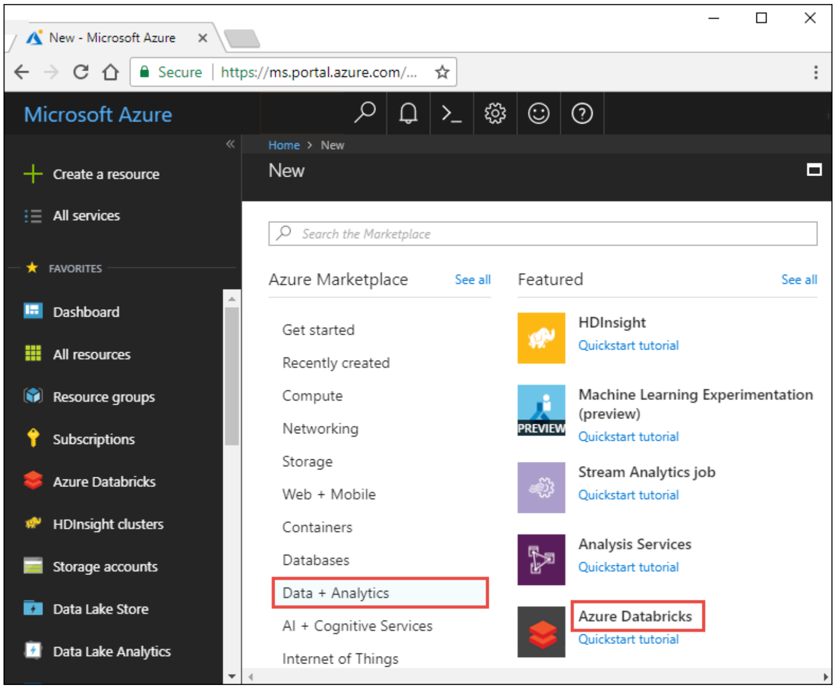
Azure Data Bricks

LAB Overview

This lab introduces you how to create Azure Data Bricks and launch Spark SQL job using them.

Task 1: Create an Azure Databricks workspace

1. Sign in the Azure portal at https://portal.azure.com
2. In the Azure portal, select **Create a resource > Data + Analytics > Azure Databricks**.



3. Under **Azure Databricks Service**, provide the values to create a Databricks workspace:

   - **Workspace name:** databricksstudentXX
   - **Location:** West Europe

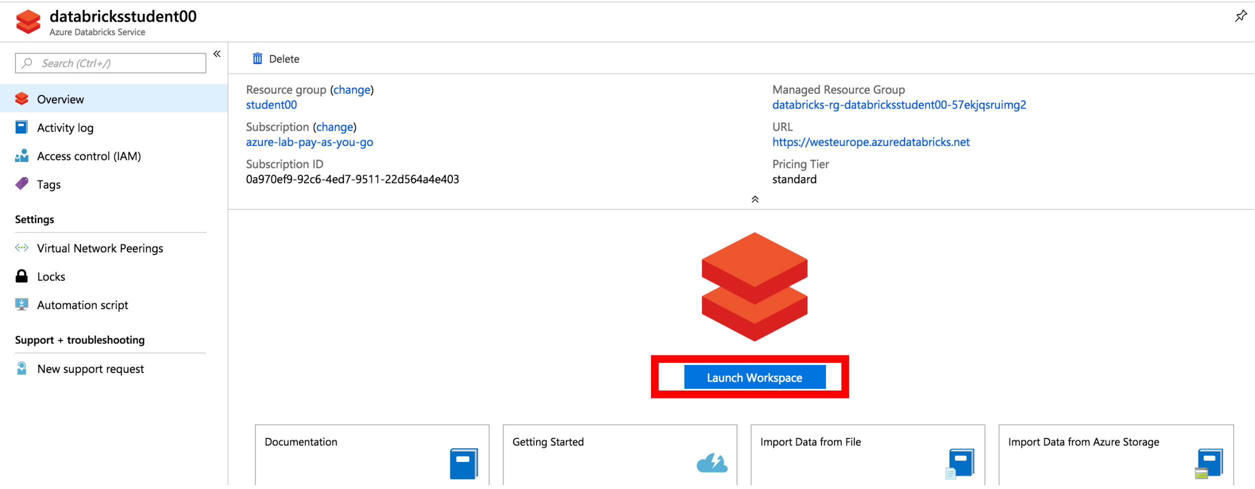
- **Subscription:** XXXXXX
- **Resource Group:**
    - **Create New**: studentXX
- **PricingTier:** Standard

When you finish, click on button **Create.**

Task 2: Create a Spark in Databricks

In this section you will learn how to upload some blob file in Azure Storage Account using Azure Portal.

1. In the Azure portal, go to the Databricks workspace that you created, and then click Launch Workspace.



databricksstudent00
Azure Databricks Service

Resource group (change)
student00

Managed Resource Group
databricks-rg-databricksstudent00-57ekjqsruimg2

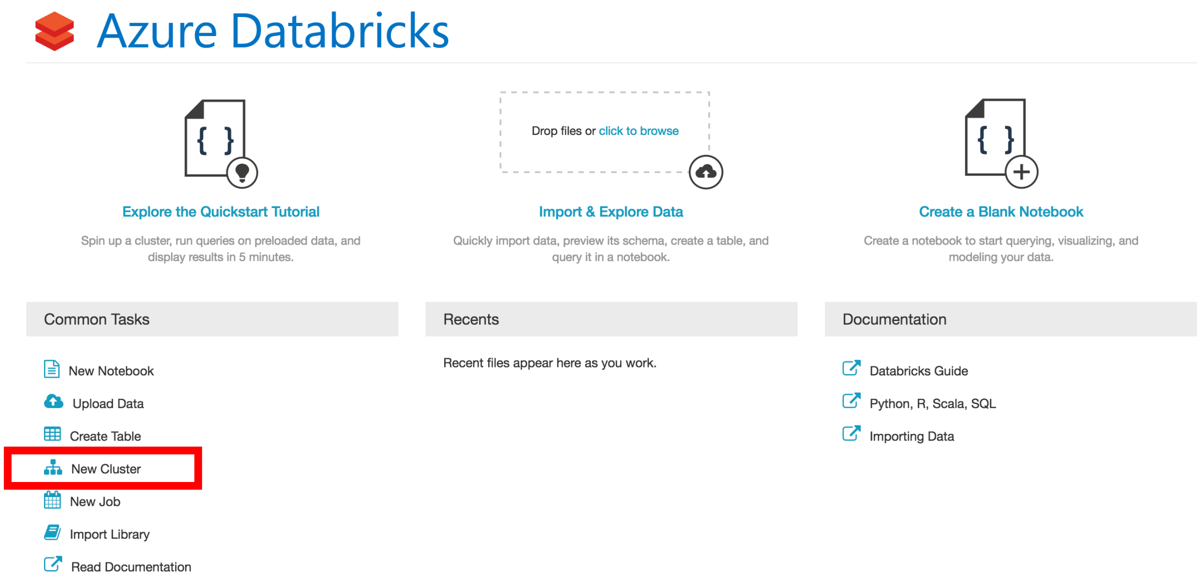
Subscription (change)
azure-lab-pay-as-you-go

URL
https://westeurope.azuredatabricks.net

Subscription ID
0a970ef9-92c6-4ed7-9511-22d564a4e403

Pricing Tier
standard

2. You are redirected to the Azure Databricks portal. From the portal, click Cluster.



Azure Databricks

Drop files or click to browse

Explore the Quickstart Tutorial
Spin up a cluster, run queries on preloaded data, and display results in 5 minutes.

Import & Explore Data
Quickly import data, preview its schema, create a table, and query it in a notebook.

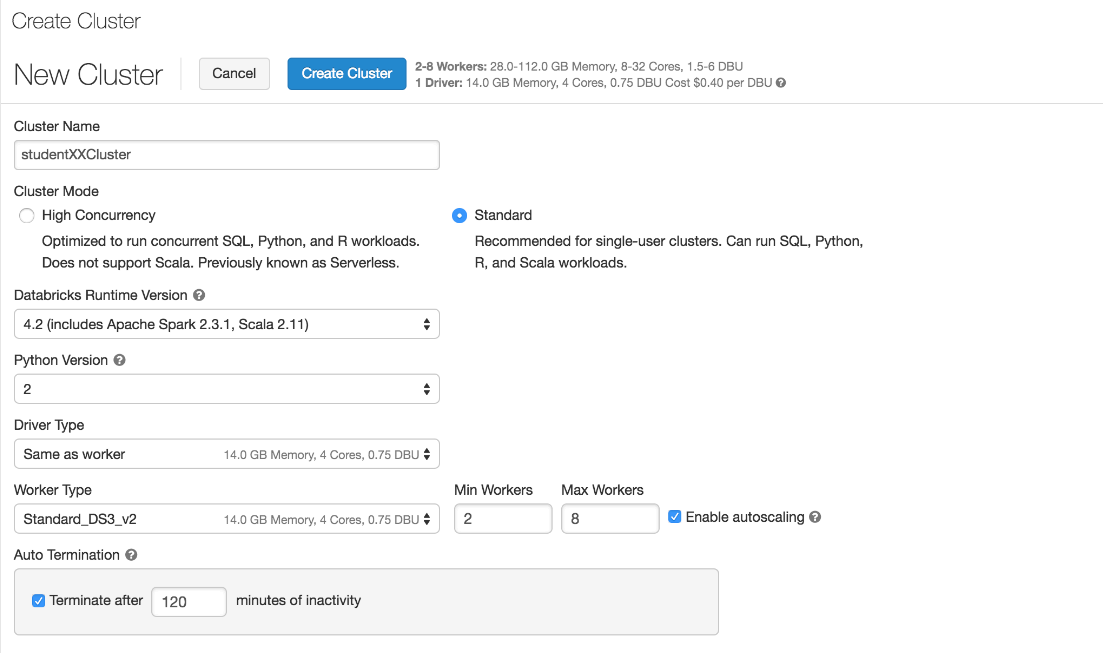
Create a Blank Notebook
Create a notebook to start querying, visualizing, and modeling your data.

Common Tasks
- New Notebook
- Upload Data
- Create Table
- New Cluster
- New Job
- Import Library
- Read Documentation

Recents
Recent files appear here as you work.

Documentation
- Databricks Guide
- Python, R, Scala, SQL
- Importing Data

3. In the New cluster page, provide the values to create a cluster.
   - **Cluster Name**: studentxxCluster
   - **Databricks Runtime Version**: 4.0
   - **Terminate after**: 120



Create Cluster

New Cluster | Cancel | Create Cluster

2-8 Workers: 28.0-112.0 GB Memory, 8-32 Cores, 1.5-6 DBU
1 Driver: 14.0 GB Memory, 4 Cores, 0.75 DBU Cost $0.40 per DBU

Cluster Name: studentXXCluster

Cluster Mode
- ○ High Concurrency — Optimized to run concurrent SQL, Python, and R workloads. Does not support Scala. Previously known as Serverless.
- ● Standard — Recommended for single-user clusters. Can run SQL, Python, R, and Scala workloads.

Databricks Runtime Version: 4.2 (includes Apache Spark 2.3.1, Scala 2.11)

Python Version: 2

Driver Type: Same as worker — 14.0 GB Memory, 4 Cores, 0.75 DBU

Worker Type: Standard_DS3_v2 — 14.0 GB Memory, 4 Cores, 0.75 DBU | Min Workers: 2 | Max Workers: 8 | ☑ Enable autoscaling

Auto Termination
☑ Terminate after 120 minutes of inactivity

Next click on **Create Cluster**.

Task 3: Download a sample data file

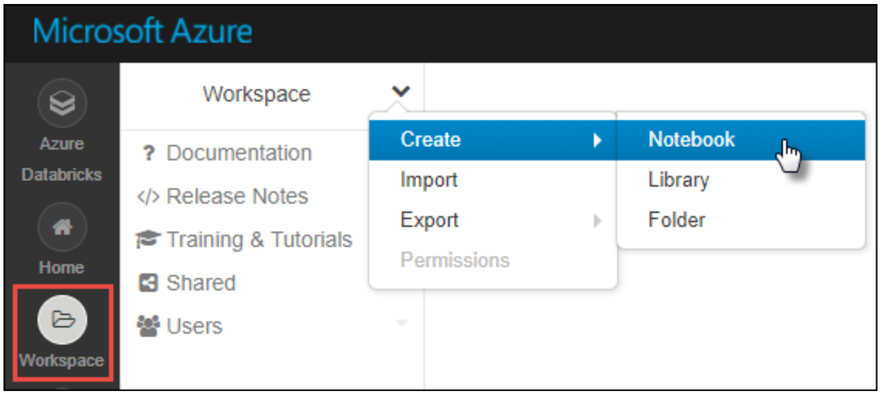
Download a sample JSON data file and save it into Azure blob storage.

1. Download this sample JSON data file from Github onto your local computer. Right-click and save as to save the raw file locally. https://raw.githubusercontent.com/Azure/usql/master/Examples/Samples/Data/json/radiowebsite/small_radio_json.json
2. Go to Azure Storage Account **studentXX**.
3. Open the storage account in the Azure portal.
4. Select **Blobs**.
5. Select **+ Container** to create a new empty container.
6. Provide a **Name** for the container **databricks**.
7. Select **Private (non anonymous access)** access level.
8. Once the container is created, select the container name.
9. Select the **Upload** button.
10. On the **Files** page, select the **Folder icon** to browse and select the sample file small_radio_json.json for upload.
11. Select **Upload** to upload the file.

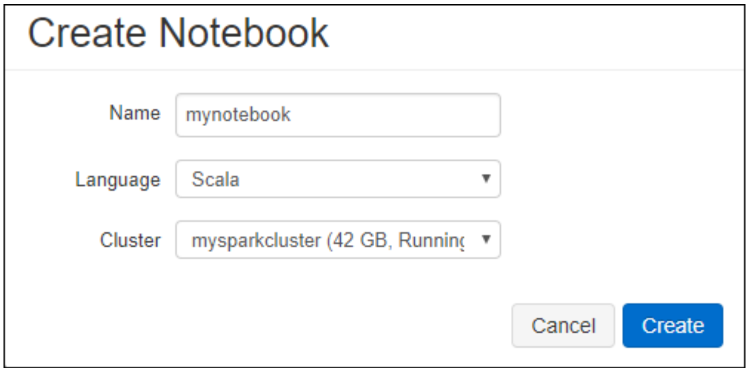
Task 4: Run a Spark SQL job

In this section you learn how to management of Azure Storage Account using Azure Storage Explorer.

1. In the left pane, click Workspace. From the Workspace drop-down, click Create, and then click Notebook.



2. In the **Create Notebook** dialog box, enter a name, select **Scala** as the language, and select the Spark cluster that you created earlier.



Create Notebook

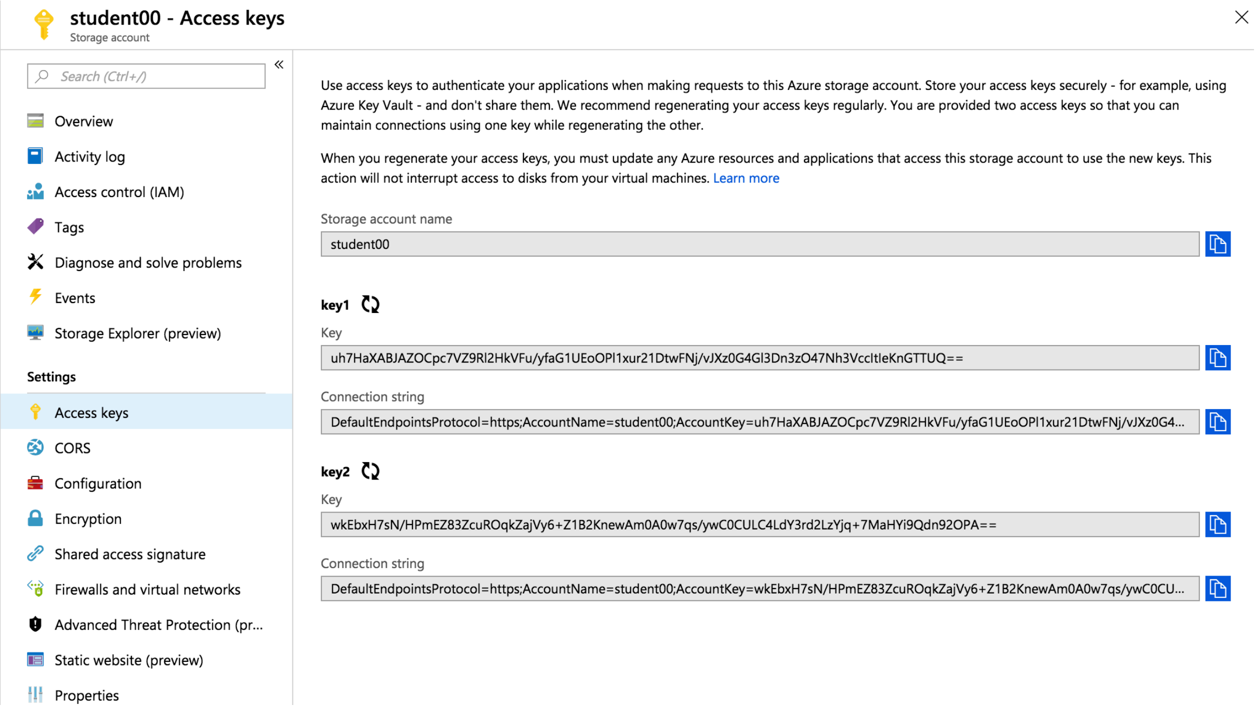
Name: mynotebook

Language: Scala

Cluster: mysparkcluster (42 GB, Running

3. Click **Create.**

4. Mount the storage account with DBFS. The Azure Storage account path is mounted to /mnt/mypath.
5. Paste the code to notebook:
   dbutils.fs.mount(source = "wasbs://databricks@{YOUR STORAGE ACCOUNT NAME}.blob.core.windows.net/", mountPoint = "/mnt/mypath", extraConfigs = Map("fs.azure.account.key.{YOUR STORAGE ACCOUNT NAME}.blob.core.windows.net" -> "{YOUR STORAGE ACCOUNT ACCESS KEY}"))
6. Open Azure Storage account **studentXX** and replace values from step 5 {YOUR STORAGE ACCOUNT NAME} with storage account name and {YOUR STORAGE ACCOUNT ACCESS KEY} using key1 value.



student00 - Access keys
Storage account

Use access keys to authenticate your applications when making requests to this Azure storage account. Store your access keys securely - for example, using Azure Key Vault - and don't share them. We recommend regenerating your access keys regularly. You are provided two access keys so that you can maintain connections using one key while regenerating the other.

When you regenerate your access keys, you must update any Azure resources and applications that access this storage account to use the new keys. This action will not interrupt access to disks from your virtual machines. Learn more

Storage account name
student00

key1

Key
uh7HaXABJAZOCpc7VZ9Rl2HkVFu/yfaG1UEoOPl1xur21DtwFNj/vJXz0G4Gl3Dn3zO47Nh3VccltleKnGTTUQ==

Connection string
DefaultEndpointsProtocol=https;AccountName=student00;AccountKey=uh7HaXABJAZOCpc7VZ9Rl2HkVFu/yfaG1UEoOPl1xur21DtwFNj/vJXz0G4...

key2

Key
wkEbxH7sN/HPmEZ83ZcuROqkZajVy6+Z1B2KnewAm0A0w7qs/ywC0CULC4LdY3rd2LzYjq+7MaHYi9Qdn92OPA==

Connection string
DefaultEndpointsProtocol=https;AccountName=student00;AccountKey=wkEbxH7sN/HPmEZ83ZcuROqkZajVy6+Z1B2KnewAm0A0w7qs/ywC0CU...

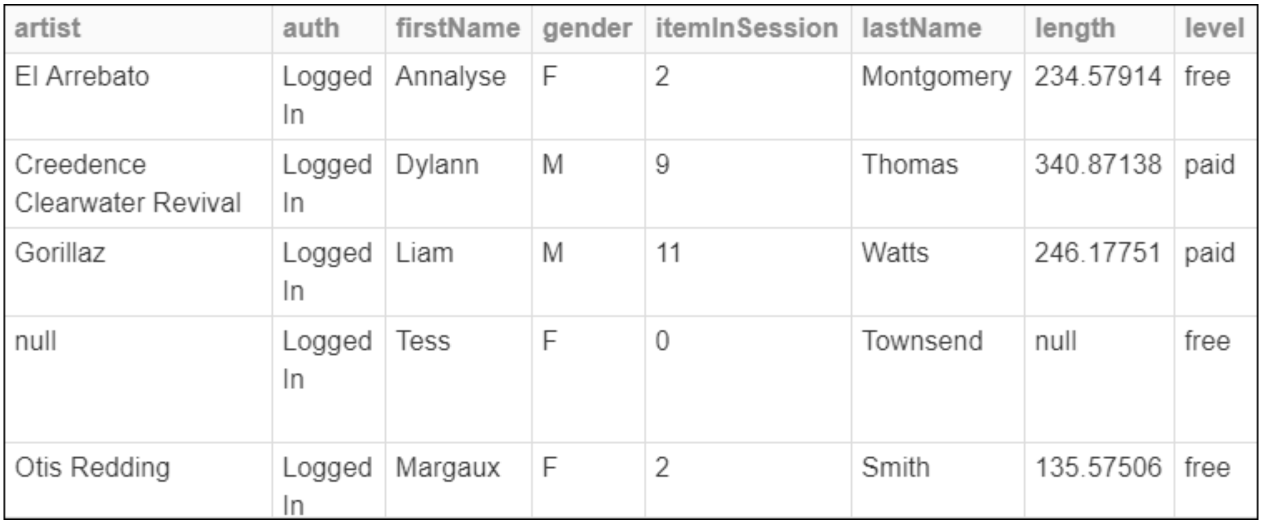
7. Run a SQL statement in notebook to create a temporary table using data from the sample JSON data file, **small_radio_json.json**.

```
%sql

DROP TABLE IF EXISTS radio_sample_data;

CREATE TABLE radio_sample_data

USING json

OPTIONS (

 path "/mnt/mypath/small_radio_json.json"

)
```

8. Select data from temporary table using command:

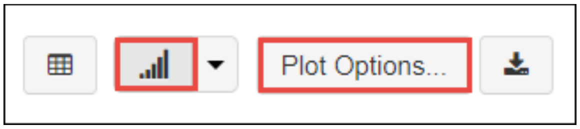
```
%sql

SELECT * from radio_sample_data
```

9. You will see a tabular output.

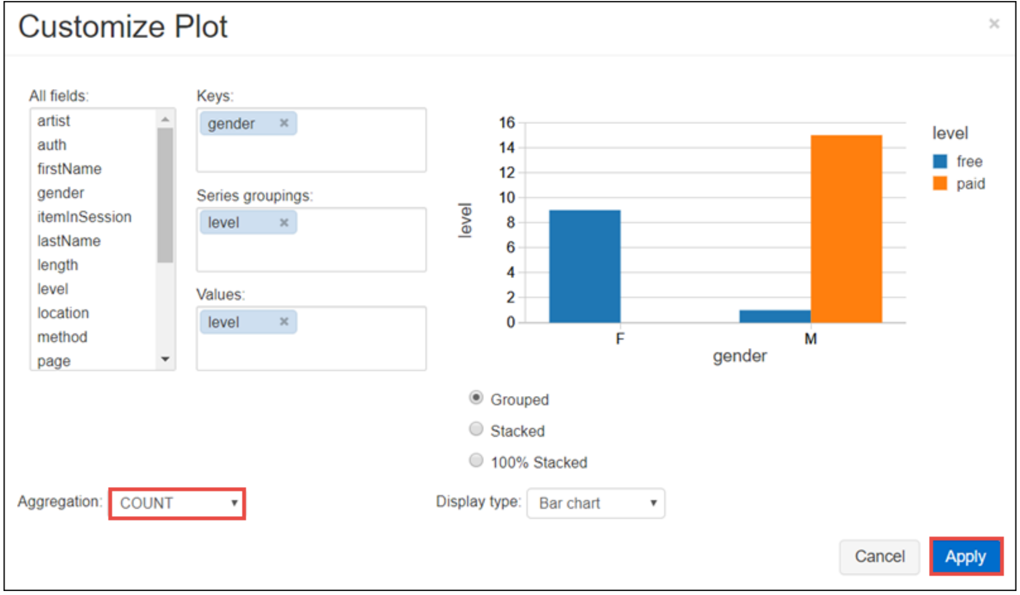


artist	auth	firstName	gender	itemInSession	lastName	length	level
El Arrebato	Logged In	Annalyse	F	2	Montgomery	234.57914	free
Creedence Clearwater Revival	Logged In	Dylann	M	9	Thomas	340.87138	paid
Gorillaz	Logged In	Liam	M	11	Watts	246.17751	paid
null	Logged In	Tess	F	0	Townsend	null	free
Otis Redding	Logged In	Margaux	F	2	Smith	135.57506	free

10. You now create a visual representation of this data to show for each gender, how many users have free accounts and how many are paid subscribers. From the bottom of the tabular output, click the Bar chart icon, and then click Plot Options.



11. In Customize Plot, drag-and-drop values as shown in the screenshot.
    a. Set **Keys** to **gender**.
    b. Set **Series groupings** to **level**.
    c. Set **Values** to **level**.
    d. Set **Aggregation** to **COUNT**.



Customize Plot

Keys: gender ×

Series groupings: level ×

Values: level ×

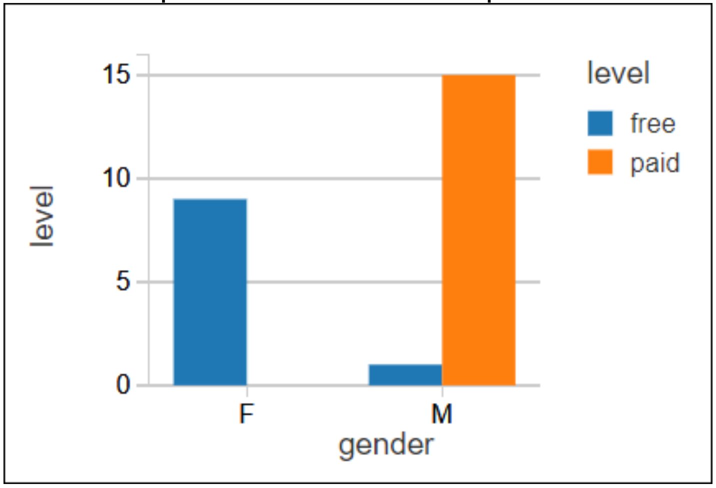
● Grouped
○ Stacked
○ 100% Stacked

Aggregation: COUNT ▾

Display type: Bar chart ▾

Click **Apply.**

12. The output shows the visual representation as depicted in the following screenshot.



13. Stop the cluster.