**Two Six Capital**

**Candidate Data Engineering Challenge**

Overview

This data engineering challenge reflects the kind of tasks we undertake when working on private equity deals. A Data Engineer is someone who has specialized their skills in creating software solutions around big data. This means that the Data Engineer must be capable of adapting to many situations and working with a vast panoply of tools.

Statement

The challenge is split into two parts.

**First**, setup a REST API that is accessible from the web in order to compute metrics against a given the dataset. The data received should be cleaned, processed, and stored in a SQL database that feeds the REST API.

**Second,** answer a few short follow-up technical questions.

Deliverables

1. All the code behind this challenge
2. A report containing answers to the technical questions
3. An IP address to query the API

Recommended Steps:

We recommend you follow the following steps when working on this challenge.

1. Read this document fully
2. Choose a cloud provider or self-host for the REST API and the SQL database
3. Read and understand the data
4. Clean the data if necessary
5. Load the data into a SQL database of your choosing
6. Deploy a REST API that follows the given structure to query the data
7. Make the API available via a public IP
8. Test that you can access the public IP
9. Answer the follow up technical questions

Metrics

1. **Revenue**
2. **Active User Count**
   * Number of users who have made at least one transaction in the given year
3. **New User Count** 
   * Number of users who have joined in the given year
4. **Average Revenue per Active User**
   * Revenue / Active User Count

Data

Each CSV file has the following columns:

* transaction\_date – date when transaction was made
* user – unique customer identifier
* amount – gross revenue, i.e., how much a customer spent
* join\_date – date of a customer’s first purchase
* region – geographical region where the transaction was made

API Structure

The API you create must follow the following structure for each metric:

* Revenue: <URL>/revenue/
* Active User Count: <URL>/activeusers/
* New User Count: <URL>/newusercount/
* Average Revenue per Active User: <URL>/arpau/

Where <URL> will be the public IP address of the API server and would be of the form: http://xxx.xxx.x.x

The API must respond with a yearly time series formatted as a JSON payload of the form.

{ “newusercount” : { “2014” : xxx, “2015”: xxx, “2016”: xxx, “2017”: xxx}} where xxx is a placeholder for the actual value for that year. Please do the same for all other metrics.

This means that when someone performs a valid API call, the server must perform the right SQL query against the database and return the response as structured above. The entire request-response process should be very quick.

Technical Questions

Please answer these questions briefly. No need to write any code to answer them.

1. What changes would you make in order to be able to segment each metric:
   1. By region
   2. By year joined
2. What tests would you write in order to check for:
   1. Correctness of data
   2. Correctness of metrics
   3. Correctness of API behavior
3. Suppose additional data were to be sent daily, what changes would you make to allow for the API to report updated numbers daily?
4. What would you add to the API you have built to give more value to a business?

Notes on Implementation

In order to perform this task, please feel free to use the cloud provider of your choice (preferred), or self-hosted. Cloud providers such as AWS or Azure provide the necessary services to implement this via their free tier.

Feel free to use the backend framework of your choice and the SQL backend of your choice. Please choose whichever technologies allow you to perform best at this task, we do not grade based on framework or programming language. We prefer to focus on results and ability to explain the choices you have made at each step.

Guidelines

* We allow each candidate 1 full week with the challenge, but you should expect to spend no more than 8 hours on this data engineering challenge.
* Please feel free to take as much time you need to complete this challenge.
* We highly recommend you spend some time doing data exploration and cleaning.
* We highly recommend you test your code and the API before submitting.
* We recommend you attempt the questions one step at a time and document each step of the process.
* Feel free to either submit your code and report by email or via SFTP. If you prefer to submit via SFTP, please place the files in a folder on your machine and allow SSH access for us to download.