Azure Data Engineer

Azure Basics

**1. Azure hierarchy**

**1. Management Groups (The Top-Level)**

* **What it is**: This is the **highest level** of Azure’s structure. It's like a **parent folder** that helps organize your entire Azure environment.
* **Purpose**: It helps you **manage and organize multiple subscriptions** across your organization. You can group subscriptions based on departments, teams, or projects.
* **Example**: Imagine a company with multiple divisions, like Finance, IT, and Marketing. You could have separate management groups for each of these divisions.

**2. Subscriptions (Middle Layer)**

* **What it is**: Underneath management groups, you have **subscriptions**. A **subscription** is like a **container** for Azure resources, where you’ll define limits on resources and billing.
* **Purpose**: It helps with organizing resources, managing access, and billing. You can have multiple subscriptions for different projects or teams. Each subscription has its own **resource limits** and **billing**.
* **Example**: If your company has different projects, like a website and an app, you could create separate subscriptions for them. One for the website, one for the app.

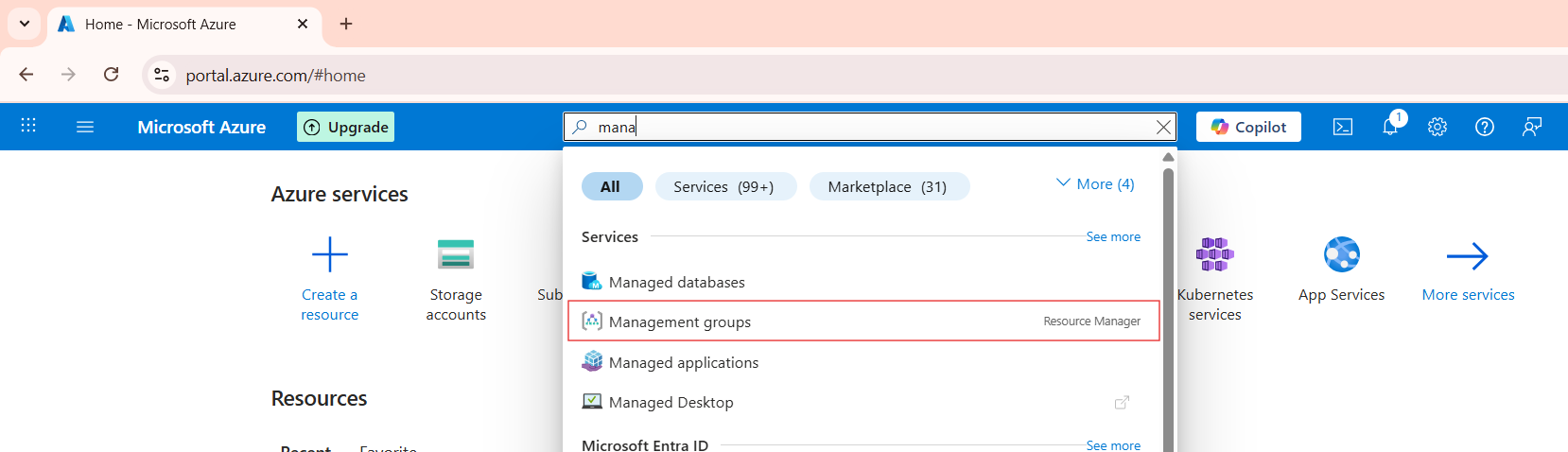
**3. Resource Groups (Sub-Containers)**

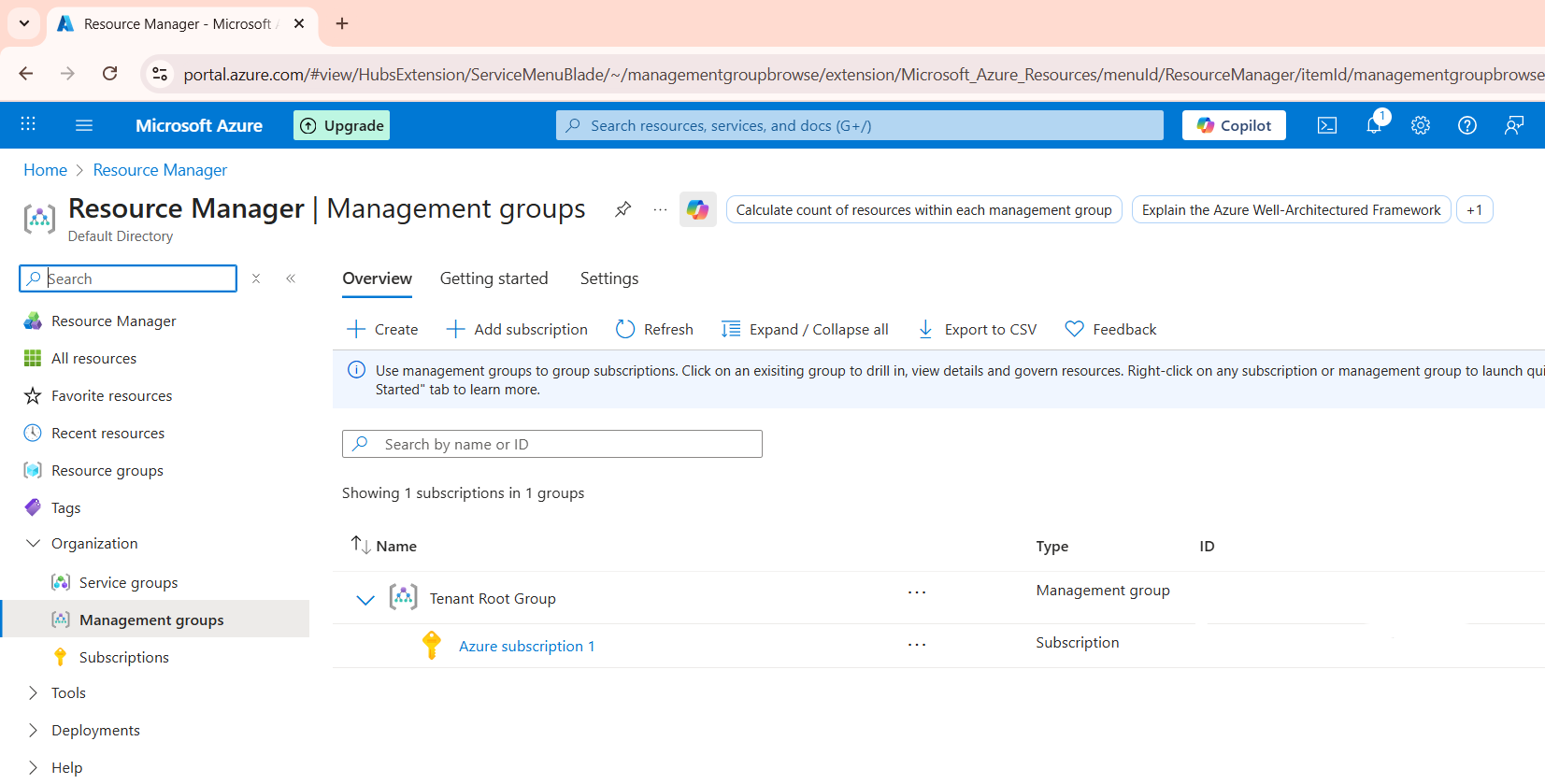
* **What it is**: Inside each subscription, you can have **resource groups**. These are containers that hold related resources.
* **Purpose**: They help organize resources based on their lifecycle and permissions. All the resources in a group are usually related to the same project or service.
* **Example**: If you're building a web app, you might have a resource group called "rg-ohg365-dev" where you store everything related to the app, such as databases, storage accounts, and virtual machines.

**4. Resources (The Actual Items)**

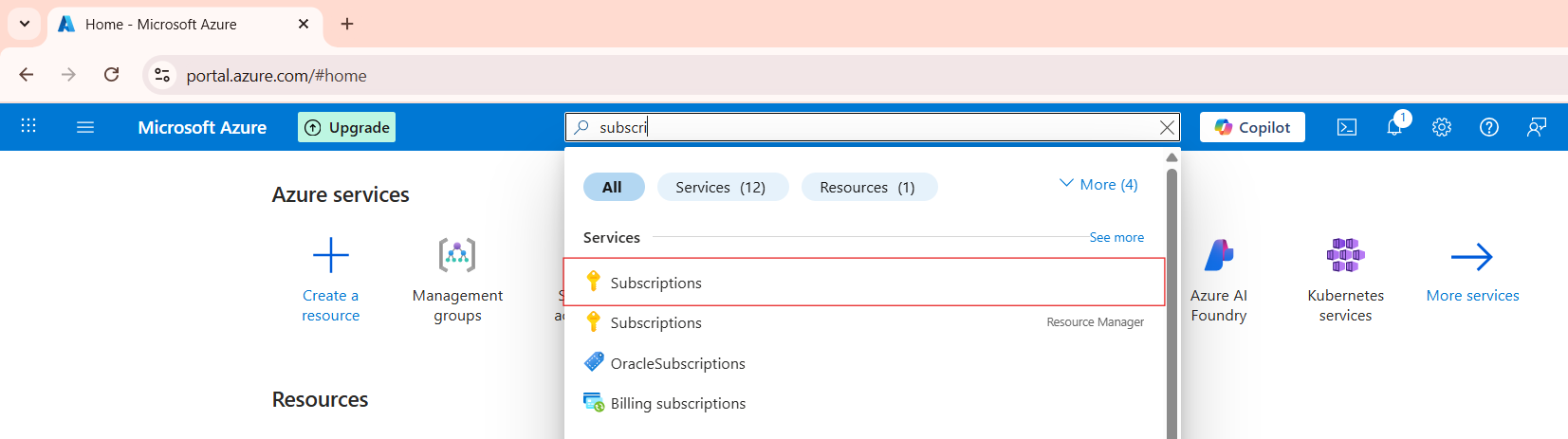
* **What it is**: These are the **individual services** or **products** that you create in Azure, like **virtual machines (VMs)**, **storage accounts**, **databases**, or **networks**.
* **Purpose**: This is where the actual work happens! Resources are the **building blocks** of your cloud environment.
* **Example**: In your "rg-ohg365-dev" resource group, you could have resources like a VM to run your website, a database to store your data, and a storage account for storing files.

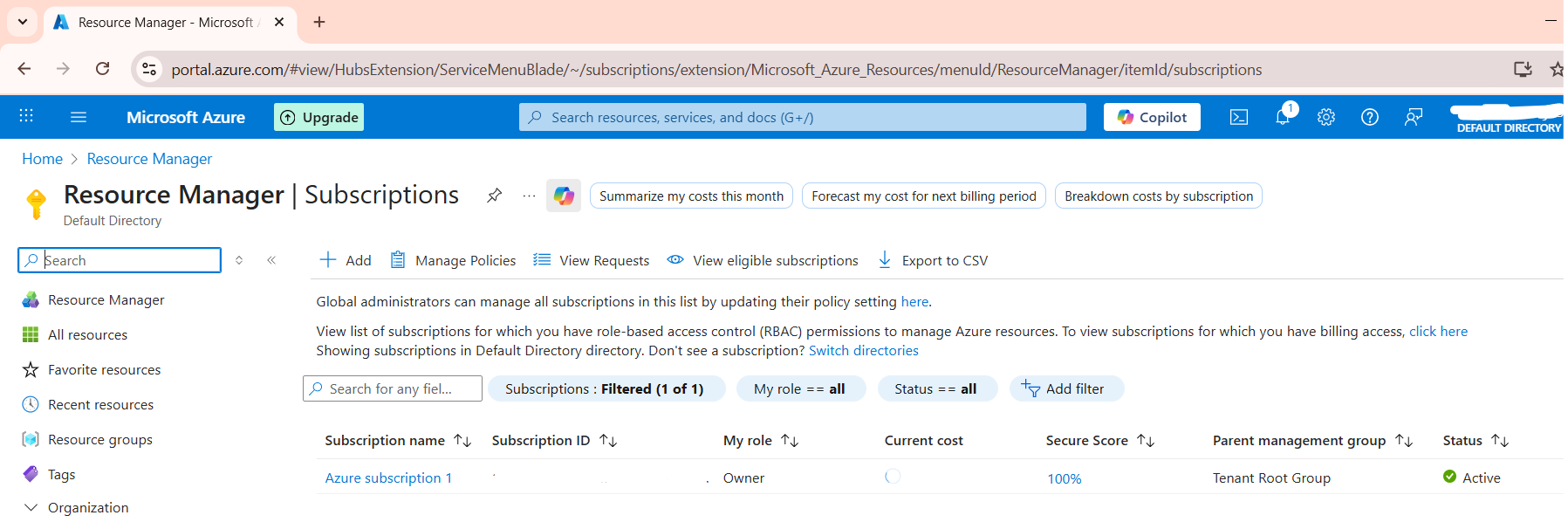
**Management Groups**: High-level containers for organizing multiple subscriptions.



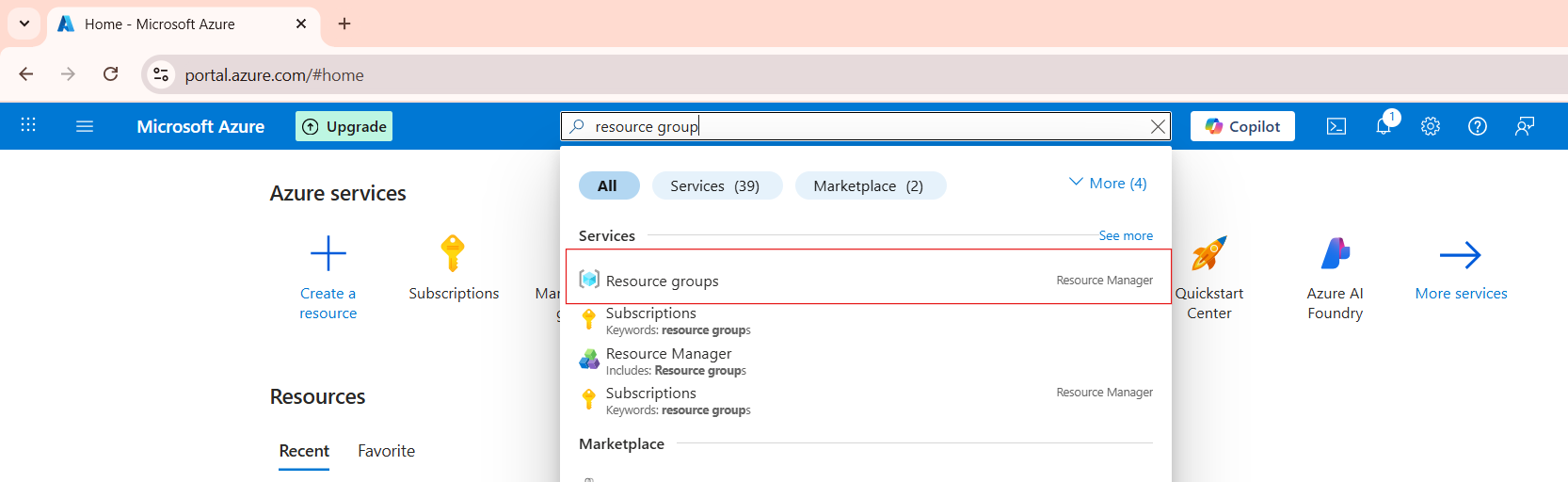


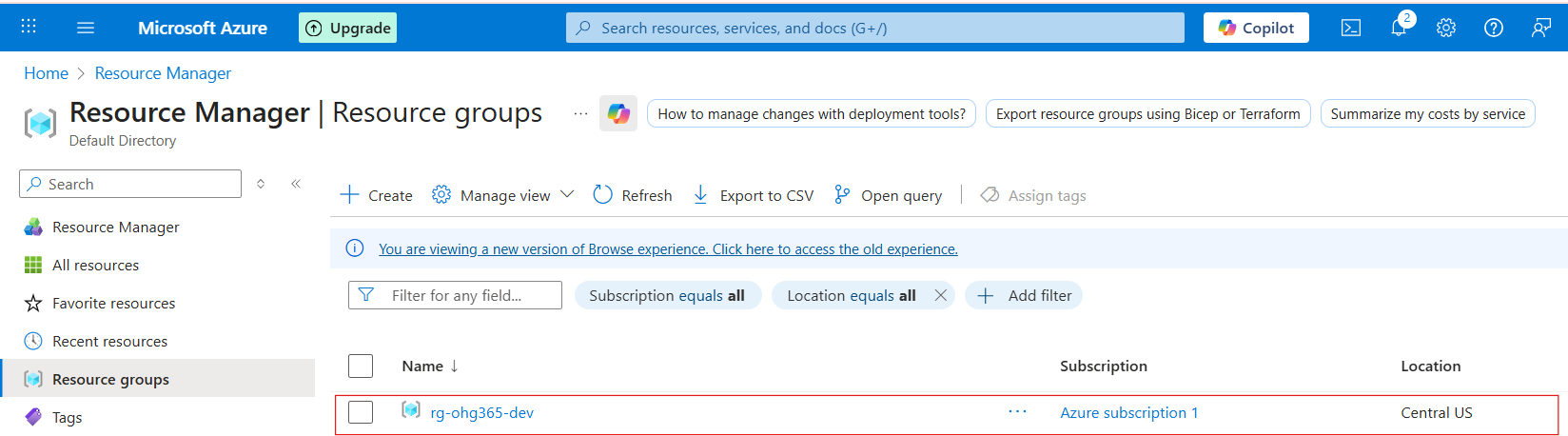
**Subscriptions**: They hold resources and manage access to resources.





**Resource Groups**: Containers inside subscriptions to organize and manage resources by project or lifecycle.



****

**Resources**: The actual services you use in Azure.

**2. Resource Group**

**2.1 What is a Resource Group?**

A **Resource Group** in Azure is like a **folder** that holds all the resources (services) related to a project or app.

For example:  
If you build a website, you might have:

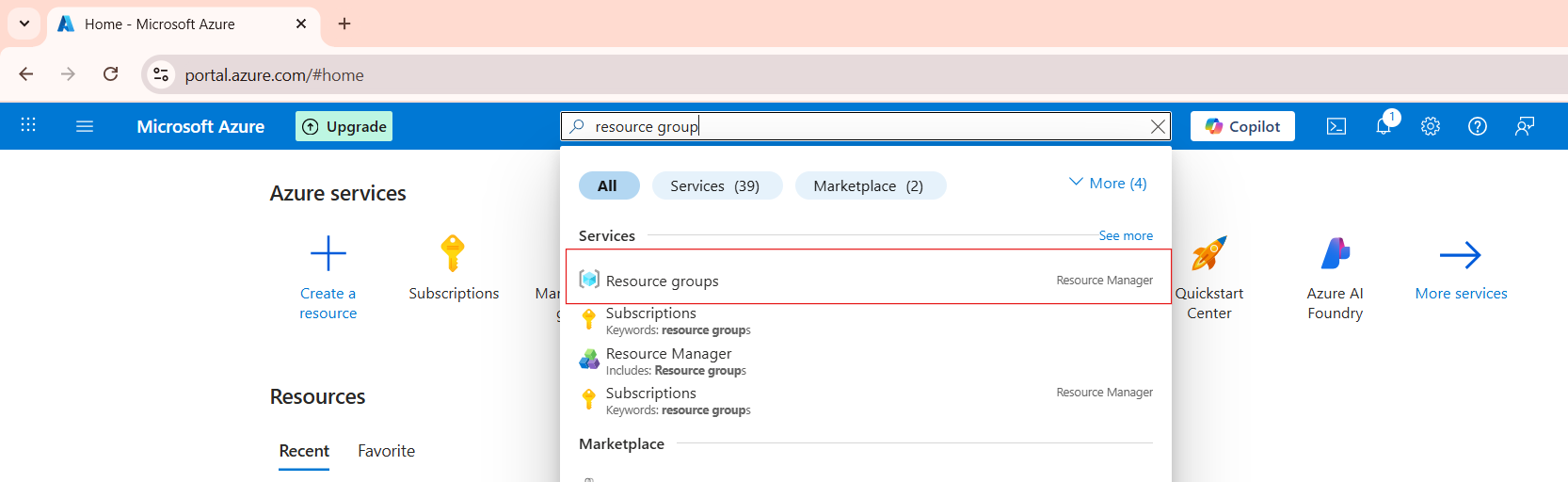
* A **Virtual Machine (VM)** for the web server
* A **Storage Account** for images
* A **Database** for user data

You can put all of these inside **one Resource Group** — making it easier to manage, monitor, and delete them together.

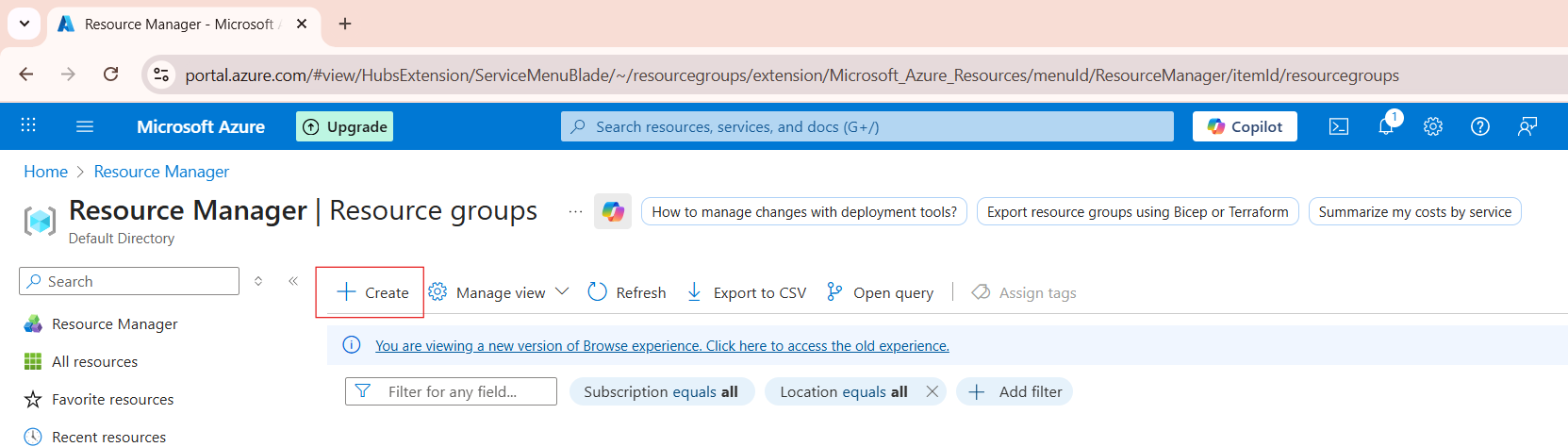
**2.2 Create a Resource Group**

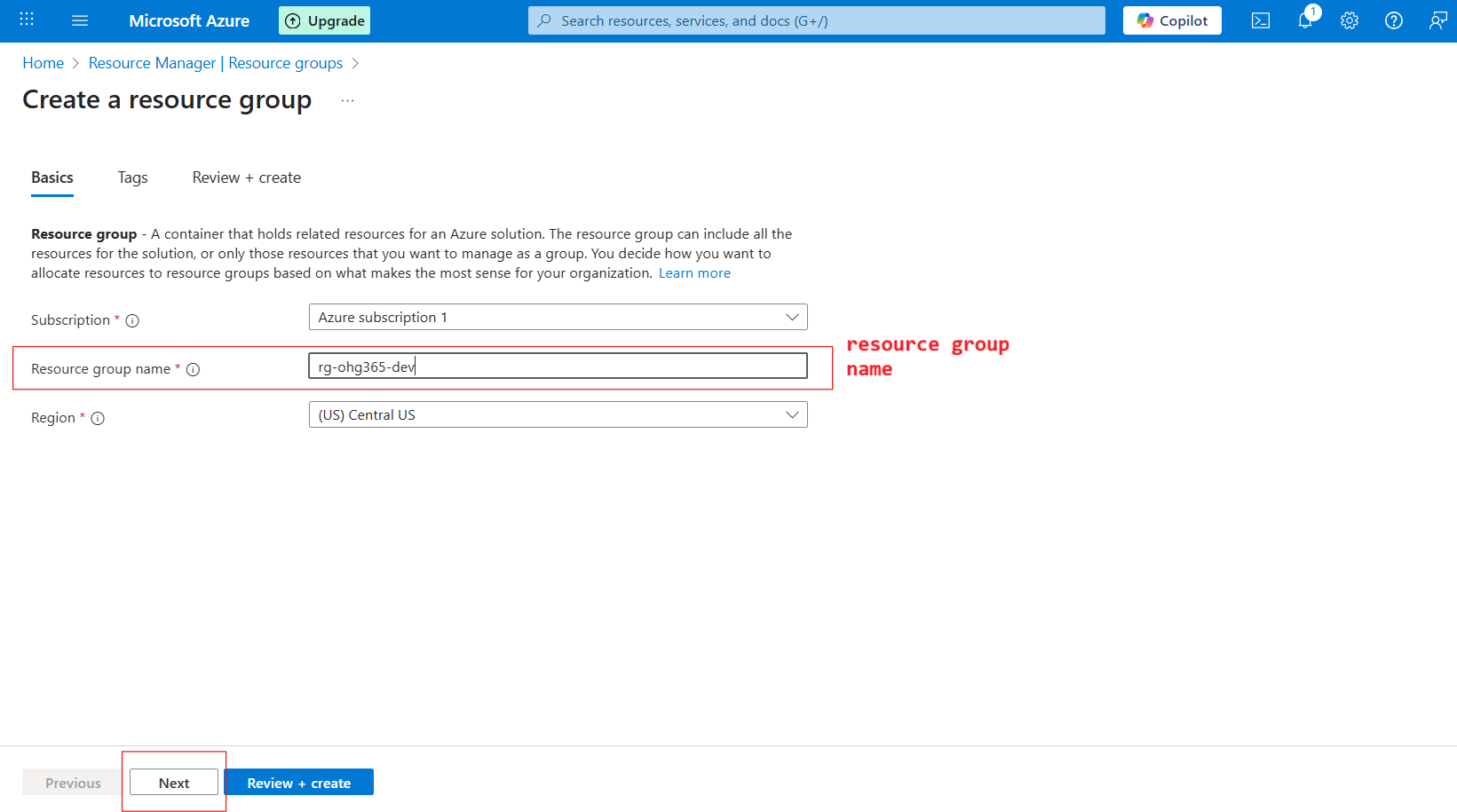
**Sign in to the Azure Portal**

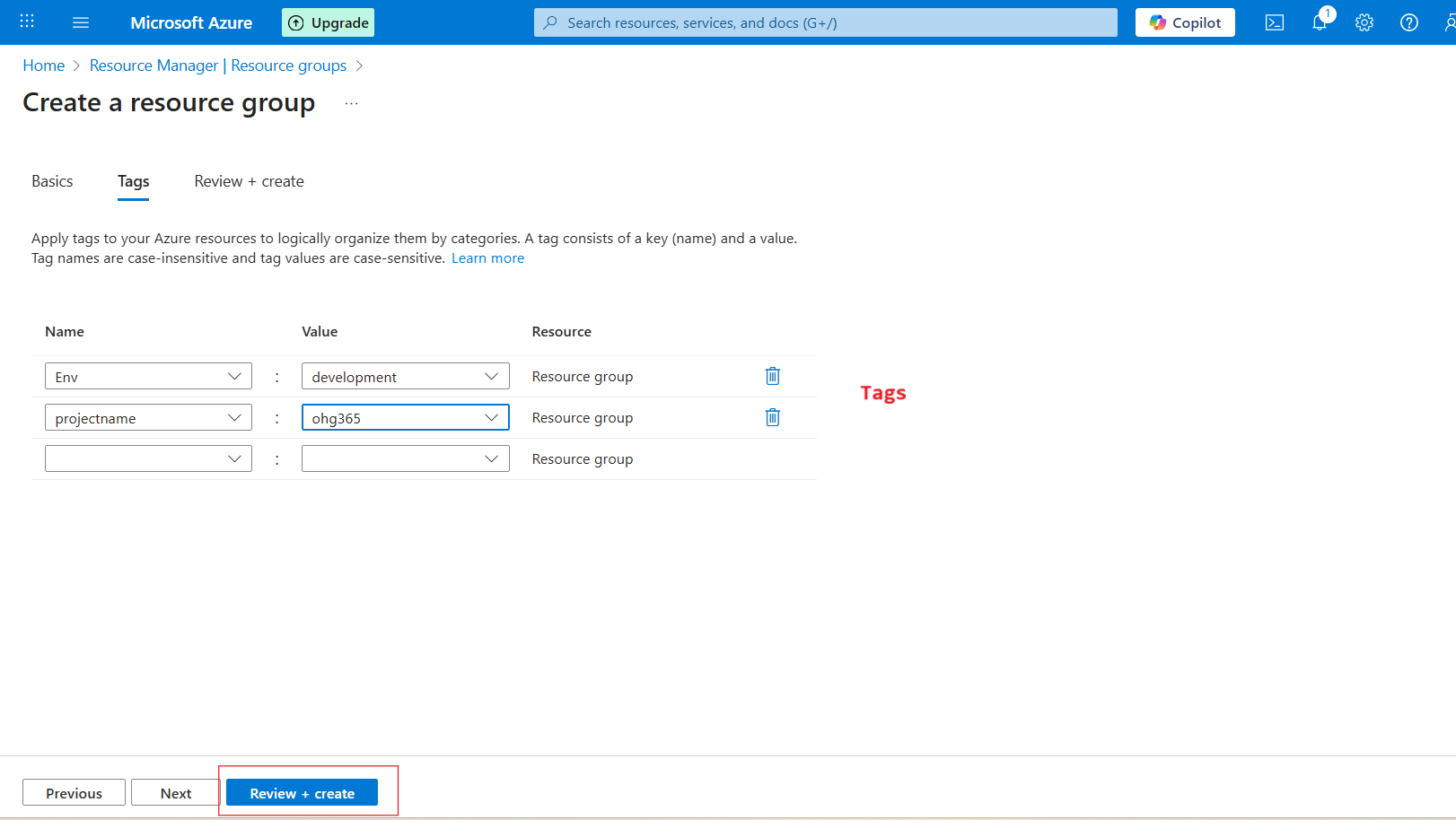
**In the search menu, search for Resource groups.**

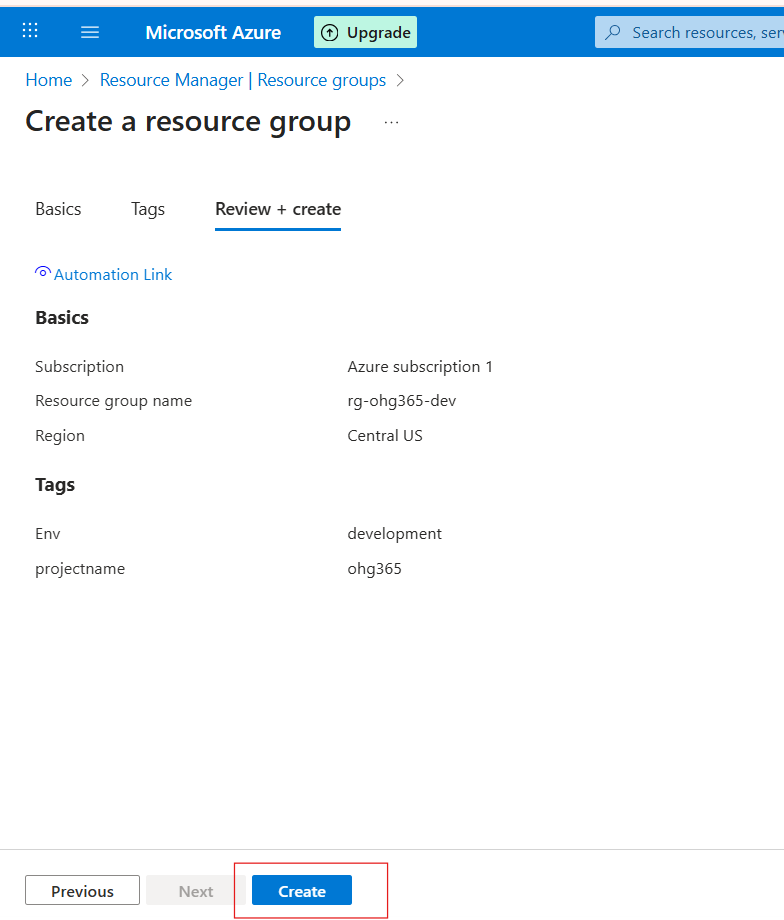


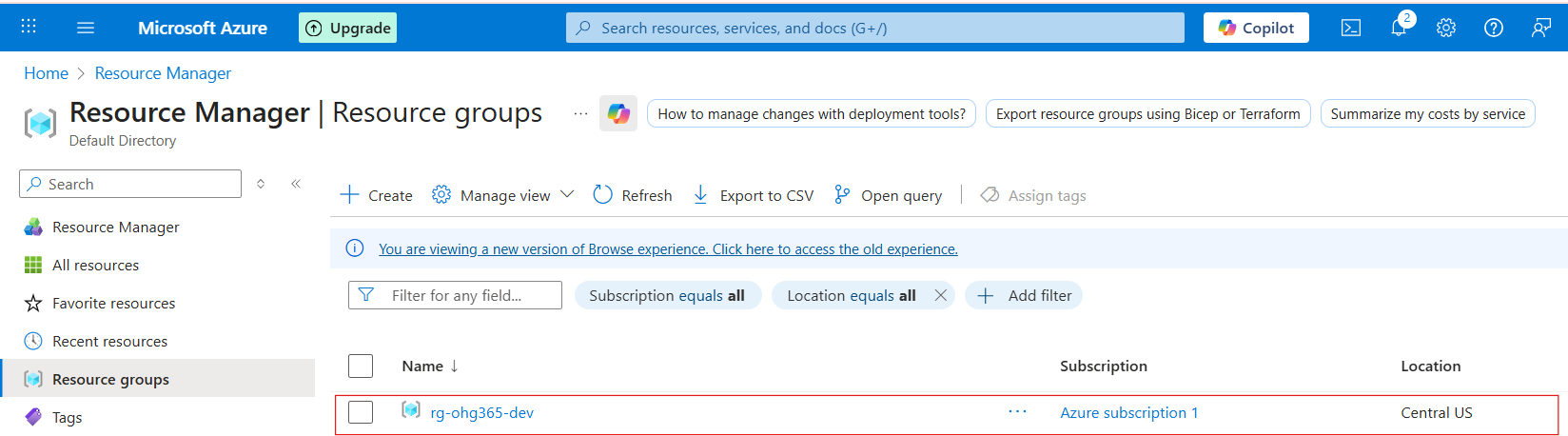
**Click on the resource group and click on the create button**

****

****

****

****

****

**Overview of resource group:**

**Name: rg-ohg365-dev → This is your Resource Group’s name.  
Usually, names include clues about the project or environment:**

* **rg = Resource Group**
* **ohg365 = Project or team name**
* **dev = Environment (like dev, test, or prod)**

**Buttons:**

* **➕ Create: Add new Azure resources (like VMs, storage, databases).**
* **⚙️ Manage view: Customize how your resources list looks.**
* **🗑️ Delete resource group: Deletes the entire group and all its resources (be careful!).**
* **🔁 Refresh: Updates the view if new resources were added.**
* **📤 Export to CSV: Export your resource details (helpful for reports).**
* **📊 Open query: Create or run custom filters using Azure Resource Graph.**

**3. Azure Blob Storage**

Azure Blob Storage is Microsoft’s cloud-based service designed to store large amounts of data of various types — including structured, semi-structured, and unstructured data.

It is ideal for storing files such as CSV, text, Excel, JSON, Parquet, Avro, XML, images, videos, backups, and logs.

The term “Blob” stands for Binary Large Object, meaning it can store any type of binary data. Blob Storage provides a flat namespace, meaning all files (blobs) are stored in containers within a storage account, rather than in a traditional hierarchical folder system.

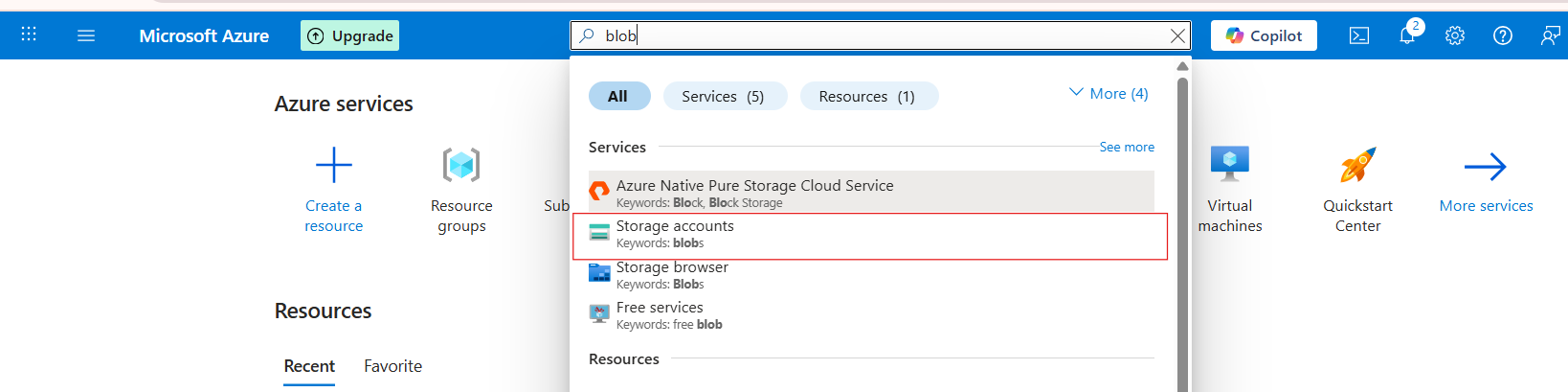
Azure Blob Storage is highly scalable, secure, and cost-efficient, making it suitable for:

* Storing and serving large files (media, documents, etc.)
* Data lakes and analytics workloads
* Backup and disaster recovery
* Archiving and compliance storage

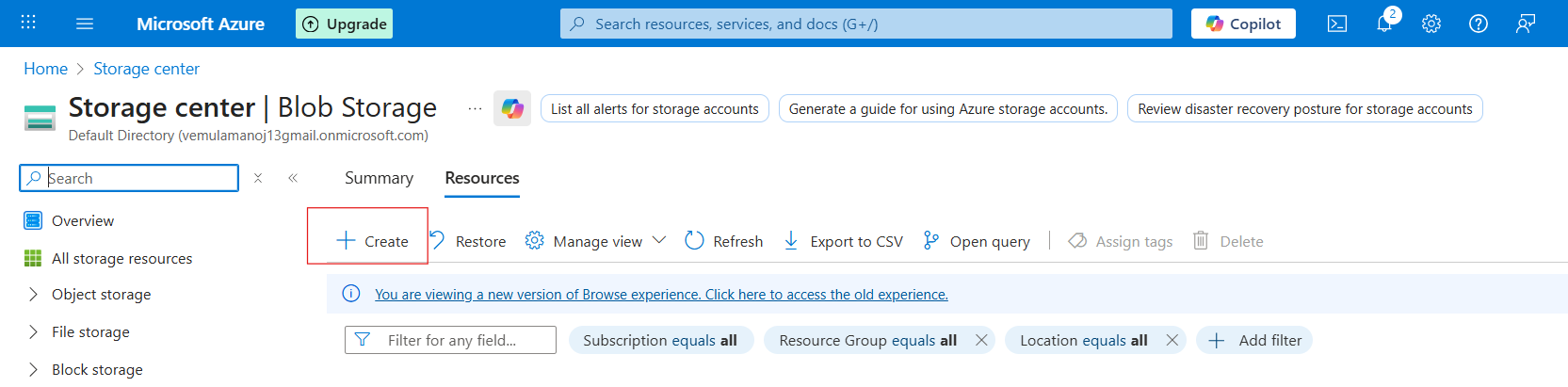
**3.1 Create an Azure Blob Storage:**

**Go to Azure Portal**

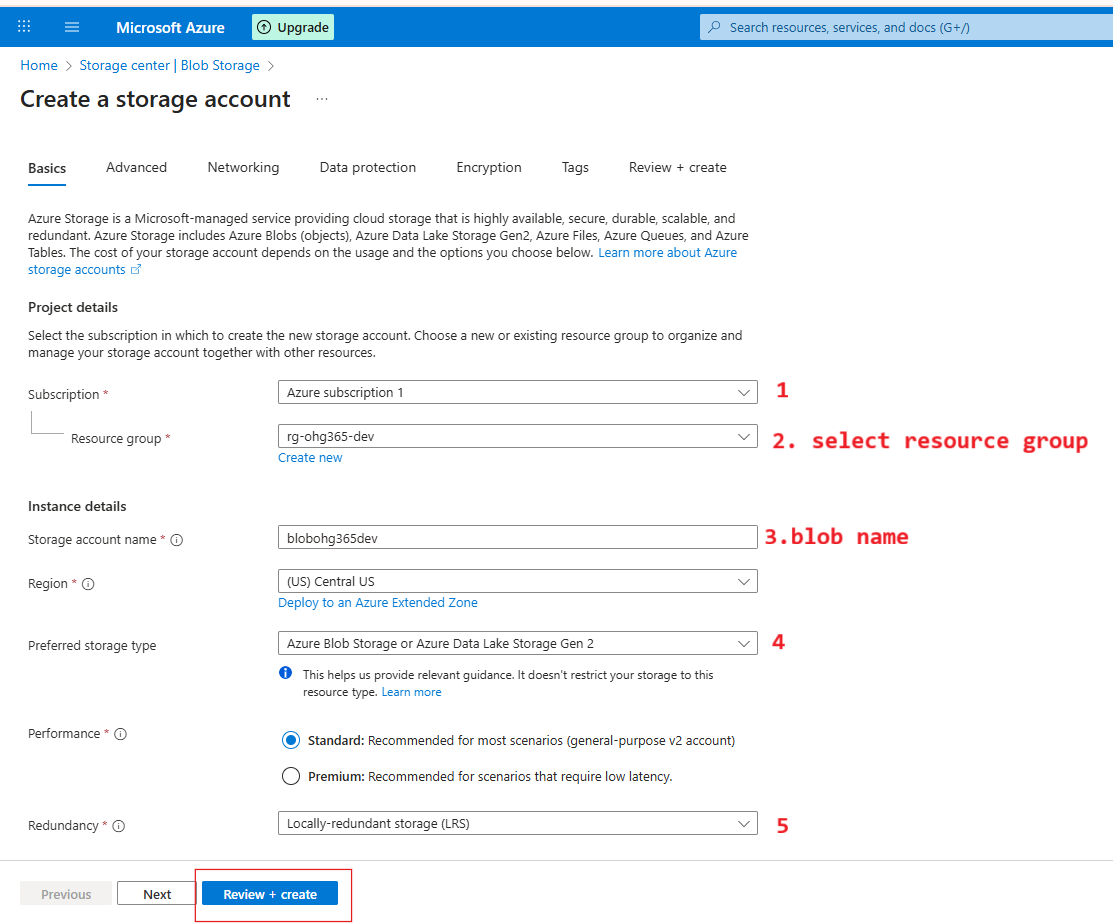
**In the search bar, type “Storage Accounts” or “blob”**

****

**Click on storage accounts and click on create button**

****

**Click “Review + Create”**

****

**1️⃣ Subscription**

* **This is where you choose which Azure Subscription will own this storage account.**
* **A subscription is linked to your billing and access control.**
* **Example: You might have separate subscriptions for development, testing, or production environments.  
  🟦 You selected: *Azure subscription 1***

**2️⃣ Resource Group**

* **Choose or create a Resource Group to organize related Azure resources.**
* **Resource Groups act like folders — all your related resources (VMs, storage, databases) are stored here for easy management.**
* **In your case, you selected rg-ohg365-dev, which is perfect for development resources.  
  🟦 Tip: Keeping related resources in the same group helps you track cost, permissions, and manage everything easily.**

**3️⃣ Storage Account Name**

* **This is the unique name for your storage account (like a domain name).**
* **It must be globally unique, lowercase, and 3–24 characters long.**
* **This name will form part of the URL to access your data.**

**🧩 Example:  
If your name is blobohg365dev**

**4️⃣ Preferred Storage Type**

* **Select what kind of storage service you want to enable.**
* **The default (and most common) option is:  
  Azure Blob Storage or Azure Data Lake Storage Gen2**

**🧠 This means your account will support:**

* **Blob storage (for files, media, etc.)**
* **Data Lake capabilities (for analytics and big data processing)**

**🟦 Tip: Keep this as default unless you have a specific need for file shares or queue services.**

**5️⃣ Performance & Redundancy Settings**

**⚙️ Performance:**

* **Standard: Uses HDD-based storage — cheaper, good for general use.**
* **Premium: Uses SSD-based storage — faster, ideal for workloads needing low latency (like databases or VMs).**

**🟦 You selected: *Standard (recommended)***

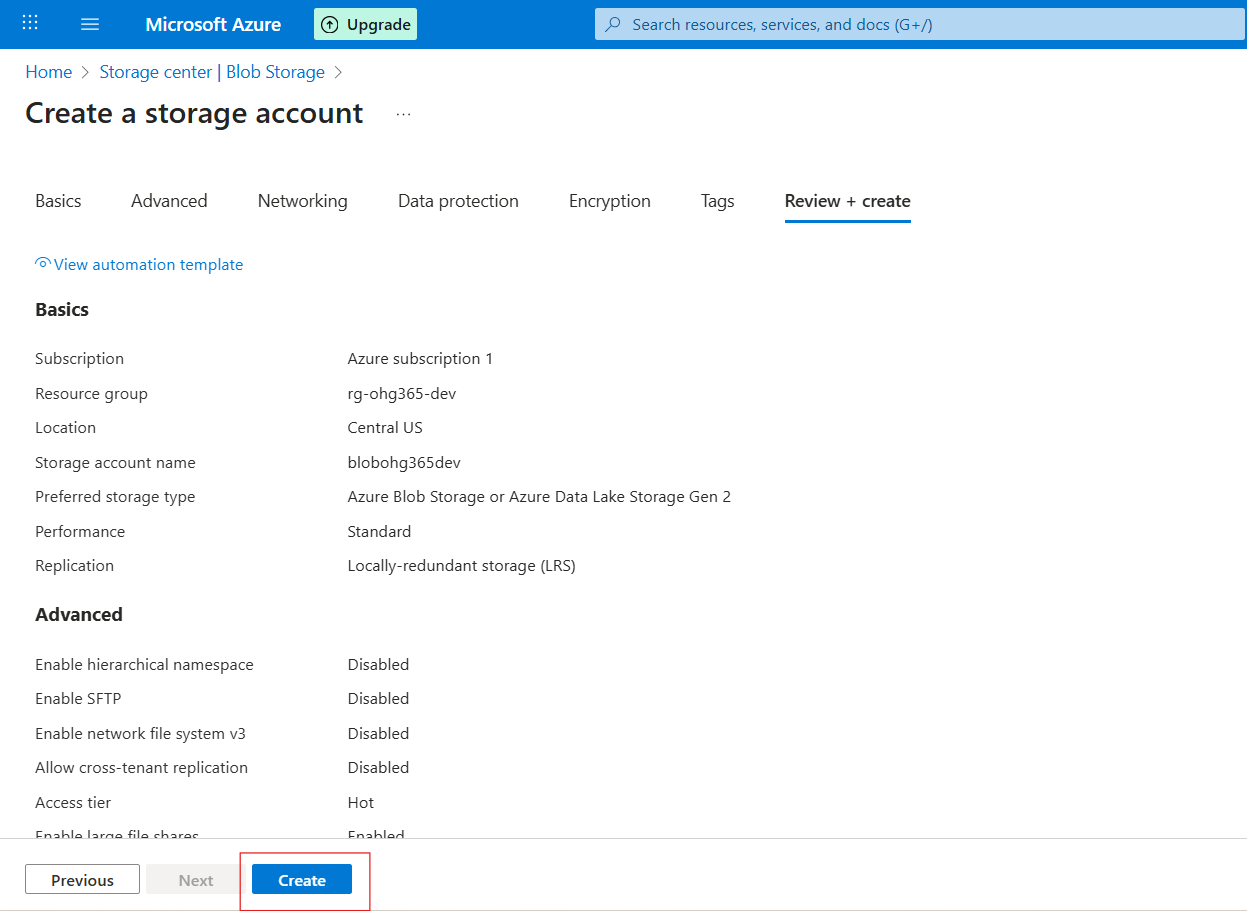
**🧭 Redundancy:**

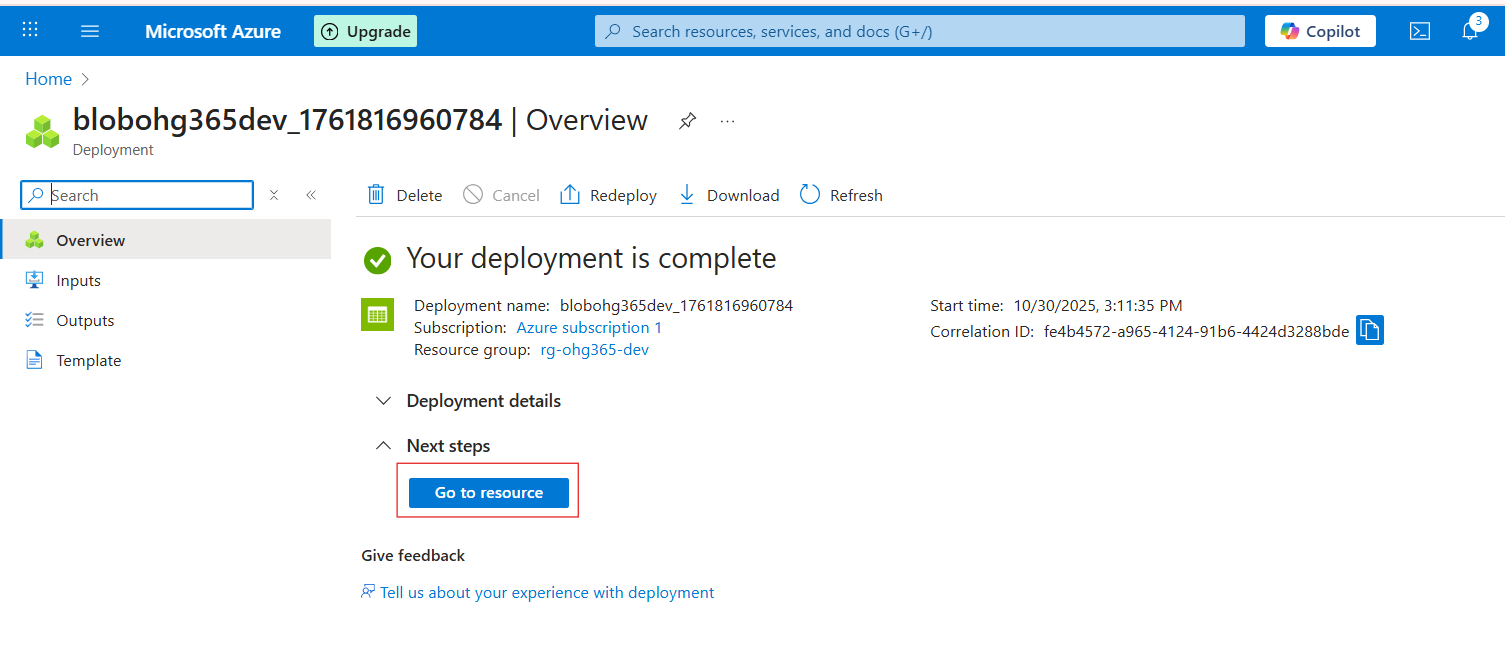
**Defines how Azure will replicate your data to keep it safe.**

| **Option** | **Meaning** | **Copies of Data** |
| --- | --- | --- |
| **LRS (Locally-redundant storage)** | **Keeps 3 copies in one data center** | **3** |
| **ZRS (Zone-redundant storage)** | **Copies across 3 availability zones in the same region** | **3** |
| **GRS (Geo-redundant storage)** | **Copies data to another region (for disaster recovery)** | **6** |
| **RA-GRS (Read-access Geo-redundant)** | **Same as GRS but allows read access to secondary region** | **6** |

**🟦 You selected: *LRS* (best for development/testing)**

**Click on create**





**Navigate to Your Resource Group**

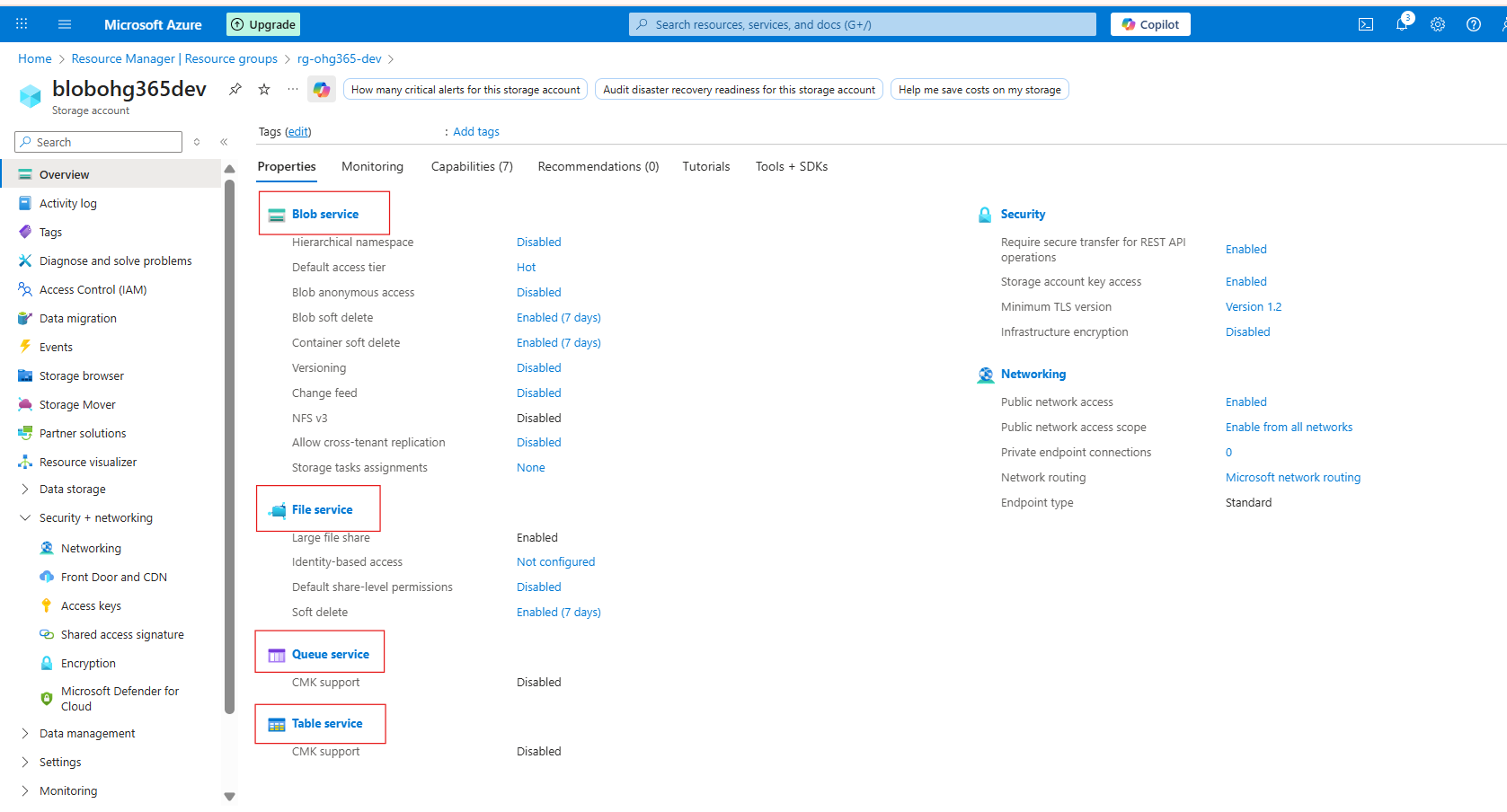
* In the left-hand menu, click on **Resource groups**
* Find and click your resource group — in your case:  
  👉 **rg-ohg365-dev**
* **Check for the Storage Account**
* Inside the **Overview** tab of your resource group, you’ll see a list of all resources.
* Look for an item that looks like this:  
  **Type:** *Storage account*  
  **Name:** *blobohg365dev* (or whatever name you used)

If it appears there, 🎉 congratulations — your **Blob Storage account** has been successfully created



**3.2 Types of Azure Storage Services**:

Azure Storage provides **four main types of services** under one **Storage Account**.



**1️⃣ Blob Service**

🧱 *Used to store unstructured or semi-structured data like files, images, videos, logs, and backups.*

**🔹 Description:**

* Stores data as **Blobs (Binary Large Objects)** inside **containers**.
* Best for storing **flat files** and **large objects** that don’t fit in a database.
* Data can be text, binary, documents, media, or backups.

**💾 Example Uses:**

* Storing images or videos for websites
* Backups and archives
* Data lake for analytics
* Hosting static websites

**📦 Example File Types:**

.txt, .csv, .json, .xml, .jpg, .mp4, .zip, .bak

**2️⃣ File Service**

📁 *Used for shared file storage that behaves like a traditional file server.*

**🔹 Description:**

* Provides **Azure Files**, a fully managed **file share** in the cloud.
* Uses the **SMB (Server Message Block)** or **NFS (Network File System)** protocols — the same used by on-premises file servers.
* Can be **mounted** to Windows, Linux, or macOS systems.

**💾 Example Uses:**

* Shared network drives for teams
* “Lift and shift” of on-premises file servers
* Application configurations shared across multiple VMs

**📦 Example Scenario:**

You have multiple virtual machines needing access to the same configuration files — you can store those files in **Azure Files** and mount them just like a shared folder.

**3️⃣ Queue Service**

📬 *Used for reliable messaging between application components.*

**🔹 Description:**

* Provides **asynchronous communication** between services using message queues.
* Stores messages in a **queue**, which can be processed later by background services or workers.
* Ensures messages are **delivered at least once** and processed in **FIFO (First-In, First-Out)** order.

**💾 Example Uses:**

* Sending background jobs (like image processing or email sending)
* Decoupling app components for scalability
* Event-driven architecture

**📦 Example Scenario:**

A web app uploads an image → sends a message to a **queue** → a background process picks it up and resizes the image.

**4️⃣ Table Service**

🧮 *Used to store large amounts of structured, non-relational data.*

**🔹 Description:**

* Provides **NoSQL key-value storage**.
* Stores data in **tables** with **entities (rows)** and **properties (columns)**.
* Flexible schema — you can add or remove columns anytime.

**💾 Example Uses:**

* Storing user profiles, IoT data, or metadata
* Fast lookups by key
* Lightweight applications needing scalable, cheap storage

**📦 Example Scenario:**

You have millions of IoT sensors sending temperature data — you can store this efficiently in **Azure Table Storage**.

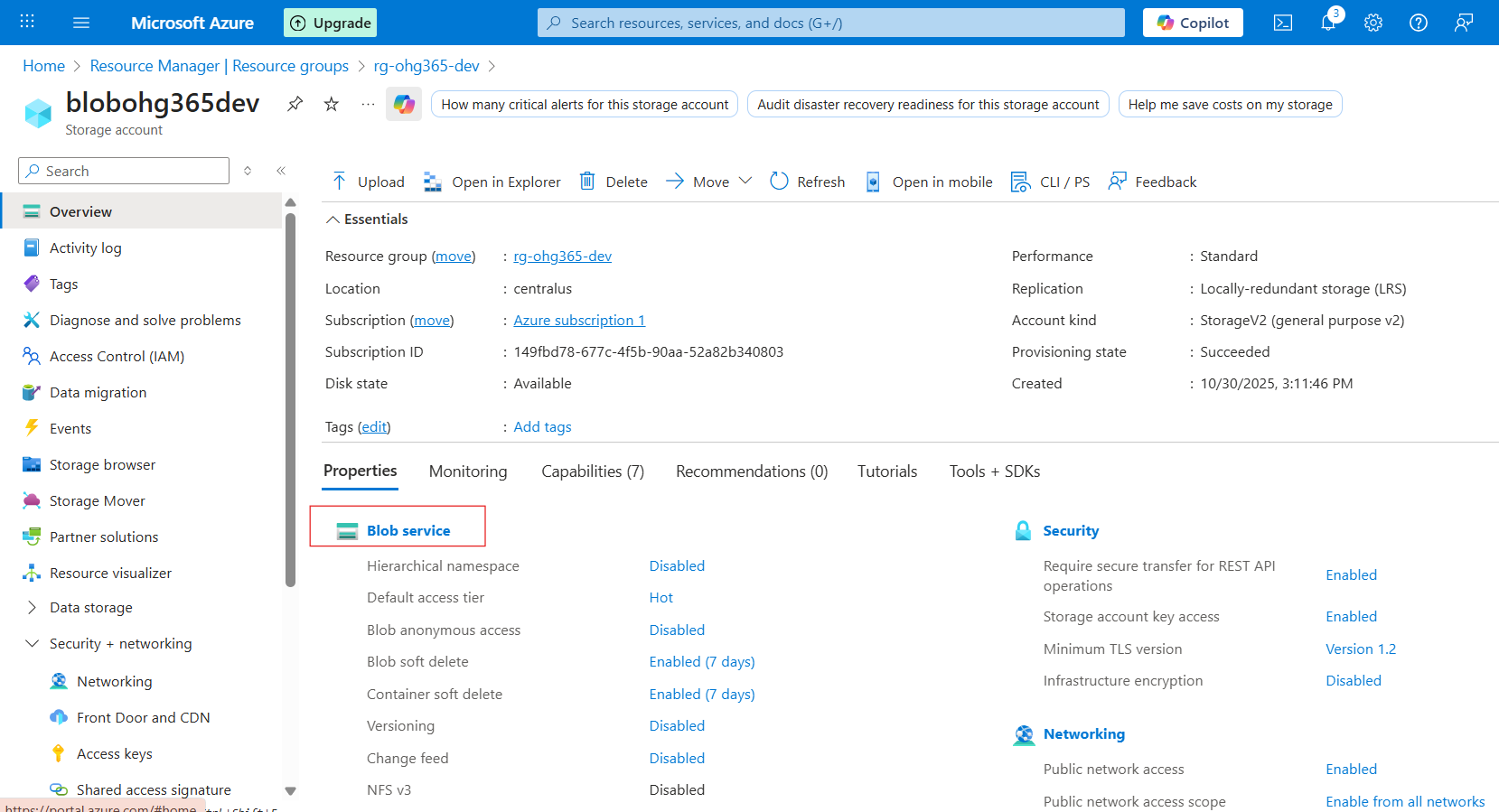
**🧠 Summary Table**

| **Storage Service** | **Type of Data** | **Description** | **Example Use Case** |
| --- | --- | --- | --- |
| **Blob Service** | Unstructured / Semi-structured | Stores large objects (files, media, backups) | Images, videos, logs |
| **File Service** | File-based | Shared file storage via SMB/NFS | Shared drives, app configs |
| **Queue Service** | Messaging | Message-based communication between components | Background tasks, event processing |
| **Table Service** | Structured (NoSQL) | Key-value, schema-less table storage | User profiles, IoT data |

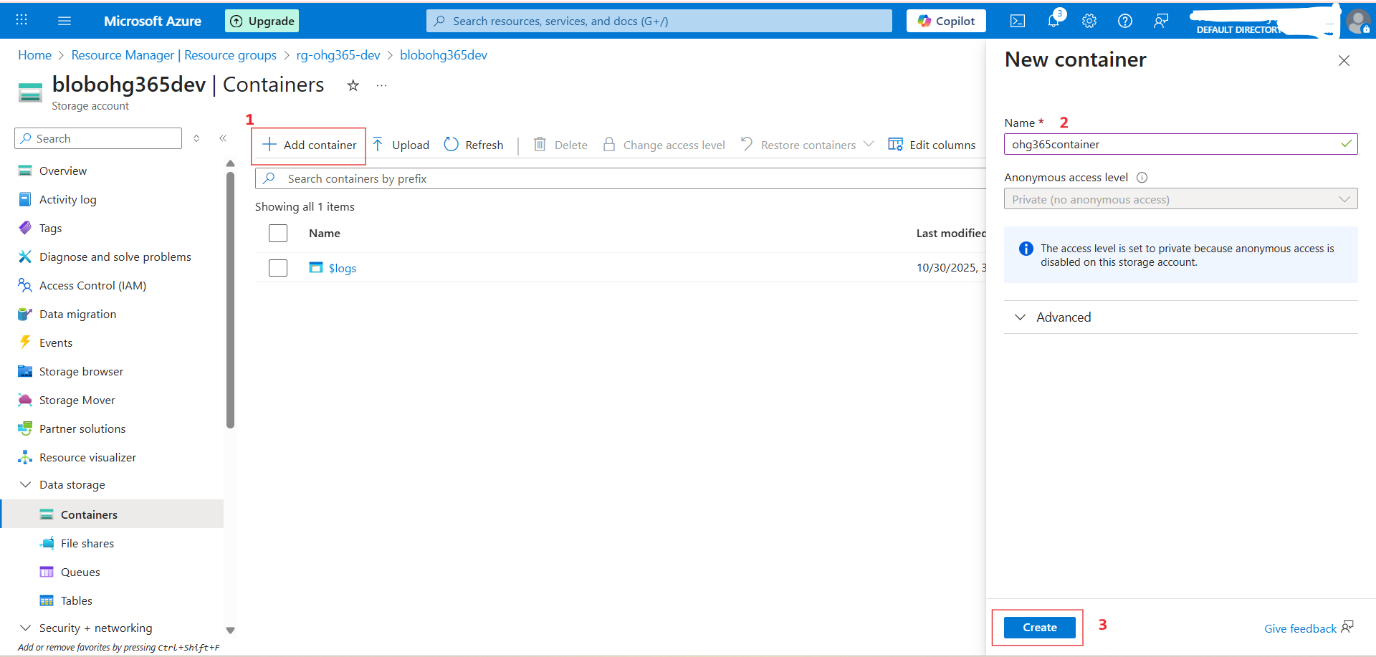
**Explore Blob Service**

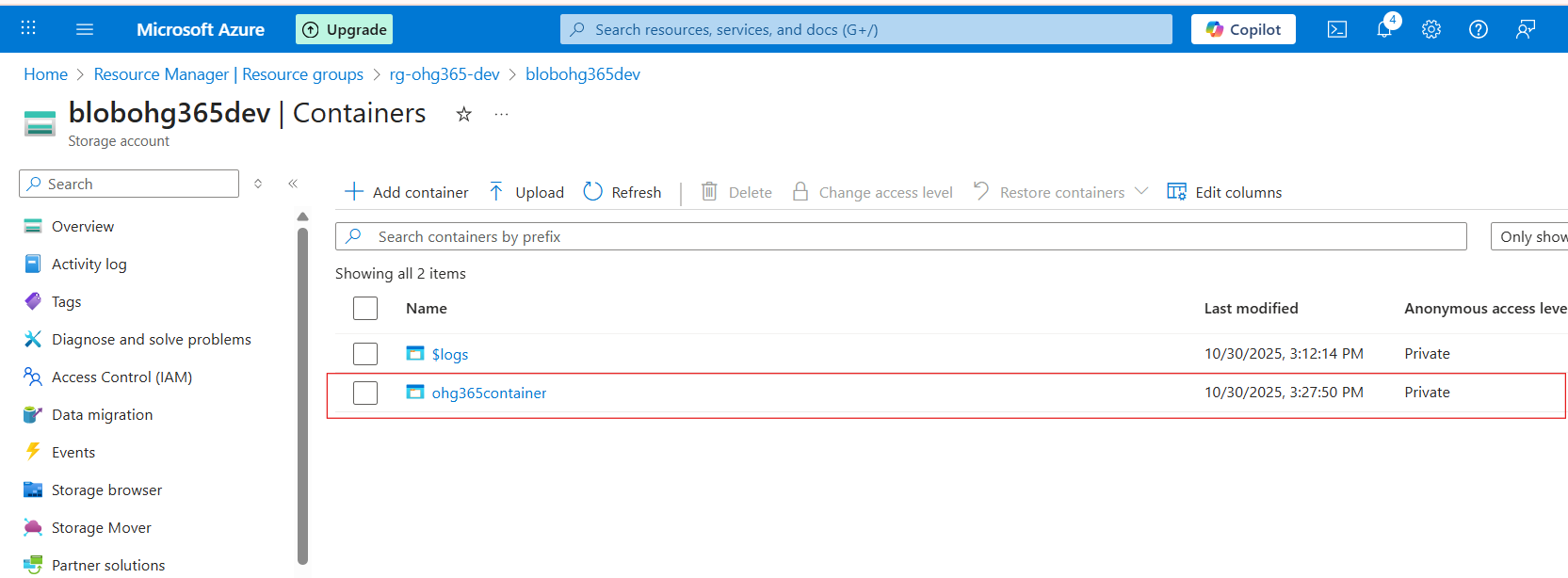
Once your storage account is created:

1. Go to your **Storage Account**



1. Under **Data storage**, click **Containers** → This is where your blobs live.
2. Click **➕ Container** to create one:
   * Name: images, videos, or backups (any name)
   * **Public access level:**
     + *Private (default)* – Only you can access
     + *Blob (anonymous read)* – Anyone with the link can read blobs
     + *Container (public)* – Everyone can see contents
   * Click **Create**





**Upload & Manage Blobs**

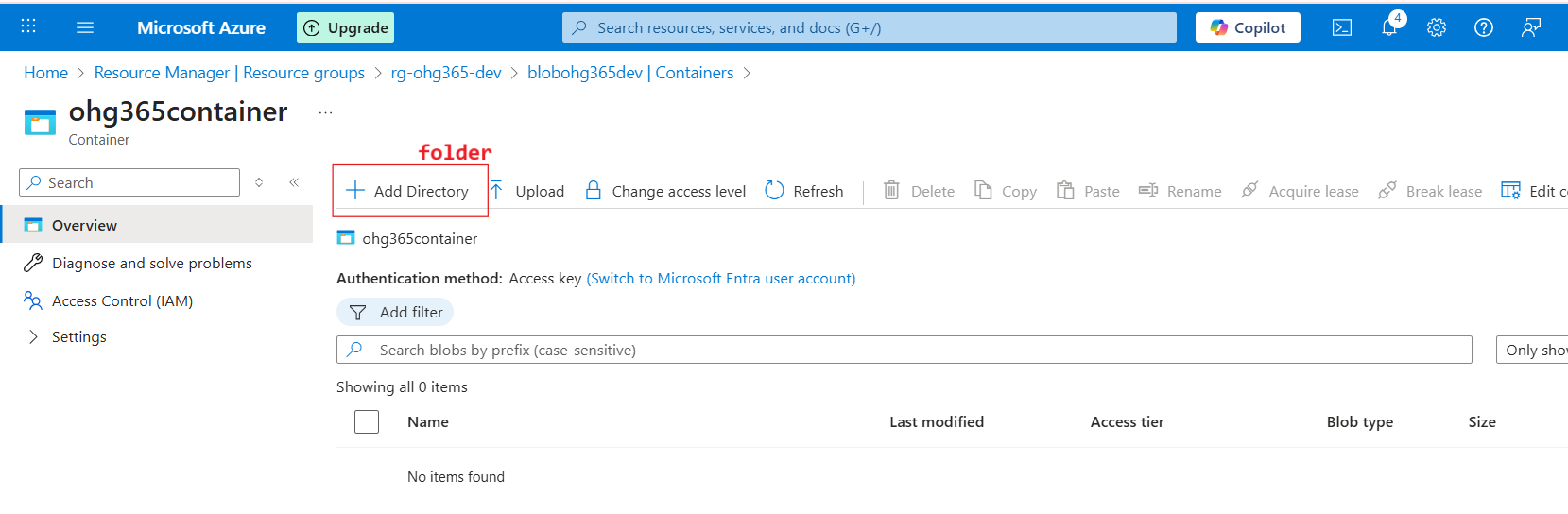
1. Click your new container (e.g., images)
2. Click **Upload**
3. Choose a file from your computer (like a .jpg, .txt, or .mp4)
4. Once uploaded, you can:
   * View **Properties** (size, type, last modified)
   * Get the **URL** to access the file
   * Change the **access tier**

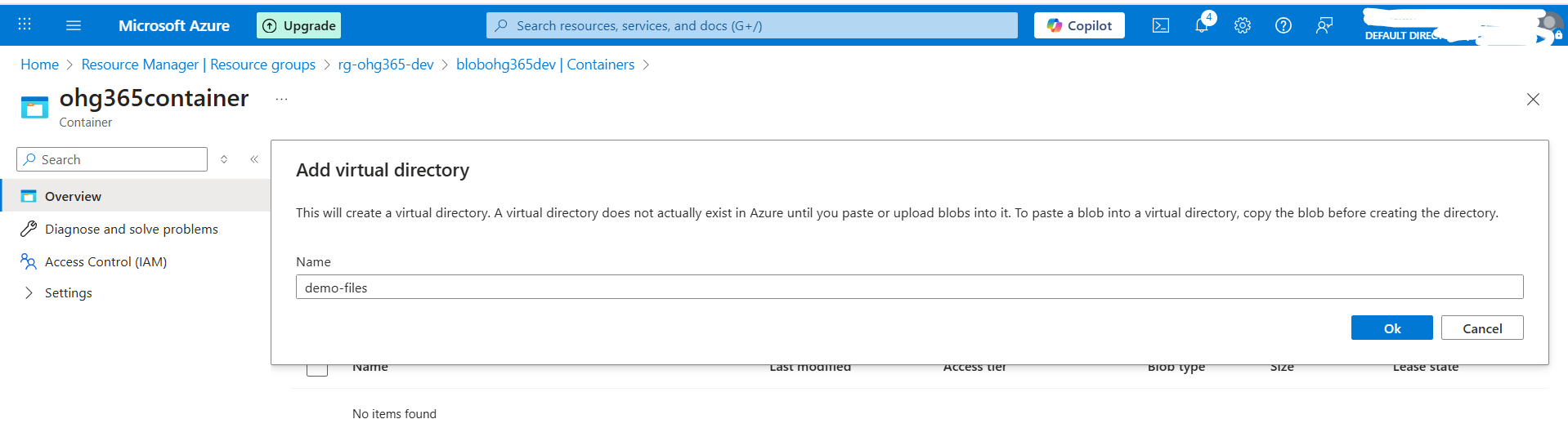
Storage Account

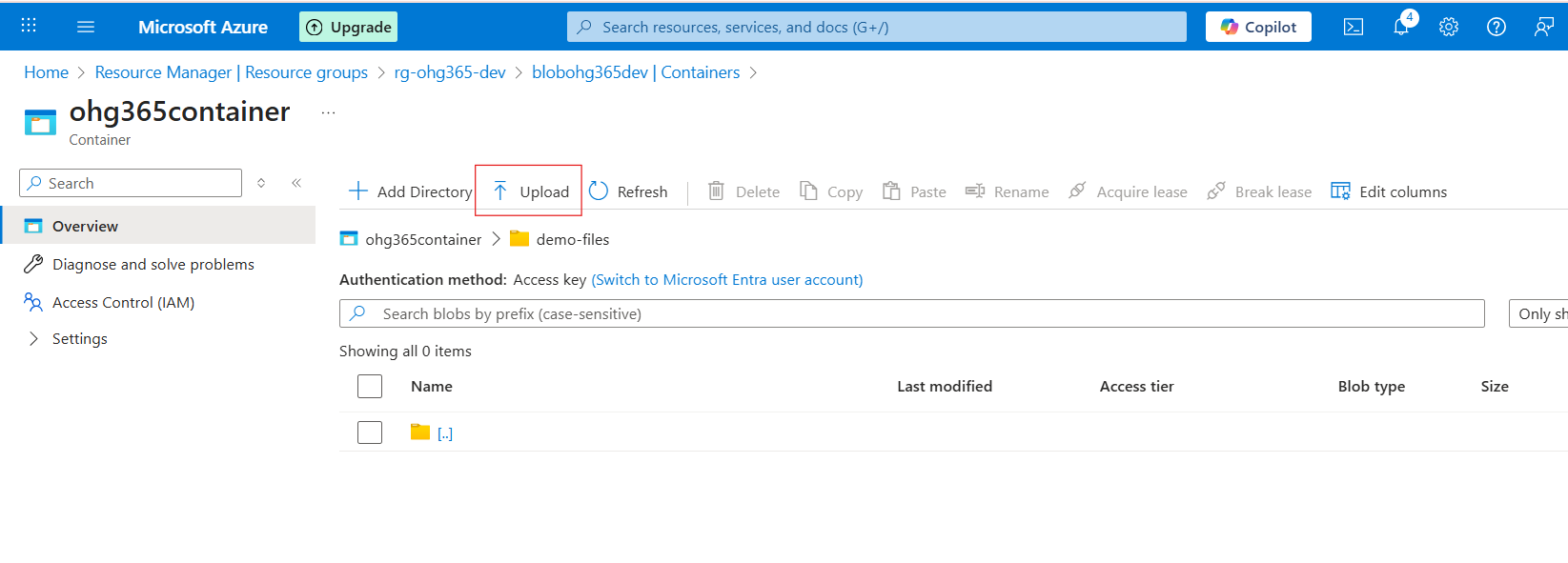
└── Blob Service

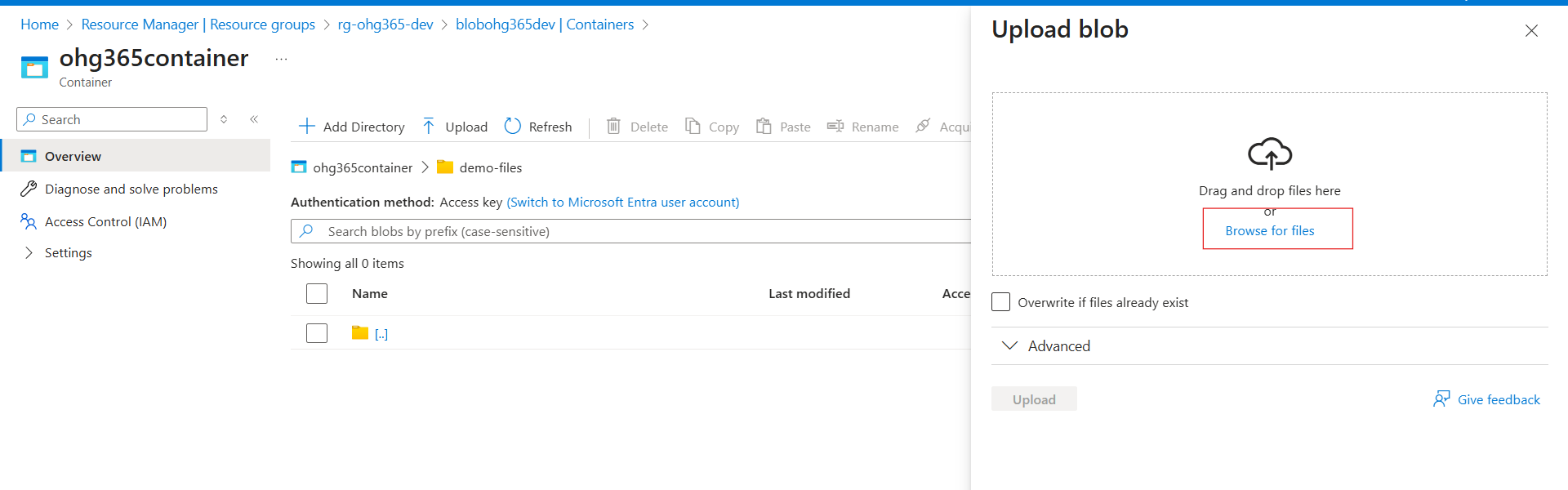
└── Container (like a folder)

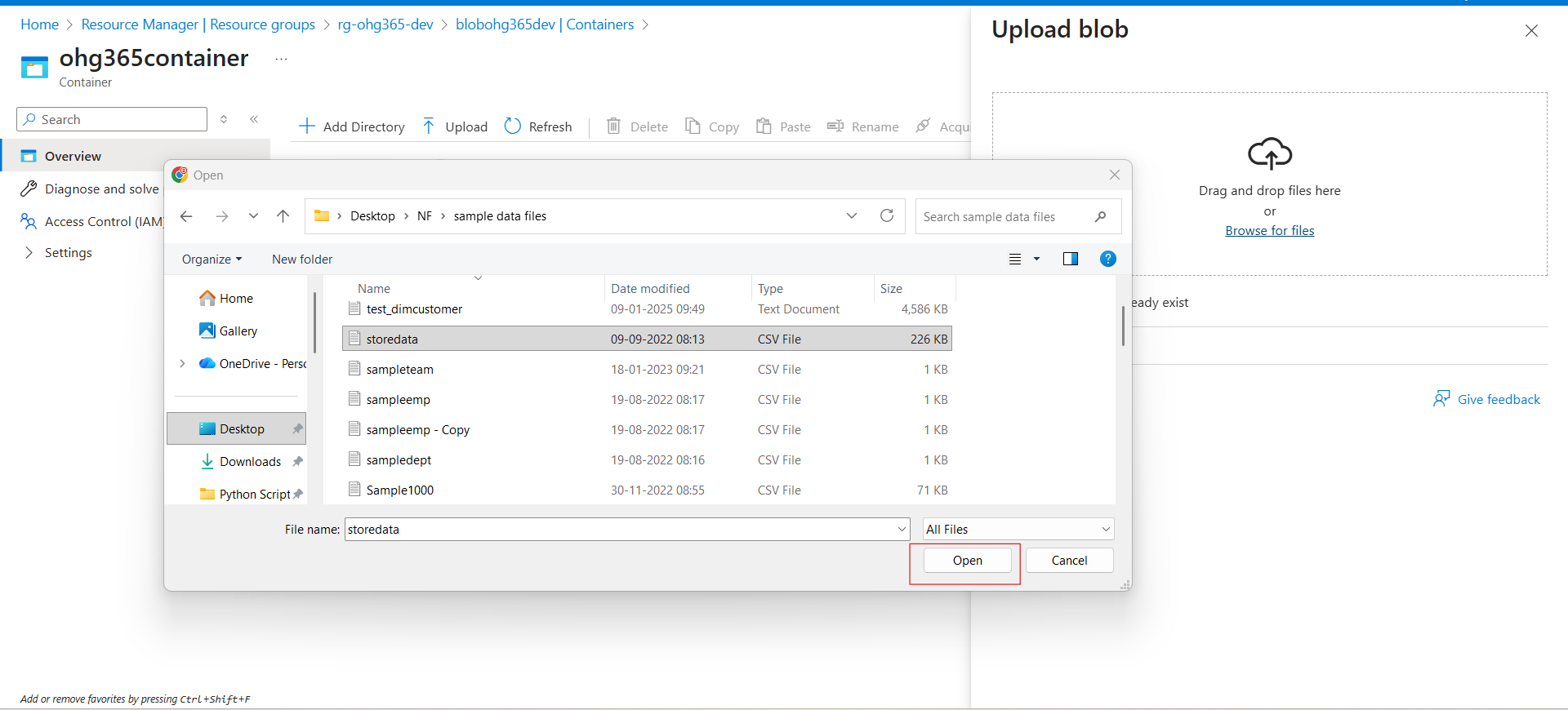
└── Blob (the actual file)

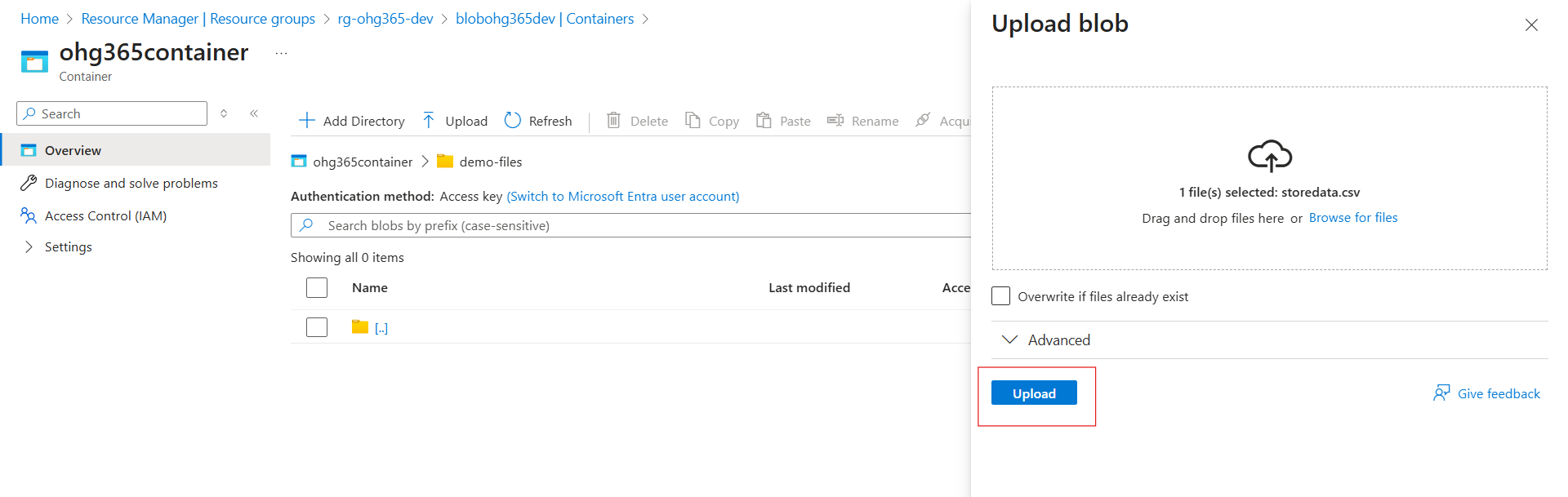






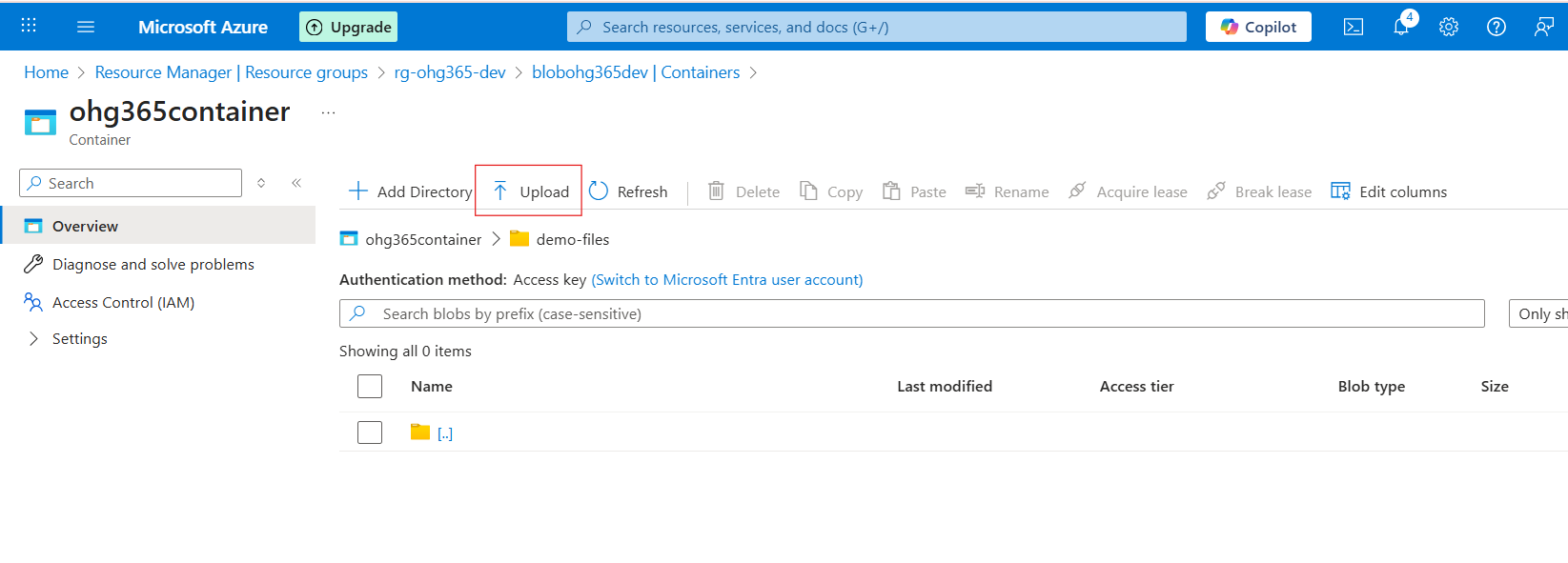


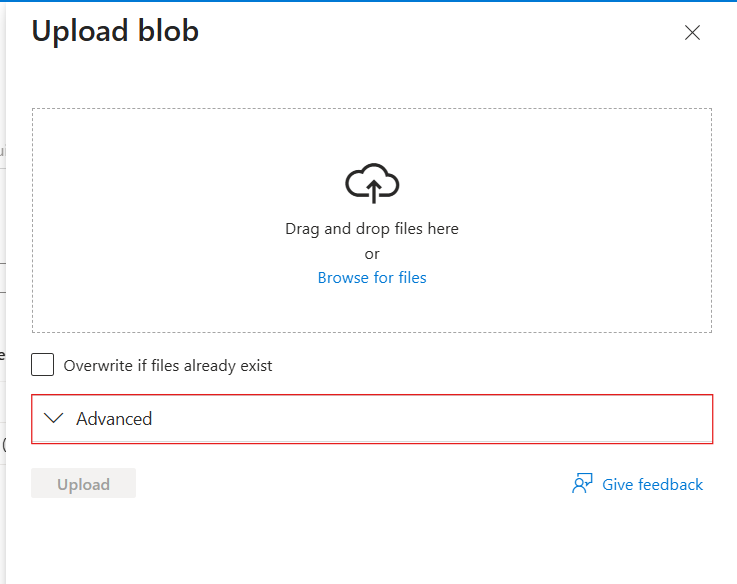


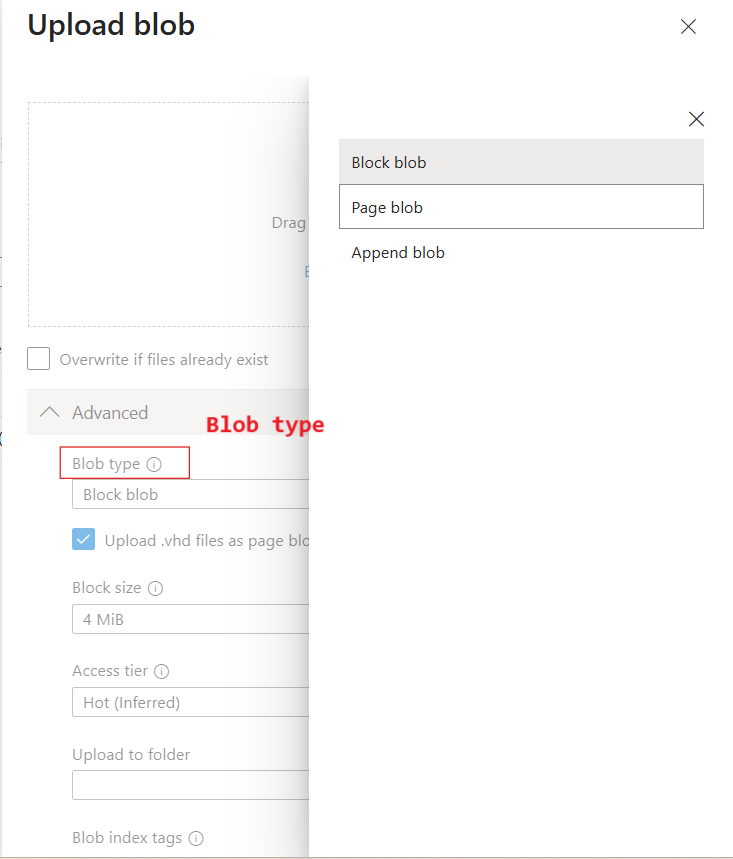


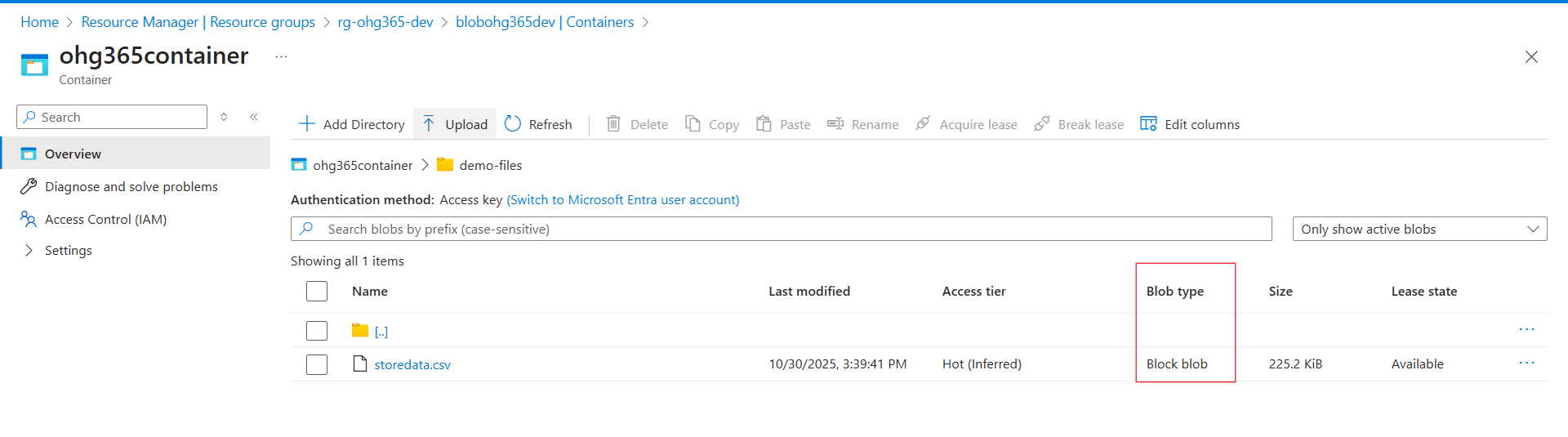


**Types of Blob Types:**



****

****

****

| **Blob Type** | **Best For** | **Example Use** |
| --- | --- | --- |
| 🧱 **Block Blob** | Storing text or binary data | Images, videos, documents, CSVs |
| 📜 **Append Blob** | Data that is constantly added to (append-only) | Logs, telemetry, audit data |
| 📄 **Page Blob** | Random read/write access | Virtual machine disks (VHD files) |

**🧱 1. Block Blob**

**Most common blob type** used in Azure.

**💡 What it is:**

* Stores **text** and **binary** data (files like .txt, .jpg, .mp4, .csv, etc.)
* Data is split into **blocks**, and each block is identified by a block ID.
* You can upload or update blocks individually and commit them together.

**✅ Use Cases:**

* Storing images, videos, PDFs, and backups.
* Data files for analytics (CSV, JSON, Parquet).
* Large files uploaded in chunks.

**📘 Example:**

You upload a 500 MB video file — Azure divides it into smaller **blocks** and uploads each part separately for speed and reliability.

**📜 2. Append Blob**

**Special type of blob for data that grows over time.**

**💡 What it is:**

* Optimized for **append operations** — you can only add new data to the end, not modify or delete existing data.
* Each time you add new information, it’s appended to the blob.

**✅ Use Cases:**

* Storing log files.
* Application telemetry or diagnostics data.
* Streaming data that’s constantly being added.

**📘 Example:**

You’re logging website visits. Each time a new visitor arrives, their data (timestamp, IP, etc.) is appended to the existing log file.

**📄 3. Page Blob**

**Designed for random read/write operations.**

**💡 What it is:**

* Data is stored in **fixed-size 512-byte pages**.
* Allows **fast read and write access** to specific parts of the blob.
* Commonly used for storing **Virtual Hard Disk (VHD)** files that power Azure Virtual Machines.

**✅ Use Cases:**

* Storing Azure VM disks (OS and data disks).
* Large databases that require random access.
* Any workload that reads/writes frequently to specific sections of a file.

**📘 Example:**

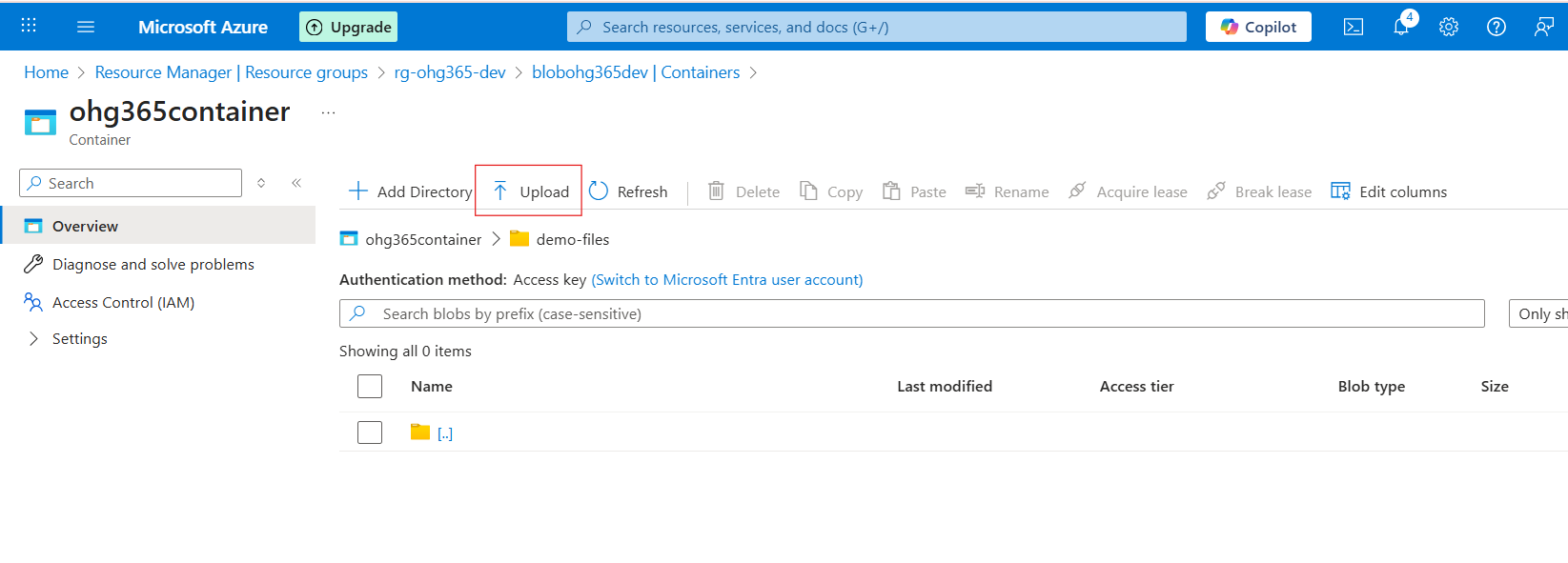
When you start an Azure Virtual Machine, its disk (a .vhd file) is stored as a **Page Blob**, allowing the VM to quickly read or write data anywhere on the disk.

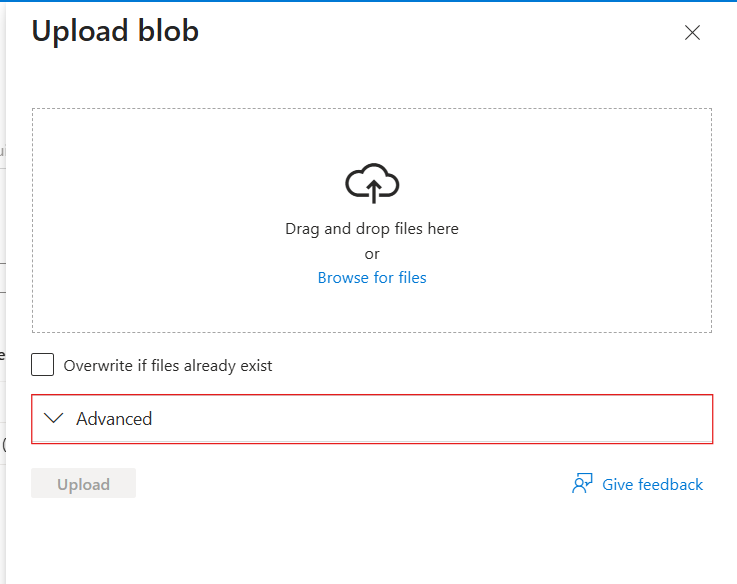
| **Blob Type** | **Structure** | **Read/Write Behavior** | **Common Use** |
| --- | --- | --- | --- |
| **Block Blob** | Data stored as blocks | Upload/replace blocks | Files, media, documents |
| **Append Blob** | Sequentially added blocks | Append-only | Logs, telemetry, streaming data |
| **Page Blob** | Fixed 512-byte pages | Random read/write | VM disks, large databases |

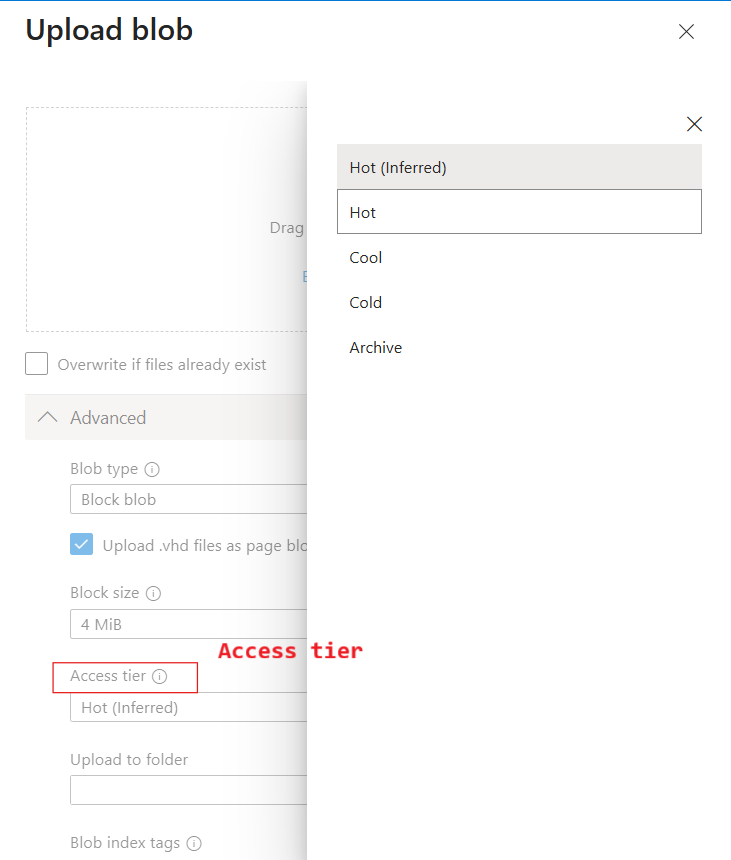
Types of Access Tiers:

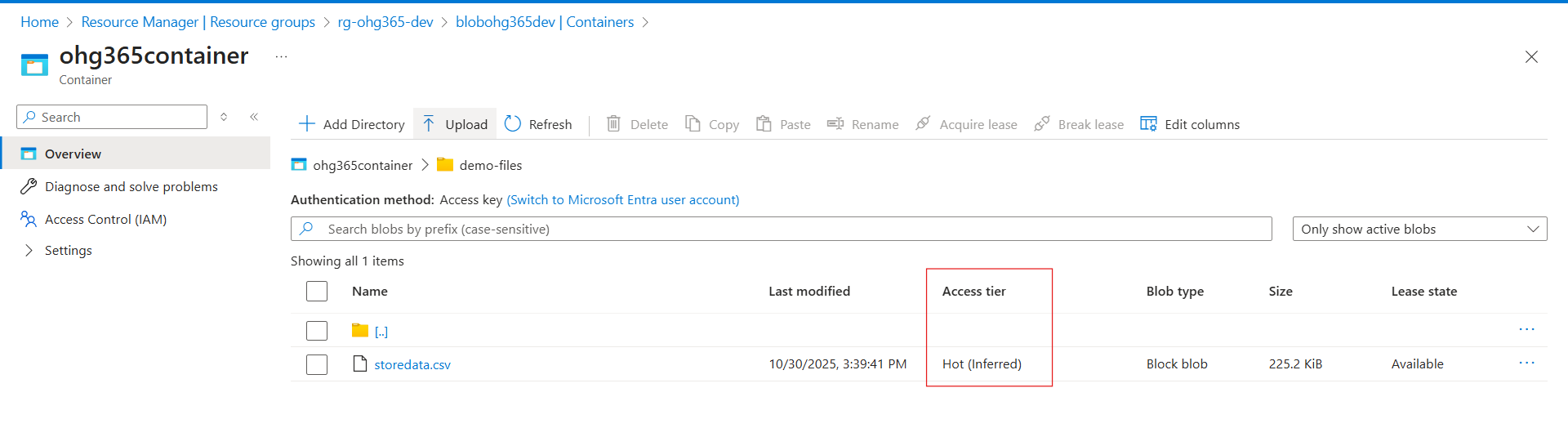
Azure lets you store data in **different tiers** based on how often you need it.  
This helps **save money** 💰 by matching storage cost to usage.

| **Tier** | **Cost** | **Availability** | **Best For** |
| --- | --- | --- | --- |
| **Hot** | 💰 Highest cost | 🔥 Always available | Frequently accessed data (e.g., active apps, websites) |
| **Cool** | 💸 Cheaper | 🕓 Slight delay in access | Infrequently accessed data (e.g., monthly reports) |
| **Cold** | 💧 Cheaper than Cool | ⏱️ Slower access | Rarely accessed data but still retrievable |
| **Archive** | 🧊 Cheapest | 💤 Retrieval takes hours | Long-term backups, compliance storage |



****





4. **Azure Data Lake Storage Gen2 (ADLS Gen2)**

Azure Data Lake Storage Gen2 is a highly scalable and secure **cloud storage service optimized for big data analytics** and data lakes. It builds on Azure Blob Storage capabilities but adds **file system semantics, hierarchical namespaces, and enhanced performance** for analytics workloads.

ADLS Gen2 is designed to store massive volumes of **structured, semi-structured, and unstructured data**, making it ideal for big data and machine learning scenarios.

**Key Features and Use Cases:**

* **Hierarchical Namespace:** Unlike traditional Blob Storage, ADLS Gen2 supports folders and directories, enabling efficient organization and faster file operations at scale.
* **Optimized for Analytics:** Supports Hadoop Distributed File System (HDFS) and integrates seamlessly with analytics frameworks like Azure Databricks, HDInsight, and Azure Synapse Analytics.
* **Supports Multiple Data Types:** You can store CSV, JSON, Parquet, Avro, ORC, images, videos, logs, backups, and more.
* **Security and Compliance:** Provides enterprise-grade security with Azure Active Directory integration, role-based access control (RBAC), and encryption at rest and in transit.
* **Cost-effective and Scalable:** Automatically scales to handle petabytes of data and millions of files with optimized storage tiers and pricing options.

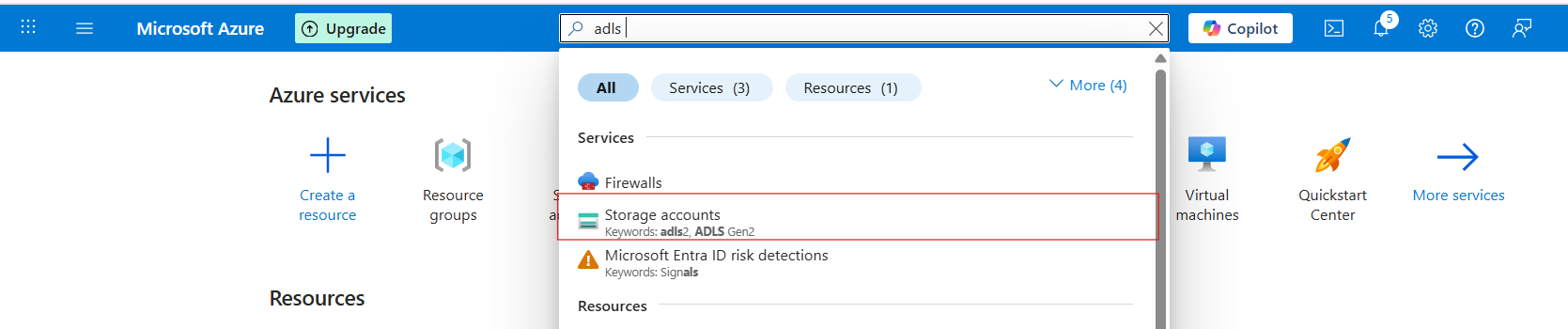
**Common Uses of ADLS Gen2:**

* Building **data lakes** for big data analytics and machine learning.
* Storing large datasets for **ETL (Extract, Transform, Load)** processes.
* Integrating with analytics tools to perform complex queries and transformations.
* Secure and compliant **storage for sensitive data** and audit logs.
* Archiving and long-term data retention with tiered storage options.

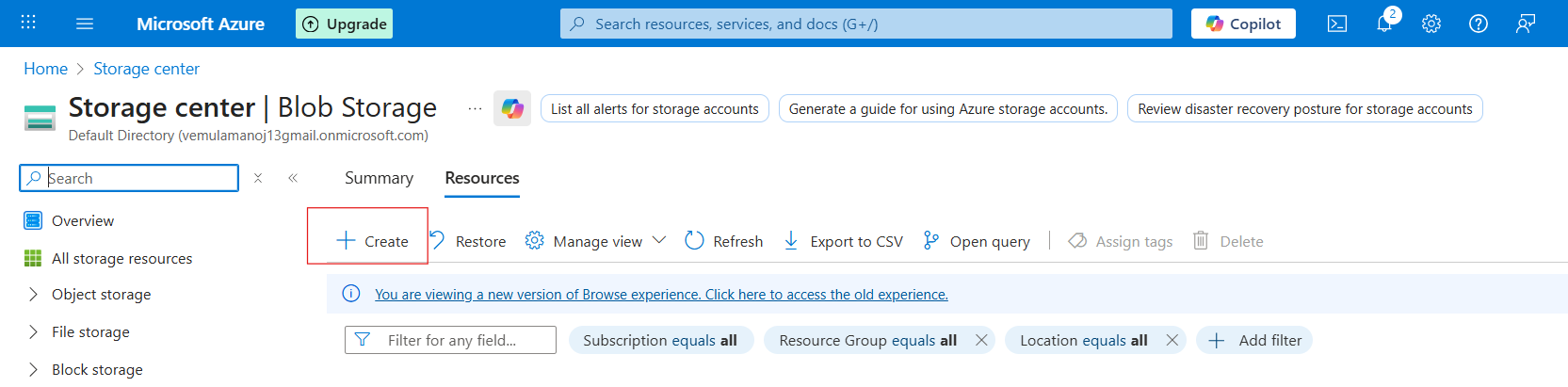
**4.1 Create an Azure Data Lake Storage**

**Go to Azure Portal**

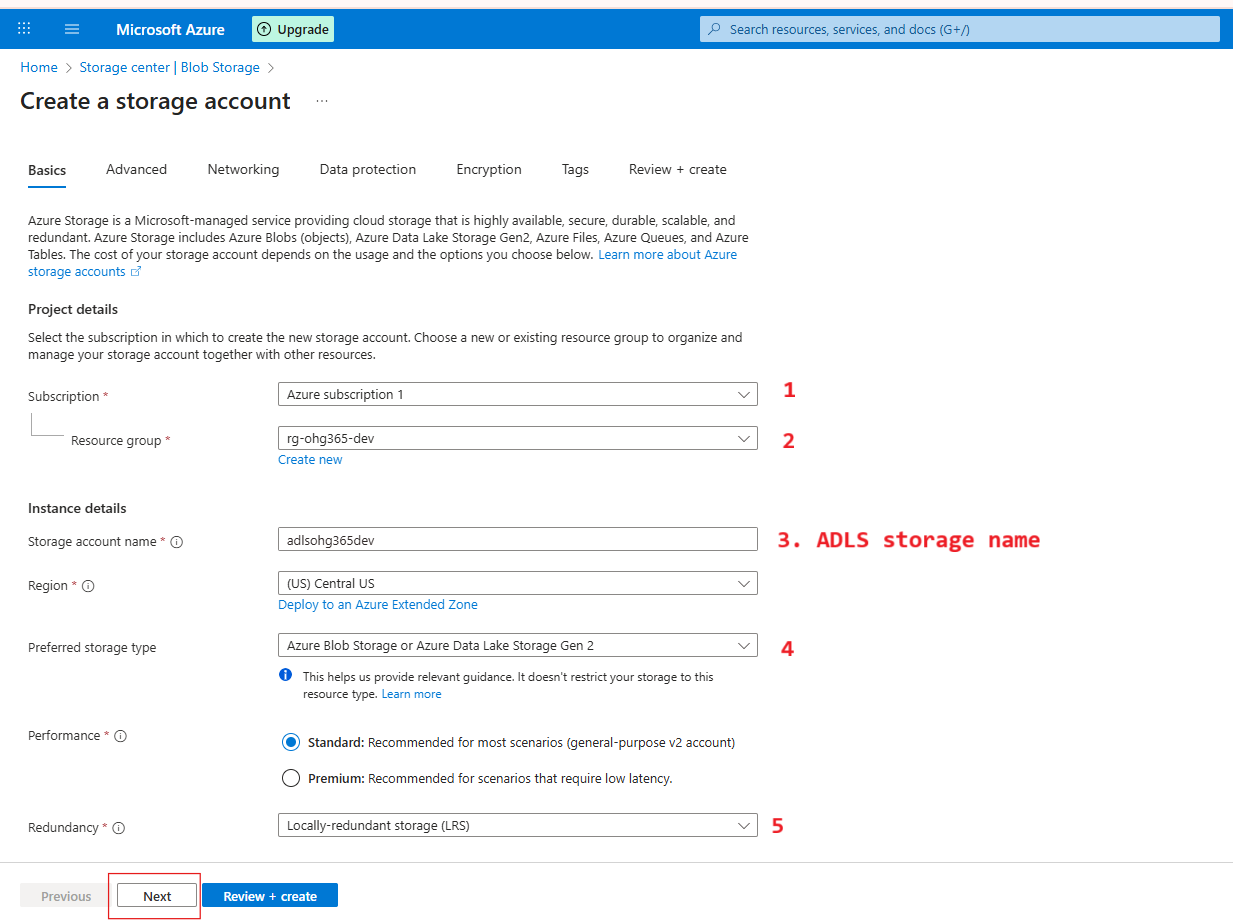
**In the search bar, type “Storage Accounts” or “ADLS Gen2”**

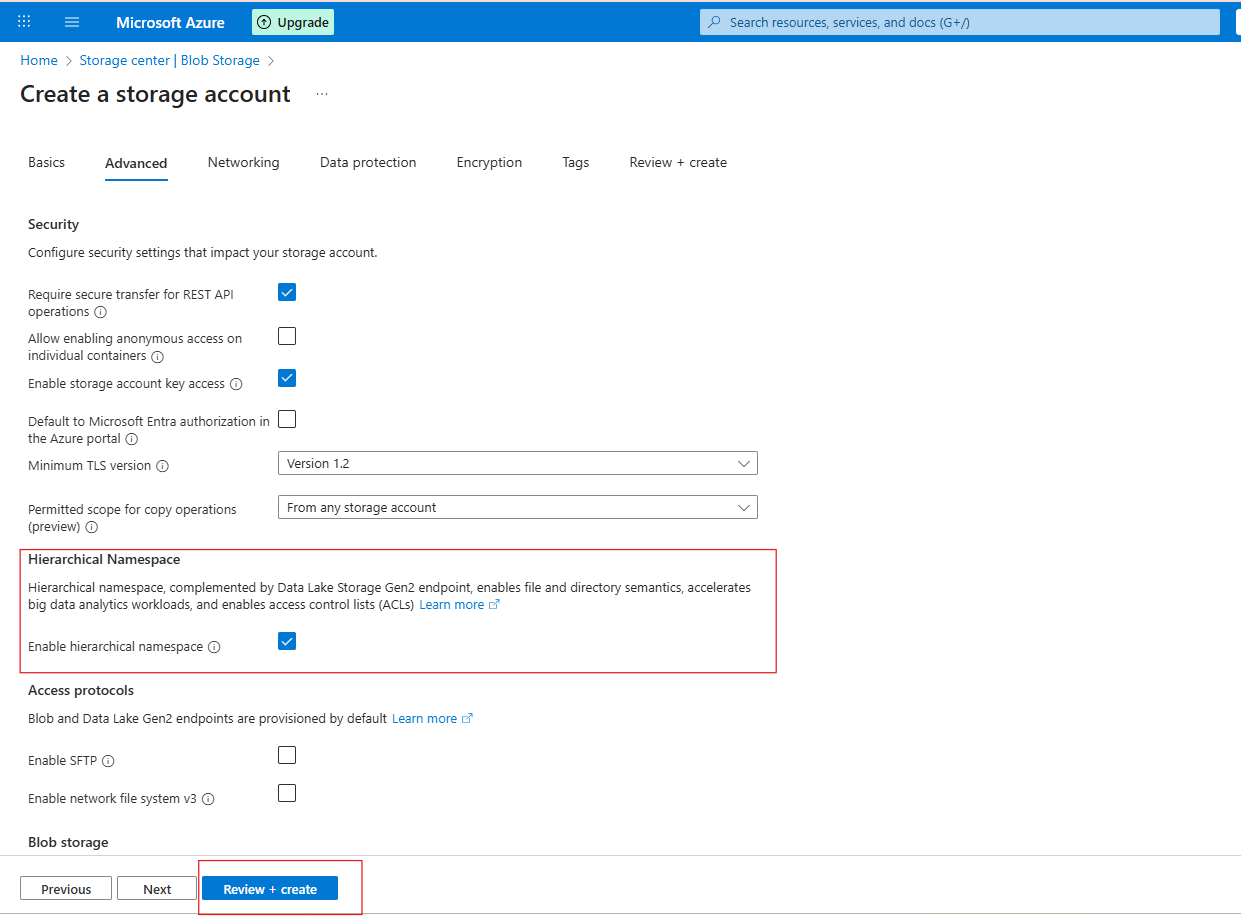
****

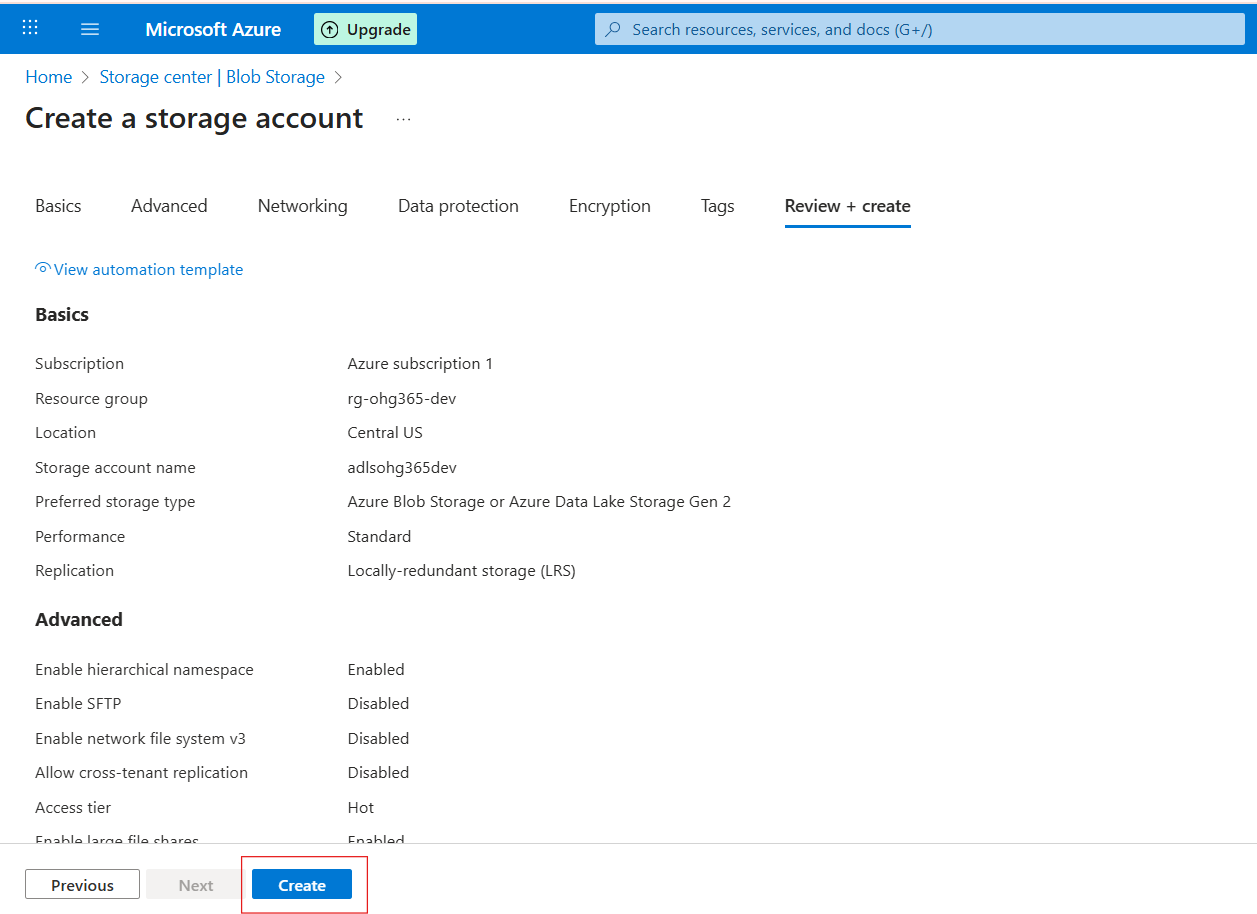
**Click on storage accounts and click on create button**

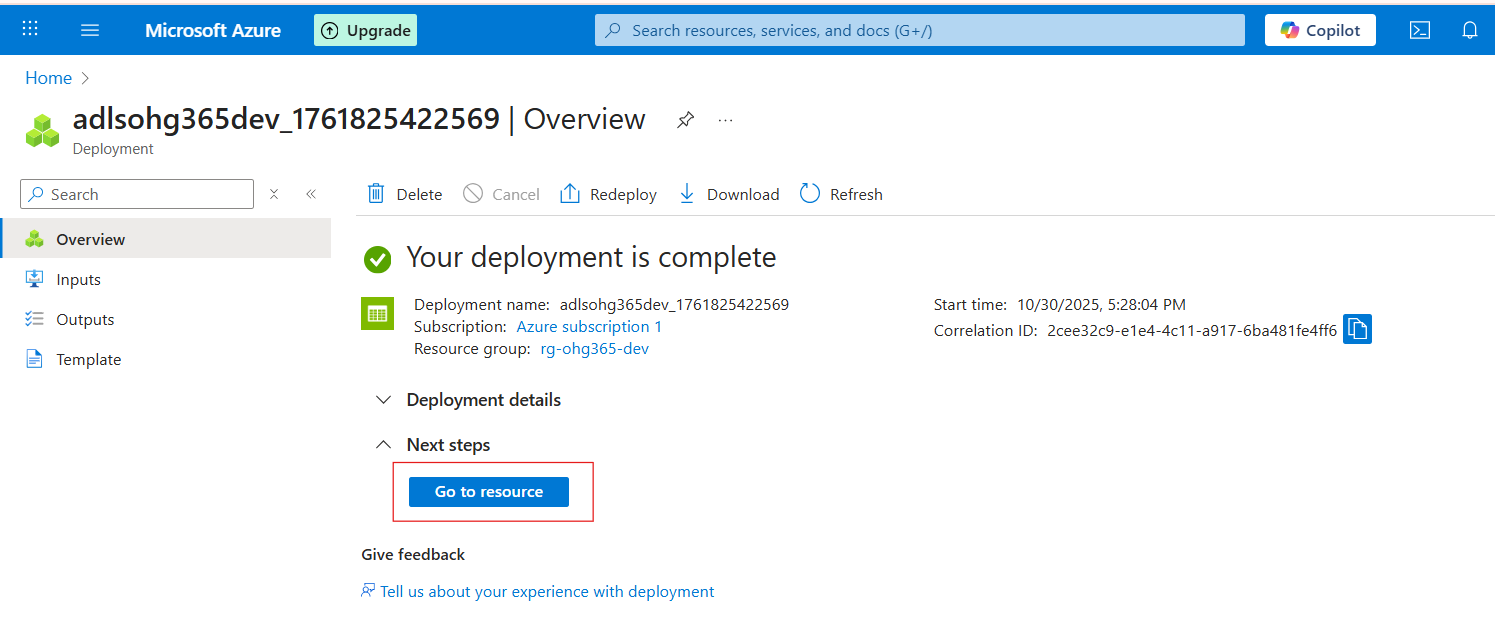
****

**Fill required details**

****

****

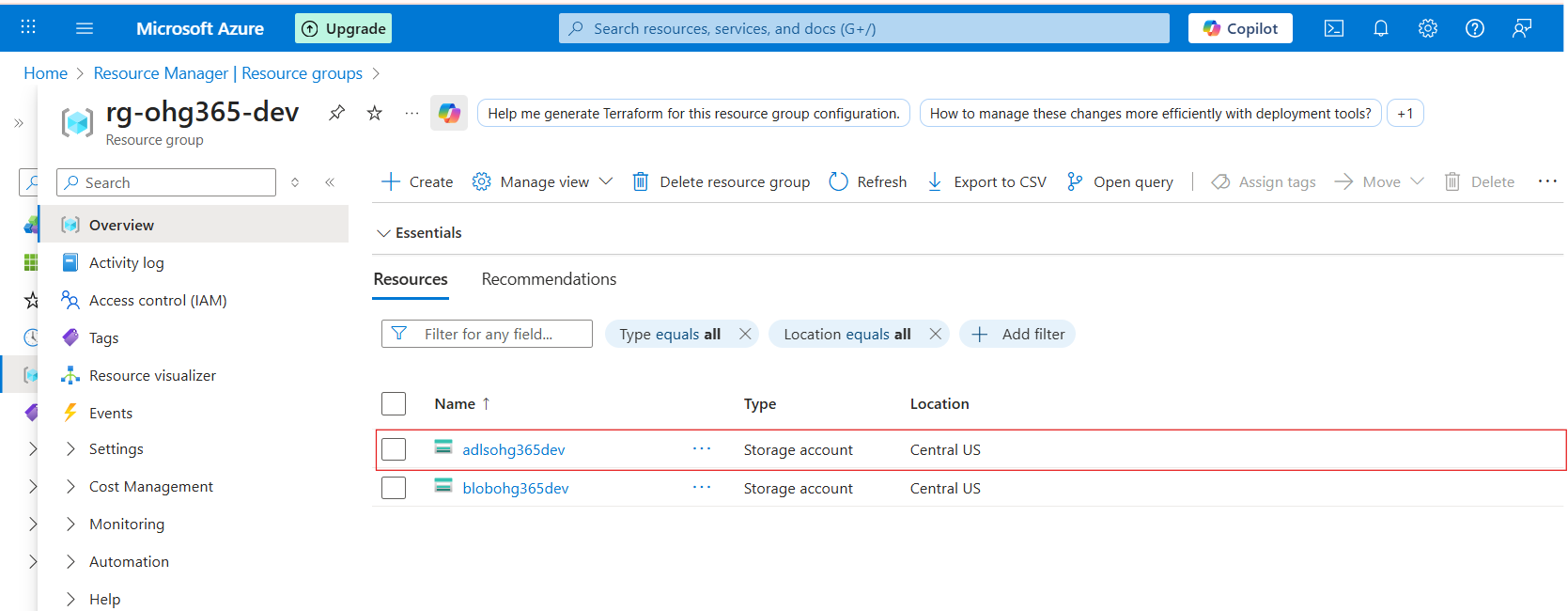
****

****

**Navigate to Your Resource Group**

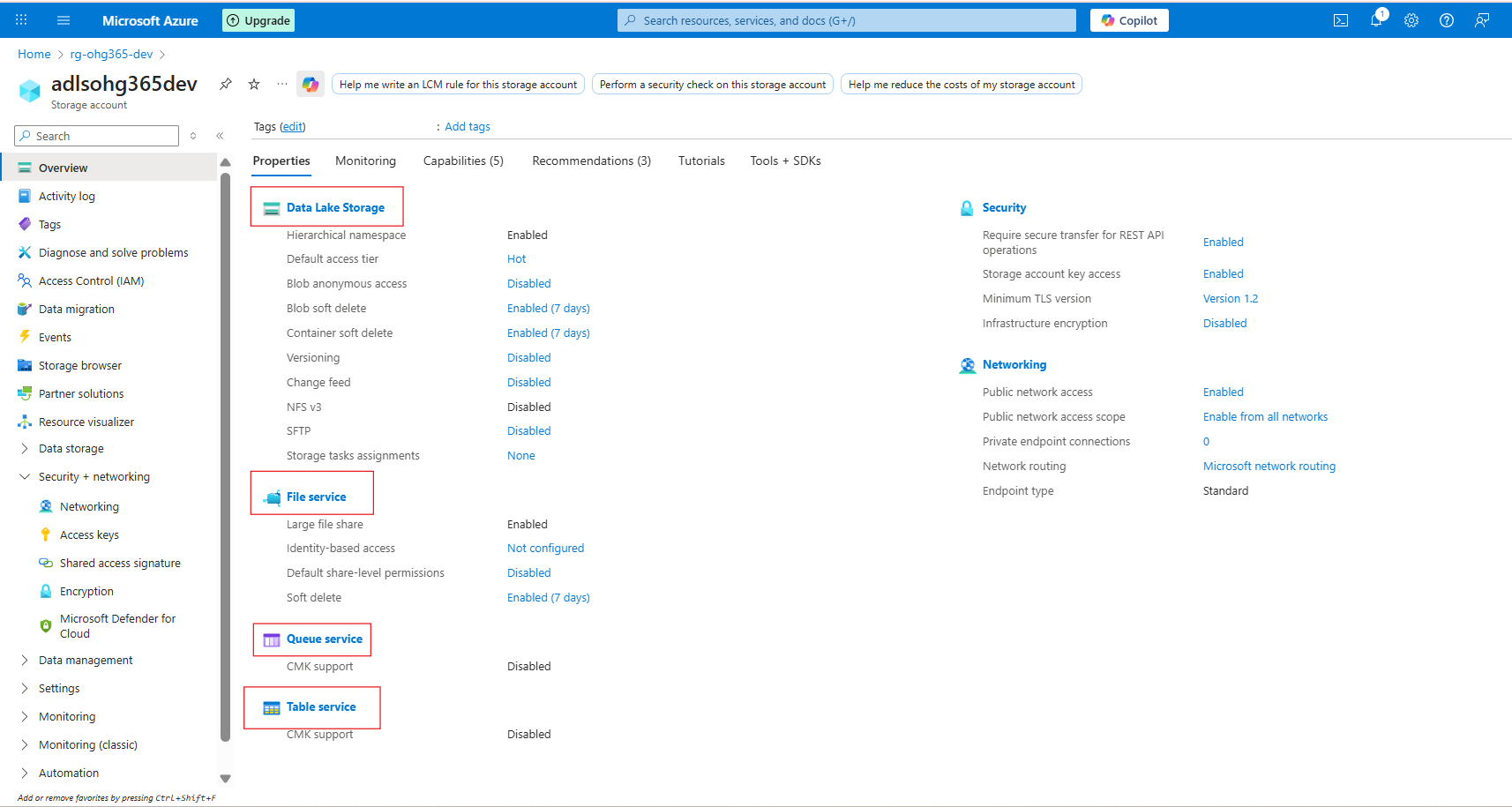
* In the left-hand menu, click on **Resource groups**
* Find and click your resource group — in your case:  
  👉 **rg-ohg365-dev**
* **Check for the Storage Account**
* Inside the **Overview** tab of your resource group, you’ll see a list of all resources.
* Look for an item that looks like this:  
  **Type:** *Storage account*  
  **Name:** [***adlsohg365dev***](https://portal.azure.com/#resource/subscriptions/149fbd78-677c-4f5b-90aa-52a82b340803/resourceGroups/rg-ohg365-dev/providers/Microsoft.Storage/storageAccounts/adlsohg365dev)(or whatever name you used)

If it appears there, 🎉 congratulations — your **ADLS Gen2 Storage account** has been successfully created



**3.2 Types of Azure Storage Services**:

Azure Storage provides **four main types of services** under one **Storage Account**.



**1️⃣ Data Lake Storage**

**🧱 Purpose: Designed to store large volumes of unstructured or semi-structured data such as files, images, videos, logs, and backups.  
🔹 Key Features:**

* **Data is stored as Blobs (Binary Large Objects) within containers.**
* **Ideal for storing large, raw data that doesn’t fit into traditional databases.**
* **Supports various formats: text, binary, documents, media, and backups.**

**💾 Common Use Cases:**

* **Hosting media content like images and videos for websites.**
* **Long-term storage for backups and archival data.**
* **Centralized data lake for analytics and big data processing.**
* **Hosting static websites.**

**📦 Supported File Types:  
.txt, .csv, .json, .xml, .jpg, .mp4, .zip, .bak, and more.**

**2️⃣ File Service**

📁 *Used for shared file storage that behaves like a traditional file server.*

**🔹 Description:**

* Provides **Azure Files**, a fully managed **file share** in the cloud.
* Uses the **SMB (Server Message Block)** or **NFS (Network File System)** protocols — the same used by on-premises file servers.
* Can be **mounted** to Windows, Linux, or macOS systems.

**💾 Example Uses:**

* Shared network drives for teams
* “Lift and shift” of on-premises file servers
* Application configurations shared across multiple VMs

**📦 Example Scenario:**

You have multiple virtual machines needing access to the same configuration files — you can store those files in **Azure Files** and mount them just like a shared folder.

**3️⃣ Queue Service**

📬 *Used for reliable messaging between application components.*

**🔹 Description:**

* Provides **asynchronous communication** between services using message queues.
* Stores messages in a **queue**, which can be processed later by background services or workers.
* Ensures messages are **delivered at least once** and processed in **FIFO (First-In, First-Out)** order.

**💾 Example Uses:**

* Sending background jobs (like image processing or email sending)
* Decoupling app components for scalability
* Event-driven architecture

**📦 Example Scenario:**

A web app uploads an image → sends a message to a **queue** → a background process picks it up and resizes the image.

**4️⃣ Table Service**

🧮 *Used to store large amounts of structured, non-relational data.*

**🔹 Description:**

* Provides **NoSQL key-value storage**.
* Stores data in **tables** with **entities (rows)** and **properties (columns)**.
* Flexible schema — you can add or remove columns anytime.

**💾 Example Uses:**

* Storing user profiles, IoT data, or metadata
* Fast lookups by key
* Lightweight applications needing scalable, cheap storage

**📦 Example Scenario:**

You have millions of IoT sensors sending temperature data — you can store this efficiently in **Azure Table Storage**.

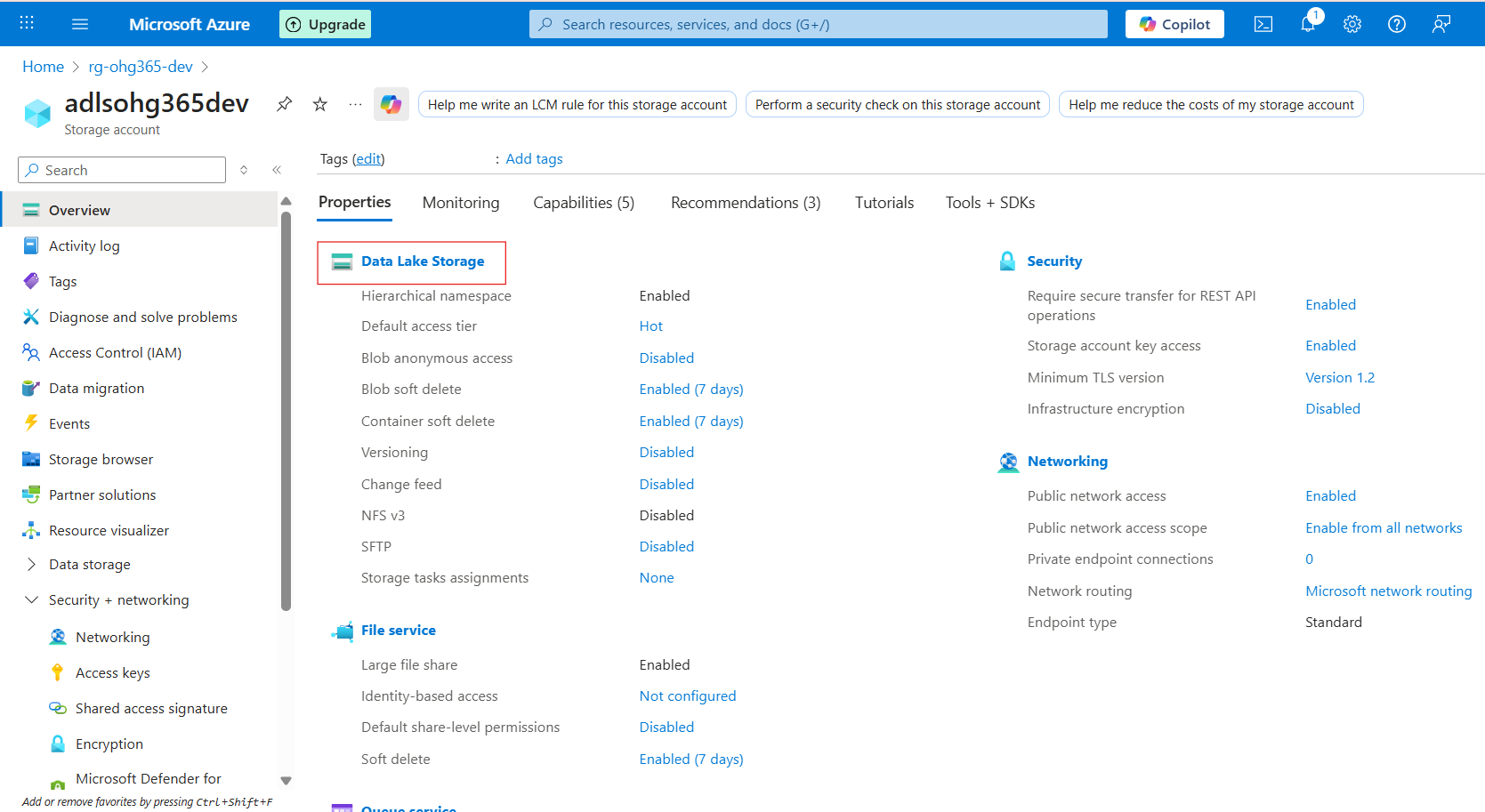
**🧠 Summary Table**

| **Datalake Service** | **Type of Data** | **Description** | **Example Use Case** |
| --- | --- | --- | --- |
| **Blob Service** | Unstructured / Semi-structured | Stores large objects (files, media, backups) | Images, videos, logs |
| **File Service** | File-based | Shared file storage via SMB/NFS | Shared drives, app configs |
| **Queue Service** | Messaging | Message-based communication between components | Background tasks, event processing |
| **Table Service** | Structured (NoSQL) | Key-value, schema-less table storage | User profiles, IoT data |

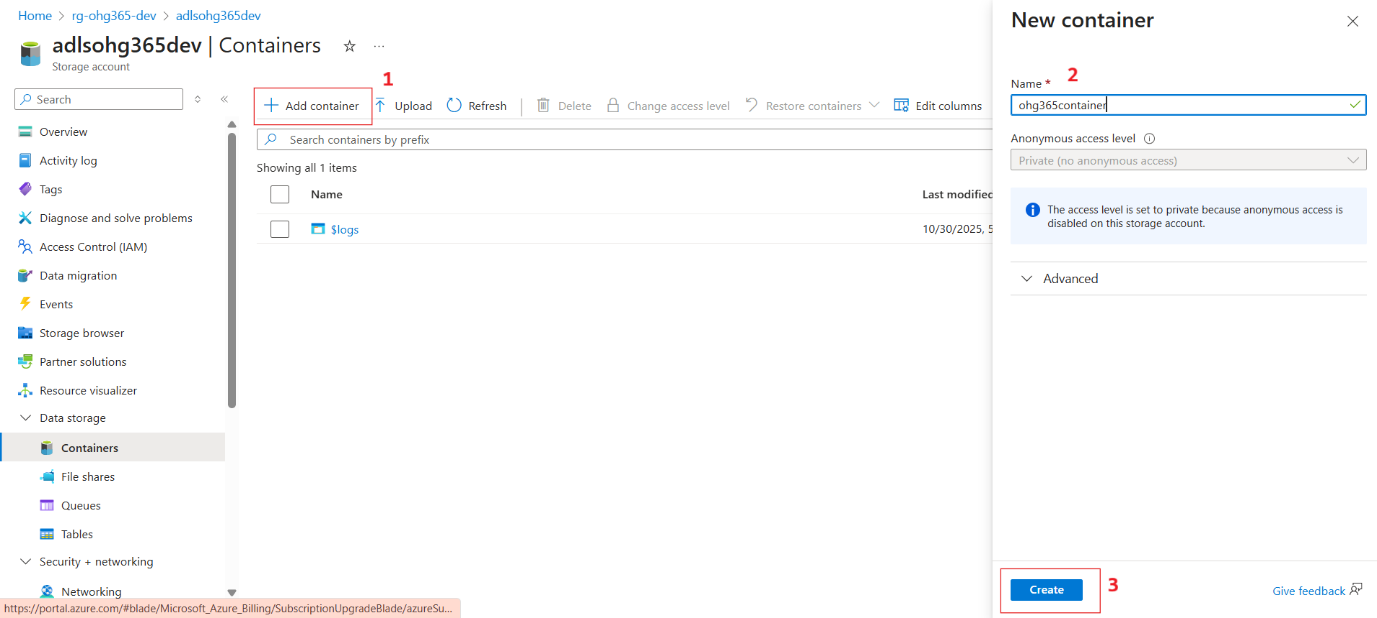
**Explore Datalake Service**

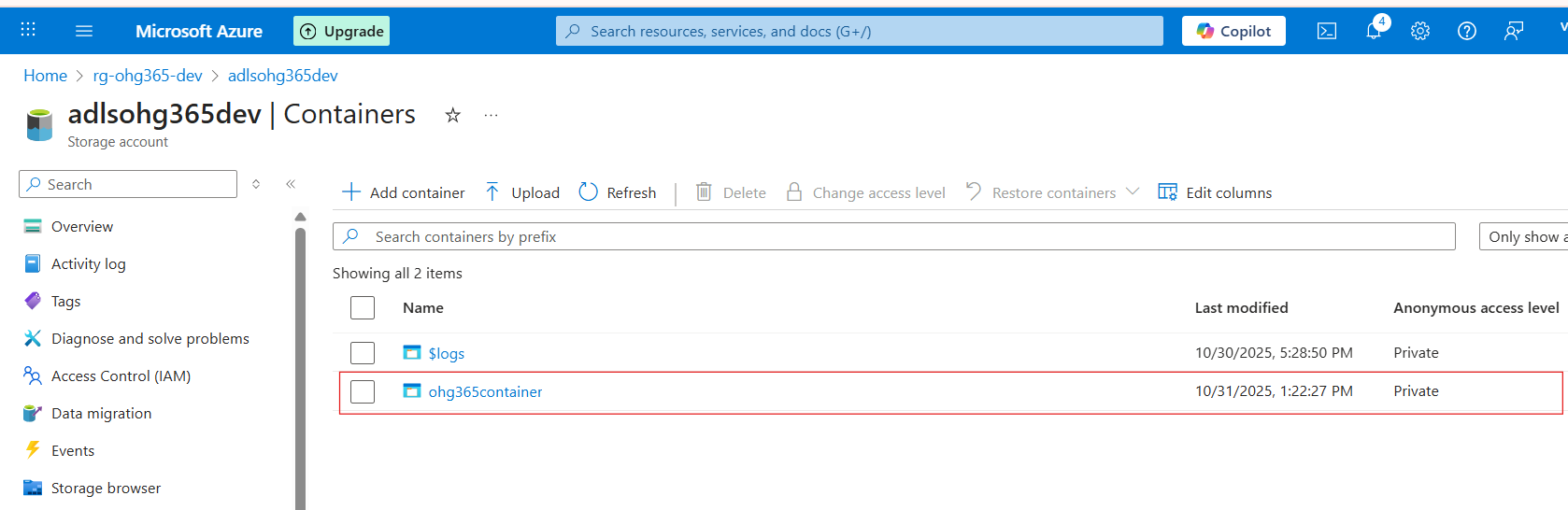
Once your storage account is created:

1. Go to your **Storage Account**



1. Under **Data storage**, click **Containers** → This is where your blobs live.
2. Click **➕ Container** to create one:
   * Name: images, videos, or backups (any name)
   * **Public access level:**
     + *Private (default)* – Only you can access
     + *Blob (anonymous read)* – Anyone with the link can read blobs
     + *Container (public)* – Everyone can see contents
   * Click **Create**





**Upload & Manage Blobs**

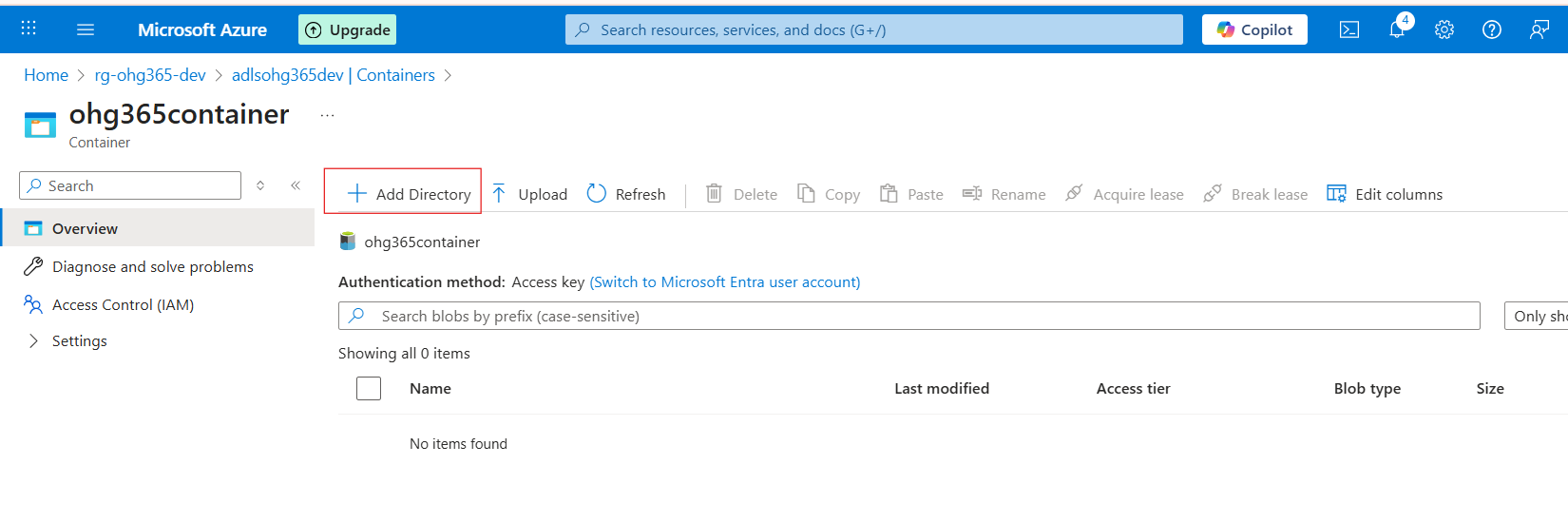
1. Click your new container (e.g., images)
2. Click **Upload**
3. Choose a file from your computer (like a .jpg, .txt, or .mp4)
4. Once uploaded, you can:
   * View **Properties** (size, type, last modified)
   * Get the **URL** to access the file
   * Change the **access tier**

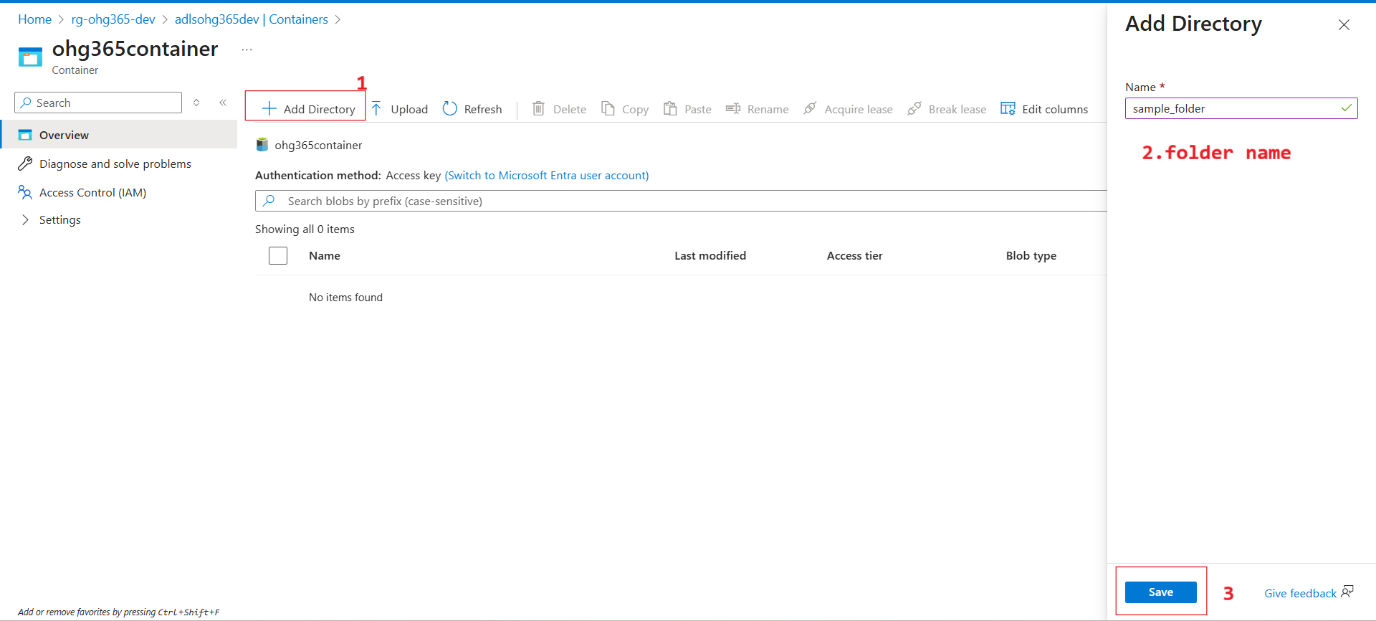
Storage Account

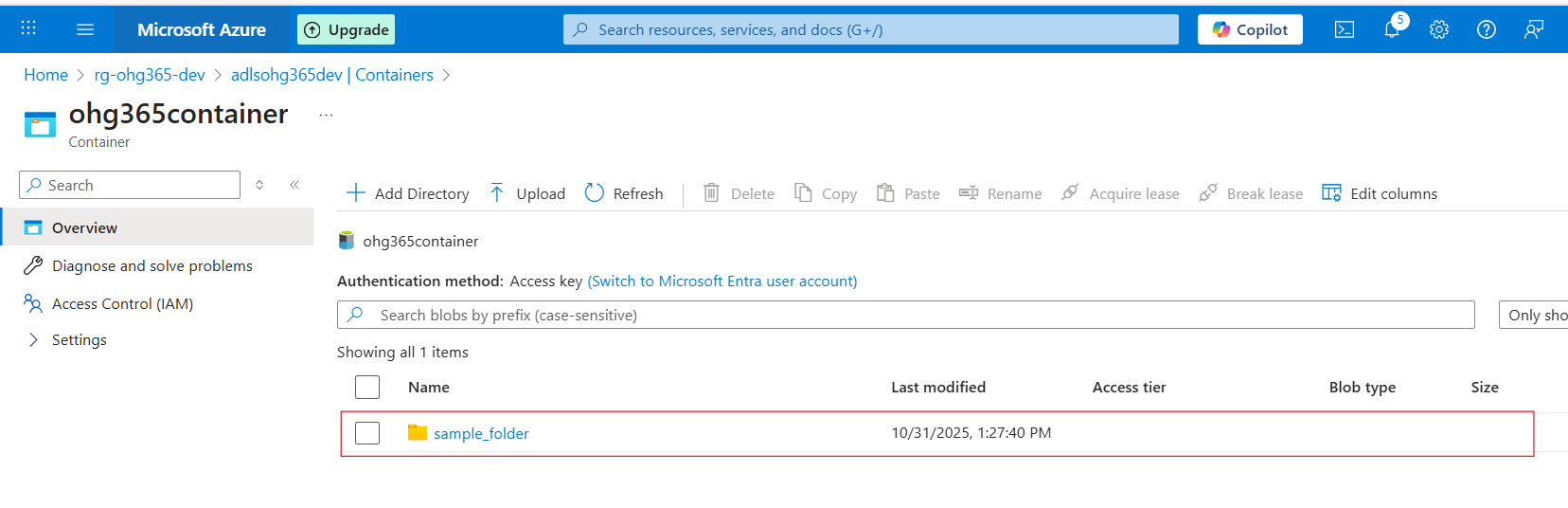
└── DataLake Service

└── Container (like a folder)

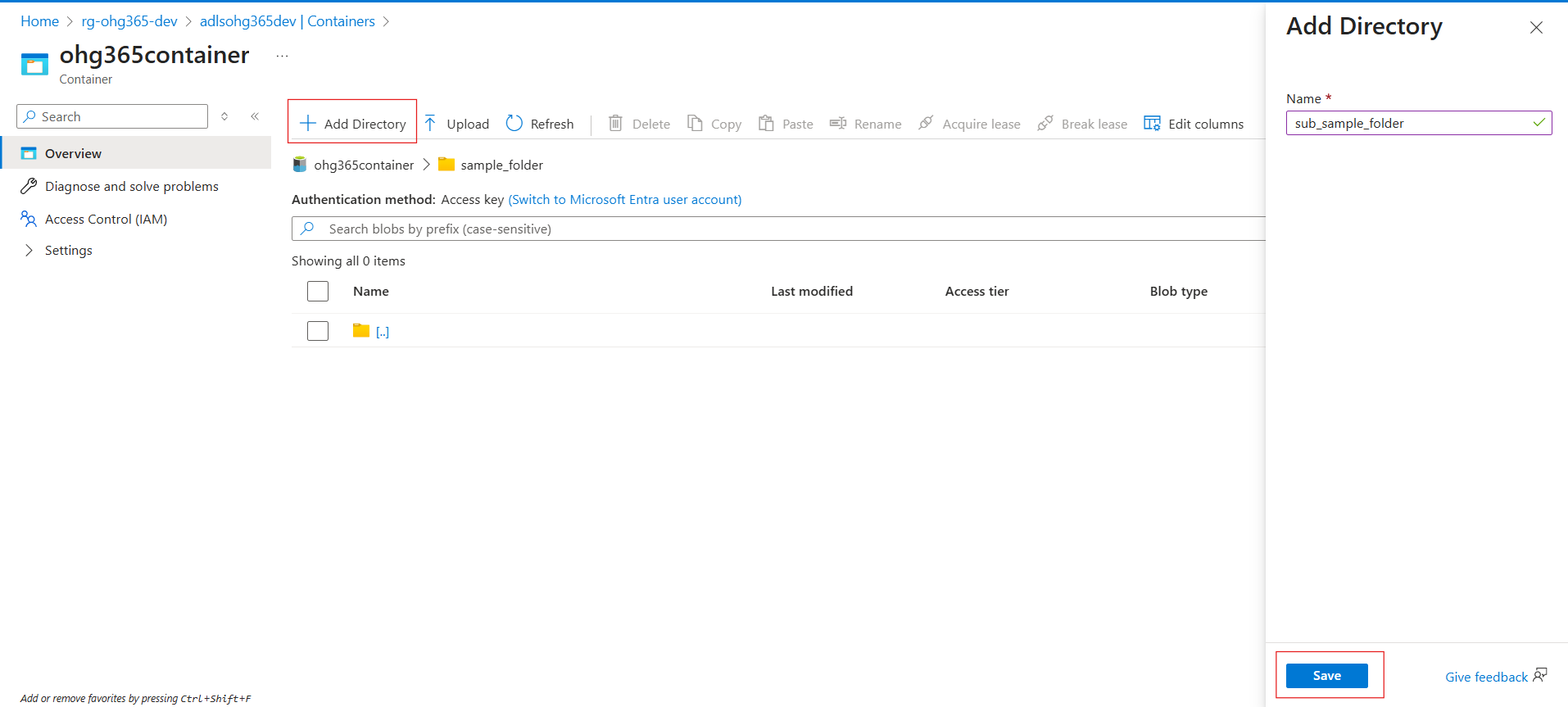
└── Blob (the actual file)

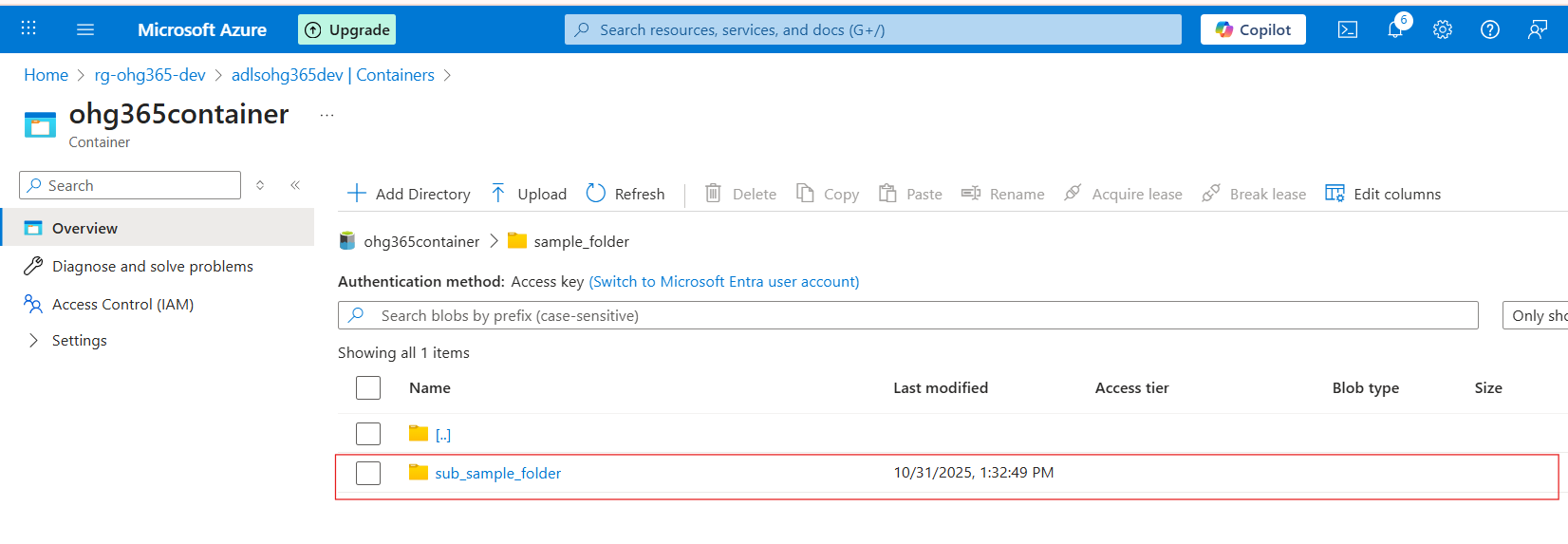




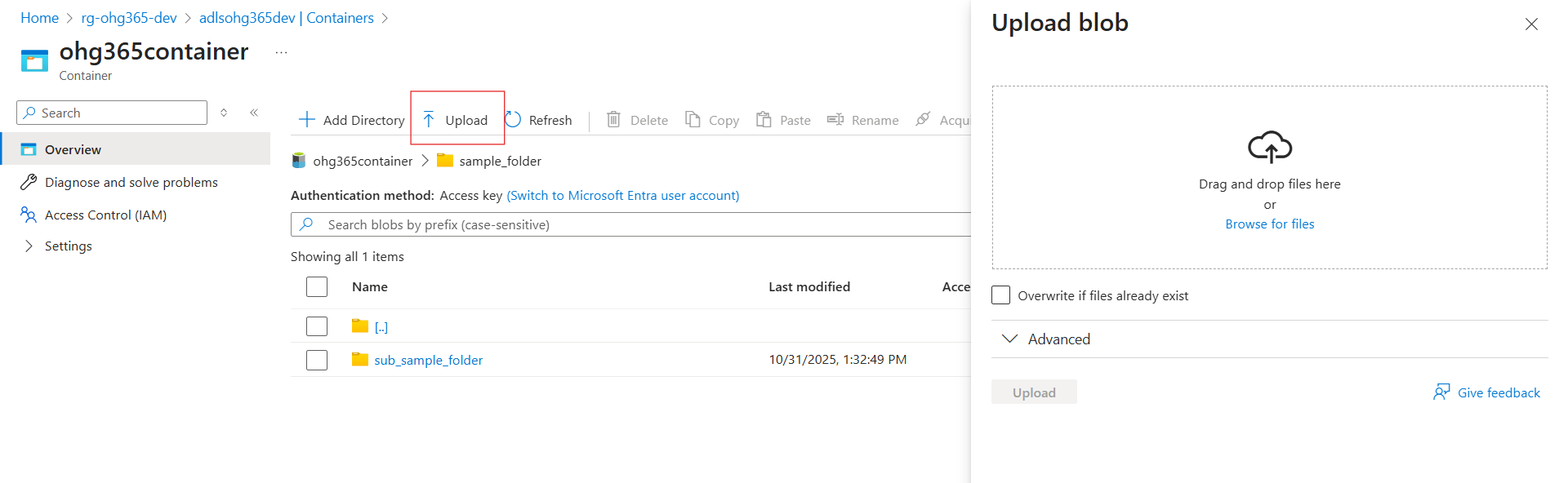


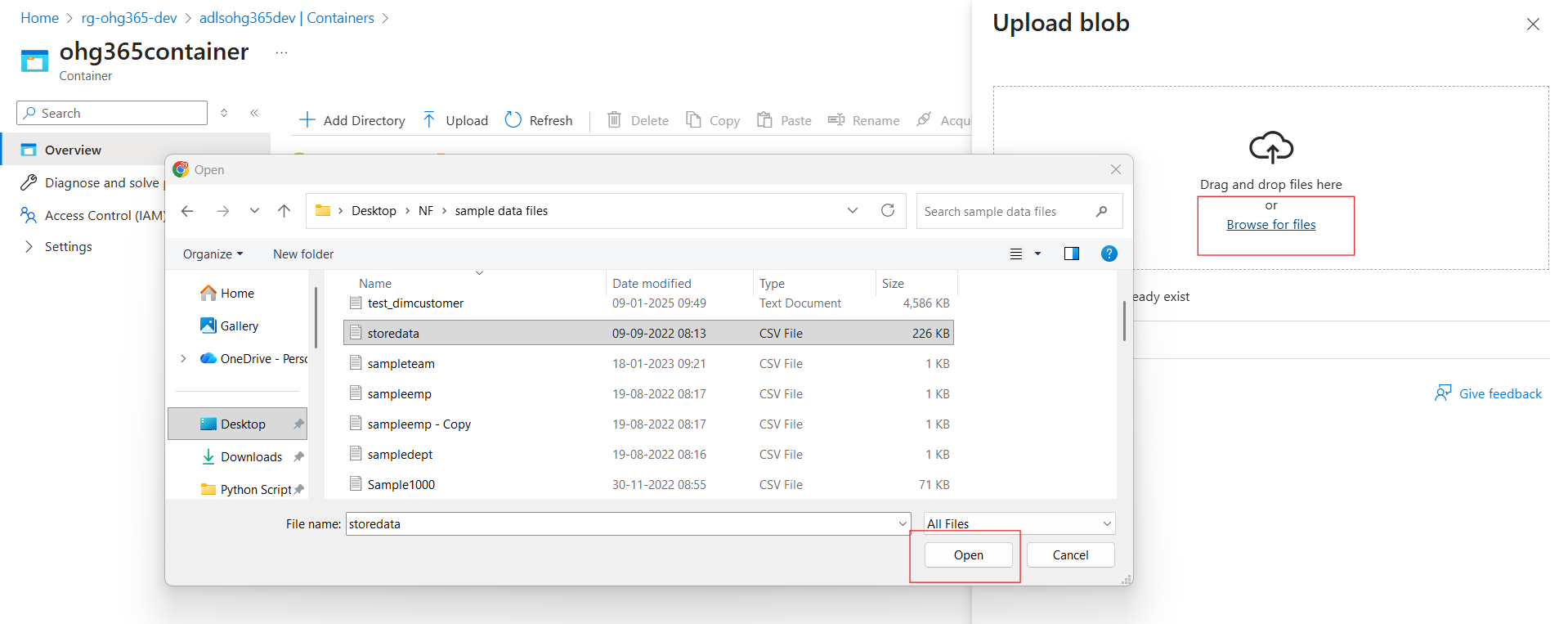
Subfolder:

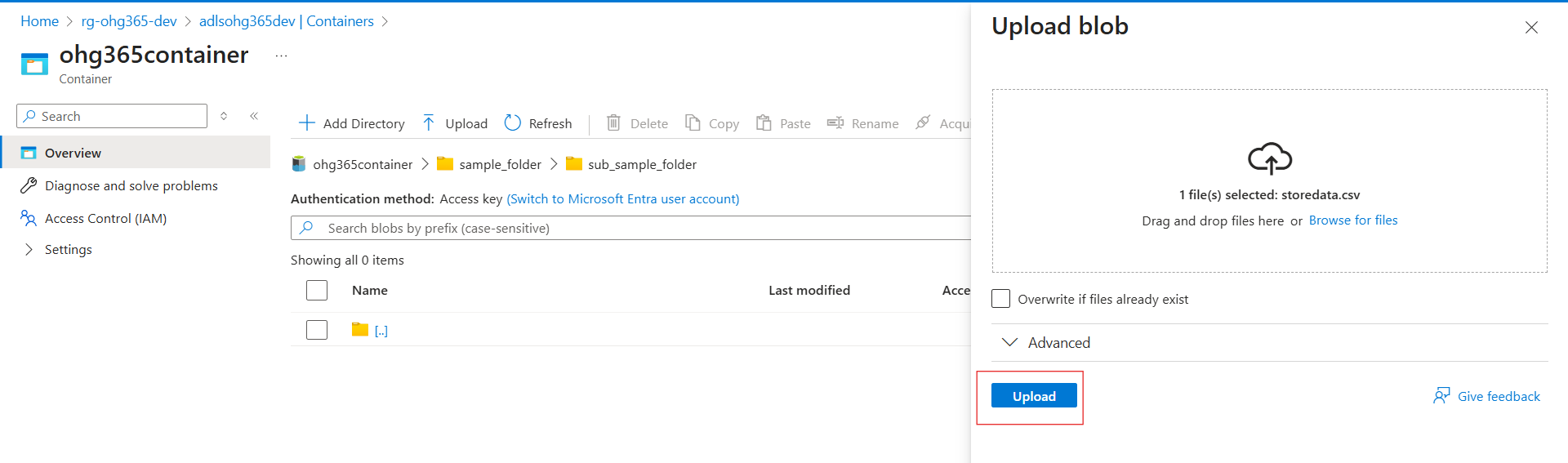


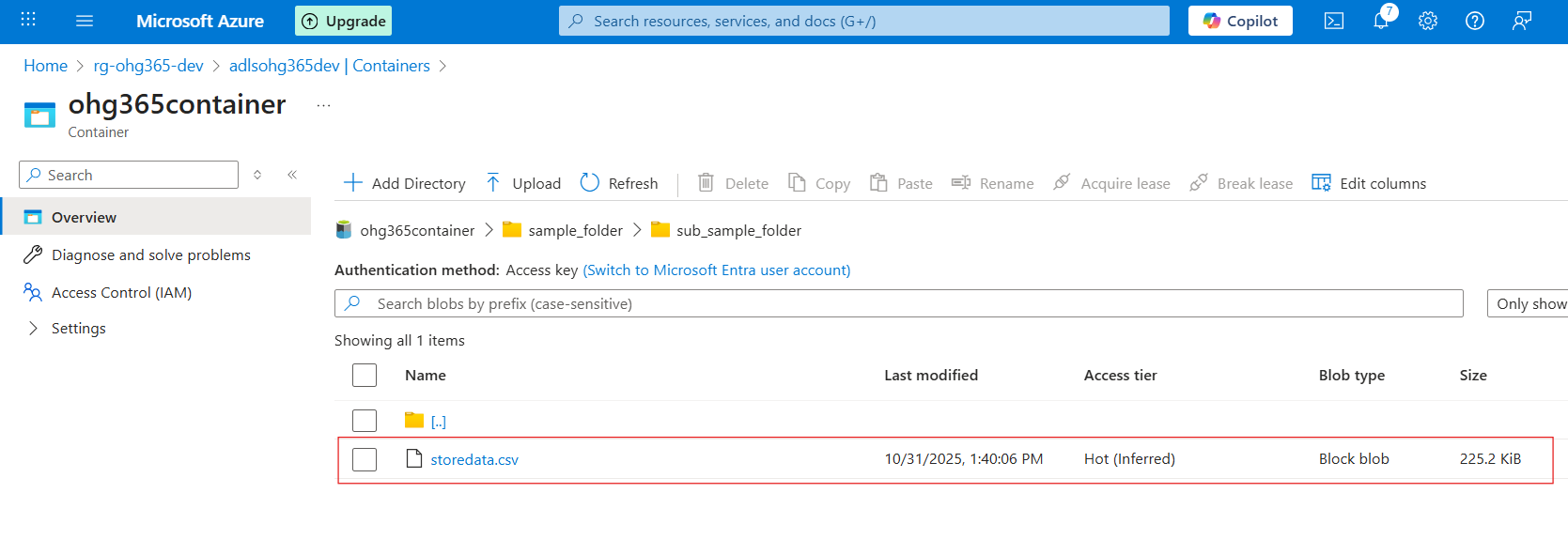


Upload file:

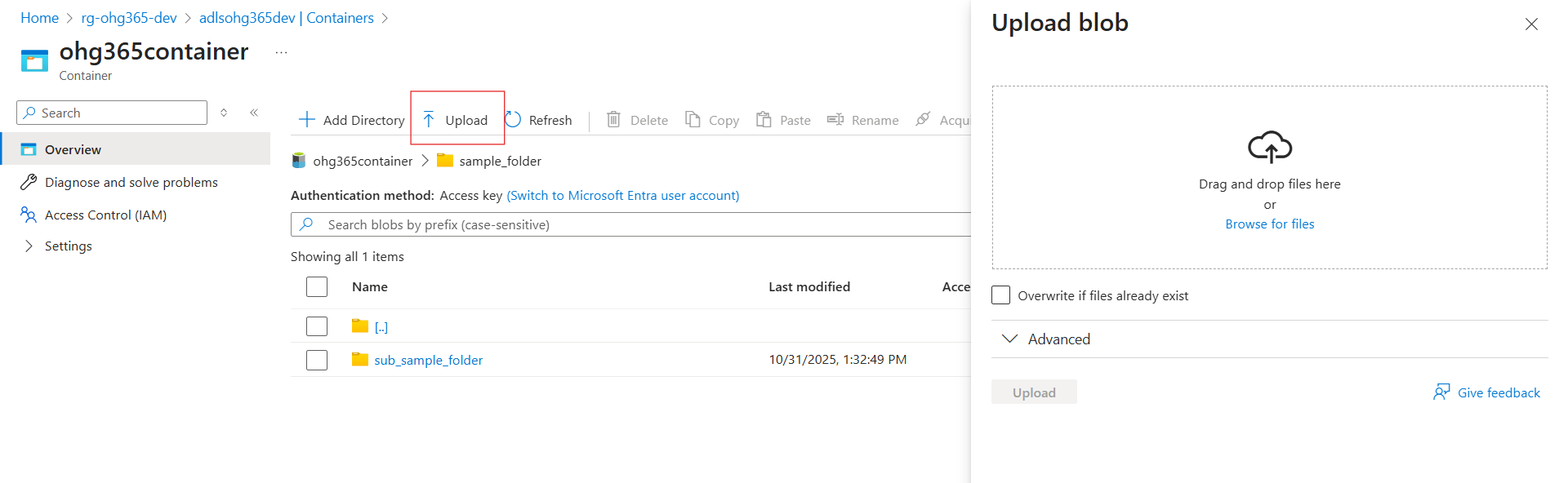


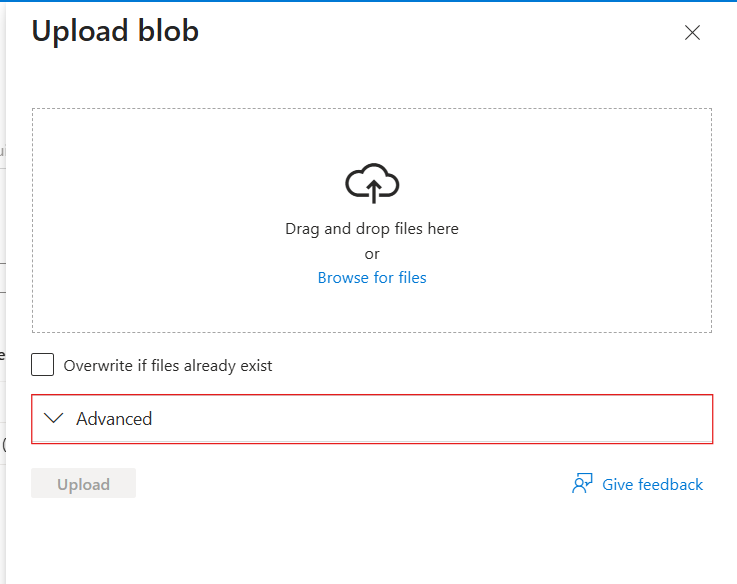


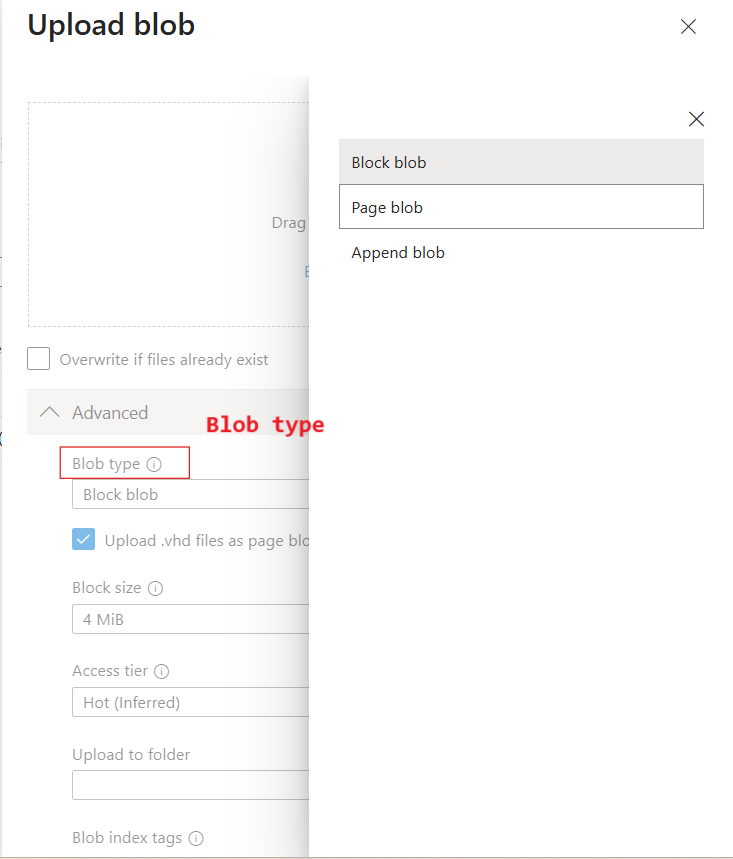


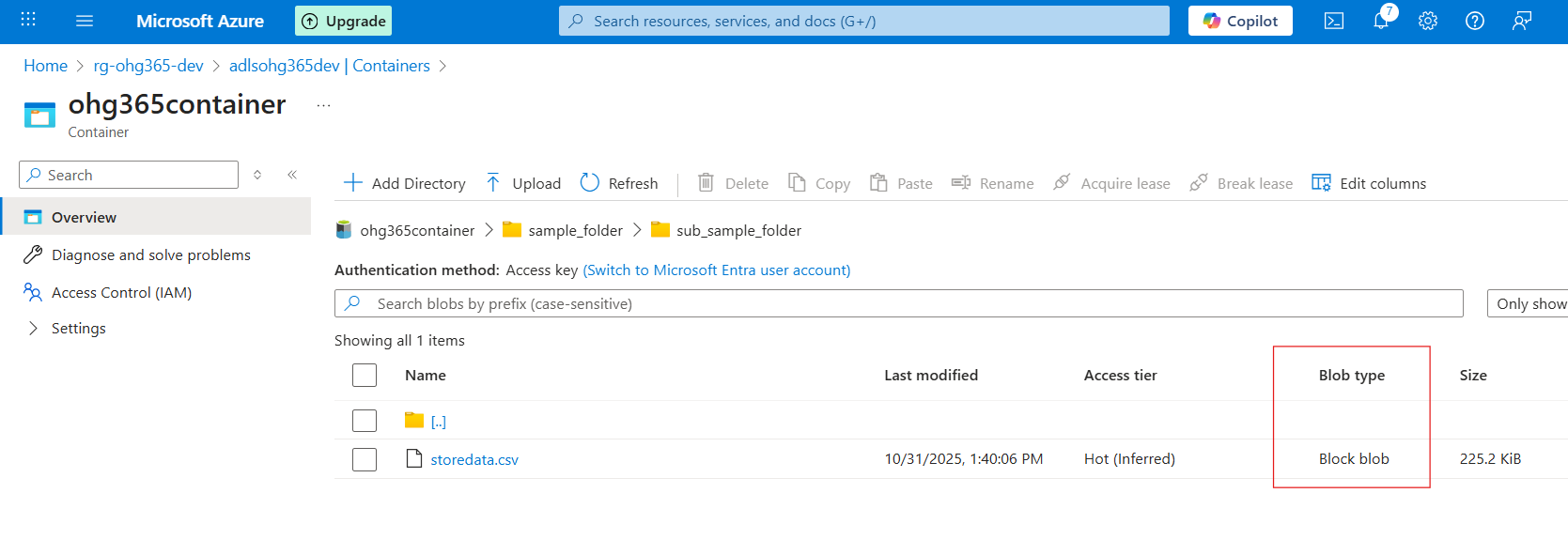


**Types of Blob Types:**



****

****

****

| **Blob Type** | **Best For** | **Example Use** |
| --- | --- | --- |
| 🧱 **Block Blob** | Storing text or binary data | Images, videos, documents, CSVs |
| 📜 **Append Blob** | Data that is constantly added to (append-only) | Logs, telemetry, audit data |
| 📄 **Page Blob** | Random read/write access | Virtual machine disks (VHD files) |

**🧱 1. Block Blob**

**Most common blob type** used in Azure.

**💡 What it is:**

* Stores **text** and **binary** data (files like .txt, .jpg, .mp4, .csv, etc.)
* Data is split into **blocks**, and each block is identified by a block ID.
* You can upload or update blocks individually and commit them together.

**✅ Use Cases:**

* Storing images, videos, PDFs, and backups.
* Data files for analytics (CSV, JSON, Parquet).
* Large files uploaded in chunks.

**📘 Example:**

You upload a 500 MB video file — Azure divides it into smaller **blocks** and uploads each part separately for speed and reliability.

**📜 2. Append Blob**

**Special type of blob for data that grows over time.**

**💡 What it is:**

* Optimized for **append operations** — you can only add new data to the end, not modify or delete existing data.
* Each time you add new information, it’s appended to the blob.

**✅ Use Cases:**

* Storing log files.
* Application telemetry or diagnostics data.
* Streaming data that’s constantly being added.

**📘 Example:**

You’re logging website visits. Each time a new visitor arrives, their data (timestamp, IP, etc.) is appended to the existing log file.

**📄 3. Page Blob**

**Designed for random read/write operations.**

**💡 What it is:**

* Data is stored in **fixed-size 512-byte pages**.
* Allows **fast read and write access** to specific parts of the blob.
* Commonly used for storing **Virtual Hard Disk (VHD)** files that power Azure Virtual Machines.

**✅ Use Cases:**

* Storing Azure VM disks (OS and data disks).
* Large databases that require random access.
* Any workload that reads/writes frequently to specific sections of a file.

**📘 Example:**

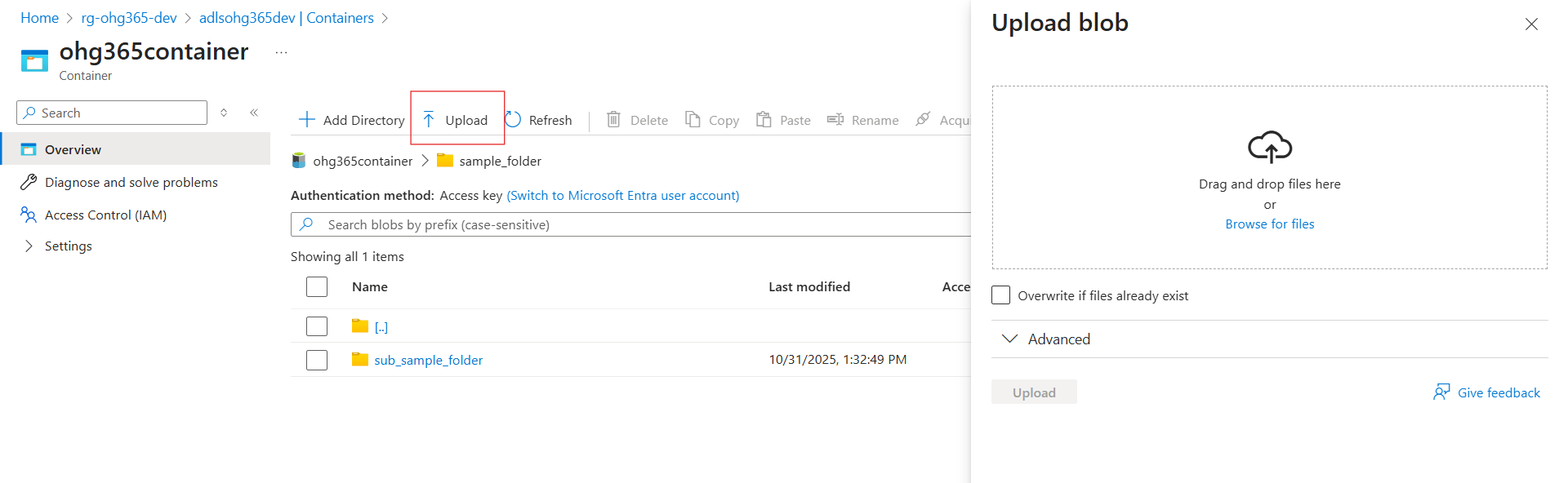
When you start an Azure Virtual Machine, its disk (a .vhd file) is stored as a **Page Blob**, allowing the VM to quickly read or write data anywhere on the disk.

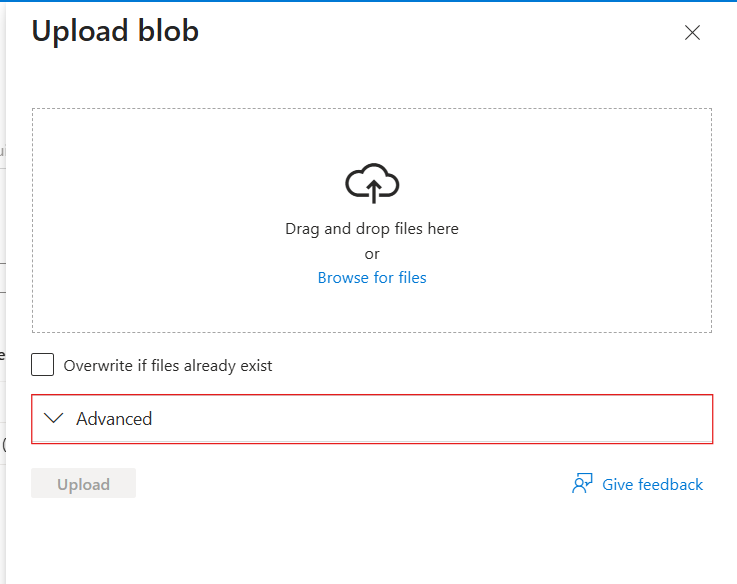
| **Blob Type** | **Structure** | **Read/Write Behavior** | **Common Use** |
| --- | --- | --- | --- |
| **Block Blob** | Data stored as blocks | Upload/replace blocks | Files, media, documents |
| **Append Blob** | Sequentially added blocks | Append-only | Logs, telemetry, streaming data |
| **Page Blob** | Fixed 512-byte pages | Random read/write | VM disks, large databases |

Types of Access Tiers:

Azure lets you store data in **different tiers** based on how often you need it.  
This helps **save money** 💰 by matching storage cost to usage.

| **Tier** | **Cost** | **Availability** | **Best For** |
| --- | --- | --- | --- |
| **Hot** | 💰 Highest cost | 🔥 Always available | Frequently accessed data (e.g., active apps, websites) |
| **Cool** | 💸 Cheaper | 🕓 Slight delay in access | Infrequently accessed data (e.g., monthly reports) |
| **Cold** | 💧 Cheaper than Cool | ⏱️ Slower access | Rarely accessed data but still retrievable |
| **Archive** | 🧊 Cheapest | 💤 Retrieval takes hours | Long-term backups, compliance storage |



****

