

Introduction to Spark

In this lab exercise you will get familiar with PySpark, the interactive mode of Apache Spark. You will learn the following topics:

- Exercise 1: Use the interactive Spark environment from Databricks
- Exercise 2: Create a DataFrame based on a Python data set
- Exercise 3: Create a DataFrame from a text file and work with the file system mode
- Exercise 4: Analyze more examples and visualize the results

Exercise 1: Use the interactive Spark environment from Databricks

Get a Spark account from Databricks at: <https://community.cloud.databricks.com/>

Do not be alarmed, initially it will look like you get access to a trial account, but you will be able to choose the Community Edition in the next step:

Try Databricks

AN OPEN AND UNIFIED DATA ANALYTICS PLATFORM FOR DATA ENGINEERING, MACHINE LEARNING, AND ANALYTICS

From the original creators of Apache Spark™, Delta Lake, MLflow, and Koalas

Select a platform

DATABRICKS PLATFORM – FREE TRIAL

For businesses

- Collaborative environment for Data teams to build solutions together
- Unlimited clusters that can scale to any size, processing data in your own account
- Job scheduler to execute jobs for production pipelines
- Fully collaborative notebooks with multi-language support, dashboards, REST APIs
- Native integration with the most popular ML frameworks (scikit-learn, TensorFlow, Keras,...), Apache Spark™, Delta Lake, and MLflow
- Advanced security, role-based access controls, and audit logs
- Single Sign On support
- Integration with BI tools such as Tableau, Qlik, and Looker
- 14-day full feature trial (excludes cloud charges)

COMMUNITY EDITION


For students and educational institutions

- Single cluster limited to 6GB and no worker nodes
- Basic notebooks without collaboration
- Limited to 3 max users
- Public environment to share your work


GET STARTED

By clicking "Get Started" for the Community Edition, you agree to the [Databricks Community Edition Terms of Service](#)


CHOOSE YOUR CLOUD



Please note that Azure Databricks is provided by Microsoft and is subject to Microsoft's terms.

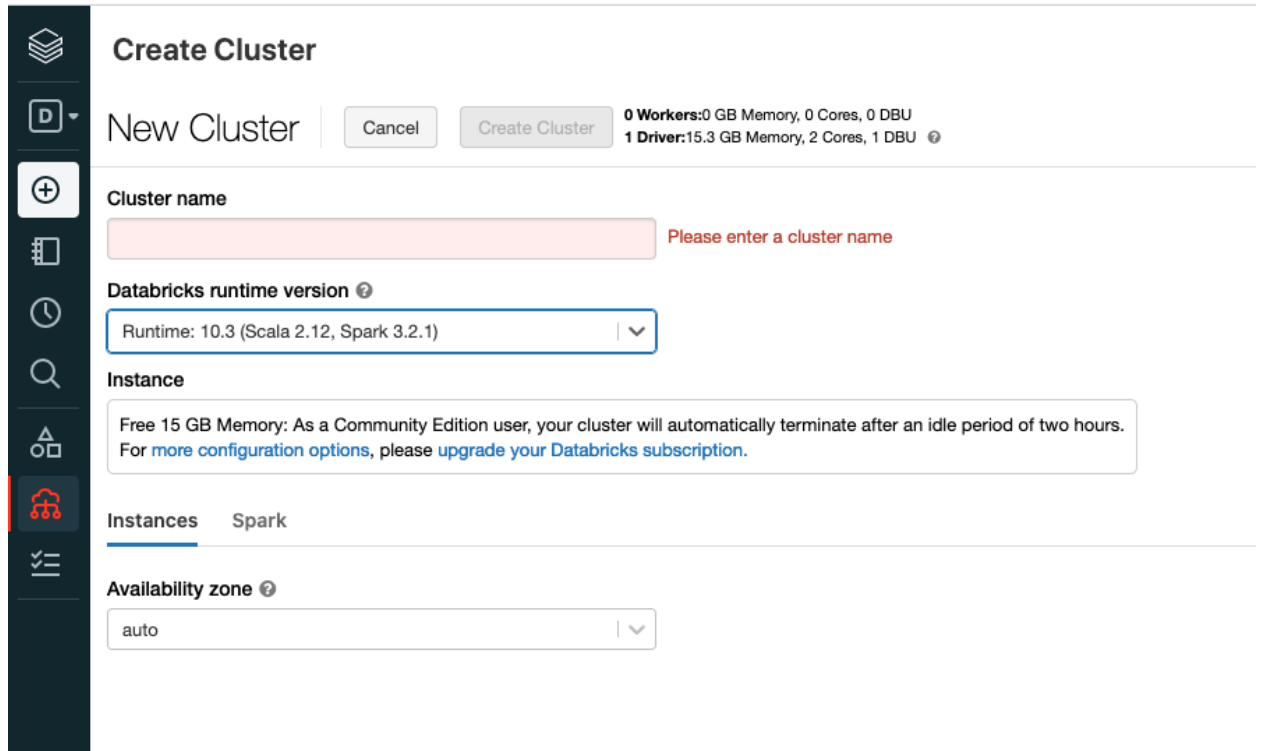


By clicking on the "AWS" button to get started, you agree to the [Databricks Terms of Service](#).



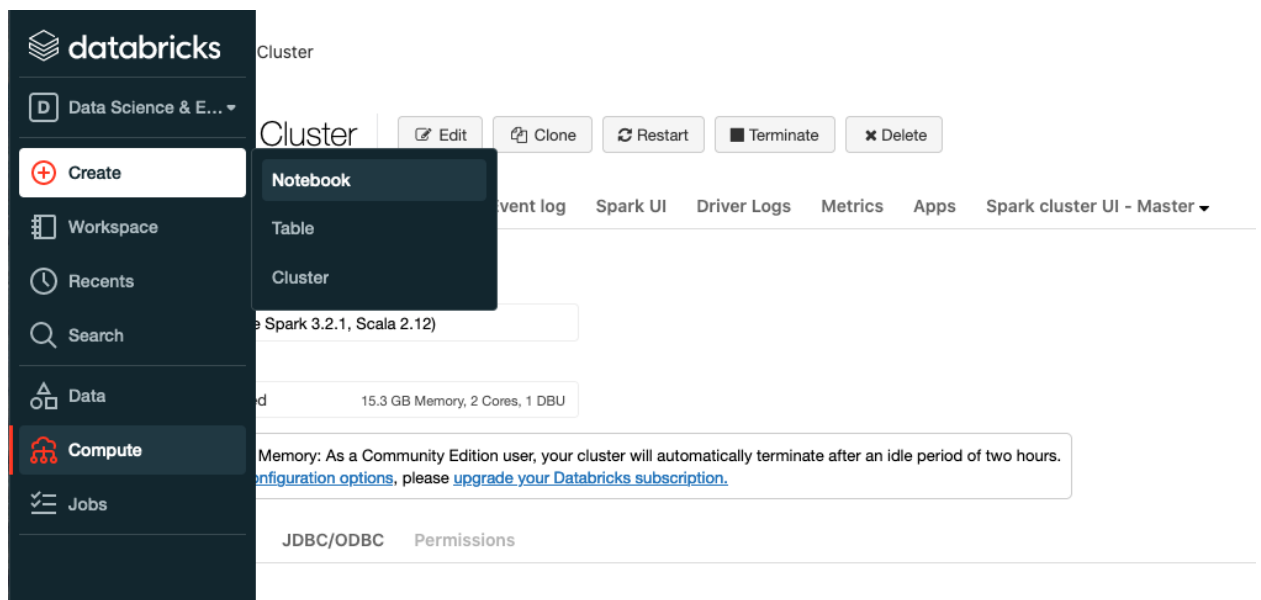
By clicking on the "Google Cloud" button to get started, you agree to the [Databricks Terms of Service](#).

Create a new cluster with Spark 3.x. Click on “**Compute**”. Choose the latest Spark version.



The screenshot shows the 'Create Cluster' interface in Databricks. On the left is a dark sidebar with navigation icons. The main area has a header 'Create Cluster' and a sub-header 'New Cluster'. Below this are two buttons: 'Cancel' and 'Create Cluster'. To the right of these buttons, the current cluster configuration is shown: '0 Workers: 0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 15.3 GB Memory, 2 Cores, 1 DBU'. The 'Cluster name' field is empty with a red error message 'Please enter a cluster name'. The 'Databricks runtime version' dropdown is set to 'Runtime: 10.3 (Scala 2.12, Spark 3.2.1)'. The 'Instance' section contains a warning: '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.' Below this, the 'Instances' tab is selected, showing 'Spark' as the instance profile. The 'Availability zone' dropdown is set to 'auto'.

Create a new Python Notebook.



The screenshot shows the Databricks 'Compute' menu open, with options for 'Notebook', 'Table', and 'Cluster'. The 'Notebook' option is highlighted. In the background, the 'Cluster' page is visible, showing the same configuration as the previous screenshot. The 'Cluster' page has a header with 'Cluster' and buttons for 'Edit', 'Clone', 'Restart', 'Terminate', and 'Delete'. Below the buttons are tabs for 'Event log', 'Spark UI', 'Driver Logs', 'Metrics', 'Apps', and 'Spark cluster UI - Master'. The 'Cluster' page also shows the same configuration details as the 'Create Cluster' page.

Create Notebook

Name

Exercise 1

Default Language

Python

Cluster

Big Data Cluster

Cancel

Create

Ready to go.

Exercise 2: Create a DataFrame based on a Python data set

- Create a Python dataset (list) that contains the following data:

Zurich, 400000, Limmat
Vienna, 1900000, Danube
Paris, 2200000, Seine
Rome, 2900000, Tiber
London, 8700000, Thames

- Turn the list into a DataFrame
- Show two different ways to select the city on the river Danube
 - Via DataFrame API
 - Via SQL (temp table)

Exercise 3: Create a DataFrame from a text file

- Save the cities in Exercise 2 in a text file on your laptop.
- Create a DataFrame by reading the text file

Hint: To upload a file from your laptop to Databricks, follow these steps:

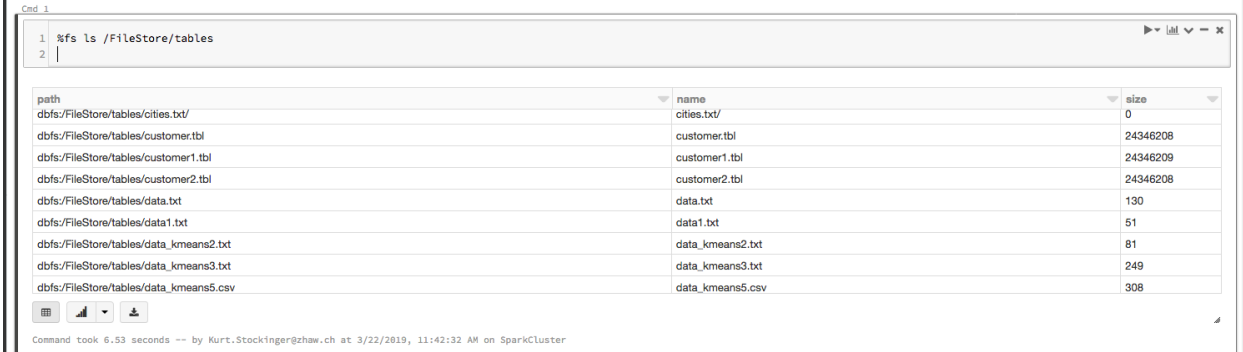
- Data / Create Table / Upload File / Drop files to upload, or browse.
- Save the location of the file in a text-file in order to use it later on (see bottom of the figure)

The screenshot shows the Databricks 'Create New Table' interface. On the left is a dark sidebar with navigation icons. The main panel has the title 'Create New Table' and a 'Data source' section with tabs for 'Upload File', 'S3', 'DBFS', 'Other Data Sources', and 'Partner Integrations'. The 'Upload File' tab is active. Below it, the 'DBFS Target Directory' is set to '/FileStore/tables/' with an optional text input and a 'Select' button. A message states: 'Files uploaded to DBFS are accessible by everyone who has access to this workspace. [Learn more](#)'. Under the 'Files' section, a file named 'data.txt' (0.1 KB) is shown with a green checkmark and a 'Remove file' link. At the bottom, a green checkmark confirms: 'File uploaded to /FileStore/tables/data.txt'.

You can look at the files stored by Databricks using the following command in the Python notebook. Note that you need to add “%fs” in front of the command to invoke the **file system mode** as opposed to the Python-mode:

```
%fs ls /FileStore/tables/
```

A possible result might be:



Cmd 1

```
1 %fs ls /FileStore/tables/
2
```

path	name	size
dbfs:/FileStore/tables/cities.txt/	cities.txt/	0
dbfs:/FileStore/tables/customer.tbl	customer.tbl	24346208
dbfs:/FileStore/tables/customer1.tbl	customer1.tbl	24346209
dbfs:/FileStore/tables/customer2.tbl	customer2.tbl	24346208
dbfs:/FileStore/tables/data.txt	data.txt	130
dbfs:/FileStore/tables/data1.txt	data1.txt	51
dbfs:/FileStore/tables/data_kmeans2.txt	data_kmeans2.txt	81
dbfs:/FileStore/tables/data_kmeans3.txt	data_kmeans3.txt	249
dbfs:/FileStore/tables/data_kmeans5.csv	data_kmeans5.csv	308

Command took 6.53 seconds -- by Kurt.Stockinger@zhaw.ch at 3/22/2019, 11:42:32 AM on SparkCluster

You can inspect the file content with the following command in the file system mode:

```
%fs head /FileStore/tables/data.txt
```



Cmd 3

```
1 %fs head /FileStore/tables/data.txt
```

```
city,population,river
Zurich,400000,Limmat
Vienna,1900000,Danube
Paris,2200000,Seine
Rome,2900000,Tiber
London,8700000,Thames
```

Command took 0.34 seconds -- by Kurt.Stockinger@zhaw.ch at 3/22/2019, 11:46:45 AM on SparkCluster

You can also delete a file:

%fs rm /FileStore/tables/data.txt

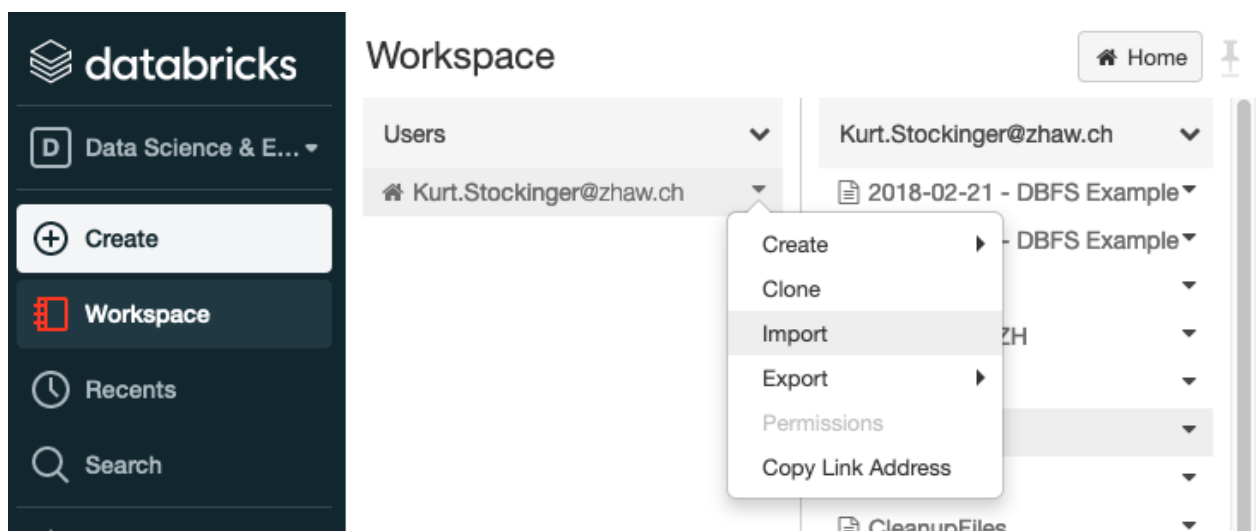
```
1  %fs  /FileStore/tables/data.txt
```

```
res2: Boolean = true
```

```
Command took 1.21 seconds -- by Kurt.Stockinger@zhaw.ch at 3/1/2022, 9:37:31 AM on Big Data Cluster
```

Exercise 4: Execute and analyze the examples in **DataFrameExample.ipynb**

- Note: Import the Notebook into Databricks as follows:
 1. Workspace/Users
 2. Select your user
 3. Select the arrow on third column (see below) and chose **Import**



4. Drag file **DataFrameExample.ipynb** into respective area
5. Click **Import**

Analyze the code to explore different ways of working with Spark DataFrames and to visualize results, e.g. using Pandas and Matplotlib as shown below.


```

1 # Make some nice plots using Pandas and Matplotlib
2
3 import pandas as pd
4 import matplotlib.pyplot as plt
5 plt.clf()
6 pdDF = nonNullDF.toPandas()
7 pdDF.plot(x='firstName', y='salary', kind='bar', rot=45)
8 display()
9

```

► (1) Spark Jobs

