hp

[Email address]

Abstract

**In this project, we will focus only on sales data, and we need to create a data mart around  
product sales in stores.  
A data mart around sales will help the company generate actionable insights such as the following:  
• Total sales in a day/week/quarter (both in terms of sales revenue and units sold)  
• Top 5 most selling item categories in a day (both in terms of amount and quantity)  
• Top 5 most profitable items in a day/week/quarter  
• Top 5 most profitable stores in a day/week/month/quarter  
• Total profit or loss in U.S. dollars every day in percentage**

IndustrY GRADE Project

Forecasting Trends and Demand

**The Goal of the Project Below are some of the high-level goals of this project:**

• Get an overall understanding of the retail domain

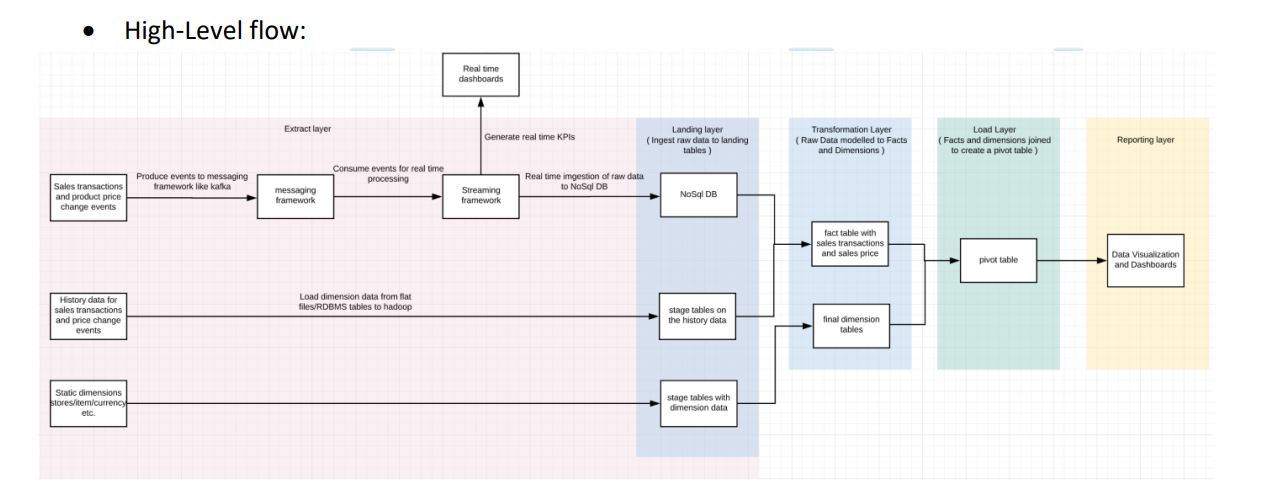
• Learn the fundamentals & standards of ETL and data warehousing

• Real-time and batch ingestion of data from multiple sources to Big Data storage like Hive/HBase/HDFS using Kafka and Spark

• Data cleansing/wrangling/transformation using Hive and Spark • Lambda architecture where data can be processed in both batch and real-time

• Reporting KPIs using dashboards

**High -Level Flow**



Forecasting Trends and Demand

[edureka\_918210@ip-20-0-41-62 ~]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart

Found 4 items

drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata

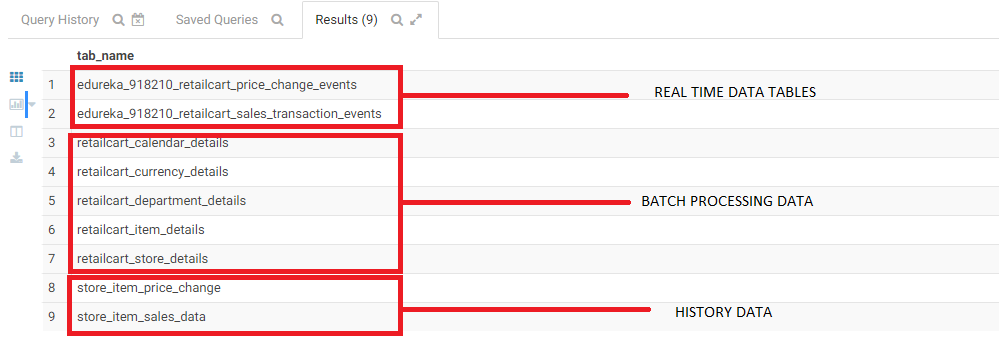
drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-26 16:23 /bigdatapgp/common\_folder/project\_retailcart/connector\_jars

drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-26 16:19 /bigdatapgp/common\_folder/project\_retailcart/history\_data

drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-27 19:14 /bigdatapgp/common\_folder/project\_retailcart/realtimedata

[edureka\_918210@ip-20-0-41-62 ~]$

FINAL TABLES DETAILS

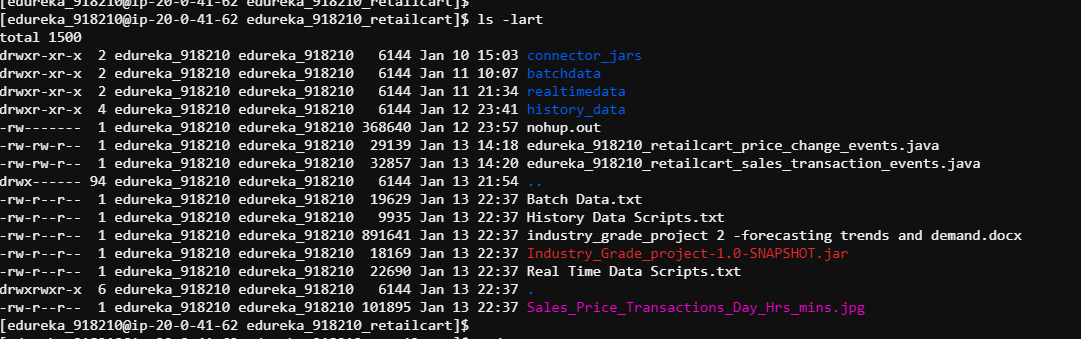


Please find above the tables , generated in 3 different Process Workflows:

1. Batch Data
2. History Data
3. Real Time Data

**Location Of the Scripts/Jar Details**

/mnt/home/edureka\_918210/edureka\_918210\_retailcart



[edureka\_918210@ip-20-0-41-62 project\_retailcart]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart/batchdata

Found 5 items

-rw-r--r-- 3 evaluationuser01 supergroup 1822863 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_calendar\_details.txt

-rw-r--r-- 3 evaluationuser01 supergroup 1356 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_currency\_details.txt

-rw-r--r-- 3 evaluationuser01 supergroup 271441 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_department\_details.txt

-rw-r--r-- 3 evaluationuser01 supergroup 107275790 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_item\_details.txt

-rw-r--r-- 3 evaluationuser01 supergroup 831617 2020-07-26 16:12 /bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_store\_details.txt

[edureka\_918210@ip-20-0-41-62 project\_retailcart]$

[edureka\_918210@ip-20-0-41-62 project\_retailcart]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart/connector\_jars

Found 1 items

-rw-r--r-- 3 evaluationuser01 supergroup 1006959 2020-07-26 16:23 /bigdatapgp/common\_folder/project\_retailcart/connector\_jars/mysql-connector-java-5.1.48-bin.

jar

[edureka\_918210@ip-20-0-41-62 project\_retailcart]$

[edureka\_918210@ip-20-0-41-62 project\_retailcart]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart/history\_data

Found 2 items

drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-26 16:18 /bigdatapgp/common\_folder/project\_retailcart/history\_data/store\_item\_price\_change

drwxr-xr-x - evaluationuser01 supergroup 0 2020-07-26 16:20 /bigdatapgp/common\_folder/project\_retailcart/history\_data/store\_item\_sales\_data

[edureka\_918210@ip-20-0-41-62 project\_retailcart]$

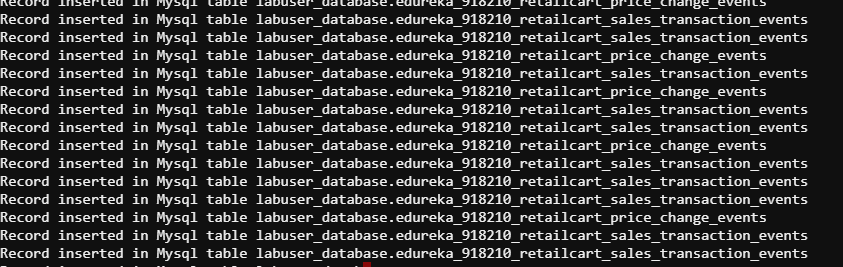
[edureka\_918210@ip-20-0-41-62 project\_retailcart]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart/realtimedata

Found 2 items

-rw-r--r-- 3 evaluationuser01 supergroup 41546 2020-07-27 09:56 /bigdatapgp/common\_folder/project\_retailcart/realtimedata/price\_change\_event.txt

-rw-r--r-- 3 evaluationuser01 supergroup 5029 2020-07-27 16:42 /bigdatapgp/common\_folder/project\_retailcart/realtimedata/real\_time\_simulator.py

**RECORD INSERTED**



[edureka\_918210@ip-20-0-41-62 project\_retailcart]$

/bigdatapgp/common\_folder/project\_retailcart/batchdata/retailcart\_calendar\_details.txt

"calendar\_date","date\_desc","week\_day\_nbr","week\_number","week\_name","year\_week\_number","month\_number","month\_name","quarter\_number","quarter\_name","half\_year\_number","half\_year\_name","geo\_region\_cd"

[edureka\_918210@ip-20-0-41-62 batchdata]$ hadoop fs -ls /user/edureka\_918210/project\_retailcart/batchdata

Found 1 items

-rw-r--r-- 3 edureka\_918210 hadoop 1822689 2021-01-10 15:56 /user/edureka\_918210/project\_retailcart/batchdata/retailcart\_calendar\_details.txt

[edureka\_918210@ip-20-0-41-62 batchdata]$

2011-02-20 Sunday, February 20, 2011 2 4 Week 04 201104 1 February 1 Q1 1 1st Half US

import org.apache.spark.ml.classification.LogisticRegression

import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator

import org.apache.spark.ml.feature.{StringIndexer, VectorAssembler}

import org.apache.spark.ml.linalg.DenseVector

import org.apache.spark.mllib.evaluation.BinaryClassificationMetrics

import org.apache.spark.sql.SparkSession

import org.apache.log4j.Logger

import org.apache.log4j.Level

import org.apache.spark.ml.feature.{HashingTF, IDF, Tokenizer}

import org.apache.spark.ml.feature.QuantileDiscretizer

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

import org.apache.spark.ml.classification.LogisticRegression

import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator

import org.apache.spark.ml.feature.{StringIndexer, VectorAssembler}

import org.apache.spark.ml.feature.{HashingTF, IDF, Tokenizer}

import org.apache.spark.ml.linalg.DenseVector

import org.apache.spark.mllib.evaluation.BinaryClassificationMetrics

import org.apache.spark.sql.SparkSession

import org.apache.log4j.Logger

import org.apache.log4j.Level

import org.apache.spark.ml.feature.{HashingTF, IDF, Tokenizer}

import org.apache.spark.ml.feature.QuantileDiscretizer

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

import org.apache.spark.ml.feature.MinMaxScaler

import org.apache.spark.ml.linalg.Vectors

import org.apache.spark.ml.classification.RandomForestClassifier

import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

import org.apache.spark.ml.feature.StringIndexer

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.ml.classification.RandomForestClassifier

import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

import org.apache.spark.ml.feature.StringIndexer

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.sql.functions.\_

import org.apache.spark.ml.feature.{OneHotEncoder, StringIndexer}

import org.apache.spark.ml.Pipeline

import org.apache.spark.ml.classification.{RandomForestClassificationModel, RandomForestClassifier}

import org.apache.spark.ml.evaluation.MulticlassClassificationEvaluator

import org.apache.spark.ml.feature.{IndexToString, StringIndexer, VectorIndexer}

import org.apache.spark.ml.evaluation.RegressionEvaluator

import org.apache.spark.ml.regression.LinearRegression

import org.apache.spark.ml.tuning.{ParamGridBuilder,TrainValidationSplit}

import org.apache.log4j.\_

import org.apache.spark.ml.feature.VectorAssembler

import org.apache.spark.mllib.linalg.Vectors

import org.apache.spark.ml.Pipeline

import org.apache.spark.ml.classification.LogisticRegression

import org.apache.spark.ml.evaluation.BinaryClassificationEvaluator

import org.apache.spark.ml.feature.{HashingTF, Tokenizer}

import org.apache.spark.ml.linalg.Vector

import org.apache.spark.ml.tuning.{CrossValidator, ParamGridBuilder}

import org.apache.spark.sql.Row

// $example off$

import org.apache.spark.sql.SparkSession

import org.apache.spark.sql.{Row, SparkSession}

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

import org.apache.spark.sql.functions.\_

import org.apache.spark.sql.Row;

import org.apache.spark.sql.types.{StructType, StructField, StringType};

**retailcart\_calendar\_details**

"calendar\_date date\_desc week\_day\_nbr week\_number week\_name year\_week\_number month\_number month\_name quarter\_number quarter\_name half\_year\_number half\_year\_name geo\_region\_cd"

spark.read.textFile("/user/edureka\_918210/project\_retailcart/batchdata/retailcart\_calendar\_details.txt").createOrReplaceTempView("retailcart\_calendar\_details");

2011-02-20 Sunday, February 20, 2011 2 4 Week 04 201104 1 February 1 Q1 1 1st Half US

val retailcart\_calendar =

spark.sql(""" Select

split(value,'\t')[0] as calendar\_date,

split(value,'\t')[1] as date\_desc,

split(value,'\t')[2] as week\_day\_nbr,

split(value,'\t')[3] as week\_number,

split(value,'\t')[4] as week\_name,

split(value,'\t')[5] as year\_week\_number,

split(value,'\t')[6] as month\_number,

split(value,'\t')[7] as month\_name,

split(value,'\t')[8] as quarter\_number,

split(value,'\t')[9] as quarter\_name,

split(value,'\t')[10] as half\_year\_number,

split(value,'\t')[11] as half\_year\_name,

split(value,'\t')[12] as geo\_region\_cd

from retailcart\_calendar\_details """)

val retailcart\_calendar2 = retailcart\_calendar.selectExpr("calendar\_date", "date\_desc", "cast(week\_day\_nbr as integer) week\_day\_nbr", "week\_number", "week\_name", "cast(year\_week\_number as integer) year\_week\_number","month\_number","month\_name","quarter\_number","quarter\_name","half\_year\_number","half\_year\_name","geo\_region\_cd")

**retailcart\_department\_details**

department\_number department\_category\_number department\_sub\_catg\_number department\_description department\_category\_description department\_sub\_catg\_desc geo\_region\_cd

spark.read.textFile("/user/edureka\_918210/project\_retailcart/batchdata/retailcart\_department\_details.txt").createOrReplaceTempView("retailcart\_department\_details");

val retailcart\_department =

spark.sql(""" Select

split(value,'\t')[0] as department\_number,

split(value,'\t')[1] as department\_category\_number,

split(value,'\t')[2] as department\_sub\_catg\_number,

split(value,'\t')[3] as department\_description,

split(value,'\t')[4] as department\_category\_description,

split(value,'\t')[5] as department\_sub\_catg\_desc,

split(value,'\t')[6] as geo\_region\_cd

from retailcart\_department\_details """)

val retailcart\_department2 = retailcart\_department.selectExpr("cast(department\_number as integer) department\_number", "cast(department\_category\_number as integer) department\_category\_number", "cast(department\_sub\_catg\_number as integer) department\_sub\_catg\_number", "department\_description", "department\_category\_description", "department\_sub\_catg\_desc","geo\_region\_cd")

**retailcart\_item\_details**

item\_id geo\_region\_cd item\_description unique\_product\_cd unique\_product\_cd\_desc department\_number department\_category\_number department\_sub\_catg\_number vendor\_name vendor\_number item\_status\_cd item\_status\_desc unit\_cost

spark.read.textFile("/user/edureka\_918210/project\_retailcart/batchdata/retailcart\_item\_details.txt").createOrReplaceTempView("retailcart\_item\_details");

val retailcart\_item =

spark.sql(""" Select

split(value,'\t')[0] as item\_id,

split(value,'\t')[1] as geo\_region\_cd,

split(value,'\t')[2] as item\_description,

split(value,'\t')[3] as unique\_product\_cd,

split(value,'\t')[4] as unique\_product\_cd\_desc,

split(value,'\t')[5] as department\_number,

split(value,'\t')[6] as department\_category\_number,

split(value,'\t')[7] as department\_sub\_catg\_number,

split(value,'\t')[8] as vendor\_name,

split(value,'\t')[9] as vendor\_number,

split(value,'\t')[10] as item\_status\_cd,

split(value,'\t')[11] as item\_status\_desc,

split(value,'\t')[12] as unit\_cost

from retailcart\_item\_details """)

182396492 US PR SV SH COC 7940045968 SUAVE SHAM dept-num:2 dept-catg-num:561 dept-sub-catg-num:1203 PUERTO RICO SUPPLIES GROUP

INC 77641 A ACTIVE 41.68

74277172 US REV CS POWDER-880 30997630008 REV POWDER dept-num:46 dept-catg-num:7112 dept-sub-catg-num:16006 REVLON INC 430

306 A ACTIVE NULL

val retailcart\_item2 = retailcart\_item.selectExpr("cast(item\_id as integer) item\_id", "geo\_region\_cd", "item\_description", "unique\_product\_cd", "unique\_product\_cd\_desc", "department\_number","department\_category\_number","department\_sub\_catg\_number","vendor\_name","vendor\_number","item\_status\_cd","item\_status\_desc","cast(unit\_cost as double) unit\_cost")

**retailcart\_store\_details**

store\_id geo\_region\_cd store\_name sub\_division\_name sub\_division\_number region\_number region\_name market\_number market\_name city\_name open\_date open\_status\_desc postal\_cd state\_prov\_cd

spark.read.textFile("/user/edureka\_918210/project\_retailcart/batchdata/retailcart\_store\_details.txt").createOrReplaceTempView("retailcart\_store\_details");

val retailcart\_store =

spark.sql(""" Select

split(value,'\t')[0] as store\_id,

split(value,'\t')[1] as geo\_region\_cd,

split(value,'\t')[2] as store\_name,

split(value,'\t')[3] as sub\_division\_name,

split(value,'\t')[4] as sub\_division\_number,

split(value,'\t')[5] as region\_number,

split(value,'\t')[6] as region\_name,

split(value,'\t')[7] as market\_number,

split(value,'\t')[8] as market\_name,

split(value,'\t')[9] as city\_name,

split(value,'\t')[10] as open\_date,

split(value,'\t')[11] as open\_status\_desc,

split(value,'\t')[12] as postal\_cd,

split(value,'\t')[13] as state\_prov\_cd

from retailcart\_store\_details """)

val retailcart\_store2 = retailcart\_store.selectExpr("cast(store\_id as integer) store\_id", "geo\_region\_cd", "store\_name", "sub\_division\_name", "sub\_division\_number", "region\_number","region\_name","market\_number","market\_name","city\_name","open\_date","open\_status\_desc","postal\_cd","state\_prov\_cd")

**retailcart\_currency\_details**

currency\_id currency\_code currency\_name usd\_exchange\_rate

spark.read.textFile("/user/edureka\_918210/project\_retailcart/batchdata/retailcart\_currency\_details.txt").createOrReplaceTempView("retailcart\_currency\_details");

val retailcart\_currency =

spark.sql(""" Select

split(value,'\t')[0] as currency\_id,

split(value,'\t')[1] as currency\_code,

split(value,'\t')[2] as currency\_name,

split(value,'\t')[3] as usd\_exchange\_rate

from retailcart\_currency\_details """)

val retailcart\_currency2 = retailcart\_currency.selectExpr("currency\_id", "currency\_code", "currency\_name", "cast(usd\_exchange\_rate as double) usd\_exchange\_rate")

**TO SAVE THE SPARK DATAFRAME INTO MYSQL : the mysql connector jar file is downloaded and connected.**

spark2-shell --jars /mnt/home/edureka\_918210/project\_retailcart/connector\_jars/mysql-connector-java-5.1.48-bin.jar

retailcart\_calendar2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","retailcart\_calendar2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

retailcart\_department2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","retailcart\_department2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

retailcart\_item2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","retailcart\_item2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

retailcart\_store2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","retailcart\_store2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

retailcart\_currency2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","retailcart\_currency2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

mysql -u edu\_labuser -h dbserver.edu.cloudlab.com -p

password - edureka

CREATE TABLE retailcart\_calendar\_details SELECT \* FROM retailcart\_calendar2;

CREATE TABLE retailcart\_department\_details SELECT \* FROM retailcart\_department2;

CREATE TABLE retailcart\_item\_details SELECT \* FROM retailcart\_item2;

CREATE TABLE retailcart\_store\_details SELECT \* FROM retailcart\_store2;

CREATE TABLE retailcart\_currency\_details SELECT \* FROM retailcart\_currency2;

**MYSQL TO HIVE**

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table retailcart\_calendar\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects --split-by calendar\_date --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table retailcart\_department\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects --split-by department\_number --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table retailcart\_item\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects --split-by item\_id --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table retailcart\_store\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects --split-by store\_id --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table retailcart\_currency\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects --split-by currency\_id --password edureka

**HISTORY DATA**

**4.2 History data for sales transactions and price change events:**

**Store\_Item\_Price\_Change**

val spark = SparkSession.builder().appName("Industry\_Grade\_Project-spark").master("local[\*]").enableHiveSupport().getOrCreate();

val store\_item\_price\_change = spark.read.orc("/bigdatapgp/common\_folder/project\_retailcart/history\_data/store\_item\_price\_change")

scala> store\_item\_price\_change.printSchema

root

|-- \_col0: integer (nullable = true)

|-- \_col1: integer (nullable = true)

|-- \_col2: timestamp (nullable = true)

|-- \_col3: string (nullable = true)

|-- \_col4: string (nullable = true)

|-- \_col5: date (nullable = true)

|-- \_col6: decimal(15,2) (nullable = true)

|-- \_col7: decimal(15,2) (nullable = true)

val store\_item\_price\_change2 = store\_item\_price\_change.toDF("item\_id","store\_id","price\_chng\_activation\_ts","geo\_region\_cd","price\_change\_reason","business\_date","prev\_price\_amt","curr\_price\_amt")

scala> store\_item\_price\_change2.printSchema

root

|-- item\_id: integer (nullable = true)

|-- store\_id: integer (nullable = true)

|-- price\_chng\_activation\_ts: timestamp (nullable = true)

|-- geo\_region\_cd: string (nullable = true)

|-- price\_change\_reason: string (nullable = true)

|-- business\_date: date (nullable = true)

|-- prev\_price\_amt: decimal(15,2) (nullable = true)

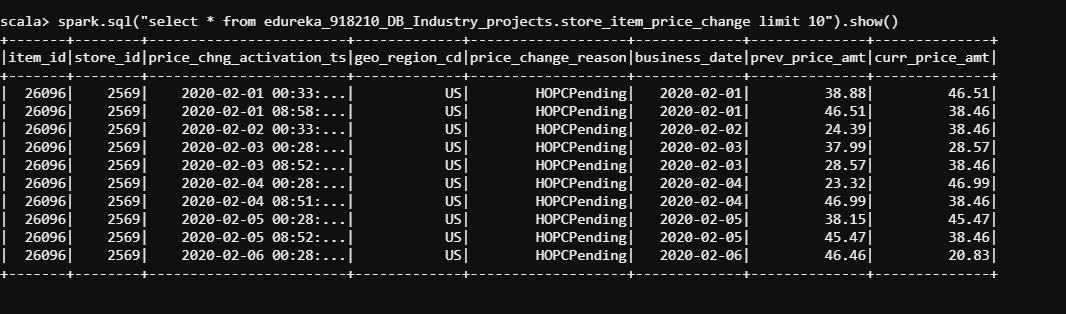
|-- curr\_price\_amt: decimal(15,2) (nullable = true)

**SPARK TO HIVE**

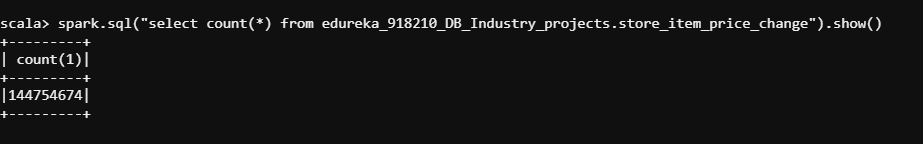
**Store\_item\_price\_change**

store\_item\_price\_change2.write.mode("overwrite").saveAsTable("edureka\_918210\_DB\_Industry\_projects.store\_item\_price\_change")

scala> spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.store\_item\_price\_change limit 10").show()



scala> spark.sql("select count(\*) from edureka\_918210\_DB\_Industry\_projects.store\_item\_price\_change").show()



**Store\_item\_sales\_data**

val spark = SparkSession.builder().appName("Industry\_Grade\_Project-spark").master("local[\*]").enableHiveSupport().getOrCreate();

val store\_item\_sales\_data = spark.read.orc("/bigdatapgp/common\_folder/project\_retailcart/history\_data/store\_item\_sales\_data")

scala> store\_item\_sales\_data.printSchema

root

|-- \_col0: integer (nullable = true)

|-- \_col1: date (nullable = true)

|-- \_col2: integer (nullable = true)

|-- \_col3: integer (nullable = true)

|-- \_col4: byte (nullable = true)

|-- \_col5: string (nullable = true)

|-- \_col6: string (nullable = true)

|-- \_col7: integer (nullable = true)

|-- \_col8: decimal(9,2) (nullable = true)

|-- \_col9: date (nullable = true)

|-- \_col10: string (nullable = true)

|-- \_col11: short (nullable = true)

val store\_item\_sales\_data2 = store\_item\_sales\_data.toDF("sales\_id","Sales\_date","store\_id","item\_id","scan\_type","geo\_region\_cd","currency\_code","scan\_id","sold\_unit\_quantity","scan\_date","scan\_time","scan\_dept\_nbr")

scala> store\_item\_sales\_data2.printSchema

root

|-- sales\_id: integer (nullable = true)

|-- Sales\_date: date (nullable = true)

|-- store\_id: integer (nullable = true)

|-- item\_id: integer (nullable = true)

|-- scan\_type: byte (nullable = true)

|-- geo\_region\_cd: string (nullable = true)

|-- currency\_code: string (nullable = true)

|-- scan\_id: integer (nullable = true)

|-- sold\_unit\_quantity: decimal(9,2) (nullable = true)

|-- scan\_date: date (nullable = true)

|-- scan\_time: string (nullable = true)

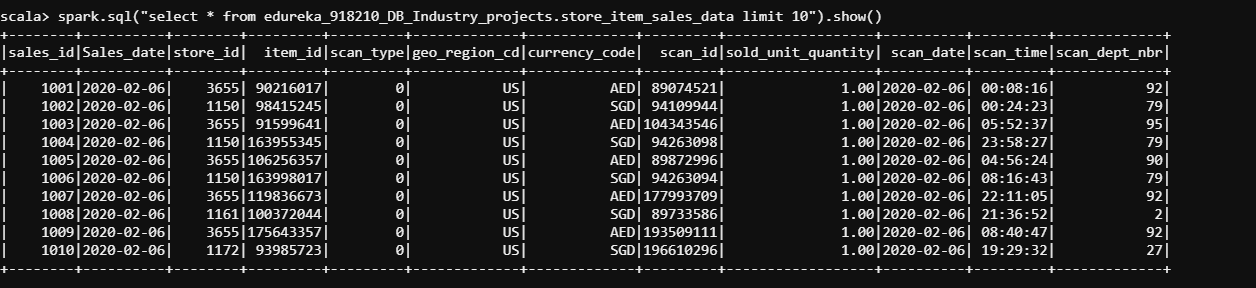
|-- scan\_dept\_nbr: short (nullable = true)

=======================================================================================================

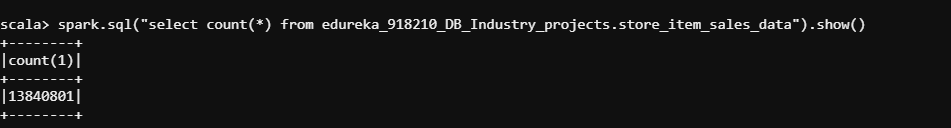
SPARK TO Hive

store\_item\_sales\_data2.write.mode("overwrite").saveAsTable("edureka\_918210\_DB\_Industry\_projects.store\_item\_sales\_data")

spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.store\_item\_sales\_data limit 10").show()

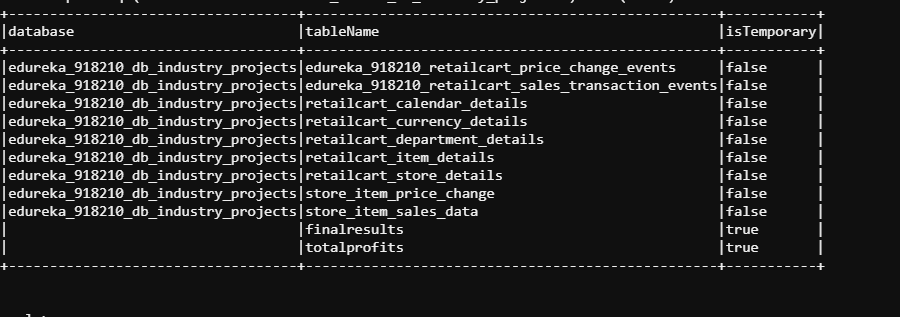


scala> spark.sql("select count(\*) from edureka\_918210\_DB\_Industry\_projects.store\_item\_sales\_data").show()



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

scala> spark.sql("Show tables in edureka\_918210\_DB\_Industry\_projects").show(false)



\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*.

**REAL TIME DATA**

/mnt/home/edureka\_918210/project\_retailcart/realtimedata

[edureka\_918210@ip-20-0-41-164 realtimedata]$ pwd

/mnt/home/edureka\_918210/project\_retailcart/realtimedata

[edureka\_918210@ip-20-0-41-164 realtimedata]$

[edureka\_918210@ip-20-0-41-164 realtimedata]$ ls -lart

total 60

-rwxr-xr-x 1 edureka\_918210 edureka\_918210 41546 Jan 10 15:03 price\_change\_event.txt

-rwxr-xr-x 1 edureka\_918210 edureka\_918210 5029 Jan 10 15:03 real\_time\_simulator.py

drwxr-xr-x 2 edureka\_918210 edureka\_918210 6144 Jan 10 15:03 .

python2 realtimedata/real\_time\_simulator.py edureka\_918210

labuser\_database.username\_retailcart\_sales\_transaction\_events

labuser\_database.username\_retailcart\_price\_change\_events

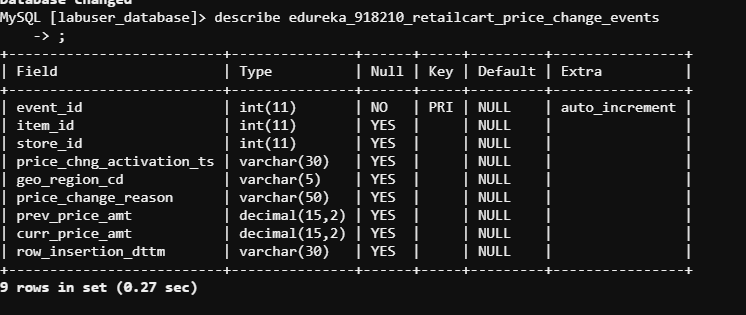
mysql -u edu\_labuser -h dbserver.edu.cloudlab.com -p

password - edureka

DB - labuser\_database

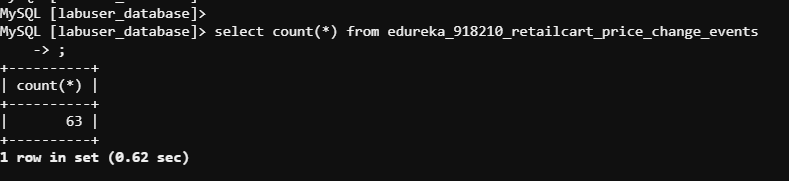
MySQL [labuser\_database]> describe edureka\_918210\_retailcart\_price\_change\_events

-> ;



MySQL [labuser\_database]> select count(\*) from edureka\_918210\_retailcart\_price\_change\_events

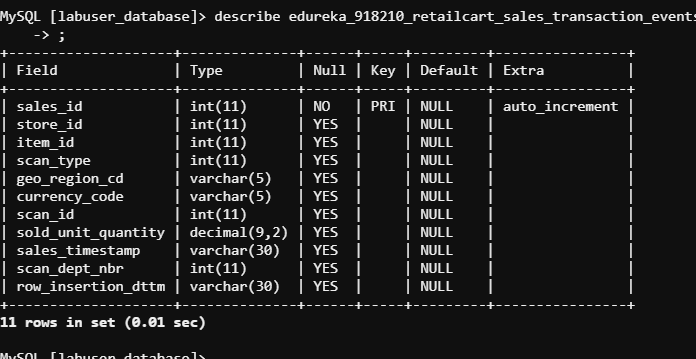
-> ;



MySQL [labuser\_database]>

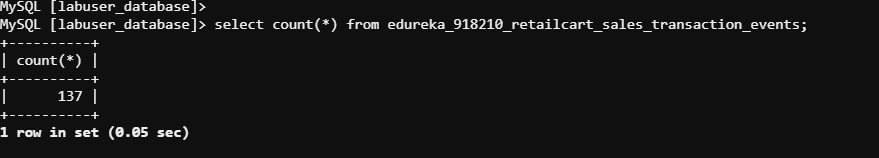
MySQL [labuser\_database]>

MySQL [labuser\_database]> describe edureka\_918210\_retailcart\_sales\_transaction\_events



MySQL [labuser\_database]>

MySQL [labuser\_database]> select count(\*) from edureka\_918210\_retailcart\_sales\_transaction\_events;



MySQL [labuser\_database]>

LOCATION

/mnt/home/edureka\_918210/project\_retailcart/history\_data - File Location

flume1.conf - File Name

SPARK STREAMING - FLUME

agent.channels.ch1.type = memory

agent.sources.sql-source.channels = ch1

agent.channels = ch1

agent.sinks = HDFS

agent.sources = sql-source

agent.sources.sql-source.type = org.keedio.flume.source.SQLSource

agent.sources.sql-source.connection.url = jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database

agent.sources.sql-source.user = edu\_labuser

agent.sources.sql-source.password = edureka

agent.sources.sql-source.table = labuser\_database.edureka\_918210\_retailcart\_sales\_transaction\_events

agent.sources.sql-source.columns.to.select = \*

agent.sources.sql-source.incremental.column.name = sales\_id

agent.sources.sql-source.run.query.delay=5000

agent.sinks.HDFS.channel = ch1

agent.sinks.HDFS.type = hdfs

agent.sinks.HDFS.hdfs.path = hdfs://nameservice1/user/edureka\_918210/project\_retailcart

agent.sinks.HDFS.hdfs.fileType = DataStream

agent.sinks.HDFS.hdfs.writeFormat = Text

agent.sinks.HDFS.hdfs.rollSize = 268435456

agent.sinks.HDFS.hdfs.rollInterval = 0

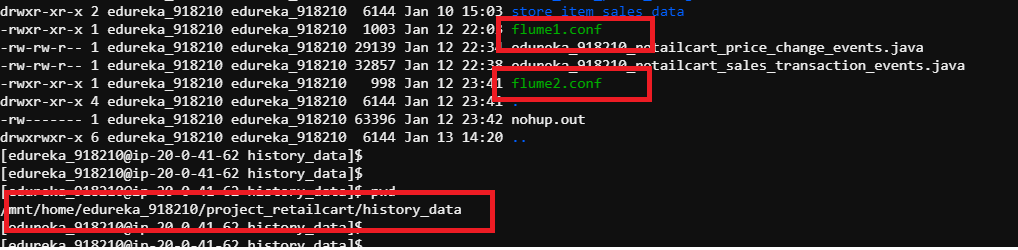
agent.sinks.HDFS.hdfs.rollCount = 0

**COMMANDS :**

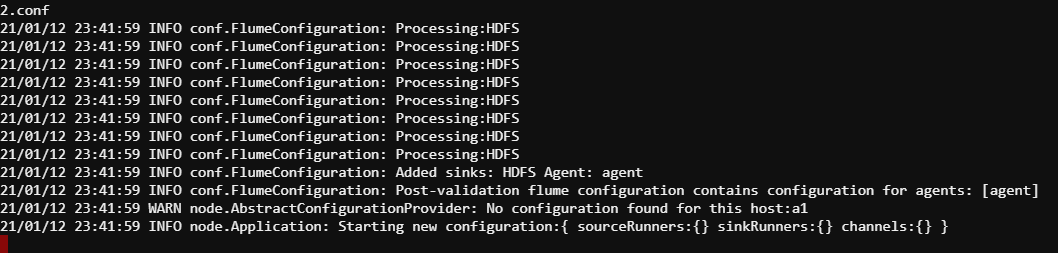
**FLUME CONFIGURATION :**

nohup flume-ng agent -n a1 -f /mnt/home/edureka\_918210/project\_retailcart/history\_data/flume1.conf -Dflume.root.logger=INFO,console &

nohup flume-ng agent -n a1 -f /mnt/home/edureka\_918210/project\_retailcart/history\_data/flume1.conf -Dflume.root.logger=INFO,console & --- for labuser\_database.edureka\_918210\_retailcart\_price\_change\_events



**FLUME Logger**



**hive tables from mysql**

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database \

--username edu\_labuser \

--P \

--split-by event\_id \

--columns event\_id,item\_id,store\_id,price\_chng\_activation\_ts,geo\_region\_cd,price\_change\_reason,prev\_price\_amt,curr\_price\_amt,row\_insertion\_dttm \

--table edureka\_918210\_retailcart\_price\_change\_events \

--target-dir /user/edureka\_918210/project\_retailcart/edureka\_918210\_retailcart\_price\_change\_events \

--fields-terminated-by "," \

--hive-import \

--create-hive-table \

--hive-table edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_price\_change\_events

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database \

--username edu\_labuser \

--P \

--split-by sales\_id \

--columns sales\_id,store\_id,item\_id,scan\_type,geo\_region\_cd,currency\_code,scan\_id,sold\_unit\_quantity,sales\_timestamp,scan\_dept\_nbr,row\_insertion\_dttm \

--table edureka\_918210\_retailcart\_sales\_transaction\_events \

--target-dir /user/edureka\_918210/project\_retailcart/edureka\_918210\_retailcart\_sales\_transaction\_events \

--fields-terminated-by "," \

--hive-import \

--create-hive-table \

--hive-table edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_sales\_transaction\_events

/user/edureka\_918210/edureka\_918210\_retailcart\_price\_change\_events

/user/edureka\_918210/edureka\_918210\_retailcart\_sales\_transaction\_events

spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_price\_change\_events").show(false)

spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_sales\_transaction\_events").show(false)

**MYSQL TO HBase**

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table edureka\_918210\_retailcart\_sales\_transaction\_events --hbase-table 'edureka\_918210\_retailcart\_sales\_transaction\_events' --column-family cf2 --username edu\_labuser --hbase-create-table --columns sales\_id,store\_id,item\_id,scan\_type,geo\_region\_cd,currency\_code,scan\_id,sold\_unit\_quantity,sales\_timestamp,scan\_dept\_nbr,row\_insertion\_dttm --hbase-row-key sales\_id -m 1 --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table edureka\_918210\_retailcart\_price\_change\_events --hbase-table 'edureka\_918210\_retailcart\_price\_change\_events' --column-family cf2 --username edu\_labuser --hbase-create-table --columns event\_id,item\_id,store\_id,price\_chng\_activation\_ts,geo\_region\_cd,price\_change\_reason,prev\_price\_amt,curr\_price\_amt,row\_insertion\_dttm --hbase-row-key event\_id -m 1 --password edureka

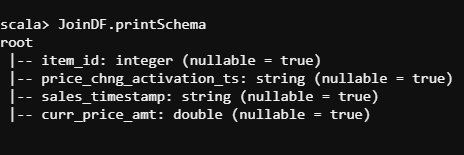
**SAVE TO DF and CREATE JOINS**

val edureka\_918210\_retailcart\_price\_change\_events\_DF = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_price\_change\_events")

val edureka\_918210\_retailcart\_sales\_transaction\_events\_DF = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects.edureka\_918210\_retailcart\_sales\_transaction\_events")

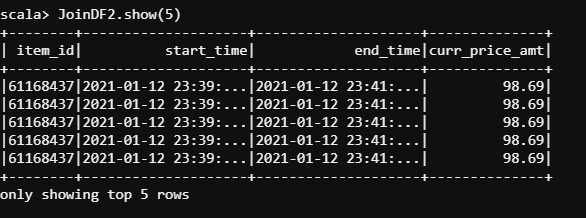
val JoinDF = edureka\_918210\_retailcart\_price\_change\_events\_DF.join(edureka\_918210\_retailcart\_sales\_transaction\_events\_DF,"item\_id").select("item\_id","price\_chng\_activation\_ts","sales\_timestamp","curr\_price\_amt")

scala> JoinDF.printSchema



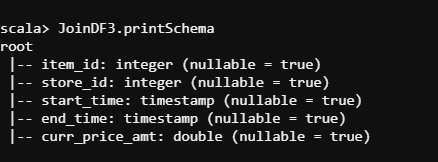
val JoinDF2 = JoinDF.selectExpr("item\_id","cast(price\_chng\_activation\_ts as timestamp) start\_time","cast(sales\_timestamp as timestamp) end\_time","curr\_price\_amt")

scala> JoinDF2.show(5)



val JoinDF3 = JoinDF2.join(edureka\_918210\_retailcart\_sales\_transaction\_events\_DF,"item\_id").select("item\_id","store\_id","start\_time","end\_time","curr\_price\_amt")

scala> JoinDF3.printSchema



scala> JoinDF3.repartition(1).write.format("csv").save("/user/edureka\_918210/project\_retailcart/sales\_result")

[edureka\_918210@ip-20-0-41-164 project\_retailcart]$ hadoop fs -ls /user/edureka\_918210/project\_retailcart/sales\_result

Found 2 items

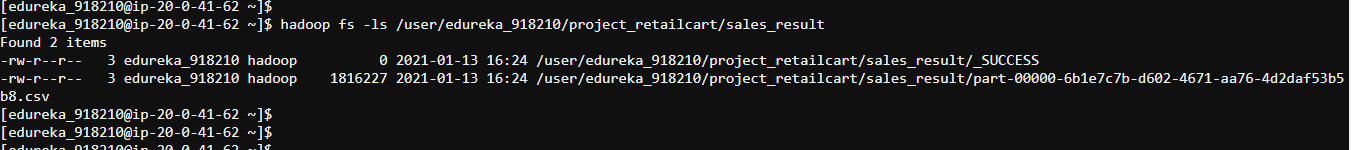
-rw-r--r-- 3 edureka\_918210 hadoop 0 2021-01-13 16:24 /user/edureka\_918210/project\_retailcart/sales\_result/\_SUCCESS

-rw-r--r-- 3 edureka\_918210 hadoop 1816227 2021-01-13 16:24 /user/edureka\_918210/project\_retailcart/sales\_result/part-00000-6b1e7c7b-d602-4671-aa76-4d2daf53b5

b8.csv

[edureka\_918210@ip-20-0-41-164 project\_retailcart]$

[edureka\_918210@ip-20-0-41-164 project\_retailcart]$



**OBJECTIVES :**

Total sales in a day/week/quarter (both in terms of sales revenue and units sold)

• Top 5 most selling item categories in a day (both in terms of amount and quantity)

• Top 5 most profitable items in a day/week/quarter

• Top 5 most profitable stores in a day/week/month/quarter

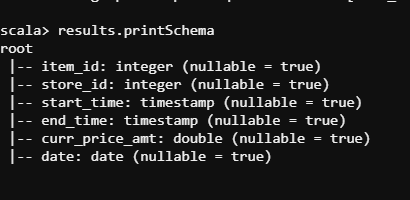
• Total profit or loss in U.S. dollars every day in percentage

Top 5 most selling item categories in a day (both in terms of amount and quantity)

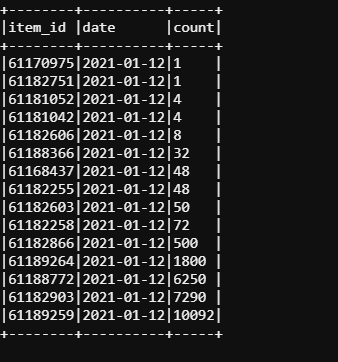
======================================================================================

scala> JoinDF3.printSchema

val results = JoinDF3.withColumn("date", to\_date($"start\_time"))



scala> results.groupBy("item\_id","date").count().orderBy("count").show(false)

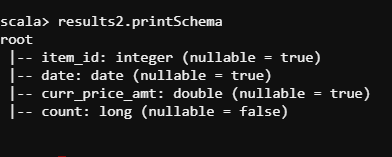


• Top 5 most profitable items in a day/week/quarter

=========================================================

val results2 = results.groupBy("item\_id","date","curr\_price\_amt").count().orderBy("count")

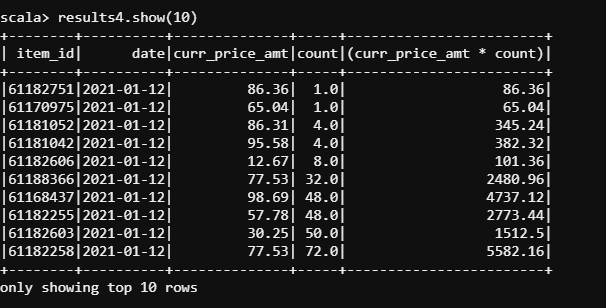
scala> results2.printSchema



val results3 = results2.selectExpr("item\_id","date","curr\_price\_amt","cast(count as double) count")

val results4 = results3.selectExpr("item\_id","date","curr\_price\_amt","count","curr\_price\_amt \* count")

scala> results4.show(10)



**Top 5 most profitable stores in a day/week/month/quarter**

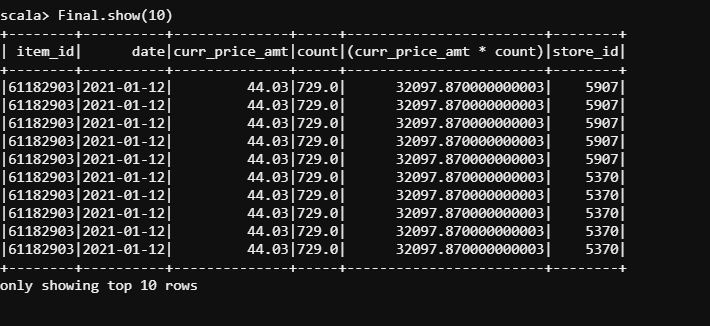
JoinDF3.printSchema

results4.printSchema

val JoinDF4 = JoinDF3.selectExpr("item\_id","store\_id")

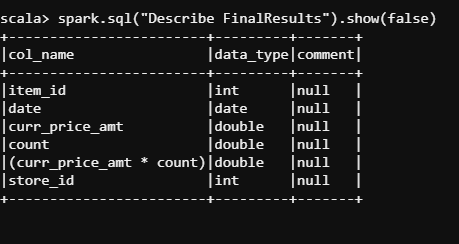
val Final = results4.join(JoinDF4,"item\_id").select("item\_id","date","curr\_price\_amt","count","(curr\_price\_amt \* count)","store\_id")

scala> Final.show(10)

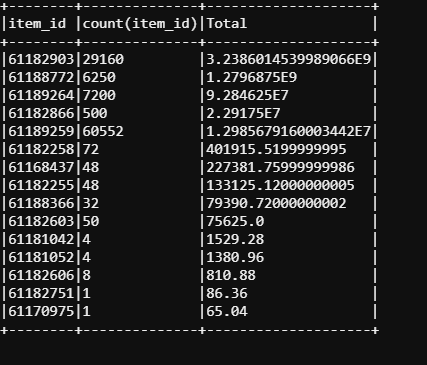


Final.createOrReplaceTempView("FinalResults")

spark.sql("Describe FinalResults").show(false)

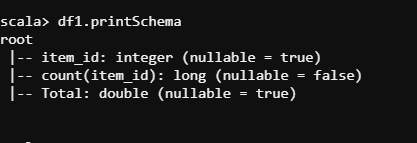


scala> spark.sql("SELECT item\_id,count(item\_id),sum((curr\_price\_amt \* count)) as Total FROM FinalResults GROUP BY item\_id ORDER BY TOTAL desc").show(false)



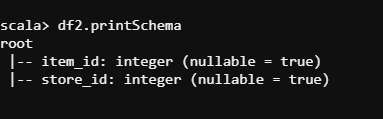
val df1 = spark.sql("SELECT item\_id,count(item\_id),sum((curr\_price\_amt \* count)) as Total FROM FinalResults GROUP BY item\_id ORDER BY TOTAL desc")

scala> df1.printSchema



val df2 = Final.selectExpr("item\_id","store\_id")

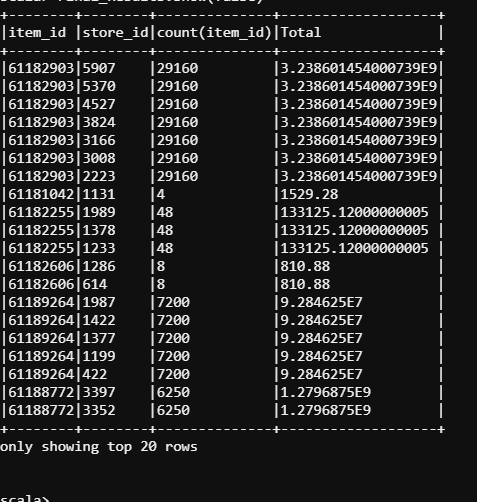
scala> df2.printSchema



val Final\_Results = df2.join(df1,"item\_id").distinct()

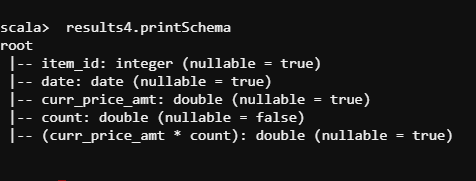
**Top 5 most profitable stores in a day/week/month/quarter**

scala> Final\_Results.show(false)



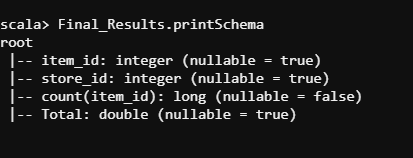
**• Total profit or loss in U.S. dollars every day in percentage**

scala> results4.printSchema



val results5 = results4.selectExpr("item\_id","date")

scala> Final\_Results.printSchema

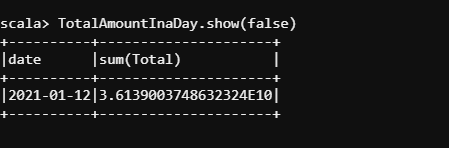


val Final\_Results2 = Final\_Results.join(results5,"item\_id").distinct()

Final\_Results2.createOrReplaceTempView("totalprofits")

val TotalAmountInaDay = spark.sql("select date,sum(Total) from totalprofits group by date")

scala> TotalAmountInaDay.show(false)



**TABLEAU REPRESENTATION OF SALES PRICE IN A DAY/WEEK/MONTH**

