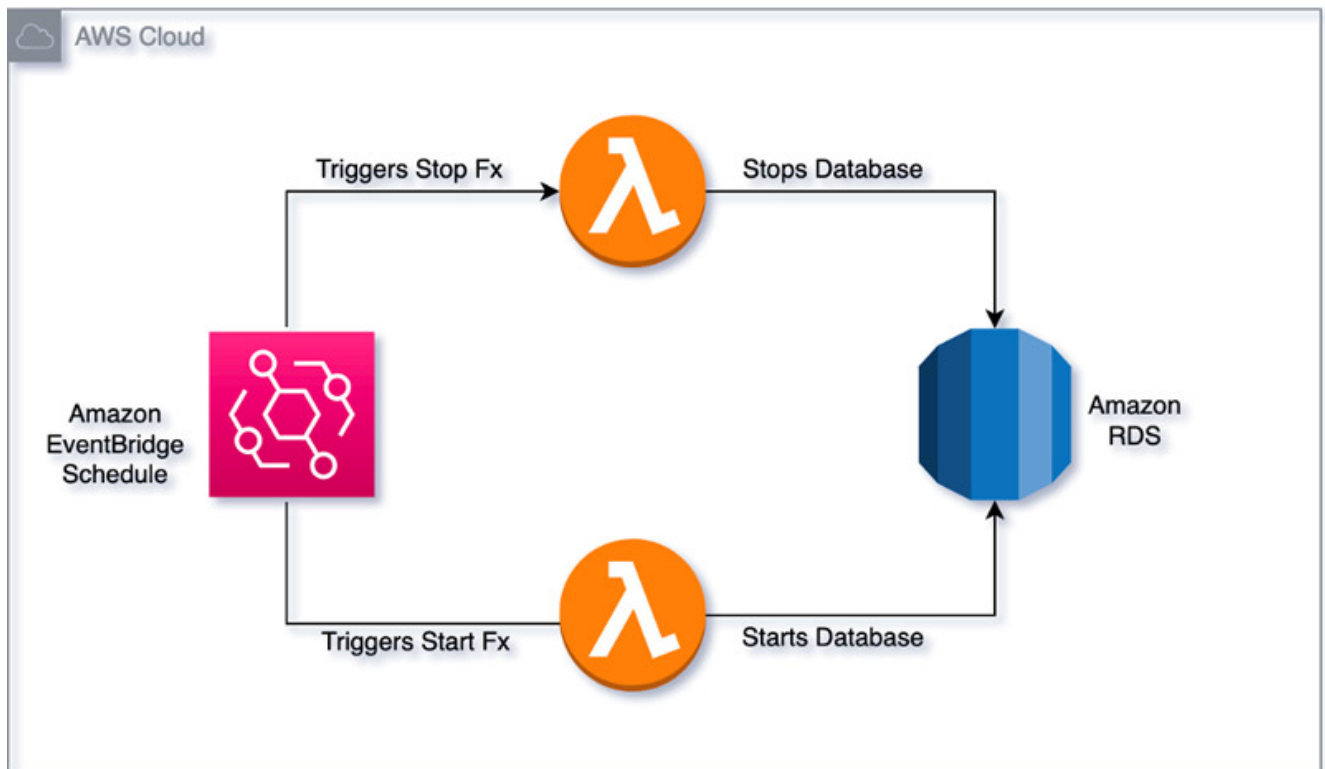


[AWS] Lambda + EventBridge 를 통한 RDS 데이터베이스를 자동으로 중지 및 실행하기

RDS 인스턴스는 DB 인스턴스 시간, 프로비저닝된 스토리지, 백업 스토리지, I/O 요청, 프로비저닝된 IOPS 및 데이터 전송에 대해 청구됩니다. 사용하지 않는 리소스들을 중지하지 않을 경우 지속적으로 비용이 발생하며 이 문서에서는 **Amazon EventBridge**와 **AWS Lambda**의 조합을 사용하여 일정에 따라 **모든 지역**에 대하여 미사용 RDS 인스턴스를 중지/시작하기 위한 자동화된 솔루션을 구현하는 방법을 살펴해보겠습니다.



단계 :

1. 사전 구성
2. IAM 정책 및 역할 생성
3. Lambda 함수 생성
4. EventBridge 규칙 생성

5. 테스트

1. 사전 구성

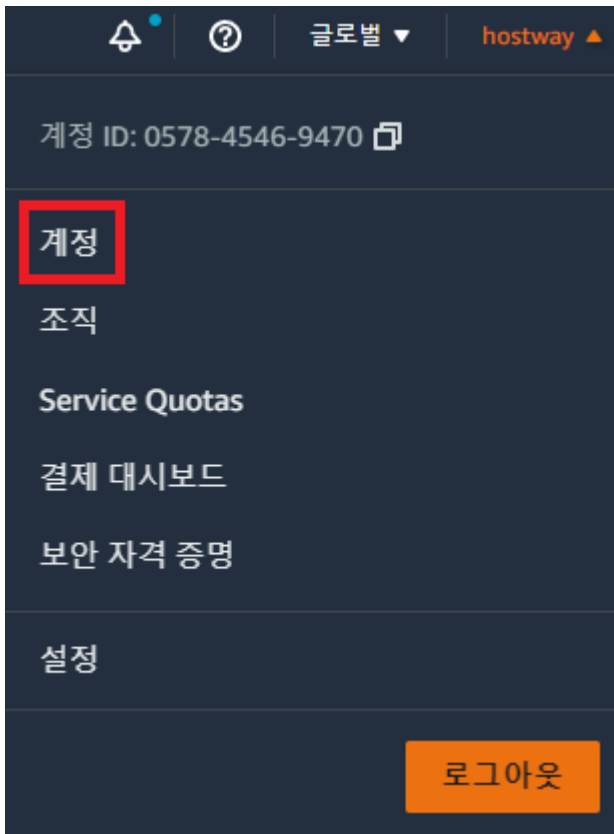
AWS 서비스에 프로그래밍 방식으로 연결하려면 엔드포인트를 사용해야 합니다.

그러나 **일부 리전(opt-in)**들은 기본적으로 비활성화 되어 있으며 해당 리전들의 DB 인스턴스 정보를 가져오기 위해선 활성화를 진행해주어야 합니다.

opt-in region

- 아프리카(케이프타운)
- 아시아 태평양(홍콩)
- 유럽(밀라노)
- 중동(바레인)
- 아시아 태평양(자카르타)

1. 권한이 있는 계정으로 콘솔에 로그인 후 우측 상단 계정 선택 - 계정- AWS 리전 비활성화 되어있는 5개의 리전들을 활성화 상태로 변경합니다.



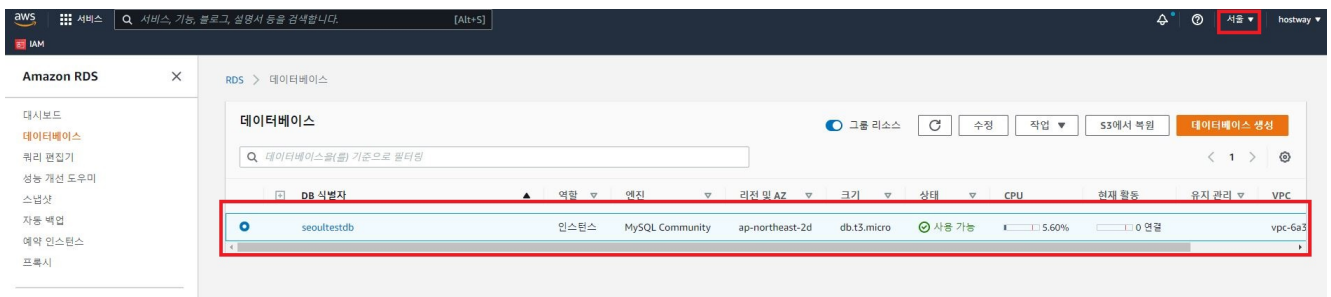
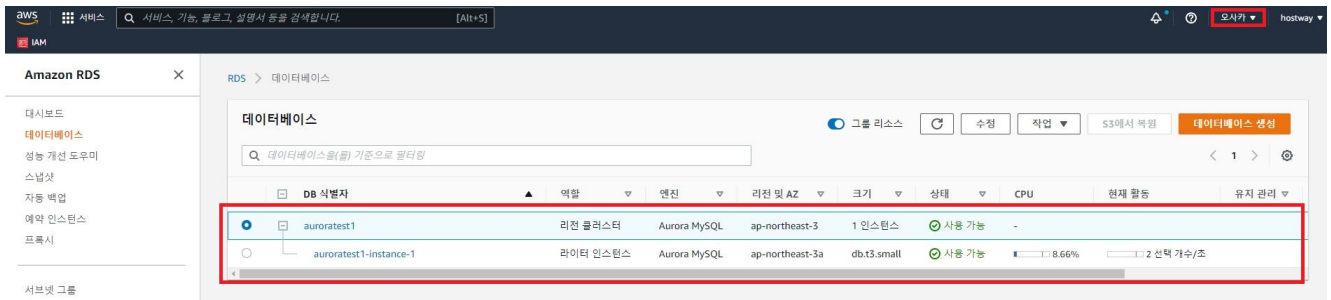
리전	상태	작업
중동 (바레인)	비활성화됨	활성화
아시아 태평양 (자카르타)	비활성화됨	활성화
아프리카 (케이프타운)	비활성화됨	활성화
아시아 태평양 (홍콩)	비활성화됨	활성화
유럽 (밀라노)	비활성화됨	활성화

2. 테스트를 진행할 DB 인스턴스를 생성합니다. 총 3개 리전에 대하여 테스트 해보겠습니다.

홍콩 : MariaDB

오사카 : AWS Aurora

서울 : Mysql



2. IAM 정책 및 역할 생성

1. RDS Stop/Start 역할을 위한 정책을 생성합니다. IAM 콘솔에서 정책 - 정책 생성


```

    "Effect": "Allow",
    "Action": [
        "rds:StartDBCluster",
        "rds:StopDBCluster",
        "rds:ListTagsForResource",
        "rds:DescribeDBInstances",
        "rds:StopDBInstance",
        "rds:DescribeDBClusters",
        "rds:StartDBInstance"
    ],
    "Resource": "*"
}
]
}

```

3. **RDS-Stop-Start-Policy** 이름 입력 후 정책 생성을 클릭합니다.

정책 생성

1 2 3

정책 검토

이름*

영숫자 및 '+,.,@,_' 문자를 사용합니다. 최대 128자입니다.

설명

최대 1000자입니다. 영숫자 및 '+,.,@,_' 문자를 사용합니다.

요약

Q 필터:

서비스 ▾	엑세스 레벨	리소스	요청 조건
허용 (1 / 331 서비스) 나머지 330 표시			
RDS	제한: 목록, 읽기, 쓰기	모든 리소스	없음

태그

키 값

리소스와 연결된 태그가 없습니다.

* 필수

취소

이전

정책 생성

4. RDS Stop/Start 를 위한 역할을 생성합니다. IAM 콘솔에서 역할 - 역할 만들기

Identity and Access Management(IAM)

대시보드
역세스 관리
사용자 그룹
사용자
역할
정책
자격 증명 공급자
계정 설정
보고서 액세스
역세스 분석기
아카이브 규칙
분석기
설정
자격 증명 보고서
조직 활동
SCP(서비스 제어 정책)
권한 콘솔
IAM Identity Center
신규

IAM > 역할

역할 (79) 정보
IAM 역할은 단기간 동안 유효한 자격 증명을 가진 특정 권한이 있는 자격 증명입니다. 신뢰할 수 있는 개체가 역할을 맡을 수 있습니다.
 검색

1
2
3
4

☐ 역할 이름
신뢰할 수 있는 개체
마지막 활동

☐	AWSServiceRoleForApplicationMigrationService	AWS 서비스: mgn (서비스 연결 역할)	29분 전
☐	AWSServiceRoleForAutoScaling	AWS 서비스: autoscaling (서비스 연결 역할)	7일 전
☐	AWSServiceRoleForAWSLicenseManagerRole	AWS 서비스: license-manager (서비스 연결 역할)	142일 전
☐	AWSServiceRoleForBackup	AWS 서비스: backup (서비스 연결 역할)	18시간 전
☐	AWSServiceRoleForClientVPN	AWS 서비스: clientvpn (서비스 연결 역할)	81일 전
☐	AWSServiceRoleForClientVPNConnections	AWS 서비스: clientvpn-connections (서비스 연결 역할)	-
☐	AWSServiceRoleForCloudFrontLogger	AWS 서비스: logger.cloudfront (서비스 연결 역할)	90일 전
☐	AWSServiceRoleForCloudWatchCrossAccount	AWS 서비스: cloudwatch-crossaccount (서비스 연결 역할)	136일 전
☐	AWSServiceRoleForCloudWatchEvents	AWS 서비스: events (서비스 연결 역할)	-
☐	AWSServiceRoleForCodeStarNotifications	AWS 서비스: codestar-notifications (서비스 연결 역할)	209일 전
☐	AWSServiceRoleForConfig	AWS 서비스: config (서비스 연결 역할)	209일 전
☐	AWSServiceRoleForElasticCache	AWS 서비스: elasticsearch (서비스 연결 역할)	91일 전
☐	AWSServiceRoleForElasticLoadBalancing	AWS 서비스: elasticloadbalancing (서비스 연결 역할)	5일 전
☐	AWSServiceRoleForGlobalAccelerator	AWS 서비스: globalaccelerator (서비스 연결 역할)	-

5. AWS Lambda를 신뢰할 수 있는 엔터티로 선택 후 다음을 클릭합니다.

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔터티 선택
2단계
권한 추가
3단계
이름 지정, 검토 및 생성

신뢰할 수 있는 엔터티 선택
신뢰할 수 있는 엔터티 유형

☒ AWS 서비스
EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.

☐ AWS 계정
사용자 또는 서드 파티에 속한 다른 AWS 계정의 엔터티가 이 계정에서 작업을 수행하도록 허용합니다.

☐ 웹 자격 증명
지정된 외부 웹 자격 증명 공급자와 연동된 사용자가 이 역할을 맡아 이 계정에서 작업을 수행하도록 허용합니다.

☐ SAML 2.0 연동
기밀 디렉터리에서 SAML 2.0과 연동된 사용자가 이 계정에서 작업을 수행할 수 있도록 허용합니다.

☐ 사용자 지정 신뢰 정책
다른 사용자가 이 계정에서 작업을 수행할 수 있도록 사용자 지정 신뢰 정책을 생성합니다.

사용 사례
EC2, Lambda 등의 AWS 서비스가 이 계정에서 작업을 수행하도록 허용합니다.
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.
다른 AWS 서비스의 사용 사례:
사용 사례를 조회할 서비스 선택

취소
다음

6. 생성한 정책 RDS-Stop-Start-Policy 필터링 후 선택 - 다음을 클릭합니다.

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔터티 선택
2단계
권한 추가
3단계
이름 지정, 검토 및 생성

권한 추가

권한 정책 (선택된 1/804)
새 역할에 연결할 정책을 하나 이상 선택합니다.
 1개 일치

1

☒ 정책 이름
☒ RDS-STOP-START-

☒	정책 이름	유형	설명
☒	RDS-Stop-Start-Policy	고려 관...	

▶ 권한 경계 설정 - 선택 사항
권한 경계를 설정하여 이 역할이 가질 수 있는 최대 권한을 제어합니다. 이는 일반적인 설정은 아니지만 권한 관리를 다른 사용자에게 위임하는 데 사용할 수 있습니다.

취소
이전
다음

7. RDS-Stop-Start-Role 이름 입력 후 역할 생성을 클릭합니다.

IAM > 역할 > 역할 생성

1단계
신뢰할 수 있는 엔티티 선택

2단계
권한 추가

3단계
이름 지정, 검토 및 생성

이름 지정, 검토 및 생성

역할 세부 정보

역할 이름
이 역할을 신뢰하는 엔티티 이름은 이 이름과 일치합니다.
RDS-Stop-Start-Role
최대 64자입니다. 영문자 및 '*'='.'='@'='_' 문자를 사용하세요.

설명
이 역할에 대하여 간단한 설명을 추가합니다.
Allows Lambda functions to call AWS services on your behalf.
최대 1,000자입니다. 영문자 및 '*'='.'='@'='_' 문자를 사용하세요.

3. Lambda 함수 생성

AWS Lambda 함수를 생성하여 RDS DB 인스턴스를 Stop 시키도록 하겠습니다.

Region은 어느 지역이던 상관 없으나 여기선 서울 Region에 생성해보겠습니다.

1. RDS 인스턴스를 Stop할 Lambda 함수를 생성합니다. Lambda 콘솔 함수 - 함수 생성

AWS Lambda

대시보드
애플리케이션
함수
추가 리소스
코드 서명 구성
계정
복제본
관련 AWS 리소스
Step Functions 상태 머신

Lambda > 함수

함수 (0) 마지막으로 가져온 항목 35초 전 작업 **함수 생성**

Q 태그 및 속성별 필터 또는 키워드별 검색

함수 이름	설명	패키지 유형	런타임	마지막 수정
표시할 데이터가 없습니다.				

2. 다음과 같이 입력 후 함수를 생성합니다.

함수 이름 : StopRDS

런타임 : Python 3.9

권한 : 기본 실행 역할 변경 - 기존 역할 사용 - 생성한 역할 RDS-Stop-Start-Role 선택

함수 이름
 함수의 용도를 설명하는 이름을 입력합니다.

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.

아키텍처 정보
 함수 코드에 대해 원하는 명령 세트 아키텍처를 선택합니다.
☒ x86_64
☐ arm64

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
 함수에 대한 권한을 정의하는 역할을 선택합니다. 사용자 지정 역할을 생성하려면 IAM 콘솔로 이동합니다.

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

기존 역할
 생성된 기존 역할 중에 이 Lambda 함수와 함께 사용할 역할을 선택합니다. 이 역할에는 Amazon CloudWatch Logs에 로그를 업로드할 수 있는 권한이 있어야 합니다.

IAM 콘솔에서 RDS-Stop-Start-Role 역할을 확인합니다.

▶ 고급 설정

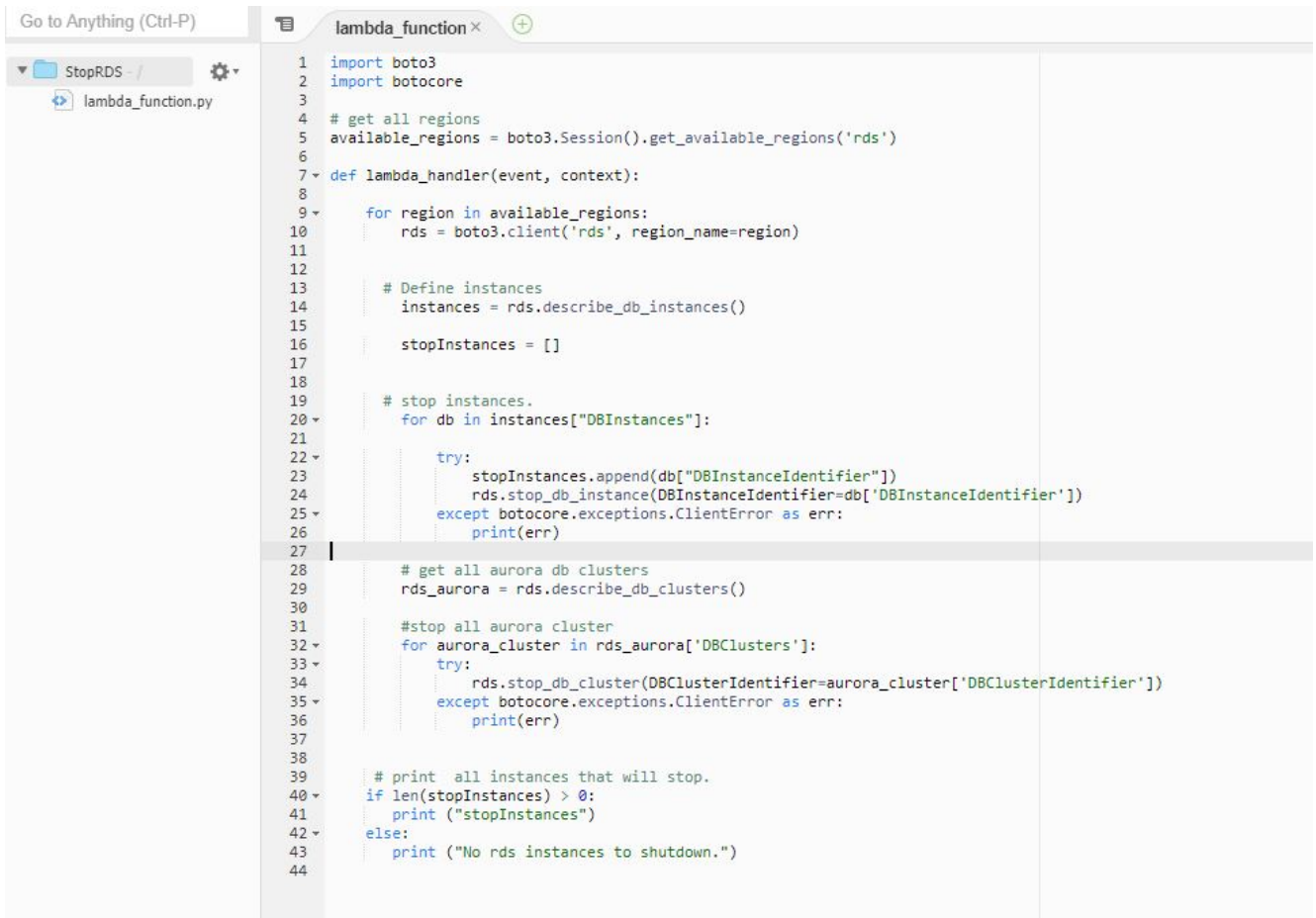
취소

3. Boto3, Botocore를 이용하여 Python 코드 작성을 진행합니다.

Boto : AWS에서 Python 프로그래밍 언어의 사용을 개선하도록 설계된 소프트웨어 개발 키트(SDK)이며 가장 최신 버전인 **Boto3** 까지 나와있습니다.

□

Botocore : AWS 도구에 대해 기본적인 액세스를 제공하며 low-level의 클라이언트 요청을 만들고 API에서 결과를 가져옵니다.



```
1 import boto3
2 import botocore
3
4 # get all regions
5 available_regions = boto3.Session().get_available_regions('rds')
6
7 def lambda_handler(event, context):
8
9     for region in available_regions:
10         rds = boto3.client('rds', region_name=region)
11
12         # Define instances
13         instances = rds.describe_db_instances()
14
15         stopInstances = []
16
17         # stop instances.
18         for db in instances["DBInstances"]:
19             try:
20                 stopInstances.append(db["DBInstanceIdentifier"])
21                 rds.stop_db_instance(DBInstanceIdentifier=db["DBInstanceIdentifier"])
22             except botocore.exceptions.ClientError as err:
23                 print(err)
24
25     # get all aurora db clusters
26     rds_aurora = rds.describe_db_clusters()
27
28     #stop all aurora cluster
29     for aurora_cluster in rds_aurora["DBClusters"]:
30         try:
31             rds.stop_db_cluster(DBClusterIdentifier=aurora_cluster["DBClusterIdentifier"])
32         except botocore.exceptions.ClientError as err:
33             print(err)
34
35     # print all instances that will stop.
36     if len(stopInstances) > 0:
37         print ("stopInstances")
38     else:
39         print ("No rds instances to shutdown.")
40
41
42
43
44
```

StopRDS 함수 코드

```
import boto3
import botocore

# get all regions
available_regions = boto3.Session().get_available_regions('rds')

def lambda_handler(event, context):

    for region in available_regions:
        rds = boto3.client('rds', region_name=region)

        # Define instances
        instances = rds.describe_db_instances()

        stopInstances = []
```

```

        # stop instances.
        for db in instances["DBInstances"]:
            try:

                stopInstances.append(db["DBInstanceIdentifier"])

                rds.stop_db_instance(DBInstanceIdentifier=db['DBInstanceIdentifier'])
            except botocore.exceptions.ClientError as err:
                print(err)

            # get all aurora db clusters
            rds_aurora = rds.describe_db_clusters()

            # stop all aurora cluster
            for aurora_cluster in rds_aurora['DBClusters']:
                try:

                    rds.stop_db_cluster(DBClusterIdentifier=aurora_cluster['DBClusterIdentifier'])
                except botocore.exceptions.ClientError as err:
                    print(err)

            # print all instances that will stop.
            if len(stopInstances) > 0:
                print ("stopInstances")
            else:
                print ("No rds instances to shutdown.")

```

stop_db_instance API는 Aurora 클러스터의 일부인 인스턴스를 종료할 수 없기 때문에 별도로 Aurora DB에 대하여 **stop_db_cluster** API 호출을 해주어야 합니다.

StartRDS 함수 작성을 하고자 할 경우 아래 코드를 입력하여 함수를 생성하면 됩니다.

StartRDS 함수 코드

```

import boto3
import botocore

```

```

# get all regions
available_regions
boto3.Session().get_available_regions('rds')
=
[]
def lambda_handler(event, context):
[]
    for region in available_regions:
        rds = boto3.client('rds', region_name=region)
[]
[]
        # Define instances
        instances = rds.describe_db_instances()
[]
        startInstances = []
[]
[]
        # start instances.
        for db in instances["DBInstances"]:
[]
            try:

startInstances.append(db["DBInstanceIdentifier"])

        rds.start_db_instance(DBInstanceIdentifier=db['DBInstanceIdentifier'])
            except botocore.exceptions.ClientError as err:
                print(err)
[]
        # get all aurora db clusters
        rds_aurora = rds.describe_db_clusters()
[]
        # stop all aurora cluster
        for aurora_cluster in rds_aurora['DBClusters']:
            try:

rds.start_db_cluster(DBClusterIdentifier=aurora_cluster['DBClusterIdentifier'])
            except botocore.exceptions.ClientError as err:
                print(err)
[]
[]

```

```
# print all instances that will stop.
if len(startInstances) > 0:
    print ("startInstances")
else:
    print ("No rds instances to start.")
```

4. EventBridge 규칙 생성

1. Amazon EventBridge 콘솔에서 규칙 - 규칙 생성을 클릭합니다.



2. 규칙 세부 정보에서 **RDS_Instances_Stop** 규칙 이름으로 입력 및 규칙 유형 일정 선택 후 다음을 클릭합니다.

Amazon EventBridge > 규칙 > 규칙 생성

1단계
규칙 세부 정보 정의

2단계
일정 정의

3단계
대상 선택

4단계 - 선택 사항
태그 구성

5단계
검토 및 생성

규칙 세부 정보 정의 정보

규칙 세부 정보

이름

RDS_Instances_Stop

숫자, 대/소문자, 마침표(.), 대시(-), 밑줄(_)를 사용하여 최대 64자로 지정합니다.

설명 - 선택 사항

Stop DB Instances

이벤트 버스 정보

이 규칙이 적용되는 이벤트 버스(기본 이벤트 버스 또는 사용자 지정 또는 파트너 이벤트 버스)를 선택합니다.

default ▼

☒ 선택한 이벤트 버스에 대해 규칙 활성화

규칙 유형 정보

☐ 이벤트 패턴이 있는 규칙
이벤트가 정의된 이벤트 패턴과 일치할 때 실행되는 규칙입니다. EventBridge는 지정된 대상으로 이벤트를 전송합니다.

☒ 일정
일정에 따라 실행되는 규칙입니다.

[취소](#) [다음](#)

3. Cron 표현식을 작성합니다. 0 9 * * ? *

아래는 매일 오후 6시에 Lambda 함수를 트리거하는 표현식입니다.

자세한 cron 표현식은 공식 문서에 잘 나와 있습니다.

https://docs.aws.amazon.com/ko_kr/lambda/latest/dg/services-cloudwatchevents-expressions.html

Amazon EventBridge > 규칙 > 규칙 생성

1단계
규칙 세부 정보 정의

2단계
일정 정의

3단계
대상 선택

4단계 - 선택 사항
태그 구성

5단계
검토 및 생성

일정 정의

일정 패턴

요구 사항에 가장 잘 맞는 일정 유형을 선택합니다.

☒ 매월 첫 번째 월요일 오전 8시(PST)와 같이 특정 시
간에 실행되는 세분화된 일정입니다.

☐ 10분마다와 같이 일정한 빈도로 실행되는 일정입
니다.

Cron 식 정보

일정에 대한 cron 표현식 정의

cron (

0

)

분

시간

일

월

요일

연도

0

9

*

*

?

*

다음 10개 트리거 날짜

현지 시간대

2022년 8월 8일 오후 06:00 GMT+9

2022년 8월 9일 오후 06:00 GMT+9

2022년 8월 10일 오후 06:00 GMT+9

2022년 8월 11일 오후 06:00 GMT+9

2022년 8월 12일 오후 06:00 GMT+9

2022년 8월 13일 오후 06:00 GMT+9

2022년 8월 14일 오후 06:00 GMT+9

2022년 8월 15일 오후 06:00 GMT+9

2022년 8월 16일 오후 06:00 GMT+9

2022년 8월 17일 오후 06:00 GMT+9

취소

이전

다음

4. AWS 서비스 선택 후 **StopRDS** 함수를 대상으로 지정한 뒤 다음을 클릭합니
다.

Amazon EventBridge > 규칙 > 규칙 생성

1단계
규칙 세부 정보 정의

2단계
일정 정의

3단계
대상 선택

4단계 - 선택 사항
태그 구성

5단계
검토 및 생성

대상 선택

권한
참고: EventBridge 콘솔을 사용할 때, EventBridge는 선택된 대상에 대한 적절한 권한을 자동으로 구성합니다. AWS CLI, SDK 또는 CloudFormation을 사용하는 경우 적절한 권한을 구성해야 합니다.

대상 1개

대상 유형
EventBridge 이벤트 버스, EventBridge API 대상(SaaS 파트너) 또는 다른 AWS 서비스를 대상으로 선택합니다.

☐ EventBridge 이벤트 버스
☐ EventBridge API 대상
☒ **AWS 서비스**

대상 선택 | 정보
이벤트가 이벤트 파티션과 일치하거나 일정이 트리거될 때 호출할 대상을 선택합니다(규칙당 5개의 대상으로 제한).

Lambda 함수

함수
StopRDS

▶ 버전/별칭 구성

▶ 추가 설정

다른 대상 추가

취소 | 이전 | **다음**

5. EventBridge 규칙이 정상적으로 생성되었습니다.

Amazon EventBridge > 규칙

규칙

규칙은 특정 유형의 이벤트를 감시합니다. 일치하는 이벤트가 발생하면 규칙에 연결된 대상으로 해당 이벤트가 라우팅됩니다. 규칙은 하나 이상의 대상에 연결할 수 있습니다.

이벤트 버스 선택

이벤트 버스
이벤트 버스 이름을 선택하거나 입력하십시오.
default

규칙 (4/4)

Q 규칙 찾기 | 모든 상태

이름	상태	유형	설명
☐ AutoScalingManagedRule	Enabled	관리형	This rule is used to route Instance notifications to EC2 Auto Scaling
☐ RDS_Instances_Start	Disabled	예약된 표준	Start DB Instances
☐ RDS_Instances_Stop	Enabled	예약된 표준	Stop DB Instances
☐ StopEC2Instance	Enabled	예약된 표준	

5. 테스트

생성한 Lambda 함수, EventBridge 규칙으로 3개 Region DB 들에 대하여 테스트를 진행해보겠습니다.

1. Amazon Lambda 콘솔에서 **StopRDS** 선택 후 테스트를 클릭, **StopTest** 이름을 작성한 뒤 기본 템플릿 hello-world 로 저장합니다.

The screenshot shows the 'Test' tab in the Amazon Lambda console. The 'Test Event' section is active, and the 'Save' button is highlighted with a red box. The 'Event Name' field is set to 'StopTest', also highlighted with a red box. The 'Event Type' is set to 'Function'. The 'Template' dropdown is set to 'hello-world'. The 'Event JSON' section shows a sample JSON object with three key-value pairs.

코드 | **테스트** | 모니터링 | 구성 | 별칭 | 버전

테스트 이벤트 저장 **테스트**

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

이벤트 작업 테스트

☒ 새 이벤트 생성 ☐ 저장된 이벤트 편집

이벤트 이름

StopTest

문자, 숫자, 언더바 및 밑줄을 사용하여 최대 25자로 구성합니다.

이벤트 공유 설정

☒ 프라이빗
이 이벤트는 Lambda 콘솔 및 이벤트 생성자만 사용할 수 있습니다. 총 10개를 구성할 수 있습니다. 자세히 알아보기

☐ 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

이벤트 JSON JSON 형식 지정

```
1 {  
2   "key1": "value1",  
3   "key2": "value2",  
4   "key3": "value3"  
5 }
```

2. 저장된 템플릿 에서 테스트를 클릭합니다.

코드 | **테스트** | 모니터링 | 구성 | 별칭 | 버전

테스트 이벤트 삭제 저장 **테스트**

이벤트를 저장하지 않고 함수를 호출하려면 이벤트를 수정한 다음 테스트를 선택합니다. Lambda는 수정된 이벤트를 사용하여 함수를 호출하지만 변경 사항 저장을 선택할 때까지 원래 이벤트를 덮어쓰지 않습니다.

이벤트 작업 테스트
☐ 새 이벤트 생성 ☒ 저장된 이벤트 편집

이벤트 이름
 StopTest ↻

이벤트 JSON JSON 형식 지정

```

1 {
2   "key1": "value1",
3   "key2": "value2",
4   "key3": "value3"
5 }
```

3. 테스트가 정상적으로 성공했다면 다음과 같은 화면을 볼 수 있습니다.

코드 | **테스트** | 모니터링 | 구성 | 별칭 | 버전

실행 결과: 성공 (로그) ×
 ▶ 세부 정보

테스트 이벤트 삭제 저장 **테스트**

4. 이제 3개 리전에 대한 RDS 데이터베이스를 확인해보겠습니다.

데이터베이스

그룹 리소스

수정

작업

S3에서 복원

데이터베이스 생성

Q 데이터베이스(들) 기준으로 필터링

DB 식별자

mariaDbhongkong

역할

인스턴스

엔진

MariaDB

클래스

db.t3.micro

상태

중지 중

CPU

-

현재 활동

0 연결

유지 관리

VPC

vpc-0ea2134f0c1c6990a

다중 AZ

예

데이터베이스

그룹 리소스

↺

수정

작업 ▼

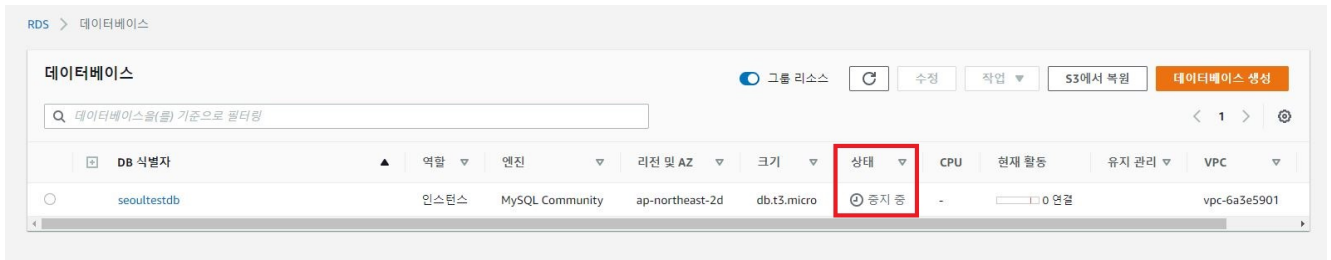
S3에서 복원

데이터베이스 생성

Q 데이터베이스(들) 기준으로 필터링

< 1 > ⚙

DB 식별자	역할	엔진	리전 및 AZ	크기	상태	CPU	현재 활동	유지 관리	VPC
auroratest1	리전 클러스터	Aurora MySQL	ap-northeast-3	1 인스턴스	🕒 중지 중	-			
auroratest1-instance-1	라이터 인스턴스	Aurora MySQL	ap-northeast-3a	db.t3.small	🕒 중지 중	-	2 선택 개수/초		vpc-0d1fa



서울 리전 **Mysql** 인스턴스, opt-in 리전인 홍콩 **Mssql** 정상적으로 Stop 되고 있으며
오사카 리전 **Aurora DB Cluster**도 정상적으로 Stop 중인 상태를 확인할 수 있습니다.

감사합니다.