Configuring the Azure Databricks environment

In this experiment, we'll learn how to configure the Azure Databricks environment by creating an Azure Databricks workspace, cluster, and cluster pools.

Getting ready

To get started, log into **`https://portal.azure.com`** using your Azure credentials.

How to do it...

An Azure Databricks workspace is the starting point for writing solutions in Azure Databricks. A workspace is where you create clusters, write notebooks, schedule jobs, and manage the Azure Databricks environment.

An Azure Databricks workspace can be created in an Azure managed virtual network or customer managed virtual network. We'll create the environment using a customer managed virtual network.

Creating an Azure Databricks service or workspace

Let's get started with provisioning the virtual network:

1. In Azure portal, type **`Virtual Net`** into the search box and select **Virtual Networks** from the search results:

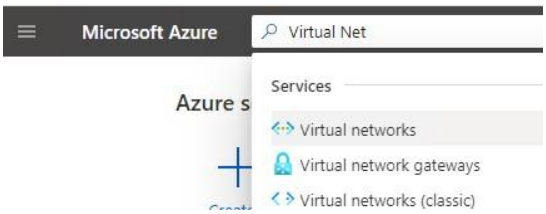
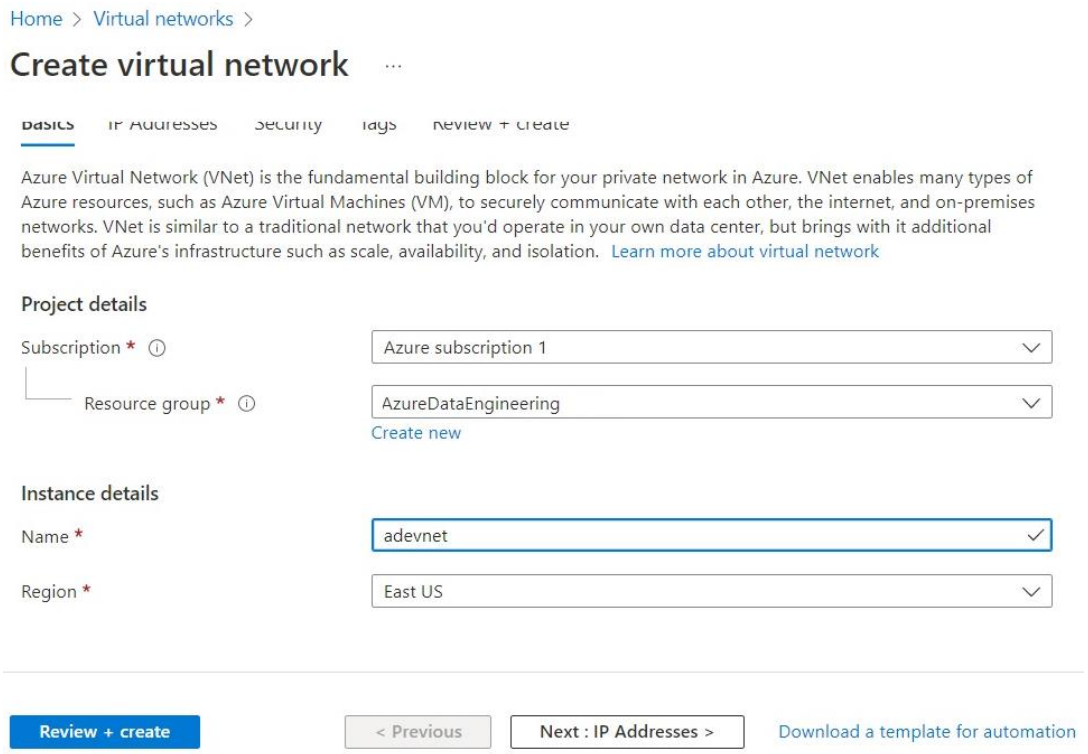


Figure 9.1 – Selecting virtual networks

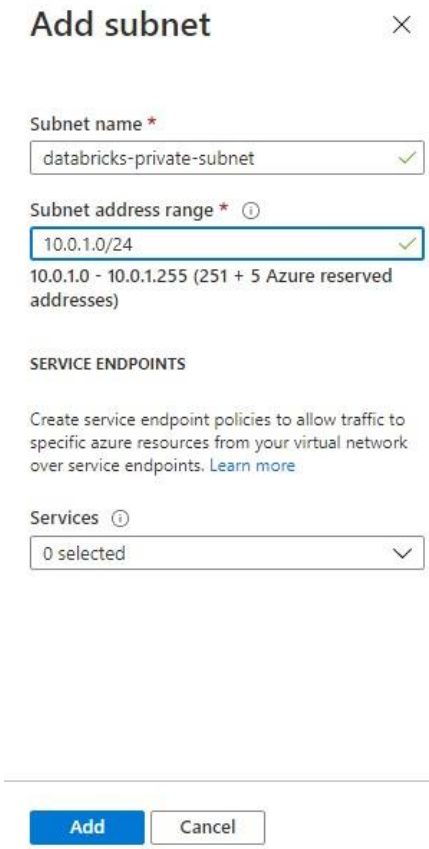
2. On the **Virtual networks** page, click **Add**. On the **Create virtual network** page, under the **Basics** tab, provide **Resource group** with a name, virtual network name, and region:



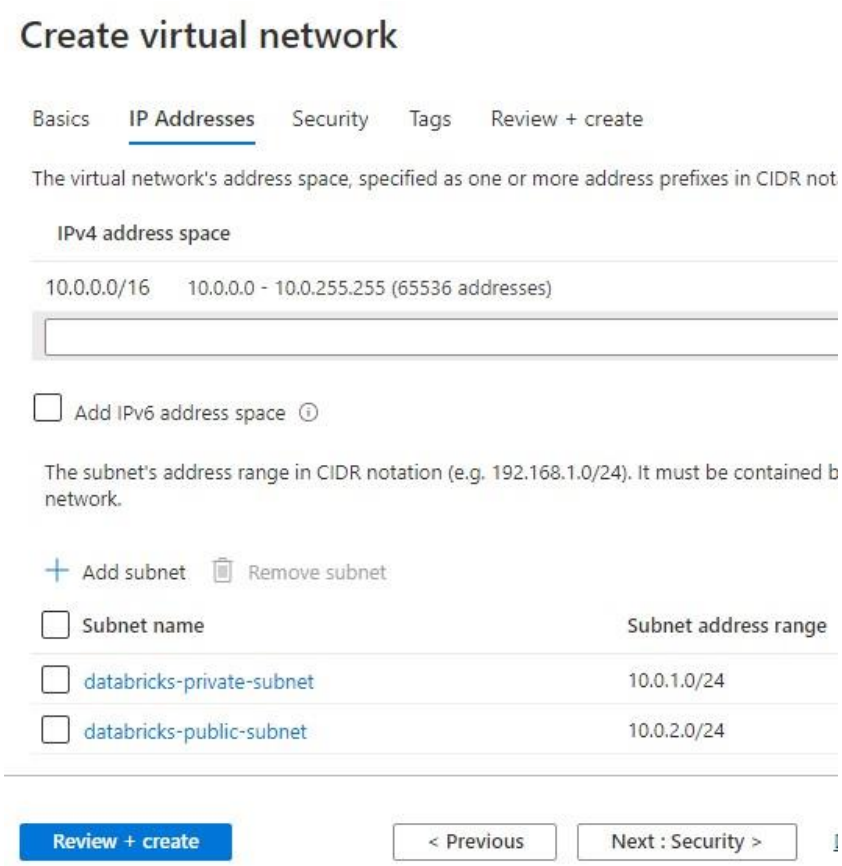
3. Click the **Next: IP Addresses >** button to go to the IP addresses tab. On the IP addresses tab, the IPv4 address space is listed as 10.0.0.0/16 by default. Leave it as-is. Select the default subnet and click **Remove subnet** to delete it:

Figure 9.3 – Removing the default subnet

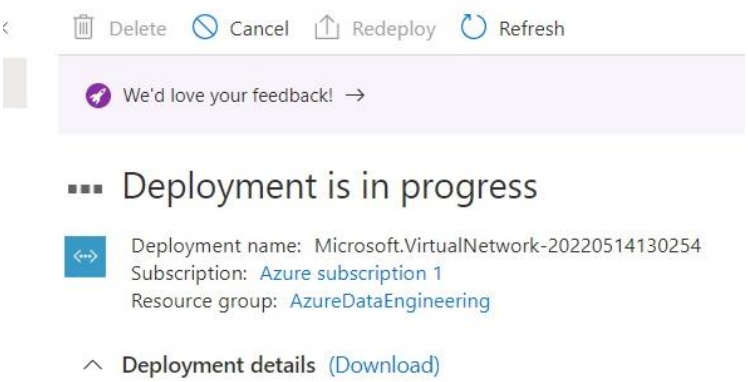
4. Click **Add subnet**. In the **Add subnet** dialog box, provide a subnet name of **`databricks-private-subnet`** and a subnet address range of **`10.0.1.0/24`**. Click **Add** to add the subnet:



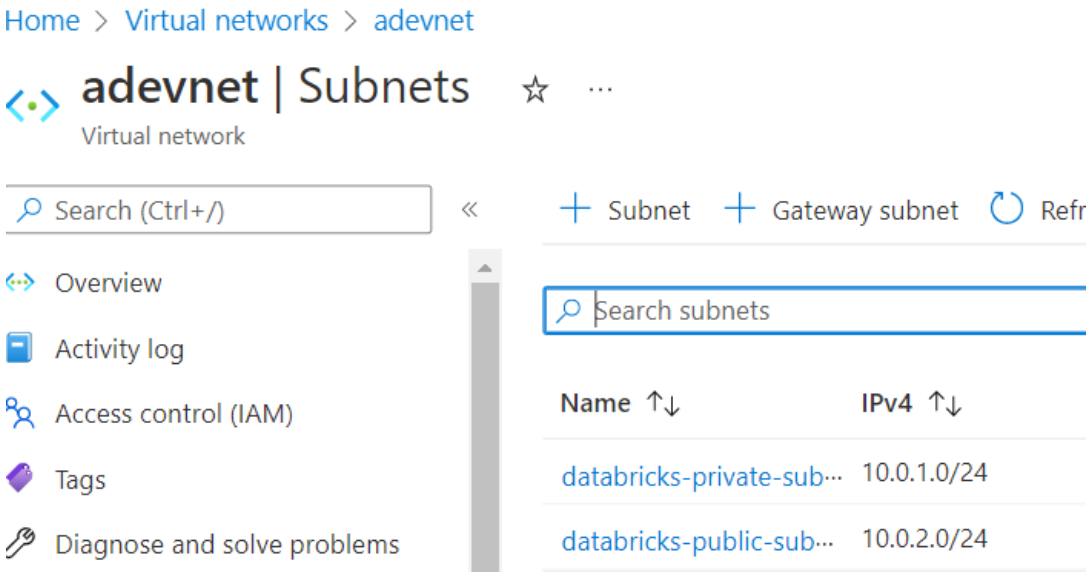
5. Similarly, add another subnet with a name of **`databricks-public-subnet`** and an address range of **`10.0.2.0/24`**:



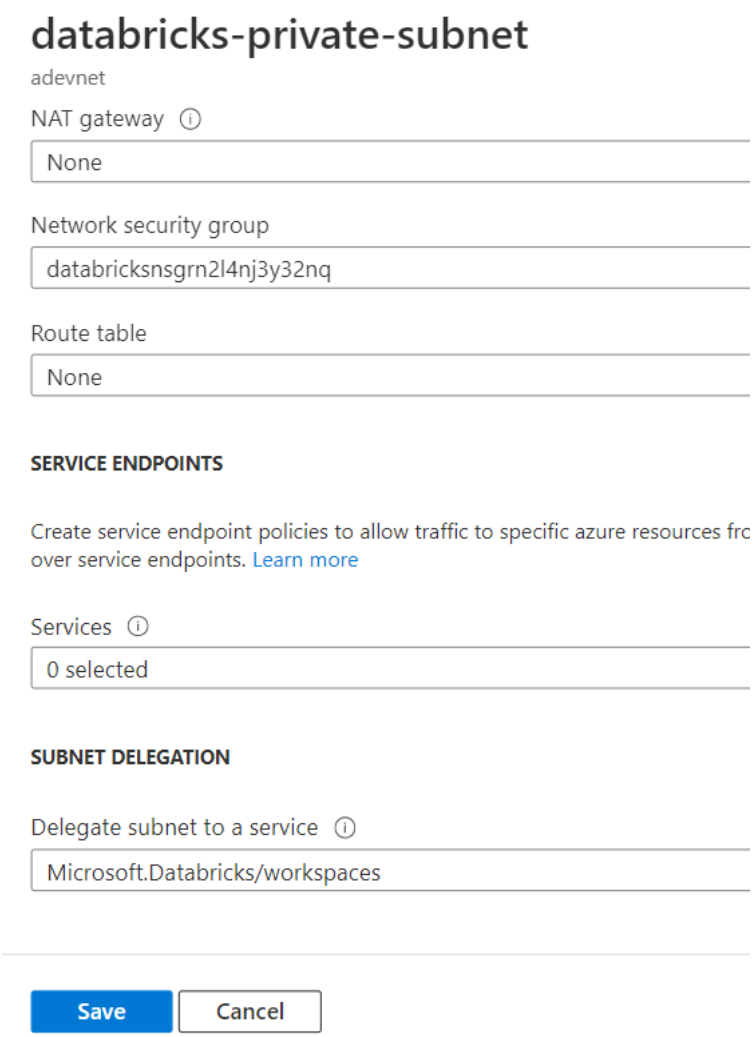
6. Click **Review + create** and then **Create** to create the virtual network. It usually takes 2-5 minutes to create this virtual network.



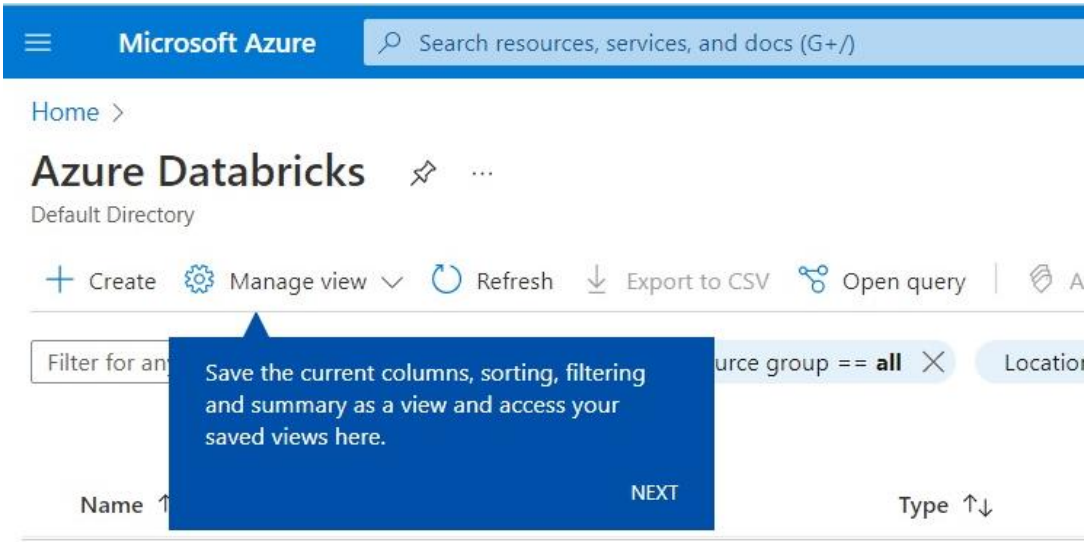
7. Once the virtual network has been created, click on **Go to Resources** to open the **Virtual network** page. On the **Virtual network** page, select **Subnets** from the **Settings** section:



8. Click on **databricks-private-subnet**. On the **databricks-private-subnet** page, scroll down to the bottom and select **Microsoft.Databricks/workspaces** from the **Delegate subnet to a service** dropdown:

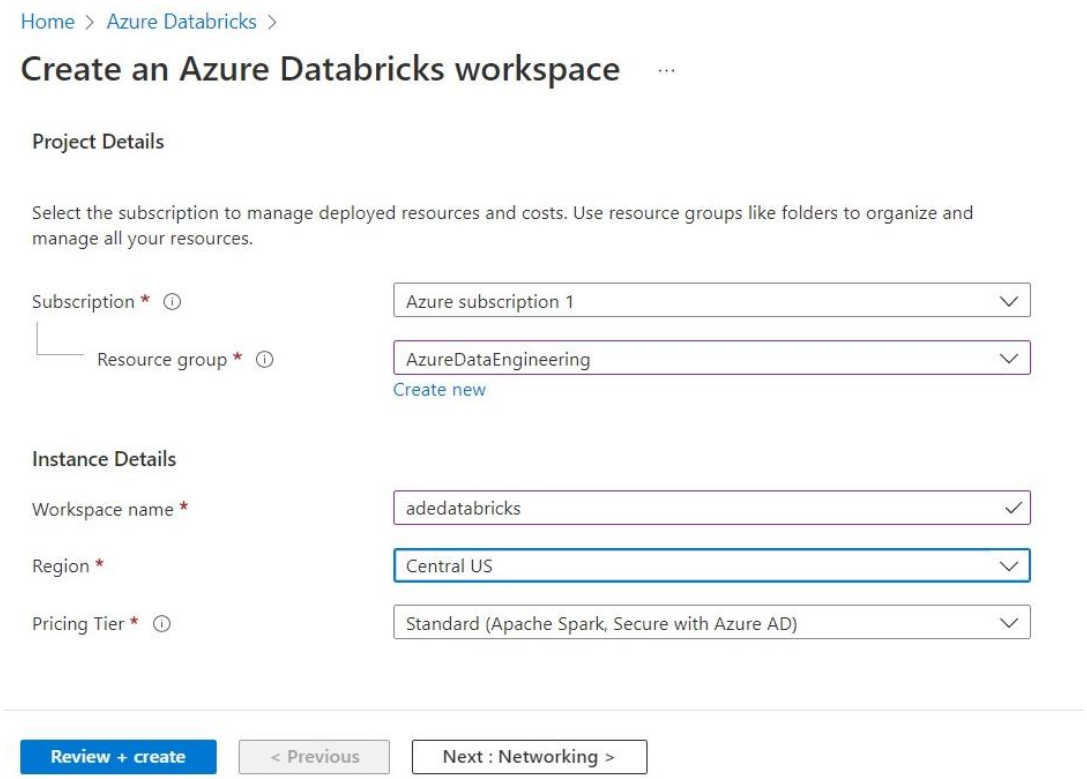


9. Click **Save** to apply the change. Follow the preceding steps to modify **`databricks-public-subnet`**, similar to how we modified **`databricksprivate-subnet`**. This completes the virtual network and subnet configuration.
10. In the Azure portal, type **`Azure Databricks`** into the search box and then select **Azure Databricks** from the search list. On the **Azure Databricks** page, click **Create** to create a new Azure Databricks service or workspace.

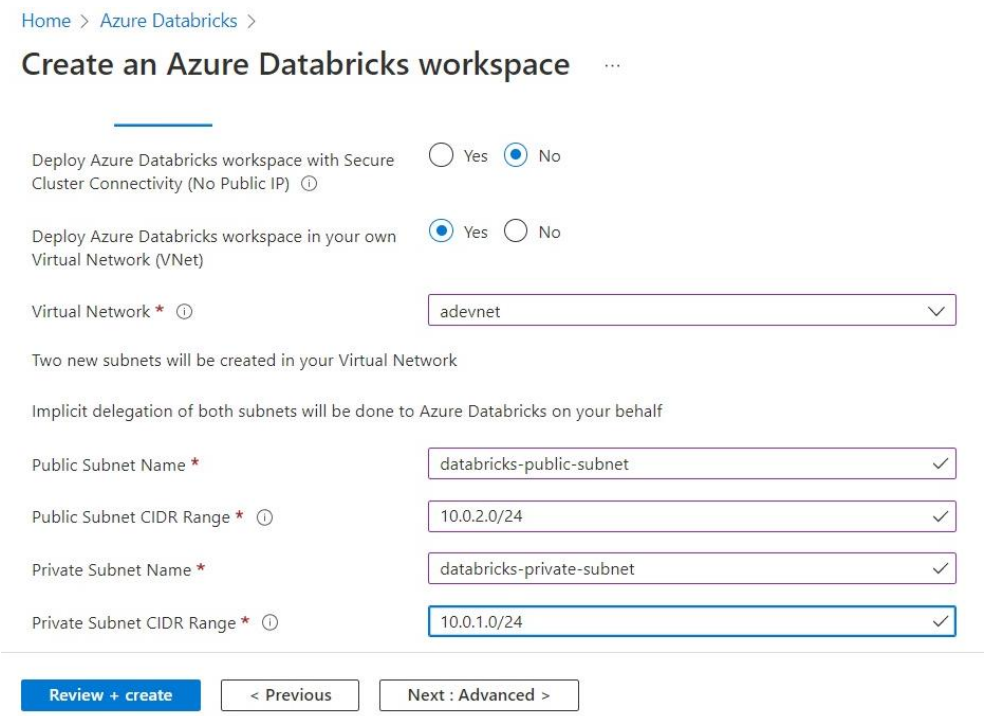


****** ProTip: Open the Databricks Blade jn a new browser tab so that you can refer back to the work done in the Azure VNet Blade**

11. On the **Azure Databricks Workspace** page, under the **Basics** tab, provide the **Resource group**, **Workspace name**, **Location**, and **Pricing Tier** values (Azure Databricks has two pricing tiers, **Standard** and **Premium**. Premium tier includes all the features of the Standard tier and role-based access. For more details about these tiers, please visit **`https://azure.microsoft.com/en-in/pricing/details/databricks/`**.):



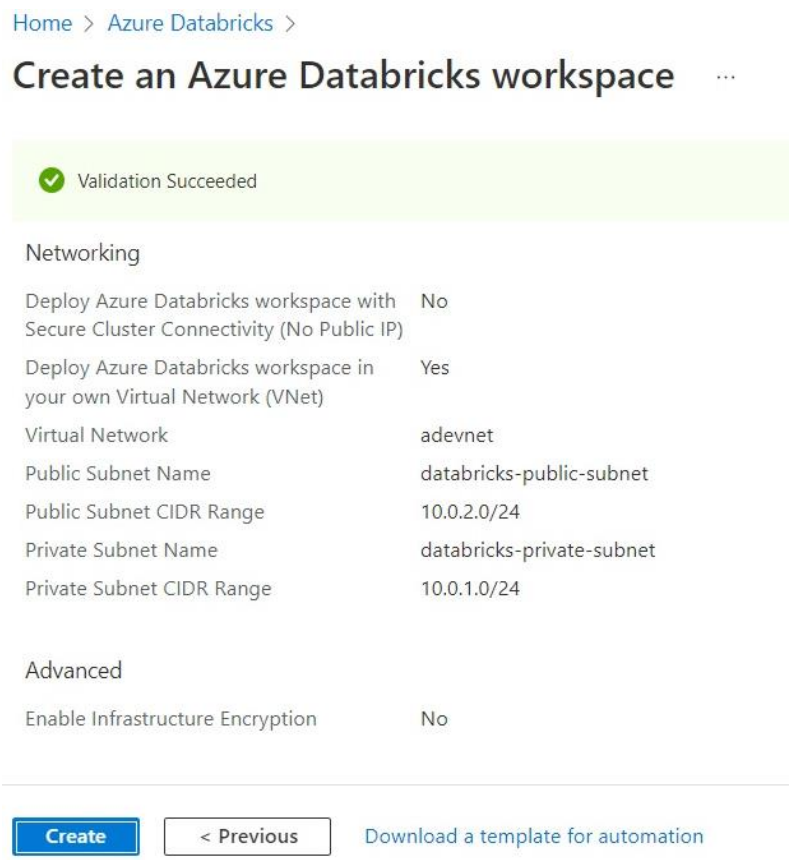
12. Click **Next: Networking >** to go to the **Networking** tab. On the **Networking** tab, select **Yes** for **Deploy Azure Databricks workspace in your own Virtual Network (Vnet)**. Select the virtual network we created earlier from the **Virtual Network** dropdown. Provide the public subnet name, private subnet name, public subnet CIDR range, and private subnet CIDR range that we created earlier:



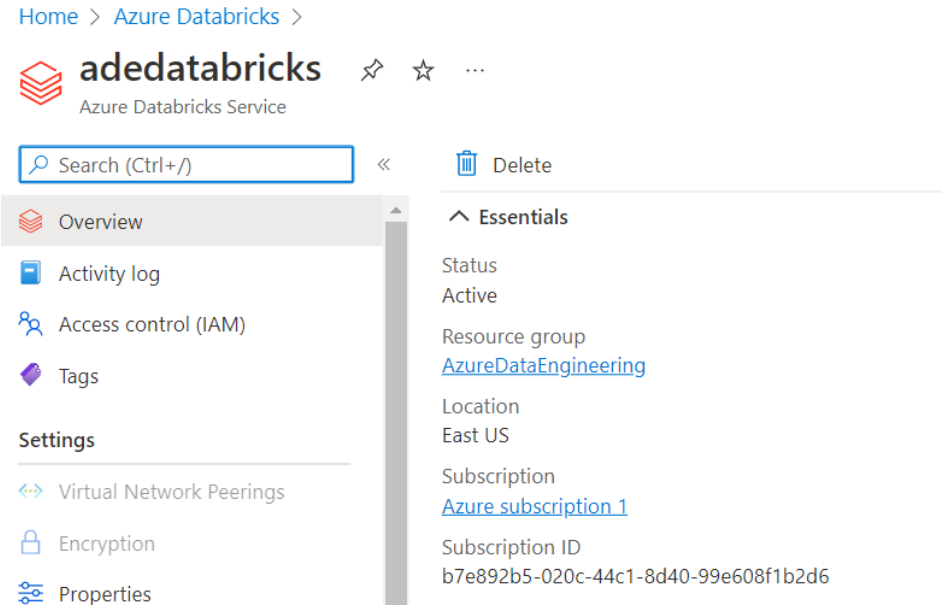
Azure Databricks uses one public and one private subnet. The public subnet allows you to access the Azure Databricks control plane. Databricks clusters are deployed on the private subnet. It's recommended to not provision any other service in the Databricks private subnet.

> **Note**
> If the public and private subnet don't exist, they'll be created as part of your Azure Databricks workspace automatically.

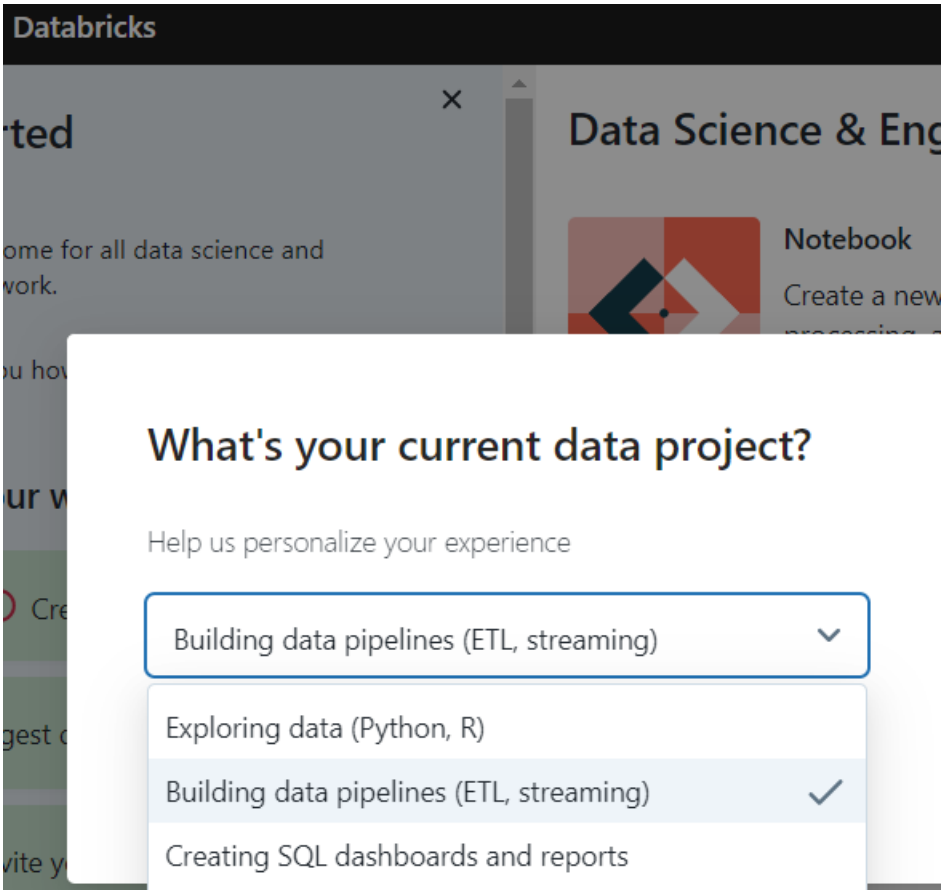
13. Click **Review + create** and then click **Create** after successful validation. Next we'll provision the Databricks workspace.



14. Once the Azure Databricks Workspace has been deployed, select **Go to resource** to go to the newly created Azure Databricks workspace:



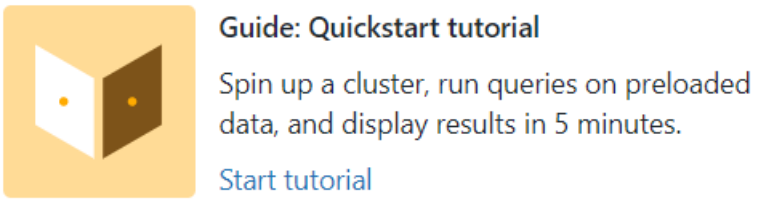
15. Click on **Launch Workspace** to open the Azure Databricks workspace, the Databricks launch will ask you for your preference and we'll choose **Pipelines** and select **Finish**:



Databricks authentication is done through Azure **Active Directory** (**AD**). The AD username is displayed at the top-right corner of the workspace page.

An Azure Databricks workspace allows us to create clusters, notebooks, jobs, data sources, and folders so that we can write data transformation or MLFlow experiments. It also helps us organize multiple projects into different folders.

Take 5 minutes to go through the Quickstart, select **Start tutorial**

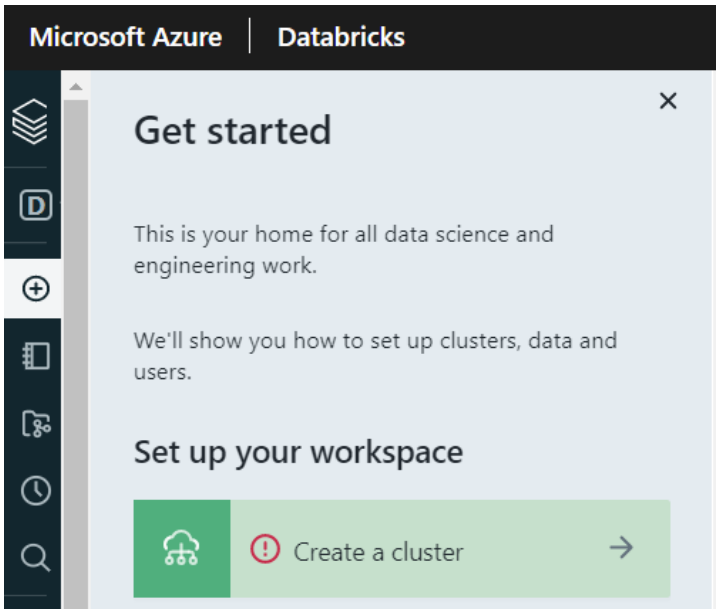


Now, let's create some Azure Databricks clusters.

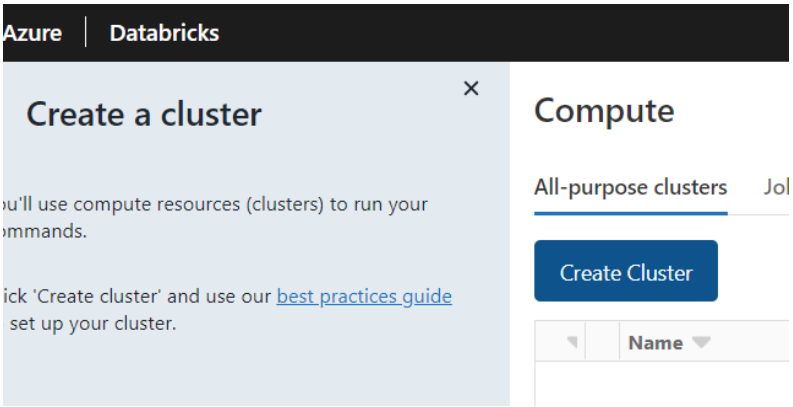
Creating Azure Databricks clusters

Follow these steps:

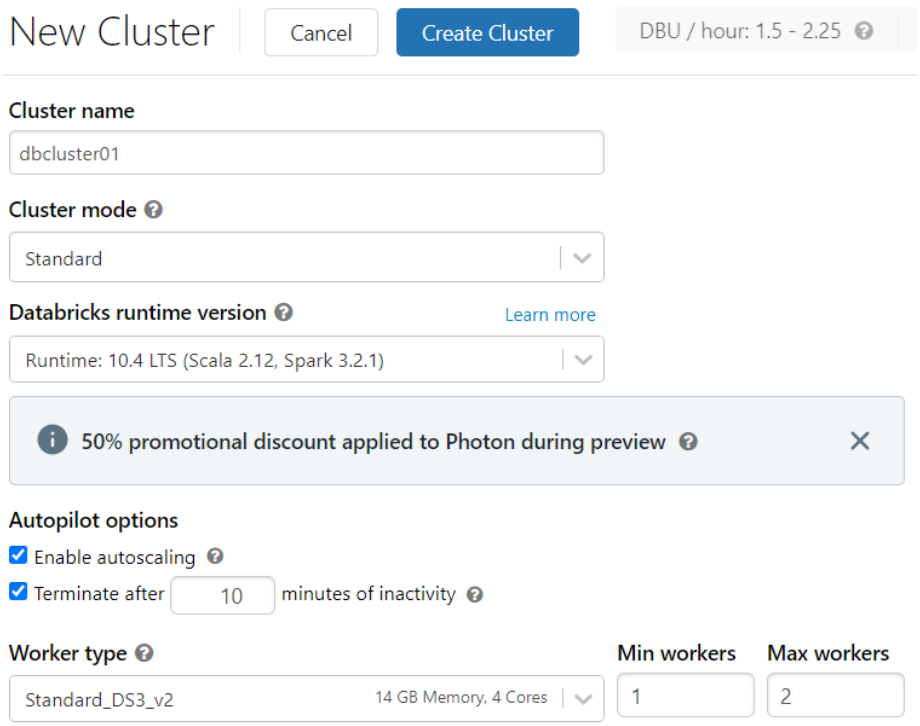
1. To create a cluster, select **Create a cluster** from the Get started menu of the Databricks workspace:



There are two types of clusters: **Interactive** and **Automated**. Interactive clusters are created manually by users so that they can interactively analyze the data while working on, or even developing, a data engineering solution. Automated clusters are created automatically when a job starts and are terminated as and when the job completes.



2. Click **Create Cluster** to create a new cluster. On the **New Cluster** page, provide a cluster name of **`dbcluster01`**. Then, set **Cluster Mode** to **Standard**, **Default Worker Standard_DS3_V2**, **Terminate after** to **`10`** minutes of inactivity, **Min Workers** to **`1`**, **Max Workers** to **`2`**, and leave the rest of the options with their defaults:



There are two cluster modes: **Standard** and **High Concurrency**. Standard cluster mode uses single-user clusters, optimized to run tasks one at a time. In a standard cluster, if there are tasks from multiple users, then a failure in one task may cause the other task to fail as well. Note that one task can consume all the cluster's resources, causing another task to wait. High Concurrency cluster mode is optimized to run multiple tasks in parallel; however, it only supports R, Python, and SQL workloads and doesn't supports Scala.
These autoscaling options allow Databricks to provision as many clusters as required to process a task within the limit, as specified by the **Min Workers** and **Max Workers** options.

The **Terminate after** option terminates the clusters when there's no activity for a given amount of time. In our case, the cluster will auto terminate after 10 minutes of inactivity. This option helps save costs.

There are two types of cluster nodes: **Worker Type** and **Driver Type**. The driver type node is responsible for maintaining a notebook's state information, interpreting the commands being run from a notebook or a library, and running the Apache Spark master. The worker type nodes are the Spark executor nodes, and these are responsible for distributed data processing.

The **Advanced Options** section can be used to configure Spark configuration parameters, environment variables, tags, configure SSH in the clusters, enable logging, and run custom initialization scripts at the time of cluster creation.

3. Click **Create Cluster** to create the cluster. It will take around 5-10 minutes to create the cluster and may take more time, depending on the number of worker nodes that have been selected:

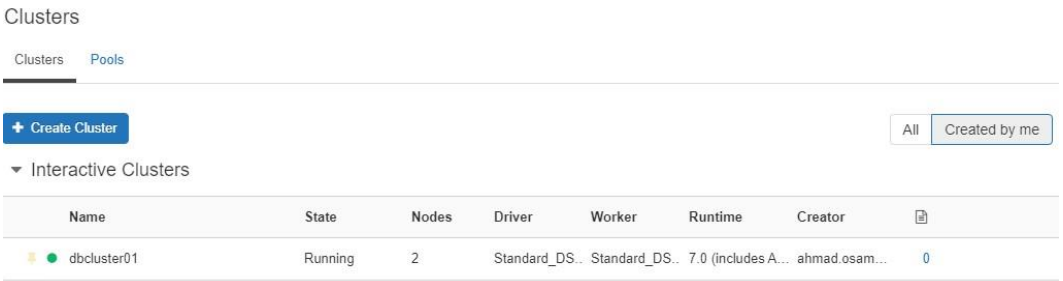


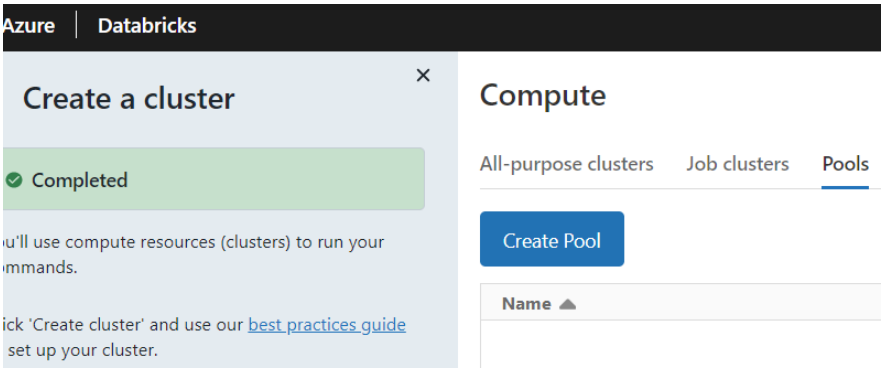
Figure 9.15 – Viewing your clusters

Observe that there are two nodes – one driver and worker – even though the max number of worker nodes is two. Databricks will only create the two worker nodes when the autoscaling condition you've set is reached.

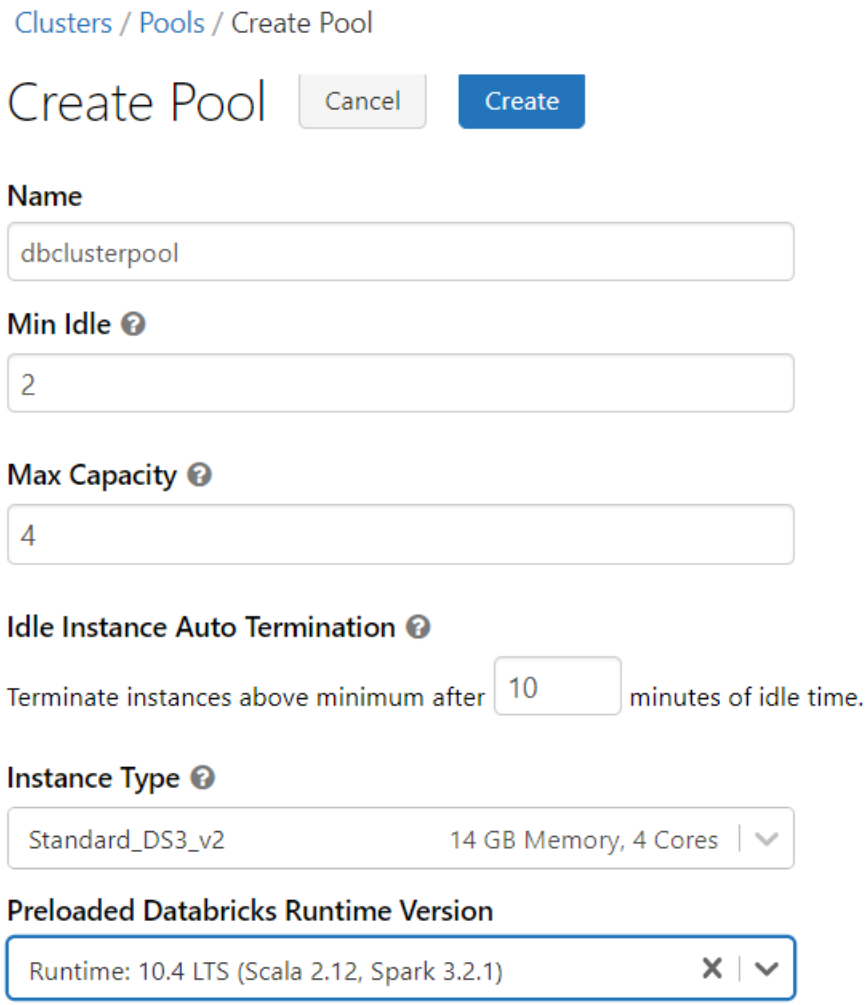
Creating Azure Databricks Pools

Azure Databricks Pools optimize autoscaling by keeping a set of idle, ready-to-use instances without the need for creating instances when required. These idle instances are not charged for. To create Azure Databricks Pools, execute the following steps:

1. In your Azure Databricks workspace, on the **Clusters** page, select the **Pools**



2. Select **Create Pool** to create a new pool. Provide the pool's name, then set **Min Idle** to `2`, **Max Capacity** to `4`, and **Idle Instance Auto Termination** to `10`. Leave **Instance Type** as its default of **Standard_DS3_v2** and set **Preloaded Databricks Runtime Version** to **10.4 LTS**:

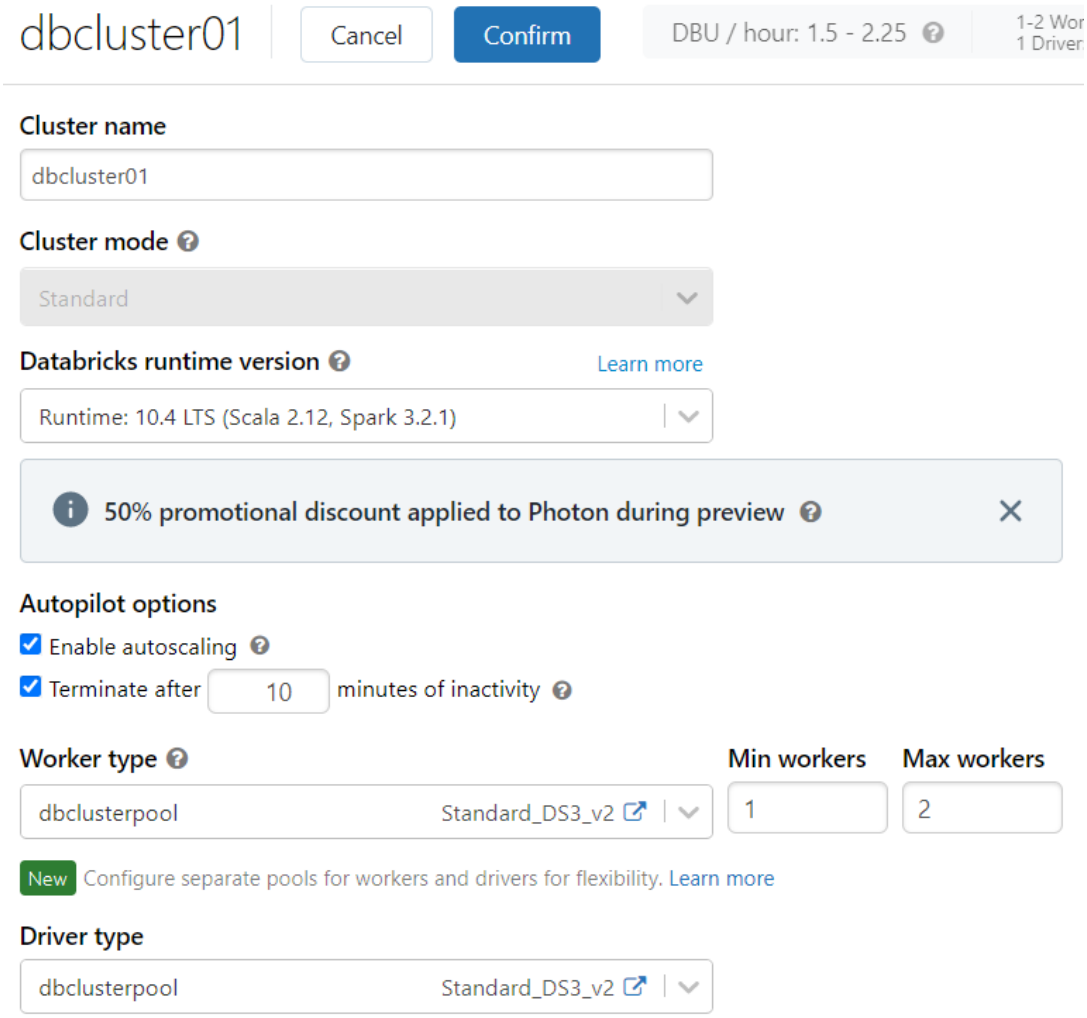


Min Idle specifies the number of instances that will be kept idle and available and won't terminate. The **Idle Instance Auto Terminate** settings doesn't apply to these instances.

Max Capacity limits the maximum number of instances to this number, including idle and running ones. This helps with managing cloud quotas and their costs. The Azure Databricks runtime is a set of core components or software that run on your clusters. There are different runtimes, depending on the type of workload you have. To find out

more about the Azure Databricks runtime, please visit **`https://docs.microsoft.com/en-us/azure/databricks/runtime/`**.

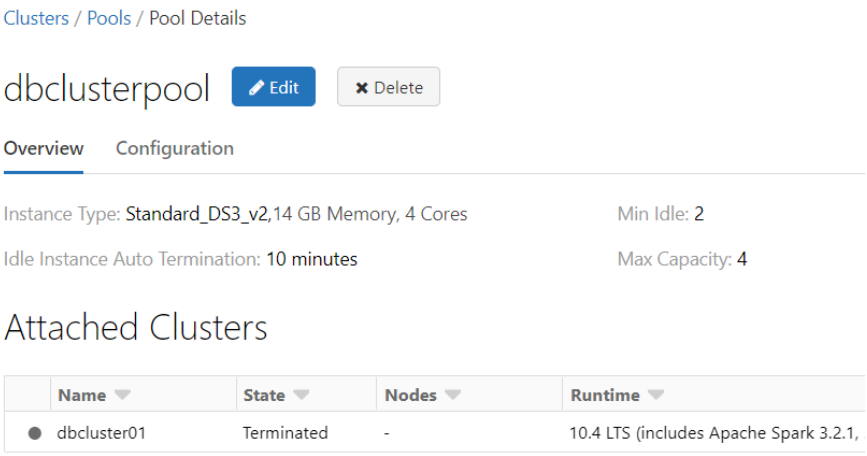
3. Click **Create** to create the pool. We can attach a new of existing or cluster to a pool by specifying the pool name under the **Pool** option. In the workspace, navigate to the **Clusters** page and select **dbcluster01**, which we created in *Step 2* of the previous section. On the **dbcluster01** page, click **Edit** and select **dbclusterpool** from the **Worker type** drop-down list and from the **Driver type** drop-down list:



4. Click **Confirm** to apply these changes. The cluster will now show up in the **Attached Clusters** list:

We can add multiple clusters to a pool; however, we should modify the number of idle clusters and their maximum instance capacity accordingly.

Whenever an instance, such as **`dbcluster01`**, requires an instance, it'll attempt to allocate the pool's idle instance. If an idle instance isn't available, the pool expands to provision new instances.



Congratulations you've created and configured your Azure Databricks cluster