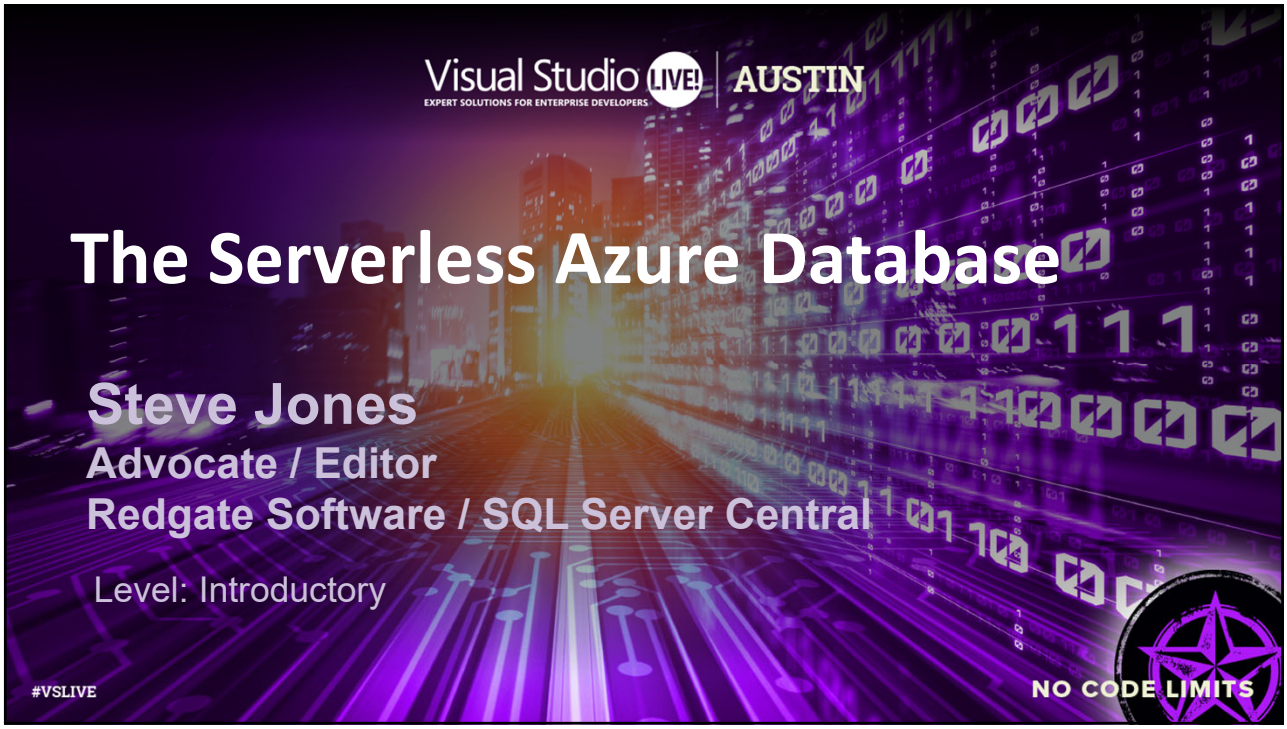


Agenda

- Who am I?
- The idea of serverless
- Azure SQL DB – Serverless
- How it works
- Synapse and Cosmos DB

Visual Studio LIVE!

Steve Jones

Advocate, Redgate Software
Editor, SQLServerCentral
he/him

30 years SQL Server data experience

DBA, developer, manager, writer, speaker in a variety of companies and industries

Founder, SQL Server Central

Currently the editor in chief, with the goal of helping you learn to be a better data professional every day

14-year Microsoft Data Platform MVP

I have been honored to be recognized by Microsoft for the as a Data Platform MVP working with SQL Server

/in/way0utwest @way0utwest sjones@sqlservercentral.com www.voiceofthedba.com

Visual Studio LIVE!

Lowering the administrative costs

THE IDEA OF SERVERLESS

Visual Studio LIVE!

Servers Have Overhead

- Maintenance (patching, monitoring, logs)
- Administrative support (security, deployments)
- Decisions about provisioning
  - Cloud or on-premises
  - Sizing for the workload
  - Adding additional servers for large workloads
- Lead time impacts availability and flexibility
  - Time to procure new hardware (on-premises)
  - Time to configure a new system
  - Reconfigure or deploy in the cloud

Enter Serverless

- This is a marketing term
  - Obviously, there are servers
  - You don't need to think about them
- Just-in-time computing
  - Features and computing provided on demand
  - Scalable as needed
  - Pay for what is used, as it is used
  - Code deployed to a server as needed
- Function-as-a-Service (FaaS)
  - Provide a place to run code as needed
  - All infrastructure is hidden from developers

The Current Cloud for App Functions

- Cloud computing requires lots of resources
- Some are not in use (rented) by customers
- The available compute time is sliced down to be used by the function call, as opposed to the server/VM/etc.
- Various offerings from vendors
  - AWS Lambda
  - GCP Google Cloud Functions
  - Azure Functions
  - IBM OpenWhisk

Visual Studio LIVE!

There is Demand

- Overprovisioning is a major source of cloud billing surprises
- Developers are taking advantage of linking lots of small services to power applications
- Speed of deployment increases is attractive
- Growth of IoT and microservices are suited for serverless
- Large scale MPP operations work well
- Vendors are offering more than FaaS
  - More complex compute platforms
  - Databases (our focus)

Visual Studio LIVE!

Azure Serverless Databases

- Azure SQL Database includes a serverless option
  - Paused compute when not in use*
  - No elastic pools
- Azure Synapse Analytics
  - Serverless SQL Pools
- Azure Cosmos DB
  - Serverless billing
  - Auto-scaling

Lowering the administrative costs

AZURE SQL DB – SERVERLESS

Visual Studio LIVE!

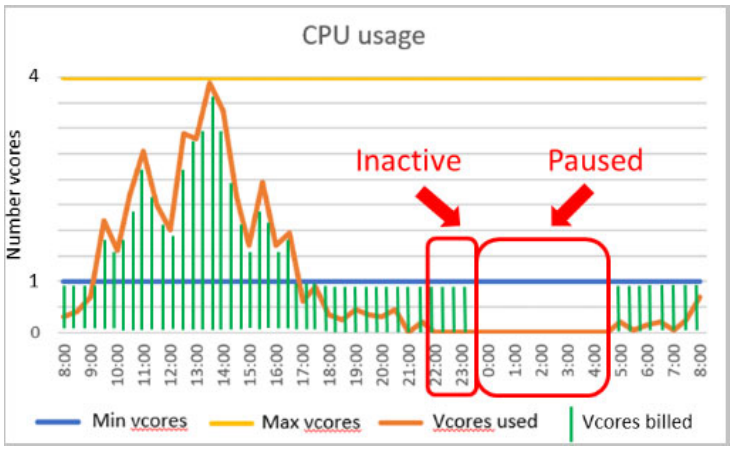
A Pause-able Database

- Azure SQL Database has a serverless option
- This allows you to provision an auto-scalable database
  - Set min vCores
  - Set max vCores
  - Specify pause parameter
- After an idle period, the database shuts down compute
- Only billed for storage during shut down periods
- Database “awakens” when needed

Visual Studio LIVE!

The Serverless Database Lifecycle

- Active database
  - Client workload running
- Inactive Period
  - Both must be true
  - Sessions = 0
  - CPU = 0 for user workload
- Paused Period
  - Inactive period meets auto-pause setting
  - Compute cost = 0
  - Storage billed at normal cost



Restarting the Database

- Auto-resume when
  - Login from client
  - Threat detection settings changed (server or database)
  - Changing sensitivity labels
  - View audit records or changing policy
  - Data masking changes
  - More (including "view" for many settings)
- First client connection request gives error
  - Code 40613
  - Need retry logic

The Specifications

- Gen 5 hardware (will change over time)
  - Other types are not available
- CPU
  - 0.5 vCores (min)
  - 40 vCores (max) – this will determine min vCores
  - Cannot set 0.5 min AND 40 max
- Memory
  - Proportional with vCore range
  - Scales up to 120GB
- Auto-pause
  - Can be enabled or disabled
  - 1 hour to 7 days

Memory Scaling

- Memory is scaled based on your min/max vCores
- Typically rule of 3 (max vCores x 3)
- Growth of cache memory similar to provisioned compute
  - Aggressive up to max
  - Quickly scales up based on workload
- Reclamation is more aggressive than non-serverless
  - Similar to memory pressure reclamation in SQL Server
  - Can result in less predictable performance

Limitations

- Auto-pause unavailable if you use:
  - Geo-replication
  - Long-term backup retention
  - Database is used for Elastic Job database
  - Others
- The database still auto-scales

Billing

- Goal is price-performance optimized solution
- Pay for storage always
- Pay for compute when used
  - Bill on vCores and memory used
  - Pay by the second
  - Includes the inactive (pre-pause) period
- If usage is below minimum, billed at minimum

Visual Studio LIVE!

Billing Details

- Billing per second
- Billing CPU and memory
- Memory normalized to compare to CPU
- Cost = vCore price * Max of (
  - Min vCores
  - vCores Used
  - (Min memory in GB) / 3
  - (Memory used in GB) / 3
- Always billed for storage
  - ~US$0.12/GB
  - Billed for db size + backups + redundancy
- App_cpu_billed metric aggregated over 1 minute and reported

Visual Studio LIVE!

Cost Scenarios

- Active database
  - Using 3 vCores
  - 8GB RAM (2.67 vCores normalized)
  - Cost: ($0.523 / vCore) * 3 * seconds in use
  - 3 vCores > 2.67 GB
- Low Usage
  - Min vCores 2
  - Using 1 vCores
  - Using 2GB RAM (.67 vCores normalized)
  - Cost: ($0.523 / vCore) * 2 vCores * seconds in use
  - 2 vCore > 1 > .67 GB

Visual Studio LIVE!

Cost Scenarios

- Low Activity database
  - Using .25 vCores
  - Using 1GB RAM (.33 vCores normalized)
  - Cost: ($0.523 / vCore) * .5 * seconds in use
  - 0.5 min vCore, .5 > .33 GB > 0.25
- High Memory Usage
  - Min vCores 0.5
  - Using 2 vCores
  - Using 12GB RAM (4 vCores normalized)
  - Cost: ($0.523 / vCore) * 4 * seconds in use
  - 4 GB > 2 vCores > 0.5

Visual Studio LIVE!

Check a connection and set up a Serverless Database

DEMO

Visual Studio LIVE!

Inside a Serverless Server

HOW IT WORKS

Visual Studio LIVE!

Architecture

- Data in storage (mdf/ndf/ldf)
- Database activated (deployed, unpaused)
  - Server instance with resource capacity chosen
  - Storage is attached
  - Recovery completes, database available
- Machine usually chosen to avoid interrupting workload
  - Based on max vCores

CPU Scaling

- CPU scales from min->max vCores based on workload
- Scaling for more CPU is usually subsecond
  - If capacity exists on the machine
- If no capacity, then load balancing occurs to a new machine
  - Similar to provisioned compute scale up
  - On the order of minutes
  - Connections dropped

Memory Scaling

- CPU and memory scale independently
- If CPU scaling is 2 -> 16 vCores
  - Memory is 6GB -> 48GB
- Memory-to-vCore ratio adapts
  - Up to 24GB to 1 vCore
- Scale-up time is sub-second (typically)
- Can take minutes if load balancing needed

Architecture

- 3 Tiers
- Control Plane with gateway and management service
- Data plane for compute
  - Tenants as collections of physical machines
- Storage Plane
  - Remote SSD
- More: https://youtu.be/E23D9iXSCJQ?t=832

From Pause

- Client opens connection
- First connection fails
- Next attempt, connection is transferred to data plane
- Find machine with capacity
  - Data files attached to instance
  - Once done, db is online
- Next connection succeeds once database is online
- Control Plane monitors the instance for capacity (RG manager)
  - If issues, service fabric will look for new machine
  - Orchestrate the workload to a new machine

Visual Studio LIVE!

While the Database Is In Use

- Memory manager monitoring the database
- If idle, will reclaim memory
- If needed, more memory allocated (up to max)

Visual Studio LIVE!

Serverless v Provisioned Compute

- Provisioned
  - Compute + Storage always on
  - Scale up/down is possible, but hard and slow(ish)
  - Billed for storage + compute 24/7
- Serverless
  - Storage always on, compute can pause automatically
  - Scaling happens quickly
  - Memory scales with usage
    - Aggressively, up and down

Use Serverless

- Single database with intermittent, unpredictable usage
- Periods of inactivity well over an hour
- Unknown usage
  - New app
  - Radial change to application or client base

Choose Provisioned

- Single database with regular, predictable workload
- Databases that cannot tolerate issues with memory trims or delays in resume
- Multiple databases with unpredictable workloads that can be in elastic pools

Visual Studio LIVE!

Inside a Serverless Server

SYNAPSE – SERVERLESS POOLS

Visual Studio LIVE!

SQL Pools

- Run T-SQL (ish) workloads
- One of two ways:
  - Dedicated (provisioned) – evolution of SQL DW
  - Serverless
- Serverless endpoint created for you with a Synapse workspace
  - Always 1 serverless SQL pool
- Accesses data stored in data lake, CosmosDB, or Dataverse
- No need to scale nodes. System handles this

Visual Studio LIVE!

SQL Pool Billing

- Based on data processed
  - Some things included, some not
  - Date read, intermediate results, data written to storage charged as compute
  - Storage always charged
- Set a budget for a time period (in TB)
  - Daily, Weekly and/or Monthly
  - Can set all or some of these
- Can set in portal or sp_set_data_processed_limit
- Budget is for data processed
- In-process queries not rejected if over limit
- New queries rejected

Visual Studio LIVE!

NoSQL Pay-as-you-go

COSMOS DB – SERVERLESS

Visual Studio LIVE!

Consumption Based Database

- Cosmos DB uses Request Units (RU) for billing
  - 1 RU ~= point read of 1KB document
- Provisioned Throughput – set min/max RUs/sec
- Serverless
  - Limited to 50GB/container
  - Billed on per-hour basis
  - Billed for storage separately
- Serverless must be chosen when creating a CosmosDB account
  - Is an account type
  - Limited to single region

Visual Studio LIVE!

Summary

- Serverless is an option using the vCore model for Azure SQL DB
  - Compute can be paused; storage always billed
  - Compared to Provisioned Azure SQL DB, costs can be lower
  - Good for bursty workloads
- Synapse always has a Serverless pool
  - Cost containment on data processed limits (day/week/month)
- CosmosDB has serverless
  - Limitations
  - Bill per use

Visual Studio LIVE!

The End

Thank you

Fill out Evaluations

Any Questions?

www.voiceofthedba.com

sjones@sqlservercentral.com

@way0utwest

/in/way0utwest

Visual Studio LIVE!

References

- https://docs.microsoft.com/en-us/azure/azure-sql/database/serverless-tier-overview#cache-reclamation

Visual Studio LIVE!

Images

Visual Studio LIVE!