Azure Best Practices and Configuration Notes

Bill Deitrick, BEMA Software Services - RX 2023

SQL on Azure

Hosting Models

1. SQL Server on Azure Virtual Machines
    1. Infrastructure-as-a-Service (IaaS)
2. SQL Server Managed Instances
    1. Platform-as-a-Service (PaaS)
3. Azure SQL Database (preferred for Rock)
    1. Database-as-a-Service (Daas)
        1. Azure SQL Database is simplest to operate and typically the most cost-effective option
    2. Significant benefit of the managed service: Fully automated, managed backups (with Point-in-time Recovery included up to 35 days retention with DTU).
    3. Rock is designed for compatibility with Azure SQL database, but there are some differences to keep in mind if coming from on-prem or one of the other hosting models in Azure.
        1. Three-part names only work if referencing the executing database; all queries with four-part names will fail in Azure SQL
        2. Cross-database queries are possible but work differently in Azure SQL
        3. Azure SQL servers always use UTC; make sure your queries explicitly set the time zone if using GETDATE() or similar
        4. There is no SQL Server Agent in Azure SQL; use Rock jobs or other Azure-native methods

Azure SQL Purchasing Models

1. There are two different purchasing models for Azure SQL. These represent the same service and are mostly functionally equivalent; the difference is how you you're paying for the service.
2. DTU Model
    1. "Database Transaction Unit" (DTU): A synthetic metric representing a blend of compute, memory, IO, and storage capacity.
    2. Storage can be increased in certain increments if needed.
    3. Tends to be the simplest to configure, manage, and operate.
    4. Cost-effective for most Rock workloads, offers the most predictable cost month-to-month (Point-in-Time backups included with up to 35 days retention for most tiers).
3. vCore Model

        1. Newer alternative to the DTU model; compute/memory/IO purchased independently of storage capacity
        2. Greater flexibility than DTU, can tune more closely to specific workloads
        3. Additional architectural options (Serverless, Hyperscale)
        4. Offers Reserved Capacity and Hybrid Benefit
            1. Reserved Capacity: make a 1 or 3-year purchasing commitment to save up to 70% on compute.
            2. Hybrid Benefit: save on SQL licensing portion of cost by bringing licenses you already own, covered by Software Assurance, to Azure.
    4. Choosing DTU or vCore
        1. DTU is appropriate for most churches
            1. DTU offers the most predictable monthly cost (point-in-time recovery backup storage included)
            2. DTU is simpler to configure and manage
            3. Reserved Capacity isn't available when using Sponsorship credits; most churches see greater savings using Sponsorship credits than Reserved Capacity.
        2. Reach for vCore when additional flexibility is needed or offers cost optimization over the DTU model

Azure SQL Performance Tiers

1. Azure SQL Database offers two different performance tiers Standard (DTU Model)/General Purpose (vCore Model) and Premium (DTU Model)/Business Critical (vCore Model)
    1. While the terminology is different, the two tiers are architecturally similar between DTU and vCore
2. Standard/General Purpose Tier
    1. Used by most churches for Rock workloads.
    2. Storage and compute are remote within the Azure datacenter; this translates to less IO potential than Premium/Business Critical.
    3. Lower cost than Premium/Business Critical
3. Premium/Business Critical Tier
    1. The largest churches with the busiest Rock environments should consider Premium tier/Business Critical
    2. Storage is all-flash, local to compute for superior IO performance
    3. Read-only replicas provide redundancy and read scale-out for Rock read-only contexts
    4. Higher cost than Standard/General Purpose Tier
4. When to choose Premium/Business Critical over Standard/General Purpose
    1. IO limits are being hit on Standard tier
    2. Operations against objects with large numbers of records (such as emails to 10s of thousands of recipients) are causing database timeouts, even on higher levels of the Standard tier

        1. Consider time-based automatic scaling between tiers if the workload is predictable to decrease cost
    3. Dataviews, reports, and pledge analytics are performing poorly or affecting performance of the rest of Rock

Disaster Recovery for Azure SQL Database

1. All Azure SQL databases have automatic, platform-managed point-in-time backups enabled by default
    1. Especially if running DTU model, consider increasing retention from the default 7 days to the max 35 days (the full 35 days is included with the cost for DTU, but there is a per-GB cost for vCore)
2. Ensure Geo-replication is enabled
    1. Geo-replication copies point-in-time backups to a paired region with a Recovery Point Objective (RPO) of 1 hour
3. Consider enabling Long-term Retention (LTR) backups for longer-term data retention
    1. If within the retention period, these can be restored even if the whole server object is deleted.
4. Perform test restores, and be familiar with the recovery tools and process in case you need them!

Azure SQL Firewall Configuration

1. Avoid the “Allow All Azure Access” option for SQL Server firewall settings.
    1. This grants access to other Azure subscriptions, not just your own!
    2. We want to explicitly allow VNETs, hosts, or other Microsoft services that should be able to connect.
        1. For supported Microsoft cloud services, such as PowerBI, deploy On-Prem Data Gateway to provide SQL access.
2. Utilize Service Endpoints or Private Link to connect the web server to Azure SQL
    1. Deployed to the subnet hosting your Rock server in your Azure VNET, both of these services will permit access to your database from any resource deployed in the subnet.
    2. Service Endpoints provide secure, optimized connectivity to Azure SQL’s public endpoints.
        1. These are simpler to deploy and recommended unless there is a specific need for Private Link functionality.
    3. Private Link creates a private endpoint for Azure SQL within your VNET.
        1. This service does have a nominal cost and is more complex to deploy than Service Endpoints
        2. Private Link offers additional functionality since the Azure SQL service is given a local endpoint in your VNET.
    4. Start with Service Endpoints, and deploy Private Link if the additional functionality offered by Private Link is required.

Additional Azure SQL Recommendations

1. Deploy a Delete Lock on the production database
    1. If a production *server* is deleted and no LTR backups have been configured for a particular database, there is not a straightforward path to recovering the database.
    2. Delete Locks prevent any delete actions agains the Azure Rest APIs, and guard your production database from accidental deletion of the database's server resource.
2. Enable Azure AD Database Authentication
    1. Enable Azure AD Authentication to provide Azure AD single sign-on for administrative and developer access to your database at the SQL level.
        1. This provides for centrally controlled access rather than a shared database password or individual passwords maintained at the database level.
    2. Be sure to enable *both* database accounts and Azure AD accounts to that Rock can still connect.
3. Enable Microsoft Defender for SQL
    1. Microsoft Defender is Microsoft's suite of enhanced security tools for Azure, including a SQL-specific option.
    2. This service is highly affordable at $15 per server per month.
    3. Defender for SQL scans for vulnerabilities and detects attacks like SQL injection.
4. Premium Tier Recommendations
    1. If deploying Premium or Business Critical SQL performance tiers, make sure you are fully leveraging the capabilities of that tier:
        1. Ensure the database is configured for Zone Redundancy; SLA increases from 99.99% to 99.995%
        2. Leverage the read-only replicas in your Rock configuration

Compute in Azure

VM SKU Recommendations

1. Recommended VM SKUs for Rock are B-series, Fsv2, or Dasv5 SKUs
    1. For the same number of cores, B-series is the least costly, followed by Fsv2 and then Dasv5 VMs
2. B Series (Burstable) VMs
    1. Usually the preferred option for Rock, suitable for all but the busiest Rock environments
    2. Do not have the full power of the CPU available all the time
    3. During periods of low utilization, credits are accumulated to "burst" to full CPU performance
        1. When burst credits are exhausted, performance is throttled
    4. Other Constraints
        1. Less disk throughput than other VM SKUs; this typically isn't an issue for Rock workloads

        2. Less bandwidth than similar F or D-series SKUs, this also typically isn't an issue for Rock workloads.
    5. B-Series VMs offer significant savings compared to other SKUs, particularly on Windows licensing
    6. Rock workloads tend to be bursty, not needing full CPU performance all the time (assuming a VM is sized appropriately)
    7. Start here for Rock workloads, and move to another VM series if needed.
3. Fsv2 (Compute Optimized) VMs
    1. Able to access full CPU performance at all times
    2. Higher CPU to memory ratio than other SKUs; most demanding Rock environments will hit CPU limitations before memory
    3. Constraints:
        1. If more memory is needed, another SKU (D-Series) with more a more balanced memory to CPU ratio may be needed
4. Dasv5 (General Purpose, AMD)
    1. Able to access full CPU performance at all times
    2. Higher memory to CPU ratio compare to F-series
    3. Consider if F-series does not provide sufficient memory

VM Configuration and Management Recommendations

1. Disk Layout and Configuration
    1. Configure Premium SSDs in for a 99.9% uptime SLA for the VM (standard SSDs offer a 99.5% SLA)
    2. Put Rock application files on a dedicated data disk rather than the OS disk (C Drive)
        1. This allows operations on the Rock data only at the block/management plane level.
        2. Moving wwwroot to an alternate location is straightforward in IIS manager
2. Patching
    1. Install Windows patches on a monthly cadence
    2. Configure automatic update management using one of the two Azure services:
        1. Update Management in Azure Automation
            1. This is the older solution (no longer in development), does not support Windows Server 2022
        2. Azure Update Management
            1. This is the new solution, currently in preview, but is currently under active development and receiving new features
3. Authentication
    1. Avoid domain-joining Rock application servers in Azure if possible
    2. Deploy the AAD Authentication extension to facilitate centrally-managed administrative authentication on the Rock Server
4. Consider Cost Optimizations: Hybrid Benefit and Reserved Instances or Savings Plans
    1. Hybrid Benefit

            1. Use existing licenses (with current Software Assurance) to cover the licensing cost of Windows on Azure VMs
            2. This typically isn't worth exploring for B-series VMs because Windows licensing costs are significantly reduced for these VMs, but may be worth exploring for F or D-series
    2. Reserved Instances/Savings Plans
        1. Reserved Instances
            1. Commitment to purchase a a specific family of VM SKUs in a specific region for 1 or 3 years
            2. Up to approximately 70% savings over pay-as-you-go costs for Compute
        2. Savings Plan
            1. More flexible than reserved instances, but less compute savings
            2. Commitment to a certain spending level for Azure VMs for 1 or 3 years
    3. Reserved Instances and Savings Plans cannot be purchased on a subscription using Azure sponsorship benefits
5. Manage CPU Quota
    1. Ensure extra CPU quota is available if needed; this allows scaling up when there is an unexpected increase in traffic

Disaster Recovery for Virtual Machines

1. Utilize an Azure Recovery Services Vault to back up your Rock VM
    1. Ensure your vault is configured for geo-redundancy (the default configuration)
    2. Configure an appropriate schedule and retention policy if something other than the defaults (nightly backup, 30-day retention) is desired
    3. Test restoring data to ensure familiarity with the tool!

Networking

1. Deploy or migrate to Standard SKU Public IP Addresses (Basic Public IP Addresses will be retired in 2025)
2. Deploy Network Security Groups at the firewall level
3. Deploy IP address ranges in your VNETs and subnets compatible with your existing IP space in case VPN connectivity is required
4. Avoid VPNs if at all possible
    1. VPNs add expense, complexity, and additional points of failure
    2. Keeping Rock isolated from on-prem creates a natural security boundary

Monitoring and Alerting

Azure Monitor

1. Azure Monitor is the native metrics, logging, and alerting tool provided by Microsoft

Azure Monitor Metrics

1. Azure Monitor collects performance metrics for all resources in a time-series database
   1. Metrics are retained for 90 days by default
   2. Different resource types will have different metrics available
2. Metrics are available in the configuration blade for each resource; use Dashboards to create a single-pane-of-glass for metrics from multiple resources
   1. Get started with the BEMA template

Azure Monitor Alerts

1. Azure Monitor Alerts provide notification functionality based on Azure Monitor Metrics and Logs
2. Alert Rules define notification conditions
3. Action Groups control the destination of alerts
4. Alert Processing Rules allow alert scheduling (i.e., turning off alerts for SQL when overnight maintenance jobs run)

Development Environments

Development Environment Recommendations

1. Most churches should have a development environment for testing upgrades and exploring new features
2. Development environments should be:
   1. Fully separated from production
      1. Development environments can run on the same VMs and SQL servers as production, but this is not the ideal configuration.
      2. Full separation/completely independent environments for VMs ensures that actions taken in the development environment will not affect production, and avoid mistakes such as accidentally connecting development to the production database
   2. Scaled down from production
      1. Development environments should generally run with less powerful configurations than production, and can be scaled up temporarily when needed
   3. Deallocated/scaled down when not in use
      1. Azure Automation Runbooks can be used to automatically spin up development environments during business hours and spin them down at the end of the day
   4. Have a fresh copy of production data

        1. The most useful development environments will resemble production as closely as possible
        2. Azure Automation Runbooks can be used to automate the process for refreshing development environments on-demand or on a schedule
    5. Enforcing HTTPS
        1. Even development environments should be configured for HTTPS
            1. Using an OS-level tool such as Certify the Web or Win-acme in development provides flexibility over the ACME plugin for Rock

Orchestration and Automation

Orchestration and Automation Overview

1. Automating regular operations and defining environments as code provides repeatable, self-documenting processes for managing Azure resources that can increase operational efficiency

Automation Tooling in Azure

1. Azure Automation
    1. Azure Automation accounts are a go-tool tool for automation and orchestration of Rock-related Azure resources
    2. Automation Runbooks enable execution of custom PowerShell scripts to manipulate Azure resources
        1. Runbooks have all of the Azure Management cmdlets available, and custom modules (such as the sqlserver module) can optionally be installed
        2. Runbooks can be executed on a schedule or on-demand
    3. Automation accounts offer a free tier; 500 minutes of runtime per-month are provided for free
2. Azure Functions
    1. In an automation context, Azure Functions are a more general option fur running arbitrary code more targeted at building APIs than Azure resource orchestration
    2. Functions can be triggered on a schedule or with HTTP requests
    3. Azure Functions offer a great choice when implementing control of Azure resources within Rock
        1. Standard Rock functionality, such as Lava webrequest or workflow actions, can be used to trigger Azure functions
    4. Azure Functions have a free tier available
3. Azure Logic Apps
    1. Azure Logic Apps offer a visual development experience similar to Rock workflows

    2. Logic Apps can be used for some orchestration, and can be triggered by Azure Monitor Alert rules
4. Managed Identity
    1. Leverage System Assigned Managed Identity to give Automation Accounts, Functions, and Logic Apps permissions to Azure resources

Automation and Orchestration Example Use Cases

1. Scheduled SQL Scaling
    1. Scaling SQL for known periods of higher utilization is a common means of providing capacity when needed and cost savings when higher performance is unnecessary
    2. Prior to the release of the Triumph plugin, Automation Runbooks were commonly used to accomplish this
2. Automatic Scaling and allocation/deallocation for development environments
    1. Powering off dev environments and scaling down development databases outside of business hours can significantly reduce development environment cost
    2. Automation Runbooks can be used to schedule development environment availability in accordance with business hours
3. Automating development environment refreshes
    1. Manual development environment refreshes are tedious and error prone
    2. Automation Runbooks can be used to automate refreshes, taking this to a single-click operation that typically completes in around 15 minutes
4. Control/Orchestration of Azure Resources
    1. Function Apps can provide an API surface to control Azure resources from within Rock
    2. This can be a great way to allow users without Azure access to control the availability or trigger refreshes of development or sandbox environments

Infrastructure as Code

1. Most Rock environments have been deployed by pointing and clicking in the Azure portal
2. Infrastructure as Code allows defining and managing environments and resources as code instead of creating them manually
    1. Infrastructure as code is self-documenting (can read the code to determine how resources have been configured), and produces consistent results each time
3. Azure Bicep is typically the best tool for creating infrastructure as code templates for Azure
4. Bicep templates can be a great way to automate spinning up new development or sandbox environments