

Serverless iot data processing

Steps:

- Getting setup
- Create a bigquery table
- Create a pub/sub topic
- Create a Cloud function
- Setup the iot hardware


Getting setup

Self-paced environment setup

If we don't already have a Google account (Gmail or G Suite), we must create one. Regardless of whether we already have a Google account or not, make sure to take advantage of the \$300 free trial!


Sign-in to Google Cloud Platform console (console.cloud.google.com). we can use the default project ("My First Project") for this lab or we can choose to create a new project. If we'd like to create a new project, we can use the the Manage resources page. The project ID needs to be a unique name across all Google Cloud projects (the one shown below has already been taken and won't work for we). Take note of wer project ID (i.e. wer project ID will be ____) as it will be needed later.

New Project

 You have 11 projects remaining in your quota. [Learn more.](#)

Project name 


lot2Analytics


Your project ID will be `iot2analytics`  [Edit](#)


Create

Cancel

(If we forget wer project ID, it is located in the Cloud Console in the Home area.)

 Home


 Cloud Launcher


 Billing


API

APIs & Services

>


 Support >

 IAM & admin >

 Getting started

DASHBOARD

ACTIVITY

 Project info

Project name

lot2Analytics

Project ID

iot2analytics

Project number

902178000722

→

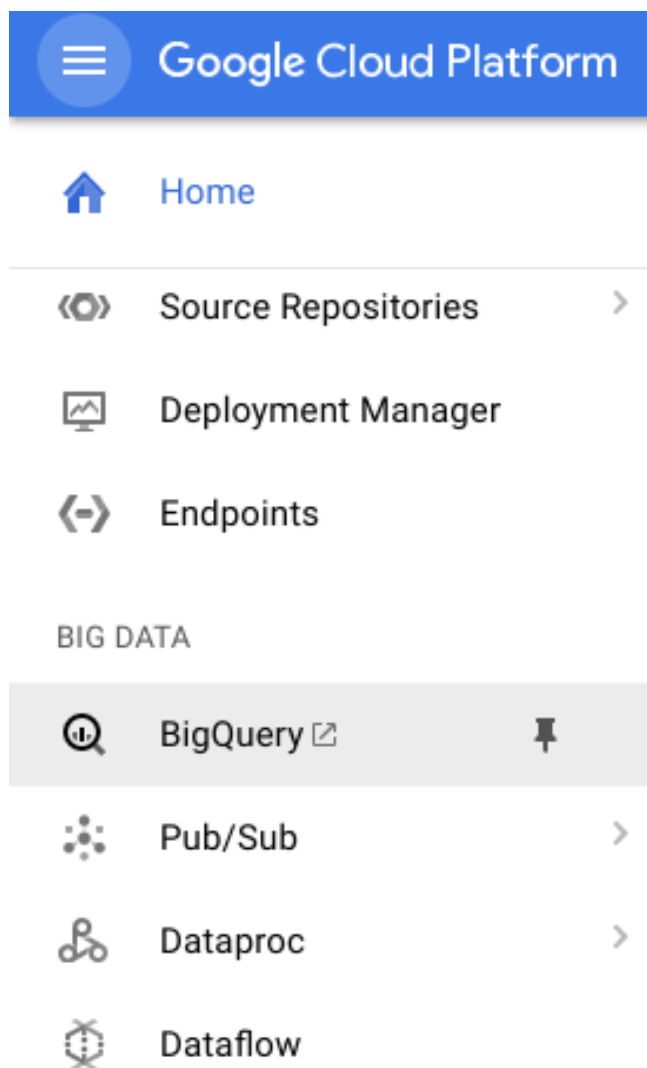
Go to project settings

Running through this codelab shouldn't cost more than a few dollars, but it could be more if we decide to use more resources or if we leave them running. Make sure to go through the Cleanup section at the end of the codelab.

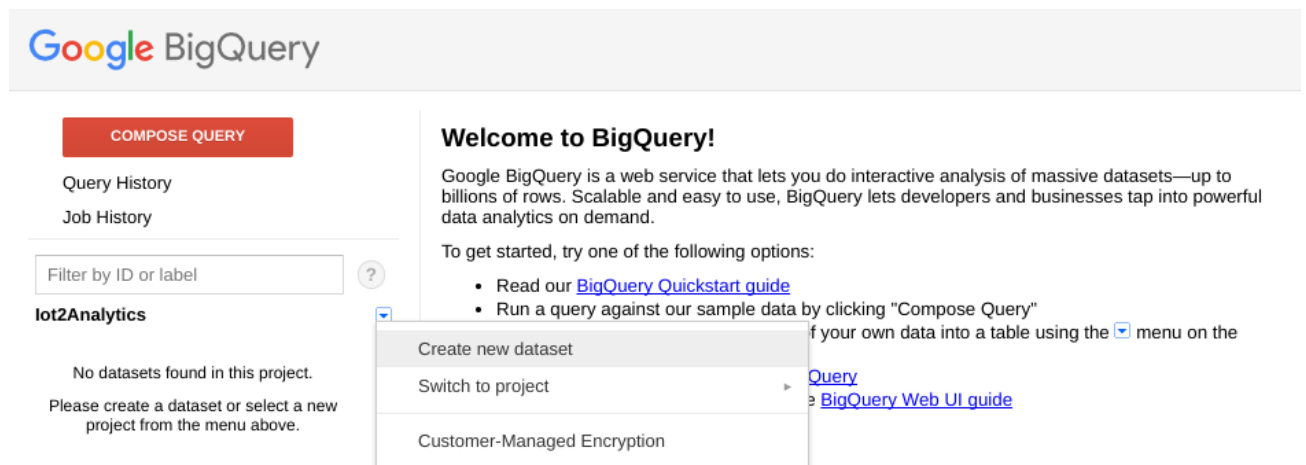
Create a bigquery table

BigQuery is a serverless, highly scalable, low cost enterprise data warehouse and will be an ideal option to store data being streamed from IoT devices while also allowing an analytics dashboard to query the information.

Let's create a table that will hold all of the IoT weather data. Select BigQuery from the Cloud console. **This will open BigQuery in a new window** (don't close the original window as we'll need to access it again).



Click on the down arrow icon next to wer project name and then select "Create new dataset"



Enter "weatherData" for the Dataset, select a location where it will be stored and Click "OK"

Create Dataset ✕

Dataset ID

weatherData ?

Data location

US ?

Data expiration

☒ Never ☐ In days.

OK

Cancel

Click the "+" sign next to wer Dataset to create a new table

Google BigQuery

COMPOSE QUERY

Query History

Job History

Filter by ID or label ?

lot2Analytics ▼

▶ weatherData

+

▼

For **Source Data**, select **Create empty table**. For **Destination table name**, enter weatherDataTable. Under **Schema**, click the **Add Field** button until there are a total of 9 fields. Fill in the fields as shown below

making sure to also select the appropriate **Type** for each field. When everything is complete, click on the **Create Table** button.

(Note:Any spelling errors for the field names or the table name can cause issues when Cloud Functions attempts to add data later in this codelab.)

Create Table

Source Data ☐ Create from source ☒ Create empty table

Destination Table

Table name

weatl . weatherDataTable ?

Table type

Native table ?

Schema

Name	Type	Mode	
sensorID	STRING	NULLABLE	×
timecollected	TIMESTAMP	NULLABLE	×
zipcode	INTEGER	NULLABLE	×
latitude	FLOAT	NULLABLE	×
longitude	FLOAT	NULLABLE	×
temperature	FLOAT	NULLABLE	×
humidity	FLOAT	NULLABLE	×
dewpoint	FLOAT	NULLABLE	×
pressure	FLOAT	NULLABLE	×
Add Field			Edit as Text

Options

Partitioning

None

Encryption Type

Default

Create Table

We get output like this

Table Details: weatherDataTable

Schema	Details	Preview	
sensorID	STRING	NULLABLE	Describe this field...
timecollected	TIMESTAMP	NULLABLE	Describe this field...
zipcode	INTEGER	NULLABLE	Describe this field...
latitude	FLOAT	NULLABLE	Describe this field...
longitude	FLOAT	NULLABLE	Describe this field...
temperature	FLOAT	NULLABLE	Describe this field...
humidity	FLOAT	NULLABLE	Describe this field...
dewpoint	FLOAT	NULLABLE	Describe this field...
pressure	FLOAT	NULLABLE	Describe this field...
Add New Fields			

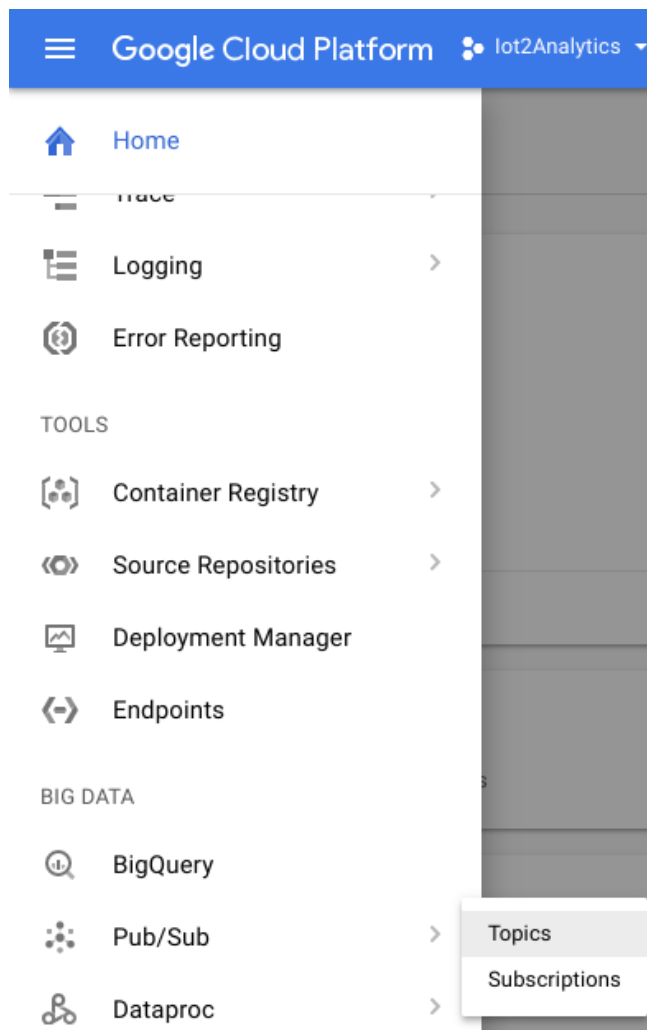
now we have a data warehouse setup to receive our weather data.

Create a Pub/Sub topic

Cloud Pub/Sub is a simple, reliable, scalable foundation for stream analytics and event-driven computing systems. As a result, it is perfect for handling incoming IoT messages and then allowing downstream systems to process them.

If you are still in the window for BigQuery, switch back to the Cloud Console. If you closed the Cloud Console, go to <https://console.cloud.google.com>


From the Cloud Console, select Pub/Sub and then Topics.



If we see an Enable API prompt, click the Enable API button.

Big Data Pub/Sub

Reliable real-time messaging

Connect your services with reliable, many-to-many, asynchronous messaging hosted on Google's infrastructure. To get started, create a topic for posting asynchronous messages to multiple subscribers [Learn more](#) 

Enable API

Click on the Create a topic button

Big Data

Pub/Sub

Reliable real-time messaging

Connect your services with reliable, many-to-many, asynchronous messaging hosted on Google's infrastructure. To get started, create a topic for posting asynchronous messages to multiple subscribers [Learn more](#) ↗

Create a topic

Enter "weatherdata" as the topic name and the click Create

Create a topic


A topic forwards messages from publishers to subscribers.


Name ?


projects/iot2analytics/topics/ weatherdata

CANCEL CREATE

we should see the newly created topic


 Pub/Sub


 Topics

 Subscriptions

Topics

+ CREATE TOPIC

 DELETE




<input type="checkbox"/> Topic name	Subscriptions	
<input type="checkbox"/> projects/iot2analytics/topics/weatherdata	0	


now we have a Pub/Sub topic to both publish IoT messages to and to allow other processes to access those messages.


Secure publishing to the topic


If plan to publish messages to the Pub/Sub topic from resources outside of your Google Cloud Console (e.g. an IoT sensor), it will be necessary to more tightly control access using a service account and to ensure the security of the connection by creating a trust certificate.


From the Cloud Console, select IAM & Admin and then Service accounts


 Google Cloud Platform  lot2Analytics 

 Home

 APIs & Services

 Support

 IAM & admin

 Getting started

COMPUTE

Topics

+ CRE

☐ Topic name

IAM

Identity

Quotas

Service accounts

Click on the Create service account button

Permissions

Service account management

A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. [Learn more](#)

Create service account

In the Role dropdown, select the Pub/Sub Publisher role

Create service account

Service account name ?

Service account ID

@iot2analytics.iam.gs

☐ Furnish a new private key

Downloads a file that contains the private key. Store the file securely. The private key can't be recovered if lost.

☐ Enable G Suite Domain-wide Delegation

Allows this service account to be authorized to access all domain resources without manual authorization on their part. [Learn more](#)

Role ?

Pub/Sub Publisher

Selected

✓ Pub/Sub Publisher

Cloud Scheduler

Cloud Security Scanner

Cloud Tasks

Cloud Trace

Datastore

Error Reporting

IAM

Logging

Monitoring

Organization Policy

Pub/Sub

Reserve Partner

Resource Manager

Roles

Pub/Sub Admin

Pub/Sub Editor

✓ Pub/Sub Publisher

Pub/Sub Subscriber

Pub/Sub Viewer

Manage roles

Publish messages to a topic.

Type in a service account name (iotWeatherPublisher), check the Furnish a new private key checkbox, make sure that Key type is set to JSON and click on "Create"

Create service account

Service account name ?

iotWeatherPublisher

Role ?

Pub/Sub Publisher ▼

Service account ID

iotweatherpublisher

@iot2analytics.iam.gserviceaccount.com



☒ Furnish a new private key

Downloads a file that contains the private key. Store the file securely because this key can't be recovered if lost.

Key type

☒ JSON

Recommended

☐ P12

For backward compatibility with code using the P12 format

☐ Enable G Suite Domain-wide Delegation

Allows this service account to be authorized to access all users' data on a G Suite domain without manual authorization on their part. [Learn more](#)

CANCEL CREATE

The security key will download automatically. There is only one key, so it is important to not lose it. Click Close.








Service account and key created

Service account `iotWeatherPublisher` created. The account's private key `lot2Analytics-a68b9ef614d3.json` saved on your computer.

 `lot2Analytics-a68b9ef614d3.json` allows access to your cloud resources, so store it securely. [Learn more](#)

CLOSE

We should see that a service account has been created and that there is a Key ID associated with it.

Service Accounts					
<div><div> CREATE SERVICE ACCOUNT</div><div> DELETE</div><div> PERMISSIONS</div></div>					
Service accounts for project "lot2Analytics"					
A service account represents a Google Cloud service identity, such as code running on Compute Engine VMs, App Engine apps, or systems running outside Google. Learn more					
<div><div> Find a service account</div></div>					
<input type="checkbox"/>	Service account name ^	Service account ID	Key ID	Key creation date	Options
<input type="checkbox"/>	 <code>iotWeatherPublisher</code>	<code>iotweatherpublisher@lot2analytics.iam.gserviceaccount.com</code>	<code>a68b9ef614d3430cc1c587bd35b2d7a6b5f2691b</code>	 Jan 29, 2018	

In order to easily access the key later, we will store it in Google Cloud Storage. From the Cloud Console, select Storage and then Browser.



Home



Datastore



Storage



SQL



Spanner

Browser

Transfer

Transfer Appliance

Settings

Click the Create Bucket button

Cloud Storage Buckets

Cloud Storage lets you store unstructured objects in containers called buckets. You can serve static data directly from Cloud Storage, or you can use it to store data for other Google Cloud Platform services.

Create bucket

or

Take the quickstart

Choose a name for the storage bucket (it must be a name that is globally unique across all of Google Cloud) and click on the Create button

Create a bucket

Name ?

Must be unique across Cloud Storage. **Privacy:** Do not include sensitive information in your bucket name. Others can discover your bucket name if it matches a name they're trying to use.

keystore-iot2analytics

Default storage class ?

☒ Multi-Regional

Use to stream videos and host hot web content.
Best for data accessed frequently around the world.

☐ Regional

Use to store data and run data analytics.
Best for data accessed frequently in one part of the world.

☐ Nearline

Use to store rarely accessed documents.
Best for data accessed less than once per month.

☐ Coldline

Use to store very rarely accessed documents.
Best for data accessed less than once per year.

Multi-Regional location

Redundant across 2+ regions within your selected location.

United States

Specify labels

Create

Cancel

Locate the security key that was automatically downloaded and either drag/drop or upload it into the storage bucket

The image shows the Google Cloud Storage browser interface. At the top, there's a navigation bar with buttons: 'Browser', 'UPLOAD FILES', 'UPLOAD FOLDER', 'CREATE FOLDER', 'REFRESH', 'SHARE PUBLICLY', and 'DELETE'. Below this is a search bar with the placeholder text 'Filter by prefix...'. The main content area shows the path 'Buckets / keystore-iot2analytics'. A message states: 'There are no live objects in this bucket. If you have object versioning enabled, this bucket may contain archived versions of objects, which aren't visible in the console. You can list archived object versions using gsutil or the APIs.' In the center, there's a large light gray circle containing a white document icon with a plus sign. Below the icon, the text reads: 'Drop files here or use the upload button'.

After the key upload is complete, it should appear in the Cloud Storage browser

Browser

UPLOAD FILES

UPLOAD FOLDER

CREATE FOLDER

REFRESH


SHARE PUBLICLY

DELETE

Filter by prefix...

Buckets

 / keystore-lot2analytics

<input type="checkbox"/>	Name	Size	Type	Storage class	Last modified	Share publicly	
<input type="checkbox"/>	 lot2Analytics-a68b9ef614d3.json	2.28 KB	application/octet-stream	Multi-Regional	1/29/18, 1:42 PM	<input type="checkbox"/>	⋮

Upload 1 of 1 complete

lot2Analytics-a68b9ef614d3.json

Finished

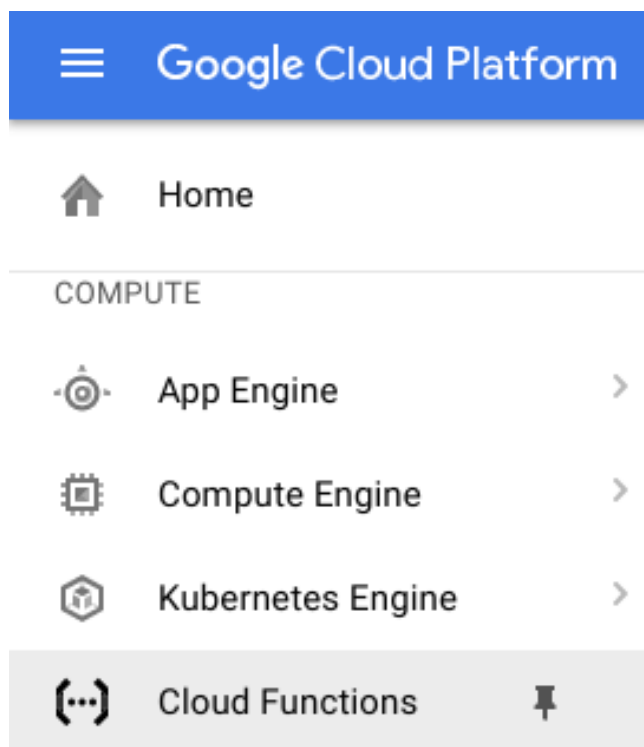
×

Make note of the storage bucket name and the security key file name for later.

Create a Cloud function

Cloud computing has made possible fully serverless models of computing where logic can be spun up on-demand in response to events originating from anywhere. For this lab, a Cloud Function will start each time a message is published to the weather topic, will read the message and then store it in BigQuery.

From the Cloud Console, select Cloud Functions



If you see an API message, click on the Enable API button

Google Cloud Functions BETA

Google Cloud Functions is a lightweight, event-based, asynchronous compute solution that allows you to create small, single-purpose functions that respond to cloud events without the need to manage a server or a runtime environment

 Cloud Functions API not enabled

[Enable API](#)

Click on the Create function button

Google Cloud Functions

Google Cloud Functions is a lightweight, event-based, asynchronous compute solution that allows you to create small, single-purpose functions that respond to cloud events without the need to manage a server or a runtime environment

Create function

In the Name field, type `function-weatherPubSubToBQ`. For Trigger select Cloud Pub/Sub topic and in the Topic dropdown select `weatherdata`. For source code, select inline editor. In the `index.js` tab, paste the following code over what is there to start with. **Make sure to change the constants for `projectId`, `datasetId` and `tableId` to fit your environment.**

Code

```
/**
 * Background Cloud Function to be triggered by PubSub.
 *
 * @param {object} event The Cloud Functions event.
 * @param {function} callback The callback function.
 */
exports.subscribe = function (event, callback) {
  const BigQuery = require('@google-cloud/bigquery');
  const projectId = "myProject"; //Enter your project ID here
  const datasetId = "myDataset"; //Enter your BigQuery dataset name here
  const tableId = "myTable"; //Enter your BigQuery table name here -- make sure it is setup
  correctly
  const PubSubMessage = event.data;
  // Incoming data is in JSON format
  const incomingData = PubSubMessage.data ? Buffer.from(PubSubMessage.data,
    'base64').toString() : '{"sensorID':'na','timecollected':'1/1/1970
    00:00:00','zipcode':'00000','latitude':'0.0','longitude':'0.0','temperature':'-273','humidity':'-1','de
    wpoint':'-273','pressure':'0'}';
  const jsonData = JSON.parse(incomingData);
  var rows = [jsonData];

  console.log(`Uploading data: ${JSON.stringify(rows)}`);
```

```

// Instantiates a client
const bigquery = BigQuery({
  projectId: projectId
});

// Inserts data into a table
bigquery
  .dataset(datasetId)
  .table(tableId)
  .insert(rows)
  .then((foundErrors) => {
    rows.forEach((row) => console.log('Inserted: ', row));

    if (foundErrors && foundErrors.insertErrors != undefined) {
      foundErrors.forEach((err) => {
        console.log('Error: ', err);
      })
    }
  })
  .catch((err) => {
    console.error('ERROR:', err);
  });
// [END bigquery_insert_stream]

callback();
};

```

In the package.json tab, paste the following code over the placeholder code that is there

```

{
  "name": "function-weatherPubSubToBQ",
  "version": "0.0.1",
  "private": true,
  "license": "Apache-2.0",
  "author": "Google Inc.",
  "dependencies": {
    "@google-cloud/bigquery": "^0.9.6"
  }
}

```

If the Function to execute is set to "HelloWorld", change it to "subscribe". Click the Create button

Cloud Functions

Create function

Name

function-weatherPubSubToBQ

Memory allocated

256 MB

Trigger

☐ HTTP trigger

☒ Cloud Pub/Sub topic

☐ Cloud Storage bucket

Topic

weatherdata

Source code

☒ Inline editor

☐ ZIP upload

☐ ZIP from Cloud Storage

☐ Cloud Source repository

index.js

package.json

1 {

2 "name": "function-weatherPubSubToBQ",

3 "version": "0.0.1",

4 "private": true,

5 "license": "Apache-2.0",

6 "author": "Google Inc.",

7 "dependencies": {

8 "@google-cloud/bigquery": "^0.9.6"

9 }

10 }

Function to execute

subscribe

More

Function deployment might take a few minutes. To build and test the function locally, use the local emulator

Create

Cancel

It will take about 2 minutes until your function will show that it has deployed

Cloud Functions	Overview	CREATE FUNCTION	REFRESH	DELETE	COPY
<div>Filter functions</div> <div>Columns</div>					
<input type="checkbox"/> Name	Region	Trigger	Memory allocated	Executed function	Last deployed
<input type="checkbox"/> function-weatherPubSubToBQ	us-central1	topic: weatherdata	256 MB	subscribe	1/29/18, 3:34 PM

Congratulations! now we just connected Pub/Sub to BigQuery via Functions.