Fetch Rewards

Data Engineering Take Home: ETL off a SQS Queue

You may use any programming language to complete this exercise. We strongly encourage you to write a README to explain how to run your application and summarize your thought process.

What do I need to do?

This challenge will focus on your ability to write a small application that can read from an AWS SQS Queue, transform that data, then write to a Postgres database. This project includes steps for using docker to run all the components locally, **you do not need an AWS account to do this take home.**

Your objective is to:

1. read JSON data containing user login behavior from an AWS SQS Queue, that is made available via a custom localstack image that has the data pre loaded.
2. Fetch wants to hide personal identifiable information (PII). The fields `device_id` and `ip` should be masked, but in a way where it is easy for data analysts to identify duplicate values in those fields.
3. Once you have flattened the JSON data object and masked those two fields, write each record to a Postgres database that is made available via a custom postgres image that has the tables pre created.

Note the target table's DDL is:

```
-- Creation of user_logins table
CREATE TABLE IF NOT EXISTS user_logins(
    user_id             varchar(128),
    device_type         varchar(32),
    masked_ip           varchar(256),
    masked_device_id    varchar(256),
    locale              varchar(32),
    app_version         integer,
    create_date         date
);
```

You will have to make a number of decisions as you develop this solution:

- How will you read messages from the queue?
- What type of data structures should be used?
- How will you mask the PII data so that duplicate values can be identified?
- What will be your strategy for connecting and writing to Postgres?

- Where and how will your application run?

The recommended time to spend on this take home is 2-3 hours. Make use of code stubs, doc strings, and a next steps section in your README to elaborate on ways that you would continue fleshing out this project if you had the time.

Questions

For this assignment an ounce of communication and organization is worth a pound of execution. Please answer the following questions:

- How would you deploy this application in production?
- What other components would you want to add to make this production ready?
- How can this application scale with a growing dataset.
- How can PII be recovered later on?
- What are the assumptions you made?

Project Setup

1. A Github, GitLab, Bitbucket, etc... account.
2. You will need the following installed on your local machine
    a. docker -- docker install guide
    b. docker-compose
    c. pip install awscli-local
    d. Psql - install
3. Use the following docker images with test data baked in:
    a. Postgres
    b. Localstack

    Example docker-compose yaml to run the test environment:

```
version: "3.9"
services:
  localstack:
    image: fetchdocker/data-takehome-localstack
    ports:
      - "4566:4566"
  postgres:
    image: fetchdocker/data-takehome-postgres
    ports:
      - 5432:5432
```

4. Postgres creds:

   a. Password: postgres
   b. Username: postgres

5. Test local access
   a. Read a message from the queue using awslocal, `awslocal sqs receive-message --queue-url http://localhost:4566/000000000000/login-queue`
   b. Connect to the Postgres database, verify the table is created
      i. psql -d postgres -U postgres -p 5432 -h localhost -W
      ii. postgres=# select * from user_logins;

All done, now what?

Upload your codebase to a public Git repo (GitHub, Bitbucket, etc.) and please submit your Link where it says to - under the exercise via Green House our ATS. Please double-check this is publicly accessible.

Please assume the evaluator does not have prior experience executing programs in your chosen language and needs documentation to understand how to run your code.