**STRICTLY CONFIDENTIAL – DO NOT DISTRIBUTE**

Quantexa - Coding Assignment

## Introduction

The aim of this exercise is to test your programming ability. It is a take home assignment which has been sent to you before a potential interview. If you are invited in an interview, we will discuss your solution to the problem.

The solution should be written in Spark/Scala, as this is the primary language used at Quantexa. Ideally it should be written in a functional style. You are allowed to use any functionality from the standard library in your solution. If you write the solution in Spark have a think about the scalability of your solution (if the data set were much larger, say).

## Assignment

### Data

You have been provided with two CSV files. The files have the following columns:

flightData.csv

|  |  |
| --- | --- |
| Field | Description |
| passengerId | Integer representing the id of a passenger |
| flightId | Integer representing the id of a flight |
| from | String representing the departure country |
| to | String representing the destination country |
| date | String representing the date of a flight |

passengers.csv

|  |  |
| --- | --- |
| Field | Description |
| passengerId | Integer representing the id of a passenger |
| firstName | String representing the first name of a passenger |
| lastName | String representing the last name of a passenger |

### Questions

Using the data provided, we would like you to answer the following 4 questions which requires calculating some statistics from the data. The output for each of the questions can either be provided as 4 files or the code can simply output the results to the console.

#### **Question 1**

Find the total number of flights for each month.

The output should be in the following format:

|  |  |
| --- | --- |
| Month | Number of Flights |
| 1 | 123 |
| 2 | 456 |
| … | … |

#### **Question 2**

Find the names of the 100 most frequent flyers.

The output should be in the following format:

|  |  |  |  |
| --- | --- | --- | --- |
| Passenger ID | Number of Flights | First name | Last name |
| 123 | 100 | Firstname | Lastname |
| 456 | 75 | Firstname | Lastname |
| … | … | … | … |

#### **Question 3**

Find the greatest number of countries a passenger has been in without being in the UK. For example, if the countries a passenger was in were: UK -> FR -> US -> CN -> UK -> DE -> UK, the correct answer would be 3 countries.

The output should be in the following format:

|  |  |
| --- | --- |
| Passenger ID | Longest Run |
| 45 | 4 |
| 23 | 6 |
| … | … |

#### **Question 4**

Find the passengers who have been on more than 3 flights together.

The output should be in the following format:

|  |  |  |
| --- | --- | --- |
| Passenger 1 ID | Passenger 2 ID | Number of flights together |
| 56 | 78 | 6 |
| 12 | 34 | 8 |
| … | … | … |

#### For extra marks

Find the passengers who have been on more than N flights together within the range (*from*,*to*).

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Passenger 1 ID | Passenger 2 ID | Number of flights together | From | To |
| 56 | 78 | 6 | 2017-01-01 | 2017-03-01 |
| 12 | 34 | 8 | 2017-04-05 | 2017-12-01 |
| … | … | … | … | … |

The function should look something like:

**def** flownTogether(atLeastNTimes: Int, from: Date, to: Date) = {  
 ...  
}

## **Submission**

We recommend submitting your code as a zip file, including any VCS files (e.g., .git directory if you’ve used Git) and README if applicable.

You work will be assessed on

* correctness
* performance
* code and engineering quality
* testing strategies
* presentation, including documentation and ease of usage