Introduction

3 minutes

Meet Tailwind Traders

The Tailwind Trader's CTO asks, "What is our storage solution for non-relational data?" The CTO's question is a reasonable and an area Azure Architects can help. In this module, we'll explore different storage strategies. Storage strategies will include data types, storage accounts, blob storage, file storage, disk storage, storage security, and data protection.

Learning objectives

In this module, you'll learn how to:

- Design for data storage.
- Design for Azure storage accounts.
- Design for Azure blob storage.
- Design for Azure files.
- Design an Azure disk solution.
- Design for storage security.

Skills measured

The content in the module will help you prepare for Exam AZ-305: Designing Microsoft Azure Infrastructure Solutions. The module concepts are covered in:

Design a data storage solution for non-relational data

- Recommend a data storage solution to balance features, performance, and cost

- Design a data solution for protection and durability
- Recommend access control solutions to data storage

Prerequisites

- Conceptual knowledge of storage accounts, blobs, files, disks, and data protection.
- Working experience with creating and securing storage systems.

Next unit: Design for data storage

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

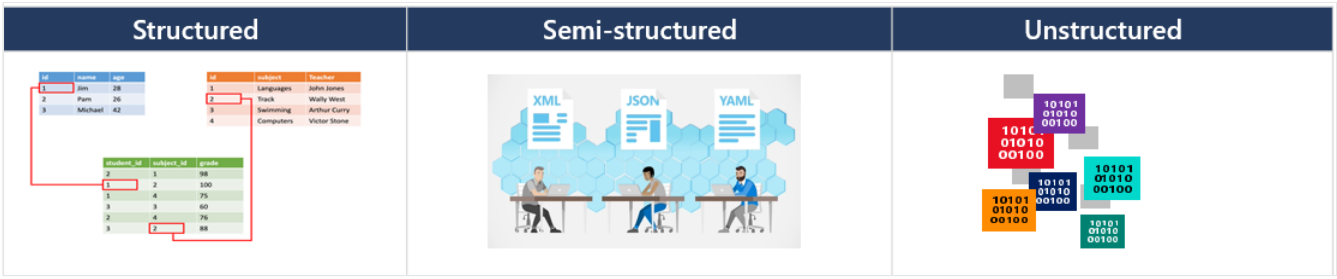
100 XP

Design for data storage

3 minutes

Data classifications – structured, semi-structured, and unstructured

To design Azure storage, you first must determine what type of data you have.



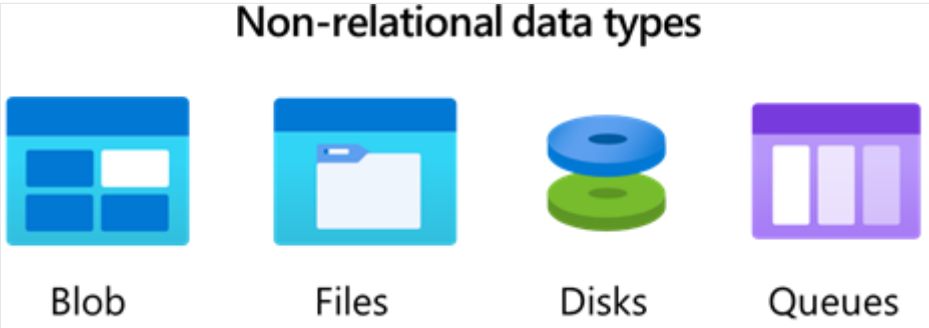
- **Structured data** includes relational data and has a shared schema. Structured data is often stored in database tables with rows, columns, and keys. Structured data is often used for application storage like an ecommerce website.
- **Semi-structured** is less organized than structured data and isn't stored in a relational format. Semi-structured data fields don't neatly fit into tables, rows, and columns. Semi-structured data contains tags that clarify how the data is organized. The expression and structure of the data in this style is defined by a serialization language. Examples of semi-structured data include Hypertext Markup Language (HTML) files, JavaScript Object Notation (JSON) files, and Extensible Markup Language (XML) files
- **Unstructured data** is the least organized type of data. The organization of unstructured data is ambiguous. Examples of unstructured data include:
  - Media files, such as photos, videos, and audio files
  - Office files, such as Word documents
  - Text files

Important

This module will only cover unstructured data. These data types are often referred to as non-relational data.

Azure non-relational storage objects

In Azure, non-relational data is contained in several different storage data objects. There are four data storage objects we'll focus on.



Azure Blob storage is an object store used for storing vast amounts of unstructured data. Blob stands for Binary Large Object, which includes objects such as images and multimedia files.

Azure Files is a shared storage service. You can access files with Server Message Block (SMB) on Windows or Network File Share (NFS) on Linux.

Azure managed disks are block-level storage volumes that are managed by Azure and used with Azure virtual machines. Managed disks are like a physical disk in an on-premises server but, virtualized.

Azure Queue Storage is a service for storing large numbers of messages. Queues are commonly used to create a backlog of work to process asynchronously.

> **Tip**
>
> Before beginning your study, think about which non-relational data types are of most interest to you or your organization.

Next unit: Design for Azure storage accounts

Continue >

100 XP

Design for Azure storage accounts

3 minutes

Once you've determined your data storage requirements, you need to create storage accounts. An Azure storage account group together all the Azure Storage services you need. The storage account provides a unique namespace that's accessible from anywhere (assuming you have the correct permissions) in the world over HTTPS. Data in your storage account is durable and highly available, secure, and massively scalable.

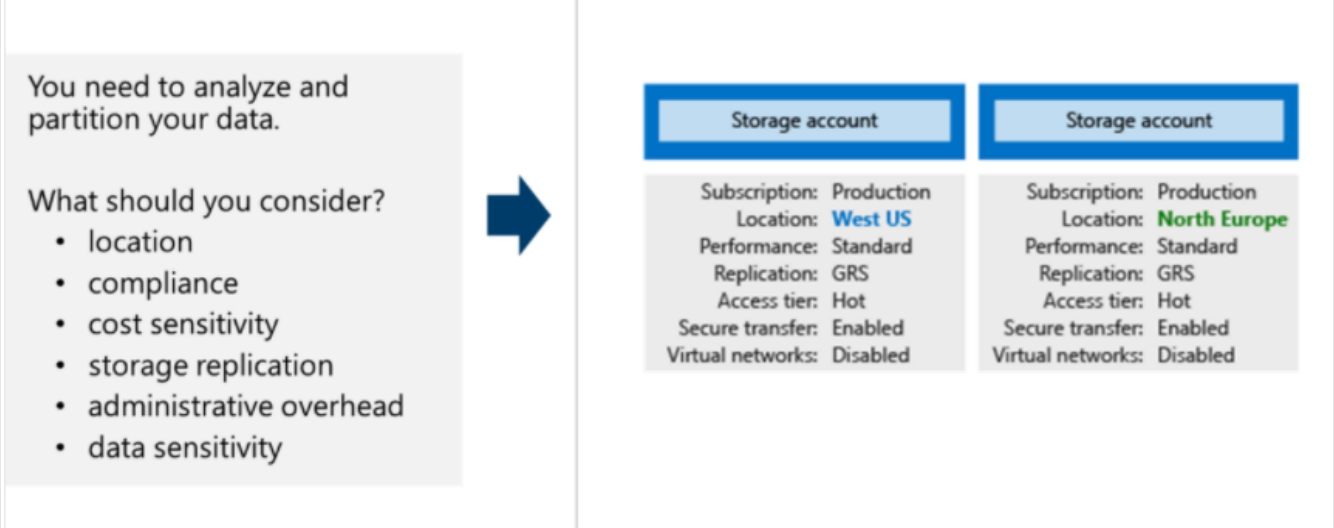
Select the appropriate storage account kind

Azure Storage offers several kinds of storage accounts. Each kind supports different features and has its own pricing model. Consider these differences to determine the kind of account that is best for your applications. The types of storage accounts are:

Storage Account	Supported Services	Recommended usage
Standard general-purpose v2	Blob (including Data Lake Storage), Queue, and Table storage, Azure Files	Supports all the storage services: Blob, Azure Files, Queue, Disk (Page Blob), and Table.
Premium block blobs	Blob storage (including Data Lake Storage)	Premium block blobs are ideal for applications that require high transaction rates. Also ideal for situations that use smaller objects or require consistently low storage latency. This storage is designed to scale with your applications.
Premium file shares	Azure Files	Recommended for enterprise or high-performance scale applications. Use Premium file shares if you need a storage account that supports both SMB and NFS file shares.
Premium page blobs	Page blobs only	Premium high-performance page blob scenarios. Page blobs are ideal for storing index-based and sparse data structures like OS and data disks for virtual machines and databases.

Determine the number of storage accounts

A storage account represents a collection of settings like location, replication strategy, and subscription owner. Organizations often have multiple storage accounts so they can implement different sets of requirements. The following illustration shows two storage accounts that differ in one setting. That one difference is enough to require separate storage accounts.



Considerations when deciding how many storage accounts to create

- **Location**. Do you have data that is specific to a country or region? For performance reasons, you might want to locate the data close to your users. You may need one storage account for each location.
- **Compliance**. Does your company have regulatory guidelines for keeping data in a specific location? Does your company have internal requirements for auditing or storing data?
- **Cost**. A storage account by itself has no financial cost; however, the settings you choose for the account do influence the cost of services in the account. Geo-redundant storage costs more than locally redundant storage. Premium performance and the hot access tier increase the cost of blobs. Do you need to keep track of expenses or billing by department or project? Are you working with partners where storage costs need to be separated?
- **Replication**. Does your data storage have different replication strategies? For example, you could partition your data into critical and non-critical categories. You could place

your critical data into a storage account with geo-redundant storage. You could put your non-critical data in a different storage account with locally redundant storage.

- **Administrative overhead**. Each storage account requires some time and attention from an administrator to create and maintain. It also increases complexity for anyone who adds data to your cloud storage. Everyone in this role needs to understand the purpose of each storage account so they add new data to the correct account.
- **Data sensitivity**. Do you have some data that is proprietary and some for public consumption? If so, you could enable virtual networks for the proprietary data and not for the public data. This may require separate storage accounts.
- **Data isolation**. Regulatory, compliance, or local policies may require data to be segregated. Perhaps data from one application should be separated from data in another application?

Tip

Take a few minutes to think about your organization's storage accounts. Are the storage accounts already in place? Would you make any changes? What type of storage accounts will you need, and why?

Next unit: Design for data redundancy

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design for data redundancy

3 minutes

Azure Storage always stores multiple copies of your data. This redundancy ensures the data is protected from planned and unplanned events. These events can include transient hardware failures, network or power outages, and massive natural disasters. Storage redundancy ensures that your storage account meets its availability and durability targets.

When deciding which redundancy option is best for your scenario, consider the tradeoffs between lower costs and higher availability. The factors that help determine which redundancy option you should choose include:

• How your data is replicated in the primary region.

• Whether your data is replicated to a second region. The secondary region is geographically distant to the primary region. This helps to protect against regional disasters.

• Whether your application requires read access to the replicated data in the secondary region if the primary region becomes unavailable for any reason.

Redundancy in the primary region

Azure Storage offers two options for how your data is replicated in the primary region.

Locally redundant storage (LRS) is the lowest-cost redundancy option and offers the least durability compared to other options. LRS protects your data against server rack and drive failures. However, if the data center fails, all replicas of a storage account using LRS may be lost or unrecoverable. LRS is a good choice for the following scenarios:

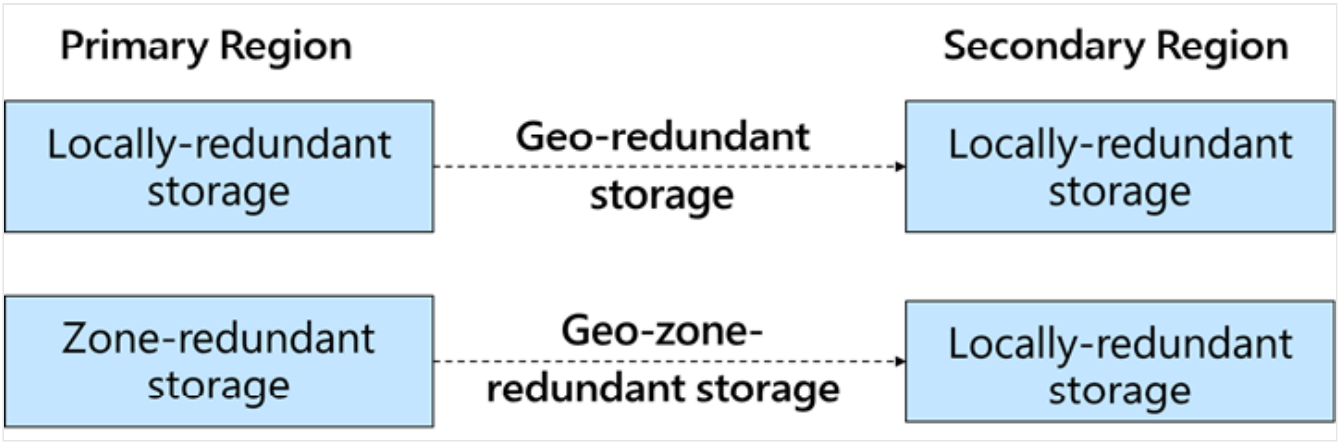
• If your application stores data that can be easily reconstructed if data loss occurs.

• If your application is restricted to replicating data only within a country or region due to data governance requirements.

Zone-redundant storage (ZRS) replicates synchronously across three Azure availability zones in the primary region. With ZRS, your data is still accessible for both read and write operations even if a zone becomes unavailable. ZRS provides excellent performance, low latency, and resiliency for your data if it becomes temporarily unavailable. However, ZRS by itself may not protect your data against a regional disaster where multiple zones are permanently affected.

Redundancy in a secondary region

For applications requiring high durability, you can choose to copy the data in your storage account to a secondary region. When you create a storage account, you select the primary region for the account. The paired secondary region is determined based on the primary region and can't be changed. Azure Storage offers two options for copying your data to a secondary region: Geo-redundant storage (GRS), and Geo-zone-redundant storage (GZRS).



• The primary difference between GRS and GZRS is how data is replicated in the primary region. Within the secondary region, data is always replicated synchronously with LRS.

• If the primary region becomes unavailable, you can choose to fail over to the secondary region. After the failover has completed, the secondary region becomes the primary region, and you can again read and write data.

• Data is replicated to the secondary region asynchronously A failure that affects the primary region may result in data loss if the primary region cannot be recovered.

• With GRS or GZRS, the data in the secondary region isn't available for read or write access unless there is a failover to the secondary region. For read access to the secondary region, configure your storage account to use read-access geo-redundant storage (RA-GRS) or read-access geo-zone-redundant storage (RA-GZRS).

Tip

If your storage account is configured for read access to the secondary region, then you can design your applications to seamlessly shift to reading data from the secondary region if the primary region becomes unavailable for any reason.

Next unit: Design for Azure blob storage

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design for Azure blob storage

3 minutes

When designing blob storage, there are two main areas we'll highlight. The first is determining the blob access tier. Blob storage type affects storage availability, latency, and cost. The second area is deciding if immutable storage is needed.

Availability, Latency, and Cost	Immutable Storage
Premium blob storage	Legal hold policies
Hot, cool, and archive access tiers	Time-based retention policies

Determine the Azure blob access tier

Optimize storage costs by placing your data in the appropriate access tier.

Feature	Premium	Hot tier	Cool tier	Archive tier
Availability	99.9%	99.9%	99%	Offline
Availability (RA-GRS reads)	N/A	99.99%	99.9%	Offline
Usage charges	Higher storage costs, lower access, and transaction cost	Higher storage costs, lower access, and transaction costs	Lower storage costs, higher access, and transaction costs	Lowest storage costs, highest access, and transaction costs
Minimum storage duration	N/A	N/A	30 days	180 days

Feature	Premium	Hot tier	Cool tier	Archive tier
Latency (time to first byte)	Single-digit milliseconds	milliseconds	milliseconds	hours

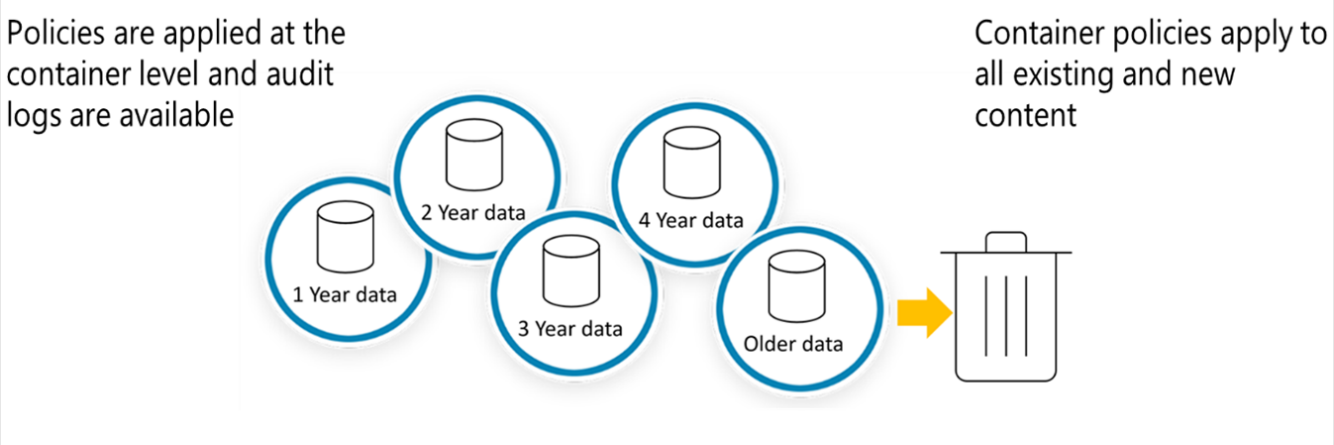
- **Premium blob storage.** The premium blob storage account types are best suited for I/O intensive workloads that require low and consistent storage latency. Premium blob storage uses solid-state drives (SSDs) for fast and consistent response times. This storage is best for workloads that perform many small transactions. An example would be a mapping app that requires frequent and fast updates.
- **Standard Hot access tier**. By default, new storage accounts are created in the hot access tier. The hot tier is optimized for frequent reads and writes of objects in the storage account. The hot tier has higher storage costs than cool and archive tiers, but the lowest access costs. A good usage case is data that is actively being processed.
- **Standard Cool access tier**. The cool access tier is optimized for storing large amounts of data that is infrequently accessed. This tier is intended for data that will remain in the cool tier for at least 30 days. The cool access tier has lower storage costs and higher access costs compared to hot storage. A usage case for the cool access tier is short-term backup and disaster recovery datasets and older media content. This content wouldn't be viewed frequently but must be available immediately.
- **Standard Archive access tier**. The archive access tier is optimized for data that can tolerate several hours of retrieval latency. Data must remain in the archive tier for at least 180 days or be subject to an early deletion charge. The archive tier is the most cost-effective option for storing data. But, accessing that data is more expensive than accessing data in the other tiers. Data for the archive tier includes secondary backups, original raw data, and legally required compliance information.

> **Tip**
>
> Take a few minutes to determine the data sets and access tiers your organization will need. How did you decide on the access tier? How will you manage the data sets?

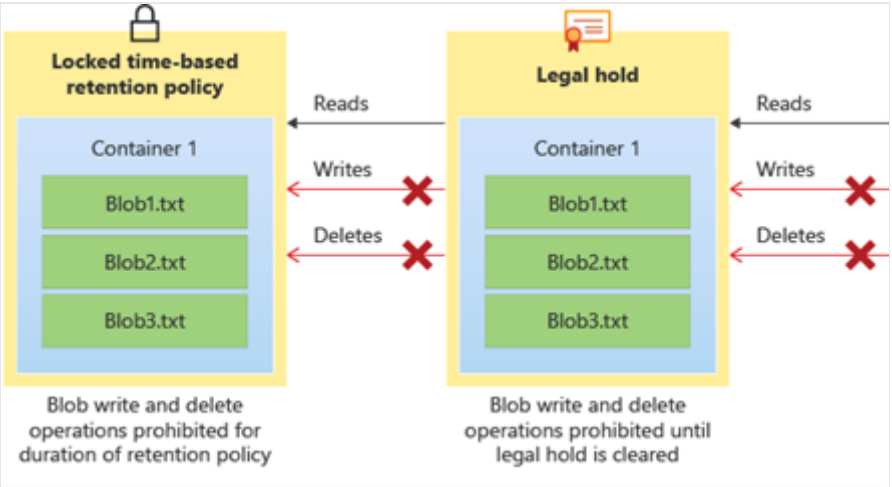
Determine requirements for Azure blob immutable storage

Immutable storage for Azure Blob Storage enables users to store business-critical data in a WORM (Write Once, Read Many) state. While in a WORM state, data can't be modified or

deleted for a user-specified interval. By configuring immutability policies for blob data, you can protect your data from overwrites and deletes. Policies are applied at the container level and audit logs are available.



The next diagram shows how time-based retention policies and legal holds prevent write and delete operations.



Immutable storage for Azure Blob storage supports two types of immutability policies.

- **Time-based retention policies**: With a time-based retention policy, users can set policies to store data for a specified interval. When a time-based retention policy is in place, objects can be created and read, but not modified or deleted. After the retention period has expired, objects can be deleted but not overwritten.
- **Legal hold policies**: A legal hold stores immutable data until the legal hold is explicitly cleared. When a legal hold is set, objects can be created and read, but not modified or deleted.

> **Tip**
>
> Take a few minutes to determine if your organization will need immutable blob storage policies. Which data sets and policies would be most helpful?

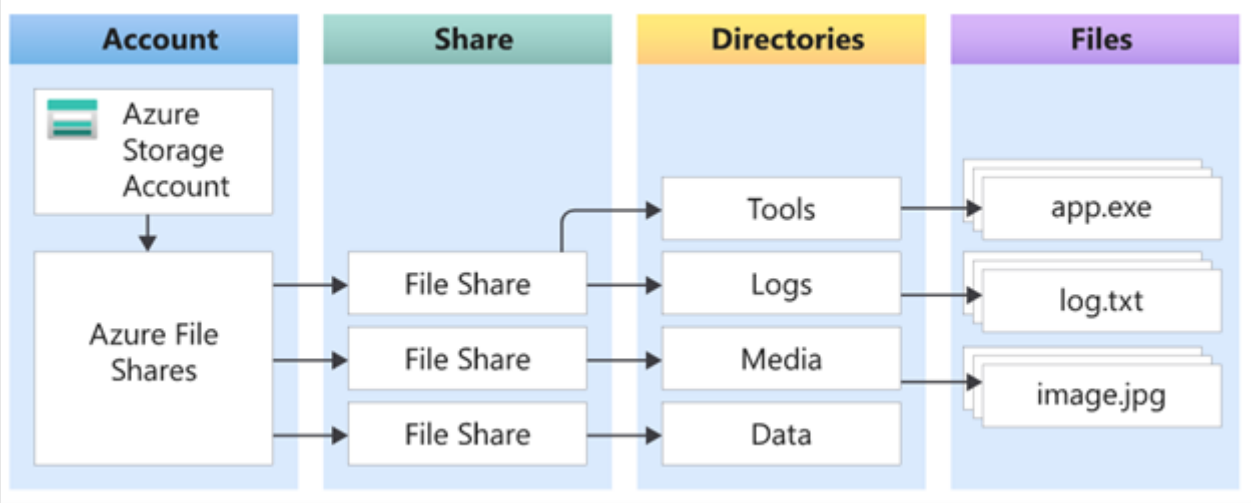
100 XP

Design for Azure Files

3 minutes

Moving a company's shared files into the cloud-based Azure Files requires an analysis of the options and a plan for the implementation. There's an important decision to make. How are you going to access and update the files? You could choose to replace your existing Server Message Block (SMB) file shares with their equivalent in Azure Files. The other option is to set up an instance of Azure File Sync. If you choose to use Azure File Sync, there's more flexibility on how files are secured and accessed.

What are Azure Files?



You can think of Azure Files as a standard file share, hosted on Azure, that you can access with the industry standard SMB/CIFS protocol. You can mount or connect to an Azure file share at the same time on all the main operating systems.

Azure Files can be used to add to or replace a company's existing on-premises NAS devices or file servers. Some reasons why your organization will want to use Azure Files are:

- Developers can store apps and configuration files in a file share and connect new VMs to the shared files. This action reduces the time to get new machines into production.
- With file shares on Azure, a company doesn't need to buy and deploy expensive redundant hardware and manage software updates. The shares are cross-platform, and you can connect to them from Windows, Linux, or macOS.

- All the resilience of the Azure platform is inherited by your file share, which makes files globally redundant. You also gain options to use the integrated snapshots feature and set up automatic backups by using Recovery Services vaults.
- All the data is encrypted in transit by using HTTPS and is stored encrypted when at rest.

Choose your data access method

Azure file shares can be used in two ways: by directly mounting these serverless Azure file shares (SMB) or by caching Azure file shares on-premises using Azure File Sync.

- **Direct mount of an Azure file share**: Since Azure Files provides SMB access, you can mount Azure file shares on-premises or in the cloud, Mounting uses the standard SMB client available in Windows, macOS, and Linux. Because Azure file shares are serverless, deploying for production scenarios doesn't require managing a file server or NAS device. Direct mounting means you don't have to apply software patches or swap out physical disks.
- **Cache Azure file share on-premises with Azure File Sync**: Azure File Sync lets you centralize your organization's file shares. Azure Files provides the flexibility, performance, and compatibility of an on-premises file server. Azure File Sync transforms an on-premises (or cloud) Windows Server into a quick cache of your Azure file share.

Choose your performance level

Because Azure Files stores files in a storage account, you can choose between standard or premium performance storage accounts.

Performance level	Latency	IOPS	Bandwidth
Standard	Double-digit ms	10,000 IOPS	300-MBps
Premium	Single-digit ms	100,000 IOPS	5-GBps

Standard performance accounts use HDD to store data. With HDD, the costs are lower but so is the performance. SSD arrays back the premium storage account's performance, which comes with higher costs. Currently, premium accounts can only use file storage accounts with ZRS storage in a limited number of regions.

Determine your storage tier

Azure Files offers four different tiers of storage, premium, transaction optimized, hot, and cool. These tiers allow you to tailor your shares to the performance and price requirements of your scenario.

Storage tier	Usage
Premium	File shares are backed by solid-state drives (SSDs) and provide consistent high performance and low latency. Used for the most intensive IO workloads. Suitable workloads include databases, web site hosting, and development environments. Can be used with both Server Message Block (SMB) and Network File System (NFS) protocols.
Transaction optimized	Used for transaction heavy workloads that don't need the latency offered by premium file shares. File shares are offered on the standard storage hardware backed by hard disk drives (HDDs).
Hot	Storage optimized for general purpose file sharing scenarios such as team shares. Offered on standard storage hardware backed by HDDs.
Cool	Cost-efficient storage optimized for online archive storage scenarios. Offered on storage hardware backed by HDDs.

When to use Azure files instead of Azure blobs or Azure NetApp Files

Let’s take a minute to review when you should select Azure blob storage or Azure NetApp Files instead of Azure file storage.

NetApp Files is a fully managed, highly available, enterprise-grade NAS service. NetApp Files can handle the most demanding, high-performance, low-latency workloads. It enables the migration of workloads, which are deemed "un-migratable" without.

Your decision on which technology depends on the use case, protocol, and performance required.

Category	Azure Blob Storage	Azure Files	Azure NetApp Files

Category	Azure Blob Storage	Azure Files	Azure NetApp Files
Use cases	Blob Storage is best suited for large scale read-heavy sequential access workloads where data is ingested once and modified later. Blob Storage offers the lowest total cost of ownership, if there is little or no maintenance. Some example scenarios are: Large scale analytical data, throughput sensitive high-performance computing, backup and archive, autonomous driving, media rendering, or genomic sequencing.	Azure Files is a highly available service best suited for random access workloads. For NFS shares, Azure Files provides full POSIX file system support and can easily be used from container platforms like Azure Container Instance (ACI) and Azure Kubernetes Service (AKS) with the built-in CSI driver, in addition to VM-based platforms. Some example scenarios are: Shared files, databases, home directories, traditional applications, ERP, CMS, NAS migrations that don't require advanced management, and custom applications requiring scale-out file storage.	Fully managed file service in the cloud, powered by NetApp, with advanced management capabilities. NetApp Files is suited for workloads that require random access and provides broad protocol support and data protection capabilities. Some example scenarios are: On-premises enterprise NAS migration that requires rich management capabilities, latency sensitive workloads like SAP HANA, latency-sensitive or IOPS intensive high performance compute, or workloads that require simultaneous multi-protocol access.
Available protocols	NFS 3.0 REST Data Lake Storage Gen2	SMB NFS 4.1 REST	NFS 3.0 and 4.1 SMB
Performance (Per volume)	Up to 20,000 IOPS, up to 100 GiB/s throughput.	Up to 100,000 IOPS, up to 80 Gib/s throughput.	Up to 460,000 IOPS, up to 36 Gib/s throughput.

Tip

Take a few minutes to think through your company file share strategy. Will you need files shares? Will you need file sync?

Next unit: Design for Azure disk solutions

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design for Azure disk solutions

3 minutes

For this module, we'll concern ourselves with data disks. Data disks are used by virtual machines to store data. For example, database files, website static content, or custom application code would be stored on data disks. The number of data disks you can add depends on the virtual machine size. Each data disk has a maximum capacity of 32,767 GB.

Tip

Microsoft recommends always using managed disks. With **managed disks**, you specify the disk size, the disk type, and provision the disk. Once you provision the disk, Azure handles the rest.

Determine the type of data disk

Azure offers several types of data disks. When selecting a disk type, consider your scenario, throughput, and IOPS. The following table provides a comparison of the four disk types.

Detail	Ultra-disk	Premium SSD	Standard SSD	Standard HDD
Disk type	SSD	SSD	SSD	HDD
Scenario	IO-intensive workloads such as SAP HANA, top tier databases (for example, SQL, Oracle), and other transaction-heavy workloads.	Production and performance sensitive workloads	Web servers, lightly used enterprise applications and dev/test	Backup, non-critical, infrequent access
Max throughput	2,000 MB/s	900 MB/s	750 MB/s	500 MB/s

Detail	Ultra-disk	Premium SSD	Standard SSD	Standard HDD
Max IOPS	160,000	20,000	6,000	2,000

- **Ultra-disk storage**. Azure Ultra Disk storage provides the best performance. Choose this option when you need the fastest storage performance in addition to high throughput, high input/output operations per second (IOPS), and low latency. Ultra-disk storage may not be available in all regions.
- **Premium SSD storage**. Azure Premium SSD-managed disks provide high throughput and IOPS with low latency. These disks offer a slightly less performance compared to Ultra Disk Storage. Premium SSD storage is available in all regions.
- **Standard SSD.** Azure Standard SSD-managed disks are a cost-effective storage option for VMs that need consistent performance at lower speeds. Standard SSD disks aren't as fast as Premium SSD disks or Ultra Disk Storage. You can attach Standard SSD disks to any VM.
- **Standard HDD.** In Azure Standard HDD-managed disks, data is stored on conventional magnetic disk drives that have moving spindles. Disks are slower and the variation in speeds is higher compared to solid-state drives (SSDs). Like Standard SSD disks, you can use Standard HDD disks for any VM.

> **Tip**
>
> Read more about how to **Select a disk type for Azure IaaS VMs - managed disks - Azure Virtual Machines | Microsoft Docs**.

Improve performance with disk caching

Azure virtual machine disk caching is about optimizing read and write access to the virtual hard disk (VHD) files. These VHDs are attached to Azure virtual machines. Here are the recommended disk cache settings for data disks.

Disk caching setting	Recommendation
None	Use for write-only and write-heavy disks.

Disk caching setting	Recommendation
Read only	Use for read-only and read-write disks. Provides low read latency and high read IOPS and throughput.
Read & write	Use only if your application properly handles writing cached data to persistent disks.

Warning: Disk Caching isn't supported for disks 4 TiB and larger. When multiple disks are attached to your VM, each disk that is smaller than 4 TiB will support caching. Changing the cache setting of an Azure disk detaches and reattaches the target disk. When it's the operating system disk, the VM is restarted.

Secure your data disks with encryption

There are several encryption types available for your managed disks. Encryption types includes Azure Disk Encryption (ADE), Server-Side Encryption (SSE) and encryption at host.

- Azure Disk Encryption ADE encrypts the virtual machine's virtual hard disks (VHDs). If VHD is protected with ADE, the disk image will only be accessible by the virtual machine that owns the disk.
- Server-Side Encryption (also referred to as encryption-at-rest or Azure Storage encryption) is performed on the physical disks in the data center. If someone directly accesses the physical disk, the data will be encrypted. When the data is accessed from the disk, it's decrypted and loaded into memory.
- Encryption at host ensures that data stored on the VM host is encrypted at rest and flows encrypted to the Storage service. Disks with encryption at host enabled aren't encrypted with SSE. Instead, the server hosting your VM provides the encryption for your data, and that encrypted data flows into Azure Storage.

> **Note**
>
> To fully protect your data disks, combine encryption services.

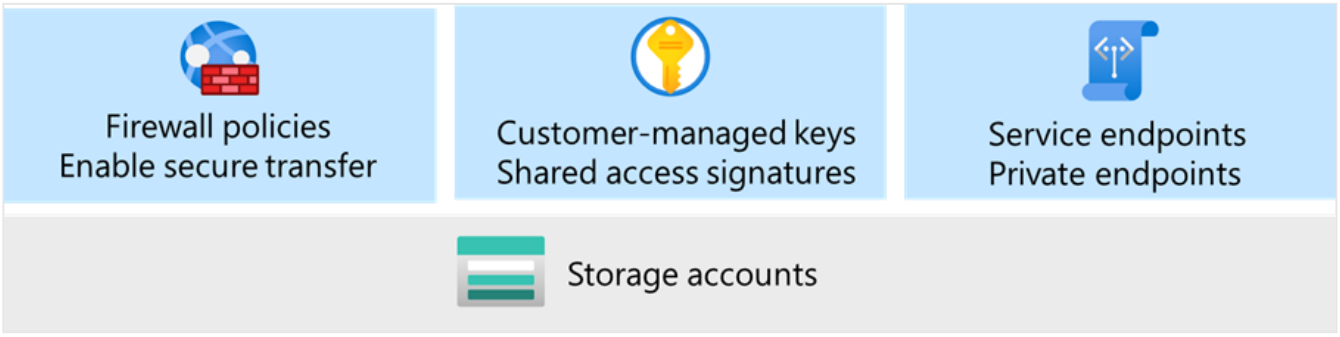
Next unit: Design for storage security

100 XP

Design for storage security

3 minutes

Azure Storage provides a layered security model. This model enables you to secure and control the level of access to your storage accounts. In this unit, we'll cover some best practices for storage security.



Grant limited access to Azure Storage resources

The Azure security baseline for Azure Storage baseline provides a comprehensive list of ways to secure your Azure storage.

Use Shared Access Signatures

One of the most common ways is to use a Shared Access Signature. A SAS provides secure delegated access to resources in your storage account. With a SAS, you have granular control over how a client can access your data. For example:

A shared access signature (SAS) provides secure delegated access to resources in your storage account. With a SAS, you have granular control over how a client can access your data. For example:

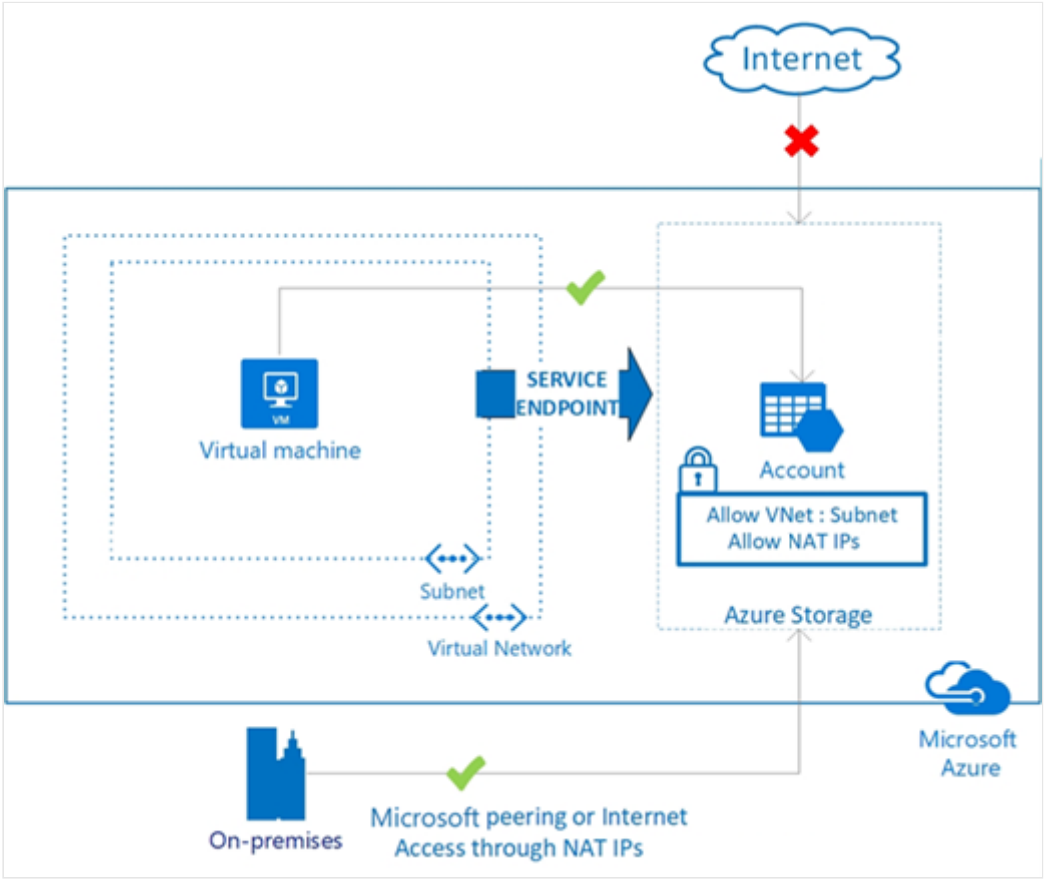
- What resources the client may access.
- What permissions they have to those resources.
- How long the SAS is valid.

Enable firewall policies and rules

- Configure firewall rules to limit access to your storage account. Requests can be limited to specific IP addresses or ranges, or to a list of subnets in an Azure virtual network. The Azure storage firewall provides access control for the public endpoint of your storage account. You can also use the firewall to block all access through the public endpoint when using private endpoints. Your storage firewall configuration also enables select trusted Azure platform services to access the storage account securely.

Restrict network access using service endpoints

Use virtual network service endpoints to provide direct connection to your Azure storage.

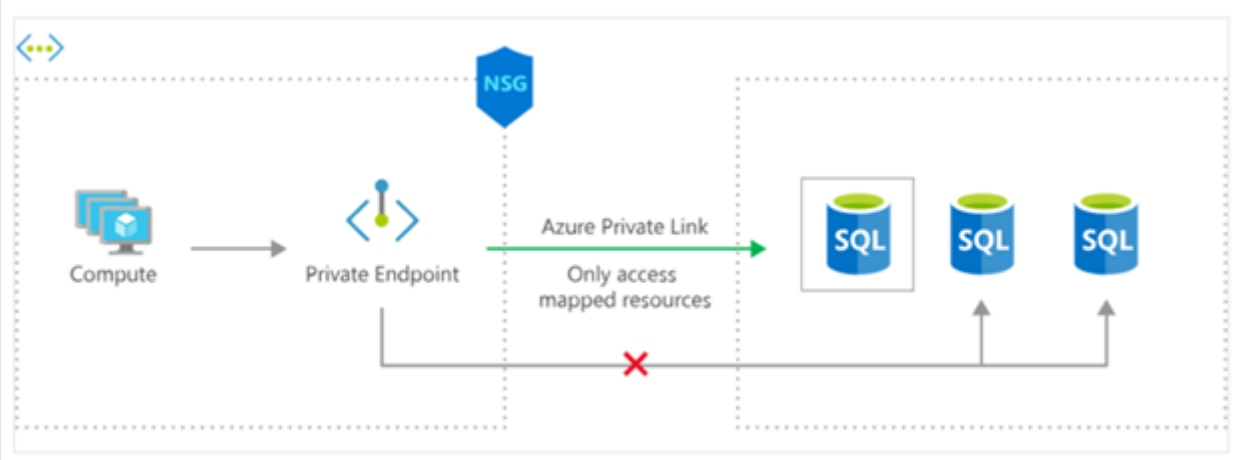


Service endpoints provide several advantages.

- Allows you to secure Azure storage accounts to your virtual networks.
- Provides optimal routing by always keeping traffic destined to Azure Storage on the Azure backbone network.
- Service Endpoints enable private IP addresses in the VNet to reach the service endpoint. A public IP address isn't needed.
- Enables on-premises networks to access resources using NAT IP addresses.

Determine when to use private endpoints

A private endpoint is a special network interface for an Azure service in your virtual network. When you create a private endpoint for your storage account, it provides secure connectivity between clients on your VNet and your storage.



Enable secure transfer

You can configure your storage account to accept requests from secure connections. This is done by setting the Secure transfer required property for the storage account. When you require secure transfer, any requests originating from non-secure connections are rejected. Microsoft recommends that you always require secure transfer for all your storage accounts.

Use customer-managed encryption keys

Data in your storage account is automatically encrypted. Azure Storage encryption offers two options for managing encryption keys at the level of the storage account:

- **Microsoft-managed keys.** By default, Microsoft manages the keys used to encrypt your storage account.
- **Customer-managed keys.** You can optionally choose to manage encryption keys for your storage account. Customer-managed keys must be stored in Azure Key Vault. When you bring-your-own-key (BYOK), you gain full control over who can use the encryption keys and who can access the encrypted data.

Next unit: Knowledge check

Continue >

200 XP

Knowledge check

3 minutes

Tailwind Traders wants to reduce storage costs by reducing duplicate content and, whenever applicable, migrating it to the cloud. The company would like a solution that centralizes maintenance while still providing nation-wide access for customers. Customers should be able to browse and purchase items online even in a case of a failure affecting an entire Azure region. Here are some specific requirements.

- **Warranty document retention.** The company's risk and legal teams requires warranty documents be kept for three years.
- **New photos and videos.** The company would like each product to have a photo or video to demonstrate the product features.
- **External vendor development.** A vendor will create and develop some of the online ecommerce features. The developer will need access to the HTML files, but only during the development phase.
- **Product catalog updates.** The product catalog is updated every few months. Older versions of the catalog aren't viewed frequently but must be available immediately if accessed.

1. What is the best way for Tailwind Traders to protect their warranty information?

- ○ Legal hold policy
- ◉ Time-based retention policy

  ✔ **That's correct. With a time-based retention policy, users can set policies to store data for a specified interval. When a time-based retention policy is in place, objects can be created and read, but not modified or deleted.**
- ○ Private endpoint for storage account

2. What type of storage should Tailwind Traders use for their photos and videos?

- ○ Blob storage

✔ That's correct. Blob storage is best for their photos.

- ◉ File storage

  ✘ That's incorrect. Photos could be uploaded to file storage, but blob storage would be more appropriate.

- ○ Disk storage

3. What is the best way to provide the developer access to the ecommerce HTML files?

- ○ Enable secure transfer
- ◉ Enable firewall policies and rules

  ✘ That's incorrect. Firewall policies and rules are for network access.

- ○ Shared access signatures

  ✔ That's correct. Shared access signatures provide secure delegated access. This functionality can be used to define permissions and how long access is allowed.

4. Which access tier should be used for the older versions of the product catalog?

- ○ Hot access tier
- ○ Cool access tier

  ✔ That's correct. The cool access tier is for content that wouldn't be viewed frequently but must be available immediately if accessed.

- ◉ Archive access tier.

  ✘ That's incorrect. The older catalogs are accessed although infrequently. There's an early deletion charge if data is accessed sooner than 180 days.

Next unit: Summary and resources

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Introduction

3 minutes

Many organizations have an aging or under-engineered data platform strategy. There's been a significant trend of moving existing systems to the cloud, building new applications quickly with the cloud, and offloading some on-premises costs. You need a plan for how to move data workloads to the cloud. And you need to understand how to set up your organization for success.

Meet Tailwind Traders

Tailwind Traders is a fictitious home improvement retailer. It operates retail hardware stores across the globe and online. It currently manages an on-premises datacenter that hosts the company's retail website. The datacenter also stores all of the data and streaming video for its applications. The on premises SQL server also provides storage for customer data, order history, and product catalogs, apart from data storage for your internal-only training portal website.

Tailwind Traders wants to effectively manage database needs by migrating it to the cloud. You are tasked with finding a cost efficient database solution that provides low latency and high availability.

The Tailwind Trader's CTO asks, "What is our storage solution for relational data?" The CTO's question is a reasonable and an area Azure Architects can help. In this module, we'll explore different storage solutions that solve different types of problems. Storage solutions will include Azure SQL Database, Azure SQL Managed Instance, SQL Server in an Azure virtual machine, SQL edge, Azure table storage and Cosmos DB. You will also learn how to design your solution with data encryption.

Learning objectives

In this module, you'll be able to:

- Design for Azure SQL Database
- Design for Azure SQL Managed Instance
- Design for SQL Server on Azure VM
- Recommend a solution for Database Scalability
- Design encryption for data at rest, data in transmission, and data in use
- Design for Azure SQL Edge
- Design for Azure tables.
- Design for Azure Cosmos DB.

Skills measured

The content in the module will help you prepare for Exam AZ-305: Designing Microsoft Azure Infrastructure Solutions. The module concepts are covered in:

Design a Data Storage Solution for Relational Data

- Recommend database service tier sizing
- Recommend a solution for database scalability
- Recommend a solution for encrypting data at rest, data in transmission, and data in use

Recommend a Data Storage Solution

- Recommend a solution for storing relational data

Prerequisites

- Working experience with database solutions
- Conceptual knowledge of SQL Server

Next unit: Design for Azure SQL Database

Design for Azure SQL Database

3 minutes

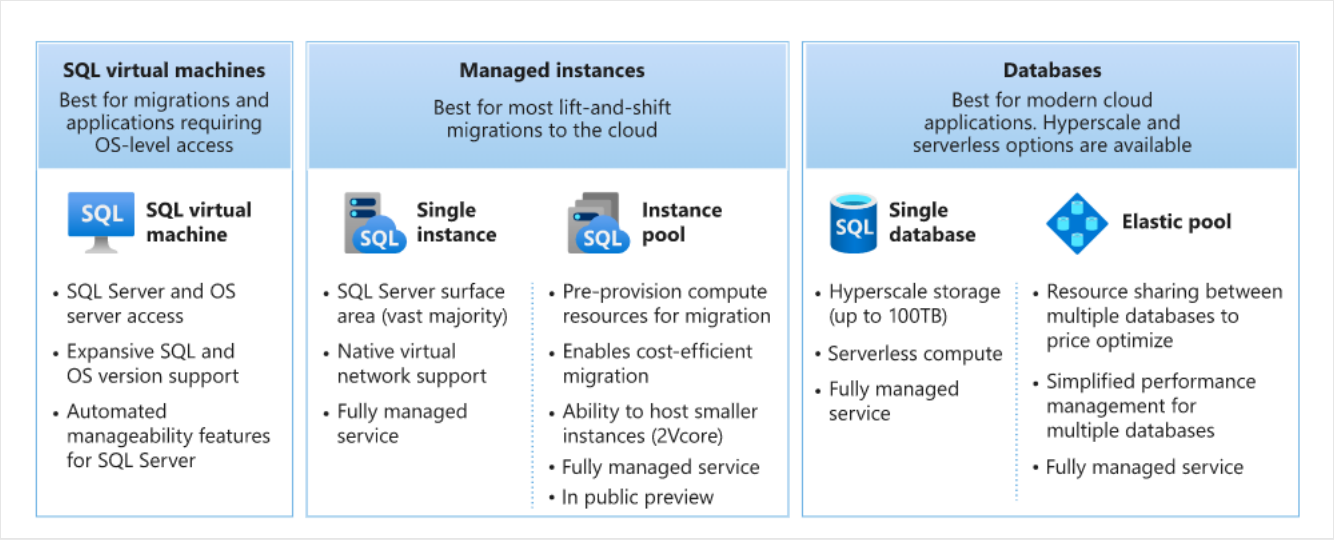
The CTO has asked you to design databases for Azure to meet all of the needs of existing structured data on premises and suggest solutions for any new relational data workloads that Tailwinds might need. Structured data includes relational data and has a shared schema. Structured data is often stored in database tables with rows, columns, and keys. Structured data is often used for application storage like an ecommerce website.

Within the umbrella of the Azure SQL platform, there are many deployment options and choices that you need to make to meet your needs. These options give you the flexibility to get and pay for exactly what you need. Here, we'll cover some of the considerations you need to make when you choose various Azure SQL deployment options. We'll also cover some of the technical specifications for each of these options. The deployment options discussed here include SQL Server on virtual machines, Azure SQL Managed Instance, Azure SQL Database, Azure SQL Managed Instance pools, and Azure SQL Database elastic database pools.

Analyze Azure SQL deployment options

As displayed in the following graphic, Azure offers SQL Server in the following ways:

- SQL Server on Azure VMs
- Managed instances:
  - Single instances
  - Instance pool
- Databases:
  - Single database
  - Elastic pool



Azure SQL Database

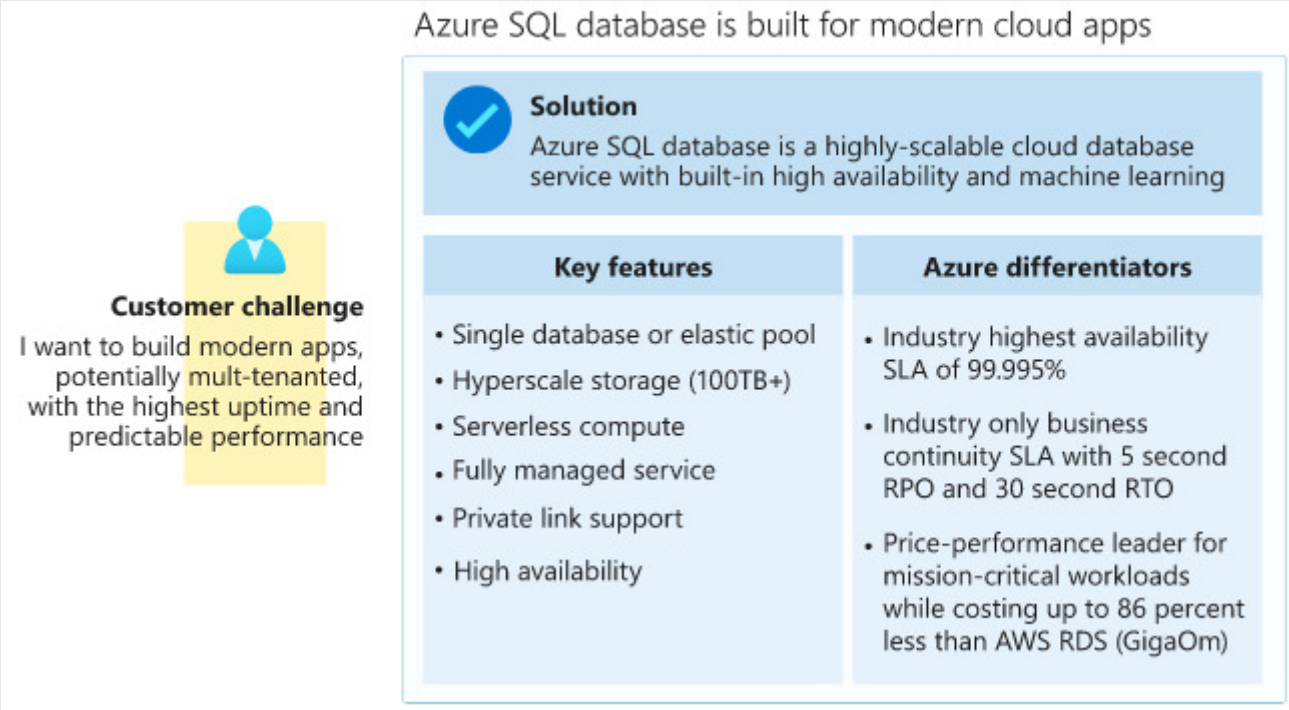
Azure SQL Database is a PaaS deployment option of Azure SQL that abstracts both the OS and the SQL Server instance. It is a highly scalable, intelligent, relational database service built for the cloud with the industry's highest availability SLA.

SQL Database is also the only deployment option that supports scenarios that require:

- Very large databases (currently up to 100TB)
- Autoscaling for unpredictable workloads (serverless)

In the following image, AccuWeather provides an example of using SQL Database.

AccuWeather has been analyzing and predicting the weather for more than 55 years. The company wanted to access Azure for its big data, machine learning, and AI capabilities. AccuWeather wants to focus on building new models and applications, not on managing databases. The company chose SQL Database to use with other services, like Azure Data Factory and Azure Machine Learning to quickly and easily deploy new internal applications to make sales and customer predictions.



What are SQL elastic pools?

When you create your Azure SQL Database, you can create a SQL elastic pool. They enable you to buy a set of compute and storage resources that are shared among all the databases in the pool. Each database can use the resources they need, within the limits you set, depending on current load.

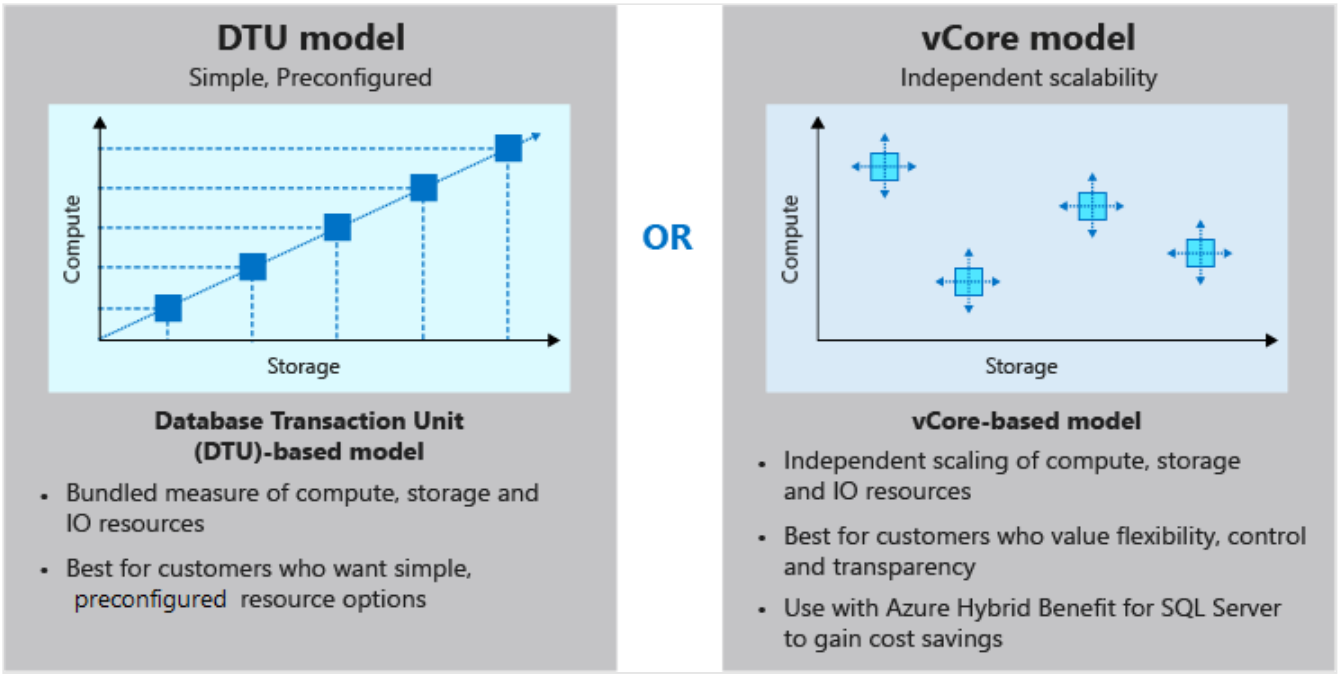
Read more about SQL elastic pools.

Analyze Azure purchasing models

The next decision to be made after deciding on deployment option is the purchasing model.

Azure SQL Database has two purchasing models as shown in the following graphic:

- DTU
- vCore



- vCores stands for Virtual cores and lets you choose the number of vCores which gives you greater control over the compute and storage resources that you create and pay for.
- DTU (Database Transaction Unit) is a combined measure of compute, storage, and IO resources. DTU model is a simple, preconfigured purchase option. This is unavailable on SQL managed Instance.
- Serverless model is a compute tier for single databases in Azure SQL Database. It automatically scales compute, based on workload demand and bills only for the amount of compute used.

The vCore-based model is recommended because it allows you to independently select compute and storage resources. The DTU-based model is a bundled measure of compute, storage, and I/O resources.

The vCore model also allows you to use Azure Hybrid Benefit for SQL Server and/or reserved capacity (pay in advance) to save money. Neither of these options is available in the DTU model.

In the table below, each purchasing model is recommended based on the requirements.

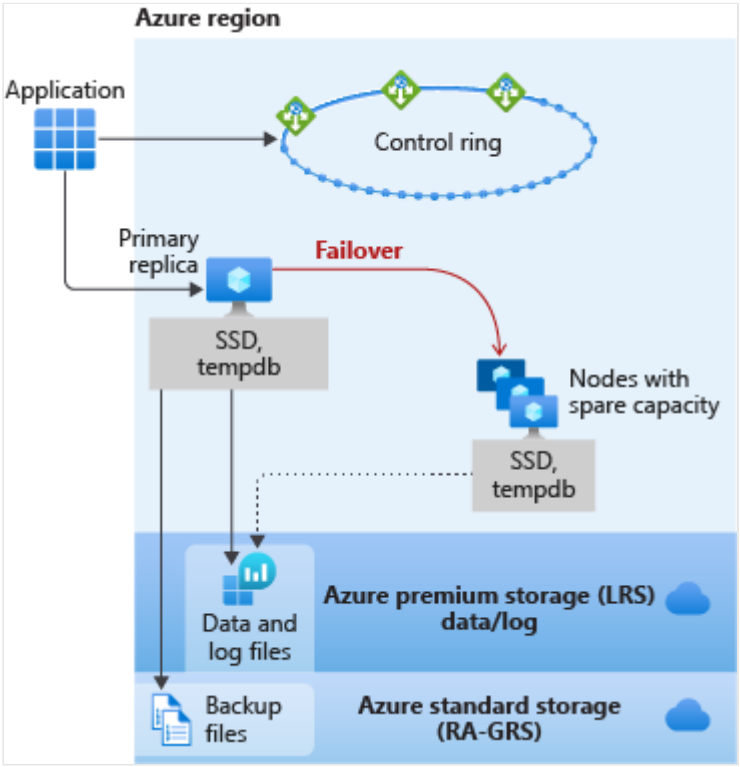
Recommendation	Requirement
DTU Model	When you need a bundled measure of compute, storage and I/O resources
vCore Model	When you need the flexibility of selecting compute and storage resources independently

Analyze Azure database service tiers

Based on performance, availability and storage needs, Azure offers three database service tiers, within the vCore module discussed in the following table.

Azure SQL Database and Azure SQL Managed Instance ensure 99.99% availability, even in the cases of infrastructure failures.

Service tiers available to Azure SQL Database and SQL Managed Instance are - General Purpose and Business Critical. Azure SQL Database also has Hyperscale service tier that is unavailable for Azure SQL Managed Instance.



In the preceding image, with **General Purpose service tier**, the primary replica uses locally attached SSD for the tempdb. The data and log files are stored in Azure Premium Storage. The backup files are then stored in Azure Standard Storage. When failover occurs, the Azure service fabric will identify a node with spare capacity and spin up a new SQL Server instance. The database files will then be attached, recovery will be run, and gateways will be updated to point applications to the new node.

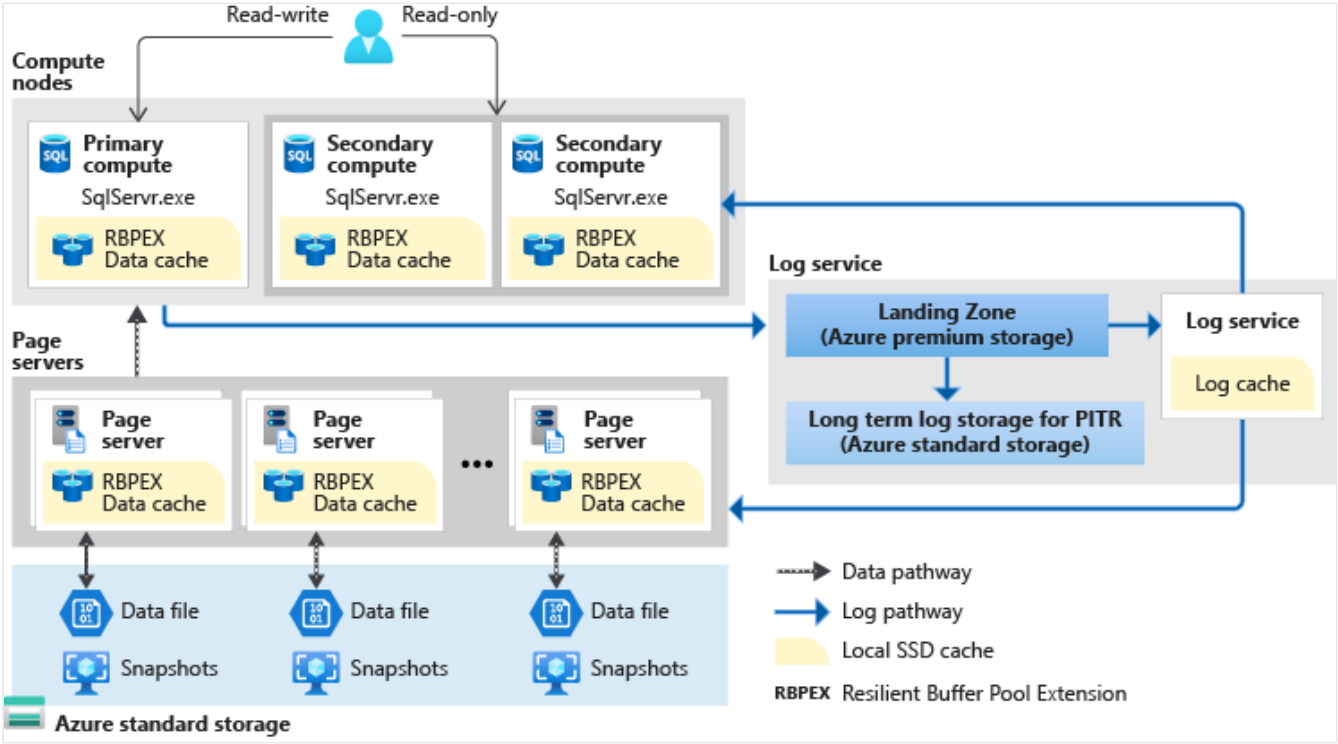
Business critical architecture is meant for mission-critical applications that need low latency and minimal downtime. Business Critical is like deploying an Always On availability group (AG) behind the scenes, the way the data and log files are stored differs from General purpose, in that they are stored on the direct attached SSD.

In the Business Critical scenario, the data and log files are all running on direct-attached SSD, which significantly reduces network latency. In this architecture group, there are three

secondary replicas. If any type of failure occurs, failing over to a secondary replica is fast because the replica already exists and has the data attached to it.

Azure SQL Database Hyperscale is a fully managed service that adapts to changing requirements by rapidly scaling storage up to 100 TB. Flexible, cloud-native architecture allows storage to grow as needed and enables you to back up data almost instantaneously and restore your database in minutes—regardless of the size of the data operation.

Hyperscale provides fast database restores, rapid scale outs, and scale ups. A Hyperscale database grows as needed - and you are billed only for the capacity you use.



The image above illustrates the different types of nodes in Hyperscale architecture. The compute node is where the relational engine lives and where language, query, and transaction processing occur. Compute nodes have SSD-based caches (labeled RBPEX - Resilient Buffer Pool Extension). There are one or more secondary compute nodes that act as hot standby nodes for failover purposes, as well as act as read-only compute nodes. Page servers are systems representing a scaled-out storage engine. The job of a page server is to serve database pages out to the compute nodes on demand. Page servers also maintain covering SSD-based caches to enhance performance. The log service accepts log records from the primary compute replica, persists them in a durable cache, and forwards the log records to the rest of compute replicas (so they can update their caches) as well as the relevant page server(s), so that the data can be updated there. In this way, all data changes from the primary compute replica are propagated through the log service to all the secondary compute replicas and page servers.

The following table compares the different scenarios when the three service tiers can be selected based on your specific needs.

Recommendation	Requirement
General Purpose	When you need balanced compute and storage options for business workloads
Business Critical	When you need low latency requirements and highest resilience to failures for business applications
Hyperscale	When you need highly scalable storage and have read-scale requirements for business workloads

Next unit: Design for Azure SQL Managed Instance

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design for Azure SQL Managed Instance

3 minutes

Azure SQL Managed Instance

Azure SQL Managed Instance is a PaaS deployment option of Azure SQL. It provides an instance of SQL Server, but removes much of the overhead of managing a virtual machine. Most of the features available in SQL Server are available in SQL Managed Instance.

SQL Managed Instance is ideal for customers who want to use instance-scoped features and want to move to Azure without re-architecting their applications.

SQL Managed Instance instance-scoped features include:

- SQL Server Agent
- Service Broker
- Common language runtime (CLR)
- Database Mail
- Linked servers
- Distributed transactions (preview)
- Machine Learning Services

Let us explore at another industry scenario. Komatsu is a manufacturing company that produces and sells heavy equipment for construction. The company had multiple mainframe applications for different types of data.

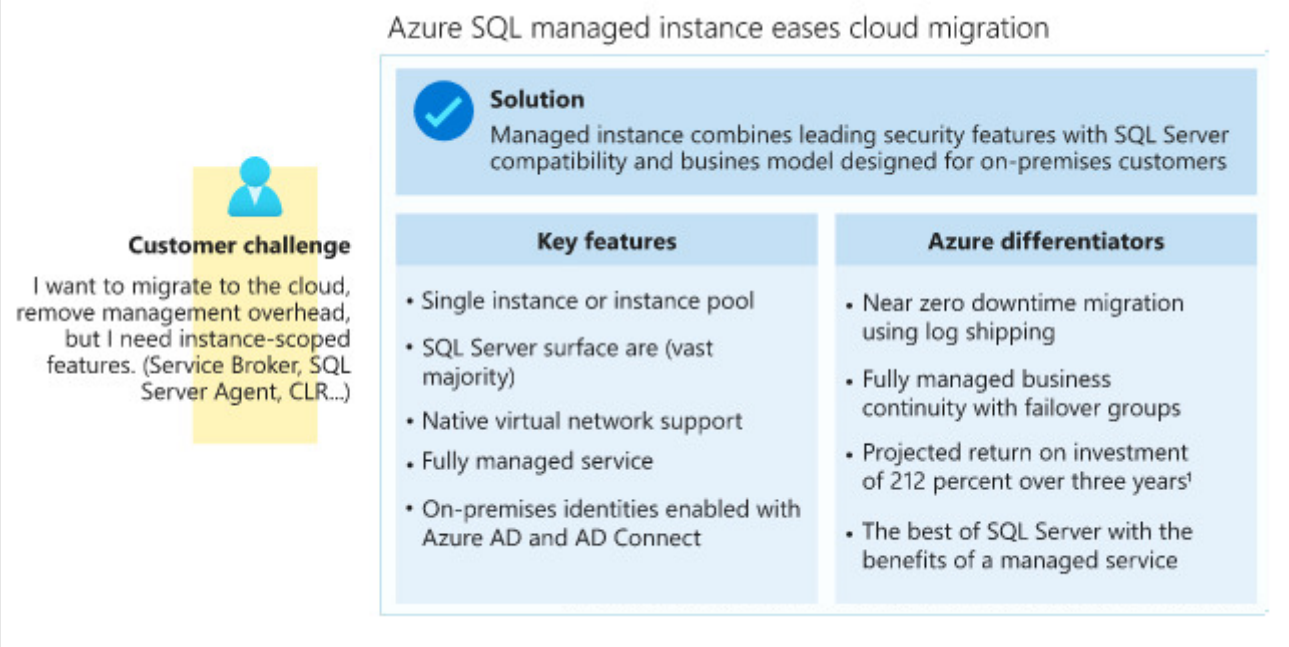
Komatsu wanted to consolidate these applications to get an overall view. Additionally, Komatsu wanted a way to reduce overhead.

Because the company uses a large surface area of SQL Server features, the IT department chose to move to Azure SQL Managed Instance. They were able to move about 1.5 terabytes of data smoothly and get the following benefits:

- Automatic patching and version updates

- Automated backups
- High availability
- Reduced management overhead

The following image shows Komatsu's challenge and their Azure solution consideration.



Scalability for Azure SQL Managed Instance

- SQL Managed Instance uses vCores mode and enables you to define maximum CPU cores and maximum of storage allocated to your instance. All databases within the managed instance will share the resources allocated to the instance.

Review this comparison of SQL Database and SQL Managed Instance

Next unit: Design for SQL Server on Azure VM

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design for SQL Server on Azure VM

3 minutes

SQL Server on Azure VMs

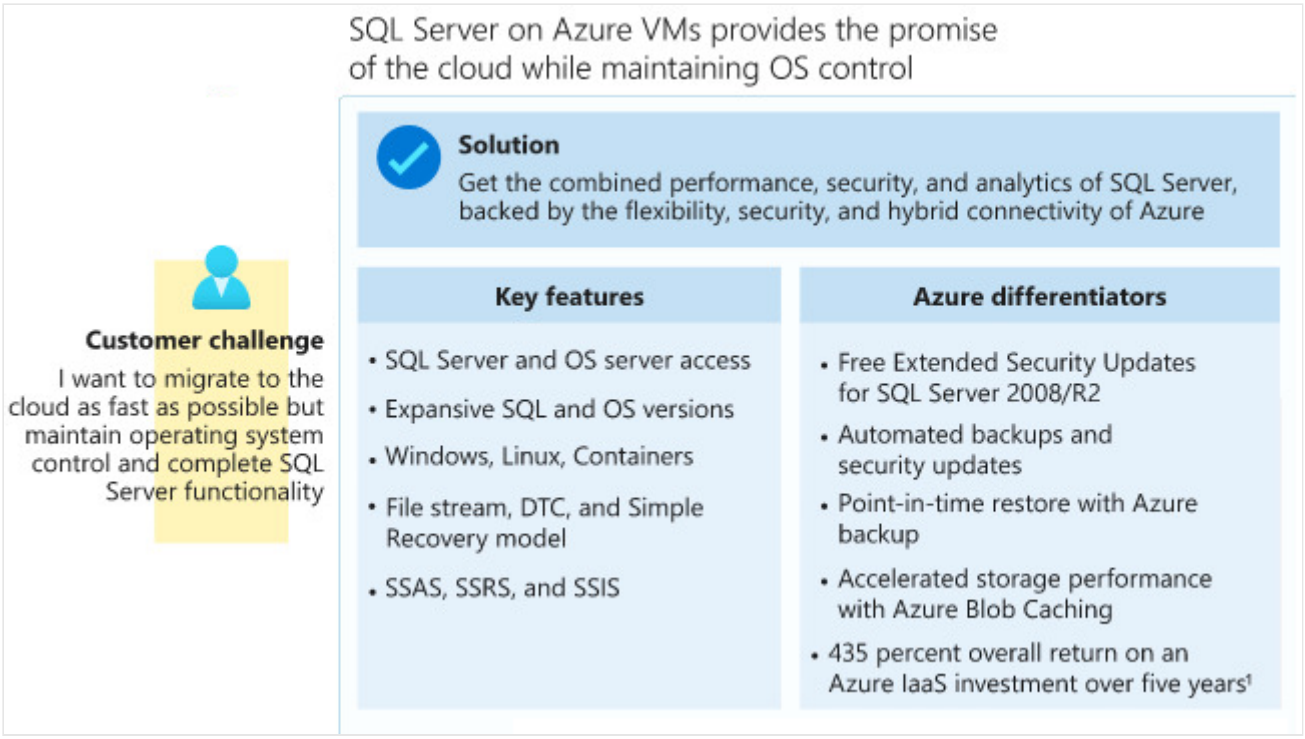
SQL Server on Azure Virtual Machines is a version of SQL Server that runs in an Azure VM. This means:

- All your SQL Server skills should directly transfer, though Azure can help automate backups and security patches.
- You have access to the full capabilities of SQL Server
- You're responsible for updating and patching the OS and SQL Server

Consider the following image. Allscripts is a leading manufacturer of healthcare software, serving physician practices, hospitals, health plans, and the pharmaceutical industry.

To transform its applications frequently and host them securely and reliably, Allscripts wanted to move to Azure quickly.

In just three weeks, the company used Azure Site Recovery to migrate dozens of acquired applications running on approximately 1,000 VMs to Azure.



Tip

If you have existing on-premises Windows Server and SQL Server licenses, you can leverage **Azure Hybrid Benefit** .

Now that we have reviewed the different deployment options, let's compare the different requirements and the solutions recommended.

Recommendation	Requirement
SQL Virtual machines	When considering migrations and applications requiring OS level access
Managed Instances	When considering Lift and Shift migrations to the cloud
Databases	When considering modern cloud applications solution

Next unit: Recommend a solution for database scalability

Continue >

100 XP

Recommend a solution for database scalability

3 minutes

Tailwind Traders, a fictitious home improvement retailer, looking to migrate to cloud had to choose a low latency and high availability database solution for storing relational data. Based on the organization's needs, they decided to implement Azure SQL Database, which is a fully managed service with an availability SLA of 99.995% in a Business critical service tier using Availability Zones.

The next decision to make for Tailwind Traders is how to handle scalability. Specifically, the scalability of a relational database. You have to help them design a solution that is dynamically scalable to handle the incoming workload. Your solution should be able to handle an expanding number of requests over time without a negative effect on availability or performance. How will you help them in this situation?

Azure SQL Database enables you to easily change resources allocated to your databases with minimal downtime, including:

- CPU power
- Memory
- IO throughput
- Storage

You can easily scale an Azure SQL Database using the Azure portal without the worry of new hardware purchase or changing the existing infrastructure.

Dynamic Scalability for Azure SQL Database enables you to:

- Use either DTU or vCore models to define maximum amount of resources that will be assigned to each database with a single database implementation
- Use Elastic pools that allow you to purchase resources for the group and set minimum and maximum resource limits for the databases within the pool.

Types of scaling in Azure SQL Database

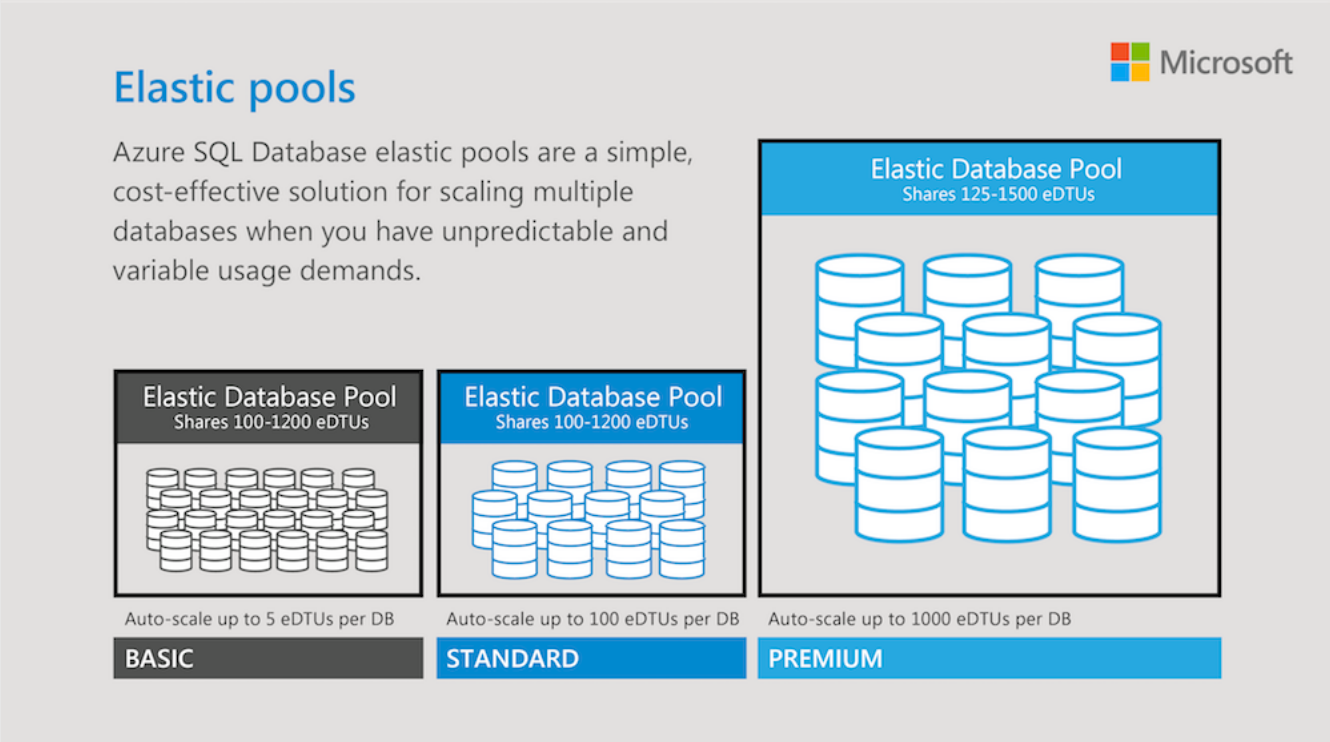
There are two types of scaling you can implement in Azure SQL Database. These are:

- **Vertical** - Vertical scaling refers to increasing or decreasing the compute size of an individual database, also known as "scaling up."
- **Horizontal** - Horizontal scaling refers to adding or removing databases in order to adjust capacity or overall performance, also called "scaling out". Horizontal scaling is managed using the Elastic Database client library.

Design Vertical scaling solution

Imagine a scenario where a small business experiences rapid growth globally and needs to maintain and scale separate Azure SQL Databases for each location. However, the rates of growth and database load vary significantly, so resource requirements are unpredictable. How would you manage scaling to meet the demands of this organization?

In this case, it is ideal to choose SQL Elastic pools, to scale, manage performance, and manage costs for a set of Azure SQL Databases.



The above image shows SQL elastic pools and the scaling capability for the different service tiers. The databases within the pool share the allocated resources. In situations where there is a low average utilization, but infrequent, high utilization spikes you can allocate enough capacity in the pool to manage the spikes for the group. To properly configure SQL elastic pools to reduce server costs, the right purchasing model and the service tier (DTU-based model in basic, standard & premium or VCore-based model in general purpose or business critical) must be selected.

Read more about SQL elastic pools.

Design Horizontal Scaling Solution

There are two types of horizontal scaling. These are:

- Read Scale-Out
- Sharding

Read Scale-Out

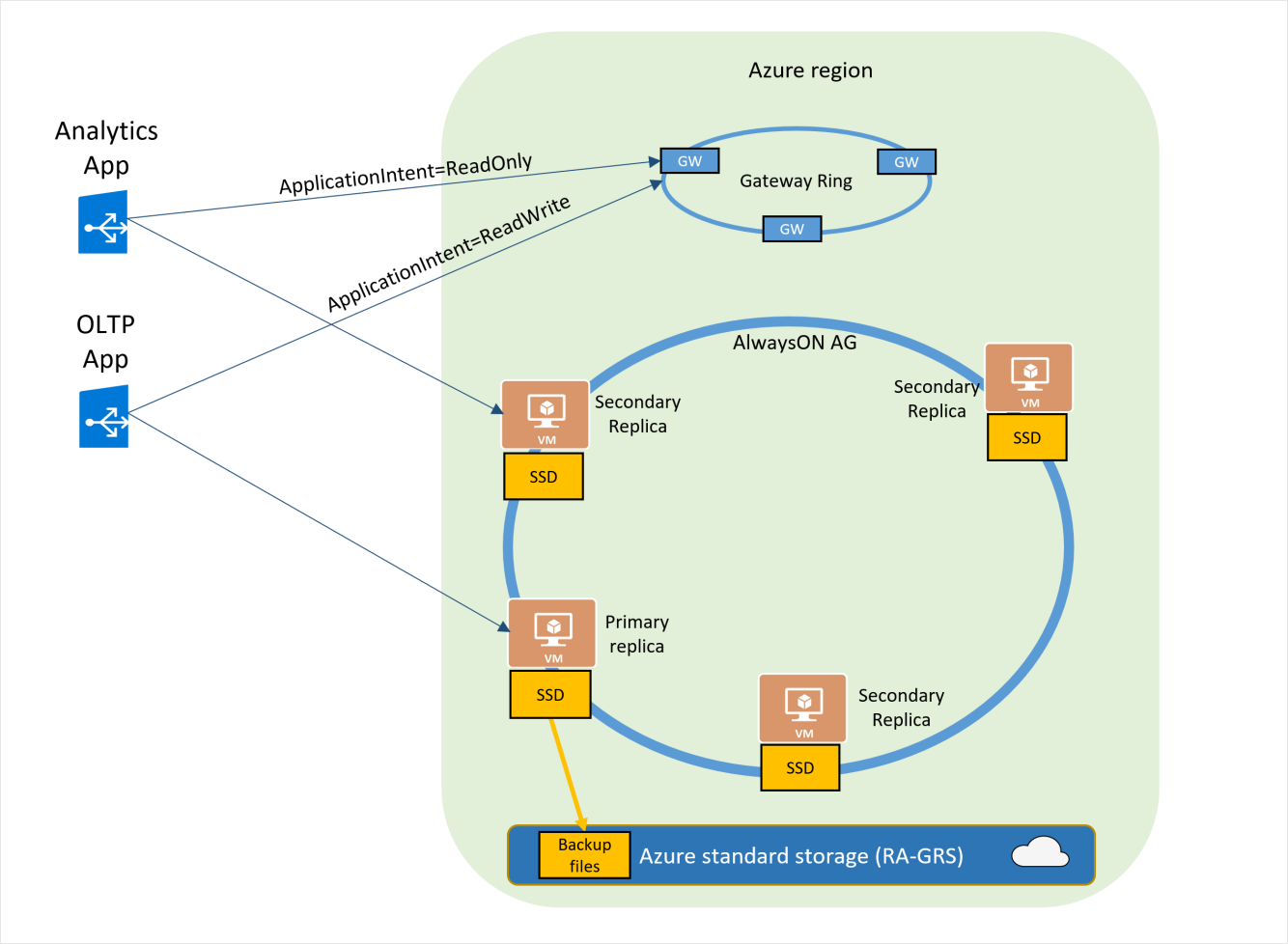
Imagine a scenario where an application's database is accessed for OLTP for updating the database as well as an Analytics app for read-only workload to render visualizations. What can you do to offload some of the compute capacity so that performance of the application isn't affected?

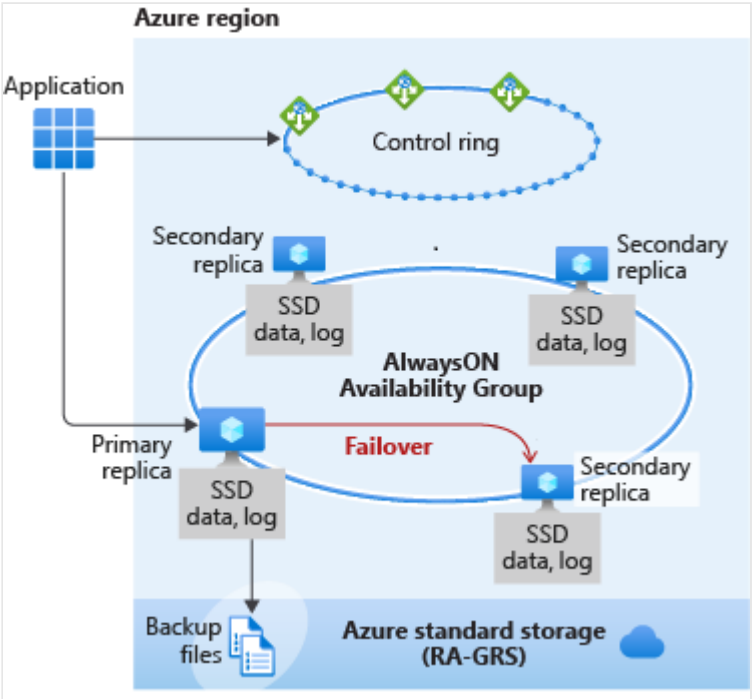
An easy choice here is to use the pre-provisioned read scale-out feature for certain service tiers. The read scale-out feature allows you to offload read-only workloads using the compute capacity of one of the read-only replicas, instead of running them on the read-write replica. This way, some read-only workloads can be isolated from the read-write workloads and will not affect their performance.

The following table shows Read Scale-out provisioning for Azure SQL Database and Azure SQL Managed Instance:

Azure SQL Managed Instance	Azure SQL Database
For the basic, standard and general purpose tier, read scale-out feature is unavailable	For the basic, standard and general purpose tier, read scale-out feature is unavailable
For the Business Critical tier, read scale-out is auto-provisioned	For the Premium and Business Critical tier, read scale-out is auto-provisioned
Hyperscale tier is unavailable in Azure Managed Instance	Read scale-put feature is available in Hyperscale tier if atleast one secondary replica is created

The following image shows read scale-out in a business critical service tier:





In the image above, in Business Critical scenario, the data and log files are all running on direct-attached SSD, which significantly reduces network latency. In this architecture group, there are three secondary replicas. If any type of failure occurs, failing over to a secondary replica is fast because the replica already exists and has the data attached to it.

The premium/business critical tier of Azure SQL Database or Azure Managed Instance has an Always On Availability Group. This group is for disaster recovery and high-availability of the application. There is a primary read-write replica and several secondary read-only replicas. The secondary replicas are provisioned with the same compute size as the primary replica. You set the connection string option to decide whether the connection is routed to the write replica or to a read-only replica.

You can disable and re-enable read scale-out on single databases and elastic pool databases in the Premium or Business Critical service tiers using the following methods:

- Azure portal
- Azure PowerShell
- REST API

> **Note**
>
> Data changes made on the primary replica propagate to read-only replicas asynchronously. Within a session connected to a read-only replica, reads are always transactionally consistent. However, because data propagation latency is variable, different replicas can return data at slightly different points in time relative to the primary and each other.

Sharding

Imagine a scenario where you need to solve a database problem for an application accessing the database that has a huge amount of transaction throughput that exceeds the database capability. How will you provision the database for performance and availability?

A possible solution for the above scenario is Horizontal Scaling or horizontal partitioning by Sharding. This is a technique to distribute large amounts of identically structured data across a number of independent databases.

Reasons for Sharding include:

- If the total amount of data is too large to fit constraints of a single database
- If the transaction throughput of the overall workload exceeds capacities of an individual database
- When different customers or tenents' data needs physical isolation from each other

- Within an organization, there is a geographical separation of data for compliance reasons

Adopt suitable database scaling strategy

The following table identifies key points to remember before choosing Vertical/Horizontal scaling.

Requirement	Description
Do you have to manage and scale multiple Azure SQL Databases that have varying and unpredictable resource requirements?	**SQL elastic pools.** Vertical scale up is a good solution for this scenario. Elastic pools solve this problem by ensuring that databases get the performance resources they need when they need it. They provide a simple resource allocation mechanism within a predictable budget. There is no per-database charge for elastic pools. You are billed for each hour a pool exists at the highest eDTU or vCores, regardless of usage or whether the pool was active for less than an hour.
Do you have different sections of the database residing in different parts of the world for compliance concerns?	**Horizontal scaling by Sharding** works best. Sharding enables you to split your data into several databases and scale them independently. The shard map manager is a special database that maintains global mapping information about all shards (databases) in a shard set. The metadata allows an application to connect to the correct database, based upon the value of the sharding key.
Are there dependencies such as commercial BI or data integration tools where multiple databases contribute rows into a single overall result for use in Excel, Power BI, or Tableau?	Use **Elastic database tools** and elastic query feature within it to access data spread across multiple databases. Elastic query is available on standard tier, querying can be done in T-SQL that spans multiple databases in Azure SQL Database. Cross-database queries can be executed to access remote tables, and to connect Microsoft and third-party tools (Excel, Power BI, Tableau, etc.) to query across data tiers. Using this feature, you can scale out queries to large data tiers and visualize the results in business intelligence (BI) reports.

Next unit: Recommend a solution for database availability

Continue >

100 XP

Recommend a solution for database availability

3 minutes

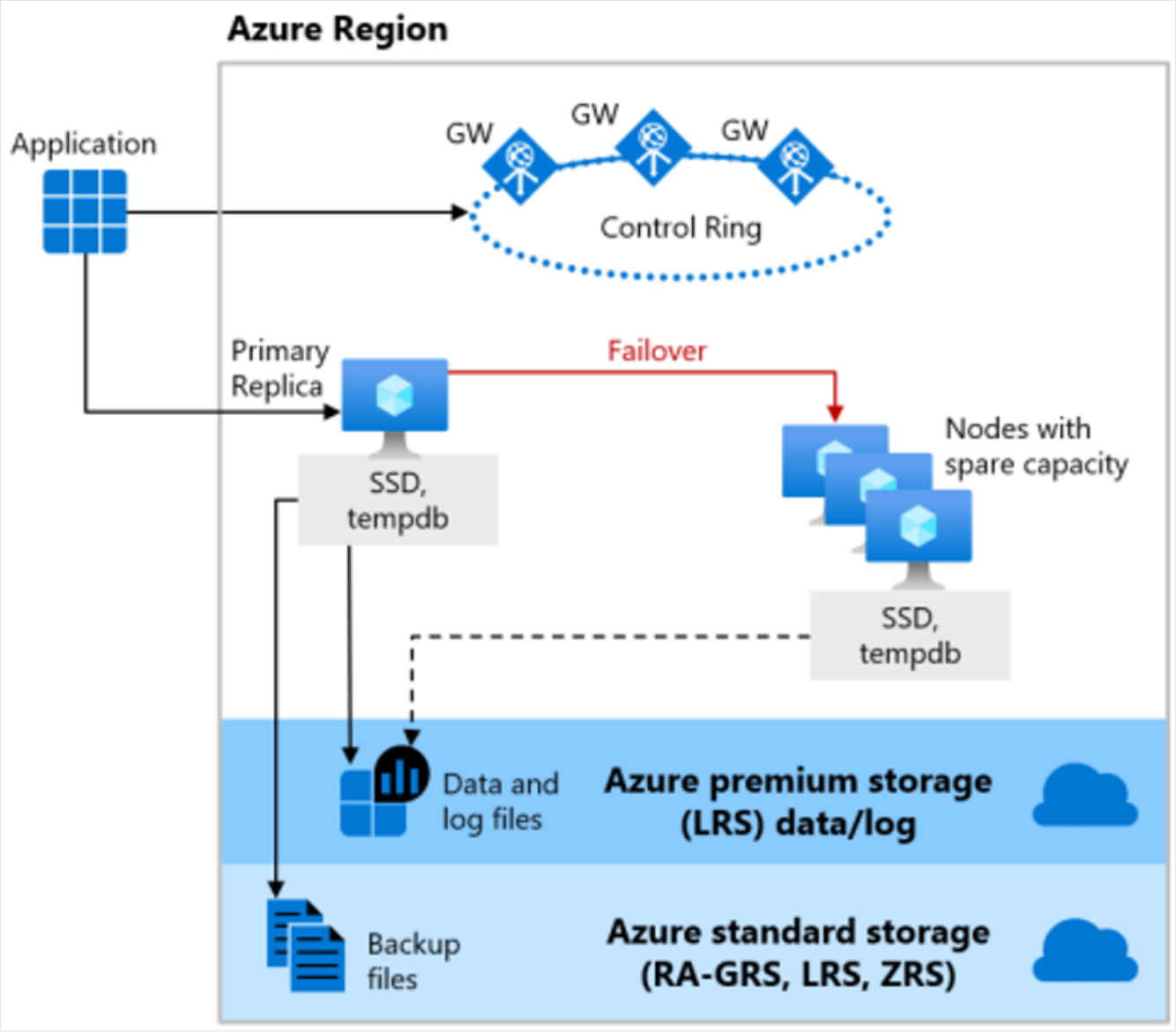
To understand the availability options and capabilities in Azure SQL, you need to understand service tiers. The service tier you select will determine the underlying architecture of the database or managed instance that you deploy.

There are two purchasing models to consider: DTU and vCore. This unit will focus on the vCore service tiers and their architectures for high availability. You can equate the DTU model's Basic and Standard tiers to General Purpose, and its Premium tiers to Business Critical.

General Purpose

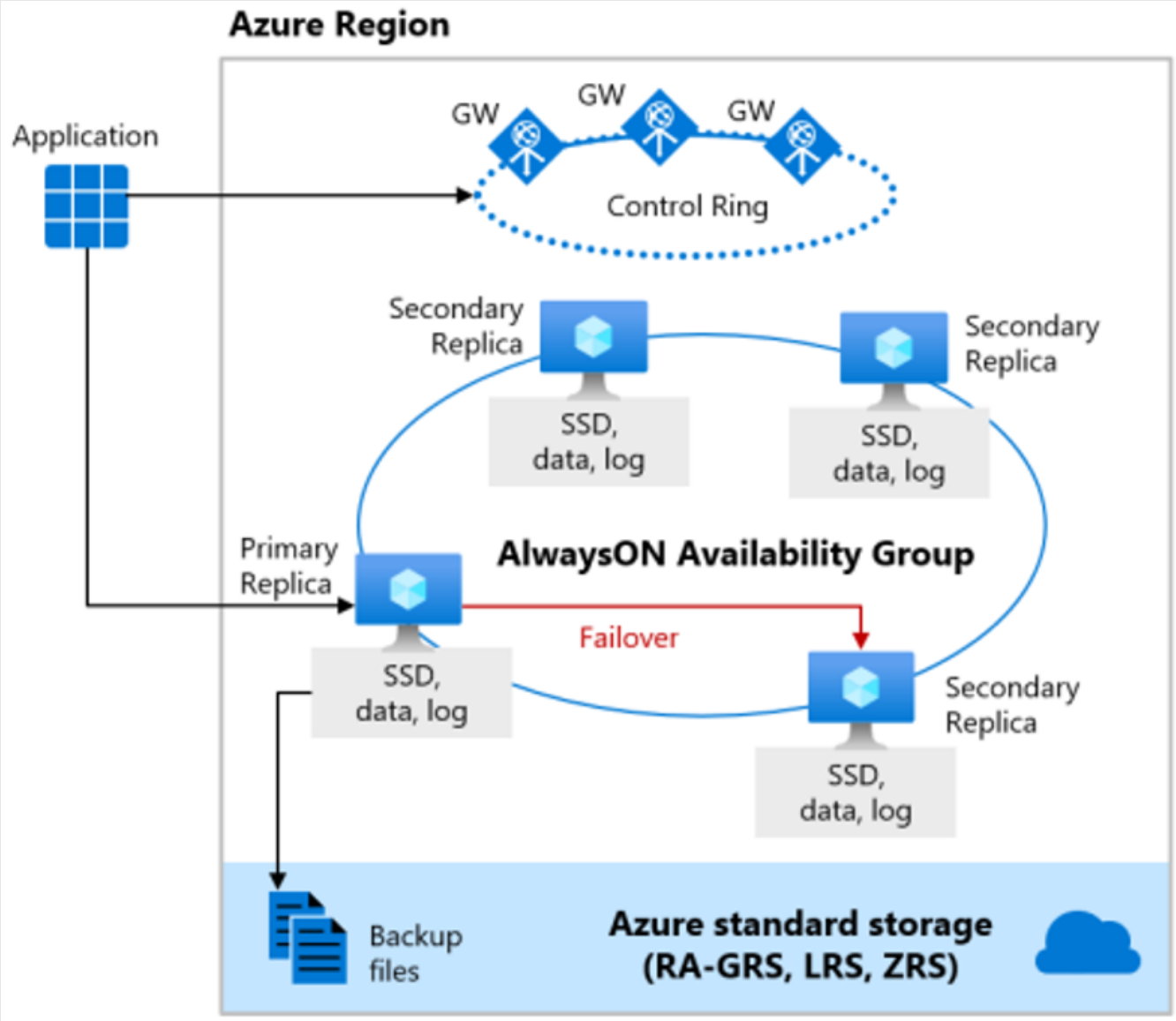
Databases and managed instances in the General Purpose service tier have the same availability architecture. Using the following figure as a guide, first consider the application and control ring. The application connects to the server name, which then connects to a gateway (GW) that points the application to the server to connect to, running on a VM. With General Purpose, the primary replica uses locally attached SSD for the tempdb. The data and log files are stored in Azure Premium Storage, which is locally redundant storage (multiple copies in one region). The backup files are then stored in Azure Standard Storage, which is RA-GRS by default. In other words, it's globally redundant storage (with copies in multiple regions).

All of Azure SQL is built on Azure Service Fabric, which serves as the Azure backbone. If Azure Service Fabric determines that a failover needs to occur, the failover will be similar to that of a failover cluster instance (FCI). The service fabric will locate a node with spare capacity and spin up a new SQL Server instance. The database files will then be attached, recovery will be run, and gateways will be updated to point applications to the new node. No virtual network or listener or updates are required. This capability is built in.



Business Critical

The next service tier to consider is Business Critical, which can generally achieve the highest performance and availability of all Azure SQL service tiers (General Purpose, Hyperscale, Business Critical). Business Critical is meant for mission-critical applications that need low latency and minimal downtime.



Using Business Critical is like deploying an Always On availability group (AG) behind the scenes. Unlike in the General Purpose tier, in Business Critical, the data and log files are all running on direct-attached SSD, which significantly reduces network latency. (General Purpose uses remote storage.) In this AG, there are three secondary replicas. One of them can be used as a read-only endpoint (at no additional charge). A transaction can complete a commit when at least one of the secondary replicas has hardened the change for its transaction log.

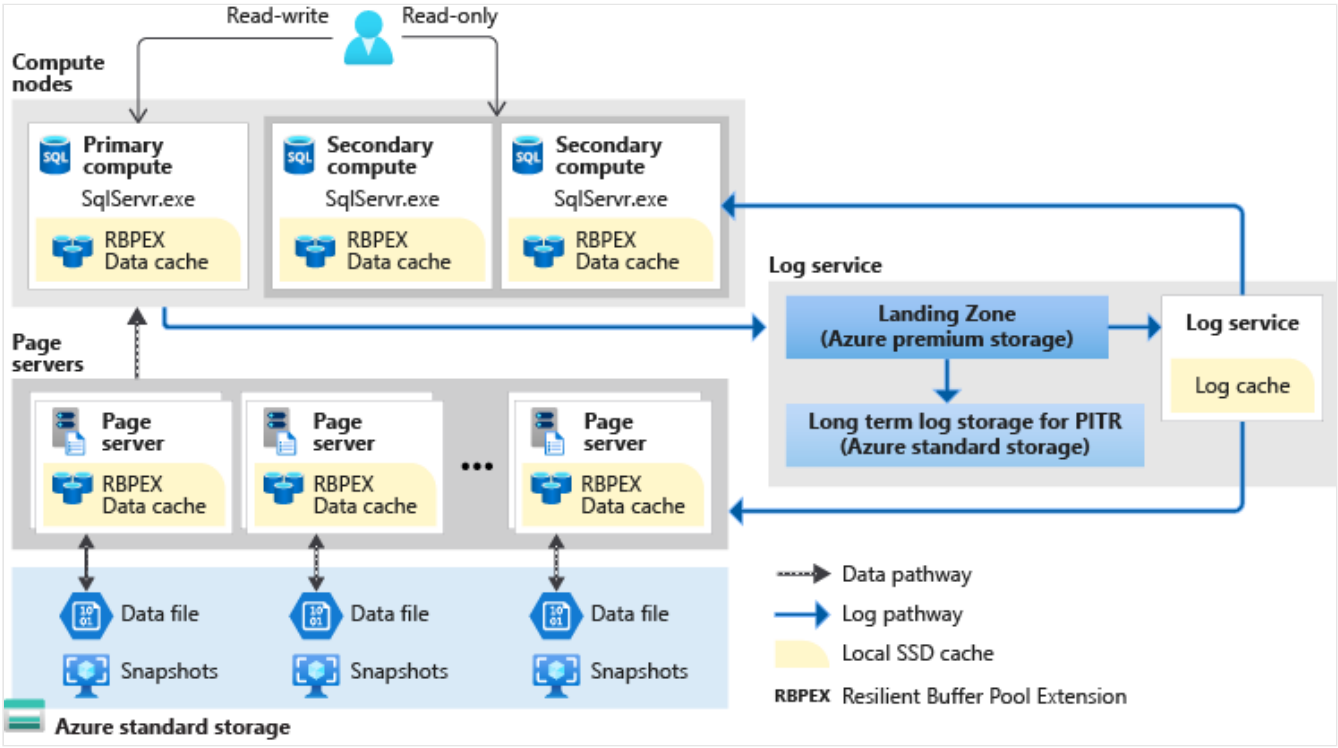
Read scale-out with one of the secondary replicas supports session-level consistency. So if the read-only session reconnects after a connection error caused by replica unavailability, it might be redirected to a replica that's not 100% up to date with the read-write replica. Likewise, if an application writes data by using a read-write session and immediately reads it by using a read-only session, the latest updates might not immediately be visible on the replica. The latency is caused by an asynchronous transaction log redo operation.

If any type of failure occurs and the service fabric determines a failover needs to occur, failing over to a secondary replica is fast because the replica already exists and has the data attached to it. In a failover, you don't need a listener. The gateway will redirect your connection to the

primary even after a failover. This switch happens quickly, and then the service fabric takes care of spinning up another secondary replica.

Hyperscale

The Hyperscale service tier is available only in Azure SQL Database. This service tier has a unique architecture because it uses a tiered layer of caches and page servers to expand the ability to quickly access database pages without having to access the data file directly.



Because this architecture uses paired page servers, you can scale horizontally to put all the data in caching layers. This new architecture also allows Hyperscale to support databases as large as 100 TB. Because it uses snapshots, nearly instantaneous database backups can occur regardless of size. Database restores take minutes rather than hours or days. You can also scale up or down in constant time to accommodate your workloads.

It's interesting to note how the log service was pulled out in this architecture. The log service is used to feed the replicas and the page servers. Transactions can commit when the log service hardens to the landing zone. So the consumption of the changes by a secondary compute replica isn't required for a commit. Unlike in other service tiers, you can determine whether you want secondary replicas. You can configure zero to four secondary replicas, which can all be used for read-scale.

As in the other service tiers, an automatic failover will happen if service fabric determines it needs to. But the recovery time will depend on the existence of secondary replicas. For example, if you don't have replicas and a failover occurs, the scenario will be similar to that of the General Purpose service tier: the service fabric first needs to find spare capacity. If you

have one or more replicas, recovery is faster and more closely aligns to that of the Business Critical service tier.

Business Critical maintains the highest performance and availability for workloads with small log writes that need low latency. But the Hyperscale service tier allows you to get a higher log throughput in terms of MB/second, provides for the largest database sizes, and provides up to four secondary replicas for higher levels of read scale. So you'll need to consider your workload when you choose between the two.

Geo-replication and auto-failover groups

After you choose a service tier (and consider Availability Zones as applicable), you can consider some other options for getting read-scale or the ability to fail over to another region: geo-replication and auto-failover groups. In SQL Server on-premises, configuring either of these options is something that would take a lot of planning, coordination, and time.

The cloud, and Azure SQL specifically, have made this process easier. For both geo-replication and auto-failover groups, you can get configured with a few clicks in the Azure portal or a few commands in the PowerShell/Azure CLI.

Next unit: Design security for data at rest, data in transit, and data in use

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design security for data at rest, data in transit, and data in use

3 minutes

You've been hired as a Senior Database Administrator to help ensure the security of the Azure SQL Database environment. Your company is an online sporting goods retailer that stores customer data like phone numbers, addresses, and credit cards in Azure SQL Database hosted on the cloud. What solution would you implement to prevent unauthorized data access by someone that is working to support this cloud hosted database?

Classifying stored data by sensitivity and business impact helps organizations determine the risks associated with the data. In this unit, we will learn the different data states and encryption modes in detail.

The three basic tenets of good information protection are:

- Data discovery
- Classification
- Protection

Large organizations, governments, and military entities have been using data classification to manage their data's integrity. The result of data classification is metadata that enables us to label the data as:

- Public
- Confidential
- Restricted

After the data is classified, you can implement data protection measures for highly-classified data. In this section, we explore data encryption for structured data.

Data exists in one of three basic states - data-at-rest, data-in-motion, data-in-process. This table shows different data states with possible encryption methods.

DATA STATE	ENCRYPTION METHOD
Data-at-rest	TDE, Always Encrypted
Data-in-motion	SSL/TLS, Always Encrypted
Data-in-process	Dynamic data masking

> **Note**
>
> Defense in depth is a strategy that employs a layered approach to slow the advance of an attack aimed at acquiring unauthorized access to information.

Protect data-at-rest

Data encryption at rest is a mandatory step toward data privacy, compliance, and data sovereignty. Encryption helps mitigate risks related to unauthorized data access. Data-at-rest needs to be protected from unauthorized or offline access to raw files or backups to an unsecured server or making a copy of all the database and transaction log files to another unsecured server.

Transparent data encryption (TDE)

TDE protects Azure SQL Database, Azure SQL Managed Instance, and Azure Synapse Analytics against the threat of malicious offline activity by encrypting data at rest. It performs real-time encryption and decryption of the database, associated backups, and transaction log files at rest without requiring changes to the application. TDE is enabled by default to all newly deployed Azure SQL Databases. With Azure SQL Managed Instance, databases created after February 2019 have TDE enabled.

- TDE performs encryption and decryption of the data at the page level.
- The data is encrypted as the data is written to the data page on disk and decrypted when the data page is read into memory.
- The end result is that all data pages on disk are encrypted.
- Database backups will also be encrypted because a backup operation just copies the data pages from the database file to the backup device. No decryption is done during the backup operation.

- TDE encrypts the storage of an entire database by using a symmetric key called the Database Encryption Key (DEK).
- Service-managed TDE - where the DEK is protected by a built-in server certificate.
- Customer-managed TDE - the TDE Protector that encrypts the DEK is supplied by customer and stored in a customer-owned and managed in their key management system

> **Note**
>
> To use TDE with databases in an Always On Availability Group, the certificate used to encrypt the database must be backed up and restored to the other servers within the Availability Group that will be hosting copies of the database.

Azure's Azure Key Vault service for TDE

Azure Key Vault is a tool for storing and accessing sensitive data like passwords, certificates or keys. It is a centralized storage solution for application secrets and also helps monitor how and when the keys/certificates are being accessed. Azure Key vault easily integrates with other Azure services. Key Vault data has its own RBAC policies, separate from the Azure subscription so users need explicit access to be granted.

In the scenario discussed at the beginning of the unit, you are implementing a solution to protect data-at- rest from unauthorized data access by someone that is working to support this cloud hosted database. **Customer-managed TDE -** Bring Your Own Key implementation of TDE is a good solution for the above scenario. The TDE Protector that encrypts the Database Encryption Key (DEK) is a customer-managed asymmetric key, which is stored in a customer-owned and managed Azure Key Vault (Azure's cloud-based external key management system) and never leaves the key vault.

> **Important**
>
> You can read more here **Azure SQL TDE with customer-managed key**

Protect data-in-transit

SQL Database, SQL Managed Instance, and Azure Synapse Analytics enforce encryption (SSL-Secure Sockets Layer/TLS - Transport Layer Security) at all times for all connections. This ensures all data is encrypted "in transit" between the client and server. **Transport Layer Security** (TLS) is used by all drivers that Microsoft supplies or supports for connecting to

databases in Azure SQL Database or Azure SQL Managed Instance. In some circumstances, you might want to isolate the entire communication channel between your on-premises and cloud infrastructures by using a VPN.

The next table has different scenarios and solutions for VPN Gateway, TLS and HTTPS.

SCENARIO	SOLUTION
Secure access from multiple workstations located on-premises to an Azure virtual network	Use site-to-site VPN
Secure access from an individual workstation located on-premises to an Azure virtual network	Use point-to-site VPN
Move large data sets over a dedicated high-speed wide-area network (WAN) link	Use Azure ExpressRoute
Interact with Azure Storage through the Azure portal	All transactions occur via HTTPS. You can also use Storage REST API over HTTPS to interact with Azure Storage and Azure SQL Database.

Protect data-in-use

Consider the scenario where you have customer service representatives accessing the database containing customer Phone Number and email address. You want to still enable them to verify the users calling in so except the last four digits of the customer's phone number, the rest needs to be encrypted and not revealed to the service representative. How would you implement a solution for this case?

Dynamic Data Masking

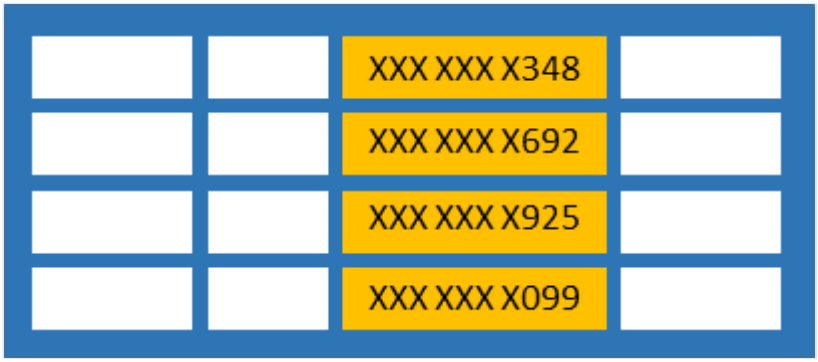
Dynamic data masking is a policy-based security feature that hides the sensitive data in the result set of a query over designated database fields, while the data in the database is not changed. It helps prevent unauthorized access to sensitive data by enabling customers to designate how much of the sensitive data to reveal with minimal impact on the application layer.

- Dynamic data masking automatically discovers potentially sensitive data in Azure SQL Database and SQL Managed Instance and provides actionable recommendations to mask

these fields.

- It works by obfuscating the sensitive data in the result set of a query
- Data masking policy can be set up in Azure portal only for Azure SQL Database
- Dynamic data masking can be set up using PowerShell cmdlets and REST API

In the image below, the last three digits of phone number column and rest of the characters are masked.



Data masking rules consist of the column to apply the mask to and how the data should be masked. Use the standard mask or specify your own custom masking logic.
The following table shows the column that needs to be masked and the logic:

SCENARIO	SOLUTION
Default	Displays the default value for that data type instead
Credit card	XXXX-XXXX-XXXX-1234
Email	aXX @XXXX.com
Random Number	Generated random number between selected boundaries
Custom Text	Adds padding string between exposed prefix and exposed suffix characters

> **Note**
>
> Note that Dynamic Data Masking is a presentation layer feature, and unmasked data can always be viewed by administrators.

Consider a scenario where a database is queried and accessed by users of all authorization levels. A user that doesn't have administrator privileges queries a table that has some sensitive customer data such as phone number, email etc. along with purchase, and product information.
How will you implement a solution that suppresses the sensitive information while allowing the user to view the other fields?

Dynamic data masking for data-in-use - You can set up a dynamic data masking policy using default masking logic for email, credit cards etc.. Or you can specify custom text or random number for the field in question. You can allow other users to view the non-masked versions by adding them to the SQL users, excluded from the masking list.

Always Encrypted feature for data-at-rest and data-in-transit

Always Encrypted is a feature designed to protect sensitive data stored in specific database columns from access (for example, credit card numbers, national identification numbers, etc.). This includes database administrators or other privileged users who are authorized to access the database to perform management tasks. The data is always encrypted, which means the encrypted data is decrypted only for processing by client applications with access to the encryption key. This key can be stored either in the Windows Certificate Store or in Azure Key Vault.

How Always Encrypted works

Step by step process for Always Encrypted is explained below:

- Always Encrypted uses two types of keys: column encryption keys and column master keys.
- A column encryption key is used to encrypt data in an encrypted column. A column master key is a key-protecting key that encrypts one or more column encryption keys.
- The Database Engine only stores encrypted values of column encryption keys and the information about the location of column master keys, which are stored in external trusted key stores, such as Azure Key Vault, Windows Certificate Store
- To access data stored in an encrypted column in plaintext, an application must use an Always Encrypted enabled client driver. Encryption and decryption occurs via the client driver.
- The driver transparently collaborates with the Database Engine to obtain the encrypted value of the column encryption key for the column as well as the location of its corresponding column master key.

- The driver contacts the key store, containing the column master key, in order to decrypt the encrypted column encryption key value, and then it uses the plaintext column encryption key to encrypt the parameter.
- The driver substitutes the plaintext values of the parameters targeting encrypted columns with their encrypted values, and it sends the query to the server for processing.
- The server computes the result set, and for any encrypted columns included in the result set, the driver attaches the encryption metadata for the column, and then the driver decrypts the results and returns plaintext values to the application.

> **Tip**
>
> Read about Always Encrypted **best practices**

> **Important**
>
> Always Encrypted does not work with Dynamic Data Masking. It is not possible to encrypt and mask the same column. You need to prioritize protecting data in use vs. masking the data for your app users via Dynamic Data Masking.

Consider this scenario - A customer has an on-premises client application at their business location. As part of their cloud migration of their database, they are evaluating Azure SQL Database. They plan to outsource the database maintenance to third parties. They worry that the sensitive data shouldn't be accessed by the third party vendors or the cloud administrators. What will you suggest?

Always Encrypted for data-at-rest and data-in-transit - Always Encrypted ensures the separation of duties between database administrators and application administrators. By storing the Always Encrypted keys in a trusted key store hosted on-premises, they can ensure Microsoft cloud administrators have no access to sensitive data. Always Encrypted enables customers to confidently store sensitive data outside of their direct control.

Next unit: Design for Azure SQL Edge

Continue >

Design for Azure SQL Edge

3 minutes

Azure SQL Edge is an optimized relational database engine geared for IoT and IoT Edge deployments. Azure SQL Edge is built on the same engine as SQL Server and Azure SQL. This means that the developers with SQL server skills can reuse their code to build edge-specific solutions on Azure SQL Edge. Azure SQL Edge provides capabilities to stream, process, and analyze relational and non-relational such as JSON, graph and time-series data.

Azure SQL Edge is available with two different editions, as described in the following table. These editions have identical feature sets, and only differ in terms of their usage rights and the amount of memory and cores they can access on the host system.

- Azure SQL Edge Developer. For development only. Each Azure SQL Edge Developer container is limited to up to 4 cores and 32 GB memory.
- Azure SQL Edge. For production. Each Azure SQL Edge container is limited to up to 8 cores and 64 GB memory.

Understand Azure SQL Edge deployment models

Azure SQL Edge supports two deployment modes.

- **Connected deployment through Azure IoT Edge.** Azure SQL Edge is available on the Azure Marketplace and can be deployed as a module for Azure IoT Edge
- **Disconnected deployment.** Disconnected deployment through Azure SQL Edge container images which can be pulled from docker hub and deployed either as a standalone docker container or on a Kubernetes cluster.

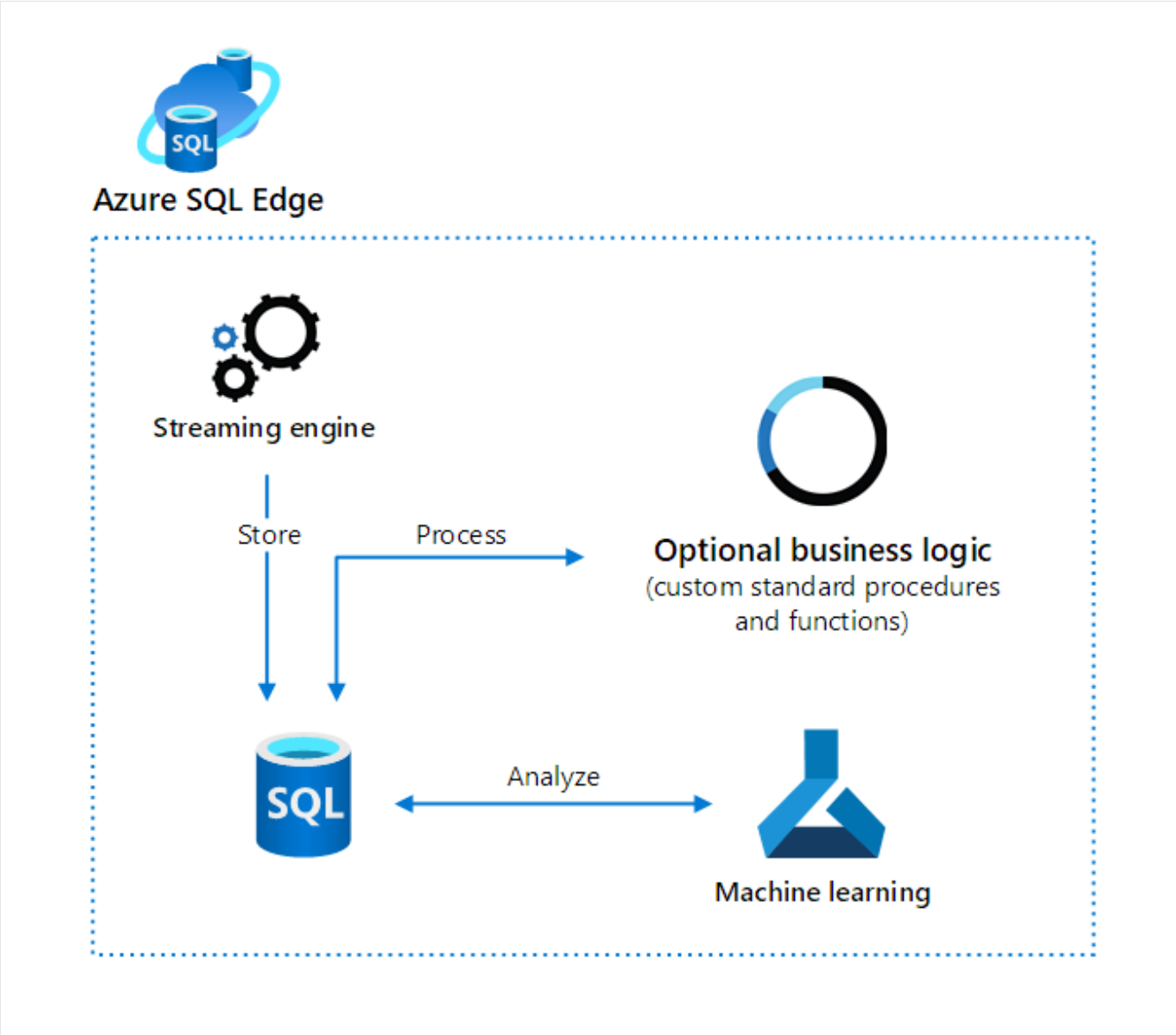
> **Important**
>
> Azure SQL Edge is optimized for IoT use cases and workloads. SQL Server and SQL Database, in contrast, are built for mission-critical, data-management solutions, and line-of-business (LOB) apps.

How does Azure SQL Edge work?

Azure SQL Edge is a containerized Linux application. The startup-memory footprint is less than 500 megabytes (MB). This allows users design and build apps that run on many IoT devices. SQL Edge can:

- Capture continuous data streams in real time
- Integrate the data in a comprehensive organizational data solution

The following diagram shows SQL Edge's capability of capturing and storing streaming data.

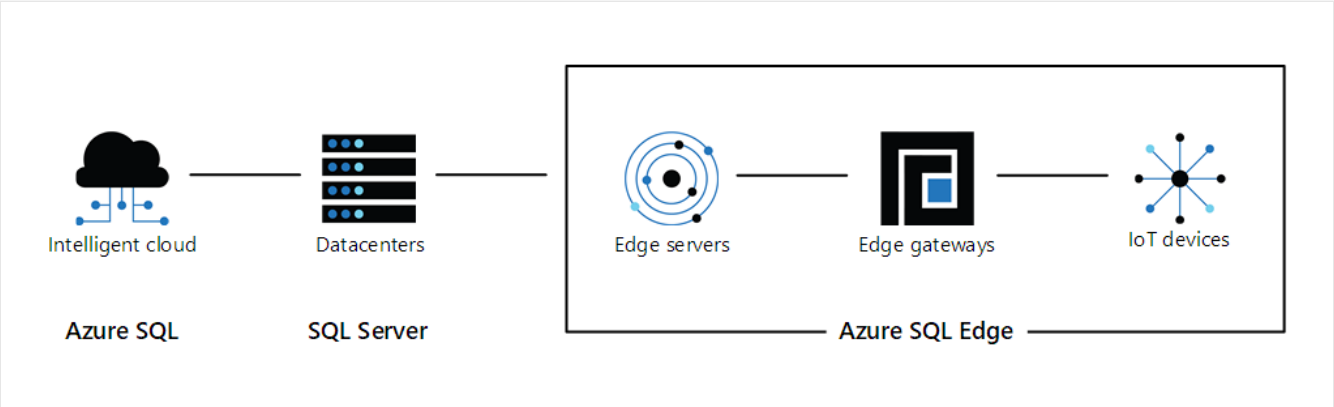


With IoT data it is important not only to capture continuous real-time streams of increasing amounts of data but also derive valuable insights about it. Azure SQL Edge supports a built-in streaming engine to help address these needs. It has the following features

- The Streaming engine allows transformation, Windowed aggregation, Simple anomaly detection and classification of incoming data streams

- A time-series storage engine that allows storage of time-indexed data. This can be aggregated and stored in cloud for future analysis

The following diagram shows Azure SQL Edge interacts with components at the network edge including edge gateways, IoT devices, and edge servers.



Security is a primary concern when deploying IoT apps to the edge. Because Azure SQL Edge is based on SQL Server technology, one of the most secure database platforms available, it has the same security features of SQL Server Enterprise. Also the same security policies and practices are extended from cloud to the edge.

Securing Azure SQL Edge deployments involves the steps described in the following table:

1. Platform and system security. This includes the physical docker host, the operating system on the host, and the networking systems connecting the physical device to applications and clients.

2. Authentication and authorization. SQL authentication refers to the authentication of a user when connecting to Azure SQL Edge using username and password. Authorization refers to the permissions assigned to a user within a database in Azure SQL Edge

3. Database object security. "Securables" are the server, database, and objects the database contains. Encryption enhances security. Data protection with Transparent Data Encryption (TDE) enables compliance with many security regulations. Always Encrypted provides separation between users who own the data and those who manage it

4. Application security. Azure SQL Edge security best practices include writing secure client applications.

Scenario - Real time ingestion of data

Let's imagine you work for an automotive manufacturing company. You're working on an IoT app that ingests data from several IoT sensors in the vehicles your company manufactures. It's important that the data is usable all the time, regardless of whether the vehicles' apps are

online or offline. Another goal is to use the data to help with product development. This means that the data must synchronize easily with cloud-based database systems built in Azure SQL.

You've been tasked to recommend a solution specifically for SQL Server and should be powerful enough to support edge compute. It should be secure enough to help meet the privacy needs of IoT applications.

Azure SQL Edge is best suited to support the above requirement due to its small footprint and is edge-optimized for IoT devices.

When do we use Azure SQL Edge?

Azure SQL Edge is ideal for:

Requirement	SQL Edge capability
Connectivity limitations	Azure SQL Edge supports solutions that work with, or without, network connectivity.
Slow or intermittent broadband connection	Azure SQL Edge provides a powerful, local database. It negates needing to forward all data to a cloud-based database, which eliminates latency.
Data security and privacy concerns	Azure SQL Edge implements RBAC and ABAC, encryption, and data classification. This helps you secure and control access to your IoT apps' data.
Synchronization and connectivity to back-end systems	Azure SQL Edge provides ease of exchanging data with other systems like Azure SQL Database, SQL Server, and Azure Cosmos DB.
Familiarity	Azure SQL Edge shares the same codebase as SQL Server. Developers with skills in SQL Server or SQL Database can reuse their code and skills

Next unit: Design for Azure Cosmos DB and tables

Continue >

100 XP

Design for Azure Cosmos DB and tables

3 minutes

Azure Cosmos DB is a fully managed NoSQL database service for modern app development. It has single-digit millisecond response times and guaranteed speed at any scale.

As a fully managed service, Azure Cosmos DB takes database administration off your hands with automatic management, updates, and patching. It also handles capacity management with cost-effective serverless and automatic scaling options that respond to application needs to match capacity with demand.

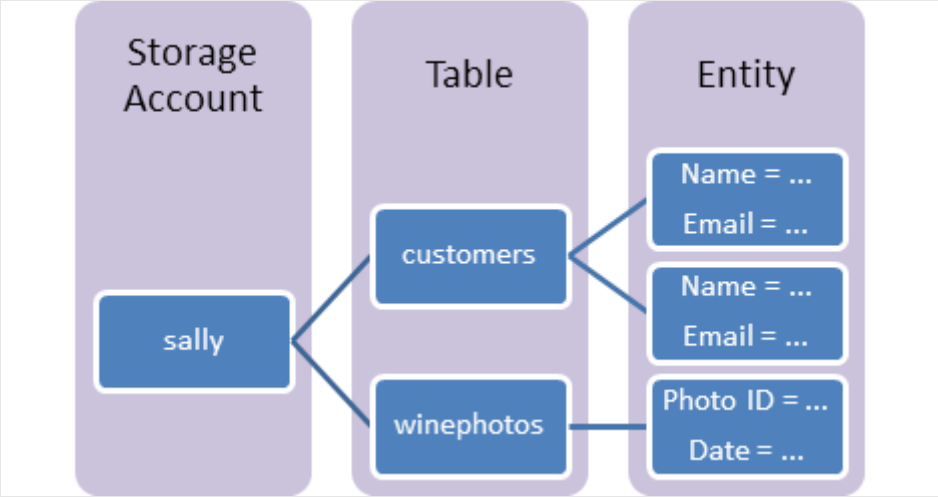
Some of the features of Azure Cosmos DB are:

- Automatic and instant scalability.
- Enterprise-grade security.
- Business continuity is assured with 99.999% SLA-backed availability.
- Turnkey multiple region data distribution anywhere in the world.
- Open-source APIs and SDKs for most popular languages.
- Takes care of database administration with automatic management, updates, and patching.
- Build fast with no-ETL analytics over operational data.

> **Note**
>
> Azure Cosmos DB is flexible and stores data in atom-record-sequence (ARS) format. The data is then abstracted and projected as an API.

Differences between Azure Storage tables and Azure Cosmos DB tables

Azure Table storage is a service that stores non-relational structured data (also known as structured NoSQL data) in the cloud, providing a key/attribute store with a schemaless design. Because Table storage is schemaless, it's easy to adapt your data as the needs of your application evolve..



Common uses of Table storage include:

- Storing TBs of structured data capable of serving web scale applications. For example, you can store any number of entities in a table, and a storage account may contain any number of tables, up to the capacity limit of the storage account.
- Storing datasets that don't require complex joins, foreign keys, or stored procedures. Example datasets include web applications, address books, device information.
- Quickly querying data using a clustered index. Access to Table storage data is fast and cost-effective for many types of applications. Table storage is typically lower in cost than traditional SQL for similar volumes of data.

Azure Cosmos DB provides the Table API for applications that are written for Azure Table storage and that need premium capabilities like high availability, scalability, and dedicated throughput.

There are some differences in behavior between Azure Storage tables and Azure Cosmos DB tables to remember if you are considering a migration. For example:

- You are charged for the capacity of an Azure Cosmos DB table as soon as it is created, even if that capacity isn't used. This charging structure is because Azure Cosmos DB uses a reserved-capacity model to ensure that clients can read data within 10 ms. In Azure Storage tables, you are only charged for used capacity, but read access is only guaranteed within 10 seconds.
- Query results from Azure Cosmos DB are not sorted in order of partition key and row key as they are from Storage tables.

- Row keys in Azure Cosmos DB are limited to 255 bytes.
- Batch operations are limited to 2 MBs.
- Cross-Origin Resource Sharing (CORS) is supported by Azure Cosmos DB.
- Table names are case-sensitive in Azure Cosmos DB. They are not case-sensitive in Storage tables.

While these differences are small, you should take care to review your apps to ensure that a migration does not cause unexpected problems.

Other benefits to moving to Cosmos DB

Applications written for Azure Table storage can migrate to the Cosmos DB Table API with few code changes. Azure Cosmos DB Table API and Azure Table storage share the same table data model and expose the same create, delete, update, and query operations through their SDKs.

If you currently use Azure Table Storage, you gain the following benefits by moving to the Azure Cosmos DB Table API:

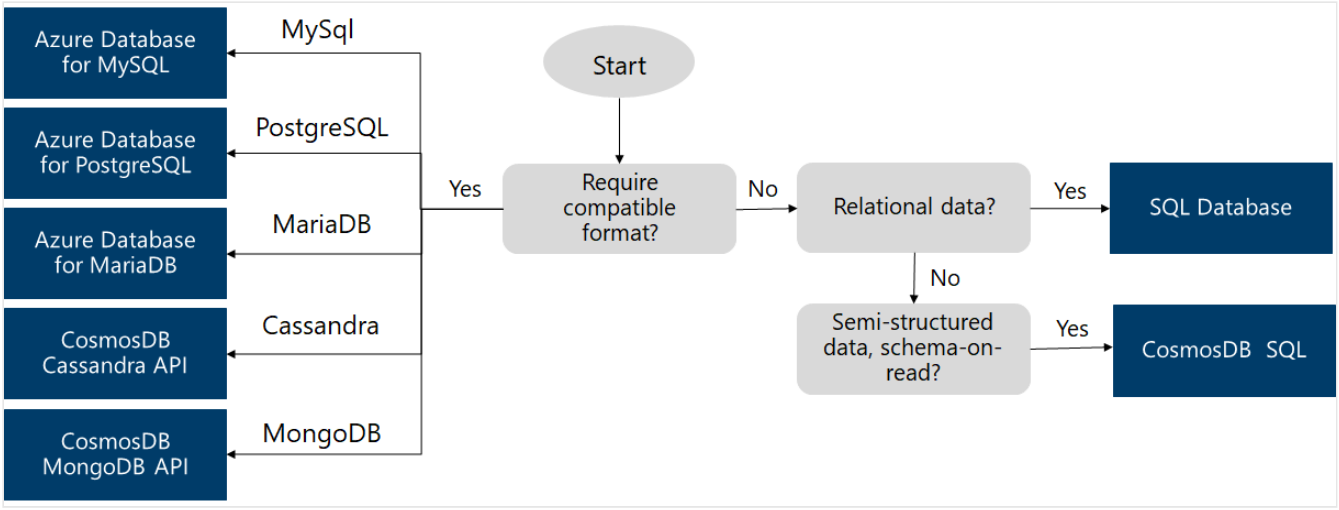
Feature	Azure Table Storage	Azure Cosmos DB Table API
Latency	Fast, but no upper bounds on latency.	Single-digit millisecond latency for reads and writes, backed with <10-ms latency reads and <15-ms latency writes at the 99th percentile, at any scale, anywhere in the world.
Throughput	Variable throughput model. Tables have a scalability limit of 20,000 operations.	Highly scalable with dedicated reserved throughput per table that's backed by SLAs. Accounts have no upper limit on throughput and support >10 million operations/s per table (in provisioned throughput mode).
Global distribution	Single region with one optional readable secondary read region for high availability.	Turnkey global distribution from one to 30+ regions.

Feature	Azure Table Storage	Azure Cosmos DB Table API
Indexing	Only primary index on PartitionKey and RowKey. No secondary indexes.	Automatic and complete indexing on all properties, no index management.
Query	Query execution uses index for primary key, and scans otherwise.	Queries can take advantage of automatic indexing on properties for fast query times.
Consistency	Strong within primary region	Five well-defined consistency levels to trade off availability, latency, throughput, and consistency.
Pricing	Consumption-based	Available in both consumption-based and provisioned capacity modes.
SLAs	99.99% availability	99.99% availability SLA for all single region accounts and all multi-region accounts with relaxed consistency, and 99.999% read availability on all multi-region database accounts.

What database APIs does Cosmos DB provide?

Azure Cosmos DB offers multiple database APIs, to provide native interface for a range of NoSQL databases. Using these APIs, Azure Cosmos DB helps you to use the ecosystems, tools, and skills you already have for data modeling and querying.

In this unit, we will examine some real-life scenarios where Core (SQL) API, API for MongoDB, Cassandra API and Gremlin API are applicable. The following diagram summarizes the workflow for selecting an appropriate data storage solution.



When to use Core (SQL) API

Let's imagine you work for a company that's an e-commerce retailer that sells automotive parts. The company stores a lot of product catalogs as JSON files. For semi-structured data storage, they've decided to migrate to Azure Cosmos DB. You need to choose an API for Cosmos DB that will:

- Enable fast scaling for new products.
- Supports current querying method of using SQL for JSON data.

Your requirements are:

- Use the team's existing skill set of SQL querying for JSON objects.
- Globally accessible data with guaranteed throughput.
- Support for adding new product categories quickly.

Based on the above criteria, you decide on Core (SQL) API to store the catalog for your customer facing e-commerce site. This API stores data in document format. Why choose Core (SQL) API?

- Single-digit millisecond latency for read and writes.
- Globally distributed database with automatic failover.
- Read and write availability with 99.999% SLA's.
- Leverage provisioned throughput or autoscale.
- Recommended for use cases such as storing cache data, session management repository, user & profile management, and product recommendation.

When to use MongoDB API

Let's imagine Tailwind Traders uses Mongo DB to store purchase orders for different home improvement products. The products are not limited to any structure and can have many attributes. Tailwind Traders would like to migrate to Azure Cosmos DB. You need to help database staff to choose the right API.

Your requirements are:

- Existing database uses MongoDB to process purchase orders.
- The operations team wants to migrate with few code changes and as little downtime as possible.
- The development team has spent lot of effort on building custom SDK.
- The volume of orders is increasing so scalability needs to be considered.
- There is a need to run analytics against real-time data for business intelligence (BI) use cases.

An effective choice here is to pick MongoDB API for Azure Cosmos DB. Azure Cosmos DB API for MongoDB implements the wire protocol for MongoDB. You can leverage your MongoDB experience and continue to use your favorite MongoDB drivers, SDKs, and tools by pointing your application to the API for MongoDB account's connection string.

The advantages of MongoDB are described in the next table.

Advantage	Description
Instantaneous scalability	The Autoscale feature allows you to scale your database up/down with zero warmup period.
Automatic and transparent sharding	The API for MongoDB manages all of the infrastructure for you. This includes sharding and the number of shards.
High Availability	99.999% availability is configurable.

Advantage	Description
Serverless deployments	This API for Azure Cosmos DB offers a serverless capacity mode. With Serverless, you are only charged per operation, and don't pay for the database when you don't use it.
Fast upgrades	All API versions are contained within one codebase, so version changes takes seconds with zero downtime.
Real time analytics at scale	This API offers the ability to run complex analytical queries for BI applications against your database data in real time. This is fast because of columnar data store and no ETL pipelines.

When to use the Cassandra API

Let's imagine you are modeling a solution for a use case involving storing of health tracker data. This also involves telemetry and sensor data. Currently this is being done in Apache Cassandra. You need to help them choose the right API for scaling their database needs.

Your requirements are:

- Your developers are currently using Cassandra Query Language (CQL), Cassandra-based tools (like cqlsh), and Cassandra client drivers
- The solution should support horizontal scaling.
- Online load balancing and cluster growth is desired.
- There should be a flexible schema.

An effective choice here is to pick Cassandra API for Azure Cosmos DB. Cassandra API is wire protocol compatible with the Apache Cassandra. This API stores data in column-oriented schema. Cassandra API currently only supports OLTP scenarios. The API supports CQL version 3.x.

> **Note**
>
> The serverless capacity mode is now available on Azure Cosmos DB's Cassandra API.

What are the advantages of Cassandra API?

The advantages of using Cassandra API are as follows:

Feature	Description
Built-in tools	Uses native Apache Cassandra features, tools, and ecosystem with the API.
Fully managed	The Cassandra API manages the OS, Java VM, garbage collection, read/write performance, nodes, and clusters. You don't need the node tool commands, such as repair and decommission, that are used in Apache Cassandra.
Regional writes	The Cassandra API allows you to choose single region or multiple region write configurations.
Integration	You can minimize latency by provisioning throughput (RUs) in the Cassandra API. You can configure Azure Cosmos containers in autoscale provisioned throughput.

Tip

The advantage of multiple region write configurations is to maintaining strong consistency across multiple regions and avoid cross-region conflict scenarios.

When to use the Gremlin API

Let's imagine you are required to analyze and provision a new non-relational database application on Azure ecosystem for Tailwind Traders. You are looking to store social media entity relationships and be able to traverse them quickly in the database.

Your requirements are:

- Store, retrieve, and manipulate graph data and visualize it using Data Explorer.
- Process high volumes of transactions without affecting performance.
- Overcome traditional graph database limitations of flexibility and relational approaches.
- Users should have capability of working with Graph query language to ingest and query data.

An effective choice here is to pick Azure Cosmos DB's Gremlin API which is based on the Apache TinkerPop. Apache TinkerPop is a graph computing framework that uses the Gremlin query language. A graph is a structure that's composed of vertices and edges. Vertices represent objects and edges denote the relationships between vertices.

Use cases for this type of database are storing organizational hierarchies, online fraud detection systems, social media graphs, and IoT. Gremlin API currently only supports online transactional processing (OLTP) scenarios.

What are the advantages of Gremlin API?

- Gremlin API has a wire protocol support with the open-source Gremlin, so you can use the open-source Gremlin SDKs to build your application.
- You can store massive graphs with billions of vertices and edges. The data will be automatically distributed using graph partitioning allowing to elastically scale throughput and storage.
- You can query the graphs with millisecond latency and evolve the graph structure easily.
- Gremlin API also works with Apache Spark and GraphFrames for complex analytical graph scenarios.
- Multi-region replication provides automatic regional failover mechanism to ensure the continuity of your application.

Next unit: Knowledge check

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

✓ 200 XP

Knowledge check

3 minutes

Tailwind Traders has several workloads being migrated to Azure. It is important you review each applications database requirement to recommend the appropriate solution:

- **Inventory application**. You plan to migrate the inventory application with a database to the cloud, but you use a third-party application that requires Windows authentication.
- **HR application.** You need to migrate the HR application and DB to the cloud and remove some of the management associated with SQL Server, but your application uses CLR and Service Broker capabilities from SQL Server.
- **Migrating databases**. There are a few large on-premises SQL server databases that need to be migrated to the cloud that contains anywhere from 40 TB to 80 TB of data.

Choose the best response for each of the questions below. Then select **Check your answers**.

1. Which Azure SQL deployment option will be easiest to use for the Inventory application?

- ○ SQL Server in an Azure virtual machine

  ✓ **that's correct. SQL Server in an Azure VM allows for Windows authentication**

- ◉ Azure SQL Managed Instance

  ✗ **that's incorrect. SQL managed instance doesn't meet the requirement of Windows authentication**

- ○ Azure SQL Database, single database

2. Which Azure SQL deployment option should be used for the HR application

- ○ Azure SQL Managed Instance

  ✓ **that's correct. Azure SQL Managed Instance is the only PaaS service that supports instance-scoped features like CLR and Service Broker.**

- ◉ SQL Server in an Azure VM

  ✗ **that's incorrect. This doesn't meet the requirements of removing management of SQL Server**

- ○ Azure SQL Database

3. What database service tier should be selected for the databases being migrated?

- ○ Hyper Scale
  - ✔ that's correct. Hyper scale can support up to 100 TB
- ◉ Business Critical
  - ✘ that's incorrect. Business critical cant support up to 80 TB of data
- ○ General purpose

Next unit: Summary and resources

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

Introduction

3 minutes

Meet Tailwind Traders

Imagine that you work as an Architect for Tailwind Traders, a company that specializes in hardware manufacturing with online sales. In addition to historical data, there is data streaming from real-time critical manufacturing processes, product quality control data, historical production logs, product volumes in stock etc.

Keeping in line with the cloud migration strategy your company is implementing, you must analyze and architect a cloud data solution designed to handle the ingestion, processing, and analysis of data that is too large and complex for traditional database systems. You have been asked to determine how best to combine data from multiple sources; reformat it into analytical models, and save these models for subsequent querying, reporting, and visualization.

Learning objectives

In this module, you'll be able to:

- Design a data integration solution with Azure Data Factory
- Design a data integration solution with Azure Data Lake
- Design a data integration and analytics solution with Azure Databricks
- Design a data integration and analytics solution with Azure Synapse Analytics
- Design a strategy for hot/warm/cold data path

- Design Azure Stream Analytics solution for Data Analysis

Skills measured

The content in the module will help you prepare for Exam AZ-305: Designing Microsoft Azure Infrastructure Solutions. The module concepts are covered in:

Design Data Storage Solutions

Design Data Integration

- Recommend a solution for data integration
- Recommend a solution for data analysis

Prerequisites

- Working experience with data integration solutions
- Conceptual knowledge of data integration solutions

Next unit: Design a data integration solution with Azure Data Factory

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design a data integration solution with Azure Data Factory

3 minutes

A significant challenge for a fast-growing home improvement retailer like Tailwind Traders is that it generates high volume of data stored in relational, non-relational and other storage systems in both the cloud and on-premises. Management wants actionable business insights from this data as near real time as possible. Additionally, sales team wants to set up and roll out up-selling and cross-selling solutions. How will you create a large-scale data ingestion solution in the cloud? What Azure services and solutions will you adopt to help with the movement and transformation of data between various data stores and compute resources?

Azure Data Factory is a cloud-based ETL and data integration service that can help you create and schedule data-driven workflows (called pipelines) that can ingest data from disparate data stores. You can use Azure Data Factory to

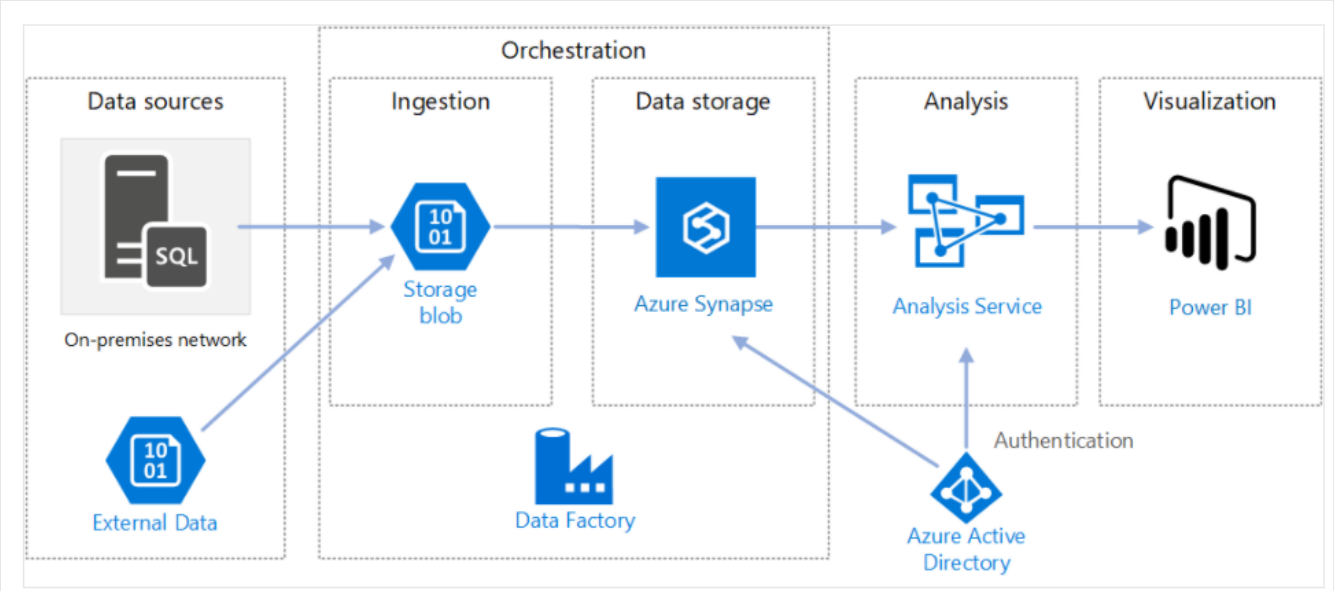
- Orchestrate data movement.
- Transform data at scale.

Data-driven workflows

There are four major steps in creating and implementing a data-driven workflow in Azure Data Factory.

1. **Connect and collect** – Data ingestion is the first step to collecting all the data from different sources into a centralized location.
2. **Transform and enrich** – Now you will use a compute service like Azure Databricks and Azure HDInsight Hadoop to transform the data
3. **Continuous integration and delivery (CI/CD) and publish** – Support for CI/CD through GitHub and Azure DevOps enables to deliver your ETL process incrementally before publishing the data to the analytics engine.
4. **Monitor** – Via the Azure portal, you can monitor the pipeline for scheduled activities and for any failures.

The following graphic shows Azure Data Factory orchestrating the ingestion of data from different data sources. Data is ingested into Storage Blob and stored in Azure Synapse Analytics. Analysis and Visualization components are also connected to Azure Data Factory. Azure data factory provides a common management interface for all of your data integration needs.

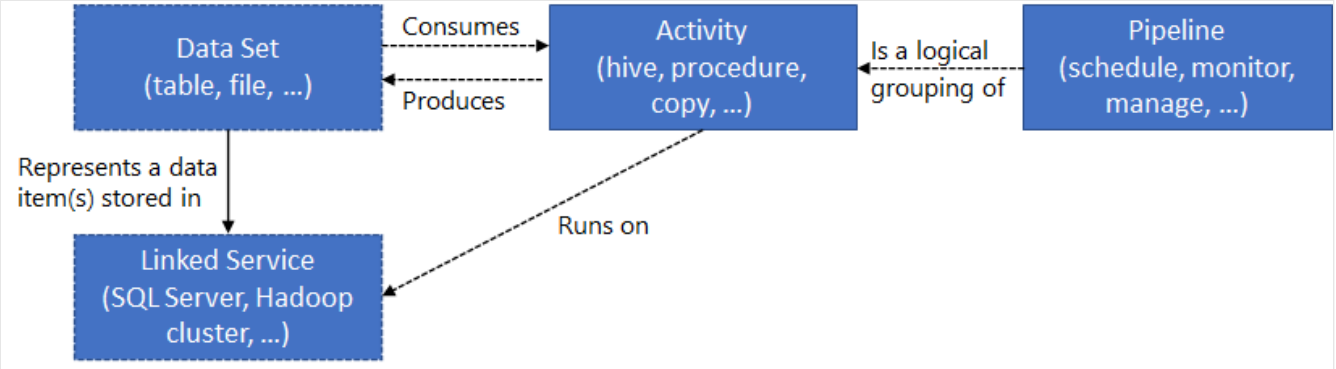


When to use Azure Data Factory

Let us evaluate Azure Data Factory against the following decision criteria:

- **Requirements for data integration** – Azure Data Factory serves two communities – Big data community and the Relational data warehousing community that uses SQL Server Integration Services (SSIS). Depending on your organization's data needs you would set up pipelines in the cloud using Azure Data Factory that can access data from both cloud and on-premises data services.
- **Coding resources** – If you prefer a graphical interface to set up pipelines, then Azure Data Factory authoring and monitoring tool is the right fit for your needs. Azure Data Factory provides a low code/no code process for working with data sources
- **Support for multiple data sources** – Azure Data Factory supports 90+ connectors to integrate with disparate data sources.
- **Serverless infrastructure –** There are advantages in using a fully managed, serverless solution for data integration - No need to maintain, configure or deploy servers, and the ability to scale with fluctuating workloads.

Components of Azure Data Factory



Azure Data factory has the following components as shown in the image above that work together to provide the platform for data movement and data integration.

- **Pipelines and activities** - A logical grouping of activities that perform a task. An activity is a single processing step in a pipeline. Azure Data Factory supports data movement, data transformation, and control activities.
- **Datasets** – These are data structures within your data stores.
- **Linked services** - Define the required connection information needed for Azure Data Factory to connect to external resources.
- **Data Flows** - Data flows allow data engineers to develop data transformation logic without writing code. Data flow activities can be operationalized using existing Azure Data Factory scheduling, control, flow, and monitoring capabilities.
- **Integration Runtimes** – It is the bridge between the activity and linked Services objects. There are three types of Integration Runtime, including Azure, Self-hosted, and Azure-SSIS.

Let's review how ADF's components are involved in a data preparation and movement scenario for Tailwind Traders. They have many different data sources to connect to and that data needs to be ingested and transformed through stored procedures that are run on the data. Finally, the data should be pushed to analytics platform for analysis.

- In this scenario, Linked Service enables Tailwind Traders to ingest data from different sources and it stores connection strings to fire up compute services on demand.
- You can execute stored procedures for data transformation that happens through Linked Service in Azure SSIS, which is the integration runtime environment for Tailwind Traders.
- The datasets components are used by the activity object and the activity object contains the transformation logic.
- You can trigger the pipeline, which is all the activities grouped together.

- You can then use Data Factory to publish the final dataset to another linked service that is consumed by technologies such as Power BI or Machine Learning.

Next unit: Design a data integration solution with Azure Data Lake

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design a data integration solution with Azure Data Lake

3 minutes

Suppose you are a data engineering consultant doing work for Tailwind Traders. They have multiple sources of data, from websites to Point of Sale (POS) systems, and more recently from social media sites to Internet of Things (IoT) devices. They are interested in using Azure to analyze all their business data. In this role, you will provide guidance on how Azure can enhance their existing BI systems. You will also offer advice about using Azure's storage capabilities to add value to their BI solution. Because of their needs, you plan to recommend Azure Data Lake Storage. Data Lake Storage provides a repository where you can upload and store huge amounts of unstructured data with an eye toward high-performance big data analytics.

Understand Azure Data Lake Storage

A data lake is a repository of data that is stored in its natural format, usually as blobs or files. Azure Data Lake Storage is a comprehensive, scalable, and cost-effective data lake solution for big data analytics built into Azure. Azure Data Lake Storage combines a file system with a storage platform to help you quickly identify insights into your data. Data Lake Storage Gen2 builds on Azure Blob storage capabilities to optimize it specifically for analytics workloads. This integration enables analytics performance, high-availability, security, and durability capabilities of Azure Storage.

> **Note**
>
> The current implementation of Azure's data lake storage service is Azure Data Lake Storage Gen2.

What are the features of Azure Data Lake Storage?

To better understand Azure Data Lake Storage, you can examine its following characteristics:

Feature	Description
Data storage	Azure Data Lake Storage can store any type of data by using the data's native format. With support for any data format and massive data sizes, Azure Data Lake Storage can work with structured, semi-structured, and unstructured data.
Data access	Azure Data Lake Storage is primarily designed to work with Hadoop and all frameworks that use the Apache Hadoop Distributed File System (HDFS) as their data access layer.
Data costs	Azure Data Lake Storage is priced at Azure Blob Storage levels.
Data performance	Azure Data Lake Storage supports high throughput for input/output–intensive analytics and data movement.
Data security	The Azure Data Lake Storage access control model supports both Azure role-based access control (RBAC) and Portable Operating System Interface for UNIX (POSIX) access control lists (ACLs).
Data redundancy	Azure Data Lake Storage utilizes Azure Blob replication models. These models provide data redundancy in a single datacenter with locally redundant storage (LRS).
Data scalability	Azure Data Lake Storage offers massive storage and accepts numerous data types for analytics.
Data analysis	Data analysis frameworks that use HDFS as their data access layer can directly access.

How Azure Data Lake Storage works

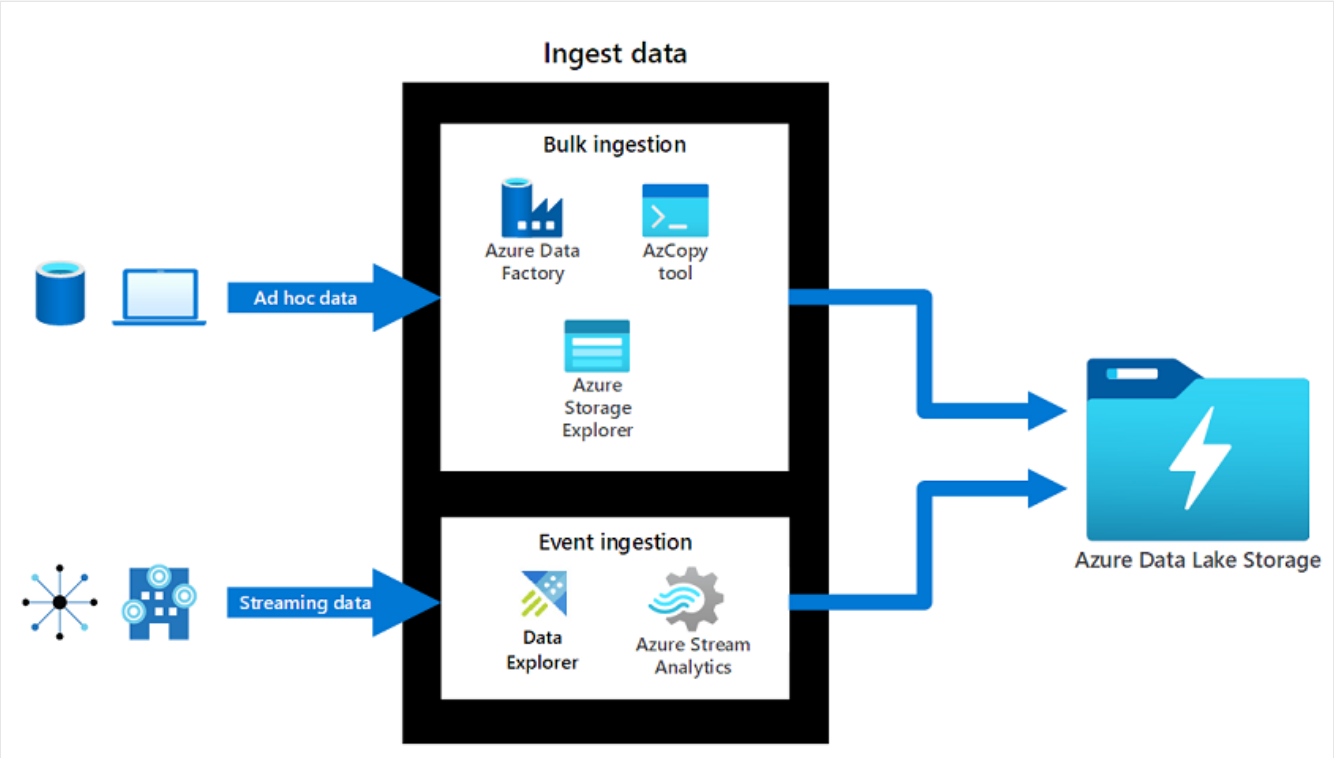
There are three important steps to use Azure Data Lake Storage. These are:

Ingesting data - Azure Data Lake Storage offers many different data ingestion methods.

- For ad hoc data -Use tools like AzCopy, CLI, PowerShell, Storage Explorer etc.
- For relational data-Use the Azure Data Factory service. It enables you to transfer data from any of the sources including Cosmos DB, SQL Database, Managed instances etc.

- For streaming data-Use tools such as Apache Storm on Azure HDInsight, Azure Stream Analytics etc.

Following diagram shows how ad hoc data and streaming data are bulk ingested or ad hoc ingested in Azure Data Lake Storage.



Accessing stored data

The easiest way to access your data is to use Azure Storage Explorer. Storage Explorer is a standalone application with a graphical user interface (GUI) for accessing your Azure Data Lake Storage data. You can also use PowerShell, Azure CLI, HDFS CLI or other programming language SDKs for accessing the data.

Setting access control

To control who can access the data stored in Azure Data Lake Storage, you can implement one or both of the following authorization mechanisms:

- Azure RBAC
- ACL

When to use Azure Data Lake Storage

Now we will consider whether Azure Data Lake Storage is the right choice for your organization's big data requirements.

Requirement	Description
When you need a Data warehouse on the cloud for managing large volumes of data	Azure Data Lake Storage runs on virtual hardware on the Azure platform, making storage scalable, fast, and reliable without incurring massive charges. It separates storage costs from compute costs. As your data volume grows, only your storage requirements change.
When you need to manage a diverse collection of data types such as JSON files, CSV, log files or other diverse formats	Azure Data Lake Storage enables data democratization for your organization by storing all your data formats (including raw data) in a single location. Eliminating data silos enables users to use a tool such as Azure Data Explorer to access and work with every data item in their storage account.
When you need real time data ingestion and storage	Azure Data Lake Storage can ingest real-time data directly from an instance of Apache Storm on Azure HDInsight, Azure IoT Hub, Azure Event Hubs, or Azure Stream Analytics. It also works with semi-structured data and lets you ingest all your real-time data into your storage account.

When to choose Azure Blob Storage over Azure Data Lake

Here are some criteria that will help you decide when to pick one storage solution over the other. In the following table, the two storage solutions are compared against a set of criteria.

Criteria	Azure Data Lake	Azure Blob Storage
Data type	Good for storing large volumes of text data	Good for storing unstructured non-text based data such as photos, videos, backup etc.
Geographic redundancy	Need to set up replication of data	By default, provides geo redundant storage
Namespaces support	Supports hierarchical namespaces	Supports flat namespaces
Hadoop compatibility	Hadoop services can use data stored in Data Lake	Is not Hadoop compatible
Security	Allows for more granular access	Granular access not supported

100 XP

Design a data integration and analytic solution with Azure Databricks

3 minutes

Azure Databricks is a fully managed, cloud-based Big Data and Machine Learning platform, which empowers developers to accelerate AI and innovation. Azure Databricks provides data science and engineering teams with a single platform for Big Data processing and Machine Learning. Azure Databricks' managed Apache Spark platform makes it simple to run large-scale Spark workloads.

Azure Databricks offers three environments for developing data intensive applications: Databricks SQL, Databricks Data Science & Engineering, and Databricks Machine Learning.

Environment	Description
Databricks SQL	Provides an easy-to-use platform for analysts who want to run SQL queries on their data lake, create multiple visualization types to explore query results from different perspectives, and build and share dashboards.
Databricks Data Science & Engineering	Provides an interactive workspace that enables collaboration between data engineers, data scientists, and machine learning engineers. For a big data pipeline, the data (raw or structured) is ingested into Azure through Azure Data Factory in batches, or streamed near real-time using Apache Kafka, Event Hub, or IoT Hub. This data lands in a data lake for long term persisted storage, in Azure Blob Storage or Azure Data Lake Storage. As part of your analytics workflow, use Azure Databricks to read data from multiple data sources and turn it into breakthrough insights using Spark.
Databricks Machine Learning	An integrated end-to-end machine learning environment incorporating managed services for experiment tracking, model training, feature development and management, and feature and model serving.

Let us analyze a situation for Tailwind Traders, a heavy machinery manufacturing organization and arrive at the best Azure Databricks environment that is applicable for them. Tailwind Traders is leveraging Azure cloud services for their big data needs – the data is both batch and

streaming data. They have data engineers, data scientists and data analysts who collaborate to produce quick insightful reporting for many stakeholders.

In the above scenario, you would choose Databricks Data Science and Engineering environment because:

- It provides an integrated Analytics 'workspace' based on Apache Spark that allows collaboration between different users
- Using Spark components like Spark SQL and Dataframes it can handle structured data and integrates with real-time data ingestion tools like Kafka and Flume for processing streaming data
- Secure data integration capabilities built on top of Spark that enable you to unify your data without centralization and Data scientists can visualize data in a few clicks, and use familiar tools like Matplotlib, ggplot, or d3.
- The runtime abstracts out the infrastructure complexity and the need for specialized expertise to set up and configure your data infrastructure so users can use the languages they are skilled in like Python, Scala, R and explore the data
- This also integrates deeply with Azure databases and stores: Synapse Analytics, Cosmos DB, Data Lake Store, and Blob storage for Tailwind Traders big data storage needs
- Integration with Power BI allows for quick and meaningful insights which is a requirement for Tailwind Traders.
- Databricks SQL is not the right choice since it cannot handle unstructured data
- Databricks Machine Learning is also not the right environment choice since machine learning is not their requirement right now.

How Azure Databricks works

Azure Databricks has a Control plane and a Data plane.

- The Control Plane hosts Databricks jobs, notebooks with query results, and the cluster manager. The Control plan also has the web application, hive metastore, and security access control lists (ACLs) and user sessions. These components are managed by Microsoft in collaboration with Databricks and do not reside within your Azure subscription.
- The Data Plane contains all the Databricks runtime clusters hosted within the workspace. All data processing and storage exists within the client subscription. This means no data

processing ever takes place within the Microsoft/Databricks-managed subscription.

When to use Azure Databricks

Azure Databricks is entirely based on Apache Spark and as such is a great tool for those already familiar with the open-source cluster-computing framework. As a unified analytics engine, it's designed specifically for big data processing and data scientists can take advantage of built-in core API for core languages like SQL, Java, Python, R, and Scala.

You can use Azure Databricks as a solution for the following scenarios.

- **Data science preparation of data** - Create, clone and edit clusters of complex, unstructured data, turn them into specific jobs and deliver them to data scientists and data analysts for review
- **Find relevant insights in the data** - Recommendation engines, churn analysis, and intrusion detection are common scenarios that many organizations solve across multiple industries using Azure Databricks.
- **Improve productivity across data and analytics teams** - Create a collaborative environment and shared workspaces for data engineers, analysts, and scientists to work together across the data science lifecycle with shared workspaces, saving teams precious time and resources.
- **Big data workloads** – Leverage Data Lake and engine to get the best performance and reliability for your big data workloads, and to create no-fuss multi-step data pipelines
- **Machine learning programs** – Leverage integrated end-to-end machine learning environment incorporating managed services for experiment tracking, model training, feature development and management, and feature and model serving

---

Next unit: Design a data integration and analytic solution with Azure Synapse Analytics

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

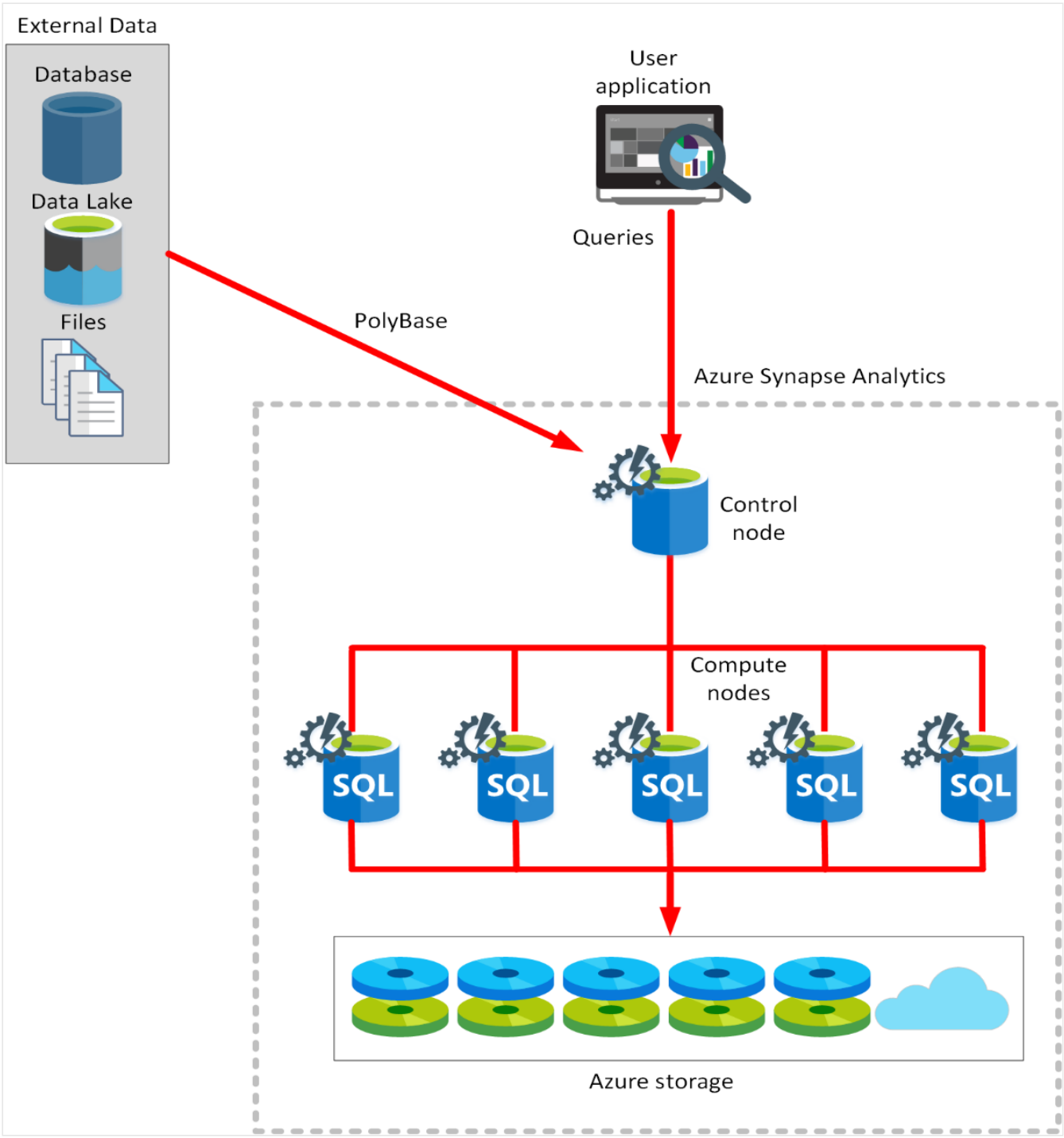
Design a data integration and analytic solution with Azure Synapse Analytics

3 minutes

Tailwind Traders is serving clients with stock market information. You need a combination of batch and stream processing. The up-to-the-second data might be used to help monitor real-time where an instant decision is required to make informed split-second buy or sell decisions. Historical data is equally important for a view of trends in performance. What kind of data warehouse and data integration solution you would recommend that provides access to the streams of raw data, and the cooked business information derived from this data?

With Azure Synapse Analytics, you can ingest data from external sources and then transform and aggregate this data into a format suitable for analytics processing.

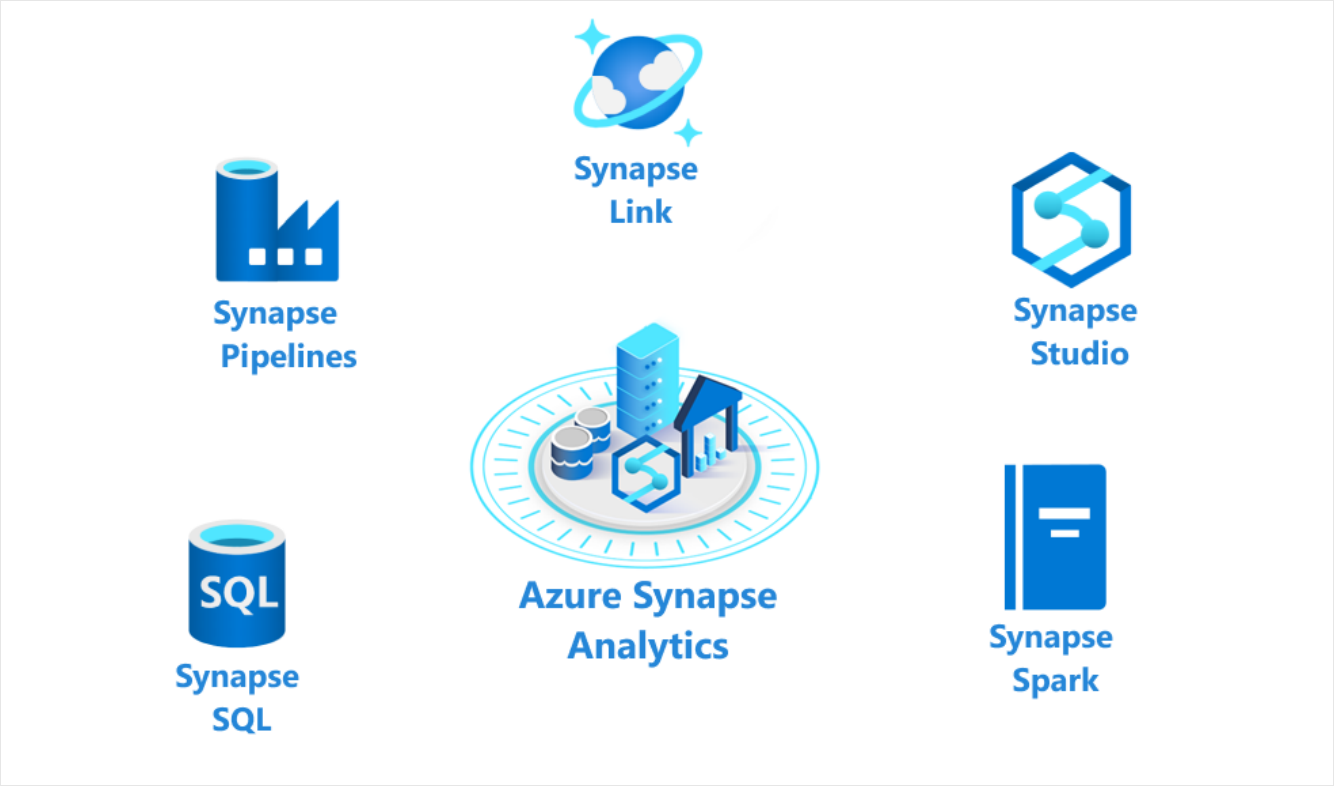
In this unit we will explore the high-level architecture and component parts of Azure Synapse Analytics and how to get started using Azure Synapse Analytics for data integration.



Azure Synapse Analytics leverages a massively parallel processing (MPP) architecture. This architecture includes a control node and a pool of compute nodes.

The Control node is the brain of the architecture. It's the front end that interacts with all applications. The Compute nodes provide the computational power. The data to be processed is distributed evenly across the nodes. You submit queries in the form of Transact-SQL statements, and Azure Synapse Analytics runs them. Azure Synapse Analytics uses a technology named PolyBase that enables you to retrieve and query data from relational and non-relational sources. You can save the data read in as SQL tables within the Synapse Analytics service.

Components of Azure Synapse Analytics



Azure Synapse Analytics is composed of the following elements:

- **Synapse SQL pool**: Synapse SQL offers both serverless and dedicated resource models to work with using node-based architecture. For predictable performance and cost, you can create dedicated SQL pools, for unplanned or ad hoc workloads, you can use the always-available, serverless SQL endpoint.

- **Synapse Spark pool**: This is a cluster of servers running Apache Spark to process data. You write your data processing logic using one of the four supported languages: Python, Scala, SQL, and C# (via .NET for Apache Spark). Apache Spark for Azure Synapse integrates Apache Spark-the open source big data engine used for data preparation, data engineering, ETL, and machine learning.

- **Synapse Pipelines**: Azure Synapse Pipelines leverages the capabilities of Azure Data Factory and is the cloud-based ETL and data integration service that allows you to create data-driven workflows for orchestrating data movement and transforming data at scale. You could include activities that transform the data as it is transferred, or you might combine data from multiple sources together.

- **Synapse Link**: This component allows you to connect to Cosmos DB. You can use it to perform near real-time analytics over the operational data stored in a Cosmos DB database.

- **Synapse Studio**: This is a web-based IDE that can be used centrally to work with all capabilities of Azure Synapse Analytics. You can use Synapse Studio to create SQL and Spark pools, define and run pipelines, and configure links to external data sources.

> ⓘ **Note**
>
> Read about **Analytics end-to-end with Azure Synapse**

Analytics end-to-end with Azure Synapse

When to use Azure Synapse Analytics?

- When you have a variety of data sources use Azure Synapse Analytics for code-free ETL and data flow activities.
- When you have a need to implement Machine Learning solutions using Apache Spark, use Azure Synapse Analytics for built-in support for AzureML.
- When you have existing data stored on a data lake and need integration with the Data Lake and additional input sources, Azure Synapse Analytics provides seamless integration between the two.
- When management needs real-time analytics, you can use features like Azure Synapse Link to analyze in real-time and offer insights.

What kind of analytics can you do with Azure Synapse Analytics?

Following table shows the range of analytical types that Azure Synapse Analytics supports:

Analytics type	Description
Descriptive analytics - "What is happening?"	Azure Synapse Analytics leverages the dedicated SQL pool capability that enables you to create a persisted data warehouse to perform this type of analysis. You can also make use of the serverless SQL pool to prepare data from files stored in a data lake to create a data warehouse interactively.

Analytics type	Description
Diagnostic analytics - "Why is it happening?"	You can use the serverless SQL pool capability within Azure Synapse Analytics that enables you to interactively explore data within a data lake. Serverless SQL pools can quickly enable a user to search for additional data that may help them to understand why questions.
Predictive analytics - "What is likely to happen?"	Azure Synapse Analytics uses its integrated Apache Spark engine and Azure Synapse Spark pools for predictive analytics with other services such as Azure Machine Learning Services, or Azure Databricks.
Prescriptive analytics - "What needs to be done?"	This type of analytics uses predictive analytics real-time or near real-time data. Azure Synapse Analytics provides this capability through both Apache Spark, Azure Synapse Link, and by integrating streaming technologies such as Azure Stream Analytics.

When to choose Azure Data Factory over Azure Synapse Analytics

Here are some criteria that will help you decide when to pick Azure Data Factory solution over Azure Synapse Analytics. In the following table, the two solutions are compared against a set of criteria.

Criteria	Azure Data Factory	Azure Synapse Analytics
Data sharing	Can be shared across different data factories	No sharing of data
Solution templates	Provided with Azure Data Factory template gallery	Provided with Synapse Workspace Knowledge center
Integration Runtime cross region support	Support Cross region data flows	Does not support cross region data flows
Monitoring	Integrated with Azure Monitor	Diagnostic logs in Azure Monitor

Criteria	Azure Data Factory	Azure Synapse Analytics
Monitoring of Spark Jobs for Data Flow	Not supported	Supported by the Synapse Spark pools

Next unit: Design a strategy for hot, warm, cold data path

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design a strategy for hot, warm, cold data path

3 minutes

Traditionally data was stored on premises without taking into consideration its usage and lifecycle. In the cloud, data can be stored based on access, lifecycle, and other compliance requirements. Let us learn how the concept of hot, warm, and cold data path determines how we store and compute data.

Understand Warm Path

Let us explore a common scenario for IoT device data aggregation. The devices could be sending data while not really producing anything. This highlights a very common challenge when trying to extract insight out of IoT data: the data you are looking for is not available in the data you are getting. Therefore, we need to infer utilization by combining the data we are getting with other sources of data, and applying rules to determine of whether or not the machine is producing. In addition, these rules may change from company to company since they may have different interpretations of what “producing” is.

The warm data path is about analyzing as the data flows through the system. We process this stream in near-real time, save it to the warm storage, and push it to the analytics clients.

The Azure platform provides many options for processing the events and one popular choice is Stream Analytics.

Stream Analytics can execute complex analysis at scale, for example, tumbling/sliding/hopping windows, stream aggregations, and external data source joins. For even more complex processing, performance can be extended by cascading multiple instances of Event Hubs, Stream Analytics jobs, and Azure functions.

Warm storage can be implemented with various services on the Azure platform, such as Azure SQL Database and Azure Cosmos DB.

Understand Cold Path

The warm path is where the stream processing occurs to discover patterns over time. However, there might be a need to calculate the utilization over a period in the past, with different pivots, and aggregations and to merge those results with the warm path results to present a unified view to the user.

The cold path includes the batch layer and the serving layers. The combination provides a long-term view of the system.

The cold path contains the long-term data store for the solution. It also contains the batch layer, which creates pre-calculated aggregate views to provide fast query responses over long periods. The technology options available for this layer on Azure platform is quite diverse.

Azure Storage is a good solution for the cold storage. Azure Storage includes Azure Blobs (objects), Azure Data Lake Storage Gen2, Azure Files, Azure Queues, and Azure Tables. The cold storage can be either Blobs, Data Lake Storage Gen2, or Azure Tables., or a combination of those.

To store massive amounts of unstructured data, such as JSON, or XML documents containing the unprocessed data received by the IoT application, Blob storage, Azure Files, or Azure Data Lake Storage Gen2 are the best options.

Azure Data Factory is a great solution for creating the batch views on the serving layer. It is a cloud-based managed data integration service that allows you to create data-driven workflows in the cloud for orchestrating and automating data movement and data transformation. It can process and transform the data by using services such as Azure HDInsight Hadoop, Spark, and Azure Databricks. This allows you to build machine learning models and consume them with the analytics clients.

Understand Hot Path

Hot path is typically used for processing or displaying data in real-time. It is employed for real time alerting and streaming operations are performed using this data. **A hot path** is where latency-sensitive data, results need to be ready in seconds or less and it flows for rapid consumption by analytics clients.

When to use Hot/Warm/Cold data path

Let us analyze the requirements where you will decide upon hot, warm, or cold data path.

Requirement	Description

Requirement	Description
When data requirements are known to change frequently When processing or displaying data in real time	Use Hot data path
When data is rarely used. The data might be stored for compliance or legal reasons Used for data that is consumed for long term analytics and batch processing	Use Cold data path
When you need to store or display a recent subset of data Used for data that is consumed for small analytical and batch processing	Use Warm data path

Next unit: Design Azure Stream Analytics solution for data analysis

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Design Azure Stream Analytics solution for data analysis

3 minutes

Tailwind Traders, a fictitious home improvement retailer has begun the digital transformation of their applications and services to help with the growth of the company. A present scenario involves accessing, storing and doing analysis on sensor data from the GPS on their delivery trucks that are on the road delivering goods. Management wants a solution for being able to do real time analytics on the streaming data from the GPS on the trucks and make decisions in real time.

On further analysis, you find that they would like this data present in an existing Power BI visualization dashboard.

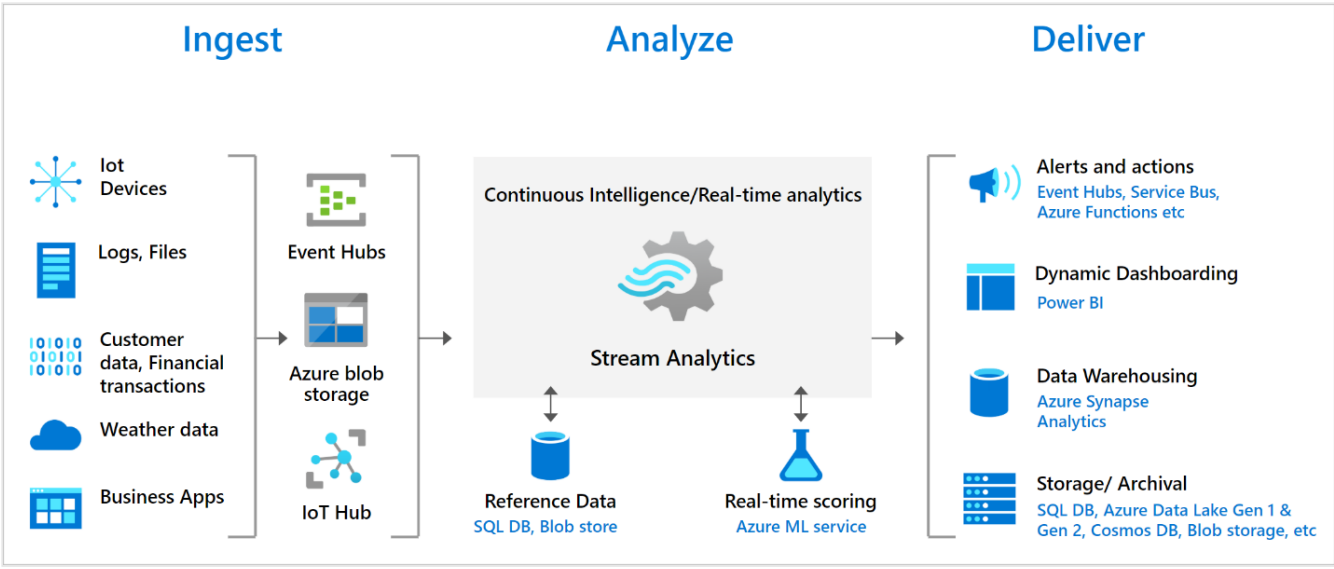
How will you help address this problem? Azure Stream Analytics can help in this scenario.

Describe Azure Stream Analytics

The process of consuming data streams, analyzing them, and deriving actionable insights is called stream processing. Azure Stream Analytics is a fully managed (PaaS offering), real-time analytics and complex event-processing engine. It offers the possibility to perform real-time analytics on multiple streams of data from sources such as IoT device data, sensors, clickstreams and social media feeds.

Azure Stream Analytics works on the following concepts:

- **Data stream**- Data streams are continuous data generated by applications, IoT devices or sensors. These data streams are analyzed, and actionable insights are extracted from it. For example, monitoring data stream from industrial and manufacturing equipment, Monitoring data of water pipelines by utility providers. Data streams help understand change over time.
- **Event processing** - Event processing refers to consumption and analysis of a continuous data stream to extract actionable insights from the events happening within that stream. For example, a car passing through a tollbooth should include temporal information, such as a timestamp, indicating when it occurred.



The image shows the Stream Analytics pipeline, and how data is ingested, analyzed, and then sent for presentation or action.

Key features of Azure Stream Analytics

Stream Analytics ingests data from Azure Event Hubs (including Azure Event Hubs from Apache Kafka), Azure IoT Hub, or Azure Blob Storage. The query, which is based on SQL query language, can be used to easily filter, sort, aggregate, and join streaming data over a period. You can also extend this SQL language with JavaScript and C# user-defined functions (UDFs). You can understand stream processing by reading more about it.

An Azure Stream Analytics job consists of an input, query, and an output. You can do the following with the job output:

- Route data to storage systems like Azure Blob storage, Azure SQL Database, Azure Data Lake Store, and Azure CosmosDB.
- You can send data to Power BI for real-time visualization.
- You can store data in Data Warehouse service like Azure Synapse Analytics to train a machine learning model based on historical data or perform batch analytics.
- You can trigger custom downstream workflows by sending the data to services like Azure Functions, Service Bus Topics or Queues.

> **Note**
>
> You can get some information about **Overview of Azure Stream Analytics Cluster** for single-tenant deployment of Stream Analytics.

When to consider Azure Stream Analytics

Following are some common enterprise data use-cases where Azure Stream Analytics can be effectively leveraged.

Use case	Azure Stream Analytics consideration
Analyze real-time telemetry streams from IoT devices	Gathering real-time sensor data in Azure Stream Analytics from building automation systems relaying temperature, humidity, fan runtimes and making adjustments to maintain optimum building temperature while reducing costs.
Web logs/clickstream analytics	A consumer goods retailer can offer real-time product suggestions to users based on e-commerce analytics.
Geospatial analytics	Data sources include sensors, social media, satellite imagery, mobile devices etc. For example, extreme weather prediction of wildfires and hurricanes can help airlines with routing and send out alerts of adverse weather conditions to people.
Remote monitoring and predictive maintenance of high value assets	Monitoring high value assets such as Industrial equipment by gathering operational data in Azure Stream Analytics and maximizing the useful life of their equipment can be achieved through predictive maintenance to avoid disruption of operation. For example, data gathered from electrical power transformers used by utility companies.
Real-time analytics on Point of Sale data	Credit card fraudulent transactions can be detected and determined at point of sale depending on unusually large transaction or unusual location usage of the credit card. Alert triggers can be set up on data gathered in Azure Stream Analytics.

In the Tailwind Traders scenario discussed above, we can leverage Azure Stream Analytics to visualize real-time locations of the trucks through Power BI. For management decisions on analytical workloads, data can be stored in Data warehouse like Cosmos DB, or Azure Data Lake for future analysis.

> **ⓘ Important**
>
> Azure Stream Analytics supports processing events in CSV, JSON and Avro data formats.

Benefits of Azure Stream Analytics

- Fully Managed - It is offered as a Platform as a Service offering, so there is no overhead of provisioning any hardware or infrastructure. Azure Stream Analytics fully manages your job, so you can focus on your business logic and not on the infrastructure.
- Low cost - Billing is done by Streaming Units (SUs) consumed which represent the amount of CPU and memory resources allocated. Scaling up and down are based on business needs which can also bring cost down. No maintenance costs are involved.
- Run in cloud or Intelligent edge - It can run in the cloud, for large-scale analytics, or run on IoT Edge or Azure Stack for ultra-low latency analytics.
- Performance - Stream Analytics offers reliable performance guarantees. It supports higher performance by partitioning, allowing complex queries to be parallelized and executed on multiple streaming nodes. Stream Analytics can process millions of events every second and it can deliver results with ultra-low latencies.
- Security - In terms of security, Azure Stream Analytics encrypts all incoming and outgoing communications and supports TLS 1.2. Built-in checkpoints are also encrypted. Stream Analytics does not store the incoming data since all processing is done in-memory.

Next unit: Knowledge check

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

✓ 200 XP

Knowledge check

3 minutes

Tailwind Traders has several workloads being migrated to Azure. It is important the data solutions are designed to handle the ingestion, processing, and analysis of data based on the following requirements:

- **Manufacturing Data.** The company has been collecting manufacturing logs that are collected from the assembly line. They want to analyze these logs to gain insights into material behavior and quality assurance. To analyze these logs, the company needs to use reference data such as material information, chemical information, and origin information that is in an on-premises data store. The company wants to utilize this data from the on-premises data store, combining it with other log data that it has in a cloud data store and run stored procedures on the data to gain insights.
- **Real time Data.** Tailwind Traders needs real-time data ingestion and storage for multiple data sources like their websites, point of sale systems and social media sites. They need a solution to analyze this data and provide useful insights to the CEO.
- **Historical company data.** The company is required to store 5 TB of company data for legal reasons. This data is rarely used or referenced but must not be deleted. They need a cost effective method for storing this data.

Choose the best response for each of the questions below. Then select **Check your answers**.

1. Based on the manufacturing data requirement what solution meets the requirements?

○ Azure Data Factory

✔ **that's correct. Azure Data Factory can connect different data sources and run stored procedures on the data.**

◉ Azure Databricks

✘ **that's incorrect. Azure Databricks does not meet the requirement for the manufacturing data.**

○ Azure Data Lake

2. Based on the real time data requirements what solution provides real-time data ingestion and storage of multiple data sources?

○ Azure Data Lake

✔ that's correct. Azure Data Lake can ingest real-time data directly from multiple sources.

○ Azure Databricks

◉ Azure Blob Storage

✘ that's incorrect. Azure Databricks does not meet the requirement for real-time data ingestion from multiple data sources.

3. Based on the historical company data requirements what solution meets the requirements?

○ Cold storage

✔ that's correct. Cold storage is the appropriate solution for data that is rarely used

○ Hot storage

◉ Warm storage

✘ that's incorrect. Warm storage is not the best solution for data that is rarely accessed.

Next unit: Summary and resources

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆