**Data Engineer - Candidate Questions**

The following questions are designed to test the candidate’s knowledge and skills in relation to the position being filled.

**Q1.**

Problem, given the table definitions below write a single SQL query that fulfils the following business requirement.

**Source Tables**

You are given the following tables:

REVENUE\_ANALYSIS

ACTIVITY\_DATE DATE NOT NULL, Date wager was madeMEMBER\_ID INTEGER NOT NULL, Unique Player identifier  
GAME\_ID SMALLINT NOT NULL, Unique Game identifier  
WAGER\_AMOUNT REAL NOT NULL, Total amount wagered on the gameNUMBER\_OF\_WAGERS INTEGER NOT NULL, Number of wagers on the gameWIN\_AMOUNT REAL NOT NULL, Total amount won on the gameACTIVITY\_YEAR\_MONTH INTEGER NOT NULL, YYYYMMBANK\_TYPE\_ID SMALLINT DEFAULT 0 NOT NULL 0=Real money, 1=Bonus money

The “REVENUE\_ANALYSIS” table provides a summary of each member’s activity on a given date, listing the total amounts wagered and won for each game played that day. If a member does not play on a given day there will be no entries in the table. You can assume this is a very large table ~100M rows.

CALENDAR

CALENDAR\_DATE DATE NOT NULL, Base date (YYYY-MM-DD)  
CALENDAR\_YEAR INTEGER NOT NULL, 2010, 2011 etc  
CALENDAR\_MONTH\_NUMBER INTEGER NOT NULL, 1-12  
CALENDAR\_MONTH\_NAME VARCHAR(100), January, February etc  
CALENDAR\_DAY\_OF\_MONTH INTEGER NOT NULL, 1-31  
CALENDAR\_DAY\_OF\_WEEK INTEGER NOT NULL, 1-7  
CALENDAR\_DAY\_NAME VARCHAR(100), Monday, Tuesday etc  
CALENDAR\_YEAR\_MONTH INTEGER NOT NULL, 201011, 201012, 201101 etc

This CALENDAR table provides a base “date” dimension, one row per day from 2015 to 2020

The required solution is a view with the following columns:

MEMBER\_ID,  
CALENDAR\_YEAR\_MONTH,  
MEMBER\_LIFECYCLE\_STATUS,  
LAPSED\_MONTHS

Active means that the member has made a minimum of one **real money** wager in the month.

Each month a member has a certain lifecycle type. The member’s status will change on a monthly basis based on their previous and current month's activity. The statuses are:

**New** = First time they place a real money wager

**Retained** = Active in the prior calendar month and the current calendar month  
**Unretained** = Active in the prior calendar month, but not active in the current calendar month  
**Reactivated** = Not active in the prior calendar month, but active in the current calendar month  
**Lapsed** = Not active in the prior calendar month or the current calendar month

The view should display one row per member per month, starting from the month in which they first placed a real money wager. This view should give their lifecycle status for that month, and if the member has lapsed, it should show a rolling count of the number of months since they were last active.

We have provided the following files (DB\_Setup.zip) to help set up the environment:

* table\_setup.sql is generated from DB2 and contains the DDL of CALENDAR and REVENUE\_ANALYSIS tables
* Calendar\_Test\_Data.csv holds the CALENDAR data between 2015-2020
* Revenue\_Analysis\_Test\_Data.csv holds a minimal amount of example data for REVENUE\_ANALYSIS, please feel free to add more test data to REVENUE\_ANALYSIS to cover different scenarios.

**Q2.**

Based on the REVENUE\_ANALYSIS dataset above, build a RESTful web service that can be queried for:

* the total win amount for a given member,
* the total wager amount for a given member, and
* the number of wagers placed by a given member.

Your solution should also meet the following requirements:

* All responses should be in JSON format.
* Each RESTful endpoint should accept an optional parameter for calculating the totals for a specific month, and an optional parameter for a specific game type.
* The web service can be built using either Java or Python. Ideally, we would like it to be provided with a build and dependency manager, such as Maven, with all dependencies accessible through publicly available repositories. Otherwise, please package all dependencies in your solution. Regardless of the option, please provide instructions for building and deploying the application in a README.md file.
* You are not limited to any particular framework and we don't expect any data to persist beyond the lifespan of the server.
* Where possible, you should write tests for the application.

Copyright

This document is produced by Gamesys Ltd, exclusively for use of its employees, authorised agents, and affiliated companies, in accordance with the confidentiality marking shown below.

It contains Gamesys confidential and proprietary information; under no circumstances should it be delivered or disclosed to any person not employed by Gamesys or its affiliated companies, without express written permission of Gamesys Limited.