Azure Storage - Detailed Reference

Azure Storage — Detailed Reference (Topics 29-33)

Topic 29: Azure Storage Accounts

The foundational storage service. All Azure Storage types live inside a Storage Account. You cannot use Blob, Files, Queues, or Tables without first creating a Storage Account.

Storage Account Types

Type	What it Supports	When to Use
General-purpose v2	Blob, File, Queue, Table, Data Lake Storage Gen2	Default choice — covers everything. Always use this for new deployments.
General-purpose v1	Blob, File, Queue, Table (no Data Lake, no access tiers)	Legacy. Do not use for new deployments. Cannot convert to v2 without migration.

Performance Tiers

Tier	What it Means	Backed By	Use Case
Standard	HDD-based, lower cost, higher latency	Magnetic drives (with SSD caching)	Most workloads, archival, non-critical data, bulk data
Premium	SSD-based, low latency, high IOPS	NVMe/SSD	High IOPS workloads, analytics, AVS datastores, high-performance databases

Premium Storage Accounts only support page blobs (for VM disks). You cannot use Premium for Blob, Files, Queues, or Tables. For high-performance file shares, use Azure Files Premium (separate from the account tier).

Replication Options — Critical Decision

This determines how resilient your data is. Choose carefully — changing replication after deployment may require migration.

Option	Full Name	What it Does	Cost	RPO	When to Use
LRS	Locally Redundant Storage	3 copies in a single data center. If the DC is destroyed, data is lost.	Lowest	0 (synchronous)	Non-critical data, temporary processing, data you can regenerate
ZRS	Zone-Redundant Storage	3 copies across 3 availability zones in	Medium	0 (synchronous)	Production workloads that need

Option	Full Name	What it Does	Cost	RPO	When to Use
		one region. Survives a zone failure.			HA within a region. No data loss if a zone goes down.
GRS	Geo-Redundant Storage	LRS in primary + async copy to paired region. Survives region failure.	Higher	~15 min (asynchronous)	DR requirement. Data survives region outage. Read from secondary requires manual failover.
GZRS	Geo-Zone-Redundant Storage	ZRS in primary + async copy to paired region. Best of both.	Highest	~15 min (asynchronous)	Production with both zone HA and region DR. Best resilience.
RA-GRS	Read-Access GRS	Same as GRS + read access to secondary region at all times.	Higher + read egress	~15 min	Need read access from secondary without failover. Read-only queries from DR region.
RA-GZRS	Read-Access GZRS	Same as GZRS + read access to secondary.	Highest + read egress	~15 min	Best resilience + always-readable secondary.

Key points about replication:

- LRS and ZRS writes are synchronous — data is written to all copies before acknowledging success.
- GRS/GZRS secondary copy is asynchronous — there is a replication lag (typically under 15 minutes, but not guaranteed).
- RA-GRS/RA-GZRS: the secondary region is always readable (read-only). Applications can fall back to reading from secondary if primary is unavailable.
- Without RA- prefix: you must initiate a manual failover to read from secondary. This changes the primary region.
- Failover with GRS/GZRS: Microsoft can initiate a forced failover in catastrophic region failure. You can also initiate a customer-managed failover (converts secondary to primary).

Always use ZRS or GZRS for production. LRS means your data dies with the data center. The cost difference is small relative to the risk.

Access Tiers (Blob only)

Azure Blob Storage supports four access tiers. You choose a default tier for the storage account, and can override per-blob.

Tier	Storage Cost	Retrieval Cost	Min Storage Duration	Access Frequency	Use Case
Hot	Highest	Lowest	None	Frequent (multiple times per day)	Active website content, frequently accessed data, short-lived data
Cool	Lower	Higher	30 days	Infrequent (once per month)	Backup data, older log files, datasets

Tier	Storage Cost	Retrieval Cost	Min Storage Duration	Access Frequency	Use Case
					accessed occasionally
Cold	Very low	Very high	90 days	Rare (once per quarter)	Compliance archives, rarely accessed backups
Archive	Lowest	Highest	180 days	Very rare (once per year or less)	Long-term retention, regulatory compliance, legal holds

Important details about access tiers:

- Retrieval cost: Hot has no per-GB retrieval charge. Cool/Cold/Archive charge per GB read.
- Early deletion penalty: If you delete a blob before the minimum storage duration, you pay the early deletion fee (prorated for the remaining days).
- Archive tier requires rehydration before reading: you submit a rehydration request, wait 1-15 hours (standard priority) or <1 hour (high priority, costs more), then read the blob.
- Rehydration options: Copy to Hot/Cool tier (preserves archive copy), or set blob tier to Hot/Cool (moves the blob).
- You cannot set Archive as the default account tier. You must explicitly set individual blobs to Archive.

Lifecycle Management

Automatically move blobs between tiers based on rules you define.

How it works: 1. Create a lifecycle management policy (JSON definition) 2. Define rules: conditions (days since last modification, blob type, prefix match) + actions (tier to Cool/Cold/Archive, delete blob, delete blob snapshot) 3. Azure evaluates the policy once per day 4. Matching blobs are moved/deleted automatically

Example policy: - Rule 1: If blob in container “logs” is > 30 days old → move to Cool - Rule 2: If blob in container “logs” is > 90 days old → move to Archive - Rule 3: If blob in container “temp” is > 7 days old → delete - Rule 4: If blob in container “backups” is > 365 days old → move to Archive - Rule 5: If blob snapshot is > 30 days old → delete snapshot

Cost: Free to configure. You just pay the storage cost of the tier the blob is in. Lifecycle management can save significant money by moving old data to cheaper tiers.

Limitations: - Policy applies at storage account level - Can filter by container name prefix or blob index tags - Cannot apply to Premium storage accounts - Minimum 24 hours between policy evaluation (not real-time)

Access Keys

Every storage account has 2 access keys (key1 and key2).

Key properties: - Full access to everything in the storage account (read, write, delete all data) - 512-bit key, auto-generated by Azure - You can regenerate either key at any time - Both keys work simultaneously

Key rotation process (no downtime): 1. Update all applications to use key2 2. Regenerate key1 in Azure Portal 3. Update any remaining apps to use the new key1 4. Regenerate key2 5. All apps now use rotated keys

Never use access keys in applications. They provide unlimited access with no granularity. Instead use: - SAS tokens (limited scope, time-bound) - Entra ID authentication (recommended — RBAC-based) - Managed Identities (for Azure resources accessing storage)

Shared Access Signatures (SAS)

Grant limited, time-bound access to storage resources without sharing your account key.

SAS Types:

Type	Scope	How Signed	Security Level
Account SAS	All services in the account	Account key	Broad access, use carefully
Service SAS	One service (blob, file, queue, table)	Account key	Scoped to one service
User Delegation SAS	One service	Entra ID credentials (not account key)	Most secure — no account key exposure

User Delegation SAS is the most secure because it is signed with your Entra ID credentials, not the storage account key. If the SAS is compromised, you can revoke it by disabling the Entra ID identity that created it. With account-key-signed SAS, you must regenerate the account key to revoke.

SAS Parameters:

Parameter	What it Controls	Example
Start time	When the SAS becomes valid	2026-04-23T00:00:00Z
Expiry time	When the SAS expires	2026-04-24T00:00:00Z
Permissions		rwdl

Parameter	What it Controls	Example
	Read (r), Write (w), Delete (d), List (l), Add (a), Create (c)	
IP range	Restrict to specific IPs	203.45.67.89/32
Protocol	HTTPS only or HTTP+HTTPS	https
Resource types	Service, container, or object	Object (sco)
Blob version / snapshot	Target a specific version	2026-04-20v1

Best practices for SAS: - Always use HTTPS - Set the shortest practical expiry time - Use User Delegation SAS when possible - Use stored access policies for service SAS (allows you to revoke by modifying the policy) - Never store SAS tokens in code — use Key Vault

---

Topic 30: Azure Blob Storage

Object storage for unstructured data. The most commonly used Azure Storage type. Stores files, images, videos, backups, logs, archives, and any binary data.

Blob Types

Type	What it is For	Max Size	How Data is Stored	When to Use
Block Blob	Text/binary files, images, videos, backups	190.7 TB (50,000 blocks x 4GB each)	Data split into blocks, uploaded in parallel, committed together	Most common blob type. 99% of use cases.
Append Blob	Log files, audit trails, append-only data	195 GB	Data can only be appended to the end. No modification of existing blocks.	Application logging, telemetry, audit trails where data must be append-only for integrity.
Page Blob	VHD disks (managed disks use this internally)	8 TB	512-byte pages. Random read/write access.	Virtual machine OS disks and data disks. Rarely used directly.

Block Blob upload process: 1. Split file into blocks (up to 4GB per block, up to 50,000 blocks) 2. Upload blocks in parallel (faster for large files) 3. Commit block list — blob becomes visible 4. If any block fails, retry only that block 5. Uncommitted blocks are garbage collected after 7 days

Blob Access Levels

Level	What it Means
Private (default)	No anonymous access. All requests must be authenticated.
Blob	Anonymous read access to blobs only. Users cannot list the container. They must know the exact blob URL.
Container	Anonymous read access to blobs AND container listing. Users can browse and download.

Never use anonymous access for sensitive data. For public content (website images, downloads), use Azure CDN in front of private storage instead of making the container public.

Blob Soft Delete

Protects against accidental deletion and ransomware.

How it works: - When soft delete is enabled and a blob is deleted, it is moved to a soft-deleted state instead of being permanently removed - Soft-deleted blobs are invisible to normal listing operations but can be listed with the “include deleted” parameter - You can restore a soft-deleted blob to its state before deletion - Retention period: 1-365 days (you choose) - During retention, you pay for the soft-deleted data at the same tier

Key points: - Soft delete applies to blob versions and snapshots too - If a blob is overwritten (not deleted), soft delete captures the previous version if versioning is also enabled - Soft delete is a per-storage-account setting — enable it on every storage account - Even if an attacker deletes the blob AND the soft-deleted version, immutability policies (WORM) prevent deletion

Blob Versioning

Every write operation creates a new version automatically.

How it works: - When versioning is enabled, every operation that modifies the blob creates a new version - The current version is the latest - Previous versions are retained and can be restored - Each version has its own tier and can be managed independently

Versioning vs Soft Delete:

Feature	Soft Delete	Versioning
Triggers	Delete or overwrite	Every write operation
Granularity	Recovers deleted/overwritten blobs	Recovers any previous state
Retention	1-365 days	Indefinite (until manually deleted)
Cost	Only soft-deleted data	Every version stored separately
Use case	Accident/ransomware protection	Complete change history

Recommendation: Enable both soft delete AND versioning on production storage accounts. Soft delete for short-term protection, versioning for complete history.

Blob Immutable Storage (WORM)

Write Once, Read Many. Data that cannot be modified or deleted for a specified period.

Two types:

Type	How it Works	When to Use
Time-based retention	Set a retention period (1 day to 146,000 days / 400 years). During retention, blobs cannot be deleted or overwritten. After	Regulatory compliance — SEC 17a-4, FINRA, CFTC.

Type	How it Works	When to Use
	retention expires, normal operations resume.	
Legal hold	Set a legal hold tag. Blobs cannot be deleted or modified until the hold is removed. No predefined duration.	Active litigation, investigation. Hold is removed when the legal matter is resolved.

Key properties: - Once a time-based retention policy is locked, it cannot be shortened (only extended). This prevents admins from circumventing the policy. - Legal holds can be added and removed freely (by authorized users). - Immutability applies to all blobs in the container (set at container level). - Overwrites are blocked. Appends to append blobs are allowed (only new data can be added). - Container deletion is blocked while immutable policies are active.

Compliance certifications: Immutable Blob Storage meets SEC 17a-4(f), CFTC 1.31(c)-(d), and FINRA requirements.

Blob Index Tags

Key-value tags on individual blobs. For categorization and searching.

How it works: - Add tags to a blob: { “project”: “alpha”, “status”: “processed”, “department”: “finance” } - Filter blobs by tags using the Find Blobs by Tags API - Example: find all blobs where project=alpha AND status=processed - Tags are indexed automatically — no scanning required - Up to 10 tags per blob, each key up to 256 chars, value up to 256 chars

Use cases: - Organize blobs without relying on container/folder structure - Search across containers using tag filters - Lifecycle management: use tags as conditions in lifecycle policies - Data classification: tag blobs with sensitivity level

---

Topic 31: Azure Files

Fully managed file shares accessible via SMB (Server Message Block) and NFS (Network File System). Like a network file share in the cloud.

Why Azure Files instead of a file server VM

- No VM to manage, patch, or maintain
- Built-in HA (data is replicated within the region)
- Accessible from anywhere via SMB/NFS
- Pay only for what you use (no idle VM costs)
- Integrates with Azure Backup natively

Use Cases

Use Case	Description
Lift-and-shift	

Use Case	Description
	Applications that expect SMB file shares. Point them to Azure Files instead of on-prem share.
Shared config	Multiple VMs reading the same configuration files from a shared location.
User home directories	FSLogix profile containers for Azure Virtual Desktop.
Replace on-prem file server	Consolidate distributed file servers into Azure Files.
Dev/test shares	Shared storage for development teams.

Share Tiers

Tier	When to Use	Max Share Size	Throughput	Cost
Transaction optimized	Heavy read/write operations, frequent transactions	100 TB	Up to 300 MB/s	Lower storage cost, higher per-transaction cost
Hot	General purpose, active workloads	100 TB	Up to 300 MB/s	Balanced cost
Cool	Infrequent access, cost optimization	100 TB	Up to 300 MB/s	Lower storage cost, higher transaction cost
Premium	Low latency, high IOPS (SSD-based)	102 TB	Up to 300+ MB/s	Highest cost, best performance

Premium tier details: - Backed by SSDs - Provisioned capacity: you pay for the capacity you provision, not what you use - IOPS scale with share size (roughly 5,000 IOPS per 100 GB provisioned) - Required for FSLogix profiles in production AVD deployments - Supports NFS and SMB

SMB vs NFS

Protocol	When to Use	AD Authentication	Multi-Protocol
SMB	Windows environments, mixed environments	Yes (Entra ID + on-prem AD)	Can enable both SMB+NFS on Premium
NFS	Linux-only environments	No	Same as above

Multi-protocol shares (Premium only): You can enable both SMB and NFS on the same share. Linux VMs use NFS, Windows VMs use SMB, accessing the same data.

Azure File Sync

Hybrid file sync between on-prem Windows Server and Azure Files.

How it works: 1. Install Azure File Sync agent on your on-prem Windows Server 2. Create a Sync Group (pairs a server endpoint with a cloud endpoint/Azure File share) 3. Files sync between server and Azure Files automatically 4. Cloud tiering: frequently accessed files

stay on-prem, cold files are tiered to Azure only (become placeholders on-prem) 5. When a tiered file is accessed, it downloads automatically from Azure

Cloud tiering details: - You set the volume free space policy (e.g., “always keep 20% free on the local volume”) - When the volume fills up, the least-recently-used files are tiered to Azure - Tiered files appear as placeholders (pointers) on the local file system - Accessing a tiered file triggers an automatic download (slight delay on first access) - You can pin specific files/folders to always stay local

Why Azure File Sync: - Replace DFSR (Distributed File System Replication) — Azure File Sync is more reliable - Centralize file shares in Azure while keeping local access for performance - Multi-server sync: multiple on-prem servers sync to the same Azure File share - Backup: Azure Files is backed up natively — no need for on-prem backup of synced data

---

Topic 32: Azure Managed Disks

Block-level storage for VMs. Managed by Azure — you do not create or manage storage accounts for VM disks.

Why Managed Disks vs Unmanaged

Feature	Unmanaged Disks	Managed Disks
Storage account	You create and manage it	Azure creates and manages it
IOPS limit	20,000 IOPS per storage account	No per-account limit (per-disk limits apply)
Scalability	Limited by storage account	Virtually unlimited
Availability sets	You must place VM disks in different storage accounts	Azure handles placement automatically
ARM templates	Complex (must create storage accounts)	Simple (just reference disk SKU)
Cost optimization	Manual tiering	Built-in tiering and snapshots

Always use Managed Disks. Unmanaged disks are legacy and provide no benefits.

Disk Types — Complete Comparison

Disk Type	Max IOPS	Max Throughput	Max Size	Burst IOPS	Cost (1TB/month)	Use Case
Standard HDD	~500	~60 MB/s	32 TB	No	~$20	Backup, non-critical, infrequent access
Standard SSD	~6,000	~300 MB/s	32 TB	Up to 3,500	~$50	Web servers, dev/test, light databases
Premium SSD	~7,500	~250 MB/s	32 TB	Up to 30,000	~$125	Production workloads, databases

Disk Type	Max IOPS	Max Throughput	Max Size	Burst IOPS	Cost (1TB/month)	Use Case
Premium SSD v2	~80,000	~1,200 MB/s	64 TB	No	~$80+ (pay for what you provision)	High-performance databases, mission-critical
Ultra Disk	~160,000	~2,000 MB/s	64 TB	No	~$140+ (pay for IOPS+throughput+capacity)	Most demanding workloads, sub-millisecond latency

Premium SSD Performance by Size

Size	Base IOPS	Base Throughput	Burst IOPS	Burst Throughput
P1 (4 GB)	120	25 MB/s	350	170 MB/s
P6 (64 GB)	240	35 MB/s	1,200	170 MB/s
P10 (128 GB)	500	100 MB/s	3,500	170 MB/s
P20 (512 GB)	2,300	150 MB/s	3,500	170 MB/s
P30 (1,024 GB)	5,000	200 MB/s	30,000	1,000 MB/s
P40 (2,048 GB)	7,500	250 MB/s	30,000	1,000 MB/s
P50 (4,096 GB)	7,500	250 MB/s	30,000	1,000 MB/s

Bursting: Premium SSDs can burst to higher IOPS/throughput for short periods (up to 30 minutes at a time, accumulated when the disk is idle). Great for boot storms and occasional spikes.

Premium SSD v2 — Sub-line CRUD

Unlike other disk types, Premium SSD v2 lets you adjust performance independently of size:

- **Size:** 1 GB to 64 TB (fine-grained, not fixed SKUs)
- **IOPS:** 3,000 to 80,000 (adjustable every 5 minutes without detaching)
- **Throughput:** 125 MB/s to 1,200 MB/s (adjustable every 5 minutes)
- **Cost:** You pay for size + IOPS + throughput separately

Example: You need 2 TB of storage with 10,000 IOPS and 300 MB/s throughput. With Premium SSD, you would need a P30 (1 TB at 5,000 IOPS) or P40 (2 TB at 7,500 IOPS). With Premium SSD v2, you provision exactly 2 TB + 10,000 IOPS + 300 MB/s. No waste.

Limitations: - No host-based encryption (only SSE) - No shared disks - Not available in all regions - Cannot convert from other disk types to Premium SSD v2 (must create new)

Ultra Disk — Maximum Performance

- Adjustable IOPS (up to 160,000) and throughput (up to 2,000 MB/s) without detaching
- Sub-millisecond latency

- Storage-class ECC for data integrity
- Supports shared disks (for clustered databases)

Ultra Disk is the only choice for: - Oracle RAC clusters requiring >20,000 IOPS - SAP HANA requiring deterministic sub-millisecond latency - Large SQL Server deployments with extreme IOPS requirements

Cost model: You pay for capacity + IOPS + throughput separately. Each is billed hourly.

Disk Encryption

Type	How it Works	Key Management	When to Use
SSE (Storage Side Encryption)	Enabled by default. Azure encrypts data at rest using platform-managed keys (PMK). No action needed.	Microsoft manages keys	Default for all disks. No cost.
SSE + CMK	You provide your own key stored in Key Vault. Azure uses your key to encrypt.	You manage keys in Key Vault	Compliance requiring customer-managed keys. Full key lifecycle control.
Azure Disk Encryption (ADE)	BitLocker (Windows) or DM-Crypt (Linux) encryption inside the VM OS.	Key Vault stores BEK/KEK	Legacy approach. Use SSE+CMK instead.

SSE is always on — you cannot disable it. Every disk is encrypted at rest.

SSE + CMK process: 1. Create a Disk Encryption Set (DES) in Azure 2. Associate the DES with a Key Vault key 3. Create disks referencing the DES 4. All data on those disks is encrypted with your key 5. If you revoke the key, the disk becomes unreadable (use with caution)

ADE limitations: - Requires VM agent and extension - Performance overhead (encryption happens in the VM OS) - More complex key management - Being superseded by SSE+CMK

Recommendation: Use SSE+CMK for compliance. Do not use ADE for new deployments.

Shared Disks

Some Premium SSDs and Ultra Disks can be shared across multiple VMs simultaneously.

How it works: - Enable shared disk feature on the disk - Attach the disk to multiple VMs (up to 2 VMs for most SKUs, up to 5 for some) - All VMs can read and write to the disk - The application (not Azure) handles concurrent access coordination (e.g., clustering software)

Use cases: - SQL Server Failover Cluster Instance (FCI) - Oracle RAC (Real Application Clusters) - Shared file system for clustered applications

Requirements: - All VMs must be in the same Availability Set or Proximity Placement Group - Only supported on Premium SSD and Ultra Disk - Application must support shared disk access (e.g., Windows Server Failover Clustering)

Topic 33: Azure Data Lake Storage

Built on Blob Storage with a hierarchical namespace. Designed for analytics and big data workloads.

Why Data Lake vs Regular Blob Storage

Feature	Blob Storage (flat namespace)	Data Lake Storage Gen2 (hierarchical namespace)
Namespace	Flat — "folder/file.csv" is just a name with slashes	True directories — folders exist as actual objects
Rename/ Move	Copy + delete (slow, expensive for large data)	Atomic rename/move (instant, free)
Permissions	Container-level or blob-level	POSIX-style ACLs (user, group, other — read/write/ execute)
Performance	Good for object access	Optimized for analytics (Spark, Databricks, Synapse)
Access	REST API, SDKs	REST API, SDKs, AND Blob API (compatible)
Best for	General purpose, static content, backups	Big data analytics, data lakes, ETL pipelines

True directories matter for analytics: - In flat Blob, listing "folder/ file.csv" requires scanning all blobs with that prefix — O(N) operation - In Data Lake, listing a directory is O(1) — instant, regardless of total objects - Moving a directory with 10,000 files in Blob = 10,000 copy+delete operations - Moving a directory in Data Lake = 1 rename operation

How to Enable Data Lake Storage Gen2

1. Create a General-purpose v2 storage account
2. Enable **Hierarchical namespace** during creation
3. This cannot be added later — you must create a new account

Important: Once hierarchical namespace is enabled: - You can still use Blob Storage APIs to access data - Some Blob features are not available (append blobs, archive tier on some configurations) - Azure Blob Fuse and Data Lake Storage Gen2 REST API are the primary access methods

POSIX ACLs (Access Control Lists)

Data Lake Storage Gen2 supports POSIX-style permissions:

Permission	What it Allows	Applies To
Read (r)	Read file contents, list directory contents	Files and directories
Write (w)	Write/modify file, create files in directory	Files and directories
Execute (x)	Execute file (for scripts), traverse directory	Files and directories

ACL types: - **Owner:** The user who created the object. Can change permissions. - **Owning group:** POSIX group associated with the object. - **Named users:** Specific Entra ID users with custom permissions. - **Named groups:** Specific Entra ID groups with custom permissions. - **Mask:** Limits the maximum permissions for named users and groups. - **Other:** All other users (like “world” permissions in Unix).

ACLs vs RBAC: - ACLs control data plane access (reading/writing files and directories) - RBAC controls management plane access (creating storage accounts, containers) - Use both together for defense in depth

Data Lake Use Cases

Use Case	How
Analytics	Data Lake is the primary storage for Azure Synapse, Databricks, and HDInsight
Data ingestion	Raw data lands in Data Lake (bronze layer)
ETL/ELT	Transform data in Data Lake (silver layer)
Serving	Curated data in Data Lake (gold layer) for BI tools
Machine Learning	Training data stored in Data Lake, accessed by Azure ML

Medallion architecture (bronze/silver/gold): - Bronze: Raw, unprocessed data as-is from source systems - Silver: Cleaned, filtered, transformed data - Gold: Aggregated, business-ready data for analytics and reporting