Analytics Engineer Skills Assessment

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

The following assessment is an opportunity to demonstrate the extent of your analytics engineering knowledge. We will be scoring this assessment on its completeness and ability to correctly respond to the prompts. However, 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 prefer that you complete them. That is, complete task `01-schema-design.md` before moving on to task `02-sql-queries.md`, etc.

Note: *the* ***Context*** *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) 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 any other person
- If you use substantial portions of code without significant modification in your answers, cite the author and/or repository 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`.

Schema Design

Target time: sixty (60) minutes

Context

We are rapidly increasing the amount of data that we are ingesting. In order to ensure that it is easy to analyze and report on, we are hoping to re-structure it in a more organized way. Specifically, we are looking to report on voter registration forms collected across collection-mode by organization.

Given the source tables below, put together an Entity Relationship Diagram (ERD) the shows how you would organize the tables. Be sure to include details about your thought process and why you made the decision you did, in the README you sumbit. Feel free to use any software you like to generate the diagram (even non-specific software such as MS Word or Google Docs), and export the final product as a pdf or image file.

Data

** PII = Personally Identifiable Information, i.e. name, address, etc.*

field_source1

column	description
fs1_id	the vendor's unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII
registration_date	the date the regisration was collected
organization	the name of the organization the collected the registration form
organization_state	the state in which the organization ran the VR program
organization_funding_level	the level of funding the organization receieved

field_source2

column	description
fs2_id	the vendor's unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII

column	description
registration_date	the date the regisration was collected
organization	the name of the organization the collected the registration form
organization_state	the state in which the organization ran the VR program
organization_funding_level	the level of funding the organization receieved

mail_source1

column	description
ms1_id	the vendor's unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII
date_mail_sent	the date the regisration was mailed out
date_mail_receieved	the date the regisration was recieved
organization	the name of the organization the collected the registration form
organization_state	the state in which the organization ran the VR program

remote_source1

column	description
rs1_id	the vendor's unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII
registration_started	the date the regisration was started online
registration_submitted	the date the regisration was was submitted online
application_step	the step the applicant reached when filling out the application
org_slug	the slug for the organization that collected the registration form
org_program_state	the state in which the organization ran the online VR program

remote_source2

column	description
rs2_id	the vendor's unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII
registration_completed_at	the date the regisration was was submitted
canvasser_id	the id of the canvasser that collected the registration
org_code	the code for the organization

applicant

column	description
id	unique id
prefix	PII
first_name	PII
middle_name	PII
last_name	PII
suffix	PII
street	PII
city	PII
state	PII
zip_code	PII
email	PII
phone_number	PII
date_of_birth	PII
registration_date	the date the regisration was was collected/submitted
org_id	the id for the organization

Deliverables

1. ERD document (pdf or jpg/png preferred)
2. README (text or markdown preferred)

SQL Queries

Target time: forty-five (45) minutes

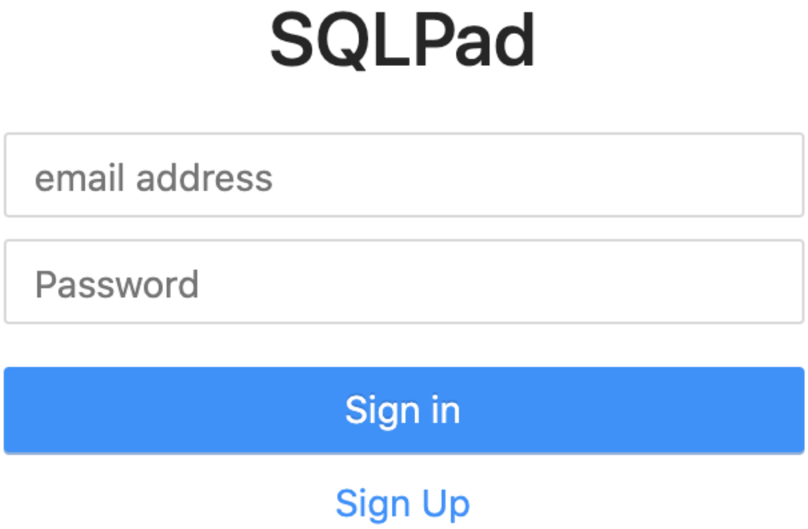
Context

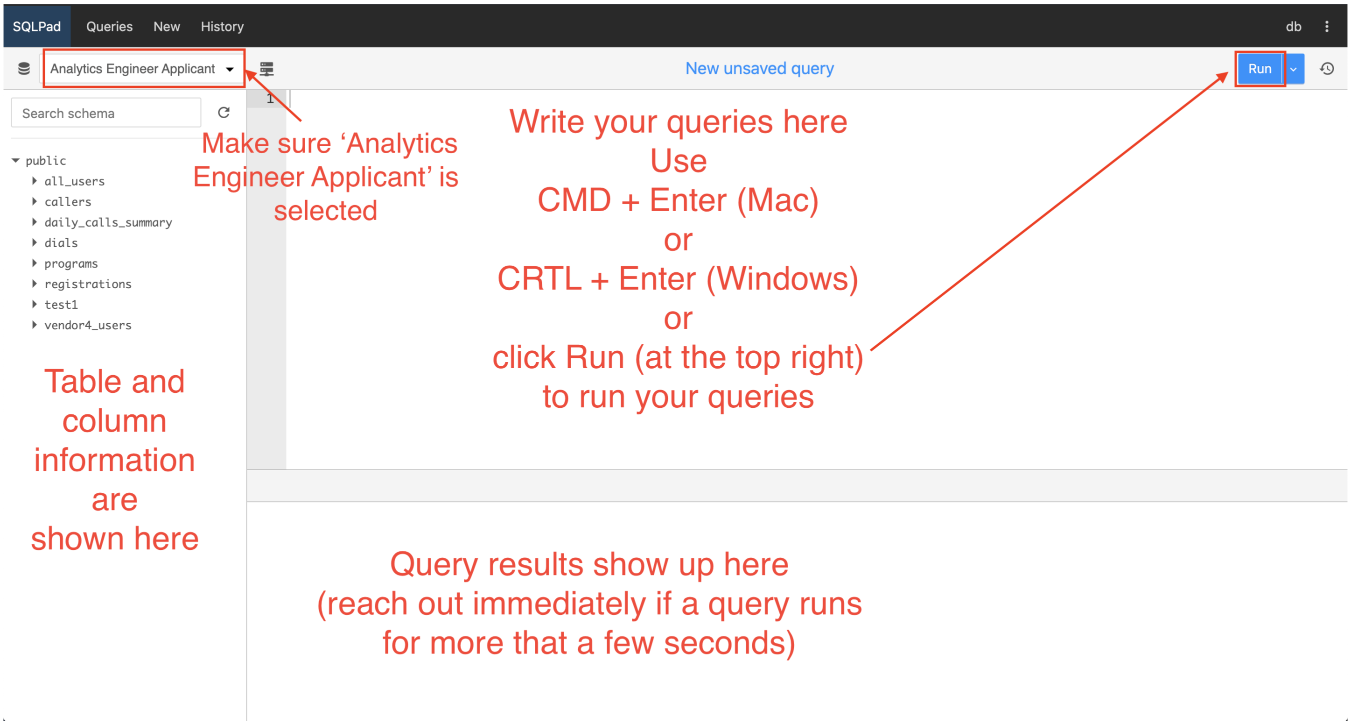
We are trying to standardize and analyze various parts of our program. Given the tables below, write queries to respond to the following tasks.

1. We are trying to insert vendor4's data into our `all_users` table. Write a *select* query that returns the stadardized data that could be inserted into the `all_users` table.
2. Write a query using the `registrations` table that returns the total number of registrations(`total`), the number of completed registrations (`completed_registrations`) and the number of valid registrations that are incomplete (`valid_incomplete_registrations`), by organization and state.
3. Write a query (using the `callers`, `dialers`, `programs` tables) that returns the `program_name`, `program_date`, `caller_name` and number of calls made (`num_calls`). Use `'autodialer'` as the name for calls without a caller.

Setup

We have set up a sandbox database and have created a unique user for you to use. Visit http://45.55.61.152/ and log in using your email address and the last four digits of your phone number. Once logged in, select **Analytics Engineer Applicant** as the connection (near the top left).





Make sure 'Analytics Engineer Applicant' is selected

Write your queries here
Use
CMD + Enter (Mac)
or
CRTL + Enter (Windows)
or
click Run (at the top right)
to run your queries

Table and column information are shown here

Query results show up here
(reach out immediately if a query runs for more that a few seconds)

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

Data

NOTE: *all data is randomly generated.*

all_users

column	type	description
id	`varchar(210)`	The user's id
first_name	`varchar(50)`	The user's first name
last_name	`varchar(50)`	The user's last name
phone_number	`bigint`	The user's phone number
zip_code	`varchar(5)`	The user's home zip code
month_registerted	`varchar(15)`	The month the user registered

vendor4_users

column	type	description
last_first_name	`varchar(110)`	The user's name (`'last_name, firs_name'`)
email	`varchar(100)`	The user's email
phone	`varchar(18)`	The user's phone number
zip_code	`varchar(10)`	The user's zip code
reg_date	`timestamp`	The date the user regisered
is_valid_registration	`bool`	A flag for whether the registration is valid
email_opt_in	`bool`	A flag for the user's email opt in status
phone_opt_in	`bool`	A flag for the user's email opt in status

registrations

column	type	description
file_id	`varchar(36)`	The id for the registration file
org	`varchar(50)`	The name of the organization that collected the registration
state	`varchar(2)`	The state in which the organization runs its program
is_complete_registration	`bool`	A flag for whether the registration is complete
is_valid_registration	`bool`	A flag for whether the registration is valid
reg_date	`timestamp`	The date the registration was collected

callers

column	type	description
id	`varchar(32)`	The caller's id
name	`varchar(100)`	The caller's full name
username	`varchar(50)`	The caller's username

dials

column	type	description
caller_id	`varchar(32)`	The caller's id
program_id	`varchar(32)`	The program's id
registrant_id	`varchar(36)`	The registrant's id
registrant_name	`varchar(100)`	The registrant's name
registrant_phone	`varchar(12)`	The registrant's phone
registrant_response	`varchar(15)`	The registrant's response

programs

column	type	description
id	`varchar(32)`	The program's id
name	`varchar(50)`	The program's name
date	`timestamp`	The program's date

Deliverables

1. queries.sql

Tips

- Add comments to the query code describing your thought process

Data Wrangling

Target time: forty-five (45) minutes

Context

We have started running a new program and we wanted to get a list of all the users that have been processed. For this program, we used three (3) different vendors and each one gave us the data in a different format. The task is to combine the data and prepare it to be loaded into our data warehouse. We anticipate that we will continue running this program long term. Therefore, consider that it would be included as part of a larger pipeline.

You may use any programming language you're comfortable with - our preference, in order, is Python, R, other. 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.

Data

NOTE: *all data is randomly generated.*

1. vendor1-users.csv
2. vendor2-users.json
   - *note: the file is line delimited*
3. vendor3-users
   - Docs

Mappings

all-users	vendor1-users	vendor2-users	vendor3-users
id	vendor1_id	vendor2_id	login.uuid
prefix	prefix		name.title
first_name	first_name	firstName	name.first
middle_name	middle_name	middleName	
last_name	last_name	lastName	name.last
suffix	suffix	suffix	
street	addr	addressLine1	location.street
city	city	city	location.city
state	state	state	location.state
zip_code	zip	zipCode	location.postcode
email	email	email	email
phone_number	phone_num	phoneNum	phone
date_of_birth	dob	birthDate	dob.date
registration_date	date_registrated	registrationDate	registered.date

Deliverables

1. all-user.csv
2. source code
3. README (text or markdown preferred)

Tips

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