DBA Tower

UNLIMITED

Hesham Zaghloul, 2024-09-17

Background

This patient called us because they used a vendor app to run their business, and it was *slow.*

The vendor wasn’t seeing these issues at other companies using the software.

The patient had tried troubleshooting performance themselves, but didn’t have the skillsets to identify the root cause of the slow queries.

Day 3 agenda

Executive summary

Technical details on 4 options to fix performance

Along the way: reliability findings

Recap, and the 2 options it really boils down to

We also had a separate slide deck for day 1that covered how your server compared to others, plus what we saw in your data leading up to the engagement. That's a separate file, and we won't cover those issues here again, but if you're reading this later, check that file out too. It was good too.

Executive Summary

Patient: (redacted)

Total data size: ~8TB

Server: 32 CPU cores, 628GB memory

SQL Server 2016 Enterprise SP3

Applications involved: (redacted 3rd party app)

(You may have a few SQL Servers, but our SQL Critical Care® process just focuses on one. However, some of these lessons learned might apply to your other servers that have the same setup issues. It's like free consulting. You're welcome.)

Your #1 pain point: performance

You were having very bad storage performance

You switched to Nutanix storage 2-3 months ago

Things were briefly better, but…

You're back to seeing:

- Slow query response time periodically
- When queries are slow, it appears that there's a pileup of queries waiting on storage
- Storage is then taking 100-200+ milliseconds

What your SQL Server told us…

You've invested a lot in SQL Server licensing, CPU power, and memory. That's great!

You've done a good job of setting up the basics.

You're not missing any easy buttons.

Your assessment was right:
when SQL Server is waiting, it is indeed waiting to read uncached data from disk.

The causing factors

The vendor app's table design wasn't meant to scale:

- We're storing each metric, each day, each store, in its own row (overly normalized) – that's not how big reporting systems are built.
- We're not archiving data quickly enough. This design works for small data – not for big data.

Result:

- Storing the data takes too much space.
- Reading the data takes too much time.

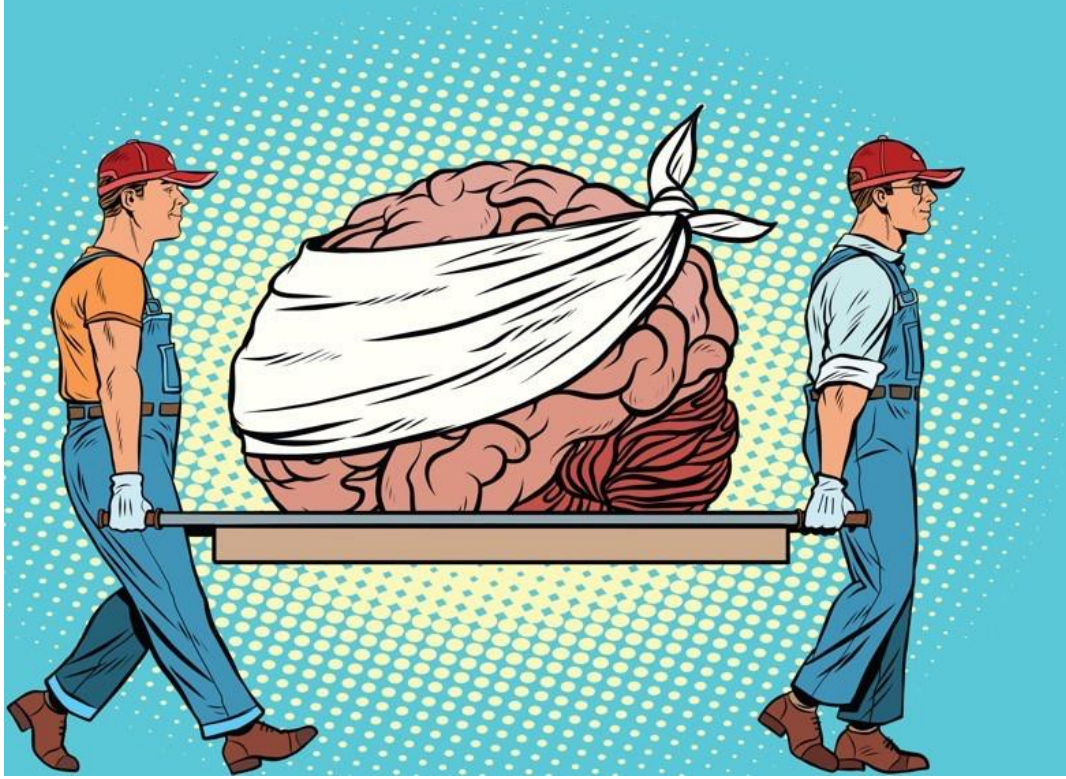
You can't change that design easily now.
We gotta figure out how to make it work.

Options to make it work

1. Hire a development DBA to focus on tuning, implementing columnstore indexes
2. Add a lot more memory
3. Buy faster, more appropriate storage
4. Build a denormalized data warehouse

You may need more than just one of these, especially if you plan to continue to grow and you refuse to archive data.

Option 1:
Hire a Dev DBA, do columnstore



	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

	Developer	Development DBA	Production DBA
Write C#, Java code	You have a few of these.		
Build queries, tables		Sometimes	
Tune queries		Daily	
Design indexes		Daily	
Monitor performance		Daily	You have one of these.
Troubleshoot outages			
Manage backups, jobs			
Install, config SQL			
Install, config OS			

A Prod DBA answers these:

"Given our current setup, how much data would we lose in a worst case scenario, in minutes?"

"What steps can we take to lose less data?"

"What bugs are in our build of SQL Server? Should we apply a newer Cumulative Update to fix them?"

"How can we patch and do maintenance with less downtime?"

"Who has access to see personally identifiable data, and how can we lock that down better?"

A Prod DBA learns:

Cluster troubleshooting

SQL Server troubleshooting

Backups & restores, including snapshots

PowerShell automation

The DBAtools PowerShell modules

How to use the above to do restore tests, patching

How to monitor the event log

A Dev DBA answers:

"What are our most resource-intensive queries?"

"How can we tune this query to use less resources?"

"What do we need to pass on to vendors to improve their applications?"

"What new SQL Server features can we use to improve performance without spending more?"

"Which tables are inefficiently indexed?"

"How do we implement columnstore here?"

"Have we hit a hardware bottleneck?"

For today, I'll be the Dev DBA.

But this is really tricky.

I'm going to give you a roadmap of the work that needs to be done.

You need an actual new head count to do it.

This is all net-new work.

The production DBA is already stretched too thin, and not covering all of the core job duties for that role.

While I’m here: prod DBA findings

“Do we really need a Dev DBA?”

I’m glad you asked: yes.

At 7-8TB of data on one server, things aren’t easy.

The production DBA will already have their hands full, and can’t do double duty.

You didn’t hire me to find homework for the production DBA, but I saw a couple of things that horrified me, and we need to lay out a quick action plan.

Backups are failing left and right.

Multiple apps are trying to back up this server.

We don’t have a clear inventory of what, or when.

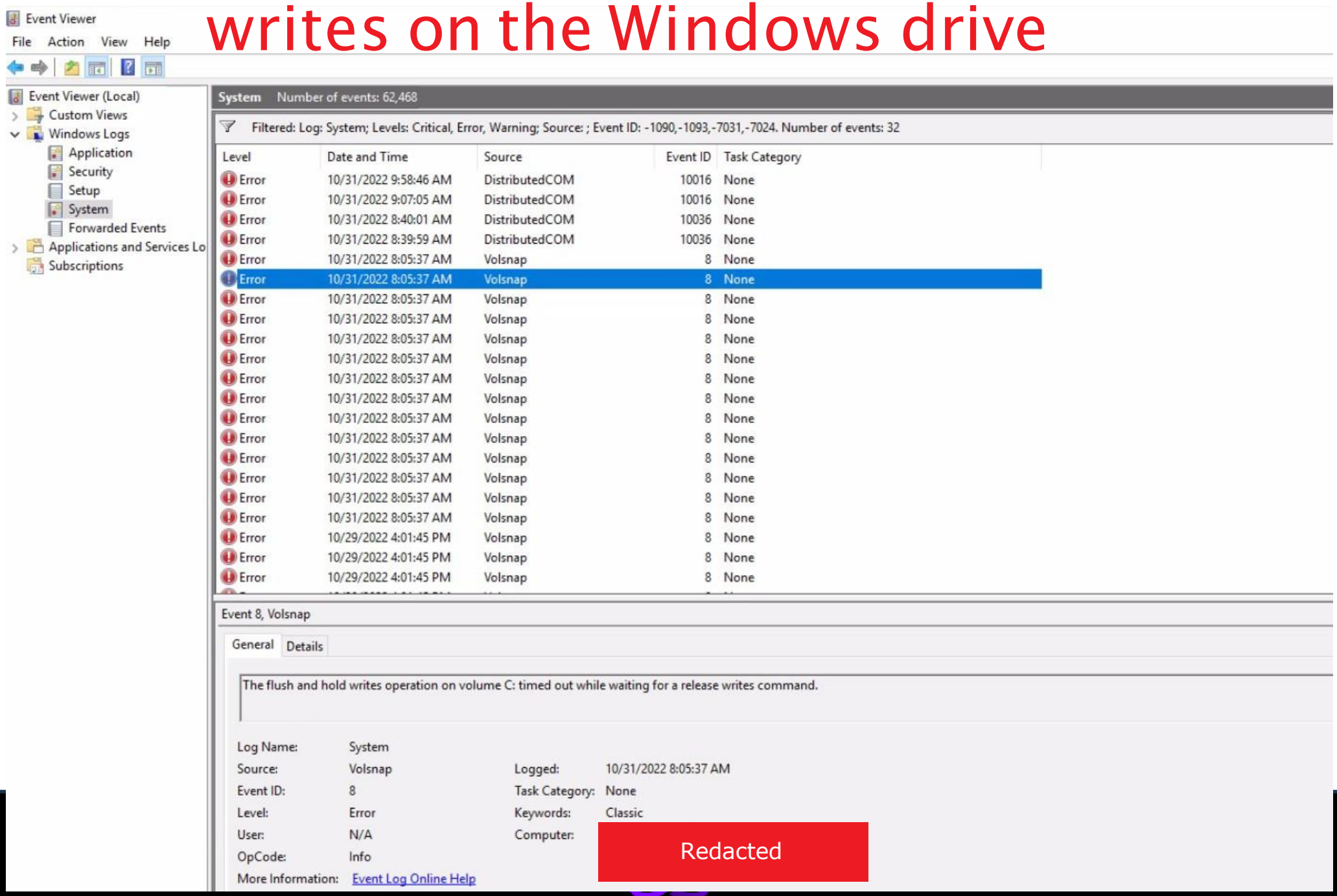
The event logs show multiple failures daily.

We haven’t done a restore test recently to know:

- How much data we would lose
- How long a restore would take
- Whether the backups are even working

I know that’s scary, but the following slides show errors out the wazoo in the system logs – and we didn’t even know these were happening…

Something's trying to freeze writes on the Windows drive



This is terribad.

If Veeam freezes Windows & SQL Server's drives during business hours, and the freeze lasts long enough to hit a timeout:

- Running queries will be blocked
- Incoming queries will pile up
- When the writes are thawed again, it'll take minutes of slow throughput to recover

These are exactly the symptoms you're witnessing.

Something's doing this at 8AM, 4PM, and midnight.

It's failing left and right.

“We just started doing that! That can’t be our problem!”

You don’t just have one performance problem.

You have LOTS of performance problems.

This is one of very many.

We just can’t get to this point because the production DBA has to be focused on things like backups.

Next backup failures…

Application Number of events: 36,532

Filtered: Log: Application; Levels: Critical, Error, Warning; Source: ; Event ID: -4098,-4350. Number of events: 2,795

Level	Date and Time	Source	Event ID	Task
Error	10/21/2022 5:00:59 AM	SQLVDI	1	Nor
Error	10/21/2022 5:00:59 AM	MSSQL$PDI	18210	Bacl
Error	10/21/2022 5:00:59 AM	MSSQL$PDI	3041	Bacl
Error	10/21/2022 5:00:59 AM	SQLVDI	1	Nor
Error	10/21/2022 5:00:59 AM	MSSQL$PDI	18210	Serv
Error	10/21/2022 5:00:59 AM	MSSQL$PDI	3041	Bacl
Error	10/21/2022 5:00:59 AM	SQLVDI	1	Nor
Error	10/21/2022 5:00:59 AM	SQLVDI	1	Nor
Error	10/21/2022 5:00:59 AM		24583	Nor
Error	10/21/2022 5:00:59 AM		3041	Bacl
Error	10/21/2022 5:00:57 AM		8194	Nor
Error	10/21/2022 5:00:21 AM	Nutanix	256	(1)
Error	10/21/2022 5:00:21 AM	Nutanix	256	(2)
Error	10/21/2022 5:00:20 AM	Nutanix	256	(2)
Error	10/21/2022 5:00:20 AM	Nutanix	256	(2)
Error	10/21/2022 5:00:20 AM	Nutanix	256	(2)
Error	10/21/2022 5:00:20 AM	Nutanix	256	(2)
Error	10/21/2022 5:00:16 AM	SQLWRITER	24583	Nor
Error	10/21/2022 5:00:16 AM	SQLWRITER	24583	Nor
Error	10/21/2022 5:00:16 AM	SQLWRITER	24583	Nor
Error	10/21/2022 5:00:16 AM	SQLWRITER	24583	Nor

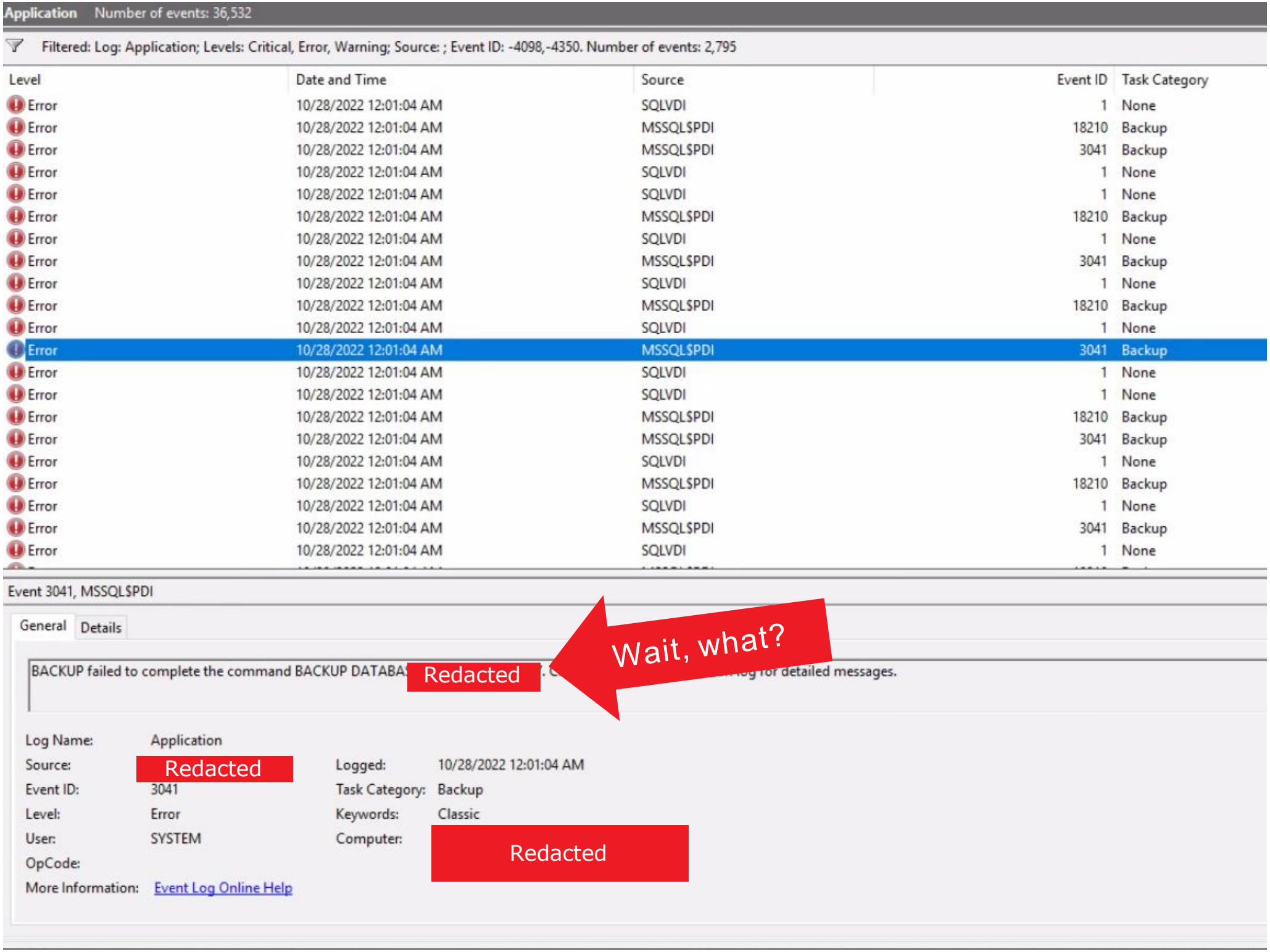
Other times too

Event 256, Nutanix

General Details

Create Snapshot for Transaction ID: [355:1666929701170096] failed with error: [Create Snapshot operation failed as Call to function: [CreateSnapshot] in the dll: [C:\Program Files\Nut
\NutanixVSSRequestor.dll] failed with error: [Windows Error 0x80042302]]

Log Name: Application

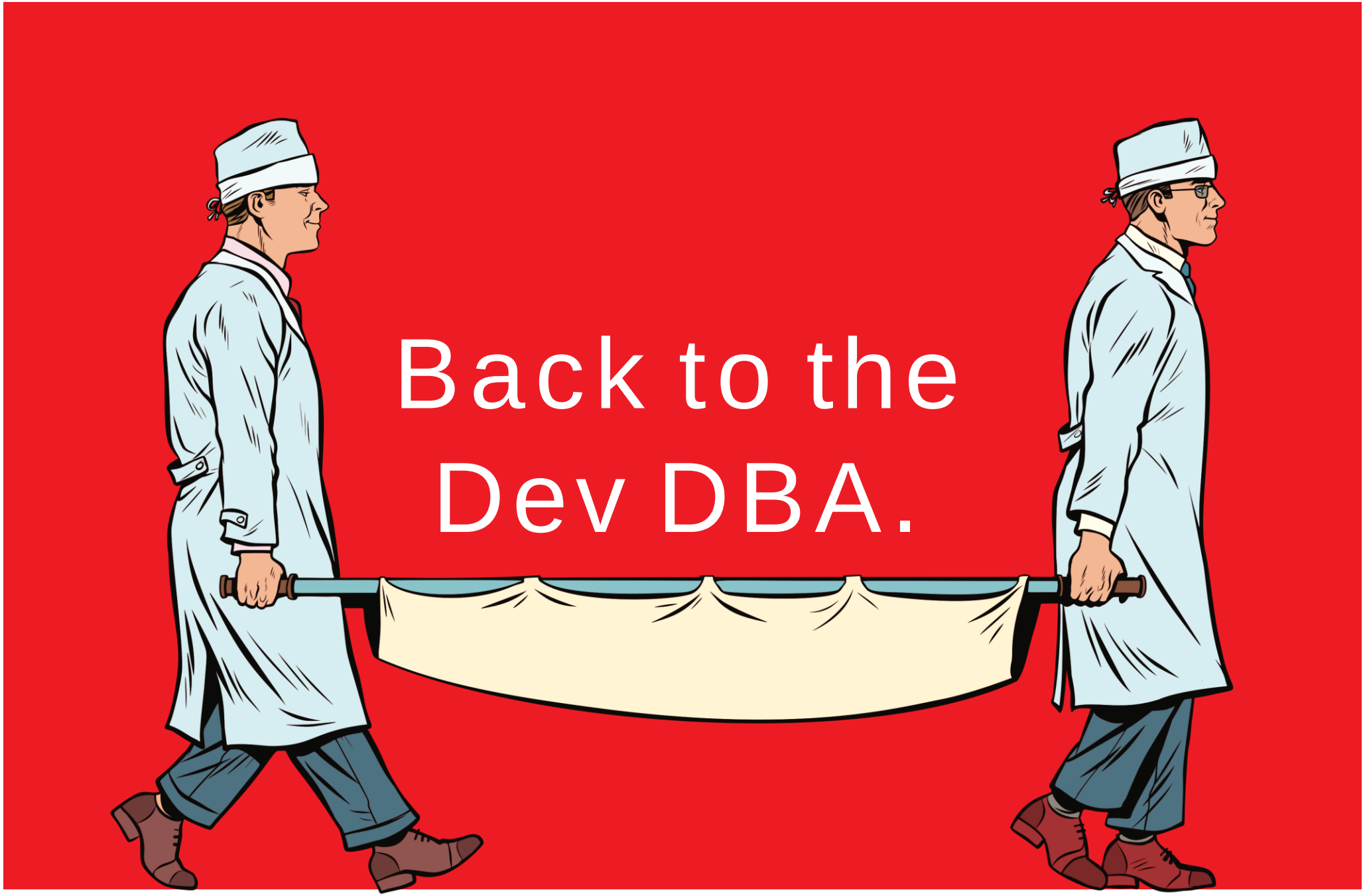

Application Number of events: 36,532
Filtered: Log: Application; Levels: Critical, Error, Warning; Source: ; Event ID: -4098,-4350. Number of events: 2,795
Level Date and Time Source Event ID Task Category
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 18210 Backup
Error 10/28/2022 12:01:04 AM MSSQL$PDI 3041 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 18210 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 3041 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 18210 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 3041 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 18210 Backup
Error 10/28/2022 12:01:04 AM MSSQL$PDI 3041 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 18210 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Error 10/28/2022 12:01:04 AM MSSQL$PDI 3041 Backup
Error 10/28/2022 12:01:04 AM SQLVDI 1 None
Event 3041, MSSQL$PDI
General Details
BACKUP failed to complete the command BACKUP DATABA
Redacted
Wait, what?
for detailed messages.
Log Name: Application
Source: Redacted
Logged: 10/28/2022 12:01:04 AM
Event ID: 3041
Task Category: Backup
Level: Error
Keywords: Classic
User: SYSTEM
Computer:
Redacted
OpCode:
More Information: Event Log Online Help

Who's backing up what?

The production DBA is responsible for:

- Knowing who's backing up what
- Taking investigative action if someone tries to start taking unauthorized production backups, which can result in performance catastrophes
- Monitoring when backups fail
- Testing that the backups are actually working

We're not doing any of those here. We need to start. Questions?


Back to the
Dev DBA.

What they'd normally do

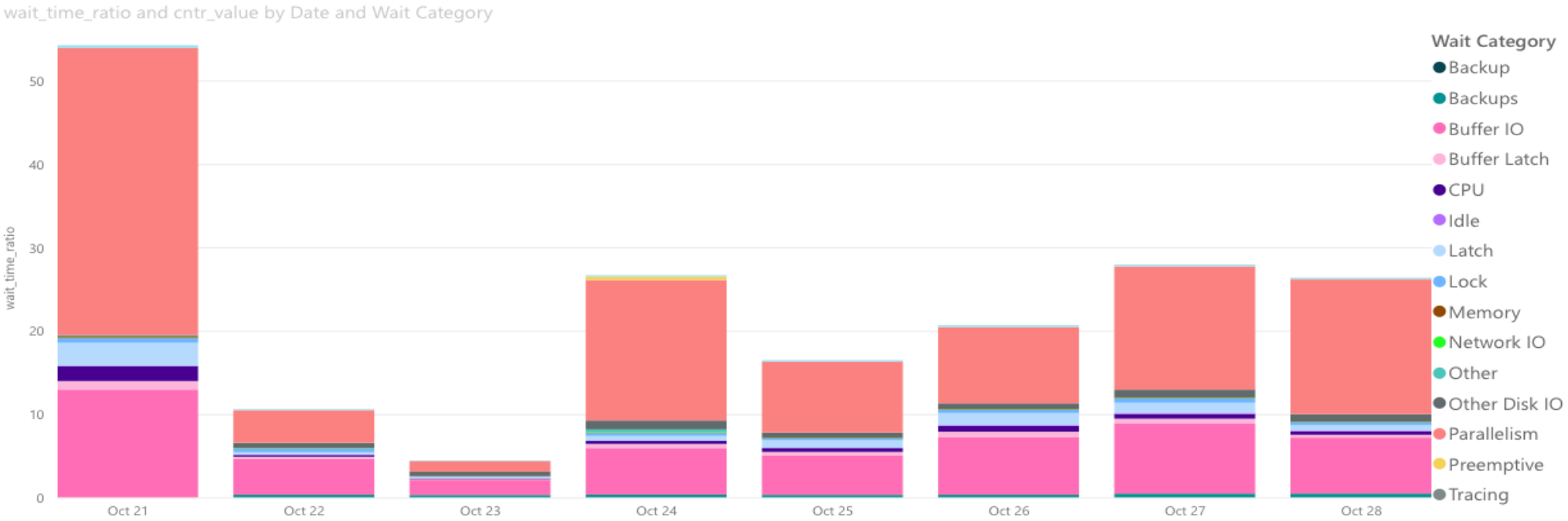
1. Review the server's top wait types
2. Investigate the queries causing those wait types
    1. If the queries are from internal developers, work with the developers to tune them
    2. If they're from the vendor app, work with them to document the overhead and suggest alternate ways to achieve the query results
3. See if indexes will help
    1. If they're your tables, fix the indexes
    2. If they're vendor tables, work with the vendor to fix them in a way that the vendor is okay with

This server is indeed waiting a lot.

Generally, when Wait Ratio > 1, people are unhappy.

The wait time ratio here is 20-50. That's bad.

So what are we waiting on?



Wait Type	Category	Minutes	avg_ms
CXPACKET	Parallelism	63,982	16
PAGEIOLATCH_SH	Buffer IO	46,442	11
CXCONSUMER	Parallelism	45,123	8
PAGEIOLATCH_EX	Buffer IO	11,491	13
LATCH_EX	Latch	7,532	2

Queries are going parallel to read a lot of data that isn't cached in memory.

That's fairly unusual in a transactional system like (vendor),
especially when the server has over half a terabyte of memory.

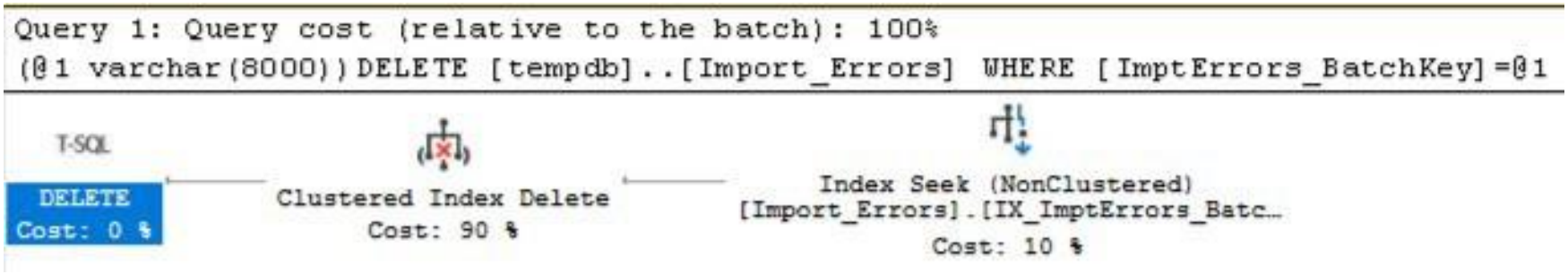
To find the root cause, we would normally look at:
Queries reading a lot of data pages
Tables that need better indexing

(See separate slide deck,
Supplemental 1:
Plan Caching Problems)

What we saw here

The top resource-intensive query list isn't accurate, but even so, I had so many questions about them.

The #1 most read-intensive query, by FAR: deleting from an import errors table:



“Who monitors these errors?”

The response: “Uh, nobody, I guess.”

So our server’s #1 resource consumer
is logging errors that nobody’s reviewing.

That’s a problem.

That’s the kind of thing a full time dev DBA digs into, and works with the rest of the staff and (vendor) to get to the bottom of the issue.

Your pain point: unpredictability

"We don't understand why a process that runs 3-4x per day can suddenly be slow out of nowhere."

Well, you don't monitor what's running on the server in terms of background processes.

This is tough here.

Due to the parameterization problem, monitoring tools will be challenged to tell you which queries are causing your performance issues.

Full-stack monitoring tools like New Relic can do it, but you have to integrate them into all of your apps – that's just not an option with 3rd party apps like PDI.

There are ways to do it, but they're labor-intensive, and involve building your own monitoring tools.

That's where the full time Dev DBA comes in.

Wait Type	Category	Minutes	avg_ms
CXPACKET	Parallelism	63,982	16
PAGEIOLATCH_SH	Buffer IO	46,442	11
CXCONSUMER	Parallelism	45,123	8
PAGEIOLATCH_EX	Buffer IO	11,491	13
LATCH_EX	Latch	7,532	2

Queries are going parallel to read a lot of data that isn't cached in memory.

That's fairly unusual in a transactional system like PDI,
especially when the server has over half a terabyte of memory.

To find the root cause, we would normally look at:
Queries reading a lot of data pages
Tables that need better indexing

How about this?

You can still do that.

And you did! And it shows.

SQL Server constantly generates a list of obvious indexes that would help performance.

You've done a good job of using index analysis tools like sp_BlitzIndex to hear those, and add them.

That's great! You found the low-hanging fruit!

The bad news: there's no more low-hanging fruit. Remaining index work is going to be much harder.

Columnstore might work well.

Columnstore indexes were designed for this exact scenario: tables with billions of rows, queried in unpredictable patterns.

- Redacted_Table_1: 3.5B rows, >1TB
- Redacted_Table_2: 1.3B rows, ~300GB
- Redacted_Table_3: ~500M rows, ~500GB

Why columnstore?

- Awesome compression: 70-90% likely here
- That means more data can be cached in RAM
- That means less time spent waiting on storage

Plus, report-style queries run dramatically faster on newer versions of SQL Server – aim for 2019+.

It's not an easy task, though.

Switching from rowstore tables to columnstore involves serious work to:

- Design the sorting strategy
- Design the initial build process
- Test the builds in a load test environment, timing them to see how long the outage will be
- Picking report queries to test before/after
- Going live with the change
- Managing index maintenance going forward (an absolute requirement for columnstore)

If I was the Dev DBA…

1. Get a server provisioned identical to production
2. Restore the production databases onto it
3. Identify the smallest, safest "big" table that's a good candidate for columstore
4. Do a few proof-of-concept implementation & tests
5. Show the results to the company and to PDI, and discuss doing it for real in production

Realistically, this is weeks of dedicated work.
And this is just one part of their job!

Then, repeat steps 3-5 on progressively larger tables.

Your Dev DBA's job description

Job duties:

- Manage performance on a multi-terabyte SQL Server OLTP (transactional) database running 3rd party application code
- Migrate multi-billion row tables to columnstore
- Implement forced parameterization
- React to parameter sniffing emergencies

Required experience to do that task:

- Experience tuning multi-terabyte databases on Microsoft SQL Server
- Experience with billion-row columnstore tables

This is going to be hard to find.

The few people doing this usually work for:

- Large companies with big budgets
- Consulting companies where they get billed out
- Cloud vendors (Microsoft, AWS, Google)

And there are risks with hiring:

- You have to choose the right candidate (tough when you don't know the right interview questions, can't assess skills well)
- You have to give them the political backing to let them push changes through
- They have to make the right changes
- The changes will take months to finish

You should use a consultant.

We already have Database development service that you can get benefit of it.

Below is the link for our service details with the prices:

https://dbatower.com/services/sql-development-service/

Questions on the Dev DBA option?

One small note on filegroups

You mentioned you were considering making big changes to the filegroups & files.

That absolutely can pay off – but let's leave that to the dev DBA to prioritize.

I need the production DBA 100% focused on getting the backups & restores under control.

Back to the list of options

1. Hire a development DBA to focus on tuning, implementing columnstore indexes
2. Add a lot more memory
3. Buy faster, more appropriate storage
4. Build a denormalized data warehouse

Option 2:
add memory

Queries are waiting to read data.

Makes sense – you've got some big tables:

- Redacted_Table_1: 3.5B rows, >1TB
- Redacted_Table_2: 1.3B rows, ~300GB
- Redacted_Table_3: ~500M rows, ~500GB
- And many more
- And you're not purging old data quickly

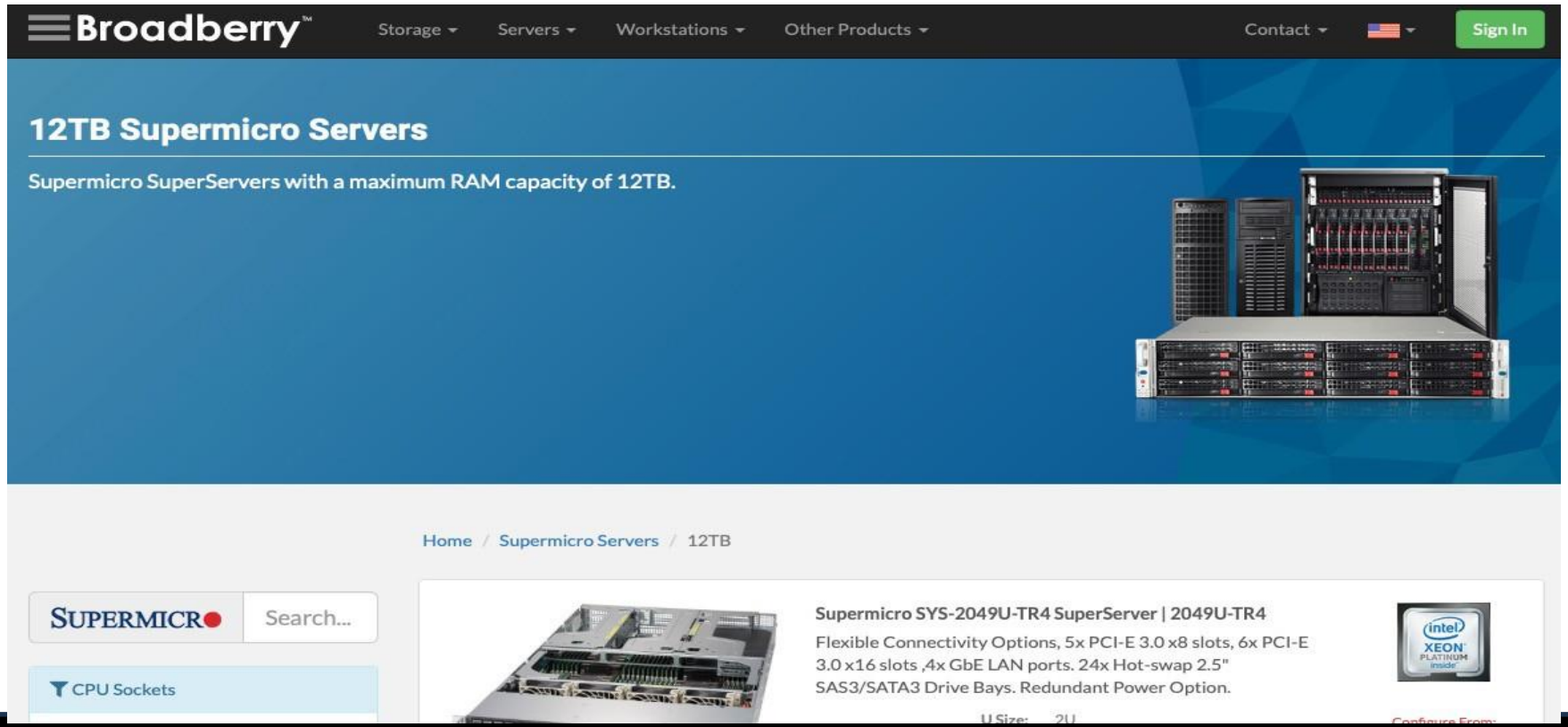
You've only got about 600GB of memory for caching.

You can't cache 7TB of data with 600GB RAM.

“How much do we need?”

Well, if you want to stop waiting for storage…
you would need 7TB (or more) of memory.

And they are indeed out there:



Companies do do this.

This includes 2 12TB drives.

You could have bought a pair of these instead of the Nutanix.

Plus one for your DR site.

Compare the price to:

- Hiring a dev DBA employee or consultant
- Waiting for them to put in columnstore

If you take this approach

Provision a UNC file path for backups.

Write SQL Server backups directly to the UNC path.

Sysadmins back up the UNC path.

Under absolutely no circumstances do the sysadmins install any backup software on these boxes. I don't trust your current approach to backups. It's a mess.

If something goes wrong on this box, don't restore it:

- Fail over to the alternate box
- Erase the failed box and reinstall it

Back to the list of options

1. Hire a development DBA to focus on tuning, implementing columnstore indexes
2. Add a lot more memory
3. Buy faster, more appropriate storage
4. Build a denormalized data warehouse

Let's be real: you're not doing the bottom two.
But I want to put in a few slides to talk about 'em.

Option 3: more appropriate storage

Before you buy something

Ask a simple question:

"Can we talk to other customers of yours, that have our database sizes, and they're happy with your solution?"

You should have asked Nutanix,
"Can we talk to happy customers of yours that have 5-10TB SQL Server databases on Nutanix?"

You would have learned a lot, and gone elsewhere.

Nutanix has its use cases!

It’s great for many virtualization deployments.

If you were interested in buying more appropriate storage for a multi-terabyte data warehouse, you could reuse your Nutanix gear for other VMs.

We briefly discussed your interest in replacing the Nutanix, and it’s a non-starter – so let’s move on.

In the future, though, seriously: ask for references.

Option 4: build a denormalized warehouse

“Can’t we just move to Azure?”

Sure! But we need to be clear on what Azure is.

Azure is a giant pool of services:

- Azure SQL DB: inappropriate for a 7TB warehouse
- Azure SQL DB Hyperscale: the migration would be challenging due to the 100 MB/sec log throughput limit today, although that might change in the future
- Azure SQL DB Managed Instances: might work, see pricing info on the following page. Keep in mind that your RPO/RTO goals dictate two of these servers, at least, and do not include storage.

These prices are per-server

Premium-series, memory-optimized

Premium-series logical CPUs are based on the latest Intel(R) Xeon Scalable 2.8 GHz processor (Ice Lake), 1 vCore = 1 hyper thread. This memory-optimized version provides nearly twice the memory per vCore and is great for database workloads that require even better memory performance than the standard next generation offering.

vCORE	Memory (GB)	Included Storage	Pay as you go
4	54.4	First 32 GB/month	$2,533.83/month
8	108.8	First 32 GB/month	$5,066.93/month
16	217.6	First 32 GB/month	$10,134.59/month
24	326.4	First 32 GB/month	$15,201.52/month
32	435.2	First 32 GB/month	$20,268.45/month
40	544	First 32 GB/month	$25,335.38/month
64	870.4	First 32 GB/month	$40,536.90/month

Max storage: 5TB

Max storage: 16TB

The only option for you

Challenges with Mgd Instances

It gets expensive, quickly.

You can only do the largest instance size due to your database size. (Sure would be nice if you archived!)

Even that big instance size doesn’t have much more memory than you have today.

So… you pay a lot to be right back where you started.

So why do people love the cloud?

Several reasons:

- They have smaller databases
  (like many Nutanix customers - a lot of shops just don't have big, high-performance databases)
- They work harder on performance tuning
- They have very flexible performance needs (only run at high utilization rates, certain times of year)
- Or, they re-architect for cloud-native architecture.

Two ways to move to the cloud

Lift-and-shift:

- Copy your existing implementation up to Azure
- Minimal application changes required
- Cheap to migrate, but expensive long term

Or rearchitect for cloud-native:

- Rewrite many of your things from the ground up
- Leverage cloud-specific services, things that weren't available when you first wrote your app (event-based processing, serverless)
- Expensive to migrate, but cheaper long term

Cloud-native data warehouses

Another term: The Modern Data Warehouse

https://www.youtube.com/watch?v=TZHykX6cEyc

Rewrite your application to use:

- Azure Synapse Analytics
- Azure Data Lake Storage
- Power BI for reporting

This is a complete rewrite of how you do reporting, and consulting companies specialize in this process.

This is indeed doable! Companies are doing it today. (I just don't think you're interested in it. Questions?)

Back to our options

It really comes down to two:

1. Hire a development DBA to focus on tuning, implementing columnstore indexes
2. Add a lot more memory (like 6-12TB)
3. ~~Buy faster, more appropriate storage~~
4. ~~Build a denormalized data warehouse~~

Questions on those two?

Recap

The findings in one slide

Production DBA: focus on getting the backups under control, start testing them, and work on DR.

Management: decide whether to:

- Hire a full time Dev DBA or a consultant, and focus them on implementing columnstore
- Move to much larger servers (6-12TB RAM)
- Or both

Much longer term, 2-3 years out, you could:

- Rearchitect the data warehouse with cloud-native solutions like Azure Synapse Analytics

Got questions going forward?

Email us at help@DBATower

If I can answer it in 5-15 minutes, it's free.

Don't overthink it: ask the question, and I'll ask for supporting resources I need to get the right answer.

This should get you pain relief.

Any questions about the findings?