

1.

## Deploy Postgres database using PVC & PV cluster

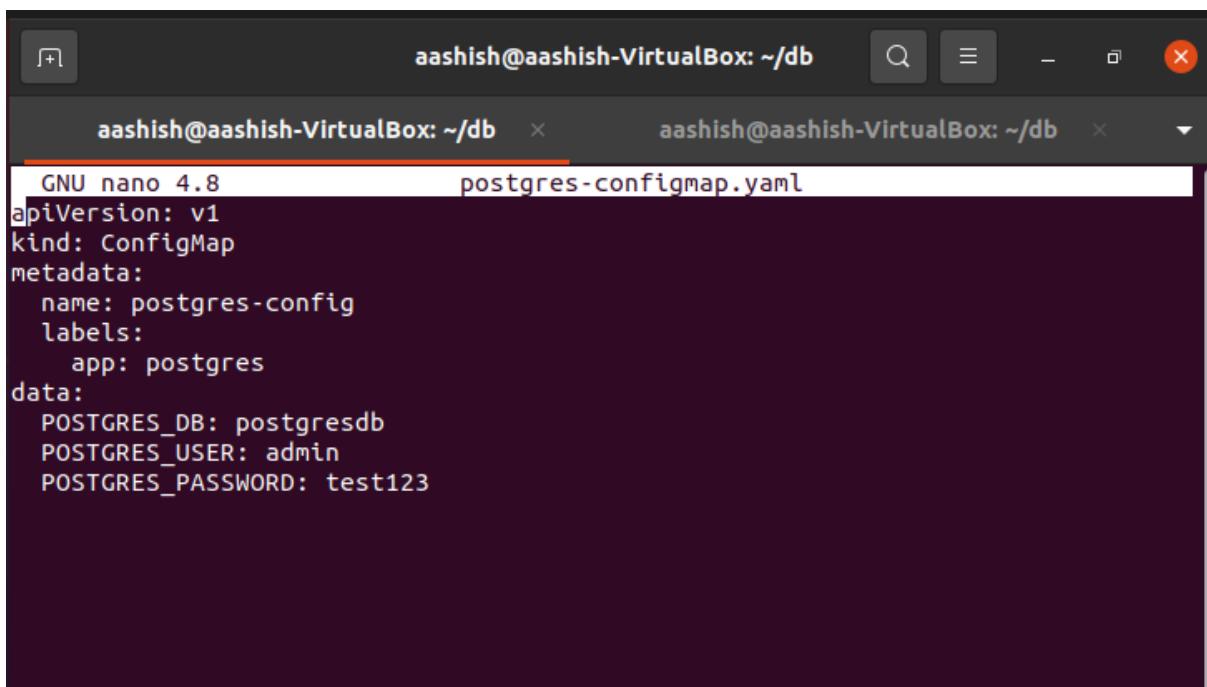
### Answer:

I have created a directory named db to store all the files for Postgres database.

The ConfigMap resource contains the data that is used during the deployment process.

So, first of all, we create a ConfigMap YAML file named **postgres-configmap.yaml** in a text editor as follows;

```
- sudo nano postgres-configmap.yaml
```



The screenshot shows a terminal window titled "aashish@aashish-VirtualBox: ~/db". It has two tabs open, both labeled "aashish@aashish-VirtualBox: ~/db". The current tab is active and displays the following YAML code:

```
GNU nano 4.8          postgres-configmap.yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: postgres-config
  labels:
    app: postgres
data:
  POSTGRES_DB: postgresdb
  POSTGRES_USER: admin
  POSTGRES_PASSWORD: test123
```

We save the file and exit. Then apply the resource with kubectl:

```
- sudo kubectl apply -f postgres-configmap.yaml
```

```
aashish@aashish-VirtualBox:~/db$ sudo kubectl apply -f postgres-configmap.yaml
configmap/postgres-config created
```

Next, we create a YAML file for storage configuration named **postgres-storage.yaml** using a text editor. Here, I have placed both PV and PVC configurations in one file but we can do it separately as well. The **postgres-storage.yaml** file is given below;

```
aashish@aashish-VirtualBox: ~/db
```

```
GNU nano 4.8          postgres-storage.yaml
```

```
kind: PersistentVolume
apiVersion: v1
metadata:
  name: postgres-pv-volume
  labels:
    type: local
    app: postgres
spec:
  storageClassName: manual
  capacity:
    storage: 5Gi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: "/mnt/data"
---
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: postgres-pv-claim
  labels:
    app: postgres
```

```
kind: PersistentVolumeClaim
apiVersion: v1
metadata:
  name: postgres-pv-claim
  labels:
    app: postgres
spec:
  storageClassName: manual
  accessModes:
    - ReadWriteMany
  resources:
    requests:
      storage: 5Gi
```

```
^G Get Help  ^O Write Out  ^W Where Is  ^K Cut Text  ^J Justify
^X Exit       ^R Read File   ^\ Replace   ^U Paste Text ^T To Spell
```

We save the file and exit. Then, apply the resources as follows:

- **sudo kubectl apply -f postgres-storage.yaml**

```
aashish@aashish-VirtualBox:~/db$ sudo kubectl apply -f postgres-storage.yaml
persistentvolume/postgres-pv-volume created
persistentvolumeclaim/postgres-pv-claim created
```

To check the PVC connection with PV, we use following command;

- sudo kubectl get pvc

```
aashish@aashish-VirtualBox:~/db$ sudo kubectl get pvc
NAME           STATUS    VOLUME          CAPACITY   ACCESS MODES   STO
RAGECLASS     AGE
postgres-pv-claim  Bound    postgres-pv-volume  5Gi        RWX          man
ual           14m
```

The status of the PVC is Bound, and the PVC is ready to be used in the PostgreSQL deployment.

Now, we create a deployment YAML file named `postgres-deployment.yaml` as follows;

- sudo nano postgres-deployment.yaml

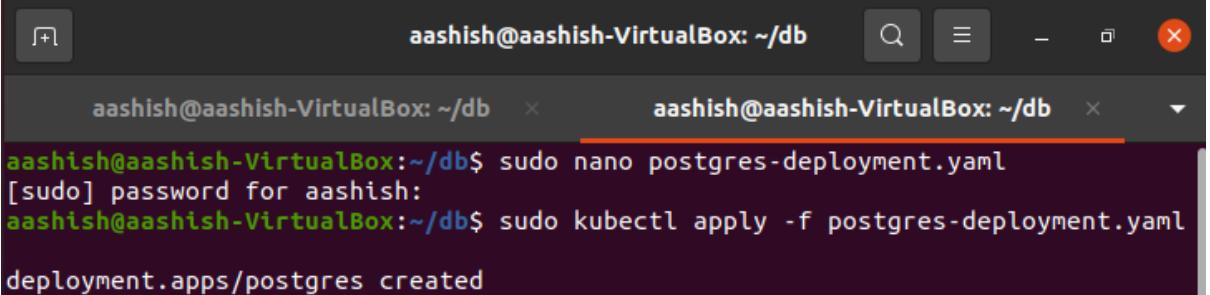
aashish@aashish-VirtualBox: ~/db

GNU nano 4.8                    postgres-deployment.yaml

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: postgres
spec:
  replicas: 1
  selector:
    matchLabels:
      app: postgres
  template:
    metadata:
      labels:
        app: postgres
    spec:
      containers:
        - name: postgres
          image: postgres:10.1
          imagePullPolicy: "IfNotPresent"
          ports:
            - containerPort: 5432
              containerPort: 5432
          envFrom:
            - configMapRef:
                name: postgres-config
          volumeMounts:
            - mountPath: /var/lib/postgresql/data
              name: postgredb
      volumes:
        - name: postgredb
          persistentVolumeClaim:
            claimName: postgres-pv-claim
```

We save the file and exit. Then apply the deployment as follows;

- **sudo kubectl apply -f postgres-deployment.yaml**



The screenshot shows a terminal window with two tabs. The current tab is titled 'aashish@aashish-VirtualBox: ~/db' and contains the command: 'aashish@aashish-VirtualBox:~/db\$ sudo kubectl apply -f postgres-deployment.yaml'. Below this, the output shows: 'deployment.apps/postgres created'. The previous tab is also titled 'aashish@aashish-VirtualBox: ~/db' and shows the command 'sudo nano postgres-deployment.yaml' being run.

```
aashish@aashish-VirtualBox:~/db$ sudo kubectl apply -f postgres-deployment.yaml
[sudo] password for aashish:
aashish@aashish-VirtualBox:~/db$ deployment.apps/postgres created
```

Hence, the Postgres database using PV and PVC has been deployed successfully.