Write a Spark program to read the data from XML file

1. **aggregate data of the price and customers year and monthly**
2. **Create a pivot table with a total price based on months as columns**
3. **Find the total sum of each month total price**

Note: download spark compatible XML jar from maven repo (https://mvnrepository.com/artifact/com.databricks/spark-xml)

```
import org.apache.log4j.Level
import org.apache.log4j.Logger
import org.apache.spark.SparkConf
import org.apache.spark.sql.SparkSession
import scala.collection.mutable.Map
import
org.apache.spark.sql.types.{StructType,DoubleType,IntegerType,StringType}
import org.apache.spark.sql.functions.
{year,month,to_date,col,asc,desc,sum,count,date_format,lit,when,avg,round}


object DataframeAvgOrdersperMonth extends App{

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

      /**
       * Spark Settings and Session Creation
       */

      val sparkConf = new SparkConf()
      sparkConf.set("spark.app.name", "Dataframe Avg Orders per Month")
      sparkConf.set("spark.master", "local[2]")

      val spark = SparkSession.builder()
      .config(sparkConf)
      .getOrCreate()

      /**
       *  external schema for the data
       */

      val orderSchema = new StructType()
      .add("Date",StringType)
      .add("Price",DoubleType)
```

```
      .add("Quantity",IntegerType)
      .add("cust_id",IntegerType)
      .add("order_id",IntegerType)

      /**
       * reading the data from XML file and provide the schema for the data
       */
      val ordersData = spark.read.format("xml")
      .option("rootTag", "dataset")
      .option("rowTag", "record")
      .schema(orderSchema)
      .load("D:\\BIGDATA\\Spark\\dataset.xml")

      /**
       * change the date datatype from String to Date format
       */
      val dateFormatChangedDf = ordersData.withColumn("Date",
to_date(col("Date"),"dd/MM/yyyy"))

      /**
       * get the year and month values from the Date
       */
      val yearandMonthDf = dateFormatChangedDf.withColumn("Year",
year(col("Date")))
      .withColumn("Month", month(col("Date")))
      .drop("Date")

      /**
       * Group by Year and Month column
       * product of Quantity and Price columns then Sum as total_price
       * aggregate on total_price, customer id
       */
      val group_and_AggDf =
yearandMonthDf.groupBy(col("Year"),col("Month"))

.agg(sum(col("Price")*col("Quantity")).as("total_price"),count("cust_id").a
s("customers"))
      .orderBy(desc("Year"),asc("Month"), desc("total_price"))
      .withColumn("Month",when(col("Month").equalTo(1), "Jan")
                  .when(col("Month").equalTo(2), "Feb")
                  .when(col("Month").equalTo(3), "Mar")
                  .when(col("Month").equalTo(4), "Apr")
                  .when(col("Month").equalTo(5), "May")
```

```
                .when(col("Month").equalTo(6), "June")
                .when(col("Month").equalTo(7), "July")
                .when(col("Month").equalTo(8), "Aug")
                .when(col("Month").equalTo(9), "Sep")
                .when(col("Month").equalTo(10), "Oct")
                .when(col("Month").equalTo(11), "Nov")
                .when(col("Month").equalTo(12), "Dec")).persist()

      /**
       * Sequence of column names matching with months
       */
      val monthNames = Seq("Jan","Feb","Mar","Apr","May",
                  "June","July","Aug","Sep","Oct","Nov","Dec","TOTAL")


 /**
  * create Pivot table with total_price in each month
  */
      val pivotDf = group_and_AggDf.groupBy(col("Year"))
      .pivot("Month",monthNames).agg(round(sum("total_price")))
      .na.fill(0)
      .orderBy(desc("year")).persist()

      /**
       * Sum of each Month column total as final TOTAL
       */
      val agg_Month_Cols_PivotDf = pivotDf.withColumn("TOTAL",
col("Jan")+col("Feb")+col("Mar")+col("Apr")+col("May")+col("June")+

col("July")+col("Aug")+col("Sep")+col("Oct")+col("Nov")+col("Dec")).persist
()

            /**
             * Mapping the column Names and Sum of each row data of the
each column
             */
      val col_Names_Map_Pivot = monthNames.map(colName =>
sum(colName).cast("double").as("sum_"+colName))

      /**
       * group and aggregate columns data
       */
      val sum_of_each_MonthDf =
```

```
agg_Month_Cols_PivotDf.groupBy().agg(col_Names_Map_Pivot.head,
col_Names_Map_Pivot.tail:_*)

      /**
       * show group year,month and aggregate of number of customers and
price
       */
      group_and_AggDf.show(Int.MaxValue)

      /**
       * show aggregated Mpnths Pivot
       */
      agg_Month_Cols_PivotDf.show(false)


      /**
       * show each month total Sum of all years data
       */
      sum_of_each_MonthDf.show(false)

}
```

Output :

```
+----+-----+-----------+---------+
|Year|Month|total_price|customers|
+----+-----+-----------+---------+
|2021|  Jan|   278709.0|       10|
|2021|  Feb|   254177.0|        9|
|2021|  Mar|   356598.0|       12|
|2021|  Apr|   298676.0|       11|
|2021|  May|   340813.0|       13|
|2021| June|   304124.0|        8|
|2020|  Jan|   296414.0|       12|
|2020|  Feb|   319061.0|       11|
|2020|  Mar|   495942.0|       17|
|2020|  Apr|   469917.0|       15|
|2020|  May|   308355.0|       14|
|2020| June|   446086.0|       15|
|2020| July|   534506.0|       16|
|2020|  Aug|   321026.0|       13|
```

```
|2020|  Sep|   473957.0|       18|
|2020|  Oct|   294329.0|       14|
|2020|  Nov|   517655.0|       16|
|2020|  Dec|   425055.0|       18|
|2019|  Jan|   500228.0|       14|
|2019|  Feb|   578801.0|       17|
|2019|  Mar|   232452.0|       10|
|2019|  Apr|   214692.0|        8|
|2019|  May|   678439.0|       18|
|2019| June|   604798.0|       20|
|2019| July|   421109.0|       15|
|2019|  Aug|   252178.0|       12|
|2019|  Sep|   454186.0|       17|
|2019|  Oct|   373513.0|       13|
|2019|  Nov|   637708.0|       20|
|2019|  Dec|   567476.0|       18|
|2018|  Jan|   330000.0|       15|
|2018|  Feb|   597295.0|       15|
|2018|  Mar|   443088.0|       16|
|2018|  Apr|   672902.0|       17|
|2018|  May|   389004.0|       12|
|2018| June|   264449.0|       12|
|2018| July|   659173.0|       20|
|2018|  Aug|   332562.0|       11|
|2018|  Sep|   461426.0|       16|
|2018|  Oct|   438085.0|       16|
|2018|  Nov|   332060.0|       10|
|2018|  Dec|   387240.0|       11|
|2017|  Jan|   379500.0|       12|
|2017|  Feb|   265728.0|       14|
|2017|  Mar|   475037.0|       19|
|2017|  Apr|   608103.0|       15|
|2017|  May|   351169.0|       14|
|2017| June|   203252.0|        9|
|2017| July|   349716.0|       12|
|2017|  Aug|   554264.0|       20|
|2017|  Sep|   266969.0|       12|
|2017|  Oct|   381406.0|       16|
|2017|  Nov|   323816.0|       14|
|2017|  Dec|   367502.0|       11|
|2016|  Jan|   320838.0|       11|
|2016|  Feb|   202473.0|       11|
|2016|  Mar|   531201.0|       15|
```

```
|2016|  Apr|   250707.0|        9|
|2016|  May|   471688.0|       14|
|2016| June|   362345.0|       11|
|2016| July|   468708.0|       19|
|2016|  Aug|   526461.0|       17|
|2016|  Sep|   543991.0|       14|
|2016|  Oct|   438761.0|       15|
|2016|  Nov|   363899.0|        9|
|2016|  Dec|   442441.0|       15|
|2015| July|   220951.0|        6|
|2015|  Aug|   604183.0|       16|
|2015|  Sep|   382306.0|       12|
|2015|  Oct|   436579.0|       16|
|2015|  Nov|   514767.0|       16|
|2015|  Dec|   328845.0|       11|
+----+-----+-----------+---------+

+----+--------+--------+--------+--------+--------+--------+--------+--------+------
--+--------+--------+--------+--------+---------+
|Year|Jan     |Feb     |Mar     |Apr     |May     |June    |July    |Aug
|Sep     |Oct     |Nov     |Dec     |TOTAL    |
+----+--------+--------+--------+--------+--------+--------+--------+--------+------
--+--------+--------+--------+--------+---------+
|2021|278709.0|254177.0|356598.0|298676.0|340813.0|304124.0|0.0     |0.0
|0.0     |0.0     |0.0     |0.0     |1833097.0|
|2020|296414.0|319061.0|495942.0|469917.0|308355.0|446086.0|534506.0|321026
.0|473957.0|294329.0|517655.0|425055.0|4902303.0|
|2019|500228.0|578801.0|232452.0|214692.0|678439.0|604798.0|421109.0|252178
.0|454186.0|373513.0|637708.0|567476.0|5515580.0|
|2018|330000.0|597295.0|443088.0|672902.0|389004.0|264449.0|659173.0|332562
.0|461426.0|438085.0|332060.0|387240.0|5307284.0|
|2017|379500.0|265728.0|475037.0|608103.0|351169.0|203252.0|349716.0|554264
.0|266969.0|381406.0|323816.0|367502.0|4526462.0|
|2016|320838.0|202473.0|531201.0|250707.0|471688.0|362345.0|468708.0|526461
.0|543991.0|438761.0|363899.0|442441.0|4923513.0|
|2015|0.0     |0.0     |0.0     |0.0     |0.0     |0.0
|220951.0|604183.0|382306.0|436579.0|514767.0|328845.0|2487631.0|
+----+--------+--------+--------+--------+--------+--------+--------+--------+------
--+--------+--------+--------+--------+---------+

+---------+---------+---------+---------+---------+---------+---------+----
-----+---------+---------+---------+---------+----------+
|sum_Jan  |sum_Feb  |sum_Mar  |sum_Apr  |sum_May  |sum_June |sum_July
```

```
|sum_Aug   |sum_Sep   |sum_Oct   |sum_Nov   |sum_Dec   |sum_TOTAL |
+---------+---------+---------+---------+---------+---------+---------+---
-----+---------+---------+---------+---------+----------+
|2105689.0|2217535.0|2534318.0|2514997.0|2539468.0|2185054.0|2654163.0|2590
674.0|2582835.0|2362673.0|2689905.0|2518559.0|2.949587E7|
+---------+---------+---------+---------+---------+---------+---------+---
-----+---------+---------+---------+---------+----------+
```

To generate data :