PySpark Coding Challenge

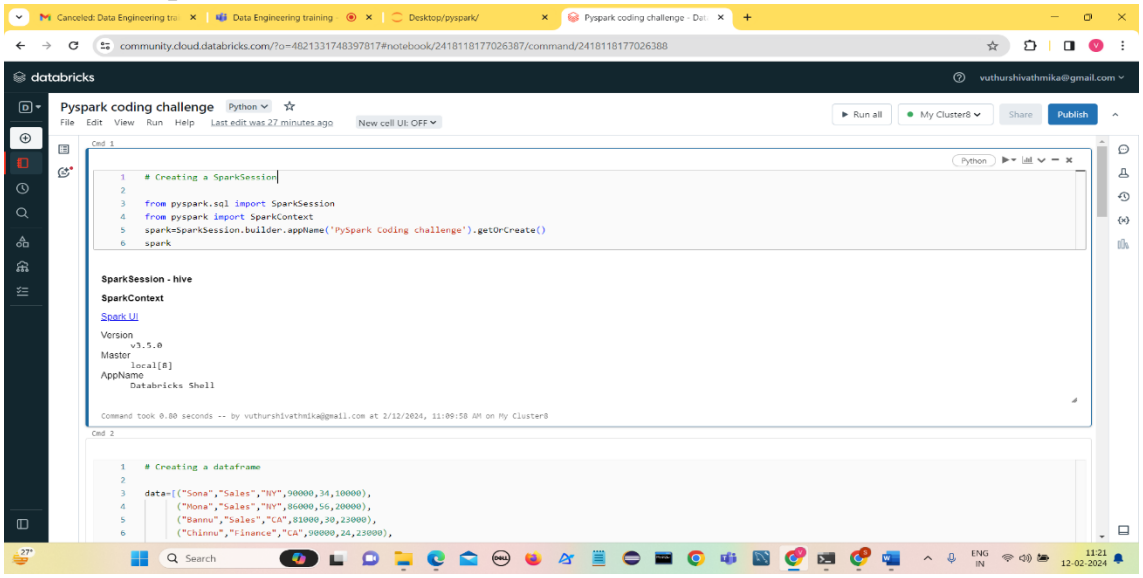
1) Execute Manipulating, Droping, Sorting, Aggregations, Joining, GroupeBy DataFrames

a) Manipulating Dataframes

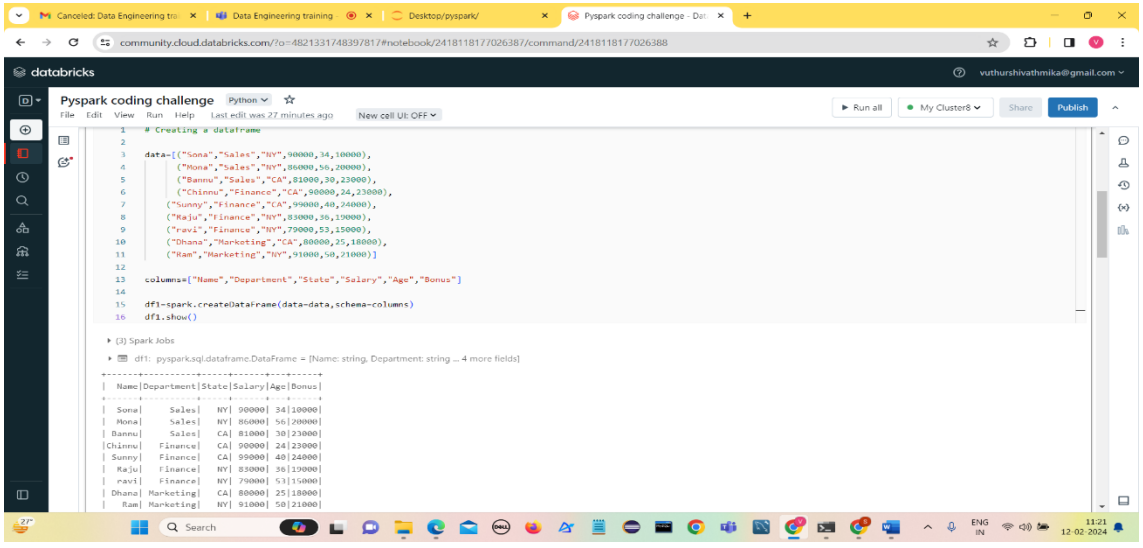
Manipulating dataframes include printing schema, displaying the records, Counting number of rows and columns, names of columns , adding new columns etc.,

Explanation:

Create a Spark Session

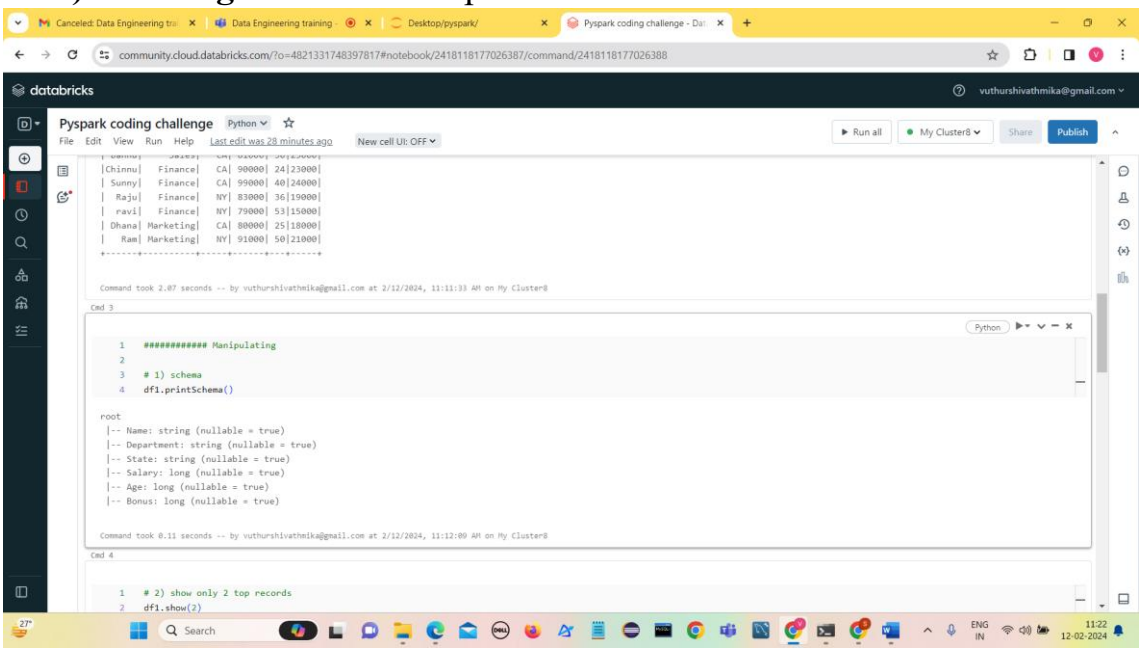


Create a dataframe using Spark Session containing data and columns



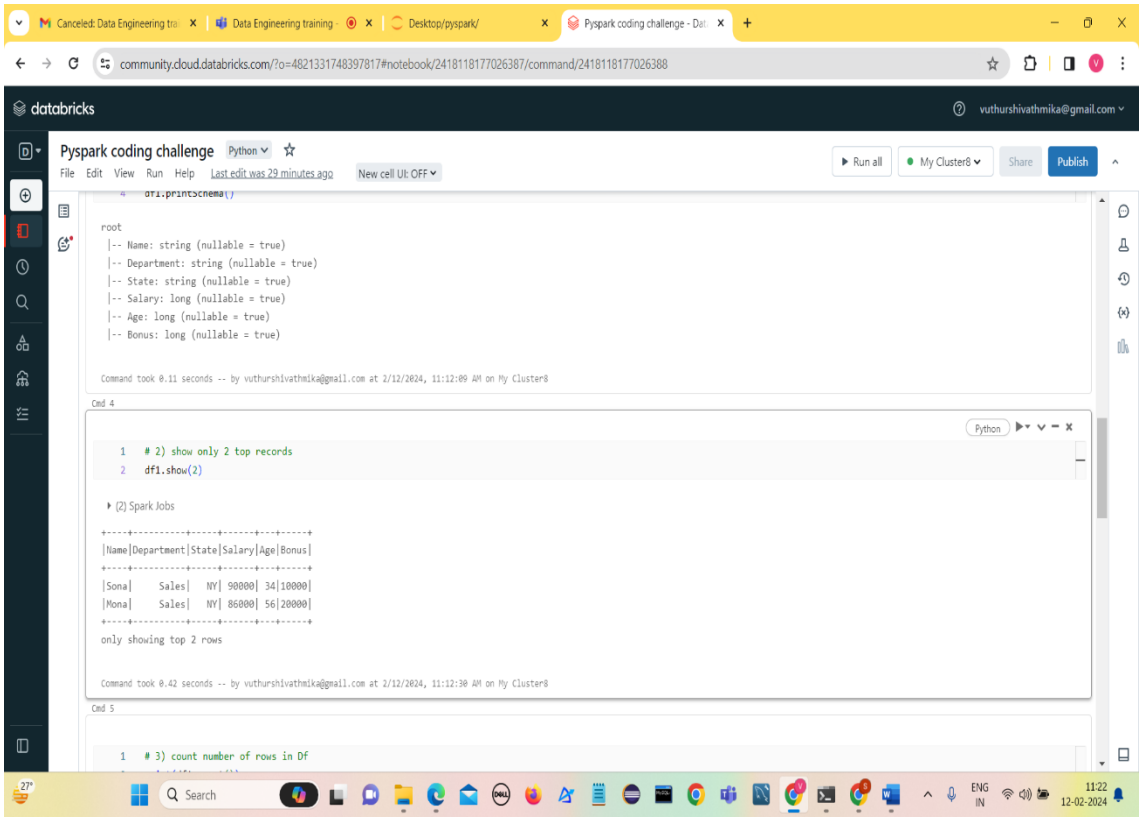
a) Manipulating

1) Printing the Schema—prints the schema of dataframe

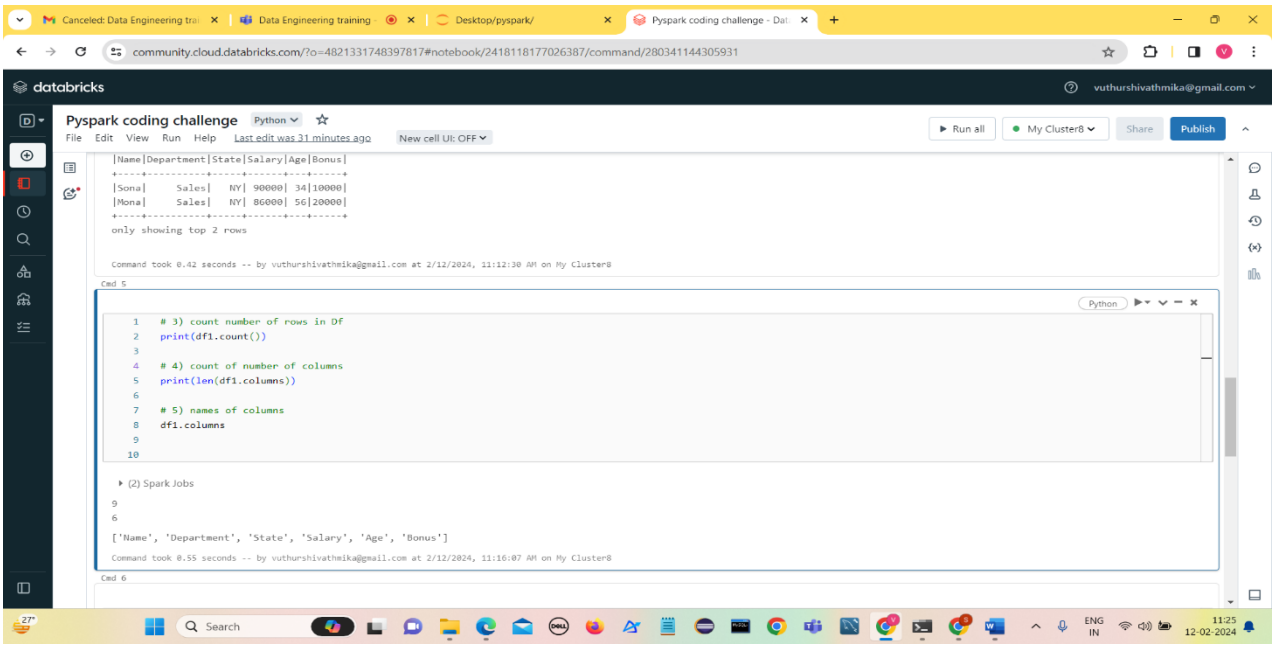


2) Displaying required number of rows

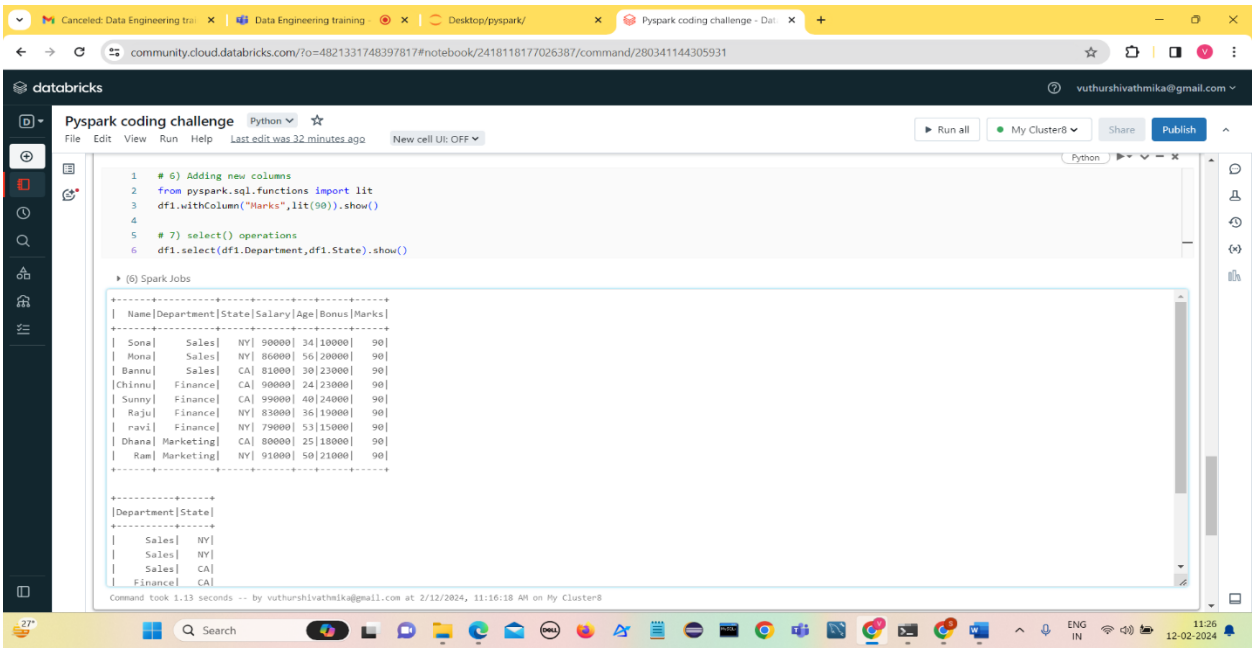
The code is used to display the required number of rows from dataframe.



3) **Count number of rows in a dataframe** : It counts number of rows in dataframe
4) **Count number of columns:** It returns the number of columns in a dataframe
5) **Names of columns:** It returns the names of columns of dataframe



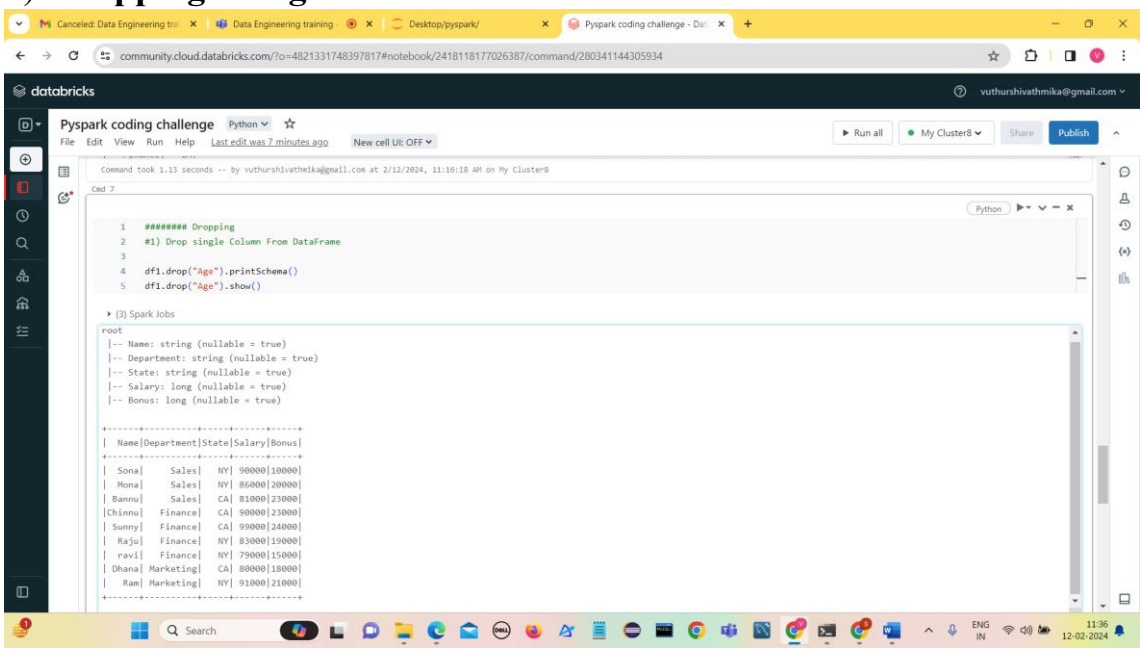
6) **Adding new columns**—It adds new columns using withColumn and lit() is used to fill the column with constant
7) **Select operations**—select() is used to select the required data from dataframe



b) Dropping

Dropping in dataframe means dropping the columns from the dataframe. Dropping can be done in two ways like we can drop a single column or at a time we can drop multiple columns.

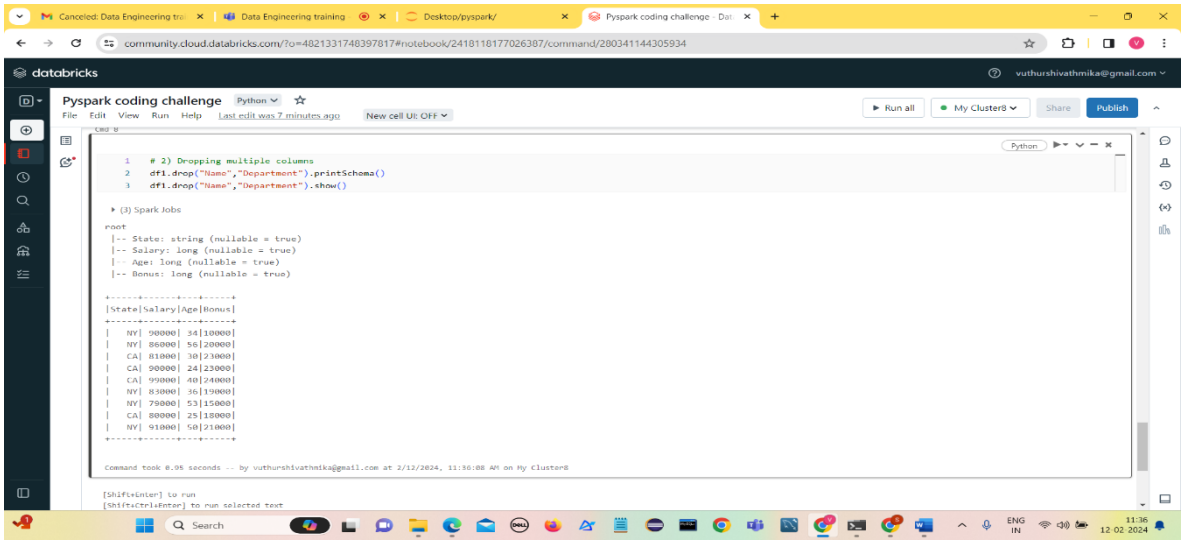
1) Dropping a single column



Explanation:

In the above code Age column is dropped using drop column in dataframe.

2) Dropping two columns or multiple columns at a time

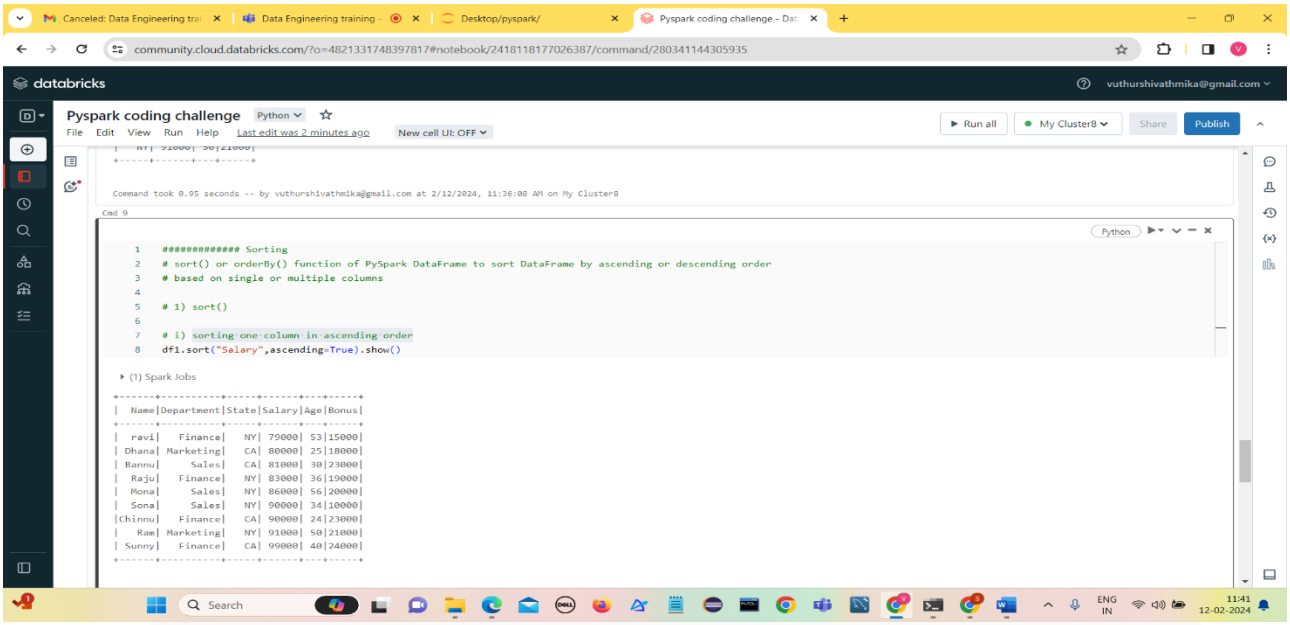


In the above code two columns Name and Department are dropped.

c) Sorting

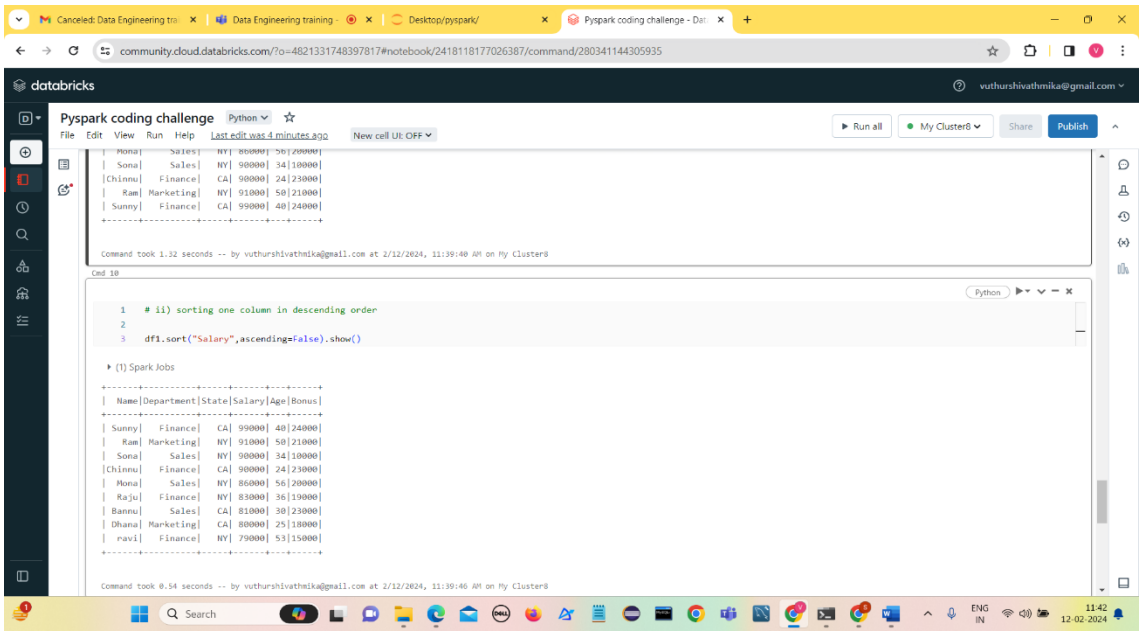
sort() or orderBy() function of PySpark DataFrame is used to sort DataFrame by ascending or descending order based on single or multiple columns.

1) Sort single column in ascending order.



In this code sort method is used to sort Salary column in ascending order

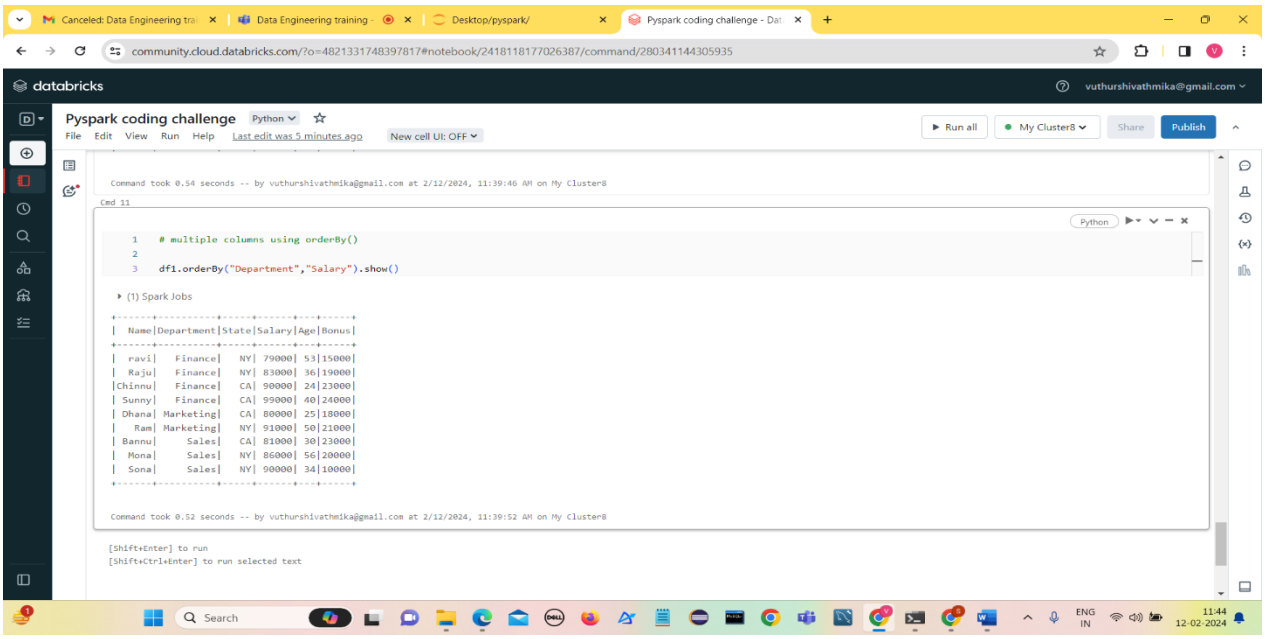
2) Sorting in decending order



In this code sort method is used to sort Salary column in descending order

3) orderBy()

orderBy() is also same as sort() method used to sort in ascending or descending order.

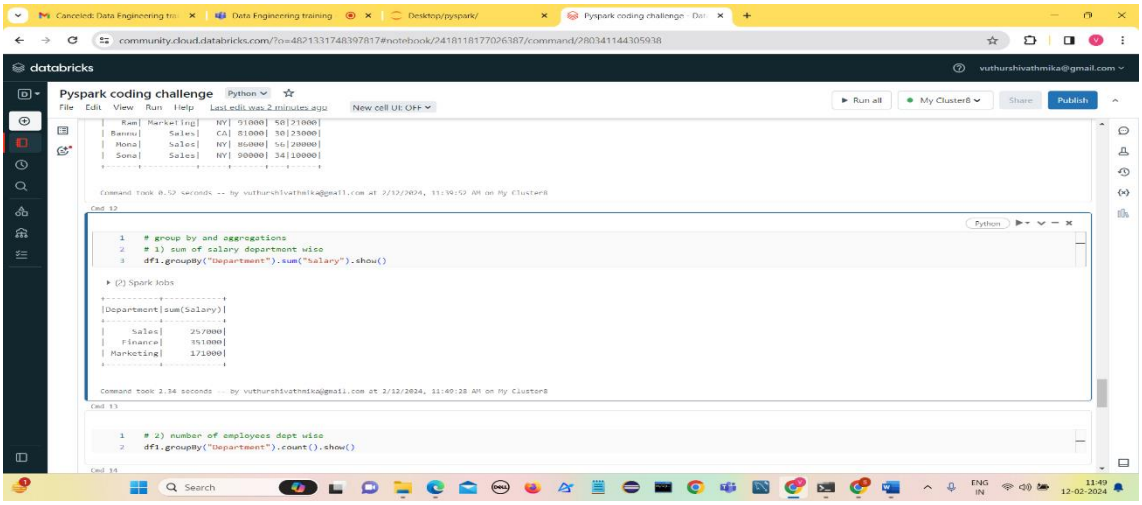


d) Group by

e) Aggregations

PySpark groupBy() function is used to collect the identical data into groups on DataFrame and perform count, sum, avg, min, and max functions on the grouped data.

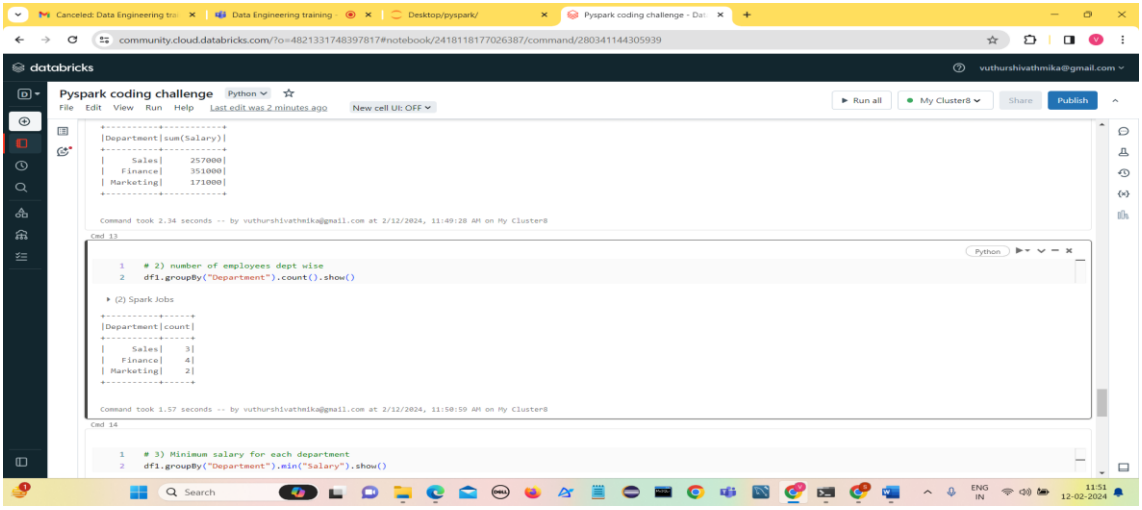
1) sum of salary department wise



The above code gives the sum of salary department wise

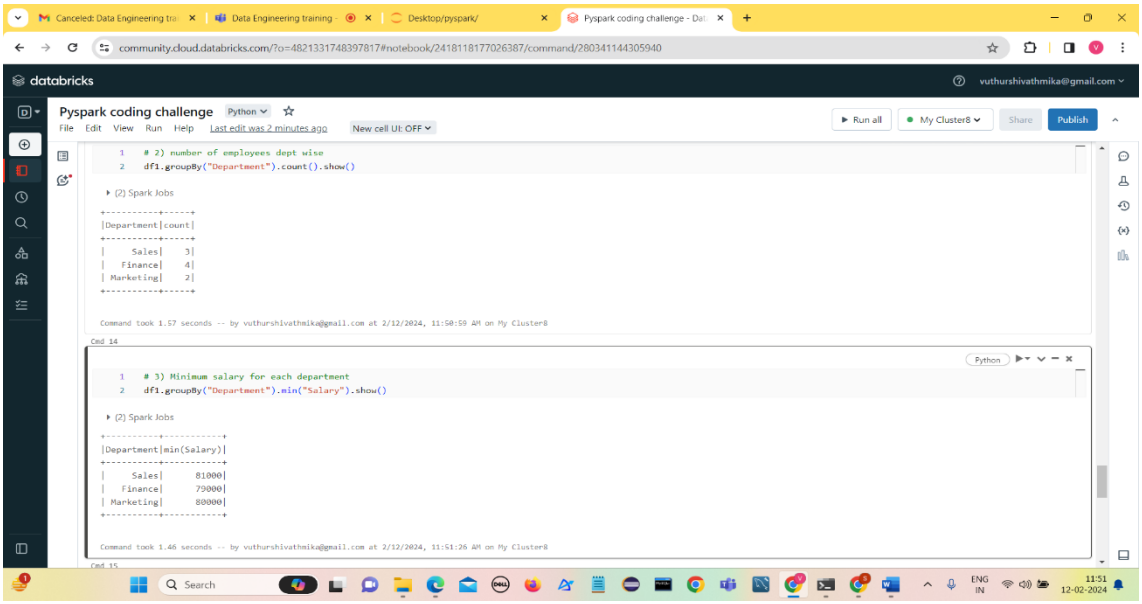
2) Count()

This code gives the number of employees in each department

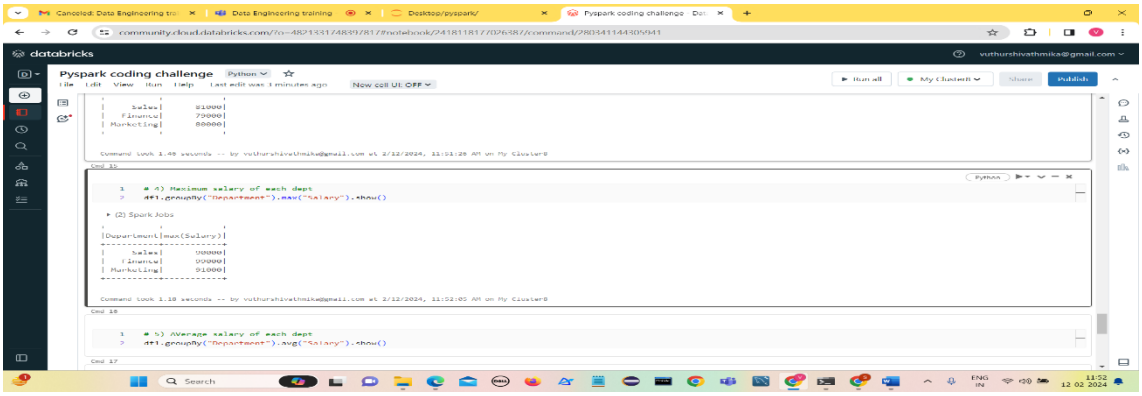


3) Min()

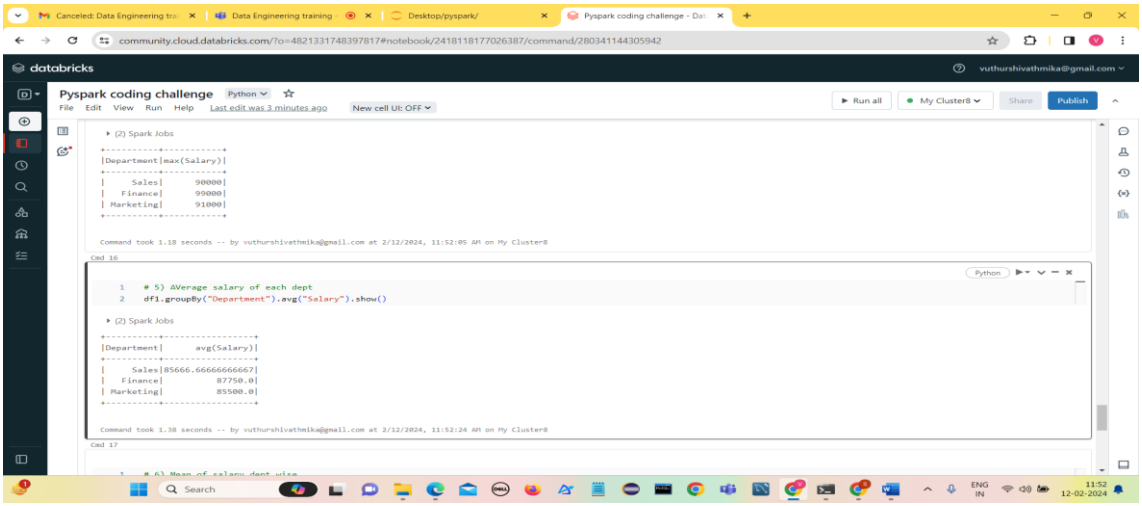
This code the minimum salary in each department



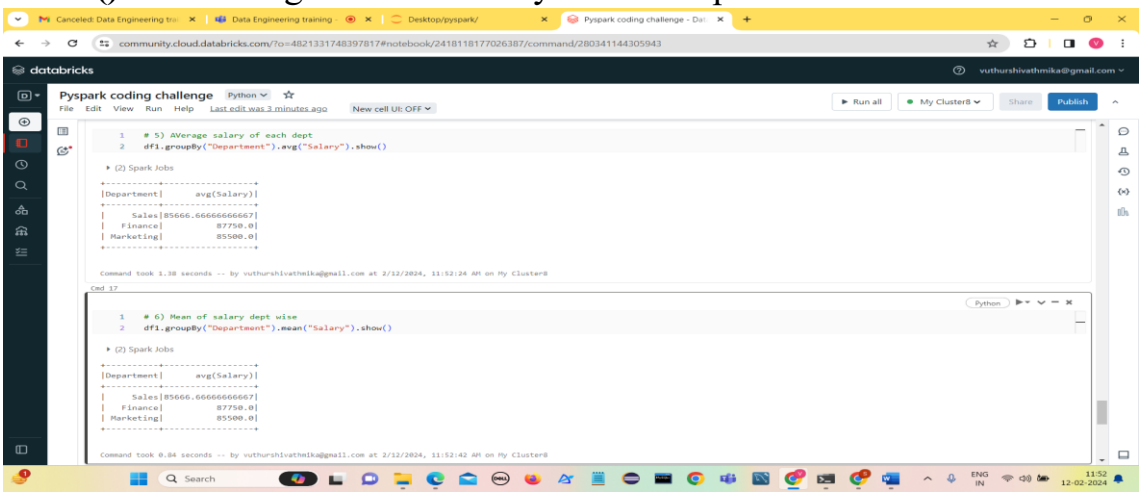
4) Max() This code gives maximum salary in each department



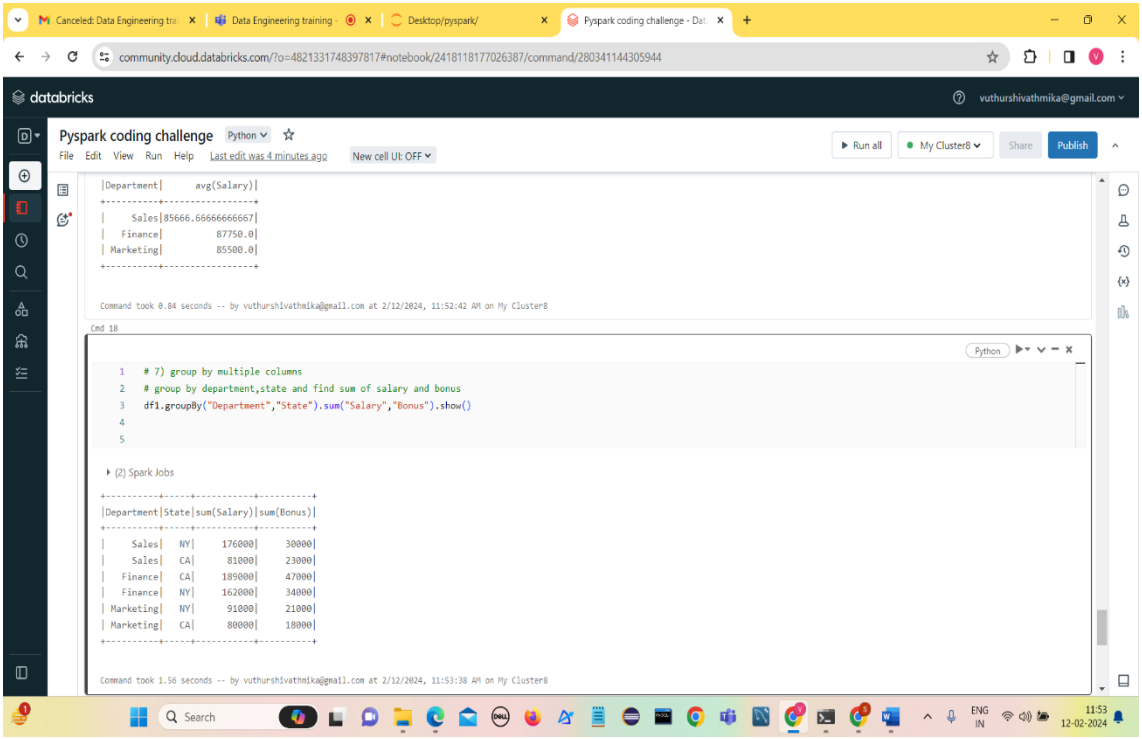
5) Avg() This code gives average salary in each department



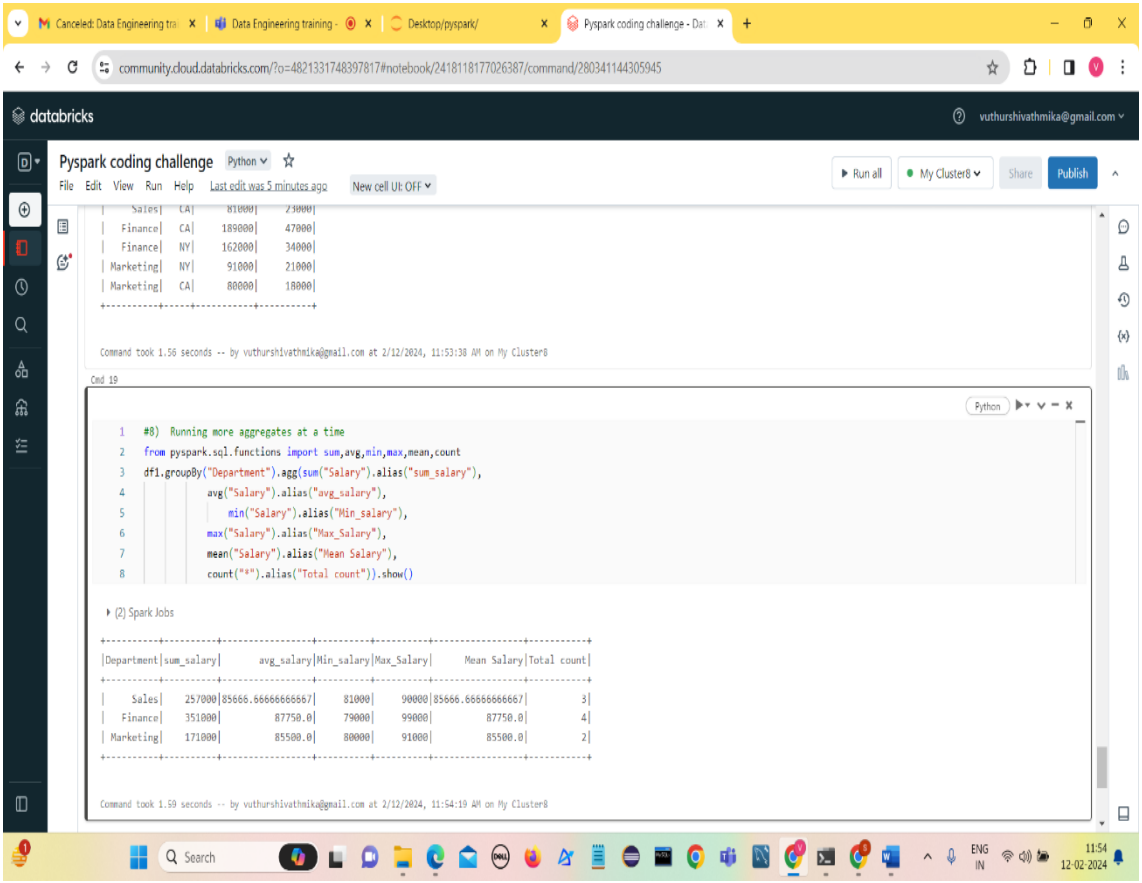
6) Mean() This code gives mean salary in each department



7) Group by multiple columns and sum of two columns



8) Running more than one aggregates at a time using agg() function



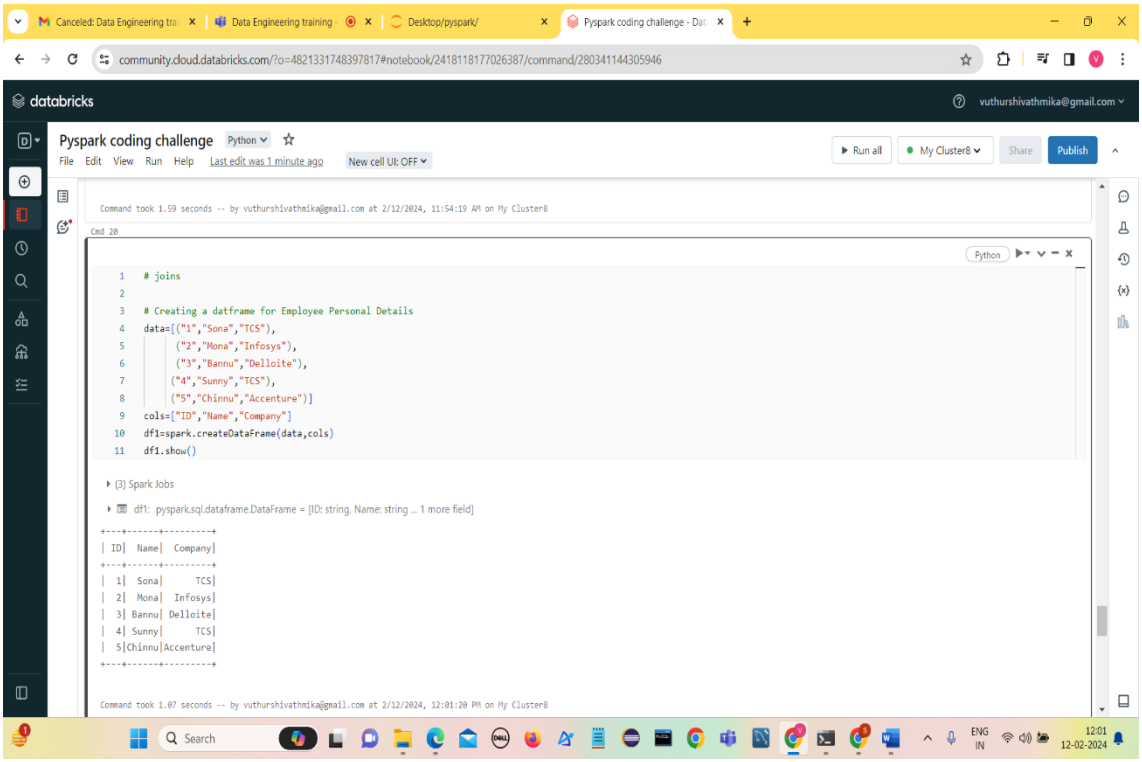
f) Joins

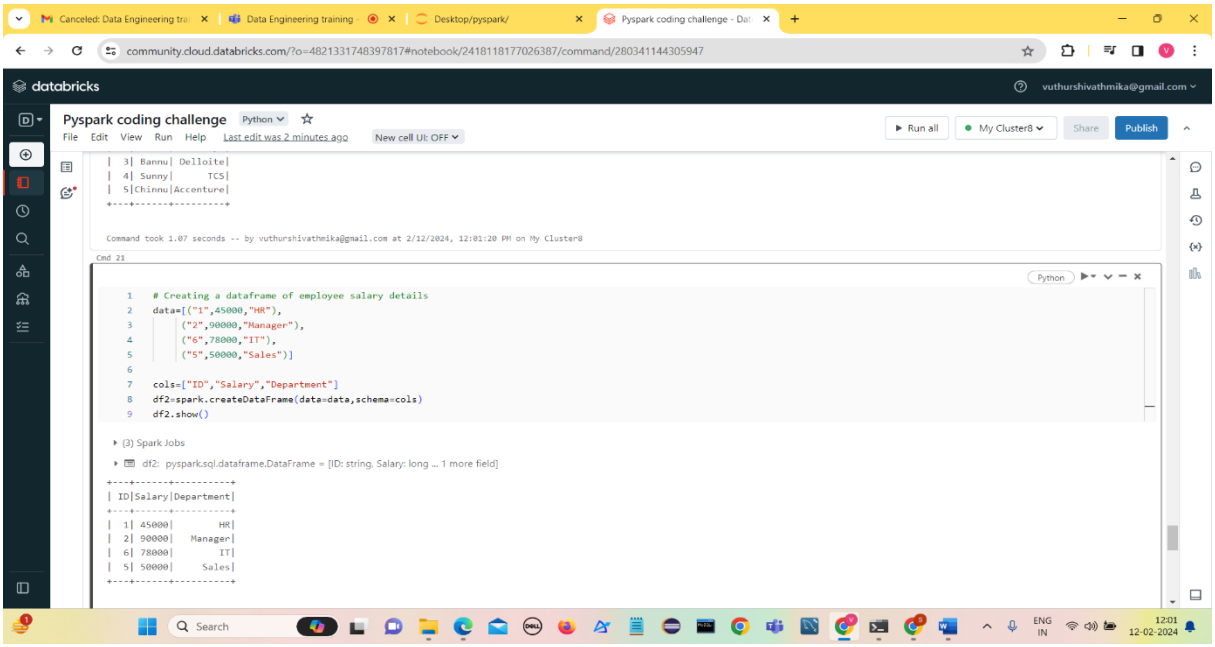
PySpark Join is used to combine two DataFrames based on the common column.

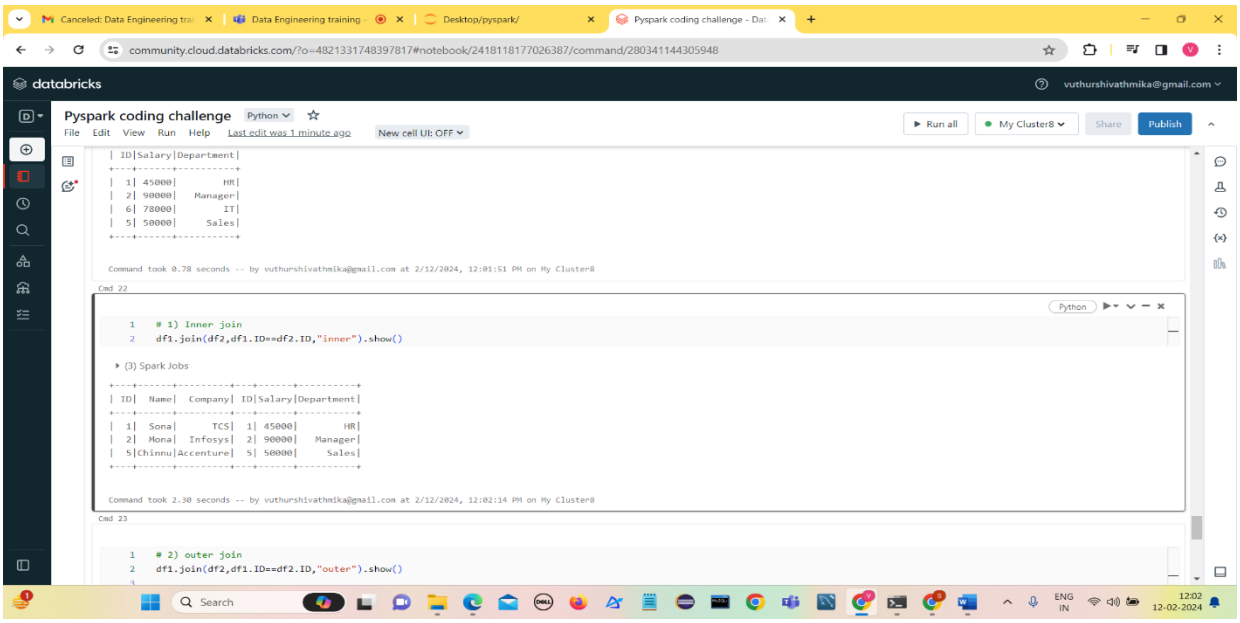
1) Inner Join:

Join records when key values are matched, and dropped when they are not matched.

Create two dataframes employee personal details and employee salary details for performing joins.

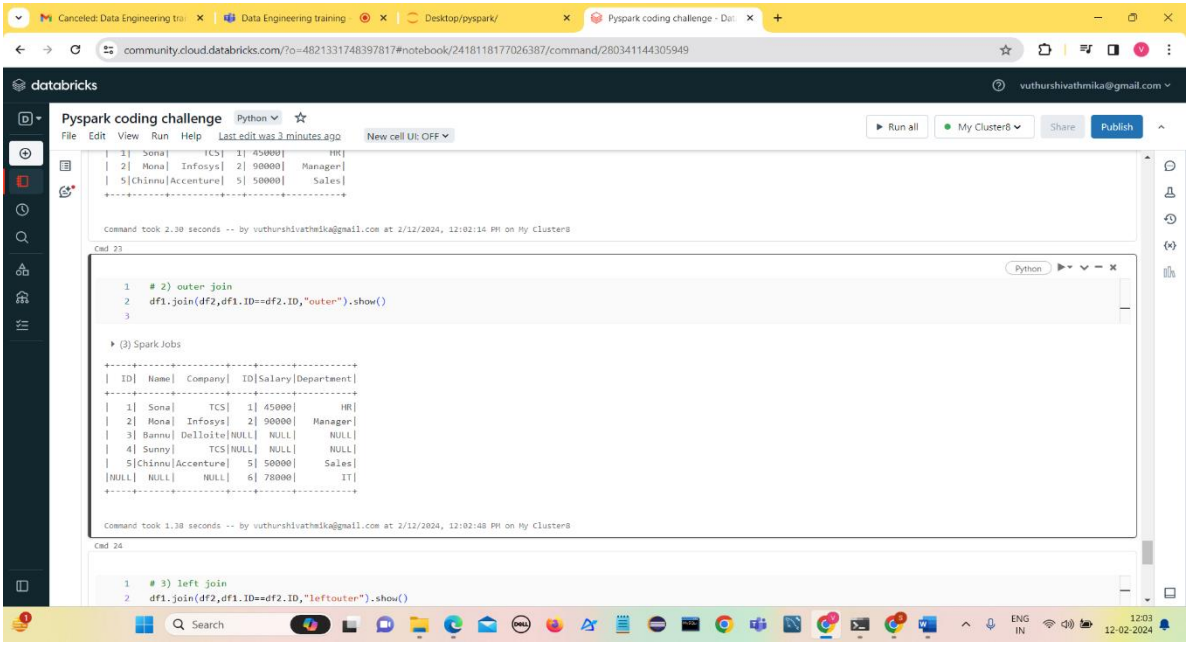






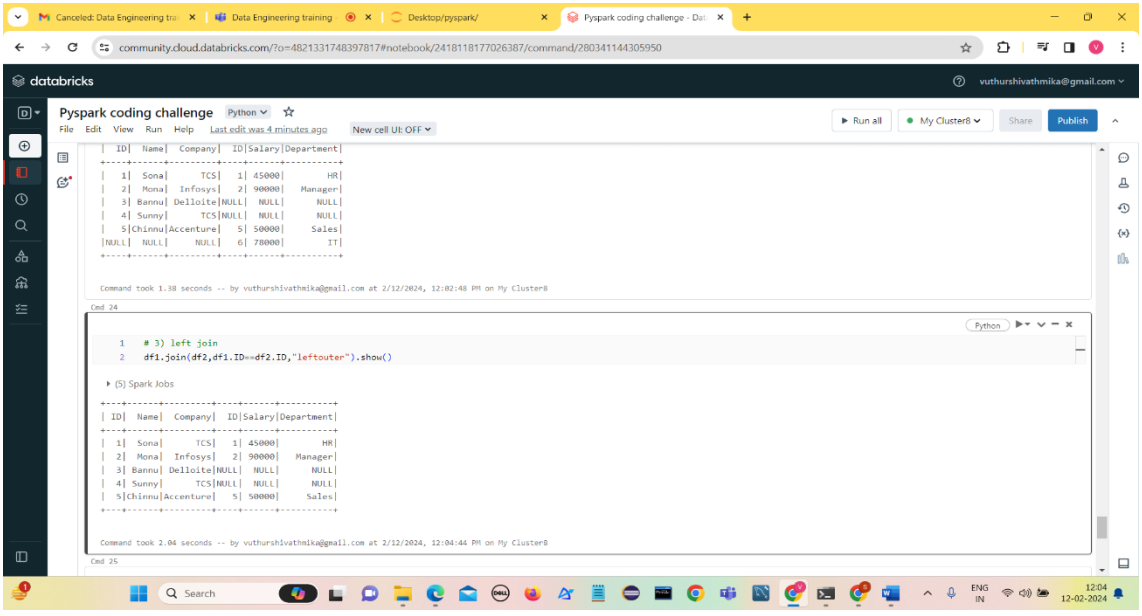
2) Outer Join

Returns all rows from both datasets, if join expression doesn't match it returns null values. (right join + left join)



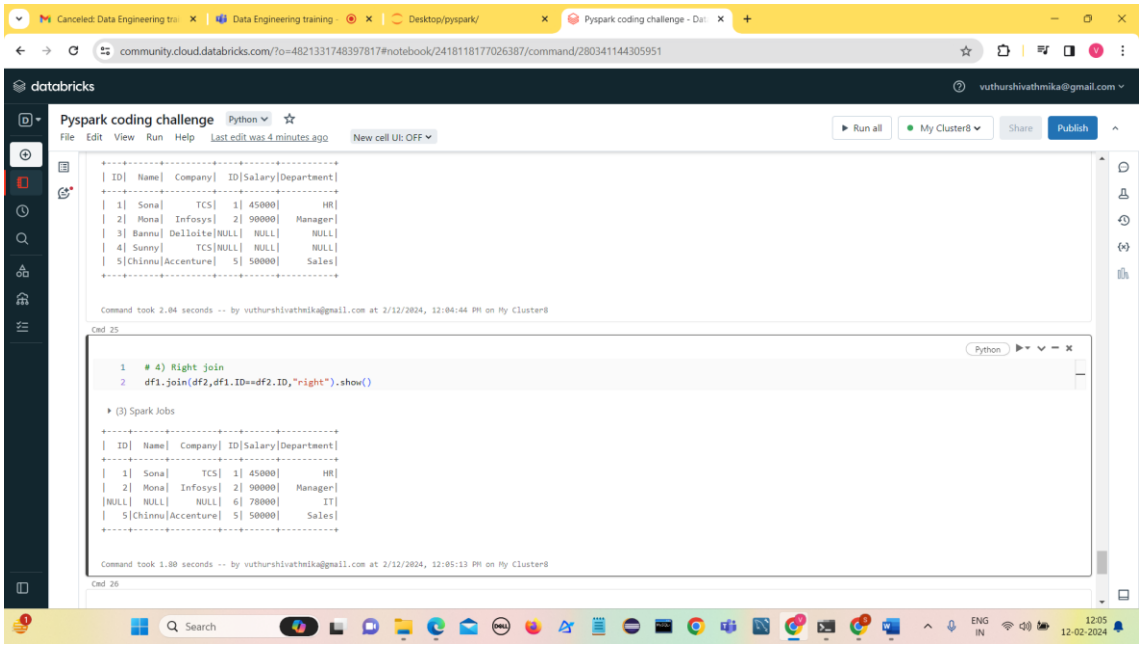
3) Left Join

Returns all rows from the left dataset irrespective of match found on the right dataset ,when join doesn't match it assigns null for that record



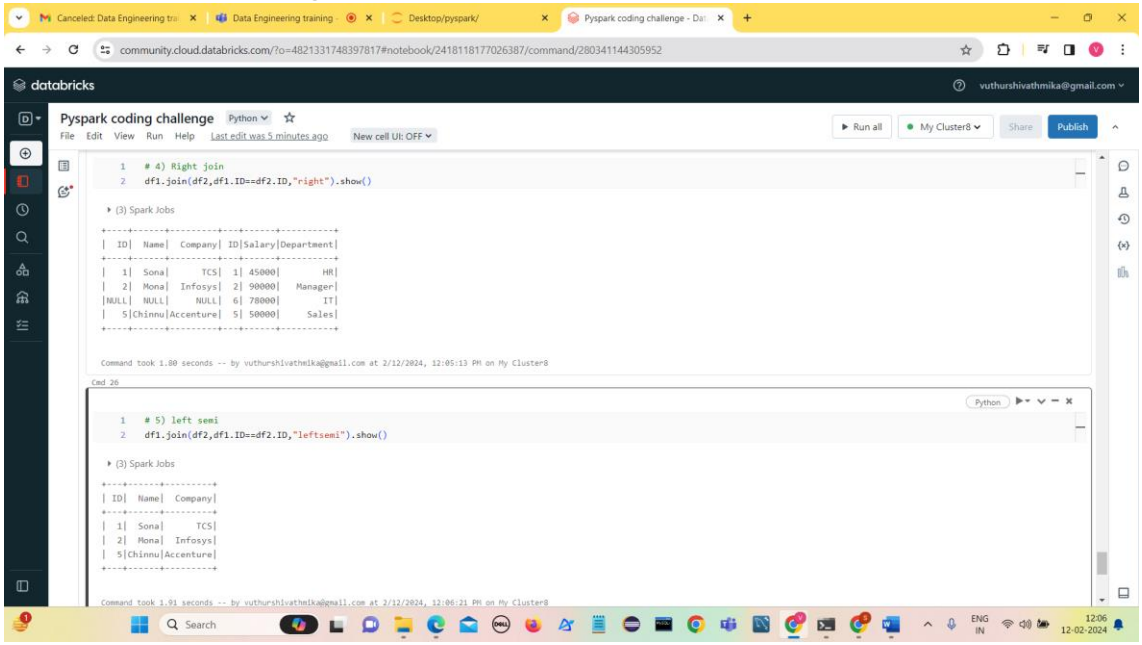
4) Right join

Returns all rows from the right dataset irrespective of match found on the left dataset ,when join doesn't match it assigns null for that record



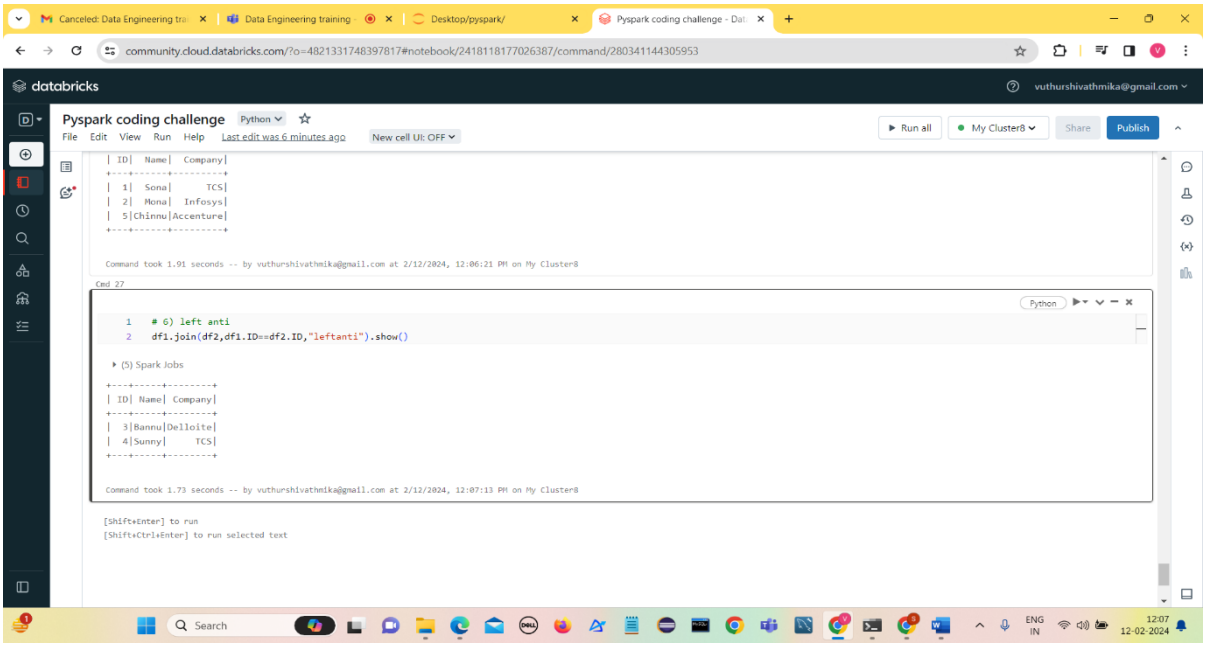
5) Left semi join

returns columns from the only left dataset for the records match in the right dataset on join expression, records not matched on join expression are ignored from both left and right datasets.



6) Left Anti Join

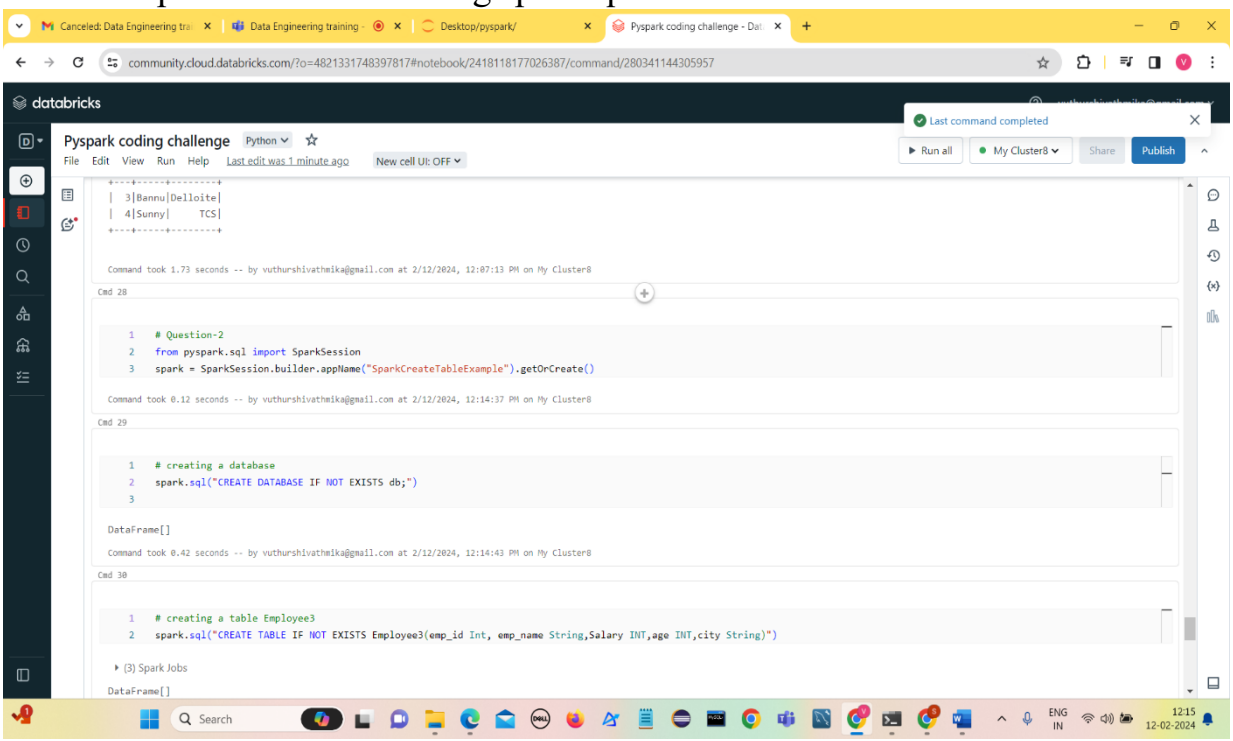
leftanti join returns only columns from the left dataset for non-matched record



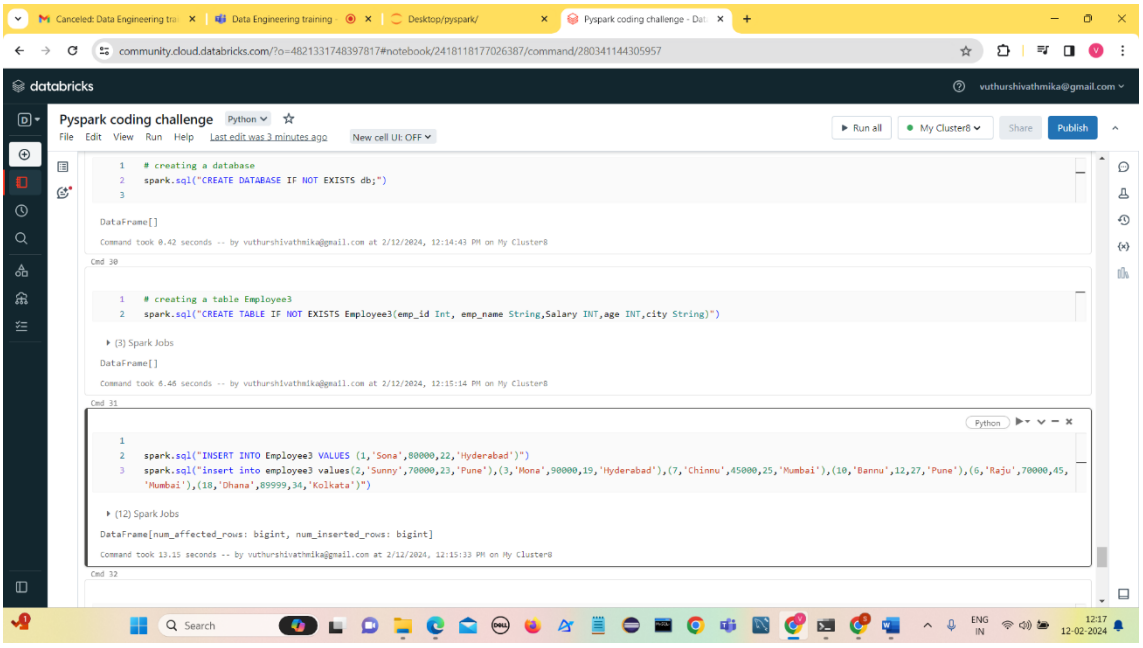
Question-2
Execute Pyspark -sparksql joins & Applying Functions in a Pandas DataFrame

Pyspark SQL JOINS

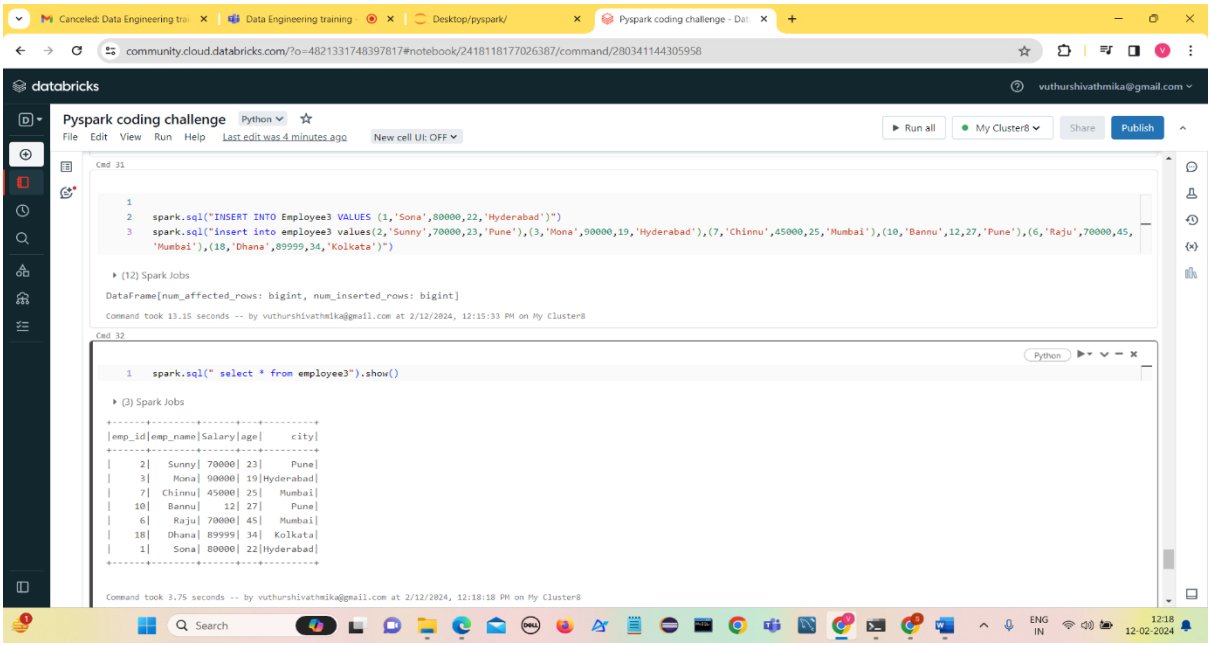
Create a SparkSession and using sparksql create a database



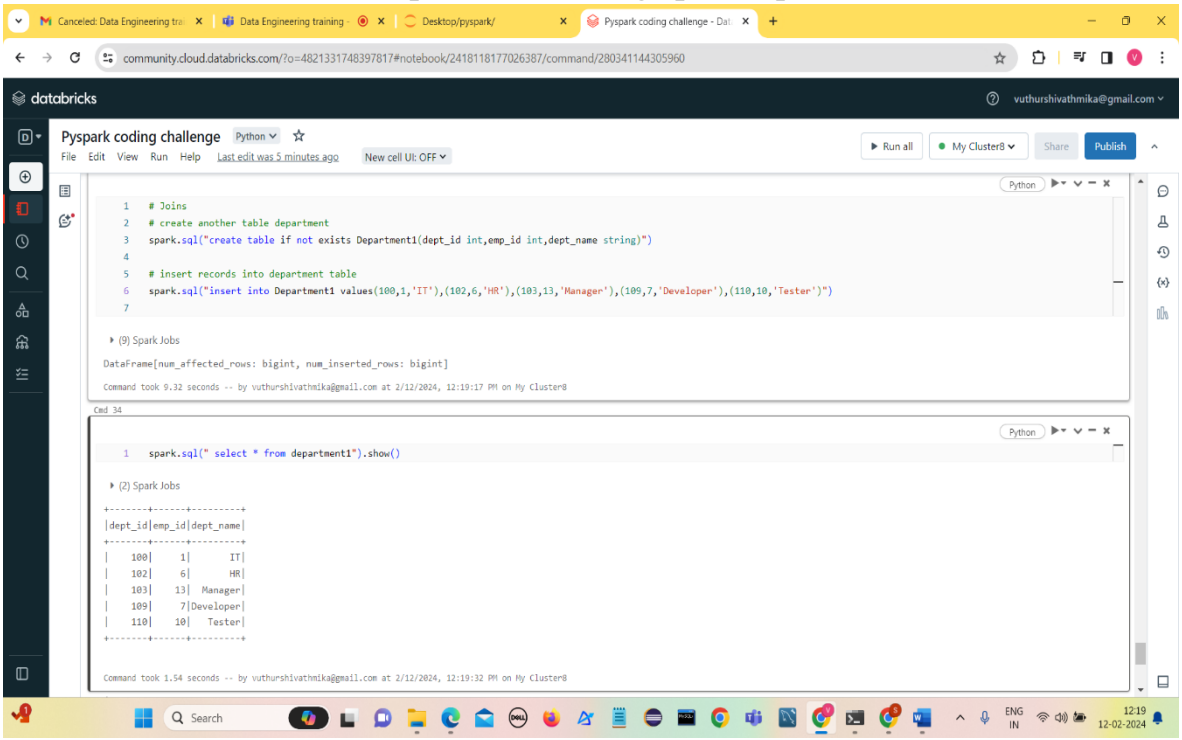
After creating a database create a table Employee3 and insert values into it



Displaying the records of Employee3 table

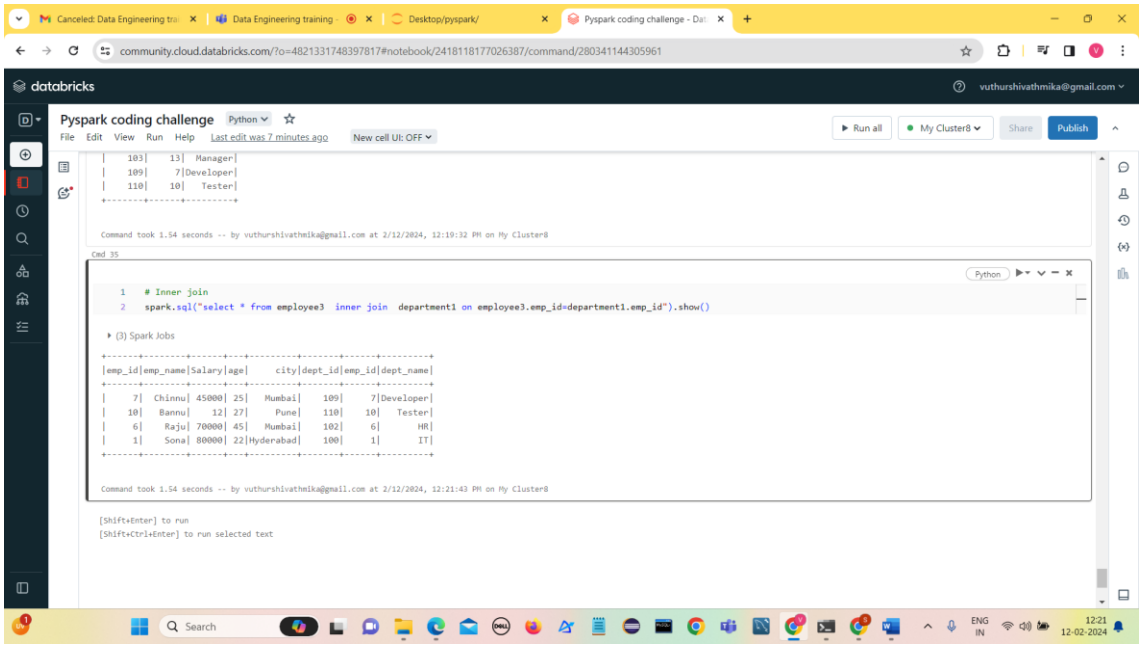


Create another table for Department1 using spark.sql command



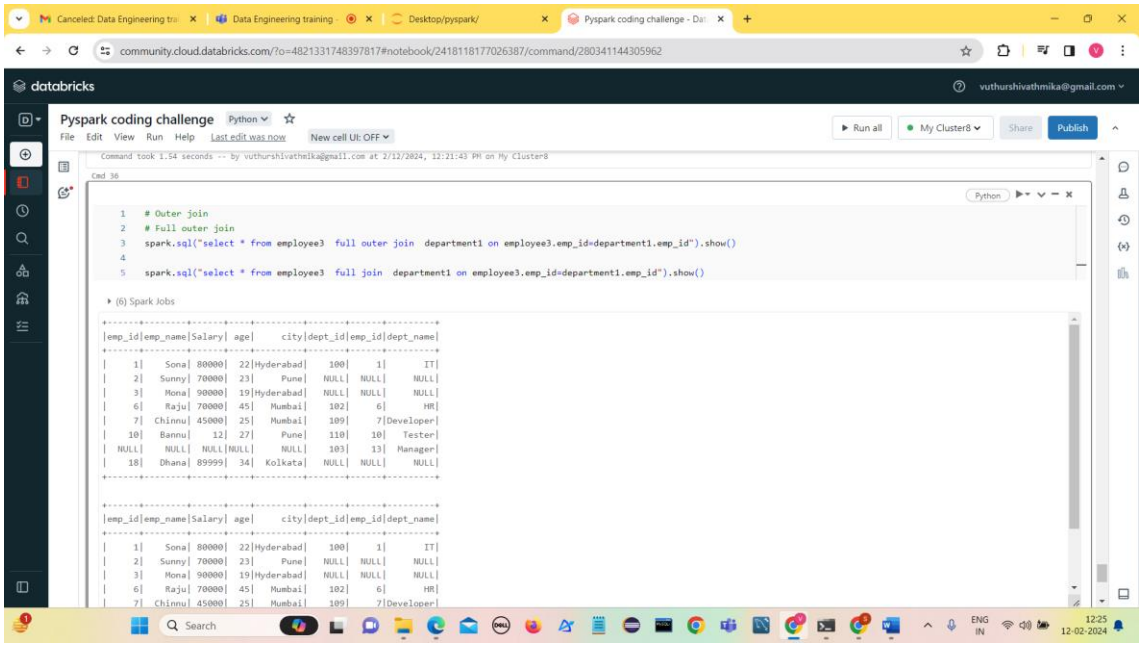
1) Inner Join

Join records when key values are matched, and dropped when they are not matched.



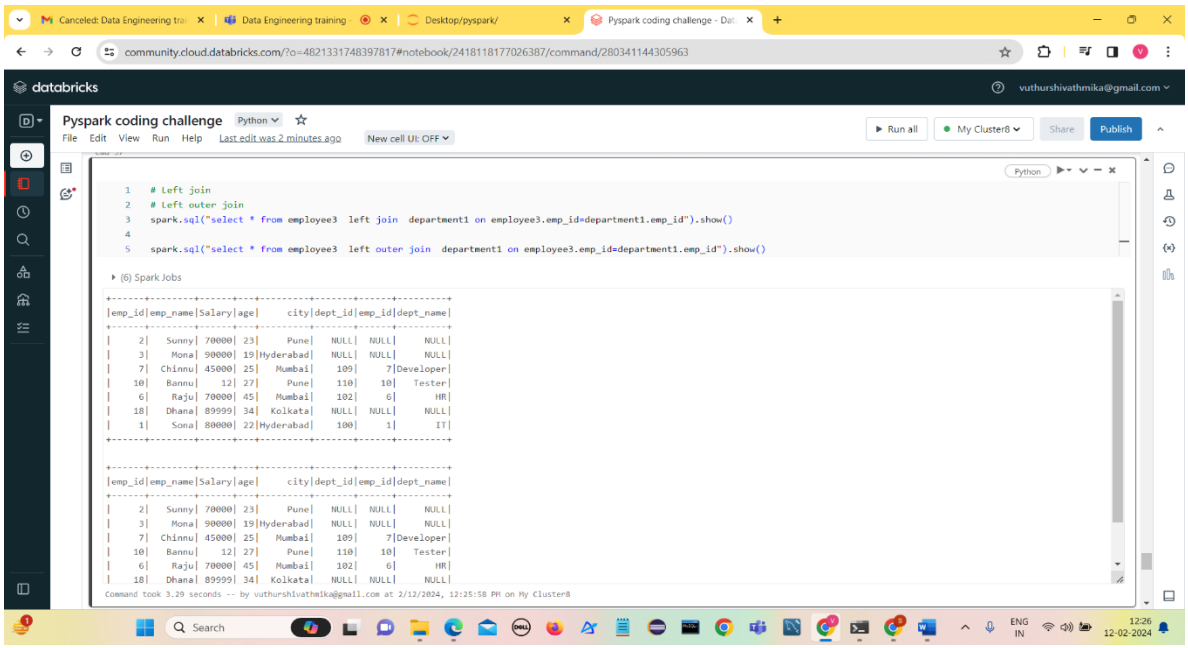
2) Outer join

Returns all rows from both tables, if join expression doesn't match it returns null values. (right join + left join)



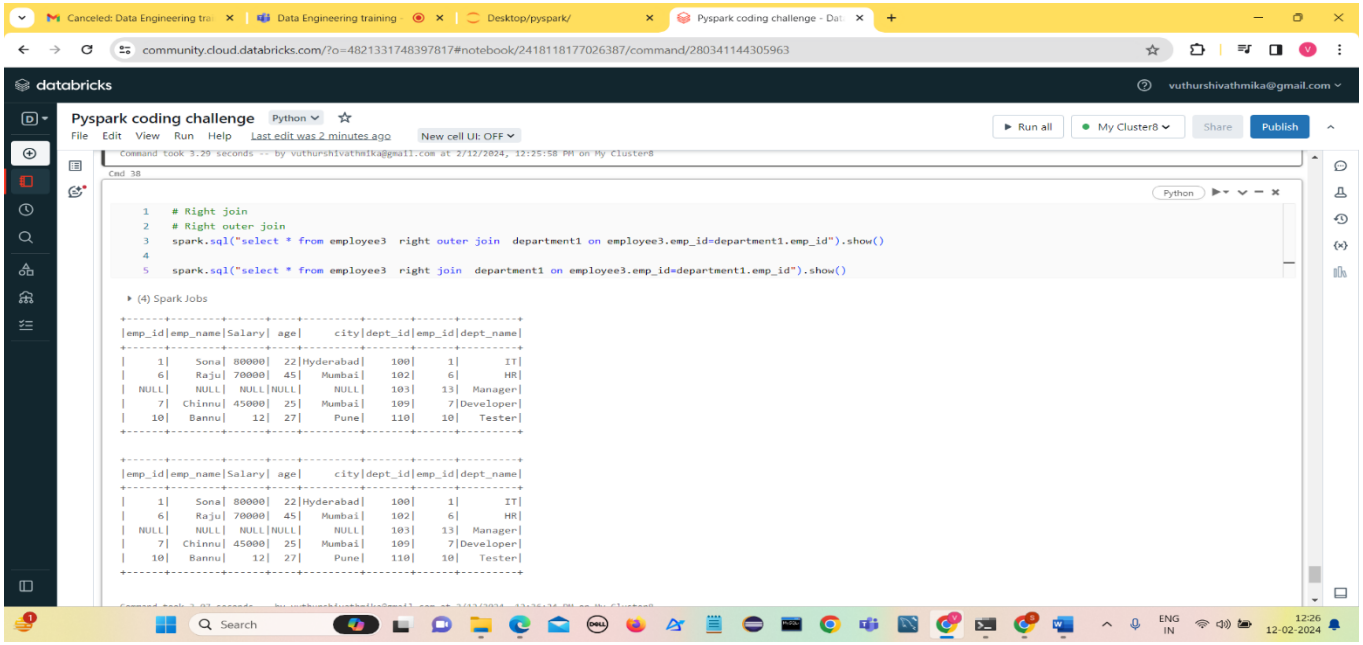
3) Left Join

Returns all rows from the left table irrespective of match found on the right table ,when join doesn't match it assigns null for that record



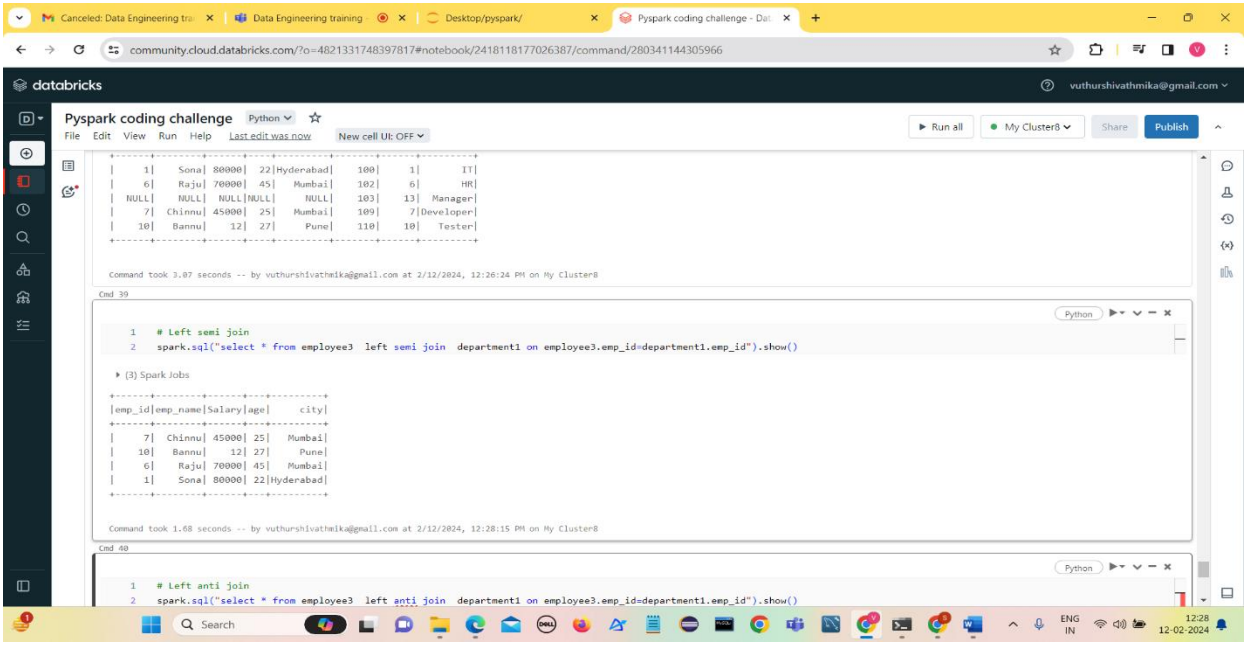
4) Right Join

Returns all rows from the right table irrespective of match found on the left table ,when join doesn't match it assigns null for that record



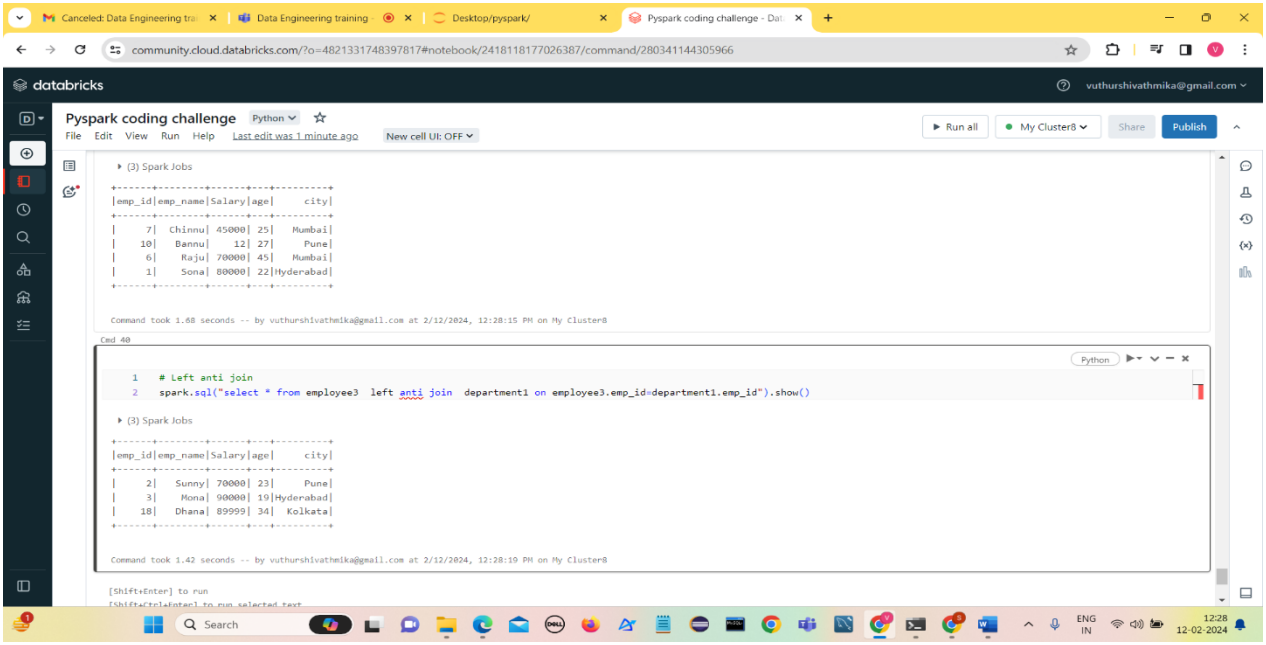
5) Left semi join

returns columns from the only left table for the records match in the right table on join expression, records not matched on join expression are ignored from both left and right tables.



6) Left anti join

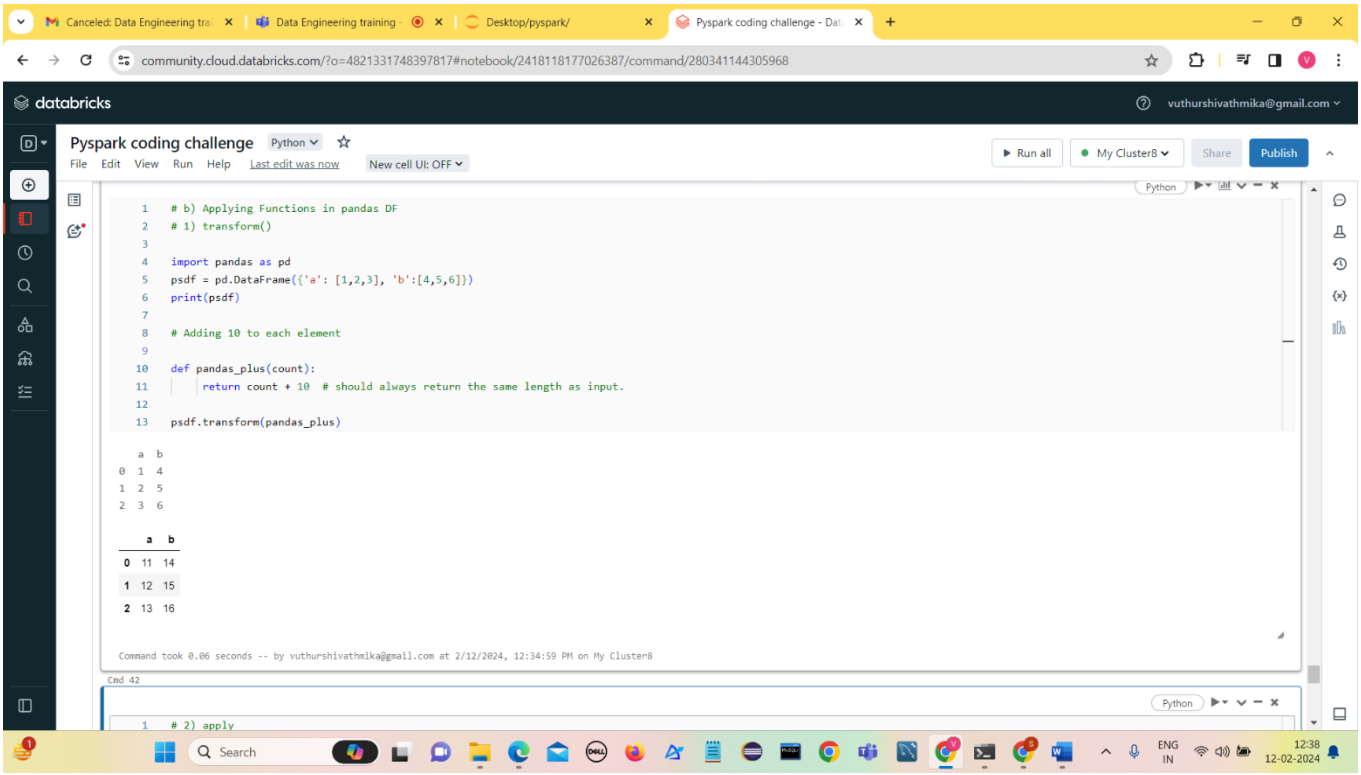
leftanti join returns only columns from the left table for non-matched record



b) Applying functions in pandas dataframe

1) Transform():

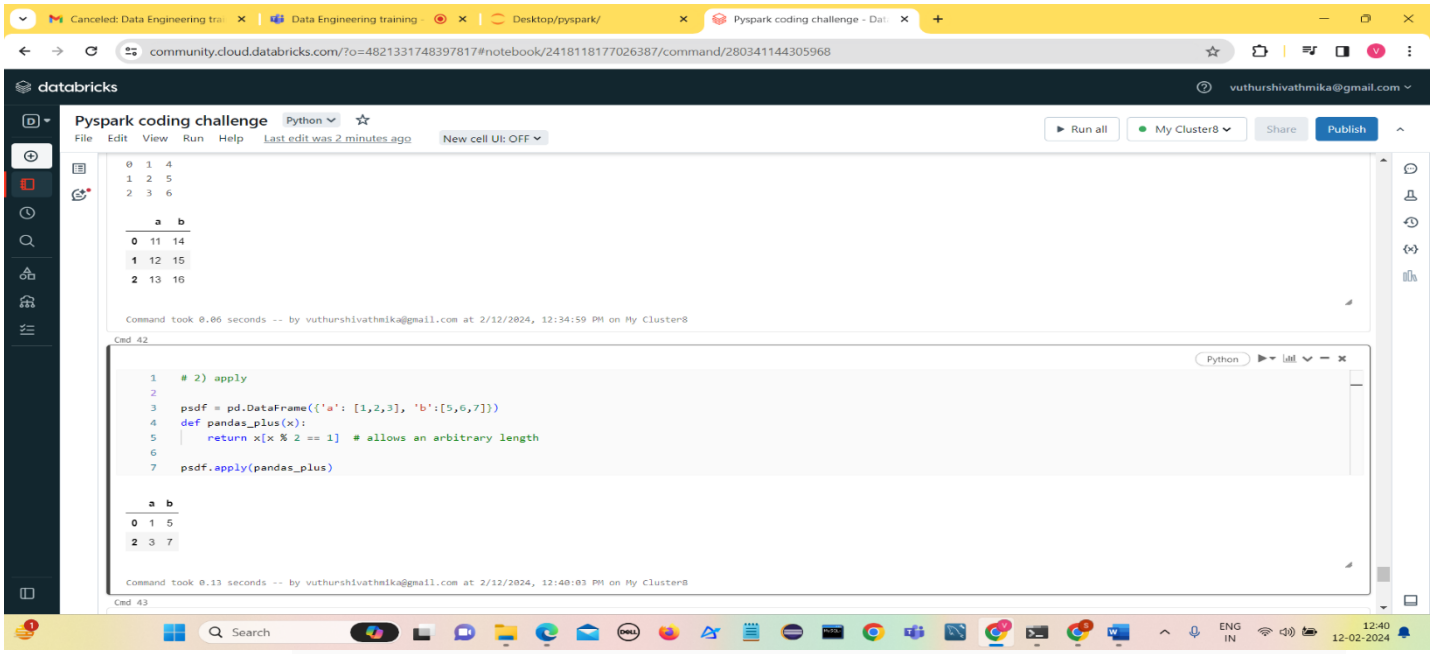
- Each function takes a pandas Series, and the pandas API on Spark computes the functions in a distributed manner
- It should return output as same length of input.



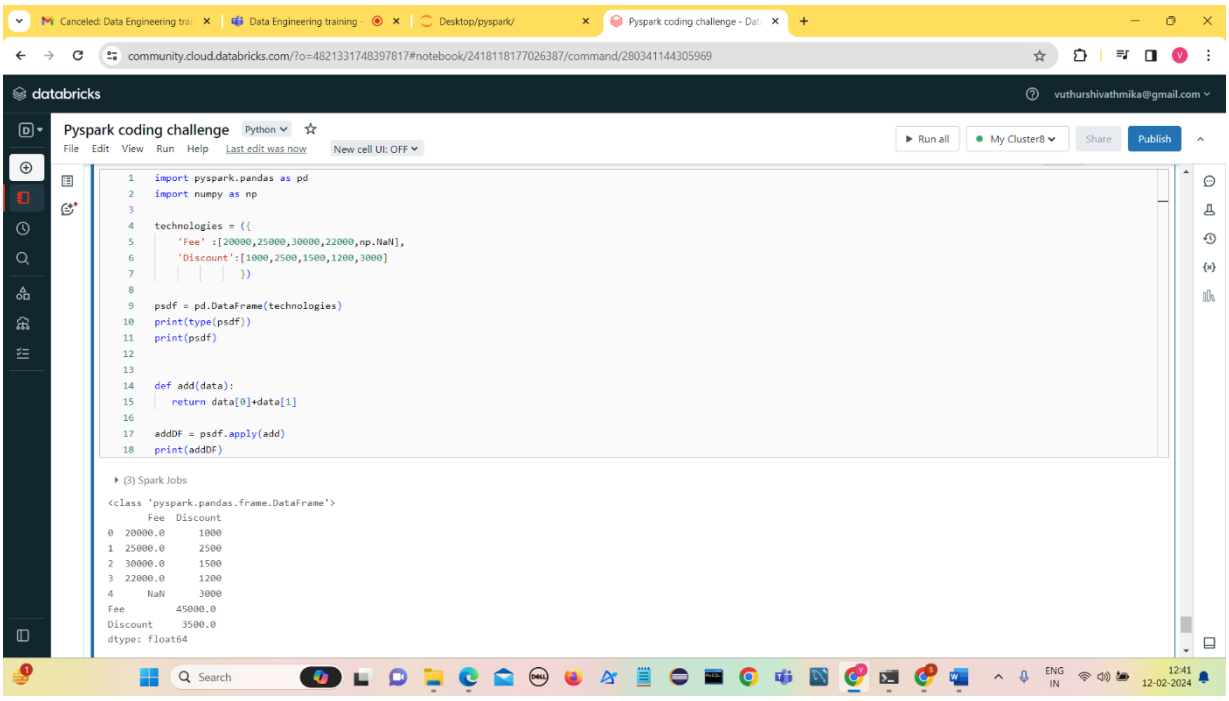
2) apply()

- Each function takes a pandas Series, and the pandas API on Spark computes the functions in a distributed manner

- It is not mandatory to return same length as output



Example:



Vuthur Sriganga

Data Engineering Batch-1

vuthursriganga@gmail.com