Welches Change-Data-Capture-Verfahren ist geeignet?

Dominik Schuster
areto consulting gmbh
Köln

Schlüsselworte
Change Data Capture, Delta Extract, Data Warehouse

Einleitung
In einer schnell und dynamisch wachsenden Welt entwickelt sich der Faktor „Informationsgewinn“ gerade in den letzten Jahrzenten zu einer entscheidenden Rolle, um am Markt bestehen und Wettbewerbsvorteile generieren zu können. Um dies zu erreichen, müssen gezielte und prägnante Abfragen auf Geschäftsdaten ermöglicht werden, die eine solide und aktuelle Datenbasis voraussetzen. Um die Aktualität der Daten im Data Warehouse gewährleisten zu können, müssen Änderungen an Datenbeständen (Deltas) in Quellsystemen ausfindig gemacht und an das Data Warehouse propagiert werden. Hierfür wurden verschiedene Verfahren entwickelt, die diese Aufgaben komfortabel, sowie automatisiert erfüllen.
Die Auswahl eines geeigneten Verfahrens kann sich durch die individuellen Bedürfnisse der Unternehmen und Anzahl der möglichen Methoden als nicht trivial erweisen.
Eines der ersten und zugleich wichtigsten Entscheidungskriterien ist der Zustand des Quellsystems. Dabei führen Alter, Umfang des nötigen Zugriffs und Quellsystem-Historisierung zum vorzeitigen Ausschluss einiger Verfahren. Andere Verfahren stellen sich als echte Allzweckwaffen heraus, wohingegen manche eine bestimmte Datenbank-Distribution (Oracle) voraussetzen.
Im Folgenden werden ausgewählte Delta-Ermittlungsverfahren betrachtet und bewertet.

Snapshot Differential
Ein grundlegendes Verfahren stellt die Ermittlung von Deltas über sog. Snapshots dar. Hierfür wird eine alte Version einer Tabelle mit einer neueren verglichen, um das „Differential“, also die Unterschiede festzustellen. Dafür muss zu jedem Extraktionsintervall das volle Abbild der Tabelle kopiert werden, was bei großen Datenmengen zu Problemen führen kann. Die benötigte Minus-Operation ist außerdem vergleichsweise aufwendig, da jedes Tupel verglichen werden muss. Deletes sowie einfache Updates können bei Bedarf ausfindig gemacht werden. Einschränkung dabei stellen mehrfach Änderungen eines Datensatzes in einem Extraktionsintervall dar, welche aber für nahtlose Historisierung durchaus erwünscht sein können. Mit anderen Worten: Die Granularität, mit der Änderungen erfasst werden können, hängt maßgeblich vom gewählten Extraktions-Intervall ab.

Audit Columns
Eine weitere Möglichkeit zur Delta-Ermittlung bieten Quellsysteme, die Prüfspalten zur Verfügung stellen. Dabei wird jedem neuen oder geänderten Datensatz im Quellsystem, in einer zusätzlichen Spalte ein Zeitstempel, oder eine Versionsnummer zugeordnet. Diese Prüfspalte kann dann im nächsten Extraktionsintervall als Selektionskriterium verwendet, und somit nur die aktuellen Daten abgefragt werden.
Dieses Verfahren birgt zwei Schwächen. Zum einen werden keine Mehrfachänderungen von Datensätzen zwischen zwei Extraktionsintervallen erfasst, da in operationalen Systemen vorzugsweise nur die letzte Änderung eines Datensatz vorbehalten wird. Zum anderen können Änderungen verloren gehen, wenn Transaktionen durchgeführt werden, aber zu Beginn des Extraktionsprozesses noch nicht

festgeschrieben wurden. In diesem Fall werden die Datensätze von der aktuellen Extraktion nicht erfasst, erhalten aber den entsprechenden Zeitstempel. Dadurch können sie auch im nächsten Extraktionsintervall nicht entdeckt werden und die Änderung bleibt unbemerkt (Pending Commits). Die Mechanik den Zeitstempel hinzuzufügen sollte außerdem in der Datenbank über einen Trigger geschehen. Ein Zeitstempel der schon auf Applikationsseite geschrieben wird birgt die Gefahr, dass unerwartete Schreibvorgänge (bspw. vom DBA ausgeführte Datenverschiebungen) ohne Zeitstempel geschrieben werden.

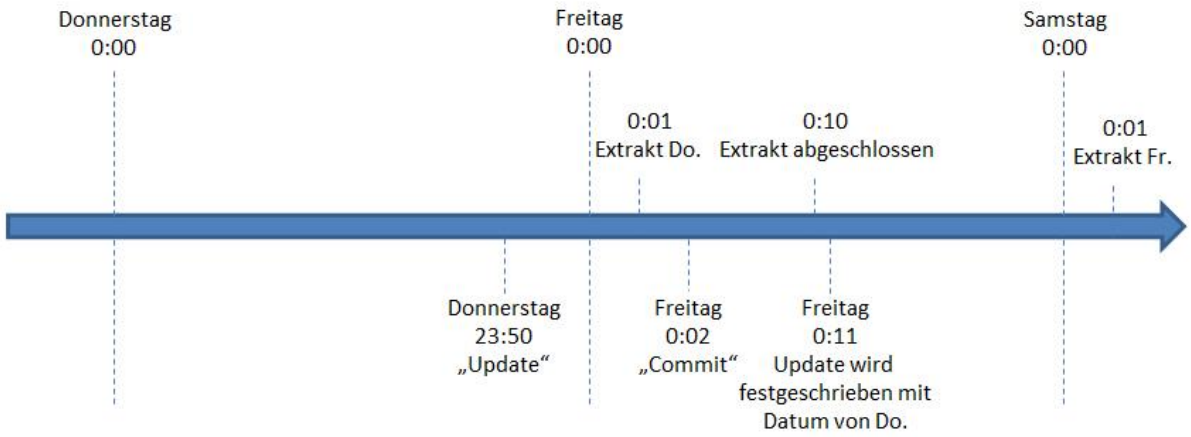

Abb. 1 - Pending Commits

Den Pending Commits kann mit verschiedenen Möglichkeiten entgegengewirkt werden. Das zu ladende Zeitintervall könnte vergrößert werden mit Hinnahme dadurch resultierender Redundanzen. Ab Oracle 10g kann außerdem die Prüfspalte ORA_ROWSCN verwendet werden, die dem Datensatz eine System Change Number (SCN) zuordnet, die erst beim eigentlichen Commit des Datensatzes geschrieben wird.

Um ORA_ROWSCN effektiv nutzen zu können, sollte diese auf Datensatzebene gespeichert werden und nicht auf Blocksatzebene. Dafür muss bei der Erstellung einer entsprechenden Tabelle die Klausel ROWDEPENDENCIES aktiviert werden. Dies kann einen großen Eingriff in das Quellsystem bedeuten, da bestehende Tabellen nachträglich nicht geändert werden können!

Neben den Allzweckwaffen „Snapshot Differential“ und „Audit Columns“ bietet Oracle alternativen an.

Oracle Change Data Capture (CDC)
Oracle bietet mit CDC ein umfangreiches Framework an um Deltas zu ermitteln. Dazu gehören im Allgemeinen ein eigens für die Delta-Ermittlung geschaffene CDC-Staging Area, die sich je nach CDC-Verfahren mit auf dem Quellsystem oder im Data Warehouse befindet. Weiterhin besteht sie aus zwei unterschiedlichen Schemata, dem Publisher und Subscriber.
Der Publisher stellt alle Funktionalitäten bereit, um Änderungen an Datenbeständen zu ermitteln und darzustellen. Unterschiedliche Quellsysteme werden durch Change-Sources repräsentiert. Change-Sources können je nach CDC-Verfahren vordefiniert sein, oder müssen selbst definiert werden. Anschließend können Change-Sets und Change-Tables erstellt werden. Change-Tables beinhalten die eigentlichen Änderungen in Form von Logical Change Records (LCR). Mehre Change-Tables werden in Change-Sets zusammengefasst, die die Art des CDC-Verfahrens, also die Art, wie Änderungen erkannt werden sollen, beschreiben.
Subscriber regeln den multiplen Zugriff auf die vom Publisher bereitgestellten Daten. Hierfür werden Subscriptions erstellt, die ein oder mehrere Changetables eines Changesets verwalten und den Datenzugriff in Form von Views ermöglichen. Der ETL-Prozess des Data Warehouse kann anschließend auf die Views des Subscribers zugreifen.

Abbildung 2 dient dem Verständnis der grundlegenden Logik des CDC-Frameworks.

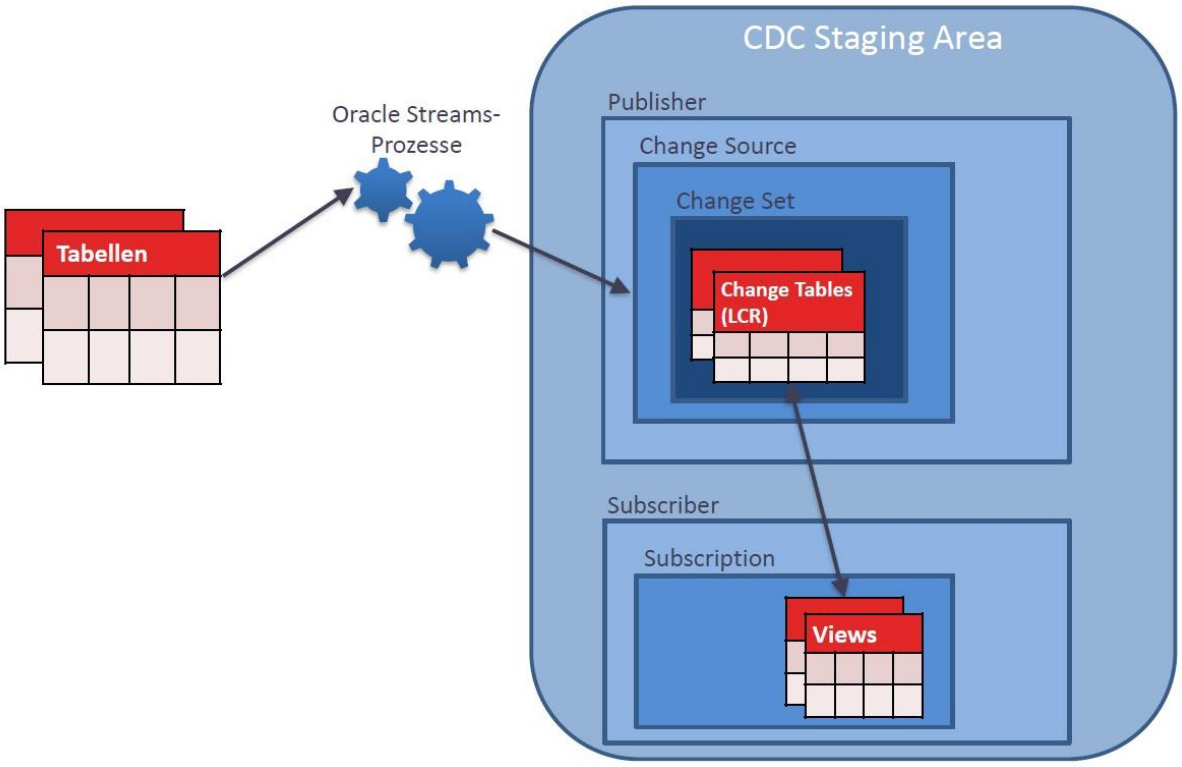

Abb. 2 - Grundlegender Aufbau von Oracle Change Data Capture

Deltas können mit Oracle CDC Trigger-basiert (synchron) und Log-gestützt (asynchron) ermittelt werden. Konkret kann zwischen folgenden Verfahren unterschieden werden:

- Synchrones CDC
- Asynchrones CDC AutoLog Online / Archive
- Asynchrones HotLog
- Asynchrones Distributed HotLog

Synchrones CDC
Zur Propagation und anschließenden Darstellung der Änderungen wird auf Trigger zurückgegriffen. Dies ist mit einem invasiven Eingriff in das Quellsystem verbunden, wenn diese nicht im Vorhinein vorhanden sind. Die Bereitstellung der Datenänderungen wird außerdem fest in die Transaktion integriert, sodass diese erst nach beschreiben der Change Tables im Publisher abgeschlossen ist. Dies ist zugleich größter Vor- und Nachteil dieses Verfahrens, da Änderungen zwar in Echtzeit propagiert werden können, das Quellsystem aber einer höheren Belastung ausgesetzt wird.

Die Implementierung vom Synchronen CDC findet rein auf dem Quellsystem statt. Als Change-Source muss die vordefinierte „sync_source“, auf dem Quellsystem verwendet werden. Ein User für die Funktionalitäten des Publishers muss erstellt und mit den nötigen Rechten ausgestattet werden.

```
CREATE USER cdc_publisher IDENTIFIED EXTERNALLY DEFAULT TABLESPACE
ts_cdc_publisher
QUOTA UNLIMITED ON SYSTEM;
QUOTA UNLIMITED ON SYSAUX;
GRANT CREATE SESSION TO cdc_publisher;
GRANT CREATE TABLE TO cdc_publisher;
GRANT CREATE TABLESPACE TO cdc_publisher;
GRANT UNLIMITED TABLESPACE TO cdc_publisher;
GRANT SELECT_CATALOG_ROLE TO cdc_publisher;
GRANT EXECUTE_CATALOG_ROLE TO cdc_publisher;
GRANT ALL ON [Tabelle] TO cdc_publisher;
GRANT EXECUTE ON DBMS_CDC_PUBLISH TO cdc_publisher;
```

Dazu gehört selbstverständlich auch das Execute-Recht zum Ausführen der vom Datenbank Managementsystem bereitgestellten CDC Prozeduren (ab Release 9.2).
Anschließend kann der User „cdc_publisher“ das benötigte Change Set erstellen und dabei die vorher definierte change_source angeben.

```
BEGIN
    DBMS_CDC_PUBLISH.CREATE_CHANGE_SET(
    change_set_name    => 'Produkte_taeglich',
    description        => '[Beschreibung des Change Set]',
    change_source_name => 'SYNC_SOURCE');
END;
```

Analog dazu können nun mit der Prozedur create_change_table, Tabellen für die LCR’s erstellt werden. Als Parameter können hier die Quelltabelle, zugehöriges Change Set und die gewünschten Attribute für das LCR übergeben werden.
Über den Parameter „capture_values“ kann definiert werden, ob nur das alte Tupel, das neue Tupel, oder beide aufgeführt werden. Mit dem optionalen Parameter „ddl_markers“ kann die entsprechende DDL-Anweisung (Delete, Update, Insert) anhand eines Flags repräsentiert werden.

```
BEGIN
   DBMS_CDC_PUBLISH.CREATE_CHANGE_TABLE(
   owner             => 'CDC_Publisher',
   change_table_name => 'CT_Produkte',
   change_set_name   => '[Produkte_taeglich]',
   source_schema     => 'SH',
   source_table      => 'PRODUKTE',
   column_type_list  => 'PROD_ID NUMBER(6),
                         PROD_NAME VARCHAR2(50),
                         PROD_LIST_PRICE NUMBER(8,2)',
   capture_values    => '['OLD' / 'NEW' / 'BOTH'],
   rs_id             => 'y',
   row_id            => 'n',
   user_id           => 'n',
   timestamp         => 'n',
   object_id         => 'n',
   source_colmap     => 'y',
   target_colmap     => 'y',
   options_string    => 'TABLESPACE TS_CDC_Publisher),
   ddl_markers       => 'y';
END;
```

Abschließend wird ein Subscriber mit den nötigen Leserechten für die Change Tables erstellt, der dann als Schnittstelle zum Data Warehouse fungiert.

```
GRANT SELECT ON cdc_Publisher.CT_Produkte TO subscriber1;
```

Passend zu den Change-Sets können vom Subscriber „Subscriptions“ angelegt werden.

```
BEGIN
       DBMS_CDC_SUBSCRIBE.CREATE_SUBSCRIPTION(
       change_set_name   => 'Produkte_taeglich',
       description       => 'Änderungen an Produkten',
       subscription_name => 'SUB_Produkte');
END;
```

Ähnlich wie bei Change-Set und Change-Table können der Subscription anschließend mehrere Views zugeordnet werden, mit gewünschten Attributen.

```
BEGIN
       DBMS_CDC_SUBSCRIBE.SUBSCRIBE(
       subscription_name => 'SUB_Produkte',
       source_schema     => 'SH',
       source_table      => 'PRODUCTS',
       column_list       => 'PROD_ID, PROD_NAME, PROD_LIST_PRICE',
       subscriber_view   => 'SALES_VIEW');
END;
```

Anschließend muss die Subscription noch aktiviert werden.

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

Nachdem die Einstellungen getätigt wurden, kann der Subsciber die Subscription anweisen, alle neuen Datenänderungen aus den Change Tables des Publishers in die erstellten Views zu übertragen.

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

Die beschriebene Kapselung von Publisher und Subscriber hat den Vorteil, dass mit einem Change Set mehrere Views für unterschiedliche Subscriber (typischer Weise von unterschiedlichen Anwendungen) erstellt werden können. Die Extraktionsintervalle unterschiedlicher Subscriber können also unabhängig vom Extraktionsintervall der Publishers konfiguriert werden. Weiterhin finden erwünschte Synergieeffekte statt. Da die Change Tables des Change Sets regelmäßig geleert werden müssen, können Subscriber durch die Prozedur DBMS_CDC_SUBSCRIBE_ PURGE_WINDOW mitteilen, dass sie die erwünschten Änderungen erhalten haben. Erst wenn alle Subscriber dies bestätigen, werden die Change Tables geleert. Neben dieser Entleerungsstrategie finden sich noch weitere Prozeduren und Konfigurationsmöglichkeiten, die einen flexiblen Umgang mit dieser Problematik erlauben.
Die gesamte Subscriber-Logik ist unabhängig vom eingesetzten CDC-Verfahren, sodass weitere CDC-Verfahren lediglich bis hin zur Bereitstellung der Änderungen in Change Tables des Publishers erläutert werden.

Asynchrones CDC AutoLog
Grundsätzlich geht es beim AutoLog darum, den Log-Writer-Prozess (LGWP) im Quellsystem auszunutzen, der primär für das Anfertigen von Log-Dateien zuständig ist. Dafür stehen zwei Modi, AutoLog Online und AutoLog Archive zur Verfügung. Der CDC Staging-Bereich befindet sich bei diesem Verfahren außerhalb des Quellsystems im DWH.

Im Online Modus wird der LGWP dahingehend konfiguriert, Änderungen zusätzlich an einen Remote Fileserver im CDC Staging Bereich zu propagieren. Der Remote Fileserver schreibt diese dann in die Standby-Redo-Log-Dateien. Über den sogenannten Downstream-Capture-Prozess werden die Änderungen dann ausgelesen und als LCR in die Change Tables eingepflegt.

Im Archive Modus werden die Log Dateien archiviert, bevor sie an den File Server übertragen werden. Für die Archivierung und anschließende Übertragung ist der Archiver-Prozess im Oracle Quellsystem zuständig. Für die anschließende Auswertung der vom File Server kopierten Log-Files ist dann wiederum der Downstream-Capture-Prozess zuständig.

Dass Quellsystem muss für die entsprechenden AutoLog Modi konfiguriert werden.

```
compatible = 11.0
global_names = true
java_pool_size = 50000000
log_archive_dest_1="location=/oracle/dbs mandatory reopen=5
                    valid_for=(online_logfile,primary_role)"
log_archive_dest_2 ="service=stagingdb lgwr async optional noregister reopen=5
                     valid_for=(online_logfile,primary_role)" // Online
log_archive_dest_2 = "service=stagingdb arch optional noregister reopen=5
                      template=/usr/oracle/dbs/arch_%s_%t_%r.dbf" // Archive
log_archive_dest_state_1=enable
log_archive_dest_state_2=enable
log_archive_format="arch_%s_%t_%r.dbf"
job_queue_processes = 2
parallel_max_servers = <current_value> + 5
processes = <current_value> + 7
sessions = <current value> + 2
streams_pool_size = <current_value> + 21 MB
undo_retention = 3600
```

Nachdem der LGWP in die Lage versetzt wurde, Änderungsprotokolle zusätzlich an andere Quellen zu senden, müssen diverse Einstellungen getätigt werden, damit die darin enthaltenen Änderungen in die LCR gelangen. Mit Hilfe von Abbildung 2 können wesentliche Prozesse beim AutoLog im Online und Archive Modus dargestellt werden.

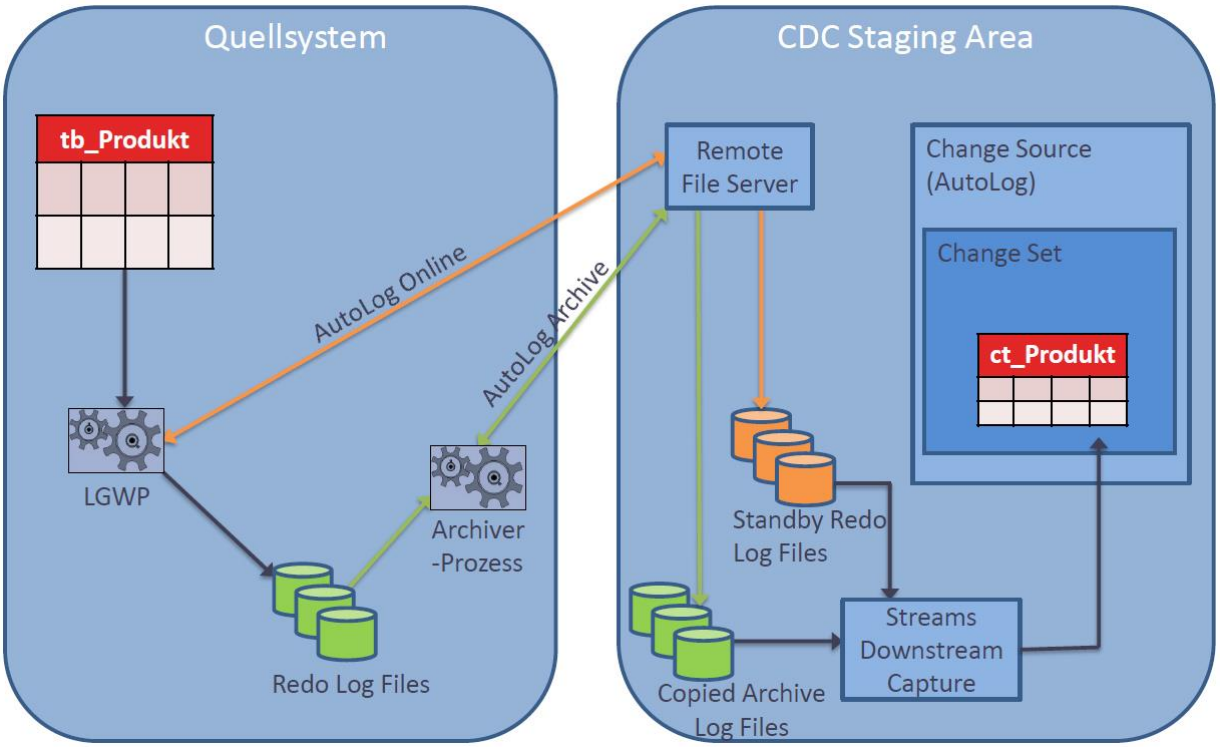

Abb. 3 - Asynchrones CDC AutoLog Online und Archive

Die beiden Modi gleichen sich in ihren Anforderungen. Quell-, sowie Zielsystem müssen identisch in Hardware-, Betriebssystem- und Oracle-Konfiguration sein, bis ins Patchlevel. Zum verwenden dieser Techniken werden Log Dateien benötigt, also muss auf gewünschten Tabellen Logging aktiviert sein. Der zusätzliche Overhead im Quellsystem fällt bei beiden Methoden gering aus. Durch die Archivierung der Log Dateien können im Archive Modus zusätzliche Latenzzeiten entstehen.

Asynchrones CDC HotLog

HotLog verbindet den Weg, die CDC Staging Area im Quellsystem zu implementieren, mit der Propagation von Deltas mithilfe von Redo Log Files. Der Streams-Local-Capture-Prozess liest die Änderungen dabei direkt aus den Online Redo Log Files des operationalen Systems und stellt diese dann aufbereitet in den Change Tables zur Verfügung.

Anhand von Abbildung 3 kann ein Quellsystem mit HotLog-Konfiguration vereinfacht dargestellt werden.

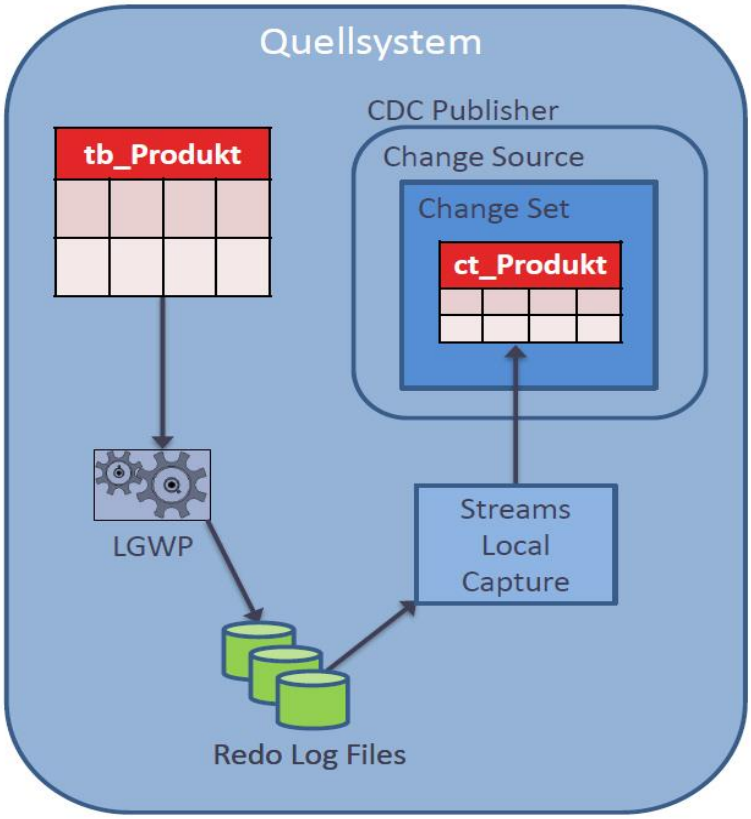

Abb. 4 - Asynchrones CDC HotLog

Durch die im Quellsystem implementierte CDC-Logik entsteht hier ähnlich wie beim Synchronen CDC ein hoher Grad an zusätzlicher Belastung. Die Belastung kann durch den Local-Capture-Prozess besser reguliert werden, da die Propagation der Änderungen nicht Teil einer Transaktion ist, sondern im Anschluss aus den Online Redo Log Files ermittelt werden. Als Vorteil kann auch hier die Unabhängigkeit von zusätzlichen Systemen und damit verbundenen Hardware-, Verwaltungs- und Lizenzkosten genannt werden.

Asynchrones CDC Distributed HotLog
Distributed HotLog basiert auf demselben System wie HotLog, mit einem bestimmten Zusatz. Nachdem Änderungen in der Change Source des Quellsystems ermittelt wurden, werden diese direkt in Change Tables eines Zweitsystems übertragen. Um Distributed HotLog verwenden zu können, müssen auf Quell,- und Zielsystem Publisher konfiguriert werden. Zur Kommunikation der beiden Publisher untereinander werden Database Links verwendet. Das Quellsystem erstellt einen Database Link zum DWH-System, mit den Login des DWH-Publishers.

```
CREATE DATABASE LINK staging_db
CONNECT TO staging_cdc_publisher IDENTIFIED BY asd123
USING 'staging_db';
```

Analog dazu erstellt der DWH-Publisher einen Database Link zum Quellsystem. Anschließend kann die benötigte Distributed HotLog Change Source, sowie die gewünschten Change Sets und Change Tables im DWH-System erstellt werden. Dabei wird als source_database der Database Link zum Quellsystem angegeben (source_db). Die CDC-Vorgänge werden auf beide Systeme aufgeteilt. Die

Change Source befindet sich im Quellsystem und die Change Sets + Change Tables befinden sich im DWH-System. Alle Konfigurationsmöglichkeiten werden vom Publisher im DWH-Bereich gesteuert.

Abbildung 4 verdeutlicht die wesentlichen Prozesse bei der Distributed HotLog Methode.

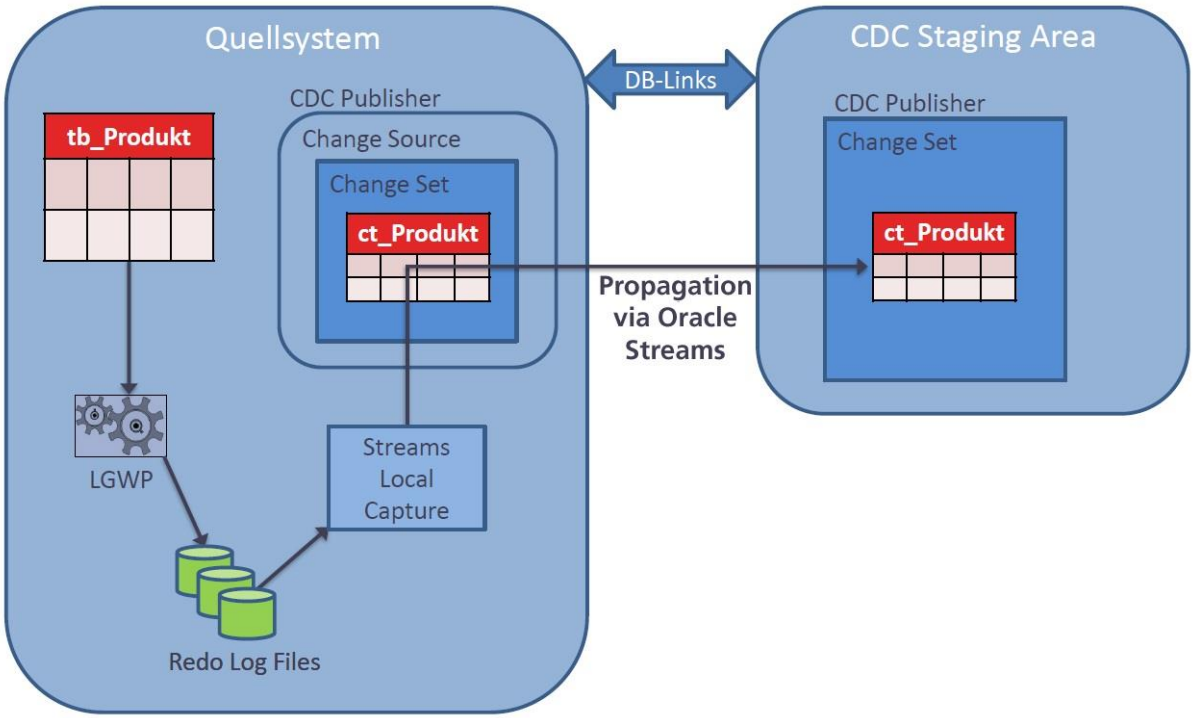

Abb. 4 - Distributed HotLog

Quellsystem und Zielsystem müssen dabei nicht zwangsweise identische in Plattform, Betriebssystem und Hardware sein. Auch die Oracle Version muss ab Oracle Release 9.2 nicht identisch sein. Dieses und die Lastverteilung können als die wesentlichen Vorteile dieses Verfahrens genannt werden. Nachteilig bleibt dabei, dass überhaupt ein zweites System inklusive Administrationsaufwände von Nöten ist.

Oracle CDC-Verfahren stellen eine gute Möglichkeit dar, die Propagation von Änderungen an die individuellen Anforderungen eines Unternehmens anzupassen. Es ist möglich, Änderungen Near-Realtime zu erhalten, oder das Quellsystem der geringsten Belastung auszusetzen. Weiterhin ist es möglich, Deltas an viele verschiedene Zielsysteme zu übermitteln. Die Publisher-Subscriber-Logik sorgt für eine Kapselung von Erkennung und Vermittlung der Änderungen und ermöglicht das performante Haushalten von Änderungsbeständen durch diverse Purge-Funktionalitäten.
Dennoch stößt Oracle CDC an bestimmten Stellen an seine Grenzen, wenn es bspw. um heterogene Quellsysteme mit unterschiedlichen Datenstrukturen geht. Zwar können einige Systeme wie MSSQL und DB2 durch spezielle CDC Adapter im CDC-Framework verwaltet werden. Aber auch das ist meist mit gewissen Einschränkungen verbunden.

Oracle Goldengate (OGG)
OGG ermöglicht Real-Time Data Integration auf hohem Level. Ähnlich zu Asynchronen CDC-Verfahren werden Änderungen anhand von Log-Dateien ausfindig gemacht. OGG kann dabei nahezu eine beliebige Anzahl an heterogenen Datenbanken untereinander synchronisieren. Dafür werden Log-

Files vom sog. Extract-Prozess gelesen und Änderungen anschließend in Trail Files (Local Trail Files) abgespeichert.

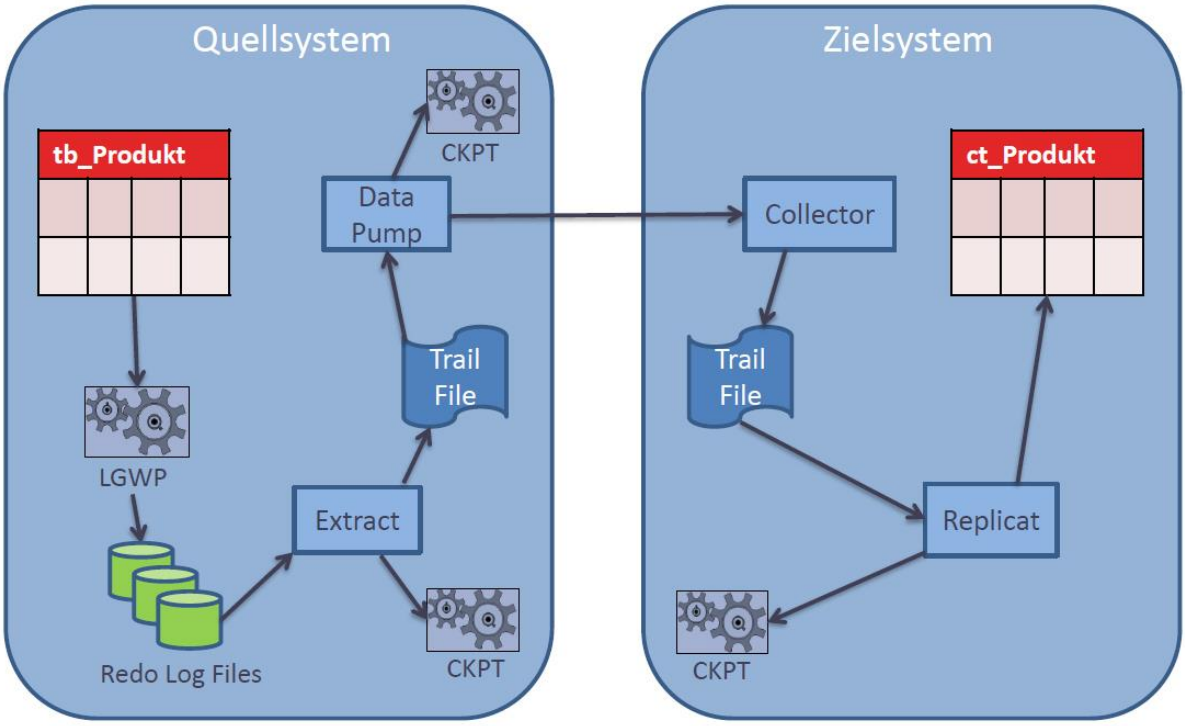

Abb. 5 - Vereinfachte Darstellung von Oracle Golden Gate (im Classic Capture Mode)

Trail Files sind in einem Oracle internen Format geschrieben und enthalten einen Header und einen Body, womit jedes Trail File eindeutig bestimmt werden kann. Jede entdeckte Änderung wird mit einer Goldengate internen Commit Sequence Number (CSN) versehen, welche aus Prüfnummern der entsprechenden Datenbanken gebildet werden. Im Falle einer Oracle Datenbank entspricht die CSN eines Datensatzes der System Change Number (SCN), die Oracle seinerseits vergeben hat. Nachdem die Änderungen in die Trail Files geschrieben wurden, können diese vom sog. Data Pump für ihre Reise zum Zielsystem vorbereitet werden. Der Data Pump verbindet mehrere Trail Files zu einem Datenpaket und kann zusätzlich diverse Transformationen und Bereinigungen durchführen, was wiederum die Weiterverarbeitung erleichtern kann. Der Collector im Zielsystem erhält anschließend die Trail Files, wertet diese aus und erstellt eigene Trail Files (Remote Trail Files), welche an den Replicat-Prozess weitergeleitet werden. Der Replicat Prozess setzt die Änderungen dann in native SQL-Befehle auf der Zieltabelle um. Während dem Extract-, Data Pump- und Replicat-Prozess werden Check Points erstellt, die den genauen Fortschritt bei der Bearbeitung der Trail Files dokumentieren. So weiß der Data Pump bei einem Netzwerkfehler genau, an welcher Stelle er die Übertragung wieder aufnehmen muss. Dadurch kann ein „No Point of Failure“-Zustand gewährleistet werden. Alle Bestandteile von Goldengate sind Modular und können einzeln angepasst werden. Neben dem beschriebenen „Classic Capture Mode“ können im „Integrated Capture Mode“ bspw. als Extract-Prozess auf Streams Prozesse zugegriffen werden und ähnlich wie im HotLog, bzw. AutoLog Modus LCR`s erstellt werden. Anschließend werden dann wieder Trail Files erstellt, sodass wieder dem grundlegenden Aufbau gefolgt werden kann.
Ein Manager-Prozess läuft auf Ziel-, sowie Quellsystem mit, überwacht die einzelnen Goldengate Prozesse und stößt im Falle eines Fehlers die nötigen Schritte an.

Nachteilig an OGG können die anfallenden Lizenzkosten werden, welche für jedes System und jeden Prozessor einzeln berechnet werden.

Fazit

Die Betrachtung der einzelnen Verfahren hat gezeigt, dass es für nahezu jeden Wunsch ein passendes Verfahren gibt. Speziell bei Altsystemen und kleineren Tabellen kann die Delta-Ermittlung über Snapshot Differentials das gewünschte Ergebnis erzielen. Wenn Audit-Columns im Quellsystem zur Verfügung stehen, kann auch deren Verwendung von Nutzen sein. Für eine Konstellation mit mehreren verteilten Zielsystemen könnten die Möglichkeiten rund um das CDC-Framework zielführend sein. Bei Heterogenen Systemen kann Goldengate die richtige Lösung sein.

Welches Change-Data-Capture-Verfahren schlussendlich das geeignete ist, hängt nicht zuletzt von den individuellen Anforderungen des entsprechenden Unternehmens ab. Die aufgeführten Methoden haben gezeigt, dass die Festlegung auf eines der Verfahren in bestimmter Weise mit Trade-Offs verbunden ist. Faktoren wie Performanceeinfluss auf Quellsysteme, Dauer der Propagation und Kosten spielen die entscheidende Rolle.

Kontaktadresse:
Dominik Schuster
areto consulting gmbh
Julius-Bau-Straße, 2
D- 51063 Köln

Telefon: +49 221 66 95 75-0
Telefax: +49 221 66 95 75-99
E-Mail Dominik.Schuster@areto-consulting.de
Internet: www.areto-consulting.de