ASSIGNMENT 6

NAME – JYOTHI CHANDANA VOLETI ROLL NUMBER – DXC-262-AB-1218
BATCH – DXC-262-ANALYTICS-B12-AZURE COMPANY – DXC TECHNOLOGY
EMPLOYEE DOMAIN –AZURE ANALYTICS
TRAINING UNDER – MANIPAL PRO LEARN TRAINER NAME – MR. AJAY KUMAR
DATE OF SUBMISSION – 6TH JUNE 2022 NO.OF QUESTIONS: 9

1. **Explain what is In-Memory computation in detail?**

Ans: <u>In-Memory Computation:</u>

- It is the technique of running computer calculations entirely in computer memory i.e, like in **RAM.**
- It helps computation at faster speed.
- It processes the data together in clusters using some specialized system software.

2. **Explain advantages of Spark Framework?**

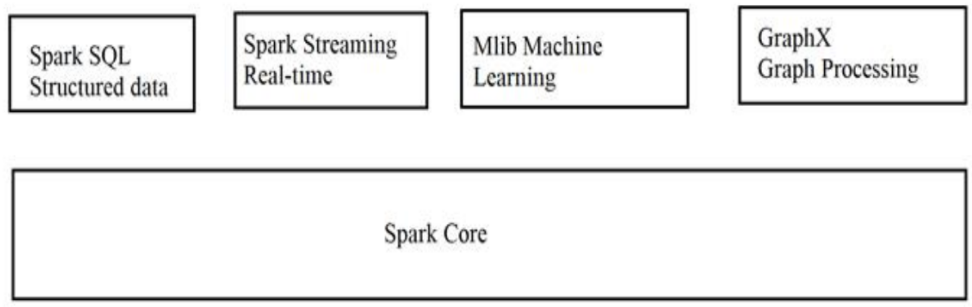
Ans: <u>Advantages of Spark Framework:</u>

- **Dynamic in Nature:** With Spark, you can easily develop parallel applications.
- **Advanced Analytics:** Spark not only supports 'MAP' and 'reduce'. It also supports ML, Graph Algorithms, Streaming data, SQL,etc.
- **Multilingual:** Spark supports many languages for code such as Java, Python,etc.
- **Open-Source:** It has a massive Open-Source community.
- **Demand for spark developers**
- **Increased Access to Big Data**
- **Ease of Use**
- **Speed**

3. **Explain Components of Spark with block diagrams?**

Ans: Block Diagram:

Components of Apache Spark:



Spark Sql : It is used to perform structured data analysis, especially if the data is too voluminous.

Spark Streaming: It mainly enables you to create analytical and interactive applications for live streaming data.

MLLIB: It has the provision to support many machine learning algorithms.

GraphX: For graphs and graphical computations, Spark has its own graph computation engine, called GraphX.

4. **Explain benefits of In- Memory Computation?**

Ans: <u>Benefits of In- Memory Computation:</u>

1. Grow Revenue
2. Reduce risk
3. Ability to reduce cost
4. Better, faster, decision making
5. Identify competitive opportunities
6. More efficient application
7. Best suited for real-time analytics

5. Explain the major difference between Hadoop and Spark?

Ans:

DIFFERENCE BETWEEN HADOOP AND SPARK:

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 more types of computations.
2	Hadoop's MapReduce model reads and writes from a disk, thus slowing down the processing speed.	Spark reduces the number of read/write cycles to disk and stores 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 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 the features of Spark?

Ans: <u>Features of Spark:</u>

- **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.
- **Easy to Use:** It supports various languages like Java, Python, Scala, Sql, R. It facilitates writing applications in Java, Scala, Python, R, and SQL. It also provides more than 80 high-level operators.
- **Supports Various Libraries:** It provides a collection of libraries including SQL and DataFrames, MLlib for machine learning, GraphX, and Spark Streaming.
- **Supports Real Time Streaming**
- **Lightweight:** It is a light unified analytics engine which is used for large scale data processing. Runs Everywhere - It can easily run on Hadoop, Apache Mesos, Kubernetes, standalone, or in the cloud.

7. Write a pyspark program to create a dataframe from RDD and explain with screenshots and steps?

Ans: Step 1: Install pyspark using the command- "pip install pyspark".

```
[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 28 kB/s
Collecting py4j==0.10.9.3
  Downloading py4j-0.10.9.3-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 22.8 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=bbe37cfc3196a934519e28fa7df66de31fedb2d4e9e017a
  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 2: Create a session using the commands in the screenshot.

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

Step 3:Here, now we will create a variable "rdd" and then create a dataframe for that as shown in the screenshot.

```
#create pyspark dataframe from RDD consisting of a list of tuples
rdd=spark.sparkContext.parallelize([
      (1, 2., 'str1', date(2022,6,6),datetime(2022,6,6,12,21)),
      (2, 3., 'str2', date(2022,7,6),datetime(2022,7,6,12,21)),
      (3, 5., 'str3', date(2022,8,6),datetime(2022,8,6,12,21))
])
df=spark.createDataFrame(rdd,schema=['a','b','c','d','e'])
df
```

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

Step 4: Now Let us display it. Using the command "df.show()".

```
df.show()
```

```
+---+---+----+----------+-------------------+
|  a|  b|   c|         d|                  e|
+---+---+----+----------+-------------------+
|  1|2.0|str1|2022-06-06|2022-06-06 12:21:00|
|  2|3.0|str2|2022-07-06|2022-07-06 12:21:00|
|  3|5.0|str3|2022-08-06|2022-08-06 12:21:00|
+---+---+----+----------+-------------------+
```

8. **Explain what RDD is and why it is needed?**

Ans: RDD:

- RDD (Resilient Distributed Dataset) is a basic data structure used in Spark to execute the MapReduce operations faster and efficiently.
- There are two ways to create RDD:

1. Parallelizing existing data in the driver program.
2. Referencing a dataset in an external storage system.

9. **Write a pyspark program to make the column in Upper Case and explain with screenshots and steps?**

Ans: Step 1: Create a dataframe to work on

```
#create pyspark dataframe from RDD consisting of a list of tuples
rdd=spark.sparkContext.parallelize([
      (1, 2., 'str1', date(2022,6,6),datetime(2022,6,6,12,21)),
      (2, 3., 'str2', date(2022,7,6),datetime(2022,7,6,12,21)),
      (3, 5., 'str3', date(2022,8,6),datetime(2022,8,6,12,21))
])
df=spark.createDataFrame(rdd,schema=['a','b','c','d','e'])
df
```

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

Step 2: Import "Upper"

```
from pyspark.sql import Column
from pyspark.sql.functions import upper
type(df.Name)==type(upper(df.Name))==type(df.Name.isNull())
```

```
True
```

Step 3: Let us see the data using df.show() command

```
df.show()
```

```
+---+---+----+----------+-------------------+
|  a|  b|   c|         d|                  e|
+---+---+----+----------+-------------------+
|  1|2.0|str1|2022-06-06|2022-06-06 12:21:00|
|  2|3.0|str2|2022-07-06|2022-07-06 12:21:00|
|  3|5.0|str3|2022-08-06|2022-08-06 12:21:00|
+---+---+----+----------+-------------------+
```

Step 4: Now use the Command to change the "c" Column to upper using the command in the screenshot.

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

```
+---+---+----+----------+-------------------+-------+
|  a|  b|   c|         d|                  e|upper_c|
+---+---+----+----------+-------------------+-------+
|  1|2.0|str1|2022-06-06|2022-06-06 12:21:00|   STR1|
|  2|3.0|str2|2022-07-06|2022-07-06 12:21:00|   STR2|
|  3|5.0|str3|2022-08-06|2022-08-06 12:21:00|   STR3|
+---+---+----+----------+-------------------+-------+
```