

# Big Data & Analytics

Databricks: A Tutorial

---

**Dr. Ignacio Castineiras**  
Department of Computer Science

# Databricks: An Online Platform for Data Engineers

- Databricks is a spin-off company leveraging Spark as a **cloud-based** big data processing tool <https://databricks.com/>



databricks<sup>®</sup>

# Databricks: An Online Platform for Data Engineers

- The Databricks Community Edition is free to use.  
It will be the approach we will follow.



# Databricks: An Online Platform for Data Engineers

- The Databricks Community Edition is free to use.  
It will be the approach we will follow.
- The Community Edition provides us with a local mode-based  
Spark configuration (similar to using Spark in our own local machine).



# Databricks: An Online Platform for Data Engineers

- The Databricks Community Edition is free to use.  
It will be the approach we will follow.
- The Community Edition provides us with a local mode-based Spark configuration (similar to using Spark in our own local machine).
- Specifically, it provides us with **A Cluster of 1 Databricks Unit (DBU)**:  
*“A unit of processing capability per hour”*
  - 1 physical machine with
  - 2 cores and
  - 8GB of memory storage.



# Databricks: An Online Platform for Data Engineers

- On the other hand, the Databricks Enterprise Edition allows us to rent a cluster in a price-per-usage basis, and configure it to target the Business Unit specific needs.



# Databricks: An Online Platform for Data Engineers

- On the other hand, the Databricks Enterprise Edition allows us to rent a cluster in a price-per-usage basis, and configure it to target the Business Unit specific needs.
- The cluster is indeed hosted by Microsoft Azure or Amazon Web Services, with Databricks providing the interface to automate the cluster setup.



Microsoft  
Azure



# Databricks: An Online Platform for Data Engineers

- On the other hand, the Databricks Enterprise Edition allows us to rent a cluster in a price-per-usage basis, and configure it to target the Business Unit specific needs.
- The cluster is indeed hosted by Microsoft Azure or Amazon Web Services, with Databricks providing the interface to automate the cluster setup.



- The range of prices depends on the functionality provided by Databricks, and can be consulted at: <https://databricks.com/product/pricing#dbu>



# Databricks: An Online Platform for Data Engineers

*How to...*  
Sign up / sign in.

# Databricks: An Online Platform for Data Engineers

1. We register/sign up to Databricks Community Edition at:  
<https://databricks.com/signup/signup-community>

 databricks Platform Solutions Customers Learn Partners Events Open Source Company

## Sign Up for Databricks Community Edition

First Name \*

Last Name \*

Company Name \*

Work Email \*

Phone Number

What is your intended use case? \*

How would you describe your role? \*

Keep me informed with the occasional update about Databricks and Apache Spark™.

By clicking "Sign Up", you agree to the [Terms of Service](#) and the [Privacy Policy](#).

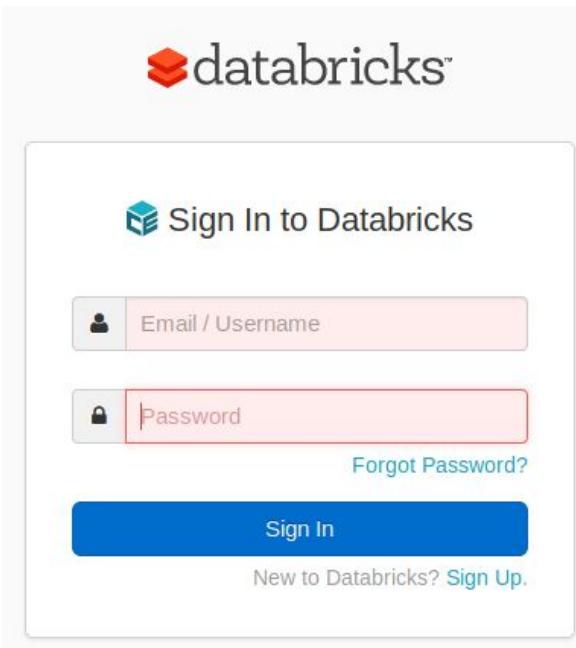
I'm not a robot   
reCAPTCHA  
Privacy • Terms

[Sign Up](#)

# Databricks: An Online Platform for Data Engineers

- Once registered, we sign in at:

<https://community.cloud.databricks.com>



# Databricks: An Online Platform for Data Engineers

*How to...*  
Create a new cluster.

# Databricks: An Online Platform for Data Engineers

3. We are redirected to the main page of the web interface. Our first step is to set up a new Community Edition cluster by clicking in **Clusters**.



Welcome to databricks™

  
[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](#)
-  [Create Table](#)
-  [New Cluster](#)

**Recents**

-  [34\\_job\\_inspection\\_5.scala](#)
-  [33\\_job\\_inspection\\_4.scala](#)
-  [32\\_job\\_inspection\\_3.scala](#)

**What's new in v3.2**

- Instance Pools
- Databricks Runtime 6.0 will drop Python 2 support

[View latest release notes](#)

# Databricks: An Online Platform for Data Engineers

3. We set up the new cluster with the following parameters:

The screenshot shows the 'Create Cluster' interface in Databricks. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title 'Create Cluster' and a sub-section 'New Cluster'. It includes a 'Cancel' button and a 'Create Cluster' button (which is highlighted in blue). Below these are sections for 'Cluster Name' (containing 'MyCluster'), 'Databricks Runtime Version' (set to 'Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)'), 'Python Version' (set to '3'), and 'Instance' (with a note about free memory and upgrade options). At the bottom, there are tabs for 'Instances' (selected) and 'Spark', and a dropdown for 'Availability Zone' (set to 'us-west-2c').

# Databricks: An Online Platform for Data Engineers

3. We set up the new cluster with the following parameters:

The screenshot shows the 'Create Cluster' interface in Databricks. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title 'Create Cluster' and a sub-section 'New Cluster'. It includes a 'Cancel' button and a 'Create Cluster' button. To the right of the buttons are resource specifications: '0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 6.0 GB Memory, 0.88 Cores, 1 DBU'. Below these are several configuration fields:

- Cluster Name:** A text input field containing 'MyCluster', which is highlighted with a red rectangle.
- Databricks Runtime Version:** A dropdown menu showing 'Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)'.
- Python Version:** A dropdown menu showing '3'.
- Instance:** A note stating 'Free 6GB Memory: As a Community Edition user, your cluster will automatically terminate after an idle period of two hours.' It also links to 'more configuration options' and 'upgrade your Databricks subscription'.
- Instances:** A tab labeled 'Spark' is selected, indicated by a blue underline.
- Availability Zone:** A dropdown menu showing 'us-west-2c'.

# Databricks: An Online Platform for Data Engineers

3. We set up the new cluster with the following parameters:

The screenshot shows the 'Create Cluster' interface in Databricks. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title 'Create Cluster' and a sub-section 'New Cluster'. It includes a 'Cancel' button and a 'Create Cluster' button. To the right of the buttons are resource details: '0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 6.0 GB Memory, 0.88 Cores, 1 DBU'. Below these are input fields for 'Cluster Name' (containing 'MyCluster'), 'Databricks Runtime Version' (set to 'Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)', highlighted with a red box), 'Python Version' (set to '3'), and an 'Instance' section which contains a note about free memory and links to configuration options. At the bottom are tabs for 'Instances' (selected) and 'Spark', and a dropdown for 'Availability Zone' (set to 'us-west-2c').

# Databricks: An Online Platform for Data Engineers

3. We set up the new cluster with the following parameters:

The screenshot shows the 'Create Cluster' interface in Databricks. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area has a title 'New Cluster' with a 'Cancel' button and a 'Create Cluster' button. To the right of the button are cluster specifications: '0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 6.0 GB Memory, 0.88 Cores, 1 DBU'. Below these are fields for 'Cluster Name' (MyCluster), 'Databricks Runtime Version' (Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)), and 'Python Version' (set to 3). A red box highlights the 'Python Version' dropdown. At the bottom, there's an 'Instance' section with a note about free memory and a link to upgrade the subscription, and tabs for 'Instances' (selected) and 'Spark'.

# Databricks: An Online Platform for Data Engineers

3. We set up the new cluster with the following parameters:

The screenshot shows the 'Create Cluster' interface in the Databricks web application. On the left, a sidebar menu lists 'databricks', 'Home', 'Workspace', 'Recents', 'Data', 'Clusters', 'Jobs', and 'Search'. The main area is titled 'Create Cluster' and 'New Cluster'. It includes fields for 'Cluster Name' (set to 'MyCluster'), 'Databricks Runtime Version' (set to 'Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)'), 'Python Version' (set to '3'), and an 'Instance' section with a note about free memory and upgrade options. At the bottom, tabs for 'Instances' (selected) and 'Spark' are visible, along with an 'Availability Zone' dropdown set to 'us-west-2c'. A red box highlights the 'Create Cluster' button and its tooltip information: '0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU' and '1 Driver: 6.0 GB Memory, 0.88 Cores, 1 DBU'.

Create Cluster

New Cluster | Cancel | Create Cluster

0 Workers: 0.0 GB Memory, 0 Cores, 0 DBU  
1 Driver: 6.0 GB Memory, 0.88 Cores, 1 DBU

Cluster Name

MyCluster

Databricks Runtime Version

Runtime: 5.5 LTS (Scala 2.11, Spark 2.4.3)

Python Version

3

Instance

Free 6GB 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](#).

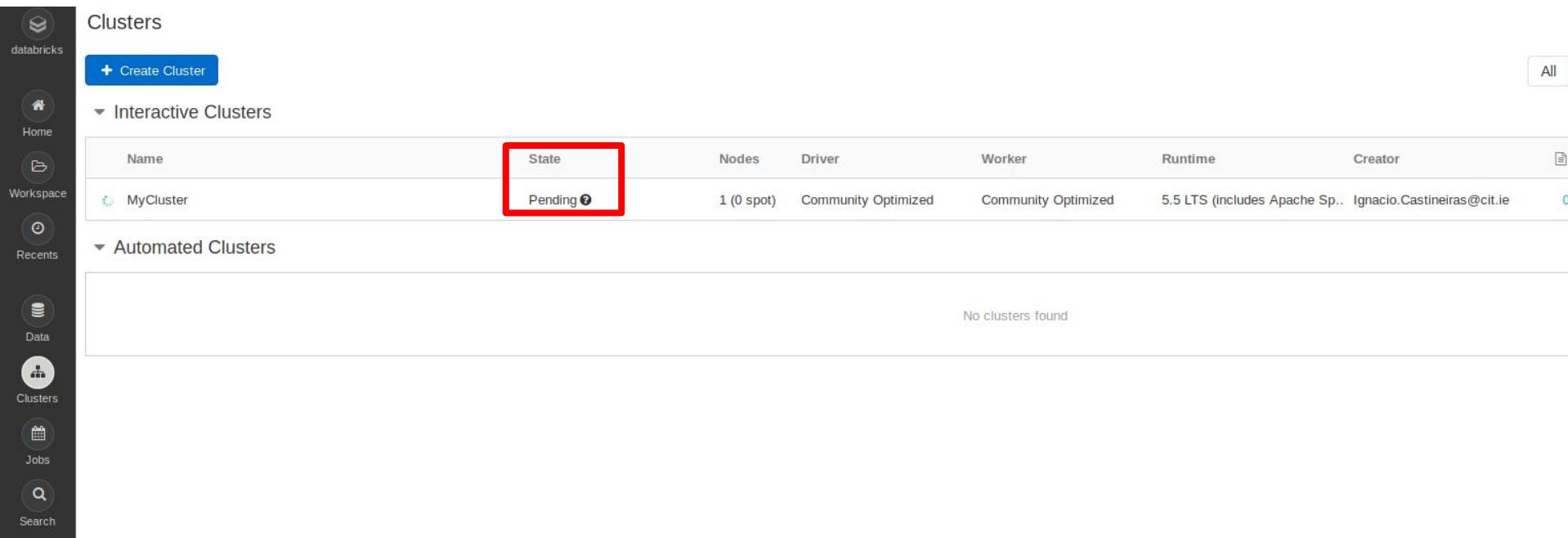
Instances | Spark

Availability Zone

us-west-2c

# Databricks: An Online Platform for Data Engineers

3. The cluster will take a couple of minutes to be set up:



The screenshot shows the Databricks interface for managing clusters. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data, Clusters (which is selected and highlighted in blue), Jobs, and Search. The main area is titled 'Clusters' and shows a table of existing clusters. A blue button at the top left says '+ Create Cluster'. Under 'Interactive Clusters', there's one entry: 'MyCluster' with a status of 'Pending' (indicated by a red box). The table columns include Name, State, Nodes, Driver, Worker, Runtime, and Creator. The 'Creator' column shows 'Ignacio.Castineiras@cit.ie'. Under 'Automated Clusters', it says 'No clusters found'.

| Name      | State   | Nodes      | Driver              | Worker              | Runtime                       | Creator                    |
|-----------|---------|------------|---------------------|---------------------|-------------------------------|----------------------------|
| MyCluster | Pending | 1 (0 spot) | Community Optimized | Community Optimized | 5.5 LTS (includes Apache Sp.. | Ignacio.Castineiras@cit.ie |

# Databricks: An Online Platform for Data Engineers

3. The cluster will take a couple of minutes to be set up:

The screenshot shows the Databricks interface with the 'Clusters' tab selected. On the left, there is a sidebar with icons for Home, Workspace, Recents, Data, Clusters (which is selected), Jobs, and Search. The main area is titled 'Clusters' and shows a table of clusters. A blue button at the top left says '+ Create Cluster'. Under 'Interactive Clusters', there is one entry: 'MyCluster' (indicated by a green dot) with a status of 'Running'. The table columns are Name, State, Nodes, Driver, Worker, Runtime, and Creator. The 'State' column for MyCluster is highlighted with a red box. Below the table, under 'Automated Clusters', it says 'No clusters found'. At the bottom right of the main area, there is a small note: '5.5 LTS (includes Apache Sp.. Ignacio.Castineiras@cit.ie)'.

| Name      | State   | Nodes      | Driver              | Worker              | Runtime                       | Creator                    |
|-----------|---------|------------|---------------------|---------------------|-------------------------------|----------------------------|
| MyCluster | Running | 1 (0 spot) | Community Optimized | Community Optimized | 5.5 LTS (includes Apache Sp.. | Ignacio.Castineiras@cit.ie |

# Databricks: An Online Platform for Data Engineers

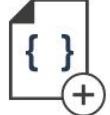
3. We can come back to the main page of the web interface by clicking in **databricks**.



Welcome to  databricks™

  
[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
-  Create Table
-  New Cluster

**Recents**

-  34\_job\_inspection\_5.scala
-  33\_job\_inspection\_4.scala
-  32\_job\_inspection\_3.scala

**What's new in v3.2**

- Instance Pools
- Databricks Runtime 6.0 will drop Python 2 support

[View latest release notes](#)

# Databricks: An Online Platform for Data Engineers

*How to...*

Create a new program / Spark Application.

# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.



Welcome to databricks™

The screenshot shows the Databricks workspace interface. On the left, there's a sidebar with icons for Home, Workspace (highlighted with a red box), Recents, Data, Clusters, Jobs, and Search. The main area has three main sections: 'Explore the Quickstart Tutorial', 'Import & Explore Data', and 'Create a Blank Notebook'. Below these are 'Common Tasks' (New Notebook, Create Table, New Cluster), 'Recents' (list of recent Scala notebooks), and 'What's new in v3.2' (list of changes). A 'Welcome to databricks™' message is at the top.

**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
- Create Table
- New Cluster

**Recents**

- 34\_job\_inspection\_5.scala
- 33\_job\_inspection\_4.scala
- 32\_job\_inspection\_3.scala

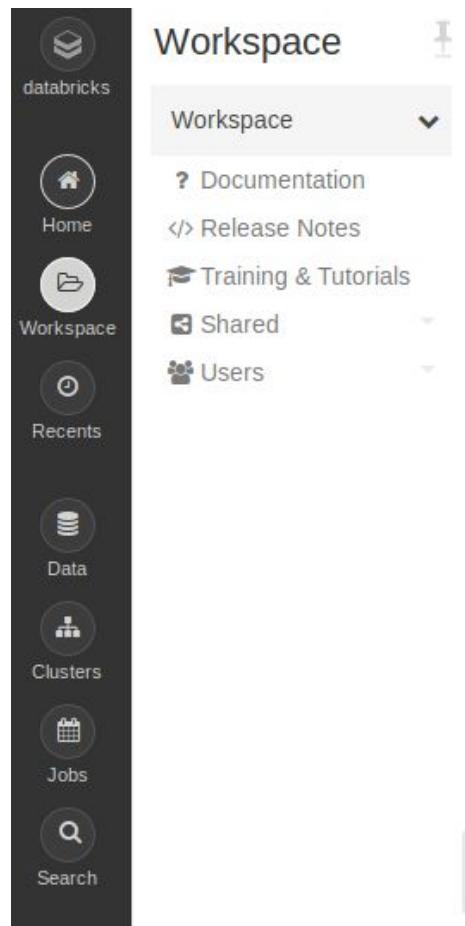
**What's new in v3.2**

- Instance Pools
- Databricks Runtime 6.0 will drop Python 2 support

[View latest release notes](#)

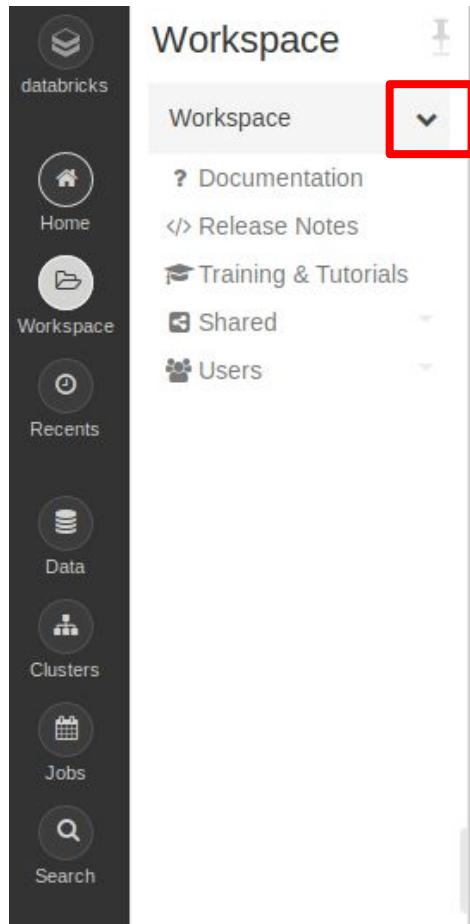
# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.



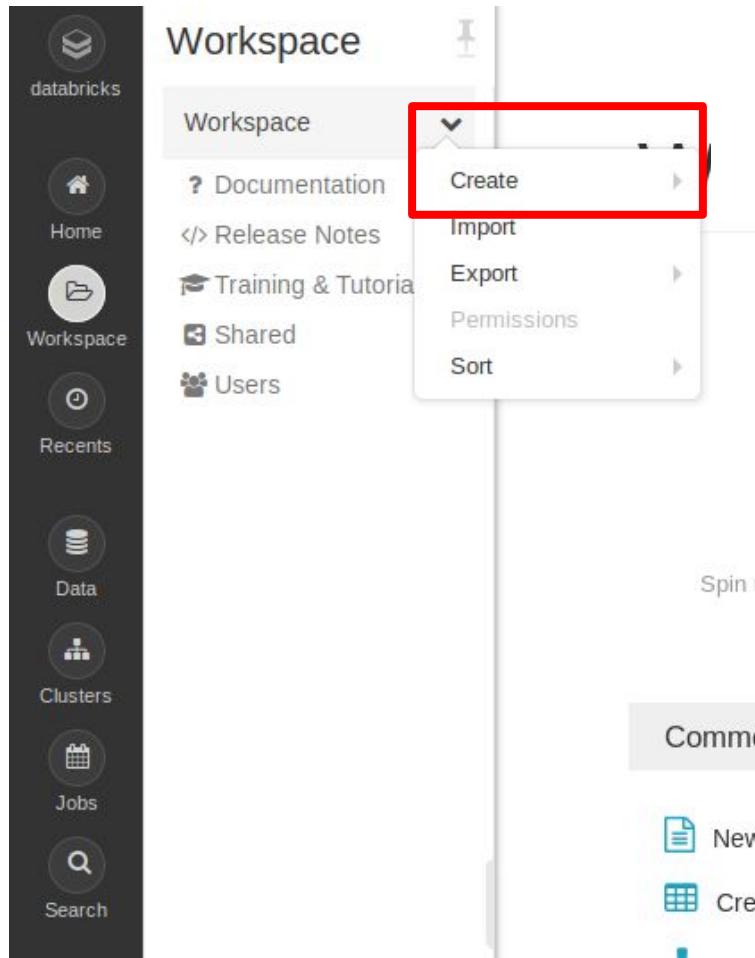
# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.



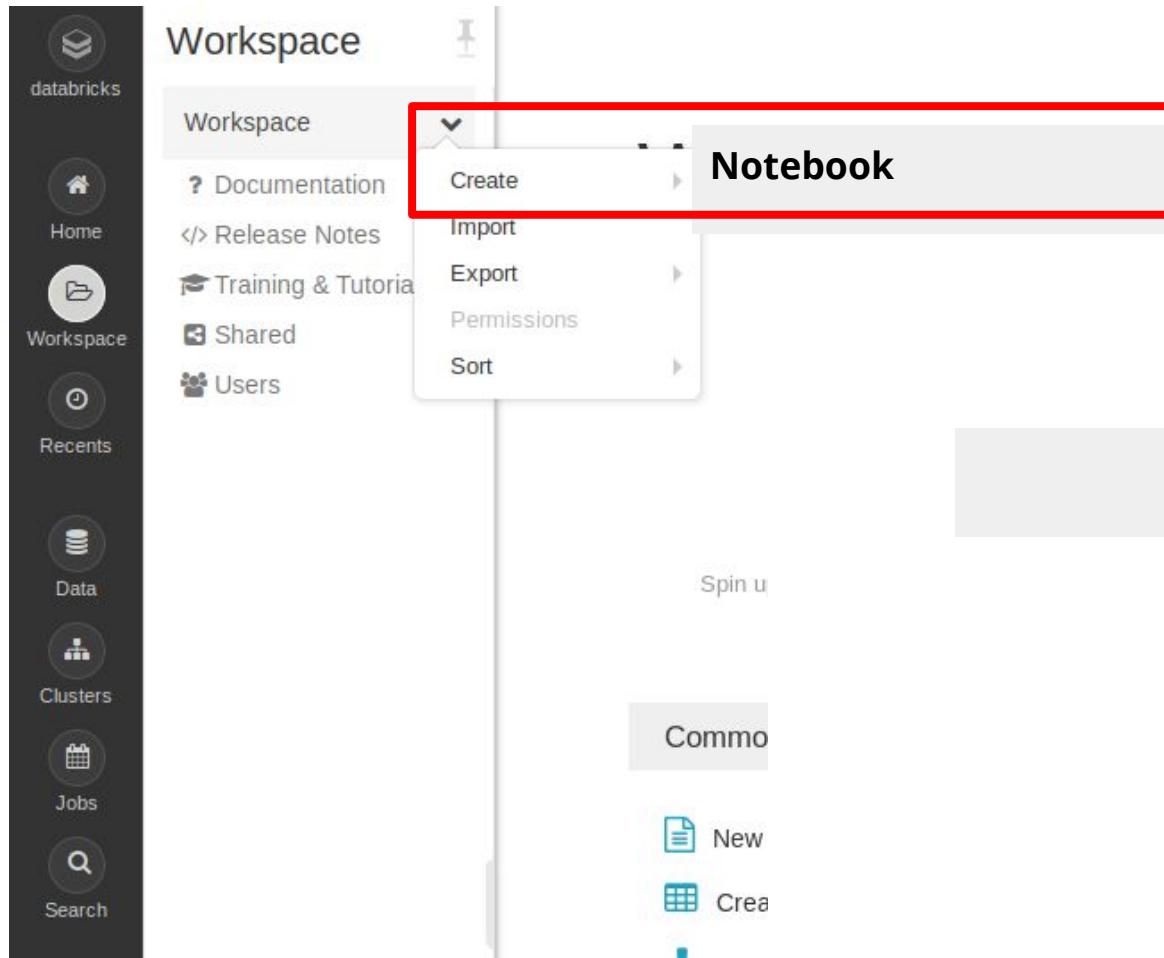
# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.



# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.



# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.

The screenshot shows the Databricks workspace interface. On the left, there is a sidebar with various icons for Workspace, Home, Workspace, Training & Tutorials, Shared, Users, Data, Clusters, Jobs, and Search. The main area displays a "Welcome to databricks™" message with a "Create Notebook" button. A modal dialog box titled "Create Notebook" is open, containing fields for "Name" (set to "my\_first\_program.py"), "Language" (set to "Python"), and "Cluster" (set to "MyCluster"). At the bottom right of the dialog are "Cancel" and "Create" buttons. Below the dialog, there are sections for "Common Tasks" (with "New Notebook" and "Create Table" options) and "Recents" (with a placeholder message: "Recent files appear here as you work").

# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.

The screenshot shows the Databricks workspace interface. On the left, there is a sidebar with various icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area displays a "Welcome to databricks™" message with a "Create Notebook" button. A modal dialog box is open, titled "Create Notebook". It contains three input fields: "Name" (with the value "my\_first\_program.py" highlighted by a red box), "Language" (set to "Python"), and "Cluster" (set to "MyCluster"). At the bottom right of the dialog are "Cancel" and "Create" buttons.

Workspace

Workspace

Documentation

Release Notes

Training & Tutorials

Shared

Users

Welcome to databricks™

Create Notebook

Name: my\_first\_program.py

Language: Python

Cluster: MyCluster

Explore the Quickstart Tutorial

Spin up a cluster, run queries on preloaded data, and see results in 5 minutes.

Common Tasks

New Notebook

Create Table

Recents

Recent files appear here as you work.

# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.

The screenshot shows the Databricks workspace interface. On the left, there is a sidebar with various icons for Home, Workspace, Data, Clusters, Jobs, and Search. The main area displays a "Welcome to databricks" message with a "Create Notebook" button. A modal window titled "Create Notebook" is open, prompting for "Name" (set to "my\_first\_program.py"), "Language" (set to "Python", which is highlighted with a red box), and "Cluster" (set to "MyCluster"). At the bottom right of the modal are "Cancel" and "Create" buttons.

Workspace

Workspace

Documentation

Release Notes

Training & Tutorials

Shared

Users

Welcome to databricks

Create Notebook

Name: my\_first\_program.py

Language: Python

Cluster: MyCluster

Explore the Quickstart Tutorial

Spin up a cluster, run queries on preloaded data, and see results in 5 minutes.

Common Tasks

New Notebook

Create Table

Recents

Recent files appear here as you work.

# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.

The screenshot shows the Databricks workspace interface. On the left, there is a sidebar with various icons for Workspace, Home, Training & Tutorials, Shared, Users, Recents, Data, Clusters, Jobs, and Search. The main area displays a "Welcome to databricks™" message with a "Create Notebook" button. A modal dialog box titled "Create Notebook" is open, prompting for a "Name" (set to "my\_first\_program.py"), "Language" (set to "Python"), and "Cluster" (set to "MyCluster"). The "Cluster" field is highlighted with a red box. At the bottom right of the dialog are "Cancel" and "Create" buttons.

# Databricks: An Online Platform for Data Engineers

4. Our next step is to create a new Spark application as a new notebook in our **Workspace**.

The screenshot shows the Databricks workspace interface. On the left, there is a sidebar with various navigation icons: Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The 'Workspace' icon is selected. The main area displays a 'Welcome to databricks™' message with a 'Create Notebook' button. A modal dialog box titled 'Create Notebook' is open in the center. It contains three input fields: 'Name' with the value 'my\_first\_program.py', 'Language' set to 'Python', and 'Cluster' set to 'MyCluster'. At the bottom right of the modal, there are 'Cancel' and 'Create' buttons, with 'Create' being highlighted by a red rectangle.

Workspace

Workspace

Documentation

Release Notes

Training & Tutorials

Shared

Users

Welcome to databricks™

Create Notebook

Name: my\_first\_program.py

Language: Python

Cluster: MyCluster

Explore the Quickstart Tutorial

Spin up a cluster, run queries on preloaded data, and see results in 5 minutes.

Common Tasks

New Notebook

Create Table

Recents

Recent files appear here as you work.

# Databricks: An Online Platform for Data Engineers

4. We are redirected to the main web page to edit our notebook.  
At the beginning it is empty.

The screenshot shows the Databricks workspace interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area displays a notebook titled "my\_first\_program.py (Python)". The notebook has one command cell, "Cmd 1", which contains the number "1". Below the cell is the instruction "Shift+Enter to run" and a link to "shortcuts". The top navigation bar includes tabs for "File", "View: Code", "Permissions", "Run All", "Clear", and buttons for "Publish", "Comments", "Runs", and "Revision history".

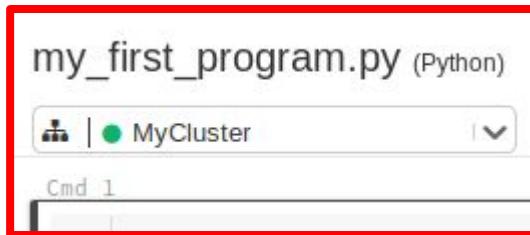
# Databricks: An Online Platform for Data Engineers

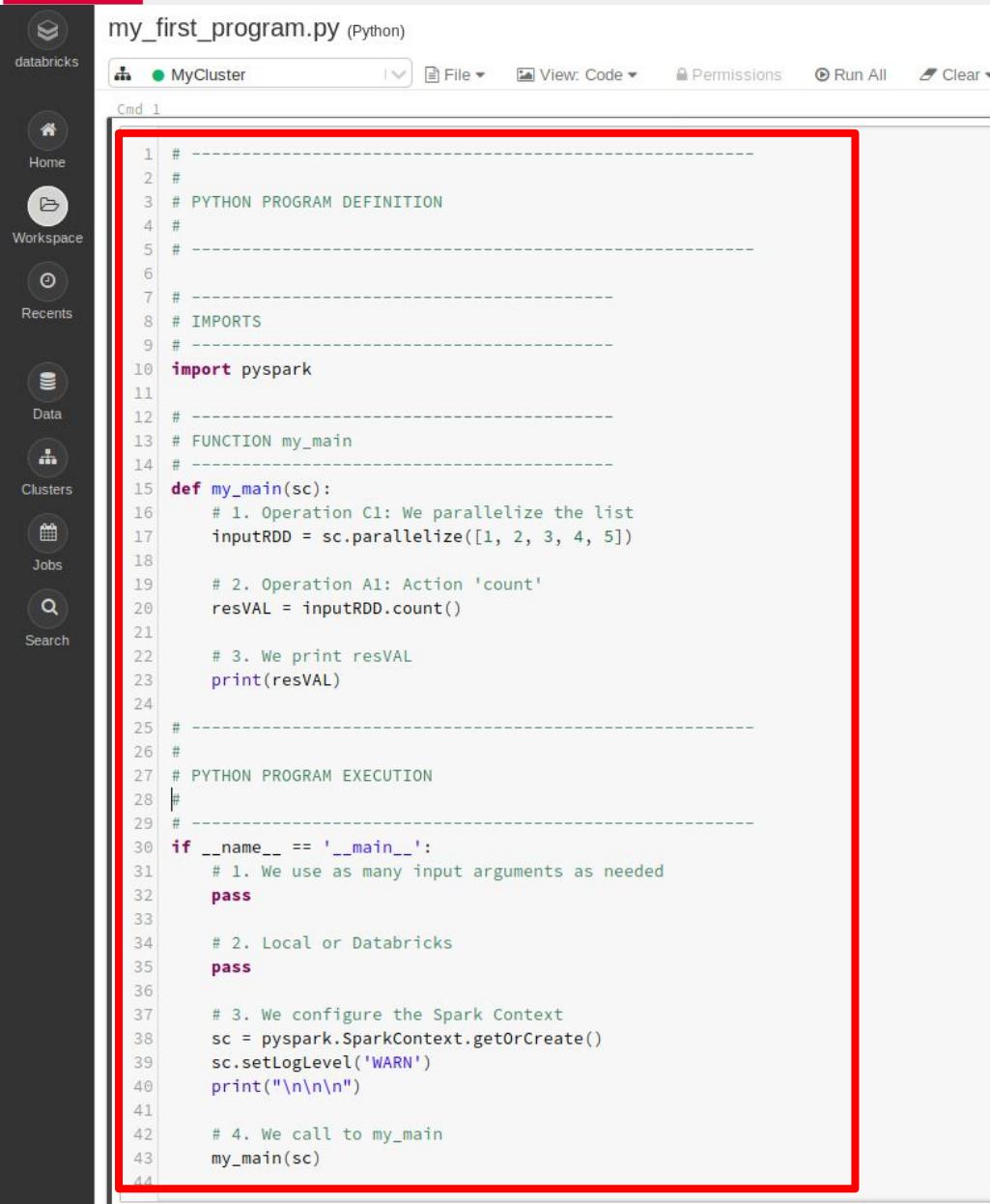
4. We are redirected to the main web page to edit our notebook.  
As we can see our program called **my\_first\_program.py** is automatically attached to our cluster **MyCluster**. Otherwise we can attach it by ourselves.

The screenshot shows the Databricks workspace interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area displays a notebook titled "my\_first\_program.py (Python)". Above the code editor, there is a dropdown menu showing "MyCluster" selected. The top navigation bar includes File, View, Permissions, Run All, Clear, Publish, Comments, Runs, and Revision history. The code editor itself is currently empty, with the placeholder "Shift+Enter to run" visible.

# Databricks: An Online Platform for Data Engineers

4. We are redirected to the main web page to edit our notebook.  
As we can see our program called **my\_first\_program.py** is automatically attached to our cluster **MyCluster**. Otherwise we can attach it by ourselves.





my\_first\_program.py (Python)

MyCluster

File View: Code Permissions Run All Clear

Cmd 1

```
1 # -----
2 #
3 # PYTHON PROGRAM DEFINITION
4 #
5 # -----
6 #
7 # -----
8 # IMPORTS
9 #
10 import pyspark
11 #
12 # -----
13 # FUNCTION my_main
14 #
15 def my_main(sc):
16     # 1. Operation C1: We parallelize the list
17     inputRDD = sc.parallelize([1, 2, 3, 4, 5])
18 #
19     # 2. Operation A1: Action 'count'
20     resVAL = inputRDD.count()
21 #
22     # 3. We print resVAL
23     print(resVAL)
24 #
25 # -----
26 #
27 # PYTHON PROGRAM EXECUTION
28 #
29 #
30 if __name__ == '__main__':
31     # 1. We use as many input arguments as needed
32     pass
33 #
34     # 2. Local or Databricks
35     pass
36 #
37     # 3. We configure the Spark Context
38     sc = pyspark.SparkContext.getOrCreate()
39     sc.setLogLevel('WARN')
40     print("\n\n\n")
41 #
42     # 4. We call to my_main
43     my_main(sc)
44
```

4. We fill in our program in the Jupiter Notebook being created.

The screenshot shows the Databricks workspace interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data, Clusters, Jobs, and Search. The main area is titled "my\_first\_program.py (Python)". It shows a single code cell labeled "Cmd 1" containing the following Python code:

```
1 # -----
2 #
3 # PYTHON PROGRAM DEFINITION
4 #
5 # -----
6
7 # -----
8 # IMPORTS
9 #
10 import pyspark
11
12 # -----
13 # FUNCTION my_main
14 #
15 def my_main(sc):
16     # 1. Operation C1: We parallelize the list
17     inputRDD = sc.parallelize([1, 2, 3, 4, 5])
18
19     # 2. Operation A1: Action 'count'
20     resVAL = inputRDD.count()
21
22     # 3. We print resVAL
23     print(resVAL)
24
25 #
26 #
27 # PYTHON PROGRAM EXECUTION
28 #
29 #
30 if __name__ == '__main__':
31     # 1. We use as many input arguments as needed
32     pass
33
34     # 2. Local or Databricks
35     pass
36
37     # 3. We configure the Spark Context
38     sc = pyspark.SparkContext.getOrCreate()
39     sc.setLogLevel('WARN')
40     print("\n\n\n")
41
42     # 4. We call to my_main
43     my_main(sc)
```

4. We fill in our program in the Jupiter Notebook being created.

### Note:

As not being a big fan of Jupiter Notebooks, let's please treat them as if they were regular Python programs.

That is, let's fill the entire Python program into the **Cmd 1** cell of the notebook.

# Databricks: An Online Platform for Data Engineers



The screenshot shows the Databricks workspace interface. On the left, there's a sidebar with icons for 'databricks' (home), 'Home', and 'Workspace'. The main area displays a Python notebook titled 'my\_first\_program.py (Python)'. The notebook has one cell named 'Cmd 1' which contains the following code:

```
1 # -----
2 #
3 # PYTHON PROGRAM DEFINITION
4 #
5 # -----
```

A red box highlights the 'Cmd 1' cell.

4. We fill in our program in the Jupiter Notebook being created.

Note:

As not being a big fan of Jupiter Notebooks, let's please treat them as if they were regular Python programs.

That is, let's fill the entire Python program into the **Cmd 1** cell of the notebook.

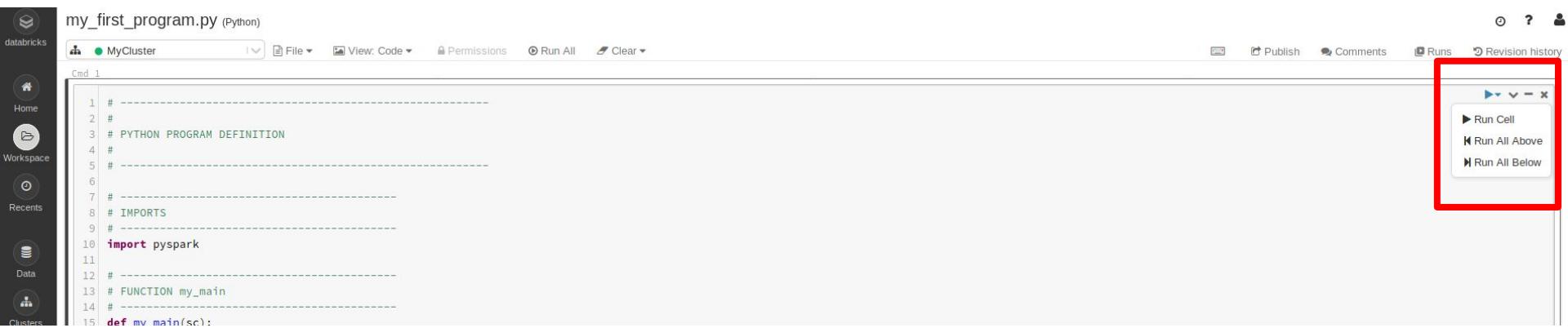
# Databricks: An Online Platform for Data Engineers

*How to...*

Run the new program / Spark Application.

# Databricks: An Online Platform for Data Engineers

- Once the program (Spark application) is ready, we can execute it in our cluster.



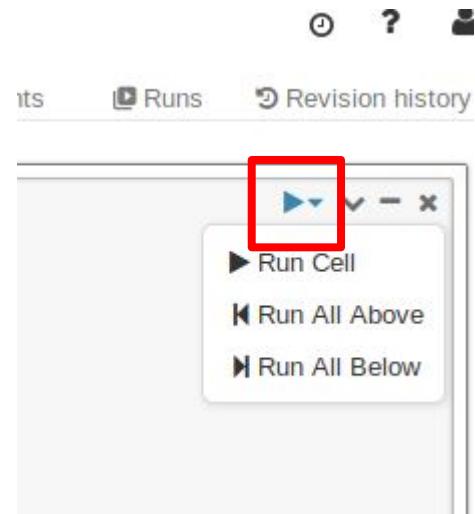
The screenshot shows the Databricks workspace interface. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data, and Clusters. The main area displays a Python notebook titled "my\_first\_program.py". The code in the notebook is:

```
1 # -----
2 #
3 # PYTHON PROGRAM DEFINITION
4 #
5 #
6 #
7 # -----
8 # IMPORTS
9 #
10 import pyspark
11 #
12 # -----
13 # FUNCTION my_main
14 #
15 def my_main(sc):
```

At the top of the notebook, there are several tabs: "MyCluster" (selected), "File", "View: Code", "Permissions", "Run All", "Clear", "Publish", "Comments", "Runs", and "Revision history". To the right of the code, there's a panel with three run control buttons: "Run Cell", "Run All Above", and "Run All Below". A red box highlights this panel.

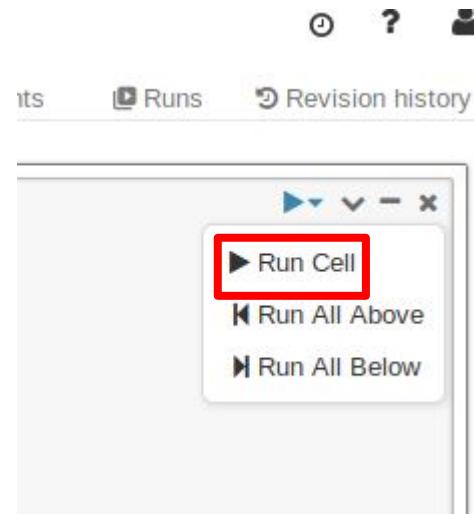
# Databricks: An Online Platform for Data Engineers

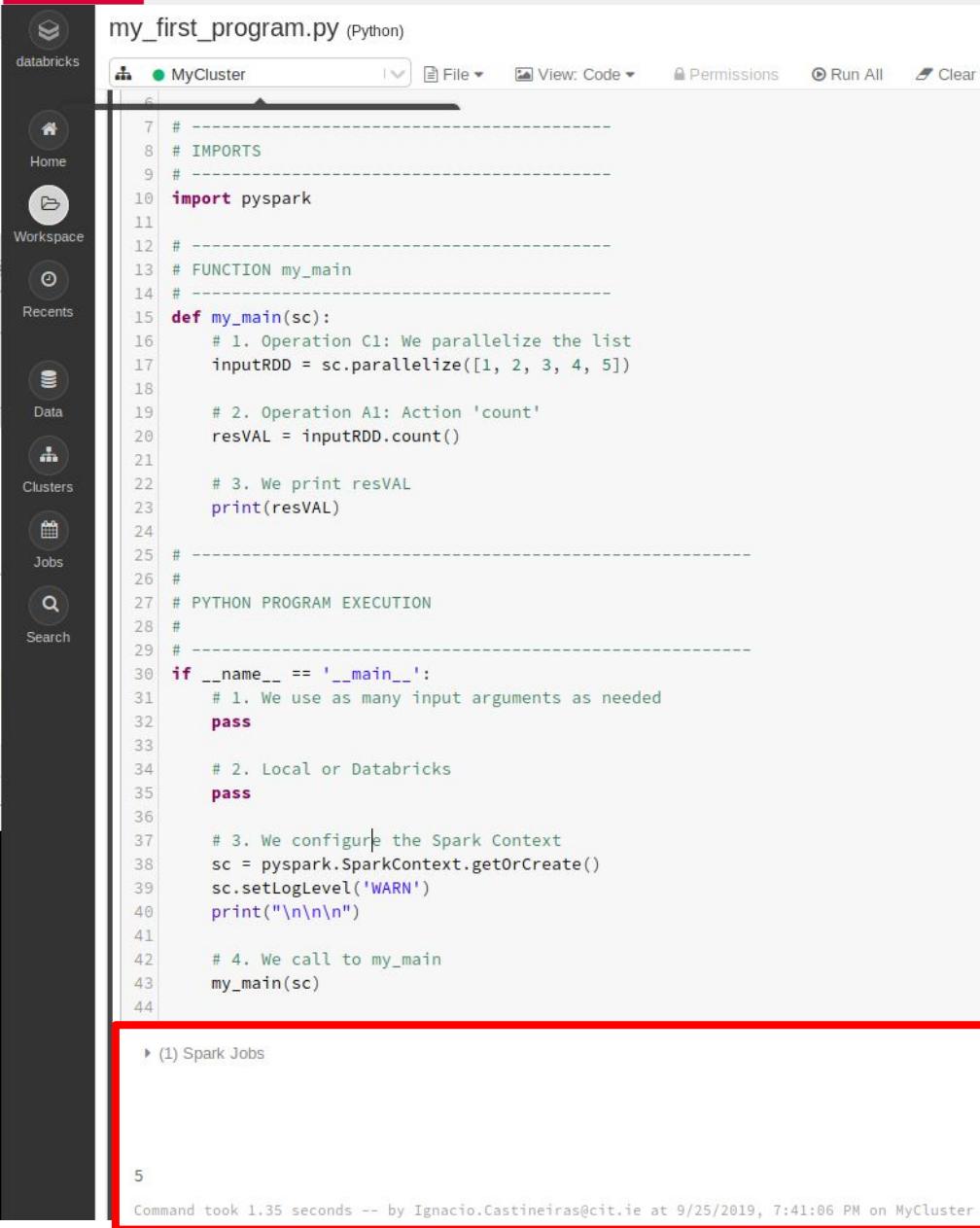
- Once the program (Spark application) is ready, we can execute it in our cluster.



# Databricks: An Online Platform for Data Engineers

- Once the program (Spark application) is ready, we can execute it in our cluster.





my\_first\_program.py (Python)

MyCluster

File View: Code Permissions Run All Clear

```
6
7 # -----
8 # IMPORTS
9 # -----
10 import pyspark
11
12 # -----
13 # FUNCTION my_main
14 #
15 def my_main(sc):
16     # 1. Operation C1: We parallelize the list
17     inputRDD = sc.parallelize([1, 2, 3, 4, 5])
18
19     # 2. Operation A1: Action 'count'
20     resVAL = inputRDD.count()
21
22     # 3. We print resVAL
23     print(resVAL)
24
25 # -----
26 #
27 # PYTHON PROGRAM EXECUTION
28 #
29 #
30 if __name__ == '__main__':
31     # 1. We use as many input arguments as needed
32     pass
33
34     # 2. Local or Databricks
35     pass
36
37     # 3. We configure the Spark Context
38     sc = pyspark.SparkContext.getOrCreate()
39     sc.setLogLevel('WARN')
40     print("\n\n\n")
41
42     # 4. We call to my_main
43     my_main(sc)
44
```

(1) Spark Jobs

5

Command took 1.35 seconds -- by Ignacio.Castineiras@cit.ie at 9/25/2019, 7:41:06 PM on MyCluster

And we can evaluate its result.

# Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.

```
34      # 2. Local or Databricks
35      pass
36
37      # 3. We configure the Spark Context
38      sc = pyspark.SparkContext.getOrCreate()
39      sc.setLogLevel('WARN')
40      print("\n\n\n")
41
42      # 4. We call to my_main
43      my_main(sc)
44
```

▶ (1) Spark Jobs

5

Command took 1.35 seconds -- by Ignacio.Castineiras@cit.ie at 9/25/2019, 7:41:06 PM on MyCluster

# Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.

```
34      # 2. Local or Databricks
35      pass
36
37      # 3. We configure the Spark Context
38      sc = pyspark.SparkContext.getOrCreate()
39      sc.setLogLevel('WARN')
40      print("\n\n\n")
41
42      # 4. We call to my_main
43      my_main(sc)
44
```

▶ (1) Spark Jobs

5

Command took 1.35 seconds -- by Ignacio.Castineiras@cit.ie at 9/25/2019, 7:41:06 PM on MyCluster

# Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.

```
34      # 2. Local or Databricks
35      pass
36
37      # 3. We configure the Spark Context
38      sc = pyspark.SparkContext.getOrCreate()
39      sc.setLogLevel('WARN')
40      print("\n\n\n")
41
42      # 4. We call to my_main
43      my_main(sc)
44
```

▶ (1) Spark Jobs

5

Command took 1.35 seconds -- by Ignacio.Castineiras@cit.ie at 9/25/2019, 7:41:06 PM on MyCluster

# Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.

```
34     # 2. Local or Databricks
35     pass
36
37     # 3. We configure the Spark Context
38     sc = pyspark.SparkContext.getOrCreate()
39     sc.setLogLevel('WARN')
40     print("\n\n\n")
41
42     # 4. We call to my_main
43     my_main(sc)
44
```

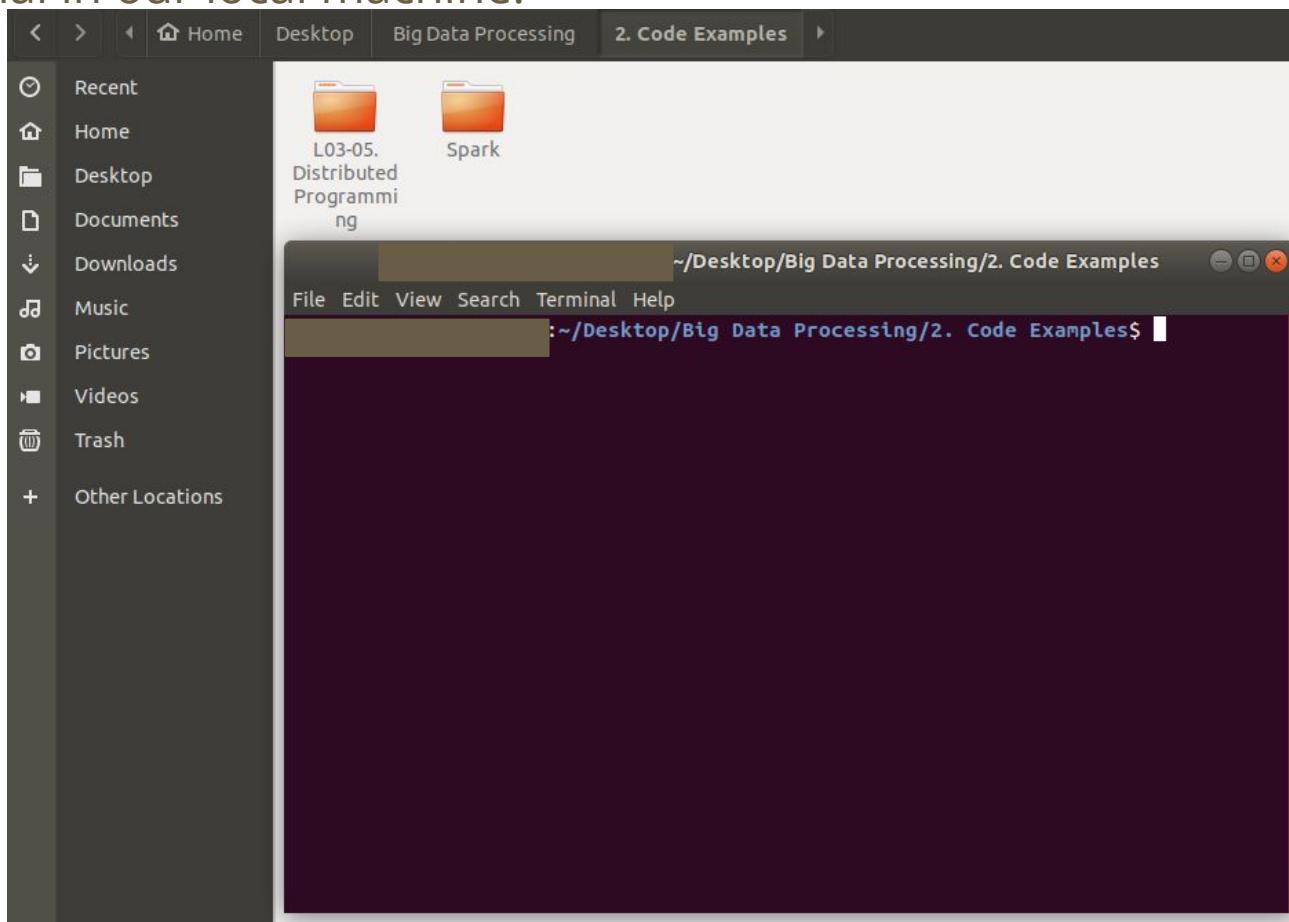
▼ (1) Spark Jobs  
▶ Job 0 [View](#) (Stages: 1/1)

# Databricks: An Online Platform for Data Engineers

*How to...*  
Install the Databricks  
Command Line Application (CLI)

# Databricks: An Online Platform for Data Engineers

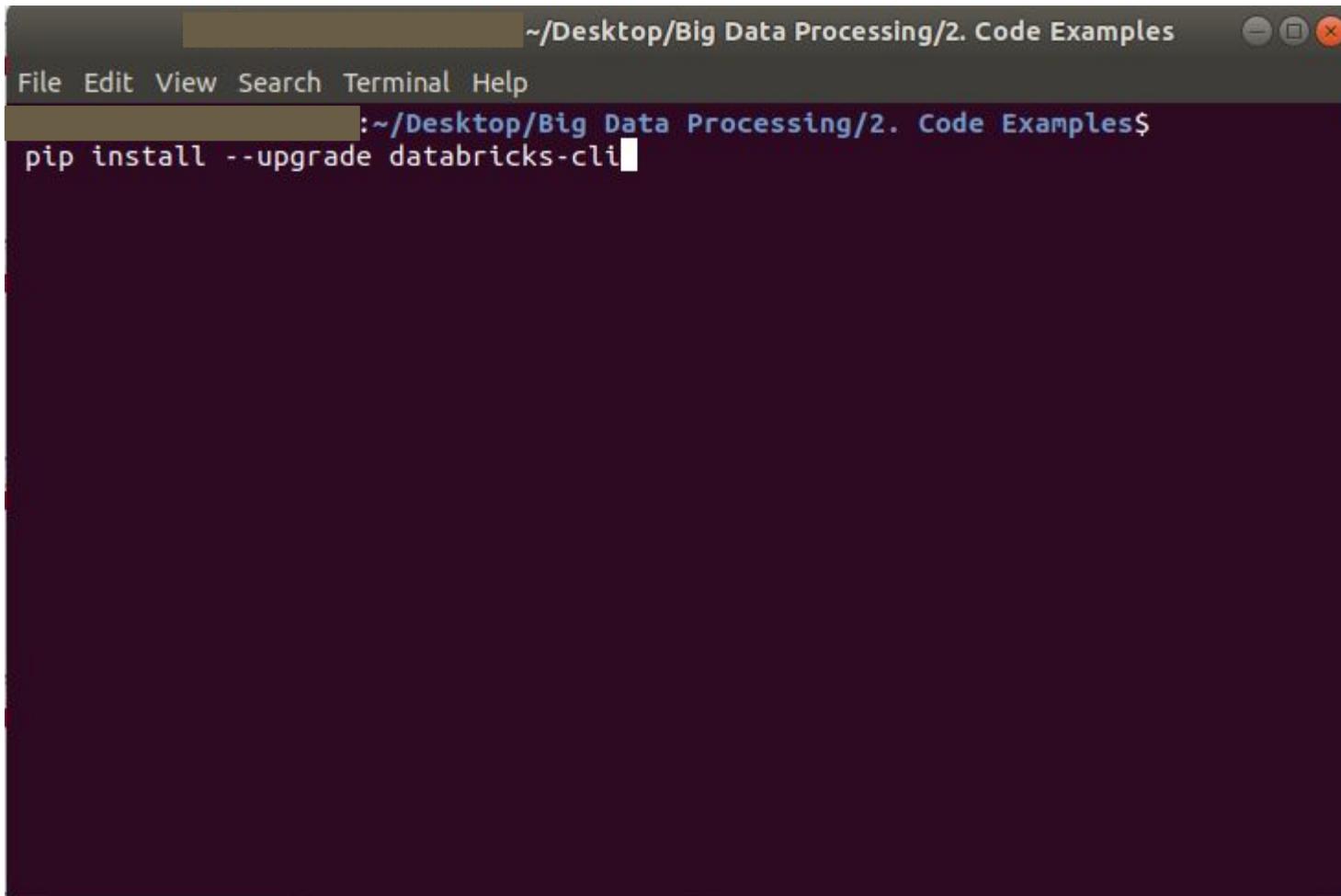
6. Databricks provides a Command Line Interface (CLI).  
The CLI allows us to perform some operations on Databricks from a terminal in our local machine.



# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

We can install the Databricks CLI via **pip**.

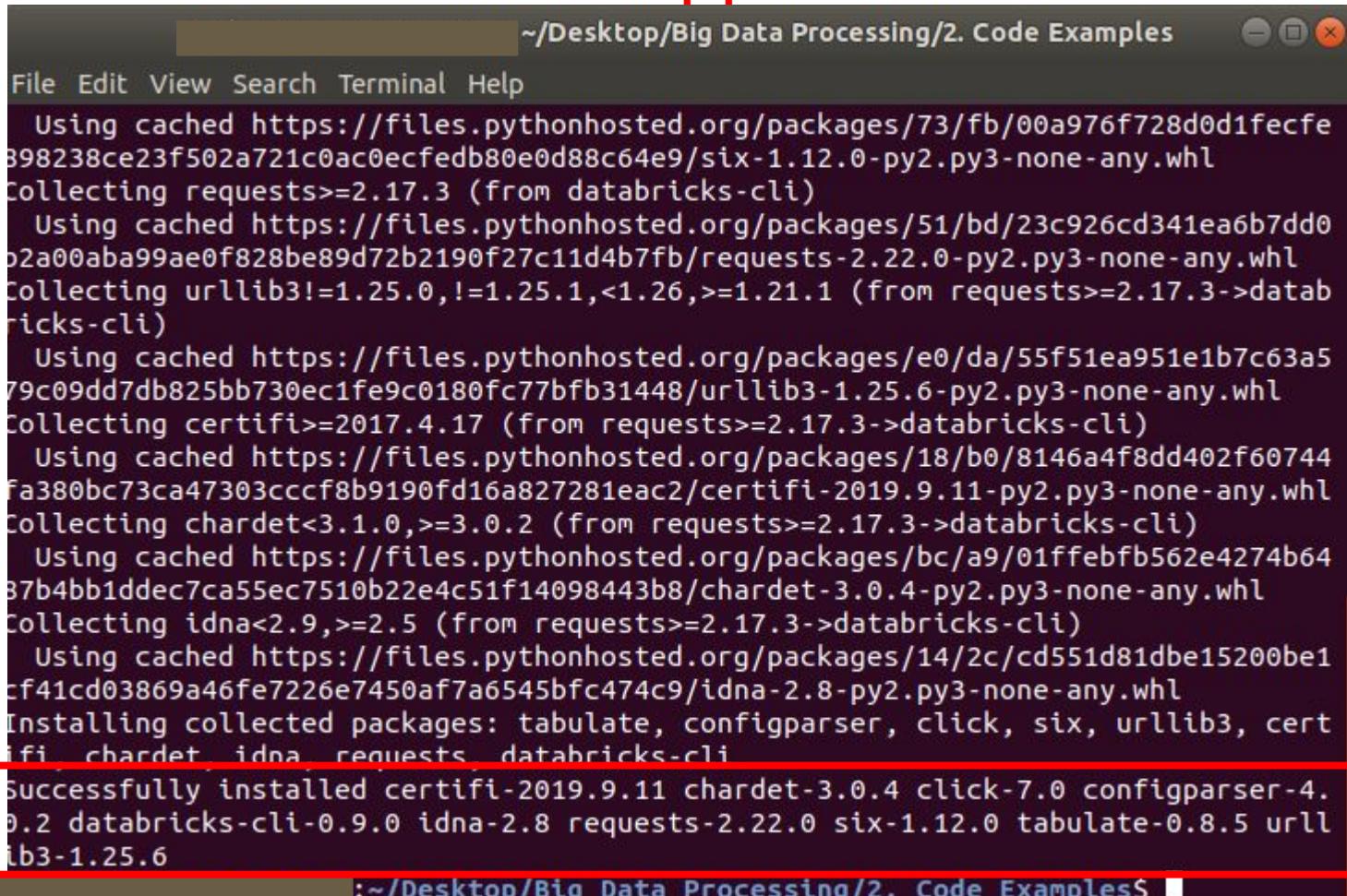


A screenshot of a terminal window titled '~/Desktop/Big Data Processing/2. Code Examples'. The window has a dark background and light-colored text. At the top, there is a menu bar with File, Edit, View, Search, Terminal, and Help. Below the menu is a command prompt line starting with ':~/Desktop/Big Data Processing/2. Code Examples\$'. A cursor is visible at the end of the line, where the command 'pip install --upgrade databricks-cli' is being typed.

# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

We can install the Databricks CLI via **pip**.



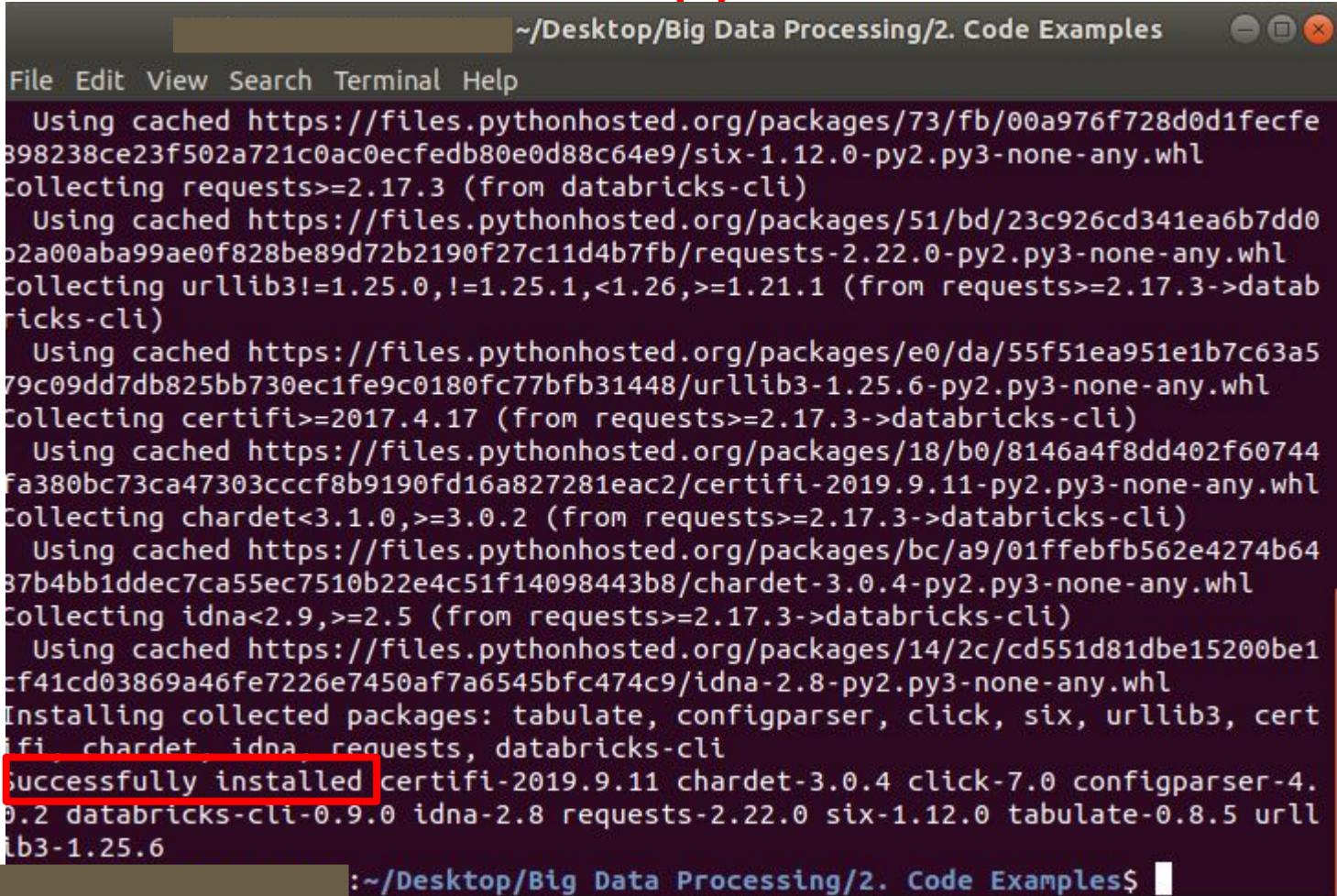
The screenshot shows a terminal window titled "/Desktop/Big Data Processing/2. Code Examples". The terminal is displaying the output of a pip command to install the Databricks CLI. The output shows the download and extraction of various Python packages from pythonhosted.org, including certifi, chardet, idna, requests, tabulate, configparser, click, six, urllib3, and databricks-cli. The final message indicates successful installation of all collected packages.

```
~/Desktop/Big Data Processing/2. Code Examples$ pip install databricks-cli
Using cached https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1fecfe
398238ce23f502a721c0ac0ecfedb80e0d88c64e9/six-1.12.0-py2.py3-none-any.whl
Collecting requests>=2.17.3 (from databricks-cli)
  Using cached https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0
b2a00aba99ae0f828be89d72b2190f27c11d4b7fb/requests-2.22.0-py2.py3-none-any.whl
Collecting urllib3!=1.25.0,!>1.25.1,<1.26,>=1.21.1 (from requests>=2.17.3->datab
ricks-cli)
  Using cached https://files.pythonhosted.org/packages/e0/da/55f51ea951e1b7c63a5
79c09dd7db825bb730ec1fe9c0180fc77bfb31448/urllib3-1.25.6-py2.py3-none-any.whl
Collecting certifi>=2017.4.17 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/18/b0/8146a4f8dd402f60744
fa380bc73ca47303cccf8b9190fd16a827281eac2/certifi-2019.9.11-py2.py3-none-any.whl
Collecting chardet<3.1.0,>=3.0.2 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/bc/a9/01ffebfb562e4274b64
87b4bb1ddec7ca55ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl
Collecting idna<2.9,>=2.5 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/14/2c/cd551d81dbe15200be1
cf41cd03869a46fe7226e7450af7a6545bfc474c9/idna-2.8-py2.py3-none-any.whl
Installing collected packages: tabulate, configparser, click, six, urllib3, cert
ifi, chardet, idna, requests, databricks-cli
Successfully installed certifi-2019.9.11 chardet-3.0.4 click-7.0 configparser-4.
0.2 databricks-cli-0.9.0 idna-2.8 requests-2.22.0 six-1.12.0 tabulate-0.8.5 url
lib3-1.25.6
~/Desktop/Big Data Processing/2. Code Examples$
```

# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

We can install the Databricks CLI via **pip**.



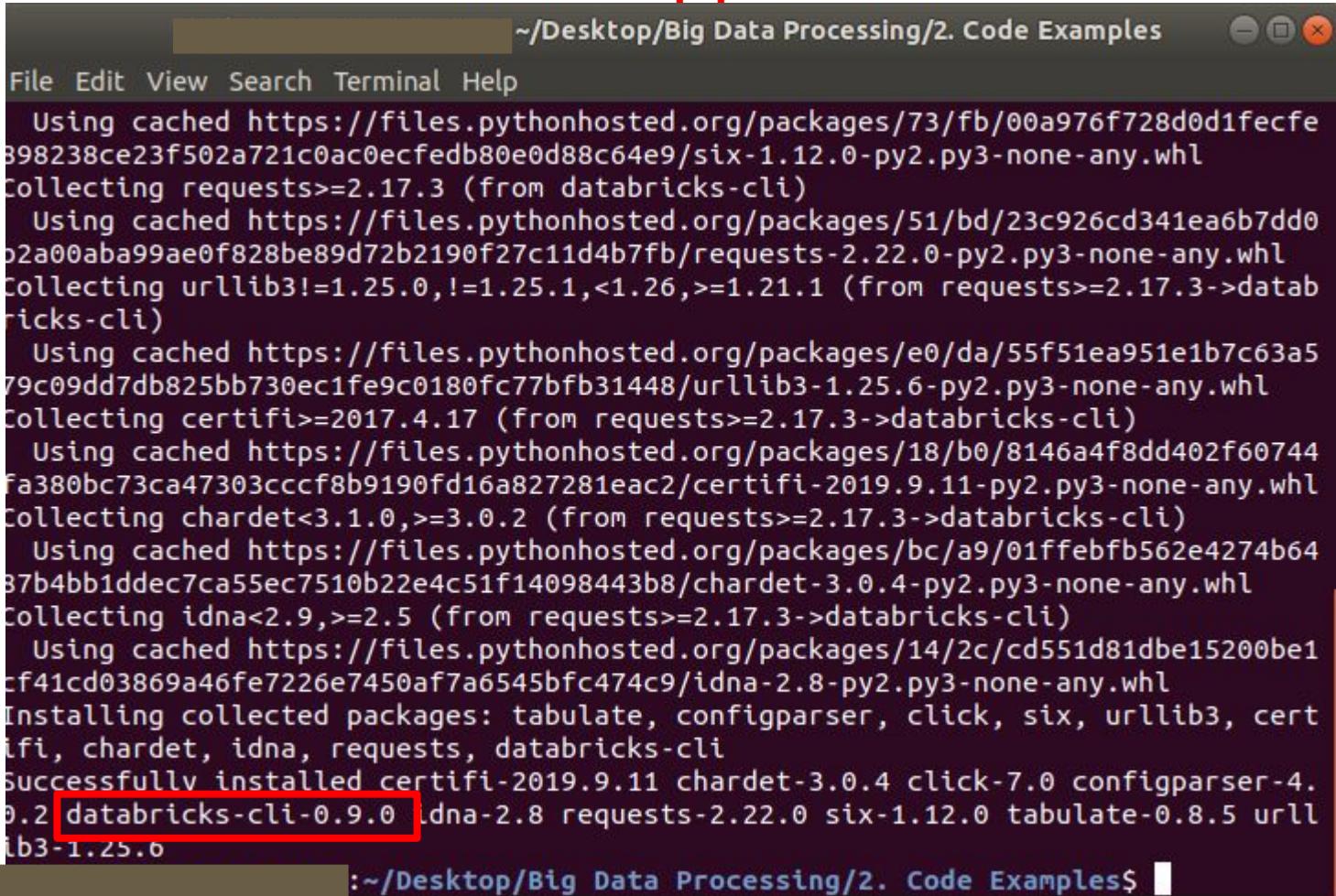
The screenshot shows a terminal window titled "/Desktop/Big Data Processing/2. Code Examples". The window contains the output of a pip command to install the Databricks CLI. The output shows the download and extraction of various Python packages from pythonhosted.org, including certifi, chardet, idna, requests, and databricks-cli. The final line of the output, "Successfully installed certifi-2019.9.11 chardet-3.0.4 click-7.0 configparser-4.0.2 databricks-cli-0.9.0 idna-2.8 requests-2.22.0 six-1.12.0 tabulate-0.8.5 urllib3-1.25.6", is highlighted with a red box.

```
~/Desktop/Big Data Processing/2. Code Examples
File Edit View Search Terminal Help
Using cached https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1fecfe
398238ce23f502a721c0ac0ecfedb80e0d88c64e9/six-1.12.0-py2.py3-none-any.whl
Collecting requests>=2.17.3 (from databricks-cli)
  Using cached https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0
b2a00aba99ae0f828be89d72b2190f27c11d4b7fb/requests-2.22.0-py2.py3-none-any.whl
Collecting urllib3!=1.25.0,!>1.25.1,<1.26,>=1.21.1 (from requests>=2.17.3->datab
ricks-cli)
  Using cached https://files.pythonhosted.org/packages/e0/da/55f51ea951e1b7c63a5
79c09dd7db825bb730ec1fe9c0180fc77bfb31448/urllib3-1.25.6-py2.py3-none-any.whl
Collecting certifi>=2017.4.17 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/18/b0/8146a4f8dd402f60744
fa380bc73ca47303cccf8b9190fd16a827281eac2/certifi-2019.9.11-py2.py3-none-any.whl
Collecting chardet<3.1.0,>=3.0.2 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/bc/a9/01ffebfb562e4274b64
87b4bb1ddec7ca55ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl
Collecting idna<2.9,>=2.5 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/14/2c/cd551d81dbe15200be1
cf41cd03869a46fe7226e7450af7a6545bfc474c9/idna-2.8-py2.py3-none-any.whl
Installing collected packages: tabulate, configparser, click, six, urllib3, cert
ifi, chardet, idna, requests, databricks-cli
Successfully installed certifi-2019.9.11 chardet-3.0.4 click-7.0 configparser-4.
0.2 databricks-cli-0.9.0 idna-2.8 requests-2.22.0 six-1.12.0 tabulate-0.8.5 url
ib3-1.25.6
:~/Desktop/Big Data Processing/2. Code Examples$
```

# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

We can install the Databricks CLI via **pip**.



The screenshot shows a terminal window titled "/Desktop/Big Data Processing/2. Code Examples". The terminal is displaying the output of a pip command to install the Databricks CLI. The output shows the process of collecting and installing various Python packages from pythonhosted.org, including certifi, chardet, idna, requests, and databricks-cli. The final line of the output, "Successfully installed databricks-cli-0.9.0", is highlighted with a red box.

```
~/Desktop/Big Data Processing/2. Code Examples$ pip install databricks-cli
Using cached https://files.pythonhosted.org/packages/73/fb/00a976f728d0d1fecfe
398238ce23f502a721c0ac0ecfedb80e0d88c64e9/six-1.12.0-py2.py3-none-any.whl
Collecting requests>=2.17.3 (from databricks-cli)
  Using cached https://files.pythonhosted.org/packages/51/bd/23c926cd341ea6b7dd0
b2a00aba99ae0f828be89d72b2190f27c11d4b7fb/requests-2.22.0-py2.py3-none-any.whl
Collecting urllib3!=1.25.0,!>1.25.1,<1.26,>=1.21.1 (from requests>=2.17.3->datab
ricks-cli)
  Using cached https://files.pythonhosted.org/packages/e0/da/55f51ea951e1b7c63a5
79c09dd7db825bb730ec1fe9c0180fc77bfb31448/urllib3-1.25.6-py2.py3-none-any.whl
Collecting certifi>=2017.4.17 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/18/b0/8146a4f8dd402f60744
fa380bc73ca47303cccf8b9190fd16a827281eac2/certifi-2019.9.11-py2.py3-none-any.whl
Collecting chardet<3.1.0,>=3.0.2 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/bc/a9/01ffebfb562e4274b64
87b4bb1ddec7ca55ec7510b22e4c51f14098443b8/chardet-3.0.4-py2.py3-none-any.whl
Collecting idna<2.9,>=2.5 (from requests>=2.17.3->databricks-cli)
  Using cached https://files.pythonhosted.org/packages/14/2c/cd551d81dbe15200be1
cf41cd03869a46fe7226e7450af7a6545bfc474c9/idna-2.8-py2.py3-none-any.whl
Installing collected packages: tabulate, configparser, click, six, urllib3, cert
ifi, chardet, idna, requests, databricks-cli
Successfully installed certifi-2019.9.11 chardet-3.0.4 click-7.0 configparser-4.
0.2 databricks-cli-0.9.0 idna-2.8 requests-2.22.0 six-1.12.0 tabulate-0.8.5 url
ib3-1.25.0
~/Desktop/Big Data Processing/2. Code Examples$
```

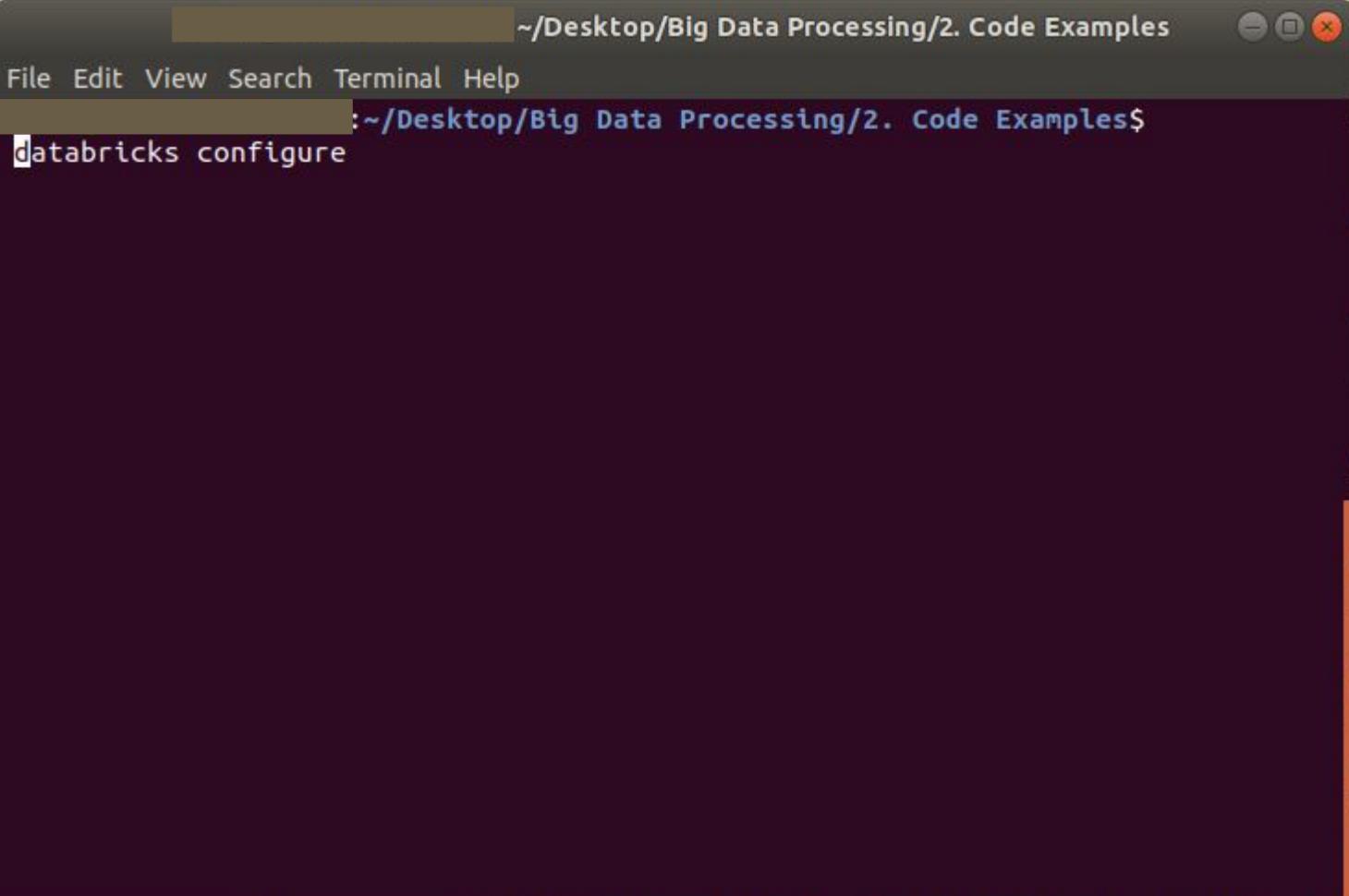
# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).  
Once installed, we must configure the Databricks CLI.

# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

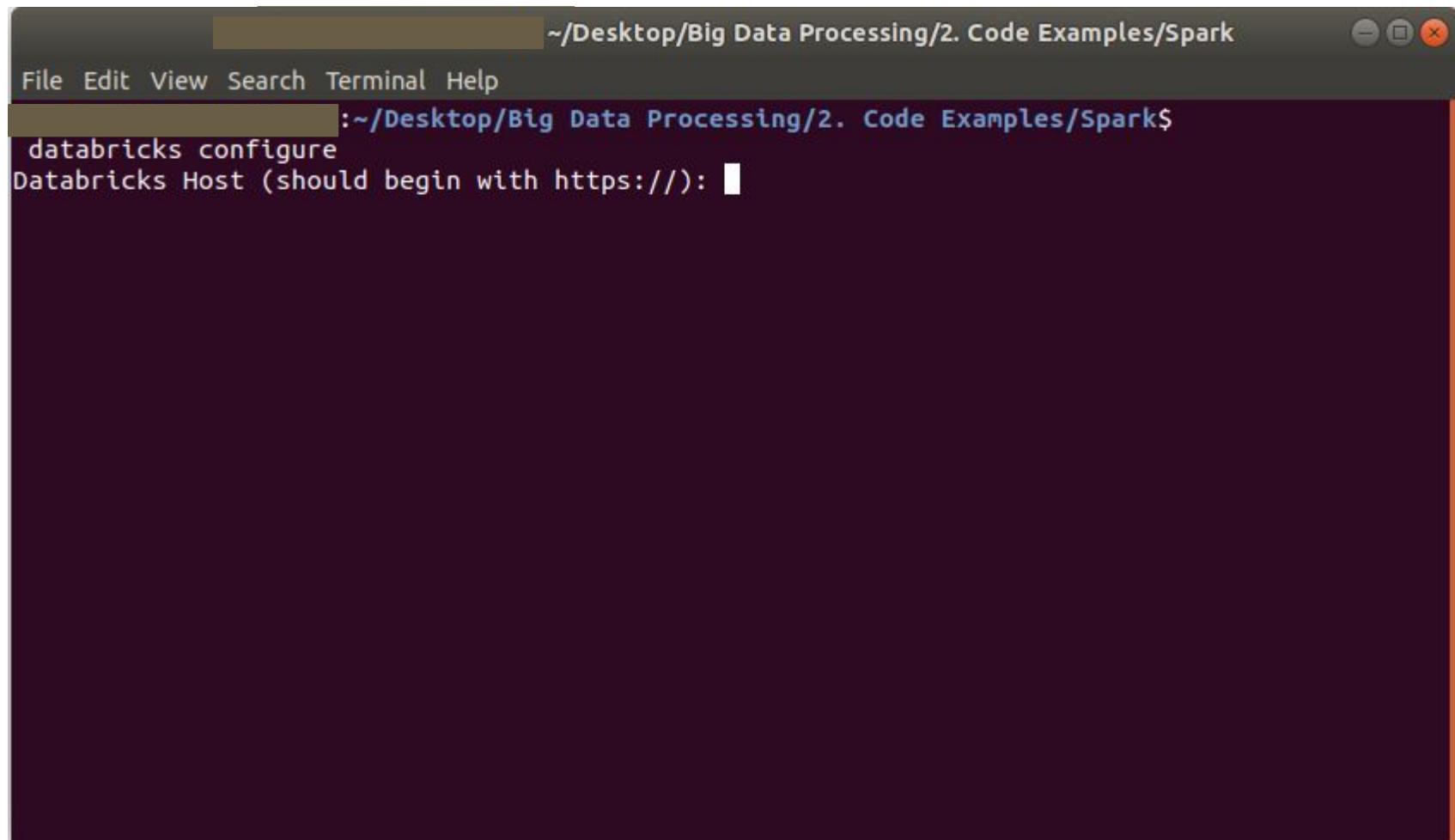
We trigger the command **databricks configure**



A screenshot of a terminal window titled '~/Desktop/Big Data Processing/2. Code Examples'. The window has a dark background and light-colored text. At the top, there is a menu bar with 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. Below the menu bar, the current directory is shown as ':~/Desktop/Big Data Processing/2. Code Examples\$'. A cursor is visible at the beginning of the line, followed by the command 'databricks configure'. The rest of the terminal window is blank, showing a large white space.

# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - a. We must indicate our databricks host.



A screenshot of a terminal window titled '~/Desktop/Big Data Processing/2. Code Examples/Spark'. The window has a dark theme with a light gray header bar. The title bar shows the path. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The main terminal area has a dark background and light-colored text. It displays the command 'databricks configure' followed by the prompt 'Databricks Host (should begin with https://):'. The cursor is visible at the end of the prompt.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark
File Edit View Search Terminal Help
:~/Desktop/Big Data Processing/2. Code Examples/Spark$
databricks configure
Databricks Host (should begin with https://): █
```

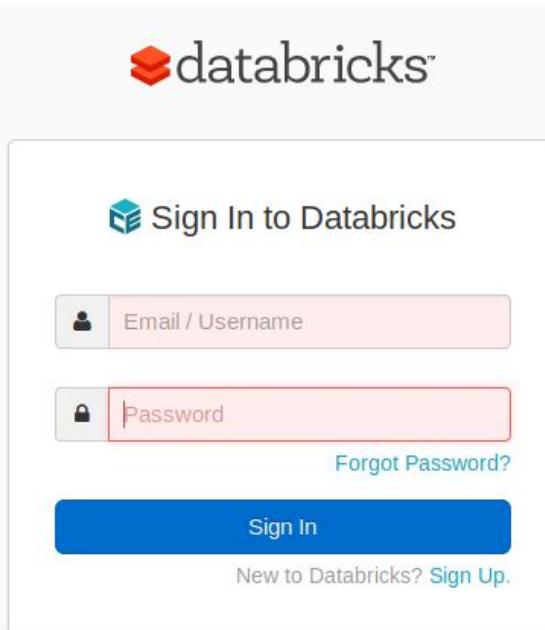
# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).

a. We must indicate our databricks host.

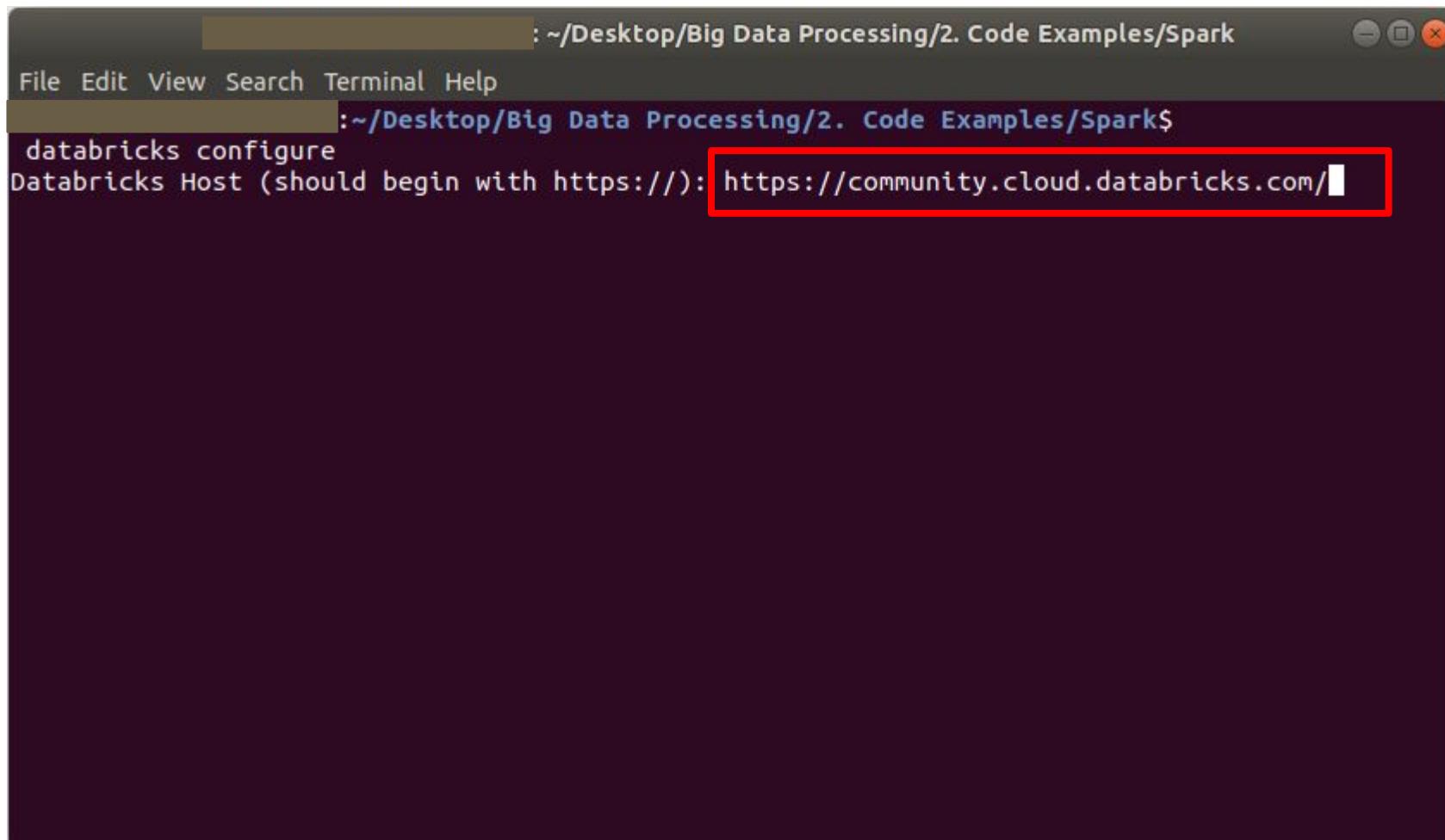
In our case, our host is the website for Databricks Community Edition:

<https://community.cloud.databricks.com>



# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - a. We must indicate our databricks host.

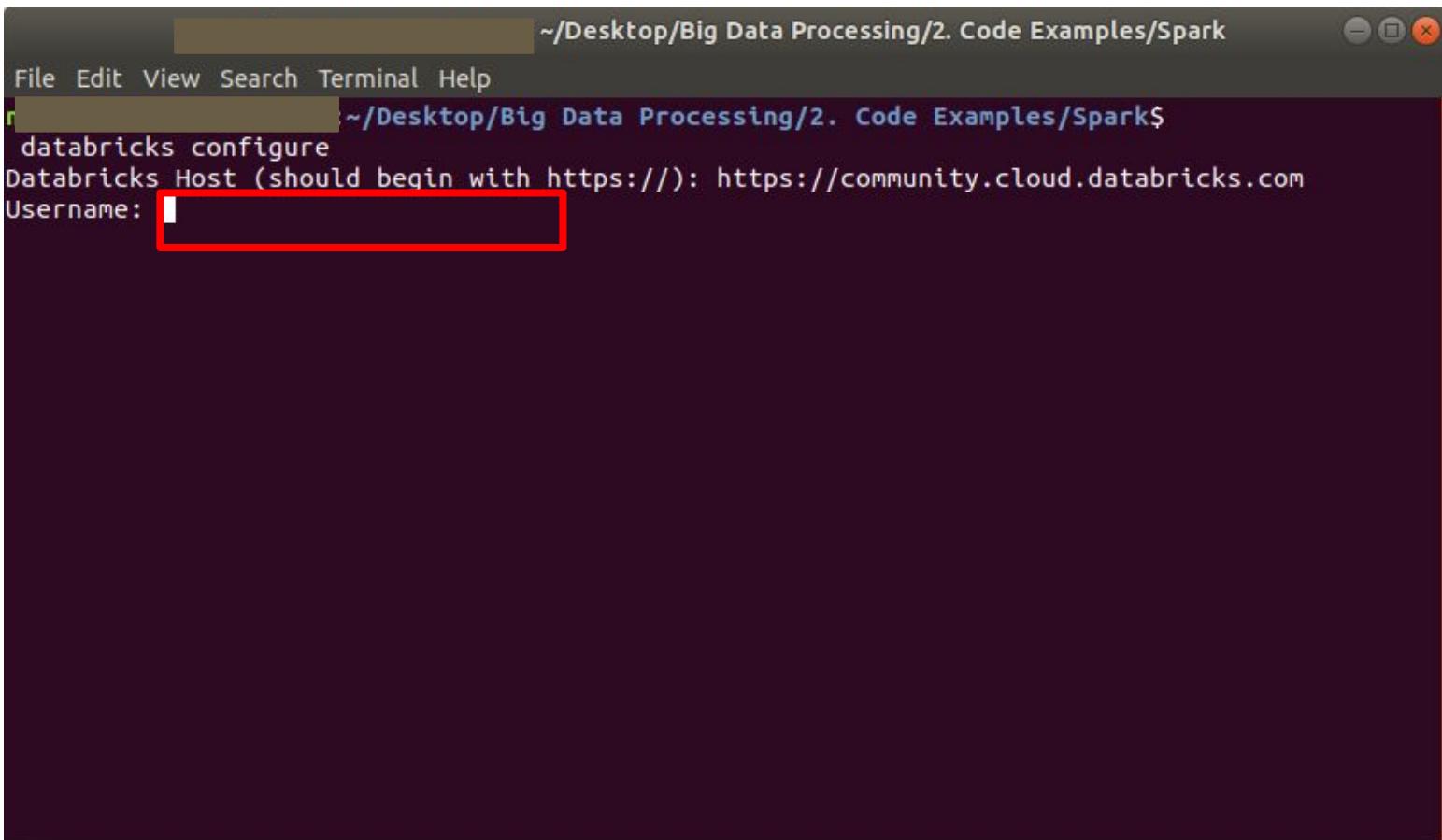


A screenshot of a terminal window titled 'Terminal' with the path '/Desktop/Big Data Processing/2. Code Examples/Spark'. The window has a dark background and light-colored text. A red box highlights the input field where the URL 'https://community.cloud.databricks.com/' is being typed. The terminal shows the command 'databricks configure' followed by the prompt 'Databricks Host (should begin with https://):'. The URL is partially typed in the input field.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark$ databricks configure
Databricks Host (should begin with https://): https://community.cloud.databricks.com/
```

# Databricks: An Online Platform for Data Engineers

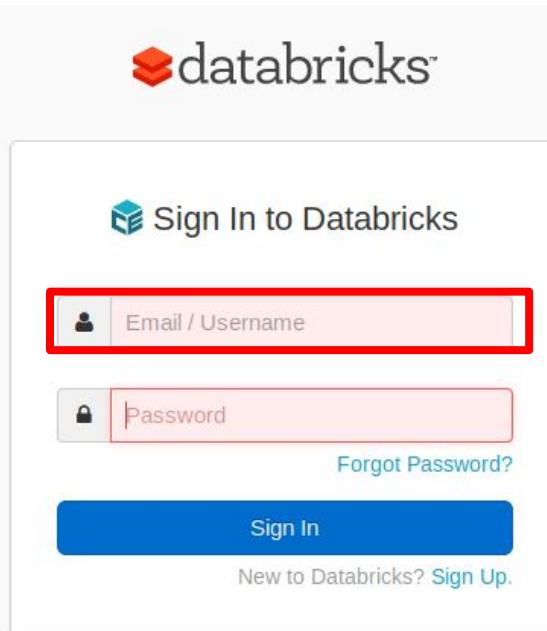
6. Databricks provides a Command Line Interface (CLI).
  - b. We must indicate our username (same as when we sign in at <https://community.cloud.databricks.com>).



The screenshot shows a terminal window with a dark background and light-colored text. The title bar reads: ~/Desktop/Big Data Processing/2. Code Examples/Spark. The menu bar includes: File, Edit, View, Search, Terminal, Help. The command line shows: databricks configure. Below the command, there is a prompt: Databricks Host (should begin with https://): https://community.cloud.databricks.com. A red rectangular box highlights the input field where the URL is typed. The terminal window has standard window controls (minimize, maximize, close) in the top right corner.

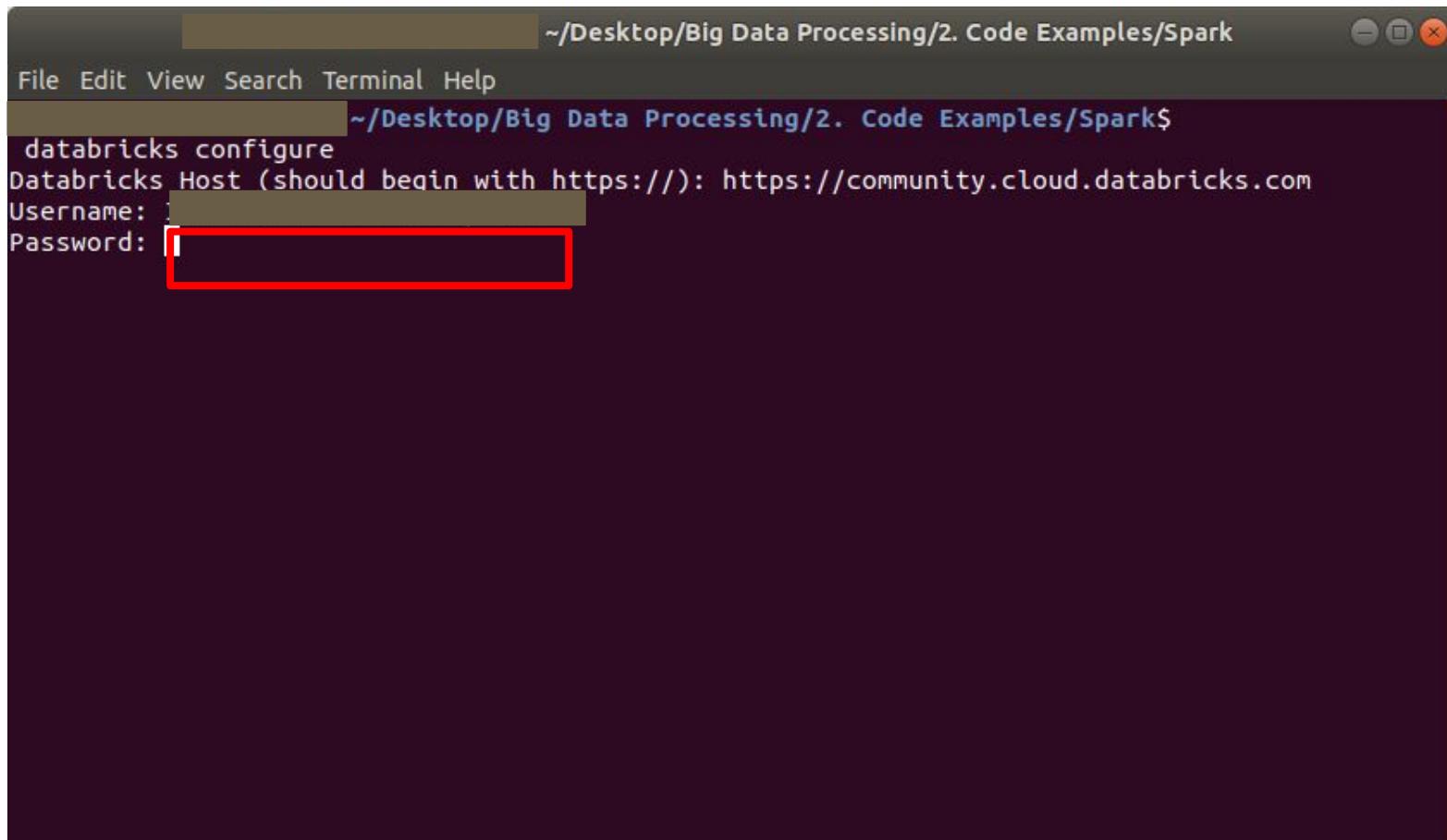
# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - b. We must indicate our username (same as when we sign in at <https://community.cloud.databricks.com>).



# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - c. We must indicate our password (same as when we sign in at <https://community.cloud.databricks.com>).

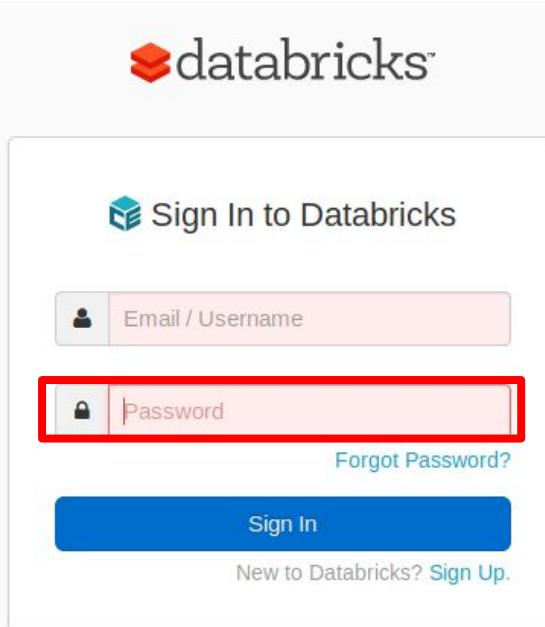


A screenshot of a terminal window titled '~/Desktop/Big Data Processing/2. Code Examples/Spark'. The window has a dark background and a light-colored title bar. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The terminal prompt is ' ~/Desktop/Big Data Processing/2. Code Examples/Spark\$'. The command 'databricks configure' is entered, followed by prompts for 'Databricks Host (should begin with https://): https://community.cloud.databricks.com', 'Username:', and 'Password:'. The 'Password:' field is highlighted with a red rectangular border.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark$  
databricks configure  
Databricks Host (should begin with https://): https://community.cloud.databricks.com  
Username:  
Password:
```

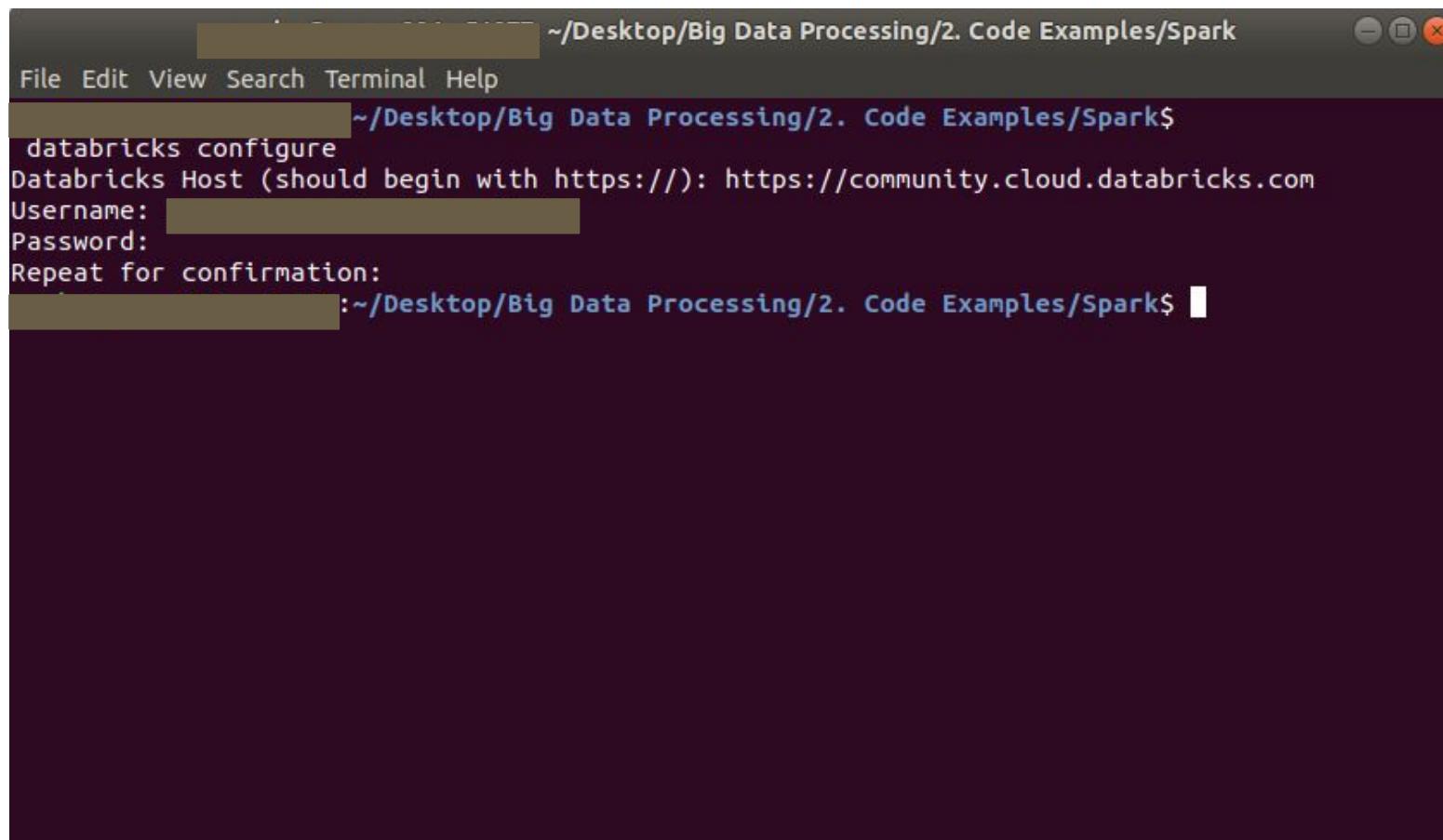
# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - c. We must indicate our password (same as when we sign in at <https://community.cloud.databricks.com>).



# Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
  - c. We must indicate our password (same as when we sign in at <https://community.cloud.databricks.com>).



The screenshot shows a terminal window with a dark background and light-colored text. The title bar reads: ~/Desktop/Big Data Processing/2. Code Examples/Spark. The menu bar includes: File, Edit, View, Search, Terminal, Help. The command entered is: `databricks configure`. The terminal then prompts for the Databricks Host: `Databricks Host (should begin with https://): https://community.cloud.databricks.com`. It then asks for the Username and Password. Finally, it asks for confirmation: `Repeat for confirmation:`. The command prompt at the bottom is: `:~/Desktop/Big Data Processing/2. Code Examples/Spark$`.

# Databricks: An Online Platform for Data Engineers

*How to...*

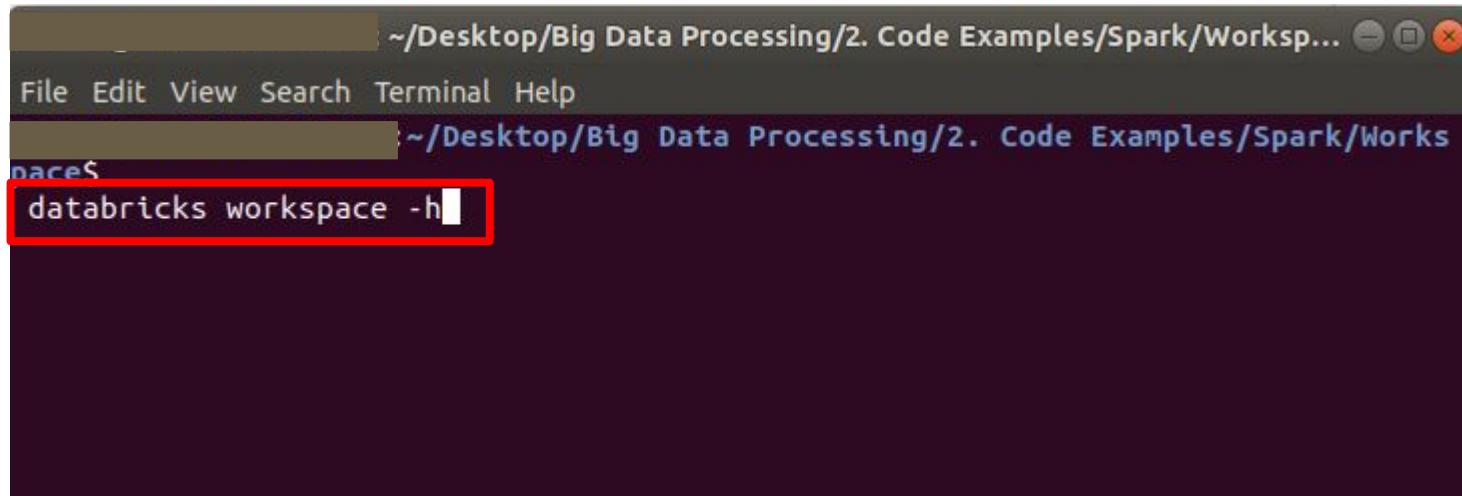
Upload an entire directory  
from our local machine  
to our Databricks Workspace

# Databricks: An Online Platform for Data Engineers

7. The Databricks CLI has some **workspace** specific commands:

# Databricks: An Online Platform for Data

7. The Databricks CLI has some **workspace** specific commands:



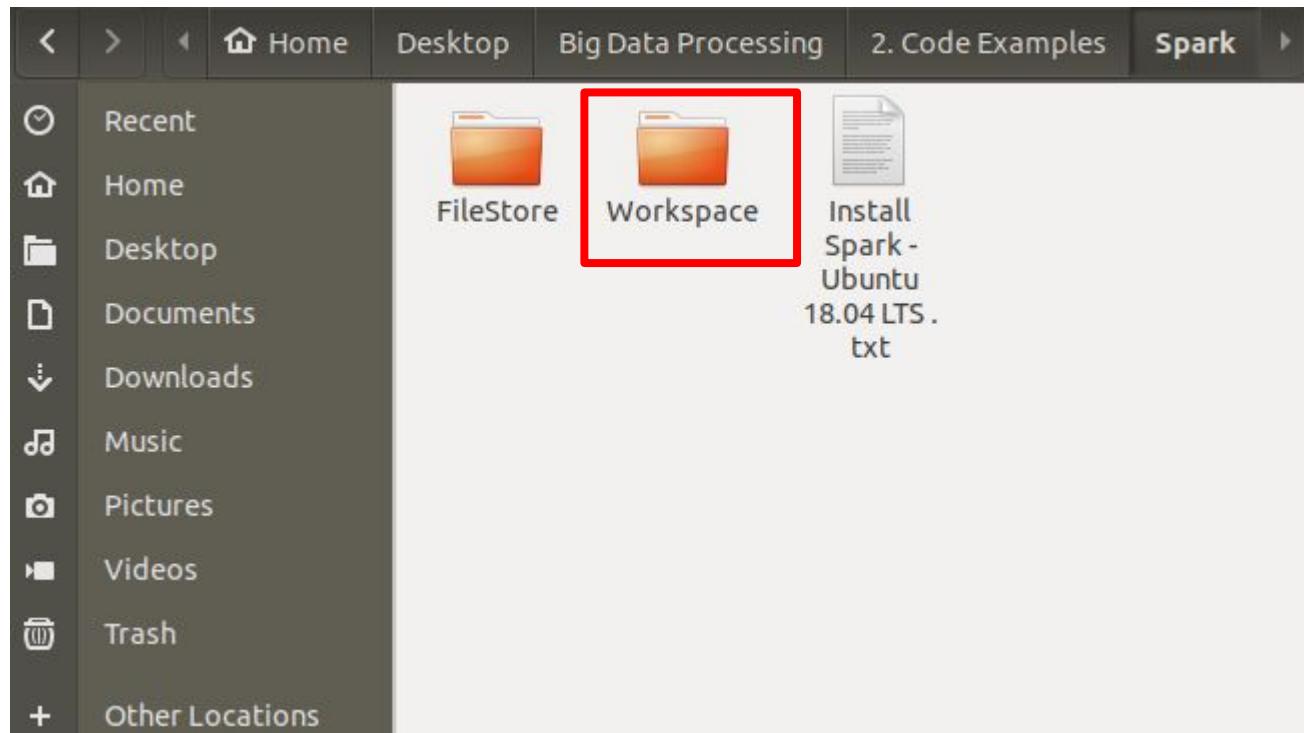
A screenshot of a terminal window titled ' ~/Desktop/Big Data Processing/2. Code Examples/Spark/Worksp...'. The window has a dark background and light-colored text. The menu bar includes 'File', 'Edit', 'View', 'Search', 'Terminal', and 'Help'. The current directory is shown in the title bar as ' ~/Desktop/Big Data Processing/2. Code Examples/Spark/Works...'. The command 'databricks workspace -h' is typed in the terminal, and it is highlighted with a red rectangle. The terminal is currently empty, showing only the command prompt.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark/Worksp... ◻ ◻ ◻
File Edit View Search Terminal Help
:~/Desktop/Big Data Processing/2. Code Examples/Spark/Works
pace$ databricks workspace -h
Usage: databricks workspace [OPTIONS] COMMAND [ARGS]...
Utility to interact with the Databricks workspace. Workspace paths must be
absolute and be prefixed with `/'.
Options:
-v, --version    0.9.0
--debug          Debug Mode. Shows full stack trace on error.
--profile TEXT  CLI connection profile to use. The default profile is
                  "DEFAULT".
-h, --help        Show this message and exit.

Commands:
delete          Deletes objects from the Databricks workspace. rm and delete are
                  synonyms.
export          Exports a file from the Databricks workspace.
export_dir       Recursively exports a directory from the Databricks workspace.
import          Imports a file from local to the Databricks workspace.
import_dir      Recursively imports a directory to the Databricks workspace. import_dir
list            List objects in the Databricks workspace. ls and list are
                  synonyms.
ls              List objects in the Databricks Workspace. ls and list are
                  synonyms.
mkdirs          Make directories in the Databricks Workspace.
rm              Deletes objects from the Databricks workspace. rm and delete are
                  synonyms.
pace$
```

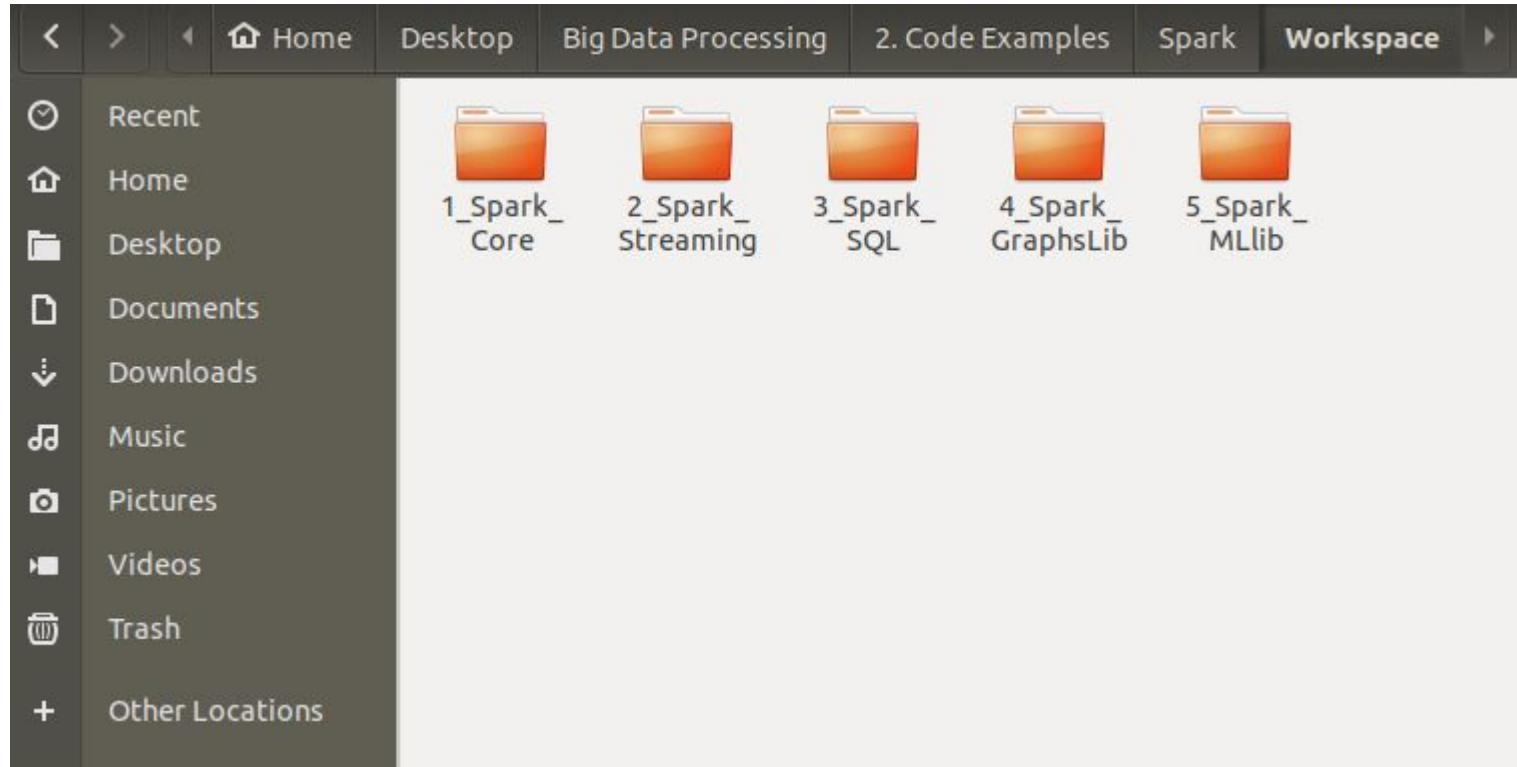
# Databricks: An Online Platform for Data Engineers

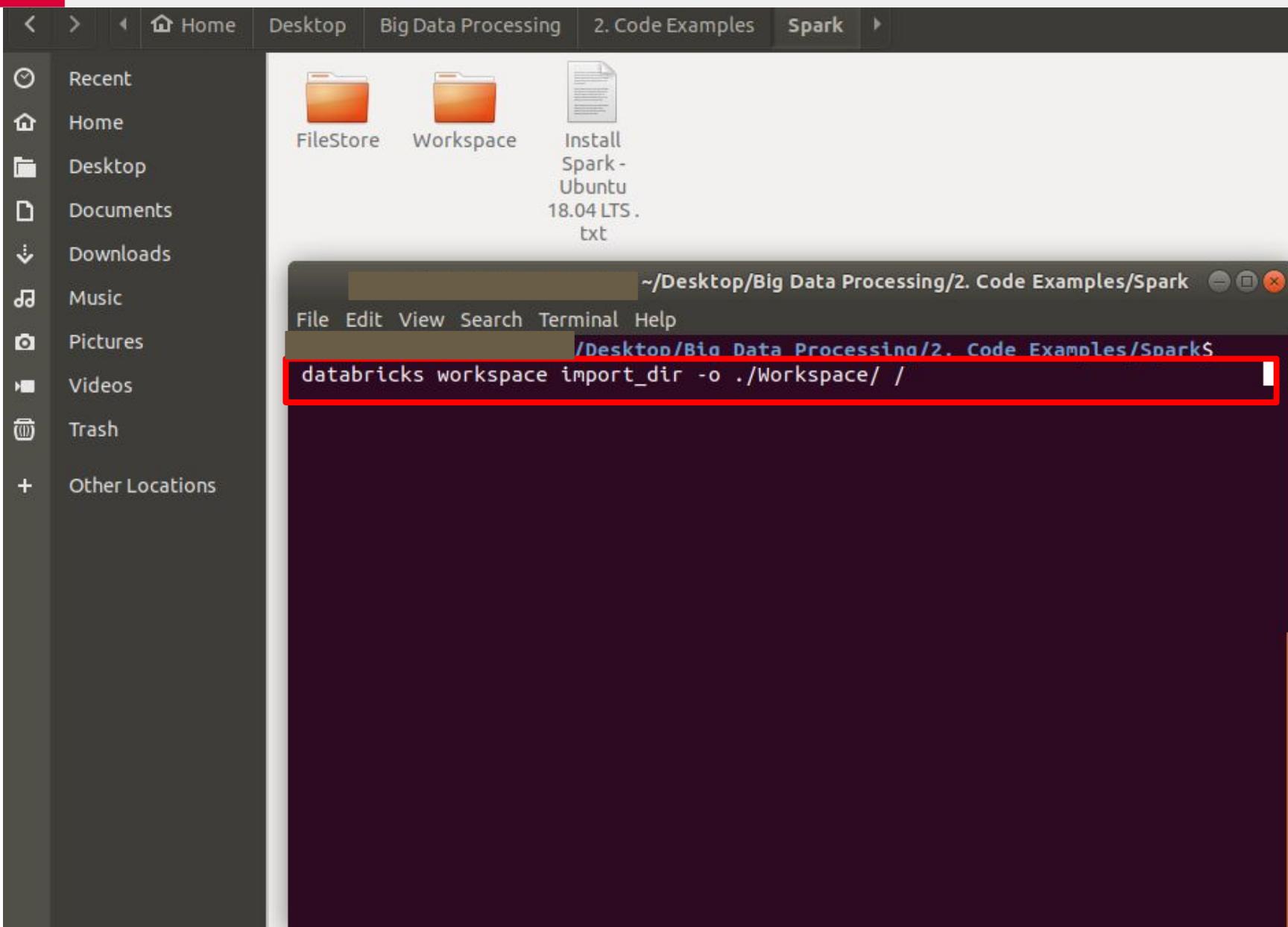
7. We can use the command **import\_dir** command to upload our set of Spark code examples (available in Canvas) to our Databricks Workspace.



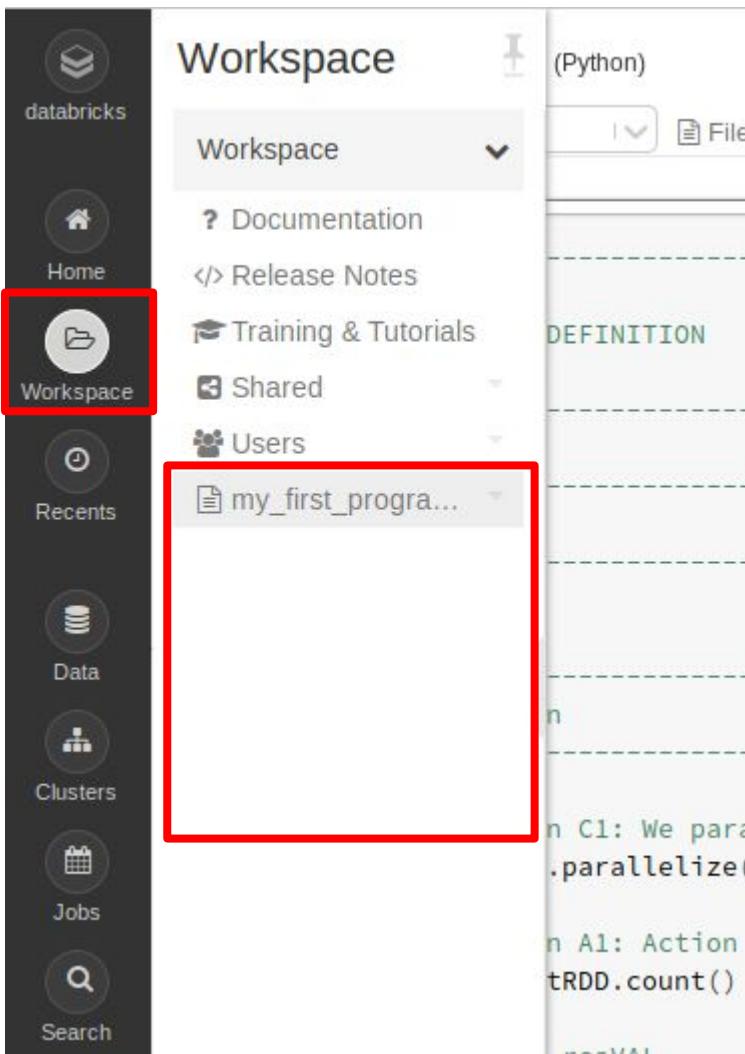
# Databricks: An Online Platform for Data Engineers

7. We can use the command **import\_dir** command to upload our set of Spark code examples (available in Canvas) to our Databricks Workspace.





# Databricks: An Online Platform for Data Engineering

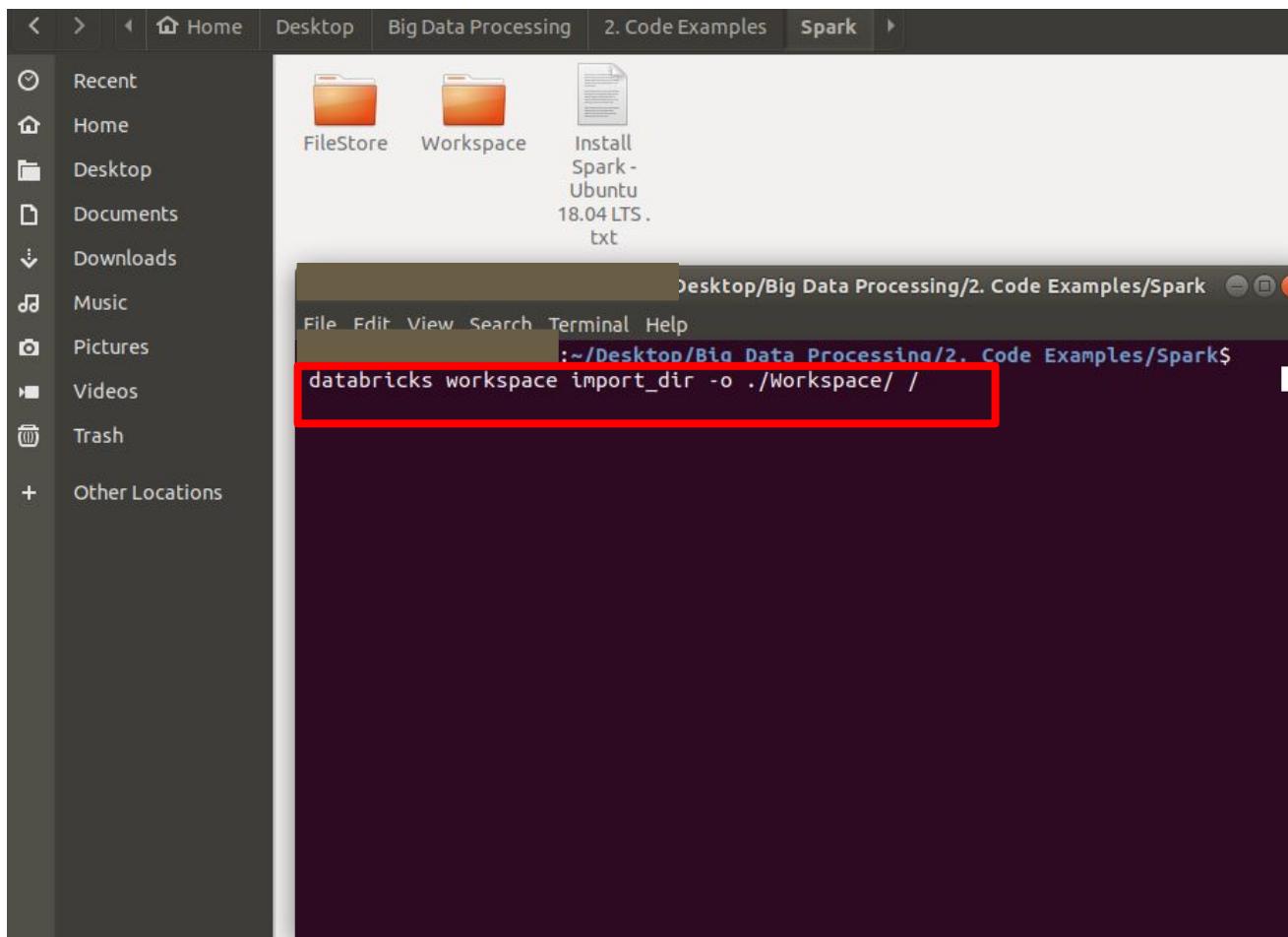


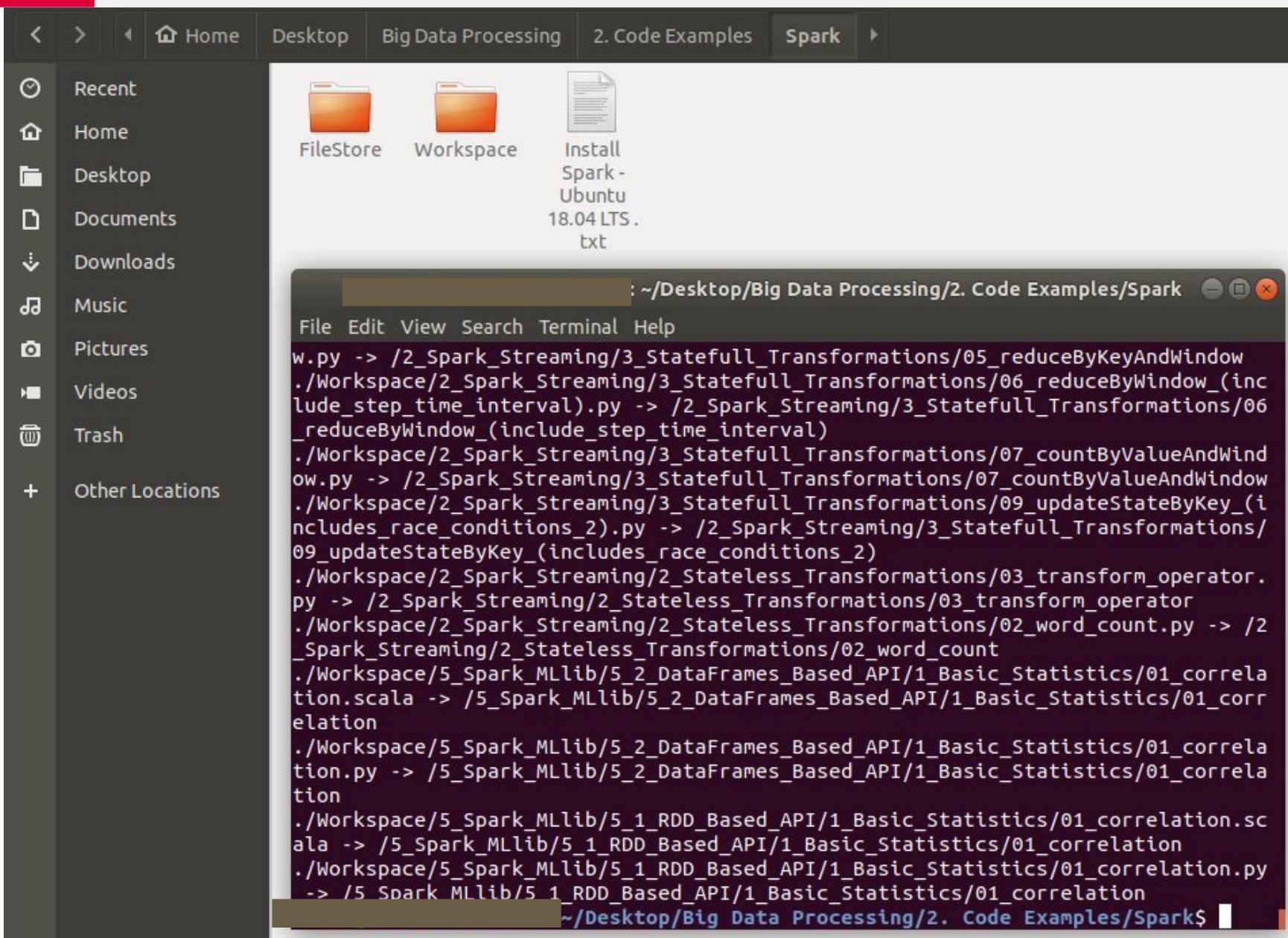
As we can see, before running the command, our **Databricks Workspace** only has the Python program we created before.

# Databricks: An Online Platform for Data Engineering

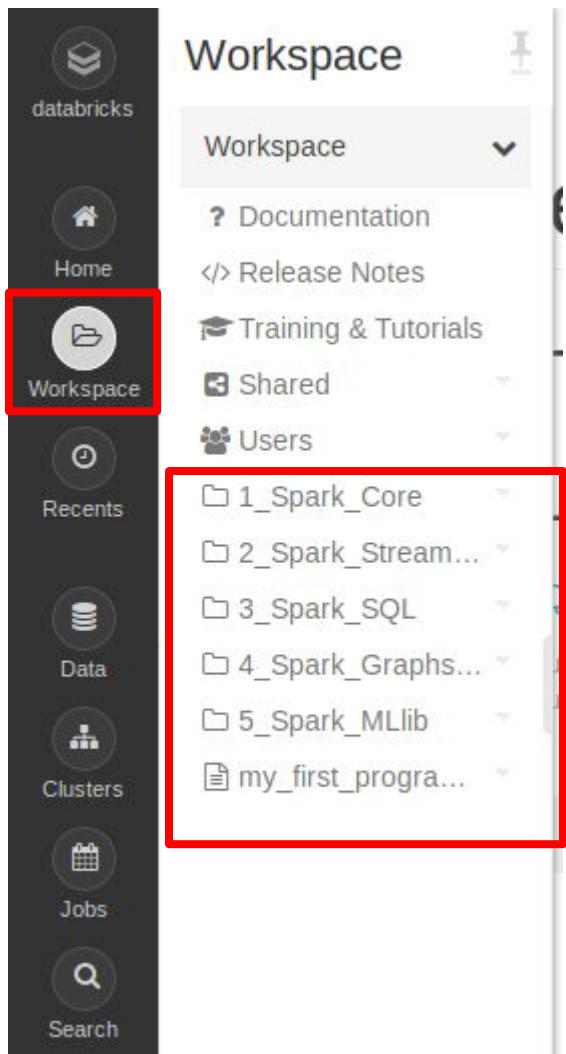
7. Running the command will take a couple of minutes.

```
> databricks workspace import_dir -o ./Workspace/ /
```





# Databricks: An Online Platform for Data Engineering



As we can see, after running the command, our **Databricks Workspace** contains our set of Spark code examples that we will use during the semester.

# Databricks: An Online Platform for Data Engineers

*How to...*  
Upload a Dataset  
from our local machine  
to the Databricks File System (DBFS).

# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



## Welcome to databricks™



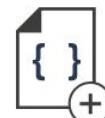
[Explore the Quickstart Tutorial](#)

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](#)

[Create Table](#)

[New Cluster](#)

### Recents

34\_job\_inspection\_5.scala

33\_job\_inspection\_4.scala

32\_job\_inspection\_3.scala

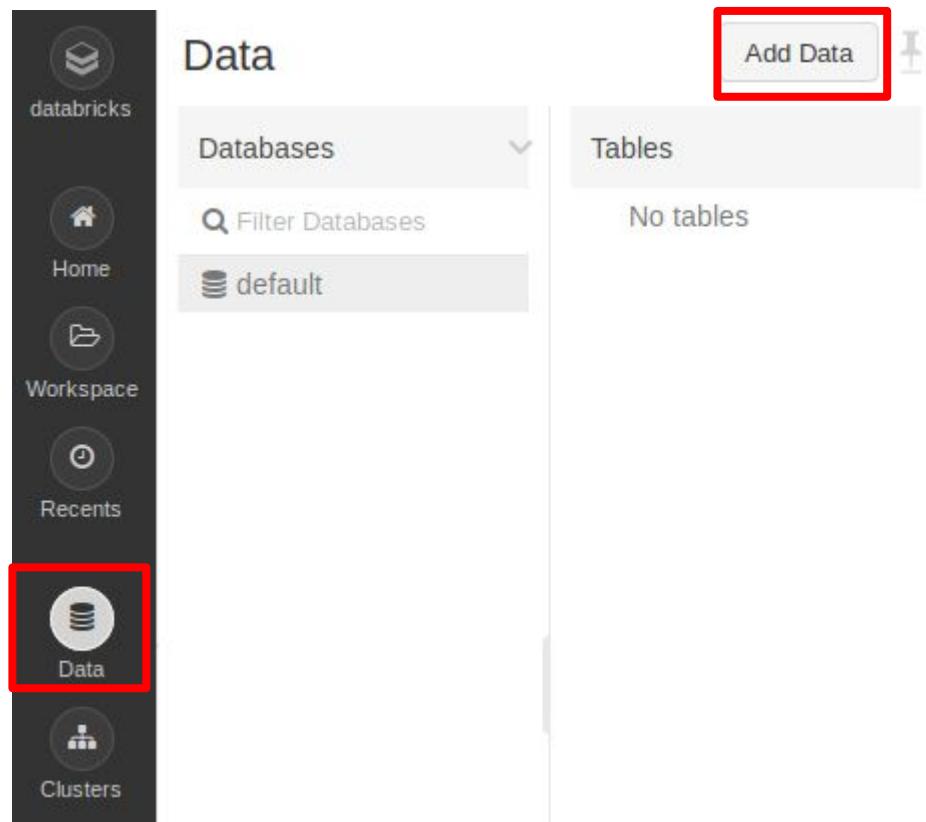
### What's new in v3.2

- Instance Pools
- Databricks Runtime 6.0 will drop Python 2 support

[View latest release notes](#)

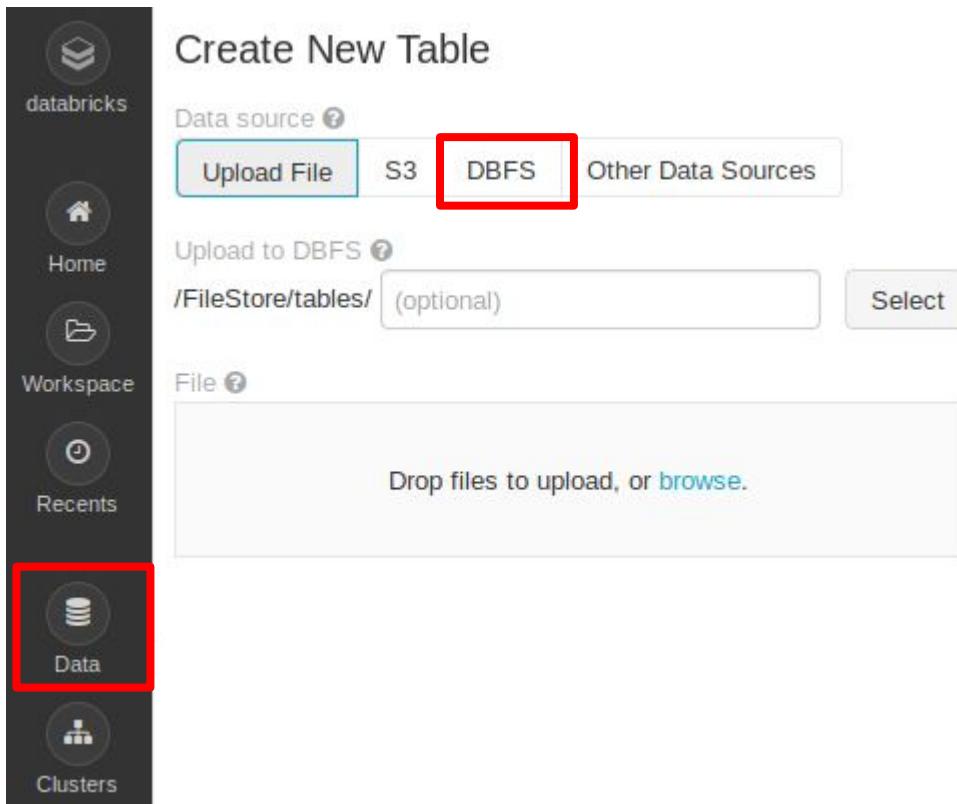
# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



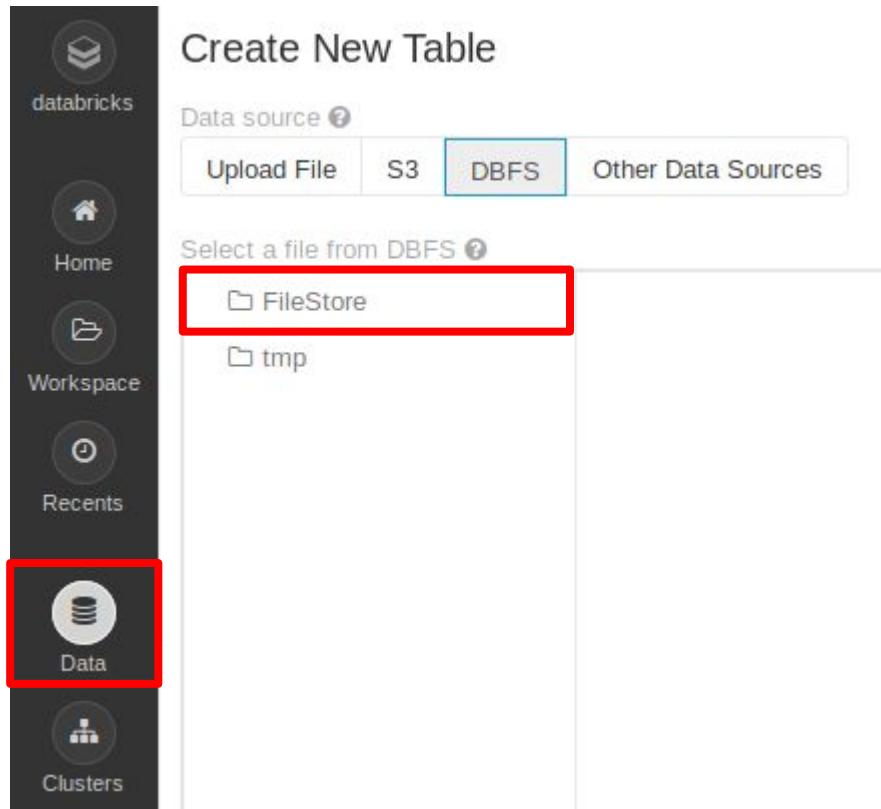
# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



Any **Dataset** being used during the semester is to be placed at **dbfs:/FileStore/tables/**

# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.

The screenshot shows the Databricks Data interface. On the left is a sidebar with icons for Home, Workspace, Recents, Data (which is selected and highlighted with a red box), and Clusters. The main area is titled "Create New Table" and shows the "Data source" section with tabs for Upload File, S3, DBFS (which is selected and highlighted with a blue box), and Other Data Sources. Below this, a "Select a file from DBFS" section shows a tree view of the DBFS structure. The root directory "FileStore" is highlighted with a red box. Inside "FileStore", there are sub-directories "jars" and "tables", also highlighted with a red box. The "tmp" directory is shown below "FileStore".

Any **Dataset** being used during the semester is to be placed at  
**dbfs:/FileStore/tables/**

# Databricks: An Online Platform for Data Engineers

8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.

The screenshot shows the Databricks interface with the 'Data' tab selected in the sidebar. The main area is titled 'Create New Table' and displays the 'Data source' section. The 'DBFS' tab is selected, highlighted with a blue border. Below it, a dropdown menu titled 'Select a file from DBFS' shows a tree view of the DBFS structure. The root directory 'FileStore' is selected, indicated by a red box. Inside 'FileStore', there are two sub-directories: 'jars' and 'tables'. A large red box encloses the entire right-hand pane where the file tree is displayed.

Any **Dataset** being used during the semester is to be placed at  
**dbfs:/FileStore/tables/**

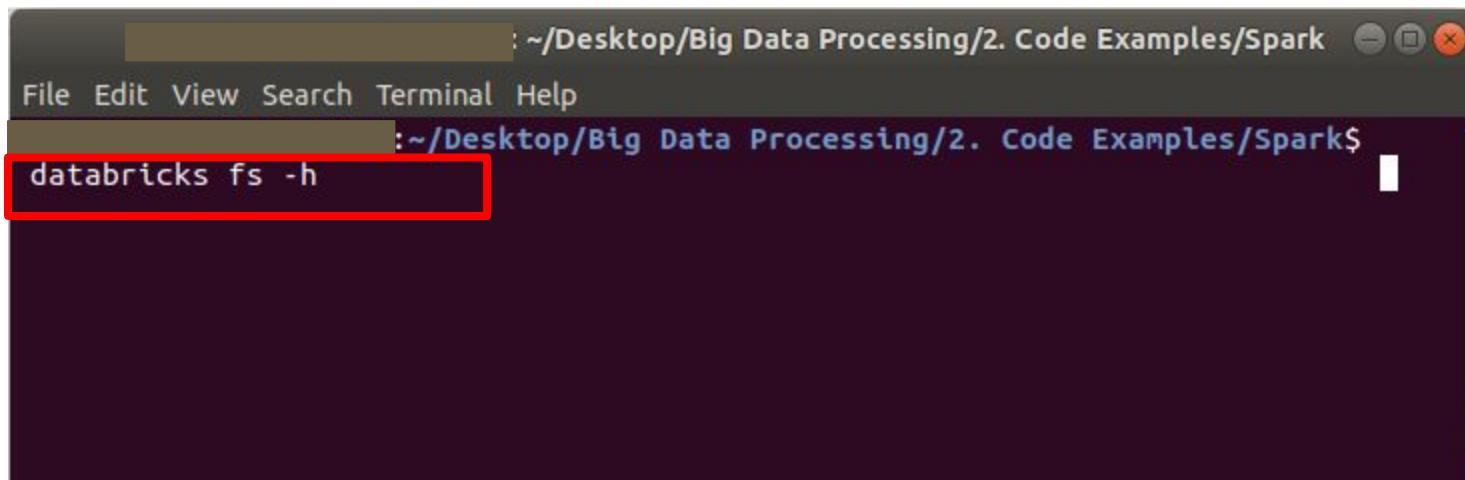
As we can see by the moment there is no Dataset store at DBFS.

# Databricks: An Online Platform for Data Engineers

8. The CLI has some **Databricks File System (DBFS)** specific commands:

# Databricks: An Online Platform for Data Engineers

8. The CLI has some **Databricks File System (DBFS)** specific commands:



A screenshot of a terminal window titled 'Terminal'. The window shows the path '/Desktop/Big Data Processing/2. Code Examples/Spark'. The command 'databricks fs -h' is entered at the prompt. The entire command line is highlighted with a red rectangular box.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark
```



```
File Edit View Search Terminal Help
```

```
~/Desktop/Big Data Processing/2. Code Examples/Spark$
```

```
databricks fs -h
```

```
Usage: databricks fs [OPTIONS] COMMAND [ARGS]...
```

```
Utility to interact with DBFS.
```

```
DBFS paths are all prefixed with dbfs:. Local paths can be absolute or  
local.
```

```
Options:
```

```
-v, --version    0.9.0  
--debug         Debug Mode. Shows full stack trace on error.  
--profile TEXT  CLI connection profile to use. The default profile is  
                  "DEFAULT".  
-h, --help       Show this message and exit.
```

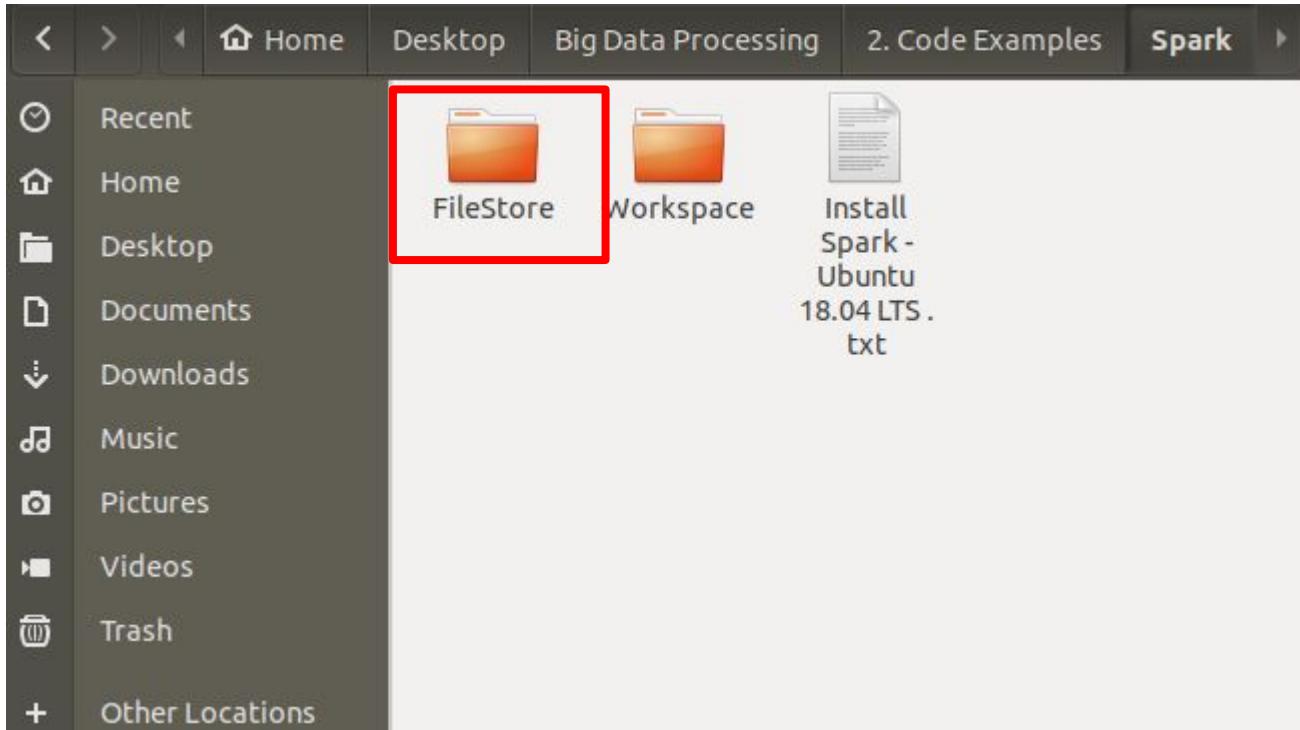
```
Commands:
```

```
cat      Show the contents of a file.  
configure Configures host and authentication info for the CLI.  
cp      Copy files to and from DBFS.  
ls       List files in DBFS.  
mkdirs   Make directories in DBFS.  
mv       Moves a file between two DBFS paths.  
rm       Remove files from dbfs.
```

```
~/Desktop/Big Data Processing/2. Code Examples/Spark$
```

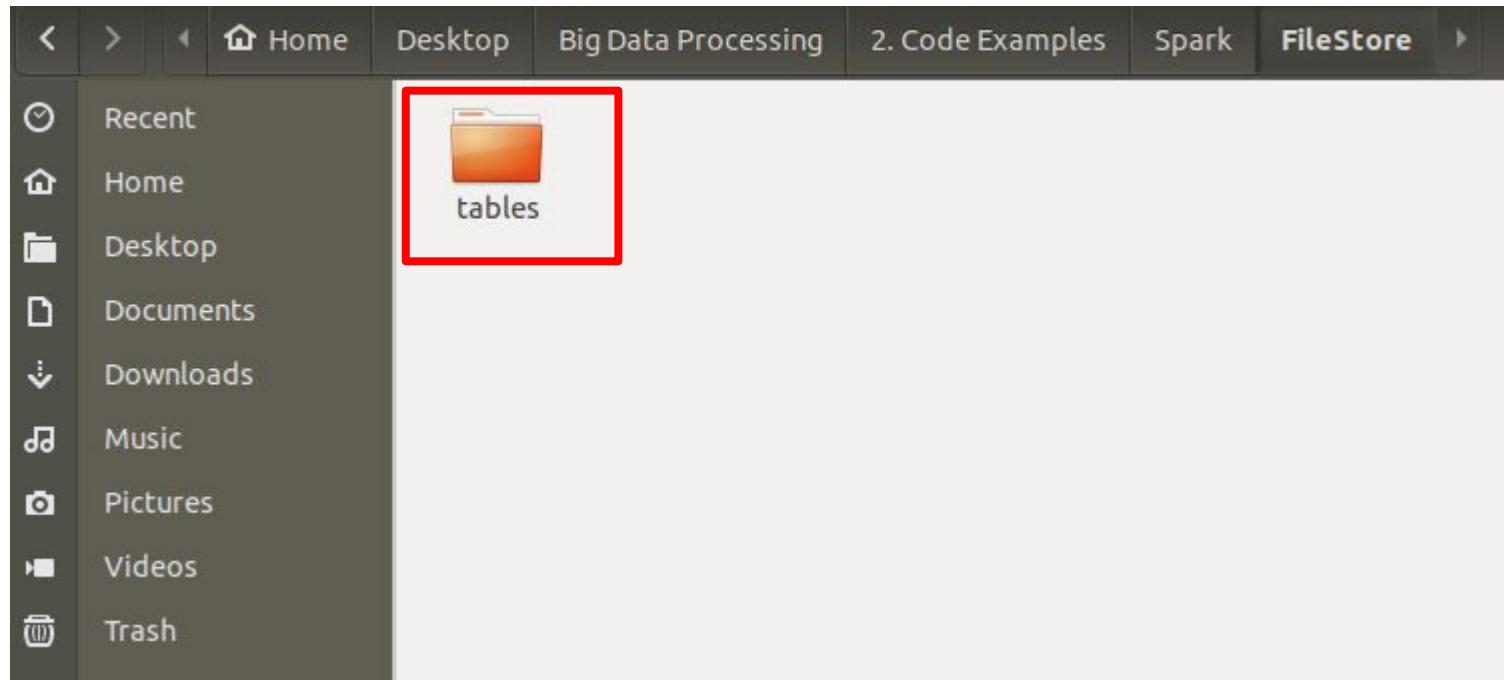
# Databricks: An Online Platform for Data Engineers

8. We can use the command **cp** to upload our set of Datasets to DBFS.



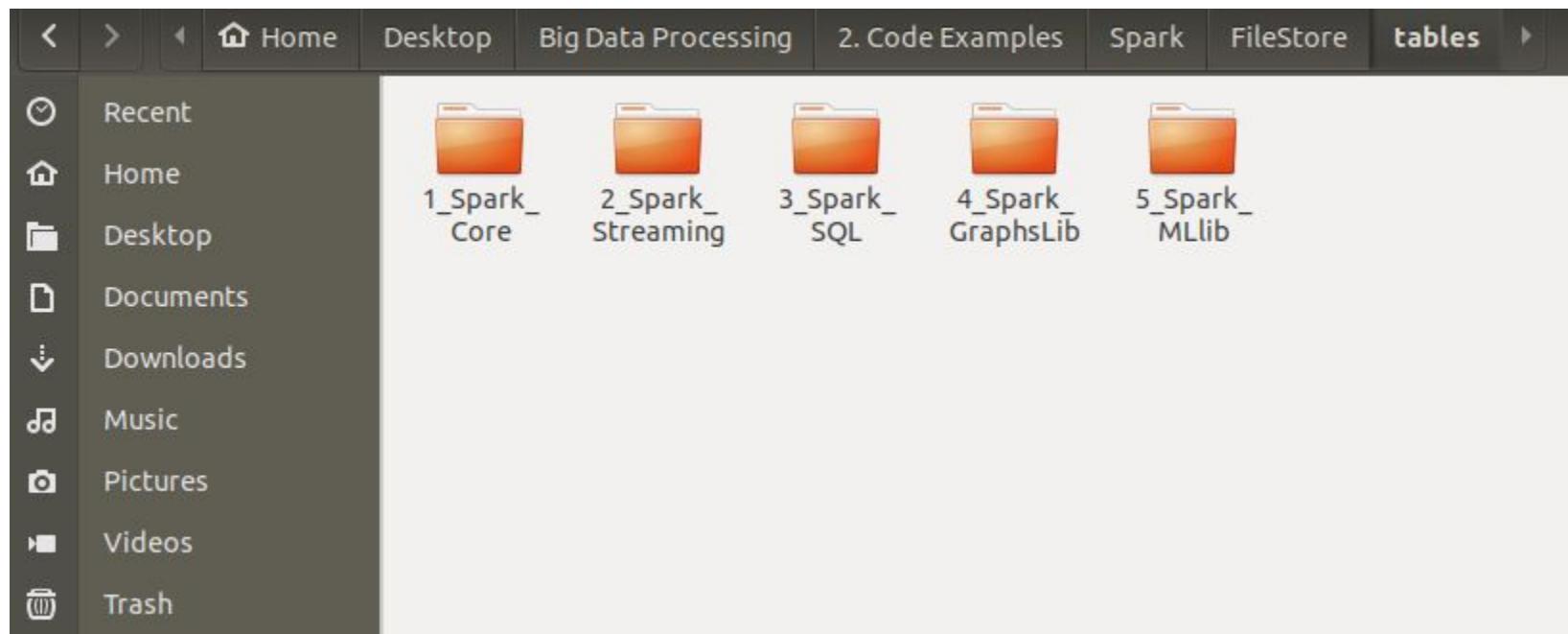
# Databricks: An Online Platform for Data Engineers

8. We can use the command **cp** to upload our set of Datasets to DBFS.



# Databricks: An Online Platform for Data Engineers

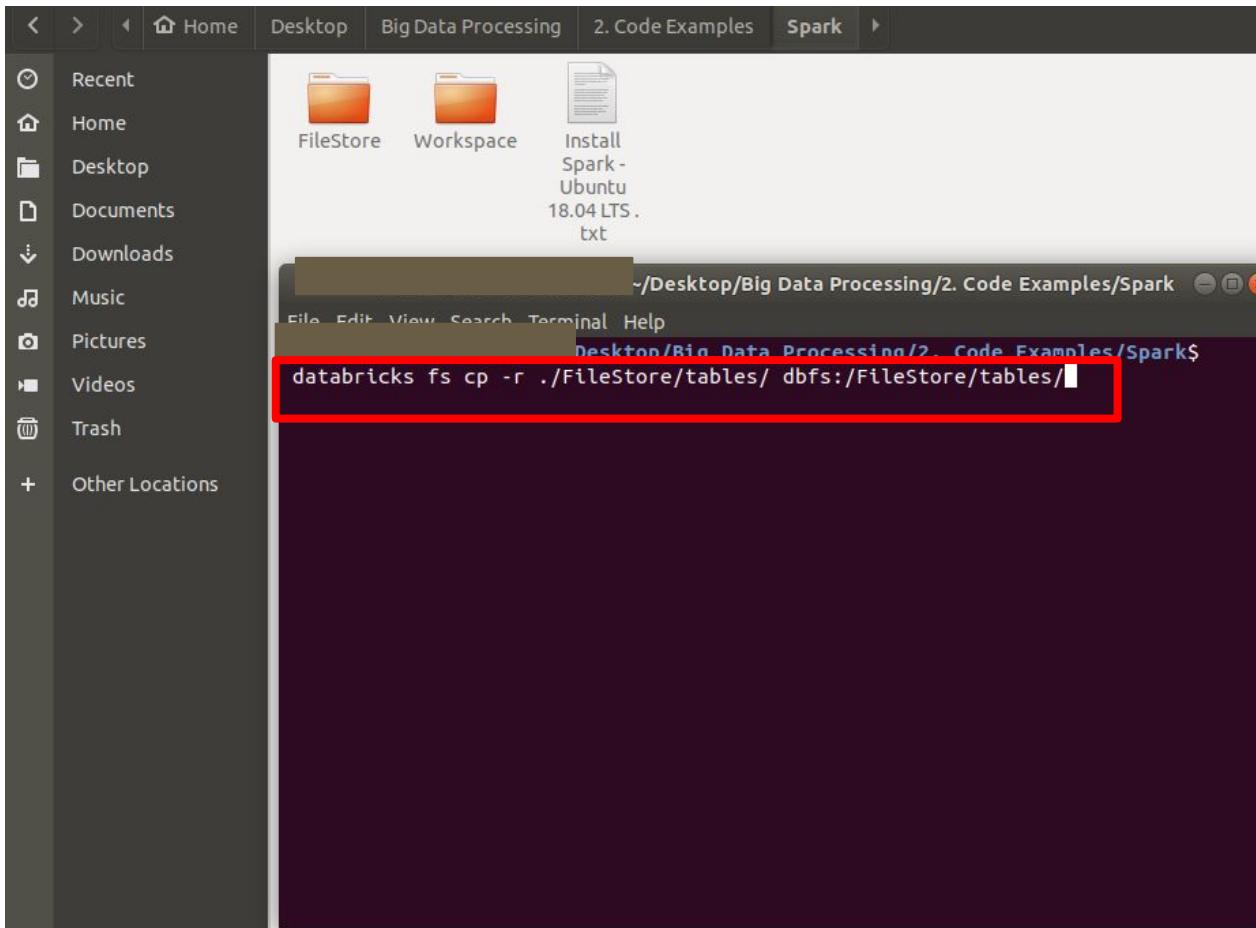
8. We can use the command **cp** to upload our set of Datasets to DBFS.



# Databricks: An Online Platform for Data Engin

8. Running the command will take a couple of minutes.

```
> databricks fs cp -r ./FileStore/tables dbfs:/FileStore/tables/
```



# Databricks: An Online Platform for Data Engin

8. Running the command will take a couple of minutes.

```
> databricks fs cp -r ./FileStore/tables dbfs:/FileStore/tables/
```

The screenshot shows a desktop environment with a file manager window open on the left and a terminal window open on the right. The file manager window displays a sidebar with links to Recent, Home, Desktop, Documents, Downloads, Music, Pictures, Videos, and Trash. It also shows icons for FileStore and Workspace. A file named 'Install Spark - Ubuntu 18.04 LTS.txt' is visible in the main pane. The terminal window has a title bar reading '~Desktop/Big Data Processing/2. Code Examples/Spark'. The terminal itself contains a series of commands being run, all of which are being redirected from local paths to Databricks File System (dbfs) paths. The commands involve copying files from 'FileStore/tables' to 'dbfs:/FileStore/tables'. The terminal prompt at the bottom is ':~/Desktop/Big Data Processing/2. Code Examples/Spark\$'.

```
./FileStore/tables/2_Spark_Streaming/merge_solutions.py -> dbfs:/FileStore/tables/2_Spark_Streaming/merge_solutions.py  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_03.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_03.txt  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_02.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_02.txt  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_05.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_05.txt  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_04.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_04.txt  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_01.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_01.txt  
./FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_06.txt -> dbfs:/FileStore/tables/2_Spark_Streaming/my_dataset/my_tiny_file_06.txt  
./FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/pearson_3_vars_dataset.csv -> dbfs:/FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/pearson_3_vars_dataset.csv  
./FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/pearson_2_vars_dataset.csv -> dbfs:/FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/pearson_2_vars_dataset.csv  
./FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/spearman_2_vars_dataset.csv -> dbfs:/FileStore/tables/5_Spark_MLLib/1_Basic_Statistics/spearman_2_vars_dataset.csv
```

# Databricks: An Online Platform for Data Engineering

The screenshot shows the Databricks interface for creating a new table. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data (which is highlighted with a red box), and Clusters. The main area is titled 'Create New Table' and has a 'Data source' section with tabs for Upload File, S3, DBFS (which is selected and highlighted with a blue box), and Other Data Sources. Below this, it says 'Select a file from DBFS' and shows a tree view of the DBFS structure. The 'FileStore' folder is expanded, showing 'jars' and 'tables'. The 'tables' folder is also expanded, showing sub-folders named '1\_Spark\_Core', '2\_Spark\_Streaming', '3\_Spark\_SQL', '4\_Spark\_GraphsLib', and '5\_Spark\_MLlib'. Red boxes highlight the 'DBFS' tab, the 'FileStore' folder, and the 'tables' folder.

As we can see, after running the command, our **Datasets** are now stored at `dbfs:/FileStore/tables/`

# Databricks: An Online Platform for Data Engineers

*How to...*

Run a program / Spark Application  
reading in a Dataset from DBFS  
and possibly storing the results at DBFS.

# Databricks: An Online Platform for Data Engineering

- As we can see, Spark Core contains **my\_dataset** of 4 text files:  
comedies.txt    histories.txt    poems.txt    tragedies.txt

The screenshot shows the Databricks Data interface. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data (which is selected and highlighted with a red box), and Clusters. The main area is titled "Create New Table" and shows a "Data source" section with tabs for Upload File, S3, DBFS (which is selected and highlighted with a blue border), and Other Data Sources. Below this, a "Select a file from DBFS" section shows a tree view of files under "DBFS". The tree includes "jars" and "tables" (which is selected and highlighted with a grey background). Under "tables", there are five items: "1\_Spark\_Core", "2\_Spark\_Streaming", "3\_Spark\_SQL", "4\_Spark\_GraphsLib", and "5\_Spark\_MLib". To the right of these, there are two more sections: "my\_result\_merge.py" and "my\_result". Under "my\_result", there are four files: "comedies.txt", "histories.txt", "poems.txt", and "tragedies.txt". The "my\_dataset" folder and its contents are highlighted with red boxes.

| Category  | File/Folder       |
|-----------|-------------------|
| Tables    | 1_Spark_Core      |
|           | 2_Spark_Streaming |
|           | 3_Spark_SQL       |
|           | 4_Spark_GraphsLib |
|           | 5_Spark_MLib      |
|           | my_dataset        |
| my_result | comedies.txt      |
|           | histories.txt     |
|           | poems.txt         |
|           | tragedies.txt     |

# Databricks: An Online Platform for Data Engineering

- As we can see, Spark Core contains the folder **my\_result**, being empty.

Create New Table

Data source

Upload File S3 DBFS Other Data Sources

Select a file from DBFS

jars

tables

1\_Spark\_Core

2\_Spark\_Streaming

3\_Spark\_SQL

4\_Spark\_GraphsLib

5\_Spark\_MLLib

my\_result\_merge.py

my\_dataset

my\_result

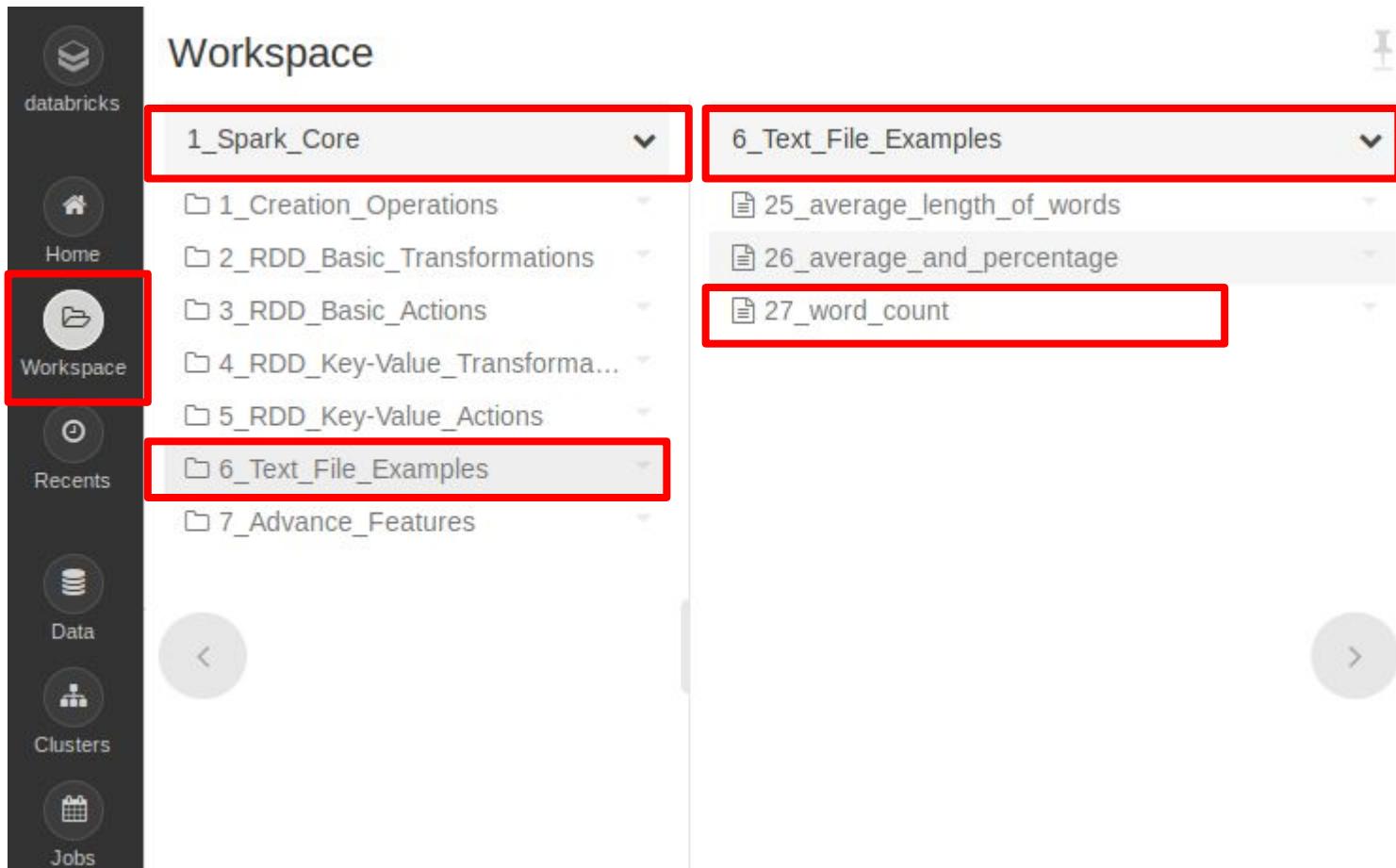
Data

Clusters

# Databricks: An Online Platform for Data Engineering

- Without explaining the program itself, our code example:

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)



# Databricks: An Online Platform for Data Engin

- Without explaining the program itself, our code example

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)

- Reads the **Dataset** from  
**dbfs:/FileStore/1\_Spark\_Core/my\_dataset**

# Databricks: An Online Platform for Data Engin

- Without explaining the program itself, our code example

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)

- Reads the **Dataset** from  
**dbfs:/FileStore/1\_Spark\_Core/my\_dataset**
- Computes the word count of the words appearing in the files of  
my\_dataset: comedies.txt, histories.txt, poems.txt, tragedies.txt

# Databricks: An Online Platform for Data Engin

- Without explaining the program itself, our code example

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)

- Reads the **Dataset** from  
**dbfs:/FileStore/1\_Spark\_Core/my\_dataset**
- Computes the word count of the words appearing in the files of  
my\_dataset: **comedies.txt, histories.txt, poems.txt, tragedies.txt**
- Writes the results to new files placed in the directory  
**dbfs:/FileStore/1\_Spark\_Core/my\_result**

# Databricks: An Online Platform for Data Engineering

- Without explaining the program itself, our code example

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)

```
27_word_count (Python)
MyCluster File View: Code Permissions Run All
194 # (i) Specifies the function F to be executed.
195 # (ii) Define any input parameter such this function F has to be called with
196 #
197 #
198 if __name__ == '__main__':
199     # 1. We use as many input arguments as needed
200     pass
201
202     # 2. Local or Databricks
203     local_False_databricks_True = True
204
205     # 3. We set the path to my_dataset and my_result
206     my_local_path =
207     my_databricks_path = "/"
208
209     my_dataset_dir = "FileStore/tables/1_Spark_Core/my_dataset/"
210     my_result_dir = "FileStore/tables/1_Spark_Core/my_result"
211
```

# Databricks: An Online Platform for Data Engineering

9. Running the program / Spark application leads to no visual results:

[Workspace/1\\_Spark\\_Core/6\\_Text\\_File\\_Examples/27\\_word\\_count.py](#)

27\_word\_count (Python)

MyCluster

File View: Code Permissions Run All Clear

```
1 | else:
2 |     dbutils.fs.rm(my_result_dir, True)
3 |
4 | # 5. We configure the Spark Context
5 | sc = pyspark.SparkContext.getOrCreate()
6 | sc.setLogLevel('WARN')
7 | print("\n\n\n")
8 |
9 |
10| # 6. We call to our main function
11| my_main(sc, my_dataset_dir, my_result_dir)
```

▶ (3) Spark Jobs

Command took 11.98 seconds -- by Ignacio.Castineiras@cit.ie at 9/27/2019, 6:00:52 PM on MyCluster

# Databricks: An Online Platform for Data Engineering

9. However, the folder **my\_result** in the DBFS has been populated:

The screenshot shows the Databricks Data browser interface. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data (which is highlighted with a red box), and Clusters. The main area is titled "Create New Table (Python)" and shows the "Data source" tab selected (DBFS). The "DBFS" tab is highlighted with a blue border. Below that, it says "Select a file from DBFS". The left pane shows a tree view with "jars" and "tables" under "tables". The right pane shows a list of files and folders: "1\_Spark\_Core", "2\_Spark\_Streaming", "3\_Spark\_SQL", "4\_Spark\_GraphsLib", "5\_Spark\_MLlib", "my\_result\_merge.py", "my\_dataset", "my\_result" (highlighted with a red box), and a folder containing "\_SUCCESS" and four "part-XXXX" files (highlighted with a red box).

| File/Folder        | Description                 |
|--------------------|-----------------------------|
| 1_Spark_Core       | Spark Core library          |
| 2_Spark_Streaming  | Spark Streaming library     |
| 3_Spark_SQL        | Spark SQL library           |
| 4_Spark_GraphsLib  | Spark GraphX library        |
| 5_Spark_MLlib      | Spark MLlib library         |
| my_result_merge.py | Script for merging results  |
| my_dataset         | Dataset file                |
| my_result          | Result folder (highlighted) |
| _SUCCESS           | Success marker file         |
| part-00000         | Partition 00000             |
| part-00001         | Partition 00001             |
| part-00002         | Partition 00002             |
| part-00003         | Partition 00003             |

# Databricks: An Online Platform for Data Engineering

9. Unfortunately we cannot visualise the content of these DBFS files.

The screenshot shows the Databricks Data interface. On the left, there's a sidebar with icons for Home, Workspace, Recents, Data (which is highlighted with a red box), and Clusters. The main area is titled "Create New Table (Python)". Under "Data source", the "DBFS" tab is selected (also highlighted with a red box). The "Select a file from DBFS" section shows a tree view of files. A red box highlights three specific items: "1\_Spark\_Core" under "tables", "my\_result" under "tables", and a folder containing "\_SUCCESS" and four "part-XXXX" files under "tables".

Create New Table (Python)

Data source

Upload File S3 DBFS Other Data Sources

Select a file from DBFS

jars

tables

1\_Spark\_Core

2\_Spark\_Streaming

3\_Spark\_SQL

4\_Spark\_GraphsLib

5\_Spark\_MLlib

my\_result\_merge.py

my\_dataset

my\_result

\_SUCCESS

part-00000

part-00001

part-00002

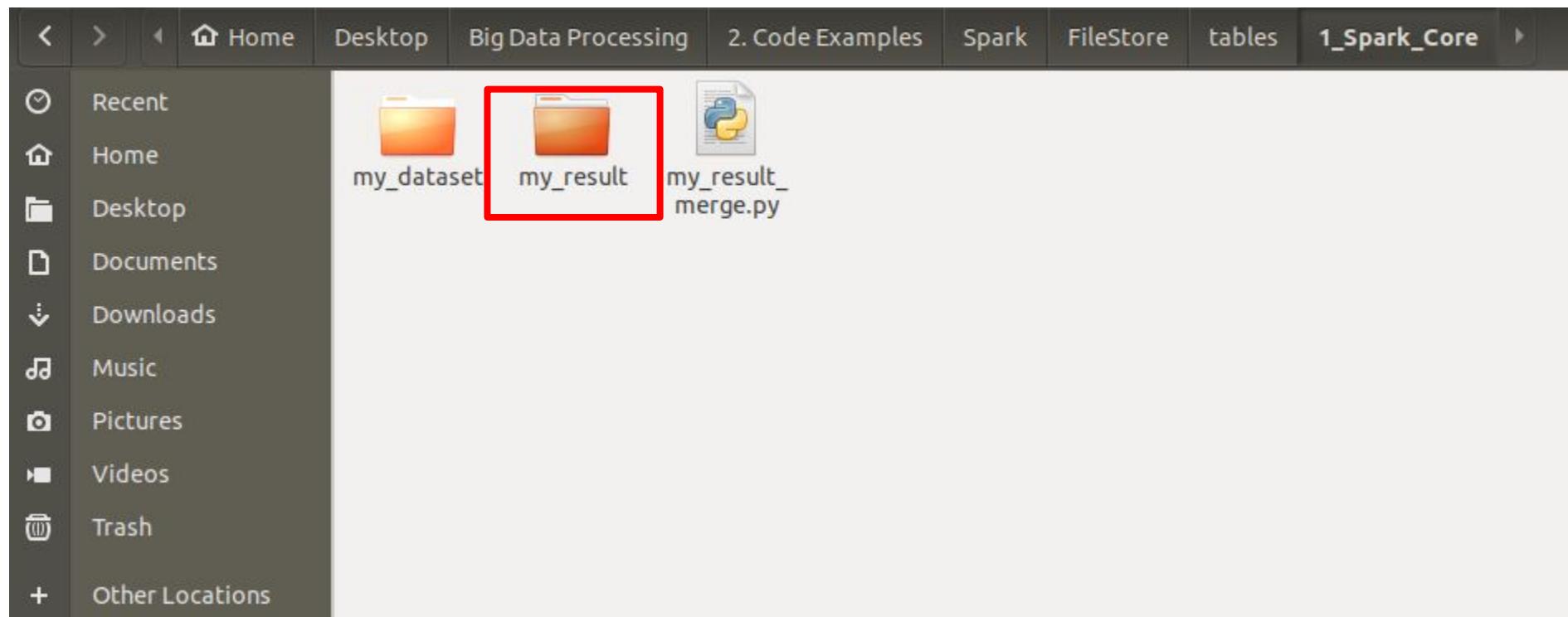
part-00003

# Databricks: An Online Platform for Data Engineering

9. But we can use the DBFS command cp to copy them to our local machine!

# Databricks: An Online Platform for Data Engineering

- As we can see, before running the command `cp` the local folder **my\_result** is empty.



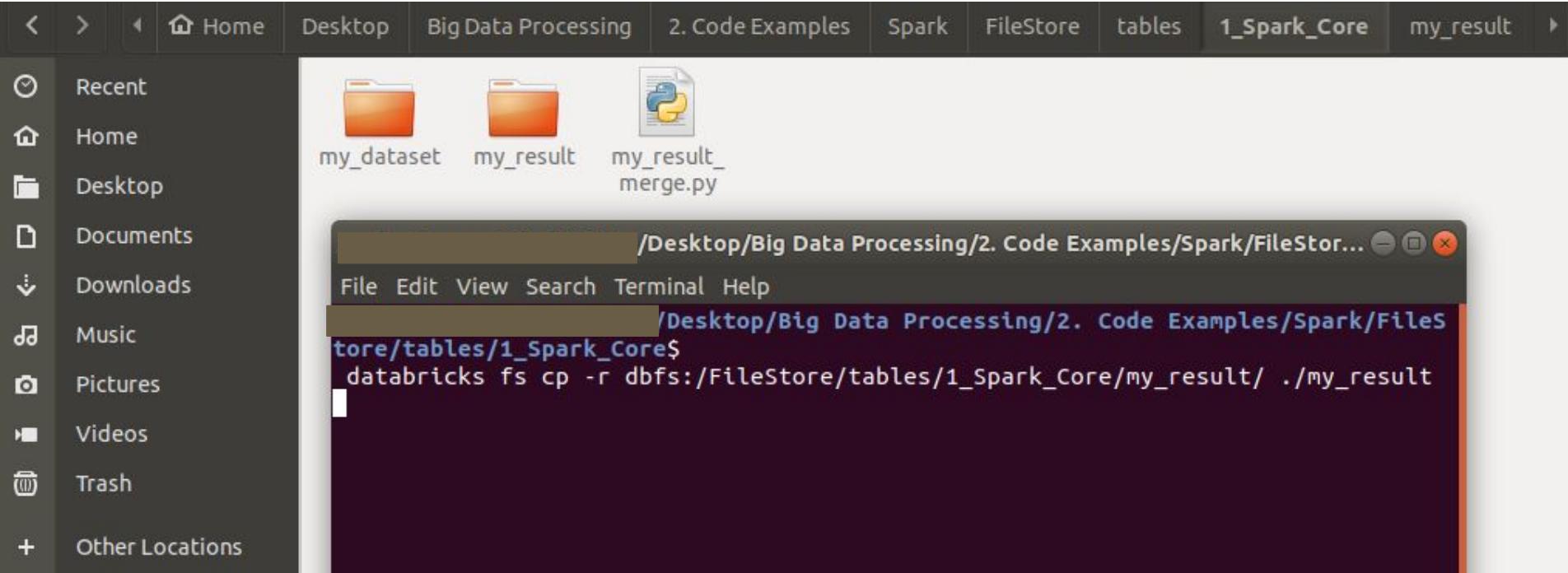
# Databricks: An Online Platform for Data Engineering

- As we can see, before running the command `cp` the local folder **my\_result** is empty.

The screenshot shows a file explorer window with a sidebar on the left containing icons for Recent, Home, Desktop, Documents, Downloads, Music, Pictures, Videos, Trash, and Other Locations. The main pane displays a folder icon with the text "Folder is Empty". The top navigation bar includes icons for back, forward, and search, followed by tabs for Home, Desktop, Big Data Processing, 2. Code Examples, Spark, FileStore, tables, 1\_Spark\_Core, and my\_result.

# Databricks: An Online Platform for Data Engineering

9. We run the command **cp** to copy the files.



# Databricks: An Online Platform for Data Engineering

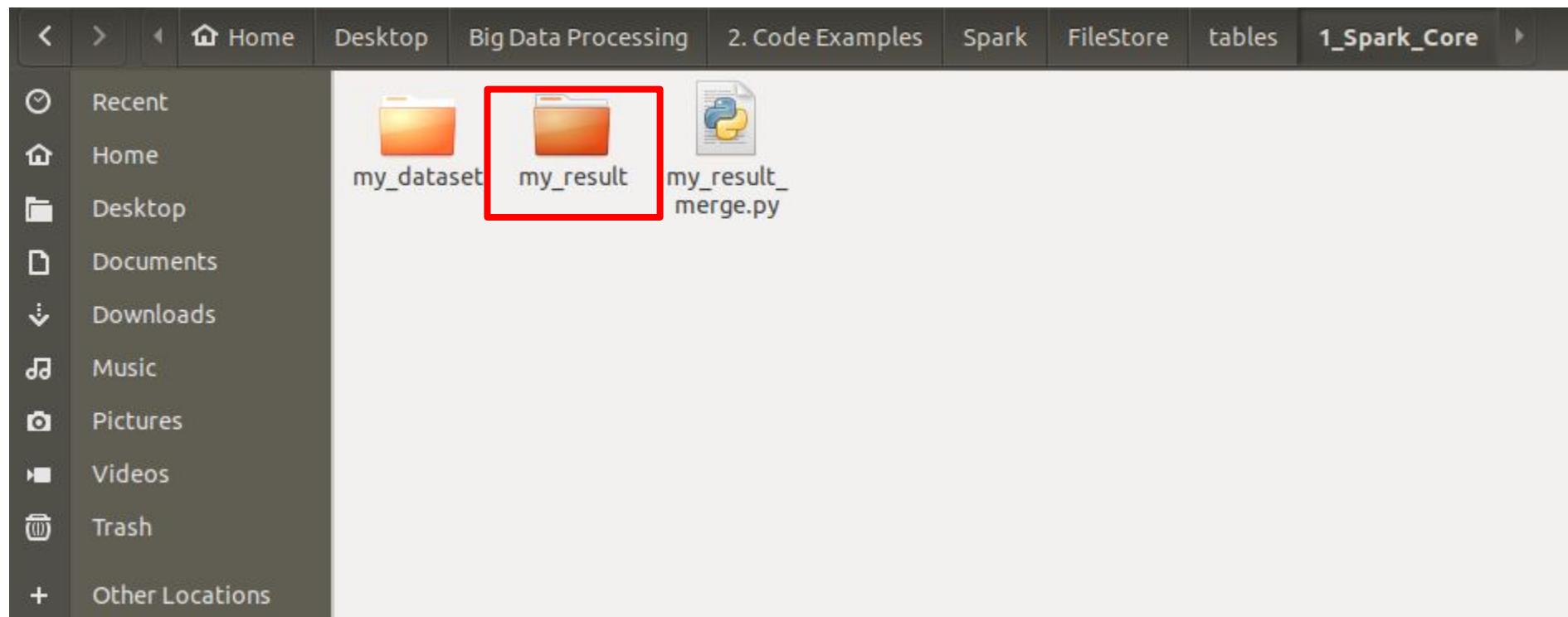
9. We run the command **cp** to copy the files.

The screenshot shows a desktop environment with a file manager window and a terminal window. The file manager window is open at `/Desktop/Big Data Processing/2. Code Examples/Spark/FileStore/tables/1_Spark_Core$`. It contains three items: `my_dataset`, `my_result`, and `my_result_merge.py`. The terminal window below it shows the command `databricks fs cp -r dbfs:/FileStore/tables/1_Spark_Core/my_result/ ./my_result` being run, followed by several lines of output indicating the copying of files from Databricks File Store to the local directory. The terminal prompt is `dbfs:/FileStore/tables/1_Spark_Core$`.

```
~/Desktop/Big Data Processing/2. Code Examples/Spark/FileStore/tables/1_Spark_Core$ databricks fs cp -r dbfs:/FileStore/tables/1_Spark_Core/my_result/ ./my_result
dbfs:/FileStore/tables/1_Spark_Core/my_result/_SUCCESS -> ./my_result/_SUCCESS
dbfs:/FileStore/tables/1_Spark_Core/my_result/part-00000 -> ./my_result/part-00000
dbfs:/FileStore/tables/1_Spark_Core/my_result/part-00001 -> ./my_result/part-00001
dbfs:/FileStore/tables/1_Spark_Core/my_result/part-00002 -> ./my_result/part-00002
dbfs:/FileStore/tables/1_Spark_Core/my_result/part-00003 -> ./my_result/part-00003
~/Desktop/Big Data Processing/2. Code Examples/Spark/FileStore/tables/1_Spark_Core$
```

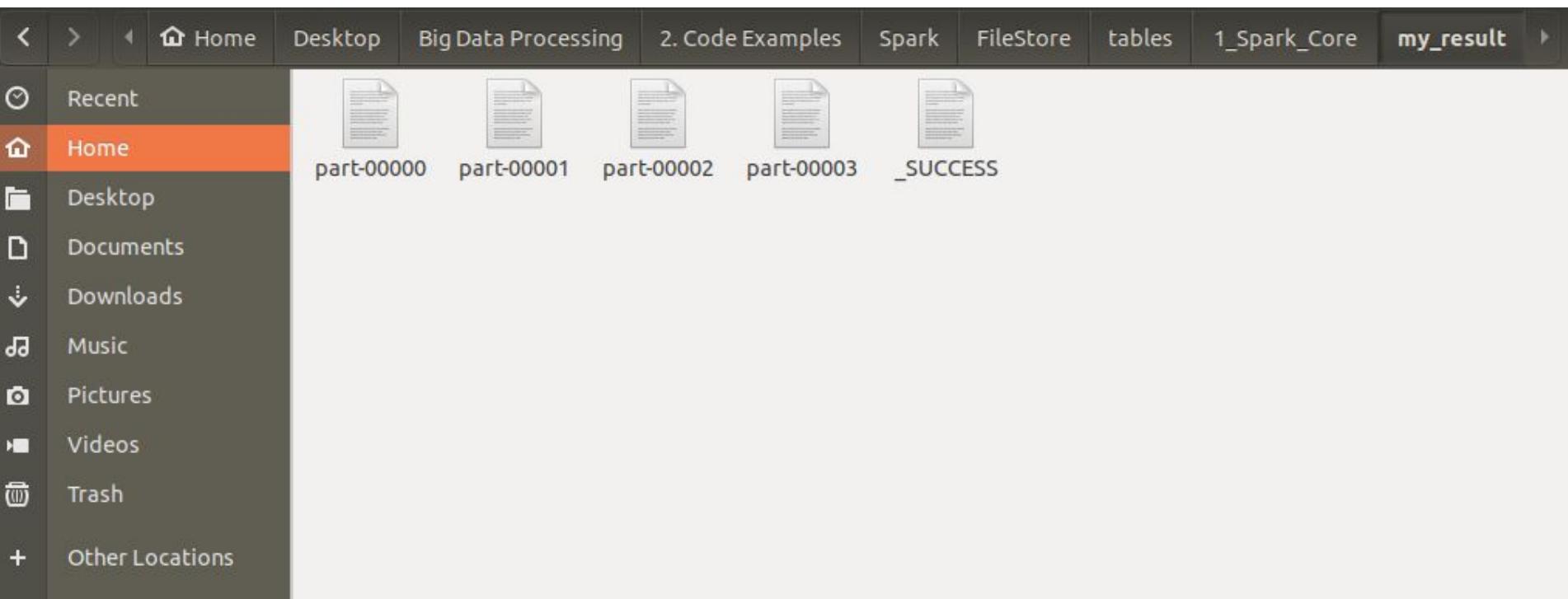
# Databricks: An Online Platform for Data Engineering

9. We run the command **cp** to copy the files.



# Databricks: An Online Platform for Data Engineering

9. And now that the solution files of running the Spark Application are back to our computer we can open them to see their content.



# Databricks: An Online Platform for Data Engineering

9. And now that the solution files of running the Spark Application are back to our computer we can open them to see their content.



part-00000  
~/Desktop/Big Data Processing/2. Code Examples/Spark/FileStore/tables/1\_Spark\_Core/my\_result

```
('the', 29831)
('and', 27499)
('i', 21411)
('to', 20375)
('of', 18503)
('a', 15342)
('you', 13995)
('my', 12952)
('in', 11689)
('that', 11496)
('is', 9545)
('not', 8849)
('with', 8253)
```

# Databricks: An Online Platform for Data Engineers

*Databricks provides much more functionality,  
and I encourage you to explore it.*

*That being said, the one covered in these slides  
represents all the functionality we need to use  
for this module.*

Thank you for your attention!