AWS RDS

작성일: 2024-08-20
작성자: 박성우

(참조:
https://velog.io/@ghldjfldj/AWS-RDS%EB%9E%80-%EB%AC%B4%EC%97%87%EC%9D%B8%EA%B0%80-zuaaizv4)

RDS(Relational Database Service) 란 무엇인가?

간단히 말하면 관계형 데이터베이스를 제공하는 AWS의 서비스이다. 유저가 사용하기 쉽도록 인프라 등을 자동화 시켜주고 유저들은 앤드포인트로 접속할 수 있는 데이터베이스를 제공받는다.

아마존에서 말하는 RDS

Amazon RDS를 사용하면 클라우드에서 관계형 데이터베이스를 간편하게 설정, 운영 및 확장할 수 있다. 하드웨어 프로비저닝, 데이터베이스 설정, 패치 및 백업과 같은 시간 소모적인 관리 작업을 자동화 하면서, 비용 효율적이고 크기 조정 가능한 용량을 제공한다. 사용자가 애플리케이션에 집중해 애플리케이션에 필요한 빠른 성능, 고가용성, 보안 및 호환성을 제공할 수 있도록 지원한다.

RDS 특징

- 관계형 데이터베이스를 제공하는 서비스
    - Relational Database Service: 관계형 데이터베이스

- 가상머신 위에서 동작
    - 단 시스템에 직접 로그인 불가 -> OS패치, 관리 등은 AWS의 역할, RDS는 Serverless 서비스가 아니다.
    - 단 Aurora Serverless는 예외

- CloudWatch와 연동
    - DB인스턴스의 모니터링(EC2와 동일 ex: 디테일 모니터링, cpu, Storage사용량)
    - DB에서 발생하는 여러 로그(Error Log, General Log 등)을 확인 가능

- 내부에서는 EC2 활용

- VPC 안에서 동작
    - 기본적으로 Public IP를 부여하지 않아 외부에서 접근이 불가능
    - 설정에 따라 Public으로 오픈 가능
- 서브넷과 보안 그룹지정 필요(이를 통해서 방화벽 역할을 수행)
- EC2 타입의 지정 필요
- Storage는 EBS활용
    - EBS 타입의 선택 필요
    - 생성시 EBS의 용량을 지정해서 생성(필요한 용량을 미리 지정해야함, 추후에 용량 늘리기 가능, Aurora의 경우는 미리 지정하지 않고 사용한 양만큼 지불)

- Parameter Group: Root 유저만 설정 가능한 DB의 설정값들을 모아 그룹화한 개념
    - DB 클러스터에 파라미터 그룹을 적용시켜 설정값을 적용
    - 마이너 버전 엔진 업데이트는 자동으로 업데이트 설정 가능
    - 기타 업데이트(파라미터 그룸, EBS 사이즈, 인스턴스 변경)의 경우 점검시간을 설정하여 특정 시간에 업데이트가 이루어질 수 있도록 설정 가능

- RDS 인증방법
    - 전통적인 유저 / 패스원드 방식
        - AWS Secret Manager(RDS를 비롯해 DB, 여러 인증을 관리해주는 서비스, 자동으로 일정 주기마다 패스워드 변경, 다른 서비스들이 RDS 혹은 서비스들을 접근할 때 암호를 하드코딩할 필요 없게 만들어줌) 와 연동하여 자동 로테이션 가능

- IAM DB인증
  - 데이터베이스를 IAM 유저 크레덴셜 / Role을 통해 관리 가능
- Kerberos 인증(ms active directory 안에 들어있는 프로토콜)

RDS 가격모델

컴퓨팅 파워 + 스토리지 용량 + 백업 용량 + 네트워크 비용
Reserved Instance 구매 가능
- EC2 와 마찬가지로 일정 기간을 계약하여 저렴한 가격에 서비스를 이용 가능

RDS에서 제공하는 DB 엔진

라이선스 비용 발생(오픈 소스가 아님)

- MS SQL Server
- Oracle
  - Oracle OLAP

라이선스 비용이 필요 없음(오픈소스, 기반 별도의 사용량을 제외한 라이센스가 필요 없다.)

- My SQL Server
- PostgreSQL
- MariaDB

그 외

- Amazon Aurora

RDS 암호화

암호화 지원
- SQL 서버 혹은 Oracle에서는 TDE(Transparent Data Encryption) 지원
- 모든 엔진에서 EBS 볼륨 암호화 지원(볼륨 자체가 EBS 기반이기 때문)
  - Default Master Key 혹은 생성한 Master Key 선택 가능
- 자동 백업, 스냅샷, Read Replica 등에 적용

RDS 백업

자동 백업
- 매일마다 스냅샷을 만들고 트래젝션 로그를 저장(1일차 스냅샷 제작, 2일차 스냅샷 제작, 3일차 스냅샷 제작하는 과정을 거친다. 이 사이에 있는 모든 내용에 대한 트랜잭션 로그를 모두 저장한다.)
- 데이터는 S3에 저장되며 데이터베이스의 크기만큼 공간을 점유한다.
- 저장된 데이터를 바탕으로 일정 기간 내의 특정 시간으로 롤백 가능(롤백은 기존 DB를 롤백하는 것이 아닌 스냅샷을 기준으로 새로운 DB를 만든다. 트랜잭션 로그를 사용해서 만든다)
  - 다른 DB 클러스터를 새로 생성
- 1 ~ 35일까지 보관을 지원한다
- Backup을 시행할 때 약간의 딜레이 발생 가능성
- 기본적으로 사용 상태로 설정되어 있다.

수동 백업(DB 스냅샷)
- 유저 혹은 다른 프로세스로부터 요청에 따라 만들어지는 스냅샷
- 데이터베이스가 삭제된 이후에도 계속 보관
- 스냅샷의 복구는 항상 새로운 DB Instance를 생성하여 수행

RDS 구성 아키텍쳐

RDS Multi AZ
두개 이상의 AZ에 걸쳐 DB를 구축하고 원본과 다른 DB(standby)를 자동으로 동기화(Snyc)

- SQL Server, Oracle, MySQL, PostgreSQL, MariaDB에서 지원
- Aurora의 경우 다중 AZ를 설계 단계에서 지원

원본 DB의 장애 발생 시 자동으로 다른 DB가 원본으로 승격 된다.(DNS가 StandbyDB로)
StandbyDB는 접근이 불가능하다.
퍼포먼스의 상승 효과가 아닌 안정성을 위한 서비스이다.

읽기 전용 복제본(Read Replica)
원래 데이터베이스의 읽기 전용 복제본을 생성(Async)

- 쓰기는 원본 데이터베이스에, 읽기는 복제본에 처리하여 워크로드 분산
- MySQL, PostgreSQL, MariaDB, Oracle, Aurora에서 지원

안정성이 아닌 퍼포먼스를 위한 서비스이다
총 5대까지 생성이 가능하다
각각의 복제본은 고유 DNS가 할당된다 -> 접근이 가능

- 원본 DB의 장애 발생 시 수동으로 DNS 변경이필요하다

복제본 자체에 Multi-AZ 설정 가능(MySQL, MariaDB, PostgreSQL, Oracle)
Multi-AZ DB에 Read Replica 설정 가능
자동 백업이 활성화 되어 있어야 읽기 전용 복제본 생성 가능
각 DB의 엔진 버전이 다를 수 있다.

RDS Multi Region
다른 리전에 지속적으로 동기화 시키는 DB 클러스터를 생성

- Async 복제

주로 로컬 퍼포먼스 혹은 DR 시나리오(재난 시나리오)로 활용한다.
각 리전별로 자동 백업이 가능하다
리전별로 Multi - AZ 가 가능하다.

Multi-AZ

(참조:
https://support.bespinglobal.com/ko/support/solutions/articles/73000544793--aws-rds-%EC%9D%B4%EC%A4%91%ED%99%94-%EA%B5%AC%EC%84%B1-multi-az-)

Multi-az 배포에서 mysql RDS는 자동으로 서로 다른 AZ(가용 영역)에 동기식 예비 복제본을 프로비저닝하고 유지하여 데이터 이중화를 제공 (HA)
그 후, Master db 서버에 장애 발생 시 Standby 인스턴스를 master로 승격시키고, 새로운 Standby db 인스턴스를 생성하여 장애 조치를 수행
이 작업들은 관리자가 직접 개입할 필요 없이 RDS Multi-AZ 기능을 사용하시면 자동으로 진행됩니다.

Multi-AZ 작동 방식

1. 현재 Master db 인스턴스의 snapshot이 생성됩니다.

2. 생성된 snapshot을 이용하여 다른 AZ에 대기 인스턴스가 생성됩니다.

3. 기본 인스턴스와 Standby 인스턴스 간에 동기식으로 복제되어 데이터 중복성, snapshot 및 백업 중 I/O 중단 제거, 시스템 백업 중 지연 시간 급증을 최소화

실습

프라이빗 서브넷에 위치한 RDS 인스턴스에 Bastion Host를 통해 안전하게 접속하는 방법을 설명

1. VPC 및 서브넷 설정

1) VPC 생성: AWS Management Console에서 VPC를 생성
2) 서브넷 생성: 퍼블릭 서브넷과 프라이빗 서브넷을 각각 생성
3) 인터넷 게이트웨이 연결: 퍼블릭 서브넷에 인터넷 게이트웨이를 연결
4) 라우팅 테이블 설정: 퍼블릭 서브넷의 라우팅 테이블에 인터넷 게이트웨이를 추가

2. 보안 그룹 설정

Bastion Host 보안 그룹:
- 인바운드 규칙: SSH(포트 22)를 허용
- 아웃바운드 규칙: 모든 트래픽을 허용

RDS 보안 그룹:
- 인바운드 규칙: MySQL(포트 3306)을 Bastion Host의 보안 그룹에서만 허용
- 아웃바운드 규칙: 모든 트래픽을 허용

3. Bastion Host 생성

- EC2 인스턴스 생성: 퍼블릭 서브넷에 Bastion Host로 사용할 EC2 인스턴스를 생성
- 보안 그룹 연결: Bastion Host 보안 그룹을 연결
- 탄력적 IP 할당: Bastion Host에 탄력적 IP를 할당

4. RDS 인스턴스 생성

- RDS 인스턴스 생성: 프라이빗 서브넷에 MySQL RDS 인스턴스를 생성
- 보안 그룹 연결: RDS 보안 그룹을 연결

5. SSH 터널링 설정

SSH 터널링 명령어:

```
ssh -i <private_key.pem> -L 3306:<RDS 엔드포인트>:3306
ec2-user@<Bastion Host IP>
```

<private_key.pem>: Bastion Host에 접속할 때 사용하는 SSH 키 파일.
<RDS 엔드포인트>: RDS 인스턴스의 엔드포인트.
<Bastion Host IP>: Bastion Host의 탄력적 IP.

6. 로컬에서 RDS 접속

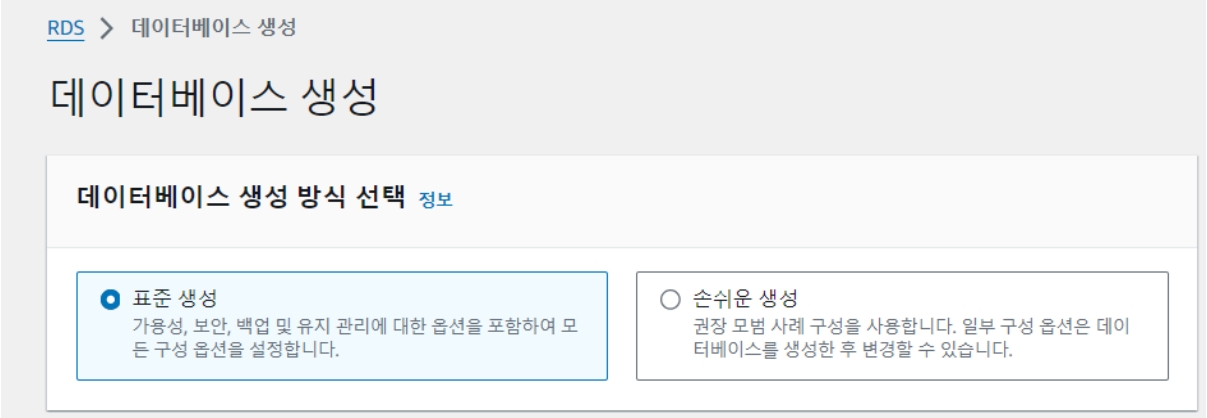
MySQL 클라이언트 사용:

```
mysql -h 127.0.0.1 -P 3306 -u <사용자 이름> -p
```

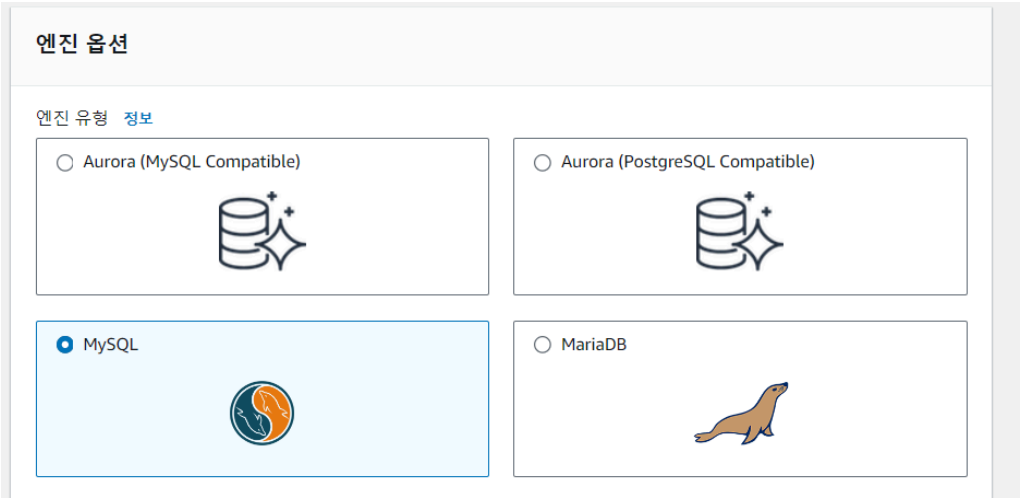
127.0.0.1: 로컬 호스트.
3306: 로컬 포트.
<사용자 이름>: RDS 인스턴스의 사용자 이름.

생성실습

데이터베이스 생성



엔진옵션 MySQL 선택



템플릿 옵션


템플릿
해당 사용 사례를 충족하는 샘플 템플릿을 선택하세요.
프로덕션
고가용성 및 빠르고 일관된 성능을 위해 기본값을 사용하세요.
개발/테스트
이 인스턴스는 프로덕션 환경 외부에서 개발 용도로 마련되었습니다.
프리 티어
RDS 프리 티어를 사용하여 새로운 애플리케이션을 개발하거나, 기존 애플리케이션을 테스트하거나 Amazon RDS에서 실무 경험을 쌓을 수 있습니다. 정보
가용성 및 내구성
배포 옵션 정보
아래의 배포 옵션은 위에서 선택한 엔진에서 지원하는 배포 옵션으로 제한됩니다.
다중 AZ DB 클러스터
기본 DB 인스턴스와 읽기 가능한 예비 DB 인스턴스 2개가 있는 DB 클러스터를 생성합니다. 각 DB 인스턴스는 서로 다른 가용 영역(AZ)에 있습니다. 고가용성, 데이터 이중화를 제공하고 읽기 워크로드를 처리하기 위한 용량을 늘립니다.
다중 AZ DB 인스턴스(다중 AZ DB 클러스터 스냅샷에는 지원되지 않음)
다른 AZ에 기본 DB 인스턴스와 예비 DB 인스턴스를 생성합니다. 고가용성 및 데이터 이중화를 제공하지만 예비 DB 인스턴스는 읽기 워크로드에 대한 연결을 지원하지 않습니다.
단일 DB 인스턴스(다중 AZ DB 클러스터 스냅샷에는 지원되지 않음)
예비 DB 인스턴스가 없는 단일 DB 인스턴스를 생성합니다.

템플릿
해당 사용 사례를 충족하는 샘플 템플릿을 선택하세요.
프로덕션
고가용성 및 빠르고 일관된 성능을 위해 기본값을 사용하세요.
개발/테스트
이 인스턴스는 프로덕션 환경 외부에서 개발 용도로 마련되었습니다.
프리 티어
RDS 프리 티어를 사용하여 새로운 애플리케이션을 개발하거나, 기존 애플리케이션을 테스트하거나 Amazon RDS에서 실무 경험을 쌓을 수 있습니다. 정보
가용성 및 내구성
배포 옵션 정보
아래의 배포 옵션은 위에서 선택한 엔진에서 지원하는 배포 옵션으로 제한됩니다.
다중 AZ DB 클러스터
기본 DB 인스턴스와 읽기 가능한 예비 DB 인스턴스 2개가 있는 DB 클러스터를 생성합니다. 각 DB 인스턴스는 서로 다른 가용 영역(AZ)에 있습니다. 고가용성, 데이터 이중화를 제공하고 읽기 워크로드를 처리하기 위한 용량을 늘립니다.
다중 AZ DB 인스턴스
다른 AZ에 기본 DB 인스턴스와 예비 DB 인스턴스를 생성합니다. 고가용성 및 데이터 이중화를 제공하지만 예비 DB 인스턴스는 읽기 워크로드에 대한 연결을 지원하지 않습니다.
단일 DB 인스턴스
예비 DB 인스턴스가 없는 단일 DB 인스턴스를 생성합니다.

설정

나머지 건들지 않고 생성 버튼 클릭
몇 분 뒤 생성

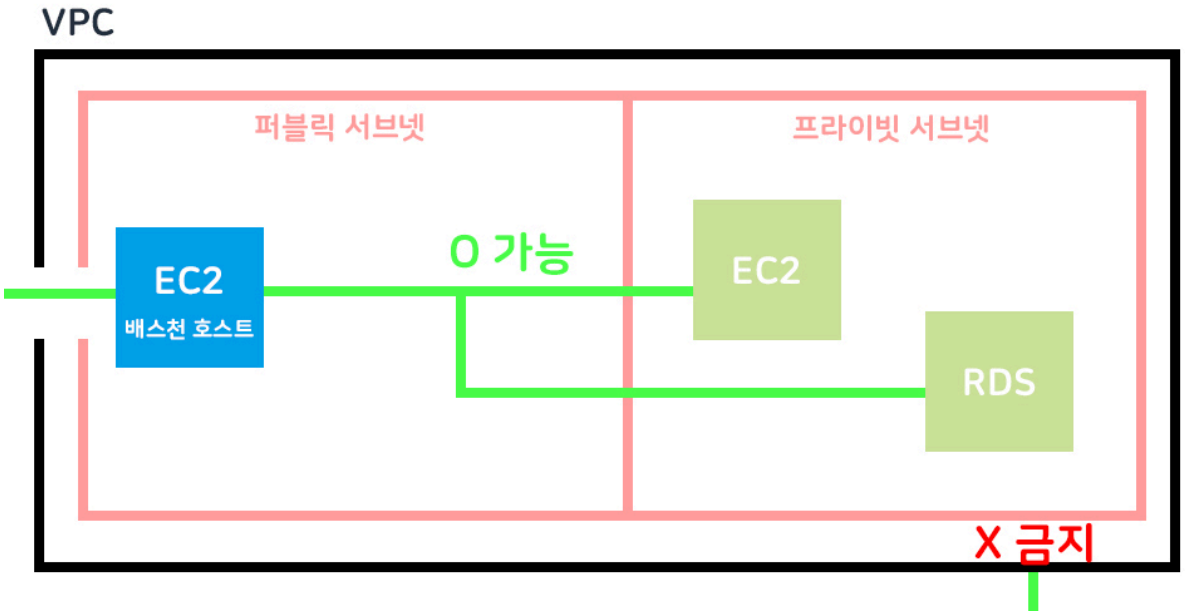
Bastion 호스트로 프라이빗 **RDS** 접속하기

Bastion 호스트?

구축한 인프라 또는 네트워크의 입구에서 허용된 사람만 출입할 수 있는 역할을 수행
이 역할을 수행하는 서버를 Bastion 호스트 또는 Bastion 서버 라고 한다.

특정 서버에서 DB에 접근하는 것을 허용한 후, 그 서버를 통해서 DB에 접속하는 방법을 사용하게 되는데 이때 통로 역할을 해주는 서버가 Bastion 호스트

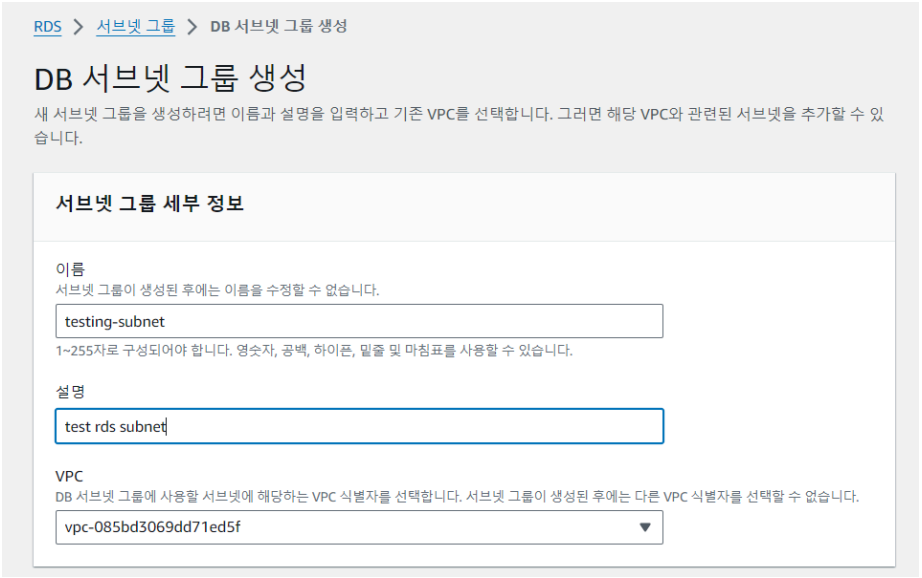
구조



실습

DB 서브넷 그룹 생성

생성



DB 서브넷 그룹 생성: DB 인스턴스를 만들기 위해서는 DB 서브넷 그룹이 있어야 한다. 다음의 경로로 먼저 이동한다. RDS > 서브넷 그룹 > DB 서브넷 그룹
그리고, 아래와 같이 값을 설정해주면 된다.
-> 서브넷: RDS를 위치시킬 곳을 정하면 된다. 단, private 서브넷으로 설정해주자.
-> 가용 영역: 위의 서브넷이 속한 가용영역을 선택해주면 된다. 헷갈리면 모두 선택해줘도 무방하다.

보안 그룹 생성

- EC2 > 보안그룹 으로 이동
- '보안 그룹 생성' 버튼 클릭
- 그리고, 이름을 써주고 생성한 VPC를 선택해준다. 그리고 인바운드 규칙은 아래와 같이 설정해주고, 아웃바운드 규칙은 그대로 두자.

RDS 생성

```
엔진 유형: MySQL
템플릿: 프로덕션
가용성 및 내구성: 단일 DB 인스턴스
인스턴스 구성: 버스터블 클래스 (db.m6gd.large)(변경가능)
스토리지: 범용 SSD(gp2)
할당된 스토리지: 100GB
VPC, 서브넷 그룹, 보안 그룹: 미리 생성한 것들로 설정
가용 영역: ap-northeast-2a (이건 프라이빗 서브넷을 어디에 생성했는지에
따라서 사람마다 다름)
모니터링: off
(나머지는 기본값)
```

참고

- https://velog.io/@dlwldbs/AWS-%EC%8B%A4%EC%8A%B5-RDS-%EC%83%9D%EC%84%B1

- https://velog.io/@juhyeon1114/aws-bastion-ec2-rds

- https://velog.io/@juhyeon1114/AWS-VPC-%EC%95%8C%EC%95%84%EB%B3%B4%EA%B8%B0