**Description**

You can use Azure Database Migration Service to migrate the databases from a SQL Server instance to Arc Enabled SQL Managed Instance. In this tutorial, you migrate the [Sample](https://github.com/MicrosoftDocs/azure-docs/blob/main/sql/samples/adventureworks-install-configure#download-backup-files) (Adventure Works) database from SQL Server 2019 to Arc Enabled SQL Server Manged Instance by using Azure Data Studio

In this Lab, you will learn how to:

* Create Kubernates cluster
* Create Data Controller
* Create Arc Enabled SQL Managed Instance
* Use the Azure Data Studio to migrate the database
* Run the migration.
* Monitor the migration.

**Learning Objectives**

Upon completion of this Lab, you will be able to:

* Migrate Azure SQL Server to Azure Arc Enabled SQL Managed Instance (IaaS to PaaS)
* Connect to an Azure SQL database and SQL server through Azure CLI

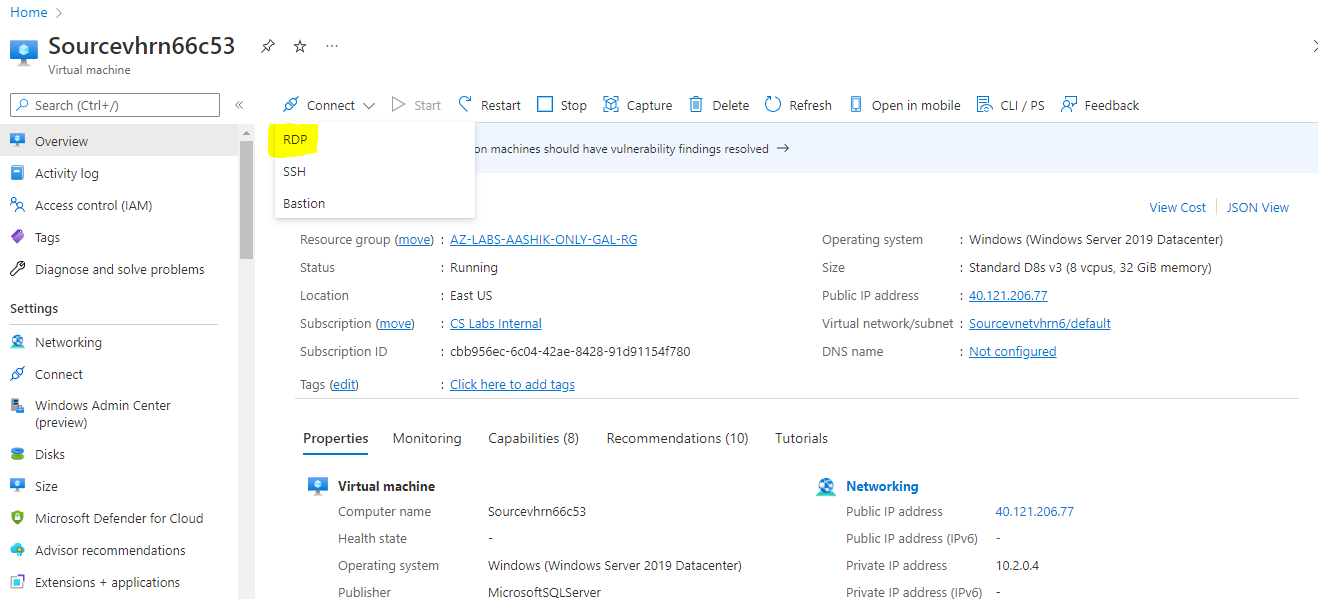
**Intended Audience**

This lab is intended for:

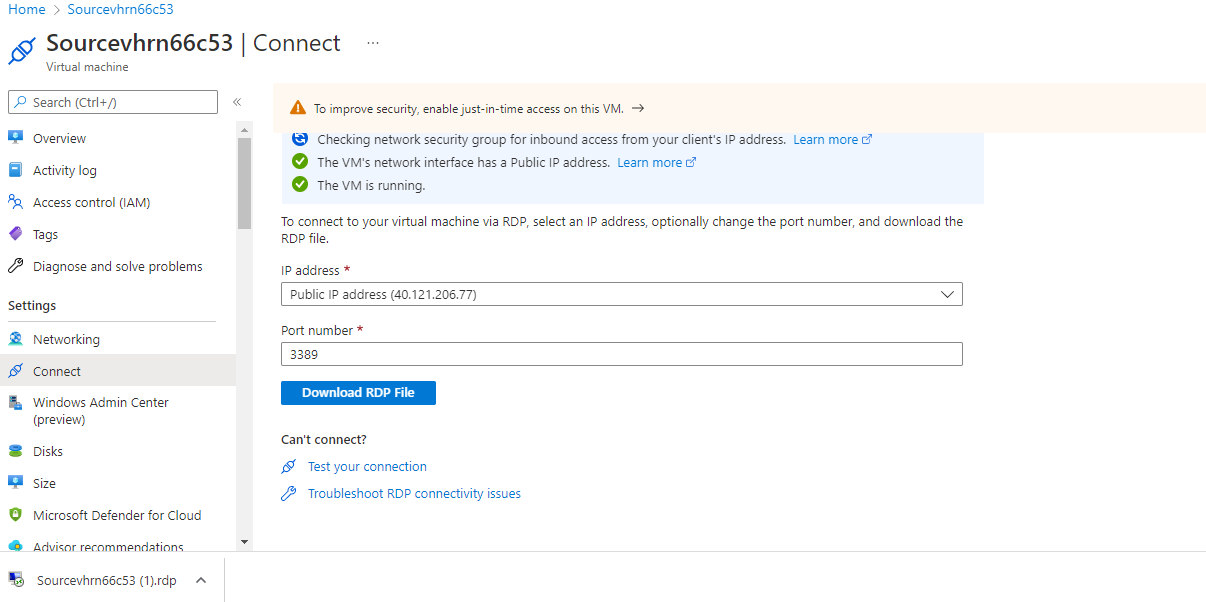
* Microsoft CSA Team members

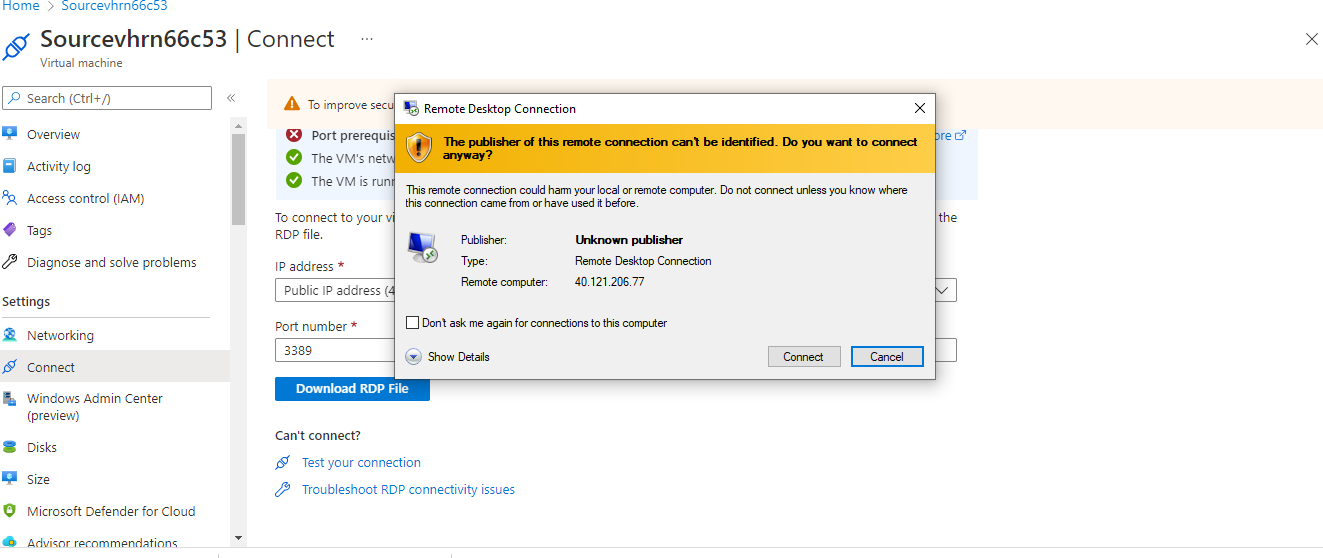
**Steps to carry out migration activity**

Login to Source VM by **RDP**

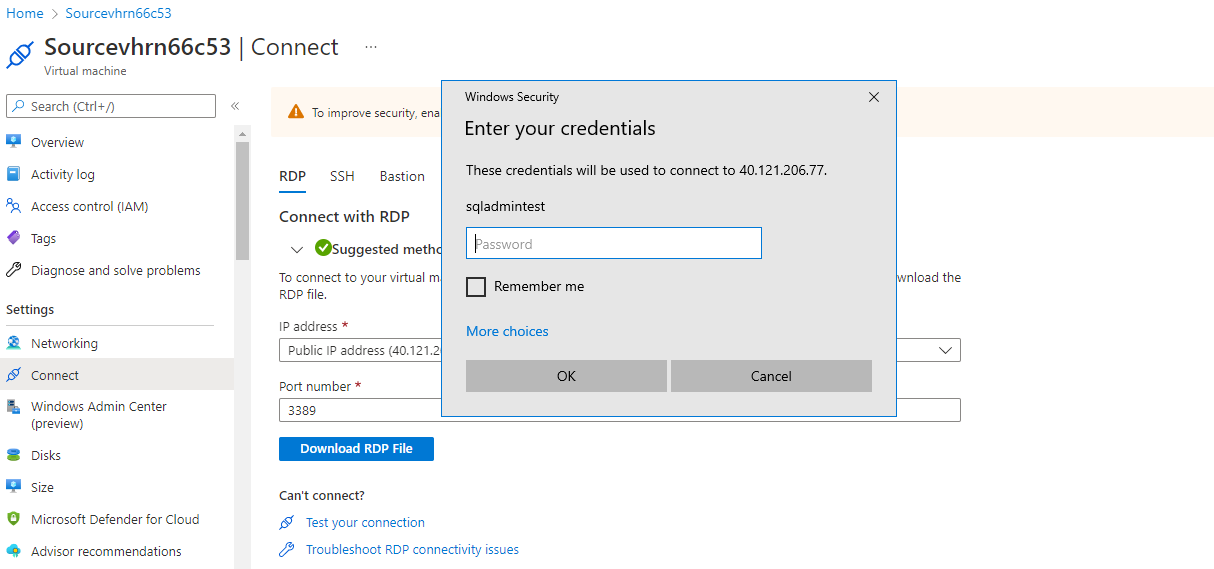


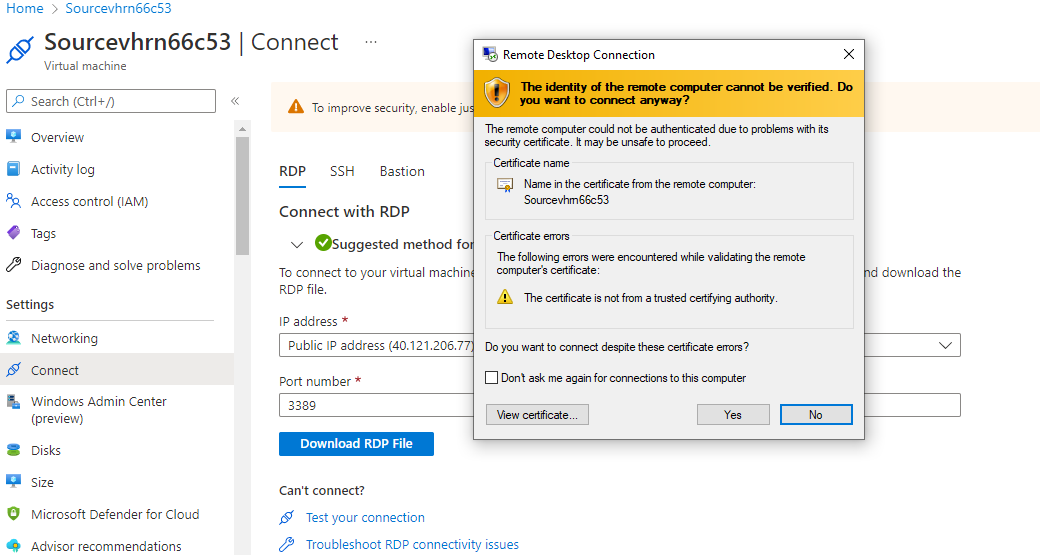
Select **Download RDP**





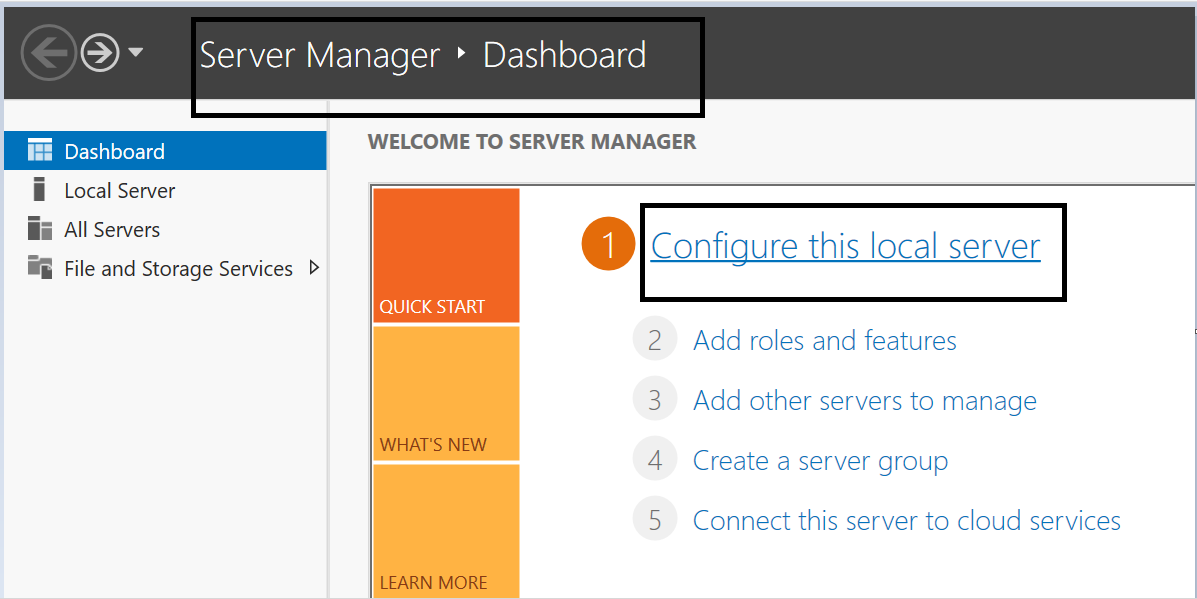
Provide **SQL Username**: sqladmin and **Password: Password@123**





**After Login to SQL Source VM disable IE Enhanced Security Configuration**

Search for Server Manager, Go to Dashboard and choose Configure this local server.



In properties pane navigate to IE enhanced security configuration and click on **On**.

Graphical user interface, text, application

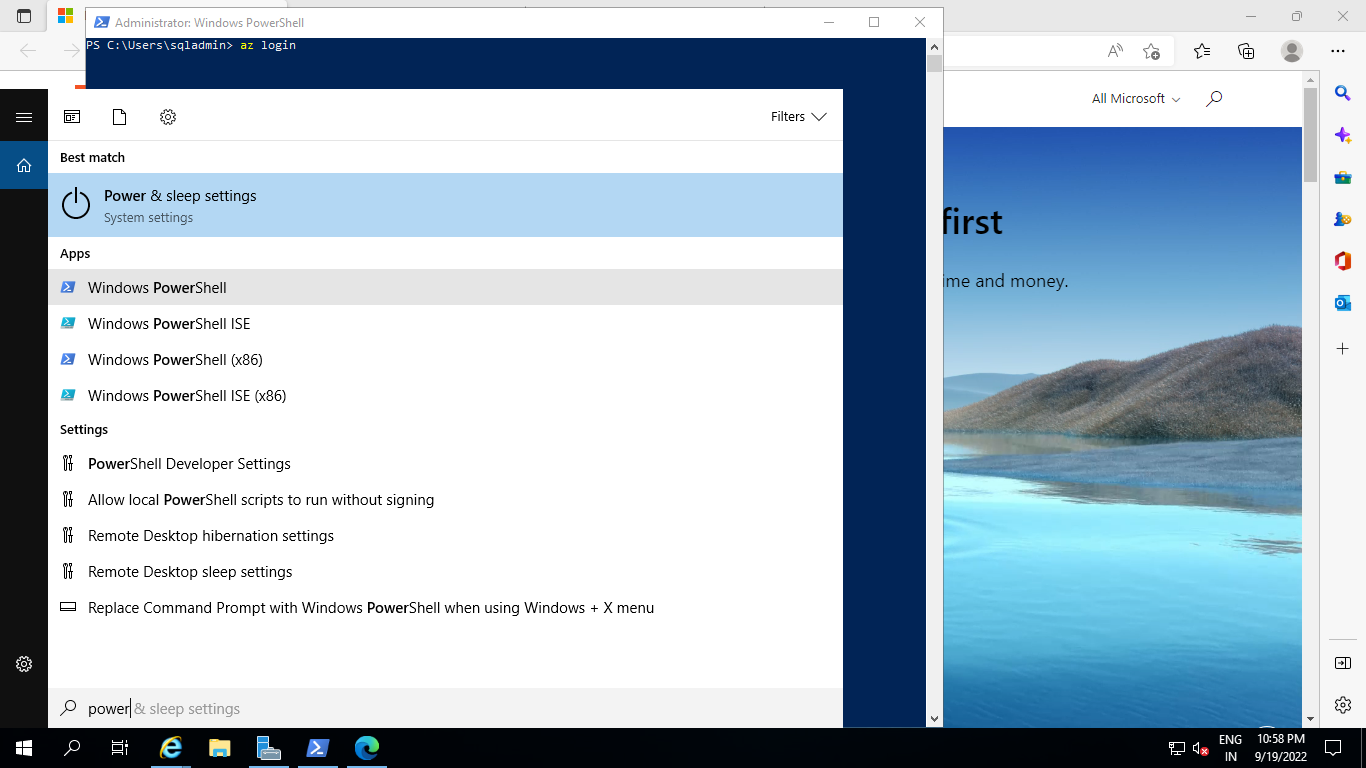
Description automatically generated

Choose **Off** in both options and click on save. (These steps help in seamless login to azure subscription through Azure data studios).

Graphical user interface, text, application, chat or text message

Description automatically generated

Search for Windows PowerShell



Type az login



Provide Microsoft Azure credentials to login Azure portal

Copy output to notepad for future reference for subscription id details**:**

Example

{

"cloudName": "AzureCloud",

"homeTenantId": "72f988bf-86f1-41af-91ab-2d7cd011db47",

"id": "cbb956ec-6c04-42ae-8428-91d91154f780",

"isDefault": false,

"managedByTenants": [],

"name": "CS Labs Internal",

"state": "Enabled",

"tenantId": "72f988bf-86f1-41af-91ab-2d7cd011db47",

"user": {

"name": "v-aashikma@microsoft.com",

"type": "user"

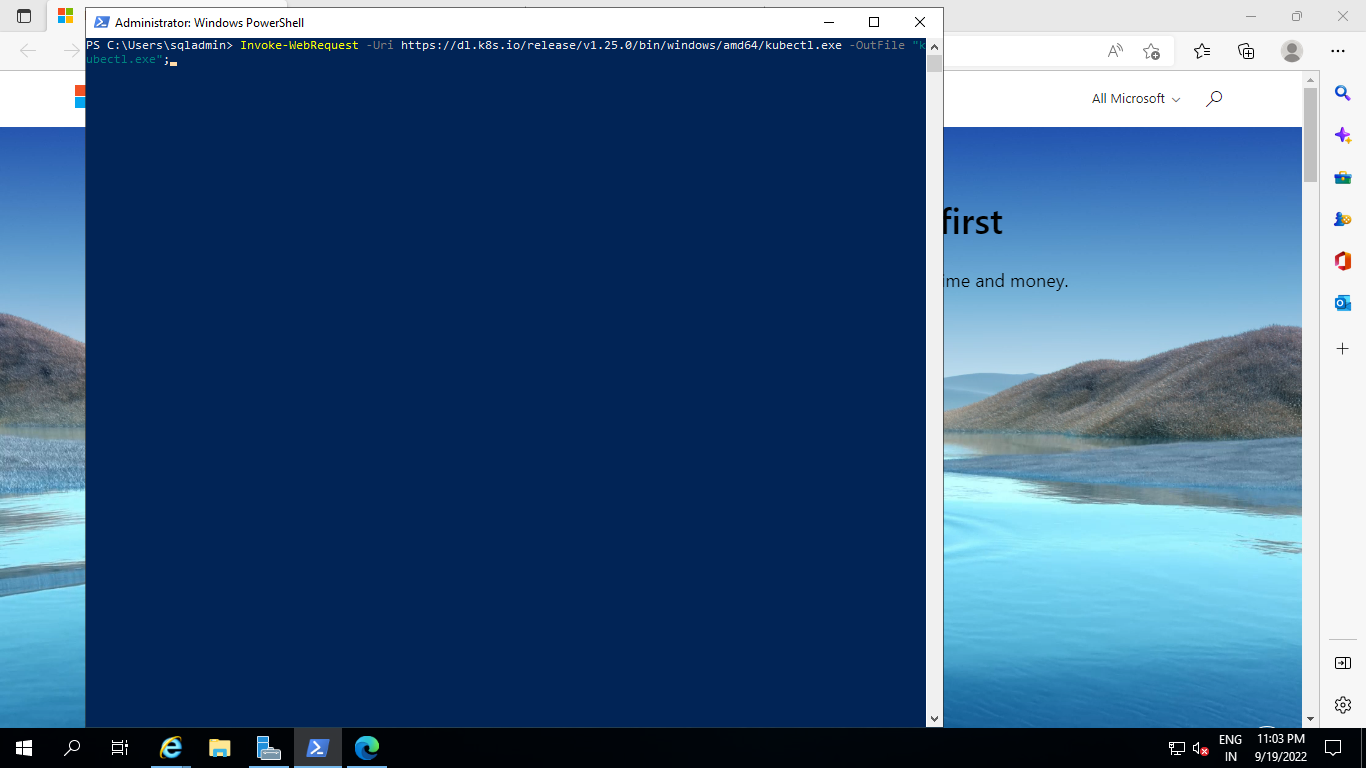
}

}

**Install Kubernates Extensions (Kubectl Extensions):**

Enter below Command in Azure Powershell:

Invoke-WebRequest -Uri https://dl.k8s.io/release/v1.25.0/bin/windows/amd64/kubectl.exe -OutFile "kubectl.exe"; Start-Process -FilePath "kubectl.exe"



**Set metrics and logs service credentials**

These services require a credential for each service. The credential is a username and a password. For this step, set an environment variable with the values for each credential.

The environment variables include passwords for log and metric services. The passwords must be at least eight characters long and contain characters from three of the following four categories: Latin uppercase letters, Latin lowercase letters, numbers, and non-alphanumeric characters.

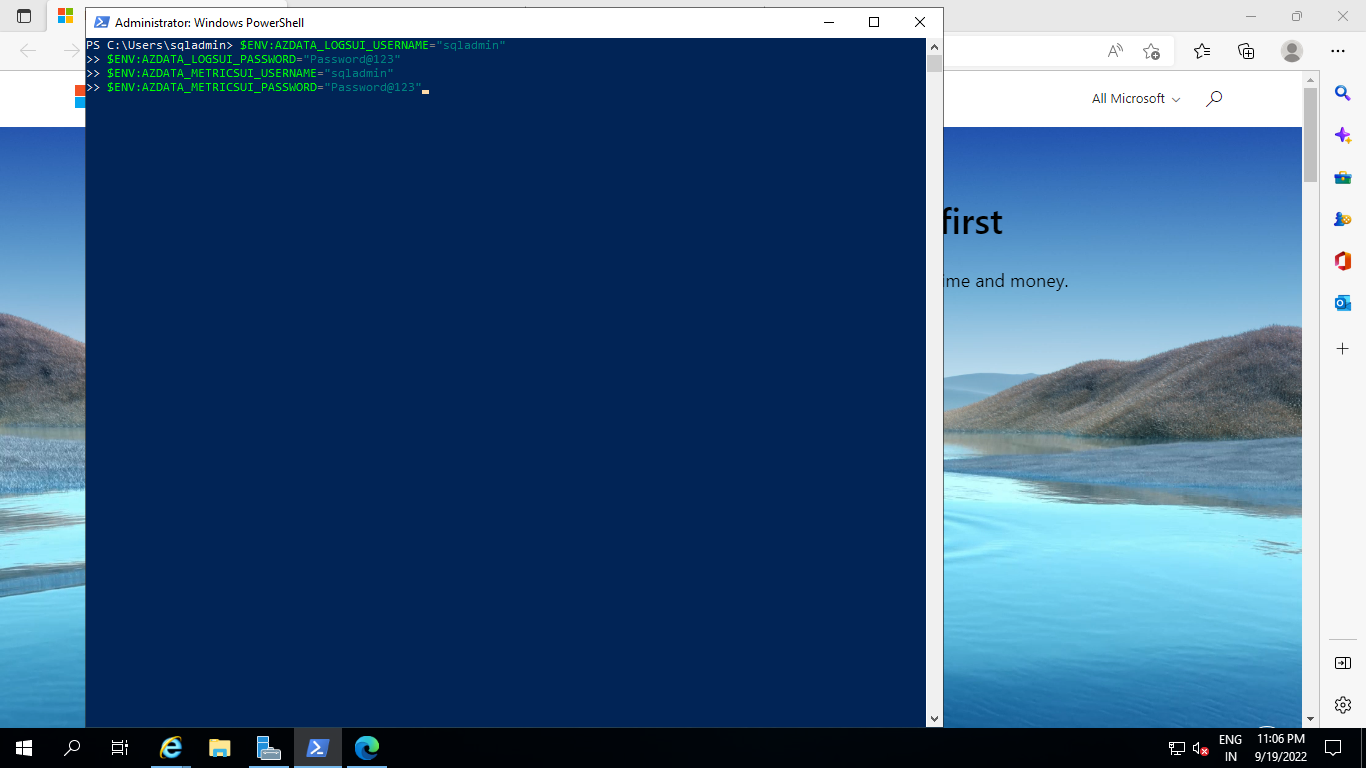
Run the following command to set the credential.

$ENV:AZDATA\_LOGSUI\_USERNAME="sqladmin"

$ENV:AZDATA\_LOGSUI\_PASSWORD="Password@123"

$ENV:AZDATA\_METRICSUI\_USERNAME="sqladmin"

$ENV:AZDATA\_METRICSUI\_PASSWORD="Password@123"



**Create and connect to your Kubernetes cluster**

You need to create and access to a Kubernetes cluster. The steps in this section deploy a cluster on Azure Kubernetes Service (AKS).

Follow the steps below to deploy the cluster from the Azure CLI.

Create Kubernetes cluster

Create the cluster in the resource group that was created already.

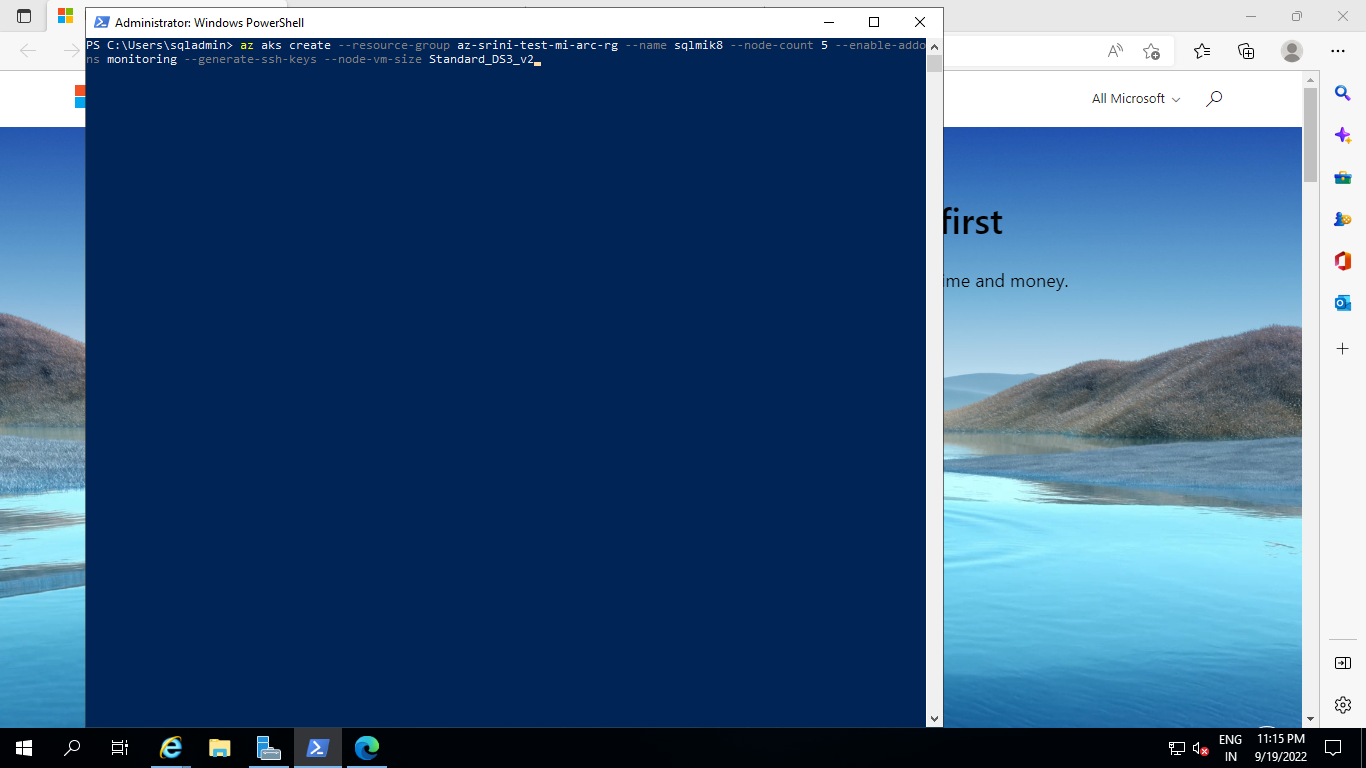
The following example creates a five node cluster, with monitoring enabled, and generates public and private key files if missing.

az aks create --resource-group <<resource group–name>><<Kubernates cluster name>> --node-count 5 --enable-addons monitoring --generate-ssh-keys --node-vm-size Standard\_DS3\_v2

* User has to choose current resource group name and custom cluster name

Example:

az aks create --resource-group az-srini-test-mi-arc-rg --name sqlmik8 --node-count 5 --enable-addons monitoring --generate-ssh-keys --node-vm-size Standard\_DS3\_v2



Get credentials

You will need to get credential to connect to your cluster.

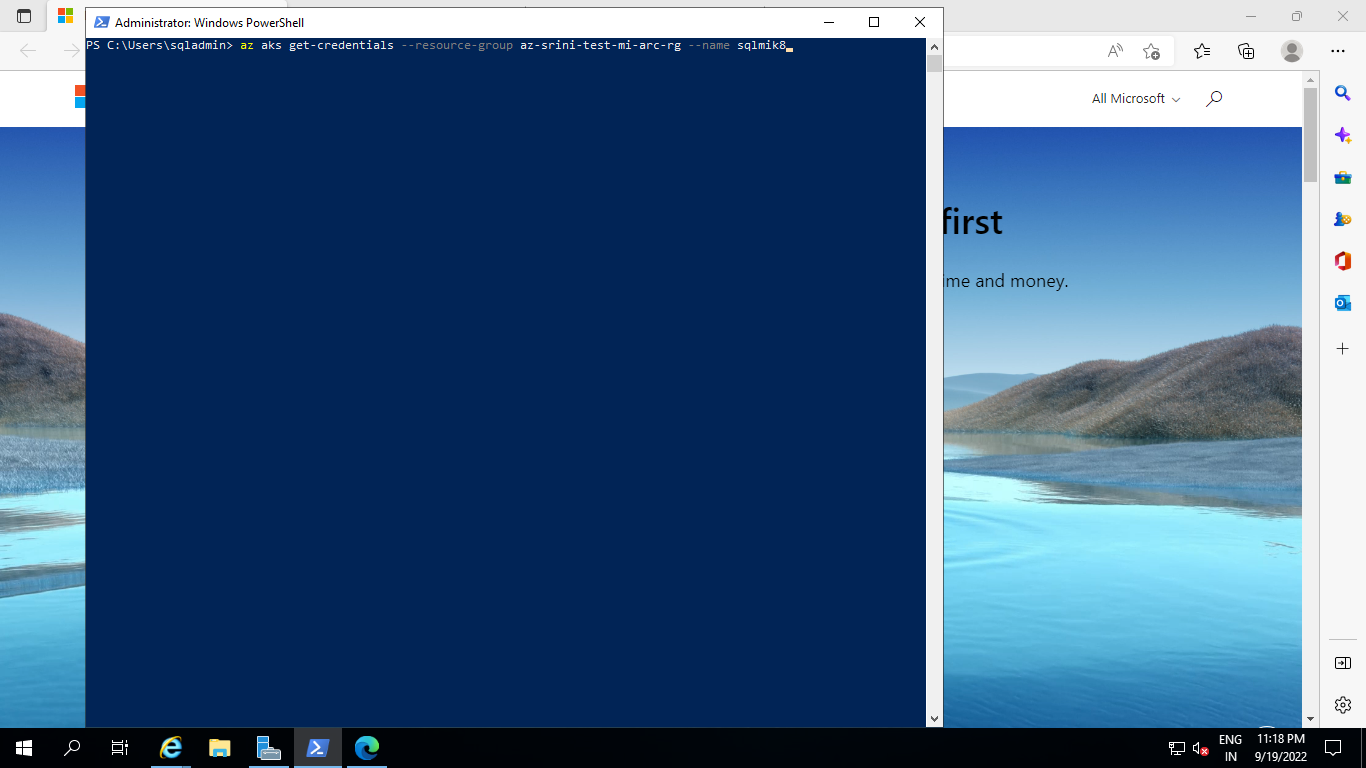
Run the following command to get the credentials:

az aks get-credentials --resource-group<<resource group–name>>

--name<<Kubernates cluster name>>

Example:

az aks get-credentials --resource-group az-srini-test-mi-arc-rg --name sqlmik8

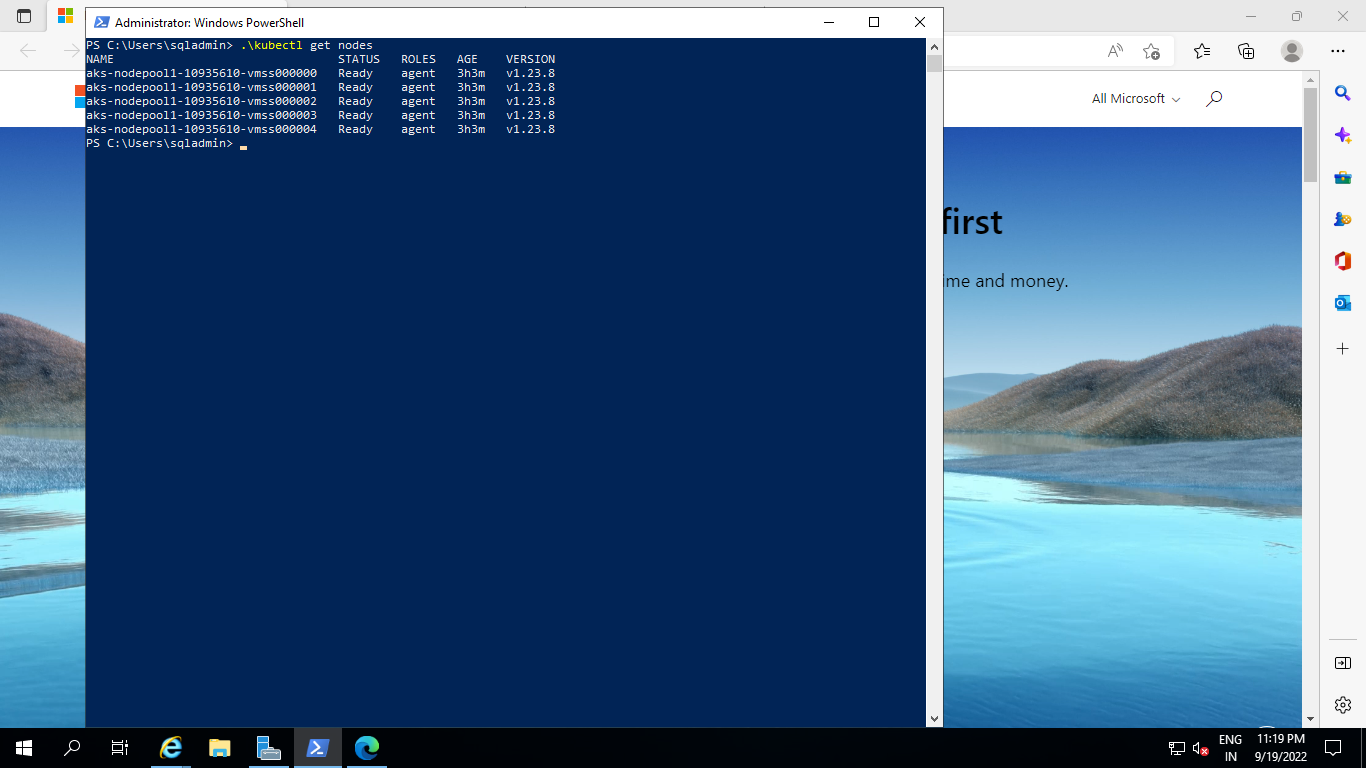


Verify cluster

To confirm the cluster is running and that you have the current connection context, run

.\kubectl get nodes

The command returns a list of nodes. For example:



**Convert AKS to Arc Enabled AKS**

Give below command to covert created Azure Kubernates cluster (AKS) to Arc Enabled Azure Kubernates cluster (AKS)

az connectedk8s connect --resource-group <resource-group name> --name <AKS Name>

Example

az connectedk8s connect --resource-group az-srini-test-mi-arc-rg --name sqlmik8



**Create the data controller**

Now that our cluster is up and running, we are ready to create the data controller in indirectly connected mode.

The CLI command to create the data controller is:

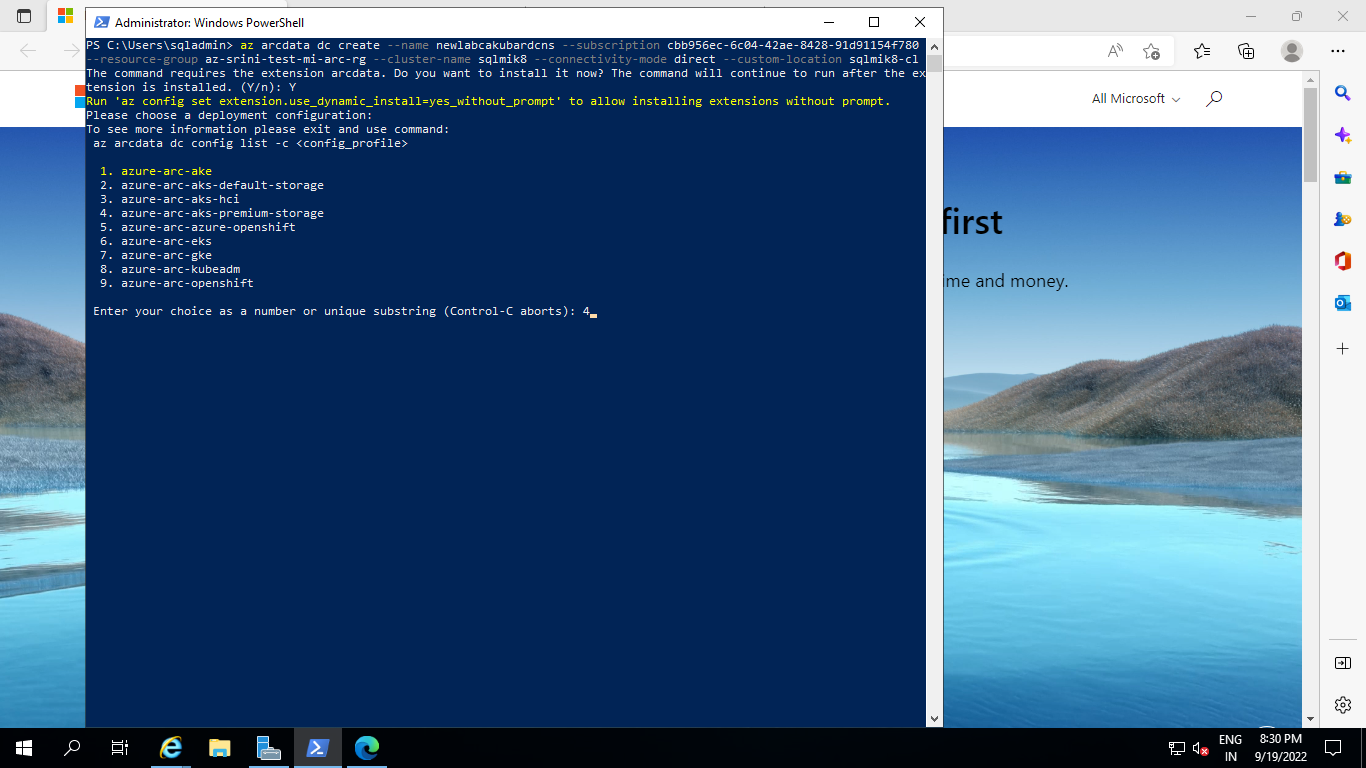
az arcdata dc create --name <data controller name> --subscription <subscription id> --resource-group <resource group name> --cluster name <Kubernates cluster name> --connectivity-mode direct –custom-location<custom location name>

Example:

az arcdata dc create --name newlabcakubardcns --subscription cbb956ec-6c04-42ae-8428-91d91154f780 --resource-group az-aashik-test-mi-arc-rg --cluster-name newlabcakubcluns --connectivity-mode direct --custom-location newlabcakubcluns-cl

Allow arcdata extension,

Enter your choice as a number or unique substring (Control-C aborts): 4 (azure-arc-aks-premium storage)



**Monitor deployment**

You can also monitor the creation of the data controller with the following command:

kubectl get datacontroller --namespace <namespace>

The command returns the state of the data controller. For example, the following results indicate that the deployment is in progress:

NAME STATE

<namespace> DeployingMonitoring

Once the state of the data controller is ‘READY’, then this step is completed. For

Example:

NAME STATE

<namespace> Ready

**Create an instance of Azure Arc-enabled SQL Managed Instance**

Now, we can create the Azure MI for directly connected mode with the following command:

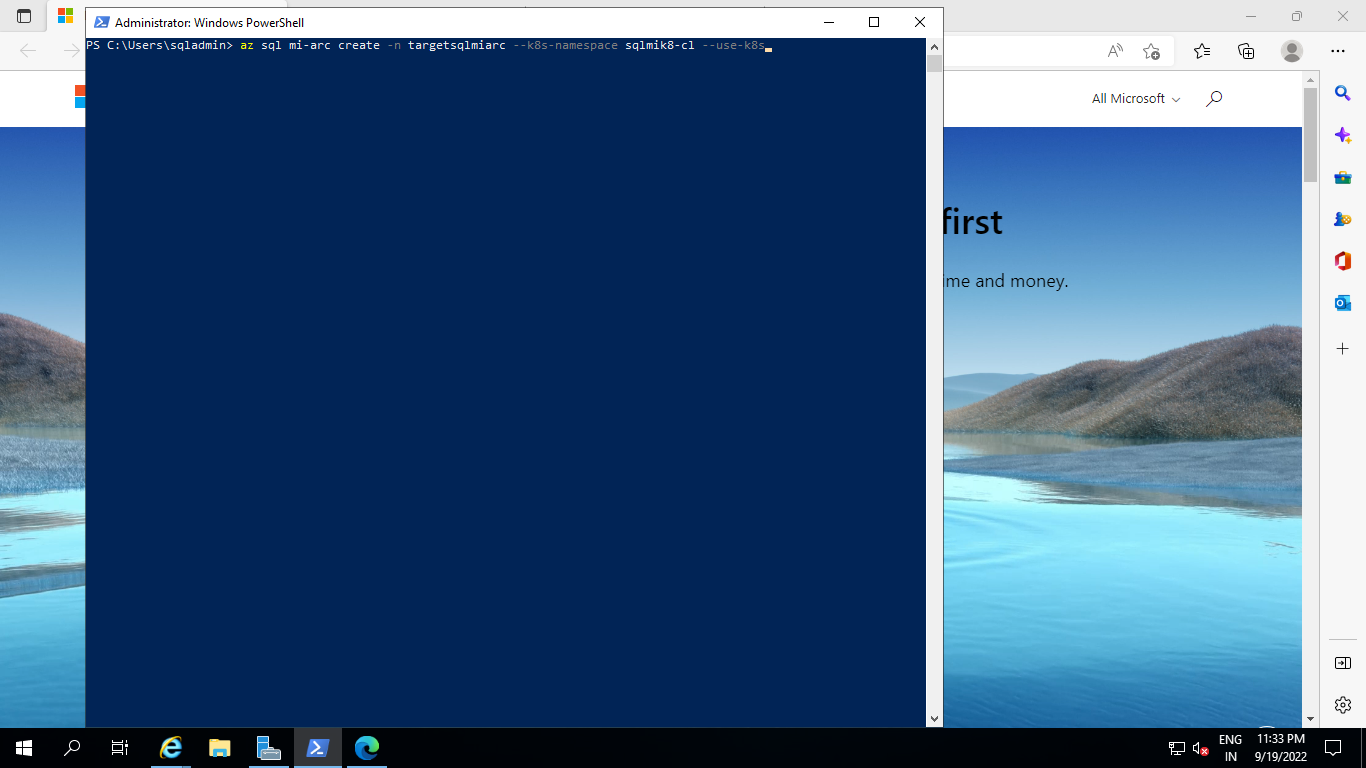
az sql mi-arc create -n <instanceName> --k8s-namespace <namespace> --use-k8s

Example:

az sql mi-arc create -n newlabcakubsqlmins --k8s-namespace newlabcakubcluns-cl --use-k8s

Enter

Username – sqladmin, Password: Password@123



To know when the instance has been created, run:

kubectl get sqlmi -n <namespace>[

Once the state of the managed instance namespace is ‘READY’, then this step is completed. For example:

NAME STATE

<namespace> Ready

**Copy the backup file into an Azure SQL Managed Instance - Azure Arc pod using kubectl**

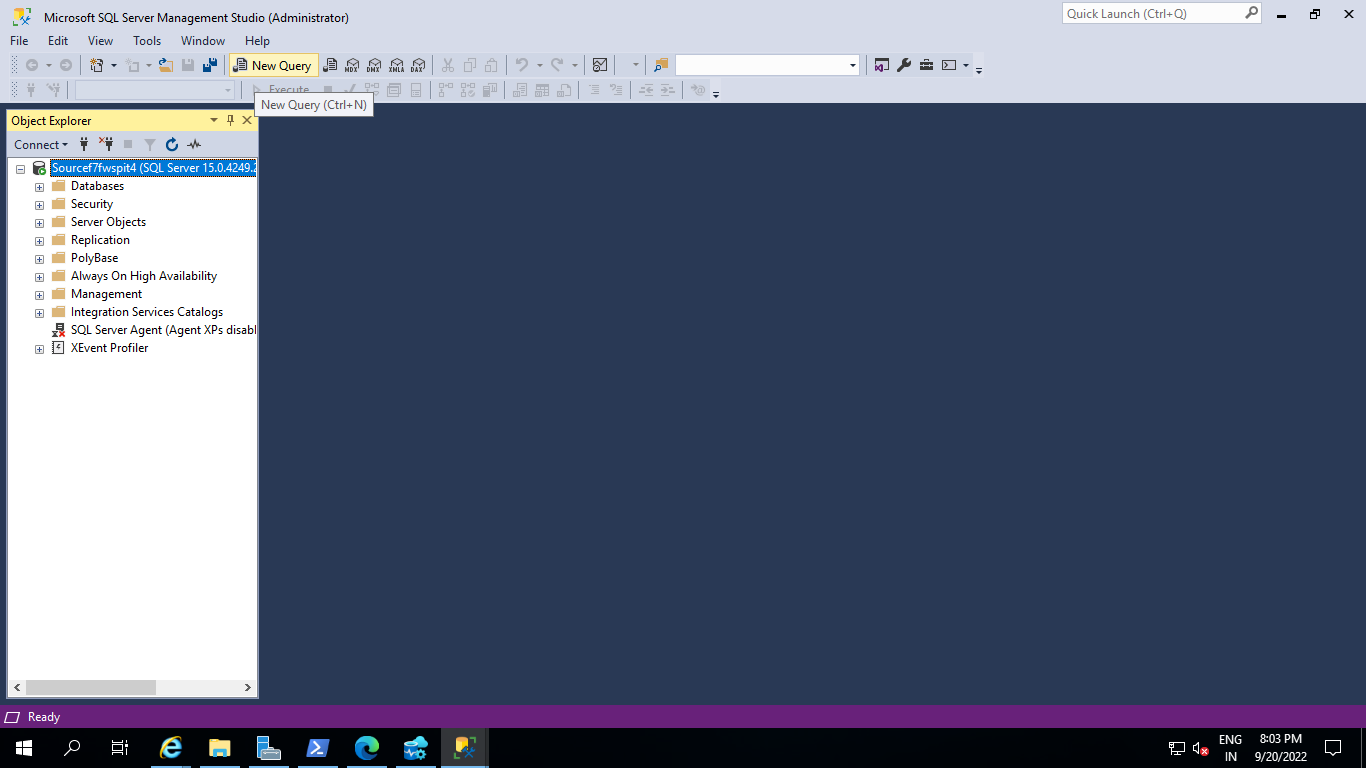
This method shows you how to take a backup file that you create via any method and then copy it into local storage in the Azure SQL managed instance pod so you can restore from there much like you would on a typical file system on Windows or Linux. In this scenario, you will be using the command kubectl cp to copy the file from one place into the pod's file system.

### Prerequisites

* Install and configure kubectl to point to your Kubernetes cluster where Azure Arc data services is deployed
* Have a tool like Azure Data Studio or SQL Server Management Server installed and connected to the SQL Server where you want to create the backup file OR have an existing .bak file already created on your local file system.
* Connect to source SQL server using Azure Data studio or Microsoft SQL Server management Studio

### Step 1: Backup the database from Source SQL

Search Microsoft SQL Server Management Studio and open



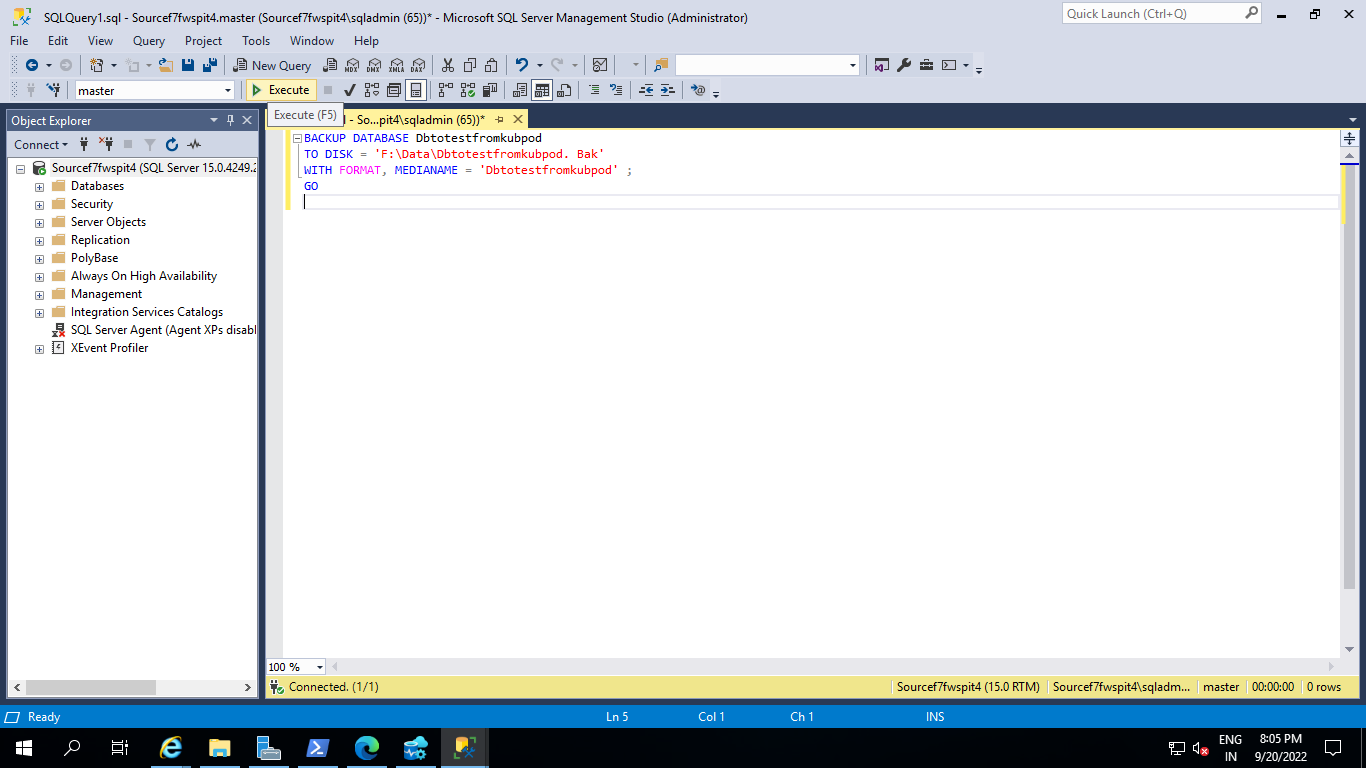
Backup the SQL Server database to your local file path like any typical SQL Server backup to disk:

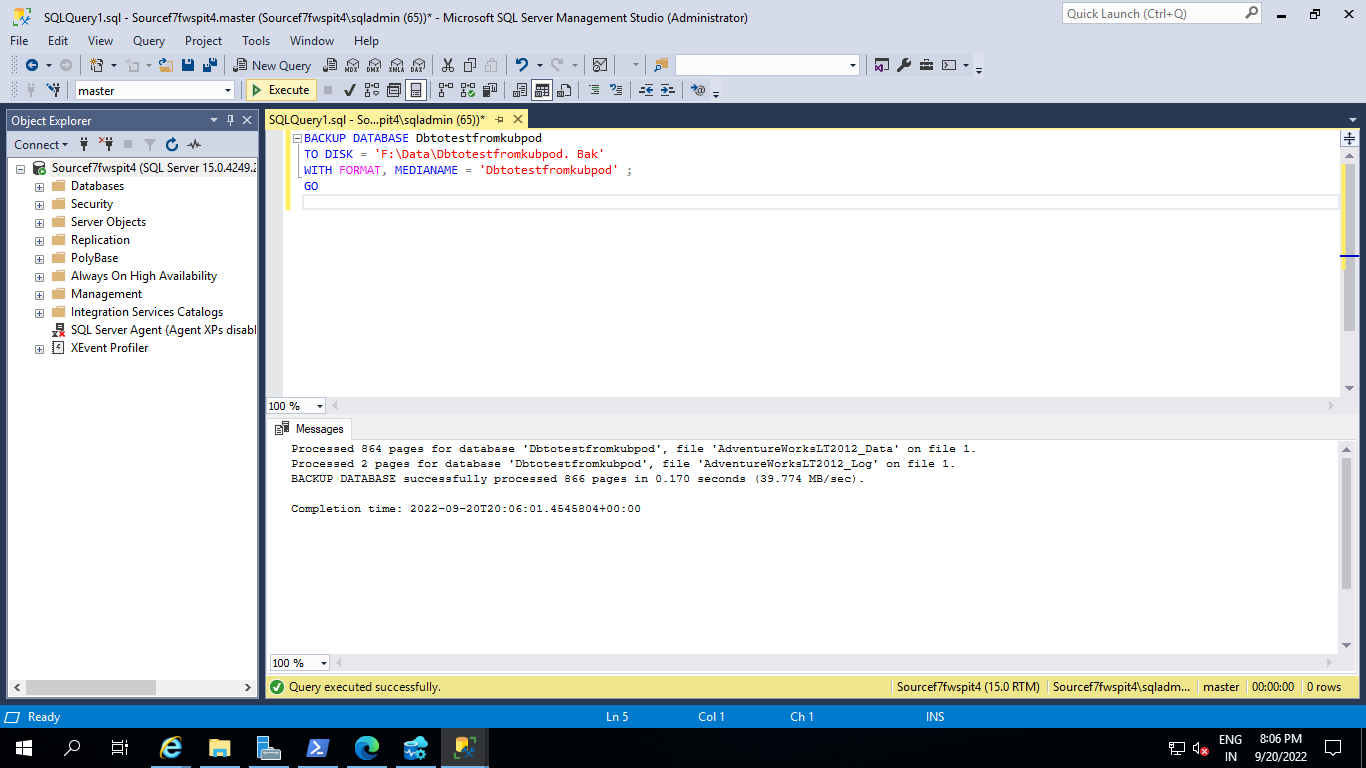
BACKUP DATABASE Dbtotestfromkubpod

TO DISK = 'F:\Data\Dbtotestfromkubpod. Bak'

WITH FORMAT, MEDIANAME = 'Dbtotestfromkubpod' ;

GO



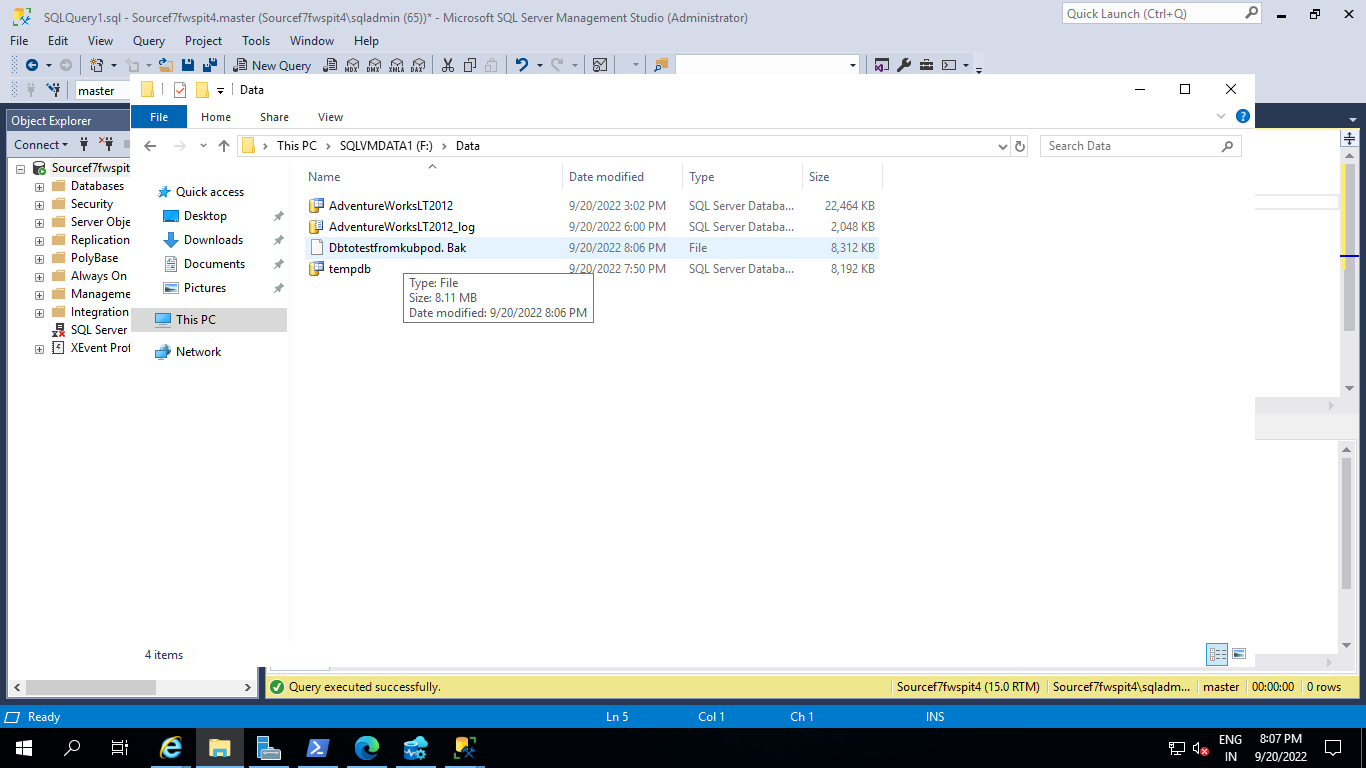


Move the backup to folder where kubectl is installed ( C:\Users\sqladmin Folder)

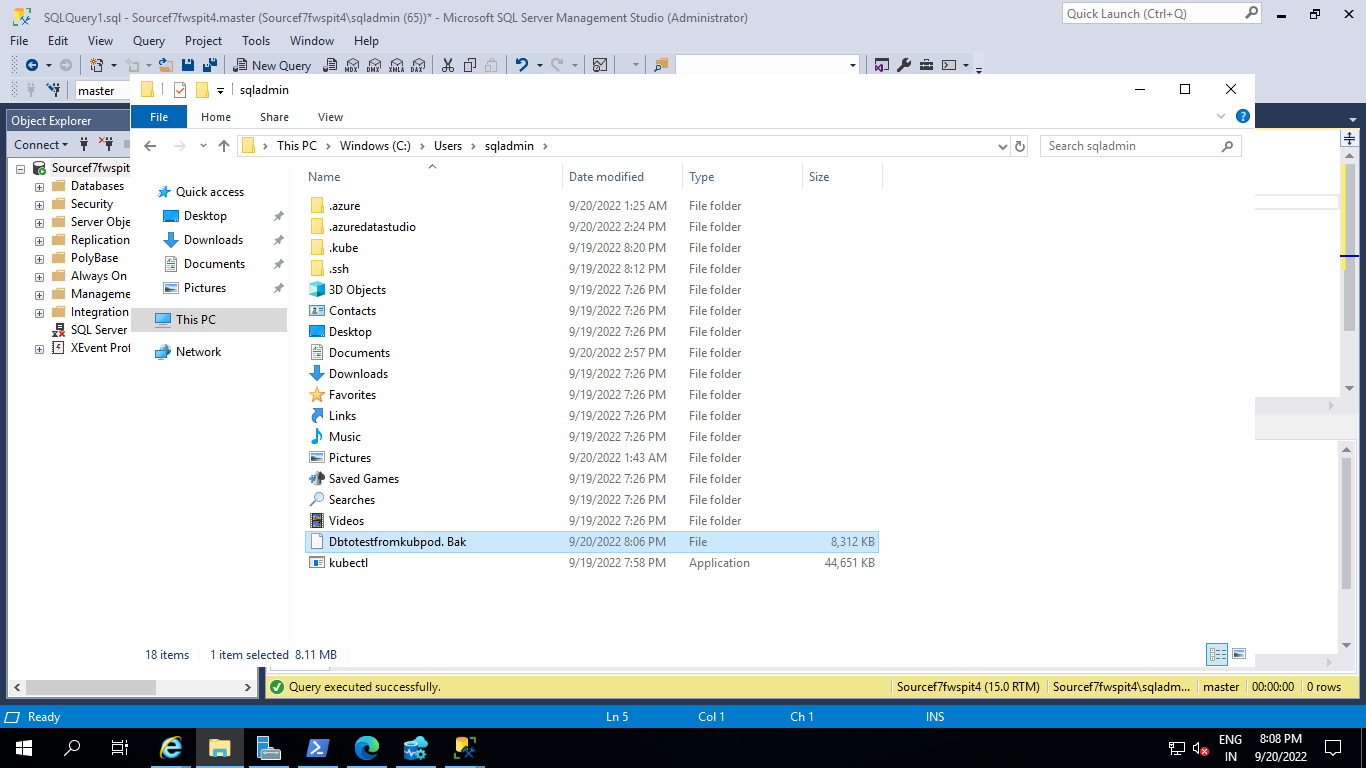
Example:

Move-Item –Path F:\Data\Dbtotestfromkubpod.bak -Destination C:\Users\sqladmin\Dbtotestfromkubpod.bak

Copy Backup file form F:\Data Folder with File name Dbtotestfromkubpod.bak



Paste Dbtotestfromkubpod.bak file to C:\Users\sqladmin\ Folder



### Step 2: Copy the backup file into the pod's file system

Find the name of the pod where the sql instance is deployed. Typically it should look like pod/<sqlinstancename>-0

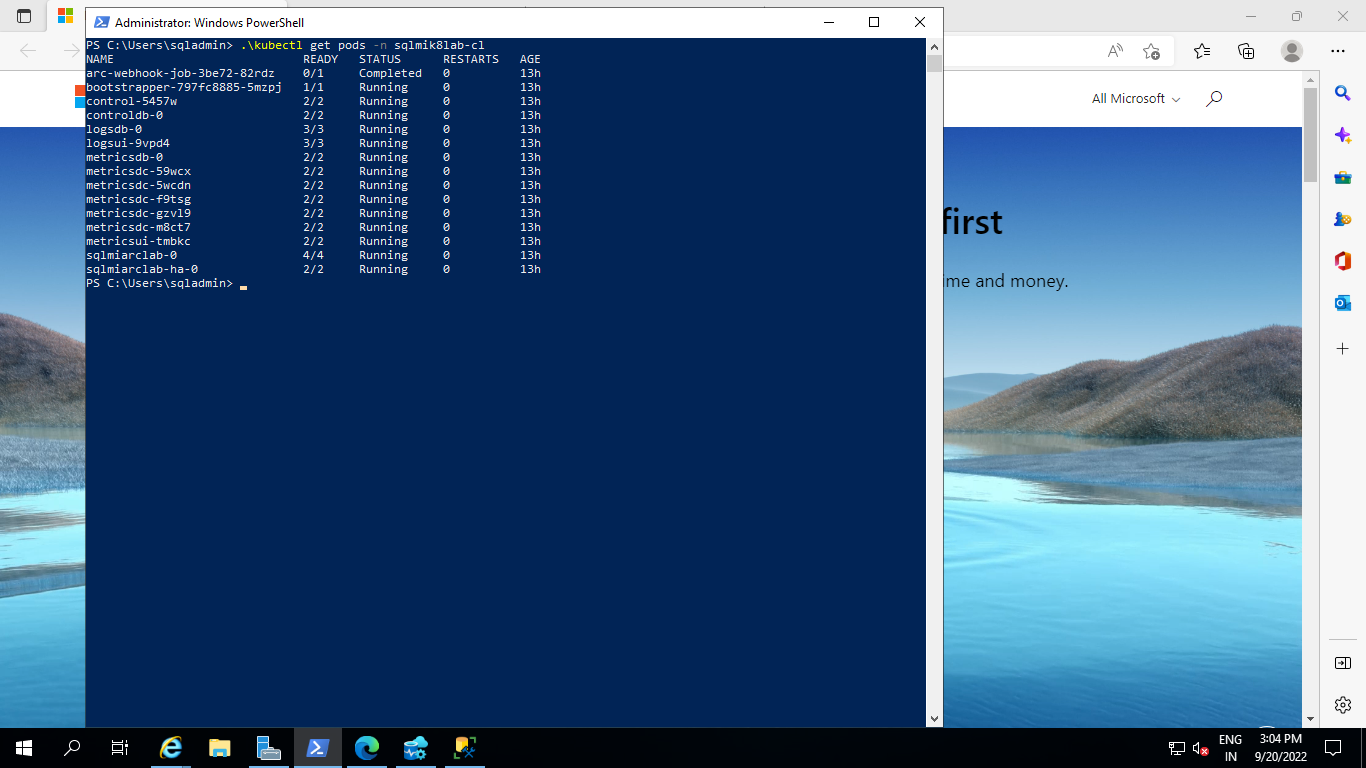
Get the list of all pods by running:

kubectl get pods -n <namespace of data controller>

Example:

.\kubectl get pods -n sqlmik8lab-cl

-- Get the list of all pods make sure sql mi is within



Example:

Copy the backup file from the local storage to the sql pod in the cluster.

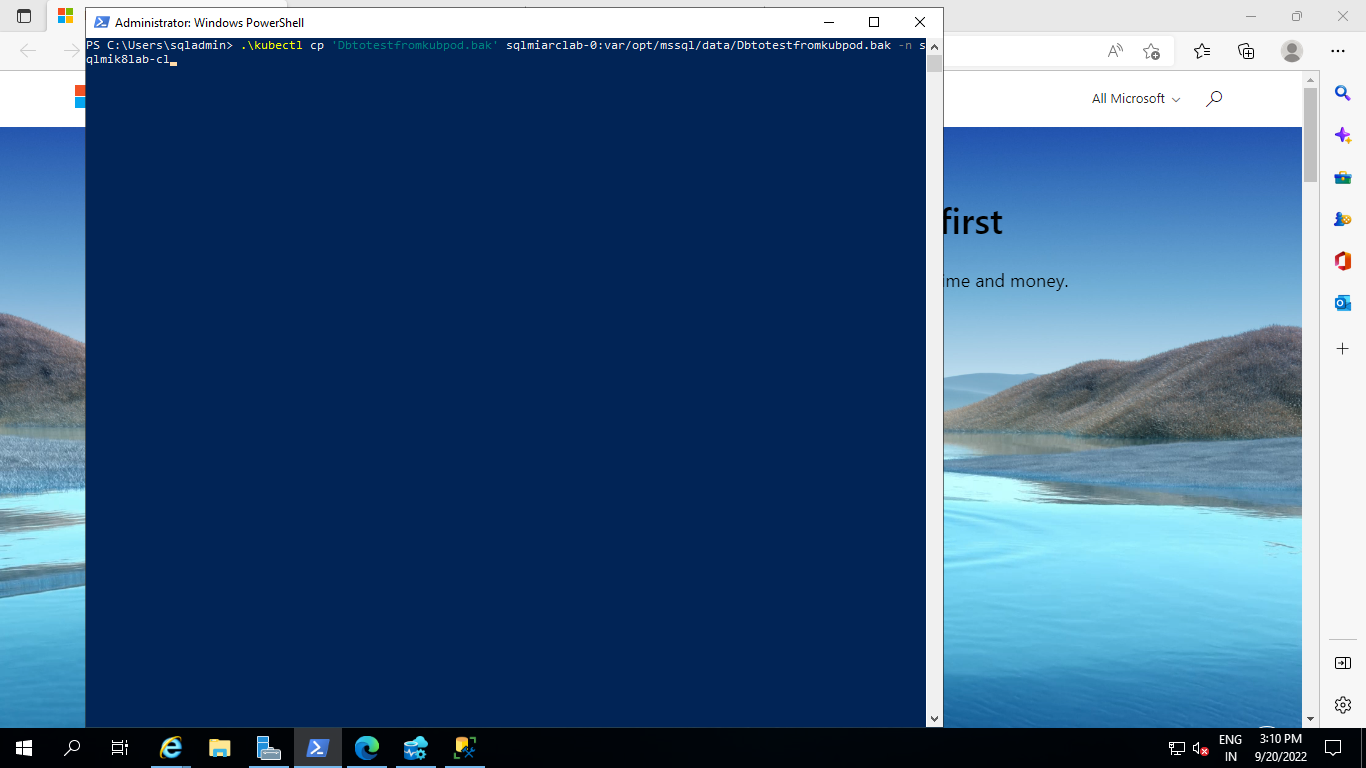
kubectl cp <source file location> <pod name>:var/opt/mssql/data/<file name> -n <namespace name>

Example:

.\kubectl cp 'Dbtotestfromkubpod.bak' newlabcakubsqlmins-0:var/opt/mssql/data/Dbtotestfromkubpod.bak -n newlabcakubcluns-cl

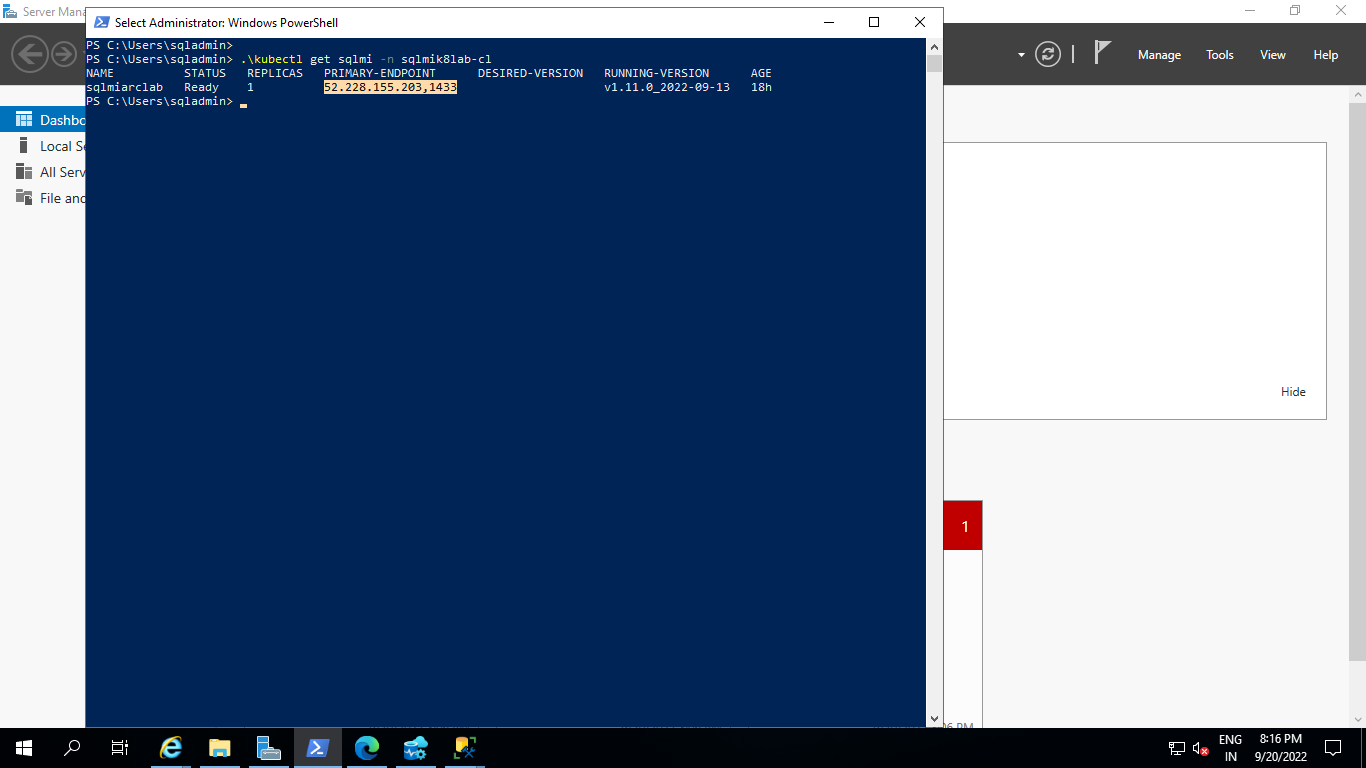
#Example:

kubectl cp C:\Backupfiles\test.bak sqlinstance1-0:var/opt/mssql/data/test.bak -n arc

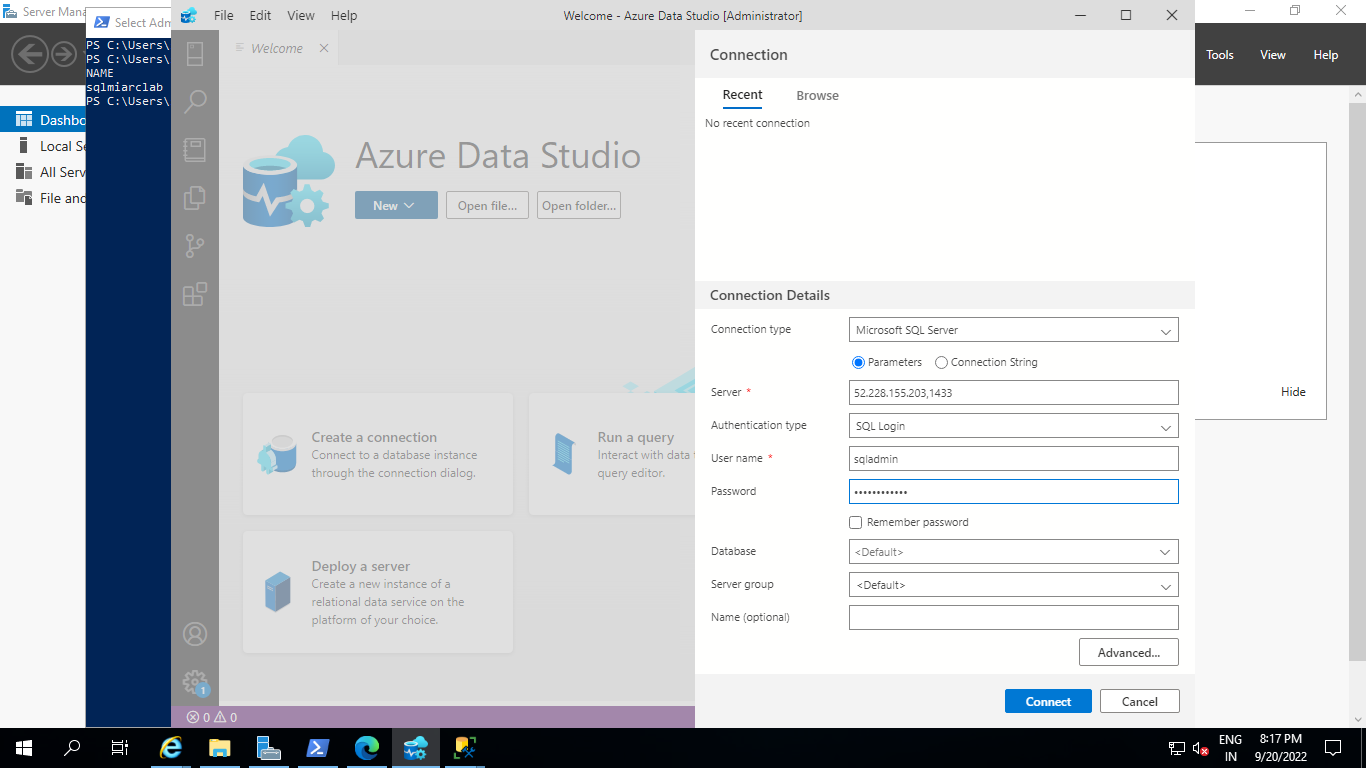


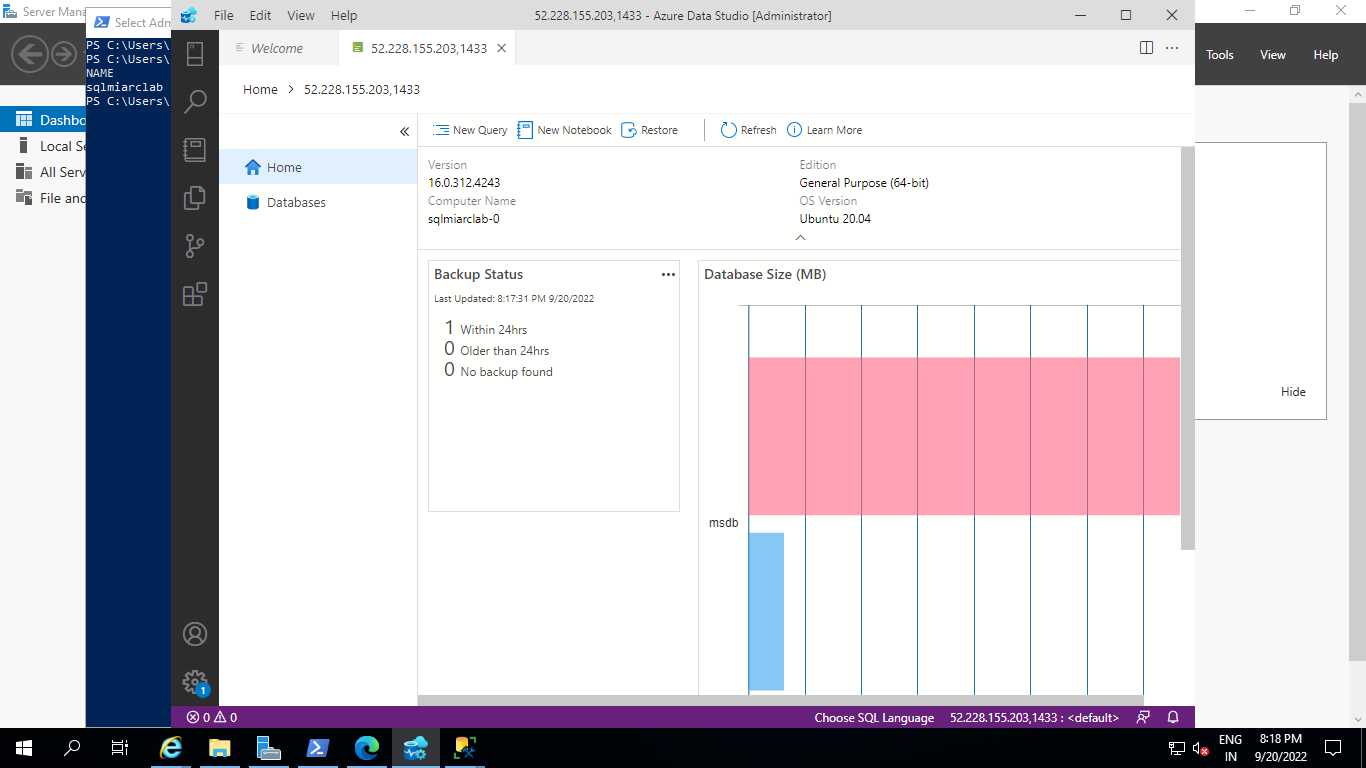
.\kubectl get sqlmi -n newlabcakubcluns-cl

Get Arc Enabled SQL MI IP address from above command output and use to connect with Azure data studio



Connect to Arc Enabled Azure SQL MI with Azure Data studio or Microsoft SQL Server studio with Public IP and execute below Restore command in New Query section





### Step 3: Restore the database

Prepare and run the RESTORE command to restore the backup file to the Azure SQL managed instance - Azure Arc

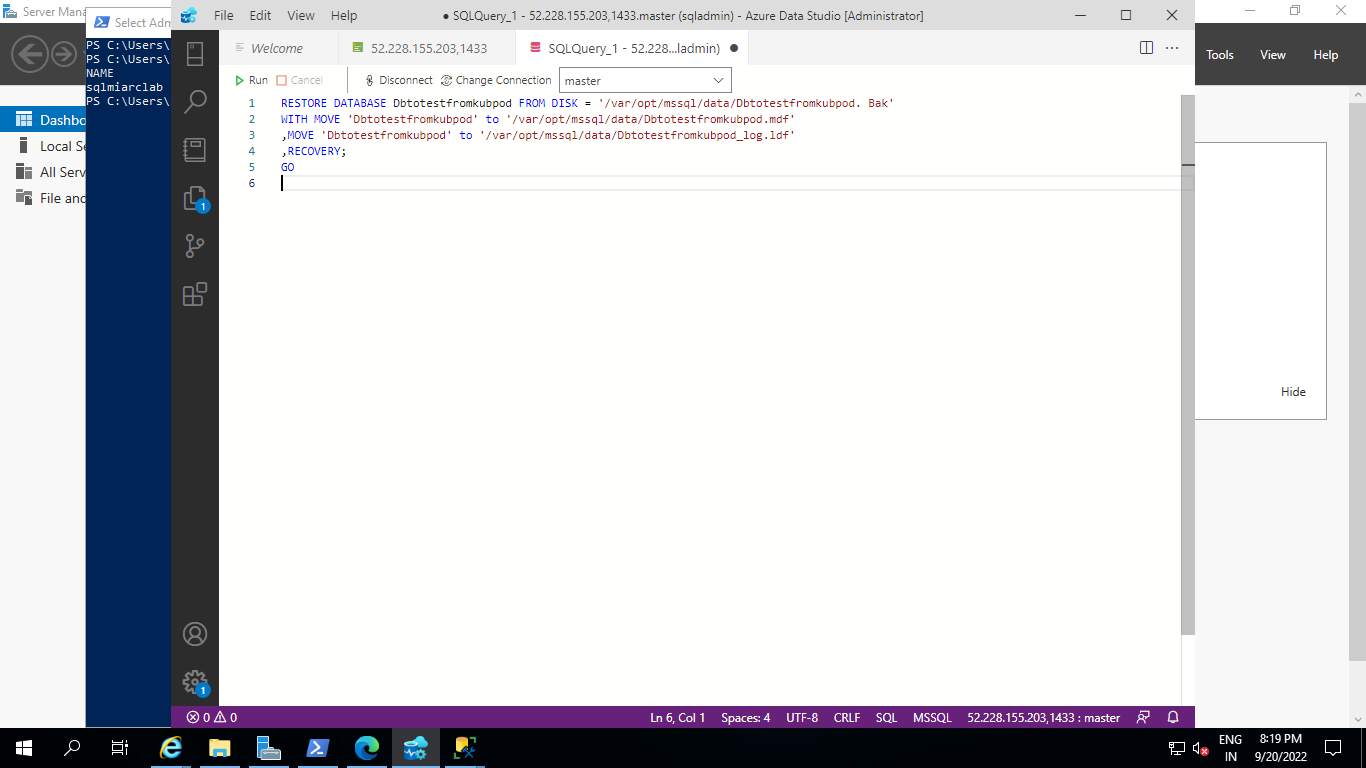
RESTORE DATABASE test FROM DISK = '/var/opt/mssql/data/<file name>.bak'

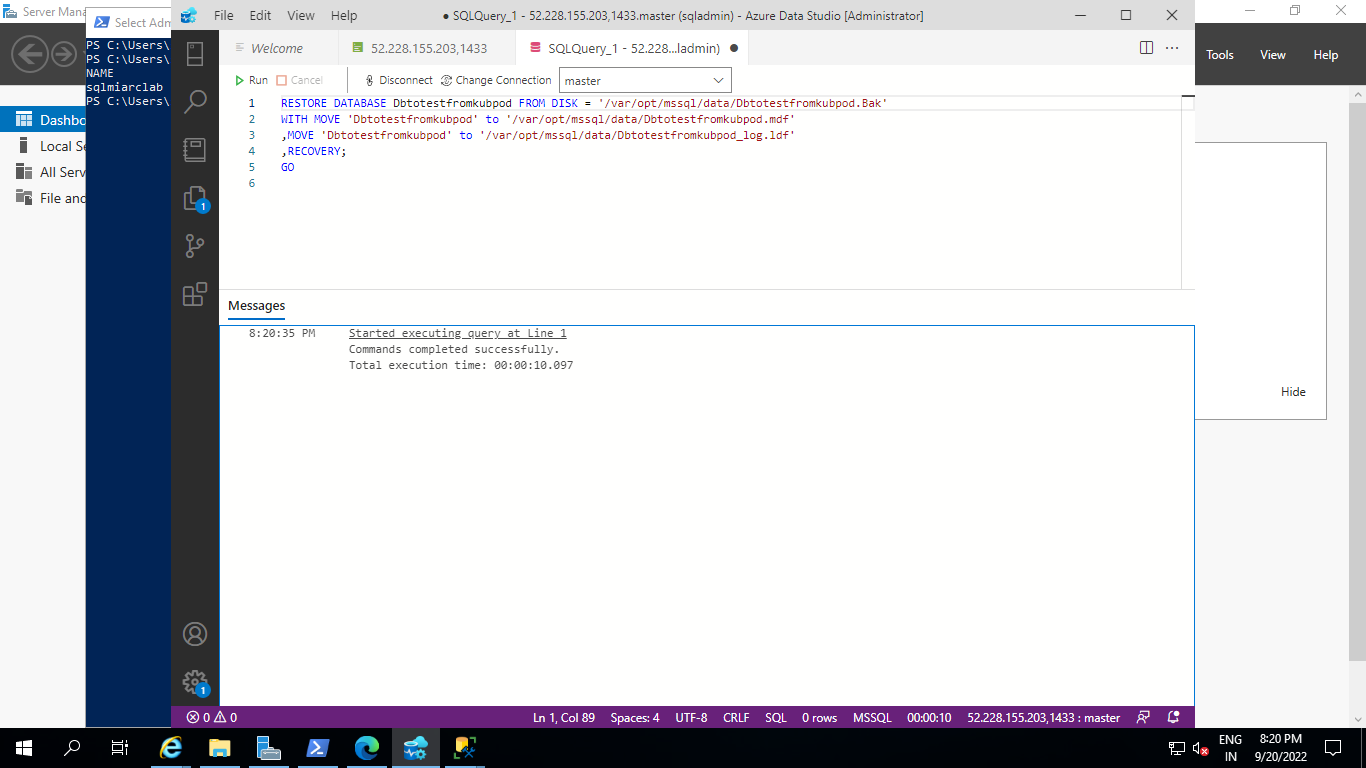
WITH MOVE '<database name>' to '/var/opt/mssql/data/<file name>.mdf'

,MOVE '<database name>' to '/var/opt/mssql/data/<file name>\_log.ldf'

,RECOVERY;

GO





Example:

RESTORE DATABASE Dbtotestfromkubpod FROM DISK = '/var/opt/mssql/data/Dbtotestfromkubpod. Bak'

WITH MOVE 'Dbtotestfromkubpod' to '/var/opt/mssql/data/Dbtotestfromkubpod.mdf'

,MOVE 'Dbtotestfromkubpod' to '/var/opt/mssql/data/Dbtotestfromkubpod\_log.ldf'

,RECOVERY;

GO

Backup file (Dbtotestfromkubpod.bak) from Source SQL Database successfully restored in Arc Enabled Azure SQL MI Server.

