100 XP

Introduction

3 minutes

Data sets have unique lifecycles. Early in the lifecycle, people access some data often. But the need for access drops drastically as the data ages. Some data stays idle in the cloud and is rarely accessed once stored.

After completing this module, you'll be able to:

- Describe how each of the access tiers are optimized.
- Create and implement a lifecycle policy.
- Rehydrate blob data stored in an archive tier.

Next unit: Explore the Azure Blob storage lifecycle

Continue

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Explore the Azure Blob storage lifecycle

3 minutes

Data sets have unique lifecycles. Early in the lifecycle, people access some data often. But the need for access drops drastically as the data ages. Some data stays idle in the cloud and is rarely accessed once stored. Some data expires days or months after creation, while other data sets are actively read and modified throughout their lifetimes.

Access tiers

Azure storage offers different access tiers, allowing you to store blob object data in the most cost-effective manner. Available access tiers include:

- **Hot -** Optimized for storing data that is accessed frequently.
- **Cool** - Optimized for storing data that is infrequently accessed and stored for at least 30 days.
- **Archive -** Optimized for storing data that is rarely accessed and stored for at least 180 days with flexible latency requirements, on the order of hours.

The following considerations apply to the different access tiers:

- The access tier can be set on a blob during or after upload.
- Only the hot and cool access tiers can be set at the account level. The archive access tier can only be set at the blob level.
- Data in the cool access tier has slightly lower availability, but still has high durability, retrieval latency, and throughput characteristics similar to hot data.
- Data in the archive access tier is stored offline. The archive tier offers the lowest storage costs but also the highest access costs and latency.
- The hot and cool tiers support all redundancy options. The archive tier supports only LRS, GRS, and RA-GRS.
- Data storage limits are set at the account level and not per access tier. You can choose to use all of your limit in one tier or across all three tiers.

Manage the data lifecycle

Azure Blob storage lifecycle management offers a rich, rule-based policy for General Purpose v2 and Blob storage accounts. Use the policy to transition your data to the appropriate access tiers or expire at the end of the data's lifecycle. The lifecycle management policy lets you:

- Transition blobs to a cooler storage tier (hot to cool, hot to archive, or cool to archive) to optimize for performance and cost
- Delete blobs at the end of their lifecycles
- Define rules to be run once per day at the storage account level
- Apply rules to containers or a subset of blobs (using prefixes as filters)

Consider a scenario where data gets frequent access during the early stages of the lifecycle, but only occasionally after two weeks. Beyond the first month, the data set is rarely accessed. In this scenario, hot storage is best during the early stages. Cool storage is most appropriate for occasional access. Archive storage is the best tier option after the data ages over a month. By adjusting storage tiers in respect to the age of data, you can design the least expensive storage options for your needs. To achieve this transition, lifecycle management policy rules are available to move aging data to cooler tiers.

> **Note**
>
> Data stored in a premium block blob storage account cannot be tiered to Hot, Cool, or Archive using Set Blob Tier or using Azure Blob Storage lifecycle management. To move data, you must synchronously copy blobs from the block blob storage account to the Hot tier in a different account using the Put Block From URL API or a version of AzCopy that supports this API. The Put Block From URL API synchronously copies data on the server, meaning the call completes only once all the data is moved from the original server location to the destination location.

Next unit: Discover Blob storage lifecycle policies

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Discover Blob storage lifecycle policies

3 minutes

A lifecycle management policy is a collection of rules in a JSON document. Each rule definition within a policy includes a filter set and an action set. The filter set limits rule actions to a certain set of objects within a container or objects names. The action set applies the tier or delete actions to the filtered set of objects.:

JSON Copy

```
{
  "rules": [
    {
      "name": "rule1",
      "enabled": true,
      "type": "Lifecycle",
      "definition": {...}
    },
    {
      "name": "rule2",
      "type": "Lifecycle",
      "definition": {...}
    }
  ]
}
```

A policy is a collection of rules:

Parameter name	Parameter type	Notes
`rules`	An array of rule objects	At least one rule is required in a policy. You can define up to 100 rules in a policy.

Each rule within the policy has several parameters:

Parameter name	Parameter type	Notes	Required
`name`	String	A rule name can include up to 256 alphanumeric characters. Rule name is case-sensitive. It must be unique within a policy.	True
`enabled`	Boolean	An optional boolean to allow a rule to be temporary disabled. Default value is true if it's not set.	False
`type`	An enum value	The current valid type is Lifecycle.	True
`definition`	An object that defines the lifecycle rule	Each definition is made up of a filter set and an action set.	True

Rules

Each rule definition includes a filter set and an action set. The filter set limits rule actions to a certain set of objects within a container or objects names. The action set applies the tier or delete actions to the filtered set of objects.

The following sample rule filters the account to run the actions on objects that exist inside `container1` and start with `foo`.

- Tier blob to cool tier 30 days after last modification
- Tier blob to archive tier 90 days after last modification
- Delete blob 2,555 days (seven years) after last modification
- Delete blob snapshots 90 days after snapshot creation

JSON Copy

```
{
  "rules": [
    {
      "name": "ruleFoo",
      "enabled": true,
      "type": "Lifecycle",
```

```
      "definition": {
        "filters": {
          "blobTypes": [ "blockBlob" ],
          "prefixMatch": [ "container1/foo" ]
        },
        "actions": {
          "baseBlob": {
            "tierToCool": { "daysAfterModificationGreaterThan": 30 },
            "tierToArchive": { "daysAfterModificationGreaterThan": 90 },
            "delete": { "daysAfterModificationGreaterThan": 2555 }
          },
          "snapshot": {
            "delete": { "daysAfterCreationGreaterThan": 90 }
          }
        }
      }
    }
  ]
}
```

Rule filters

Filters limit rule actions to a subset of blobs within the storage account. If more than one filter is defined, a logical AND runs on all filters. Filters include:

Filter name	Filter type	Is Required
blobTypes	An array of predefined enum values.	Yes
prefixMatch	An array of strings for prefixes to be match. Each rule can define up to 10 prefixes. A prefix string must start with a container name.	No
blobIndexMatch	An array of dictionary values consisting of blob index tag key and value conditions to be matched. Each rule can define up to 10 blob index tag condition.	No

Rule actions

Actions are applied to the filtered blobs when the run condition is met.

Lifecycle management supports tiering and deletion of blobs and deletion of blob snapshots. Define at least one action for each rule on blobs or blob snapshots.

Action	Base Blob	Snapshot	Version
tierToCool	Supported for blockBlob	Supported	Supported
enableAutoTierToHotFromCool	Supported for blockBlob	Not supported	Not supported
tierToArchive	Supported for blockBlob	Supported	Supported
delete	Supported for blockBlob and appendBlob	Supported	Supported

> **Note**
>
> If you define more than one action on the same blob, lifecycle management applies the least expensive action to the blob. For example, action `delete` is cheaper than action `tierToArchive`. Action `tierToArchive` is cheaper than action `tierToCool`.

The run conditions are based on age. Base blobs use the last modified time to track age, and blob snapshots use the snapshot creation time to track age.

Action run condition	Condition value	Description
daysAfterModificationGreaterThan	Integer value indicating the age in days	The condition for base blob actions
daysAfterCreationGreaterThan	Integer value indicating the age in days	The condition for blob snapshot actions

Next unit: Implement Blob storage lifecycle policies

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Implement Blob storage lifecycle policies

3 minutes

You can add, edit, or remove a policy by using any of the following methods:

- Azure portal
- Azure PowerShell
- Azure CLI
- REST APIs

Below are the steps and some examples for the Portal and Azure CLI.

Azure portal

There are two ways to add a policy through the Azure portal: Azure portal List view, and Azure portal Code view.

Azure portal List view

1. Sign in to the Azure portal .
2. Select **All resources** and then select your storage account.
3. Under **Data management**, select **Lifecycle management** to view or change your rules.
4. Select the **List view** tab.
5. Select **Add rule** and then fill out the **Action set** form fields. In the following example, blobs are moved to cool storage if they haven't been modified for 30 days.
6. Select **Filter set** to add an optional filter. Then, select Browse to specify a container and folder by which to filter.
7. Select **Review + add** to review the policy settings.
8. Select **Add** to add the new policy.

Azure portal Code view

1. Follow the first three steps above in the **List view** section.
2. Select the **Code view** tab. The following JSON is an example of a policy that moves a block blob whose name begins with *log* to the cool tier if it has been more than 30 days since the blob was modified.

JSON Copy

```
{
  "rules": [
    {
      "enabled": true,
      "name": "move-to-cool",
      "type": "Lifecycle",
      "definition": {
        "actions": {
          "baseBlob": {
            "tierToCool": {
              "daysAfterModificationGreaterThan": 30
            }
          }
        },
        "filters": {
          "blobTypes": [
            "blockBlob"
          ],
          "prefixMatch": [
            "sample-container/log"
          ]
        }
      }
    }
  ]
}
```

3. Select **Save**.

Azure CLI

To add a lifecycle management policy with Azure CLI, write the policy to a JSON file, then call the `az storage account management-policy create` command to create the policy.

Azure CLI Copy

```
az storage account management-policy create \
    --account-name <storage-account> \
    --policy @policy.json \
    --resource-group <resource-group>
```

A lifecycle management policy must be read or written in full. Partial updates are not supported.

Next unit: Rehydrate blob data from the archive tier

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Rehydrate blob data from the archive tier

3 minutes

While a blob is in the archive access tier, it's considered to be offline and can't be read or modified. In order to read or modify data in an archived blob, you must first rehydrate the blob to an online tier, either the hot or cool tier. There are two options for rehydrating a blob that is stored in the archive tier:

- **Copy an archived blob to an online tier**: You can rehydrate an archived blob by copying it to a new blob in the hot or cool tier with the Copy Blob or Copy Blob from URL operation. Microsoft recommends this option for most scenarios.
- **Change a blob's access tier to an online tier**: You can rehydrate an archived blob to hot or cool by changing its tier using the Set Blob Tier operation.

Rehydrating a blob from the archive tier can take several hours to complete. Microsoft recommends rehydrating larger blobs for optimal performance. Rehydrating several small blobs concurrently may require additional time.

Rehydration priority

When you rehydrate a blob, you can set the priority for the rehydration operation via the optional `x-ms-rehydrate-priority` header on a Set Blob Tier or **Copy Blob/Copy Blob From URL** operation. Rehydration priority options include:

- **Standard priority**: The rehydration request will be processed in the order it was received and may take up to 15 hours.
- **High priority**: The rehydration request will be prioritized over standard priority requests and may complete in under one hour for objects under 10 GB in size.

To check the rehydration priority while the rehydration operation is underway, call Get Blob Properties to return the value of the `x-ms-rehydrate-priority` header. The rehydration priority property returns either *Standard* or *High*.

Copy an archived blob to an online tier

You can use either the **Copy Blob** or **Copy Blob from URL** operation to copy the blob. When you copy an archived blob to a new blob an online tier, the source blob remains unmodified in the archive tier. You must copy the archived blob to a new blob with a different name or to a different container. You cannot overwrite the source blob by copying to the same blob.

Copying an archived blob to an online destination tier is supported within the same storage account only. You cannot copy an archived blob to a destination blob that is also in the archive tier.

The following table shows the behavior of a blob copy operation, depending on the tiers of the source and destination blob.

	Hot tier source	**Cool tier source**	**Archive tier source**
Hot tier destination	Supported	Supported	Supported within the same account. Requires blob rehydration.
Cool tier destination	Supported	Supported	Supported within the same account. Requires blob rehydration.
Archive tier destination	Supported	Supported	Unsupported

Change a blob's access tier to an online tier

The second option for rehydrating a blob from the archive tier to an online tier is to change the blob's tier by calling **Set Blob Tier**. With this operation, you can change the tier of the archived blob to either hot or cool.

Once a **Set Blob Tier** request is initiated, it cannot be canceled. During the rehydration operation, the blob's access tier setting continues to show as archived until the rehydration process is complete.

To learn how to rehydrate a blob by changing its tier to an online tier, see Rehydrate a blob by changing its tier.

⊗ Caution

Changing a blob's tier doesn't affect its last modified time. If there is a lifecycle management policy in effect for the storage account, then rehydrating a blob with **Set Blob Tier** can result in a scenario where the lifecycle policy moves the blob back to the archive tier after rehydration because the last modified time is beyond the threshold set for the policy.

Next unit: Knowledge check

Continue >

How are we doing? ☆ ☆ ☆ ☆ ☆

200 XP

Knowledge check

3 minutes

Check your knowledge

1. Which access tier is considered to be offline and can't be read or modified?

- ○ Cool
- ◉ Archive

  ✔ That's correct. Blobs in the archive tier must be rehydrated to either the hot or cool tier before it can be read or modified.
- ○ Hot

2. Which of the following storage account types supports lifecycle policies?

- ◉ General Purpose v1

  ✘ That's incorrect. General Purpose v1 accounts need to be upgraded to v2 before lifecycle policies are supported.
- ○ General Purpose v2

  ✔ That's correct. Azure Blob storage lifecycle management offers a rich, rule-based policy for General Purpose v2 and Blob storage accounts.
- ○ FileStorage

Next unit: Summary

Continue

How are we doing? ☆ ☆ ☆ ☆ ☆

100 XP

Summary

3 minutes

In this module you learned how to:

- Describe how each of the access tiers are optimized.
- Create and implement a lifecycle policy.
- Rehydrate blob data stored in an archive tier.

Module complete:

Unlock achievement

How are we doing? ☆ ☆ ☆ ☆ ☆