Assignment - 28.1

Aviation data analysis

You can download the datasets from the following links:

Delayed_Flights.csv

There are 29 columns in this dataset. Some of them have been mentioned below:

• Year: 1987 – 2008

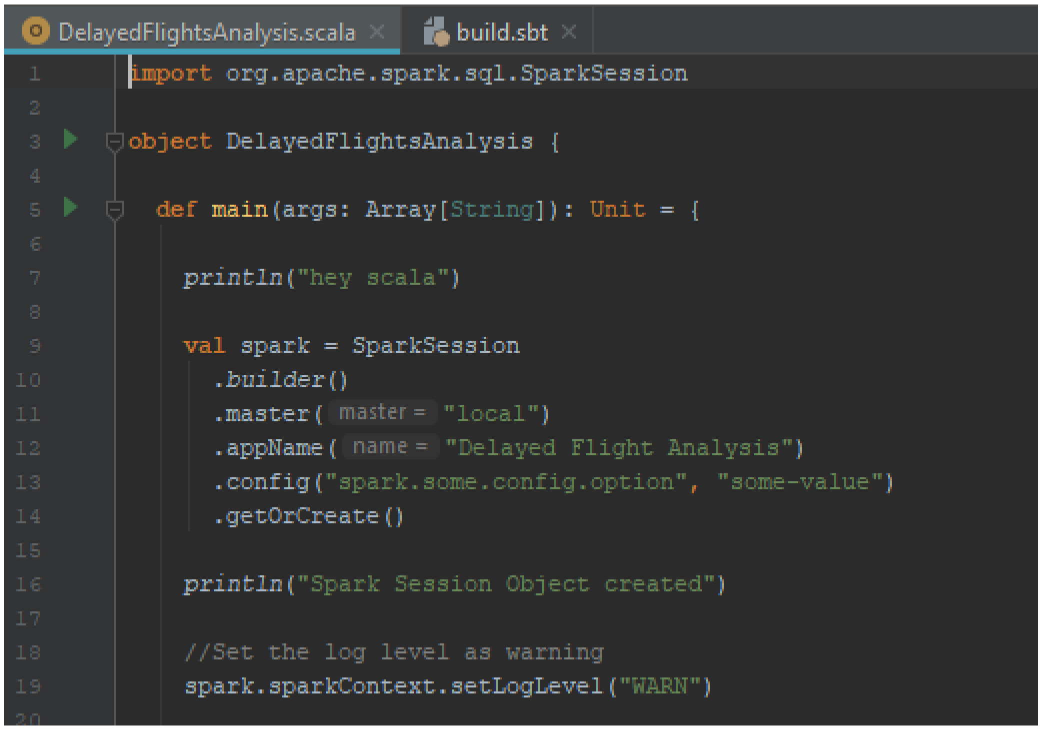
• Month: 1 – 12

• FlightNum: Flight number

• Canceled: Was the flight canceled?

• CancelleationCode: The reason for cancellation.

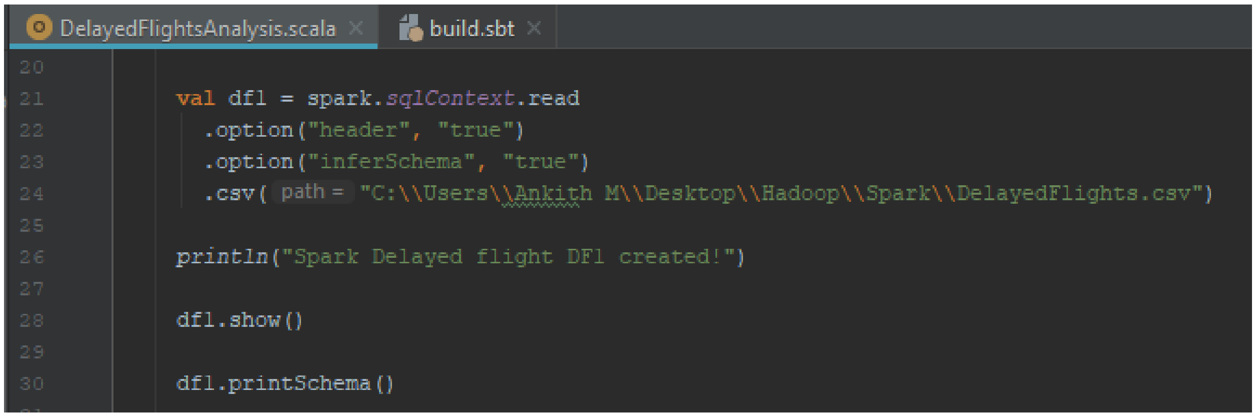
Now the very first thing is that we are going to implement this using Spark SQL. So as per requirement, we proceed to set up the Spark Context and load the input CSV file as shown below.



DelayedFlightsAnalysis.scala build.sbt

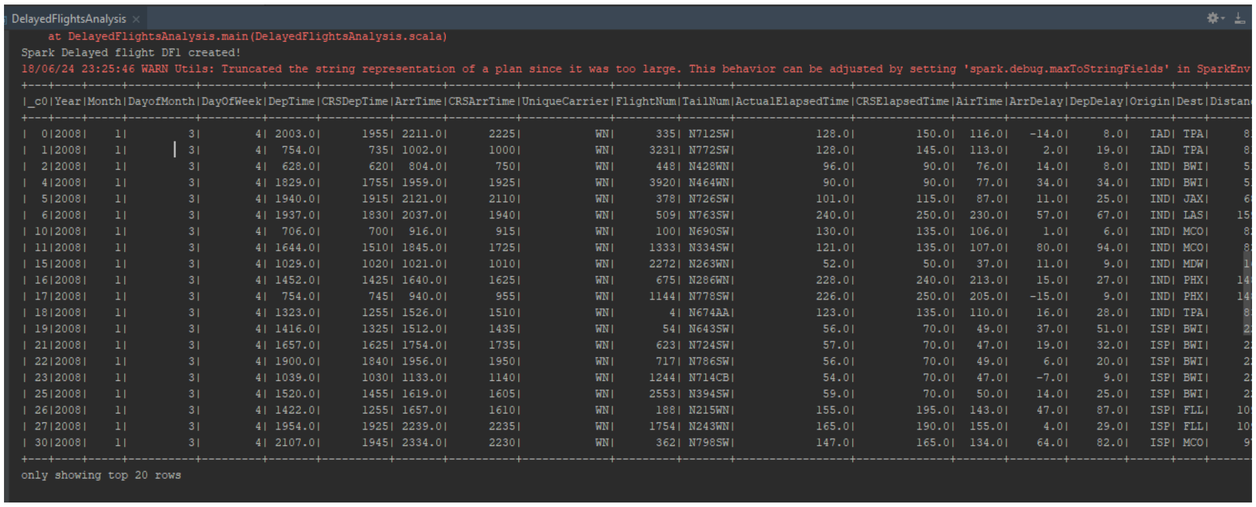
```
import org.apache.spark.sql.SparkSession

object DelayedFlightsAnalysis {

  def main(args: Array[String]): Unit = {

    println("hey scala")

    val spark = SparkSession
      .builder()
      .master( master = "local")
      .appName( name = "Delayed Flight Analysis")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")

    //Set the log level as warning
    spark.sparkContext.setLogLevel("WARN")
```

Now to load the file.

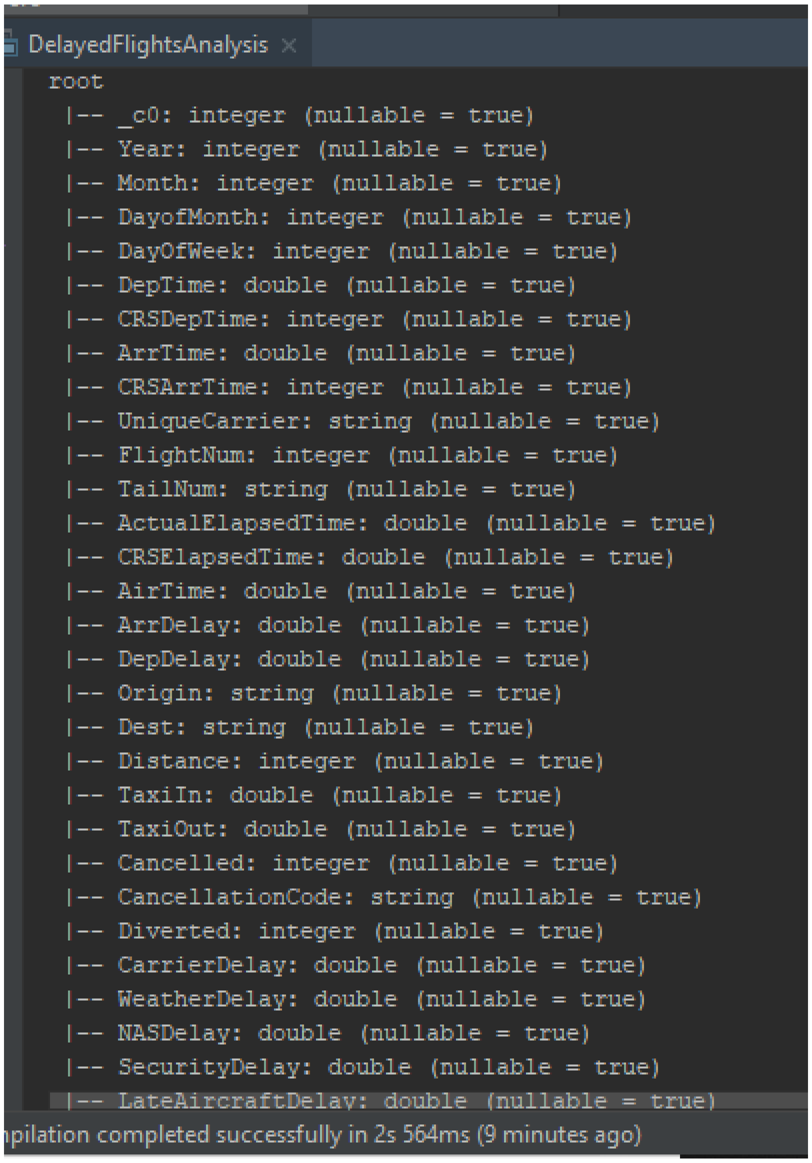


```scala
val df1 = spark.sqlContext.read
  .option("header", "true")
  .option("inferSchema", "true")
  .csv( path = "C:\\Users\\Ankith M\\Desktop\\Hadoop\\Spark\\DelayedFlights.csv")

println("Spark Delayed flight DF1 created!")

df1.show()

df1.printSchema()
```

Output of DataFrame created after reading the file and schema of the file.



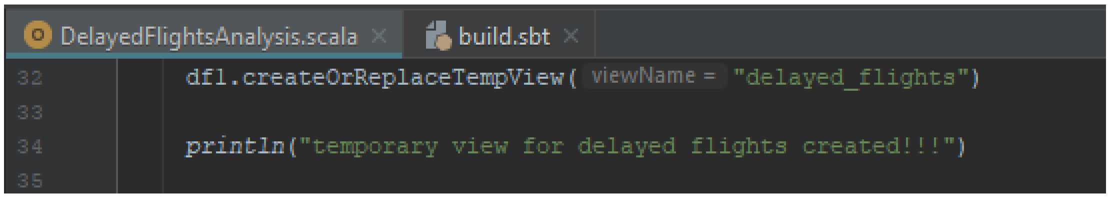
```
    at DelayedFlightsAnalysis.main(DelayedFlightsAnalysis.scala)
Spark Delayed flight DF1 created!
18/06/24 23:25:46 WARN Utils: Truncated the string representation of a plan since it was too large. This behavior can be adjusted by setting 'spark.debug.maxToStringFields' in SparkEnv
+---+----+-----+----------+---------+-------+----------+-------+----------+-------------+---------+-------+-----------------+--------------+-------+--------+--------+------+----+-------
|_c0|Year|Month|DayofMonth|DayOfWeek|DepTime|CRSDepTime|ArrTime|CRSArrTime|UniqueCarrier|FlightNum|TailNum|ActualElapsedTime|CRSElapsedTime|AirTime|ArrDelay|DepDelay|Origin|Dest|Distan
+---+----+-----+----------+---------+-------+----------+-------+----------+-------------+---------+-------+-----------------+--------------+-------+--------+--------+------+----+-------
|  0|2008|    1|         3|        4| 2003.0|      1955| 2211.0|      2225|           WN|      335| N712SW|            128.0|         150.0|  116.0|   -14.0|     8.0|   IAD| TPA|     8
|  1|2008|    1|         3|        4|  754.0|       735| 1002.0|      1000|           WN|     3231| N772SW|            128.0|         145.0|  113.0|     2.0|    19.0|   IAD| TPA|     8
|  2|2008|    1|         3|        4|  628.0|       620|  804.0|       750|           WN|      448| N428WN|             96.0|          90.0|   76.0|    14.0|     8.0|   IND| BWI|     5
|  4|2008|    1|         3|        4| 1829.0|      1755| 1959.0|      1925|           WN|     3920| N464WN|             90.0|          90.0|   77.0|    34.0|    34.0|   IND| BWI|     5
|  5|2008|    1|         3|        4| 1940.0|      1915| 2121.0|      2110|           WN|      378| N726SW|            101.0|         115.0|   87.0|    11.0|    25.0|   IND| JAX|     6
|  6|2008|    1|         3|        4| 1937.0|      1830| 2037.0|      1940|           WN|      509| N763SW|            240.0|         250.0|  230.0|    57.0|    67.0|   IND| LAS|    15
| 10|2008|    1|         3|        4|  706.0|       700|  916.0|       915|           WN|      100| N690SW|            130.0|         135.0|  106.0|     1.0|     6.0|   IND| MCO|     8
| 11|2008|    1|         3|        4| 1644.0|      1510| 1845.0|      1725|           WN|     1333| N334SW|            121.0|         135.0|  107.0|    80.0|    94.0|   IND| MCO|     8
| 15|2008|    1|         3|        4| 1029.0|      1020| 1021.0|      1010|           WN|     2272| N263WN|             52.0|          50.0|   37.0|    11.0|     9.0|   IND| MDW|     1
| 16|2008|    1|         3|        4| 1452.0|      1425| 1640.0|      1625|           WN|      675| N286WN|            228.0|         240.0|  213.0|    15.0|    27.0|   IND| PHX|    14
| 17|2008|    1|         3|        4|  754.0|       745|  940.0|       955|           WN|     1144| N778SW|            226.0|         250.0|  205.0|   -15.0|     9.0|   IND| PHX|    14
| 18|2008|    1|         3|        4| 1323.0|      1255| 1526.0|      1510|           WN|        4| N674AA|            123.0|         135.0|  110.0|    16.0|    28.0|   IND| TPA|     8
| 19|2008|    1|         3|        4| 1416.0|      1325| 1512.0|      1435|           WN|       54| N643SW|             56.0|          70.0|   49.0|    37.0|    51.0|   ISP| BWI|     2
| 21|2008|    1|         3|        4| 1657.0|      1625| 1754.0|      1735|           WN|      623| N724SW|             57.0|          70.0|   47.0|    19.0|    32.0|   ISP| BWI|     2
| 22|2008|    1|         3|        4| 1900.0|      1840| 1956.0|      1950|           WN|      717| N786SW|             56.0|          70.0|   49.0|     6.0|    20.0|   ISP| BWI|     2
| 23|2008|    1|         3|        4| 1039.0|      1030| 1133.0|      1140|           WN|     1244| N714CB|             54.0|          70.0|   47.0|    -7.0|     9.0|   ISP| BWI|     2
| 25|2008|    1|         3|        4| 1520.0|      1455| 1619.0|      1605|           WN|     2553| N394SW|             59.0|          70.0|   50.0|    14.0|    25.0|   ISP| BWI|     2
| 26|2008|    1|         3|        4| 1422.0|      1255| 1657.0|      1610|           WN|      188| N215WN|            155.0|         195.0|  143.0|    47.0|    87.0|   ISP| FLL|    10
| 27|2008|    1|         3|        4| 1954.0|      1925| 2239.0|      2235|           WN|     1754| N243WN|            165.0|         190.0|  155.0|     4.0|    29.0|   ISP| FLL|    10
| 30|2008|    1|         3|        4| 2107.0|      1945| 2334.0|      2230|           WN|      362| N798SW|            147.0|         165.0|  134.0|    64.0|    82.0|   ISP| MCO|     9
+---+----+-----+----------+---------+-------+----------+-------+----------+-------------+---------+-------+-----------------+--------------+-------+--------+--------+------+----+-------
only showing top 20 rows
```



```
root
 |-- _c0: integer (nullable = true)
 |-- Year: integer (nullable = true)
 |-- Month: integer (nullable = true)
 |-- DayofMonth: integer (nullable = true)
 |-- DayOfWeek: integer (nullable = true)
 |-- DepTime: double (nullable = true)
 |-- CRSDepTime: integer (nullable = true)
 |-- ArrTime: double (nullable = true)
 |-- CRSArrTime: integer (nullable = true)
 |-- UniqueCarrier: string (nullable = true)
 |-- FlightNum: integer (nullable = true)
 |-- TailNum: string (nullable = true)
 |-- ActualElapsedTime: double (nullable = true)
 |-- CRSElapsedTime: double (nullable = true)
 |-- AirTime: double (nullable = true)
 |-- ArrDelay: double (nullable = true)
 |-- DepDelay: double (nullable = true)
 |-- Origin: string (nullable = true)
 |-- Dest: string (nullable = true)
 |-- Distance: integer (nullable = true)
 |-- TaxiIn: double (nullable = true)
 |-- TaxiOut: double (nullable = true)
 |-- Cancelled: integer (nullable = true)
 |-- CancellationCode: string (nullable = true)
 |-- Diverted: integer (nullable = true)
 |-- CarrierDelay: double (nullable = true)
 |-- WeatherDelay: double (nullable = true)
 |-- NASDelay: double (nullable = true)
 |-- SecurityDelay: double (nullable = true)
 |-- LateAircraftDelay: double (nullable = true)
```

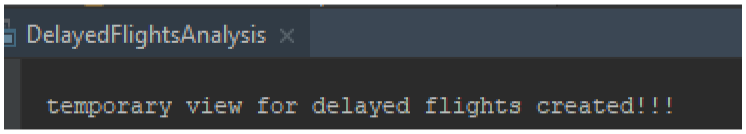
pilation completed successfully in 2s 564ms (9 minutes ago)

Now proceed to create a temporary view as below –



```scala
    dfl.createOrReplaceTempView( viewName = "delayed_flights")

    println("temporary view for delayed flights created!!!")
```

Output –

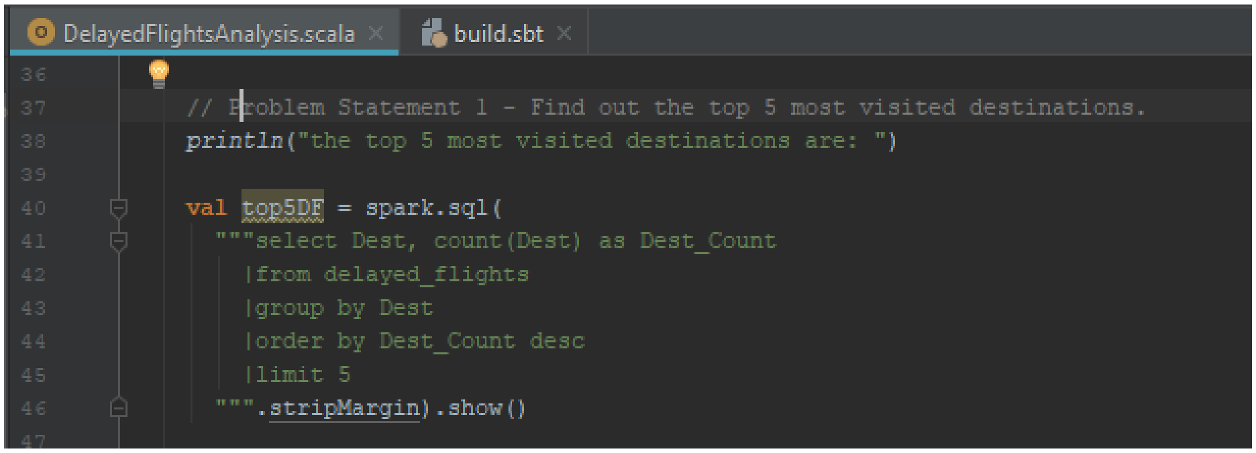


```
temporary view for delayed flights created!!!
```

Once the table is registered as view now we can proceed to use Spark SQL to meet each of the Problem Statements one by one.

Problem Statement 1 - Find out the top 5 most visited destinations.

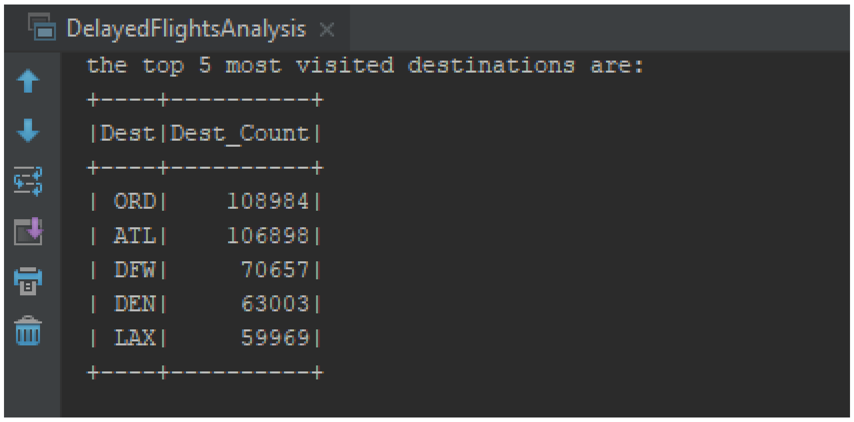
Answer:



DelayedFlightsAnalysis.scala   build.sbt

```
// Problem Statement 1 - Find out the top 5 most visited destinations.
println("the top 5 most visited destinations are: ")

val top5DF = spark.sql(
  """select Dest, count(Dest) as Dest_Count
    |from delayed_flights
    |group by Dest
    |order by Dest_Count desc
    |limit 5
  """.stripMargin).show()
```

Output:

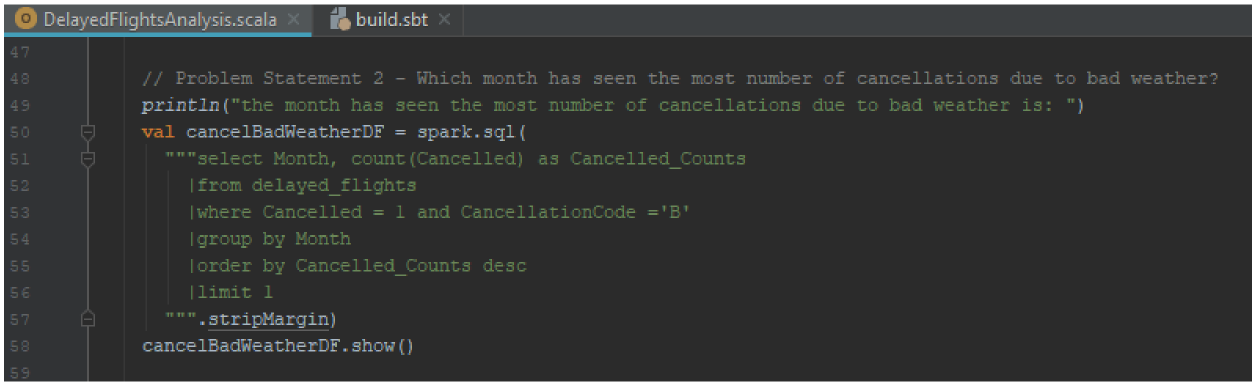


DelayedFlightsAnalysis

```
the top 5 most visited destinations are:
+----+----------+
|Dest|Dest_Count|
+----+----------+
| ORD|    108984|
| ATL|    106898|
| DFW|     70657|
| DEN|     63003|
| LAX|     59969|
+----+----------+
```

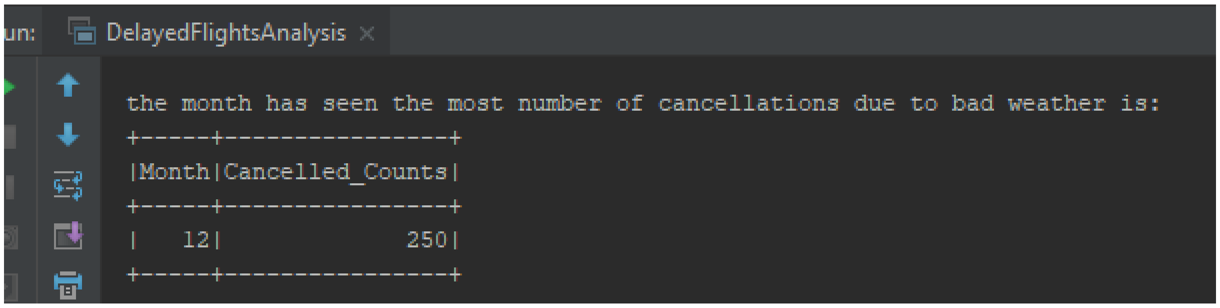
Problem Statement 2 - Which month has seen the most number of cancellations due to bad weather?

Answer:


DelayedFlightsAnalysis.scala build.sbt

```
    // Problem Statement 2 - Which month has seen the most number of cancellations due to bad weather?
    println("the month has seen the most number of cancellations due to bad weather is: ")
    val cancelBadWeatherDF = spark.sql(
      """select Month, count(Cancelled) as Cancelled_Counts
        |from delayed_flights
        |where Cancelled = 1 and CancellationCode ='B'
        |group by Month
        |order by Cancelled_Counts desc
        |limit 1
      """.stripMargin)
    cancelBadWeatherDF.show()
```

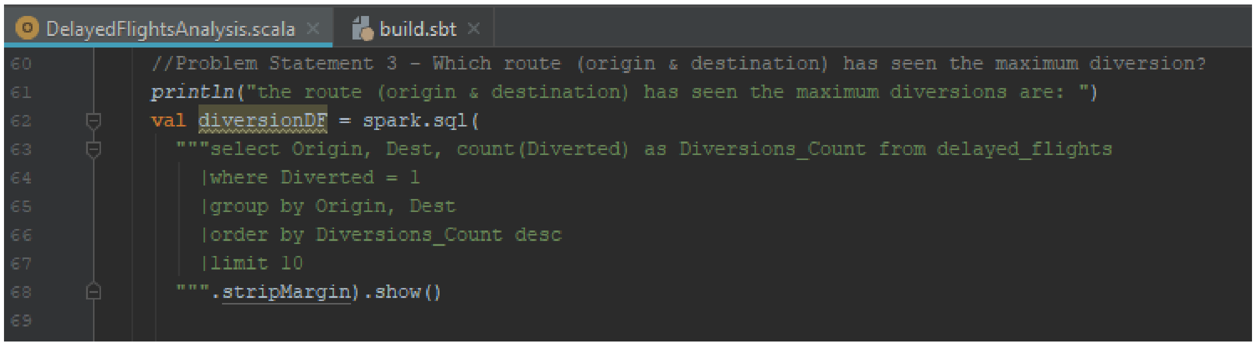
Output:


un: DelayedFlightsAnalysis

```
the month has seen the most number of cancellations due to bad weather is:
+-----+----------------+
|Month|Cancelled_Counts|
+-----+----------------+
|   12|             250|
+-----+----------------+
```

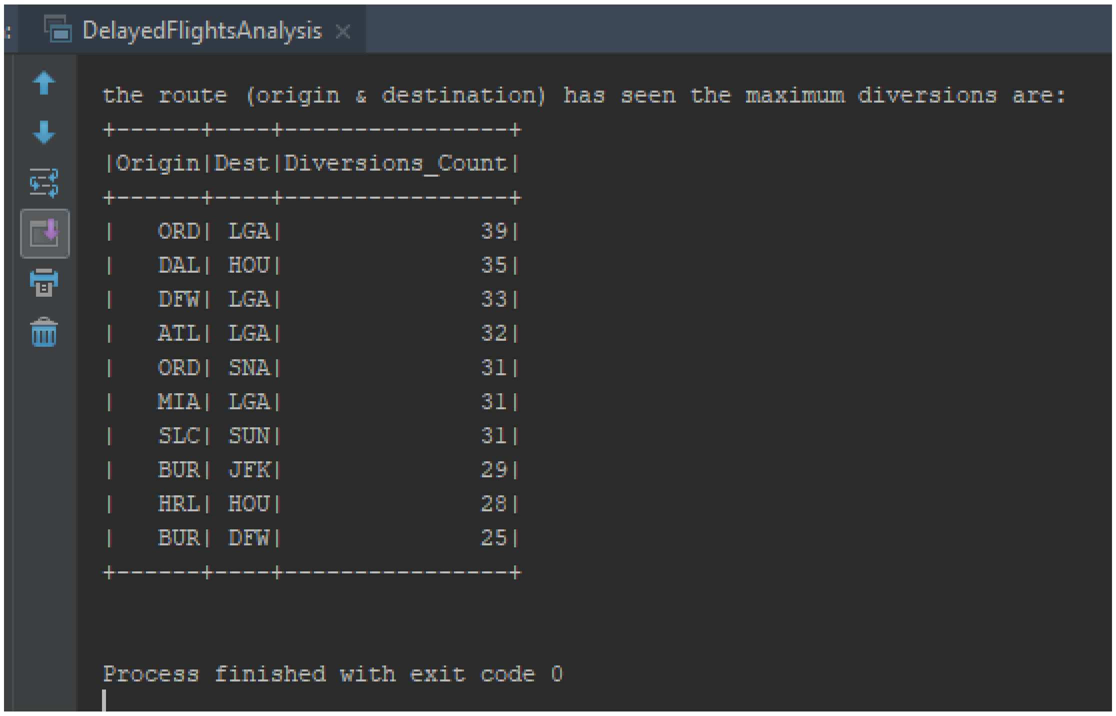
Problem Statement 3 - Which route (origin & destination) has seen the maximum diversion?

Answer:


DelayedFlightsAnalysis.scala build.sbt

```
    //Problem Statement 3 - Which route (origin & destination) has seen the maximum diversion?
    println("the route (origin & destination) has seen the maximum diversions are: ")
    val diversionDF = spark.sql(
      """select Origin, Dest, count(Diverted) as Diversions_Count from delayed_flights
        |where Diverted = 1
        |group by Origin, Dest
        |order by Diversions_Count desc
        |limit 10
      """.stripMargin).show()
```

Output –


DelayedFlightsAnalysis

```
the route (origin & destination) has seen the maximum diversions are:
+------+----+----------------+
|Origin|Dest|Diversions_Count|
+------+----+----------------+
|   ORD| LGA|              39|
|   DAL| HOU|              35|
|   DFW| LGA|              33|
|   ATL| LGA|              32|
|   ORD| SNA|              31|
|   MIA| LGA|              31|
|   SLC| SUN|              31|
|   BUR| JFK|              29|
|   HRL| HOU|              28|
|   BUR| DFW|              25|
+------+----+----------------+


Process finished with exit code 0
```

Please find below, the complete code for this use case as a whole.

```
import org.apache.spark.sql.SparkSession

object DelayedFlightsAnalysis {

  def main(args: Array[String]): Unit = {

    println("hey scala")

    val spark = SparkSession
      .builder()
      .master("local")
      .appName("Delayed Flight Analysis")
      .config("spark.some.config.option", "some-value")
      .getOrCreate()

    println("Spark Session Object created")

    //Set the log level as warning
    spark.sparkContext.setLogLevel("WARN")

    val df1 = spark.sqlContext.read
      .option("header", "true")
      .option("inferSchema", "true")
      .csv("C:\\Users\\Ankith M\\Desktop\\Hadoop\\Spark\\DelayedFlights.csv")

    println("Spark Delayed flight DF1 created!")

    df1.show()

    df1.printSchema()
    df1.createOrReplaceTempView("delayed_flights")

    println("temporary view for delayed flights created!!!")


    // Problem Statement 1 - Find out the top 5 most visited destinations.
    println("the top 5 most visited destinations are: ")

    val top5DF = spark.sql(
      """select Dest, count(Dest) as Dest_Count
        |from delayed_flights
        |group by Dest
        |order by Dest_Count desc
        |limit 5
      """.stripMargin).show()

    // Problem Statement 2 - Which month has seen the most number of cancellations due
to bad weather?
    println("the month has seen the most number of cancellations due to bad weather
is: ")
    val cancelBadWeatherDF = spark.sql(
      """select Month, count(Cancelled) as Cancelled_Counts
        |from delayed_flights
        |where Cancelled = 1 and CancellationCode ='B'
        |group by Month
        |order by Cancelled_Counts desc
        |limit 1
      """.stripMargin)
    cancelBadWeatherDF.show()

    //Problem Statement 3 - Which route (origin & destination) has seen the maximum
diversion?
    println("the route (origin & destination) has seen the maximum diversions are: ")
    val diversionDF = spark.sql(
      """select Origin, Dest, count(Diverted) as Diversions_Count from delayed_flights
        |where Diverted = 1
        |group by Origin, Dest
        |order by Diversions_Count desc
        |limit 10
      """.stripMargin).show()


  }
}
```