

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Strategien für effiziente Indizierung

Uwe Ricken
Microsoft Certified Master: SQL Server 2008
Microsoft Valued Professional: SQL Server

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Uwe Ricken

db Berater GmbH

- Seit 1991 in der IT
- Mai 2013: MCM – SQL Server 2008
- Juli 2013: MVP SQL Server

www: <http://www.db-berater.de>

email: uwe.ricken@db-berater.de

blog: <http://db-berater.blogspot.de>

twitter: <https://twitter.com/dbberater>

xing: http://www.xing.com/profile/Uwe_Ricken

Linkedin: <http://www.linkedin.com/pub/uwe-ricken/76/822/605>

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Uwe Ricken

db Berater GmbH

Microsoft
CERTIFIED
Master
SQL Server® 2008

The Microsoft Certified Solutions Master (MCSM) and Microsoft Certified Master (MCM) programs recognize the most experienced IT professionals who demonstrate and validate their deep technical skills to successfully design and implement technical solutions in a complex business environment.

Microsoft
CERTIFIED
Solutions Expert
Data Platform

Microsoft Certified Solutions Expert (MCSE) certifications are designed to recognize IT professionals who can design and build solutions across multiple technologies, both on-premises and in the cloud.

Microsoft
Most Valuable
Professional

The Microsoft Most Valuable Professional (MVP) Award is our way of saying thank you to exceptional, independent community leaders who share their passion, technical expertise, and real-world knowledge of Microsoft products with others.

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

AGENDA

- Indexobjekte (Heaps / Cluster / Non Cluster / ColumnStore)
- Inside SQL Server – Data Storage Internals
- Parameter Sniffing
- SARGable Queries
- Sort Spills
- Statistiken
- Ausführungspläne

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Anatomie eines Index
HEAP

```
graph TD; R[R] --> L1[L]; R --> L2[L]; R --> L3[L];
```

- HEAP = Relation ohne Ordnungskriterium
- ein ROOT-Knoten (Root Page)
- n Leaf-Elemente (Data Pages)
 - In Leafs werden Datensätze gespeichert
- Leafs besitzen KEINE Verbindung miteinander

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Anatomie eines Index
Clustered Index

- Clustered Index ist Relation mit einem Ordnungskriterium (Clustered Key)
- Index_id = 1
- Clustered Index ist in B-Tree-Struktur organisiert.
- Jede Indexzeile in einem B-Tree zeigt auf eine neue Indexseite oder eine Datenseite in der nächsten Ebene
- Unterste Ebene (0) beinhaltet die Datenzeilen (Leaf)

```
graph TD; R[R] --> I1[I]; R --> I2[I]; I1 --> L1[L]; I1 --> L2[L]; I2 --> L3[L]; I2 --> L4[L]; L1 <--> L2; L2 <--> L3; L3 <--> L4;
```

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Anatomie eines Index

Non Clustered Index

- NCI weisen eine Struktur auf, die von den Datenzeilen getrennt ist.
- Jeder Schlüsselwerteintrag verfügt über einen Zeiger auf die Datenzeile, die den Schlüsselwert enthält.
- Der Blattebene des NCI können Nichtschlüsselspalten hinzu gefügt werden.

<http://technet.microsoft.com/en-us/library/ms177484.aspx>

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Columnstore Indexes

	ProductKey	OrderDateKey	DueDateKey	ShipDateKey	CustomerKey	PromotionKey	CurrencyKey	SalesTerritoryKey	SalesOrderNumber
1	310	20060321	20060402	20060328	19783	3	6	9	4F17EAB3-2B9F-440D-B
2	310	20060321	20060402	20060328	19783	6	9	9	6E94E530-4883-4EE0-B
3	310	20060321	20060402	20060328	19783	1	6	9	29C65098-EE4F-4A7A-A
4	310	20060321	20060402	20060328	19783	6	9	9	57662670-A13A-408A-B
5	310	20060321	20060402	20060328	19783	4	9	9	8C3P234C-C8C3-45F2-B
6	310	20060321	20060402	20060328	19783	6	9	9	94738E11-E241-4D69-B
7	310	20060321	20060402	20060328	19783	3	6	9	64B113C1-0E31-4A45-B
8	310	20060321	20060402	20060328	19783	6	9	9	94D23A87-9434-4331-A

Page: 100
Page: 101
Page: 102

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Columnstore Indexes

LOB Page	LOB Page	LOB Page	LOB Page	LOB Page
ProductKey	OrderDateKey	DueDateKey	SalesOrderNumber	SalesAmount
310	20050818	20050830	FAB81D7A-4ED4-420F-8	3578.27
310	20050818	20050830	B829593B-851A-4264-8	3578.27
310	20050818	20050830	53202119-0183-4160-8	3578.27
310	20050818	20050830	487DB9CD-914D-4D6E-9	3578.27
310	20050818	20050830	1B06485E-94F6-4F99-8	3578.27
310	20050818	20050830	C88A12BA-5E4F-4B7D-A	3578.27
310	20050818	20050830	2429F162-9811-45E3-A	3578.27
310	20050818	20050830	FBE6FA39-6376-4EE7-8	3578.27
310	20050818	20050830	450F3216-2CA5-4FBF-9	3578.27

Strategien für effiziente Indizierung

Uwe Ricken

SNEK3, März 2014

3

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Columnstore Indexes

ProductKey	OrderDateKey	DueDateKey	ShipDateKey	CustomerKey	PromotionKey	CurrencyKey
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
214	20080526	20080708	20080703	15594	1	6
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100
217	20080506	20080518	20080513	13233	1	100

Segmente

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Anatomie eines Index

Data Storage Internals

HEAP / INDEX

Partition 1

Partition 2

Partition (n)

AllocationUnit / IAM

in row data

row overflow data

LOB data

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Anatomie eines Index

Data Storage Internals

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %
SELECT * FROM (dbo.tbl_master) WHERE (c1)=1

Index Scan (Index Scan)

Index Scan (Index Scan)

Schlüsselzugriff (Clustered)

Abfrage 2: Abfragekosten (in Relation zum Batch): 58 %
SELECT * FROM (dbo.tbl_master) WHERE (c1)=1

Clustered Index Scan (Clustered)

DEMO

Datei: 0002 - Anatomie eines Index.sql

Strategien für effiziente Indizierung

Uwe Ricken

SNEK3, März 2014

4

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Parameter Sniffing

- SQL Server erstellt basierend auf Parameterwert einen Ausführungsplan
- Ausführungsplan wird im Plan Cache gespeichert
- Alle weiteren Abfragen mit unterschiedlichen Parameterwerten verwenden den gespeicherten Ausführungsplan

• DAS KANN ZU PROBLEMEN FÜHREN!

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Parameter Sniffing

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %
SELECT * FROM dbo.tbl_masters WHERE (c3)=45

Abfrage 2: Abfragekosten (in Relation zum Batch): 58 %
SELECT * FROM dbo.tbl_masters WHERE (c3)=45

DEMO

Datei: 11 - Parameter Sniffing (demo).sql

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Parameter Sniffing - Lösungen

- OPTION (RECOMPILE)
- OPTION (OPTIMIZE FOR (@VARIABLE=VALUE))
- OPTION (OPTIMIZE FOR (@VARIABLE UNKNOWN))
- Verwendung von lokalen Variablen!

Strategien für effiziente IndizierungUwe Ricken

SNEK3, März 2014

5

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SARGable Abfragen

- SARGable = **S**earch**A**RGumentable Queries
- SARGable ist eine Abfrage dann, wenn ein Datensatz gefunden wird, indem ein Index effektiv verwendet werden kann
- Funktionen auf der Linken Seite eines Predicates verhindern die Verwendung eines Index, da sie für jeden Wert im Attribut eine Berechnung durchführen müssen.

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SARGable Abfragen

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %
SELECT * FROM dbo.tbl_masters WHERE (c3)=45

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %
SELECT * FROM dbo.tbl_masters WHERE (c3)=45

Abfrage 2: Abfragekosten (in Relation zum Batch): 58 %
SELECT * FROM dbo.tbl_masters WHERE (c3)=45

DEMO

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SARGable Abfragen - Lösungen

- Vermeidung von Funktionen, die eine Neuberechnung des Attributs erfordern.
- Vermeidung von Funktionen die eine Typenkonvertierung durchführen
- Erstellung von berechneten Attributen mit Index

Strategien für effiziente IndizierungUwe Ricken

SNEK3, März 2014

6

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Wie führt SQL Server eine Abfrage aus?

Workers (threadpool)

1. Request creates a new Task

2. Idle Worker picks up a pending Task

3. An execution plan is compiled

4. The Query plan is executed. Operators access data through Buffer Pool

5. Result set is returned during execution

6. Task is complete, Worker returns to idle

Task Execution

Query Execution

Cache

Plan Cache

Data Cache (Buffer pool)

Database

Network

Tasks Queue

Abbildung: <http://rusanu.com/2013/08/01/understanding-how-sql-server-executes-a-query/>

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SPILL to tempdb

- SQL Server versucht immer, Abfragen ausschließlich im RAM auszuführen.
- Vor der Ausführung einer Abfrage kalkuliert SQL Server den benötigten Speicher auf Basis von Statistiken.
- Reicht der zugewiesene Speicher nicht aus, müssen Operationen in TEMPDB ausgeführt werden!
- SPILL's können mit Extended Events aufgezeichnet werden

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SPILL to tempdb

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %

SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81

Hashed Aggregate (Clustered)

Hash Match (Clustered)

Clustered Index Scan (Clustered)

Abfrage 2: Abfragekosten (in Relation zum Batch): 58 %

SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81

Clustered Index Scan (Clustered)

Abfrage 3: Abfragekosten (in Relation zum Batch): 100 %

SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81

Clustered Index Scan (Clustered)

DEMO

Datei: 11 - Parameter Sniffing (demo).sql

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

SPILL to tempdb - Lösungen

- Generelle Zuweisung von mehr Arbeitsspeicher für Ausführung von Abfragen
 - Minimum = 1 MB
 - Kann mit sp_configure 'Min Memory per Query' angepasst werden
- Überlistung des Query Optimizers durch "künstliche" Vergrößerung der Zeilengröße
- Verwendung von CTE um in "kleinen Häppchen" zu sortieren
- Erstellung eines Index für das Sortierkriterium

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Statistiken

- Statistiken werden von SQL Server verwendet, um optimale Ausführungspläne zu generieren
- Wenn Statistiken veraltet sind, kann sich die Ausführung einer Abfragen dramatisch verschlechtern
- Informationen über Statistiken können mit DMV oder DBCC abgerufen werden
- sys.stats
- DBCC SHOW_STATISTICS
- Aktualisierung erfolgt mit UPDATE STATISTICS

Strategien für effiziente IndizierungUwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Statistiken

Abfrage 1: Abfragekosten (in Relation zum Batch): 42 %
SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81

```
graph TD
    A[SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81] --> B[Hashed Loop Join]
    B --> C[Index Seek [NonClustered] [tbl_master].[tbl_master_c3]]
    B --> D[Schlüsselwerte (Clustered) [tbl_master].[tbl_master_id]]
```

Abfrage 2: Abfragekosten (in Relation zum Batch): 58 %
SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81

```
graph TD
    A[SELECT * FROM [dbo].[tbl_master] WHERE [c3]=81] --> B[Clustered Index Scan (Clustered) [tbl_master].[tbl_master_id]]
```

DEMO

Datei: 11 - Parameter Sniffing (demo).sql

Strategien für effiziente IndizierungUwe Ricken

SNEK3, März 2014

8

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Statistiken - Arbeitshinweise

- Genereller Schwellenwert sind 500 Datensätze und 20% der Datensätze im Index wurden aktualisiert.
- Dieser Schwellenwert ist nicht immer ideal. In diesem Fall müssen Statistiken eventuell manuell aktualisiert werden.
- Bei sehr großen Datenmengen bietet sich die Verwendung von TF2371 an! (<http://tinyurl.com/3jtvmt2>)
- Bei einem Index-Rebuild werden Statistiken automatisch aktualisiert!

Strategien für effiziente Indizierung

Uwe Ricken

SNEK22.-23.3.2014 Nürnberg
SQL Server und .NET-Entwickler-Konferenz

Fragen???

Herzlichen Dank für Eure Aufmerksamkeit
Für weitere Gespräche stehe nach der Session zur Verfügung

Strategien für effiziente Indizierung

Uwe Ricken
