PASS Regionalgruppentreffen

powered by PASS Deutschland e.V.

PASS Microsoft Data Platform Community

DEUTSCHLAND e.V.

RG RUHRGEBIET

SQL Server 2017

Das Performance Perpettum Mobile

Frank Geisler

Frank Geisler

CEO
GDS Business Intelligence GmbH

Chapter Leader PASS Chapter Ruhrgebiet

Microsoft P-TSP

MVP Data Platform

Author

Speaker

Topics:
Business Intelligence, SharePoint, SQL Server
Programming, Software Engineering

frank_geisler@geislers.net

redgate Friend of Redgate 2015

MVP Microsoft Most Valuable Professional

Agenda

- Performance Probleme bei der Abfrageausführung

- Adaptive Abfrageverarbeitung
  - Interleaved Execution for MSTFVs
  - Batch Mode Memory Grant Feedback
  - Batch Mode Adaptive Joins

- Automatisches Tuning
  - Query Plan Regression
  - Query Store
  - Automatisches Tuning

PASS Regionalgruppentreffen

powered by PASS Deutschland e.V.

PASS Microsoft Data Platform Community

DEUTSCHLAND e.V.

RG RUHRGEBIET

Performance Probleme bei der Abfrageausführung

Abfrageausführung und Kardinalitätsschätzungen

Während der Optimierung der Abfrage ist der Prozess zur **Abschätzung der Kardinalität** dafür zuständig für jeden Schritt in der Abfrageverarbeitung die Kardinalität zu schätzen.

Kardinalitätsabschätzung verwendet **statistische Methoden** und **Annahmen**

Ist die Schätzung gut genug können **gute Entscheidungen** bzgl. der **Reihenfolge** der Operationen und der Auswahl der **physischen Algorithmen** getroffen werden.

Gründe für fehlerhafte Schätzungen

Fehlende Statistiken

Veraltete Statistiken

Ungenügende Sampling-Rate

Schlechtes Parameter Sniffing

Abfragen außerhalb des Datenmodells

- z.B. MSTVFs, Tabellen Variablen, XQuery

Annahmen treffen nicht auf die abgefragten Daten zu

- z.B. Unabhängigkeit vs. Abhängigkeit

Kosten falscher Schätzungen

Langsame Ausführungszeit wegen ungenügender Abfragepläne

Exessive Ressoucennutzung (CPU, Speicher, IO)

Auslagerung auf Disk

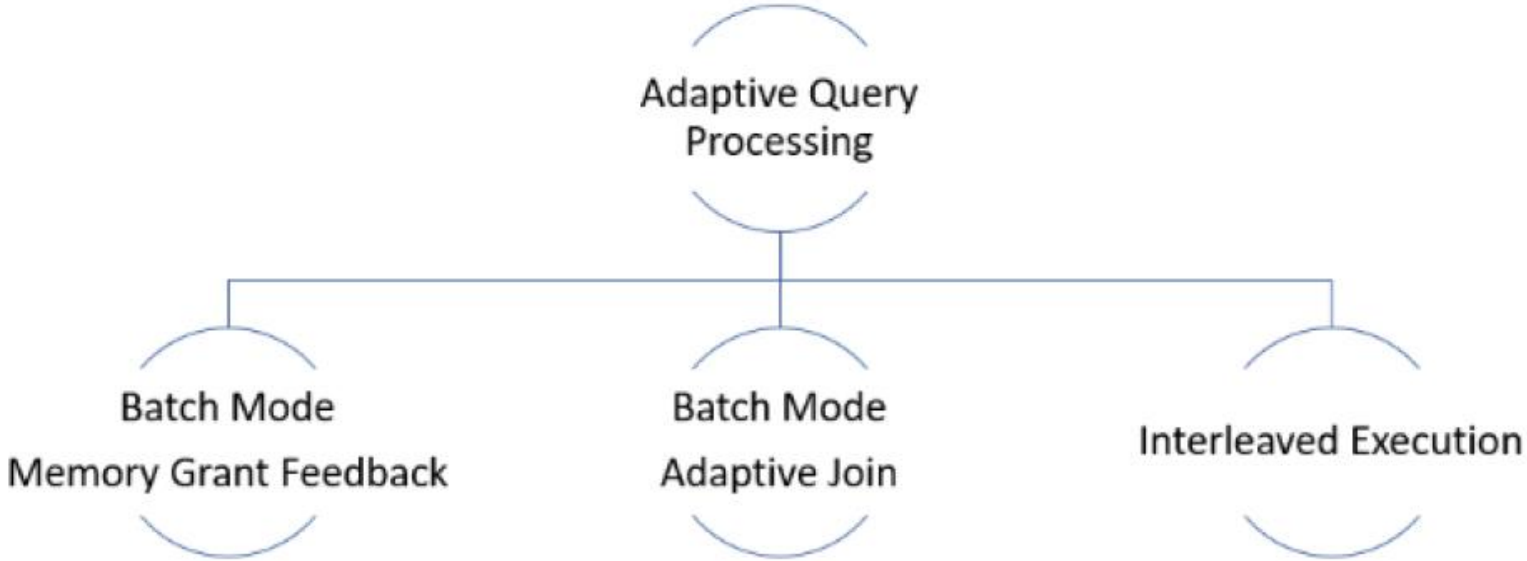
Reduzierter Durchsatz und Parallelverarbeitung

T-SQL Refactoring um Off-Model Ausdrücke zu umgehen

Adaptive Abfragenverarbeitung

Adaptive Abfragenverarbeitung

Idee: Automatische Performance Optimierung im SQL Server

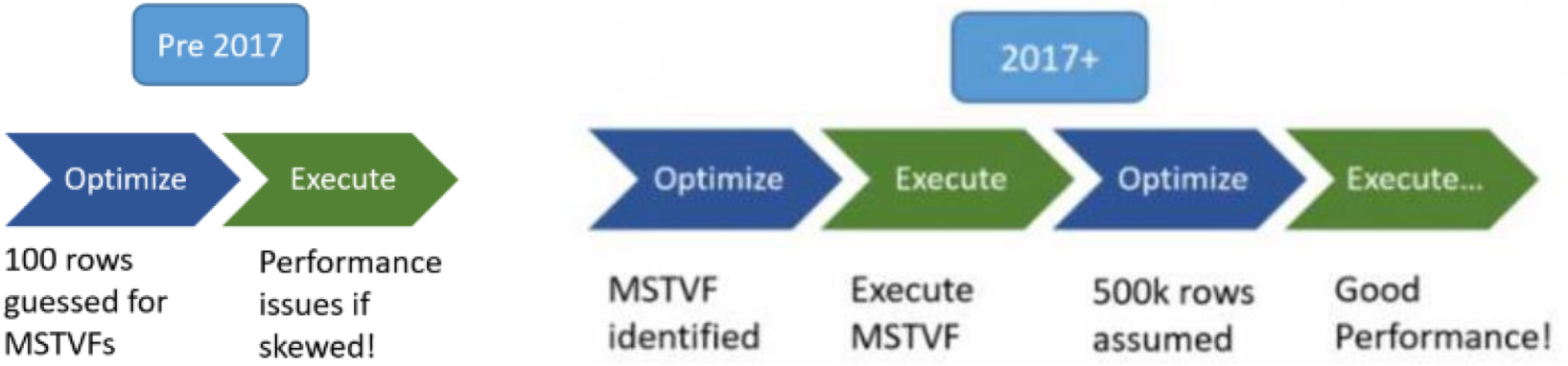


Adaptive Abfragenverarbeitung

- Drei unterschiedliche Methoden der adaptiven Abfragenapassung
  - Interleaved Execution for MSTVFs
  - Batch Mode Memory Grant Feedback
  - Batch mode adaptive joins

- Aktivieren der Adaptiven Abfrageanpassung:
  - `ALTER DATABASE [<database name>] SET COMPATIBILITY_LEVEL = 140;`

Interleaved Execution for MSTVFs



Extended Events

- interleaved_exec_stats_update
- interleaved_exec_status
- interleaved_exec_disabled_reason

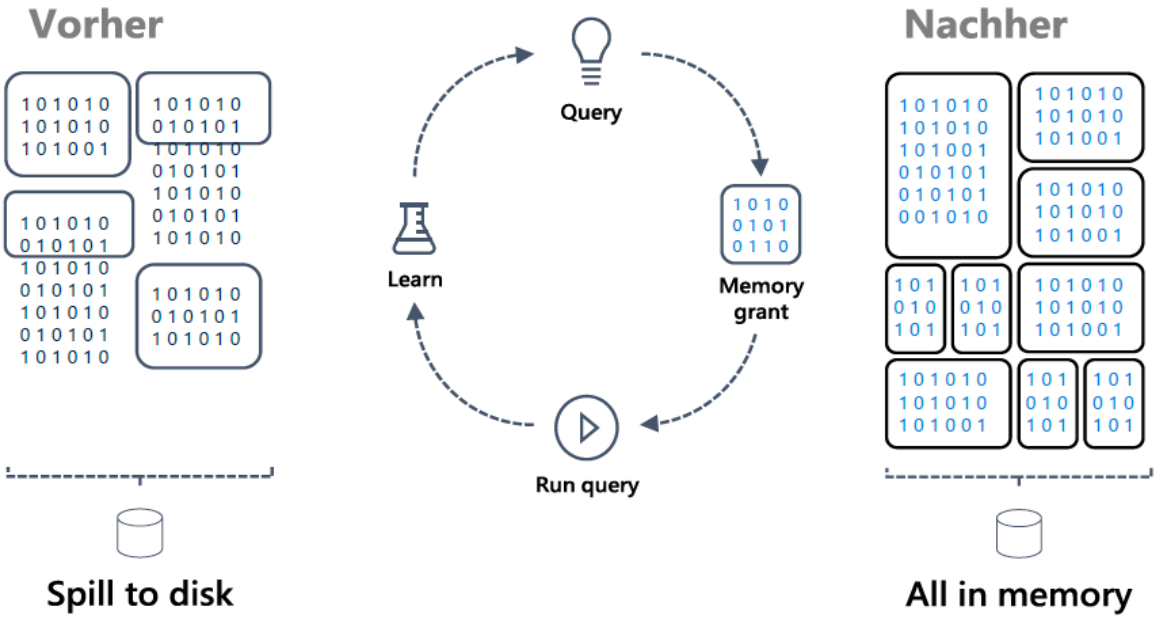
Überblick Interleaved Execution

Erwarteter Overhead?	• Minimal, da MSTVFs bereits materialisiert werden.
Gecacheter Plan	• Die erste Ausführung wird gecached und dann bei weiteren Ausführungen wiederverwendet
Plan Attribute	• Enthält Interleaved Execution Candidates • Is Interleaved Executed
Xevents	• Ausführungsstatus status, CE Aktualisierung, Grund warum nicht verwendet

Batch Mode Memory Grant Feedback

- Query Post-Execution Plan enthält minimale und ideale Speichergröße für Abfrageausführung
- Performance ist nicht gut wenn Memory Grant nicht passt
- Vor SQL 2017:
  - Schätzung der Kardinalität
  - Schätzung des minimalen und optimalen Speichers
- Probleme mit Kardinalität → Problem mit Speicherschätzung (Statistiken!)
- In SQL 2017:
  - Batch Mode Memory Grant feedback ermittelt wie viel Speicher wirklich benötigt wurde und aktualisiert Abfrageplan

Batch Mode Memory Grant Feedback



Überblick Batch Mode Memory Grant Feedback

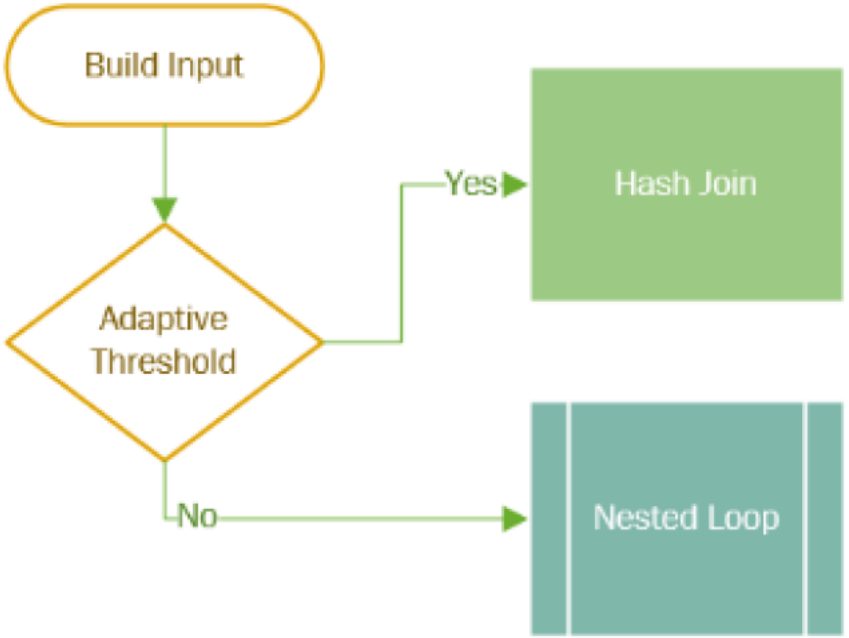
Erwarteter Overhead?	•Wenn es Oszillierende Werte gibt wird Batch Mode Memory Grant Feedback deaktiviert.
Gecacheter Plan	•Erste Ausführung wird überwacht und dann bei weiteren Ausführungen werden die Kardinalitätswerte der ersten Ausfügung verwendet
Xevents	•Spill report, und updates by feedback
Erwartete Größe?	•Bei zu wenig Speicher – Spill Größe plus Buffer. •Bei zu viel Speicher – weniger Speicherverschwendung.
RECOMPILE Szenarien	•Memory grant Größe geht wieder auf Originalwert zurück.

Batch Mode adaptive Joins

- Adapive Join Strategie für
  - Nested Loops
  - Hash Joins

- Merge Join wird momentan nicht unterstützt

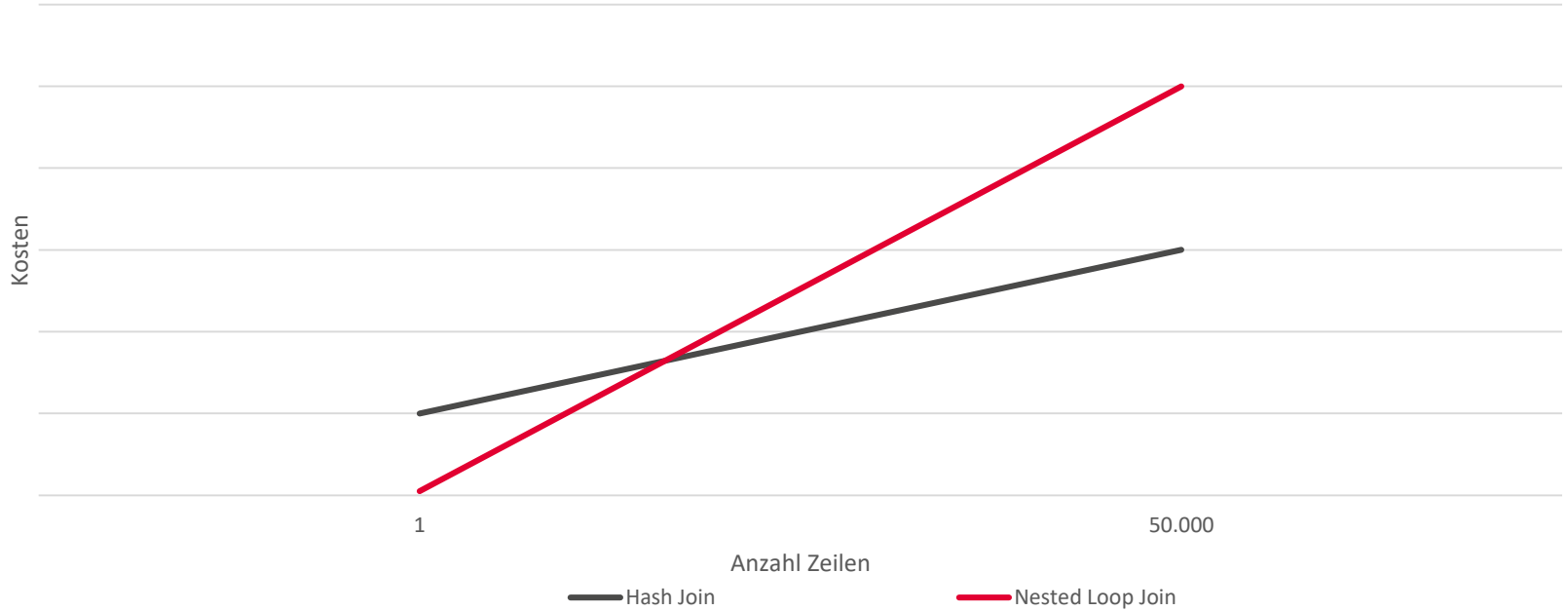
Batch mode adaptive joins



Große Datenmenge

Kleine Datenmenge

Adaptive Join Schwellwert



Überblick Batch Mode Adaptive Join

Erwarteter Overhead?	• Memory wird auch für NL (Nested Loop) Szenarien reserviert. Wenn NL immer optimal ist hat man mehr overhead
Plan Attribute	• Adaptive Threshold Rows, Estimated und Actual Join Type
Xevents	• Adaptive join skipped
Gecacheter Plan	• Ein einzelner gecachter Plan kann sowohl mit Szenarien mit wenigen als auch mit vielen Datensätzen umgehen.

Demo

Adaptive Query Processing

Automatisches Tuning

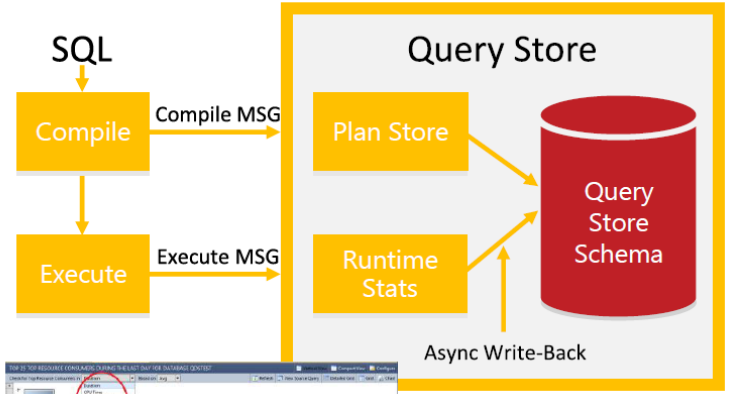
Das Problem: Query Plan Regression

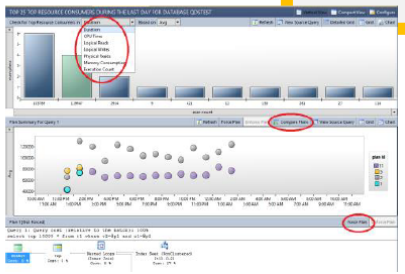
- SQL Server verwendet nach einiger Zeit einen suboptimalen Abfrageplan
- Symptome: Abfrage läuft erst schnell und dann plötzlich ohne Änderungen langsamer
- Beispiel - Sehr unterschiedliche Rückgabedatenmengen:
  - Tabelle hat B-Tree Index und Columnstore-Index
  - Abfrage 1 mit einem Parameter liefert ein Datensatz → optimal B-Tree Index
  - Abfrage 2 mit einem anderen Parameter liefert 1 Mio. Datensätze → optimal Columnstore-Index
  - Wird Plan gecached wird auf für Abfrage 2 B-Tree genutzt → **Schlecht**

SQL Server Query Store (ab SQL Server 2016)

- „Flugschreiber“ für die Datenbank
- Erleichtert die Analyse
  - Sammelt Abfragepläne
  - Sammelt Performance-Metriken
  - Ab SQL Server 2017 auch Wait Statistiken

- Hilfreich für:
  - Langsam gewordene Abfragen
  - Top-N Queries (Zeit, CPU, Speicher…)
  - Welche Abfragen warten am längsten?





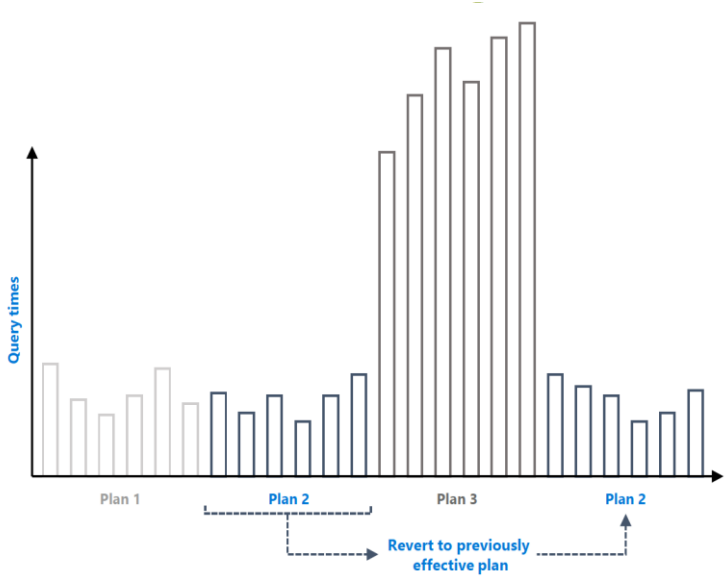
Works across restarts / recompiles

SQL Server 2017 – Automatisches Tuning

- Automatische Plankorrektur
- Erkennen und korrigieren von „schlechten“ Abfrageplanänderungen
- Basis dafür ist der Query Store
- Automatisches Erzwingen des letzten „guten“ Abfrageplans

DMVs:
- sys.dm_db_tuning_recommendations

```
ALTER DATABASE CURRENT SET AUTOMATIC_TUNING
(FORCE_LAST_GOOD_PLAN = ON);
```



Demo

Automatisches Tuning

Ressourcen

- Introducing Interleaved Execution for Multi Statement Table valued Functions
- Introducing Batch Mode Adaptive Memory Grant Feedback
- Introducing Batch Mode Adaptive Joins
- You Shall not Regress – How SQL Server 2017 prevents plan regression
- SQL Server 2017 Demos auf GitHub

Vielen Dank für die Aufmerksamkeit!