Hands-on activity

Introduction

To get you up and running with Redis Streams, let's revisit the Python temperature monitoring example that we presented in section 2.5.

The code consists of:

- A producer, which writes temperature data to a Redis stream
- A consumer whose job it is to simulate writing data from the stream to a data warehouse
- A second consumer that consumes the stream to calculate a rolling average temperature for use by a real-time display

All three components interact with a stream named **stream:weather** in Redis.

In this exercise, we'll use your environment and run each component to get a feel for how they interact with each other.

The Producer

This exercise assumes that you have cloned the course GitHub repository and followed the setup instructions. Note that you will need to activate your Python virtual environment and set any required environment variables for each new terminal session that you use here.

Now that you have your environment up and running, let's try starting each of the components, beginning with the producer. Using one terminal window, type:

```
cd src/intro   python producer.py
```

You should see the producer start up, and begin adding temperature messages to the stream. The output should look like this:

```
$ python producer.py Wrote {"postal_code": 94016, "temp_f": 51} with ID 1554921766941-0 Wrote {"postal_code": 94016, "temp_f": 50} with ID 1554921767946-0 Wrote {"postal_code": 94016, "temp_f": 51} with ID 1554921768948-0 Wrote {"postal_code": 94016, "temp_f": 52} with ID 1554921769949-0 Wrote {"postal_code": 94016, "temp_f": 51} with ID 1554921770953-0
```

Leave the producer running.

The Consumers

Next, we'll start the both the data warehouse and rolling average consumer processes. Go ahead and enter these commands into a second terminal window:

```
cd src/intro ./run-consumers.sh
```

The data warehouse consumer will start reading from the stream and produce output similar to the following:

$ ./run-consumers.sh Wrote [["stream:weather", [["1554922470338-0", {"postal_code": "94016", "temp_f": "34"}]]]] to data warehouse.

Wrote [["stream:weather", [["1554922471341-0", {"postal_code": "94016", "temp_f": "33"}]]]] to data warehouse.

Wrote [["stream:weather", [["1554922472342-0", {"postal_code": "94016", "temp_f": "32"}]]]] to data warehouse.

The rolling average consumer will also start, and its log output will be in yellow. You can expect to see output similar to that shown below.

Processing: [["stream:weather", [["1554922911708-0", {"postal_code": "94016", "temp_f": "30"}]]]] Rolling Average: 30.0

Processing: [["stream:weather", [["1554922912709-0", {"postal_code": "94016", "temp_f": "29"}]]]] Rolling Average: 29.5

Processing: [["stream:weather", [["1554922913711-0", {"postal_code": "94016", "temp_f": "30"}]]]] Rolling Average: 29.666666666666668

Processing: [["stream:weather", [["1554922914714-0", {"postal_code": "94016", "temp_f": "29"}]]]] Rolling Average: 29.5

Processing: [["stream:weather", [["1554922915718-0", {"postal_code": "94016", "temp_f": "30"}]]]] Rolling Average: 29.6

Stopping the System

To stop the producer, click on the its terminal window in your lab environment and press **Ctrl + C**.

Stop both consumers by using the idle terminal window to run a script:

./stop-consumers.sh

Need Help?

If you get stuck or have any questions, please don't hesitate to reach out on Discord.

← Previous

Next →

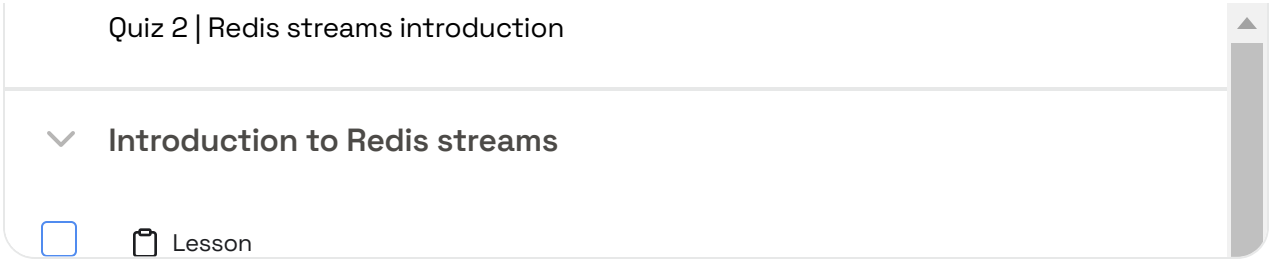
Modules

Overview of Redis streams learning path

- [ ] Lesson
  Welcome to Redis streams learning path
- [ ] Lesson
  Environment setup

Introduction to distributed systems

- [ ] Lesson
  Introduction to distributed systems
- [ ] Assessment
  Quiz 1 | Redis streams introduction
- [ ] Lesson
  Streams processing
- [ ] Lesson
  Stream pipelines
- [ ] Assessment


Quiz 2 | Redis streams introduction
Introduction to Redis streams
Lesson