Extended Datatype Support (EDS) for Oracle Data Guard SQL Apply and Oracle Streams

Oracle Maximum Availability Architecture White Paper
June 2010

Maximum Availability Architecture

Oracle Best Practices For High Availability

ORACLE

Introduction

Oracle Data Guard SQL Apply and Oracle Streams are highly flexible features of Oracle Database that enable information sharing between databases. A primary capability of SQL Apply and Oracle Streams is the ability to replicate information from one database to another. Data Guard SQL Apply is designed for simpler, one-way replication of an entire database. Oracle Streams is a full featured information distribution solution capable of more sophisticated replication scenarios (for example, *n*-way multi-master replication, hub & spoke replication, many-to-one replication, selective replication of only a portion of a database, and data transformations).

SQL Apply and Oracle Streams provide database support for a wide variety of data types, but do not provide native support for data movement of some advanced data types. However, by using Extended Datatype Support, you can take advantage of the flexibility of SQL Apply and Oracle Streams to accommodate several more advanced data types. This Maximum Availability Architecture (MAA) white paper provides an overview of using Extended Datatype Support with SQL Apply and Oracle Streams.

Extended Datatype Support

Extended Datatype Support (EDS) enables SQL Apply and Oracle Streams to replicate changes to tables that contain data types not natively supported from one database to another. With EDS, SQL Apply and Oracle Streams can replicate natively supported data types plus the additional data types, as described in the following My Oracle Support (formerly OracleMetaLink) notes:

- Note 949516.1 for SQL Apply in Oracle Database 11*g* Release 2
- Note 559353.1 for SQL Apply in Oracle Database 10*g* Release 2 and Oracle Database 11*g* Release 1
- Note 556742.1 for Oracle Streams

Without EDS, only the tables that contain natively supported data types can be replicated, as described in the following documentation:

- For a list of data types natively supported by Oracle Streams refer to the *Oracle Streams Concepts and Administration* guide for the Oracle Database release you are using.

- For a list of data types natively supported by Data Guard SQL Apply refer to the *Oracle Data Guard Concepts and Administration* guide for the Oracle Database release you are using.

EDS for SQL Apply is available in Oracle Database 10*g* Release 2 starting with Patch Set 3 (10.2.0.4) and in Oracle Database 11*g* starting with Release 1 Patch Set 1 (11.1.0.7). EDS for Oracle Streams is available with any release of Oracle Database 10*g* Release 2 and any release of Oracle Database 11*g* Release 1.

How Extended Datatype Support Works

When EDS is in place, changes to tables that contain only supported data types are handled by SQL Apply or Streams in a normal fashion. Changes to tables that contain unsupported data types are handled by EDS, which represents unsupported data types with supported data types in a logging table. Then, SQL Apply or Streams replicates the changes to the logging table.

This is accomplished by creating the following database objects for each base table that contains a data type not natively supported:

- **Logging table**—The structure of the logging table is similar to the base table it represents, except that columns defined as an unsupported data type in the base table are instead represented as a natively supported type in the logging table. SQL Apply or Oracle Streams replicates the changes made to the logging table instead of the base table. For example, the table `CUST` contains the column `ADDRESS`, which has the user-defined data type `ADDRESS_TYP`:

  ```
  Table CUST
     custid          varchar2(10)
     name            varchar2(50)
     address         address_typ
  ```

  The `ADDRESS_TYP` data type is defined using all supported data types:

  ```
  Type ADDRESS_TYP
  ```

```
   street          varchar2(200)
   city            varchar2(200)
   state           char(2)
   zip             varchar2(20)
```

The logging table definition contains column definitions that are only supported types:

```
Columns for table CUST with ADDRESS_TYP expanded
   custid          varchar2(10)
   name            varchar2(50)
   street          varchar2(200)
   city            varchar2(200)
   state           char(2)
   zip             varchar2(20)
```

In addition to representing unsupported columns with only the supported data types, the logging table also contains the type of DML performed on the base table. There are additional differences between the base table and its logging table, but these differences do not need to be described for the purposes of this example. Note that the logging table does not continue to grow in size. See the Base table trigger description for an explanation.

- **Base table trigger**—A trigger on the base table is created and fired for every row modification performed by any DML operation on the base table. The trigger inserts a row into the logging table, and then immediately deletes the row to prevent growth of the logging table. The row data contains the DML type performed on the base table (`INSERT`, `UPDATE`, or `DELETE`) as well as the necessary values required to replay the statement on the target database.
- **Logging table trigger or DML handler**—When using EDS with SQL Apply, a trigger is created on the logging table on the target database. With Streams, a DML handler is created instead to process changes made to the logging table. When changes are made on the source database to the logging table by the base table trigger, each change is captured and propagated to the target database. When these changes are processed on the target database, the logging table trigger or DML handler is called to apply the changes to the base table on the target database.

Large Object (LOB) Support

LOB columns[1] in base tables that require EDS are handled as a special case because table triggers do not fire when there is a piecewise update (via OCI or DBMS_LOB) of a LOB. EDS uses a fast refreshable, ON COMMIT materialized view to capture and propagate all changes made to LOB columns. Changes to all other columns of the base table are propagated through the logging table.

> **Note:** EDS currently supports LOB columns outside of an object column. LOB attributes within an object column are not supported.

EDS Example

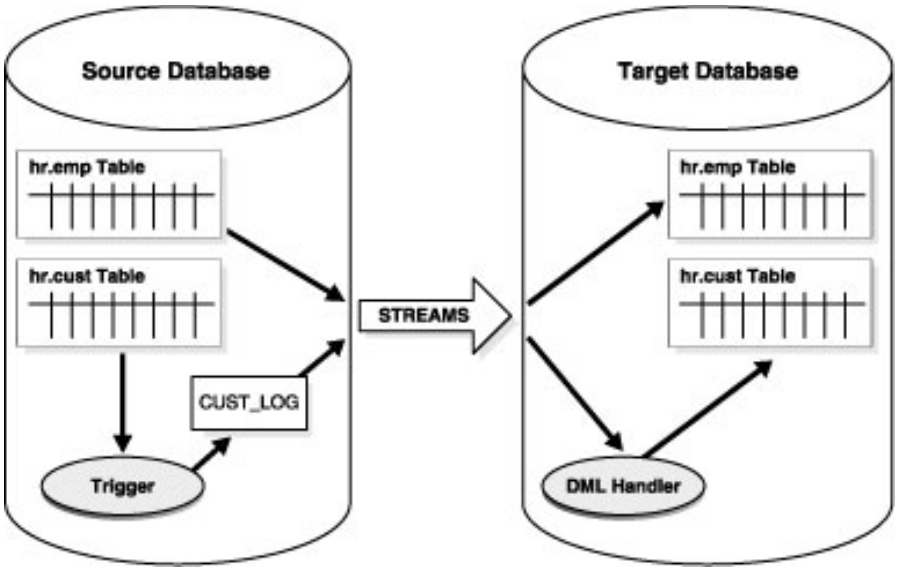
In the following example of using Streams with EDS, the HR.EMP table contains only columns natively supported by Streams and the table is replicated normally. The HR.CUST table contains an object column with a simple object type ADDRESS_TYP, which is not natively supported by Streams, but is supported by EDS. On the source database, EDS uses the logging table CUST_LOG and a trigger on HR.CUST. On the target database, EDS uses a DML handler.

When an insert occurs to HR.CUST, the base-table trigger fires to insert a row into CUST_LOG, indicating that the DML performed was INSERT. The object attributes for ADDRESS_TYP are flattened and inserted as their scalar equivalents. Oracle Streams captures and propagates changes that occur to CUST_LOG. When the Streams Apply component processes a change record for CUST_LOG, the DML handler is run to insert the row into the HR.CUST table on the target database.

Figure 1 shows an example of EDS processing a change record.

[1] SQL Apply and Oracle Streams natively support tables with LOB columns. However, tables with LOB columns that require EDS are only supported with the Oracle Streams EDS framework.

Figure 1: Example of EDS Processing a Change Record



EDS with SQL Apply works in a similar fashion, except that a logging table and trigger for the logging table are used in place of the DML Handler.

When Do I Use EDS?

Use EDS with SQL Apply or Oracle Streams to reduce downtime for certain planned maintenance activities, such as rolling database upgrade or platform migration, when your database contains data types that are not natively supported. EDS is intended to be in place temporarily while the planned maintenance activity is performed.

Rolling Database Upgrade with SQL Apply and EDS

Data Guard SQL Apply (logical standby) is the preferred method for performing a rolling database upgrade from Oracle Database 10*g* Release 2 to a later Oracle Database release (for example, to a later Oracle Database 10g Release 2 patch set, or to a later release such as Oracle Database 11*g*). Beginning with Oracle Database 10*g* Release 2 (10.2.0.4) Patch Set 3, SQL Apply supports the ability for triggers to fire on the logical standby database, which provides the basis of EDS.

To perform a rolling database upgrade with minimal downtime using SQL Apply with EDS, see the following MAA white papers:

- Rolling Database Upgrades for Physical Standby Databases using Transient Logical Standby 11g
- Rolling Database Upgrades using Data Guard SQL Apply

For additional information about EDS logging table and EDS trigger examples, see My Oracle Support (formerly Oracle*MetaLink*) Note 559353.1.

Rolling Database Upgrade and Platform Migration with Oracle Streams and EDS

To perform a database upgrade or a platform migration with minimal downtime using Oracle Streams with EDS, consider using the EDS package described in My Oracle Support Note 556742.1. This package implements the methods described in this white paper that you can use with any release of Oracle Database 10*g* Release 2 and any release of Oracle Database 11*g* Release 1.

How Do I Get EDS?

For additional information about using EDS with SQL Apply for Oracle Database 11*g* Release 2, see My Oracle Support Note 949516.1. For additional information about using EDS with SQL Apply for Oracle Database 10*g* Release 2 and Oracle Database 11*g* Release 1, see My Oracle Support Note 559353.1.

To obtain additional information about EDS for Streams, see My Oracle Support Note 556742.1.

Conclusion

This paper describes how, by using EDS with SQL Apply or Oracle Streams, you can maintain natively supported data types plus some advanced data types to:

- Make changes to tables that contain supported or unsupported data types:
  - Changes to supported data types are handled by SQL Apply or Oracle Streams in a normal fashion.
  - Changes to unsupported data types are handled by EDS, which represents and substitutes unsupported columns with only the supported data types.
- Handle many unsupported data types while in maintenance mode during a SQL Apply rolling upgrade process.

  The support for certain unsupported data types is provided as PL/SQL code and scripts that are installed temporarily while the planned maintenance activity is taking place.

Extended Datatype Support (EDS) for Oracle Data Guard SQL Apply and Oracle Streams
June 2010
Author: Douglas Utzig
Contributors: Alan Downing, Pat McElroy, Joe Meeks, Ashish Ray, Lawrence To, Viv Schupmann

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200
oracle.com

Oracle is committed to developing practices and products that help protect the environment

0109