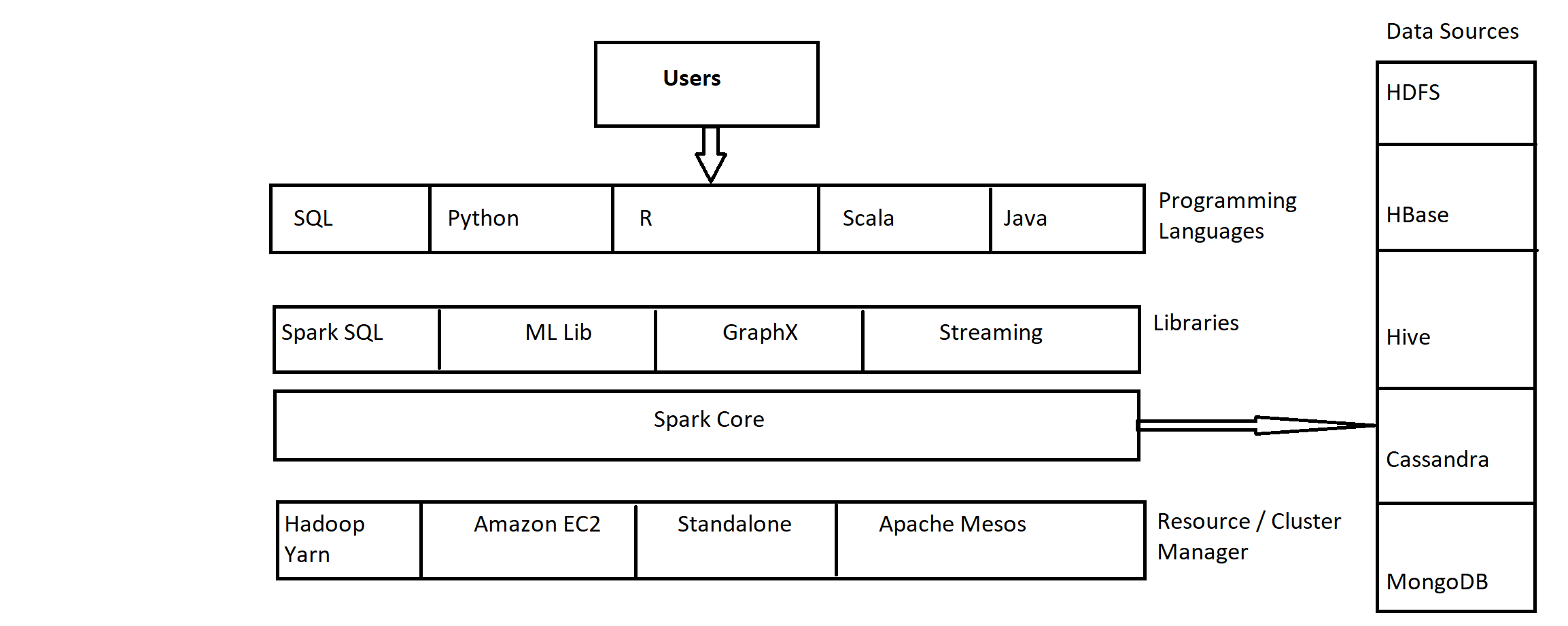
|  |  |  |
| --- | --- | --- |
| **UMBC – CSEE Department** |  | Dr. Waleed Youssef |
| **Data Science Program** |  | youssef1@umbc.edu |
| **Spring 2023** |  |  |
| **DATA 603 – Big Data Platforms** | | |
|  |  |  |
| **Homework #5 – Apache Spark** | | |
|  |  |  |

**Questions:**

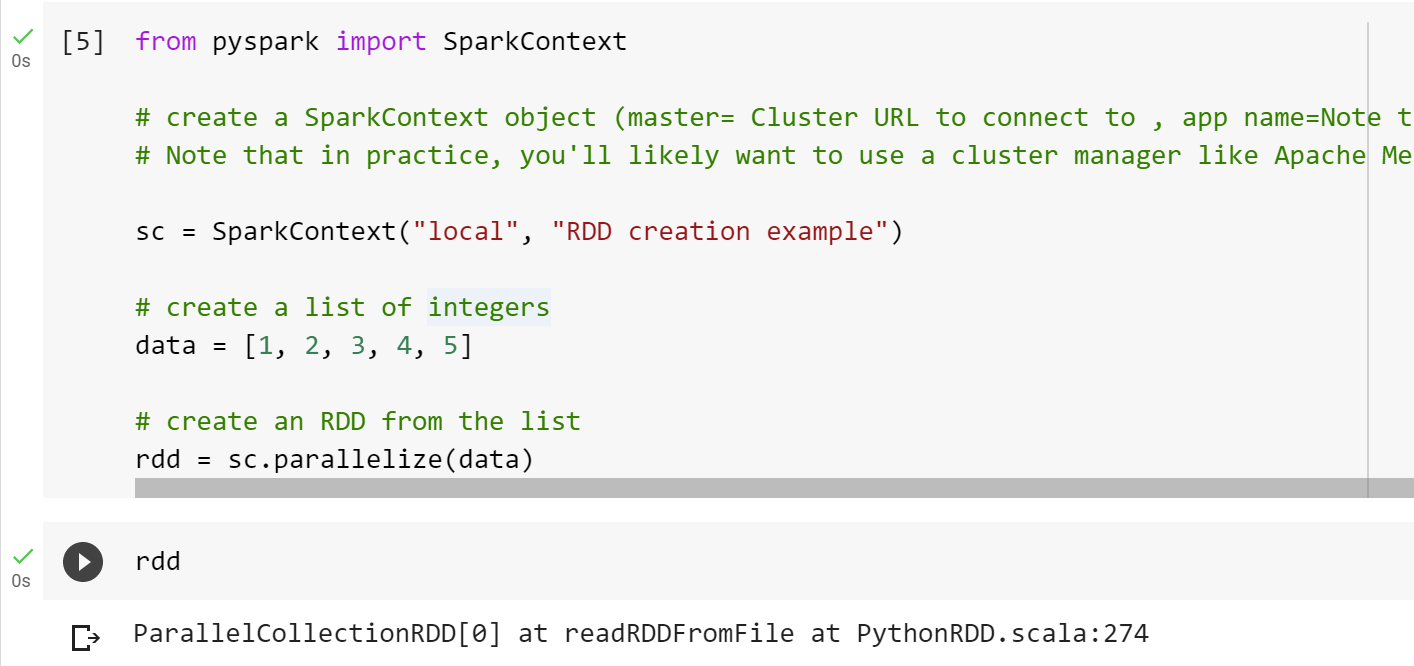
1. **[10 points]** Draw a detailed architecture diagram describing the operation of the Spark platform? 
2. **[10 points]** Describe RDDs? How are they created? What types of processing can be performed on them?

* Give 3 examples of each type of RDD processing.

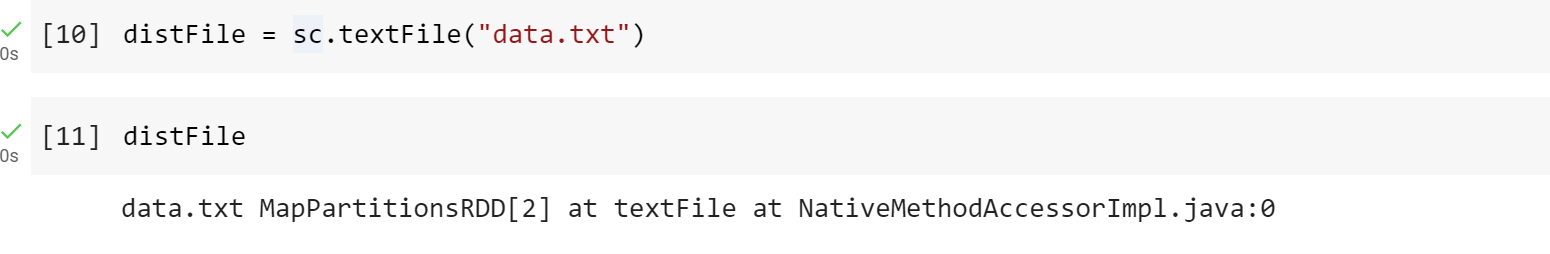
RDD is an immutable, fault-tolerant, and distributed collection of elements of your data, partitioned across nodes in your cluster that can be operated in parallel with a low-level API that offers transformations and actions.[1]

How are they Created?

Parallelizing an existing collection in driver program, or referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat() [2]

Example: Using Parallelizing 

Example 2: PySpark can create distributed datasets from any storage source supported by Hadoop, including your local file system, HDFS, Cassandra, HBase, Amazon S3, etc. Spark supports text files, SequenceFiles, and any other Hadoop InputFormat.



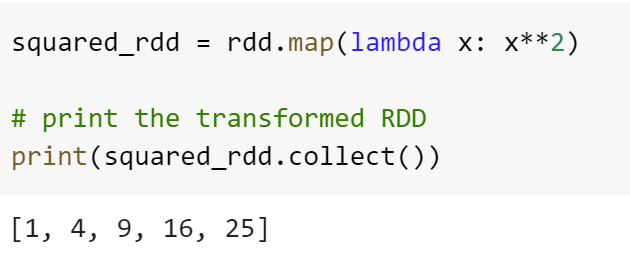
There are two types of RDD processing: Transformations and Actions. Here are three examples of each type of processing:

**Transformations**:

Operations applied on an RDD to create a new RDD. Examples include filter(), union(), map(), flatMap(), distinct(), reduceByKey() and sortBy() [3]

Examples:

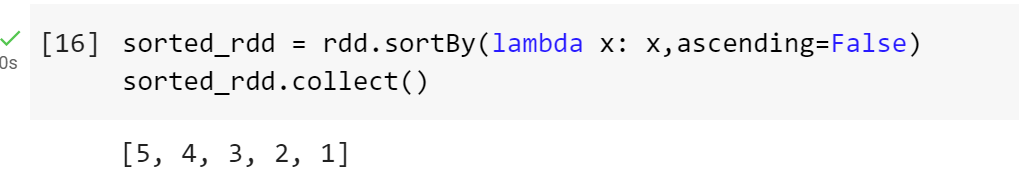
1. map(): Applies a function to each element in the RDD and returns a new RDD with the transformed values.



1. filter(): Returns a new RDD with only the elements that satisfy the given condition



1. sortBy(): Returns a new RDD with sorting all the values in RDD.



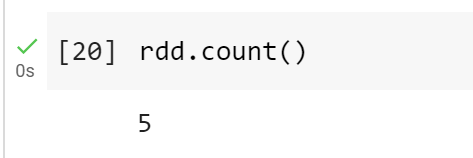
**Actions**:

These are the operations that are applied on RDD, which instructs Spark to perform computation and send the result back to the driver. [4] such as collect(),count(),reduce(), cache(), saveAsTextFile(), etc.

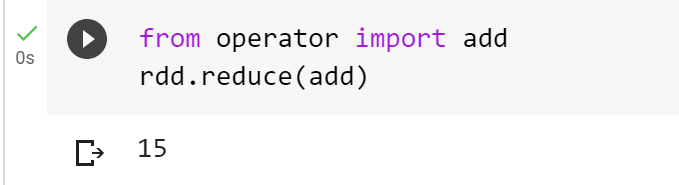
1. collect(): Returns a list that contains all of the elements in this RDD.



1. count(): Returns the total number of elements in RDD.



1. reduce(): Reduces the elements of this RDD using the specified commutative and associative binary operator.



1. **[10 points]** Compare Scala, Java, and Python in details? Describe which language you plan on using and why?

Scala: It stands for scalable language. It is an open-source, multi-paradigm, high-level programming language with a robust static type system. Scala is known for combining functional and object-oriented programming concepts, allowing for every value to be treated as an object and every operator to be implemented as a method. It supports modular composition and higher-order functions, and also allows for imperative and declarative programming. Scala is highly parallelizable and can run on the Java Virtual Machine, seamlessly inter-operating with Java. Its concise and statically typed nature makes it type-safe and efficient. Scala is considered a decisive tool for data science and machine learning at large scale due to its powerful language features and optimization of code complexity.

Spark gives Scala an edge over other programming languages to process streams in real-time. It has made Scala the computational engine for the fast data processing. Spark is written in Scala and runs on JVM. Scala has a variety of libraries available for Big Data analytics and Machine Learning. These include Spark Streaming for real-time data processing, Spark SQL for structured data processing, Spark MLlib for machine learning algorithms and Spark GraphX for graph analytics.[5]

Java: Java is a class-based object-oriented programming language for building web and desktop applications. It is the most popular programming language and the language of choice for Android programming. Each operating system has different JVM, however the output they produce after execution of bytecode is same across all operating systems. That is why we call java as platform independent language. It includes features such as Abstraction, encapsulation, inheritance, and polymorphism. Java is the base of many Big Data tools including Apache Hadoop, Spark, Storm, Mahout, and more. Developers can use it to develop an unlimited tech stack, because of its support for multithreading and scalability.

Python: Python is a dynamic, interpreted (bytecode-compiled) language. There are no type declarations of variables, parameters, functions, or methods in source code. This makes the code short and flexible, and you lose the compile-time type checking of the source code. Python comes with the Python Package Index (PIP), the open-source repository that contains all the third-party packages available for Python. This library consists of packages to help users in various tasks, from simple tasks like JSON parsing to complete data transformation, analytics, and visualizations packages. For data analytics it provide packages like pandas, numpy, scikit-learn, etc. For data visualization it has packages like matplotlib, seaborn, plotly, others. It also supports various bigdata related packages like pyspark, others. All of which makes development much more convenient than any other programming languages. Python provides features for identifying and processing unstructured data which can include voice, text, and image data as well.

Python and Hadoop are open-source big data platforms and that’s why Python is securely compatible with Hadoop. Most developers prefer to use Python along with Hadoop rather than Java or Scala because of the huge amount of Python supporting libraries for data analytics. Python also has the PyDoop Package which provides excellent support for Hadoop to Python developers. Pydoop package provides access to the HDFS API for Hadoop which allows you to read and write data files from global file systems. Pydoop also provides the MapReduce API which is used for solving complex data science concepts using minimal programming efforts which is the hallmark of Python. This is also an excellent reason to choose Python over other programming languages for Big Data. [6]

I prefer Python over Java or Scala as it has great libraries and make development much easier and it is makes it easier to learn new features and capabilities of python. It has a large community which can help when you are struck and need a solution. Python can be easily integrated with other programming languages and big data technologies, such as Hadoop, Spark, and Hive. This makes it easier for developers to use Python for specific tasks within a larger big data ecosystem. Python is a versatile language that can be used for many different purposes, including web development, scientific computing, data analysis, and machine learning. This versatility makes it easier for developers to use a single language across multiple parts of the big data pipeline.

1. **[20 points]** Write a Spark program to count the number of images in a URL and then display the URLs of these images.

For example, the program should read the URL 🡪 [www.cnn.com](http://www.cnn.com)

Then, it should display the output as:

1. There are <n> images at the <<URL>> site
2. The images are:

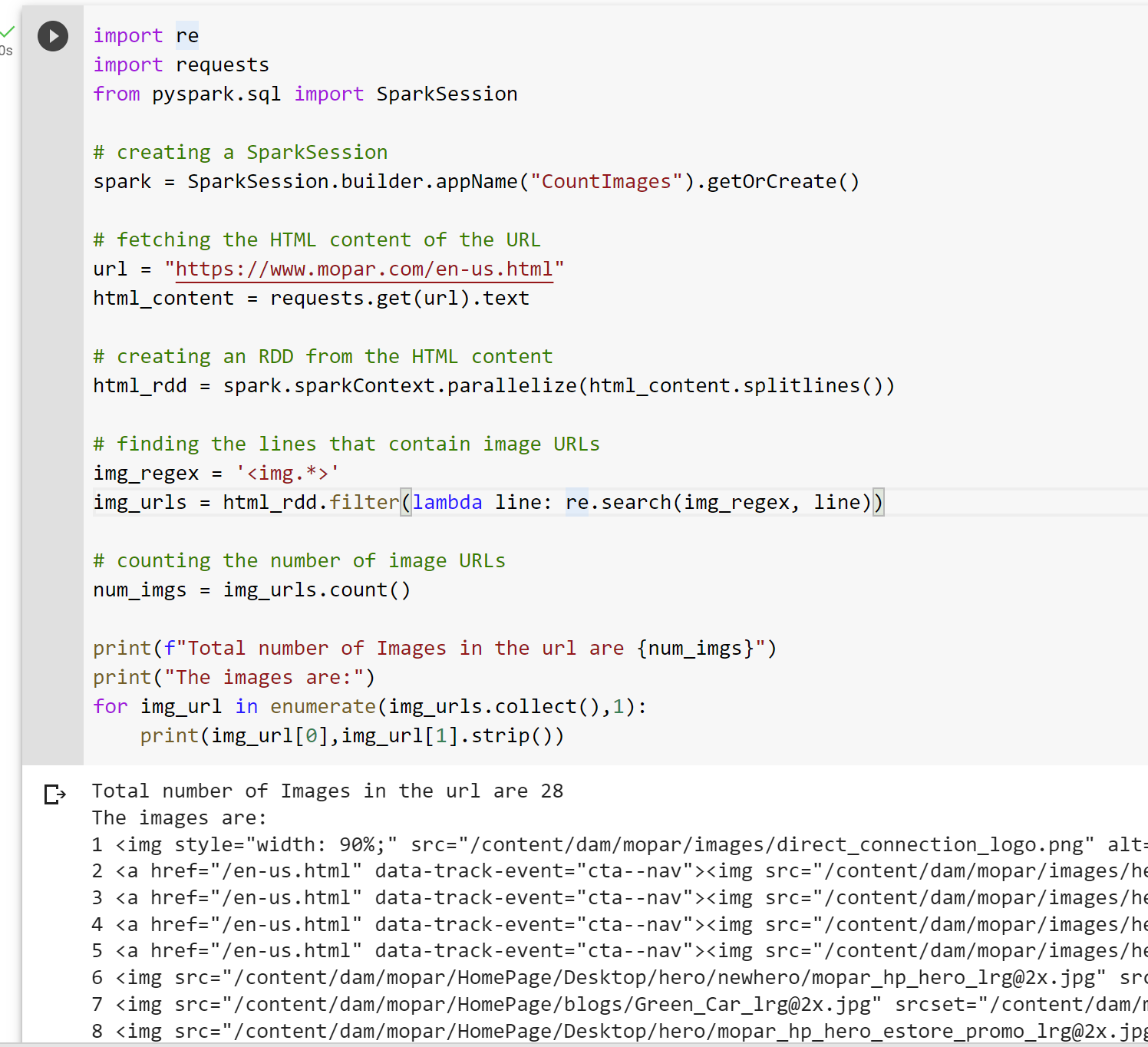
<<Image URL 1>>

<<Image URL 2>>

<<Image URL 3>>

…

<<Image URL n>>





**References:**

[1] What is a Resilient Distributed Dataset (RDD)? (n.d.). Databricks. from <https://www.databricks.com/glossary/what-is-rdd>

[2] RDD Programming Guide - Spark 3.3.2 Documentation. (n.d.). Apache Spark. from <https://spark.apache.org/docs/latest/rdd-programming-guide.html#parallelized-collections>

[3] Apache Spark RDD | Understanding the Basics of Apache Spark RDD. (2021, August 30). Analytics Vidhya. from <https://www.analyticsvidhya.com/blog/2021/08/understanding-the-basics-of-apache-spark-rdd/>

[4] *PySpark - RDD*. (n.d.). Tutorialspoint. from <https://www.tutorialspoint.com/pyspark/pyspark_rdd.htm>

[5] 4 most used languages in big data projects: Scala. (n.d.). ITNEXT. from <https://itnext.io/4-most-used-languages-in-big-data-projects-scala-59ff02e707c9>

[6] Kiran. (2020, May 2). 10 Reasons Why You Should Choose Python For Big Data. GeeksforGeeks. from <https://www.geeksforgeeks.org/10-reasons-why-you-should-choose-python-for-big-data/>