DF_Pracrtice_21L-7289

April 3, 2022

BIG DATA ANALYTICS

PySpark DataFrame Practice

0.0.1 You are provided dataset "Movies.csv" that contains information about 1600 movies with properties such as year, length, main actor and actress, director and popularity.

0.0.2 Load the given dataset into Spark Data-Frames and answer the following queries using Data Frame functions only. You are not allowed to write the SparkSQL queries.

---

[1]:
```
import findspark
```

[2]:
```
findspark.init()
```

[3]:
```
import pyspark
```

[4]:
```
from pyspark.sql import SparkSession
```

[88]:
```
spark = ss.builder.appName('PYSPARK-DATAFRAME').getOrCreate()
```

[89]:
```
df = spark.read.csv('Movies.csv', header=True, inferSchema=True)
```

[90]:
```
df.printSchema()
```

```
root
 |-- Year: integer (nullable = true)
 |-- Length: integer (nullable = true)
 |-- Title: string (nullable = true)
 |-- Genre: string (nullable = true)
 |-- Actor: string (nullable = true)
 |-- Actress: string (nullable = true)
 |-- Director: string (nullable = true)
 |-- Popularity: integer (nullable = true)
 |-- Awards: string (nullable = true)
 |-- Image: string (nullable = true)
```

[107]:
```
df.show(5)
```

```
+----+------+--------------------+------+-----------------+-------------+-------
--------+----------+------+----------------+
|Year|Length|               Title| Genre|            Actor|      Actress|
Director|Popularity|Awards|           Image|
+----+------+--------------------+------+-----------------+-------------+-------
--------+----------+------+----------------+
|1990|   111|Tie Me Up! Tie Me…|Comedy|  BanderasAntonio|AbrilVictoria|
AlmodóvarPedro|        68|    No|NicholasCage.png|
|1991|   113|          High Heels|Comedy|       BoséMiguel|AbrilVictoria|
AlmodóvarPedro|        68|    No|NicholasCage.png|
|1983|   104|        Dead ZoneThe|Horror|WalkenChristopher|
AdamsBrooke|CronenbergDavid|        79|    No|NicholasCage.png|
|1979|   122|                Cuba|Action|      ConnerySean|  AdamsBrooke|
LesterRichard|         6|    No| seanConnery.png|
|1978|    94|      Days of Heaven| Drama|      GereRichard|  AdamsBrooke|
MalickTerrence|        14|    No|NicholasCage.png|
+----+------+--------------------+------+-----------------+-------------+-------
--------+----------+------+----------------+
only showing top 5 rows
```

1. Find the title, year, and director of action films that won an award.

[109]:
```
df.filter("Awards = 'Yes'").select(['Title','Year','Director']).show(5)
```

```
+--------------------+----+--------------+
|               Title|Year|      Director|
+--------------------+----+--------------+
| Fanny and Alexander|1982|BergmanIngmar|
|     A Man & a Woman|1966|LelouchClaude|
|Un Hombre y una M…|1966|LelouchClaude|
|   Official StoryThe|1985|    PuenzoLuiz|
|   Wild Strawberries|1957|BergmanIngmar|
+--------------------+----+--------------+
only showing top 5 rows
```

2. For each award-winning actor, find the movies he acted it. Print the names of the movies and the director of the movie.

[108]:
```
df.filter("Awards = 'Yes'").select(['Actor','Title','Director']).show(5)
```

```
+--------------------+--------------------+--------------+
|               Actor|               Title|      Director|
+--------------------+--------------------+-------------+
|       AhlstedtBörje| Fanny and Alexander|BergmanIngmar|
|TrintignantJean-L…|      A Man & a Woman|LelouchClaude|
|TrintignantJean-L…|Un Hombre y una M…|LelouchClaude|
|       AlterioHector|    Official StoryThe|    PuenzoLuiz|
|      SjöströmVictor|    Wild Strawberries|BergmanIngmar|
+--------------------+--------------------+-------------+
only showing top 5 rows
```

3. Find the top 10 most popular movies that did not win an award

```
[146]: df.select("Title","Popularity","Awards").filter(df["Awards"]=='No').
         ↪sort(df["Popularity"].desc()).show(10)
```

```
+--------------------+----------+------+
|               Title|Popularity|Awards|
+--------------------+----------+------+
|         Let It Ride|        88|    No|
|       Great RaceThe|        88|    No|
|      New Year's Day|        88|    No|
|        Final Notice|        88|    No|
|   Fellini Satyricon|        88|    No|
|Guilty by Suspicion|        88|    No|
|     Time MachineThe|        88|    No|
|           Raw Nerve|        88|    No|
|Long Voyage HomeThe|        88|    No|
|           Class Act|        88|    No|
+--------------------+----------+------+
only showing top 10 rows
```

4. Find the 10 least popular movies that were released before 1980.

```
[145]: df.select("Title","Popularity","Year").filter((df["Year"]<1980) &
                                                       (df["Popularity"].isNotNull()) ).
         ↪sort(df["Popularity"].asc()).show(10)
```

```
+--------------------+----------+----+
|               Title|Popularity|Year|
+--------------------+----------+----+
|             Airport|         0|1970|
|       Anna Christie|         0|1930|
|             Shalako|         0|1968|
|    Tales of Tomorrow|         0|1953|
|   Shout at the Devil|         0|1976|
|           Holocaust|         1|1978|
|            Stavisky|         1|1974|
|    Anderson TapesThe|         1|1971|
|          Indiscreet|         1|1958|
|Law of the Golden…|         1|1949|
+--------------------+----------+----+
only showing top 10 rows
```

5. Find the average length of the movies of each genre.

[144]:
```
df.groupBy("Genre").avg("Length").show()
```

```
+---------------+------------------+
|          Genre|       avg(Length)|
+---------------+------------------+
|          Crime|              66.0|
|        Romance|             127.0|
|      Adventure|             119.0|
|           null|             120.5|
|          Drama|113.30455259026688|
|            War|         116.90625|
|        Fantasy|             102.0|
|        Mystery|103.00990099009901|
|          Music|100.48780487804878|
|Science Fiction|106.47368421052632|
|         Horror| 93.92727272727272|
|          Short|              40.0|
|        Western|  93.0091743119266|
|         Comedy| 96.50540540540541|
|         Action|             104.5|
|       Westerns|             124.8|
+---------------+------------------+
```

6. Find the actor and actress pair who has acted in more than three Comedies together

[305]:
```
df2 = df.filter( (df["Actor"].isNotNull())
          & (df["Actress"].isNotNull())  ).where(df["Genre"]==
                                        "Comedy").groupBy("Actor",
                                                    "Actress").
 ↪count()
```

[315]:
```
df2.withColumnRenamed("count", "Comedy Movie count").where("count >=3").show()
```

```
+------------+----------------+------------------+
|       Actor|         Actress|Comedy Movie count|
+------------+----------------+------------------+
|TracySpencer|HepburnKatharine|                 6|
|  AllenWoody|     KeatonDiane|                 5|
+------------+----------------+------------------+
```

7. Find the names of actors who acted in movies of both 'Comedy' and 'Drama' Genre.

[322]:
```
df.select(df["Actor"]).filter((df["Genre"]=="Comedy")  &␣
↪(df["Genre"]=="Drama")).show(10)
```

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

8. Find the names of actors who acted in movies of both 'Comedy' or 'Drama' Genre.

[321]:
```
df.select(df["Actor"]).filter((df["Genre"]=="Comedy")  |␣
↪(df["Genre"]=="Drama")).show(10)
```

```
+--------------------+
|               Actor|
+--------------------+
|      BanderasAntonio|
|          BoséMiguel|
|         GereRichard|
|     BergenRobert D.|
|  LambertChristopher|
|     DepardieuGérard|
|        AhlstedtBörje|
|          TognazziUgo|
|TrintignantJean-L...|
|TrintignantJean-L...|
+--------------------+
only showing top 10 rows
```

9. Find the names of actors who did not act in any 'Comedy'.

[323]:
```
df.select("Actor").filter(~(df["Genre"]=="Comedy")).show(10)
```

```
+------------------+
|             Actor|
+------------------+
| WalkenChristopher|
|       ConnerySean|
|       GereRichard|
|        MooreRoger|
|      ConnorsChuck|
|   BergenRobert D.|
|LambertChristopher|
|   DepardieuGérard|
|      AhlstedtBörje|
|       TognazziUgo|
+-------------------+
only showing top 10 rows
```

[326]:
```
### Rechecking our result...

df.select("Genre").filter(df["Actor"]=="WalkenChristopher").show(10)
```

```
+--------+
|   Genre|
+--------+
| Horror|
|Mystery|
+--------+
```

10. Find each actor, find the mean, max, and min ranking of his movies.

[334]:
```
import pyspark.sql.functions as func
df.groupBy("Actor").agg(func.mean("Popularity").alias("Mean Movie Ranking"),
                        func.max("Popularity").alias("Max Movie Ranking"),
                        func.min("Popularity").alias("Min Movie Ranking")).
 ↪show(8)
```

```
+---------------+------------------+-----------------+-----------------+
|          Actor|Mean Movie Ranking|Max Movie Ranking|Min Movie Ranking|
+---------------+------------------+-----------------+-----------------+
|     BoséMiguel|              68.0|               68|               68|
|   CottenJoseph|              58.0|               74|               32|
|       BrownTom|              77.0|               77|               77|
|     DillonMatt|               7.5|               11|                4|
|  KeatonMichael|              59.0|               59|               59|
|ShimuraTakashi|              36.0|               36|               36|
|   LintDerek De|              71.0|               71|               71|
|    WillisBruce|              48.0|               76|                7|
+---------------+------------------+-----------------+-----------------+
only showing top 8 rows
```

11. List the number of movies released in each decade starting from the 1960's.

[373]:
```
df2 = df.select("Year").groupBy("Year").count().where("Year>=1960").
 ↪sort(df["Year"].asc())
```

[393]:
```
schema = [  {"Decades":'1960~1969',"Movie Release Count":df2.
 ↪filter((df["Year"]>='1960') &(df["Year"]<'1970') ).count()},
            {"Decades":'1970~1979',"Movie Release Count":df2.
 ↪filter((df["Year"]>='1970') &(df["Year"]<'1980') ).count()},
            {"Decades":'1980~1989',"Movie Release Count":df2.
↪filter((df["Year"]>='1980') &(df["Year"]<'1990') ).count()},
            {"Decades":'1990~2000',"Movie Release Count":df2.
↪filter((df["Year"]>='1990') &(df["Year"]<'2000') ).count()},
         ]

df3 = spark.createDataFrame(schema)
df3.show()
```

```
+---------+-------------------+
|  Decades|Movie Release Count|
+---------+-------------------+
|1960~1969|                 10|
|1970~1979|                 10|
|1980~1989|                 10|
|1990~2000|                  6|
+---------+-------------------+
```

12. Find the number of movies released each year

[396]:
```
df.select("Year").groupBy("Year").count().sort(df["Year"].asc()).show(10)
```

```
+----+-----+
|Year|count|
+----+-----+
|1920|    1|
|1923|    1|
|1924|    3|
|1925|    1|
|1926|    4|
|1927|    3|
|1928|    5|
|1929|    5|
|1930|    3|
|1931|    9|
+----+-----+
only showing top 10 rows
```

13. Find the number of movies released in each year of each genre. Consider only the movies with a length greater than 100 minutes.

[406]:
```
df.select("Year","Genre").where(df["Length"]>100).groupBy("Year","Genre").
↪count().sort(df["Year"].asc()).show(10)
```

```
+----+---------------+-----+
|Year|          Genre|count|
+----+---------------+-----+
|1920|           Drama|    1|
|1924|           Drama|    2|
|1925|           Drama|    1|
|1926|          Action|    1|
|1926|Science Fiction|    1|
|1926|           Drama|    1|
|1928|             War|    1|
|1928|           Drama|    2|
|1929|           Drama|    1|
|1931|         Western|    2|
+----+---------------+-----+
only showing top 10 rows
```

14. Sort the movie's release before 1990 by the title.

[412]:
```
df.select("Title", "Year").sort(df["Title"]).where("Year<1990").show(20)
```

```
+--------------------+----+
|               Title|Year|
+--------------------+----+
|2001: A Space Ody…|1968|
|             48 Hrs.|1982|
|               8 1/2|1963|
|A Big Hand for th…|1966|
|   A Child Is Waiting|1962|
|A Chorus LineThe …|1985|
|   A Clockwork Orange|1971|
|A Coeur Joie(Head…|1967|
|    A Cry in the Dark|1988|
|   A Dry White Season|1989|
|       A Fine Madness|1966|
|  A Fish Called Wanda|1988|
|A Fistful of Dollars|1964|
|      A Guy Named Joe|1943|
|     A Lesson in Love|1954|
|A Little Night Music|1977|
|      A Man & a Woman|1966|
|A Man & a Woman: …|1986|
|A Man for All Sea…|1966|
|     A Matter of Time|1976|
+--------------------+----+
only showing top 20 rows
```

15. Find the movies with long titles. A movie title is considered long if it is greater than 50 alphabets.

[422]:
```
from pyspark.sql.functions import length
df.select("Title").where(length(df["Title"]) > 50).show()
```

```
+--------------------+
|               Title|
+--------------------+
|Fawlty TowersGour…|
+--------------------+
```

[423]:
```
len("Fawlty TowersGourmet NightWaldorf Salad & The Kipper & the Corpse")
```

[423]:
```
65
```

[ ]:
```
```

[ ]:
```
```

[ ]:
```
```