CORK INSTITUTE OF TECHNOLOGY
INSTITIÚID TEICNEOLAÍOCHTA CHORCAÍ

Big Data Processing

L06: Gentle Introduction to Apache Spark

Dr. Ignacio Castineiras
Department of Computer Science

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

Introduction to Apache Spark

Apache Spark is an

- open-source
- distributed
- general-purpose
- cluster-computing

framework designed for 3 purposes:

Introduction to Apache Spark

Apache Spark is an

- open-source
- distributed
- general-purpose
- cluster-computing

framework designed for 3 purposes:

1. Be easy to use - you can develop applications locally, using a high-level API letting you focus on the content of your computation.

Introduction to Apache Spark

Apache Spark is an

- open-source
- distributed
- general-purpose
- cluster-computing

framework designed for 3 purposes:

1. Be easy to use - you can develop applications locally, using a high-level API letting you focus on the content of your computation.
2. Be fast - enabling interactive use and complex algorithms.

Introduction to Apache Spark

Apache Spark is an

- open-source
- distributed
- general-purpose
- cluster-computing

framework designed for 3 purposes:

1. Be easy to use - you can develop applications locally, using a high-level API letting you focus on the content of your computation.
2. Be fast - enabling interactive use and complex algorithms.
3. Be general - allowing to combine multiple types of computations, including text, SQL, graph and machine learning processing (both offline and online) that might previously have required different engines.

Introduction to Apache Spark

Apache Spark is an

- open-source
- distributed
- general-purpose
- cluster-computing

framework designed for 3 purposes:

1. Be easy to use - you can develop applications locally, using a high-level API letting you focus on the content of your computation.
2. Be fast - enabling interactive use and complex algorithms.
3. Be general - allowing to combine multiple types of computations, including text, SQL, graph and machine learning processing (both offline and online) that might previously have required different engines.

All in all, Spark enables to process large quantities of data, beyond what can fit on a single machine, with a high-level, transparent parallelizable, system agnostic and relatively easy-to-use API.

Introduction to Apache Spark

- Spark started in 2009 as a research project in the UC Berkeley RAD Lab, later to become the AMPLab: https://amplab.cs.berkeley.edu/

Introduction to Apache Spark

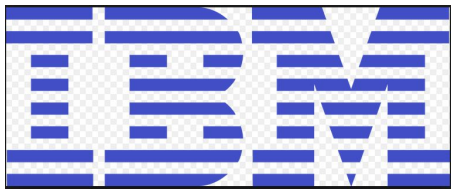
- Spark started in 2009 as a research project in the UC Berkeley RAD Lab, later to become the AMPLab: https://amplab.cs.berkeley.edu/
- Spark was later donated to the Apache Software Foundation, which has maintained it since: http://spark.apache.org/
  Nowadays, Spark has over 1,000 contributors.

Introduction to Apache Spark

- Spark started in 2009 as a research project in the UC Berkeley RAD Lab, later to become the AMPLab: https://amplab.cs.berkeley.edu/
- Spark was later donated to the Apache Software Foundation, which has maintained it since: http://spark.apache.org/
  Nowadays, Spark has over 1,000 contributors.
- On top of the Apache avenue, the creators of Spark also created Databricks, a spin-off company leveraging Spark as a cloud-based big data processing tool https://databricks.com/. It would be our access point to Spark this semester.

Introduction to Apache Spark

- Spark is the state-of-the-art computational engine for big data analytics, and it is used by some of the most important companies, including:



- The full list of companies creating products and projects for use with Apache Spark can be seen at: http://spark.apache.org/powered-by.html

Introduction to Apache Spark

- Spark itself is written in Scala, and runs on the Java Virtual Machine (JVM). It offers simple APIs the proper Scala, and also in Java, Python and R.

Introduction to Apache Spark

- Spark itself is written in Scala, and runs on the Java Virtual Machine (JVM). It offers simple APIs the proper Scala, and also in Java, Python and R.

- In this module we are going to work with Spark in Python. Full stop. That being said, for completeness and, in some cases, to ease some explanations, for each code example presented in this module, I will provide you with the Python and Scala versions of it. But, once again, I don't expect you to write a single line of code in Scala.

Introduction to Apache Spark

- Spark itself is written in Scala, and runs on the Java Virtual Machine (JVM).
  It offers simple APIs the proper Scala, and also in Java, Python and R.

- In this module we are going to work with Spark in Python. Full stop.
  That being said, for completeness and, in some cases, to ease some explanations, for each code example presented in this module, I will provide you with the Python and Scala versions of it.
  But, once again, I don't expect you to write a single line of code in Scala.

- It is important to remark, though, that the Spark code written in Python is often slower than equivalent code written in the JVM, since Scala is statically typed, and the cost of JVM communication (from Python to Scala) can be very high.

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

A Spark Application: General Overview

A bit more technical details... Spark in a nutshell

We said Apache Spark is an open-source, distributed, general-purpose cluster-computing framework designed to be easy to use, fast and general.

To do so, Spark puts together the main concepts we have studied from Distributed programming and some more we will see in the next lecture on functional programming.

A Spark Application: General Overview

1. Spark extends the notion of computation in a single computer to a cluster of n > 1 computers, which can work together towards a common goal (e.g., our lab in room C127, with its 20 computers working together as a cluster).

A Spark Application: General Overview

2. Spark extends the notion of algorithm to the one of meta-algorithm / planner, tackling a concrete problem by coordinating other algorithms on the divide, map and reduce stages of solving it.

A Spark Application: General Overview

2. Spark extends the notion of algorithm to the one of meta-algorithm / planner, tackling a concrete problem by coordinating other algorithms on the divide, map and reduce stages of solving it.
    - Coming back to our lab C127, one of the 20 machines (e.g., machine 1) takes the role of the meta-algorithm (in Spark argot, this means running the driver process).

A Spark Application: General Overview

2. Spark extends the notion of algorithm to the one of meta-algorithm / planner, tackling a concrete problem by coordinating other algorithms on the divide, map and reduce stages of solving it.
    - Coming back to our lab C127, one of the 20 machines (e.g., machine 1) takes the role of the meta-algorithm (in Spark argot, this means running the driver process).
    - This machine (specifically this driver process) coordinates the remaining 19 machines (e.g., machies 2 - 20) by scheduling, distributing, and monitoring the work of their algorithms (in Spark argot, this means coordinating the 19 executor processes operating on them).

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: Resilient Distributed Datasets (RDDs).

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: Resilient Distributed Datasets (RDDs).

    - Thanks to RDDs, a user can write a Spark program as if it was going to be executed in sequential mode by a single machine.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: Resilient Distributed Datasets (RDDs).

    - Thanks to RDDs, a user can write a Spark program as if it was going to be executed in sequential mode by a single machine.
    - Then, under the hood, and also thanks to RDDs, Spark scales and distributes the program execution among the different nodes of the cluster.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

- The ADT private part of an RDD is implemented on data structures which are indeed distributed among different nodes of a cluster.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

- The ADT private part of an RDD is implemented on data structures which are indeed distributed among different nodes of a cluster.
- The complete picture of all existing RDDs, how they are distributed, and the dependencies among them is kept by the driver process.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

- The ADT private part of an RDD is implemented on data structures which are indeed distributed among different nodes of a cluster.
- The complete picture of all existing RDDs, how they are distributed, and the dependencies among them is kept by the driver process.
- Moreover, these RDDs are tried to be kept in memory as long as possible for their fast access.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

- However, the ADT public part of an RDD presents it as a unique and indivisible logical unit, offering an extensive API for its access and manipulation.

A Spark Application: General Overview

3. Spark makes the entire distributed computing environment transparent to the user by offering a novel abstract data type: RDDs.

An RDD is basically a collection of items distributed across many compute nodes that can be manipulated in parallel.

- However, the ADT public part of an RDD presents it as a unique and indivisible logical unit, offering an extensive API for its access and manipulation.

  Such this RDD API supports some of the most important functional programming (FP) features: polymorphism, higher-order functions and partial application.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

   This lazy evaluation-based computation model turns Spark into a declarative programming framework (even if programs are exposed -written- in imperative programming languages as Python, Java or Scala).

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

   This lazy evaluation-based computation model turns Spark into a declarative programming framework (even if programs are exposed -written- in imperative programming languages as Python, Java or Scala).

   This declarative nature of Spark provides an explicit isolation between:

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

   This lazy evaluation-based computation model turns Spark into a declarative programming framework (even if programs are exposed -written- in imperative programming languages as Python, Java or Scala).

   This declarative nature of Spark provides an explicit isolation between:
   - Its declarative semantics: the way a program describes -or declares- the properties a solution might have.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

   This lazy evaluation-based computation model turns Spark into a declarative programming framework (even if programs are exposed -written- in imperative programming languages as Python, Java or Scala).

   This declarative nature of Spark provides an explicit isolation between:
   - Its declarative semantics: the way a program describes -or declares- the properties a solution might have.
   - Its operational semantics: the way a program is analysed to schedule/plan the required computations leading to such desired solution.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - On the declarative semantics side we will study the API of the aforementioned RDDs.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - On the declarative semantics side we will study the API of the aforementioned RDDs.
    - In particular, we will study the two classes of operations that allow accessing and manipulating RDDs (transformations and actions).

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - On the declarative semantics side we will study the API of the aforementioned RDDs.
    - In particular, we will study the two classes of operations that allow accessing and manipulating RDDs (transformations and actions).
    - All in all, we will study how each Spark program is structured towards the following life-cycle:
        a. Create some input RDDs from external data.
        b. Transform them to define new RDDs using transformations.
        c. Persist any intermediate RDDs that are to be reused.
        d. Launch actions to kick off a distributed computation.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - On the operational semantics side we will discuss the anatomy of a Spark application via its jobs, stages and tasks.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - On the operational semantics side we will discuss the anatomy of a Spark application via its jobs, stages and tasks.
    - We will discuss the impact of different modelling, RDD partitions and lazy evaluation in the overall performance of a Spark application.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - For now, it suffices to understand that Spark uses lazy evaluation to analyse the program and create a logical plan of action for solving it, which is based on a Direct Acyclic Graph (DAG) with the dependencies among the computations to be performed.

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - For now, it suffices to understand that Spark uses lazy evaluation to analyse the program and create a logical plan of action for solving it, which is based on a Direct Acyclic Graph (DAG) with the dependencies among the computations to be performed.
    - Such DAG is organised towards atomically (indivisible) units of computation to be performed on partitions of RDDs. They are called tasks, and each task is assigned to a single core/executor process (in a single machine).

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - For now, it suffices to understand that Spark uses lazy evaluation to analyse the program and create a logical plan of action for solving it, which is based on a Direct Acyclic Graph (DAG) with the dependencies among the computations to be performed.
    - Such DAG is organised towards atomically (indivisible) units of computation to be performed on partitions of RDDs. They are called tasks, and each task is assigned to a single core/executor process (in a single machine).
    - The DAG is further optimised and organised towards stages (groups of tasks that can be pipelined together).

A Spark Application: General Overview

4. On planning the optimal execution of a program, Spark makes use of the last functional programming feature: lazy evaluation.

    - For now, it suffices to understand that Spark uses lazy evaluation to analyse the program and create a logical plan of action for solving it, which is based on a Direct Acyclic Graph (DAG) with the dependencies among the computations to be performed.
    - Such DAG is organised towards atomically (indivisible) units of computation to be performed on partitions of RDDs. They are called tasks, and each task is assigned to a single core/executor process (in a single machine).
    - The DAG is further optimised and organised towards stages (groups of tasks that can be pipelined together).
    - Finally, the logical plan stated by the DAG is translated into a physical plan by scheduling and tracking the execution of the tasks over the computers of the cluster.

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
but to have it *"ready"* for using Spark has additional requirements:

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.
  2. A common Spark version has to be installed across the cluster.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.
  2. A common Spark version has to be installed across the cluster.
  3. A Distributed File System has to be in place to offer a global storage view of the cluster.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.
  2. A common Spark version has to be installed across the cluster.
  3. A Distributed File System has to be in place to offer a global storage view of the cluster.
  4. OS and security updates have to be taken into account as well.

Two Different Roles: IT Manager vs. Data Engineer

- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.
  2. A common Spark version has to be installed across the cluster.
  3. A Distributed File System has to be in place to offer a global storage view of the cluster.
  4. OS and security updates have to be taken into account as well.

- All these tasks make room for a role as Cluster IT Manager, which is out of the scope of this module, and thus we are not going to cover at all.

Two Different Roles: IT Manager vs. Data Engineer

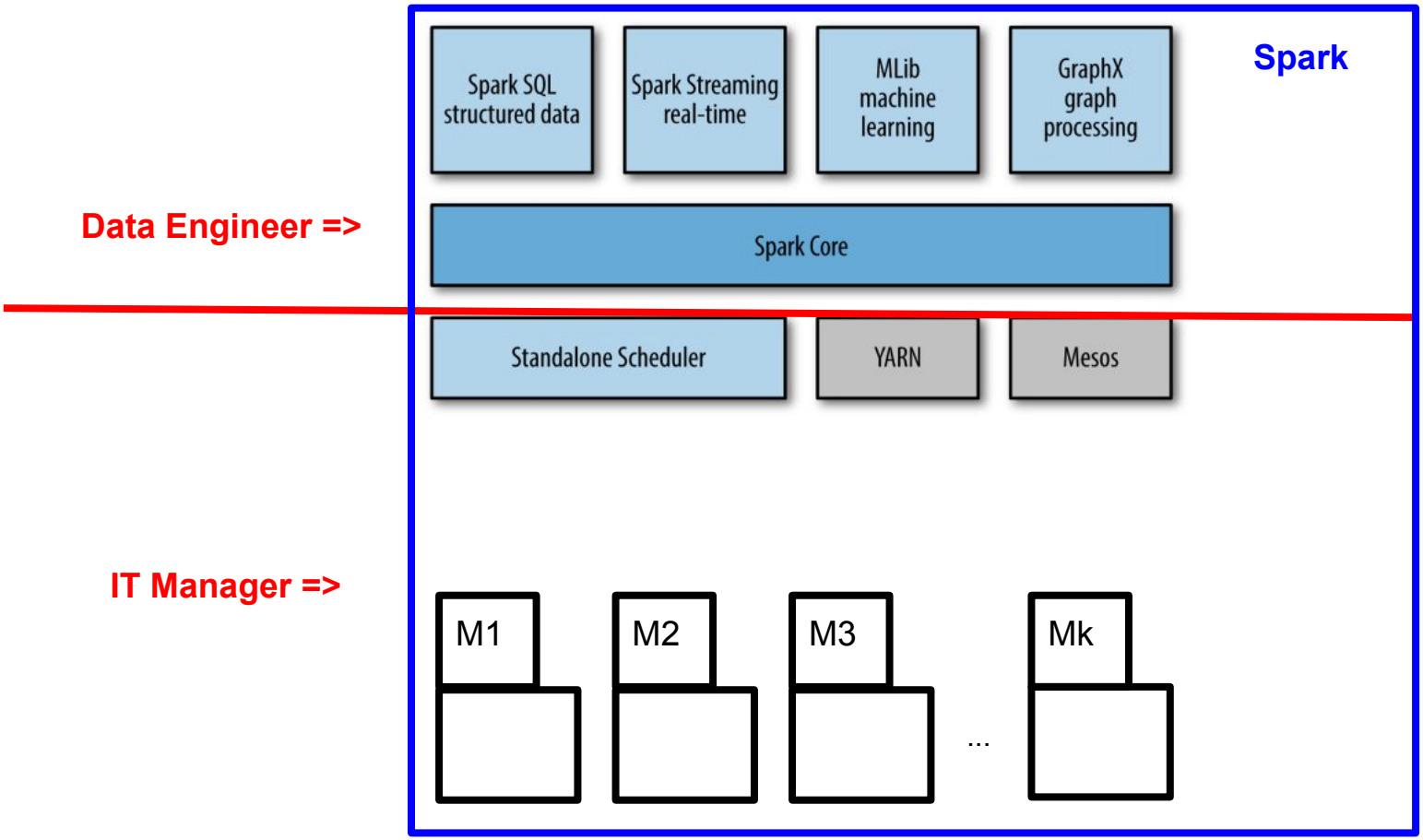
- The fact of Spark being a cluster-computing framework for data analysis requires to have a cluster on the first place.

- Having a cluster of n > 1 computers is not complicated,
  but to have it *"ready"* for using Spark has additional requirements:
  1. All computers must be able to communicate.
  2. A common Spark version has to be installed across the cluster.
  3. A Distributed File System has to be in place to offer a global storage view of the cluster.
  4. OS and security updates have to be taken into account as well.

- All these tasks make room for a role as Cluster IT Manager, which is out of the scope of this module, and thus we are not going to cover at all.
  If you want to learn more about the topic perhaps a good book is:
  - ❖ Spark Operations: Operationalizing Apache Spark at Scale.
    Timothy Chen et al. O'Reilly, 2016.

Two Different Roles: IT Manager vs. Data Engineer

- We are going to be 100% focused on the role of the Data Scientist or Data Engineer, who uses Spark on top of a *"ready"* cluster.



Two Different Roles: IT Manager vs. Data Engineer

- As we saw in the previous picture, Spark contains multiple components, some of them IT Manager-related and some Data Engineer-related.

Two Different Roles: IT Manager vs. Data Engineer

- As we saw in the previous picture, Spark contains multiple components, some of them IT Manager-related and some Data Engineer-related.
- On the Data Engineer side of things, the amount of components aims to efficiently support a wide range of workloads, including stream processing, graph-based processing, machine learning processing (and any combination of the aforementioned).

Two Different Roles: IT Manager vs. Data Engineer

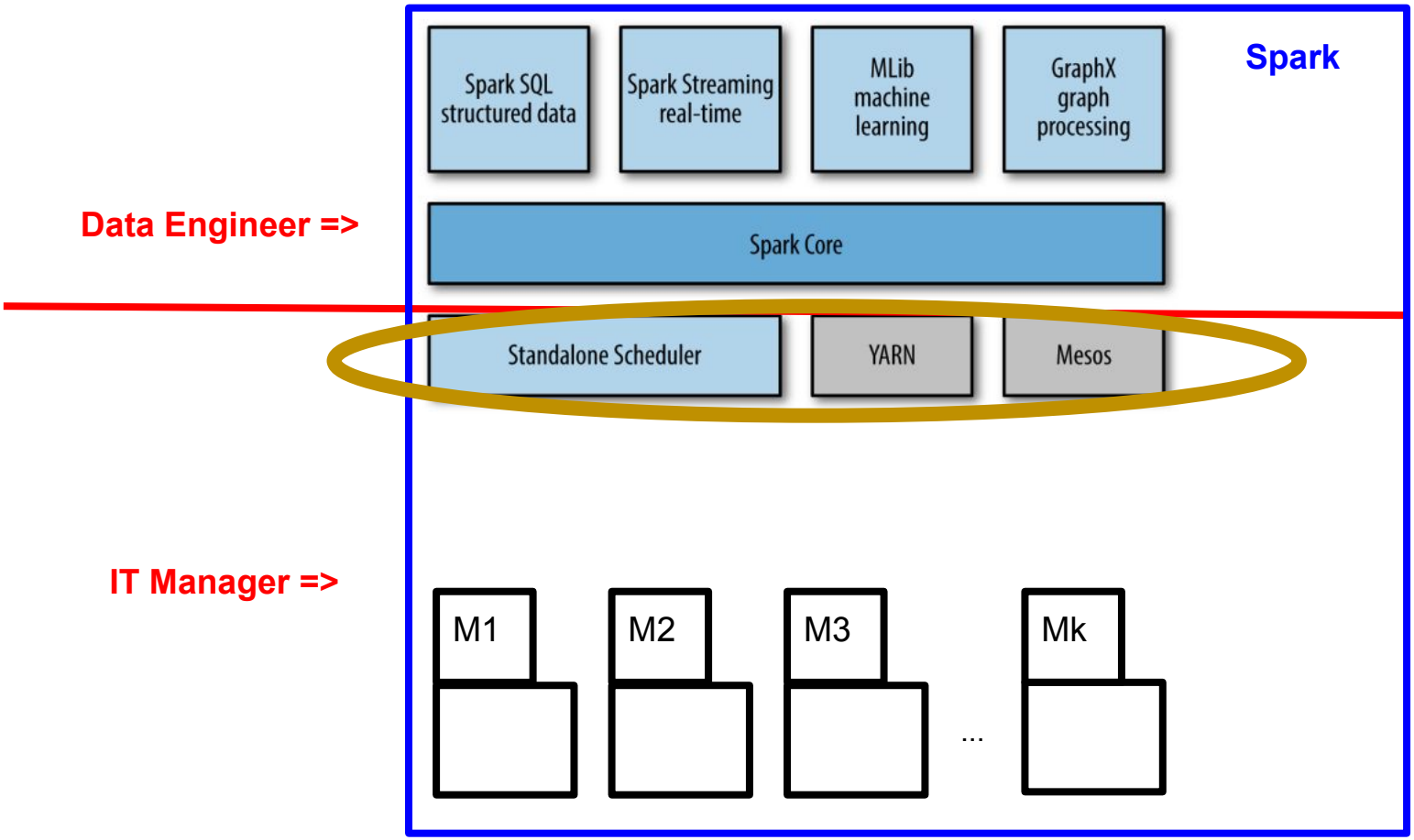
- As we saw in the previous picture, Spark contains multiple components, some of them IT Manager-related and some Data Engineer-related.
- On the Data Engineer side of things, the amount of components aims to efficiently support a wide range of workloads, including stream processing, graph-based processing, machine learning processing (and any combination of the aforementioned).
- By supporting these workloads in the same engine, Spark makes it easy and inexpensive to combine different processing types, which is handy for domain applications requiring the use of several of them.

Two Different Roles: IT Manager vs. Data Engineer

- As we saw in the previous picture, Spark contains multiple components, some of them IT Manager-related and some Data Engineer-related.
- On the Data Engineer side of things, the amount of components aims to efficiently support a wide range of workloads, including stream processing, graph-based processing, machine learning processing (and any combination of the aforementioned).
- By supporting these workloads in the same engine, Spark makes it easy and inexpensive to combine different processing types, which is handy for domain applications requiring the use of several of them.
- We describe each of these components now...

Two Different Roles: IT Manager vs. Data Engineer

- IT Manager => Cluster Manager Component



Two Different Roles: IT Manager vs. Data Engineer

- On its own, Spark is not a data storage solution; it performs computations on Java Virtual Machines (JVMs) that last only for the duration of a Spark application.

Two Different Roles: IT Manager vs. Data Engineer

- On its own, Spark is not a data storage solution; it performs computations on Java Virtual Machines (JVMs) that last only for the duration of a Spark application.

- Spark can be run locally on a single machine with a single JVM (called local mode). However, Spark is designed to efficiently scale up from one to many thousands of compute nodes.

Two Different Roles: IT Manager vs. Data Engineer

- On its own, Spark is not a data storage solution; it performs computations on Java Virtual Machines (JVMs) that last only for the duration of a Spark application.

- Spark can be run locally on a single machine with a single JVM (called local mode). However, Spark is designed to efficiently scale up from one to many thousands of compute nodes.

- When used in a cluster, Spark is used in tandem with a distributed storage system and a cluster manager.

Two Different Roles: IT Manager vs. Data Engineer

- On its own, Spark is not a data storage solution; it performs computations on Java Virtual Machines (JVMs) that last only for the duration of a Spark application.

- Spark can be run locally on a single machine with a single JVM (called local mode). However, Spark is designed to efficiently scale up from one to many thousands of compute nodes.

- When used in a cluster, Spark is used in tandem with a distributed storage system and a cluster manager.
  - The storage system is used to provide the input dataset used by Spark application, as well as to stable store the results such application produces.

Two Different Roles: IT Manager vs. Data Engineer

- On its own, Spark is not a data storage solution; it performs computations on Java Virtual Machines (JVMs) that last only for the duration of a Spark application.

- Spark can be run locally on a single machine with a single JVM (called local mode). However, Spark is designed to efficiently scale up from one to many thousands of compute nodes.

- When used in a cluster, Spark is used in tandem with a distributed storage system and a cluster manager.
  - The storage system is used to provide the input dataset used by Spark application, as well as to stable store the results such application produces.
  - The cluster manager is used to orchestrate the distribution of Spark applications across the cluster.

Two Different Roles: IT Manager vs. Data Engineer

- This cluster manager is in charge of putting in contact the storage of the dataset with its further processing (via one or more of the data engineering components).
  Spark can run over a variety of cluster managers, including Hadoop YARN, Apache Mesos, and a simple cluster manager included in Spark itself called the Standalone Scheduler.

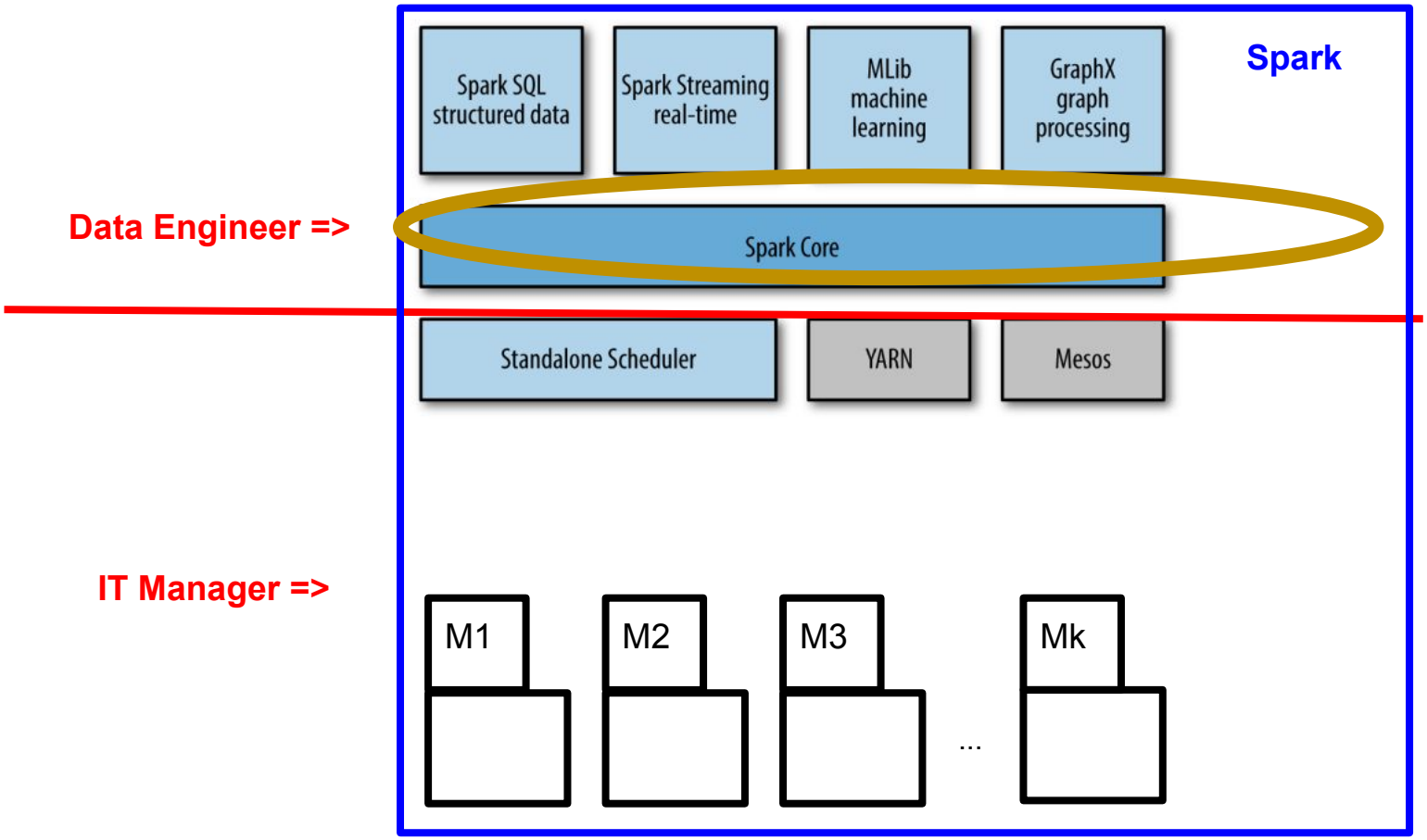
Two Different Roles: IT Manager vs. Data Engineer

- This cluster manager is in charge of putting in contact the storage of the dataset with its further processing (via one or more of the data engineering components).
  Spark can run over a variety of cluster managers, including Hadoop YARN, Apache Mesos, and a simple cluster manager included in Spark itself called the Standalone Scheduler.

- In terms of file storage, Spark can work on Hadoop Distributed File System (HDFS) or similar (when using Spark with the Databricks online system we will use their specific Databricks File System - DBFS).

Two Different Roles: IT Manager vs. Data Engineer

- Data Engineer => Spark Core



Two Different Roles: IT Manager vs. Data Engineer

- Spark Core contains the basic functionality of Spark, including:

Two Different Roles: IT Manager vs. Data Engineer

- Spark Core contains the basic functionality of Spark, including:
  - The main data abstraction being offered to users to express their programs: The Resilient Distributed Datasets (RDDs).

Two Different Roles: IT Manager vs. Data Engineer

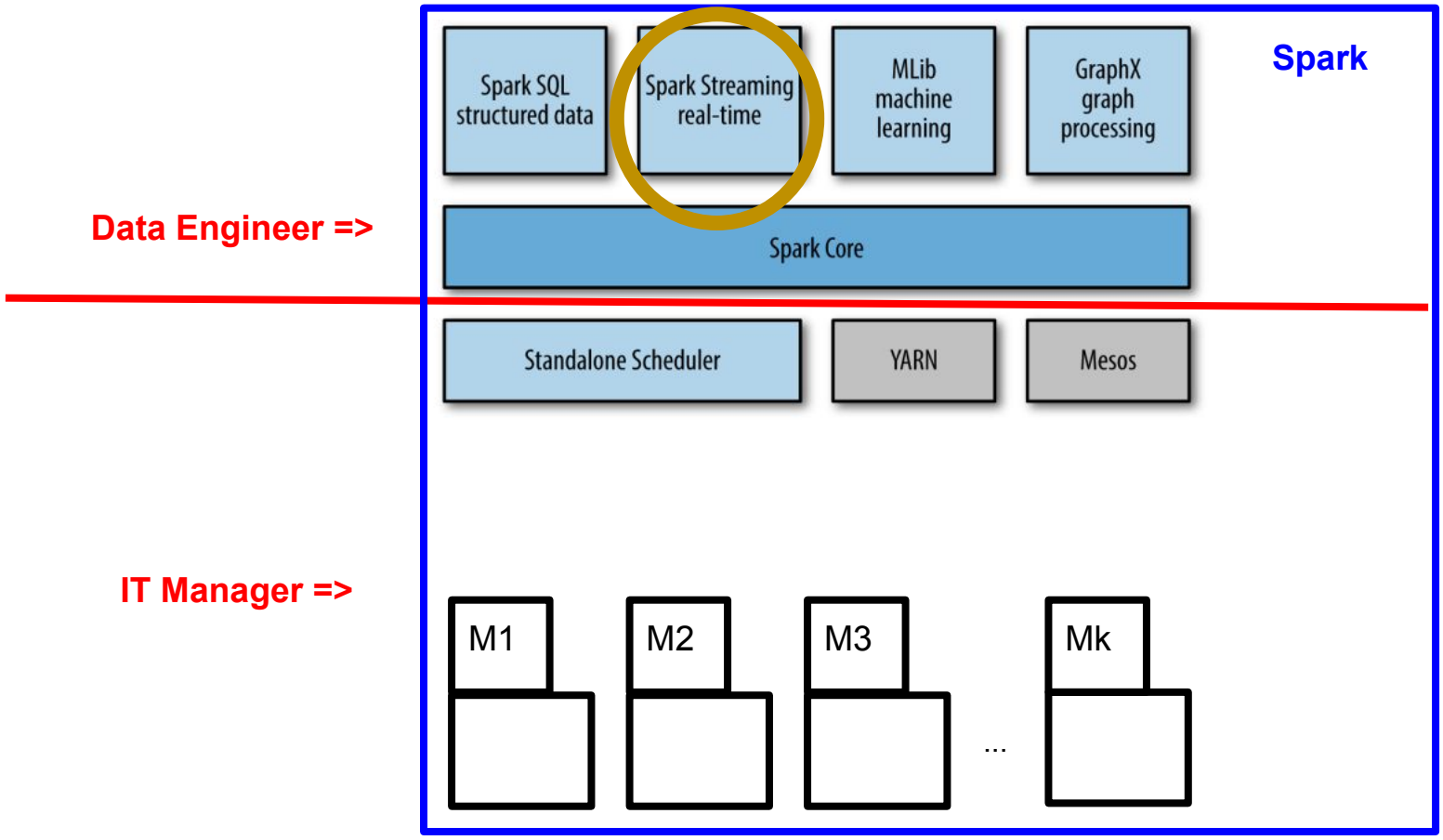
- Spark Core contains the basic functionality of Spark, including:
  - The main data abstraction being offered to users to express their programs: The Resilient Distributed Datasets (RDDs).
  - The implementation of the driver and executor processes that Spark uses to run each application.
    - This includes the Spark driver functionality to generate a logical plan (Direct Acyclic Graph per job, with concrete stages and tasks) and a physical plan (schedule and tracking of the tasks execution).

Two Different Roles: IT Manager vs. Data Engineer

- Spark Core contains the basic functionality of Spark, including:
  - The main data abstraction being offered to users to express their programs: The Resilient Distributed Datasets (RDDs).
  - The implementation of the driver and executor processes that Spark uses to run each application.
    - This includes the Spark driver functionality to generate a logical plan (Direct Acyclic Graph per job, with concrete stages and tasks) and a physical plan (schedule and tracking of the tasks execution).
  - Other functionality such as memory management, fault recovery and the interaction with storage systems.

Two Different Roles: IT Manager vs. Data Engineer

- Data Engineer => Spark Streaming



Two Different Roles: IT Manager vs. Data Engineer

- Spark Streaming is a Spark component that enables processing of live streams of data.
  - Examples of data streams include log-files generated by production web servers, or queues of messages containing status updates posted by users of a web service.

Two Different Roles: IT Manager vs. Data Engineer

- Spark Streaming is a Spark component that enables processing of live streams of data.
  - Examples of data streams include log-files generated by production web servers, or queues of messages containing status updates posted by users of a web service.

- The main data abstraction being offered is called DStream (from discretised stream), and is based on the amalgamation of RDDs over time via batch intakes.

Two Different Roles: IT Manager vs. Data Engineer

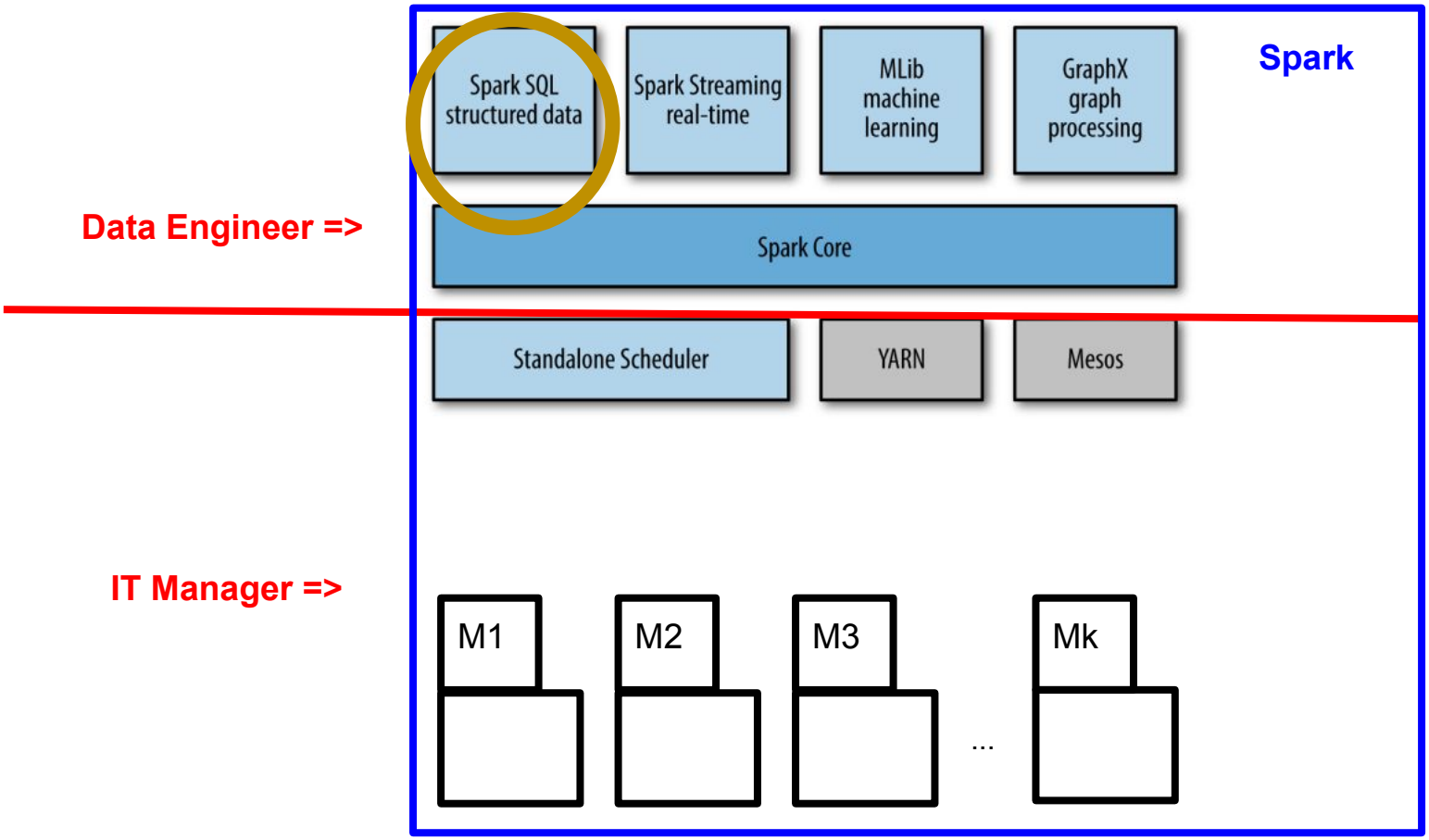
- Spark Streaming is a Spark component that enables processing of live streams of data.
  - Examples of data streams include log-files generated by production web servers, or queues of messages containing status updates posted by users of a web service.

- The main data abstraction being offered is called DStream (from discretised stream), and is based on the amalgamation of RDDs over time via batch intakes.

- The library offers a good range of stateless and stateful operations. However, as is the case with all RDD-based components, it has not been developed further to entice users to use its counterpart DataFrame-based components.

Two Different Roles: IT Manager vs. Data Engineer

- Data Engineer => Spark SQL



Two Different Roles: IT Manager vs. Data Engineer

- Spark SQL is the module integrating relational processing with Spark's functional programming API.

Two Different Roles: IT Manager vs. Data Engineer

- Spark SQL is the module integrating relational processing with Spark's functional programming API.
- The users benefit of a higher-level data abstraction (DataFrames) for ingesting, querying and persisting (semi)structured data using relational queries via a Domain Specific Language (DSL).

Two Different Roles: IT Manager vs. Data Engineer

- Spark SQL is the module integrating relational processing with Spark's functional programming API.
- The users benefit of a higher-level data abstraction (DataFrames) for ingesting, querying and persisting (semi)structured data using relational queries via a Domain Specific Language (DSL).
- Under the hood, Spark SQL translates a DataFrame-based program into an equivalent RDD-based one via:

Two Different Roles: IT Manager vs. Data Engineer

- Spark SQL is the module integrating relational processing with Spark's functional programming API.
- The users benefit of a higher-level data abstraction (DataFrames) for ingesting, querying and persisting (semi)structured data using relational queries via a Domain Specific Language (DSL).
- Under the hood, Spark SQL translates a DataFrame-based program into an equivalent RDD-based one via:
  - A Catalyst optimiser (which semantically analyse the query expressed by the user).

Two Different Roles: IT Manager vs. Data Engineer

- Spark SQL is the module integrating relational processing with Spark's functional programming API.

- The users benefit of a higher-level data abstraction (DataFrames) for ingesting, querying and persisting (semi)structured data using relational queries via a Domain Specific Language (DSL).

- Under the hood, Spark SQL translates a DataFrame-based program into an equivalent RDD-based one via:
  - A Catalyst optimiser (which semantically analyse the query expressed by the user).
  - A Tungsten encoder optimiser (which translates the query to an equivalent binary RDD-based format).

Two Different Roles: IT Manager vs. Data Engineer

- As mentioned before, Spark SQL has also incorporated its own library for processing streaming datasets, called Structured Streaming.

Two Different Roles: IT Manager vs. Data Engineer

- As mentioned before, Spark SQL has also incorporated its own library for processing streaming datasets, called Structured Streaming.

- Based on DataFrames, it allows users to express a streaming computation in a transparent way, just as a batch computation on static data. The Spark SQL engine takes care of running it incrementally and continuously and updating the final result as streaming data continues to arrive.

Two Different Roles: IT Manager vs. Data Engineer

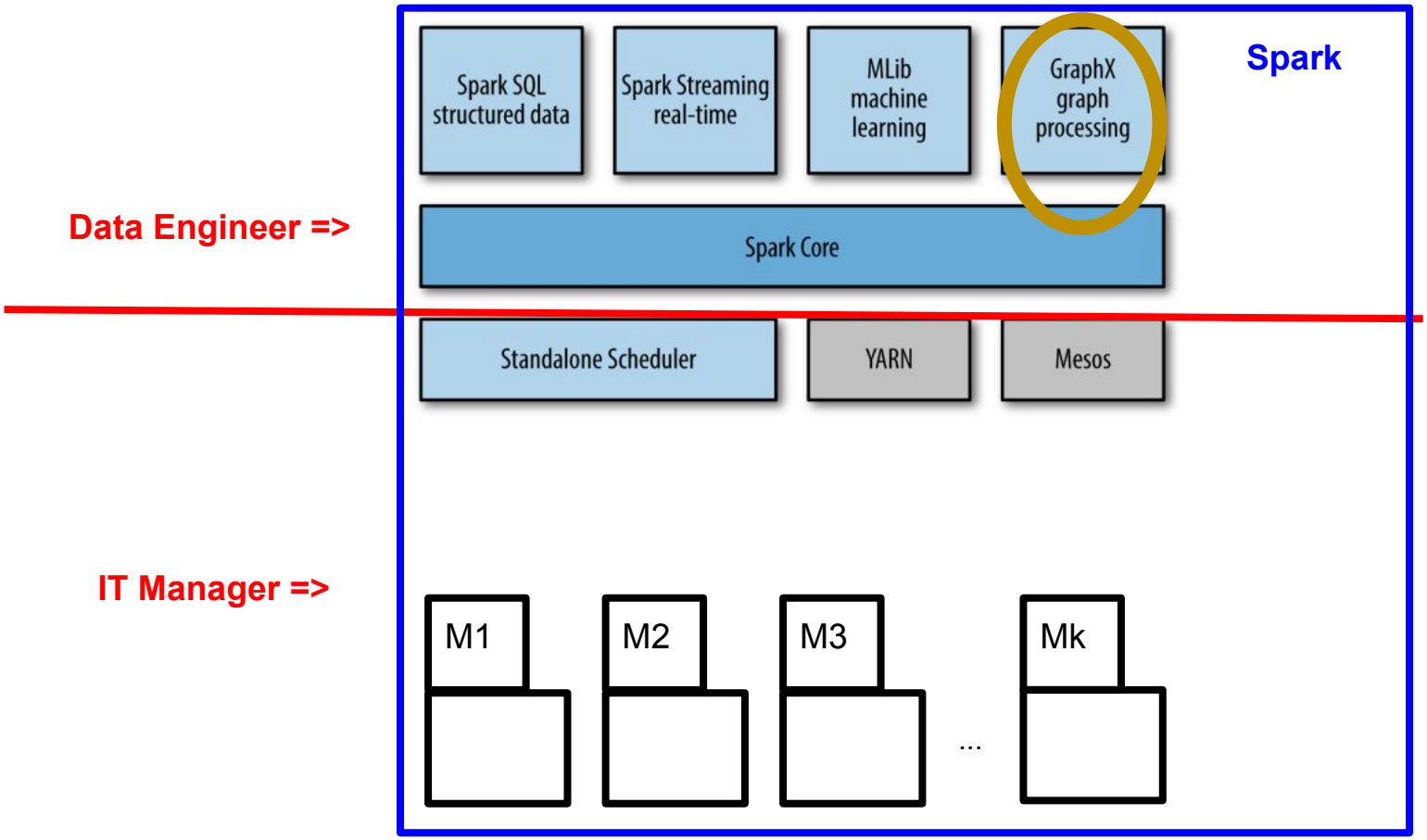
- As mentioned before, Spark SQL has also incorporated its own library for processing streaming datasets, called Structured Streaming.

- Based on DataFrames, it allows users to express a streaming computation in a transparent way, just as a batch computation on static data. The Spark SQL engine takes care of running it incrementally and continuously and updating the final result as streaming data continues to arrive.

- The system also supports managing late-time data arrival offering end-to-end exactly-once fault-tolerance guarantees through checkpointing and write ahead Logs.

Two Different Roles: IT Manager vs. Data Engineer

- Data Engineer => Graph



Two Different Roles: IT Manager vs. Data Engineer

- GraphX is a library for manipulating graphs (e.g., a social network□s friend graph) and performing graph-parallel computations.

Two Different Roles: IT Manager vs. Data Engineer

- GraphX is a library for manipulating graphs (e.g., a social network□s friend graph) and performing graph-parallel computations.
- GraphX extends the Spark RDD API (allowing us to create directed graphs with arbitrary properties attached to each vertex and edge) and provides a library with common graph algorithms.
-

Two Different Roles: IT Manager vs. Data Engineer

- GraphX is a library for manipulating graphs (e.g., a social network□s friend graph) and performing graph-parallel computations.

- GraphX extends the Spark RDD API (allowing us to create directed graphs with arbitrary properties attached to each vertex and edge) and provides a library with common graph algorithms.

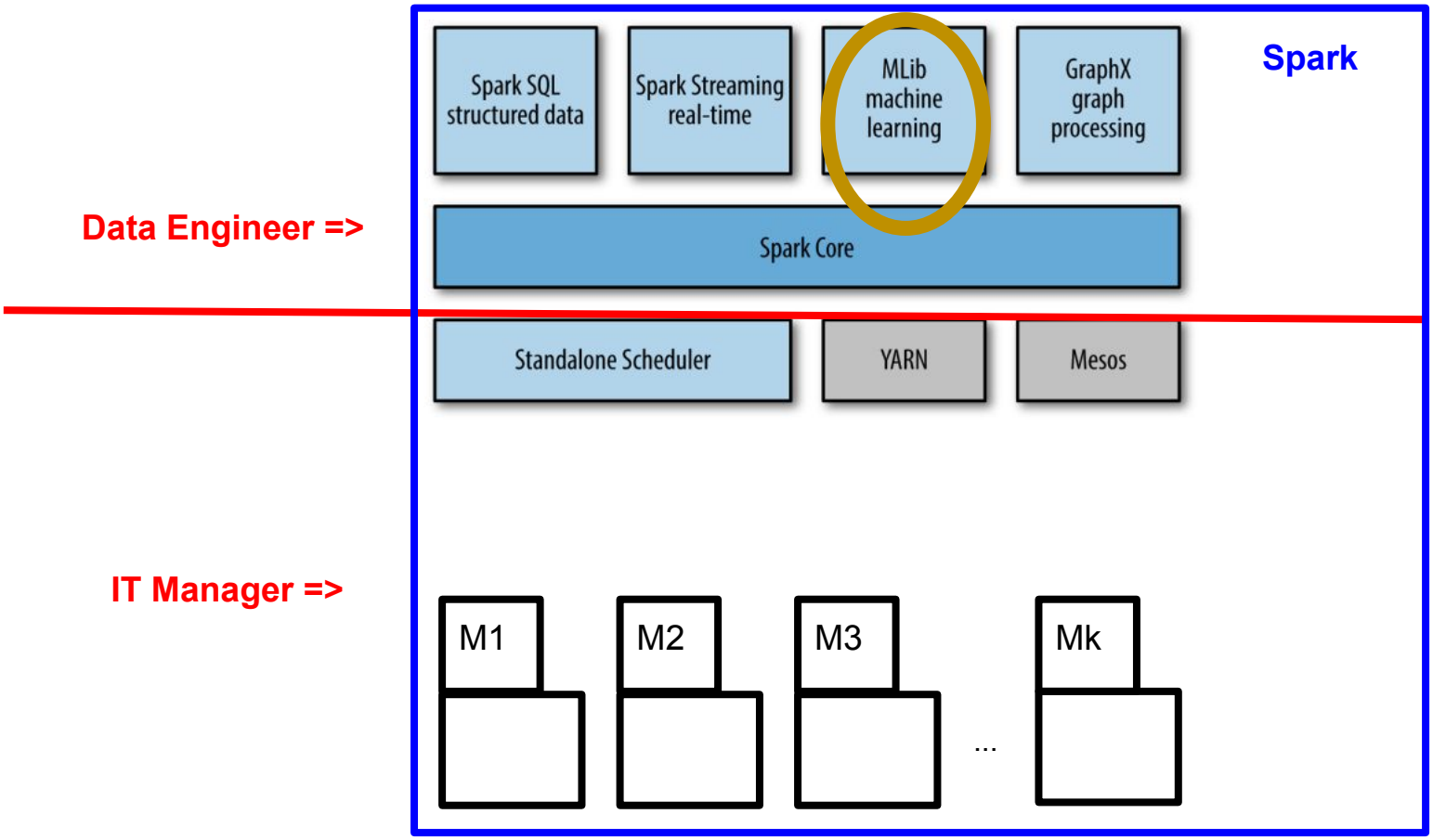
- Unfortunately this library is only available via Scala and Java, and thus for working in Python we will use its DataFrame counterpart GraphFrames. As its name indicates, this library is based on the DataFrames abstraction, although it provides its own graph-based language notation for users to expressing their programs.

Two Different Roles: IT Manager vs. Data Engineer

- GraphX is a library for manipulating graphs (e.g., a social network□s friend graph) and performing graph-parallel computations.
- GraphX extends the Spark RDD API (allowing us to create directed graphs with arbitrary properties attached to each vertex and edge) and provides a library with common graph algorithms.
- Unfortunately this library is only available via Scala and Java, and thus for working in Python we will use its DataFrame counterpart GraphFrames. As its name indicates, this library is based on the DataFrames abstraction, although it provides its own graph-based language notation for users to expressing their programs.
- It is very unlikely that we will have time to cover this component at all in our 22 lectures. However, I will pass you on all the code examples I have created for it.

Two Different Roles: IT Manager vs. Data Engineer

- Data Engineer => MLlib



Two Different Roles: IT Manager vs. Data Engineer

- Finally, Spark also comes with components for common machine learning functionality, called MLlib (RDD-based) and ML (DataFrame-based).

Two Different Roles: IT Manager vs. Data Engineer

- Finally, Spark also comes with components for common machine learning functionality, called MLlib (RDD-based) and ML (DataFrame-based).

- They provide multiple types of machine learning algorithms, including classification, regression, clustering, and collaborative filtering, as well as supporting functionality such as model evaluation and data import. All of these methods are designed to scale out across a cluster.

Two Different Roles: IT Manager vs. Data Engineer

- Finally, Spark also comes with components for common machine learning functionality, called MLlib (RDD-based) and ML (DataFrame-based).

- They provide multiple types of machine learning algorithms, including classification, regression, clustering, and collaborative filtering, as well as supporting functionality such as model evaluation and data import. All of these methods are designed to scale out across a cluster.

- It is very unlikely that we will have time to cover this component at all in our 22 lectures. Indeed, I am only starting learning it myself.
  However, if I have time to create some code examples for it I will pass then on to you at the end of the semester.

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

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: 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 abstracts the IT management tasks by allowing to set up and configure a cluster with just a few clicks.

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 abstracts the IT management tasks by allowing to set up and configure a cluster with just a few clicks.
- It provides an easy interface for performing Data Engineering tasks (import, edit and running Spark applications) on top of the cluster being created.

Databricks: An Online Platform for Data Engineers

- The Databricks Community Edition is free to use.
  It will be our approach in this module.

Databricks: An Online Platform for Data Engineers

- The Databricks Community Edition is free to use.
  It will be our approach in this module.

- 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 our approach in this module.

- 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.

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

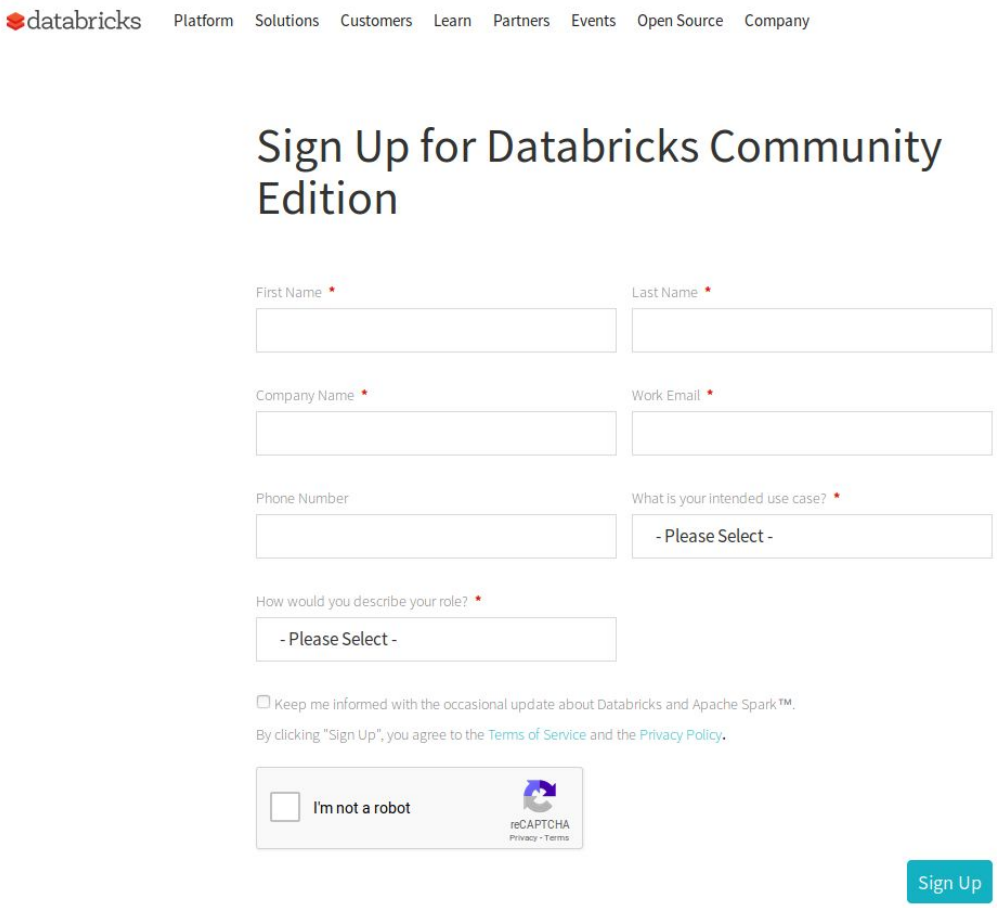
Databricks: An Online Platform for Data Engineers

How to...

Sign up / sign in.

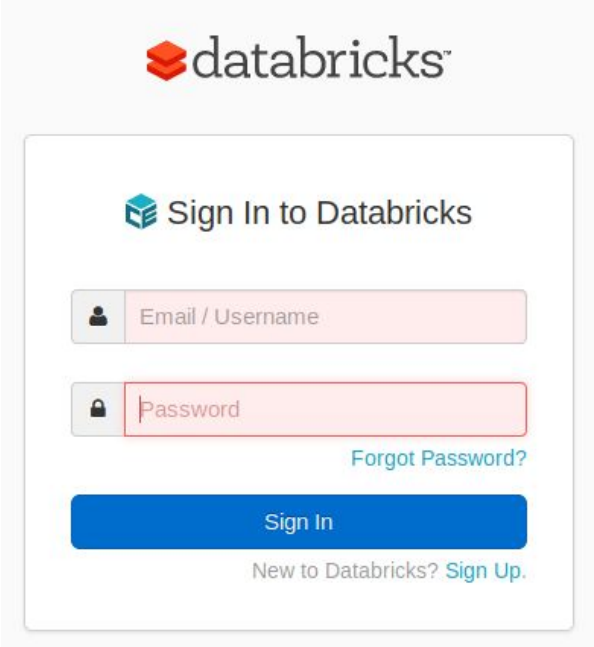
Databricks: An Online Platform for Data Engineers

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



Databricks: An Online Platform for Data Engineers

2. Once registered, we sign in at: https://community.cloud.databricks.com



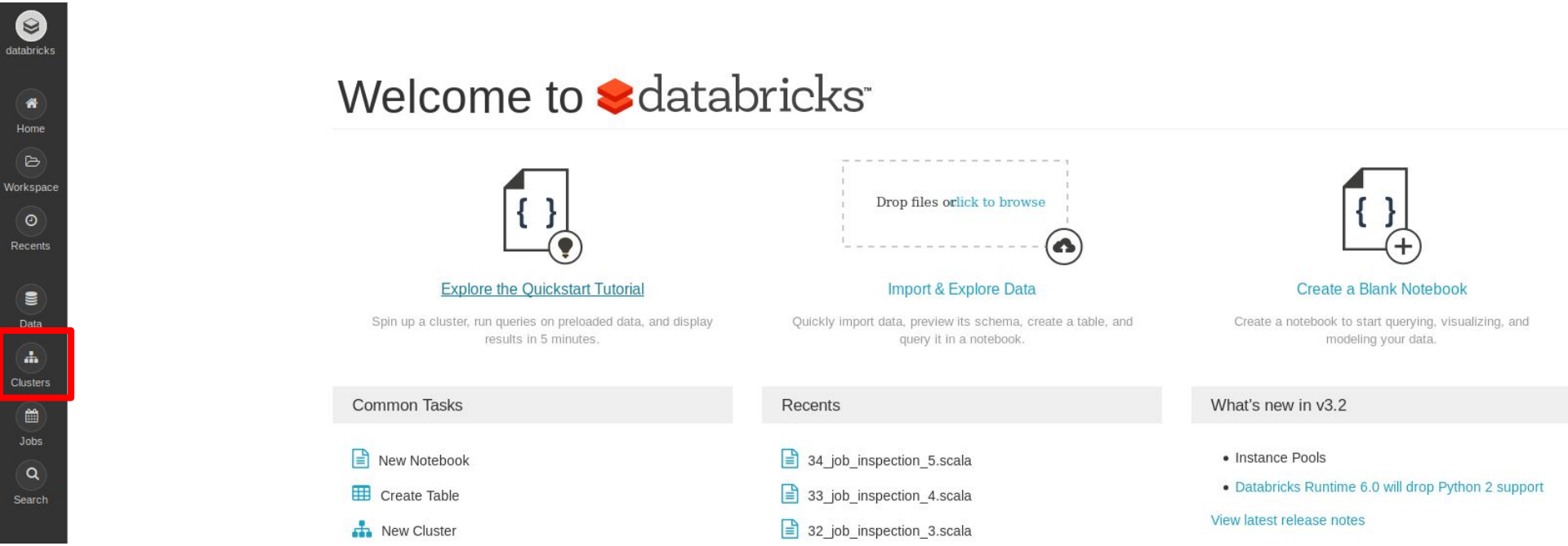
Databricks: An Online Platform for Data Engineers

How to...

Create a new cluster.

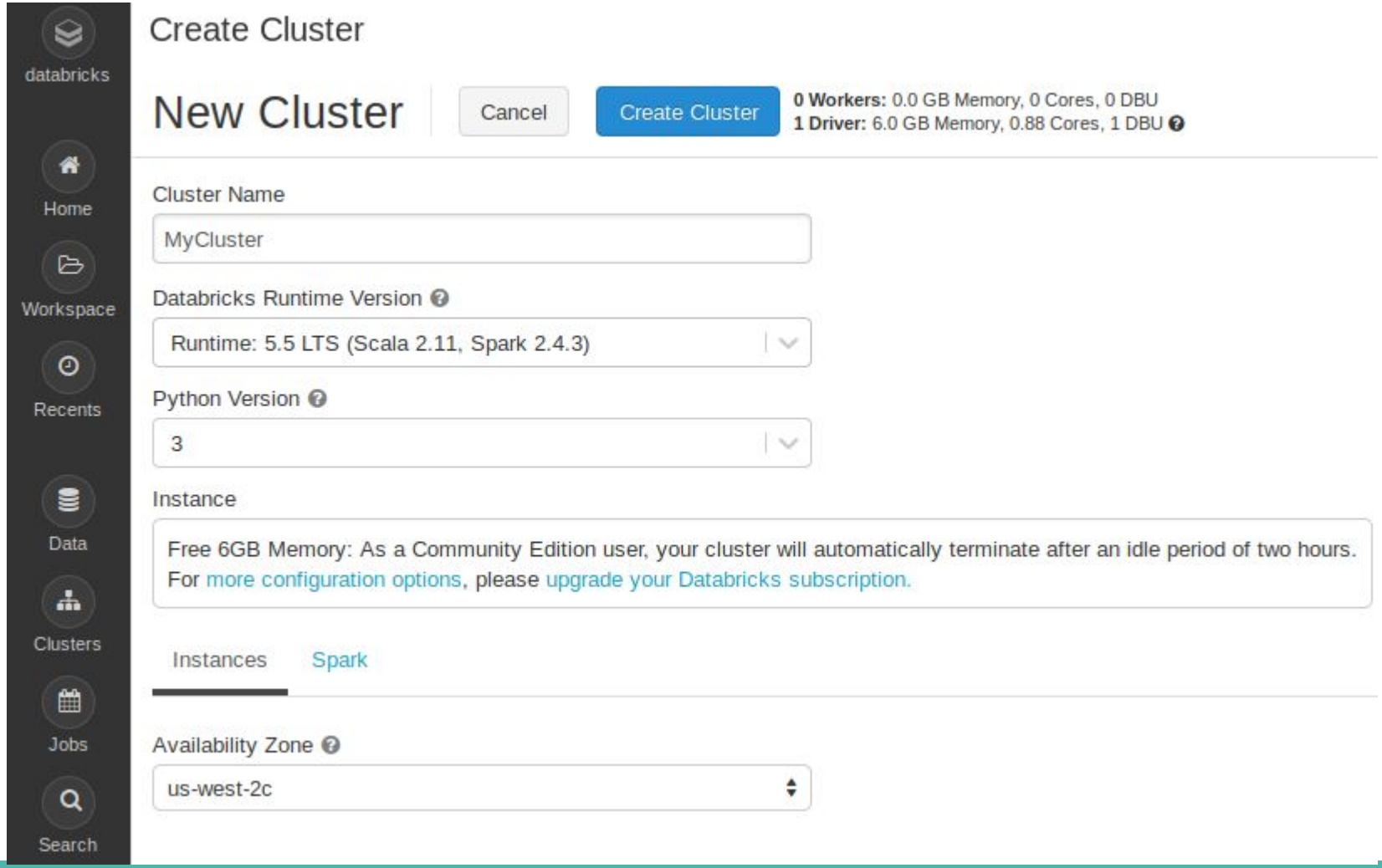
Databricks: An Online Platform for Data Engineers

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**.



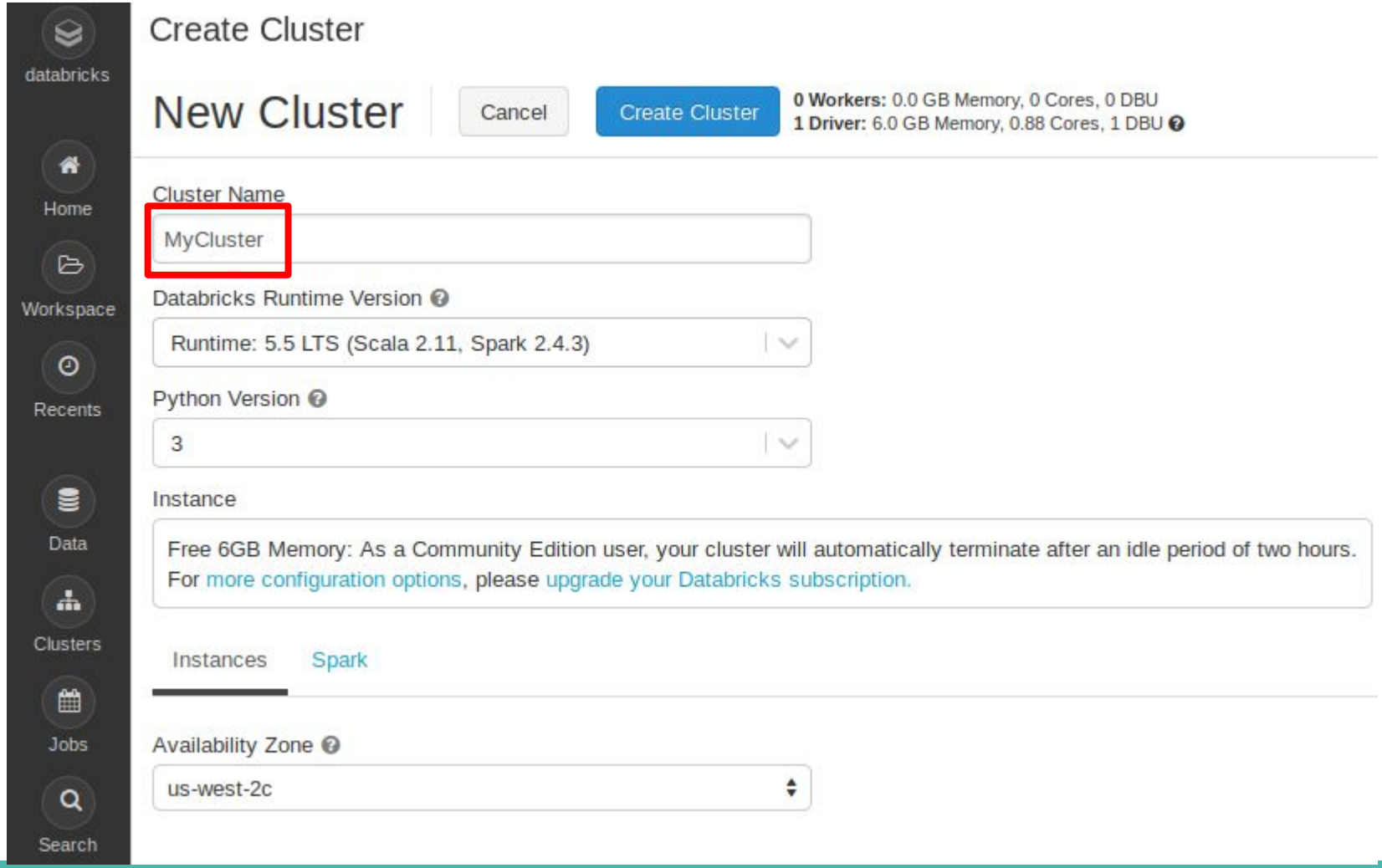
Databricks: An Online Platform for Data Engineers

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



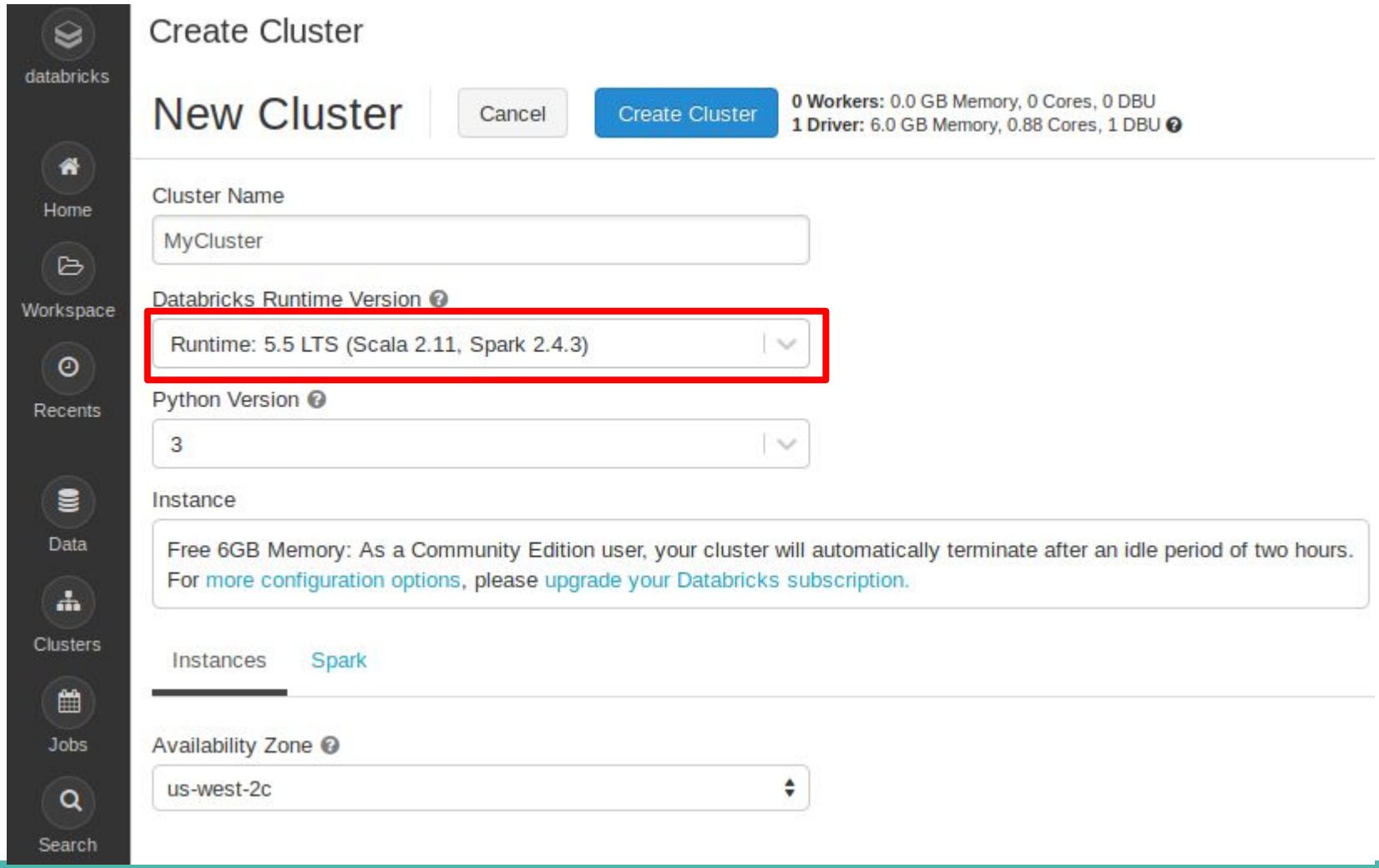
Databricks: An Online Platform for Data Engineers

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



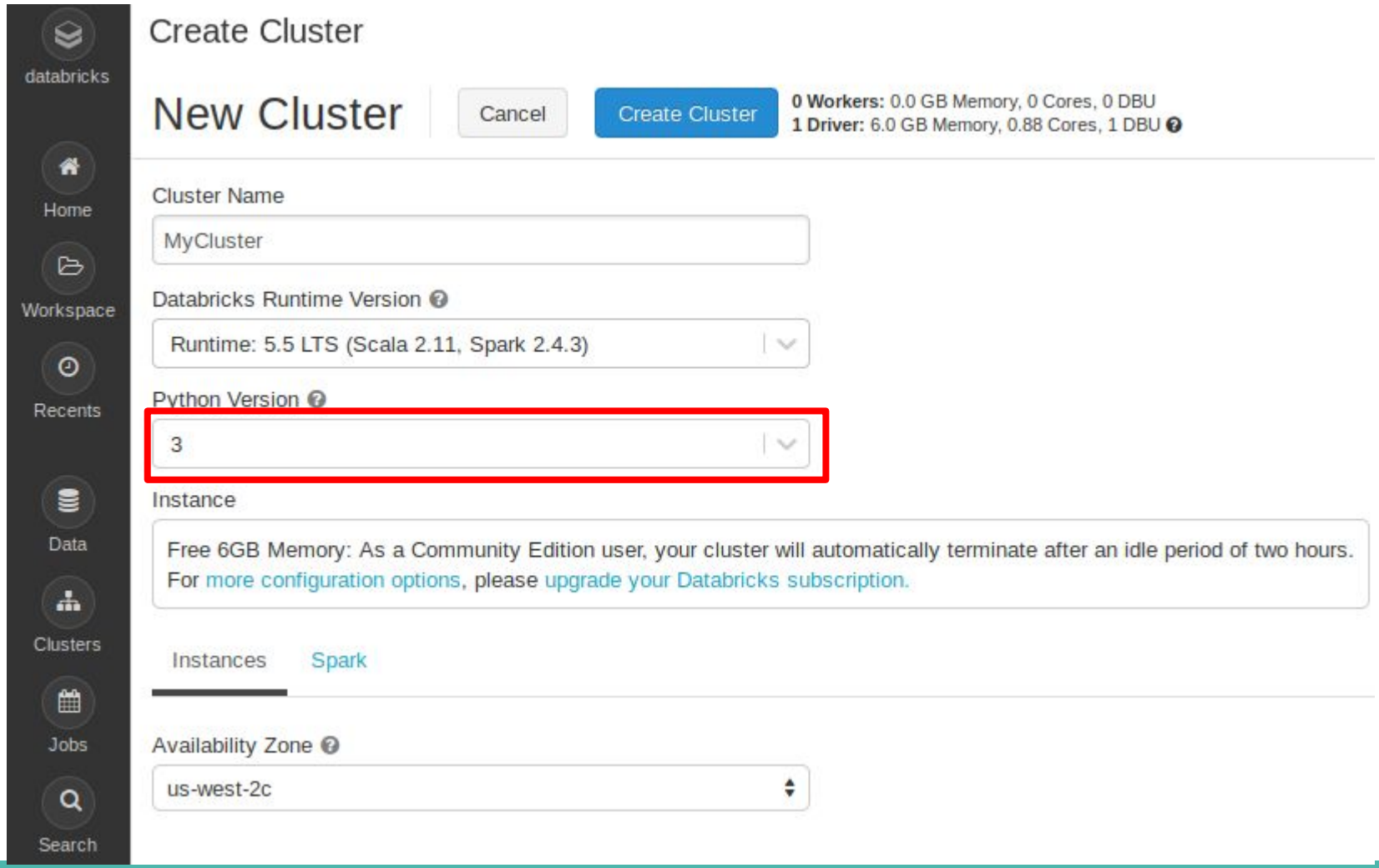
Databricks: An Online Platform for Data Engineers

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



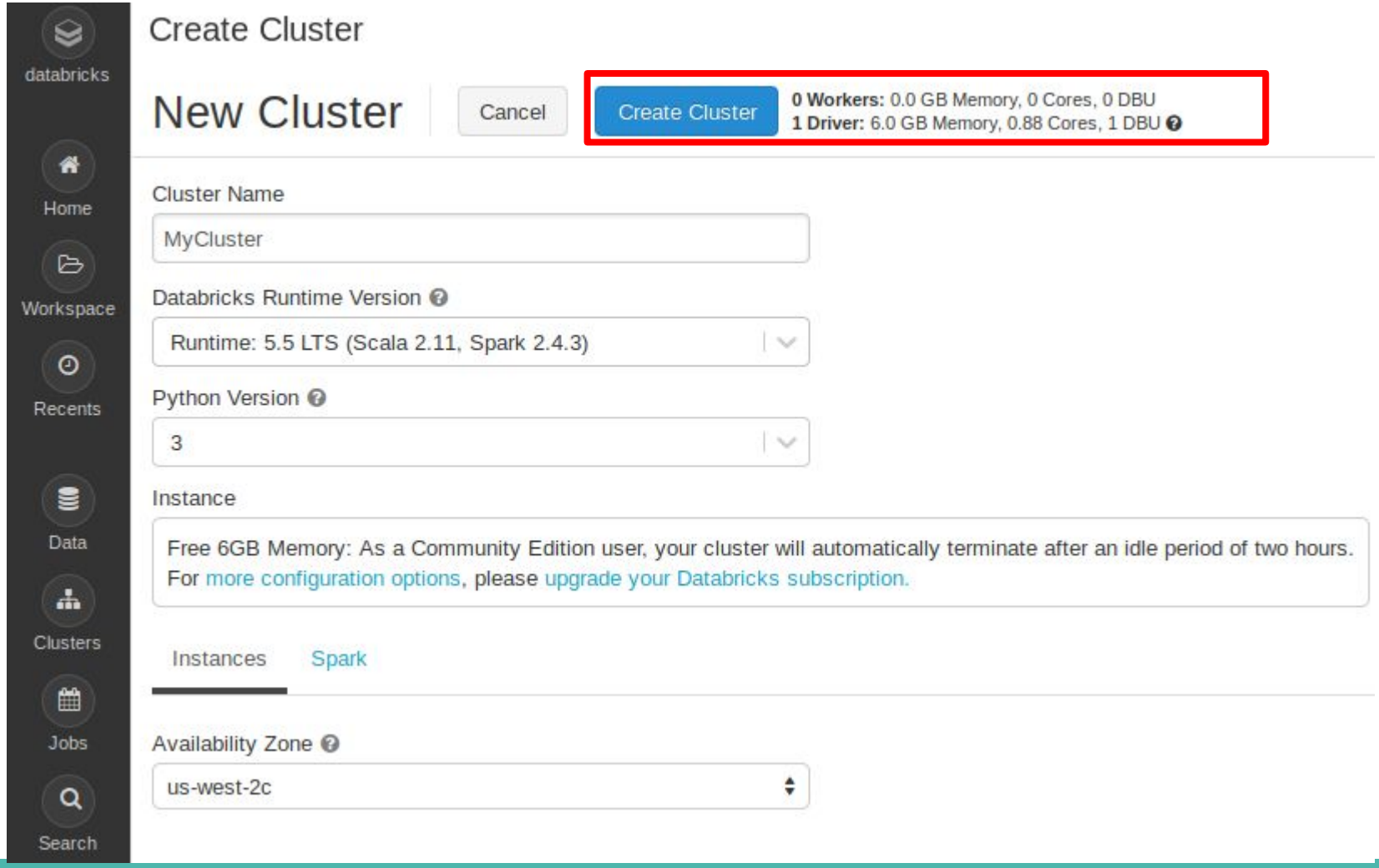
Databricks: An Online Platform for Data Engineers

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



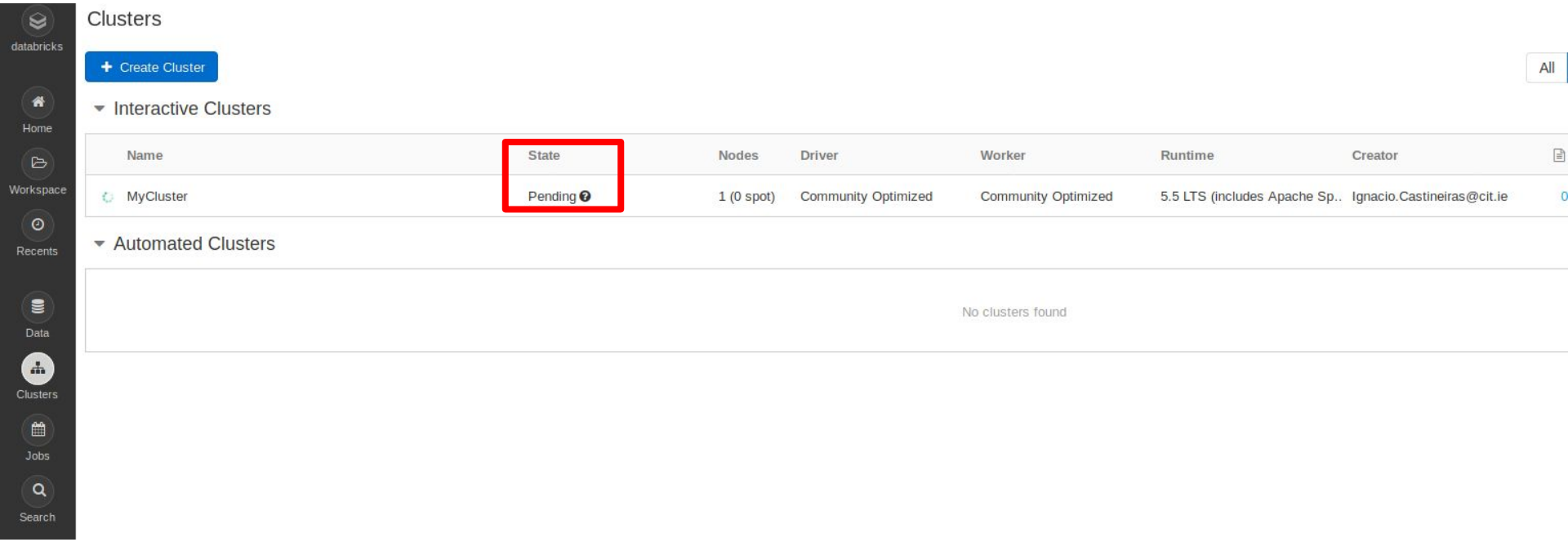
Databricks: An Online Platform for Data Engineers

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



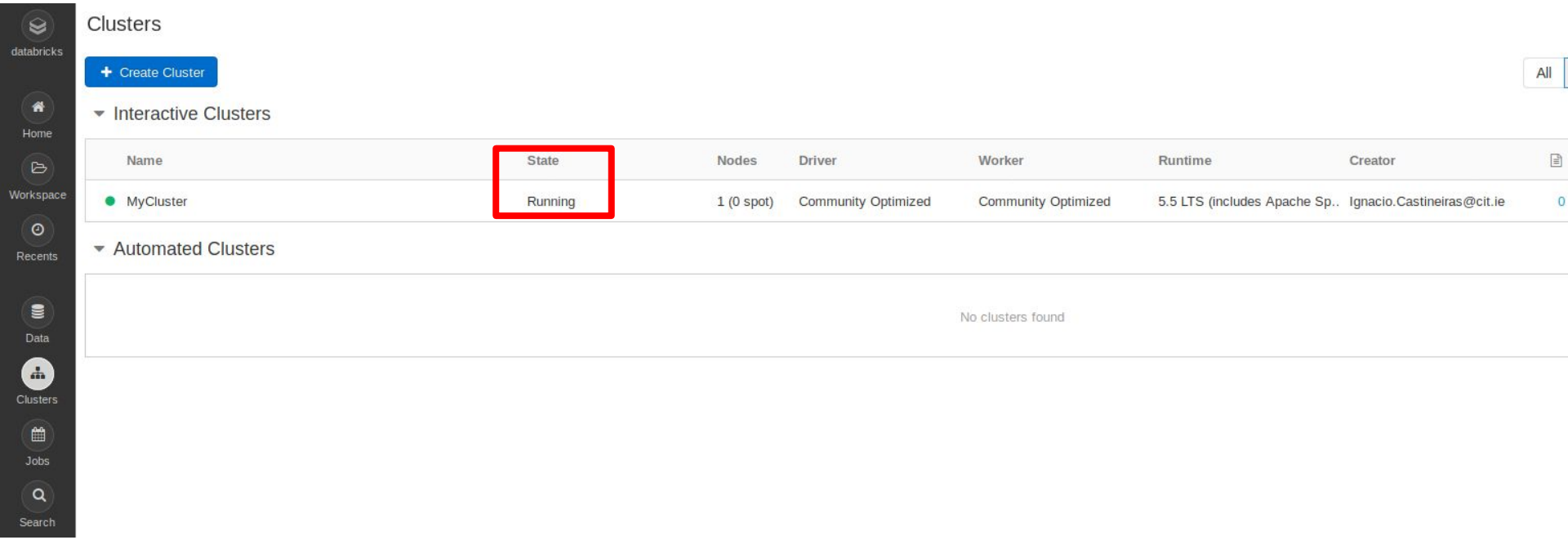
Databricks: An Online Platform for Data Engineers

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



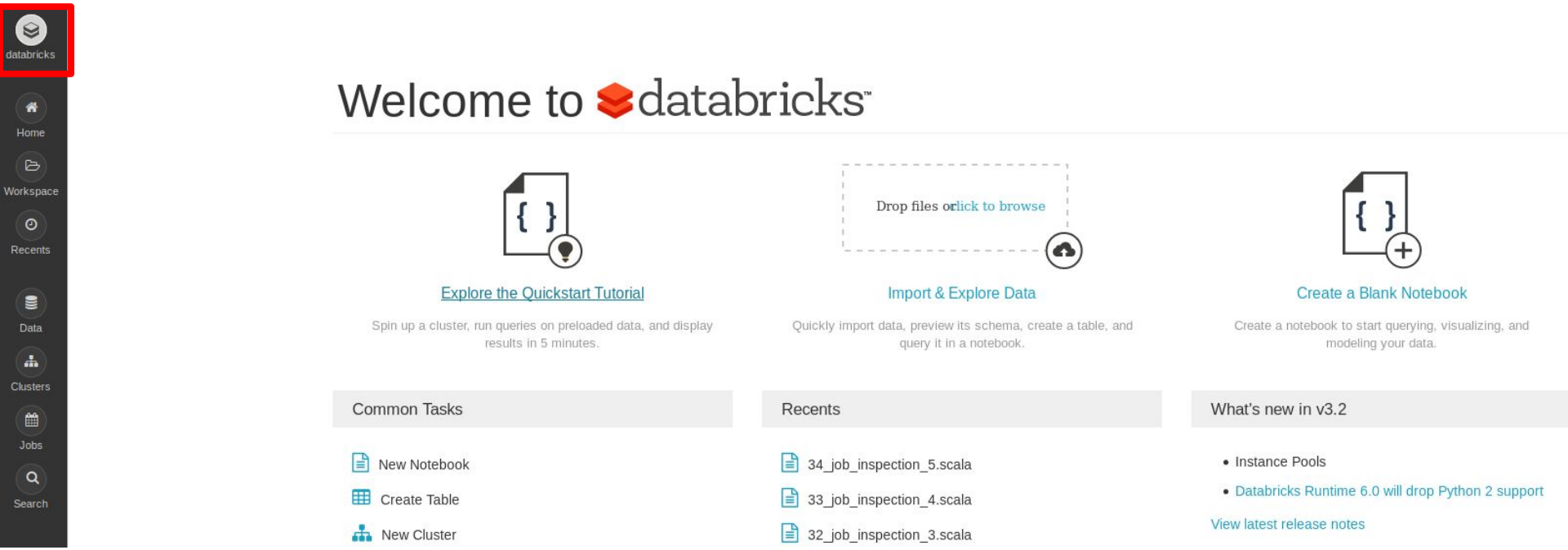
Databricks: An Online Platform for Data Engineers

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



Databricks: An Online Platform for Data Engineers

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



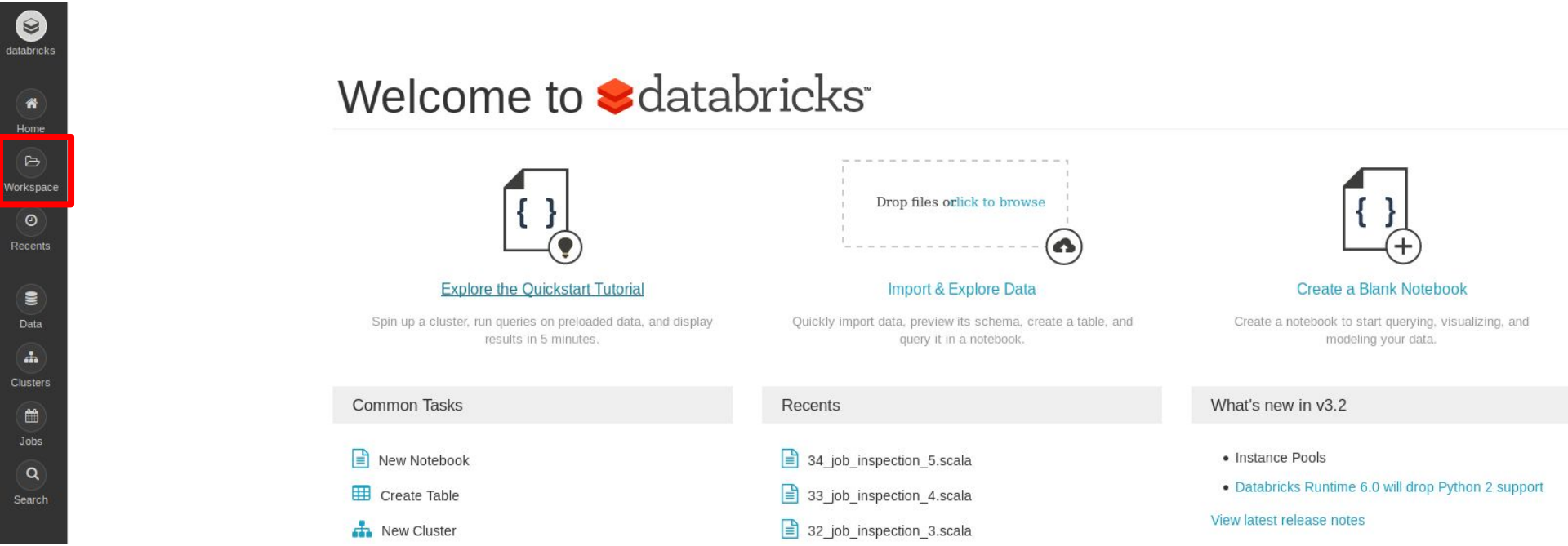
Databricks: An Online Platform for Data Engineers

How to...

Create a new program / Spark Application.

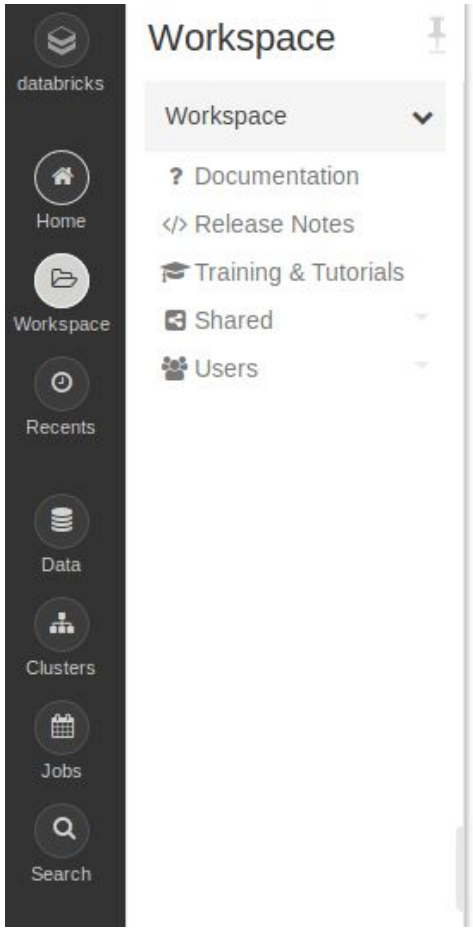
Databricks: An Online Platform for Data Engineers

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



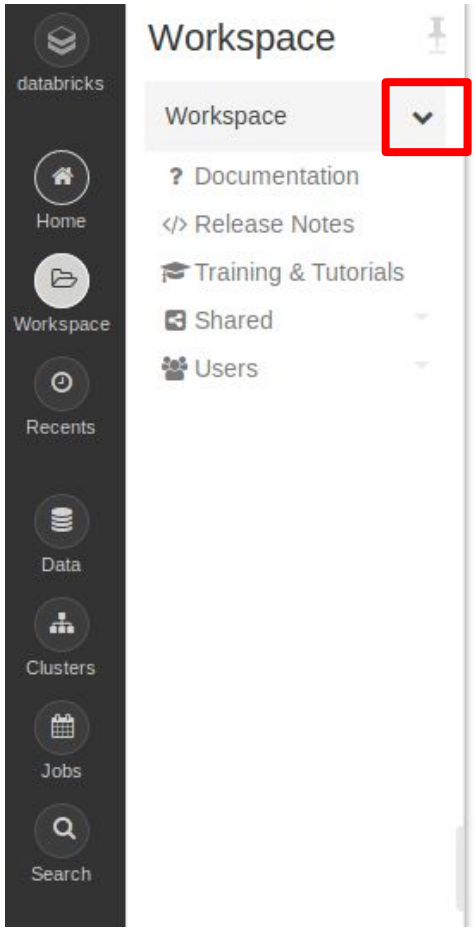
Databricks: An Online Platform for Data Engineers

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



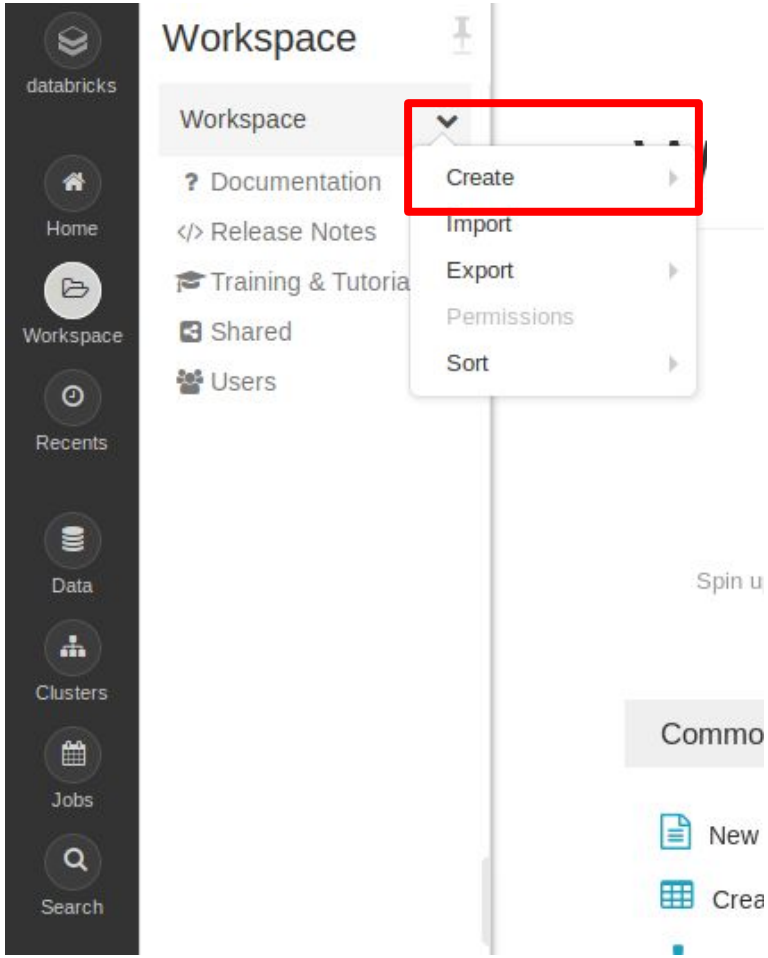
Databricks: An Online Platform for Data Engineers

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



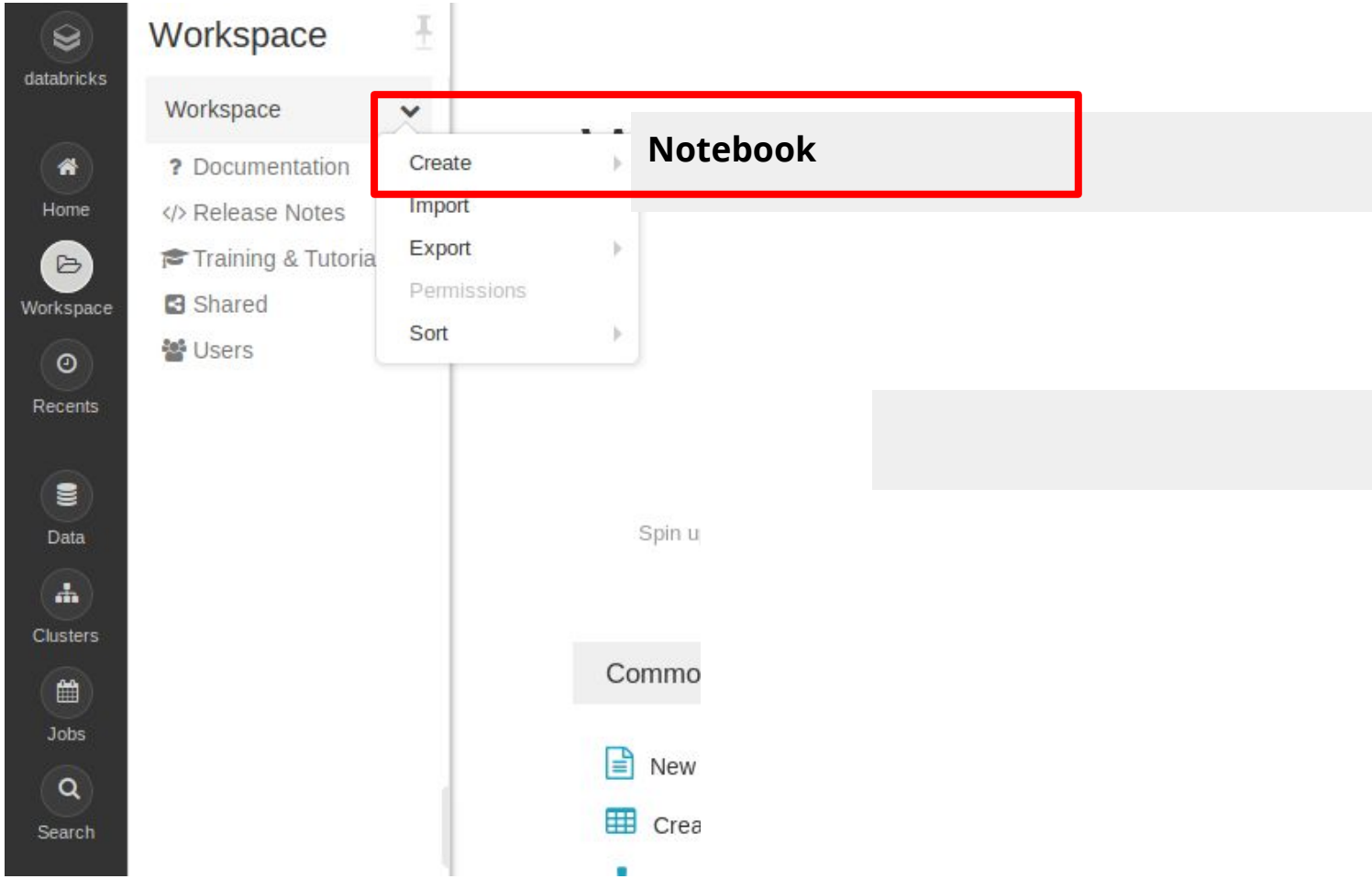
Databricks: An Online Platform for Data Engineers

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



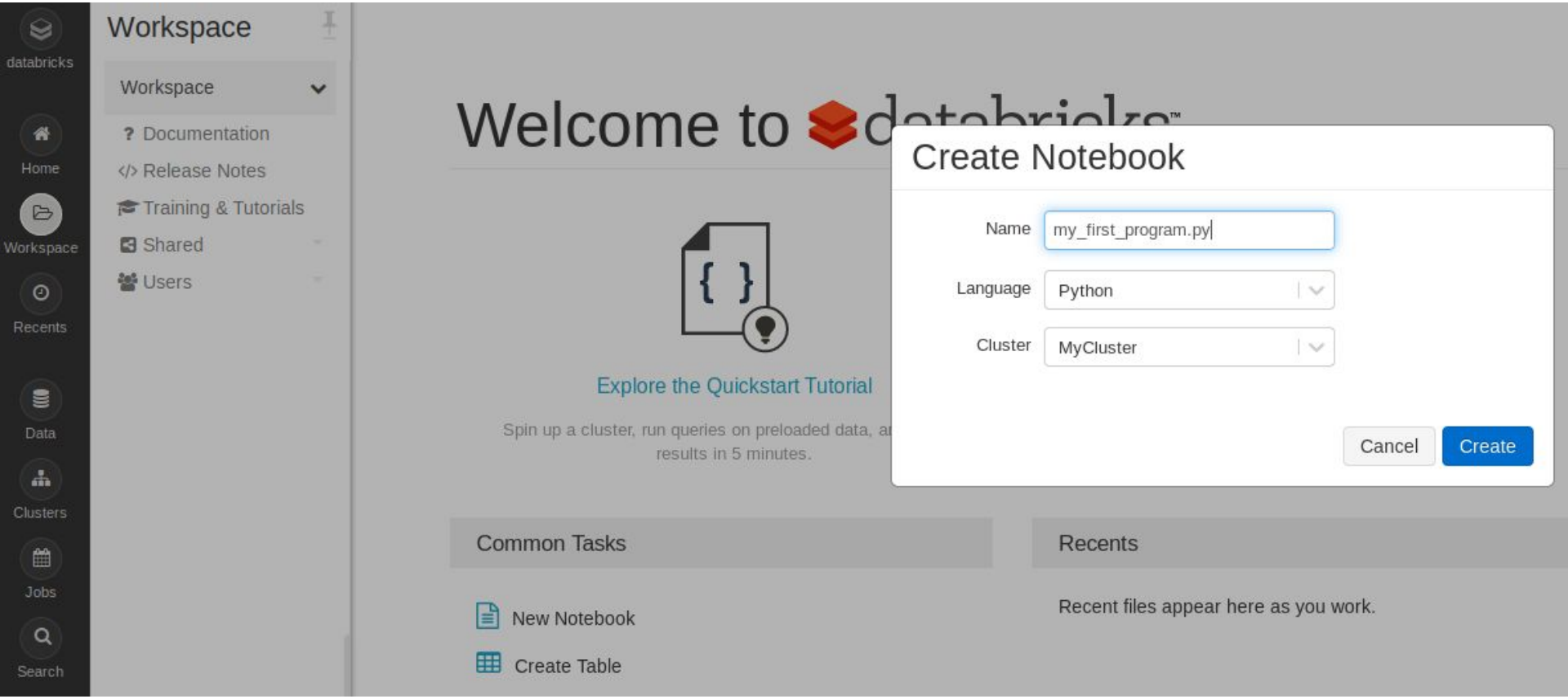
Databricks: An Online Platform for Data Engineers

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



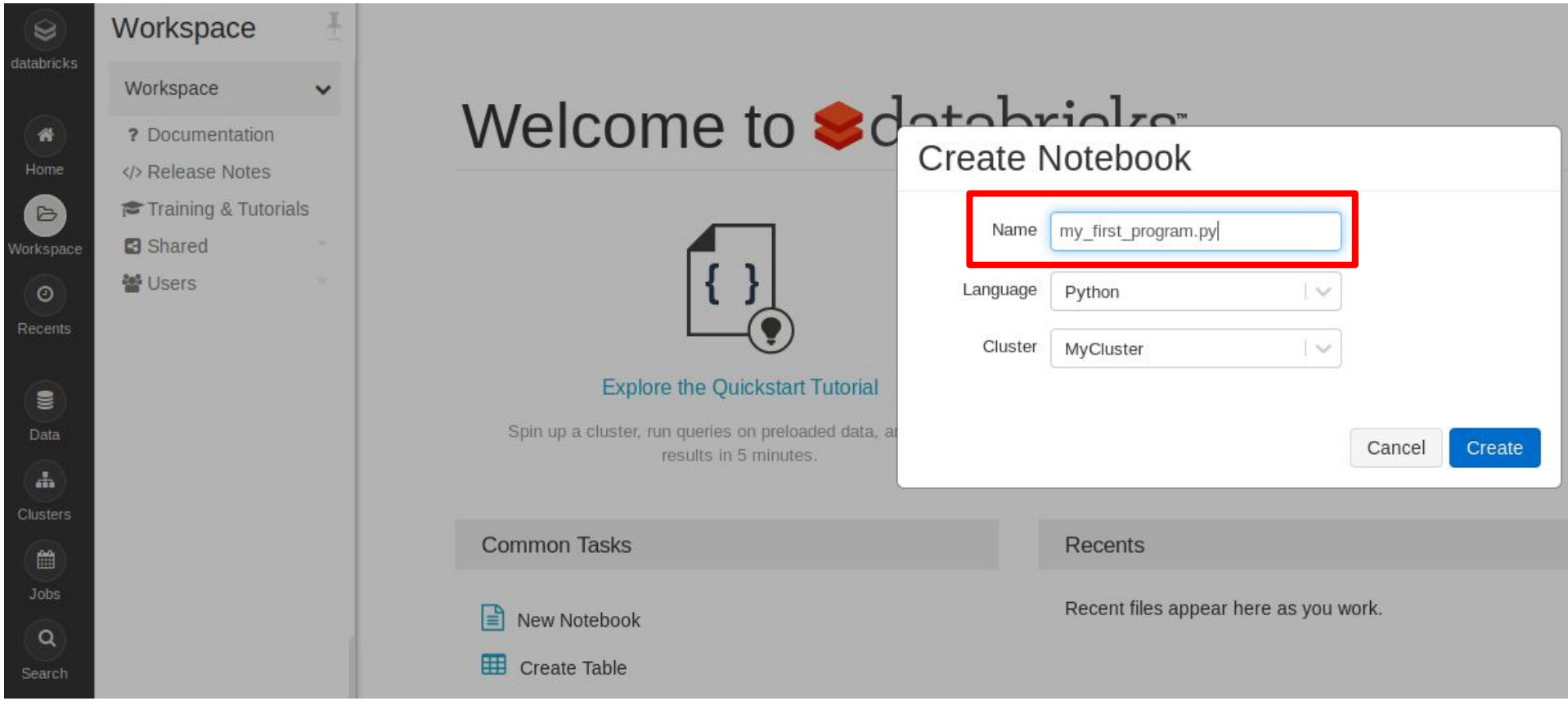
Databricks: An Online Platform for Data Engineers

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



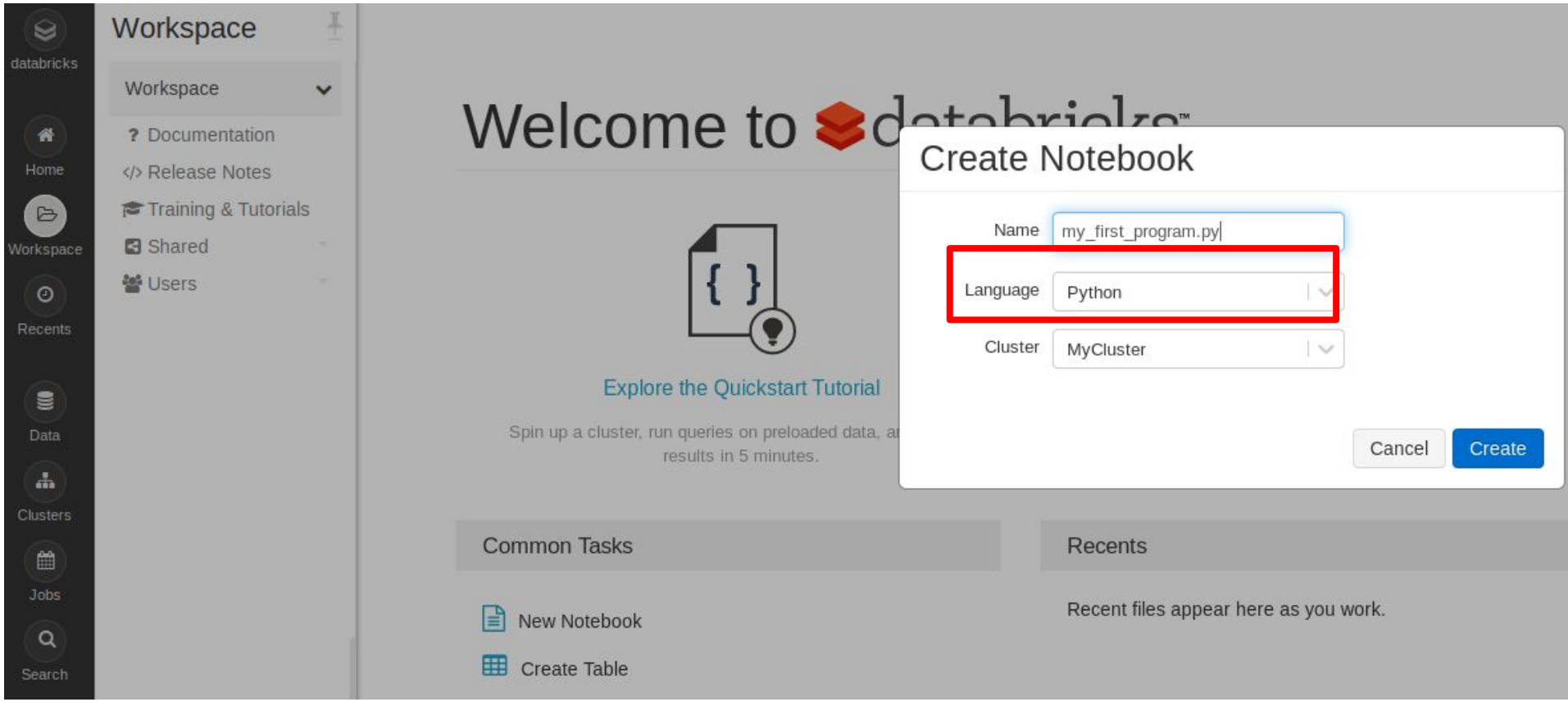
Databricks: An Online Platform for Data Engineers

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



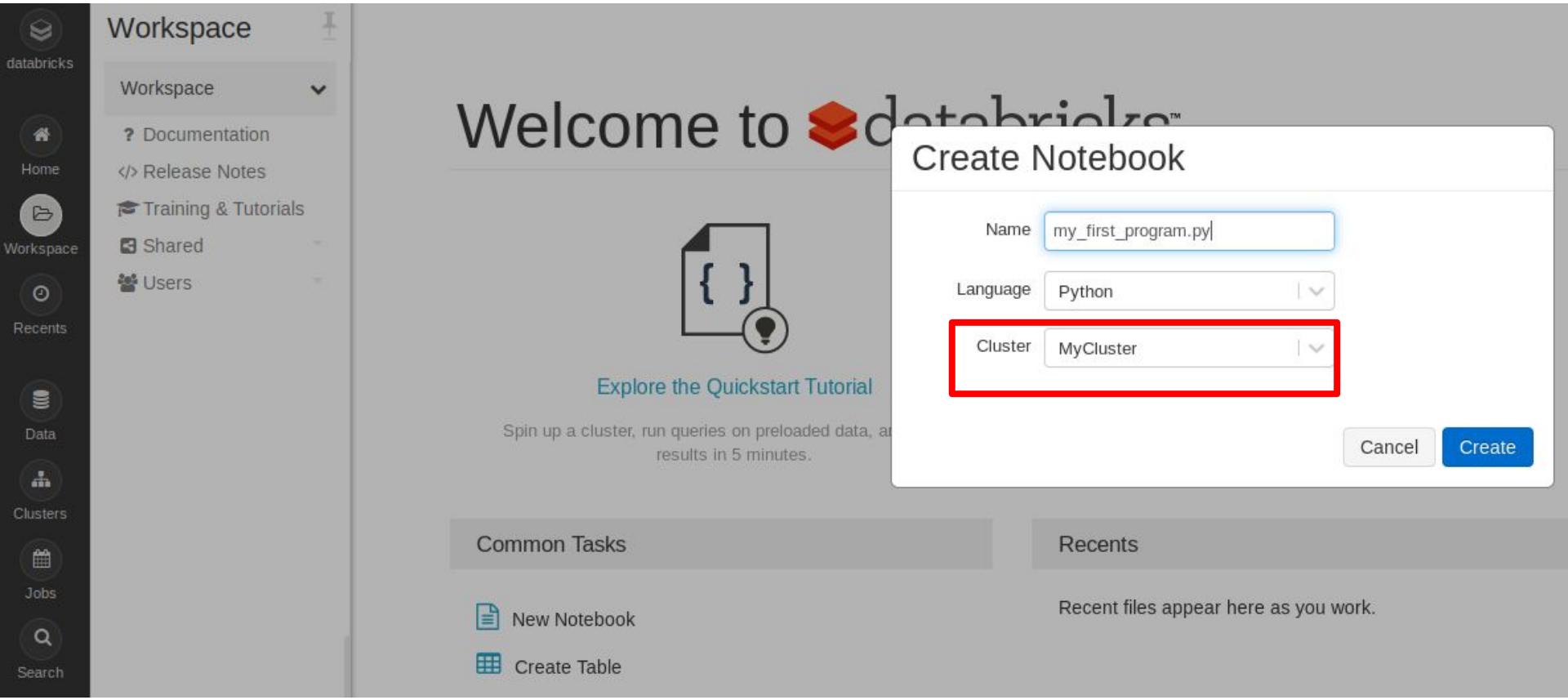
Databricks: An Online Platform for Data Engineers

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



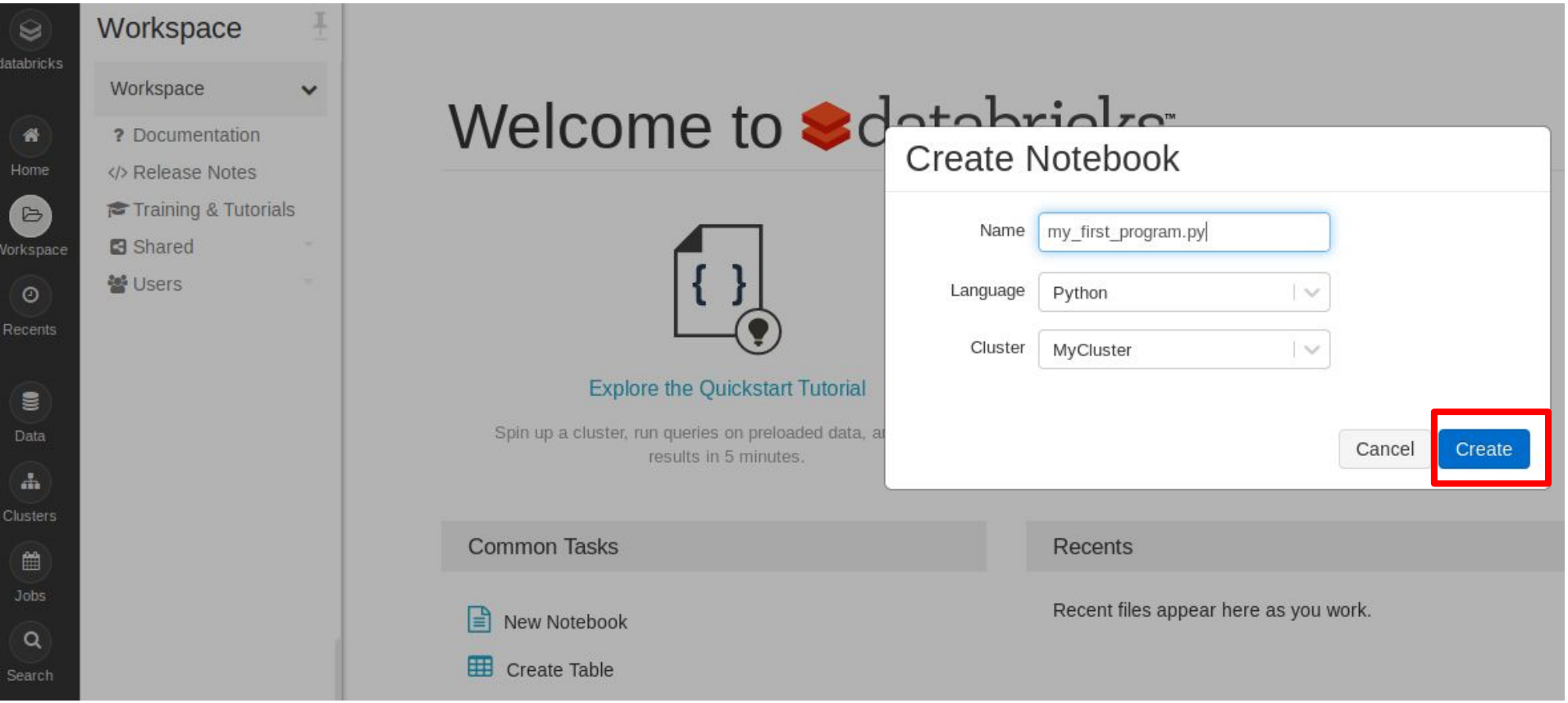
Databricks: An Online Platform for Data Engineers

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



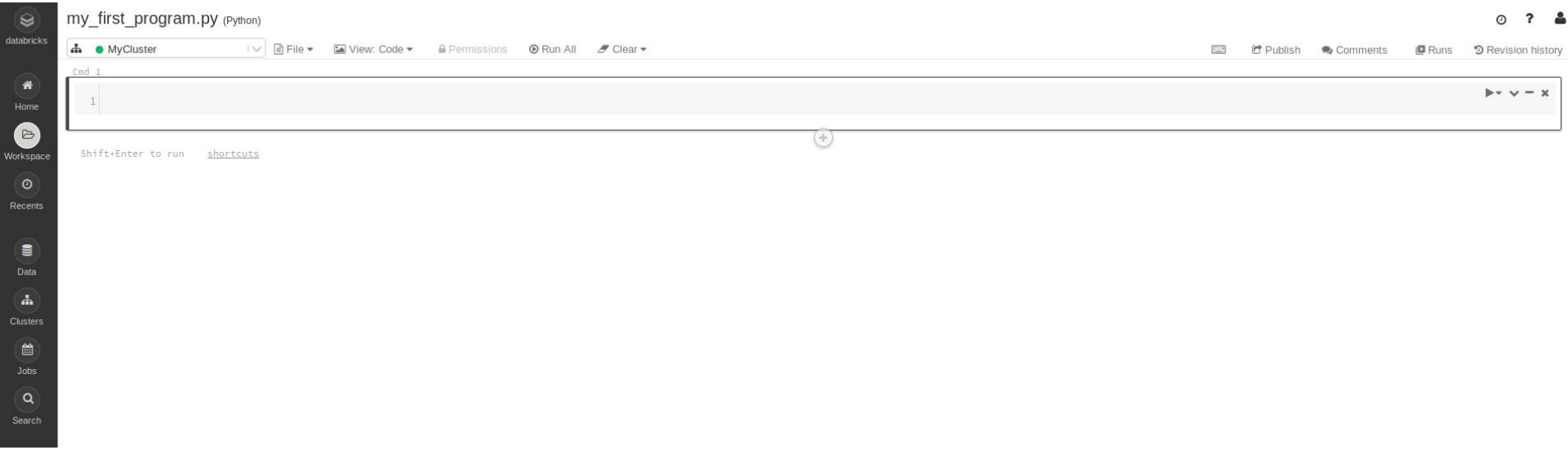
Databricks: An Online Platform for Data Engineers

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

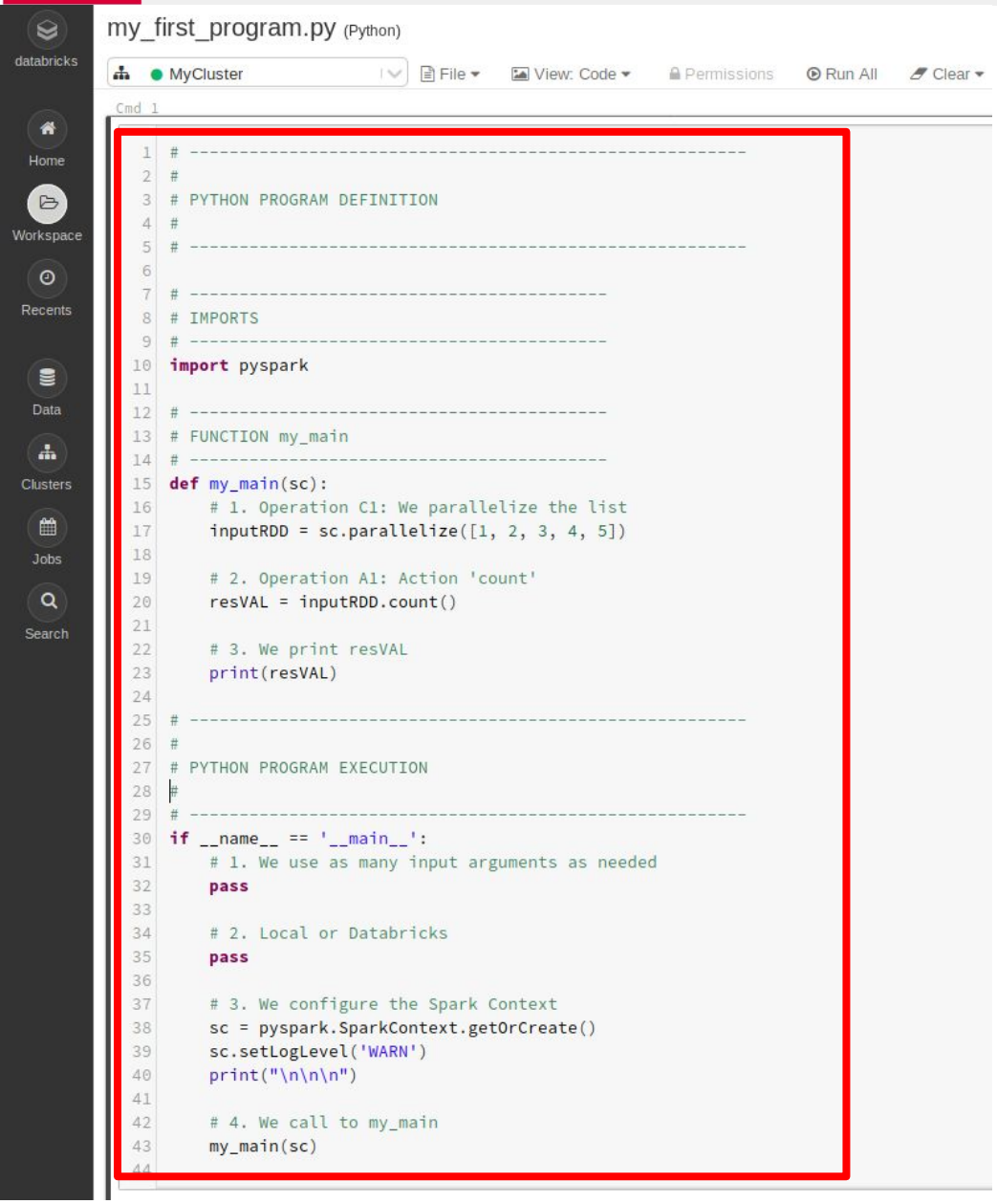


Databricks: An Online Platform for Data Engineers

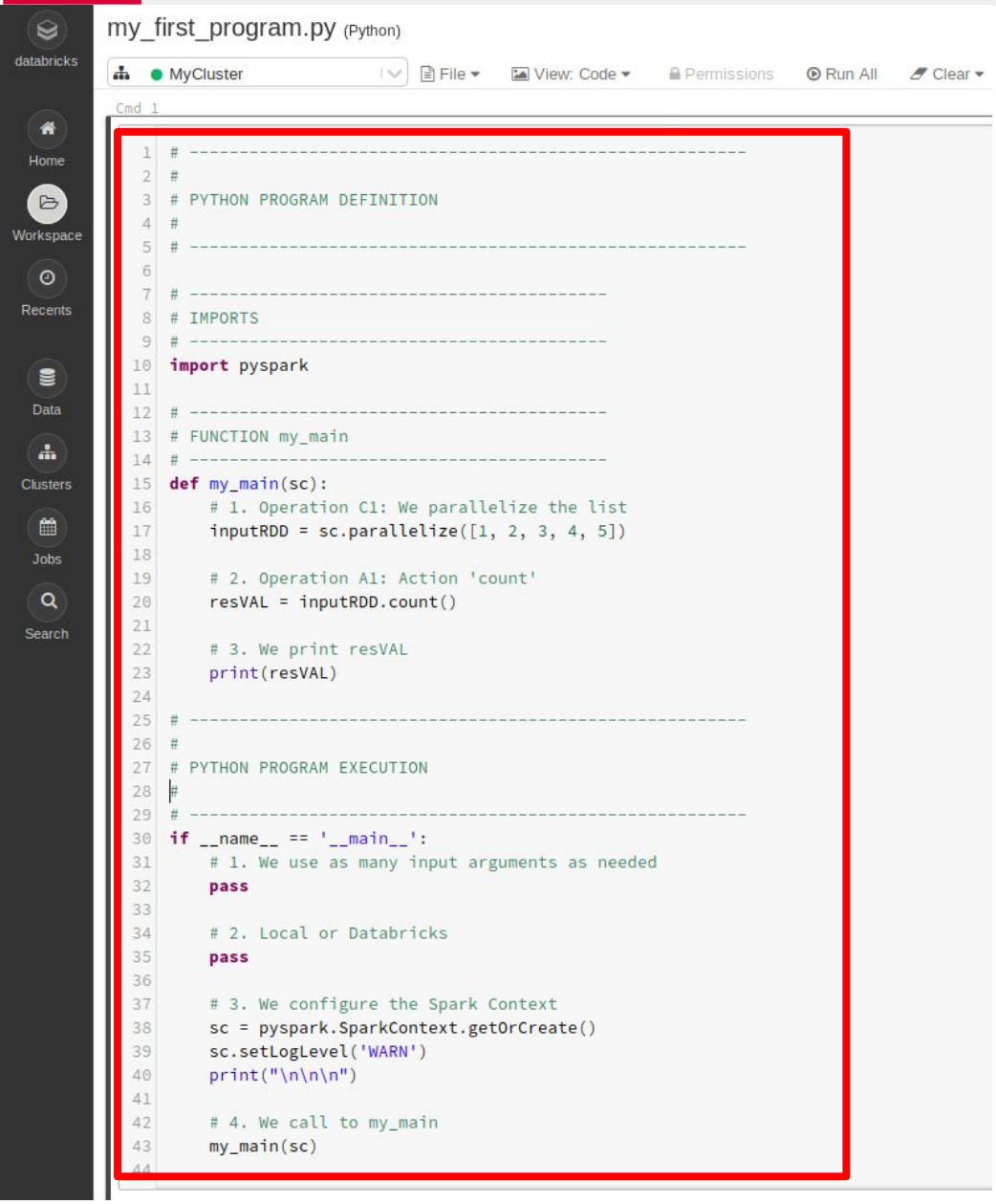
4. We are redirected to the main web page to edit our notebook.
   As we can see our program called **my_first_program.py** is automatically attached to our cluster **MyCluster**. Otherwise we can attach it by ourselves.

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.



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



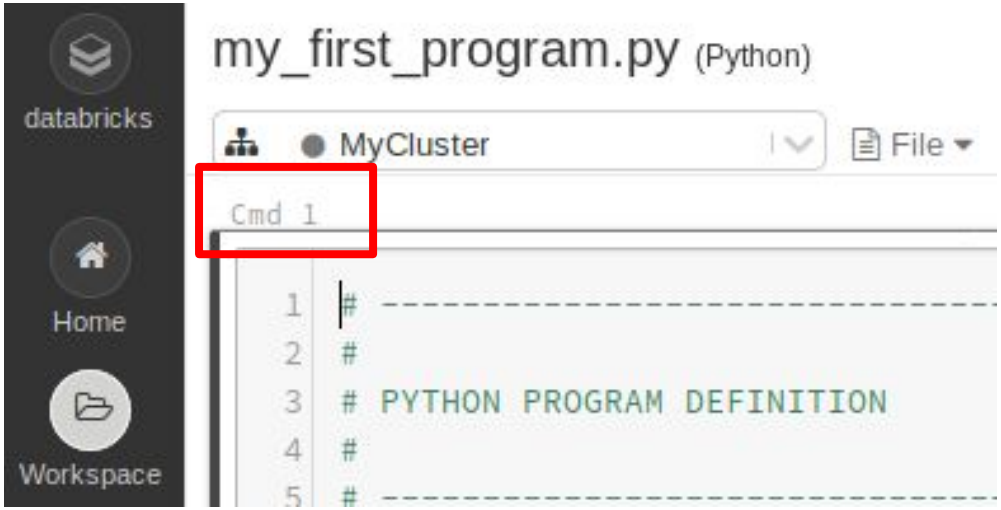
4. We fill in our program in the Jupiter Notebook being created.

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



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...

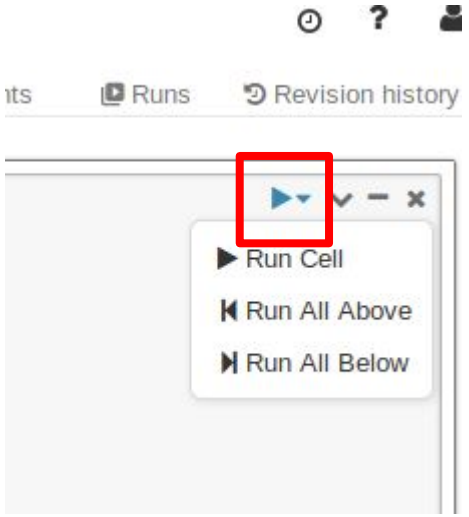
Run the new program / Spark Application.

Databricks: An Online Platform for Data Engineers

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

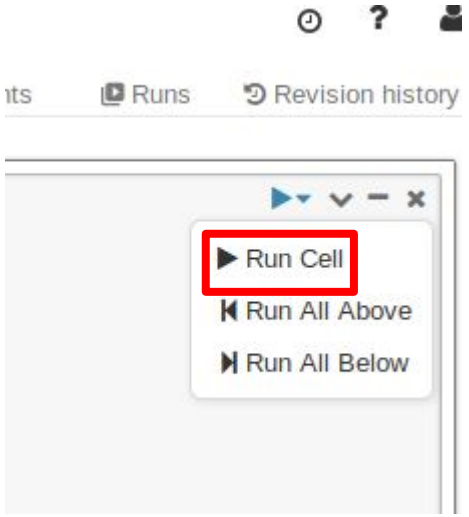
Databricks: An Online Platform for Data Engineers

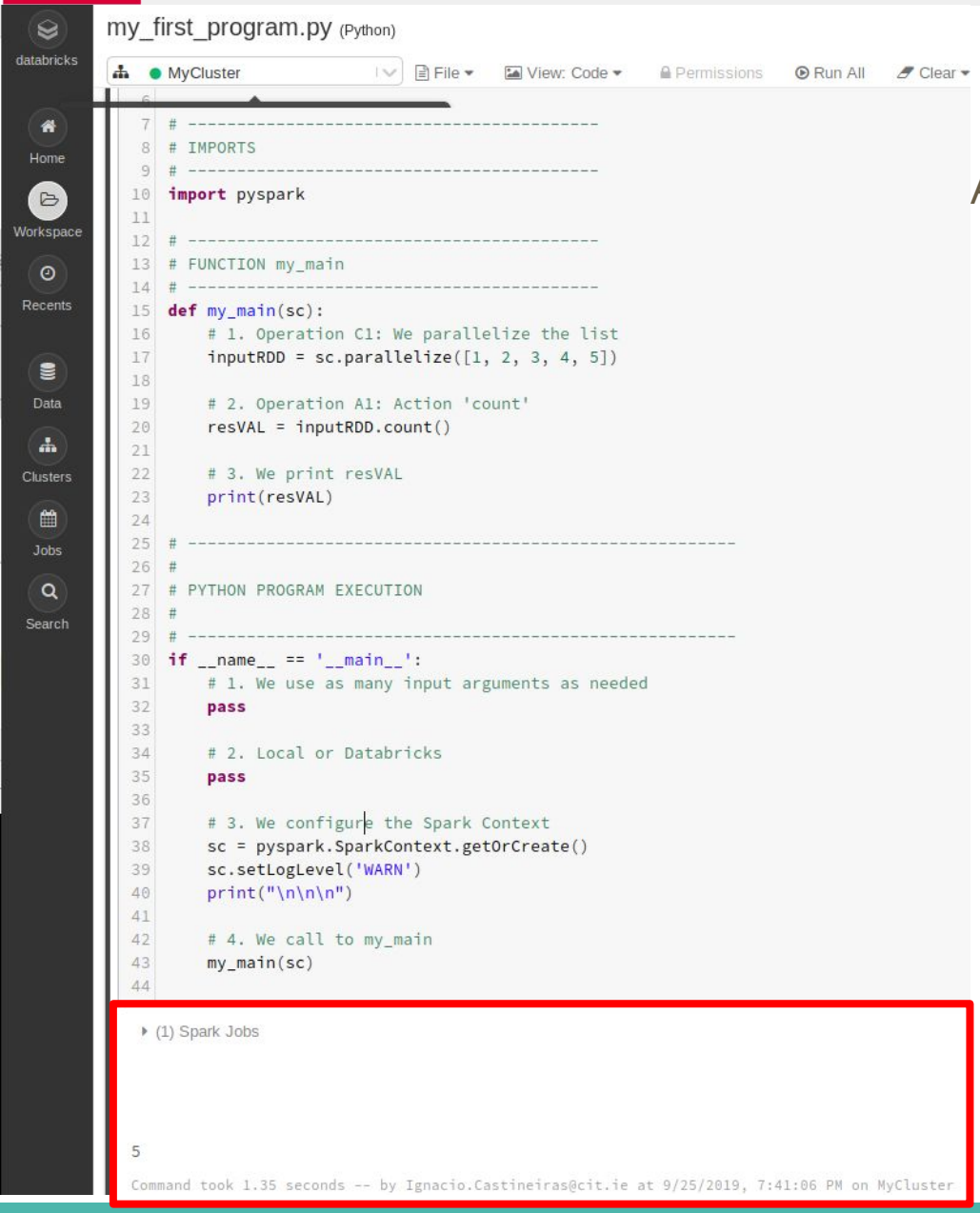
5. Once the program (Spark application) is ready, we can execute it in our cluster.



Databricks: An Online Platform for Data Engineers

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





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
```

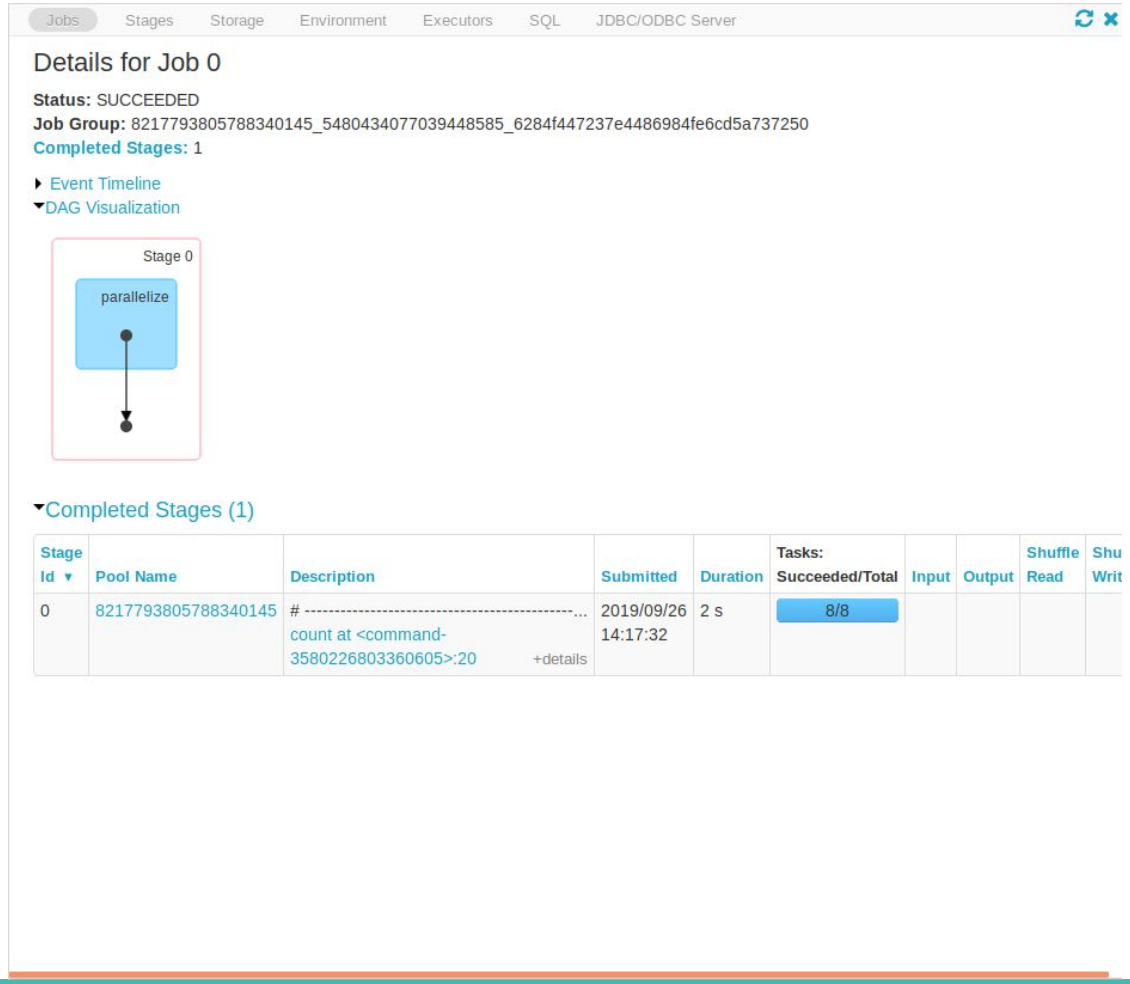
▾ (1) Spark Jobs
▸ Job 0 View (Stages: 1/1)

5

Command took 2.74 seconds -- by Ignacio.Castineiras@cit.ie at 9/26/2019, 3:17:32 PM on MyCluster

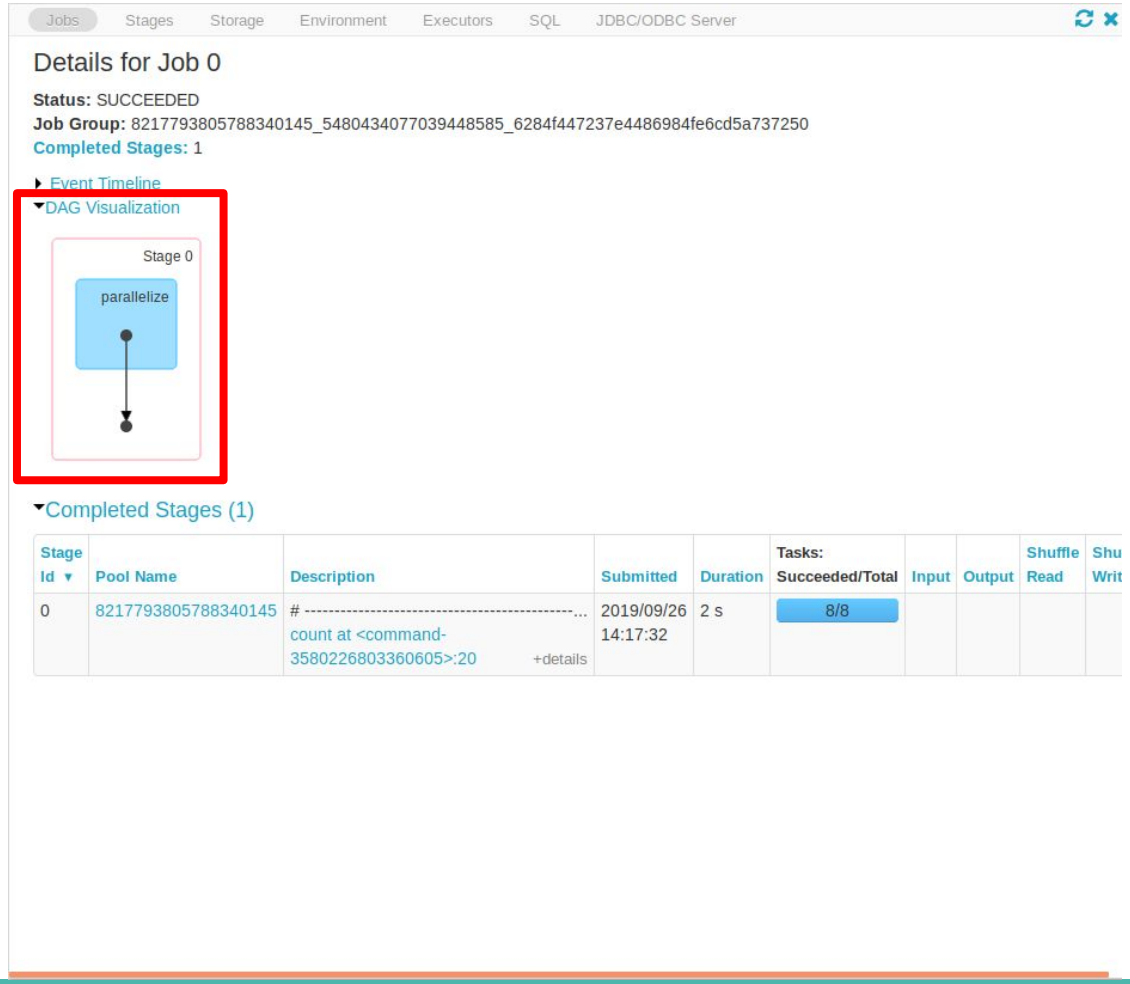
Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.
   This includes inspecting the DAG and the Task Scheduling.



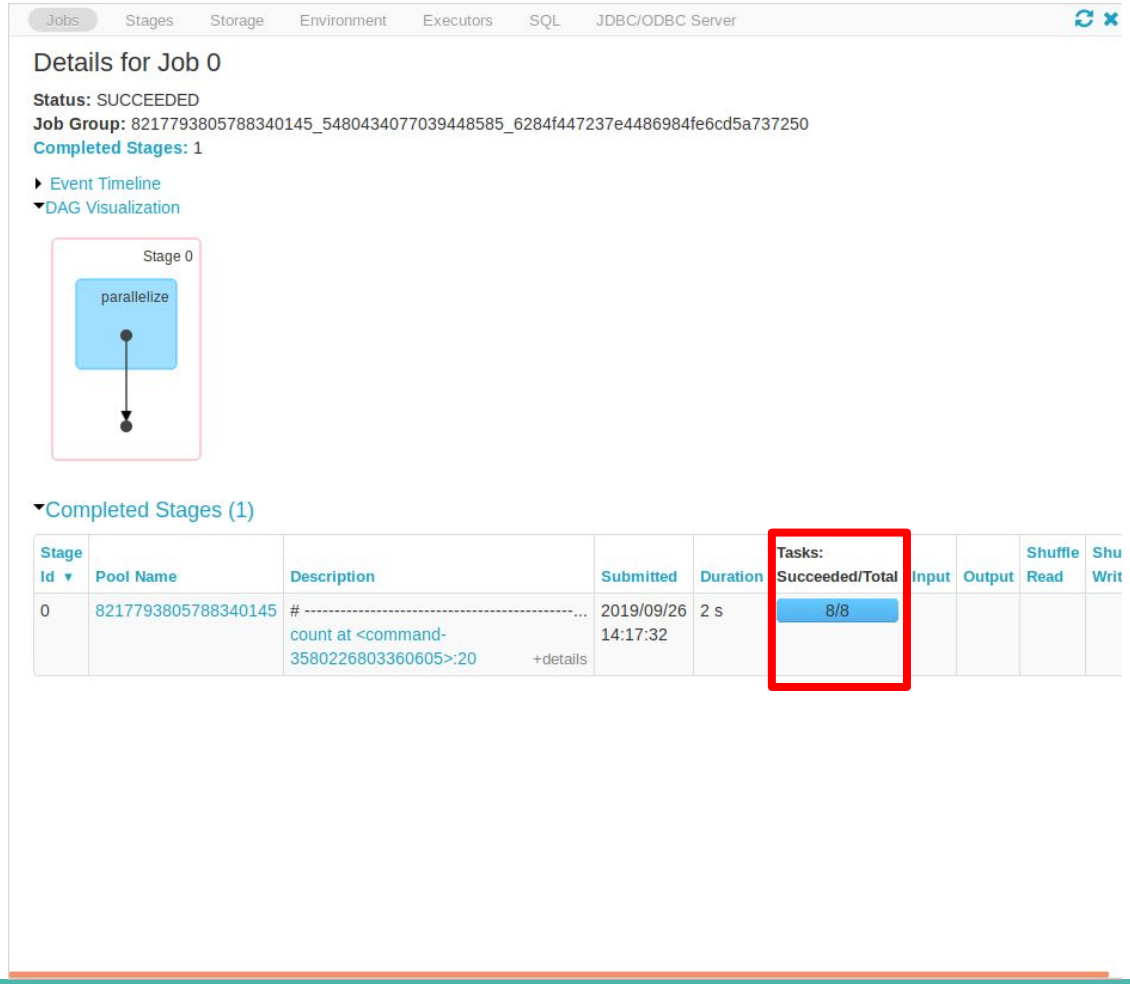
Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.
   This includes inspecting the **DAG** and the Task Scheduling.



Databricks: An Online Platform for Data Engineers

5. And we can evaluate its result.
   This includes inspecting the DAG and the **Task Scheduling**.



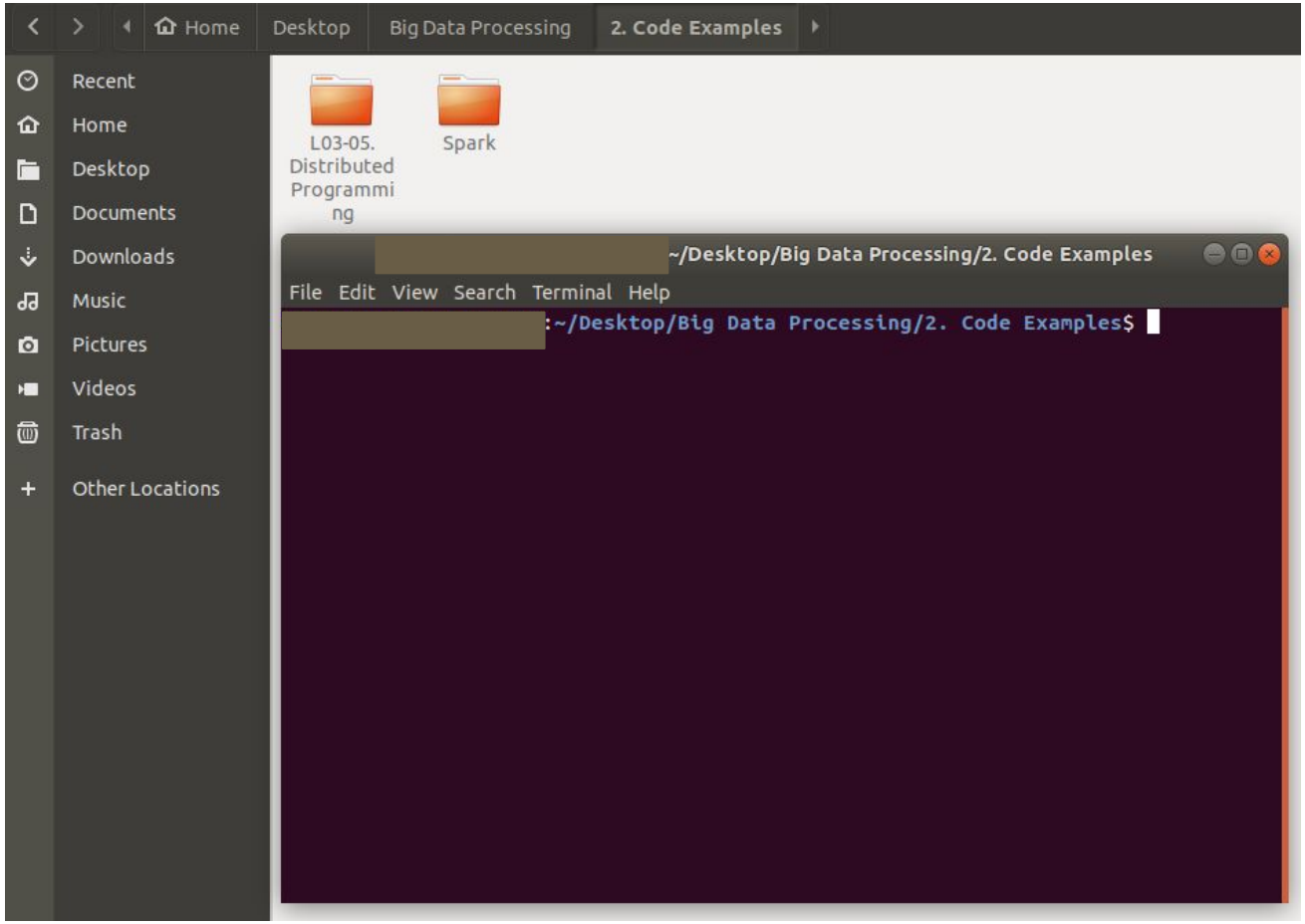
Databricks: An Online Platform for Data Engineers

How to...

Install the Databricks Command Line Application (CLI)

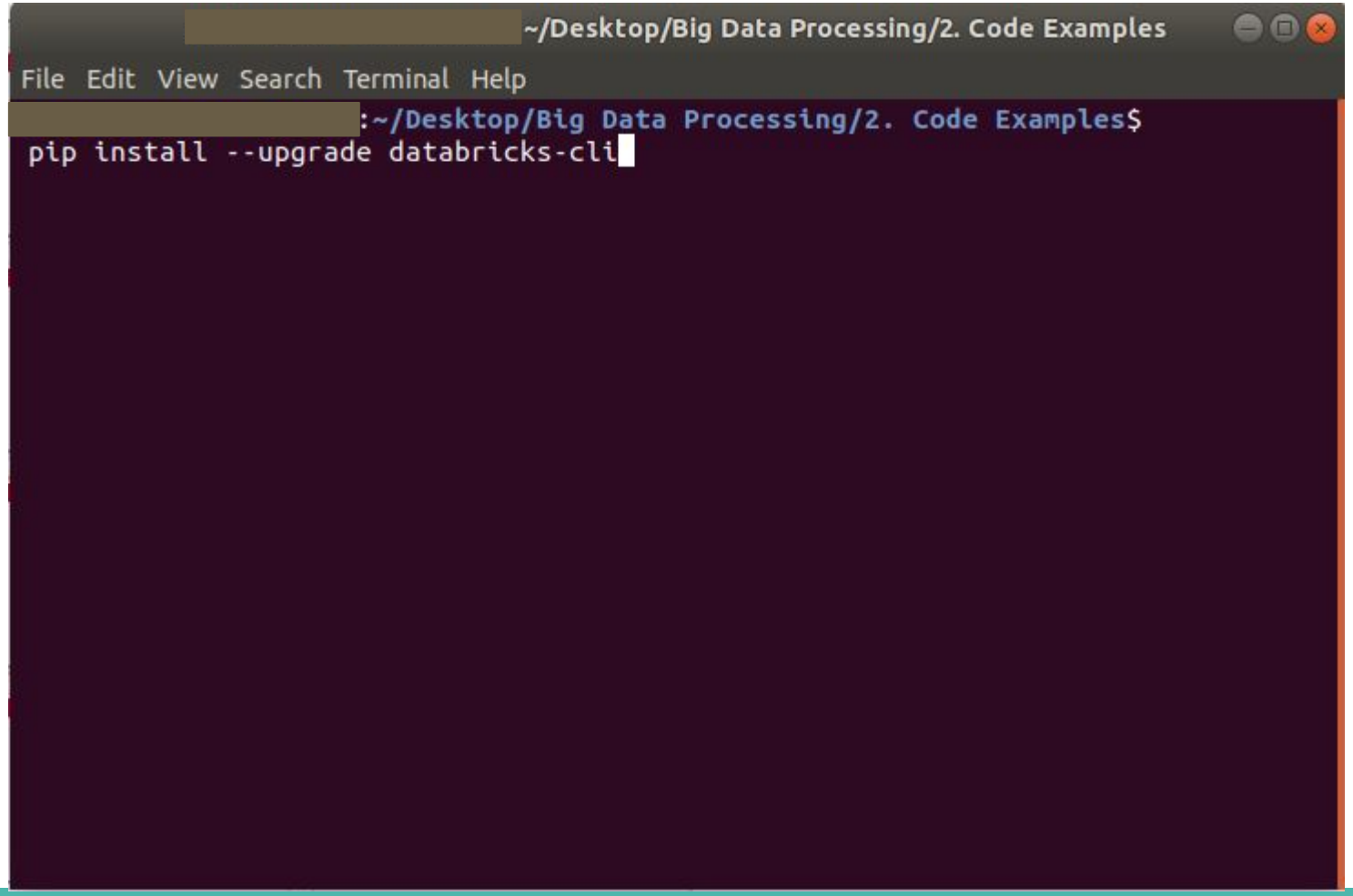
Databricks: An Online Platform for Data Engineers

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.



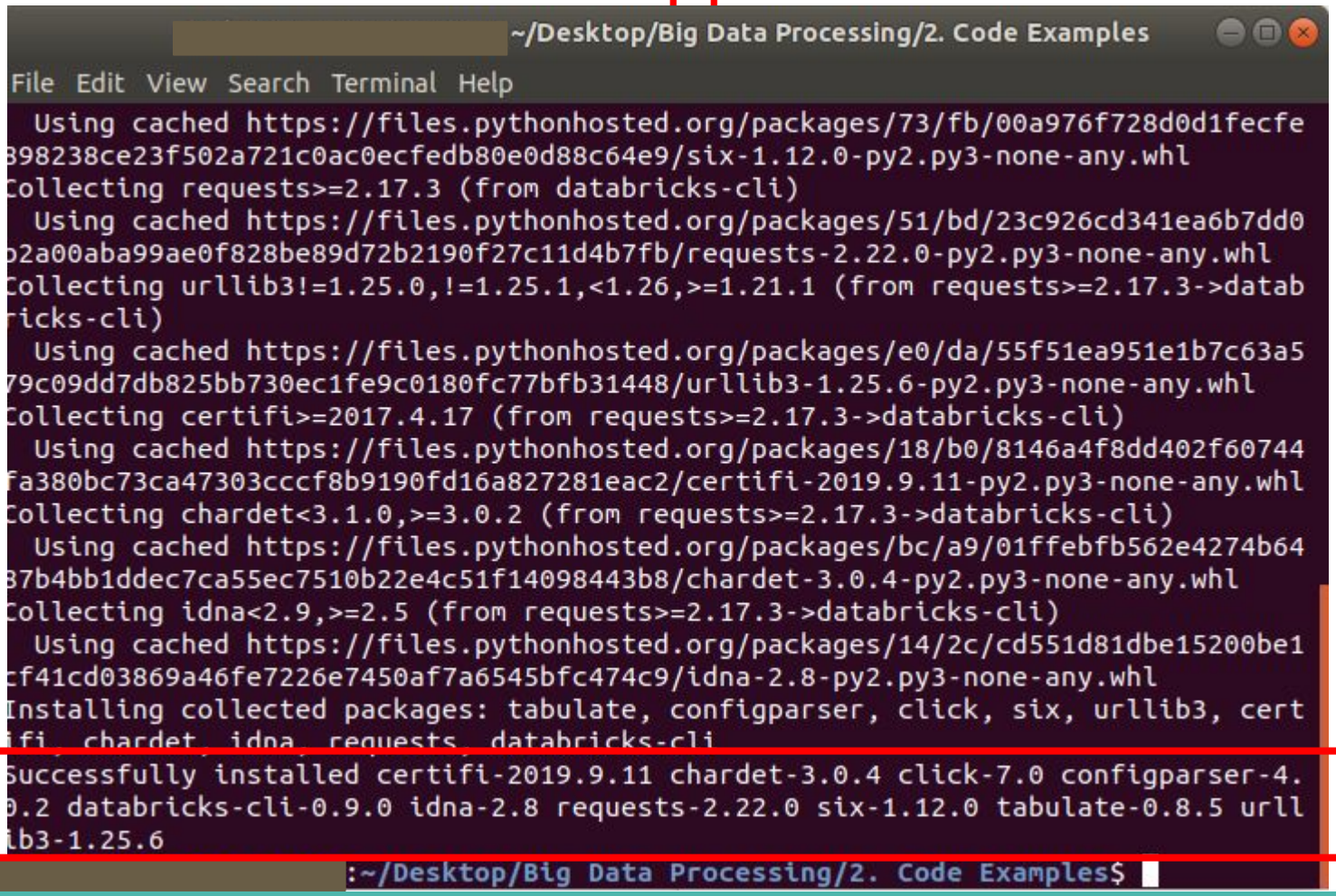
Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
   We can install the Databricks CLI via **pip**.



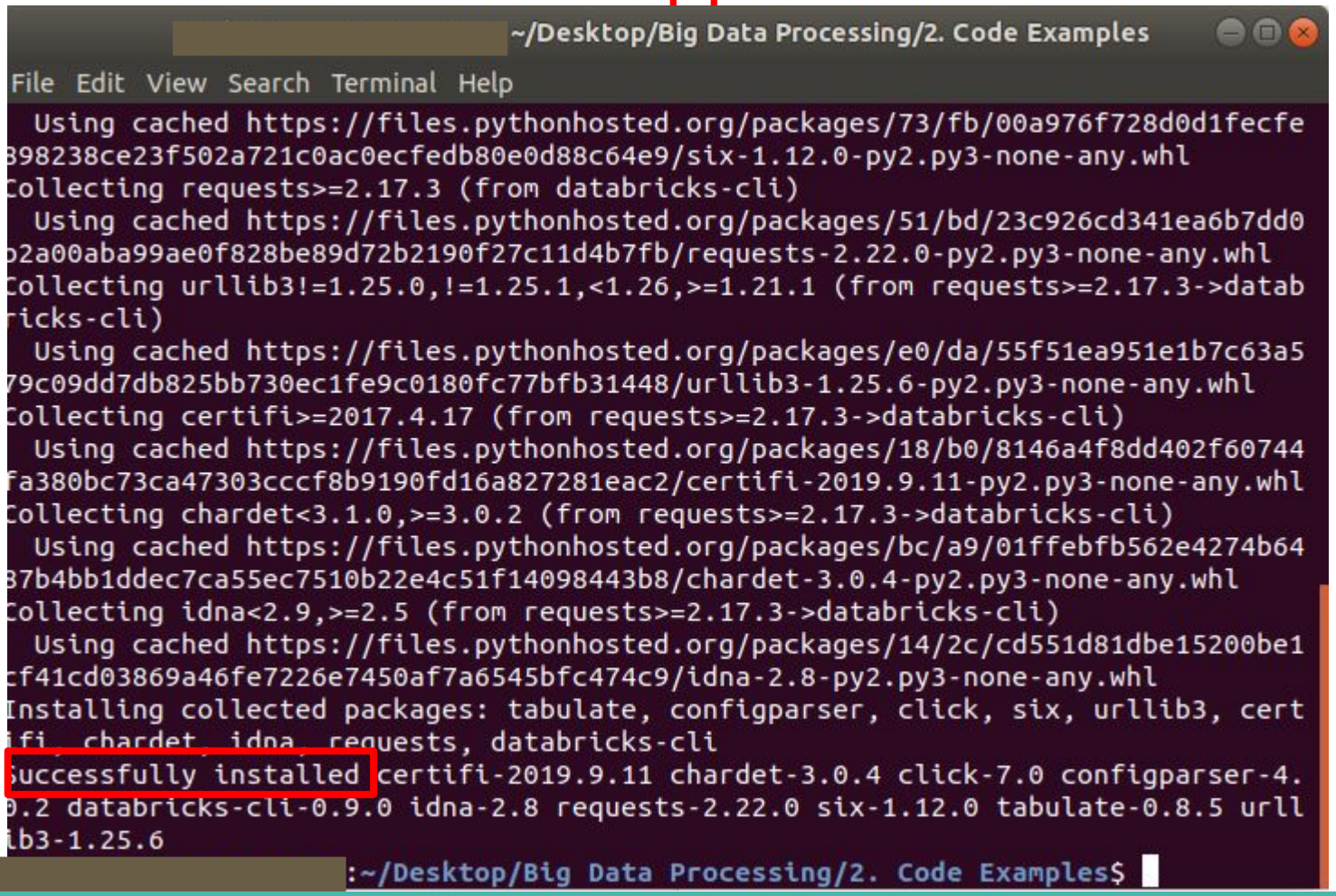
Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
   We can install the Databricks CLI via **pip**.



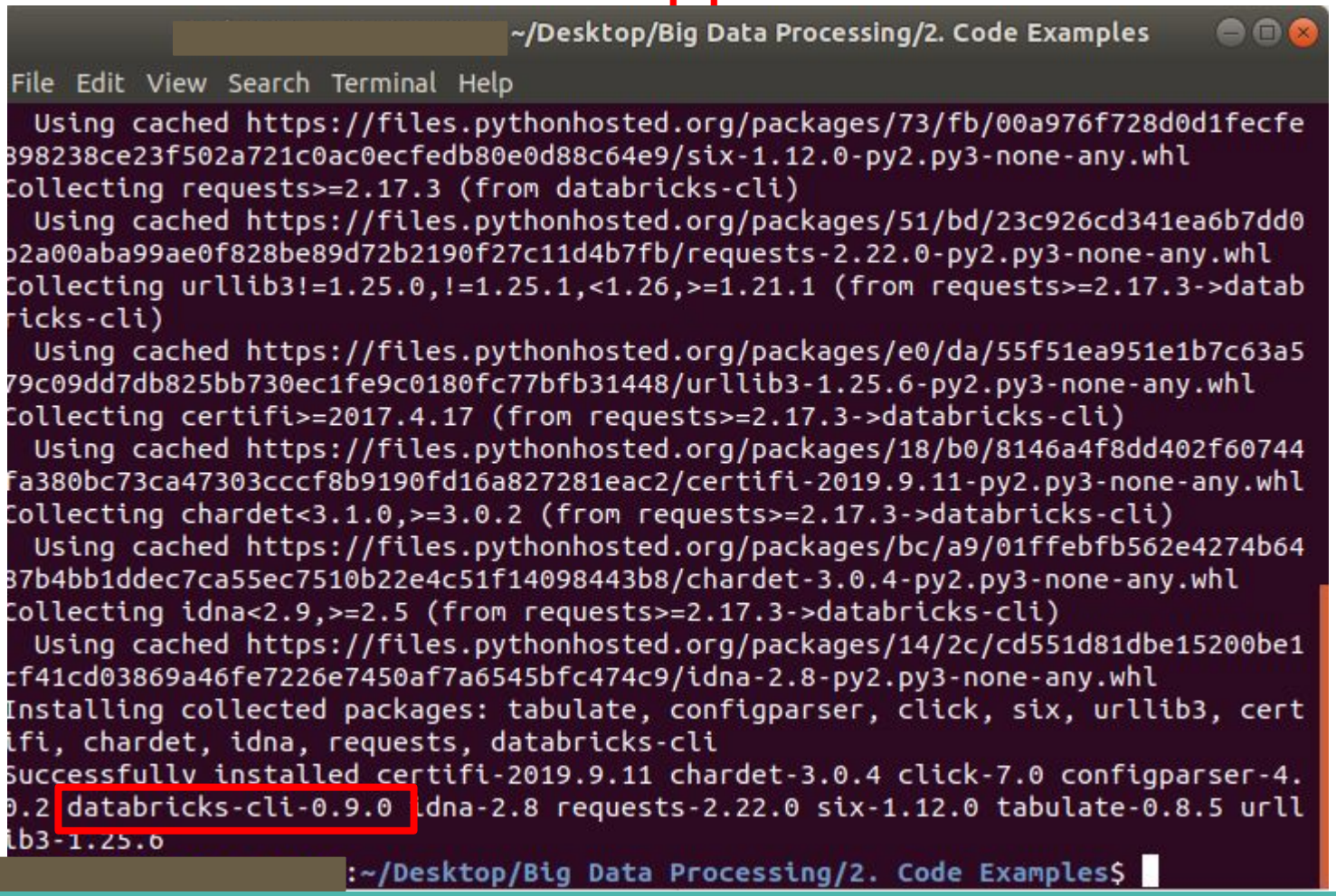
Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
   We can install the Databricks CLI via **pip**.



Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
   We can install the Databricks CLI via **pip**.

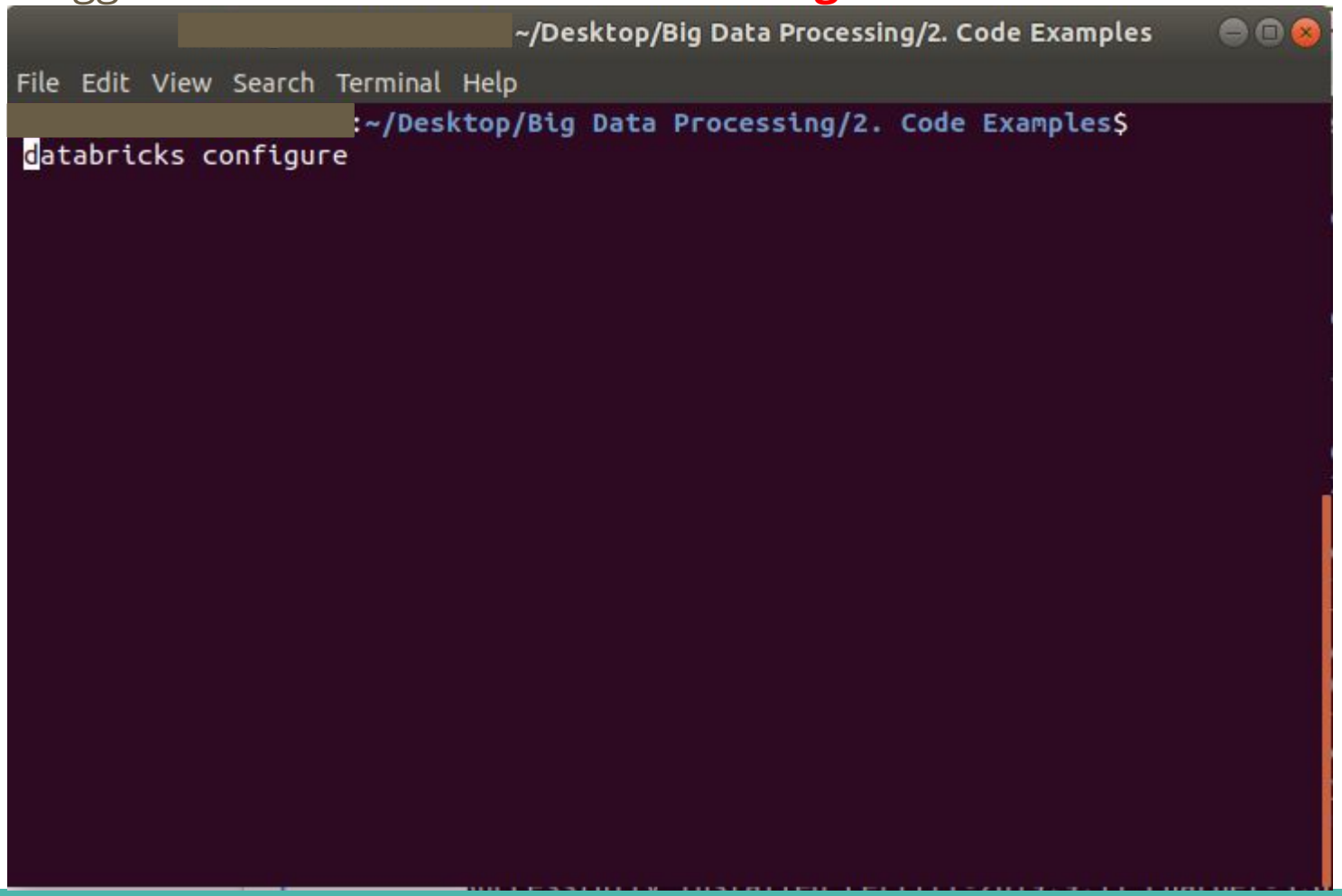


Databricks: An Online Platform for Data Engineers

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

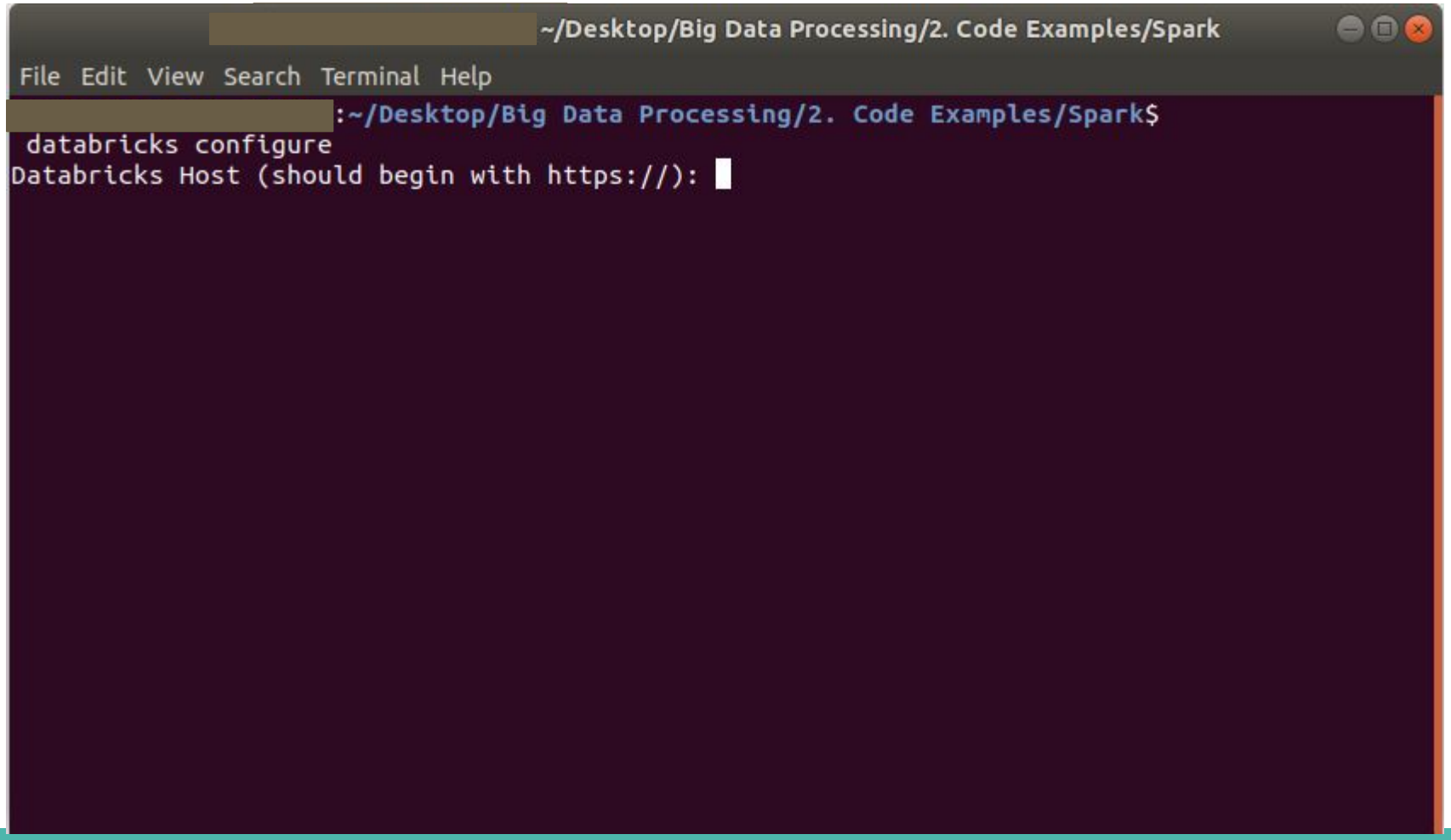
Databricks: An Online Platform for Data Engineers

6. Databricks provides a Command Line Interface (CLI).
   We trigger the command **databricks configure**



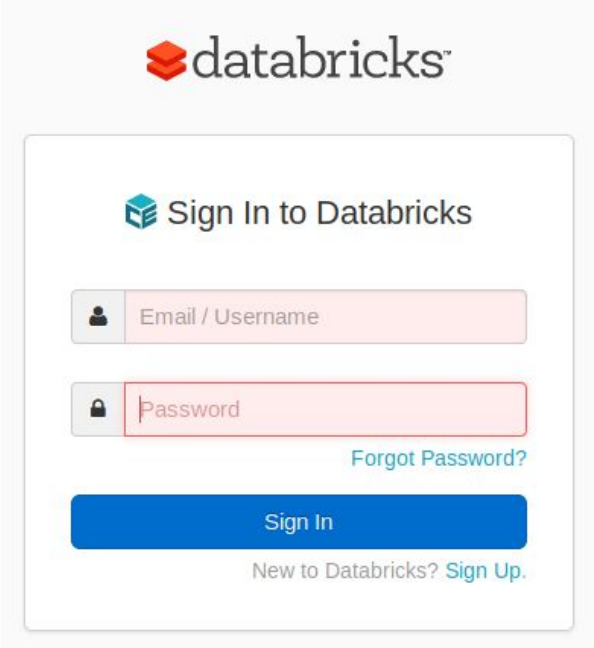
Databricks: An Online Platform for Data Engineers

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



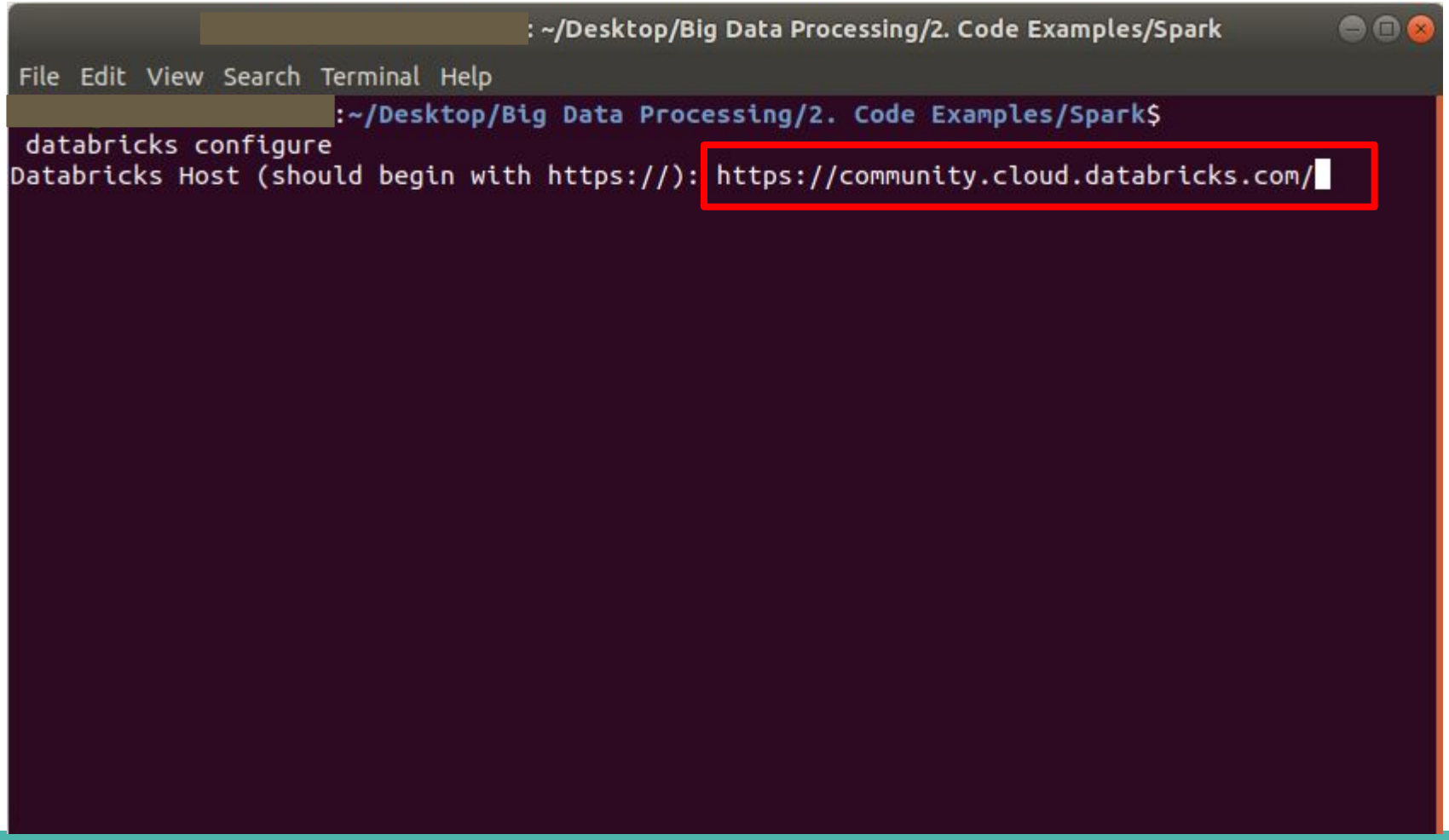
Databricks: An Online Platform for Data Engineers

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



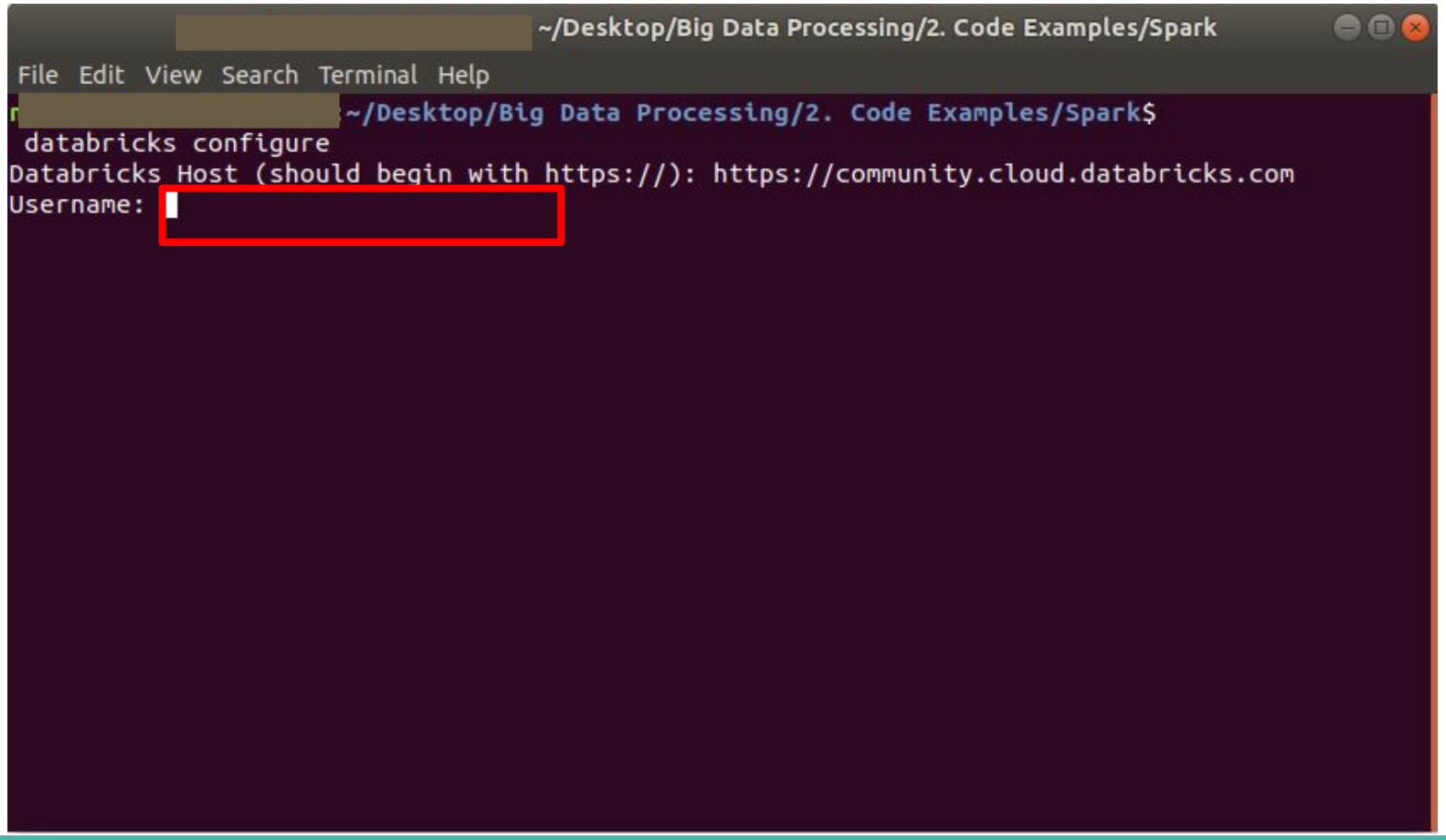
Databricks: An Online Platform for Data Engineers

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



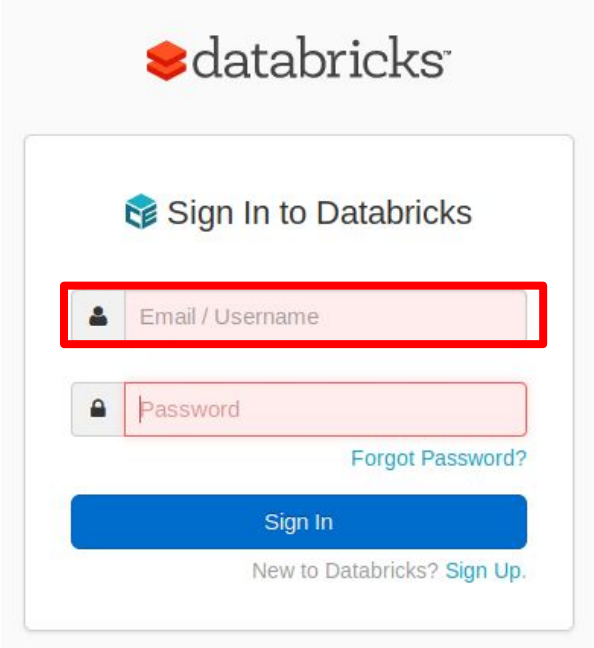
Databricks: An Online Platform for Data Engineers

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).



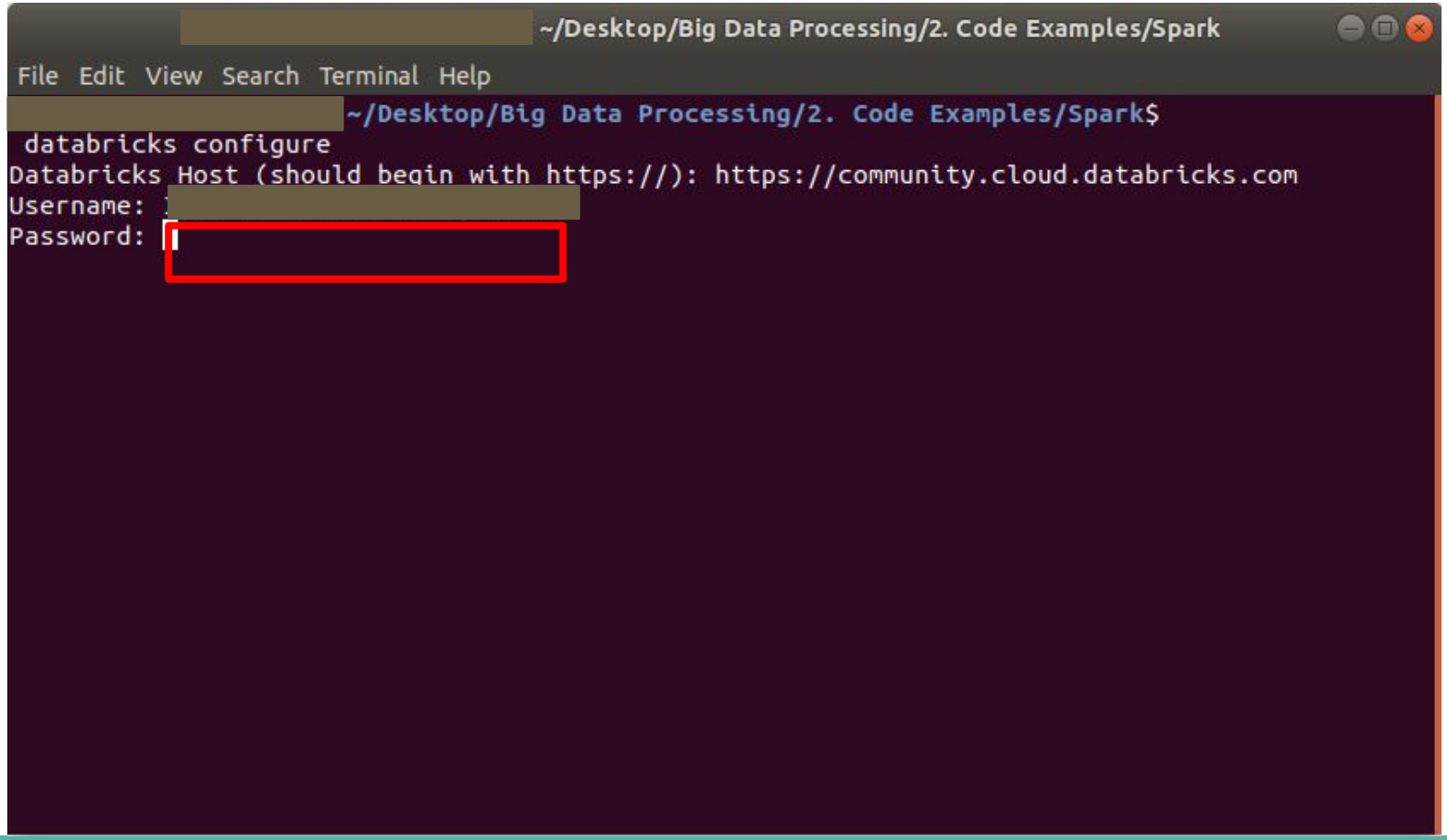
Databricks: An Online Platform for Data Engineers

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



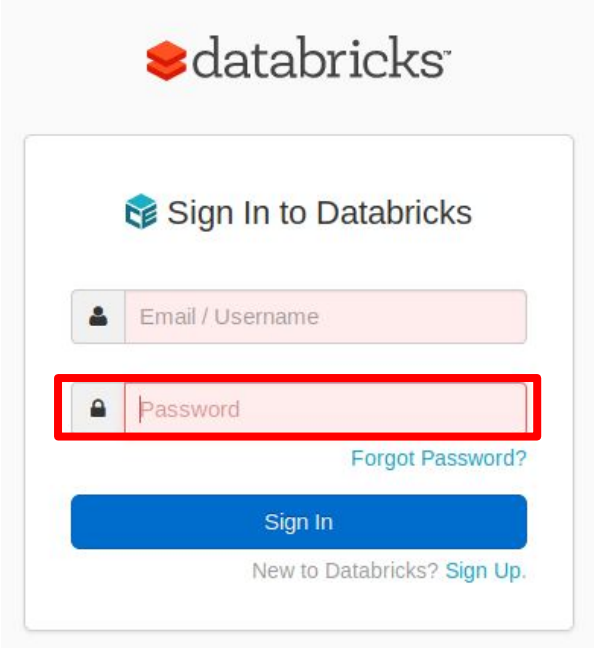
Databricks: An Online Platform for Data Engineers

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).



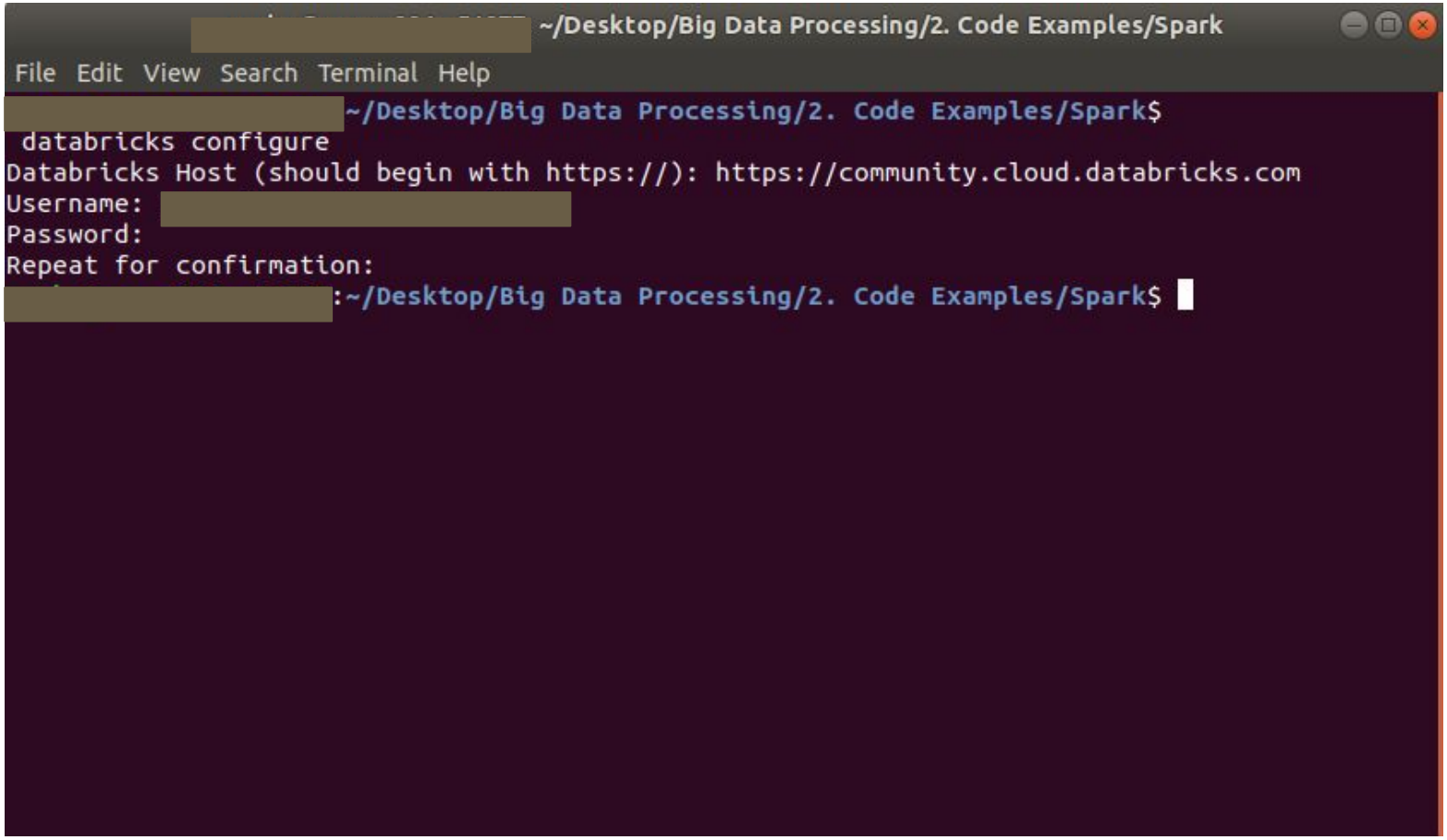
Databricks: An Online Platform for Data Engineers

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).



Databricks: An Online Platform for Data Engineers

How to...

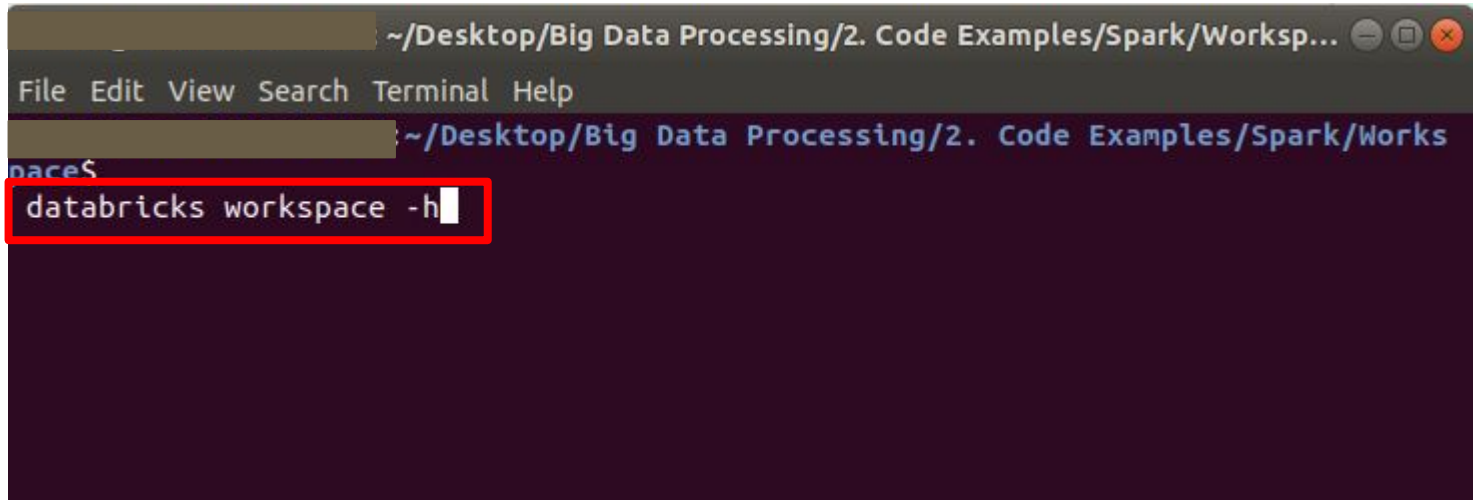
Upload an entire directory
from our local machine
to our Databricks Workspace

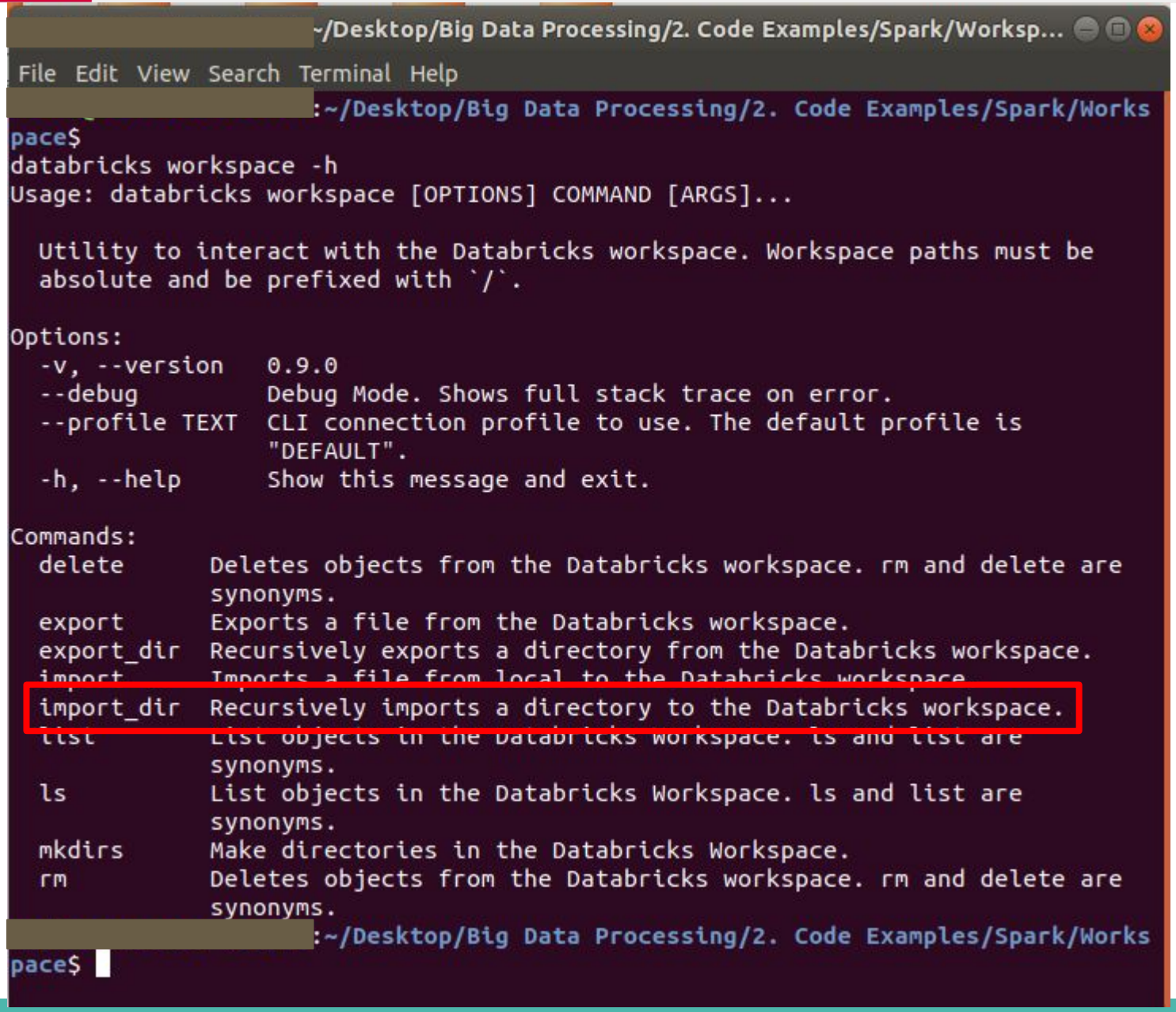
Databricks: An Online Platform for Data Engineers

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

Databricks: An Online Platform for Data

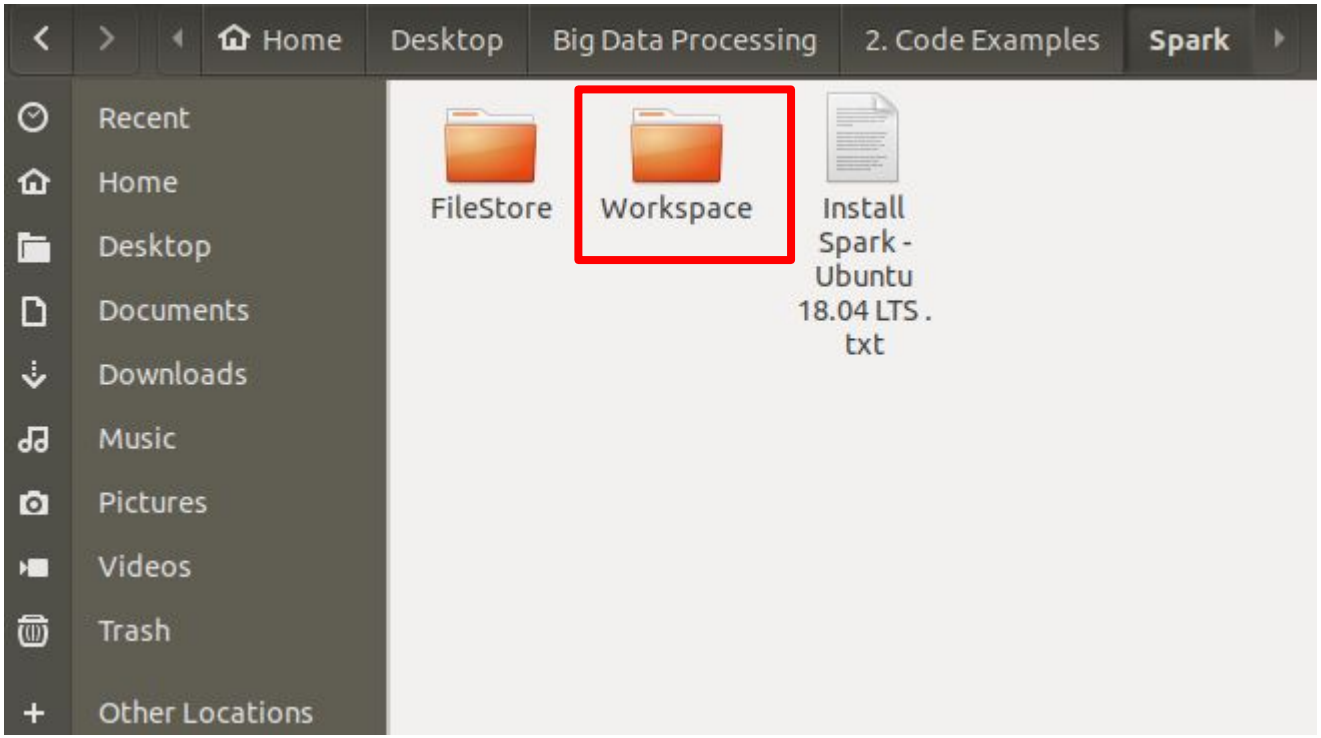
7. The Databricks CLI has some **workspace** specific commands:




~/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_dir Recursively imports a directory to the Databricks workspace.
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.
:~/Desktop/Big Data Processing/2. Code Examples/Spark/Works
pace$

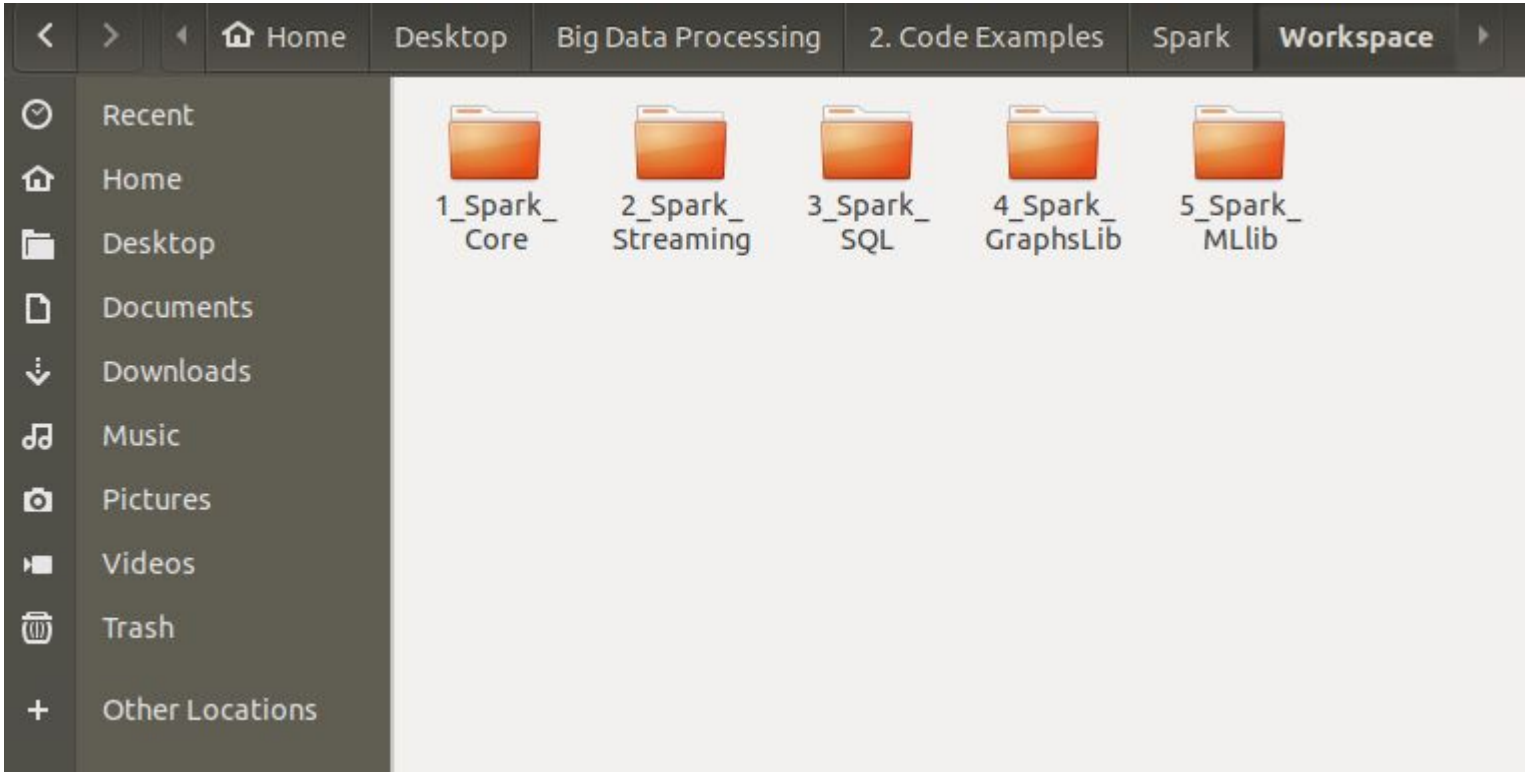
Databricks: An Online Platform for Data Engineers

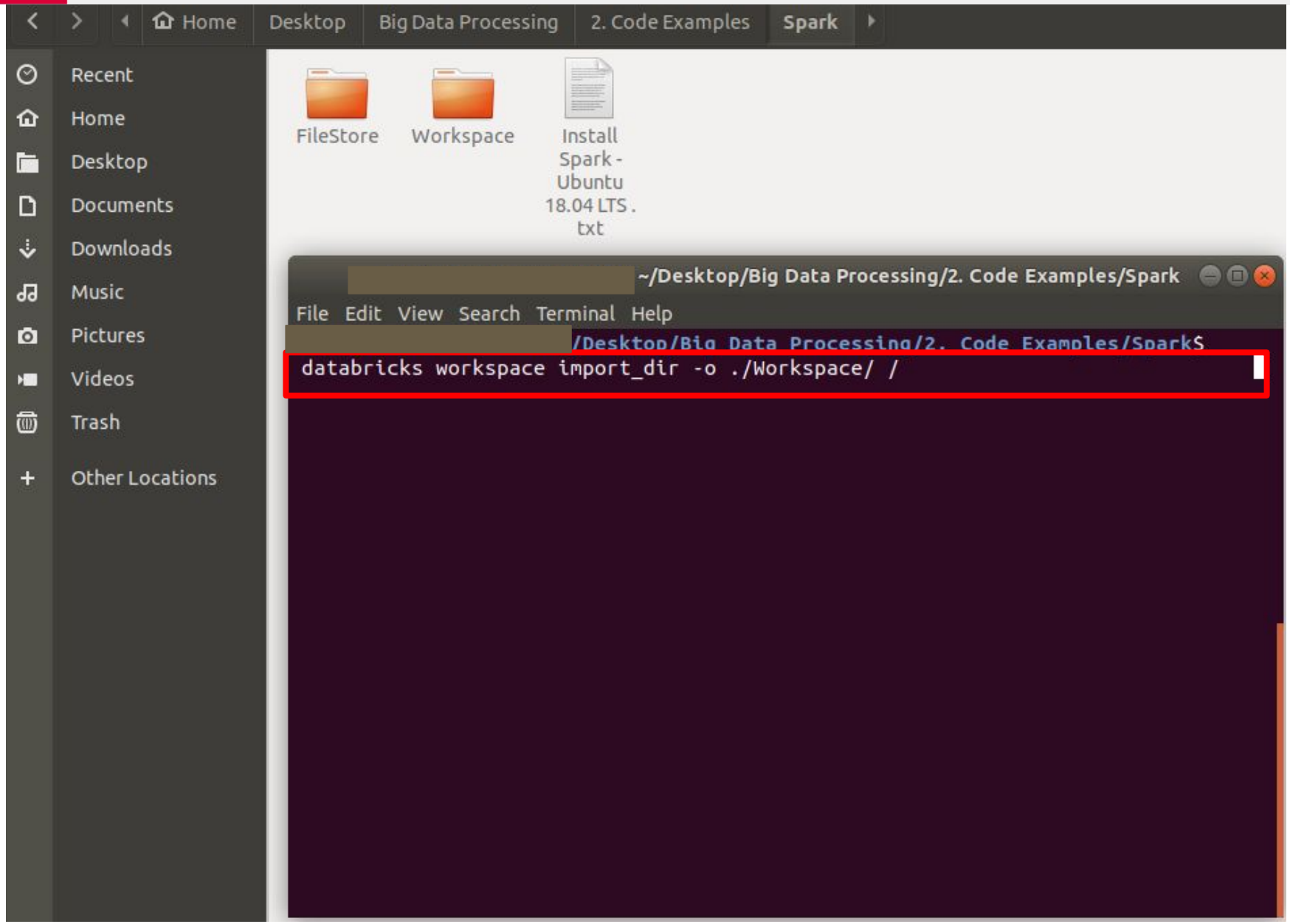
7. We can use the command **import_dir** command to upload our set of Spark code examples (available in Canvas) to our Databricks Workspace.



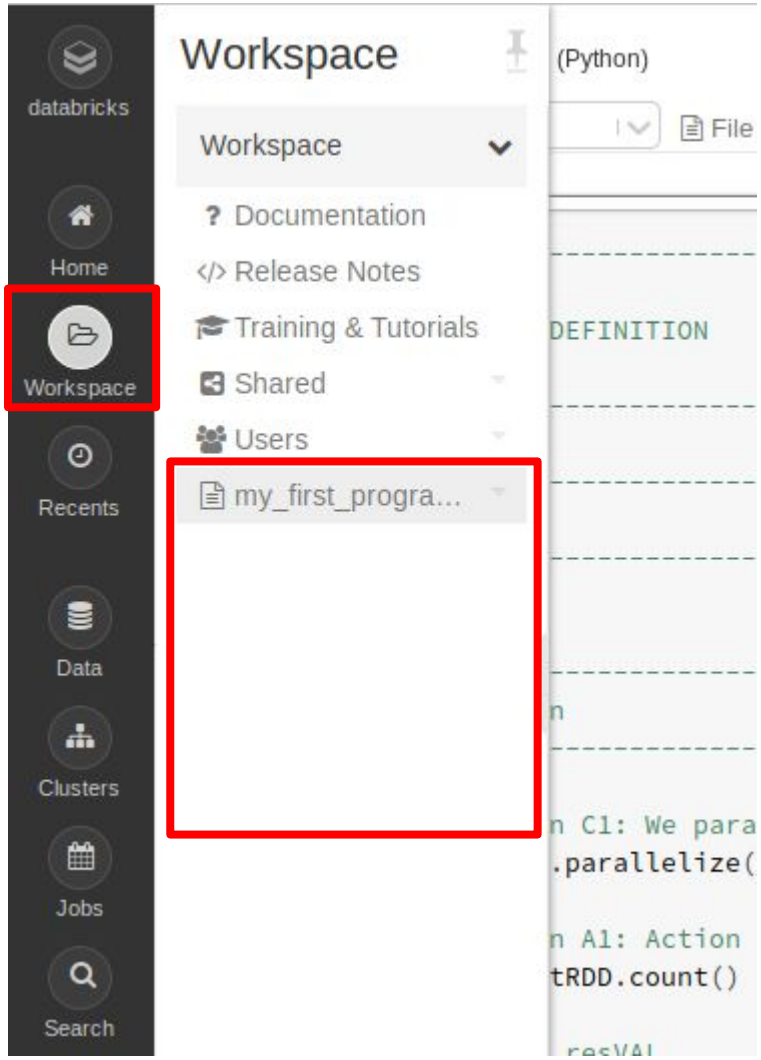
Databricks: An Online Platform for Data Engineers

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




Home
Desktop
Big Data Processing
2. Code Examples
Spark
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
FileStore
Workspace
Install Spark - Ubuntu 18.04 LTS . txt
~/Desktop/Big Data Processing/2. Code Examples/Spark
File Edit View Search Terminal Help
databricks workspace import_dir -o ./Workspace/ /

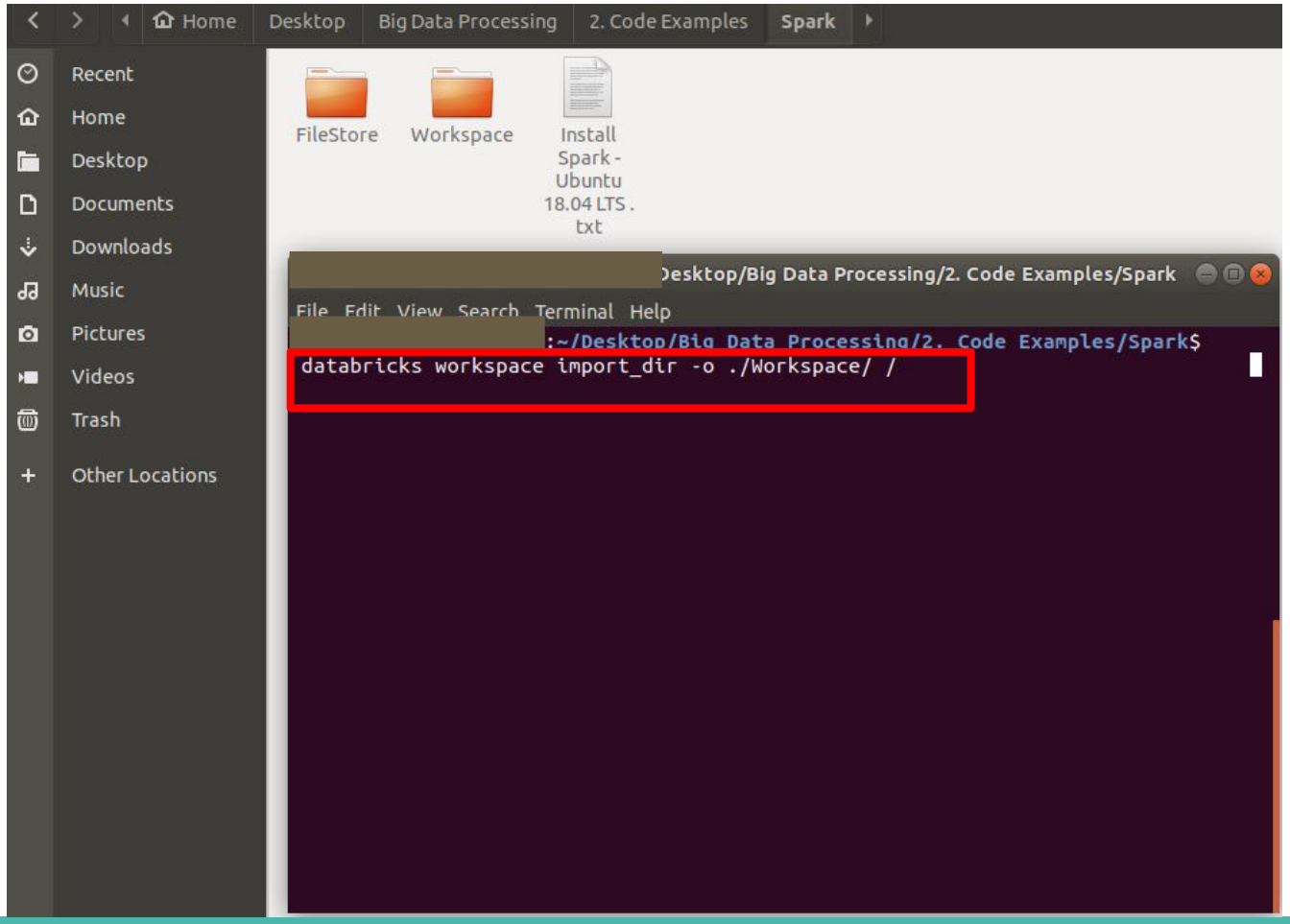
Databricks: An Online Platform for Data Engin

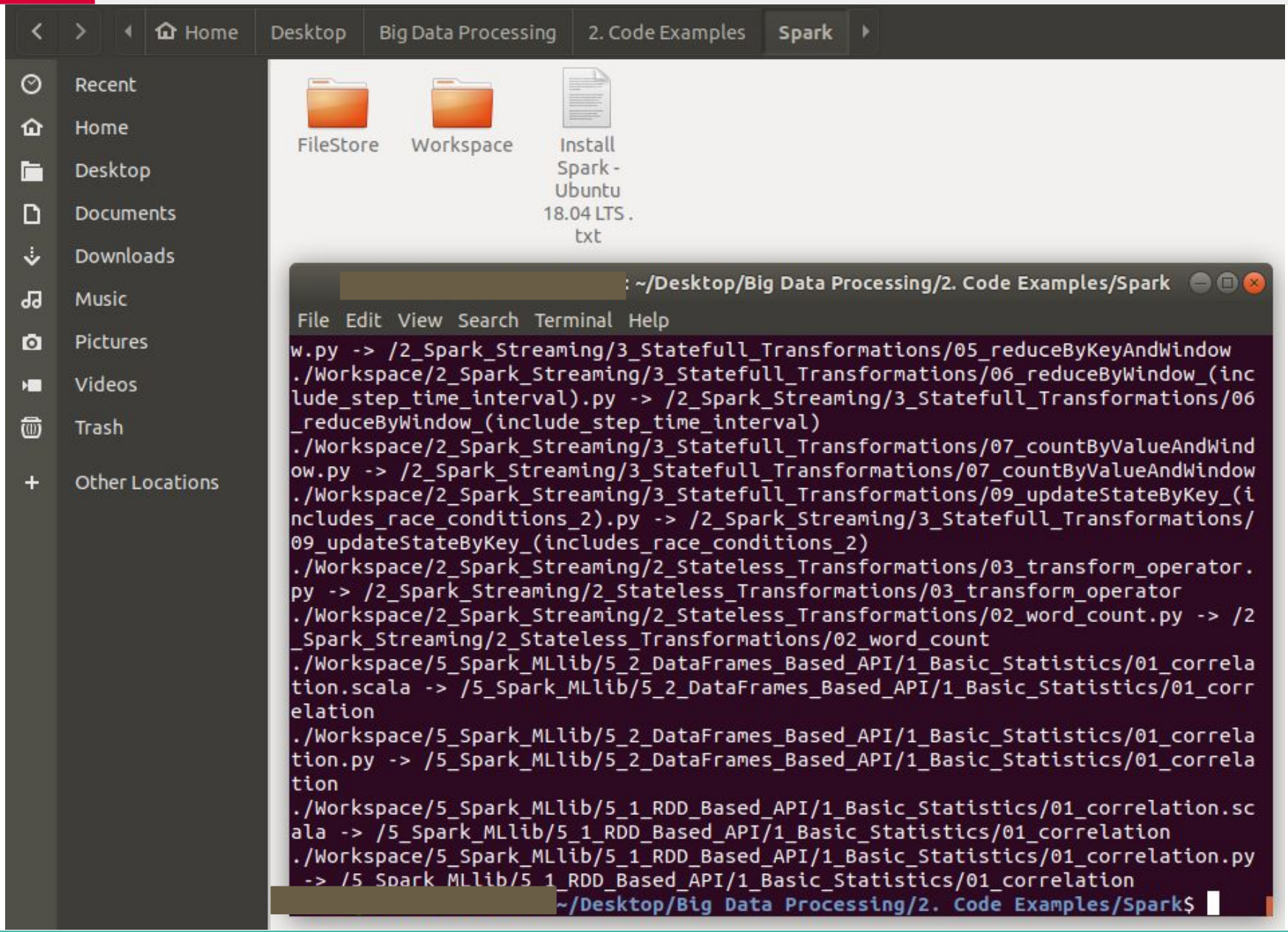


As we can see, before running the command, our **Databricks Workspace** only has the Python program we created before.

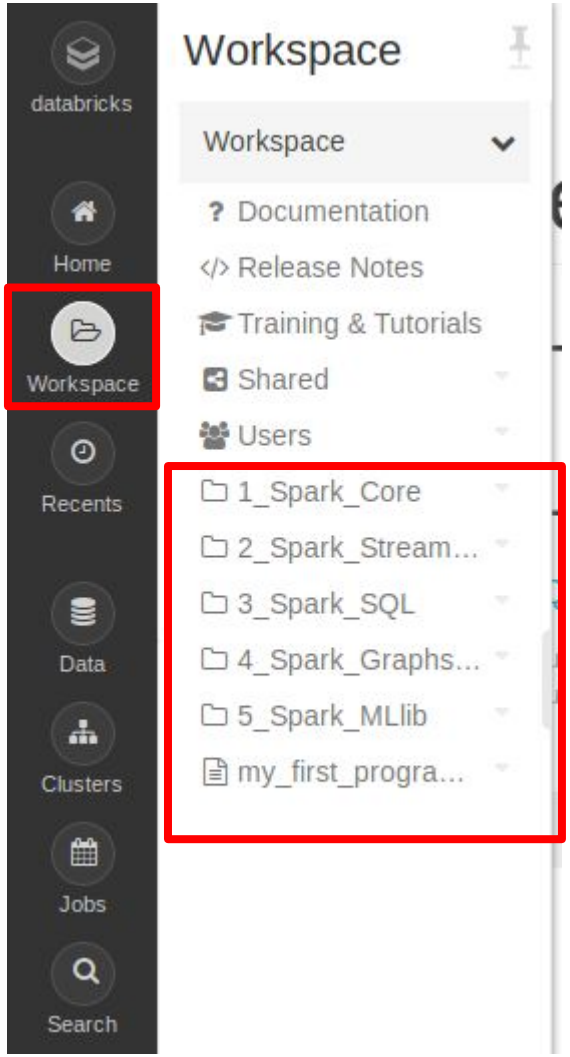
Databricks: An Online Platform for Data Engin

7. Running the command will take a couple of minutes.
   > databricks workspace import_dir -o ./Workspace/ /




Home
Desktop
Big Data Processing
2. Code Examples
Spark
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Trash
Other Locations
FileStore
Workspace
Install Spark - Ubuntu 18.04 LTS .txt
~/Desktop/Big Data Processing/2. Code Examples/Spark
File Edit View Search Terminal Help
w.py -> /2_Spark_Streaming/3_Statefull_Transformations/05_reduceByKeyAndWindow
./Workspace/2_Spark_Streaming/3_Statefull_Transformations/06_reduceByWindow_(include_step_time_interval).py -> /2_Spark_Streaming/3_Statefull_Transformations/06_reduceByWindow_(include_step_time_interval)
./Workspace/2_Spark_Streaming/3_Statefull_Transformations/07_countByValueAndWindow.py -> /2_Spark_Streaming/3_Statefull_Transformations/07_countByValueAndWindow
./Workspace/2_Spark_Streaming/3_Statefull_Transformations/09_updateStateByKey_(includes_race_conditions_2).py -> /2_Spark_Streaming/3_Statefull_Transformations/09_updateStateByKey_(includes_race_conditions_2)
./Workspace/2_Spark_Streaming/2_Stateless_Transformations/03_transform_operator.py -> /2_Spark_Streaming/2_Stateless_Transformations/03_transform_operator
./Workspace/2_Spark_Streaming/2_Stateless_Transformations/02_word_count.py -> /2_Spark_Streaming/2_Stateless_Transformations/02_word_count
./Workspace/5_Spark_MLlib/5_2_DataFrames_Based_API/1_Basic_Statistics/01_correlation.scala -> /5_Spark_MLlib/5_2_DataFrames_Based_API/1_Basic_Statistics/01_correlation
./Workspace/5_Spark_MLlib/5_2_DataFrames_Based_API/1_Basic_Statistics/01_correlation.py -> /5_Spark_MLlib/5_2_DataFrames_Based_API/1_Basic_Statistics/01_correlation
./Workspace/5_Spark_MLlib/5_1_RDD_Based_API/1_Basic_Statistics/01_correlation.scala -> /5_Spark_MLlib/5_1_RDD_Based_API/1_Basic_Statistics/01_correlation
./Workspace/5_Spark_MLlib/5_1_RDD_Based_API/1_Basic_Statistics/01_correlation.py -> /5_Spark_MLlib/5_1_RDD_Based_API/1_Basic_Statistics/01_correlation
~/Desktop/Big Data Processing/2. Code Examples/Spark$

Databricks: An Online Platform for Data Engin



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.

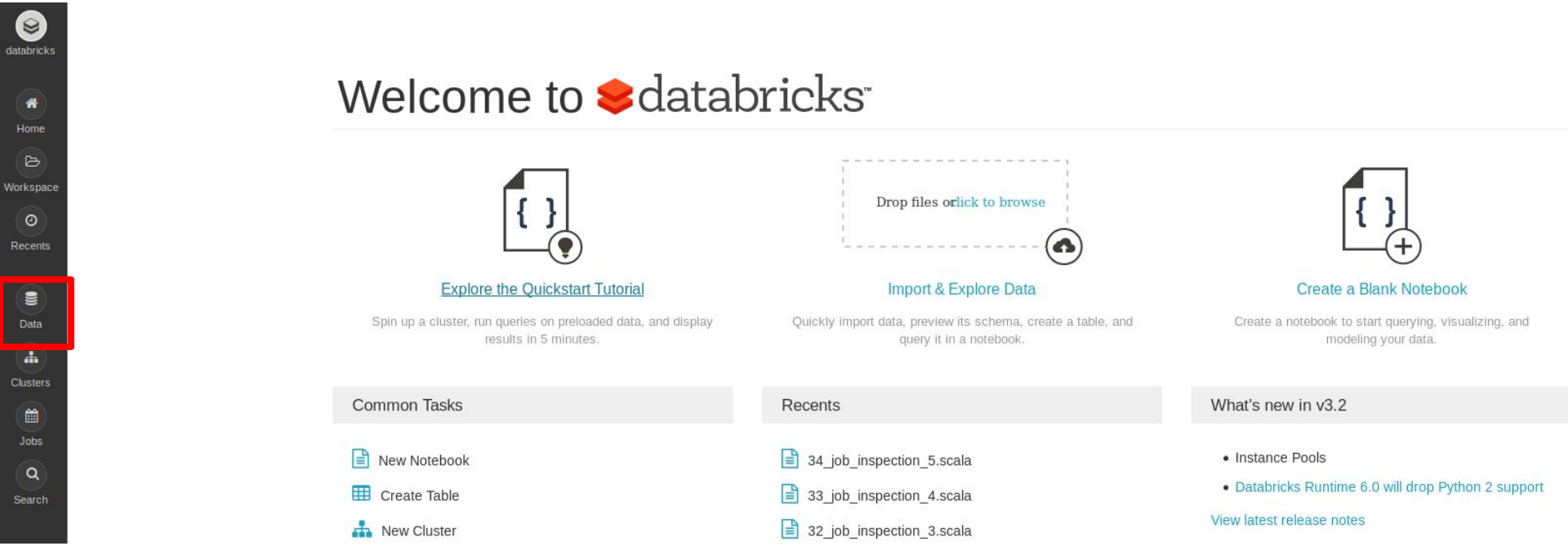
Databricks: An Online Platform for Data Engineers

How to...

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

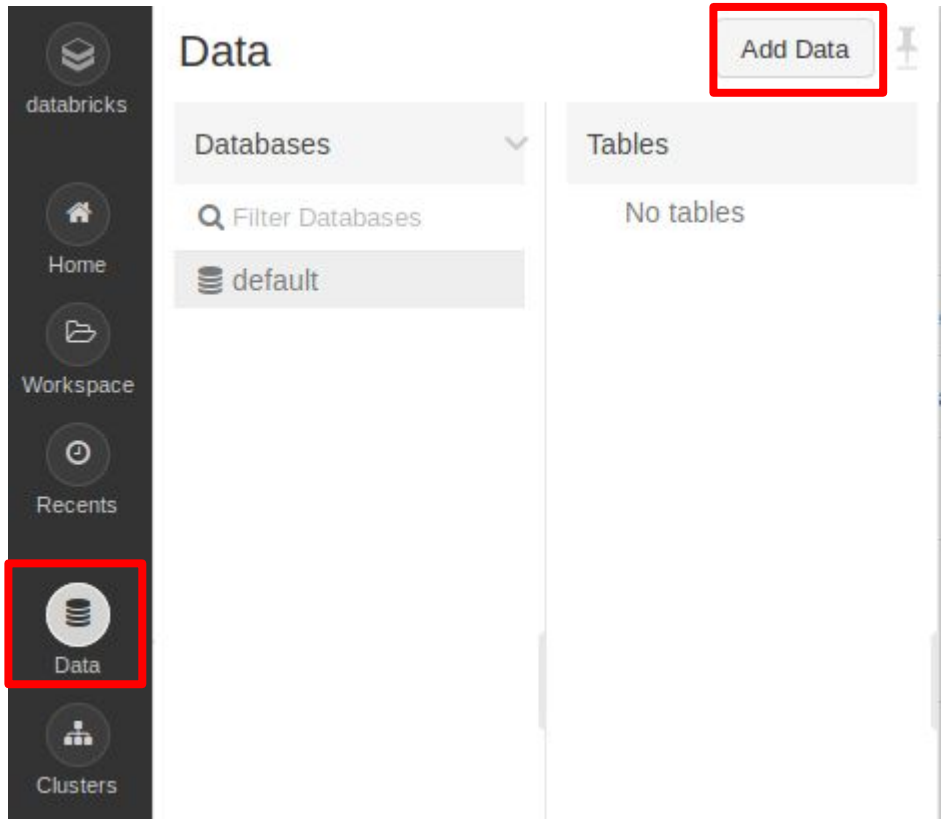
Databricks: An Online Platform for Data Engineers

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



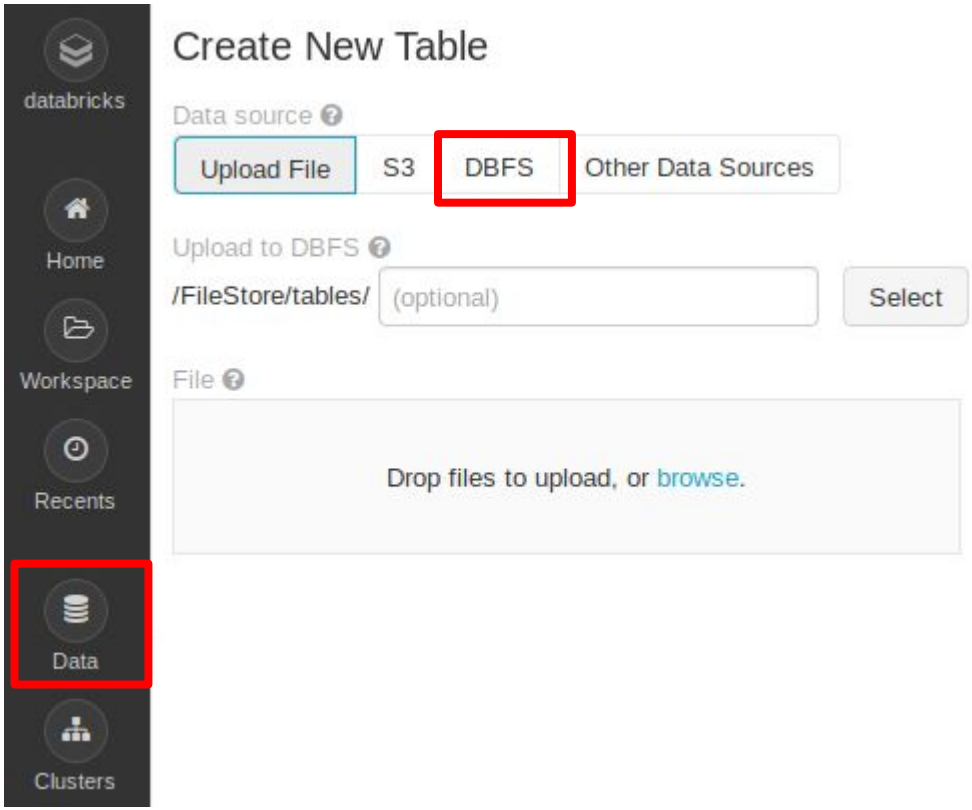
Databricks: An Online Platform for Data Engineers

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



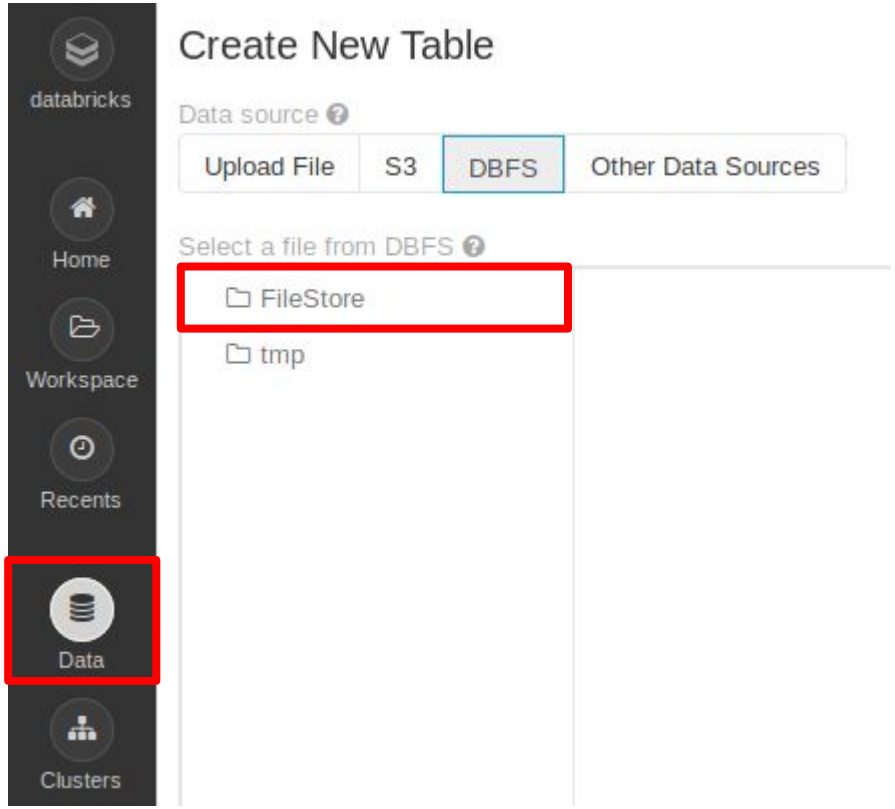
Databricks: An Online Platform for Data Engineers

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

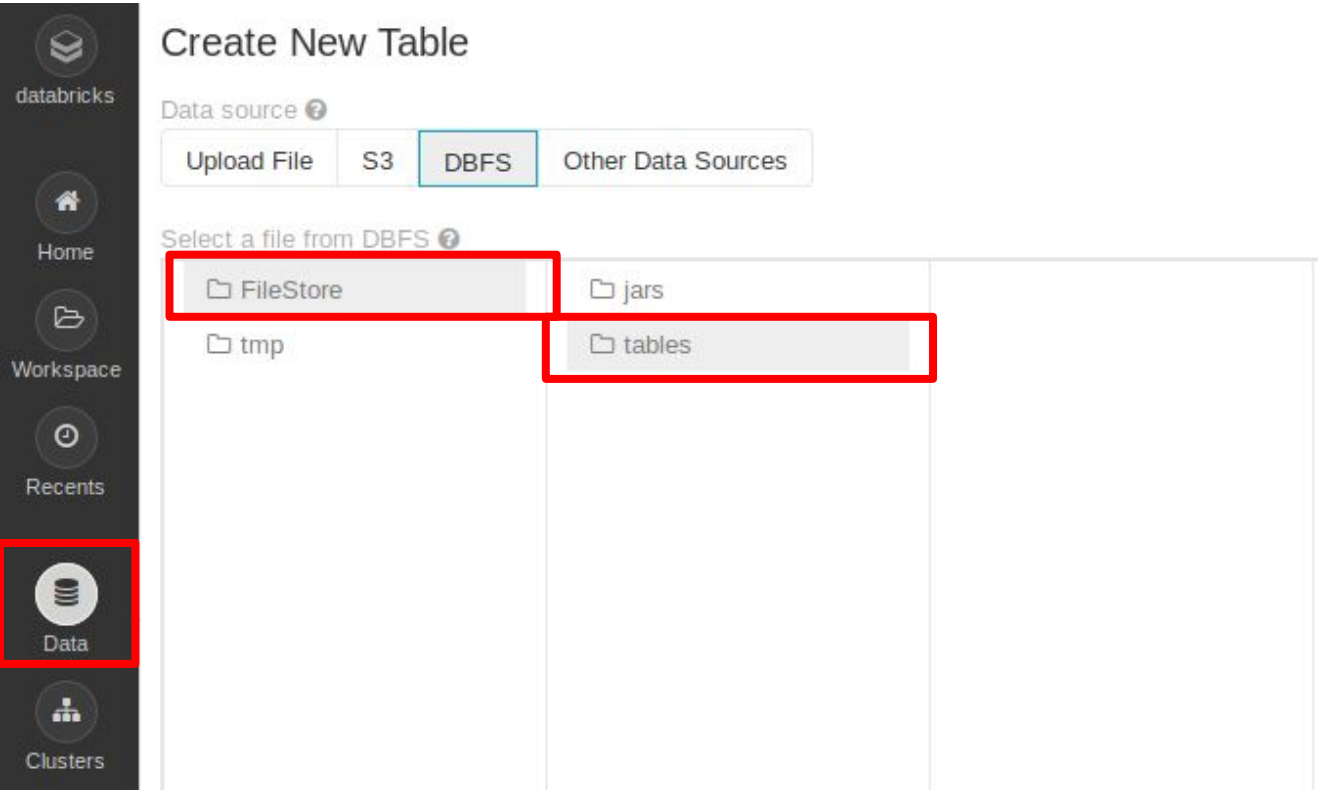
8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



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

Databricks: An Online Platform for Data Engineers

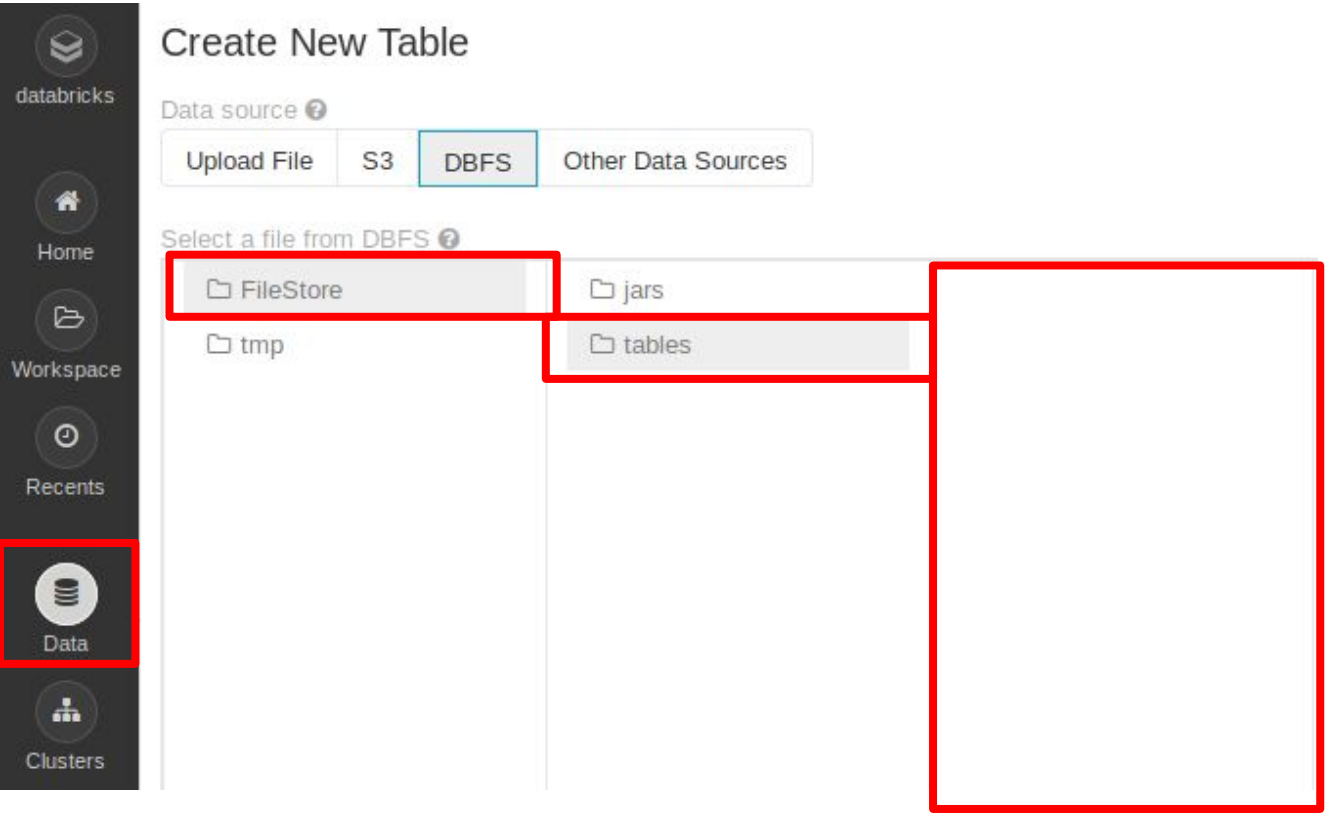
8. We can see the files stored in the online Databricks File System (DBFS) by clicking in **Data**.



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**.



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

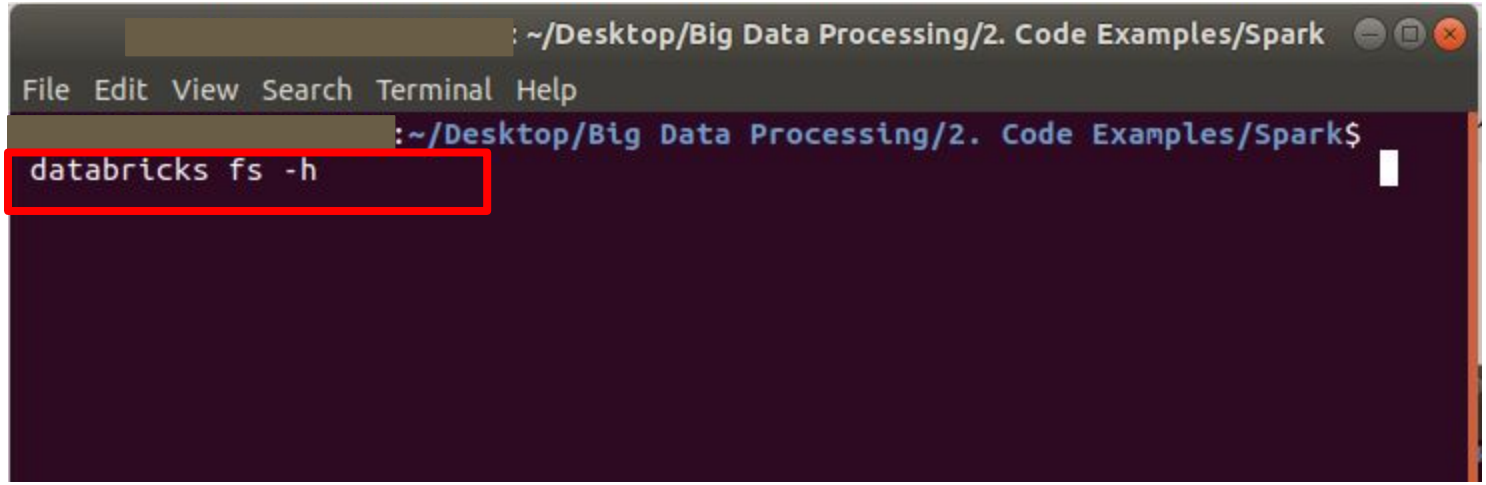
As we can see by the moment there is no Dataset store at DBFS.

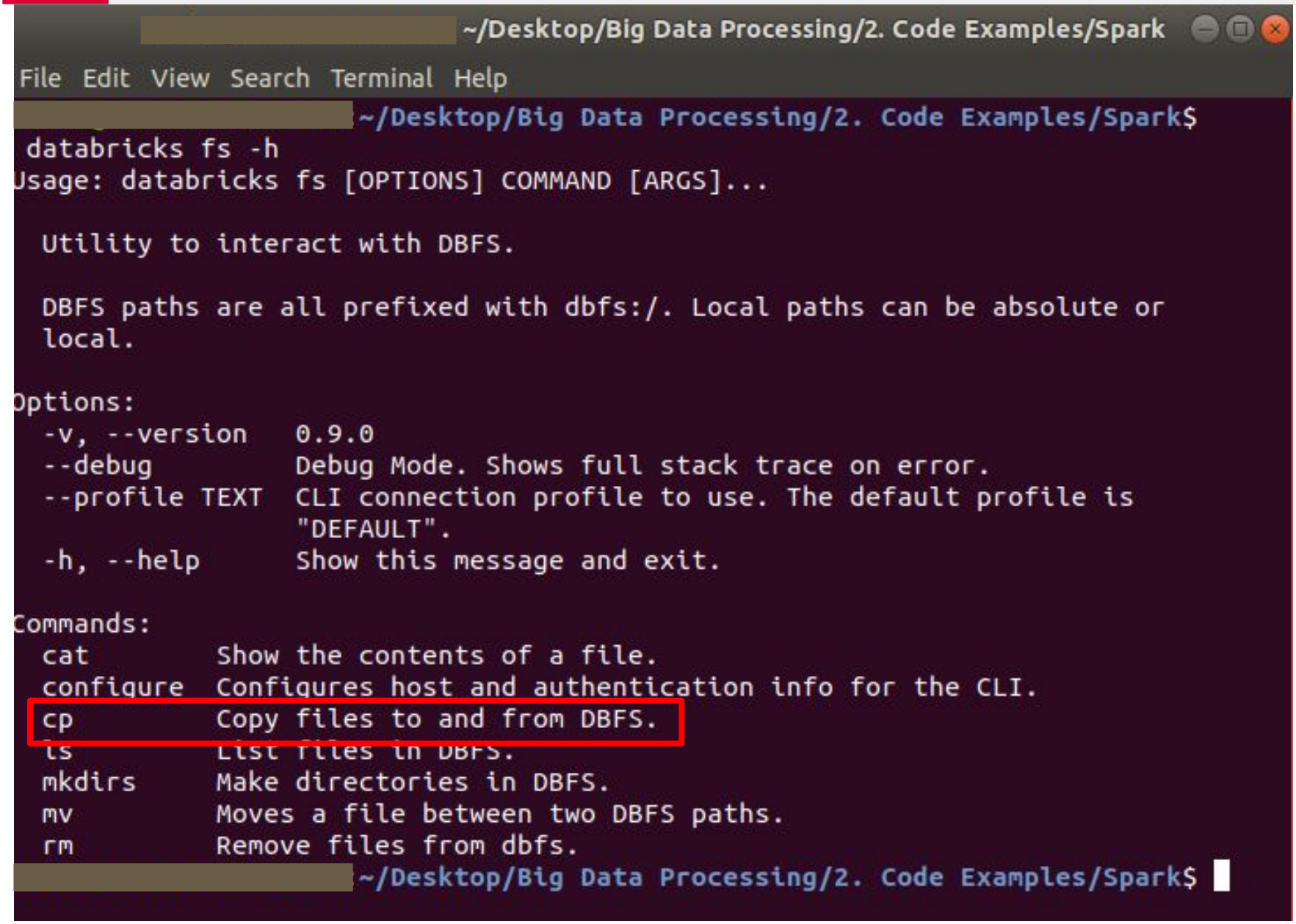
Databricks: An Online Platform for Data Engineers

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

Databricks: An Online Platform for Data Engineers

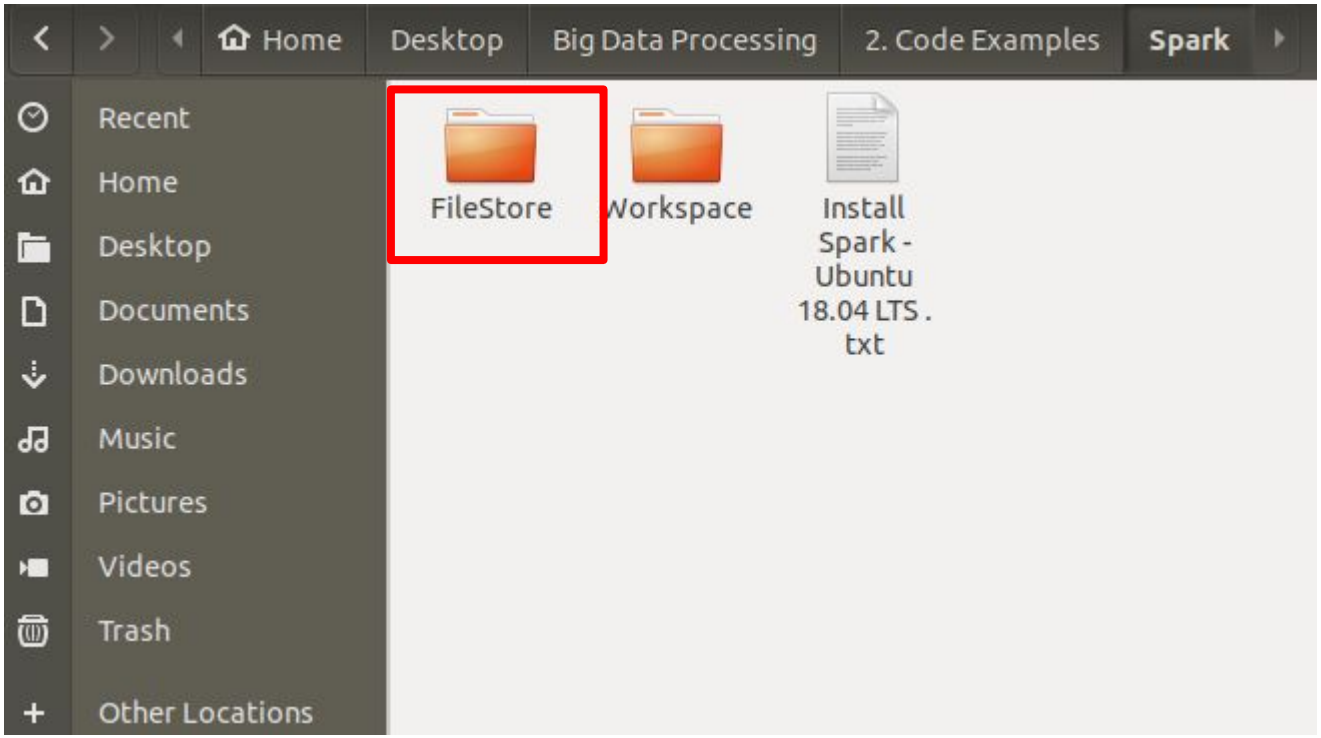
8. The CLI has some **Databricks File System (DBFS)** specific commands:




~/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$

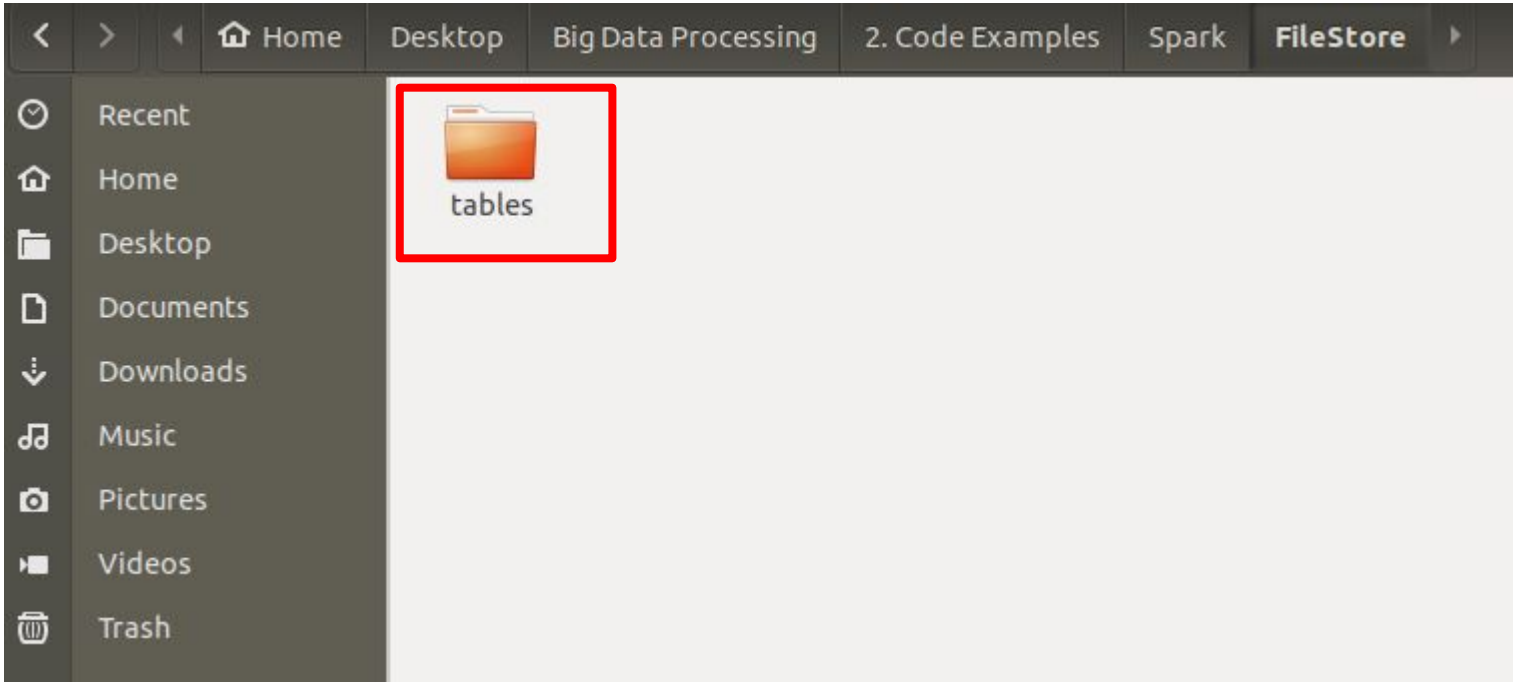
Databricks: An Online Platform for Data Engineers

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



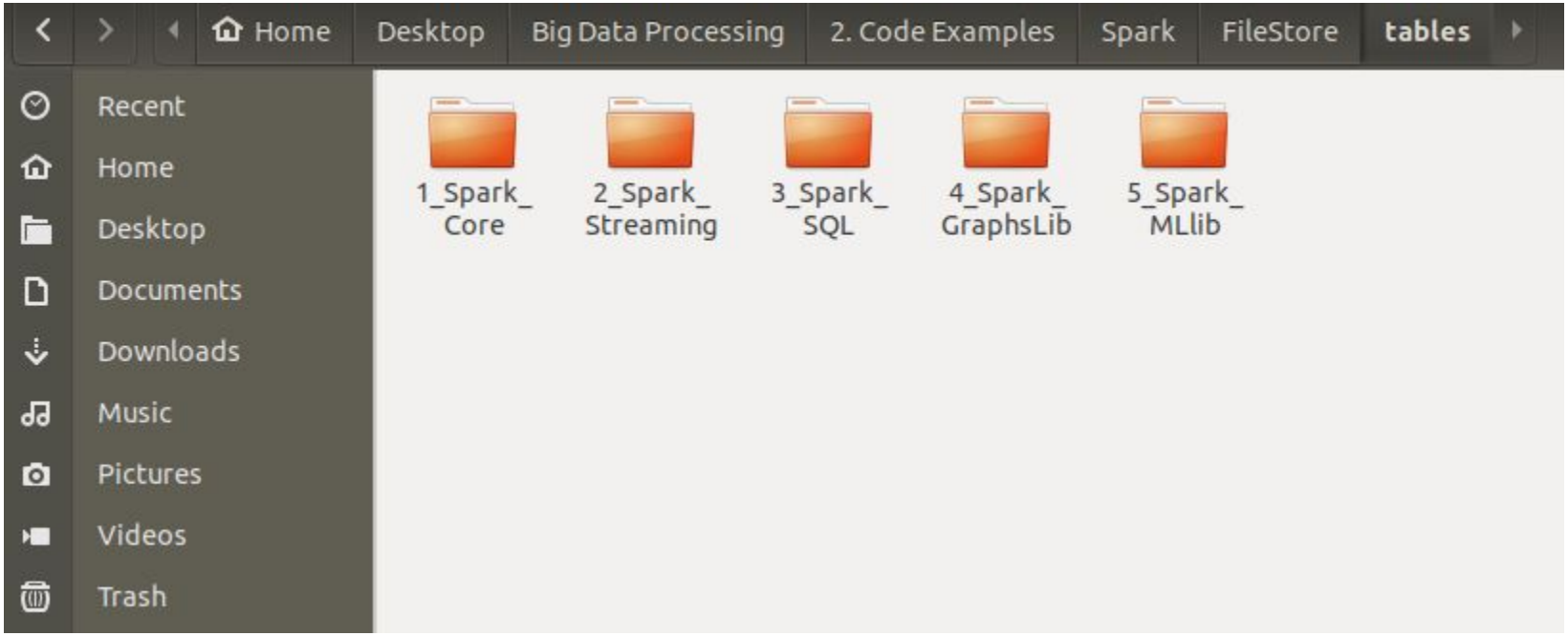
Databricks: An Online Platform for Data Engineers

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



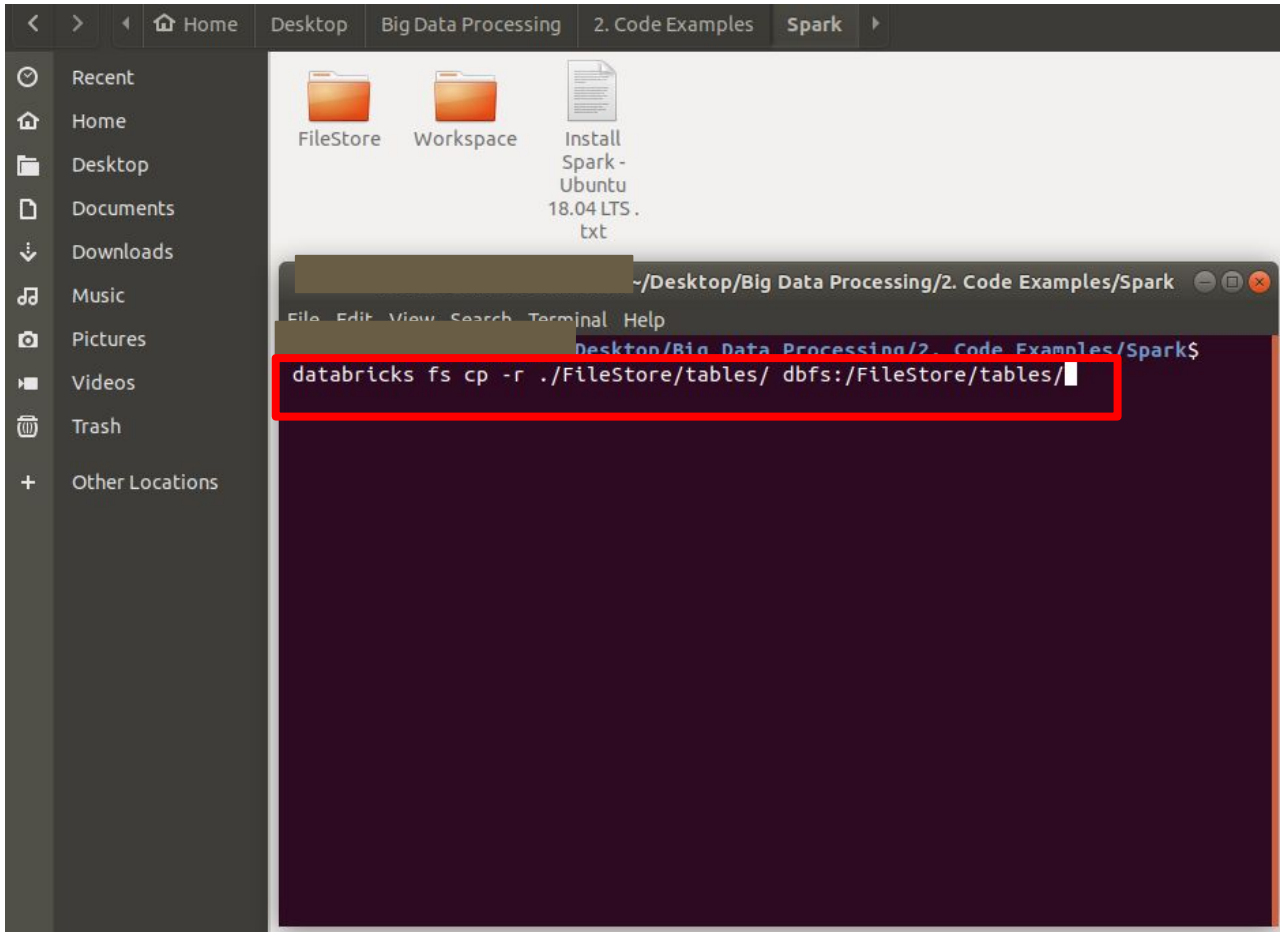
Databricks: An Online Platform for Data Engineers

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



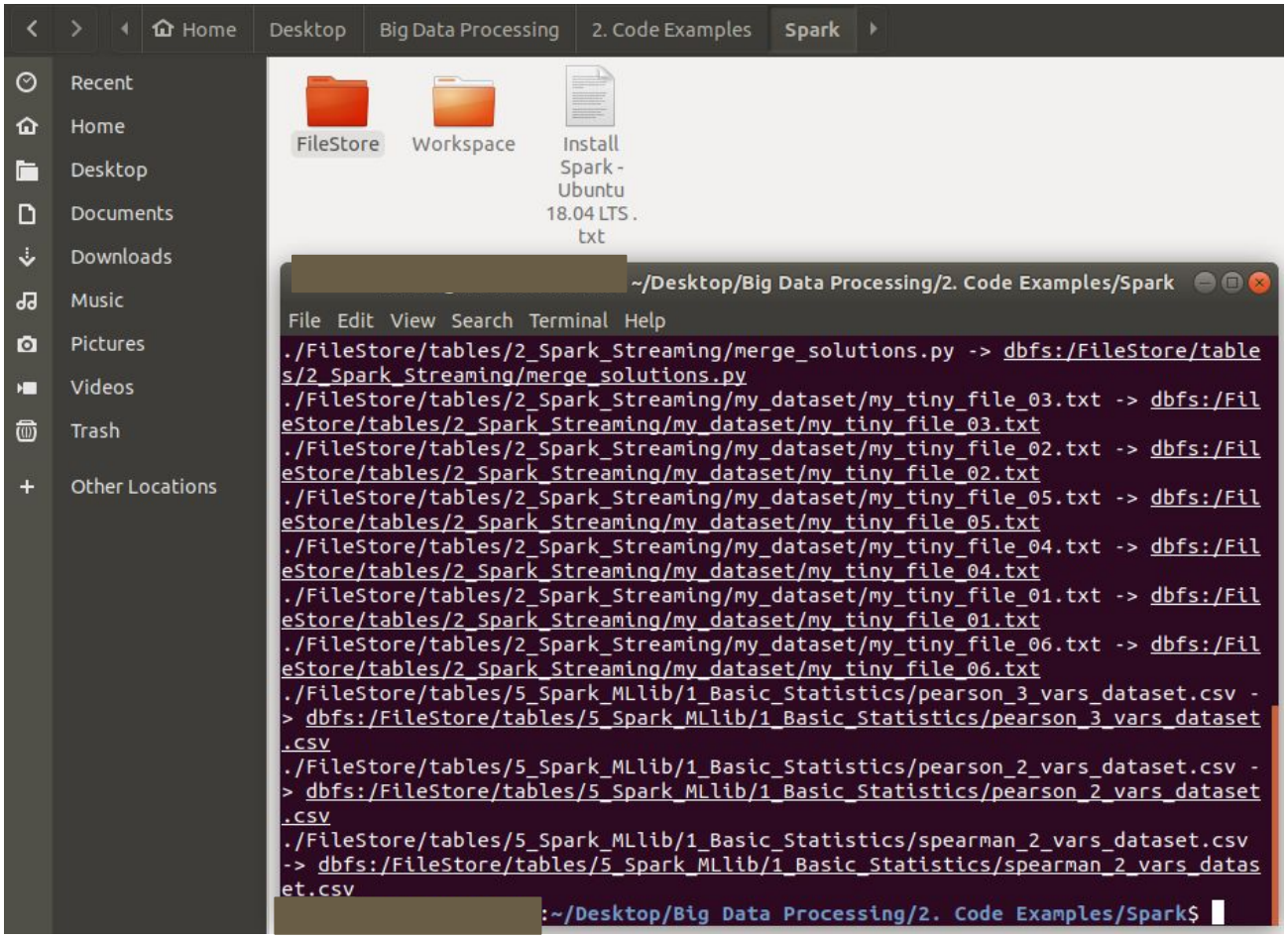
Databricks: An Online Platform for Data Engin

8. Running the command will take a couple of minutes.
   > databricks fs cp -r ./FileStore/tables dbfs:/FileStore/tables/

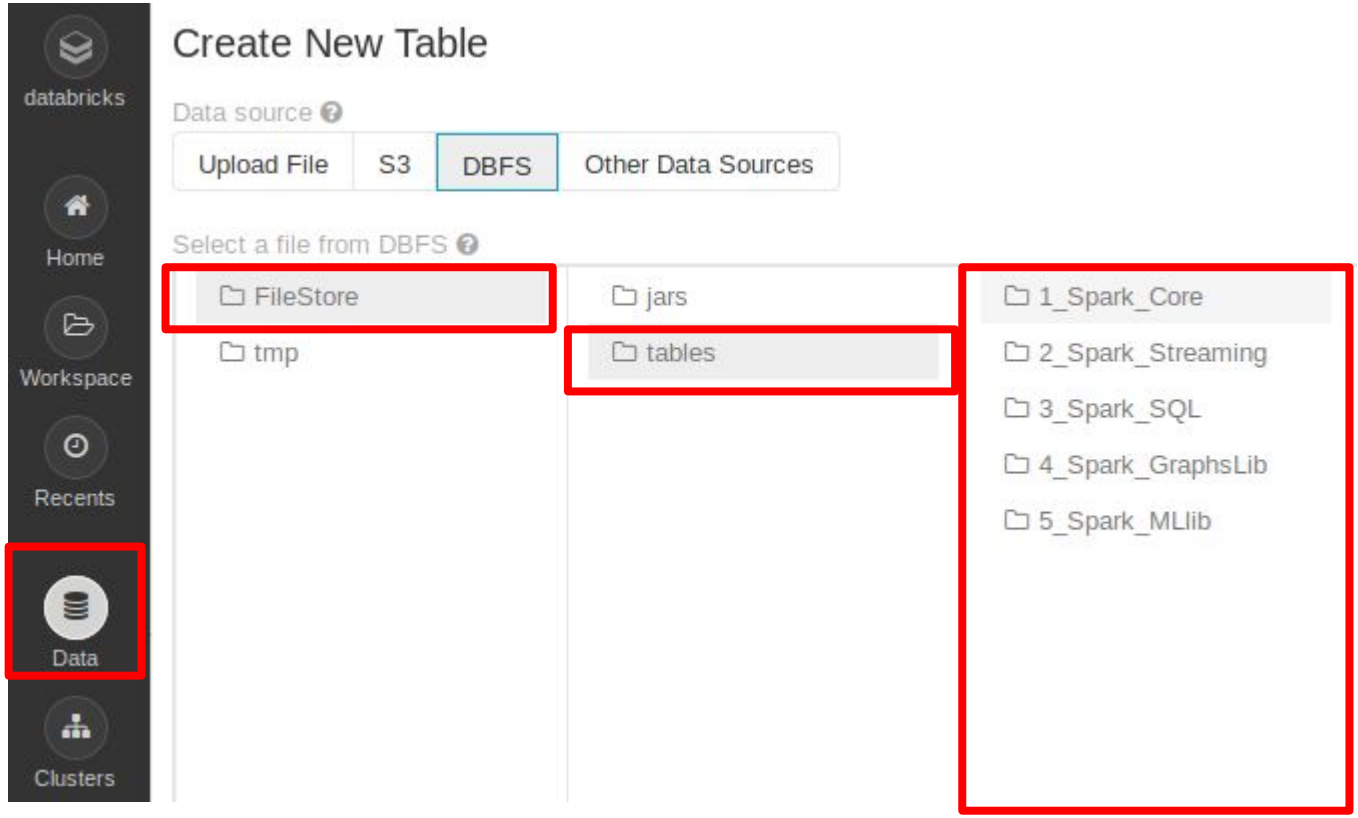


Databricks: An Online Platform for Data Engin

8. Running the command will take a couple of minutes.
   > databricks fs cp -r ./FileStore/tables dbfs:/FileStore/tables/



Databricks: An Online Platform for Data Engin



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

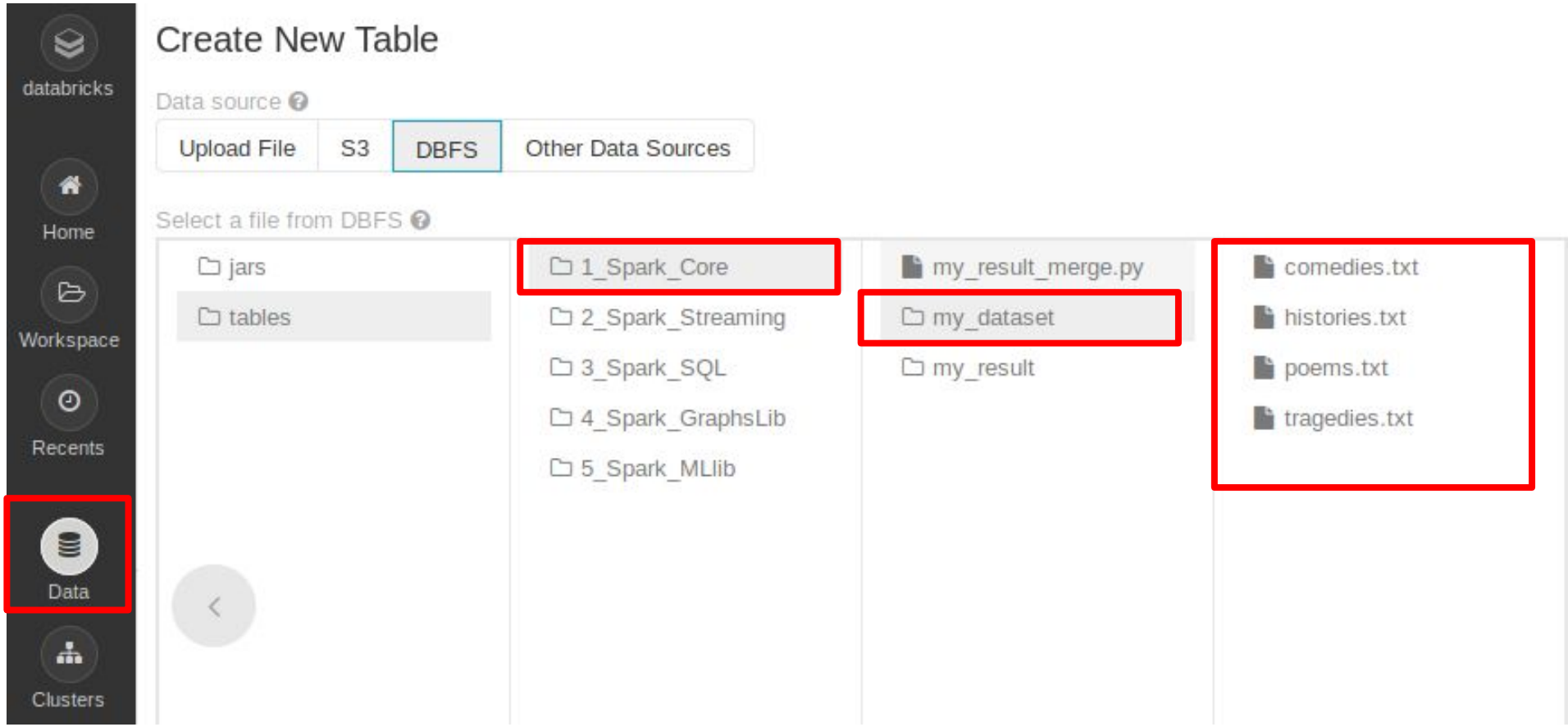
Databricks: An Online Platform for Data Engineers

How to...

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

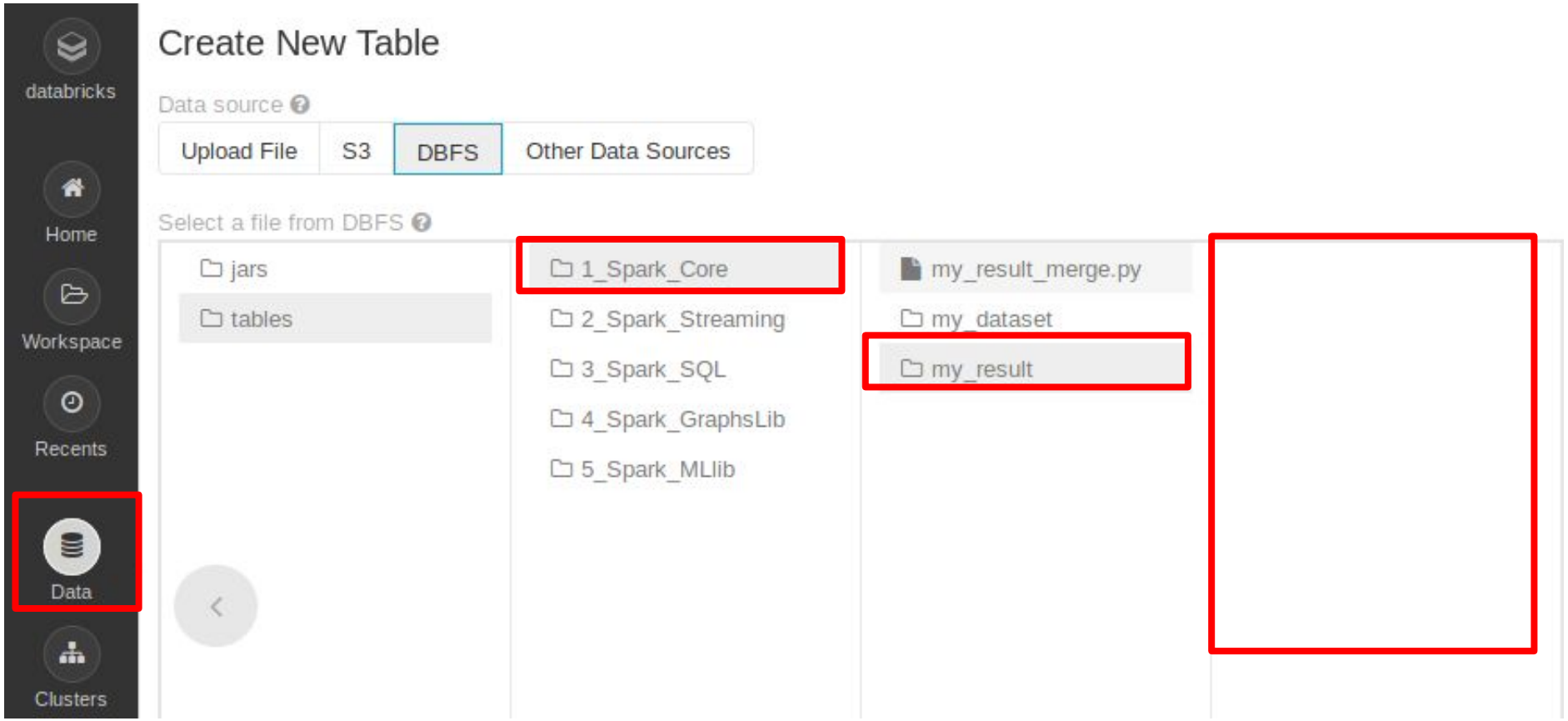
Databricks: An Online Platform for Data Engin

9. As we can see, Spark Core contains **my_dataset** of 4 text files:
   comedies.txt histories.txt poems.txt tragedies.txt



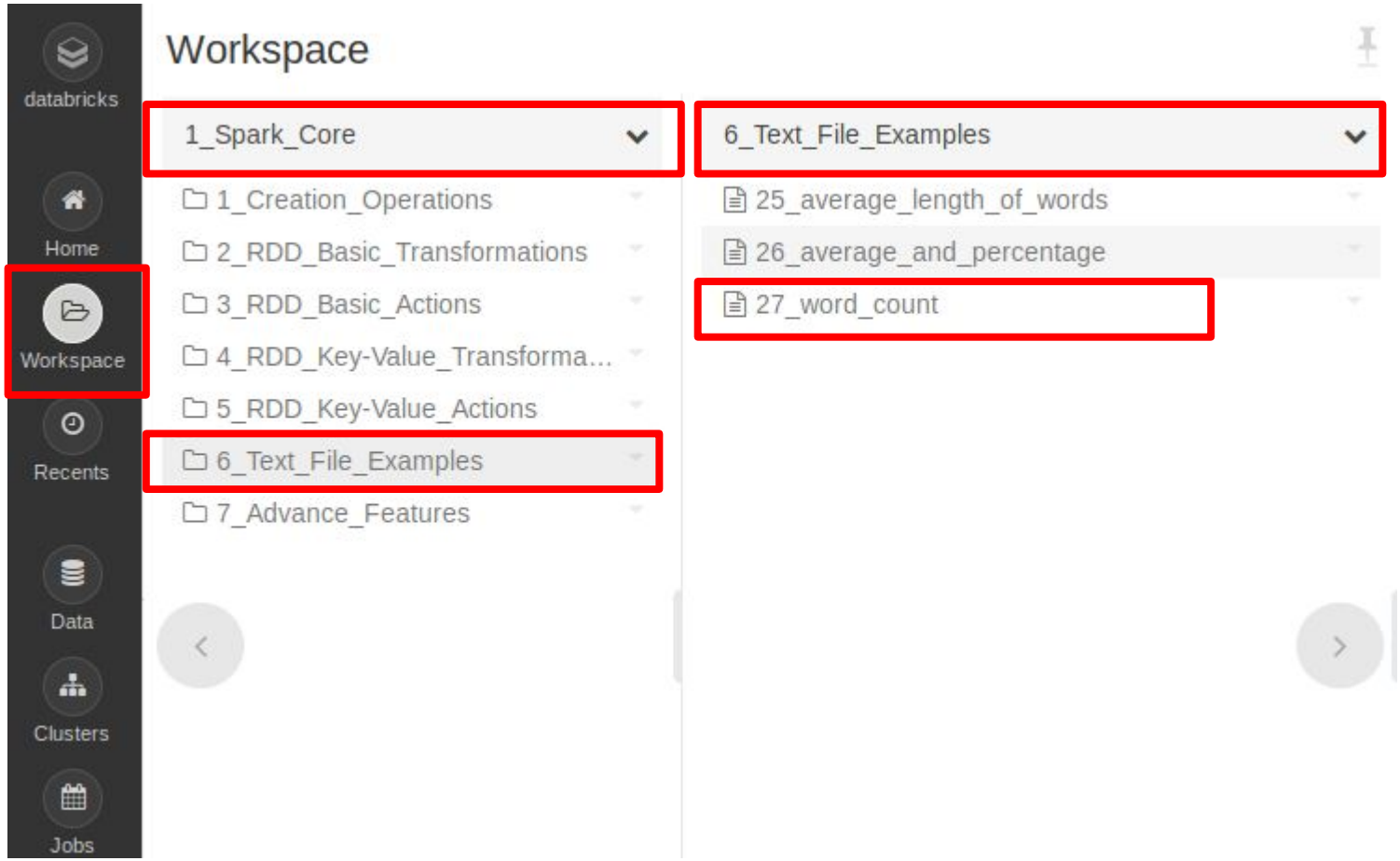
Databricks: An Online Platform for Data Engin

9. As we can see, Spark Core contains the folder **my_result**, being empty.



Databricks: An Online Platform for Data Engin

9. 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

9. 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

9. 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

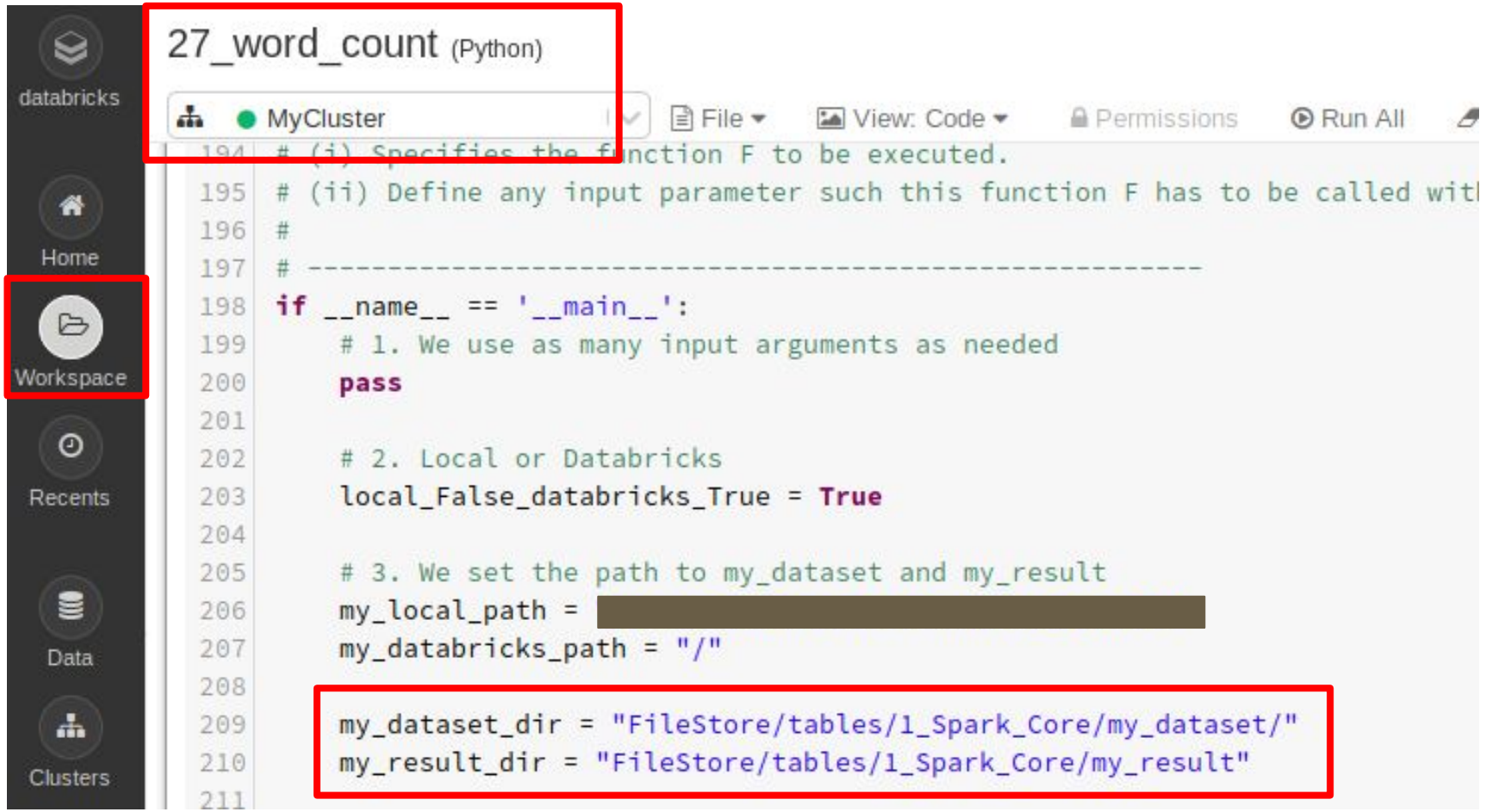
Databricks: An Online Platform for Data Engin

9. 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**

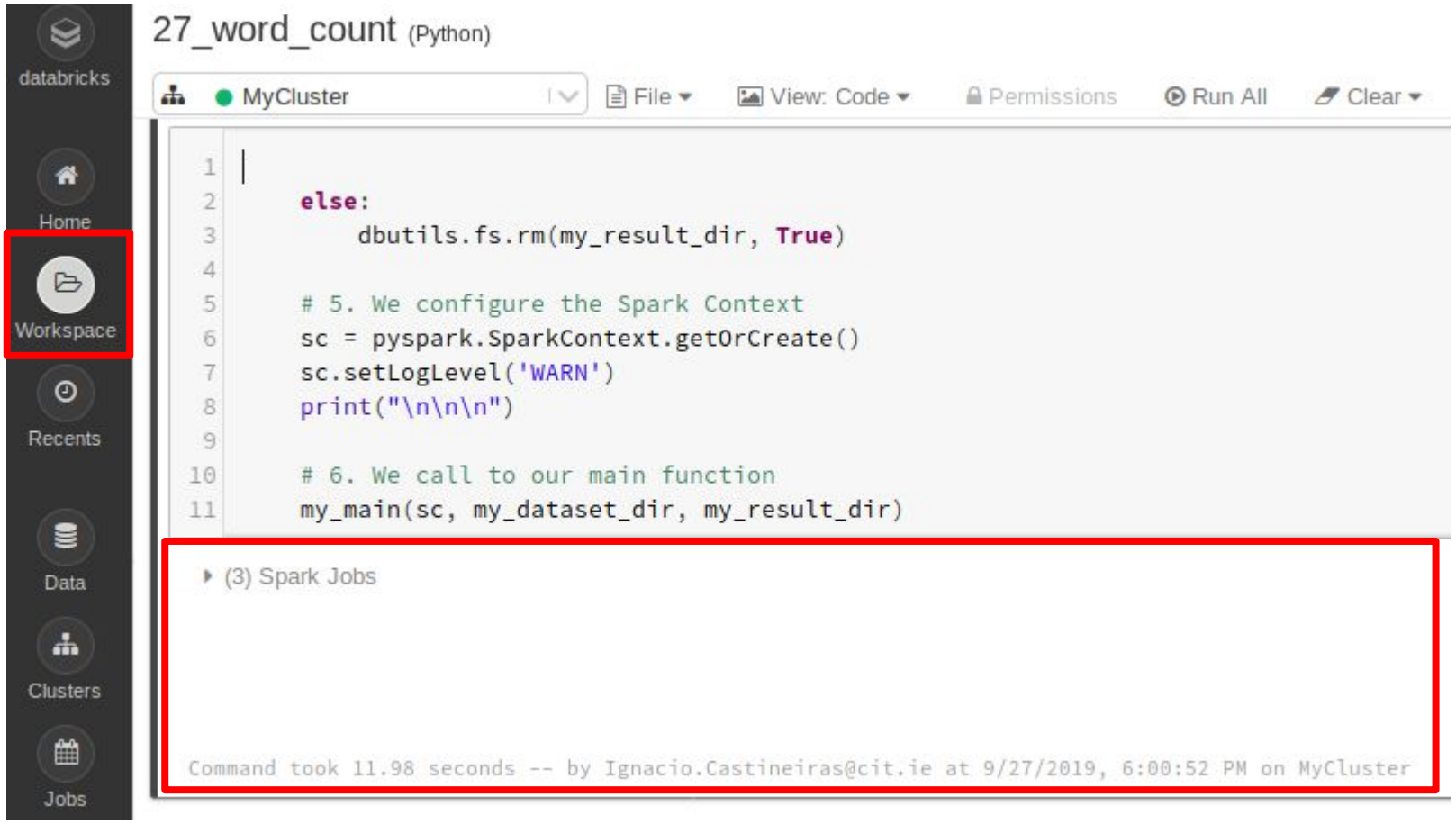
Databricks: An Online Platform for Data Engin

9. Without explaining the program itself, our code example Workspace/1_Spark_Core/6_Text_File_Examples/27_word_count.py



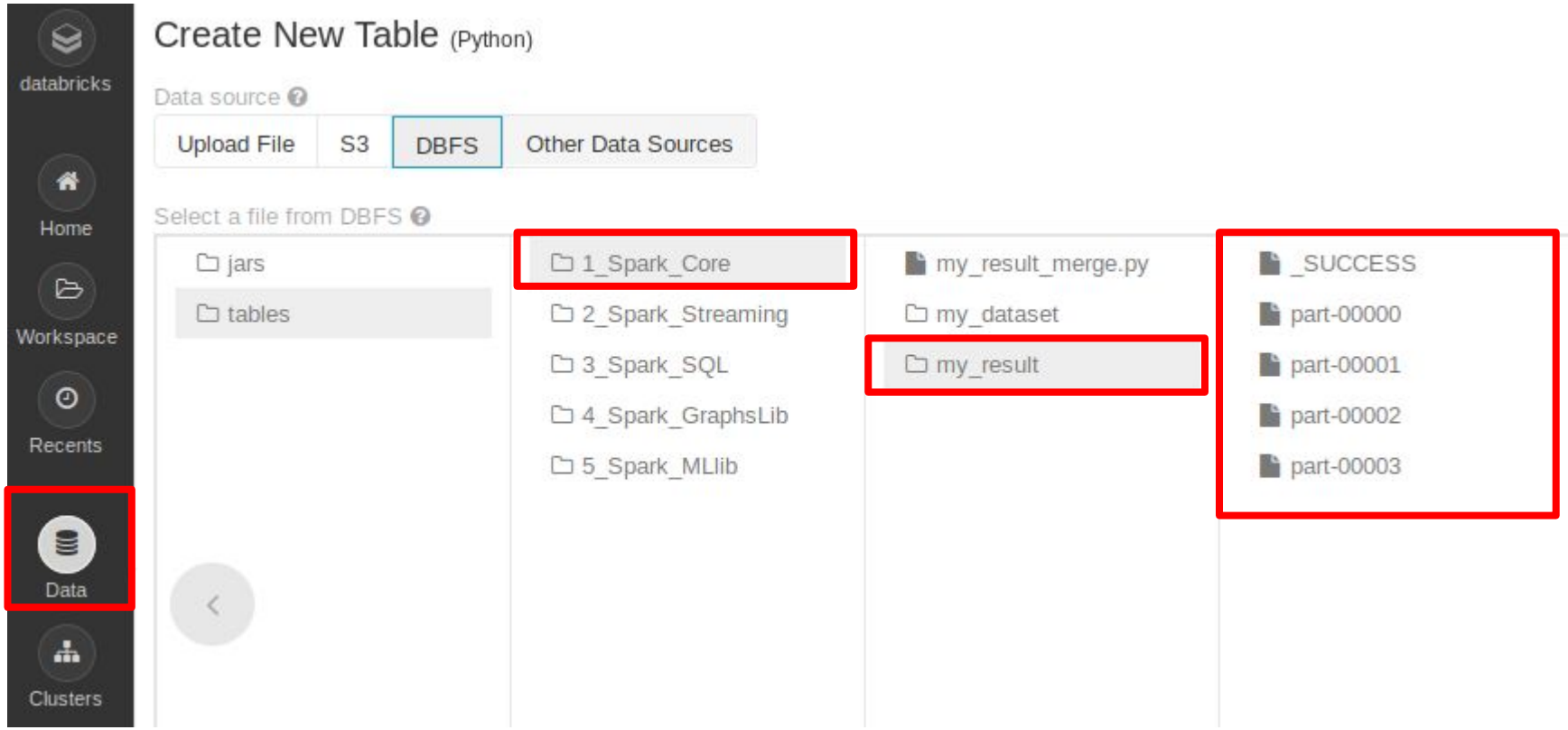
Databricks: An Online Platform for Data Engin

9. Running the program / Spark application leads to no visual results:
   Workspace/1_Spark_Core/6_Text_File_Examples/27_word_count.py



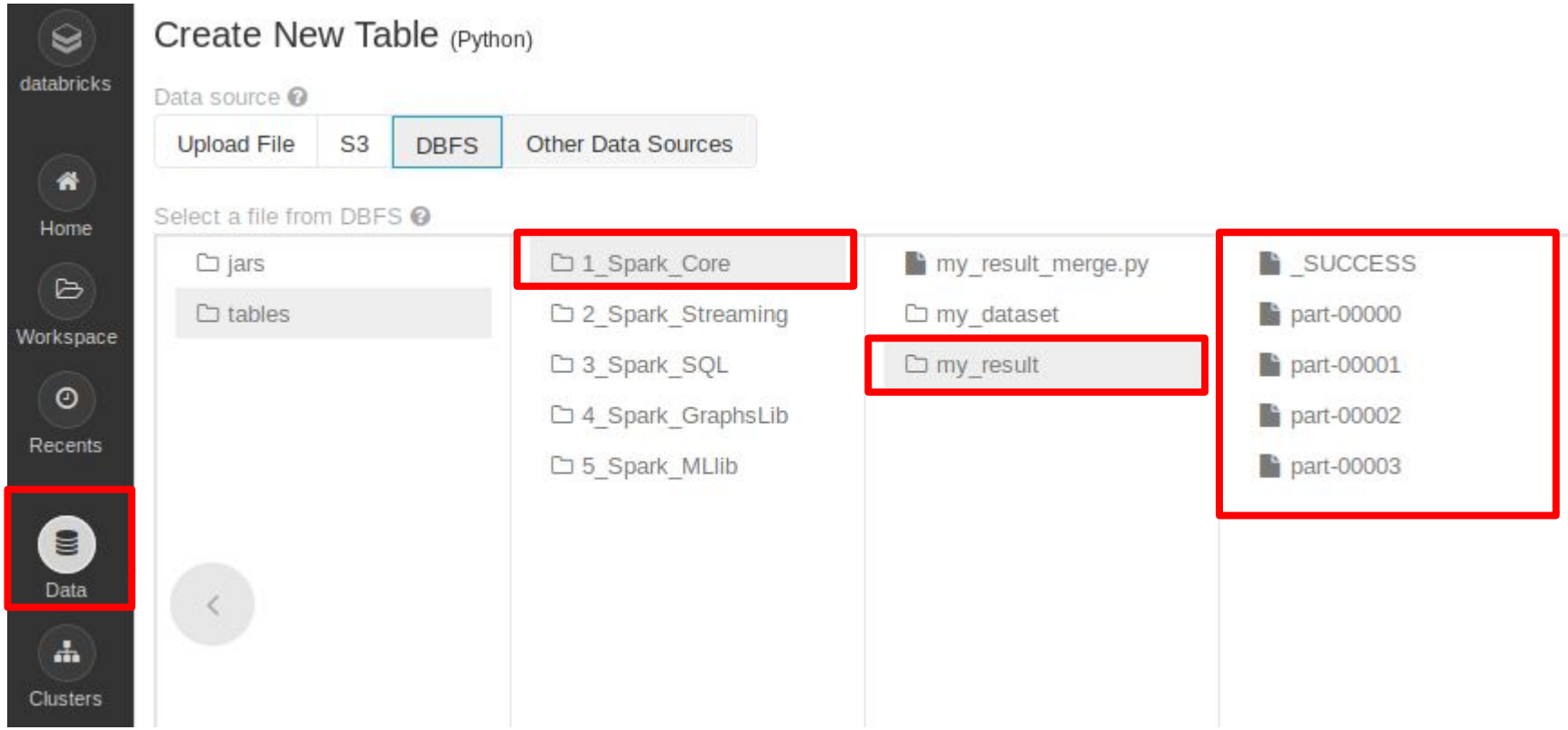
Databricks: An Online Platform for Data Engin

9. However, the folder **my_result** in the DBFS has been populated:



Databricks: An Online Platform for Data Engin

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

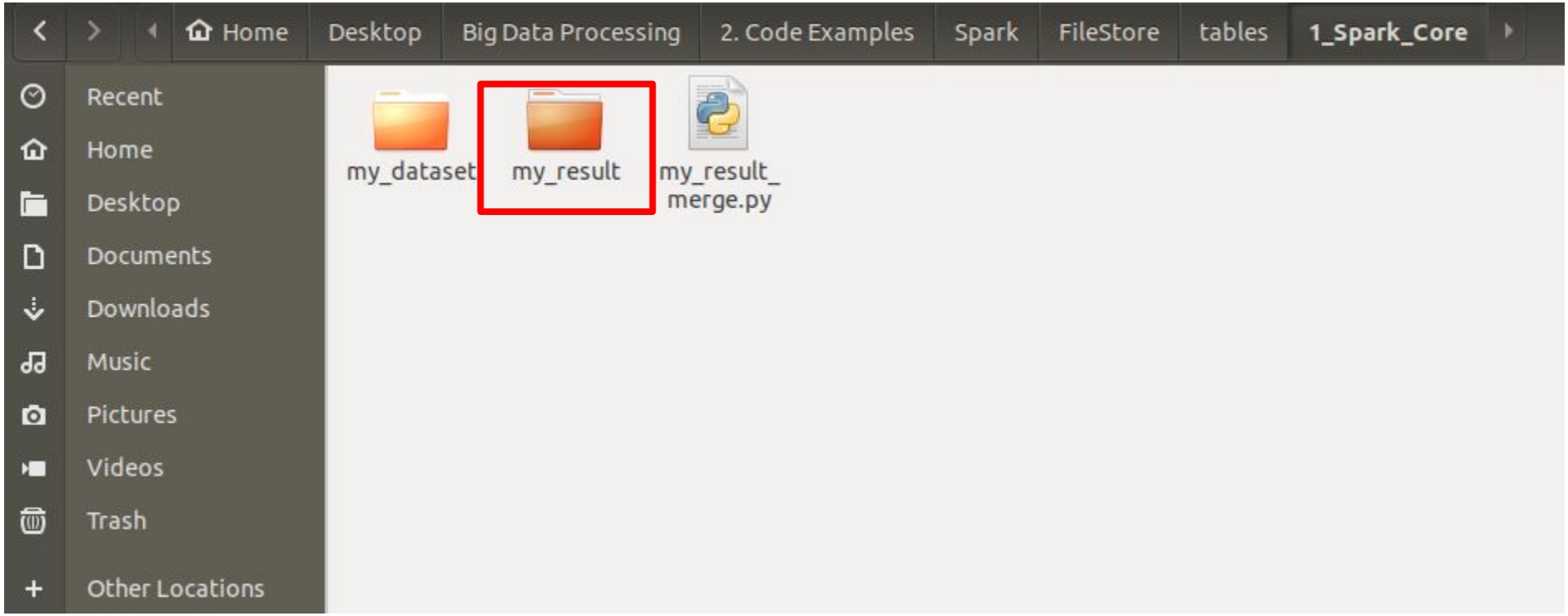


Databricks: An Online Platform for Data Engin

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

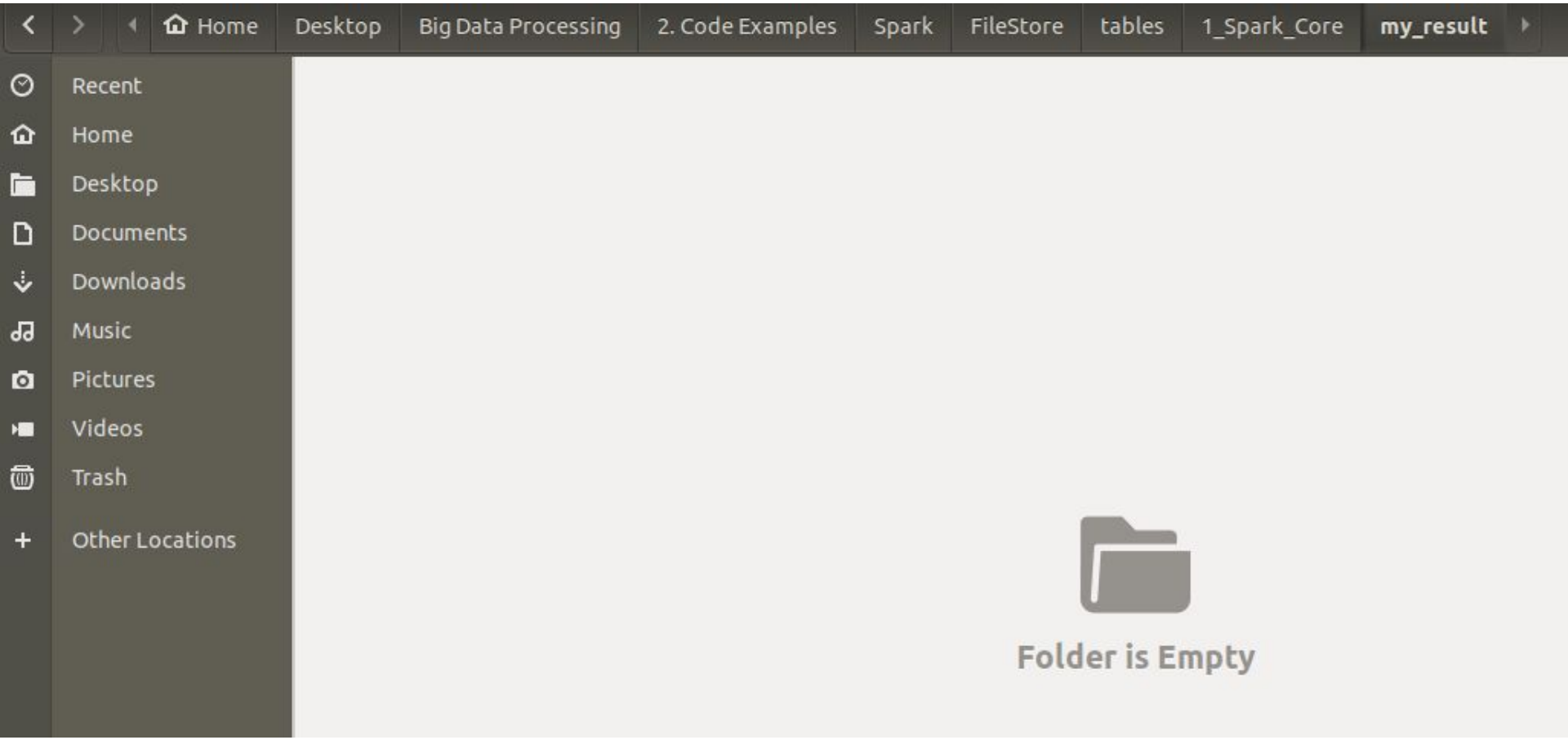
Databricks: An Online Platform for Data Engin

9. As we can see, before running the command **cp** the local folder **my_result** is empty.



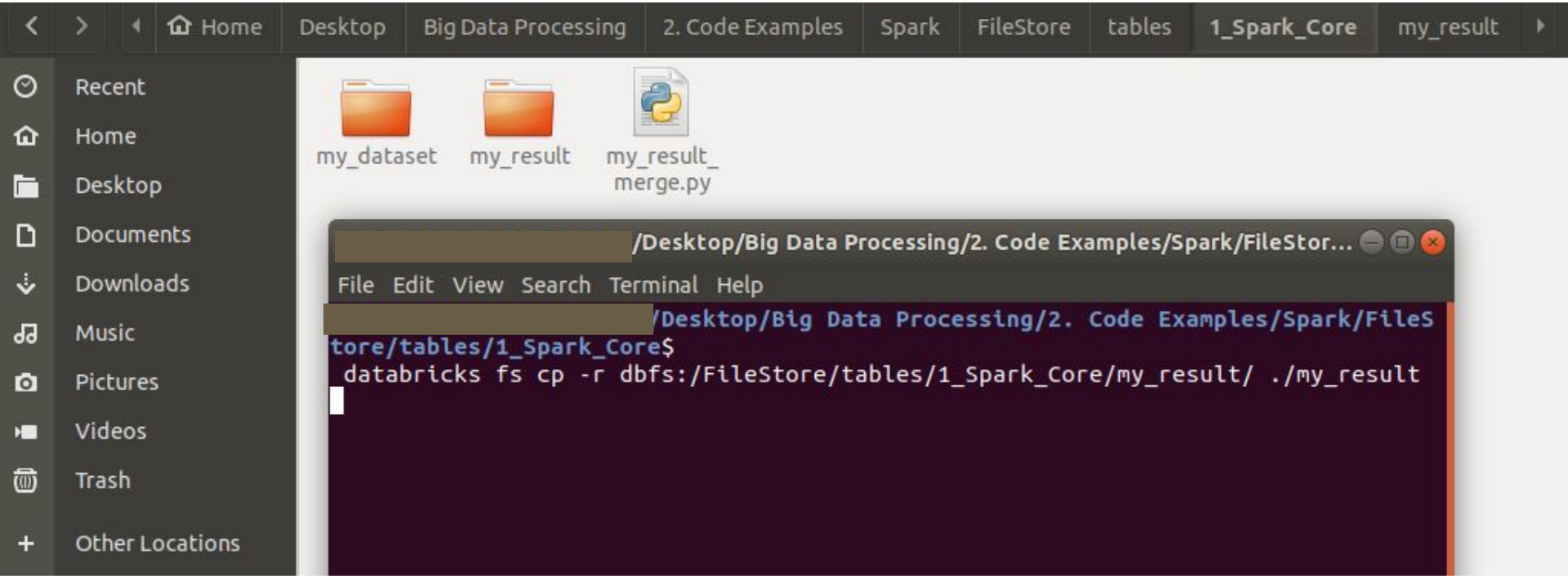
Databricks: An Online Platform for Data Engin

9. As we can see, before running the command **cp** the local folder **my_result** is empty.



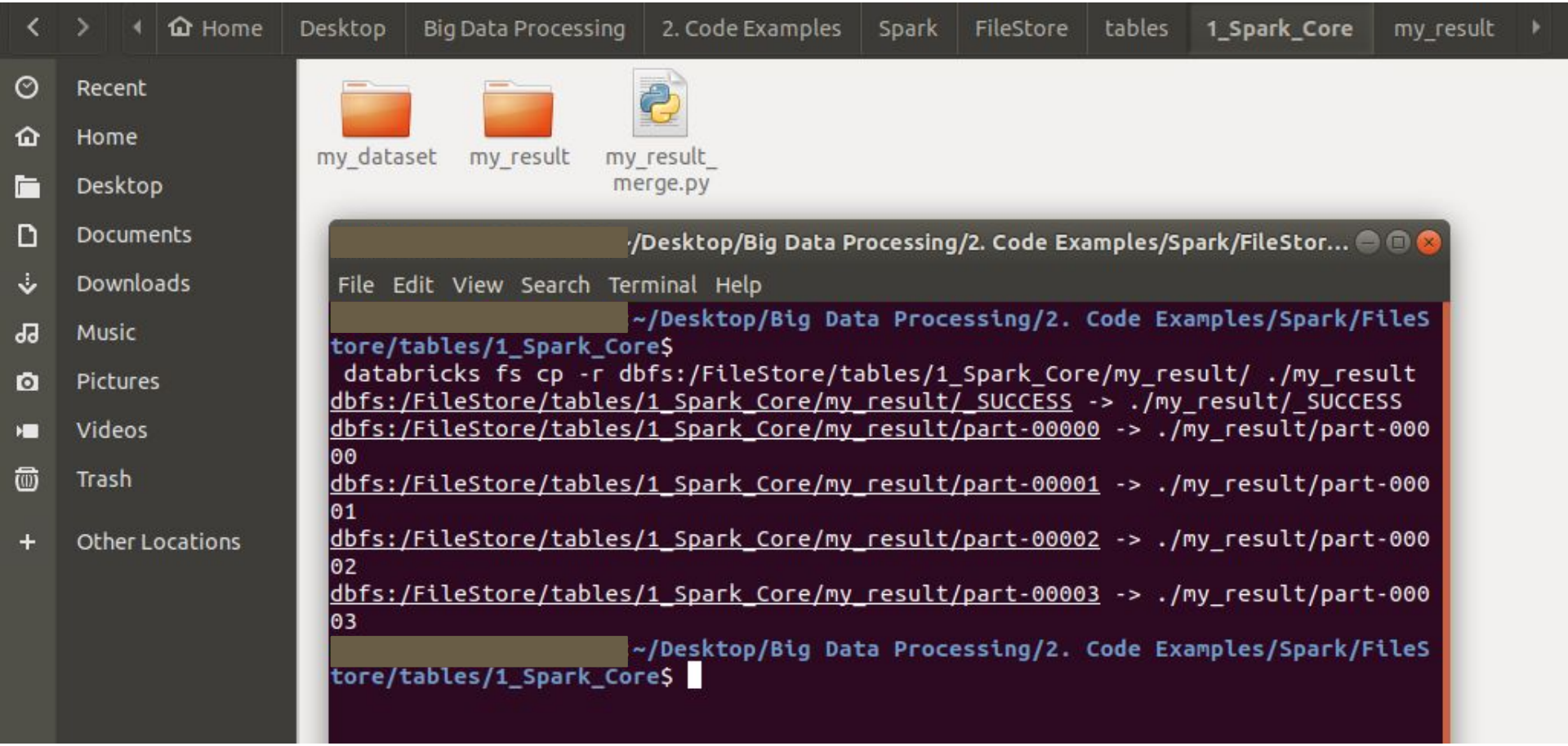
Databricks: An Online Platform for Data Engin

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



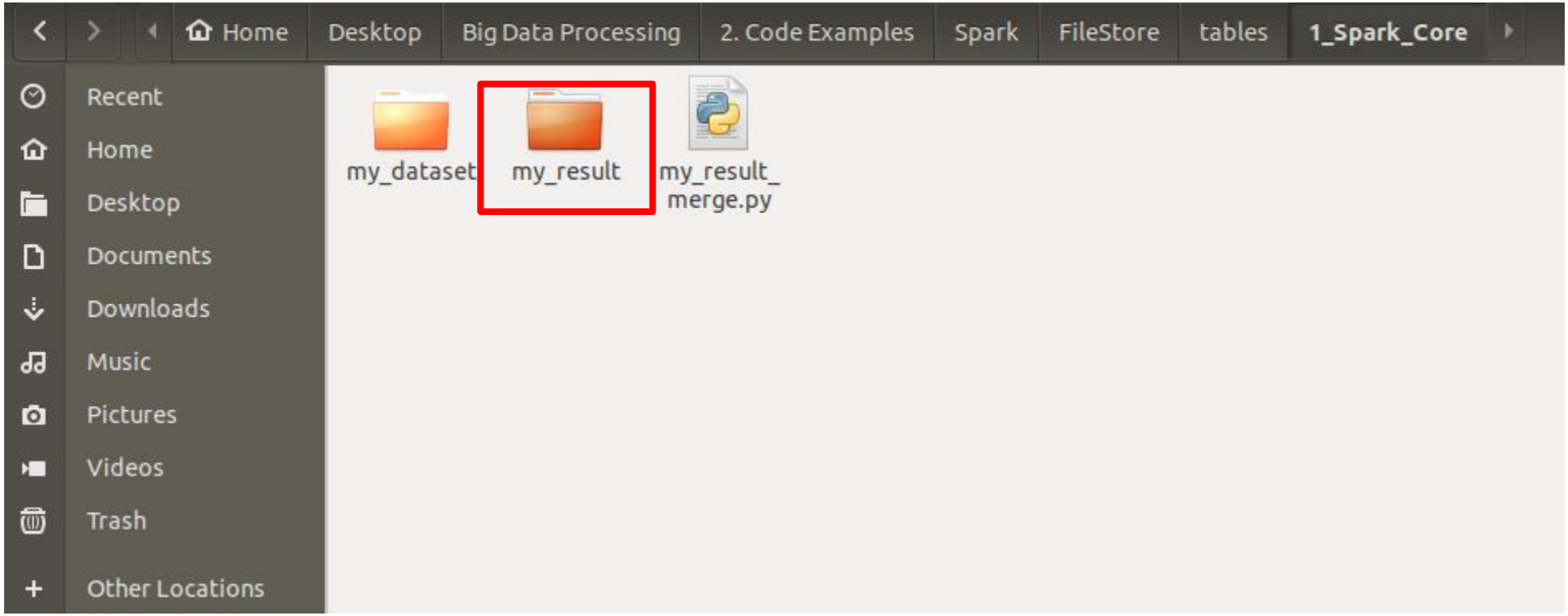
Databricks: An Online Platform for Data Engin

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



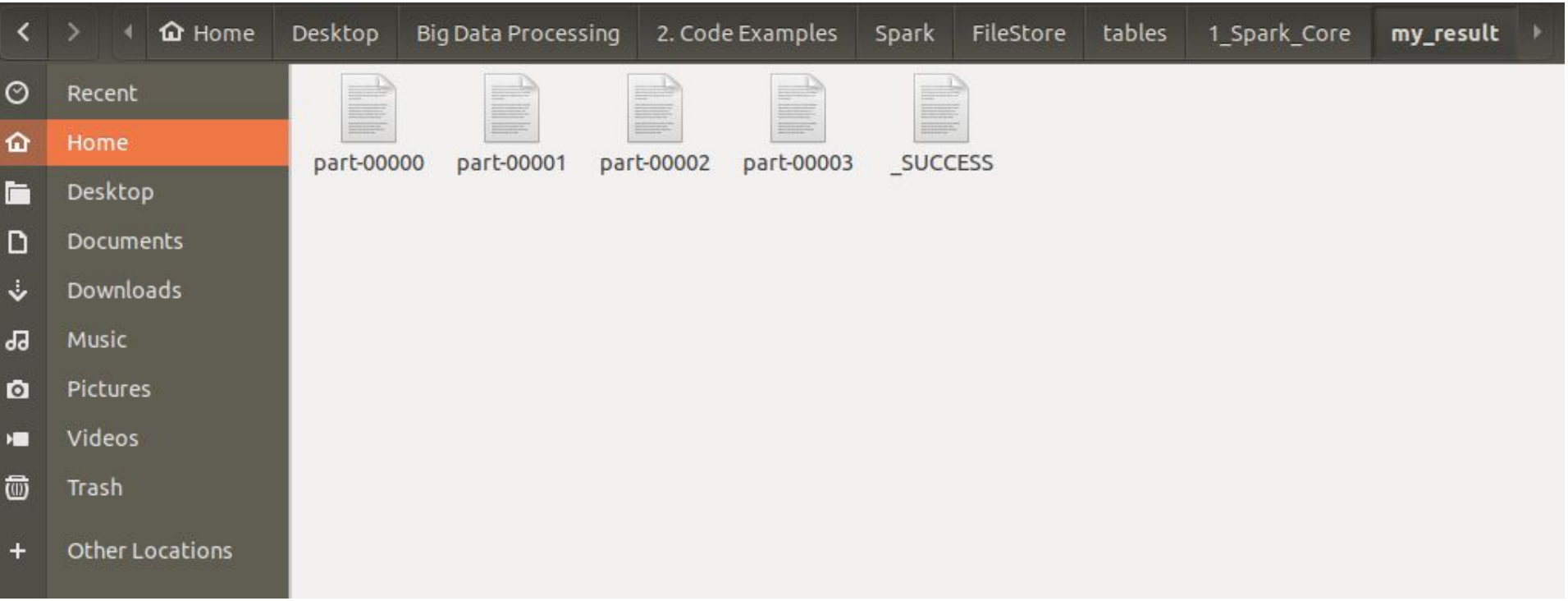
Databricks: An Online Platform for Data Engin

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



Databricks: An Online Platform for Data Engin

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 Engin

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 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.

Outline

1. Introduction to Apache Spark.
2. A Spark Application: General Overview.
3. Two Different Roles: IT Manager vs. Data Engineer.
4. Databricks: An Online Platform for Data Engineers.

Thank you for your attention!