

WordPress Deployment on EC2 with Amazon RDS (MySQL)



Abhishek Bhagat



EC2
instance



WordPress



RDS
MySQL instance



Introducing Today's Project!

What the project is about?

This project showcases how I deployed a **WordPress application** on an **EC2 instance**, using **Amazon RDS** for the backend MySQL database. It follows a multi-tier architecture approach - separating the application and database layers - which improves scalability, maintainability, and security.



How I Used AWS Services in This Project

- **Amazon EC2:** Hosted Apache, PHP, and WordPress on Amazon Linux.
- **Amazon RDS (MySQL):** Used as the backend database for persistent storage.
- **Security Groups:** Controlled and restricted access between EC2 and RDS.
- **MySQL CLI:** Used on EC2 to connect to RDS and manually create the WordPress database and user.
- **Manual Setup:** WordPress downloaded and configured manually.



One Thing I Didn't Expect...

Initially, WordPress failed to connect to the database. After some debugging, I realized that the **RDS security group needed an inbound rule** allowing MySQL (port 3306) traffic **only from the EC2 security group** not just any IP. Once I corrected the rule and updated wp-config.php, the connection worked perfectly.



This Project Took Me...

~45 minutes to provision the infrastructure, install packages, troubleshoot connection errors, and fully deploy WordPress.



Project Steps Breakdown

1. Created RDS Instance (MySQL)

- MySQL engine
- Username: wpuser, Password: yourpassword
- DB Name: wordpressdb
- Public access: **Disabled**
- Security group: Inbound port 3306 from EC2 SG only

Settings

DB instance identifier [Info](#)
Type a name for your DB instance. The name must be unique across all DB instances owned by your AWS account in the current AWS Region.

wordpressdb

The DB instance identifier is case-insensitive, but is stored as all lowercase (as in "mydbinstance"). Constraints: 1 to 63 alphanumeric characters or hyphens. First character must be a letter. Can't contain two consecutive hyphens. Can't end with a hyphen.

▼ Credentials Settings

Master username [Info](#)
Type a login ID for the master user of your DB instance.

admin

1 to 16 alphanumeric characters. The first character must be a letter.

Credentials management
You can use AWS Secrets Manager or manage your master user credentials.

☐ Managed in AWS Secrets Manager - most secure
RDS generates a password for you and manages it throughout its lifecycle using AWS Secrets Manager.

☒ Self managed
Create your own password or have RDS create a password that you manage.

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

Master password [Info](#)

Password strength [Info](#)
Neutral

Minimum constraints: At least 8 printable ASCII characters. Can't contain any of the following symbols: / * @

Confirm master password [Info](#)

Aurora and RDS

Dashboard

Databases

Query editor

Performance insights

Snapshots

Exports in Amazon S3

Automated backups

Reserved instances

Proxies

Subnet groups

Parameter groups

Option groups

Custom engine versions

Zero-ETL integrations [New](#)

Events

Event subscriptions

Recommendations **0**

Successfully created database **wordpressdb** [View connection details](#)

You can use settings from wordpressdb to simplify configuration of suggested database add-ons while we finish creating your DB for you.

Databases (1) [Group resources](#) [Modify](#) [Actions](#) [Create database](#)

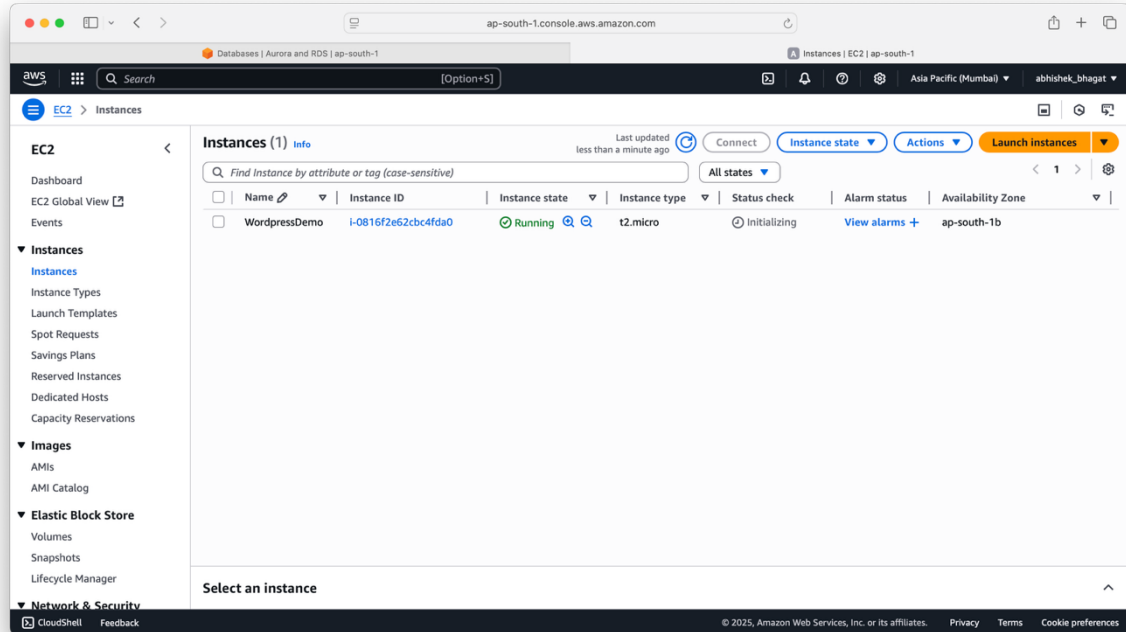
Filter by databases

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations
wordpressdb	Available	Instance	MySQL Community	ap-south-1b	db.t3.micro	



2. Launched EC2 Instance

- Amazon Linux 2023 AMI
- Connected via SSH using downloaded .pem key
- Installed MySQL client for DB setup



```
abhishekbhagat ~ ec2-user@ip-172-31-10-253:~ -- ssh -i ~/Downloads/devopsdemo.pem ec2-user@43.204.148.160 -- 126...
Install 5 Packages

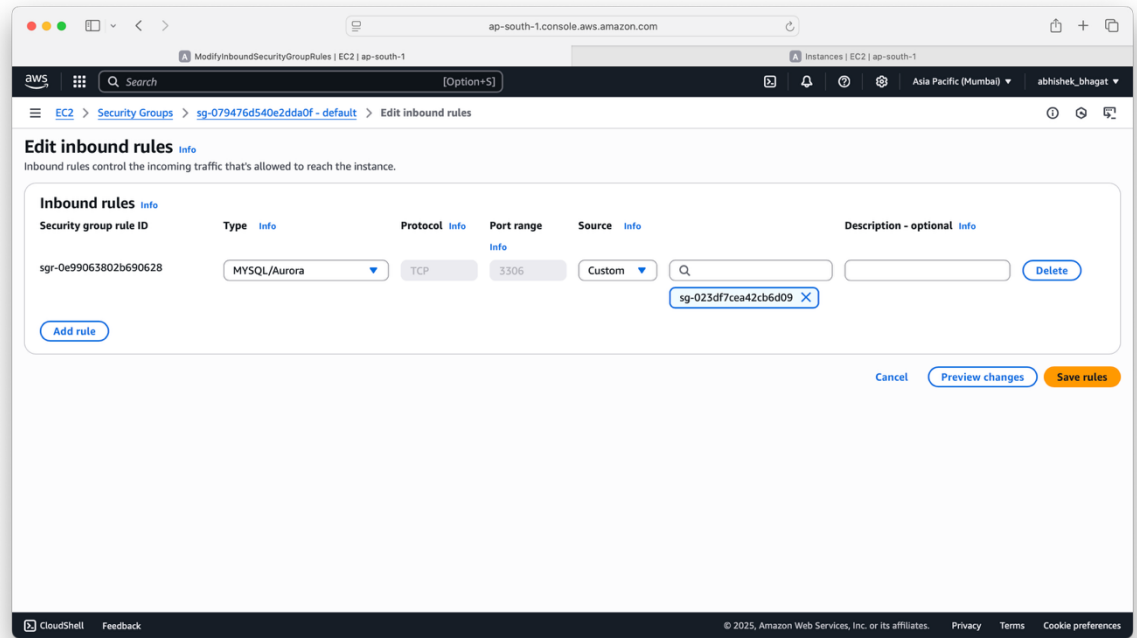
Total download size: 1.8 M
Installed size: 19 M
Is this ok [y/N]: y
Downloading Packages:
(1/5): mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch.rpm                293 kB/s | 9.9 kB   00:00
(2/5): mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64.rpm                    5.0 MB/s | 211 kB   00:00
(3/5): mariadb105-10.5.29-1.amzn2023.0.1.x86_64.rpm                          24 MB/s | 1.5 MB   00:00
(4/5): mariadb105-common-10.5.29-1.amzn2023.0.1.x86_64.rpm                   886 kB/s | 28 kB   00:00
(5/5): perl-Sys-Hostname-1.23-477.amzn2023.0.7.x86_64.rpm                     567 kB/s | 16 kB   00:00
-----
Total                                                                           18 MB/s | 1.8 MB   00:00
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing      : mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch                1/1
  Installing     : mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.x86_64                1/5
  Installing     : mariadb105-common-3:10.5.29-1.amzn2023.0.1.x86_64                  2/5
  Installing     : mariadb105-3:10.5.29-1.amzn2023.0.1.x86_64                        3/5
  Installing     : perl-Sys-Hostname-1.23-477.amzn2023.0.7.x86_64                     4/5
  Installing     : mariadb105-3:10.5.29-1.amzn2023.0.1.x86_64                        5/5
Running scriptlet: mariadb105-3:10.5.29-1.amzn2023.0.1.x86_64                        5/5
  Verifying      : mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64                  1/5
  Verifying      : mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch            2/5
  Verifying      : mariadb105-3:10.5.29-1.amzn2023.0.1.x86_64                      3/5
  Verifying      : mariadb105-common-3:10.5.29-1.amzn2023.0.1.x86_64                4/5
  Verifying      : perl-Sys-Hostname-1.23-477.amzn2023.0.7.x86_64                   5/5

Installed:
mariadb-connector-c-3.3.10-1.amzn2023.0.1.x86_64          mariadb-connector-c-config-3.3.10-1.amzn2023.0.1.noarch
mariadb105-3:10.5.29-1.amzn2023.0.1.x86_64              mariadb105-common-3:10.5.29-1.amzn2023.0.1.x86_64
perl-Sys-Hostname-1.23-477.amzn2023.0.7.x86_64

Complete!
[ec2-user@ip-172-31-10-253 ~]$ mysql --version
mysql Ver 15.1 Distrib 10.5.29-MariaDB, for Linux (x86_64) using EditLine wrapper
[ec2-user@ip-172-31-10-253 ~]$ export MYSQL_HOST=wordpressdb.cdmaioamqi0j.ap-south-1.rds.amazonaws.com
```

3. Connected RDS Instance to EC2

- Allow your EC2 instance to access your Amazon RDS Instance by changing the inbound rules of the Security Group.



4. Connect to RDS and Create WordPress User

- Find your RDS endpoint under Connectivity & security.
- Set an environment variable:

```
export MYSQL_HOST=<your-rds-endpoint>
```

- Connect to MySQL with your master credentials:

```
mysql --host=$MYSQL_HOST --user=<master_user> \  
--password=<master_password> \  
--database=wordpress
```

- In the MySQL console, run:

```
CREATE USER 'wordpress' IDENTIFIED BY '<strong_password>';  
GRANT ALL PRIVILEGES ON wordpress.* TO 'wordpress';  
FLUSH PRIVILEGES;
```

5. Installed Apache, PHP

```
sudo yum update -y
```

```
sudo yum install -y httpd php php-fpm php-json php-mysqlnd
```

```
sudo systemctl enable httpd
```

```
sudo systemctl start httpd
```

```
abhishekbhagat — ec2-user@ip-172-31-10-253:~ — ssh -i ~/Downloads/devopsdemo.pem ec2-user@43.204.148.160 — 126...
[ec2-user@ip-172-31-10-253 ~]$ sudo yum install -y httpd
Last metadata expiration check: 0:06:47 ago on Fri Jul 4 11:32:13 2025.
Dependencies resolved.
=====
Package                Architecture      Version           Repository        Size
=====
Installing:
httpd                  x86_64            2.4.62-1.amzn2023    amazonlinux        48 k
Installing dependencies:
apr                    x86_64            1.7.5-1.amzn2023.0.4    amazonlinux        129 k
apr-util               x86_64            1.6.3-1.amzn2023.0.1    amazonlinux        98 k
generic-logos-httpd    noarch            18.0.0-12.amzn2023.0.3    amazonlinux        19 k
httpd-core              x86_64            2.4.62-1.amzn2023      amazonlinux        1.4 M
httpd-filesystem        noarch            2.4.62-1.amzn2023      amazonlinux        14 k
httpd-tools             x86_64            2.4.62-1.amzn2023      amazonlinux        81 k
libbrotli               x86_64            1.0.9-4.amzn2023.0.2    amazonlinux        315 k
mailcap                 noarch            2.1.49-3.amzn2023.0.3    amazonlinux        33 k
Installing weak dependencies:
apr-util-openssl        x86_64            1.6.3-1.amzn2023.0.1    amazonlinux        17 k
mod_http2               x86_64            2.0.27-1.amzn2023.0.3    amazonlinux        166 k
mod_lua                  x86_64            2.4.62-1.amzn2023      amazonlinux        61 k
=====
Transaction Summary
=====
Install 12 Packages

Total download size: 2.3 M
Installed size: 6.9 M
Downloading Packages:
(1/12): apr-util-openssl-1.6.3-1.amzn2023.0.1.x86_64.rpm           506 kB/s | 17 kB  00:00
(2/12): apr-util-1.6.3-1.amzn2023.0.1.x86_64.rpm                 2.5 MB/s | 98 kB  00:00
(3/12): apr-1.7.5-1.amzn2023.0.4.x86_64.rpm                       2.8 MB/s | 129 kB  00:00
(4/12): generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch.rpm     893 kB/s | 19 kB  00:00
(5/12): httpd-2.4.62-1.amzn2023.x86_64.rpm                       2.1 MB/s | 48 kB  00:00
(6/12): httpd-filesystem-2.4.62-1.amzn2023.noarch.rpm             626 kB/s | 14 kB  00:00
(7/12): httpd-core-2.4.62-1.amzn2023.x86_64.rpm                  30 MB/s | 1.4 MB  00:00
(8/12): httpd-tools-2.4.62-1.amzn2023.x86_64.rpm                 2.3 MB/s | 81 kB  00:00
(9/12): libbrotli-1.0.9-4.amzn2023.0.2.x86_64.rpm               11 MB/s | 315 kB  00:00
(10/12): mailcap-2.1.49-3.amzn2023.0.3.noarch.rpm                1.6 MB/s | 33 kB  00:00
(11/12): mod_http2-2.0.27-1.amzn2023.0.3.x86_64.rpm             7.3 MB/s | 166 kB  00:00
```

6. Installed and Configured WordPress

```
wget https://wordpress.org/latest.tar.gz
```

```
tar -xzf latest.tar.gz
```

```
sudo cp -r wordpress/* /var/www/html/
```

```
sudo chown -R apache:apache /var/www/html/
```

```
cd /var/www/html
```

```
sudo cp wp-config-sample.php wp-config.php
```

```
sudo nano wp-config.php
```



In wp-config.php, I updated:

```
define('DB_NAME', 'wordpressdb');
```

```
define('DB_USER', 'wpuser');
```

```
define('DB_PASSWORD', 'yourpassword');
```

```
define('DB_HOST', '<rds-endpoint>:3306');
```

! Issues Faced

- WordPress page failed to load the first time due to DB connection issues.
- Fixed by adjusting RDS security group and verifying WordPress config file.
- Also had to ensure file permissions were correct in /var/www/html.
- Restarted Apache Server `sudo systemctl restart httpd`

✓ Final Outcome

- Successfully deployed a fully working WordPress site using Amazon EC2 and RDS.
- Frontend served via Apache and backend data persisted in a managed MySQL database - all with secure access control and scalable architecture.