

AWS Lambda 운영

유정현

목차

1. Lambda, S3 연동 프로젝트
2. Lambda, DynamoDB 연동 프로젝트
3. Lambda를 사용한 Crawling

1. Lambda + S3 연동

전세계적으로 고유한 이름을 가진 bucket 3개 생성

Create bucket [Info](#)

Buckets are containers for data stored in S3. [Learn more](#)

General configuration

Bucket name

Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)

AWS Region

Asia Pacific (Seoul) ap-northeast-2 ▼

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Choose bucket

<input type="radio"/>	python-lambda-aespa	Asia Pacific (Seoul) ap-northeast-2	<u>Bucket and objects not public</u>
<input type="radio"/>	python-lambda-aespa2	Asia Pacific (Seoul) ap-northeast-2	<u>Bucket and objects not public</u>
<input type="radio"/>	python-lambda-aespa3	Asia Pacific (Seoul) ap-northeast-2	Bucket and objects not public

Python 3.8을 사용하는 lambda function 생성

Function name

Enter a name that describes the purpose of your function.

list-s3-buckets

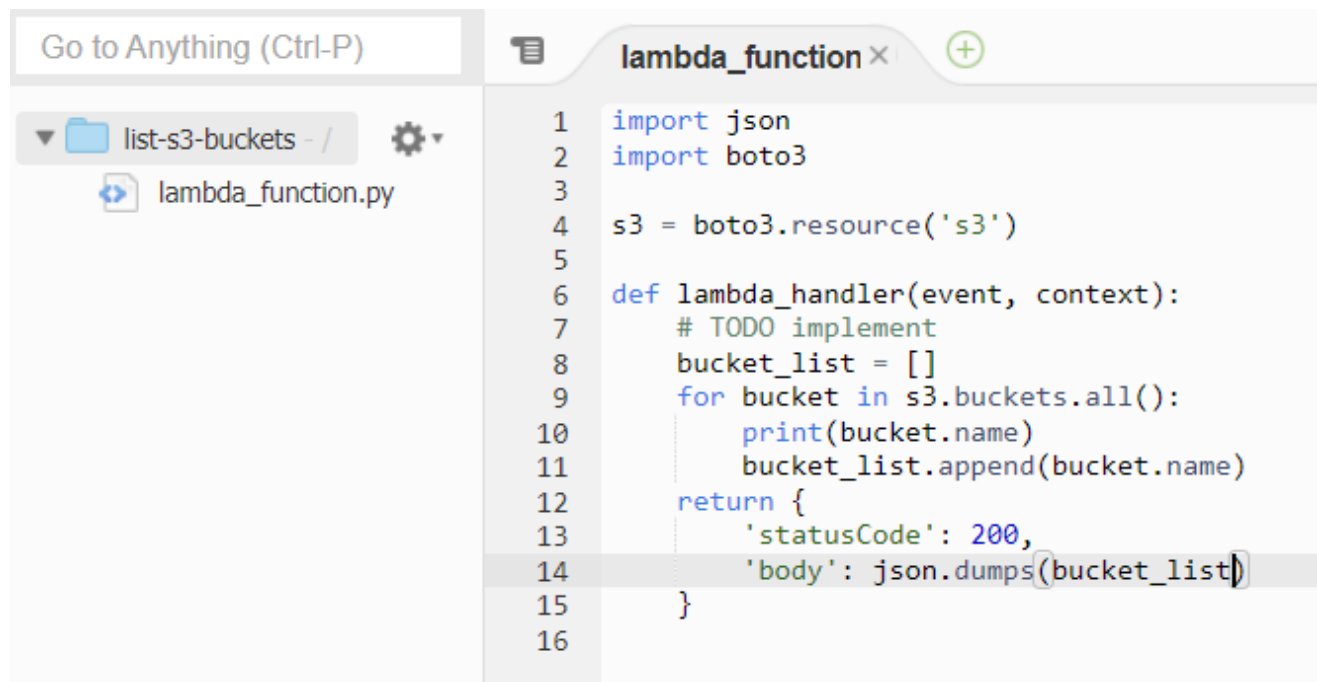
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.8

Bucket의 이름을 가져오는 python 코드로 수정



The screenshot shows a code editor with a file named `lambda_function.py` open. The editor has a sidebar on the left with a search bar "Go to Anything (Ctrl-P)" and a file explorer showing the project structure. The main editor area displays the following Python code:

```
1 import json
2 import boto3
3
4 s3 = boto3.resource('s3')
5
6 def lambda_handler(event, context):
7     # TODO implement
8     bucket_list = []
9     for bucket in s3.buckets.all():
10         print(bucket.name)
11         bucket_list.append(bucket.name)
12     return {
13         'statusCode': 200,
14         'body': json.dumps(bucket_list)
15     }
16
```

Function test 실행 시 AccessDenied 오류가 발생

▼ Execution results Status: **Failed** Max memory used: 69 M

Test Event Name S3ListTest	
Response { "errorMessage": "An error occurred (AccessDenied) when calling the ListBuckets operation: Access Denied", "errorType": "ClientError", "stackTrace": [" File \"/var/task/lambda_function.py\", line 9, in lambda_handler\n for bucket in s3.buckets.all():\n", " File \"/var/runtime/boto3/resources/collection.py\", line 83, in __iter__\n for page in self.pages():\n", " File \"/var/runtime/boto3/resources/collection.py\", line 161, in pages\n pages = [getattr(client, self._py_operation_name)(**params)]\n", " File \"/var/runtime/botocore/client.py\", line 386, in _api_call\n return self._make_api_call(operation_name, kwargs)\n", " File \"/var/runtime/botocore/client.py\", line 705, in _make_api_call\n raise error_class(parsed_response, operation_name)\n",] }	
Function Logs START RequestId: 000fd40f-ca5a-4e68-980c-c7ab6d31f664 Version: \$LATEST [ERROR] ClientError: An error occurred (AccessDenied) when calling the ListBuckets operation: Access Denied Traceback (most recent call last): ..File "/var/task/lambda_function.py", line 9, in lambda_handlerfor bucket in s3.buckets.all(): ..File "/var/runtime/boto3/resources/collection.py", line 83, in __iter__for page in self.pages(): ..File "/var/runtime/boto3/resources/collection.py", line 161, in pagespages = [getattr(client, self._py_operation_name)(**params)] ..File "/var/runtime/botocore/client.py", line 386, in _api_callreturn self._make_api_call(operation_name, kwargs) ..File "/var/runtime/botocore/client.py", line 705, in _make_api_call	

Function에 연결된 role에 S3ReadOnlyAccess policy 추가
bucket의 이름만 읽어올 것이기 때문에 최소 권한만 부여

Code

Test

Monitor

Configuration

Aliases

Versions

General configuration

Triggers

Permissions

Execution role

Role name

[list-s3-buckets-role-4usq6hht](#)

Attach policies

Add inline policy

Policy name	Policy type	
 AmazonS3ReadOnlyAccess	AWS managed policy	✕
AWSLambdaBasicExecutionRole-d23cecf1-44b9-49b1-bd2a-c1019165a5c8	Managed policy	✕


API Gateway를 생성해 function에 엔드포인트 생성

Triggers (1)

Enable

Disable

Fix

<input type="checkbox"/>	Trigger
<input type="checkbox"/>	<div>API Gateway: list-s3-buckets-API arn:aws:execute-api:ap-northeast-2:892169241323:if675ipk47/*/list-s3-buckets API endpoint: https://if675ipk47.execute-api.ap-northeast-2.amazonaws.com/default/list-s3-buckets ▶ Details</div>

접속 시 배열로 bucket 이름을 가져오는 것 확인

```
["python-lambda-aespa", "python-lambda-aespa2", "python-lambda-aespa3"]
```

2. Lambda + DynamoDB 연동

Name을 partition key로 가지는 planets table 생성

planets

Actions View items

< Overview Indexes Monitor Global tables Backups Exports and streams Ac >

General information

Partition key name (String)	Sort key -	Capacity mode Provisioned	Table status ✔ Active ✔ No active alarms
--------------------------------	---------------	------------------------------	------------------------------------------------

현재 table에 있는 item

Items returned (4)

Actions Create item

< 1 > ⚙️ 🔗

<input type="checkbox"/>	name ▼	온도 ▼
<input type="checkbox"/>	화성	추움
<input type="checkbox"/>	지구	따뜻
<input type="checkbox"/>	금성	뜨거움
<input type="checkbox"/>	수성	엄청 뜨거움

(1) read items

Python 3.8을 사용하는 lambda function 생성

Basic information

Function name

Enter a name that describes the purpose of your function.

DDGetItem

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.8

table의 전체 item을 읽어오는 코드 작성

lambda_function ×

Execution results ×



```
import json
import boto3

dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('planets')

def lambda_handler(event, context):
    response = table.scan()
    output = ''
    for r in response['Items']:
        s = f"{r['name']} : {r['온도']}\n"
        print(s)
        output += s
    return {
        'statusCode': 200,
        'body': output
    }
```

(1) read items

Python 3.8을 사용하는 lambda function 생성

Basic information

Function name

Enter a name that describes the purpose of your function.

DDGetItem

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.8

table의 전체 item을 읽어오는 코드 작성

lambda_function ×

Execution results ×

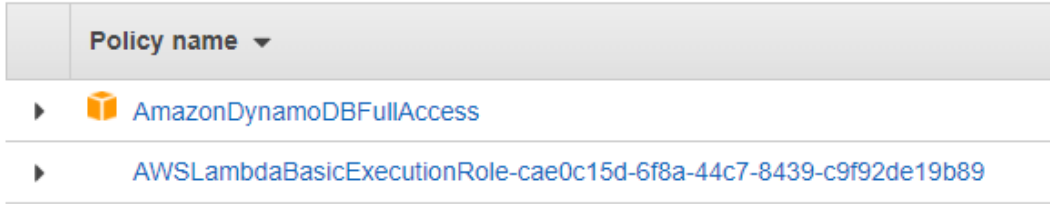


```
import json
import boto3

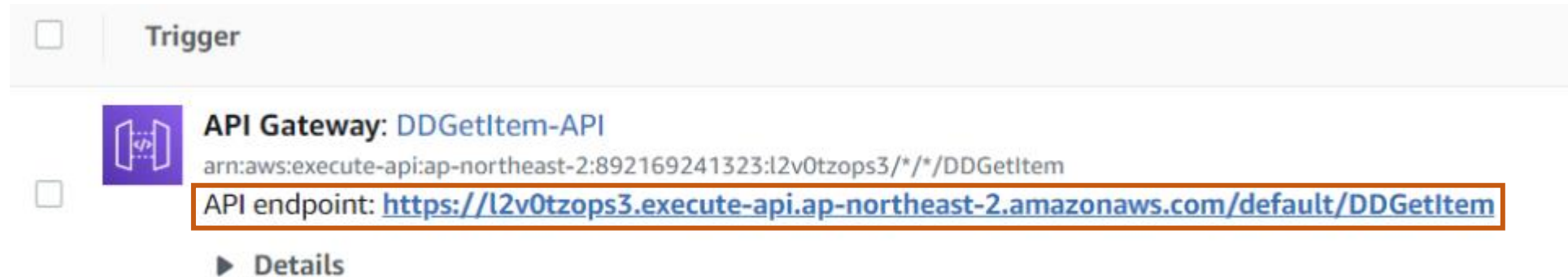
dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('planets')

def lambda_handler(event, context):
    response = table.scan()
    output = ''
    for r in response['Items']:
        s = f"{r['name']} : {r['온도']}\n"
        print(s)
        output += s
    return {
        'statusCode': 200,
        'body': output
    }
```

function에 연결 된 role에 DynamoDBFullAccess 부여



API Gateway를 생성해 function에 엔드포인트 생성



접속 시 table의 item을 전부 가져오는 것 확인

지구 : 따뜻
금성 : 뜨거움
수성 : 엄청 뜨거움
화성 : 추움

(2) write item

Python 3.8을 사용하는 lambda function 생성

Basic information

Function name

Enter a name that describes the purpose of your function.

DDPutItem

Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime [Info](#)

Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.

Python 3.8


table에 item을 삽입하는 코드 작성

```
import boto3

dynamodb = boto3.resource('dynamodb')
table = dynamodb.Table('planets')

def lambda_handler(event, context):
    table.put_item(
        Item = {
            'name': '해왕성',
            '온도': '매우매우 추움'
        }
    )
    return {
        'statusCode': 200,
        'body': 'Item added'
    }
```

Read, write 둘 다 필요하기 때문에 function에 연결된 role에 DynamoDBFullAccess 부여

Policy name ▾
▶  AmazonDynamoDBFullAccess
▶ AWSLambdaBasicExecutionRole-cae0c15d-6f8a-44c7-8439-c9f92de19b89

코드 실행 시 성공 메시지

✔ Execution result: succeeded ([logs](#))

▼ Details

The area below shows the result returned by your function execution. [Learn more](#) about returning results from your function.

```
{
  "statusCode": 200,
  "body": "Item added"
}
```

(1)에서 만든 엔드포인트로 접속 시 item 추가된 것 확인 가능

지구 : 따뜻
금성 : 뜨거움
수성 : 엄청 뜨거움
화성 : 추움
해왕성 : 매우매우 추움