

Using Mappings with FindMap Function

The process involves integrating a Mappings section in the CloudFormation template for environment-specific configurations. Add a mapping called EnvironmentOptions with top-level keys (e.g., Production and Test) and sub-keys (e.g., DbClass with respective values). Introduce an EnvironmentName parameter of type String with valid values (Production, Test) and a default value (Test). Replace the DbClass parameter with a dynamic mapping using Fn::FindInMap. Save and upload the updated template to the AWS CloudFormation Console. Create stacks for Test and Production environments, verify the corresponding instance classes (db.t3.micro and db.t2.small), and delete stacks to clean up resources.

Activity

1. Find the template files in our GitHub repository under the same name as the heading for easy access and edits. Find and Save the attached template locally, open it in VS Code for edits.
2. Defined a Mappings section in the template between Parameters and Resources sections as a best practice. Created a mapping named EnvironmentOptions for environment-specific configurations. Added top-level keys for environments (Production and Test) under the mapping. Defined a second-level key (DbClass) with values (db.t2.small for Production and db.t2.micro for Test). Created a new parameter EnvironmentName of type String to capture the environment name during stack creation.

```
62     Description: A valid VPC id in your AWS account
63     EnvironmentName:
64       Type: String
65       AllowedValues:
66         - Production
67         - Test
68       Default: Test
69
70 # The db.t2 class was deprecated! Please use these values for a smooth learning experience.
71 Mappings:
72   EnvironmentOptions:
73     Production:
74       DbClass: db.t4g.small
75     Test:
76       DbClass: db.t3.micro
77
78 Resources:
```

- Limited the valid values for EnvironmentName to Production and Test and set a default value (Test). Removed the DbClass parameter from the template and metadata since it is now mapped dynamically.
- Updated the DBInstanceClass property of the DatabaseInstance resource to use the Fn::FindInMap function with inputs EnvironmentOptions, EnvironmentName, and DbClass. Demonstrated both the long and short formats of Fn::FindInMap in the template.

```

125 DatabaseInstance:
126   Type: AWS::RDS::DBInstance
127   DeletionPolicy: Delete
128   Properties:
129     DBInstanceClass: !FindInMap [ EnvironmentOptions, !Ref EnvironmentName, DbClass ]
130     Engine: mysql
131     MultiAZ: !Ref MultiAZ
132     PubliclyAccessible: true
133     AllocatedStorage: !Ref AllocatedStorage

```

- Change the Image Id, Subnet id and Vpc id according to your region. Saved and uploaded the updated template to AWS CloudFormation Console.
- Created a stack for the test environment, verified the parameter grouping.

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Database Instance Settings

MultiAZ

Enable Multi-AZ?

false

Allocated Storage Size

Database storage size in GB

8

MasterUsername

Master username for the db instance

MasterUserPassword

Master user password for the db instance

Network Settings

VpcId

A valid VPC id in your AWS account

vpc-0b0c8badfef0024a4

Network Settings

VpcId

A valid VPC id in your AWS account

vpc-0b0c8badfef0024a4

DbSubnets

Db subnet ids as a list: <subnet1>,<subnet2>,...

Select List<AWS::EC2::Subnet::Id>

subnet-0619479663b4084af

subnet-08c772de46d053876

subnet-0a1bbb2feaa84bb95

SecurityGroupPorts

Port numbers as a list: <web-server-port>,<database-port>

80,3306

EnvironmentName

Test

Cancel

Previous

Next

7. Observe the instance class from the mapping (db.t3.micro).

The screenshot shows the Amazon RDS console for the instance `eventstack-databaseinstance-bp69dmzag9sn`. The instance is in the `Creating` state and has the `db.t3.micro` class. The console displays various tabs including **Connectivity & security**, **Monitoring**, **Logs & events**, **Configuration**, **Zero-ETL integrations**, **Maintenance & backups**, and **Data**. The **Connectivity & security** tab is active, showing details for the endpoint, port, VPC, and security groups.

Summary					
DB identifier	eventstack-databaseinstance-bp69dmzag9sn	Status	Creating	Role	Instance
Engine	MySQL Community	Class	db.t3.micro	Current activity	
Region & AZ	eu-west-1a				

8. Created another stack for the production environment, selected Production for EnvironmentName, and verified the instance class from resource pane of stack(DataBaseInstance) and checked the class(db.t2.small).

Parameters

Parameters are defined in your template and allow you to input custom values when you create or update a stack.

Database Instance Settings

MultiAZ
Enable Multi-AZ?

Allocated Storage Size
Database storage size in GB

MasterUsername
Master username for the db instance

MasterUserPassword
Master user password for the db instance

Network Settings

VpcId
A valid VPC id in your AWS account

Network Settings

VpcId
A valid VPC id in your AWS account

DbSubnets
Db subnet ids as a list: <subnet1>,<subnet2>,...

SecurityGroupPorts
Port numbers as a list: <web-server-port>,<database-port>

EnvironmentName

The screenshot shows the Amazon RDS console for the instance `productionstack-databaseinstance-zudvtwquuhd4`. The instance is in the `Creating` state and has the `db.t4g.small` class. The console displays various tabs including **Connectivity & security**, **Monitoring**, **Logs & events**, **Configuration**, **Zero-ETL integrations**, **Maintenance & backups**, and **Data**. The **Connectivity & security** tab is active, showing details for the endpoint, port, VPC, and security groups.

Summary					
DB identifier	productionstack-databaseinstance-zudvtwquuhd4	Status	Creating	Role	Instance
Engine	MySQL Community	Class	db.t4g.small	Current activity	
Region & AZ	eu-west-1b				

9. Deleted both test and production stacks to clean up resources.

