Partitionierung im DWH mit ORACLE 11g und 12c

Reinhard Wahl
Metafinanz-Informationssysteme GmbH
München

Schlüsselworte

Partitioning, Composite, Interval, Reference, ILM, 11g, 12c

Einleitung

Die erste Partitionierung, Range-Partitioning, wurde 1997 von Oracle in der Version 8 eingeführt.

Welchen Benefit bringt Partitionierung im dimensionalen Data Warehouse?

Ziel der Partitionierung ist es zunächst mal, die zu verarbeitende Datenmenge bei der Abfrage deutlich zu reduzieren (Fachbegriff: Partition Pruning).
Außerdem kann durch PEL (Partition Exchange Load) auch der Ladevorgang beschleunigt und die Verfügbarkeit erhöht werden.

Aber neben der Performance gibt es weitere Vorteile durch Partitionierung, z.B. die Verwaltung der Daten wird vereinfacht.

Alte Daten von aktuellen Daten zu trennen, bietet auch Vorteile beim Backup-Vorgang. Sie werden nicht so oft verändert oder gar nicht mehr verändert. Das spart Platz und Zeit.

In mandantenfähigen System bringt die Partitionierung sowohl Vorteile bei der Sicherheit, als auch bei der Trennung bzw. Parallelisierung der Prozesse.

Local Index Partitioning erhöht die Verfügbarkeit, wenn ein Index Rebuild notwendig ist, denn es ist deutlich schneller nur einen Teil neu aufbauen zu müssen.

Wo wird Partitionierung generell im dimensionalen Data Warehouse eingesetzt?

Das hängt natürlich hauptsächlich von den Datenmengen und deren Struktur ab.

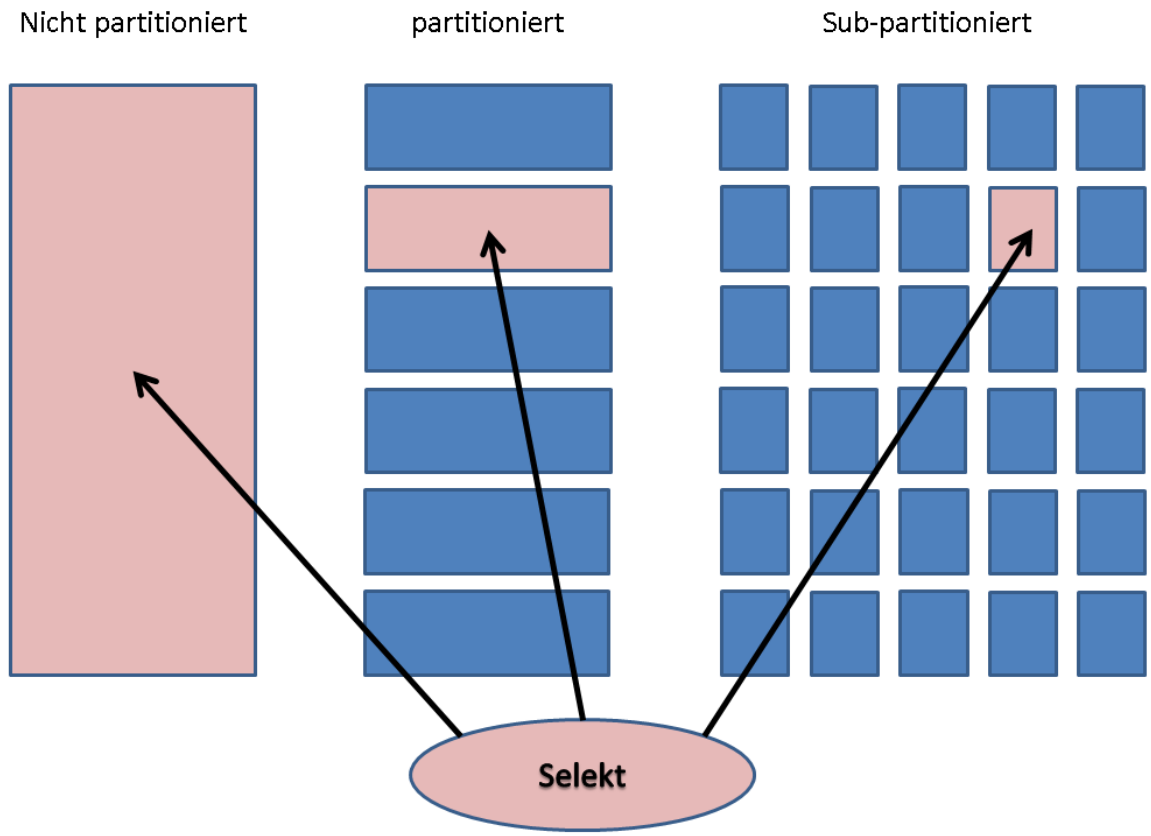
Fakten sind üblicherweise Ereignisse und werden mit einem Zeitstempel versehen. Aktuelle Daten werden häufiger abgefragt als ältere. Mit Range- bzw. Interval-Partitionierung kann man die Trennung solcher Daten gut umsetzen.

Das gleiche kann für Slowly Changing Dimensions des Type SCD2 gelten.

Zwischen Fakten und Dimensionen bestehen Beziehungen. Da kann eine Reference-Partitionierung sehr gute Performance-Gewinne erzielen.

Das Prinzip:

Zerlegt man eine Tabelle in kleinere Stücke, so müssen nur Teile beim Selekt durchsucht werden.
Die Teilung kann in zwei Ebenen geschehen: Partition und Sub-Partition.



Bisherige Methoden der Partitionierung

8.0 Range-Partitioning und Range-Partitioned Global Indexes

Was ist Range-Partitionierung?

Sie wird für Daten verwendet, die man durch ‚kleiner als' gut in Bereiche teilen kann. Das sind Zahlen, Datums-/Zeitwerte aber auch alphanumerische Daten.
Im Data Warehouse ist das Datum als Partitioning-Key sehr beliebt, weil oft z.B. monatsweise abgefragt wird. ***Slowly Changing Dimensions*** des Typs 2 (SCD II) sind hier ideale Range-Partitioning-Tabellen.

Das läßt sich nicht nur für Tabellen sondern auch für Indizes über die gesamte Tabelle durchführen. D.h. der Index kann ganz anders als die Tabelle partitioniert werden.

8i **Hash- Partitioning** und die erste **Composite-Methode Range-Hash**
Hash-Partitioning basiert auf einem Algorithmus, der auf den Parititioning Key angewendet wird. Sie ist sehr nützlich um eine Gleichverteilung der Datenmenge auf

verschiedene Devices zu erreichen und kann dadurch z.B. im ***Staging***-Bereich den ETL-Prozess parallelisieren.

9i **List- Partitioning**

List-Partitioning wird verwendet, um diskrete Werte in Partitionen zu verteilen.
Das sind zum Beispiel: Länder, Regionen, Produkte, Tarife o.ä.

In Release 2 wurde dann auch noch die **Composite-Methode Range-List** eingeführt.

Mit

10g wurde die Partitionierung der Indizes durch den **Global Hash Partitioned Index** erweitert.

Dies nur als Rückblick.

Methoden zur Partitionierung mit 11g

1. ***Interval-Partitioning***

   Wer kennt das nicht?
   ORA-14400: Eingefügter Partitionsschlüssel kann keiner Partition zugeordnet werden

   Was ist passiert?
   Es wurde eine Range-Partitioned-Table angelegt z.B. wie üblicherweise in einem Data Warehouse nach dem Datum. Es wurden auch vorsorglich schon einige Partitionen für die Zukunft angelegt. Aber leider holt einen die Zeit immer wieder ein und wenn kein Automatismus im ETL-Prozess programmiert ist, kracht es irgendwann. Aber das muss nicht sein. Abhilfe am Beispiel Range-Partitioning von oben schafft hier Interval Partitioning

   Und dann klappt‘s auch mit dem nächsten Monat.
   Kleiner Nachteil: Der Name der Partition wird vom System generiert.
   Dies lässt sich aber leicht ändern.

2. *noch* ***mehr Composite-Partitioning***
   Composite-Partitionen werden je nach Struktur der Daten und der Abfragen angewendet. Ob es sinnvoll ist, muss in jedem einzelnen Fall geprüft werden.

   I) **Range-Range**
   Dies ermöglicht logische Partitionierung über zwei Dimensionen.
   Oft gibt es Anwendungen, bei denen mehrere Zeit-Dimensionen verwendet werden z.B. *order_date* und *shipping_date*.
   Oder Banken wollen Ihre Kunden über ihren „schlechten“ Kontostand informieren. Hierfür ist die Rang-Range-Partitionierung ideal!

   II) **List-Range**
   Ist ideal für große Tabellen, die mit verschiedenen Dimensionen verbunden sind.
   Ob es besser als Range-List ist, hängt davon ab, was mehr abgefragt wird, die List- oder die Range-Argumente. Während Range-List Interval-Partitionierung erlaubt, ist

dies bei List-Range nicht möglich. List-Range macht auch bei Historisierung im Normalfall keinen Sinn.

III) **Hash-Hash**
Der Vollständigkeitshalber erwähnt: eingeführt in Release 2.
Dazu findet sich kaum brauchbare Literatur und ist wohl der letzte Versuch irgendwie eine Partitionierung verwenden zu können.

IV-V) Der Vollständigkeit halber seien noch **List-Hash** und **List-List** erwähnt.

3. ***Partitionen auf virtuelle Spalten***
   Eine sehr interessante Option sind die neu eingeführten virtuellen Spalten und die Möglichkeit, nach diesen zu partitionieren. Vor allem, wenn die vorhandenen Spalten kein vernünftiges Partitionierungskriterium bieten, kann man mit einer virtuellen Spalte eines erschaffen (z.B. Aufteilung in Regionen oder Abteilungen durch Teile/Substrings einer Identifikations- oder Bestellnummer).
   Es funktioniert wie für jede andere Spalte einschließlich Interval-Partitionierung. Die virtuelle Spalte darf allerdings keine PL/SQL-Funktion verwenden.
   Ein Beispiel mit RANGE-RANGE-Partitionierung:

4. ***REFERENCE-Partitioning***
   Die Master-Detail-Beziehung kann hier in die Partitionierung einfließen und dadurch kann ein Full partition-wise Joins sowohl seriell als auch parallel ausgeführt werden.
   Für ***Fakten*** im Data Warehouse ist das eine interessante Option.

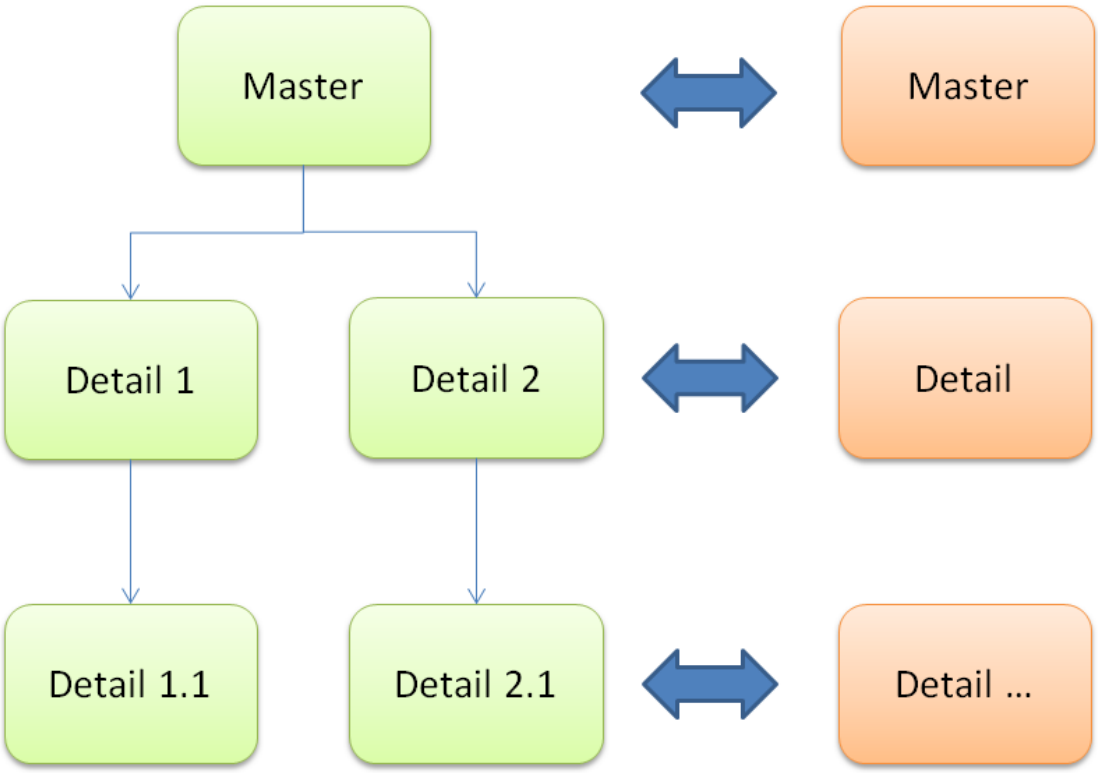
5. ***SYSTEM-Partitioning***
   Gibt es kein offensichtliches Kriterium für eine Partitionierung, kann man das auch dem System überlassen.

   Wermutstropfen: ORACLE weiß nicht, wo ein Datensatz gespeichert werden soll, da es kein Entscheidungskriterium fehlt. Dies muss durch die Applikation bei der DML sichergestellt und die gewünschte Partition angegeben werden.
   Die Frage ist: Tut man sich damit einen gefallen?
   Vermutlich ist die HASH-Partitionierung in den allermeisten Fällen die bessere Wahl.

Methoden zur Partitionierung mit 12c

I) und drei Erweiterung der REFERENCE-Partitionierung
   a. die Master-Tabelle kann auch INTERVAL-partitioniert sein
   b. die CASCADE-Option gibt es nun auch für die TRUNCATE-Operation auf die Master-Tabelle

c. die CASCADE-Option gibt es auch für Partition-EXCHANGE-Operationen, so dass die Partitionen der Detail-Tabelle an die Änderung der Master-Tabelle angepasst wird.



II) vor 12c haben einige Partitionsverwaltungs-Befehle die Partition gegen DML-Transaktionen geblockt, bevor diese zu Ende war. z.B. ALTER PARTITION MOVE. Das war für die Verfügbarkeit großer Partitionen problematisch.
Mit 12c funktioniert dies ONLINE ohne Ausfall.

Merge- und Split-Operationen konnten vor 12c nur auf 2 Partitionen zugleich durchgeführt werden. Das hieß z.B., eine Jahrespartition in 12 Monatspartition aufzuteilen benötigte man 11 verschiedene hintereinander ausgeführte Befehle. Mit 12c läßt sich dies mit einem Befehl erledigen.

Ebenso ADD, DROP und TRUNCATE PARTITION wurden erweitert, so dass mehrere Partitionen gleichzeitig bearbeitet werden könne.

III) Asynchrone Global Index-Verwaltung für DROP und TRUNCATE PARTITION

Dieses Feature erlaubt es die Anpassung eines globalen Indexes zu verschieben, so dass dieser nicht in den UNUSABLE-Status wechselt. Die Änderung wird zunächst nur auf die Metadaten des Indexes angewendet. Sie wird per DEFAULT per Job PMO_DEFERRED_GIDX_MAINT_JOB um 2 Uhr früh durchgeführt oder kann manuell mit Prozedur DBMS_PART.CLEANUP_GIDX für eine bestimmte Tabelle in einem bestimmten Schema bzw. durch ALTER INDEX REBUILD|COALESCE gestartet werden.

Verwaltung der Partitionen

Teilweise schon angesprochen, hier nochmal zusammengefasst.

1. Online Move Partition
   Ein Feature, mit dem Partitionen verschoben werden können ohne die Verfügbarkeit zu beinträchtigen und die GLOBAL INDEXES werden mit gleichzeitig angepasst - ohne REBUILD.
   Befehl: ALTER TABLE MOVE PARTITION ONLINE
2. INTERVAL Partitioning
3. Partition maintenance operations on multiple partitions
4. Asynchronous global index maintenance for DROP and TRUNCATE
5. Exchange Partition
6. … u.v.m.

Kurzer Blick auf die Möglichkeiten in ORACLE-Docs:
11g: http://docs.oracle.com/cd/E18283_01/server.112/e16541/part_admin002.htm
12c: http://docs.oracle.com/database/121/VLDBG/part_admin002.htm#VLDBG14070

Information Lifecycle Management - ILM

In der Regel werden Daten, die häufig zugreifbar sein müssen, auf einer überwiegend homogenen hoch-performen Plattform abgelegt und die weniger wichtigen, älteren Daten auf Bänder ge-stream-t. Es gibt aber im Data Warehouse noch mehr als nur schwarz-weiß, denn Daten sind

aktuell – weniger aktuell – historisch – archiviert

Manche Daten werden mehr oder weniger oft benötigt und eine Speicherung auf den schnellen Platten ist teuer und die Rückspeicherung von Band dauert lange.

Hier bietet ILM Unterstützung. Mithilfe einer Heat Map und ADO (Automatic Data Optimization) kann ORACLE Daten in einem Multiple Storage Tier verteilen, die weniger genutzten auf billigere und langsamere Platten auslagern und die häufig gebrauchten auf den schnellen.

Dies lässt sich in 4 einfachen Schritten realisieren:

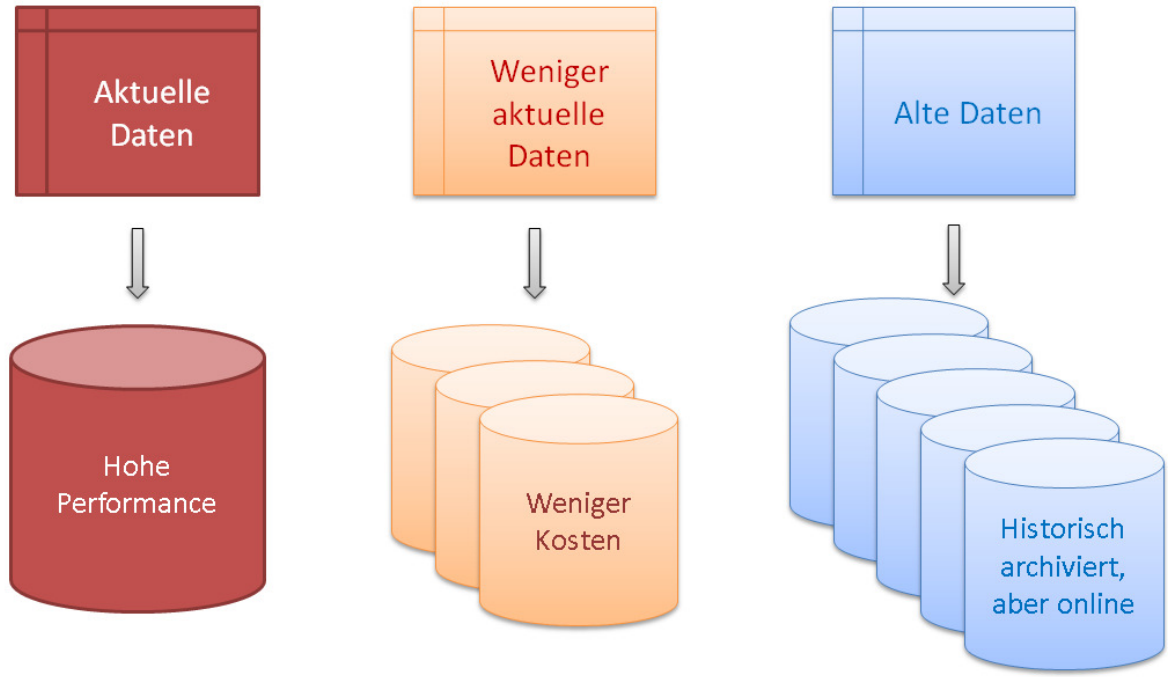
1. Definition der Daten-Klassen
2. Aufbau der Speicher-Struktur
3. Regelung Datenzugriffe und -verschiebung
4. Definition und Sicherstellung Lebensdauer, Unveränderlichkeit, Datenschutz, Überwachung, Verfallszeitpunkt

ORACLE bietet dafür den ILM Assistant an.

Er unterstützt bei der Definition und Erstellung und kann die Kostenersparnis berechnen.

Neben der Kostenersparnis ist die Verfügbarkeit eine der Hauptvorteile.

Wie lange dauern Rückspeicherungen von Bandlaufwerken und wie oft ist es schon vorgekommen, dass die Technik gar nicht mehr verfügbar war?
Die Daten in einer Datenbank zu halten macht dies sehr viel einfacher.



Zusätzlich zum „Storage Tiering“ kann auch noch „Compression Tiering“ eingesetzt werden. Unterschiedliche Komprimierungen werden verschiedenen Zugriffsverhalten gerecht. Z.B. können „kältere“ Daten höher verdichtet werden.

Fazit:

In den letzten beiden Versionen 11g und 12c der ORACLE Datenbank wurden einige sehr nützliche Hilfsmittel zur Partitionierung hinzugefügt.
Die mächtigen Features der Partitionierung können aber bei Ladevorgängen, Backup, Archivierung, Performanceoptimierung und Administration unterstützen.
Es gibt natürlich kein generelles Rezept, da die Anforderungen oft sehr individuell sind.

Vermutlich ist die Range-Partitionierung die am häufigsten verwendete Option.
Die neuen Features Reference-, Interval-, Virtual-Column- und Composite-Partitionierung haben den Werkzeugkasten enorm erweitert.

Man sollte diese Möglichkeiten austesten und für seine Bedürfnisse verwenden.
Vielleicht hat der eine oder andere an manchen Stellen bisher keine Möglichkeit der Partitionierung gefunden.

Jetzt gilt es das bestehende Data Warehouse noch mal daraufhin zu untersuchen, die neuen Optionen einbinden zu können.

Kontaktadresse:
Reinhard Wahl
Metafinanz-Informationssysteme GmbH
Leopoldstraße 146
D-00000 München

Telefon: +49 (0) 89 36 05 31 5082
Fax: +49 (0) 89 36 05 31 5015
E-Mail Reinhard.Wahl@metafinanz.de
Internet: www.metafinanz.de

Literatur:

- Oracle® Database - VLDB and Partitioning Guide 11g Release 2 (11.2) E16541-05
- An Oracle White Paper - October 2010 Partitioning with Oracle Database 11g Release 2
- docs.oracle.com
- Partitionierung für Fortgeschrittene von Frank Schneede, ORACLE Deutschland B.V. & Co. KG
- Partitionspflege mit Oracle 11g leicht gemacht von Frank Schneede, ORACLE Deutschland GmbH
- http://www.databasejournal.com/features/oracle/oracle-database-12c-new-features-for-partitioned-tables.html
- Information Lifecycle Management for Business Data An Oracle White Paper June 2007
- ORACLE DATABASE 12c: INFORMATION LIFECYCLE MANAGEMENT http://www.oracle.com/us/solutions/sap/nl23-db12c-ilm-en-2209394.pdf