

Kunde:	DOAGNews
Ort, Datum:	Artikel im Heft Q3 /2006
Thema / Themen:	Artikel von merlin.zwo
Projekt:	Change Data Capture (CDC)
Autor:	Jana Mildorf

- Oracle & Technologien
- Systementwicklung
- Individuelle Lösungen

OLAP Systeme bestehen oftmals getrennt von den Produktivdatenbanken. Zum Abgleich der Daten werden nächtliche Importe, aktualisierende Inserts oder andere Replikationsmechanismen verwendet. Durch diese Trennung des OLAP Systems von der OLTP Datenbank können aktuelle Änderungen nicht in die Auswertungen einfließen.

Das Change Data Capture (CDC) von Oracle protokolliert jede Änderung auf dem Produktivsystem und stellt sie in so genannten change tables zur weiteren Verwendung bereit. Eine change table ist physisch gesehen eine partitionierte Tabelle (partitioned table). Sie beinhaltet Informationen über die Art der Operation, den Benutzernamen, Werte der einzelnen Spalten aus der Originaltabelle, sowie ein Timestamp und die SCN (System Change Number). Eine oder mehrere change tables werden zu einem change set zusammengefasst. Innerhalb eines change sets kann man joinen. Dadurch sind komplexe Abfragen, die sich auf einen bestimmten Zeitpunkt beziehen, oder auch Vergleiche von Zeitpunkten auf verschiedenen Datenbanken möglich ohne einen Datenbank-Link zu verwenden.

Beim CDC sind die Aufgaben klar verteilt: Es gibt einen Herausgeber (publisher) und einen Abonnenten (subscriber). Change sets und change tables gehören dem Herausgeber. Er erstellt diese und gibt sie dem Abonnenten frei. Der Abonnent legt sich pro change table eine View mit den gewünschten Änderungsdaten an. Ein Abonnent kann eine Person oder eine Anwendung sein. Beide, Herausgeber und Abonnent, verwenden PL/SQL-Packages um ihre jeweiligen Aufgaben zu erfüllen.

Typen des CDC

Beim Change Data Capture unterscheidet man zwischen zwei Typen, dem synchronen CDC und dem asynchronen CDC. Das asynchrone CDC gibt es wiederum in zwei Modi: lokal (HotLog) und nicht lokal (AutoLog).

Das synchrone CDC gibt es seit der Oracle 9i in der Enterprise Edition, in der 10g ist es bereits in der Standard Edition enthalten. Das asynchrone CDC gibt es seit der 10g EE.

Funktionsweisen des CDC

Sobald eine DML-Operation durchgeführt wird, zeichnet das synchrone CDC die geänderten Daten mit Hilfe eines Systemtriggers auf. Der Trigger wird automatisch auf der Ursprungstabelle angelegt.

Der Vorteil des synchronen Verfahrens ist die zeitidentische Erfassung. Allerdings können die Transaktionen durch die eingesetzten Trigger verlangsamt werden. Problematisch wird es, wenn nicht in die change table geschrieben werden kann. Dann kann das ganze System zum Stillstand kommen.

Das asynchrone, lokale CDC (HotLog) schreibt die Änderungsinformationen mit Hilfe von ORACLE Streams in die change tables. Die Änderungsinformationen werden durch das sogenannte hot mining aus den Redolog Dateien geholt. Dadurch sind die Informationen zeitnah und das System ist bei einer hohen Schreibrate stabil. Leider unterstützt Oracle Streams bestimmte Datenstrukturen nicht: BFILE, USER-DEFINED TYPES, OBJECT TYPES, REFS, VARRAYS sowie NESTED TABLES. Partitioned IOTs und IOTs mit der Overflow-Klausel werden ebenfalls nicht unterstützt.

Beim asynchronen, nicht lokalen CDC (AutoLog) werden die Redolog-Dateien per RFS (remote file server) auf das Zielsystem kopiert und dann mit Oracle Streams ausgelesen (Downstream Capture).

Der Abonnent (subscriber)

Der Abonnent kann durch den Herausgeber (publisher) oder einem anderen DBA-User angelegt werden. Die Systemprivilegien für den Abonnenten können minimal sein, er muss sich nur auf der DB anmelden können. Als Objektprivilegien benötigt der Abonnent SELECT Rechte auf die entsprechende change table, auf die Originaltabelle sind diese nicht nötig. Der Abonnent benutzt die Package DBMS_CDC_SUBSCRIBE. Diese ist für jeden User zugänglich, da sie ein GRANT EXECUTE TO PUBLIC und ein PUBLIC SYNONYM besitzt.

```
CREATE USER cdcssubscriber  
IDENTIFIED BY <passwort>  
DEFAULT TABLESPACE users TEMPORARY TABLESPACE temp  
/  
GRANT CONNECT TO cdcpublisher;
```

Der Abonnent kann der View ALL_SOURCE_TABLES entnehmen, auf welche change table er Leserechte besitzt. Dann legt er für diese change table ein Abonnement an, indem er die Procedure CREATE_SUBSCRIPTION aufruft.

```
BEGIN
DBMS_CDC_SUBSCRIBE.CREATE_SUBSCRIPTION (
  change_set_name => 'CHANGE_SET_EMP'
  description      => 'Änderungen der Tabelle EMP'
  subscription_name => 'EMP_ABO' );
END;
```

Der subscription name ist der eindeutige Name des Abonnements. Der change set name ist der Name des change sets. Die description ist eine frei wählbare Beschreibung.

Dieses Abonnement kann vom Abonnenten mit DBMS_CDC_SUBSCRIBE.DROP_SUBSCRIPTION wieder gelöscht werden. Änderungen an dem Abonnement sind nicht möglich, d.h. es muss gelöscht und neu angelegt werden.

Um das Abonnement einer Tabelle und deren Spalten zuzuordnen, wird die Package SUBSCRIBE aufgerufen.

```
BEGIN
DBMS_CDC_SUBSCRIBE.SUBSCRIBE(
  subscription_name => 'EMP_ABO',
  source_schema     => 'SCOTT',
  source_table      => 'EMP',
  column_list       => 'EMPNO, ENAME, JOB, MGR, HIREDATE,
SAL, COMM, DEPTNO',
  subscriber_view   => 'EMP_ABO_VIEW' );
END;
```

Source schema, source table sowie die column list beziehen sich auf die Originaltabelle. Es können nur die Spalten angegeben werden, die auch in der change table enthalten sind. Der Parameter subscriber_view legt den Namen der View für die Abfragen fest. Will man über mehrere Abonnements joinen, muss man die Procedure SUBSCRIBE mit der publication id aufrufen, anstelle des source schemas und der source table. Es werden dadurch zwei Views erstellt, über die man anschließend joinen kann. Voraussetzung ist, dass mindestens eine Spalte in beiden Basistabellen identisch heißen. Die Spaltennamen und publication_id lassen sich aus der View ALL_PUBLISHED_COLUMNS auslesen.

```
BEGIN
  DBMS_CDC_SUBSCRIBE.SUBSCRIBE(
    subscription_name => 'EMP_ABO',
    publication_id    => 1,
    column_list       => 'EMPNO, ENAME, JOB, MGR, HIREDATE, SAL,
COMM, DEPTNO',
    subscriber_view   => 'EMP_ABO1_VIEW');
END;
```

```
BEGIN
  DBMS_CDC_SUBSCRIBE.SUBSCRIBE(
    subscription_name => 'EMP_ABO',
    publication_id    => 2,
    column_list       => 'EMPNO, ENAME, JOB, MGR, HIREDATE, SAL,
COMM, DEPTNO',
    subscriber_view   => 'EMP_ABO2_VIEW');
END;
```

Durch die Procedure ACTIVATE_SUBSCRIPTION wird das Abonnement aktiviert.

```
BEGIN
  DBMS_CDC_SUBSCRIBE.ACTIVATE_SUBSCRIPTION(
    subscription_name => 'EMP_ABO';
END;
```

Nun muss festgelegt werden, was gelesen werden soll. Mit der Procedure EXTEND_WINDOW werden die Änderungsdaten geholt.

```
BEGIN
  DBMS_CDC_SUBSCRIBE.EXTEND_WINDOW(
    subscription_name => 'EMP_ABO');
END;
```

In der in der subscriber view (hier: EMP_ABO_VIEW) kann man nun die Änderungsdaten ansehen.

Der Herausgeber (publisher)

Der Herausgeber verwendet die Package DBMS_CDC_PUBLISH und besitzt das Recht auf die change tables SELECT-Rechte zu vergeben. Beim asynchronen CDC benötigt er zusätzlich die DBA Rolle und alle Rechte eines Oracle Streams Administrators (DBMS_STREAMS_AUTH). Beim nicht lokalen Modus benötigt er zusätzlich ein CREATE SEQUENCE Recht.

```
CREATE USER cdcpublisher
IDENTIFIED BY <passwort>
DEFAULT TABLESPACE users
TEMPORARY TABLESPACE temp
QUOTA UNLIMITED ON SYSTEM
```

```
QUOTA UNLIMITED ON sysaux
/
GRANT CONNECT, RESOURCE TO cdcpublisher;
GRANT CREATE TABLESPACE TO cdcpublisher;
GRANT EXECUTE_CATALOG_ROLE TO cdcpublisher;
GRANT SELECT_CATALOG_ROLE TO cdcpublisher;
```

/* Asynchrones CDC */

```
GRANT DBA TO cdcpublisher;
BEGIN
DBMS_STREAMS_AUTH.grant_admin_privilege('CDCPUBLISHER');
END;
```

Der Herausgeber stellt die geänderten Daten zur Verfügung. Dazu muss er ein change_set und die dazu gehörigen change_tables einrichten. Beim lokalen Modus muss man zusätzlich noch eine change source anlegen. Diese ist bei den anderen beiden Verfahren bereits vordefiniert: SYNC_SOURCE und HOTLOG_SOURCE

Das synchrone CDC basiert auf Systemtriggern. Sie werden im Schema SYS angelegt und ziehen beim CREATE, ALTER oder DROP Befehl auf der Datenbank. Dabei wird die Procedure DBMS_CDC_IPUBLISH.CHANGE_TABLE_TRIGGER aufgerufen.

Vorraussetzung für das synchrone CDC ist lediglich die Aktivierung von Java und eine JAVA_POOL_SIZE von mindestens 50M.

Das asynchrone CDC holt sich die geänderten Daten aus den Redolog-Dateien. Dazu muss die zu protokollierende Datenbank im Archivelogmodus laufen und es dürfen keine direct path- oder nologging- Operationen auf den betroffenen Tabellen ausgeführt werden. Sicherheitshalber kann man das Force Logging aktivieren. Beim nicht lokalen Modus müssen das SYS Passwort identisch und beide Datenbanken im Archivelogmodus sein.

```
ALTER DATABASE FORCE LOGGING;
```

Eine weitere Grundvoraussetzung ist das Aktivieren des Supplemental Logging. Das Supplemental Logging schreibt erweiterte Informationen in das Logfile, sobald ein UPDATE durchgeführt wird. Hierbei muss auf die erhöhte Menge von Redo-Informationen und Anzahl der Checkpoints geachtet werden.

```
ALTER DATABASE ADD SUPPLEMENTAL LOG DATA;
```

Desweiteren müssen noch einige Initialisierungsparameter überprüft bzw. angepasst werden. Siehe hierzu Oracle® Database Data Warehousing Guide 10g Release 1 (10.1) S. 16-21 ff.

Einrichten des Log Miner Datadictionary

Für das asynchrone, nicht lokale CDC muss das Log Miner Data Dictionary auf der lokalen Datenbank eingerichtet werden. Als Parameter muss die erste SCN übergeben werden.

```
DECLARE
f_scn NUMBER := ?;
BEGIN
DBMS_CAPTURE_ADM.BUILD(f_scn);
END;
/
```

Einrichten eines change set

Durch die Procedure CREATE_CHANGE_SET wird ein change set angelegt. Der Parameter change_source_name mit dem Schlüsselwort SYNC_SOURCE stellt das Verfahren auf synchron ein.

Eine change set wird beim asynchronen, lokalen CDC ebenfalls mit der Procedure CREATE_CHANGE_SET eingerichtet. Der Parameter change_source_name muss allerdings auf HOTLOG_SOURCE gesetzt sein. Dadurch wird Oracle Streams vollständig konfiguriert. Die Konfiguration lässt sich in der View DBA_CAPTURE überprüfen.

```
BEGIN
  DBMS_CDC_PUBLISH.CREATE_CHANGE_SET(
    change_set_name => 'CHANGE_SET_EMP',
    change_source_name => 'HOTLOG_SOURCE');
END;
```

Beim asynchronen, nicht lokalen Verfahren wird ein change set auf der Zieldatenbank in zwei Schritten angelegt. Zuerst wird eine change source definiert. Der Parameter first_scn muss der ersten SCN des Log Miner Dictionary entsprechen. Anschließend wird das eigentliche change set erstellt.

```
BEGIN
DBMS_CDC_PUBLISH.CREATE_CHANGE_SOURCE(
change_source_name => 'quelle',
source_database => 'PROD',
first_scn => '??');
END;

BEGIN
DBMS_CDC_PUBLISH.CREATE_CHANGE_SET(
change_set_name => 'CHANGE_SET_EMP',
change_source_name => 'HOTLOG_SOURCE');
END;
```

Einrichten einer change table

Change tables für das synchrone CDC liegen immer in der lokalen Datenbank. Beim asynchronen CDC hängt das vom Modus ab: bei dem lokalen Modus (HotLog) liegt die change table ebenfalls auf der lokalen Datenbank. Von hier aus können die Daten dann auf herkömmliche Weise in das DataWarehouse System übertragen werden.

Beim nicht lokalen Modus (AutoLog) befinden sich die change tables direkt auf der Zieldatenbank. Der Herausgeber steuert den Zugriff auf die change table durch das Vergeben von Select-Rechten.

```
BEGIN
DBMS_CDC_PUBLISH.CREATE_CHANGE_TABLE(
owner          => 'cdcpublisher',
change_table_name => 'EMPHI_CT',
change_set_name  => 'CHANGE_SET_EMP',
source_schema   => 'SCOTT',
source_table     => 'EMP',
column_type_list => 'EMPNO NUMBER(4), ENAME VARCHAR2(10),
JOB VARCHAR2(9), MGR NUMBER(4), HIREDATE DATE, SAL
NUMBER(7,2), COMM NUMBER(7,2), DEPTNO NUMBER(2)',
capture_values   => 'both',
rs_id           => 'y',
row_id          => 'n',
user_id         => 'y',
timestamp       => 'y',
object_id       => 'n',
source_colmap    => 'n',
target_colmap    => 'y',
options_strings  => 'TABLESPACE users');
END;
```

Der Parameter source_colmap steht beim synchrone Verfahren auf 'y', beim asynchronen auf 'n'.

Beim synchronen Verfahren werden mit Anlegen der change table die Trigger automatisch aktiviert. Das CDC ist somit funktionsfähig. Beim asynchronen Verfahren müssen noch zwei Schritte durchgeführt werden:

Festlegen der SCN, ab der Oracle Streams Daten erfasst:

```
BEGIN
DBMS_CAPTURE_ADM.PREPARE_TABLE_INSTANTIATION
    (table_name => 'SCOTT.EMP' );
END;
```

Starten der Streams-Prozesse

```
BEGIN
    DBMS_CDC_PUBLISH.ALTER_CHANGE_SET(
        change_set_name => 'CHANGE_SET_EMP',
        enable_capture   => 'y' );
END;
```

Mit den Views CHANGE_TABLES, DBA_SOURCE_TABLES und DBA_PUBLISHED_COLUMNS kann man das CDC verwalten.

Bereinigen der change tables

Mit den Procedures DBMS_CDC_PUBLISH.PURGE, DBMS_CDC_PUBLISH.PURGE_CHANGE_SET und DBMS_CDC_PUBLISH.PURGE_CHANGE_TABLE kann der Herausgeber bei Bedarf die change table aufräumen.

Die Procedure PURGE löscht alle Daten aus der change table, die nicht mehr benötigt werden. Mit den zwei anderen Procedures wird die change_table selbst bzw. das ganze change_set gelöscht.

Man kann das Aufräumen der Daten auch mit einem DB-Job und DBMS_CDC_PUBLISH.PURGE automatisieren.

Zusammenfassung der einzelnen Verfahren

Das synchrone CDC kann ab der 9i eingesetzt werden und ist in der 10g bereits in der SE erhältlich. Die Änderungsdaten sind bei diesem Verfahren am aktuellsten, da sie nicht erst in die Redolog-Dateien geschrieben werden müssen. Allerdings eignet sich dieses Verfahren nicht für Datenbanken, auf denen viele Transaktionen stattfinden.

Für das asynchrone CDC ist eine 10g EE sowie eventuelle Änderungen an den Systemparametern erforderlich. Beim nicht lokalen Verfahren muss ein unterbrechungsfreier Netzverkehr gewährleistet werden, da die archivierten Redolog-Dateien mit Hilfe des Oracle Net an die Zieldatenbank gesendet werden. Insgesamt ist das asynchrone CDC technologisch um einiges gereifter als das synchrone Verfahren und lässt sich auch in komplexere Systeme einbinden.

Jana Mildorf
Systementwicklerin
merlin.zwo InfoDesign GmbH & Co. KG
Tel.: 07052-933-669
Fax: 07052-933-670
E-Mail: jana.mildorf@merlin-zwo.de