

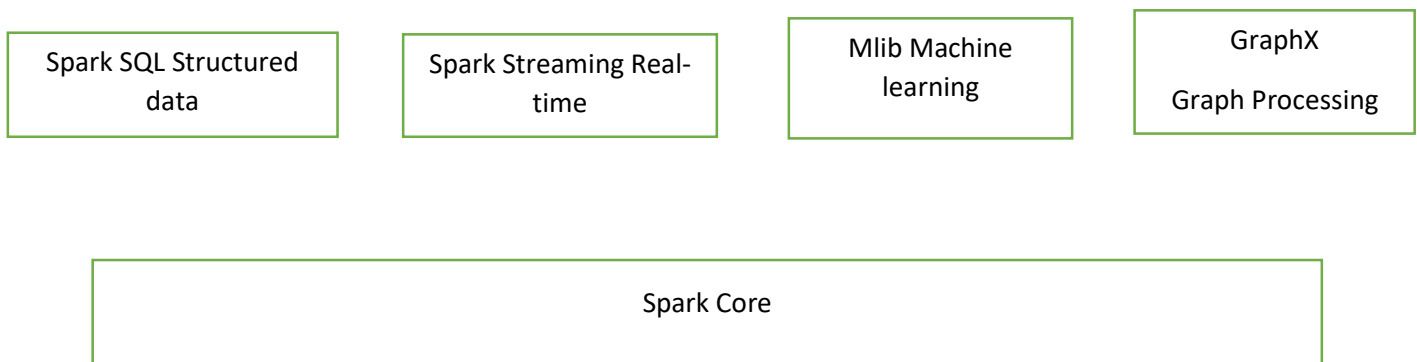
## ASSIGNMENT 6

NAME: Aishee Bhattacharya

BATCH: DXC-262-Analytics-B12-Azure

DATE: 06/06/2022

1. Explain what is in-Memory computation in details?
  - Processing in memory is one approach to overcoming the von Neumann bottle neck which is a limitation throughout caused by latency inherent in the standard computer architecture.
  - In-memory computing primary relies on keeping data in a sever's RAM as means of processing at faster speeds.
  - In-memory computation especially applies to problems that require extensive access to data analytics, reporting or data warehousing and big data applications.
2. Explain advantages of Spark framework ?
  - Apache Spark is an open-source cluster computing(In-memory) framework.
  - Spark was built on the top of Hadoop MapReduce, Spark process the data much quicker than Hadoop framework.
  - Hadoop 1 x times faster- Spark will be 100xtimes faster.
3. Explain components of Spark with block diagram ?



4. Explain benefits 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:
  1. Avoid movement of detailed data
  2. Calculate first ,then move the results.

5. Explain major difference between Hadoop & Spark ?

Spark is faster compared to Hadoop because it uses RAM instead of reading and writing intermediate data to disks. Whereas, Hadoop stores data on multiple sources and processes it in batches via MapReduce.

6. Explain features of Spark?

- 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 to write the application in Java ,Scala,Python,R, and SQL.It provides more than 80 high-level operations.
- Supports various libraries: It provides a collection of libraries including SQL and DataFrames, MLib for machine learning, GraphX and Spark streaming.
- Supports Realtime 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 Py-Spark program to create Dataframe from RDD & explain with screenshots & steps ?

```
[7] # Creating pyspark dataframe from Pandas DF
pandas_df=pd.DataFrame({
    'a':[1,2,3],
    'b':[2,3,4],
    'c':['string1','string2','string3'],
    'd':[date(2022,6,6),date(2022,7,6),date(2022,7,8)],
    'e':[datetime(2022,6,6,12,30),datetime(2022,6,7,12,30),datetime(2022,6,8,12,30)]
})
df=spark.createDataFrame(pandas_df)

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

[11] ## Create PySpark dataframe from RDD consisting of a list of tuples
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]

[12] df.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 |
+-----+-----+-----+-----+
```

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

- It is basic building block of Spark
- The RDD(Resilient Distributed Dataset) is the Spark's core abstraction.
- It is a collection of elements, partitioned across nodes of the cluster so that we can execute various parallel operations on it.
- There are two ways to create RDDs:
  1. Parallelizing an existing data in the driver program
  2. Referencing a dataset in an external storage system, such as filesystem, HDFS, HBase or any data source offering a Hadoop Input Format.

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

The screenshot shows a Google Colab notebook titled 'Untitled0.ipynb'. The notebook contains three code cells. The first cell imports the 'Column' class from 'pyspark.sql' and the 'upper' function from 'pyspark.sql.functions'. It also checks the type of the 'c' column in a DataFrame 'df' and prints the result, which is 'True'. The second cell calls 'df.select(df.c).show()' to display the contents of column 'c'. The output shows three rows of string data. The third cell calls 'df.withColumn("upper\_c", upper(df.c)).show()' to create a new column 'upper\_c' with the uppercase values of column 'c'. The output shows the same three rows with the new column added. The notebook interface includes a menu bar (File, Edit, View, Insert, Runtime, Tools, Help), a toolbar with icons for code, text, and search, and a status bar at the bottom showing the temperature (35°C) and the time (17:45 on 06-06-2022).

```
[29] 
```

```
[32] from pyspark.sql import Column
      from pyspark.sql.functions import upper
      type(df.c)==type(upper(df.c))== type (df.c.isNull())

      True

df.select(df.c).show()

+-----+
|      c|
+-----+
|string1|
|string2|
|string3|
+-----+

df.withColumn("upper_c", upper(df.c)).show()

+-----+-----+-----+-----+-----+
| a| b| c| d| e| upper_c|
+-----+-----+-----+-----+-----+
| 1| 2| 0| string1| 2022-06-06 12:30:00| STRING1|
| 2| 3| 0| string2| 2022-06-07 12:30:00| STRING2|
| 3| 4| 0| string3| 2022-06-08 12:30:00| STRING3|
+-----+-----+-----+-----+-----+
```

0s completed at 4:59 PM

35°C Mostly sunny

17:45 06-06-2022