Oracle In-Memory-Option: 'Game Changer' für Data Warehousing und Business Intelligence

Dr.-Ing. Holger Friedrich
sumIT AG

Schlüsselworte
In-Memory-Option, Columnar Database, Performance, Data Warehouse, Business Intelligence

Einleitung

Mit Release 12.1.2 fügte Oracle ihrer relationalen Datenbank einen spaltenorientierten in-memory Datenspeicher hinzu. Dieses neue Feature erbringt eine massive Beschleunigung von analytischen Datenbankabfragen. Noch dazu, ist der spaltenorientierte Datenspeicher transparent integriert. Nicht eine einzige Zeile Applikationsprogrammcode muss geändert werden, um ihn zu nutzen und von seiner Leistung zu profitieren.
Dieses Paper stellt die Grundzüge der Oracle-In-Memory-Option vor. Die Nutzung des spaltenorientierten Datenspeichers auf den Gebieten des Data Warehousing und der Business Intelligence wird geprüft und bewertet. Typische Anwendungsszenarien werden gezeigt, die signifikant beschleunigt werden können. Dies hilft 'tief hängende Früchte' zu identifizieren für die sich der Einsatz der In-Memory-Option lohnt. Es werden aber auch typische Arbeitsschritte und Abfrage-Konfigurationen aufgezeigt, die weniger davon profitieren.

Columnar Stores

Dass Oracle seine neue Datenbankoption ‚In-Memory-Option' benennt, ist zwar ein guter Marketingeinfall, aber auch ungenau oder sogar ein wenig irreführend. Schliesslich geschieht die gesamte Datenverarbeitung der Datenbank im Hauptspeicher, also ‚in memory'. Zudem ist mit dem Shared Buffer im SGA bereits seit Anbeginn der Oracle Datenbank ein In-Memory-Datenspeicher vorhanden. So hätte die Option ‚Opracle Columnar Store' genannt werden können, was dann aber den neuen Teil der Verarbeitungsfunktionalität unterschlagen hätte, der speziell auf den Columnar Store zugeschnitten ist.
Oracle ist mitnichten der Erfinder des Gebiets der spaltenorientierten In-Memory-Datenbanken (Columnar Store + Verarbeitungslogik). Das Konzept ist bereits Jahrzehnte alt. Abbildung 1 zeigt einige der Hauptmitbewerber, die zum Teil bereits seit vielen Jahren am Markt um Kunden werden.

Abb. 1: Auswahl einiger Mitbewerber Oracles auf dem Markt der spaltenorientierten Datenbanken

Oracles Variante
Oracle ist also nicht der erste Bewerber auf dem Markt der spaltenorientierten Datenbanken bzw. Datenbankoptionen. Aber, ganz so wie Apple es oftmals macht, hat Oracle das Thema wohldurchdacht und kommt nun mit einer Architektur auf den Markt die grosse Vorteile gegenüber den Lösungen anderer Anbieter hat. Abbildung 2 zeigt die High-level-Architektur, welche Oracle dabei verfolgt.

Abb. 2: Die Einbettung des Columnar Store der In-memory-Option in Oracles RDBMS

Insbesondere kennzeichnet sich die Architektur durch folgende Merkmale aus.

- Der Columnar Store wird transparent neben der traditionellen zeilenorientierten Darstellung, als Teil der Shared Global Area im Hauptspeicher gehalten.
- Der Nutzer muss sich nicht exklusiv für eine der beiden Darstellungsformen entscheiden.
- Bei DML-Operationen, die weiterhin auf der zeilenorientierten Repräsentation ausgeführt werden, wird der Columnar Store, mittels eines Journal-Mechanismus, transaktional korrekt zeitgleich aktualisiert.
- Die persistente Speicherung von Information erfolgt unverändert in zeilenorientierten Datenblöcken auf Festplatten und/oder Flash-Storage. Auf Speichermedien, die dafür eingerichtet sind (z.B.Exa-Storage), kann auch eine spaltenorientierte Speicherung mittels Hybrid Columnar Compression erfolgen, dies ist aber unabhängig von der In-Memory-Option.
- Das gesamte Ökosystem und alle anderen Optionen der Oracle-Datenbank funktionieren ohne Einschränkungen und unverändert zusammen mit der Nutzung der In-Memory-Option. Dies beinhaltet Security, RMAN, Data Guard, Real Application Cluster und viele weitere.

Technologische Leckerbissen
Die In-Memory-Option besteht, wie geschrieben aus dem Columnar Data Store und der Verarbeitungsfunktionalität, die auf diesem im Zusammenspiel mit den anderen Speichermechanismen der Oracle Datenbank (SGA, PGA, Flash, Festplatten) arbeitet. Damit die Verarbeitungsmechanismen der Datenbank den Columnar Store so effizient wie möglich nutzen wurden eine Reihe von neuen Funktionen implementiert und Technologien verwendet. Herausragende Beispiele sind im Folgenden aufgelistet..

1. In-memory Storage-Indexing, analog Exadata-Storage auf den Datenstrukturen des Columnar Store.
2. Binäre Datenkompression im Columnar Store, konfigurierbar analog Hybrid Columnar Compression.
3. Feingranulare Konfiguration der spaltenrientiert zur Verfügung zu stellenden Inhalte bis hinunter zu Spalten und (Sub-)Partitionen von Objekten.
4. Transparente Abfragen über die ganze Speicherhierarchie.
5. Transaktionale Konsistenz des Columnar-Store in Echtzeit bei DML-Operationen.
6. Parallele Abfrageverarbeitung.
7. SIMD Vektorverarbeitung.

8. In-memory Fehlertoleranz und Beschleunigung auf Real Application Clustern.
9. On-demand-Aufbau multidimensionaler Aggregationsstrukturen.

Beurteilung der Leistungsfähigkeit
Spaltenorientierte Datenbanken zeichnen sich durch sehr schnelle Beantwortung von Abfragen auf wenigen Spalten aus, welche wenig weitere Verarbeitung wie Joins, Sortierung oder Aggregation erfordern. Dies liegt daran, dass das schnelle Daten-Scanning für solche Abfragen durch die gemeinsame Speicherung der Werte einer Spalte unterstützt wird. Hinzu kommet, dass sich die Daten im schnellen Hauptspeicher befinden, so sie in den Columnar Data Store hineingeladen wurden. Schliesslich beschleunigt Oracles spezielle Funktionalität zur Aggregation etc. (siehe oben) die weitere Verarbeitung der Daten nach dem eigentlichen Scanning-Schritt ebenfalls.

Aber es handelt sich letztendlich ‘nur' um ein Stück fortschrittlicher Technologie, welches einen bestimmten Zweck erfüllt. Dahinter steckt weder Magie, noch ist die In-Memory-Option eine eierlegende Wollmilchsau, die wie durch Zauberei alle Anwendungsszenarien in einer relationalen Datenbank um Grössenordnungen beschleunigt. Logik und Messungen zeigen, dass

- die In-Memory-Option bestimmte Queries extrem beschleunigt,
- insbesondere das Scanning von Daten auf eingeschränkter Spaltenanzahl profitiert,
- Joins durch Bloom-Filtering und Star-Aggregationen durch *Vector-By*-Verarbeitung beschleunigt werden,
- aber sortieren, klassisches aggregieren und weitere Funktionen nur begrenzt schneller werden. Dies ist logisch, da diese Operationen dann wie gehabt im PGA mit den altbekannten Algorithmen und Datenstrukturen verarbeitet werden.

Abbildung 3 gibt einen qualitativen Eindruck davon, dass Scanning-Operationen sehr stark und andere Operationen nur bedingt beschleunigt werden. Insgesamt führt dies dann zu einer moderaten bis sehr extremen Beschleunigung der Verarbeitung, je nach Art der Datenbankanfragen

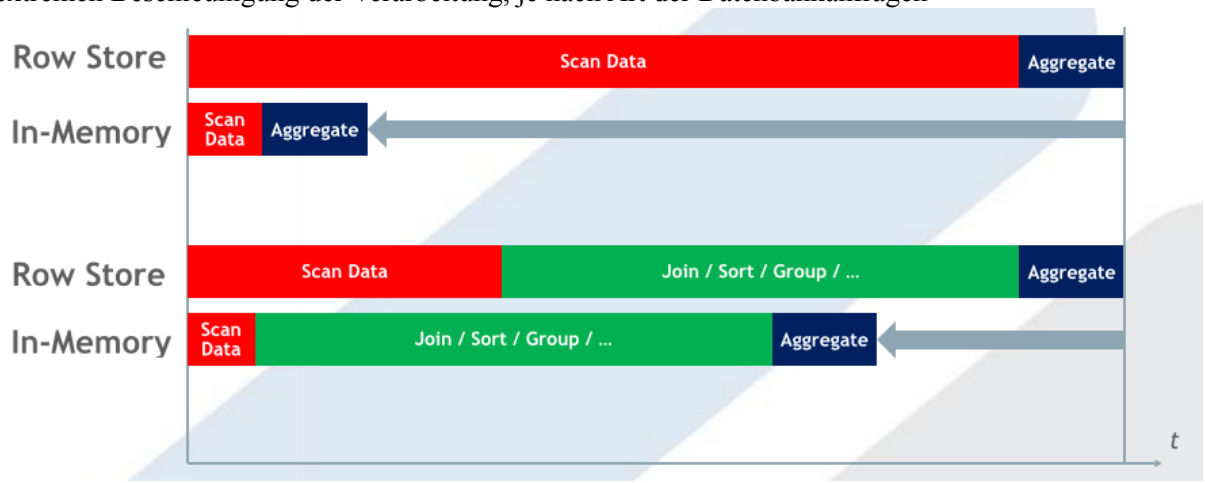

Abb. 3: Qualitative Darstellung der Beschleunigung einfacher & komplexer Abfragen mittels In-Memory-Option

Data Warehousing und Business Intelligence
Was heisst nun die Verfügbarkeit der In-Memory-Option und die daraus resultierende extrem beschleunigte Verarbeitung vieler Abfragen für den Bereich Data Warehousing und Business Intelligence? Es heisst, dass auch hier manche Abfragen extreme Beschleunigung erfahren, andere wiederum nicht. Zudem heisst es, dass manche Dinge anders, nämlich mit weniger Aufwand implementiert und betrieben werden können, sodass die gleiche oder sogar wesentlich bessere Performance für den Endkunden mit geringeren Kosten und Fehlerwahrscheinlichkeit bereitgestellt werden kann.

Analytical Queries
Business Intelligence, inklusive klassischem Berichtswesen umfassen verschiedene Arten von Abfragen, wie sie in folgender Übersicht zusammengetragen sind.

- **Einfache Listen/Berichte**. Sie sind charakterisiert durch wenige Joins, eine begrenzte Zahl von Spalten und allenfalls einfache, eindimensionale Aggregation.
  Solche Abfragen erfahren eine sehr hohe Beschleunigung (siehe Abbildung 3 oben).
- **Komplexe Berichte**. Diese beinhalten komplexe Joins, Aggregationen und die Berechnung analytischer Funktionen, welche Fensterfunktionen, Sortierung etc beinhalten.
  Solche Abfragen werden in ihrem Scanning-Anteil stark beschleunigt, im Berechnungsanteil aber weniger. Es bleibt ein signifikanter Performance-Gewinn. Er ist aber nicht so überwältigend, wie bei einfachen Berichten (siehe Abbildung 3 unten).
- **Werkzeugbasiertes OLAP**. BI-Werkzeuge, wie zum Beispiel Oracle Business Intelligence (OBIEE) erzeugen sehr komplexe Queries. Oft werden WITH-Clauses aneinandergereiht und rekursiv genutzt. Zudem werden massiv analytische Funktionen zur Berechnung komplexer Kennzahlen eingesetzt.
  Der Performance-Gewinn wird für solche komplexe Berechnungen kleiner, da Materialisierung von Zwischenresultaten bei der Verkettung von WITH-Clauses in temporären, zeilenorientierten Tabellen die spaltenorientierte Verarbeitung durchbricht und der Sortierungs- und Aggregationsaufwand der analytischen Anweisungen anfällt.
  Die Notwendigkeit zur Vorberechnung und Materialisierung von Aggregaten für komplexe Kennzahlen bleibt daher bestehen. Einfache Reports auf Dashboards werden allerdings extrem beschleunigt (siehe oben). Konsequenterweise kann man auf die Vorberechnung von Aggregaten auf einfache Kennzahlen zumeist verzichten.
- **Dimensional Queries**. Abfragen in der Form klassischer Star-Queries werden durch die neue *Vector-By*-Transformation des Optimizers und die darauf zugeschnittene Verarbeitung extrem beschleunigt. Dies heisst, dass man bei der Nutzung der In-memory-Option für diese Art Abfragen oftmals auf komplexe, Performance fressende und aufwendig zu wartende Konstrukte wie Bitmap-Indizes auf Faktentabellen und Materialized Views für Aggregate verzichten kann.

Data Warehouse Lade-Queries
Während der Fokus der Diskussion bei spaltenorientierten Datenbanken zumeist auf Business-Intelligence-Abfragen oder der Beschleunigung des operativen Reportings für OLTP-Applikationen liegt, wird oftmals vergessen, dass es im Bereich Data Warehousing weitere Aufgaben gibt, die grosse Systemlast erzeugen und zumeist unter grossem Zeitdruck stattfinden. Es sind dies Aktivitäten, die sich rund um das laden des Data Warehouse abspielen. Auch einige dieser Aufgabenbereiche können durchaus vom Einsatz der In-Memory-Option profitieren.
Die folgenden Bereiche und Charakteristik ihrer Abfragen sind hier von Interesse.

- **Datenqualitätsmanagement**. Beim Datenqualitätsmanagement werden typischer Weise Formate der Werte einzelner Spalten oder auch Wertabhängigkeiten verschiedener Spalten untersucht.
  Hier ist naturgemäss viel Scanning im Spiel. Diese Queries können bei Einsatz der In-Memory-Option, wie zuvor gezeigt, massiv beschleunigt werden. Wenn die durchzuführenden Tests jedoch sehr komplex sein sollten, zum Beispiel komplexe Formattests mit `REGEXP_LIKE`, nimmt der Grad der Beschleunigung wieder ab.
- **Metadaten-Transformation**. Beim laden in ein Data-Warehouse werden oftmals Wertebereiche von Attributen gleicher Bedeutung, aber verschiedener Quellsysteme, auf einheitliche Wertedomänen abgebildet. Wenn zum Beispiel das männliche Geschlecht in Quellsystem X mit dem Wert ‚0' codiert ist, der quellsystemunabhängige Standard im DWH jedoch männlich mit dem Zeichen ‚M' kodiert, muss diese Transformation während des Ladeprozesses geleistet

werden.
Dies ist ein typischer Fall von Scanning mit einfachem Join gegen eine Mapping-Tabelle ohne Aggregation. Solche Abfragen können durch die In-Memory-Option stark beschleunigt werden.

- **Schlüsseltransformationen**.Ein weiterer Aspekt beim laden eines DWH, der oben beschriebener Metadatentransformation ähnlich ist, ist die Transformation von Quellsystemschlüsseln in Referenzen in DWH-eigene Surrogate Keys. Den Vorgang gibt es beim Reporting auch wieder in entgegengesetzter Richtung, wenn in Berichten oder Analysen zur besseren Verständlichkeit Quellsystemschlüssel ausgewiesen werden sollen. Im Gegensatz zur Metadatentransformation werden beim laden jedoch oftmals in einem SQL-Query mehrere Referenzen durch Outer Joins an mehrere Tabellen transformiert.
  Neben dem schnellen Scanning im Columnar Store ist hier insbesondere der Einsatz von Bloom-Filtern zur Durchführung der Joins ein Vorteil. Bloom-Filter können als erster Schritt eines Joins auf dem Columnar Store direkt eingesetzt werden und dort das schnelle Scanning zur Filterung nutzen. So kann der Vorgang der Schlüsseltransformation durch den Einsatz der In-Memory-Option signifikant beschleunigt werden.

Fazit

Der Einsatz der Oracle In-Memory-Option ist fürwahr ein Game Changer für den Bereich des Data Warehousing und der Business Intelligence. Die Art wie Oracle die Herausforderung technisch gelöst hat ist, im Gegensatz zu Lösungen von Nischenanbietern, absolut enterprise-ready. Zudem erlaubt sie die nahtlose Weiterverwendung bestehender Infrastrukturen, Hochverfügbarkeitslösungen und anderer Datenbank-Optionen. Im Gegensatz zu den Lösungen anderer grosser Player des Datenbankmarktes sind bei der Oracle-Lösung keine Neuinvestition in Infrastruktur und insbesondere keine Änderungen an bestehenden Applikationsimplementierungen notwendig. Möchte man an bestehenden Applikationen etwas ändern, dann lediglich, um noch mehr Leistung zu gewinnen, nicht weil es für den Betrieb notwendig wäre.
Die In-Memory-Option beschleunigt insbesondere Abfragen mit hohem Scanning-Anteil. Auch Joins werden beschleunigt, so diese seitens der Datenbank mit Bloom-Filtering ausgeführt werden können. Andere Operationen, wie Sortierung und Aggregation profitieren nicht in gleichem Masse. Eine Ausnahme sind dabei jedoch dimensionale Queries, die durch die Anwendung der *Vector-Group-By*-Logik extreme Beschleunigung erfahren können. Sowohl im Umfeld des DWH ladens, als auch des Berichtswesens und der Business Intelligence finden sich sehr viele Anwendungsfälle und Abfragen, die entsprechend ihrer Struktur und Komplexität vom Einsatz der In-Memory-Option profitieren.

Kontaktadresse:

Dr.-Ing. Holger Friedrich
sumIT AG
Täfernstrasse 28
CH-5405 Baden-Dättwill

Telefon:	+41 (0) 56 – 470 2500
Fax:	+41 (0) 79 – 320 8179
E-Mail	holger.friedrich@sumit.ch
Internet:	www.sumit.ch