

# **Building a Full SQL Server Telemetry & Alerting Pipeline From the Ground Up**

It has been a challenging and incredibly rewarding journey going from helpdesk to sysadmin and now into DBA work in a data-heavy environment. Before we had commercial tools, when I began learning the ropes as a Jr. DBA we had at the time 60+ SQL Servers where we relied primarily on internally built email alerts — which worked well for years before I joined.

As the environment grew, so did the volume and complexity of operational signals. The email alerting functioned reliably, but the sheer scale of new activity meant that issues were increasingly difficult to triage quickly. Important alerts were scattered across dozens of emails or required manually running ad-hoc queries. New emails weren't needed — centralized visibility was.

This became the foundation for a centralized telemetry, monitoring, and alerting project that helped surface key SQL Server metrics in a more visual and actionable way.

## **The Solution: SQL Server Big Brother**

I built a centralized telemetry + monitoring system (“SQL Big Brother”) powered by ~30 custom PowerShell and Python scripts. These scripts collect metrics across our DEV/TEST/PROD SQL Servers every five minutes and write them into a centralized SQL monitoring database.

That data powers:

### **1. A unified Power BI dashboard**

- Hosted on our on-prem Power BI report server
- Auto-refreshes every 5 minutes
- Provides an “at-a-glance” operational snapshot across the entire SQL estate
- Includes 15+ granular drill-down pages (blocking, TempDB, failed jobs, backups, login anomalies, disk space, business specific monitoring, etc.) Complete with sliders and other drill down tools.

## 2. Automated alert ticketing

A companion set of scripts leverages the Freshworks API to automatically generate tickets when conditions are met — such as blocking thresholds, job failures, disk pressure, missing backups, and more.

This has closed a long-standing gap in documenting operational incidents and ensures DBAs can respond quickly with clear audit trails.

### A Real Example of Preventing Production Issues

One notable case occurred earlier this year:

On a major production SQL Server, hidden shadow copy files had begun consuming disk space. These weren't visible through normal workflows, and no traditional SQL alerting surfaced them.

But on our **disk space heat map**, which shows disk utilization trends across the estate, one of the production drives began drifting from "green" into "yellow." That visual deviation was subtle — but it immediately stood out.

When we investigated, we discovered that a component of our Commvault backup infrastructure was unexpectedly generating large shadow copies. Had it continued, the production system would have run out of disk space — causing a critical outage.

The centralized dashboard provided **early detection**, long before the issue became severe. That single catch prevented:

- A major production impact
- Potential data loss
- Emergency after-hours remediation
- Service disruption to our business

This was one of several early warning wins that reinforced the helpfulness of the project.

## **What the System Monitors**

The dashboard and automated alerts allow the DBAs to monitor:

- SQL Agent Job Failures
- Down Servers
- Last Server Restart
- Noteworthy Login Failures
- Lower Environment Refresh Status
- Heavy Sessions (Blocking, Waits, Memory, etc.)
- Azure Backup Status
- Logshipping Health
- Missing/Failed SQL Backups
- Noteworthy Database States
- Terminated Users With SQL Logins
- SQL Server Encryption Status
- CPU Usage/History
- TempDB Condition
- Disk Space Monitoring & Low Disk Alerting
- Memory Dump Ticket Generation

## Outcome

This project transformed our team's ability to observe, respond to, and understand the health of our SQL Server environment. Instead of relying solely on scattered emails or manual queries, the team now has:

- **Centralized telemetry**
- **Visual situational awareness**
- **Automated issue detection**
- **Consistent ticket documentation**
- **Faster triage and response cycles**
- **Actionable insights that helped prevent real incidents**

What began as a personal learning initiative quickly became a full monitoring ecosystem for our SQL Server environment — one that improved visibility, reduced triage time, prevented outages, and elevated the DBA team's operational awareness.

This project challenged me to deepen my scripting, automation, data engineering, and monitoring skills, and it remains one of the most meaningful engineering efforts I've contributed to.

Screenshots:

Main page. Cards above utilize conditional formatting as tuned for our needs. All clickable cards



Heavy Sessions page allow for a quick look at all noteworthy sessions in our whole environment, allows for easy drill-down and filtering.

SQL Server Heavy Sessions
()
All
11/8/2025 6:03:26 PM

ServerName	Duration (Seconds)	SPID	BlockingSPID	BlockCnt	Start	Login	Host	Program
TstSrvr44	0	59	0	0	Nov 8 2025 6:03PM	FakeLogin2	DmmyHost2	Microsoft JDBC Driver for SQL Server
TestSrvr11	1809	144	0	0	Nov 8 2025 5:33PM	FakeLogin3	DmmyHost4	SQLAgent
TestSrvr11	1806	145	0	0	Nov 8 2025 5:33PM	FakeLogin3	DmmyHost4	SQLAgent
TestSrvr11	2003	150	0	0	Nov 8 2025 5:30PM	FakeLogin3	DmmyHost4	SQLAgent
TestSrvr11	3803	159	0	0	Nov 8 2025 5:00PM	FakeLogin3	DmmyHost4	SQLAgent
TestSrvr11	1808	162	0	0	Nov 8 2025 5:33PM	FakeLogin3	DmmyHost4	SQLAgent
TestSrvr11	1804	176	0	0	Nov 8 2025 5:33PM	FakeLogin3	DmmyHost4	SQLAgent
DmmyBox6	2487787	87	0	0	Oct 10 2025 11:00PM	FakeLogin4	DmmyHost5	RSPowerBI
DmmyBox6	2376193	89	0	0	Oct 12 2025 6:00AM	FakeLogin4	DmmyHost6	RSPowerBI

SPID	CPU (ms)	Logical Reads (MB)	Writes (MB)	TempDB (MB)	Wait
60	62828224	115569827	577983	359889	(1ms)CXPACKET
80	13169132	22069383	1470828	3	
144	2019714	903777	57350	0	
176	468377	107255	1	0	(1757ms)WAITFOR
162	309274	51820	1626	0	(1353ms)WAITFOR
159	232561	859	4	0	(2ms)BACKUPBUFFER

**SQL Command**

```
@ClaimedBy = 'IndexMaint_Worker 1'
exec master.dbo.sp_IndexMaint_SpawnWorker
@Jobid = 126,
@ClaimedBy = 'IndexMaint_Worker 2'
exec master.dbo.sp_IndexMaint_SpawnWorker
@Jobid = 126,
@ClaimedBy = 'IndexMaint_Worker 3'
exec master.dbo.sp_IndexMaint_SpawnWorker
@Jobid = 126,
@ClaimedBy = 'IndexMaint_Worker 4'
EXEC sp_server_diagnostics 10
EXEC sp_trace_getdata 2,0
```

**CPU Usage Breakdown**

63M (78.48%)	13M (16.45%)	3M (4.11%)
--------------	--------------	------------

Login Failures page offers the team quick insight into all failed attempts at connecting. We can see who, where, and when they occurred along with visual aids to make finding possible trends and issues easier.

# SQL Server Login Failures

This page reflects the capture of all login failures through out all servers.  
New data will be refreshed around every 5 minutes.

11/8/2025 5:00:04 PM

**53**

Login Failures

All

Server Name	Count	Percentage
DmmySrv3	37	69.81%
DmmySrv8	16	30.19%
LoginUser	2	3.77%

LoginUser	Count	Percentage
DOM\FakeUsr80	34	64.15%
DmmySrv8	16	30.19%
LoginUser	2	3.77%

ClientIP	Count	Percentage
192.168	34	64.15%
192.168	16	30.19%
10.158	2	3.77%

Server Name	Log Date	LoginUser	ClientIP	Failed Login	Reason
DmmySrv3	11/8/2025 4:51:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 4:21:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 3:51:49 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 3:21:49 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 2:51:49 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 2:21:49 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 1:51:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 1:21:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv88	11/8/2025 1:36:25 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv88	11/8/2025 1:36:25 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv88	11/8/2025 1:23:24 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv88	11/8/2025 1:23:23 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv3	11/8/2025 1:21:49 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 12:51:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv88	11/8/2025 12:23:10 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv88	11/8/2025 12:23:10 PM	DOM\UMBOT1	192.168	Login failed for user '	: Reason: Failed to open the explicitly specified database [CLIENT: 192.168]
DmmySrv3	11/8/2025 12:21:48 PM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 11:51:49 AM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 11:21:48 AM	DOM\FakeUsr80	192.168	Login failed for user	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 10:51:49 AM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]
DmmySrv3	11/8/2025 10:21:49 AM	DOM\FakeUsr80	192.168	Login failed for user '	: Reason: Could not find a login matching the name provided. [CLIENT: 192.168]

Example of an automated alert ticket utilizing the data gathered from this project. Tickets in this example are for a missing backup and for SQL Agent job failures.

Tickets > #INC-[REDACTED]

Missing Backup Alert [REDACTED] on [REDACTED]  
[REDACTED] reported 38 minutes ago (Tue, Dec 2 8:05 AM) via Phone

Details Related tickets Tasks Assets Associations Impacted services Responders Activity Resolution

Description

Backup Check Failure Detected

Server: [REDACTED]  
Database: [REDACTED]  
Status: No recent backup found  
Checked At: 12/02/2025 08:00:23

Please verify the backup job for this database.

Resolve this incident faster with Freddy

View similar tickets for resolution insights

30 Similar tickets →

Properties

2 Freddy suggestions >  Highlighted below for your review

Priority: Low Status: Open

Source: Phone Type: Incident

Urgency: High Impact: Medium

Group: DBA (IT Applications Support)

Agent: [REDACTED]

Department: IT Systems

Conversations

Reply Forward Add note

Tickets List - \* My Group's Unresolved Tickets X

Select all Sort by: Subject

Export 1 - 15 of 15

Subject	Created Date	Reques...	Status	Priority	Assigned to
SQL Job Step Failure: SSIS Server Maintenance Job on [REDACTED] #INC-191636	Dec 2, 2025 12:15 AM	[REDACTED]	Open	Low	DBA (IT Applications Support)/ None
SQL Job Step Failure: SSIS Server Maintenance Job on [REDACTED] #INC-192010	Dec 2, 2025 11:15 PM	[REDACTED]	Open	Low	DBA (IT Applications Support)/ None
SQL Job Step Failure: Refresh [REDACTED] on [REDACTED] #INC-192019	Dec 3, 2025 1:15 AM	[REDACTED]	Open	Low	DBA (IT Applications Support)/ None

Quick overhead visual of how the whole thing works.

