https://docs.microsoft.com/en-us/learn/modules/deploy-azure-sql-database/2-explain-paas-options-deploy-sql-server-azure
Azure SQL database in this section. Azure SQL Database is available in three different deployment options:

A single database – A single database that is billed and managed on a per database level

Elastic Pools – A group of databases that are managed together and share a common set of resources

Hyperscale – a single database offering that allows databases to scale much beyond the 4-TB limit of an Azure SQL Databa

Even in the cloud all services are backed by physical hardware. The Azure SQL Database allows you to choose from two different purchasing models:

- Database Transaction Unit (DTU) – DTUs are calculated based on a formula combining compute, storage, and I/O resources.
- vCore – The vCore model allows you to purchase a specified number of vCores based on your given workloads. vCore is the default purchasing model when purchasing Azure SQL Database resources. vCore databases have a specific relationship between the number of cores and the amount of memory and storage provided to the database.

Service Tier Options

The DTU model is available in three different service tiers:

- Basic
- Standard
- Premium

You can purchase vCore databases in three different service tiers as well:

- General Purpose – This tier is for general purpose workloads. It is backed by Azure premium storage. It will have higher latency than Business Critical
- Business Critical – This tier is for high performing workloads offering the lowest latency of either service tier. This tier is backed by local SSDs instead of Azure blob storage. It also offers the highest resilience to failure as well as providing a built-in read-only database replica that can be used to off-load reporting workloads
- Hyperscale – Hyperscale databases can scale far beyond the 4 TB limit the other Azure SQL Database offerings and have a unique architecture that supports databases of up to 100 TB

Backups

Backups are stored in Azure blob geo-redundant storage and by default are retained for between 7 and 35 days, based on the service tier of the database. Basic and vCore databases default to seven days of retention, and on the vCore databases this value can be adjusted by the administrator. The retention time can be extended by configuring long-term retention (LTR), which would allow you to retain backups for up to 10 years.

Database Backups are taken on a given schedule:

- Full – Once a week
- Differential – Every 12 hours
- Log – Every 5-10 minutes depending on transaction log activity

f the need to restore a database arises, there are several options available. Due to the nature of Platform as a Service, you cannot manually restore a database using conventional methods, such as issuing the T-SQL command RESTORE DATABASE.

Regardless of which restore method is implemented, it is not possible to restore over an existing database. If a database needs to be restored, the existing database must be dropped or renamed prior to initiating the restore. Furthermore, keep in mind that depending on the platform service tier, restore times could fluctuate. It is recommended that you test the restore process to obtain baseline metrics on how long a restore could potentially take.

The available restore options are:

Restore using the Azure portal – Using the Azure portal you have the option of restoring a database to the same Azure SQL Database server, or you can use the restore to create a new database on a new server in any Azure region.

Restore using scripting Languages – Both PowerShell and Azure CLI can be utilized in order to restore a database.

Active geo-replication

Geo-replication is a business continuity feature that asynchronously replicates a database to up to four secondary replicas. As transactions are committed to the primary (and its replicas within the same region), the transactions are sent to the secondaries to be replayed. Because this communication is done asynchronously, the calling application does not have to wait for the secondary replica to commit the transaction prior to SQL Server returning control to the caller.

The secondary databases are readable and can be used to offload read-only workloads, thus freeing up resources for transactional workloads on the primary or placing data closer to your end users. Furthermore, the secondary databases can be in the same region as the primary or in another Azure region.

Failover groups

Failover groups are built on top of the technology used in geo-replication, but provide a single endpoint for connection. The major reason for using failover groups is that the technology provides endpoints, which can be utilized to route traffic to the appropriate replica. Your application can then connect after a failover without connection string changes.

Serverless

The name “Serverless” can be a bit confusing as you still deploy your Azure SQL Database to a logical server, to which you connect. Azure SQL Database serverless is a compute tier that will automatically scale up or down the resources for a given database

based on demand. If the workload no longer requires compute resources, the database will become “paused” and you will not be charged during the period when the database is in this state. When a connection attempt is made, the database will “resume” and become available. Resuming the database is not instantaneous.

Another difference between serverless and the normal vCore model of Azure SQL Database is that with serverless you can specify a minimum and maximum number of vCores. Memory and I/O limits are proportional to the range that is specified.

databases that are not deployed as serverless are referred to as “provisioned”.

The setting to control pausing is referred to as the autopause delay and has a minimum value of 60 minutes and a maximum value of seven days. If the database has been idle for that period of time, it will then pause. Once the database has been inactive for the specified amount of time, it will be paused until a subsequent connection is attempted. Any applications using serverless should be configured to handle connection errors and include retry logic, as connecting to a paused database will generate a connection error.

Serverless is not fully compatible with all features in Azure SQL Database as some features are running background processes all the time. These features include:

- Geo-replication
- Long-term backup retention
- A job database in elastic jobs
- The sync database in SQL Data Sync (Data Sync is a service that replicates data between a group of databases)

Hyperscale

Azure SQL Database has been limited to 4 TB of storage per database for many years. This restriction is due to a physical limitation of the Azure infrastructure. Azure SQL Database Hyperscale changes the paradigm and allows for databases to be 100 TB or more. Hyperscale introduces new horizontal scaling techniques to add compute nodes as the data sizes grow. The cost of Hyperscale is the same as the cost of Azure SQL Database; however, there is a per terabyte cost for storage. You should note that once an Azure SQL Database is converted to Hyperscale, you cannot convert it back to a “regular” Azure SQL Database.

Deploy single SQL database via

1. **Powershell/CLI**
2. **The azure portal**
3. **Azure Resource Manager templates**

A Resource Manager template gives you the most granular control over your resources, and Microsoft provides a GitHub repository called "Azure-Quickstart-Templates", which hosts Azure Resource Manager templates that you can reference in your deployments.

Deploy SQL database elastic pool

Elastic pools are a deployment option in which you purchase Azure compute resources (CPU, memory, and storage) that is then shared among multiple databases defined as belonging to the same pool. An easy comparison to an on-premises SQL Server is that an elastic pool is like a SQL Server instance that has multiple user databases. By using elastic pools, you can easily manage pool resources while at the same time potentially saving costs.

Elastic pools also facilitate easy scalability up to the set limits such that if a single database within the pool needs resources due to an unpredictable workload, the resources are there.

If the entire pool needs additional resources, a simple slider option within the Azure portal will facilitate scaling the elastic pool up or down.

Deploy SQL managed instance

While many organizations initially migrate to Azure using IaaS offerings, the platform as a service (PaaS) service offering allows for additional benefits. One key benefit is that you no longer have to install or patch SQL Server as that is performed by the service. Additionally, consistency checking, and backups are also part of the managed service, and there are additional security and performance tools that are included in the PaaS offerings.

Azure SQL Managed Instance is a fully functional SQL Server instance that is almost 100% compatible with your on-premises ecosystem including features like SQL Agent, access to tempdb, cross-database query and common language runtime (CLR). The service uses the same infrastructure as Azure SQL Database and includes all the benefits of the PaaS service such as automatic backups, automatic patching, and built-in high availability, just to name a few.

Summary

- 1 minute

In this module, you learned about the platform as a service options for SQL Server on Azure. Azure SQL Database is a flexible platform, well suited for software as service applications and has a number of platform options for large databases, multiple databases, or development environments that do not need to be online 24x7. Azure SQL Database Managed Instance is a PaaS version of SQL Server that allows you to easily move your on-premises environments into a managed service. Managed Instance also supports many server-level capabilities that are not available with Azure SQL Database.

Now that you've reviewed this module, you should be able to:

- Gain an understanding SQL Server in a Platform as a Service (PaaS) offering
- Understand PaaS provisioning and deployment options
- Understand elastic pools
- Examine Managed Instances

- Configure a template for PaaS deployment
-

Deploy MariaDB, MySQL, and PostgreSQL on Azure

Introduction

When development speed is of the essence, open source database platforms usually are unmatched when it comes to delivering a fully managed ready to roll database for applications. Many developers prefer open source systems for this very reason. Microsoft has rounded out their relational database offerings by providing open source database platforms such as MySQL, MariaDB, and PostgreSQL. These platforms are a great compliment to the existing Azure SQL Server.

Describe open source offerings

The Azure ecosystem offers three different open source database platforms. Each platform has different attributes that facilitate moving to the cloud quickly and painlessly. Each of these services comes with native high availability, automatic patching, automatic backups and the highest level of security protection. These offerings are fully supported by Microsoft from the service all the way through the database engine.

Azure Database for MySQL is a full managed enterprise grade database service that is ready to easily lift and shift customer environments to the Azure cloud.

Customers can use their existing frameworks and languages to ensure that a migration doesn’t disrupt any business activity. In addition, the service has

built-in high availability and dynamic scaling, which helps to meet any fluctuation in performance demands.

Azure Database for MariaDB is similar to the MySQL offering. It also allows for continued use of frameworks and languages of your choice. High availability and dynamic scaling are also provided by the service to ensure that customer demands are met within a moment's notice.

Azure Database for PostgreSQL helps customers to build large scalable applications. The service allows for horizontal scaling and is available with Hyperscale which allows for unparalleled performance.

It also integrates with several native features such as geospatial support, rich indexing, and JSONB along with other extensions. Azure Database for PostgreSQL Hyperscale is ideal for multi-tenant applications, with minor code changes to allow for data sharding.

Service tiers

There are three service tiers for each offering. Each tier has an ideal workload for which it is designed and allows you to choose from a variety of performance options.

- Basic – This tier is best for light workloads that need minimal compute and I/O performance.
- General Purpose – This tier is great for most workloads requiring scalable I/O throughput along with a healthy balance of compute and memory.
- Memory Optimized – This tier is suitable for workloads that demand high performance and require in-memory speed for quick processing of transactions along with higher concurrency.

Azure Database for PostgreSQL Hyperscale

While Hyperscale shares a name with Azure SQL Database and while they both offer horizontal scalability supporting very large data volumes, the Hyperscale

technology for PostgreSQL is implemented differently. Hyperscale allows the servers for Azure Database for PostgreSQL (called nodes) to work together in a “shared nothing” architecture design. Nodes are added to a server group, and each server group has a coordinator node, and multiple workers nodes. Applications send their queries to the coordinator node, which relays the query to relevant worker nodes and gathers the results.

Hyperscale databases are sharded, which means that the data in a single table can be split across multiple nodes, using a type of table called a distributed table. This sharding allows for both parallelization and distribution of queries across nodes. The worker nodes and coordinator nodes can be scaled independently of each other.

Creating a Hyperscale PostgreSQL deployment is different than deploying a single instance of the service. In this case, Hyperscale allows you to deploy additional worker nodes along with a coordinator node. You can deploy up to 20 worker nodes by default (this number is a soft limit; for additional nodes, you can contact Microsoft support). You can also configure high availability for each node to ensure that any disruptions have minimal impact on your applications.

Deploy MySQL and MariaDB to Azure

HA configuration

High availability for both Azure Database for MySQL and Azure Database for MariaDB comes packaged in with the service so there is less administrative work that needs to be done. The service provides a guaranteed high level of availability, offering an uptime of 99.99%. This percentage of uptime equates to a maximum of 52.60 minutes of downtime per year.

Transactions on either platform are written synchronously to storage. If a node interruption occurs, the database server will automatically create a new node and subsequently attach the storage to the new node. Any transactions in flight are not committed and active connections to the database are dropped. As

mentioned with Azure SQL Database, it is important to ensure that applications that connect to the database service include retry logic, also known as connection resiliency, in their database connections.

The Azure Service makes it easy to scale out read workloads using read replicas for MySQL and MariaDB. The example below illustrates the procedure for MySQL however MariaDB uses a similar process.

Explain database migration options

Many organizations are migrating their Oracle databases to Azure PostgreSQL to reduce licensing costs. Migrating to the Azure Database platform is made easier by the Azure Database Migration Service (DMS). The DMS supports both homogenous (for example, MySQL in a VM to Azure Database for MySQL) and heterogenous (for example, Oracle in a VM to Azure Database for PostgreSQL) migrations.

DMS performs an initial load of your on-premises database or database running in an Azure VM to Azure Database, and then continuously syncs new database transactions to the Azure target.

When you are ready to cut over to the target Azure service, you can stop the replication, and switch the connection strings in your application to the Azure Database.

Check your knowledge

1. You are deploying a mission critical MySQL database to support an e-commerce site which depends on low latency. What should you configure your application to do in order to handle transient errors?

- (●) Connection resiliency ✓
  - That's correct. Connection resiliency should be built into your application to handle transient failures, like failover or database resizing.
- ( ) Secure Socket Layer (SSL)
- ( ) An Azure Virtual Network

2. Which of the following is a benefit of Azure Database for PostgreSQL?

- (●) Managed automated backups ✓
  - That's correct. Automated backup is a feature of the service.
- ( ) Automatic Query Tuning
- ( ) Automated cross-region disaster recovery

3. What's the process to upgrade a major version of Azure Database for MySQL?

- ( ) The Service performs the update for you
- (●) Create a dump and restore to a server at the higher version ✓
  - That's correct. It is required to create a backup and restore to upgrade to a new major version.
- ( ) Change the version of the server in the portal

Summary

- 1 minute

The open source offerings on the Azure Database for MariaDB, MySQL, and PostgreSQL platform cover the most popular open source offerings and offer value-added services like built-in high availability and backups. These offerings also provide easy scaling up and down for the flexibility your application needs.

Now that you've reviewed this module, you should be able to:

- **Understand the service tiers available for open source database platforms**

- Deploy MySQL, MariaDB, and PostgreSQL
- Configure high availability option for MySQL and MariaDB
- Configure Scale Out reads for MySQL and MariaDB
- Describe the SSL Options for MySQL and MariaDB
-

Protect data in-transit and at rest

Explore encryption options available within Azure SQL, on-premises SQL Server and Open Source data platforms and Secure Enclaves. Implement database and instance firewalls.

Check your knowledge

1. Which security object is required in order to enable transparent data encryption?

- ○ Credential
- ◉ Master Key ✓

  That's correct. A master key is required in order to encrypt databases.
- ○ Login

2. Which feature prevents members of the sysadmin role from viewing the values of data in a table?

- ◉ Always Encrypted ✓

  That's correct. Always Encrypted only stores encrypted data in the SQL Server so the data is never visible to a user of the database without the column key, held by the application.
- ○ Dynamic Data Masking
- ○ Transparent Data Encryption

3. Which is a valid location for Always Encrypted keys?

- ◉ Azure Key Vault ✓

  That's correct. Azure Key Vault allows for the storage of both Always Encrypted and TDE keys.
- ○ Azure Automation
- ○ Azure Blob Storage

4. Which feature provides a private IP address for an Azure SQL Database?

- ○ Network Endpoints
- ◉ Private Link ✓

  That's correct. Private link provides a private IP address in an Azure vNet for an Azure SQL Database.
- ○ Database Firewall

5. Which technique can be used to create database firewall rules in Azure SQL Database?

- ○ Running a PowerShell script
- ○ Running a Azure CLI script
- ◉ Executing a T-SQL statement ✓

  That's correct. This is the only way to create a database firewall rule.

Summary

- 1 minute

This module explored the encryption options available within Microsoft SQL Server, Azure SQL Database, Azure SQL Database for MySQL and Azure SQL Database for PostgreSQL. Each of the various platforms support different database encryption options. In this module you explored these data encryption options and how to configure them.

This module also explored the practices of securing the firewalls of an Azure SQL Server. Furthermore it showed you how Always Encrypted is used to protect data in transit.

Now that you've reviewed this module, you should be able to:

- **Understand the difference between database and instance firewalls in Azure SQL Database**
- **Understand the data encryption options available in the various platforms**
- **Understand the role of Azure Key Vault in Transparent Data Encryption**
- **Understand what Always Encrypted Enclaves are used for**

- Understand what Dynamic Data Masking is used for and how to configure it

Implement compliance controls for sensitive data

Describe data classification

- 4 minutes

Confidential data stored within a Microsoft SQL Server or Azure SQL Database should be classified within the database. This classification allows the SQL Server users as well as other applications to know the sensitivity of the data that is being stored. Data classification with the database is done on a column by column basis. It is possible for a single table to have some columns be public, some columns be confidential, and some columns be highly confidential.

Data classification was first introduced into SQL Server Management Studio and used extended properties of objects to store its data classification information. Starting with SQL Server 2019 (and in Azure SQL Database) this metadata is now stored in a catalog view called sys.sensitivity_classifcations.

The Azure portal provides a management pane for data classification of your Azure SQL Database as shown below. You can reach this screen by clicking Data Discovery and Classification in the Advanced Data Security screen, which is in the Security section of the main blade for your Azure SQL Database.

In both the Azure portal and SQL Server Management Studio, you can configure data classification. The classification engine scans your database and locates columns with names that indicate that the column could have sensitive information. One example is that a column named email would be classified by default as containing sensitive personal information. Verify and validate this data, since it is based on column name, so a column named column1 that contained email addresses would not be classified as sensitive personal information.

Columns can be classified using the sensitivity wizard in SQL Server Management Studio, from the Advanced Data Security screen in the Azure portal for Azure SQL Database, or by using the ADD SENSITIVITY CLASSFICATION T-SQL command as shown below:

SQL

```
Copy
ADD SENSITIVITY CLASSIFICATION TO

[Application].[People].[EmailAddress]

WITH (LABEL='PII', INFORMATION_TYPE='Email')

GO
```

Classification of data allows you to easily identify the sensitivity of data within the database. Knowing what columns contain sensitive data allows for easier audits and allows you to more easily identity which columns are good choices for data encryption. Classification will allow other employees within the company to make better decisions on how to handle the data which is available within the database.

To get maximum benefit out of Advanced Threat Protection you will want to enable auditing on your databases. Auditing will allow for deeper investigation into the source of the problem if ATP detects an anomaly. ATP supports alerts for the following threats:

Vulnerability to SQL injection—This alert looks for T-SQL code coming into your database that may be vulnerable to SQL injection attacks. An example would be a stored procedure call that did not sanitize user inputs.

Potential SQL injection—This alert is triggered when an attacker is actively attempting to execute a SQL injection attack.

Access from unusual location—This alert is triggered when a user logs in from an unusual geographic location.

Access from unusual Azure data center—This alert is looking for attacks from an Azure data center that is not normally accessed.

Access from unfamiliar principal—This alert is raised when a user or applications log in to a database that they have not previously accessed.

Access from a potentially harmful application—This alert detects common tools that are used to attack databases.

Brute force SQL credentials—This alert is triggered when there a high number of login failures with different credentials.

---

1. Where is the data from data classification stored in SQL Server 2019?

- ○ In the extended properties for each object
- ◉ In the sys.sensitivity_classifications catalog view

  That's correct. This is the correct view.
- ○ In the sys.all_columns catalog view

2. Which of the following threats is analyzed by Advanced Threat Protection?

- ○ Weak passwords
- ○ Open Firewall rules
- ◉ SQL Injection

  That's correct. One of the key features of ATP is SQL Injection inspection.

3. Which attack type is commonly associated with dynamic SQL?

- ◉ SQL Injection

  That's correct. SQL Injection attacks are common with poorly coded dynamic SQL.
- ○ Brute Force
- ○ Data Exfiltration

Configure database authentication and authorization

Introduction

- 1 minute

Azure SQL Database has several authentication and authorization options which are different from the options in SQL Server. This is because Azure SQL Database and Azure SQL Managed Instance rely on Azure Active Directory instead of Windows Server Active Directory.

This module explores the practices of granting permissions and what the various permissions do within a database. This module also explores the concept of “least privilege”. While the built-in roles in SQL Server and other database engines provide broad brushes of security privileges, many applications need more granular security on database objects.

Describe Active Directory and Azure Active Directory

what is the difference between Azure Active Directory and Windows Server Active Directory, which we’ll refer to simply as ‘Active Directory’?. This is especially confusing for new administrators because Azure Active Directory interacts with Active Directory. Both solutions provide authentication services and identity management, but in different ways—Active Directory uses a protocol called Kerberos to provide authentication using tickets, and it is queried by the Lightweight Directory Access Protocol (LDAP). Azure Active Directory uses HTTPS protocols like SAML and OpenID Connect for authentication and uses OAuth for authorization.

The two services have different use cases—for example, you cannot join a Windows Server to an Azure Active Directory domain and work together in most organizations to provide a single set of user identities. A service called Azure Active Directory Connect connects your Active Directory identities with your Azure Active Directory.

Describe authentication and identities

Both on-premises SQL Server installations and SQL Server installations within Azure Virtual Machines support two modes of authentication:
SQL Server authentication and Windows Authentication.

When using SQL Server authentication, SQL Server-specific login name and password information is stored within SQL Server, either in the master database, or in the case of contained users, within the user database.

Using Windows Authentication, users to connect to the SQL Server using the same Active Directory account they use to log into their computer (as well as accessing file shares and applications).

Active Directory authentication is considered to be more secure because SQL Server authentication allows for login information to be seen in plain text while being passed across the network.

In addition, Active Directory authentication makes it easier to manage user turnover. If a user leaves the company and you use Windows authentication, the administrator would only have to lock the single Windows account of that user, instead of identifying each occurrence of a SQL login.

Azure SQL Database similarly supports two different modes of authentication, SQL Server authentication and Azure Active Directory authentication. SQL Server authentication is the same authentication method that has been supported in SQL Server since it was first introduced, where user credentials are stored within either the master database the user database.

Authentication via Azure Active Directory allows the user to enter the same username and password, which is used to access other resources such as the Azure portal or Microsoft 365.

Azure Active Directory can be configured to sync with the on-premises Active Directory. This option allows users to have the same usernames and passwords to access on-premises resources as well as Azure resources. Azure Active directory adds on additional security measures by allowing the administrator to easily configure multi-factor authentication (MFA).

With MFA enabled on an account, after the correct username and password is supplied, a second level of authentication is required. By default, MFA can be configured to use the Windows Authenticator application, which will then send a push notification to the phone

. Additional options for the default MFA action include sending the recipient a text message with an access code, or having the user enter an access code that was generated with the Microsoft Authenticator application. If a user has MFA enabled, they have to use the Universal Authentication with MFA option in Azure Data Studio and SQL Server Management Studio.

Describe Security Principals

Security Principals are entities that can request SQL Server resources and to which you can (usually) grant permissions. There are several sets of security principals in SQL Server. Security principals exist at either the server level or the database level and can be either individuals or collections. Some sets have a membership controlled by the SQL Server administrators, and some have a fixed membership.

Logins and Server Roles are the server-level security principals we will be discussing. New logins can be added by administrators, but new server roles cannot be added.

At the database level, we'll look at users, database roles, application roles.

Describe database and object permissions

All Relational Database Management platforms have 4 basic permissions which control data manipulation language (DML) operations. These permissions are SELECT, INSERT, UPDATE, and DELETE. These permissions apply to all SQL Server platforms as well as Azure SQL Database for MySQL and Azure SQL Database for PostgreSQL.

All of these permissions can be granted, revoked or denied on tables and views. If a permission is granted using the GRANT statement, then the permission is given to the user or role referenced in the GRANT statement. Users can also be denied permissions using the DENY command. If a user is granted a permission and denied the same permission, the DENY will always supersede the grant, and the user will be denied access to the specific object.

Table and view permissions

Tables and views represent the objects on which permissions can be granted within a database. Within those tables and views, you can additionally restrict the columns that are accessible to a given security principal (user or login). SQL Server and Azure SQL Database also include row-level security which can be used to further restrict acceSS

Explain execute as user

- **10 minutes**

The EXECUTE AS [user], or EXECUTE AS [login] (only availably in SQL Server and Azure SQL Managed Instance) commands allow for the user context to be changed. As subsequent commands and statements will be executed using the new context with the permissions granted to that context.

If a user has a permission and the user no longer needs to have that permission, permissions can be removed (either grants or denies) using the REVOKE

command. The revoke command will remove any GRANT or DENY permissions for the right specified to the user specified.

Ownership Chains

A concept called chaining applies to permissions, which allows users to inherit permissions from other objects. The most common example of chaining is a function or stored procedure that accesses a table during its execution. If the procedure has the same owner as the table, the stored procedure is able to be executed and access the table, even though the user does not have rights to access the table directly. This access is available because the user inherits the rights to access the table from the stored procedure, but only for the duration of the execution of the stored procedure, and only within the context of the stored procedures execution.

In the example below, run as a database owner or server administrator, a new user is created and added as a member of a new *SalesReader* role, which is then granted permission to select from any object and execute any procedure in the Sales schema. A stored procedure is then created in the Sales schema that accesses a table in the Production schema.

Explain the policy of least privilege

- 3 minutes

The principle of least privilege is fairly simple. The basic idea behind the concept is that users and applications should only be given the permissions needed in order for them to complete the task. Applications should only have permissions that they need to do in order to complete the task at hand.

As an example, if an application accesses all data through stored procedures, then the application should only have the permission to execute the stored procedures, with no access to the tables.

Dynamic SQL

Dynamic SQL is a concept where a query is built programmatically. Dynamic SQL allows T-SQL statements to be generated within a stored procedure or a query itself. A simple example is shown below.

SQL

Copy

```
SELECT 'BACKUP DATABASE ' + name + ' TO DISK =''\\backup\sql1\' + name
+ '.bak'''

FROM sys.databases
```

The above statement will generate a list of T-SQL statements to back up all of the database on the server. Typically, this generated T-SQL will be executed using sp_executesql or passed to another program to execute.

Check your knowledge

1. Which protocol is used by Azure Active Directory for Authorization?

- ○ Kerberos
- ○ LDAP
- ◉ OAuth ✓

  That's correct. Azure Active Directory uses HTTPS protocols like SAML and OpenID Connect for authentication and uses OAuth for authorization.

2. Which database stores the information about logins in SQL Server?

- ◉ master ✓

  That's correct. Logins are stored in the master database.

- ○ model
- ○ msdb

3. Which role allows users to create users within a database?

- ○ db_datareader
- ◉ db_accessadmin ✓

  That's correct. Access admin can add users to the database and create them.

- ○ db_securityadmin

4. Which permission allows the user to perform any option against a database object?

- (●) Control ✓
  That's correct. Control allows the user to drop or modify an object.
- ( ) Delete
- ( ) View Definition

5. Which database object can be granted insert access?

- ( ) Functions
- (●) Tables ✓
  That's correct. Tables can have data inserted into them.
- ( ) Procedures

6. What feature allows a user to execute a stored procedure even if she doesn't have permission to access the tables referenced in the stored procedure?

- (●) Ownership chaining ✓
  That's correct. Ownership chaining effectively gives the user temporary access to the objects called by the procedure.
- ( ) Principle of least privilege
- ( ) Granular security

1. Which type of storage should be used in conjunction with Azure VMs for SQL Server data files?

- ○ Table storage
- ○ Blob storage
- ◉ Disk storage ✓

That's correct. Disk storage is designed for VMs and is the best storage option.

2. Which of the following can be limited using Resource Governor?

- ○ Buffer pool allocation
- ◉ Write IOPs ✓

That's correct. Write IOPs can be controlled by RG.

- ○ Recompilation

3. Which is an option from the SQL Server Resource Provider for Azure VMs?

- ◉ Storage configuration ✓

That's correct. The resource provider will configure your storage pool.

- ○ Changing max degree of parallelism
- ○ Maintenance plan

Check your knowledge

1. Which intelligent query processing feature allows for faster calculations of a large number of rows?

- (•) Batch mode on row store ✓
  - That's correct. Batch mode is most effective in aggregation queries.
- ( ) Approximate count distinct
- ( ) Interleaved execution

2. Which component of resource governor allows you to configure limits on system resources?

- ( ) Workload groups
- ( ) Classifier functions
- (•) Resource pools ✓
  - That's correct. Resource pools are where resources are assigned in resource governor.

3. Which database setting affects the way the query optimizer generates execution plans?

- ( ) Recovery model
- ( ) Optimize for ad hoc workloads
- (•) Compatibility level ✓
  - That's correct. This setting affects the way the plan is created based on a number of engine

SQL Server and Azure SQL have added many features in recent releases to improve performance. Many of these features can be enabled at the individual database level, and may also be controlled using the compatibility level of the database. Azure Database for MariaDB/MySQL/PostgreSQL includes the Query Store, which helps you identity problematic queries.

Now that you've reviewed this module, you should be able to:

- **Understand database scoped configuration options**
- **Understand the features of Intelligent Query Processing**
- **Know your performance monitoring options in Azure Database for PostgreSQL and MySQL**

Introduction

- 1 minute

Database design is an important aspect of database performance, even though it's not always under the control of the database administrator. You may be working with third- party vendor applications that you did not build. Whenever possible, it's important to design your database properly for the workload, whether it's an online transaction processing (OLTP) or data warehouse workload. Many design decisions, such as choosing the right datatypes, can make large differences in the performance of your databases.

Describe normalization

- 13 minutes

Database normalization is a design process used to organize a given set of data into tables and columns in a database. Each table should contain data relating to a specific 'thing' and only have data that supports that same 'thing' included in the table. The goal of this process is to reduce duplicate data contained within your database, to reduce the performance impact of database inserts and updates.

For example, a customer address change is much easier to implement if the only place the customer address is stored in the Customers table. The most common forms of normalization are first, second, and third normal form and are described below.

First normal form

First normal form has the following specifications:

- Create a separate table for each set of related data
- Eliminate repeating groups in individual tables
- Identify each set of related data with a primary key

In this model, you should not use multiple columns in a single table to store similar data. For example, if product can come in multiple colors, you should not have multiple columns in a single row containing the different color values. The first table, below (ProductColors), is not in first normal form as there are

repeating values for color. For products with only one color, there is wasted space. And what if a product came in more than three colors? Rather than having to set a maximum number of colors, we can recreate the table as shown in the second table, ProductColor. We also have a requirement for first normal form that there is a unique key for the table, which is column (or columns) whose value uniquely identifies the row. Neither of the columns in the second table is unique, but together, the combination of ProductID and Color is unique. When multiple columns are needed, we call that a composite key.

1. What type of database design should you use for a data warehouse when you want to reduce the data volume of your dimensions?

- (●) Snowflake schema ✓
  That's correct. Snowflake schema would reduce the data volume.
- ( ) Star Schema
- ( ) 3rd normal form

2. What is the minimum number of rows you need to bulk insert into a columnstore index?

- (●) 102,400 ✓
  That's correct. 102,400 is the minimum number of rows to bulk insert into a columnstore index, fewer rows will use a normal insert operation.
- ( ) 1,000,000
- ( ) 1000

3. Which compression type offers the highest level of compression?

- (●) Columnstore Archival ✓
  That's correct. Columnstore archival provides the highest level of compression in SQL Server.
- ( ) Page Compression
- ( ) Row Compression

1. Which type of execution plan is stored in the plan cache?

- [x] Estimated execution plan

That's correct. The estimated execution plan is stored in the plan cache.

- [ ] Actual execution plan
- [ ] Live Query Stats

2. Which DMV should you use to find index utilization?

- [x] sys.dm_db_index_usage_stats

That's correct. sys.dm_db.index_usage_states shows the read and write operations against each index.

- [ ] sys.dm_db_missing_index_details
- [ ] sys.dm_exec_query_plan_stats

3. Which of the following wait types would indicate excessive CPU consumption?

- [x] SOS_SCHEDULER_YIELD

That's correct. The SOS_SCHEDULER_YIELD wait is the only one of these wait types that is associated with CPU.

- [ ] RESOURCE_SEMAPHORE
- [ ] PAGEIOLATCH_SH

1. What type of index is best used on a data warehouse fact table?

- [x] Clustered Columnstore

That's correct. A clustered columnstore index will provide the best performance for a data warehouse fact table.

- [ ] Nonclustered Columnstore
- [ ] Clustered b-tree

2. Which DMV provides information about server level wait statistics?

- [ ] sys.dm_db_index_physical_stats
- [x] sys.dm_os_wait_stats

That's correct. This shows the wait stats across the server.

- [ ] sys.dm_exec_session_wait_stats

3. Which DMV can you use to capture the last Actual Execution Plan for a given query?

- [ ] sys.dm_exec_cached_plans
- [ ] sys.dm_exec_query_plan
- [x] sys.dm_exec_query_plan_stats

That's correct. In SQL Server 2019 you can query sys.dm_exec_query_plan_stats to get the last actual execution plan for a query.