DXC AZURE ANALYTICS

Assignment-6

Name : Nandi Vomkara Aditya Mohan   Date of submission : 06-06-2022

ID : DXCAB1203   batch: Azure Analytics

1.Explain what is in-memory computation in detail?

Ans)

In-memory computation works by eliminating all slow data accesses and relying exclusively on data stored in RAM. Overall computation performance is greatly improved by removing the latency commonly seen when accessing hard disk drives or SSDs. Software running on one or more computers manages the computation as well as the data in memory, and in the case of multiple computers, the software divides the computation into smaller tasks which are distributed out to each computer to run in parallel. In-memory computation is often done in the technology known as in memory data grids.

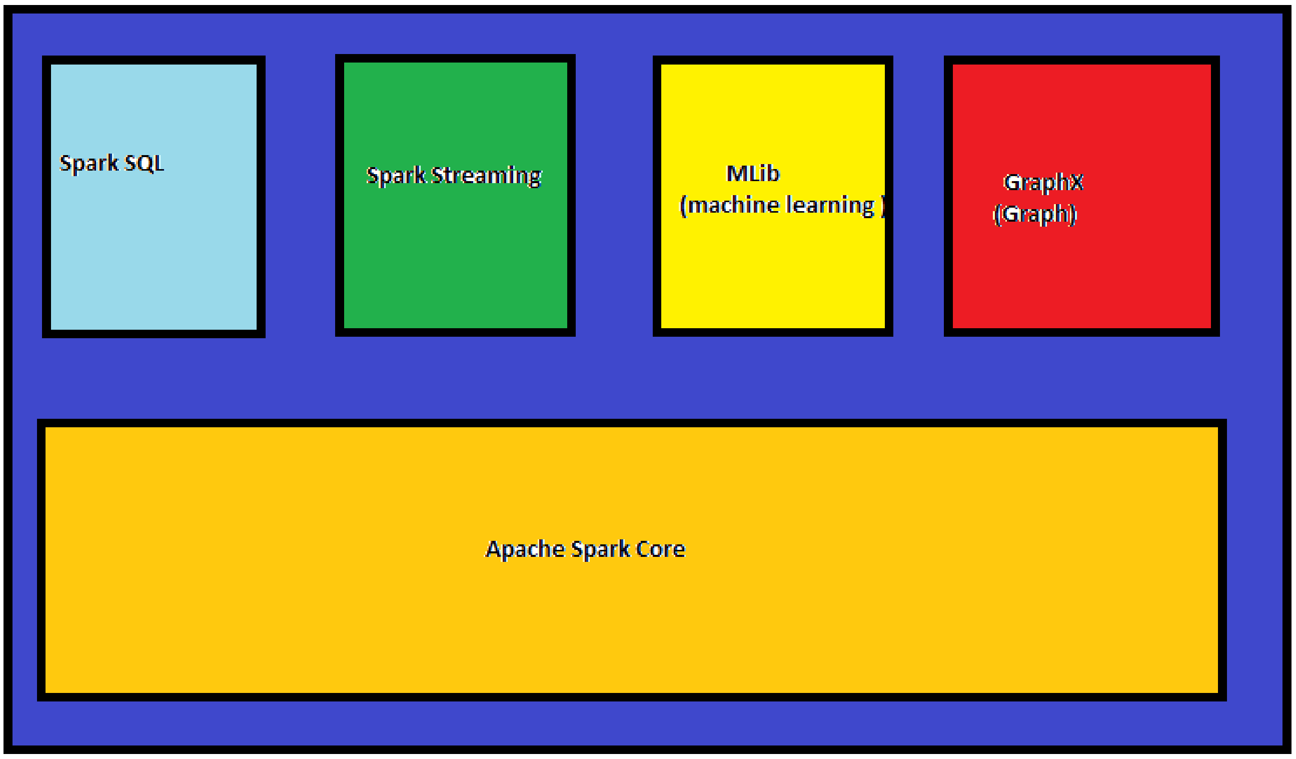
2.Explain the advantage of spark framework?

Ans) advantages of Apache Spark:

- Speed : When comes to Big Data, processing speed always matters. Apache Spark is wildly popular with data scientists because of its speed. Spark is 100x faster than Hadoop for large scale data processing. Apache Spark uses in-memory(RAM) computing system whereas Hadoop uses local memory space to store data. Spark can handle multiple petabytes of clustered data of more than 8000 nodes at a time.
- Ease of Use : Apache Spark carries easy-to-use APIs for operating on large datasets. It offers over 80 high-level operators that make it easy to build parallel apps.
- Advanced Analytics : Spark not only supports 'MAP' and 'reduce'. It also supports Machine learning (ML), Graph algorithms, Streaming data, SQL queries, etc.
- Dynamic in Nature : With Apache Spark, you can easily develop parallel applications. Spark offers you over 80 high-level operators.
- Multilingual : Apache Spark supports many languages for code writing such as Python, Java, Scala, etc.
- Apache Spark is powerful: Apache Spark can handle many analytics challenges because of its low-latency in-memory data processing capability. It has well-built libraries for graph analytics algorithms and machine learning.
- Open-source community : The best thing about Apache Spark is, it has a massive Open-source community behind it.

3.Explain the components of spark with block diagram?

Ans)



Apache Spark Core

Spark Core is the underlying general execution engine for spark platform that all other functionality is built upon. It provides In-Memory computing and referencing datasets in external storage systems.

Spark SQL

Spark SQL is a component on top of Spark Core that introduces a new data abstraction called SchemaRDD, which provides support for structured and semi-structured data.

Spark Streaming

Spark Streaming leverages Spark Core's fast scheduling capability to perform streaming analytics. It ingests data in mini-batches and performs RDD (Resilient Distributed Datasets) transformations on those mini-batches of data.

MLlib (Machine Learning Library)

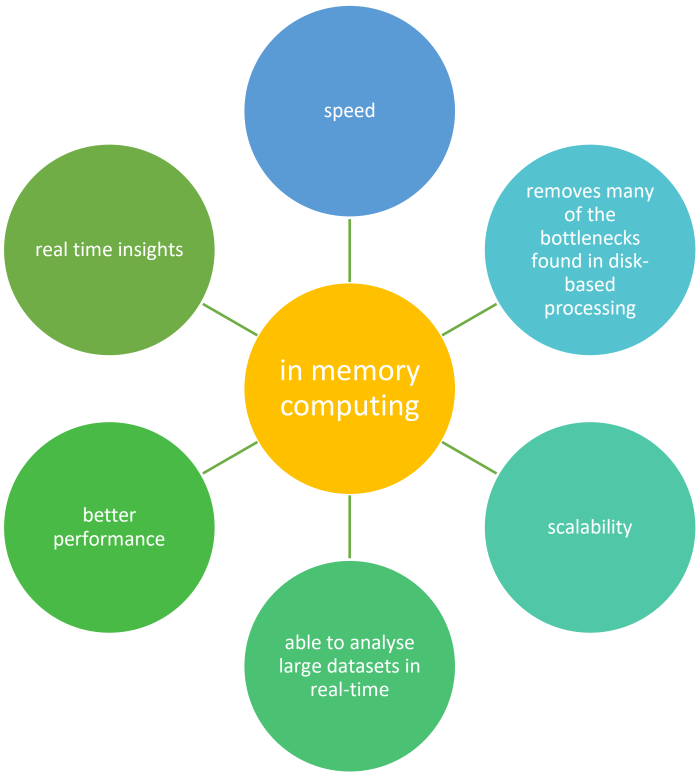
MLlib is a distributed machine learning framework above Spark because of the distributed memory-based Spark architecture. It is, according to benchmarks, done by the MLlib developers against the Alternating Least Squares (ALS) implementations. Spark MLlib is nine times as fast as the Hadoop disk-based version of Apache Mahout (before Mahout gained a Spark interface).

GraphX

GraphX is a distributed graph-processing framework on top of Spark. It provides an API for expressing graph computation that can model the user-defined graphs by using Pregel abstraction API. It also provides an optimized runtime for this abstraction.

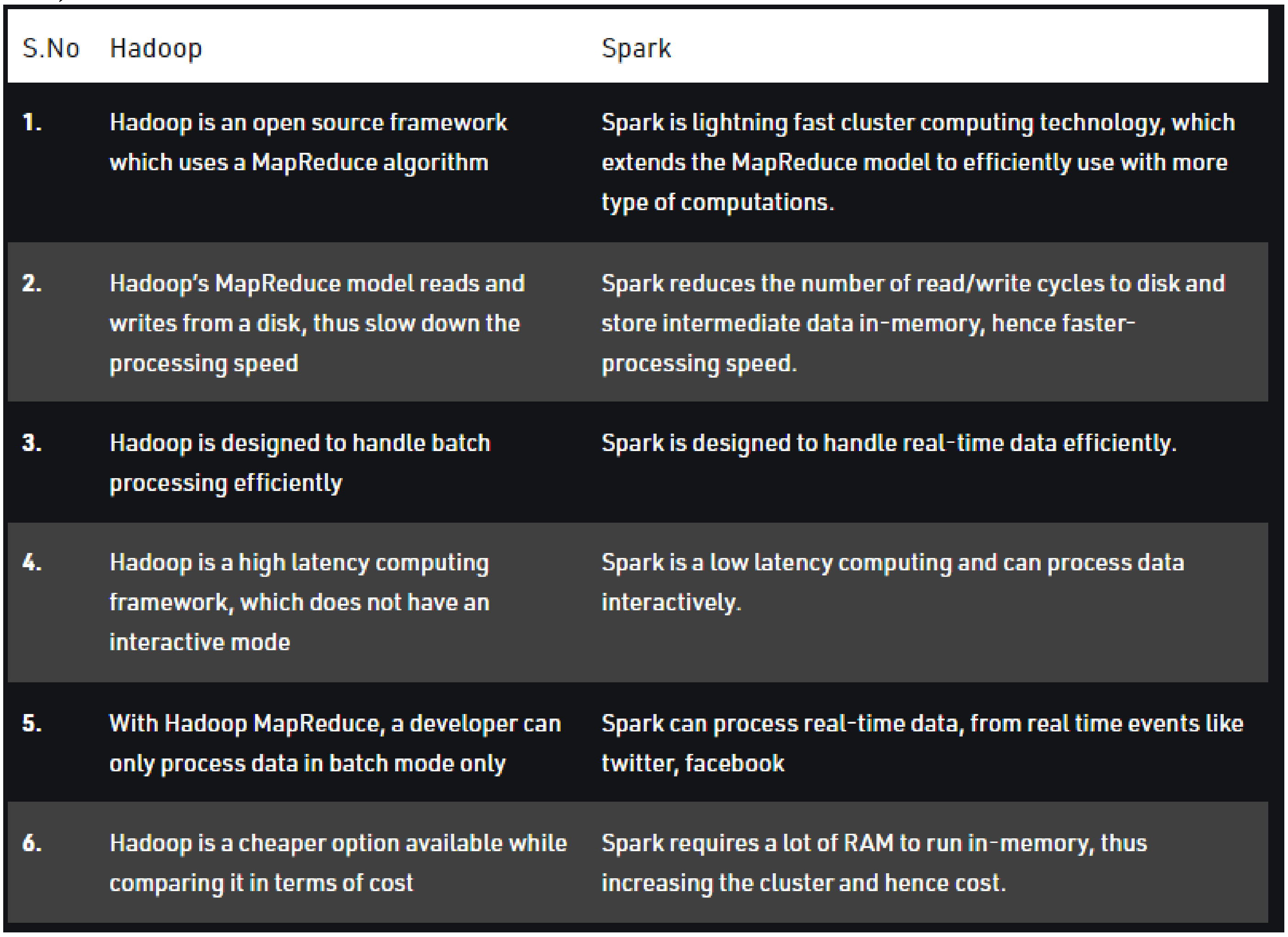
4.Explain benefits of in memory computation?

Ans)


speed
real time insights
removes many of the bottlenecks found in disk-based processing
in memory computing
better performance
scalability
able to analyse large datasets in real-time

5. explain major Difference between Hadoop & spark

Ans)



S.No	Hadoop	Spark
1.	Hadoop is an open source framework which uses a MapReduce algorithm	Spark is lightning fast cluster computing technology, which extends the MapReduce model to efficiently use with more type of computations.
2.	Hadoop's MapReduce model reads and writes from a disk, thus slow down the processing speed	Spark reduces the number of read/write cycles to disk and store intermediate data in-memory, hence faster-processing speed.
3.	Hadoop is designed to handle batch processing efficiently	Spark is designed to handle real-time data efficiently.
4.	Hadoop is a high latency computing framework, which does not have an interactive mode	Spark is a low latency computing and can process data interactively.
5.	With Hadoop MapReduce, a developer can only process data in batch mode only	Spark can process real-time data, from real time events like twitter, facebook
6.	Hadoop is a cheaper option available while comparing it in terms of cost	Spark requires a lot of RAM to run in-memory, thus increasing the cluster and hence cost.

6.explain features of spark

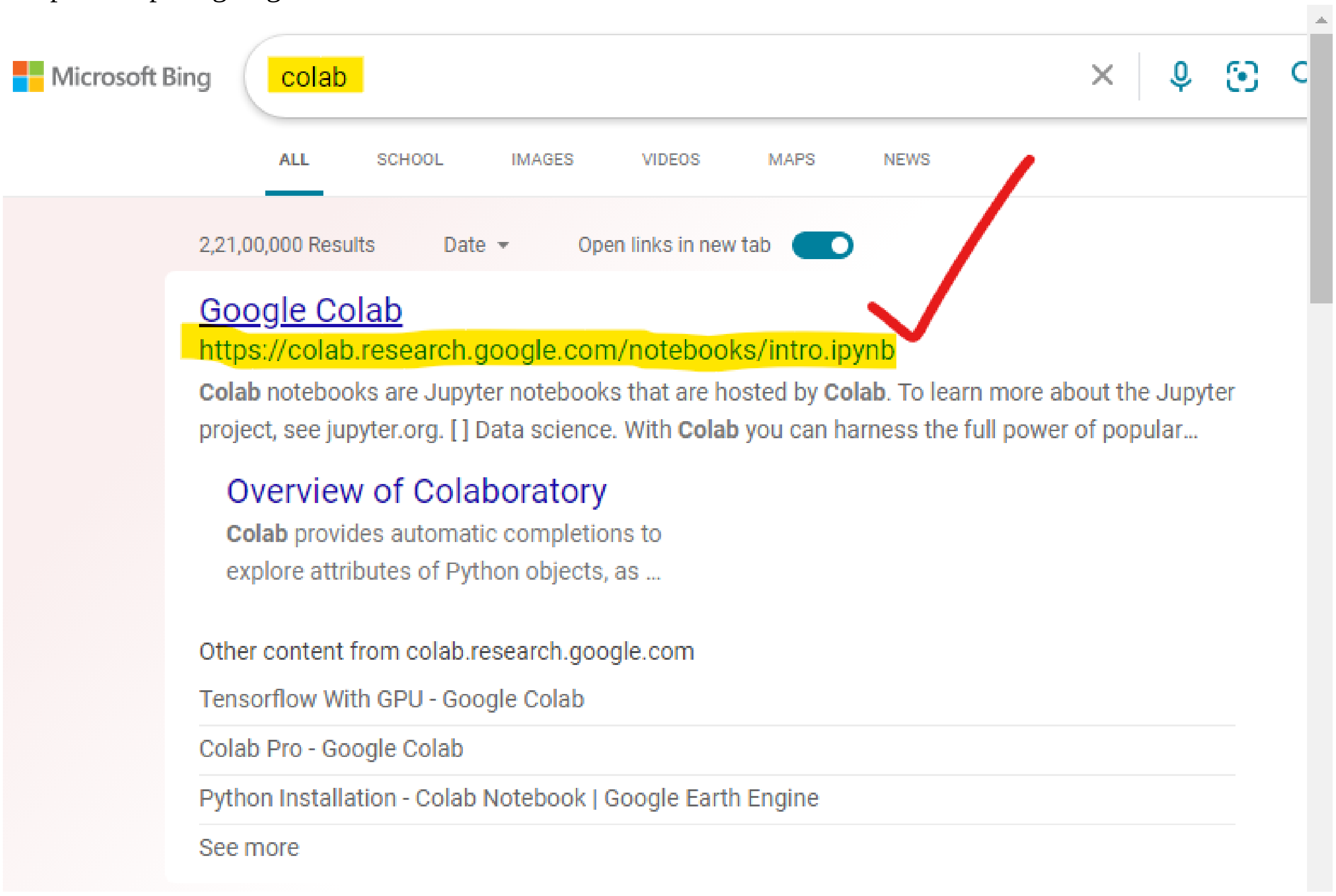
Ans) The below mentioned are the some of the features of spark

- Reusability
- Speed
- Advanced analytics
- In memory computing
- Real time stream processing
- Lazy evaluation
- Dynamic in nature
- Fault tolerance

7.write a py-spark program to create data frame from RDD & explain with screen shots & steps

Ans) these are the following steps to create a data frame from RDD

Step-1 : open google colab

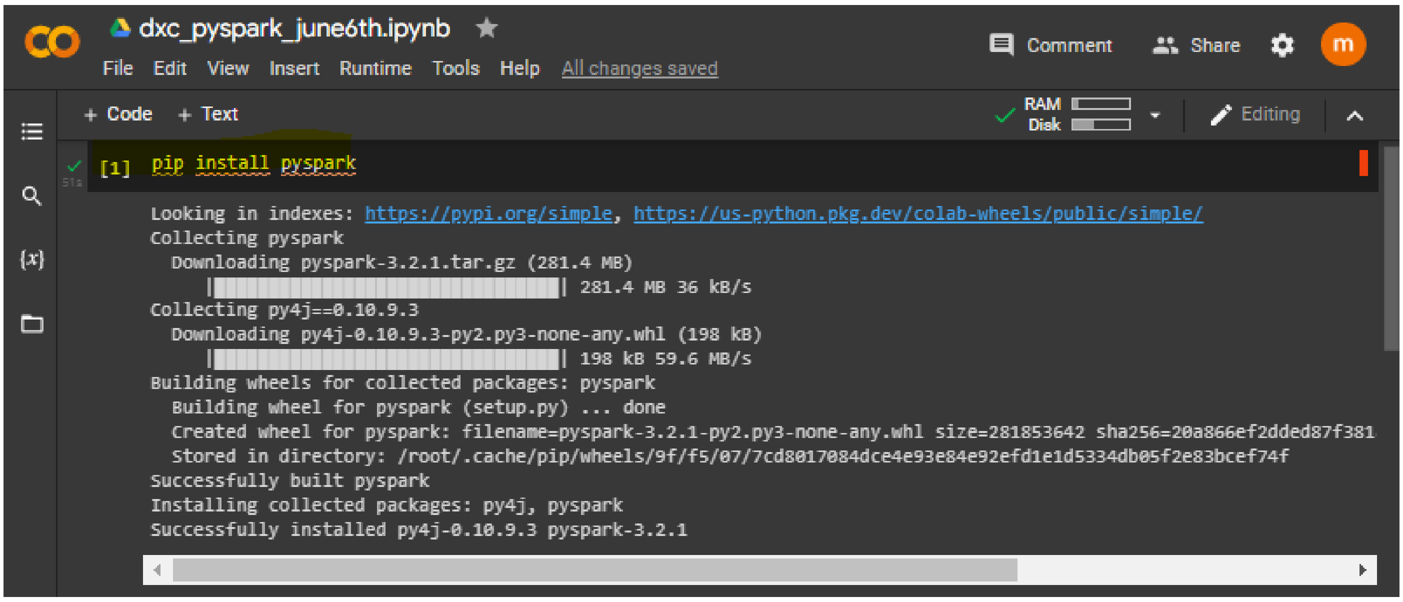


Google Colab
https://colab.research.google.com/notebooks/intro.ipynb
Colab notebooks are Jupyter notebooks that are hosted by **Colab**. To learn more about the Jupyter project, see jupyter.org. [ ] Data science. With **Colab** you can harness the full power of popular...

Overview of Colaboratory
Colab provides automatic completions to explore attributes of Python objects, as ...

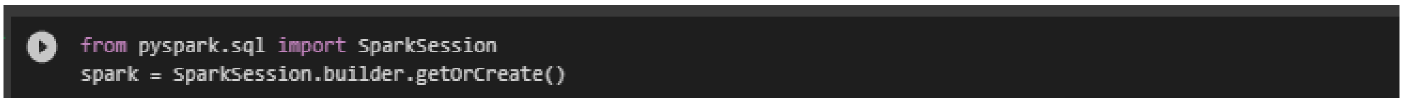
Other content from colab.research.google.com
Tensorflow With GPU - Google Colab
Colab Pro - Google Colab
Python Installation - Colab Notebook | Google Earth Engine
See more

Step 2: open a new notebook and install pyspark using pip command


dxc_pyspark_june6th.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Comment
Share
+ Code + Text
RAM
Disk
Editing

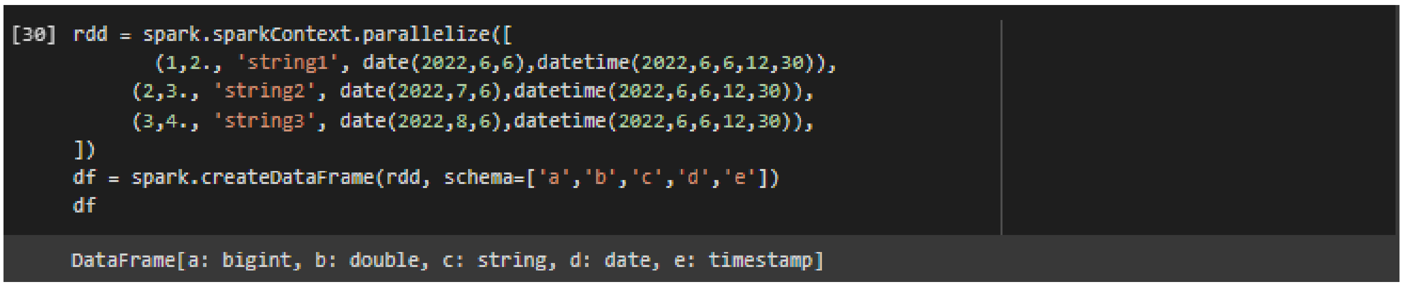
```
[1] pip install pyspark

    Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
    Collecting pyspark
      Downloading pyspark-3.2.1.tar.gz (281.4 MB)
         |████████████████████████████████| 281.4 MB 36 kB/s
    Collecting py4j==0.10.9.3
      Downloading py4j-0.10.9.3-py2.py3-none-any.whl (198 kB)
         |████████████████████████████████| 198 kB 59.6 MB/s
    Building wheels for collected packages: pyspark
      Building wheel for pyspark (setup.py) ... done
      Created wheel for pyspark: filename=pyspark-3.2.1-py2.py3-none-any.whl size=281853642 sha256=20a866ef2dded87f381
      Stored in directory: /root/.cache/pip/wheels/9f/f5/07/7cd8017084dce4e93e84e92efd1e1d5334db05f2e83bcef74f
    Successfully built pyspark
    Installing collected packages: py4j, pyspark
    Successfully installed py4j-0.10.9.3 pyspark-3.2.1
```

Step 3 :import spark session from spark.sql



```
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()
```

Step 4:



```
[30] rdd = spark.sparkContext.parallelize([
            (1,2., 'string1', date(2022,6,6),datetime(2022,6,6,12,30)),
          (2,3., 'string2', date(2022,7,6),datetime(2022,6,6,12,30)),
          (3,4., 'string3', date(2022,8,6),datetime(2022,6,6,12,30)),
     ])
     df = spark.createDataFrame(rdd, schema=['a','b','c','d','e'])
     df

     DataFrame[a: bigint, b: double, c: string, d: date, e: timestamp]
```

8.Explain what is RDD & why it is needed

Ans) Resilient Distributed Datasets (RDD) is a fundamental data structure of Spark. It is an immutable distributed collection of objects. Each dataset in RDD is divided into logical partitions, which may be computed on different nodes of the cluster. RDDs can contain any type of Python, Java, or Scala objects, including user-defined classes.

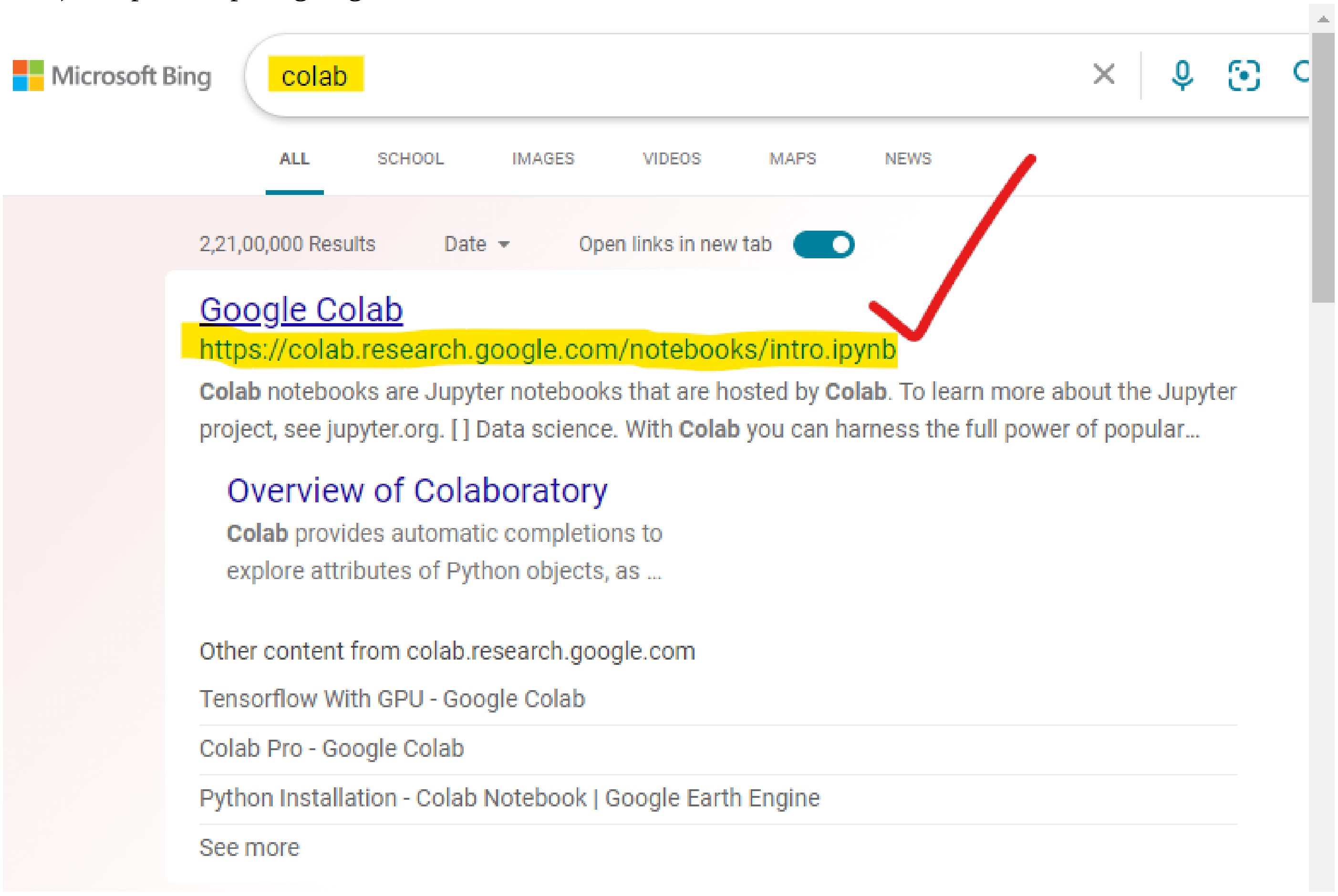
Formally, an RDD is a read-only, partitioned collection of records. RDDs can be created through deterministic operations on either data on stable storage or other RDDs. RDD is a fault-tolerant collection of elements that can be operated on in parallel.

There are two ways to create RDDs − parallelizing an existing collection in your driver program, or referencing a dataset in an external storage system, such as a shared file system, HDFS, HBase, or any data source offering a Hadoop Input Format.

And it is used for Spark makes use of the concept of RDD to achieve faster and efficient MapReduce operations. Let us first discuss how MapReduce operations take place and why they are not so efficient.

9.write a py-spark program to make a column in uppercase & explain with screen shots & steps.

Ans) Step-1 : open google colab

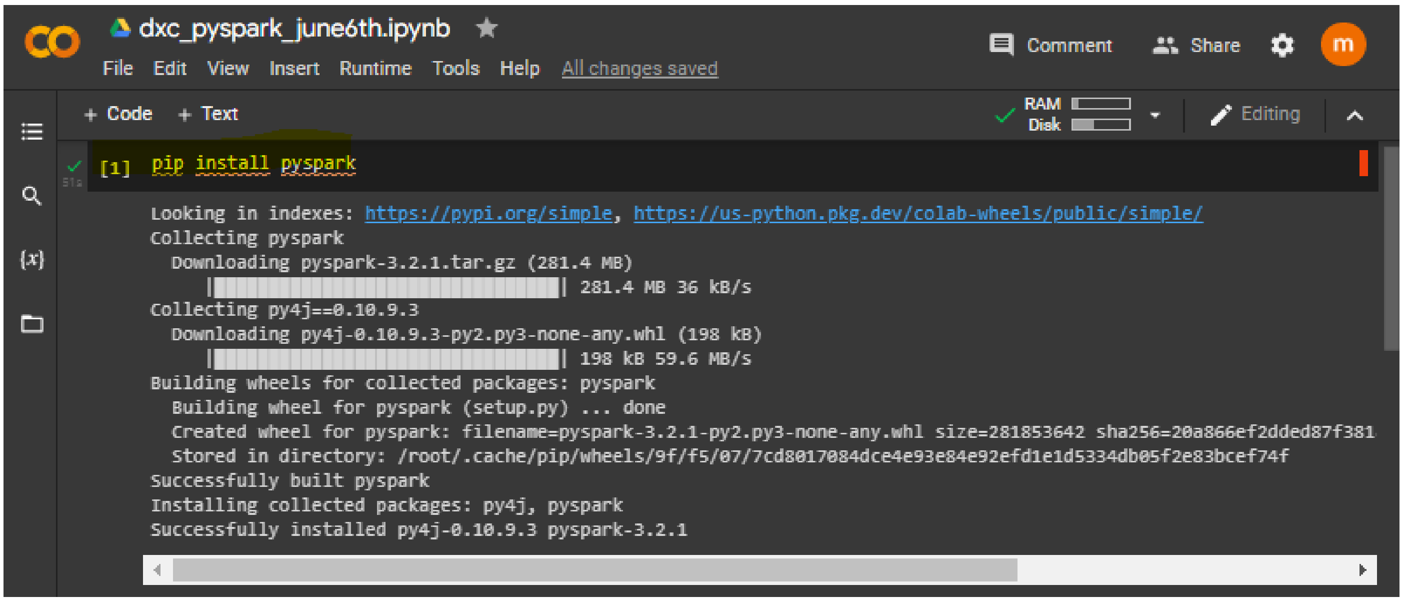


Google Colab
https://colab.research.google.com/notebooks/intro.ipynb
Colab notebooks are Jupyter notebooks that are hosted by Colab. To learn more about the Jupyter project, see jupyter.org. [ ] Data science. With Colab you can harness the full power of popular...

Overview of Colaboratory
Colab provides automatic completions to explore attributes of Python objects, as ...

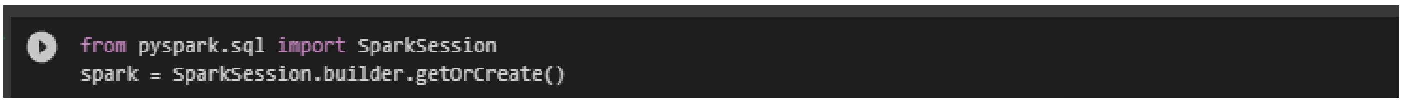
Other content from colab.research.google.com
Tensorflow With GPU - Google Colab
Colab Pro - Google Colab
Python Installation - Colab Notebook | Google Earth Engine
See more

Step 2: open a new notebook and install pyspark using pip command



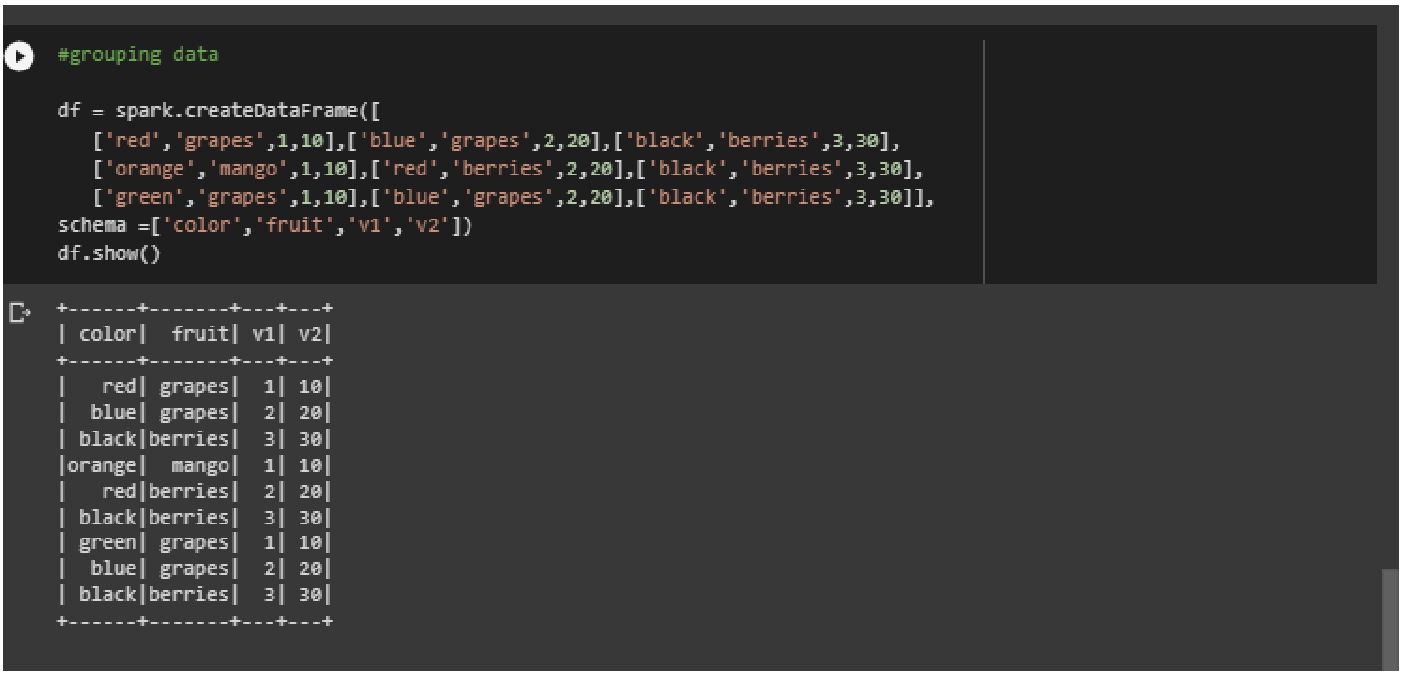
```
[1] pip install pyspark

Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pyspark
  Downloading pyspark-3.2.1.tar.gz (281.4 MB)
     |████████████████████████████████| 281.4 MB 36 kB/s
Collecting py4j==0.10.9.3
  Downloading py4j-0.10.9.3-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 59.6 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.2.1-py2.py3-none-any.whl size=281853642 sha256=20a866ef2dded87f381
  Stored in directory: /root/.cache/pip/wheels/9f/f5/07/7cd8017084dce4e93e84e92efd1e1d5334db05f2e83bcef74f
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9.3 pyspark-3.2.1
```

Step 3 :import spark session from spark.sql



```
from pyspark.sql import SparkSession
spark = SparkSession.builder.getOrCreate()
```

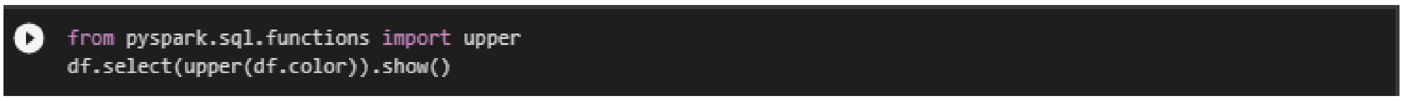
Step 4: to perform this case we are creating a data frame



```
#grouping data

df = spark.createDataFrame([
   ['red','grapes',1,10],['blue','grapes',2,20],['black','berries',3,30],
   ['orange','mango',1,10],['red','berries',2,20],['black','berries',3,30],
   ['green','grapes',1,10],['blue','grapes',2,20],['black','berries',3,30]],
schema =['color','fruit','v1','v2'])
df.show()
```

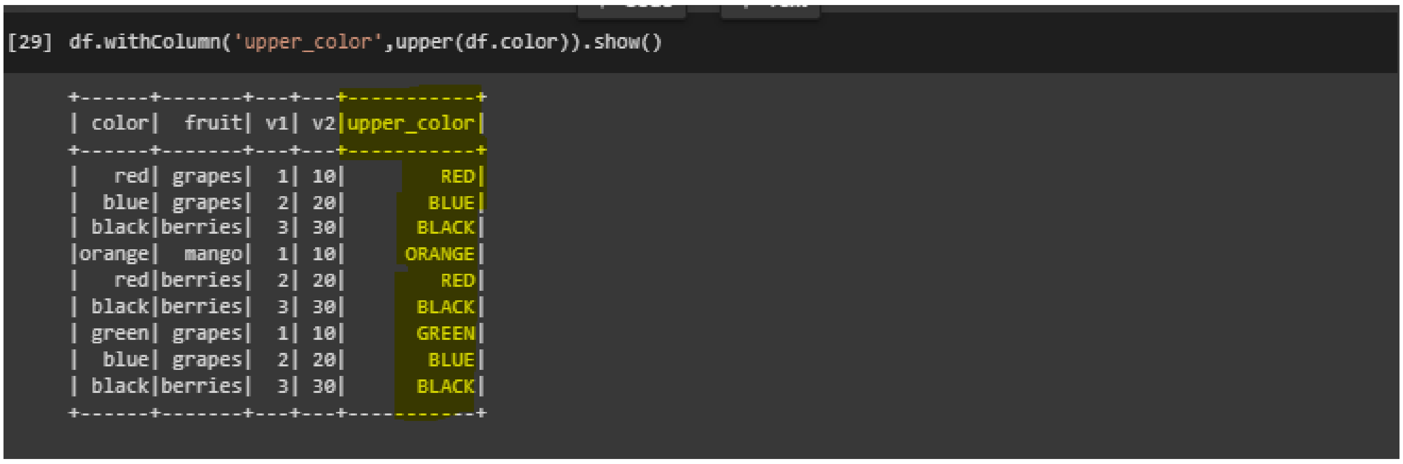
```
+------+-------+---+---+
| color|  fruit| v1| v2|
+------+-------+---+---+
|   red| grapes|  1| 10|
|  blue| grapes|  2| 20|
| black|berries|  3| 30|
|orange|  mango|  1| 10|
|   red|berries|  2| 20|
| black|berries|  3| 30|
| green| grapes|  1| 10|
|  blue| grapes|  2| 20|
| black|berries|  3| 30|
+------+-------+---+---+
```

Step 5: To convert column into upper case we have to import upper command fom pyspark.sql.functions



```
from pyspark.sql.functions import upper
df.select(upper(df.color)).show()
```

Step 6: The final output can be clearly demonstrated as by the below mentioned statements and the final output is mentioned in the below mentioned screen shot



```
[29] df.withColumn('upper_color',upper(df.color)).show()

+------+-------+---+---+-----------+
| color|  fruit| v1| v2|upper_color|
+------+-------+---+---+-----------+
|   red| grapes|  1| 10|        RED|
|  blue| grapes|  2| 20|       BLUE|
| black|berries|  3| 30|      BLACK|
|orange|  mango|  1| 10|     ORANGE|
|   red|berries|  2| 20|        RED|
| black|berries|  3| 30|      BLACK|
| green| grapes|  1| 10|      GREEN|
|  blue| grapes|  2| 20|       BLUE|
| black|berries|  3| 30|      BLACK|
+------+-------+---+---+-----------+
```