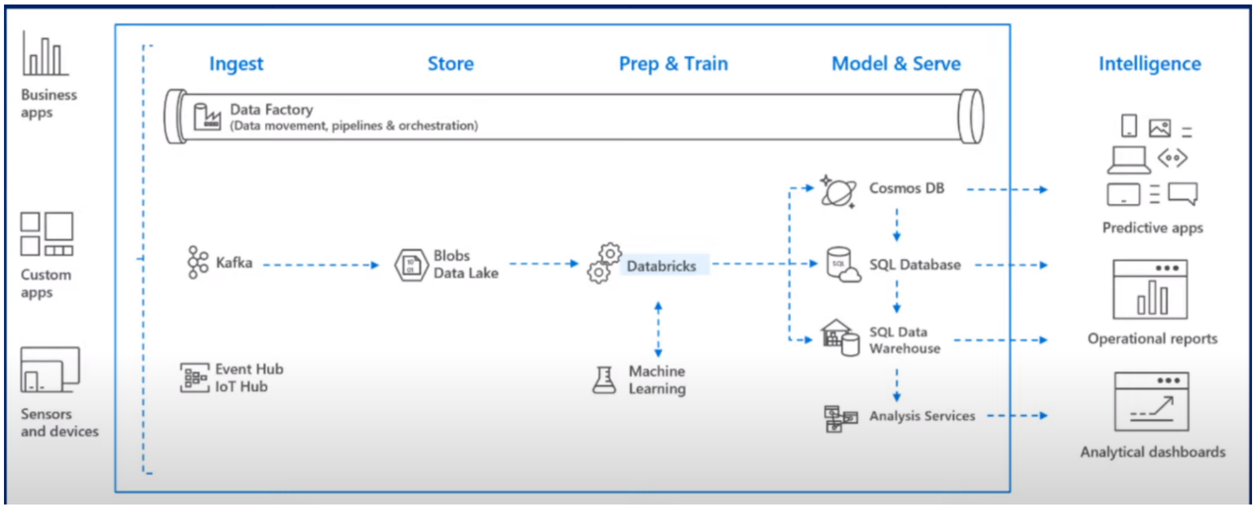
Azure Databricks

Azure Documentation

AWS Documentation

Azure Databricks is a fast, easy to use, and collaborative Apache Spark-based analytics platform optimized for Azure. It makes it easy to process large datasets, train machine learning models, and run real-time queries on streaming data.



Interactive workspace

The Azure Databricks workspace provides an interactive notebook interface for running queries and developing applications. You can easily upload your data, write queries in languages including Scala, Python, SQL, and R, and visualize the results. The workspace allows you to collaborate with teammates, share notebooks, and run notebooks concurrently.
You get a full set of tools for the entire data and AI lifecycle in a single place. Easily ingest data, explore and visualize it, build models, train and evaluate them, deploy them, and monitor them. Share your work with colleagues and manage access and permissions.
The workspace includes:

- Notebooks: Interactive documents that include code, visualizations, and narrative text.
- Dashboard: A web interface for monitoring clusters, jobs, and runs.
- Data: A file browser for uploading and managing data.
- Experiments: A tool for organizing runs into experiments to track and compare models.
- Models: A registry for managing machine learning models.
- Jobs: A tool for scheduling and monitoring jobs.
- Clusters: A interface for provisioning and managing clusters.
- Access control: Granular control over who can access, edit, and run content.

Integrated with Azure

Azure Databricks is fully integrated with Azure, so you can leverage additional Azure services. Pull in data from Azure Storage, Azure Data Lake Storage, or Azure Cosmos DB. Use MLflow to manage machine learning models and register them in Azure Machine Learning. Or deploy models as web services with Azure Machine Learning or as Functions with Azure Functions.
Azure Databricks also integrates with Azure Active Directory for authentication and authorization, and you can use Azure Data

Factory for orchestrating data movement. You get all the benefits of the Azure cloud, including global data centers, advanced networking, and enterprise-grade security and compliance.

Managed platform

Azure Databricks is a fully managed service. We handle the infrastructure for you so you can focus on building applications and analyzing data. We automatically provision and scale clusters to meet your needs. And we keep the platform up to date with the latest versions of Apache Spark and libraries as well as regular security patches.
You get all of this without having to manage any servers or runtime environments. We take care of cluster deployment, configuration, and optimization. And we monitor the health of the clusters and the service and proactively work to resolve issues.

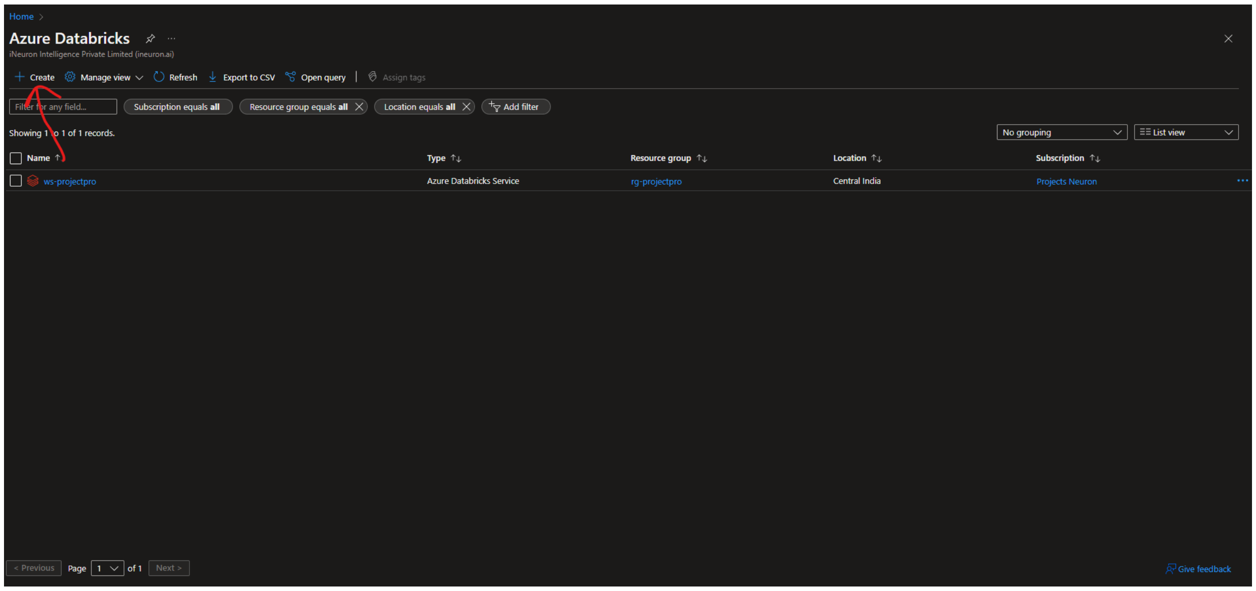
Rich set of libraries

Azure Databricks provides a rich set of libraries that you can easily install and use. In addition to the Apache Spark libraries, we include libraries for machine learning, graph processing, and streaming. We also include Delta Lake, an open format storage layer that brings reliability to data lakes.
You get a curated set of libraries that have been tested and optimized to work on Databricks. Easily find and install the libraries you need without having to hunt through repositories.

Caution - We can’t use Azure Free trail subscription to work with Azure Databricks. We need to upgrade it as pay-as-you-go subscription.

Creating Databricks Workspace

Search for “Azure Databricks” in the search bar and then click on “Create” button



Showing 1 to 1 of 1 records.

Name	Type	Resource group	Location	Subscription
ws-projectpro	Azure Databricks Service	rg-projectpro	Central India	Projects Neuron

Use Databricks for free

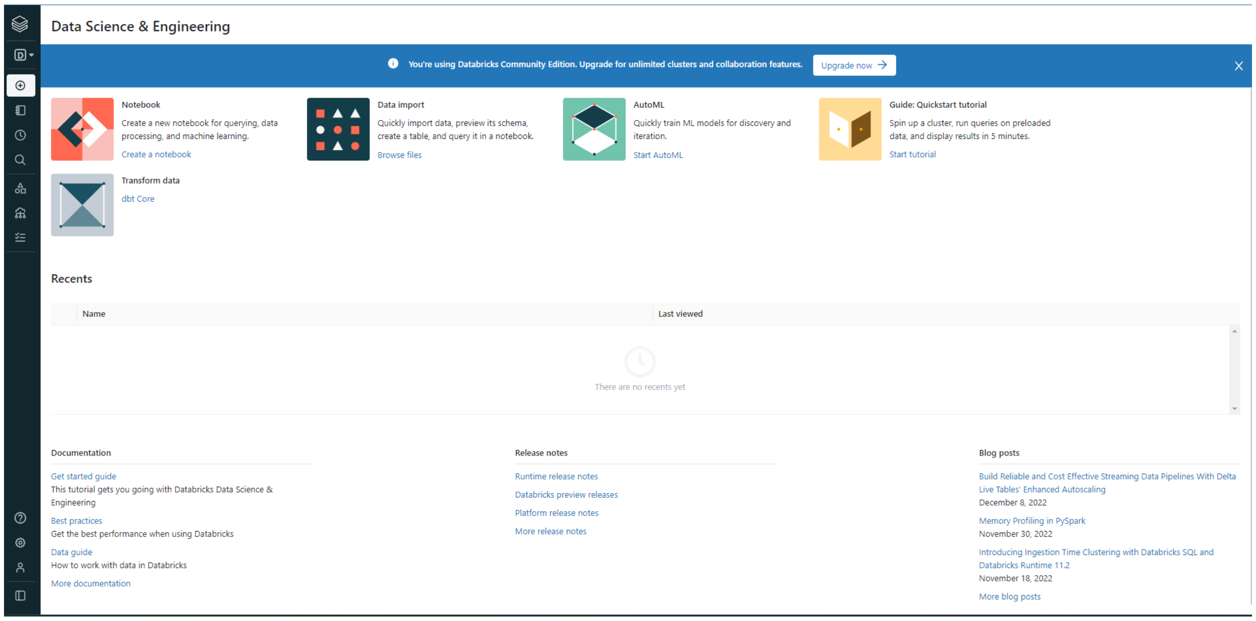
Use the Databricks community edition. Link

Workspace in Databricks

A workspace is an environment for accessing all our Databricks assets. The workspace organizes below objects into folders

- Notebooks
- Libraries
- Experiments

We can manage the workspace using the workspace UI, Databricks CLI or Databricks REST APIs.


Data Science & Engineering

You're using Databricks Community Edition. Upgrade for unlimited clusters and collaboration features. Upgrade now

Notebook
Create a new notebook for querying, data processing, and machine learning.
Create a notebook

Data import
Quickly import data, preview its schema, create a table, and query it in a notebook.
Browse files

AutoML
Quickly train ML models for discovery and iteration.
Start AutoML

Guide: Quickstart tutorial
Spin up a cluster, run queries on preloaded data, and display results in 5 minutes.
Start tutorial

Transform data
dbt Core

Recents

Name	Last viewed

There are no recents yet

Documentation
Get started guide
This tutorial gets you going with Databricks Data Science & Engineering
Best practices
Get the best performance when using Databricks
Data guide
How to work with data in Databricks
More documentation

Release notes
Runtime release notes
Databricks preview releases
Platform release notes
More release notes

Blog posts
Build Reliable and Cost Effective Streaming Data Pipelines With Delta Live Tables' Enhanced Autoscaling
December 8, 2022
Memory Profiling in PySpark
November 30, 2022
Introducing Ingestion Time Clustering with Databricks SQL and Databricks Runtime 11.2
November 18, 2022
More blog posts

The “?” icon can be help if you want to report a bug, view documentation etc.

Workspace Assests

- Cluster/Compute → A cluster refers to a group of virtual machines that are configured and managed to run distributed data processing jobs, typically using Apache Spark. Clusters in Databricks are used to perform large-scale data processing tasks such as ETL (Extract, Transform, Load) jobs, machine learning algorithms, and data analytics. Clusters can be customized based on the requirements of the specific workload, with options for configuring the number and size of virtual machines, selecting the type of virtual machine instances, and configuring the amount of memory and processing power allocated to each node in the cluster.
- Notebook → A notebook is an interactive document that allows you to create and execute code, visualize data, and communicate insights. Notebooks are organized into cells, which can contain code, markdown text, or visualizations. We can use notebooks in Databricks for data exploration, data analysis, machine learning, and collaboration. Notebooks allow you to write code in a variety of languages, including Python, R, Scala, SQL, and more. Notebooks in Databricks also come with a number of built-in features that make them particularly useful for data science workflows. For example, you can use Databricks notebooks to connect to and query data stored in a variety of data sources, such as SQL databases, data lakes, and data warehouses. We can also use notebooks to create visualizations and dashboards that can be shared with others in your organization.
- Jobs → Jobs is a mechanism for running code in Databricks.
- Libraries → A library is a collection of code, dependencies, and settings that can be used to extend the functionality of your Databricks workspace. Libraries can contain various types of files, such as Python packages, JAR files, and files in other formats. By adding a library to your Databricks workspace, you can access new functionality that is not already built into Databricks. For example, you might want to use a particular Python library that is not included by default in Databricks, or you might want to add a custom library that you have developed yourself. You can add libraries to your workspace by uploading them directly, or by referencing them from a repository such as Maven, PyPI, or CRAN. Once you have added a library, you can use it in your notebooks and jobs by importing its modules or using its functions.
- Data → Data refers to any piece of information that is stored in a structured or unstructured format within the platform. This can include files, tables, streams, and other types of data sources. Data can be ingested into Databricks from various sources, such as cloud storage platforms like Amazon S3 or Azure Blob Storage, or from other data services such as Apache Kafka or JDBC-compliant databases. Once the data is in Databricks, it can be manipulated, transformed, and analyzed using various tools and programming languages such as SQL, Python, R, or Scala. Databricks also provides a number of pre-built libraries and APIs for machine learning and other advanced analytics tasks, which can be used to derive insights and make data-driven decisions.

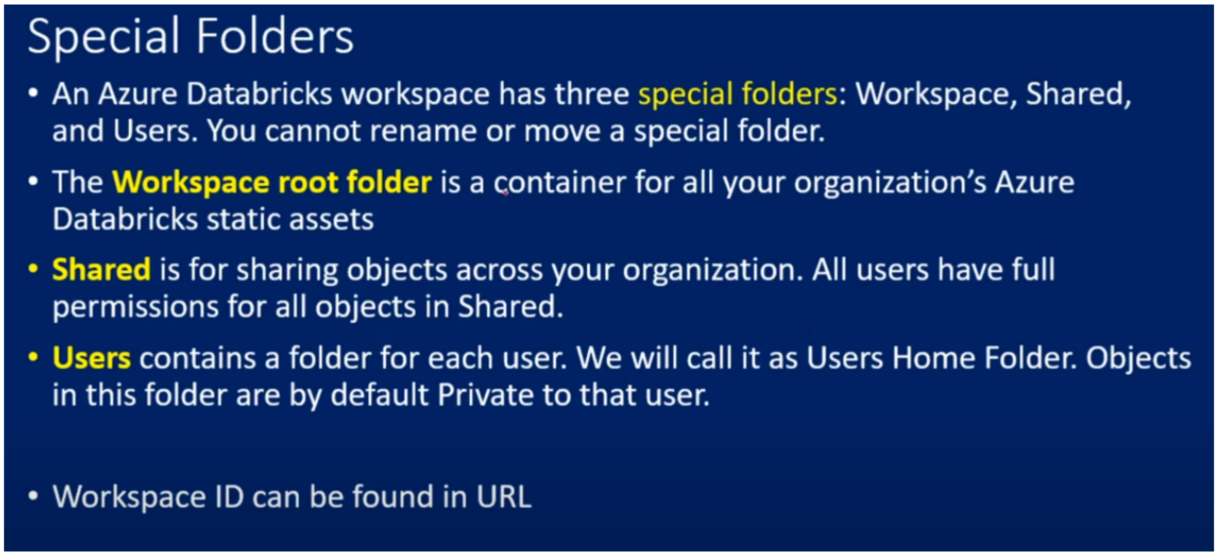
- Experiments → Experiments refer to a set of machine learning (ML) workflows that allow you to track and compare the performance of different models on your dataset. These experiments can help you find the best model for your data and business problem.

  Databricks experiments allow you to:

  1. Organize and track the results of multiple ML models and hyperparameter settings
  2. Compare the performance of different models on your dataset using metrics such as accuracy, precision, recall, F1-score, etc.
  3. Visualize and analyze the results of the experiments using charts and tables
  4. Share the results with your team members or stakeholders

  To run an experiment in Databricks, you can use the Databricks MLflow library, which provides an API for logging and tracking the experiment runs. You can use this library to log the parameters, metrics, and artifacts of your experiment runs, and then compare and analyze the results in the MLflow UI or programmatically.

  Overall, experiments in Databricks can help you optimize your ML models and improve their accuracy and performance, which can lead to better business outcomes.



Special Folders

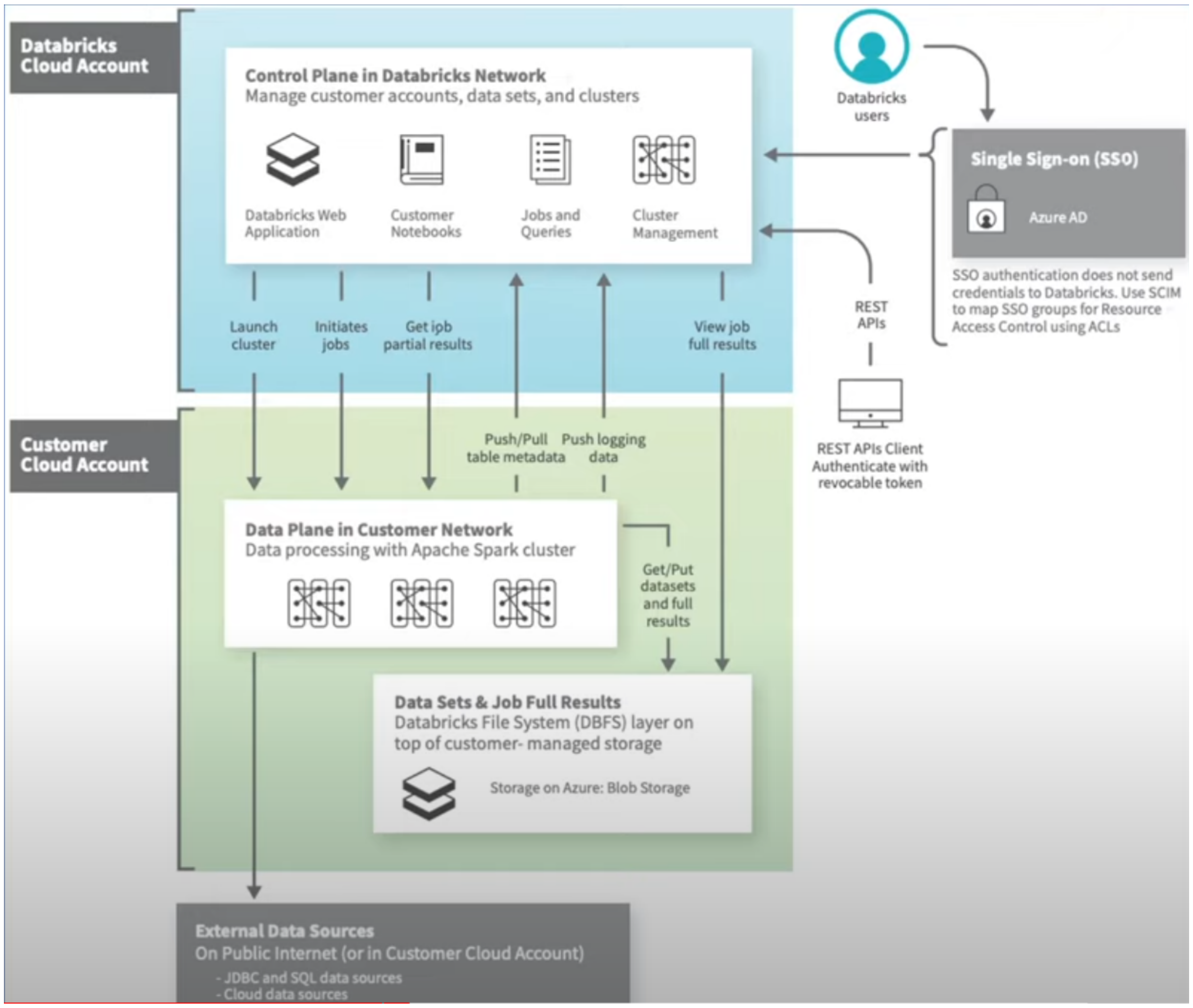
- An Azure Databricks workspace has three special folders: Workspace, Shared, and Users. You cannot rename or move a special folder.
- The **Workspace root folder** is a container for all your organization's Azure Databricks static assets
- **Shared** is for sharing objects across your organization. All users have full permissions for all objects in Shared.
- **Users** contains a folder for each user. We will call it as Users Home Folder. Objects in this folder are by default Private to that user.
- Workspace ID can be found in URL

Azure Databricks Architecture

Azure Databricks is structured to enable secure cross-functional team collaboration while keeping a significant amount of backend services managed by Azure Databricks so you can stay focused on your data science, data analytics and data engineering tasks.

Azure Databricks operates out of a control plane and data plane

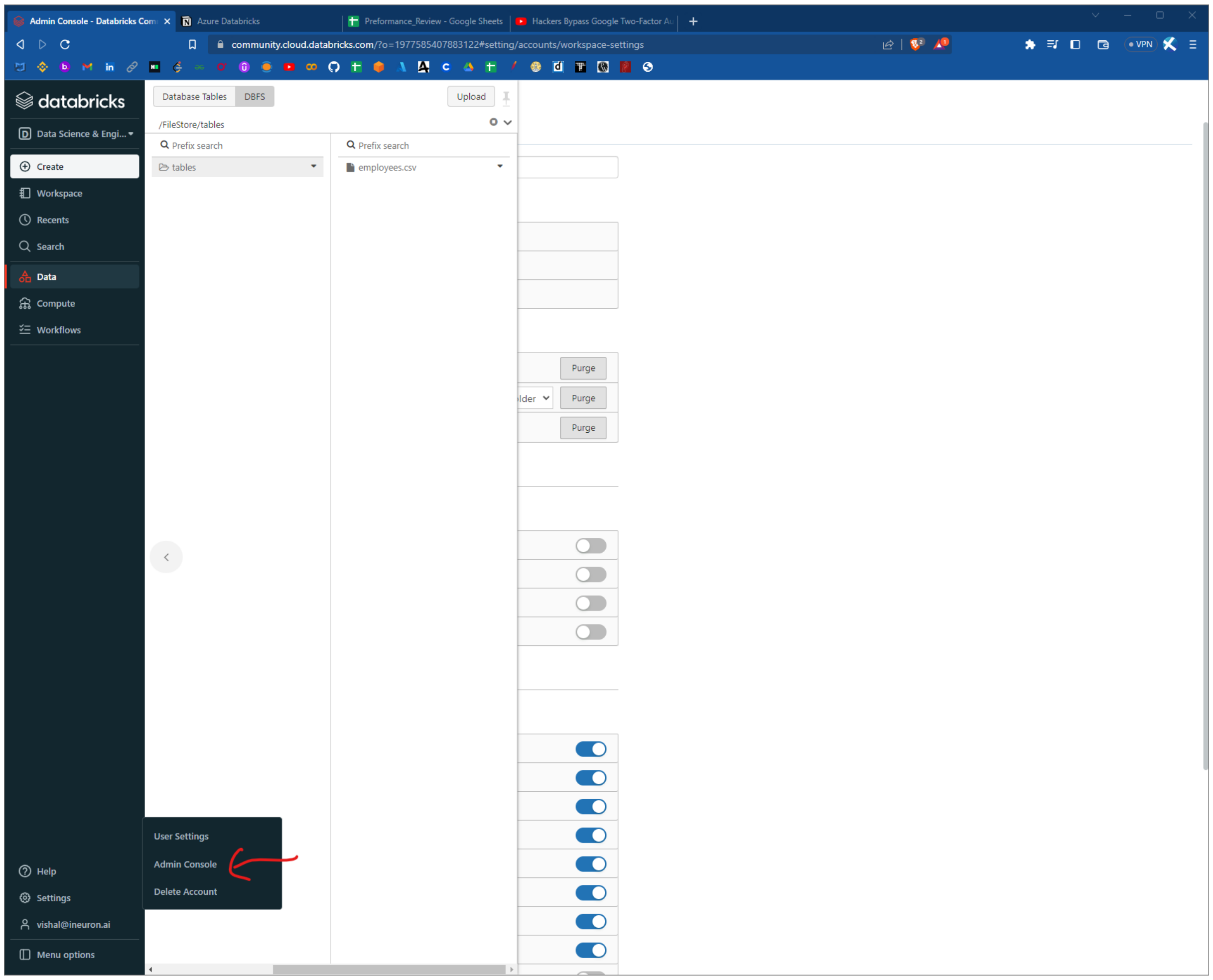
- The control plane includes the backend services that Azure Databricks manages in it's own Azure account. Notebook commands and many other workspace configurations are stored in the control plane and encrypted at rest.
- The data plane is managed by our Azure account and is where our data resides. This is also where data is processed.

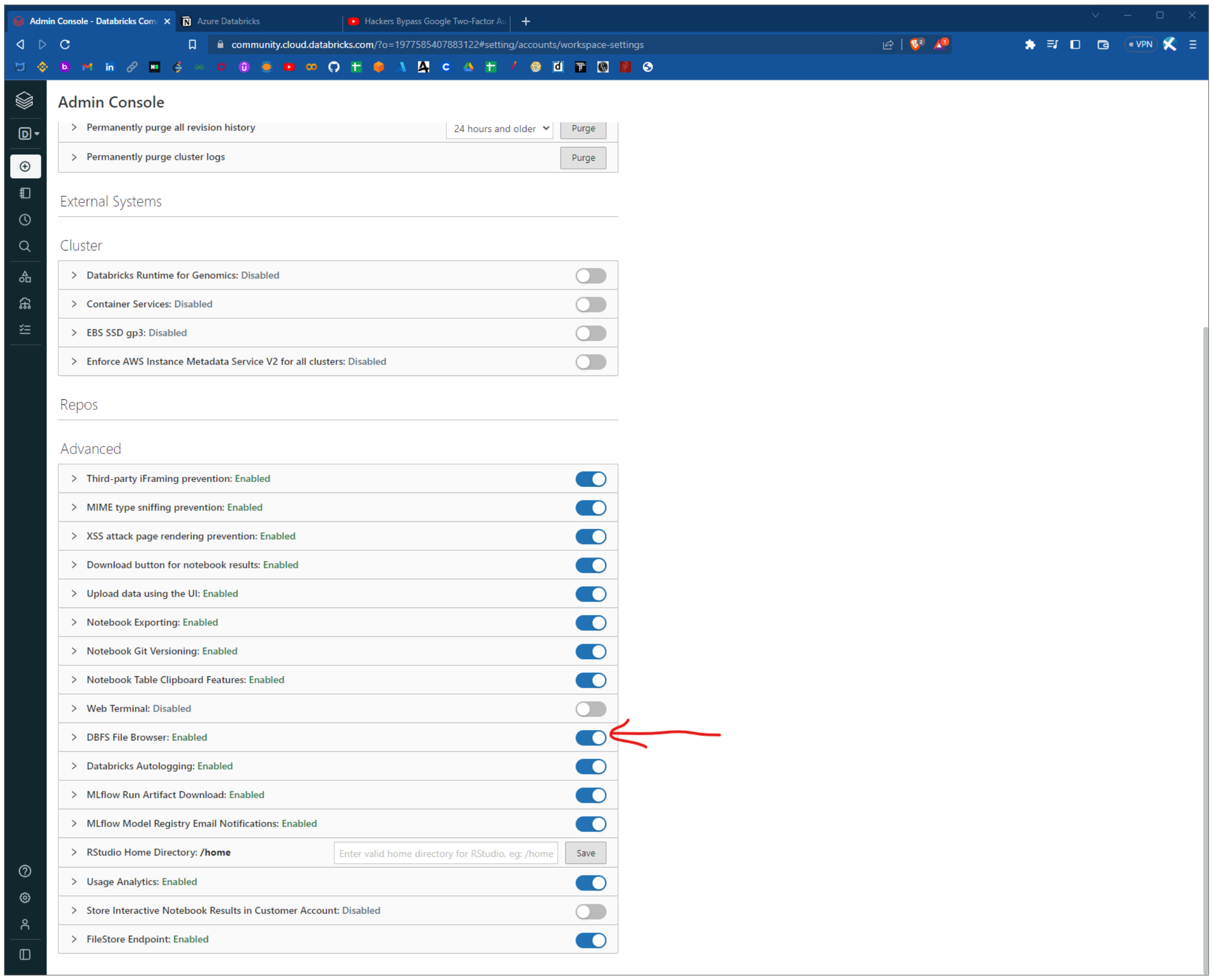

Databricks Cloud Account
Control Plane in Databricks Network
Manage customer accounts, data sets, and clusters
Databricks Web Application
Customer Notebooks
Jobs and Queries
Cluster Management
Databricks users
Single Sign-on (SSO)
Azure AD
SSO authentication does not send credentials to Databricks. Use SCIM to map SSO groups for Resource Access Control using ACLs
REST APIs
REST APIs Client Authenticate with revocable token
Launch cluster
Initiates jobs
Get job partial results
View job full results
Push/Pull table metadata
Push logging data
Customer Cloud Account
Data Plane in Customer Network
Data processing with Apache Spark cluster
Get/Put datasets and full results
Data Sets & Job Full Results
Databricks File System (DBFS) layer on top of customer- managed storage
Storage on Azure: Blob Storage
External Data Sources
On Public Internet (or in Customer Cloud Account)
- JDBC and SQL data sources
- Cloud data sources

Databricks File System

Databricks File System is a distributed file system mounted into an Azure Databricks workspace and available on Azure Databricks workspace and available on Azure Databricks clusters. DBFS is an abstract on top of scalable object storage.

The by default location of DBFS is known as DBFS root. DBFS will not be visible by default so we will have to go the admin console and enable the DBFS view.


Admin Console - Databricks Com
Azure Databricks
Preformance_Review - Google Sheets
Hackers Bypass Google Two-Factor Au
community.cloud.databricks.com/?o=1977585407883122#setting/accounts/workspace-settings
databricks
Data Science & Engi...
Create
Workspace
Recents
Search
Data
Compute
Workflows
Database Tables
DBFS
Upload
/FileStore/tables
Prefix search
tables
Prefix search
employees.csv
Purge
Purge
Purge
User Settings
Admin Console
Delete Account
Help
Settings
vishal@ineuron.ai
Menu options


Admin Console - Databricks Com
Azure Databricks
Hackers Bypass Google Two-Factor Au
community.cloud.databricks.com/?o=1977585407883122#setting/accounts/workspace-settings

Admin Console

- Permanently purge all revision history — 24 hours and older — Purge
- Permanently purge cluster logs — Purge

External Systems

Cluster

- Databricks Runtime for Genomics: Disabled
- Container Services: Disabled
- EBS SSD gp3: Disabled
- Enforce AWS Instance Metadata Service V2 for all clusters: Disabled

Repos

Advanced

- Third-party iFraming prevention: Enabled
- MIME type sniffing prevention: Enabled
- XSS attack page rendering prevention: Enabled
- Download button for notebook results: Enabled
- Upload data using the UI: Enabled
- Notebook Exporting: Enabled
- Notebook Git Versioning: Enabled
- Notebook Table Clipboard Features: Enabled
- Web Terminal: Disabled
- DBFS File Browser: Enabled
- Databricks Autologging: Enabled
- MLflow Run Artifact Download: Enabled
- MLflow Model Registry Email Notifications: Enabled
- RStudio Home Directory: /home — Enter valid home directory for RStudio, eg: /home — Save
- Usage Analytics: Enabled
- Store Interactive Notebook Results in Customer Account: Disabled
- FileStore Endpoint: Enabled

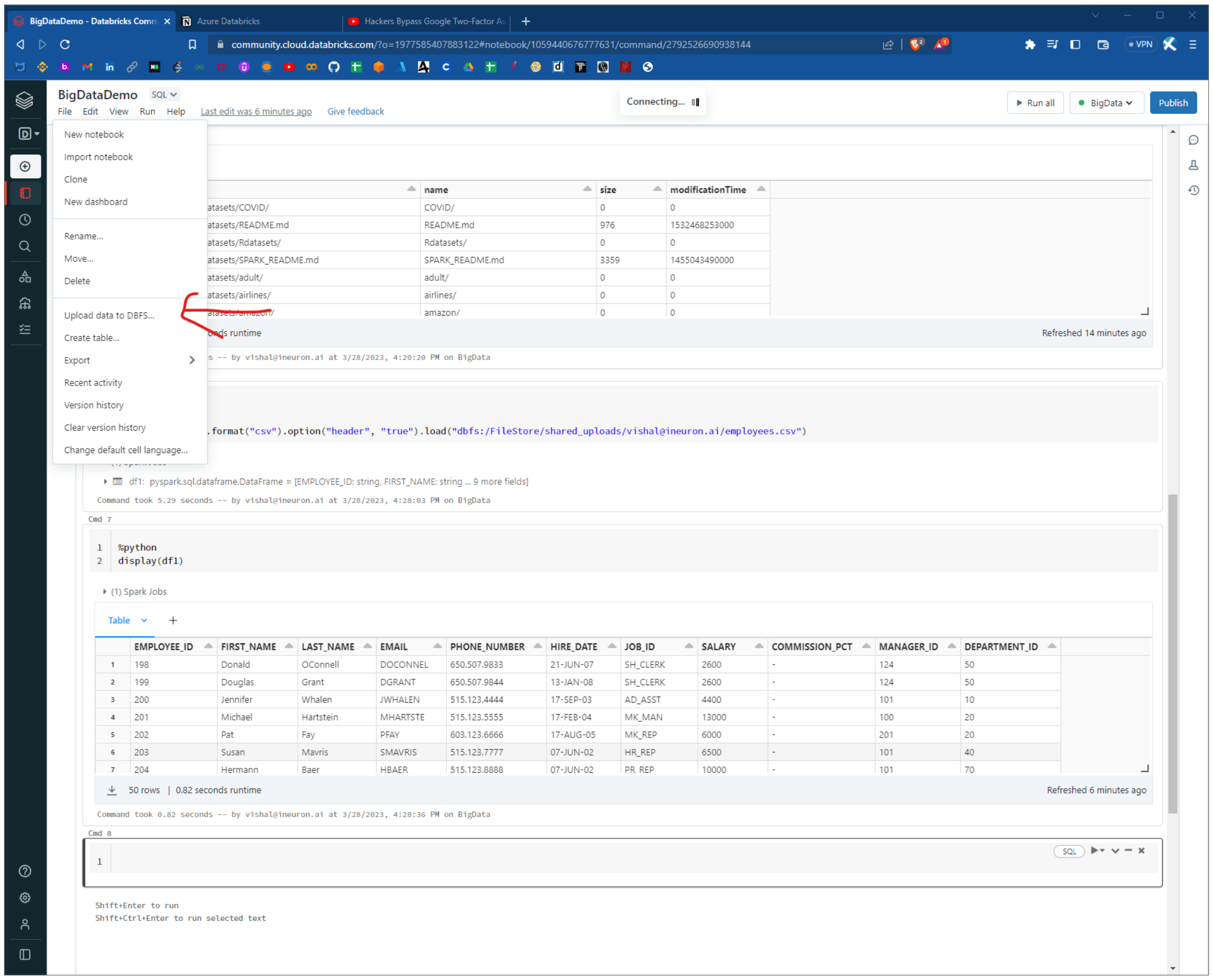
- /FileStore → import data files, generated plots and uploaded libraries
- /databricks-datasets → sample public datasets
- /databricks-results → files generated by downloading the results of a query

```
dbutils.fs.ls('dbfs:/')

display(dbutils.fs.ls('dbfs:/'))

display(dbutils.fs.ls('dbfs:/databricks-datasets/'))
```

Uploading dataset in DBFS



	name	size	modificationTime
atasets/COVID/	COVID/	0	0
atasets/README.md	README.md	976	1532468253000
atasets/Rdatasets/	Rdatasets/	0	0
atasets/SPARK_README.md	SPARK_README.md	3359	1455043490000
atasets/adult/	adult/	0	0
atasets/airlines/	airlines/	0	0
atasets/amazon/	amazon/	0	0

```
onds runtime                                                Refreshed 14 minutes ago
s -- by vishal@ineuron.ai at 3/28/2023, 4:20:20 PM on BigData
```

```
.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/vishal@ineuron.ai/employees.csv")
```

```
df1: pyspark.sql.dataframe.DataFrame = [EMPLOYEE_ID: string, FIRST_NAME: string ... 9 more fields]
Command took 5.29 seconds -- by vishal@ineuron.ai at 3/28/2023, 4:28:03 PM on BigData
Cmd 7
```

```
%python
display(df1)
```

```
(1) Spark Jobs
```

	EMPLOYEE_ID	FIRST_NAME	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	COMMISSION_PCT	MANAGER_ID	DEPARTMENT_ID
1	198	Donald	OConnell	DOCONNEL	650.507.9833	21-JUN-07	SH_CLERK	2600	-	124	50
2	199	Douglas	Grant	DGRANT	650.507.9844	13-JAN-08	SH_CLERK	2600	-	124	50
3	200	Jennifer	Whalen	JWHALEN	515.123.4444	17-SEP-03	AD_ASST	4400	-	101	10
4	201	Michael	Hartstein	MHARTSTE	515.123.5555	17-FEB-04	MK_MAN	13000	-	100	20
5	202	Pat	Fay	PFAY	603.123.6666	17-AUG-05	MK_REP	6000	-	201	20
6	203	Susan	Mavris	SMAVRIS	515.123.7777	07-JUN-02	HR_REP	6500	-	101	40
7	204	Hermann	Baer	HBAER	515.123.8888	07-JUN-02	PR_REP	10000	-	101	70

```
50 rows | 0.82 seconds runtime                              Refreshed 6 minutes ago
Command took 0.82 seconds -- by vishal@ineuron.ai at 3/28/2023, 4:28:36 PM on BigData
Cmd 8

Shift+Enter to run
Shift+Ctrl+Enter to run selected text
```

```
df1 = spark.read.format("csv").option("header", "true").load("dbfs:/FileStore/shared_uploads/vishal@ineuron.ai/employees.csv")

display(df1)
```

Databricks dbutils

Databricks Utilities (dbutils) make it easy to perform powerful combinations of tasks

- To parameterize notebooks
- To chain notebooks
- To work with secrets

dbutils utilities are available in only Python and Scala notebooks

```
dbutils.help()

dbutils.fs.help()

dbutils.fs.help("ls")
```

Databricks dbutils.data

The data utilities allows you to understand and interpret datasets. It’s available in Databricks runtime 9.0 and above. It can work with both pandas and spark dataframes.

```
dbutils.data.help()

data = [(1, 'Vishal'), (2, 'Shashank'), (3, 'Krish')]
cols = ['id', 'name']

df1 = spark.createDataFrame(data, cols)

dbutils.data.summarize(df1)
```

Databricks dbutils.fs

The file system utility allows us to access DBFS making it easier to use Azure Databricks as a file system. To list the commands run the below code

```
dbutils.fs.help()

# copy file from one to another
dbutils.fs.cp('FileStore/tables/employees.csv', 'FileStore/temp/employees.csv')

# display first 65536 bytes
df1 = dbutils.fs.head('FileStore/tables/employees.csv')
display(df1)

# display first 25 bytes
df1 = dbutils.fs.head('FileStore/tables/employees.csv',  25)
display(df1)

# make a self defined directory
dbutils.fs.mkdirs('/FileStore/data/')

# insert data into some file
dbutils.fs.put("/FileStore/data/my_file.txt", "Hello World", True)

# view file
dbutils.fs.head("/FileStore/data/my_file.txt")

# view any directory
dbutils.fs.ls('/FileStore/data')

# move file to other directory
dbutils.fs.mv("/FileStore/data/my_file.txt", "/FileStore/temp/my_file.txt")

# remove any file from location
dbutils.fs.rm("/FileStore/temp/my_file.txt")
```

Databricks dbutils.notebook

```
dbutils.notebook.help()

# the notebook utility allows you to chain together notebooks and act on their results
firstName = "Vishal"

dbutils.notebook.exit(firstName)
```

Let’s create a new notebook and run the below commands

```
dbutils.notebook.help('run')

dbutils.notebook.run('BigDataDemo', 60)
```

Databricks dbutils.widgets

The widgets utilities allows us to parameterize notebooks

```
dbutils.widgets.help()

dbutils.widgets.combobox(name="iNeuronCoursesCB", defaultValue="Big Data", choices=["Big Data", "ML", "Java dev", "Web dev"], label="iN

dbutils.widgets.dropdown(name="iNeuronCoursesDD", defaultValue="Big Data", choices=["Big Data", "ML", "Java dev", "Web dev"], label="iN

dbutils.widgets.multiselect(name="iNeuronCoursesMS", defaultValue="Big Data", choices=["Big Data", "ML", "Java dev", "Web dev"], label=

dbutils.widgets.text(name="iNeuronCoursesT", defaultValue="Big Data", label="iNeuron Courses")

dbutils.widgets.get('iNeuronCoursesMS')
```

if we pass any name as an argument that’s not present it will throw us an error

```
dbutils.widgets.get('iNeuronCoursesMS1')

dbutils.widgets.getArgument('iNeuronCoursesMS', 'Error: This widget is not available')

# remove specfic widget
dbutils.widgets.remove('iNeuronCoursesMS')

# remove all widget at once
dbutils.widgets.removeAll()
```

We can also create widgets using SQL also

```
CREATE WIDGET TEXT firstname DEFAULT 'Vishal';

CREATE WIDGET DROPDOWN gender DEFAULT 'Male' CHOICES SELECT 'Male';
```

We can write a subquery to create a dropdown

```
data = [(1, "Vishal", "Male"), (2, "Priya", "Female"), (3, "Shashank", "Male"), (4, "Sita", "Female")]
cols = ['id', 'Name', 'Gender']

df = spark.createDataFrame(data, cols)
display(df)
```

Create a temp table

```
df.createOrReplaceTempView('persons')
```

Now let’s create widget

```
CREATE WIDGET DROPDOWN genderDD DEFAULT 'Male' CHOICES SELECT gender from persons;

REMOVE WIDGET genderDD;

CREATE WIDGET DROPDOWN genderDD DEFAULT 'Male' CHOICES SELECT DISTINCT gender from persons;
```

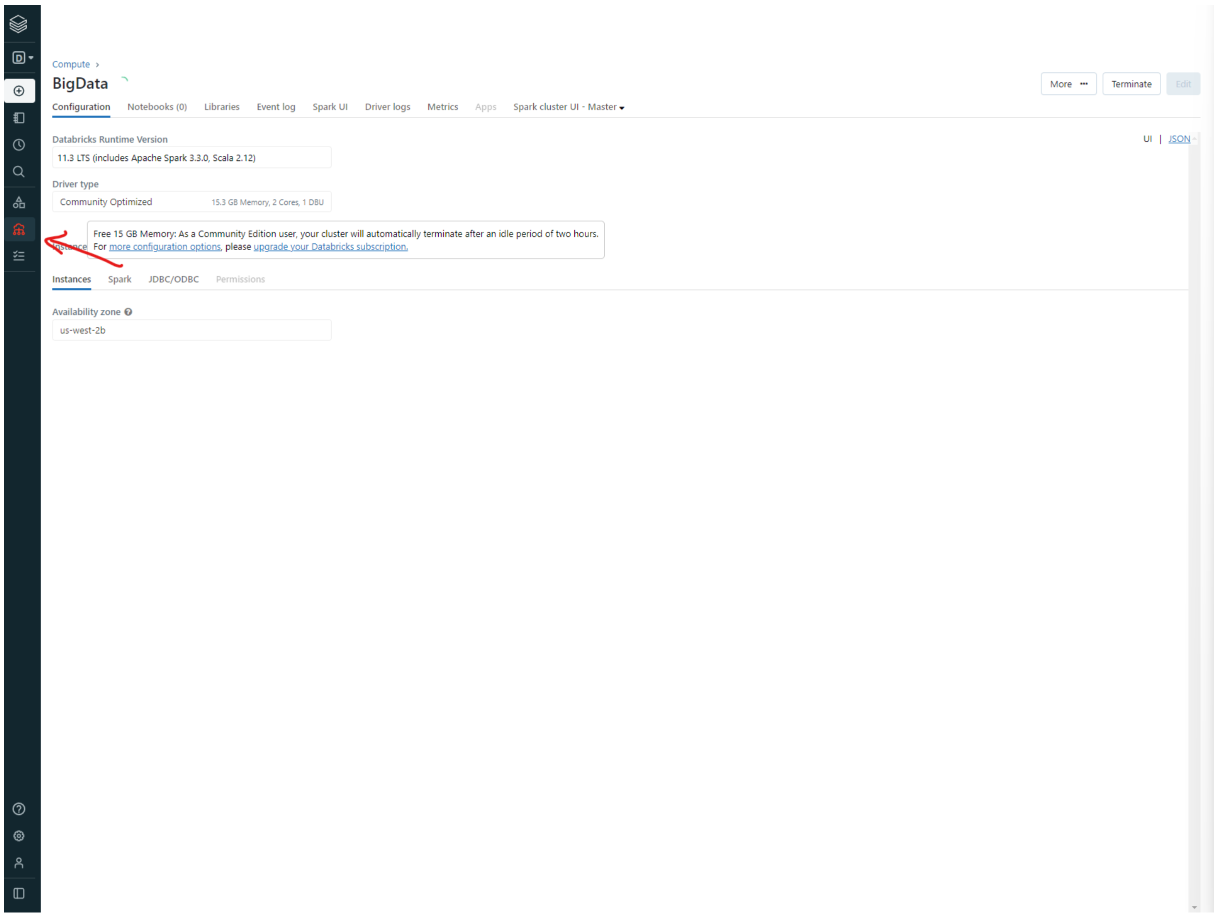
Use of widgets

```
select * from persons where gender = '$genderDD'

select * from persons where gender = getArgument('genderDD');
```

Create and run Spark Jobs

Lets create the cluster first because the cluster is the computing engine for the notebook


Compute ›

BigData

More · Terminate · Edit

Configuration · Notebooks (0) · Libraries · Event log · Spark UI · Driver logs · Metrics · Apps · Spark cluster UI - Master

UI | JSON

Databricks Runtime Version

11.3 LTS (includes Apache Spark 3.3.0, Scala 2.12)

Driver type

Community Optimized 15.3 GB Memory, 2 Cores, 1 DBU

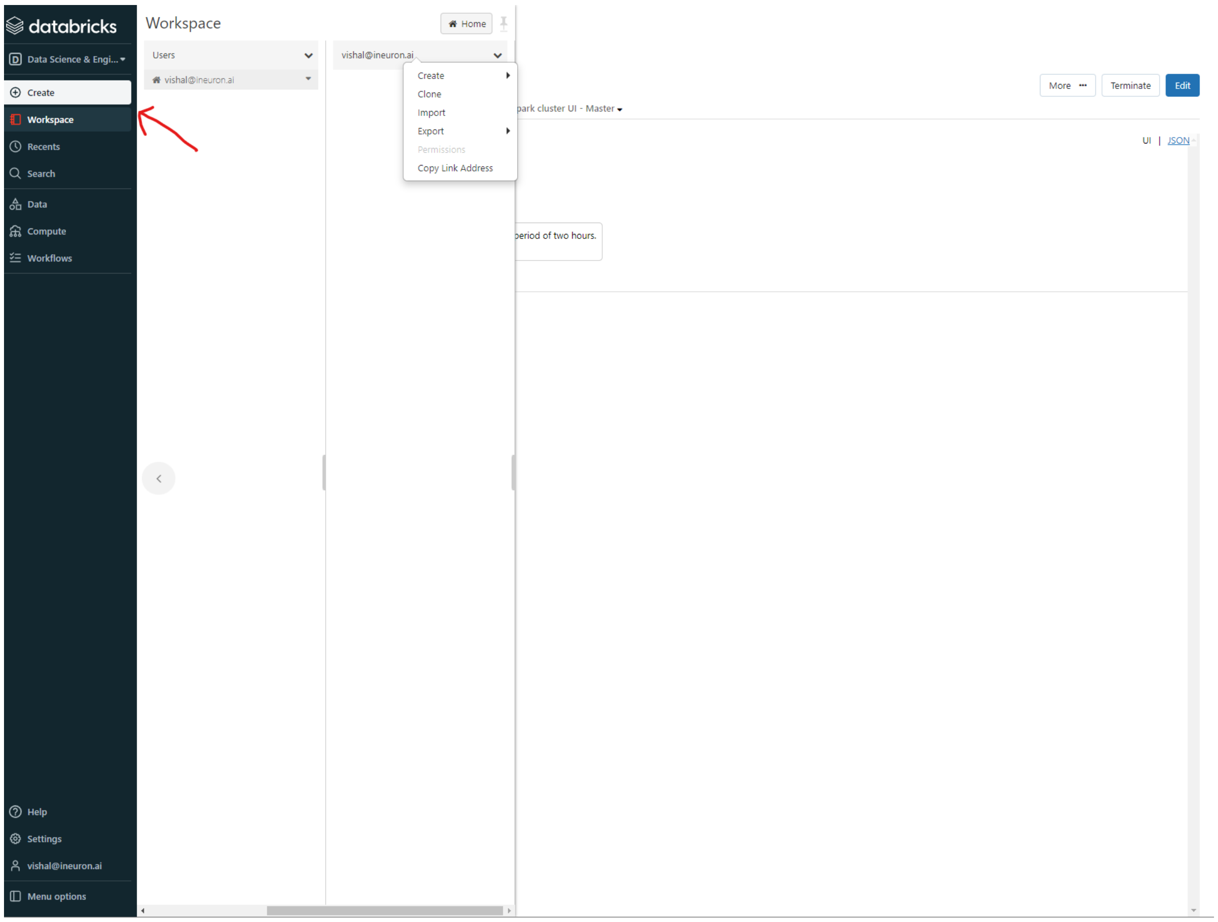
Free 15 GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours. For more configuration options, please upgrade your Databricks subscription.

Instances · Spark · JDBC/ODBC · Permissions

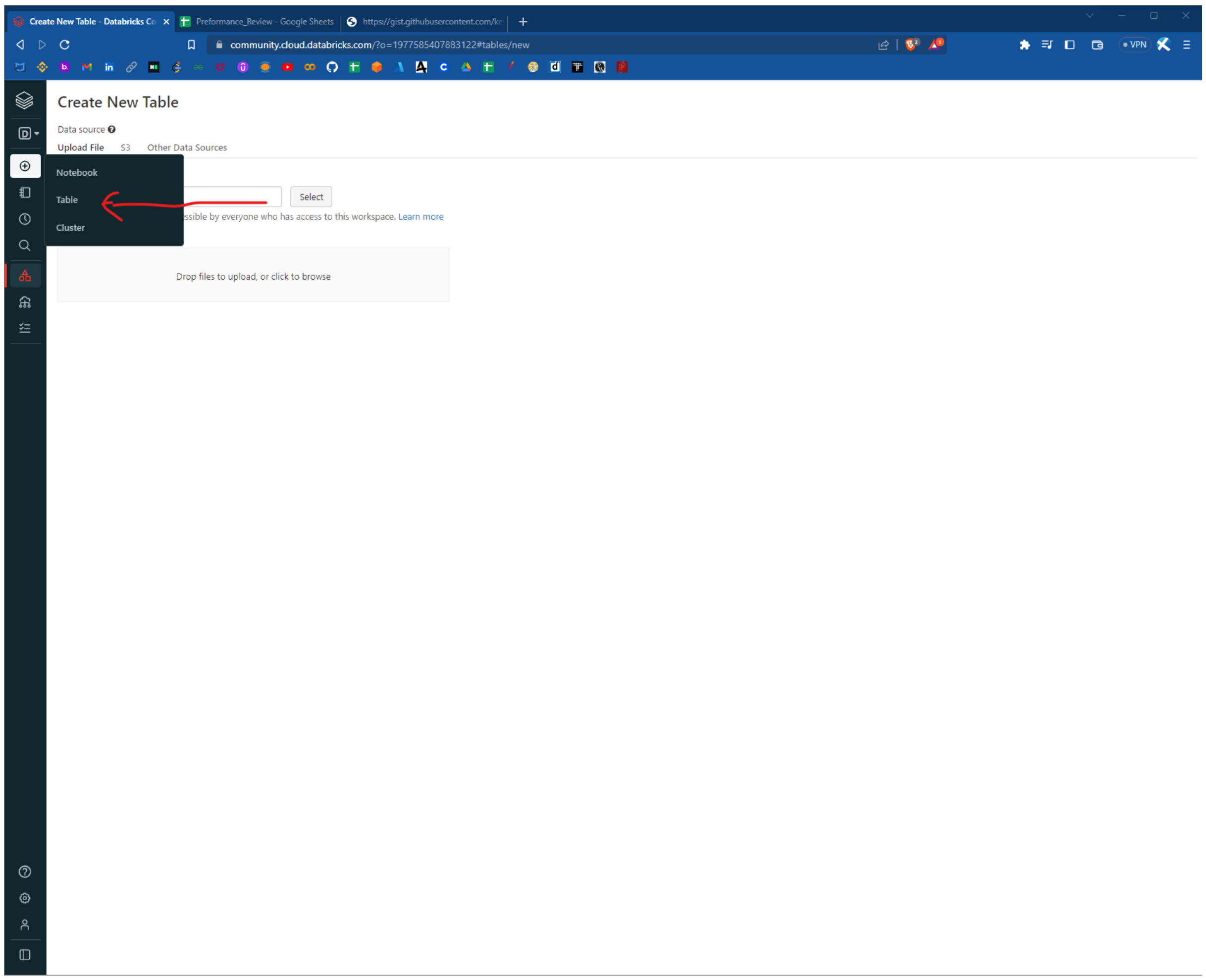
Availability zone

us-west-2b

Now let’s create the notebook



Let's upload the data to the Data Bricks File System


Create New Table
Data source
Upload File
S3
Other Data Sources
Notebook
Table
Cluster
Select
ssible by everyone who has access to this workspace. Learn more
Drop files to upload, or click to browse

Lets go back on the notebook and write few commands

```
DROP TABLE IF EXISTS employees;

CREATE TABLE employees USING CSV OPTIONS (path "/FileStore/tables/employees.csv", header "true");

SELECT * FROM employees;

SELECT DEPARTMENT_ID, avg(SALARY) AS AvgSalary FROM employees GROUP BY DEPARTMENT_ID;
```

Then there are different visualization that we can perform on top of our dataset