package com.sparkTutorial.sparkSql;

/\* 2.5 Process Filter and Spark

\* 2.2.5 - Optimizing the Process

\* Credit Card SystemReq-2.2.4Optimizing the ProcessFunctional Requirements

\* 1) Create "credit\_card\_number" directory inside the HDFS and store below information in it.

\* 2) Find which state has the highest number of branches

\* 3) Find which customers have the greatest number of transactions on their credit cards

\* 4) Find which states have the greatest number of credit card transactions

\* \*/

import static org.apache.spark.sql.functions.col;

import static org.apache.spark.sql.functions.desc;

import org.apache.log4j.Level;

import org.apache.log4j.Logger;

import org.apache.spark.SparkConf;

import org.apache.spark.api.java.JavaRDD;

import org.apache.spark.api.java.JavaSparkContext;

import org.apache.spark.sql.Dataset;

import org.apache.spark.sql.Encoders;

import org.apache.spark.sql.Row;

import org.apache.spark.sql.SparkSession;

public class case\_study {

public static void main(String[] args) {

Logger.getLogger("org").setLevel(Level.ERROR);

SparkConf conf = new SparkConf().setAppName("CaseStudy").setMaster("local[\*]");

JavaSparkContext sc = new JavaSparkContext(conf);

SparkSession session = SparkSession.builder().appName("Case\_Study").master("local[1]").getOrCreate();

//==========================================================================================================

JavaRDD<String> customer = sc.textFile("hdfs://192.168.73.129:8020/user/maria\_dev/Credit\_Card\_System/CDW\_SAPP\_CUSTOMER/");

JavaRDD<responsecust> response = customer.map(line -> {

String[] split = line.split("\t");

return new responsecust(split[0],split[1],split[2],split[3],split[4],split[5],split[6],split[7],split[8],split[9],split[10],split[11],

split[12],split[13]);

} ) ;

Dataset<responsecust> custDataset = session.createDataset(response.rdd(), Encoders.bean(responsecust.class));

//=====================================================================================================================

JavaRDD<String> branch = sc.textFile("hdfs://192.168.73.129:8020/user/maria\_dev/Credit\_Card\_System/CDW\_SAPP\_BRANCH/");

JavaRDD<responsebranch> response2 = branch.map(line -> {

String[] split = line.split("\t");

return new responsebranch(split[0],split[1],split[2],split[3],split[4],split[5],split[6],split[7]);

} ) ;

Dataset<responsebranch> branchDataset = session.createDataset(response2.rdd(), Encoders.bean(responsebranch.class));

System.out.println("=== 1.Find which state has the highest number of branches ===");

// 2) Find which state has the highest number of branches;

branchDataset.groupBy(col("BRANCH\_STATE")).count().sort(desc("count")).repartition(1).write().format("csv").option("header", true).save("hdfs://192.168.73.129:8020/user/maria\_dev/credit\_card\_number/CountByBranchState/");

//=======================================================================================================================

JavaRDD<String> creditcard = sc.textFile("hdfs://192.168.73.129:8020/user/maria\_dev/Credit\_Card\_System/CDW\_SAPP\_CREDITCARD/");

JavaRDD<responsecreditcard> response3 = creditcard.map(line -> {

String[] split = line.split("\t");

return new responsecreditcard(split[0],split[1],split[2],split[3],split[4],split[5],split[6],split[7],split[8]);

} ) ;

Dataset<responsecreditcard> creditDataset = session.createDataset(response3.rdd(), Encoders.bean(responsecreditcard.class));

// 3) Find which customers have the greatest number of transactions on their credit cards

System.out.println("=== 2.Find which customers have the greatest number of transactions on their credit cards ===");

creditDataset.groupBy(col("CUST\_SSN")).count().sort(desc("count")).repartition(1).write().format("csv").option("header", true).save("hdfs://192.168.73.129:8020/user/maria\_dev/credit\_card\_number/CountByCustomerSSN/");

// 4) Find which states have the greatest number of credit card transactions

Dataset<Row> joined = branchDataset.join(creditDataset,branchDataset.col("BRANCH\_CODE").startsWith(creditDataset.col("BRANCH\_CODE")), "inner");

System.out.println("=== 3.Find which states have the greatest number of credit card transactions ===");

joined.groupBy(col("BRANCH\_STATE")).count().sort(desc("count")).repartition(1).write().format("csv").option("header", true).save("hdfs://192.168.73.129:8020/user/maria\_dev/credit\_card\_number/CountByCeditcard/");

}

}