



BRENT OZAR
UNLIMITED®

Lab 2: Tuning Indexes for Queries

1.7 p1

My index tuning strategy

Just
once

Dedupe – reduce overlapping indexes

Eliminate – unused indexes

Weekly
for 1
month

Add – badly needed missing indexes

Do this only
AFTER the easy
stuff above

Tune – indexes for specific queries

Heaps – usually need clustered indexes

You usually only have to Dedupe/Eliminate once (maybe twice) to clean up your indexes, removing the dead weight.

Now it's safer to add the indexes your queries really need.

1.7 p2



Let's tackle the other extreme.

DropIndexes dropped all your nonclustered indexes.

The load test pushed a bunch of queries through, and this time that load test app went WAY slower.

Now you've got a bunch of missing index recommendations.



Now, add the right ones.

In this lab, your goal is to:

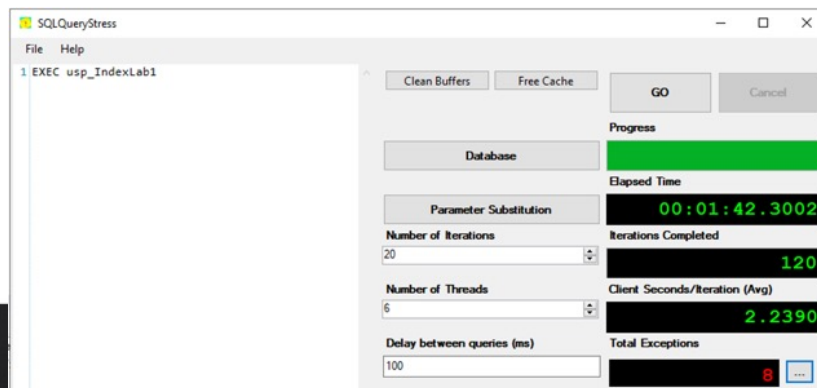
1. Analyze the plan cache with `sp_BlitzCache` (or your favorite script) then design helpful indexes
2. Script out changes you want to make, aiming for:
 - 5 or less indexes per table
 - 5 or less fields per index
3. Tell me in Slack what indexes you designed and the queries you designed 'em for.



What to expect

The workload will finish faster: we're starting with no indexes, so every good index you add should help.

It's theoretically possible to run this load in 1-2 mins:



1.7 p5

Setting up for the lab

1. Restart your SQL Server service (just to clear all stats)
2. Restore your StackOverflow database (Agent job)
3. Copy & run the setup script for Lab 1
4. After the restore finishes, run:
EXEC DropIndexes;
(because this lab is about adding indexes)
5. Start SQLQueryStress with the lab #1 workload again:
 1. File Explorer, \Labs, SQLQueryStress.exe
 2. Click File, Open, \Labs\IndexLab1.json, Go



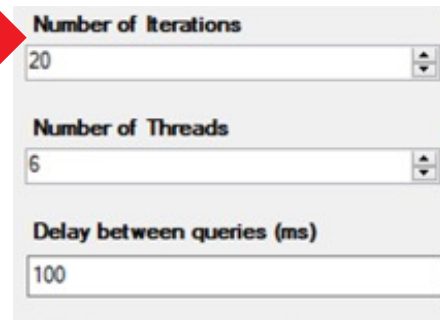
Doing the lab later?

Set this to 100000.



This makes sure your VM stays busy til you're ready.

For the success test, you'll be using the regular iterations to see if the test completes quickly enough.



Number of Iterations
20
Number of Threads
6
Delay between queries (ms)
100



How I'd budget this hour

5 minutes – poke around: run `sp_BlitzCache`, get a feel for your most resource-intensive queries.

10 minutes: pick one query to work on, design indexes to help it. But just like in real life, don't add them yet: just plan out the changes you want to make in an outage window.

Repeat that step 3-4 times, then apply your indexes.

Last 10-15 minutes: start the load test again, and see if your new indexes are getting used.



Stop your load generator now.

Click Cancel and close SQLQueryStress.

If it won't close, you may need to kill it and its queries.

(Make sure it doesn't show up in Task Manager either.)

