Nutzung der
Oracle Database InMemory Option für SAP BW

Jörn Bartels
Oracle
München

Schlüsselworte
Oracle, SAP-BW, InMemory, Star-Schema.

Einleitung
In SAP BW wurde bisher ein erweitertes Snow Flake Schema eingesetzt, um die Daten eines Cubes zu speichern. Dies war zur Zeit der Entwicklung von SAP BW durchaus sinnvoll, da es eine Speicher Optimierte Ablage der Daten bietet. Mit den neuen Möglichkeiten der InMemory Option kann dieses Schema jedoch vereinfacht werden, da die erwünschte Speicherplatz reduzierung auch anders erreicht werden kann, bzw. nicht mehr notwendig ist.
Mit der InMemory Option kann vollständig auf alle Indizes verzichtet werden, und es findet eine starke Kompression der Daten im Hauptspeicher statt. Die Lade Performance steigt durch den Verzicht auf die Indices signifikant.

Flache Tabelle
Das normele Datenmodell für einen Datacube ist ein Star Schema. Zentral eine Fakten Tabelle, und darum ein Satz von Characteristiken. Das Modell sieht dann wie folgt aus:

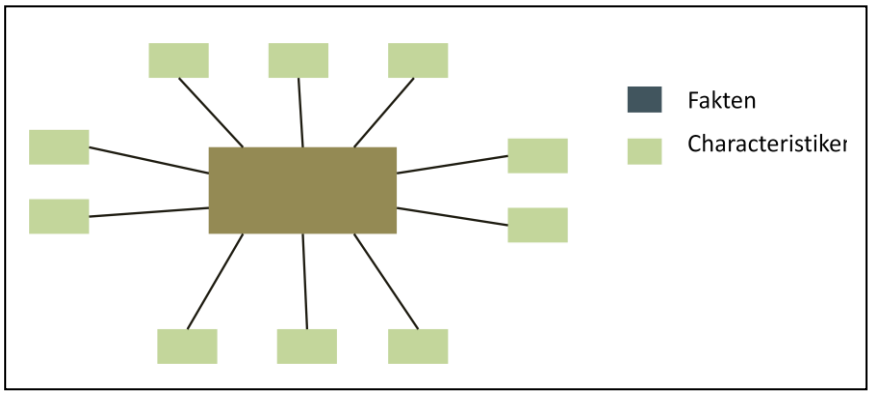

Abb. 1: Einfaches Star Schema

Dies ist immer noch ein vereinfachtes Modell des SAP-BW Schemas, da jede Chracteristik noch Attribute und Hierarchien haben kann. Tatsächlich haben wir es daher auch schon auf dieser einfachen Ebene mit einem Snow-Flake Schema zu tun. Im folgenden will ich mich ausschließlich mit der Speicherung der Faktentabelle beschäftigen. Die Faktentabelle hat prinzipiell die folgende Struktur:

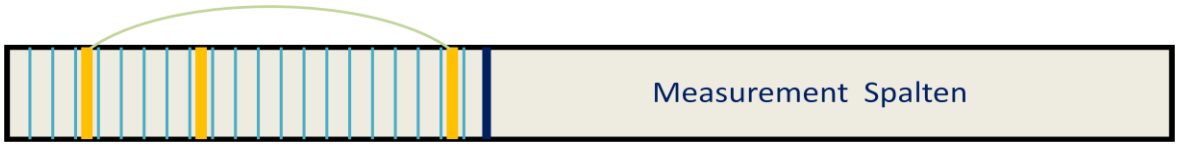

Abb. 2: Faktentabelle

Auf der linken Seite befinden sich die Schlüsselspalten, während sich auf der rechten Seite die Werte befinden. Die orangenen Felder zeigen einen möglichen Zugriff auf die Fakten Tabelle. Der Bogen stellt eine Abhängigkeit der Schlüssel von einander dar. Diese kann z. B. dadurch entstehen, dass bestimmte Artikel nur in bestimmten Ländern verkauft werden, und damit eine Unabhängigkeit der Schlüssel nicht mehr gewährleistet ist.

Die Nachteile dieser Speicherung sind die folgenden:

- Viel Platzverbrauch durch redundante Schlüsselfelder
- Langsames Laden durch die große Zahl der Indizes

Star Cube

Bei der Analyse der Schlüsselfelder kann man feststellen, das viele Gruppen der Schlüsselfelder häufig vorkommen. Durch geschickte Wahl dieser Gruppen kann man erreichen, dass in den einzelnen Gruppen möglichst wenig verschiedene Tupel auftreten.

Abb. 3: Ermittlung der Dimensionen

Diese Gruppen lassen sich dann in sogenannte Dimensions Tabellen auslagern, und zurück bleibt nur ein einzelner Schlüssel pro Dimensions Tabelle.

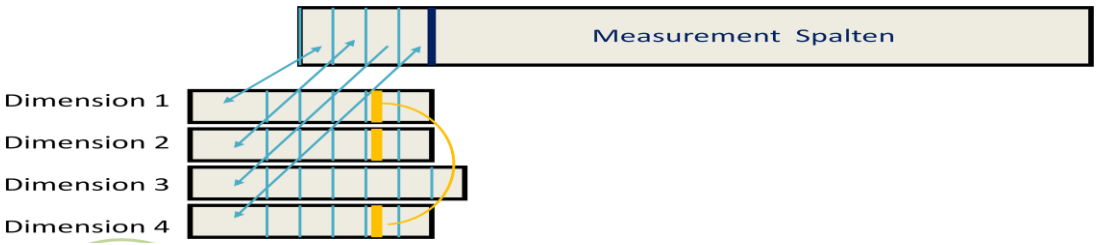

Abb. 4: Dimensions Tabellen

Die Vorteile dieser Speicherung sind:

- Platzsparend
- Weniger Indizes
- Leichter Schlüsselupdate.
- Bessere Performance durch besseres Caching

Die Nachteile sind jedoch:

- Schlechtere Performance durch zusätzliche Joins
- Sie werden leicht falsch definiert
- Schlechte Statistiken – Abhängigkeiten über mehrere Dimensionen
- Langsames Laden – Dimensions Ermittlung

Das Datenmodell für einen Star Cube sieht dann wie folgt aus:

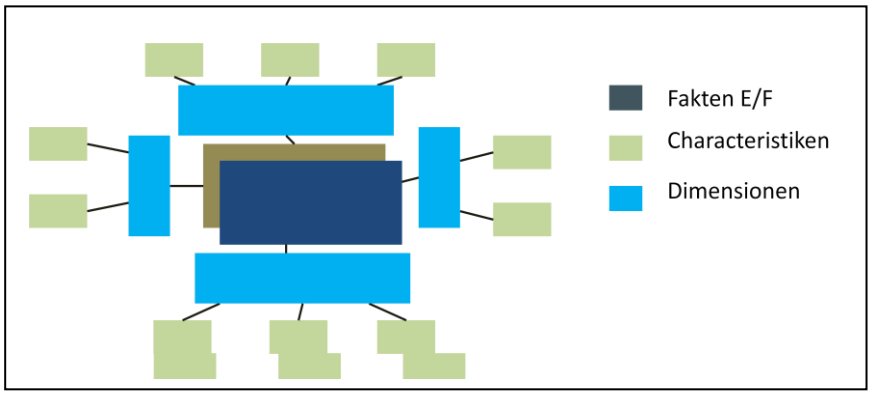

Abb. 5: Snowflake Schema

Die Fakten Tabelle wurde zusätzlich noch in zwei Tabellen aufgeteilt, um zwei unterschiedliche Partitionierungen zu realisieren.

Flat Cube

Für das neue Modell des Flat Cube kommen wir einfach zurück auf das Modell der Flat Fakten Tabelle zurück:

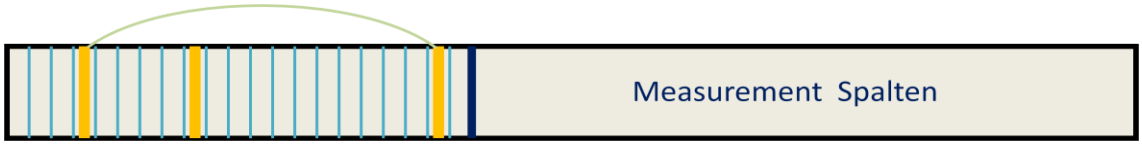

Abb. 6: Flat Faktentabelle

Aber statt die Tabelle in mehrere Dimensionstabellen auf zu spalten, wird die ganze Tabelle einfach in das Memory gelegt:

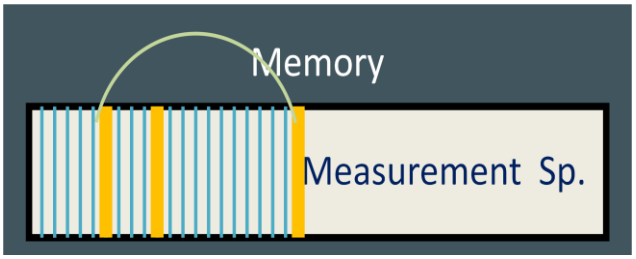

Abb. 7: Faktentabelle InMemory

Die Vorteile dieser Speicherung sind die folgenden:

- Keine Indizes notwendig
- Keine Joins für Dimensions Tabellen
- Schnelles Laden, weil es keine Indizes und Dimensionen gibt
- Komprimiert, dadurch wenig Speicher Verbrauch
- Optimierter InMemory Join (Vector Join)

Tatsächlich hat die SAP jedoch nicht alle Dimensionen abgeschafft, sodern die Paket Dimension erhalten, um eine einfache Verwaltung der einzelnen Requests, die in die Fakten Tabelle geladen werden zu ermöglichen. Diese Dimension enthält jedoch nur sehr wenig Zeilen.

Ein Vergleich des Platzverbrauchs ergab, dass die Flat Cubes im Durschschnitt mehr Speicherplatz brauchen, dies wird jedoch zum Teil dadurch kompensiert, das keinerlei Indizes mehr benötigt werden.

Hier noch eine kleine Gegenüberstellung der Zugriffspläne für einen Star Cube und einen Flat Cube:

```
-------------------------------------------------------------------------------
| Id  | Operation                                    | Name                 |
-------------------------------------------------------------------------------
|   0 | SELECT STATEMENT                             |                      |
|   1 |  SORT GROUP BY                               |                      |
|*  2 |   HASH JOIN                                  |                      |
|*  3 |    TABLE ACCESS STORAGE FULL                 | /BI0/YVC_PROD2       |
|*  4 |    HASH JOIN                                 |                      |
|   5 |     TABLE ACCESS STORAGE FULL                | /BIC/DZMK_TCSC2      |
|*  6 |     HASH JOIN                                |                      |
|   7 |      TABLE ACCESS STORAGE FULL               | /BIC/DZMK_TCSC4      |
|*  8 |      HASH JOIN                               |                      |
|   9 |       TABLE ACCESS STORAGE FULL              | /BIC/DZMK_TCSCU      |
|* 10 |       HASH JOIN                              |                      |
|  11 |        TABLE ACCESS BY INDEX ROWID BATCHED   | /BIC/DZMK_TCSCP      |
|* 12 |         INDEX RANGE SCAN                     | /BIC/DZMK_TCSCP~01   |
|* 13 |        HASH JOIN                             |                      |
|  14 |         TABLE ACCESS STORAGE FULL            | /BIC/DZMK_TCSC5      |
|* 15 |         HASH JOIN                            |                      |
|* 16 |          TABLE ACCESS STORAGE FULL           | /BIC/DZMK_TCSCT      |
|  17 |          PARTITION RANGE ITERATOR            |                      |
|* 18 |           TABLE ACCESS BY LOCAL INDEX ROWID BATCHED| /BIC/EZMK_TCSC       |
|  19 |            BITMAP CONVERSION TO ROWIDS       |                      |
|  20 |             BITMAP AND                       |                      |
|  21 |              BITMAP MERGE                    |                      |
|  22 |               BITMAP KEY ITERATION           |                      |
|  23 |                BUFFER SORT                   |                      |
|* 24 |                 TABLE ACCESS STORAGE FULL    | /BIC/DZMK_TCSCT      |
|* 25 |                BITMAP INDEX RANGE SCAN       | /BIC/EZMK_TCSC~020   |
|  26 |              BITMAP MERGE                    |                      |
|  27 |               BITMAP KEY ITERATION           |                      |
|  28 |                BUFFER SORT                   |                      |
|  29 |                 INDEX STORAGE FAST FULL SCAN | /BIC/DZMK_TCSC4~0    |
|* 30 |                BITMAP INDEX RANGE SCAN       | /BIC/EZMK_TCSC~070   |
|  31 |              BITMAP MERGE                    |                      |
|  32 |               BITMAP KEY ITERATION           |                      |
|  33 |                BUFFER SORT                   |                      |
|* 34 |                 TABLE ACCESS STORAGE FULL    | /BIC/DZMK_TCSC3      |
|* 35 |                BITMAP INDEX RANGE SCAN       | /BIC/EZMK_TCSC~060   |
-------------------------------------------------------------------------------
```

Abb. 8: Zugriffsplan Star Cube

Für den Zugriff wird eine Star Transformation mit Zugriff über Bitmap Indizes gemacht, und dafür werden drei Dimensionen ausgewählt. Die Auswahl der Dimensionen hängt sehr stark von der Qualität der Statistiken ab, und es werden im Falle der Abhängigkeit einzelner Schlüssel über Dimensionsgrenzen hinweg oft die falschen Dimensionen ausgewählt. Ein weiteres Problem können auch Schiefverteilungen der einzelnen Dimensions Schlüssel sein, da der Optimizer hier nur von einer Gleichverteilung ausgehen kann.

```
-------------------------------------------------------------------------
| Id  | Operation                                | Name                |
-------------------------------------------------------------------------
|   0 | SELECT STATEMENT                         |                     |
|   1 |  SORT GROUP BY                           |                     |
|*  2 |   HASH JOIN                              |                     |
|   3 |    JOIN FILTER CREATE                    | :BF0001             |
|   4 |     PART JOIN FILTER CREATE              | :BF0000             |
|   5 |      NESTED LOOPS                        |                     |
|   6 |       TABLE ACCESS BY INDEX ROWID BATCHED| /BIC/DZMK_TCSCP     |
|*  7 |        INDEX RANGE SCAN                  | /BIC/DZMK_TCSCP~01  |
|*  8 |       TABLE ACCESS STORAGE FULL          | /BI0/YVC_PROD2      |
|   9 |    JOIN FILTER USE                       | :BF0001             |
|  10 |     PARTITION LIST JOIN-FILTER           |                     |
|  11 |      PARTITION RANGE ITERATOR            |                     |
|* 12 |       TABLE ACCESS INMEMORY FULL         | /BIC/FZMK_TCFL      |
-------------------------------------------------------------------------
```

Abb. 9: Zugriffsplan Flat Cube

Dieser Zugriffsplan ist wesentlich einfacher und damit wesentlich schneller. Probleme mit Schiefverteilungen gibt es keine mehr, da keine Indizes genutzt werden und der InMemory Zugriff unabhängig von der Verteilung der Daten ist.

Zusammenfassung

Durch den Einsatz der Oracle Database InMemory Option ist es möglich für SAP BW die optimale Speicherungsform der Flatcubes zu wählen, die eine bessere Performance sowohl für Ladevorgänge, als auch für Abfragen ermöglichen. Es entfällt beim Laden die Pflege aller Indizes sowie die Pflege der Dimensionstabellen. Bei der Abfrage wird durch den direkten Zugriff auf den Speicher ebenfalls eine verbesserte Performance erreicht.

Für den SAP BW Administrator ergeben sich auch nur Vorteile. Er braucht keine Dimensionen mehr zu definieren, und auch die Anlage von Aggregaten entfällt vollständig.

Kontaktadresse:

Jörn Bartels
ORACLE Deutschland B.V. & Co. KG
Riesstr. 25
D-80992 München

Telefon: +49 (0) 89-1430 1120
E-Mail Joern.Bartels@oracle.com
Internet: www.oracle.com/sap