedure Parameters

Parameter	Description
start_time	A valid time from which the capture process starts capturing changes. To be valid, the time must correspond to an SCN value that is greater than or equal to the first SCN for the capture process. If a valid time is specified, then the capture process captures changes from the specified time when it is restarted. An error is returned if an invalid time is specified. If NULL and the start_scn parameter is NULL, then the start time is not changed. If NULL and the start_scn parameter is non-NULL, then the start time is changed to match the specified start SCN. The start_scn and start_time parameters are mutually exclusive. **Note:** If the capture process is enabled, then the ALTER_OUTBOUND procedure automatically stops and restarts the capture process when the start_time parameter is non-NULL. If the capture process is disabled, then the ALTER_OUTBOUND procedure automatically starts the capture process when the start_time parameter is non-NULL.
include_dml	If TRUE, then the procedure creates a DML rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a DDL rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.
source_database	The global name of the container where the specified table_names and schema_names are located. If source_database is non-NULL, then a condition is added to the outbound server's rules to filter the LCRs based on the global name of the source database.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.

Usage Notes

The following list describes the behavior of the outbound server for various combinations of the table_names and schema_names parameters:

- If both the table_names and schema_names parameters are NULL or empty, then no rules are changed for the XStream Out configuration.

  This procedure is overloaded. The table_names and schema_names parameters are defaulted to NULL. Do not specify NULL for both table_names and schema_names in the same call; otherwise, error PLS-00307 is returned.

- If both the table_names and schema_names parameters are specified, then the rules for the tables and schemas are added to or removed from the XStream Out configuration, depending on the setting of the add parameter.

- If the table_names parameter is specified and the schema_names parameter is NULL or empty, then the rules for the tables are added to or removed from the XStream Out configuration, depending on the setting of the add parameter. The existing rules for schemas are not changed for the XStream Out configuration.

- If the `table_names` parameter is `NULL` or empty and the `schema_names` parameter is specified, then the rules for the schemas are added to or removed from the XStream Out configuration, depending on the setting of the `add` parameter. The existing rules for tables are not changed for the XStream Out configuration.

For the procedure that uses the `DBMS_UTILITY.UNCL_ARRAY` type for the `table_names` and `schema_names` parameters, both parameters must be specified. To specify only tables, the `schema_names` parameter must be specified and empty. To specify only schemas, the `table_names` parameter must be specified and empty.

> **Note:**
>
> An empty array includes one `NULL` entry.

CREATE_INBOUND Procedure

This procedure creates an XStream inbound server and its queue.

> **Note:**
>
> A client application can create multiple sessions. Each session can attach to only one inbound server, and each inbound server can serve only one session at a time. However, different client application sessions can connect to different inbound servers. See *Oracle Call Interface Programmer's Guide* and *Oracle Database XStream Java API Reference* for information about attaching to an inbound server.

Syntax

```
DBMS_XSTREAM_ADM.CREATE_INBOUND(
   server_name IN VARCHAR2,
   queue_name  IN VARCHAR2,
   apply_user  IN VARCHAR2  DEFAULT NULL,
   comment     IN VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-15 CREATE_INBOUND Procedure Parameters

Parameter	Description
`server_name`	The name of the inbound server being created. A `NULL` specification is not allowed. Do not specify an owner. The specified name must not match the name of an existing outbound server, inbound server, apply process, or messaging client. **Note:** The `server_name` setting cannot exceed 30 bytes, and it cannot be altered after the inbound server is created.

Table 244-15 (Cont.) CREATE_INBOUND Procedure Parameters

Parameter	Description
queue_name	The name of the local queue used by the inbound server, specified as [*schema_name.*]*queue_name*. If the specified queue exists, then it is used. If the specified queue does not exist, then the procedure creates it. For example, to specify a queue named xstream_queue in the xstrmadmin schema, enter xstrmadmin.xstream_queue for this parameter. If the schema is not specified, then the current user is the default. **Note:** An inbound server's queue is used only to store error transactions.
apply_user	The apply user. If NULL, then the current user is the default. The client application must attach to the inbound server as the apply user. The apply user is the user in whose security domain an inbound server evaluates whether LCRs satisfy its rule sets, applies DML and DDL changes directly to database objects, runs custom rule-based transformations configured for inbound server rules, and runs apply handlers configured for the inbound server. This user must have the necessary privileges to perform these actions. This procedure grants the apply user dequeue privileges on the queue used by the inbound server and configures the user as a secure queue user. In addition to the privileges granted by this procedure, you must grant the following privileges to the apply user: • The necessary privileges to perform DML and DDL changes on the apply objects • EXECUTE privilege on the rule sets used by the inbound server • EXECUTE privilege on all rule-based transformation functions used in the rule set • EXECUTE privilege on all apply handler procedures You can grant these privileges directly to the apply user, or you can grant them through roles. In addition, the apply user must be granted EXECUTE privilege on all packages, including Oracle supplied packages, that are invoked in subprograms run by the inbound server. These privileges must be granted directly to the apply user. They cannot be granted through roles. **Note:** If the apply user for an inbound server is dropped using DROP USER . . . CASCADE, then the inbound server is also dropped automatically.
comment	An optional comment associated with the inbound server.

Usage Notes

By default, an inbound server does not use rules or rule sets. Therefore, an inbound server applies all of the LCRs sent to it by an XStream client application. However, to filter the LCRs sent to an inbound server, you can add rules and rule sets to an inbound server using the DBMS_XSTREAM_ADM and DBMS_RULE_ADM packages.

In a CDB, you can execute the CREATE_INBOUND procedure from either the root or a PDB. The inbound server is restricted to receiving LCRs from one source database and only applying the changes to its local container. If the inbound server is at the root level, then the apply user must be a common user.

> **See Also:**
>
> *Oracle Database XStream Guide*

CREATE_OUTBOUND Procedure

This procedure creates an XStream outbound server, queue, and capture process to enable client applications to stream out Oracle database changes.

> **Note:**
>
> A multitenant container database is the only supported architecture in Oracle Database 21c and later releases. While the documentation is being revised, legacy terminology may persist. In most cases, "database" and "non-CDB" refer to a CDB or PDB, depending on context. In some contexts, such as upgrades, "non-CDB" refers to a non-CDB from a previous release.

This procedure is overloaded. One `table_names` parameter is type `VARCHAR2` and the other `table_names` parameter is type `DBMS_UTILITY.UNCL_ARRAY`. Also, one `schema_names` parameter is type `VARCHAR2` and the other `schema_names` parameter is type `DBMS_UTILITY.UNCL_ARRAY`. These parameters enable you to enter the list of tables and schemas in different ways and are mutually exclusive.

> **Note:**
>
> - A client application can create multiple sessions. Each session can attach to only one outbound server, and each outbound server can serve only one session at a time. However, different client application sessions can connect to different outbound servers. See "OCIXStreamOutAttach()" in the *Oracle Call Interface Programmer's Guide* and *Oracle Database XStream Java API Reference* for information about attaching to an outbound server.
> - If the `capture_name` parameter is `NULL`, then this procedure automatically generates a name for the capture process that it creates.
> - This procedure automatically generates a name for the queue that it creates.
> - This procedure enables both the capture process and outbound server that it creates.
> - Starting with Oracle Database 11*g* Release 2 (11.2.0.2), the `capture_name` parameter is included in this procedure.

Syntax

```
DBMS_XSTREAM_ADM.CREATE_OUTBOUND(
   server_name            IN VARCHAR2,
   source_database        IN VARCHAR2  DEFAULT NULL,
   table_names            IN DBMS_UTILITY.UNCL_ARRAY,
   schema_names           IN DBMS_UTILITY.UNCL_ARRAY,
```

```
   capture_user           IN VARCHAR2  DEFAULT NULL,
   connect_user           IN VARCHAR2  DEFAULT NULL,
   comment                IN VARCHAR2  DEFAULT NULL,
   capture_name           IN VARCHAR2  DEFAULT NULL,
   include_dml            IN BOOLEAN   DEFAULT TRUE,
   include_ddl            IN BOOLEAN   DEFAULT FALSE,
   source_root_name       IN VARCHAR2  DEFAULT NULL,
   source_container_name  IN VARCHAR2  DEFAULT NULL,
   lcrid_version          IN NUMBER    DEFAULT NULL);

DBMS_XSTREAM_ADM.CREATE_OUTBOUND(
   server_name            IN VARCHAR2,
   source_database        IN VARCHAR2  DEFAULT NULL,
   table_names            IN VARCHAR2  DEFAULT NULL,
   schema_names           IN VARCHAR2  DEFAULT NULL,
   capture_user           IN VARCHAR2  DEFAULT NULL,
   connect_user           IN VARCHAR2  DEFAULT NULL,
   comment                IN VARCHAR2  DEFAULT NULL,
   capture_name           IN VARCHAR2  DEFAULT NULL,
   include_dml            IN BOOLEAN   DEFAULT TRUE,
   include_ddl            IN BOOLEAN   DEFAULT FALSE,
   source_root_name       IN VARCHAR2  DEFAULT NULL,
   source_container_name  IN VARCHAR2  DEFAULT NULL,
   lcrid_version          IN NUMBER    DEFAULT NULL);
```

Parameters

Table 244-16 CREATE_OUTBOUND Procedure Parameters

Parameter	Description
server_name	The name of the outbound server being created. A NULL specification is not allowed. Do not specify an owner. The specified name must not match the name of an existing outbound server, inbound server, apply process, or messaging client. **Note:** The server_name setting cannot exceed 30 bytes, and it cannot be altered after the outbound server is created.
source_database	The global name of the source database. The source database is where the changes to be captured originated. If non-NULL, then a condition is added to the outbound server's rules to filter the LCRs based on the global name of the source database. If NULL, then the procedure does not add a condition regarding the source database to the generated rules. In a CDB, specify the global name of the container to which the rules pertain. The container can be the root or a PDB. For example, mycdb.example.com or hrpdb.example.com. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. In a non-CDB, if non-NULL and the specified name is different from the global name of the current database, then downstream capture is assumed. In this case, configure the transmission of redo data from the source database to the downstream database before running the CREATE_OUTBOUND procedure. See *Oracle Database XStream Guide* for instructions. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically.

Table 244-16 (Cont.) CREATE_OUTBOUND Procedure Parameters

Parameter	Description
table_names	The tables for which DML and DDL changes are streamed out to the XStream client application. The tables can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.UNCL_ARRAY, where each element is the name of a table. Specify the first table in position 1. The last position must be NULL. Each table should be specified as [*schema_name.*]*table_name*. For example, hr.employees. If the schema is not specified, then the current user is the default. **See Also:** "Usage Notes" for more information about this parameter
schema_names	The schemas for which DML and DDL changes are streamed out to the XStream client application. The schemas can be specified in the following ways: • Comma-delimited list of type VARCHAR2. • A PL/SQL associative array of type DBMS_UTILITY.UNCL_ARRAY, where each element is the name of a schema. Specify the first schema in position 1. The last position must be NULL. **Note:** This procedure does not concatenate the schema_names parameter with the table_names parameter. To specify tables, enter fully qualified table names in the table_names parameter (*schema_name.table_name*). **See Also:** "Usage Notes" for more information about this parameter
capture_user	The user in whose security domain a capture process captures changes that satisfy its rule sets and runs custom rule-based transformations configured for capture process rules. If NULL, then the current user is the default. This procedure grants the capture user enqueue privilege on the queue used by the capture process and configures the user as a secure queue user. In addition, ensure that the capture user has the following privileges: • EXECUTE privilege on the rule sets used by the capture process • EXECUTE privilege on all rule-based transformation functions used in the positive rule set You can grant these privileges directly to the apply user, or you can grant them through roles. In addition, the capture user must be granted EXECUTE privilege on all packages, including Oracle supplied packages, that are invoked in rule-based transformations run by the capture process. These privileges must be granted directly to the capture user. They cannot be granted through roles. Only a user who is granted the DBA role can set a capture user. Only the SYS user can set the capture_user to SYS. A capture user does not require privileges on a database object to capture changes made to it. The capture process can pass these changes to a custom rule-based transformation function. Therefore, ensure that you consider security implications when you configure a capture process.

Table 244-16 (Cont.) CREATE_OUTBOUND Procedure Parameters

Parameter	Description
connect_user	The user who can attach to the specified outbound server to retrieve the change stream. The client application must attach to the outbound server as the specified connect user. If NULL, then the current user is the default. The connect user is the user in whose security domain an outbound server evaluates LCRs against its rule sets and runs custom rule-based transformations configured for outbound server rules. This user must have the necessary privileges to perform these actions. This procedure grants the connect user dequeue privileges on the queue used by the outbound server and configures the user as a secure queue user. In addition to the privileges granted by this procedure, grant the following privileges to the connect user: • EXECUTE privilege on the rule sets used by the outbound server • EXECUTE privilege on all rule-based transformation functions used in the rule set You can grant these privileges directly to the connect user, or you can grant them through roles. In addition, the connect user must be granted EXECUTE privilege on all packages, including Oracle supplied packages, that are invoked in subprograms run by the outbound server. These privileges must be granted directly to the apply user. They cannot be granted through roles.
comment	An optional comment associated with the outbound server.
capture_name	The name of the capture process configured to capture changes for the outbound server. Do not specify an owner. The capture process must not exist. If the specified name matches the name of an existing capture process, then an error is raised. If the name does not match the name of an existing capture process, then the procedure creates a new capture process with the specified name. If NULL, then the system creates a new capture process with a system-generated name. **Note:** The capture process name cannot be altered after the capture process is created.
include_dml	If TRUE, then the procedure creates a DML rule for DML changes. If FALSE, then the procedure does not create a DML rule. NULL is not permitted.
include_ddl	If TRUE, then the procedure creates a DDL rule for DDL changes. If FALSE, then the procedure does not create a DDL rule. NULL is not permitted.

Table 244-16 (Cont.) CREATE_OUTBOUND Procedure Parameters

Parameter	Description
source_root_name	The global name of the root in the source CDB. For example, mycdb.example.com. If this parameter is NULL, then the global name of the root in the local CDB is used. If you are configuring downstream capture, then this parameter must be a non-NULL value, and it must specify the global name of the root in the remote source CDB. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. **Note:** This parameter only applies to a CDB.
source_container_name	The short name of the source container. The container can be the root or a PDB. For example, CDB$ROOT or hrpdb. See *Oracle Database XStream Guide* for more information about setting this parameter in a CDB. **Note:** This parameter only applies to a CDB.
lcrid_version	The LCRID version for captured LCRs, either 1 or 2. If 2, then the LCRs are compatible with a database with its compatibility level at 12.2.0 or higher. If 1, then the LCRs are compatible with a database with its compatibility level at 12.1.0 or lower. If NULL, the default, and the database compatibility level is 12.2.0 or higher, then the lcrid_version is set to 2 internally. If the database compatibility level is 12.1.0 or lower, then the lcrid_version is set to 1 internally.

Usage Notes

The following list describes the behavior of the outbound server for various combinations of the table_names and schema_names parameters:

- If both the table_names and schema_names parameters are NULL or empty, then the outbound server streams all DML and DDL changes to the client application.

  This procedure is overloaded. The table_names and schema_names parameters are defaulted to NULL. Do not specify NULL for both table_names and schema_names in the same call; otherwise, error PLS-00307 is returned.
- If both the table_names and schema_names parameters are specified, then the outbound server streams DML and DDL changes for the specified tables and schemas.
- If the table_names parameter is specified and the schema_names parameter is NULL or empty, then the outbound server streams DML and DDL changes for the specified tables.
- If the table_names parameter is NULL or empty and the schema_names parameter is specified, then the outbound server streams DML and DDL changes for the specified schema.

For the procedure that uses the DBMS_UTILITY.UNCL_ARRAY type for the table_names and schema_names parameters, both parameters must be specified. To specify only tables, the

schema_names parameter must be specified and empty. To specify only schemas, the table_names parameter must be specified and empty.

> **Note:**
>
> An empty array includes one NULL entry.

DELETE_COLUMN Procedure

This procedure either adds or removes a declarative rule-based transformation which deletes a column from a row logical change record (LCR) that satisfies the specified rule.

For the transformation to be performed when the specified rule evaluates to TRUE, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

> **Note:**
>
> - The DELETE_COLUMN procedure supports the same data types supported by Oracle Replication capture processes.
> - The DELETE_COLUMN procedure is useful when you want to delete a relatively small number of columns in a row LCR. To delete most of the columns in a row LCR and keep a relatively small number of columns, consider using the KEEP_COLUMNS procedure in this package.
> - Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

> **See Also:**
>
> - *Oracle Database XStream Guide* for more information about declarative rule-based transformations and about the data types supported by capture processes
> - KEEP_COLUMNS Procedure

Syntax

```
DBMS_XSTREAM_ADM.DELETE_COLUMN(
   rule_name      IN  VARCHAR2,
   table_name     IN  VARCHAR2,
   column_name    IN  VARCHAR2,
   value_type     IN  VARCHAR2   DEFAULT '*',
   step_number    IN  NUMBER     DEFAULT 0,
   operation      IN  VARCHAR2   DEFAULT 'ADD');
```

Parameters

Table 244-17 DELETE_COLUMN Procedure Parameters

Parameter	Description
rule_name	The name of the rule, specified as [*schema_name.*]*rule_name*. If NULL, then the procedure raises an error. For example, to specify a rule in the hr schema named employees12, enter hr.employees12. If the schema is not specified, then the current user is the default.
table_name	The name of the table from which the column is deleted in the row LCR, specified as [*schema_name.*]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default.
column_name	The name of the column deleted from each row LCR that satisfies the rule.
value_type	Specify 'NEW' to delete the column from the new values in the row LCR. Specify 'OLD' to delete the column from the old values in the row LCR. Specify '*' to delete the column from both the old and new values in the row LCR.
step_number	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
operation	Specify 'ADD' to add the transformation to the rule. Specify 'REMOVE' to remove the transformation from the rule.

Usage Notes

When 'REMOVE' is specified for the operation parameter, all of the delete column declarative rule-based transformations for the specified rule are removed that match the specified table_name, column_name, and step_number parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the DELETE_COLUMN procedure when one or more of these parameters is NULL:

table_name	column_name	step_number	Result
NULL	NULL	NULL	Remove all delete column transformations for the specified rule.
NULL	NULL	non-NULL	Remove all delete column transformations with the specified step_number for the specified rule.
NULL	non-NULL	non-NULL	Remove all delete column transformations with the specified column_name and step_number for the specified rule.
non-NULL	NULL	non-NULL	Remove all delete column transformations with the specified table_name and step_number for the specified rule.
NULL	non-NULL	NULL	Remove all delete column transformations with the specified column_name for the specified rule.

table_name	column_name	step_number	Result
non-NULL	non-NULL	NULL	Remove all delete column transformations with the specified table_name and column_name for the specified rule.
non-NULL	NULL	NULL	Remove all delete column transformations with the specified table_name for the specified rule.
non-NULL	non-NULL	non-NULL	Remove all delete column transformations with the specified table_name, column_name, and step_number for the specified rule.

DROP_INBOUND Procedure

This procedure removes an inbound server configuration.

This procedure always removes the specified inbound server. This procedure also removes the queue for the inbound server if all of the following conditions are met:

- One call to the CREATE_INBOUND procedure created the queue.
- The inbound server is the only subscriber to the queue.

> **See Also:**
>
> "CREATE_INBOUND Procedure"

Syntax

```
DBMS_XSTREAM_ADM.DROP_INBOUND(
   server_name IN VARCHAR2);
```

Parameters

Table 244-18 DROP_INBOUND Procedure Parameters

Parameter	Description
server_name	The name of the inbound server being removed. Specify an existing inbound server. Do not specify an owner.

DROP_OUTBOUND Procedure

This procedure removes an outbound server configuration.

This procedure always drops the specified outbound server. This procedure also drops the queue used by the outbound server if both of the following conditions are met:

- The queue was created by the CREATE_OUTBOUND procedure in this package.
- The outbound server is the only subscriber to the queue.

If either one of the preceding conditions is not met, then the `DROP_OUTBOUND` procedure only drops the outbound server. It does not drop the queue.

This procedure also drops the capture process for the outbound server if both of the following conditions are met:

- The procedure can drop the outbound server's queue.
- The capture process was created by the `CREATE_OUTBOUND` procedure.

If the procedure can drop the queue but cannot manage the capture process, then it drops the queue without dropping the capture process.

> **See Also:**
>
> - "ADD_OUTBOUND Procedure"
> - "CREATE_OUTBOUND Procedure"

Syntax

```
DBMS_XSTREAM_ADM.DROP_OUTBOUND(
   server_name IN VARCHAR2);
```

Parameters

Table 244-19 DROP_OUTBOUND Procedure Parameters

Parameter	Description
`server_name`	The name of the outbound server being removed. Specify an existing outbound server. Do not specify an owner.

ENABLE_GG_XSTREAM_FOR_STREAMS Procedure

This procedure enables XStream optimizations and performance optimizations for Oracle Replication components.

This procedure is intended for users of Oracle Replication who want to enable XStream optimizations and optimizations. For example, you can enable the optimizations for an Oracle Replication configuration that uses capture processes and apply processes to replicate changes between Oracle databases.

These capabilities and optimizations are enabled automatically for XStream components, such as outbound servers, inbound servers, and capture processes that send changes to outbound servers. It is not necessary to run this procedure for XStream components.

When XStream optimizations are enabled, Oracle Replication components can stream ID key LCRs and sequence LCRs. The XStream performance optimizations improve efficiency in various areas, including:

- LCR processing
- Handling large transactions
- DML execution during apply
- Dependency computation and scheduling

- Capture process parallelism

Syntax

```
DBMS_XSTREAM_ADM.ENABLE_GG_XSTREAM_FOR_STREAMS(
   enable IN BOOLEAN  TRUE);
```

Parameters

Table 244-20 ENABLE_GG_XSTREAM_FOR_STREAMS Procedure Parameters

Parameter	Description
`enable`	If `TRUE`, then enable XStream performance optimizations for Oracle Replication components. If `FALSE`, then disable XStream performance optimizations for Oracle Replication components.

Usage Notes

The following usage notes apply to this procedure:

- When you run this procedure, all capture processes and apply processes are restarted.
- After you run this procedure, the `PURPOSE` column in the following views displays `XStream Streams`:
  - `ALL_APPLY`
  - `DBA_APPLY`
  - `ALL_CAPTURE`
  - `DBA_CAPTURE`
- A license for the Oracle GoldenGate product is required to enable XStream performance optimizations for Oracle Replication components.

> **See Also:**
>
> - IS_GG_XSTREAM_FOR_STREAMS Function
> - *Oracle Database XStream Guide, Chapter 1, Prerequisites for XStream"*

GET_MESSAGE_TRACKING Function

The `GET_MESSAGE_TRACKING` Function returns the tracking label for the current session.

> **See Also:**
>
> SET_MESSAGE_TRACKING Procedure

Syntax

```
DBMS_XSTREAM_ADM.GET_MESSAGE_TRACKING
RETURN VARCHAR2;
```

GET_TAG Function

This function gets the binary tag for all redo entries generated by the current session.

> **See Also:**
>
> "SET_TAG Procedure"

Syntax

```
DBMS_XSTREAM_ADM.GET_TAG
RETURN RAW;
```

Examples

The following example illustrates how to display the current logical change record (LCR) tag as output:

```
SET SERVEROUTPUT ON
DECLARE
   raw_tag RAW(2000);
BEGIN
   raw_tag := DBMS_XSTREAM_ADM.GET_TAG();
   DBMS_OUTPUT.PUT_LINE('Tag Value = ' || RAWTOHEX(raw_tag));
END;
/
```

You can also display the value by querying the `DUAL` view:

```
SELECT DBMS_XSTREAM_ADM.GET_TAG FROM DUAL;
```

IS_GG_XSTREAM_FOR_STREAMS Function

This function returns `TRUE` if XStream performance optimizations are enabled for Oracle Replication components, or this function returns `FALSE` if XStream performance optimizations are disabled for Oracle Replication components.

> **See Also:**
>
> "ENABLE_GG_XSTREAM_FOR_STREAMS Procedure"

Syntax

```
DBMS_XSTREAM_ADM.IS_GG_XSTREAM_FOR_STREAMS
RETURN BOOLEAN;
```

KEEP_COLUMNS Procedure

This procedure either adds or removes a declarative rule-based transformation which keeps a list of columns in a row logical change record (LCR) that satisfies the specified rule. The transformation deletes columns that are not in the list from the row LCR.

For the transformation to be performed when the specified rule evaluates to `TRUE`, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

This procedure is overloaded. The `column_list` parameter is type `VARCHAR2` and the `column_table` parameter is type `DBMS_UTILITY.LNAME_ARRAY`. These parameters enable you to enter the list of columns in different ways and are mutually exclusive.

Note:

- The `KEEP_COLUMNS` procedure supports the same data types supported by Oracle Replication capture processes.
- The `KEEP_COLUMNS` procedure is useful when you want to keep a relatively small number of columns in a row LCR. To keep most of the columns in a row LCR and delete a relatively small number of columns, consider using the `DELETE_COLUMN` procedure in this package.
- Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

See Also:

- *Oracle Database XStream Guide* for more information about declarative rule-based transformations and about the data types supported by Oracle Replication capture processes
- DELETE_COLUMN Procedure

Syntax

```
DBMS_XSTREAM_ADM.KEEP_COLUMNS(
   rule_name     IN  VARCHAR2,
   table_name    IN  VARCHAR2,
   column_list   IN  VARCHAR2,
   value_type    IN  VARCHAR2 DEFAULT '*',
   step_number   IN  NUMBER DEFAULT 0,
   operation     IN  VARCHAR2 DEFAULT 'ADD');

DBMS_XSTREAM_ADM.KEEP_COLUMNS(
   rule_name     IN  VARCHAR2,
   table_name    IN  VARCHAR2,
   column_table  IN  DBMS_UTILITY.LNAME_ARRAY,
   value_type    IN  VARCHAR2 DEFAULT '*',
```

```
  step_number   IN  NUMBER DEFAULT 0,
  operation     IN  VARCHAR2 DEFAULT 'ADD');
```

Parameters

Table 244-21 KEEP_COLUMNS Procedure Parameters

Parameter	Description
rule_name	The name of the rule, specified as [*schema_name.*]*rule_name*. If NULL, then the procedure raises an error. For example, to specify a rule in the hr schema named employees12, enter hr.employees12. If the schema is not specified, then the current user is the default.
table_name	The name of the table for which the columns are kept in the row LCR, specified as [*schema_name.*]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default.
column_list	The names of the columns kept for each row LCR that satisfies the rule. Specify a comma-delimited list of type VARCHAR2. The transformation removes columns that are not in the list from the row LCR. If this parameter is set to NULL, and the column_table parameter is also set to NULL, then the procedure raises an error.
column_table	The names of the columns kept for each row LCR that satisfies the rule. Specify a PL/SQL associative array of type DBMS_UTILITY.LNAME_ARRAY, where each element is the name of a column. The first schema should be in position 1. The last position must be NULL. The transformation removes columns that are not in the table from the row LCR. If this parameter is set to NULL, and the column_list parameter is also set to NULL, then the procedure raises an error.
value_type	Specify 'NEW' to keep the columns in the new values in the row LCR. Specify 'OLD' to keep the columns in the old values in the row LCR. Specify '*' to keep the columns in both the old and new values in the row LCR.
step_number	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
operation	Specify 'ADD' to add the transformation to the rule. Specify 'REMOVE' to remove the transformation from the rule.

Usage Notes

When 'REMOVE' is specified for the operation parameter, all of the keep columns declarative rule-based transformations for the specified rule are removed that match the specified table_name, column_list, column_table, and step_number parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the KEEP_COLUMNS procedure when one or more of these parameters is NULL:

table_name	column_list/ column_table	step_number	Result
NULL	NULL	NULL	Remove all keep columns transformations for the specified rule.

table_name	column_list/ column_table	step_number	Result
NULL	NULL	non-NULL	Remove all keep columns transformations with the specified step_number for the specified rule.
NULL	non-NULL	non-NULL	Remove all keep columns transformations with the specified column_list/ column_table and step_number for the specified rule.
non-NULL	NULL	non-NULL	Remove all keep columns transformations with the specified table_name and step_number for the specified rule.
NULL	non-NULL	NULL	Remove all keep columns transformations with the specified column_list/ column_table for the specified rule.
non-NULL	non-NULL	NULL	Remove all keep columns transformations with the specified table_name and column_list/column_table for the specified rule.
non-NULL	NULL	NULL	Remove all keep columns transformations with the specified table_name for the specified rule.
non-NULL	non-NULL	non-NULL	Remove all keep columns transformations with the specified table_name, column_list/column_table, and step_number for the specified rule.

MERGE_STREAMS Procedure

This procedure merges a stream that is flowing from one capture process with a stream that is flowing from another capture process.

Typically, this procedure is used to merge two streams that were split using the SPLIT_STREAMS procedure in this package. The SPLIT_STREAMS procedure clones components of the original stream when it splits the streams. Therefore, the information in this section uses the following terminology:

- The stream before it was split off has the original queue, original capture process, and original propagation.
- The stream that was split off by the SPLIT_STREAMS procedure has a cloned queue, cloned capture process, and cloned propagation.

This procedure is called by the MERGE_STREAMS_JOB procedure. The MERGE_STREAMS_JOB procedure determines whether the streams are within a user-specified merge threshold so that the streams can be merged safely. If the streams are not within the merge threshold, then the MERGE_STREAMS_JOB procedure does nothing. Typically, it is best to run the MERGE_STREAMS_JOB procedure instead of running the MERGE_STREAMS procedure directly.

However, you can choose to run the MERGE_STREAMS procedure directly when the following conditions are met:

- The problem at the destination of the split stream has been corrected, and the destination queue can accept changes.

- The cloned capture process used by the split stream is started and is capturing changes.
- The apply process at the destination database is applying the changes captured by the cloned capture process.
- The `CAPTURE_MESSAGE_CREATE_TIME` in the `GV$XSTREAM_CAPTURE` view of the cloned capture process has caught up to, or nearly caught up to, the `CAPTURE_MESSAGE_CREATE_TIME` of the original capture process. The cloned capture process might never completely catch up to the original capture process. Therefore, you can merge the split stream when the cloned capture process has nearly caught up to the original capture process.

The `MERGE_STREAMS` procedure performs the following actions:

1. Stops the cloned capture process.
2. Stops the original capture process.
3. Copies the cloned propagation back to the original propagation. The propagation has the same name as the original propagation after it is copied back.
4. Starts the original capture process from the lower SCN value of these two SCN values:
   - The acknowledged SCN of the cloned propagation.
   - The lowest acknowledged SCN of the other propagations that propagate changes captured by the original capture process.

   When the original capture process is started, it might recapture changes that it already captured, or it might capture changes that were already captured by the cloned capture process. In either case, the relevant apply processes will discard any duplicate changes they receive.
5. Drops the cloned propagation.
6. Drops the cloned capture process.
7. Drops the cloned queue.

> **See Also:**
>
> SPLIT_STREAMS Procedure

Syntax

```
DBMS_XSTREAM_ADM.MERGE_STREAMS(
   cloned_propagation_name  IN  VARCHAR2,
   propagation_name         IN  VARCHAR2  DEFAULT NULL,
   queue_name               IN  VARCHAR2  DEFAULT NULL,
   perform_actions          IN  BOOLEAN   DEFAULT TRUE,
   script_name              IN  VARCHAR2  DEFAULT NULL,
   script_directory_object  IN  VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-22 MERGE_STREAMS Procedure Parameters

Parameter	Description
cloned_propagation_name	The name of the cloned propagation used by the stream that was split off from the original stream using the SPLIT_STREAMS procedure. The name of the cloned propagation also identifies the cloned queue and capture process used by the cloned propagation. You must specify an existing propagation name. Do not specify an owner.
propagation_name	The name of the propagation that is merged back to the original stream. If NULL, then the name of the original propagation in the original stream is used. Specify NULL only if the streams were split using the SPLIT_STREAMS procedure. Specify a non-NULL value to use a name that is different from the original propagation name or if you are merging two streams that were not split by the SPLIT_STREAMS procedure. If a non-NULL value is specified, then an error is raised under either of the following conditions: • The queue specified in the queue_name parameter does not exist. • The queue specified in the queue_name parameter exists but is not used by a capture process.
queue_name	The name of the queue that is the source queue for the propagation that is merged back. If NULL, then the existing, original queue is the source queue for the propagation that is merged back. Specify NULL only if the streams were split using the SPLIT_STREAMS procedure. Specify a non-NULL value if you are merging two streams that were not split by the SPLIT_STREAMS procedure. Specify the name of the existing queue used by the capture process that will capture changes in the merged stream.
perform_actions	If TRUE, then the procedure performs the necessary actions to merge the streams directly. If FALSE, then the procedure does not perform the necessary actions to merge the streams directly. Specify FALSE when this procedure is generating a script that you can edit and then run. The procedure raises an error if you specify FALSE and either of the following parameters is NULL: • script_name • script_directory_object

Table 244-22 (Cont.) MERGE_STREAMS Procedure Parameters

Parameter	Description
script_name	If non-NULL and the perform_actions parameter is FALSE, then specify the name of the script generated by this procedure. The script contains all of the statements used to merge the streams. If a file with the specified script name exists in the specified directory for the script_directory_object parameter, then the procedure appends the statements to the existing file. If non-NULL and the perform_actions parameter is TRUE, then the procedure generates the specified script and performs the actions to split the stream directly. If NULL and the perform_actions parameter is TRUE, then the procedure performs the actions to merge the streams directly and does not generate a script. If NULL and the perform_actions parameter is FALSE, then the procedure raises an error.
script_directory_object	The directory object for the directory on the local computer system into which the generated script is placed. If the script_name parameter is NULL, then the procedure ignores this parameter and does not generate a script. If NULL and the script_name parameter is non-NULL, then the procedure raises an error. **Note:** The specified directory object cannot point to an Oracle Automatic Storage Management (ASM) disk group.

Usage Notes

You can use the MERGE_STREAMS procedure to merge two streams that were not split using the SPLIT_STREAMS procedure. Merging streams in this way can save resources and improve performance when a single database is running two or more capture processes.

MERGE_STREAMS_JOB Procedure

This procedure determines whether the original capture process and the cloned capture process are within the specified merge threshold. If they are within the merge threshold, then this procedure runs the MERGE_STREAMS procedure to merge the two streams.

Typically, this procedure is used to merge two streams that were split using the SPLIT_STREAMS procedure in this package. The SPLIT_STREAMS procedure clones components of the original stream when it splits the streams. Therefore, the information in this section uses the following terminology:

- The stream before it was split off has the original queue, original capture process, and original propagation.
- The stream that was split off by the SPLIT_STREAMS procedure has a cloned queue, cloned capture process, and cloned propagation.

If the auto_merge_threshold parameter was set to a positive number in the SPLIT_STREAMS procedure that split the streams, then a merge job runs the MERGE_STREAMS_JOB procedure automatically according to its schedule. The schedule name is specified for the schedule_name parameter, and the merge job name is specified for the merge_job_name parameter when the

MERGE_STREAMS_JOB procedure is run automatically. The merge job and its schedule were created by the SPLIT_STREAMS procedure.

If the auto_merge_threshold parameter was set to NULL or 0 (zero) in the SPLIT_STREAMS procedure that split the streams, then you can run the MERGE_STREAMS_JOB procedure manually. In this case, it is not run automatically.

See Also:

- MERGE_STREAMS Procedure
- SPLIT_STREAMS Procedure

Syntax

```
DBMS_XSTREAM_ADM.MERGE_STREAMS_JOB(
   cloned_propagation_name IN VARCHAR2,
   propagation_name        IN VARCHAR2 DEFAULT NULL,
   queue_name              IN VARCHAR2 DEFAULT NULL,
   merge_threshold         IN NUMBER,
   schedule_name           IN VARCHAR2 DEFAULT NULL,
   merge_job_name          IN VARCHAR2 DEFAULT NULL);
```

Parameters

Table 244-23 MERGE_STREAMS_JOB Procedure Parameters

Parameter	Description
cloned_propagation_name	The name of the cloned propagation used by the stream that was split off from the original stream using the SPLIT_STREAMS procedure. The name of the cloned propagation also identifies the cloned queue and capture process used by the cloned propagation. You must specify an existing propagation name. Do not specify an owner.
propagation_name	The name of the propagation that is merged back to the original stream. If NULL, then the name of the original propagation in the original stream is used. Specify NULL only if the streams were split using the SPLIT_STREAMS procedure. Specify a non-NULL value to use a name that is different from the original propagation name or if you are merging two streams that were not split by the SPLIT_STREAMS procedure. If a non-NULL value is specified, then an error is raised under either of the following conditions: • The queue specified in the queue_name parameter does not exist. • The queue specified in the queue_name parameter exists but is not used by a capture process.

Table 244-23 (Cont.) MERGE_STREAMS_JOB Procedure Parameters

Parameter	Description
queue_name	The name of the queue that is the source queue for the propagation that is merged back. If NULL, then the existing, original queue is the source queue for the propagation that is merged back. Specify NULL only if the streams were split using the SPLIT_STREAMS procedure. Specify a non-NULL value if you are merging two streams that were not split by the SPLIT_STREAMS procedure. Specify the name of the existing queue used by the capture process that will capture changes in the merged stream.
merge_threshold	The merge threshold in seconds. The value of the CAPTURE_MESSAGE_CREATE_TIME column for each capture process in the GV$XSTREAM_CAPTURE dynamic performance view determines whether the streams are merged. Specifically, if the difference, in seconds, between the CAPTURE_MESSAGE_CREATE_TIME of the cloned capture process and the original capture process is less than or equal to the value specified for this parameter, then this procedure runs the MERGE_STREAMS procedure to merge the streams. If the difference is greater than the value specified by this parameter, then this procedure does nothing.
schedule_name	The name of the schedule for the merge job. If NULL, then no schedule name is specified. Typically, you set this parameter to NULL when the auto_merge_threshold parameter was set to NULL or 0 (zero) in the SPLIT_STREAMS procedure that split the streams. Specify NULL if you run this procedure manually.
merge_job_name	The name of the job that merges the streams. If NULL, then no merge job name is specified. Typically, you set this parameter to NULL when the auto_merge_threshold parameter was set to NULL or 0 (zero) in the SPLIT_STREAMS procedure that split the streams. Specify NULL if you run this procedure manually.

Usage Notes

You can use the MERGE_STREAMS_JOB procedure to merge two streams that were not split using the SPLIT_STREAMS procedure. Merging streams in this way can save resources and improve performance when a single database is running two or more capture processes.

After the MERGE_STREAMS_JOB procedure completes, you can query the DBA_CAPTURE and DBA_PROPAGATION views to determine whether the streams were merged. If the streams were merged, then the cloned capture process and cloned propagation do not appear in these views.

If the streams were merged and the schedule_name and merge_job_name parameters were non-NULL, then the specified schedule and merge job are deleted automatically.

PURGE_SOURCE_CATALOG Procedure

This procedure removes all Oracle Replication data dictionary information at the local database for the specified object.

> **Note:**
>
> A multitenant container database is the only supported architecture in Oracle Database 21c and later releases. While the documentation is being revised, legacy terminology may persist. In most cases, "database" and "non-CDB" refer to a CDB or PDB, depending on context. In some contexts, such as upgrades, "non-CDB" refers to a non-CDB from a previous release.

You can use this procedure to remove Oracle Replication metadata that is not needed currently and will not be needed in the future.

Syntax

```
DBMS_XSTREAM_ADM.PURGE_SOURCE_CATALOG(
   source_database     IN  VARCHAR2,
   source_object_name  IN  VARCHAR2,
   source_object_type  IN  VARCHAR2,
   source_root_name    IN  VARCHAR2);
```

Parameters

Table 244-24 PURGE_SOURCE_CATALOG Procedure Parameters

Parameter	Description
source_database	In a non-CDB, specify the global name of the source database containing the database object. In a CDB, specify the global name of the container containing the database object. The container can be the root or a PDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is .NET, then the procedure specifies DBS1.NET automatically.
source_object_name	The name of the object specified as [*schema_name*.]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default.
source_object_type	Type of the object. Currently, TABLE is the only possible object type.
source_root_name	The global name of the source root containing the object in a CDB. The source root is where the changes being captured originated in a CDB. If you do not include the domain name, then the procedure appends it to the database name automatically. For example, if you specify DBS1 and the domain is EXAMPLE.COM, then the procedure specifies DBS1.EXAMPLE.COM automatically. If the source_root_name parameter is NULL, then the global name of the local root is the default. **Note:** This parameter only applies to a CDB.

Usage Notes

The global name of the source database containing the object must be specified for the `source_database` parameter. If the current database is not the source database for the object, then the procedure removes data dictionary information about the object from the current database, not the source database.

For example, suppose changes to the `hr.employees` table at the `dbs1.net` source database are being applied to the `hr.employees` table at the `dbs2.net` destination database. Also, suppose `hr.employees` at `dbs2.net` is not a source at all. In this case, specifying `dbs2.net` as the `source_database` for this table results in an error. However, specifying `dbs1.net` as the `source_database` for this table while running the `PURGE_SOURCE_CATALOG` procedure at the `dbs2.net` database removes data dictionary information about the table at `dbs2.net`.

Do not run this procedure at a database if either of the following conditions is true:

- Logical change records (LCRs) captured by the capture process for the object are or might be applied locally without reinstantiating the object.
- LCRs captured by the capture process for the object are or might be forwarded by the database without reinstantiating the object.

> **Note:**
>
> These conditions do not apply to LCRs that were not created by the capture process. That is, these conditions do not apply to user-created LCRs.

RECOVER_OPERATION Procedure

This procedure provides options for split and merge operations that stopped because they encountered an errors.

This procedure either rolls forward the operation, rolls back the operation, or purges all of the metadata about the operation. Split and merge operations might be run in an XStream Out environment in which multiple outbound servers use the same capture process.

This procedure only can perform these actions for split and merge operations using the `split_threshold` and `merge_threshold` capture process parameters set to non-`NULL` values to enable automatic split and merge.

Information about the operation is stored in the following data dictionary views when the operation is in process:

- `DBA_RECOVERABLE_SCRIPT`
- `DBA_RECOVERABLE_SCRIPT_PARAMS`
- `DBA_RECOVERABLE_SCRIPT_BLOCKS`
- `DBA_RECOVERABLE_SCRIPT_ERRORS`

The data dictionary views are populated at the database that contains the capture process.

When the operation completes successfully, metadata about the operation is moved from the `DBA_RECOVERABLE_SCRIPT` view to the `DBA_RECOVERABLE_SCRIPT_HIST` view. The other views, `DBA_RECOVERABLE_SCRIPT_PARAMS`, `DBA_RECOVERABLE_SCRIPT_BLOCKS`, and

DBA_RECOVERABLE_SCRIPT_ERRORS, retain information about the operation until it is purged automatically after 30 days.

When one of these operations encounters an error and stops, metadata about the operation remains in these views. In this case, you can either roll forward, roll back, or purge the metadata about the operation using the RECOVER_OPERATION procedure. If you choose to roll forward the operation, then correct conditions that caused the errors reported in DBA_RECOVERABLE_SCRIPT_ERRORS before proceeding.

Run the RECOVER_OPERATION procedure at the database that contains the capture process.

Note:

To run the RECOVER_OPERATION procedure, both databases must be Oracle Database 10*g* Release 2 or later databases.

See Also:

- "SPLIT_STREAMS Procedure"
- "MERGE_STREAMS Procedure"
- "MERGE_STREAMS_JOB Procedure"

Syntax

```
DBMS_XSTREAM_ADM.RECOVER_OPERATION(
   script_id       IN  RAW,
   operation_mode  IN  VARCHAR2  DEFAULT 'FORWARD');
```

Parameters

Table 244-25 RECOVER_OPERATION Procedure Parameters

Parameter	Description
script_id	The operation id of the operation that is being rolled forward, rolled back, or purged. Query the SCRIPT_ID column of the DBA_RECOVERABLE_SCRIPT data dictionary view to determine the operation id.

Table 244-25 (Cont.) RECOVER_OPERATION Procedure Parameters

Parameter	Description
operation_mode	If FORWARD, then the procedure rolls forward the operation. Specify FORWARD to try to complete the operation. If ROLLBACK, then the procedure rolls back all of the actions performed in the operation. If the rollback is successful, then this option also moves the metadata about the operation from the DBA_RECOVERABLE_SCRIPT view to the DBA_RECOVERABLE_SCRIPT_HIST view. The other views retain information about the operation for 30 days. If PURGE, then the procedure moves the metadata about the operation from the DBA_RECOVERABLE_SCRIPT view to the DBA_RECOVERABLE_SCRIPT_HIST view without rolling the operation back. The other views retain information about the operation for 30 days.

REMOVE_QUEUE Procedure

This procedure removes the specified ANYDATA queue.

Specifically, this procedure performs the following actions:

1. Waits until all current enqueue and dequeue transactions commit.
2. Stops the queue, which means that no further enqueues into the queue or dequeues from the queue are allowed.
3. Drops the queue.
4. If the drop_unused_queue_table parameter is set to TRUE, then drops the queue table if it is empty and no other queues are using it.
5. If the cascade parameter is set to TRUE, then drops all of the XStream clients that are using the queue.

> **Note:**
>
> The specified queue must be a ANYDATA queue.

Syntax

```
DBMS_XSTREAM_ADM.REMOVE_QUEUE(
   queue_name                IN  VARCHAR2,
   cascade                   IN  BOOLEAN  DEFAULT FALSE,
   drop_unused_queue_table   IN  BOOLEAN  DEFAULT TRUE);
```

Parameters

Table 244-26 REMOVE_QUEUE Procedure Parameters

Parameter	Description
queue_name	The name of the queue to remove, specified as [*schema_name*.]*queue_name*. For example, strmadmin.streams_queue. If the schema is not specified, then the current user is the default.
cascade	If TRUE, then the procedure drops any XStream clients that use the queue. If FALSE, then the procedure raises an error if there are any XStream clients that use the queue. Before you run this procedure with the cascade parameter set to FALSE, make sure no XStream clients are using the queue currently.
drop_unused_queue_table	If TRUE and the queue table for the queue is empty, then the procedure drops the queue table. The queue table is not dropped if it contains any messages or if it is used by another queue. If FALSE, then the procedure does not drop the queue table.

REMOVE_RULE Procedure

This procedure removes the specified rule or all rules from the rule set associated with the specified capture process, apply process, or propagation.

> **Note:**
>
> If a rule was automatically created by the system, and you want to drop the rule, then you should use this procedure to remove the rule instead of the DBMS_RULE_ADM.DROP_RULE procedure. If you use the DBMS_RULE_ADM.DROP_RULE procedure, then some metadata about the rule might remain.

Syntax

```
DBMS_XSTREAM_ADM.REMOVE_RULE(
   rule_name         IN  VARCHAR2,
   streams_type      IN  VARCHAR2,
   streams_name      IN  VARCHAR2,
   drop_unused_rule  IN  BOOLEAN  DEFAULT TRUE,
   inclusion_rule    IN  BOOLEAN  DEFAULT TRUE);
```

Parameters

Table 244-27 REMOVE_RULE Procedure Parameters

Parameter	Description
rule_name	The name of the rule to remove, specified as [*schema_name*.]*rule_name*. If NULL, then the procedure removes all rules from the specified capture process, apply process, or propagation rule set. For example, to specify a rule in the hr schema named prop_rule1, enter hr.prop_rule1. If the schema is not specified, then the current user is the default.
streams_type	The type of XStream client: • Specify capture for a capture process. • Specify propagation for a propagation. • Specify apply for an apply process.
streams_name	The name of the XStream client, which can be a capture process, propagation, or apply process. Do not specify an owner. If the specified XStream client does not exist, but there is metadata in the data dictionary that associates the rule with this client, then the procedure removes the metadata. If the specified XStream client does not exist, and there is no metadata in the data dictionary that associates the rule with this client, then the procedure raises an error.
drop_unused_rule	If TRUE and the rule is not in any rule set, then the procedure drops the rule from the database. If TRUE and the rule exists in any rule set, then the procedure does not drop the rule from the database. If FALSE, then the procedure does not drop the rule from the database.
inclusion_rule	If inclusion_rule is TRUE, then the procedure removes the rule from the positive rule set for the XStream client. If inclusion_rule is FALSE, then the procedure removes the rule from the negative rule set for the XStream client.

REMOVE_SUBSET_OUTBOUND_RULES Procedure

This procedure removes subset rules from an outbound server configuration.

The names of the specified insert, update, and delete rules must match those generated by the ADD_SUBSET_OUTBOUND_RULES procedure. To view the rule names for subset rules, run the following query:

```
SELECT RULE_OWNER, SUBSETTING_OPERATION, RULE_NAME
   FROM ALL_XSTREAM_RULES
   WHERE SUBSETTING_OPERATION IS NOT NULL;
```

Note:

- This procedure removes the declarative rule-based transformation associated with each rule it removes.
- This procedure does not remove rules from the outbound server's capture process.

See Also:

"ADD_SUBSET_OUTBOUND_RULES Procedure"

Syntax

```
DBMS_XSTREAM_ADM.REMOVE_SUBSET_OUTBOUND_RULES(
   server_name      IN VARCHAR2,
   insert_rule_name IN VARCHAR2,
   update_rule_name IN VARCHAR2,
   delete_rule_name IN VARCHAR2);
```

REMOVE_XSTREAM_CONFIGURATION Procedure

This procedure removes the XStream configuration at the local database.

Syntax

```
DBMS_XSTREAM_ADM.REMOVE_XSTREAM_CONFIGURATION(
   container IN VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-28 REMOVE_XSTREAM_CONFIGURATION Procedure Parameters

Parameter	Description
container	If CURRENT, then the XStream configuration is removed from the current container. CURRENT can be specified while connected to the root or to a PDB in a CDB. If ALL, then the XStream configuration is removed from all of the containers in the CDB. To specify ALL, the procedure must be invoked in the root. If a container name, then the XStream configuration is removed from the specified container. To specify root, use CDB$ROOT while connected to the root. To specify a PDB, the procedure must be invoked in the root. **Note:** This parameter only applies to a CDB.

Usage Notes

Specifically, this procedure performs the following actions at the local database:

- Drops all capture processes
- If any tables have been prepared for instantiation, then aborts preparation for instantiation for the table using the ABORT_TABLE_INSTANTIATION procedure in the DBMS_CAPTURE_ADM package

- If any schemas have been prepared for instantiation, then aborts preparation for instantiation for the schema using the ABORT_SCHEMA_INSTANTIATION procedure in the DBMS_CAPTURE_ADM package
- If the database has been prepared for instantiation, then aborts preparation for instantiation for the database using the ABORT_GLOBAL_INSTANTIATION procedure in the DBMS_CAPTURE_ADM package
- Drops propagations that were created using either the DBMS_XSTREAM_ADM package or the DBMS_PROPAGATION_ADM package. Before a propagation is dropped, its propagation job is disabled. Does not drop propagations that were created using the DBMS_AQADM package.
- Disables all propagation jobs used by propagations
- Drops all apply processes. If there are apply errors in the error queue for an apply process, then this procedure deletes these apply errors before it drops the apply process.
- Removes specifications for DDL handlers used by apply processes, but does not delete the PL/SQL procedures used by these handlers
- Removes specifications for message handlers used by apply processes, but does not delete the PL/SQL procedures used by these handlers
- Removes specifications for precommit handlers used by apply processes, but does not delete the PL/SQL procedures used by these handlers
- Removes the instantiation SCN and ignore SCN for each apply object and schema and for the entire database
- Removes messaging clients
- Unsets message notification specifications that were set using the SET_MESSAGE_NOTIFICATION procedure in the DBMS_XSTREAM_ADM package
- Removes specifications for procedure DML handlers and error handlers, but does not delete the PL/SQL procedures used by these handlers
- Removes update conflict handlers
- Removes specifications for substitute key columns for apply tables
- Drops rule sets and rules that were created using the DBMS_XSTREAM_ADM package.
- Drops unused rule sets that were used by capture processes, propagations, apply processes, and messaging clients, and removes the rules in these rule sets. These rules and rule sets are removed regardless of whether they were created using the DBMS_XSTREAM_ADM package or the DBMS_RULE_ADM package.

This procedure stops capture processes and apply processes before it drops them.

This procedure does not drop rule sets or rules if they meet both of the following conditions:

- The rule sets or rules were created using the DBMS_RULE_ADM package.
- The rule sets or rules were not used by a capture process, propagation, apply process, or messaging client.

> **Note:**
>
> Running this procedure is dangerous. You should run this procedure only if you are sure you want to remove the entire XStream configuration at a database. If an Oracle Replication configuration exists at the database, then this procedure also removes the entire Oracle Replication configuration.

> **Note:**
>
> - Running this procedure repeatedly does not cause errors. If the procedure fails to complete, then you can run it again.
> - This procedure commits multiple times.

RENAME_COLUMN Procedure

This procedure either adds or removes a declarative rule-based transformation which renames a column in a row logical change record (LCR) that satisfies the specified rule.

For the transformation to be performed when the specified rule evaluates to `TRUE`, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

> **Note:**
>
> - The `RENAME_COLUMN` procedure supports the same data types supported by Oracle Replication capture processes.
> - Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

> **See Also:**
>
> *Oracle Database XStream Guide* for more information about declarative rule-based transformations and about the data types supported by Oracle Replication capture processes

Syntax

```
DBMS_XSTREAM_ADM.RENAME_COLUMN(
   rule_name         IN  VARCHAR2,
   table_name        IN  VARCHAR2,
   from_column_name  IN  VARCHAR2,
   to_column_name    IN  VARCHAR2,
   value_type        IN  VARCHAR2  DEFAULT '*',
```

```
  step_number       IN  NUMBER    DEFAULT 0,
  operation         IN  VARCHAR2  DEFAULT 'ADD');
```

Parameters

Table 244-29 RENAME_COLUMN Procedure Parameters

Parameter	Description
rule_name	The name of the rule, specified as [*schema_name.*]*rule_name*. If NULL, then the procedure raises an error. For example, to specify a rule in the hr schema named employees12, enter hr.employees12. If the schema is not specified, then the current user is the default.
table_name	The name of the table in which the column is renamed in the row LCR, specified as [*schema_name.*]*object_name*. For example, hr.employees. If the schema is not specified, then the current user is the default.
from_column_name	The name of the column to be renamed in each row LCR that satisfies the rule.
to_column_name	The new name of the column in each row LCR that satisfies the rule.
value_type	Specify 'NEW' to rename the column in the new values in the row LCR. Specify 'OLD' to rename the column in the old values in the row LCR. Specify '*' to rename the column in both the old and new values in the row LCR.
step_number	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
operation	Specify 'ADD' to add the transformation to the rule. Specify 'REMOVE' to remove the transformation from the rule.

Usage Notes

When 'REMOVE' is specified for the operation parameter, all of the rename column declarative rule-based transformations for the specified rule are removed that match the specified table_name, column_name, and step_number parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the RENAME_COLUMN procedure when one or more of these parameters is NULL:

table_name	from_column_name	to_column_name	step_number	Result
NULL	NULL	NULL	NULL	Remove all rename column transformations for the specified rule.
NULL	NULL	NULL	non-NULL	Remove all rename column transformations with the specified step_number for the specified rule.
NULL	NULL	non-NULL	non-NULL	Remove all rename column transformations with the specified to_column_name and step_number for the specified rule.

table_name	from_column_name	to_column_name	step_number	Result
NULL	non-NULL	non-NULL	non-NULL	Remove all rename column transformations with the specified table_name and step_number for the specified rule.
NULL	NULL	non-NULL	NULL	Remove all rename column transformations with the specified column_name for the specified rule.
non-NULL	NULL	non-NULL	NULL	Remove all rename column transformations with the specified table_name and column_name for the specified rule.
NULL	non-NULL	NULL	NULL	Remove all rename column transformations with the specified table_name for the specified rule.
NULL	non-NULL	non-NULL	NULL	Remove all rename column transformations with the specified table_name, column_name, and step_number for the specified rule.
non-NULL	NULL	non-NULL	NULL	Remove all rename column transformations with the specified table_name, column_name, and step_number for the specified rule.
non-NULL	non-NULL	non-NULL	NULL	Remove all rename column transformations with the specified table_name, column_name, and step_number for the specified rule.
non-NULL	non-NULL	non-NULL	non-NULL	Remove all rename column transformations with the specified table_name, column_name, and step_number for the specified rule.

RENAME_SCHEMA Procedure

This procedure either adds or removes a declarative rule-based transformation which renames a schema in a row logical change record (LCR) that satisfies the specified rule.

For the transformation to be performed when the specified rule evaluates to TRUE, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

> **Note:**
>
> Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

> **See Also:**
>
> *Oracle Database XStream Guide* for more information about declarative rule-based transformations

Syntax

```
DBMS_XSTREAM_ADM.RENAME_SCHEMA(
   rule_name          IN  VARCHAR2,
   from_schema_name   IN  VARCHAR2,
   to_schema_name     IN  VARCHAR2,
   step_number        IN  NUMBER    DEFAULT 0,
   operation          IN  VARCHAR2  DEFAULT 'ADD');
```

Parameters

Table 244-30 RENAME_SCHEMA Procedure Parameters

Parameter	Description
`rule_name`	The name of the rule, specified as `[schema_name.]rule_name`. If `NULL`, then the procedure raises an error. For example, to specify a rule in the `hr` schema named `employees12`, enter `hr.employees12`. If the schema is not specified, then the current user is the default.
`from_schema_name`	The name of the schema to be renamed in each row LCR that satisfies the rule.
`to_schema_name`	The new name of the schema in each row LCR that satisfies the rule.
`step_number`	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
`operation`	Specify `'ADD'` to add the transformation to the rule. Specify `'REMOVE'` to remove the transformation from the rule.

Usage Notes

When `'REMOVE'` is specified for the `operation` parameter, all of the rename schema declarative rule-based transformations for the specified rule are removed that match the specified `from_schema_name`, `to_schema_name`, and `step_number` parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the `RENAME_SCHEMA` procedure when one or more of these parameters is `NULL`:

from_schema_name	to_schema_name	step_number	Result
`NULL`	`NULL`	`NULL`	Remove all rename schema transformations for the specified rule.
`NULL`	`NULL`	non-`NULL`	Remove all rename schema transformations with the specified `step_number` for the specified rule.

from_schema_name	to_schema_name	step_number	Result
NULL	non-NULL	non-NULL	Remove all rename schema transformations with the specified to_schema_name and step_number for the specified rule.
non-NULL	NULL	non-NULL	Remove all rename schema transformations with the specified from_schema_name and step_number for the specified rule.
NULL	non-NULL	NULL	Remove all rename schema transformations with the specified to_schema_name for the specified rule.
non-NULL	non-NULL	NULL	Remove all rename schema transformations with the specified from_schema_name and to_schema_name for the specified rule.
non-NULL	NULL	NULL	Remove all rename schema transformations with the specified from_schema_name for the specified rule.
non-NULL	non-NULL	non-NULL	Remove all rename schema transformations with the specified from_schema_name, to_schema_name, and step_number for the specified rule.

RENAME_TABLE Procedure

This procedure either adds or removes a declarative rule-based transformation which renames a table in a row logical change record (row LCR) that satisfies the specified rule.

For the transformation to be performed when the specified rule evaluates to TRUE, the rule must be in the positive rule set of an XStream client. XStream clients include capture processes, propagations, and apply processes.

> **Note:**
>
> Declarative transformations can transform row LCRs only. Therefore, a DML rule must be specified when you run this procedure. If a DDL rule is specified, then the procedure raises an error.

> **See Also:**
>
> *Oracle Database XStream Guide* for more information about declarative rule-based transformations

Syntax

```
DBMS_XSTREAM_ADM.RENAME_TABLE(
   rule_name         IN  VARCHAR2,
   from_table_name   IN  VARCHAR2,
   to_table_name     IN  VARCHAR2,
   step_number       IN  NUMBER    DEFAULT 0,
   operation         IN  VARCHAR2  DEFAULT 'ADD');
```

Parameters

Table 244-31 RENAME_TABLE Procedure Parameters

Parameter	Description
`rule_name`	The name of the rule, specified as `[schema_name.]rule_name`. If `NULL`, then the procedure raises an error. For example, to specify a rule in the `hr` schema named `employees12`, enter `hr.employees12`. If the schema is not specified, then the current user is the default.
`from_table_name`	The name of the table to be renamed in each row LCR that satisfies the rule, specified as `[schema_name.]object_name`. For example, `hr.employees`. If the schema is not specified, then the current user is the default.
`to_table_name`	The new name of the table in each row LCR that satisfies the rule, specified as `[schema_name.]object_name`. For example, `humres.staff`. The transformation can rename the table only, the schema only, or the table and the schema. If the schema is not specified, then the current user is the default.
`step_number`	The order of execution of the transformation. **See Also:** *Oracle Database XStream Guide* for more information about transformation ordering
`operation`	Specify `'ADD'` to add the transformation to the rule. Specify `'REMOVE'` to remove the transformation from the rule.

Usage Notes

When `'REMOVE'` is specified for the `operation` parameter, all of the rename table declarative rule-based transformations for the specified rule are removed that match the specified `from_table_name`, `to_table_name`, and `step_number` parameters. Nulls specified for these parameters act as wildcards. The following table lists the behavior of the `RENAME_TABLE` procedure when one or more of these parameters is `NULL`:

from_table_name	to_table_name	step_number	Result
`NULL`	`NULL`	`NULL`	Remove all rename table transformations for the specified rule.

from_table_name	to_table_name	step_number	Result
NULL	NULL	non-NULL	Remove all rename table transformations with the specified step_number for the specified rule.
NULL	non-NULL	non-NULL	Remove all rename table transformations with the specified to_table_name and step_number for the specified rule.
non-NULL	NULL	non-NULL	Remove all rename table transformations with the specified from_table_name and step_number for the specified rule.
NULL	non-NULL	NULL	Remove all rename table transformations with the specified to_table_name for the specified rule.
non-NULL	non-NULL	NULL	Remove all rename table transformations with the specified from_table_name and to_table_name for the specified rule.
non-NULL	NULL	NULL	Remove all rename table transformations with the specified from_table_name for the specified rule.
non-NULL	non-NULL	non-NULL	Remove all rename table transformations with the specified from_table_name, to_table_name, and step_number for the specified rule.

SET_MESSAGE_TRACKING Procedure

This procedure sets the tracking label for logical change records (LCRs) produced by the current session.

This procedure affects only the current session. Any LCRs produced by the current session are tracked, including captured LCRs and persistent LCRs.

> **Note:**
>
> The tracking label set by this procedure does not track non-LCR messages.

> **See Also:**
>
> KAWGET_MESSAGE_TRACKING

Syntax

```
DBMS_XSTREAM_ADM.SET_MESSAGE_TRACKING(
   tracking_label IN VARCHAR2  DEFAULT 'Streams_tracking',
   actions        IN NUMBER    DEFAULT DBMS_XSTREAM_ADM.ACTION_MEMORY);
```

Parameters

Table 244-32 SET_MESSAGE_TRACKING Procedure Parameters

Parameter	Description
`tracking_label`	The label used to track the LCRs produced by the session. Set this parameter to `NULL` to stop message tracking in the current session. The size limit for a label is 4,000 bytes.
`actions`	When `DBMS_XSTREAM_ADM.ACTION_MEMORY` is specified, the LCRs are tracked in memory. Currently, `DBMS_XSTREAM_ADM.ACTION_MEMORY` is the only valid setting for this parameter. The value specified for this parameter is an enumerated constant. Enumerated constants must be prefixed with the package name.

SET_PARAMETER Procedure

This procedure sets a parameter for an outbound server, an inbound server, or an outbound server's capture process.

Syntax

```
DBMS_XSTREAM_ADM.SET_PARAMETER(
   streams_name    IN  VARCHAR2,
   streams_type    IN  VARCHAR2,
   parameter       IN  VARCHAR2,
   value           IN  VARCHAR2  DEFAULT NULL,
   no_wait         IN  BOOLEAN   DEFAULT FALSE,
   source_database IN  VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-33 SET_PARAMETER Procedure Parameters

Parameter	Description
`streams_type`	The type of XStream client: • Specify `capture` for a capture process. • Specify `apply` for an outbound server or inbound server.
`streams_name`	The name of the capture process, outbound server, or inbound server. Do not specify an owner.

Table 244-33 (Cont.) SET_PARAMETER Procedure Parameters

Parameter	Description
parameter	The name of the parameter you are setting. See "Capture Process Parameters" for information about capture process parameters. See "Apply Component Parameters" for information about outbound server and inbound server parameters.
value	The value to which the parameter is set. If NULL, then the parameter is set to its default value.
no_wait	If TRUE, then the parameter is set immediately. If FALSE, then the parameter is set after synchronizing with the running capture process, inbound server, or outbound server. When you modify multiple parameters for the same process consecutively, setting this parameter to TRUE speeds up each call. However, if the process is currently running, you must set this parameter to FALSE in the last to the procedure to ensure that the process uses the modified parameter values. If the no_wait parameter is set to TRUE for the last call to the procedure, the running process might not detect the parameter changes.
source_database	If CURRENT, then the parameter is set only in the container where the procedure is invoked. CURRENT can be specified while connected to the root or to a PDB. If ALL, then the parameter is set in all containers in the CDB and all PDBs created after the procedure is invoked. To specify ALL, the procedure must be invoked in the root. If a container name, then the parameter is set in the specified container. To specify root, use CDB$ROOT while connected to the root. To specify a PDB, the procedure must be invoked in the root. **Note:** This parameter only applies to CDBs. Also, a non-null value can be specified only for the following parameters: • include_objects capture parameter • excludetag capture or apply parameter • excludetrans capture or apply parameter • excludeuser capture or apply parameter • excludeuserid capture or apply parameter • getreplicates capture or apply parameter • getapplops capture or apply parameter

SET_TAG Procedure

This procedure sets the binary tag for all redo entries subsequently generated by the current session.

Each redo entry generated by DML or DDL statements in the current session will have this tag. This procedure affects only the current session.

Syntax

```
DBMS_XSTREAM_ADM.SET_TAG(
   tag  IN RAW  DEFAULT NULL);
```

Parameters

Table 244-34 SET_TAG Procedure Parameters

Parameter	Description
tag	The binary tag for all subsequent redo entries generated by the current session. A raw value is a sequence of bytes, and a byte is a sequence of bits. By default, the tag for a session is NULL. The size limit for a tag value is 2000 bytes.

Usage Notes

To set the tag to the hexadecimal value of '17' in the current session, run the following procedure:

```
EXEC DBMS_XSTREAM_ADM.SET_TAG(tag => HEXTORAW('17'));
```

The following are considerations for the SET_TAG procedure:

- This procedure is not transactional. That is, the effects of SET_TAG cannot be rolled back.
- If the SET_TAG procedure is run to set a non-NULL session tag before a data dictionary build has been performed on the database, then the redo entries for a transaction that started before the dictionary build might not include the specified tag value for the session. Therefore, perform a data dictionary build before using the SET_TAG procedure in a session. A data dictionary build happens when the DBMS_CAPTURE_ADM.BUILD procedure is run. The BUILD procedure can be run automatically when a capture process is created.

> **See Also:**
>
> BUILD Procedure

SET_UP_QUEUE Procedure

This procedure creates a queue table and a ANYDATA queue.

Syntax

```
DBMS_XSTREAM_ADM.SET_UP_QUEUE(
   queue_table     IN  VARCHAR2  DEFAULT 'streams_queue_table',
   storage_clause  IN  VARCHAR2  DEFAULT NULL,
   queue_name      IN  VARCHAR2  DEFAULT 'streams_queue',
   queue_user      IN  VARCHAR2  DEFAULT NULL,
   comment         IN  VARCHAR2  DEFAULT NULL);
```

Parameters

Table 244-35 SET_UP_QUEUE Procedure Parameters

Parameter	Description
queue_table	The name of the queue table specified as [*schema_name*.]*queue_table_name*. For example, strmadmin.streams_queue_table. If the schema is not specified, then the current user is the default. If the queue table owner is not specified, then the procedure specifies the user who runs this procedure automatically as the queue table owner. Queue table names can be a maximum of 24 bytes.
storage_clause	The storage clause for queue table The storage parameter is included in the CREATE TABLE statement when the queue table is created. You can specify any valid table storage clause. If a tablespace is not specified here, then the procedure creates the queue table and all its related objects in the default user tablespace of the user who runs this procedure. If a tablespace is specified here, then the procedure creates the queue table and all its related objects in the tablespace specified in the storage clause. If NULL, then the procedure uses the storage characteristics of the tablespace in which the queue table is created. **See Also:** *Oracle Database SQL Language Reference*for more information about storage clauses
queue_name	The name of the queue that will function as the ANYDATA queue, specified as [*schema_name*.]*queue_name*. For example, strmadmin.streams_queue. If the schema is not specified, then the procedure uses the queue table owner. The owner of the queue table must also be the owner of the queue. The queue owner automatically has privileges to perform all queue operations on the queue. If the schema is not specified for this parameter, and the queue table owner is not specified in queue_table, then the current user is the default. Queue names can be a maximum of 24 bytes.
queue_user	The name of the user who requires ENQUEUE and DEQUEUE privileges for the queue. This user also is configured as a secure queue user of the queue. The queue user cannot grant these privileges to other users because they are not granted with the GRANT option. If NULL, then the procedure does not grant any privileges. You can also grant queue privileges to the appropriate users using the DBMS_AQADM package.
comment	The comment for the queue

Usage Notes

Set up includes the following actions:

- If the specified queue table does not exist, then this procedure runs the CREATE_QUEUE_TABLE procedure in the DBMS_AQADM package to create the queue table with the specified storage clause. If this procedure creates the queue table, then it creates a multiple consumer ANYDATA queue that is both a secure queue and a transactional queue.

Also, if the database is Oracle Database 10*g* release 2 or later, the `sort_list` setting in `CREATE_QUEUE_TABLE` is set to `commit_time`. If the database is a release before Oracle Database 10*g* release 2, the `sort_list` setting in `CREATE_QUEUE_TABLE` is set to `enq_time`.

- If the specified queue table exists, then the queue uses the properties of the existing queue table.
- If the specified queue name does not exist, then this procedure runs the `CREATE_QUEUE` procedure in the `DBMS_AQADM` package to create the queue.
- This procedure starts the queue.
- If a queue user is specified, then this procedure configures this user as a secure queue user of the queue and grants `ENQUEUE` and `DEQUEUE` privileges on the queue to the specified queue user.

  To configure the queue user as a secure queue user, this procedure creates an Advanced Queuing agent with the same name as the user name, if one does not exist. If an agent with this name exists and is associated with the queue user only, then it is used. `SET_UP_QUEUE` then runs the `ENABLE_DB_ACCESS` procedure in the `DBMS_AQADM` package, specifying the agent and the user.

> **Note:**
>
> If the agent that `SET_UP_QUEUE` tries to create exists and is associated with a user other than the user specified by `queue_user`, then the procedure raises an error. In this case, rename or remove the existing agent, and retry `SET_UP_QUEUE`.

SPLIT_STREAMS Procedure

This procedure splits one stream flowing from a capture process off from all of the other streams flowing from the capture process.

This procedure is intended for an Oracle Replication environment in which a capture process captures changes that are propagated to two or more destination databases. When one destination of a propagation stops accepting the captured changes, the changes remain in the capture process's queue. The queue can grow and begin to spill LCRs to the hard disk, degrading the performance of the Oracle Replication environment. A destination might stop accepting changes for several reasons. For example, the destination database might be down.

Specifically, this procedure performs the following actions:

1. Creates a new queue at the database running the capture process. The new queue is called the cloned queue because it is a clone of the queue used by the original stream. The new queue will be used by the new, cloned capture process, and it will be the source queue for the new, cloned propagation.
2. Creates a new propagation that propagates LCRs from the source queue created in Step 1 to the existing destination queue. The new propagation is called the cloned propagation because it is a clone of the propagation used by the original stream. The cloned propagation uses the same rule set as the original propagation.
3. Stops the capture process.
4. Queries the acknowledge SCN for the original propagation. The acknowledged SCN is the last SCN acknowledged by the apply process that applies the changes sent by the propagation.

5. Creates a new capture process. The new capture process is called the cloned capture process because it is a clone of the capture process used by the original stream. The procedure sets the start SCN for the cloned capture process to the value of the queried acknowledged SCN. The cloned capture process uses the same rule set as the original capture process.
6. Drops the original propagation.
7. Starts the original capture process with the start SCN set to the acknowledged SCN queried in Step 4.
8. If the `auto_merge_threshold` parameter is set to a positive number, then creates an Oracle Scheduler job to run the `MERGE_STREAMS_JOB` procedure at set intervals according to its schedule. When the two streams are within the specified merge threshold, the `MERGE_STREAMS_JOB` procedure runs the `MERGE_STREAMS` procedure to merge the streams automatically.

After the `SPLIT_STREAMS` procedure has finished running, the cloned capture process is disabled. When the problem at the destination database is solved, and the destination queue can accept changes, you should start the cloned capture process using the `START_CAPTURE` procedure in the `DBMS_CAPTURE_ADM` package.

Note:

If the original capture process is a downstream capture process, then you must configure the cloned capture process to read the redo log from the source database before you start the cloned capture process.

See Also:

- "MERGE_STREAMS Procedure"
- "MERGE_STREAMS_JOB Procedure"

Syntax

```
DBMS_XSTREAM_ADM.SPLIT_STREAMS(
   propagation_name          IN      VARCHAR2,
   cloned_propagation_name   IN      VARCHAR2  DEFAULT NULL,
   cloned_queue_name         IN      VARCHAR2  DEFAULT NULL,
   cloned_capture_name       IN      VARCHAR2  DEFAULT NULL,
   perform_actions           IN      BOOLEAN   DEFAULT TRUE,
   script_name               IN      VARCHAR2  DEFAULT NULL,
   script_directory_object   IN      VARCHAR2  DEFAULT NULL,
   auto_merge_threshold      IN      NUMBER    DEFAULT NULL,
   schedule_name             IN OUT  VARCHAR2,
   merge_job_name            IN OUT  VARCHAR2);
```

Parameters

Table 244-36 SPLIT_STREAMS Procedure Parameters

Parameter	Description
propagation_name	The name of the propagation that cannot send LCRs to its destination queue. The specified propagation is the propagation for the stream that is being split off from the other streams. You must specify an existing propagation name. Do not specify an owner.
cloned_propagation_name	The name of the new propagation created by this procedure for the stream that is split off. If NULL, then the system generates a propagation name.
cloned_queue_name	The name of the new queue created by this procedure for the stream that is split off. If NULL, then the system generates a queue name.
cloned_capture_name	The name of the new capture process created by this procedure for the stream that is split off. If NULL, then the system generates a capture process name.
perform_actions	If TRUE, then the procedure performs the necessary actions to split the stream directly. If FALSE, then the procedure does not perform the necessary actions to split the stream directly. Specify FALSE when this procedure is generating a script that you can edit and then run. The procedure raises an error if you specify FALSE and either of the following parameters is NULL: • script_name • script_directory_object
script_name	If non-NULL and the perform_actions parameter is FALSE, then specify the name of the script generated by this procedure. The script contains all of the statements used to split the stream. If a file with the specified script name exists in the specified directory for the script_directory_object parameter, then the procedure appends the statements to the existing file. If non-NULL and the perform_actions parameter is TRUE, then the procedure generates the specified script and performs the actions to split the stream directly. If NULL and the perform_actions parameter is TRUE, then the procedure performs the actions to split the stream directly and does not generate a script. If NULL and the perform_actions parameter is FALSE, then the procedure raises an error.
script_directory_object	The directory object for the directory on the local computer system into which the generated script is placed. If the script_name parameter is NULL, then the procedure ignores this parameter and does not generate a script. If NULL and the script_name parameter is non-NULL, then the procedure raises an error. **Note:** The specified directory object cannot point to an Oracle Automatic Storage Management (ASM) disk group.

Table 244-36 (Cont.) SPLIT_STREAMS Procedure Parameters

Parameter	Description
auto_merge_threshold	If a positive number is specified, then the stream that was split off is automatically merged back into all of the other streams flowing from the capture process by an Oracle Scheduler job. The job runs the MERGE_STREAMS_JOB procedure at set intervals according to its schedule. The value of the CAPTURE_MESSAGE_CREATE_TIME column for each capture process in the GV$XSTREAM_CAPTURE dynamic performance view determines when the streams are merged. Specifically, if the difference, in seconds, between CAPTURE_MESSAGE_CREATE_TIME of the cloned capture process and the original capture process is less than or equal to the value specified for the auto_merge_threshold parameter, then the two streams are merged automatically. The cloned capture process must be started before the split stream can be merged back with the original stream. If NULL or 0 (zero) is specified, then the split stream is not merged back with the original stream automatically. To merge the split stream with the original stream, run the MERGE_STREAM procedure manually when the CAPTURE_MESSAGE_CREATE_TIME of the cloned capture process catches up to, or nearly catches up to, the CAPTURE_MESSAGE_CREATE_TIME of the original capture process. The CAPTURE_MESSAGE_CREATE_TIME records the time when a captured change was recorded in the redo log.
schedule_name	The Oracle Scheduler schedule name, specified as [*schema_name.*]*schedule_name*. For example, strmadmin.merge_schedule. If the schema is not specified, then the current user is the default. If auto_merge_threshold is a non-NULL positive number, then the schedule is used by the job that will automatically merge the streams at the appropriate time. You can specify a schedule name to adhere to naming conventions or to track the schedule more easily. If NULL and auto_merge_threshold is a non-NULL positive number, then the system generates a schedule name. If auto_merge_threshold is NULL or 0 (zero), then this parameter must be NULL. If this procedure creates a schedule, the schedule starts when the procedure completes. You can modify the schedule to control how often the merge job is run. If an existing schedule name is specified, an error is raised.

Table 244-36 (Cont.) SPLIT_STREAMS Procedure Parameters

Parameter	Description
merge_job_name	The Oracle Scheduler job name, specified as [*schema_name*.]*merge_job_name*. For example, strmadmin.merge_job. If the schema is not specified, then the current user is the default. If auto_merge_threshold is a non-NULL positive number, then the job will automatically merge the streams at the appropriate time. Specify a merge job name to adhere to naming conventions or to track the job more easily. If NULL and auto_merge_threshold is a non-NULL positive number, then the system generates a job name. If auto_merge_threshold is NULL or 0 (zero), then this parameter must be NULL. If an existing job name is specified, an error is raised.

See Also:

Oracle Database Administrator's Guide for information about Oracle Scheduler

START_OUTBOUND Procedure

This procedure starts an XStream outbound server. The outbound server streams out the LCRs to an XStream client application.

Syntax

```
DBMS_XSTREAM_ADM.START_OUTBOUND(
   server_name      IN  VARCHAR2);
```

Parameters

Table 244-37 START_OUTBOUND Procedure Parameters

Parameter	Description
server_name	The name of the outbound server being started. A NULL specification is not allowed. Do not specify an owner.

STOP_OUTBOUND Procedure

This procedure stops an XStream outbound server. The outbound server streams out the LCRs to an XStream client application.

Syntax

```
DBMS_XSTREAM_ADM.STOP_OUTBOUND(
   server_name  IN  VARCHAR2,
   force        IN  BOOLEAN   DEFAULT FALSE);
```

Parameters

Table 244-38 STOP_OUTBOUND Procedure Parameters

Parameter	Description
`server_name`	The name of the outbound server being stopped. A `NULL` specification is not allowed. Do not specify an owner.
`force`	If `TRUE`, then the procedure stops the outbound server and its capture process as soon as possible. If `FALSE`, then the procedure stops the outbound server after ensuring that there are no gaps in the set of applied transactions. The behavior of the apply component depends on the setting specified for the `force` parameter and the setting specified for the `commit_serialization` apply component parameter.