

CS 5433 - Big Data Management Spring 2022

Spark Programming Assignment -1

4/15/2022

GROUP – 9

Task 1: Calculating the average income by marital status using the SQL 'GROUP' clause.

Spark SQL is a structured data processing Spark module. It offers data frames as a programming abstraction and may also serve as a distributed SQL query engine.

Approach:

- The code for this task is done using Sparksql in Scala. Initially we have imported the *SparkSession* using, *'import org.apache.spark.sql.SparkSession'*.
- As *sparkSession* is the entry point for reading data. We created a spark session to send the user commands and data to a spark application. We have used *.getOrCreate()* which would create a new session or get an existing one if it exists already and named application name as 'spark SQL' using *appName()* method. *SparkSession.builder()* method is created for constructing a *SparkSession*. The *.config()* is used to set config option which are by default propagated to both 'SparkConf' and 'Spark Session'.
'SparkSession.builder().appName("Spark SQL").config("spark.some.config.option", "some-value").getOrCreate()'.
- Next for reading the data from the data sources, we have used data frame reader *'spark.read.format()'* method. Make sure to check if the header was true using *option()* method. For the given file format, we have used the delimiter as "\t" which was specified in the *option()* method and then used *load()* method to load the path of the data *'/user/kaggle/kaggle_data/marketing_campaign.csv'* to load the data frame in the same format with the tabular format.
df=spark.read.format("csv").option("header","true").option("delimiter", "\t").load("/user/kaggle/kaggle_data/marketing_campaign.csv");
- Using the method *createOrReplaceTempView()* to create a local temporary view and to convert the data frame into a temporary view 'people'.
df.createOrReplaceTempView("people")
- Next, we are creating a new dataframe 'sqlDF' and running our required final sql query using *spark.sql()* method, the query would fetch the 'average_income' grouping the 'marital_status'.
val sqlDF = spark.sql("Select Marital_Status,avg(Income) as Income from people group by Marital_Status")
- Next, we have used a command to print the output of the program to the console.
sqlDF.show()
- The final step of the program is to write the output of the program to a file in csv format, the output file should be changed every time as it will not be overwritten. We can run commands in Hadoop to check the output.
sqlDF.coalesce(1).write.option("header","true").csv("/user/ssamine/spark_task_1.csv")

Instructions to run code and screenshots for Task 1:

1,1)

Run the command to start spark session. It will open the shell

Command: spark-shell

```
ssamine@hadoop-nn001:~$ spark-shell
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/usr/local/spark-3.0.1-bin-hadoop3.2/jars/spark-unsafe_2.12-3.0.1.jar) to constructor j
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2022-04-13 12:56:42,217 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Setting default log level to "WARN".
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).
2022-04-13 12:56:48,871 WARN util.Utils: Service 'SparkUI' could not bind on port 4040. Attempting port 4041.
2022-04-13 12:56:48,872 WARN util.Utils: Service 'SparkUI' could not bind on port 4041. Attempting port 4042.
2022-04-13 12:56:48,872 WARN util.Utils: Service 'SparkUI' could not bind on port 4042. Attempting port 4043.
2022-04-13 12:56:48,872 WARN util.Utils: Service 'SparkUI' could not bind on port 4043. Attempting port 4044.
2022-04-13 12:56:48,873 WARN util.Utils: Service 'SparkUI' could not bind on port 4044. Attempting port 4045.
2022-04-13 12:56:48,873 WARN util.Utils: Service 'SparkUI' could not bind on port 4045. Attempting port 4046.
2022-04-13 12:56:48,873 WARN util.Utils: Service 'SparkUI' could not bind on port 4046. Attempting port 4047.
2022-04-13 12:56:48,874 WARN util.Utils: Service 'SparkUI' could not bind on port 4047. Attempting port 4048.
2022-04-13 12:56:48,874 WARN util.Utils: Service 'SparkUI' could not bind on port 4048. Attempting port 4049.
2022-04-13 12:56:48,874 WARN util.Utils: Service 'SparkUI' could not bind on port 4049. Attempting port 4050.
2022-04-13 12:56:48,874 WARN util.Utils: Service 'SparkUI' could not bind on port 4050. Attempting port 4051.
2022-04-13 12:56:48,875 WARN util.Utils: Service 'SparkUI' could not bind on port 4051. Attempting port 4052.
2022-04-13 12:56:48,875 WARN util.Utils: Service 'SparkUI' could not bind on port 4052. Attempting port 4053.
2022-04-13 12:56:48,876 WARN util.Utils: Service 'SparkUI' could not bind on port 4053. Attempting port 4054.
2022-04-13 12:56:50,186 WARN yarn.Client: Neither spark.yarn.jars nor spark.yarn.archive is set, falling back to uploading libraries under SPARK_HOME.
Spark context Web UI available at http://hadoop-nn001:4054
Spark context available as 'sc' (master = yarn, app id = application_1647031195237_0410).
Spark session available as 'spark'.
Welcome to
  ____  __
 / ___/ /_
/ /   / __ \
/ /___/ / ___/
\____/_/

version 3.0.1
```

Figure 1,1: Shows the command for starting Spark session

1,2)

Compile and execute the .scala file using the command. Output will be printed on the console and written into file as well.

Command: load filepath

Ex: load /home/ssamine/Group_9_Program_1.scala

```
scala> load /home/ssamine/Group_9_Program_1.scala
Loading /home/ssamine/Group_9_Program_1.scala...
import org.apache.spark.sql.SparkSession
2022-04-13 12:58:37,722 WARN sql.SparkSession$Builder: Using an existing SparkSession; some spark core configurations may not take effect.
spark: org.apache.spark.sql.SparkSession = org.apache.spark.sql.SparkSession@1a611ec7
df: org.apache.spark.sql.DataFrame = [ID: string, Year Birth: string ... 27 more fields]
sqlDF: org.apache.spark.sql.DataFrame = [Marital_Status: string, Income: double]
+-----+-----+
|Marital_Status|      Income|
+-----+-----+
|      YOLO|    48432.0|
| Together|53245.53403141361|
| Married|51724.97899649942|
|   Absurd|    72365.5|
|   Widow|56481.55263157895|
| Divorced|52834.22844827586|
|     Alone|    43789.0|
|   Single|50985.35031847134|
+-----+-----+
```

Figure 1,2: Shows the command to compile, run the program and print the output on the console.

1,3)

Checking the output file

Command: `hdfs dfs -ls outputpath/outputfilename/part-r-00000`

Ex: `hdfs dfs -ls /user/ssamine/sparksql_task_1.csv/part-00000-408205aa-ff4c-4fdc-922c-52e0e9d38285-c000.csv`

```
ssamine@hadoop-mn001:~$ hdfs dfs -ls /user/ssamine/sparksql_task_1.csv/part-00000-a2474546-ebda-40dd-b7ec-14adaf5826c1-c000.csv
2022-04-13 13:00:31,086 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
-rw-r--r--  3 ssamine ssamine      193 2022-04-13 12:58 /user/ssamine/sparksql_task_1.csv/part-00000-a2474546-ebda-40dd-b7ec-14adaf5826c1-c000.csv
```

Figure 1,3: Shows the command to view the output file

1,4)

For viewing output

`hdfs dfs -cat outputpath/outputfilename/part-r-00000`

Ex: `hdfs dfs -cat /user/ssamine/sparksql_task_1.csv/part-00000-408205aa-ff4c-4fdc-922c-52e0e9d3828-c000.csv`

```
ssamine@hadoop-mn001:~$ hdfs dfs -cat /user/ssamine/sparksql_task_1.csv/part-00000-a2474546-ebda-40dd-b7ec-14adaf5826c1-c000.csv
2022-04-13 13:00:44,747 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Marital_Status,Income
YOLO,48432.0
Together,53245.53403141361
Married,51724.97899649942
Absurd,72365.5
Widow,56481.55263157895
Divorced,52834.22844827586
Alone,43789.0
Single,50995.35031847134
```

Figure 1,4: Shows the command to view the output written to output file

Sample Output:

Marital_Status,avg_Income
YOLO,48432.0
Together,53245.53403141361
Married,51724.97899649942
Absurd,72365.5
Widow,56481.55263157895
Divorced,52834.22844827586
Alone,43789.0
Single,50995.35031847134

Task 2: Calculating the average income by marital status using Python.

Pyspark is a Python-based connection for Apache Spark. It's a Spark Python API that connects Apache Spark and Python to Resilient Distributed Datasets (RDDs).

Approach:

- For this task 2 we have used pyspark, python. Initially we need to import all the required libraries. We have imported *SparkSession*, *SparkContext* and *SparkConf*. *SparkContext* represents a connection to a spark cluster, Spark Config provides configurations to run a spark application and *SparkSession* is the entry point for reading data.

```
import sys
```

```
from pyspark import SparkContext, SparkConf
```

```
from pyspark.sql.session import SparkSession
```

- Next, we make sure everything in the program after main is executed when the python interpreter runs the module directly.

```
if __name__ == "__main__":
```

- Then we set the configuration for the spark application using *SparkConf()* method and set the app name as "BDM" using *setAppName()* method.

```
conf = SparkConf().setAppName("BDM")
```

- Next we create the connection to a spark cluster using *SparkContext()* method .

```
sc = SparkContext(conf=conf)
```

- Next, we created a spark session to send the user commands and data to a spark application. We have used *.getOrCreate()* which would create a new session or get an existing one if it exists already and named application name as 'BDMPyspark' using *appName()* method. *SparkSession.builder* method is created for constructing a *SparkSession*.

```
spark = SparkSession.builder.appName("BDMPyspark").getOrCreate()
```

- Next for reading the data from the data sources, we have used data frame reader '*spark.read.csv()*' method. Make sure to check if the header was true. For the given file format, we have used the delimiter as "\t" and then used the path of the data '*/user/kaggle/kaggle_data/marketing_campaign.csv*' to load the data frame in the same format which is the tabular format.

```
df = spark.read.csv('/user/kaggle/kaggle_data/marketing_campaign.csv', sep='\t', inferSchema=True, header=True)
```

- Next created a new data frame 'res', which will give us the average income by the marital status. By using *group()* and *avg()* methods. Here, groupby will group the people who have same marital status with aggregate function average.

```
res = df.groupBy("Marital_Status").avg("Income")
```

- The output is printed to the console.

```
res.show()
```

- The final step of the program is to write the output of the program to a file in csv format.

```
res.coalesce(1).write.option("header", "true").csv("/user/vgollap/Group_9_Program_2_output.csv")
```

Instructions to run code and screenshots for Task 2:

2,1)

Run the python file using spark-submit command:

Command: spark-submit pathToPythonFile/pythonFile.py

Ex: spark-submit Group_9_Program_2.py

```
vgollap@hadoop-mn001:~$ spark-submit Group_9_Program_2.py
WARNING: An illegal reflective access operation has occurred
WARNING: Illegal reflective access by org.apache.spark.unsafe.Platform (file:/usr/local/spark-3.0.1-bin-hadoop3.2/jars/spark-unsafe_2.12-3.0.1.jar) to constructor java.nio.DirectByteBuffer(long,int)
WARNING: Please consider reporting this to the maintainers of org.apache.spark.unsafe.Platform
WARNING: Use --illegal-access=warn to enable warnings of further illegal reflective access operations
WARNING: All illegal access operations will be denied in a future release
2022-04-14 08:45:43,649 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
2022-04-14 08:45:43,656 INFO spark.SparkContext: Running Spark version 3.0.1
2022-04-14 08:45:43,544 INFO resource.ResourceUtils: =====
2022-04-14 08:45:43,545 INFO resource.ResourceUtils: Resources for spark.driver:
2022-04-14 08:45:43,546 INFO resource.ResourceUtils: =====
2022-04-14 08:45:43,546 INFO spark.SparkContext: Submitted application: RM
2022-04-14 08:45:43,609 INFO spark.SecurityManager: Changing view acls to: vgollap
2022-04-14 08:45:43,609 INFO spark.SecurityManager: Changing modify acls to: vgollap
2022-04-14 08:45:43,609 INFO spark.SecurityManager: Changing view acls groups to:
2022-04-14 08:45:43,609 INFO spark.SecurityManager: Changing modify acls groups to:
2022-04-14 08:45:43,609 INFO spark.SecurityManager: SecurityManager: authentication disabled; ui acls disabled; users  with view permissions: Set(vgollap); groups with view permissions: Set(); users  with modify permissions: Set(vgollap); groups with modify permissions: Set()
2022-04-14 08:45:43,662 INFO util.Utils: Successfully started service 'sparkDriver' on port 38043.
2022-04-14 08:45:43,902 INFO spark.SparkEnv: Registering MapOutputTracker
2022-04-14 08:45:43,935 INFO spark.SparkEnv: Registering BlockManagerMaster
2022-04-14 08:45:43,956 INFO storage.BlockManagerMasterEndpoint: Using org.apache.spark.storage.DefaultTopologyMapper for getting topology information
2022-04-14 08:45:43,957 INFO storage.BlockManagerMasterEndpoint: BlockManagerMasterEndpoint up
2022-04-14 08:45:43,995 INFO spark.SparkEnv: Registering BlockManagerMasterHeartbeat
2022-04-14 08:45:44,000 INFO storage.DiskBlockManager: Created local directory at /tmp/blockmgr-550cd714-802c-4b5a-930f-ade8590b2554
2022-04-14 08:45:44,016 INFO memory.MemoryStore: MemoryStore started with capacity 414.4 MiB
2022-04-14 08:45:44,079 INFO spark.SparkEnv: Registering OutputCommitCoordinator
2022-04-14 08:45:44,129 INFO util.log: Logging initialized @116ms to org.apache.hadoop.yarn.util.log.HdfsLog
2022-04-14 08:45:44,233 INFO server.Server: jetty-9.4.z-SNAPSHOT; built: 2019-04-29T20:42:08.989Z; git: elbcb3120a6e17ee3df052294e433fa25ce7097; jvm 11.0.14.1-LinuxUbuntu-20.04
2022-04-14 08:45:44,256 INFO server.Server: Started @8231ms
2022-04-14 08:45:44,256 INFO server.AbstractConnector: Started ServerConnector@18700b(HTTP/1.1,[http://1.1]:4040)
2022-04-14 08:45:44,286 INFO util.Utils: Successfully started service 'SparkUI' on port 4040.
2022-04-14 08:45:44,308 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@9bcb8186{/job,null,AVAILABLE,@spark}
2022-04-14 08:45:44,310 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@784b3b59{/job/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,311 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@2f7fba7f{/job/job,null,AVAILABLE,@spark}
2022-04-14 08:45:44,315 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@112e884f{/job/job/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,316 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@7e2415b4{/stages,null,AVAILABLE,@spark}
2022-04-14 08:45:44,316 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@6665d688{/stages/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,317 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@81f1a6bd{/stages/stage,null,AVAILABLE,@spark}
2022-04-14 08:45:44,319 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@11d8e788{/stages/stage/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,320 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@3a34f039{/stages/pool,null,AVAILABLE,@spark}
2022-04-14 08:45:44,321 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@7dbcb0d0{/stages/pool/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,322 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@2467943e{/storage,null,AVAILABLE,@spark}
2022-04-14 08:45:44,323 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@80c8a360{/storage/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,324 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@200a4d6d{/storage/rdd,null,AVAILABLE,@spark}
2022-04-14 08:45:44,325 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@3346cfac3{/storage/rdd/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,326 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@4c1018f7{/environment,null,AVAILABLE,@spark}
2022-04-14 08:45:44,326 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@8ba5eaca{/environment/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,327 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@1bba6e29{/executors,null,AVAILABLE,@spark}
2022-04-14 08:45:44,328 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@2436b3b1{/executors/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,329 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@830c0231{/executors/threadDump,null,AVAILABLE,@spark}
2022-04-14 08:45:44,331 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@8dcf0318{/executors/threadDump/psn,null,AVAILABLE,@spark}
2022-04-14 08:45:44,340 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@11ef8737{/static,null,AVAILABLE,@spark}
2022-04-14 08:45:44,341 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@81e1c26e{/null,null,AVAILABLE,@spark}
2022-04-14 08:45:44,342 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@2aee6eb7{/api,null,AVAILABLE,@spark}
2022-04-14 08:45:44,343 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@2e1cd228{/job/job/kill,null,AVAILABLE,@spark}
2022-04-14 08:45:44,344 INFO handler.ContextHandler: Started o.e.j.s.ServletContextHandler@7544a2e1{/stages/stage/kill,null,AVAILABLE,@spark}
2022-04-14 08:45:44,346 INFO util.SpawnUI: Bound SparkUI to 0.0.0.0 and started at http://hadoop-mn001:4040
2022-04-14 08:45:44,747 INFO client.RMProxy: Connecting to ResourceManager at hadoop-mn001.cs.okstate.edu/192.168.122.2:18032
2022-04-14 08:45:45,038 INFO yarn.Client: Requesting a new application from cluster with 12 Monitors
2022-04-14 08:45:45,441 INFO conf.Configuration: resource-type.xml not found
2022-04-14 08:45:45,441 INFO resource.ResourceUtils: Unable to find 'resource-types.xml'.
```

Figure 2,1 Running command in pyspark

2,2) Output will be printed on the console and written into file as well

```
-----+
|Marital_Status|    avg(Income)|
-----+-----+
|      YOLO    |    48432.0    |
|Together|53245.53403141361|
|  Married|51724.97899649942|
|    Absurd|    72365.5    |
|Widow|56481.55263157895|
|Divorced|52834.22844827586|
|    Alone|    43789.0    |
|    Single|50995.35031847134|
-----+-----+
```

Figure 2,2 Printed Output on console

2,3)

Checking the output file

Command: hdfs dfs -cat outputpath/outputfilename/part-r-00000

Ex: hdfs dfs -cat /user/vgollap/Group_9_Program_2_output.csv/part-00000-370cc613-7053-4fef-92cc-0cbf782ec4c9-c000.csv

```

vgollap@hadoop-nn001:~$ hdfs dfs -ls
2022-04-14 08:48:51,020 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 14 items
drwxr-xr-x - vgollap vgollap 0 2022-04-14 08:46 .sparkStaging
drwxr-xr-x - vgollap vgollap 0 2022-04-13 10:37 BDMOP.csv
-rw-r--r-- 3 vgollap vgollap 309 2022-04-10 15:58 BDMgrp.py
drwxr-xr-x - vgollap vgollap 0 2022-04-14 08:46 Group_9_Program_2_output.csv
-rw-r--r-- 3 vgollap vgollap 746 2022-04-14 08:42 Group_9_Program_2.py
drwxr-xr-x - vgollap vgollap 0 2022-03-07 00:19 PartitionerOutput
drwxr-xr-x - vgollap vgollap 0 2022-03-07 00:09 RowCountOutput
drwxr-xr-x - vgollap vgollap 0 2022-03-07 17:23 a
drwxr-xr-x - vgollap vgollap 0 2022-03-07 17:23 b
drwxr-xr-x - vgollap vgollap 0 2022-03-08 10:52 bdm1
drwxr-xr-x - vgollap vgollap 0 2022-03-08 11:25 bdm2
drwxr-xr-x - vgollap vgollap 0 2022-03-07 17:23 c
drwxr-xr-x - vgollap vgollap 0 2022-03-06 22:36 flumedata
drwxr-xr-x - vgollap vgollap 0 2022-03-07 15:05 sample
vgollap@hadoop-nn001:~$ hdfs dfs -ls Group_9_Program_2_output.csv
2022-04-14 08:49:11,175 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Found 2 items
-rw-r--r-- 3 vgollap vgollap 0 2022-04-14 08:46 Group_9_Program_2_output.csv/_SUCCESS
-rw-r--r-- 3 vgollap vgollap 189 2022-04-14 08:46 Group_9_Program_2_output.csv/part-00000-c9103e7c-0d66-46f0-94e4-7bb93969c0b3-c000.csv
vgollap@hadoop-nn001:~$ hdfs dfs -cat Group_9_Program_2_output.csv/part-00000-c9103e7c-0d66-46f0-94e4-7bb93969c0b3-c000.csv
2022-04-14 08:49:29,159 WARN util.NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable
Marital Status,avg(Income)
YOLO,48432.0
Together,53245.53403141361
Married,51724.97899649940
Absurd,72365.5
Widow,56481.55263157895
Divorced,52834.22844827586
Alone,43789.0
Single,50995.35031847134
vgollap@hadoop-nn001:~$

```

Figure 2,3 To look up the written output file

Sample Output:

Sample-output_Program_2

Marital_Status	avg(Income)
YOLO	48432.0
Together	53245.53403141360
Married	51724.97899649940
Absurd	72365.5
Widow	56481.55263157900
Divorced	52834.22844827590
Alone	43789.0
Single	50995.35031847130