

ORACLE®

ORACLE®

Oracle DB 12c: Die In-Memory-Option

Oliver Zandner

System-Berater für Oracle-DB-Technologien

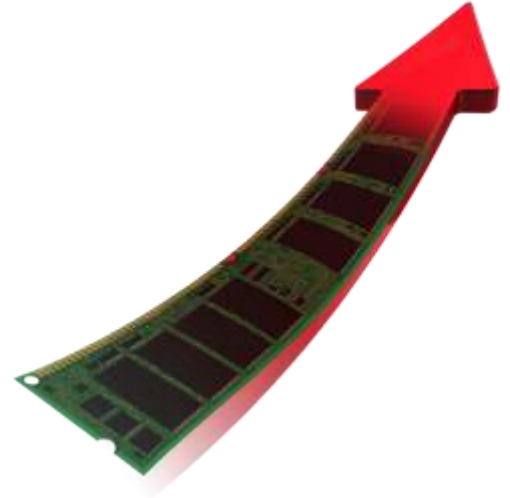
Oracle Hannover



Available July 2014

Worum geht es bei In-Memory?

- Option zur Oracle DB Enterprise Edition ab Version 12.1.0.2
- Ziel: Analytische Abfragen beschleunigen:
 - Bsp. `SUM(umsatz) FROM verkaeufe GROUP BY region;`
- DML in OLTP-DBs beschleunigen
- Ohne Änderung der Applikation



Wie lassen sich analytische Auswertungen beschleunigen?

Zeile



- **Transaktionen** laufen schneller auf Zeilen

- INSERT, UPDATE, DELETE o. SELECT
- Betrifft wenige Zeilen & viele Spalten

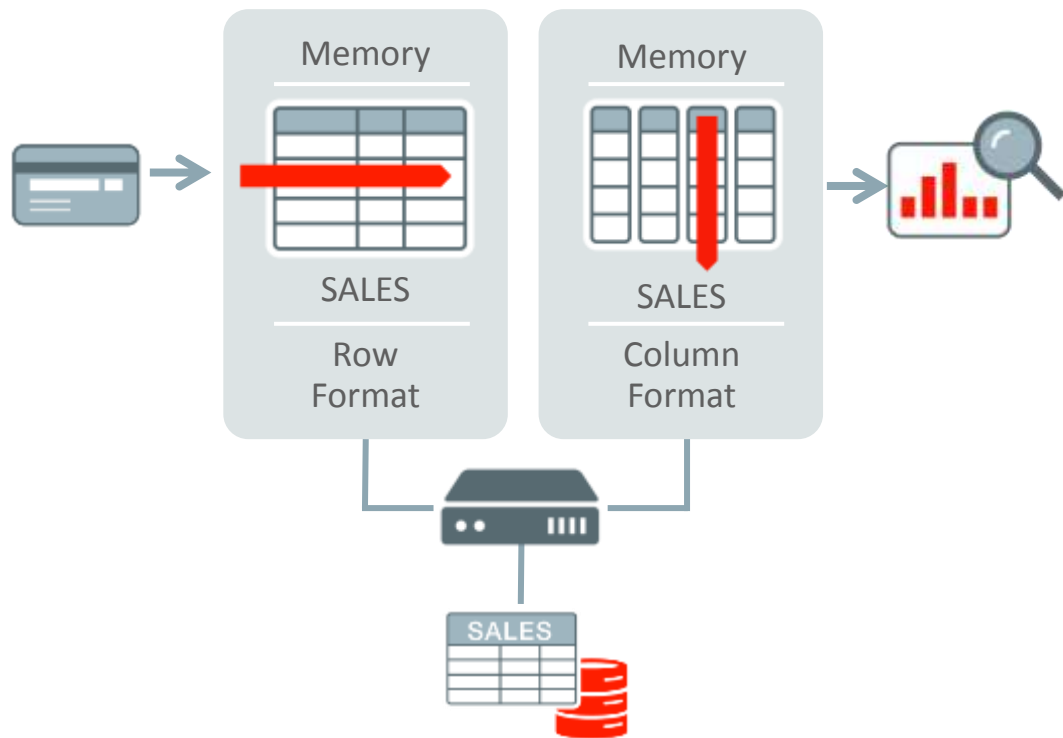
Spalte



- **Analytische Zugriffe** sind schneller auf Spalten

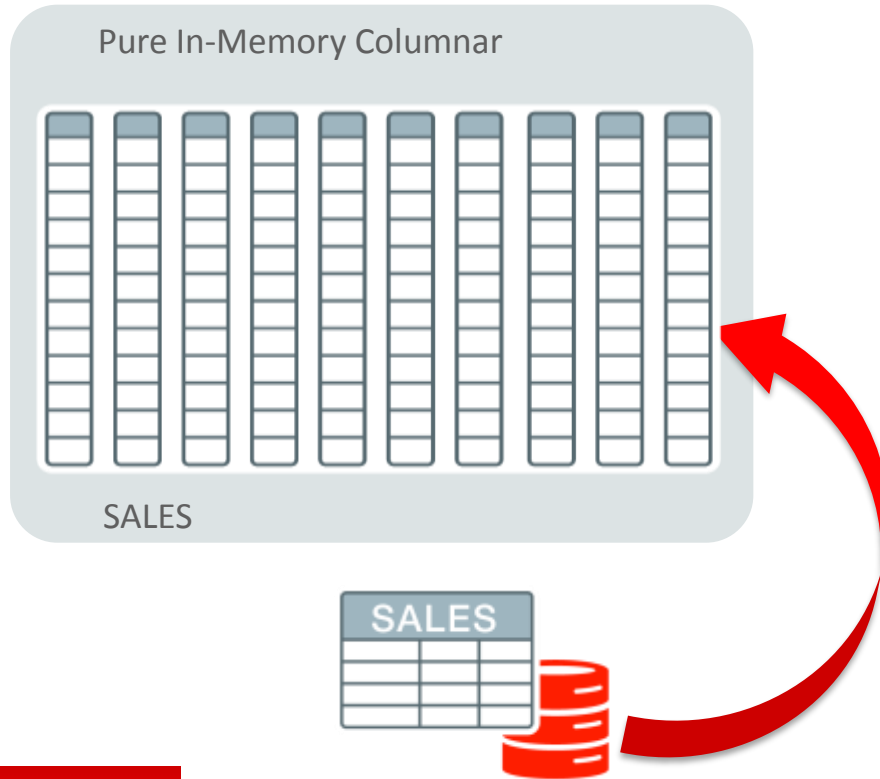
- Bsp.: Summe der Umsätze nach Regionen
- Betrifft wenige Spalten & viele Zeilen

Wie hat Oracle In-Memory implementiert? (1/2)



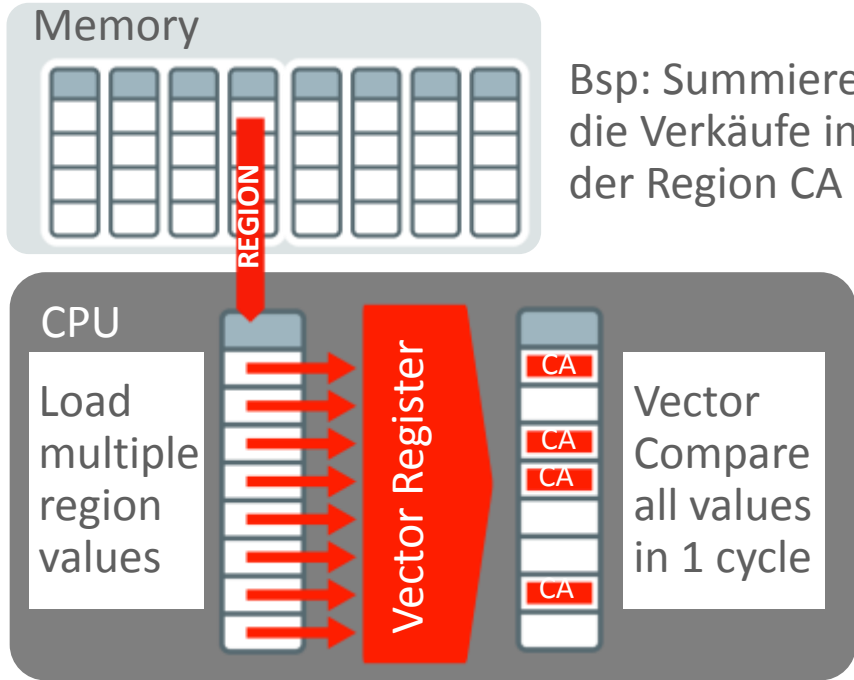
- Storage: Tabellen weiterhin zeilenweise gespeichert
- RAM: Tabellen können zusätzlich spaltenweise vorgehalten werden
- Beide Formate gleichzeitig aktiv & konsistent
- Optimizer wählt passenden Zugriffspfad

Wie hat Oracle In-Memory implementiert? (2/2)



- Spalten-Format existiert nur im RAM
- Nicht persistent, kein Logging
- DML findet auf Zeilen-Format statt
- Daten sind komprimiert (Faktor 2 - 20)
- Wird definiert für
 - Tablespaces
 - Tabellen
 - Partitionen

Was macht In-Memory so schnell?

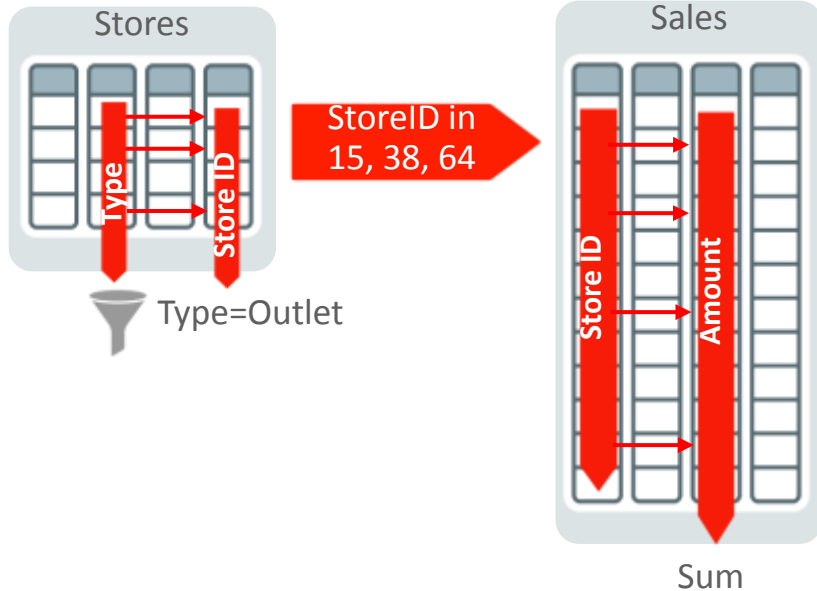


> 100x Faster

- Jeder CPU-Kern scannt einen Teil der Spalte
- Spalte wird als Vector verarbeitet (SIMD)
- Milliarden v. Werten pro Sekunde pro CPU-Kern
 - Zeilen-Format: Millionen Zeilen/Sekunde

Wie beschleunigt In-Memory Join-Operationen?

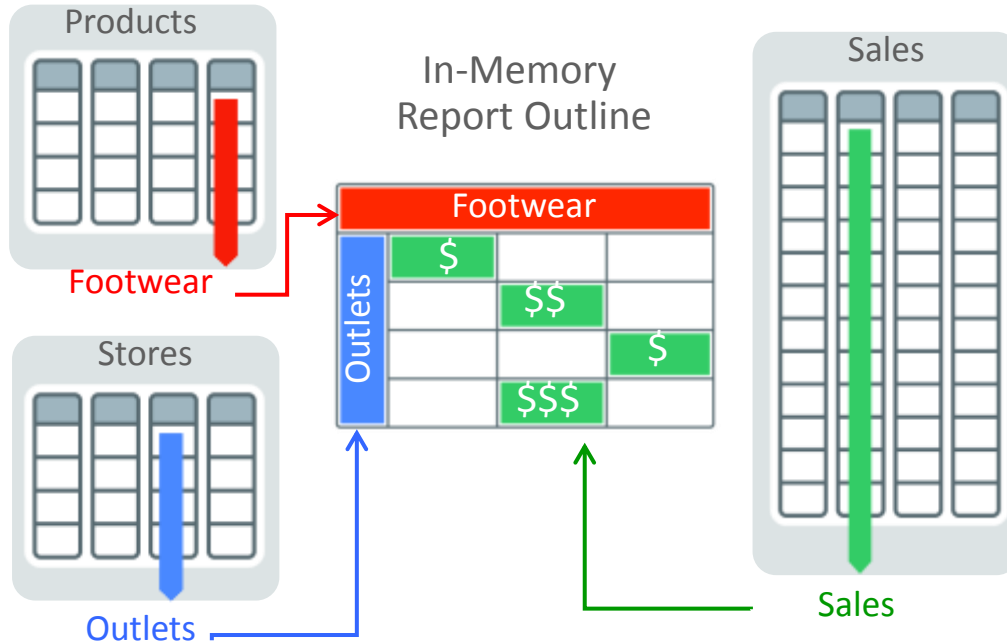
Bsp: Summiere den Umsatz in Outlet-Stores



- Filter-Bedingung der Tabelle STORES wird in Filter-Bedingung für die Tabelle SALES konvertiert
- Dadurch sind Joins **10x** schneller

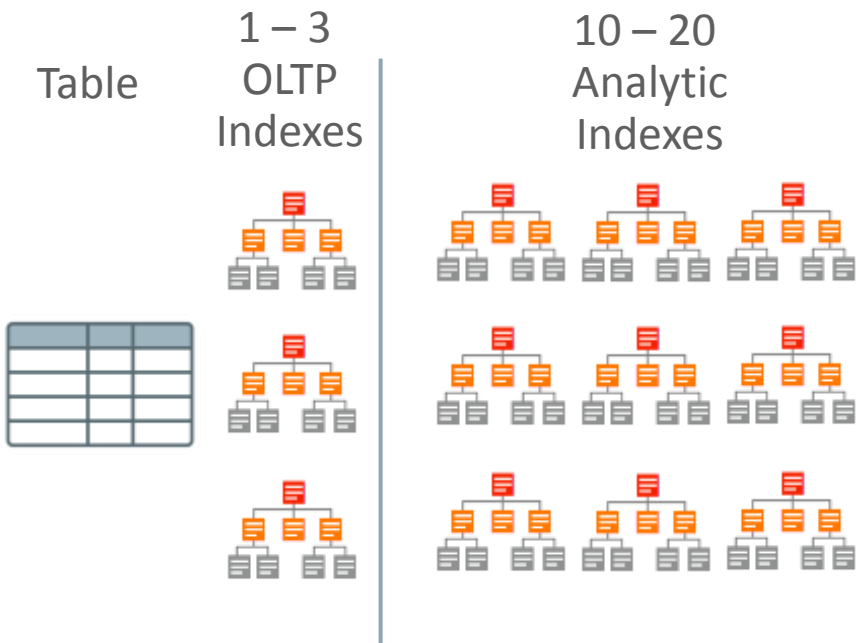
Wie beschleunigt In-Memory das Aggregieren beim Joinen?

Bsp: Summiere Umsätze mit Schuhen in Outlets gruppiert nach Standort & Waren-Kategorie



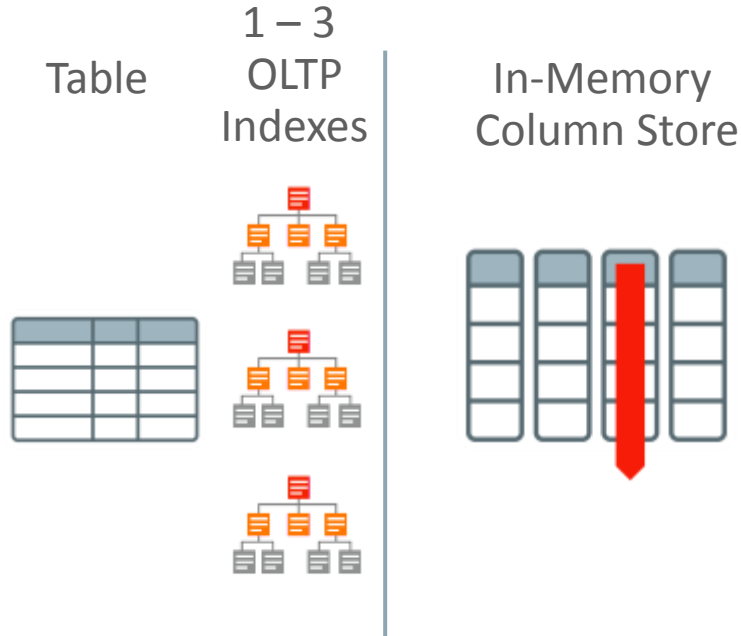
- Scan der Dimensions-Tabellen liefert distinkte Werte in der Gruppe
 - z.B. Herren-Schuhe, Frauen-Schuhe, Kinder-Schuhe, ...
- Damit wird eine temp. Vektor-Matrix aufgebaut
 - Laden 1, Herren-Schuhe, Frauen-Schuhe, Kinder-Schuhe....
- Beim Scan der Fakten-Tabellen werden die Vektoren gesucht

Wie wurden bisher analytische Reports in OLTP-DBs beschleunigt?



- Mehrheit der Indizes in OLTP-DBs wird für analytische Zugriffe benötigt
 - 1- 3 “reguläre” Indizes / 10 – 20 analytische Indizes
- Dadurch verlangsamten sich DMLs
- Indizes sind auf best. Abfrage zugeschnitten

Warum beschleunigt In-Memory auch DMLs?



- Spalten-Format existiert nur im RAM
 - Daher ist Pflege-Aufwand bei DML kleiner
 - Damit laufen OLTP & Batch schneller
- In-Memory bietet Zusatz-Nutzen
 - Besser für unbekannte Analysen
 - Weniger Aufwand b. Tuning & Admin.

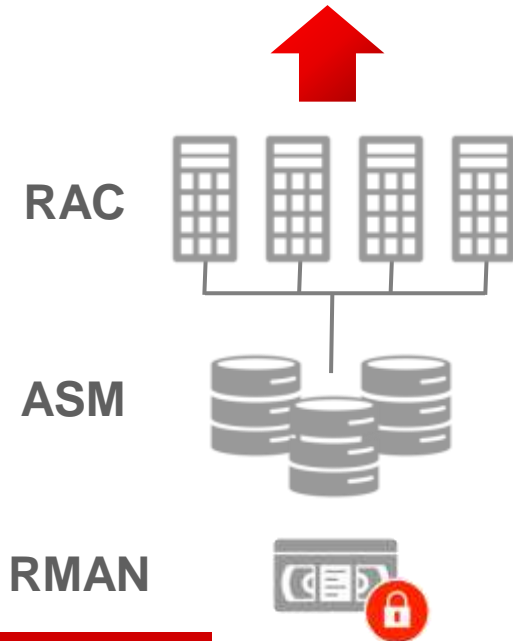
Wie lässt sich In-Memory skalieren?

- Via RAC-Option: große Tabellen lassen sich über Cluster-Knoten verteilen
- Abfragen auf große Tabellen lassen sich über RAC-Knoten verteilen
- Exadata only: Infiniband-Protokoll beschleunigt Kommunikation zwischen RAC-Knoten



Wie fügt sich In-Memory in Oracles MAA-Architektur ein?

Data Guard & GoldenGate



- Spalten-Format existiert nur im RAM
- Zeilen-Format auf Platte & Prozesse bleiben unverändert (Logging, Backup, Recovery...)
- 100% transparent für alle anderen Features & Optionen
- Fügt sich nahtlos in die Oracle-MAA-Architektur ein

Wie wird In-Memory umgesetzt?

1. Größe des Column-Store festlegen

- `inmemory_size = XXX GB`

2. In-Memory-Attribut für Tabellen / Partitionen setzen

- `alter table | partition ... inmemory;`

3. Analytische Indizes löschen, um OLTP zu beschleunigen

Wieviel Speicher benötige ich für den Column Store?

```
SQL> DECLARE
  2 l_blkcnt_cmp    PLS_INTEGER;
  3 l_blkcnt_uncmp  PLS_INTEGER;
  4 l_row_cmp       PLS_INTEGER;
  5 l_row_uncmp     PLS_INTEGER;
  6 l_cmp_ratio     PLS_INTEGER;
  7 l_comptype_str  VARCHAR2(100);
  8 BEGIN
  9 dbms_compression.get_compression_ratio (
10 scratchtbsname => 'TS_DATE',
11 ownname        => 'SSB',
12 objname        => 'LINEORDER',
13 subobjname     => NULL,
14 comptype       => dbms_compression.comp_inmemory_query_low,
15 blkcnt_cmp     => l_blkcnt_cmp,
16 blkcnt_uncmp   => l_blkcnt_uncmp,
17 row_cmp        => l_row_cmp,
18 row_uncmp      => l_row_uncmp,
19 cmp_ratio      => l_cmp_ratio,
20 comptype_str   => l_comptype_str);
21 dbms_output.put_line('compression ratio ist '|| l_cmp_ratio);
22 END;
23 /
```

compression ratio ist 2

SQL>@size.sql

NAME	ORIG_SIZE_M	IN_MEM_SIZE_M	COMP_RATIO	COMP	BYTES_NOT_POPULATED
-----	-----	-----	-----	-----	-----
LINEORDER	1216	551.5	2.20489574	FOR QUERY LOW	0

1. Objekte identifizieren
2. Compression Advisor dbms_compression verwenden
3. Wendet angegebenen Komprimierungsstufe auf ein Daten Sample an
4. Gibt Compression Ratio aus

Welche Objekte sollten in den Column Store? (Oracle In-Memory Advisor – Ausblick)

Object Type	Object	Estimated In-Memory Size	Analytics Processing Seconds	Estimated Reduced Analytics Processing Seconds	Estimated Analytics Processing Performance Improvement Factor	Benefit / Cost Ratio (Improvement Factor / In-Memory Size)
Table	SOE.LOGON	451.76MB	2114	1,887	9.3X	20.586
Table	SOE.CARD_DETAILS	607.32MB	8346	7,248	7.6X	12.514
Table	SOE.ADDRESSES	1.09GB	5237	4,621	8.5X	7.798
Partition	SOE.PRODUCT MOCKUP.Y2014Q1	812.6MB	2003	1,489	3.9X	4.799
Table	SOE.CUSTOMERS	1.10GB	108	95	8.2X	7.455
Table	SOE.ORDER_ITEMS	2.19GB	7128	6,393	9.7X	4.429
Table	SOE.ORDERS	1.34GB	3512	2,917	5.9X	4.403
Table	SOE.PRODUCT_INFORMATION	1.78MB	2873	2,205	4.3X	2.416
Partition	SOE.PRODUCT MOCKUP.Y2013Q4	1.62GB	97	1,489	3.7X	2.284
Partition	SOE.PRODUCT MOCKUP.Y2014Q2	3.37GB	642	493	4.3X	1.276

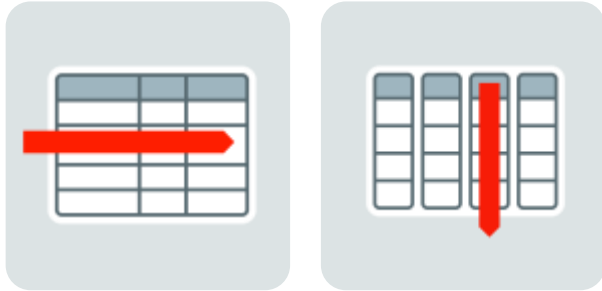
- Noch nicht verfügbar!
- Spezieller Advisor für In-Memory Abfragen
- Analysiert Datenbank Workload über AWR & ASH
- Liefert Objekte die von In-Memory Column Store profitieren würden

Welche Änderung ist in der Anwendung erforderlich?

- **SQL bleibt unverändert**
- **DB-Design & Daten bleiben unverändert**
- **Applikation bleibt unverändert**
- **Kompatibel mit anderen Features/Optionen**

In-Memory ist 100% transparent für die Applikation

Für welche Art v. Anwendung ist In-Memory optimal?



In Memory ist ideal für Anwendungen die,

- Analytische Auswertungen in der DB via SQL implementieren
- Daten via SQL mengen-orientiert verarbeiten (anstatt Einzel-Sätze)
- Parallele SQL-Verarbeitung nutzen

Fazit – Was bringt Ihnen die In-Memory-Option?

1. Neuer Bereich in der SGA (In-Memory-Column-Store)
2. Beschleunigt analytisch Zugriffe
3. Macht analytische Indizes überflüssig - beschleunigt OLTP
4. Anwendung & DB bleiben unverändert
5. Liefert Antworten aus Ihrer OLTP-DB in Echtzeit

ORACLE®

DATABASE IN-MEMORY



ORACLE®

ORACLE®

Hardware and Software Engineered to Work Together

ORACLE®