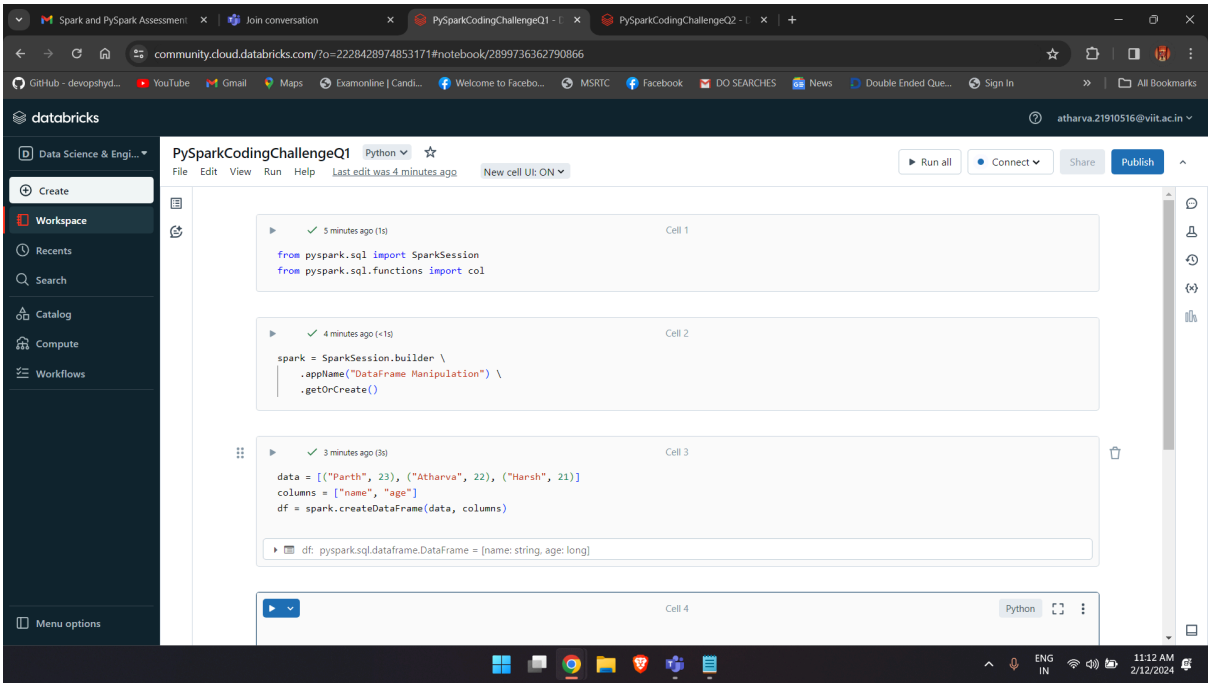
Name : Parth Nandedkar
Date : 12 Feb 2024
Topics : PySpark
Batch : Data Engineering Batch-1

Q1. Execute Manipulating, Dropping, Sorting, Aggregations, Joining, GroupeBy DataFrames.

In Apache Spark, a DataFrame is a distributed collection of data organised into named columns, similar to a table in a relational database or a spreadsheet in a spreadsheet application.

We can perform different operations on it using Spark.

For now I am using DataBricks to execute the commands.



In the above screenshot I've imported a module SparkSession to generate a Spark Session so that further activities can be done.

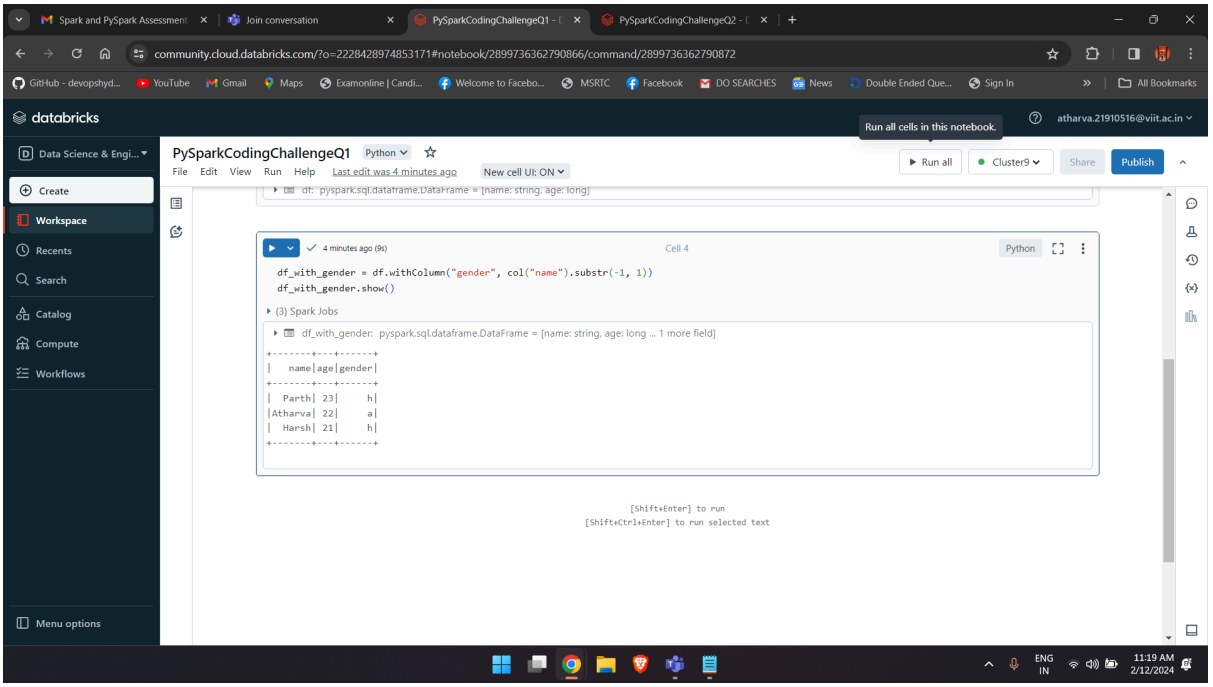
And for performing the rest of the applications the sql.functions col module is imported.

Data Frame named as **df** is created by using method createDataFrame()

List named as **data** is given as parameter and list named as **columns**
Is also given as a parameter to that function and the data frame is created.

Adding Column :

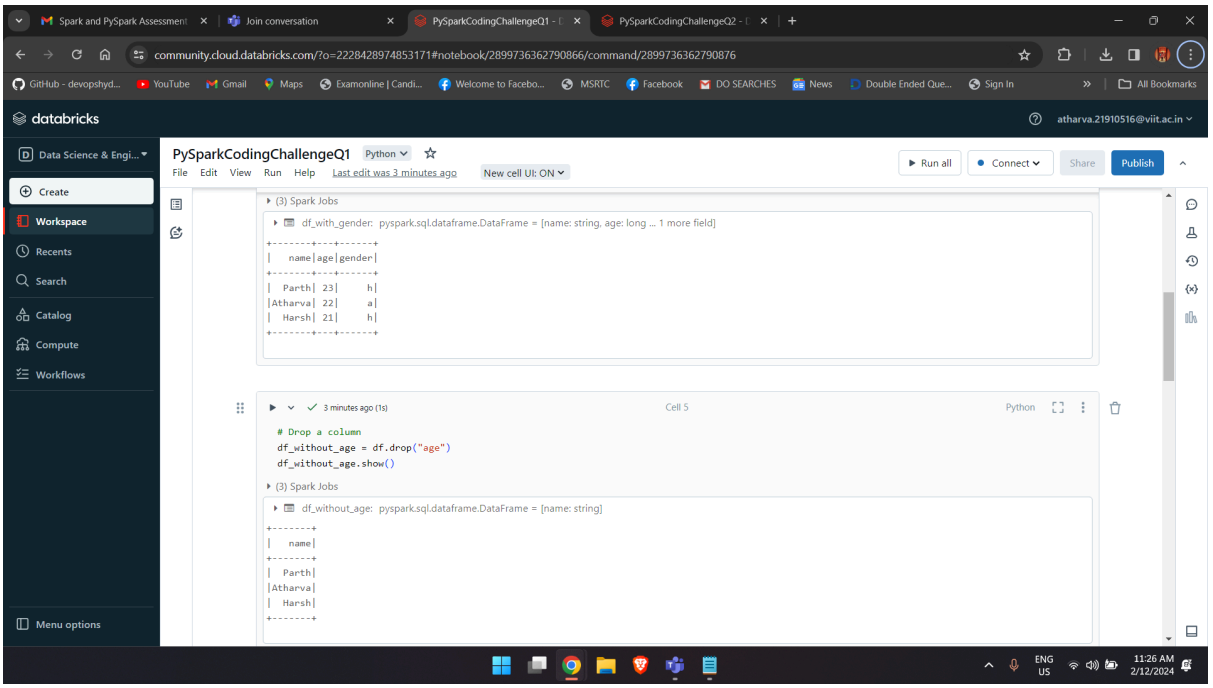
As there were only 2 columns **name** and **age** we are adding a new column named gender**.**



By using the withColumn() method we created a new column gender and we used show() method to see if changes are reflected or not.

Dropping Column :

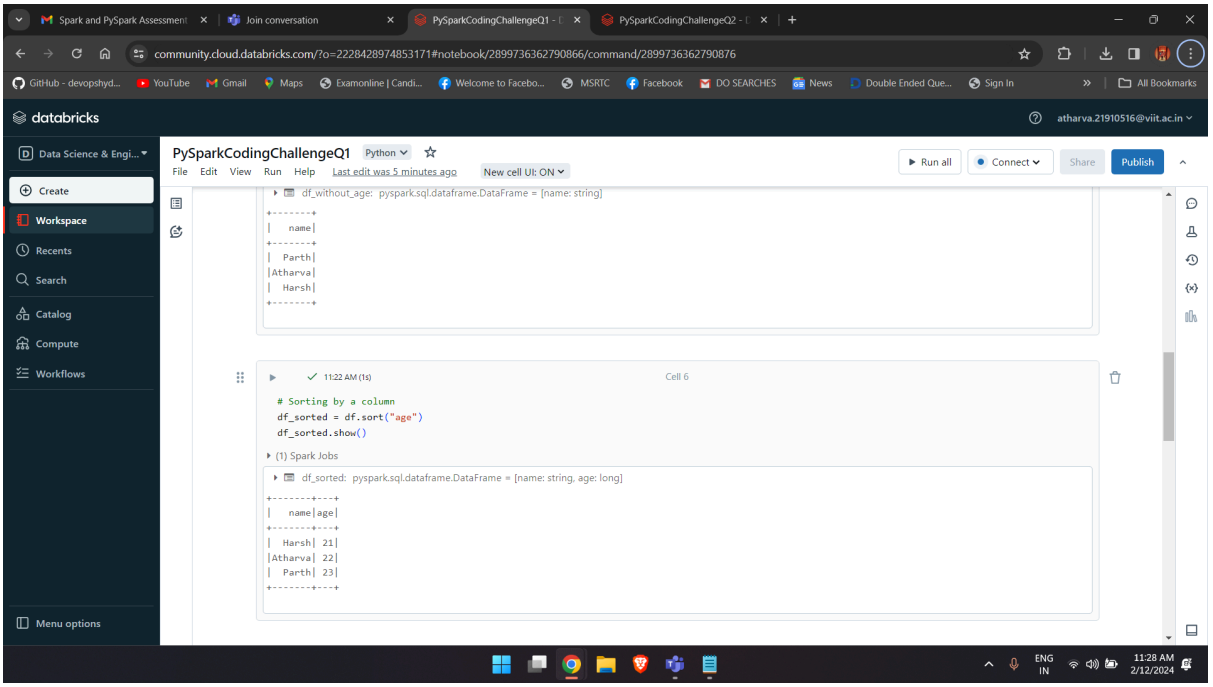
Columns of unnecessary data can be dropped as per the application need here to demonstrate we can drop age column.



By using **drop()** method we dropped **age** column and to see weather chanes are reflected or not we are using **show()** method.

Sorting Data :

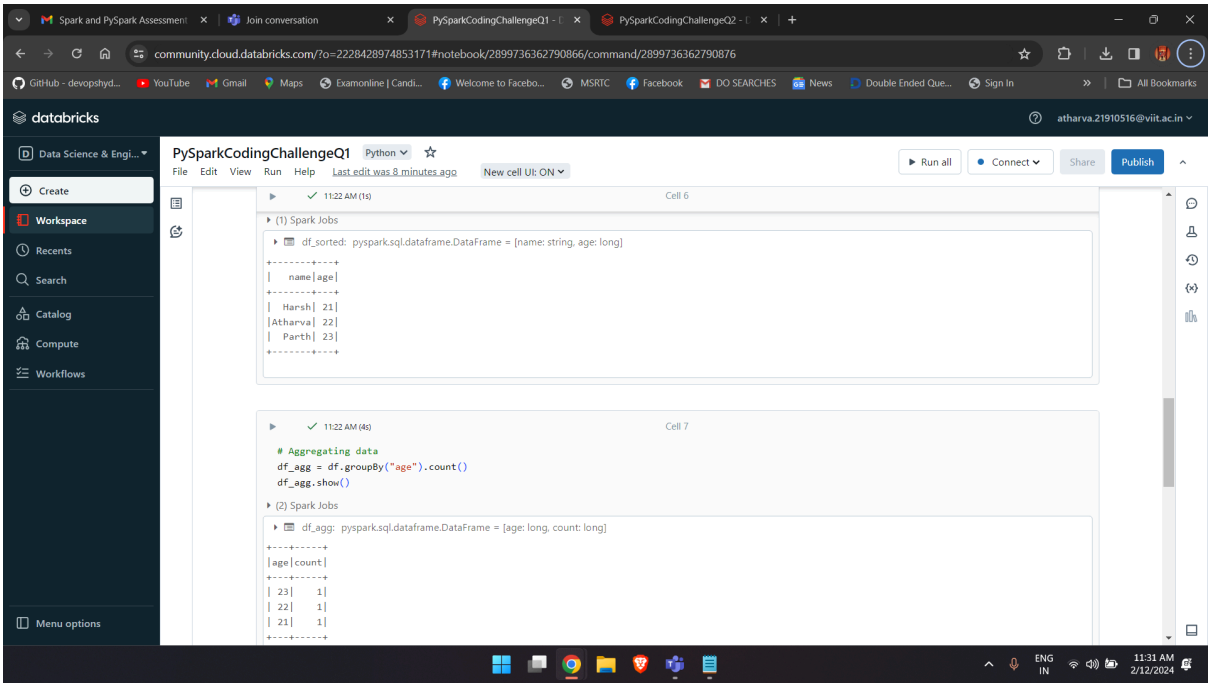
According to application need we can sort data in the order by giving a particular column as reference as names cannot be sorted we can use age on the above example :



According to age the data is sorted as **Harsh has age 21**, **Atharva 22** and **Parth 23** accordingly the data is sorted as we used **sort()** method on data object.

Aggregating Data And Group By :

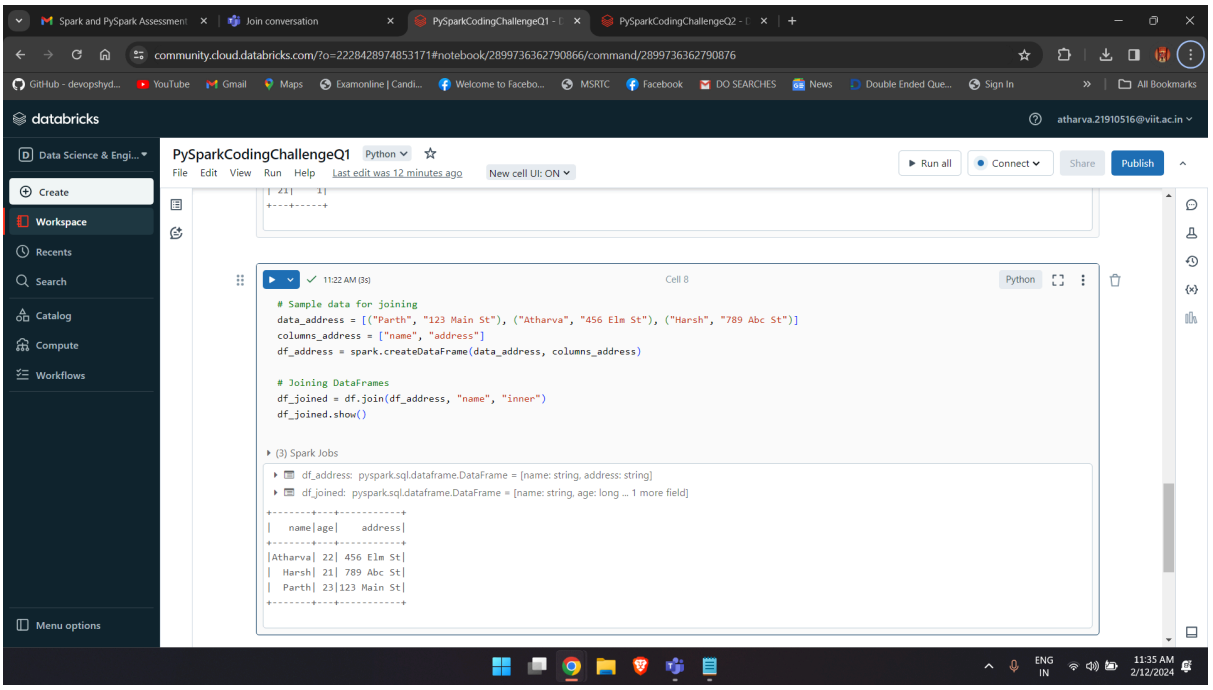
According to data we can distribute data in particular sets by using aggregations here on count method we are getting data to show how many people there are of same age .



As there are 3 age groups 21,22,23 and each contains 1 person thats why the output shows 21 -1, 22-1, 23 -1.
Which is the right output.

Using Joins on data :

Joins can be used to get desired data from two different datasets combined based on the common condition so that the operation performing could be easy and data storing in different datasets to achieve normalisation can be achieved .



Here in the above example we added one more data frame which includes the same names but the second column is of Address.

As Names are common by using **join()** method we can connect df1 and df2.

As all the 3 names are common all the names , ages and address accordingly are displayed.

Here we used **inner** join as we need to put the right address to the right person so only common values will be joined.