TPCH-Example-Solution-Notebook

October 5, 2020

[1]:
```
import sys
from operator import add
from itertools import combinations

import requests

from pyspark.sql.types import *

from pyspark.sql import functions as f
from pyspark.sql.functions import udf

from pyspark.sql.functions import lit
from pyspark import sql

sqlContext = sql.SparkSession.builder \
        .master("local") \
        .appName("Word Count") \
        .getOrCreate()
```

[2]:
```
# lineitems = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(sys.arg[1])
path="./tpch_tables_scale_0.1/"
# path is where you have the folder. It can be a distributed path like S3, gc␣
 ↪or hdfs

customer = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"customer.tbl")
order = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"orders.tbl")
lineitems = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"lineitem.tbl")
part = sqlContext.read.format('csv').options(header='true', inferSchema='true',␣
 ↪ sep ="|").load(path+"part.tbl")
supplier = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"supplier.tbl")
partsupp = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"partsupp.tbl")
```

```
region = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"region.tbl")
nation = sqlContext.read.format('csv').options(header='true',␣
 ↪inferSchema='true',  sep ="|").load(path+"nation.tbl")
```

[3]:
```
customer.show()
```

```
+-------+-----------------+--------------------+---------+--------------+-----
--+----------+--------------------+
|CUSTKEY|               NAME|             ADDRESS|NATIONKEY|
PHONE|ACCBATL|MKTSEGMENT|             COMMENT|
+-------+-----------------+--------------------+---------+--------------+-----
--+----------+--------------------+
|      1|Customer#000000001|   IVhzIApeRb ot,c,E|       15|25-989-741-2988|
711.56|  BUILDING|to the even, regu…|
|      2|Customer#000000002|XSTf4,NCwDVaWNe6t…|       13|23-768-687-3665|
121.65|AUTOMOBILE|l accounts. blith…|
|      3|Customer#000000003|        MG9kdTD2WBHm|
1|11-719-748-3364|7498.12|AUTOMOBILE| deposits eat sly…|
|      4|Customer#000000004|         XxVSJsLAGtn|
4|14-128-190-5944|2866.83| MACHINERY| requests. final,…|
|      5|Customer#000000005|KvpyuHCplrB84WgAi…|        3|13-750-942-6364|
794.47| HOUSEHOLD|n accounts will h…|
|      6|Customer#000000006|sKZz0CsnMD7mp4Xd0…|
20|30-114-968-4951|7638.57|AUTOMOBILE|tions. even depos…|
|      7|Customer#000000007|TcGe5gaZNgVePxU5k…|
18|28-190-982-9759|9561.95|AUTOMOBILE|ainst the ironic,…|
|      8|Customer#000000008|I0B10bB0AymmC, 0P…|
17|27-147-574-9335|6819.74|  BUILDING|among the slyly r…|
|      9|Customer#000000009|xKiAFTjUsCuxfeleN…|
8|18-338-906-3675|8324.07| FURNITURE|r theodolites acc…|
|     10|Customer#000000010|6LrEaV6KR6PLVcgl2…|
5|15-741-346-9870|2753.54| HOUSEHOLD|es regular deposi…|
|     11|Customer#000000011|PkWS 3HlXqwTuzrKg…|       23|33-464-151-3439|
-272.6|  BUILDING|ckages. requests …|
|     12|Customer#000000012|       9PWKuhzT4Zr1Q|
13|23-791-276-1263|3396.49| HOUSEHOLD| to the carefully…|
|     13|Customer#000000013|nsXQu0oVjD7PM659u…|
3|13-761-547-5974|3857.34|  BUILDING|ounts sleep caref…|
|     14|Customer#000000014|     KXkletMlL2JQEA |        1|11-845-129-3851|
5266.3| FURNITURE|, ironic packages…|
|     15|Customer#000000015|YtWggXoOLdwdo7b0y…|
23|33-687-542-7601|2788.52| HOUSEHOLD| platelets. regul…|
|     16|Customer#000000016| cYiaeMLZSMAOQ2 d0W,|
10|20-781-609-3107|4681.03| FURNITURE|kly silent courts…|
|     17|Customer#000000017|izrh 6jdqtp2eqdtb…|        2|12-970-682-3487|
6.34|AUTOMOBILE|packages wake! bl…|
```

```
|     18|Customer#000000018|3txGO AiuFux3zT0Z…|
6|16-155-215-1315|5494.43|  BUILDING|s sleep. carefull…|
|     19|Customer#000000019|uc,3bHIx84H,wdrmL…|
18|28-396-526-5053|8914.71| HOUSEHOLD| nag. furiously c…|
|     20|Customer#000000020|       JrPk8Pqplj4Ne|       22|32-957-234-8742|
7603.4| FURNITURE|g alongside of th…|
+-------+------------------+--------------------+---------+---------------+-----
--+----------+--------------------+
only showing top 20 rows
```

[4]:
```
order.select("ORDERKEY", "CUSTKEY").show()
```

```
+--------+-------+
|ORDERKEY|CUSTKEY|
+--------+-------+
|       1|   3691|
|       2|   7801|
|       3|  12332|
|       4|  13678|
|       5|   4450|
|       6|   5563|
|       7|   3914|
|      32|  13006|
|      33|   6697|
|      34|   6101|
|      35|  12760|
|      36|  11527|
|      37|   8612|
|      38|  12484|
|      39|   8177|
|      64|   3212|
|      65|   1627|
|      66|  12920|
|      67|   5662|
|      68|   2855|
+--------+-------+
only showing top 20 rows
```

[5]:
```
lineitems.select("ORDERKEY", "SUPPKEY", "PARTKEY", "QUANTITY").show()
```

```
+--------+-------+-------+--------+
|ORDERKEY|SUPPKEY|PARTKEY|QUANTITY|
+--------+-------+-------+--------+
|       1|    785|  15519|      17|
|       1|    732|   6731|      36|
|       1|    371|   6370|       8|
```

```
|       1|    465|    214|      28|
|       1|    160|   2403|      24|
|       1|     67|   1564|      32|
|       2|    138|  10617|      38|
|       3|    181|    430|      45|
|       3|    658|   1904|      49|
|       3|    370|  12845|      27|
|       3|    191|   2938|       2|
|       3|    115|  18310|      28|
|       3|    984|   6215|      26|
|       4|    579|   8804|      30|
|       5|    858|  10857|      15|
|       5|    394|  12393|      26|
|       5|      8|   3754|      50|
|       6|    228|  13964|      37|
|       7|     11|  18206|      12|
|       7|    790|  14525|       9|
+--------+-------+-------+--------+
only showing top 20 rows
```

[6]:
```
# Question 1
# Implement a pyspark code that can find out the top-10 sold products.

lineitems\
    .select("ORDERKEY", "PARTKEY")\
    .withColumn("COUNT", lit(1))\
    .groupBy("PARTKEY")\
    .agg(f.sum("COUNT").alias("TOTAL"))\
    .orderBy("TOTAL", ascending=False)\
    .limit(10)\
    .show()
```

```
+-------+-----+
|PARTKEY|TOTAL|
+-------+-----+
|  10620|   56|
|   6140|   54|
|  15584|   52|
|   8051|   52|
|  10715|   51|
|  10597|   51|
|   2292|   51|
|  14422|   50|
|  17670|   50|
|  19444|   50|
+-------+-----+
```

[7]:
```
# ------------------------------------------------------------------------
# Question 2

# Find the top-10 customers based on the number of products ordered.

# Collect all df to be used
l = lineitems.select("ORDERKEY", "PARTKEY", "QUANTITY")
o = order.select("ORDERKEY", "CUSTKEY")

# Join, group, add quantity, and sort
o.join(l, ["ORDERKEY"], 'full')\
    .drop("ORDERKEY")\
    .groupBy("CUSTKEY")\
    .agg(f.sum("QUANTITY").alias("TOTAL"))\
    .orderBy("TOTAL", ascending=False)\
    .limit(10)\
    .show()
```

```
+-------+-----+
|CUSTKEY|TOTAL|
+-------+-----+
|   8362| 4082|
|   9454| 3870|
|    346| 3817|
|   6958| 3760|
|   1105| 3737|
|  14707| 3710|
|  11998| 3709|
|  14398| 3670|
|   8542| 3660|
|   8761| 3658|
+-------+-----+
```

[8]:
```
# ------------------------------------------------------------------------
# Question 3
# Find the top-10 customers that have ordered products from the same supplier.

# Collect all df to be used
l = lineitems.select("ORDERKEY", "PARTKEY")
o = order.select("ORDERKEY", "CUSTKEY")
p = partsupp.select("PARTKEY", "SUPPKEY")

# Join orders and lineitems
ol = o.join(l, ["ORDERKEY"], 'full')\
```

```
    .drop("ORDERKEY")

# Join orders, lineitems, and part supplier.
# Group by both "CUSTKEY" and "SUPPKEY", then sort
ol.join(p, ["PARTKEY"], 'full')\
    .drop("PARTKEY")\
    .withColumn("COUNT", f.lit(1))\
    .groupBy("CUSTKEY", "SUPPKEY")\
    .agg(f.sum("COUNT").alias("COUNT"))\
    .orderBy("COUNT", ascending=False)\
    .limit(10)\
    .show()
```

```
+-------+-------+-----+
|CUSTKEY|SUPPKEY|COUNT|
+-------+-------+-----+
|   4567|    844|    7|
|   4792|    592|    6|
|  11809|     17|    6|
|  14767|      8|    6|
|   2173|    572|    6|
|   6139|    233|    6|
|    874|    430|    6|
|    154|    380|    5|
|   6889|    729|    5|
|   8794|    545|    5|
+-------+-------+-----+
```

[9]:
```
# ------------------------------------------------------------------------
# Question 4 and 5
# Find the customers who have not ordered products from their own country and␣
 ↪have ordered only foreign products.

# Collect all df to be used
o = order.select("CUSTKEY", "ORDERKEY")
l = lineitems.select("ORDERKEY", "SUPPKEY")
c = customer.select("CUSTKEY", f.col("NATIONKEY").alias("CUST_NATION")) \
    .withColumn("CUSTKEY", f.col("CUSTKEY").cast(IntegerType()))
s = supplier.select("SUPPKEY", f.col("NATIONKEY").alias("SUPP_NATION")) \
    .withColumn("SUPPKEY", f.col("SUPPKEY").cast(IntegerType()))

# Join dfs together
# ("CUSTKEY", "SUPPKEY")
ol = o.join(l, ["ORDERKEY"], 'full') \
    .drop("ORDERKEY")
# ("CUSTKEY", "SUPPKEY", "CUST_NATION" )
```

```
olc = ol.join(c, ["CUSTKEY"], 'full')
# ("CUSTKEY", "CUST_NATION", "SUPP_NATION" )
olcs = olc.join(s, ["SUPPKEY"], 'full') \
    .drop("SUPPKEY")

# Define UDF (to check condition)
only_own = udf(lambda x, y: all(i is x for i in y), BooleanType())
only_foreign = udf(lambda x, y: x not in y, BooleanType())

# Aggregate, collect SUPP_NATION, and check condition
result = olcs\
    .filter(f.col("SUPP_NATION").isNotNull())\
    .groupBy("CUSTKEY", "CUST_NATION") \
    .agg(f.collect_set("SUPP_NATION").alias("SUPP_NATIONS")) \
    .withColumn("ONLY_SAME", only_own(f.col("CUST_NATION"), f.
↪col("SUPP_NATIONS"))) \
    .withColumn("ONLY_FOREIGN", only_foreign(f.col("CUST_NATION"), f.
↪col("SUPP_NATIONS")))


# Q4 Answer
result.filter(result["ONLY_SAME"] == True).show()
```

```
+-------+-----------+------------+---------+------------+
|CUSTKEY|CUST_NATION|SUPP_NATIONS|ONLY_SAME|ONLY_FOREIGN|
+-------+-----------+------------+---------+------------+
+-------+-----------+------------+---------+------------+
```

[10]:
```
# Q5 Answer
result.filter(result["ONLY_FOREIGN"] == True).show()
```

```
+-------+-----------+--------------------+---------+------------+
|CUSTKEY|CUST_NATION|        SUPP_NATIONS|ONLY_SAME|ONLY_FOREIGN|
+-------+-----------+--------------------+---------+------------+
|   7262|         13|[0, 15, 9, 1, 16,…|    false|        true|
|  13802|          5|[0, 9, 1, 2, 17, …|    false|        true|
|  10106|         18|[15, 9, 1, 16, 2,…|    false|        true|
|   1628|         18|[15, 9, 1, 16, 2,…|    false|        true|
|   6302|         11|[0, 9, 16, 5, 17,…|    false|        true|
|   9023|         10|[0, 15, 9, 1, 2, …|    false|        true|
|   2279|         23|[15, 9, 1, 16, 17…|    false|        true|
|  12767|          1|[15, 9, 16, 2, 17…|    false|        true|
|   6145|         21|[0, 15, 9, 1, 16,…|    false|        true|
|  10217|         22|[9, 1, 2, 17, 3, …|    false|        true|
|   5459|         19|[15, 1, 16, 2, 17…|    false|        true|
|   1124|          1|[15, 9, 2, 17, 24…|    false|        true|
```

```
|   2500|          0|[15, 9, 1, 16, 2,…|    false|        true|
|   4442|          7|[0, 9, 1, 16, 17,…|    false|        true|
|   7967|          3|[0, 15, 9, 1, 16,…|    false|        true|
|   8170|         13|[0, 15, 9, 1, 16,…|    false|        true|
|  10310|         12|[1, 16, 13, 19, 2…|    false|        true|
|   6620|         13|[15, 9, 16, 2, 17…|    false|        true|
|  14201|          1|[0, 15, 16, 2, 17…|    false|        true|
|  13528|         13|[0, 15, 9, 1, 2, …|    false|        true|
+-------+-----------+------------------+---------+------------+
only showing top 20 rows
```

[11]:
```
# Q6 Answer


def jaccard_similarity(list1, list2):
    s1 = set(list1)
    s2 = set(list2)
    return len(s1.intersection(s2)) / len(s1.union(s2))

jaccard_similarity_udf = udf(lambda x,y: jaccard_similarity(x,y),
                             FloatType())

o = order.select("CUSTKEY", "ORDERKEY")
l = lineitems.select("ORDERKEY", "PARTKEY")

ol = o.join(l, ["ORDERKEY"], 'full')\
    .drop("ORDERKEY")\
    .groupBy("CUSTKEY")\
    .agg(f.collect_set("PARTKEY").alias("PART_LIST"))

ol.crossJoin(ol)\
    .toDF("CUSTKEY1","PART_LIST1","CUSTKEY2","PART_LIST2")\
    .filter( f.col("CUSTKEY1") != f.col("CUSTKEY2"))\
    .dropDuplicates(["CUSTKEY1", "CUSTKEY2"])\
    .withColumn("JACCARD", jaccard_similarity_udf(f.col("PART_LIST1"), f.
 ↪col("PART_LIST2")))\
    .orderBy(f.desc("JACCARD"))\
    .limit(10)\
    .show()
```

```
+--------+--------------------+--------+--------------------+----------+
|CUSTKEY1|          PART_LIST1|CUSTKEY2|          PART_LIST2|   JACCARD|
+--------+--------------------+--------+--------------------+----------+
|    8456|[15747, 18343, 41…|   10376|[13032, 18343, 15…|0.09090909|
|   10376|[13032, 18343, 15…|    8456|[15747, 18343, 41…|0.09090909|
|   10901|[10142, 9529, 124…|    4808|[17169, 19122, 33…|0.06666667|
|    4808|[17169, 19122, 33…|   10901|[10142, 9529, 124…|0.06666667|
```

```
|    7532|[15572, 2151, 174…|    5390|[5452, 16969, 755…|0.06451613|
|    5390|[5452, 16969, 755…|    7532|[15572, 2151, 174…|0.06451613|
|    2489|[6418, 7101, 7102…|    4283|[13060, 12044, 12…|0.06349207|
|    4283|[13060, 12044, 12…|    2489|[6418, 7101, 7102…|0.06349207|
|    2768|[19866, 1648, 123…|    4385|[1648, 7100, 1122…|    0.0625|
|    4385|[1648, 7100, 1122…|    2768|[19866, 1648, 123…|    0.0625|
+--------+------------------+--------+------------------+----------+
```

[13]:
```
# Q7 Answer
l = lineitems.select("ORDERKEY", "PARTKEY")

l_partslist = l\
    .groupBy("ORDERKEY")\
    .agg(f.collect_set(f.col("PARTKEY")).alias("PARTLIST"))

def combin(x):
    return [ '+'.join([str(i),str(j)]) for i,j in combinations(sorted(x),2) ]

cmb = udf(lambda x: combin(x), ArrayType(StringType()))

l_partslist\
    .where(f.size(f.col("PARTLIST")) > 1)\
    .withColumn("PARTCOMBLIST", cmb(f.col("PARTLIST")))\
    .drop("PARTLIST")\
    .withColumn("PARTCOMB", f.explode("PARTCOMBLIST"))\
    .withColumn("COUNT", f.lit(1))\
    .groupBy("PARTCOMB")\
    .agg(f.sum(f.col("COUNT")).alias("TOTAL"))\
    .orderBy(f.col("TOTAL").desc())\
    .limit(10)\
    .show()
```

```
+-----------+-----+
|   PARTCOMB|TOTAL|
+-----------+-----+
|  595+11837|    3|
|11004+15109|    3|
| 6031+15277|    3|
|14524+14743|    3|
|   250+7045|    3|
|11630+14244|    3|
|14405+17144|    3|
|   364+3823|    3|
| 5850+11561|    3|
|12966+16068|    3|
+-----------+-----+
```

[ ]: