

# Tips & Tricks for Writing Better Queries



Joe Webb  
Microsoft SQL Server MVP  
[joew@webbtechsolutions.com](mailto:joew@webbtechsolutions.com)



## A bit of background

### Joe Webb

- Started in IT in 1993 after receiving a BSEE & MBA from Auburn University
- Founded WebbTech Solutions in 1996
  - Providing consulting, mentoring, & technical training
- Began working with SQL Server in 1996 with version 6.0
- 5-time recipient of the Microsoft MVP Award for SQL Server
- Author of 3 books
- Served 6 years on the Board of Directors for PASS ([sqlpass.org](http://sqlpass.org))
- Spoken at over 20 conferences in North America & Europe

## Agenda

### The Basics

- What is Better?
- Tools of the Trade
- Overview of Execution Plans

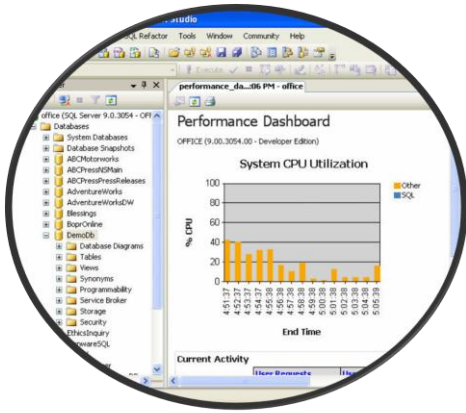
### Accessing Database Objects

- Improving CRUD operations
- Do's and Don'ts
- Tips and Tricks

## What is "better"?

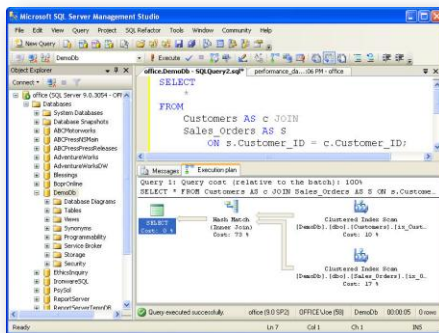


## Tools of the Trade



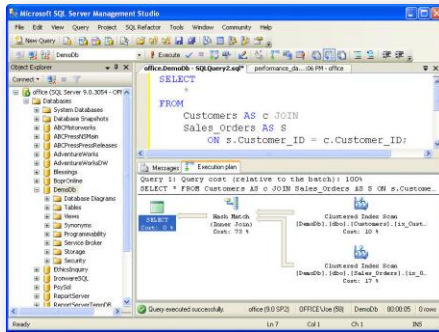
- Performance Monitor
- SQL Profiler
- Performance Dashboard
- Query Execution Plans
- Client Statistics

## Execution Plans



- Provide graphical, text-based, or XML-based representation for the data retrieval methods chosen by the Query Optimizer
- Execution costs for specific statements are displayed

## Understanding Execution Plans



- Estimated & Actual Execution Plan
- Displays cost of each physical operation
- Read from right to left, top to bottom

## Understanding Execution Plans

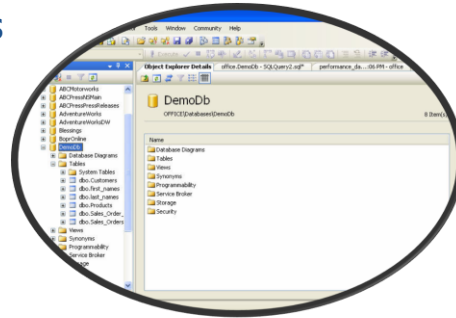
Clustered Index Scan	
Scanning a clustered index, entirely or only a range.	
<b>Physical Operation</b>	Clustered Index Scan
<b>Logical Operation</b>	Clustered Index Scan
<b>Estimated I/O Cost</b>	5.5372
<b>Estimated CPU Cost</b>	1.10016
<b>Estimated Operator Cost</b>	6.63736 (10%)
<b>Estimated Subtree Cost</b>	6.63736
<b>Estimated Number of Rows</b>	1000000
<b>Estimated Row Size</b>	64 B
<b>Ordered</b>	False
<b>Node ID</b>	1
<b>Object</b>	
[DemoDb].[dbo].[Customers].[ix_Customer_ID][c]	
<b>Output List</b>	
[DemoDb].[dbo].[Customers].Customer_ID, [DemoDb].[dbo].[Customers].Last_Name, [DemoDb].[dbo].[Customers].First_Name, [DemoDb].[dbo].[Customers].Email_Address	

- Physical operation
- Logical operation
- I/O cost
- CPU cost
- Rows
- Row size



## Accessing Database Objects

- Temporary Tables
- User Defined Functions
- Cursors
- Stored Procedures
- Joining tables
- Limiting searches
- Transactions
- Triggers
- Indexes



## Temporary Objects

- Temporary tables frequently add Disk I/O to tempdb
- Created as a heap by default. Consider indexing to improve performance
- Consider using other techniques instead
- If you use them:
  - Create using DDL rather than SELECT INTO
  - Create all temporary objects at the start of a procedure to avoid recompiles
  - Create temporary objects outside of transactions
  - Don't compound the problem by creating Cursors on temporary tables



## Functions

### User-Defined Functions

- Can behave like a Cursor or correlated subquery– be careful!
- Avoid on large result sets

### Built-in Functions

- UPPER() and LOWER() are generally not needed
- Placing columns inside of functions within the WHERE clause precludes the use of indexes



## Cursors

- AVOID!!!
- AVOID!!!
- AVOID!!!
- If you think they are required, try harder to rewrite!
- If absolutely required, use FAST-FORWARD READ-ONLY





## Stored Procedures

- Use them
- SET NOCOUNT ON
- Validate parameters early
- Return only the columns required
- Consider OUTPUT parameters for single row results
- Be mindful of widely varying parameter inputs
  - Can lead to inefficient plan reuse
- Monitor for recompiles and cache misses



## Joining Tables

### Columns

- Join on indexed columns
- Joins on numeric values usually out perform joins on character values
- Beware of implicit type conversions

### Join types

- Keep result sets small with WHERE clause and limited columns
- Frequently better than correlated sub-queries; test!
- Be wary of Joins hints



## Limiting Searches

- Operators in the WHERE clause make a big difference
  - Avoid <> and LIKE
  - Use EXISTS, NOT EXISTS, and LEFT JOINS instead of NOT IN
- If you must use LIKE, make the first character a literal
- ORs can prevent the use of proper indexes
  - Use Index hints or UNIONS
- Don't place columns inside functions!



## Transactions

- Keep Transactions as short as possible
- Avoid excessively large batches
- Consider indexes to prevent table locks
- Keep reads to a minimum during a Transaction
- Judiciously use nested Transactions



## Triggers

- Consider the use of Triggers
  - Declarative Referential Integrity usually out performs integrity implement through Triggers
  - Constraints typically out perform data validation implemented through Triggers
- Catch errors before entering a Trigger
- Order Triggers such that the one most likely to cause a Rollback is fired first
- Use UPDATE() and COLUMNS\_UPDATED() early in the Trigger
- Don't make time intensive calls within triggers



## Indexes

- Make sure you strike a good balance for indexing
  - Too many adversely affects data input
  - Too few adversely affects reads
- Order of columns in an index matters!
- Covering indexes are the fastest query possible



## Other Tips

- Despite what you see in examples, avoid `SELECT * FROM My_Table`
- Consider the impact of duplicate rows
  - Avoid `SELECT DISTINCT` if it doesn't cause a problem
  - Use `UNION ALL` instead of `UNION` if duplicate rows are acceptable
- Use 2 part names when referencing objects



## More Tips

- Use `sp_executesql` rather than `execute` for dynamic SQL
- Consider using `READ UNCOMMITTED` if “dirty reads” are acceptable
- Don't use `ORDER BY` unless you really need it
- Be verbose with comments!



## Additional Resources

- <http://blogs.msdn.com/sqlqueryprocessing/default.aspx>
- <http://www.sql-server-performance.com>
- <http://www.sqlteam.com>
- <http://www.sqlservercentral.com>
- <http://www.sqlblog.com>
- <http://www.sqlpass.org/>
- Inside SQL Server 2005 Series



## Questions?

Thank you!

Contact/follow me:

Email: [joew@webbtechsolutions.com](mailto:joew@webbtechsolutions.com)

Blog: <http://www.webbtechsolutions.com/blog>

Twitter: <http://www.twitter.com/joewebb>

Rate this session: <http://speakerrate.com/joewebb>