Datenreplikation mit Oracle Streams – Grundlagen und Praxiserfahrungen

Sebastian Graf
PROMATIS software GmbH
Pforzheimer Strasse 160, 76275 Ettlingen

Schlüsselworte:
Replikation, Oracle Streams, Oracle GoldenGate, Logminer, Advanced Queueing, Capture, Propagate, Apply

Einleitung

In vielen global operierenden Unternehmen besteht die Anforderung, Daten weltweit an unterschiedlichen Standorten bereitzustellen. In einem solchen Szenario stellt sich unweigerlich die Frage, ob eine Single Instance an einem Standort in der Lage ist, die Anforderungen, insbesondere in Bezug auf Antwortzeiten, zu erfüllen. Falls dies nicht der Fall ist, müssen Daten lokal bereitgestellt werden. Dabei müssen mehrere Instanzen kontinuierlich abgeglichen werden, um einen global konsistenten Datenbestand sicherzustellen. Hier kommt Oracle Streams ins Spiel. Streams erlaubt den Aufbau einer Umgebung, in der mehrere Instanzen automatisch Transaktionen von einer Instanz in die jeweils anderen Instanzen replizieren.
Der Vortrag stellt die wesentlichen Konzepte von Oracle Streams vor und erläutert, welche Schritte zum Aufbau einer solchen Umgebung notwendig sind. Ferner wird anhand von praktischen Beispielen erläutert, wo die Grenzen von Oracle Streams sind und welche Fehler beim Aufbau einer solchen Umgebung auf gar keinen Fall gemacht werden dürfen. Abgerundet wird der Beitrag mit einem Überblick über das Produkt Oracle GoldenGate, mit dem nicht nur Oracle Datenbestände repliziert werden können, sondern eine Datenverteilung in heterogenen Umfeldern möglich ist.

Allgemeines zum Thema Replikation mit Oracle

Was ist das größte Problem einer zentralisierten Datenhaltungsarchitektur? Ganz einfach! Sie ist in einer globalisierten und verteilen Welt, wie sie die meisten Unternehmen heute vorfinden, nicht mehr zeitgemäß und schon gar nicht wirtschaftlich betreibbar. Die Entwicklung eines Produktes findet beispielsweise in Nordamerika statt. Die ersten Prototypen werden in Europa entwickelt und die Produktion findet schließlich in Asien statt. Und selbstverständlich sollen alle Standorte auf einen unternehmensweit einheitlichen Datenpool zugreifen.
Oracle hat diese Herausforderungen sehr früh erkannt und bereits mit Oracle 7 Techniken zur Datenreplikation zur Verfügung gestellt. Für die ersten Gehversuche auf diesem Terrain verwendete Oracle PL/SQL Logik und Trigger, um die Verteilung von Daten über Datenbank Links zur Verfügung zu stellen. Diese allgemein als Advanced Replication bekannt gewordene Technologie war bis Oracle 9i das Mittel der Wahl beim Aufbau einer verteilten Datenhaltung unter Oracle. Ab Version 10*g* jedoch wird von Oracle ganz klar die Technologie Oracle Streams präferiert. Die beiden wesentlichen Unterschiede zwischen Advanced Replication und Streams bestehen grob gesagt darin, dass Streams das Thema Replikation etwas weiter aus dem Zugriff des Anwendungsentwicklers entfernt und weiter in die Datenbank hinein verlagert hat, und dass Streams deutlich mehr Flexibilität beim Aufbau und Betrieb einer replizierten Umgebung bietet.

Einführung in Oracle Streams

Das Oracle Streams zugrunde liegende Verfahren, Datensätze zwischen unterschiedlichen Instanzen zu verteilen, ist eigentlich relativ einfach und folgt stets dem gleichen Grundmuster: Ein Datensatz,

der von einer Quellinstanz in eine Zielinstanz zu verteilen ist, durchläuft immer die drei Teilschritte Capture, Propagate und Apply.
Der Capture Prozess ist dabei für die Identifikation derjenigen Operationen in der Quelldatenbank verantwortlich, die in die Zieldatenbank(en) zu replizieren ist. Dabei bedient sich der Capture Prozess des Logminers, welcher die Redo Log Dateien der Quelldatenbank nach den in Frage kommenden Operationen durchsucht. Auf diese Weise können einzelnen Tabellen, ganze Schemata oder sogar die ganze Datenbank Instanz der Replikation unterworfen werden. Ferner ist es, im Unterschied zur Vorgängerversion Advanced Replication, möglich, neben DML auch DDL Kommandos zu replizieren. Das bedeutet, dass via Streams Replikation Änderungen an den Datenstrukturen von einer Instanz zur Anderen repliziert werden können, ohne dabei die Replikationsprozesse in Wartung nehmen zu müssen. Hat der Capture Prozess eine zu replizierende Operation erkannt, dann erzeugt er einen so genannten Logical Change Record (LCR), der den Zustand des betroffenen Datenbankobjekts vor und nach der entsprechenden Manipulation beschreibt, und stellt diesen in eine Queue, die zur Verteilung der LCRs zuständig ist. Neben LCRs können auch anwendungsspezifische Nachrichten, so genannte User Messages, zwischen den Systemen austauscht werden.

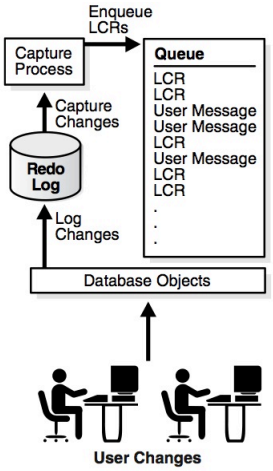

Abbildung 1: Der Capture Prozess

Der Propagate Prozess entnimmt die LCRs, die zu einer Transaktion gehören, aus der Ausgangsqueue und verteilt diese über das Netzwerk in die dafür vorgesehene Queue auf dem Zielsystem. Dieser Verteilungsmechanismus basiert in der Regel auf einem oder mehreren Datenbank-Links, die die netzwerktechnische Verbindung zwischen den beteiligten Oracle Instanzen herstellen.

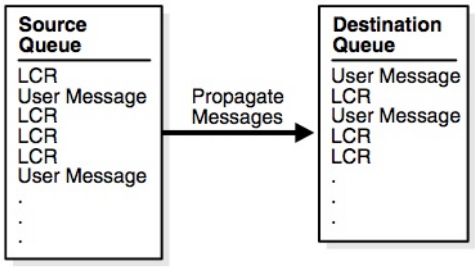

Abbildung 2: Der Propagate Prozess

Der Apply Prozess schließlich entnimmt die auf dem Zielsystem angekommenen LCRs und User Messages aus der Eingangsqueue und wendet die Änderungen im Zielsystem an. Weitere wichtige Aufgaben des Apply Prozesses bestehen zum Einen in der Anwendung von Regeln, den so genannten

Transformationen. Diese Transformationen beschreiben was zu tun ist, wenn ein LCR nicht 1:1 auf dem Zielsystem angewendet werden soll. Die zweite wichtige Aufgabe ist das Behandeln von Konflikten, die immer dann auftreten, wenn ein auf dem Zielsystem ankommender LCR nicht angewendet werden kann. Gerade dies ist für eine Streams Umgebung von größter Bedeutung, da in einer solchen Umgebung die Kommunikation der unterschiedlichen Instanzen asynchron erfolgt, man aber trotzdem sicherstellen möchte, dass auf allen an der Replikation beteiligten Instanzen stets konsistente Datenstände vorzufinden sind. Ein Umstand, der in der Folge noch im Detail zu diskutieren sein wird.

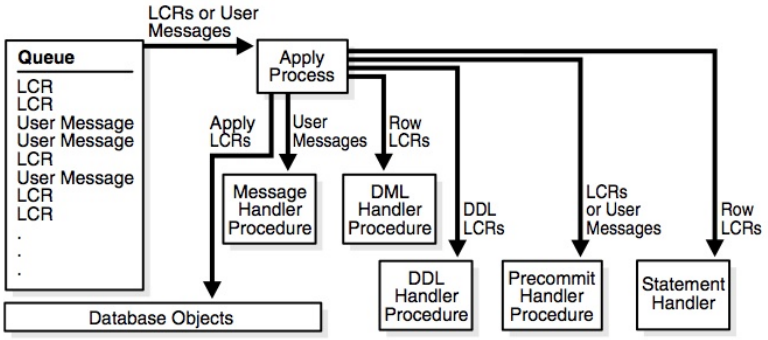

Abbildung 3: Der Apply Prozess

Bezüglich der bereits erwähnten Transformationsregeln, die auf einen LCR angewendet werden können, ist zu bemerken, dass mit diesen Regeln nicht nur die Funktionsweise des Apply Prozesses, wie oben beschrieben, beeinflusst werden kann, sondern, dass diese Regeln ganz allgemein verwendet werden können, um das Verhalten der Capture, Propagate und Apply Prozesse zu beeinflussen. Über diese so genannten Rule Sets hat der Administrator einer Streams Umgebung die Möglichkeit, die Art und Weise, wie Daten verteilt werden, nahezu völlig individuell zu gestalten.

Unterschiedliche Streams Architekturen

Mit Oracle Streams lassen sich unterschiedliche Streaming Architekturen aufbauen. Die beiden prominentesten Vertreter sind Hub-and-Spoke und N-Way Replication Architekturen.

In einer Hub-and-Spoke Architektur, auch Stern-Architektur genannt, kommuniziert eine zentrale Instanz mit mehreren anderen Spoke Instanzen direkt, wohingegen die Spoke Instanzen nicht direkt miteinander kommunizieren. Diese Architektur wird immer dann verwendet, wenn in einer zentralen Instanz alle Daten aus den Spoke Datenbanken gesammelt werden sollen, die Spoke Datenbanken aber nur einen lokal benötigten Datenausschnitt bereitstellen sollen.

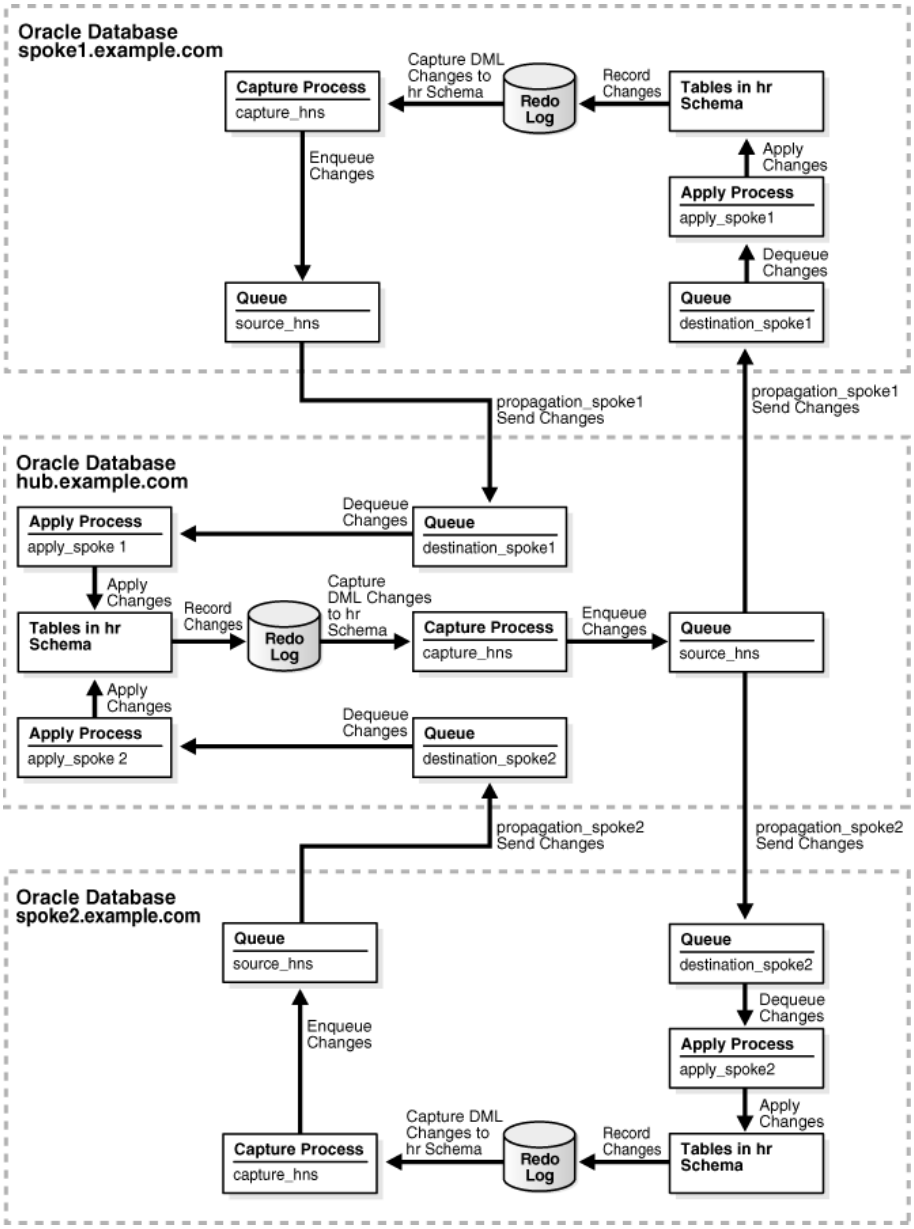

Abbildung 4: Hub-and-Spoke Architektur

In einer N-Way Architektur kommunizieren alle beteiligten Instanzen direkt miteinander. Diese Architektur kommt dann zum Einsatz, wenn an allen Standorten der identische Datenbestand vorzuhalten ist.

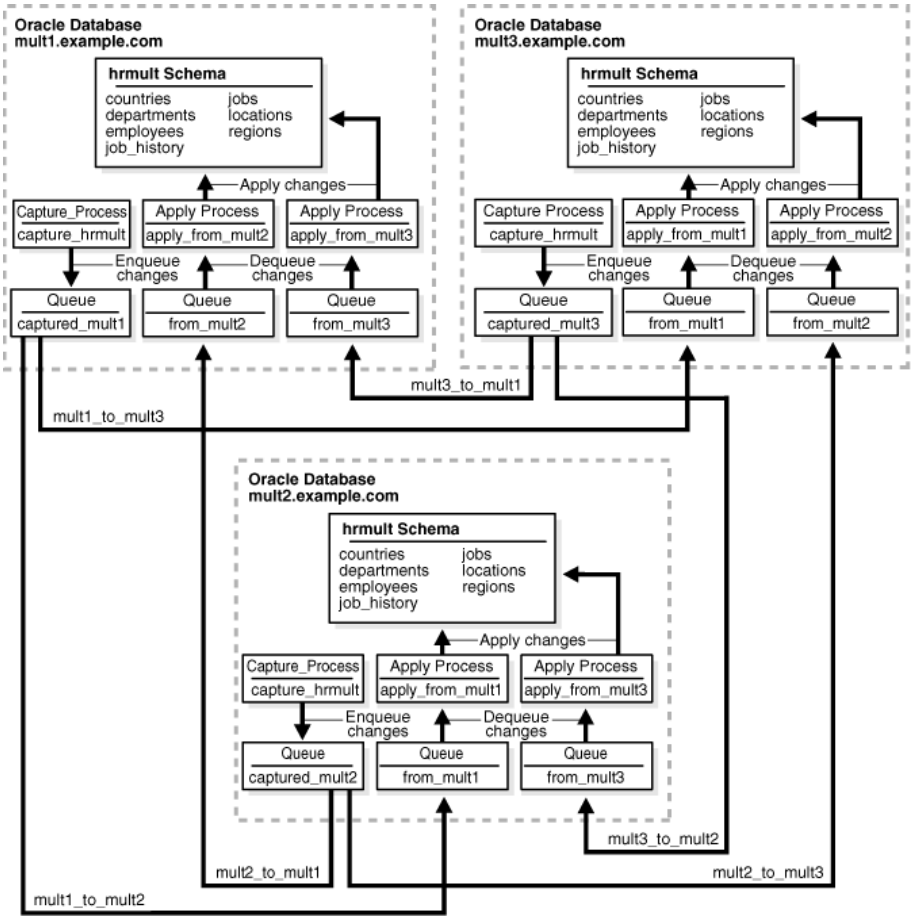

Abbildung 5: N-Way Replication Architektur

Einrichten einer Streams Umgebung

Das Einrichten einer Oracle Streams Umgebung kann prinzipiell auf zwei unterschiedlichen Wegen erfolgen: Unter Verwendung des DBMS_STREAMS_ADM Packages können entsprechende Skripten erstellt werden, die ein DBA auf den beteiligten Umgebungen ausführen kann. Das genannte Package bietet alle benötigten Funktionen zur Einrichtung einer Streams Umgebung. Zur Überwachung einer Streams Umgebung gibt es, neben dem bereits erwähnten Package, die folgenden für Streams relevanten Packages:

- DBMS_APPLY_ADM
- DBMS_CAPTURE_ADM
- DBMS_COMPARISON
- DBMS_PROPAGATION_ADM
- DBMS_RULE
- DBMS_RULE_ADM
- DBMS_STREAMS
- DBMS_STREAMS_ADVISOR_ADM
- DBMS_STREAMS_AUTH
- DBMS_STREAMS_HANDLER_ADM
- DBMS_STREAMS_MESSAGING
- DBMS_STREAMS_TABLESPACE_ADM

- UTL_SPADV

Für detailliertere Informationen sei der interessierte Leser auf die einschlägige Oracle Literatur verwiesen.

```
BEGIN
  DBMS_STREAMS_ADM.ADD_TABLE_RULES(
    table_name      => 'VETRIEB.TBL_PF_MAP',
    streams_name    => 'apply_1_2',
    streams_type    => 'apply',
    queue_name      => 'strmadmin.apply_queue_1_2',
    include_dml     => true,
    include_ddl     => true,
    source_database => 'INFOSYS1');
END;
```

Abbildung 6: Beispiel für eine Apply Regel

Das Beispiel in Abbildung 6 beschreibt die Definition einer Apply Regel für die Tabelle VERTRIEB.TBL_PF_MAP, die festlegt, dass alle in der Queue *apply_queue_1_2* ankommenden DML und DDL Operationen der Quelldatenbank *INFOSYS1* anzuwenden sind.

Ein wesentlich einfacheres Vorgehen zur Einrichtung und Wartung einer Streams Umgebung bietet jedoch der Enterprise Manager (EM). Mit dem EM kann eine Umgebung mit wenigen Handgriffen und unter Verwendung diverser Setup Wizards relativ komfortabel aufgesetzt werden. Ferner bietet der EM wertvolle Unterstützung beim Betrieb und der Überwachung einer Streams Umgebung.
Egal welche Strategie zur Einrichtung einer Streams Umgebung gewählt wird, müssen die folgenden Punkte beachtet werden:

- Alle beteiligten Datenbankinstanzen müssen im ARCHIVELOG Modus laufen.
- Alle beteiligten Datenbankinstanzen müssen über einen Streams Pool verfügen.
- Die für Streams relevanten Datenbank Parameter müssen überprüft und ggf. angepasst werden.
- Die beteiligten Datenbankinstanzen müssen netztechnisch über einen Datenbank Link verbunden sein.
- Auf allen beteiligten Datenbanken muss ein Benutzer, der sog. Streams Admin, eingerichtet und mit speziellen Berechtigungen ausgestattet sein.
- Ferner sollten die Best-Practice Vorschläge von Oracle zum Aufbau einer Streams Umgebung unbedingt beachtet werden! Diese sind in der Dokumentation *Oracle Streams Concepts and Administration* zu finden.

Lessons Learned
Obwohl der Aufbau einer replizierten Umgebung mit den oben beschriebenen Verfahren sehr einfach zu bewerkstelligen ist, und es sich bei Oracle Streams um eine ausgereifte und vielfach erprobte Technologie zur Verteilung von Daten handelt, müssen bei der Planung einer solchen Umgebung diverse potentielle Probleme berücksichtigt werden.
Der erste Stolperstein bei der Planung einer Streams Umgebung besteht in der Regel in der Nichtbeachtung der Tatsache, dass es sich bei Streams um einen asynchronen Daten-Verteilungsmechanismus handelt. Dieser Umstand ist auf jeden Fall zu beachten, wenn es darum geht die Service Level Agreements (SLAs) für die geplante Lösung zu definieren. Streams garantiert zwar den verlustfreien Transport von Daten von einer Instanz zu einer anderen, dabei kann aber nicht von

einer Synchronität ausgegangen werden. Das bedeutet, dass ein auf der Quellinstanz geänderter Datensatz auf der Zielumgebung erst mit einer gewissen Verzögerung sichtbar werden wird. Insbesondere maßgeblich für den zeitlichen Versatz ist dabei die Latenz und der Durchsatz des verwendeten Netzwerkes. Dieser Umstand wird besonders dann von zentraler Bedeutung, wenn über ein Wide Area Network (WAN) über Standortgrenzen hinweg kommuniziert wird.
Insbesondere in N-Way Umgebungen kann die Nichtbeachtung der Zeitverzögerung zu Problemen führen, die das Potential haben, einen kompletten Datenbestand in kürzester Zeit zu „vernichten“. Im folgenden Beispiel wird der Preis eines Produktes in zwei Instanzen nahezu zeitgleich verändert. Beide Änderungen werden via Capture und Propagate an die jeweils andere Instanz geschickt. Unglücklicherweise kann ein Apply auf beiden Seiten nicht durchgeführt werden, weil die beiden Instanzen auf Basis des jeweils empfangenen LCRs erkennen, dass sich der Datensatz nicht mehr im erwarteten Ausgangszustand befindet, da sich der Preis ja lokal schon geändert hat. Diese Fehlersituation führt dazu, dass, je nach Konfiguration der Umgebung, entweder die kompletten Replikationsabläufe zum Erliegen kommen, oder die Replikation weiter läuft und die aufgelaufenen Fehler in einer Fehlerqueue abgelegt werden, wo sie von einem Administrator später bewertet und korrigiert werden können.

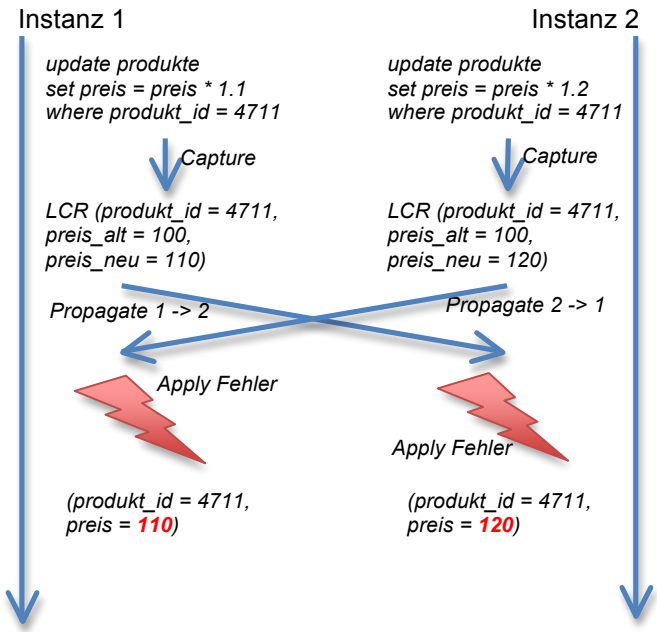

Abbildung 7: Replikationsfehler bei N-Way Replication

In unserem Beispiel ist das Produkt am Ende in beiden Instanzen mit unterschiedlichen Preisen ausgezeichnet. D.h. die beiden Datenbestände weisen Inkonsistenzen auf. Dieser Umstand erscheint auf den ersten Blick noch nicht dramatisch, weil der „Schiefstand“ vermeintlich leicht zu korrigieren ist. Dabei ist aber zu beachten, dass das beschriebene DML Statement in der Regel Bestandteil einer größeren Transaktion war, die somit in Summe auf beiden Seiten der Replikation gescheitert ist. Umso dramatischer wird der Umstand dadurch, dass eine derart gescheiterte Transaktion zu Folgefehlern bei nachfolgenden Transaktionen führen wird, die auf den eben vermeintlich geänderten Daten aufsetzen. In hochtransaktionalen Systemen kann so innerhalb kürzester Zeit ein Flächenbrand entstehen, der durchaus in der Lage ist große Teile des Datenbestandes zu vernichten. Der Lösungsansatz, diese Probleme manuell von einem Administrator über die entsprechende Oberfläche des Enterprise Managers bereinigen zu lassen, ist in der Regel ein hoffnungsloses Unterfangen.

Daher ist es in N-Way Umgebungen auf jeden Fall unerlässlich vor Inbetriebnahme der Replikation Vorkehrungen zu treffen, dass es nicht zu solchen Replikationsfehlern kommen kann. Oracle bietet hier das Verfahren der automatischen Konfliktauflösung an. Dabei kann bei der Konfiguration der Umgebung definiert werden, wie diese später im Betrieb automatisch mit derartigen Situationen umgehen soll. Die diversen Optionen reichen von der Verwendung von vordefinierten Konfliktauflösungsstrategien bis hin zur individuellen Implementierung der passenden Strategien. Die prominenteste der vordefinierten Strategien ist die Latest-Timestamp Methode. Diese Methode setzt voraus, dass in der betroffenen Tabelle eine Spalte vom Typ Timestamp vorhanden ist, in welcher der Zeitpunkt der letzten Änderung des Datensatzes vermerkt ist. Kommt es zu einem Update Konflikt und ist für die Tabelle dieses Auflösungsverfahren definiert, dann wird in beiden Instanzen der zuletzt gültige Datensatz verwendet. Soll dieses Verfahren zum Einsatz kommen und befinden sich die einzelnen Instanzen in unterschiedlichen Zeitzonen, dann sollte man sich bei der Vergabe des Zeitstempels auf eine gemeinsame Zeitzone einigen!

```
DECLARE
cols DBMS_UTILITY.NAME_ARRAY;
BEGIN
cols(1) := 'PK_RVS';
cols(2) := 'RET_CODE';
cols(3) := 'LAST_RUN';
cols(4) := 'CURR_TIMESTAMP_GMT';
DBMS_APPLY_ADM.SET_UPDATE_CONFLICT_HANDLER(
object_name       => VERTRIEB.TBL_RVS',
method_name       => 'MAXIMUM',
resolution_column => 'CURR_TIMESTAMP_GMT',
column_list       => cols);
END;
```

Abbildung 8: Beispiel für einen Update-Conflict-Handler

Abbildung 8 zeigt die Definition eines Update-Conflict-Handlers für die Tabelle *VERTRIEB.TBL_RVS*, der auf der Latest-Timestamp Methode basiert.
Weitere Konfliktauflösungsmethoden finden sich in der entsprechenden Oracle Dokumentation zum Thema Streams.

Ein weiterer zentraler Denkfehler bei der Planung einer N-Way Umgebung besteht darin, Streams als fundamentalen Bestandteil in einer Failover Lösung zu verwenden. Wobei hier Failover so zu verstehen ist, dass bei einem Ausfall der Anwendungsinfrastruktur auf Application-Server Seite an einem Standort alle betroffenen Anwender auf einen anderen Application-Server, welcher eine andere DB Instanz verwendet, umgeleitet werden. In einem solchen Szenario sind Replikationsfehler, wie oben beschrieben, quasi vorprogrammiert. Soll auf Anwendungsseite ein Failover-Mechanismus implementiert werden, dann muss das auch genau auf dieser Ebene getan werden. Die Datenbank dafür zu missbrauchen ist hier kein guter Ratgeber.

Replikation in heterogenen Umfeldern
Mit Oracle Streams stellt Oracle einen verlässlichen und effizienten Mechanismus zur Verteilung von Daten zwischen einzelnen Oracle Datenbankinstanzen zur Verfügung. Leider haben insbesondere größere Unternehmen deutlich höhere Anforderungen an die Synchronisation unterschiedlicher Datenquellen. So sind neben Oracle Datenbanken in vielen Fällen Datenbanken anderer Hersteller im Einsatz, die ebenfalls in eine Datenreplikationsarchitektur integriert werden sollen. An dieser Stelle kommt Oracle GoldenGate ins Spiel. Mit GoldenGate lassen sich Daten zwischen Oracle, DB2, SQL Server, MySQL, Sybase, Teradata und TimesTen Datenbanken austauschen. Dabei entspricht die

technische Konzeption, die GoldenGate zugrunde liegt, im Wesentlichen der von Oracle Streams: Auf dem Quellsystem werden die aktuellen Transaktionen ausgewertet und diejenigen, welche für die Verteilung relevant sind, werden gesammelt und in ein so genanntes Trail File verpackt. Ein Trail File ist ein plattformunabhängiges Dateiformat, welches von GoldenGate zum Informationsaustausch zwischen den unterschiedlichen Plattformen verwendet wird. Nach Erstellung des Trail Files wird dieses über das Netzwerk zum Zielsystem transportiert und dort in geeigneter Weise auf dem Zielsystem ausgebracht. Mit Bezug auf die Replikationsarchitekturen kennt GoldenGate 6 verschiedene Topologien:

- **Unidirektionale Verteilung:** Informationen von einer Instanz werden auf eine zweite Instanz verteilt. Diese Topologie wird häufig dann verwendet, wenn auf der zweiten Instanz Reporting Anwendungen betrieben werden, die den Betrieb auf der Quellinstanz nicht stören sollen.

- **Bidirektionale Verteilung:** Hier werden die Daten zweier Instanzen ständig in beide Richtungen abgeglichen. Die Topologie eignet sich zum Aufbau einer Failover Architektur.

- **Peer-to-Peer Verteilung:** In einer Peer-to-Peer Umgebung werden mehrere Instanzen gegeneinander abgeglichen. Peer-to-Peer wird häufig eingesetzt, wenn die transaktionale Last zwischen mehreren Instanzen aufgeteilt werden soll.

- **Broadcast Verteilung:** Bei der Broadcast Verteilung werden die Daten einer Quellinstanz auf mehrere Zielinstanzen verteilt. Broadcast stellt somit eine verallgemeinerte Variante der unidirektionalen Verteilung dar.

- **Konsolidierung:** Bei der Konsolidierung werden Daten aus mehreren Quellinstanzen in eine Zielinstanz übertragen. Die Konsolidierung ist die klassische Architektur, welche beim Aufbau einer DWH / Data Mart Lösung zum Einsatz kommt.

- **Kaskadierende Architektur:** Die kaskadierende Architektur entspricht einer Verallgemeinerung der Broadcast Verteilung (s.u.).

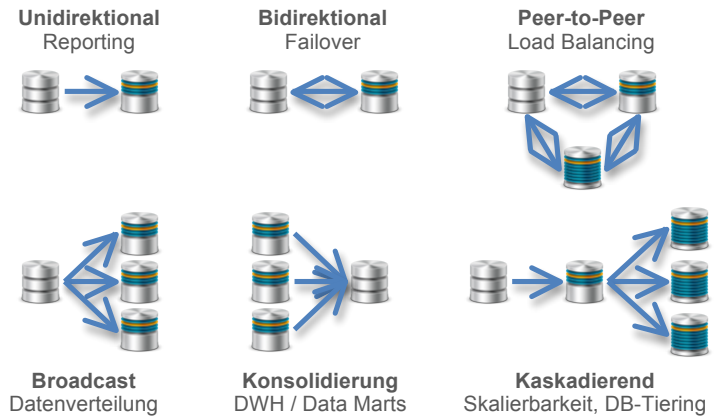

Abbildung 9: GoldenGate Replikationsarchitekturen

Zusammenfassung

Mit Oracle Streams stellt Oracle ein einfaches und doch umfangreiches, stabiles und performantes Werkzeug zur Replikation von Daten zwischen unterschiedlichen Oracle Instanzen zur Verfügung.

Die Konfiguration einer Streams Umgebung kann in wenigen Minuten mit dem Enterprise Manager vorgenommen werden, welcher auch die notwendigen Werkzeuge zur Überwachung einer Streams Umgebung bietet. Trotzdem ist der angehende Streams Anwender gut beraten, bestimmte potentiell auftauchende Probleme bereits im Vorfeld der Nutzung von Streams zu berücksichtigen und entsprechende Vorkehrungen zu treffen.
Gehen die Anforderungen der Datenreplikation über das einfache Replizieren zwischen Oracle Datenbankinstanzen hinaus, dann stellt Oracle mit dem Produkt GoldenGate eine Umgebung zur Verfügung, mit der Daten selbst in heterogenen Umgebungen ausgetauscht werden können.

Kontaktadresse:
Dipl.-Inform. Sebastian Graf
PROMATIS software GmbH
Pforzheimer Strasse 160
D-76275 Ettlingen

Telefon:	+49 (0) 7243-2179-0
Fax:	+49 (0) 7243-2179-99
E-Mail	sebastian.graf@promatis.de
	sebastian.graf@doag.org
Internet:	http://www.promatis.de
	http://www.horus.biz