Contents

Lab 3: Relational Database Service (RDS)

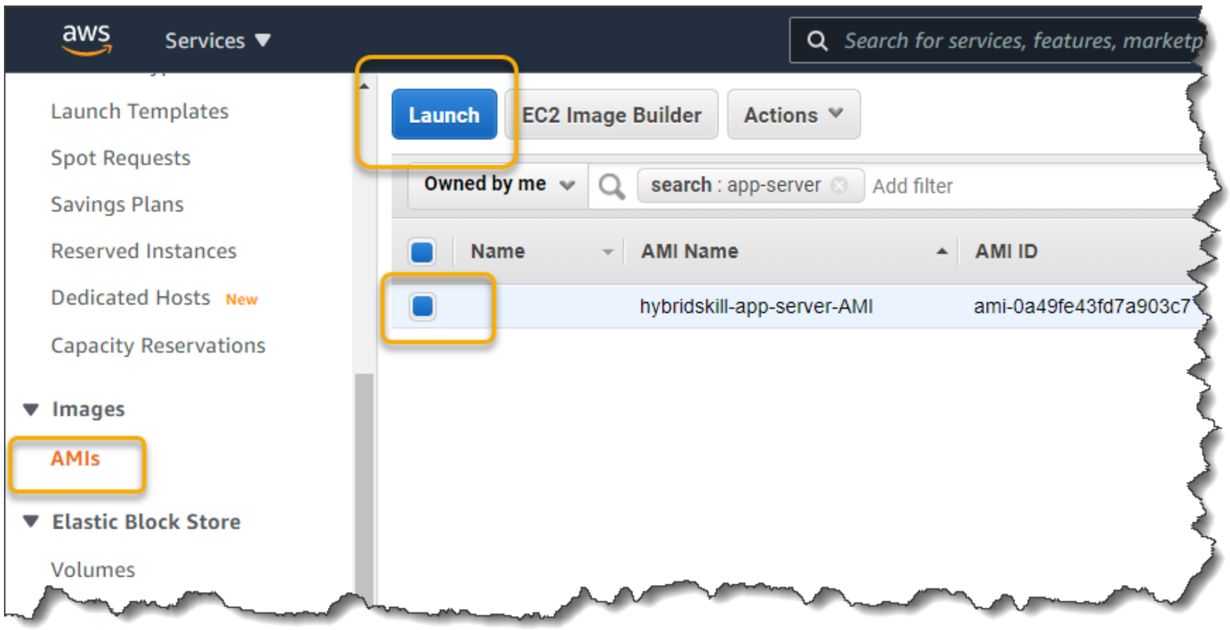
In this lab we are going to create an RDS instance. We will then migrate an applications local database to RDS and finally switch the application to use RDS.

Task Breakdown

- Create a WordPress instance using your AMI
- Create a Security Group for RDS
- Create a Subnet Group for RDS
- Launch a RDS instance
- Take back up of local database
- Migrate to RDS
- Point application to RDS
- Manage RDS through CLI

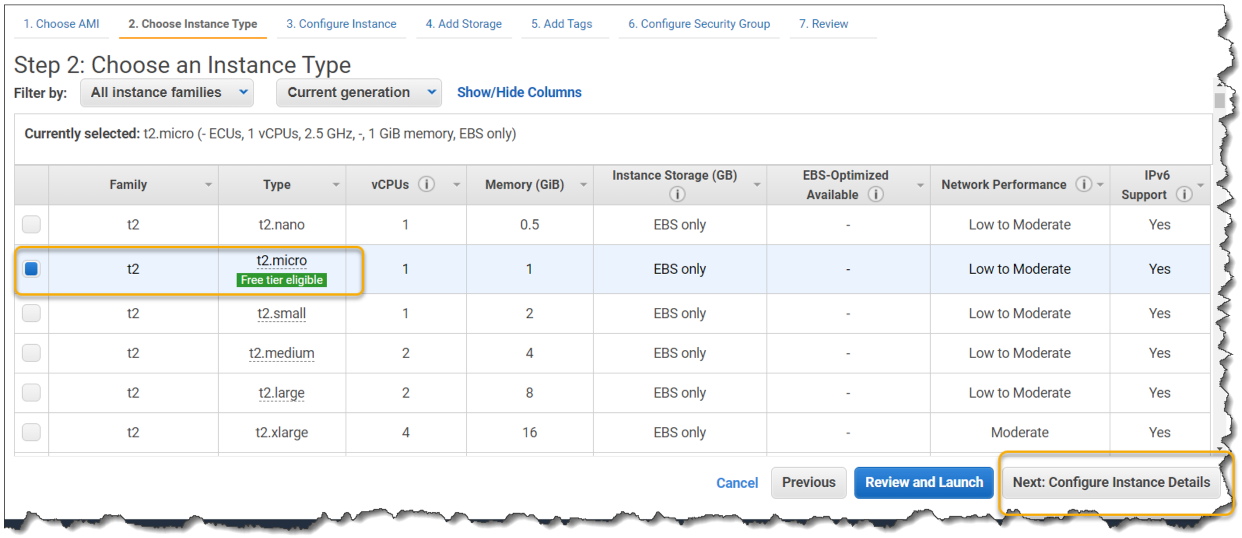
Task 1: Create a Wordpress instance using your AMI

1. On the main **EC2 dashboard** under **Images**, Click **AMIs** and find the **Wordpress image** you had created in the previous lab. **Select** and Click **Launch**.



Name	AMI Name	AMI ID
	hybridskill-app-server-AMI	ami-0a49fe43fd7a903c7

2. As before select **t2.micro** as the **Instance Type** and Click **Next: Configure Instance Details**



1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Add Tags | 6. Configure Security Group | 7. Review

Step 2: Choose an Instance Type

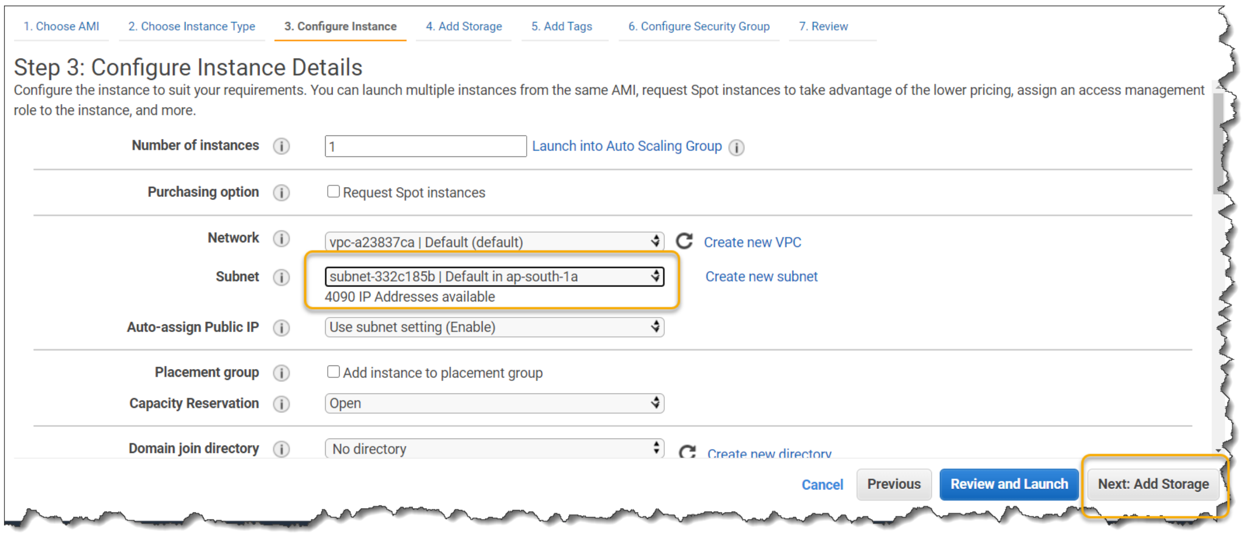
Filter by: All instance families | Current generation | Show/Hide Columns

Currently selected: t2.micro (- ECUs, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
✓	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
	t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
	t2	t2.xlarge	4	16	EBS only	-	Moderate	Yes

Cancel | Previous | Review and Launch | Next: Configure Instance Details

3. For the **Subnet**, select the same AZ (**subnet-xxxx ap-south-1a** in my case). Leave all options as default and click **Next: Add storage.**



1. Choose AMI | 2. Choose Instance Type | 3. Configure Instance | 4. Add Storage | 5. Add Tags | 6. Configure Security Group | 7. Review

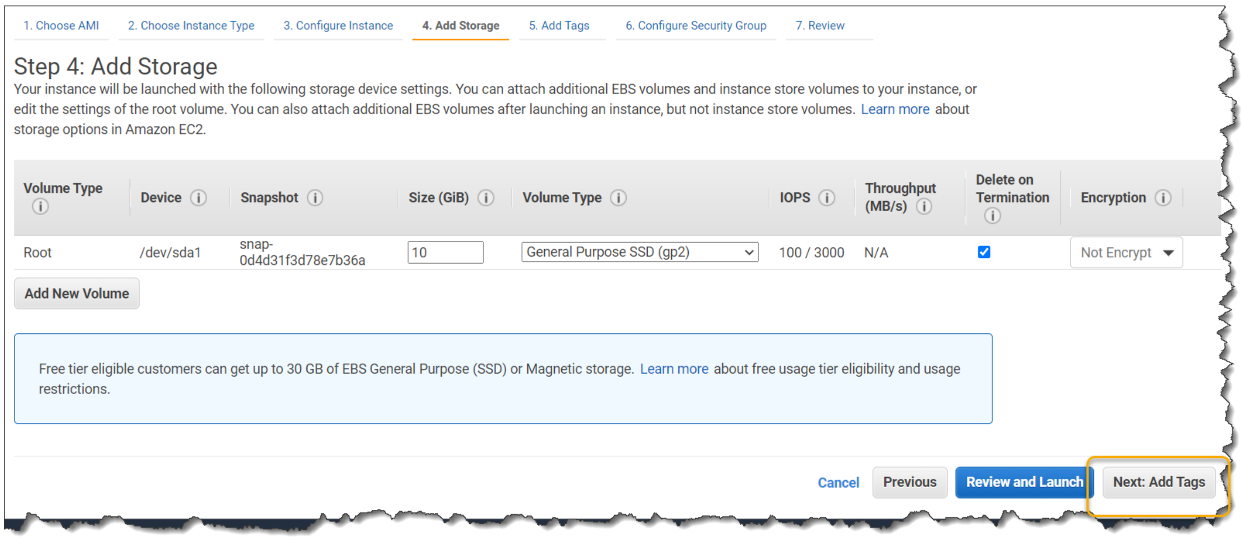
Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot instances to take advantage of the lower pricing, assign an access management role to the instance, and more.

Number of instances	1 — Launch into Auto Scaling Group
Purchasing option	☐ Request Spot instances
Network	vpc-a23837ca | Default (default) — Create new VPC
Subnet	subnet-332c185b | Default in ap-south-1a — 4090 IP Addresses available — Create new subnet
Auto-assign Public IP	Use subnet setting (Enable)
Placement group	☐ Add instance to placement group
Capacity Reservation	Open
Domain join directory	No directory — Create new directory

Cancel | Previous | Review and Launch | Next: Add Storage

4. Leave all options as default and click **Next: Add Tags.**



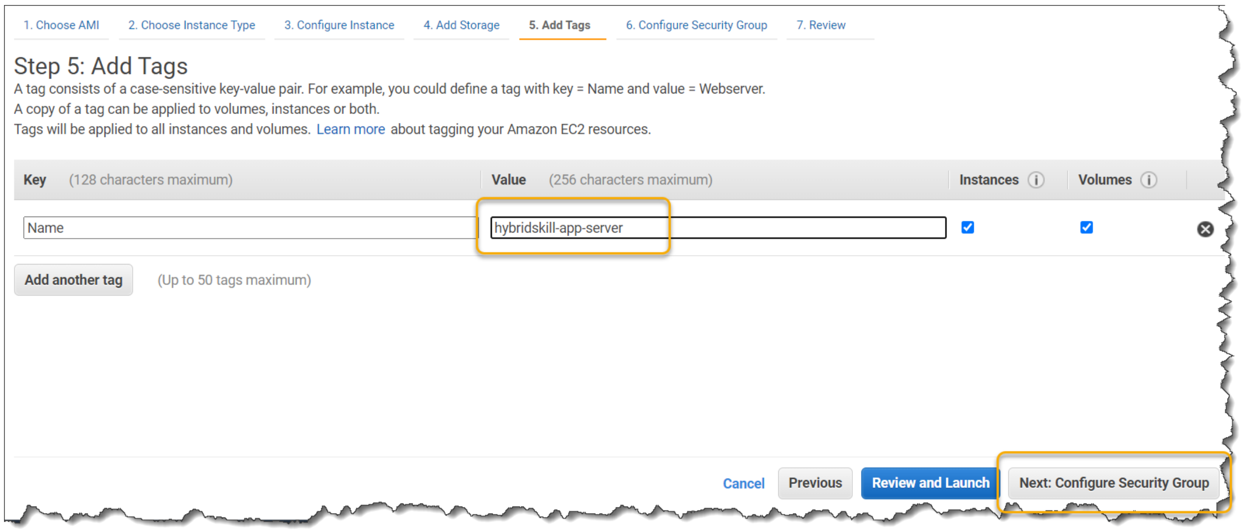
Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance, or edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more about storage options in Amazon EC2.

Volume Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Throughput (MB/s)	Delete on Termination	Encryption
Root	/dev/sda1	snap-0d4d31f3d78e7b36a	10	General Purpose SSD (gp2)	100 / 3000	N/A	✓	Not Encrypt

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

5. Give **hybridskill-app-server-01** as the name of the instance and click **Next: Configure Security Group**

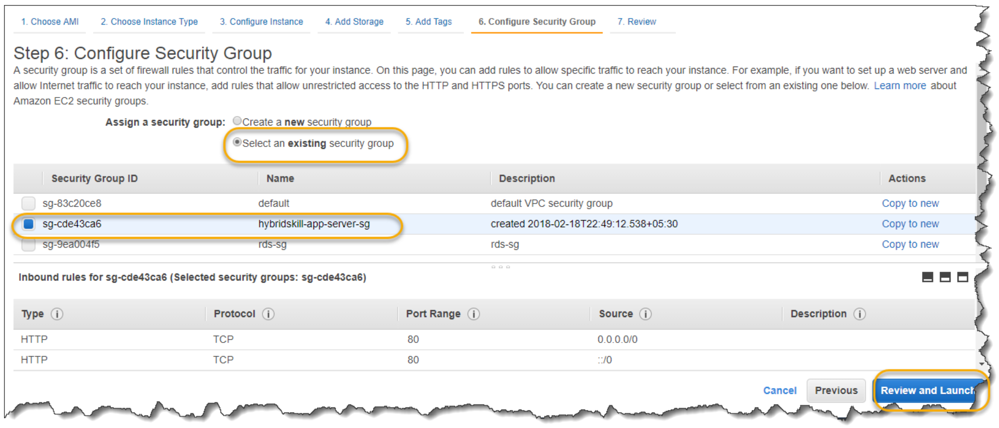


Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.
A copy of a tag can be applied to volumes, instances or both.
Tags will be applied to all instances and volumes. Learn more about tagging your Amazon EC2 resources.

Key (128 characters maximum)	Value (256 characters maximum)	Instances	Volumes
Name	hybridskill-app-server	✓	✓

6. Click **Select an existing security group,** select your **hybridskill-app-server-**sg created in the previous lab as the name of your security group and click **Review and Launch.**



Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want to set up a web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from an existing one below. Learn more about Amazon EC2 security groups.

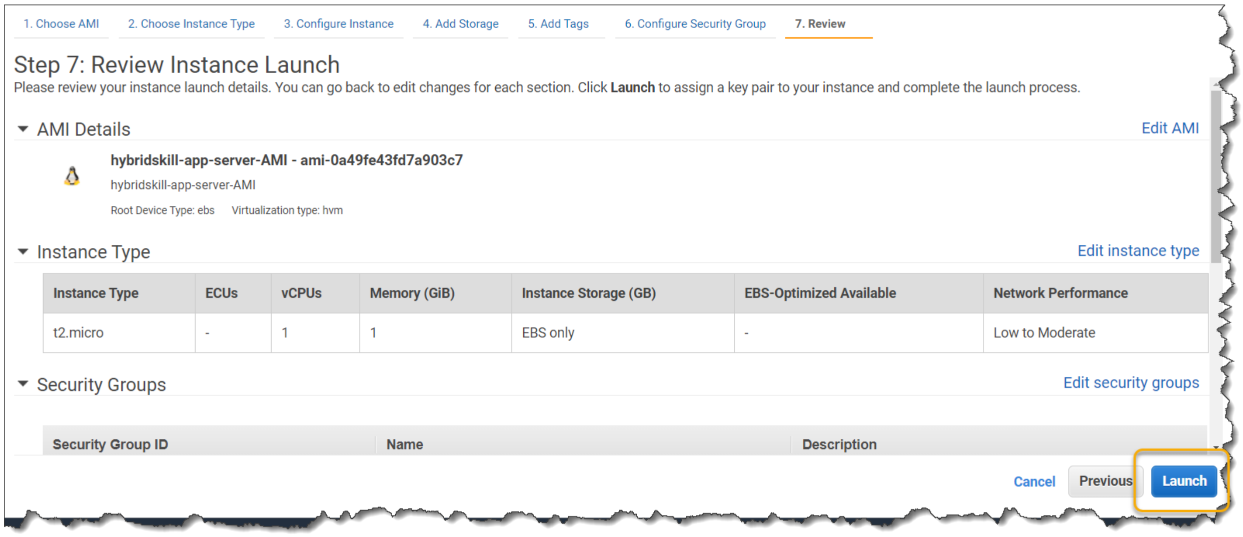
Assign a security group: Create a **new** security group [✓ Select an **existing** security group]

	Security Group ID	Name	Description	Actions
	sg-83c20ce8	default	default VPC security group	Copy to new
✓	sg-cde43ca6	hybridskill-app-server-sg	created 2018-02-18T22:49:12.538+05:30	Copy to new
	sg-9ea004f5	rds-sg	rds-sg	Copy to new

Inbound rules for sg-cde43ca6 (Selected security groups: sg-cde43ca6)

Type	Protocol	Port Range	Source	Description
HTTP	TCP	80	0.0.0.0/0	
HTTP	TCP	80	::/0	

7. Review your settings and click **Launch.**



Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

AMI Details — Edit AMI

hybridskill-app-server-AMI - ami-0a49fe43fd7a903c7
hybridskill-app-server-AMI
Root Device Type: ebs Virtualization type: hvm

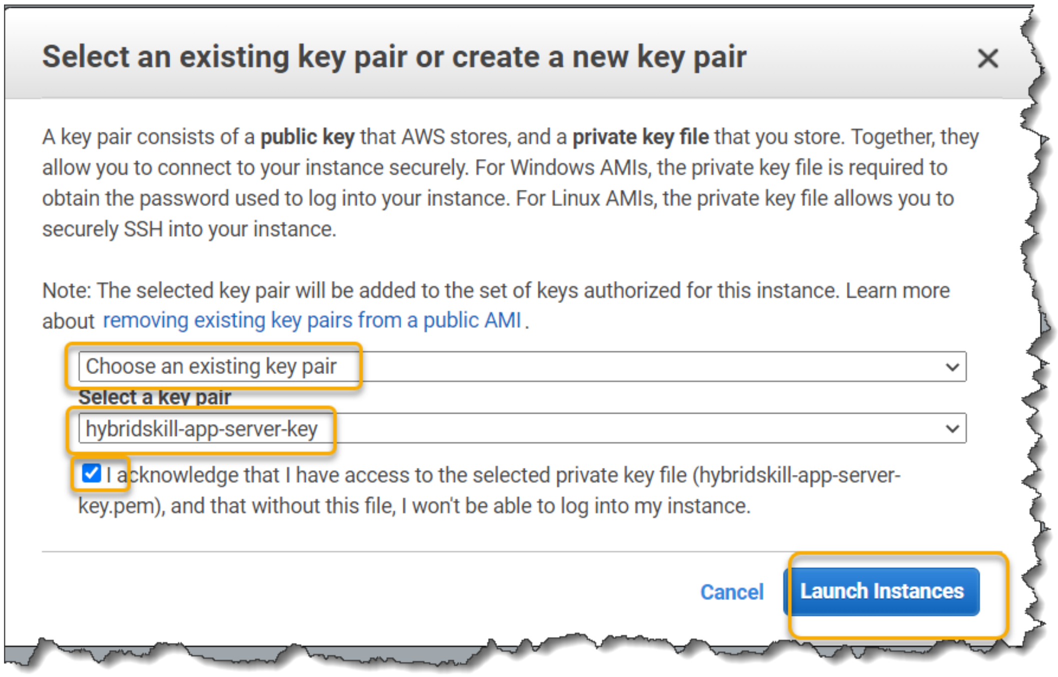
Instance Type — Edit instance type

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	-	1	1	EBS only	-	Low to Moderate

Security Groups — Edit security groups

Security Group ID	Name	Description

8. Select **Choose an existing key-pair** select the key you created earlier and Click **Launch Instances**


Select an existing key pair or create a new key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about removing existing key pairs from a public AMI .

Choose an existing key pair

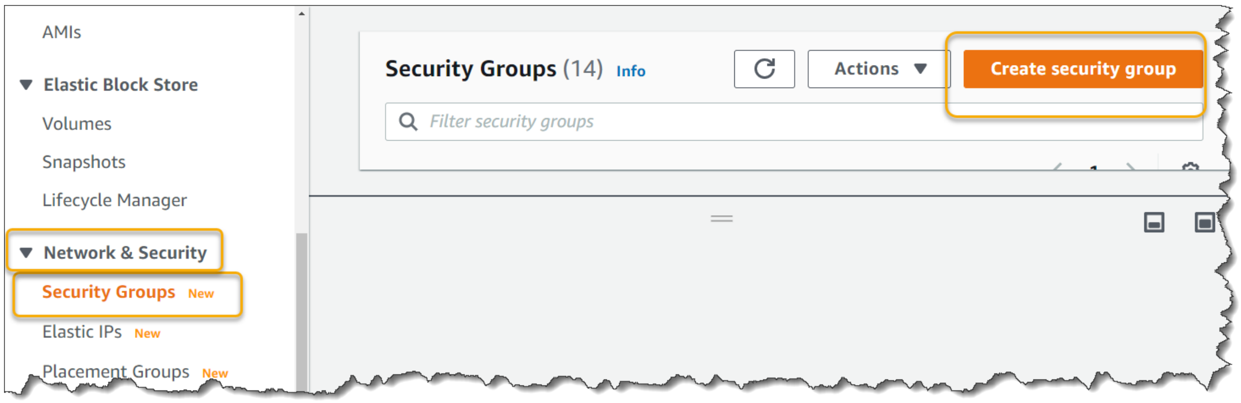
Select a key pair

hybridskill-app-server-key

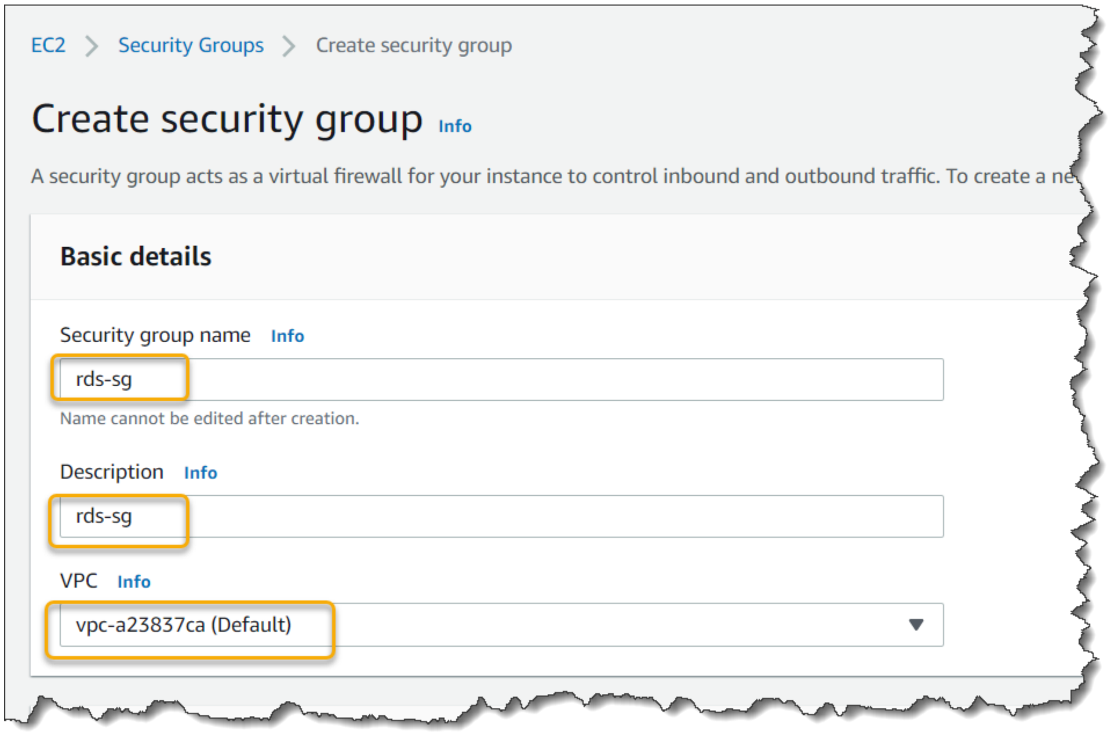
[✓] I acknowledge that I have access to the selected private key file (hybridskill-app-server-key.pem), and that without this file, I won't be able to log into my instance.

Task 2: Create a Security Group for RDS

1. On the main EC2 dashboard under **Network & Security**, click **Security Groups** and click **Ceate Security Group.**



2. Enter **rds-sg** as the **Security group name**, enter a short **Description** and make sure the default **VPC** is selected.


EC2 > Security Groups > Create security group

Create security group Info

A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a ne

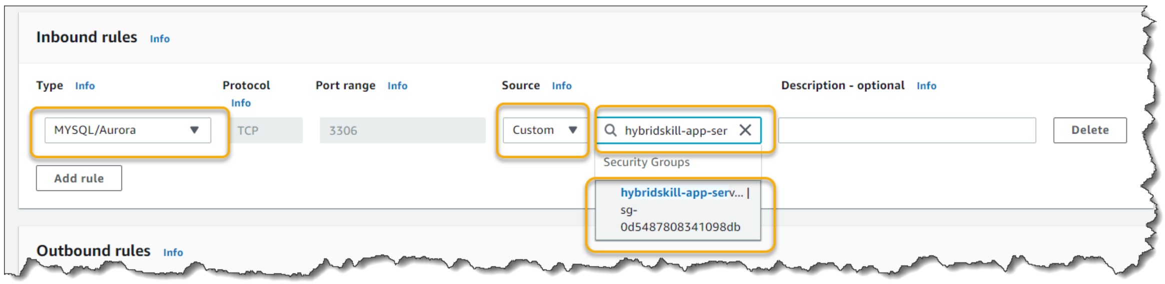
Basic details

Security group name Info
rds-sg
Name cannot be edited after creation.

Description Info
rds-sg

VPC Info
vpc-a23837ca (Default)

Under Inbound Rules, select **MySQL/Aurora** from the dropdown and under **Source.** select **Custom** and find and select your **app-server security group** you used for your EC2 instance **.**


Inbound rules Info

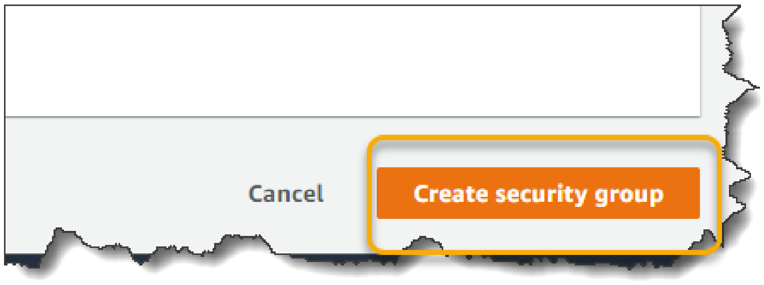
Type	Protocol	Port range	Source		Description - optional	
MYSQL/Aurora	TCP	3306	Custom	hybridskill-app-ser		Delete

Security Groups
hybridskill-app-serv... | sg-0d5487808341098db

Add rule

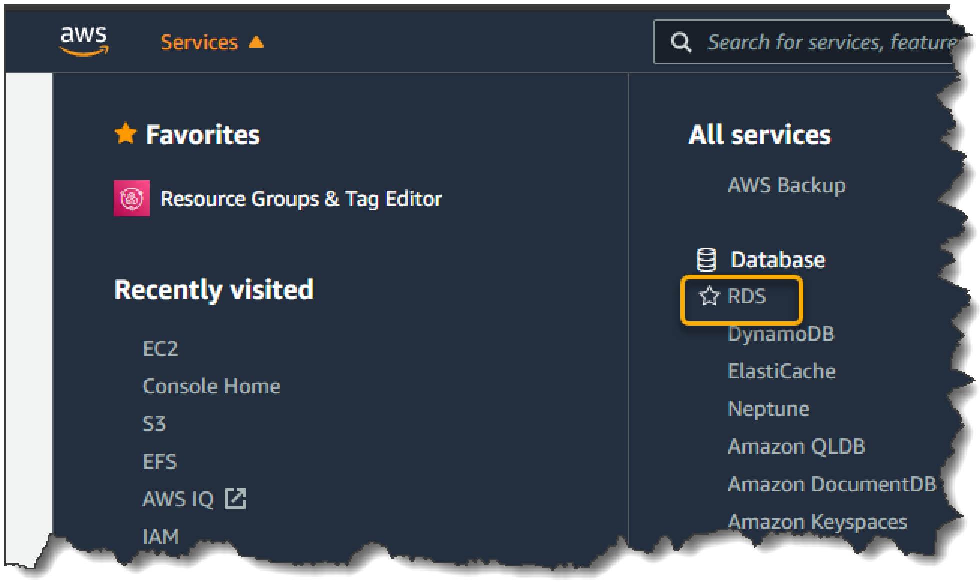
Outbound rules Info

Leave other options as default and click **Create security group**

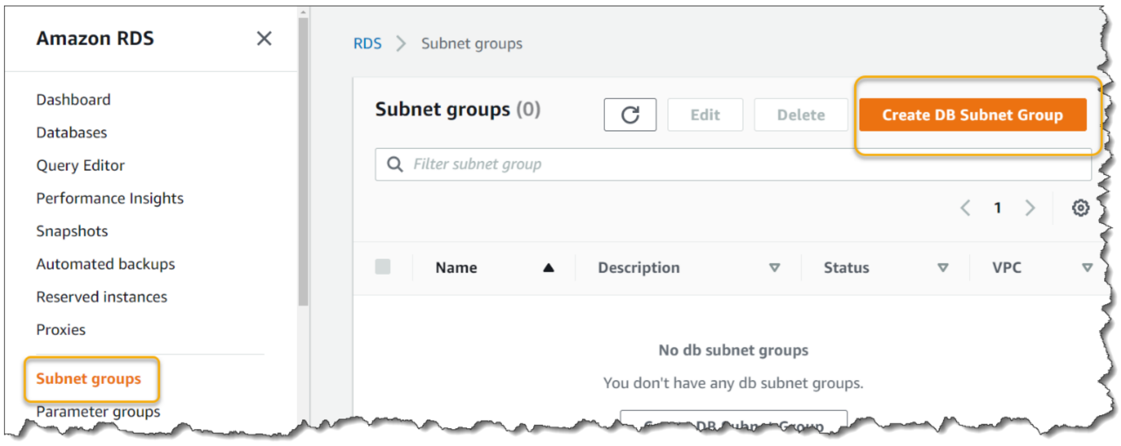


Task 3: Create a Subnet Group for RDS

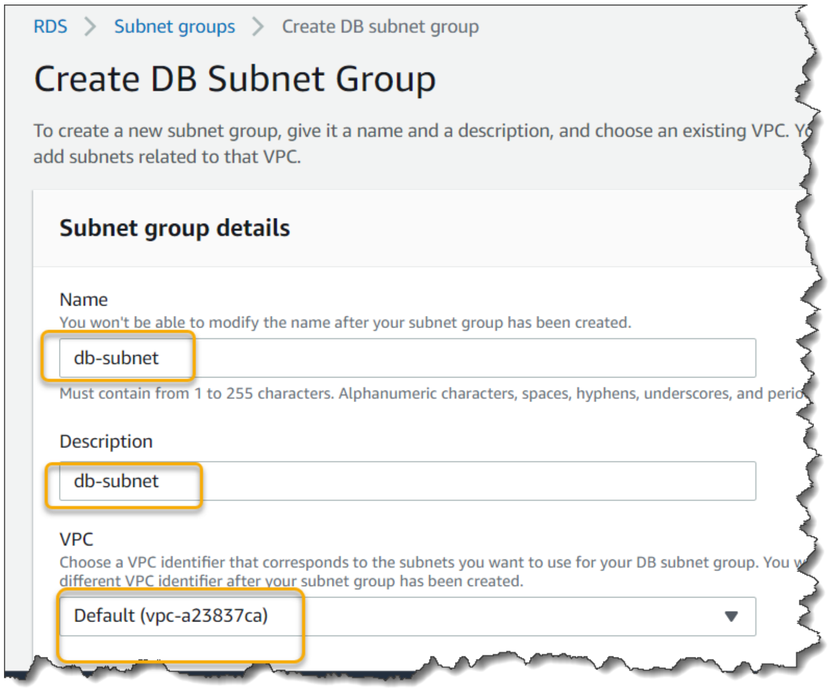
1. Click **Services** under **Database**, click **Relational Database Service.**



2. From the left hand side of the RDS dashboard, click **Subnet groups** and click **Create DB Subnet Group.**



3. Enter the **Name** as **db-subnet**, enter a short **Description**, select the **default VPC**



RDS > Subnet groups > Create DB subnet group

Create DB Subnet Group

To create a new subnet group, give it a name and a description, and choose an existing VPC. Y
add subnets related to that VPC.

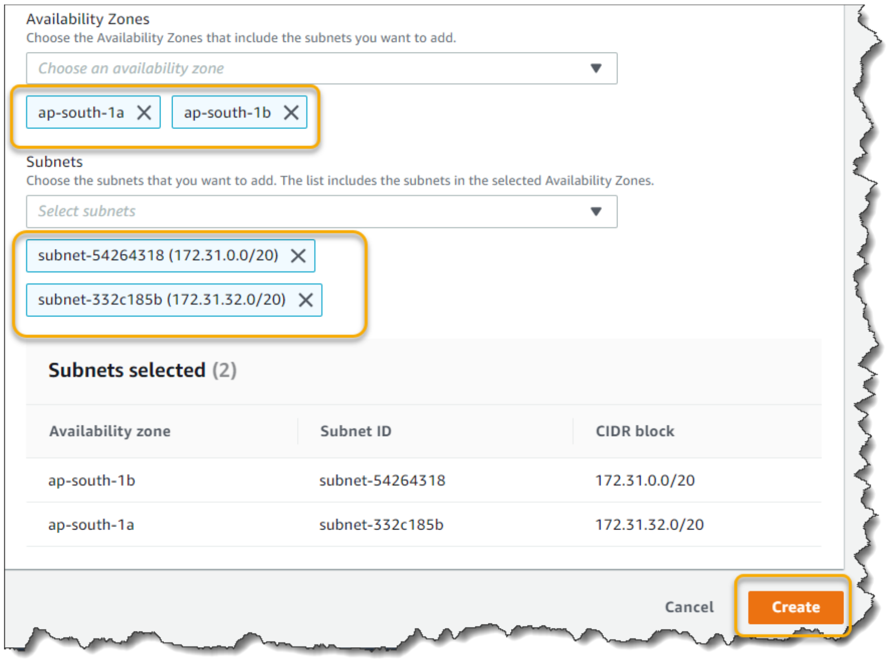
Subnet group details

Name
You won't be able to modify the name after your subnet group has been created.
db-subnet
Must contain from 1 to 255 characters. Alphanumeric characters, spaces, hyphens, underscores, and perio

Description
db-subnet

VPC
Choose a VPC identifier that corresponds to the subnets you want to use for your DB subnet group. You w
different VPC identifier after your subnet group has been created.
Default (vpc-a23837ca)

Choose **Availability zones a** and **b** and their respective **subnets** and click **Create.**



Availability Zones
Choose the Availability Zones that include the subnets you want to add.
Choose an availability zone
ap-south-1a ap-south-1b

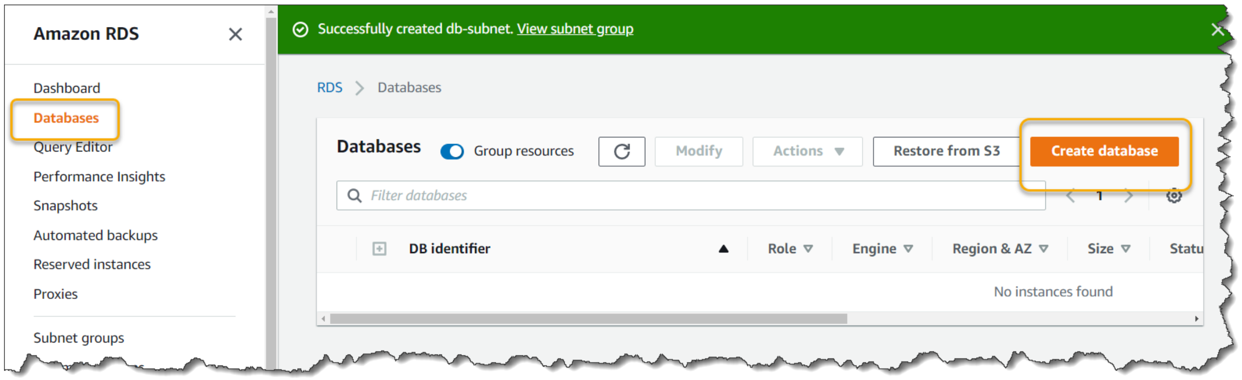
Subnets
Choose the subnets that you want to add. The list includes the subnets in the selected Availability Zones.
Select subnets
subnet-54264318 (172.31.0.0/20)
subnet-332c185b (172.31.32.0/20)

Subnets selected (2)

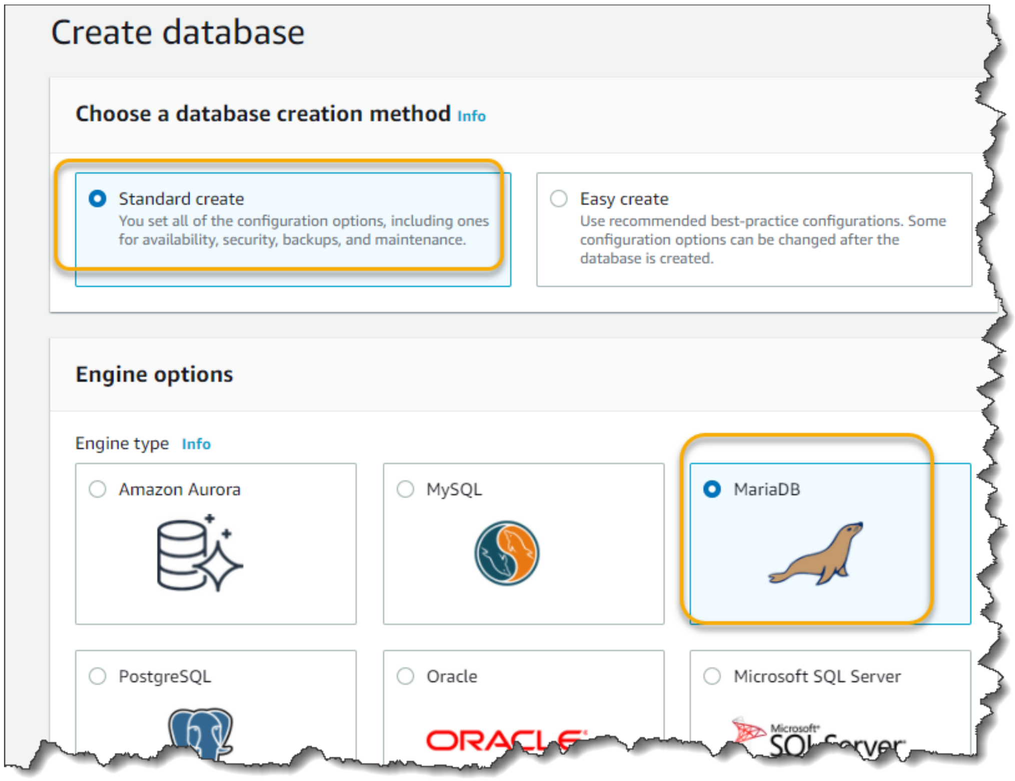
Availability zone	Subnet ID	CIDR block
ap-south-1b	subnet-54264318	172.31.0.0/20
ap-south-1a	subnet-332c185b	172.31.32.0/20

Task 4: Create an RDS Instance.

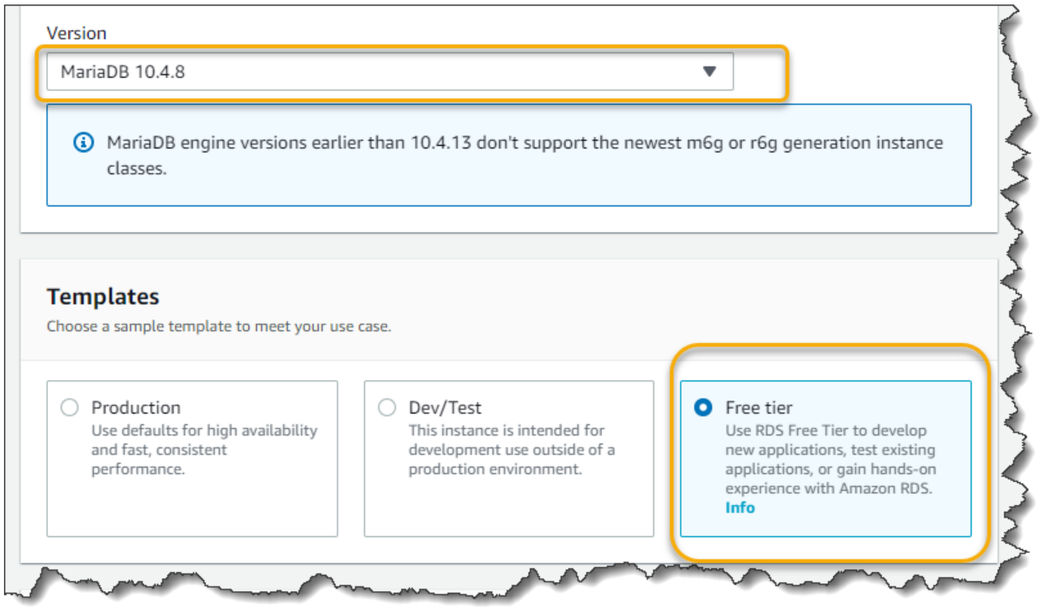
1. On the main RDS dashboard, click **Databases** and then **Create database**.



2. . Choose **Standard Create**. Select **MariaDB** as the database engine and click **Next.**


Create database
Choose a database creation method Info
Standard create
You set all of the configuration options, including ones for availability, security, backups, and maintenance.
Easy create
Use recommended best-practice configurations. Some configuration options can be changed after the database is created.
Engine options
Engine type Info
Amazon Aurora
MySQL
MariaDB
PostgreSQL
Oracle
Microsoft SQL Server

Choose the latest version of MariaDB and Make sure the **Free Tier Template is** selected.


Version

MariaDB 10.4.8

MariaDB engine versions earlier than 10.4.13 don't support the newest m6g or r6g generation instance classes.

Templates

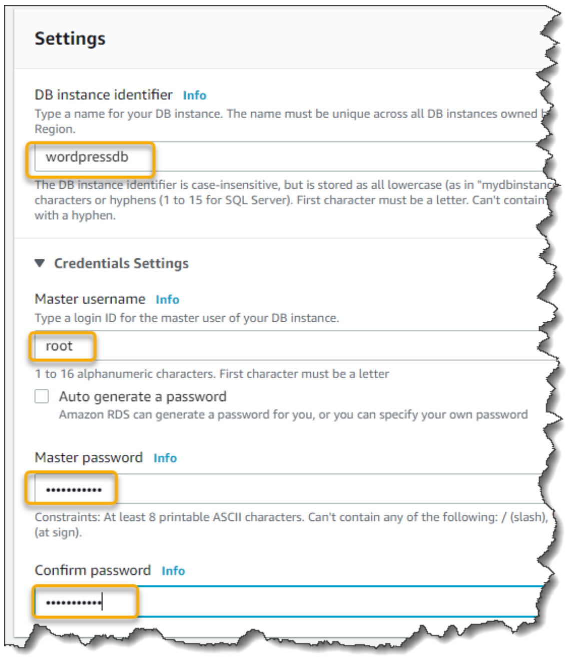
Choose a sample template to meet your use case.

Production
Use defaults for high availability and fast, consistent performance.

Dev/Test
This instance is intended for development use outside of a production environment.

[✓ Free tier]
Use RDS Free Tier to develop new applications, test existing applications, or gain hands-on experience with Amazon RDS. Info

3. Under **Settings** Enter **wordpressdb** as **DB instance identifier**, enter **root** as the **Master Username**. Enter and confirm a strong **Master password**.


Settings

DB instance identifier Info

Type a name for your DB instance. The name must be unique across all DB instances owned
Region.

wordpressdb

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstanc
characters or hyphens (1 to 15 for SQL Server). First character must be a letter. Can't contain
with a hyphen.

▼ Credentials Settings

Master username Info

Type a login ID for the master user of your DB instance.

root

1 to 16 alphanumeric characters. First character must be a letter

☐ Auto generate a password
Amazon RDS can generate a password for you, or you can specify your own password

Master password Info

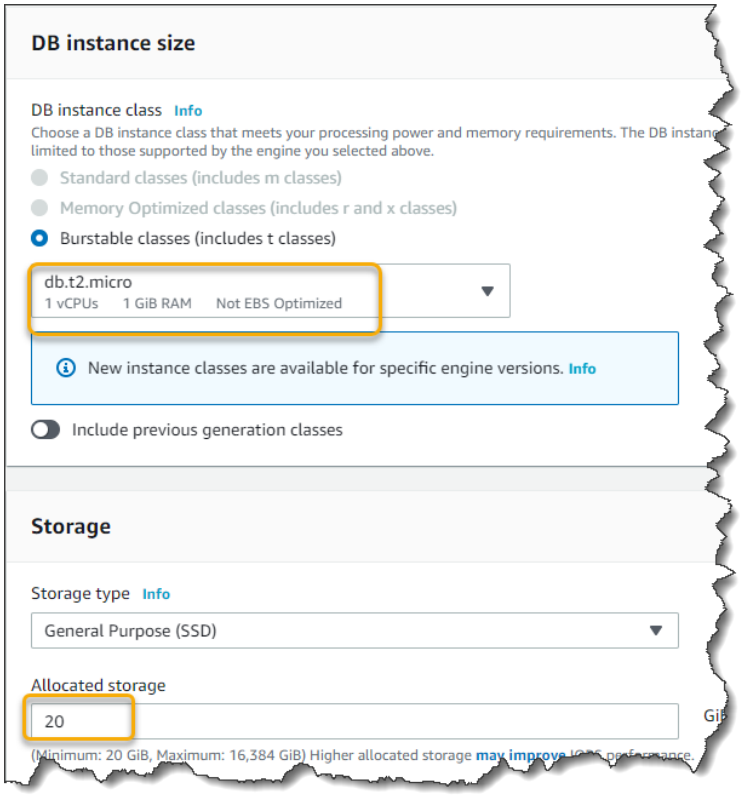
•••••••••••

Constraints: At least 8 printable ASCII characters. Can't contain any of the following: / (slash),
(at sign).

Confirm password Info

•••••••••••

4. Make sure instance size is **db.t2.micro** and storage size is **20 GB**, leave all options as default and scroll down.


DB instance size

DB instance class Info

Choose a DB instance class that meets your processing power and memory requirements. The DB instanc
limited to those supported by the engine you selected above.

- Standard classes (includes m classes)
- Memory Optimized classes (includes r and x classes)
- Burstable classes (includes t classes)

db.t2.micro
1 vCPUs 1 GiB RAM Not EBS Optimized

New instance classes are available for specific engine versions. Info

Include previous generation classes

Storage

Storage type Info

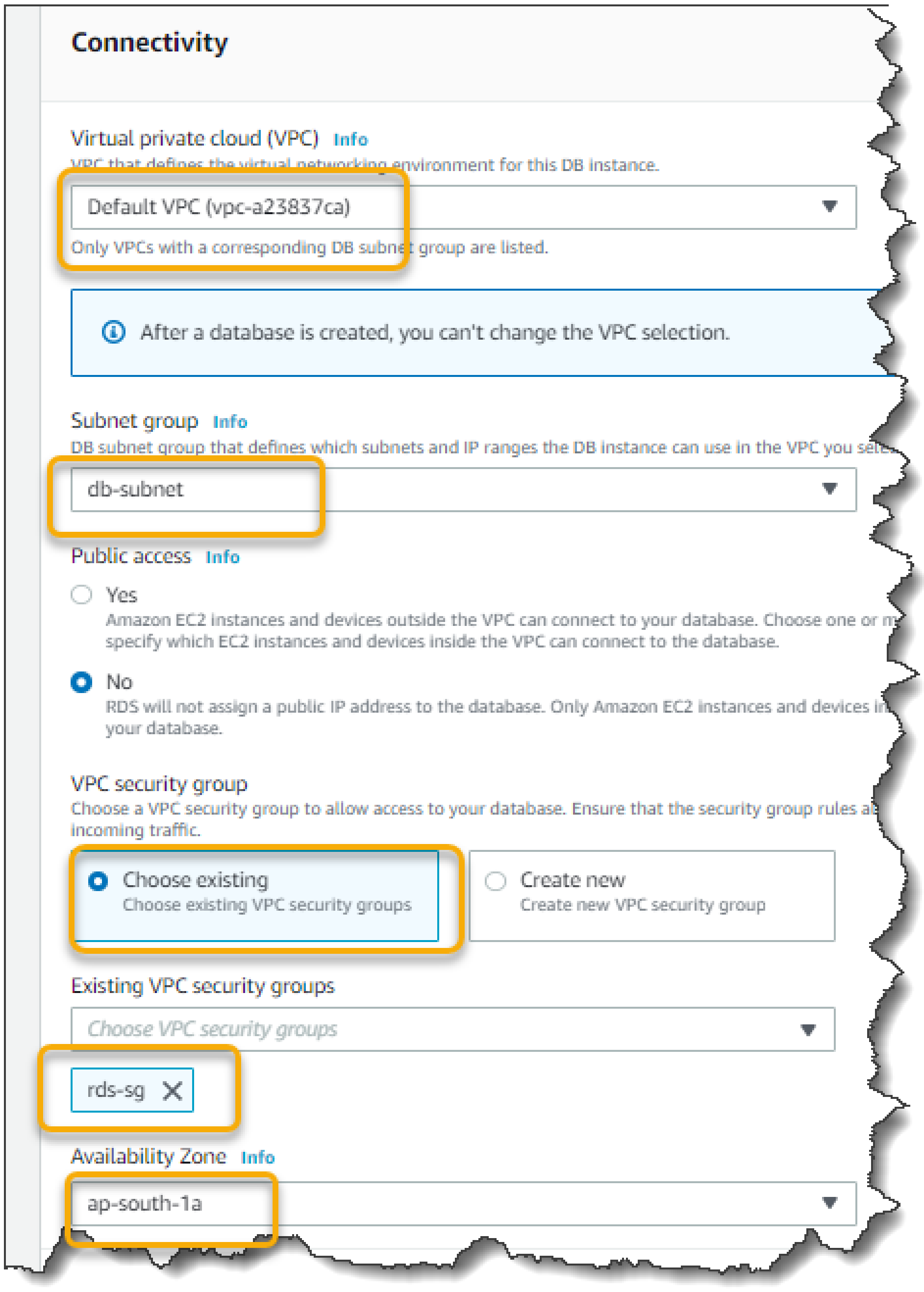
General Purpose (SSD)

Allocated storage

20 Gi

(Minimum: 20 GiB, Maximum: 16,384 GiB) Higher allocated storage may improve IOPS performance.

5. Select your **Default VPC**, select the **db-subnet** group you create earlier. Make sure **Public accessibility** is set to **No**. *(This should be the same zone as your EC2 instance).* **Select existing VPC Security groups and select rds-sg** *(The one you created earlier).* Select **ap-south-1a** as the **Availabilty Zone** Scroll down for further options.



Connectivity

Virtual private cloud (VPC) Info
VPC that defines the virtual networking environment for this DB instance.
Default VPC (vpc-a23837ca)
Only VPCs with a corresponding DB subnet group are listed.

After a database is created, you can't change the VPC selection.

Subnet group Info
DB subnet group that defines which subnets and IP ranges the DB instance can use in the VPC you sele
db-subnet

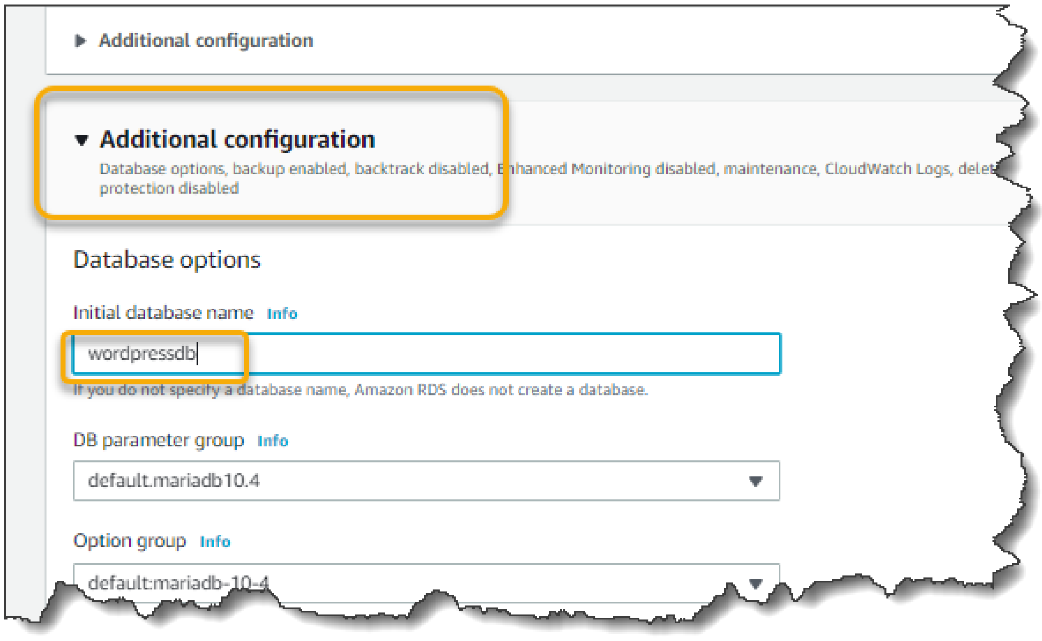
Public access Info
Yes [✓ No]
Yes: Amazon EC2 instances and devices outside the VPC can connect to your database. Choose one or m specify which EC2 instances and devices inside the VPC can connect to the database.
No: RDS will not assign a public IP address to the database. Only Amazon EC2 instances and devices in your database.

VPC security group
Choose a VPC security group to allow access to your database. Ensure that the security group rules a incoming traffic.
[✓ Choose existing] Choose existing VPC security groups
Create new: Create new VPC security group

Existing VPC security groups
Choose VPC security groups
rds-sg

Availability Zone Info
ap-south-1a

6. Expand **Additonal configuration**, enter the **Database name** as **wordpressdb** and scroll down to the bottom.



▾ Additional configuration

Database options, backup enabled, backtrack disabled, Enhanced Monitoring disabled, maintenance, CloudWatch Logs, delet protection disabled

Database options

Initial database name Info

wordpressdb

If you do not specify a database name, Amazon RDS does not create a database.

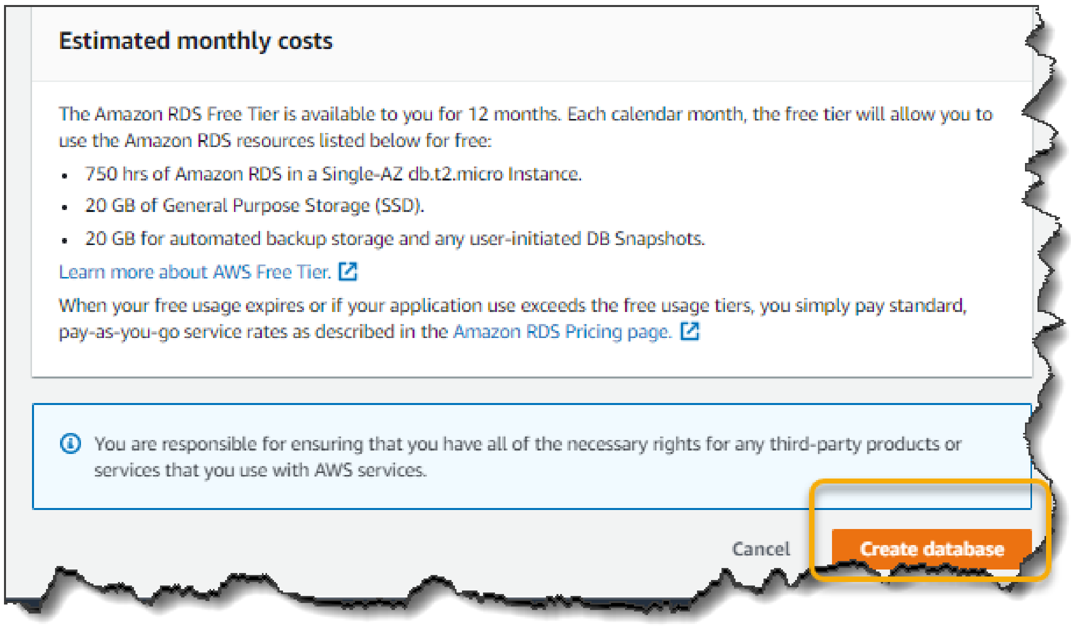
DB parameter group Info

default.mariadb10.4

Option group Info

default:mariadb-10-4

7. Leave all options as default and click **Create database**.



Estimated monthly costs

The Amazon RDS Free Tier is available to you for 12 months. Each calendar month, the free tier will allow you to use the Amazon RDS resources listed below for free:

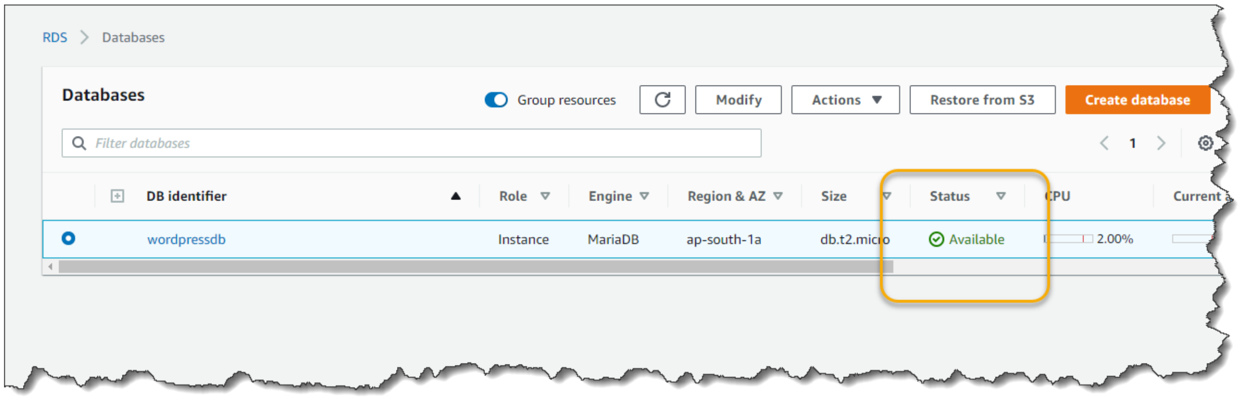
- 750 hrs of Amazon RDS in a Single-AZ db.t2.micro Instance.
- 20 GB of General Purpose Storage (SSD).
- 20 GB for automated backup storage and any user-initiated DB Snapshots.

Learn more about AWS Free Tier.

When your free usage expires or if your application use exceeds the free usage tiers, you simply pay standard, pay-as-you-go service rates as described in the Amazon RDS Pricing page.

You are responsible for ensuring that you have all of the necessary rights for any third-party products or services that you use with AWS services.

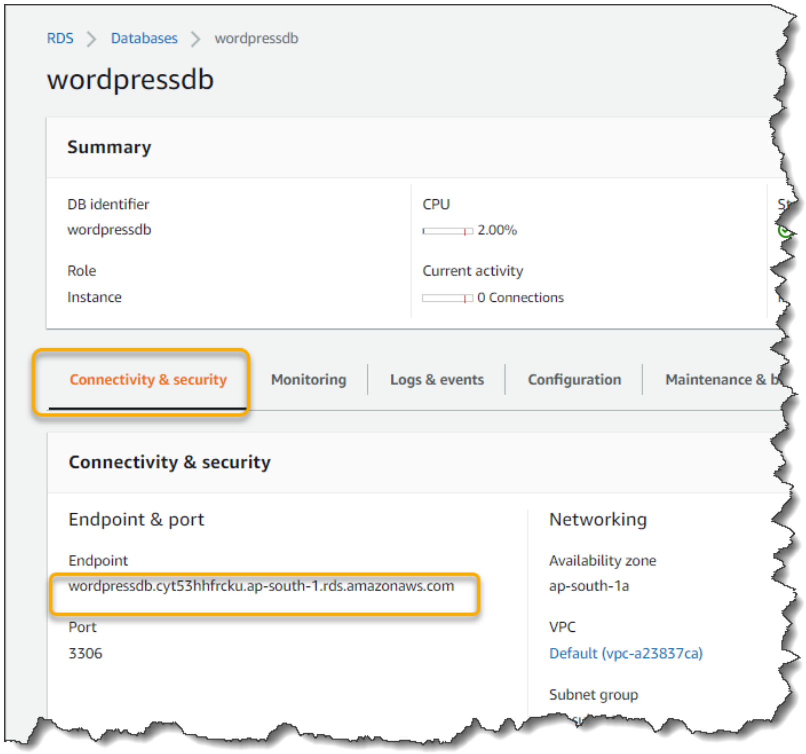
8. Wait a few minutes till the **Status** of the RDS instance becomes **available**


RDS > Databases

Databases

	DB identifier	Role	Engine	Region & AZ	Size	Status	CPU	Current
●	wordpressdb	Instance	MariaDB	ap-south-1a	db.t2.micro	Available	2.00%	

9. Select your instance and under **Connectivity & Security** make a note of the **Endpoint**.


RDS > Databases > wordpressdb

wordpressdb

Summary

DB identifier
wordpressdb

Role
Instance

CPU
2.00%

Current activity
0 Connections

Connectivity & security

Endpoint & port

Endpoint
wordpressdb.cyt53hhfrcku.ap-south-1.rds.amazonaws.com

Port
3306

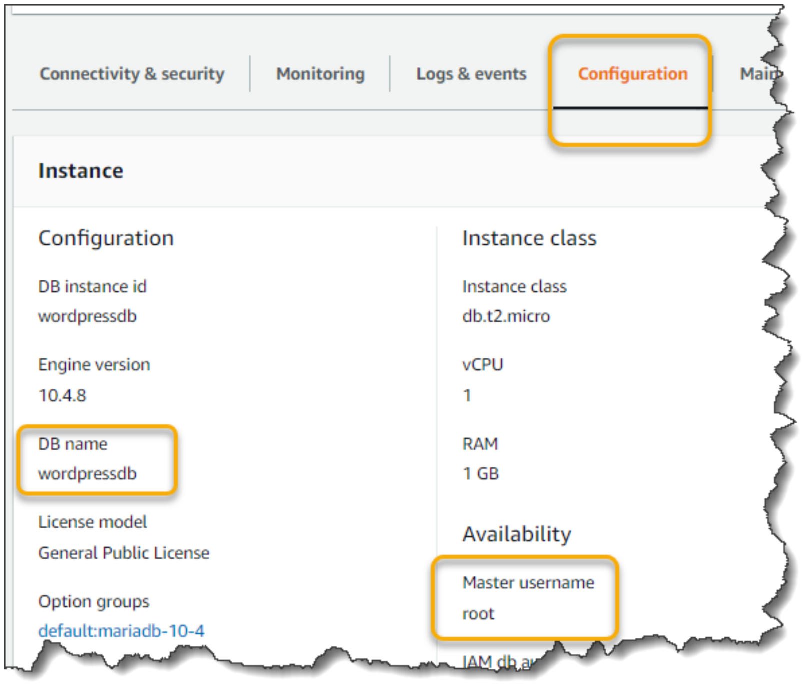
Networking

Availability zone
ap-south-1a

VPC
Default (vpc-a23837ca)

Subnet group

Also under **Configuration** Make a note of the **DB Name, Username**



Instance

Configuration

DB instance id
wordpressdb

Engine version
10.4.8

DB name
wordpressdb

License model
General Public License

Option groups
default:mariadb-10-4

Instance class

Instance class
db.t2.micro

vCPU
1

RAM
1 GB

Availability

Master username
root

We will use this late on to take the backup of your database.

Task 5: Taking a backup of the Application Database.

1. Log in to your EC2 machine using SSH. Run the following commands.

```
cd /var/www/html/
sudo cp wp-config.php wp-config.php.bk
cat wp-config.php
```

2. Make a note of **DB_NAME, DB _USER** and **DB_PASSWORD.**

```
 */

// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpressdb');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'hybridskill@123');

/** MySQL hostname */
define('DB_HOST', 'localhost');

/** Database Charset to use in creating database tables. */
```

3. Enter the following commands to take the backup of your databases.

```
cd
mysql -u root -p wordpressdb
```

4. Enter the password of your database and you should be logged into your local database server. Take a look at the data by typing the below command.

```
show tables;
```

5. You should see the various tables present in your database. These are used by the wordpress application.

```
+------------------------+
| Tables_in_wordpressdb |
+------------------------+
| wp_commentmeta         |
| wp_comments            |
| wp_links               |
| wp_options             |
| wp_postmeta            |
| wp_posts               |
| wp_term_relationships |
| wp_term_taxonomy       |
| wp_termmeta            |
| wp_terms               |
| wp_usermeta            |
| wp_users               |
+------------------------+
12 rows in set (0.00 sec)
```

6. Next we are going to use a utility called mysqldump to take a backup of your database.

```
mysqldump -u root -p wordpressdb>backup.sql
tail backup.sql
```

7. Verify if the database dump is completed successfully

```
/*!40101 SET SQL_MODE=@OLD_SQL_MODE */;
/*!40014 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS */;
/*!40014 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS */;
/*!40101 SET CHARACTER_SET_CLIENT=@OLD_CHARACTER_SET_CLIENT */;
/*!40101 SET CHARACTER_SET_RESULTS=@OLD_CHARACTER_SET_RESULTS */;
/*!40101 SET COLLATION_CONNECTION=@OLD_COLLATION_CONNECTION */;
/*!40111 SET SQL_NOTES=@OLD_SQL_NOTES */;

-- Dump completed on 2018-02-20 20:43:58
```

Task 6: Restore Database to RDS.

1. Login to RDS by using a mysql client.

```
mysql --user=root --password --host=wordpressdb.cipyq5pcgii3.ap-south-1.rds.amazonaws.com --database=wordpressdb
```

2. Type the following command to verify the RDS is empty.

```
MariaDB [wordpressdb]> show tables;
Empty set (0.00 sec)
```

3. Type the following command to restore the data to RDS.
   Type Ctrl-C to exit

```
mysql --user=root --password=hybridskill123 --host=wordpressdb.cipyq5pcgii3.ap-south-1.rds.amazonaws.com --database=wordpressdb < backup.sql
```

4. Log back to RDS. Type the following command to verify the data has moved to RDS.

```
MariaDB [wordpressdb]> show tables;
+-----------------------+
| Tables_in_wordpressdb |
+-----------------------+
| wp_commentmeta        |
| wp_comments           |
| wp_links              |
| wp_options            |
| wp_postmeta           |
| wp_posts              |
| wp_term_relationships |
| wp_term_taxonomy      |
| wp_termmeta           |
| wp_terms              |
| wp_usermeta           |
| wp_users              |
+-----------------------+
12 rows in set (0.01 sec)
```

Task 6: Switch the application to RDS.

1. Run the following commands.

Type Ctrl-C to exit

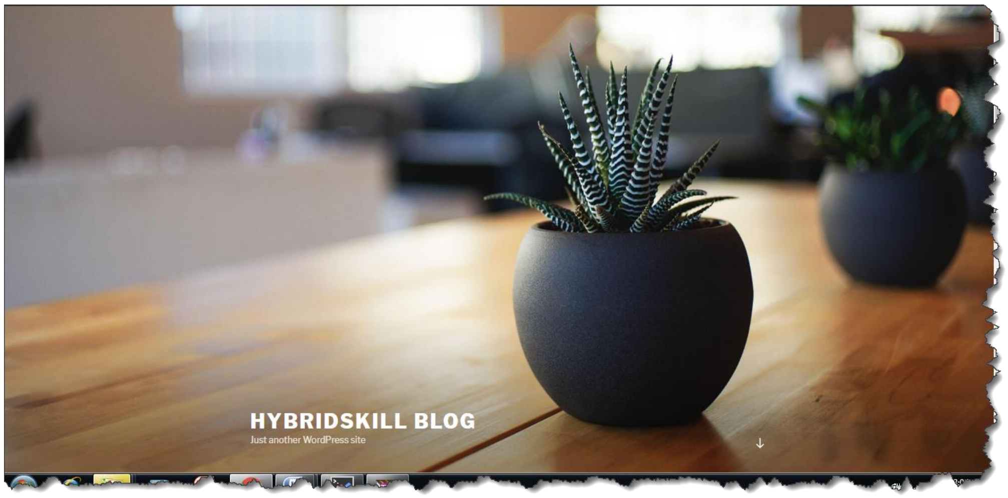
```
cd /var/www/html/
sudo vi wp-config.php
```

2. Enter your **RDS database name** for your **DB_NAME**. for **DB_USER** and **DB_PASSWORD**, enter your **RDS Master Username** and **Password** , and for **DB_HOST** enter your **RDS Endpoint**.

```
// ** MySQL settings - You can get this info from your web host ** //
/** The name of the database for WordPress */
define('DB_NAME', 'wordpressdb');

/** MySQL database username */
define('DB_USER', 'root');

/** MySQL database password */
define('DB_PASSWORD', 'YOURPASSWORD');

/** MySQL hostname */
define('DB_HOST', 'wordpressdb.xxxxxxxxxxxxxx.ap-south-1.rds.amazonaws.com');

/** Database Charset to use in creating database tables. */
define('DB_CHARSET', 'utf8');

/** The Database Collate type. Don't change this if in doubt. */
define('DB_COLLATE', '');
```

3. Type <Esc> <Shift + : > and type wq <enter> to save and exit the editor.

4. Enter the public IP of the instance on a browser and make sure the blog is up.



As a final check you can also stop the local mysql database. This will confirm wordpress is using RDS and not the local database.

```
sudo service mariadb stop
```

Task 7: Manage RDS through CLI

Now that we have explored RDS through the console, let's do the same through the CLI. Run the following commands on the command line interface that you had setup earlier.

1. Create a SG for RDS

```
:~$ aws ec2 create-security-group --group-name rds-sg-test --description "SG for rds"
{
    "GroupId": "sg-0774da6c"
}
```

2. Whitelist the app server for RDS

```
:~$ aws ec2 authorize-security-group-ingress --group-name  rds-sg-test --protocol tcp
--port 3306 --source-group hybridskill-sg-test
```

3. Create rds subnet group
   a) Run below command and get Subnet id

```
:~$aws ec2 describe-subnets
{
    "Subnets": [
        {
            "AvailabilityZone": "ap-south-1b",
            "AvailableIpAddressCount": 4089,
            "CidrBlock": "172.31.0.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "State": "available",
            "SubnetId": "subnet-3f312b72",
            "VpcId": "vpc-d6a00fbe",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": []
        },
        {
            "AvailabilityZone": "ap-south-1a",
            "AvailableIpAddressCount": 4090,
            "CidrBlock": "172.31.16.0/20",
            "DefaultForAz": true,
            "MapPublicIpOnLaunch": true,
            "State": "available",
            "SubnetId": "subnet-7e248f16",
            "VpcId": "vpc-d6a00fbe",
            "AssignIpv6AddressOnCreation": false,
            "Ipv6CidrBlockAssociationSet": []
        }
]
}
```

b) Create RDS Subnet Group:

```
:~$aws  rds  create-db-subnet-group  --db-subnet-group-name  awscli-test-sg  --db-subnet-
group-description "creating via awscli" --subnet-ids subnet-3f312b72 subnet-7e248f16
{
    "DBSubnetGroup": {
        "DBSubnetGroupName": "awscli-test-sg",
        "DBSubnetGroupDescription": "creating via awscli",
        "VpcId": "vpc-d6a00fbe",
        "SubnetGroupStatus": "Complete",
        "Subnets": [
            {
                "SubnetIdentifier": "subnet-3f312b72",
                "SubnetAvailabilityZone": {
                    "Name": "ap-south-1b"
                },
                "SubnetStatus": "Active"
            },
            {
                "SubnetIdentifier": "subnet-7e248f16",
                "SubnetAvailabilityZone": {
                    "Name": "ap-south-1a"
                },
                "SubnetStatus": "Active"
            }
        ],
        "DBSubnetGroupArn":    "arn:aws:rds:ap-south-1:123456789123:subgrp:awscli-test-
sg"
    }
}
```

4. Create RDS Create RDS instance:

```
:~$aws   rds   create-db-instance   --db-instance-identifier   testing-again   --db-name
wordpress  --allocated-storage 20  --db-instance-class db.t2.micro --engine mariadb --
master-username root --master-user-password hybridskill123
```

5. Login into RDS and check if database is created

```
:~$  mysql  --host  testing-again.cipyq5pcgii3.ap-south-1.rds.amazonaws.com  -u  root  -
phybridskill123
sql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| innodb             |
| mysql              |
| performance_schema |
| wordpress          |
+--------------------+
```