Level: Advanced

Microsoft Azure Exam AZ-104 Certification

← Back to the **Course**

Azure Files and Azure Blob Storage - **Practice Mode**

Completed on Mon, 13 Oct 2025

1st	5/5	100.00%	PASS
Attempt	Marks Obtained	Your Score	Result

Download Report

Domain wise Quiz Performance Report

No.	Domain	Total Question	Correct	Incorrect	Unattempted	Marked for Review
1	Implement and manage storage	5	5	0	0	0
Total	All Domains	5	5	0	0	0

Review the Answers

Filter By

Question 1

Correct

Domain: Implement and manage storage

View Case Study

You are tasked with creating a new Blob container in the tailwindstorage1 storage account to securely store incoming application logs. The container must block public access and only accept authenticated requests from managed identities. What should you do to meet these requirements? What do you need to do to meet these needs?

A. Enable public read access for the container and assign the Reader role to managed identities

B. Set the container's access level to Private and assign the Storage Blob Data Contributor role to managed identities right

C. Configure shared access signatures (SAS) for the container and allow public access

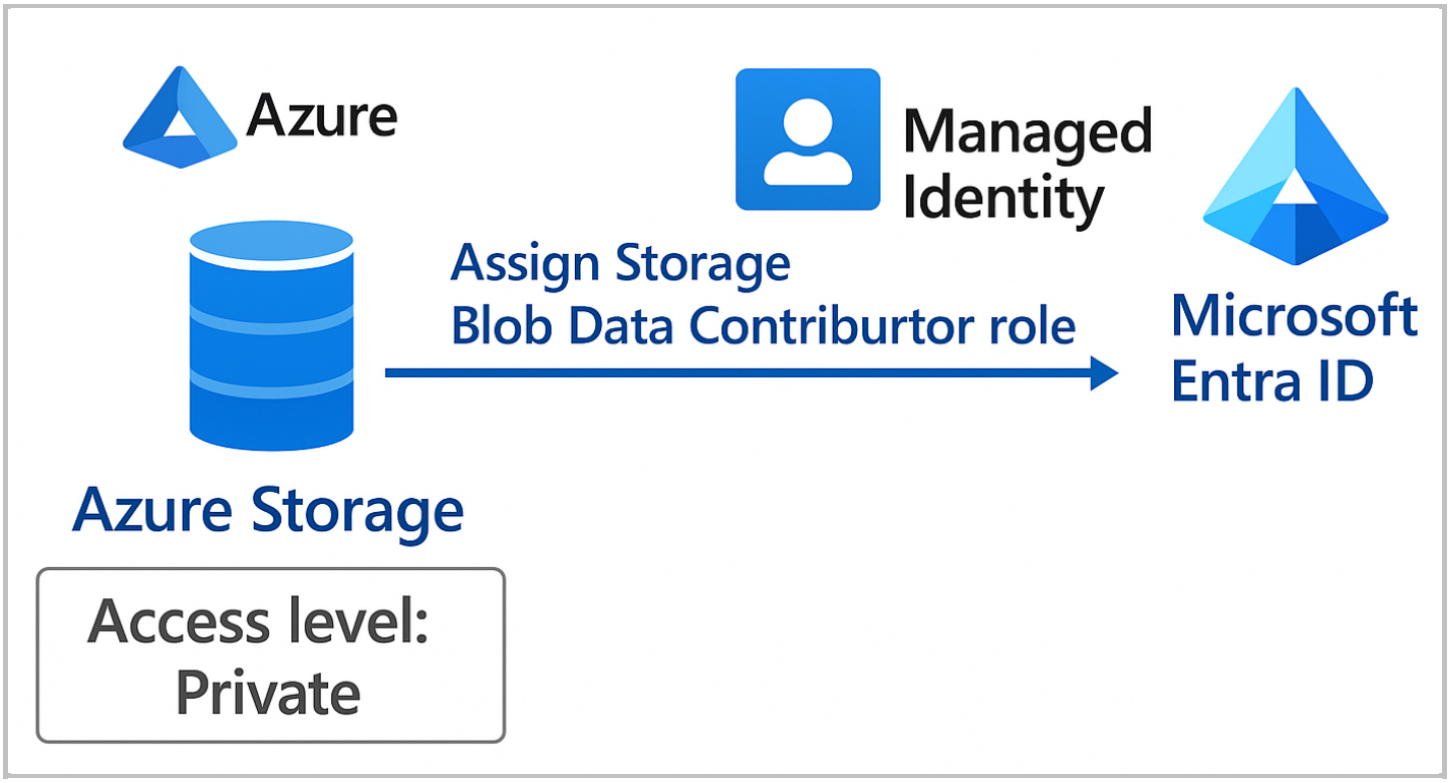
D. Set the container's access level to Blob and disable encryption

---

Explanation:

Correct Answer: B

Option B: Set the container's access level to Private and assign the Storage Blob Data Contributor role to managed identities is correct because setting the container's access level to Private ensures that public access is completely blocked, meeting the first requirement. By default, a Private access level allows only authenticated requests to access the container. Assigning the Storage Blob Data Contributor role to managed identities enables them to perform both read and write operations on blobs within the container. This role is specifically designed for scenarios where managed identities require full access to blob data. Furthermore, Microsoft Entra ID integration ensures that requests are authenticated securely, and the use of managed identities eliminates the need for credentials to be stored in application code, enhancing security. This configuration meets all the requirements for secure storage and access



Option A: Enable public read access for the container and assign the Reader role to managed identities is incorrect because enabling public read access contradicts the requirement to block public access to the container. Public read access allows anyone with the container URL to view its content without authentication, which does not meet the security requirement. Additionally, the Reader role only provides read-only access to data and does not allow users to write or upload logs to the container. Managed identities need a role that grants both read and write permissions for this scenario.

Option C: Configure shared access signatures (SAS) for the container and allow public access is incorrect because while shared access signatures (SAS) can restrict access based on permissions and expiration, allowing public access would violate the requirement to block public access entirely. SAS tokens are useful for temporary, granular access to storage resources but do not inherently enforce managed identity-based authentication. This option does not align to enable secure, role-based access control using Microsoft Entra ID and managed identities.

Option D: Set the container's access level to Blob and disable encryption is incorrect because setting the container access level to Blob allows public read access to blobs within the container, which fails to block public access as required. Disabling encryption contradicts best practices for securing data in Azure Storage. Encryption at rest is automatically enabled by Azure Storage and cannot be disabled for compliance and data security reasons. This option is technically infeasible and does not satisfy any of the stated requirements.

References:

https://learn.microsoft.com/en-us/azure/storage/blobs/anonymous-read-access-configure?tabs=portal

https://learn.microsoft.com/en-us/azure/storage/blobs/assign-azure-role-data-access?tabs=portal

Ask our Experts

Did you like this **Question?**

Question 2 Correct

Domain: Implement and manage storage

View Case Study

The IT team has requested that you configure Azure Files in the tailwindstorage2 storage account to allow recovery of deleted files for 30 days. What should you do first to implement this requirement?

A. Create file share snapshots and store them in the Archive tier

B. Configure Azure Backup for the file shares and set the retention period to 30 days

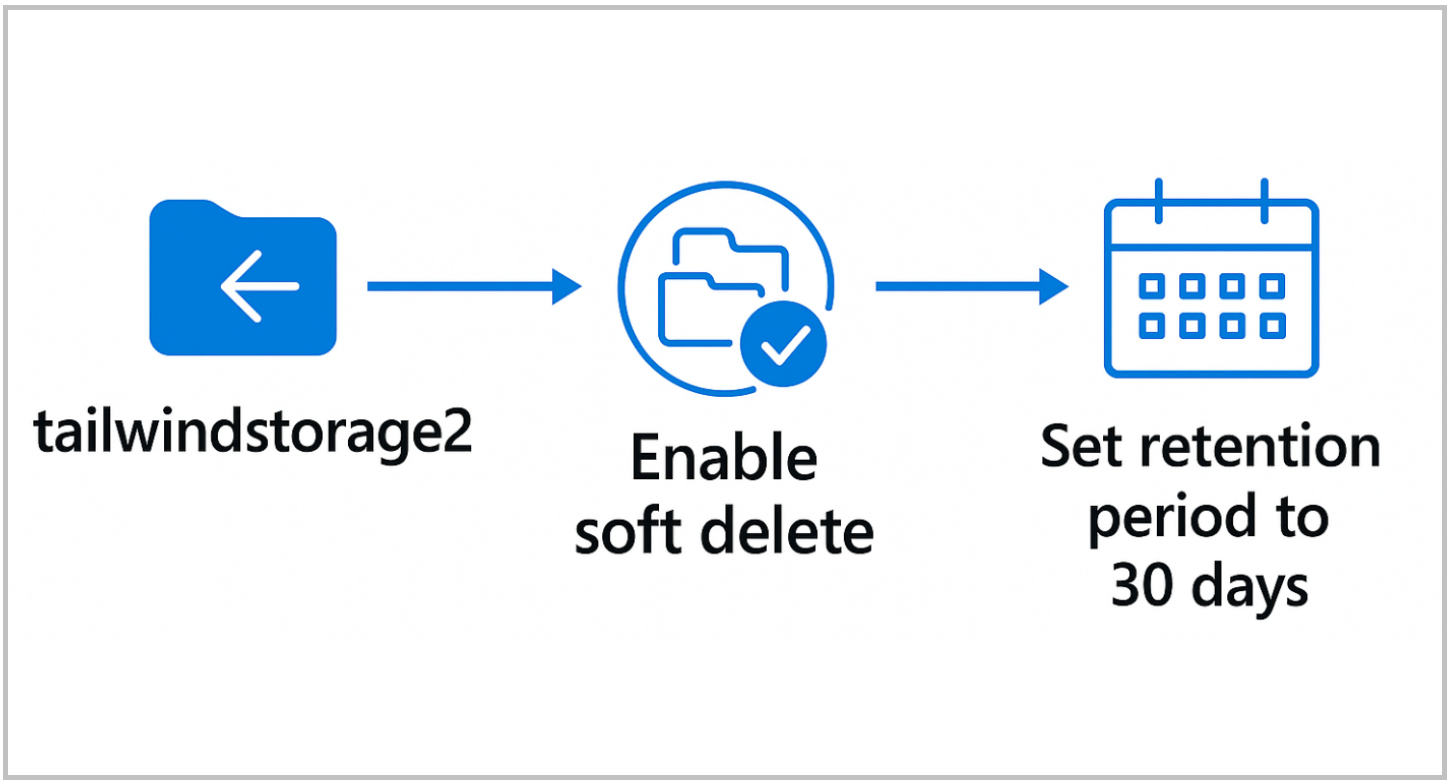
C. Enable versioning for the file shares

D. Enable soft delete for file shares in the storage account right

Explanation:

Correct Answer: D

Option D: Enable soft delete for file shares in the storage account is correct because enabling soft delete for Azure file shares allows you to recover files that are deleted or overwritten, as long as the soft delete retention period is configured. By enabling this feature in the tailwindstorage2 storage account and setting the retention period to 30 days, any deleted files in the Azure Files shares can be restored during that time frame. Soft delete ensures that the recovery process is seamless, without requiring additional backup infrastructure or manual intervention. This is a built-in feature in Azure Files and aligns directly with the requirement to recover deleted files for 30 days. Furthermore, soft delete protects against accidental deletions or malicious activities, offering an additional layer of resilience without introducing extra costs for infrastructure like Azure Backup.



Option A: Create file share snapshots and store them in the Archive tier is incorrect because while snapshots allow point-in-time recovery of file shares, they do not provide the ability to automatically recover deleted files without manual intervention. Additionally, snapshots are stored in the same tier as the file share and cannot be moved to the Archive tier, as Azure Files does not support tiering like Azure Blob Storage. This option introduces unnecessary complexity and does not fulfill the requirement to enable automatic recovery of deleted files within a 30-day retention period.

Option B:Configure Azure Backup for the file shares and set the retention period to 30 days is incorrect because while Azure Backup supports backing up file shares and provides a retention period, it is not the most efficient solution for recovering deleted files in this scenario. Azure Backup is designed for more comprehensive backup and restore operations, often involving longer-term retention and more complex recovery requirements. Configuring Azure Backup for this task would introduce additional costs and complexity, such as setting up backup policies and recovery points, which are unnecessary for a simple 30-day soft delete requirement. Furthermore, Azure Backup is not required for enabling or utilizing the soft delete feature.

Option C: Enable versioning for the file shares is incorrect because file versioning is not a feature available for Azure Files. While Azure Blob Storage supports versioning for tracking and restoring previous versions of blobs, Azure Files relies on snapshots and soft delete for data protection. Choosing this option demonstrates a misunderstanding of Azure Files capabilities, as enabling versioning is not applicable in this context.

Reference:

https://learn.microsoft.com/en-us/azure/storage/files/storage-files-enable-soft-delete?tabs=azure-portal

Ask our Experts

Did you like this **Question?**

Question 3 Correct

Domain: Implement and manage storage

View Case Study

You need to implement a lifecycle policy on the tailwindstorage3 storage account to transition logs from the Hot tier to the Cool tier after 45 days and then to the Archive tier after 90 days. What is the first step to achieve this?

A. Configure monitoring for blob storage tier changes in the Azure portal

B. Define a JSON policy with rules specifying age-based transitions and upload it to the storage account right

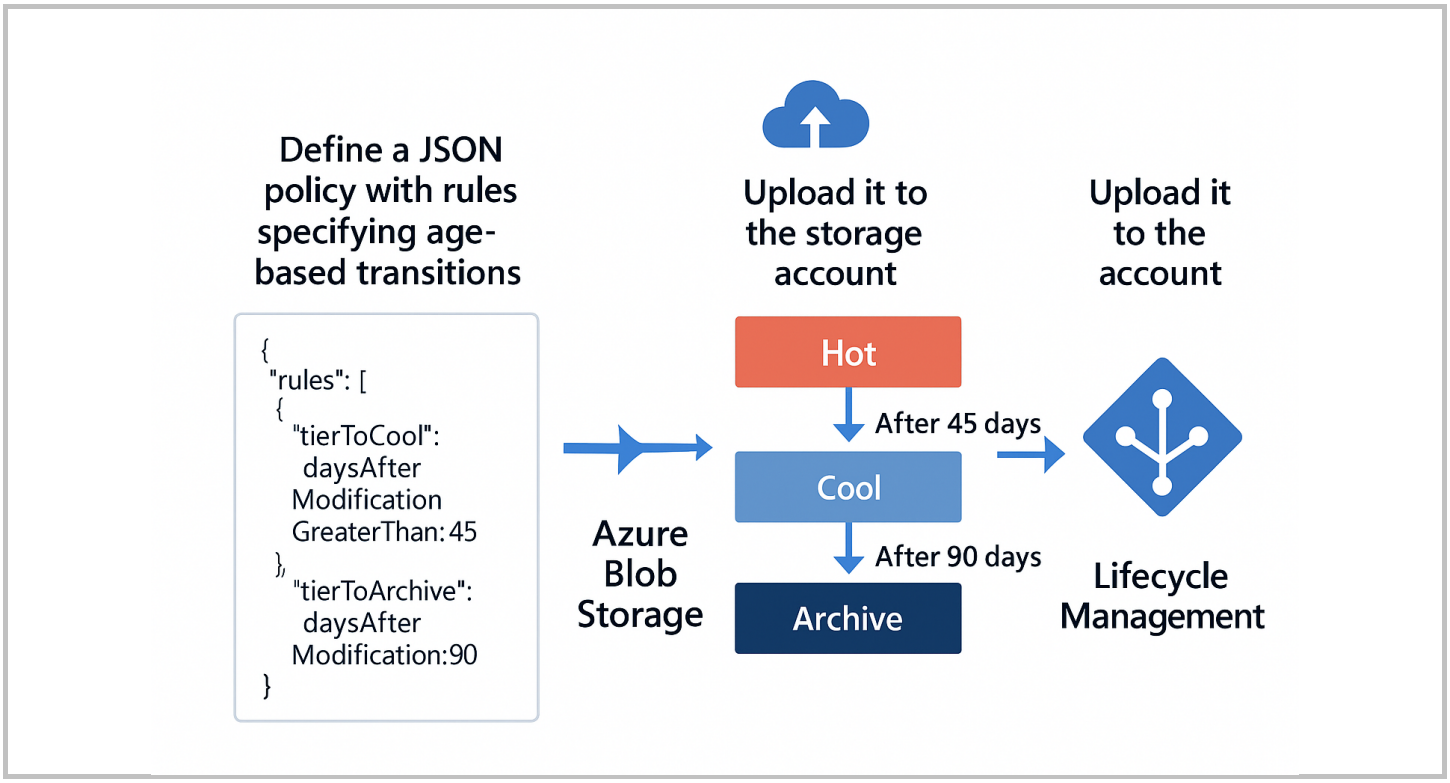
C. Enable blob versioning for all containers in the storage account

D. Modify the replication type of the storage account to LRS

Explanation:

Correct Answer: B

Option B: Define a JSON policy with rules specifying age-based transitions and upload it to the storage account is correct - This is the correct approach to implement automated lifecycle management in Azure Blob Storage. You must define a JSON-based lifecycle policy that specifies rules to transition blobs from the Hot tier to Cool after 45 days, and then to Archive after 90 days. This policy is then uploaded via the Azure portal under the Lifecycle Management section of the storage account. JSON policies give fine-grained control using filters (e.g., blob prefix, type) and remove the need for manual intervention. This method aligns with cost optimization and Azure's built-in automation features for storage tiering.



Option A: Configure monitoring for blob storage tier changes in the Azure portal is incorrect - Monitoring is useful for tracking changes and alerts, but it does not help implement lifecycle policies. It provides insights, not control mechanisms. Lifecycle rules are not configured through monitoring but through explicit policy definitions.

Option C: Enable blob versioning for all containers in the storage account is incorrect - Blob versioning is used for data protection — it allows recovery of previous blob versions in case of accidental overwrite or deletion. It does not support or affect tier transitions or lifecycle automation, which is the core requirement here.

Option D: Modify the replication type of the storage account to LRS is incorrect - LRS (Locally Redundant Storage) defines how many and where copies of the data are kept, for durability purposes. It has no role in lifecycle management or in transitioning blob tiers based on age.

Reference:

https://learn.microsoft.com/en-us/azure/storage/blobs/lifecycle-management-overview

Ask our Experts

Did you like this **Question?**

Question 4

Correct

Domain: Implement and manage storage

View Case Study

The IT team wants to store images in tailwindstorage1 that are accessed frequently for 30 days and rarely thereafter. The images must remain available with minimal latency even after the first 30 days. Which storage tier configuration should you use?

A. Keep the images permanently in the Hot tier to ensure low-latency access

B. Store the images directly in the Archive tier to minimize costs

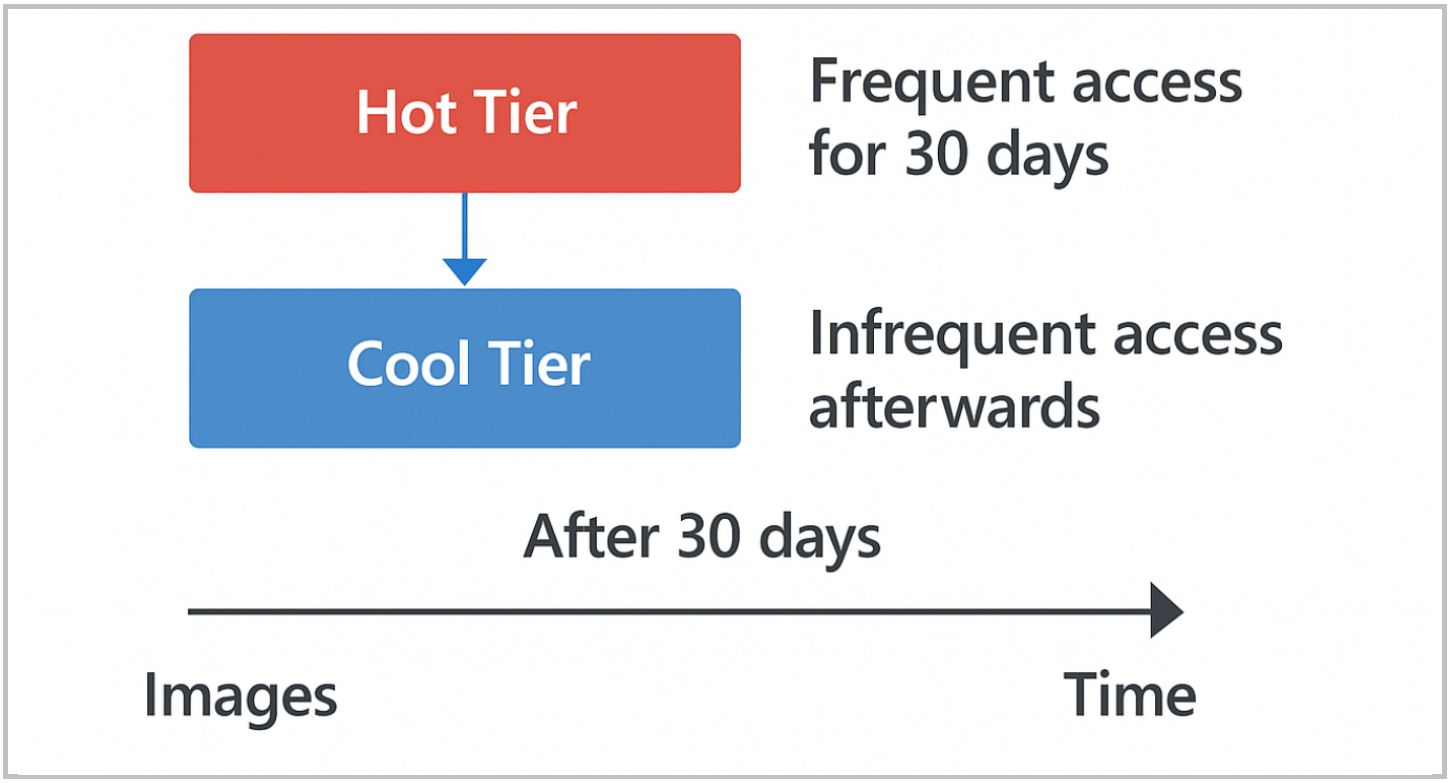
C. Use the Cool tier for the first 30 days and transition to the Hot tier afterward

D. Use the Hot tier for the first 30 days and transition to the Cool tier afterward right

Explanation:

Correct Answer: D

Option D: Use the Hot tier for the first 30 days and transition to the Cool tier afterward is correct because the Hot tier in Azure Blob Storage is designed for data that needs frequent access, making it suitable for the first 30 days when image access is high. After this period, the Cool tier becomes a cost-efficient choice for less frequently accessed data, as it still supports low-latency access. This strategy ensures a balance between cost and performance. The Hot tier guarantees fast access during the initial period of frequent use, while the Cool tier reduces costs for infrequent use without compromising performance. Automating the transition using Azure Blob Storage lifecycle policies simplifies cost management while maintaining data accessibility. This approach aligns with the IT team's goals of frequent initial access and lower costs for rare access later.



Option A: Keep the images permanently in the Hot tier to ensure low-latency access is incorrect because while the Hot tier offers low latency and optimal performance, it is the most expensive tier. Keeping the images permanently in the Hot tier is not cost-efficient, especially after the first 30 days when access frequency decreases. The IT team's requirement specifies rare access after the initial 30 days, making the Cool tier a better choice for the latter period. By not transitioning to a lower-cost tier, this approach disregards the cost-saving potential while still meeting the performance requirements.

Option B: Store the images directly in the Archive tier to minimize costs is incorrect because the Archive tier is the most cost-effective storage option but is unsuitable for data that requires frequent or low-latency access. Data in the Archive tier is offline and must be rehydrated before access, leading to significant delays and operational overhead. This makes it impractical for scenarios where images need to be accessed frequently during the first 30 days. The Archive tier does not meet the requirement of ensuring availability with minimal latency, even for rarely accessed images.

Option C:Use the Cool tier for the first 30 days and transition to the Hot tier afterward is incorrect because the Cool tier is designed for data that is infrequently accessed, and using it during the first 30 days when images are accessed frequently is not optimal. This would result in unnecessary latency and potential performance issues during the period of high access demand. Transitioning to the Hot tier afterward also increases costs without any justification, as the access frequency decreases after the first 30 days. This configuration fails to align with the cost and performance requirements described in the scenario.

Reference:

https://learn.microsoft.com/en-us/azure/storage/blobs/access-tiers-overview

Ask our Experts

Did you like this **Question?**

Question 5 Correct

Domain: Implement and manage storage

View Case Study

A developer accidentally deleted a critical blob in the tailwindstorage3 storage account where blob versioning is enabled. Which action should you take to recover the deleted blob?

A. Enable soft delete for blobs and restore the blob from the deleted items

B. Use the List Blob Versions API to identify and copy the required blob version right

C. Restore the entire container from a backup snapshot

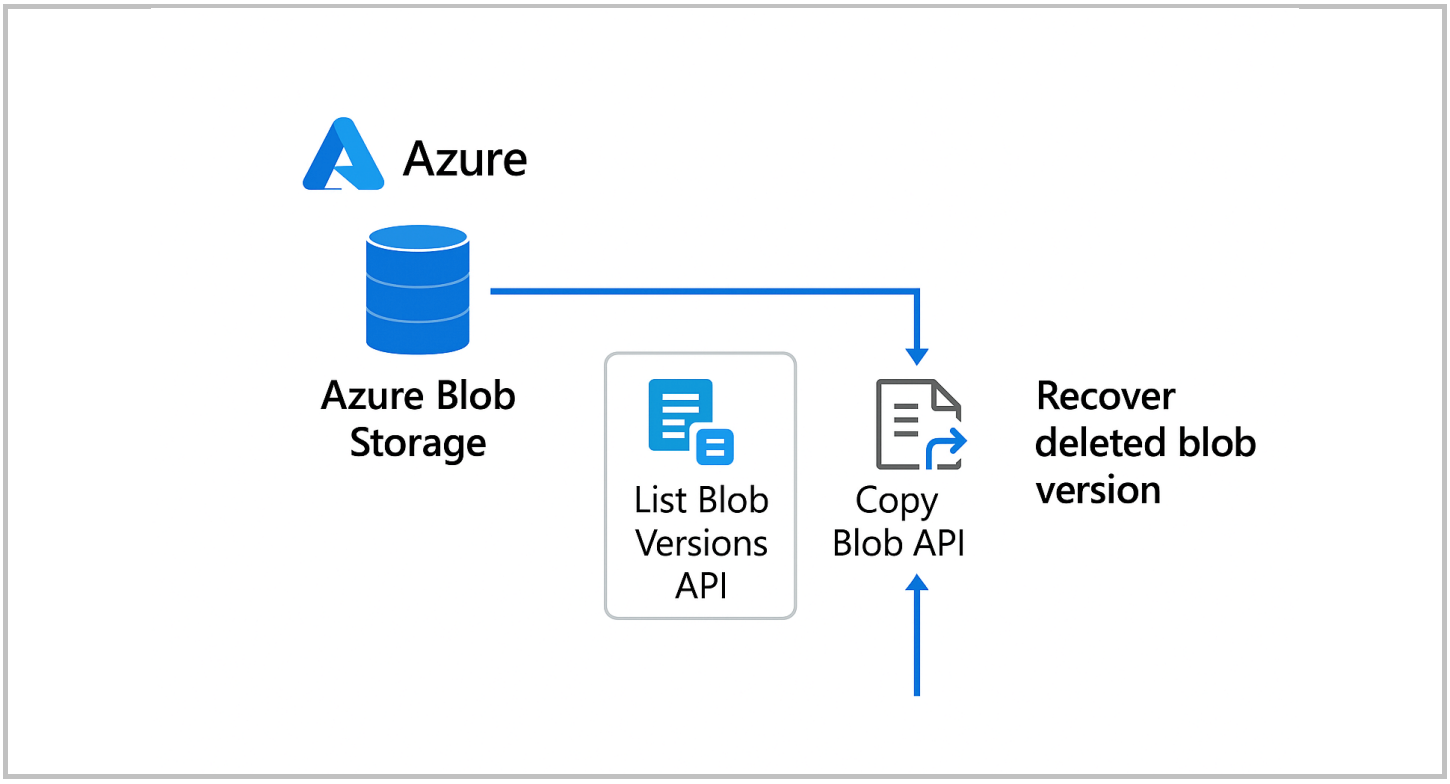
D. Configure blob lifecycle management to retrieve the archived version of the blob

---

Explanation:

Correct Answer: B

Option B: Use the List Blob Versions API to identify and copy the required blob version is correct because since blob versioning is already enabled, Azure automatically preserves earlier versions of blobs, even if they are deleted. By using the List Blob Versions API, you can locate the previous version of the deleted blob. Then, you can recover it using the Copy Blob operation to either the same or a new blob path. This method allows for precise, non-disruptive recovery without restoring the entire container or relying on other mechanisms like snapshots or lifecycle policies.



Option A: Enable soft delete for blobs and restore the blob from the deleted items is incorrect - while soft delete is a recovery

mechanism, it must be enabled before the deletion occurs. Enabling it afterward does not help. Additionally, since versioning is already enabled, recovery should be performed using versioning features, not soft delete.

Option C: Restore the entire container from a backup snapshot is incorrect - This is a broad and disruptive action. There's no indication a snapshot exists, and recovering a full container for one deleted blob is excessive. Blob versioning offers granular recovery — the appropriate method in this case.

Option D: Configure blob lifecycle management to retrieve the archived version of the blob is incorrect - Lifecycle management is used for tier transitions or deletion automation, not recovery. It does not retrieve deleted blobs or versions and operates independently of versioning. It's irrelevant to this recovery scenario.

Reference:

https://learn.microsoft.com/en-us/azure/storage/blobs/versioning-overview

Ask our Experts

Did you like this **Question?**

Finish Review