Microsoft

Module 03: Develop solutions that use blob storage

Microsoft

Topics

- Azure Blob storage core concepts
- Managing the Azure Blob storage lifecycle
- Working with Azure Blob storage

Lesson 01: Azure Blob storage core concepts

Azure Storage overview

Disks

Persistent disks for Azure IaaS VMs

Premium storage disk options

Storage Accounts

Files

Fully managed file shares in the cloud

SMB and REST access

"Lift and shift" legacy apps

Sync with on-premises

Blobs

Highly scalable, REST-based cloud object store

Block blobs: Sequential file I/O

Page blobs: Random-write pattern data

Append blobs

Tables

Massive auto-scaling NoSQL store

Dynamic scaling based on load

Queues

Reliable queues at scale for cloud services

Decouple and scale components

Message visibility

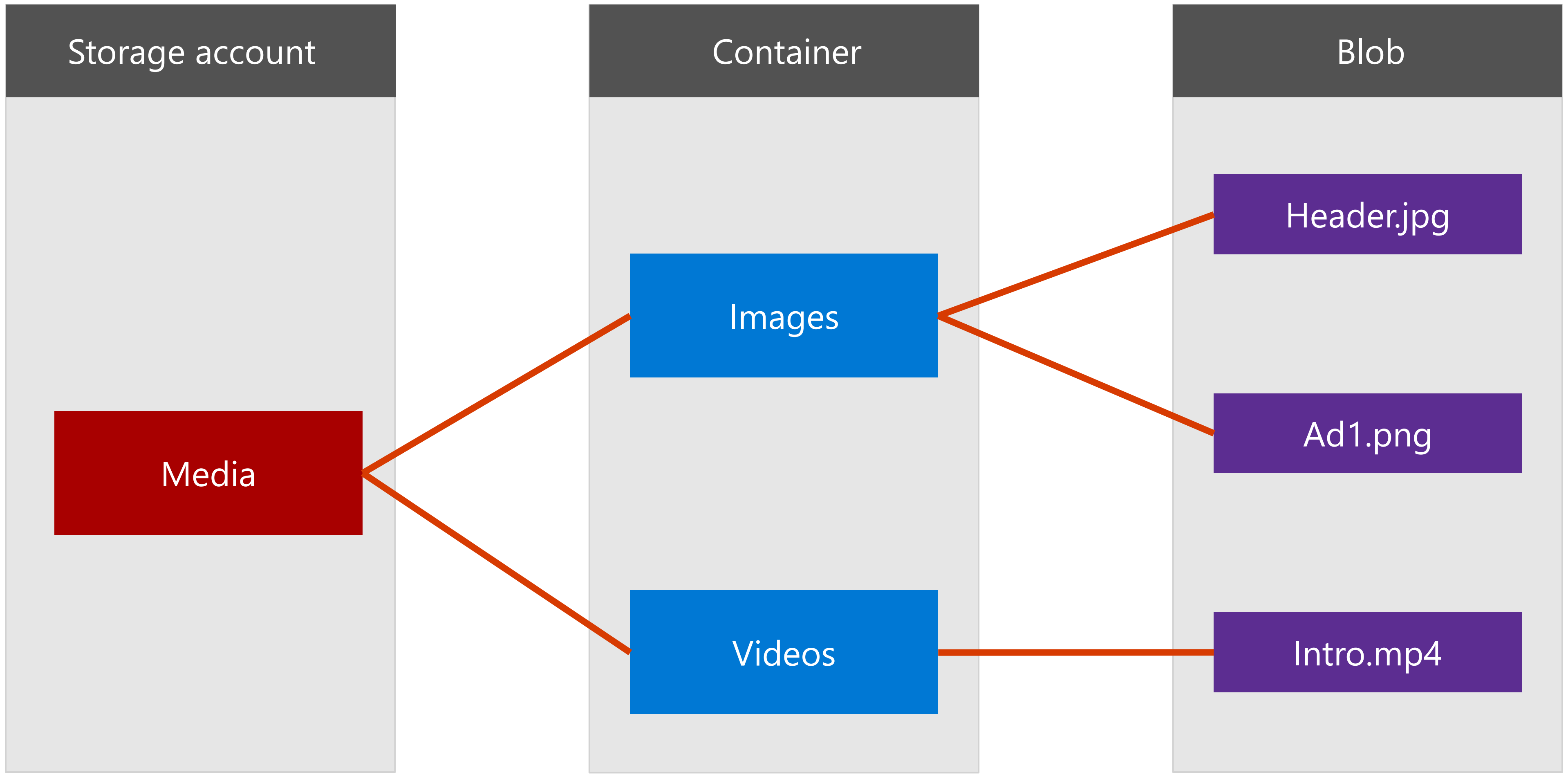
Built on a unified Distributed Storage System

Durability, Encryption at Rest, Strongly Consistent Replication, Fault Tolerance, Auto Load-Balancing

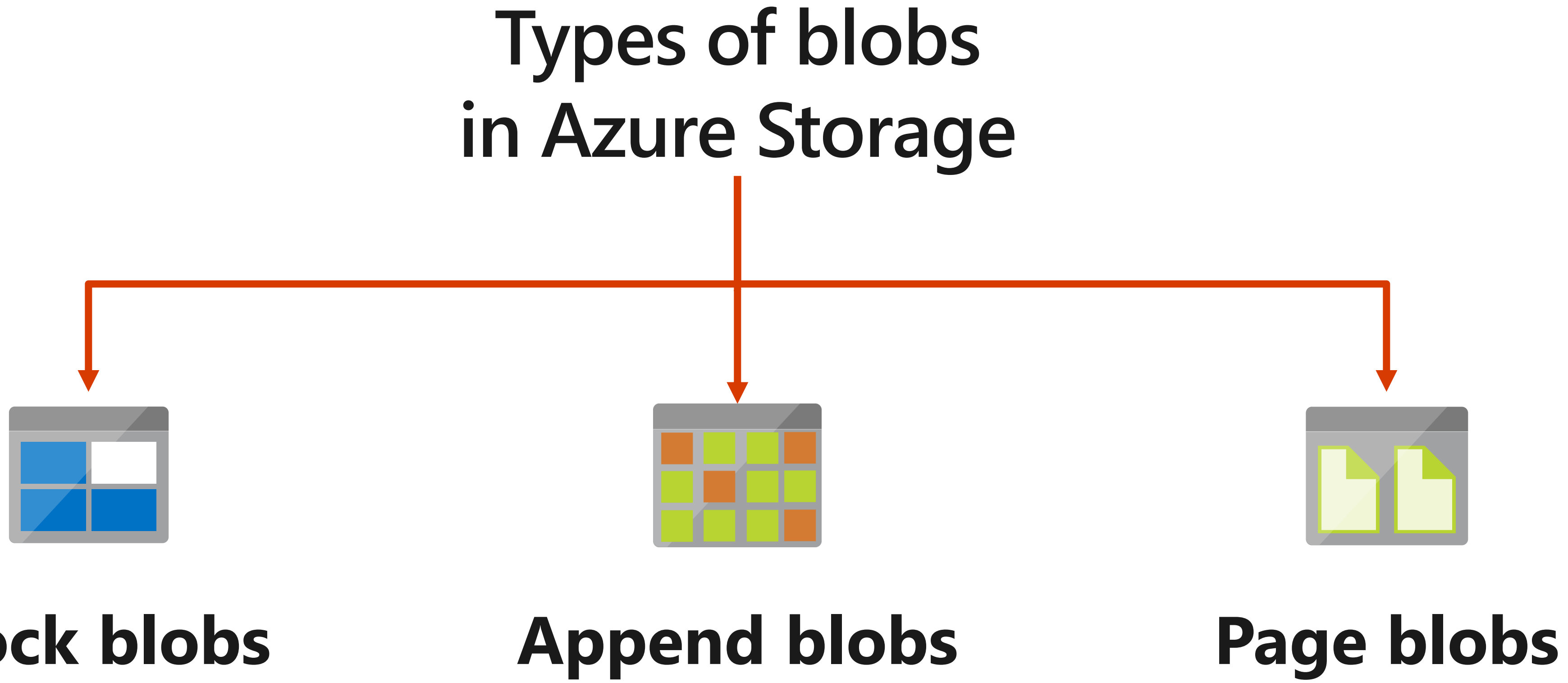
Azure Blob storage

- Object storage solution in the cloud
- Blob storage is designed for:
  - Serving images or documents directly to a browser
  - Storing files for distributed access
  - Streaming video and audio
  - Writing to log files
  - Storing data for backup and restore, disaster recovery, and archiving
  - Storing data for analysis by an on-premises or Azure-hosted service
- Accessible via a HTTP/HTTPS API

Azure Blob storage resource hierarchy



Blob types

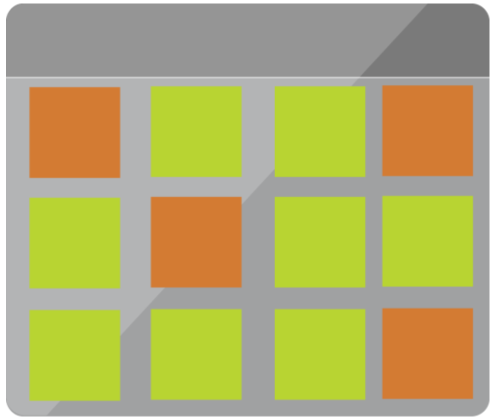

Types of blobs
in Azure Storage
Block blobs
Append blobs
Page blobs

Block blobs

- Comprise blocks of data
- Ideal for data that is stored in blocks—up to 100-MB chunks
- Simultaneous upload of large blobs with a single write operation
- A single block blob can include up to 50,000 blocks

Append blobs

- Append blobs include the following characteristics:
  - They are composed of blocks
  - They are optimized for append operations
  - They are ideal for performant logging

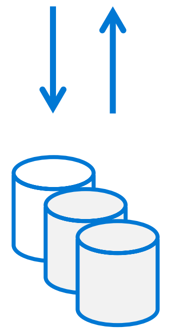


Page blobs

- Composed of 512-byte pages
- Similar to hard disk storage
- Ideal for virtual hard disks
- Pages created by initializing the page blob and specifying the size
- Content to be added within 512-byte page boundaries
- Writes to page blobs commit immediately

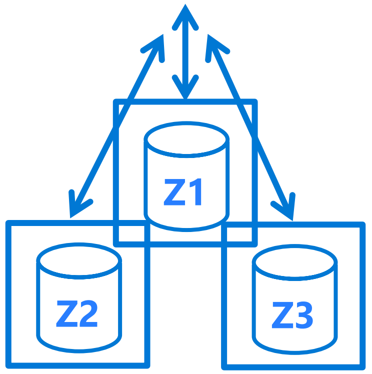
Storage durability options

Single region



LRS

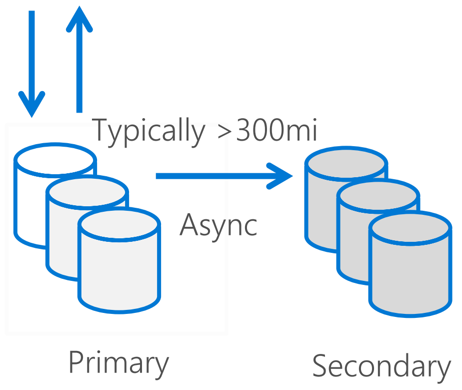
- Three replicas, one region
- Protects against disk, node, rack failures
- Write is acknowledged when all replicas are committed
- Superior to dual-parity RAID



ZRS

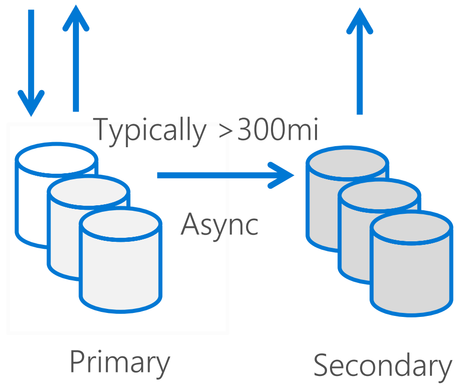
- Three replicas, three zones, one region
- Protects against disk, node, rack, and zone failures
- Synchronous writes to all three zones

Multiple regions



GRS

- Six replicas, two regions (three per region)
- Protects against major regional disasters
- Asynchronous copy to secondary

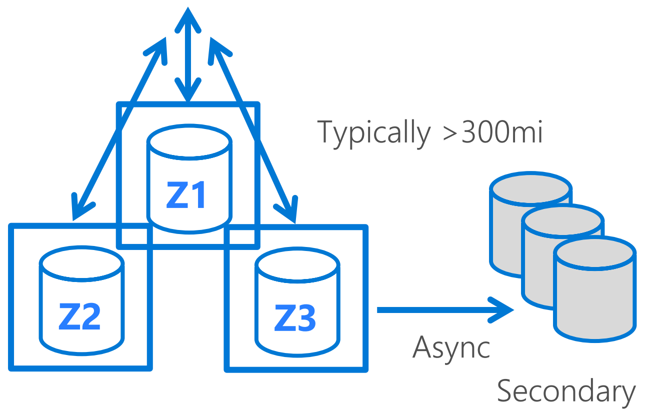


RA-GRS

- GRS + read access to secondary
- Separate secondary endpoint
- Recovery point objective (RPO) delay to secondary can be queried

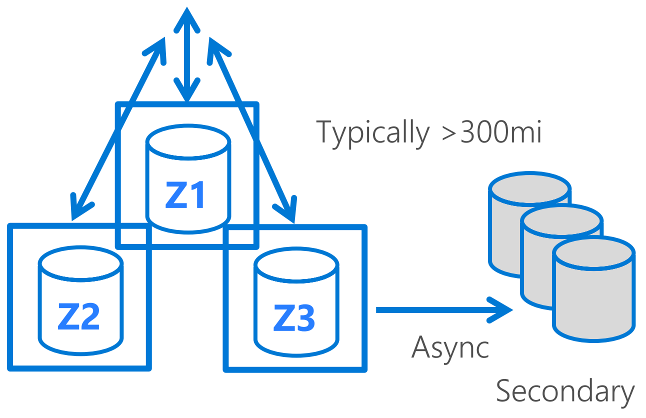
Storage durability options (continued)

Multiple regions



GZRS

- Six replicas, 3+1 zones, two regions
- Protects against disk, node, rack, zone, and region failures
- Synchronous writes to all three zones and asynchronous copy to secondary



RA-GZRS

- GZRS + read access to secondary
- Separate secondary endpoint
- RPO delay to secondary can be queried

Demonstration: Create a block blob storage account

Lesson 02: Managing the Azure Blob storage lifecycle

Storage tiers

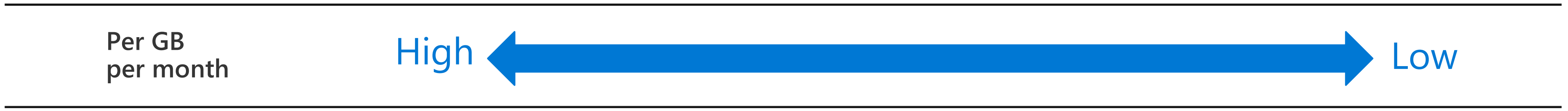
You can use storage tiers to tune performance and cost to a ratio that’s ideal for your solution

Performance tier

Access tiers

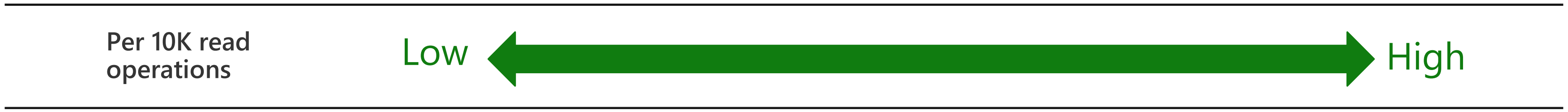
Premium	Hot	Cool	Archive
Low and consistent latency data	Frequently accessed data	Less frequently accessed data	Rarely accessed data

Storage tier pricing



Premium	Hot	Cool	Archive
Low and consistent latency data	Frequently accessed data	Less frequently accessed data	Rarely accessed data

Storage tier pricing (continued)



Premium

Low and consistent latency data

Hot

Frequently accessed data

Cool

Less frequently accessed data

Archive

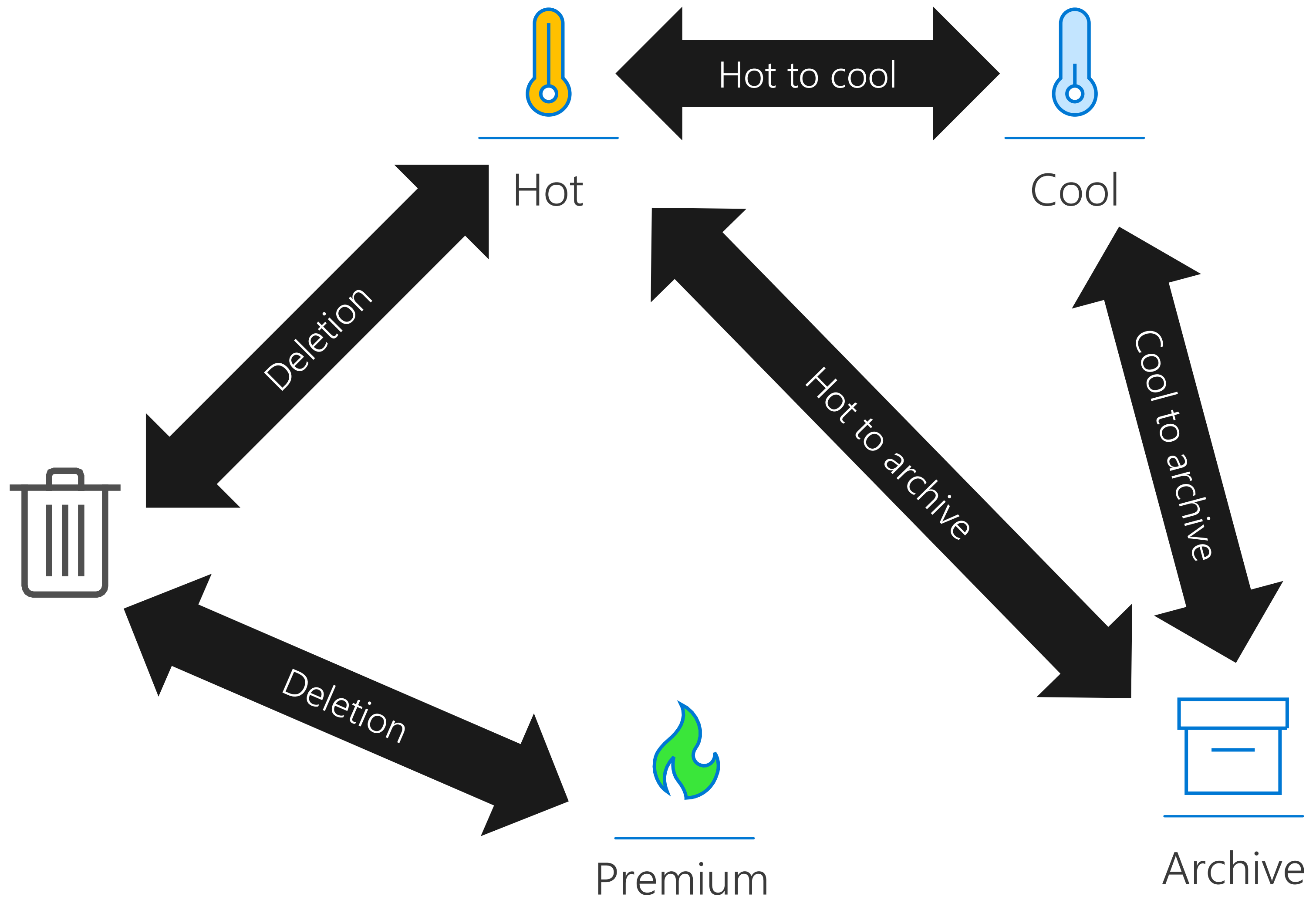
Rarely accessed data

Lifecycle management

Rule-based automation for data tiering and retention management:

- Rules run daily at the storage account
- Supports:
  - General-purpose v2 storage accounts
  - Blob storage
  - Premium BlockBlob (only supports deletion for lifecycle management)
- Prefix filters enable targeting of containers or sets of blobs

Example of lifecycle management flows



Policy example

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

Parameter name	Parameter type	Required
name	String	True
enabled	Boolean	False
type	An enum value	True
definition	An object that defines the lifecycle rule	True

Demonstration: Adding a policy to Azure Blob storage

Lesson 03: Working with Azure Blob storage

Managing blob properties and metadata

- Containers and blobs support custom metadata
  - Represented by using HTTP headers
- Metadata headers are set on requests
  - During the creation of a new resource
  - During a special operation that explicitly creates a property on an existing resource
- Metadata headers start with the **x-ms-meta-*** prefix:

```
x-ms-meta-name:string-value
```

Blob container properties

Property	Description
ETag	This is a standard HTTP header that gives a value that is unchanged unless a property of the container is changed. This value can be used to implement optimistic concurrency with the blob containers.
LastModified	This property indicates when the container was last modified.
PublicAccess	This property indicates the level of public access that is allowed on the container. Valid values include Blob, Container, Off, and Unknown.
HasImmutabilityPolicy	This property indicates whether the container has an immutability policy. An immutability policy will help ensure that blobs are stored for a minimum amount of retention time.
HasLegalHold	This property indicates whether the container has an active legal hold. A legal hold will help ensure that blobs remain unchanged until the hold is removed.

Manipulating blob container properties in .NET

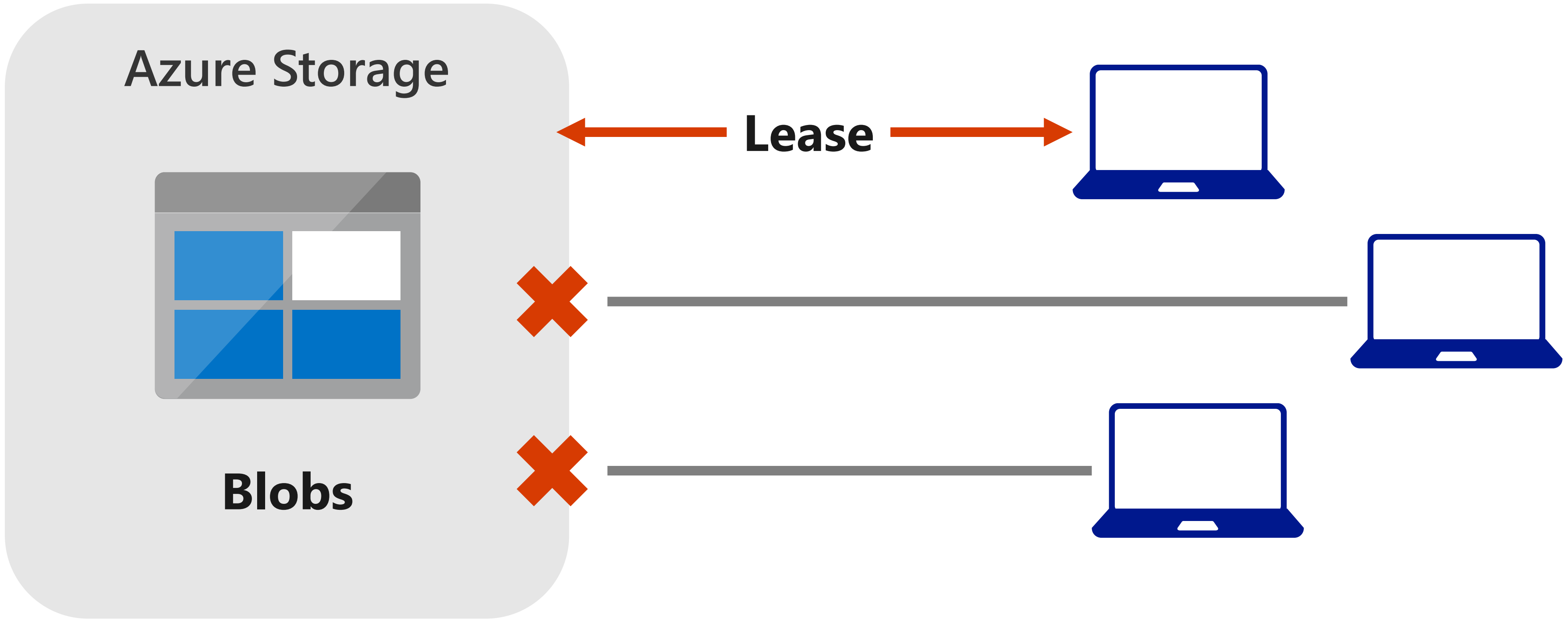
```
CloudBlobClient client = storageAccount.CreateCloudBlobClient();

CloudBlobContainer container = client.GetContainerReference("images");

container.CreateIfNotExists();

await container.FetchAttributesAsync();

container.Properties.*
```

Manipulating blob container metadata in .NET

```
CloudBlobClient client = storageAccount.CreateCloudBlobClient();

CloudBlobContainer container = client.GetContainerReference("images");

container.CreateIfNotExists();

container.Metadata.Add("docType", "textDocuments");
container.Metadata["category"] = "guidance";

await container.SetMetadataAsync();
```

Demonstration: Using the Azure Blob storage client library for .NET v11

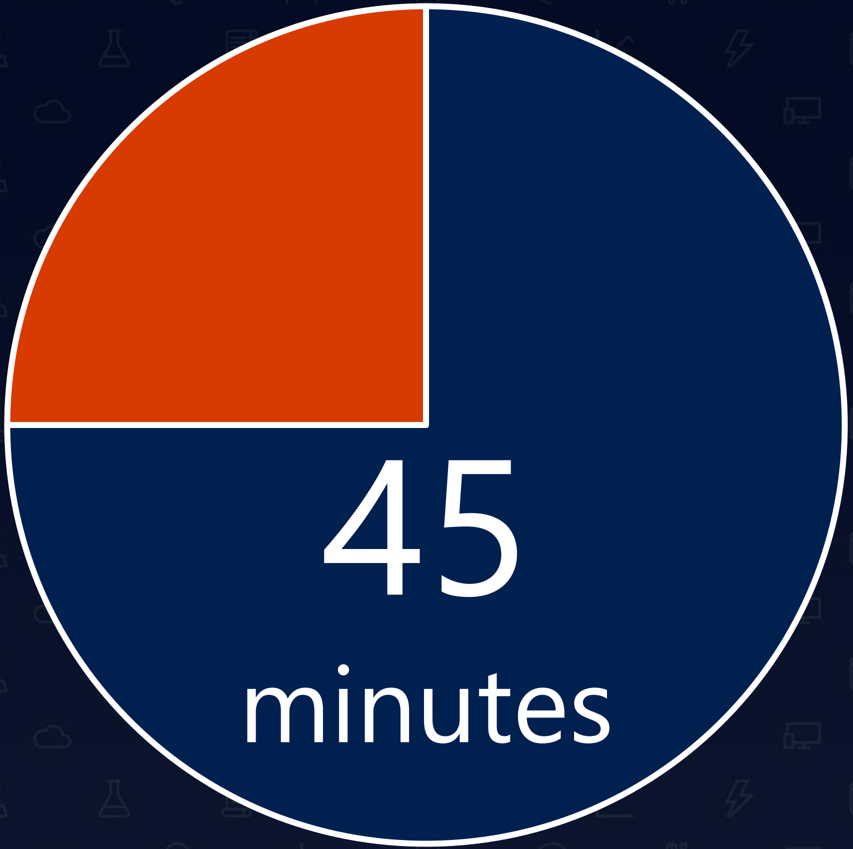
Exclusive access for modifying a blob



Lease Blob operation

- Establishes a lock on a blob for write and delete
  - Duration is typically 15 to 60 seconds
  - Optionally, you can establish an infinite lock
- Operation has five modes
  - Acquire
  - Renew
  - Change
  - Release
  - Break (end the lease but prevent other clients from acquiring a new lease)

Lab: Retrieving Azure Storage resources and metadata by using the .NET SDK



Lab: Retrieving Azure Storage resources and metadata by using the .NET SDK

Duration

Lab sign-in information

AZ204-SEA-DEV

Username: Admin

Password: Pa55w.rd

Microsoft