In Kubernetes, containers are **ephemeral** — which means:

When a Pod is deleted or recreated (due to scaling, updates, or node failure), any data inside the container is lost.

✅ **Use Case**: You need **persistent storage** for:

* Databases (MySQL, PostgreSQL, MongoDB)
* Logs
* Uploaded files
* Application state

**✅ Step 1: Understand the Core Concepts**

| **Component** | **Description** |
| --- | --- |
| **PV (PersistentVolume)** | A piece of storage in the cluster (e.g., Azure Disk, NFS, localPath). Defined by the admin. |
| **PVC (PersistentVolumeClaim)** | A request for storage by a user (Pod). |
| **StorageClass** (optional) | A blueprint that defines how PVs are dynamically provisioned. |

✅ Create StorageClass → ✅ Create PVC → ✅ Create ConfigMap & Secret → ✅ Create Headless Service → ✅ Deploy StatefulSet

**✅ Step 1: Create a StorageClass (Immediate Provisioning)**

Ensures disk is provisioned as soon as the PVC is created

**🧱 Step-by-Step with Real-World Analogy**

**🔹 Step 1: StorageClass — “Ordering the right kind of disk”**

[**https://kubernetes.io/docs/concepts/storage/storage-classes/**](https://kubernetes.io/docs/concepts/storage/storage-classes/)

apiVersion: storage.k8s.io/v1

kind: StorageClass

metadata:

name: managed-premium-retain-sc

provisioner: disk.csi.azure.com # ✅ CSI driver

reclaimPolicy: Retain # Keeps disk even after PVC deletion

volumeBindingMode: WaitForFirstConsumer # Wait until pod is scheduled

allowVolumeExpansion: true # Allow resizing PVC

parameters:

skuName: Premium\_LRS # Can also use StandardSSD\_LRS, etc.

kind: Managed # ✅ Always use Managed for Azure Disks

**1️⃣ provisioner**

**What it does:**  
Defines the plugin (CSI driver) responsible for creating the disk.

**Common Azure Values:**

| **Provisioner** | **Description** |
| --- | --- |
| disk.csi.azure.com | Azure **Managed Disks** (block) |
| file.csi.azure.com | Azure **Files** (shared SMB storage) |

**Use Case:**

* Use disk.csi.azure.com for **databases**, apps needing persistent block storage.
* Use file.csi.azure.com for **shared access** (e.g., web servers).

**2️⃣ reclaimPolicy**

**What it does:**  
Specifies what happens to the **Azure Disk** or **File Share** when the PVC is deleted.

**Options:**

| **Value** | **What It Does** |
| --- | --- |
| Delete | Delete the disk from Azure |
| Retain | Keep the disk in Azure (for backup or manual recovery) |

**Best Practice:**

* Use Retain if you want to manually recover data.
* Use Delete for ephemeral or scratch storage.

reclaimPolicy: Retain

**3️⃣ volumeBindingMode**

**What it does:**  
Controls **when and where the volume (disk)** gets bound to a pod.

**Options:**

| **Mode** | **Meaning** |
| --- | --- |
| Immediate | Volume is provisioned as soon as PVC is created |
| WaitForFirstConsumer | Volume is only created **when a pod is scheduled**, ensuring it lands in the same **availability zone** |

**Best Practice:**  
Always use WaitForFirstConsumer in **zonal AKS clusters** (prevents zone mismatch).

volumeBindingMode: WaitForFirstConsumer

**4️⃣ allowVolumeExpansion**

**What it does:**  
Enables the ability to **increase PVC size** after creation.

**Options:**

| **Value** | **Meaning** |
| --- | --- |
| true | You can edit the PVC and increase storage |
| false | PVC size is fixed after creation |

**Best Practice:**  
Set to true for databases or dynamic workloads.

**5️⃣ parameters.storageaccounttype**

**What it does:**  
Specifies the **performance tier/type** of Azure Disk/File to use.

**🔹 For disk.csi.azure.com (Azure Disks)**

| **skuName** | **Description** |
| --- | --- |
| Standard\_LRS | HDD (cheapest) |
| StandardSSD\_LRS | Standard SSD |
| Premium\_LRS | Premium SSD |
| UltraSSD\_LRS | Ultra-fast SSD (special use cases) |
| Premium\_ZRS | Premium SSD with zone-redundancy |

🔹 For file.csi.azure.com (Azure Files)

skuName Description

Standard\_LRS Standard performance

Premium\_LRS High-performance File share

6️⃣ parameters.kind

What it does:

Defines the type of Azure disk to use.

Value Description

Managed Use Azure Managed Disks (recommended)

Best Practice:

Always set to Managed — unmanaged disks are deprecated.

Example:

parameters:

kind: Managed

===Final Examples====

**Create and use a volume with Azure Disks in Azure Kubernetes Service (AKS)**

**🔹 Step 2: PVC (PersistentVolumeClaim) — “Making a storage request”**

Now you say:

"Hey Kubernetes, I need 5Gi of Premium SSD storage to save PostgreSQL database data!"

**A PVC (PersistentVolumeClaim)** is a Kubernetes object used by an application to **request storage**.

Kubernetes will then **find or create a PersistentVolume (PV)** that matches the request. In AKS, this usually results in **creating an Azure Disk dynamically** using a StorageClass.

apiVersion: v1

kind: PersistentVolumeClaim

metadata:

name: postgres-pvc

spec:

accessModes:

- ReadWriteOnce

resources:

requests:

storage: 5Gi

storageClassName: azure-disk-premium

**✅ apiVersion: v1**

* The version of the Kubernetes API for this resource.
* PVC is part of the **core API group**, so it uses v1.

**✅ kind: PersistentVolumeClaim**

* Tells Kubernetes: this is a **PVC object**, not a Pod or Service.

**✅ metadata.name: azure-managed-disk-pvc**

* The name you assign to the PVC.
* You’ll use this name in your pod to mount the disk.

**✅ spec.accessModes**

This tells Kubernetes how the storage will be accessed by the pod.

| **Value** | **Meaning** | **Azure Support?** |
| --- | --- | --- |
| ReadWriteOnce | Only one node/pod can read/write at a time | ✅ Yes (Azure Disk) |
| ReadOnlyMany | Many pods can read, but not write | ❌ Not supported on Azure Disk |
| ReadWriteMany | Many pods can read/write concurrently | ✅ Yes (with Azure Files) |

**Use case for ReadWriteOnce**: Databases (e.g., PostgreSQL, MySQL)

**✅ storageClassName: managed-premium-retain-sc**

This tells Kubernetes:

“Provision the storage based on the rules in the StorageClass named managed-premium-retain-sc.”

**✅ resources.requests.storage: 5Gi**

You’re asking for **5 GiB of storage**.

💡 Note:

* You can later expand this size (if allowVolumeExpansion: true is set in the StorageClass).
* Make sure the requested size aligns with your workload (e.g., logs, databases, uploads).
* **✅ Final Summary Table**

| **Field** | **Purpose** | **Example / Notes** |
| --- | --- | --- |
| accessModes | How the volume can be used | ReadWriteOnce for Azure Disks |
| storageClassName | Which policy/template to use for disk | Points to a StorageClass |
| resources.requests.storage | Size of disk | 5Gi, 10Gi, etc. |
| volumeMode | Format of volume | Filesystem (default), or Block |
| selector | Match a manually created PV | Only used with **static provisioning** |
| dataSource | Clone from snapshot or existing volume | Advanced use case – for backups/clones |

apiVersion: v1

kind: Pod

metadata:

name: nginx-on-azure-disk

spec:

containers:

- name: nginx

image: nginx:latest

ports:

- containerPort: 80

volumeMounts:

- mountPath: /mnt/azure

name: disk-storage

volumes:

- name: disk-storage

persistentVolumeClaim:

claimName: azure-managed-disk-pvc

kubectl get pod nginx-on-azure-disk

kubectl get pvc azure-managed-disk-pvc

kubectl get pv

✅ 1. **Verify Data Persistence After Deleting the Pod**

This ensures that your data stored on the **Azure Disk** is not lost even if the pod is deleted (since it’s stored outside the pod on a persistent volume).

**🔧 Step-by-Step**

**1️⃣ Exec into the pod and write a file**

kubectl exec -it nginx-on-azure-disk -- /bin/bash

echo "Hello from Azure Disk" > /mnt/azure/test.txt

cat /mnt/azure/test.txt

exit

kubectl delete pod nginx-on-azure-disk

3️⃣ Recreate the pod using the same PVC:

kubectl apply -f nginx-on-azure-disk.yaml

kubectl exec -it nginx-on-azure-disk -- cat /mnt/azure/test.txt

✅ You should still see:

Hello from Azure Disk