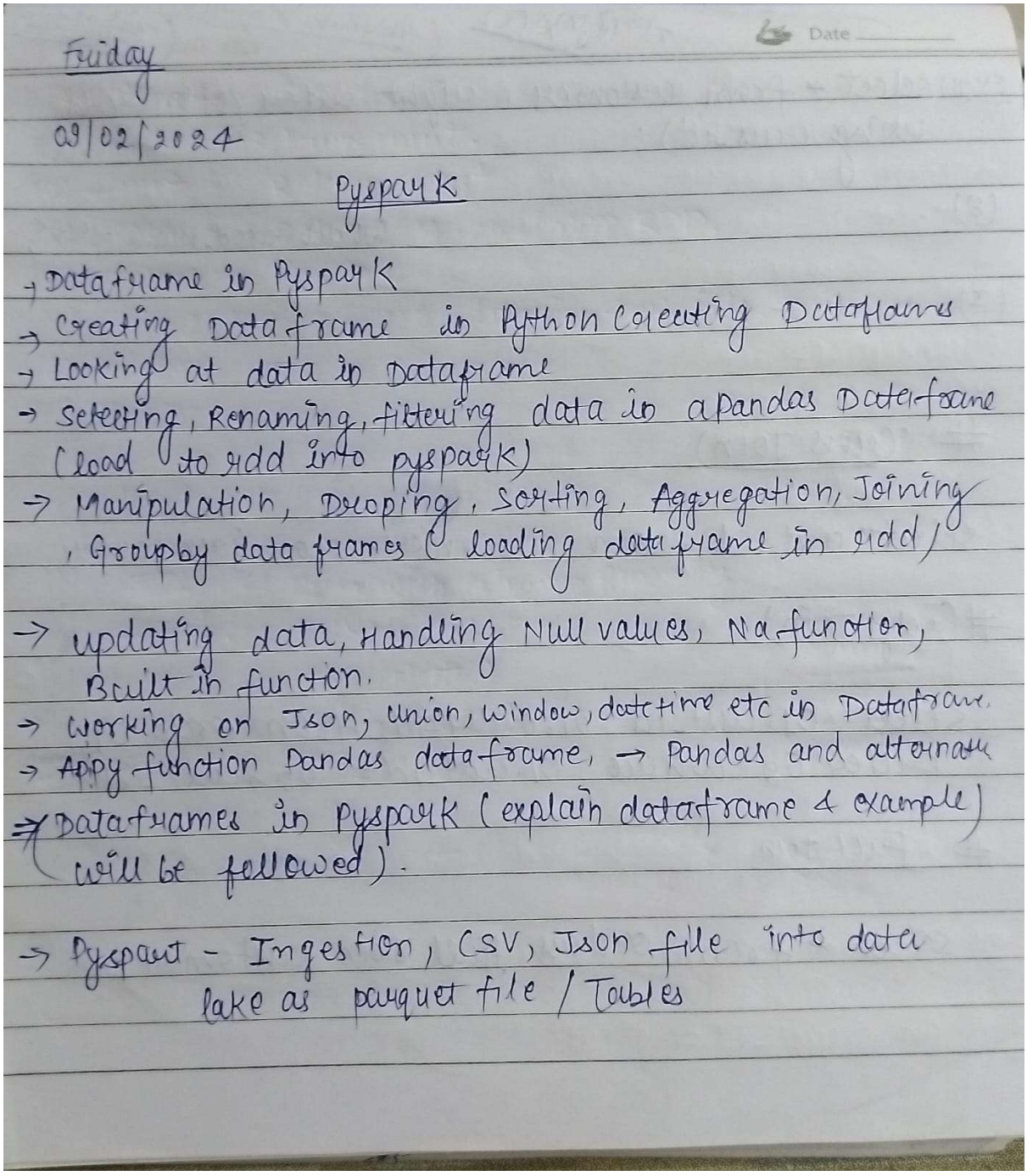
Name: Abhishek Kanoujia

DATA ENGINEERING BATCH 1

DAY 15 ASSIGNMENT

Class hand written notes:-



Friday

09/02/2024

PySpark

- Dataframe in Pyspark
- Creating Dataframe in Python Creating Dataframes
- Looking at data in Dataframe
- Selecting, Renaming, filtering data in a pandas Dataframe (load to rdd into pyspark)
- Manipulation, Droping, Sorting, Aggregation, Joining, Groupby data frames (loading dataframe in rdd)
- updating data, Handling Null values, Na function, Built in function.
- working on Json, union, window, datetime etc in Dataframe.
- Appy function Dandas dataframe, → Pandas and alternate
- Dataframes in Pyspark (explain dataframe & example) will be followed).
- Pyspark - Ingestion, CSV, Json file into data lake as parquet file / Tables

Pyspark: Transformation such as filter, Join, simple Aggregation, group by window function.

Pyspark: Create local and temporary view - hand on.

(Another Module - spark SQL) ⇒

⇒ Create database, table and view.
⇒ Transform such as filter, Join, simple Aggregation, Group by, window functions etc.
⇒ Create local and temporary view.

(Implementing full refresh and incremental load pattern using partition:)

Hands:on - Processing Json and csv data with Pyspark
Hands on - ETL (Extract, Transform, Load) with Pyspark.

Method: spark.read.format()

⇒ Parquet, orc, avro are file formats apart from csv, textfile we have.

Syntax: spark.read.format("text").load("output.txt")
----> it convert textfile into dataframe.
-----> when it run on the pyspark Notebook, it stores as RDD.

Ex: Program.

```
from pyspark.sql import SparkSession.
(it import spark session through pyspark.sql modul
```

Select: df.selectExpr("split(value,')).show()
df.selectExpr("Avg(AGE) as average").show.

Columns: ⇒ dataframe.withcolumn("columnname", lit(val
(Add column)

concat_ws(): concat two column and make new one.

Syntax:

dataframe.withColumn("Details", concat_ws("-", "Name", "company")).show().

groupBy():

df_pyspark.groupBy("Department").sum("salary")

databricks

abhishek7621cse@gmail.com

select and pandas pyspark fri

Python

File Edit View Run Help Last edit was 2 minutes ago New cell UI: OFF

Run all | My Cluster | Share | Publish

Cmd 1

```
import pandas as pd
data = [["James","","Smith",30,"M",60000],
["Michael","Rose","",50,"M",70000],
["Robert","","Williams",42,"",400000],
["Maria","Anne","Jones",38,"F",500000],
["Jen","Mary","Brown",45,None,0]]
columns=['First Name','Middle Name','Last Name','Age','Gender','Salary']
# Create the pandas DataFrame
pandasDF=pd.DataFrame(data=data, columns=columns)
# print dataframe.
print(pandasDF)
```

```
  First Name Middle Name Last Name  Age Gender  Salary
0      James                Smith   30      M   60000
1    Michael        Rose             50      M   70000
2     Robert              Williams   42         400000
3      Maria        Anne     Jones   38      F  500000
4        Jen        Mary     Brown   45   None       0
```

Command took 5.93 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:17:51 PM on My Cluster

Cmd 2

```
print(pandasDF.count())
```

```
First Name     5
Middle Name    5
Last Name      5
Age            5
Gender         4
Salary         5
dtype: int64
```

Command took 0.15 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:18:11 PM on My Cluster

Cmd 3

```

print(pandasDF.max())
```

```
First Name       Robert
Middle Name        Rose
Last Name      Williams
Age                  50
Salary           500000
dtype: object
<command-864506446244 21>:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  S
elect only valid columns before calling the reduction.
  print(pandasDF.max())
```

Command took 0.14 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:22:41 PM on My Cluster

Cmd 4

```
print(pandasDF.mean())
```

```
<command-864506446244 22>:1: FutureWarning: Dropping of nuisance columns in DataFrame reductions (with 'numeric_only=None') is deprecated; in a future version this will raise TypeError.  S
elect only valid columns before calling the reduction.
  print(pandasDF.mean())
Age          41.0
Salary    206000.0
dtype: float64
```

Command took 0.13 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:18:31 PM on My Cluster

Cmd 5

```

```

```
NameError: name 'createOrReplaceTempView' is not defined
```

Command took 1.07 minutes -- by abhishek7621cse@gmail.com at 2/9/2024, 5:18:43 PM on My Cluster

Cmd 6

```
from pyspark.sql import SparkSession
# Create spark session
spark = SparkSession \
.builder \
.appName("SparkByExamples.com") \
.enableHiveSupport() \
.getOrCreate()
data = [("James","Smith","USA","CA"),
("Michael","Rose","USA","NY"),
("Robert","Williams","USA","CA"),
("Maria","Jones","USA","FL")
]
columns = ["firstname","lastname","country","state"]
# Create dataframe
sampleDF = spark.sparkContext.parallelize(data).toDF(columns)
sampleDF.show()
```

▸ (5) Spark Jobs

▸ sampleDF: pyspark.sql.dataframe.DataFrame = [firstname: string, lastname: string ... 2 more fields]

```
+---------+--------+-------+-----+
|firstname|lastname|country|state|
+---------+--------+-------+-----+
|    James|   Smith|    USA|   CA|
```

```
|    James|    Smith|    USA|    CA|
|  Michael|     Rose|    USA|    NY|
|   Robert| Williams|    USA|    CA|
|    Maria|    Jones|    USA|    FL|
+---------+---------+-------+------+
```

Command took 13.67 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:19:09 PM on My Cluster

Cmd 7

```
# Create Temporary View/Table
sampleDF.createOrReplaceTempView("Person")
```

Command took 0.63 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:19:19 PM on My Cluster

Cmd 8

```
# Run SQL Query
spark.sql("select firstname, lastname from Person").show()
```

▸ (3) Spark Jobs

```
+---------+--------+
|firstname|lastname|
+---------+--------+
|    James|   Smith|
|  Michael|    Rose|
|   Robert|Williams|
|    Maria|   Jones|
+---------+--------+
```

Command took 2.31 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:19:29 PM on My Cluster

Cmd 9

```
# Importing necessary libraries
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]

```
+------+----------+------+------+
|  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|
+------+----------+------+------+
```

Command took 1.34 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:24:21 PM on My Cluster

Cmd 10

```
# Rename the column name from DOB to DateOfBirth
# Print the dataframe
df.withColumnRenamed("DOB","DateOfBirth").show()
```

▸ (3) Spark Jobs

```
+------+-----------+------+------+
|  Name|DateOfBirth|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|
+------+-----------+------+------+
```

Command took 0.80 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:24:39 PM on My Cluster

Cmd 11

```
# Rename the column name 'Gender' to 'Sex'
# Then for the returning dataframe
# again rename the 'salary' to 'Amount'
df.withColumnRenamed("Gender","Sex").withColumnRenamed("salary","Amount").show()
```

▸ (3) Spark Jobs

```
+------+----------+---+------+
|  Name|       DOB|Sex|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|
```

```
+------+----------+---+------+
```

Command took 0.81 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:25:04 PM on My Cluster

Cmd 12

```
# 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|
+------+----------+------+------+
|   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|
+------+----------+------+------+
```

Command took 0.78 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:25:18 PM on My Cluster

Cmd 13

```
# 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|
+------+----------+------+------+
|   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|
+------+----------+------+------+
```

Command took 0.72 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:25:28 PM on My Cluster

Cmd 14

```
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|
+--------+-------------+-----------+-----------+
|     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|
+--------+-------------+-----------+-----------+
```

Command took 0.60 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 5:25:41 PM on My Cluster

Cmd 15

```

```

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

Pyspark hands on 09-02-2024

In [36]:
```
from pyspark.sql import SparkSession

spark =SparkSession.builder.appName("Practice").getOrCreate()

df_pyspark= spark.read.csv("Bibadata.csv",header=True, inferSchema=True)

df_pyspark.show()
```

```
+-------+---+------+---------+-------------+-----+-------+------+-----+
|Product|Age|Gender|Education|MaritalStatus|Usage|Fitness|Income|Miles|
+-------+---+------+---------+-------------+-----+-------+------+-----+
|  TM195| 18|  Male|       14|       Single|    3|      4| 29562|  112|
|  TM195| 19|  Male|       15|       Single|    2|      3| 31836|   75|
|  TM195| 19|Female|       14|    Partnered|    4|      3| 30699|   66|
|  TM195| 19|  Male|       12|       Single|    3|      3| 32973|   85|
|  TM195| 20|  Male|       13|    Partnered|    4|      2| 35247|   47|
|  TM195| 20|Female|       14|    Partnered|    3|      3| 32973|   66|
|  TM195| 21|Female|       14|    Partnered|    3|      3| 35247|   75|
|  TM195| 21|  Male|       13|       Single|    3|      3| 32973|   85|
|  TM195| 21|  Male|       15|       Single|    5|      4| 35247|  141|
|  TM195| 21|Female|       15|    Partnered|    2|      3| 37521|   85|
|  TM195| 22|  Male|       14|       Single|    3|      3| 36384|   85|
|  TM195| 22|Female|       14|    Partnered|    3|      2| 35247|   66|
|  TM195| 22|Female|       16|       Single|    4|      3| 36384|   75|
|  TM195| 22|Female|       14|       Single|    3|      3| 35247|   75|
|  TM195| 23|  Male|       16|    Partnered|    3|      1| 38658|   47|
|  TM195| 23|  Male|       16|    Partnered|    3|      3| 40932|   75|
|  TM195| 23|Female|       14|       Single|    2|      3| 34110|  103|
|  TM195| 23|  Male|       16|    Partnered|    4|      3| 39795|   94|
|  TM195| 23|Female|       16|       Single|    4|      3| 38658|  113|
|  TM195| 23|Female|       15|    Partnered|    2|      2| 34110|   38|
+-------+---+------+---------+-------------+-----+-------+------+-----+
only showing top 20 rows
```

In [2]:
```
df_pyspark.groupBy("MaritalStatus").sum("Income").show()
```

```
+-------------+-----------+
|MaritalStatus|sum(Income)|
+-------------+-----------+
|       Single|    3702883|
|    Partnered|    5966641|
+-------------+-----------+
```

In [3]:
```
df_pyspark.groupBy("Gender").sum("Income").show()
```

```
+------+-----------+
|Gender|sum(Income)|
+------+-----------+
|Female|    3786997|
|  Male|    5882527|
+------+-----------+
```

In [8]:
```
df_pyspark.groupBy("MaritalStatus").min("Income").show()
```

```
+-------------+-----------+
|MaritalStatus|min(Income)|
+-------------+-----------+
|       Single|      29562|
|    Partnered|      30699|
+-------------+-----------+
```

In [9]:
```
df_pyspark.groupBy("MaritalStatus").max("Income").show()
```

```
+-------------+-----------+
|MaritalStatus|max(Income)|
+-------------+-----------+
|       Single|      92131|
|    Partnered|     104581|
+-------------+-----------+
```

In [10]:
```
df_pyspark.groupBy("MaritalStatus").avg("Income").show()
```

```
+-------------+-----------------+
|MaritalStatus|      avg(Income)|
+-------------+-----------------+
|       Single|50724.42465753425|
|    Partnered|          55763.0|
+-------------+-----------------+
```

In [12]:
```
df_pyspark.groupBy("MaritalStatus").mean("Income").show()
```

```
+-------------+-----------------+
|MaritalStatus|      avg(Income)|
+-------------+-----------------+
|       Single|50724.42465753425|
|    Partnered|          55763.0|
+-------------+-----------------+
```

In [13]:
```
df_pyspark.groupBy("MaritalStatus").count().show()
```

```
+-------------+-----+
|MaritalStatus|count|
+-------------+-----+
|       Single|   73|
|    Partnered|  107|
+-------------+-----+
```

In [15]:
```
df_pyspark.groupBy("MaritalStatus").pivot("Product").sum("Income").show()
```

```
+-------------+-------+-------+-------+
|MaritalStatus|  TM195|  TM498|  TM798|
+-------------+-------+-------+-------+
|       Single|1416702|1155603|1130578|
|    Partnered|2296740|1782816|1887085|
+-------------+-------+-------+-------+
```

Handling Missing Values Pyspark

In [16]:
```
df_pyspark1=spark.read.csv("test3.csv",header=True,inferSchema=True)
df_pyspark1.show()
```

```
+---------+----+----------+------+
|     Name| age|Experience|Salary|
+---------+----+----------+------+
|    Krish|  31|        10| 30000|
|Sudhanshu|  30|         8| 25000|
|    Sunny|  29|         4| 20000|
|     Paul|  24|         3| 20000|
|   Harsha|  21|         1| 15000|
|  Shubham|  23|         2| 18000|
|   Mahesh|NULL|      NULL| 40000|
|     NULL|  34|        10| 38000|
|     NULL|  36|      NULL|  NULL|
+---------+----+----------+------+
```

Dropping rows based on null values

In [17]:
```
df_pyspark1.na.drop().show()
```

```
+---------+---+----------+------+
|     Name|age|Experience|Salary|
+---------+---+----------+------+
|    Krish| 31|        10| 30000|
|Sudhanshu| 30|         8| 25000|
|    Sunny| 29|         4| 20000|
|     Paul| 24|         3| 20000|
|   Harsha| 21|         1| 15000|
|  Shubham| 23|         2| 18000|
+---------+---+----------+------+
```

if all values in rows are null then

This command drops rows where all values are null.

In [18]:
```
df_pyspark1.na.drop(how="all").show()
```

```
+---------+----+----------+------+
|     Name| age|Experience|Salary|
+---------+----+----------+------+
|    Krish|  31|        10| 30000|
|Sudhanshu|  30|         8| 25000|
|    Sunny|  29|         4| 20000|
|     Paul|  24|         3| 20000|
|   Harsha|  21|         1| 15000|
|  Shubham|  23|         2| 18000|
|   Mahesh|NULL|      NULL| 40000|
|     NULL|  34|        10| 38000|
|     NULL|  36|      NULL|  NULL|
+---------+----+----------+------+
```

atleast 2 non null values should be present.

This drops rows where at least 2 or more values are null.

In [30]:
```
df_pyspark1.na.drop(how="any",thresh=2).show()
```

```
+---------+----+----------+------+
|     Name| age|Experience|Salary|
+---------+----+----------+------+
|    Krish|  31|        10| 30000|
|Sudhanshu|  30|         8| 25000|
|    Sunny|  29|         4| 20000|
|     Paul|  24|         3| 20000|
|   Harsha|  21|         1| 15000|
|  Shubham|  23|         2| 18000|
|   Mahesh|NULL|      NULL| 40000|
|     NULL|  34|        10| 38000|
+---------+----+----------+------+
```

only in that column rows get deleted

his drops rows where the "salary" column has null values.

In [20]:
```
df_pyspark1.na.drop(how="any",subset=["salary"]).show()
```

```
+---------+----+----------+------+
|     Name| age|Experience|Salary|
+---------+----+----------+------+
|    Krish|  31|        10| 30000|
|Sudhanshu|  30|         8| 25000|
|    Sunny|  29|         4| 20000|
|     Paul|  24|         3| 20000|
```

```
|   Harsha|  21|         1| 15000|
|  Shubham|  23|         2| 18000|
|   Mahesh|NULL|      NULL| 40000|
|     NULL|  34|        10| 38000|
+---------+----+----------+------+
```

orderBy() and sort() in Pyspark DataFrame

In [21]:
```
from pyspark.sql import SparkSession
spark =SparkSession.builder.appName("Practice").getOrCreate()
df_pyspark= spark.read.csv("test2.csv",header=True,inferSchema=True)
```

In [22]:
```
df_pyspark.show()
```

```
+---------+-----------+------+
|     Name| Departments|salary|
+---------+-----------+------+
|    Krish|Data Science| 10000|
|    Krish|         IOT|  5000|
|   Mahesh|    Big Data|  4000|
|    Krish|    Big Data|  4000|
|   Mahesh|Data Science|  3000|
|Sudhanshu|Data Science| 20000|
|Sudhanshu|         IOT| 10000|
|Sudhanshu|    Big Data|  5000|
|    Sunny|Data Science| 10000|
|    Sunny|    Big Data|  2000|
+---------+-----------+------+
```

Sort based on single column

In [23]:
```
df_pyspark.sort("salary").show()
```

```
+---------+-----------+------+
|     Name| Departments|salary|
+---------+-----------+------+
|    Sunny|    Big Data|  2000|
|   Mahesh|Data Science|  3000|
|   Mahesh|    Big Data|  4000|
|    Krish|    Big Data|  4000|
|    Krish|         IOT|  5000|
|Sudhanshu|    Big Data|  5000|
|    Krish|Data Science| 10000|
|Sudhanshu|         IOT| 10000|
|    Sunny|Data Science| 10000|
|Sudhanshu|Data Science| 20000|
+---------+-----------+------+
```

sort based on descending order

In [24]:
```
df_pyspark.sort(df_pyspark["salary"].desc()).show()
```

```
+---------+-----------+------+
|     Name| Departments|salary|
+---------+-----------+------+
|Sudhanshu|Data Science| 20000|
|    Krish|Data Science| 10000|
|Sudhanshu|         IOT| 10000|
|    Sunny|Data Science| 10000|
|    Krish|         IOT|  5000|
|Sudhanshu|    Big Data|  5000|
|   Mahesh|    Big Data|  4000|
|    Krish|    Big Data|  4000|
|   Mahesh|Data Science|  3000|
|    Sunny|    Big Data|  2000|
+---------+-----------+------+
```

Sort based on first column then second column

In [25]:
```
df_pyspark.sort("salary","Name").show()
```

```
+---------+-----------+------+
|     Name| Departments|salary|
+---------+-----------+------+
|    Sunny|    Big Data|  2000|
|   Mahesh|Data Science|  3000|
|    Krish|    Big Data|  4000|
|   Mahesh|    Big Data|  4000|
|    Krish|         IOT|  5000|
|Sudhanshu|    Big Data|  5000|
|    Krish|Data Science| 10000|
|Sudhanshu|         IOT| 10000|
|    Sunny|Data Science| 10000|
|Sudhanshu|Data Science| 20000|
+---------+-----------+------+
```

Alternatively orderBy can also be used. (S)ort based on single column)

In [26]:
```
df_pyspark.orderBy("salary").show()
```

```
+---------+-----------+------+
|     Name| Departments|salary|
+---------+-----------+------+
|    Sunny|    Big Data|  2000|
|   Mahesh|Data Science|  3000|
|   Mahesh|    Big Data|  4000|
|    Krish|    Big Data|  4000|
|    Krish|         IOT|  5000|
|Sudhanshu|    Big Data|  5000|
|    Krish|Data Science| 10000|
|Sudhanshu|         IOT| 10000|
|    Sunny|Data Science| 10000|
```

```
|Sudhanshu|Data Science| 20000|
+---------+------------+------+
```

In [39]:

```
An error occurred: An error occurred while calling o366.showString.
: org.apache.spark.SparkException: Job aborted due to stage failure: Task 0 in stage 69.0 failed 1 times, most recent failur
e: Lost task 0.0 in stage 69.0 (TID 68) (LAPTOP-42FDQGJC executor driver): org.apache.spark.SparkException: Python worker fai
led to connect back.
        at org.apache.spark.api.python.PythonWorkerFactory.createSimpleWorker(PythonWorkerFactory.scala:203)
        at org.apache.spark.api.python.PythonWorkerFactory.create(PythonWorkerFactory.scala:109)
        at org.apache.spark.SparkEnv.createPythonWorker(SparkEnv.scala:124)
        at org.apache.spark.api.python.BasePythonRunner.compute(PythonRunner.scala:174)
        at org.apache.spark.api.python.PythonRDD.compute(PythonRDD.scala:67)
        at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:364)
        at org.apache.spark.rdd.RDD.iterator(RDD.scala:328)
        at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
        at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:364)
        at org.apache.spark.rdd.RDD.iterator(RDD.scala:328)
        at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
        at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:364)
        at org.apache.spark.rdd.RDD.iterator(RDD.scala:328)
        at org.apache.spark.rdd.MapPartitionsRDD.compute(MapPartitionsRDD.scala:52)
        at org.apache.spark.rdd.RDD.computeOrReadCheckpoint(RDD.scala:364)
```

In [ ]:

In [ ]:

databricks

abhishek7621cse@gmail.com

Pyspark3 Python

File Edit View Run Help Last edit was 25 minutes ago New cell UI: OFF

Run all Terminated Share Publish

Cmd 1

```python
from pyspark.sql import SparkSession

spark =SparkSession.builder.appName("Practice").getOrCreate()

df_pyspark= spark.read.csv("/FileStore/tables/Bibadata-2.csv",header=True, inferSchema=True)

df_pyspark.show()
```

(3) Spark Jobs

df_pyspark: pyspark.sql.dataframe.DataFrame = [Product: string, Age: integer ... 7 more fields]

```
| TM195| 19|Female|       14|    Partnered|    4|      3| 30699|   66|
| TM195| 19|  Male|       12|       Single|    3|      3| 32973|   85|
| TM195| 20|  Male|       13|    Partnered|    4|      2| 35247|   47|
| TM195| 20|Female|       14|    Partnered|    3|      3| 32973|   66|
| TM195| 21|Female|       14|    Partnered|    3|      3| 35247|   75|
| TM195| 21|  Male|       13|       Single|    3|      3| 32973|   85|
| TM195| 21|  Male|       15|       Single|    5|      4| 35247|  141|
| TM195| 21|Female|       15|    Partnered|    2|      3| 37521|   85|
| TM195| 22|  Male|       14|       Single|    3|      3| 36384|   85|
| TM195| 22|Female|       14|    Partnered|    3|      2| 35247|   66|
| TM195| 22|Female|       16|       Single|    4|      3| 36384|   75|
| TM195| 22|Female|       14|       Single|    3|      3| 35247|   75|
| TM195| 23|  Male|       16|    Partnered|    3|      1| 38658|   47|
| TM195| 23|  Male|       16|    Partnered|    3|      3| 40932|   75|
| TM195| 23|Female|       14|       Single|    2|      3| 34110|  103|
| TM195| 23|  Male|       16|    Partnered|    4|      3| 39795|   94|
| TM195| 23|Female|       16|       Single|    4|      3| 38658|  113|
| TM195| 23|Female|       15|    Partnered|    2|      2| 34110|   38|
+-------+---+------+---------+-------------+-----+-------+------+-----+
only showing top 20 rows
```

Command took 2.76 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:18:49 PM on pyspark

Cmd 2

```python
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),
        (('Jenis'),'1980-02-17','F',1200)]
columns = ['Name','DOB','Gender','salary']
df = spark.createDataFrame(data=data,schema=columns)
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|
+------+----------+------+------+
```

Command took 1.85 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:39:37 PM on pyspark

Cmd 3

```python
# Joins

```

Command took 0.13 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:43:13 PM on pyspark

Cmd 4

```python
from pyspark.sql import SparkSession
spark =SparkSession.builder.appName('pyspark - example join').getOrCreate()
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","F",2000),(5,"Brown",2,"2010",
"40","",-1),(6, "Brown", 2, "2010","50","",-1)]
empColumns = ["emp_id","name","superior_emp_id","year_joined", "emp_dept_id","gender","salary"]

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()
```

(6) Spark Jobs

empDF: pyspark.sql.dataframe.DataFrame = [emp_id: long, name: string ... 5 more fields]

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|       2010|        10|     M|  1000|
|     4|   Jones|             2|       2005|        10|     F|  2000|
|     5|   Brown|             2|       2010|        40|      |    -1|
```

```
|     6|   Brown|              2|       2010|         50|      |    -1|
+------+--------+---------------+-----------+-----------+------+------+

root
 |-- dept_name: string (nullable = true)
 |-- dept_id: long (nullable = true)

+---------+-------+
|dept_name|dept_id|
+---------+-------+
|  Finance|     10|
|Marketing|     20|
|    Sales|     30|
|       IT|     40|
+---------+-------+
```

Command took 1.98 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:47:17 PM on pyspark

Cmd 5

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

▸ (3) Spark Jobs

```
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|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|     F|  2000|  Finance|     10|
|     2|    Rose|              1|       2010|         20|     M|  4000|Marketing|     20|
|     5|   Brown|              2|       2010|         40|      |    -1|       IT|     40|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
```

Command took 4.70 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:47:54 PM on pyspark

Cmd 6

```
1  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"outer").show()
2  #Or
3  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"full").show()
4  #Or
5  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"fullouter").show()
```

▸ (9) Spark Jobs

```
|     1|   Smith|             -1|       2018|         10|     M|  3000|  Finance|     10|
|     3|Williams|              1|       2010|         10|     M|  1000|  Finance|     10|
|     4|   Jones|              2|       2005|         10|     F|  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|      |    -1|       IT|     40|
|     6|   Brown|              2|       2010|         50|      |    -1|     null|   null|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+

+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|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|     F|  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|      |    -1|       IT|     40|
|     6|   Brown|              2|       2010|         50|      |    -1|     null|   null|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
```

Command took 5.18 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:48:19 PM on pyspark

Cmd 7

```
1  # Left Join
2  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"left").show()
3  #Or
4  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"leftouter").show()
```

▸ (12) Spark Jobs

```
|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|     F|  2000|  Finance|     10|
|     5|   Brown|              2|       2010|         40|      |    -1|       IT|     40|
|     6|   Brown|              2|       2010|         50|      |    -1|     null|   null|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+

+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|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|     F|  2000|  Finance|     10|
|     5|   Brown|              2|       2010|         40|      |    -1|       IT|     40|
|     6|   Brown|              2|       2010|         50|      |    -1|     null|   null|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
```

Command took 4.07 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:48:53 PM on pyspark

Cmd 8

```
1  # Right Join
2  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"right").show()
3  #Or
```

```
4  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"rightouter").show()
5
```

▸ (12) Spark Jobs

```
|emp_id|    name|superior_emp_id|year_joined|emp_dept_id|gender|salary|dept_name|dept_id|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|     4|   Jones|              2|       2005|         10|     F|  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|      |    -1|       IT|     40|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+


+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|emp_id|    name|superior_emp_id|year_joined|emp_dept_id|gender|salary|dept_name|dept_id|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
|     4|   Jones|              2|       2005|         10|     F|  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|      |    -1|       IT|     40|
+------+--------+---------------+-----------+-----------+------+------+---------+-------+
```

Command took 3.91 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:49:16 PM on pyspark

Cmd 9

```
1  # Left semi Join
2  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"leftsemi").show()
3
```

▸ (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|     F|  2000|
|     2|    Rose|              1|       2010|         20|     M|  4000|
|     5|   Brown|              2|       2010|         40|      |    -1|
+------+--------+---------------+-----------+-----------+------+------+
```

Command took 1.65 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:49:30 PM on pyspark

Cmd 10

Python

```
1  # Leftanti Join
2  empDF.join(deptDF,empDF.emp_dept_id ==  deptDF.dept_id,"leftanti").show()
3
```

▸ (4) Spark Jobs

```
+------+-----+---------------+-----------+-----------+------+------+
|emp_id| name|superior_emp_id|year_joined|emp_dept_id|gender|salary|
+------+-----+---------------+-----------+-----------+------+------+
|     6|Brown|              2|       2010|         50|      |    -1|
+------+-----+---------------+-----------+-----------+------+------+
```

Command took 1.59 seconds -- by abhishek7621cse@gmail.com at 2/9/2024, 3:49:45 PM on pyspark

Cmd 11

```
1
```

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