Fehlerbehandlung mittels DML Error Logging

Andreas Buckenhofer
Daimler TSS GmbH
Ulm

Schlüsselworte
DML Error Logging, DBMS_ERRLOG, LOGGING / NOLOGGING, Direct Path

Einleitung

Eine satzbasierte Verarbeitung gilt als effizienter als die Verarbeitung einzelner Datensätze. Treten Datenfehler - vor allem wenn diese nicht automatisiert korrigiert werden können - in den Eingangsdaten auf, so werden die Daten häufig einzelsatz-basiert im ETL Tool oder in PL/SQL-Schleifen gefiltert. Seit Oracle 10.2 besteht die Möglichkeit fehlerhafte Daten mittels DML Error Logging satzbasiert zu verarbeiten und in Fehlertabellen zu protokollieren.

In dem Dokument

- wird das DML Error Logging (DBMS_ERRLOG) erläutert
- werden Ergebnisse aus Performancemessungen dargestellt
- werden Erfahrungen/Möglichkeiten sowie Grenzen in DWH-Projekten aufgezeigt (z.B. Implementierung eines SQL-Codegenerators).

Anwendung DML Error Logging

Für die Aufnahme von Fehlern wird eine sog. Error-Log-Tabelle angelegt mittels

```
EXEC DBMS_ERRLOG.CREATE_ERROR_LOG('target', 'errtarget');
```

Dadurch wird eine Error-Log Tabelle angelegt, die folgende durch Oracle vordefinierte Felder aufweist:

```
SQL> DESC err$_dest
 Name                           Null?    Type
 ------------------------------ -------- --------------
 ORA_ERR_NUMBER$                         NUMBER
 ORA_ERR_MESG$                           VARCHAR2(2000)
 ORA_ERR_ROWID$                          ROWID
 ORA_ERR_OPTYP$                          VARCHAR2(2)
 ORA_ERR_TAG$                            VARCHAR2(2000)
 ID                                      VARCHAR2(4000)
 CODE                                    VARCHAR2(4000)
 DESCRIPTION                             VARCHAR2(4000)

SQL>
```

Table 20-3 Mandatory Error Description Columns

Column Name	Data Type	Description
ORA_ERR_NUMBER$	NUMBER	Oracle error number
ORA_ERR_MESG$	VARCHAR2(2000)	Oracle error message text
ORA_ERR_ROWID$	ROWID	Rowid of the row in error (for update and delete)
ORA_ERR_OPTYP$	VARCHAR2(2)	Type of operation: insert (I), update (U), delete (D) Note: Errors from the update clause and insert clause of a MERGE operation are distinguished by the U and I values.
ORA_ERR_TAG$	VARCHAR2(2000)	Value of the tag supplied by the user in the error logging clause

Damit Fehler in die Error-Log-Tabelle eingefügt werden, müssen SQL Statements erweitert werden durch den folgenden Befehlszusatz

```
LOG ERRORS [INTO [schema.]table] [('simple_expression')] [REJECT LIMIT
integer|UNLIMITED]
```

Beispiel:

```
insert into target select id  from source log errors into errtarget reject
limit unlimited;
```

Die Unterstützung der Datentypen beim Fehlerlogging ist wie folgt:

Table 20-4 Error Logging Table Column Data Types

DML Table Column Type	Error Logging Table Column Type	Notes
NUMBER	VARCHAR2(4000)	Able to log conversion errors
CHAR/VARCHAR2(n)	VARCHAR2(4000)	Logs any value without information loss
NCHAR/NVARCHAR2(n)	NVARCHAR2(4000)	Logs any value without information loss
DATE/TIMESTAMP	VARCHAR2(4000)	Logs any value without information loss. Converts to character format with the default date/time format mask
RAW	RAW(2000)	Logs any value without information loss
ROWID	UROWID	Logs any rowid type
LONG/LOB		Not supported
User-defined types		Not supported

Beispielsitzung

Eine Beispielsitzung mit und ohne DML-Errorlogging läuft wie folgt ab:

1.) Einfügen von fehlerhaften Daten ohne DML Errorlogging führt zu einer Fehlermeldung

```
INSERT INTO target
```

```
SELECT *
FROM   source;


SELECT *
       *
ERROR at line 2:
ORA-01400: cannot insert NULL into ("TEST"."target"."CODE")
```

Der Fehler führt zu einem Rollback des insert-Statements, egal wie viele Daten bereits erfolgreich eingefügt wurden.

2.) Einfügen von fehlerhaften Daten mit DML Errorlogging führt dazu, dass die korrekten Daten in die Zieltabelle eingefügt werden udn die fehlerhaften Daten in die Errorlogging-Tabelle geschrieben werden.

```
INSERT INTO target
SELECT *
FROM   source
LOG ERRORS INTO errtarget ('INSERT') REJECT LIMIT UNLIMITED;


99998 rows created.



SELECT ora_err_number$, ora_err_mesg$
FROM   errtarget
WHERE  ora_err_tag$ = 'INSERT';


ORA_ERR_NUMBER$ ORA_ERR_MESG$
--------------- ------------------------------------------------------

           1400 ORA-01400: cannot insert NULL into
("DEMO"."target"."CODE")
```

```
           1400 ORA-01400: cannot insert NULL into
("DEMO"."target"."CODE")


rows selected.
```

Performanz

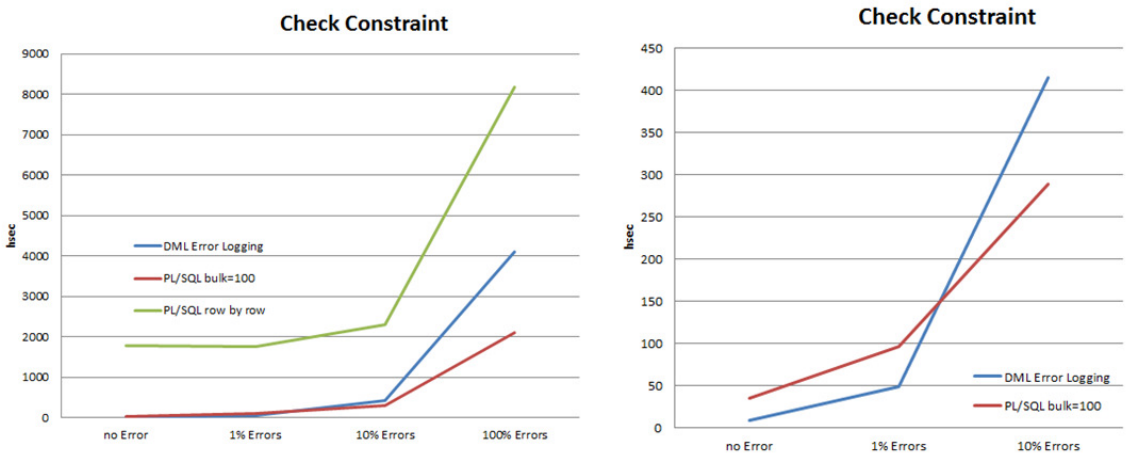
Wird eine Trace auf eine Sitzung mit DML Error Logging durchgeführt, so tauchen die erwarteten Einfüge-Operationen in die Error-Log Tabelle auf.

```
INSERT INTO "ERRTARGET" (ORA_ERR_NUMBER$, ORA_ERR_MESG$, ORA_ERR_ROWID$,
  ORA_ERR_OPTYP$, ORA_ERR_TAG$, "ID")
VALUES
 (:1, :2, :3, :4, :5, :6)
```

call	count	cpu	elapsed	disk	query	current	rows
Parse	1	0.00	0.00	0	0	0	0
Execute	9	0.00	0.00	2	4	49	9
Fetch	0	0.00	0.00	0	0	0	0
total	10	0.00	0.00	2	4	49	9

Sind die Eingangsdatensätze zu einem großen Teil fehlerhaft, d.h. es müssen viele Daten geloggt werden, so ist der Performanceverlust deutlich spürbar. Bei nur wenigen Fehlern in den Eingangsdaten halten sich Performanceverluste in Grenzen.

In dem folgenden Diagramm sind Performancemessungen für Check-Constraints enthalten. Wurden andere Fehler geloggt (z.B. unique Constraints, Datentypen) ergab sich ein ähnliches Bild.



Die x-Achse der Liniendiagramme enthält die Information, wie viele Datensätze als fehlerhaft abgewiesen werden (0% = keine Datenfehler, 1%, 10%, 100% = alle Daten fehlerhaft). Die y-Achse enthält die benötigte CPU-Zeit. Das linke Diagramm vergleicht drei Vorgehensweisen (siehe unten), das rechte Diagramm vergleicht zwei Vorgehensweisen (siehe unten) im Bereich 0%-10% Fehler.

Verglichen werden drei Vorgehensweisen:

- DML Error Logging
- PL/SQL bulk=100: schleifenbasierte Verarbeitung mit Bulk-Operationen (limit-Klausel = 100)
- PL/SQL row by row: schleifenbasierte Verarbeitung ohne Bulk-Operationen (einzelsatzbasiert)

Bei nur wenigen Datenfehlern (< ca 5%) erweist sich DML Error Logging als schneller als die beiden anderen Vorgehensweisen. Sind mehr Daten fehlerhaft, so ist eine Verarbeitung mit PL/SQL und Bulk-Operationen schneller. Eine PL/SQL-Verarbeitung ohne Bulk-Operationen ist stets deutlich langsamer und sollte auf jeden Fall vermieden werden.

Bewertung

Der große Vorteil von DML Error Logging ist die Möglichkeit, fehlerhafte Daten zu filtern mit nur geringem Codeaufwand. Muss eine Fehlerbehandlung z.B. mittels PL/SQL erstellt werden, so entstehen deutlich größere Codeblöcke.

Beim DML Error Logging muss mit Performanceverlusten gerechnet werden: je mehr Fehler geloggt werden müssen, umso spürbarer ist der Performanceverlust. Mittels direct path Load kann der Beladprozess beschleunigt werden, jedoch ist dieses Vorgehen nicht möglich, wenn erwartet wird, dass UNIQUE-Constraints verletzt werden (Einschränkung durch Oracle).

Die Anwendung von DML Error Logging hat sich vor allem bewährt

- bei der Beladung des Data Mart aus dem Core Warehouse, da die Daten im Core Warehouse bereinigt vorliegen und mit keinen oder nur wenigen Fehlern gerechnet wird.
- bei der Implementierung eines SQL-Codegenerators. Die SQL-Statements müssen „nur“ durch die logging-Klausel ergänzt werden und es muss kein weiterer Code für das Logging der Fehler implementiert werden.

Kontaktadresse:
Andreas Buckenhofer
Daimler TSS GmbH
Wilhelm-Runge-Str. 11
D-89081 Ulm

Telefon: +49 (0) 731-505 6345
E-Mail andreas.buckenhofer@daimler.com
Internet: www.daimler-tss.de