mvdbs: Updatable Materialized View

Tobias Lerch, Yanick Eberle, Pascal Schwarz

27. April 2013

1 Einleitung

Ihre Lösung muss aus folgenden Teilen bestehen:

- Eine Beschreibung Ihrer Szenarios

2 Replikation einrichten

Die Master Site und die Materialized View Site wurde bereits eingerichtet, somit müssen nur noch die jeweiligen Gruppen erstellt werden.

2.1 Erstellen der Master Group

Wir verbinden uns als Benutzer repadmin auf den Telesto Server und führen folgende SQL Statements aus.

```
1 BEGIN
2         DBMS_REPCAT.CREATE_MASTER_REPGROUP (
3         gname => 'mvdbs10_repg');
4 END;
```

Das Resultat des SQL Developers ist ein einfaches „anonymer Block abgeschlossen“. Somit ist die Master Gruppe erstellt.

```
1 BEGIN
2 DBMS_REPCAT.CREATE_MASTER_REPOBJECT (
3 gname => 'mvdbs10_repg',
4 type => 'TABLE',
5 oname => 'filialen',
6 sname => 'mvdbs10',
7 use_existing_object => TRUE,
8 Copy_rows => FALSE);
9 END;
```

Die Relation Filialen ist nun ein Replikationsobjekt und wird der Master Gruppe hinzugefügt.

```
1  BEGIN
2  DBMS_REPCAT.GENERATE_REPLICATION_SUPPORT (
3  sname => 'mvdbs10',
4  oname => 'filialen',
5  type => 'TABLE',
6  min_communication => TRUE);
7  END;
```

Dieses Statement erstellt die Trigger und Packages, welche für die Replikation gebraucht werden.

```
1  BEGIN
2  DBMS_REPCAT.RESUME_MASTER_ACTIVITY (
3  gname => 'mvdbs10_repg');
4  END;
```

Die Änderungen werden in den Replikationsprozess aufgenommen.

2.2 Erstellen der Materialized View Group

Wir verbinden uns als Benutzer mvdbs10 auf den Telesto Server und führen folgende SQL Statements aus.

```
1  CREATE MATERIALIZED VIEW LOG ON mvdbs10.filialen;
```

Auf Telesto wurde nun die Materialized View erstellt und mit „materialized view LOG erstellt.“bestätigt.

```
1  CREATE DATABASE LINK telesto.janus.fhnw.ch
2  CONNECT TO proxy_refresher IDENTIFIED BY &password;
```

Der Database Link wird als Benutzer mvdbs10 auf dem Server ganymed erstellt.

```
1  BEGIN
2  DBMS_REPCAT.CREATE_MVIEW_REPGROUP (
3  gname => 'mvdbsx2_repg',
4  master => 'telesto.janus.fhnw.ch',
5  propagation_mode => 'ASYNCHRONOUS');
6  END;
```

Dieses Statement erstellt eine neue Materialized View Group.

```
1  BEGIN
2  DBMS_REFRESH.MAKE (
3  name => 'mviewadmin.mvdbs10_refg',
4  list => '',
5  next_date => SYSDATE,
6  interval => 'SYSDATE + 1/24',
7  implicit_destroy => FALSE,
8  rollback_seg => '',
9  push_deferred_rpc => TRUE,
10 refresh_after_errors => FALSE);
11 END;
```

Es wird eine Refresh Gruppe erstellt, welche einen stündlichen refresh definiert.

```
1 CREATE MATERIALIZED VIEW mvdbs10.filialen
2 REFRESH FAST WITH PRIMARY KEY FOR UPDATE
3 AS SELECT * FROM mvdbs10.filialen@telesto.janus.fhnw.ch;
```

Als Benutzer mvdbs10 auf dem Server ganymed wird die Materialized View erstellt.

```
1 BEGIN
2 DBMS_REPCAT.CREATE_MVIEW_REPOBJECT (
3 gname => 'mvdbs10_repg',
4 sname => 'mvdbs10',
5 oname => 'filialen',
6 type => 'SNAPSHOT',
7 min_communication => TRUE);
8 END;
```

Als Benutzer mviewadmin auf ganymed wird die Relation Filialen als Replikationsobjekt zu der Materialized View Group hinzugefügt.

```
1 BEGIN
2 DBMS_REFRESH.ADD(
3 name => 'mviewadmin.mvdbs10_refg',
4 list => 'mvdbs10.filialen',
5 lax => TRUE);
6 END;
```

Die Materialized View wird zur Refresh Grupppe hinzugefügt.

```
1 BEGIN
2 DBMS_REFRESH.REFRESH (
3 name => 'mviewadmin.mvdbs10_refg' );
4 end;
```

Mit diesem Statement kann der Refresh direkt ausgeführt werden.

3 Testszenarien

3.1 Ohne Konflikt

3.1.1 updates

Beschreibung:
Master Site:
—SQL-Query:
—Log:
Materialized View Site:
—SQL-Query:
—Log: