Azure API Training <> Revanture

Day 4

Azure Storage Account

What is an Azure Storage Account?

An Azure storage account contains all of your Azure Storage data objects: blobs, files, queues, and tables.

The storage account provides a unique namespace for your Azure Storage data that's accessible from anywhere in the world over HTTP or HTTPS.

Data in your storage account is durable and highly available, secure, and massively scalable.

Project

Create a Storage Account

Full forms

- Zone Redundant Storage (ZRS)
- Locally Redundant Storage (LRS)
- Geo-redundant Storage (GRS/RA-GRS)

Types of Storage Account

Type of storage account	Supported storage services	Redundancy options	Usage
Standard general-purpose v2	Blob Storage (including Data Lake Storage[1]), Queue Storage, Table Storage, and Azure Files	Locally redundant storage (LRS) / geo-redundant storage (GRS) / read-access geo-redundant storage (RA-GRS) Zone-redundant storage (ZRS) / geo-zone-redundant storage (GZRS) / read-access geo-zone-redundant storage (RA-GZRS)[2]	Standard storage account type for blobs, file shares, queues, and tables. Recommended for most scenarios using Azure Storage. If you want support for network file system (NFS) in Azure Files, use the premium file shares account type.
Premium block blobs[3]	Blob Storage (including Data Lake Storage[1])	LRS ZRS[2]	Premium storage account type for block blobs and append blobs. Recommended for scenarios with high transaction rates or that use smaller objects or require consistently low storage latency. Learn more about example workloads.
Premium file shares[3]	Azure Files	LRS ZRS[2]	Premium storage account type for file shares only. Recommended for enterprise or high-performance scale applications. Use this account type if you want a storage account that supports both Server Message Block (SMB) and NFS file shares.
Premium page blobs[3]	Page blobs only	LRS ZRS[2]	Premium storage account type for page blobs only. Learn more about page blobs and sample use cases.

Standard General-purpose v2

You can store

- Images, videos, documents
- Logs and analytics data
- Background job messages
- Database-like tables

Premium Block Blobs

You can store:

- Videos, large images, or frequently accessed files
- Files that need quick upload/download

Premium File Shares

You can store:

- Application data shared between multiple servers
- Config files or reports generated by teams

Premium Page Blobs

You can store

- OS disks and data disks for virtual machines

Ideal Use Cases for Storage Accounts

Workload	Account kind	Performance	Redundancy	Hierarchical namespace enabled	Default access tier	Soft delete enabled
Cloud native	General purpose v2	Standard	ZRS, RA-GRS	No	Hot	Yes
Analytics	General purpose v2	Standard	ZRS[1], RA-GRS	Yes[2]	Hot	Yes
High performance computing (HPC)	General purpose v2	Standard	ZRS, RA-GRS	Yes	Hot	Yes
Backup and archive	General purpose v2	Standard	ZRS, RA-GRS	No	Cool[3]	Yes
Machine learning and artificial intelligence	General purpose v2	Standard	ZRS, RA-GRS	Yes	Hot	No

Standard endpoints

Storage service	Endpoint
Blob Storage	`https://<storage-account>.blob.core.windows.net`
Static website (Blob Storage)	`https://<storage-account>.web.core.windows.net`
Data Lake Storage	`https://<storage-account>.dfs.core.windows.net`
Azure Files	`https://<storage-account>.file.core.windows.net`
Queue Storage	`https://<storage-account>.queue.core.windows.net`
Table Storage	`https://<storage-account>.table.core.windows.net`

Containers

A container organizes a set of blobs, similar to a directory in a file system. A storage account can include an unlimited number of containers, and a container can store an unlimited number of blobs.

A container name must be a valid DNS name, as it forms part of the unique URI (Uniform resource identifier) used to address the container or its blobs. Follow these rules when naming a container:

- Container names can be between 3 and 63 characters long.
- Container names must start with a letter or number, and can contain only lowercase letters, numbers, and the dash (-) character.
- Two or more consecutive dash characters aren't permitted in container names.

Blobs

Azure Storage supports three types of blobs:

- **Block blobs** store text and binary data. Block blobs are made up of blocks of data that can be managed individually. Block blobs can store up to about 190.7 TiB.
- **Append blobs** are made up of blocks like block blobs, but are optimized for append operations. Append blobs are ideal for scenarios such as logging data from virtual machines.
- **Page blobs** store random access files up to 8 TiB in size. Page blobs store virtual hard drive (VHD) files and serve as disks for Azure virtual machines. For more information about page blobs, see Overview of Azure page blobs

Block Blobs

Imagine an online store like Amazon. Each product has:

- Product images
- Specification PDFs
- Promotional videos

Append Blobs

E-commerce Example

- Let’s say your platform logs user activity or transactions:
- Every time someone places an order or cancels one, an entry is written to a log file.

You don’t want to overwrite previous logs — just append new entries.

Page Blobs

E-commerce Example:

Suppose your system hosts a database on a virtual machine — e.g., your product catalog or customer database.

That VM's disk file (VHD) is stored as a page blob.

So, /vhds/ecommerce-db.vhd is a page blob that Azure uses to run the VM efficiently — fast reads/writes to random parts of the file.

Access Tier for Storage Accounts

- **Hot tier** - An online tier optimized for storing data that is accessed or modified frequently. The hot tier has the highest storage costs, but the lowest access costs.
- **Cool tier** - An online tier optimized for storing data that is infrequently accessed or modified. Data in the cool tier should be stored for a minimum of 30 days. The cool tier has lower storage costs and higher access costs compared to the hot tier.
- **Cold tier** - An online tier optimized for storing data that is rarely accessed or modified, but still requires fast retrieval. Data in the cold tier should be stored for a minimum of 90 days. The cold tier has lower storage costs and higher access costs compared to the cool tier.
- **Archive tier** - An offline tier optimized for storing data that is rarely accessed, and that has flexible latency requirements, on the order of hours. Data in the archive tier should be stored for a minimum of 180 days.

Azure Blob Storage lifecycle management

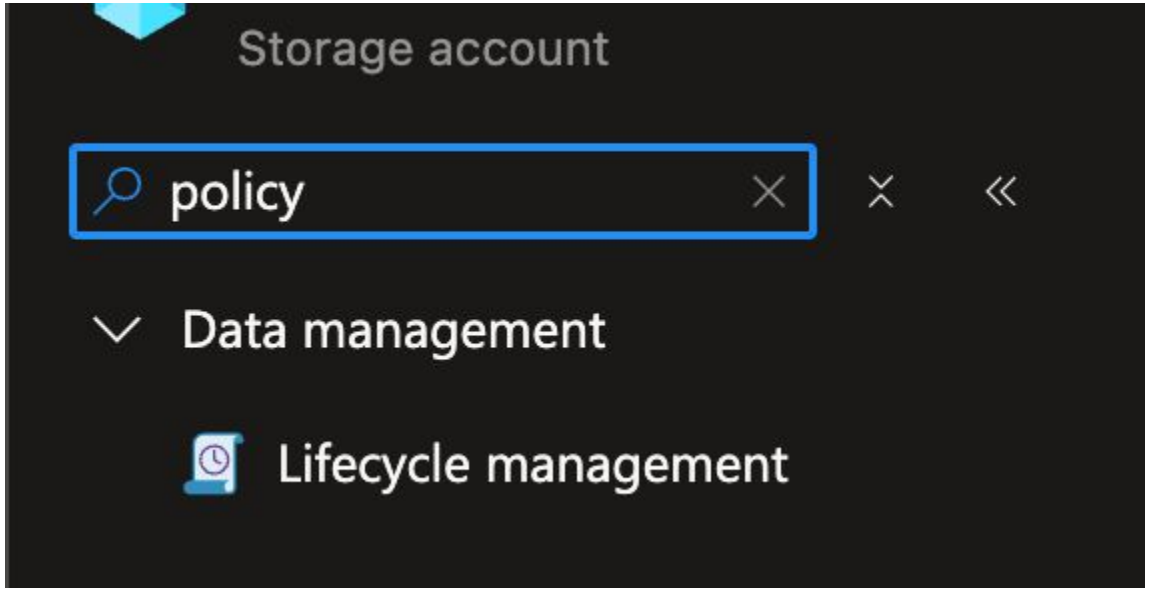
- Azure Blob Storage empowers organizations to efficiently manage and scale their data storage needs, even as data volumes grow, and usage patterns evolve. By using blob lifecycle management, customers can proactively optimize costs by implementing rule-based policies that automatically transition data to cooler tiers or expire it when it's no longer needed.

Azure Blob Storage lifecycle management

With the lifecycle management policy, you can:

- Transition current versions of a blob, previous versions of a blob, or blob snapshots to a cooler storage tier if these objects aren't accessed or modified for a period of time, to optimize for cost.
- Transition blobs back from cool to hot immediately when they're accessed.
- Delete current versions of a blob, previous versions of a blob, or blob snapshots at the end of their life cycles.
- Apply rules to an entire storage account, to select containers, or to a subset of blobs using name prefixes or blob index tags as filters.

Azure Blob Storage lifecycle management



Cost Optimization

- **Access tier optimization** – Use Hot tier for frequently accessed data, Cool tier for infrequently accessed data, and Archive tier for long-term retention
- **Lifecycle management policies** – Implement automated policies to transition blobs between tiers based on age or access patterns
- **Reserved capacity** – Purchase reserved capacity for predictable storage needs to achieve significant cost savings
- **Data compression** – Compress data before storing to reduce storage volume and associated costs
- **Redundancy level selection** – Choose appropriate redundancy (LRS, ZRS, GRS, RA-GRS) based on durability requirements vs cost trade-offs

Security and Access Control

Data protection

- Use the Azure Resource Manager deployment model
- Enable Microsoft Defender for all of your storage accounts
- Turn on soft delete for blobs
- Turn on soft delete for containers
- Lock storage account to prevent accidental or malicious deletion or configuration changes
- Store business-critical data in immutable blobs
- Use Encryption to Protect Data
- Require secure transfer (HTTPS) to the storage account
- Limit shared access signature (SAS) tokens to HTTPS connections only
- Disallow cross-tenant object replication

Identity and access management

- Use Microsoft Entra ID
- RBAC for Blobs
- Issue SAS Token
- Secure your account access keys with Azure Key Vault
- Regenerate your account keys periodically
- Disallow Shared Key authorization
- Disable anonymous read access to containers and blobs

Networking

- Configure the minimum required version of Transport Layer Security (TLS) for a storage account.
- Enable the Secure transfer required option on all of your storage accounts
- Enable firewall rules
- Allow trusted Microsoft services to access the storage account
- Use private endpoints
- Limit network access to specific networks

Logging/Monitoring

- Track how requests are authorized
- Set up alerts in Azure Monitor

Project

SAS Tokens and Access Policies

Project

Upload Blob with Python

Uploading a blob via python

- Upload a blob via python Source Code

Metadata and Custom Properties

Metadata and Custom Properties

- Containers and blobs support custom metadata, represented as HTTP headers. Metadata headers can be set on a request that creates a new container or blob resource, or on a request that explicitly creates a property on an existing resource.

Metadata Header Format

- Metadata headers are name/value pairs. The format for the header is:

  x-ms-meta-name:string-value

Project

Metadata and Custom Header

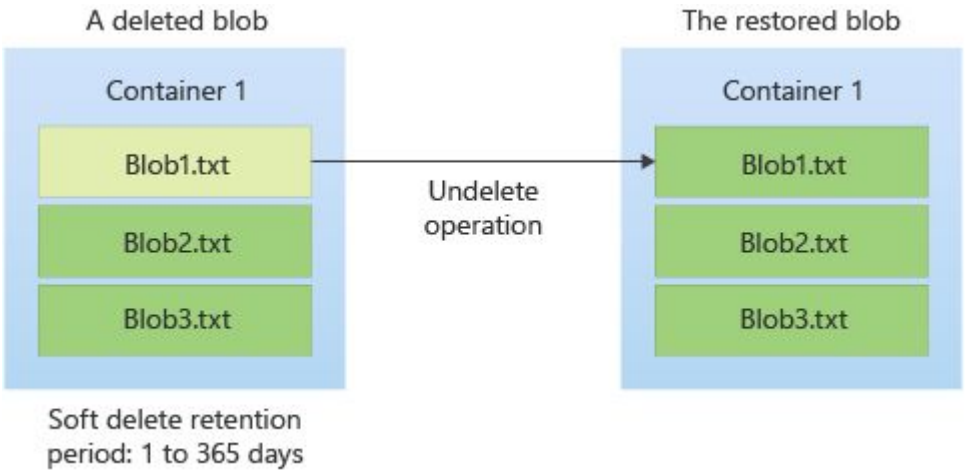
Versioning and Soft Delete

Versioning and Soft Delete

Both blob soft delete and blob versioning can help you protect from deletes and overwrites. These features can be used independently or together, depending on your workload, cost sensitivity, and recovery needs.

Blob storage customers storing critical data should enable soft delete and versioning for layered protection against unintended deletions and overwrites. Soft delete ensures your data remains recoverable for a configurable number of days.

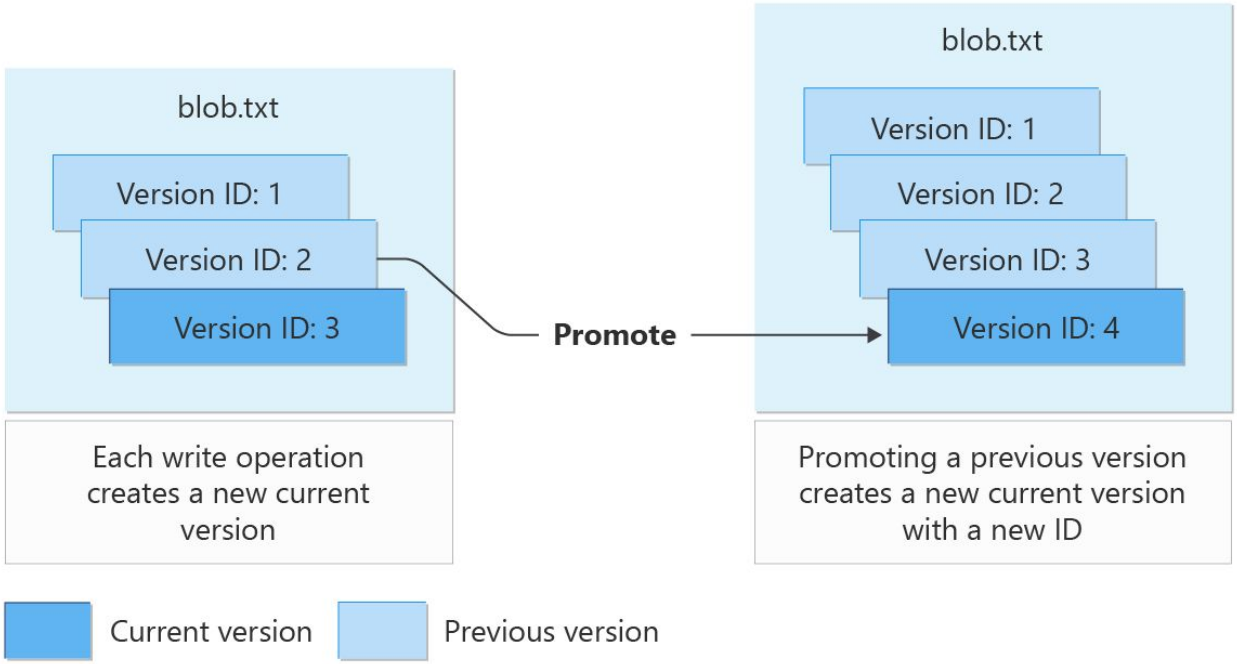
When to use soft delete



Remember

- A soft-deleted blob is created for every delete operation. Each soft-deleted blob is billed at the full size of the blob at the time of the operation.
- Avoid exceeding 1,000 soft-deleted snapshots per blob during the retention period to prevent performance degradation during blob listing operations.
- Retention is limited to a maximum of 365 days.
- *If soft delete is enabled, there is no way to permanently delete until the soft delete retention expires. All deletes are "soft" and when the retention period expires, the soft-deleted blob is permanently deleted.*
- If soft delete is disabled, all deletes are permanent, but the existing soft-deleted data are retained until the retention period expires.
- *The contents of soft-deleted blobs are not accessible via Read APIs. To access the data, you must first undelete the blob.*

When to use versioning



Remember

- Each write operation (Put Blob, Put Block List, Set Blob Metadata, and Copy Blob) creates a new version of the blob.
- Versions are retained until explicitly deleted, offering long-term recovery options.
- Avoid exceeding 1,000 versions per blob to maintain optimal performance and prevent performance degradation during blob listing operations.
- You can delete specific versions at any time. Separate roles are required to delete current versions and previous versions. That separation can be helpful to avoid mistakes. The same identity can have both of these roles assigned. Learn more.
- You can configure a lifecycle management policy or Azure Storage Actions to control the lifecycle of your versions and define retention conditions.
- Versioning is not available for accounts with hierarchical namespace enabled.

When to use both

- You need comprehensive protection against both accidental deletions and overwrites.
- You operate in a regulated environment requiring layered data protection.
- You want to ensure recovery options even if versions are deleted.
- You want to create a grace period where, when previous versions are deleted, they are retained for some period of time (soft delete retention)

When to use neither

- Your application has its own backup and recovery mechanisms.
- You have strict cost constraints and low risk of accidental data loss.
- Your data is temporary or test data and protection is not necessary.

Cost association

- Enabling soft delete or versioning for frequently overwritten data might result in increased storage capacity charges and increased latency when listing blobs.

Project

Version and Soft Delete Blob