|  |
| --- |
| package assignment |
|  |  |
|  | import org.apache.spark.SparkContext |
|  | import org.apache.spark.SparkConf |
|  | import org.apache.spark.sql.SQLContext |
|  | import org.apache.spark.sql.SQLImplicits |
|  | import org.apache.spark.sql.functions.\_ |
|  |  |
|  | import org.apache.spark.sql.functions.udf |
|  |  |
|  | object BoradCastDemo { |
|  |  |
|  | def main(args: Array[String]): Unit = { |
|  | //specify the configuration for the spark application using instance of SparkConf |
|  | val config = new SparkConf().setAppName("Assignment 21.2").setMaster("local") |
|  |  |
|  | //setting the configuration and creating an instance of SparkContext |
|  | val sc = new SparkContext(config) |
|  |  |
|  | //Entry point of our sqlContext |
|  | val sqlContext = new SQLContext(sc) |
|  |  |
|  | //to use toDF method |
|  | import sqlContext.implicits.\_ |
|  |  |
|  | //create RDD for flights |
|  | val flights = sc.parallelize(List( |
|  | ("SEA", "JFK", "DL", "418", "7:00"), |
|  | ("SFO", "LAX", "AA", "1250", "7:05"), |
|  | ("SFO", "JFK", "VX", "12", "7:05"), |
|  | ("JFK", "LAX", "DL", "424", "7:10"), |
|  | ("LAX", "SEA", "DL", "5737", "7:10"))) |
|  |  |
|  | //create RDD for airports |
|  | val airports = sc.parallelize(List( |
|  | ("JFK", "John F. Kennedy International Airport", "New York", "NY"), |
|  | ("LAX", "Los Angeles International Airport", "Los Angeles", "CA"), |
|  | ("SEA", "Seattle-Tacoma International Airport", "Seattle", "WA"), |
|  | ("SFO", "San Francisco International Airport", "San Francisco", "CA"))) |
|  |  |
|  | //create RDD for airlines |
|  | val airlines = sc.parallelize(List( |
|  | ("AA", "American Airlines"), |
|  | ("DL", "Delta Airlines"), |
|  | ("VX", "Virgin America"))) |
|  |  |
|  | //create dataframe for flights |
|  | val flightsDF = flights.toDF("source\_code", "destination\_code", "airline\_code", "flight\_number", "flight\_time") |
|  |  |
|  | //create dataframe for airport |
|  | val airportsDF = airports.toDF("airport\_code", "airport\_name", "location\_name", "location\_code") |
|  |  |
|  | //create dataframe for airline |
|  | val airlinesDF = airlines.toDF("airline\_code", "airline\_name") |
|  |  |
|  | /\* |
|  | \* We cant creat broadcast variables for data frame , however to do broadcast joins we can use Broadcast function on dataframe to make it for use in broadcast joins |
|  | \* |
|  | \* Join flightsDF with airlinesDF to get dataframe with airline information |
|  | \*/ |
|  |  |
|  | val flight\_airline\_join = broadcast(flightsDF).join(airlinesDF, flightsDF("airline\_code") === airlinesDF("airline\_code"), "left") |
|  |  |
|  | /\* |
|  | \* Join flight\_airline\_join with airportsDF to get dataframe with source information |
|  | \*/ |
|  | val flight\_airline\_airport\_withSource\_join = flight\_airline\_join.join(airportsDF, flight\_airline\_join("source\_code") === airportsDF("airport\_code"), "left") |
|  |  |
|  | /\* |
|  | \* select required columns from flight\_airline\_airport\_withSource\_join and create rdd |
|  | \*/ |
|  | val flightWithSourceDF = flight\_airline\_airport\_withSource\_join.select($"source\_code", $"destination\_code", $"location\_name" as "source\_location", $"airline\_name", $"flight\_number", $"flight\_time") |
|  |  |
|  | /\* |
|  | \* Join flightWithSourceDF with airportsDF to get dataframe with destination information |
|  | \*/ |
|  |  |
|  | val flightWithDestDF = flightWithSourceDF.join(airportsDF, flightWithSourceDF("destination\_code") === airportsDF("airport\_code"), "left") |
|  |  |
|  | /\* |
|  | \* select required columns from flightWithDestDF and create rdd |
|  | \*/ |
|  | val fightDetails = flightWithDestDF.select($"source\_location", $"location\_name" as "destination\_location", $"airline\_name", $"flight\_number", $"flight\_time") |
|  |  |
|  | /\* |
|  | \* use orderBy action on dataframe to display the output as required |
|  | \*/ |
|  | fightDetails.orderBy($"flight\_time").show |
|  |  |
|  | } |
|  | } |

