

Name : Parth Nandedkar

Date : 09 Feb 2024

Topics : PySpark

Batch : Data Engineering Batch-1

### Handwritten Notes :

09/02/2023

- Creating DF in Python (Creating DF (reading & writing file data) dataframes & examples)
- Looking at data in DFs. (Converting Dataframe to rdd)
- Selecting, Renaming, Filtering Data in Pandas DataFrame (load to rdd into pyspark)
- Manipulating ; Dropping, Sorting, Aggregations, Joining, GroupBy DFs. (Loading df into rdd)
- Updating data, Handling Null values, Na functions, Built in Functions
- Working on complex JSON files ; Union & Union ALL, window functions, Date & time functions ; Built in functions, in a Dataframe (Converting - window & executing functions)

Applying functions in a Pandas DataFrame (Converting dataframes of pandas to Pandas & alternatives) (Some more examples)

\* Another module - SparkSQL :

SparkSQL - Creating databases, tables & views

SparkSQL - Transformation such as filters, Join, Aggregations, Group By, Window functions etc

SparkSQL → Creating local & temporary views.

\* Implementing

## Hands-on exercise :> Working with Databricks Creating Widgets

Goal and Objectives

Pyspark - Ingestion of CSV, simple & complex JSON files into the data lake as parquet files / tables.

Pyspark - Transformation as such as filter, join, simple aggregation, groupby, window functions

Pyspark - Creating local and temporary views Hands-on.

Temporary	(join)	Temporary	(join)
local variable		local variable	
	lambda function		lambda function
After each command to run a	(join)	After each command to run a	(join)

## QUESTION

Reading txt file on Pyspark Dataframe:

spark.read.format("text").load(path=None, format=None, schema=None)  
(Creates DataFrame)

\* parquet, parquet, orc, avro, arc file formats apart from csv, textfile well handle all other file formats.

Spark.read.format("text").load("Output.txt")

↳ converts txtfile into dataframe.

When it runs on the pyspark notebook, it shows as RDD

- Using selectExpr()

And split method with split() to split the values (columns)

Schema is giving no. of rows as 1000000

With split method with split in the intermediate = 1000000

- Adding Column : →

dataframe.withColumn("column-name", lit(value));

- Concatenating :

~~dataframe.withColumn("column\_name", concat\_ws("separator", "existing1", "existing2"))~~

- Group By Aggregations :

- Afternoon Session →

Joins →

LEFT

RIGHT

Outer Join

Left semi Join

Right semi Join

Inner Join

Left anti join

Right anti join

## Select from CSV file :

A screenshot of a Databricks notebook titled "2024-02-09 - DBFS Example (2)" running in Python. The sidebar shows "Workspace" selected. In Cell 2, the following code is run:

```
# File location and type
file_location = "/FileStore/tables/test3-1.csv"
file_type = "csv"

# CSV options
infer_schema = "false"
first_row_is_header = "false"
delimiter = ","

# The applied options are for CSV files. For other file types, these will be ignored.
df = spark.read.format(file_type) \
    .option("inferSchema", infer_schema) \
    .option("header", first_row_is_header) \
    .option("sep", delimiter) \
    .load(file_location)

display(df)
```

The output shows a table with 10 rows:

_c0	_c1	_c2	_c3	
1	Name	Age	Experience	Salary
2	Krish	31	10	30000
3	Sudhanshu	30	8	25000
4	Sunny	29	4	20000
5	Paul	24	3	20000
6	Harshal	21	1	15000
7	Shubham	23	2	18000
8	Pratik	26	6	35000
9	Abhishek	28	9	40000
10	Vishal	27	7	32000

A screenshot of a Databricks notebook titled "2024-02-09 - DBFS Example (2)" running in Python. The sidebar shows "Workspace" selected. In Cell 2, the following code is run:

```
display(df)
```

The output shows a table with 10 rows:

_c0	_c1	_c2	_c3	
1	Name	Age	Experience	Salary
2	Krish	31	10	30000
3	Sudhanshu	30	8	25000
4	Sunny	29	4	20000
5	Paul	24	3	20000
6	Harshal	21	1	15000
7	Shubham	23	2	18000
8	Pratik	26	6	35000
9	Abhishek	28	9	40000
10	Vishal	27	7	32000

In Cell 3, the following code is run:

```
1 # Create a view or table
2
3 temp_table_name = "test3-1_csv"
4
```

The status bar indicates "Refreshed 8 hours ago".

The screenshot shows a Databricks workspace interface. On the left, there's a sidebar with options like 'Data Science & Engineering', 'Create', 'Workspace', 'Recents', 'Search', 'Catalog', 'Compute', and 'Workflows'. The main area is titled 'SelectingFile' and has a Python tab selected. A code cell labeled 'Cell 1' contains the following Python code:

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName('pyspark - example join').getOrCreate()
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),
        (['Jenise'], '1980-02-17', 'F', 1200)]
columns = ['Name', 'DOB', 'Gender', 'salary']
df = spark.createDataFrame(data=data, schema=columns)
df.show()
```

Below the code cell, a preview of the DataFrame is shown:

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
Jenise	1980-02-17	F	1200

At the top right, there are buttons for 'Run all', 'Connect', 'Share', and 'Publish'. The status bar at the bottom right shows 'ENG IN' and the date '2/9/2024'.

## GroupBy with different Aggregate Functions :

The screenshot shows a Databricks notebook titled "2024-02-09 - DBFS Example (1)" in Python. The code demonstrates various aggregate functions using a DataFrame named df\_pyspark, which is read from a CSV file named "Employees-1.csv".

```
spark = SparkSession.builder.appName("Practice").getOrCreate()
df_pyspark= spark.read.csv("/FileStore/tables/Employees-1.csv",header=True,inferSchema=True)
df_pyspark.show()

df_pyspark.groupBy("Department").sum("Salary").show()

df_pyspark.groupBy("Department").min("salary").show()

df_pyspark.groupBy("Department").max("salary").show()

df_pyspark.groupBy("Department").avg("salary").show()

df_pyspark.groupBy("Department").mean("salary").show()

df_pyspark.groupBy("Department").count().show()

df_pyspark.groupBy("Department").pivot("Name").sum("Salary").show()

df_pysp
```

The screenshot shows the output of the GroupBy operations from the previous notebook. It displays the original DataFrame and the results of summing the salary by department.

Department	sum(Salary)
IOI	4500000
Data_Engineering	5300000
Data_Science	7000000

community.cloud.databricks.com/?o=2228428974853171#notebook/2191140407625080/command/2191140407625086

File Edit View Run Help Last edit was 7 hours ago New cell UI: ON

Cell 4 Python

```
df_pyspark

▶ (22) Spark Jobs
▶ df_pyspark: pyspark.sql.dataframe.DataFrame = [EmpID: string, Name: string ... 2 more fields]

+-----+
| Department|min(salary)|
+-----+
| IOT| 1300000|
| Data Engineering| 800000|
| Data Science| 1200000|
+-----+

+-----+
| Department|max(salary)|
+-----+
| IOT| 1700000|
| Data Engineering| 2300000|
| Data Science| 2100000|
+-----+

+-----+
| Department| avg(salary)|
```

Run all Connect Share Publish

ENG IN 10:18 PM 2/9/2024

community.cloud.databricks.com/?o=2228428974853171#notebook/2191140407625080/command/2191140407625086

File Edit View Run Help Last edit was 7 hours ago New cell UI: ON

Cell 4 Python

```
df_pyspark

▶ (22) Spark Jobs
▶ df_pyspark: pyspark.sql.dataframe.DataFrame = [EmpID: string, Name: string ... 2 more fields]

| IOT| 1500000.0|
| Data Engineering| 1766666.6666666667|
| Data Science| 1750000.0|
+-----+

+-----+
| Department| avg(salary)|
+-----+
| IOT| 1500000.0|
| Data Engineering| 1766666.6666666667|
| Data Science| 1750000.0|
+-----+

+-----+
| Department|count|
+-----+
| IOT| 3|
| Data Engineering| 3|
| Data Science| 4|
+-----+
```

Run all Connect Share Publish

ENG IN 10:19 PM 2/9/2024

community.cloud.databricks.com/?o=2228428974853171#notebook/2191140407625080/command/2191140407625086

GitHub - devopshyd... YouTube Gmail Maps Examonline | Cand... MSRTC Facebook DO SEARCHES News Double Ended Que... Sign In All Bookmarks atharva.21910516@vit.ac.in

**databricks**

2024-02-09 · DBFS Example (1) Python ⭐

File Edit View Run Help Last edit was 7 hours ago New cell UI: ON

Run all Connect Share Publish

Cell 4 Python

df\_pysp

(22) Spark Jobs

```
df_pyspark: pypark.sql.DataFrame = [EmpID: string, Name: string ... 2 more fields]
| IOT| 1500000.0|
|Data Engineering|1766666.666666667|
| Data Science| 1750000.0|
+-----+
| Department|count|
+-----+
| IOT| 3|
|Data Engineering| 3|
| Data Science| 4|
+-----+
+-----+-----+-----+-----+-----+
| Department|Avinash| Dilip| Dinesh| Harry| Joy| Khan| Kiran| Pramod|Sabesta| Smith|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| IOT|1300000| null| null|1780000| null| null| null| null|1500000| null|
|Data Engineering| null| null|2200000| null|2300000| null|800000| null| null| null|
| Data Science| null|1980000| null| null| null|1800000| null|1200000| null|2100000|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

Menu options

ENG IN 10:19 PM 2/9/2024

The screenshot shows a Databricks notebook interface. The left sidebar contains navigation links for Data Science & Engineering, Workspace, Recents, Search, Catalog, Compute, and Workflows. The main area shows a cell titled "df\_pysp" with the following content:

```
df_pyspark: pypark.sql.DataFrame = [EmpID: string, Name: string ... 2 more fields]
| IOT| 1500000.0|
|Data Engineering|1766666.666666667|
| Data Science| 1750000.0|
+-----+
| Department|count|
+-----+
| IOT| 3|
|Data Engineering| 3|
| Data Science| 4|
+-----+
+-----+-----+-----+-----+-----+
| Department|Avinash| Dilip| Dinesh| Harry| Joy| Khan| Kiran| Pramod|Sabesta| Smith|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
| IOT|1300000| null| null|1780000| null| null| null| null|1500000| null|
|Data Engineering| null| null|2200000| null|2300000| null|800000| null| null| null|
| Data Science| null|1980000| null| null| null|1800000| null|1200000| null|2100000|
+-----+-----+-----+-----+-----+-----+-----+-----+-----+-----+
```

The notebook title is "2024-02-09 · DBFS Example (1)" and it is set to Python. The top right has buttons for "Run all", "Connect", "Share", and "Publish". The bottom right shows system status: ENG IN, 10:19 PM, 2/9/2024.

## Sorting data :

The screenshot shows a Databricks workspace interface. On the left is a sidebar with options like Create, Workspace, Recents, Search, Catalog, Compute, and Workflows. The main area is titled "2024-02-09 - DBFS Example (1)" and is set to Python. A code cell contains the following Python code:

```
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("Practice").getOrCreate()
df_pyspark = spark.read.csv("//FileStore/tables/Employees-1.csv", header=True, inferSchema=True)
df_pyspark.show()
df_pyspark.na.drop().show()
df_pyspark.na.drop(how="all").show()
df_pyspark.na.drop(how="any", thresh=2).show()
df_pyspark.sort("salary").show()
df_pyspark.sort(df_pyspark["salary"].desc()).show()
```

The code cell output shows two tables. The first table has columns [EmpID], [Name], [Salary], and [Department]. The second table also has columns [EmpID], [Name], [Salary], and [Department]. Both tables contain approximately 15 rows of employee data.

This screenshot shows the same workspace after the sorting command was run. The main area displays the sorted data frame. The salary column is sorted in descending order. The first few rows show employees with higher salaries at the top. The salary column is highlighted in yellow.

EmpID	Name	Salary	Department
e101	Pramod	1200000	Data Science
e120	Dinesh	2200000	Data Engineering
e205	Sabesta	1500000	IOT
e331	Harry	1700000	IOT
e421	Avinash	1300000	IOT
e231	Joy	2300000	Data Engineering
e222	Smith	2100000	Data Science
e339	Khan	1800000	Data Science
e150	Dilip	1900000	Data Science
e131	Kiran	800000	Data Engineering

2024-02-09 - DBFS Example (1) Python New cell UI: ON

(8) Spark Jobs

```
df_pyspark = pyspark.sql.DataFrame.empId: string, Name: string ... 2 more fields
```

EmpID	Name	Salary	Department
e101	Pramod	1200000	Data Science
e120	Dinesh	2200000	Data Engineering
e205	Sabesta	1500000	IOT
e331	Harry	1700000	IOT
e421	Avinash	1300000	IOT
e231	Joy	2300000	Data Engineering
e222	Smith	2100000	Data Science
e339	Khan	1800000	Data Science
e150	Dilip	1900000	Data Science
e131	Kiran	800000	Data Engineering

  

EmpID	Name	Salary	Department
e101	Pramod	1200000	Data Science
e120	Dinesh	2200000	Data Engineering
e205	Sabesta	1500000	IOT
e331	Harry	1700000	IOT

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

ENG IN 10:21 PM 2/9/2024

2024-02-09 - DBFS Example (1) Python New cell UI: ON

(8) Spark Jobs

```
df_pyspark = pyspark.sql.DataFrame.empId: string, Name: string ... 2 more fields
```

EmpID	Name	Salary	Department
e150	Dilip	1900000	Data Science
e131	Kiran	800000	Data Engineering

  

EmpID	Name	Salary	Department
e101	Pramod	1200000	Data Science
e120	Dinesh	2200000	Data Engineering
e205	Sabesta	1500000	IOT
e331	Harry	1700000	IOT
e421	Avinash	1300000	IOT
e231	Joy	2300000	Data Engineering
e222	Smith	2100000	Data Science
e339	Khan	1800000	Data Science
e150	Dilip	1900000	Data Science
e131	Kiran	800000	Data Engineering

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

ENG IN 10:21 PM 2/9/2024

2024-02-09 - DBFS Example (1) Python New cell UI: ON

```
(8) Spark Jobs
df_pyspark: pyspark.sql.dataframe.DataFrame = [EmpID: string, Name: string ... 2 more fields]
| e131| Kiran| 800000|Data Engineering|
+---+---+---+
| e101| Pramod|1200000| Data Science|
| e120| Dinesh|2200000|Data Engineering|
| e205|Sabesta|1500000| IOT|
| e331| Harry|1700000| IOT|
| e421|Avinash|1300000| IOT|
| e231| Joy|2300000|Data Engineering|
| e222| Smith|2100000| Data Science|
| e339| Khan|1800000| Data Science|
| e150| Dilip|1900000| Data Science|
| e131| Kiran| 800000|Data Engineering|
+---+---+---+
| EmpID| Name| Salary| Department|
+---+---+---+
| e231| Joy|2300000|Data Engineering|
+---+---+---+
```

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

2024-02-09 - DBFS Example (1) Python New cell UI: ON

```
(8) Spark Jobs
df_pyspark: pyspark.sql.dataframe.DataFrame = [EmpID: string, Name: string ... 2 more fields]
| e131| Kiran| 800000|Data Engineering|
+---+---+---+
| e101| Pramod|1200000| Data Science|
| e421|Avinash|1300000| IOT|
| e205|Sabesta|1500000| IOT|
| e331| Harry|1700000| IOT|
| e339| Khan|1800000| Data Science|
| e150| Dilip|1900000| Data Science|
| e222| Smith|2100000| Data Science|
| e120| Dinesh|2200000|Data Engineering|
| e231| Joy|2300000|Data Engineering|
+---+---+---+
| EmpID| Name| Salary| Department|
+---+---+---+
| e231| Joy|2300000|Data Engineering|
+---+---+---+
```

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

```
(8) Spark Jobs
df_pyspark = pyspark.sql.DataFrame.DataFrame = [EmpID: string, Name: string ... 2 more fields]
| e150| Dilip|1900000| Data Science|
| e222| Smith|2100000| Data Science|
| e120| Dinesh|2200000|Data Engineering|
| e231| Joy|2300000|Data Engineering|
+----+-----+
|EmpID| Name| Salary| Department|
+----+-----+
| e231| Joy|2300000|Data Engineering|
| e120| Dinesh|2200000|Data Engineering|
| e222| Smith|2100000| Data Science|
| e150| Dilip|1900000| Data Science|
| e339| Khan|1800000| Data Science|
| e331| Harry|1700000| IOT|
| e205| Sabesta|1500000| IOT|
| e421| Avinash|1300000| IOT|
| e101| Pramod|1200000| Data Science|
| e131| Kiran| 800000|Data Engineering|
+----+-----+
```

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

## Different types of JOINS :

```
File Edit View Run Help Last edit was 4 hours ago New cell UI: ON
```

```
Cell 1
emp = [(1,"Smith",-1,"2018","10","M",3000),
        (2,"Rose",1,"2010","20","M",4000),
        (3,"Williams",1,"2010","10","M",1000),
        (4,"Jones",2,"2005","10","M",2000),
        (5,"Brown",2,"2010","40","M",-1),
        (6,"Brown",2,"2010","50","M",-1)]
```

```
Cell 2
empColumns = ["emp_id", "name", "superior_emp_id", "year_joined", "emp_dept_id", "gender", "salary"]
```

```
Cell 3
empDF = spark.createDataFrame(data=emp, schema = empColumns)
empDF.printSchema()
empDF.show()

dept = [("Finance", 10), ("Marketing", 20), ("Sales", 30), ("IT",40)]
deptColumns = ["dept_name", "dept_id"]

deptDF = spark.createDataFrame(data=dept, schema = deptColumns)
deptDF.printSchema()

deptDF.show()
```

```
File Edit View Run Help Last edit was 4 hours ago New cell UI: ON
```

```
Cell 3
empDF.printSchema()
empDF.show()

dept = [("Finance", 10), ("Marketing", 20), ("Sales", 30), ("IT",40)]
deptColumns = ["dept_name", "dept_id"]

deptDF = spark.createDataFrame(data=dept, schema = deptColumns)
deptDF.printSchema()

deptDF.show()
```

```
empDF: pyspark.sql.DataFrame.DataFrame
  emp_id: long
  name: string
  superior_emp_id: long
  year_joined: string
  emp_dept_id: string
  gender: string
  salary: long

deptDF: pyspark.sql.DataFrame.DataFrame
  dept_name: string
  dept_id: long

root
|-- emp_id: long (nullable = true)
|-- name: string (nullable = true)
```

Screenshot of a Databricks workspace titled "Joins". The sidebar shows "Recent" and "Search" items. The main area displays two DataFrames:

```
gender: string  
salary: long  
  
deptDF: pyspark.sql.dataframe.DataFrame  
  dept_name: string  
  dept_id: long  
  
+---+---+---+---+  
| 1| Smith| -1| 2018| 10| M| 3000|  
| 2| Rose| 1| 2010| 20| M| 4000|  
| 3| Williams| 1| 2018| 10| M| 1000|  
| 4| Jones| 2| 2005| 10| M| 2000|  
| 5| Brown| 2| 2018| 40| M| -1|  
| 6| Brown| 2| 2010| 50| M| -1|  
+---+---+---+---+  
  
root  
|-- dept_name: string (nullable = true)  
|-- dept_id: long (nullable = true)  
  
+---+  
|dept_name|dept_id|  
+---+  
| Finance| 10|  
| Marketing| 20|  
| Sales| 30|  
| IT| 40|  
+---+
```

Screenshot of a Databricks workspace titled "Joins". The sidebar shows "Recent" and "Search" items. The main area displays two cells of code and their results:

Cell 4:

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id, "inner").show()
```

Result:

```
+---+---+---+---+---+---+  
|emp_id| name|superior_emp_id|year_joined|emp_dept_id|gender|salary|dept_name|dept_id|  
+---+---+---+---+---+---+  
| 1| Smith| -1| 2018| 10| M| 3000| Finance| 10|  
| 3| Williams| 1| 2010| 10| M| 1000| Finance| 10|  
| 4| Jones| 2| 2005| 10| M| 2000| Finance| 10|  
| 2| Rose| 1| 2010| 20| M| 4000| Marketing| 20|  
| 5| Brown| 2| 2018| 40| M| -1| IT| 40|  
+---+---+---+---+---+  
  
Cell 5:
```

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id, "outer").show()
```

Result:

```
+---+---+---+---+---+---+  
|emp_id| name|superior_emp_id|year_joined|emp_dept_id|gender|salary|dept_name|dept_id|  
+---+---+---+---+---+---+  
| 1| Smith| -1| 2018| 10| M| 3000| Finance| 10|  
| 3| Williams| 1| 2010| 10| M| 1000| Finance| 10|
```

Screenshot of a Databricks workspace showing a Python notebook titled "Joins". The notebook contains two cells:

**Cell 5:**

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"outer").show()
```

**Cell 6:**

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"full").show()
```

The output of Cell 5 shows the result of an outer join between empDF and deptDF. The resulting DataFrame includes columns: emp\_id, name, superior\_emp\_id, year\_joined, emp\_dept\_id, gender, salary, dept\_name, and dept\_id. The data is as follows:

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
4	Jones	2	2005	10	M	2000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
null	null	null	null	null	null	null	Sales	30
5	Brown	2	2010	40	M	-1	IT	40
6	Brown	2	2010	50	M	-1	null	null

The output of Cell 6 shows the result of a full join between empDF and deptDF. The resulting DataFrame includes columns: emp\_id, name, superior\_emp\_id, year\_joined, emp\_dept\_id, gender, salary, dept\_name, and dept\_id. The data is identical to Cell 5.

Screenshot of a Databricks workspace showing a Python notebook titled "Joins". The notebook contains two cells:

**Cell 6:**

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"full").show()
```

**Cell 7:**

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"fullouter").show()
```

The output of Cell 6 is identical to the output of Cell 5 in the previous screenshot.

The output of Cell 7 shows the result of a full outer join between empDF and deptDF. The resulting DataFrame includes columns: emp\_id, name, superior\_emp\_id, year\_joined, emp\_dept\_id, gender, salary, dept\_name, and dept\_id. The data is identical to Cell 5.

community.cloud.databricks.com/?o=2228428974853171#notebook/1349101789246736/command/542454768880586

File Edit View Run Help Last edit was 4 hours ago New cell UI: ON

Run all Connect Share Publish

Joins Python

(3) Spark Jobs

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"fullouter").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
4	Jones	2	2005	10	M	2000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
null	null	null	null	null	null	null	Sales	30
5	Brown	2	2010	40	M	-1	IT	40
6	Brown	2	2010	50	M	-1	null	null

Cell 8

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"left").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
3	Williams	1	2010	10	M	1000	Finance	10
4	Jones	2	2005	10	M	2000	Finance	10
5	Brown	2	2010	40	M	-1	IT	40
6	Brown	2	2010	50	M	-1	null	null

10:24 PM 2/9/2024 ENG IN

community.cloud.databricks.com/?o=2228428974853171#notebook/1349101789246736/command/542454768880586

File Edit View Run Help Last edit was 4 hours ago New cell UI: ON

Run all Connect Share Publish

Joins Python

(6) Spark Jobs

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"left").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
3	Williams	1	2010	10	M	1000	Finance	10
4	Jones	2	2005	10	M	2000	Finance	10
5	Brown	2	2010	40	M	-1	IT	40
6	Brown	2	2010	50	M	-1	null	null

Cell 9

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"leftouter").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
3	Williams	1	2010	10	M	1000	Finance	10

10:24 PM 2/9/2024 ENG IN

community.cloud.databricks.com/?o=2228428974853171#notebook/1349101789246736/command/542454768880586

File Edit View Run Help Last edit was 4 hours ago New cell UI: ON

Run all Connect Share Publish

**Joins** Python

(6) Spark Jobs

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"leftouter").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
3	Williams	1	2010	10	M	1000	Finance	10
4	Jones	2	2005	10	M	2000	Finance	10
5	Brown	2	2010	40	M	-1	IT	40
6	Brown	2	2010	50	M	-1	null	null

05:39 PM (DS)

empDF.join(deptDF,empDF.emp\_dept\_id == deptDF.dept\_id,"right").show()

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
4	Jones	2	2005	10	M	2000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
null	null	null	null	null	null	null	Sales	30
5	Brown	2	2010	40	M	-1	IT	40

05:39 PM (DS)

community.cloud.databricks.com/?o=2228428974853171#notebook/1349101789246736/command/542454768880586

File Edit View Run Help Last edit was 4 hours ago New cell UI: ON

Run all Connect Share Publish

**Joins** Python

(6) Spark Jobs

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"right").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
4	Jones	2	2005	10	M	2000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
1	Smith	-1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
null	null	null	null	null	null	null	Sales	30
5	Brown	2	2010	40	M	-1	IT	40

05:40 PM (BS)

```
empDF.join(deptDF,empDF.emp_dept_id == deptDF.dept_id,"rightouter").show()
```

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
4	Jones	2	2005	10	M	2000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
1	Smith	-1	2018	10	M	3000	Finance	10

05:40 PM (BS)

The screenshot shows a Databricks workspace interface. On the left sidebar, there are links for Data Science & Engineering, Create, Workspace, Recents, Search, Catalog, Compute, Workflows, and Menu options. The main area displays two code cells in Python:

**Cell 11:**

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id,"rightouter").show()
```

(6) Spark Jobs

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary	dept_name	dept_id
4	Jones	2	2005	10	M	2000	Finance	10
3	Williams	1	2010	10	M	1000	Finance	10
1	Smith	1	2018	10	M	3000	Finance	10
2	Rose	1	2010	20	M	4000	Marketing	20
null	null	null	null	null	null	null	Sales	30
5	Brown	2	2010	40	M	-1	IT	40

**Cell 12:**

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id,"leftsemi").show()
```

(3) Spark Jobs

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary
4	Jones	2	2005	10	M	2000
3	Williams	1	2010	10	M	1000
1	Smith	1	2018	10	M	3000

The screenshot shows a Databricks workspace interface. On the left sidebar, there are sections for Data Science & Engineering, Create, Workspace, Recents, Search, Catalog, Compute, and Workflows. The main area is titled "Joins" and "Python".

**Cell 12:**

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id,"leftsemi").show()
```

(3) Spark Jobs

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary
1	Smith	-1	2018	10	M	3000
3	Williams	1	2010	10	M	1000
4	Jones	2	2005	10	M	2000
2	Rose	1	2010	20	M	4000
5	Brown	2	2010	40	M	-1

**Cell 13:**

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id,"leftanti").show()
```

(6) Spark Jobs

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary
6	Brown	2	2010	50	M	-1

Screenshot of a Databricks workspace showing a notebook titled "Joins".

The sidebar on the left includes:

- Data Science & Engineering
- Create
- Workspace (selected)
- Recents
- Search
- Catalog
- Compute
- Workflows

The main area displays a table and some code:

	4	Jones	2	2005	10	M	2000
	2	Rose	1	2010	20	M	4000
	5	Brown	2	2010	40	M	-1

```
empDF.join(deptDF, empDF.emp_dept_id == deptDF.dept_id, "leftanti").show()
```

Cell 13 output:

emp_id	name	superior_emp_id	year_joined	emp_dept_id	gender	salary
6	Brown	2	2010	50	M	-1

Bottom status bar:

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

System tray icons: Windows, Task View, Chrome, File Explorer, Edge, Task Manager.

User information: atharva.21910516@viit.ac.in