AKS Storage - Storage Classes, Persistent Volume Claims

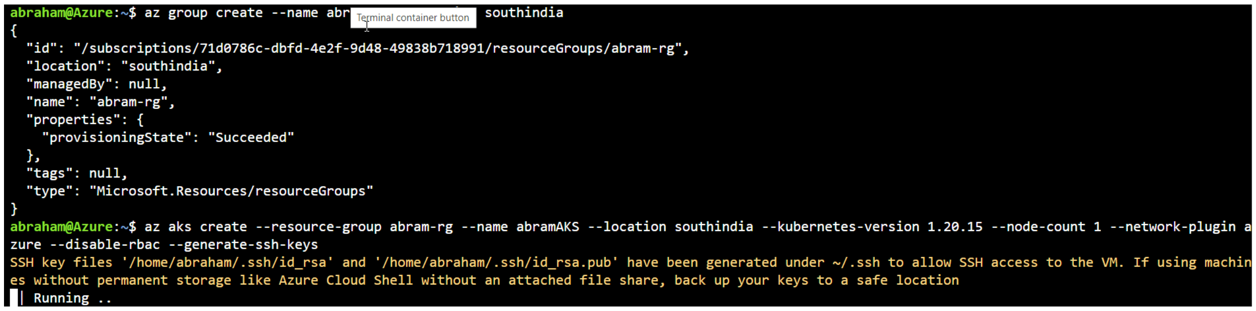
Step-01: Introduction

- We are going to create a MySQL Database with persistence storage using Azure Disks

Kubernetes Object	YAML File
Storage Class	01-storage-class.yml
Persistent Volume Claim	02-persistent-volume-claim.yml
Config Map	03-UserManagement-ConfigMap.yml
Deployment, Environment Variables, Volumes, VolumeMounts	04-mysql-deployment.yml
ClusterIP Service	05-mysql-clusterip-service.yml

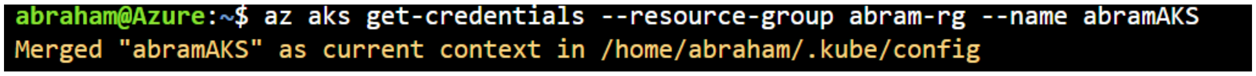
Create a AKS cluster

```
#abraham@Azure:~$ az group create --name abram-rg --location southindia
#abraham@Azure:~$ az aks create --resource-group abram-rg --name abramAKS --location southindia --kubernetes-version 1.20.15 --node-count 1 --network-plugin azure --disable-rbac --generate-ssh-keys
```



```
abraham@Azure:~$ az group create --name abr                southindia
{
  "id": "/subscriptions/71d0786c-dbfd-4e2f-9d48-49838b718991/resourceGroups/abram-rg",
  "location": "southindia",
  "managedBy": null,
  "name": "abram-rg",
  "properties": {
    "provisioningState": "Succeeded"
  },
  "tags": null,
  "type": "Microsoft.Resources/resourceGroups"
}
abraham@Azure:~$ az aks create --resource-group abram-rg --name abramAKS --location southindia --kubernetes-version 1.20.15 --node-count 1 --network-plugin a
zure --disable-rbac --generate-ssh-keys
SSH key files '/home/abraham/.ssh/id_rsa' and '/home/abraham/.ssh/id_rsa.pub' have been generated under ~/.ssh to allow SSH access to the VM. If using machin
es without permanent storage like Azure Cloud Shell without an attached file share, back up your keys to a safe location
 | Running ..
```

# az aks get-credentials --resource-group abram-rg --name abramAKS



```
abraham@Azure:~$ az aks get-credentials --resource-group abram-rg --name abramAKS
Merged "abramAKS" as current context in /home/abraham/.kube/config
```

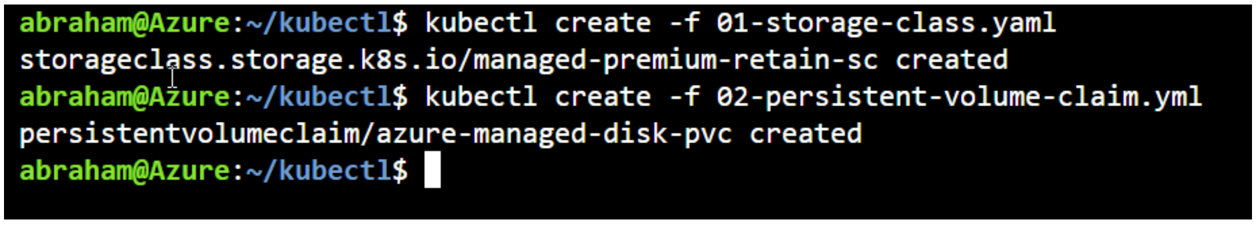
Step-02: Create following Kubernetes manifests

Create Storage Class manifest

```
abraham@Azure:~/kubectl$ ls
01-storage-class.yaml  02-persistent-volume-claim.yml  03-UserManagement-ConfigMap.yml
04-mysql-deployment.yml  05-mysql-clusterip-service.yml

abraham@Azure:~/kubectl$ cat 01-storage-class.yaml
apiVersion: storage.k8s.io/v1
kind: StorageClass
metadata:
  name: managed-premium-retain-sc
provisioner: kubernetes.io/azure-disk
reclaimPolicy: Retain  # Default is Delete, recommended is retain
volumeBindingMode: WaitForFirstConsumer # Default is Immediate, recommended is
WaitForFirstConsumer
allowVolumeExpansion: true
parameters:
  storageaccounttype: Premium_LRS # or we can use Standard_LRS
  kind: managed # Default is shared (Other two are managed and dedicated)
abraham@Azure:~/kubectl$

abraham@Azure:~/kubectl$ cat 02-persistent-volume-claim.yml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: azure-managed-disk-pvc
```

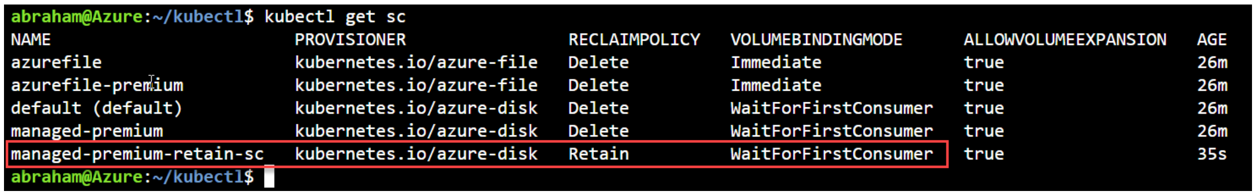
```
spec:
  accessModes:
  - ReadWriteOnce
  storageClassName: managed-premium-retain-sc
  resources:
    requests:
      storage: 5Gi
abraham@Azure:~/kubectl$


abraham@Azure:~/kubectl$ cat 03-UserManagement-ConfigMap.yml
apiVersion: v1
kind: ConfigMap
metadata:
  name: usermanagement-dbcreation-script
data:
  mysql_usermgmt.sql: |-
    DROP DATABASE IF EXISTS webappdb;
    CREATE DATABASE webappdb;
abraham@Azure:~/kubectl$

abraham@Azure:~/kubectl$ cat 04-mysql-deployment.yml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: mysql
spec:
  replicas: 1
  selector:
    matchLabels:
      app: mysql
  strategy:
    type: Recreate
  template:
    metadata:
      labels:
        app: mysql
    spec:
      containers:
        - name: mysql
          image: mysql:5.6
          env:
            - name: MYSQL_ROOT_PASSWORD
              value: dbpassword11
          ports:
            - containerPort: 3306
              name: mysql
          volumeMounts:
            - name: mysql-persistent-storage
              mountPath: /var/lib/mysql
            - name: usermanagement-dbcreation-script
              mountPath: /docker-entrypoint-initdb.d #https://hub.docker.com/_/mysql Refer
Initializing a fresh instance
      volumes:
        - name: mysql-persistent-storage
          persistentVolumeClaim:
            claimName: azure-managed-disk-pvc
        - name: usermanagement-dbcreation-script
          configMap:
            name: usermanagement-dbcreation-script
abraham@Azure:~/kubectl$

abraham@Azure:~/kubectl$ cat 05-mysql-clusterip-service.yml
apiVersion: v1
kind: Service
metadata:
  name: mysql
spec:
  selector:
    app: mysql
  ports:
    - port: 3306
  clusterIP: None # This means we are going to use Pod IP
abraham@Azure:~/kubectl$
```

```
# Create Storage Class & PVC
kubectl apply -f kube-manifests/01-storage-class.yml
kubectl apply -f kube-manifests/02-persistent-volume-claim.yml
```

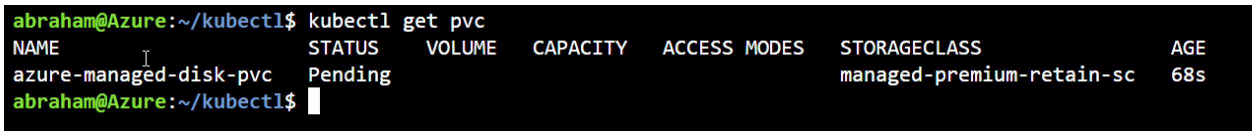


```
abraham@Azure:~/kubectl$ kubectl create -f 01-storage-class.yaml
storageclass.storage.k8s.io/managed-premium-retain-sc created
abraham@Azure:~/kubectl$ kubectl create -f 02-persistent-volume-claim.yml
persistentvolumeclaim/azure-managed-disk-pvc created
abraham@Azure:~/kubectl$
```

```
# List Storage Classes
kubectl get sc
```


```
abraham@Azure:~/kubectl$ kubectl get sc
NAME                      PROVISIONER                RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
azurefile                 kubernetes.io/azure-file   Delete          Immediate              true                   26m
azurefile-premium         kubernetes.io/azure-file   Delete          Immediate              true                   26m
default (default)         kubernetes.io/azure-disk   Delete          WaitForFirstConsumer   true                   26m
managed-premium           kubernetes.io/azure-disk   Delete          WaitForFirstConsumer   true                   26m
managed-premium-retain-sc kubernetes.io/azure-disk   Retain          WaitForFirstConsumer   true                   35s
abraham@Azure:~/kubectl$
```

```
# List PVC
kubectl get pvc
```


```
abraham@Azure:~/kubectl$ kubectl get pvc
NAME                     STATUS    VOLUME   CAPACITY   ACCESS MODES   STORAGECLASS                AGE
azure-managed-disk-pvc   Pending                                      managed-premium-retain-sc   68s
abraham@Azure:~/kubectl$
```

```
# List PV
kubectl get pv
```

Create ConfigMap manifest

- **We are going to create a `usermgmt` database schema during the mysql pod creation time which we will leverage when we deploy User Management Microservice.**

Create MySQL Deployment manifest

- **Environment Variables**
- **Volumes**
- **Volume Mounts**

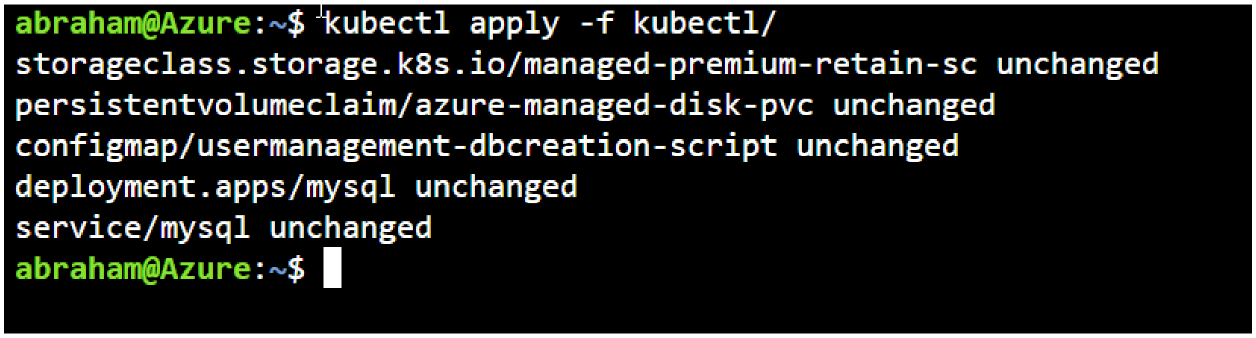
Create MySQL ClusterIP Service manifest

- **At any point of time we are going to have only one mysql pod in this design so `ClusterIP: None` will use the `Pod IP Address` instead of creating or allocating a separate IP for `MySQL Cluster IP service`.**

```
NOte : cat 03-UserManagement-ConfigMap.yml
```

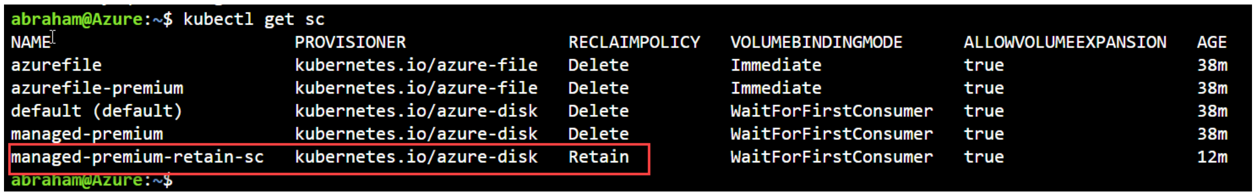
Step-03: Create MySQL Database with all above manifests

```
# Create MySQL Database
kubectl apply -f kube-manifests/
```



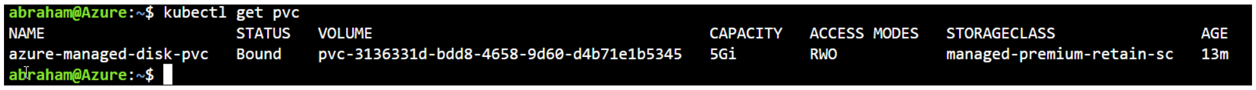
```
abraham@Azure:~$ kubectl apply -f kubectl/
storageclass.storage.k8s.io/managed-premium-retain-sc unchanged
persistentvolumeclaim/azure-managed-disk-pvc unchanged
configmap/usermanagement-dbcreation-script unchanged
deployment.apps/mysql unchanged
service/mysql unchanged
abraham@Azure:~$
```

```
# List Storage Classes
kubectl get sc
```



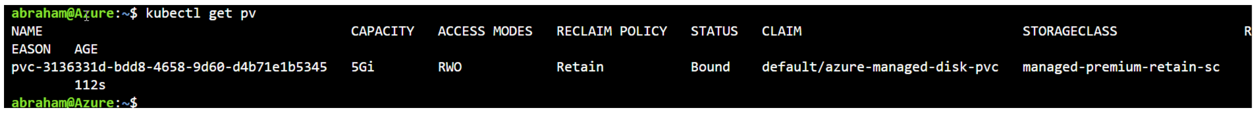
```
abraham@Azure:~$ kubectl get sc
NAME                        PROVISIONER                 RECLAIMPOLICY   VOLUMEBINDINGMODE      ALLOWVOLUMEEXPANSION   AGE
azurefile                   kubernetes.io/azure-file    Delete          Immediate              true                   38m
azurefile-premium           kubernetes.io/azure-file    Delete          Immediate              true                   38m
default (default)           kubernetes.io/azure-disk    Delete          WaitForFirstConsumer   true                   38m
managed-premium             kubernetes.io/azure-disk    Delete          WaitForFirstConsumer   true                   38m
managed-premium-retain-sc   kubernetes.io/azure-disk    Retain          WaitForFirstConsumer   true                   12m
abraham@Azure:~$
```

```
# List PVC
kubectl get pvc
```



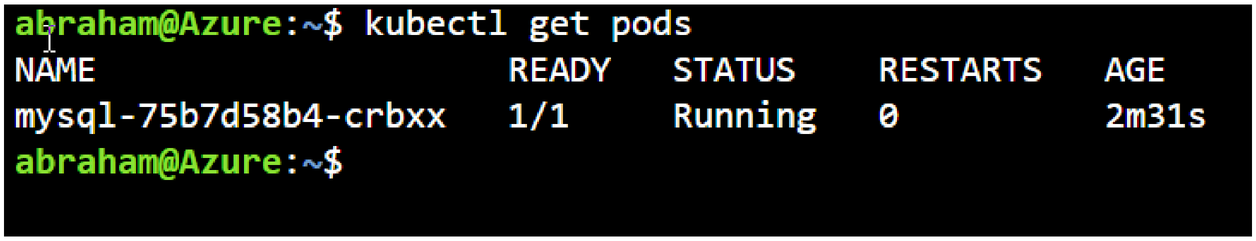
```
abraham@Azure:~$ kubectl get pvc
NAME                     STATUS   VOLUME                                     CAPACITY   ACCESS MODES   STORAGECLASS                AGE
azure-managed-disk-pvc   Bound    pvc-3136331d-bdd8-4658-9d60-d4b71e1b5345   5Gi        RWO            managed-premium-retain-sc   13m
abraham@Azure:~$
```

```
# List PV
kubectl get pv
```

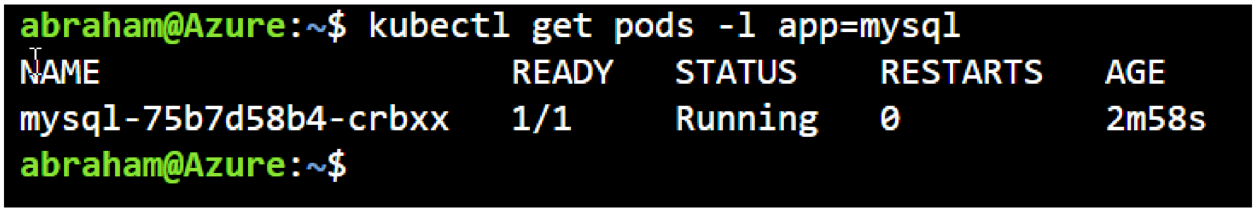


```
abraham@Azure:~$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS   CLAIM                            STORAGECLASS                R
EASON   AGE
pvc-3136331d-bdd8-4658-9d60-d4b71e1b5345   5Gi        RWO            Retain           Bound    default/azure-managed-disk-pvc   managed-premium-retain-sc
        112s
abraham@Azure:~$
```

```
# List pods
kubectl get pods
```


```
abraham@Azure:~$ kubectl get pods
NAME                       READY   STATUS    RESTARTS   AGE
mysql-75b7d58b4-crbxx      1/1     Running   0          2m31s
abraham@Azure:~$
```

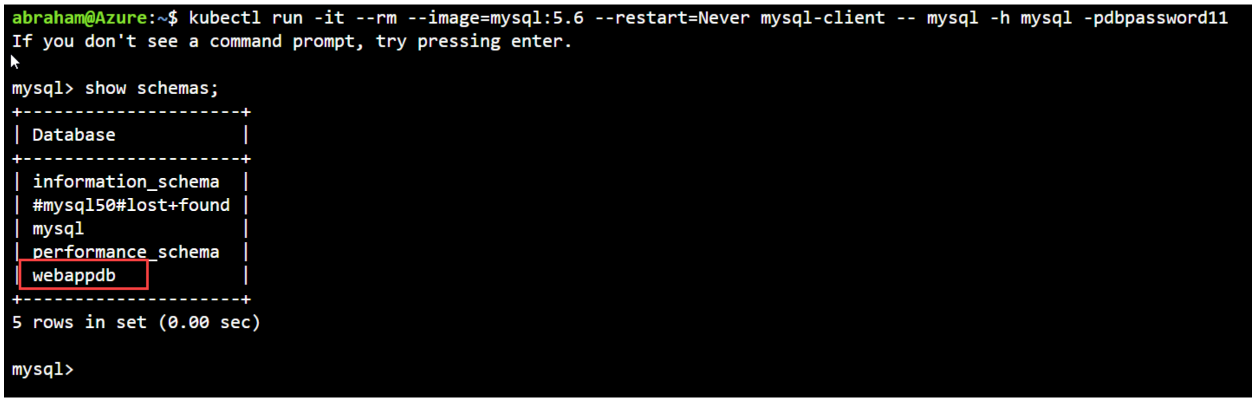
```
# List pods based on  label name
kubectl get pods -l app=mysql
```


```
abraham@Azure:~$ kubectl get pods -l app=mysql
NAME                       READY   STATUS    RESTARTS   AGE
mysql-75b7d58b4-crbxx      1/1     Running   0          2m58s
abraham@Azure:~$
```

abraham@Azure:~$ kubectl logs -f mysql-75b7d58b4-crbxx

Step-04: Connect to MySQL Database

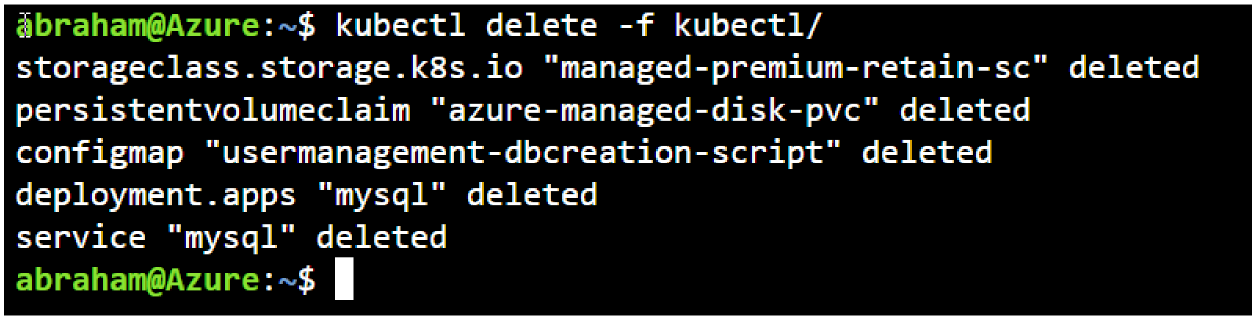
```
# Connect to MYSQL Database
kubectl run -it --rm --image=mysql:5.6 --restart=Never mysql-client -- mysql -h
mysql -pdbpassword11
```


```
abraham@Azure:~$ kubectl run -it --rm --image=mysql:5.6 --restart=Never mysql-client -- mysql -h mysql -pdbpassword11
If you don't see a command prompt, try pressing enter.

mysql> show schemas;
+---------------------+
| Database            |
+---------------------+
| information_schema  |
| #mysql50#lost+found |
| mysql               |
| performance_schema  |
| webappdb            |
+---------------------+
5 rows in set (0.00 sec)

mysql>
```

```
# Verify usermgmt schema got created which we provided in ConfigMap
mysql> show schemas;
```

Step-05: Clean-Up

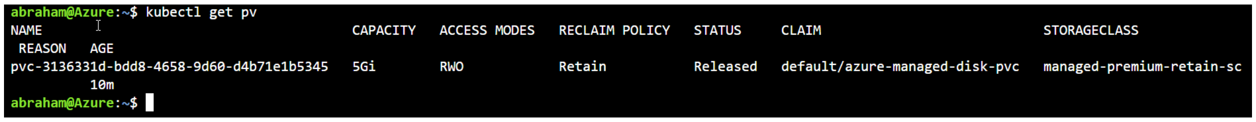
```
# Delete All
kubectl delete -f kube-manifests/
```



```
abraham@Azure:~$ kubectl delete -f kubectl/
storageclass.storage.k8s.io "managed-premium-retain-sc" deleted
persistentvolumeclaim "azure-managed-disk-pvc" deleted
configmap "usermanagement-dbcreation-script" deleted
deployment.apps "mysql" deleted
service "mysql" deleted
abraham@Azure:~$
```

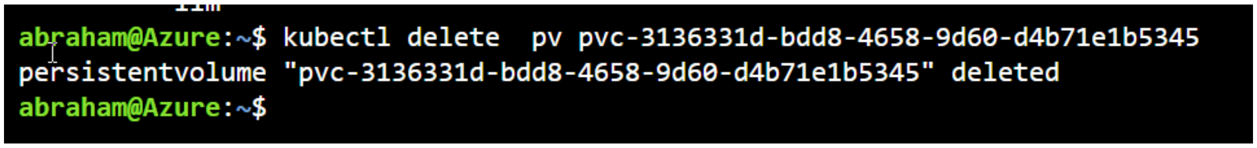
Step-06: Delete PV exclusively - It exists due to retain policy

```
# List PV
kubectl get pv
```



```
abraham@Azure:~$ kubectl get pv
NAME                                       CAPACITY   ACCESS MODES   RECLAIM POLICY   STATUS     CLAIM                            STORAGECLASS
 REASON   AGE
pvc-3136331d-bdd8-4658-9d60-d4b71e1b5345   5Gi        RWO            Retain           Released   default/azure-managed-disk-pvc   managed-premium-retain-sc
          10m
abraham@Azure:~$
```

```
# Delete PV exclusively
kubectl get pv
kubectl delete pv <PV-NAME>
```



```
abraham@Azure:~$ kubectl delete  pv pvc-3136331d-bdd8-4658-9d60-d4b71e1b5345
persistentvolume "pvc-3136331d-bdd8-4658-9d60-d4b71e1b5345" deleted
abraham@Azure:~$
```

```
# Delete Azure Disks
Go to All Services -> Disks -> Select and Delete the Disk
```