Ch06

Streaming with Pub/Sub and Dataflow

CHAPTER ASSIGNMENT

Name _____ Class _____ No. _____ Date _____

Answer every question in the space given. Show your working. You may use the chapter page to help you.

Questions from *Data Engineering with Google Cloud Platform*, 1st edition, Adi Wijaya (Packt, 2022), and from the Guru Jazzy chapter page, for classroom use.

Exercises from the book

Ex1 **Exercise – Publishing event streams to cloud Pub/Sub**

In this exercise, we will try to stream data from Pub/Sub publishers. The goal is to create a data pipeline that can stream the data to a BigQuery table, but instead of using a scheduler (as we did in Chapter 4, Building Orchestration for Batch Data Loading Using Cloud Composer), we will submit a Dataflow job that will run as an application to flow data from Pub/Sub to a BigQuery table. In the exercise, we will use the bike-sharing dataset we used in Chapter 3, Building a Data Warehouse in BigQuery. Here are the overall steps in this Pub/Sub section:

(1) Creating a Pub/Sub topic

(2) Creating and running a Pub/Sub publisher using Python

(3) Creating a Pub/Sub subscription

Ex2 **Exercise – Using Cloud Dataflow to stream data from Pub/Sub to GCS**

In this exercise, we will learn how to develop Beam code in Python to create data pipelines. Learning Beam will be challenging at first as you will need to get used to the specific coding pattern, so in this exercise, we will start with a HelloWorld-level code. But the benefit of using Beam is it's a general framework. Generally, you can create a batch or streaming pipeline with similar code. You can also run using different runners. In this exercise, we will use Direct Runner and Dataflow. As a summary, here are the steps:

(a) Creating a HelloWorld application using Apache Beam

(b) As a practice exercise, let's try using ParDo. For ParDo, we need to follow the format by declaring the Split function as a Python class. In the Split class, we need a process function that accepts elements. Notice that the function returns an array instead of a single object.

(c) Creating a Dataflow streaming job without aggregation

(d) Creating a Dataflow streaming job with aggregation

Ex3 Creating a streaming job with aggregation

In this final section, we want to aggregate the streaming data. For example, let's sum up the duration seconds from the bike-trip data for each starting station ID.

You can use the `beam_stream_aggr.py` code example to check the final code. For our practice, continue by trying to do it yourself. To do that, go back to your Beam streaming pipeline code. Go to the Parse JSON step. This time, we want to define a fixed time window of 60 seconds. In the later steps, we will sum up the `duration_sec` value.

Sorting activities from the chapter page

S1 Which fits?

Options: **Batch** · **Streaming**

No.	Scenario	Answer
1	The CEO wants last month’s revenue, calculated once from purchase history.	
2	A card transaction must be checked for fraud the moment it happens.	
3	A sales dashboard should tick up second by second during a flash sale.	
4	Daily fact tables are rebuilt each night for the operations team.	

S2 Name the piece

Options: **Publisher** · **Topic** · **Subscription** · **Subscriber**

No.	Scenario	Answer
1	The application code that sends each bike-trip JSON message.	
2	The named store that holds the messages, like a table holds rows.	
3	An entity that has an interest in a topic. Two of them on one topic each receive identical messages.	
4	One of several workers that share the load of a single subscription.	

S3 Subscription settings that matter

Options: **Pull** · **Push**

No.	Scenario	Answer
1	Dataflow, using the Pub/Sub client, will read messages for a data pipeline.	
2	The consumer cannot use the Pub/Sub client and instead exposes an HTTP endpoint.	

S4 Which runner?

Options: **DirectRunner** · **DataflowRunner**

No.	Scenario	Answer
1	Quickly test a change on the Cloud Shell machine before deploying.	
2	Run the pipeline in production with autoscaling and a monitoring graph.	

Quiz from the chapter page: Check yourself

1 Why are most data pipelines still batch?

A. Streaming is impossible on GCP

B. Users usually ask for figures over a period, and batch is much simpler to control

C. Batch is always faster

D. Streaming cannot write to BigQuery

---

2 A topic has two subscriptions. What does each receive?

A. Half of the messages each

B. Identical copies of all messages

C. Only unacked messages

D. Nothing until you enable both

---

3 What happens to a message that is delivered but never acknowledged?

A. It is deleted immediately

B. It is sent again until it is acked or expires

C. It moves to another topic

D. It becomes a BigQuery row

---

4 Why aggregate a stream inside time windows?

A. Streams have no timestamps

B. A stream never ends, so a sum needs a boundary in time to know when to emit a result

C. Windows make Pub/Sub cheaper

D. BigQuery requires them

---

5 A streaming Dataflow job you started is still running after your exercise. What is the consequence?

A. Nothing, it stops itself

B. It keeps billing for worker time until you stop it

C. It deletes your topic

D. It switches to batch mode

---

6 How is Dataflow different from Spark on Dataproc, from a user's point of view?

A. It cannot process big data

B. It is serverless: you submit a pipeline and never set up a cluster

C. It only runs Java

D. It stores data in HDFS

End of assignment · Check your answers against the chapter at gurujazzy.com/subjects/gcp-data-engineering/chapter-06-streaming-with-pubsub-and-dataflow/