2. Develop for Azure Storage → 2.1 Develop solutions that use Azure Blob Storage → 2.2.3 Implement storage policies and data lifecycle management

1. What is Azure Blob Lifecycle Management?
2. How do you define a lifecycle management rule?
3. How do you move blobs between access tiers (Hot, Cool, Archive)?
4. How do you configure auto-delete for old blobs?
5. How do you apply rules based on blob metadata or naming patterns?
6. How do you create lifecycle management rules using the Azure Portal?
7. How do you create lifecycle policies programmatically (Azure CLI, SDK)?
8. What are best practices for setting retention and tiering policies?
9. How does Archive rehydration work and what are the implications?
10. How do you monitor and troubleshoot lifecycle policy actions?

---

1. What is Azure Blob Lifecycle Management?

- A feature that automatically moves, deletes, or archives blobs based on rules and conditions like age, last modified date, or access tier.
- Reduces storage costs and enforces data retention policies.

---

2. How do you define a lifecycle management rule?

- A rule consists of:
  - **Filters** (prefix match, blob type, metadata conditions).
  - **Actions** (move to Cool/Archive tier, delete blob).
- Rules are evaluated **daily** by Azure.

---

3. How do you move blobs between access tiers (Hot, Cool, Archive)?

- Define a lifecycle rule that moves blobs based on conditions:
  - Example: Move to **Cool** if not modified for 30 days.
  - Example: Move to **Archive** if not modified for 180 days.
- No manual intervention needed once the policy is active.

---

4. How do you configure auto-delete for old blobs?

- Set a **Delete** action in a lifecycle management rule.
- Example: Delete blobs **90 days** after the last modified date.
- Can combine delete action with filters (e.g., only for blobs with a specific prefix).

---

5. How do you apply rules based on blob metadata or naming patterns?

- Use **filters** when defining the rule:
  - **Prefix match**: Target blobs under a virtual folder path.
  - **Blob index tags**: Target blobs with specific metadata conditions (e.g., env=prod).
- Example filter:

```
"prefixMatch": ["logs/"],
"blobTypes": ["blockBlob"]
```

6. How do you create lifecycle management rules using the Azure Portal?

- Navigate to Storage Account → **Data Management** → **Lifecycle Management**.
- Add a rule:
  - Define conditions (e.g., last modified > 30 days).
  - Specify actions (move, delete).
- Save and enable the rule — applies automatically to matching blobs.

---

7. How do you create lifecycle policies programmatically (Azure CLI, SDK)?

- **Azure CLI Example:**

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

- **C# SDK Example:**

```
var managementPolicy = new ManagementPolicy
{
  Policy = JsonConvert.DeserializeObject<ManagementPolicySchema>(policyJson)
};
await storageAccount.UpdateAsync(managementPolicy: managementPolicy);
```

- Policy is defined in a JSON file describing rules and actions.

---

8. What are best practices for setting retention and tiering policies?

- Use Cool tier for infrequently accessed data (accessed > 30 days).
- Use Archive tier for rarely accessed data (accessed > 180 days).
- Avoid aggressive delete rules unless compliance requirements mandate it.
- Test lifecycle rules in non-production first to verify behavior.

---

9. How does Archive rehydration work and what are the implications?

- **Rehydration** moves blobs from Archive to Hot or Cool tier to make them accessible.
- It can take **hours** (up to 15 hours standard, faster rehydration possible).
- Rehydration is **asynchronous**; status must be polled using blob properties.

---

10. How do you monitor and troubleshoot lifecycle policy actions?

- Review **Activity Log** in Azure Portal for lifecycle policy operations.
- Check **Blob properties** to verify last tier change date.
- Use **Azure Monitor metrics** to track storage capacity changes by tier.