AZURE BLOB STORAGE

Presenter: Mai Hoang Giang

Presenter: Mai Hoang Giang

• Content

- What is it?
- Redundancy options
- Access tiers
- Security
- Manage by using Azure portal
- Develop with .NET
- Discussion

•1

•What is it?

Presenter: Mai Hoang Giang

• What is it?

Storage solution for cloud

Is optimized for storing massive amounts of unstructured data: text, binary

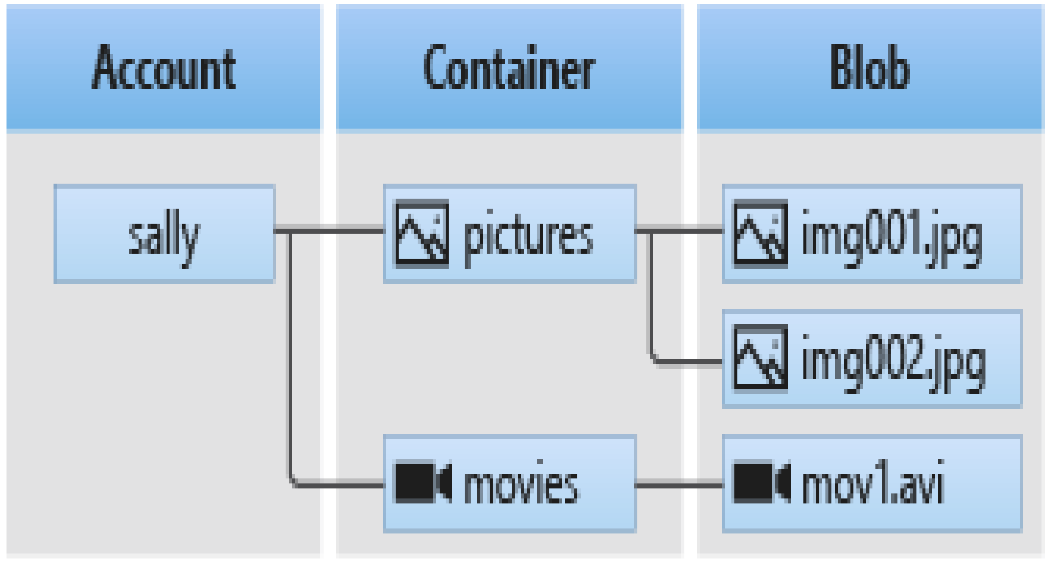
Presenter: Mai Hoang Giang

• What is designed for?

- Storing images or documents and accessing directly to a browser.

Storing files for distributed access.

• Resources



Example: mystorageaccount
http://mystorageaccount.blob.core.windows.net

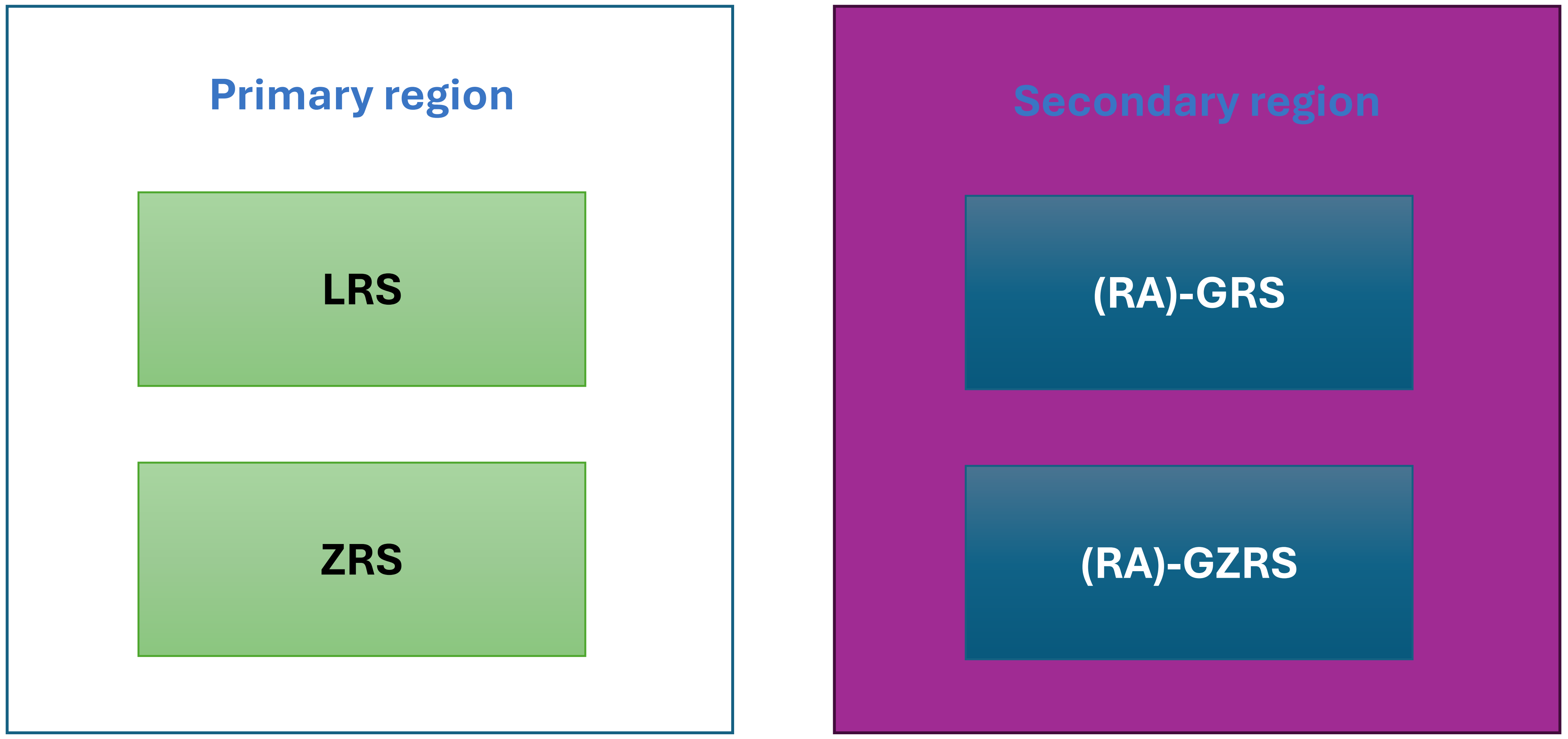
- Storage account
  - Standard General purpose v2
  - Premium file shares
  - Premium Block blob
  - Premium Page blob
- Container
- Blob
  - Block blob
  - Append blob
  - Page blob

•2

• Redundancy Options

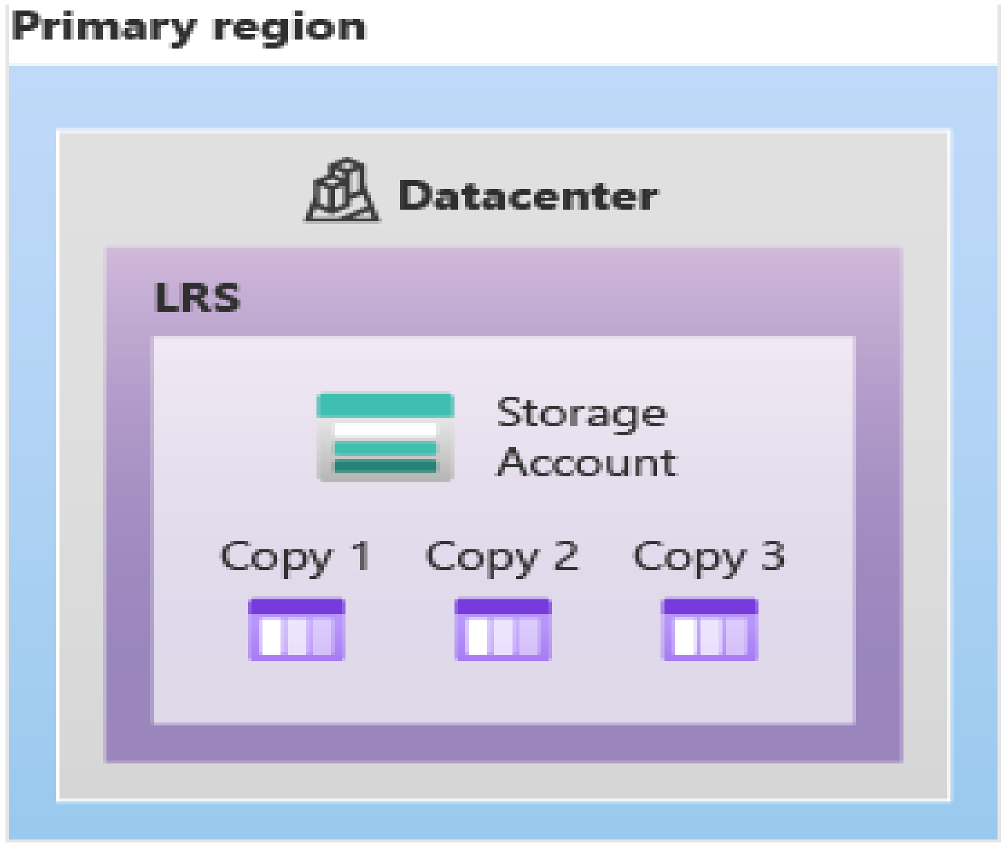
Presenter: Mai Hoang Giang

- **Redundancy options**



• Redundancy in primary region - LRS

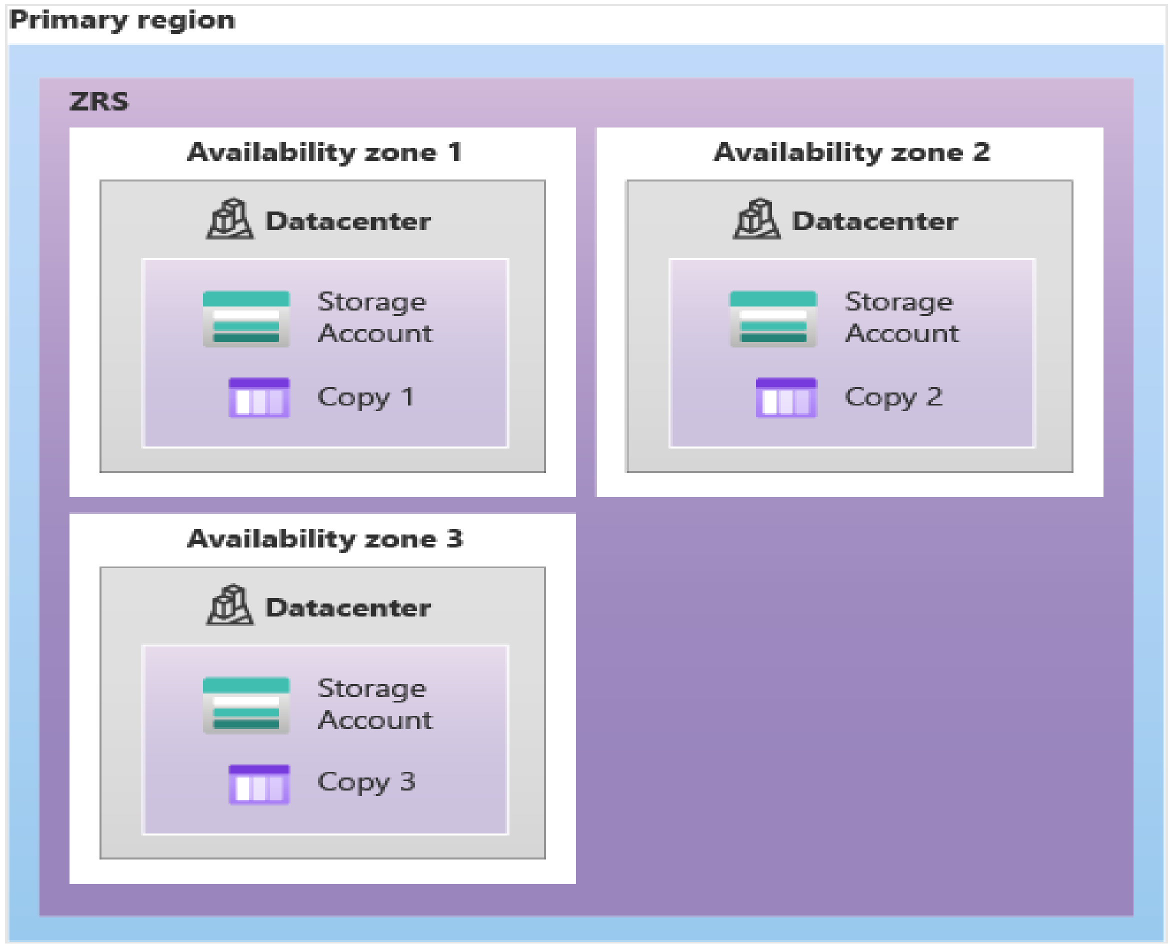
- Lowest cost
- Least durability
- Protect again server failures



Presenter: Mai Hoang Giang

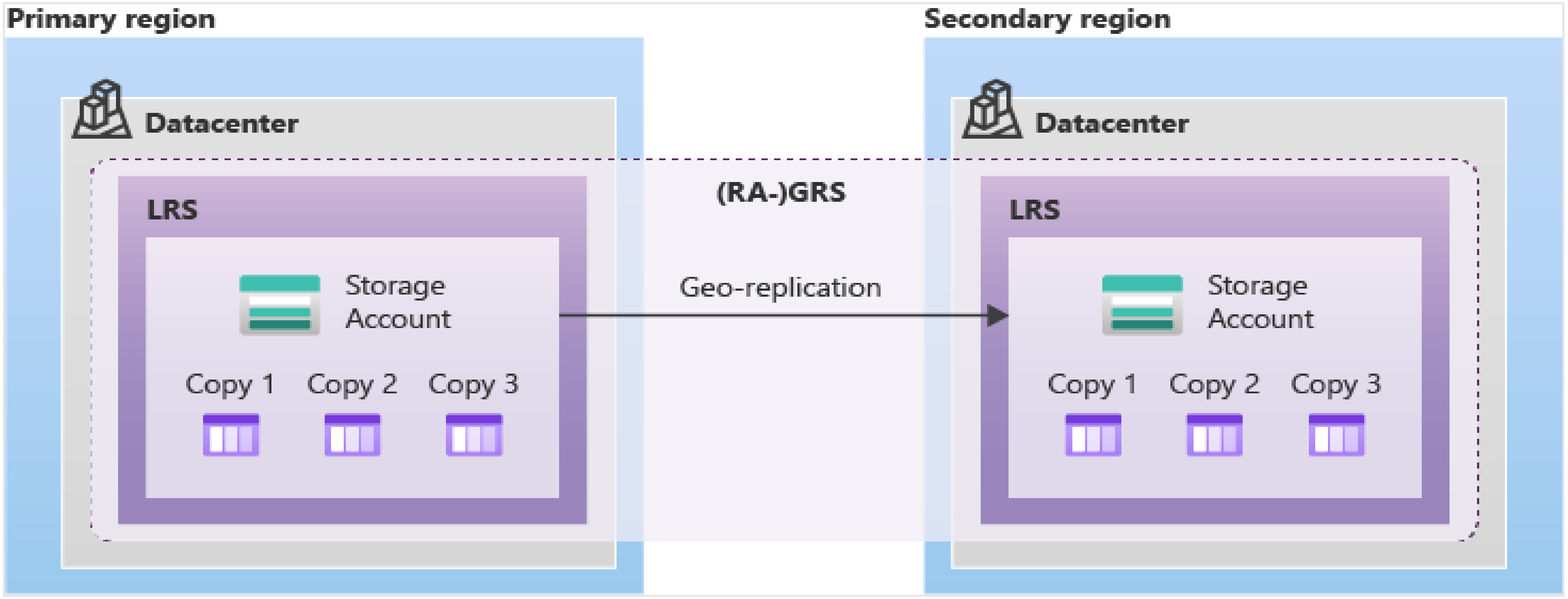
• Redundancy in primary region - ZRS

- Zone is a separate physical location
- Data is still accessible if a zone becomes unavailable.
- Synchronously replicate data



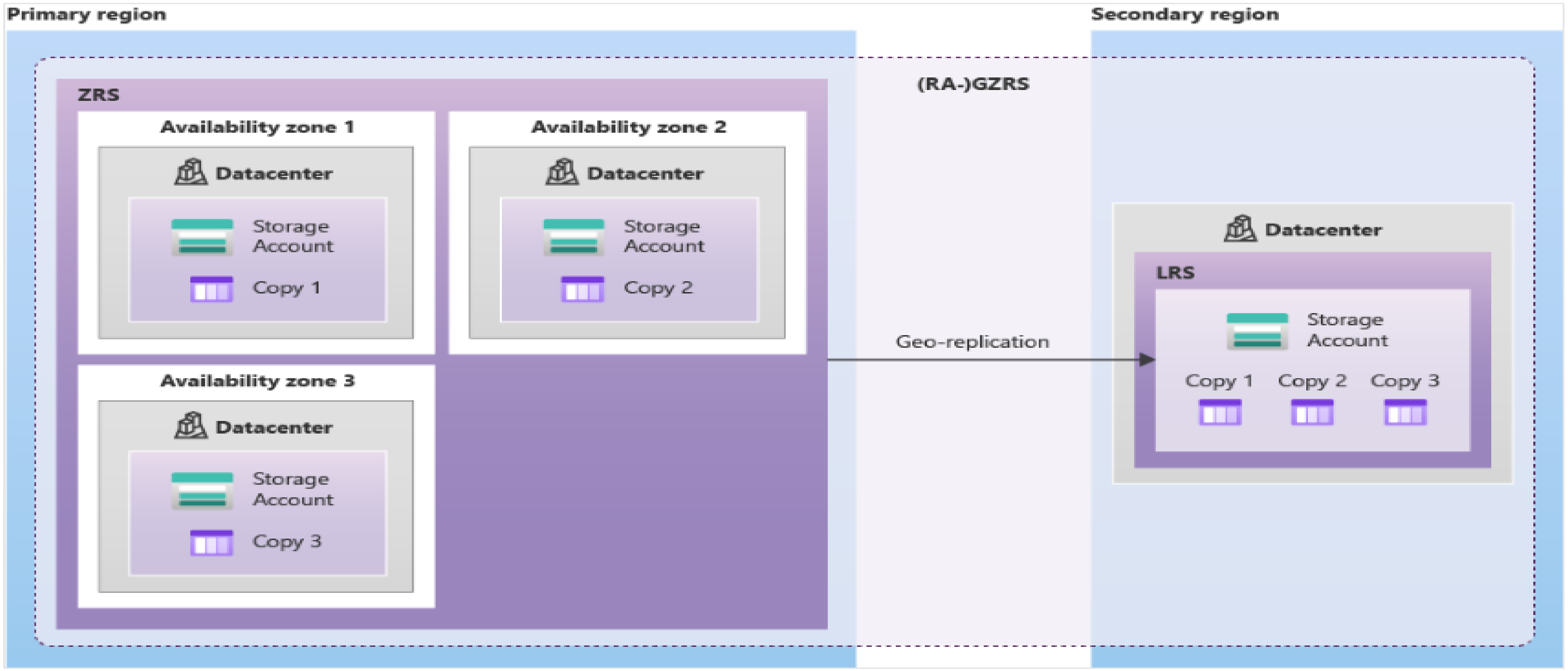
Presenter: Mai Hoang Giang

• Redundancy in secondary region - GRS



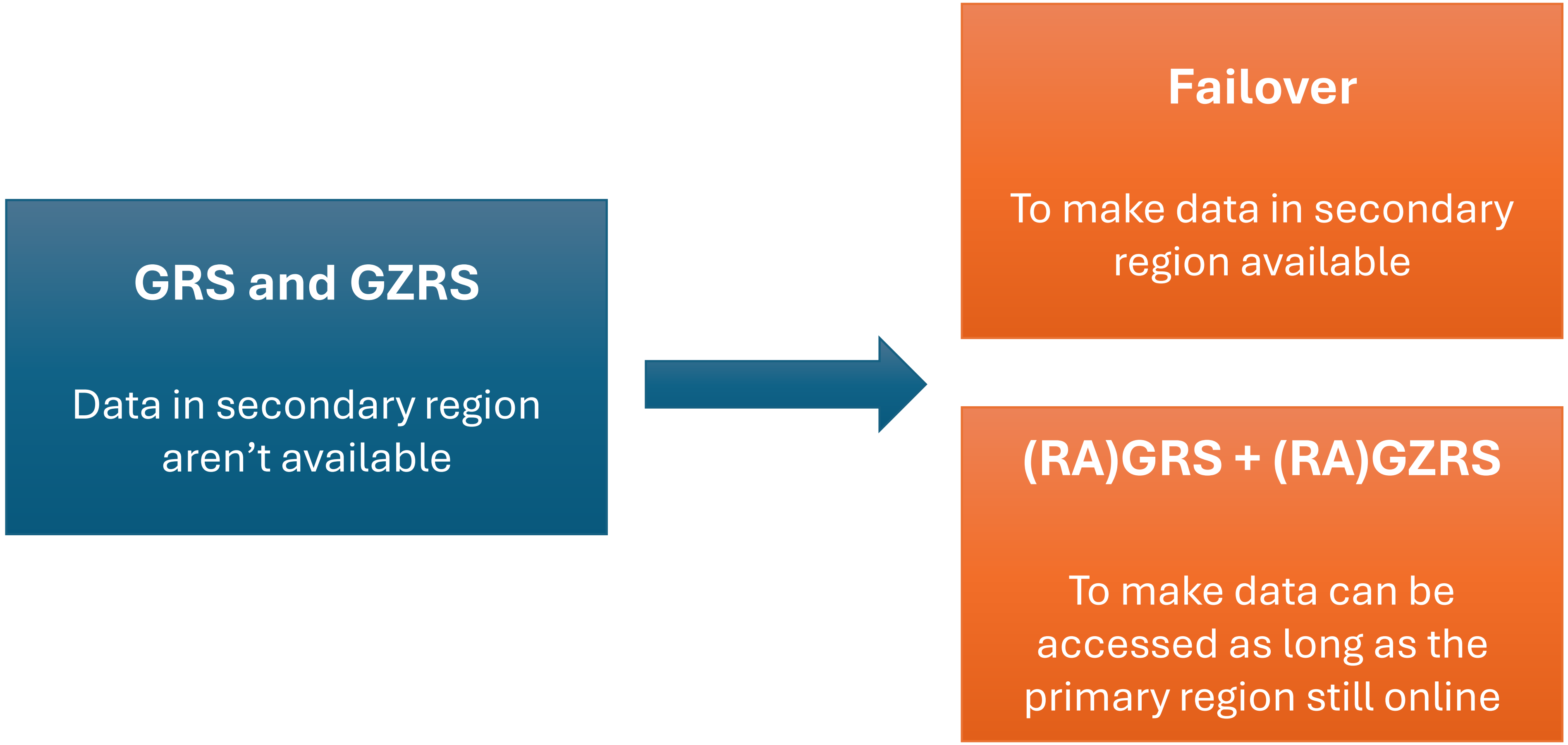
Presenter: Mai Hoang Giang

• Redundancy in secondary region - GZRS



Presenter: Mai Hoang Giang

• Redundancy – Read access to data



GRS and GZRS

Data in secondary region aren’t available

Failover

To make data in secondary region available

(RA)GRS + (RA)GZRS

To make data can be accessed as long as the primary region still online

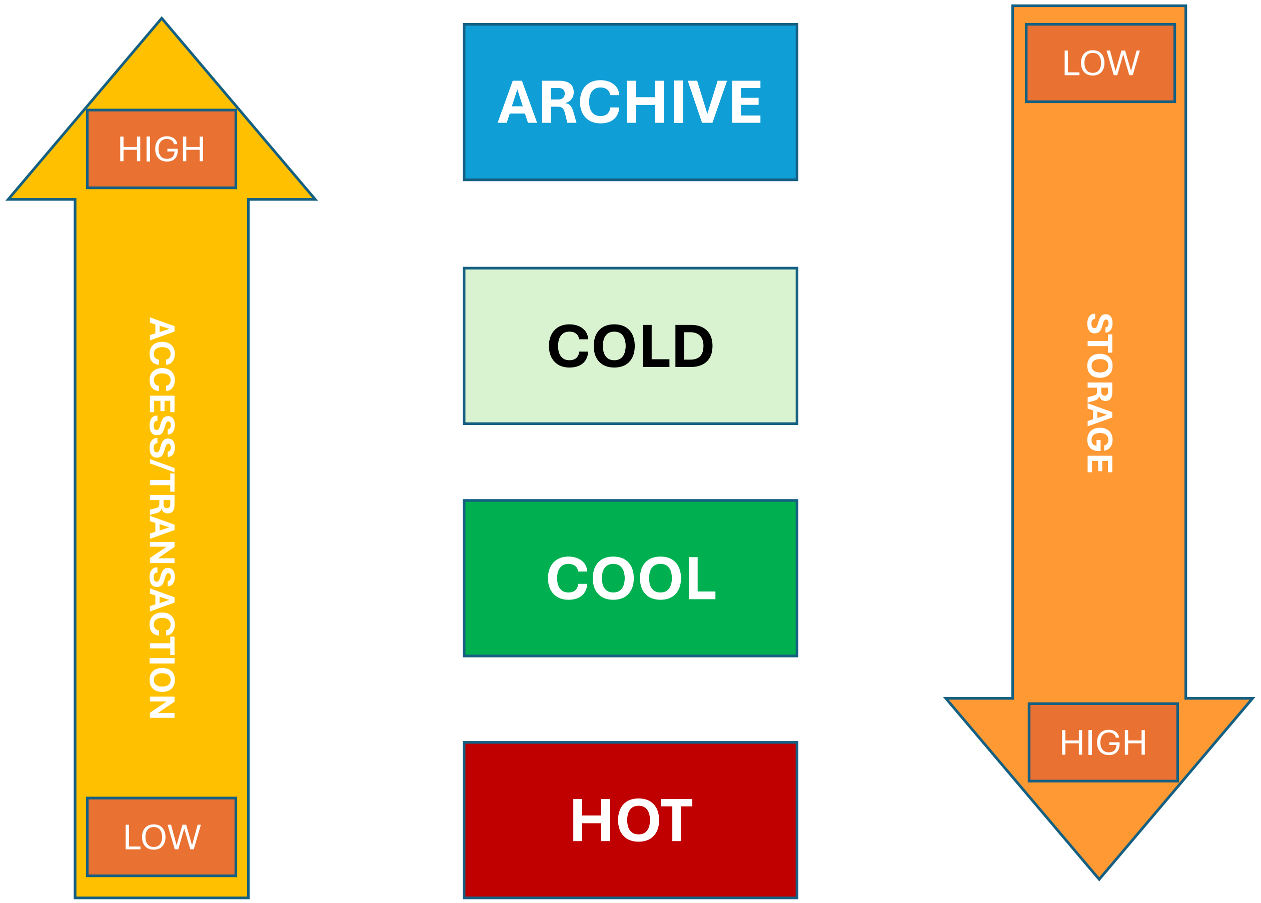
Presenter: Mai Hoang Giang

3

Storage Access Tiers

Presenter: Mai Hoang Giang

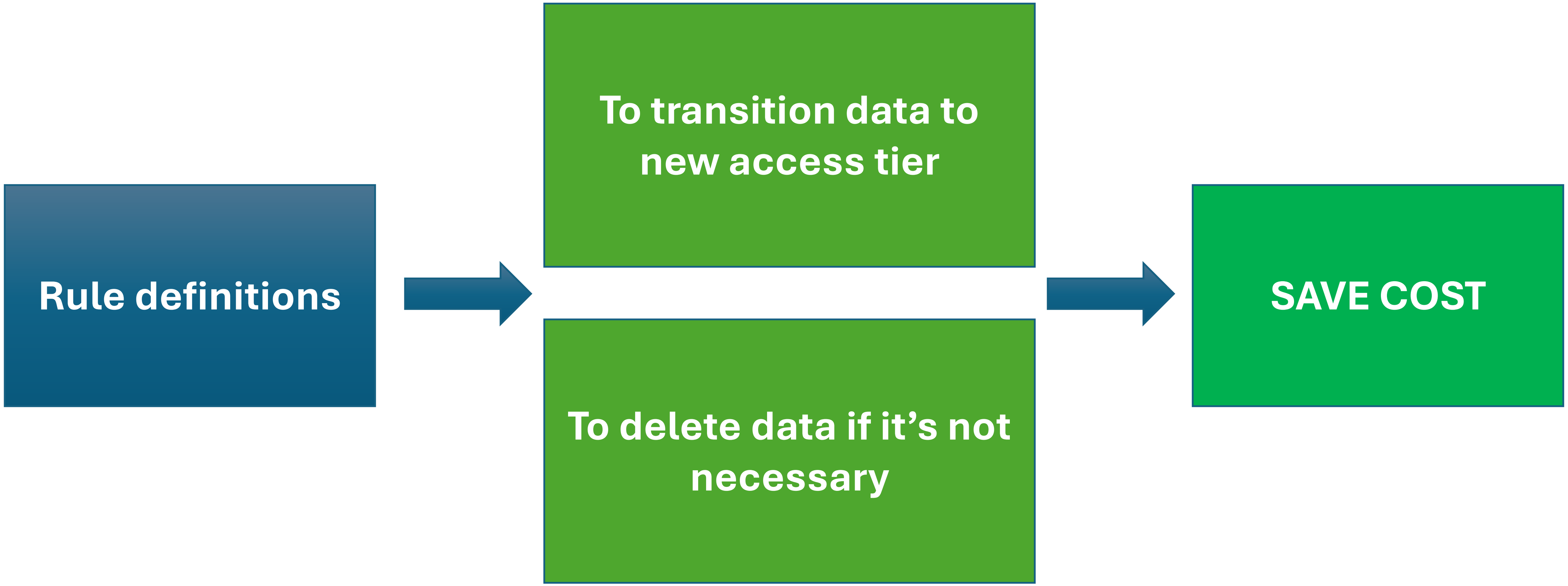
• Access tiers



• Access tiers comparation table

	Hot tier	Cool tier	Cold tier	Archive tier
Availability	99,9%	99%	99%	99%
Usage charges	Higher storage cost Lowest access cost	Lowest storage cost Higher access cost	Lowest storage cost Higher access cost	Lowest storage cost Higher access cost
Minimum store day		30d	90d	180d
Latency	Milliseconds	Milliseconds	Milliseconds	Hours
Usecase	Accessed frequently	Accessed less frequently	Rarely accessed	Rarely accessed, latency

• Blob lifecycle management


Rule definitions
To transition data to new access tier
To delete data if it’s not necessary
SAVE COST

Presenter: Mai Hoang Giang

4

Security

Presenter: Mai Hoang Giang

• Authorization

Microsoft Entra ID

Share access signatures (SAS)

Share key by using connection string

High security

Low security

Presenter: Mai Hoang Giang

• Networking

Deny access to traffic from all networks

Grant access to traffic from specific virtual networks

Grant access to traffic from selected public internet IP address ranges

$\Rightarrow$

To limit resources access to storage account for security reason

5

Manage by using Azure portal

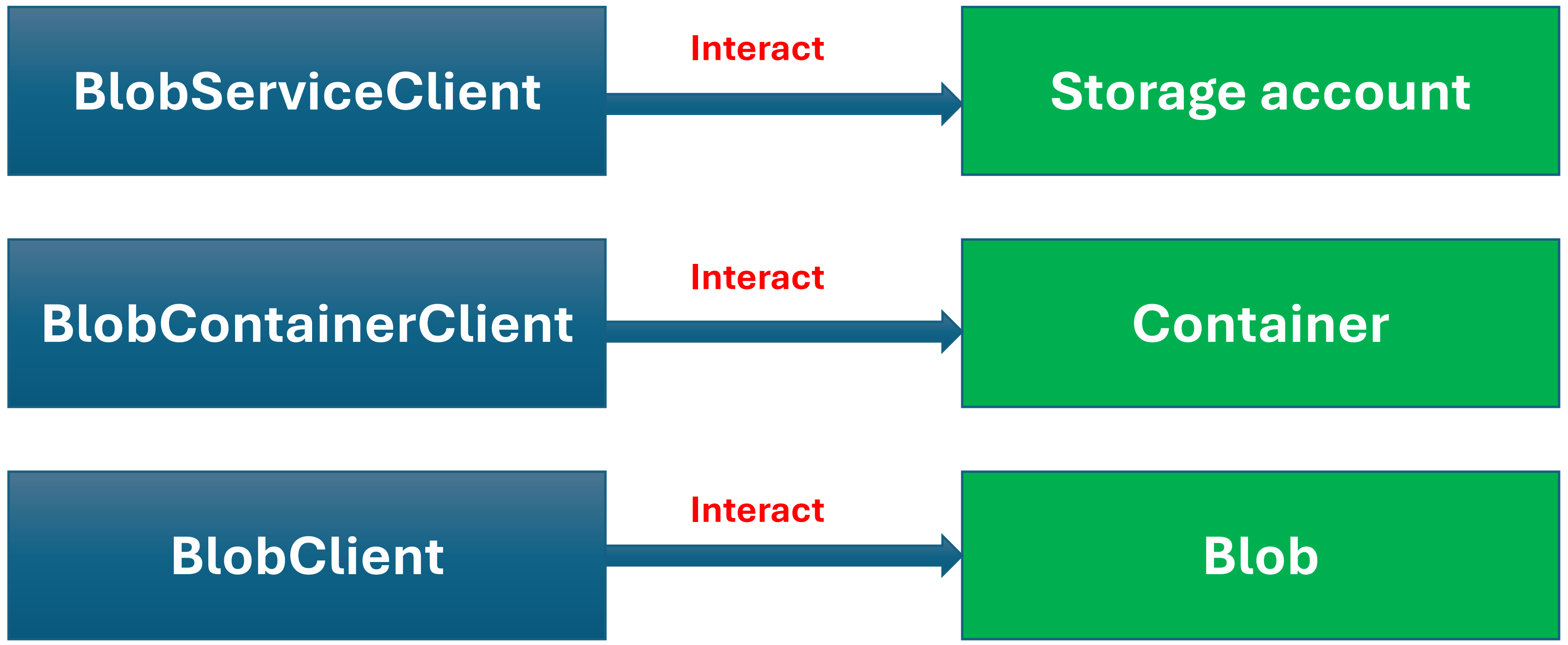
Presenter: Mai Hoang Giang

6

Develop with .Net

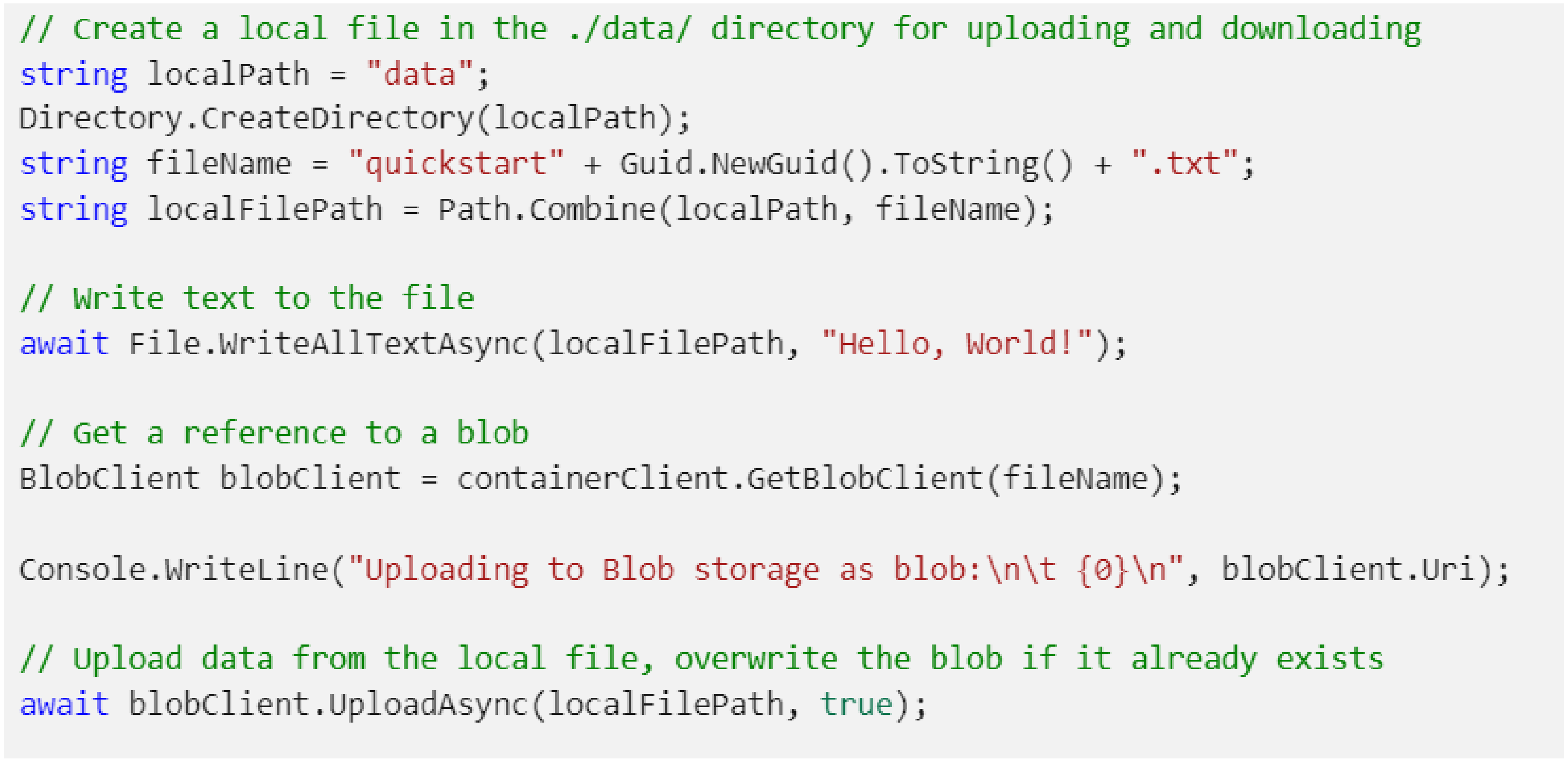
Presenter: Mai Hoang Giang

• Client Services



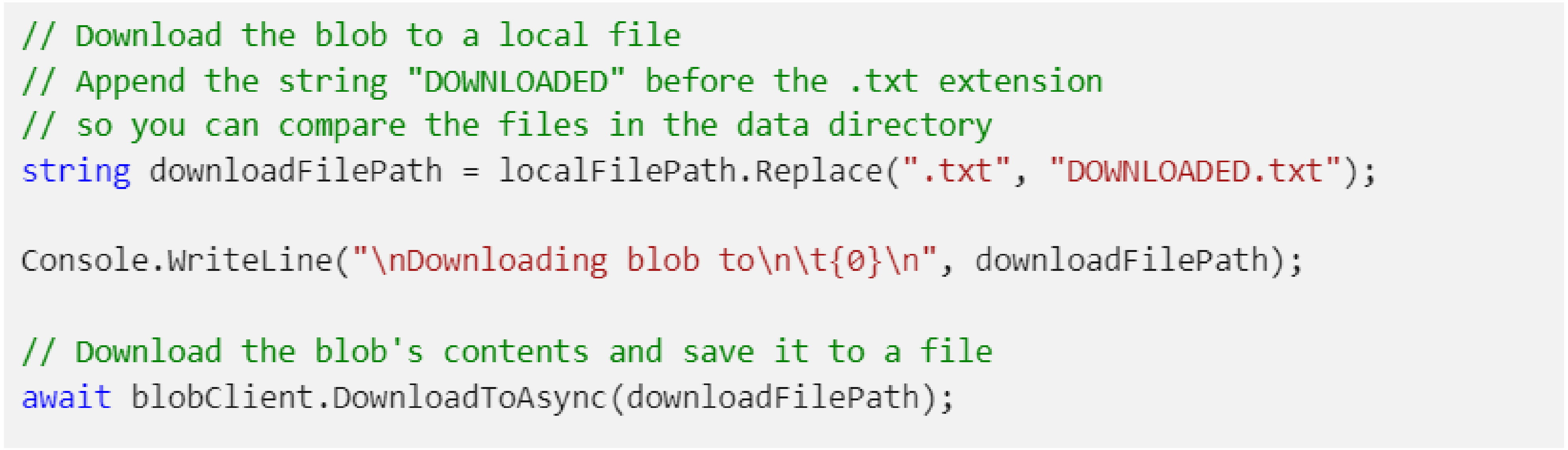
Presenter: Mai Hoang Giang

• Upload Blob to container



```
// Create a local file in the ./data/ directory for uploading and downloading
string localPath = "data";
Directory.CreateDirectory(localPath);
string fileName = "quickstart" + Guid.NewGuid().ToString() + ".txt";
string localFilePath = Path.Combine(localPath, fileName);

// Write text to the file
await File.WriteAllTextAsync(localFilePath, "Hello, World!");

// Get a reference to a blob
BlobClient blobClient = containerClient.GetBlobClient(fileName);

Console.WriteLine("Uploading to Blob storage as blob:\n\t {0}\n", blobClient.Uri);

// Upload data from the local file, overwrite the blob if it already exists
await blobClient.UploadAsync(localFilePath, true);
```

• Download Blob from container



```
// Download the blob to a local file
// Append the string "DOWNLOADED" before the .txt extension
// so you can compare the files in the data directory
string downloadFilePath = localFilePath.Replace(".txt", "DOWNLOADED.txt");

Console.WriteLine("\nDownloading blob to\n\t{0}\n", downloadFilePath);

// Download the blob's contents and save it to a file
await blobClient.DownloadToAsync(downloadFilePath);
```

• Retry options with RetryOptions class

Delay

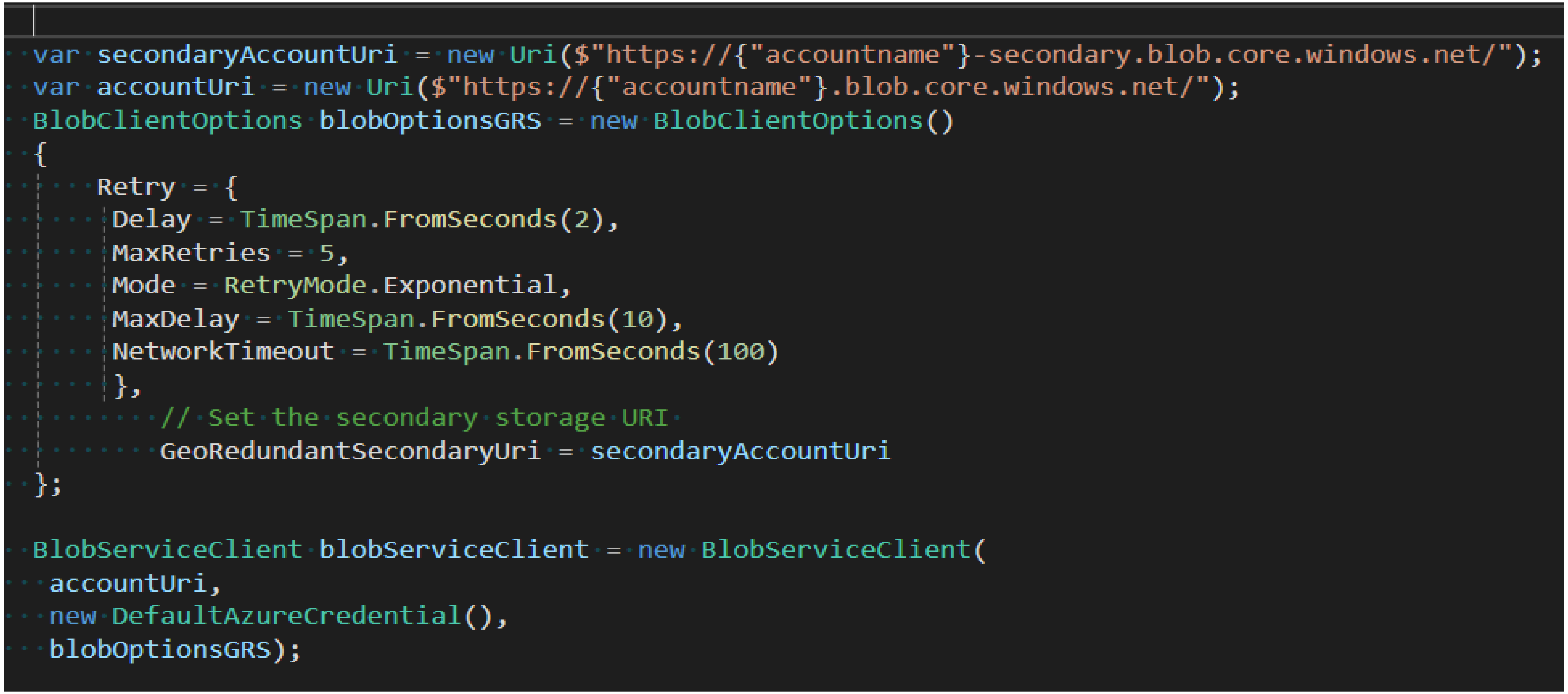
MaxDelay

MaxRetries

Mode

Timeout

• RetryOptions class - Implementation



```
var secondaryAccountUri = new Uri($"https://{"accountname"}-secondary.blob.core.windows.net/");
var accountUri = new Uri($"https://{"accountname"}.blob.core.windows.net/");
BlobClientOptions blobOptionsGRS = new BlobClientOptions()
{
    Retry = {
     Delay = TimeSpan.FromSeconds(2),
     MaxRetries = 5,
     Mode = RetryMode.Exponential,
     MaxDelay = TimeSpan.FromSeconds(10),
     NetworkTimeout = TimeSpan.FromSeconds(100)
     },
        // Set the secondary storage URI
        GeoRedundantSecondaryUri = secondaryAccountUri
};

BlobServiceClient blobServiceClient = new BlobServiceClient(
  accountUri,
  new DefaultAzureCredential(),
  blobOptionsGRS);
```

Presenter: Mai Hoang Giang