BRENT OZAR
UNLIMITED

Watch Brent Tune Queries

BRENT OZAR DOUG LANE JES BORLAND KENDRA LITTLE JEREMIAH PESCHKA

Microsoft CERTIFIED Master SQL Server 2008

Microsoft CERTIFIED Professional

Microsoft CERTIFIED Professional

Microsoft CERTIFIED Master SQL Server 2008

Microsoft CERTIFIED IT Professional
Database Developer 2008
Database Developer
Database Administrator 2008

MVP Microsoft Most Valuable Professional x 4

Microsoft Partner
Silver Data Platform

amazon web services | Partner Network

BrentOzar.com/go/tunequeries

Be Creepy.

Blitz first for obvious problems

End user requirements gathering

Capture query metrics

Read the metrics and plan

Experiment with the query cost

Execution plan review

Parallelism opportunities

Index improvements

Blitz the box first

sp_Blitz™ for SQL Server
setting that's been changed
and influences the plan

sp_BlitzIndex™ looking for
disabled indexes, heaps,
obvious missing indexes

Make sure the query
doesn't do writes
(or if it does, tune in dev)

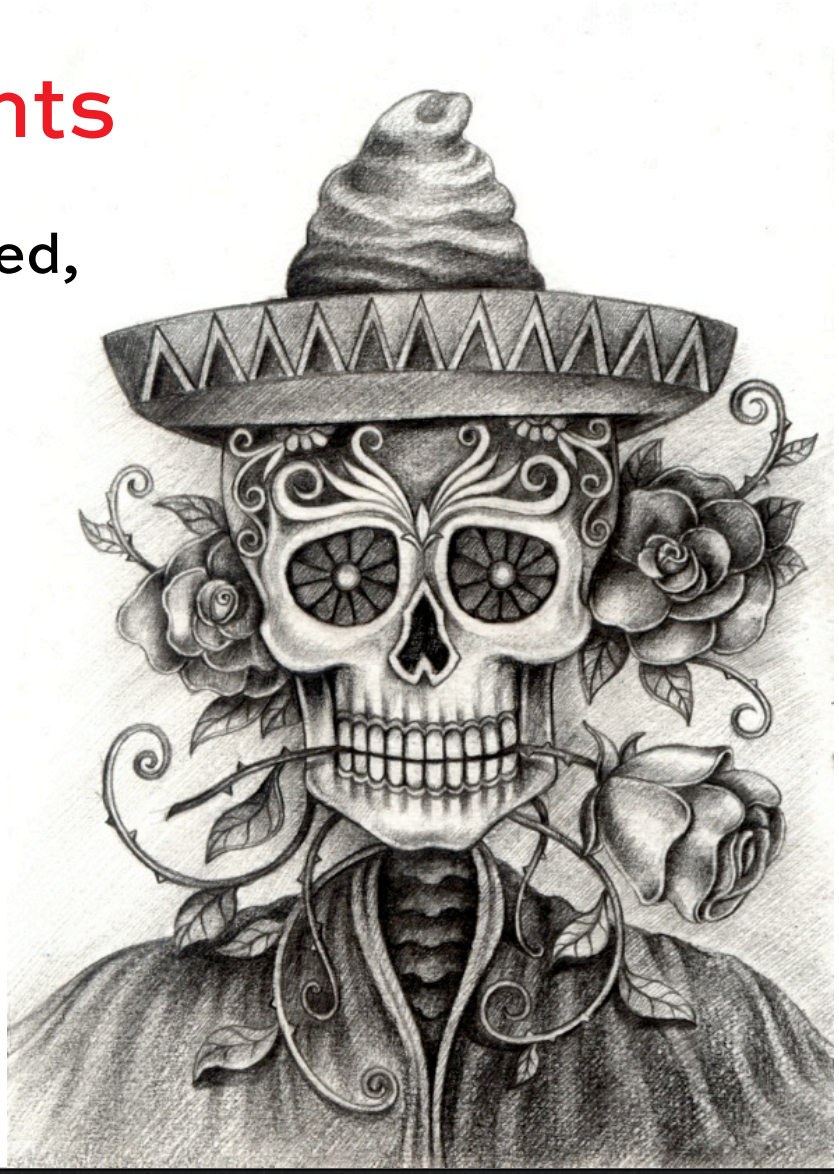
End user requirements

Find out if it's machine-generated, inline, dynamic, or stored proc

Define your finish line

Can we run it less often, or run it somewhere else?

Get the business purpose of the query output



Capture query metrics

Run the query with your SSMS tuning settings on, save the metadata

Make sure you've got the right query and the right plan

Make sure it's not a parameter sniffing problem

Start a separate window to compare before/after, and iterate there

Read metrics, plan

Identify the logical reads,
CPU time, duration,
query cost

What's the biggest problem:
reads, CPU, or duration?

Does the query's duration/
reads/CPU match up with the
amount of work it's doing?

Experiment with query cost

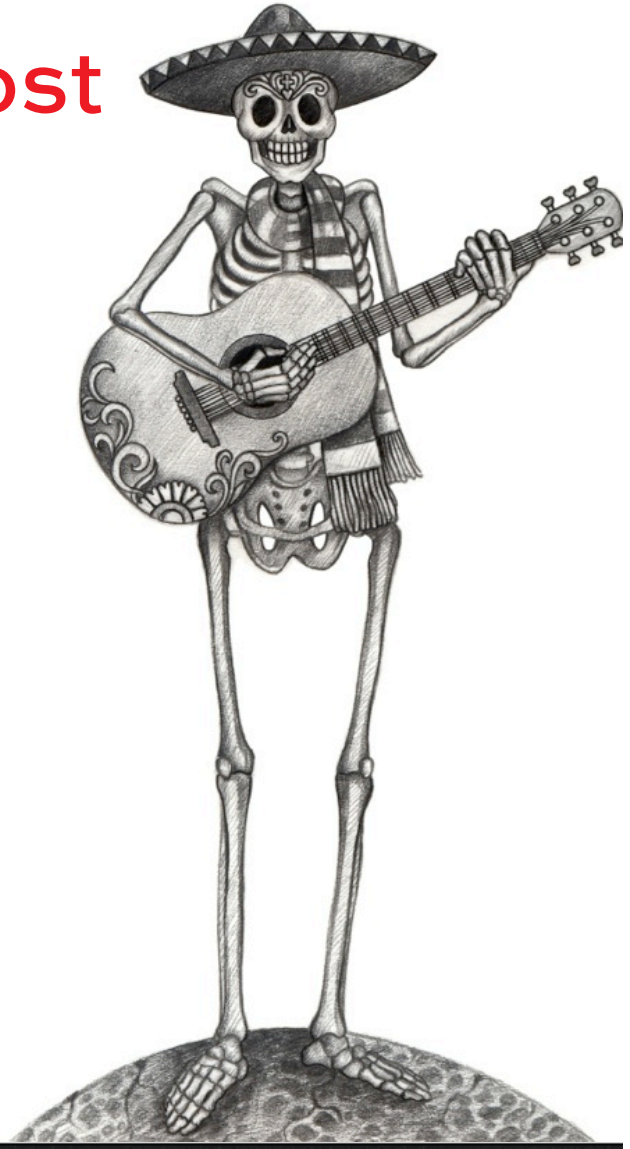
Remove the ORDER BY

Change the list of fields in
the SELECT to just be SELECT 1

Switch table variables to temp tables

Did the cost change take you
to the finish line? Start asking
tough questions.

For any non-INNER joins, make sure
they really need to be something else,
and if we need the data.



Execution plan review

Look at the plan's properties for Optimization Level.

If not full, what's the Reason for Early Termination?

Look at the top right operator.
Any implicit conversions or SARGability problems?

Are the estimated vs actual row counts way off?

Scan through rest's estimated vs actual

If est vs actual is off, are statistics
up to date on the underlying tables?

Stats still off? Split it into a query
inserting into a temp table, then a
bigger query joining to the temp table.

Are any functions involved?

Parallelism opportunities

Is the query going parallel,
and if so, is it benefitting?

If not, does the query
exceed the server's
Cost Threshold for Parallelism?

Does it have a long duration
(>1 second) that's matched
by identical CPU time?

Are there any parallelism
inhibitors in the query?
BrentOzar.com/go/serialudf

Index improvements

Key lookups? Can we widen an existing index into a covering index, or add one?

Is there an indexed view involved that SQL should use, but isn't?

Let's be creepy.

Blitz first for obvious problems

End user requirements gathering

Capture query metrics

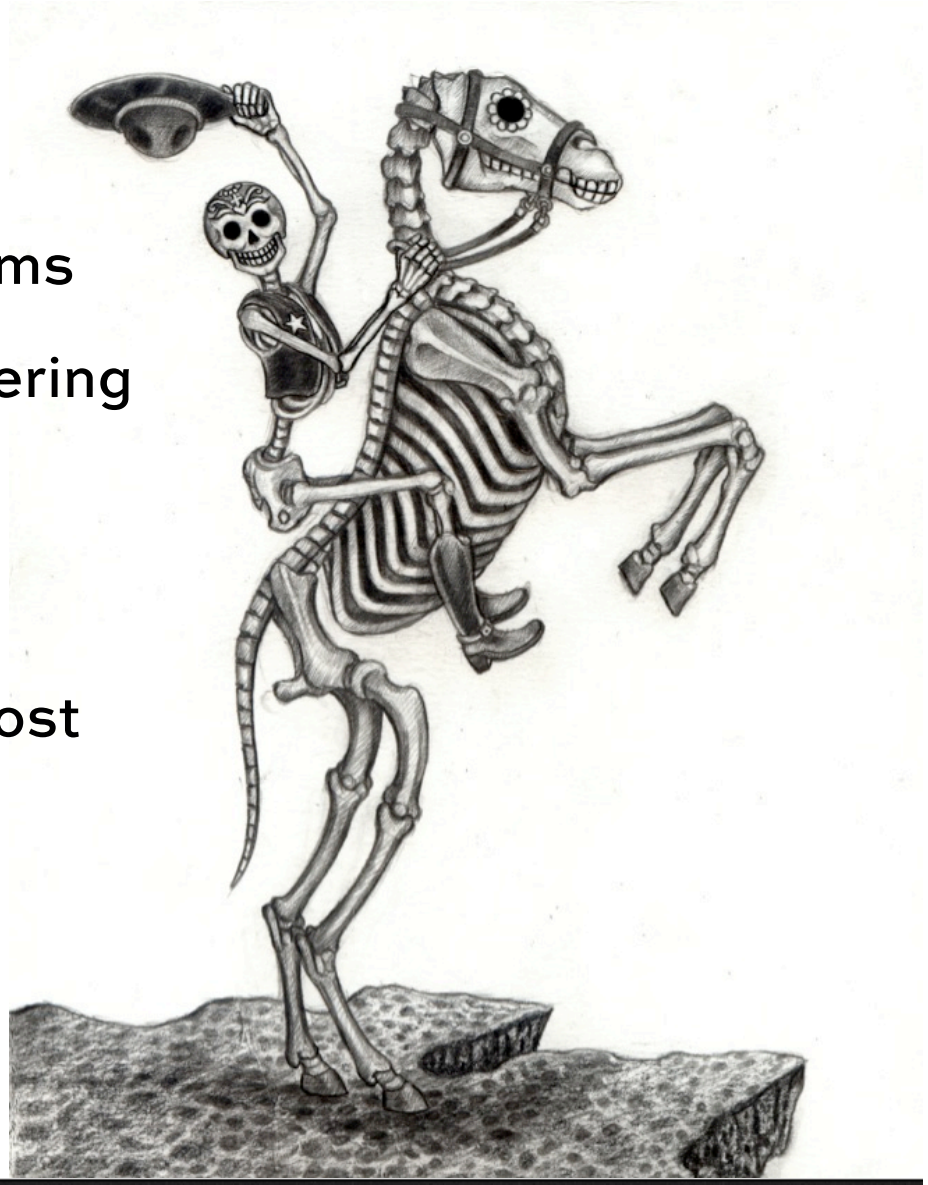
Read the metrics and plan

Experiment with the query cost

Execution plan review

Parallelism opportunities

Index improvements



BrentOzar.com/go/tunequeries