He was off last week, Performance tuning is a poison chalice though, you'll start spotting horrors and know you can fix, only for a DBA to tell you they've fixed it by defragmenting indexes or some other temporary fix 🤦‍♂️

Seriously though, spend some time watching Brent Ozar

Others are Pinal Dave and Itzik Ben Gan and Erik Darling, can't go far wrong with those

Our biggest problem (and SQL Servers) is parameter sniffing, crack that and you'll be most of the way there

google "Brent Ozar parameter sniffing"

That's a good place to start

Yep, that's because there's only one good answer and that's to improve the indexing and design of the database so that the query is efficient. That's virtually impossible to get done here

It's an instant fail for me if the candidate says use index or join hints or god forbid recompile

you'll find all of those all over our code

and a bunch of nolocks too ffs

most of the people interviewing you will probably not be able to solve these issue and just looking for someone who does 😄

http://sqlsolarprod01:8123/iwc/main.iwc

have sa look at any server and the tuning tab for great example of what not to do

yep, have a look at some poorly performing queries and let me know what you think about how to improve them. I'll have a look then tell you what I'd do to diagnose them

This is a good one Michael: <http://sqlsolarprod01:8123/iwc/qpa.iwc?db_id=20&sqlHash=3741465016&startTime=2020-08-20T00:00:00.000Z&endTime=2020-11-18T00:00:00.000Z&tab=intelligentAnalysis>

Every so often it gets a bad plan

The query is triggered by a dot net process which as a 7 second timeout

The bad plans coincide with when data is being loaded into BureauData

Mostly I think when BureauData is being loaded over night

They're all written like that, I think you were following the pattern

If you get the execution plans up for it you'll see the difference between a good and a bad plan

What I think is happening from what Cordell has said is the load drops indexes, it's then that the bad plan gets locked in until the load completes and the indexes rebuild

thats bad yes, because the execution time exceeds 7 seconds, then the dot net job times out and it goes back into the queue. SQL Server continues executing it and tries to send the results back but the connection is dropped

it's like a denial of service attack 😄

All we're getting at the moment is suggestions like "update statistics", "defag indexes", none of these will fix it

yep

it can be an easy fix

The query joins these tables

FROM BDC.CustomerAccountSelect ca

INNER JOIN @AccountNumber a

ON ca.AccountNumber = a.AccountNumber

INNER JOIN BDC.CustomerCalculationValues cc

ON ca.PersonID = cc.PersonID

what we should do is create a table of all of the results for all accounts, then point the stored proc at this. Rebuild the data after each bureaudata load

It looks to normally run in about 100ms when it has a good plan, this would drop to less than 10ms which is a good saving for a proc that runs about 1 million times per day

Look through each server at the worst performing queries, there's some great examples of bad queries, most are only bad because we never expected they'd have to deal with so many records.

look at this one, it's many queries, each component of it can be optimised, runs about 100,000 times per day and takes 200ms<http://sqlsolarprod01:8123/iwc/qpa.iwc?db_id=17&sqlHash=3777612654&startTime=2020-10-17T23:00:00.000Z&endTime=2020-11-18T00:00:00.000Z&tab=sqlText>

If you've got a test server or even if you can run it on production see if you can improve it, measure your results using "SET STATISTICS TIME, IO ON"

Paste the statistics into this web page to make them easier to read: <https://statisticsparser.com/>