DataEng S24: PubSub

[this lab activity references tutorials at cloud.google.com]

Make a copy of this document and use it to record your results. Store a PDF copy of the document in your git repository along with your code before submitting for this week. For your code, you create several publisher/receiver programs or you might make various features within one program. There is no one single correct way to do it. Regardless, store your code in your repository.

The goal for this week is to gain experience and knowledge of using an asynchronous data transport system (Google PubSub). Complete as many of the following exercises as you can. Proceed at a pace that allows you to learn and understand the use of PubSub with python.

Submit: use the in-class activity submission form which is linked from the Materials page on the class website. Submit by 10pm PT this Friday.

A. [MUST] PubSub Tutorial

1. Get your cloud.google.com account up and running
    a. Redeem your GCP coupon
    b. Login to your GCP console
    c. Create a new, separate VM instance
2. Complete this PubSub tutorial: link Note that the tutorial instructs you to destroy your PubSub topic, but you should not destroy your topic just yet. Destroy the topic after you finish the following parts of this in-class assignment.

B. [MUST] Create Sample Data

1. Get data from https://busdata.cs.pdx.edu/api/getBreadCrumbs for two Vehicle IDs from among those that have been assigned to you for the class project.
2. Save this data in a sample file (named bcsample.json)
3. Update the publisher python program that you created in the PubSub tutorial to read and parse your bcsample.json file and send its contents, one record at a time, to the my-topic PubSub topic that you created for the tutorial.
4. Use your receiver python program (from the tutorial) to consume your records.

C. [MUST] PubSub Monitoring

1. Review the PubSub Monitoring tutorial: link and work through the steps listed there. You might need to rerun your publisher and receiver programs multiple times to trigger enough activity to monitor your my-topic effectively.

D. [MUST] PubSub Storage

1. What happens if you run your receiver multiple times while only running the publisher once?

A) If I run the receiver code multiple times while only running the publisher once, I have received the data only once. This is because the publisher sends the data only once, and each time when I run the receiver, it listens for the data being published. Since the data is only published once, each subsequent run of the receiver won't receive any new data.

2. Before the consumer runs, where might the data go, where might it be stored?

A) Before the consumer runs in Google Cloud Platform's Pub/Sub, the data is stored within the Pub/Sub service itself, specifically within the topic it was published to. Pub/Sub retains messages until they're delivered to a subscriber or until they expire. The storage is distributed across servers for durability. Pub/Sub's architecture scales automatically to handle high throughput, distributing data efficiently across clusters for reliable and scalable message processing.

3. Is there a way to determine how much data PubSub is storing for your topic? Do the PubSub monitoring tools help with this?

A) Yes, Google Cloud Pub/Sub's monitoring tools, part of Google Cloud Operations Suite, allow you to track metrics such as undelivered messages and message size. While they don't give precise storage amounts, they help estimate how much data Pub/Sub is storing for your topics and subscriptions.

4. Create a “topic_clean.py” receiver program that reads and discards all records for a given topic. This type of program can be very useful for debugging your project code.

A) We can utilize a cleaner program to read and acknowledge messages in the topic, ensuring they're cleared if needed

E. [SHOULD] Multiple Publishers

1. Clear all data from the topic (run your topic_clean.py program whenever you need to clear your topic)
2. Run two versions of your publisher concurrently, have each of them send all of your sample records. When finished, run your receiver once. Describe the results.

A) The receiver is encountering a problem where duplicate messages are being repeatedly sent, causing a significant increase in message volume within the pipeline. This duplication issue is evident in the Pub/Sub metrics observed in the Google Cloud Platform console, where the message rate appears to fluctuate irregularly.