Case Study: Instacart
Domain: E-commerce
Instacart is a grocery ordering and delivery app that aims to make it easy to fill your refrigerator and pantry with your personal favorites and staples when you need them. Instacart's data science team plays a big part in providing this delightful shopping experience. Currently , they use transactional data to develop models that predict which products a user will buy again, try for the first time, or add to their cart next during a session.
The dataset is a relational set of files describing customers' orders over time. The dataset is anonymized and contains a sample of over 3 million grocery orders from more than 200,000 Instacart users.
Tasks:
As a Big Data consultant, you are helping the data science team to explore the dataset using Spark:

1. **Load data into Spark DataFrame**

```
ordersDF =
spark.read.csv("/user/edureka_524533/Datasets/orders.csv",inferSchema=True,header=True)

priorDF=spark.read.csv("/user/edureka_524533/Datasets/order_products__prior.csv",inferSchem
a=True,header=True)

trainDF =
spark.read.csv("/user/edureka_524533/Datasets/order_products__train.csv",inferSchema=True,h
eader=True)

prodDF=spark.read.csv("/user/edureka_524533/Datasets/products.csv",inferSchema=True,heade
r=True)

aisleDF=spark.read.csv("/user/edureka_524533/Datasets/aisles.csv",inferSchema=True,header=
True)

deptDF=spark.read.csv("/user/edureka_524533/Datasets/departments.csv",inferSchema=True,he
ader=True)
```

2. **Merge all the data frames based on the common key and create a single DataFrame**

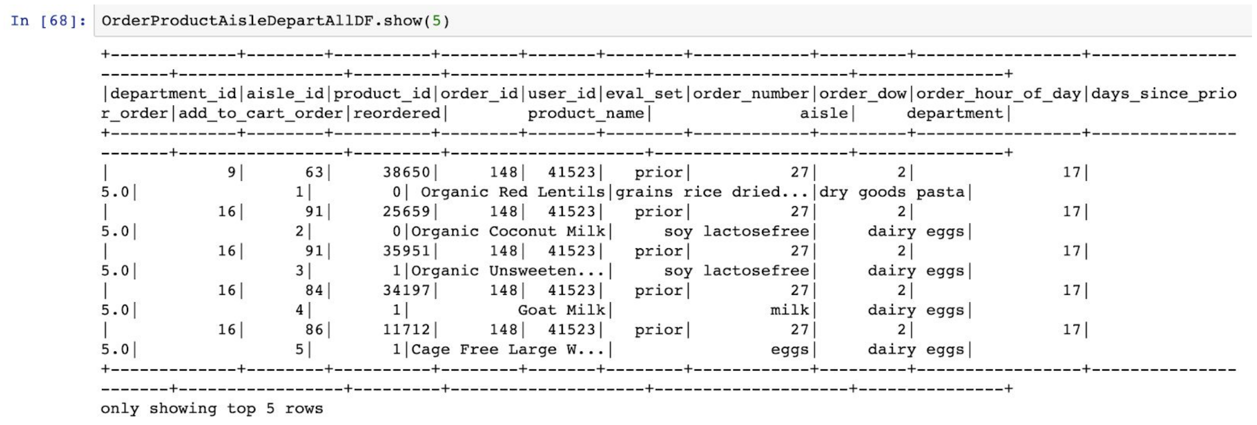
```
# Divide Order DF into three : Prior data, Train data, Test Data
orderPriorDF = ordersDF.where(ordersDF['eval_set']=="prior")
orderTrainDF = ordersDF.where(ordersDF['eval_set']=="train")
orderTestDF = ordersDF.where(ordersDF['eval_set']=="test")

# Now Join each individual Order DF with its counter orders_product*.csv DF
orderPriorJoinDF = orderPriorDF.join(priorDF,on=['order_id'],how='left_outer')
orderTrainJoinDF = orderTrainDF.join(trainDF,on=['order_id'],how='left_outer')
orderTestJoinDF = orderTestDF.join(trainDF,on=['order_id'],how='left_outer')

#Now Join all the three OrderDFs: orderPriorJoin, orderTrainJoin, orderTestJoin
Orders1 = orderPriorJoinDF.unionAll(orderTrainJoinDF)
ordersAllDF=Orders1.unionAll(orderTestJoinDF)
```

# Now we need tp Join this DF with Products.csv
orderProductAllDF = ordersAllDF.join(prodDF,on='product_id',how='left_outer')

# Now Join the aisle details using aisles.csv
orderProductAisleAllDF = orderProductAllDF.join(aisleDF,on='aisle_id',how='left_outer')

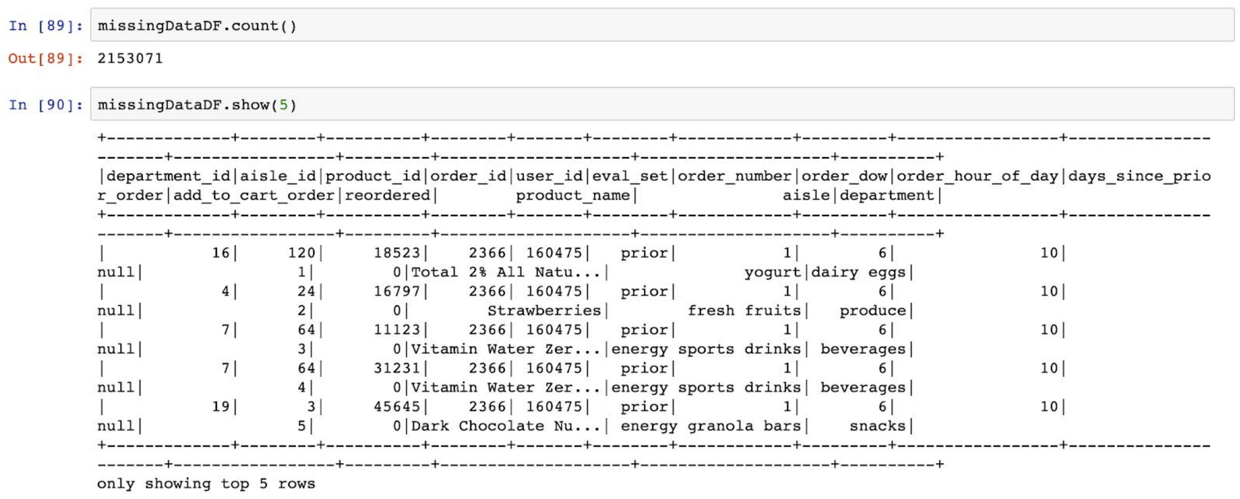
# Now join the department details as well using departments.csv
OrderProductAisleDepartAllDF =
orderProductAisleAllDF.join(deptDF,on='department_id',how='left_outer')

OrderProductAisleDepartAllDF.show(5)



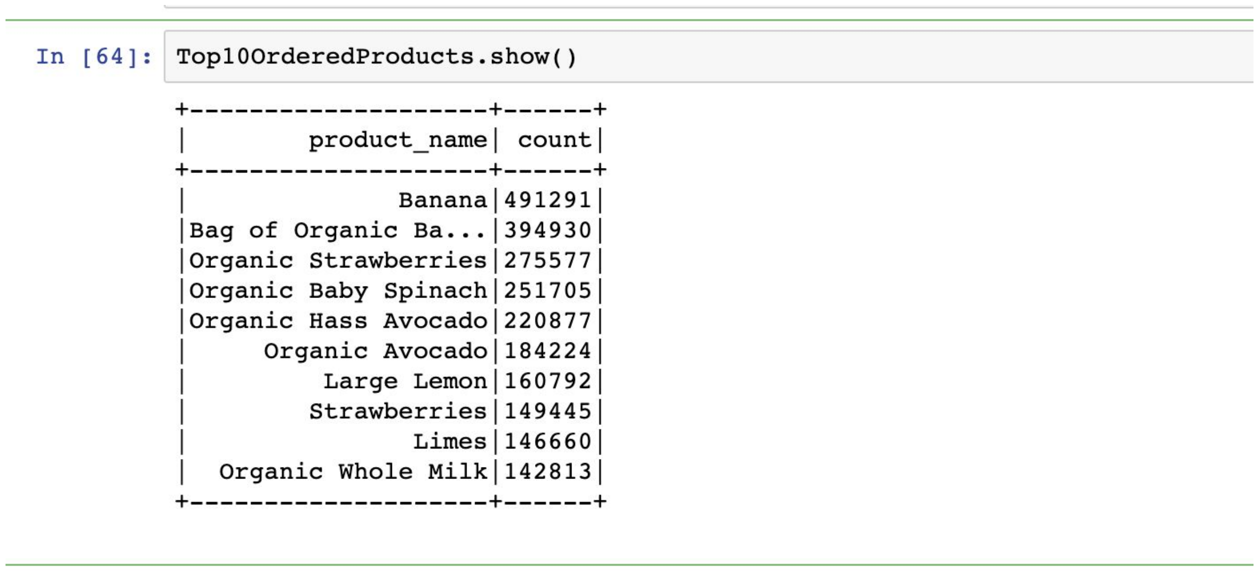
```
In [68]: OrderProductAisleDepartAllDF.show(5)

+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+---------------+
|department_id|aisle_id|product_id|order_id|user_id|eval_set|order_number|order_dow|order_hour_of_day|days_since_prio
r_order|add_to_cart_order|reordered|        product_name|               aisle|     department|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+---------------+
|            9|      63|     38650|     148|  41523|   prior|          27|        2|               17|
5.0|                1|        0| Organic Red Lentils|grains rice dried...|dry goods pasta|
|           16|      91|     25659|     148|  41523|   prior|          27|        2|               17|
5.0|                2|        0|Organic Coconut Milk|     soy lactosefree|     dairy eggs|
|           16|      91|     35951|     148|  41523|   prior|          27|        2|               17|
5.0|                3|        1|Organic Unsweeten...|     soy lactosefree|     dairy eggs|
|           16|      84|     34197|     148|  41523|   prior|          27|        2|               17|
5.0|                4|        1|           Goat Milk|                milk|     dairy eggs|
|           16|      86|     11712|     148|  41523|   prior|          27|        2|               17|
5.0|                5|        1|Cage Free Large W...|                eggs|     dairy eggs|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+---------------+
only showing top 5 rows
```

3. **Check missing data**
missingDataDF =
OrderProductAisleDepartAllDF.filter(OrderProductAisleDepartAllDF['department_id'].isNull()|
OrderProductAisleDepartAllDF['aisle_id'].isNull()|
OrderProductAisleDepartAllDF['product_id'].isNull()|
OrderProductAisleDepartAllDF['order_id'].isNull()|
OrderProductAisleDepartAllDF['user_id'].isNull()|
OrderProductAisleDepartAllDF['eval_set'].isNull()|
OrderProductAisleDepartAllDF['order_number'].isNull()|
OrderProductAisleDepartAllDF['order_dow'].isNull()|
OrderProductAisleDepartAllDF['order_hour_of_day'].isNull()|
OrderProductAisleDepartAllDF['days_since_prior_order'].isNull()|
OrderProductAisleDepartAllDF['add_to_cart_order'].isNull()|
OrderProductAisleDepartAllDF['reordered'].isNull()|
OrderProductAisleDepartAllDF['product_name'].isNull()|
OrderProductAisleDepartAllDF['aisle'].isNull()|
OrderProductAisleDepartAllDF['department'].isNull())



```
In [89]: missingDataDF.count()

Out[89]: 2153071

In [90]: missingDataDF.show(5)

+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+----------+
|department_id|aisle_id|product_id|order_id|user_id|eval_set|order_number|order_dow|order_hour_of_day|days_since_prio
r_order|add_to_cart_order|reordered|        product_name|               aisle|department|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+----------+
|           16|     120|     18523|    2366| 160475|   prior|           1|        6|               10|
null|                1|        0|Total 2% All Natu...|              yogurt|dairy eggs|
|            4|      24|     16797|    2366| 160475|   prior|           1|        6|               10|
null|                2|        0|        Strawberries|        fresh fruits|   produce|
|            7|      64|     11123|    2366| 160475|   prior|           1|        6|               10|
null|                3|        0|Vitamin Water Zer...|energy sports drinks| beverages|
|            7|      64|     31231|    2366| 160475|   prior|           1|        6|               10|
null|                4|        0|Vitamin Water Zer...|energy sports drinks| beverages|
|           19|       3|     45645|    2366| 160475|   prior|           1|        6|               10|
null|                5|        0|Dark Chocolate Nu...| energy granola bars|    snacks|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+---------------
-------+-----------------+---------+--------------------+--------------------+----------+
only showing top 5 rows
```

4. **List the most ordered products (top 10)**
   gpByProductDF = OrderProductAisleDepartAllDF.groupby('product_name').count()
   Top10OrderedProducts = gpByProductDF.orderBy(col('count').desc()).limit(10)
   Top10OrderedProducts.show()



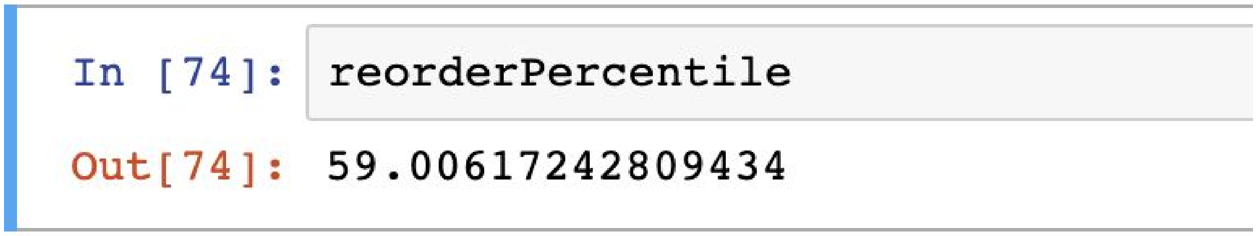
```
In [64]: Top10OrderedProducts.show()

+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|491291|
|Bag of Organic Ba...|394930|
|Organic Strawberries|275577|
|Organic Baby Spinach|251705|
|Organic Hass Avocado|220877|
|     Organic Avocado|184224|
|         Large Lemon|160792|
|        Strawberries|149445|
|               Limes|146660|
|  Organic Whole Milk|142813|
+--------------------+------+
```

5. **Do people usually reorder the same previous ordered products?**
   reorderedCount =
   OrderProductAisleDepartAllDF.where(OrderProductAisleDepartAllDF['reordered']==1).count()

   notReorderedCount =
   OrderProductAisleDepartAllDF.where(OrderProductAisleDepartAllDF['reordered']==0).count()

   Total = reorderedCount + notReorderedCount

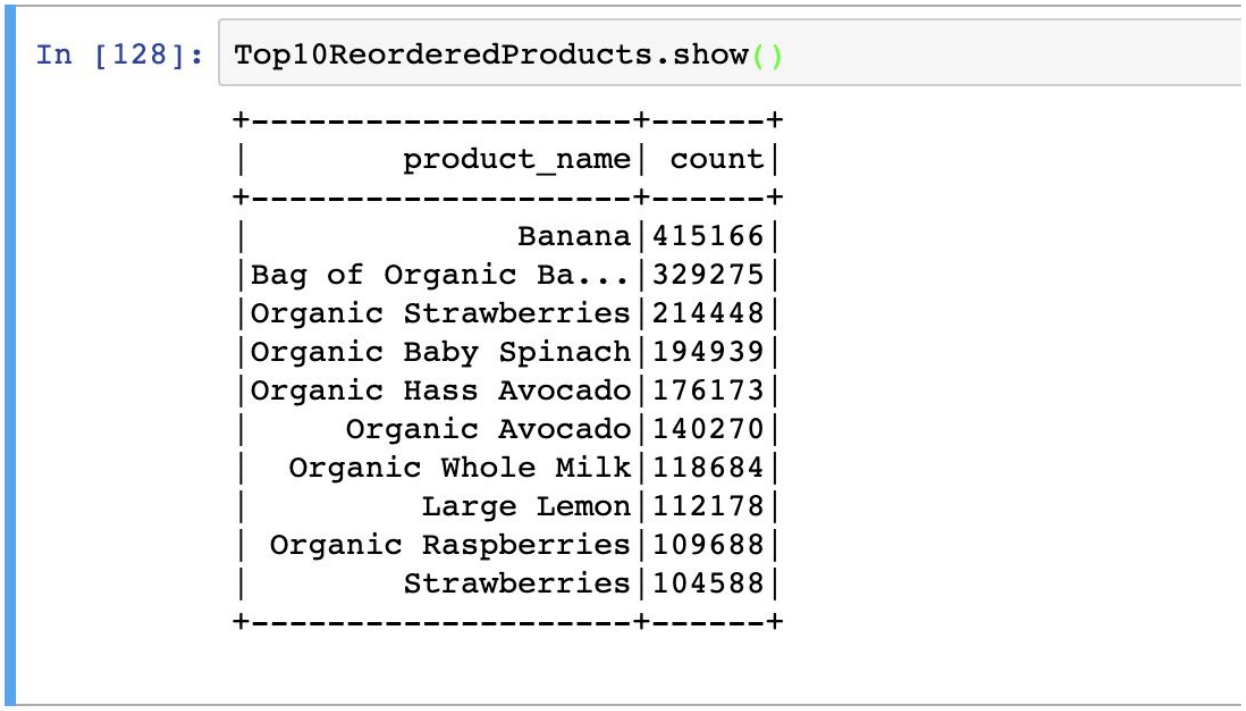
   reorderPercentile = (reorderedCount/Total)*100



```
In [74]: reorderPercentile
Out[74]: 59.00617242809434
```

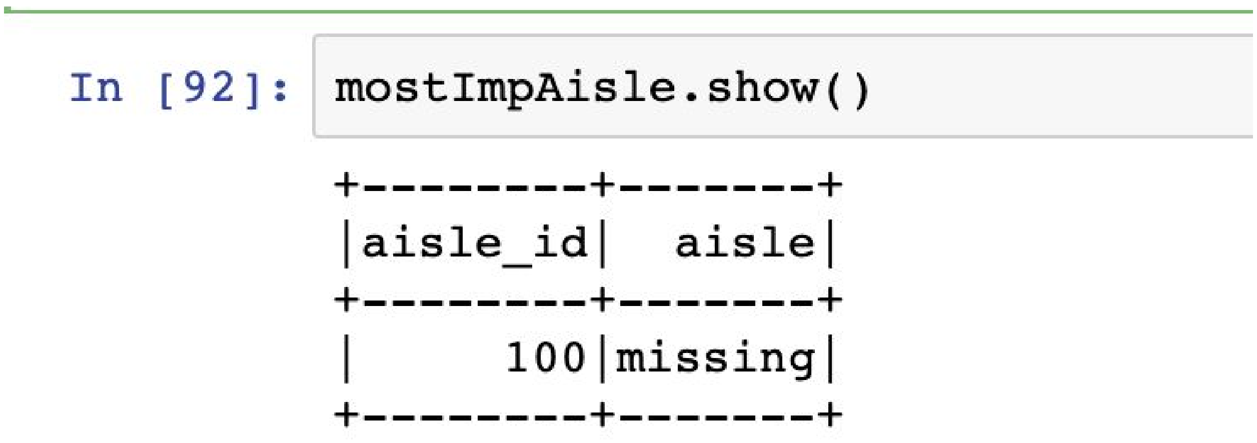
6. **List most reordered products**
   reorderedProdDF =
   OrderProductAisleDepartAllDF.where(OrderProductAisleDepartAllDF['reordered']==1)

   reorderedGpByProdDF = reorderedProdDF.groupby('product_name').count()
   Top10ReorderedProducts = reorderedGpByProdDF.orderBy(col('count').desc()).limit(10)
   Top10ReorderedProducts.show()



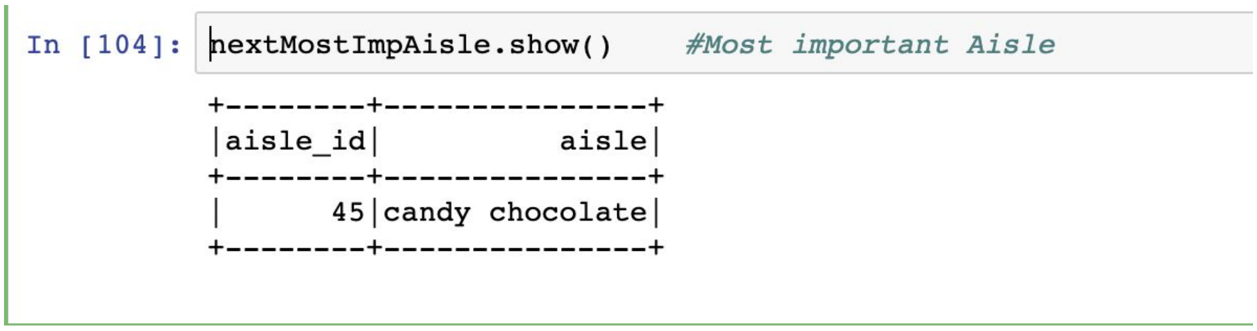
```
In [128]: Top10ReorderedProducts.show()

+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|415166|
|Bag of Organic Ba...|329275|
|Organic Strawberries|214448|
|Organic Baby Spinach|194939|
|Organic Hass Avocado|176173|
|     Organic Avocado|140270|
|  Organic Whole Milk|118684|
|         Large Lemon|112178|
| Organic Raspberries|109688|
|        Strawberries|104588|
+--------------------+------+
```

7. **Most important department and aisle (by number of products)**
   totProdPerAisle = prodDF.groupby('aisle_id').count()
   val = totProdPerAisle.agg({'count':'max'}).collect()[0][0]
   totProdPerAisle.createOrReplaceTempView("AisleCountTab")
   aisle_id = spark.sql("Select * from AisleCountTab where count = {0}".format(val)).collect()[0][0]
   mostImpAisle = aisleDF.where(aisleDF['aisle_id']==aisle_id)



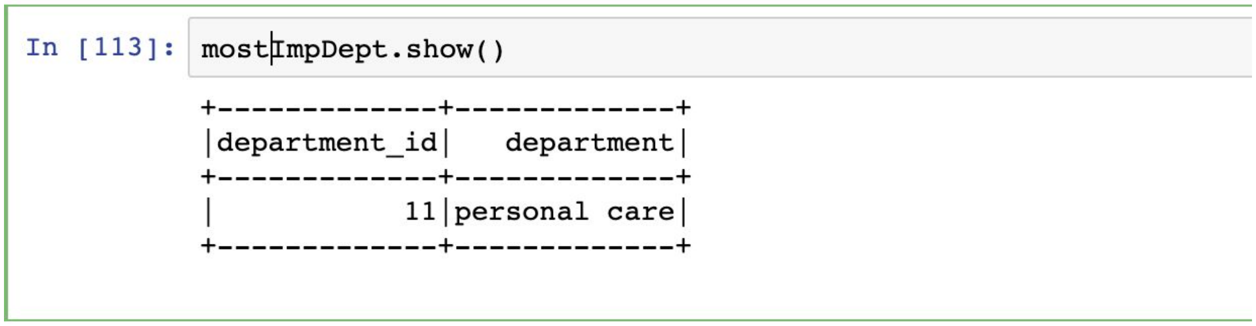
```
In [92]: mostImpAisle.show()

+--------+-------+
|aisle_id|  aisle|
+--------+-------+
|     100|missing|
+--------+-------+
```

#Looking for next most important aisle as 100 is missing from aisle.csv
nextTotProdPerAisleDF = totProdPerAisle.where(totProdPerAisle['aisle_id']!=aisle_id)
nextval = nextTotProdPerAisleDF.agg({'count':'max'}).collect()[0][0]
nextTotProdPerAisleDF.createOrReplaceTempView("NextAisleCountTab")
aisle_id = spark.sql("Select * from NextAisleCountTab where count =
{0}".format(nextval)).collect()[0][0]
nextMostImpAisle = aisleDF.where(aisleDF['aisle_id']==aisle_id)
nextMostImpAisle.show()



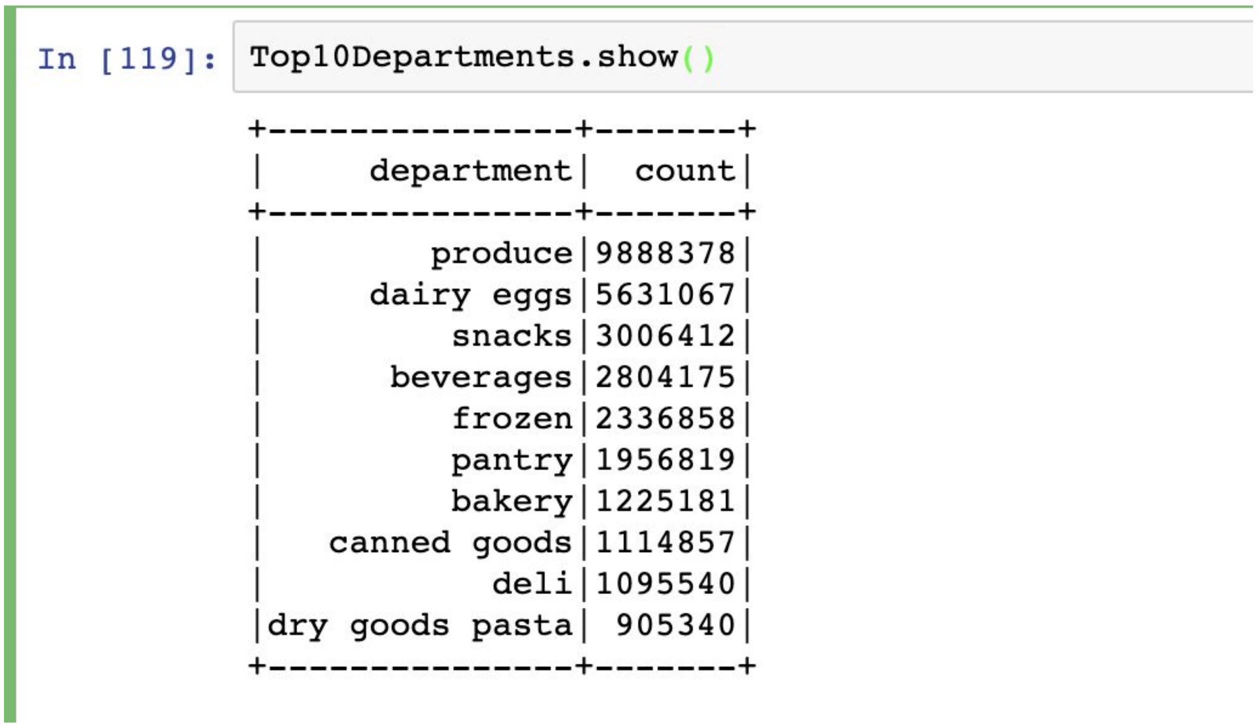
```
In [104]: nextMostImpAisle.show()    #Most important Aisle

+--------+---------------+
|aisle_id|          aisle|
+--------+---------------+
|      45|candy chocolate|
+--------+---------------+
```

Most important Department:
totProdPerDeptDF = prodDF.groupby('department_id').count()
maxCount = totProdPerDeptDF.agg({'count':'max'}).collect()[0][0]
totProdPerDeptDF.createOrReplaceTempView("DeptCountTab")
dept_id = spark.sql("Select * from DeptCountTab where count =
{0}".format(maxCount)).collect()[0][0]
mostImpDept = deptDF.where(deptDF['department_id']==dept_id)
mostImpDept.show()



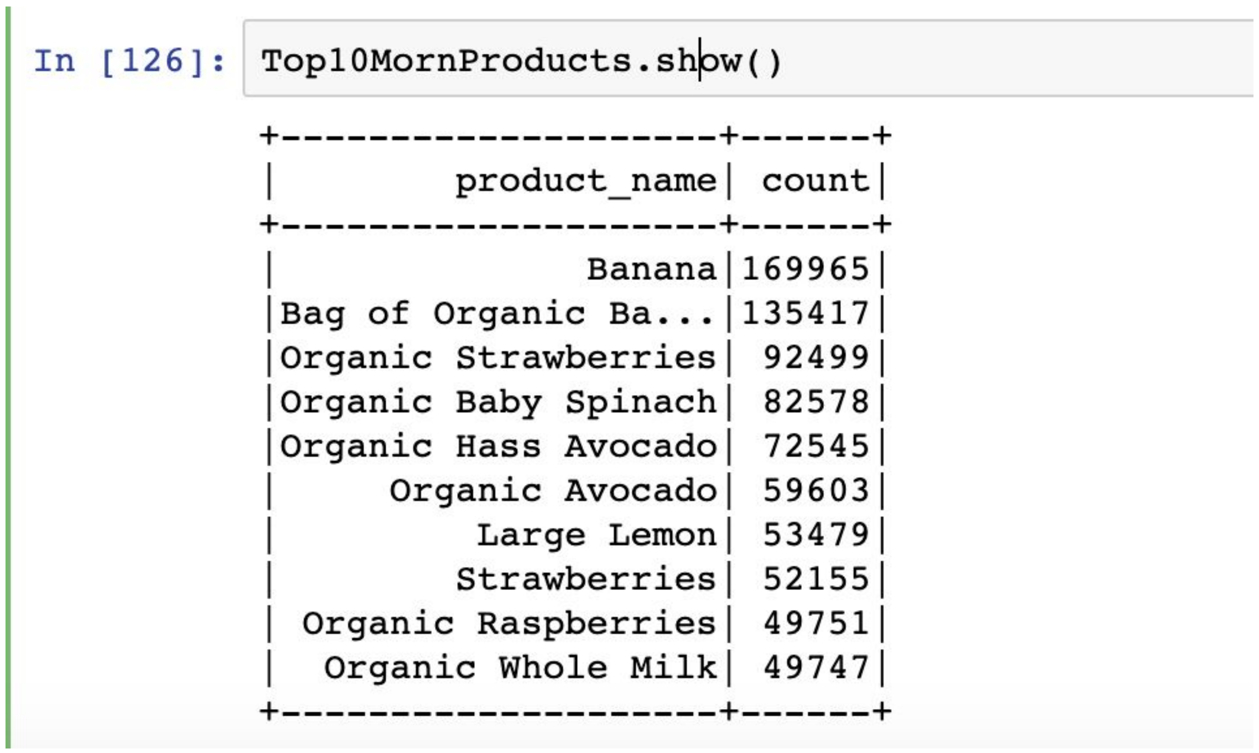
```
In [113]: mostImpDept.show()

+-------------+-------------+
|department_id|   department|
+-------------+-------------+
|           11|personal care|
+-------------+-------------+
```

8. **Get the Top 10 departments**
   totProdOrderedPerDeptDF = OrderProductAisleDepartAllDF.groupby('department').count()
   Top10Departments = totProdOrderedPerDeptDF.orderBy(col('count').desc()).limit(10)
   Top10Departments.show()



```
In [119]: Top10Departments.show()

+---------------+-------+
|     department|  count|
+---------------+-------+
|        produce|9888378|
|    dairy eggs|5631067|
|         snacks|3006412|
|      beverages|2804175|
|         frozen|2336858|
|         pantry|1956819|
|         bakery|1225181|
|   canned goods|1114857|
|           deli|1095540|
|dry goods pasta| 905340|
+---------------+-------+
```

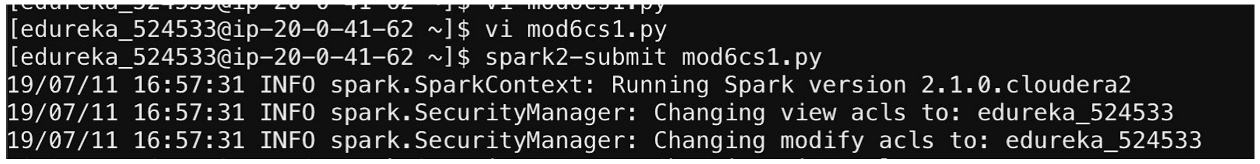
9. **List top 10 products ordered in the morning (6 AM to 11 AM)**
   morningOrdersDF=OrderProductAisleDepartAllDF.where((OrderProductAisleDepartAllDF['order_hour_of_day']>=6)&(OrderProductAisleDepartAllDF['order_hour_of_day']<=11))
   totProdOrderedMornDF = morningOrdersDF.groupby('product_name').count()
   Top10MornProducts = totProdOrderedMornDF.orderBy(col('count').desc()).limit(10)
   Top10MornProducts.show()



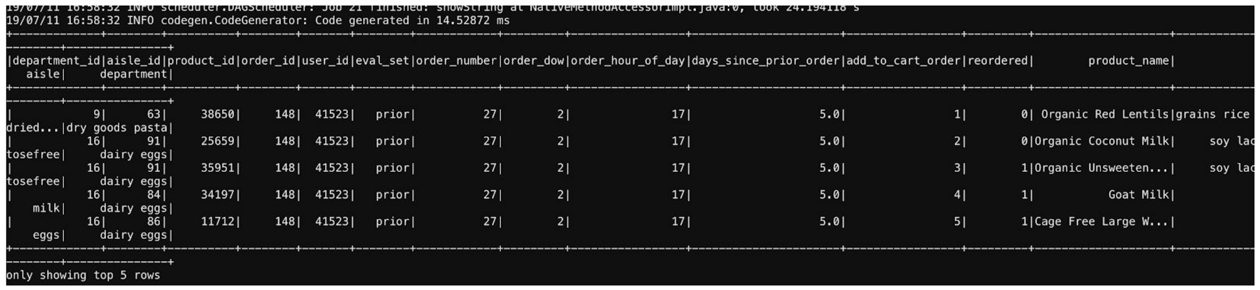
```
In [126]: Top10MornProducts.show()

+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|169965|
|Bag of Organic Ba...|135417|
|Organic Strawberries| 92499|
|Organic Baby Spinach| 82578|
|Organic Hass Avocado| 72545|
|     Organic Avocado| 59603|
|         Large Lemon| 53479|
|        Strawberries| 52155|
| Organic Raspberries| 49751|
|  Organic Whole Milk| 49747|
+--------------------+------+
```

10. **Create a spark - submit application for the same and print the findings in the log**
    Module: mod6cs1.py

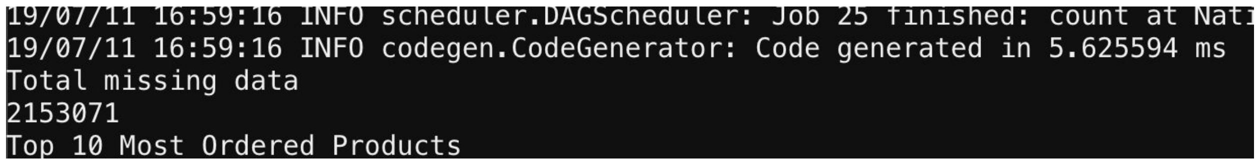
Screen Shots:



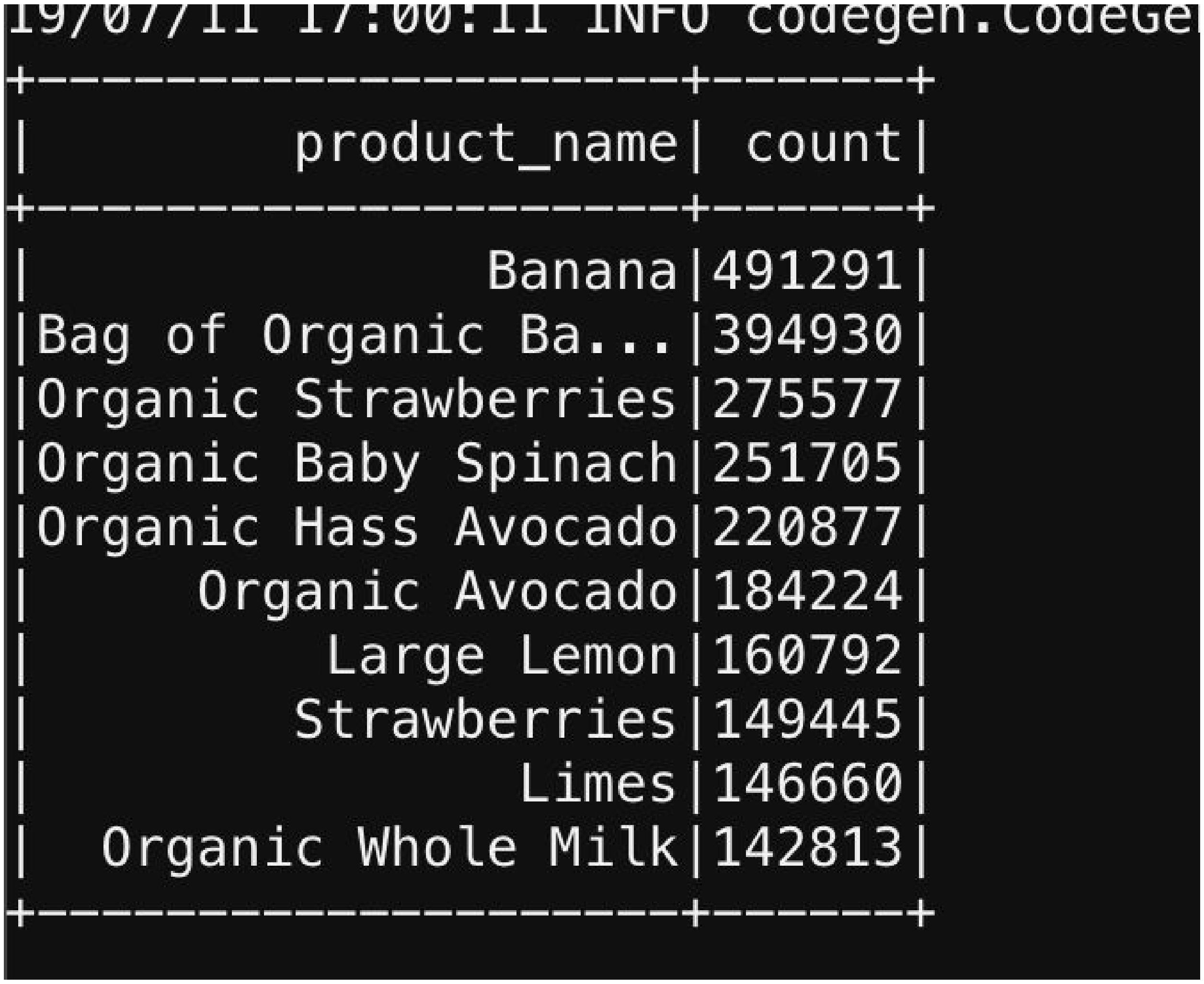
```
[edureka_524533@ip-20-0-41-62 ~]$ vi mod6cs1.py
[edureka_524533@ip-20-0-41-62 ~]$ vi mod6cs1.py
[edureka_524533@ip-20-0-41-62 ~]$ spark2-submit mod6cs1.py
19/07/11 16:57:31 INFO spark.SparkContext: Running Spark version 2.1.0.cloudera2
19/07/11 16:57:31 INFO spark.SecurityManager: Changing view acls to: edureka_524533
19/07/11 16:57:31 INFO spark.SecurityManager: Changing modify acls to: edureka_524533
```



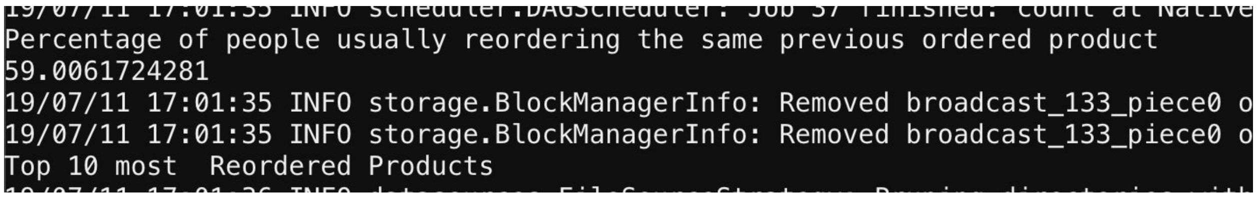
```
19/07/11 16:58:32 INFO scheduler.DAGScheduler: Job 21 finished: showString at NativeMethodAccessorImpl.java:0, took 24.194118 s
19/07/11 16:58:32 INFO codegen.CodeGenerator: Code generated in 14.52872 ms
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+----------------------+-----------------+---------+--------------------+----------
--------+---------------+
|department_id|aisle_id|product_id|order_id|user_id|eval_set|order_number|order_dow|order_hour_of_day|days_since_prior_order|add_to_cart_order|reordered|        product_name|
   aisle|     department|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+----------------------+-----------------+---------+--------------------+----------
--------+---------------+
|            9|      63|     38650|     148|  41523|   prior|          27|        2|               17|                   5.0|                1|        0| Organic Red Lentils|grains rice
dried...|dry goods pasta|
|           16|      91|     25659|     148|  41523|   prior|          27|        2|               17|                   5.0|                2|        0|Organic Coconut Milk|     soy lac
tosefree|     dairy eggs|
|           16|      91|     35951|     148|  41523|   prior|          27|        2|               17|                   5.0|                3|        1|Organic Unsweeten...|     soy lac
tosefree|     dairy eggs|
|           16|      84|     34197|     148|  41523|   prior|          27|        2|               17|                   5.0|                4|        1|           Goat Milk|
    milk|     dairy eggs|
|           16|      86|     11712|     148|  41523|   prior|          27|        2|               17|                   5.0|                5|        1|Cage Free Large W...|
    eggs|     dairy eggs|
+-------------+--------+----------+--------+-------+--------+------------+---------+-----------------+----------------------+-----------------+---------+--------------------+----------
--------+---------------+
only showing top 5 rows
```



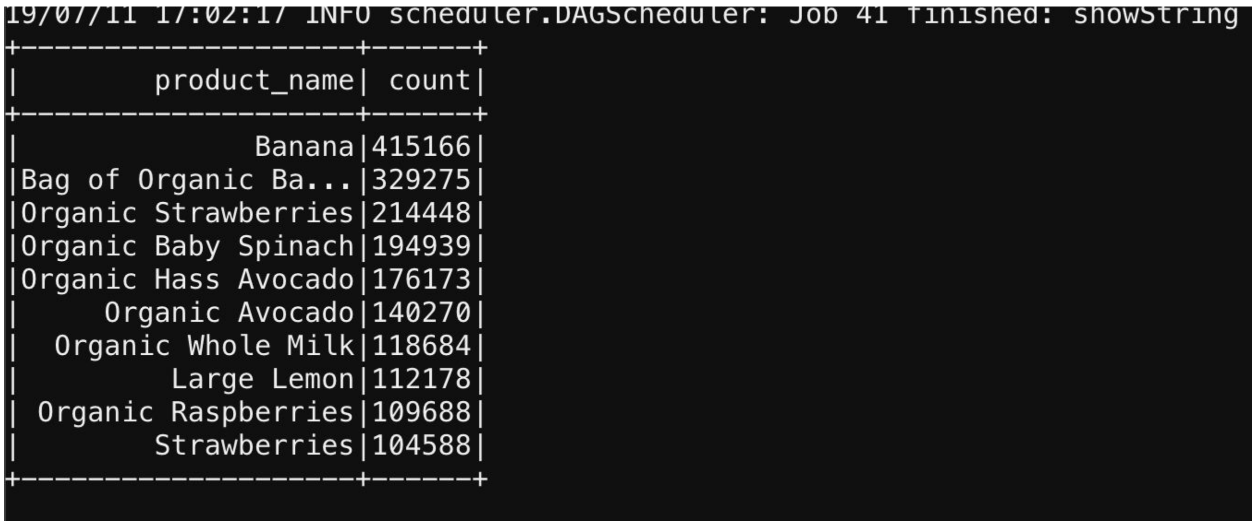
```
19/07/11 16:59:16 INFO scheduler.DAGScheduler: Job 25 finished: count at Nati
19/07/11 16:59:16 INFO codegen.CodeGenerator: Code generated in 5.625594 ms
Total missing data
2153071
Top 10 Most Ordered Products
```



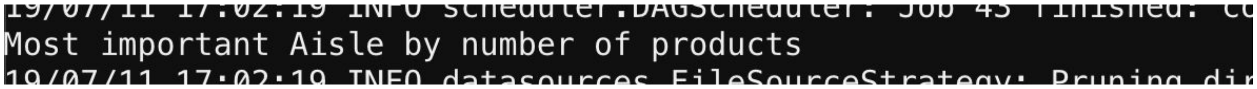
```
19/07/11 17:00:11 INFO codegen.CodeGe
+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|491291|
|Bag of Organic Ba...|394930|
|Organic Strawberries|275577|
|Organic Baby Spinach|251705|
|Organic Hass Avocado|220877|
|     Organic Avocado|184224|
|         Large Lemon|160792|
|        Strawberries|149445|
|               Limes|146660|
|  Organic Whole Milk|142813|
+--------------------+------+
```



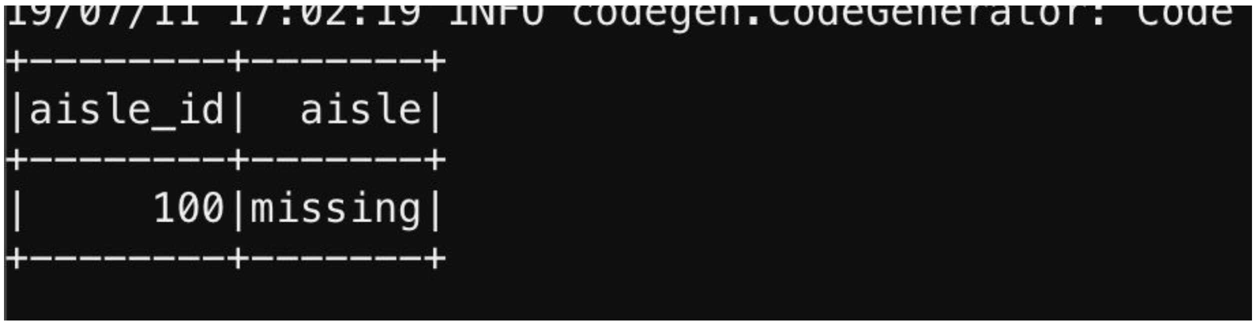
```
19/07/11 17:01:35 INFO scheduler.DAGScheduler: Job 37 finished: count at Native
Percentage of people usually reordering the same previous ordered product
59.0061724281
19/07/11 17:01:35 INFO storage.BlockManagerInfo: Removed broadcast_133_piece0 o
19/07/11 17:01:35 INFO storage.BlockManagerInfo: Removed broadcast_133_piece0 o
Top 10 most  Reordered Products
19/07/11 17:01:36 INFO datasources.FileSourceStrategy: Pruning directories with
```



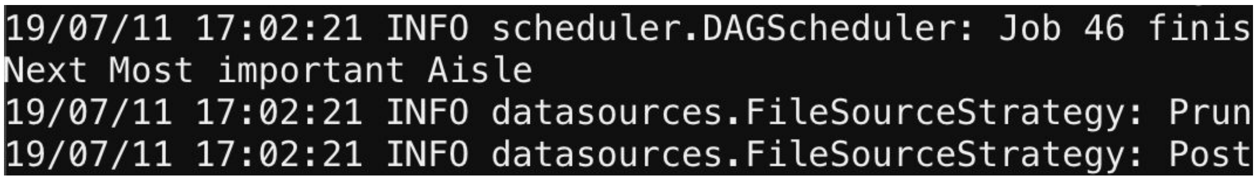
```
19/07/11 17:02:17 INFO scheduler.DAGScheduler: Job 41 finished: showString
+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|415166|
|Bag of Organic Ba...|329275|
|Organic Strawberries|214448|
|Organic Baby Spinach|194939|
|Organic Hass Avocado|176173|
|     Organic Avocado|140270|
|  Organic Whole Milk|118684|
|         Large Lemon|112178|
| Organic Raspberries|109688|
|        Strawberries|104588|
+--------------------+------+
```



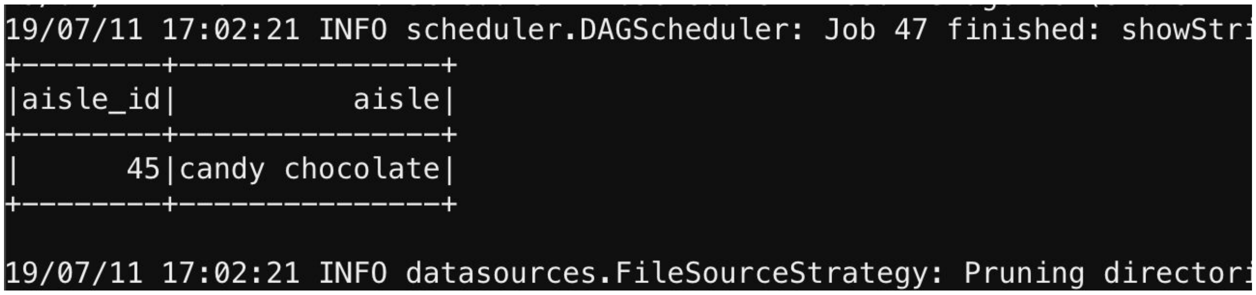
```
19/07/11 17:02:19 INFO scheduler.DAGScheduler: Job 43 finished: co
Most important Aisle by number of products
19/07/11 17:02:19 INFO datasources.FileSourceStrategy: Pruning dir
```



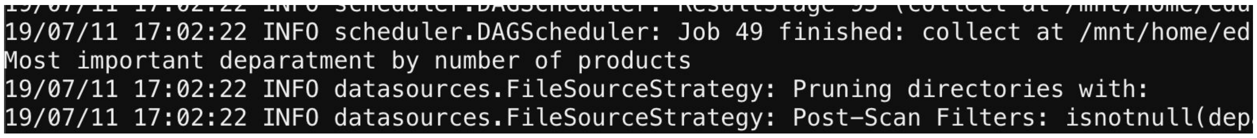
```
19/07/11 17:02:19 INFO codegen.CodeGenerator: Code
+--------+-------+
|aisle_id|  aisle|
+--------+-------+
|     100|missing|
+--------+-------+
```



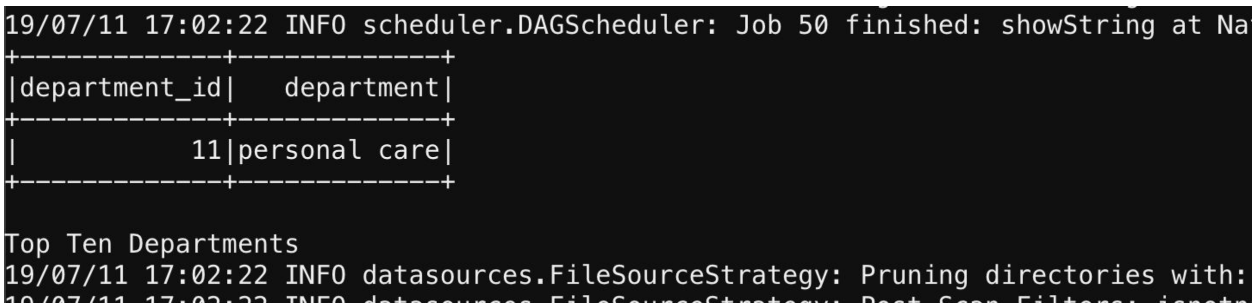
```
19/07/11 17:02:21 INFO scheduler.DAGScheduler: Job 46 finis
Next Most important Aisle
19/07/11 17:02:21 INFO datasources.FileSourceStrategy: Prun
19/07/11 17:02:21 INFO datasources.FileSourceStrategy: Post
```



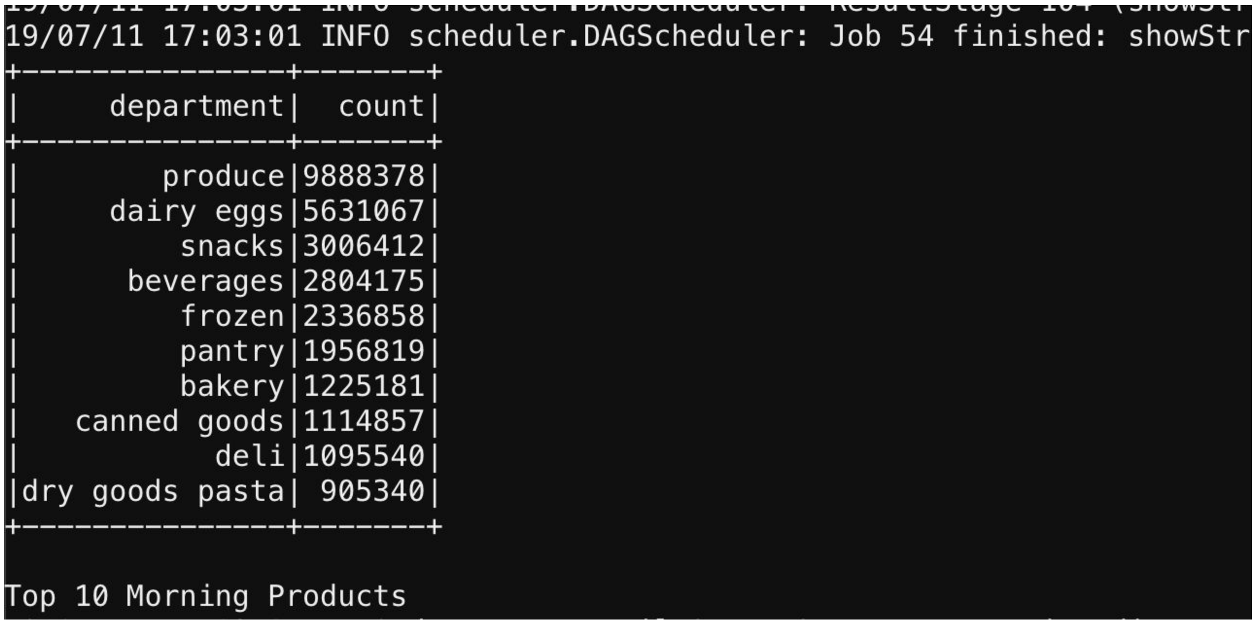
```
19/07/11 17:02:21 INFO scheduler.DAGScheduler: Job 47 finished: showStri
+--------+---------------+
|aisle_id|          aisle|
+--------+---------------+
|      45|candy chocolate|
+--------+---------------+

19/07/11 17:02:21 INFO datasources.FileSourceStrategy: Pruning directori
```



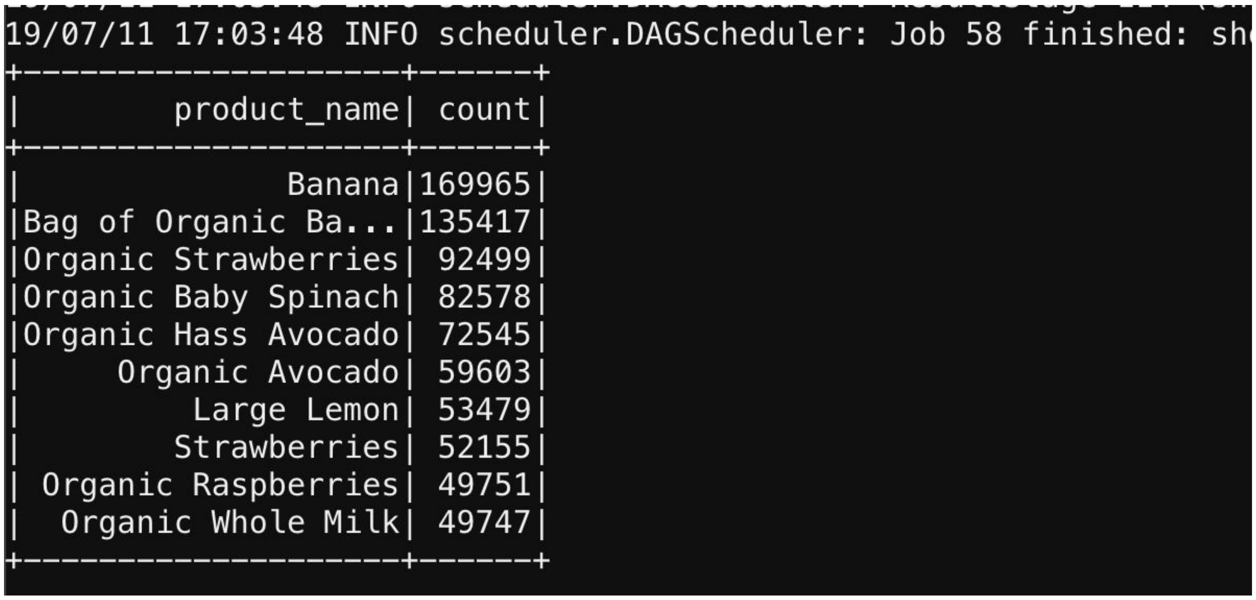
```
19/07/11 17:02:22 INFO scheduler.DAGScheduler: ResultStage 95 (collect at /mnt/home/edu
19/07/11 17:02:22 INFO scheduler.DAGScheduler: Job 49 finished: collect at /mnt/home/ed
Most important deparatment by number of products
19/07/11 17:02:22 INFO datasources.FileSourceStrategy: Pruning directories with:
19/07/11 17:02:22 INFO datasources.FileSourceStrategy: Post-Scan Filters: isnotnull(dep
```



```
19/07/11 17:02:22 INFO scheduler.DAGScheduler: Job 50 finished: showString at Na
+-------------+-------------+
|department_id|   department|
+-------------+-------------+
|           11|personal care|
+-------------+-------------+

Top Ten Departments
19/07/11 17:02:22 INFO datasources.FileSourceStrategy: Pruning directories with:
```



```
19/07/11 17:03:01 INFO scheduler.DAGScheduler: Job 54 finished: showStr
+---------------+-------+
|     department|  count|
+---------------+-------+
|        produce|9888378|
|     dairy eggs|5631067|
|         snacks|3006412|
|      beverages|2804175|
|         frozen|2336858|
|         pantry|1956819|
|         bakery|1225181|
|   canned goods|1114857|
|           deli|1095540|
|dry goods pasta| 905340|
+---------------+-------+

Top 10 Morning Products
```



```
19/07/11 17:03:48 INFO scheduler.DAGScheduler: Job 58 finished: sh
+--------------------+------+
|        product_name| count|
+--------------------+------+
|              Banana|169965|
|Bag of Organic Ba...|135417|
|Organic Strawberries| 92499|
|Organic Baby Spinach| 82578|
|Organic Hass Avocado| 72545|
|     Organic Avocado| 59603|
|         Large Lemon| 53479|
|        Strawberries| 52155|
| Organic Raspberries| 49751|
|  Organic Whole Milk| 49747|
+--------------------+------+
```

Spark 2.1.0.cloudera2

History Server

Event log directory: hdfs://nameservice1/user/spark/applicationHistory

Last updated: 11/07/2019, 22:49:04

Show 20 entries

Search:

App ID	App Name	Started	Completed	Duration	Spark User	Last Updated	Event Log
application_1528714825862_135445	Module 6 Case Study 1	2019-07-11 16:08:46	2019-07-11 17:10:41	1.0 h	edureka_524533	2019-07-11 17:10:41	Download