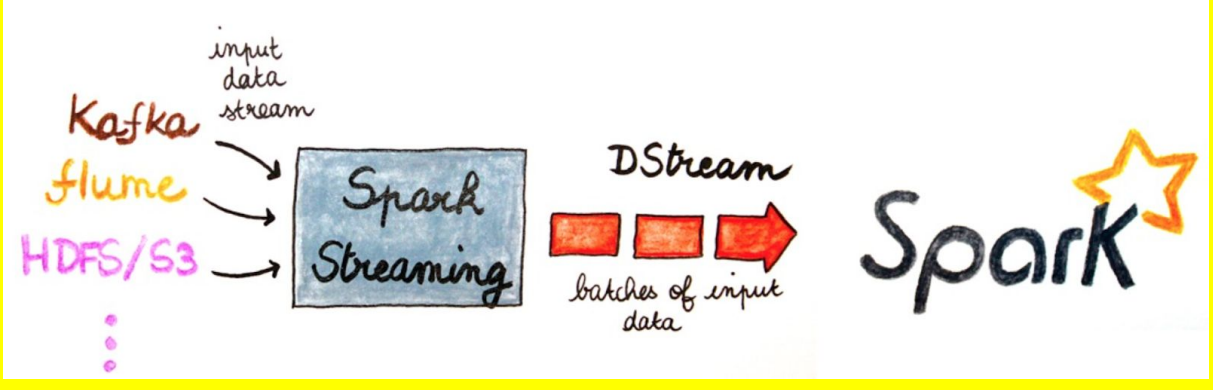
Spark Structured Streaming-Java Application: Read CSV file into Data Frame and Execute multiple queries in stream analytics With Spark Structured Streaming And Java

I. Introduction:

In this documentation, we are focused to parse data from a CSV file, perform some queries and output the result in the output using the **Spark Core** and **Spark Structured Streaming** based on **Spark SQL APIs**, and also **Java**.



Note: This Image is about Spark Streaming but in this project we will work with Spark Structured Streaming.

II. Technologies:

- Java 8
- Spark Core 2.4.7

- Spark Structured Streaming (in Spark SQL)2.4.7
- Spark SQL 2.4.7
- Maven
- Intelllj IDEA

III. Implement some queries using Java and Spark Structured Streaming:

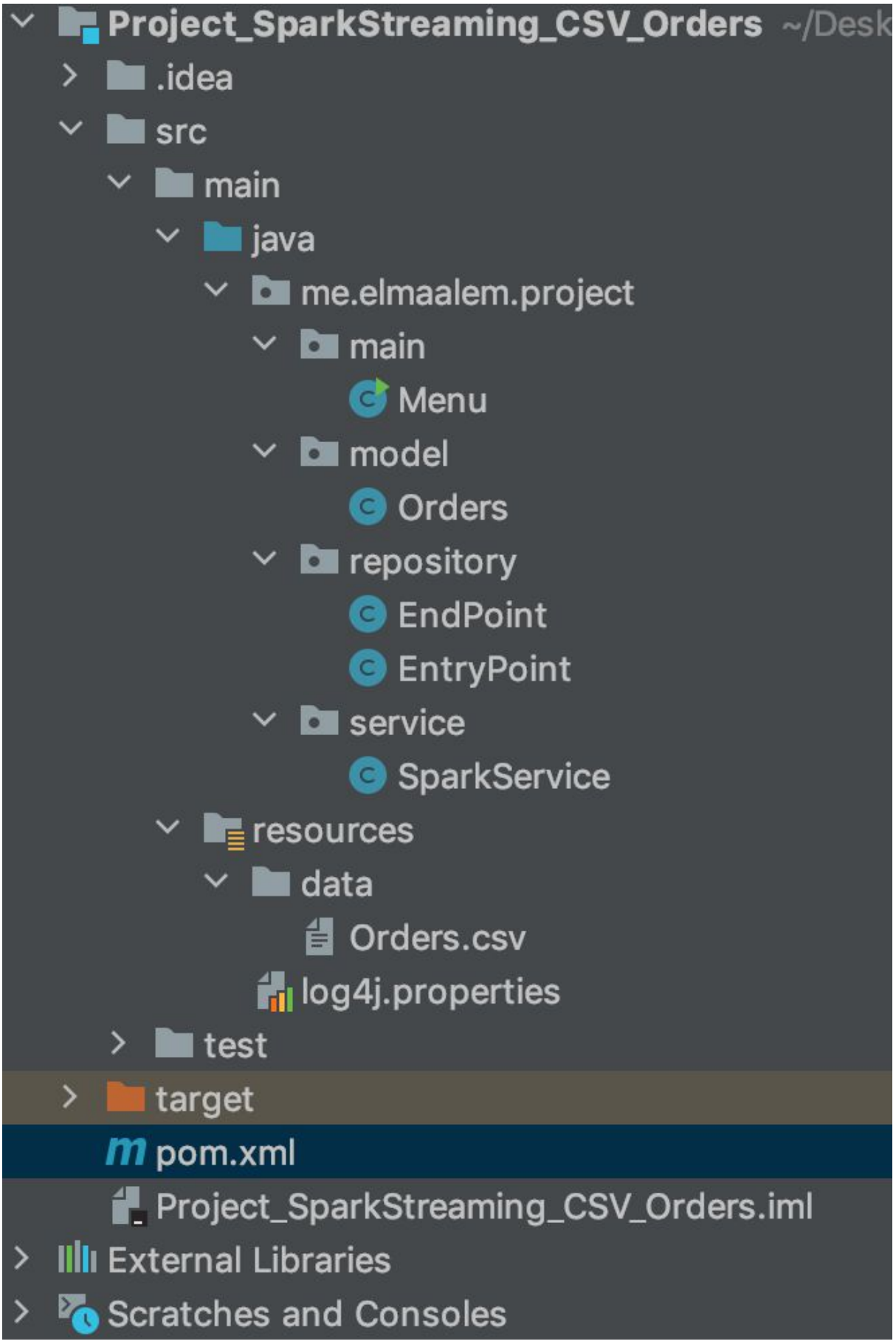
-Let's have a look at the **orders** dataset which we will use for this queries:

orderId	date	quantity	sales	mode	profit	unitPrice	customerName	customerSegment	productCategory
3	10-13-2010	6	261.54	Regular Air	-213.25	38.94	Muhammed MacIntyre	Small Business	Office Supplies
6	02-20-2012	2	6.93	Regular Air	-4.64	2.08	Ruben Dartt	Corporate	Office Supplies
32	07-15-2011	26	2808.08	Regular Air	1054.82	107.53	Liz Pelletier	Corporate	Furniture
32	07-15-2011	24	1761.4	Delivery Truck	-1748.56	70.89	Liz Pelletier	Corporate	Furniture
32	07-15-2011	23	160.2335	Regular Air	-85.129	7.99	Liz Pelletier	Corporate	Technology
32	07-15-2011	15	140.56	Regular Air	-128.38	8.46	Liz Pelletier	Corporate	Technology
35	10-22-2011	30	288.56	Regular Air	60.72	9.11	Julie Creighton	Corporate	Office Supplies
35	10-22-2011	14	1892.848	Regular Air	48.987	155.99	Julie Creighton	Corporate	Technology
36	11-02-2011	46	2484.7455	Regular Air	657.477	65.99	Sample Company A	Home Office	Technology
65	03-17-2011	32	3812.73	Regular Air	1470.3	115.79	Tamara Dahlen	Corporate	Technology
32	07-15-2008	26	<null>	Regular Air	<null>	107.53	Liz Pelletier	Corporate	Furniture
32	07-15-2008	24	<null>	Delivery Truck	<null>	70.89	Liz Pelletier	Corporate	Furniture
32	07-15-2008	23	<null>	Regular Air	<null>	7.99	Liz Pelletier	Corporate	Technology
32	07-15-2008	15	<null>	Regular Air	<null>	8.46	Liz Pelletier	Corporate	Technology
35	10-22-2008	30	<null>	Regular Air	<null>	9.11	Julie Creighton	Corporate	Office Supplies
35	10-22-2008	14	<null>	Regular Air	<null>	155.99	Julie Creighton	Corporate	Technology
36	10-22-2008	46	<null>	Regular Air	<null>	65.99	Sample Company A	Home Office	Technology
65	10-22-2008	32	<null>	Regular Air	<null>	115.79	Tamara Dahlen	Corporate	Technology
66	01-19-2009	41	108.15	Regular Air	7.57	2.88	Arthur Gainer	Consumer	Office Supplies
69	06-03-2009	42	1186.06	Regular Air	511.69	30.93	Jonathan Doherty	Corporate	Furniture
69	06-03-2009	28	51.53	Express Air	0.35	1.68	Jonathan Doherty	Corporate	Office Supplies
70	12-17-2010	48	90.05	Regular Air	-107	1.86	Helen Wasserman	Home Office	Office Supplies
70	12-17-2010	46	7804.53	Regular Air	2057.166	205.99	Helen Wasserman	Home Office	Technology
96	04-16-2009	37	4158.1235	Regular Air	1228.887	125.99	Keith Dawkins	Home Office	Technology
97	01-28-2010	26	75.57	Regular Air	28.24	2.89	Craig Yedwab	Consumer	Office Supplies
129	11-18-2012	4	32.72	Regular Air	-22.59	6.48	Pauline Chand	Corporate	Office Supplies

-These are **queries** to be exported:

- Get all Orders from csv file
- Get Orders By Customer Name
- Get Orders(Name,Sales,Profit,Category) By Customer Name and Order Date
- Get Orders By Product Category When Profit must be greater than 0 And Sorted by Customer Name
- Get Orders By Product Category In period and sorted by Sales

IV. Project Structure :


Project_SparkStreaming_CSV_Orders ~/Desk
.idea
src
main
java
me.elmaalem.project
main
Menu
model
Orders
repository
EndPoint
EntryPoint
service
SparkService
resources
data
Orders.csv
log4j.properties
test
target
pom.xml
Project_SparkStreaming_CSV_Orders.iml
External Libraries
Scratches and Consoles

V. Setup Dependencies on pom.xml:

After Adding below dependencies on **pom.xml**,It will download all the required packages.

```
      xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>me.elmaalem</groupId>
  <artifactId>Project_SparkStreaming_CSV_Orders</artifactId>
  <version>1.0-SNAPSHOT</version>
  <build>
      <plugins>
          <plugin>
              <groupId>org.apache.maven.plugins</groupId>
              <artifactId>maven-compiler-plugin</artifactId>
              <configuration>
                  <source>8</source>
                  <target>8</target>
              </configuration>
          </plugin>
      </plugins>
  </build>

  <dependencies>
      <!-- Dependency of Apache Spark Core-->
      <dependency>
          <groupId>org.apache.spark</groupId>
          <artifactId>spark-core_2.11</artifactId>
          <version>2.4.7</version>
      </dependency>
      <!-- Dependency of Apache Spark SQL (Spark Structured Streaming include in Spark SQL)-->
      <dependency>
          <groupId>org.apache.spark</groupId>
          <artifactId>spark-sql_2.11</artifactId>
          <version>2.4.7</version>
      </dependency>
  </dependencies>
```

VI. Configure Log4j file on spark console :

We would like to stop various **INFO messages** that are coming on the spark console to get just the result on the console without logging messages.

```
21/01/24 00:05:57 INFO TaskSetManager: Finished task 0.0 in stage 0.0 (TID 0) in 430 ms on localhost (executor driver) (1/1)
21/01/24 00:05:57 INFO TaskSchedulerImpl: Removed TaskSet 0.0, whose tasks have all completed, from pool
21/01/24 00:05:57 INFO DAGScheduler: ResultStage 0 (show at Service.java:14) finished in 0.669 s
21/01/24 00:05:57 INFO DAGScheduler: Job 0 finished: show at Service.java:14, took 0.729501 s
+-------+----------+--------+---------+-------------+------------------+---------+----------------+---------------+---------------+
|orderId|      date|quantity|    sales|         mode|            profit|unitPrice|    customerName|customerSegment|productCategory|
+-------+----------+--------+---------+-------------+------------------+---------+----------------+---------------+---------------+
```

We create a new file **log4j.properties** in order to stop these messages.Here are the contents of **log4j.properties:**

```
#Stop INFO messages displaying on Spark console to get just the result expected
log4j.rootCategory=ERROR, console
log4j.appender.console=org.apache.log4j.ConsoleAppender
log4j.appender.console.target=System.err
log4j.appender.console.layout=org.apache.log4j.PatternLayout
log4j.appender.console.layout.ConversionPattern=%d{yy/MM/dd HH:mm:ss} %p %c{1}: %m%n
```

VII. Define Data Model:

In the **model** package, we define **Orders** class.
model/Orders.class:

```
public class Orders implements Serializable {

    private int orderId;
    private LocalDate date;
    private int quantity;
    private Optional<Double> sales;
    private String mode;
    private Optional<Double> profit;
    private double unitPrice;
    private String customerName;
    private String customerSegment;
    private String productCategory;
    private static final DateTimeFormatter FORMATTER = DateTimeFormatter.ofPattern("MM-dd-yyyy");

    public Orders(){}
    public Orders(int orderId, String date, int quantity, double sales, String mode, double profit, double unitPrice,
                  String customerName, String customerSegment, String productCategory) {...}

    public String getProductCategory() { return productCategory; }

    public void setProductCategory(String productCategory) { this.productCategory = productCategory; }

    public int getOrderId() { return orderId; }

    public void setOrderId(int orderId) { this.orderId = orderId; }

    public LocalDate getDate() { return date; }

    public void setDate(String date) { this.date = LocalDate.parse(date,FORMATTER); }

    public int getQuantity() { return quantity; }

    public void setQuantity(int quantity) { this.quantity = quantity; }

    public Optional<Double> getSales() { return sales; }
```

VIII. Create a Repository to working with Dataframe(Orders.csv):

Let's create a repository to interact with **Orders** from the csv file.
In the **repository** package, we create two classes. The first one is **EntryPoint** which is responsible for reading streaming data and loading them into a **spark dataframe** with a custom schema.

```java
public class EntryPoint {

    public EntryPoint() { }

    private static SparkSession sparkSession(){
        return SparkSession
                .builder()
                .appName(" Application with Spark Streaming and Java")
                .master("local[*]")
                .getOrCreate();
    }

    private static StructType customSchema(){
        return new StructType(new StructField[] {
                new StructField("orderId", DataTypes.IntegerType, true,
Metadata.empty()),
                new StructField("date", DataTypes.DateType, true,
Metadata.empty()),
                new StructField("quantity", DataTypes.IntegerType, true,
Metadata.empty()),
                new StructField("sales", DataTypes.DoubleType, true,
Metadata.empty()),
                new StructField("mode", DataTypes.StringType, true,
Metadata.empty()),
                new StructField("profit", DataTypes.DoubleType, true,
Metadata.empty()),
                new StructField("unitPrice", DataTypes.DoubleType, true,
Metadata.empty()),
                new StructField("customerName", DataTypes.StringType, true,
Metadata.empty()),
                new StructField("customerSegment", DataTypes.StringType,
true, Metadata.empty()),
                new StructField("productCategory", DataTypes.StringType,
true, Metadata.empty())
        });
    }

    public static Dataset<Orders> getDataset(){
        Encoder<Orders> orderEncoder = Encoders.bean(Orders.class);

        return sparkSession()
                .readStream()
                .option("header","true")
                .option("treatEmptyValuesAsNulls", "true")
                .option("dateFormat", "MM-dd-yyyy")
                .option("delimiter",",")
                .schema(customSchema())
                .csv("src/main/resources/data/Orders*.csv")
                .as(orderEncoder);
    }
}
```

However, the second one is **EndPoint** class which is responsible for saving the content of the streaming Dataset out into the console using two output modes (“**Append**”, “**Complete**”). We are using **Complete** mode in the **last two queries** because the **Append** mode doesn't support **sorting** operations and **aggregation**.

```
public class EndPoint {

   public EndPoint(){}

   public static void displayDatasetWithOrders(Dataset<Orders> dataset, int
numberRows) throws StreamingQueryException {
       dataset.writeStream()
               .format("console")
               .outputMode("append")
               .option("numRows",numberRows)
               .start()
               .awaitTermination();
   }

   public static void displayDatasetWithRows(Dataset<Row> dataset, int
numberRows , String outputMode) throws StreamingQueryException {
       dataset.writeStream()
               .format("console")
               .outputMode(outputMode)
               .option("numRows",numberRows)
               .start()
               .awaitTermination();
   }
}
```

IX. Create a Spark Service:

SparkService class uses **EntryPoint and EndPoint** classes for **5** functions:

→ **listOrders(int numberRows)**: Get all Orders from csv file
→ **listOrdersMatchCustomerName(String customerName)**: Get Orders By Customer Name
→ **listOrdersMatchCustomerNameAndOrderDate(String customerName, String orderDate)**: Get Orders(Name,Sales,Profit,Category) By Customer Name and Order Date
→ **listOrdersMatchCategoryAndProfitPositiveAndSortByCustomerName(String productCategory):** Get Orders By Product Category When Profit must be greater than 0 And Sorted by Customer Name

→ **listOrdersMatchCategoryAndPeriodDateAndSortBySales(String productCategory, String startDate, String endDate):** Get Orders By Product Category In period and sorted by Sales

Here is the code of **service/SparkService.java:**

```
public class SparkService {

   public void listOrders(int numberRows) throws StreamingQueryException
{
        displayDatasetWithOrders(getDataset(),numberRows);
   }

   public void listOrdersMatchCustomerName(String customerName) throws
StreamingQueryException {

        Dataset<Orders> orders =
getDataset().filter((FilterFunction<Orders>) order ->
order.getCustomerName().equals(customerName));

        displayDatasetWithOrders(orders,1008);

   }

   public void listOrdersMatchCustomerNameAndOrderDate(String
customerName, String orderDate) throws StreamingQueryException {
        Dataset<Row> orders = getDataset()
                .filter((FilterFunction<Orders>) order ->
order.getCustomerName().equals(customerName))

.select("customerName","date","sales","profit","productCategory")
                .where("date == \"" + orderDate + "\"");

        displayDatasetWithRows(orders,1008, "append");
   }

   public void
listOrdersMatchCategoryAndProfitPositiveAndSortByCustomerName(String
productCategory) throws StreamingQueryException {
        Dataset<Row> orders = getDataset()
                .filter((FilterFunction<Orders>) order ->
order.getProductCategory().equals(productCategory))
                .filter("profit > 0.0")
                .groupBy("customerName","date","sales","profit")
                .count()
                .sort("customerName");

        displayDatasetWithRows(orders,1008, "complete");
   }

   public void listOrdersMatchCategoryAndPeriodDateAndSortBySales(String
productCategory, String startDate, String endDate) throws
```

```
StreamingQueryException {
        Dataset<Row> orders = getDataset()
                .filter((FilterFunction<Orders>) order ->
order.getProductCategory().equals(productCategory))
                .where("date < \""+ endDate +"\" and date > \""+ startDate
+"\"")

.groupBy("customerName","date","sales","quantity","profit","unitPrice","c
ustomerSegment")
                .count()
                .sort("customerName","date");

        displayDatasetWithRows(orders,1008, "complete");
    }
}
```

X. Creating a Menu Driven Program :

Let's create a **Menu** class under package **Main** to obtain input from a user by displaying a list of options.

main/Menu.java:

```java
public class Menu {

    public static Scanner scanner = new Scanner(System.in);
    public static SparkService sparkService = new SparkService();

    public static void main(String[] args) {

        try {
            int menuOption = 0;
            String customerName;
            String productCategory;
            String orderDate;
            String startDate;
            String endDate;

            do {
                // Setting menuOption equal to return value from showMenu();
                menuOption = showMenu();

                switch (menuOption) {...}

            } while (menuOption != 6);

            // Exiting message when user decides to quit Program
            System.out.println("Thanks for using this Program...");

        } catch (Exception ex) {
            System.out.println("Sorry problem occured within Program");
            scanner.next();
        } finally {
            scanner.close();
        }
    }
    public static int showMenu() {...}
}
```

XI. Output:

While executing each query , you will be able to see below its content in the console.

1. Menu:

```
Menu:
1. Get All Orders form CSV file
2. Get Orders By Customer Name
3. Get Orders(Name,Sales,Profit,Category)  By Customer Name and Order Date
4. Get Orders By Product Category When Profit must be greater than 0 And Sorted by Customer Name
5. Get Orders By Product Category In period and sorted by Sales
6. Quit Program
Enter the number of Query from above...
```

2. First Query:

```
6. Quit Program
Enter the number of Query from above...
1
Enter the rows number of Orders that you want show :
15
-----------------------------------------
Batch: 0
-----------------------------------------
+-------+----------+--------+---------+--------------+--------+---------+-----------------+--------------+---------------+
|orderId|      date|quantity|    sales|          mode|  profit|unitPrice|     customerName|customerSegment|productCategory|
+-------+----------+--------+---------+--------------+--------+---------+-----------------+--------------+---------------+
|      3|2010-10-13|       6|   261.54|   Regular Air| -213.25|    38.94|Muhammed MacIntyre| Small Business|Office Supplies|
|      6|2012-02-20|       2|     6.93|   Regular Air|   -4.64|     2.08|       Ruben Dartt|      Corporate|Office Supplies|
|     32|2011-07-15|      26|  2808.08|   Regular Air| 1054.82|   107.53|     Liz Pelletier|      Corporate|      Furniture|
|     32|2011-07-15|      24|   1761.4|Delivery Truck|-1748.56|    70.89|     Liz Pelletier|      Corporate|      Furniture|
|     32|2011-07-15|      23| 160.2335|   Regular Air| -85.129|     7.99|     Liz Pelletier|      Corporate|     Technology|
|     32|2011-07-15|      15|   140.56|   Regular Air| -128.38|     8.46|     Liz Pelletier|      Corporate|     Technology|
|     35|2011-10-22|      30|   288.56|   Regular Air|   60.72|     9.11|   Julie Creighton|      Corporate|Office Supplies|
|     35|2011-10-22|      14| 1892.848|   Regular Air|  48.987|   155.99|   Julie Creighton|      Corporate|     Technology|
|     36|2011-11-02|      46|2484.7455|   Regular Air| 657.477|    65.99|  Sample Company A|    Home Office|     Technology|
|     65|2011-03-17|      32|  3812.73|   Regular Air|  1470.3|   115.79|     Tamara Dahlen|      Corporate|     Technology|
|     32|2008-07-15|      26|     null|   Regular Air|    null|   107.53|     Liz Pelletier|      Corporate|      Furniture|
|     32|2008-07-15|      24|     null|Delivery Truck|    null|    70.89|     Liz Pelletier|      Corporate|      Furniture|
|     32|2008-07-15|      23|     null|   Regular Air|    null|     7.99|     Liz Pelletier|      Corporate|     Technology|
|     32|2008-07-15|      15|     null|   Regular Air|    null|     8.46|     Liz Pelletier|      Corporate|     Technology|
|     35|2008-10-22|      30|     null|   Regular Air|    null|     9.11|   Julie Creighton|      Corporate|Office Supplies|
+-------+----------+--------+---------+--------------+--------+---------+-----------------+--------------+---------------+
only showing top 15 rows
```

3. Second Query:

```
6. Quit Program
Enter the number of Query from above...
2
Enter the customer name :  Ex= Liz Pelletier
Liz Pelletier
-------------------------------------------
Batch: 0
-------------------------------------------
+-------+----------+--------+--------+-------------+--------+---------+-------------+---------------+---------------+
|orderId|      date|quantity|   sales|         mode|  profit|unitPrice| customerName|customerSegment|productCategory|
+-------+----------+--------+--------+-------------+--------+---------+-------------+---------------+---------------+
|     32|2011-07-15|      26| 2808.08|   Regular Air| 1054.82|   107.53|Liz Pelletier|      Corporate|      Furniture|
|     32|2011-07-15|      24|  1761.4|Delivery Truck|-1748.56|    70.89|Liz Pelletier|      Corporate|      Furniture|
|     32|2011-07-15|      23|160.2335|   Regular Air| -85.129|     7.99|Liz Pelletier|      Corporate|     Technology|
|     32|2011-07-15|      15|  140.56|   Regular Air| -128.38|     8.46|Liz Pelletier|      Corporate|     Technology|
|     32|2008-07-15|      26|    null|   Regular Air|    null|   107.53|Liz Pelletier|      Corporate|      Furniture|
|     32|2008-07-15|      24|    null|Delivery Truck|    null|    70.89|Liz Pelletier|      Corporate|      Furniture|
|     32|2008-07-15|      23|    null|   Regular Air|    null|     7.99|Liz Pelletier|      Corporate|     Technology|
|     32|2008-07-15|      15|    null|   Regular Air|    null|     8.46|Liz Pelletier|      Corporate|     Technology|
+-------+----------+--------+--------+-------------+--------+---------+-------------+---------------+---------------+

|
```

4. Third Query:

```
6. Quit Program
Enter the number of Query from above...
3
Enter the customer name : Ex= Liz Pelletier
Liz Pelletier
Enter the order date with this format (yyyy-MM-dd) : Ex= 2008-07-15
2008-07-15
-------------------------------------------
Batch: 0
-------------------------------------------
+-------------+----------+-----+------+---------------+
| customerName|      date|sales|profit|productCategory|
+-------------+----------+-----+------+---------------+
|Liz Pelletier|2008-07-15| null|  null|      Furniture|
|Liz Pelletier|2008-07-15| null|  null|      Furniture|
|Liz Pelletier|2008-07-15| null|  null|     Technology|
|Liz Pelletier|2008-07-15| null|  null|     Technology|
+-------------+----------+-----+------+---------------+

|
```

5. *Fourth Query :*

```
6. Quit Program
Enter the number of Query from above...
4
Enter the product category : Ex: Office Supplies
Office Supplies
-----------------------------------------
Batch: 0
-----------------------------------------
+------------------+----------+--------+-----------------+-----+
|      customerName|      date|   sales|           profit|count|
+------------------+----------+--------+-----------------+-----+
|      Aaron Bergman|2011-05-28|   14.76|             1.32|    1|
|         Adam Hart|2011-09-10|  330.21|            83.24|    1|
|       Alan Barnes|2011-10-10|  282.07|           140.01|    1|
|  Alan Schoenberger|2011-05-02| 5410.95|          2077.91|    1|
|       Alan Shonely|2011-12-10|  294.52|            15.34|    1|
|       Alan Shonely|2011-08-12|  240.14|            57.78|    1|
|Aleksandra Gannaway|2011-02-07| 1348.57|            19.57|    1|
|     Alice McCarthy|2010-12-17|  881.65|            41.27|    1|
|       Allen Golden|2011-10-06|  391.42|            25.03|    1|
|   Allen Rosenblatt|2009-11-25|  248.26|             93.8|    1|
|       Andrew Allen|2010-07-14|  278.68|           8.8995|    1|
|        Andy Reiter|2010-03-16|  692.73|            94.97|    1|
|          Ann Blume|2009-11-22|  259.72|            78.05|    1|
|          Ann Chong|2009-08-12|  384.33|          87.6775|    1|
|       Annie Cyprus|2011-08-31|  187.83|            85.96|    1|
|       Annie Cyprus|2011-05-06|   67.24|              4.9|    1|
|      Annie Thurman|2012-02-19|   56.73|             8.33|    1|
|  Anthony O'Donnell|2011-03-01| 1222.68|           300.97|    1|
|     Arianne Irving|2009-09-21|   66.09|            13.71|    1|
|       Art Ferguson|2010-12-29|  430.88|             39.0|    1|
|          Art Miller|2010-06-09|   29.79|             3.76|    1|
```

6. *Fifth Query:*

```
6. Quit Program
Enter the number of Query from above...
5
Enter the product category : Ex: Office Supplies
Office Supplies
Enter the start date : Ex: 2010-03-16
2010-03-16
Enter the end date : Ex: 2011-08-31
2011-08-31
-------------------------------------------
Batch: 0
-------------------------------------------
+------------------+----------+--------+--------+------------------+---------+---------------+-----+
|      customerName|      date|   sales|quantity|            profit|unitPrice|customerSegment|count|
+------------------+----------+--------+--------+------------------+---------+---------------+-----+
|      Aaron Bergman|2011-05-28|   14.76|       5|              1.32|     2.88|      Corporate|    1|
|     Aaron Smayling|2010-12-04|   38.26|      22|             -2.34|     1.76| Small Business|    1|
|     Aaron Smayling|2011-04-29|   76.16|      12|            -24.03|     5.98| Small Business|    1|
| Adam Shillingsburg|2011-08-11|  199.11|      30|            -83.33|     6.48|      Corporate|    1|
|  Alan Schoenberger|2011-05-02| 5410.95|      36|           2077.91|   162.93|      Corporate|    1|
|       Alan Shonely|2011-08-12|  240.14|      36|             57.78|     6.88| Small Business|    1|
|Aleksandra Gannaway|2011-02-07| 1225.52|      36|          -1191.13|    35.48|      Corporate|    1|
|Aleksandra Gannaway|2011-02-07| 1348.57|      47|             19.57|    30.98|      Corporate|    1|
|     Alice McCarthy|2010-12-17|  881.65|      50|             41.27|    18.97|      Corporate|    1|
|       Andrew Allen|2010-07-14|  278.68|      20|            8.8995|    14.48|      Corporate|    1|
|    Andrew Gjertsen|2010-06-04|  436.17|      40|           -141.27|     10.9|       Consumer|    1|
|      Anna Haberlin|2010-05-16|  225.21|      38|            -66.16|     5.98|      Corporate|    1|
|       Annie Cyprus|2011-05-06|   67.24|      23|               4.9|     2.84|    Home Office|    1|
|       Annie Cyprus|2011-07-08|   59.38|       7|           -3.0475|     8.69|    Home Office|    1|
|  Anthony O'Donnell|2011-03-01| 1222.68|      21|            300.97|    59.98|       Consumer|    1|
|       Art Ferguson|2010-12-29|  430.88|      40|              39.0|    11.09|      Corporate|    1|
```

7. *Quit Program:*

```
Menu:
1. Get All Orders form CSV file
2. Get Orders By Customer Name
3. Get Orders(Name,Sales,Profit,Category)  By Customer Name and Order Date
4. Get Orders By Product Category When Profit must be greater than 0 And Sorted by Customer Name
5. Get Orders By Product Category In period and sorted by Sales
6. Quit Program
Enter the number of Query from above...
6
Quitting Program...
Thanks for using this Program...

Process finished with exit code 0
```

XII. Wrapping Up:

In this project, we have created a spark application using **Spark Core** and **Spark Structured Streaming** with **Java**. Here, we have loaded the CSV file into **Data Frame**. And perform some queries in stream analytics. Finally, We saving the content of the streaming Dataset out into the console.

If you want to test the examples above, you will find my Github code link:
Read CSV file into Data Frame and Execute multiple queries in stream analytics