IBM Db2 BLU Technologie für den Mittelstand. Ein kostenoptimierter Ansatz.

Jörg Kremer, Head of Consulting

mip Management Informationspartner GmbH

1 Db2 BLU – Einordnung und technische Grundlagen

Business-Intelligence-(BI)-Lösungen dienen der effizienten Analyse großer Mengen von Unternehmensdaten, beispielsweise zur Berechnung und Darstellung von KPIs (Key Performance Indicators). Sie sind heutzutage wichtiger Bestandteil der Management-Werkzeuge für Unternehmen unterschiedlichster Größe. Eingesetzt wurden diese Systeme zunächst in größeren Unternehmen, die typischerweise auch größere Investitionen zur Implementierung einer BI-Lösung tätigen konnten. Durch den wachsenden Konkurrenzdruck kommt jedoch kaum ein mittelständisches Unternehmen ohne gute Indikatoren beziehungsweise Kennzahlen aus. Auch hier sind die Datenmengen stark gewachsenen. BI-Systeme sind typischerweise abfrageorientiert und müssen mit großen Datenmengen umgehen können.

Im Laufe der letzten Jahre ist die zu analysierende Datenmenge stark exponentiell gewachsen. Traditionelle relationale Datenbanksysteme bekamen zunehmend Performanceprobleme bei der Auswertung dieser Daten. Um dieser Entwicklung Herr zu werden, gab es im Laufe der Zeit verschiedenste Ansätze. Zum einen wurden Datenbankserver mit immer mehr Ressourcen (CPUs, RAM, schnellem Storage) ausgestattet. Zum anderen wurde die Rechenleistung zunehmend auf mehrere Server verteilt (verteilte Datenbanken), beziehungsweise geclustert. Zudem kamen immer mehr Appliance-Systeme auf den Markt, die stark optimierte Hardware mit Möglichkeiten der Ressourcenverteilung kombinierten. Oft wurden eigene spezialisierte Prozessoren für zeitaufwändige Datenbankoperationen, wie Sortierung, Filterung oder Aggregation, bereitgestellt. Die Netezza mit FPGA-Prozessoren ist ein gutes Beispiel hierfür. Diese Ansätze haben alle gemeinsam, dass die Hardware zur Performancesteigerung optimiert wurde. Die meisten Appliance-Systeme, ebenso das Betreiben eines Datenbank-Clusters, sind dabei mit hohen Einstiegskosten und hohen laufenden Kosten verbunden. Diese lohnen sich nur dann, wenn massive Datenmengen analysiert werden müssen und die zu erwartenden Effekte durch den Einsatz eines BI-Systems eine solche hohe Investition rechtfertigen. Für den Mittelstand spielten und spielen diese Ansätze daher eher eine untergeordnete Rolle.

Eine andere Entwicklung zeichnete sich seit Beginn des letzten Jahrzehntes ab. Die zumeist für die Performance kritischste Komponente wurde ins Visier genommen, der Storage. Denn schlechte Performance geht oft mit schlechter I/O (Input/Output) -Geschwindigkeit des Storage (Festplatten, SSDs etc.) einher. Die Grundidee ist, möglichst viele Operationen mit dem Speicher des Datenbankservers abzuwickeln und Zugriffe auf den Storage idealerweise komplett zu vermeiden. Um wiederrum Daten nach Speichermengen optimiert abzuspeichern, beziehungsweise im Speicher zu halten, ist es sinnvoll, die Daten nicht zeilen- sondern spaltenorientiert abzulegen. Es entstanden die spaltenorientierten In-Memory-Datenbanken (IMDB). IBM brachte im Rahmen dieser Entwicklung im Juni 2013 die Datenbankversion Db2 10.5 mit BLU Accelerator auf den Markt.

Dabei wurden folgende vier Kernfunktionalitäten geschaffen:

- Keine Limitierung der In-Memory-Performance auf das verfügbare RAM des Servers
- Spezielle bedarfsoptimierte Komprimierung
- Data Skipping
- Nutzung der CPU-Caches

1.1 In-Memory-Performance ist nicht auf RAM des Servers limitiert

Das Datenbankmanagementsystem greift auf den Storage zu, für den Fall, dass die Datenmenge für den Speicher zu groß wird und Operationen nicht mehr innerhalb des Speichers abgewickelt werden können. Das war beim Erscheinen von Db2 BLU ein sehr wichtiges Differenzierungsmerkmal gegenüber der Konkurrenzprodukte, die immer so viel RAM benötigten, dass die komplette Datenmenge im Speicher gehalten werden konnte. Diese Limitierung führt zu sehr hohen Hardware-Investitionen, da RAM klassischerweise sehr viel teurer ist als Storage, selbst wenn es ein maximal schneller Storage, wie etwa ein Flash-Storage, ist.

1.2 Spezielle bedarfsorientierte Komprimierung

Eine grundlegende Technik, um möglichst viele Daten im Speicher und auch auf dem Storage halten zu können, ist die komprimierte Ablage. Der Nachteil ist, dass eine Dekomprimierung immer CPU-Last erzeugt, was wiederrum die Gesamtperformance eines Systems stören kann. Daher bietet Db2 BLU sehr ausgefeilte Ansätze, um Daten möglichst selten dekomprimieren zu müssen und stattdessen mit den komprimierten Daten direkt zu arbeiten:

- Je häufiger Daten innerhalb der Menge an Ausprägungen in einer Spalte vorkommen, desto höher wird die Komprimierung gewählt.
- Nutzung von sogenannten Offset-Coding. So würde für die Daten 100, 101, 102 und 103 nur einmal der Wert 100 gespeichert werden und für 101, 102 und 103 lediglich der Offset zu 100, also 1,2,3.
- Die Komprimierung stellt sicher, dass die Sortierung innerhalb der Komprimierung erhalten bleibt. Dadurch ist es möglich ohne Dekomprimierung Daten zu vergleichen bzw. zu joinen etc..
- Die Komprimierung orientiert sich an den Größen der Speicherregister der CPU-Caches. Deswegen werden weniger CPU-Zugriffe bei der Verwendung der so komprimierten Daten gebraucht. Außerdem beschleunigt sich dadurch auch die selten notwendige Dekomprimierung.

Zusammengefasst werden mit der bedarfsorientierten Komprimierung gleich mehrere Effekte gleichzeitig erreicht:

- Verringerung des I/O
- Weniger Speicherbedarf
- Reduzierung der CPU-Nutzung

1.3 Data Skipping

Das Grundprinzip des Data-Skipping besteht darin, innerhalb einer Datenmenge Werte zu identifizieren, die zur Ausführung einer Abfrage auf die Daten nicht gebraucht werden. So werden die entsprechenden Speicherseiten nicht angesprochen. Hierzu legt Db2 eine Metatabelle, die sogenannte Synopses Table an, die durch inserts und updates automatisch auf einem aktuellen Stand gehalten wird. Sie enthält vereinfacht gesprochen einen Negativ-Index, der hilft zu erkennen, wo Daten nicht gefunden werden.

Data Skipping wirkt sich, wie die bedarfsorientierte Komprimierung, positiv auf IO, Speicher und CPU-Nutzung aus.

1.4 Nutzung der CPU-Caches

Das Grundprinzip hier ist, SIMD-Instruktionen zu nutzen (Anm.d. Verf.: Single Instruction Multiple Data). Es werden mehrere Daten mit einer CPU-Instruktion verglichen. Zudem werden die Level 1, Level 2 und Level 3 Caches der CPUs maximal genutzt, um Zugriffe auf den RAM des Servers zu vermeiden. Das Konzept geht einher mit der, unter Kapitel 1.2, beschriebenen Methode der bedarfsorientierten Komprimierung.

2 Db2 BLU – Herausforderungen

Mit Hilfe der BLU Technologie kann die Abfrageperformance in ganz erheblichen Maße gesteigert werden. Die Vorgehensweise ist, Zugriffe auf den Storage weitestgehend zu vermeiden und IO-Performance, RAM- und CPU-Nutzung durch verschiedene parallel arbeitende und sich ergänzende Funktionen zu optimieren. Speziell für BI-Systeme, die typischerweise abfragelastig sind, ist dies ein sehr großer Mehrwert, der ohne allzu hohe Hardware-Kosten erreicht werden kann. Zudem ist mit der Restriktion auf maximal 16 CPU-Kerne und maximal 128 GB RAM der Einsatz der Technik auch ohne eine Enterprise Lizenz möglich. Im Falle der noch aktuellen Version11.1 reicht eine Advanced Workgroup Edition, im Falle der aktuellsten Version 11.5 die Standard Edition aus. Allerdings bestehen BI-Systeme in der Regel nicht ausschließlich aus Datamarts. Das eigentliche Core Data Warehouse eines BI-Systems ist typischerweise:

- in dritter Normalform modelliert und
- weniger mit Abfrage- als eher mit insert/update/delete-Workload konfrontiert.

Und genau hier stoßen In-Memory Datenbanken im Allgemeinen an ihre Grenzen. Während sie für die Abfrageperformance bestens optimiert sind, sind schreibende Transaktionen typischerweise weniger performant als bei klassischen zeilenorientierten Datenbanken. Insbesondere dann, wenn in der where clause von updates oder deletes mehrere Bedingungen gesetzt sind, bricht die Performance gegenüber klassischen zeilenorientierten Datenbanken leider ein.

Diese Herausforderung führte zunächst dazu, dass IBM im August 2014 mit dem Fixpack 4 sogenannte Shadow Tables einführte. Das Grundkonzept dahinter ist, dass schreiblastige Aktivitäten auf klassische zeilenorientierte Tabellen angewendet werden. Wird die BLU Beschleunigung gebraucht, setzt man eine spaltenorientierte materialisierte View auf die

zeilenorientierte Tabelle auf, die sogenannte Shadow Table. Alle Tabellen, die typischerweise in Abfragen mit der Shadow Table hinzu gejoined werden, müssen zur Nutzung der BLU Beschleunigung auch spaltenorientiert sein, ob mit oder ohne Shadow Table. Das Datenbankmanagementsystem erkennt dann an der Art der Anfrage, wann die spaltenorientierte bzw. die klassische zeilenorientierte Quelle genutzt wird. Um die materialisierte spaltenorientierte View zeitnah aktuell zu halten, kommt IBM Data Replication zum Einsatz. In der Praxis gibt es allerdings einige kleinere Restriktionen, beispielsweise werden bestimmte Zeitfunktionen oder auch Integritätsmechanismen (etwa restricted foreign keys) nicht unterstützt. Sehr viel Schreiblast führt erhöht die Fehleranfällligkeit der Replikation, die immer wieder manuelle Eingriffe in den Prozess erfordert. Außerdem werden Backup/Recovery-Konzepte und der Umgang mit Logarchiven etwas komplexer in der Handhabung. In einem großen Data Warehouse Projekt bei einem DAX 30 Automobilkonzern, hatten wir die Chance, Db2 BLU mit dem Konzept der Shadow Tables im Beta-Stadium zu testen. Somit konnten wir über unseren Kunden Feedback an die Db2-Entwicklung der IBM zurückspiegeln. Das Konzept der Shadow Tables ist auch in der aktuellen Version der Db2 weiterhin enthalten. Die Weiterentwicklung geht aber eher in die Richtung, die Scheib-Transaktionen in BLU Tabellen stärker zu parallelisieren. Es wird letztlich versucht, ohne die Verwendung von Shadow Tables auszukommen. Dies funktioniert zwischenzeitlich auch zufriedenstellend.

Allerdings sind schreibende und verändernde Transaktionen in zeilenorientierten Tabellen immer noch schneller. Kann man sich bei der Bewirtschaftung eines ausgegliederten Datamarts noch mit Staging-Tabellen und Bulk Loads behelfen, sind BLU Tabellen, die quasi immer verfügbar sein sollen und auf die viele schreibende Transaktionen treffen, nach wie vor keine befriedigende Lösung.

Folgt man der Empfehlung der IBM zur optimalen Nutzung der Db2 mit maximaler Performance für alle Transaktionen, trennt man typischerweise die gesamte Installation in eine BLU Datenbank und eine klassische zeilenbasierte Datenbank, auch innerhalb einer Instanz. Das macht vor allem dahingehend Sinn, als dass zur optimalen Nutzung von BLU diverse Datenbankparameter speziell eingestellt werden sollten und diese Empfehlungen für klassische zeilenorientierte Ansätze eher weniger sinnvoll sind. So sollte etwa der Sortheap Speicher Parameter bei BLU Datenbanken sehr groß gewählt sein, die Page Size muss 32Bit sein etc. Bei gemischten Workloads kommt es daher zu Speicherengpässen und ggf. Performanceverlusten bei der Nutzung der zeilenorientierten Tabellen, wenn alle Vorgaben und Empfehlungen umgesetzt werden. Andersherum funktioniert die BLU Beschleunigung nicht hundertprozentig, wenn die Empfehlungen nicht umgesetzt werden. Dabei sind die Vorgaben, wie etwa die große Page Size, zwingende Voraussetzung. Eine korrespondierende Installation könnte so aussehen, dass Datamarts in einer BLU Datenbank liegen und der ganze Rest in einer zeilenorientierten Datenbank.

Der Nachteil an der Trennung der Datenbanken ist, dass man nun gezwungen ist, Daten zwischen den Datenbanken zu transferieren. Somit ist es dann durch Nutzung des Federation-Features möglich, Tabellen aus beiden Welten miteinander zu joinen. Tendenziell braucht eine solche getrennte Lösung mehr Ressourcen und erzeugt damit potenziell auch höhere Lizenzkosten. Daher entstand aus dem Kundenprojekt zusammen mit dem Automobilhersteller heraus ein Kompromiss-Ansatz. Dieser kommt mit nur einer Datenbank aus und erhält dennoch eine ausgewogene Performance, sowohl

für Abfragen als auch für schreibende und verändernde Transaktionen. Dabei kommt das Data Warehouse mit seiner mehr als ein Terrabyte großen Datenbank sogar mit der „kleinen“ Db2-Lizenz aus. Dieser Lösungsansatz bei einem Großunternehmen, der unter erheblichen Kostendruck entstanden ist, eignet sich ganz hervorragend für mittelständigen Unternehmen. So kommen sie in den Genuss der BLU Beschleunigung, ohne ein allzu komplexes System aufsetzen und verwalten zu müssen und ohne eine Enterprise Lizenz der Db2 nutzen zu müssen.

3 Db2 BLU - für kleinere Installationen oder das Beste aus zwei Welten

Wird beim Einsatz der Db2 die BLU Beschleunigung verwendet oder wenn es generell um besondere Features geht, wird stets der Weg beschrieben, der das beste Ergebnis erzeugt. Dabei steht jedoch nicht immer ein gutes Verhältnis zwischen Mitteleinsatz bzw. Kosten und Nutzen im Fokus, sondern eine Maximierung der Performance und Funktionalität. Der Werkzeugkasten, den Db2 seit der Einführung von BLU bietet, ist jedoch inzwischen so mächtig, dass auch eine kleine Installation ausgesprochen effizient und performant sein kann. Dies gilt insbesondere für ein Data Warehouse bzw. BI-System in der Größenklasse bis ein Terrabyte. Dies unter der Voraussetzung, dass die Datamarts so aggregiert sind, dass sie idealerweise mit weniger als 100 Gigabyte Daten auskommen. Denn nur die Datamarts werden in dem Konzept mit BLU Tabellen abgebildet. Wenn die Datenmenge nicht zu groß ist, wird weniger Speicher exklusiv für BLU benötigt, so dass die „kleine“ Db2 Lizenz ausreicht (bis Db2 11.1.x die Advanced Workgroup Edition bzw. ab Version 11.5 die Standard Edition), um das Szenario abzubilden. Da das Aufsetzen einer Db2 grundsätzlich auf die konkrete Projektsituation, insbesondere den spezifischen Workload und das spezifische Mengengerüst abgestimmt sein muss, ist das schlussendliche Konfigurieren immer individuell.

Nachfolgend einige grundsätzliche Überlegungen dazu.

3.1 Defaults der Datenbank

Folgt man der Installationsanweisung, ist eine der ersten Tätigkeiten das Setzen eines instanzweiten gültigen Parameters:

- db2set workload = analytics

Das hat zur Folge, dass alle Parameter für neue Datenbanken die Vorgaben nutzen, die für BLU gebraucht werden. Insbesondere ist dann BLU beziehungsweise Spaltenorientierung die Vorgabe für neue Tabellen. Genau diese Vorgabe sollte man umstellen, da in dem hier beschriebenen Szenario nur Datamart-Tabellen BLU und die Mehrzahl der Tabellen klassisch zeilenorientiert angelegt werden. Außerdem sollte man bei einem gemischten Einsatz auf Auto-Maintenance eher verzichten und dies zielgerichtet in den Skripten bzw. in die ETL-Prozesse einbauen.

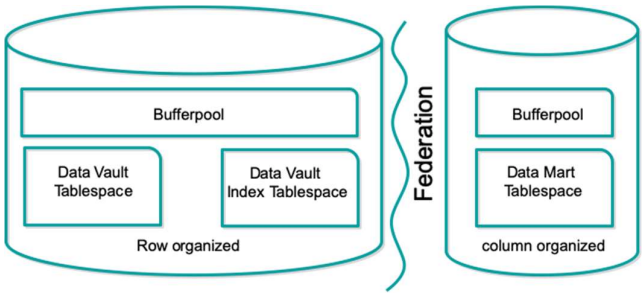
- db2 update db cfg for %DBNAME using dft_table_org=row
- db2 update db cfg for %DBNMAE using AUTO_RUNSTATS=OFF

- db2 update db cfg for %DBNMAE using AUTO_REORG=OFF

Nun gilt es, einige Speicherbereiche sinnvoll zu setzen. Laut Empfehlung sind hier zwei Bereiche besonders speicherhungrig. Abhängig von der Anzahl der gleichzeitigen Zugriffe, gelten folgende Empfehlungen seitens IBM:

- Bis 20 concurrent User: 40% des verfügbaren Speichers fix dem Buffepool zuordnen, der zu den Tablespaces gehört, dem die BLU Tabellen zugeordnet sind. Bei mehr als 20 concurrent Usern sollten es immer noch 25% des gesamten Speichers sein.
- Bis 20 concurrent User: 40% des Speichers für den Sortheap reservieren (sheapthres_shr = 40% des Speichers, sortheap = 1/5 des Thresholds). Bei mehr als 20 concurrent User: 50% des Speichers für den sortheap reservieren (sheapthres = 50% des epcihers, sortheap = 1/20 des Thresholds)
- Trennung von BLU und Nicht-BLU in 2 Instanzen.
- Generell Befüllung der BLU-Tables initial über einen Bulk-Load bzw. am besten über Staging Tabellen und nachgelagerten load from cursor

Das sieht schematisch etwa so aus:



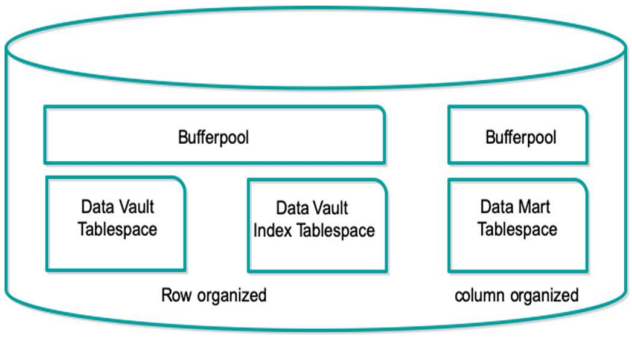
Diese Lösung erfordert tendenziell mehr CPUs und mehr Speicher. Dadurch gelangt man schnell an den Punkt, an dem die Standard-Edition der Db2 nicht mehr ausreicht und zusätzliche Lizenzkosten entstehen. Zudem verdoppeln sich diverse Maintenance Aufwände, wie etwa im Bereich der Desaster-Recovery-Strategie etc.

Daher empfiehlt es sich folgende Kompromiss-Lösung:

- Bis 20 concurrent User: 30% des verfügbaren Speichers fix dem Buffepool zuordnen, der zu den Tablespaces gehört, dem die BLU Tabellen zugeordnet sind. Bei mehr als 20 concurrent Usern sollten es immer noch 20% des gesamten Speichers sein.
- Bis 20 concurrent User: 40% des Speichers für den Sortheap reservieren (sheapthres_shr = 40% des Speichers, sortheap = 1/5 des Thresholds). Bei mehr als 20 concurrent User: 50% des Speichers für den sortheap reservieren (sheapthres = 50% des epcihers, sortheap = 1/20 des Thresholds)

- Generell Befüllung der BLU-Tables initial über einen Bulk-Load bzw. am besten über Staging Tabellen und nachgelagerten load from cursor
- Keine Trennung der Instanzen

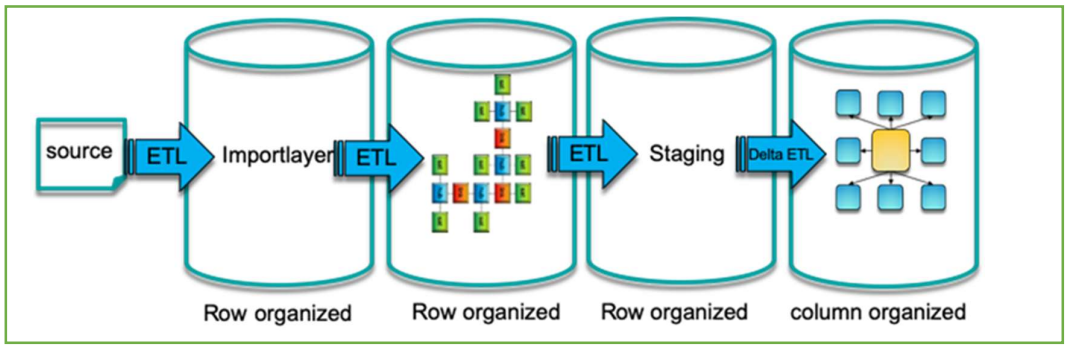
Das sieht dann folgendermaßen aus:



3.2 Datenbewirtschaftung

Bei der Datenbewirtschaftung ist darauf zu achten, dass die BLU-Tabellen insbesondere initial mit BULK-Loads befüllt werden, damit die synapsis Tabelle zu Beginn einen sinnvollen Zustand hat. Außerdem sollte vermieden werden, bei updates oder deletes auf die BLU Tabelle in der where Bedingung mehr als eine Spalte zu verwenden (im Idealfall nur den primary key, der wiederrum nur eine Spalte enthalten sollte). Unter dieser Maßgabe kann nach dem initialen BULK-Load mit insert else update / delete gearbeitet werden. Ansonsten, falls dies prozessual möglich ist, sollte mit einem truncate und anschließendem BULK Load aus der Staging Tabelle fortfahren, um die BLU-Tabellen zu befüllen. Wichtig ist zudem, dass alle Dimensionstabellen, die zu Faktentabellen hinzu gejoined werden sollen, auch BLU sind. Da Dimensionstabellen normalerweise nicht groß sind und es im Datamart keine foreign keys gibt, kann hier problemlos mit truncate / BULK-Load gearbeitet werden.

Das sieht dann schematisch folgendermaßen aus (hier ist im Core-Data Warehouse ein Data Vault verwendet. Das ist optional):



4 Empfehlungen zur Implementierung

Es wurde beschrieben, an welchen Rädern gedreht werden kann, um eine Db2 Instanz so zu konfigurieren, dass sie BLU Beschleunigung und klassische zeilenorientierte Technologie in einer Installation vereint. Minimale Lizenzkosten und ein optimierten Wartungsaufwand ermöglichen es dennoch, in den Genuss der BLU Performance für die Datamarts zu kommen, in denen die Datenabfrage im Vordergrund steht. Die konkrete Ausgestaltung und Konfiguration bedarf allerdings einer eingehenden Analyse der konkreten Projektsituation und lässt sich nicht verallgemeinern. Die Technologie der In-Memory-Datenbanken ist nicht neu, hat aber durch die stetig ansteigenden Datenmengen ihren Platz innerhalb der Datenbankmanagementsysteme gefunden. Doch die Entscheidung für ein IMDB-System ist nicht nur abhängig von der Datenmenge, sondern insbesondere vom Zweck. Es sind viele Einsatzszenarien denkbar, wie beispielsweise bei anspruchsvollen Auswertungen in den Bereichen Business Intelligence, Data Warehousing, Predictive Analytics, IoT-Anwendungen oder Echtzeitanalysen. Die Datenbank-Experten der mip GmbH unterstützen Sie und erarbeiten mit Ihnen gemeinsam die richtigen Schritte der Implementierung.

Kontakt:

Jörg Kremer
Head of Consulting

IBM Certified Database Administrator – Db2 11.1 LUW

Fußnote:

IBM Db2®, IBM Netezza® sowie andere Marken sind Marken der International Business Machines Corporation und in vielen Ländern weltweit registriert oder in Verwendung.