NAME: BHOGADI NAGA ISWARYA LAKSHMI **SUBMISSION**:06-06-2022

BATCH: DXC-262-ANALYTICS-B12-AZURE **ASSESSMENT**:6

1. Explain what is in-Memory computation in details?

In-memory computation (or in-memory computing) is the technique of running computer calculations entirely in computer memory e.g., in RAM. This term typically implies large-scale, complex calculations which require specialized systems software to run the calculations on computers working together in a cluster. As a cluster, the computers pool together their RAM so the calculation is essentially run across computers and leverages the collective RAM space of all the computers together.

2. Explain advantages of Spark framework ?

- Simple to write
- Framework Handles Errors
- Algorithms
- Libraries
- God Local Tools
- Learning Curve
- Ease of Use

3. Explain components of Spark with block diagram ?

Shark: Shark is one of the Spark Ecosystem components. It is used to perform structured data analysis, especially if the data is too voluminous. Shark also allows running unmodified Hive queries on existing Hadoop deployment.

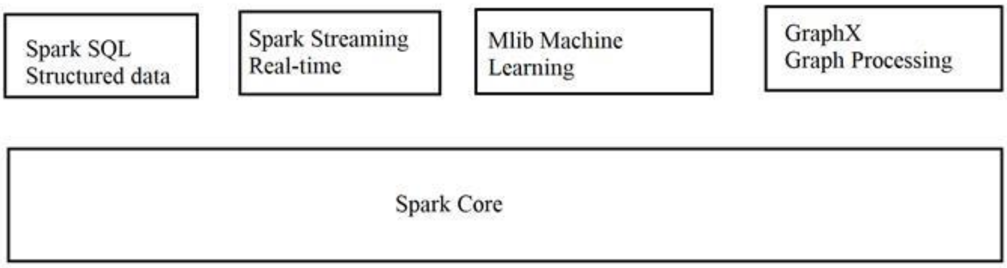
Spark Streaming: Spark Streaming is one of those unique features, which have empowered Spark to potentially take the role of Apache Storm. Spark Streaming mainly enables you to create analytical and interactive applications for live streaming data. You can do the streaming of the data and then, Spark can run its operations from the streamed data itself.

MLLib:MLLib is a machine learning library like Mahout. It is built on top of Spark, and has the provision to support many machine learning algorithms. But the point difference with Mahout is that it runs almost 100 times faster than MapReduce. It is not yet as enriched as Mahout, but it is coming up pretty well, even though it is still in the initial stage of growth.

GraphX: For graphs and graphical computations, Spark has its own Graph Computation Engine, called GraphX. It is similar to other widely used graph processing tools or databases, like Neo4j, Girafe, and many other distributed graph databases.

SparkR: There are many people from data science track, who must be aware that for statistical analysis, R is among the best. There is already an integration of R with Hadoop. Now, SparkR is a package for R language to enable R users to leverage the power of Spark from R shell.

Components of Apache Spark:



4. Explain benifits of in-Memory computation ?

- Better,faster,decision making
- Ability to reduce cost
- Identify competitive opportunities
- Grow revenue
- More efficient application
- Reduce risk
- It's best suited for performing real-time analytics, and developing and deploying real-time applications
- In-memory computing imperative:

  Avoid movement of detailed data.

  Calculate first, then move to results.

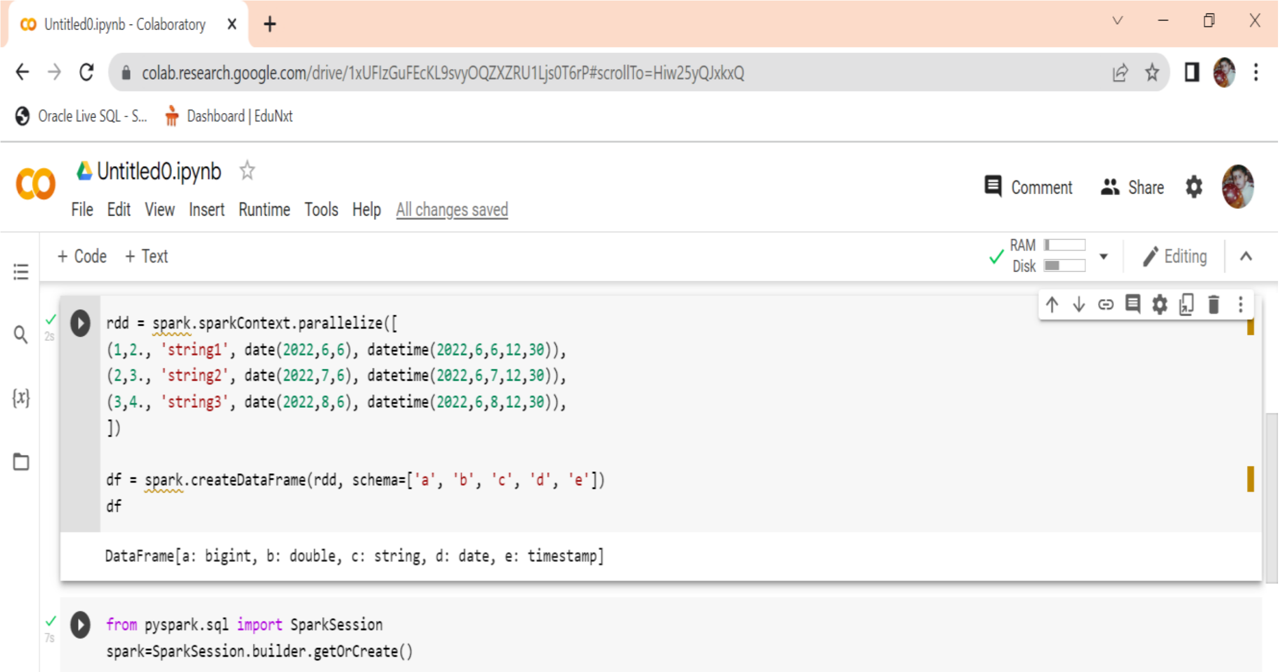
5. Explain major difference between Hadoop & Spark ?

HADOOP	SPARK
1.)Hadoop is an Apache open source framework that allows distributed processing of large data sets across clusters of computers using simple programming models.	1.)An open-source distributed general-purpose cluster-computing framework.
2.)Not a fast.	2.)Faster.
3.)Uses replication of data in multiple copies to achieve fault tolerance.	3.)Uses Resilient Distributed Dataset(RDD) for fault tolerance.
4.)Used to boost the Hadoop computational process	4.)Used to manage data storing and processing of big data applications running in clustered systems.

6. Explain features of Spark?

1. **Fast** :It provides high performance for both batch and streaming data, using a state-of-the-art DAG scheduler, a query optimizer, and a physical execution engine.
2. **Easy to Use**: It supports various languages like Java, Python, Scala, Sql, R, It facilitates to write the application in Java, Scala,Python, R, and SQL. It also provides more than 80 high-level operators.
3. **Supports Various Libraries:** It provides a collection of libraries including SQL and DataFrames, MLlib for machine learning,GraphX, and Spark Streaming.
4. **Supports Realtime Streaming**
5. **Lightweight** : It is a light unified analytics engine which is used for large scale data processing.
6. **-Runs Everywhere :** It can easily run on Hadoop, Apache Mesos, Kubernetes, standalone, or in the cloud.

7. Write a Py-Spark program to create Dataframe from RDD & explain with screenshots
& steps ?



```
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(2022,6,7,12,30)),
(3,4., 'string3', date(2022,8,6), datetime(2022,6,8,12,30)),
])

df = spark.createDataFrame(rdd, schema=['a', 'b', 'c', 'd', 'e'])
df
```

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

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

8. Explain what is RDD & why it is needed ?

RDD - Resilient Distributed Rates:

- It is basis building block of Spark,

- The RDD (Resilient Distributed Dataset) is the Spark's core abstraction.

- It is a collection of elements, partitioned across the nodes of the cluster so that we can execute various parallel operations on it.

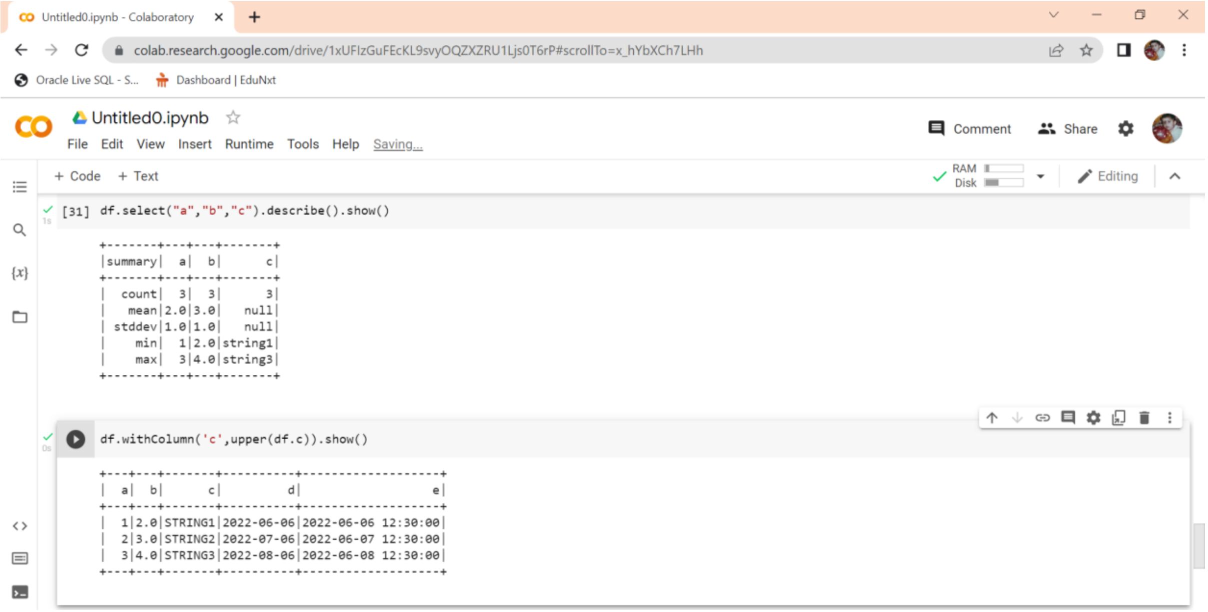
There are two ways to create RDDs:

- Parallelizing an existing data in the driver program

- Referencing a dataset in an external storage system, such as a shared filesystem, HDFS, HBase, or any data source offering a Hadoop InputFormat.

NEED: RDD (Resilient Distributed Dataset) is a basic data structure used in Spark to execute the MapReduce operations faster and efficiently. Data sharing in MapReduce take a lot of time because of replication, serialization, and disk IO. Hadoop applications take over 90 percent of the time in read-write operations.

9. Write a Py-Spark program to make the column in Upper case & explain with screenshots
& steps ?



```
[31] df.select("a","b","c").describe().show()

+-------+---+---+-------+
|summary|  a|  b|      c|
+-------+---+---+-------+
|  count|  3|  3|      3|
|   mean|2.0|3.0|   null|
| stddev|1.0|1.0|   null|
|    min|  1|2.0|string1|
|    max|  3|4.0|string3|
+-------+---+---+-------+
```

```
df.withColumn('c',upper(df.c)).show()

+---+---+-------+----------+-------------------+
|  a|  b|      c|         d|                  e|
+---+---+-------+----------+-------------------+
|  1|2.0|STRING1|2022-06-06|2022-06-06 12:30:00|
|  2|3.0|STRING2|2022-07-06|2022-06-07 12:30:00|
|  3|4.0|STRING3|2022-08-06|2022-06-08 12:30:00|
+---+---+-------+----------+-------------------+
```