Fetch Rewards
Data Engineering - Take home by Surya Vaddhiparthy

Decisions

How will you read messages from the queue?
I will use the Boto3 library and the sqs.receive_message() method to read messages from the queue.

What type of data structures should be used?
Combination of dictionaries and lists.

How will you mask the PII data so that duplicate values can be identified?
To mask the PII data, I would use a one-way hashing function such as SHA-256, which will create a unique hash for each value while still allowing duplicate values to be identified by comparing their corresponding hashes.

What will be your strategy for connecting and writing to Postgres?
To connect and write to Postgres, I would use the psycopg2 library, which is a Postgres adapter for Python. I would use psycopg2.connect() method to connect to the Postgres database, and the psycopg2.cursor() and cursor.execute() methods to write the data to the appropriate table.

Where and how will your application run?
It could run as a script on a local machine or containerized environment like docker. The script would need to run on an environment where it has access to the internet and the correct credentials to connect to the AWS SQS and Postgre.

Steps

Extracting the data from SQS:
The script connects to the local stack SQS service and receives messages from the SQS queue running on the Docker image on port 4566. The messages are then parsed as JSON data and stored in a variable.

Transforming the data:
The data is transformed by masking the device_id and IP fields, using hashing
assuming semantic versioning convention major.minor.patch structure
the major portion of the version number i.e., converting the app version to an integer to suit the data type listed in the DDL statement provided

Loading the data into PostgreSQL:
The script connects to the local stack Postgres service running on the second Docker image on port 5432. The script creates a table in the PostgreSQL database (if it doesn't exist) with the provided schema and then inserts the transformed data into the table.

Questions?

How would you deploy this application in production?

To deploy this application in production, I recommend using a cloud-based infrastructure, such as AWS or GCP, to run the application. Additionally, a CI/CD pipeline can be set up to automatically build and deploy new versions of the application as they become available.

What other components would you want to add to make this production ready?

• Implementing proper error handling and logging to track any issues that occur.
• Implementing a backup and recovery.

How can this application scale with a growing dataset?

• Using a distributed database like Amazon Aurora or Google Spanner for storing the data.
• Using a load balancer to distribute the incoming traffic across multiple instances of the application.
• Using auto-scaling to automatically add or remove instances of the application based on the load.

How can PII be recovered later?

Recovering PII (personally identifiable information) later would likely depend on the transformation technique. Hashing is irreversible. Cryptography way of masking is reversible

What are the assumptions you made?

• The script is running on a local machine and python 3.
• The user has the necessary access credentials to connect to the SQS and PostgreSQL services.
• The data is JSON formatted and follows a specific structure
• The version numbering follows version numbering semantics and hence it is important to preserve major version and drop the 2 other other digits.