Questions for Assignment 2

You have been given (on Moodle) a sample dataset of historical events, and a Spark program that implements a group by and count on a data set of historical events.

You can use this program as a template for the questions below.

Write a spark program that implements the following four queries as separate functions, and a main program that takes as argument an integer from 1 to 4, and executes the relevant query from the below:

> 1.Given an entity like Gandhi, Greece, etc. create a dataset of [entity, category1, category 2, count] considering events where entity occurs in the description

> 2.As above, given a file with entities, one per line. Assume the entities are given in a file called entities.

> 3.Find the count of all instances in each year (consider only events with granularity year)

> 4.For each year, find the rank of the year if sorted in descending order by number of events in the year.

In all cases the result should be computed using spark and output to console as in the template program.

```
!pip install pyspark==3.0.1 py4j==0.10.9
```

```
Looking in indexes: https://pypi.org/simple, https://us-python.pkg.dev/colab-wheels/public/simple/
Collecting pyspark==3.0.1
  Downloading pyspark-3.0.1.tar.gz (204.2 MB)
     |████████████████████████████████| 204.2 MB 31 kB/s
Collecting py4j==0.10.9
  Downloading py4j-0.10.9-py2.py3-none-any.whl (198 kB)
     |████████████████████████████████| 198 kB 66.8 MB/s
Building wheels for collected packages: pyspark
  Building wheel for pyspark (setup.py) ... done
  Created wheel for pyspark: filename=pyspark-3.0.1-py2.py3-none-any.whl size=204612243 sha256=3d694f1a41270084c9b8b608d887ad7c0c1beb456e65c03b1efcf6765e638a9a
  Stored in directory: /root/.cache/pip/wheels/5e/34/fa/b37b5cef503fc5148b478b2495043ba61b079120b7ff379f9b
Successfully built pyspark
Installing collected packages: py4j, pyspark
Successfully installed py4j-0.10.9 pyspark-3.0.1
```

```
from pyspark.sql import SparkSession

spark = SparkSession.builder\
        .master("local[*]")\
        .appName('PySpark_Tutorial')\
        .getOrCreate()
```

Loading Data

```
# Reading JSON file
json_file = '/history_english.json'
data = spark.read.json(json_file)
data.show(10)
```

```
+---------+--------------+----+--------------------+-----------+----+
|category1|     category2|date|         description|granularity|lang|
+---------+--------------+----+--------------------+-----------+----+
| By place|        Greece|-300|Pilgrims travel t...|       year|  en|
| By place|         Egypt|-300|Pyrrhus, the King...|       year|  en|
| By place|         Egypt|-300|Ptolemy concludes...|       year|  en|
| By place|Seleucid Empire|-300|Seleucus founds t...|       year|  en|
| By place|Seleucid Empire|-300|After the death o...|       year|  en|
| By place|         India|-300|The central texts...|       year|  en|
| By topic|           Art|-300|In Pella (in Mace...|       year|  en|
| By place| Roman Republic|-299|The Samnites, sei...|       year|  en|
| By place|         China|-299|The state of Qin ...|       year|  en|
| By place|         China|-299|King Wuling of Zh...|       year|  en|
+---------+--------------+----+--------------------+-----------+----+
only showing top 10 rows
```

Question 1

```
import pandas as pd
from pyspark.sql.functions import *

entity = 'Greece'

#removing rows which has category1==NULL and/or category2=NULL and taking this output in exdata for que1 and 2
data = data.where(data.category1.isNotNull())
data = data.where(data.category2.isNotNull())

exdata = data[data['description'].contains(entity)]   #finding entity in desciprion of data
exdata = exdata.withColumn('entity', lit(entity))       #adding entity column
exdata = exdata.groupby('entity','category1','category2').count()       #groupby and count

#storing output into question1-output.csv
exdata.toPandas().to_csv('question1-output.csv')

#print(exdata)
exdata.show(20)
```

```
+------+---------+--------------------+-----+
|entity|category1|           category2|count|
+------+---------+--------------------+-----+
|Greece| By place|      Roman Republic|   12|
|Greece| By place|          Asia Minor|    1|
|Greece| By place|              Greece|   24|
```

```
|Greece| By place|     Seleucid Empire|    4|
|Greece| By place|    Byzantine Empire|    4|
|Greece| By place|Eastern Roman Empire|    1|
|Greece| By place|        Roman Empire|   14|
|Greece| By place|              Europe|    5|
|Greece| By topic|           Religion|    2|
|Greece| By place|      =Mediterranean|    1|
+------+---------+--------------------+-----+
```

Question 2

```
from pyspark.sql.functions import *
from pyspark.sql.functions import col, lit

with open("/entities.txt") as f:
    entities = f.readlines()

# new line characters removal
entities = [x.strip() for x in entities]
#print(entities)

finaldata = []
for entity in entities:
  exdata = data[data['description'].contains(entity)]
  exdata = exdata.withColumn('entity', lit(entity))
  exdata = exdata.groupby('entity','category1','category2').count()
  #exdata.show()
  exdata.toPandas().to_csv('question2-output.csv', mode='a', index=False, header=False)
  finaldata.append(exdata.show())

#storing output into question2_ut.csv
#finaldata.toPandas().to_csv('question2-output.csv')


print(finaldata)
```

```
+----------+---------+------------+-----+
|Christians| By place|      Persia|    1|
|Christians| By place|      Africa|    1|
|Christians|  By area|      Africa|    1|
|Christians| By topic|    Religion|   20|
|Christians| By place|      Europe|    6|
|Christians|  By area|      Europe|    5|
|Christians| By place|        Asia|    2|
|Christians| By place|Roman Empire|   24|
+----------+---------+------------+-----+

+------+-----------+-----------------+-----+
|entity|  category1|        category2|count|
+------+-----------+-----------------+-----+
|Sicily|   By place|            Egypt|    1|
```

```
|Sicily|   By place|           Sicily|   11|
|Sicily|     Africa|The Eighth Crusade|    1|
|Sicily|   By place|    Roman Republic|   41|
|Sicily|By location|            Europe|    1|
|Sicily|   By place|  =Southern Europe|    2|
|Sicily|   By topic|  War and politics|    2|
|Sicily|    By area|    =Mediterranean|    1|
|Sicily|   By topic|          Religion|    1|
|Sicily|    By area|            Europe|   13|
|Sicily|   By place|            Africa|    8|
|Sicily|   By place|          Carthage|    2|
|Sicily|   By place|      Roman Empire|   11|
|Sicily|   By place|            Europe|   23|
|Sicily|   By place|     = Middle East|    1|
|Sicily|   By place|    =Mediterranean|    1|
|Sicily|    By area|            Africa|    2|
|Sicily|   By place|            Greece|    1|
|Sicily|   By place|  Byzantine Empire|    4|
+------+-----------+------------------+-----+

+-------+---------+---------+-----+
| entity|category1|category2|count|
+-------+---------+---------+-----+
|Fa-Hien| By topic| Religion|    2|
|Fa-Hien| By place|     Asia|    3|
+-------+---------+---------+-----+

+------+---------+------------------+-----+
|entity|category1|         category2|count|
+------+---------+------------------+-----+
| India| By place|            Greece|    1|
| India| By topic|               Art|    9|
| India| By Place|              Asia|    1|
| India| By topic|  Arts and sciences|    5|
| India| By place|            Europe|    1|
| India| By topic|Arts and technology|    1|
| India| By place|             Persia|    3|
| India|  By area|              Asia|    4|
| India| By place|       Roman Empire|    5|
| India| By place|           Americas|    1|
| India| By topic|           Medicine|    2|
| India| By topic|         Inventions|    1|
| India| By topic|           Religion|   13|
| India|   Europe|              Asia|    1|
```

Question 3

```
import pyspark
from pyspark.sql.functions import year
from pyspark.sql.functions import to_date
from pyspark.sql import SparkSession
from pyspark.sql.types import IntegerType
import pandas as pd
```

```
tempdata=[]                      #temporary list to store the data
for i in data.collect():
   tempdata.append(tuple(i))

dates = []                      #extracted the date
for i in range(len(tempdata)):
  tempdate = tempdata[i][2]
  dates.append(tempdate)

year = []                       #extracted the year from the dates
for i in range(len(dates)):
  date = dates[i]
  curryear = ''
  for j in range(len(date)):
    if(date[j]!='/'):
      curryear = curryear + date[j]
    else:
      break
  year.append(curryear)

# appending this list of years to the original data

temp_year_df = pd.DataFrame(year)
tempdata_df = pd.DataFrame(tempdata)

tempdata_df['date'] = temp_year_df[0].values
#print(tempdata_df)

finaldata = tempdata_df.groupby('date')['date'].count().reset_index(name='counts')

#storing output into question3_output.csv
finaldata.to_csv('question3-output.csv')


finaldata
```

	date	counts
0	-1	4
1	-10	4
2	-100	13
3	-101	2

Question 4

```
#counts is the count of number events in the year (year is represented in date  column)
finaldata["Ranks"] = finaldata["counts"].rank()
finaldata.sort_values("counts", inplace = True, ascending=False)

#storing output into question4-output.csv
finaldata.to_csv('question4-output.csv')


finaldata
```

	date	counts	Ranks
386	1118	52	1477.0
399	1134	40	1476.0
299	1001	32	1475.0
374	1100	31	1474.0
298	1000	28	1473.0
...	...	...	...
615	210	1	27.5
582	183	1	27.5
60	-156	1	27.5
1257	799	1	27.5
738	322	1	27.5

1477 rows × 3 columns

Colab paid products - Cancel contracts here

✓ 21s completed at 5:53 PM