DESERT ISLE GROUP

ADVANCED DATA REALIZED

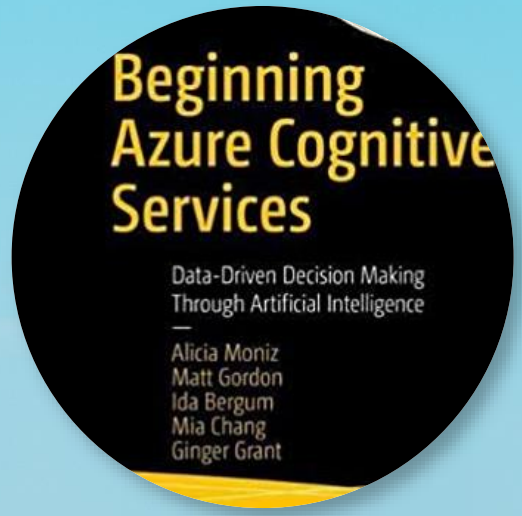
SQL DB in Microsoft Fabric

Arizona Data Platform Users Group

GINGER GRANT

January 8th, 2025

About Me:
Ginger Grant



ginger.grant@desertislesql.com

desertislesql.com

desertislesql

SQL Database in Fabric

Added in November 2024

Implementation of Azure SQL -- kinda

Stores data in OneLake

Designed for ease of use for application

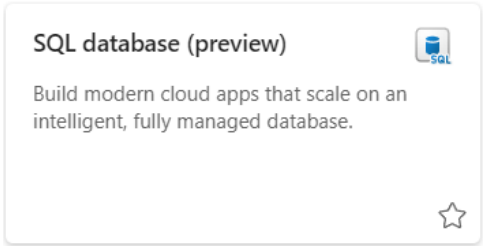
SQL DB Creation Experience

Like the rest of the Fabric tools

Little ability to tweak

Query Autotune turned on

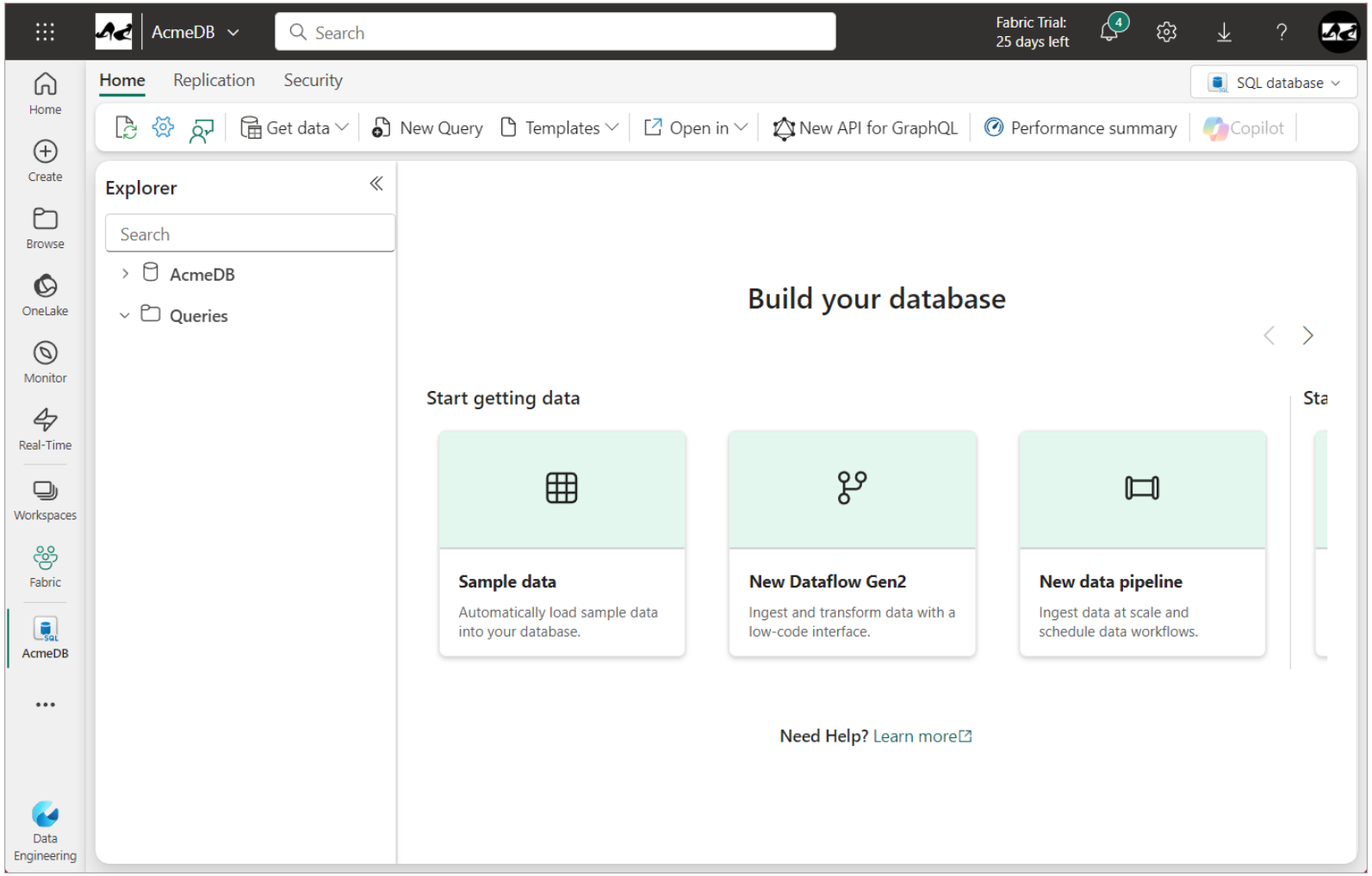
Creating a SQL DB



Click on the button

Database is created

Then you can use tools to add data to it



Demonstration

SQL DB in Fabric

Things not like SQL DB

- Data stored in One lake
- Built in Source Control
- Built in Performance Summary
- Easy GraphQL Support for applications

Using SQL DB inside of Fabric

Pros

- Query Autotune
- Easy access performance monitoring
- Built in Source Counrol
- No Management requirements

Cons

- Cannot turn off Query Autotune
- No Priority
- Unknown cost
- Few tuning options

Performance Monitoring

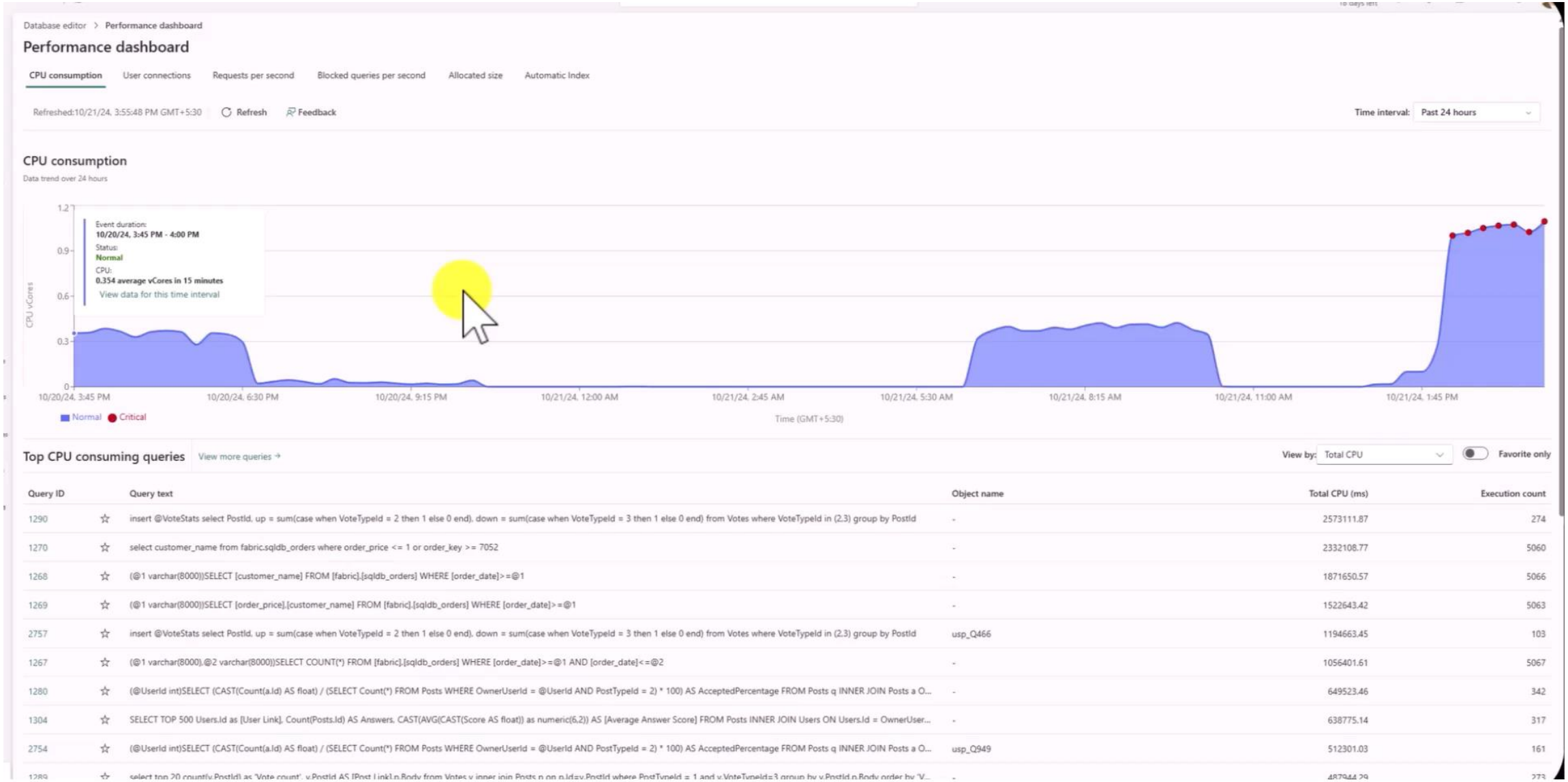
Easy to see what is going on

Review current state or see past state

Status and health are easily defined

Demonstration

SQL DB Performance Monitoring


Database editor > Performance dashboard
Performance dashboard
CPU consumption
User connections
Requests per second
Blocked queries per second
Allocated size
Automatic Index
Refreshed:10/21/24, 3:55:48 PM GMT+5:30
Refresh
Feedback
Time interval: Past 24 hours
CPU consumption
Data trend over 24 hours
Event duration:
10/20/24, 3:45 PM - 4:00 PM
Status:
Normal
CPU:
0.354 average vCores in 15 minutes
View data for this time interval
CPU vCores
1.2
0.9
0.6
0.3
0
10/20/24, 3:45 PM
10/20/24, 6:30 PM
10/20/24, 9:15 PM
10/21/24, 12:00 AM
10/21/24, 2:45 AM
10/21/24, 5:30 AM
10/21/24, 8:15 AM
10/21/24, 11:00 AM
10/21/24, 1:45 PM
Normal
Critical
Time (GMT+5:30)
Top CPU consuming queries
View more queries
View by: Total CPU
Favorite only
Query ID
Query text
Object name
Total CPU (ms)
Execution count
1290 insert @VoteStats select PostId, up = sum(case when VoteTypeId = 2 then 1 else 0 end), down = sum(case when VoteTypeId = 3 then 1 else 0 end) from Votes where VoteTypeId in (2,3) group by PostId - 2573111.87 274
1270 select customer_name from fabric.sqldb_orders where order_price <= 1 or order_key >= 7052 - 2332108.77 5060
1268 (@1 varchar(8000))SELECT [customer_name] FROM [fabric].[sqldb_orders] WHERE [order_date]>=@1 - 1871650.57 5066
1269 (@1 varchar(8000))SELECT [order_price],[customer_name] FROM [fabric].[sqldb_orders] WHERE [order_date]>=@1 - 1522643.42 5063
2757 insert @VoteStats select PostId, up = sum(case when VoteTypeId = 2 then 1 else 0 end), down = sum(case when VoteTypeId = 3 then 1 else 0 end) from Votes where VoteTypeId in (2,3) group by PostId usp_Q466 1194663.45 103
1267 (@1 varchar(8000),@2 varchar(8000))SELECT COUNT(*) FROM [fabric].[sqldb_orders] WHERE [order_date]>=@1 AND [order_date]<=@2 - 1056401.61 5067
1280 (@UserId int)SELECT (CAST(Count(a.Id) AS float) / (SELECT Count(*) FROM Posts WHERE OwnerUserId = @UserId AND PostTypeId = 2) * 100) AS AcceptedPercentage FROM Posts q INNER JOIN Posts a O... - 649523.46 342
1304 SELECT TOP 500 Users.Id as [User Link], Count(Posts.Id) AS Answers, CAST(AVG(CAST(Score AS float)) as numeric(6,2)) AS [Average Answer Score] FROM Posts INNER JOIN Users ON Users.Id = OwnerUser... - 638775.14 317
2754 (@UserId int)SELECT (CAST(Count(a.Id) AS float) / (SELECT Count(*) FROM Posts WHERE OwnerUserId = @UserId AND PostTypeId = 2) * 100) AS AcceptedPercentage FROM Posts q INNER JOIN Posts a O... usp_Q949 512301.03 161

Tuning and Monitoring

Query Store

Evaluate query performance

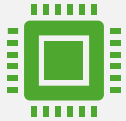
Lists queries across multiple interesting dimensions

- High CPU queries
- Longest Running queries
- Most Frequent queries
- High Read queries

Queries

Can all be sorted by total or average.

Can filter to only starred queries – allowing you to track specific queries.



Developer insight: Provides and overall view of various queries, and how they are working within the system.

Automatic Indexes

Using SQL's built-in automatic index management.

Identifies possible indexes

- Will create indexes, and monitor performance over time.
- Will revert an index if it doesn't help in the short term.
- May drop those indexes if they stop improving performance of the workload.

Never drops user created indexes

Summary

SQL DB can be your DB for analytis

Can still use what you know

Pricing and priority questions need to be resolved

Questions and Contact information

ginger.grant@desertislesql.com

desertislesql.com

desertislesql