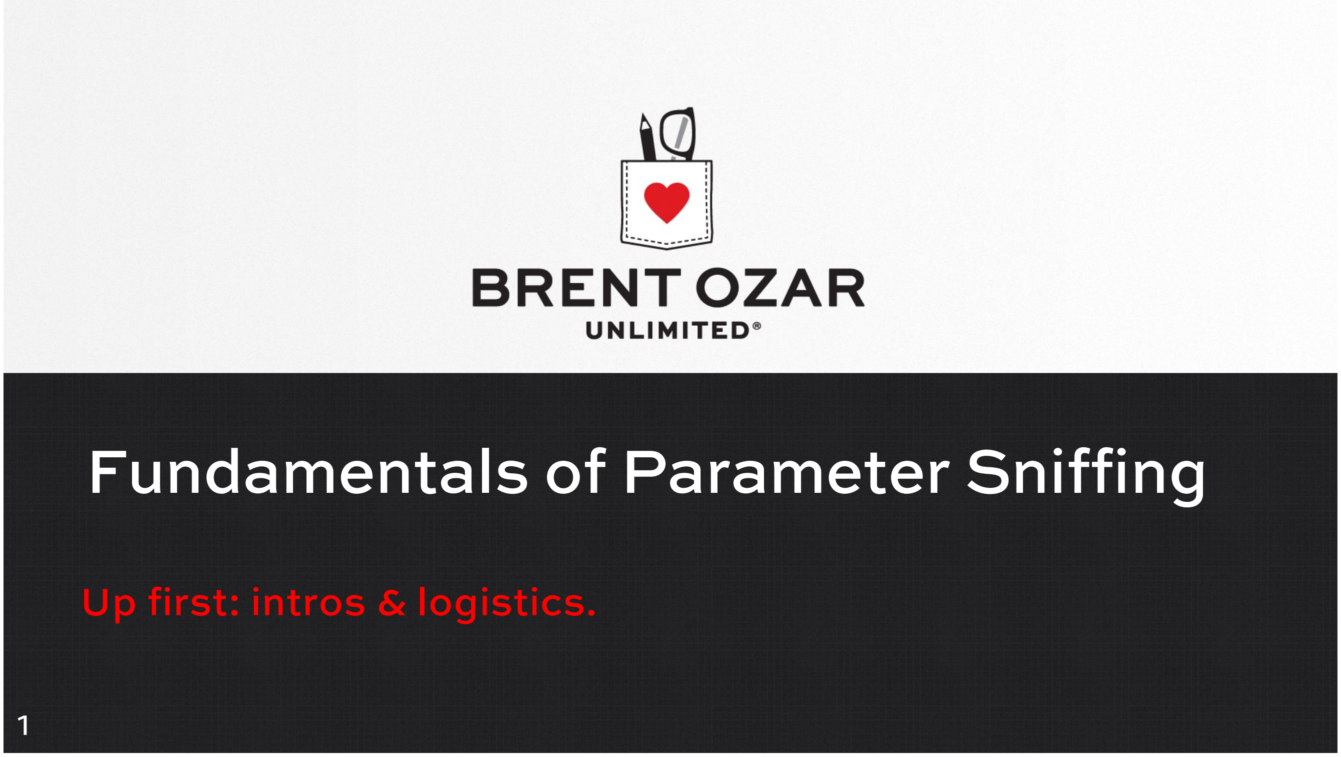
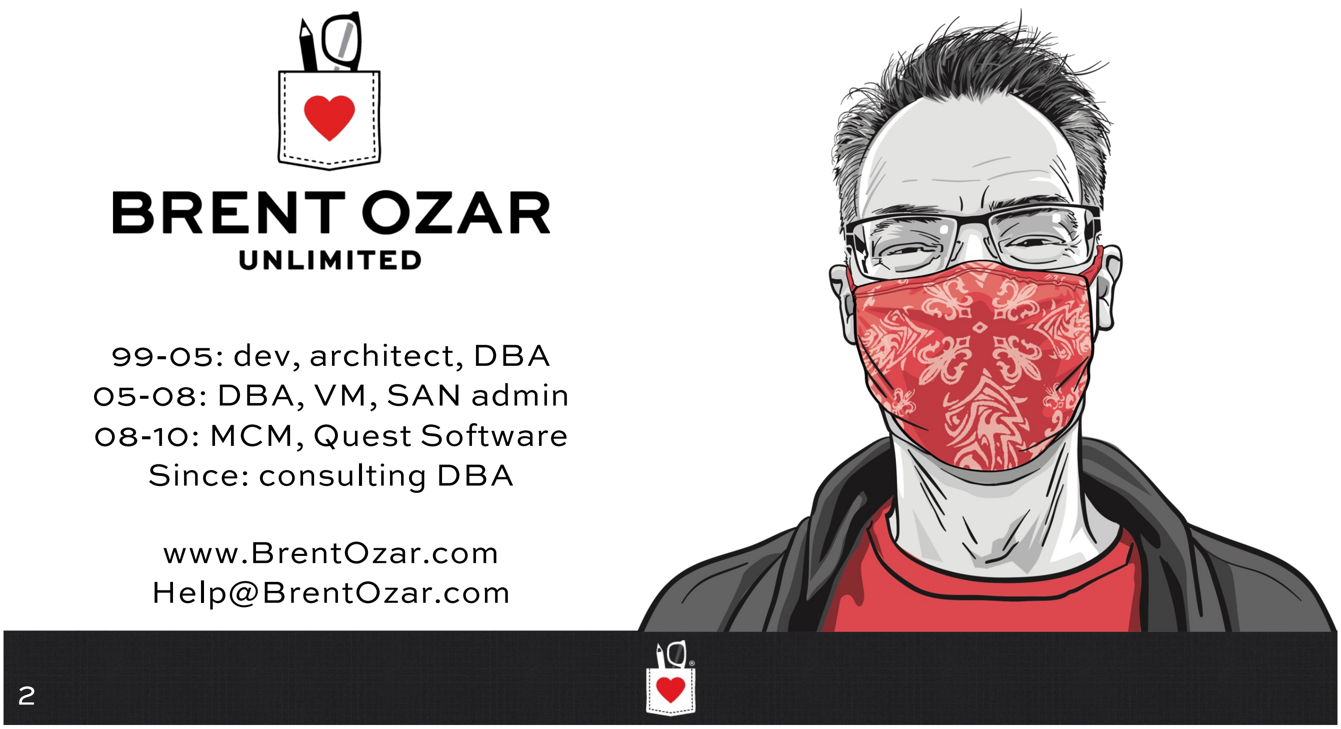


BRENT OZAR UNLIMITED®

Fundamentals of Parameter Sniffing

Up first: intros & logistics.

1



BRENT OZAR UNLIMITED

99-05: dev, architect, DBA
05-08: DBA, VM, SAN admin
08-10: MCM, Quest Software
Since: consulting DBA

www.BrentOzar.com
Help@BrentOzar.com

2

Introduce yourself in Slack:

	Developer	Development DBA	Production DBA
Write C#, Java code	Daily		
Build queries, tables	Daily	Sometimes	
Tune queries	Sometimes	Daily	
Design indexes		Daily	
Monitor performance		Daily	Sometimes
Troubleshoot outages			Daily
Manage backups, jobs			Daily
Install, config SQL			Sometimes
Install, config OS			Sometimes

Slack pro tips

Accidentally close your browser? Want to share screenshots? Lots of pro tips: BrentOzar.com/slack

To share code or T-SQL, click the + sign next to where you type text in, and choose “code or text snippet.”

To share files, upload at Imgur.com, paste URL here.

No direct messages please.

We’re going to cover

Understand what parameter sniffing is (and isn’t)

Analyze the kinds of queries that are susceptible

Lab: write a query that’s susceptible

See what triggers sniffing emergencies

Learn how to react to emergencies

Lab: tackle a sniffing emergency

See a few more examples of what sniffing isn’t

Mastering Parameter Sniffing

3-day class on reducing sniffing long term:

- Capturing different plans
- Prioritizing plan differences to fix
- Mitigate sniffing with indexes, T-SQL, database settings, server settings

This stuff is out of scope for Fundamentals.

We're using Stack Overflow data.

Open source, licensed with Creative Commons

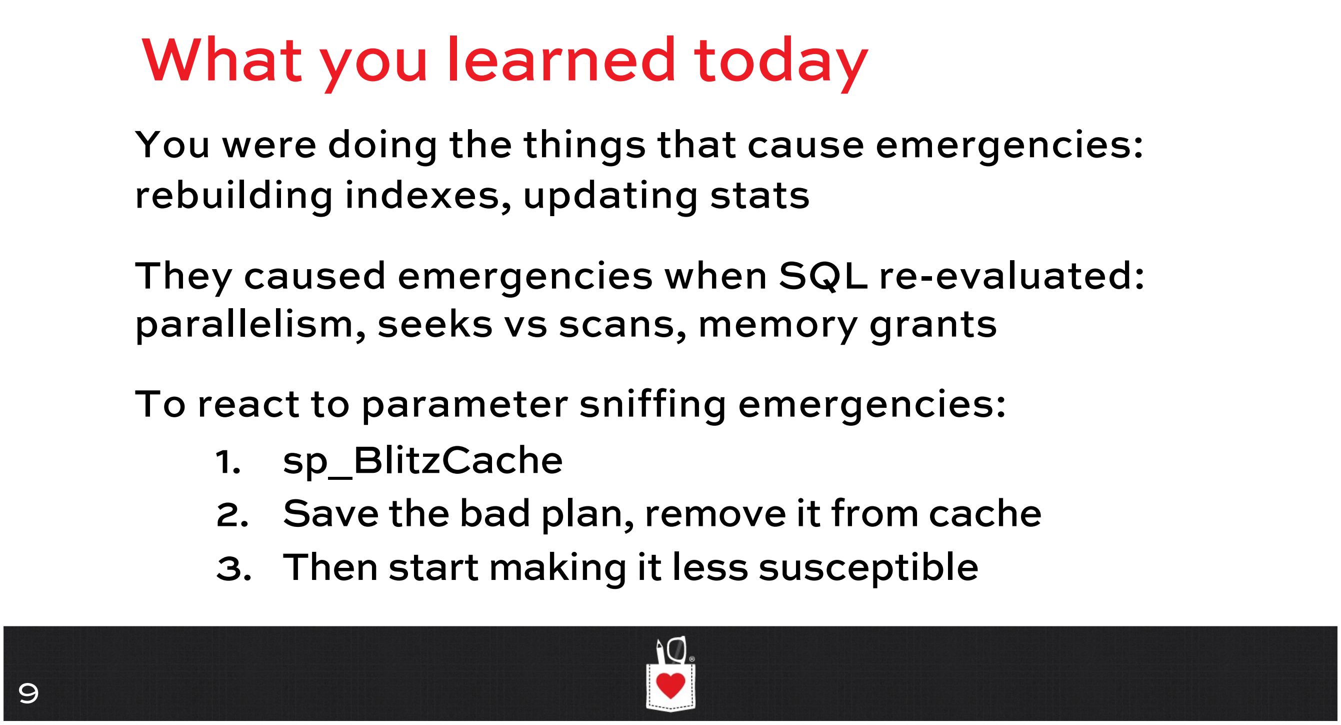
XML dump: archive.org/details/stackexchange

SQL Server db:
BrentOzar.com/go/querystack

Has to be the StackOverflow2013
database (50GB)

I'll be using SQL Server 2019.

Let's dive in!



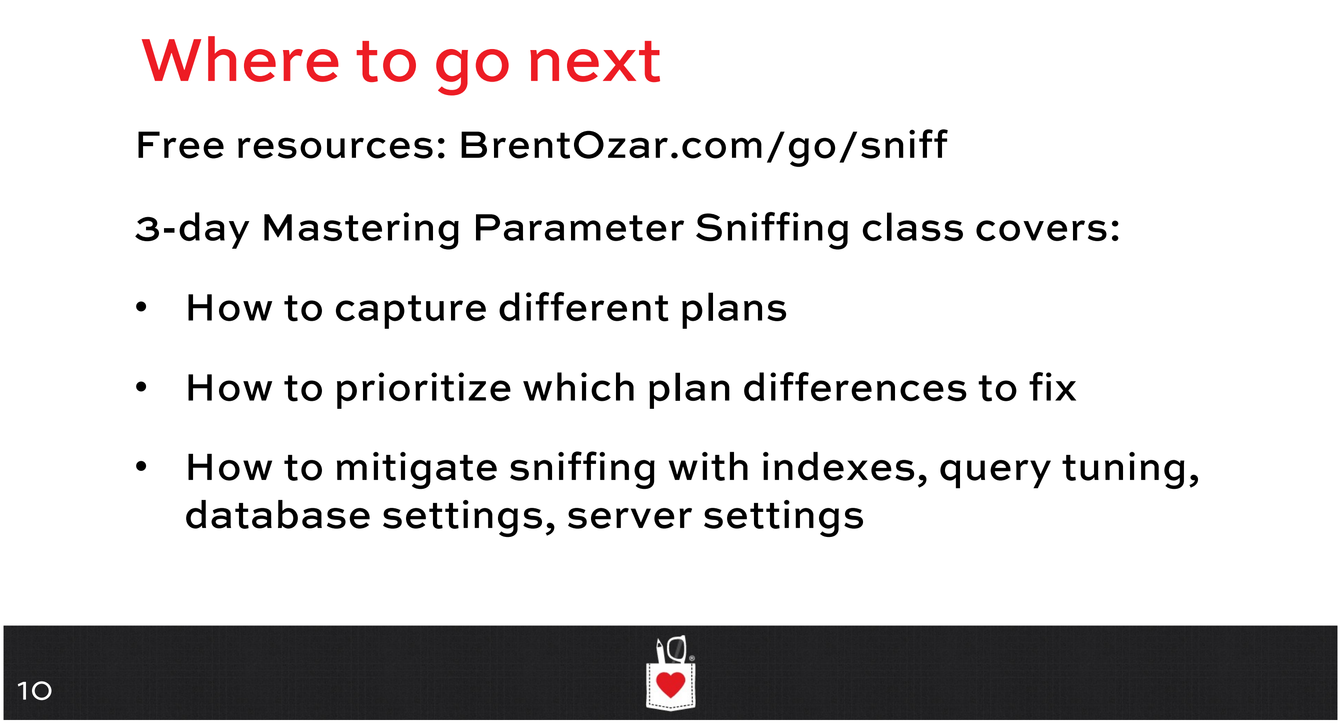
What you learned today

You were doing the things that cause emergencies: rebuilding indexes, updating stats

They caused emergencies when SQL re-evaluated: parallelism, seeks vs scans, memory grants

To react to parameter sniffing emergencies:

1. sp_BlitzCache
2. Save the bad plan, remove it from cache
3. Then start making it less susceptible



Where to go next

Free resources: BrentOzar.com/go/sniff

3-day Mastering Parameter Sniffing class covers:

- How to capture different plans
- How to prioritize which plan differences to fix
- How to mitigate sniffing with indexes, query tuning, database settings, server settings

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

Thanks, and I hope you had a great time!