Query Designing for Performance

Pinal Dave

http://blog.sqlauthority.com

Vinod Kumar M

http://blogs.extremeexperts.com

pluralsight
hardcore developer training

Writing Query is more ...complex than building Web

Getting Started

- Exists vs IN vs Joins
- NOT Exists vs NOT IN vs NOT Joins
- Avoid Select *
- Subquery vs CTE
- CTE vs Temp Variable
- Order of the Table in Join
- Hints with Joins
- Execution Plan Cache
- Parameter Sniffing and Plan Guide
- Dynamic Queries
- Execution Plan for Scalar UDF
- Dis/Advantages of Views

Exists vs IN vs Joins

- Comparison
  - IN
  - Exists
  - JOIN
- IN and EXISTS gives mostly same result and performance
- JOIN may not send the same results as IN or EXIST clause

*** 80% - 20% rule**

NOT Exists vs NOT IN vs NOT Joins

- Comparison
  - IN
  - Exists
  - JOIN
- EXISTS often gives better performance
- JOIN may not send the same results as IN or EXIST clause

*** 80% - 20% rule**

Avoid Select *

- Retrieves unnecessary data data
  - **Increase network traffic**
- Defaults to Clustered Index usage
  - **May not use optimal other index**
- Application may break as column order changes
  - **Issues when used in Views**

*** 80% - 20% rule**

Subquery vs CTE

- With respect to performance **No Difference**
- CTE Provides readability and encapsulation
- CTE can be used in recursively

*** 80% - 20% rule**

CTE vs Temp Variable

- It is Apples and Oranges comparison
- They are different and have different use

*** 80% - 20% rule**

Order of the Table in Join

- Inner Join
  - Order does not matter
- Outer Join
  - Order matters

*** 80% - 20% rule**

Hints with Joins

- Careful with table Hints
- Table hint has impact on performance

*** 80% - 20% rule**

Execution Plan Cache

- Optimizer caches the execution plan of the query when it executes first time
- Cache execution plans improves the performance (in most cases)

*** 80% - 20% rule**

Parameter Sniffing and Plan Guide

- Query Hints
  - Optimize for Unknown
- Plan Guide
  - Intended where user have no control over the input T-SQL script

*** 80% - 20% rule**

Dynamic Queries

- Try to use Static SQL as much as possible
- Unavoidable, then use D-SQL
- Prepare, Parameterize and then execute
- Use **sp_executesql** command

- *PS*: D-SQL even inside SP doesn't influence performance

*** 80% - 20% rule**

Execution Plan for Scalar UDF

- Scalar UDF hides the execution plan of function
- Scalar UDF ***may*** take more CPU power
  - Looping over table rows
  - Ignores optimizer query re-write

*** 80% - 20% rule**

Dis/Advantages of Views

- Avoid unnecessary usages of Views
- Use View with aggregate functions
- Index Views have special usages

*** 80% - 20% rule**

Summary

- Exists vs IN vs Joins
- NOT Exists vs NOT IN vs NOT Joins
- Avoid Select *
- Subquery vs CTE
- CTE vs Temp Variable
- Order of the Table in Join
- Hints with Joins
- Execution Plan Cache
- Parameter Sniffing and Plan Guide
- Dynamic Queries
- Execution Plan for Scalar UDF
- Dis/Advantages of Views

Remember: SQL Server Optimizer usually opts for most efficient execution plan.

Remember: 80%-20% Rule. There are always special cases.