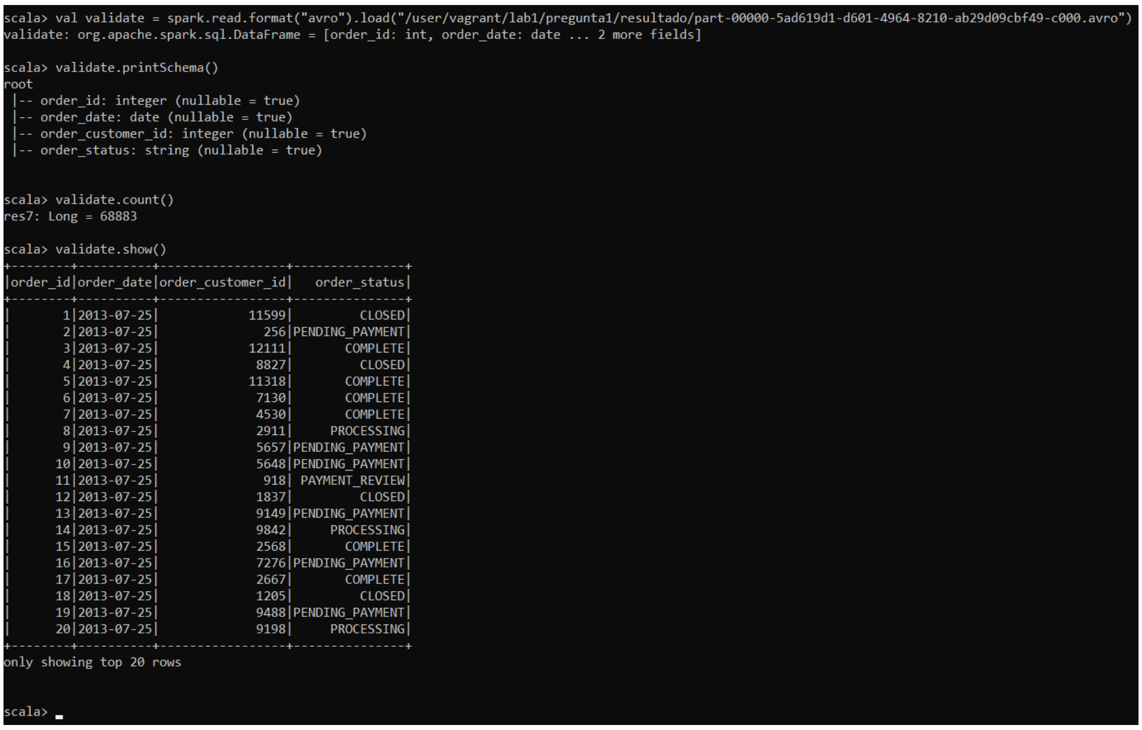
PREGUNTA 1

import org.apache.spark.sql.types._

val customSchema = StructType(Array(StructField("order_id", IntegerType, true), StructField("order_date", DateType, true), StructField("order_customer_id", IntegerType, true), StructField("order_status", StringType, true)))

val orders = spark.read.format("csv").option("inferSchema", "true").schema(customSchema).load("/public/retail_db/orders")

orders.write.format("avro").save("/user/vagrant/lab1/pregunta1/resultado")



```
scala> val validate = spark.read.format("avro").load("/user/vagrant/lab1/pregunta1/resultado/part-00000-5ad619d1-d601-4964-8210-ab29d09cbf49-c000.avro")
validate: org.apache.spark.sql.DataFrame = [order_id: int, order_date: date ... 2 more fields]

scala> validate.printSchema()
root
 |-- order_id: integer (nullable = true)
 |-- order_date: date (nullable = true)
 |-- order_customer_id: integer (nullable = true)
 |-- order_status: string (nullable = true)


scala> validate.count()
res7: Long = 68883

scala> validate.show()
+--------+----------+-----------------+---------------+
|order_id|order_date|order_customer_id|   order_status|
+--------+----------+-----------------+---------------+
|       1|2013-07-25|            11599|         CLOSED|
|       2|2013-07-25|              256|PENDING_PAYMENT|
|       3|2013-07-25|            12111|       COMPLETE|
|       4|2013-07-25|             8827|         CLOSED|
|       5|2013-07-25|            11318|       COMPLETE|
|       6|2013-07-25|             7130|       COMPLETE|
|       7|2013-07-25|             4530|       COMPLETE|
|       8|2013-07-25|             2911|     PROCESSING|
|       9|2013-07-25|             5657|PENDING_PAYMENT|
|      10|2013-07-25|             5648|PENDING_PAYMENT|
|      11|2013-07-25|              918| PAYMENT_REVIEW|
|      12|2013-07-25|             1837|         CLOSED|
|      13|2013-07-25|             9149|PENDING_PAYMENT|
|      14|2013-07-25|             9842|     PROCESSING|
|      15|2013-07-25|             2568|       COMPLETE|
|      16|2013-07-25|             7276|PENDING_PAYMENT|
|      17|2013-07-25|             2667|       COMPLETE|
|      18|2013-07-25|             1205|         CLOSED|
|      19|2013-07-25|             9488|PENDING_PAYMENT|
|      20|2013-07-25|             9198|     PROCESSING|
+--------+----------+-----------------+---------------+
only showing top 20 rows


scala>
```

PREGUNTA 2

val customerSchema = StructType(Array(

StructField("customer_id", IntegerType, true),

StructField("customer_fname", StringType, true),

StructField("customer_lname", StringType, true),

StructField("customer_email", StringType, true),

StructField("customer_password", StringType, true),

StructField("customer_street", StringType, true),

StructField("customer_state", StringType, true),

StructField("customer_city", StringType, true),

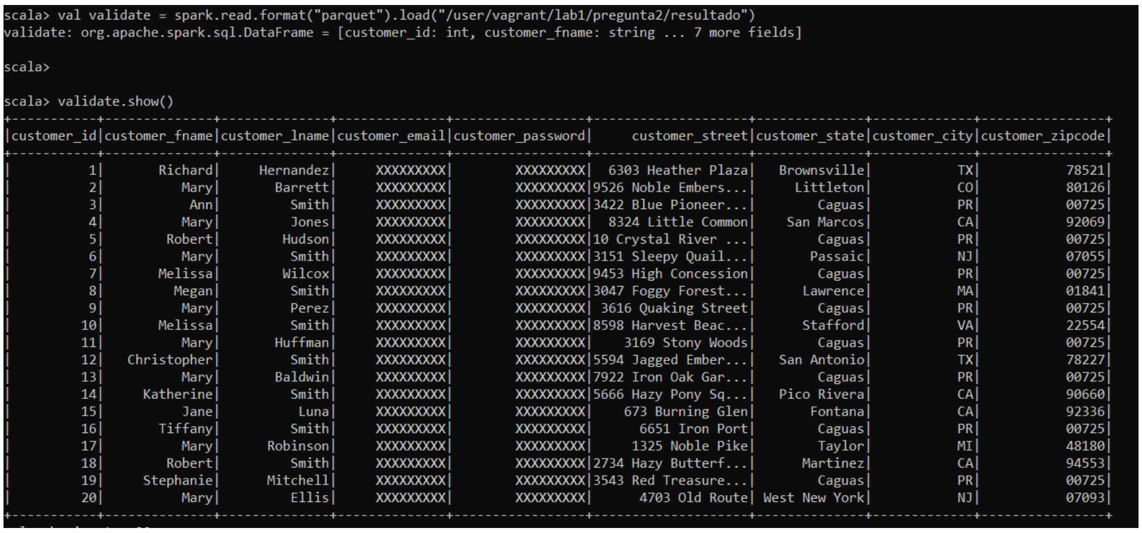
StructField("customer_zipcode", StringType, true)

)

)

val customer = spark.read.format("csv").option("inferSchema","true").schema(customerSchema).load("/public/retail_db/customers/part-00000")

customer.write.format("parquet").save("/user/vagrant/lab1/pregunta2/resultado")

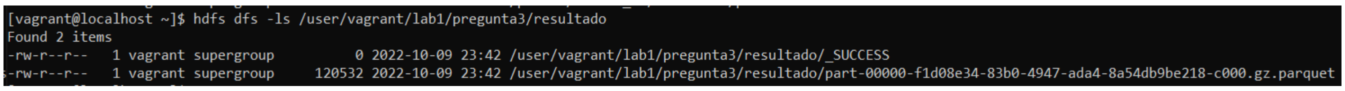


```
scala> val validate = spark.read.format("parquet").load("/user/vagrant/lab1/pregunta2/resultado")
validate: org.apache.spark.sql.DataFrame = [customer_id: int, customer_fname: string ... 7 more fields]

scala>

scala> validate.show()
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+-------------+---------------+
|customer_id|customer_fname|customer_lname|customer_email|customer_password|     customer_street|customer_state|customer_city|customer_zipcode|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+-------------+---------------+
|          1|       Richard|     Hernandez|     XXXXXXXXX|        XXXXXXXXX|  6303 Heather Plaza|   Brownsville|           TX|          78521|
|          2|          Mary|       Barrett|     XXXXXXXXX|        XXXXXXXXX|9526 Noble Embers...|     Littleton|           CO|          80126|
|          3|           Ann|         Smith|     XXXXXXXXX|        XXXXXXXXX|3422 Blue Pioneer...|        Caguas|           PR|          00725|
|          4|          Mary|         Jones|     XXXXXXXXX|        XXXXXXXXX|  8324 Little Common|    San Marcos|           CA|          92069|
|          5|        Robert|        Hudson|     XXXXXXXXX|        XXXXXXXXX|10 Crystal River ...|        Caguas|           PR|          00725|
|          6|          Mary|         Smith|     XXXXXXXXX|        XXXXXXXXX|3151 Sleepy Quail...|       Passaic|           NJ|          07055|
|          7|       Melissa|        Wilcox|     XXXXXXXXX|        XXXXXXXXX|9453 High Concession|        Caguas|           PR|          00725|
|          8|         Megan|         Smith|     XXXXXXXXX|        XXXXXXXXX|3047 Foggy Forest...|      Lawrence|           MA|          01841|
|          9|          Mary|         Perez|     XXXXXXXXX|        XXXXXXXXX| 3616 Quaking Street|        Caguas|           PR|          00725|
|         10|       Melissa|         Smith|     XXXXXXXXX|        XXXXXXXXX|8598 Harvest Beac...|      Stafford|           VA|          22554|
|         11|          Mary|       Huffman|     XXXXXXXXX|        XXXXXXXXX|    3169 Stony Woods|        Caguas|           PR|          00725|
|         12|   Christopher|         Smith|     XXXXXXXXX|        XXXXXXXXX|5594 Jagged Ember...|   San Antonio|           TX|          78227|
|         13|          Mary|       Baldwin|     XXXXXXXXX|        XXXXXXXXX|7922 Iron Oak Gar...|        Caguas|           PR|          00725|
|         14|     Katherine|         Smith|     XXXXXXXXX|        XXXXXXXXX|5666 Hazy Pony Sq...|   Pico Rivera|           CA|          90660|
|         15|          Jane|          Luna|     XXXXXXXXX|        XXXXXXXXX|    673 Burning Glen|       Fontana|           CA|          92336|
|         16|       Tiffany|         Smith|     XXXXXXXXX|        XXXXXXXXX|      6651 Iron Port|        Caguas|           PR|          00725|
|         17|          Mary|      Robinson|     XXXXXXXXX|        XXXXXXXXX|     1325 Noble Pike|        Taylor|           MI|          48180|
|         18|        Robert|         Smith|     XXXXXXXXX|        XXXXXXXXX|2734 Hazy Butterf...|      Martinez|           CA|          94553|
|         19|     Stephanie|      Mitchell|     XXXXXXXXX|        XXXXXXXXX|3543 Red Treasure...|        Caguas|           PR|          00725|
|         20|          Mary|         Ellis|     XXXXXXXXX|        XXXXXXXXX|      4703 Old Route| West New York|           NJ|          07093|
+-----------+--------------+--------------+--------------+-----------------+--------------------+--------------+-------------+---------------+
```

PREGUNTA 3

val orders_avro = spark.read.format("avro").load("/user/vagrant/lab1/pregunta1/resultado")

orders_avro.select("order_id","order_status").write.option("compression", "gzip").parquet("/user/vagrant/lab1/pregunta3/resultado")



```
[vagrant@localhost ~]$ hdfs dfs -ls /user/vagrant/lab1/pregunta3/resultado
Found 2 items
-rw-r--r--   1 vagrant supergroup          0 2022-10-09 23:42 /user/vagrant/lab1/pregunta3/resultado/_SUCCESS
-rw-r--r--   1 vagrant supergroup     120532 2022-10-09 23:42 /user/vagrant/lab1/pregunta3/resultado/part-00000-f1d08e34-83b0-4947-ada4-8a54db9be218-c000.gz.parquet
```

PREGUNTA 4

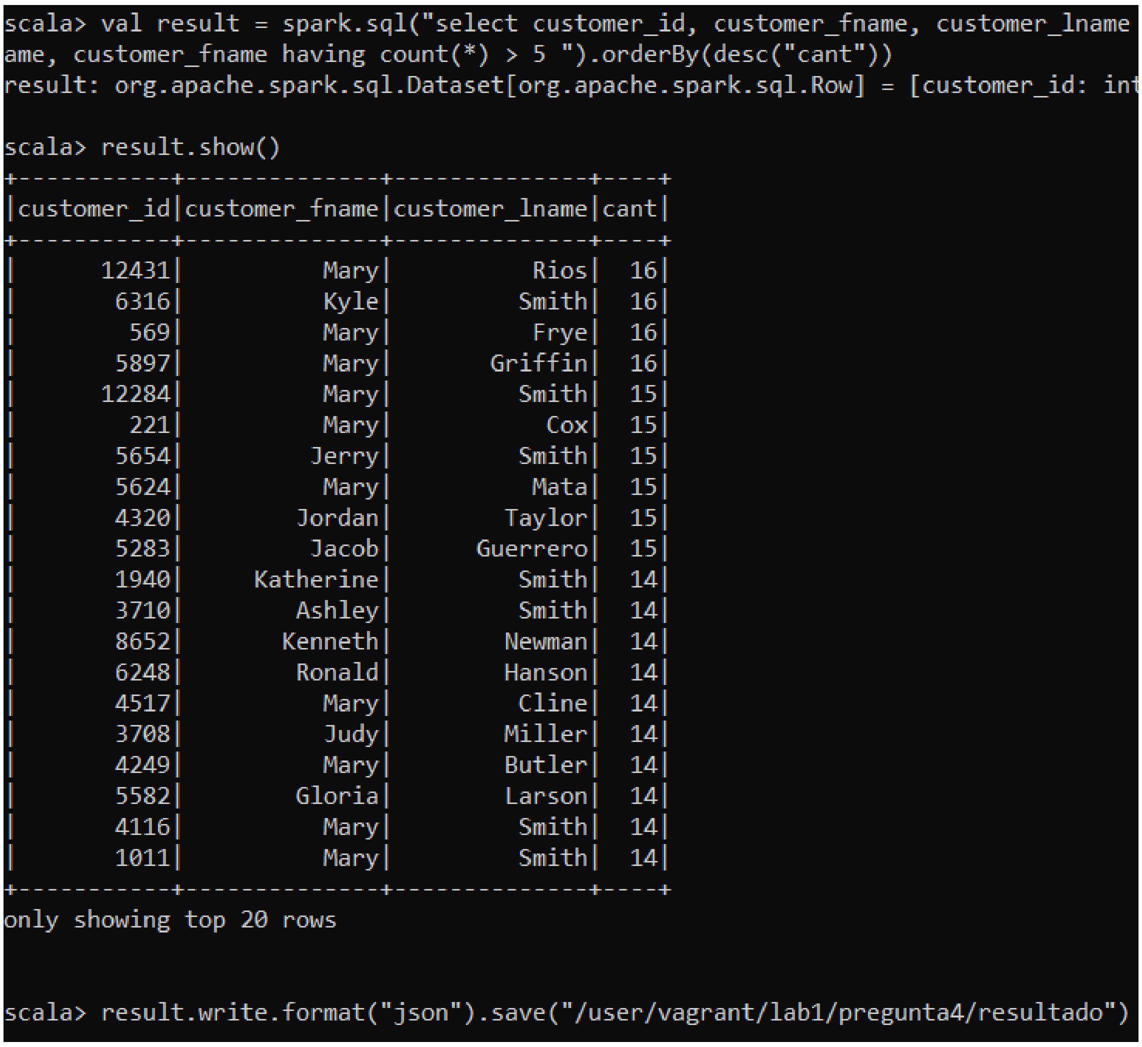
val customer = spark.read.format("parquet").load("/user/vagrant/lab1/pregunta2/resultado")

val order = spark.read.format("avro").load("/user/vagrant/lab1/pregunta1/resultado") customer.createOrReplaceTempView("customer")

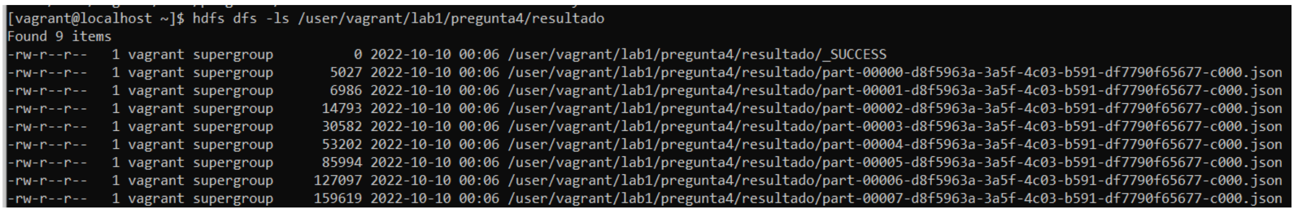
order.createOrReplaceTempView("orders_avro")

val result = spark.sql("select customer_id, customer_fname, customer_lname ,count(*) as cant from customer a inner join orders_avro b on a.customer_id = order_customer_id group by customer_id, customer_lname, customer_fname having count(*) > 5 ").orderBy(desc("cant"))

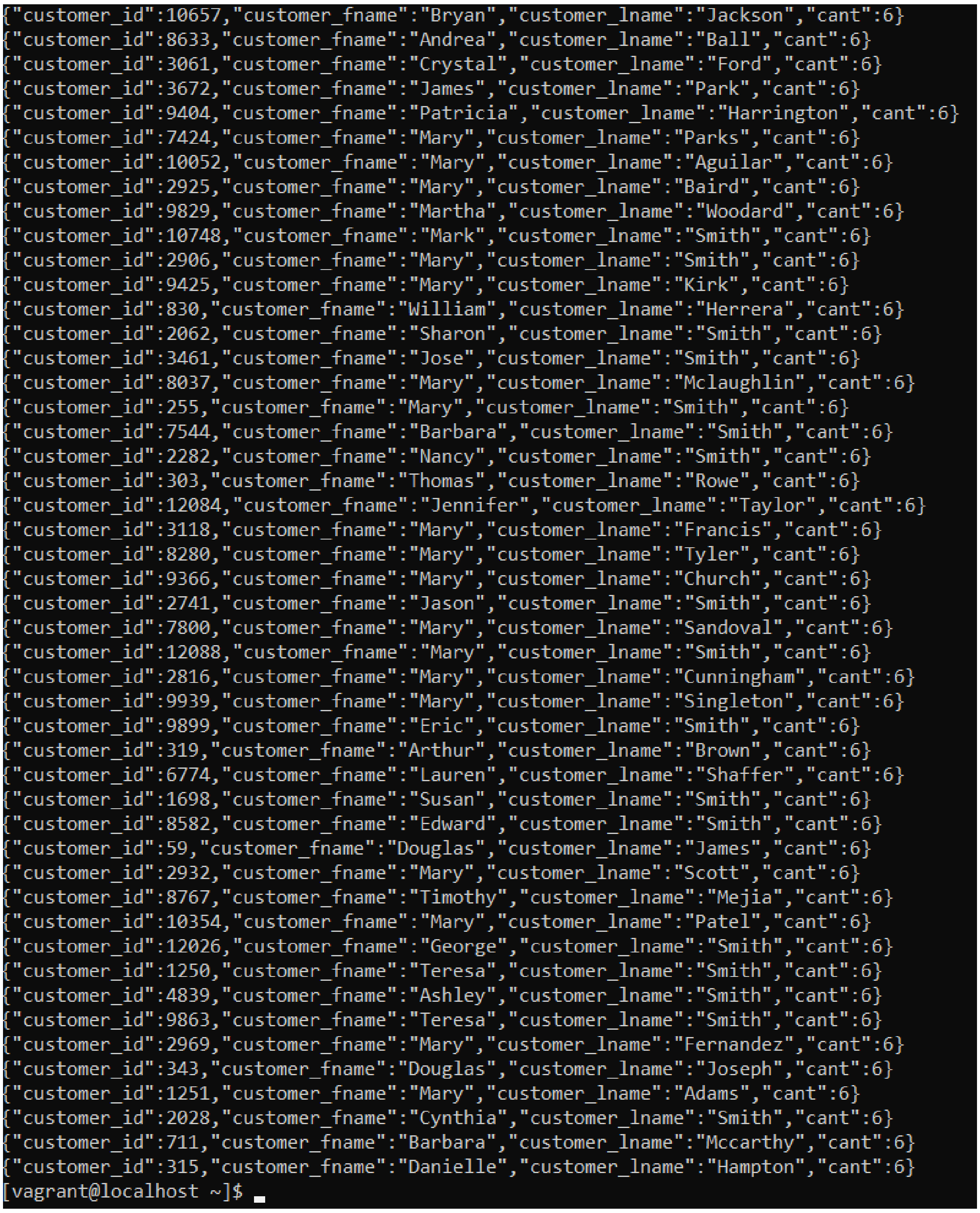
result.write.format("json").save("/user/vagrant/lab1/pregunta4/resultado")



```
scala> val result = spark.sql("select customer_id, customer_fname, customer_lname
ame, customer_fname having count(*) > 5 ").orderBy(desc("cant"))
result: org.apache.spark.sql.Dataset[org.apache.spark.sql.Row] = [customer_id: int

scala> result.show()
+-----------+--------------+--------------+----+
|customer_id|customer_fname|customer_lname|cant|
+-----------+--------------+--------------+----+
|      12431|          Mary|          Rios|  16|
|       6316|          Kyle|         Smith|  16|
|        569|          Mary|          Frye|  16|
|       5897|          Mary|       Griffin|  16|
|      12284|          Mary|         Smith|  15|
|        221|          Mary|           Cox|  15|
|       5654|         Jerry|         Smith|  15|
|       5624|          Mary|          Mata|  15|
|       4320|        Jordan|        Taylor|  15|
|       5283|         Jacob|      Guerrero|  15|
|       1940|     Katherine|         Smith|  14|
|       3710|        Ashley|         Smith|  14|
|       8652|       Kenneth|        Newman|  14|
|       6248|        Ronald|        Hanson|  14|
|       4517|          Mary|         Cline|  14|
|       3708|          Judy|        Miller|  14|
|       4249|          Mary|        Butler|  14|
|       5582|        Gloria|        Larson|  14|
|       4116|          Mary|         Smith|  14|
|       1011|          Mary|         Smith|  14|
+-----------+--------------+--------------+----+
only showing top 20 rows


scala> result.write.format("json").save("/user/vagrant/lab1/pregunta4/resultado")
```



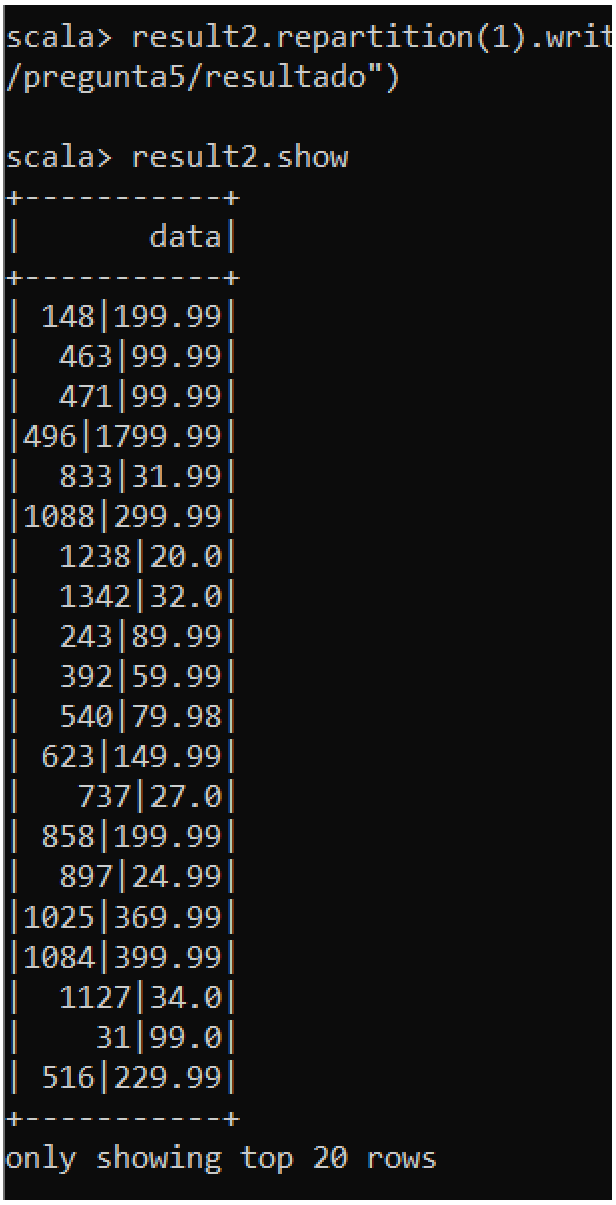
```
[vagrant@localhost ~]$ hdfs dfs -ls /user/vagrant/lab1/pregunta4/resultado
Found 9 items
-rw-r--r--   1 vagrant supergroup          0 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/_SUCCESS
-rw-r--r--   1 vagrant supergroup       5027 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00000-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup       6986 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00001-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup      14793 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00002-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup      30582 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00003-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup      53202 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00004-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup      85994 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00005-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup     127097 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00006-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
-rw-r--r--   1 vagrant supergroup     159619 2022-10-10 00:06 /user/vagrant/lab1/pregunta4/resultado/part-00007-d8f5963a-3a5f-4c03-b591-df7790f65677-c000.json
```



```
{"customer_id":10657,"customer_fname":"Bryan","customer_lname":"Jackson","cant":6}
{"customer_id":8633,"customer_fname":"Andrea","customer_lname":"Ball","cant":6}
{"customer_id":3061,"customer_fname":"Crystal","customer_lname":"Ford","cant":6}
{"customer_id":3672,"customer_fname":"James","customer_lname":"Park","cant":6}
{"customer_id":9404,"customer_fname":"Patricia","customer_lname":"Harrington","cant":6}
{"customer_id":7424,"customer_fname":"Mary","customer_lname":"Parks","cant":6}
{"customer_id":10052,"customer_fname":"Mary","customer_lname":"Aguilar","cant":6}
{"customer_id":2925,"customer_fname":"Mary","customer_lname":"Baird","cant":6}
{"customer_id":9829,"customer_fname":"Martha","customer_lname":"Woodard","cant":6}
{"customer_id":10748,"customer_fname":"Mark","customer_lname":"Smith","cant":6}
{"customer_id":2906,"customer_fname":"Mary","customer_lname":"Smith","cant":6}
{"customer_id":9425,"customer_fname":"Mary","customer_lname":"Kirk","cant":6}
{"customer_id":830,"customer_fname":"William","customer_lname":"Herrera","cant":6}
{"customer_id":2062,"customer_fname":"Sharon","customer_lname":"Smith","cant":6}
{"customer_id":3461,"customer_fname":"Jose","customer_lname":"Smith","cant":6}
{"customer_id":8037,"customer_fname":"Mary","customer_lname":"Mclaughlin","cant":6}
{"customer_id":255,"customer_fname":"Mary","customer_lname":"Smith","cant":6}
{"customer_id":7544,"customer_fname":"Barbara","customer_lname":"Smith","cant":6}
{"customer_id":2282,"customer_fname":"Nancy","customer_lname":"Smith","cant":6}
{"customer_id":303,"customer_fname":"Thomas","customer_lname":"Rowe","cant":6}
{"customer_id":12084,"customer_fname":"Jennifer","customer_lname":"Taylor","cant":6}
{"customer_id":3118,"customer_fname":"Mary","customer_lname":"Francis","cant":6}
{"customer_id":8280,"customer_fname":"Mary","customer_lname":"Tyler","cant":6}
{"customer_id":9366,"customer_fname":"Mary","customer_lname":"Church","cant":6}
{"customer_id":2741,"customer_fname":"Jason","customer_lname":"Smith","cant":6}
{"customer_id":7800,"customer_fname":"Mary","customer_lname":"Sandoval","cant":6}
{"customer_id":12088,"customer_fname":"Mary","customer_lname":"Smith","cant":6}
{"customer_id":2816,"customer_fname":"Mary","customer_lname":"Cunningham","cant":6}
{"customer_id":9939,"customer_fname":"Mary","customer_lname":"Singleton","cant":6}
{"customer_id":9899,"customer_fname":"Eric","customer_lname":"Smith","cant":6}
{"customer_id":319,"customer_fname":"Arthur","customer_lname":"Brown","cant":6}
{"customer_id":6774,"customer_fname":"Lauren","customer_lname":"Shaffer","cant":6}
{"customer_id":1698,"customer_fname":"Susan","customer_lname":"Smith","cant":6}
{"customer_id":8582,"customer_fname":"Edward","customer_lname":"Smith","cant":6}
{"customer_id":59,"customer_fname":"Douglas","customer_lname":"James","cant":6}
{"customer_id":2932,"customer_fname":"Mary","customer_lname":"Scott","cant":6}
{"customer_id":8767,"customer_fname":"Timothy","customer_lname":"Mejia","cant":6}
{"customer_id":10354,"customer_fname":"Mary","customer_lname":"Patel","cant":6}
{"customer_id":12026,"customer_fname":"George","customer_lname":"Smith","cant":6}
{"customer_id":1250,"customer_fname":"Teresa","customer_lname":"Smith","cant":6}
{"customer_id":4839,"customer_fname":"Ashley","customer_lname":"Smith","cant":6}
{"customer_id":9863,"customer_fname":"Teresa","customer_lname":"Smith","cant":6}
{"customer_id":2969,"customer_fname":"Mary","customer_lname":"Fernandez","cant":6}
{"customer_id":343,"customer_fname":"Douglas","customer_lname":"Joseph","cant":6}
{"customer_id":1251,"customer_fname":"Mary","customer_lname":"Adams","cant":6}
{"customer_id":2028,"customer_fname":"Cynthia","customer_lname":"Smith","cant":6}
{"customer_id":711,"customer_fname":"Barbara","customer_lname":"Mccarthy","cant":6}
{"customer_id":315,"customer_fname":"Danielle","customer_lname":"Hampton","cant":6}
[vagrant@localhost ~]$
```

PREGUNTA 5

```
val ProductSchema = StructType(Array(

 StructField("product_id", IntegerType, true),

 StructField("product_category_id", IntegerType, true),

 StructField("product_name", StringType, true),

 StructField("product_description", StringType, true),

 StructField("product_price", FloatType, true),
 StructField("product_image", StringType, true)
 )
)

val product = spark.read.format("csv").option("inferSchema",
"true").schema(ProductSchema).load("/public/retail_db/products/part-00000")

product.createOrReplaceTempView("product")

val result =spark.sql("select product_id, max(product_price) as max_price from product group
by product_id")

result.createOrReplaceTempView("result")

val result2 =spark.sql("select concat(product_id, '|', max_price) as data from result ")

result2.repartition(1).write.option("compression","gzip").format("text").save("/user/vagrant/l
ab1/pregunta5/resultado")
```



```
scala> result2.repartition(1).writ
/pregunta5/resultado")

scala> result2.show
+-----------+
|       data|
+-----------+
| 148|199.99|
|  463|99.99|
|  471|99.99|
|496|1799.99|
|  833|31.99|
|1088|299.99|
|  1238|20.0|
|  1342|32.0|
|  243|89.99|
|  392|59.99|
|  540|79.98|
| 623|149.99|
|   737|27.0|
| 858|199.99|
|  897|24.99|
|1025|369.99|
|1084|399.99|
|  1127|34.0|
|    31|99.0|
| 516|229.99|
+-----------+
only showing top 20 rows
```

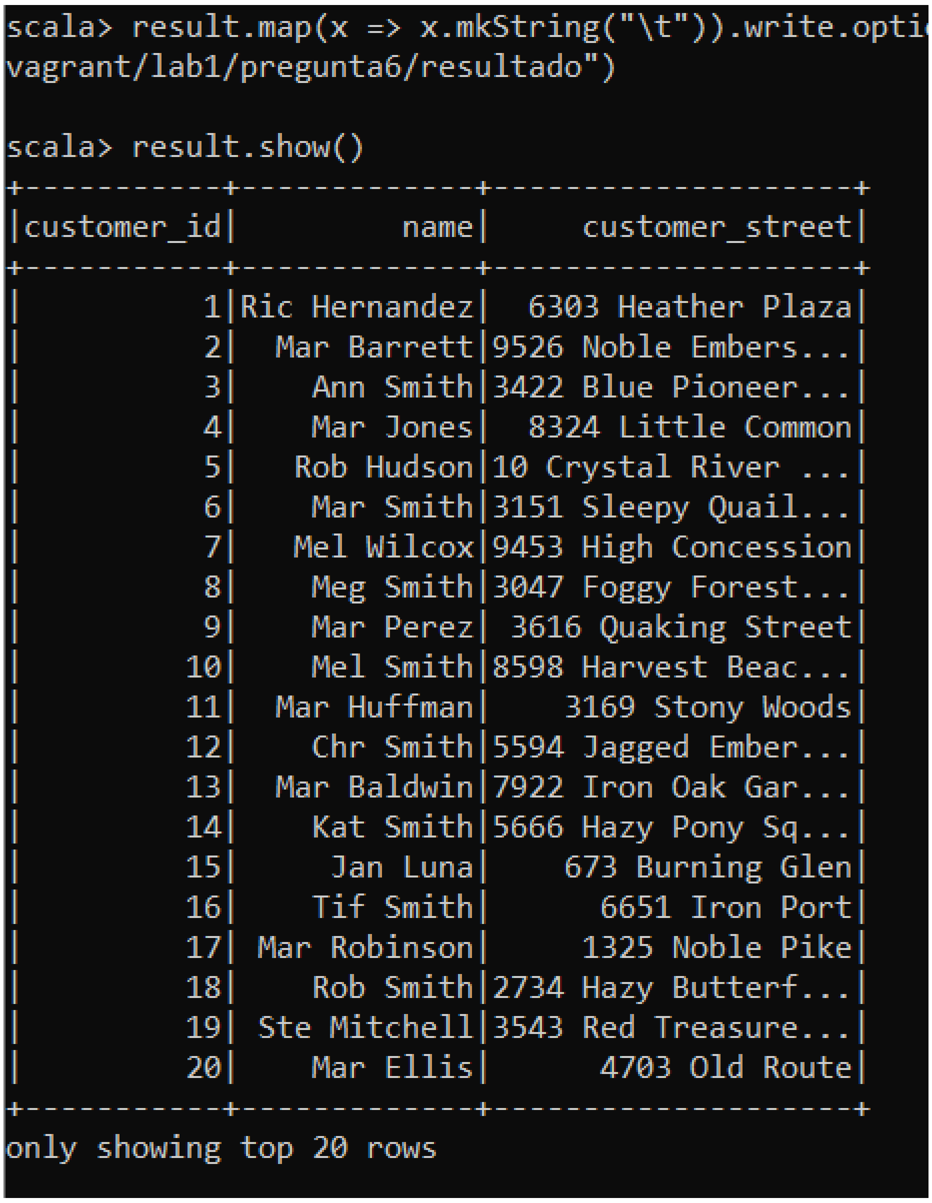
PREGUNTA 6

val customer = spark.read.format("parquet").load("/user/vagrant/lab1/pregunta2/resultado")

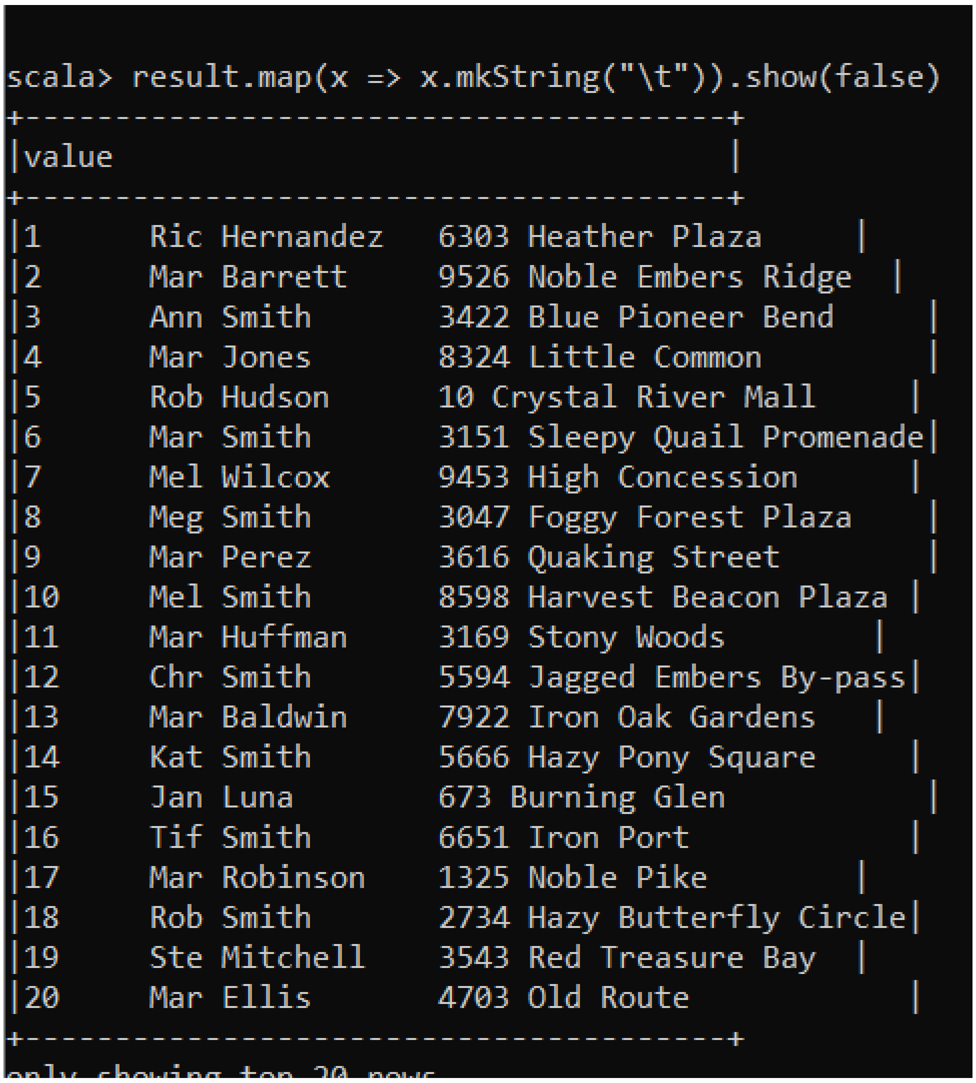
customer.createOrReplaceTempView("customer")

val result = spark.sql("select customer_id, concat(substring(customer_fname,1,3),' ', customer_lname) as name , customer_street from customer")

result.map(x =>
x.mkString("\t")).write.option("compression","bzip2").format("text").save("/user/vagrant/lab1
/pregunta6/resultado")



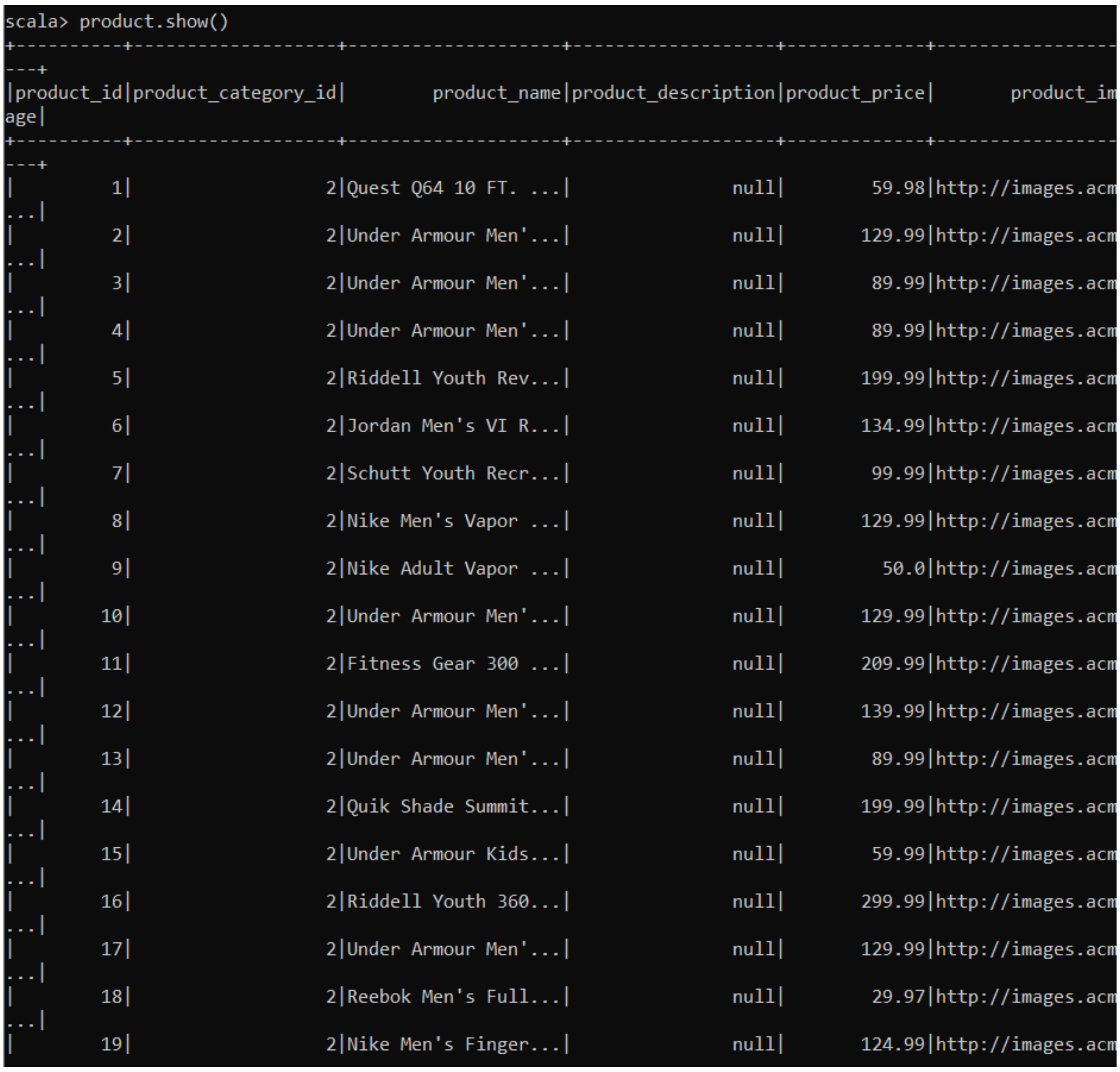
```
scala> result.map(x => x.mkString("\t")).write.opti
vagrant/lab1/pregunta6/resultado")

scala> result.show()
+-----------+------------+--------------------+
|customer_id|        name|     customer_street|
+-----------+------------+--------------------+
|          1|Ric Hernandez|  6303 Heather Plaza|
|          2| Mar Barrett|9526 Noble Embers...|
|          3|   Ann Smith|3422 Blue Pioneer...|
|          4|   Mar Jones|  8324 Little Common|
|          5|  Rob Hudson|10 Crystal River ...|
|          6|   Mar Smith|3151 Sleepy Quail...|
|          7|  Mel Wilcox|9453 High Concession|
|          8|   Meg Smith|3047 Foggy Forest...|
|          9|   Mar Perez| 3616 Quaking Street|
|         10|   Mel Smith|8598 Harvest Beac...|
|         11| Mar Huffman|     3169 Stony Woods|
|         12|   Chr Smith|5594 Jagged Ember...|
|         13| Mar Baldwin|7922 Iron Oak Gar...|
|         14|   Kat Smith|5666 Hazy Pony Sq...|
|         15|    Jan Luna|     673 Burning Glen|
|         16|   Tif Smith|       6651 Iron Port|
|         17|Mar Robinson|      1325 Noble Pike|
|         18|   Rob Smith|2734 Hazy Butterf...|
|         19|Ste Mitchell|3543 Red Treasure...|
|         20|   Mar Ellis|       4703 Old Route|
+-----------+------------+--------------------+
only showing top 20 rows
```



```
scala> result.map(x => x.mkString("\t")).show(false)
+---------------------------------------+
|value                                  |
+---------------------------------------+
|1      Ric Hernandez   6303 Heather Plaza     |
|2      Mar Barrett     9526 Noble Embers Ridge  |
|3      Ann Smith       3422 Blue Pioneer Bend     |
|4      Mar Jones       8324 Little Common         |
|5      Rob Hudson      10 Crystal River Mall     |
|6      Mar Smith       3151 Sleepy Quail Promenade|
|7      Mel Wilcox      9453 High Concession      |
|8      Meg Smith       3047 Foggy Forest Plaza    |
|9      Mar Perez       3616 Quaking Street        |
|10     Mel Smith       8598 Harvest Beacon Plaza |
|11     Mar Huffman     3169 Stony Woods        |
|12     Chr Smith       5594 Jagged Embers By-pass|
|13     Mar Baldwin     7922 Iron Oak Gardens   |
|14     Kat Smith       5666 Hazy Pony Square     |
|15     Jan Luna        673 Burning Glen           |
|16     Tif Smith       6651 Iron Port            |
|17     Mar Robinson    1325 Noble Pike        |
|18     Rob Smith       2734 Hazy Butterfly Circle|
|19     Ste Mitchell    3543 Red Treasure Bay  |
|20     Mar Ellis       4703 Old Route            |
+---------------------------------------+
only showing top 20 rows
```

PREGUNTA 7

```
val product = spark.read.format("csv").option("inferSchema",
"true").schema(ProductSchema).load("/public/retail_db/products/part-00000")

product.write.format("hive").saveAsTable("product")
```



```
scala> product.show()
+----------+-------------------+--------------------+-------------------+-------------+-----------------
---+
|product_id|product_category_id|        product_name|product_description|product_price|       product_im
age|
+----------+-------------------+--------------------+-------------------+-------------+-----------------
---+
|         1|                  2|Quest Q64 10 FT. ...|               null|        59.98|http://images.acm
...|
|         2|                  2|Under Armour Men'...|               null|       129.99|http://images.acm
...|
|         3|                  2|Under Armour Men'...|               null|        89.99|http://images.acm
...|
|         4|                  2|Under Armour Men'...|               null|        89.99|http://images.acm
...|
|         5|                  2|Riddell Youth Rev...|               null|       199.99|http://images.acm
...|
|         6|                  2|Jordan Men's VI R...|               null|       134.99|http://images.acm
...|
|         7|                  2|Schutt Youth Recr...|               null|        99.99|http://images.acm
...|
|         8|                  2|Nike Men's Vapor ...|               null|       129.99|http://images.acm
...|
|         9|                  2|Nike Adult Vapor ...|               null|         50.0|http://images.acm
...|
|        10|                  2|Under Armour Men'...|               null|       129.99|http://images.acm
...|
|        11|                  2|Fitness Gear 300 ...|               null|       209.99|http://images.acm
...|
|        12|                  2|Under Armour Men'...|               null|       139.99|http://images.acm
...|
|        13|                  2|Under Armour Men'...|               null|        89.99|http://images.acm
...|
|        14|                  2|Quik Shade Summit...|               null|       199.99|http://images.acm
...|
|        15|                  2|Under Armour Kids...|               null|        59.99|http://images.acm
...|
|        16|                  2|Riddell Youth 360...|               null|       299.99|http://images.acm
...|
|        17|                  2|Under Armour Men'...|               null|       129.99|http://images.acm
...|
|        18|                  2|Reebok Men's Full...|               null|        29.97|http://images.acm
...|
|        19|                  2|Nike Men's Finger...|               null|       124.99|http://images.acm
```

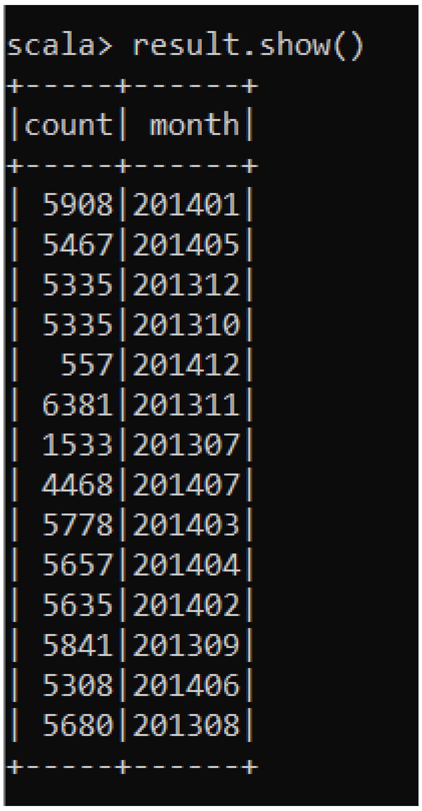
PREGUNTA 8

val orders = spark.read.format("csv").option("inferSchema", "true").schema(customSchema).load("/public/retail_db/orders/part 00000")

orders.write.format("hive").saveAsTable("orders")

val result = spark.sql("select count(*) as count,date_format(order_date,'YYYYMM') as month from orders group by date_format(order_date, 'YYYYMM')")

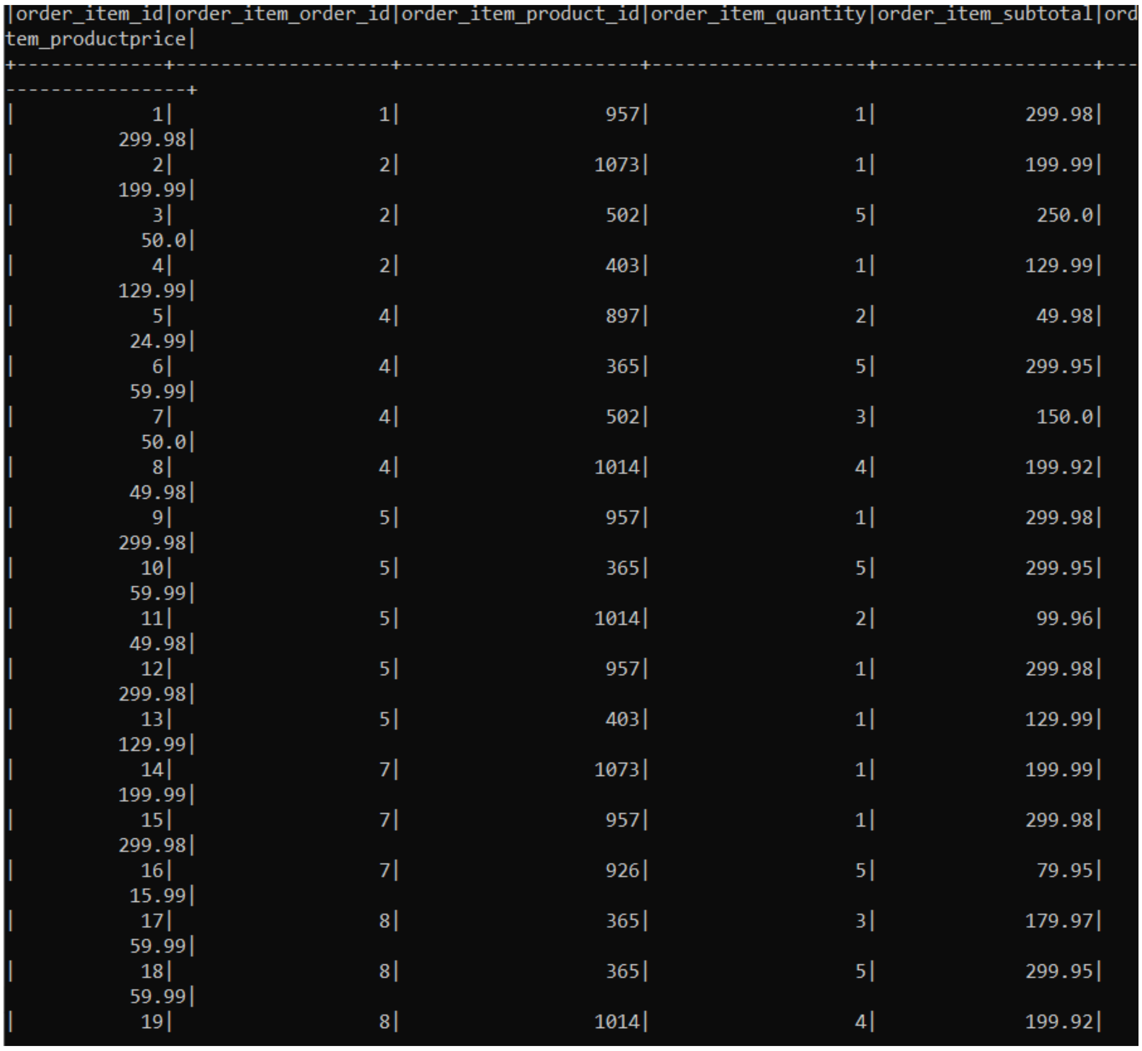
result.write.option("compression","uncompressed").format("parquet").save("/user/vagrant/lab1/pregunta8/resultado")



```
scala> result.show()
+-----+------+
|count| month|
+-----+------+
| 5908|201401|
| 5467|201405|
| 5335|201312|
| 5335|201310|
|  557|201412|
| 6381|201311|
| 1533|201307|
| 4468|201407|
| 5778|201403|
| 5657|201404|
| 5635|201402|
| 5841|201309|
| 5308|201406|
| 5680|201308|
+-----+------+
```

PREGUNTA 9

```
val itemsSchema = StructType(Array(
 StructField("order_item_id", IntegerType, true),
 StructField("order_item_order_id", IntegerType, true),
 StructField("order_item_product_id", IntegerType, true),
 StructField("order_item_quantity", IntegerType, true),
 StructField("order_item_subtotal", FloatType, true),
 StructField("order_item_productprice", FloatType, true)
 )
)
val order_items= spark.read.format("csv").option("inferSchema",
"true").schema(itemsSchema).load("/public/retail_db/order_items/part-00000")
order_items.write.format("orc").option("compression","uncompressed").save("/user/vagrant/
lab1/pregunta9/resultado")
```



```
|order_item_id|order_item_order_id|order_item_product_id|order_item_quantity|order_item_subtotal|ord
tem_productprice|
+-------------+-------------------+---------------------+-------------------+-------------------+---
----------------+
|            1|                  1|                  957|                  1|             299.98|
          299.98|
|            2|                  2|                 1073|                  1|             199.99|
          199.99|
|            3|                  2|                  502|                  5|              250.0|
            50.0|
|            4|                  2|                  403|                  1|             129.99|
          129.99|
|            5|                  4|                  897|                  2|              49.98|
           24.99|
|            6|                  4|                  365|                  5|             299.95|
           59.99|
|            7|                  4|                  502|                  3|              150.0|
            50.0|
|            8|                  4|                 1014|                  4|             199.92|
           49.98|
|            9|                  5|                  957|                  1|             299.98|
          299.98|
|           10|                  5|                  365|                  5|             299.95|
           59.99|
|           11|                  5|                 1014|                  2|              99.96|
           49.98|
|           12|                  5|                  957|                  1|             299.98|
          299.98|
|           13|                  5|                  403|                  1|             129.99|
          129.99|
|           14|                  7|                 1073|                  1|             199.99|
          199.99|
|           15|                  7|                  957|                  1|             299.98|
          299.98|
|           16|                  7|                  926|                  5|              79.95|
           15.99|
|           17|                  8|                  365|                  3|             179.97|
           59.99|
|           18|                  8|                  365|                  5|             299.95|
           59.99|
|           19|                  8|                 1014|                  4|             199.92|
```

PREGUNTA 10

val customer =

spark.read.format("parquet").load("/user/vagrant/lab1/pregunta2/customer")

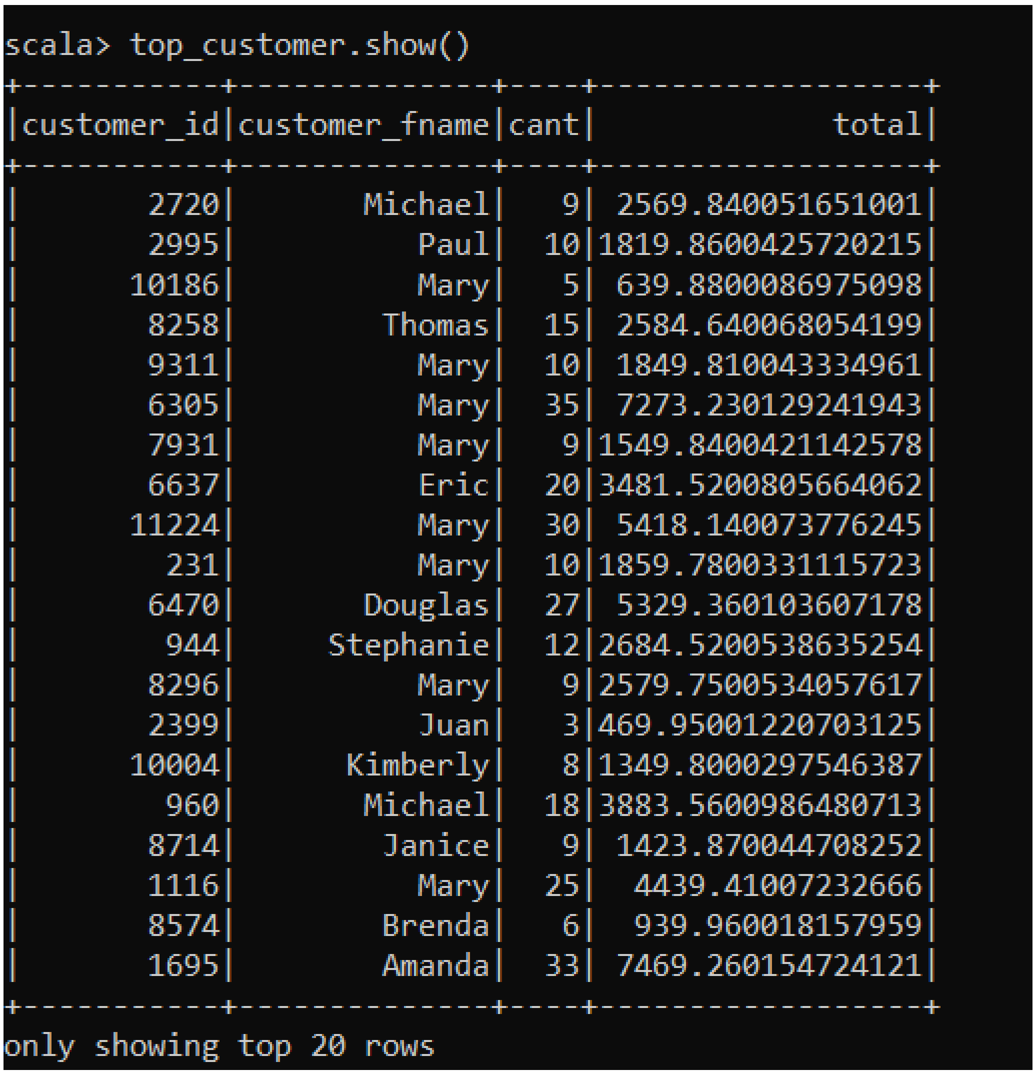
customer.createOrReplaceTempView("customer")

val order_items = spark.read.format("orc").load("/user/vagrant/lab1/pregunta9/resultado")

order_items.createOrReplaceTempView("order_items")

val top_customer = spark.sql("select customer_id, customer_fname, count(*) as cant, sum(order_item_subtotal) as total from customer a inner join orders b on a.customer_id = b.order_customer_id inner join order_items c on c.order_item_order_id = b.order_id where customer_city like 'M%' group by customer_id, customer_fname")

top_customer.write.format("parquet").option("compression","gzip").save("/user/vagrant/lab1
/pregunta10/resultado")



```
scala> top_customer.show()
+-----------+--------------+----+------------------+
|customer_id|customer_fname|cant|             total|
+-----------+--------------+----+------------------+
|       2720|       Michael|   9| 2569.840051651001|
|       2995|          Paul|  10|1819.8600425720215|
|      10186|          Mary|   5| 639.8800086975098|
|       8258|        Thomas|  15| 2584.640068054199|
|       9311|          Mary|  10| 1849.810043334961|
|       6305|          Mary|  35| 7273.230129241943|
|       7931|          Mary|   9|1549.8400421142578|
|       6637|          Eric|  20|3481.5200805664062|
|      11224|          Mary|  30| 5418.140073776245|
|        231|          Mary|  10|1859.7800331115723|
|       6470|       Douglas|  27| 5329.360103607178|
|        944|     Stephanie|  12|2684.5200538635254|
|       8296|          Mary|   9|2579.7500534057617|
|       2399|          Juan|   3|469.95001220703125|
|      10004|      Kimberly|   8|1349.8000297546387|
|        960|       Michael|  18|3883.5600986480713|
|       8714|        Janice|   9| 1423.870044708252|
|       1116|          Mary|  25|  4439.41007232666|
|       8574|        Brenda|   6|  939.960018157959|
|       1695|        Amanda|  33| 7469.260154724121|
+-----------+--------------+----+------------------+
only showing top 20 rows
```