

DynamoDB 실습

AWS DynamoDB 실습

목차

- DynamoDB생성
- Lambda에 연결해서 테스트
- 여러 데이터 동시에 입력하기 테스트

DynamoDB 생성

AWS DynamoDB 생성

- 파티션 키: 테이블 내에서 사용될 고유 키
- 정렬 키: 선택사항. 기본키의 두 번째 부분. 구매일자를 사용해서 고객 데이터를 정렬할 수 있다.

고객 아이디	Item 카테고리	가격	구매일자
String	String	Integer	String

테이블 세부 정보 [정보](#)

DynamoDB는 테이블을 생성할 때 테이블 이름과 기본 키만 필요한 스키마리스 데이터베이스입니다.

테이블 이름

테이블을 식별하는 데 사용됩니다.

문자, 숫자, 밑줄(_), 하이픈(-) 및 마침표(.)만 포함하는 3~255자의 문자입니다.

파티션 키

파티션 키는 테이블 기본 키의 일부로, 테이블에서 항목을 검색하고 확장성과 가용성을 위해 호스트에 데이터를 할당하는 데 사용되는 해시 값입니다.

문자열 ▼

1~255자이고 대소문자를 구분합니다.

정렬 키 - 선택 사항

정렬 키를 테이블 기본 키의 두 번째 부분으로 사용할 수 있습니다. 정렬 키를 사용하면 동일한 파티션 키를 공유하는 모든 항목을 정렬하거나 검색할 수 있습니다.

문자열 ▼

1~255자이고 대소문자를 구분합니다.

AWS DynamoDB 생성

- 테이블 생성 완료

테이블 (1) 정보

🔄

작업 ▼

삭제

테이블 생성

🔍 테이블 이름으로 테이블 찾기

모든 테이블 태그 ▼

< 1 > ⚙️

<input type="checkbox"/>	이름 ▲	상태	파티션 키	정렬 키	인덱스	삭제 방지	읽기 용량 모드
<input type="checkbox"/>	aws-learner-dynamodb-table	⋮ 생성 중	customer_id (S)	transaction_date (S)	0	🔒 끄기	Auto Scaling...

AWS DynamoDB 생성 정보

- 추가정보
- ARN: AWS에 통용되는 고유 식별자. 다른 리소스나 코드에서 참조할 수 있다.

DynamoDB 스트림

⊖ 끄기


복제 리전

0 리전

생성된 날짜

3월 23, 2023, 11:17:22 (UTC+09:00)

Amazon 리소스 이름(ARN)

 arn:aws:dynamodb:ap-northeast-2:453043152051:table/aws-learner-dynamodb-table

Time To Live(TTL) [정보](#)

⊖ 끄기

암호화

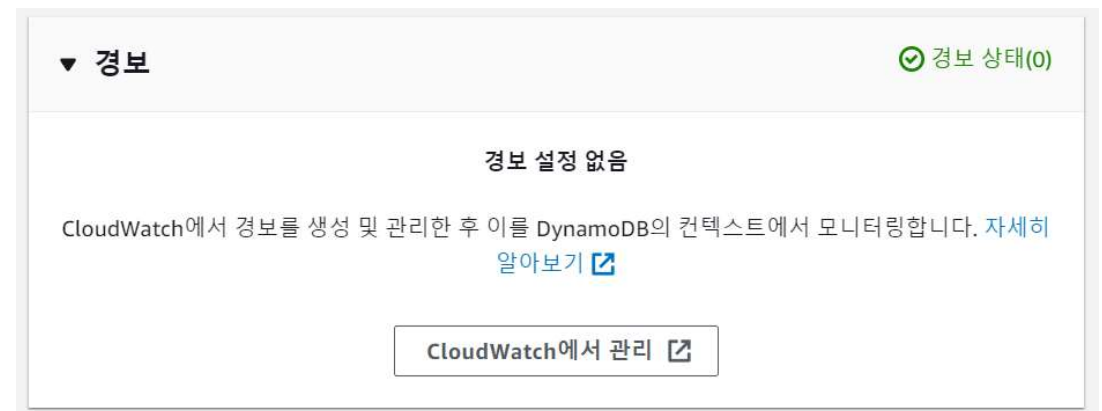
Amazon에서 소유함

삭제 방지

⊖ 끄기



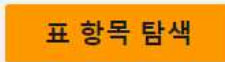
AWS DynamoDB 생성 정보

- 인덱스를 통해 글로벌 보조 인덱스를 생성해서 쿼리 성능을 올릴 수 있다.
- 모니터링을 통해 cloud watch 정보를 볼 수 있다.



DB에 값 입력

- 표 항목 탐색 들어가기
- 항목 생성 누름
- Column 만들고 값 입력.
- NoSQL이기 때문에 Column 입력이 자유롭다.

aws-learner-dynamodb-table   


반환된 항목 (0)   


< 1 >  

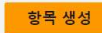
항목 없음

표시할 항목이 없습니다.



속성 

 속성 이름	값	유형	
customer_id - 파티션 키	<input type="text" value="cus153"/>	문자열	
transaction_date - 정렬 키	<input type="text" value="2023-03-18"/>	문자열	
<input type="text" value="item_category"/>	<input type="text" value="drawer"/>	문자열	
<input type="text" value="price"/>	<input type="text" value="90000"/>	숫자	

DB에 값 확인

- 반환된 항목에서 확인 할 수 있다.
- AWS 콘솔로도 쿼리를 쓸 수 있음 (쿼리, 스캔)
- 하지만 이렇게 일일이 수동으로 값을 넣는 것은 비효율적이다.
- Test가 아니면 이렇게 값을 넣을 일이 없다.

The screenshot shows the AWS DynamoDB console interface. At the top, there's a header '반환된 항목 (1)' (Returned items (1)) with buttons for '작업' (Action) and '항목 생성' (Create item). Below this is a table with columns: 'customer_id', 'transaction_date', 'item_category', and 'price'. The first row shows 'cus153', '2023-03-18', 'drawer', and '90000'. A modal window titled '항목 스캔 또는 쿼리' (Scan or query item) is open, showing options for '스캔' (Scan) and '쿼리' (Query). The '쿼리' option is selected. The modal also shows the table name 'aws-learner-dynamodb-table' and the primary key 'customer_id'. The 'transaction_date' is set to '같은' (Same) and the 'price' is '90000'. The modal has buttons for '실행' (Run) and '재설정' (Reset).

customer_id	transaction_date	item_category	price
cus153	2023-03-18	drawer	90000

▼ 항목 스캔 또는 쿼리

☐ 스캔 ☒ 쿼리

테이블 또는 인덱스 선택: 테이블 - aws-learner-dynamodb-table

속성 프로젝션 선택: 모든 속성

customer_id (파티션 키)
파티션 키 값 입력:

transaction_date (정렬 키)
정렬 키 값 입력: 같음 ☐ 내림차순 정렬

▶ 필터

실행 재설정

외부에서 코드로 데이터 삽입하기

- 실제 production에서는 일일이 AWS 콘솔로 데이터를 넣진 않는다.
- 실제로 서비스를 구현하면 코드로 값을 넣게 될 것이다.
- 이를 구현하는 방법은 크게 두가지
 1. IOT AWS를 사용하여 실시간으로 데이터를 테이블에 삽입
 2. 이벤트 감지되었을 때 Lambda 발동시켜 전처리 과정 거친 후 테이블에 데이터 삽입

Lambda에 연결해서 테스트

Lambda 함수 생성

- 파이썬으로 코드를 설정하고 lambda 생성

함수 생성 정보

AWS Serverless Application Repository applications have moved to [Create application](#).

☒ 새로 작성

간단한 Hello World 예제는 시작하십시오.

☐ 블루프린트 사용

샘플 코드 및 구축 Lambda 애플리케이션을 위한 구성 사전 설정을 일반적인 사용 사례를 살펴봅니다.

☐ 컨테이너

함수에 D

기본 정보

함수 이름

함수의 용도를 설명하는 이름을 입력합니다.

aws-learner-lambda-data-insertion

공백 없이 문자, 숫자, 하이픈 또는 밑줄만 사용합니다.

런타임 정보

함수를 작성하는 데 사용할 언어를 선택합니다. 콘솔 코드 편집기는 Node.js, Python 및 Ruby만 지원합니다.

Python 3.9



Lambda 함수에 권한 부여

- DynamoDB에 접근하려면 권한이 있어야 한다.
- '기본 실행 역할 변경'을 눌러서 권한을 부여

권한 정보

기본적으로 Lambda는 Amazon CloudWatch Logs에 로그를 업로드하는 권한을 가진 실행 역할을 생성합니다. 이 기본 역할은 나중에 트리거를 추가할 때 사용자 지정할 수 있습니다.

▼ 기본 실행 역할 변경

실행 역할

Choose a role that defines the permissions of your function. To create a custom role, go to the [IAM console](#).

- ☒ 기본 Lambda 권한을 가진 새 역할 생성
- ☐ 기존 역할 사용
- ☐ AWS 정책 템플릿에서 새 역할 생성

i 역할을 생성하는 데 몇 분 정도 걸릴 수 있습니다. 역할을 삭제하거나 이 역할에서 신뢰 또는 권한 정책을 편집하지 마십시오.

Lambda가 이름이 `aws-learner-lambda-data-insertion-role-ins3bfzn`이고 Amazon CloudWatch Logs에 로그를 업로드할 수 있는 권한이 포함

IAM 역할 만들기

- DynamoDB에 접근을 위해 IAM 역할이 필요함
- 사용 사례에서 Lambda를 고르고 다음을 누름

사용 사례 선택

일반 사용 사례

EC2

Allows EC2 instances to call AWS services on your behalf.

Lambda

Allows Lambda functions to call AWS services on your behalf.

취소

다음: 권한





IAM 역할 만들기 - 정책 생성

- 정책 생성 버튼을 누르고 서비스에는 DynamoDB를 지정, 작업에서는 쓰기를 선택하고 PutItem만 체크한다. <- 데이터 값 넣는 것은 이것으로 충분함

정책 생성

정책 필터 ▾

정책 이름 ▾

<input type="checkbox"/>	▶	 AlexaForBusinessFullAccess
<input type="checkbox"/>	▶	 AlexaForBusinessGatewayExecution
<input type="checkbox"/>	▶	 AlexaForBusinessLifesizeDelegatedAccessPolicy
<input type="checkbox"/>	▶	 AlexaForBusinessNetworkProfileServicePolicy

▶ 서비스 DynamoDB

<input type="checkbox"/>	PurchaseReservedCapacityOffe...
<input checked="" type="checkbox"/>	PutItem ?
<input type="checkbox"/>	RestoreTableFromAwsBackup ?

IAM 역할 만들기 – 정책의 리소스

- 정책 생성에서 리소스는 ARN을 의미한다.
- 모든 리소스를 지정하면 모든 테이블에 접근할 수 있고
- 특정 리소스를 지정하면 특정한 테이블에만 접근할 수 있음

▼ 리소스
닫기

☐ 특정
☒ 모든 리소스

IAM 역할 만들기 – 정책 활용

- 방금 만든 정책으로 IAM 역할을 만든다.

정책 생성

정책 필터

정책 이름

☐

 AdministratorAccess-AWSElasticBeanstalk

☐

 AmazonEC2RoleforAWSCodeDeploy

☐

 AmazonEC2RoleforAWSCodeDeployLimited☐☒

취소

이전

다음: 태그

역할 만들기

검토

생성하기 전에 아래에 필요한 정보를 입력하고 이 역할을 검토하십시오.

역할 이름*

aws-learner-dynamodb-role

영숫자 및 '+', '@', '_' 문자를 사용합니다. 최대 64자입니다.

취소

이전

역할 만들기

Lambda에 역할 삽입

- 실행 역할 – 기존 역할 사용 체크
- 기존 역할에서 방금 만든 역할을 선택

실행 역할

Choose a role that defines the permissions of your function. To create

- ☐ 기본 Lambda 권한을 가진 새 역할 생성
- ☒ 기존 역할 사용
- ☐ AWS 정책 템플릿에서 새 역할 생성

기존 역할

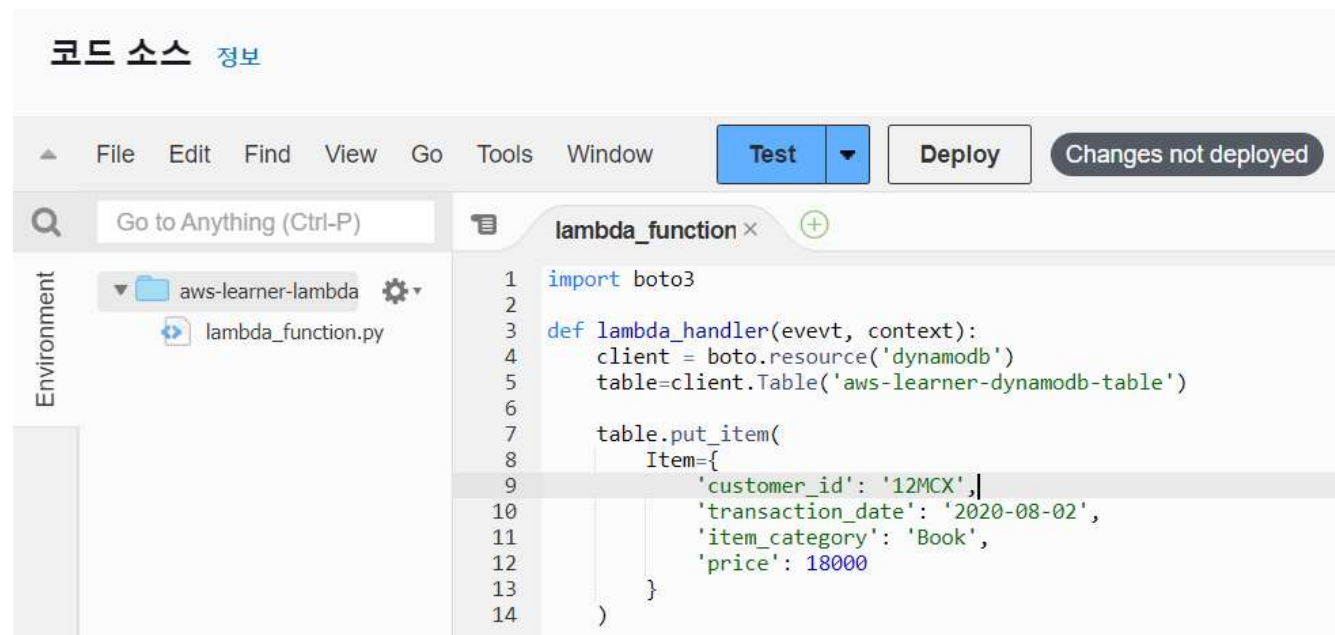
생성한 기존 역할 중에 이 Lambda 함수와 함께 사용할 역할을 선택합니다

aws-learner-dynamodb-role

[View the aws-learner-dynamodb-role role](#)  on the IAM console.

Lambda에 코드 삽입

- 역할 설정이 끝나고 나면 Lambda 에서 돌아갈 소스 코드를 작성할 수 있다.
- 처음 만들 때 파이썬으로 설정했기 때문에 파이썬 파일 자동 생성
- Boto3: AWS의 거의 모든 것을 사용할 수 있게 해주는 SDK
- Resource 안에 사용할 리소스를 적는다.
- 그렇게 만든 객체의 메서드 Table을 사용하면 테이블이 생성됨



코드 소스 정보

File Edit Find View Go Tools Window Test Deploy Changes not deployed

Go to Anything (Ctrl-P)

Environment

- aws-learner-lambda
 - lambda_function.py

```
1 import boto3
2
3 def lambda_handler(event, context):
4     client = boto.resource('dynamodb')
5     table=client.Table('aws-learner-dynamodb-table')
6
7     table.put_item(
8         Item={
9             'customer_id': '12MCX',
10            'transaction_date': '2020-08-02',
11            'item_category': 'Book',
12            'price': 18000
13        }
14    )
```

Lambda에 코드 deploy 후 test

- Deploy를 누르면 코드가 반영이 된다.
- Test를 누르면 테스트 이벤트를 구성할 수 있음
- Test를 한 번 더 누르면 테스트 이벤트가 실행됨

테스트 이벤트 구성

테스트 이벤트는 Lambda 함수를 호출하기 위해 AWS 서비스에서 생성하는 요청의 구조를 모방한 JSON 객체입니다. 이 객체를 사용하여 함수의 호출 결과를 확인합니다.

이벤트를 저장하지 않고 함수를 호출하려면 JSON 이벤트를 구성한 다음 테스트를 선택합니다.

이벤트 작업 테스트

새 이벤트 생성

저장된 이벤트 편집

이벤트 이름

awslearnerdynamodbtest

문자, 숫자, 점, 하이픈 및 밑줄을 사용하여 최대 25자로 구성합니다.

이벤트 공유 설정

프라이빗

공유 가능

이 이벤트는 공유 가능한 이벤트에 액세스하고 이를 사용할 수 있는 권한이 있는 동일한 계정 내 IAM 사용자가 사용할 수 있습니다.
[자세히 알아보기](#)

템플릿 - 선택 사항

hello-world

Go Tools Window

Test

Deploy

lambda_function ×

Execution result: ×

Execution results

Test Event Name

awslearnerdynamodbtest

Response

null

Function Logs

START RequestId: 56463e7a-5bc7-48e7-bc72-ee34a8c1d635 Version: \$L
END RequestId: 56463e7a-5bc7-48e7-bc72-ee34a8c1d635
REPORT RequestId: 56463e7a-5bc7-48e7-bc72-ee34a8c1d635 Duration:

Request ID




56463e7a-5bc7-48e7-bc72-ee34a8c1d635



Lambda test 실행 후 dynamoDB 확인

- 다시 dynamoDB로 돌아가면 12MCX라는 항목이 생긴 것을 확인
- 코드에 입력한 대로 잘 들어갔음

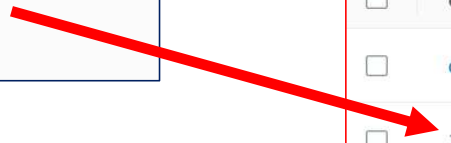
```
table.put_item(  
    Item={  
        'customer_id': '12MCX',  
        'transaction_date': '2020-08-02',  
        'item_category': 'Book',  
        'price': 18000  
    }  
)
```

반환된 항목 (2)

< 1 >  

<input type="checkbox"/>	customer_id ▾	transaction_date ▾	item_category ▾	price
<input type="checkbox"/>	cus153	2023-03-18	drawer	90000
<input type="checkbox"/>	12MCX	2020-08-02	Book	18000



값 여러 개 동시에 넣기

여러 아이템 동시에 넣기

- Boto3의 batch를 활용하면 동시에 데이터를 넣을 수 있음



The screenshot shows the AWS Lambda console with a tab for 'lambda_function' and a sub-tab for 'Execution results'. The code is a Python lambda function that uses Boto3 to write three items to a DynamoDB table named 'aws-learner-dynamodb-table' in a single batch operation. The items are represented as dictionaries with keys for customer_id, transaction_date, item_category, and price.

```
1 import boto3
2
3 def lambda_handler(event, context):
4     client = boto3.resource('dynamodb')
5     table=client.Table('aws-learner-dynamodb-table')
6
7     with table.batch_writer() as batch:
8         batch.put_item(
9             Item={
10                 'customer_id':'95IUZ',
11                 'transaction_date': '2020-10-24',
12                 'item_category':'Desk',
13                 'price':120000
14             }
15         )
16
17         batch.put_item(
18             Item={
19                 'customer_id':'75MUE',
20                 'transaction_date': '2020-10-28',
21                 'item_category':'Desk',
22                 'price':250000
23             }
24         )
25
26         batch.put_item(
27             Item={
28                 'customer_id':'28POR',
29                 'transaction_date': '2020-11-05',
30                 'item_category':'Desk',
31                 'price':50000
32             }
33         )
```

[ERROR] Batch넣기 권한 없음

- Test를 돌리면 에러가 발생
- BatchWriteItem 권한이 없다고 뜬다.

```
role/aws-learner-lambda-data-insertion is not authorized to perform: dynamodb:BatchWriteItem on resource:
```

- IAM 정책 편집을 해줘야 한다.
- BatchWriteItem 권한 추가 -> 테스트 성공

▼ ☐ 쓰기 (2개 선택)

☒ BatchWriteItem ?

Test Event Name

awslearnerdynamodbtest

Response

null

Function Logs

START RequestId: 1cfa3995-5327-4ae5-80fc-510b11cd17a1 Version: \$LATEST
END RequestId: 1cfa3995-5327-4ae5-80fc-510b11cd17a1
REPORT RequestId: 1cfa3995-5327-4ae5-80fc-510b11cd17a1 Duration: 1289.47 ms