Data Engineer Skills Assessment

Thank you for your interest in the Data Engineer position at Everybody Votes Campaign (EVC)!

The following assessment is an opportunity to demonstrate the extent of your data engineering knowledge. We will be scoring this assessment on its completeness and your ability to correctly respond to the prompts. We will also be looking at your *creativity* and *critical thinking* in your responses. It is best to show and explain your work and thought process, *even if you are unable to complete the task in full.*

We ask that you carefully read the prompts, document your code, and follow coding best practices. This includes the expectation that you are storing all your work in a git repository and making frequent commits.

The tasks are organized in the order in which we reccommend that you complete them. That is, complete task `01-data-wrangling.md` before moving on to task `02_sql_queries.md`, etc.

Note: *the* ***Overview*** *section for each of the tasks it aimed as presenting a likely scenario you could encouter at EVC.*

Guidelines

- The target time for this assessment is three (3) to four (4) hours
- You have a twenty-four (24) hour window in which to complete the assessment
- You may use online or offline resources
- You may **not** consult nor share this assessment with any other person
- If you use substantial portions of code without significant modification in your responses, cite the author and/or source where you found the information.

Deliverables

All materials should be emailed back to us in a compressed folder (i.e. git repository). Be sure to include your solutions, as well as any documents you produce. Please name files appropriately.

Setup

Before you get started on the tasks, verify that you can log into the online platform used in task `02_sql_queries.md` and that you are able to connect to the database as described in `01_data_wrangling.md`. Use the credentials shared with you in the initial email.

Data Wrangling

Target time: 60 mins

Reminder: *Before you get started on this task, verify that you can log into the online platform as described in Setup.*

Overview

We are working with new partners, Vendor X and Vendor Y, and need to ingest their data into our data warehouse. Vendor X sends us a csv file each day. Vendor Y makes their data available via their API. We need to download the files, merge them and then load them into our database.

The goal is to set up an automated process to ingest this data. We need our Data Engineer to write a script that will:

1. Download Vendor Y's data from the API
    1. API URL: https://k4clzaf58d.execute-api.us-east-1.amazonaws.com/default/handle_users
2. Merge Vendor Y's data with Vendor X's data
    1. Vendor X's data: `data/vendor_x_data.csv`
3. Standardize the column names and the data
    1. See Data section below for more info
4. Save data to a file
    1. `all_vendors.csv`
5. Import data in to the database
    1. Schema name (the result of this query using your username): `select 's_<hash>;` (use the schema name shared with you)
    2. Table name: `all_vendors`

You may use any programming language you are comfortable with - our preference is Python. In the README you sumbit, make sure to add a section about how to run the script. If you are short on time, you may write pseudo code for a script that would accomplish this task.

NOTE: *This task, although related, is independent of the SQL task. Even if you encounter challenges with this task, you will still be able to complete the SQL task.*

Data

NOTE: *all data is randomly generated.*

The imported table should have the following structure:

column_name	data_type
vendor_id	`varchar(1024)`
status	`varchar(1024)`
tracking_source	`varchar(1024)`
tracking_id	`varchar(1024)`
date_of_birth	`date`
email_address	`varchar(1024)`
citizenship_confirmed	`bool`
salutation	`varchar(1024)`
first_name	`varchar(1024)`
middle_name	`varchar(1024)`
last_name	`varchar(1024)`
name_suffix	`varchar(1024)`
home_address	`varchar(1024)`
home_unit	`int`
home_city	`varchar(1024)`
home_county	`varchar(1024)`
home_state	`varchar(1024)`
home_zip_code	`varchar(1024)`
mailing_address	`varchar(1024)`

column_name	data_type
mailing_unit	int
mailing_city	varchar(1024)
mailing_county	varchar(1024)
mailing_state	varchar(1024)
mailing_zip_code	varchar(1024)
party	varchar(1024)
race	varchar(1024)
phone	varchar(1024)
phone_type	varchar(1024)
opt_in_to_vendor_email	bool
opt_in_to_vendor_sms	bool
opt_in_to_partner_email	bool
opt_in_to_partner_smsrobocall	bool
volunteer_for_vendor	bool
volunteer_for_partner	bool
pre_registered	bool
registration_date	timestamp
finish_with_state	bool
built_via_api	bool
submitted_via_state_api	bool
registration_source	varchar(1024)
shift_id	int
shift_type	int
office	varchar(1024)
vendor_a_shift_id	int
salutation_standardized	varchar(1024)
has_mailing_address_standardized	bool
has_state_license_standardized	bool
has_ssn_standardized	bool
predicted_gender	varchar(1024)
org	varchar(1024)
evc_id	varchar(1024)
program_state	varchar(1024)
partner_id	int
field_start	timestamp
field_end	timestamp

Setup

Database connection details:

The username and password were shared with you in the inital email.

```
USERNAME: u_<hash>
PASSWORD: p_<hash>
HOST: 68.183.51.176
DATABASE: sqlpad
PORT: 5432
```

Deliverables

1. README (text or markdown preferred)
2. Source code
3. `all_vendors.csv`
4. [data imported into database]

Tips

- Ensure the final CSV has valid column names
- Write your code so that it's reusable and and flexible

SQL Queries

Target time: 60 mins

Reminder: *Before you get started on this task, verify that you can log into the online platform as described in Setup.*

Overview

We are working with **Vendor X**, and need to ingest their data into our data warehouse. They send us a daily file, `data/vendor_x_data.csv` and also send completed registration data to a **QC Vendor** we have an existing data sync with in order to quality check the registrations. *Even though you uploaded Vendor X's data as part of the data wrangling task, use the* `public.vendor_x_registrations_raw` *table for this task.*

We need our Data Engineer to:

1. Create a deduped version of the Vendor X table, `public.vendor_x_registrations_raw`, by removing duplicate rows that appear for each registration status
    1. Schema name (the result of this query using your username): `select 's_<hash>');`
    2. Table name: `vendor_x_registrations_deduped`
2. Create a table, `all_records`. Format and add the deduped set of Vendor X records to that table by editing `all_records_generation_TO_EDIT.sql`. Because completed registrations appear in the QC Vendor table `qc_vendor_data` that feeds into `all_records`, you'll need to ensure that you've removed the duplicate *complete* records that appear in both tables.
    1. Schema name (the result of this query using your username): `select 's_<hash>');`
    2. Table name: `all_records`

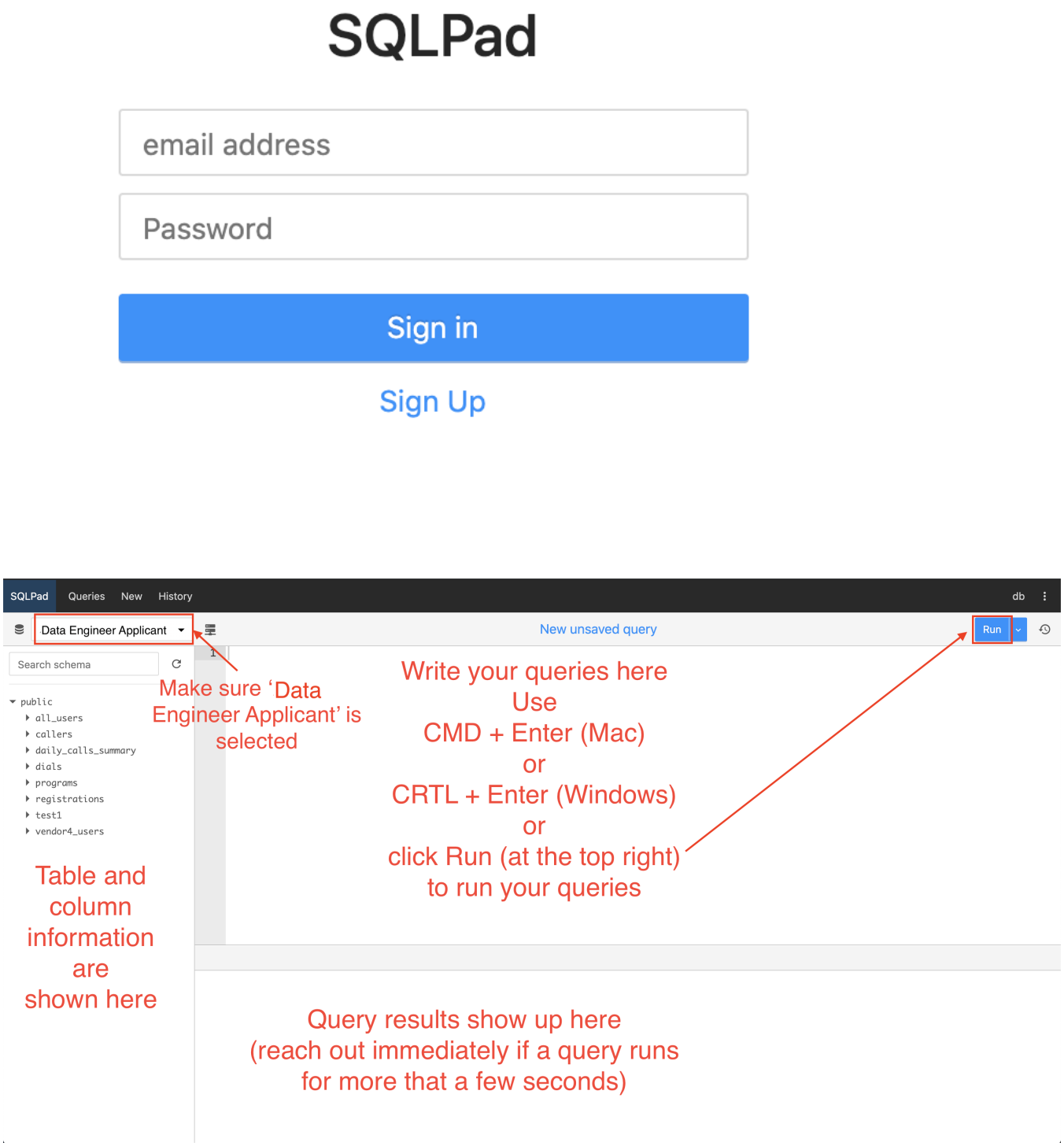
The schema name was shared with you in the inital email.

Additional information for processing Vendor X data

- Vendor X sends one record for each step in the registration process that has been completed, e.g. a complete registration would have a total of 5 rows in the data: one row for each of steps 1-4 and one row for "complete." The step or completion associated with each row is in the `status` field. We also need to dedupe the multiple records per registration that we receive from Vendor X using the `status` column. Unique registrations can be determined using the `vendor_id` field, meaning a complete registration will have 5 rows each with the same `vendor_id`.
- When inserting Vendor X registrations into all_records, tag them with program_type = 'field', and program_sub_type = 'evc_funded'. Please use the source column registration_source to populate collection_medium in the final all_records table. The fields evc_month and evc_year should be generated by extracting the relevant date parts from registration_date. There may be several other columns that do not appear in the Vendor X data and should be imputed with null values, or columns that require some transformation to match the data type in the `qc_vendor_data` table.
- `vendor_id` from Vendor X data corresponds to `application_id` in the QC Vendor data. All completed Vendor X (e.g. have reached `status = 'Complete'`) records are sent to our QC Vendor, so they appear in the `qc_vendor_data` table. Remove those duplicate "complete" records so that the final `all_records` table does not have any duplicates.

Setup

We have set up a sandbox database and have created a unique user for you to use. Visit http://68.183.51.176/ and log in using your email address and the last four digits of your phone number. Once logged in, select **Data Engineer Applicant** as the connection (near the top left).



Before you get started, verify you can access it and reach out immediately if you have any issues connecting or querying.

Deliverables:

1. [Deduped table created in the database]
2. SQL source code for creating the deduped table
3. [all_records table created in the database]

4. Edited copy of `all_records_generation_TO_EDIT.sql`

Tips

- Add comments to your code to give more context

Data Engineer Concepts

Target time: 45 mins

Context

Below are some questions about general Data Engineering concepts and how you might approach various scenarios. Be detailed in your responses (e.g. include concrete examples), but please keep your responses to no more than 2 paragraphs.

1. What do you think is important to consider before ingesting data into a pre-existing reporting structure? What questions would you ask? What information would you seek?
2. How would you build consensus around making changes to data processing or pipelines with other members of a multidisciplinary team? How do you know when you have an agreement to move forward? What might you consider when presenting your proposal to the team?
3. How do you know if you have "good data?" How would you approach cleaning data to make it "good data?"

Deliverables

1. README (text or markdown preferred) with your responses