SQL Server 2014

SNEK 3 – März 2014

Bernd Jungbluth

www.berndjungbluth.de

Vorstellung

- Bernd Jungbluth

- IT-Erfahrung
  - SQL und Datenbanken seit 1991
  - SQL Server seit Version 7.0

- Freiberuflicher Berater und Entwickler
  - Administration SQL Server
  - Entwicklung und Optimierung von SQL Server-Datenbanken
  - Datawarehouse-Systeme nach Bill Inmon
  - SQL Server Integration Services
  - SQL Server Reporting Services
  - Migration Access nach SQL Server

Agenda

- In-Memory OLTP
  - Transaktionsverarbeitung im Arbeitsspeicher
  - Speicheroptimierte Tabellen
  - Systemintern kompilierte gespeicherte Prozeduren
- Columnstore Index
  - Spaltenbasierte Indexe
  - Nicht gruppierter Columnstore Index
  - Gruppierter Columnstore Index
- Neue Möglichkeiten der Datenbanksicherung
  - Datenbanksicherung in der Cloud
  - Verschlüsselte Datenbanksicherungen
- Übersicht weiterer Neuigkeiten

In-Memory OLTP

- In-Memory OLTP
  - Codename »Hekaton«
  - Kurzform »XTP« für »Extreme Transactional Processing«
  - Das Kernthema vom SQL Server 2014
  - Verlagern der Transaktionsverarbeitung in den Arbeitsspeicher
  - Tabellen im Arbeitsspeicher – »XTP-Tabellen«
  - Gespeicherte Prozeduren im Arbeitsspeicher – »XTP-Prozeduren«

- Ziel
  - Hochperformante Transaktionsverarbeitung – OLTP
  - Anpassen der Database-Engine an die neuen Hardware-Möglichkeiten, wie CPUs mit mehr Kernen, preiswerter Arbeitsspeicher
  - Bessere Nutzung der Hardware
  - Bisher: Festplatte als primäre und Arbeitsspeicher als sekundäre Hardware
  - Effektiver: Arbeitsspeicher als primäre und Festplatte als sekundäre Hardware
  - Reaktion auf Mitbewerber wie Oracle TimesTen, SAP HANA, u.a.

- Vorteile
  - Massive Verbesserung der Performance
  - Bessere Auslastung des Arbeitsspeichers und der CPUs
  - Niedrige Latenzzeiten
  - 10 bis zu 15fach besserer Datendurchsatz
  - Unterstützt hohe Anzahl gleichzeitiger Transaktionen

- Verfügbarkeit
  - Komplette Integration in SQL Server Enterprise Edition
  - Parallel verfügbar neben herkömmlicher Database-Engine
  - Neue Technologie bei gleichbleibenden Lizenzen
  - Komplette Integration in den Leistungsumfang von SQL Server, wie Datenbanksicherung und Wiederherstellung, Erweiterte Ereignisse, Abfrageoptimierer, Hochverfügbarkeit mit AlwaysOn, Ressourcenkontrolle …
  - Komplette Unterstützung durch SQL Server Management Studio
  - Entsprechende DMVs und Systemsichten verfügbar

- Konzept
  - Vom Pessimismus zum Optimismus
  - Nicht zuerst sperren und dann schreiben
  - Einfach schreiben und bei eventuellen Konflikten entsprechend reagieren
  - Transaktionsverarbeitung ohne Latches, Sperren, Blocks und Deadlocks
  - Kein Zwischenspeichern von Daten und Objekten in der tempdb
  - Paralleles Lesen und Ändern von Daten im Arbeitsspeicher

- Funktionsweise
  - Pro Transaktion wird ein Snapshot des Datensatzes erstellt.
  - Der Snapshot wird als Version des Datensatzes im Arbeitsspeicher abgelegt.
  - Die Transaktion wird ohne Sperren am Datensatz komplett ausgeführt.
  - Erst beim Commit wird geprüft, ob es eine neue Version vom Datensatz gibt.
  - Mögliche Konflikte werden wie bei einem herkömmlichen Deadlock bereinigt.
  - Es gibt einen Gewinner, einen Verlierer und eine Fehlermeldung.
  - Die Transaktion wird beendet.

- Daten einer XTP-Tabelle
  - Sequenzielles Speichern der Versionen einzelner Datensätze
  - Pro Version eines Datensatzes eine Datenzeile im Arbeitsspeicher
  - Kein Speichern in Datenseiten und Datenbankdateien
  - Kennzeichnung einer Datenzeile mit »Begin Timestamp« und »End Timestamp«
  - Definiert Version, Gültigkeit und Sichtbarkeit einer Datenzeile
  - Begin Timestamp = Start der Gültigkeit einer Datenzeile
  - End Timestamp = Ende der Gültigkeit einer Datenzeile
  - Aus Gründen der Abwärtskompatibilität maximale Datensatzlänge von 8 KB
  - AKID jederzeit gewährleistet

- Datenermittlung und Datenverarbeitung
  - SELECT: Datenzeilen deren Gültigkeitszeitraum dem Lesezeitpunkt entsprechen
  - INSERT: Aktueller Zeitpunkt als Begin Timestamp, »unendlich« als End Timestamp
  - DELETE: Aktueller Zeitpunkt als Begin Timestamp und End Timestamp
  - UPDATE: Kennzeichnet aktuelle Datenzeile als gelöscht und legt eine neue an

- Speichern der Daten einer XTP-Tabelle
  - Als binäre Werte in 128 MB großen Dateien
  - Aktuell verarbeitete Datenzeilen im Transaktionsprotokoll
  - Auslösen des Speichervorgangs in die Dateien durch »Offline-Prüfpunkt«
- Transaktionsprotokoll
  - Benötigt weniger Speicherplatz als bei herkömmlicher Transaktionsverarbeitung
  - Reduzierte Bandbreite
  - Schnelle Solid State Devices empfehlenswert
- Dateien
  - Datendateien mit den Daten der XTP-Tabelle
  - Enthalten Datenzeilen in fortlaufenden Timestamp-Bereichen
  - Fixe Größe von 128 MB
  - Deltadateien für gelöschte Datenzeilen
  - Zu jeder Datendatei eine Deltadatei verfügbar

- Offline-Prüfpunkt
  - Automatischer Hintergrundprozess
  - Schreibt neue Datenzeilen sequenziell in fortlaufende Timestamp-Bereiche
  - Schreibt gelöschte Datenzeilen in Deltadatei
  - Erstellt bei Bedarf neue Datendatei mit fixer Größe von 128 MB

- Prüfpunkt
  - Regelmäßiges Auslösen eines Prüfpunkts – »Checkpoint«
  - Automatischer Prozess in Ruhephasen oder alle 6 Stunden
  - Spätestens bei 1GB Daten im Transaktionsprotokoll seit letztem Prüfpunkt
  - Reorganisiert Datendateien und Deltadateien
  - Verarbeitet die Einträge der Deltadateien in die Datendateien
  - Erstellt aus 2 Datendateien 1 neue Datendatei ohne die Inhalte der Deltadateien
  - Fasst mehrere Datendateien mit einer Auslastung kleiner 50% zusammen
  - Markiert volle Datendateien als schreibgeschützt
  - Entfernen nicht mehr benötigter Dateien durch Garbage Collector

- Konfiguration
  - Einstellungen an der Datenbank
  - Definition einer XTP-Tabelle

- Datenbank
  - Definition einer FILESTREAM-Dateigruppe für In-Memory OLTP notwendig
  - Dateigruppe mit Option CONTAINS MEMORY_OPTIMIZED_DATA
  - Speichert Datendateien und Deltadateien der XTP-Tabellen
  - Für In-Memory OLTP nur eine Dateigruppe pro Datenbank erlaubt
  - Definition mehrerer Verzeichnisse in Dateigruppe zur Verteilung der Last möglich
  - Unterstützt alle Wiederherstellungsmodelle
  - Unterstützt Vollsicherung und Transaktionsprotokollsicherung
  - Unterstützt keine differenzielle Sicherung
  - Konfiguration im SQL Server Management Studio und per T-SQL
  - Nachträgliches Erweitern bestehender Datenbanken möglich

- Speicheroptimierte Tabelle
  - »Memory-optimized table« oder »XTP-Tabelle«
  - Tabelle als kompilierte DLL im Arbeitsspeicher
- Definition
  - Per CREATE TABLE inklusive Angabe der Spalten
  - Mindestens 1 Index erforderlich
  - Ergänzt mit WITH (MEMORY_OPTIMIZED = ON)
  - Plus Definition der Dauerhaftigkeit per DURABILITY
- Limitationen
  - Kein Ändern der Tabellendefinition mit ALTER TABLE möglich
  - Keine LOB- und CLR-Datentypen, sowie berechnete Spalten erlaubt
  - Keine Einschränkungen, Fremdschlüsselbeziehungen und Trigger möglich
  - Kein IDENTITY vorhanden

- Index
  - Hält die Datenzeilen einer Tabelle zusammen
  - Beschreibt den Inhalt einer Tabelle und somit die Tabelle selbst
  - »Nicht gruppierte Hash-Indexe« und »Nicht gruppierte Range-Indexe«
  - Lediglich im Arbeitsspeicher vorhanden
  - Automatisches Erstellen der Indexe beim Start der Datenbank
  - Indexpflege durch Garbage Collector

- Limitationen
  - Definition nur möglich beim Erstellen der Tabelle
  - Nachträgliches Anlegen, Ändern oder Löschen von Indexen nicht erlaubt
  - Auf maximal 8 nicht gruppierte Indexe begrenzt
  - Keine gruppierten Indexe verfügbar
  - Neben Primärschlüssel keine weiteren eindeutigen Schlüssel möglich
  - BIN2-Zeichensatz bei Indexen auf Spalten mit Zeichenfolgen erforderlich
  - Keine NULL-Werte in Indexspalten erlaubt

- Nicht gruppierter Hash-Index
  - Optimiert für Suchen mit Gleichheitsprädikaten
  - Basiert auf »Hashing mit Verkettung« – Hashwerte, Hash Buckets und Chains
  - Hash Bucket = Zusammenfassen gleicher Hashwerte
  - Chain = Verweis einer Datenzeile zur nächsten Datenzeile mit gleichem Hashwert
  - Verweis eines Hash Buckets nur zur ersten Datenzeile des Chains
  - Festlegen der Größe des Hash-Index mittels BUCKET_COUNT erforderlich
  - Als Größe doppelte Anzahl der eindeutigen Werte der Indexspalte empfehlenswert
- Nicht gruppierter Range-Index
  - Optimiert für Suchen mit Ungleichheitsprädikaten und Sortierungen
  - Besteht aus einer B-Tree-Variante ohne Sperren und Latches namens Bw-Tree
  - Arbeitet mit Page Mapping Table, Index Pages und Delta Pages
  - Page Mapping Table = Zuordnung einer Speicheradresse zu einer Index Page-ID
  - Delta Pages = Enthalten Änderungen an den Werten eines Indexes
  - Variable Größe durch Anzahl der Zeilen und Größe der Indexschlüsselspalten

- Dauerhaftigkeit
  - Im Original »Durability«
  - Definition des Speicherumfangs einer XTP-Tabelle
- SCHEMA_AND_DATA
  - Standardeinstellung
  - Protokolliert die Transaktionen im Transaktionsprotokoll
  - Speichert Struktur und Daten der XTP-Tabelle als Dateien
  - Definition eines Primärschlüssels zwingend erforderlich
  - Wiederherstellen der Daten möglich
- SCHEMA_ONLY
  - Speichert lediglich die Struktur der XTP-Tabelle als Datei
  - Keine Protokollierung von Transaktionen im Transaktionsprotokoll
  - Wiederherstellen der Daten nicht möglich
  - Definition per WITH (DURABILITY=SCHEMA_ONLY)

In-Memory OLTP

- Demo

- Transaktionsverarbeitung bei In-Memory OLTP
  - Datenermittlung und Datenverarbeitung an XTP-Tabellen
  - Auf herkömmliche Art und Weise
  - Per »Systemintern kompilierte gespeicherte Prozeduren«
  - AKID jederzeit gewährleistet
  - Verwenden von kurzen Transaktionen empfehlenswert

- Herkömmliche Art und Weise
  - Per Ad-hoc-Abfragen
  - Per Sichten, Gespeicherte Prozeduren und Funktionen
  - Keine Änderung zu bisherigem T-SQL notwendig
  - Bestehende SQL-Skripte mit Ad-hoc-Abfragen weiterhin nutzbar
  - Bestehende Sichten, Gespeicherte Prozeduren und Funktionen weiterhin nutzbar
  - Kein Migrationsaufwand im Quellcode
  - Tabellenverknüpfungen von herkömmlichen Tabellen und XTP-Tabellen möglich
  - Mittlere Performance bei der Datenverarbeitung an XTP-Tabellen

- Systemintern kompilierte gespeicherte Prozedur
  - »Native Compiled Stored Procedure« oder »XTP-Prozedur«
  - Gespeicherte Prozedur als kompilierte DLL im Arbeitsspeicher
  - Kompilierter C-Code
  - Deutlich höhere Performance bei der Datenverarbeitung an XTP-Tabellen
  - Entwicklung in T-SQL ähnlich herkömmlicher Gespeicherter Prozeduren

- Limitationen
  - Einmalige Abfrageoptimierung beim Erstellen der XTP-Prozedur
  - Kein Ändern per ALTER PROC möglich
  - Eigenes neu entwickeltes und noch recht spartanisches T-SQL
  - Unterstützt nicht alle herkömmlichen T-SQL-Anweisungen
  - Unterstützt nur die Isolationsstufen Snapshot, Repeatable Read und Serializable
  - Keine Unterstützung von Read Commited und Read Uncommited
  - Kein Zugriff auf herkömmliche Tabellen möglich
  - Löschen der XTP-Prozeduren vor Änderungen an XTP-Tabellen notwendig

- Definition
  - Per CREATE PROC
  - Ergänzt mit WITH (NATIVE_COMPILATION)
  - Plus erforderlicher Definition der Rahmenbedingungen
  - SCHEMABINDING zur Definition der Schemabindung
  - EXECUTE AS zur Angabe vom Ausführungskontext
  - Grundsätzlich keine Definition von dynamischen Elementen möglich

- ATOMIC-Block
  - BEGIN ATOMIC WITH plus erforderliche Definition der Rahmenbedingungen
  - TRANSACTION ISOLATION LEVEL zur Definition der Isolationsstufe
  - LANGUAGE zur Definition der Sprache
  - Enthält den Quellcode der XTP-Prozedur
  - Führt alle Anweisungen als eine Transaktion aus
  - Startet automatisch eine neue Transaktion, wenn keine Transaktion aktiv ist
  - Legt bei bereits aktiver Transaktion einen Sicherungspunkt an

In-Memory OLTP / Systemintern kompilierte Gespeicherte Prozedur

- Demo

- Migration
  - Migration der gesamten Datenbank nicht zwingend erforderlich
  - Betrifft performancekritische Tabellen und Gespeicherte Prozeduren
  - Kalkulation des tatsächlich benötigten Arbeitsspeichers notwendig
  - Maximal 80% des Arbeitsspeichers für XTP verfügbar

- AMR-Tool
  - AMR = »Analysis, Migration and Reporting«
  - Liefert Bericht »Übersicht der Transaktionsleistungsanalyse«
  - Zeigt die für eine Migration empfohlenen Tabellen und Gespeicherten Prozeduren
  - Verfügbar im »Verwaltungs-Data Warehouse«
  - Basiert auf den Daten der Datensammlung – dem »Data Collector«
  - Verwendet Datensammlungssätze »Analyse zur Tabellenverwendung« und »Analyse zur Nutzung gespeicherter Prozeduren«
  - Auswerten von Instanzen älterer Versionen ab SQL Server 2008 möglich

- Ratgeber für die Speicheroptimierung
  - Analysiert eine herkömmliche Tabelle für die Migration zur XTP-Tabelle
  - Liefert Fehler, Warnungen und Informationen zur Migration
  - Migriert die herkömmliche Tabelle zu einer XTP-Tabelle
  - Übernimmt bei Bedarf auch die Daten in die XTP-Tabelle
  - Definition der Dauerhaftigkeit möglich
  - Definition eines Primärschlüssels möglich
  - Definition von Hash-Indexen und Range-Indexen möglich
  - Verfügbar im Kontextmenü einer herkömmlichen Tabelle

- Ratgeber für systeminterne Kompilierung
  - Analysiert eine Gespeicherte Prozedur für die Migration zur XTP-Prozedur
  - Liefert lediglich Hinweise auf in XTP-Prozeduren nicht unterstütztes T-SQL
  - Erstellt keine XTP-Prozedur
  - Verfügbar im Kontextmenü einer herkömmlichen Gespeicherten Prozedur

- Demo

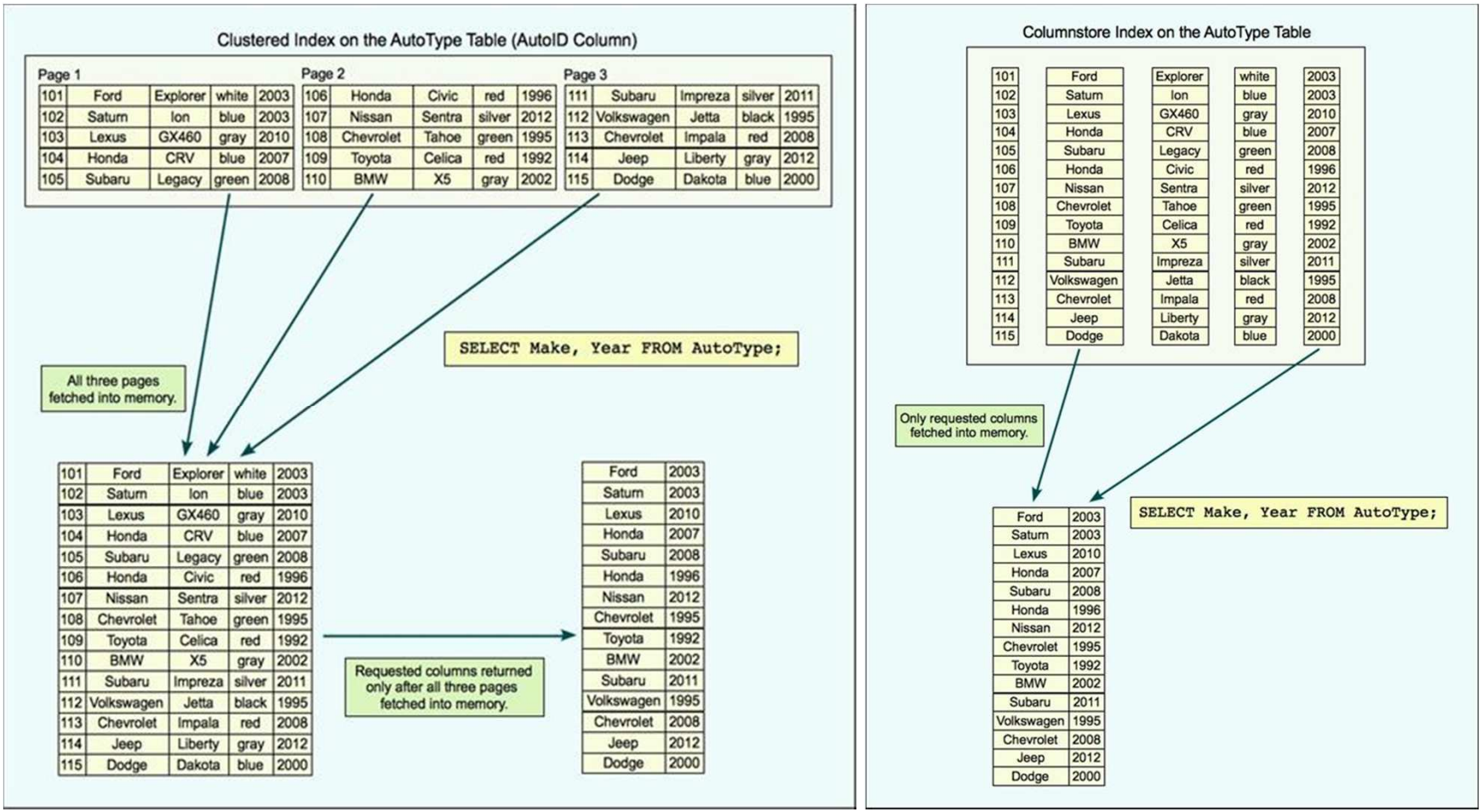
Agenda

- In-Memory OLTP
  - Transaktionsverarbeitung im Arbeitsspeicher
  - Speicheroptimierte Tabellen
  - Systemintern kompilierte gespeicherte Prozeduren
- **Columnstore Index**
  - **Spaltenbasierte Indexe**
  - **Nicht gruppierter Columnstore Index**
  - **Gruppierter Columnstore Index**
- Neue Möglichkeiten der Datenbanksicherung
  - Datenbanksicherung in der Cloud
  - Verschlüsselte Datenbanksicherungen
- Übersicht weiterer Neuigkeiten

Columnstore Index

- Columnstore Index
  - Spaltenbasierte Database-Engine für Indizes
  - Optimiert für die Verarbeitung im Arbeitsspeicher
  - Verwendet »X-Velocity In-Memory Compression Engine«
- Zeilenbasiertes Speichern
  - »Rowstore«
  - Datensätze als Zeilen einer Tabelle
  - Speichert die Daten einer oder mehrerer Zeilen auf einer Seite
  - Enthält auf den Seiten die Daten zu jeder Spalte der Zeilen
- Spaltenbasiertes Speichern
  - »Columnstore«
  - Speichert die Daten einer Spalte auf einer Seite
  - Enthält auf den Seiten nur die Daten der jeweiligen Spalte
  - Speichert die Daten in komprimierter und verschlüsselter Form

Columnstore Index / Unterschied Rowstore und Columnstore



Quelle: https://www.simple-talk.com/sql/database-administration/columnstore-indexes-in-sql-server-2012/

- Vorteile
  - Starke Performance beim Lesen der Daten einer oder mehrerer Spalten
  - Starke Performance beim Aggregieren der Daten einer oder mehrerer Spalten
  - Steigerung der Abfrageperformance bis zum 10fachen
  - Weniger Speicherbedarf durch bessere Komprimierung
  - Im Gegensatz zur herkömmlichen Methode effizientere Datenzugriffe
  - Weniger I/Os
- Einsatzmöglichkeiten
  - In Datawarehouse-Systemen
  - An Faktentabellen in mehrdimensionalen Datenbanken
  - Ab einer Datenmenge von 1.000.000 Datensätze pro Tabelle
  - Nicht geeignet bei Tabellen mit geringer Datenmenge
  - Nicht geeignet bei Tabellen mit vielen Datenänderungen
  - Nicht geeignet in OLTP-Systemen

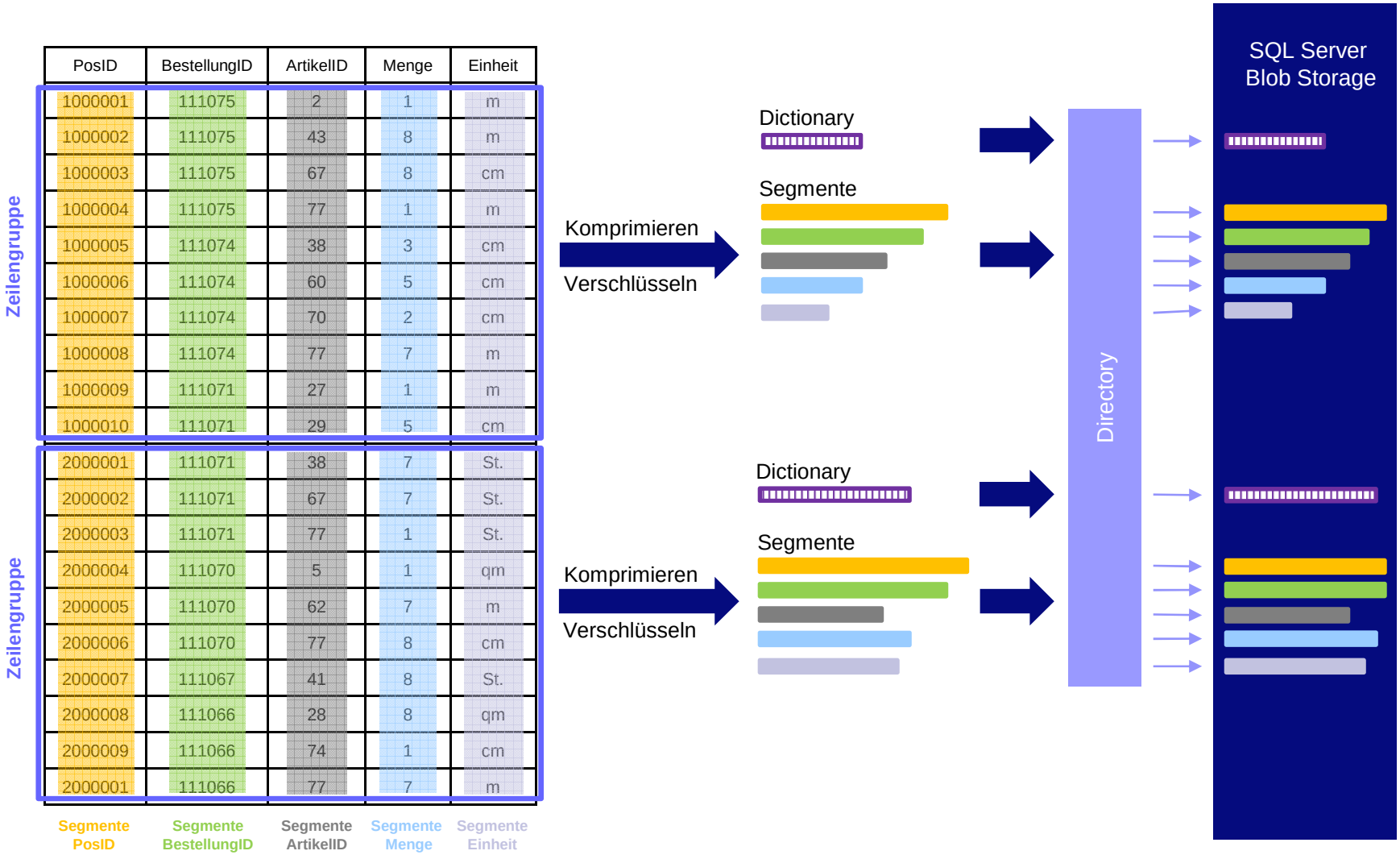
Columnstore Index

- Nicht gruppierter Columnstore Index
  - Im Original »nonclustered Columnstore Index«
  - Eingeführt mit SQL Server 2012
  - Weniger Limitationen in SQL Server 2014
  - Schreibgeschützter Index
  - Verhindert Datenmanipulation an der Tabelle
  - Optimal für Datawarehouse-Systeme
- Gruppierter Columnstore Index
  - Im Original »clustered Columnstore Index«
  - Verfügbar mit SQL Server 2014
  - Weiterentwicklung des nicht gruppierten Columnstore Index
  - Kein Schreibschutz an Index und Tabelle
  - Optimal für Realtime Datawarehouse-Systeme
  - Nicht geeignet in OLTP-Systemen
  - Gruppierter Columnstore Index als Indexierung einer Tabelle ausreichend

- Limitationen
  - Kein Ändern des Columnstore Index per ALTER INDEX möglich
  - Maximal 1024 Spalten
  - Nicht alle Datentypen erlaubt, bspw. LOB- und CLR-Datentypen, XML, u.a.
  - Keine Spalten mit geringer Dichte – Sparse-Columns – erlaubt
  - Keine Sortierung im Index möglich
  - Keine Include- und Filtered-Index möglich
  - Keine Unterstützung bei Replikation, Filestream, Change Tracking und CDC
  - Nur verfügbar in Enterprise Edition

- Funktionsweise
  - Teilt die Zeilen einer Tabelle in Zeilengruppen auf
  - Trennt die Spalten einer Zeilengruppe in Segmente
  - Komprimiert und verschlüsselt die Segmente
  - Erstellt lokale und globale Dictionaries zu den Daten der Segmente
  - Speichert Segmente und Dictionaries als LOB im SQL Server-Blob-Storage

Columnstore Index / **Funktionsweise**



Columnstore Index / **Begriffe**

- Zeilengruppe
  - Aufteilung der Zeilen einer Tabelle in Gruppen
  - Maximal 1.048.576 Zeilen pro Zeilengruppe möglich
- Segment
  - Beinhaltet die Daten einer Spalte aus einer Zeilengruppe
  - Kleinste Speichereinheit eines Columnstore-Index
- Dictionary
  - Inhaltsverzeichnisse für die Segmente
  - Lokale pro Segment und globale übergreifend für alle Segmente
- Directory
  - Enthält Verweise zu den Speicherorten der Segmente und Dictionaries
  - Verwaltung der Segmente und Dictionaries im SQL Server-Blob-Storage auf herkömmliche Art und Weise über 8 KB-Datenseiten

- Gruppierter Columnstore Index
  - Nicht nur ein Index, sondern die Tabelle selbst
  - Speichert Daten der Tabelle in komprimierten Segmenten
  - Beinhaltet keine sortierten Daten
- Limitationen
  - Kein Primärschlüssel an Tabelle erlaubt
  - Keine Fremdschlüssel und eindeutige Schlüssel erlaubt
  - Keine weiteren Indexe erlaubt
  - Plus die weiteren bereits genannten Limitationen des Columnstore Index
- Datenverarbeitung
  - Hinzufügen neuer Zeilen über Deltastore
  - Löschen vorhandener Zeilen über Deleted Bitmap
  - Ändern vorhandener Zeilen über Deltastore und Deleted Bitmap
  - Verarbeiten der Daten aus Deltastore und Deleted Bitmap zu Segmenten

Clustered ColumnStore Index

- Demo

- Deltastore
  - Speicherort für neue Zeilen und neue Versionen geänderter Zeilen
  - Zeilenbasiertes Speichern der Daten in 8 KB-Datenseiten
- Deleted Bitmap
  - Enthält Markierungen zu gelöschten Zeilen
  - Speichert lediglich logische Verweise zu gelöschten Zeilen
- Datenverarbeitung
  - INSERT: Speichern der neuen Zeilen im Deltastore
  - Löschen und Ändern abhängig vom Verarbeitungsstand der Deltastores
  - DELETE: Markieren der Zeilen im Deleted Bitmap
  - Zeile noch im Deltastore: Löschen der Zeile im Deltastore
  - UPDATE: Markieren der Zeile im Deleted Bitmap
    und Hinzufügen der Zeile mit den neuen Werten im Deltastore
  - Zeile noch im Deltastore: Ändern der Zeile im Deltastore

- Speichern der Datenänderungen im Columnstore Index
  - Verschieben der Zeilen aus Deltastore in den Columnstore Index
  - Löschen der im Delete Bitmap markierten Zeilen im Columnstore Index
- Tuple Mover
  - Automatischer Prozess alle 5 Minuten
  - Manuell per Systemprozedur »sp_cci_tuple_mover«
  - Manuell bei ALTER INDEX REBUILD und ALTER INDEX REORGANIZE
  - Manuell bei ALTER TABLE REBUILD
- Funktionsweise vom Tuple Mover
  - Ermittelt die Segmente der betroffenen Zeilengruppen
  - Entkomprimiert und entschlüsselt die Daten der Segmente
  - Aktualisiert die Segmente mit den Daten vom Deltastore und Deleted Bitmap
  - Komprimiert und verschlüsselt die Segmente
  - Sperrt dabei die Segmente und zugehörigen Deltastores exklusiv

Gruppierter ColumnStore Index

- Demo

Agenda

- In-Memory OLTP
  - Transaktionsverarbeitung im Arbeitsspeicher
  - Speicheroptimierte Tabellen
  - Systemintern kompilierte gespeicherte Prozeduren
- Columnstore Index
  - Spaltenbasierte Indexe
  - Nicht gruppierter Columnstore Index
  - Gruppierter Columnstore Index
- **Neue Möglichkeiten der Datenbanksicherung**
  - **Datenbanksicherung in der Cloud**
  - **Verschlüsselte Datenbanksicherungen**
- Übersicht weiterer Neuigkeiten

Neue Möglichkeiten der Datenbanksicherung

- Datenbanksicherung in der Cloud
  - Speichern der Sicherungsdatei im Windows Azure-Blob-Speicherdienst
  - Stichwort: »SQL Server-Sicherung über URL«
- SQL Server Managed Backup für Windows Azure
  - Neuer Dienst zum Automatisieren von Datenbanksicherungen
  - Globale Konfiguration für Sicherungen auf SQL Server Instanz-Ebene
  - Betrifft alle vorhandenen und zukünftigen Datenbanken
  - Konfiguration von Ausnahmen pro Datenbank möglich
  - Ermittelt die notwendigen Sicherungstypen automatisch
  - Speichert die Sicherungsdateien im Windows Azure-Blob-Speicherdienst
- Verschlüsselte Datenbanksicherungen
  - Verschlüsselung per Zertifikat oder asymmetrischem Schlüssel
  - Unterstützte Algorithmen: AES 128, AES 192, AES 256 und Triple DES
  - Nur verfügbar bei »SQL Server-Sicherung über URL«

Neue Möglichkeiten der Datenbanksicherung

- Demo

Agenda

- In-Memory OLTP
  - Transaktionsverarbeitung im Arbeitsspeicher
  - Speicheroptimierte Tabellen
  - Systemintern kompilierte gespeicherte Prozeduren
- Columnstore Index
  - Spaltenbasierte Indexe
  - Nicht gruppierter Columnstore Index
  - Gruppierter Columnstore Index
- Neue Möglichkeiten der Datenbanksicherung
  - Datenbanksicherung in der Cloud
  - Verschlüsselte Datenbanksicherungen
- **Übersicht weiterer Neuigkeiten**

Weitere Neuerungen

- Integration in Windows Azure
  - Speichern von Datenbankdateien im Windows Azure-Blob-Speicherdienst
  - Bereitstellen lokaler Datenbanken in virtuellen Windows Azure-Computern
  - Integration von AlwaysOn in Windows Azure
- Verzögerte Dauerhaftigkeit
  - Asynchrones Schreiben ins Transaktionsprotokoll
  - Geringere Latenzzeiten
  - Möglicher Datenverlust
- Verbesserter Abfrageoptimierer
  - Überarbeitete Logik der Kardinalitätsschätzung
  - Nur für Datenbanken mit Kompatibilitätsmodus 120 verfügbar
  - Ausführlicher Test des neuen Abfrageverhaltens nach der Migration auf SQL Server 2014 empfehlenswert

Weitere Neuerungen

- T-SQL
  - Verbessertes SELECT INTO durch mögliche Parallelisierung
  - Inlinespezifikation von Indexen bei der Definition einer Tabelle möglich
- Weiteres
  - Konfiguration von SSD als Pufferpoolerweiterung möglich
  - Acht sekundäre Replikas bei AlwaysOn möglich
  - Neuerstellen einzelner Partitionen bei partitionierten Tabellen möglich
  - Erstellen von Statistiken pro Partition mit INCREMENTAL möglich
  - Neue Funktionen in der Ressourcenkontrolle – »Resource Goveneur«
  - Online-Prozesse mit WAIT_AT_LOW_PRIORITY konfigurierbar
  - Neue Sicherheitserweiterungen auf Serverebene
- Business Intelligence
  - Neuerungen in Analysis Services gleich denen aus SQL Server 2012 SP1
  - Keine Neuerungen in Reporting Services und Integration Services

Zusammenfassung und Fazit

- In-Memory OLTP
  - Enorme Performanceverbesserung
  - Zurzeit noch geringe T-SQL-Unterstützung
  - Mittlerer Migrationsaufwand bei Tabellen
  - Hoher Migrationsaufwand bei Gespeicherten Prozeduren
- Columnstore Index
  - Enorme Performanceverbesserung
  - Nur sinnvoll in Datawarehouse-Systemen
  - Nicht geeignet in OLTP-Systemen
- Fazit
  - Nicht viele Neuerungen in SQL Server 2014
  - Beginn einer neuen Ära in der Welt der Datenbanken
  - In-Memory-Technologie und Columnstore-Index

Links

- In-Memory OLTP
  - Whitepaper von Kalen Delaney http://t.co/T6zToWc6y6
  - Aktivieren und Auswerten des AMR-Tools http://www.mssqltips.com/sqlservertip/3137/getting-started-with-the-amr-tool-for-migration-to-sql-server-inmemory-oltp-tables/
  - Liste nicht unterstützter T-SQL-Befehle für eine XTP-Prozedur http://msdn.microsoft.com/de-de/library/dn133189%28v=sql.120%29.aspx
- Columnstore Index
  - Blog von Niko Neugebauer http://www.nikoport.com/columnstore
  - PDF von Microsoft zu Apollo3: http://research.microsoft.com/pubs/193599/Apollo3%20-%20Sigmod%202013%20-%20final.pdf
- SQL Server 2014
  - Übersicht Neuerungen: http://msdn.microsoft.com/de-de/library/bb510411(v=sql.120).aspx
  - Übersicht nicht mehr verfügbar oder abgekündigter Funktionen: http://msdn.microsoft.com/de-de/library/cc280407(v=sql.120).aspx

Danke

Vielen Dank für die Aufmerksamkeit ☺