

Industry GRADE PROJECT 3

Customer Retention Strategy



Edureka

Edureka

**Business Challenge/Requirement**

Customer retention and acquisition strategies are on top of every organization’s agenda.

To offer better customer service and boost loyalty, a company has to invest in a state-of-the-art CRM tool.

In pursuit of these goals, every organization implements CRM as a strategy that integrates the concepts of data mining and data warehousing. The data collected through the CRM helps the leadership team make actionable decisions in real time.

It helps them build and retain long-term and profitable relationships with customers. FutureCart Inc. is a hypothetical leading retail company with an omnipresence in India with more than 5000 retail stores and hypermarkets across and e-commerce in the country. The company has formed a dedicated team to handle after-sales services. The team is entrusted with the responsibility to address customer complaints and delight them - and eventually increase brand loyalty

**Abstract**

The company has multiple contact centers across India to provide support service to their customers

• Customers can reach out to the care team over different communication channels depending on their preference and convenience: Calls, Chat, or Email. • CCR (Customer Care Representative) registers the complaint by collecting all the necessary details - - which is called a case

• A case can have a status -- open or closed

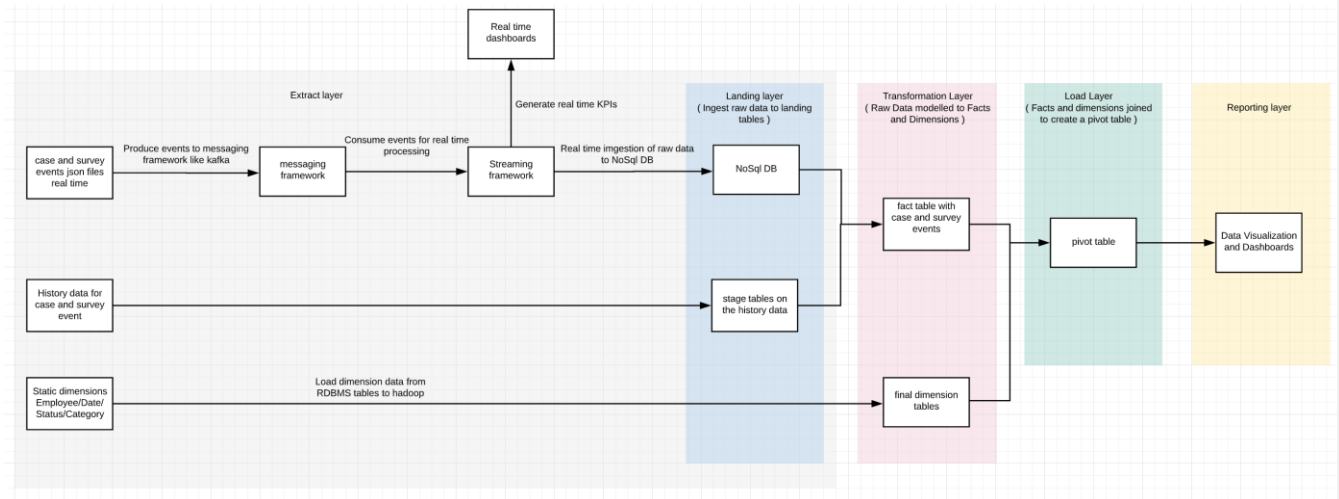
• Each case can belong to a category and sub-category. This category and sub-category will determine case priority. Depending on the priority key, CCR has an SLA (in hours) to close the case within the SLA hours

• Once a case is closed, the customer is sent a survey link to rate the overall experience of interacting with the contact center representative

• The customer can take a survey or leave it unattended. The customer can rate the experience on a scale of 1-10 on various questions

• Survey response is captured for that particular case

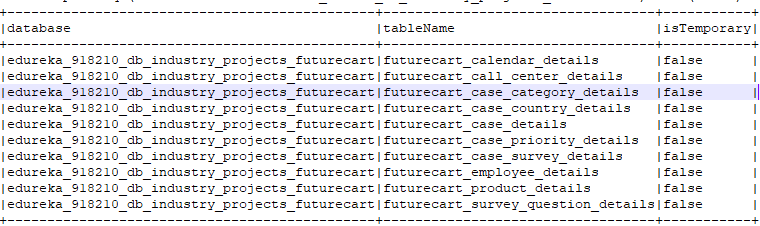
**Data-Flow Architecture Diagram**



**Details of the fact and dimension tables :**

Hive DB - edureka\_918210\_DB\_Industry\_projects\_futurecart

Fact Tables :



**Customer Retention Strategy**

[edureka\_918210@ip-20-0-41-164 batchdata]$ hadoop fs -ls /user/edureka\_918210/project\_futurecart/batchdata

Found 10 items

-rw-r--r-- 3 edureka\_918210 hadoop 1822863 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_calendar\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 864 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_call\_center\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 965 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_category\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 4231 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_country\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 75311462 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 523 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_priority\_details.txt

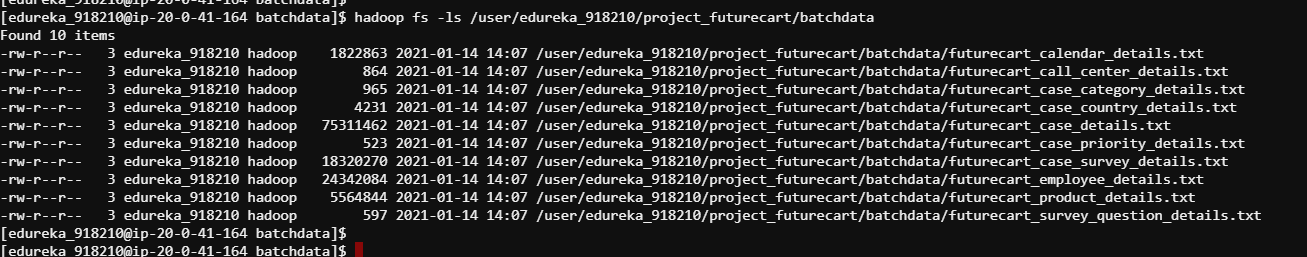
-rw-r--r-- 3 edureka\_918210 hadoop 18320270 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_survey\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 24342084 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_employee\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 5564844 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_product\_details.txt

-rw-r--r-- 3 edureka\_918210 hadoop 597 2021-01-14 14:07 /user/edureka\_918210/project\_futurecart/batchdata/futurecart\_survey\_question\_details.txt

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



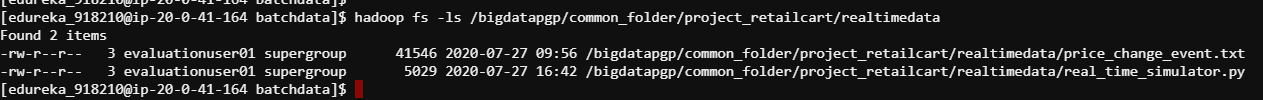
[edureka\_918210@ip-20-0-41-62 project\_retailcart]$ hadoop fs -ls /bigdatapgp/common\_folder/project\_retailcart/realtimedata

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

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



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};

**BATCH DATA**

**futurecart\_calendar\_details**

futurecart\_calendar\_details.txt

**Header 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\_futurecart/batchdata/futurecart\_calendar\_details.txt").createOrReplaceTempView("futurecart\_calendar\_details");

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

val futurecart\_calendar\_details =

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,

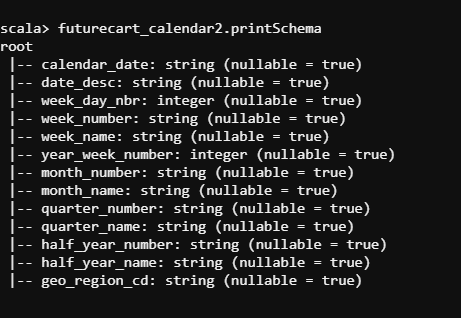
split(value,'\t')[10] as half\_year\_number,

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

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

from futurecart\_calendar\_details """)

val futurecart\_calendar2 = futurecart\_calendar\_details.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")



**futurecart\_call\_center\_details.txt**

call\_center\_id call\_center\_vendor location country

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_call\_center\_details.txt").createOrReplaceTempView("futurecart\_call\_center\_details");

val futurecart\_call\_center\_details =

spark.sql(""" Select

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

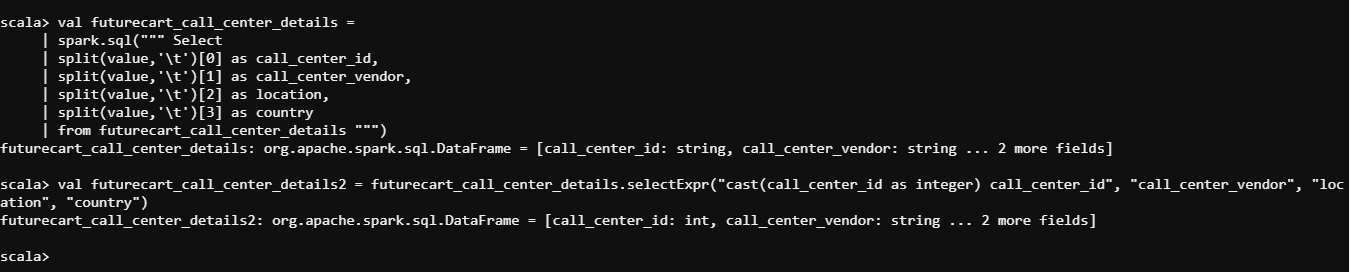
split(value,'\t')[1] as call\_center\_vendor,

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

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

from futurecart\_call\_center\_details """)

val futurecart\_call\_center\_details2 = futurecart\_call\_center\_details.selectExpr("cast(call\_center\_id as integer) call\_center\_id", "call\_center\_vendor", "location", "country")



**futurecart\_case\_category\_details.txt**

category\_key sub\_category\_key category\_description sub\_category\_description priority

CAT1 SCAT1 Subscription Renewal P1

CAT1 SCAT2 Subscription Termination P6

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_category\_details.txt").createOrReplaceTempView("futurecart\_case\_category\_details");

val futurecart\_case\_category\_details =

spark.sql(""" Select

split(value,'\t')[0] as category\_key,

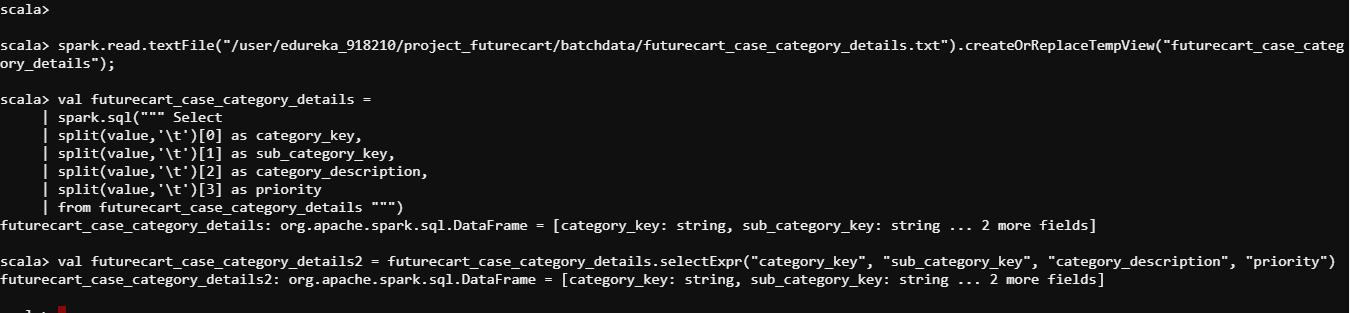
split(value,'\t')[1] as sub\_category\_key,

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

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

from futurecart\_case\_category\_details """)

val futurecart\_case\_category\_details2 = futurecart\_case\_category\_details.selectExpr("category\_key", "sub\_category\_key", "category\_description", "priority")



**futurecart\_case\_country\_details.txt**

id name alpha\_2 alpha\_3

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_country\_details.txt").createOrReplaceTempView("futurecart\_case\_country\_details");

val futurecart\_case\_country\_details =

spark.sql(""" Select

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

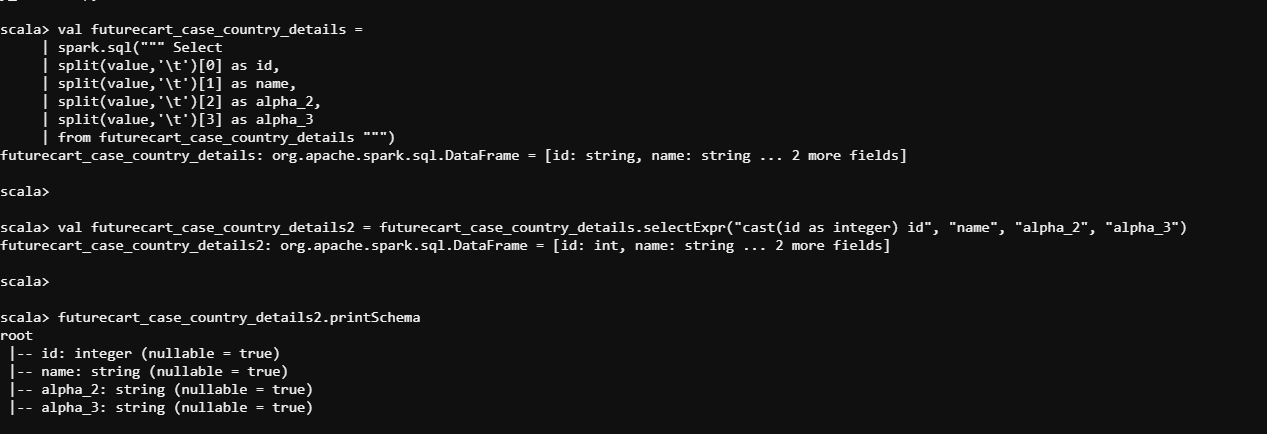
split(value,'\t')[1] as name,

split(value,'\t')[2] as alpha\_2,

split(value,'\t')[3] as alpha\_3

from futurecart\_case\_country\_details """)

val futurecart\_case\_country\_details2 = futurecart\_case\_country\_details.selectExpr("cast(id as integer) id", "name", "alpha\_2", "alpha\_3")



**futurecart\_case\_details.txt**

case\_no create\_timestamp last\_modified\_timestamp created\_employee\_key call\_center\_id status category sub\_category communication\_mode country\_cd product\_code

2024 2020-04-20 01:01:29 2020-04-20 01:01:29 274649 C-104 Open CAT1 SCAT1 Email PY 997719

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_details.txt").createOrReplaceTempView("futurecart\_case\_details");

val futurecart\_case\_details =

spark.sql(""" Select

split(value,'\t')[0] as case\_no,

split(value,'\t')[1] as create\_timestamp,

split(value,'\t')[2] as last\_modified\_timestamp,

split(value,'\t')[3] as created\_employee\_key,

split(value,'\t')[4] as call\_center\_id,

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

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

split(value,'\t')[7] as sub\_category,

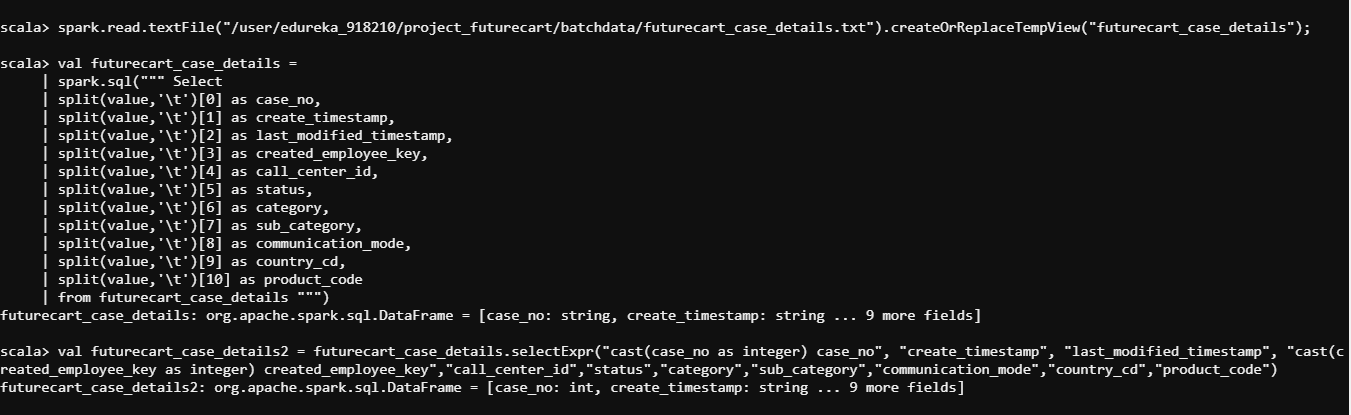
split(value,'\t')[8] as communication\_mode,

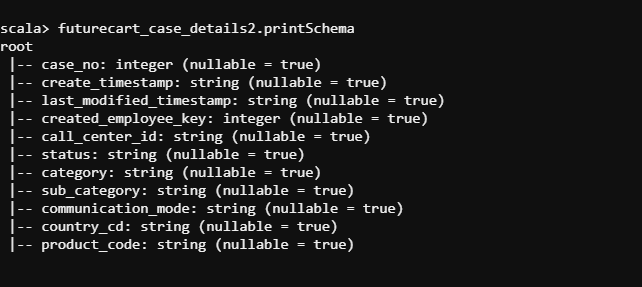
split(value,'\t')[9] as country\_cd,

split(value,'\t')[10] as product\_code

from futurecart\_case\_details """)

val futurecart\_case\_details2 = futurecart\_case\_details.selectExpr("cast(case\_no as integer) case\_no", "create\_timestamp", "last\_modified\_timestamp", "cast(created\_employee\_key as integer) created\_employee\_key","call\_center\_id","status","category","sub\_category","communication\_mode","country\_cd","product\_code")





**futurecart\_case\_priority\_details.txt**

priority\_key priority severity SLA

P1 Highest Critical 1

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_priority\_details.txt").createOrReplaceTempView("futurecart\_case\_priority\_details");

val futurecart\_case\_priority\_details =

spark.sql(""" Select

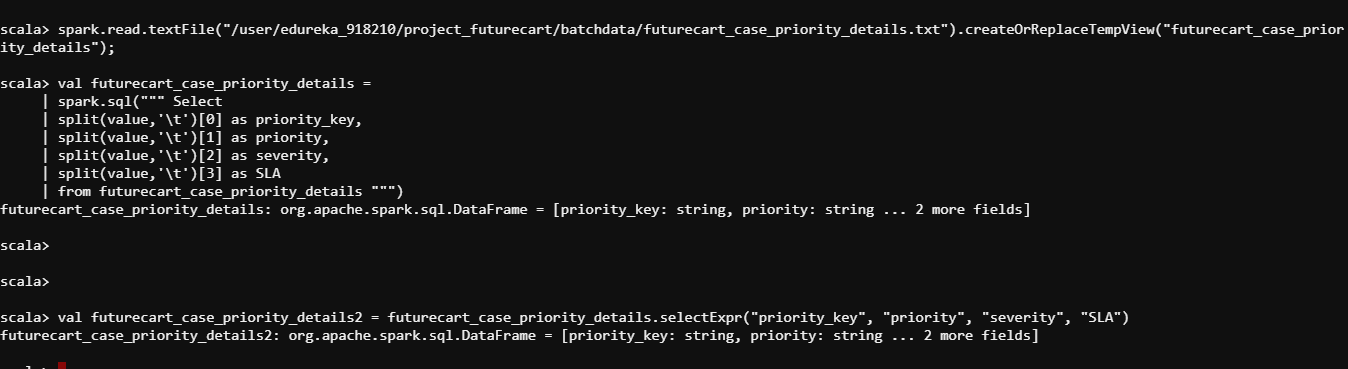
split(value,'\t')[0] as priority\_key,

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

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

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

from futurecart\_case\_priority\_details """)



**futurecart\_case\_survey\_details.txt**

/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_survey\_details.txt

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_case\_survey\_details.txt").createOrReplaceTempView("futurecart\_case\_survey\_details");

survey\_id case\_no survey\_timestamp q1 q2 q3 q4 q5

S-1000 130114 2020-04-26 00:08:28 2 7 1 N 7

val futurecart\_case\_survey\_details =

spark.sql(""" Select

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

split(value,'\t')[1] as case\_no,

split(value,'\t')[2] as survey\_timestamp,

split(value,'\t')[3] as q1,

split(value,'\t')[4] as q2,

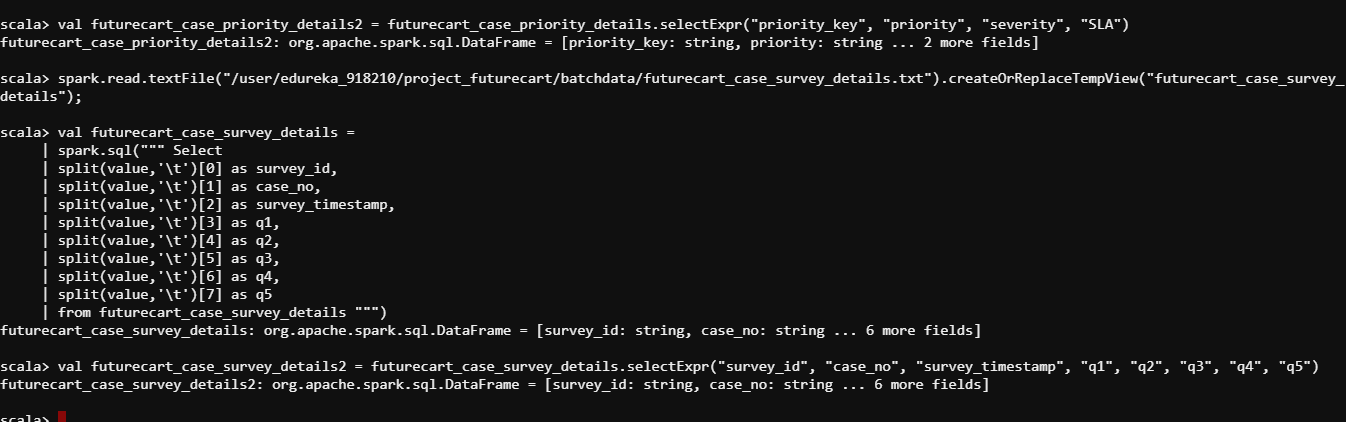
split(value,'\t')[5] as q3,

split(value,'\t')[6] as q4,

split(value,'\t')[7] as q5

from futurecart\_case\_survey\_details """)

val futurecart\_case\_survey\_details2 = futurecart\_case\_survey\_details.selectExpr("survey\_id", "case\_no", "survey\_timestamp", "q1", "q2", "q3", "q4", "q5")



**futurecart\_employee\_details.txt**

/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_employee\_details.txt

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_employee\_details.txt").createOrReplaceTempView("futurecart\_employee\_details");

emp\_key first\_name last\_name email gender ldap hire\_date manager

10001 Georgi Facello Georgi.Facello01@testmail.com M 5941CF7D 2014-04-06

val futurecart\_employee\_details =

spark.sql(""" Select

split(value,'\t')[0] as emp\_key,

split(value,'\t')[1] as first\_name,

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

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

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

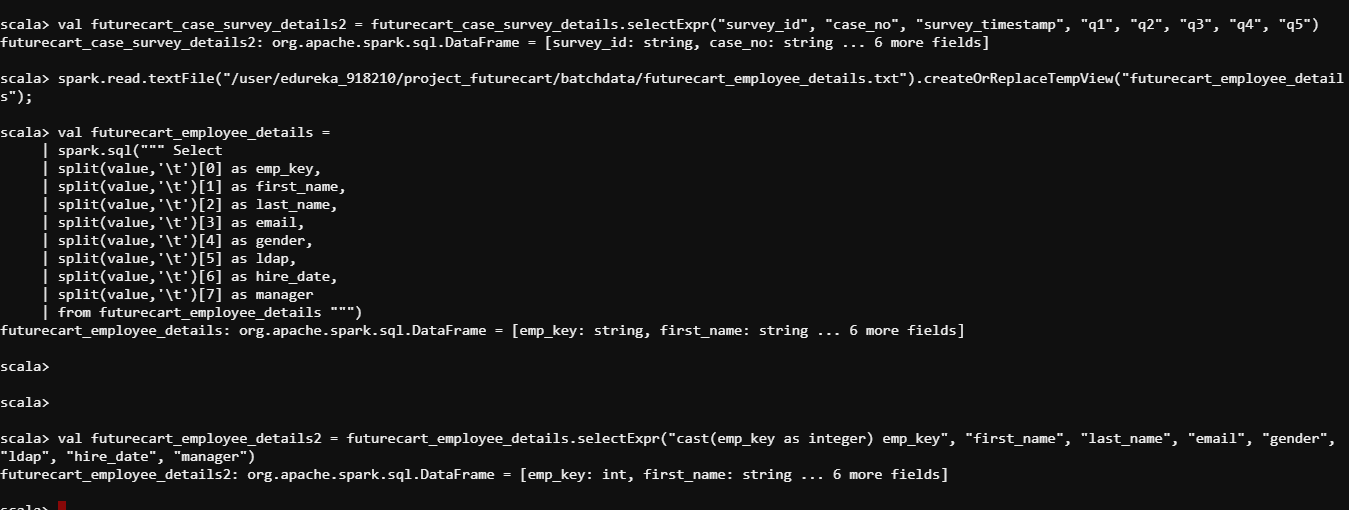
split(value,'\t')[5] as ldap,

split(value,'\t')[6] as hire\_date,

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

from futurecart\_employee\_details """)

val futurecart\_employee\_details2 = futurecart\_employee\_details.selectExpr("cast(emp\_key as integer) emp\_key", "first\_name", "last\_name", "email", "gender", "ldap", "hire\_date", "manager")



**futurecart\_product\_details.txt**

/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_product\_details.txt

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_product\_details.txt").createOrReplaceTempView("futurecart\_product\_details");

product\_id department brand commodity\_desc sub\_commodity\_desc

25671 GROCERY National FRZN ICE ICE - CRUSHED/CUBED

26081 MISC. TRANS. National NO COMMODITY DESCRIPTION NO SUBCOMMODITY DESCRIPTION

val futurecart\_product\_details =

spark.sql(""" Select

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

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

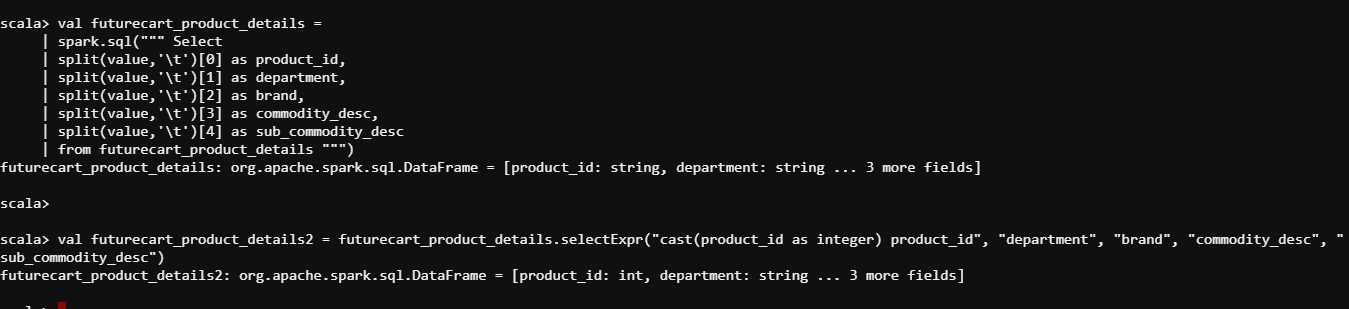
split(value,'\t')[2] as brand,

split(value,'\t')[3] as commodity\_desc,

split(value,'\t')[4] as sub\_commodity\_desc

from futurecart\_product\_details """)

val futurecart\_product\_details2 = futurecart\_product\_details.selectExpr("cast(product\_id as integer) product\_id", "department", "brand", "commodity\_desc", "sub\_commodity\_desc")



**futurecart\_survey\_question\_details**

question\_id question\_desc response\_type range negative\_response\_range neutral\_response\_range positive\_response\_range

Q1 How would you rate your overall experience with the customer support process? Scale 1-10 1-4 5-7 8-10

/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_survey\_question\_details.txt

spark.read.textFile("/user/edureka\_918210/project\_futurecart/batchdata/futurecart\_survey\_question\_details.txt").createOrReplaceTempView("futurecart\_survey\_question\_details");

val futurecart\_survey\_question\_details =

spark.sql(""" Select

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

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

split(value,'\t')[2] as response\_type,

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

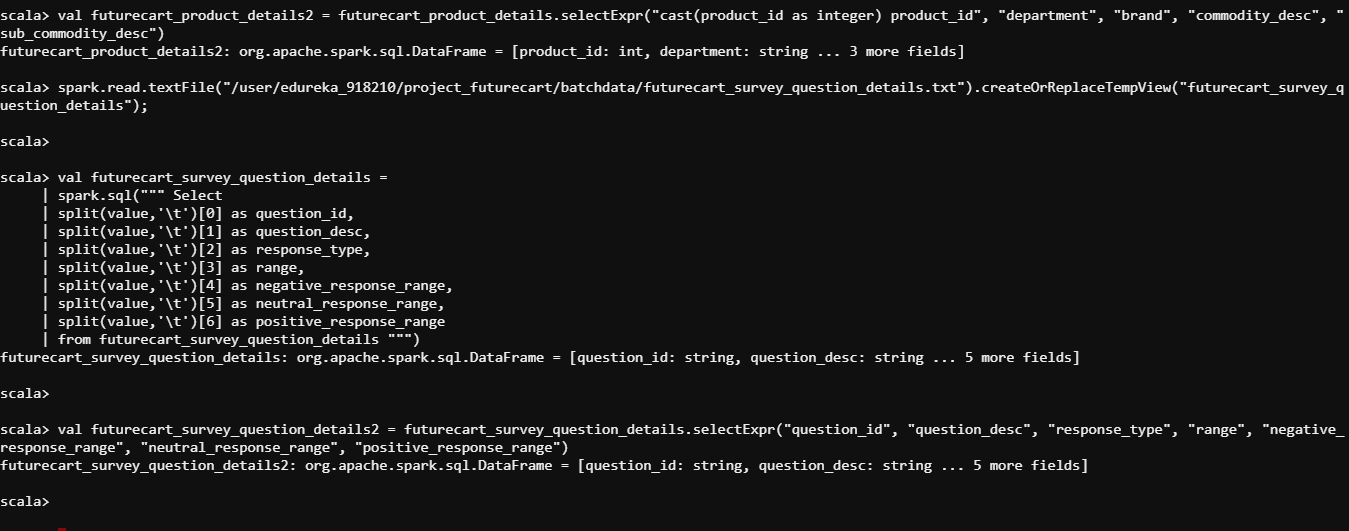
split(value,'\t')[4] as negative\_response\_range,

split(value,'\t')[5] as neutral\_response\_range,

split(value,'\t')[6] as positive\_response\_range

from futurecart\_survey\_question\_details """)

val futurecart\_survey\_question\_details2 = futurecart\_survey\_question\_details.selectExpr("question\_id", "question\_desc", "response\_type", "range", "negative\_response\_range", "neutral\_response\_range", "positive\_response\_range")



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

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

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

futurecart\_call\_center\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_call\_center\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_case\_category\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_case\_category\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_case\_country\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_case\_country\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_case\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_case\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_case\_priority\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_case\_priority\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_case\_survey\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_case\_survey\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_employee\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_employee\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_product\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_product\_details2").option("user","edu\_labuser").option("password","edureka").option("driver","com.mysql.jdbc.Driver").mode("overwrite").save

futurecart\_survey\_question\_details2.write.format("jdbc").option("url","jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database").option("dbtable","futurecart\_survey\_question\_details2").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 futurecart\_calendar\_details SELECT \* FROM futurecart\_calendar2;

CREATE TABLE futurecart\_call\_center\_details SELECT \* FROM futurecart\_call\_center\_details2;

CREATE TABLE futurecart\_case\_category\_details SELECT \* FROM futurecart\_case\_category\_details2;

CREATE TABLE futurecart\_case\_country\_details SELECT \* FROM futurecart\_case\_country\_details2;

CREATE TABLE futurecart\_case\_details SELECT \* FROM futurecart\_case\_details2;

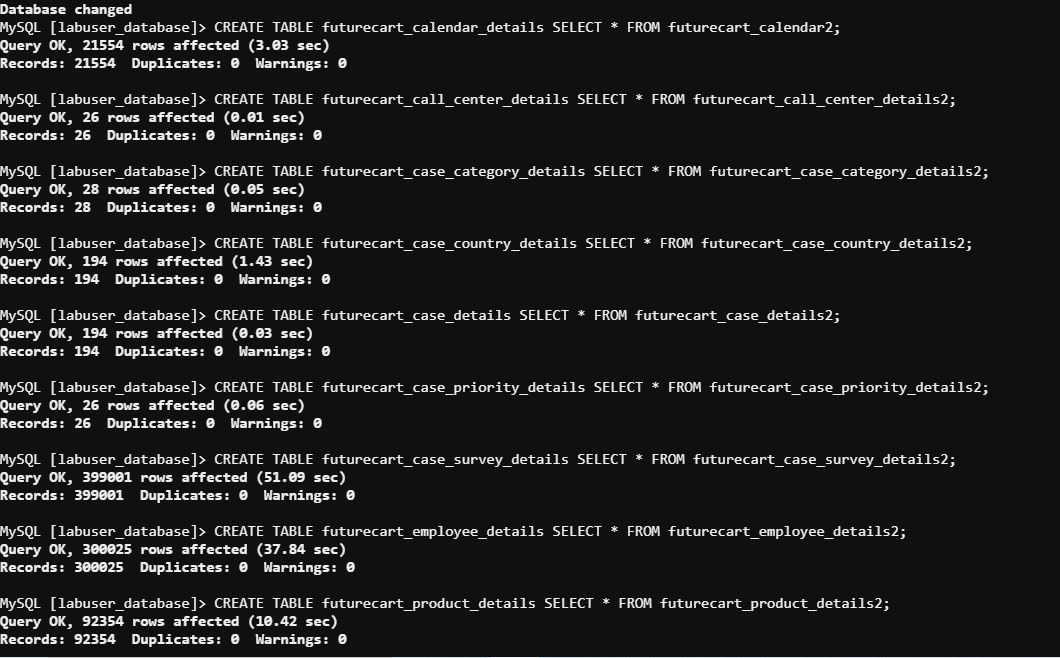
CREATE TABLE futurecart\_case\_priority\_details SELECT \* FROM futurecart\_case\_priority\_details2;

CREATE TABLE futurecart\_case\_survey\_details SELECT \* FROM futurecart\_case\_survey\_details2;

CREATE TABLE futurecart\_employee\_details SELECT \* FROM futurecart\_employee\_details2;

CREATE TABLE futurecart\_product\_details SELECT \* FROM futurecart\_product\_details2;

CREATE TABLE futurecart\_survey\_question\_details SELECT \* FROM futurecart\_survey\_question\_details2;



DB Details :edureka\_918210\_DB\_Industry\_projects\_futurecart

**MYSQL TO HIVE**

MYSQL TO HIVE

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

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

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

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table futurecart\_case\_category\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects\_futurecart --split-by sub\_category\_key --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table futurecart\_case\_country\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects\_futurecart --split-by id --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table futurecart\_case\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects\_futurecart --split-by case\_no --password edureka

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table futurecart\_case\_priority\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects\_futurecart --split-by priority\_key --password edureka

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

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table futurecart\_employee\_details -m 2 --hive-import --username edu\_labuser --hive-database edureka\_918210\_DB\_Industry\_projects\_futurecart --split-by emp\_key --password edureka

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

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

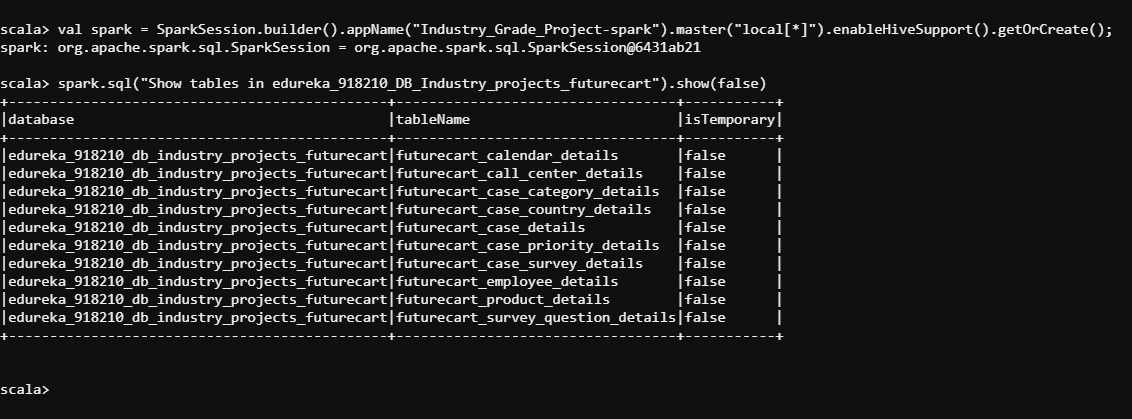


SPARK TO Hive

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

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

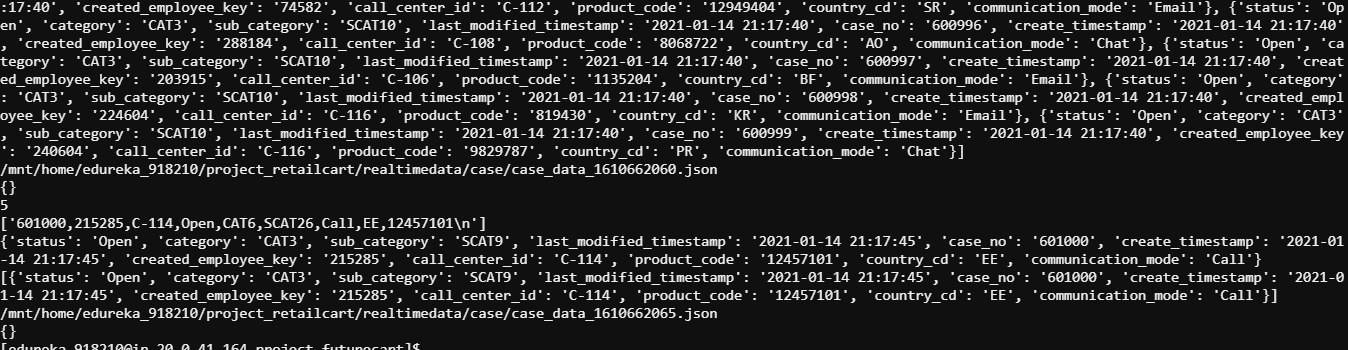
spark.sql("Show tables in edureka\_918210\_DB\_Industry\_projects\_futurecart").show(false)

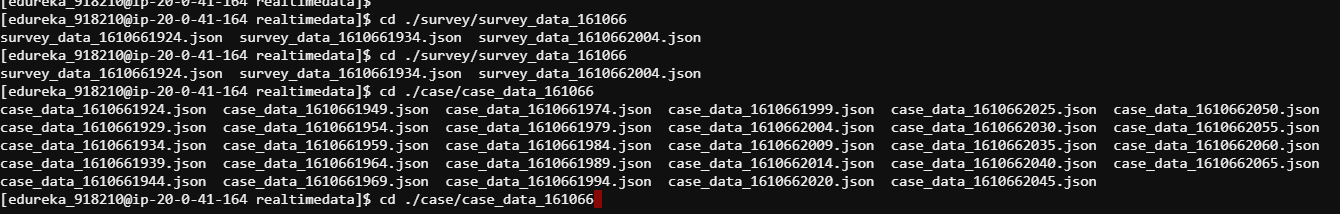


REAL TIME DATA

hdfs dfs -get /bigdatapgp/common\_folder/project\_futurecart/realtimedata

python2 realtimedata/realtime\_simulator.py --outputLocation /mnt/home/edureka\_918210/project\_retailcart/realtimedata





python2 realtimedata/realtime\_simulator.py --outputLocation /mnt/home/edureka\_918210/project\_retailcart/realtimedata

**LOCATION :**

/mnt/home/edureka\_918210/project\_futurecart/realtimedata/case - File Location

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

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

val RealTimeDFCase = spark.read.json("/user/edureka\_918210/project\_futurecart/realtimedata/case")

RealTimeDFCase: org.apache.spark.sql.DataFrame = [call\_center\_id: string, case\_no: string ... 9 more fields]

scala>

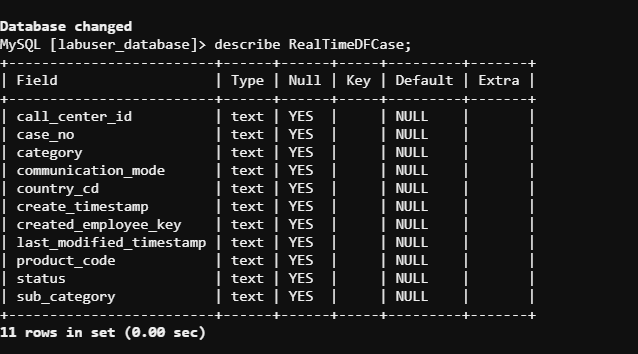
scala> RealTimeDFCase.printSchema

scala> RealTimeDFCase.show(4)

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

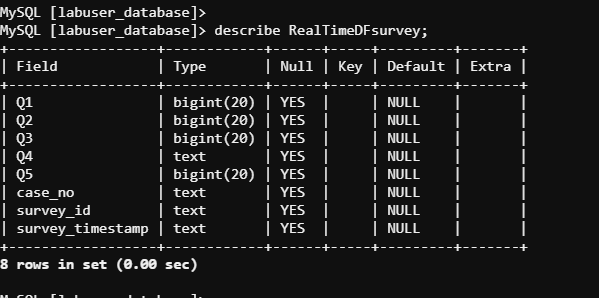
mysql -u edu\_labuser -h dbserver.edu.cloudlab.com -p

MySQL [labuser\_database]> describe RealTimeDFCase;



val RealTimeDFsurvey = spark.read.json("/user/edureka\_918210/project\_futurecart/realtimedata/survey")

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



**hive tables from mysql**

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

--username edu\_labuser \

--P \

--split-by survey\_id \

--columns Q1,Q2,Q3,Q4,Q5,case\_no,survey\_id,survey\_timestamp \

--table RealTimeDFsurvey \

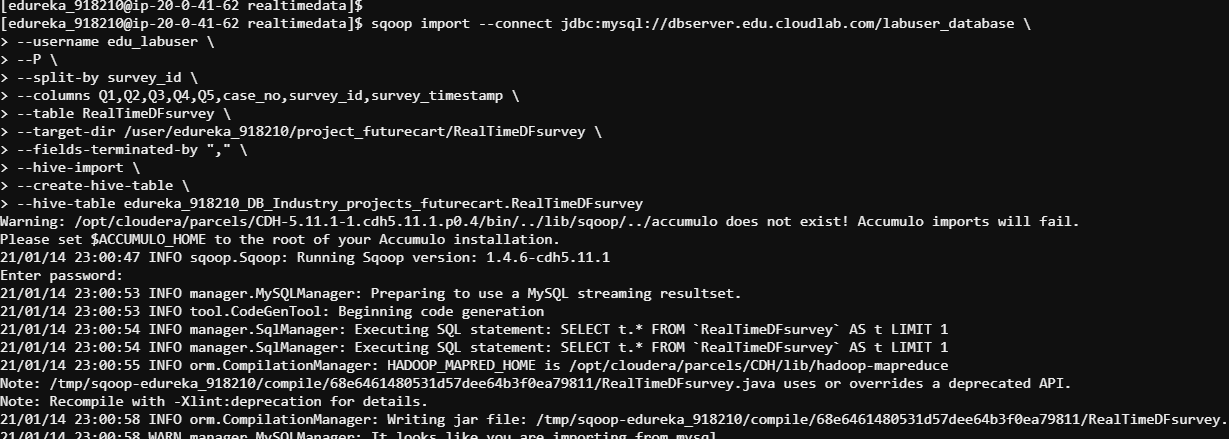
--target-dir /user/edureka\_918210/project\_futurecart/RealTimeDFsurvey \

--fields-terminated-by "," \

--hive-import \

--create-hive-table \

--hive-table edureka\_918210\_DB\_Industry\_projects\_futurecart.RealTimeDFsurvey



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

--username edu\_labuser \

--P \

--split-by call\_center\_id \

--columns call\_center\_id,case\_no,category,communication\_mode,country\_cd,create\_timestamp,created\_employee\_key,last\_modified\_timestamp,product\_code,status,sub\_category \

--table RealTimeDFCase \

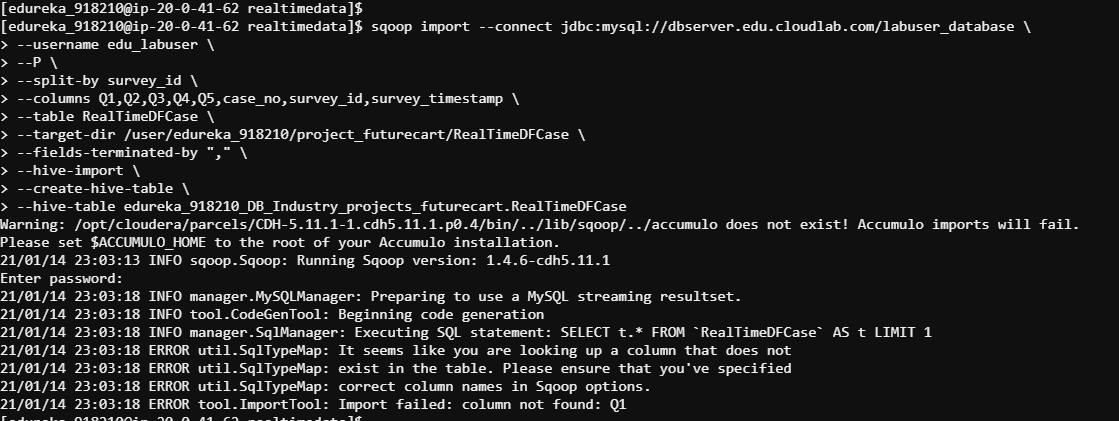
--target-dir /user/edureka\_918210/project\_futurecart/RealTimeDFCase \

--fields-terminated-by "," \

--hive-import \

--create-hive-table \

--hive-table edureka\_918210\_DB\_Industry\_projects\_futurecart.RealTimeDFCase

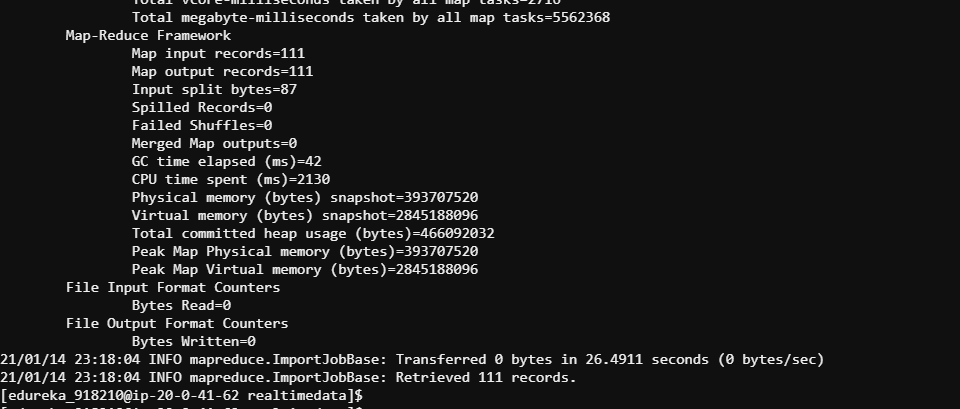


/user/edureka\_918210/project\_futurecart/RealTimeDFsurvey -- HDFS location

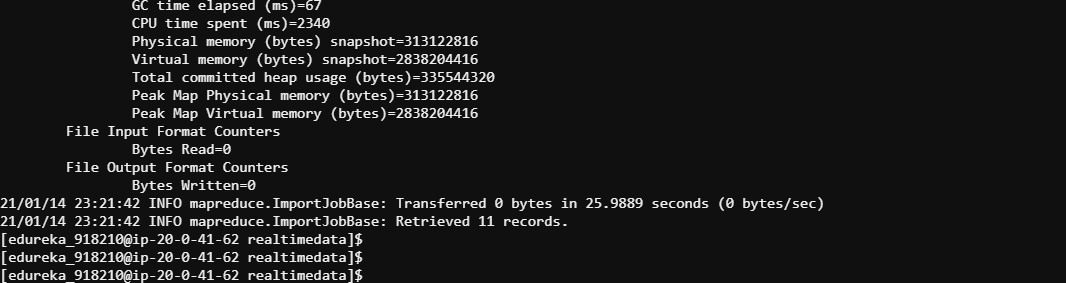
/user/edureka\_918210/project\_futurecart/RealTimeDFCase -- HDFS location

MYSQL TO HBase

sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table RealTimeDFCase --hbase-table ' RealTimeDFCase' --column-family cf2 --username edu\_labuser --hbase-create-table --columns call\_center\_id,case\_no,category,communication\_mode,country\_cd,create\_timestamp,created\_employee\_key,last\_modified\_timestamp,product\_code,status,sub\_category --hbase-row-key call\_center\_id -m 1 --password edureka



sqoop import --connect jdbc:mysql://dbserver.edu.cloudlab.com/labuser\_database --table RealTimeDFsurvey --hbase-table 'RealTimeDFsurvey' --column-family cf2 --username edu\_labuser --hbase-create-table --columns Q1,Q2,Q3,Q4,Q5,case\_no,survey\_id,survey\_timestamp --hbase-row-key survey\_id -m 1 --password edureka

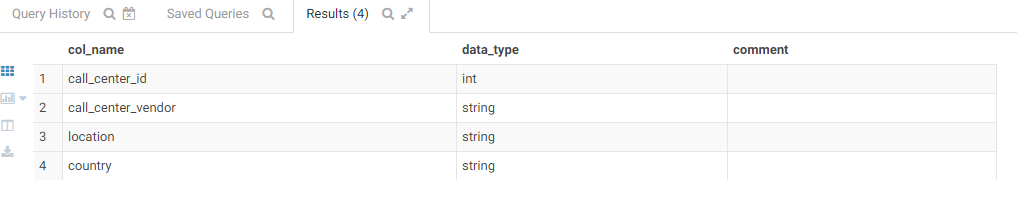


**Tables Descriptions**

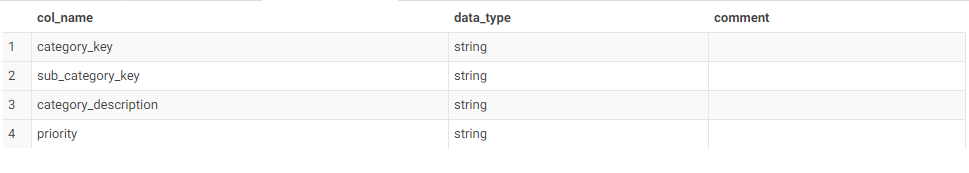
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_calendar\_details**



**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_call\_center\_details**



**describe edureka\_918210\_DB\_Industry\_projects\_futurecart. futurecart\_case\_category\_details**



**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_country\_details**



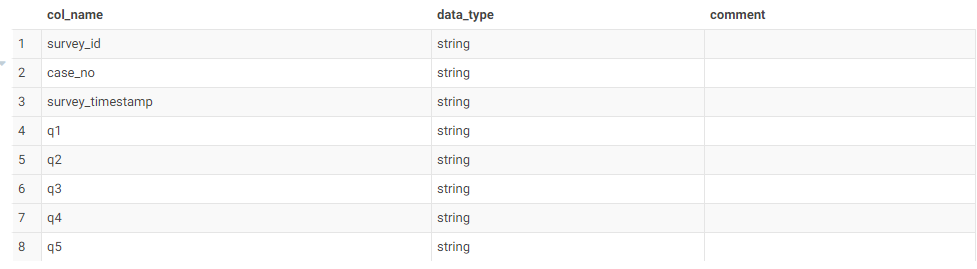
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_details**



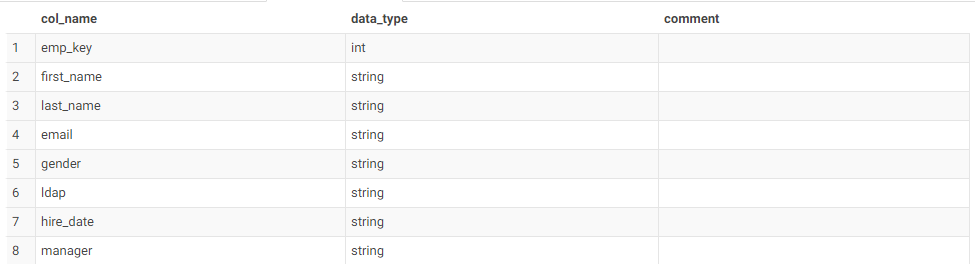
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_priority\_details**



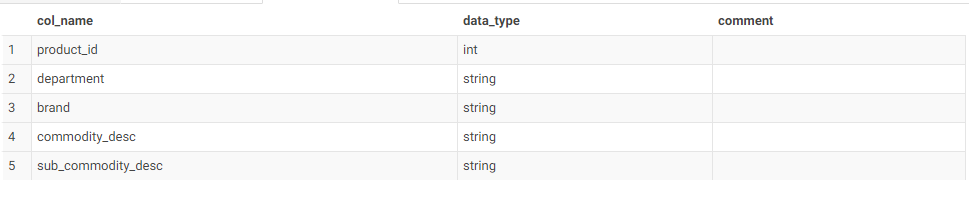
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_survey\_details**



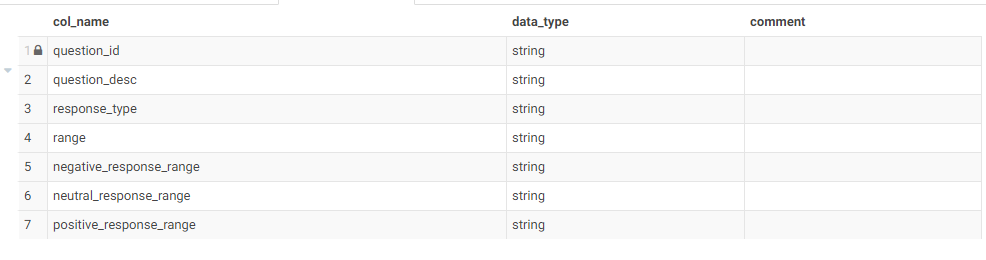
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_employee\_details**



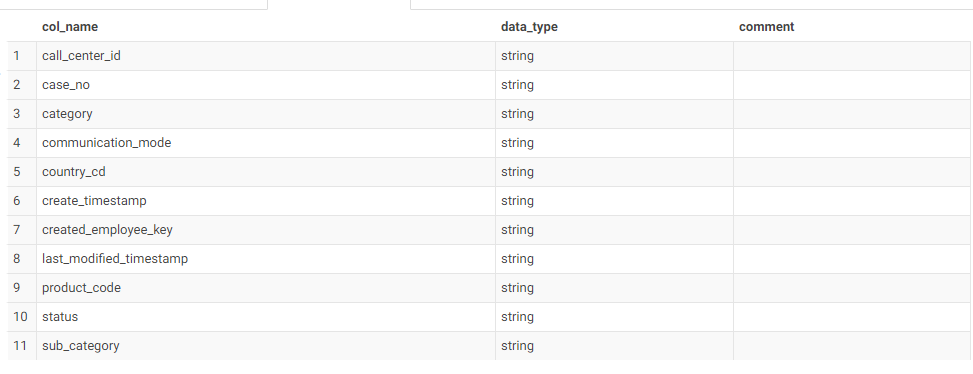
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_product\_details**



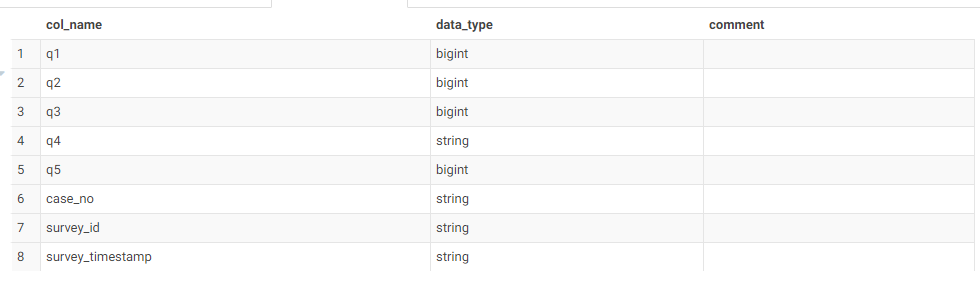
**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_survey\_question\_details**



**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.realtimedfcase**



**describe edureka\_918210\_DB\_Industry\_projects\_futurecart.realtimedfsurvey**



**PROJECT KPIs**

KPIs (Both on real-time data and batch-processed data)

• Total numbers of cases

• Total open cases in the last 1 hour

• Total closed cases in the last 1 hour

• Total priority cases 4

• Total positive/negative responses in the last 1 hour

• Total number of surveys in the last 1 hour

• Total open cases in a day/week/month

• Total closed cases in a day/week/month

• Total positive/negative responses in a day/week/month

• Total number of surveys in a day/week/month

Real-time KPIs

• Total numbers of cases that are open and closed out of the number of cases received

• Total number of cases received based on priority and severity

val futurecart\_calendar\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_calendar\_details")

val futurecart\_call\_center\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_call\_center\_details")

val futurecart\_case\_category\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_category\_details")

val futurecart\_case\_country\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_country\_details")

val futurecart\_case\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_details")

val futurecart\_case\_priority\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_priority\_details")

val futurecart\_case\_survey\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_survey\_details")

val futurecart\_employee\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_employee\_details")

val futurecart\_product\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_product\_details")

val futurecart\_survey\_question\_details = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_survey\_question\_details")

val RealTimeDFCase = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.RealTimeDFCase")

val RealTimeDFsurvey = spark.sql("select \* from edureka\_918210\_DB\_Industry\_projects\_futurecart.RealTimeDFsurvey")

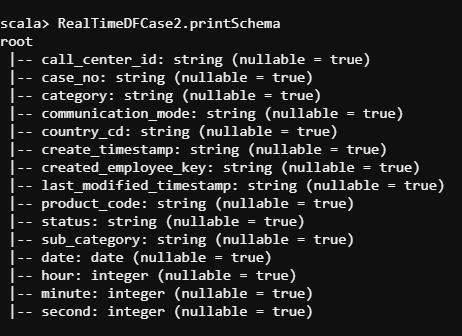
**Total numbers of cases**

select count(\*) from edureka\_918210\_DB\_Industry\_projects\_futurecart.futurecart\_case\_country\_details

193

**Total open cases in the last 1 hour**

val RealTimeDFCase2 = RealTimeDFCase.withColumn("date", to\_date($"last\_modified\_timestamp")).withColumn("hour", hour(col("last\_modified\_timestamp"))).withColumn("minute", minute(col("last\_modified\_timestamp"))).withColumn("second", second(col("last\_modified\_timestamp")))



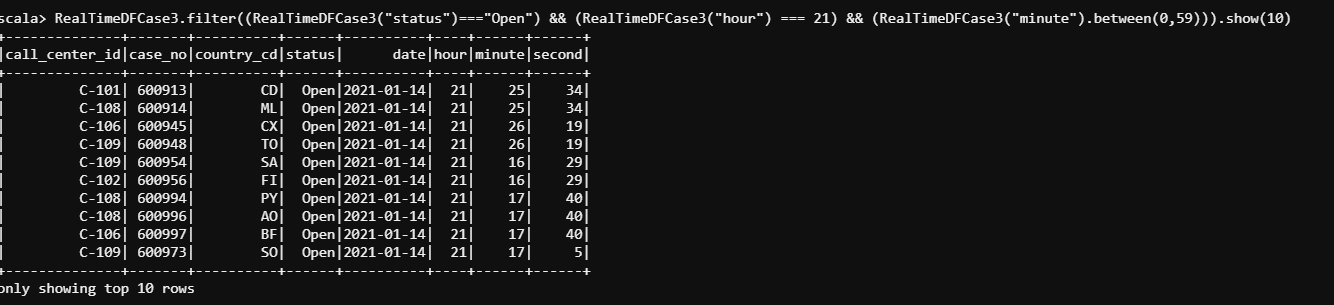
val RealTimeDFCase3 = RealTimeDFCase2.orderBy(desc("hour")).select("call\_center\_id","case\_no","country\_cd","status","date","hour","minute","second")

RealTimeDFCase3.filter(RealTimeDFCase3("status")==="Open").show(10)

scala> RealTimeDFCase3.filter((RealTimeDFCase3("status")==="Open") && (RealTimeDFCase3("hour") === 21) && (RealTimeDFCase3("minute").between(0,59))).count()

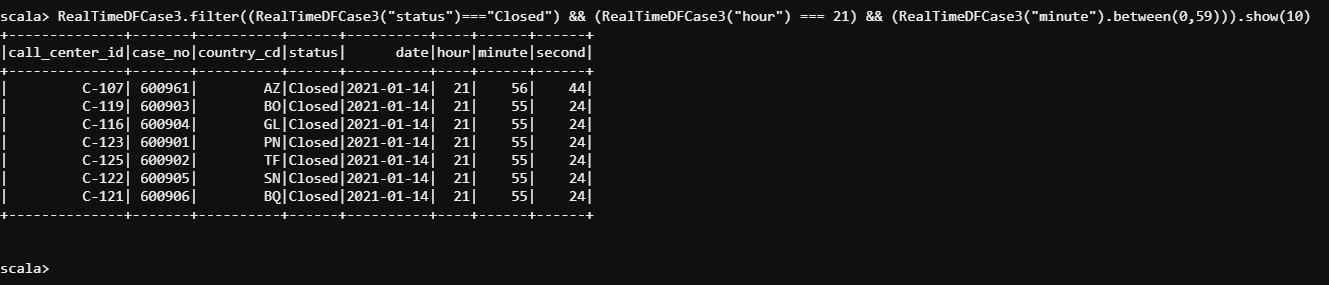
res1: Long = 100

RealTimeDFCase3.filter((RealTimeDFCase3("status")==="Open") && (RealTimeDFCase3("hour") === 21) && (RealTimeDFCase3("minute").between(0,59))).show(10)



**Total closed cases in the last 1 hour**

RealTimeDFCase3.filter((RealTimeDFCase3("status")==="Closed") && (RealTimeDFCase3("hour") === 21) && (RealTimeDFCase3("minute").between(0,59))).show(10)



scala> RealTimeDFCase3.filter((RealTimeDFCase3("status")==="Closed") && (RealTimeDFCase3("hour") === 21) && (RealTimeDFCase3("minute").between(0,59))).count()

res3: Long = 7

scala>

**Total priority cases 4**

scala> futurecart\_case\_priority\_details.filter((futurecart\_case\_priority\_details("priority\_key")==="P4")).count()

res6: Long = 1 scala>

**Total positive/negative responses in the last 1 hour**

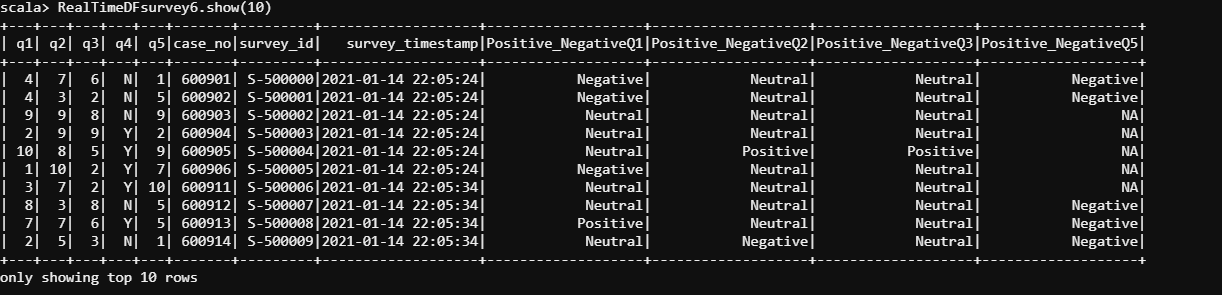
val RealTimeDFsurvey2 = RealTimeDFsurvey.withColumn("Positive\_NegativeQ1", when(col("q1").isin("1", "4"), "Negative").otherwise(when(col("q1").isin("5", "7"), "Positive").otherwise("Neutral")))

val RealTimeDFsurvey3 = RealTimeDFsurvey2.withColumn("Positive\_NegativeQ2", when(col("q2").isin("1", "5"), "Negative").otherwise(when(col("q2").isin("6", "8"), "Positive").otherwise("Neutral")))

val RealTimeDFsurvey4 = RealTimeDFsurvey3.withColumn("Positive\_NegativeQ3", when(col("q3").isin("1", "4"), "Negative").otherwise(when(col("q3").isin("5", "7"), "Positive").otherwise("Neutral")))

val RealTimeDFsurvey5 = RealTimeDFsurvey4.withColumn("Positive\_NegativeQ4", when(col("q4").isin("1", "4"), "NA").otherwise(when(col("q4").isin("5", "7"), "NA").otherwise("NA")))

val RealTimeDFsurvey6 = RealTimeDFsurvey4.withColumn("Positive\_NegativeQ5", when(col("q5").isin("1", "5"), "Negative").otherwise(when(col("q5").isin("6", "8"), "Positive").otherwise("NA")))



**Total number of surveys in the last 1 hour**

scala> RealTimeDFsurvey.count()res18: Long = 11 scala>

**Total opened cases in a day/week/month**

**=====================================**

scala> futurecart\_case\_details.printSchema

root

|-- case\_no: integer (nullable = true)

|-- create\_timestamp: string (nullable = true)

|-- last\_modified\_timestamp: string (nullable = true)

|-- created\_employee\_key: integer (nullable = true)

|-- call\_center\_id: string (nullable = true)

|-- status: string (nullable = true)

|-- category: string (nullable = true)

|-- sub\_category: string (nullable = true)

|-- communication\_mode: string (nullable = true)

|-- country\_cd: string (nullable = true)

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

scala> futurecart\_case\_details.count()

res4: Long = 808265

val futurecart\_case\_details2 = futurecart\_case\_details.withColumn("date", to\_date($"last\_modified\_timestamp")).withColumn("month", month(col("last\_modified\_timestamp"))).withColumn("day", day(col("last\_modified\_timestamp")))

scala> futurecart\_case\_details2.printSchema

root

|-- case\_no: integer (nullable = true)

|-- create\_timestamp: string (nullable = true)

|-- last\_modified\_timestamp: string (nullable = true)

|-- created\_employee\_key: integer (nullable = true)

|-- call\_center\_id: string (nullable = true)

|-- status: string (nullable = true)

|-- category: string (nullable = true)

|-- sub\_category: string (nullable = true)

|-- communication\_mode: string (nullable = true)

|-- country\_cd: string (nullable = true)

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

|-- date: date (nullable = true)

|-- month: integer (nullable = true)

|-- day: integer (nullable = true)

futurecart\_case\_details2.filter((futurecart\_case\_details2("status")==="Open").groupBy("month")

futurecart\_case\_details2.filter((futurecart\_case\_details2("status")==="Open").groupBy("day")

**Total closed cases in a day/week/month**

**=======================================**

scala> futurecart\_case\_details2.printSchema

root

|-- case\_no: integer (nullable = true)

|-- create\_timestamp: string (nullable = true)

|-- last\_modified\_timestamp: string (nullable = true)

|-- created\_employee\_key: integer (nullable = true)

|-- call\_center\_id: string (nullable = true)

|-- status: string (nullable = true)

|-- category: string (nullable = true)

|-- sub\_category: string (nullable = true)

|-- communication\_mode: string (nullable = true)

|-- country\_cd: string (nullable = true)

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

|-- date: date (nullable = true)

|-- month: integer (nullable = true)

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

futurecart\_case\_details2.filter((futurecart\_case\_details2("status")==="Closed").groupBy("month")

futurecart\_case\_details2.filter((futurecart\_case\_details2("status")==="Closed").groupBy("day")

**Total positive/negative responses in a day/week/month**

RealTimeDFsurvey6.filter(("Positive\_NegativeQ1")==="Positive").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ1")==="Negative").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ1")==="Positive").count().groupBy("day")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ1")==="Negative").count().groupBy("day")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ2")==="Positive").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ2")==="Negative").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ2")==="Positive").count().groupBy("day")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ2")==="Negative").count().groupBy("day")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ3")==="Positive").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ3")==="Negative").count().groupBy("month")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ3")==="Positive").count().groupBy("day")

RealTimeDFsurvey6.filter(("Positive\_NegativeQ3")==="Negative").count().groupBy("day")

**Total number of surveys in a day/week/month**

**============================================**

scala> futurecart\_case\_details2.printSchema

root

|-- case\_no: integer (nullable = true)

|-- create\_timestamp: string (nullable = true)

|-- last\_modified\_timestamp: string (nullable = true)

|-- created\_employee\_key: integer (nullable = true)

|-- call\_center\_id: string (nullable = true)

|-- status: string (nullable = true)

|-- category: string (nullable = true)

|-- sub\_category: string (nullable = true)

|-- communication\_mode: string (nullable = true)

|-- country\_cd: string (nullable = true)

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

|-- date: date (nullable = true)

|-- month: integer (nullable = true)

futurecart\_case\_details2.filter((futurecart\_case\_details2("case\_no").count()).groupBy("month")

futurecart\_case\_details2.filter((futurecart\_case\_details2("case\_no").count()).groupBy("day")