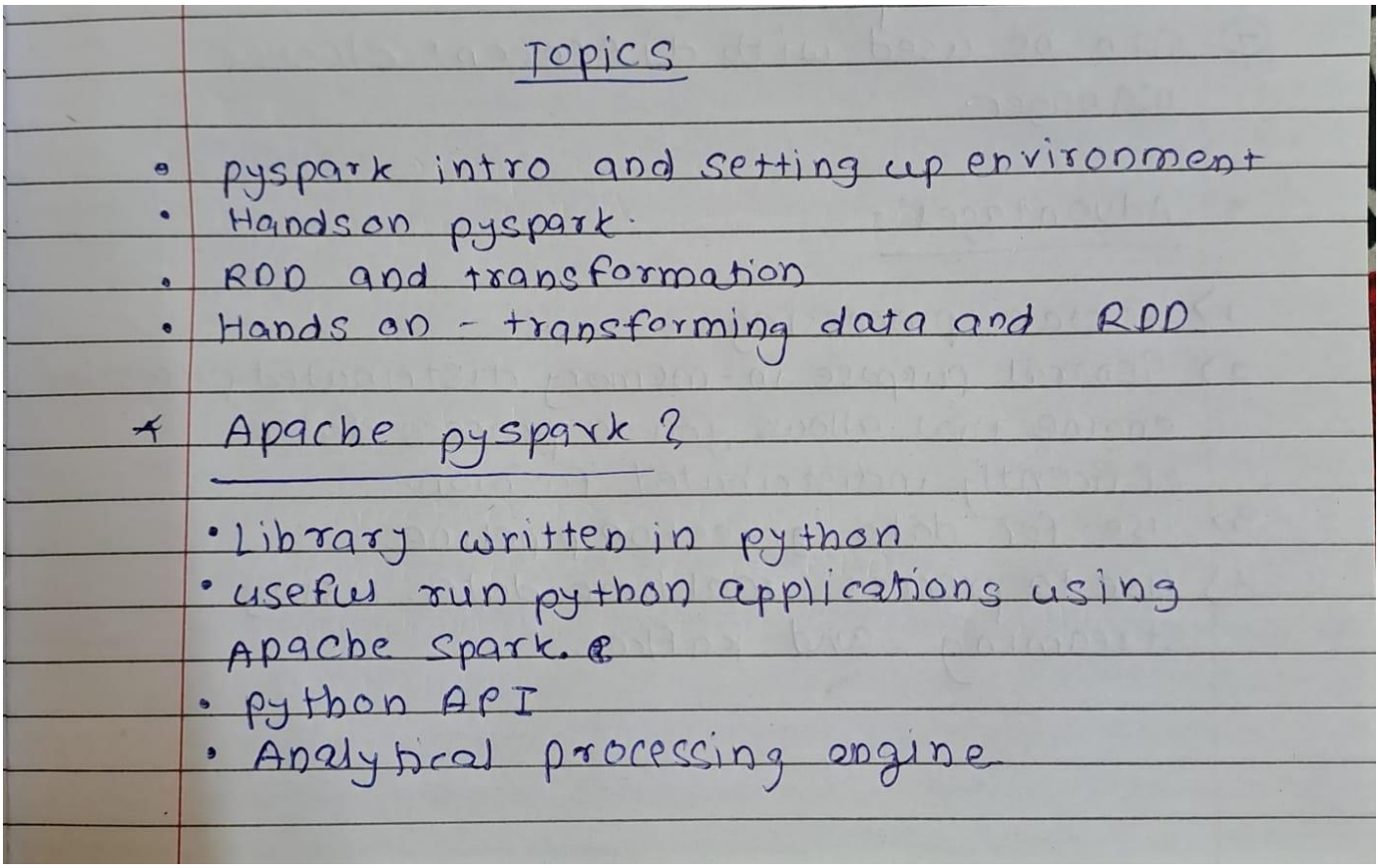
PySpark Assignment 2

Pratik Wani

- Hand Written Notes:



PAGE NO.:

* Apache Spark?

- Unified Analytical engine used for large dataset
- faster
- can run on single node or multiple nodes
- In memory processing.
- Numpy + Tenserflow

* Features:-

1. In memory computation
2. Distributed processing
3. immutable
4. Lazy evalution
5. Supports ANSI SQL
6. Cache and persistance
7. can be used with different cluster manager

* Advantages:-

1) Faster processing
2) General purpose in-memory distributed processing engine that allows you to process data efficently indistributed fashion.
3) use for data ingestion pipeline
4) use to process real time data using Streaming and Kafka.

PAGE NO.:

* Apache kafka :-
  - Event streaming platform
  - Distributed event streaming platform.

* versions

Py → 3.8 and above

Java → Java 8, 11, 13, 17 and above.

Scala → 2.12 and 2.13

R → 3.5

* python modules and package :-

- Pyspark RDD
- pyspark dataframe
- Pyspark Streaming
- pyspark MyLib
- pyspark Graphframes
- Pyspark Resource (NEW in 3.0)

* commands :

Read data from file
→ create txt file
→ val filepath = "...path..."
→ val textRDD = sc.textFile(filepath)
→ textRDD.collect()

PAGE NO.:

* RDD creation :-

2 ways:

1st

using paralize() :-
- create session
- make dataset
- then passing it in paralize()

2nd

using txt file.

textfile ("textfilename")

- Codes:
- Jupyter Notebook:

    - Read CSV

```
In [1]: import pyspark

In [2]: from pyspark.sql import SparkSession

In [3]: spark = SparkSession.builder.appName("practice").getOrCreate()

In [4]: spark

Out[4]: SparkSession - in-memory
        SparkContext

        Spark UI
        Version
        v3.5.0
        Master
        local[*]
        AppName
        practice
```

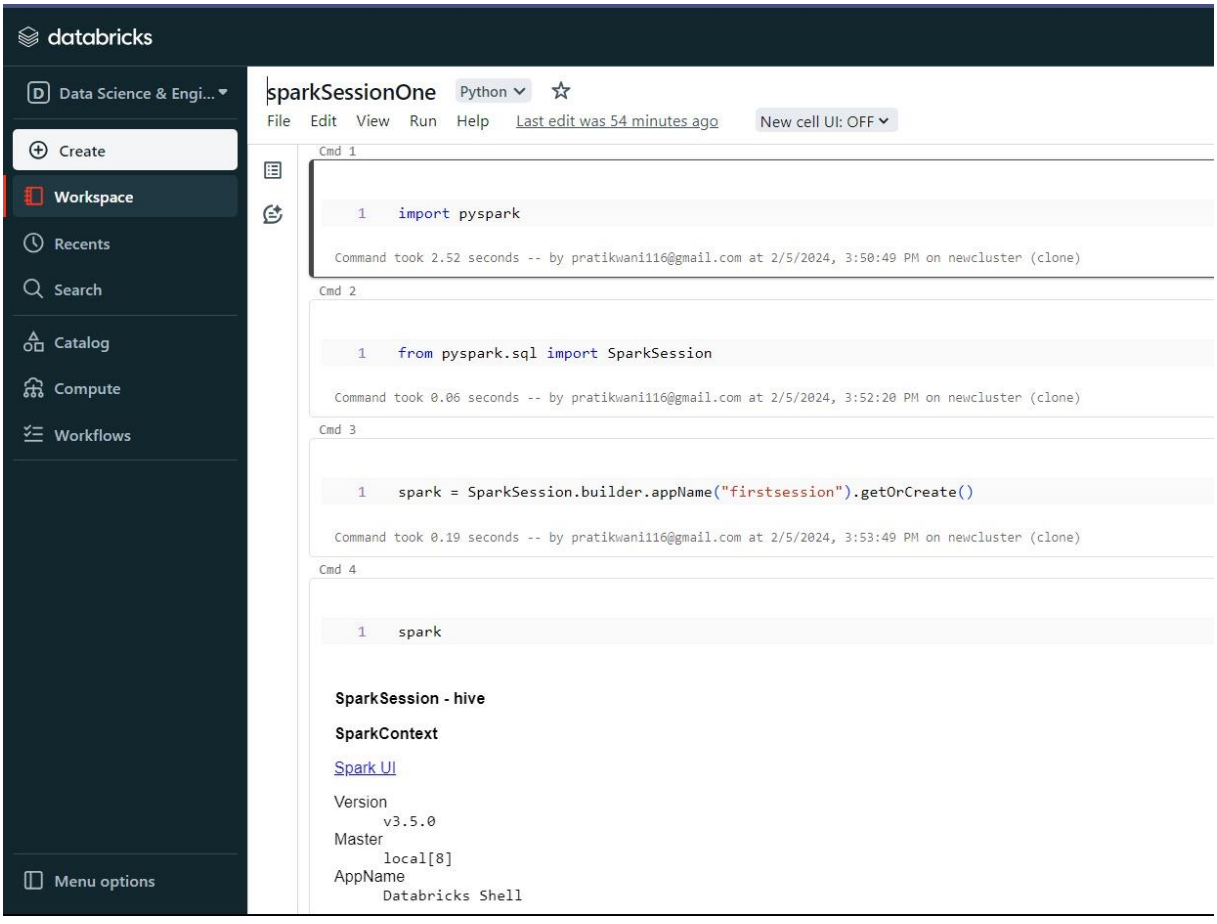
```
In [10]: df = spark.read.csv('student_salary_info.csv')
         df

Out[10]: DataFrame[_c0: string, _c1: string, _c2: string]

In [11]: df.show()

         +------+---+------+
         |   _c0|_c1|   _c2|
         +------+---+------+
         |  Name|Age|Salary|
         |Pratik| 21| 50000|
         | Vikas| 23|100000|
         | Rushi| 22|120000|
         +------+---+------+
```

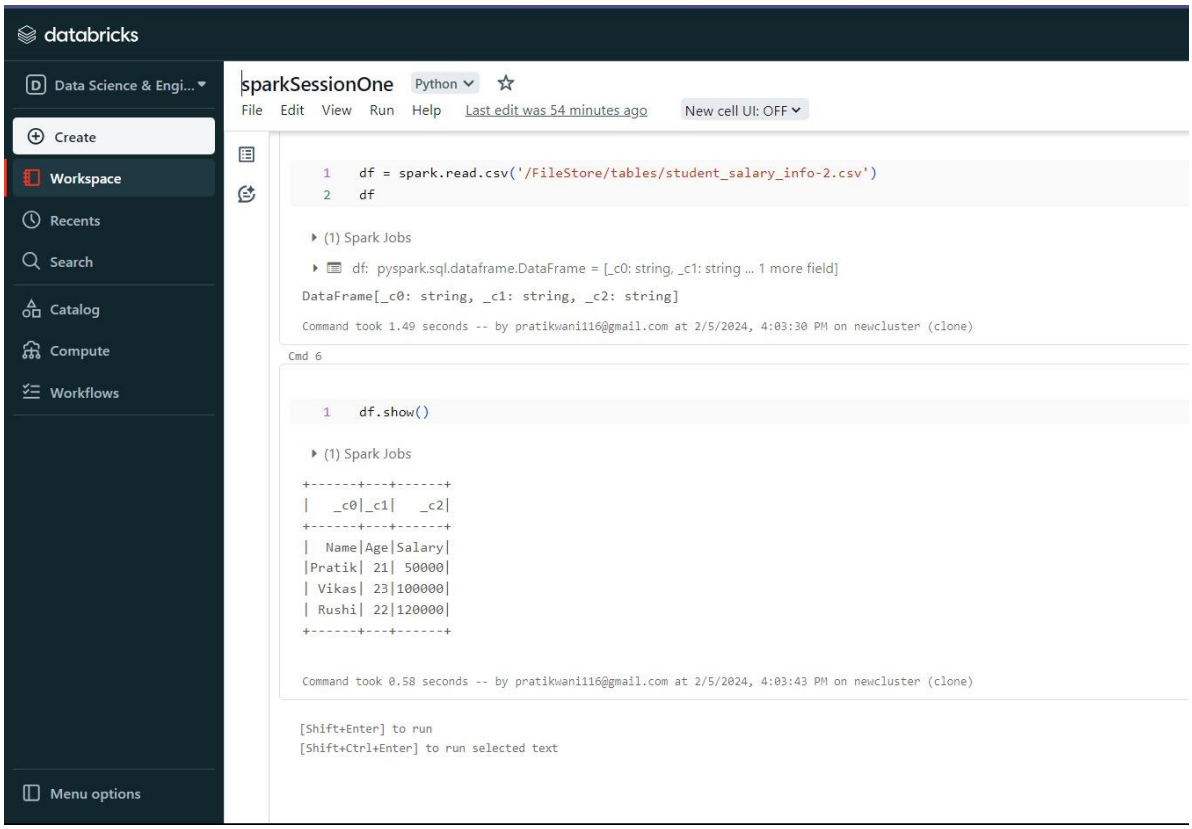
- Using Paralize()

```
In [1]: from pyspark.sql import SparkSession

In [2]: spark = SparkSession.builder \
            .master("local[1]") \
            .appName("SparkByExamples.com") \
            .getOrCreate()

In [3]: dataList = [("Shiva", 20000), ("Pratik", 100000), ("vikas", 3000)]

In [4]: rdd=spark.sparkContext.parallelize(dataList)

In [5]: rdd2 = spark.sparkContext.textFile("/path/test.txt")

In [7]: result = rdd.collect()

In [8]: result
Out[8]: [('Shiva', 20000), ('Pratik', 100000), ('vikas', 3000)]

In [9]: result
Out[9]: ('RDD Contents:', [('Shiva', 20000), ('Pratik', 100000), ('vikas', 3000)])

In [10]: rdd2=spark.sparkContext.textFile("myfile.txt")

In [11]: rdd2.collect()
Out[11]: ['Hello!!', 'My name is pratik arun wani', 'Pyspark', 'Databricks']
```

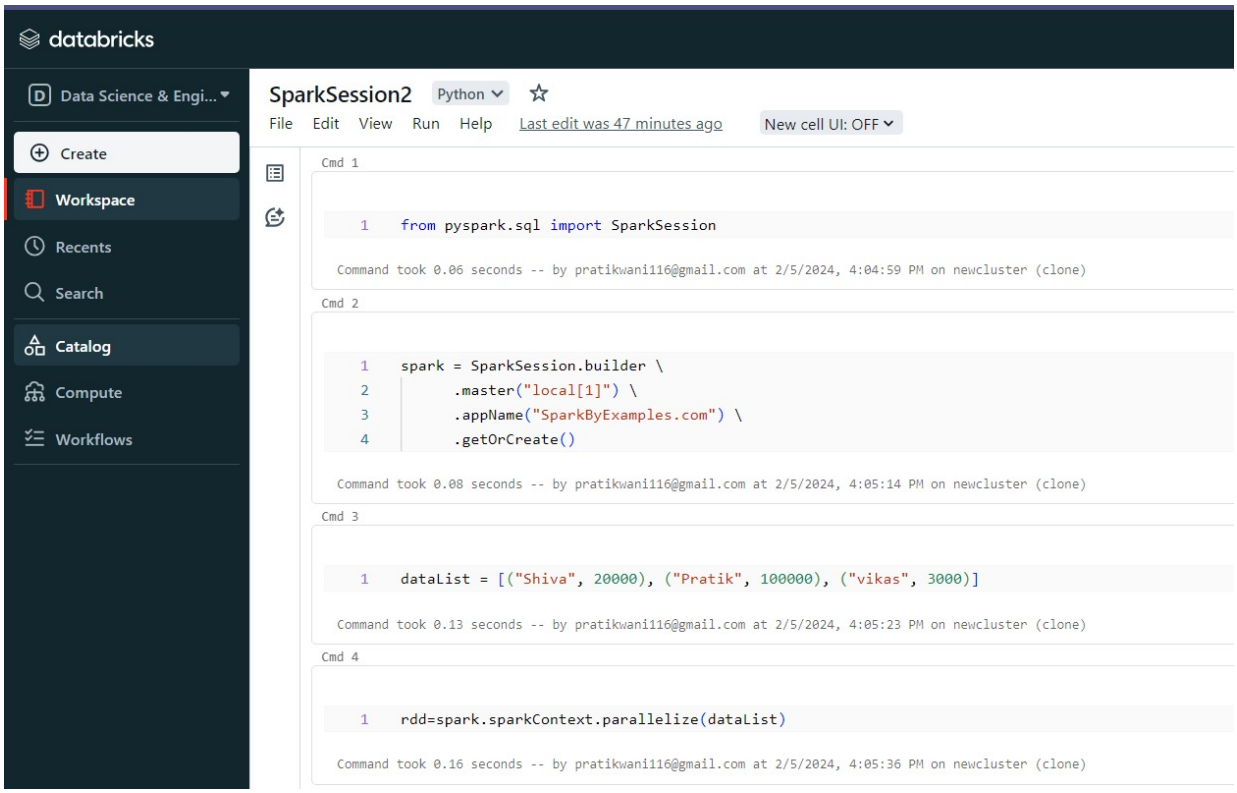
- Databricks:

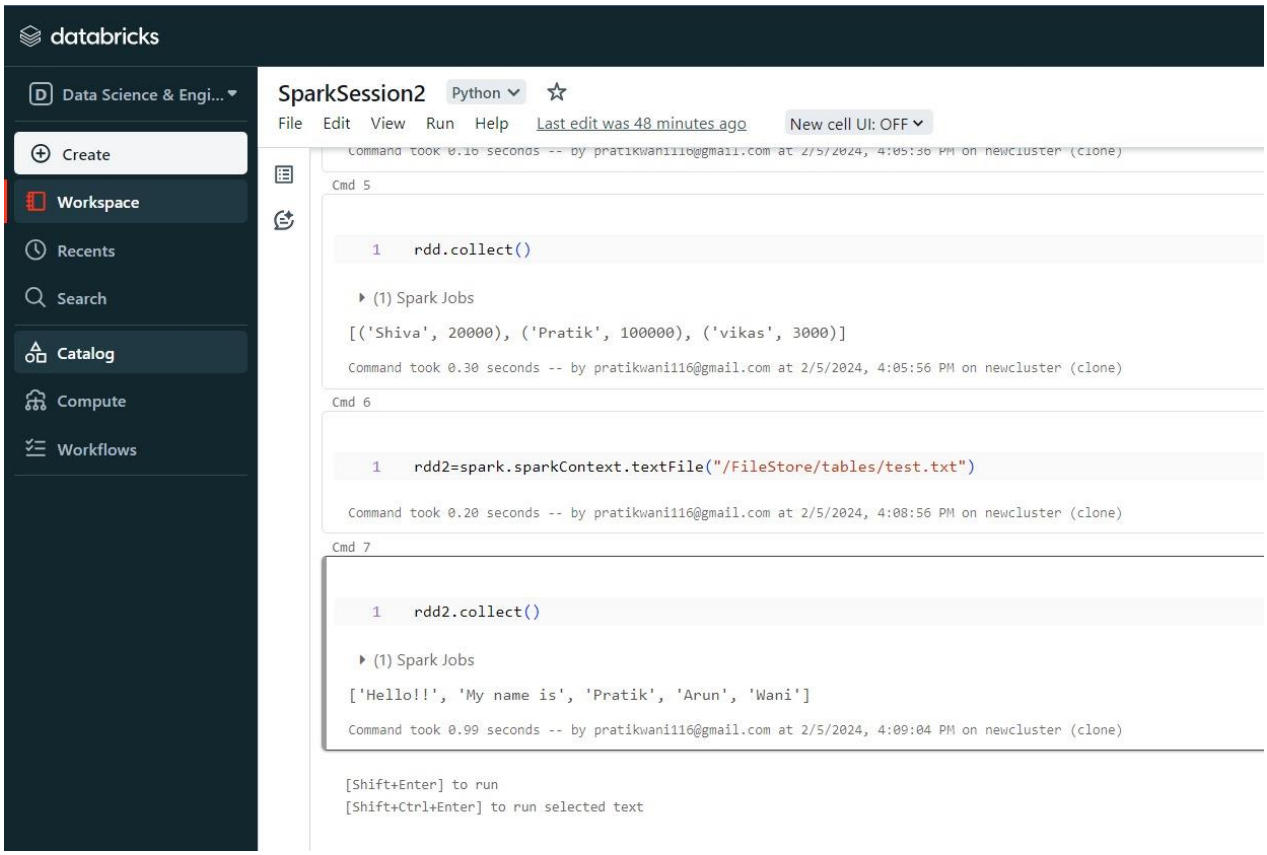
  - Read CSV





- Using Paralize()





- Local Host 4040/jobs