**Mail Alban:**

Dat zal dan vooral over AWS DMS zijn gegaan denk ik, dat vaak in combinatie met SCS wordt gebruikt:  
<https://docs.aws.amazon.com/dms/?id=docs_gateway>  
<https://docs.aws.amazon.com/SchemaConversionTool/?id=docs_gateway>  
SCS is gratis te downloaden, ik heb er zodoende ook al even mee gespeeld ;)  
  
DMS kan data streamen van de bron- naar het target-endpoint. Dit stukje lijkt me specifiek relevant, over het opzetten van streaming CDC voor een Oracle source endpoint:  
<https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Source.Oracle.html#CHAP_Source.Oracle.CDC>  
  
Dan is er ook nog de serverless (niet echt natuurlijk, Amazon managed die voor je) ETL service Glue:  
<https://docs.aws.amazon.com/glue/?id=docs_gateway>

**Onderzoek**

select supplemental\_log\_data\_min, supplemental\_log\_data\_pk from v$database;

--no no

ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;

To enable primary key logging for all tables in the database enter:

ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (PRIMARY KEY) COLUMNS;

If replicating Oracle data to Redshift, enable supplemental logging on all columns for all tables in the database:

ALTER DATABASE ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;

Alternatively, to enable only for selected tables:

ALTER TABLE <schema>.<table name> ADD SUPPLEMENTAL LOG DATA (ALL) COLUMNS;

To activate your changes, enter:

alter system switch logfile;

If using Amazon RDS for Oracle, use the following commands instead:

exec rdsadmin.rdsadmin\_util.alter\_supplemental\_logging(p\_action => 'ADD');

exec rdsadmin.rdsadmin\_util.alter\_supplemental\_logging(p\_action => 'ADD', p\_type => 'PRIMARY KEY');

exec rdsadmin.rdsadmin\_util.switch\_logfile;

select supplemental\_log\_data\_min, supplemental\_log\_data\_pk from v$database;

If a table does not have a primary key you have two options: • You can add supplemental logging to all columns involved in the first unique index on the table (sorted by index name.) • You can add supplemental logging on all columns of the table. To add supplemental logging on a subset of columns in a table, that is those involved in a unique index, run the following command.

ALTER TABLE table\_name ADD SUPPLEMENTAL LOG GROUP example\_log\_group (ID,NAME) ALWAYS;

Create or configure a database account to be used by AWS DMS

CREATE SESSION

SELECT ANY TRANSACTION

SELECT on V\_$ARCHIVED\_LOG

SELECT on V\_$LOG

SELECT on V\_$LOGFILE

SELECT on V\_$DATABASE

SELECT on V\_$THREAD

SELECT on V\_$PARAMETER

SELECT on V\_$NLS\_PARAMETERS

SELECT on V\_$TIMEZONE\_NAMES

SELECT on V\_$TRANSACTION

SELECT on ALL\_INDEXES

SELECT on ALL\_OBJECTS

SELECT on ALL\_TABLES

SELECT on ALL\_USERS

SELECT on ALL\_CATALOG

SELECT on ALL\_CONSTRAINTS

SELECT on ALL\_CONS\_COLUMNS

SELECT on ALL\_TAB\_COLS

SELECT on ALL\_IND\_COLUMNS

SELECT on ALL\_LOG\_GROUPS

SELECT on SYS.DBA\_REGISTRY

SELECT on SYS.OBJ$

SELECT on DBA\_TABLESPACES

SELECT on ALL\_TAB\_PARTITIONS

SELECT on ALL\_ENCRYPTED\_COLUMNS

\* SELECT on all tables migrated

If you want to capture and apply changes (CDC) you also need the following privileges.

EXECUTE on DBMS\_LOGMNR

SELECT on V\_$LOGMNR\_LOGS

SELECT on V\_$LOGMNR\_CONTENTS

LOGMINING /\* For Oracle 12c and higher. \*/

\* ALTER for any table being replicated (if you want DMS to add supplemental logging)

For Oracle versions before 11.2.0.3, you need the following privileges. If views are exposed, you need the following privileges.

SELECT on DBA\_OBJECTS /\* versions before 11.2.0.3 \*/

SELECT on ALL\_VIEWS (required if views are exposed)

With AWS DMS, you can perform one-time migrations, and you can replicate ongoing changes to keep sources and targets in sync. If you want to migrate to a different database engine, you can use the AWS **Schema Conversion Tool (AWS SCT)** to translate your database schema to the new platform

The following diagram illustrates the AWS DMS replication process. For information about what AWS Regions support AWS DMS

At a high level, when using AWS DMS you do the following: • Create a replication server. • Create source and target endpoints that have connection information about your data stores. • Create one or more migration tasks to migrate data between the source and target data stores. A task can consist of three major phases: • The full load of existing data • The application of cached changes • Ongoing replication

**AWS DMS creates the target schema objects necessary to perform the migration**. However, AWS DMS takes a minimalist approach and creates only those objects required to efficiently migrate the data. In other words, **AWS DMS creates tables, primary keys, and in some cases unique indexes**, but doesn't create any other objects that are not required to efficiently migrate the data from the source. For example**, it doesn't create secondary indexes, nonprimary key constraints, or data defaults**

If your migration is heterogeneous (between two databases that use different engine types), you can use the AWS Schema Conversion Tool (AWS SCT) to generate a complete target schema for you. If you use the tool, any dependencies between tables such as foreign key constraints need to be disabled during the migration's "full load" and "cached change apply" phases. If performance is an issue, removing or disabling secondary indexes during the migration process helps.

Sources for AWS DMS

You can use the following data stores as source endpoints for data migration using AWS DMS.

**On-premises and EC2 instance databases**• Oracle versions 10.2 and later (for versions 10.x), 11g and up to 12.2, 18c, and 19c for the Enterprise, Standard, Standard One, and Standard Two editions

Note   
• Support for Oracle version 19c as a source is available in AWS DMS versions 3.3.2 and later.   
• Support for Oracle version 18c as a source is available in AWS DMS versions 3.3.1 and later.

**Amazon RDS instance databases, and Amazon Simple Storage Service (Amazon S3)**• Oracle versions 11g (versions 11.2.0.4 and later) and up to 12.2, 18c, and 19c for the Enterprise, Standard, Standard One, and Standard Two editions.

Note   
• Support for Oracle version 19c as a source is available in AWS DMS versions 3.3.2 and later.   
• Support for Oracle version 18c as a source is available in AWS DMS versions 3.3.1 and later.

An ARN for an AWS DMS resource uses the following syntax:

**arn:aws:dms:region:account number:resourcetype:resourcename**

In this syntax, the following apply:   
• region is the ID of the AWS Region where the AWS DMS resource was created, such as us-west-2.

• account number is your account number with dashes omitted. To find your account number, sign in to your AWS account at http://aws.amazon.com, choose My Account/Console, and then choose My Account.   
• resourcetype is the type of AWS DMS resource.

Migrating your source schema to your target database using AWS SCT

In this section, you use the AWS Schema Conversion Tool to migrate your source schema to your target database. To migrate your source schema to your target database

1. Install the AWS Schema Conversion Tool. For more information, see Installing, verifying, and updating the AWS SCT in the AWS Schema Conversion Tool User Guide

AWS DMS provides full support for using large object data types (BLOBs, CLOBs, and NCLOBs).

The following **source** endpoints have full LOB support: 42 AWS Database Migration Service User Guide Improved LOB performance   
• Oracle   
• Microsoft SQL Server   
• ODBC

The following target endpoints have full LOB support:   
• Oracle   
• Microsoft SQL Server

The following target endpoint has limited LOB support. You can't use an unlimited LOB size for this target endpoint.   
• Amazon Redshift   
• Amazon S3

**Configuring a CDC task to use Binary Reader with an RDS for Oracle source for AWS DMS**

You can configure AWS DMS to access the source Amazon RDS for Oracle instance redo logs using Binary Reader for CDC.

**Note**

To use Oracle LogMiner, the minimum required user account privileges are sufficient. For more information, see [User account privileges required on an AWS-managed Oracle source for AWS DMS](https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Source.Oracle.html#CHAP_Source.Oracle.Amazon-Managed.Privileges).

To use AWS DMS Binary Reader, specify additional settings and extra connection attributes for the Oracle source endpoint, depending on your AWS DMS version.

Binary Reader support is available in the following versions of Amazon RDS for Oracle:

* Oracle 11.2 – Versions 11.2.0.4V11 and later.
* Oracle 12.1 – Versions 12.1.0.2.V7 and later.
* Oracle 12.2 – All versions.
* Oracle 18.0 – All versions.
* Oracle 19.0 – All versions.

**To configure CDC using Binary Reader**

1. Log in to your Amazon RDS for Oracle source database as the master user and run the following stored procedures to create the server-level directories.
2. exec rdsadmin.rdsadmin\_master\_util.create\_archivelog\_dir;

exec rdsadmin.rdsadmin\_master\_util.create\_onlinelog\_dir;

1. Grant the following privileges to the Oracle user account that is used to access the Oracle source endpoint.
2. GRANT READ ON DIRECTORY ONLINELOG\_DIR TO *db\_user*;

GRANT READ ON DIRECTORY ARCHIVELOG\_DIR TO *db\_user*;

1. Set the following extra connection attributes on the Amazon RDS Oracle source endpoint:
   * **For RDS Oracle versions 11.2 and 12.1, set the following (RDS 11.2 ??? Kan dat dan toch ???)**.
   * useLogminerReader=N*;useBfile=Y;accessAlternateDirectly=false;useAlternateFolderForOnline=true;*

oraclePathPrefix=/rdsdbdata/db/{$DATABASE\_NAME}\_A/*;usePathPrefix=/rdsdbdata/log/;replacePathPrefix=true;*

* + For RDS Oracle versions 12.2, 18.0, and 19.0, set the following.

useLogminerReader=N*;useBfile=Y*

**Note**

Make sure there's no white space following the semicolon separator (;) for multiple attribute settings, for example oneSetting;thenAnother.

For more information configuring a CDC task, see [Configuration for CDC on an Oracle source database](https://docs.aws.amazon.com/dms/latest/userguide/CHAP_Source.Oracle.html#CHAP_Source.Oracle.CDC.Configuration).

Limitations on using Oracle as a source for AWS DMS

Limitations on using Oracle as a source for AWS DMS

The following limitations apply when using an Oracle database as a source for AWS DMS:

• AWS DMS doesn't support long object names (over 30 bytes).

• AWS DMS doesn't support function-based indexes.

• If you manage supplemental logging and carry out transformations on any of the columns, make sure

that supplemental logging is activated for all fields and columns. For more information on setting up

supplemental logging, see the following topics:

• **For a self-managed Oracle source database,** **see Setting up supplemental logging** (p. 86).

• For an AWS-managed Oracle source database, see Configuring an AWS-managed Oracle source for

AWS DMS (p. 90).

• AWS DMS doesn't support the multi-tenant container root database (CDB$ROOT). It does support a

PDB using the Binary Reader.

• AWS DMS doesn't support deferred constraints.

• AWS DMS supports the rename table table-name to new-table-name syntax for all supported

Oracle versions 11 and later. This syntax isn't supported for any Oracle version 10 source databases.

• AWS DMS doesn't replicate data changes that result from partition or subpartition operations (ADD,

DROP, EXCHANGE, and TRUNCATE). Such updates might cause the following errors during replication:

• For ADD operations, updates and deletes on the added data might raise a "0 rows affected" warning.

• For DROP and TRUNCATE operations, new inserts might raise "duplicates" errors.

• EXCHANGE operations might raise both a "0 rows affected" warning and "duplicates" errors.

To replicate changes that result from partition or subpartition operations, reload the tables in

question. After adding a new empty partition, operations on the newly added partition are replicated

to the target as normal.

• AWS DMS doesn't support data changes on the target that result from running the CREATE TABLE AS

statement on the source. However, the new table is created on the target.

• AWS DMS doesn't capture changes made by the Oracle DBMS\_REDEFINITION package, for example

the table metadata and the OBJECT\_ID field.

• AWS DMS maps empty BLOB and CLOB columns to NULL on the target.

• When capturing changes with Oracle 11 LogMiner, an update on a CLOB column with a string length

greater than 1982 is lost, and the target is not updated.

• During change data capture (CDC), AWS DMS doesn't support batch updates to numeric columns

defined as a primary key.

• AWS DMS doesn't support certain UPDATE commands. The following example is an unsupported

UPDATE command.

UPDATE TEST\_TABLE SET KEY=KEY+1;

Here, TEST\_TABLE is the table name and KEY is a numeric column defined as a primary key.

• AWS DMS truncates any data in LONG or LONG RAW columns that is longer than 64 KB to 64 KB.

• AWS DMS doesn't replicate tables whose names contain apostrophes.

• AWS DMS doesn't support CDC from dynamic views.

• When you use AWS DMS Binary Reader to access the redo logs, AWS DMS doesn't support CDC for

index-organized tables with an overflow segment. Or you can consider using LogMiner for such tables.

• When you use Oracle LogMiner to access the redo logs, AWS DMS has the following limitations:

• For Oracle 12 only, AWS DMS doesn't replicate any changes to LOB columns.

• For all Oracle versions, AWS DMS doesn't replicate the result of UPDATE operations on XMLTYPE and

LOB columns

• AWS DMS doesn't replicate results of the DDL statement ALTER TABLE ADD column data\_type

DEFAULT default\_value. Instead of replicating default\_value to the target, it sets the new

column to NULL. Such a result can also happen even if the DDL statement that added the new

column was run in a prior task.

If the new column is nullable, Oracle updates all the table rows before logging the DDL itself. As a

result, AWS DMS captures the changes but doesn't update the target. With the new column set to

NULL, if the target table has no primary key or unique index subsequent updates raise a "zero rows

affected" message.

• Oracle LogMiner doesn't support connections to a pluggable database (PDB). To connect to a PDB,

access the redo logs using Binary Reader.

• **When you use Binary Reader, AWS DMS has these limitations**:

• It doesn't support table clusters.

• It supports only table-level SHRINK SPACE operations. This level includes the full table, partitions,

and subpartitions.

• It doesn't support changes to index-organized tables with key compression.

• It doesn't support implementing online redo logs on raw devices.

• AWS DMS doesn't support connections to an Amazon RDS Oracle source using an Oracle Automatic

Storage Management (ASM) proxy.

• AWS DMS doesn't support virtual columns.

• AWS DMS doesn't support the ROWID data type or materialized views based on a ROWID column.

• AWS DMS doesn't load or capture global temporary tables.

• For S3 targets using replication, enable supplemental logging on every column so source row

updates can capture every column value. An example follows: alter table yourtablename add

supplemental log data (all) columns;.