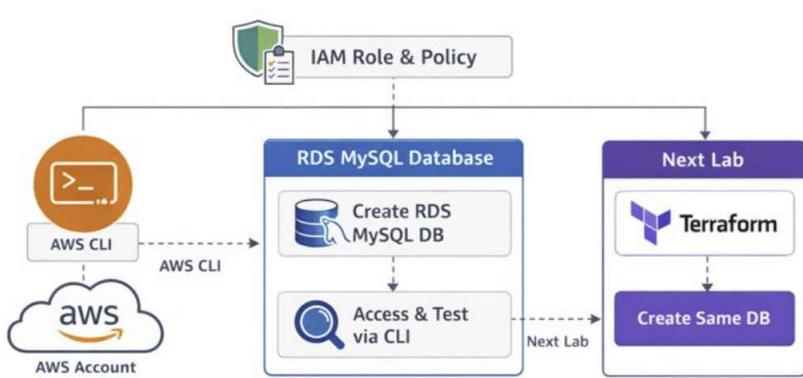


## Terraform Lab 5 — MySQL RDS via CLI

### (DVA-C02 Foundation)



## Objectifs

- Crée **via CLI** une base de données RDS – Engine : MySQL
- Vérifier que la base fonctionne correctement
- Crée **via Terraform** la même base (prochain lab)

---

## Prérequis

- **OS** : Windows
  - **Compte AWS actif**
  - **AWS CLI installé**
  - **Terraform** installé et dans le PATH
- 

## Contents

1 Présentation :	2
2 Prérequis techniques.....	2
3 création via CLI .....	2
3.1 création de la bd .....	2
3.2 Vérifier l'état de la base .....	2
3.3 Récupérer l'endpoint de la base.....	3

3.4 Vérifier l'état de la base.....	3
3.5 Vérifier le groupe de sécurité associé .....	4
3.6 Vérifier le subnet group .....	4
3.7 rendre la base publique pour les tests.....	5
3.8 vérifier si RDS est accessible depuis cloudshell .....	5
3.8.1 Test avec nc .....	5
3.8.2 Vérification des règles dans le groupe de RDS:.....	6
3.9 Se connecter à RDS via AWS cloudShell.....	8

## 1 Présentation :

Objectif : Créer une **instance Amazon RDS MySQL** à l'aide du **AWS CLI**, sans passer par la console AWS, puis faire le même exercice lors du prochain lab avec Terraform.

- ✓ Crédit d'une base RDS d'engine MySQL

## 2 Prérequis techniques

- OS : Windows
- Un compte AWS actif
- AWS CLI installé et configuré

## 3 création via CLI

### 3.1 création de la bd

```
aws rds create-db-instance ^
--db-instance-identifier test-mysql-db ^
--db-instance-class db.t3.micro ^
--engine mysql ^
--master-username admin ^
--master-user-password Test1234! ^
--allocated-storage 20 ^
--no-publicly-accessible
```

### 3.2 Vérifier l'état de la base :

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db ^
--query "DBInstances[0].DBInstanceStatus"
```

La base est en cours de backup . ( backup automatique dans RDS)

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].DBInstanceState"
"backing-up"
```

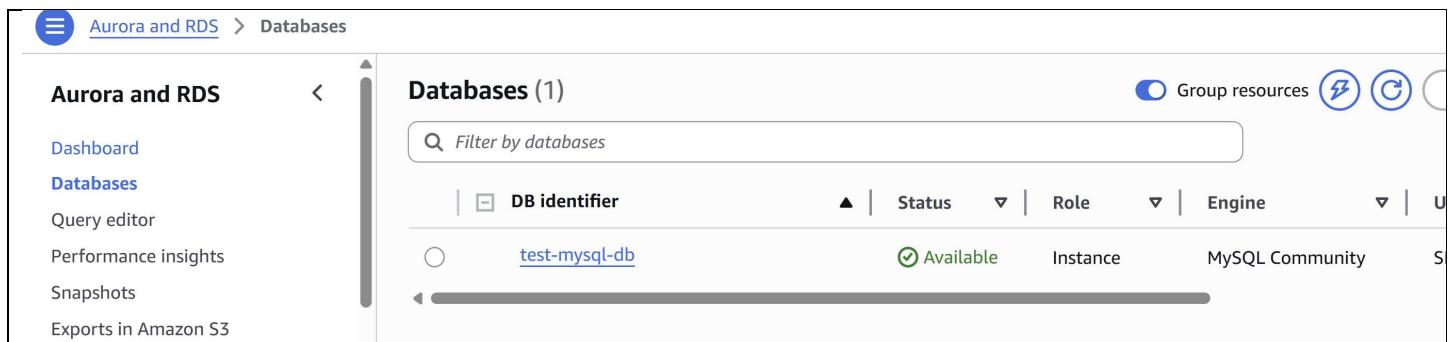
```
C:\Users\henri>
```

La base est maintenant disponible

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].DBInstanceState"
"available"
```

```
C:\Users\henri>
```

On vérifie aussi au niveau de la console AWS :



The screenshot shows the AWS Aurora and RDS console. On the left, there's a sidebar with options like Dashboard, Databases, Query editor, Performance insights, Snapshots, and Exports in Amazon S3. The main area is titled 'Databases (1)' and shows a table with one row. The row contains the DB identifier 'test-mysql-db', a status indicator with a green checkmark and the word 'Available', the label 'Instance', and 'MySQL Community' under the Engine column. There are also columns for Role and Engine.

### 3.3 Récupérer l'endpoint de la base

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db ^
--query "DBInstances[0].Endpoint.Address"
```

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].Endpoint.Address"
"test-mysql-db.cl6r3jzwpko6.ca-central-1.rds.amazonaws.com"
```

```
C:\Users\henri>
```

L'endpoint est l'adresse de connexion MySQL.

### 3.4 Vérifier l'état de la base

La commande ci-dessous donne toutes les informations sur la base :

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db
```

La commande ci-dessous indique seulement l'état public ou privé de l'instance

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db ^
--query "DBInstances[0].PubliclyAccessible"
```

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].PubliclyAccessible"
false
C:\Users\henri>
```

False, : la valeur booléenne indique que la base n'est pas publique

### 3.5 Vérifier le groupe de sécurité associé

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db ^
--query "DBInstances[0].VpcSecurityGroups[*].VpcSecurityGroupId"
```

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].VpcSecurityGroups[*].VpcSecurityGroupId"
[
    "sg-0c38036d664ab78a6"
]
C:\Users\henri>
```

### 3.6 Vérifier le subnet group

```
aws rds describe-db-instances ^
--db-instance-identifier test-mysql-db ^
--query "DBInstances[0].DBSubnetGroup.DBSubnetGroupName"
```

```
C:\Users\henri>aws rds describe-db-instances ^
More? --db-instance-identifier test-mysql-db ^
More? --query "DBInstances[0].DBSubnetGroup.DBSubnetGroupName"
"default"
C:\Users\henri>
```

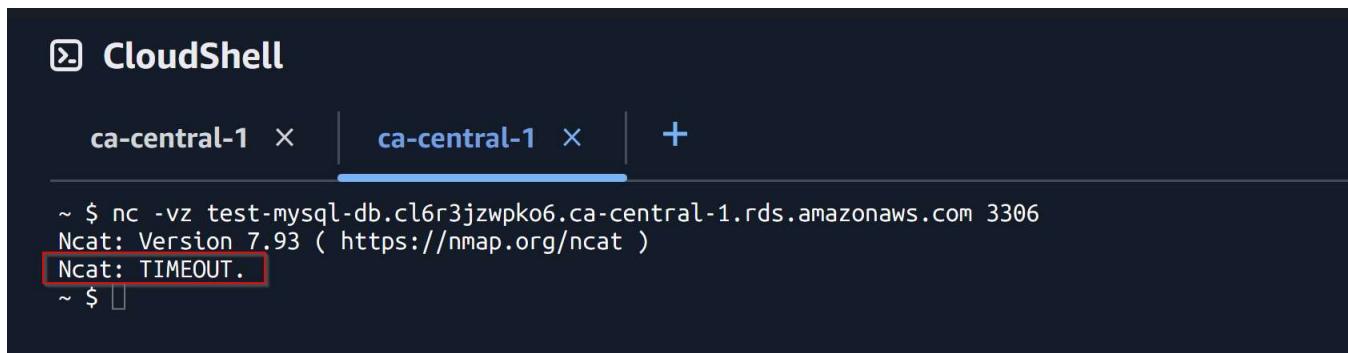
### 3.7 rendre la base publique pour les tests

```
aws rds modify-db-instance ^
--db-instance-identifier test-mysql-db ^
--publicly-accessible ^
--apply-immediately
```

### 3.8 vérifier si RDS est accessible depuis cloudshell

#### 3.8.1 Test avec nc

```
nc -vz test-mysql-db.cl6r3jzwpo6.ca-central-1.rds.amazonaws.com 3306
```



```
CloudShell
ca-central-1 X | ca-central-1 X | +
~ $ nc -vz test-mysql-db.cl6r3jzwpo6.ca-central-1.rds.amazonaws.com 3306
Ncat: Version 7.93 ( https://nmap.org/ncat )
Ncat: TIMEOUT.
~ $
```

Ne marche pas car l'ip du cloudshell pas autorisé au niveau du security group de RDS

On récupère alors l'ip publique de cloudshell

```
$ curl ifconfig.me
```

```
~ $ curl ifconfig.me
3.99.172.204~ $
```

On rajoute cette adresse ip au security group de RDS :

```
aws ec2 authorize-security-group-ingress \
--group-id <SG_ID> \
--protocol tcp \
--port 3306 \
--cidr <CLOUDSHELL_IP>/32
```

Donc, en mettant les bons paramètres :

```
aws ec2 authorize-security-group-ingress \
--group-id sg-0c38036d664ab78a6 \
--protocol tcp \
--port 3306 \
--cidr 3.99.172.204/32
```

Avant d'exécuter la commande précédente , on vérifie les règles de sécurité du groupe pour RDS

On a :

Inbound rules (2)						
<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-05dddc972b705f355	-	All traffic	All	All
<input type="checkbox"/>	-	sgr-0721f529a21a8b159	IPv4	MySQL/Aurora	TCP	3306

Après exécution de la commande, on constate : une règle a été rajoutée

Inbound rules (3)						
<input type="checkbox"/>	Name	Security group rule ID	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-075affd5d5c6f85d0	IPv4	MySQL/Aurora	TCP	3306
<input type="checkbox"/>	-	sgr-05dddc972b705f355	-	All traffic	All	All
<input type="checkbox"/>	-	sgr-0721f529a21a8b159	IPv4	MySQL/Aurora	TCP	3306

### 3.8.2 Vérification des règles dans le groupe de RDS:

Saisir au niveau de cloudshell la commande :

```
aws ec2 describe-security-groups \
--group-ids sg-0c38036d664ab78a6 \
--query 'SecurityGroups[0].IpPermissions' \
--output table
```

ci-dessous le contenu n'est pas montré dans son entièreté, pour des raisons de confidentialité.

## CloudShell

ca-central-1 +

```
~ $ aws ec2 describe-security-groups \
>   --group-ids sg-0c38036d664ab78a6 \
>   --query 'SecurityGroups[0].IpPermissions' \
>   --output table
-----
|      DescribeSecurityGroups      |
+-----+-----+-----+
| FromPort | IpProtocol | ToPort  |
+-----+-----+-----+
|       | -1        |          |
+-----+-----+-----+
|      UserIdGroupPairs           |
+-----+-----+
| GroupId     | UserId    |
+-----+-----+
```

Mais on voit bien dans la suite du tableau, la présence de l'Adresse ip de notre instance de type console cloudshell : 3.99.172.204/32

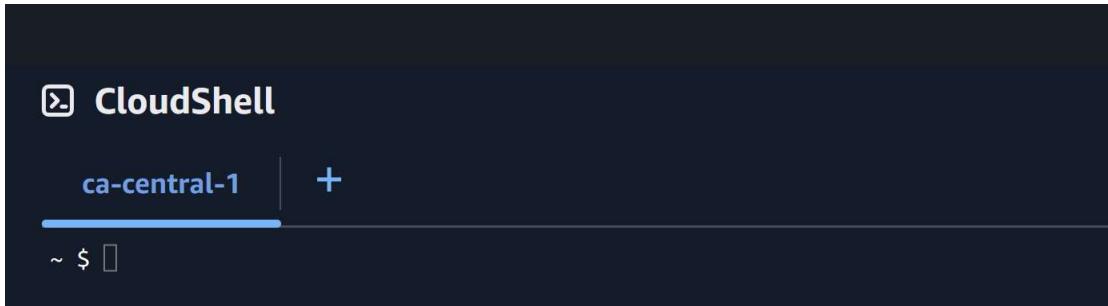
## CloudShell

ca-central-1 +

```
.... skipping...
-----
|      DescribeSecurityGroups      |
+-----+-----+-----+
| FromPort | IpProtocol | ToPort  |
+-----+-----+-----+
|       | -1        |          |
+-----+-----+-----+
|      UserIdGroupPairs           |
+-----+-----+
| GroupId     | UserId    |
+-----+-----+
| [REDACTED] | [REDACTED] |
+-----+-----+
|      DescribeSecurityGroups      |
+-----+-----+-----+
| FromPort | IpProtocol | ToPort  |
+-----+-----+-----+
| 3306     | tcp       | 3306    |
+-----+-----+-----+
|      IpRanges                  |
+-----+-----+
|      CidrIp                   |
+-----+-----+
| [REDACTED] |
| 3.99.172.204/32               |
+-----+-----+
```

### 3.9 Se connecter à RDS via AWS cloudShell

Aller dans AWS , lancer la console cloudbshell

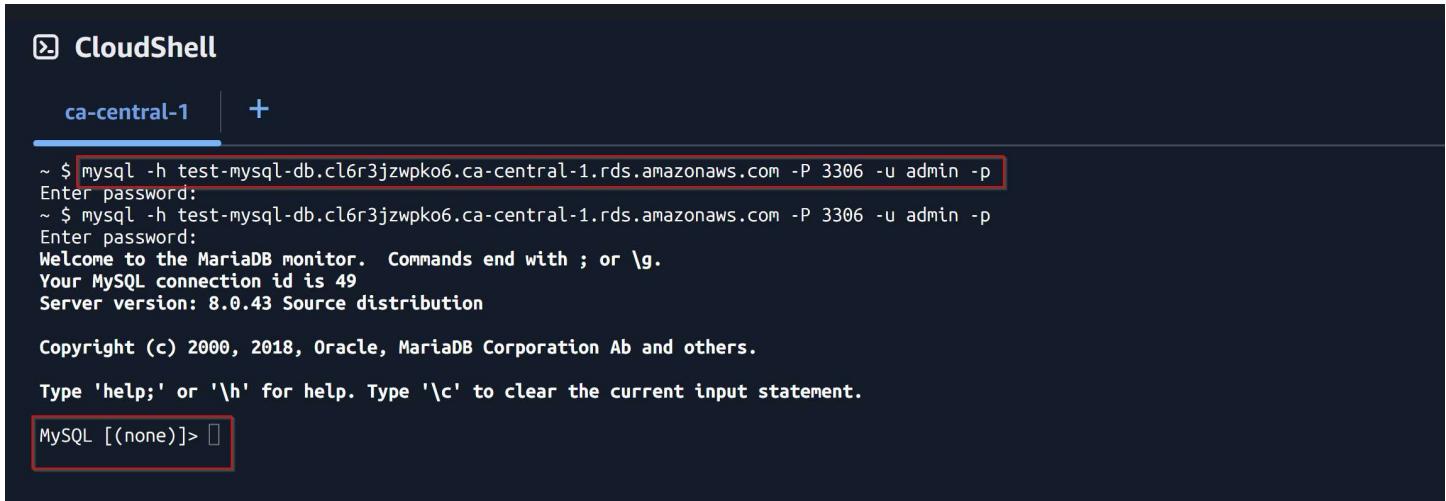


The screenshot shows the AWS CloudShell interface. At the top, it says "CloudShell". Below that is a list of sessions: "ca-central-1" (which is highlighted with a blue bar) and a "+" sign indicating the ability to open a new session. The main area is a terminal window with the prompt "~ \$".

Taper à nouveau la commande :

```
mysql -h test-mysql-db.cl6r3jzwpko6.ca-central-1.rds.amazonaws.com -P 3306 -u admin -p
```

cela fonctionne, maintenant, on obtient le prompt de MySQL



The screenshot shows the AWS CloudShell interface with a MySQL session running. The session is connected to the database "test-mysql-db" located at "cl6r3jzwpko6.ca-central-1.rds.amazonaws.com" with port 3306, user "admin", and password redacted. The MySQL monitor displays the following information:

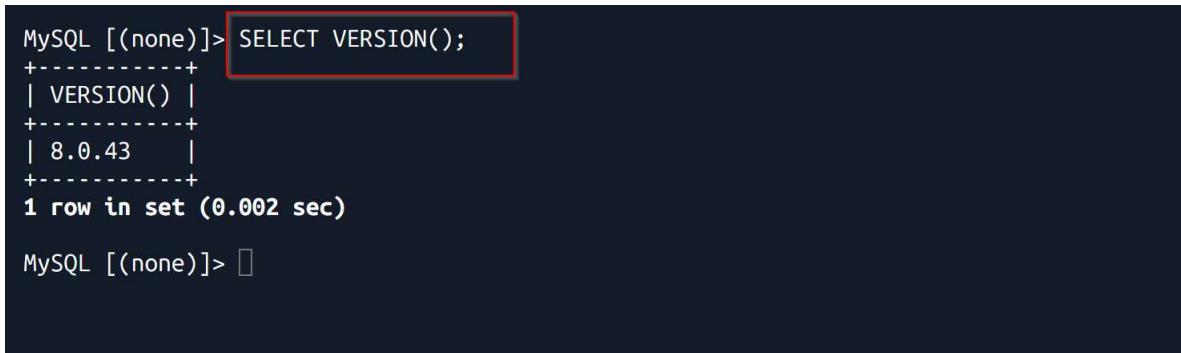
```
~ $ mysql -h test-mysql-db.cl6r3jzwpko6.ca-central-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
~ $ mysql -h test-mysql-db.cl6r3jzwpko6.ca-central-1.rds.amazonaws.com -P 3306 -u admin -p
Enter password:
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MySQL connection id is 49
Server version: 8.0.43 Source distribution

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MySQL [(none)]>
```

Vérifier la versions :



The screenshot shows the AWS CloudShell interface with a MySQL session running. The user has run the command "SELECT VERSION();". The output shows the MySQL version is 8.0.43.

```
MySQL [(none)]> SELECT VERSION();
+-----+
| VERSION() |
+-----+
| 8.0.43   |
+-----+
1 row in set (0.002 sec)

MySQL [(none)]>
```

Vérifier les bases de données :

```
MySQL [(none)]> SHOW DATABASES;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
+-----+
4 rows in set (0.011 sec)
```