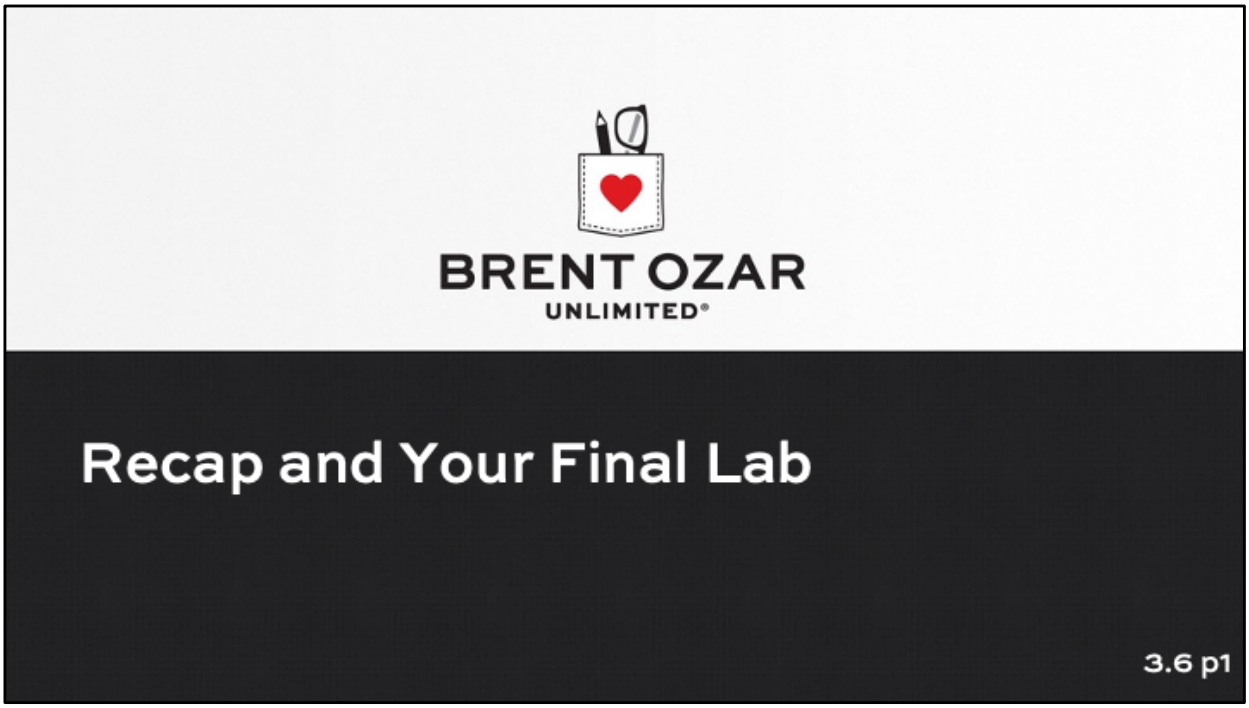


The first round of tuning is easy.

The first round of tuning tweaks is ***very*** effective:
you can make a huge difference in a couple of hours.

Subsequent rounds are harder,
and produce diminishing returns.

Work hard enough at tuning an application, and you
start running out of free/easy options.

Management needs to hear that message:
"We've already pushed all the easy buttons."

3.6 p2

The right nonclustered indexes...

1. Reduce PAGEIOLATCH waits because we can grab a few pages from a tiny in-memory index rather than scanning an entire table from disk.
2. Reduce blocking by:
    1. Letting us close transactions faster
    2. Helping us find rows we want to update faster

3.6 p3

The wrong nonclustered indexes

Slow down deletes, updates, and inserts because we have to lock and touch all these extra pages.

Reduce our memory effectiveness because we have to cache all these pages we don't really need (since we're touching them for DUIs.)

Slow down maintenance jobs: backups, checkdb, index rebuilds, stats updates.

3.6 p4

My index tuning goals

Dedupe near-identical indexes

Eliminate unused indexes

Add highly needed missing indexes

Tune resource-intensive queries

Heaps often need clustered indexes

3.6 p5

Use the D.E.A.T.H. method first.

Just once	**Dedupe** – reduce overlapping indexes with sp_BlitzIndex **Eliminate** – unused indexes with sp_BlitzIndex
Weekly for 1 month	**Add** – badly needed missing indexes with sp_BlitzIndex
Do this only AFTER the easy stuff above	**Tune** – indexes for specific queries with sp_BlitzCache **Heaps** – usually need clustered indexes (this is political)

3.6 p6

Tools are key.

Whether you use monitoring tools or DMV scripts:

- Know how to use the tool. Spend time reading the manual and experimenting with options.
- Practice, practice, practice. Don't take things for granted – run experiments to test yourself.
- Measure before & after.
  Prove that your change made a difference.

3.6 p7

FirstResponderKit.org

sp_BlitzIndex

- Index health in this one database
- Add @Mode = 4 for smaller tables, smaller issues

sp_BlitzCache

- Most resource-intensive plans in the cache
- @SortOrder = 'reads' shows which queries read the most data (and probably need indexes)

All open source.

3.6 p8

Changing things up

Drop all the nonclustered indexes before you start

Try different VM sizes:

- More or less virtual CPUs, memory
- Faster or slower storage

Try different workloads:

- 4, 8, 16, 64 threads in SQLQueryStress
- Shorter durations between queries

3.6 p9

Curious about new features?

Wonder about the impact of Query Store?

Would columnstore indexes help the workload?

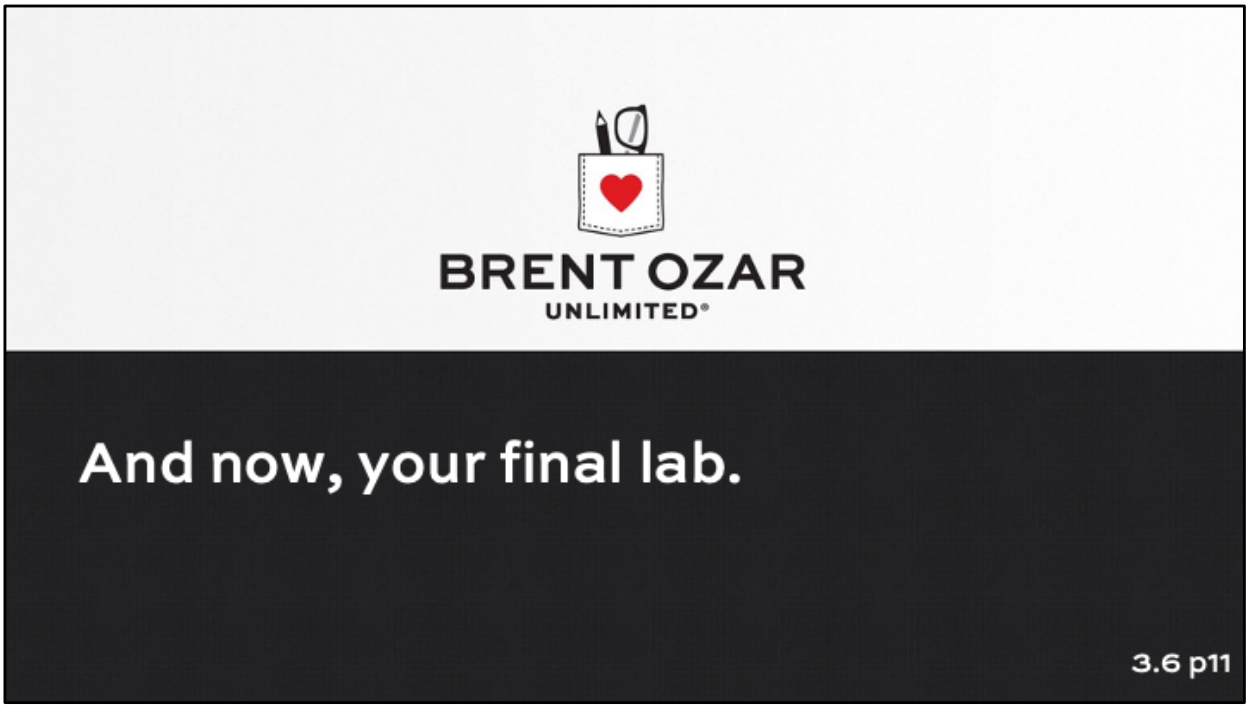
Will In-Memory OLTP (Hekaton) speed it up 100X?

How much would a synchronous AG slow it down?

How does the new Cardinality Estimator impact it?

Run experiments and find out! This is how you learn.

3.6 p10

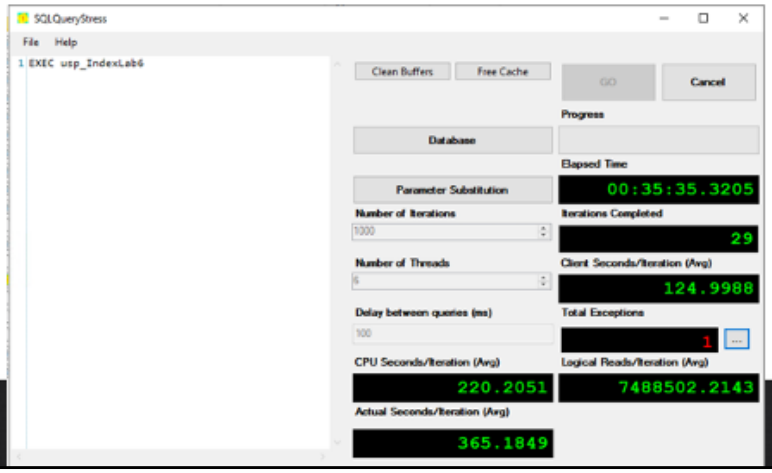


Setting up for the final lab

1. Restart SQL Server service (just to clear all stats)
2. Restore your StackOverflow database (Agent job)
3. Copy & run the setup script for Lab 6, takes 10 min.
4. Start SQLQueryStress with the lab #6 workload:
    1. File Explorer, \Labs, SQLQueryStress.exe
    2. File, Open, \Labs\IndexLab6.json, Go

Let it run for at least 30 minutes.

It's not going to be quick, but you want to populate the index DMVs so you can do the D.E.A. steps.



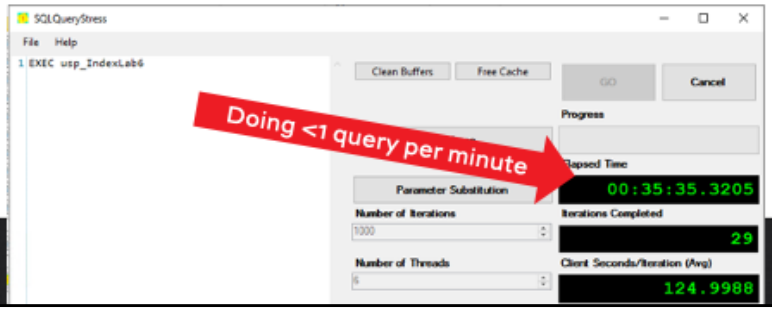
3.6 p13

After 30 minutes, stop it.

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.)

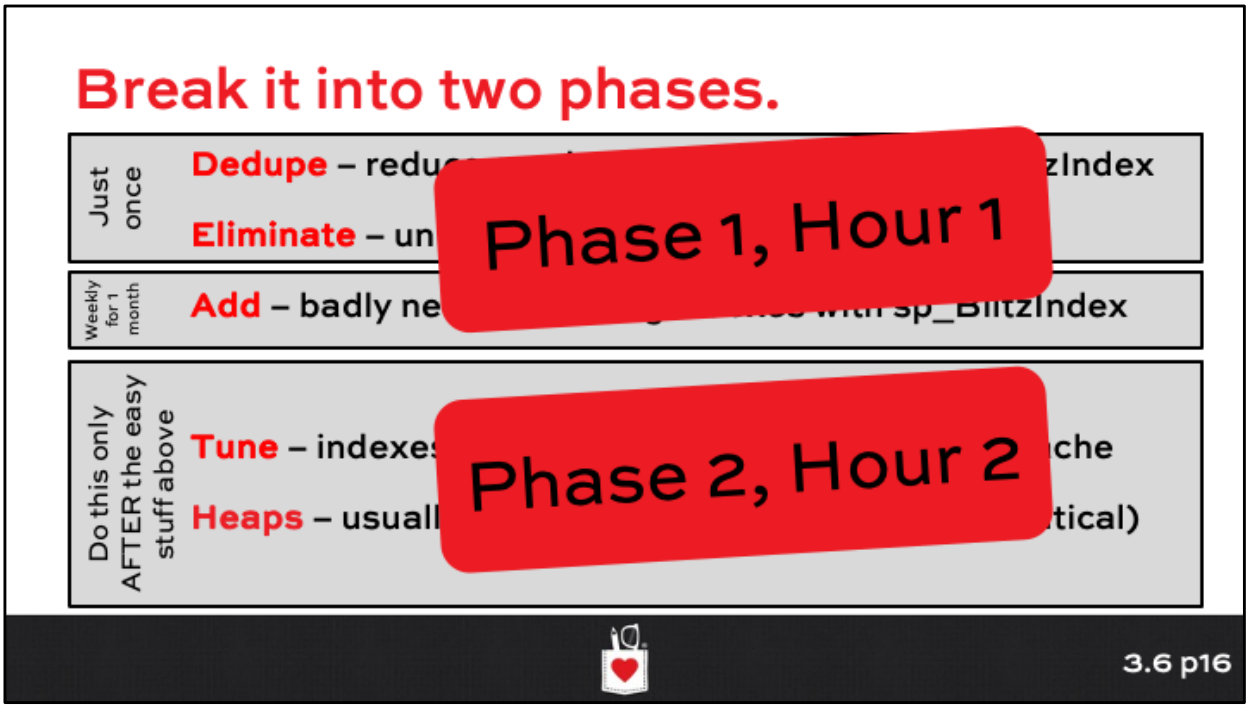


3.6 p14

Tackle as much of this as you can.

Just once	**Dedupe** – reduce overlapping indexes with sp_BlitzIndex **Eliminate** – unused indexes with sp_BlitzIndex
Weekly for 1 month	**Add** – badly needed missing indexes with sp_BlitzIndex
Do this only AFTER the easy stuff above	**Tune** – indexes for specific queries with sp_BlitzCache **Heaps** – usually need clustered indexes (this is political)

3.6 p15



How I'd budget hour 1

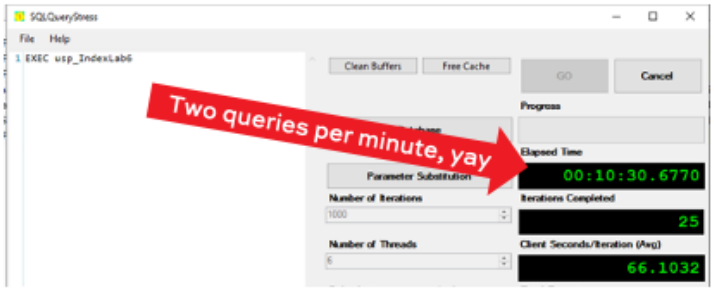
Run sp_BlitzIndex. For each table, do the D.E.A:

- Dedupe & eliminate to get rid of the dead weight
- Add indexes Clippy wants (but you interpret 'em)
- Aim for the 5 & 5 guideline
- It'll take 20-30 minutes to apply your changes

Restart SQL Server to clear DMVs, then run the load test again for another 30 minutes to populate the plan cache.

My SQLQueryStress after hour 1

Twice as fast, but still not amazing:



Time for T part of the D.E.A.T.H. Method...

3.6 p18

How I'd budget hour 2

Run sp_BlitzIndex again and see if you missed anything – sometimes indexes aren't used after all, or Clippy has other ideas. As your indexes create...

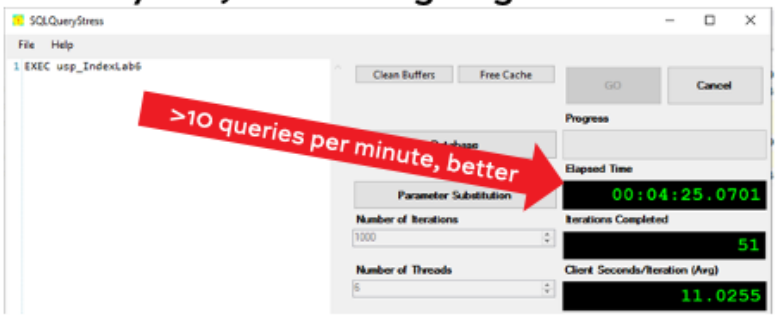
Do the T part:

- sp_BlitzCache @SortOrder = 'reads'
- See if you can hand-tune indexes for a couple of top queries, 30-45 minutes max

Kick off the load test one more time.

My SQLQueryStress after DEA #2

Awww yeah, continuing to get better:



Finally, doing the T part of D.E.A.T.H...

3.6 p20

In the T part...

You'll find queries where you wanna say,
"We should really change this query."

That's cool! You can!

That's also a good sign that you're ready for the next class in the rotation, Mastering Query Tuning.

After all, you can't fix every query with indexes.

3.6 p21

I'll stay on Slack for the hour.

You can start working on the lab now.

I'll monitor Slack for questions.

Want to watch me do it? Go watch
the Instant Replay videos now.

Your VMs will stay on for 3 hours.
(Need longer? Email help@brentozar.com.)

3.6 p22

You can do this.

For questions, leave comments on the relevant module.

For private help after the class, email Help@BrentOzar.com with:

- A note that you were in this class
- sp_Blitz @CheckServerInfo = 1
- sp_BlitzFirst @SinceStartup= 1