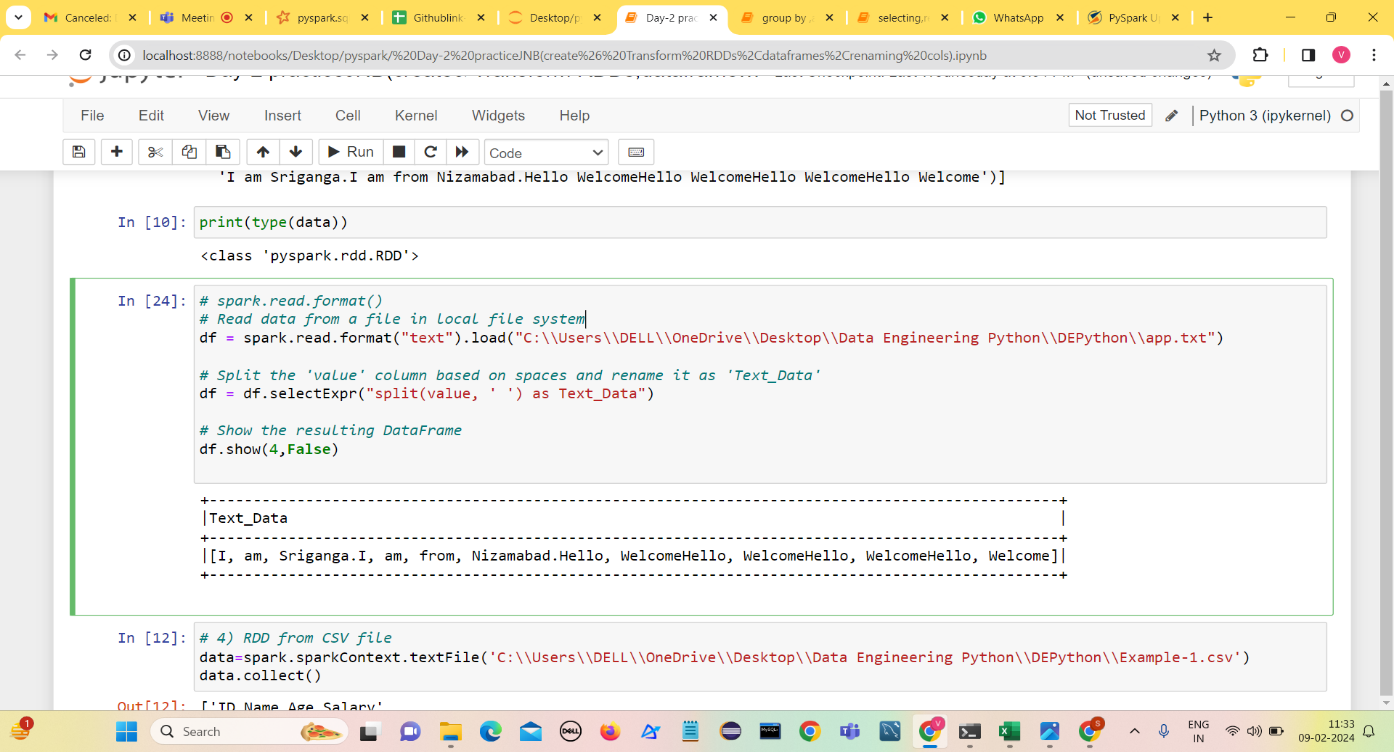
**Pyspark Day-4**

**Creating dataframes**

**spark.read.format():**

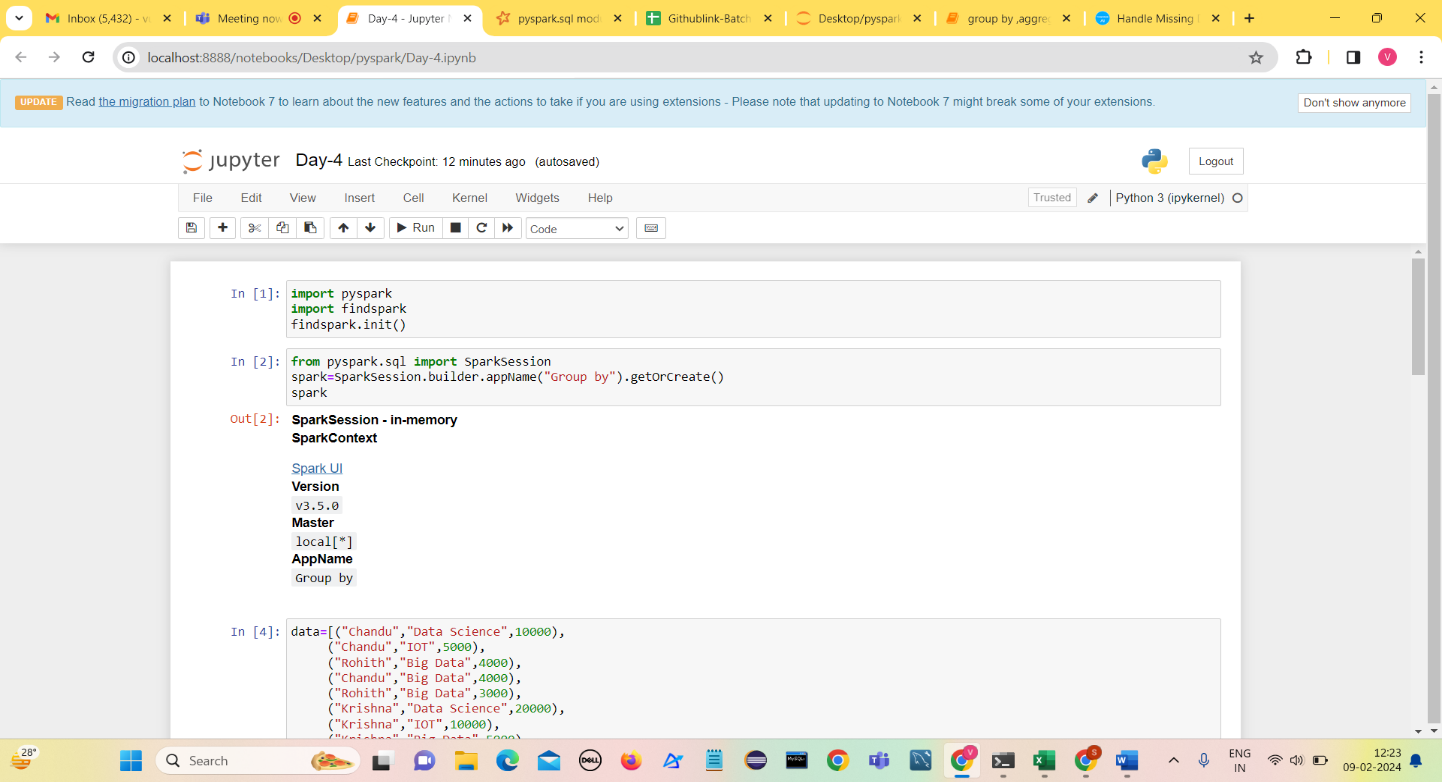
spark.read.format(“text”).load(“output.txt”)

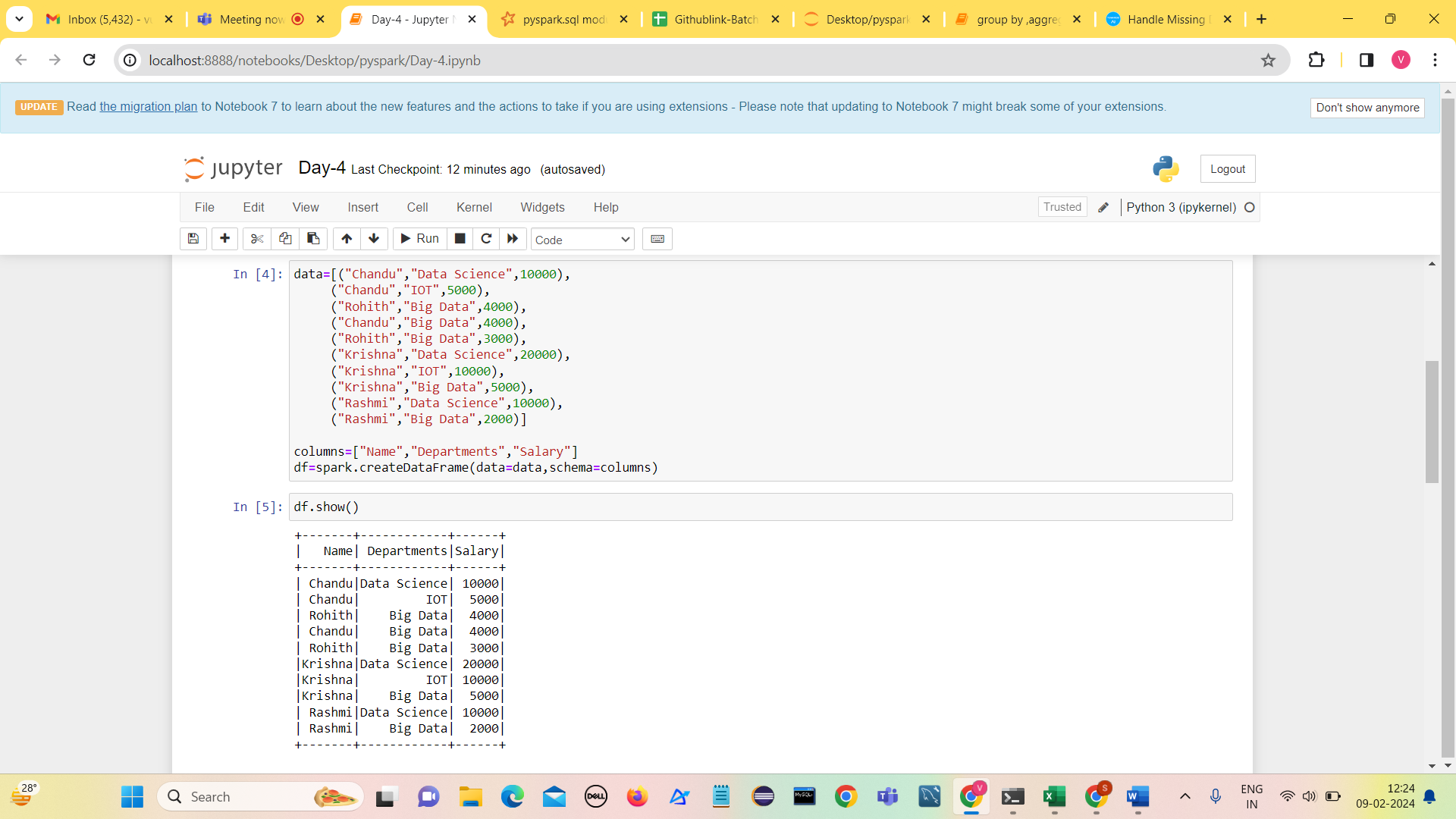
**Group By and Aggregations**

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.

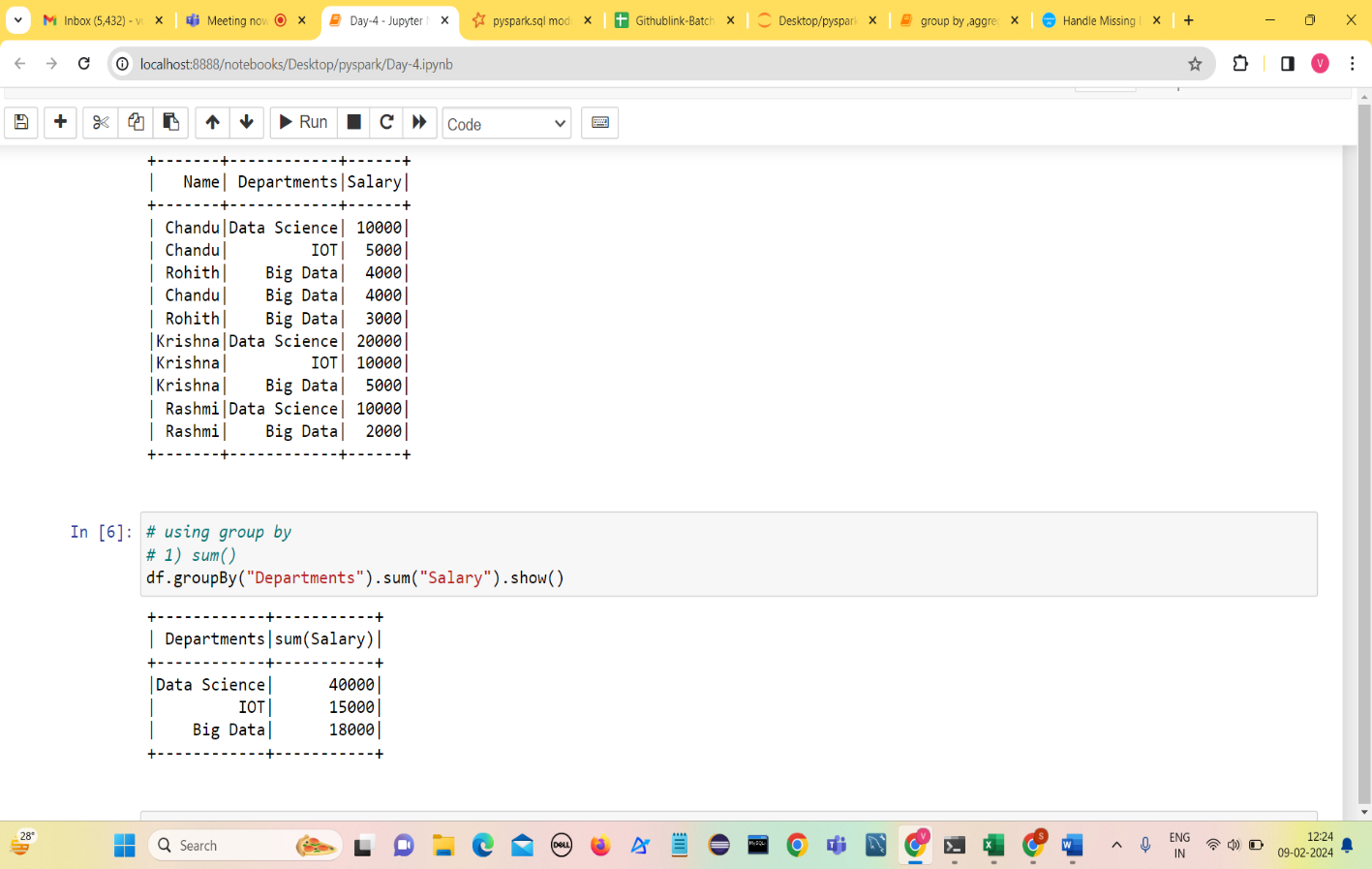
Create a dataframe with data and columns to perform groupby and aggregation functions

.

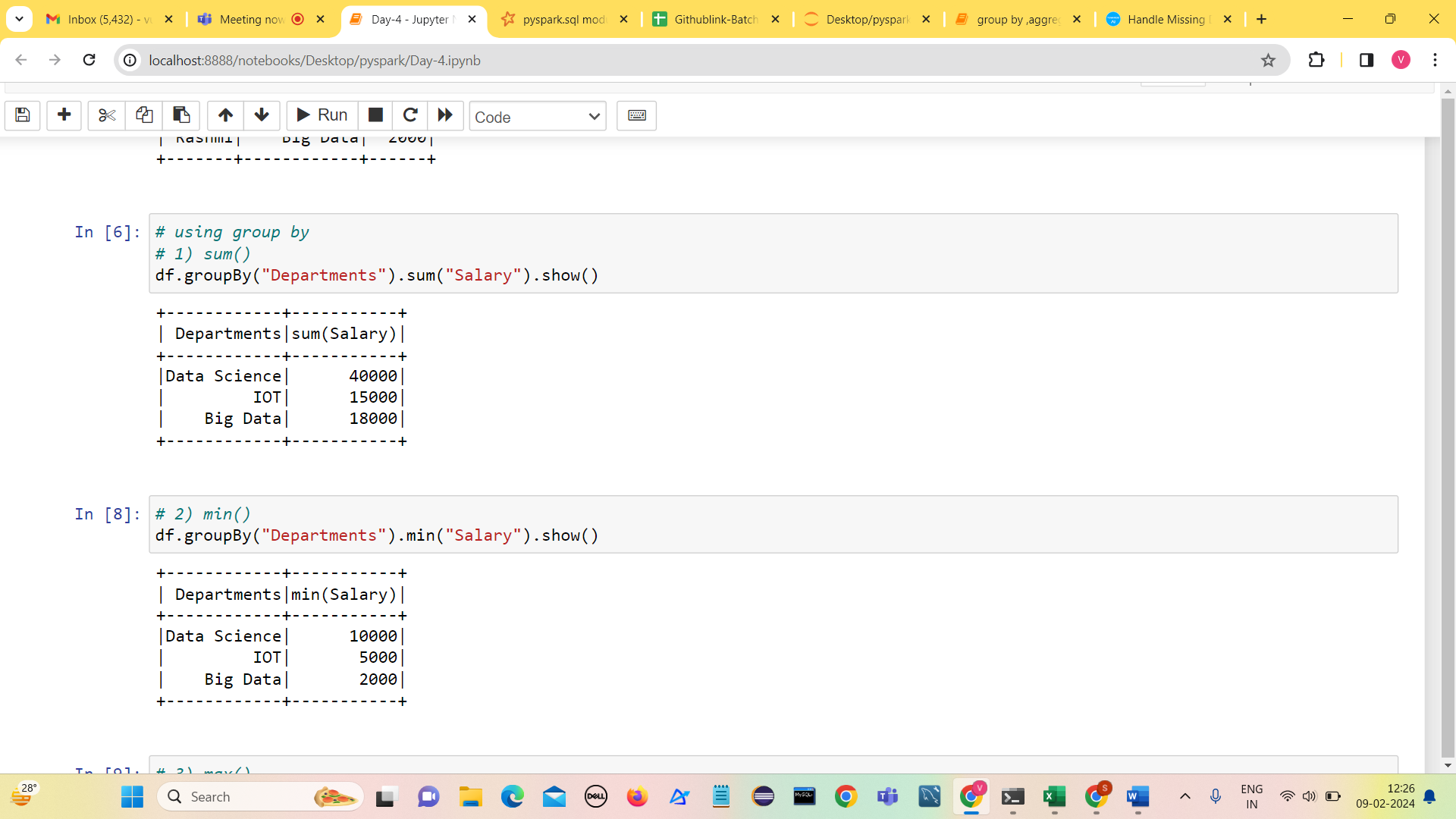


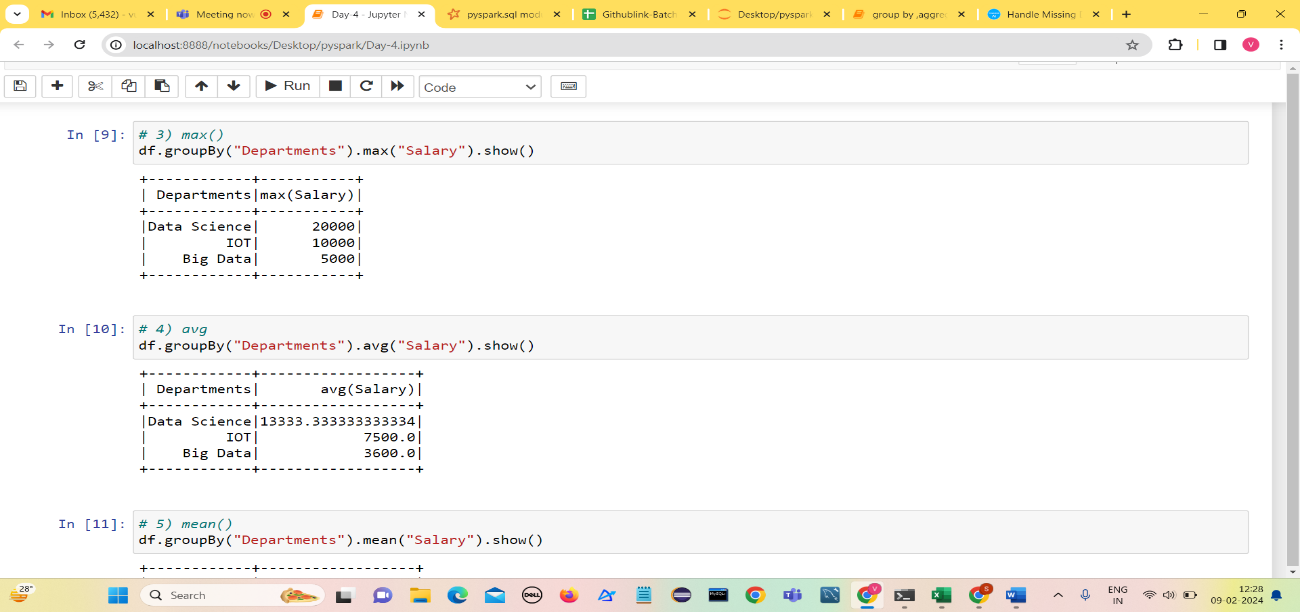
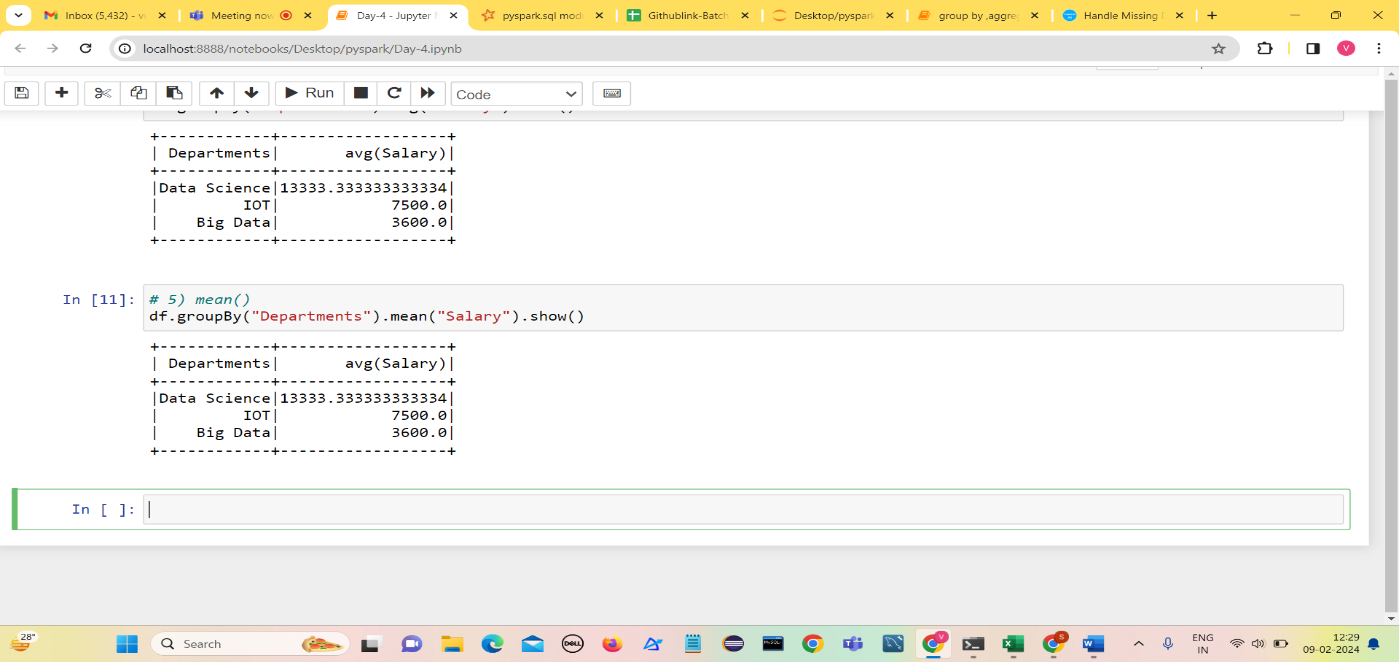


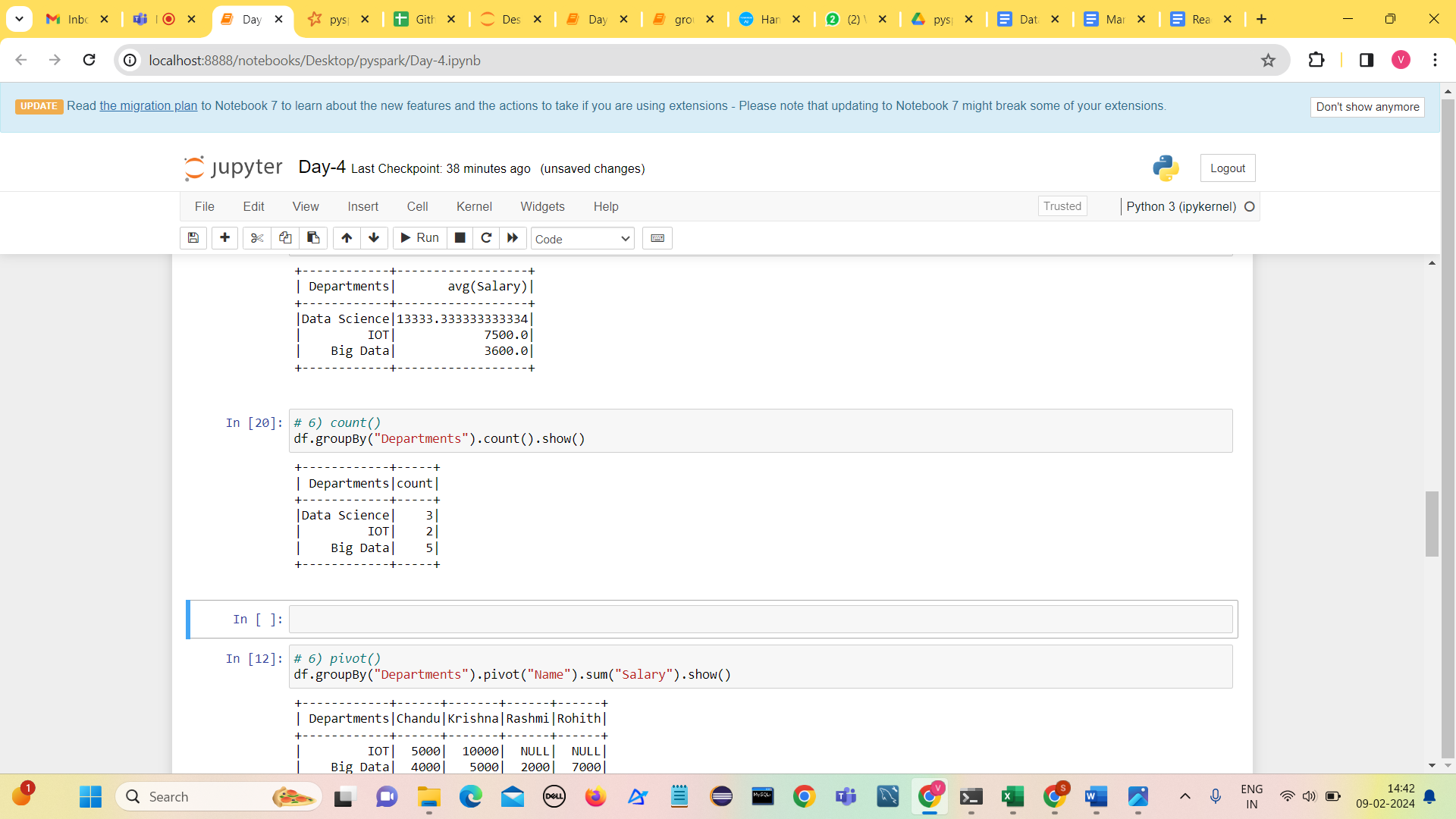
1. **using groupBy to find sum of salary**

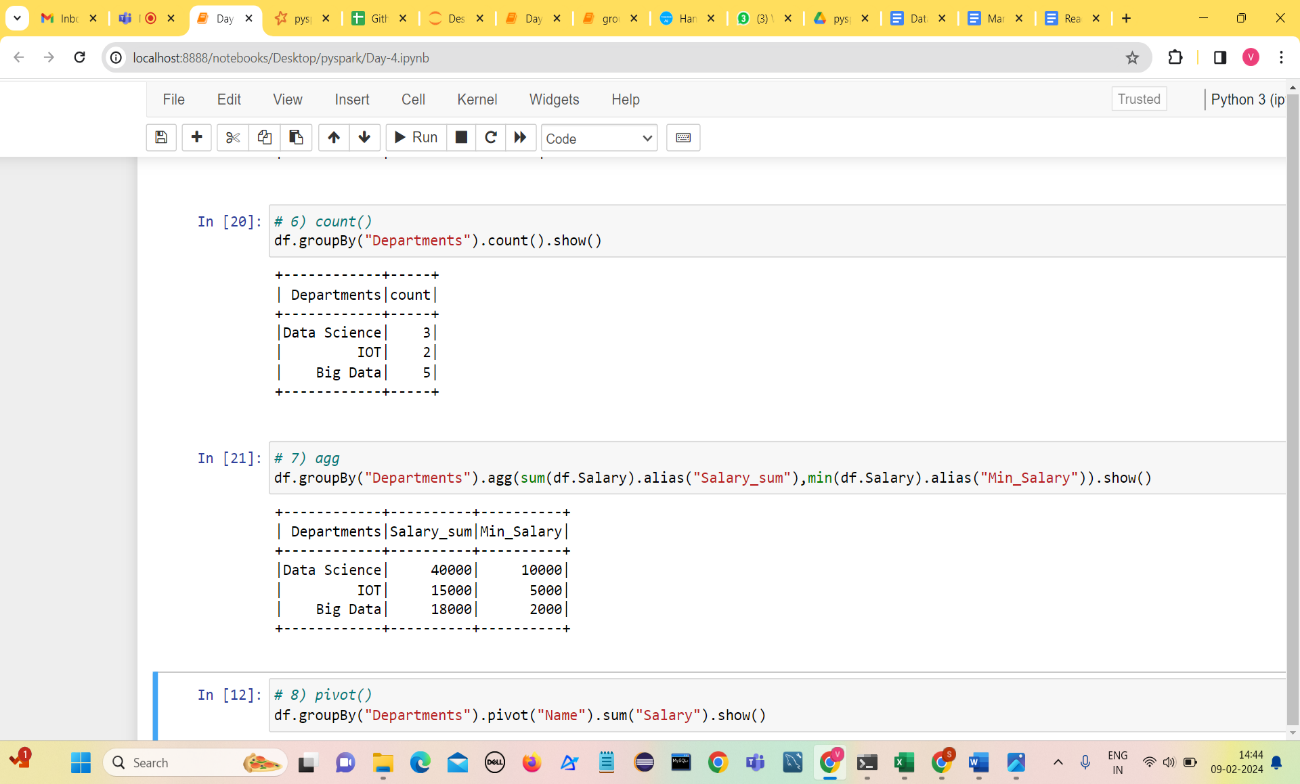


1. **min()**

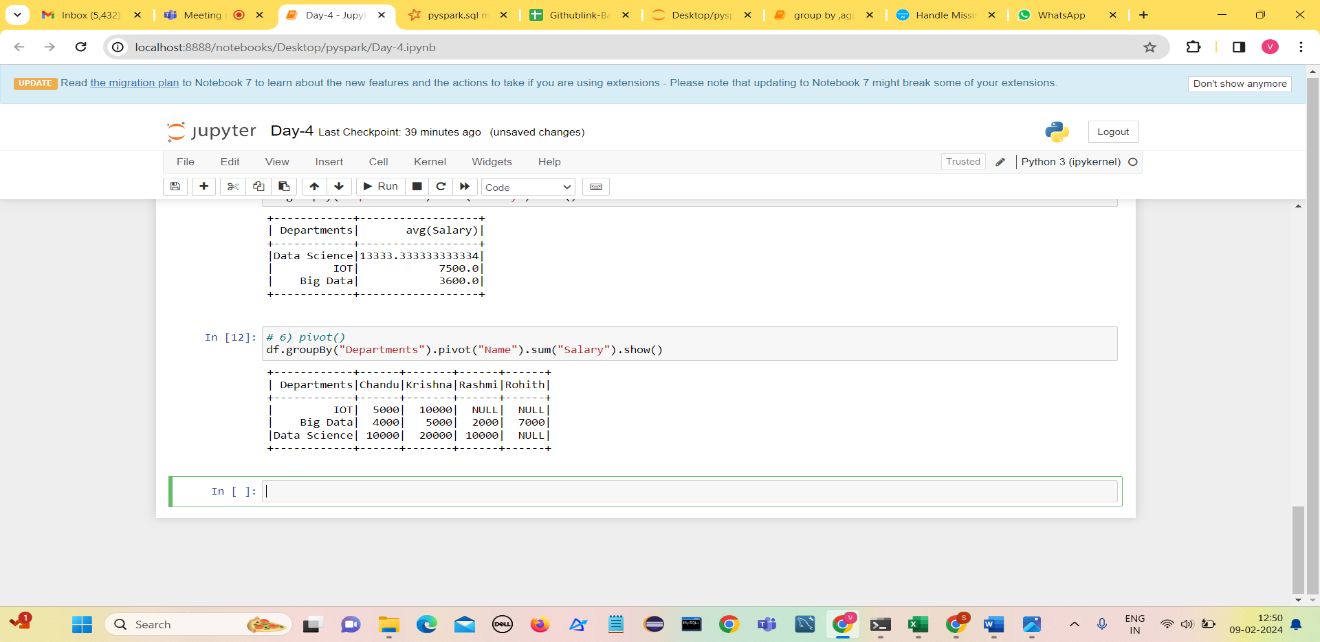


1. **max()**
2. **avg ()**
3. **mean()**
4. **Count()**



1. **agg()**
2. **Pivot()**

It is an aggregation where one of the grouping column values is transposed into individual columns with distinct data.



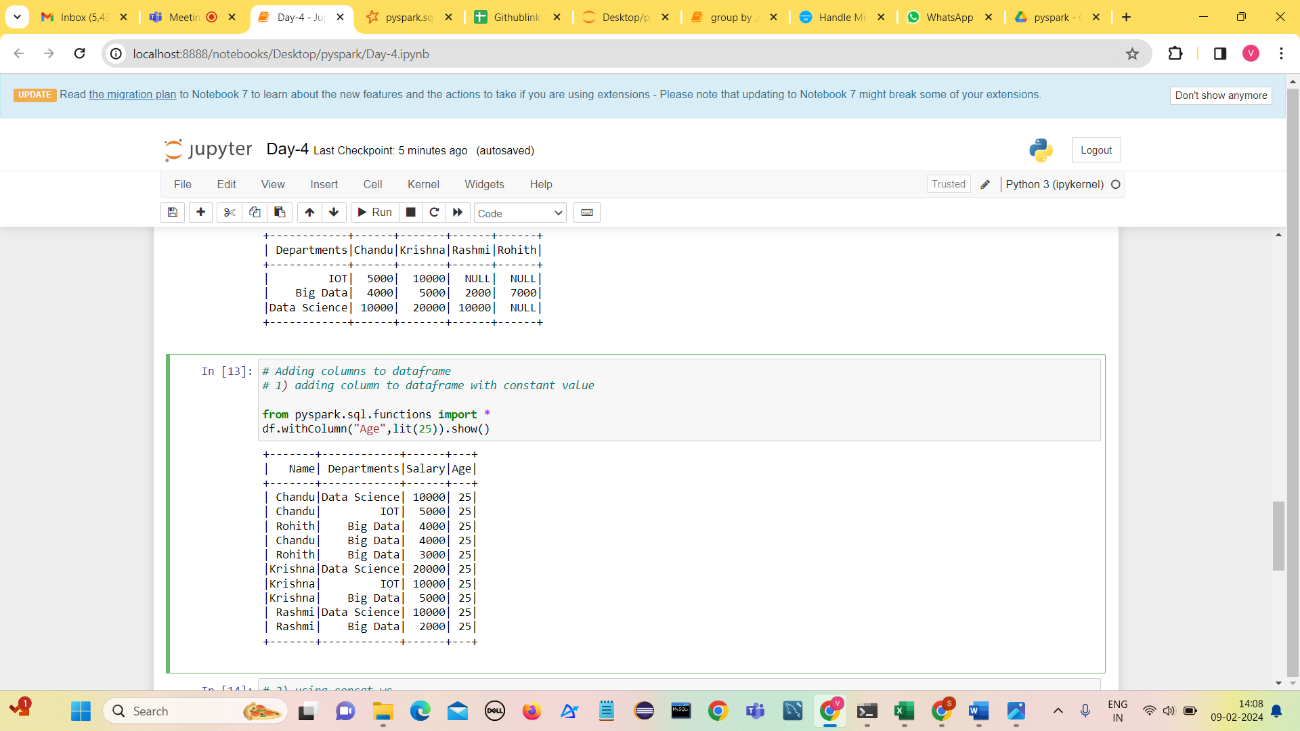
**Adding columns to dataframe**

**Method 1: Add New Column With Constant Value**

In this approach to add a new column with constant values, call the lit() function parameter of the withColumn() function and pass the required parameters into these functions.

Here, the lit() is available in pyspark.sql. Functions module.

**Syntax**:

**dataframe.withColumn("column\_name", lit(value))**

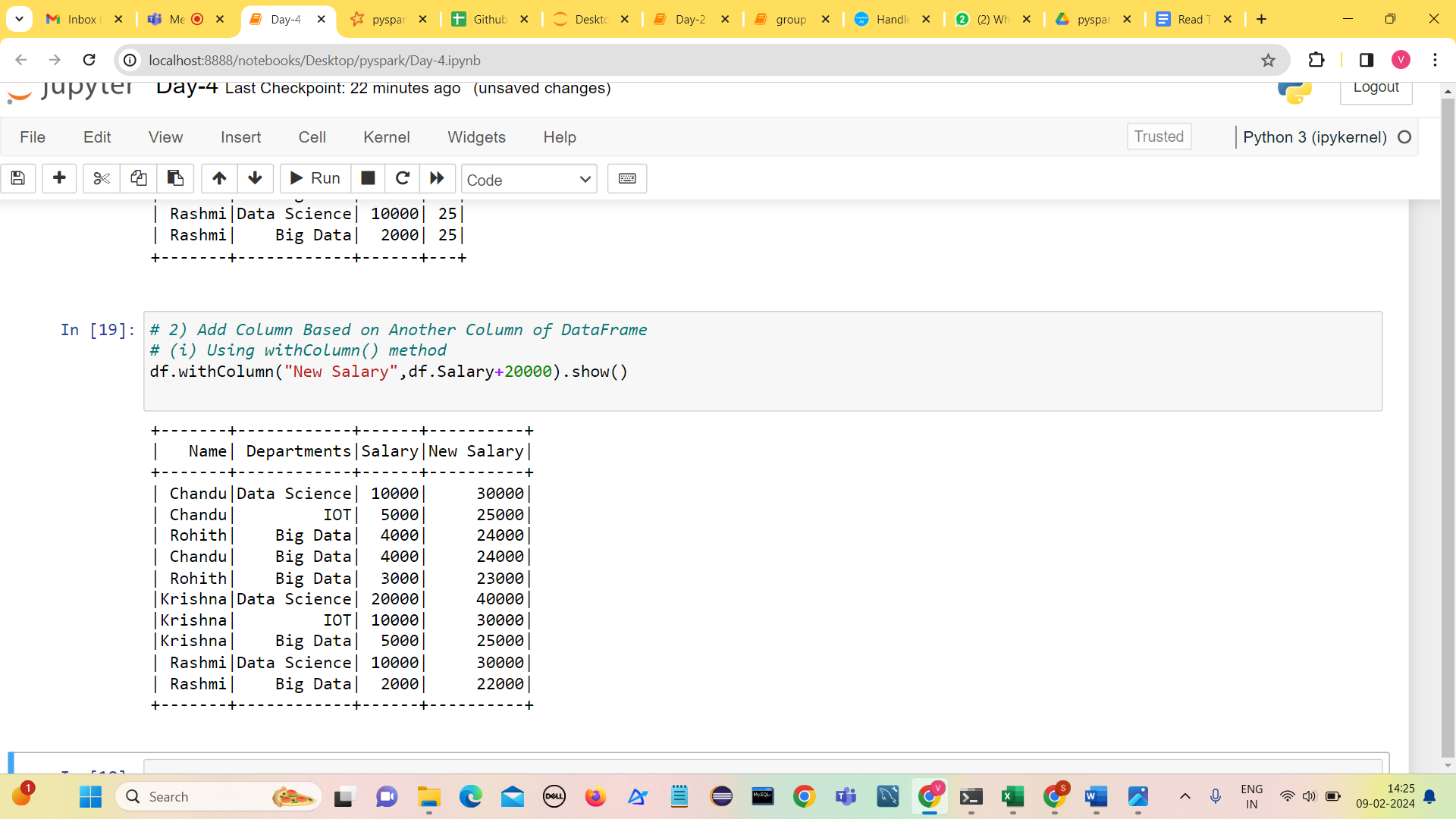
**Method 2: Add Column Based on Another Column of DataFrame**

Under this approach, we can add a new column based on an existing column in the given dataframe.

1. **Using withColumn() method**

**Syntax**:

**dataframe.withColumn("column\_name", dataframe.existing\_column)**

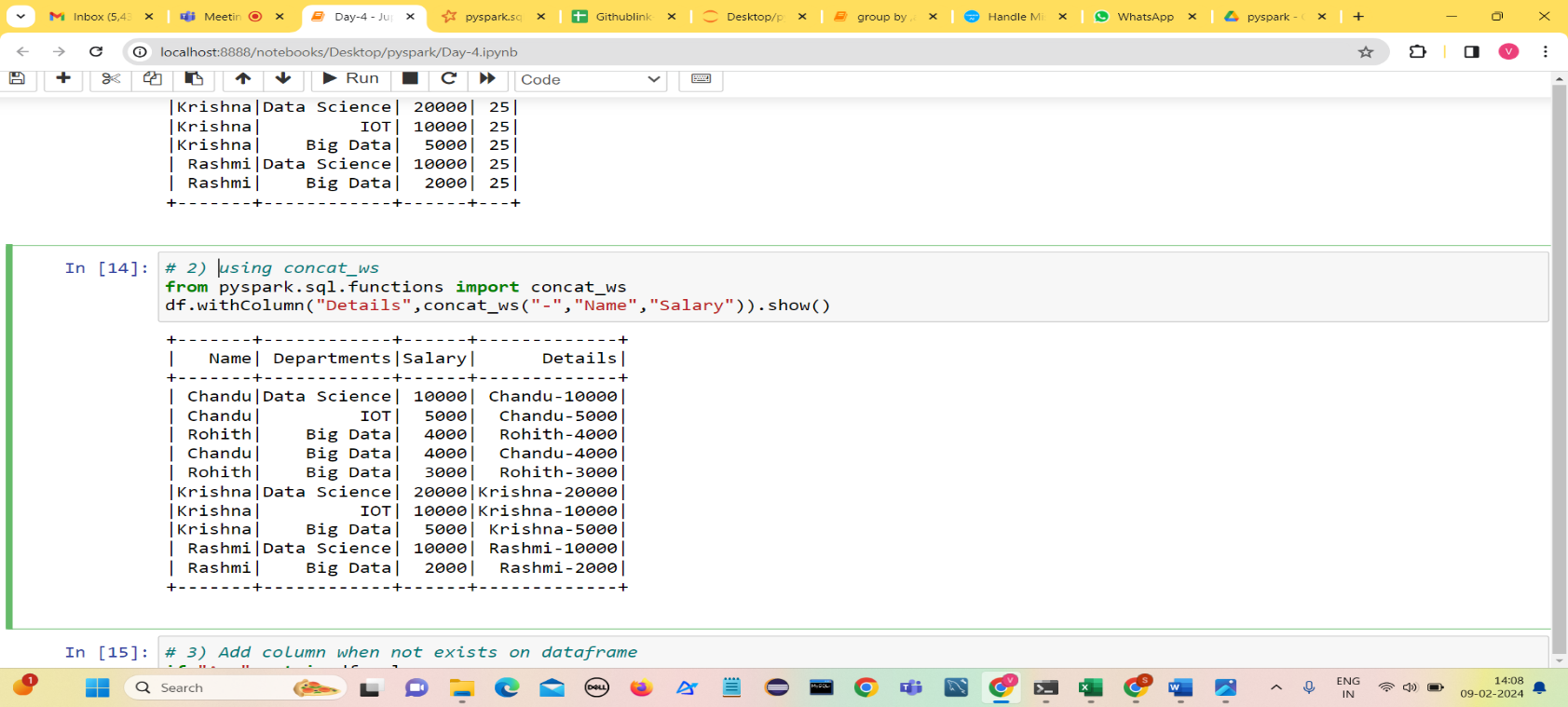


1. **using concat\_ws:**

concat the two existing columns and make them as a new column by importing this method from pyspark.sql.functions module.

**Syntax**:

**dataframe.withColumn(“column\_name”, concat\_ws(“Separator”,”existing\_column1″,’existing\_column2′))**

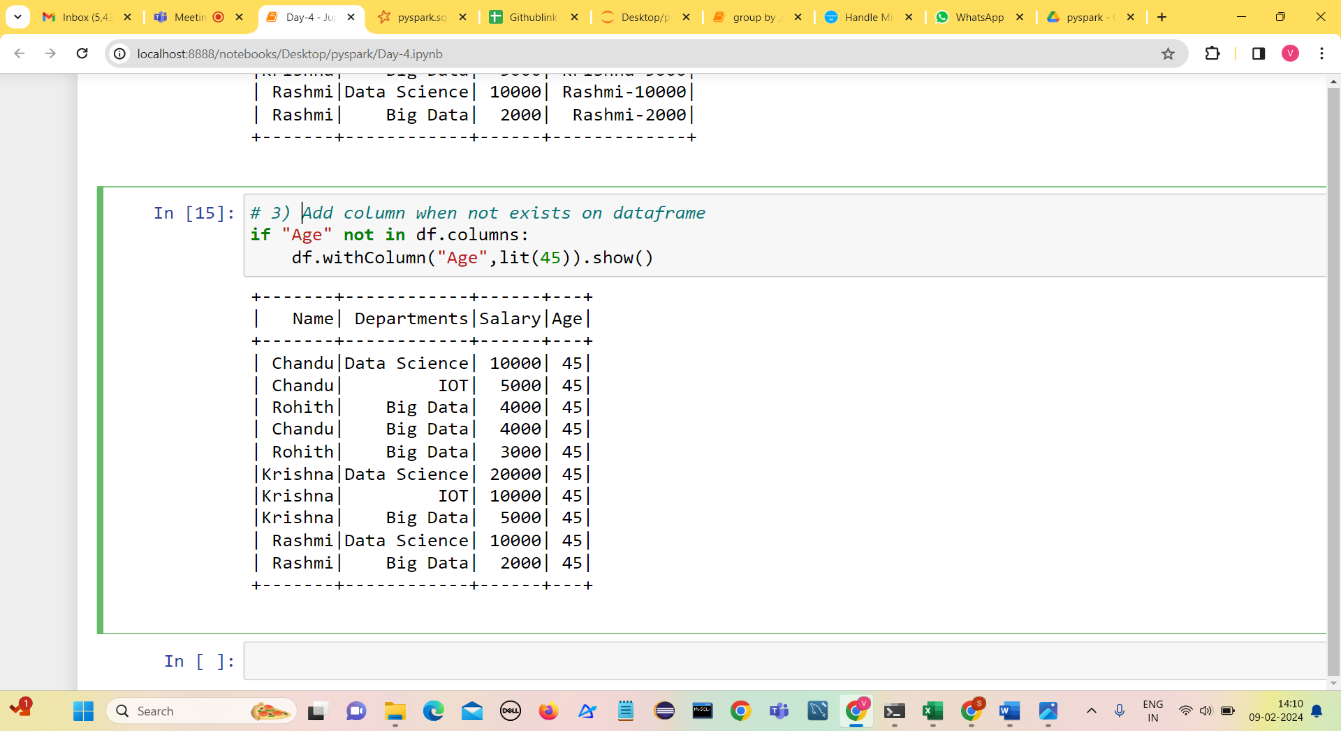


**Method-3 :Add column when not exists on dataframe**

In this method, we can add a column when it is not existed ,by adding a column with the lit() function and checking using if the condition.

Syntax:

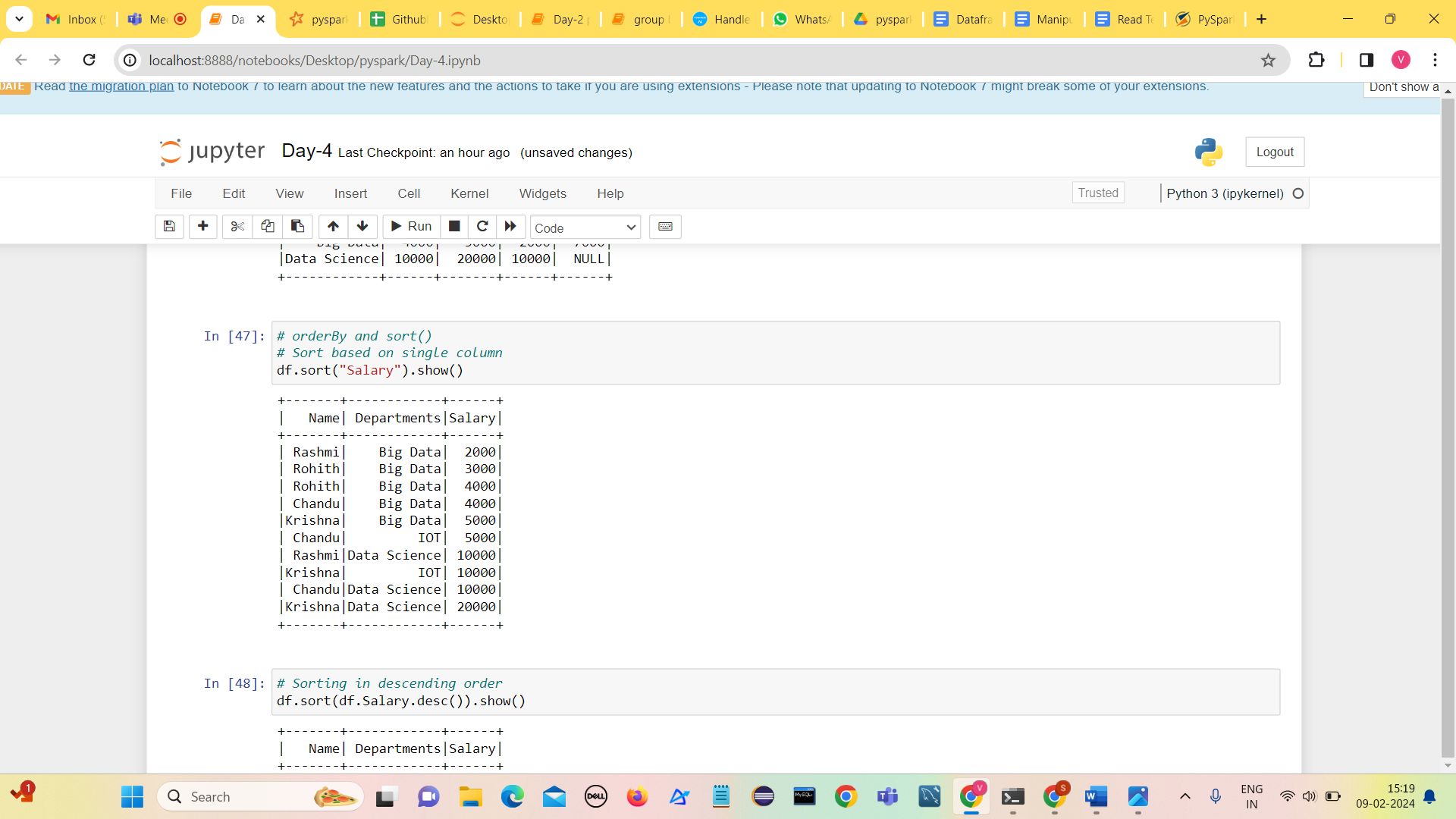
if 'column\_name' not in dataframe.columns:

   dataframe.withColumn("column\_name",lit(value))

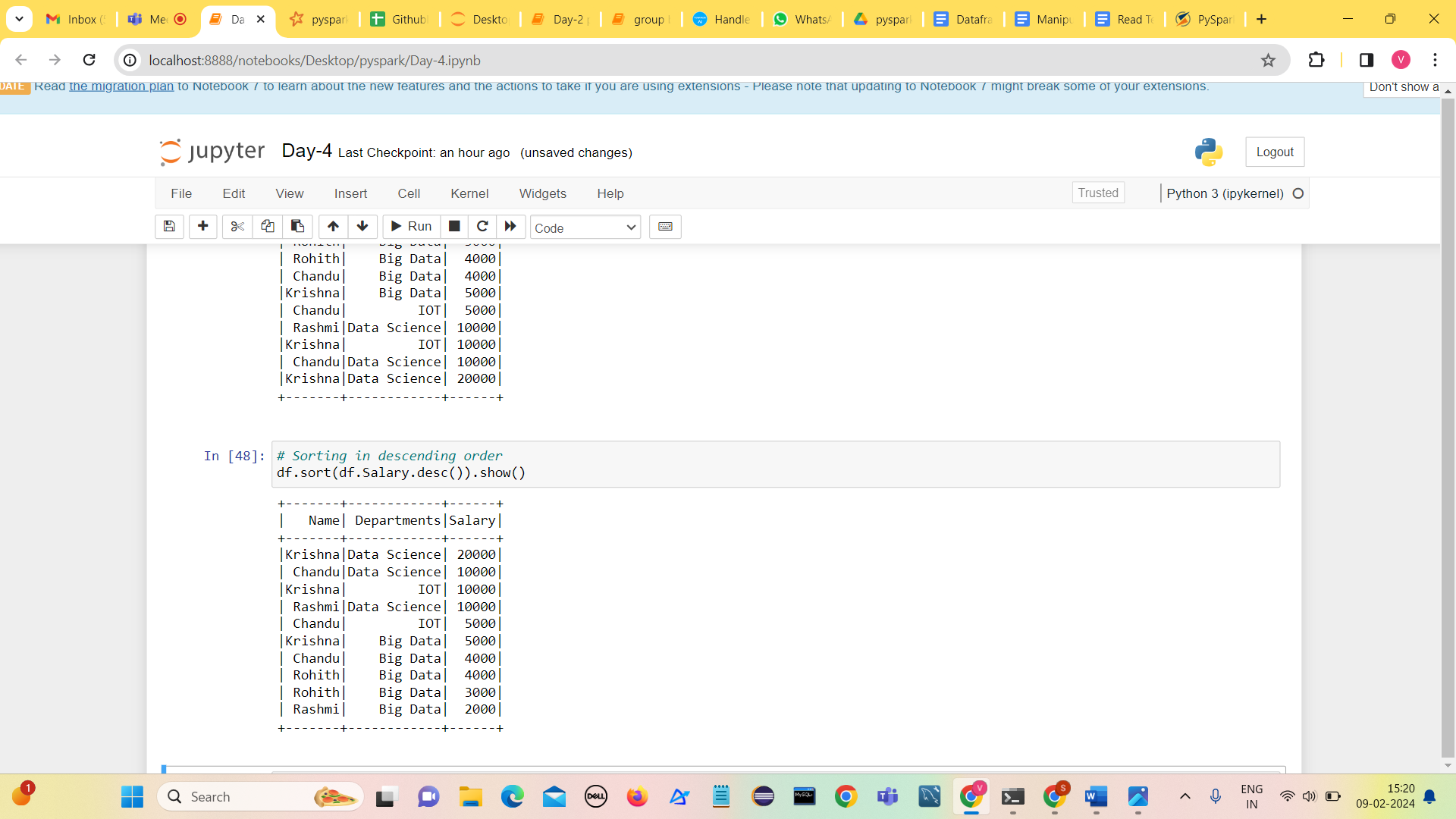
**orderBy() and sort() in Pyspark DataFrame**

**sort()** — To sort a dataframe by using one or more columns and by default sorts in ascending order

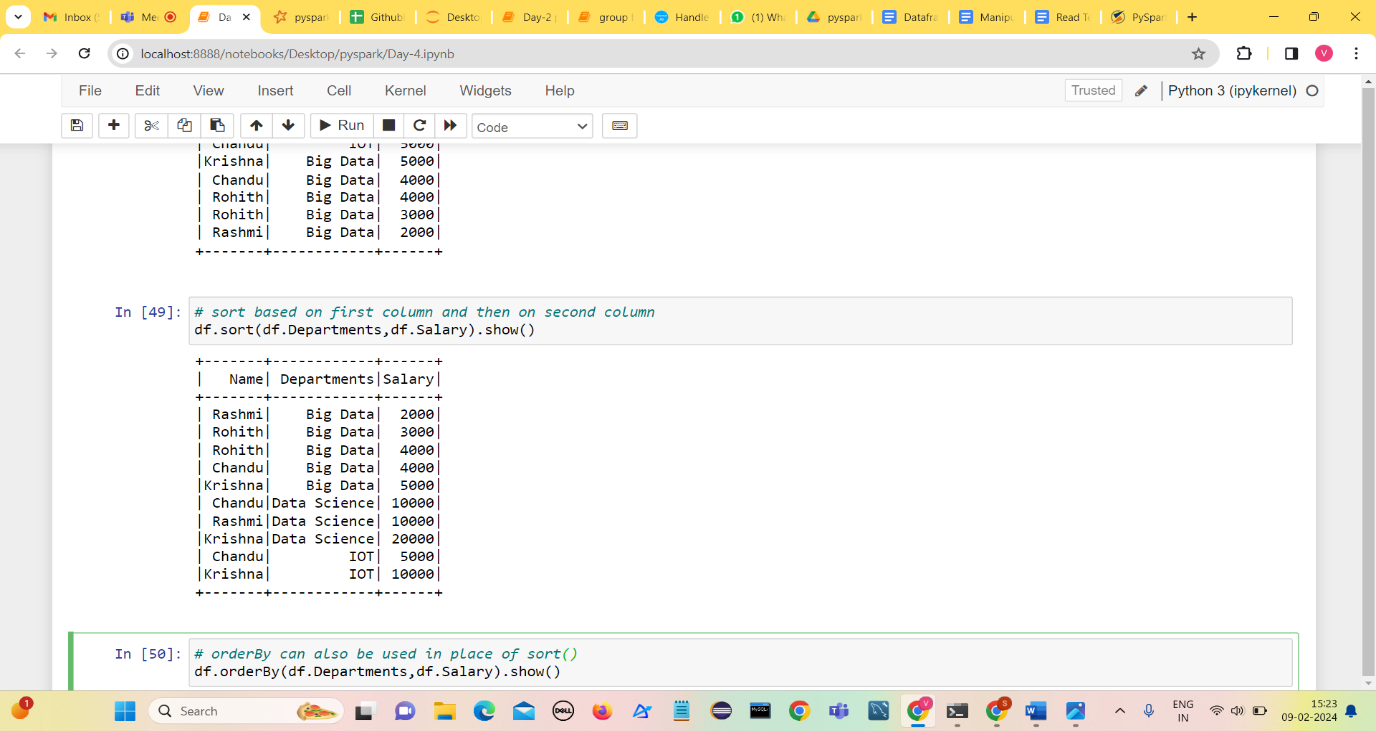
1. **Sort based on single column**

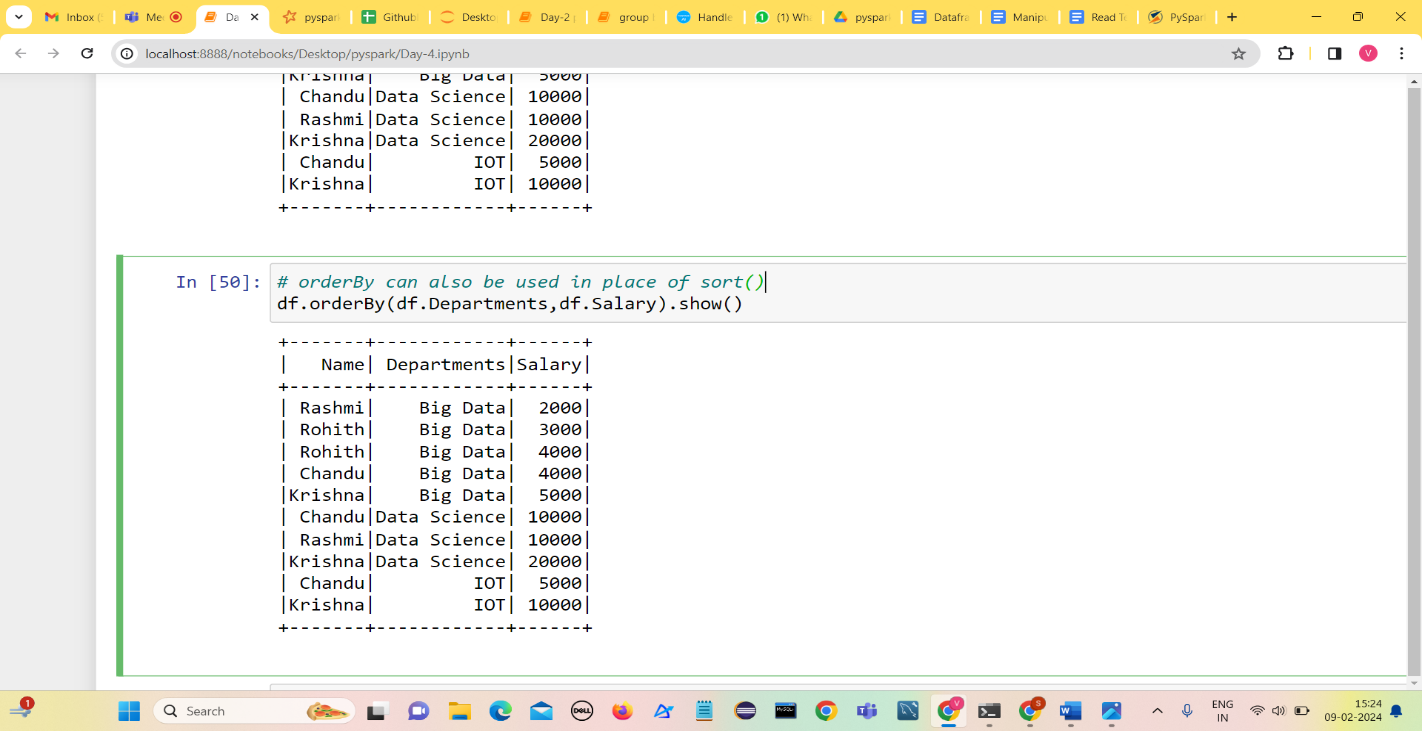


1. **Sorting in descending order using desc()**

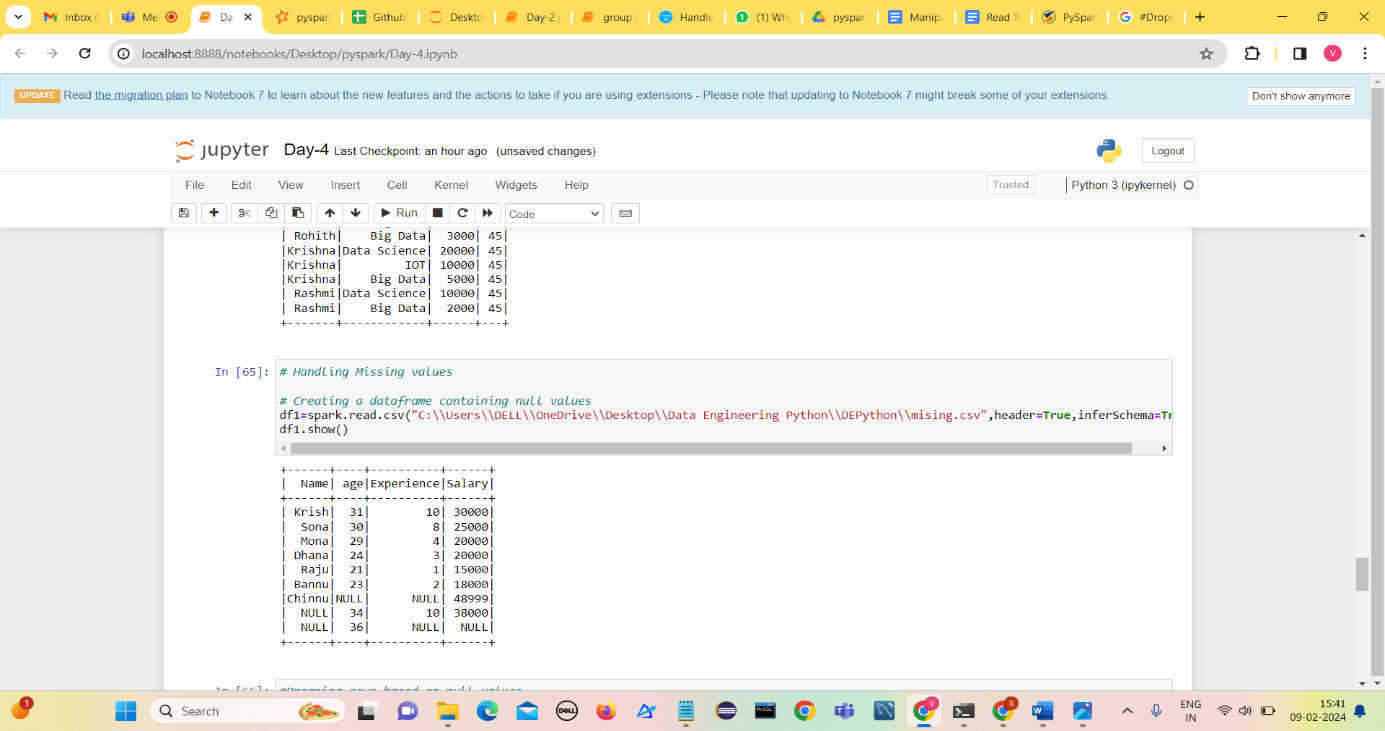


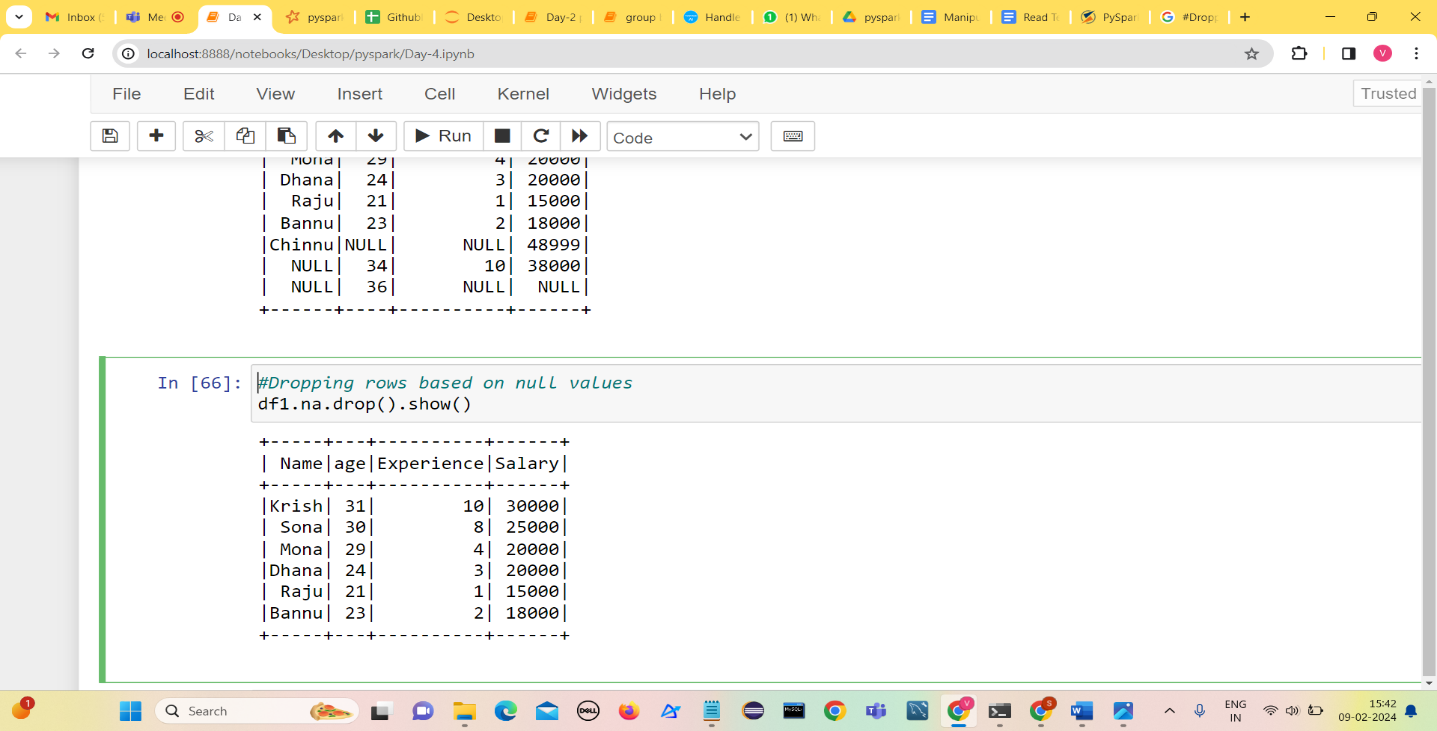
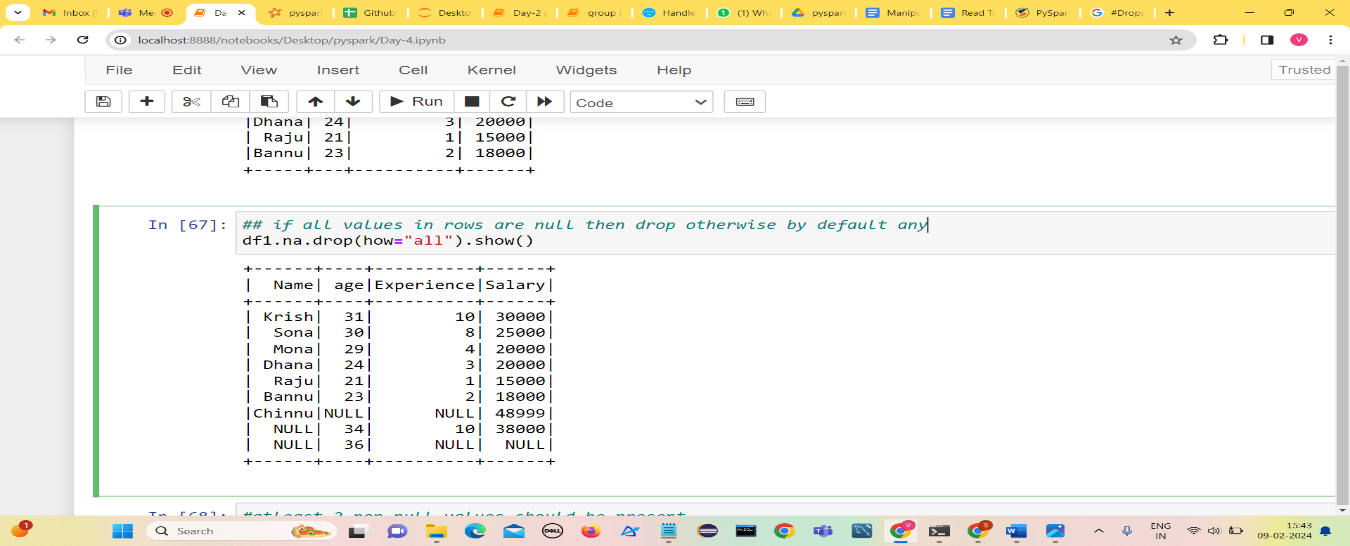
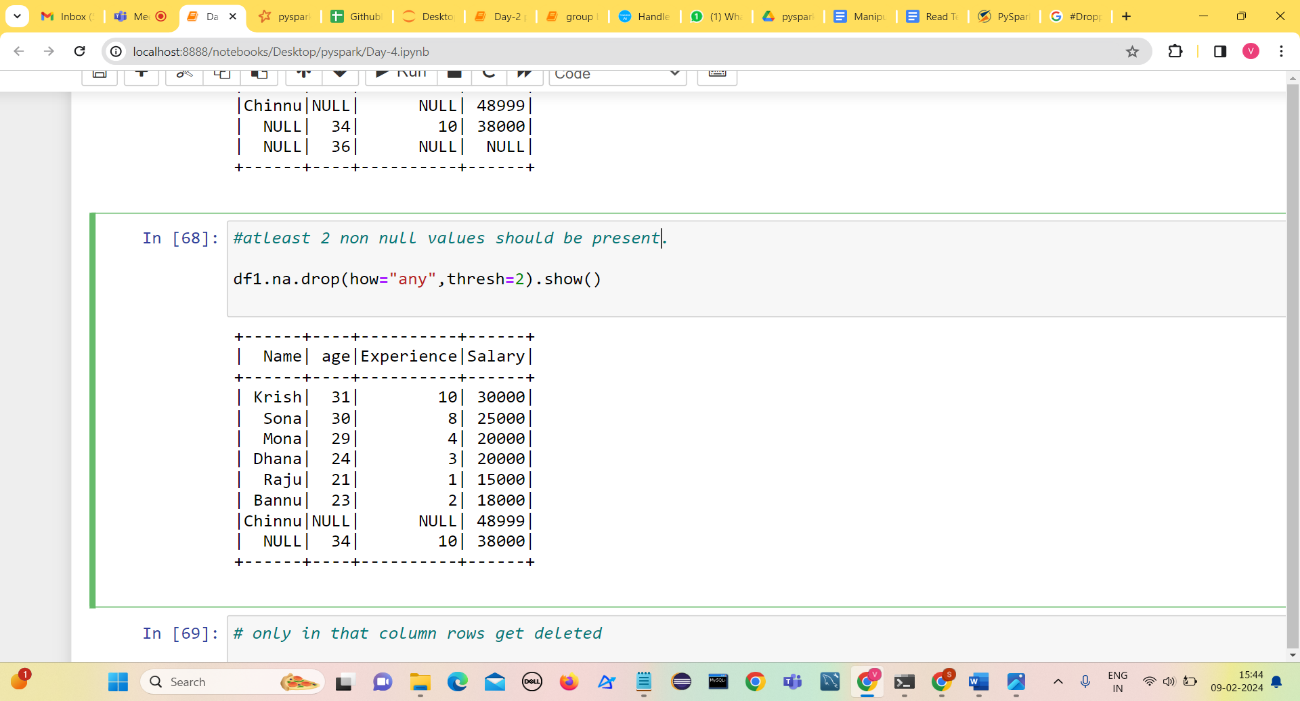
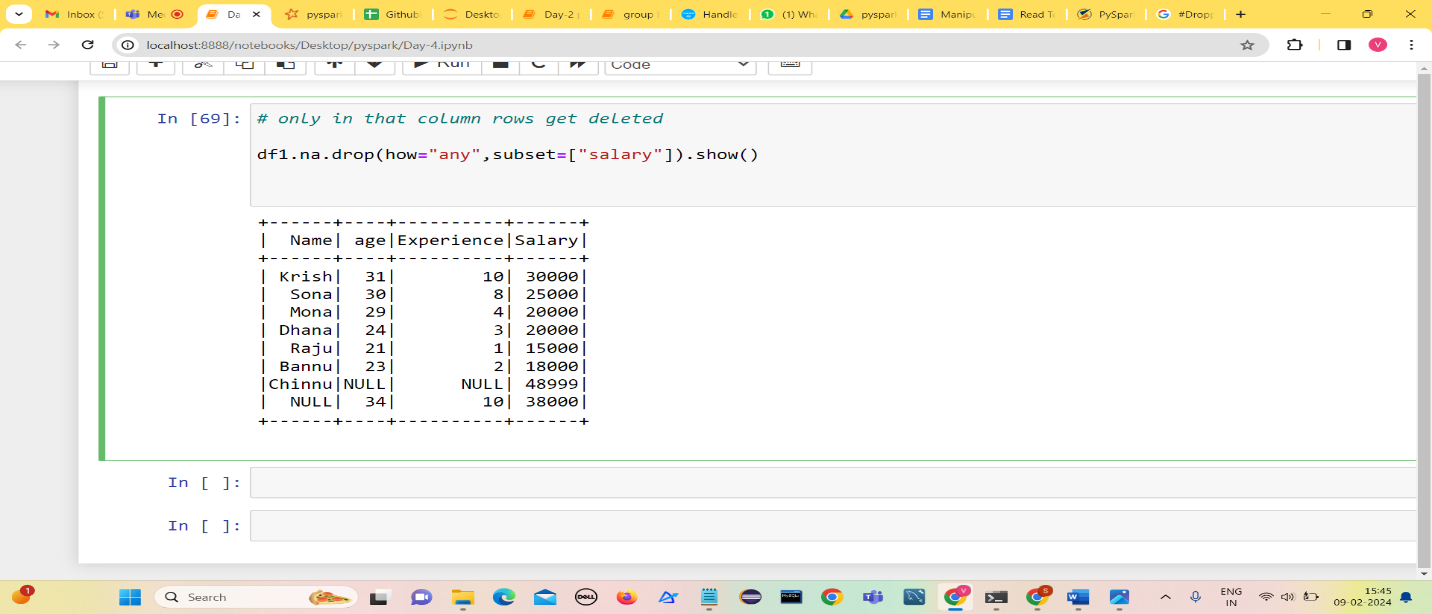
1. **Sort based on first column and then on second column**



1. **orderBy can also be used in place of sort()**

**Handling Missing Values Pyspark**

Create a dataframe containing null values

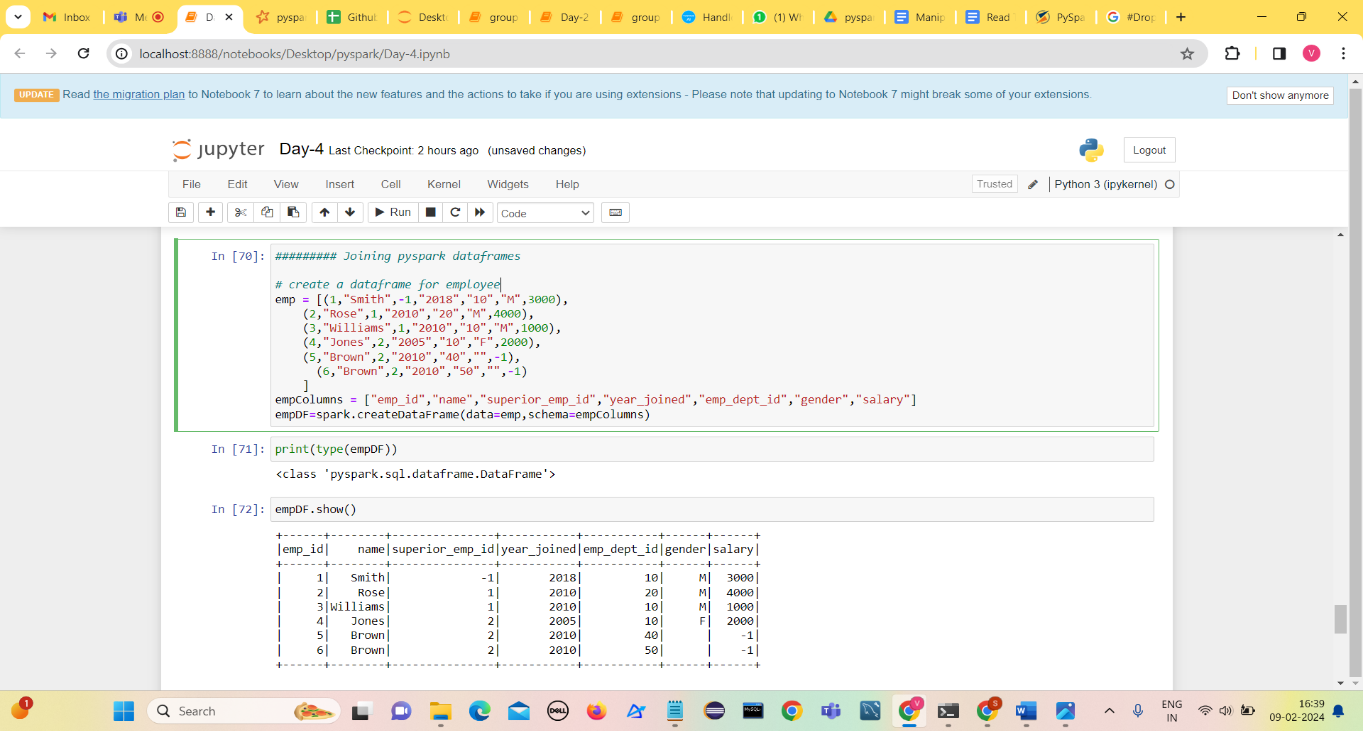
1. **Dropping rows based on null values**
2. **if all values in rows are null then drop otherwise by default any**
3.  **atleast 2 non null values should be present**
4. **only in that column rows get deleted**

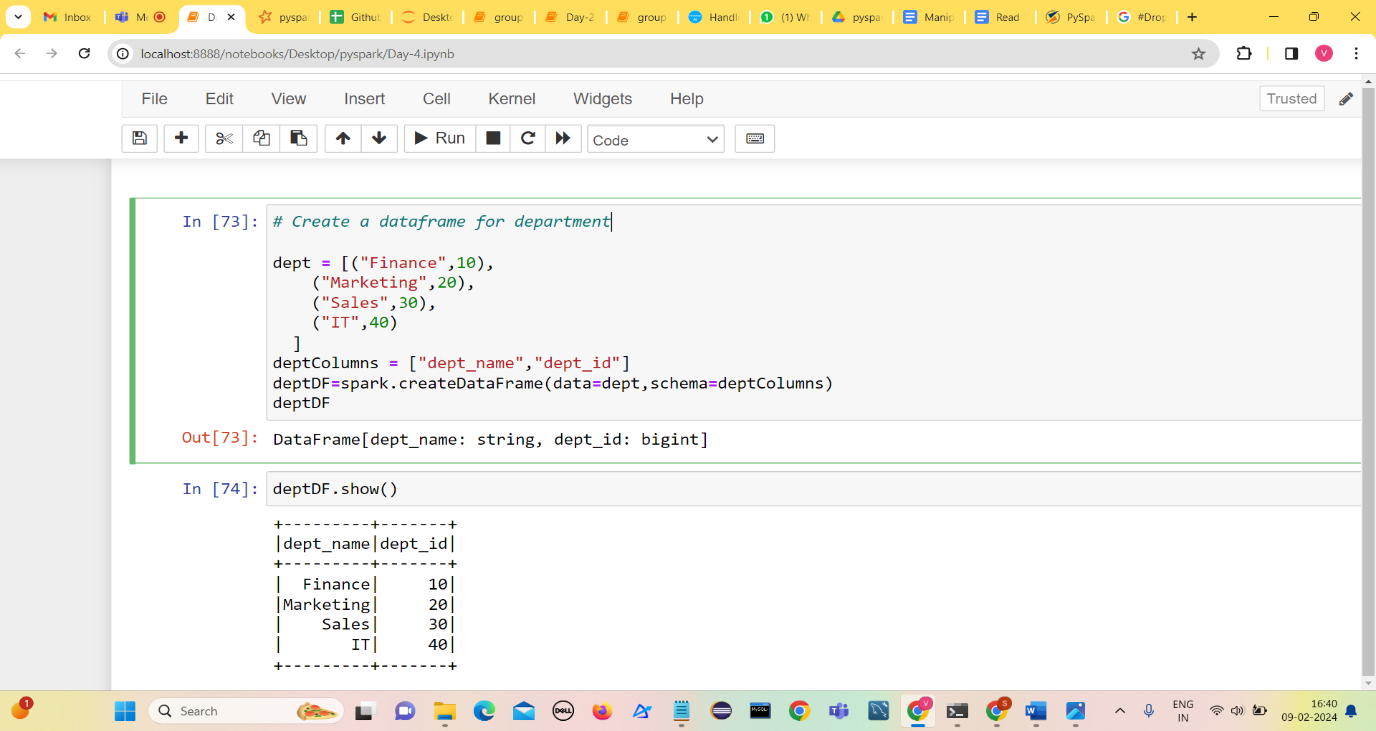
**Joins in PySpark**

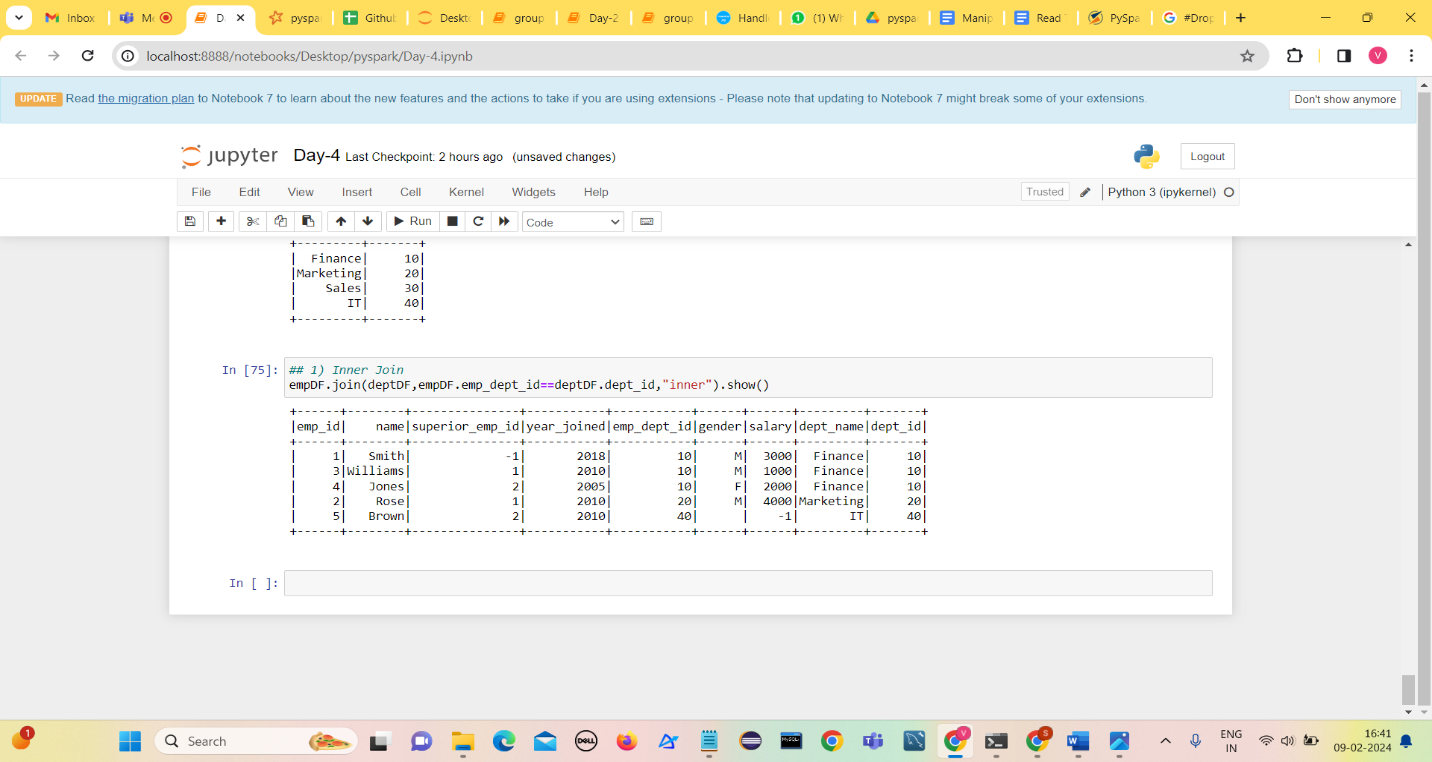
PySpark Join is used to combine two DataFrames based on the common column.

1. **Inner Join:**

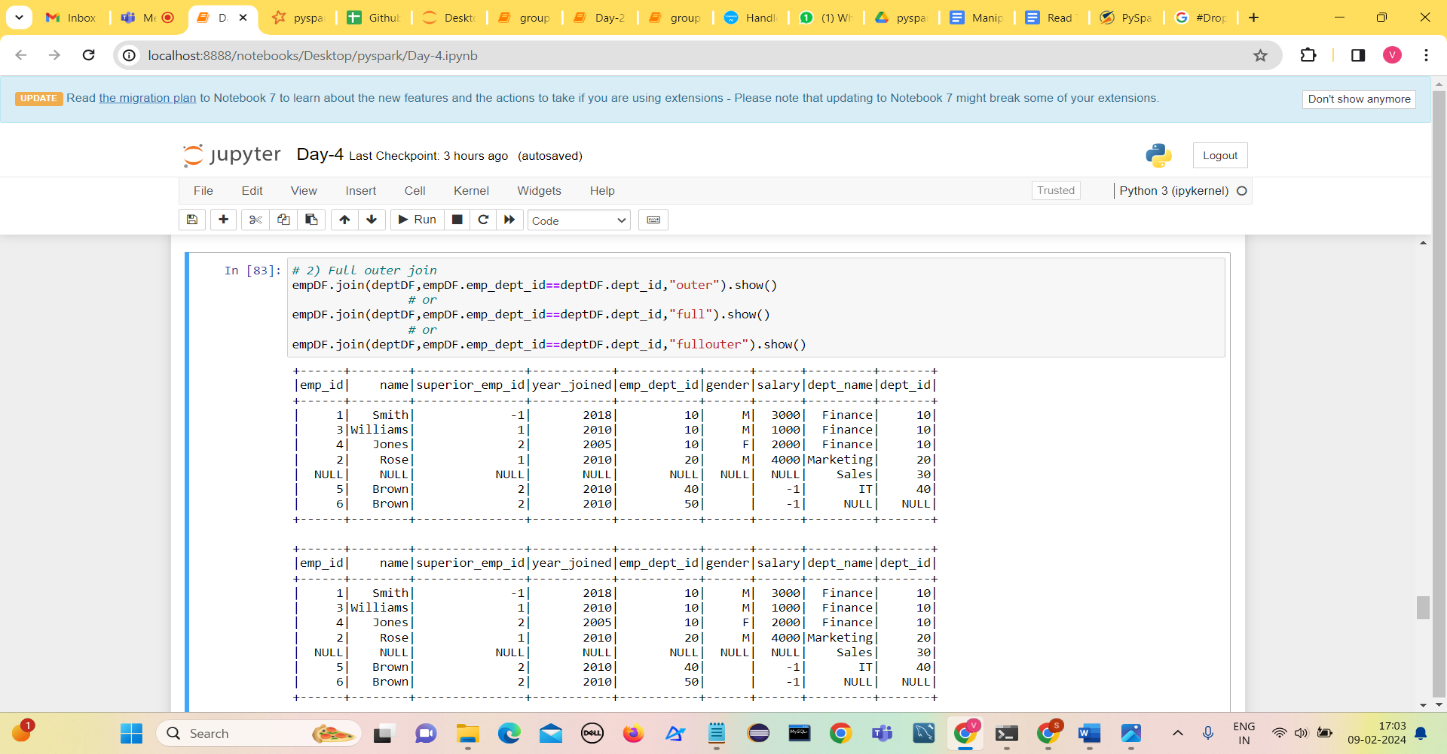
Join records when key values are matched, and dropped when they are not matched.

Create two dataframes empDF and deptDF for performing joins.

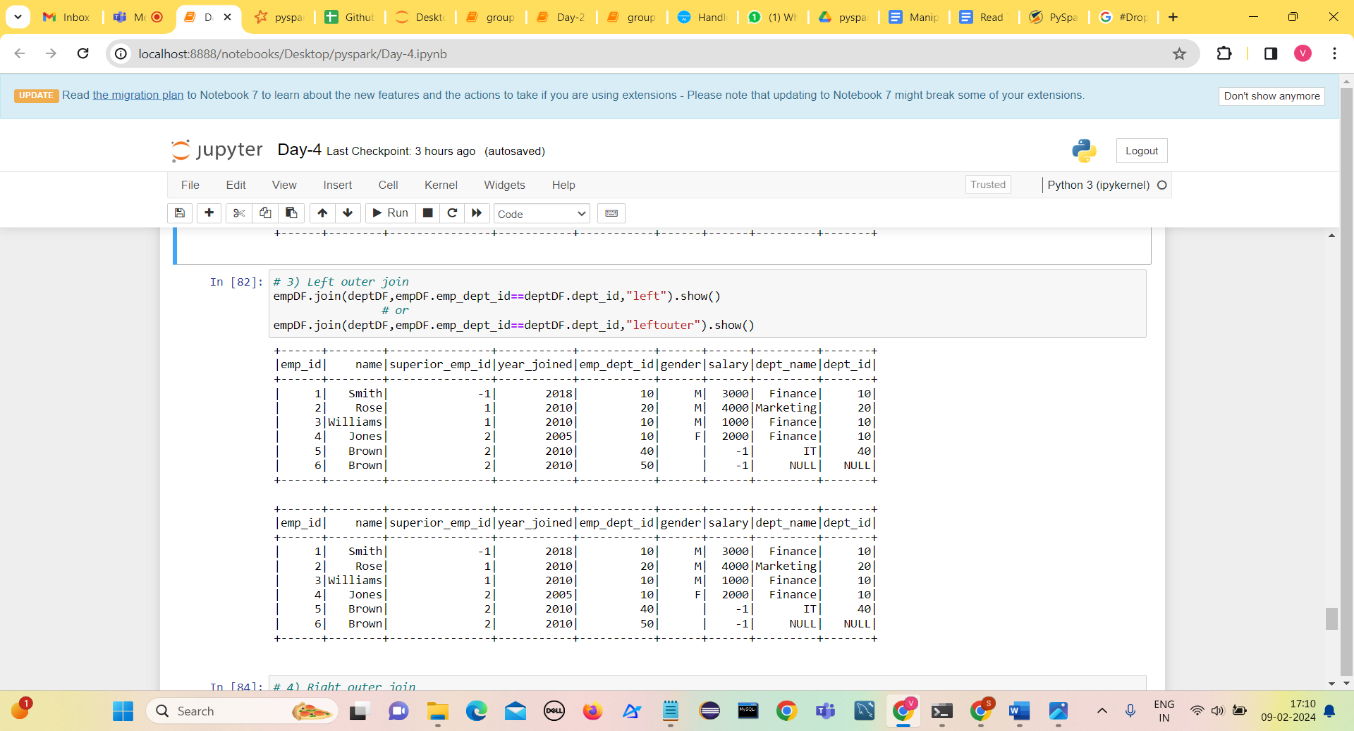


**Inner join example:**

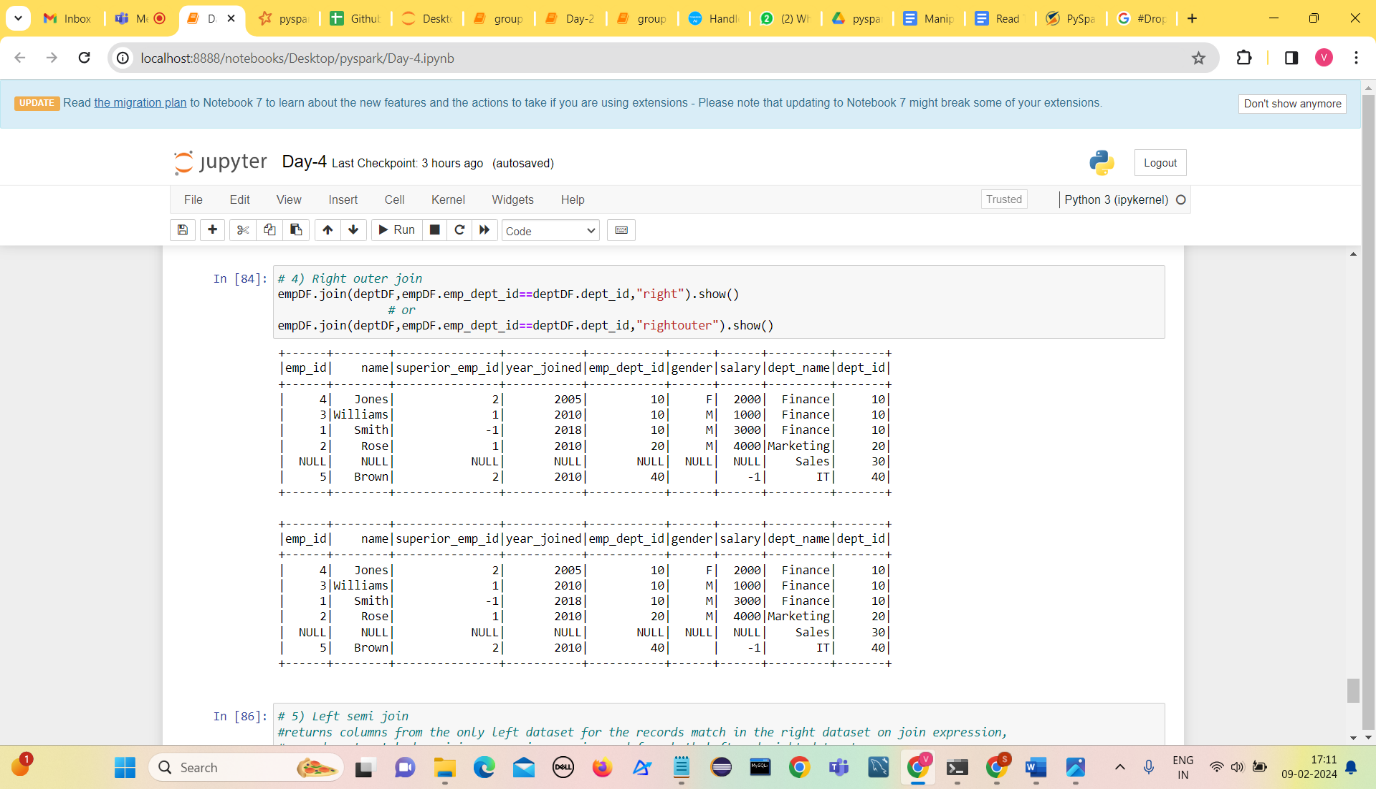
1. **Outer Join**

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

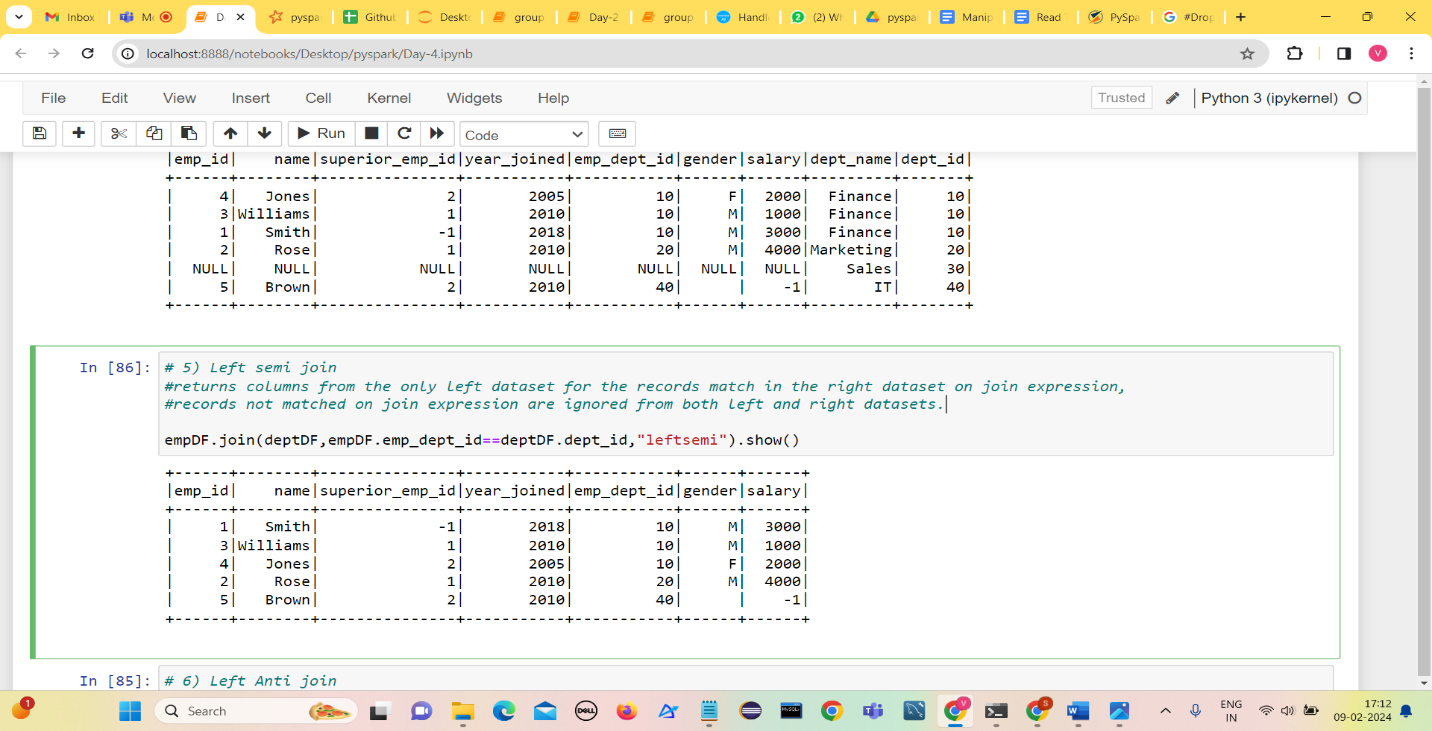
1. **Left Join / Left outer join**

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.

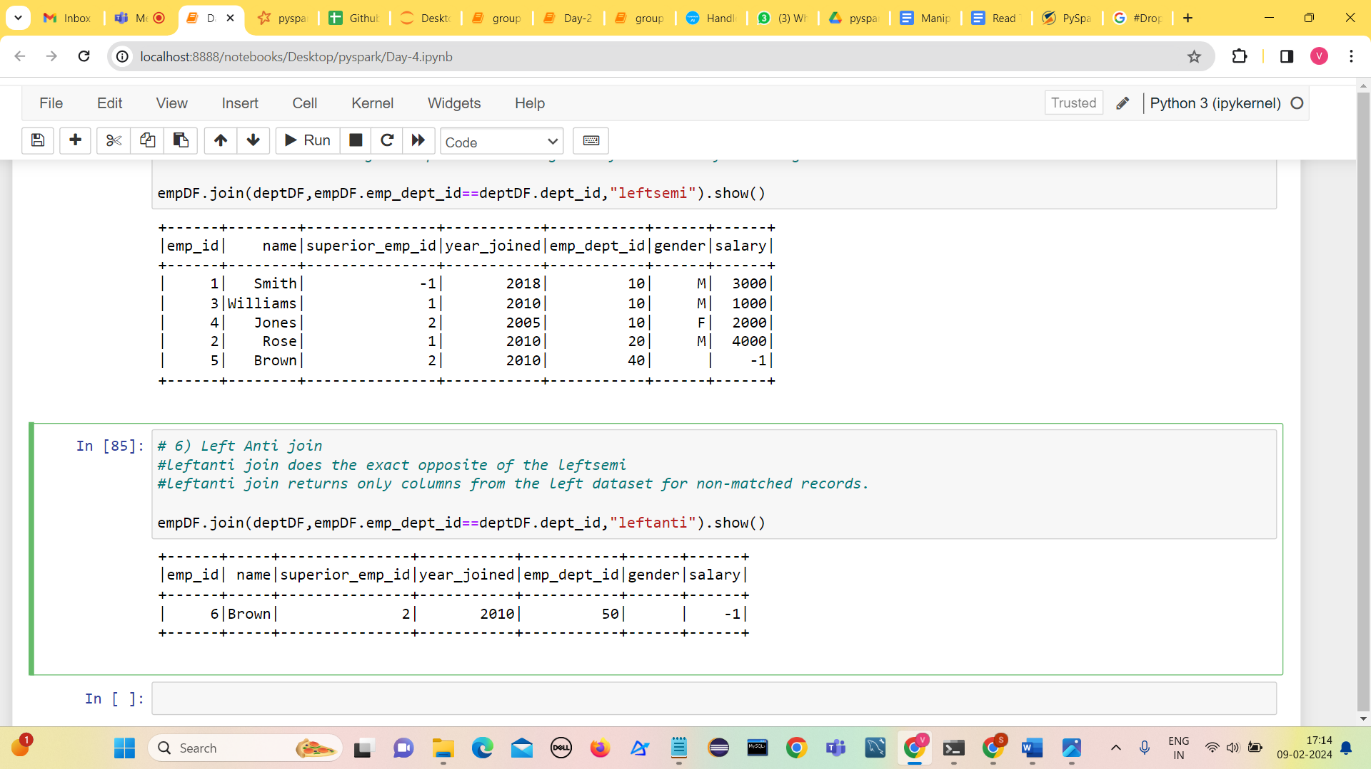
1. **Left Join / Left outer join**

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.

1. **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.

1. **Left Anti Join**

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