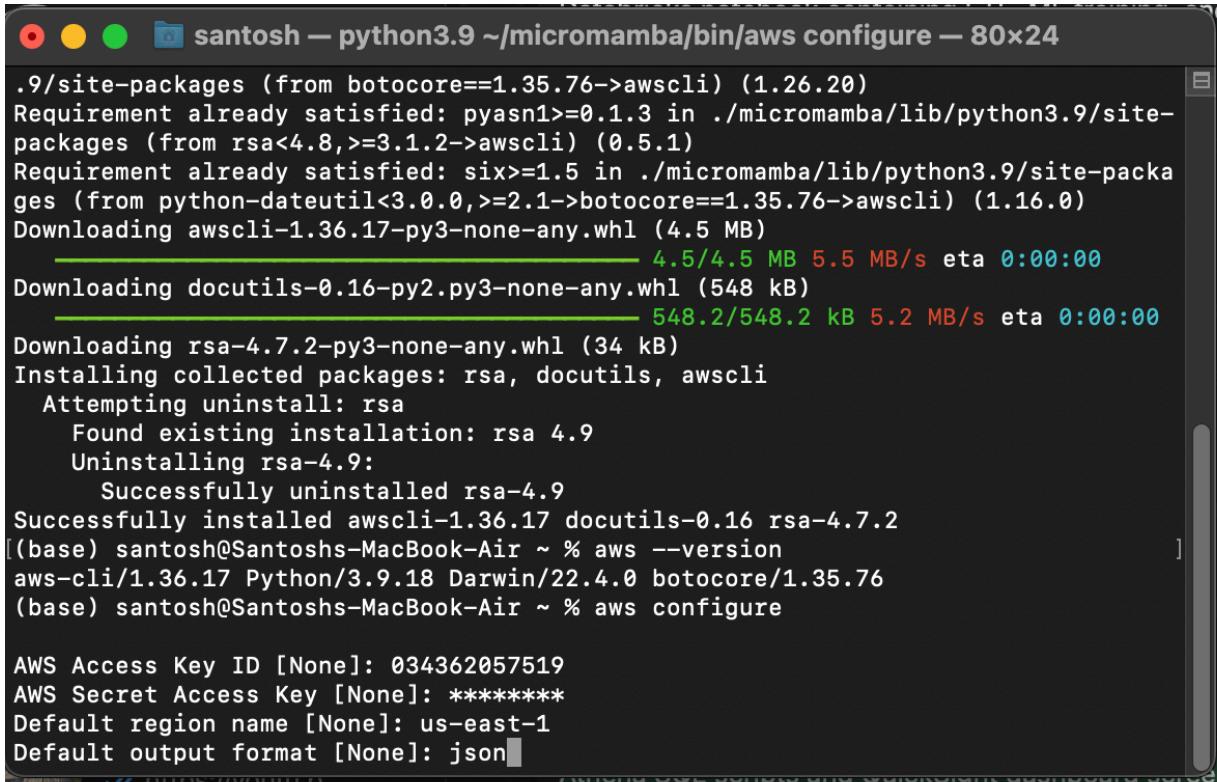


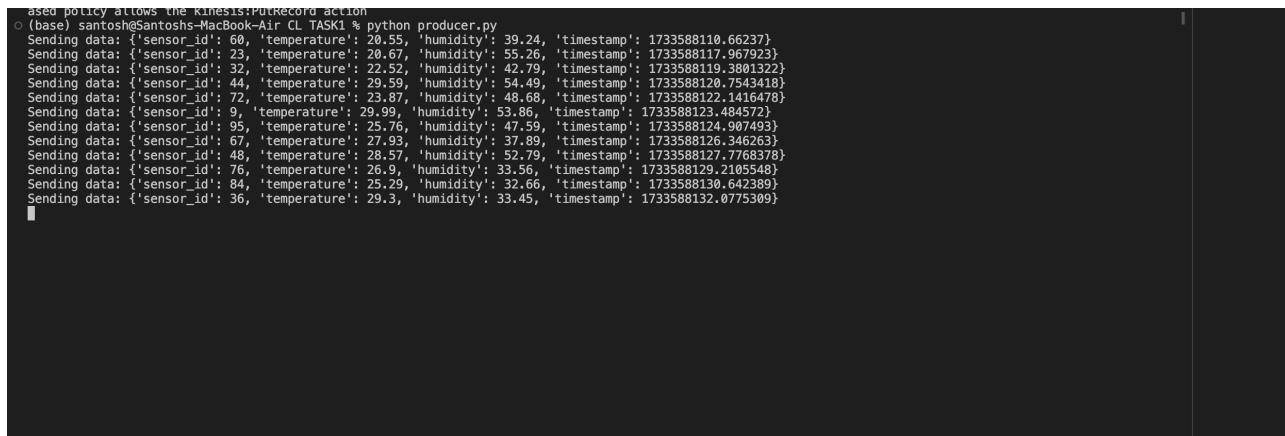
Step 1: Connect to AWS Account with AccountID and Secret Pass Key:



```
santosh — python3.9 ~/micromamba/bin/aws configure — 80x24
.9/site-packages (from botocore==1.35.76->awscli) (1.26.20)
Requirement already satisfied: pyasn1>=0.1.3 in ./micromamba/lib/python3.9/site-
packages (from rsa<4.8,>=3.1.2->awscli) (0.5.1)
Requirement already satisfied: six>=1.5 in ./micromamba/lib/python3.9/site-pac-
ges (from python-dateutil<3.0.0,>=2.1->botocore==1.35.76->awscli) (1.16.0)
Downloading awscli-1.36.17-py3-none-any.whl (4.5 MB)
4.5/4.5 MB 5.5 MB/s eta 0:00:00
Downloading docutils-0.16-py2.py3-none-any.whl (548 kB)
548.2/548.2 kB 5.2 MB/s eta 0:00:00
Downloading rsa-4.7.2-py3-none-any.whl (34 kB)
Installing collected packages: rsa, docutils, awscli
  Attempting uninstall: rsa
    Found existing installation: rsa 4.9
    Uninstalling rsa-4.9:
      Successfully uninstalled rsa-4.9
Successfully installed awscli-1.36.17 docutils-0.16 rsa-4.7.2
[(base) santosh@Santoshs-MacBook-Air ~ % aws --version
aws-cli/1.36.17 Python/3.9.18 Darwin/22.4.0 botocore/1.35.76
(base) santosh@Santoshs-MacBook-Air ~ % aws configure

AWS Access Key ID [None]: 034362057519
AWS Secret Access Key [None]: *****
Default region name [None]: us-east-1
Default output format [None]: json
```

Step 2: run the code “producer.py” and sent the real time data to AWS:



```
ased policy allows the kinesis:PutRecord action
(base) santosh@Santoshs-MacBook-Air CL TASK1 % python producer.py
Sending data: {'sensor_id': 60, 'temperature': 20.55, 'humidity': 39.24, 'timestamp': 1733588110.66237}
Sending data: {'sensor_id': 23, 'temperature': 20.67, 'humidity': 55.26, 'timestamp': 1733588117.967923}
Sending data: {'sensor_id': 32, 'temperature': 22.52, 'humidity': 42.79, 'timestamp': 1733588119.3881322}
Sending data: {'sensor_id': 44, 'temperature': 29.59, 'humidity': 54.49, 'timestamp': 1733588120.7543418}
Sending data: {'sensor_id': 72, 'temperature': 23.87, 'humidity': 48.68, 'timestamp': 1733588122.1416478}
Sending data: {'sensor_id': 9, 'temperature': 29.99, 'humidity': 53.86, 'timestamp': 1733588123.484572}
Sending data: {'sensor_id': 95, 'temperature': 25.76, 'humidity': 47.59, 'timestamp': 1733588124.907493}
Sending data: {'sensor_id': 67, 'temperature': 27.93, 'humidity': 37.89, 'timestamp': 1733588126.346263}
Sending data: {'sensor_id': 48, 'temperature': 28.57, 'humidity': 52.79, 'timestamp': 1733588127.7768378}
Sending data: {'sensor_id': 76, 'temperature': 26.9, 'humidity': 33.56, 'timestamp': 1733588129.2105548}
Sending data: {'sensor_id': 84, 'temperature': 25.29, 'humidity': 32.66, 'timestamp': 1733588130.642389}
Sending data: {'sensor_id': 36, 'temperature': 29.3, 'humidity': 33.45, 'timestamp': 1733588132.0775309}
```

Step 3:Login to AWS and step into kenosis and create a lamb function to get the input data stream and store it in a S3 bucket

Lambda > Functions > KinesisProcessor

Code source Info

You are using the new console editor.

Upload from Switch to the old editor

```

KinesisProcessor
lambda_function.py
lambda_function.py
1 import json
2 import boto3
3
4 s3_client = boto3.client('s3')
5 BUCKET_NAME = "my-transformed-data-bucket"
6
7 def lambda_handler(event, context):
8     transformed_data = []
9
10    for record in event['Records']:
11        # Decode and parse Kinesis record
12        payload = json.loads(record['kinesis']['data'])
13        print("Received payload: (payload)")
14
15        # Transform: Add geo-location (simulated example)
16        payload['geo_location'] = "Lat-(payload['sensor_id']),Lon-(payload['sensor_id'])"
17        transformed_data.append(payload)
18
19    # Save transformed data to S3
20    filename = f"transformed_data_{int(time.time())}.json"
21    s3_client.put_object(
22        Bucket=BUCKET_NAME,
23        Key=filename,
24        Body=json.dumps(transformed_data)
25    )
26
27    print(f"Saved transformed data to S3: (filename)")

```

Amazon Q Tip 1/3: Start typing to get suggestions ([ESC] to exit)

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy

AWS Search [Option+S]

Lambda > Functions > KinesisProcessor

Code source Info

You are using the new console editor.

Upload from Switch to the old editor

```

KinesisProcessor
lambda_function.py
lambda_function.py
1 import json
2 import boto3
3
4 s3_client = boto3.client('s3')
5 BUCKET_NAME = "my-transformed-data-bucket"
6
7 def lambda_handler(event, context):
8     transformed_data = []
9
10    for record in event['Records']:
11        # Decode and parse Kinesis record
12        payload = json.loads(record['kinesis']['data'])
13        print("Received payload: (payload)")
14
15        # Transform: Add geo-location (simulated example)
16        payload['geo_location'] = "Lat-(payload['sensor_id']),Lon-(payload['sensor_id'])"
17        transformed_data.append(payload)
18
19    # Save transformed data to S3
20    filename = f"transformed_data_{int(time.time())}.json"
21    s3_client.put_object(
22        Bucket=BUCKET_NAME,
23        Key=filename,
24        Body=json.dumps(transformed_data)
25    )
26
27    print(f"Saved transformed data to S3: (filename)")

```

Amazon Q Tip 1/3: Start typing to get suggestions ([ESC] to exit)

Create new test event Invoke Save

Create new test event

Event Name: KinesisTestEvent

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

Execution Results

Status: Succeeded

Test Event Name: KinesisTestEvent

Response:

```

{
  "statusCode": 200,
  "body": "\"Hello from Lambda!\""
}

```

Function Logs:

```

START RequestId: c467a47e-c133-488e-8abb-71f8b6c93c56 Version: $LATEST
END RequestId: c467a47e-c133-488e-8abb-71f8b6c93c56
REPORT RequestId: c467a47e-c133-488e-8abb-71f8b6c93c56 Duration: 1.92 ms Billed Duration: 2 ms Memory Size: 128 MB
Max Memory Used: 30 MB
Request ID: c467a47e-c133-488e-8abb-71f8b6c93c56

```

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy

Lambda > Functions > KinesisProcessor

KinesisProcessor

The trigger sensor-datastream was successfully added to function KinesisProcessor. The trigger is in a disabled state.

Throttle Copy ARN Actions

Function overview Info

Diagram Template

KinesisProcessor

Layers (0)

Add destination

Kinesis

Add trigger

Export to Infrastructure Composer Download

Description

Last modified: 35 minutes ago

Function ARN

arn:aws:lambda:us-east-1:054362057519:function:KinesisProcessor

Function URL Info

Code Test Monitor Configuration Aliases Versions

Code source Info

You are using the new console editor.

Upload from Switch to the old editor

```

KinesisProcessor
lambda_function.py
lambda_function.py
1 import json
2 import boto3
3
4 s3_client = boto3.client('s3')
5 BUCKET_NAME = "my-transformed-data-bucket"
6
7 def lambda_handler(event, context):
8     transformed_data = []
9
10    for record in event['Records']:
11        # Decode and parse Kinesis record
12        payload = json.loads(record['kinesis']['data'])
13        print("Received payload: (payload)")
14
15        # Transform: Add geo-location (simulated example)
16        payload['geo_location'] = "Lat-(payload['sensor_id']),Lon-(payload['sensor_id'])"
17        transformed_data.append(payload)
18
19    # Save transformed data to S3
20    filename = f"transformed_data_{int(time.time())}.json"
21    s3_client.put_object(
22        Bucket=BUCKET_NAME,
23        Key=filename,
24        Body=json.dumps(transformed_data)
25    )
26
27    print(f"Saved transformed data to S3: (filename)")

```

Amazon Q Tip 1/3: Start typing to get suggestions ([ESC] to exit)

CloudShell Feedback

© 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

Step 4: Create a Database and table inside it to store the transformed data in S3 buckets, set an IAM and edit the manage preferences

The screenshot shows the AWS Glue Crawler configuration page. The crawler is named 'samplecrawler' and is successfully starting. It uses the IAM role 'AWSGlueServiceRole-testrole'. The database is 'transformeddatadb' and the state is 'READY'. The crawler runs once, with the most recent run being successful. The run details show a duration of 01 min 03 s and a status of 'Running'.

Step 5: Now Step into to Amazon Athena and connect the Database and start Querrying to retrieve data i.e(Data Analysis)

The screenshot shows the Amazon Athena Query editor. A query is running, selecting sensor_id and event_count from the 'transformeddatadb' database. The results table shows one row with sensor_id 1 and event_count 24. The query was completed in 75 ms with 0.55 KB scanned.

#	sensor_id	event_count
1	1	24

Step 6: use QuickSight for analysis the datasets from Athena and connect to the database

