Vergleichende Untersuchung von Datenbanksystemen mit In-Memory-Technologien

Robert Pietzschmann,.....

06.01.2017

Inhaltsverzeichnis

1 Einleitung

Während unseres Projektseminars war es unsere Aufgabe Datenbanksysteme mit In-Memory Technologie vergleichend zu untersuchen. Betreuer bzw. Auftraggeber dessen war Prof. Dr. oec. Gunter Gräfe.
Wir hatten dabei mehrere Schwerpunkte. Es musste untersucht werden, wie Hochverfügbarkeit und Performancesicherung bei In-Memory-Technologien umgesetzt wurden. Es mussten Strategien erarbeitet werden, die zeigen wie Anforderungen an Verfügbarkeit, Ausfalltoleranz, Performance und Konsistenz in den unterschiedlichen Systemen, um transaktionale Daten zu verwalten und auf der anderen Seite die Analyse von großen Daten umgesetzt wurden. Dazu mussten wir uns in die Technologie der In-Memory Datenbank SAP Hana Express und dem MS SQL Server 2016 einarbeiten. Neben diesen beiden Hauptsystemen war noch gefordert selbiges bei NO-SQL Systemen zu überprüfen. Dort wählten wir Cassandra, sowie eine Maria DB mit vorgestelltem Memcache aus. Weiterhin sollten wir auf mehreren Systemen die verschiedenen Technologien umsetzen und in einem selbstgewählten Beispielszenario mit Daten füllen. Diese wurden für die Vergleiche zwischen den Systemen benötigt. Ein möglicher Zusatz dazu sollte der Entwurf einer Übungsaufgabe für das Modul „Erweiterte Datenbanksysteme“.

2 Gruppenmitglieder

Unsere Gruppe bestand während des Projektes aus sechs Mitgliedern. Marcel Kunz, Robin Arnoldt, Clemens Köhler, Phillipp Winkler, Moritz Buchwälder und Robert Pietzschmann.

3 Theorie

3.1 Grundlagen In-Memory Technologie

Der wichtigste Unterschied von einem In-Memory Datenbanksystem gegenüber einem herkömmlichen ist, dass die Datenhaltung im Hauptspeicher des Rechners erfolgt. Das führt zu erheblichen Vorteilen bei der Performance, da die Zugriffzeit so enorm verkürzt wird und Zugriffsalgorithmen sind einfacher. Nachteil ist vor allen der wesentlich höhere Preis für Arbeitsspeicher als für Festplatten. Da aber auch die Preise für jene mit ausreichender Speicherkapazität inzwischen nicht mehr unbezahlbar sind, werden In-Memory Datenbanken immer beliebter. Erst mit der Entwicklung der Kapazität der Hauptspeicher im letzten Jahrzehnt wurde es möglich In-Memory basierte Systeme zu schaffen. Wichtigste Vorraussetzung dafür war die Entwicklung von 64 Bit Systemen. Durch diese wurde es möglich ausreichend großen Arbeitsspeicher zu entwickeln. Im allgemeinen gilt für Speicherarchitekturen, dass je langsamer sie sind, desto billiger werden sie. Zuletzt veränderte sich zwar auch der Rest der Speicher grundlegend durch den Einzug von SSD Festplatten (Flash-Speichern), die wesentlich schneller sind als die Hard Disk. Allerdings sind diese aus Sicht der Software immer noch eine Platte wegen der Persistenz und der Nutzungseigenschaften. Dies hat zur Folge, dass für den Zugriff auf diese immer noch die gleichen Methoden verwendet werden wie für die Hard Disks. Zur vollen Ausnutzung der Geschwindigkeit des Flash Speichermediums müssen diese Algorithmen ständig ernuert werden. Der Hauptspeicher hat dagegen den Vorteil, dass ein direkter Zugriff möglich ist. So dauert das sequenziellen Lesen von einem MB hier nur 250000 ns gegenüber 1851851,9 ns bei einer aktuell schnellen SSD und 30000000 ns bei einer Hard Disk. Alle normalen Operationen der Datenbank werden deshalb im Hauptspeicher augeführt und die Festplatte wird nur für Backups bzw. als Archi genutzt. Als Problem ergibt sich, trotz der wesentlich besseren Performance ein Bottleneck beim Lader der Daten vom RAM in den Cache.
Einhergehend mit der Datenhaltung im Hauptspeicher gibt es einer weitere wesentliche Änderung gegenüber konventionellen Datenbanksystemen. So erfolgt die Datenhaltung spaltenorientiert. Dabei werden alle Reihen (Tupel) in angrenzenden Blöcken gespeichert. Dadurch eignen sie sich perfekt für einen schnellen lesenden Zugriff, was beispielsweise für die Aggregation sinnvoll ist. Zur Reduzierung der Datenmenge werden bei In-Memory Datenbanksystemen verschiedenen Kompressionsverfahren genutzt.
Aufgrund der Datenhaltung im Hauptspeicher ergibt sich noch ein Nachteil aus ökologischer und ökonomischer Sicht. So ist der Energieverbrauch einer von Main-Memory

Datenbankmanagementsystemen erheblich viel höher als der von konventionellen. So verbraucht ein „System“ ganze 1,5 Megawatt.

3.2 In-Memory Technologie im Vergleich zur konventionellen Datenhaltung

3.3 Kompressionsverfahren

Aus dem neuen Performance Bottleneck zwischen Hauptspeicher und Cache ergibt sich die Notwendigkeit Komprimierungsverfahren zu nutzen um eine schnellere Arbeit zu gewährleisten. Weiterhin sind diese notwendig, da trotz der aktuell relativ großen Arbeitsspeicherkapazitäten die Datenbanken oft noch größere Mengen an Daten enthalten, weshalb ohne Komprimierung eine Datenhaltung im Hauptspeicher nicht möglich wäre.

3.3.1 Kompressionsverfahren bei SAP HANA

Basis für die sehr effiziente Kompression stellt hier die spalten-orientierte Speicherung dar. Durch die Verringerung von Bits die zur Darstellung der Daten genutzt werden kann sowohl die Zugriffszeit, als auch der Speicherverbrauch verringert werden.
Grundlage stellt in der HANA das „Dictionary Encoding“ dar. Dabei werden Werte mit einer großen Länge (wie Texte) als Integer Wert gespeichert. Dabei ist es einfach zu verstehen und nicht schwer zu implementieren, was zu höheren Vorteilen in der Performance führt. Es arbeitet dabei spaltenweise. Hat die Tabelle zum Beispiel eine Spalte fName, so wird hier jedem Vornamen eine Positionsnummer zugeordnet. In dem sogenannten „Dictionary“ werden nun die Namen jeweils einzeln eingetragen und ebenso mit IDs versehen. Die beiden IDs werden dann einfach ein einem „Attribut Vector“ verknüpft. Je mehr Dopplungen von Namen in der Spalte fname sind, desto höher ist die Komprimierung. Angenommen jeder Vorname besteht aus 49 Byte und es gibt acht Milliarden Menschen auf der Erde, dann werden rund 365,1GB benötigt um alle zu speichern. Dahingegen braucht der „Attribut Vector“ 23 Bit pro Eintrag multipliziert mit den 8 Millionen Einträgen also 184 Milliarden Bit, was 21,4GByte entspricht. Das „Dictionary“ verbraucht bei 49 Byte pro Eintrag und 5 Millionen Vornamen nur 0,23GByte. Dividiert man nun die Menge ohne Kompression von 365,1GByte mit den 21,63GByte nach Kompression, sieht man, dass man der Speicherverbrauch um den Faktor 17 gesunken ist, was 6 Prozent des ursprünglichen Speicherbedarfs darstellt.

4 Quellen

Einleitung:
- https://de.wikipedia.org/wiki/raussuchenGlobalfoundries (20.12.2016, 14.15 Uhr)
Kompression:
-(07.01.2018 14.51)