Plan and Implement Data Platform Resources

1. deploy database offerings on selected platforms

- SQL Server Virtual Machine
  - Go to Azure SQL
    - This is not a single service, but a group of SQL services.
      - +Create, SQL database.
  - Server name needs to be unique throughout Azure.
  - Admin account password must be at least 12 characters long.
  - In additional settings, add existing data if required, either from a backup or sample data.
  - You should use Managed Service Accounts (MSA) for a single computer running a service.
    - A Group Managed Service Account (gMSA) is used for assigning the MSA to multiple servers.
- Azure SQL Managed Instance
  - Go to Azure SQL, +Create, SQL managed instance.
  - You need:
    - Subscription and Resource group,
    - Managed instance name,
    - Region,
    - Managed instance admin login and password.
- Azure VM with SQL Server
  - Create a Virtual Network.
    - In IP Addresses – default (subnet) – Edit subnet , add the Service Endpoint "Microsoft.Sql"
  - Create the Azure VM
    - In Networking, select "Private endpoint", then "+Add private endpoint" and select the subnet from above.
  - When created, in "Firewalls and virtual networks", click "+Add client IP", and "Allow Azure services and resources to access this server".

2. configure customized deployment templates

- See topics 9, 10 and 11.

3. apply patches and updates for hybrid and IaaS deployment

- IaaS deployment means VM, a hosted infrastructure.

  - PaaS SQL Database and Managed Instance have built-in patching, and they always use the latest stable Database Engine version.

- Operating systems updates are as per standard Microsoft Windows Update.
  - Windows Update may do some updates automatically.
- Patches are as per standard SQL Server Patches.
  - You have full control of the database engine, e.g. when to apply patches.
  - However, you can also enable “Automated Patching”
    - You need SQL Server 2008 R2 or above, and Windows Server 2008 R2 or above.
    - In SQL Server settings which creating the VM, or
    - In the Azure resource, go to Settings – SQL Server configuration – Patching.
    - It may take a few minutes for this to be configured.
    - Settings include day, start hour, window duration
      (number of minutes to download and install).
  - You can configure it when:
    - creating a new VM (under SQL Server Settings tab – Automated patching), or
    - for existing VMs, by going to Azure Portal – the relevant VM – Settings – SQL Server configuration – Patching.
    - By using PowerShell, with:
      - "New-AzVMSqlServerAutoPatchingConfig -Enable" setting the schedule, and
      - "Set-AzVMSqlServerExtension" installing the extension with the schedule.
  - You can also enable automatic registration.
    - This daily checks whether there are any unregistered VMs in the subscription, and if so, registers them in lightweight mode.
      - To take advantage of all of the features, you would still need to manually upgrade.
    - To do this, go to Azure Portal – SQL virtual machines (plural) – and at the top, click on "Automatic SQL Server VM registration".

4. evaluate requirements for the deployment

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
PaaS	PaaS	IaaS
Azure manages the database.		You need to manage your VM, and gives you control of the database.
Resources always running unless dropped (apart from serverless, when paused).	Resources always running unless dropped.	You can shut down resources when not in use.
	Best for most migrations to the cloud. May need some changes.	Lift-and-shift. As easy as moving from one on-prem server to another.
		Higher cost than PaaS.
Best for modern cloud applications, and fast time-to-market for new solutions are needed.	Best for new applications or existing on-prem applications for use in the cloud.	Best when you don't want any database changes, or when you require OS-level access.
Can use Azure Hybrid Benefit (Windows Server for VMs only, and SQL Server licenses with Software Assurance) and reserved capacity to reduce cost.		
Hybrid Benefit does not apply to serverless.		

5. evaluate the functional benefits/impact of possible database offerings

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
Most commonly used SQL Server features	High compatibility with SQL Server	All on-premises capabilities.
Trace flags not supported.	Only a limited number of (global) trace flags are supported.	Trace flags supported.
Built-in backups, patching and recovery.		You manage backup and patches
Databases up to 100Tb	Databases up to 8Tb	Instances up to 256 Tb. (Databases of up to instance size)
Supports serverless compute [the alternative is "provisioned"]		Lift-and-shift.
CLR not supported.	CLR supported.	
Based on latest stable Enterprise Edition.	Based on latest stable Enterprise Edition.	Choose which version of SQL Server (from 2008R2 updates), edition (Developer, Express, Web, Standard, Enterprise) and OS you use.
Can use Elastic Job Agent service.	Can use SQL Agent jobs.	Can use SQL Agent jobs.

6. evaluate the scalability of the possible database offering

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
Databases up to 100 Tb (hyperscale)	General Purpose: Instances up to 8Tb Business Critical: 1-4 Tb. Up to 100 databases.	Instances up to 256 Tb. Up to 50 instances per server. Databases of up to instance size
Size of single database or elastic pool can be changed as needed.	Size can be changed as needed.	Size of VM can be changed as needed.
You can also add more compute power (vertical scaling) or shard your data into multiple database nodes (horizontal scaling).	Can add more compute power (vertical scaling), but not easily sharding (though you can do it more manually).	Compute power of VM can be changed as needed
You can also change service tier from Standard/General Purpose (Premium disks) to Premium/Business Critical (SSDs).	You can change between Premium and Business Critical only.	
For more, see topic 9 and 13.	For more, see topic 10.	

7. evaluate the HA/DR of the possible database offering

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
Up to 99.995% availability. Minimum SLA is 99.99% availability, except for Hyperscale, which is 99.9%-99.95%.	99.99% availability	Up to 99.99% availability. However, this requires a second VM, and using Always On availability group. Minimum SLA is 95% for the VM.
At the Basic, Standard and General Purpose level, can use Locally redundant availability. At the Premium and Business Critical level or elastic pools, can use a three- to four-node cluster with Locally or Zone Redundant Availability. You can also add read-only replicas (the "Read Scale-out" feature).		Can configure availability replicas, using a domain controller VM.
At the General Purpose level using Gen5 compute hardware in certain regions, can use Zone redundant configuration (preview).		

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
Automatic backups, including full, differential and transaction log, for 7-35 days.		Can configure backups.
Can configure full database backups to Azure Storage for long-term backup retention (LTR).	Can perform copy-only backups for long-term backup retention (preview).	Can configure backups.
Point-in-time restores		With appropriate backups, can do point-in-time restores.
Can configure Active geo-replication (up to 4 readable secondary databases)	No.	Can configure geo-replication storage (asynchronously). Data file and log file needs to be on the same disk.
Can configure auto-failover groups (not Hyperscale)	Can configure auto-failover groups	Can configure Azure failover cluster instances using shared storage.

8. evaluate the security aspects of the possible database offering

Azure SQL Database	Azure SQL Managed Instance	SQL Server on Azure VM
Auditing works at the database level.	Auditing works at the server level.	Auditing works at the server level.
.xel log files are stored in Azure Blob storage.	.xel log files are stored in Azure Blob storage.	Events are stored in the file system or Windows event logs.
Can use Azure Defender for SQL, which includes: Vulnerability assessment and Threat detection (costs around $0.02/instance/hour)		
Data encryption, using Transport Layer Security (TLS), Transparent Data Encryption (TDS) and Always Encrypted. Firewalls.		
SQL authentication or Azure Active Directory authentication		Windows or SQL Server authentication.

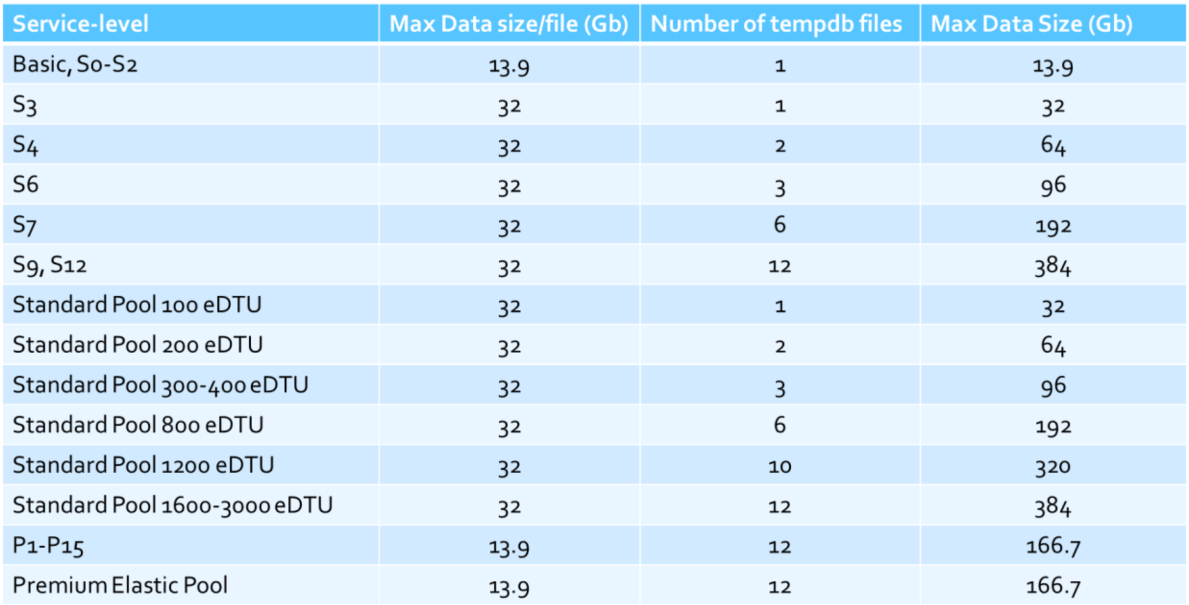
9. configure Azure SQL Database for scale and performance

- Azure SQL Database can be:
  - Single database.
  - Elastic pool.
  - Compute tier:
    - Provisioned – for regular usage patterns, or multiple databases with elastic pools.
    - Serverless computer – on-off usage with a relative low average compute.
      - Supports automatic pausing and resuming.
      - When the database is paused, you only pay for storage.
- Azure SQL Database allows for:
- vCore purchasing model
  - Specify separate amount of Number of vCores, memory, and amount/speed of storage. Look at:
    - Data Maximum Memory size,
    - Log size (30% of the Data Max size),
    - iOPS,
    - Concurrent workers, and
    - Backup retention.
  - Maximum of:
    - 80 vCores at Gen5,
    - 4 Tb memory, and
    - 4 Tb database size (apart from Hyperscale, which has up to 100 Tb).
  - Azure Hybrid Benefit and/or reserved capacity
    - Azure Hybrid Benefit allows you to bring in your existing on-prem licenses to the cloud.

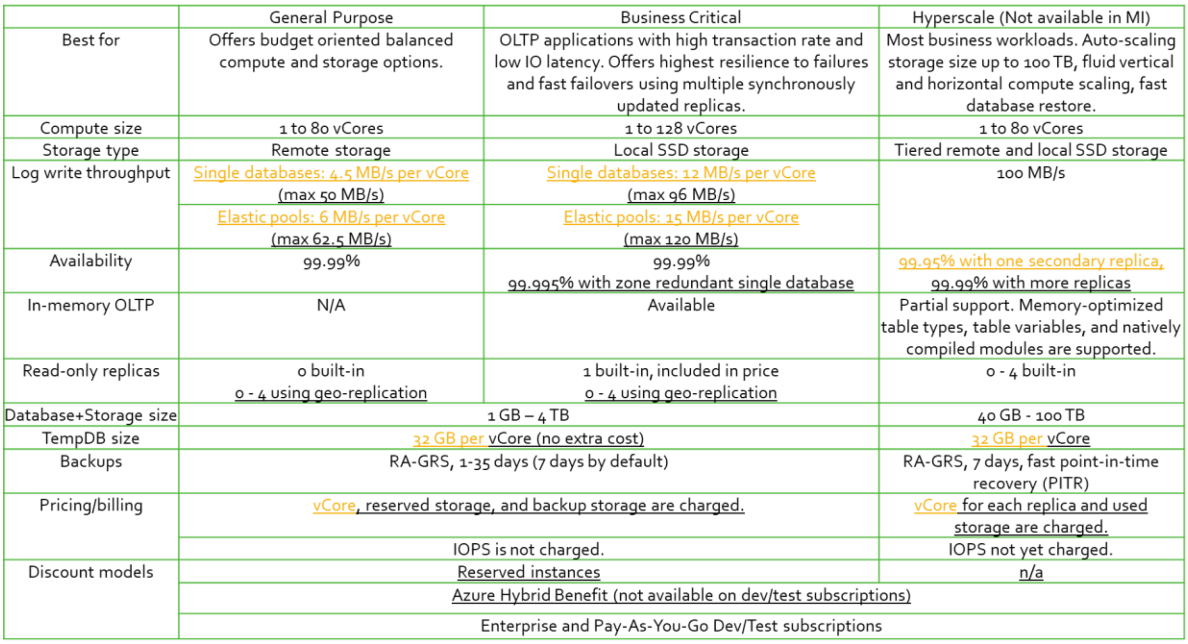
        - Reserved capacity is paying in advance at a discount.
    - Uses local SSDs. Provision in 1 Gb increments.
    - Choose from:
        - General purpose (scale computer and storage) – For most business workloads. Storage latency of 5-10 ms (about the same as SQL Server on a VM).
        - Business critical (high transaction rate and high resiliency) – uses
            - When you need low-latency I/O (1-2 ms) or frequent communications between app and database.
            - Large number of updates, or long running transactions that modify data.
            - Higher resiliency, availability and fast geo-recovery and recovery from failures, and advanced data corruption protection.
            - Free-of-charge secondary read-only replica.
        - Hyperscale (on-demand scalable storage) – Only for Azure SQL Database – say 100 Tb+ storage.
            - You cannot subsequently change out of Hyperscale. Cost the same as Azure SQL Database.
    - Backup Storage Redundancy
        - Geo-redundant backup storage (default and recommended),
        - Zone and Local Redundancy are cheaper for single region data resiliency.
    - Max data size:
        - 512 Gb for GP_S_Gen5_1
        - 1,024 Gb for GP_S_Gen5_2, 4, 6
        - 1,536 Gb for GP_S_Gen5_10 and
        - 3,072 Gb for GP_S_Gen5_12, 14, 16, 18, 20.
        - 4,096 Gb for GP_S_Gen5_24, 32 and 40.
    - Tempdb
        - Azure SQL Database creates 1 file per vCore with 32Gb per file, with caps of up to 32 files for serverless computing only.
- DTU-based purchasing model
    - For light to heavy database workloads.
    - Offers bundles of maximum number of compute, memory and I/O (reads/writes) resources for each class (cannot separate them).
    - Uses Azure Premium disks. Provision in increments of 250 Gb to 1 Tb, and 256 Gb thereafter.
    - Choose from:
        - Basic (for less demanding workloads)

        - Standard (for typical performance)
        - Premium (for I/O-intensive workloads)
        - Please note: Basic and Standard S0, S1 and S2 have less than 1 vCore, and cannot use "Change data capture".
            - Consider Basic, S0 and S1, where database files are stored in Azure Standard Storage (HDD), for development, testing and infrequently accessed workloads.
    - Consider changing to vCores if > 300 DTUs
        - Might reduce costs – no downtime when converting.
    - Calculates available for calculating DTUs needed:
        - See https://dtucalculator.azurewebsites.net/
- Can change service tier on demand (but not out of Hyperscale).
    - Don't do it when you have a long job running!
    - For the DMVs to have accurate figures, you may need to flush the Query Store after re-scaling. Use:
        - EXEC sp_query_store_flush_db;
- Choose from server or serverless model:
    - Server.
        - This is a logical server, which includes logins, firewall and auditing rules, policies and failover groups.
        - The server name must be unique in Azure.
        - Don't need to manage the instance.
        - Cannot use "USE" to change database context.
    - Serverless model.
        - Bills for compute per second.
        - Pauses databases and billing in inactive periods.
- Configure network:
    - No access.
    - Public/private endpoint.
        - You can "Add current client IP address".
    - Choose whether to "Allow Azure services and resources to access this server" (for other Azure services).
        - Or if not, you could allow specific Virtual Networks to have access.
- Connection policy

  - Proxy – uses Azure SQL Database gateways,
  - Redirect – Establish directly to the database node,
  - Default – Redirect if connection originates inside Azure, and Proxy if outside Azure.
- You can have sample data, or data based on the restore from a geo-replicated backup.
- Choose database collation:
  - CS/CI = case-[in]sensitive,
  - AS/AI = accent-[in]sensitive.
- You can select a free trial of Azure Defender.
  - Identify and mitigate potential database vulnerabilities and threat detection.
- It uses the Full Recovery model.
  - This should not be changed for Azure SQL Database.



Service-level	Max Data size/file (Gb)	Number of tempdb files	Max Data Size (Gb)
Basic, S0-S2	13.9	1	13.9
S3	32	1	32
S4	32	2	64
S6	32	3	96
S7	32	6	192
S9, S12	32	12	384
Standard Pool 100 eDTU	32	1	32
Standard Pool 200 eDTU	32	2	64
Standard Pool 300-400 eDTU	32	3	96
Standard Pool 800 eDTU	32	6	192
Standard Pool 1200 eDTU	32	10	320
Standard Pool 1600-3000 eDTU	32	12	384
P1-P15	13.9	12	166.7
Premium Elastic Pool	13.9	12	166.7



	General Purpose	Business Critical	Hyperscale (Not available in MI)
Best for	Offers budget oriented balanced compute and storage options.	OLTP applications with high transaction rate and low IO latency. Offers highest resilience to failures and fast failovers using multiple synchronously updated replicas.	Most business workloads. Auto-scaling storage size up to 100 TB, fluid vertical and horizontal compute scaling, fast database restore.
Compute size	1 to 80 vCores	1 to 128 vCores	1 to 80 vCores
Storage type	Remote storage	Local SSD storage	Tiered remote and local SSD storage
Log write throughput	Single databases: 4.5 MB/s per vCore (max 50 MB/s)	Single databases: 12 MB/s per vCore (max 96 MB/s)	100 MB/s
	Elastic pools: 6 MB/s per vCore (max 62.5 MB/s)	Elastic pools: 15 MB/s per vCore (max 120 MB/s)	
Availability	99.99%	99.99% 99.995% with zone redundant single database	99.95% with one secondary replica, 99.99% with more replicas
In-memory OLTP	N/A	Available	Partial support. Memory-optimized table types, table variables, and natively compiled modules are supported.
Read-only replicas	0 built-in 0 - 4 using geo-replication	1 built-in, included in price 0 - 4 using geo-replication	0 - 4 built-in
Database+Storage size	1 GB – 4 TB		40 GB - 100 TB
TempDB size	32 GB per vCore (no extra cost)		32 GB per vCore
Backups	RA-GRS, 1-35 days (7 days by default)		RA-GRS, 7 days, fast point-in-time recovery (PITR)
Pricing/billing	vCore, reserved storage, and backup storage are charged.		vCore for each replica and used storage are charged.
	IOPS is not charged.		IOPS not yet charged.
Discount models	Reserved instances		n/a
	Azure Hybrid Benefit (not available on dev/test subscriptions)		
	Enterprise and Pay-As-You-Go Dev/Test subscriptions		

10. configure Azure SQL Managed Instance for scale and performance

- Service Tier:
  - General Purpose
    - Most workloads (default option),
  - Business Critical
    - low-latency workloads
    - High resiliency to failures
    - Fast Failovers
- Hardware Generation
  - Compute and memory limits.
  - Up to 80 vCores,
  - 400 Gb memory,
  - Up to 100 databases, and
  - up to 16 Tb database size.
- Features in Azure MI not in Azure SQL Database include:
  - Cross-database queries,
  - Common language runtime (CLR)
    - The execution environment for .NET framework code (also known as "managed code").
    - CLR in SQL Server is called CLR integration.
  - SQL Agent, and

- The msdb system database.
- You can manually make a copy-only backup of a database (not instance).

- vCore compute model
  - SQL Managed Instances does not support the DTU-based purchased model.
  - See topic 9 for more details of the vCore compute model.
  - Cannot use Hyperscale in Azure SQL MI.
- Deploys a dedicated ring (aka "Virtual cluster") for your data.
- Tempdb
  - MI creates 12 files, regardless of the number of vCores.

	General Purpose	Business Critical
Best for	Offers budget oriented balanced compute and storage options.	OLTP applications with high transaction rate and low IO latency. Offers highest resilience to failures and fast failovers using multiple synchronously updated replicas.
Compute size	4, 8, 16, 24, 32, 40, 64, 80 vCores 2 vCores also available in pools	4, 8, 16, 24, 32, 40, 64, 80 vCores
Storage type	Remote storage	Local SSD storage
Database+Storage size	32 GB – 8 TB	32 GB – 4 TB
TempDB size	24 GB per vCore (max about 2 TB)	Up to 4 TB - limited by storage size
Log write throughput	3 MB/s per vCore (max 22 MB/s)	4 MB/s per vcore (max 48 MB/s)
Availability	99.99%	99.99% 99.995% with zone redundant single database
In-memory OLTP	N/A	Available
Read-only replicas	0 built-in	1 built-in, included in price
	0 - 4 using geo-replication	0 - 4 using geo-replication
Backups	RA-GRS, 1-35 days (7 days by default)	
Pricing/billing	vCore, reserved storage, and backup storage is charged.	
	IOPS is not charged	
Discount models	Reserved instances Azure Hybrid Benefit (not available on dev/test subscriptions) Enterprise and Pay-As-You-Go Dev/Test subscriptions	

11. configure SQL Server in Azure VMs for scale and performance

- SLA for Virtual Machines
  - 95% (18 days) - Standard HDD Managed Disks,
  - 99.5% (1.8 days) - Standard SSD Managed Disks,
  - 99.9% (8 hours) - Premium SSD or Ultra Disks,
  - 99.95% - 2+ instances in the same Availability Set
    - Different computers in the same datacenter
  - 99.99% - 2+ instances in 2+ Availability Zones in the same Azure region
    - Different physical datacenters, with independent power, cooling and networking.
- When to use SQL Server in Azure VMs:
  - When you need an older version of SQL Server or access to a Windows Operating System.
  - When you need SSAS (Analysis), SSIS (Integration) or SSRS (Reporting) (non Azure services),

  - When you need features not available in Azure SQL Database or Azure MI.
- Best practices – VM Size:
  - Choose at least 4 vCPUs.
  - Memory optimized VMs for best SQL Server workloads.
  - Higher memory-to-vCore ratio for mission critical or data warehouses.
  - Azure VM marketplace images are configured for optimal SQL Server performance.
- Best practice – Storage:
  - Place data, log and tempdb files on separate drives.
    - Data drives should be put on Premium P30 and P40 disks for cache support.
    - Log drive should be put on Premium P3o to P80 disks, or Ultra disks for submillisecond latency.
    - Tempdb should be placed on the D drive on the SSD.
  - Stripe multiple data disks using Storage Spaces (similar to RAID, but done in software) to increase I/O bandwidth. 3+ drives form a storage pool. This should be done by:
    - Creating the individual disks,
    - Creating a storage pool,
    - Creating a single virtual disk, from these resiliency types.
      - Simple
        - Needs at least 1 physical disk
        - Stripes data physical disks.
        - Maximizes disk capacity and increases throughput.
        - No resiliency (does not protect from disk failure)
        - Use for high-performance where resiliency is not required by striping.
      - Mirror
        - Needs at least 2 physical discs to protect from single disk failure.
        - 2-3 copies of the data.
        - Increases reliability, but reduces capacity.
        - Greater data throughput and lower access latency.
        - Use for most deployments.
      - Parity
        - Needs at least 3 physical discs to protect from single disk failure.
        - Stripes data and parity information across disks.
        - Increases reliability, but reduces capacity.

            - Increases resiliency.
            - User for archive and backups.
        - Creating a volume.
    - Use Local Redundant Storage, not Geo-redundant storage, on the storage account.
- Six different series are available:
    - General purpose – balanced CPU-to-memory.
        - Good for testing and development, small-medium databases or traffic web servers.
    - Compute optimized – high CPU.
        - Good for medium traffic web servers, network appliances, batch processes, and application servers.
    - Memory optimized – high memory (up to 4 Tb).
        - Good for relational database servers, medium to large caches, and in-memory analytics.
    - Storage optimized – high disk throughput
        - Good for Big Data, SQL, NoSQL databases, data warehousing and large transactional databases.
    - GPU – specialized virtual machines
        - heavy graphic rendering and video editing, as well as model training and inferencing (ND) with deep learning.
    - High performance compute – fastest machines
        - Most powerful CPU virtual machines
- Best practices – SQL Server:
    - Enable database page compression where appropriate.
    - Enable backup compression and instant file initialization.
    - Limit autogrowth and disable autoshrink.
    - Use one tempdb data file per core, up to 8 files.
    - Apply any cumulative updates for your version of SQL Server.
    - Register with the SQL IaaS Agent Extension for:
        - Automated backup,
        - Automated patching,
        - Azure Key Vault integration,
        - View information in Azure Portal about your SQL Server configuration, and more.
        - It is installed when you deploy an SQL Server VM from the Azure Marketplace.
    - Enable Autoshutdown for development and test environments.

12. calculate resource requirements

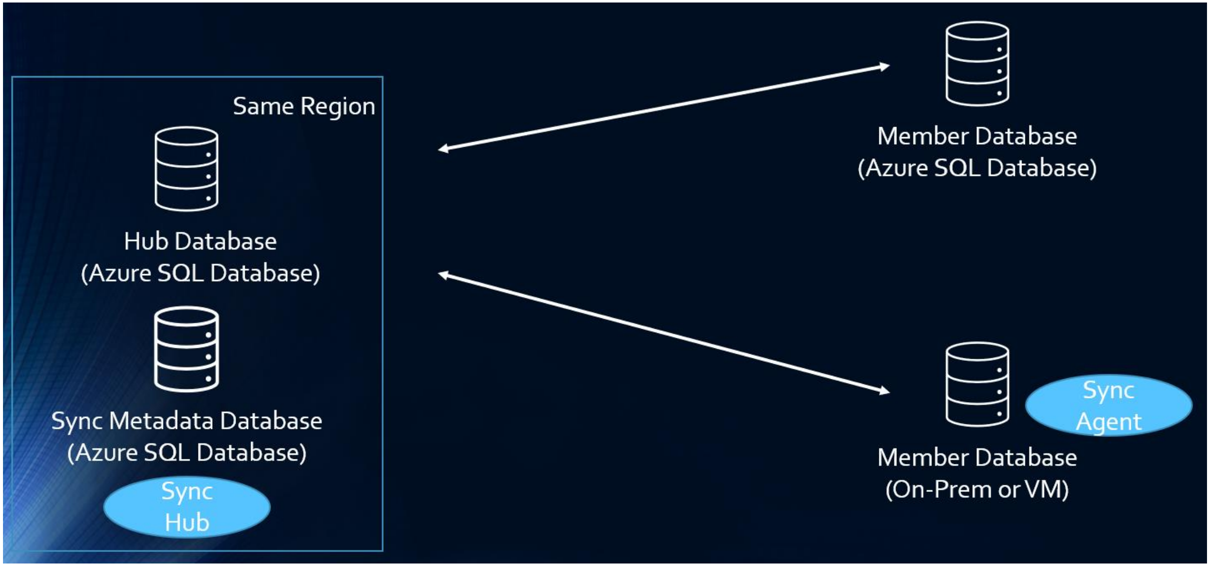
- Purchasing models:
    - DTU Model – a package of compute, storage and IO resources.
    - Simple, pre-configured resources.
- vCore-based model:
    - Independent scaling of compute, storage and IO resources.
    - Flexible, control and transparency
    - Use with Azure Hybrid Benefit for cost savings.
    - Business Critical service tier includes 3 replicas (and about 2.7x price)
- For both model:
    - Single database.
        - They can be moved in/out of elastic pool.
        - They are isolated from others and is portable.
        - They can be dynamically (i.e. manually) scaled (but not autoscaled) up and down.
    - Elastic pool.
        - Assign resources which are shared by all pool databases.
        - Can dynamically scale or autoscale resources up/down.
        - This is for multiple databases, good when they have variable usage patterns.
        - Can add databases by going to the pool and clicking on "+Add databases".
- Storage costs:
    - Based on amount provisioned – that's your maximum database size.
- Calculate costs from https://azure.microsoft.com/en-us/pricing/details/azure-sql-database/single/
- For DTU model, consider the following factors when determining how many DTUs you need:
    - Maximum storage bytes [for all databases in the pool],
    - Average DTU utilization x Number of databases,
    - Number of concurrently peaking databases x Peak DTU Utilization per database.
    - Note: Unit price for eDTU pools is 1.5x the DTU unit price for a single database.
        - Price for v-Core pools is at the same unit price as for single databases.

13. evaluate database partitioning techniques, such as database sharding

- Why partition?
    - Storage space limitations
        - Maximum storage capacity can be reached on a server.
    - Computing resource limits

        - Exceeding this may result in time out.
    - Network bandwidth
        - Exceeding this can result in failed requests.
- You can scale vertically:
    - Add disk capacity, processing power, memory and network connections.
    - However, you may reach the same problem later.
- You can partition the data store horizontally into shards.
    - Each shard has its own subset of the data.
    - It runs on a server acting as a storage node.
- You can:
    - Scale out by adding further shards
    - Use off-the-shelf hardware for each storage node
    - Balance the workload across shards
    - Locate shards near to the users who will use it.
- You can do it by implementing:
    - Lookup strategy
        - Have a shard key (an ID), and a map which shows where the data is stored.
        - Offers more control.
        - Requires additional overhead.
    - Range strategy
        - Use sequential shard keys in ranges (e.g. one per month).
        - Similar data is kept on the same storage node, so it can retrieve multiple items in a single operation.
        - Doesn’t necessarily provide optimal balancing.
        - Rebalancing shares is difficult.
    - Hash strategy
        - Data distributed evenly among the shards. Reduces hotspots (high loads for an individual server) by using some random element for distribution.
        - More even data and load distribution.
        - Computing the hash might increase overhead requirements.
        - Rebalancing shards is difficult.

14. set up SQL Data Sync



- Azure SQL Data Sync allows you to synchronize data across multiple databases.
  - Tables need to have a primary key, which cannot be changed (rows can be deleted/recreated instead).
  - Does not work with Azure SQL Managed Instance.
- You define one Azure SQL Database as the Hub Database.
- Sync Metadata Database contains the metadata and log for Data Sync. It is an Azure SQL Database in the same region as the Hub Database.
  - It should be an empty database. Data Sync creates tables and runs a frequent workload.
- Member databases are either Azure SQL Database or on-prem (not Managed Instance).
  - If you are using on-prem, you will need to install and configure a local sync agent.
    - Download it from https://www.microsoft.com/en-us/download/details.aspx?id=27693
- A sync group has these properties:
  - Sync Schema shows what data is synchronized.
  - Sync Direction can be both ways or only one direction.
  - Sync Interval is how often synchronization happens.
  - Conflict Resolution Policy is “Hub wins” or “Member wins”
    - But if there are several members, this depends on which member syncs first.
- Use in:
  - Hybrid Data Synchronization.
  - Distributed Applications, including Globally Distributed Applications.
- To set up a database and Sync Metadata Database:
  - Go to Azure portal – SQL databases.

  - Go to the Hub database.
  - Go to “Sync to other databases”.
  - Go to “New Sync Group”, and select:
    - Sync Group Name (not the database name),
    - Sync Metadata Database (recommended that a new one is created),
    - Automatic Sync (If on, choose from Seconds, Minutes, Hours or Days in Sync Frequency),
    - Conflict Resolution (Hub win or Member win),
    - Use private link (a service managed private endpoint). If yes, you will later need to approve the Private Endpoint Connection.
- To add sync members:
  - Open the Sync Group – Databases.
  - Click on “Add an Azure Database” or “Add an On-Premises Database”.
- For Azure Database, select:
  - Sync Member Name (not the database name),
  - Subscription,
  - Azure SQL Server and Database,
  - Sync Directions (To the Hub, From the Hub, or Bi-directional Sync),
  - Existing Username and Password for the member database,
  - Use private link.
- For on-prem SQL Server database:
  - Select “Choose the Sync Agent Gateway”.
  - Select “Existing agent” or “Create a new agent”. If new:
    - Download the “Azure SQL Data Sync Agent”,
    - Enter an Agent Name,
    - Select “Create and Generate Key”, and copy it to the clipboard, then click OK.
- On the on-prem SQL Server:
  - Run the Client Sync Agent app.
  - Click “Submit Agent Key”.
  - In the “Sync Metadata Database Configuration”, enter credentials for the metadata database server.
    - If automatically created, this will be the same server as the hub database.
    - You may need a firewall rule, created in the portal or in SSMS.
  - Click Register.

  - In the “SQL Server Configuration” box, connect using SQL Server or Windows authentication.
  - Click Test connection and Save.
- In the Portal, in “Configure On-Premises page”:
  - Select “Select the Database”.
  - Provide a name for the new sync member (not the database name) and the Sync Directions.
- To see if it works, go to the Database Sync Group page – Tables, and click on Refresh schema.
  - It may take a while for the data to be refreshed.

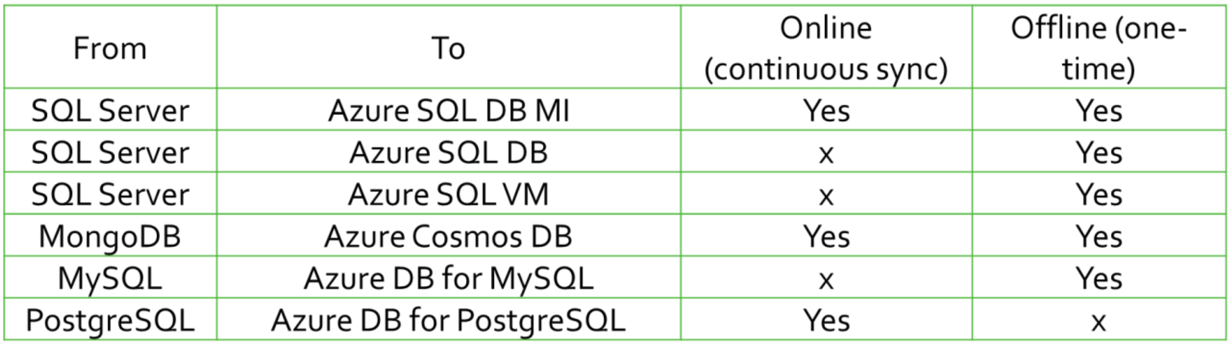
15. evaluate requirements for the migration

  - What workloads you intend to migrate.
  - The actual resource requirements:
    - Hard drive space,
    - Compute size (processing power),
    - Version of SQL Server needed.
      - This will impact on whether you can use Azure SQL Database/Managed Instance, or whether you need a VM.
    - Version of Windows Server needed (if any).
  - Downtime allowances
    - Are you allowed any downtime at all? If not, you need to do an online migration.
  - Dependences between databases, and between databases and applications.
  - Security requirements
  - Backup and restore requirements
  - Current limitations, and future limitations.
  - Location for data storage (e.g. GDPR, California Consumer Privacy Act, or similar requirements)

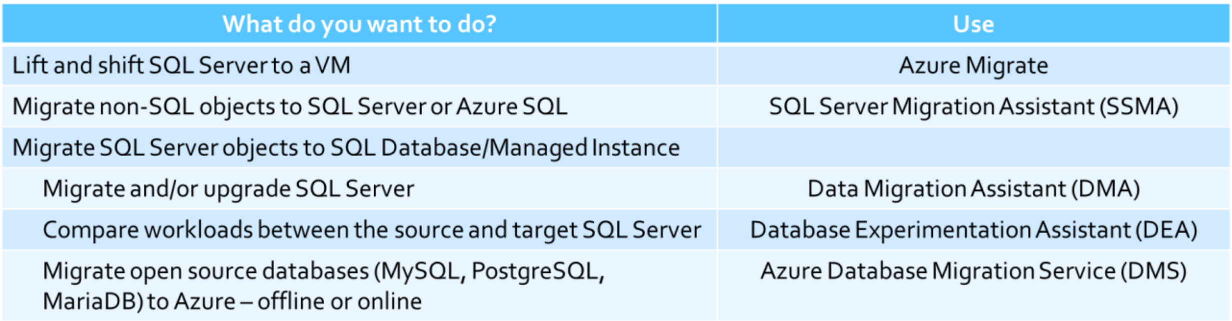
16. evaluate offline or online migration strategies

- Do you need instead to lift and shift SQL Server to a Virtual Machine?
  - If so, use Azure Migrate.
  - It can also discover and assess SQL data estate at scale (across your data center).
  - Get Azure SQL deployment recommendations, target sizing and monthly estimates.
- Do you need to migrate non-SQL objects, such as Access, DB2, MySQL, Oracle and SAP ASE databases to SQL Server or Azure SQL?
  - If so, use the SQL Server Migration Assistant (SSMA).
- Do you need to migrate SQL Server objects to SQL Database/Managed Instance? If so:

- Do you need to migrate and/or upgrade SQL Server?
    - If so, use Data Migration Assistant (DMA).
    - It can help migrate to Azure SQL Database, or to a VM or to another on-prem server.
    - It can also discover and assess SQL data estate, and recommend performance and reliability improvements for your target environment.
    - Detect compatibility issues between your current database and a target version of SQL Server or Azure SQL.
    - Move your schema, data, and uncontained objects.
- Do you need to compare workloads between the source and target SQL Server?
    - If so, use the Database Experimentation Assistant (DEA).
    - Capture the workload of a source SQL Server environment.
    - Identity compatibility issues.
- Do you need to migrate open source databases, such as MySQL, PostgreSQL or MariaDB?
    - If so, use the Azure Database Migration Service (DMS).
    - Minimal downtime (especially if online using the Premium pricing tier). Good for large migrations.
    - You need:
        - To allow outbound point 443 (HTTPS) – you may also need 1434 (UDP).
        - Enable the TCP/IP protocol.
        - Create an Azure SQL Database instance, have a server-level firewall rule to allow DMS access, and have CONTROL DATABASE permission on the target database.
        - Have CONTROL SERVER permissions on the source.
        - Does not initiate any backups, and uses existing full and log backups (not differential).



From	To	Online (continuous sync)	Offline (one-time)
SQL Server	Azure SQL DB MI	Yes	Yes
SQL Server	Azure SQL DB	x	Yes
SQL Server	Azure SQL VM	x	Yes
MongoDB	Azure Cosmos DB	Yes	Yes
MySQL	Azure DB for MySQL	x	Yes
PostgreSQL	Azure DB for PostgreSQL	Yes	x



What do you want to do?	Use
Lift and shift SQL Server to a VM	Azure Migrate
Migrate non-SQL objects to SQL Server or Azure SQL	SQL Server Migration Assistant (SSMA)
Migrate SQL Server objects to SQL Database/Managed Instance	
Migrate and/or upgrade SQL Server	Data Migration Assistant (DMA)
Compare workloads between the source and target SQL Server	Database Experimentation Assistant (DEA)
Migrate open source databases (MySQL, PostgreSQL, MariaDB) to Azure – offline or online	Azure Database Migration Service (DMS)

17. evaluate requirements for the upgrade

- You can upgrade to SQL Server 2016 or 2017 from SQL Server 2008+.
- You can upgrade to SQL Server 2019 from SQL Server 2012+.
    - If using an earlier version, databases can be migrated.
- You can also upgrade to a higher version of the same year.
- You can also upgrade to a higher version except from Enterprise (the highest):
    - SQL Server 2012+ Business Intelligence can upgrade to Enterprise.
    - Standard (or in older versions, Workgroup or Small Business) can upgrade to Standard or Enterprise.
    - Web can upgrade to Web, Standard or Enterprise.
    - Developer can upgrade to Developer, or SQL Server 2019 (only) Web, Standard or Enterprise.
    - Express can upgrade to Express, Web, Standard or Enterprise.
    - Evaluation can upgrade to Evaluation, Express, Web, Standard or Enterprise.
- If an application requires a previous version, you can use that version's compatibility level.
    - For example, SQL Server 2016 is compatibility level 130.

18. evaluate offline or online upgrade strategies

- Offline
    - You can upgrade SQL Server.
    - However, you cannot do an offline upgrade a 32-bit instance to a 64-bit instance.
    - You cannot add new features during the upgrade (but you can do it afterwards).
- Online
    - You need to do a side-by-side installation, and then decommission the previous SQL Server.
    - You can choose what features to use, and you can install a 64-bit instance, even if your previous version is 32-bit.

19. implement an online migration strategy

- See topic 20.

20. implement an offline migration strategy

- Migrating from SQL Server to Azure SQL Database - Prerequisites:
  - Enable TCP/IP protocol on your SQL Server instance.
  - Download the Data Migration Assistant.
  - Create a Virtual Network for the Azure Database Migration Service using either ExpressRoute or VPN.
  - Enable outbound port 443 of ServiceTag for ServiceBus, Storage and AzureMonitor.
  - Allow database engine access in Windows firewall, and open the Windows firewall to TCP port 1433 (unless you have changed it). You may also need to have UDP port 1434.
    - Note: port 3306 is for MySQL and 5432 is for PostgreSQL.
  - Create a server-level IP firewall rule to allow Azure Database Migration Service access.
  - Your credentials need CONTROL SERVER on the SQL Server instance, and CONTROL DATABASE on Azure SQL.
- Check for blocking issues:
  - In Data Migration Assistant, select +New and Assessment, and enter a project name.
  - Select Database Engine, SQL Server and Azure SQL Database, and either/both:
    - Check database compatibility – identifies partially supported or unsupported features which may block migration. There will be recommendations.
    - Check feature parity – recommendations, different approached, and mitigating steps.
  - Click Next and provide connection details to SQL Server.
  - Select databases, click Add and Start Assessment.
- To migrate sample schema:
  - In the DMA, click +New, Project Type, Migration.
  - Add a project name, SQL Server, and Azure SQL Database
  - Select “Schema only” in “Migration scope”.
  - Click Create, and enter your SQL Server details.
  - Click Next, then enter your Azure SQL details.
  - Click Connect, and select the relevant database.
  - Click Next, and specify the schema objects to be deployed.
    - By default, all of them are selected.
  - Click ”Generate SQL script” then “Deploy schema”.
- Register the Microsoft.DataMigration resource provider
  - In the Azure portal, go to Subscriptions.

  - In the Azure Database Migration Service subscription, click on Resource providers.
  - Find “Microsoft.DataMigration” and Register it.
- Create an instance of the Azure Database Migration Service
  - In the Azure portal, go to this service and click Create, and select:
    - Subscription, Resource Group, Name
    - Location, ”Azure” as Service mode, and
    - Pricing tier (Standard is free)
      - Standard is free for offline (one-off) migration only.
      - Premium is about $1 for 3 hours. Allows for online (continuous migration) and offline migrations, and faster speeds.
      - You can have the 4 vCore Premium DMS free for 6 months. You can use it for a total of 1 year, and create 2 DMS services per subscription.
  - On ”Create Migration Service”, select an existing VN or create a new one.
- Create a migration project
  - In the Azure portal, go to “Azure Database Migration Services”, select the relevant instance, and select ”New Migration Project”.
  - Add a project name, SQL Server, Azure SQL Database, and Data migration.
  - Click on “Create and run activity”.
- Enter source settings
  - Enter connection details (Fully Qualified Domain Name or IP Address),
  - If you haven’t a trusted certificate, check “Trust server certificate”.
  - Select databases, note the Expected downtime, and click “Next: Select target:.
- Enter target settings
  - Enter target details.
  - Click “Next: Map to target databases”. This will be mapping to new databases, unless you have a database with the same name.
  - Click “Next: Configuration migration settings and select tables to be affected.
  - Click “Next: Summary” and enter an Activity Name for the migration.
- Run the migration:
  - Click “Start migration”. You can monitor the migration from there.
  - Once complete, verify that the target database has been migrated.
- Other options:
  - Bulk Copy Program (bcp) can be used for connecting from on-prem or a VM to Azure SQL.
  - BULK INSERT – loading data from Azure Blob storage.

  - SSIS packages – ETL (extract, transform and load).
  - Spark or Azure Data Factory

21. implement an online upgrade strategy

- Side-by-side upgrade.
  - Verify that the hardware and software you intend to use is supported.
  - Backup databases, and check they can be restored.
  - Identify reports to be used to check that upgrade has been a success.
  - Install new instance of SQL Server.
    - If using Analysis Services, make sure you install the correct server mode – tabular or multidimensional.
  - Attach/restore each database.
  - Run DBCC to check for database integrity.
  - Test upgrade has been a success using relevant reports.
  - Backup and restore.
- Databases run with the previous compatibility level setting.

22. implement an offline upgrade strategy

- Upgrade strategy:
  - Go to SQL Server Installation Media – Maintenance – Edition Upgrade.
  - Enter Product Key (if applicable) and accept license terms.
  - Select the SQL Server instance to upgrade.
  - On the screen ”Read to Upgrade Edition”, click Upgrade and Close.
  - Reboot if necessary.
- There may be additional steps if upgrading from SQL Server Express.

Implement a Secure Environment

23. configure Azure AD authentication

- Azure SQL Database supports:
  - SQL Server authentication (user name and password, sent in plain text), and
  - Azure Active Directory (AD) authentication.
    - AAD can sync with on-prem Windows Server AD.
- Azure AD authentication supports:
  - Cloud-only identities,
  - Hybrid identities that support cloud authentication with Single Sign-On (SSO), using password hash or pass-through authentication.

  - Hybrid identities that support federated authentication.

- Decision tree:
  - Cloud-only identities
    - Azure AD to handle sign-in completely in the cloud
    - Do not want to enforced AD security policies during sign-in.
    - Do not have a sign-in requirement not natively supported by Azure AD.
  - Federated authentication
    - If you want to integrate with an existing federation provider, or
    - Have a sign-in requirement not natively supported by Azure AD.
  - Pass-through authentication
    - All other cases
      - Do not have a sign-in requirement not natively supported by Azure AD.
      - No integration with an existing federation provider, OR want to enforce user-level AD security policies during sign-in.
- Other authentications:
  - Apps running on an Azure VM – passwordless authentication.
  - Apps running on a non-Azure machine that is domain-joined: use managed identities
  - Apps running on a non-Azure machine that is not domain-joined: use certificate
  - Admin tools on a non-Azure machine that is not domain-joined: use Azure AD integrated authentication, or Azure AD interactive authentication with multifactor authentication.
  - Older apps where you can't change the connection string: SQL authentication.
- AAD can allow additional security such as Multi-Factor Authentication (MFA).
  - Go to the Azure Portal – Active Directory – (The relevant active directory, if more than one), and Authentication methods. These include:
    - FIDO2 (Hardware) Security Key
    - Microsoft Authenticator (Phone App)
    - Text message and
    - Temporary Access Pass
- 23. configure Azure AD authentication
- To add a new user:
  - In Azure portals – go to Azure Active Directory.
  - Go to Users – New user.
  - Enter:

        - Name
        - User name (similar to an email address), either @[DomainName].onmicrosoft.com, or a custom domain name.
        - Groups (Optional).
        - Azure AD role (Optional),
        - Job info (Optional).
    - Azure will give you an autogenerated password in Password.
    - Click Create.
    - You can delete a user from the same place.

24. create users from Azure AD identities

- You should create a second admin account as an Azure AD account, with the db_owner database roles.
- This is for Azure SQL Managed Instance.
    - Azure SQL Databases use a separate login, e.g.:
        - CREATE LOGIN MyLogin
        - WITH PASSWORD = 'MyComplexPassword';
        - CREATE USER MyLogin FOR LOGIN MyLogin;
        - GO
    - Logins can:
        - Do SQL Agent management and jobs execution,
        - Database backup and restore operations,
        - Auditing,
        - Trigger logon triggers, and
        - Setup Service Brokers and DB mail.
- Database users cannot be created using the Azure Portal.
- However, you can create logins from Azure AD users, groups or apps.
    - Syntax for Azure SQL Managed Instance
        - CREATE LOGIN login_name [FROM EXTERNAL PROVIDER] { WITH <option_list> [,..]}
    - The parameters are:
        - login_name is an existing Azure AD UserPrincipalName of the user, DisplayName group or app when used with the "FROM EXTERNAL PROVIDER" indicates Azure AD Authentication.
        - <option_list> ::=

      - PASSWORD = {'password'} – this cannot be used when FROM EXTERNAL PROVIDER is used.
      - | SID = sid
      - | DEFAULT_DATABASE = database
      - | DEFAULT_LANGUAGE = language

- Create user:
  - CREATE USER user_name
    - [{ FOR | FROM } LOGIN login_name]
    - | FROM EXTERNAL PROVIDER
    - [WITH <limited_options_list> [,...]]
  - [;]
  - <limited_options_list> ::=
    - DEFAULT_SCHEMA = schema_name
    - | DEFAULT_LANGUAGE = { NONE | lcid | language name | language alias }
    - | ALLOW_ENCRYPTED_VALUE_MODIFICATIONS = [ON | OFF]]
- Both SQL Server Administrators and Azure Active Directory Administrators for SQL Server can create:
  - Users based on SQL Server Authentication logins.
  - Contained database users based on SQL Server Authentication (without logins)
- Azure Active Directory Administrators for SQL Server only can create:
  - Contained database users based on Azure AD users and groups
- create SQL Server logins
- You cannot create an SQL Server login from the Azure portal.
  - But you can create an Azure AD admin.
- Create a login using SSMS (Managed Instance only):
  - [Name of server] – Security – New – Login…
  - Enter the user name in the Login name, or click Search…
  - Select type of authentication:
    - Windows authentication,
    - SQL authentication (you will need a password if so),
    - "Azure Active Directory – Universal with MFA [Multi-Factor Authentication] support"
      - Strong verification.
    - "Azure Active Directory – Password"

      - Uses identities in Azure AD. You can use it when your computer is logged into Windows but it is not federated with Azure.
    - "Azure Directory – Integrated"
      - When connecting from a Windows which is a federated domain.
    - Special purpose logins, which cannot connect to SQL Server, but which can own objects and have permissions:
      - "SQL user without login" (no password),
      - "Mapped to [stand-alone security] certificate",
      - "Mapped to [stand-alone] asymmetric key",
      - "Mapped to [security] Credential".
  - Select Default database (Master is the default), and
  - Select Default language.
- Create a user using SSMS (Managed Instance and Azure SQL Database):
  - [Name of server] – Security – New – User…
  - Select one of the following user types:
    - "SQL user with login"
      - Used when a person outside of your organization connects.
    - "SQL user with password". Also called a "contained database user". You can select
      - User must change password at next login
      - Enforce password expiration, and
      - Enforce password policy.
      - Used when a person outside of your organization connects.
    - "SQL user without login" (no password),
      - Can make your database more portable. Allowed in Azure SQL Database and in a contained database in SQL Server.
      - Cannot login but can be granted permissions.
    - "Mapped to [stand-alone security] certificate",
      - Cannot login to a server, but can be granted permissions and can sign modules
    - "Mapped to [stand-alone] asymmetric key",
      - Cannot login to a server, but can be granted permissions and can sign modules
    - "Windows user".

25. configure security principals

- Principal is that which receives permissions.
  - Server-level: logins and server roles
  - Database-level: users, database roles, application roles.
- Securables are that which can be secured.
  - In a server, in a database, in a schema.
- Fixed Server-wide Login permissions (MI and SQL Server in VM only)
  - sysadmin – any activity.
  - serveradmin – change server-wide configuration options and shut down the server.
  - securityadmin – GRANT, DENY and REVOKE server-level permissions, and any database-level permissions if they have access to the database.
    - This allows them to assign most server permissions.
  - processadmin – end processes.
  - setupadmin – add/remove linked servers.
  - bulkadmin – can run the BULK INSERT statement
  - diskadmin – managing disk files.
  - dbcreator – create/alter/drop/restore any database
  - public – includes all users, group and roles. When you want the same permission(s) for everyone.
- There are also Fixed Database Roles:
  - db_owner – all configuration and most maintenance activities (in Azure SQL Database, some activities require server-level permissions), including DROP database.
    - However, if you give them db_denydatareader or DENY permissions, you can deny read access to data.
  - db_securityadmin – can modify role membership for custom roles only and manage permissions. Can elevate own permissions.
  - db_accessadmin – add/remove access to the database for logins and groups.
  - db_backupoperator – can back up the database in MI or VM (not applicable in Azure SQL Database, as BACKUP/RESTORE commands not available).
  - db_ddladmin – run DDL command.
  - db_[deny]datareader – [cannot] read all data from all user tables and views.
  - db_[deny]datawriter - [cannot] add/delete/change data in all user tables.
- You can also add custom roles.
- In Azure SQL Databases, there are also two special database roles in the "master" database only:
  - dbmanager – can create/delete databases. Connects as the dbo (database owner) user.

    - loginmanager – create/delete logins in the "master" database (as per securityadmin server role in on-prem SQL Server)

- You can use:
    - sp_helprotect – returns user permissions for an object (or all objects) in the current database.
        - Does not list standard fixed server/database role permissions.
        - Not available for Azure SQL Database.
    - sp_helprole – lists the database roles.
    - sp_helprolemember – direct members of a role.
        - Not available for Azure SQL Database.
- There are also role-based access control (RBAC), which are security rights outside of databases, which include:
    - SQL DB/Managed Instance/Server Contributor – manage SQL Databases, Mis or Servers, but not get access to them. Cannot manage security-related policies.
    - SQL Security Manager – mange security-related policies for servers and databases, but no access to them.
- When deploying, Azure uses the "server admin", which is a principal in Azure SQL Database, and a member of the sysadmin role in MI.

26. configure database and object-level permissions using graphical tools

- This is for MI and VM (not Azure SQL Database).
- In a particular login:
    - Click Search.
    - Select:
        - “The server”,
        - “Specific objects”. If so, click “Object Types” and select Endpoints, Logins, Servers, Availability Groups and/or Server roles.
        - “All objects of the types” – select Endpoints, Logins, Servers, Availability Groups and/or Server roles.
- Objects that can be secured include:
    - Server
        - Availability group, Endpoint, Login, Server role and Database
    - Database
        - Application role, Assembly, Asymmetric key, Certificate, Contract, Fulltext catalog, Fulltext stoplist, Message type, Remote Service Binding, (Database) Role, Route, Schema, Search property list, Service, Symmetric key, User

- Schema
  - Type
  - XML schema collection
  - Object: Aggregate, Function, Procedure, Queue, Synonym, Table, View, External Table

27. apply principle of least privilege for all securables

- Users should have the least privilege that is necessary for them to do their job.
  - This is called the Least-privileged User Account (LUA).
- You can use Roles to assigned permissions to roles, and then users to roles.
  - This makes security administration more easy.
- You can use the following permissions:
  - GRANT
  - REVOKE (the opposite of GRANT)
    - Why use REVOKE instead of GRANT? It doesn't give permissions, but it doesn't stop permissions if they have it through another role.
  - DENY (this overrides a GRANT).
    - DENY does not apply to sysadmin members or object owners.
    - If DENY is applied to the public role, no non-sysadmin will have this permission.
- You can also prevent users from querying objects directly by allowing only access to procedures or functions.
- Objects are chained together.
  - If two objects have the same owner, then permissions in a second object called from the first are not separately checked.
- You should still reduce the number of objects affected by permissions.
  - SELECT permission in a database includes all (child) schemas, and the tables and views.
  - SELECT permissions on a schema includes all of the tables and views.
  - SELECT on a table gives SELECT permission only.
  - CONTROL gives ownership-like permissions and includes all other permissions, including ALTER, SELECT, INSERT, UPDATE.

28. implement Transparent Data Encryption (TDE)

- TDE (de-)encrypts data at the page level at rest. It is encrypted when written, and de-encrypted at read.

    - Don't confuse this with TLS – transparent layer security – which encrypts when in transit.
- It uses a symmetric Database Encryption Key (DEK).
    - You can Bring Your Own Key (BYOK).
- It is protected by the TDE protector, using a service-managed certificate or an asymmetric key in the Azure Key Vault.
    - For Azure SQL Database, it is set at the server level. New databases are encrypted by default (but not ones created through restore or database copy).
    - For Azure SQL Managed Instance, it is set at the instance level and is inherited to all encrypted databases.
    - It cannot be used to encrypted system databases.
- To enable it in Azure SQL Database only, go to the Azure Portal, then the relevant database, then go to “Transparent data encryption” and set “Data encryption” to ON.
    - You cannot do this in Azure SQL Managed Instance.
- In T-SQL, you can use:
    - ALTER DATABASE DatabaseName SET ENCRYPTION ON/OFF.
    - This can be used in Azure SQL Managed Instance.
    - However, you can’t switch the TDE protector to a key in Key Vault in T-SQL.
- You can also use PowerShell.
    - Set-AzSqlServerTransparentDataEncryptionProtector
        - Change to ServiceManaged or Azure Key Vault
    - Add-AzSqlServerKeyVaultKey
        - Adds a Key Vault key to a SQL server
    - Set-AzSqlDatabaseTransparentDataEncryption
        - Modifies TDE property for a database.
- You can also use REST API.

29. implement object-level encryption

- See topic 33.

30. implement Dynamic Data Masking

- Dynamic Data Masking is for both Azure SQL Database and Azure SQL Managed Instance.
    - It prevents access to sensitive data by putting a mask, with none or part of the data (e.g. last 4 digits of a credit card).
    - This encrypts the column (column-level encryption – CLE).
- To implement Dynamic Data Masking:
    - In the Azure Portal, go to the Database, then go to Dynamic Data Masking.

- You may see recommended fields to mask.
    - If so, you can click on "Add mask" (and then Save).
- To create a custom rule, click "Add mask".
- You can select the Schema, Table and Column to define the columns for masking.
- You can select the mask to be displayed:
    - Default value (0, xxxx, 01-01-1900),
        - XXXX for string data types. You can use fewer Xs if it less than 4 characters.
        - Use 0 for numeric data types.
        - Use 01-01-19000 for date and time data types.
    - Credit card value (xxxx-xxxx-xxxx-1234),
        - Exposes the last 4 digits of the credit card, with a constant string prefix.
    - Email (aXXX@XXXX.com),
        - Exposes the first letter, but replaces everything else with a constant string prefix.
    - Number (random number range),
        - A random number between two boundaries.
    - Custom string (prefix [padding] suffix).
        - Shows the first X characters, the last Y characters, and a custom padding string in the middle.
- Click "Add" to save this rule.
    - You can also "Discard" changes and "Delete" the mask.
- In T-SQL, this is done by using:
    - ALTER TABLE Schema.Table ALTER COLUMN ColumnName
    - ADD MASKED WITH (FUNCTION = 'partial(1, "xxxxx", 1)') – or 'email()' or 'random(1, 1000)' or 'default()'

- You can select specific SQL users who were excluded from masking.
    - Multiple users are separated by semicolons.
    - Note: Administrators are always excluded for Dynamic Data Masking – they can always read the data.
    - In T-SQL, this is done by using:
        - GRANT UNMASK to MyCustomRole;
        - GRANT UNMASK to MyUser;
        - REVOKE UNMASK to MyUser;

31. implement Azure Key Vault and disk encryption for Azure VMs

- To encrypt disks for Azure VMs:
    - In the Azure Portal, go to the VM.
    - Select Disks (left-hand side),
    - Select Additional Settings (at the top).
    - In "Encryption settings – Disks to encrypt", select "OS and data disks".
    - Then click "Select a key vault and key for encryption".
    - Next to "Select key from Azure Key Value: Key vault", select "Create new".
    - Add a name (unique amongst Azure Key Vaults) and Resource Group.
    - Go to the "Access Policies" tab, click "Enable Access to: Azure Disk Encryption for volume encryption".
    - After creating the Key Vault, leave the Key field blank, click Select, and Save.

32. configure server and database-level firewall rules

- By default, all connections to the server and database are rejected.
    - SQL Database communicates over port 1433. You need that opened on your own computer/server.
- For the most secure connection:
    - Set "Allow access to Azure services" to NO, then
    - create a reserved IP (classic deployment) for the resource that needs to connect, then
    - allow access through the IP address.
    - A public IP address is required for each resource.
- What the difference?
    - Server-level firewall rules are for users/apps to have access to all databases.
    - Database firewall rules are for an individual or app.
    - Database rules are checked before server-level rules.
- To set up a server-level firewall rule:
    - This applies to all databases in the server on Azure SQL Database only, whether single or pooled databases. It does not apply to Azure SQL Managed Instance.
    - You will need SQL Server Contributor or SQL Security Manager role, or the owner of the resource that contains the Azure SQL Server.
    - In Azure portal, go to your database.
    - On the database Overview page, click "Set server firewall".
    - Select "Add client IP" to add your current IP address. This opens port 1433.

    - A firewall rule of 0.0.0.0 enables all Azure services to bypass the server-level firewall rule – but in the portal, you need to turn on "Allow Azure services and resources to access this server" instead.
  - Click OK. The rules are then stored in the master database.
- In T-SQL:
  - To check the current server-level IP firewall rules:
    - SELECT * FROM sys.firewall_rules
  - To add a server-level IP firewall rule:
    - EXECUTE sp_set_firewall_rule @name = N'MyFirewallRule',
    - @start_ip_address = '192.168.1.1', @end_ip_address = '192.168.1.200'
  - To delete a server-level IP firewall rule:
    - EXECUTE sp_delete_firewall_rule @name = N'MyFirewallRule'
- You can also manage using PowerShell, CLI (Command Line Interface) or REST API.
- To set up a database firewall rule.
  - It can only be done using T-SQL statements, and you need CONTROL DATABASE permission at the database level.
  - You need to have set up a server-level firewall rule first.
  - Run a query such as:
    - EXECUTE sp_set_database_firewall_rule N'Example DB Rule','0.0.0.4','0.0.0.4’;
  - This rule is stored in that individual database.
- In T-SQL:
  - To check the current database-level IP firewall rules:
    - SELECT * FROM sys.database_firewall_rules
  - To add a database-level IP firewall rule:
    - EXECUTE sp_set_database_firewall_rule @name = N'MyDatabaseFirewallRule',
    - @start_ip_address = '192.168.1.1', @end_ip_address = '192.168.1.200'
  - To delete a database-level IP firewall rule:
    - EXECUTE sp_delete_database_firewall_rule @name = N'MyDatabaseFirewallRule'

33. implement Always Encrypted

- You can encrypt sensitive data using Always Encrypted in Azure SQL Database and MI.
- If you wish to use an Azure Key Vault, then you need to create it first
  - Use the Azure Portal – Key Vault to create it.
    - You need the following permissions:

    - Cryptographic Operations: Decrypt, Encrypt, Unwrap Key, Wrap Key, Verify and Sign.
    - Key Management Operations: create, get, list.
  - It costs $0.03 for 10,000 transactions. The Premium version allows for a Hardware Security Module (HSM).
- To encrypt columns in SSMS:
  - Go to Databases – NameOfDatabase – Tables – NameOfTable.
  - Right-hand click and go to "Encrypt Columns…".
  - Select the columns and choose "Encryption Table", either Deterministic or Randomized.
    - Deterministic requires the string to be in a _Bin2 collation (e.g. Latin1_General_BIN2).
    - Deterministic allows equality joins, GROUP BY, indexes and DISTINCT. Randomized prevents this.
  - In "Master Key Configuration", you can go to "Select an Azure Key Vault" and select the Key Vault.
  - The next three stages are Validation, Summary and Results.
- When the columns are encrypted, then when connecting, go to the "Additional Connection Parameters" tab, and enter:
  - Column Encryption Setting=enabled
- Database Permissions are:
  - ALTER ANY COLUMN MASTER KEY
    - Needed to create/delete a column master key.
  - ALTER ANY COLUMN ENCRYPTION KEY
    - Needed to create/delete a column encryption key.
  - VIEW ANY COLUMN MASTER/ENCRYPTION KEY DEFINITION
    - Needed to access/read the metadata of the column master/encryption keys to manage keys or query encrypted columns.
  - Use GRANT VIEW ANY COLUMN MASTER KEY DEFINITION TO NameOfUser
- Do you need role separation?
  - Security Administrator generates columns encryption keys and column master keys.
    - Needs access to the keys and the key store, but not the database.
  - Database Administrator (DBA) manages metadata about the keys in the database.
    - Does not need access to the keys or the key store.
  - Should they be different people?
    - If not, you can use either SSMS or PowerShell.

        - If so, then you can only use PowerShell.

- For the Security Administrator
    - # Create a column master key in Windows Certificate Store.
        - $storeLocation = "CurrentUser"
        - $certPath = "Cert:" + $storeLocation + "\My"
        - $cert = New-SelfSignedCertificate -Subject "AlwaysEncryptedCert" -CertStoreLocation $certPath -KeyExportPolicy Exportable -Type DocumentEncryptionCert -KeyUsage DataEncipherment -KeySpec KeyExchange
    - # Import the SqlServer module
        - Import-Module "SqlServer"
    - # Create a SqlColumnMasterKeySettings object for your column master key.
        - $cmkSettings = New-SqlCertificateStoreColumnMasterKeySettings -CertificateStoreLocation "CurrentUser" -Thumbprint $cert.Thumbprint
    - # Generate a column encryption key, encrypt it with the column master key to produce an encrypted value of the column encryption key.
        - $encryptedValue = New-SqlColumnEncryptionKeyEncryptedValue -TargetColumnMasterKeySettings $cmkSettings
    - # Share the location of the column master key and an encrypted value of the column encryption key with a DBA, via a CSV file on a share drive
        - $keyDataFile = "Z:\keydata.txt"
        - "KeyStoreProviderName, KeyPath, EncryptedValue" > $keyDataFile
        - $cmkSettings.KeyStoreProviderName + ", " + $cmkSettings.KeyPath + ", " + $encryptedValue >> $keyDataFile
    - # Read the key data back to verify
        - $keyData = Import-Csv $keyDataFile
        - $keyData.KeyStoreProviderName
        - $keyData.KeyPath
        - $keyData.EncryptedValue
- For the DBA
    - # Obtain the location of the column master key and the encrypted value of the column encryption key from your Security Administrator, via a CSV file on a share drive.
    - $keyDataFile = "Z:\keydata.txt"
    - $keyData = Import-Csv $keyDataFile
    - # Import the SqlServer module

- Import-Module "SqlServer"

- # Connect to your database.
- $serverName = "<server name>"
- $databaseName = "<database name>"
- $connStr = "Server = " + $serverName + "; Database = " + $databaseName + "; Integrated Security = True"
- $database = Get-SqlDatabase -ConnectionString $connStr

- # Create a SqlColumnMasterKeySettings object for your column master key.
- $cmkSettings = New-SqlColumnMasterKeySettings -KeyStoreProviderName $keyData.KeyStoreProviderName -KeyPath $keyData.KeyPath

- # Create column master key metadata in the database.
- $cmkName = "CMK1"
- New-SqlColumnMasterKey -Name $cmkName -InputObject $database -ColumnMasterKeySettings $cmkSettings

- # Generate a column encryption key, encrypt it with the column master key and create column encryption key metadata in the database.
- $cekName = "CEK1"
- New-SqlColumnEncryptionKey -Name $cekName -InputObject $database -ColumnMasterKey $cmkName -EncryptedValue $keyData.EncryptedValue

	Always Encrypted	Transparent Data Encryption (TDE)
SQL Server version	SQL Server 2016 and above; Azure SQL Database	SQL Server 2008 and above; Azure SQL Database
Requires SQL Server Enterprise Edition or Azure SQL Database	x (starting with SQL Server 2016 SP1)	Yes
Free in Azure SQL Database	Yes	Yes
Protects data at rest	Yes	Yes
Protects data in use	Yes	x
Protects data from SQL administrators and admins	Yes	x
Data is encrypted/decrypted on the client side	Yes	x
Data is encrypted/decrypted on the server side	x	Yes
Encrypt at column level	Yes	x (encrypts entire database)
Transparent to application	Partially	Yes
Encryption options	Yes	x
Encryption key management	Customer Managed Keys	Service or Customer Managed Keys
Protects keys in use	Yes	x
Driver required	Yes	x

34. apply a data classification strategy

- Sensitive data includes:
  - Data privacy, regulatory and national requirements (e.g. GDPR),
  - Security scenarios, including controlling access.
- To apply a data classification strategy:
  - Go to the database.
  - Under Security, go to “Data Discovery & Classification”.
  - Go to the Classification tab.
  - At the bottom of the screen, you may have “X columns with classification recommendations”.
    - Check whatever recommendations you want to accept/dismiss.
    - Click “Accept” or “Dismiss selected recommendations”.
  - To create a new classification:
    - Click on “+ Add classification”.
    - Select the Schema, Table and Column name.
    - Select the Information type:
      - [n/a], Other
      - Networking
      - Personal data: Contact Info, Name, National ID, SSN, Health, Date of Birth,

            - Credentials
            - Financial records: Credit Card, Banking, Financial
        - Select the Sensitivity Label:
            - [n/a] – Data from your own personal life.
            - Public – Freely available business data, or information that has been released to the public.
            - General – Business data not meant for the public, such as emails, documents and files which do not include confidential data.
            - Confidential or Confidential – GDPR, Highly Confidential or Highly Confidential – GDPR – Business data that would cause harm or extensive harm to your company if overshared.
        - You cannot select [n/a] for both Information Type or Sensitivity Label.
        - Then click “Add classification”.

- The following roles can modify and read a database’s data classification:
    - Owner,
    - Contributor,
    - SQL Security Manager.
- Additionally, the following roles can read (but not modify) a database’s data classification:
    - Reader, and
    - User Access Administrator.
- You can use Audit to drill down into "Security Insights", "Access to Sensitive Data" etc.
    - You can also see it in Intelligent Insights.
- You can also use T-SQL, REST API or PowerShell to manage classifications.
- In T-SQL:
    - To add a sensitivity classification:
    - ADD SENSITIVITY CLASSIFICATION TO
    - [schema.]table.column1[, schema.table.column2]… etc.
    - WITH (
    - LABEL='Highly Confidential’, -- you could also use LABEL_ID
    - INFORMATION_TYPE='Financial’, -- you could also INFORMATION_TYPE_ID
        - Networking, Contact Info, Credentials, Credit Card, Banking, Other, Name, National IS, SSN, Health, Date of Birth
    - RANK=NONE, LOW, MEDIUM, HIGH or CRITICAL
    -)

- To check sensitivity classifications:
  - SELECT * FROM sys.sensitivity_classifications
- To drop a sensitivity classification:
  - DROP SENSITIVITY CLASSIFICATION FROM [schema.]table.column1[, schema.table.column2]…

35. configure server and database audits

- You can use auditing to:
  - Retain a trail of selected database actions,
  - Report on database activities, using pre-configured reports and a dashboard.
  - Analyse reports for suspicious events, unusual activity and trends.
- Notes:
  - It is not supported for premium storage or hierarchical namespace.
  - Under high activity, Azure will prioritise other actions and may not record some audited events.
  - They are written to Append Blobs in Blob storage.
- Server policy audits:
  - For all existing and newly created databases.
  - Server policy audits always applies to the database, regardless of any database-level auditing policies. They can sit side-by-side.
  - Microsoft recommends using only server-level auditing, unless you want to audit different event types/categories for a specific database.
  - The default auditing policy includes:
    - BATCH_COMPLETED_GROUP
      - All queries and stored procedures,
    - SUCCESSFUL_DATABASE_ and FAILED_DATABASE _AUTHENTICATION_GROUP
      - Success and failed logins
  - It stores 4,000 characters of data in an audit .
- To do this:
  - Go to the Azure portal – NameOfServer or NameOfDatabase – Security – Auditing.
  - Click “Enable Azure SQL Auditing” to track these events for a particular database or server. You can select the details to be stored in:
    - An existing or new Storage account
      - The Advanced settings allow you to choose the retention period (the default, zero days, is unlimited),

      - This Advanced setting only applies to new audits.
    - An existing Monitor Log Analytics workspace, and/or
    - An existing Event Hub.
  - If you are in the database, you can click on “View server settings”.
  - If you are in the server, you can also audit Microsoft support operations.
- If you audit to an existing Monitor Log Analytics workspace:
  - You can add it by:
    - Creating an Azure Storage Container, going to Overview – Blobs, and “+Container”.
    - Give the container a name, set the Public access level to Private and click OK.
    - In the Properties, click on Properties and copy the URL for future use.
    - Go to the Storage Account, and click on “Storage Settings – Shared access signature”.
    - Add “Blob” to “Allowed services”, choose the Start date as yesterday (to avoid timezone related problems), and an End date.
    - Click “Generate SAS” and copy this token for future use. (are the highlighted needed?)
  - Choose the “Log Analytics” in the Auditing.
  - You can “View audit logs”.
  - You can then either:
    - Click on “Log Analytics” to go to the workspace, or
    - Click “View dashboard” to see a dashboard of audit logs.
- If you audit logs to the Event Hub, then:
  - You would need to set up a stream to consume these events and write them to a target.
- If you audit logs to an Azure storage account, then:
  - You can explore them in Azure Storage Explorer.
  - You can click on “View audit logs”:
    - Filter on specific dates,
    - Look at Server or Database audit policy.
  - You can use T-SQL:
    - SELECT * FROM sys.fn_get_audit_file (‘NameOfFile.sqlaudit',default,default);
  - You can use SSMS, going to File – Open – Merge Audit Files.
  - You should change your storage keys periodically.
    - In advanced properties, you can change to the secondary access storage key.

        - Then you can go to your Storage Account – Settings – Access keys, and click the regenerate icon on the primary access key.
        - You can then go back to the audit, and change it to the primary key.
        - You can then go to your Storage Account – Settings – Access keys, and click the regenerate icon on the secondary access key.

36. implement data change tracking

- Change Tracking is supported in Azure SQL Database only.
    - Change Tracking tracks whether the column was changed.
    - However, it does not track how many times nor does it track historic data. Therefore, it more lightweight and requires less storage than CDC (Change Data Capture).
    - It therefore enables applications to determines which rows have changed, and request those rows. (But you cannot see the previous data.)
    - The data is stored in an in-memory rowstore, and flushed on every checkpoint to the internal data.
    - You may wish to consider using snapshot isolation for the database, so that changes made while getting the data are not visible within the transaction:
        - ALTER DATABASE AdventureWorksLT
        - SET ALLOW_SNAPSHOT_ISOLATION ON;
        - SET TRANSACTION ISOLATION LEVEL SNAPSHOT;
        - BEGIN TRAN / COMMIT TRAN
- To enable Data Change Tracking on a database:
    - In SSMS
        - Right-hand click on the database, select Properties and go to Change Tracking.
        - Change “Change Tracking” to True.
        - Select the Retention Period and Units (by default, 2 Days) – the minimum is 1 Minute; there is no maximum.
        - Select whether data is “Auto Cleanup” in that retention period.
            - If true, Change Tracking data will be removed periodically. If an App has not got updated information in time, all data will needed to be refreshed.
            - If False, change tracking information will not be removed and will continue to grow.
    - In T-SQL
        - ALTER DATABASE MyDatabase
        - SET CHANGE_TRACKING = ON
        - (CHANGE_RETENTION = 2 DAYS, AUTO_CLEANUP = ON)

- However, you still need to enable it in a particular table.
- To enable Data Change Tracking on a table:
  - In SSMS
    - Right-hand click on the database, select Properties and go to Change Tracking.
    - Change "Change Tracking" to True.
    - If True, you can also change "Track Columns Updated" to True. This will indicate whether UPDATEs to individual columns will be tracked.
  - In T-SQL
    - ALTER TABLE Schema.Table
    - ENABLE CHANGE_TRACKING
    - WITH (TRACK_COLUMNS_UPDATED = ON)
- You can also disable Change Tracking on tables and databases
  - However, to disable it on the database, all track changing of tables needs to be disabled first.
  - In SSMS, change True to False.
  - In T-SQL for tables:
    - ALTER TABLE Schema.Table
    - DISABLE CHANGE_TRACKING
  - In T-SQL For databases:
    - ALTER DATABASE MyDatabase
    - SET CHANGE_TRACKING = OFF
- To check which tables/databases have Change Tracking enabled:
  - SELECT * from sys.change_tracking_databases
  - SELECT * from sys.change_tracking_tables -- this uses the current database. You need:
    - SELECT permission on the primary key columns for the tables
    - VIEW CHANGE TRACKING permission for the relevant table.
- To use it:
  - To get the initial sync version
    - declare @last_sync bigint;
    - SET @last_sync = CHANGE_TRACKING_CURRENT_VERSION();
  - After changes have happened:
    - SELECT CT.ProductID, CT.SYS_CHANGE_OPERATION,
    - CT.SYS_CHANGE_COLUMNS, CT.SYS_CHANGE_CONTEXT
    - FROM CHANGETABLE(CHANGES *Schema.Table*, @last_sync) AS CT

  - Check that you don’t have to refresh the entire table:
    - IF (@last_sync < CHANGE_TRACKING_MIN_VALID_VERSION(OBJECT_ID('*Schema.Table*'))
- Change Data Capture (CDC) is supported in Azure SQL Database, Azure SQL Managed Instance and SQL Server on VM.
  - CDC tracks historic data.
  - Needs a minimum of 1 vCore or 100 DTUs or eDTUs.
  - Cannot be used in Azure SQL Database Free, Basic or Standard tier Single Database (S0, S1, S2).
  - Cannot be used in Azure SQL Database Elastic Pool with vCore < 1 or eDTUs < 100.
- Before you can enable it for a table, you must switch it on for the database.
  - EXEC sys.sp_cdc_enable_db
    - Returns 0 for Success, 1 for Failure.
    - Only use it on user databases, not system databases.
    - It creates the Change Data Capture objects, including metadata tables and DDL triggers.
    - You need sysadmin to run it.
- Then you can enable it for a table:
  - EXEC sys.sp_cdc_enable_table . Some of the arguments are:
    - @source_schema = N'HumanResources'
    - , @source_name = N'Department'
      - The name of the table.
    - , @role_name = N'cdc_admin'
      - The database role used to gate access to change data. Could be a new role.
    - , @captured_column_list = N'DepartmentID, Name, GroupName'
      - Columns to be captured. Needs the primary key.
      - Cannot use encrypted columns.
    - Returns 0 for Success, 1 for Failure.
- You can query your configuration using:
  - EXECUTE sys.sp_cdc_help_change_data_capture
- You can view changed rows by using:
  - DECLARE @from_lsn binary(10), @to_lsn binary(10);
  - SET @from_lsn = sys.fn_cdc_get_min_lsn('HR_Department');
  - SET @to_lsn = sys.fn_cdc_get_max_lsn();

  - SELECT * FROM cdc.fn_cdc_get_all_changes_HR_Department (@from_lsn, @to_lsn, N'all');
    - _$operation = 1 (delete), 2 (insert), 3 or 4 (update)

37. perform a vulnerability assessment

- Azure Defender for SQL costs around $15/server/month.
- In the Azure portal, go to the SQL database.
- Go to Security – Security Center.
- Next to "Enabled at the subscription level", click "Configure".
  - Select your storage account.
  - Select "Periodic recurring scans" to On if you want weekly scans.
  - Enter an email address for your scan reports and alerts.
    - You can also send emails to admins and subscription owners.
- To view details of the findings, go to Security – Security Center - "View additional findings in Vulnerability Assessment".
  - Findings include an overview, number of issues found, severity risk summary, and findings list.
    - You can click on an issue for more details.
    - You can "Approve as Baseline" specific results. Any similar results are put in the "Passed" section.
    - You can also "Disable rule"
  - You can also click "Scan" to do an on-demand scan.
  - Click "Scan History" to view previous scans.
- Click on "Export Scan Results" to download an Excel report.

Monitor and Optimize Operational Resources

38. prepare an operational performance baseline

- Metrics are numerical values that are collected at regular intervals and have a timestamp, name, value and other labels.
  - They are stored in a time-series database which is suitable for alerting and fast detection of issues.
  - It is lightweight and allow for near-real time alerting.
- View in Metrics Explorer.
  - Go to Azure Portal – database – Monitoring – Metrics.
  - Select:
    - Scope,

    - Metric Namespace,
    - Metric (e.g. "Data space used"), and
    - Aggregation (Min/Max/Avg or Sum/Count).
  - To change the date/time range, go to the top-right hand corner (where it says "Local time").
    - You can also change the "Show time as" from Local to UTC/GMT, and change the "Time granularity" (how often it does the aggregation).
    - Only a maximum of 30 days visible at once, but you can use the arrow at the left-right to go back up to 93 days in the past.
  - You can:
    - Change the color of a line (by clicking on the color in the legend – not the line, but the legend).
    - Edit the title,
    - Split or filter a metric, if it has a dimension (not applicable to Azure SQL Database).
    - Add a second metric onto the same chart (e.g. "Date space allocated").
    - Change the chart type (from Line to Area, Bar, Scatter and Grid).
    - Move the chart up, down, clone it, delete it, or see more settings (in the … to the right-hand side).
    - Add a new chart
    - Share it by Downloading it to Excel or copy to Link.
- Logs are events in the system, which may contain other (non-numerical) data and may be structured or free-form, with a timestamp.
  - View in Log Analytics.
- Areas which could affect SQL Server Performance include:
  - Hardware/compute/memory,
  - The Operating System (VM),
  - Database applications and
  - Client applications.
- Azure Monitor allows you to monitor resource metrics, such as processor, memory and I/O resources.
  - You may need more CPU or I/O resources if you have high DTU/processor percentage or high I/O percentage. Alternatively, your queries may need to be optimized.
- You can also use T-SQL:
  - SELECT * from sys.dm_db_resource_stats -- CPU, IO and memory
    - You get a row for every 15 seconds for about the past hour.
  - SELECT * FROM sys.dm_user_db_resource_governance -- storage

- SELECT * FROM sys.resource_usage
  - You get a row showing the hourly summary of resource usage data for user databases. Historical data is retained for 90 days.
  - However, this is currently in a preview state. It says "Do not take a dependency on the specific implementation of this feature because the feature might be changed or removed in a future release."

39. determine sources for performance metrics

- The sources for performance metrics include:
  - Azure Tenant – Tenant-wide services such as Azure Active Directory.
  - Subscription
    - Azure Activity log includes service health records and records of configuration changes.
    - Azure Service Health has information about your Azure services' health
  - Resources
    - Most Azure services submit Platform metrics to the metrics database
    - Resource logs are created internally regarding the internal operation of an Azure resource.
  - Guest operating system in Azure, other clouds, and on-prem
    - Azure Diagnostic extension for Azure VM, when enabled, submits logs and metrics
    - Log Analytics agents can be installed into your Windows or Linux VMs, running in Azure, another cloud, or on-prem
    - VM insights (preview) provides additional Azure Monitor functionality on Windows and Linux VMs
  - Other sources
    - In Application code, you can enable Application Insights to collect metrics and logs relating to the performance and operations of the app.
    - Monitoring Solutions and Insights provide additional insights of a particular service or app.
    - Container insights provide data about Azure Kubernetes Service (AKS).
    - VM Insights allow for a customized monitoring of VMs.
    - In VM, you can also look at Windows Performance Monitor (perfmon).
      - There are specific counters for SQL Server.

39. determine sources for performance metrics

- Metrics available for MI are:
  - Average CPU percentage in a selected time period,

    - I/O bytes read/written,
    - I/O requests counts,
    - Storage space reserved/used,
    - Virtual core count (4-80 vCores).

- Metrics available for Azure SQL Database are:
    - Blocked by firewall,
    - Deadlocks,
    - CPU %,
    - Data I/O % or Log I/O %,
    - Data space used/allocated/used %,
    - DTU Limit, Used, %
    - Failed connections, Successful connections,
    - In-memory OLTP storage %,
    - Sessions %,
    - Workers %,
    - SQL Server process core/memory %,
    - Tempdb Data/Log File Size Kilobytes,
    - Tempdb % Log Used.

40. interpret performance metrics

- If any of the following is high (close to 100%), consider upgrading to the next service tier.
- Space/components used
    - DTU percentage – CPU, memory and I/O for vCores (not DTU-based model)
        - If this is low, then you may save money downgrading.
    - Other Component metrics:
            - CPU percentage (avg_cpu_percent),
                - When high, query latency increases and queries may time out.
                - Maybe increase the compute size, or optimise queries to reduce the CPU requirements.
            - Data IO percentage (avg_data_io_percent),
            - Log IO percentage (avg_log_write_percent),
    - In-memory OLTP storage percent (xtp_storage_percent)
        - Returns zero if in-memory OLTP (memory-optimized tables, indexes, and table variables) is not used.

        - If this hits 100%, then INSERT, UPDATE, ALTER and CREATE operations will fail (SELECT and DELETE are fine).
    - Data space used percent If this is getting high, then upgrade to the next service tier, shrink the database, or scale out using sharding.
        - If in an elastic pool, consider moving it out of the pool.
    - Memory percentage (avg_memory_usage_percent)
        - This is used for caching. If you get out-of-memory errors, Increase service tier, or compute size, or optimize queries.
- Connections/requested used
    - Sessions percentage
        - Maximum concurrent sessions divided by the service tier limit. (max_session_percent)
    - Worker percentage
        - Maximum concurrent requests divided by the service tier limit. (max_worker_percent).
    - Increase service tier, or compute size, or optimize queries.
- Top queries can be shown in Intelligent Performance – Query Performance Insight.
    - Review top CPU-consuming queries
    - Individual query details
    - Top queries per duration or execution count (Custom – Metric type: Duration or Execution Count)
    - You may see icons showing performance recommendations.

41. assess database performance by using Intelligent Insights for Azure SQL Database and Managed Instance

- Not available in some regions
    - West Europe, North Europe, West US 1 and East US 1.
    - Not available for VMs.
- Compares current database workload (last hour) with the last 7-days.
    - e.g. Most repeated and largest queries.
    - Uses data from the Query Store (see topic 48), which is enabled by default in Azure SQL Database.
- Monitors using Artificial Intelligence operational thresholds, detects issues with high wait times, critical exceptions, and query parameterizations
    - Impacted metrics are increase to query duration, excessive waiting, timed-out or errored-out requests.

- Includes a “root cause analysis” in a readable form. May also contain a recommendation.

- Can be streamed to:
  - Log Analytics workspace, can be used with Azure SQL Analytics (cloud-based only monitoring solution) to see insights in the Azure portal. The typical way to view insights.
    - To add Azure SQL Analytics, go to Home in the Azure portal, click “+Create a resource”, and search for “Azure SQL analytics”
    - Can query using the Kusto Query Language.
  - Azure Event Hubs, for custom monitoring and alerting
    - Stored in Avro format, a binary JSON format
  - Azure Storage, for custom app development.
    - Stored in Extended Events format.
- How to connect
  - Connect the Intelligent Insights to the log. OR
  - Go to the database in the Azure Portal, and go to Monitoring – Diagnostic settings – Add.
    - Add all the Category Details (log and metric), and in “Destination details” check “Send to Log Analytics workspace”.
- To view Intelligent Insights:
  - Go to Azure Portal – database – Query Performance Insight
- To view more Intelligent Insights:
  - Go to the Log Analytics workspace – Workspace summary
- Intelligent Insights looks for things which affect Database performance:
  - Reaching resource limits
    - CPU reaching resource limits for Managed Instance, and
    - DTUs, worker threads and login sessions reaching resource limits for Azure SQL Database.
  - Workload increase
  - Memory pressure
    - Workers (request) waiting for memory allocations
  - Data locking
  - Increase Maximum Degree of Parallelism option (MAXDOP)
    - When there are more Parallel workers than there should have been.
  - Missing indexes
  - New queries affected performance.
  - Increased Wait Statistic.

- Multiple threads using the same TempDB resource.
- For DTU-model, shortage of available eDTUs in the elastic pool.
- New plan, or change in existing plan.
- Pricing tier downgrade.

42. configure and monitor activity and performance at the infrastructure, server, service, and database levels

- See topic 38.
- You can go to a database, can click on:
  - Metrics,
  - Performance Overview,
  - Performance recommendations, or
  - Query Performance Insight.
- You can also monitor using Dynamic Management Views:
  - https://docs.microsoft.com/en-us/azure/azure-sql/database/monitoring-with-dmvs

Event or activity	Extended Events	SQL Server Profiler	Distributed Replay	System Monitor	Activity Monitor	Transact-SQL	Error logs	Performance Dashboard
Trend analysis	Yes	Yes		Yes				
Replaying captured events		Yes (From a single computer)	Yes (From multiple computers)					
Ad hoc monitoring	Yes (XEvent Profiler)	Yes			Yes	Yes	Yes	Yes
Generating alerts				Yes				
Graphical interface	Yes	Yes		Yes	Yes		Yes	Yes
Using within custom application	Yes	Yes (Profiler Stored Procedures)				Yes		

43. implement index maintenance tasks

- Missing indexes can be found in:
  - SELECT * FROM sys.dm_db_missing_index_details
- In Azure SQL Database, in the Azure Portal – you can go to Intelligent Performance – Automatic tuning.
  - You can click on a “Create index” or “Drop index” and implement it.
  - You cannot do it in Azure SQL MI.
- For VMs, you can also use the Database Engine Tuning Advisor.
  - You open it by going to Tools – Database Engine Tuning Advisor.

  - You then need to give it details such as the Query Store or a T-SQL file (.sql extension) with your workload.
  - You can then click "Start Analysis".

44. implement statistics maintenance tasks

- Statistics are used to create query plans to improve the speed of queries.
  - The statistics contain information about the distribution of values in tables or indexed views' columns.
  - It uses it to estimate the cardinality, or number of rows, in a result.
  - This enables the Query Optimizer to create better quality plans (e.g. seek vs scan).
- Usually, the Query Optimizer determines when statistics might be out of date and then updates them. However, you may wish to manually update them if:
  - Query execution times are slow,
  - Insert operations occur on ascending or descending key columns, such as IDENTITY or timestamp columns.
  - After maintenance operations, such as a bulk insert (but not rebuilding or reorganizing an index, as they do not change the data distribution).
- The stored procedure sp_updatestats updates statistics for all user-defined and internal tables.
- To update a particular table or indexed view, you can use:
  - UPDATE STATISTICS Schema.Table
  - You can add:
    - WITH FULLSCAN – This scans all of rows. It is the same as SAMPLE 100 PERCENT
    - WITH SAMPLE X PERCENT or ROWS. This is the approximately percentage of number of rows to be used for updating statistics.
    - WITH RESAMPLE – its most recent sample rate.
  - You can append this with
    - PERSIST_SAMPLE_PERCENT = ON or OFF – If ON, this will then be the default for future statistics updates (unless you specify the number of rows).

45. configure database auto-tuning

- Auto-tuning is a process which learns about your workload and identifies potential issues and improvement: Learn – Adapt – Verify – repeat.
- You can configure database auto-tuning by:
  - In the Azure Portal, go to the database or server.
  - Go to "Automatic tuning".
- In Azure SQL Database and Azure SQL MI:
  - You can configure FORCE_LAST_GOOD_PLAN (it is <u>enabled</u> by default)

    - This says that the last good plan should be forced whenever some plan change regression is found – when the estimated gain is >10 seconds, or the number of errors in the new plan is > recommend plan.

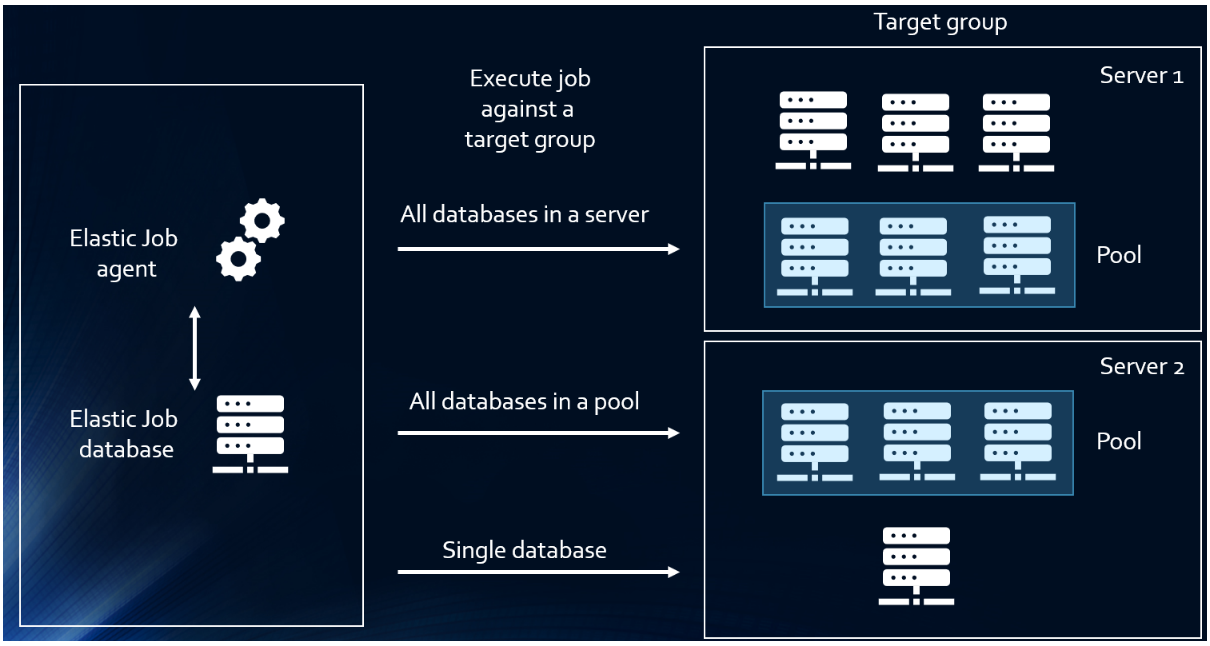
- In Azure SQL Database only, you can automate index maintenance by:
    - You can change “Create Index” and “Drop Index” from Inherit (from server) to OFF or ON [the default for servers is OFF for both of these]. These will override the server settings.
    - Indexes will only be auto-created if the CPU, data I/O and log I/O are lower than 80%.
    - You can view which indexes are auto-created by going to:
        - SELECT * FROM sys.indexes WHERE auto_created = 1
    - The performance of queries using the auto-created index will be reviewed. If is doesn’t improve performance, it is automatically dropped.
    - You can do this in T-SQL for a single database as follows:
        - ALTER DATABASE DatabaseName SET AUTOMATIC_TUNING = AUTO | INHERIT | CUSTOM
        - ALTER DATABASE DatabaseName SET AUTOMATIC_TUNING (FORCE_LAST_GOOD_PLAN = ON, CREATE_INDEX = ON, DROP_INDEX = OFF)
            - CREATE_INDEX and DROP_INDEX cannot be done in Azure SQL MI.
    - Recommendations, if any, can be found in:
        - SELECT * FROM sys.dm_db_tuning_recommendations

46. automate database maintenance tasks

- A lot of database maintenance tasks are already automated in Azure, such as updates, backups and creation of indexes.
- You can create elastic job agents to automate maintenance tasks and/or run T-SQL queries.
    - You could: manage credentials, collect performance data or telemetry data.
    - Update reference data or load or summarise data from databases or Azure Blob storage.
    - Targets can be in different servers, subscriptions or regions, but must be in the same Azure cloud.
        - One or more databases, all databases in a server or elastic pool or shard map.
    - This is the equivalent of SQL Agent Jobs, which are available in SQL MI, but are not available in Azure SQL Database.
- You need:
    - Elastic Job agent – the Azure resource which runs the jobs. This is free.
    - Job database – an existing Azure SQL Database stores job related data, such as metadata, logs, results and job definitions. It also contains stored procedures and other objects for jobs.
        - This is charged for as an Azure SQL Database.

        - You need a Standard (S0 or above) or Premium service tier. S1 or above is recommended, but if you run frequent jobs or against a big target group, you may need more.
    - Target group – servers, elastic pools, databases and databases of shard map(s) which are affected.
        - If a server or elastic group, all databases in the server at the time of running the job will be affected. You will need to give the master database credential, so the databases can be enumerated. You can also exclude individual databases or all databases in an elastic pool.
    - Job – unit of work which contained job steps, each of which specify the T-SQL script and other details.
        - Scripts must be "idempotent", capable of running twice with the same result.
    - Job output – this can be saved in a table.
    - Job history – stored for 45 days in jobs.job_executions
- Create the job database.
    - An empty S0 or higher database.
    - Create a credential for running the jobs in the Job database.
        - CREATE MASTER KEY ENCRYPTION BY PASSWORD='<an6?%9++Vyd%Ut9';
        - CREATE DATABASE SCOPED CREDENTIAL MasterCred WITH IDENTITY = 'MasterU' SECRET = '<an6?%9++Vyd%Ut9'
        - CREATE DATABASE SCOPED CREDENTIAL RunJob WITH IDENTITY = 'JobU' SECRET = '<an6?%9++Vyd%Ut9'
- Create an Elastic Job agent in the Portal.
    - Go to the Azure Portal
    - Search for Elastic Job agents
    - Click Add, enter the name, and click OK.
    - Select the Azure SQL job database.
- Define the target group in T-SQL (or PowerShell).
    - In the job database:
    - EXEC jobs.sp_add_target_group 'GrpDatabase';
    - EXEC jobs.sp_add_target_group_member
        - @target_group_name = 'GrpDatabase',
        - @target_type = 'SqlDatabase'
            - or 'SqlServer', -- or 'PoolGroup'
                - if wanting to exclude, @membership_type = 'Exclude'

            - If targeting a server or pool, @refresh_credential_name = 'RefreshPassword',
        - @server_name = 'DataBaseName.database.windows.net';
    - To view the recently created target group and target group members
        - SELECT * FROM jobs.target_groups WHERE target_group_name='GrpDatabase';
        - SELECT * FROM jobs.target_group_members WHERE target_group_name='GrpDatabase';
- In each database, you will need a job agent credential in each affected database. You could use PowerShell for this.
    - In the Master Database:
        - CREATE LOGIN MasterU WITH PASSWORD ='<an6?%9++Vyd%Ut9'
        - CREATE USER MasterU FROM LOGIN MasterU
        - CREATE LOGIN JobU WITH PASSWORD = '<an6?%9++Vyd%Ut9'
    - In the target user database:
        - CREATE USER JobU FROM LOGIN JobU
        - ALTER ROLE db_owner ADD MEMBER JobU
- Create a job in T-SQL (or PowerShell) in the Elastic Job database
    - EXEC jobs.sp_add_job @job_name='My first job', @description='Look at objects'
- Create job steps in T-SQL (or PowerShell).
    - EXEC jobs.sp_add_jobstep @job_name='My first job',
- @command='SELECT * FROM sys.objects',
- @credential_name='RunJob',
- @target_group_name='GrpDatabase'
- Run/schedule the job in T-SQL.
    - EXEC jobs.sp_start_job 'My first job' -- run now
    - EXEC jobs.sp_update_job
        - @job_name='Sample T-SQL',
        - @enabled=1,
        - @schedule_interval_type='Minutes' – Or Hours, Days, Weeks, Months or Once,
        - @schedule_interval_count=1
- Monitor job execution in the Portal or T-SQL (or PowerShell).
    - select * from jobs.job_executions



47. manage storage capacity

- This applies to Azure SQL Database, not Azure SQL Managed Instance.
- You may need:
  - Add more space or decrease the maximum capacity to a database elastic pool.
  - Change to a different service tier.
- Terminology:
  - Data space used
    - Generally increases with inserts and decreases with deletes, but dependent on fragmentation.
  - Data space allocated
    - Can grow automatically, but does not automatically decrease after deletes.
  - Data space allocated but unused
    - Can be reclaimed when data files are shrinked.
  - Data maximum size.
    - The maximum that "Data space allocated" can be.
- This applies to Azure SQL Database, not Azure SQL Managed Instance.
  - Display the allocated space:
    - [Single database – In the master database]
      - SELECT database_name, allocated_storage_in_megabytes FROM sys.resource_stats
    - [Elastic pool – In the master database]

        - SELECT elastic_pool_name, elastic_pool_storage_limit_mb, avg_allocated_storage_percent FROM master.sys.elastic_pool_resource_stats

    - Display maximum size:
        - [Go to the relevant database. If using Master, results be NULL]
        - SELECT DATABASEPROPERTYEX('DatabaseName', 'MaxSizeInBytes')
    - To shrink a transaction log file
        - SELECT file_id, size FROM sys.database_files WHERE type = 1 -- "1" = Log file. Size is in 8 Kb pages.
        - DBCC SHRINKFILE (2); --Where "2" is the file_id.
        - Will impact database performance when running; should be done when less used.
        - DBCC SHRINKDATABASE(MyDatabase) will shrink all the data and log files in the MyDatabase database. (Note the lack of quote marks.)

48. configure Query Store to collect performance data

- What is Query Store?
    - It contains 3 stores:
        - Plan store for execution plan data.
        - Runtime stats store for execution statistics data.
        - Wait stats store.
- When would you use Query Store?
    - Fix queries which are regressed due to changes in the execution plan.
    - How many times has a query been executed?
    - What are the Top X queries, by execution time, memory consumption, waiting on resources?
    - Look at query plans for a given query.
    - Look at CPU, I/O and memory used for a particular database.
    - What are the waits for a query?
- Query Store is:
    - Disabled by default for new SQL Server databases (e.g. on a VM), but
    - Enabled by default for new Azure SQL Databases.
- To enable Query Store:
    - In SSMS:
        - Right-hand click on a database, and go to Properties.
        - Go to the Query Store tab.

        - To enable Query Store generally, change "Operation Mode (Requested)" to "Read write".
        - To enable wait stats, change "Wait Statistics Capture Mode" to "On".
    - In T-SQL:
        - ALTER DATABASE Database_Name SET QUERY_STORE = ON (OPERATION_MODE = READ_WRITE);
        - ALTER DATABASE Database_Name SET QUERY_STORE = ON (WAIT_STATS_CAPTURE_MODE = ON);
- It may up to a day to collect sufficient data to represent your workload.
- Options:
    - Is it collecting runtime stats?
        - Use SELECT actual_state, actual_state_desc, readonly_reason FROM sys.database_query_store_options;
        - If actual_state = 2, then it is READ_WRITE. If actual_state = 1, then it is READ_ONLY.
    - To change how often it collects stats: ("Statistics Collection Interval"):
        - In T-SQL:
            - ALTER DATABASE DatabaseName
            - SET QUERY_STORE (INTERVAL_LENGTH_MINUTES = 15);
        - You can choose from 1, 5, 10, 15, 30, 60 or 1440 minutes. A query will have a maximum of 1 row collected for this time period.
    - You can change multiple options in T-SQL:
    - ALTER DATABASE DatabaseName SET QUERY_STORE (
    - MAX_STORAGE_SIZE_MB = 500,
                - The space allocated to the Query Store. The default is 100 Mb in SQL Server 2016/2017, and 1 Gb in SQL Server2019.
                - If it reaches the limit, Query Store no longer collects new data and changes to read-only mode. This will reduce the performance accurate, because the Query Store will become stale.
            - SELECT actual_state, actual_state_desc, readonly_reason FROM sys.database_query_store_options;
            - readonly_reason would = 65536 if Query Store reached the MAX_STORAGE_SIZE_MB.
                - To prevent it from reaching the limit, increase the MAX_STORAGE_SIZE_MB. If you can't allocate extra space, then decrease the Data Flush time.

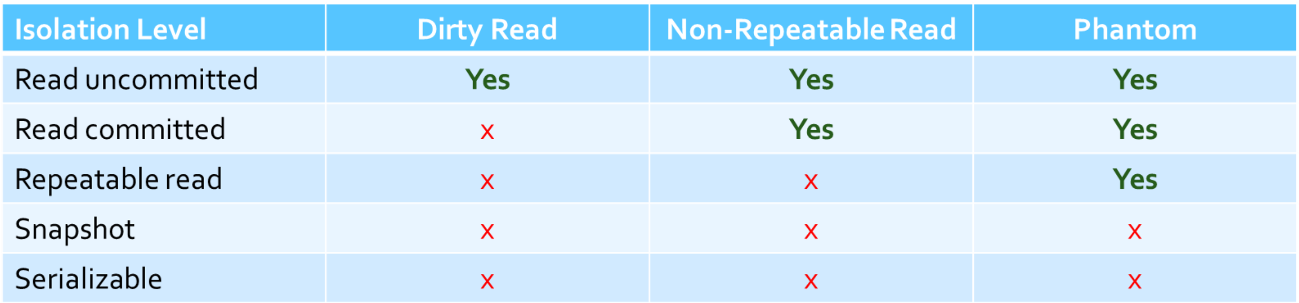
    - DATA_FLUSH_INTERVAL_SECONDS = 3000,
        - how long (in seconds) the data is retained in memory before being saved to disk.
        - Have a higher value if you don't have a large number of queries running being generated. However, if the SQL Server crashes or restarts, then anything new will not be saved.
        - Having a lower value may have a negative impact of performance, as it will save more often.
    - SIZE_BASED_CLEANUP_MODE = AUTO,
        - whether automatic data cleanup occurs when size limit is reached.
        - When Query Store data reaches about 90% of MAX_STORAGE_SIZE_MB, a clean-up begins. It will remove oldest/least expensive query data, and stops when size is about 80% of MAX_STORAGE_SIZE_MB.
        - If you need it checking more quickly, reduce the DATA_FLUSH_INTERVAL_SECONDS period.
    - OPERATION_MODE = READ_WRITE,
    - CLEANUP_POLICY = (STALE_QUERY_THRESHOLD_DAYS = 30),
        - how long data is retained in days.
        - You can automatically delete Query data that you don't need.
    - INTERVAL_LENGTH_MINUTES = 15,
        - in this number of minutes, each query has a maximum of 1 row. Statistics are aggregated for each query during this time.
    - QUERY_CAPTURE_MODE = AUTO,
        - capture "All" queries, None, Custom or Auto (ignore infrequent queries and queries with small compile/execution times). Default was "All", but in SQL Server 2019 and Azure SQL is "Auto".
    - MAX_PLANS_PER_QUERY = 1000,
    - WAIT_STATS_CAPTURE_MODE = ON);
- To clear:
    - ALTER DATABASE DatabaseName SET QUERY_STORE CLEAR;
    - or click the "Purge Query Data" button in SSMS.

49. identify sessions that cause blocking

- Blocking can occur when:

  - Session 1 locks a resource (e.g. row, page or entire table), and then
  - Session 2 requests that resource.
- Blocking is caused by:
  - Poor transactional design, or
  - Long running transactions.
- To emulate this, we are going to have an explicit transaction.
  - Implicit transactions automatically add a BEGIN and COMMIT TRANSACTION.
  - Explicit transactions require you to add the BEGIN, and COMMIT/ROLLBACK TRANSACTION.
- Session 1
  - BEGIN TRANSACTION
  - UPDATE [SalesLT].[Address]
  - SET City = 'Toronto ON'
  - where City = 'Toronto'
- Session 2
  - BEGIN TRANSACTION
  - UPDATE [SalesLT].[Address]
  - SET City = 'Toronto'
  - where City in ('Toronto ON', 'Toronto')
- To view locks:
  - SELECT * FROM sys.dm_tran_locks
- To view blocking:
  - SELECT session_id, blocking_session_id,
  - start_time, status, command,
  - DB_NAME(database_id) as [database],
  - wait_type, wait_resource, wait_time,
  - open_transaction_count
  - FROM sys.dm_exec_requests
  - WHERE blocking_session_id > 0
- For the session_id, look at the numbers in brackets at the top of SSMS.
- To reduce blocking, you can change the TRANSACTION ISOLATION LEVEL of a session:
  - SET TRANSACTION ISOLATION LEVEL …
  - READ UNCOMMITTED – No blocking, but would have dirty reads.

  - READ COMMITTED – No dirty reads, as would not read statements that have been modified but not committed.
    - If READ_COMMITTED_SNAPSHOT is OFF (the default on SQL Server), may block.
    - If READ_COMMITTED_SNAPSHOT is ON (the default on Azure SQL Database), uses a snapshot and therefore does not block.
  - REPEATABLE READ – No direct reads, but blocks.
  - SNAPSHOT – The data read remains the same until the end of the transaction. No blocks unless the database is being recovered.
    - Needs ALLOW_SNAPSHOT_ISOLATION to be ON.
  - SERIALIZABLE - No dirty reads, as would not read statements that have been modified but not committed. However, blocks updates/inserts.
- To see the current level, use
  - DBCC USEROPTIONS
- There are database options as well.
  - ALTER DATABASE NameOfDatabase SET ALLOW_SNAPSHOT_ISOLATION ON
    - DML statements start generating row versions – allows snapshots but doesn't enable it.
  - ALTER DATABASE NameOfDatabase SET READ_COMMITTED_SNAPSHOT ON
    - DML statements start generating row versions – means that TRANSACTION ISOLATION LEVEL READ COMMITTED does not block.



Isolation Level	Dirty Read	Non-Repeatable Read	Phantom
Read uncommitted	Yes	Yes	Yes
Read committed	x	Yes	Yes
Repeatable read	x	x	Yes
Snapshot	x	x	x
Serializable	x	x	x

- You can use Extended Events in MI and SQL Server on VM.
  - Lightweight tracing system, used for:
    - Troubleshooting blocking and deadlocking performance issues
    - Identifying long-running queries
    - Monitoring Data Definition Language (DDL) operations
    - Logging missing column statistics
    - Observing Memory Pressure in your database
    - Long-running physical I/O operations

- To create a new session:
    - In SSMS, go to Management – Extended Events – and right-hand click on Sessions and go to "New Session Wizard" or "New Session…"
    - Give the session a name.
    - Use a template if applicable, from:
        - Locks and Blocks,
            - Count Query Locks
        - Profiler Equivalents,
            - SP (Stored Procedure) Counts,
            - Standard (Stored Procedures and T-SQL batches),
            - TSQL (Debug client applications),
            - TSQL_SPs (analyze the component steps of SPs),
            - TSQL_Duration (identify slow queries),
            - TSQL_Locks (deadlocks),
            - TSQL_Replay (benchmark testing),
            - Tuning (Stored Procedures and T-SQL batches).
        - Query Execution,
            - Query Batch/Detail Sampling (20% of active sessions),
            - Query Batch/Detail Tracking (understand query flow),
            - Query Wait Statistic (query hash and query plan hash)
        - System Monitoring
            - Activity Tracking (general activity),
            - Connection Tracking (connection activity),
            - Database Log File IO Tracking (for database log files).
- Select Events to Capture
    - You can filter by Category.
- Capture Global Fields
    - Such as "session_id".
- Set Session Event Filters.
    - So you don't have to capture every event.
- Specify Session Data Storage.
    - Event Tracing for Windows (ETW)

        - Correlates SQL Server events with Windows OS events. Processes data synchronously
    - Event Counter
        - Counts how many times each event occurs. Processes data synchronously
    - Event File – disk file.
        - Creating large records (asynchronous)
    - Histogram
        - Counts how many times events occurs, for event fields and actions separately (asynchronous).
    - Pair Matching
        - Detect start events without an corresponding end event (asynchronous).
    - Ring Buffer
        - Smaller data sets or continuous data collection (asynchronous).

50. assess growth/fragmentation of databases and logs

- You can assess the growth in a database by using the following. It shows the growth over a time period:
    - For a database
        - SELECT database_name, start_time, storage_in_megabytes
        - FROM sys.resource_stats
        - ORDER BY database_name, start_time
        - Historical data is aggregated every 5 minutes and is retained for approximately 14 days.
    - For an elastic pool
        - SELECT start_time, elastic_pool_name, elastic_pool_storage_limit_mb, avg_allocated_storage_percent FROM master.sys.elastic_pool_resource_stats
        - ORDER BY start_time
    - To view the current log size:
        - SELECT file_id, type_desc, size, max_size, growth
        - FROM sys.database_files
        - WHERE type = 1
- Fragmented indexes can:
    - Degrade query performance, because
    - more I/O requests (with smaller data in each) are required.
    - Each page can be fragmented between 0% and 100%.

- To assess fragmentation of database indexes:
    - SELECT db_name(database_id) as DBName, object_name(object_id) as ObjectName, avg_fragmentation_in_percent, page_count, *
    - FROM sys.dm_db_index_physical_stats(NULL,NULL,NULL,NULL,NULL)
        - --The arguments are: database_id (use db_id to look it up), object_id (use object_id to look it up), index_id, partition_number and mode (the scan level).
    - order by avg_fragmentation_in_percent * page_count desc
    - You can also check it by right-hand clicking on the index in SSMS, going to Properties – Fragmentation.
- You can also use:
    - DBCC SHOWCONTIG
    - However, this has been deprecated.
- To assess columnstore indexes, look at
    - SELECT deleted_rows, total_rows FROM sys.dm_db_colum_store_row_group_physical_stats
        - Where more than >20% of rows have been deleted (due to DELETE or UPDATE), reorganize. This removes rows marked as deleted.
- Indexes can be reorganized or rebuilt
- ALTER INDEX [IndexName or ALL] ON Schema.Table …
    - REORGANIZE
        - REORGANIZE is always ONLINE.
    - REBUILD
        - [WITH ONLINE = ON]
            - An offline rebuild is generally quicker, but locks the index during this time.
            - Online rebuilds only require a lock right at the end.
        - [WITH (FILLFACTOR = 70)] -- FILLFACTOR leaves free space for inserted/updated rows.
        - [MAX_DURATION = 30 RESUMABLE = ON]-- it pauses after 30 minutes – cannot be used with "ALL". Used in SQL Server 2017+ or Azure SQL Database.
    - PAUSE / ABORT / RESUME – pauses/stops/restarts an ONLINE REORGANIZE or REBUILD. Used in SQL Server 2017+ or Azure SQL Database.
- Generally, REORGANIZE if >10% and <30%, and REBUILD is >30% - but this is a guide only.

51. assess performance-related database configuration parameters

- Auto Create Statistics
    - The database generates information about the contents of each column. Can be useful for deciding whether to use a scan or seek.

- Auto Shrink
  - ALTER DATABASE CURRENT SET AUTO_SHRINK ON; -- will enable auto-shrink.
  - Not recommended, as while it is less impactful to database performance, it is less effective.
  - Also, what happens when it needs to grow again?
- See also topic 55.

52. configure storage and infrastructure resources

- Virtual Machines:
  - When creating a VM, the "SQL Server settings – Change configuration" shows the storage.
  - Under "Configure storage":
    - Select "Transactional processing" or "Data warehousing" – this changes your stripe configuration, optimising it for traditional OLTP workloads or analytic/reporting workloads.
  - All of the SQL Server VM marketplace images follow default storage best practices.
    - See topic 11.
  - After setting the VM, when using disk caching for Premium SSD, you can select the disk caching level (by going to Settings – Disks):
    - It should be ReadOnly for SQL Server data files, as this improves reads from cache (VM memory and local SSD), which is much faster than from disk (Azure Blob storage).
    - It should be None for SQL Server Log files, as the data is written sequentially.
    - ReadWrite caching should not be used for the SQL Server files, as SQL Server does not support data consistency with this cache type. However, it could be used for the O/S drive, but it is not recommended to change the O/S caching level.
    - Any changes will require a reboot.
- For Azure SQL Database and Managed Instance:
  - You can configure the storage by going to Settings – "Compute + storage".

53. configure server and service account settings for performance

- See topic 9.

54. configure Resource Governor for performance

- Resource Governor is used in Azure SQL Database. However, it is not configurable.
- In VMs and Azure SQL MI, you can use Resource Governor to balance resources used by different sessions.
  - You can divide resources (CPU, physical I/O, and memory) differently, based on which workload it is in. This can improve performance on critical workloads.
- Terminology:

  - Resource pool – the physical resources. Two resource pools are created when SQL Server is installed: internal and default.
    - Without Resource Govenor enabled, all new sessions are classified into the default workload group, and system requests into the internal workload group.
  - Workload group – a container for requests which have similar criteria, and
  - Classification – that criteria.
- To enable/disable Resource Governor:
  - In SSMS
    - Right hand click Management – Resource Governor, and select Properties.
    - Click "Enable Resource Governor", and click OK.
  - In T-SQL
    - ALTER RESOURCE GOVERNOR RECONFIGURE [or DISABLE];
    - GO
- Create a Resource Pool
  - In SSMS
    - Right hand click Management – Resource Governor, and select Properties.
    - Click on the first column in the empty pool. It now has a *.
    - Double-click the empty cell in the Name, and enter the resource pool Name.
    - Add other values.
  - In T-SQL:
    - CREATE RESOURCE POOL myPool
    - WITH (MAX_CPU_PERCENT = 20); -- If you want to delete, use DROP RESOURCE POOL X
    - GO
    - ALTER RESOURCE GOVERNOR RECONFIGURE;
    - GO
  - Settings:
    - MIN_CPU_PERCENT and MAX_CPU_PERCENT
      - Guaranteed average and maximum average CPU.
      - The Max_CPU_PERCENT only applies if there is >100% if all requests were honoured.
      - e.g. Department A has min of 60%, and Department B has max of 40%.
    - CAP_CPU_PERCENT
      - Hard limit for CPU (not maximum average)

        - MIN_ and MAX_MEMORY_PERCENT
            - Memory may remain in a pool, even with no requests.
        - MIN_ and MAX_IOPS_PER_VOLUME
            - The physical I/O operations per seconds (IOPS).

- Workload Groups:
    - Requests go into the default group if:
        - There are no criteria,
        - The resource pool specified is non-existent.
        - There is a general classification failure.
- Create a Workload Group
    - In SSMS
        - Right hand click Management – Resource Governor, and select Properties.
        - Click on the relevant resource pool.
        - Go down to the "Workload groups for resource pool", and enter a name, with any other values.
    - In T-SQL
        - CREATE WORKLOAD GROUP myGroup -- or ALTER, if you wish to change it, or DROP to delete it.
        - USING myPool; -- or [default];
        - GO
- Create a classifier function in T-SQL:
    - CREATE FUNCTION fnClassifierTime()
    - RETURNS sysname
    - WITH SCHEMABINDING
    - AS
    - BEGIN
    - if DATEPART(HOUR,GETDATE())<8 or DATEPART(HOUR,GETDATE())>17
        - BEGIN
            - RETURN 'gOutsideOfficeHours';
        - END
    - RETURN 'gInsideOfficeHours';
    - END
- Register this classified function:

  - ALTER RESOURCE GOVERNOR with (CLASSIFIER_FUNCTION = dbo.fnClassifierTime);
  - ALTER RESOURCE GOVERNOR RECONFIGURE;
  - GO
- T-SQL
  - SELECT * FROM sys.resource_governor_configuration
    - Returns the stored Resource Governor state.
  - SELECT * FROM sys.dm_resource_governor_resource_pools
    - Returns information about the current resource pool state, the current configuration of resource pools, and resource pool statistics.
  - SELECT * FROM sys.dm_resource_governor_workload_groups
    - Returns workload group statistics and the current in-memory configuration of the workload group.

55. implement database-scoped configuration

- See also topic 51.
- In SSMS, you can:
  - Right-hand click on a database, and go to Properties and go to Options.
  - The settings in the topic are under "Database Scoped Configurations".
- In T-SQL, you can use:
  - ALTER DATABASE SCOPED CONFIGURATION [FOR SECONDARY] SET … = ON/OFF; -- for secondary is geo-replicated secondary database(s) (they all have the same settings).
- GLOBAL_TEMPORARY_TABLE_AUTO_DROP
  - Drop global temporary tables when not in use by any session.
    - Set in individual databases in Azure SQL Database.
    - Set in tempdb in MI and VMs.
- LAST_QUERY_PLAN_STATS
  - Enables/disables actual execution plans in sys.dm_exec_query_plan_stats.
- LEGACY_CARDINALITY_ESTIMATION
  - The query optimizer cardinality estimation model changed in SQL 2014. Should only be turned on for compatibility purposes.
- MAXDOP – intra-query parallelism; the maximum number of parallel threads
  - Having parallel threads should increase query speed.
  - Too high a MAXDOP may cause performance problems when executing multiple queries at the same time, as it may stave new queries of resources. Could reduce MAXDOP if this happens.

  - The default for new Azure SQL Databases is 8, which is best of most typical workloads.

- 55. implement database-scoped configuration
- OPTIMIZE_FOR_AD_HOC_WORKLOADS
  - Stores a compiled plan stub when a batch is compiled for the first time, which has a smaller memory footprint. When it is compiled/executed again, it will be replaced with a full compiled plan.
- PARAMETER_SNIFFING
  - Evaluates Stored Procedures to create an execution plan.
  - On subsequent runnings, the computer uses the same execution plan.
    - No need to spend time and CPU evaluating. However, may be suboptimal for certain parameters.
- QUERY_OPTIMIZER_HOTFIXES
  - Regardless of the compatibility level, enables or disables query optimization hotfixes.
    - So you can have a compatibility level for SQL Server 2012, but have query optimization hotfixes that were released after this version.
- There are many more, but these are the main ones.

56. configure compute resources for scaling

- See topics 9-11.

57. configure Intelligent Query Processing (IQP)

- IQP is a suite of new features, improving performance. It is supported in Azure SQL Database, Azure SQL Managed Instance for compatibility level 150. For SQL Server VM, this is SQL Server 2019 and level 150.
  - There are 7 different features, some of which are also available on lower levels.
  - You can check which settings are enabled by:
    - SELECT * FROM sys.database_scoped_configurations
  - Note: server-wide configuration options can be found in:
    - SELECT * FROM sys.configurations
    - These can be configured (but not in Azure SQL Database) by using
      - EXEC sp_configure 'Configuration', X
- You can disable any of them (except APPROX_COUNT_DISTINCT) for all queries in a single database, or for a single query:
  - All queries: ALTER DATABASE SCOPED CONFIGURATION SET X = OFF. 'X' is the first heading.
  - One query - add at the end of the query: OPTION (USE HINT('Y'). 'Y' is the second heading.
  - To disable APPROX_COUNT_DISTINCT, don't use that function!

- [DISABLE_] BATCH_MODE_ADAPTIVE_JOINS
    - For Azure SQL Database, and SQL Server 2017 or higher. Needs a Columnstore index in the query or a table being referenced in the join, or batch mode enabled for rowstore.
    - Selects the Join type (Hash Join or Nested Loops Join) during runtime based on actual input rows, when it has scanned the first input.
    - It defines a threshold (where the small number of rows makes a Nested Loops join better than a Hash join) that is used to decide when to switch to a Nested Loops plan.
    - Enabled by default in SQL Server 2017 under compatibility level 140, and Azure under compatibility level 140.
- APPROX_COUNT_DISTINCT
    - You can use the new aggregation APPROX_COUNT_DISTINCT.
    - Provides an approximate COUNT DISTINCT for big data – decreases memory and performance requirement. It guarantees up to a 2% error rate (within a 97% probability).
        - Use where absolute precision is not important, but responsiveness is.
    - Available in all compatibility levels of Azure SQL Database, and in SQL Server 2019 or higher.
- BATCH_MODE_ON_ROWSTORE / DISALLOW_BATCH_MODE
    - For Data Warehouse workloads.
    - Queries can work on batches of rows instead of one row at a time, when catched.
    - This happens automatically when the query plan decides it is appropriate in Compatibility Mode 140 for Batch Mode, and Mode 150 (SQL Server 2019+) for Row mode. No changes are required.
- [DISABLE_] INTERLEAVED_EXECUTION_TVF
    - Enabled by default in (Azure or SQL Server 2017+) and Compatibility Level 140+.
    - Use the actual cardinality of a multi-statement table valued functions on first compilation, rather than a fixed guess (100 rows from SQL Server 2014).
    - Statements must be read-only – so no INSERT, UPDATE or DELETEs.
- [DISABLE_]BATCH_MODE_MEMORY_GRANT_FEEDBACK (Batch or Row mode)
    - Enabled by default in (Azure or SQL Server 2017+) and Compatibility Level 140+.
    - SQL Server looks how much memory is allocated to a cached query, and then allocates same amount of memory next time (instead of guessing, then adding more, more, more).
        - If a query spills to disk, add more memory for consecutive executions. If it wastes 50+% of the memory, reduce memory for consecutive executions.
- [DISABLE_] TSQL_SCALAR_UDF_INLINING
    - Enabled by default in (Azure or SQL Server 2019+) and Compatibility Level 150+.
    - Scalar UDFs often perform poorly due to:

        - Running multiple times, once per row.
        - Unable to actually work out the performance cost.
        - Unable to optimize more than one SELECT statement at once.
        - No parallelism in queries which invoke UDFs.
    - Scalar UDFs are transformed into equivalent relational expressions inlined into the query, often resulting in performance gains.
        - Does not work with all UDFs, including those which have multiple RETURN statements.
        - Can also be disabled for a specific UDF by adding "WITH INLINE = OFF" before "AS BEGIN".
- [DISABLE_] DEFERRED_COMPILATION_TV
    - Similar to INTERLEAVED_EXECUTION_TVF, but for Table Variables.
    - Use the actual cardinality of the table variable encountered on first compilation instead of a fixed guess (1 row).

Optimize Query Performance

58. determine the appropriate type of execution plan

- To display Execution Plans in SSMS:
    - Click on "Display Estimated Execution Plan" (using query optimizer)
        - This happens immediately without running the query.
    - Click on "Include Actual Execution Plan" (including additional runtime statistics).
        - This is displayed as a separate tab when the query is run.
        - or use
            - SET SHOWPLAN_ALL ON;
            - GO
                - or SHOWPLAN_TEXT
            - Neither version executes the statement, but displays execution information.
    - Live Query Statistics (updated while the query is running).
- Execution plans run from the right to the left.
    - The arrow thickness represent number of rows.
        - You can hover over them for more details.
        - More estimated rows = more memory reserved.
    - Note the degree of parallelism (also shown in properties).
- There are 3 main types of joins between tables:

  - Nested Loops joins.
    - Use when
      - Input1 is small.
      - Input2 is large.
      - Input2 is indexed on the join.
    - Uses least I/O and fewest comparisons.
    - It uses the top input (in the execution plan) and takes 1 row.
    - It then searches for matches rows in the bottom input.
  - Merge joins
    - Use when
      - Input1 and Input 2 is not small.
      - Input1 and Input2 are sorted on their join – or if not, possibly when Input1 and Input2 are of a similar size. Then, the Sort might be worth the time compared with the Hash Join.
    - Can be very fast.
  - Hash joins
    - For large, unsorted, nonindexed inputs.
    - Also used in the middle of complex queries, as intermediate results are often not indexed or suitably sorted.
- In SQL Server 2017, a Batch mode Adaptive Join was introduced.
  - This converts into a Hash Join or Nested Loops join after the first input has been scanned, when it uses Batch mode.
  - More in topic 57.

59. identify problem areas in execution plans

- Are you using SELECT *?
  - Can you narrow down the columns? If so, maybe you can then use indexes.
- Have a look at the cost of each operation.
  - Is there one that can be improved?
    - Is there a Sort? It's expensive – do you really need it? If so, could you have an Index which has already sorted on those columns?
    - Do you use parameters? If so, and the performance is based, can you WITH RECOMPILE the stored procedure, or use OPTION (RECOMPILE) for queries.
  - Is there a Scan when you are using a WHERE?
    - If so, could a Seek be better? It may need an index.

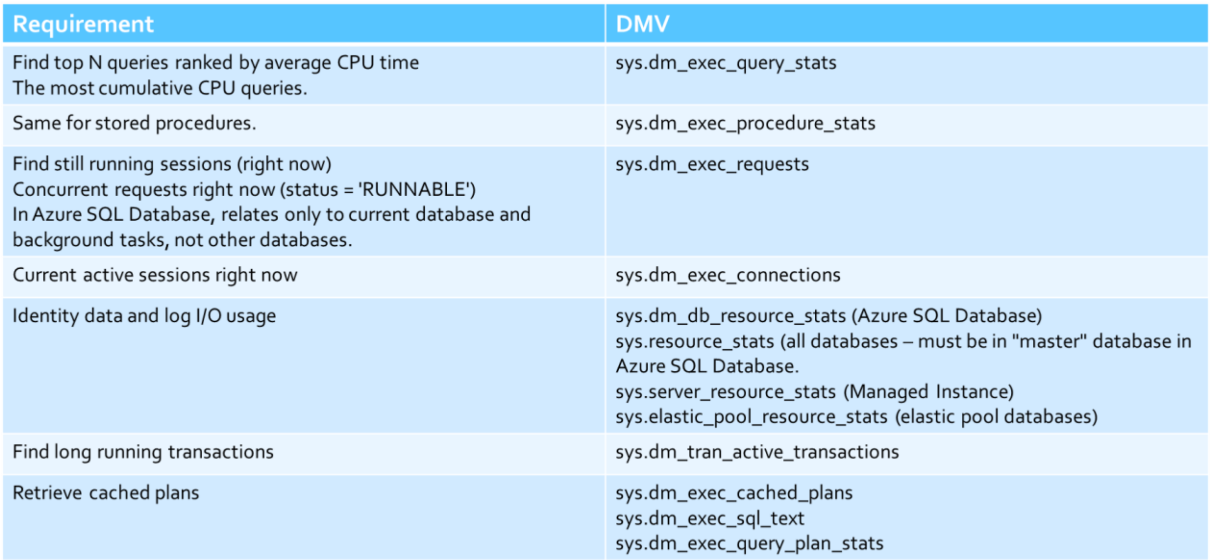
        - Are you using a Heap? Do you need a clustered index?
        - Could you use a SARGable predicate in the WHERE clause?
            - e.g. don't use YEAR, use BETWEEN for dates.
            - don't use LEFT – use LIKE for strings.
            - don't use ISNULL(X, 'Y') function – use (X IS NULL or X = 'Y')
    - Is there a RID Lookup or a Key Lookup Operator?
        - If so, could you use an INCLUDE with the index? This writes the data into the index, but in a separate part of the index away from the Key – so it's quicker, but doesn't slow down the index much.
            - It's also useful for Unique indexes – INCLUDE columns are included in the Uniqueness test.
    - Are the field types too wide?
        - This will increase the row size, increasing time to retrieve the data.
- Different loops
    - Are you using a Hash Join when, with some changes, a Merge Join or Nested Loop could be used?
        - Maybe need an index?
- Are you using a cursor?
    - Use a set-based operation instead.

60. extract query plans from the Query Store

- To use in T-SQL:
    - The query plans are stored in:
        - SELECT * FROM sys.query_store_plan
    - Statistics about it can be found:
        - SELECT * FROM sys.query_store_runtime_stats
    - However, you can see the queries:
        - SELECT Txt.query_text_id, Txt.query_sql_text, Qry.*
        - FROM sys.query_store_query AS Qry
        - INNER JOIN sys.query_store_query_text AS Txt
        - ON Qry.query_text_id = Txt.query_text_id ;
- To use in SSMS:
    - Note – you can click on "Configure" to change the time period. You can also click on "Track the selected query in a new window".
    - Go to the database – Query Stores:

- Regressed Queries
    - Have your query speed got worse? Have a look at Duration, CPU Time, Logical Reads, Physical Reads, and more
- Overall Resource Consumption
    - Are the resources used more during particular days, or daily/night?
- Top Resource Consuming Queries
    - The most extreme values in Duration, Execution Count, CPU Time etc.
- Queries with Forced Plans
- Queries with High Variation
    - Varied amount of duration, CPU time, I/O and memory.
- Queries Wait Statistics
    - You can click on the categories (e.g. High Memory, Lock, Buffer I/O or CPU waits) to get detail on that category.
    - Data from sys.query_store_wait_stats
- Tracked Queries
    - Track individual queries – you need to enter the Query ID.
    - You can force query to use a particular plan ("Force Plan")
    - Circle = query completed. Square = cancelled by client. Triangle = Failed due to an exception aborted execution.
    - Look for any missing indexes in the Query view.

61. determine the appropriate Dynamic Management Views (DMVs) to gather query performance information



Requirement	DMV
Find top N queries ranked by average CPU time The most cumulative CPU queries.	sys.dm_exec_query_stats
Same for stored procedures.	sys.dm_exec_procedure_stats
Find still running sessions (right now) Concurrent requests right now (status = 'RUNNABLE') In Azure SQL Database, relates only to current database and background tasks, not other databases.	sys.dm_exec_requests
Current active sessions right now	sys.dm_exec_connections
Identity data and log I/O usage	sys.dm_db_resource_stats (Azure SQL Database) sys.resource_stats (all databases – must be in "master" database in Azure SQL Database. sys.server_resource_stats (Managed Instance) sys.elastic_pool_resource_stats (elastic pool databases)
Find long running transactions	sys.dm_tran_active_transactions
Retrieve cached plans	sys.dm_exec_cached_plans sys.dm_exec_sql_text sys.dm_exec_query_plan_stats

- You can retrieve the last execution plans using:

- SELECT *
- FROM sys.dm_exec_cached_plans AS cp
- CROSS APPLY sys.dm_exec_sql_text(plan_handle) AS st
- CROSS APPLY sys.dm_exec_query_plan_stats(plan_handle) AS qps;

- You can also store execution plans using:
  - Extended Events
    - Lightweight profiling
    - Add at the end of a query OPTION(USE HINT ('QUERY_PLAN_PROFILE') would add it to Extended Events
- Find top N queries ranked by average CPU time using sys.dm_exec_query_stats
  - SELECT TOP 5 query_stats.query_hash AS "Query Hash",
  - SUM(query_stats.total_worker_time) / SUM(query_stats.execution_count) AS "Avg CPU Time",
  - MIN(query_stats.statement_text) AS "Statement Text"
  - FROM
  - (SELECT QS.*,
    - SUBSTRING(ST.text, (QS.statement_start_offset/2) + 1,
    - ((CASE statement_end_offset
    - WHEN -1 THEN DATALENGTH(ST.text)
    - ELSE QS.statement_end_offset END
      - QS.statement_start_offset)/2) + 1) AS statement_text
  - FROM sys.dm_exec_query_stats AS QS
  - CROSS APPLY sys.dm_exec_sql_text(QS.sql_handle) as ST) as query_stats
  - GROUP BY query_stats.query_hash
  - ORDER BY 2 DESC;
- Find which queries use the most cumulative CPU:
  - SELECT
  - highest_cpu_queries.plan_handle,
  - highest_cpu_queries.total_worker_time,
  - q.dbid, q.objectid, q.number, q.encrypted, q.[text]
  - FROM
  - (SELECT TOP 50 qs.plan_handle, qs.total_worker_time
  - FROM sys.dm_exec_query_stats qs

  - ORDER BY qs.total_worker_time desc) AS highest_cpu_queries
  - CROSS APPLY sys.dm_exec_sql_text(plan_handle) AS q
  - ORDER BY highest_cpu_queries.total_worker_time DESC;

62. identify performance issues using DMVs

- Long running queries that consume CPU are still running
  - SELECT TOP 10 req.session_id, req.start_time, cpu_time 'cpu_time_ms', OBJECT_NAME(ST.objectid, ST.dbid) 'ObjectName', SUBSTRING(REPLACE(REPLACE(SUBSTRING(ST.text, (req.statement_start_offset / 2)+1, ((CASE statement_end_offset WHEN -1 THEN DATALENGTH(ST.text) ELSE req.statement_end_offset END-req.statement_start_offset)/ 2)+1), CHAR(10), ' '), CHAR(13), ' '), 1, 512) AS statement_text
  - FROM sys.dm_exec_requests AS req
  - CROSS APPLY sys.dm_exec_sql_text(req.sql_handle) AS ST
  - ORDER BY cpu_time DESC;
- Data from your database
  - USE master
  - GO
  - SELECT * FROM sys.resource_stats
  - WHERE database_name = 'X'
  - ORDER BY start_time DESC;
- Current active sessions
  - SELECT * FROM sys.dm_exec_connections
  - SELECT @@SPID gives the current session.

63. identify and implement index changes for queries

- Requirements for Indexes:
  - Big tables (small tables may use a Scan anyway).
  - Small column size (the best are numeric, but smaller text columns are OK too).
  - Use columns which are in WHERE (SARGable columns) and JOIN clauses.
    - If using LIKE '%text%', then an index (apart from a full-text index) will not help.
    - Additional columns can be included using INCLUDE (covered queries). This can make the index key smaller and more efficient.
  - Clustered or Non-clustered?
    - Only one clustered index per table. It also used in PRIMARY KEYs. It re-sorts the table. Use for frequently used queries and range queries.

- - - - Should be used with the UNIQUE property – but it is possible to create one which doesn't.
      - Should be either UNIQUE or have many distinct values.
      - Accessed sequentially (in ranges).
      - IDENTITY
      - Frequently used.
    - As many non-clustered indexes as you want. It creates a separate index.
  - Columnstore indexes are available in almost all service tiers.
  - Only need a small part indexes?
    - Use a Filtered Index
  - Do you need room to grow?
    - Use the FillFactor option to leave space for growth.
  - Ascending or Descending, based on how you want the results to appear.
- Too many indexes?
  - If you INSERT, UPDATE, DELETE or MERGE, then all indexes need to be adjusted.
- Create in T-SQL:
  - CREATE [UNIQUE] [NONCLUSTERED/CLUSTERED] INDEX [Name] ON Schema.Table (Columns) [INCLUDE (Columns)] [WHERE … - filtered index]
- Create in SSMS:
  - Right-hand click on Indexes in the relevant table and select "New Index" – "[Non-]Clustered Index".

64. recommend query construct modifications based on resource usage

- Missing indexes:
  - SELECT * FROM sys.dm_db_missing_index_details
    - In Azure SQL Database, only gives information about databases to which user has access.
    - In creating the index, put equality before inequality – both of these should be the key – and INCLUDE the included columns.
- Full query from https://docs.microsoft.com/en-us/azure/azure-sql/database/performance-guidance:
  - SELECT
  - CONVERT (varchar, getdate(), 126) AS runtime
  - , mig.index_group_handle
  - , mid.index_handle
  - , CONVERT (decimal (28,1), migs.avg_total_user_cost * migs.avg_user_impact *

        - (migs.user_seeks + migs.user_scans)) AS improvement_measure
    - , 'CREATE INDEX missing_index_' + CONVERT (varchar, mig.index_group_handle) + '_' +
        - CONVERT (varchar, mid.index_handle) + ' ON ' + mid.statement + '
        - (' + ISNULL (mid.equality_columns,'')
        - + CASE WHEN mid.equality_columns IS NOT NULL
        - AND mid.inequality_columns IS NOT NULL
        - THEN ',' ELSE '' END + ISNULL (mid.inequality_columns, '') + ')'
        - + ISNULL (' INCLUDE (' + mid.included_columns + ')', '') AS create_index_statement
    - , migs.*
    - , mid.database_id
    - , mid.[object_id]
    - FROM sys.dm_db_missing_index_groups AS mig
    - INNER JOIN sys.dm_db_missing_index_group_stats AS migs
    - ON migs.group_handle = mig.index_group_handle
    - INNER JOIN sys.dm_db_missing_index_details AS mid
    - ON mig.index_handle = mid.index_handle
    - ORDER BY migs.avg_total_user_cost * migs.avg_user_impact * (migs.user_seeks + migs.user_scans) DESC

65. assess the use of hints for query performance

- Hints only affect one particular DELETE, INSERT, SELECT, UPDATE or MERGE query.
    - Microsoft says that the Query Optimizer typically selects the best execution plan, so only use this as a last resort.
- Join hints can be LOOP, HASH, MERGE JOIN.
    - For example, INNER LOOP JOIN instead of INNER JOIN.
        - LOOP cannot be specified with a RIGHT or FULL Join.
- For query hints, end your query with OPTION (<hints>)
    - For example, OPTION (MERGE JOIN)
    - {HASH | ORDER } GROUP
    - {MERGE | HASH | CONCAT} UNION
    - {LOOP | MERGE | HASH} JOIN
    - KEEPFIXED PLAN
        - The query won't be recompiled because the statistics change. It will only recompile if the schema of the underlying tables changes or sp_recompile is run against these tables.

  - KEEP PLAN
    - Recompiles less often when statistics change
  - OPTIMIZE FOR UNKNOWN
    - Uses the average selectivity of a predicate, as opposed to the runtime parameter used when the query is compiled and optimized.
  - ROBUST PLAN
    - Creates a plan that works for the maximum potential row size. If it isn't, then performance may be impaired.
- If using parameters in a stored procedure, you can say
  - OPTION (OPTIMIZE FOR (@param 2))
  - or
  - OPTION (OPTIMIZE FOR (@param UNKNOWN))
  - You could also have WITH RECOMPILE before "AS BEGIN".
- Otherwise, the stored procedure will be optimised as per the first running.
- See also topic 57.

66. identify data quality issues with duplication of data

- Minimize duplication of data
- Reduce data modification issues
- Simplify queries

67. identify normal form of database tables

- 1st Normal Form (1NF)
  - Requirements
    - The Values in each column must be atomic (indivisible),
    - Each value contains only a single value.
  - Actions
    - Eliminate repeating groups in individual tables
    - Create a separate table for each set of related data
    - Identify each set of related data with a primary key
- 2nd Normal Form (2NF)
  - Requirements
    - Is in 1st Normal Form
    - Reduce repeating information

- Actions
    - Create separate tables for values that apply to multiple records.
    - Relate tables with a foreign key
- 3rd Normal Form (3NF)
    - Requirements
        - Is in 2nd Normal Form
        - Values that are not part of a record’s key are to be removed from the table.
    - Action
        - Remove fields that are not dependent on the key
- 67. identify normal form of database tables
- 4th Normal Form (4NF)
    - Requirements
        - It should be in 3rd Normal Form.
        - It should have no multi-valued dependency
    - Actions
        - Separate out non-trivial multivalued dependences
- 5th Normal Form (5NF)
    - Requirements
        - It should be in 4th Normal Form.
        - It should have no join dependency
    - Actions
        - Remove join dependences to remove redundancy

68. assess index design for performance

- See topic 63.

69. validate data types defined for columns

- Exact numerics
    - bigint, int, smallint, tinyint, bit
    - decimal/numeric
    - money, smallmoney
- Approximate numerics
    - float, real
- Date and time
    - date, datetime, datetime2, datetimeoffset, smalldatetime, time

- Character strings
    - char, varchar, text, varchar
    - nchar, nvarchar, ntext, nvarchar(max)
- Other data types
    - binary, varbinary
    - cursor, geography, geometry, hierarchyid, rowversion, sql_variant, table, uniqueidentifier, xml

70. recommend table and index storage including filegroups

- This is for Azure SQL MI and VM.
    - Azure SQL Database supports only one database file (except in Hyperscale).
- There are three type of database files:
    - Primary file
        - Start-up information.
        - There is only one primary file per database.
        - Recommended filename extension ".mdf".
    - Secondary file
        - Additional, but optional, user-defined data files (zero to multiple). Cannot be used in Azure SQL Database.
        - Can be on separate disks.
        - Recommended filename extension ".ndf".
    - Transaction Log
        - Information needed to recover database.
        - One to multiple transaction logs.
        - Recommended filename extension ".ldf".
    - A file can only be used by one database.
    - Simple databases can have a single data file and a single transaction log file.
- Have two different file name:
    - Logical file name – used in T-SQL statements.
    - O/S file name – its location, including directory path (you can set this on VM only).
- Storage size
    - Can grow automatically, by a percentage or a fixed file size ("growth increment").
    - Has a maximum size.
- Filegroups

  - Contains multiple files for admin, data allocation or storage purposes. Not used in Azure SQL Database.
  - By default, the "default" filegroup is the PRIMARY filegroup. However, you can change it.
    - ALTER DATABASE DatabaseName
      MODIFY FILEGROUP FileGroupName DEFAULT;
  - The primary filegroup contains the primary file, system tables. The default filegroup (which may be the same) contains any other objects where you have not specified a filegroup.
  - Other filegroups are called "User-defined" filegroups.
    - There are other filegroups, called "Memory Optimized Data" and "Filestream".
  - A file can only be contained in one filegroup.
  - A filegroup can only be used by one database.
  - Transaction logs are not part of a filegroup.
  - If you use multiple data files, Microsoft recommends that you create a second file group for the other files and make that filegroup the default filegroup.
- You can create files and filegroups in T-SQL and in SSMS.
  - ALTER DATABASE [MyDatabase]
    - ADD FILEGROUP [NewFileGroup]
  - GO
  - ALTER DATABASE [MyDatabase]
    - ADD FILE (NAME = N'NewData',
    - FILENAME = N'C:\PathToData\NewData.ndf' ,
    - SIZE = 8192KB , FILEGROWTH = 65536KB)
      - or FILEGROWTH = 10%
    - TO FILEGROUP [NewFileGroup]
- For information on a table, use:
  - sp_help 'Schema.TableName'

71. evaluate table partitioning strategy

- Data can be partitioned – physically divided into different data stores.
  - Scalable – there are hardware limits, but if you divide data into partitions, each on a separate server, it can be scaled out.
  - Increase performance – Smaller amount of data in a single partition, and multiple data stores can be accessed at the same time.
  - Security considerations – apply different security to sensitive and non-sensitive partitions.
  - Administration – have different strategies of monitoring, management and backup per partition. Backups for a single partition are quicker than for the entire data.

  - Have different hardware or services – Premium or Standard where needed.
  - Increase availability – if one instance fails, only that partition is temporarily unreadable.
- Partitions can be:
  - Horizontal partitioning (sharding).
    - All partitions have the same schema.
    - Each partition, or shard, holds a subset of the data (rows).
    - If some data is fairly static or small, consider replicating it in all partitions, to reduce cross-partition access.
  - Vertical partitioning.
    - Each partition holds a subset of the columns.
    - Some columns may be needed less often, and they could be separated away, and used only when needed.
    - Some columns may also be more sensitive, and could be separated away.
    - All partitions would need to be capable of being joined – for instance, by the same primary key in each.
  - Functional partitioning.
    - Different tables in each partition relating to function.
    - Store data could be in one partition, and employee data in another.
    - Some tables could be more sensitive, and could be separated away into another partition.
  - These techniques can be combined.
  - Keep the data, where possible, geographically close to the users.
- Consider the backup, archiving (including deleting) and High Availability, Disaster Recovery requirements for each partition.

72. evaluate the use of compression for tables and indexes

- Why compress?
  - Reduced space – useful for data which is infrequently used.
  - However, it requires extra time and CPU, both to compress and retrieve data.
  - You can compress at the row level, the page (8,192 characters) level, or none.
    - For row compression
      - Numeric types (apart from tinyint) storage will be reduced, maybe down to 1 byte. Tinyint already takes 1 byte.
      - Some date types will be compressed: datetime, datetime2 and datetimeoffset. The others would not benefit from compression.

            - char and nchar will be compressed, up to 50% in English, German, Hindi and Turkish, but only up to 40% in Vietnamese and 15% in Japanese. varchar and nvarchar types would not benefit from compression.
        - Page compression consists of three operations in this order:
            - Row compression
            - Prefix compression
                - If values in the same column start with the same characters, this can be optimised.
                - A common prefix per column is moved to the Compression Information structure immediately after the page header.
                - A reference is made in that value to the prefix, and the number of characters which are in common.
            - Dictionary compression
                - If values after prefix compression in <u>any</u> column are the same, this can be optimised.
                - Again, these common values are moved to the Compression Information structure, and a reference is made.
            - Pages are uncompressed at first. Page compression is only used when additional rows can be fitted on a full page.
    - Compression does not affect backup and restore.
- Available in:
    - Azure SQL Database, Azure SQL MI and
    - SQL Server on VMs
        - from SQL Server 2016 SP1 in all editions, and
        - before than, only in the Enterprise edition.
- You can compress:
    - Tables stored with a clustered index or without (a heap).
        - You cannot use data compression with tables which have SPARSE columns.
        - To change the compression option in a clustered index, you need to drop the clustered index, preferably OFFLINE, and then rebuild the table.
    - A complete nonclustered index.
        - By default, they are not compressed.
    - A complete indexed view.
- You cannot compress system tables.
- Different partitions can be compressed using different settings.

- Would compression be useful?
    - The following storage procedure cannot be used in Azure SQL Database.
        - EXEC sp_estimate_data_compression_savings
            - 'SchemaName',
            - 'TableName',
            - Index_ID – either zero for a Heap, 1 for a clustered Index, or >1 for Non-clustered Index. NULL if a table, and not an index,
                - To get the index number, use:
                    - SELECT name, index_id
                    - FROM sys.indexes
                    - WHERE OBJECT_NAME (object_id) = N'TableName';
            - PartitionNumber (or NULL)
                - To get the partition number, use:
                    - SELECT *
                    - FROM sys.partitions
                    - WHERE OBJECT_NAME (object_id) = N'TableName';
            - 'ROW' (or 'PAGE', 'NONE')
- To enable compression:
    - In SSMS
        - Right-hand click on the table or index, and go to Storage – Manage Compression
        - Click next, and select the compression type for each partition.
            - You can also click on "Use same compression type for all partitions".
            - You can also click on "Calculate" to calculate space requirements (not Azure SQL Database).
        - Select whether to run immediately or to create a script (to a file, clipboard, or new query window).
            - If using this on a VM, you may also get "Schedule – you could select: one time, recurring (Daily, Weekly or Monthly), when SQL Server Agent starts, or whenever the CPUs become idle.
    - In T-SQL - table
        - ALTER TABLE Schema.TableName
        - REBUILD PARTITION = 1 | ALL WITH (DATA_COMPRESSION = PAGE | ROW | NONE);
    - In T-SQL – index
        - ALTER INDEX IndexName ON Schema.TableName

        - REBUILD PARTITION = ALL WITH (DATA_COMPRESSION = PAGE | ROW | NONE);

- For columnstore objects:
    - Initially used in data warehouses, but then expanded.
    - Columns are always compressed.
    - Indexes work best when you scan large amounts of data, like fact tables in data warehouses.
        - They are generally clustered. Non-clustered only uses when you have a data type not supported by a clustered index – e.g. XML, text and image.
    - They can be further compressed using archival compression.
        - Best used when the data is not often read, but you need the data to be retained for regulatory or business reasons.
        - Saves space, but there is a high CPU cost to uncompressing it, which is more than any I/O saving.

Perform Automation of Tasks

73. manage schedules for regular maintenance jobs

- For Azure SQL Database, see topic 46.
- This is for SQL Server on a VM, and Azure SQL MI, but not Azure SQL Database, as it uses SQL Server Agent.
    - SQL Server Agent doesn't need to be enabled on Azure SQL MI – it is always running.
    - It doesn't have all of the functionality of on-prem SQL Server, but it has most of it.
- To create a new job:
    - Go to SQL Server Agent (right-hand click it and Start if needed on a VM) – Jobs.
    - Right-hand click and go to "New Job".
    - Enter a job name.
    - Go to the Steps tab and click New.
    - Enter the First Step name, select the database, and which user is running the command, and enter your T-SQL command.
    - Click "Parse" to check the syntax.
    - Add additional sets as needed.
- To create a schedule for a job:
    - Go to SQL Server Agent (right-hand click it and Start if needed on a VM) – Jobs.
    - Right-hand click and go to Properties.
    - Go to Schedules and click New.
    - Enter:
        - A name,

        - Whether it is:
            - One time,
            - Recurring (Daily, Weekly, Monthly – and when),
            - "Start whenever the CPUs become idle" or
            - "Start automatically when SQL Server Agent starts" – this setting is not supported in MI.
    - If you subsequently want to Edit or Remove it, you can click those buttons.
    - If you want to import a previously made schedule, click "Pick" and then choose the schedule.
- To do this in T-SQL:
        - USE msdb ;
        - GO
        - EXEC sp_add_schedule
        - @schedule_name = N'ScheduleName' ,
        - @freq_type = 4,
            - 1 = Once, 4 = Daily, 8 = Weekly, 16 = Monthly (day of month), 32 = Monthly (Xth Sunday, for example), 64 = When SQL Agent service stats, 128 = When computer is idle.
        - @freq_interval = 1, -- Fairly complex
        - @active_start_time = 012345 ;
        - GO
        - EXEC sp_attach_schedule
        - @job_name = N'JobName',
        - @schedule_name = N'ScheduleName' ;
        - GO
- To view schedules:
    - USE msdb ;
    - GO
    - select *
    - from sysschedules
- You can see when jobs have run by:
    - Going to SQL Server – Job Activity Monitor

74. configure multi-server automation

- See topic 46 for Azure SQL Database.

- For MI and VM, you need a master server and one or more target servers.
    - A target server can be linked to only one master server.
- To create a Master server:
    - Right-hand click on SQL Server Agent, and go to Multi Server Administration – Make this a Master.
    - Enter any addresses for notifications.
    - Add your target servers (by clicking on "Add Connection", if they are not already registered).
    - After checking that the servers are compatible, you can "create a new login if necessary and assign it rights to the MSX".
        - MSX is the "Master Server".
- To create a Target server:
    - Right-hand click on SQL Server Agent, and go to Multi Server Administration – Make this a Target.
    - Select the Master Server.
    - You can "create a new login if necessary and assign it rights to the MSX".
- When creating jobs:
    - You can go to the Targets page and select "Target local server" or "Target multiple servers".

75. configure notifications for task success/failure/non-completion

- For Azure SQL Database, see topic 46.
- To create an operator:
    - Go to SQL Server Agent (right-hand click it and Start if needed on a VM) – Operators.
    - Right-hand click and select "New Operator".
    - Enter Name and e-mail name and/or pager e-mail name (and pager timings).
        - Pager functionality has been deprecated, and will be removed in a future version.
- In T-SQL, use:
    - USE msdb ;
    - GO
    - EXEC dbo.sp_add_operator
        - @name = N'OperatorName',
        - @enabled = 1, -- 1 = enabled, 0 = not enabled.
        - @email_address = N'EmailAddress'
    - There are also pager arguments as well.
- To configure notifications:

  - Go to SQL Server Agent (right-hand click it and Start if needed on a VM) – Jobs.
  - Right-hand click a job and go to Properties.
  - Go to Notifications, and
    - Select Email, Page(r), "Write to the Windows Application event log" and "Automatically delete job"
    - When the job fails, succeeds, or completes (fails or succeeds).
    - This is for the entire task, not individual steps.
    - The email and pager need to be already created.
- In T-SQL, use:
  - USE msdb ;
  - GO
  - EXEC dbo.sp_add_notification
  - @alert_name = N'NameOfAlert',
  - @operator_name = N'OperatorName',
  - @notification_method = 1 ;
    - 1 = Email, 2 = Pager, 4 = Net Send.
    - Pager and Net Send have been deprecated.
- To configure Database Mail, you need to:
  - Enable Database Mail.
    - This is not necessary on Azure SQL MI, as it is enabled by default.
    - In SSMS, go to the Server – Management.
    - Right-hand click on "Database Mail", and click Configure Database Mail.
  - Create a Database Mail account for the SQL Server Agent service account to use.
    - In SSMS, go to the Server – Management.
    - Double-click "Data Mail".
    - In the "Database Mail Configuration Wizard", select "Manage Database Mail accounts and profiles".
    - Select "Create a new account".
  - Create a Database Mail profile for the SQL Server Agent service account to use and add the user to the DatabaseMailUserRole in the msdb database.
    - As above, but select "Create a new profile".
    - For Azure MI, your profile must be called AzureManagedInstance_dbmail_profile if you want to send e-mail using SQL Agent jobs.
  - Set the profile as the default profile for the msdb database.

        - In the "Manage Profile Security", "Default Profile" should say "Yes".

76. create event notifications based on metrics

- This is for SQL Server VM, as it uses the SQL Server Agent. It is not relevant for Azure SQL Database or MI (which has SQL Server Agent, but doesn't allow for event notifications).
- To create an event notification in SQL Server.
- Go to SQL Server Agent (right-hand click it and Start if needed on a VM) – Alerts.
- Right-hand click and go to "New Alert".
- Enter a Name for this alert.
- Select an Alert type:
    - SQL Server event
        - This is based on an error number or severity.
    - WMI event alert
        - This uses the Windows Management Instrumentation to monitor events in SQL Server.
    - SQL Server performance conditions. You select the:
        - Object, such as Databases or General Statistics.
        - Counter, such as Transactions/sec.
        - Instance – a database.
        - Alert if counter falls below, becomes equal to, or rises above a Value.
- In the Response page, you can:
    - Execute an SQL Server Agent job.
        - You can click New Job, or View [Existing] job (once you have selected one),
    - and/or Notify an operator
        - You can click "New Operator", or View [Existing] operator (once you have selected one).
- In the Options page, you can:
    - Include the alert error text in email or pager,
    - Add an additional notification message, and
    - Have a delay between responses. 0 minutes and 0 seconds indicate that you want a response for every occurrence of the alert.

77. create event notifications for Azure resources

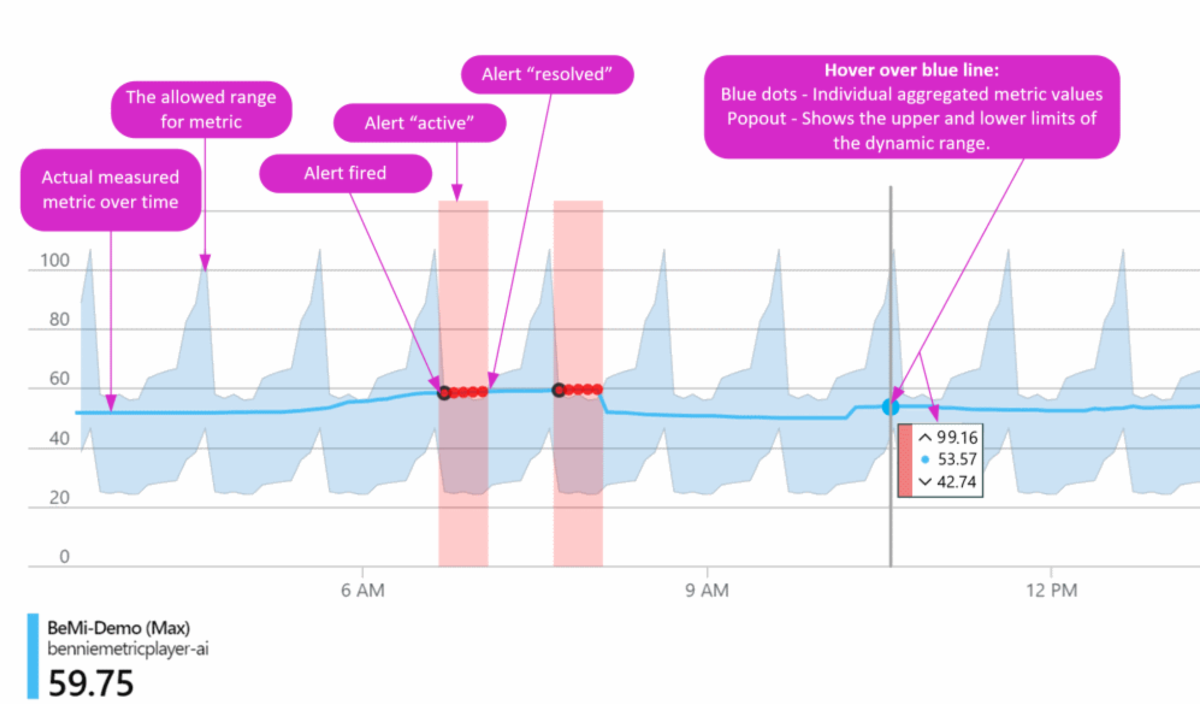
- To create an event notification:
    - Go to the Azure Portal, and the specific database.
    - Go to Monitor – Metrics or Alerts.

- If "Metrics":
  - Select a metric.
  - Click "+ New alert rule".
- If "Alerts":
  - Click "+ New alert rule".
  - Select a metric.
- Click on Conditions:
  - Optionally, select your signal type and monitor service
  - Select your signal (measure).
  - Select the Threshold – static or dynamic.
    - Dynamic thresholds learns the data and models it using algorithms and methods, detecting pattern such as seasonality (hourly, daily, weekly).
  - If static, select:
    - the operator (>, >=, < or <=),
    - The aggregation type (Avg, Min, Max, Count, Sum), and
    - the threshold value.
  - If dynamic, select
    - Select the operator (greater than the upper threshold and/or below the lower threshold)
    - The aggregation type, and
    - The Threshold sensitivity:
      - High – more alerts based on small deviations. greater than the upper threshold and/or smaller than the lower threshold).
      - Medium (default), and
      - Low – fewer alerts based on large deviations.
    - You can also select, in Advanced settings:
      - The evaluation period, and
      - The number of violations to occur within that evaluation period in order to trigger the alert.
- Under "Evaluated based on", select:
  - Aggregation granularity period – how often the measures are grouped together,
  - The frequency of evaluation – how often should it be checked.

- Under to the "Actions" section,
  - Select an existing action group, or

- Click on "Create action group", and select:
  - Email,
  - SMS (text message),
  - Azure app Push Notifications, and/or
  - Voice.
- And/or select Actions:
  - Automation Runbook, Azure Function,
  - ITSM, Logic App, Webhook.

- Under the "Alert rule details", enter:
  - The name
  - Description (optional),
  - Subscription and resource group,
  - Severity – from 0 (Critical) to 4 (Verbose),
  - Whether it is enabled on creation, and
  - Whether the automatically resolve alerts.
    - The alert period is shown in a different color when "unresolved"
      - The line turns from blue to red dots, and the background turns light red as well.



- To create a Kusto query from logs:
  - Go to the Azure Portal, and the specific database.

  - Go to Monitoring – Logs.
  - Select a query and click "Run".
  - Have a look at the results.
- To create an alert rule
  - Click "New alert rule"
  - Under Measurement, select:
    - Measure,
    - Aggregation type (Average, Total, Maximum or Minimum),
    - Aggregation granularity (5, 10, 15, 30 or 45 minutes, 1-6 hours, or 1-2 days).
  - You can also split by dimensions.
  - Enter your "Alert logic":
    - Operator (>, >=, <, <= or =),
    - Threshold value, and
    - Frequency of evaluation (5, 10, 15, 30 or 45 minutes, 1-6 hours, or 1-2 days).
- Go to the "Actions" tab and:
  - Select an existing action group, or
  - Click on "Create action group", and select:
    - Email,
    - SMS (text message),
    - Azure app Push Notifications, and/or
    - Voice.
  - And/or select Actions:
    - Automation Runbook, Azure Function,
    - ITSM, Logic App, Webhook
- Go to the "Details" tab, and:
  - Select the severity from 0 (Critical) to 4 (Verbose).
  - Enter an "alert rule name" and description (optional).
- You can also do all this by:
  - Going to the Azure Portal, and the specific database, and go to Monitoring – Alerts, "+New alert rule", and selecting:
    - Resource,
    - Condition and Alert logic,

        - The signal could be a platform metric, or an activity log (an administrative operation).
    - Actions Groups, and
    - Alert Details.

78. create alerts for server configuration changes

- In MI or SQL Server on VM, changes are made to a server configuration by using sp_configure:
- This tracing may already be enabled. To enable the tracing:
    - sp_configure 'show advanced options', 1
    - GO
    - RECONFIGURE
    - GO
    - sp_configure 'default trace enabled', 1
    - GO
    - RECONFIGURE
    - GO
- You can see what changes have been made by:
    - In SSMS, you can right-hand click on the server instance (not the database),
    - Go to Reports – Standard Reports – Configuration Changes History.
- For SQL Database, see topic 77.

79. create tasks that respond to event notifications

- As per 76, using an Error event.

80. perform automated deployment methods for resources

- Use an ARM Template to deploy resources.
    - Written in JSON.
    - https://docs.microsoft.com/en-us/azure/azure-sql/virtual-machines/windows/create-sql-vm-resource-manager-template?tabs=CLI
- You can also use Azure Cloud Shell
    - Using PowerShell
        - https://docs.microsoft.com/en-us/azure/azure-sql/database/single-database-create-quickstart?tabs=azure-powershell
        - https://docs.microsoft.com/en-us/azure/azure-sql/managed-instance/scripts/create-configure-managed-instance-powershell
    - Or CLI (command line interface)

        - https://docs.microsoft.com/en-us/azure/azure-sql/database/single-database-create-quickstart?tabs=azure-cli

- To deploy a database, you can use:
    - If you are using an Azure Pipeline, you can use a DACPAC (data-tier application portable artifact)
        - This gets added to your azure-pipelines.yml (yml stands for "Yet Another Markup Language").
    - SQL scripts, together with PowerShell.

81. automate backups

- For VMs, you can create automated backups.
    - This is done through the installation of the Sql Server IaaS Agent Extension to enable automated backups (this can be done through the "Create a Virtual Machine" process).
    - Needs to be:
        - Windows Server 2012 and SQL Server 2014 Standard/Enterprise (for Automated Backup version 1), or
        - Windows Server 2012 R2 or higher, and SQL Server 2016 or higher Standard/Enterprise/Developer (for Automated Backup version 2).
    - You can specify:
        - Retention period – up to 30 days.
        - Storage account.
        - Encryption (with password).
        - Whether system databases (Master, Model and msdb) are backed-up.
        - Whether you configure a manual or automated backup schedule. Automated backs-up depends on log growth. If manual, you specify:
            - Frequency – Weekly or Daily. If weekly, it will back up each database once a week, even if it needs to span over several days to do so.
            - Backup start time,
            - Backup time window (hours), and
            - Log backup frequency (minutes).
    - You must use the Full recovery model.
    - You can back up the default instance or a single named instance. If there is no default instance and multiple named instances, it will fail.

82. automate performance tuning and patching

- For PaaS (Azure SQL Database and MI), patching happens automatically.
    - You have no control over when it happens, but it has minimal impact if you use "retry logic".

  - If you have a database quorum, there should be at least one primary replica online.
  - Business Critical and Premium databases should also have at least one secondary replica online.
- On VMs, you could use "Automated Patching" – see topic 3.

83. implement policies by using automated evaluation modes

- This is not relevant to Azure SQL Database or MI.
- To create a new policy:
  - Go to Management – Policy Management – Policies.
  - Right-hand click and go to "New Policy".
  - Enter a name for the policy.
  - If you are intending to have this run to a schedule, click "Enabled" if you want the schedule to be enabled.
  - Next to "Check condition", click on "New condition".
  - In this new box, enter a name, a facet, and what you are checking (at least one field, an operator and a value).
    - Note - != means <>.
    - These conditions are what SHOULD be – the policy will fail if this is NOT the case.
  - In the Against targets, select target types. If this is blank, then it will be targeted against the server.
  - In the Evaluation Mode, select:
    - "On demand",
    - "On change: prevent" – uses DDL triggers
    - "On change: log" – logs to event notification.
    - "On schedule" – select/pick an existing or create a new schedule.
- Once created, if you want it to be run, right-hand click on it and go to Evaluate.
- To edit it, right-hand click and go to Properties.

Plan and Implement a High Availability and Disaster Recovery Environment

84. recommend HADR strategy based on RPO/RTO requirements

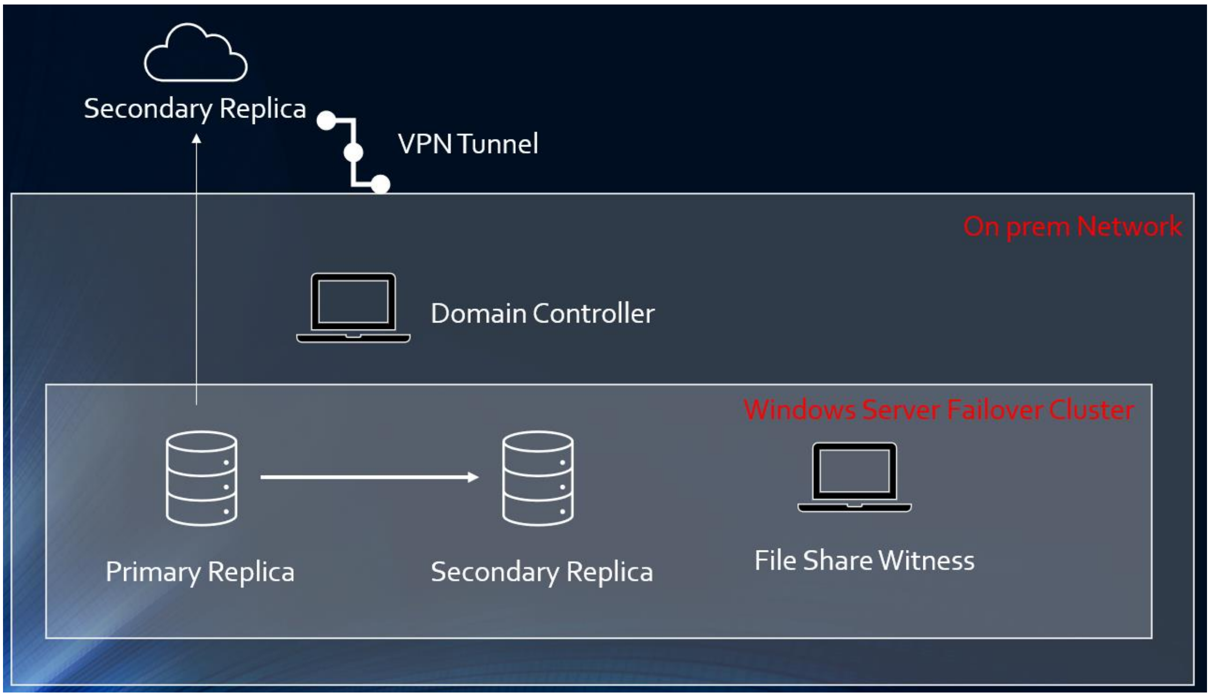
- Azure SQL Database offers a Service Level Agreement (SLA) of:
  - If using Hyperscale tier:
    - 99.9% for a zero replicas (8 hours 45 minutes over a year, or 43 minutes 48 seconds over a month),

        - 99.95% for one replica (4 hours 22 minutes over a year, or 21 minutes 54 seconds over a month).
    - Other Azure Database tiers:
        - 99.99% (52 minutes over a year, or 4 minutes 23 seconds over a month) – this is for other Azure SQL Database tiers and Azure SQL Managed Instance.
        - However, if you are in Business Critical/Premium tiers, and you have Zone Redundant Deployments, this increases to 99.995% (26 minutes over a year, or 2 minutes 11 seconds over a month).
            - A Database that includes multiple synchronized replicas provisioned in different Availability Zones
- For VMs:
    - The maximum SLA is 99.99% for the VM itself.
    - However, the SQL Server may fail, even though the VM is healthy – so the actual SLA will lower.
- Terminology:
    - RPO – Recovery Point Objective of 5 seconds (how much data you can lose)
    - RTO – Recovery Time Objective of 30 seconds (how long until you can use it again – maximum "Failover" time)
        - If exceeded, you get a credit of 100% of the total monthly cost of the Secondary
- If you have geo-replication, then you have a guarantee of:
    - Geo-restore for geo-replicated backups
        - RPO – 1 hour, RTO – 12 hours
    - Auto-failover groups
        - RPO – 5 seconds, RTO – 1 hour
    - Manual database failover (to geo-replicated secondary)
        - RPO – 5 seconds, RTO – 30 seconds

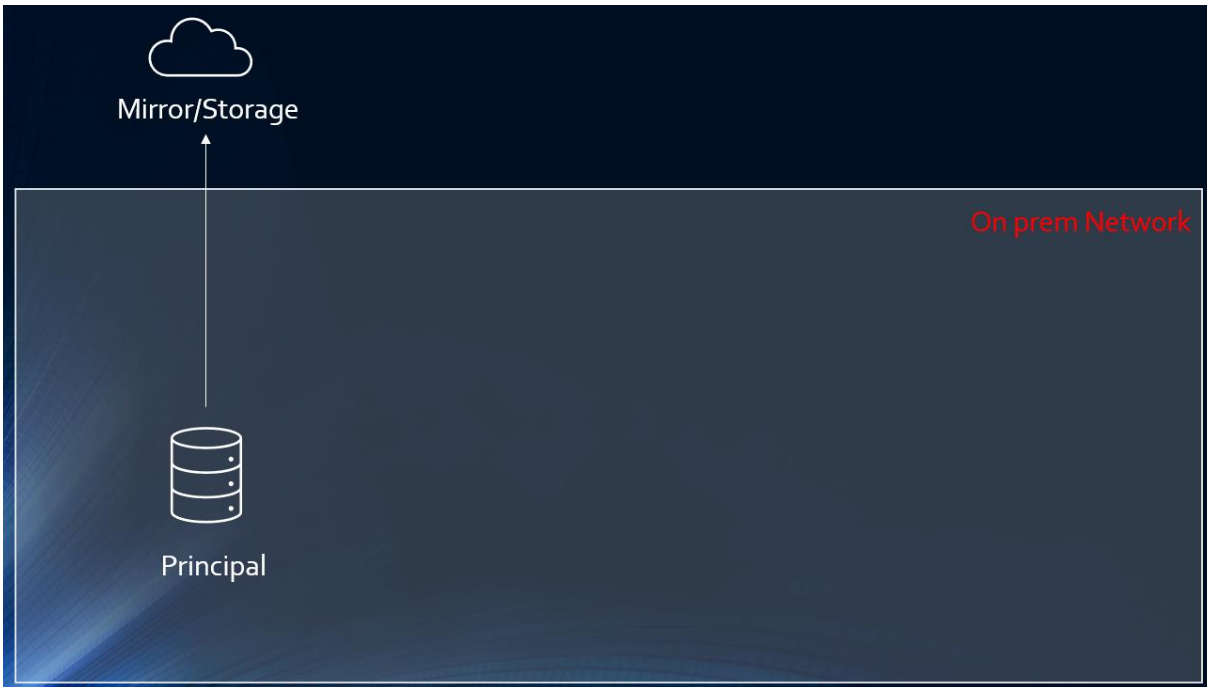
85. evaluate HADR for hybrid deployments

- Availability groups
    - 2-9 SQL Server instances on VMs or VMs and on-premises data center.
    - Data is committed on a primary, then sent out to secondaries.
    - You can use synchronous commit for secondary replica in on-prem network.
        - Transactions are not committed on the primary until they can be committed on the secondary.
        - However, asynchronous is also available, for lowest patency or for geographically spread secondaries.

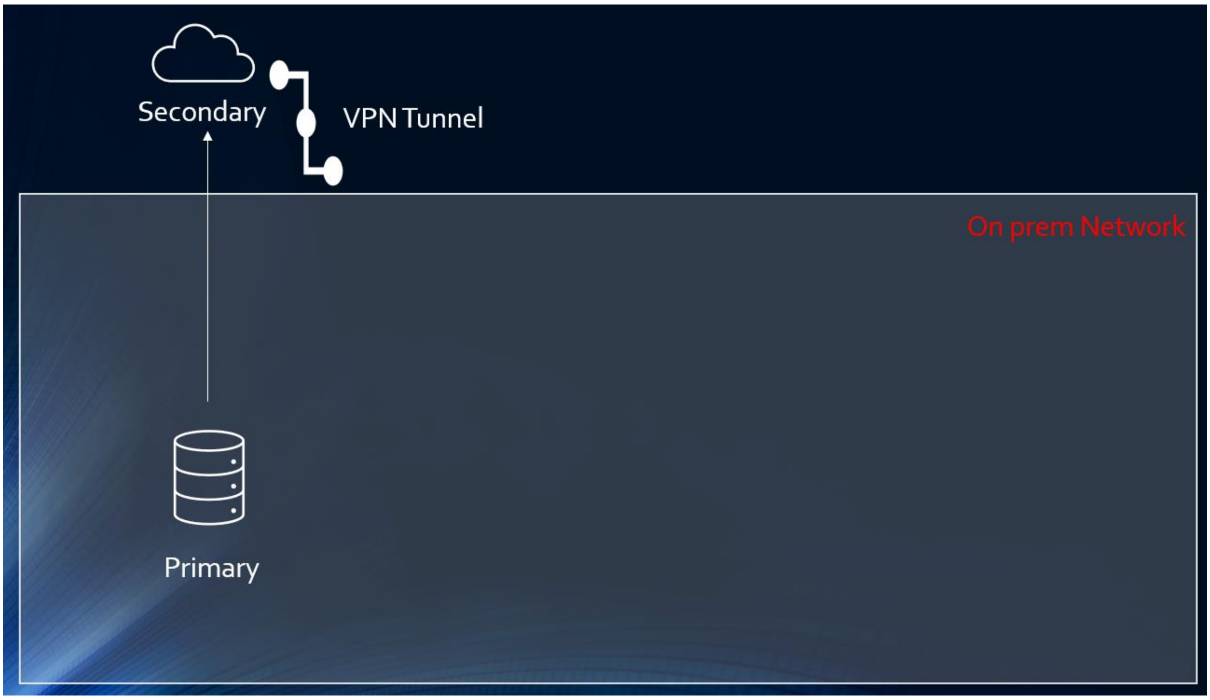
  - You need a domain controller VM, as it requires an Active Directory domain.
  - Availability replicas running Azure VMs allow for DR. It uses an asynchronous commit.
    - Transactions are committed on the primary before being replicated on the secondary.
  - You also need a VPN connection for the entire failover cluster, using a multi-subnet failover cluster.
  - For DR purposes, you also need a replica domain controller at the disaster recovery site.
  - You fail over at the database level, not the instance.
- Database mirroring
  - An Azure VM running at least SQL Server 2012, and another SQL Server running on-prem running at least SQL Server 2008R2, using server certificates.
  - No VPN required, and they don't have to be in the same Active Directory domain (but you can – but you will ned a VPN and a replica domain controller).
- Backup and restore using Azure Blog storage
- Replicate and fail over SQL Server to Azure with Azure Storage.
- Log shipping
  - An Azure VM and a SQL Server on-prem.
  - As log shipping requires Windows file sharing, you would need a VPN tunnel.
  - The secondary database is stored on a secondary server or warm standby.
- These are also available for Azure VMs only configurations.
- Evaluate HADR for hybrid deployments
  Availability groups



- Evaluate HADR for hybrid deployments
  Database mirroring / Azure Blob Storage / Azure Site Recovery

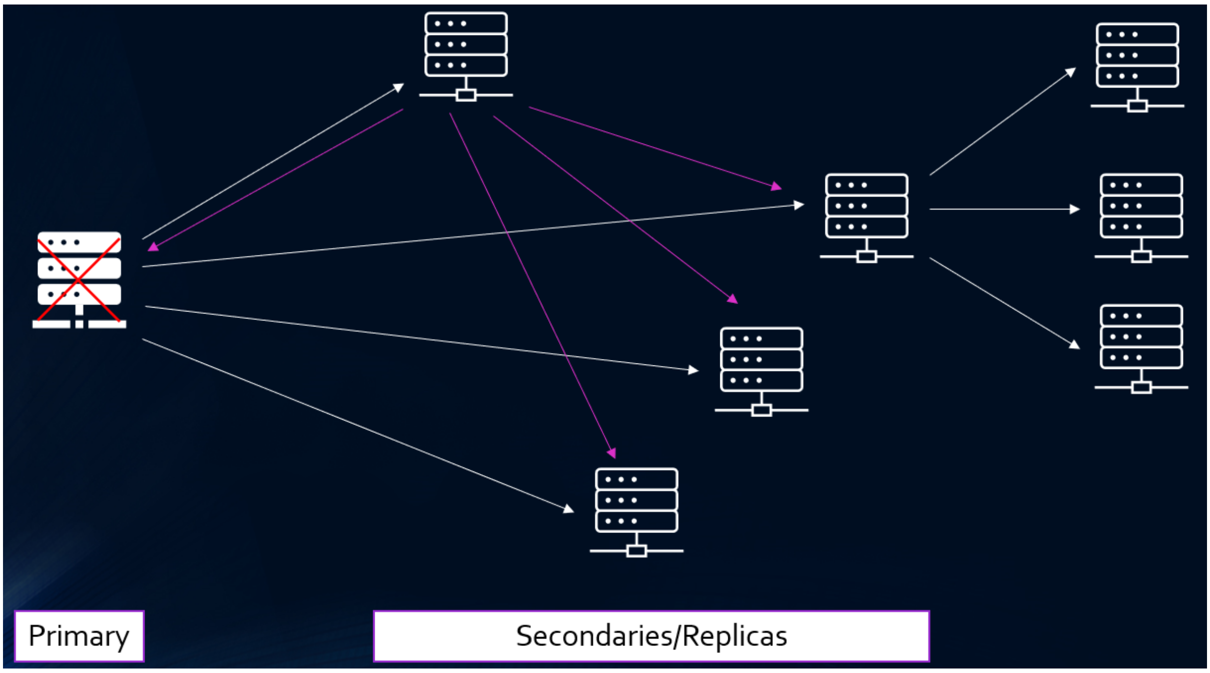


- Evaluate HADR for hybrid deployments
  Log shipping

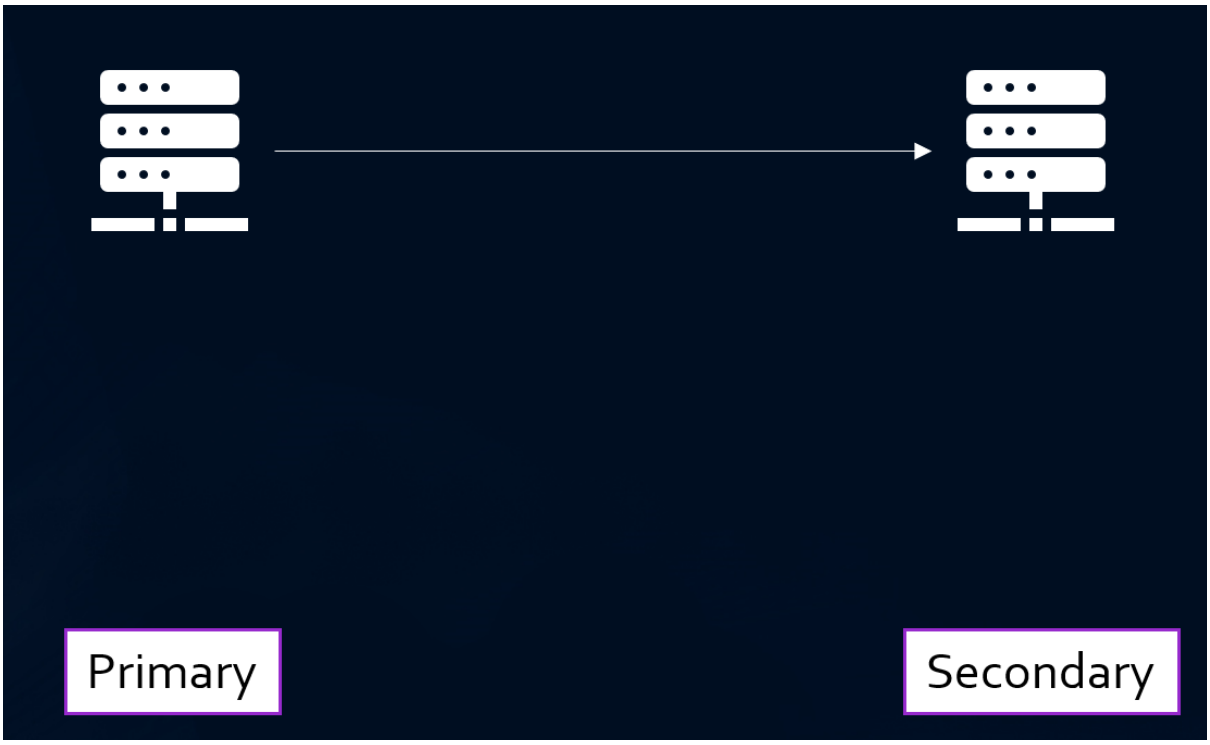


- Failover Cluster Instances (FCI) for Azure VMs
  - Allows for HA, but not DR.
    - Designed to protect against network card or disk failure – but there are other solutions in Azure.
  - There are 5 different configurations:
    - Azure shared disks for Windows Server 2019.
      - Attach a managed disk to multiple VMs at the same time.
    - Storage Spaces Direct (S2S) for a Storage Area Network for Windows Server 2016 or later.
    - Premium File Share for Windows Server 2012 or later. Uses SSD, have low latency, supported for Failover Cluster Instances.
    - Using third-party solutions.
    - Using Azure ExpressRoute.
- For Azure SQL Database and MI:
  - They use locally redundant availability as standard.
- For MI:
  - You can also configure a Failover Group
- For Azure SQL Database:
  - For more, see topic 94.
  - Automatic Asynchronous Replication
    - Secondary Database populated with primary database ("seeding").

        - Updates are then replicated automatically.
    - Readable secondary databases (replication).
        - To create a replica, go to Data Management – Replicas – Create replica.
        - You can have up to 4 per primary database.
            - Can be in different regions.
            - Want more? You can have a secondary of a secondary. Replication to that secondary takes longer.
        - They need to have at least the same service tier as the primary.
            - Need to change? When upgrading, upgrade the secondary first. When downgrade, downgrade the primary first.
            - You don't need to disconnect the secondaries unless you change between General Purpose and Business Critical.
        - More than 1 secondary means that, even if one fails, there will still be at least one until it is recreated.
        - Uses snapshot isolation mode, so updates from the primary are not delayed by long-running queries on the secondary.
- Azure Site Recovery service
    - Simple DR (but not HA) of Azure VMs from a primary to a secondary region.
    - Can also replicate on-prem VMs/servers to Azure or a secondary on-prem datacenter.
    - Not SQL Server solution, but can be used with SQL Server on the VMs.
    - Provides continuous replication.
    - Can replicate using recovery point snapshots; they capture disk data, data in memory, and transactions in process.
    - Run DR drills, planned failovers with zero-data loss, or unplanned failovers.
    - Useful to protect against ransomware.
- identify resources for HADR solutions – Replicas/ geo-replication



- identify resources for HADR solutions – failover groups



	Geo-replication	Failover groups
Automatic failover	x	Yes
Fail over multiple databases simultaneously	x	Yes
SQL Managed Instance support	x	Yes
User must update connection string after failover (This will take time)	Yes	x
Can be in same region as primary	Yes	x
Multiple replicas	Yes	x
Supports read-scale	Yes	Yes

88. test HA by using failover – FAILOVER GROUP

89. test DR by using failover or restore

- To create a failover group
  - Go to the SQL Database server (not the database).
  - Go to Settings – Failover groups – Add groups.
  - Create a failover group.
    - Enter a unique failover group name,
    - a server (or create a new server),
    - The Read/Write failover policy (automatic or manual), and
    - the Read/Write grace period (1-24 hours).
  - You can then add eligible databases into the failover group.
- To update the failover group settings:
  - Click on Settings – Failover groups – and the name of the failover group.
  - You can "edit the configuration" (read/write failover policy and grace period).
  - You can "add databases" into the failover group.
  - Once you have done all of your changes, click "Save" or "Discard".
- To test the failover:
  - Click on Settings – Failover groups – and the name of the failover group.
  - Click on Failover (or Forced [manual] failover).
    - Forced failover risks possible data loss.
  - All databases within a failover group are then fail-overed.
  - You can also use the PowerShell cmdlet:
    - Invoke-AzSqlInstanceFailover -ResourceGroupName "ResourceGroup01" -Name "ManagedInstance01"

        - This failovers an Azure SQL Managed Instance.
      - Invoke-AzSqlDatabaseFailover -ResourceGroupName "ResourceGroup01" -ServerName "Server01" -DatabaseName "Database01"
        - This failovers a single database in an Azure SQL Database, which could be a single database in an elastic pool (without affecting the other databases in the same elastic pool).
      - Invoke-AzSqlElasticPoolFailover -ResourceGroupName "ResourceGroup01" -ServerName "Server01" -ElasticPoolName "ElasticPool01"
        - This failovers all databases in an elastic pool.

88. test HA by using failover – AVAILABILITY GROUP

89. test DR by using failover or restore

- To manually fail over an availability group:
  - Go to SSMS and the server which hosts a SECONDARY replica of the availability group.
  - Go to AlwaysOn High Availability – Availability Groups.
  - Right-hand click the availability group to be failed over, and click on "Failover".
  - If the Introduction page of the wizard says "Perform a planned failover for this availability group", then you can do this without data loss.
  - In the "Select New Primary Replica" page, you can view the status of:
    - The primary replica,
    - The Windows Server Failover Cluster quorum status:
      - Normal quorum
      - Forced quorum
      - Unknown quorum, and
      - Not applicable.
  - The secondary replicas can say:
    - "No data loss" – this is a planned manual failover,
    - "Data loss, Warnings (X)", where X shows the number of warnings – this would have to be a forced failover.
  - The relevant secondary replica will then become the new primary replica.
  - In the "Connect to Replica" page, you can connect to the failover target.
- In T-SQL, you can use:
  - ALTER AVAILABILITY GROUP MyAg FAILOVER;

90. perform a database backup with options

- Performing a database backup with options is primarily for VMs.
  - MIs can do some options, but it will be a Copy Only backup.

- The General page shows:
    - Recovery model.
    - Backup type (Full, Differential or Transaction Log [not for Simple]),
    - Copy-only backup (this is independent of the sequence of other backups).
    - Backup component:
        - Database (all the database), or
        - Specific files and filegroups.
    - Back up to:
        - Disk (System file or disk-based logical backup device),
        - Tape (local or tape-based logical backup device) – this is deprecated.
        - URL – Microsoft Azure Blob storage
        - Contents shows the media contents for the selected disk/tape (not URL).
- The Media Options page shows:
    - Backup to the existing media set
        - Append to the existing backup set, preserving any prior backups
        - Overwrite all existing backup set, replacing prior backups with the current backup.
        - Check media set name and backup set expiration – requires the backup operation to verify name and expiration date.
            - Optionally, set the media set name.
    - Backup to a new media set, and erase all existing backup sets.
        - Optionally, set the new media set name and description.
    - Reliability
        - Verify backup when finished.
        - Perform checksum before writing to media
        - Continue on error (even after encountering one or more errors)
    - Transaction log
        - Backup the transaction log and truncate it to free log space. The database remains online.
        - Backup the transaction log tail (tail-log backup), and leave the database in a restoring state (not available to users until it is completely restored).
    - Tape drive (deprecated)
        - Unload the tape after backup, and
        - Release and rewind the tape before unloading.
- The Backup set page shows:

    - Name and description of the backup set name
    - Expiration duration or date:
        - After X days; if X is zero, the backup set will never expire.
        - Specific date.
    - For the Enterprise edition, you can select backup compression.
    - Encrypt backup, using AES 128, AES 192, AES 256 and Triple DES.
        - Only enabled if you append to existing backup set. Backup your certificate or keys to a different location.
- If you have a VM with IaaS Extension can configure backups in the Azure Portal.
- You can:
    - Enable/Disable the backups of system databases, and
        - Configure a backup schedule – Automated or Manual,
        - Backup frequency – Daily or Weekly,
        - Backup start time (local VM time),
        - Full backup time window (hours),
        - Log backup frequency (minutes).
- Restores need to be configured from within SQL Server.

91. perform a database restore with options

- The user must have CREATE DATABASE permissions.
    - This exists in sysadmin and dbcreator fixed server roles, and dbo (owner) for existing databases.
- The General page shows:
    - Source
        - Database – this list only contains databases backed up, based on the msdb backup history.
        - Device – tape, URL or file. This is required if the backup was taken on a different SQL Server instance.
            - You can select up to 64 devices that belong to a single media set.
        - Device database – backups on the selected device.
    - Destination
        - Database to restore.
            - Enter a new database, or choose an existing database from the dropdown list, which includes all databases on the server, excluding master and tempdb.

        - Restore to.
            - It is "To the last backup taken" by default.
            - Alternatively, you can select [Backup] Timeline, which shows the database backup history as a timeline.
    - Restore plan
        - Backup sets to restore.
            - The default is the recovery plan suggested to achieve the goal.
    - Verify backup media.
        - Check the integrity of the backup files prior to restoring them.
- The Files page shows:
    - Relocate all files to [a particular] folder, showing
        - Logical and Original File Name,
        - File Type,
        - The file path/name to "Restore As".
- The Options page shows:
    - Overwrite the existing database
        - Overwrite database files.
    - Preserve the replication settings
        - Only relevant if a database was replicated when the backup was created, and when restoring a published database to a different server (other than the creation server).
    - Restrict access to the restored database.
        - Only for db_owner, dbcreator and sysadmin members.
    - Recovery state:
        - Restore with recovery. Default option.
            - Only choose this option in a full or bulk-logged recovery model if you are also restoring all log files at the same time.
        - Restore with NoRecovery
            - Left in the Restoring state. Allows for additional backups.
        - Restore with Standby
            - Limited read-only access.
            - Need to specify a Standby file, which allows the recovery effects to be undone.
    - Tail-log backup.
        - Take tail-log backup before restoring.

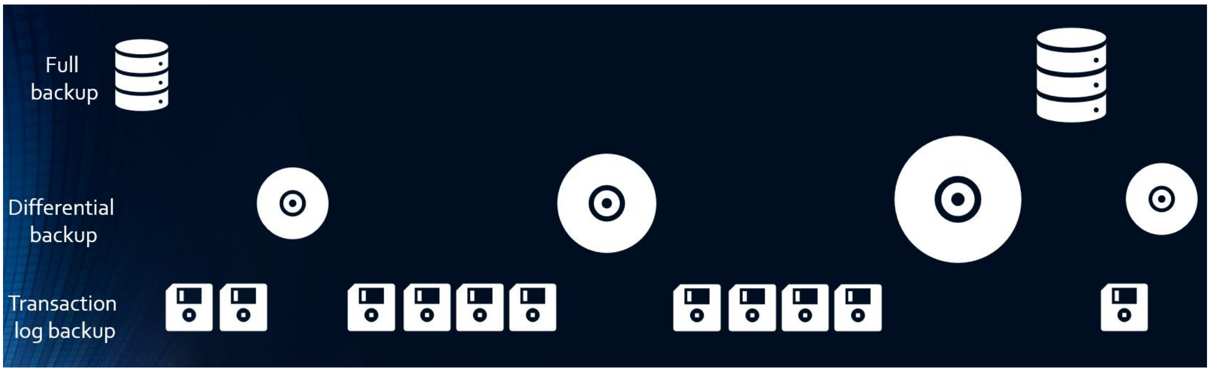
      - You can specify a backup file for the tail-log.
  - Server connections
    - Restore options may fail if there are active connections to the database.
  - Prompt before restoring each backup.
    - The "Continue with Restore" dialog box will be displayed after each backup is restored.
    - If you click "No", the database will be left in the Restoring state.

92. perform a database restore to a point in time

- For VM, see topic 91.
- For Azure SQL MI, to restore an Azure SQL database to a different region:
  - Go to the MI, click on "+New database", select the database name, and change "Use existing data" to "Backup" and select the backup.
- Database backups for Azure SQL Database and Azure SQL MI are done automatically.
  - Full backups every week,
  - differential backups every 12-24 hours, and
  - transaction log backups every 5-10 minutes.
  - You can do a:
    - point-in-time restore (PITR) of existing or deleted databases – 7 days by default
      - You can change it to 1-35 days optionally (apart from Hyperscale and Basic tier databases – basic has a maximum of 7 days).
      - Note: In MI, PITR is available for individual databases, but not for the entire instance.
    - recover to another geographic region - same
    - restore from a long-term backup – up to 10 years.
  - The first backup is scheduled immediately after a new database is created or restored.
  - For non-large database, it usually completes within 30 minutes.
- To restore a database:
  - Go to the Database overview page.
    - If recovering a deleted database, go to the server or MI, and click "Deleted database" (on the left-hand side).
  - Click "Restore".
  - Choose backup source.
  - Select the Point-in-time backup point, and then OK.
  - You cannot restore over an existing database (but you can rename it afterwards).

  - You can use PowerShell cmdlets to restore an existing database, but again, you can't restore over an existing database.

- To restore an Azure SQL database to a different region:
  - Create the database.
  - In the "Additional settings" tab, change "Use existing data" to "Backup", and select a backup.

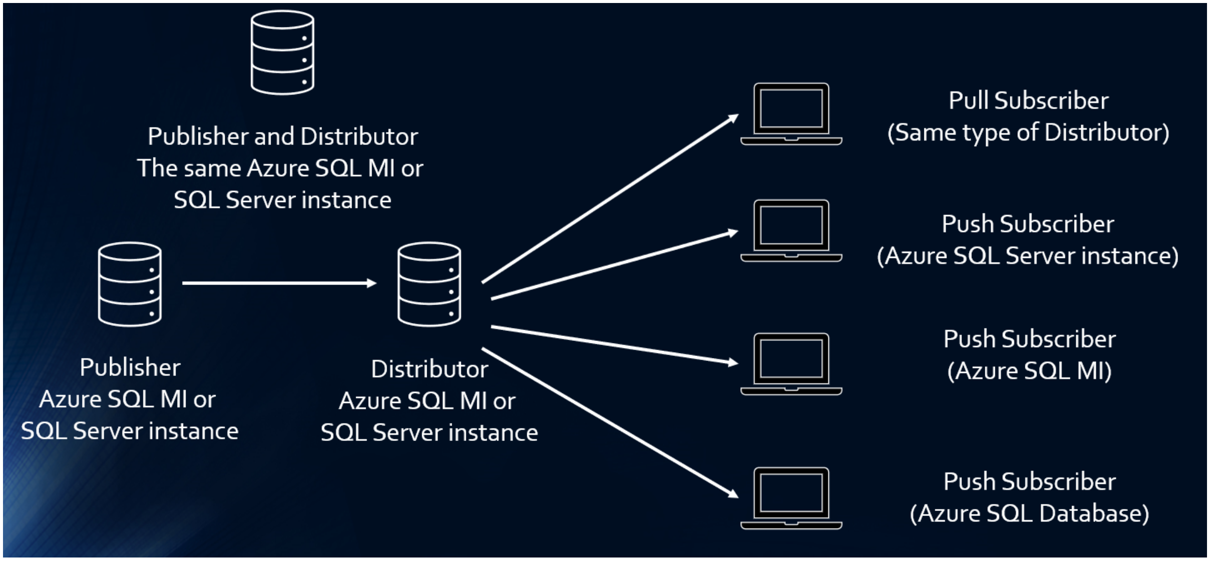


93. configure long-term backup retention

- This is for both Azure SQL Database and Azure SQL MI.
  - It is in Public Preview in SQL MI in Azure Public regions only.
- Backups can also be configured for Long-Term Retention (LTR).
- LTR backups are done by Azure.
  - You cannot control the timing nor manually create a LTR backup.
  - It may take up to 7 days before the first LTR backup will be shown in the list of available backups.
  - Ensure that you have a LTR policy on secondary databases, only to be created when they become primary.
  - Backups are stored in Azure Blob storage – a different storage container weekly.
- To configure this, go to Azure portal – the server – Backups – Retention policies – select the database(s), and configure the LTR:
  - Weekly backups,
  - Monthly backups,
  - Yearly backups and
  - WeekOfYear backups.
- To view backups, go to Azure portal – the server – Backups – Available backups – and next to the relevant database, under “Available LTR backups”, select Manage.
  - You can click on an LTR backup, and select Restore (which creates a new database) or Delete.
- Once the original database is deleted…
  - No more backups are made.

- As backups expire, they are deleted.

94. configure replication



- You can use transactional replication to push changes made in an Azure MI to:
    - A SQL Server database (on prem or Azure VM),
    - An Azure SQL Database, or
    - An instance database in Azure SQL MI
- Useful for:
    - Distributing changes to one or more databases in SQL Server, Azure SQL MI or Azure SQL Database.
    - Keep several distributed databases synchronized.
    - Migrate databases from SQL Server/Azure SQL MI to another database by continuously published the changes.
- The components of a transactional replication are:
    - Publisher
        - Publishes changes made on some tables ("articles"), and send the updates to the Distributor.
        - Can be Azure SQL MI or an SQL Server instance.
        - Cannot be Azure SQL Database (need to use Data Sync – topic 14 – for this).
    - Distributor
        - Collects changes from Publisher and distributes them to Subscribers.
        - Can be Azure SQL MI or an SQL Server instance.
            - Can be the same Azure SQL MI as the Publisher, but a different database.
        - If SQL Server instance, version needs to be the same or higher than the Publisher version.

- Pull subscriber
  - Can be Azure SQL MI or an SQL Server instance, but needs to the same type as the Distributor.
- Push subscriber.
  - Can be an Azure SQL Database.
    - However, it only supports Standard Transactional and Snapshot.
  - Can be an Azure SQL MI.
    - It supports Standard Transactional, Snapshot and Bidirectional.
  - Can be an SQL Server instance.
    - Needs to be more recent than the Publisher, or no more than 2 versions earlier.

- Create a Publication:
  - In SSMS, go to the Server – Replication, and right-hand click "Local Publications".
  - Click "New Publication".
  - Specify a Distributor.
    - If you don't, the Publisher will act as its own Distributor.
      - You will need to specify a default snapshot folder, a directory that agents can read from and write to this folder.
    - If you do, you will need to enter the Publisher password.
  - Choose a publication database.
  - Select a publication type.
    - Transaction replication – changes occur in near real time, applied to the Subscribe in the same order as they occurred on the publisher.
      - For Azure SQL Database, MI, on-prem VM
    - Merge replication – Data can be changed on both the Publisher and Subscriber.
      - When connected to the network, all rows which have changed between Publisher and Subscriber are synchronised. For on-prem, VM and MI
    - Snapshot replication – distributes data at a specific moment of time, and does not monitor for updates to the data.
      - For Azure SQL Database, MI, on-prem VM
    - Peer-to-peer – allows for changes in near real-time on multiple server instances.
      - For on-prem and VM
    - Bidirectional – allows two servers to exchange changes with each other.
      - For on-prem, VM and MI

        - Updatable subscriptions (deprecated) – when data updated at a Subscriber, it is propagated to the Publisher and then to the other Subscribers.
    - Select data, database objects and filter columns and rows from table articles to publish.
    - Set the Snapshot Agent schedule.
    - Specify the credentials for:
        - Snapshot Agent for all publications.
        - Log Reader Agent for all transactional publications.
        - Queue Reader Agent for transactional publications that allow updating subscriptions.
    - Optionally, script the publication.
    - Specify a publication name.
- How to create a Push subscription:
    - In SSMS, go to the Server – Replication, and right-hand click "Local Subscriptions".
    - Click "New Subscriptions".
    - Select Publisher and publication.
    - Select where replication agents will run.
        - Run all agents at the Distributor (push subscriptions) or
        - Merge Agent Location
    - Select subscribers and subscription databases.
    - Enter the logins and passwords for connections made by replication agents.
    - Select a synchronization schedule and when the Subscriber should be initialized.
    - Select additional options for merge or transactional publications.
- You can create readable secondary databases in the same or different region.
    - If in the same region, not as good for DR.
    - This is for Azure SQL Database, not for Azure SQL MI
        - Azure SQL Database and Azure SQL MI can both use auto-failover groups.
    - Up to 4 secondaries are supported in the same or different regions.
        - They can be part of an elastic pool.
    - They can be used for read-only access queries.
    - It replicates changes by streaming database transaction logs (unlike transactional replication, which replicates changes by executing DML commands, such as INSERT/UPDATE/DELETE).
    - You can failover to secondary databases.
- You can use it for:
    - Database migration from one server to another with minimum downtime, and

  - Creating an extra secondary as a fail back copy during application upgrades.
  - It uses asynchronous replication, so the transactions are committed on the primary before being replicated.
  - Planned and unplanned failover.
- To configure geo-replication:
  - Go to the Azure Portal – the database – Data management – Replicas

95. create an Availability Group

- You need:
  - A resource group with a domain controller.
  - One or more domain-joined VMs in Azure running SQL Server 2012+ Enterprise, or SQL Server 2016+ Standard in:
    - The same availability set (different computers in the same datacenter), or
    - Different availability zones (physical datacenters).
    - They need to be registered with the SQL IaaS Agent extension in full manageability mode and are using the same domain account for the SQL Server service on each VM.
  - Two available (not used by any entity) IP addresses.
    - One for the internal load balancer.
    - One for the availability group listener within the same subnet as the availability group.
- An availability group supports:
  - A set of primary replica (which host the primary databases), and
  - 1-8 sets of secondary replicas (only 1 allowed in SQL server Standard), each of which hosts the secondary databases (this does not replace backups).
  - There must be at least 2+ failover partners.
- Note:
  - The primary replica send transaction log records to every secondary database ("data synchronization").
  - Individual primary/secondary databases can be suspended or fail without affecting other primary/secondary databases.
  - You can configure 1+ secondary replicas to support read-only access to secondary databases, and/or to permit backups on secondary databases.
- Availability modes are:
  - Asynchronous-commit mode. The primary replica commits without acknowledgement that the secondary replica has committed.

        - Minimizes transaction latency, but there is a lag for when the data is committed onto the secondaries.
    - Synchronous-commit mode. The primary replica does not commit until the secondary replica has hardened the log.
- Failover:
    - This is when the target secondary replica transitions to being the new primary replica.
    - The former primary databases becomes secondary databases.
    - The failovers are as follows:
        - Planned manual failover – no data loss – secondary replica needs to be synchronized – for synchronous-commit mode only.
        - Automatic failover – no data loss – occurs when there is a failure to the primary replica – for synchronous-commit mode only. Needs to have a Windows Server Failover Cluster quorum and be synchronized.
    - Forced manual failover (also known as "forced failover"). For asynchronous-commit mode. This is a DR option.
        - The only type of failover that is possible if the target secondary replica is not synchronized with the primary replica.
    - After failover, Azure SQL connections are automatically redirected to the new primary node.
- To create a new cluster in the Portal:
    - In Azure portal, go to the VM – Settings – High Availability.
    - Click on "+ New Windows Server Failover Cluster".
    - Name the cluster, and give a Storage Account which is the Cloud Witness.
        - Storage Account name: 3-24 characters using numbers and lower-case letters.
    - In "Windows Server Failover Cluster", provide credentials for:
        - The SQL Server service account,
        - The cluster operator, and
        - The Bootstrap account.
    - Select the VMs to be added into the cluster.
        - A restart may be required.
        - Only relevant VMs will be shown.
    - Click Apply.
- To create an availability group:
    - In Azure portal, go to the VM – Settings – SQL Server configuration – Open - High Availability.
    - Click on "+ New Always On availability group".
    - Name the availability group.

    - Click "Configure listener".
        - Use an existing load balancer, or
        - Click "Create new".
        - For the Load balancer:
            - Type: "Internal" allows apps in the same Virtual Network to connect to the availability group.
            - Type: "External" allows apps to connect to the availability group through a public internet connection.
            - "IP address assignment" should be Static.
            - The "Resource group" and "Location" should be that where the SQL Server instances are in.
        - For the Listener:
            - The Listener name should not exceed 15 characters.
            - The Listener Port defaults to 1433.
            - The Probe Port is for the internal load balancer, which is 59999 by default.
            - DHCP (Dynamic Host Configuration Protocol) is not recommended in a production environment.
    - Click "Apply".
    - Click "+Select replica".
    - Select the VMs to be added into the availability group.
    - Click "Apply".
- In the Azure Portal – Settings – High Availability, the status of the availability group(s) are shown.
- However, you can also use the SQL Server to do this as well – and this is the way I do this in the videos to this course.

96. configure auto-failover groups

- Used in both Azure SQL Database and MI (but not VM).
- Auto-failover will not happen for at least 1 hour.
    - It may be only a minor outage.
    - When it happens, failover itself takes a few seconds.
- Use auto-failover groups when:
    - It is mission critical,
    - Your SLA does not allow for 12+ hours of downtime
        - Geo-restore has a Recovery Time Objective of 12 hours.
    - You don’t want to lose up to 1 hour of data (as per Geo-restore).

        - Auto-failover groups has a Recovery Point Objective (RPO) of 5 seconds.
    - Geo-replication is more costly than needed for the requirements.

97. integrate a database into an Always On Availability Group

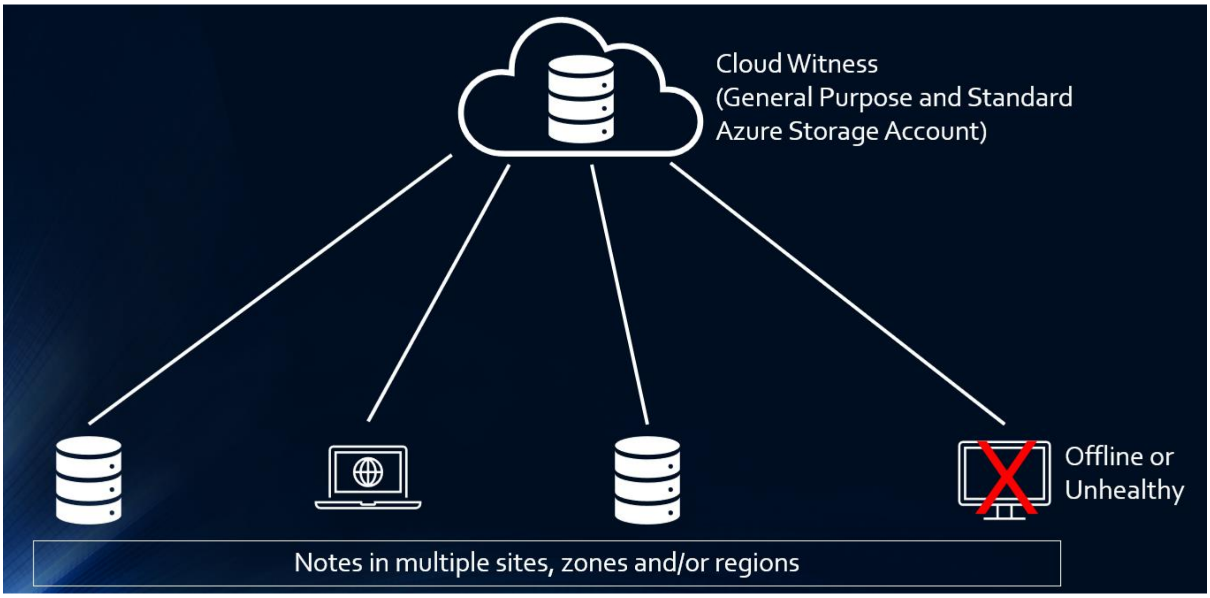
- The database must exist on the server instance that hosts the primary replica.
    - The primary database should use the full recovery model.
    - The name of the secondary database must be the same as the primary database.
    - Secondary databases do not exist until backups of the new primary database are restored to the secondary replicas (use RESTORE WITH NORECOVERY).
- In SSMS
    - Connect to one of your SQL Server VMs using (for example) RDP.
    - In SSMS, go to your SQL Server instance – Always On High Availability – Availability Groups.
    - Right-hand click on your availability group and select "Add Database…".
    - Add the database(s) to your availability group.
    - Click "OK".
- In T-SQL (see topic 108)
    - ALTER AVAILABILITY GROUP MyAG ADD DATABASE MyDb3;
    - GO
    - ALTER DATABASE Db1 SET HADR AVAILABILITY GROUP = MyAG;

98. configure quorum options for a Windows Server Failover Cluster

- Always On availability groups and Failover Cluster Instances rely on the underlying Windows Server Failover Clustering (WSFC) service.
- It monitors network connections and the health of the nodes (clustered servers)
- A two-node cluster will function without a quorum resource, but its use is recommended.
    - This then provides an odd number of votes, and a 3 quorum votes minimum.
- To configure:
    - Open Failover Cluster Manager.
    - Right-hand click the cluster, and go to More Actions – Configure Cluster Quorum Settings.
    - In the wizard, select "Select the quorum witness".
    - Select the type of witness.
- There are 3 options:
    - Cloud Witness
        - Needs Windows Server 2016+
        - Uses Microsoft Azure to provide a vote on cluster quorum.

        - Ideal for deployments in multiple sites, zones and/or regions.
        - Only 1Mb.
        - Recommended to use whenever possible, unless you have a failover cluster solution with shared storage.
        - Use General Purpose and Standard Storage (Blob storage and Premium storage are not supported).
        - Use "Locally redundant storage" for Replication type.
        - Uses port 443 (HTTPS) for communication.
        - You will need:
            - The Azure Storage Account Name,
            - Primary Access Key corresponding to the Storage Account, and
            - The endpoint server name, if you are using a different Azure service endpoint, such as Microsoft Azure in China.
        - Once finished, you can see this witness in the Failover Cluster Manager snap-in.
    - Disk Witness.
        - A small clustered disk in the Cluster Available Storage group.
        - The disk is highly available (most resilient) and can fail over between nodes.
        - Less than 1Gb.
        - Only can be used with a cluster which uses Azure Shared Disks.
        - Cannot be a Cluster Shared Volume.
    - File share witness
        - Configured on a file server running Windows Server.
        - A file share on a separate VM in the same Virtual Network.
        - Needs to be separate from the cluster workload, to allow equal opportunity to other clusters.
        - Only use if you can't use the other 2 options.
- In "Advanced quorum configuration", you have these options:
    - Select Voting Configuration.
        - By default, all nodes have a vote, but you can assign votes to only some nodes.
        - You could also have "No nodes", which then the same as "No majority (disk witness only)" – see below.
- Your cluster will then be configured in:
    - Node majority (no witness).

    - The cluster quorum is the majority of voting nodes in the active cluster membership.
  - Node majority with witness ("Node and File Share Majority" or "Node and Disk Majority")
    - Nodes have a vote, and the witness also has a vote.
  - No majority (disk witness only).
    - Only the disk witness has a vote.
    - Not recommended because it is a single point of failure.


Cloud Witness
(General Purpose and Standard
Azure Storage Account)
Offline or
Unhealthy
Notes in multiple sites, zones and/or regions

99. configure an Always On Availability Group listener

- See topic 95.
- To configure it in SSMS,
  - Go to the server instances that hosts the primary replica.
  - Go to Always On High Availability – Availability Groups.
  - Right-hand click on "Availability group Listeners", and click on "Add Listener".
  - Enter the listener DNS name – in SSMS, that is up to 15 letters, numbers, hyphens and underscores.
  - The TCP port used by the listener.
  - Select the TCP Protocol used by the listener, either:
    - Dynamic Host Configuration Protocol (DHCP) – not recommended, or
    - Static IP.
      - You must specify a static IP address for every subnet that hosts an availability replica, including Subnet and IP Address.

100. Configure failover cluster instances on Azure VMs

VM for domain controller

- Create new resource – SQLHADR
- Virtual machine name – VMDOMAINCONTROLLER
- Image – Windows Server 2019 Datacenter – Gen1
- Select inbound ports – RDP and HTTP
- Virtual network – SQLHADR-vnet

VM for SQLSERVER1

- Virtual machine name: SQLSERVER1
- Resource group – SQLHADR
- Image – Windows Server 2019 Datacenter
- Virtual network – SQLHADR-vnet

VM for SQLSERVER2

- Virtual machine name: SQLSERVER2
- Resource group – SQLHADR
- Image – Windows Server 2019 Datacenter
- Virtual network – SQLHADR-vnet

Connect to VMDOMAINCONTROLLER

- Go to VM – VMDOMAINCONTROLLER – Connect
- Enter credentials.
- Do you want your computer to be discoverable by other PCs and devices on this network? Yes
- Server Manager – Manage – Add/Remove Roles and Features
- Server Selection – note IP Address (X.Y.0.4)
- Server Roles – Active Directory Domain Services
- Promote VM as a Domain controller.
- Add a new forest – FILECATS.CO.UK
- Enter the Directory Restore password.

Join VM1 to DC

- Go to SQLSERVER1 (then SQLSERVER2) – Networking – Network Interface hyperlink – DNS servers – Custom – enter the DNS server IP address – X.Y.0.4
- Start SQLSERVER1 and SQLSERVER2.
- Then connect to the computers.
- Do you want your computer to be discoverable by other PCs and devices on this network? Yes
- Change to the domain FILECATS.CO.UK – go to Windows Explorer - My PC

Install Windows Failover Cluster Role in SQLSERVER1 and SQLSERVER2

- In SQLSERVER1 and SQLSERVER2: Server Manager – Manage – Add/Remove Roles and Features
- Features – Failover Clustering
- After reboot, go to All Services, Right click on SQLSERVER1 and select "Failover Cluster Manager".
- (There are no clusters created).
- Actions – Create cluster.
- Add SQLSERVER1 and click Browse and add SQLSERVER2.
- Enter a cluster name: SQLCLUSTERNAME
- Create the cluster.
- Close the window. There is a new cluster.
- You can click on Nodes to see the 2 computers.

SQL Server management

- SQL Server Configuration Manager 2019
- Right-hand click on SQL Server (in SQL Server Services) – go to the "Always On Availability Groups" and check "Always on Availability Group".
- Restart SQL Server (in SQL Server Configuration Manager)
- Go to SQL Server Configuration Manager – SQL Server Network Configuration – Protocols – TCP/IP and Enable.
- Restart SQL Server (in SQL Server Configuration Manager)
- Go to Windows Defender Firewall – New Rule – Port – TCP 1433 (all others as default).
- Do the same in SQLSERVER2.
- Can you open SSMS in SQLSERVER1 and connect to SQLSERVER2 database?

Create Storage Account for cloud witness

- Portal – Storage Account
- New
- Storage account name: sqlstoragewitness.
- Select "Local redundancy option".
- Create review + create.
- Go into the Storage account – Access keys.

Configure Witness

- Go to Failover Cluster Manager – the actual failover cluster (in my case, SQLCLUSTER.filecats.co.uk)
- On the right-hand sideMore Actions – Configure Cluster Quorum Settings
- Click Next.
- Click "Select the quorum witness"

- Click “Configure a cloud witness”.
- In the Storage account:
- Copy storage account name and storage account key.
- Click Next x 3.
- The Cloud Witness is now in the “Cluster Core Resources”.

Prepare databases for Always On Availability Groups – SQLSERVER1 (97, 107)

- Create database with table with demo data.
- Go to Always On High Availability – right hand-click on Availability Groups – go to “New Availability Group Wizard”.
- Next
- Enter an availability group name: SQLAVAILABILITYGROUP
- Leave the cluster type as “Windows Server Failover Cluster”.
- Click next – find we need a full backup.
- Create the backup (right-hand click on the database – Tasks – Back Up…)
- Redo the “New Availability Group Wizard”.
- Select the databases.
- Select the replicas… Click “Add Replica” and log into SQLSERVER2.
- Look at availability mode, automatic failover, and readable secondaries. (Synchronous good if you have close physical distance.)
- Select Initial Data Synchronization – automatic seeding, full database and log backup, join only, or skip.
- Finish the wizard (It’s OK for the purposes of the DP-300 course if the listener configuration has a warning).

Add listener.

- In SSMS, go to Always On High Availability – Availability Groups – NameOfGroup – right-hand click on Availability Group Listener – Add a Listener.
- Give it a name -SQLAG
- Port – 1433
- Network Mode – Static – click Add.
- Enter an iPv4 Address – X.Y.0.20.

Test failover

- Go to Always On High Availability – Availability Group – SQLAVAILABILITYGROUP
- Check which is Primary and which is Secondary.
- “Start Failover Wizard”.

- Start in the Primary.
- Select the new Primary Replica (which is currently a secondary).
- Click “Connect to Replica”, and click “Connect” to enter credentials.
- Finish.
- Check which is Primary and which is Secondary.

Perform Administration by Using T-SQL

101. evaluate database health using DMVs

- See topic 61.
- To use database DMVs, you need to have VIEW DATABASE STATE permission on the database.
  - It is not enough to have VIEW SERVER STATE.
- Overall resource usage:
  - SELECT * from sys.dm_db_resource_stats –
    - CPU, IO and memory
    - SLO is the Service Level Objective, which includes deployment option, service tier, hard and compute amount.
    - You get a row for every 15 seconds for about the past hour.
    - Use sys.server_resource_stats for MI.
  - SELECT * from sys.dm_user_db_resource_governance
    - Storage in the current database or elastic pool.
    - In MI only, you can also use sys.dm_instance_resource_governance).
  - SELECT * FROM sys.dm_os_job_object
    - CPU, memory and I/O resource at the SQL Server level.
  - SELECT * FROM sys.dm_io_virtual_file_stats(null, null)
    - I/O statistics for data and log files
  - SELECT * FROM sys.dm_os_performance_counters
    - Performance counter information, including:
      - SQL Server: Databases
      - SQL Server: General Statistics
      - SQL Server: Query Store
      - SQL Server: SQL Statistics
- Waiting on resources:
  - SELECT * FROM sys.dm_os_wait_stats OR

- SELECT * FROM sys.dm_db_wait_stats (Azure SQL Database / MI)
    - Returns information about all the waits encountered by threads that executed.
    - Top wait types
        - Governor
            - LOG_RATE_GOVERNOR – waits for Azure SQL Database
            - POOL_LOG_RATE_GOVERNOR – Elastic Pools
            - INSTANCE_LOG_GOVERNOR – MI waits
            - RBIO* - Hyperscale log governance.
            - HADR_THROTTLE_LOG_RATE – Business Critical and geo-replication latency
        - IO
            - PAGEIOLAATCH_* - data file I/O issues
            - PAGELATCH_* - tempdb I/O issues
            - WRITE_LOG – transaction log I/O issues
        - Memory Grant Wait performance issues
            - RESOURCE_SEMAPHORE – waiting for memory to become available
        - Parallel
            - CXPACKET – Max Degree of Parallelism may be too high, or indexes may needed to be created.
            - SOS_SCHEDULER_YIELD – high CPU utilization, maybe caused by missing indexes – often seen with CXPACKET waits.
- Possible blocking
    - SELECT * FROM sys.dm_exec_requests
        - Active queries, and what resource they are waiting on.
    - SELECT * FROM sys.dm_os_waiting_tasks
        - Wait types for a particular task for a specific query.

102. evaluate server health using DMVs

- See topic 61.
- To use Server-scoped DMVs, you need VIEW SERVER STATE permission on the server.
- In addition:
    - SELECT * from sys.databases
        - msdb, tempdb and model are not listed in Azure SQL Database.
    - SELECT * from sys.objects

        - All tables, queries and other objects.
    - SELECT * FROM sys.dm_os_schedulers where STATUS = 'VISIBLE ONLINE';
        - Shows the vCores.
    - SELECT SERVERPROPERTY('EngineEdition');
        - Returns 5 for SQL Database, 8 for Managed Instance, and <5 for on-prem/VM.

103. perform database consistency checks by using DBCC

- DBCC CHECKDB checks the logical and physical integrity of all objects in a specific database. It:
    - Runs DBCC CHECKALLOC, which checks the consistency of disk space allocation structures
    - Runs DBCC CHECKTABLE for all tables and index views. The DBCC checks the integrity of all pages and structures in a particular table or index view, including:
        - Data pages are correctly linked.
        - Indexes are in the correct sort order.
        - Every row in a table has a matching row in a nonclustered index (and the other way round), and is in the correct partition.
        - DBCC CHECKTABLE ('TableName' OR 'ViewName') checks a table/view (Note: it is a string.)
    - Runs DBCC CHECKCATALOG which checks for catalog consistency, using an internal database snapshot to provide transaction consistency to perform these checks.
        - Does not work on tempdb or Filestream data (binary large objects or BLOBs on the file system).
        - DBCC CHECKCATALOG checks the current database.
        - DBCC CHECKCATALOG (NameOfDatabase) checks a particular database. (Note: it is not a string.)
    - Validates the contents of every indexed view in the database, and link-level consistency between table metadata and file system directories and files.
    - DBCC CHECKDB syntax is similar as DBCC CHECKCATALOG.

103. perform database consistency checks by using DBCC

- Arguments:
    - Relevant Database
        - DBCC CHECKDB (MyDatabase)
            - Note the lack of quote marks.
        - DBCC CHECKDB or DBCC CHECKDB(0)
            - The zero indicates that the current database should be used.
    - What it should do?
        - DBCC CHECKDB (0, NOINDEX)

    - Detect errors only. Smaller execution time, as it does not do intensive checks of nonclustered indexes for user tables

- DBCC CHECKDB (0, REPAIR_REBUILD)
    - Do only repairs which have no chance of data loss. Includes quick repairs (e.g. missing rows in non-clustered indexes), and time-consuming repairs (building an index).
    - Needs to be in single user mode beforehand, i.e.
      ALTER DATABASE MyDatabase SET SINGLE_USER WITH ROLLBACK IMMEDIATE
      GO
      DBCC CHECKDB …
      GO
      ALTER DATABASE MyDatabase SET MULTI_USER
      GO
- DBCC CHECKDB (0, REPAIR_FAST)
    - No repair actions are performed. Only use for Backward compatibility reasons only.
- DBCC CHECKDB (0, REPAIR_ALLOW_DATA_LOSS)
    - Repairs any found errors.
    - REPAIR_ALLOW_DATA_LOSS may cause data loss.
        - Suggest that you create physical copies of the database files beforehand.
        - Needs to be in single user mode beforehand. Additionally, before that, run
          ALTER DATABASE MyDatabase SET EMERGENCY
          This marks it as READ_ONLY, logging is disabled, and access is limited to sysadmins.

- WITH Arguments
    - DBCC CHECKDB (0, REPAIR_REBUILD) WITH …
    - ALL_ERRORMSGS – displays all reported errors per object.
    - EXTENDED_LOGICAL_CHECKS – performs logical consistency checks on indexed views, XML indexes and spatial indexes.
    - NO_INFOMSGS – does not show informational messages.
    - TABLOCK – obtains exclusive locks, which will speed it up, but reduce concurrency.
    - ESTIMATEONLY – No database checks are done, but displays the amount of tempdb space needed to do it.
    - PHYSICAL_ONLY – limits checking to page structure integrity, record header integrity, and consistency of the database.

  - MAX_DOP = number_of_processors – overrides the max degree of parallelism in sp_configure.
- Best practices:
  - BEGIN TRANSACTION beforehand, so the user can confirm that they want to accept the results.
  - For best repairing errors, Microsoft recommends restoring from a backup.
  - After using DBCC CHECKDB, you need to inspect the referential integrity of the database – DBCC CHECKCONSTRAINTS. This checks the integrity of a constraint or all constraints in a table, or all constraints.

104. assess proper database autogrowth configuration

- To look at current settings:
  - SELECT * FROM sys.sysfiles
- To adjust auto-growth setting:
  - ALTER DATABASE MyDB
  - MODIFY FILE
  - (NAME=NameFile,FILEGROWTH=40MB or 40%);
- You can also autogrow files in a particular filegroup.
  - ALTER DATABASE MyDB
  - MODIFY FILEGROUP FilegroupName
  - AUTOGROW_ALL_FILES
    - If any file in a filegroup meets the autogrow threshold, all files in the filegroup will grow.

105. report on database free space

- To display space used, run:
  - EXEC sp_spaceused
  - It shows the unallocated space.
- To display it by file, run:
  - SELECT file_id, name, type_desc, physical_name, size, max_size
  - FROM sys.database_files
  - A max_size of -1 means that it is unlimited.
- To view the number of pages used as well as total free space for a particular database, you can use
  - SELECT allocated_extent_page_count, unallocated_extent_page_count FROM sys.dm_db_file_space_usage
    - Returns space usage information for each data file in the database.

- You can also use:
    - DBCC SQLPERF (LOGSPACE)
    - However, it only shows transaction log space statistics.
- You can also go to Reports – Standard Reports – Disk Usage on Azure VM.
- For tempdb only, you can use:
    - SELECT * FROM sys.dm_db_session_space_usage
        - Number of pages allocated/deallocated by each session.
    - SELECT * FROM sys.dm_db_task_space_usage
        - Pages allocated/deallocated by each task

106. review database configuration options

- These SET options are started with ALTER DATABASE DatabaseName SET = ON/OFF … // GO
    - AUTO_CLOSE ON/OFF
        - Whether the database is shut down after the last user exists.
    - AUTO_CREATE_STATISTICS ON/OFF
        - Creates statistics on single columns in query predicates, to improve query plans and performance.
    - AUTO_UPDATE_STATISTICS[_ASYNC]
        - Query Optimizer updates statistics when they are used by a query and might be out-of-date, after insert/update/delete/merge operations change the data distribution. _ASYNC specifies whether it is done asynchronously or not.
    - AUTO_SHRINK ON/OFF
        - Shrinks when more than 25% of the file contains unused space. Recommended to leave OFF.
    - READ_ONLY / READ_WRITE
        - Can users only read from the database (not modify it).
    - SINGLE_USER / RESTRICTED_USER / MULTI_USER
        - Only one user at a time, or only db_owner fixed database roles and dbcreator and sysadmin fixed server roles (any number), or all users which have appropriate permissions.
    - RECOVERY FULL / RECOVERY BULK_LOGGED / RECOVERY SIMPLE
        - Changes the recovery option. FULL uses transaction log backups. BULK_LOGGED only minimally logs certain large-scale (bulk) operations. Simple only allows for complete backups.
    - COMPATIBILITY_LEVEL = 100 (SQL Server 2008 and R2), 110, 120, 130, 140, 150 (SQL Server 2019)

        - In Azure SQL Database and MI and SQL Server 2014, you cannot set it below SQL Server 2008 (100).

107. prepare databases for Always On Availability Groups

- A secondary database need to be identical to the primary database.
    - Therefore, do a BACKUP and RESTORE, including tail-log backups.
- Then join it – for example:
    - ALTER DATABASE Db1 SET HADR AVAILABILITY GROUP = MyAG;

108. perform transaction log backup

- This only applies to VMs.
- You need to be using full or bulk-logged recovery models.
    - Simple recovery models do not use transaction log backups.
    - To change recovery model:
        - ALTER DATABASE NameOfDatabse
        - SET RECOVERY FULL | BULK_LOGGED | SIMPLE
        - GO
- You need the BACKUP DATABASE and backup log PERMISSIONS.
    - They are already granted in the sysadmin fixed server role, and the db_owner and db_backupoperator fixed database roles.
- Use the following command:
    - BACKUP LOG NameOfDatabase
    - TO MyPreviouslyCreatedNamedBackupDevice
    - NORECOVERY, NO_TRUNCATE
        - NORECOVERY backups the tail of the log and leaves the database in the RESTORING state.
            - Useful when failing over to a secondary database or when saving the tail before a RESTORE.
        - NO_TRUNCATE causes SQL Server to attempt to backup, regardless of the state of the database.
            - Useful if the database is damaged.
        - Suggest using NO_TRUNCATE and NORECOVERY together.
    - GO

109. perform restore of user databases

- In SQL Server in an Azure VM, you can:
    - Perform a complete restore or partial report.

- Restore to a point in time.
- You can also do the following, which we don't need to cover:
  - Restore specific file(s), filegroup(s) or page(s).
  - Restore a transaction log, or

- Use:
  - RESTORE DATABASE NameOfDatabase
  - FROM MyPreviouslyCreatedNamedBackupDevice
  - [WITH RECOVERY | NORECOVERY]
  - [FILE = BackupSetFileNumber]
    - NORECOVERY is useful when you are restoring a single file, but you need to restore more.
    - Use RECOVERY when you have finished restoring, and you want the database to be online.
  - [STOPAT = { 'datetime'| @datetime_var }
    - | STOPATMARK or STOPBEFOREMARK = { MarkName | LSNNumber } [AFTER 'datetime']
- For example:
  - RESTORE … WITH FILE = 6 NORECOVERY, STOPAT = 'Jun 19, 2024 12:00 PM';
  - RESTORE … WITH FILE = 9 RECOVERY;
  - RESTORE VERIFYONLY FROM …
    - Verifies the backup but does not restore it.
- You can only use T-SQL in an MI when doing a complete restore from an Azure Blob Storage Account:
  - RESTORE DATABASE NameOfDatabase
  - FROM URL = 'https:// … ' , 'https:// … ' (etc)
  - It can only be restored onto another MI – not an on-prem SQL Server.

110. perform database backups with options

- MIs have automatic backups. You can create full database COPY_ONLY backups, but not differential, log or file snapshot backups.
  - BACKUP DATABASE NameOfDatabase
  - TO URL = 'https:// … ' , 'https:// … ' (etc)
    - The URLs is for the Microsoft Azure storage service.
    - Maximum backup stripe size (blob size) is 195 Gb.
      - If you want more space, add additional files.
  - WITH COPY_ONLY

        - Does not interfere with the normal sequence of backups.
    - [COMPRESSION | NO_COMPRESSION]
        - This overrides the server-level default. The default is no backup compression.
    - [STATS = X]
        - Displays a message every X% percentage. The default is 10 per cent.

- This is for VMs (and Mis if using COPY_ONLY). The syntax is:
    - BACKUP DATABASE NameOfDatabase
    - [FILEGROUP = 'X', FILEGROUP = 'Y' …]
        - Useful for backing-up only part of a database.
    - TO MyPreviouslyCreatedNamedBackupDevice
        - If backing up to disk, then use TO DISK = 'FileLocation'.
        - You can also use TO TAPE = or TO URL = 'https://…'
    - [MIRROR TO AnotherBackupDevice]
        - Only for the Enterprise edition of SQL Server.
        - Must be the same time as the Primary Backup.
        - You can have up to 3 secondaries.
    - [WITH
        - COPY_ONLY
            - Creating a full backup, but is not treated as a full backup for purposes of future DIFFERENTIAL or TRANSACTION LOG backups.
        - DIFFERENTIAL
            - All the changes since the last FULL backup.
            - Without DIFFERENTIAL or LOG, it is a FULL backup.
        - COMPRESSION | NO_COMPRESSION
            - Only use in the Enterprise edition. Overrides the server-level default
        - DESCRIPTION – up to 255 characters
        - NAME – up to 128 characters. Default is blank.
        - CREDENTIAL
            - Used only when creating backups to Azure Blobs.
        - ENCRYPTION
            - Choose from AES_128, AES_192, AES_256, TRIPLE_DES_3KEY or NO_ENCRYPTION (default).
            - If you encrypt, you will also need to use SERVER CERTIFICATE or SERVER ASYMMETRIC KEY.

        - FILE_SNAPSHOT [EXPIREDATE = 'date' | RETAINDAYS = days]
            - Used when creating a snapshot of the database files and storing them into Azure Blobs.

- [WITH
    - NOINIT | INIT
        - Whether the backup operation appends to/overwrites the existing backup sets on the backup media. The default is NOINIT (append).
    - NOSKIP | SKIP
        - Checks whether a backup operation checks the expiration date and time of the backup sets on the media before overwriting them. The default is NOSKIP (Check the date/time).
    - NOFORMAT | FORMAT
        - Whether the media header should be written on the volumes used for the backup operation, overwriting any existing media header and backup sets. The default is NOFORMAT.
            - Be careful from using FORMAT, as it renders the entire media set unusable.
            - FORMAT implies SKIP.
    - NO_CHECKSUM | CHECKSUM
        - Whether backup checksums are enabled – this validates the backup. The default is NO_CHECKSUM (no generation of backup checksums).
    - STOP_ON_ERROR | CONTINUE_AFTER_ERROR
        - Whether BACKUP stops if there is a page checksum error. The default of STOP_ON_ERROR (stop if it doesn't verify). CONTINUE_ON_ERROR is best used when there are checksum errors.
    - STATS = X
        - Whether a percentage is displayed every X% (the default is 10%).
    - REWIND | NOREWIND
        - Whether a TAPE device is released and rewound. Default is REWIND (yes).
    - UNLOAD | NOUNLOAD
        - Whether the tape is rewound and unloaded. Default is UNLOAD (yes).
- [WITH
    - When using BACKUP LOG:
        - NORECOVERY
            - Backs up the tail of the log and leaves the database in the RESTORING state. Useful when failing over to a secondary database or when saving the tail of the log before a RESTORE operation.

        - STANDBY = standby_file_name
            - Backs up the tail of the log and leaves the database in a read-only and STANDBY state.
            - The file holds the rolled back changes.
        - NO_TRUNCATE
            - The log is not truncated and requires SQL Server to attempt to backup regardless of the state of the database.
            - Is generally used if the database is SUSPENDED – when the database has failed.

111. manage certificates

- To create a self-signed certificate:
    - CREATE CERTIFICATE CertificateName
        - ENCRYPTION BY PASSWORD = 'ComplicatedPassword'
            - If this is not used, the private key is encrypted using the database master key.
        - WITH SUBJECT = 'CertificateSubjectName',
            - This is a field in the certificate metadata.
        - EXPIRY_DATE = '20291231';
            - You can also have a START_DATE (in UTC). If not specified, START_DATE defaults to current date, and EXPIRY_DATE (UTC) is one year after START_DATE.
    - GO
- By default, this certificate is stored in the master database.
    - The Azure Key Vault can store customer-managed certificates ("Bring your own Key – BYOK")
- To restore a previously-created certificate, you can also use CREATE CERTIFICATE with FILE = 'path'
    - Azure SQL Database does not support creating a certificate from a file or using private key files.
- You can also ALTER CERTIFICATE
    - ALTER CERTIFICATE CertificateName
    - WITH PRIVATE KEY (ENCRYPTION BY PASSWORD = 'ComplicatedPassword')
    - You can change the password, but not the SUBJECT or DATEs.
- You can also DROP CERTIFICATE.

112. manage security principals

- (Use "GO" after each statement).
- For MI and SQL Server in VMs:

  - To create a login for a local Azure Active Directory account:
    - CREATE LOGIN [login_name] FROM EXTERNAL PROVIDER -- the last 3 words indicate Azure AD.
  - To check
    - SELECT * FROM sys.server_principals;
  - To add members into a server role:
    - ALTER SERVER ROLE server_role ADD MEMBER [login_name]
  - To create a user:
    - USE <Database Name>
    - CREATE USER [user_name] FROM LOGIN [login_name]

- For Azure SQL Database:
  - To create a user based on a local AAD account:
    - USE <Database Name>
    - CREATE USER [user_name] FROM EXTERNAL PROVIDER
  - You can create users in the master database, then create a user based on it, but it is better practice to do the above:
    - [In Master]
      CREATE LOGIN demo WITH PASSWORD = 'Pa55.w.rd'
      - To check
        - SELECT * FROM sys.sql_logins
    - [In database]
      CREATE USER demo FROM LOGIN demo
- To check users:
  - SELECT * FROM sys.database_principals
- (Use “GO” after each statement).
- To grant permissions:
  - AUTHORIZATION PERMISSION ON SECURABLE::NAME TO PRINCIPAL;
    - For example:
      GRANT SELECT ON OBJECT::Region TO Ted [WITH GRANT OPTION];
  - AUTHORIZATION can be GRANT, REVOKE or DENY.
    - REVOKE is the opposite of a GRANT.
    - DENY beats all GRANTs from other roles.
  - PERMISSION can be
    - For tables and views, SELECT, INSERT, UPDATE and DELETE.

      - They can also be CONTROL (all rights), REFERENCES (view foreign keys), TAKE OWNERSHIP, VIEW CHANGE TRACKING and VIEW DEFINITION.
    - For schema, ALTER permission on a schema is wide-ranging. You can alter, create or drop any securable in that schema. However, you cannot change ownership.
    - For functions and stored procedures, ALTER (change definition), CONTROL, EXECUTE, VIEW CHANGE TRACKING and VIEW DEFINITION.
    - You can give permissions on a stored procedure/function without giving permissions on the underlying tables/views through ownership chaining (see topic 27).
    - ALL (deprecated, maintained for backward compatibility)
      - For databases, that means BACKUP DATABASE and LOG, CREATE DATABASE, FUNCTION, PROCEDURE, RULE, TABLE, and VIEW (note – not DROP or ALTER).
      - For tables and views, ALL means DELETE, INSERT, REFERENCES, SELECT, and UPDATE.
      - For procedures, ALL means EXECUTE.
      - For scalar functions, ALL means EXECUTE and REFERENCES.
      - For table-valued functions, ALL means DELETE, INSERT, REFERENCES, SELECT and UPDATE
  - SECURABLE is OBJECT::, SCHEMA:: or DATABASE:: or SERVER::
  - PRINCIPAL is a login, user or role.
  - The optional [WITH GRANT OPTION] allows you to grant that permission to others.
- To check permissions:
  - SELECT * FROM sys.fn_my_permissions (NULL, 'DATABASE') -- login permissions.
  - SELECT * FROM sys.fn_my_permissions ('MyMITestDB','DATABASE') – user permissions.
- To test user permissions:
  - CREATE PROCEDURE proc_name WITH EXECUTE AS user_name AS …
  - or if sysadmin in MI or VM:
    - EXECUTE AS LOGIN = ‘login_name’
  - or EXECUTE AS USER = 'user_name'

113. configure permissions for users to access database objects

- To add permissions to access database objects:
  - AUTHORIZATION PERMISSION ON SECURABLE::NAME TO PRINCIPAL;
  - For example:
    - GRANT SELECT ON OBJECT::Region TO Phillip;

        - GRANT SELECT ON OBJECT::Customer(CustomerName) TO Phillip; -- This is a column.
    - AUTHORIZATION can be GRANT, REVOKE or DENY.
    - PERMISSION is:
        - CONTROL (ownership-like capabilities).
        - ALTER – allows for ALTER, CREATE and DROP.
        - ALTER ANY [Server_Securable] – CREATE, ALTER and DROP things such as LOGIN.
        - DELETE/INSERT/SELECT/UPDATE
        - TAKE OWNERSHIP – allows grantee to take ownership, but doesn't automatically take it.
        - IMPERSONATE Login/User – allows principal to impersonate, but doesn't automatically do it.
        - CREATE Server-/Database-/Schema-Securable.
        - VIEW DEFINITION – access to metadata.
        - REFERENCES – permission to create a FOREIGN KEY constraint.
    - OBJECT can be a database, schema or object
    - PRINCIPAL is a login, user or role.
- Check permissions using:
    - SELECT * FROM fn_builtin_permissions(default);
        - All permissions
    - SELECT * FROM fn_builtin_permissions('assembly');
        - Specific database.
    - SELECT * FROM fn_my_permissions('Orders55', 'object');
        - Specific object for a specific role.
    - SELECT * FROM sys.database_permissions WHERE major_id = OBJECT_ID('Yttrium');
        - Specific object.

114. configure permissions by using custom roles

- To create a role (database-level securable):
    - CREATE ROLE role_name [AUTHORIZATION owner_name]
    - If AUTHORIZATION is not given, it will be the current user.
- To add a user to a database-level role:
    - USE <Database Name>
    - ALTER ROLE db_datareader ADD MEMBER database_principal
    - You can also use DROP MEMBER instead of ADD MEMBER.

  - database_principal is a database user or user-defined role, but not a fixed database role or a server principal.
  - You need ALTER permission on the role, or ALTER ANY ROLE on the database, or db_securityadmin or db_owner.
- To alter the name of the role:
  - ALTER ROLE OriginalRoleName WITH NAME = NewRoleName;