Building an IoT Analytics Pipeline on Google Cloud Platform

## Overview

The term **Internet of Things (IoT)** refers to the interconnection of physical devices with the global Internet. These devices are equipped with sensors and networking hardware, and each is globally identifiable. Taken together, these capabilities afford rich data about items in the physical world.

[Cloud IoT Core](https://cloud.google.com/iot-core/) is a fully managed service that allows you to easily and securely connect, manage, and ingest data from millions of globally dispersed devices. The service connects IoT devices that use the standard Message Queue Telemetry Transport (MQTT) protocol to other Google Cloud Platform data services.

Cloud IoT Core has two main components:

* A **device manager** for registering devices with the service, so you can then monitor and configure them.
* A **protocol bridge** that supports MQTT, which devices can use to connect to the Google Cloud Platform.

## Create a Cloud Pub/Sub topic

[Cloud Pub/Sub](https://cloud.google.com/pubsub/) is an asynchronous global messaging service. By decoupling senders and receivers, it allows for secure and highly available communication between independently written applications. Cloud Pub/Sub delivers low-latency, durable messaging.

In Cloud Pub/Sub, publisher applications and subscriber applications connect with one another through the use of a shared string called a **topic**. A publisher application creates and sends messages to a topic. Subscriber applications create a subscription to a topic to receive messages from it.

In an IoT solution built with Cloud IoT Core, device telemetry data is forwarded to a Cloud Pub/Sub topic.

To define a new Cloud Pub/Sub topic:

1. In the GCP Console, go to **Navigation menu** > **Pub/Sub** > **Topics**.
2. Click **Create a Topic**. The **Create a topic** dialog shows you a partial URL path.

**Note:** If you see qwiklabs-resources as your project name, cancel the dialog and return to the Cloud Platform console. Use the menu to the right of the Google Cloud Platform logo to select the correct project. Then return to this step.

1. Add this string as your topic name:

iotlab

1. Then click **Create**.

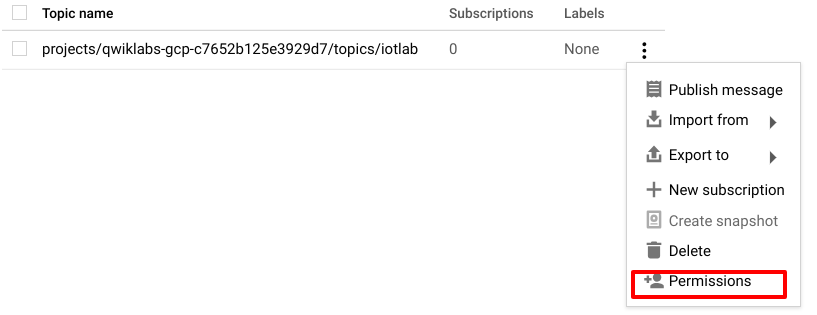
### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully created a Cloud Pub/Sub topic, you will see an assessment score.

Create a Cloud Pub/Sub Topic

Check my progress

1. In the list of topics, you will see a new topic whose partial URL ends in iotlab. Click the three-dot icon at the right edge of its row to open the context menu. Choose **Permissions**.



1. In the **Permissions** dialogue, click **Add members** and copy the below member as **New members**:

cloud-iot@system.gserviceaccount.com

1. From the Role menu, give the new member the **Pub/Sub** > **Pub/Sub Publisher** role.
2. Click **Save**.

### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully added IAM binding policy to Pub/Sub topic, you will see an assessment score.

Add IAM binding policy to Pub/Sub topic

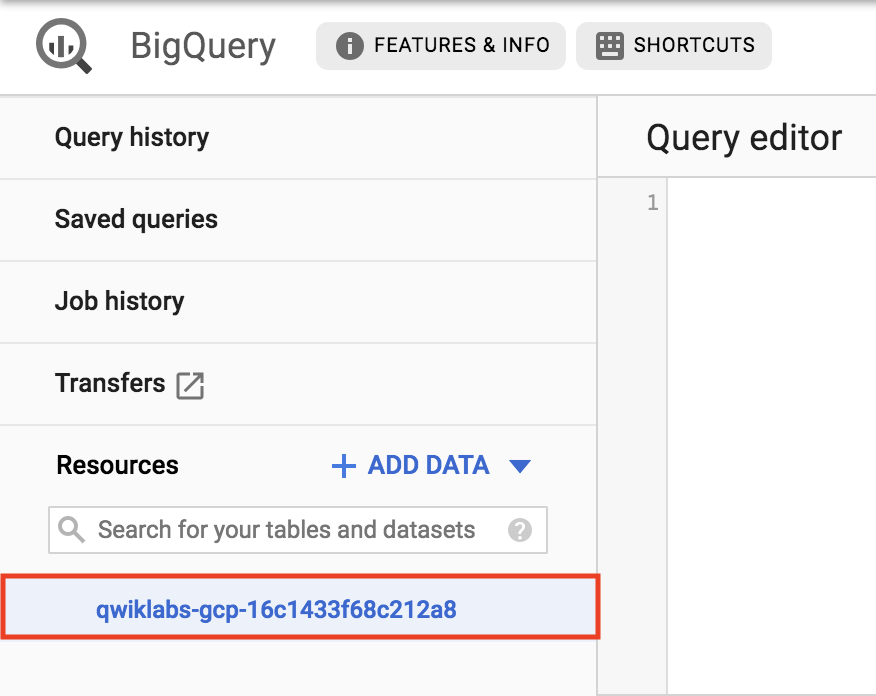
Check my progress

## Create a BigQuery dataset

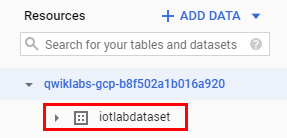
[BigQuery](https://cloud.google.com/bigquery/) is a serverless data warehouse. Tables in BigQuery are organized into datasets. In this lab, messages published into Pub/Sub will be aggregated and stored in BigQuery.

To create a new BigQuery dataset:

1. In the GCP Console, go to **Navigation menu** > **BigQuery**.
2. Click **Done**.
3. Click on your GCP Project ID from the left-hand menu:



1. On the right-hand side of the console underneath the query editor click **CREATE DATASET**.
2. Name the dataset **iotlabdataset**, leave all the other fields the way they are, and click **Create dataset**.
3. You should see your newly created dataset under your project:



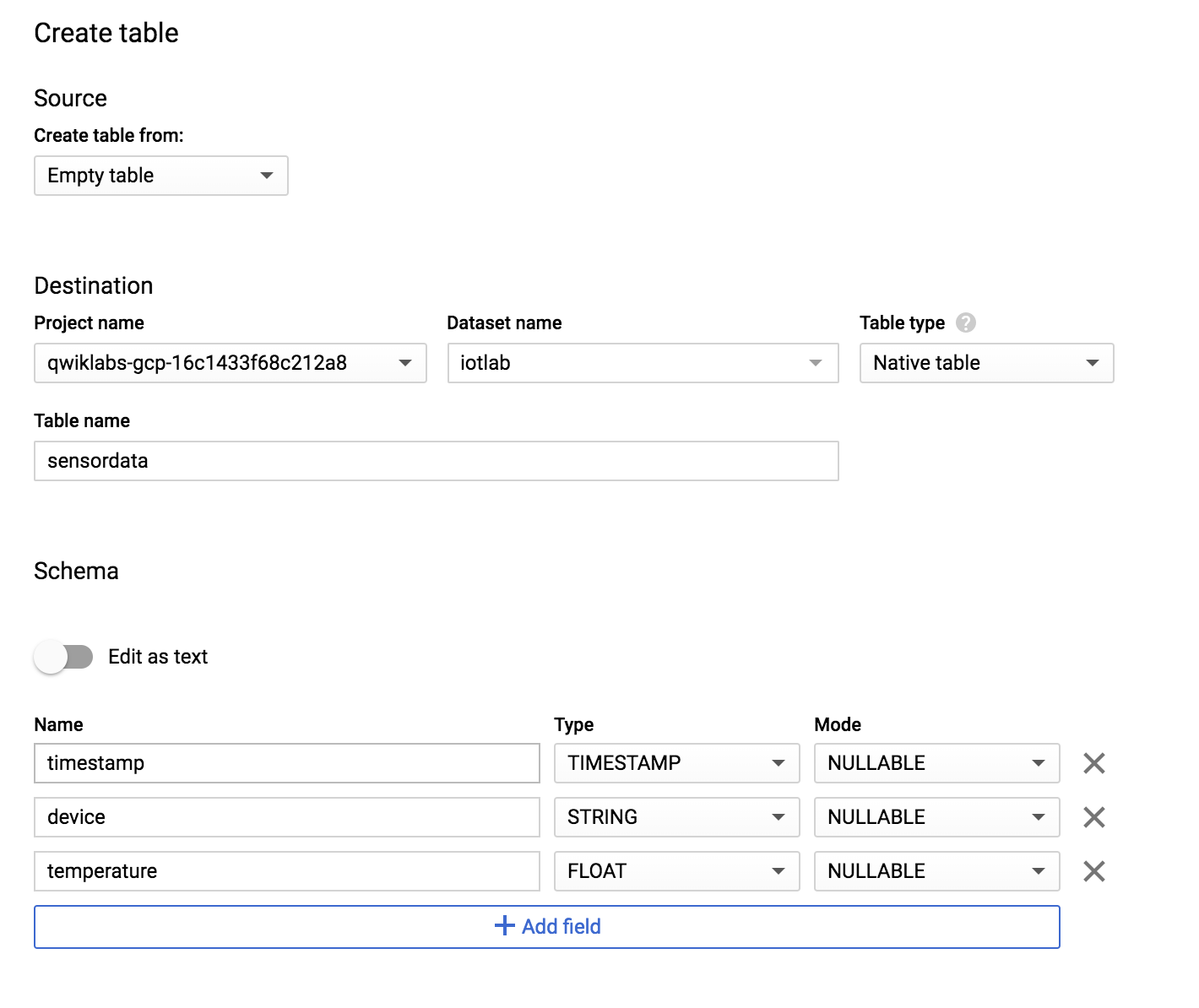
### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully created a BigQuery dataset, you will see an assessment score.

Create a BigQuery dataset

Check my progress

1. Click on your dataset, then on the right-hand side of the console click **+ CREATE TABLE**.
2. Ensure that the source field is set to **Empty table**.
3. In the **Destination** section's **Table name** field, enter "sensordata".
4. In the **Schema** section, click the **+ Add field** button and add the following fields:
   * **timestamp**, set the field's **Type** to **TIMESTAMP**.
   * **device**, set the field's **Type** to **STRING**.
   * **temperature**, set the field's **Type** to **FLOAT**.
5. Leave the other defaults unmodified. Click **Create table**.



### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully created an empty table in BigQuery Dataset, you will see an assessment score.

Create an empty table in BigQuery Dataset

Check my progress

## Create a cloud storage bucket

[Cloud Storage](https://cloud.google.com/storage/) allows world-wide storage and retrieval of any amount of data at any time. You can use Cloud Storage for a range of scenarios including serving website content, storing data for archival and disaster recovery, or distributing large data objects to users via direct download.

For this lab Cloud Storage will provide working space for your Cloud Dataflow pipeline.

1. In the GCP Console, go to **Navigation menu** > **Storage**.
2. Click **CREATE BUCKET**.
3. For **Name**, use your GCP project ID then add "-bucket".
4. For **Default storage class**, click **Multi-regional** if it is not already selected.
5. For **Location**, choose the selection closest to you.
6. Click **Create**.

### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully created a Cloud Storage bucket, you will see an assessment score.

Create a Cloud Storage Bucket

Check my progress

## Set up a Cloud Dataflow Pipeline

[Cloud Dataflow](https://cloud.google.com/dataflow/) is a serverless way to carry out data analysis. In this lab, you will set up a streaming data pipeline to read sensor data from Pub/Sub, compute the maximum temperature within a time window, and write this out to BigQuery.

1. In the GCP Console, go to **Navigation menu** > **Dataflow**.
2. In the top menu bar, click **CREATE JOB FROM TEMPLATE**.
3. In the job-creation dialog, for **Job name**, enter "iotlabflow".
4. For **Cloud Dataflow template**, choose **Cloud PubSub Topic to BigQuery**. When you choose this template, the form updates to review new fields below.
5. For **Cloud Dataflow Regional Endpoint**, choose the region as **us-central1**.
6. For **Cloud Pub/Sub input topic**, enter projects/ followed by your GCP project ID then add /topics/iotlab. The resulting string will look like this: projects/qwiklabs-gcp-d2e509fed105b3ed/topics/iotlab
7. The **BigQuery output table** takes the form of GCP project ID:dataset.table (:iotlabdataset.sensordata). The resulting string will look like this: qwiklabs-gcp-d2e509fed105b3ed:iotlabdataset.sensordata
8. For **Temporary location**, enter gs:// followed by your GCS bucket name (should be your GCP project ID if you followed the instructions) then /tmp/. The resulting string will look like this: gs://qwiklabs-gcp-d2e509fed105b3ed-bucket/tmp/
9. Click **Optional parameters**.
10. For Max workers, enter **2**.
11. For Machine type, enter **n1-standard-1**.
12. Click **Run job**.

A new streaming job is started. You can now see a visual representation of the data pipeline.

### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully set up a Cloud Dataflow Pipeline, you will see an assessment score.

Set up a Cloud Dataflow Pipeline (region: us-central1)

Check my progress

## Prepare your compute engine VM

In your project, a pre-provisioned VM instance named iot-device-simulator will let you run instances of a Python script that emulate an MQTT-connected IoT device. Before you emulate the devices, you will also use this VM instance to populate your Cloud IoT Core device registry.

To connect to the iot-device-simulator VM instance:

1. In the GCP Console, go to **Navigation menu** > **Compute Engine** > **VM Instances**. You'll see your VM instance listed as iot-device-simulator.
2. Click the **SSH** drop-down arrow and select **Open in browser window**.
3. In your SSH session on the **iot-device-simulator** VM instance, enter this command to remove the default Google Cloud Platform SDK installation. (In subsequent steps, you will install the latest version, including the beta component.)

sudo apt-get remove google-cloud-sdk -y

1. Now install the latest version of the Google Cloud Platform SDK and accept all defaults:

curl https://sdk.cloud.google.com | bash

1. End your ssh session on the **iot-device-simulator** VM instance:

exit

1. Start another SSH session on the iot-device-simulator VM instance.
2. Initialize the **gcloud** SDK.

gcloud init

If you get the error message "Command not found," you might have forgotten to exit your previous SSH session and start a new one.

If you are asked whether to authenticate with an @developer.gserviceaccount.com account or to log in with a new account, choose **log in with a new account**.

If you are asked "Are you sure you want to authenticate with your personal account? Do you want to continue (Y/n)?" enter Y.

1. Click on the URL shown to open a new browser window that displays a verification code.
2. Copy the verification code and paste it in response to the "Enter verification code:" prompt, the press **Enter**.
3. In response to "Pick cloud project to use," pick the GCP project that Qwiklabs created for you.
4. Enter this command to make sure that the components of the SDK are up to date:

gcloud components update

1. Enter this command to install the beta components:

gcloud components install beta

1. Enter this command to update the system's information about Debian Linux package repositories:

sudo apt-get update

1. Enter this command to make sure that various required software packages are installed:

sudo apt-get install python-pip openssl git -y

1. Use **pip** to add needed Python components:

sudo pip install pyjwt paho-mqtt cryptography

1. Enter this command to add data to analyze during this lab:

git clone http://github.com/GoogleCloudPlatform/training-data-analyst

## Create a registry for IoT devices

To register devices, you must create a registry for the devices. The registry is a point of control for devices.

To create the registry:

1. In your SSH session on the **iot-device-simulator** VM instance, run the following, adding your project ID as the value for **PROJECT\_ID**:

export PROJECT\_ID=

Your completed command will look like this: export PROJECT\_ID=qwiklabs-gcp-d2e509fed105b3ed

1. You must choose a region for your IoT registry. At this time, these regions are supported:

* us-central1
* europe-west1
* asia-east1

Choose the region that is closest to you. To set an environment variable containing your preferred region, enter this command followed by the region name:

export MY\_REGION=

Your completed command will look like this: export MY\_REGION=us-central1.

1. Enter this command to create the device registry:

gcloud beta iot registries create iotlab-registry \

--project=$PROJECT\_ID \

--region=$MY\_REGION \

--event-notification-config=topic=projects/$PROJECT\_ID/topics/iotlab

### **Test Completed Task**

Click **Check my progress** to verify your performed task. If you have successfully createa a Registry for IoT Devices, you will see an assessment score.

Create a Registry for IoT Devices

Check my progress

## Create a Cryptographic Keypair

To allow IoT devices to connect securely to Cloud IoT Core, you must create a cryptographic keypair.

In your SSH session on the **iot-device-simulator** VM instance, enter these commands to create the keypair in the appropriate directory:

cd $HOME/training-data-analyst/quests/iotlab/

openssl req -x509 -newkey rsa:2048 -keyout rsa\_private.pem \

-nodes -out rsa\_cert.pem -subj "/CN=unused"

This openssl command creates an RSA cryptographic keypair and writes it to a file called rsa\_private.pem.

## Add simulated devices to the registry

For a device to be able to connect to Cloud IoT Core, it must first be added to the registry.

1. In your SSH session on the **iot-device-simulator** VM instance, enter this command to create a device called temp-sensor-buenos-aires:

gcloud beta iot devices create temp-sensor-buenos-aires \

--project=$PROJECT\_ID \

--region=$MY\_REGION \

--registry=iotlab-registry \

--public-key path=rsa\_cert.pem,type=rs256

1. Enter this command to create a device called temp-sensor-istanbul:

gcloud beta iot devices create temp-sensor-istanbul \

--project=$PROJECT\_ID \

--region=$MY\_REGION \

--registry=iotlab-registry \

--public-key path=rsa\_cert.pem,type=rs256

### **Test Completed Task**

Click **Check my progress** to verify your performed task.If you have successfully added Simulated Devices to the Registry, you will see an assessment score.

Add Simulated Devices to the Registry

Check my progress

## Run simulated devices

1. In your SSH session on the **iot-device-simulator** VM instance, enter these commands to download the CA root certificates from pki.google.com to the appropriate directory:

cd $HOME/training-data-analyst/quests/iotlab/

wget https://pki.google.com/roots.pem

1. Enter this command to run the first simulated device:

python cloudiot\_mqtt\_example\_json.py \

--project\_id=$PROJECT\_ID \

--cloud\_region=$MY\_REGION \

--registry\_id=iotlab-registry \

--device\_id=temp-sensor-buenos-aires \

--private\_key\_file=rsa\_private.pem \

--message\_type=event \

--algorithm=RS256 > buenos-aires-log.txt 2>&1 &

It will continue to run in the background.

1. Enter this command to run the second simulated device:

python cloudiot\_mqtt\_example\_json.py \

--project\_id=$PROJECT\_ID \

--cloud\_region=$MY\_REGION \

--registry\_id=iotlab-registry \

--device\_id=temp-sensor-istanbul \

--private\_key\_file=rsa\_private.pem \

--message\_type=event \

--algorithm=RS256

Telemetry data will flow from the simulated devices through Cloud IoT Core to your Cloud Pub/Sub topic. In turn, your Dataflow job will read messages from your Pub/Sub topic and write their contents to your BigQuery table.

## Analyze the Sensor Data Using BigQuery

To analyze the data as it is streaming:

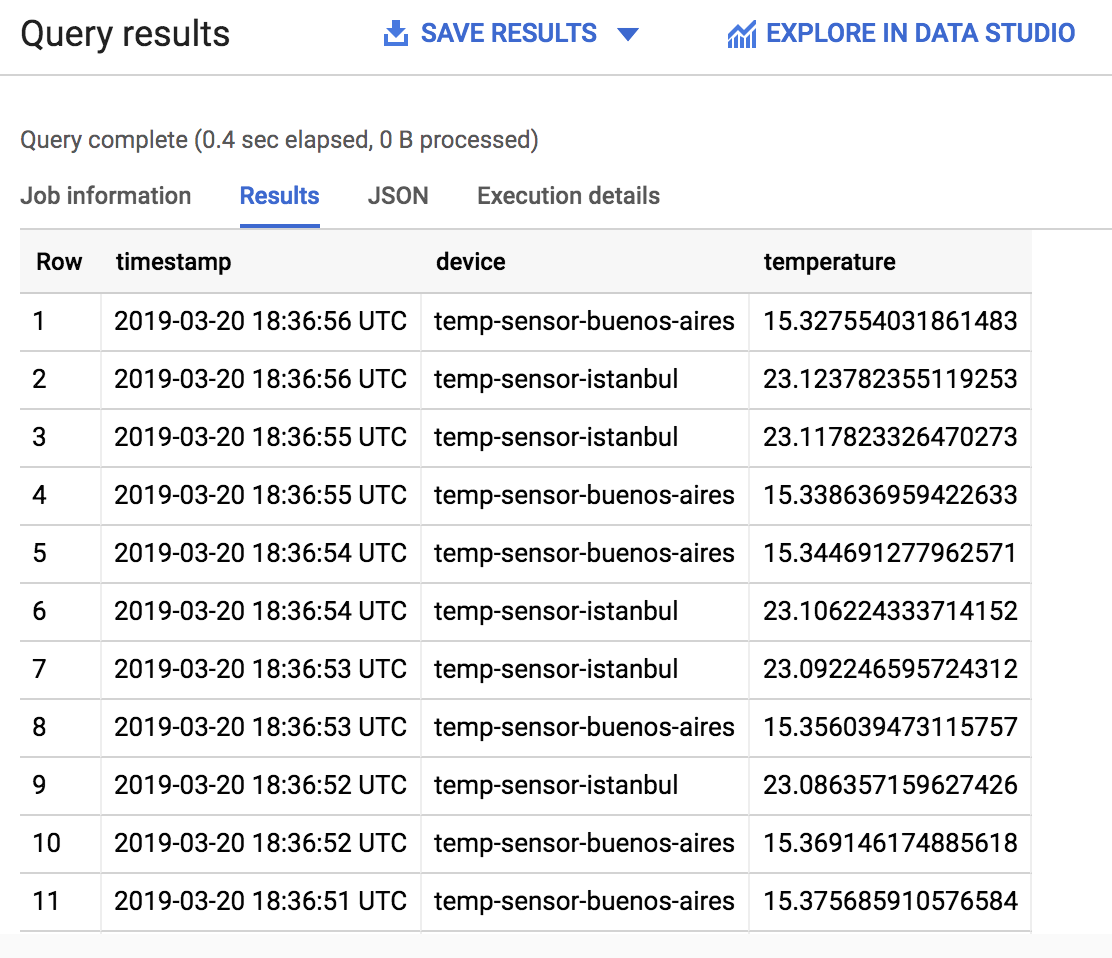
1. In the GCP Console, open the **Navigation menu** and select **BigQuery**.
2. Enter the following query in the Query editor and click **RUN**:

SELECT timestamp, device, temperature from iotlabdataset.sensordata

ORDER BY timestamp DESC

LIMIT 100

1. You should receive a similar output:



1. Browse the Results. What is the temperature trend at each of the locations?

## Test your Understanding

Below are multiple-choice questions to reinforce your understanding of this lab's concepts. Answer them to the best of your abilities.

Cloud IoT Core supports two protocols for device connection and communication:



HTTP



SMTP



IP



MQTT



TCP

Submit

When you create a device within a registry, you define the device as a Cloud IoT Core resource.

True

False