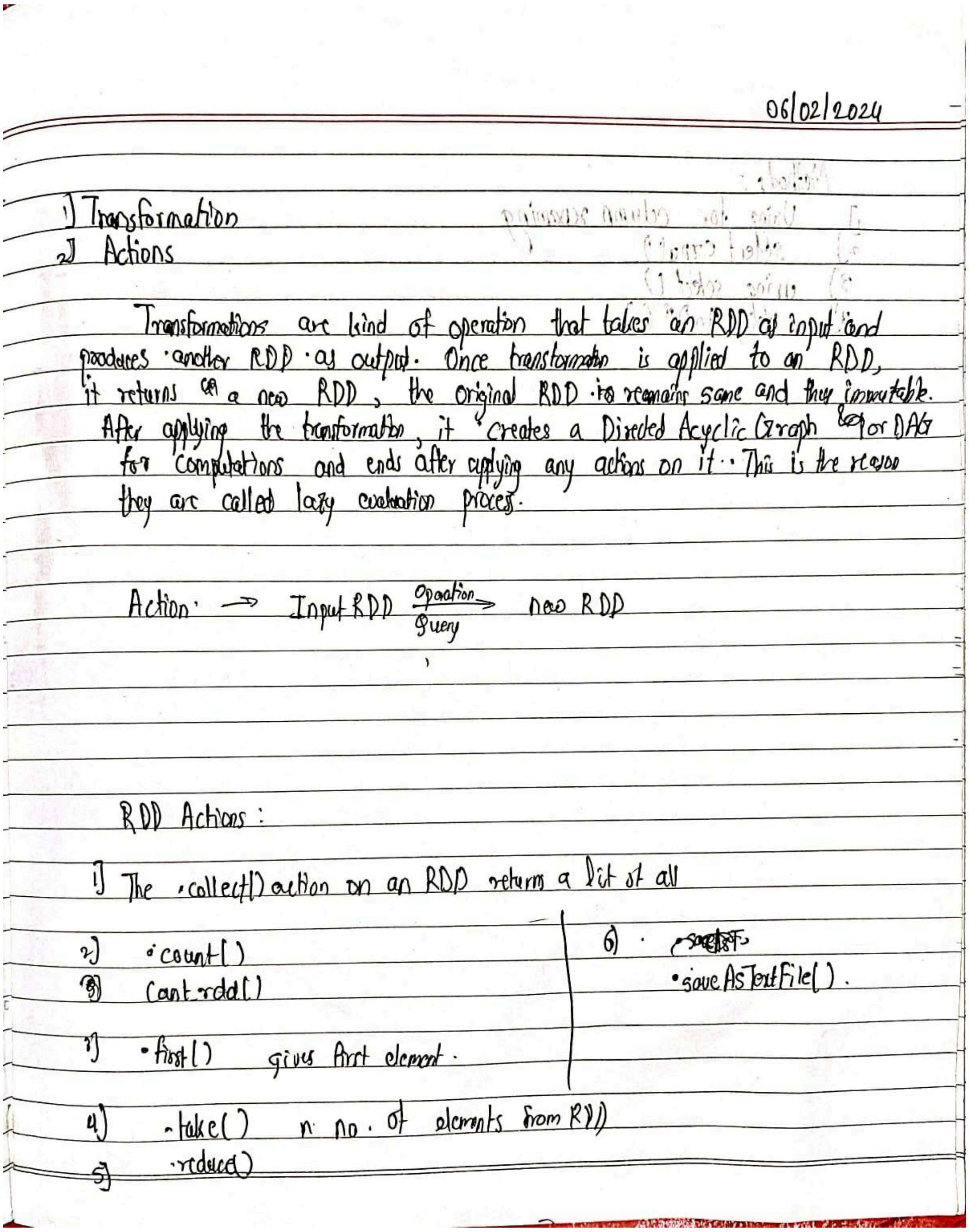
Name : Parth Nandedkar
Date : 06 Feb 2024
Topics : PySpark
Batch : Data Engineering Batch-1

**Handwritten Notes :**



06/02/2024

1) Transformation
2) Actions

Transformations are kind of operation that takes an RDD as input and produces another RDD as output. Once transformation is applied to an RDD, it returns ~~[illegible]~~ a new RDD, the original RDD ~~to~~ remains same and they immutable. After applying the transformation, it creates a Directed Acyclic Graph ~~&~~ or DAG for computations and ends after applying any actions on it. This is the reason they are called lazy evaluation process.

Action $\rightarrow$ Input RDD $\xrightarrow[\text{Query}]{\text{Operation}}$ new RDD

RDD Actions :

1) The .collect() action on an RDD returns a list of all

2) .count()
3) Count.rdd()
3) .first() gives first element.
4) .take() n no. of elements from RDD
5) .reduce()

6) ~~.saveAsT~~
.saveAsTextFile().

Methods :-

1] Using for column renaming
2] select expr()
3] using select ()
4] using toDF ()

**RDD Practice :**

Apache Spark, RDDs (Resilient Distributed Datasets) support two types of operations: transformations and actions. Let's discuss each:

Transformations:
Definition: Transformations are operations applied to an RDD that produce another RDD. They are lazy, meaning they don't compute their results immediately but instead create a lineage of transformations.

Immutability: RDDs are immutable, so transformations create a new RDD rather than modifying the existing one.

Lazy Evaluation: Transformations are lazily evaluated, meaning Spark doesn't compute the results until an action is called. This allows Spark to optimize the execution plan.

Examples of Transformations:

-map(func): Applies a function to each element of the RDD.
-filter(func): Filters the RDD based on a predicate function.
-flatMap(func): Similar to map, but each input item can be mapped to zero or more output items.
-reduceByKey(func): Combines values with the same key using a provided function.
-sortByKey(): Sorts RDD elements by key.
-groupByKey(): Groups the values for each key in the RDD.

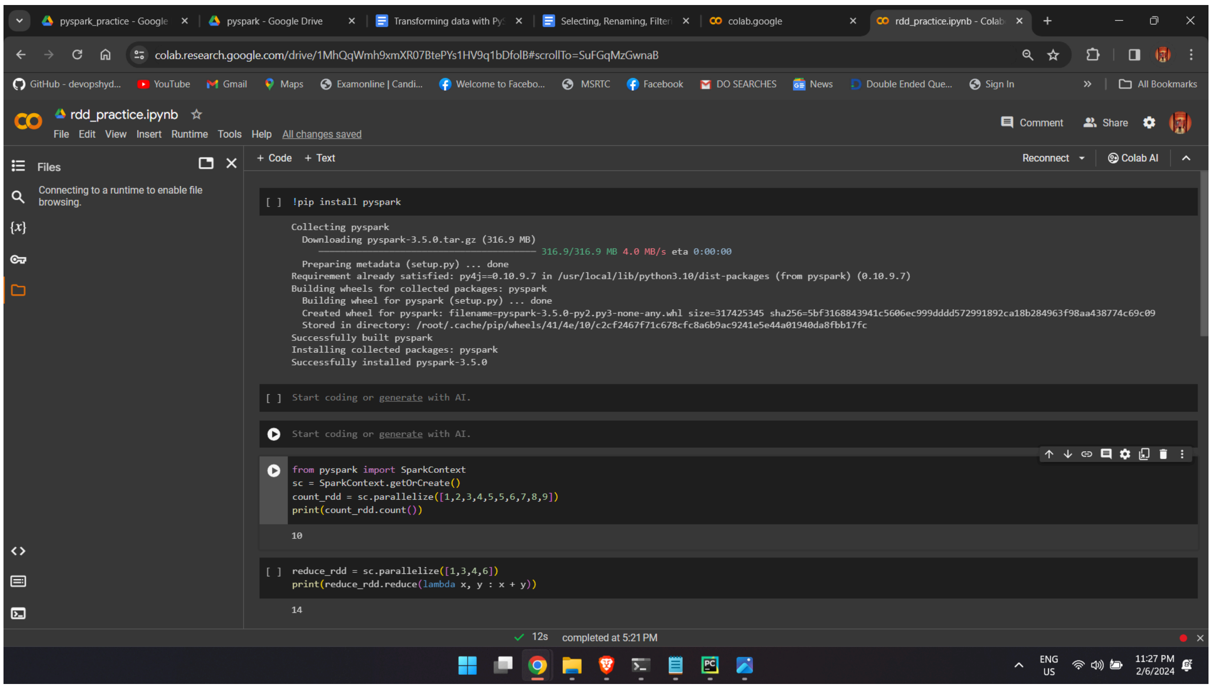
Actions:
Definition: Actions are operations that trigger computation on an RDD and return results to the driver program or write data to external storage systems.

Eager Evaluation: Actions are eager and trigger the execution of the transformation lineage, leading to actual computation.

Examples of Actions:

-collect(): Returns all elements of the RDD as an array to the driver program.
-count(): Returns the number of elements in the RDD.

-take(n): Returns the first n elements of the RDD.
-reduce(func): Aggregates the elements of the RDD using a provided function.
-foreach(func): Applies a function to each element of the RDD but does not return results to the driver program.



```
!pip install pyspark

Collecting pyspark
  Downloading pyspark-3.5.0.tar.gz (316.9 MB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 316.9/316.9 MB 4.0 MB/s eta 0:00:00
  Preparing metadata (setup.py) ... done
Requirement already satisfied: py4j==0.10.9.7 in /usr/local/lib/python3.10/dist-packages (from pyspark) (0.10.9.7)
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.5.0-py2.py3-none-any.whl size=317425345 sha256=5bf3168843941c5606ec999dddd572991892ca18b284963f98aa438774c69c09
  Stored in directory: /root/.cache/pip/wheels/41/4e/10/c2cf2467f71c678cfc8a6b9ac9241e5e44a01940da8fbb17fc
Successfully built pyspark
Installing collected packages: pyspark
Successfully installed pyspark-3.5.0
```

```
Start coding or generate with AI.
```

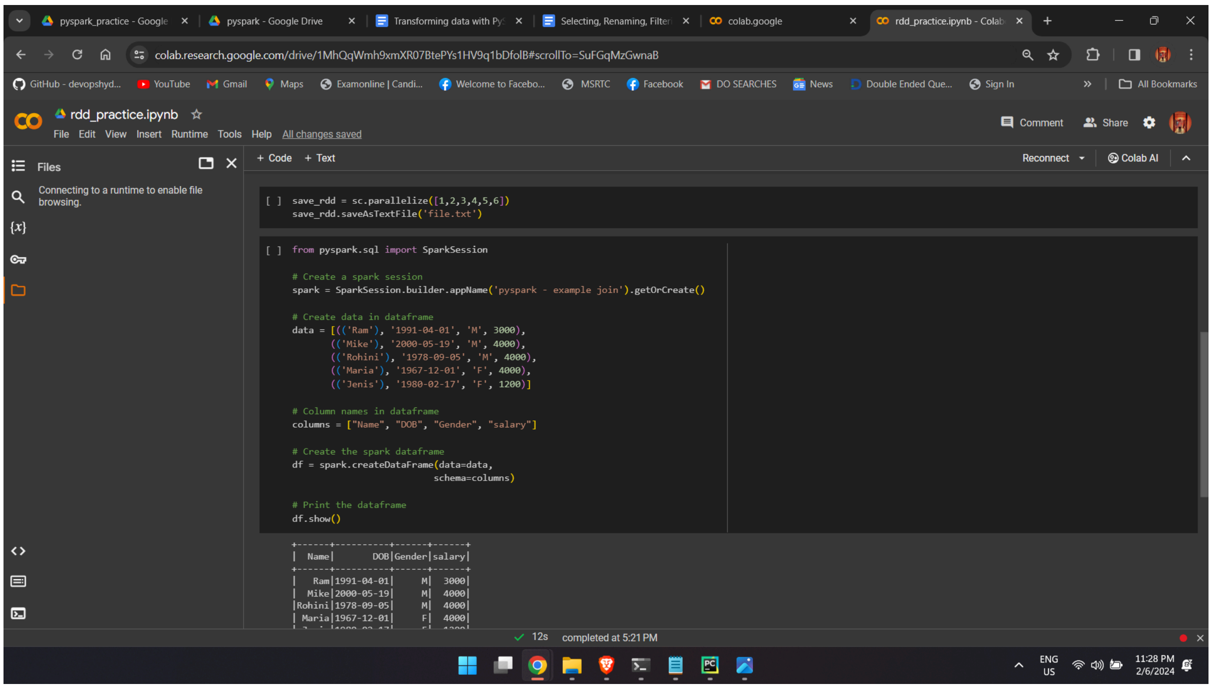
```
Start coding or generate with AI.
```

```python
from pyspark import SparkContext
sc = SparkContext.getOrCreate()
count_rdd = sc.parallelize([1,2,3,4,5,5,6,7,8,9])
print(count_rdd.count())
```

```
10
```

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

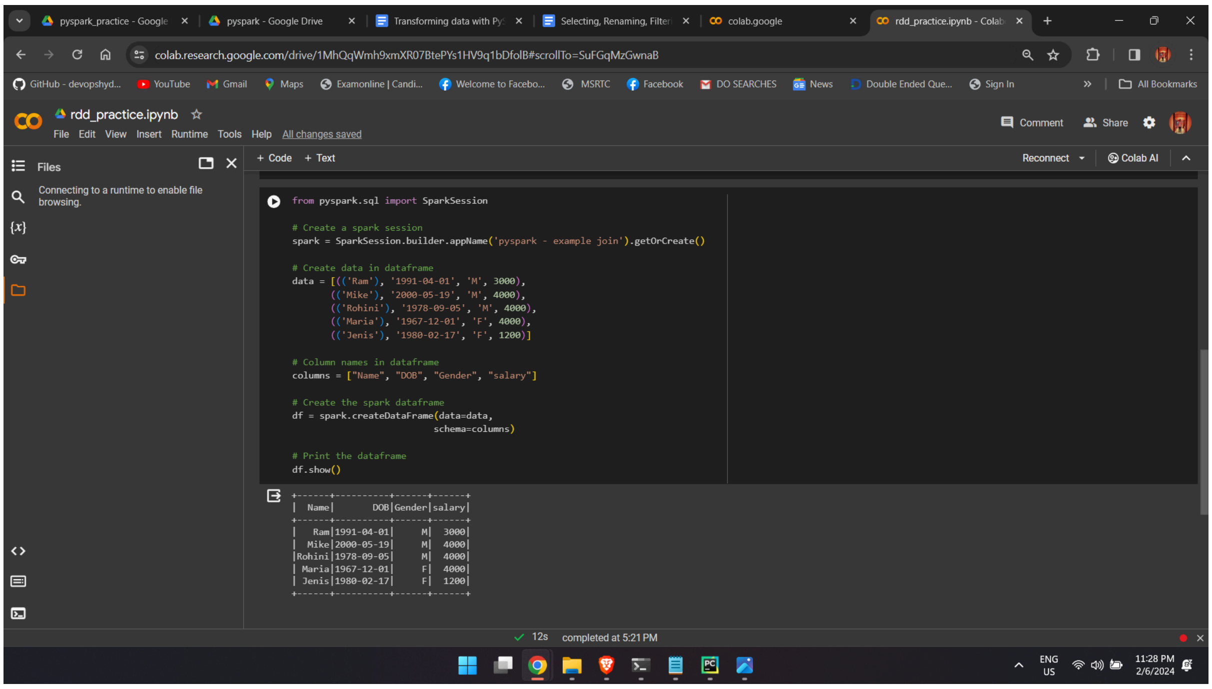
```
14
```



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

```python
from pyspark.sql import SparkSession

# Create a spark session
spark = SparkSession.builder.appName('pyspark - example join').getOrCreate()

# Create data in dataframe
data = [(('Ram'), '1991-04-01', 'M', 3000),
       (('Mike'), '2000-05-19', 'M', 4000),
       (('Rohini'), '1978-09-05', 'M', 4000),
       (('Maria'), '1967-12-01', 'F', 4000),
       (('Jenis'), '1980-02-17', 'F', 1200)]

# Column names in dataframe
columns = ["Name", "DOB", "Gender", "salary"]

# Create the spark dataframe
df = spark.createDataFrame(data=data,
                           schema=columns)

# Print the dataframe
df.show()
```

```
+------+----------+------+------+
|  Name|       DOB|Gender|salary|
+------+----------+------+------+
|   Ram|1991-04-01|     M|  3000|
|  Mike|2000-05-19|     M|  4000|
|Rohini|1978-09-05|     M|  4000|
| Maria|1967-12-01|     F|  4000|
```



rdd_practice.ipynb
File Edit View Insert Runtime Tools Help All changes saved
Files
Connecting to a runtime to enable file browsing.
+ Code + Text
Reconnect
Colab AI
Comment
Share
12s completed at 5:21 PM

```
from pyspark.sql import SparkSession

# Create a spark session
spark = SparkSession.builder.appName('pyspark - example join').getOrCreate()

# Create data in dataframe
data = [(('Ram'), '1991-04-01', 'M', 3000),
       (('Mike'), '2000-05-19', 'M', 4000),
       (('Rohini'), '1978-09-05', 'M', 4000),
       (('Maria'), '1967-12-01', 'F', 4000),
       (('Jenis'), '1980-02-17', 'F', 1200)]

# Column names in dataframe
columns = ["Name", "DOB", "Gender", "salary"]

# Create the spark dataframe
df = spark.createDataFrame(data=data,
                           schema=columns)

# Print the dataframe
df.show()
```

```
+------+----------+------+------+
|  Name|       DOB|Gender|salary|
+------+----------+------+------+
|   Ram|1991-04-01|     M|  3000|
|  Mike|2000-05-19|     M|  4000|
|Rohini|1978-09-05|     M|  4000|
| Maria|1967-12-01|     F|  4000|
| Jenis|1980-02-17|     F|  1200|
+------+----------+------+------+
```

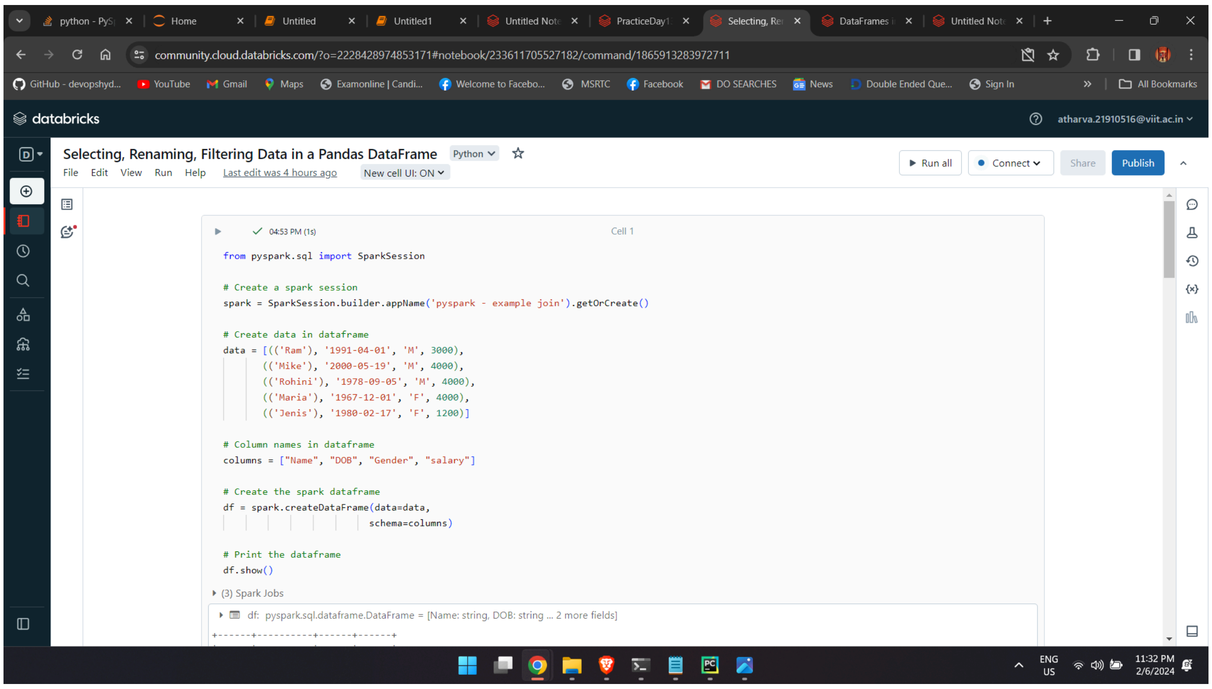
## Spark SQL - Creating databases, tables and views :

In Spark SQL, you can create databases, tables, and views to organize and query structured data. Here's how you can do it:

Creating Databases:
Database: A database is a namespace for tables. It's a logical container for tables and other database-related metadata.

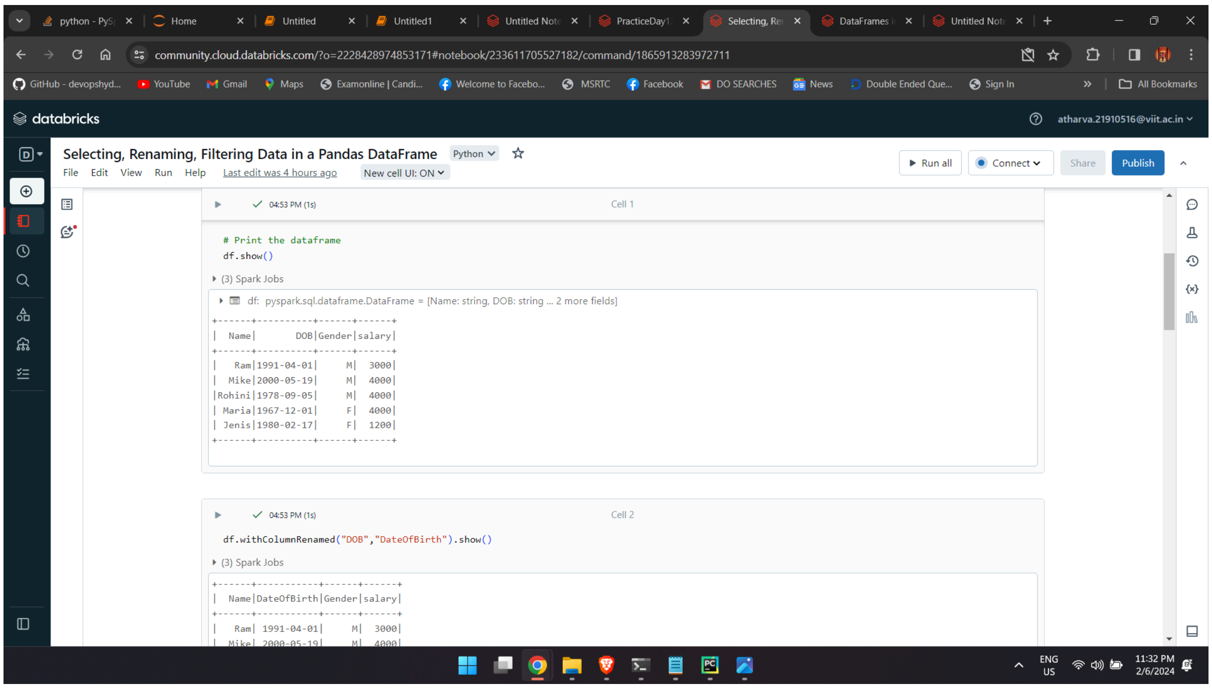
Creating Tables:
Table: A table is a structured collection of data. It has a defined schema that specifies the column names and data types.

Creating Views:
View: A view is a virtual table based on the result of a SQL query. It doesn't store data but provides a convenient way to query existing data.



```
from pyspark.sql import SparkSession

# Create a spark session
spark = SparkSession.builder.appName('pyspark - example join').getOrCreate()

# Create data in dataframe
data = [(('Ram'), '1991-04-01', 'M', 3000),
        (('Mike'), '2000-05-19', 'M', 4000),
        (('Rohini'), '1978-09-05', 'M', 4000),
        (('Maria'), '1967-12-01', 'F', 4000),
        (('Jenis'), '1980-02-17', 'F', 1200)]

# Column names in dataframe
columns = ["Name", "DOB", "Gender", "salary"]

# Create the spark dataframe
df = spark.createDataFrame(data=data,
                           schema=columns)

# Print the dataframe
df.show()
```

```
(3) Spark Jobs
df: pyspark.sql.dataframe.DataFrame = [Name: string, DOB: string ... 2 more fields]
+------+----------+------+------+
```



Selecting, Renaming, Filtering Data in a Pandas DataFrame

Cell 1

```python
# Print the dataframe
df.show()
```

(3) Spark Jobs

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

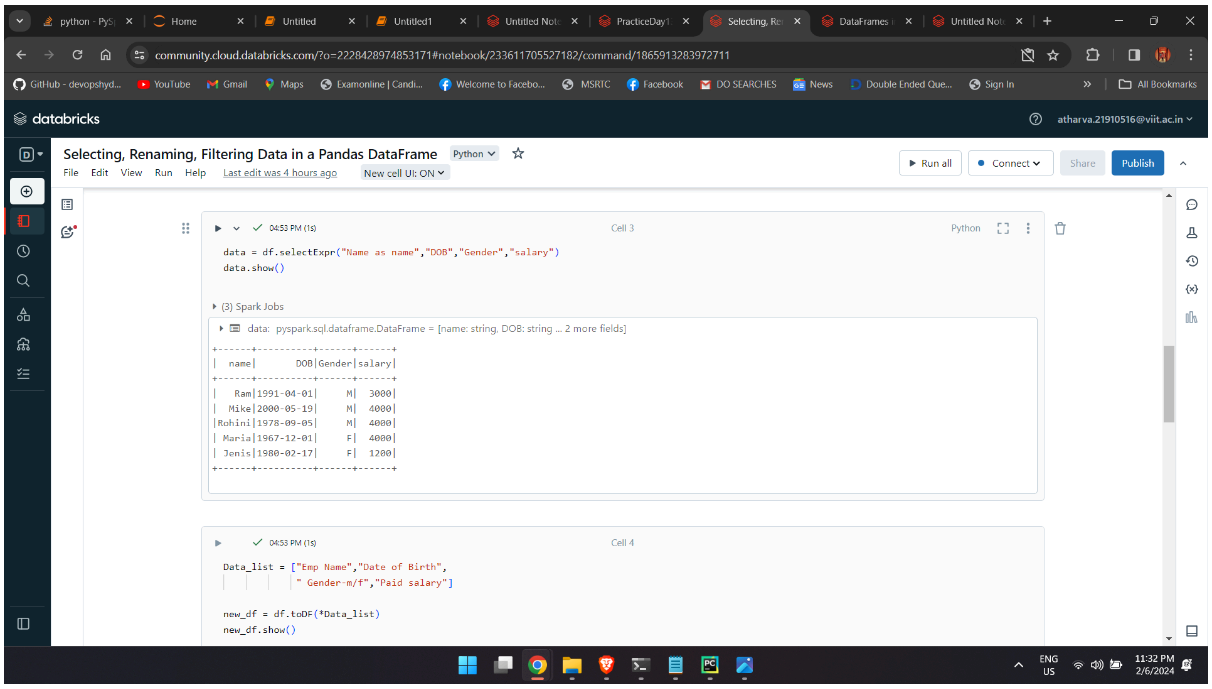
```
+------+----------+------+------+
|  Name|       DOB|Gender|salary|
+------+----------+------+------+
|   Ram|1991-04-01|     M|  3000|
|  Mike|2000-05-19|     M|  4000|
|Rohini|1978-09-05|     M|  4000|
| Maria|1967-12-01|     F|  4000|
| Jenis|1980-02-17|     F|  1200|
+------+----------+------+------+
```

Cell 2

```python
df.withColumnRenamed("DOB","DateOfBirth").show()
```

(3) Spark Jobs

```
+------+-----------+------+------+
|  Name|DateOfBirth|Gender|salary|
+------+-----------+------+------+
|   Ram| 1991-04-01|     M|  3000|
|  Mike| 2000-05-19|     M|  4000|
```



Selecting, Renaming, Filtering Data in a Pandas DataFrame

Cell 3

```python
data = df.selectExpr("Name as name","DOB","Gender","salary")
data.show()
```

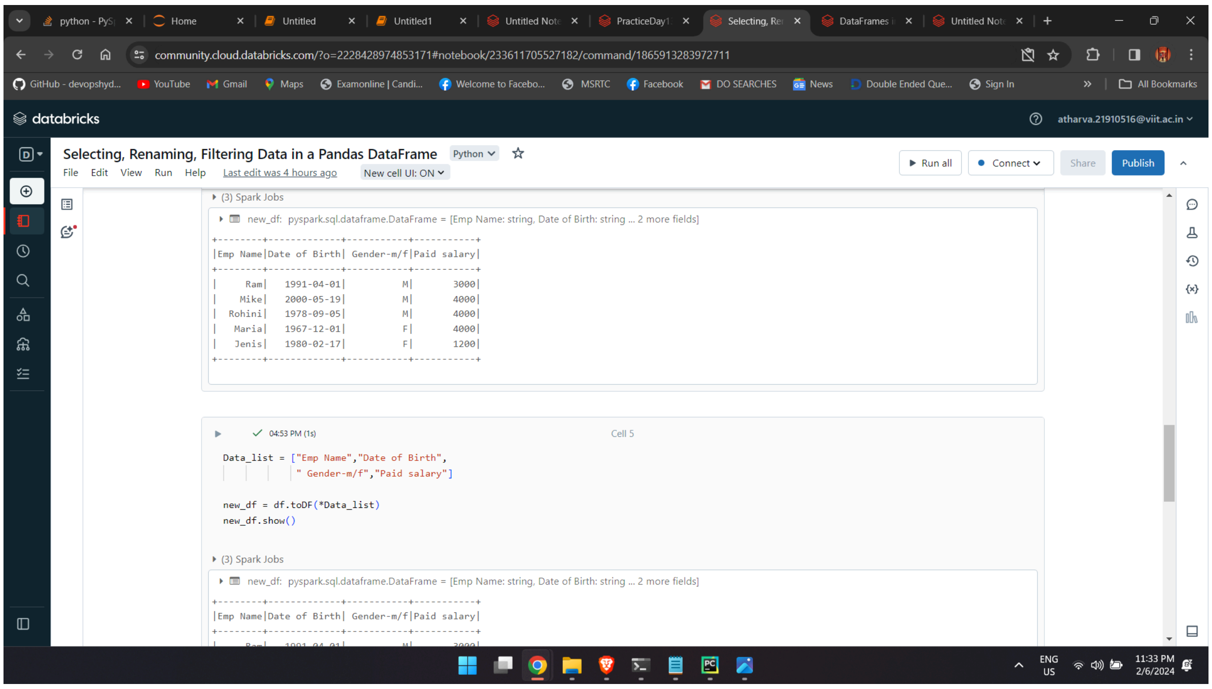
(3) Spark Jobs

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

```
+------+----------+------+------+
|  name|       DOB|Gender|salary|
+------+----------+------+------+
|   Ram|1991-04-01|     M|  3000|
|  Mike|2000-05-19|     M|  4000|
|Rohini|1978-09-05|     M|  4000|
| Maria|1967-12-01|     F|  4000|
| Jenis|1980-02-17|     F|  1200|
+------+----------+------+------+
```

Cell 4

```python
Data_list = ["Emp Name","Date of Birth",
             " Gender-m/f","Paid salary"]

new_df = df.toDF(*Data_list)
new_df.show()
```


databricks
Selecting, Renaming, Filtering Data in a Pandas DataFrame
Python
File Edit View Run Help
Last edit was 4 hours ago
New cell UI: ON
Run all
Connect
Share
Publish
atharva.21910516@viit.ac.in
community.cloud.databricks.com/?o=2228428974853171#notebook/233611705527182/command/1865913283972711

(3) Spark Jobs

new_df: pyspark.sql.dataframe.DataFrame = [Emp Name: string, Date of Birth: string ... 2 more fields]

```
+--------+-------------+----------+-----------+
|Emp Name|Date of Birth| Gender-m/f|Paid salary|
+--------+-------------+----------+-----------+
|     Ram|   1991-04-01|         M|       3000|
|    Mike|   2000-05-19|         M|       4000|
|  Rohini|   1978-09-05|         M|       4000|
|   Maria|   1967-12-01|         F|       4000|
|   Jenis|   1980-02-17|         F|       1200|
+--------+-------------+----------+-----------+
```

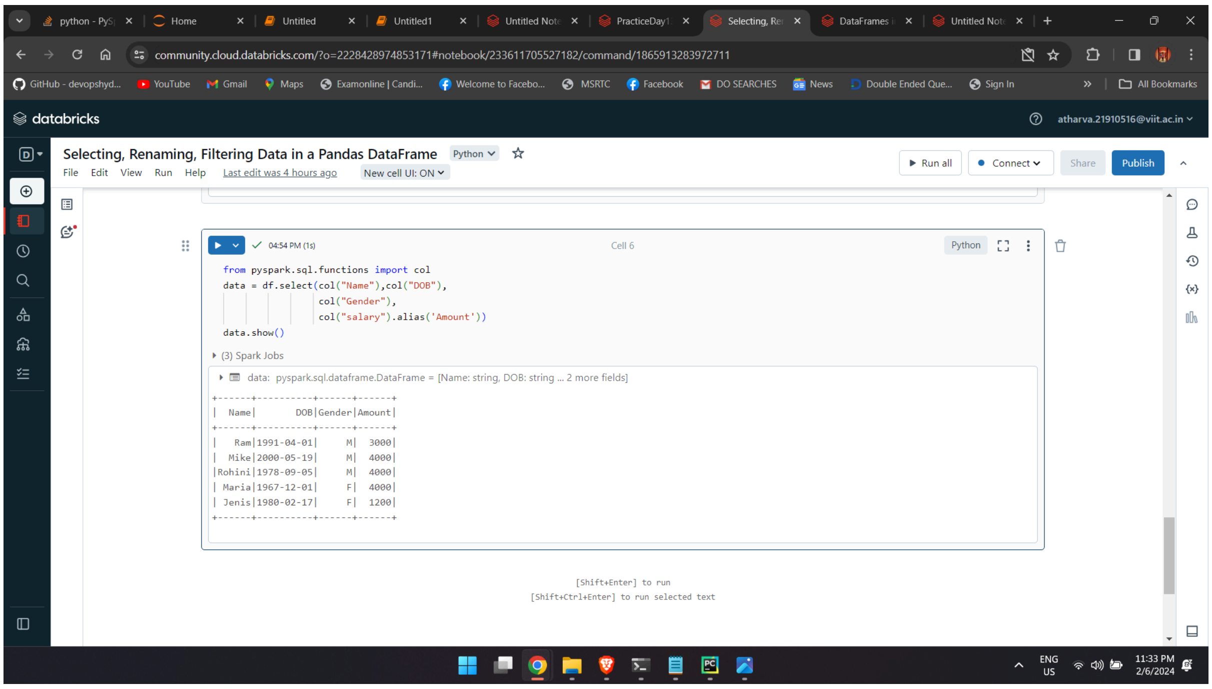
04:53 PM (1s) Cell 5

```python
Data_list = ["Emp Name","Date of Birth",
             " Gender-m/f","Paid salary"]

new_df = df.toDF(*Data_list)
new_df.show()
```

(3) Spark Jobs

new_df: pyspark.sql.dataframe.DataFrame = [Emp Name: string, Date of Birth: string ... 2 more fields]

```
+--------+-------------+----------+-----------+
|Emp Name|Date of Birth| Gender-m/f|Paid salary|
+--------+-------------+----------+-----------+
```


databricks
Selecting, Renaming, Filtering Data in a Pandas DataFrame
Python
File Edit View Run Help
Last edit was 4 hours ago
New cell UI: ON
Run all
Connect
Share
Publish
atharva.21910516@viit.ac.in

04:54 PM (1s) Cell 6

```python
from pyspark.sql.functions import col
data = df.select(col("Name"),col("DOB"),
                 col("Gender"),
                 col("salary").alias('Amount'))
data.show()
```

(3) Spark Jobs

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

```
+------+----------+------+------+
|  Name|       DOB|Gender|Amount|
+------+----------+------+------+
|   Ram|1991-04-01|     M|  3000|
|  Mike|2000-05-19|     M|  4000|
|Rohini|1978-09-05|     M|  4000|
| Maria|1967-12-01|     F|  4000|
| Jenis|1980-02-17|     F|  1200|
+------+----------+------+------+
```

[Shift+Enter] to run
[Shift+Ctrl+Enter] to run selected text