Tipps & Tricks: Bulk Binds Einführung

Bereich:	PL/SQL	Erstellung:	06/2009 EF
Versionsinfo:	9.2, 10.2, 11.1	Letzte Überarbeitung:	06/2009 EF

Bulk Binds Einführung

In vielen Applikationen müssen Tabelleninhalte in größerem Umfang ausgelesen, bearbeitet und in andere Tabellen überführt werden.
Vor Oracle 8i konnte in PL/SQL sowohl das Auslesen als auch die Verarbeitung nur über Cursor-For-Schleifen realisiert werden.

Jede Ausführung eines SQL-Statement innerhalb von PL/SQL verursacht einen sogenannten Context Switch, einen Wechsel von der PL/SQL-Engine zur SQL-Engine. Wenn die SQL-Statements innerhalb einer Schleife durchgeführt werden, kann sich dieser ständige Wechsel sehr negativ auf die Performance auswirken, weil Oracle für jede eingelesene Zeile von PL/SQL in SQL und zurück wechseln muss.

Die in Oracle 8i verfügbaren Bulk Binds verhindern diese ständigen context switches und erhöhen damit die Performance.
Damit ist es möglich geworden, viele Datensätze in einem Schritt via Bulk Collect auszulesen und per Bulk DML zu bearbeiten. Einzige Voraussetzung dafür ist, dass man mit einer Collection (associative Array, Nested Table oder Varray) arbeitet.

Bulk SELECT

Sowohl der SELECT INTO-Befehl als auch der FETCH INTO-Befehl wird nur um die Klausel BULK COLLECT erweitert.
FETCH BULK COLLECT INTO ist auch zulässig in Verbindung mit Ref Cursorn und Dynamischem SQL. Die verwendeten Collection-Variablen müssen selbst dann nicht initialisiert werden, wenn man mit Nested Tables oder VARRAYs arbeitet.

```
SELECT <spaltenliste | *>  BULK COLLECT INTO <collection[s]> FROM <tabelle>
[WHERE  ];
-- Für Cursor
FETCH <cursor> BULK COLLECT INTO <collection>
```

Bulk DML

Das Schlüsselwort FORALL leitet die Anweisung ein, die ähnlich einer FOR-Schleife eine implizit deklarierte Index-Variable, eine Unter- und Obergrenze der zu verarbeitenden Indizes und genau einen DML-Befehl beinhaltet. Trotz der vielen Ähnlichkeiten handelt es sich hierbei nicht um eine Schleife.

Die Syntax wurde seit der Einführung mit Version 8i laufend erweitert. Dokumentiert sind nur die Befehle INSERT, UPDATE und DELETE, jedoch können FORALL-Anweisungen auch geschachtelt werden, wenn man mit EXECUTE IMMEDIATE arbeitet.

Die mit 9i eingeführte optionale Klausel SAVE EXCEPTIONS ist durchaus hilfreich und sinnvoll, da sie eine Auswertung ermöglicht, für welchen Index-Wert welcher Oracle-Fehler aufgetreten ist.

Seit Version 9.2 können in PL/SQL Records in einem INSERT- oder UPDATE-Befehl als Ganzes angesprochen werden. Parallel dazu wurde bei Bulk DML die Möglichkeit eingeführt, auch Records als Datentyp der Collection anzugeben, allerdings mit der Einschränkung, dass dann - dieser neuen Syntax entsprechend nur der

komplette Record angesprochen werden darf, nicht jedoch ein einzelnes Feld daraus. Für Bulk INSERT ist diese Neuerung ein großer Gewinn, für Bulk UPDATE dagegen ist sie vor Version 11g nicht wirklich praktikabel, da für die WHERE-Bedingungen des UPDATE-Befehls eigens weitere Collections befüllt werden müssten. Ab Version 11g dürfen auch einzelne Felder eines Records angesprochen werden.

Auch die mit 10g neu eingeführte Syntax IN INDICES OF, die eine fehlerfreie Verarbeitung eines Arrays auch dann ermöglicht, wenn dieses Lücken hat, wird sich bewähren. Die Einsatzmöglichkeiten der ebenfalls neuen Variante IN VALUES OF, bei der ein zweites Array die zu verarbeitenden Indizes beinhaltet, dürften dagegen auf wenige Sonderfälle begrenzt sein.

```
FORALL I in <untergrenze>..<obergrenze> SAVE EXCEPTIONS

FORALL I in INDICES OF <collection> SAVE EXCEPTIONS
   <dml_befehl>;

FORALL I in VALUES OF <indexcollection> SAVE EXCEPTIONS
   <dml_befehl>;
```