Coding Assignment

Introduction

The aim of this exercise is to test your programming ability.

The solution should be written in Scala, as this is the primary language used at Quantexa, and you should aim to write the solution in a functional style. If you find it helpful you may use any functionality from the standard library in your solution.

Please note we will be reviewing both the functionality of your code as well as the style so please ensure you read the Scala coding guidelines, found in the Technical Best Practice, section of the training, before attempting your answers.

Data

You have been provided with a text file in comma separated format of 991 transactions spread over a month. The transactions are for multiple accounts and there are multiple types of transaction. The file has the following columns:

Field	**Description**
transactionId	String representing the id of a transaction
accountId	String representing the id of the account which made the transaction
transactionDay	Integer representing the day the transaction was made on (for simplicity we have removed any time information)
category	String representing the type of category of the transaction
transactionAmount	A double representing the value of the transaction

Assignment

Using the data provided, we would like you to answer the following 3 questions which require calculating some statistics from the data.

Question 1

Calculate the total transaction value for all transactions for each day. The output should include the day number and the total value for this day.

Question 2

Calculate the average value of transactions per account for each type of transaction (there are seven in total). The output should include the account id and the average value for each transaction type (i.e. 7 fields containing the average values).

Question 3

For each day, calculate statistics for each account number for the previous five days of transactions, not including transactions from the day statistics are being calculated for. For example, on day 10 you should consider only the transactions from days 5 to 9 (this is called a rolling time window of five days). The statistics we require to be calculated are:

- The maximum transaction value in the previous 5 days of transactions per account
- The average transaction value of the previous 5 days of transactions per account
- The total transaction value of transactions types “AA”, “CC” and “FF” in the previous 5 days per account

For example, a table containing the aforementioned statistics for 2 accounts and days 9 and 10 would look like the following:

Day	Account ID	Maximum	Average	AA Total Value	CC Total Value	FF Total Value
9	A1	50	45.2	0	97	12
9	A2	400	122.2	1800	0	0
10	A1	50	44	17	92	11
10	A2	700	150	1600	100	0

Output

The expected output which sufficiently answers the assigned questions is described in the following tables. These contain both the final output format and any accompanying data structures required (e.g. case classes).

Please note that your results should adhere to the naming and Type conventions specified in the below tables. This allows for a better and more efficient evaluation of your solution on our behalf.

Type	Name	**Description**
Sequence	question1ResultValue	A Sequence, which contains elements of the Question1Result case class.
Sequence	question2ResultValue	A Sequence, which contains elements of the Question2Result case class.
Sequence	question3ResultValue	A Sequence, which contains elements of the Question3Result case class.

Table 1: Expected output format

Type	Name	Fields	Field Type	Description
Case class	Question1Result	transactionDay transactionTotal	Int Double	A case class containing the relevant output information for question 1.
Case class	Question2Result	accountId categoryAvgValueMap	String Map[String,Double]	A case class containing the relevant output information for question 2.
Case class	Question3Result	transactionDay accountId max avg aaTotal ccTotal ffTotal	Int String Double Double Double Double Double	A case class containing the relevant output information for question 3.

Table 2: Auxiliary case classes

Code

The following code can be used to import the data and create a List of transactions. The List can, then, be used to calculate the required statistics.

Scala 2.11:

```
import scala.io.Source

object TransactionsAssignment{

//Define a case class Transaction which represents a transaction
  case class Transaction(
    transactionId: String,
    accountId: String,
    transactionDay: Int,
    category: String,
    transactionAmount: Double)

//The full path to the file to import
  val fileName = getClass.getResource("/transactions.csv").getPath

//The lines of the CSV file (dropping the first to remove the header)
  val transactionslines = Source.fromFile(fileName).getLines().drop(1)

//Here we split each line up by commas and construct Transactions
  val transactions: List[Transaction] = transactionslines.map { line =>
    val split = line.split(',')
    Transaction(split(0), split(1), split(2).toInt, split(3), split(4).toDouble)
  }.toList

//END OF GIVEN CODE

}
```