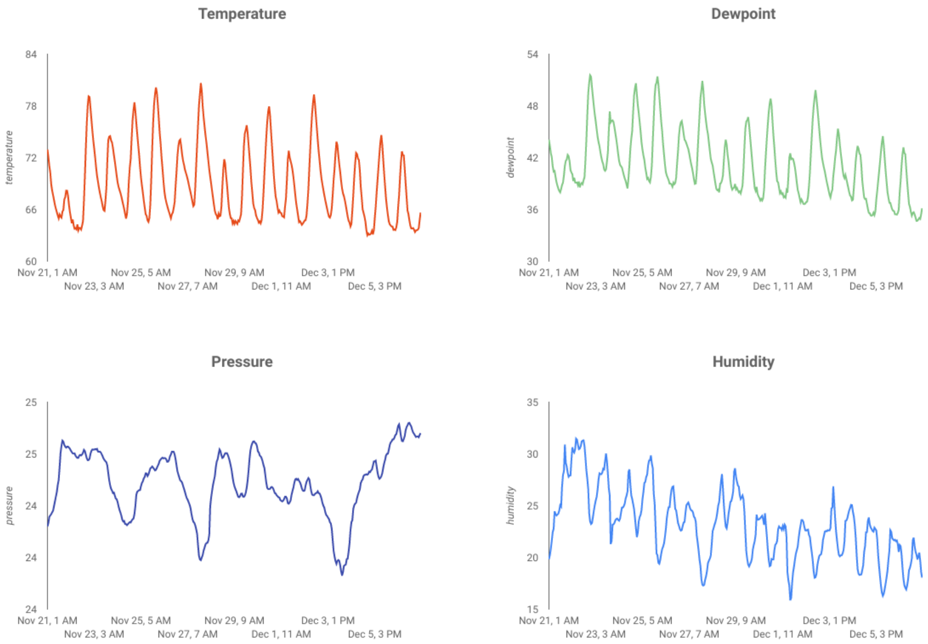
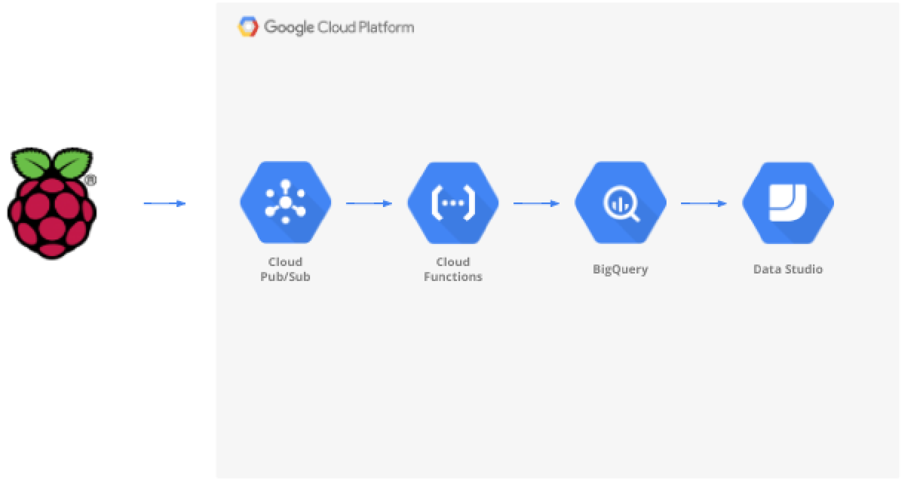
# Serverless IOT Data Processing

**PROBLEM DEFINITION:**

While multi-tier applications consisting of web, application server and database are foundational to web development and are the starting point for many websites, success will often bring challenges around scalability, integration and agility. For example, how can data be handled in real-time and how can it be distributed to multiple key business systems? These issues coupled with the demands of internet-scale applications drove the need for a distributed messaging systemand gave rise to an architectural pattern of using data pipelines to achieve resilient, real-time systems. As a result, understanding how to publish real-time data to build a data pipeline are crucial skills for developer and architect alike.

In this code lab, we are going to build a weather data pipeline that starts with an Internet of Things (IoT) device, utilizes a message queue to receive and deliver data. Leverages a serverless function to move the data to a data warehouse and then creates a dashboard that displays the information. A Raspberry Pi with a weather sensor will be used for the IoT device and several components of the Google Cloud Platform will be from the data pipeline. Building out the Raspberry Pi, while beneficial, is an optional portion of this code lab.





## DESIGN THINKING

→Create Pub/sub

Deploy a Google Cloud Function.

→Leverage Google Big Query.

→Dashboard using Google Data Studio.

→If we build out the IoT sensor, we will also learn how to utilize the Google Cloud SDK and how to secure remote access calls to the Google Cloud Platform.

If we want to build the IoT sensor portion of this code lab instead of leveraging sample data and a script, we will also need the following

→Raspberry Pi Zero W with power supply, SD memory card and case.

→USB card reader.

→USB hub (to allow for connecting a keyboard and mouse into the sole USB port on the Raspberry Pi)

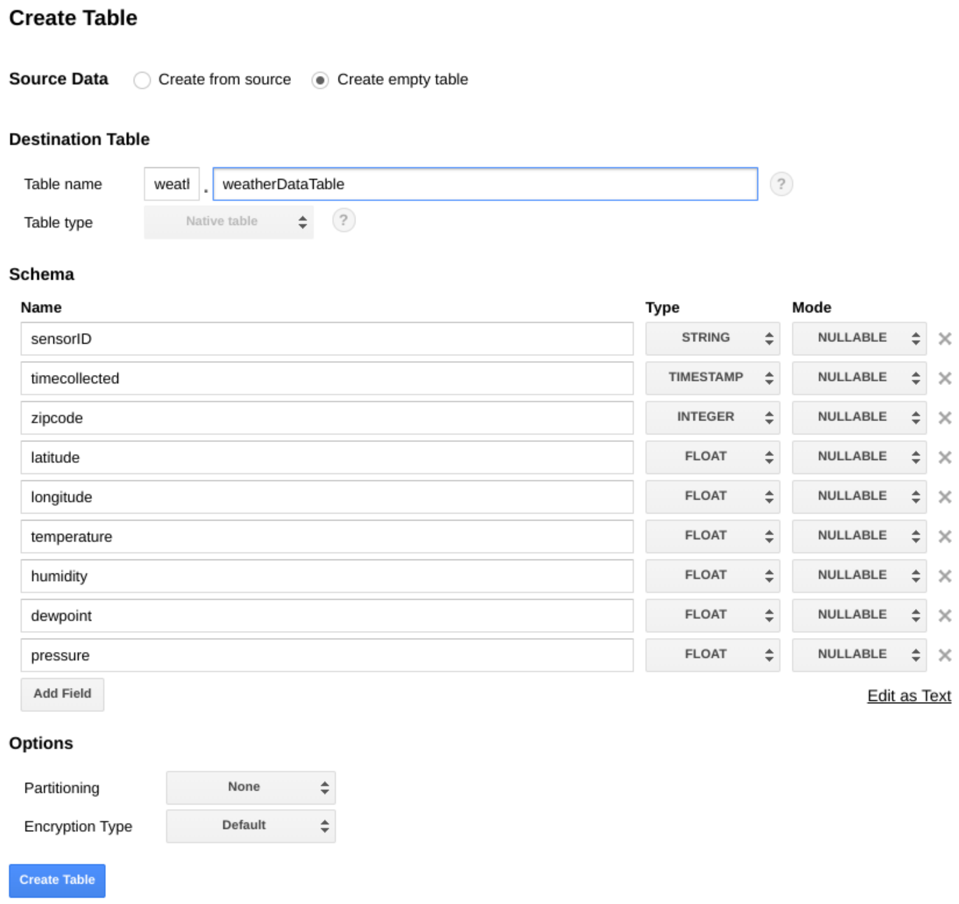
→Female-to-female breadboard wires.

→GPIO Hammer Headers.

→BME280 sensor.

**CREATE A BIG QUERY TABLE:**

Big Query 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 the IoT weather data. Select Big Query from the Cloud console. **This will open Big Query in a new window** (don't close the original window as you'll need to access it again).



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 |

Options

Partitioning: None

Encryption Type: Default

**CREATE A PUB/SUB TOPIC:**

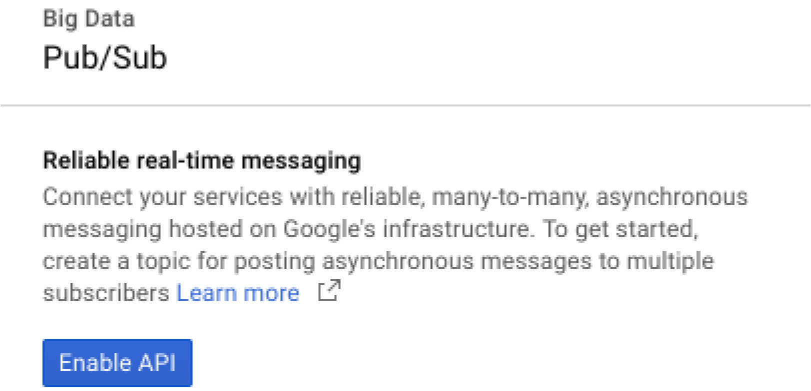
Cloud pub/sub is a simple, reliable, scalable foundation for stream analytics and event-driven computing systems. As a result, it is perfect for handling incoming IOT messages and then allowing  downstream systems to process them.

If you are still in the window for Big Query, 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 you see an Enable API prompt, click the Enable API button.

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



Big Data

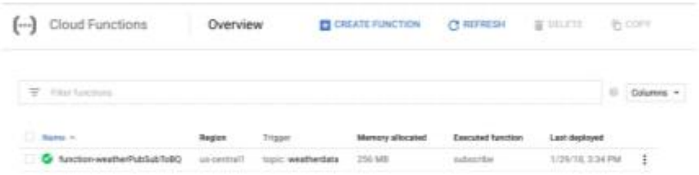
Pub/Sub

**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 CLOUD FUNCTION:**

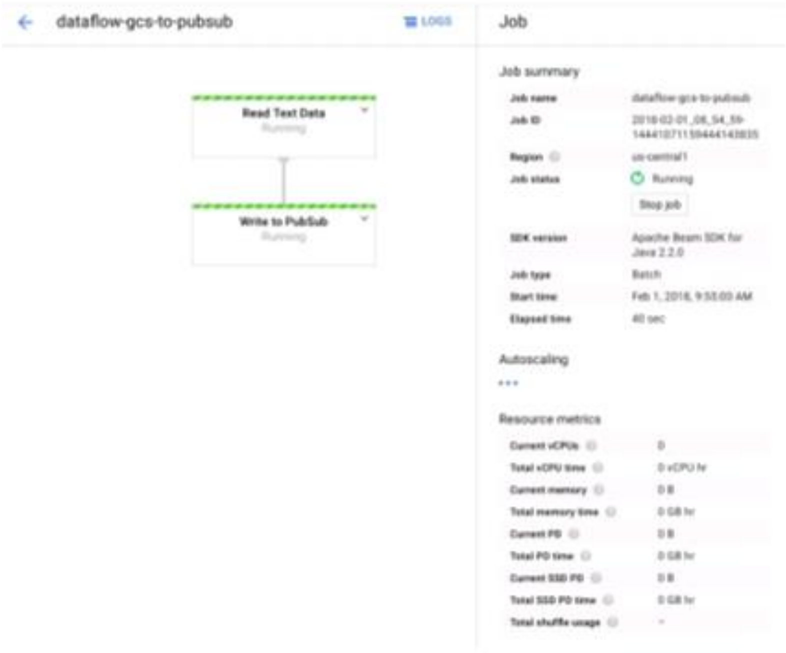
Cloud computing has made possible fully serverless models of computing where logic can be spun up on-demand in response to events originating from anywhere. For this lab, a Cloud Function will start each time a message is published to the weather topic, will read the message and then store it in Big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



| Name | Region | Trigger | Memory allocated | Executed function | Last deployed |
|---|---|---|---|---|---|
| function-weatherPubSubToBQ | us-central1 | topic: weatherdata | 256 MB | subscribe | 1/29/18, 3:34 PM |

**START THE DATA PIPELINE:**

Might need to enable compute API.



Job

Job summary

| | |
|---|---|
| Job name | dataflow-gcs-to-pubsub |
| Job ID | 2018-02-01_06_54_59-14441071159444143835 |
| Region | us-central1 |
| Job status | Running |
| SDK version | Apache Beam SDK for Java 2.2.0 |
| Job type | Batch |
| Start time | Feb 1, 2018, 9:55:03 AM |
| Elapsed time | 40 sec |

Autoscaling

Resource metrics

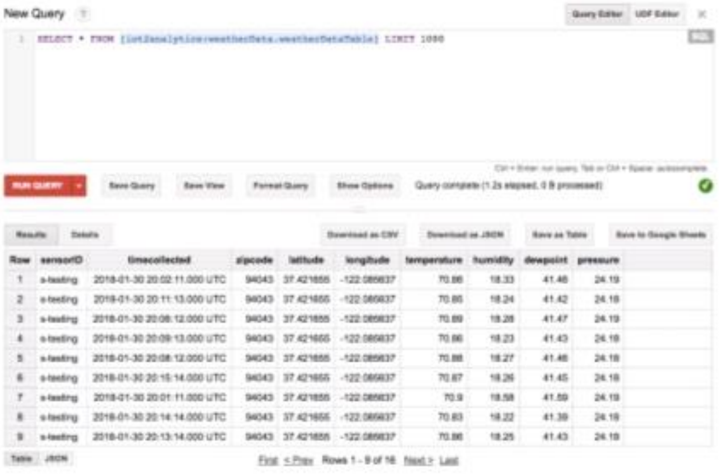
| | |
|---|---|
| Current vCPUs | 0 |
| Total vCPU time | 0 vCPU hr |
| Current memory | 0 B |
| Total memory time | 0 GB hr |
| Current PD | 0 B |
| Total PD time | 0 GB hr |
| Current SSD PD | 0 B |
| Total SSD PD time | 0 GB hr |
| Total shuffle usage | - |

**CHECK THAT DATA IS FLOWING:**

**CLOUD FUNCTION:**

**Ensure that the Cloud Function is being triggered by Pub/Sub**

```
gcloud beta functions logs read function-weatherPubSubToBQ
```



```
SELECT * FROM [iot2analytics:weatherData.weatherDataTable] LIMIT 1000
```

Ctrl + Enter: run query, Tab or Ctrl + Space: autocomplete

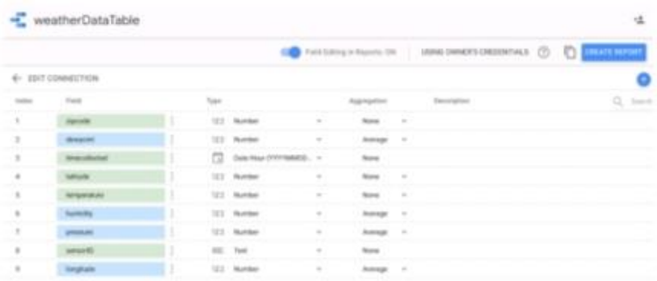
Query complete (1.2s elapsed, 0 B processed)

| Row | sensorID | timecollected | zipcode | latitude | longitude | temperature | humidity | dewpoint | pressure |
|---|---|---|---|---|---|---|---|---|---|
| 1 | s-testing | 2018-01-30 20:02:11.000 UTC | 94043 | 37.421655 | -122.085637 | 70.86 | 18.33 | 41.46 | 24.19 |
| 2 | s-testing | 2018-01-30 20:11:13.000 UTC | 94043 | 37.421655 | -122.085637 | 70.85 | 18.24 | 41.42 | 24.18 |
| 3 | s-testing | 2018-01-30 20:06:12.000 UTC | 94043 | 37.421655 | -122.085637 | 70.89 | 18.28 | 41.47 | 24.19 |
| 4 | s-testing | 2018-01-30 20:09:13.000 UTC | 94043 | 37.421655 | -122.085637 | 70.86 | 18.23 | 41.43 | 24.18 |
| 5 | s-testing | 2018-01-30 20:08:12.000 UTC | 94043 | 37.421655 | -122.085637 | 70.88 | 18.27 | 41.46 | 24.18 |
| 6 | s-testing | 2018-01-30 20:15:14.000 UTC | 94043 | 37.421655 | -122.085637 | 70.87 | 18.26 | 41.45 | 24.18 |
| 7 | s-testing | 2018-01-30 20:01:11.000 UTC | 94043 | 37.421655 | -122.085637 | 70.9 | 18.58 | 41.59 | 24.19 |
| 8 | s-testing | 2018-01-30 20:14:14.000 UTC | 94043 | 37.421655 | -122.085637 | 70.83 | 18.22 | 41.39 | 24.18 |
| 9 | s-testing | 2018-01-30 20:13:14.000 UTC | 94043 | 37.421655 | -122.085637 | 70.86 | 18.25 | 41.43 | 24.18 |

Rows 1 - 9 of 16

**CREATE A DATA STUDIO DASHBOARD:**

Google data studio turns our data into informative dashboards and reports that are easy to read, easy to share, and fully customizable.



| Index | Field | Type | Aggregation | Description |
|---|---|---|---|---|
| 1 | zipcode | Number | None | |
| 2 | dewpoint | Number | Average | |
| 3 | timecollected | Date Hour (YYYYMMDD... | None | |
| 4 | latitude | Number | None | |
| 5 | temperature | Number | None | |
| 6 | humidity | Number | Average | |
| 7 | pressure | Number | Average | |
| 8 | sensorID | Text | None | |
| 9 | longitude | Number | Average | |

Onboarding new data or building new analytics pipelines in traditional analytics architectures typically requires extensive coordination across business, data engineering, and data science and analytics teams to first negotiate requirements, schema, infrastructure capacity needs, and workload management.

For many use cases today however, business users, data scientists, and analysts are demanding easy, frictionless, self-service options to build end-to-end data pipelines because it’s hard and inefficient to predefine constantly changing schemas and spend time negotiating capacity slots on shared infrastructure. The exploratory nature of machine learning (ML) and many analytics tasks means you need to rapidly ingest new

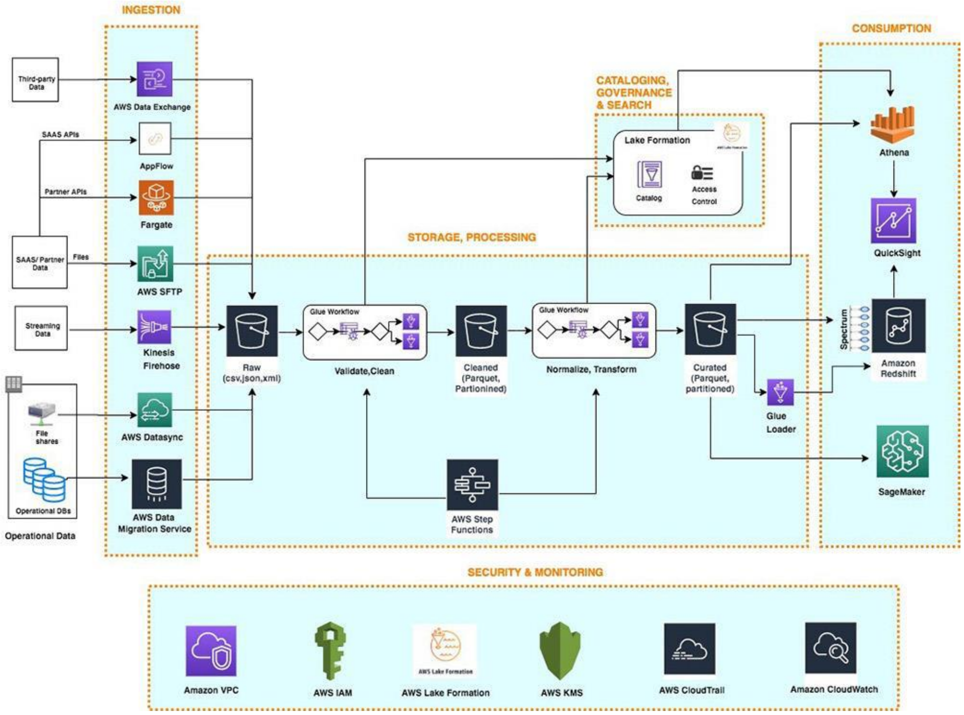
datasets and clean, normalize, and feature engineer them without worrying about operational overhead when you must think about the infrastructure that runs data pipelines.

**Serverless data analytics architecture:**

**To compose the layers described in our logical architecture, we introduced a reference architecture that uses serverless and managed services.**

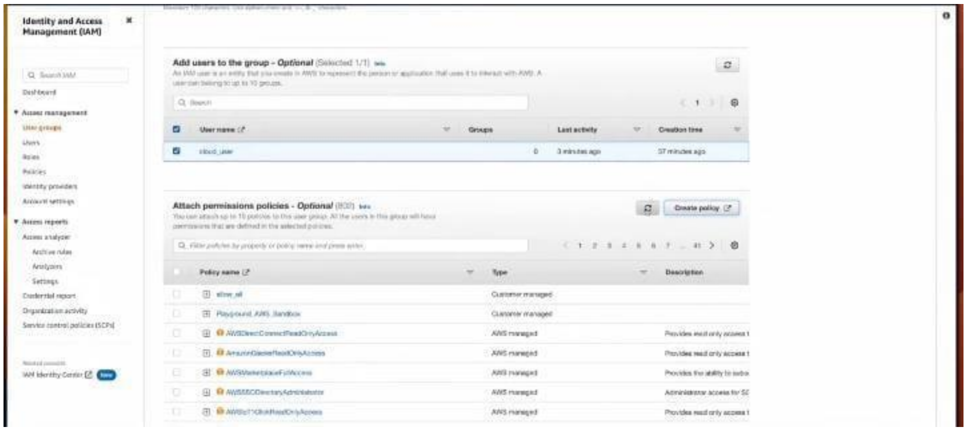
**It provides the following key benefit:**

- **Easy configuration-driven use.**
- **Freedom from infrastructure management.**
- **Pay-per-use pricing model.**



## INGESTION LAYER:

The ingestion layers in our serverless architecture are composed of a set of purpose-built services to enable data ingestion from a variety of sources.



Identity and Access Management (IAM)

Search IAM

Dashboard

▾ Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

▾ Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Organization activity

Service control policies (SCPs)

Related consoles

IAM Identity Center New

**Add users to the group - Optional** (Selected 1/1) Info

An IAM user is an entity that you create in AWS to represent the person or application that uses it to interact with AWS. A user can belong to up to 10 groups.

Search

| ☑ | User name | Groups | Last activity | Creation time |
|---|---|---|---|---|
| ☑ | cloud_user | 0 | 3 minutes ago | 37 minutes ago |

**Attach permissions policies - Optional** (830) Info

You can attach up to 10 policies to this user group. All the users in this group will have permissions that are defined in the selected policies.

Create policy

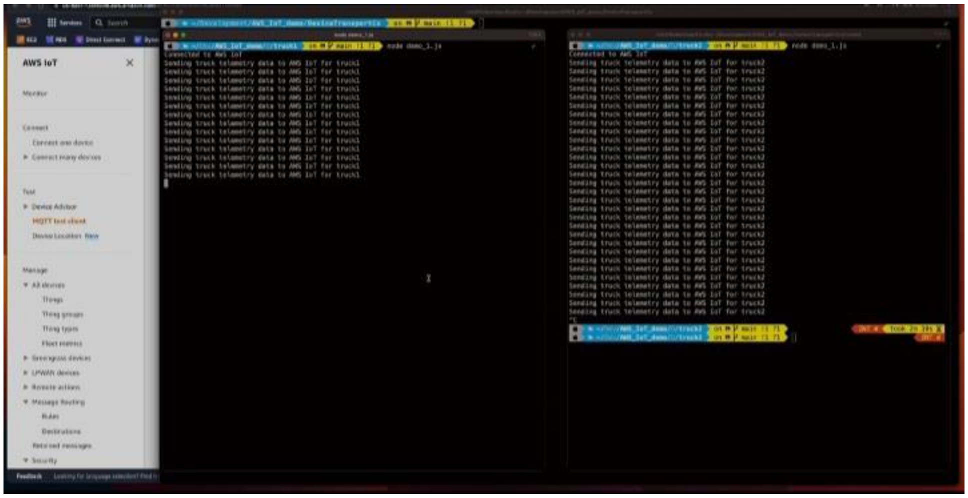
Filter policies by property or policy name and press enter

< 1 2 3 4 5 6 7 ... 41 >

| □ | Policy name | Type | Description |
|---|---|---|---|
| □ | allow_all | Customer managed | |
| □ | Playground_AWS_Sandbox | Customer managed | |
| □ | AWSDirectConnectReadOnlyAccess | AWS managed | Provides read only access t |
| □ | AmazonGlacierReadOnlyAccess | AWS managed | Provides read only access t |
| □ | AWSMarketplaceFullAccess | AWS managed | Provides the ability to subs |
| □ | AWSSSODirectoryAdministrator | AWS managed | Administrator access for SS |
| □ | AWSIoT1ClickReadOnlyAccess | AWS managed | Provides read only access t |

## STORAGE LAYER:

Data of any structure and any format can be stored as S3 objects without needing to predefine any schema. This enables services in the ingestion layer to quickly land a variety of source data into the data lake in its original source format.

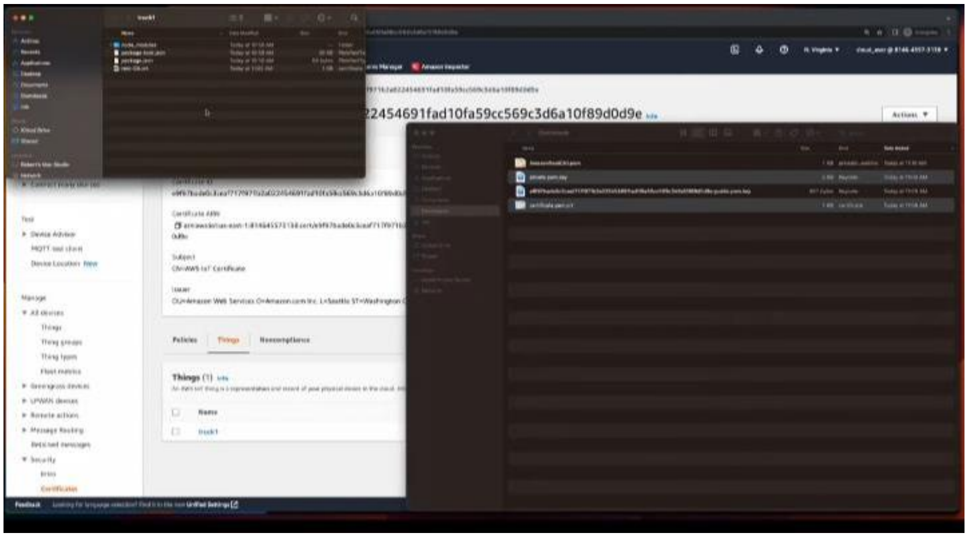


```
node demo_1.js
Connected to AWS IoT
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
Sending truck telemetry data to AWS IoT for truck1
```

```
node demo_1.js
Connected to AWS IoT
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
Sending truck telemetry data to AWS IoT for truck2
^C
```

**Cataloging and search layer:**

A data lake typically hosts a large number of datasets, and many of these datasets have evolving schema and new data partitions. A central Data Catalog that manages meta data for all the datasets in the data lake is crucial to enabling self-service discovery of data in the data lake. Additionally, separating metadata from data into a central schema enables schema-on-read for the processing and consumption layer components.



22454691fad10fa59cc569c3d6a10f89d0d9e Info

Certificate ARN

Subject
CN=AWS IoT Certificate

Issuer
OU=Amazon Web Services O=Amazon.com Inc. L=Seattle ST=Washington

Things (1) Info

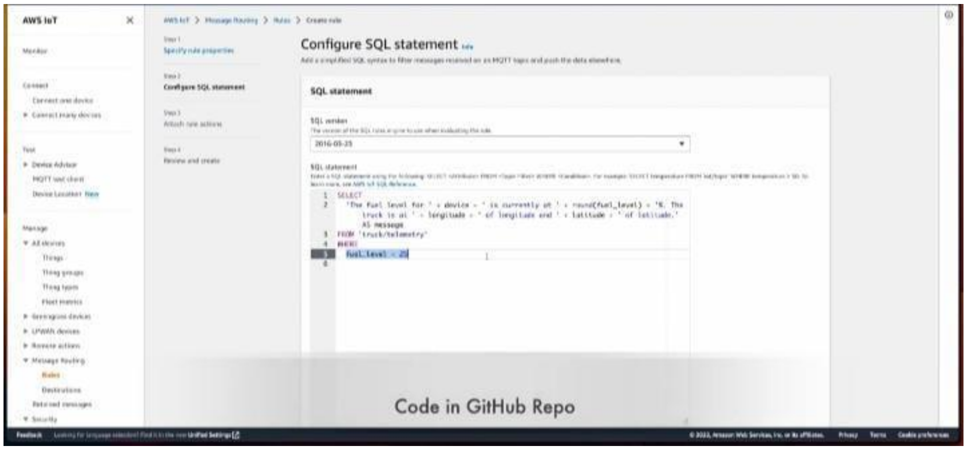
| Name |
|---|
| truck1 |

**Processing layer:**

The processing layer in our architecture is composed of two types of components:

•Components used to create multi-step data processing pipelines

•Components to orchestrate data processing pipelines on schedule or in response to event triggers (such as ingestion of new data into the landing zone)



Configure SQL statement Info

Add a simplified SQL syntax to filter messages received on an MQTT topic and push the data elsewhere.

SQL statement

SQL version

The version of the SQL rules engine to use when evaluating the rule.

2016-03-23

SQL statement

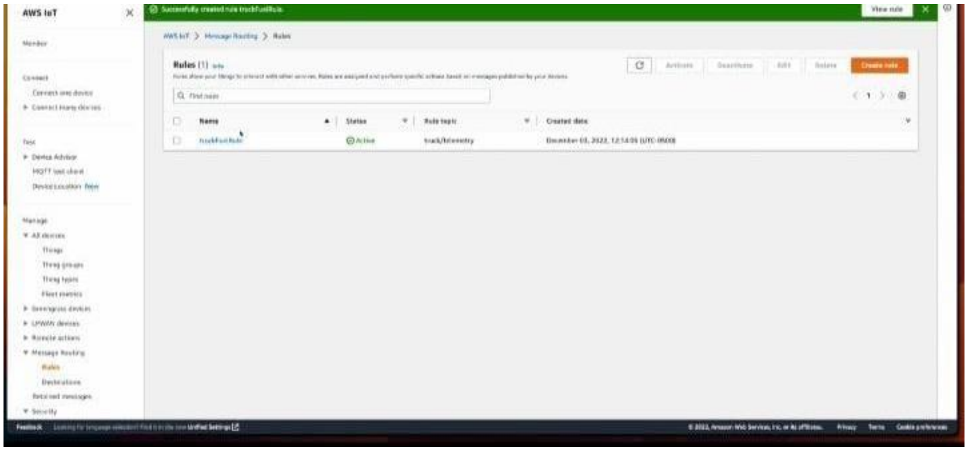
Enter a SQL statement using the following: SELECT <Attribute> FROM <Topic Filter> WHERE <Condition>. For example: SELECT temperature FROM 'iot/topic' WHERE temperature > 50. To learn more, see AWS IoT SQL Reference.

```
SELECT
  'The fuel level for ' + device + ' is currently at ' + round(fuel_level) + '%. The
      truck is at ' + longitude + ' of longitude and ' + latitude + ' of latitude.'
      AS message
FROM 'truck/telemetry'
WHERE
  fuel_level < 25
```

Code in GitHub Repo

**Consumption layer:**

The consumption layer in our architecture is composed using fully managed, purpose-built, analytics services that enable interactive SQL, BI dashboarding, batch processing, and ML.



Successfully created rule truckFuelRule.

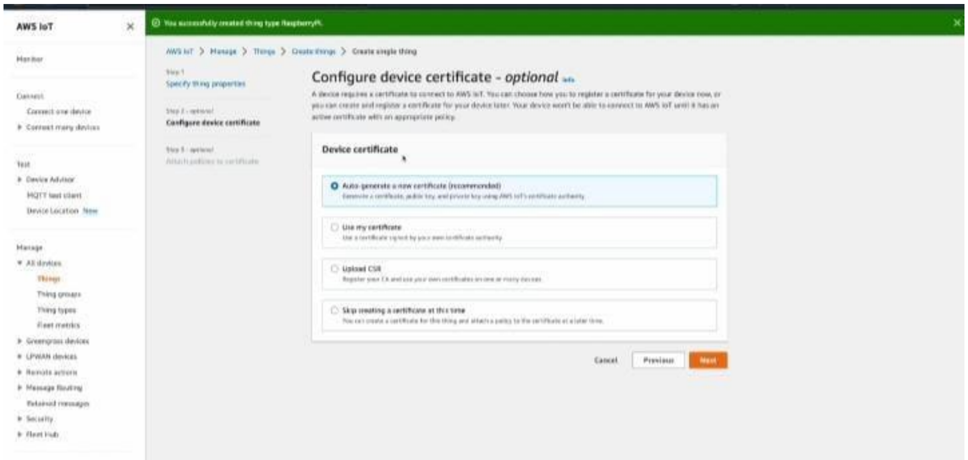
Rules (1) Info

Rules allow your things to interact with other services. Rules are analyzed and perform specific actions based on messages published by your devices.

| | Name | Status | Rule topic | Created date |
|---|---|---|---|---|
| □ | truckFuelRule | Active | truck/telemetry | December 03, 2022, 12:14:05 (UTC-08:00) |

**Security and governance layer:**

Components across all layers of our architecture protect data, identities, and processing resources by natively using the following capabilities provided by the security and governance layer.



You successfully created thing type RaspberryPi.

Configure device certificate - *optional* Info

A device requires a certificate to connect to AWS IoT. You can choose how you to register a certificate for your device now, or you can create and register a certificate for your device later. Your device won't be able to connect to AWS IoT until it has an active certificate with an appropriate policy.

Device certificate

● Auto-generate a new certificate (recommended)
Generate a certificate, public key, and private key using AWS IoT's certificate authority.

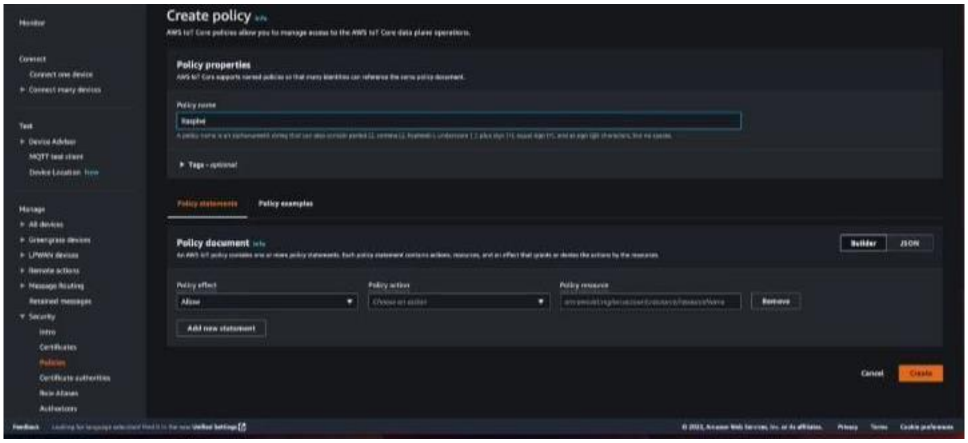
○ Use my certificate
Use a certificate signed by your own certificate authority.

○ Upload CSR
Register your CA and use your own certificates on one or many devices.

○ Skip creating a certificate at this time
You can create a certificate for this thing and attach a policy to the certificate at a later time.

**Additional considerations:**

In this post, we talked about ingesting data from diverse sources and storing it as S3 objects in the data lake and then using it to process ingested datasets until they’re in a consumable state.



Create policy

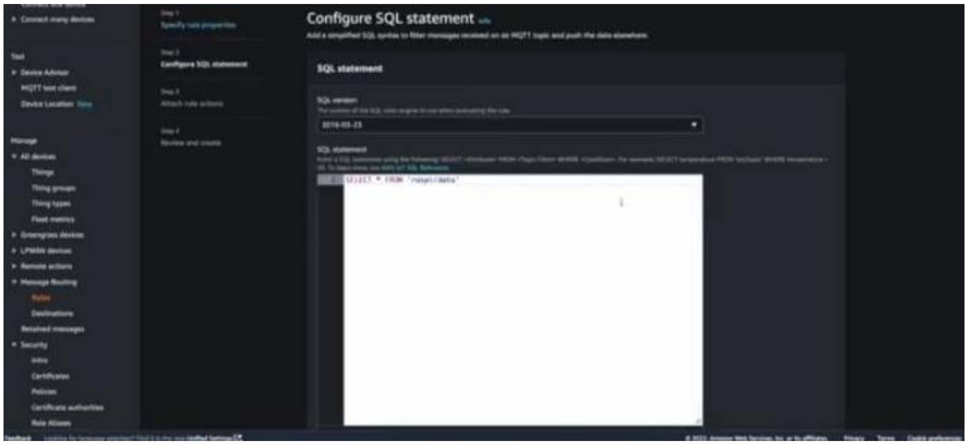
AWS IoT Core policies allow you to manage access to the AWS IoT Core data plane operations.

Policy properties

Policy name: Raspi

Policy document

Policy effect: Allow

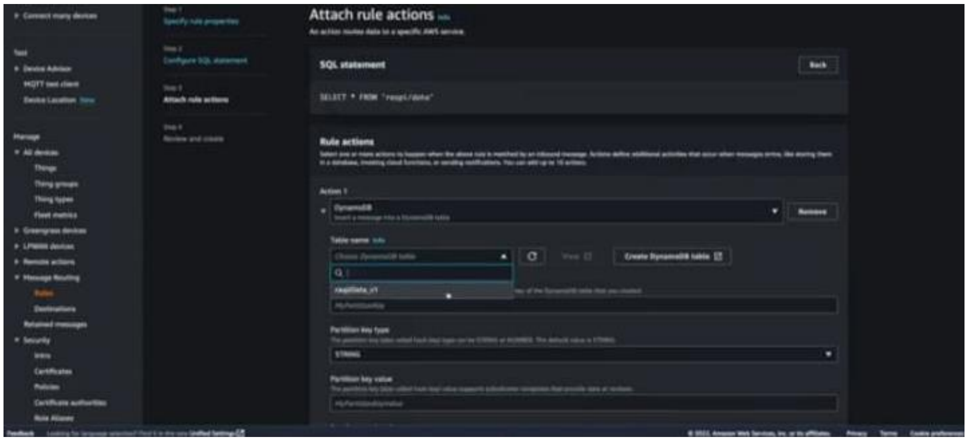


Configure SQL statement

SQL statement

SQL version: 2016-03-23

```
SELECT * FROM 'raspi/data'
```



Attach rule actions

An action routes data to a specific AWS service.

SQL statement

```
SELECT * FROM 'raspi/data'
```

Rule actions

Action 1: DynamoDB

Table name: raspidata_v1

Partition key type: STRING

**Conclusion:**

Serverless and managed services, you can build a modern, low-cost data lake centric analytics architecture in days. A decoupled, component-driven architecture allows you to start small and quickly add new purpose-built components to one of six architecture layers to address new requirements and data sources.