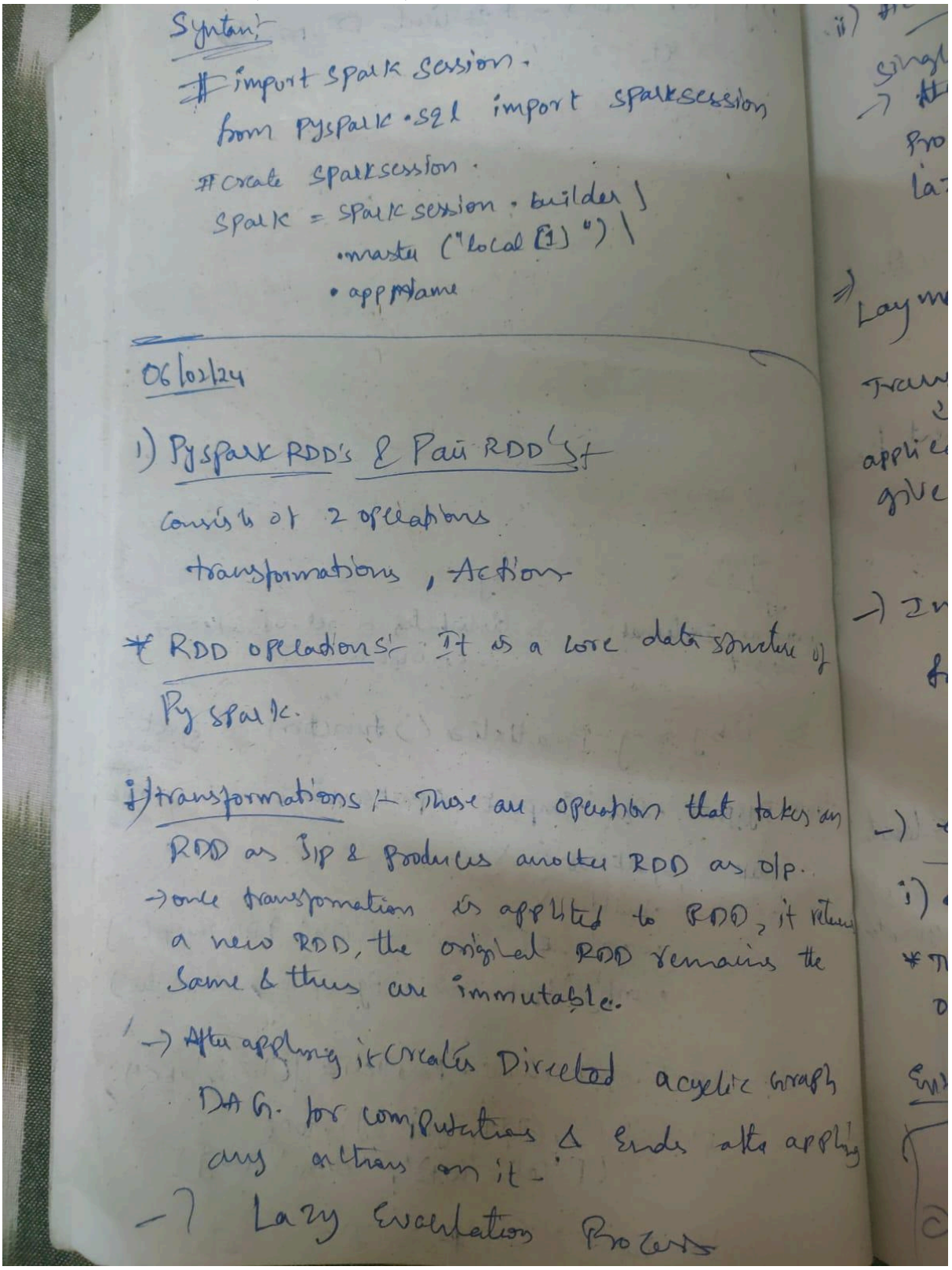
Name: Akula Sharathchandra

Batch: Data Engineering

Date:06/02/2024

TOPICS: PYSPARK RDD, hands on RDD,CREATING,SELECTING,RENAMING THE DATAFRAME.

PYSPARK RDD(THEORY):



Syntax:

```
# import spark session.
from pyspark.sql import sparksession
# create sparksession.
spark = sparksession.builder \
        .master("local[1]") \
        .appname
```

06/02/24

1) Pyspark RDD's & Pair RDD's:

Consists of 2 operations

transformations, Actions

* RDD operations:- It is a core data structure of Pyspark.

i) transformations :- These are operation that takes an RDD as I/p & produces another RDD as O/p.

→ once transformation is applied to RDD, it returns a new RDD, the original RDD remains the same & thus are immutable.

→ After applying it creates Directed acyclic graph DAG. for computations & Ends after applying any actions on it.

→ Lazy Evaluation Process

ii) Actions applied on RDD to produce a single value.
→ these are applied on resultant RDD & produces a non-RDD value, thus removing laziness of transformation of RDD.

⇒ Layman's terms,

Transformations	Action
↓	↓
applied on RDD to give another RDD	performed on RDD to give a non-RDD value.

→ Initialize a spark context

```
from pyspark import SparkContext
sc = SparkContext.getOrCreate();
```

→ Essential PySpark RDD operations

i) .collect() Actions

* The .collect() action on RDD returns a list of all elements of RDD's

Eg:

```
collect_rdd = sc.parallelize([1,2,3,4,5])
Print(collect_rdd.collect())
```

O/P: [1,2,3,4,5]

ii) .count() :- returns count of Elements from RDD.

```
count_rdd = sc.parallize ([1,2,3,4,5])
Print (count_rdd.count())
O/P = 10
```

=) count_rdd uses the .parallelize() method & prints the count of array.

iii) .first() :- returns the first Element from RDD.

```
Ent
first_rdd = sc.parallize ([1,2,3,4,])
Print (first_rdd.first())
O/P = 1
```

iv) .take() :- returns n numbers of Elements from RDD. returns from starting to n no of arguments

```
Ent
take_rdd = sc.parallize ([1,2,3,4,5])
Print (take_rdd.take (3))
O/P = [1,2,3]
```

v)

vi)

v) .reduce() :- 2 Elements from given RDD

2 operands.

-) Performed using anonymous function of lambda.

Ex:-

```
reduce_rdd = sc.parallelize([1,3,4,6])
Print(reduce_rdd.reduce(lambda x,y : x+y))
```

O/P:-

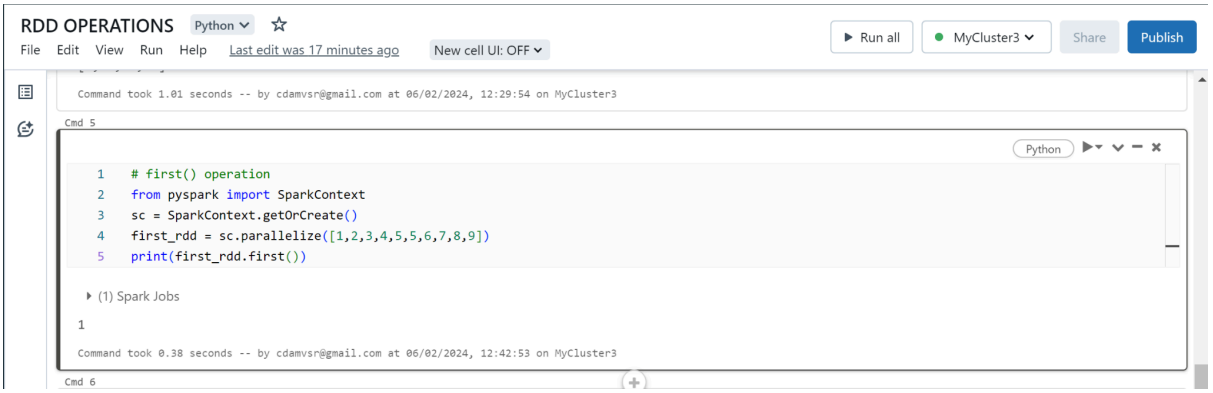
vi) .saveAsTextFile() :- used to store the resultant RDD as text file.

Ex:-

```
save_rdd = sc.parallelize([1,2,3,4,5,6])
save_rdd.saveAsTextFile('file.txt')
```

O/P:-

HANDSON OF RDD OPERATIONS IN DATABRICKS AS WELL AS IN JUPYTER:



IN JUPYTER NOTEBOOK:

In [1]:
```
!pip install findspark
```

```
Collecting findspark
  Downloading findspark-2.0.1-py2.py3-none-any.whl (4.4 kB)
Installing collected packages: findspark
Successfully installed findspark-2.0.1
```

In [2]:
```
import findspark
```

In [3]:
```
findspark.init()
```

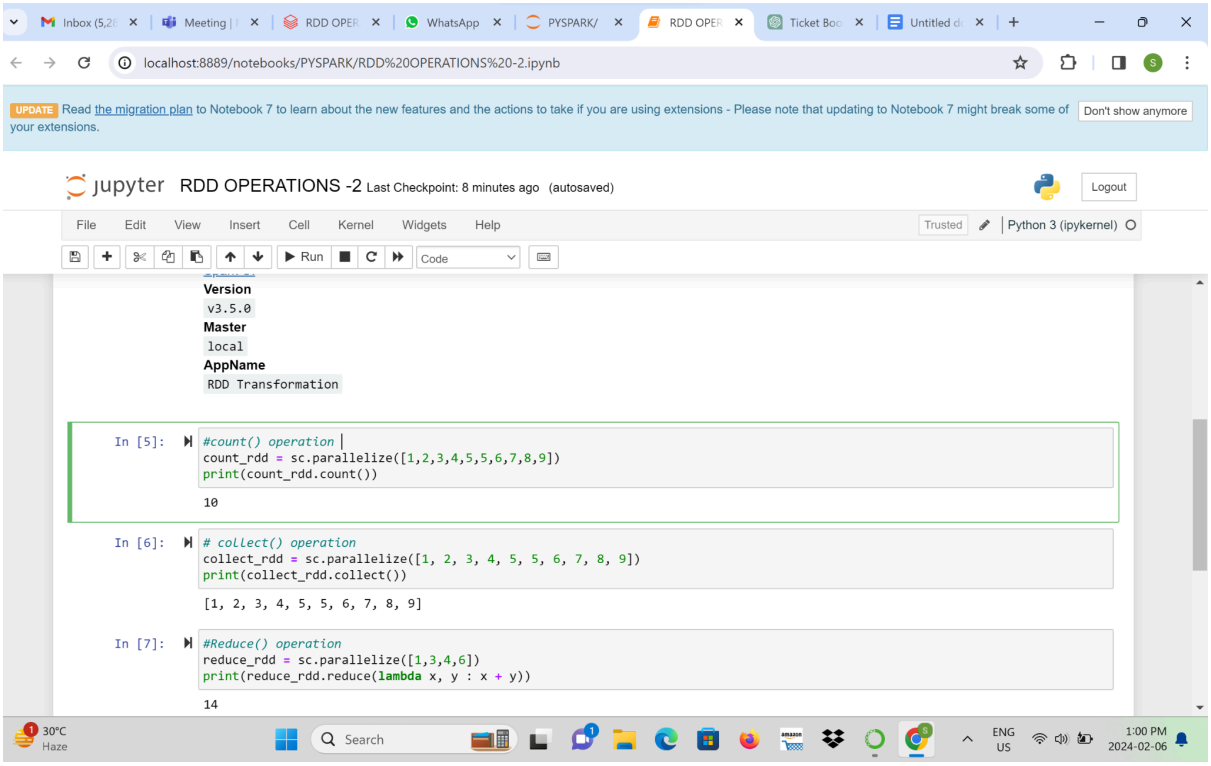
In [4]:
```
from pyspark import SparkContext
sc = SparkContext("local", "RDD Transformation")
sc
```

Out[4]:

SparkContext

Spark UI

Version
`v3.5.0`
Master



Version
`v3.5.0`
Master
`local`
AppName
`RDD Transformation`

In [5]:
```
#count() operation
count_rdd = sc.parallelize([1,2,3,4,5,5,6,7,8,9])
print(count_rdd.count())
```

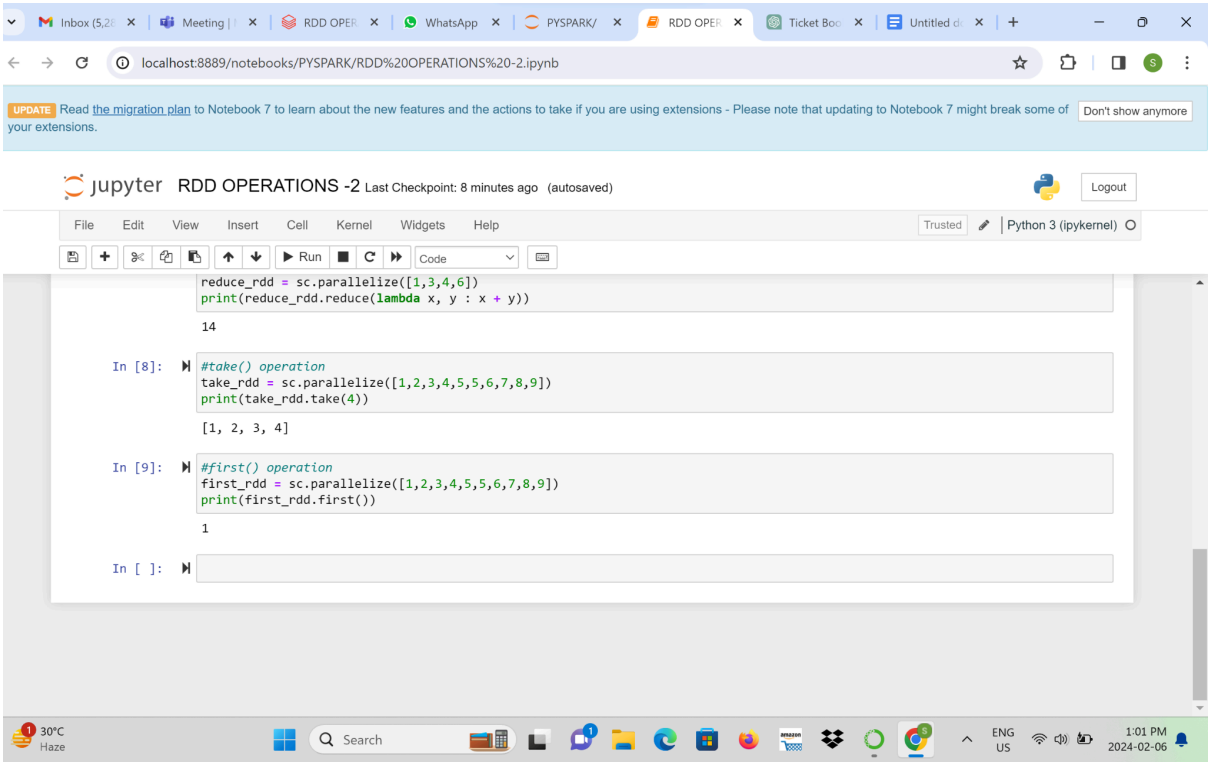
```
10
```

In [6]:
```
# collect() operation
collect_rdd = sc.parallelize([1, 2, 3, 4, 5, 5, 6, 7, 8, 9])
print(collect_rdd.collect())
```

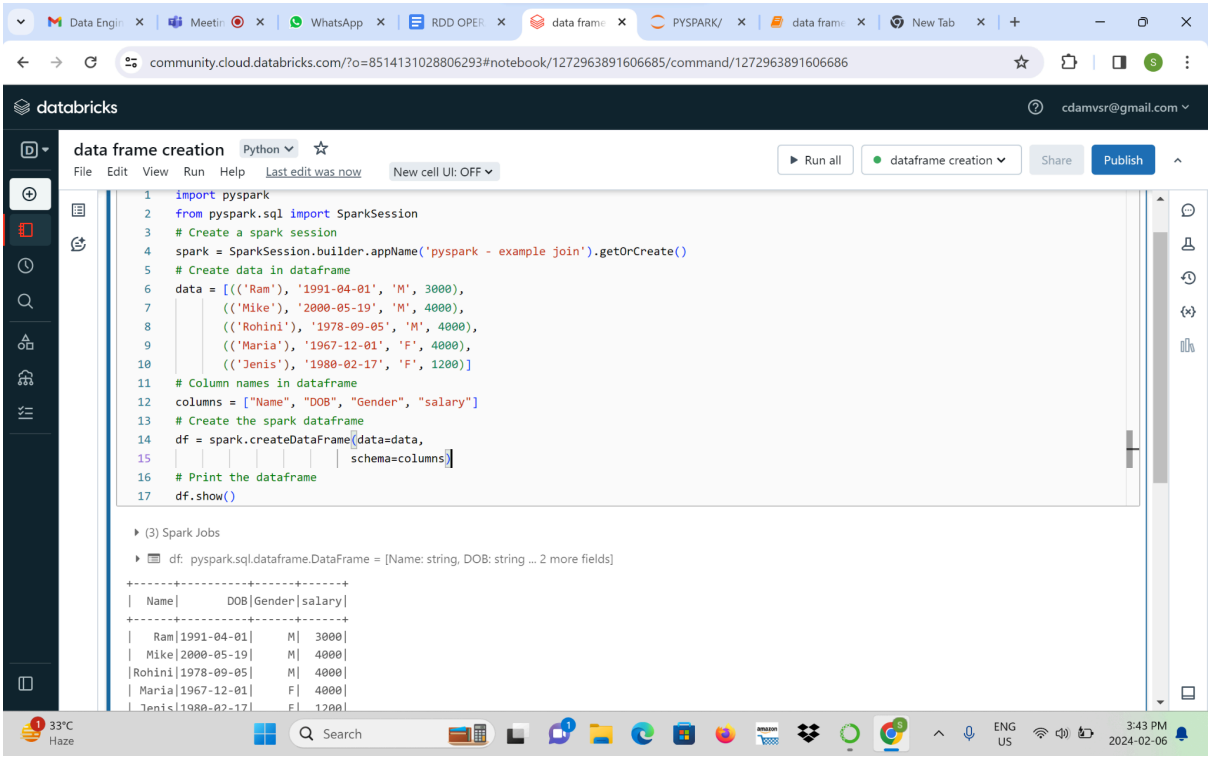
```
[1, 2, 3, 4, 5, 5, 6, 7, 8, 9]
```

In [7]:
```
#Reduce() operation
reduce_rdd = sc.parallelize([1,3,4,6])
print(reduce_rdd.reduce(lambda x, y : x + y))
```

```
14
```


Inbox (5,2
Meeting |
RDD OPER
WhatsApp
PYSPARK/
RDD OPER
Ticket Boo
Untitled d
localhost:8889/notebooks/PYSPARK/RDD%20OPERATIONS%20-2.ipynb
UPDATE Read the migration plan to Notebook 7 to learn about the new features and the actions to take if you are using extensions - Please note that updating to Notebook 7 might break some of your extensions.
Don't show anymore
jupyter RDD OPERATIONS -2 Last Checkpoint: 8 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Trusted
Python 3 (ipykernel)
Run
Code
reduce_rdd = sc.parallelize([1,3,4,6])
print(reduce_rdd.reduce(lambda x, y : x + y))
14
In [8]:
#take() operation
take_rdd = sc.parallelize([1,2,3,4,5,5,6,7,8,9])
print(take_rdd.take(4))
[1, 2, 3, 4]
In [9]:
#first() operation
first_rdd = sc.parallelize([1,2,3,4,5,5,6,7,8,9])
print(first_rdd.first())
1
In [ ]:
30°C
Haze
Search
ENG
US
1:01 PM
2024-02-06

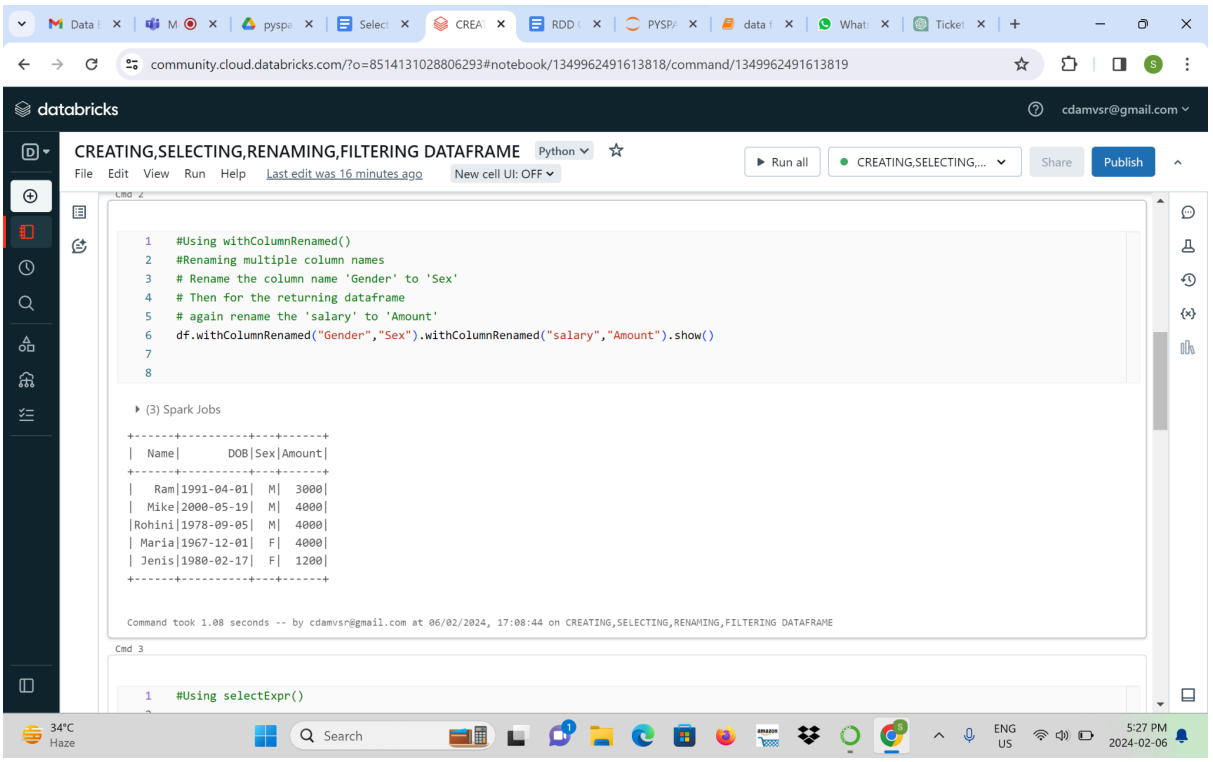
—>CREATING DATA FRAME:



Method 1: Using withColumnRenamed()

We will use of withColumnRenamed() method to change the column names of pyspark data frame.

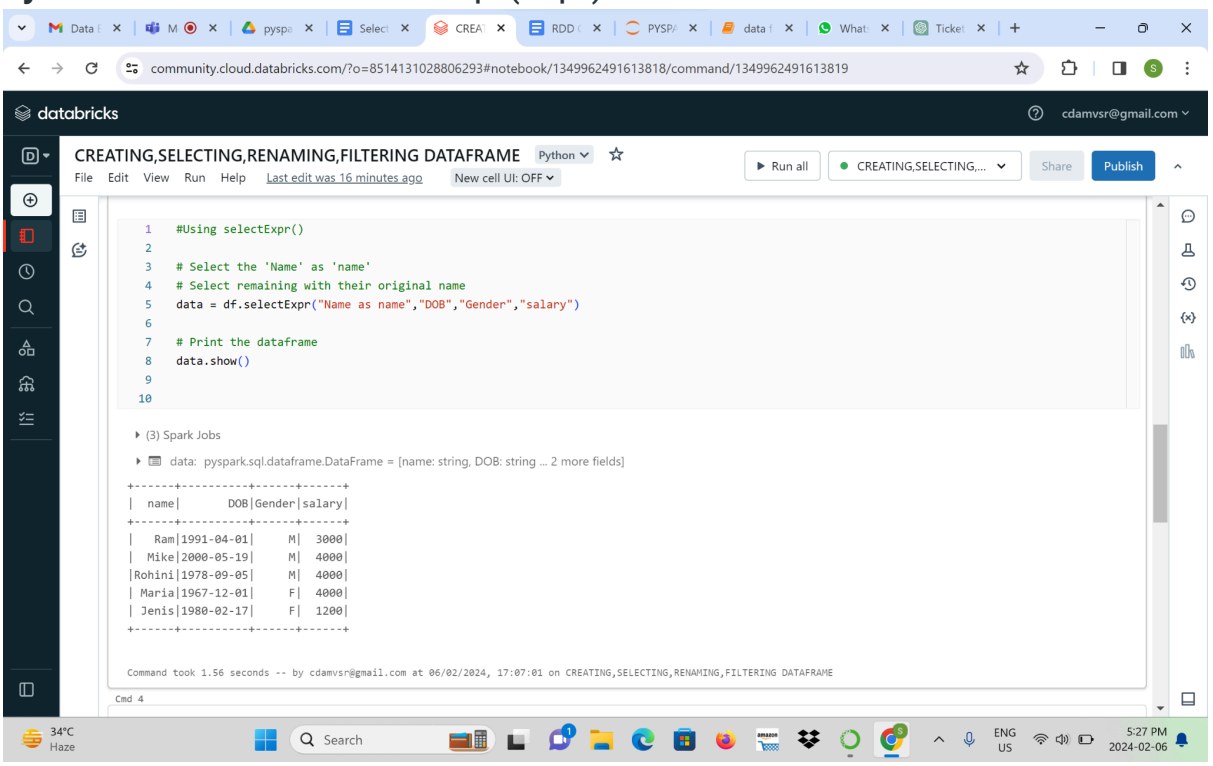
Syntax: DataFrame.withColumnRenamed(existing, new)



Method 2: Using selectExpr()

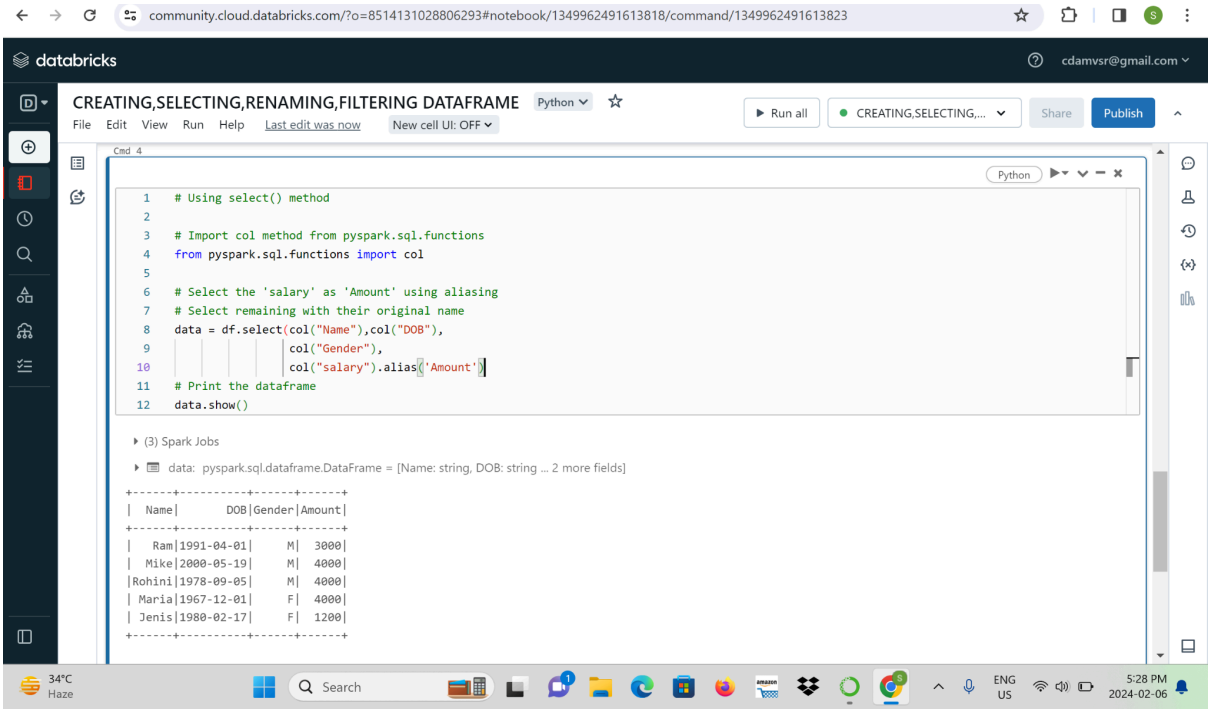
Renaming the column names using **selectExpr()** method

Syntax : DataFrame.selectExpr(expr)

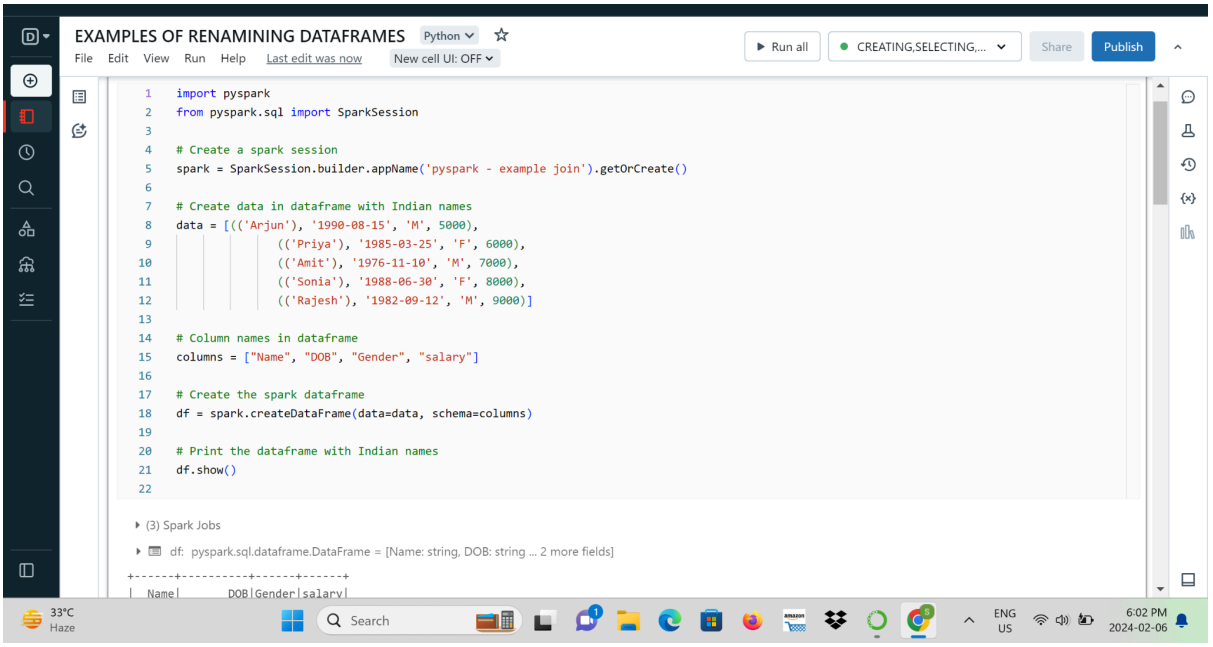


Method 3: Using select() method

Syntax: DataFrame.select(cols)



Example-2:



df: pyspark.sql.dataframe.DataFrame = [Name: string, DOB: string ... 2 more fields]

```
+------+----------+------+------+
|  Name|       DOB|Gender|salary|
+------+----------+------+------+
| Arjun|1990-08-15|     M|  5000|
| Priya|1985-03-25|     F|  6000|
|  Amit|1976-11-10|     M|  7000|
| Sonia|1988-06-30|     F|  8000|
|Rajesh|1982-09-12|     M|  9000|
+------+----------+------+------+
```

Command took 2.09 seconds -- by cdamvsr@gmail.com at 06/02/2024, 17:45:47 on CREATING,SELECTING,RENAMING,FILTERING DATAFRAME

```
#Using withColumnRenamed()
#Renaming multiple column name
df.withColumnRenamed("Gender","Sex").withColumnRenamed("salary","Amount").show()
```

(3) Spark Jobs

```
+------+----------+---+------+
|  Name|       DOB|Sex|Amount|
+------+----------+---+------+
| Arjun|1990-08-15|  M|  5000|
| Priya|1985-03-25|  F|  6000|
|  Amit|1976-11-10|  M|  7000|
| Sonia|1988-06-30|  F|  8000|
|Rajesh|1982-09-12|  M|  9000|
+------+----------+---+------+
```

Command took 0.96 seconds -- by cdamvsr@gmail.com at 06/02/2024, 17:51:17 on CREATING,SELECTING,RENAMING,FILTERING DATAFRAME

Cmd 3

```python
#Using selectExpr()
# Select the 'Name' as 'name'
# Select remaining with their original name
data = df.selectExpr("Name as name","DOB","Gender","salary")
# Print the dataframe
data.show()
```

▸ (3) Spark Jobs

▸ data: pyspark.sql.dataframe.DataFrame = [name: string, DOB: string ... 2 more fields]

```
+------+----------+------+------+
|  name|       DOB|Gender|salary|
+------+----------+------+------+
| Arjun|1990-08-15|     M|  5000|
| Priya|1985-03-25|     F|  6000|
|  Amit|1976-11-10|     M|  7000|
| Sonia|1988-06-30|     F|  8000|
|Rajesh|1982-09-12|     M|  9000|
+------+----------+------+------+
```

Command took 1.46 seconds -- by cdamvsr@gmail.com at 06/02/2024, 17:54:11 on CREATING,SELECTING,RENAMING,FILTERING DATAFRAME

Cmd 4

Python

```python
# Using select() method

# Import col method from pyspark.sql.functions
from pyspark.sql.functions import col

# Select the 'salary' as 'Amount' using aliasing
# Select remaining with their original name
data = df.select(col("Name"),col("DOB"),
                 col("Gender"),
                 col("salary").alias('Amount'))
# Print the dataframe
data.show()
```

▸ (3) Spark Jobs

▸ data: pyspark.sql.dataframe.DataFrame = [Name: string, DOB: string ... 2 more fields]

```
+------+----------+------+------+
|  Name|       DOB|Gender|Amount|
+------+----------+------+------+
| Arjun|1990-08-15|     M|  5000|
| Priya|1985-03-25|     F|  6000|
|  Amit|1976-11-10|     M|  7000|
| Sonia|1988-06-30|     F|  8000|
|Rajesh|1982-09-12|     M|  9000|
+------+----------+------+------+
```