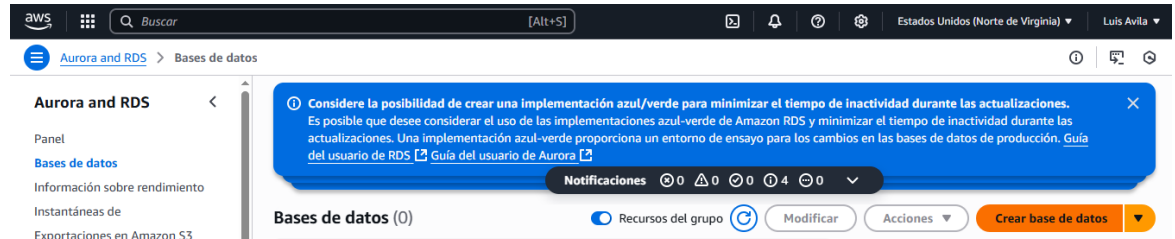


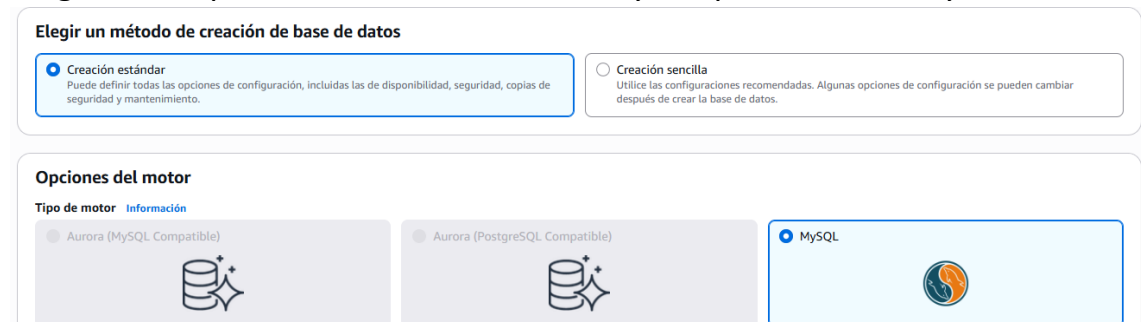
Desafío - Servicios de Base de Datos - Implementación de WordPress con Amazon RDS

1. Creación de la base de datos MySql con Amazon RDS.

En la consola de AWS le damos a la opción de “Crear base de datos”.



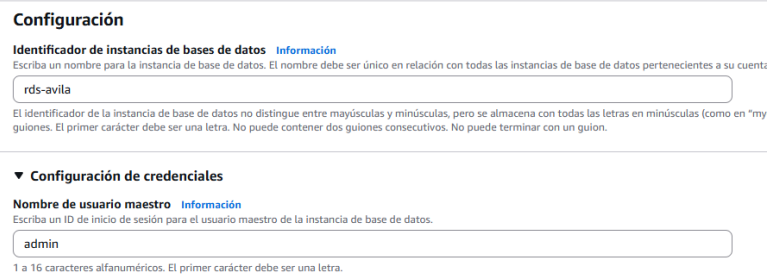
Elegimos la opción de Creación estándar y el tipo de motor MySQL.



En la plantilla elegimos por defecto la capa gratuita.



En la configuración le damos el nombre a la instancia de base de datos y en credenciales el nombre del usuario maestro.



En la administración de credenciales, elegimos la opción Autoadministrado para poder asignarle una contraseña a la BD.

Administración de credenciales
Puede usar AWS Secrets Manager o administrar sus credenciales de usuario maestro.

☐ Administrado en AWS Secrets Manager - *más seguro*
RDS genera una contraseña y la administra durante todo su ciclo de vida mediante AWS Secrets Manager.

☒ Autoadministrado
Cree su propia contraseña o pida a RDS que cree una contraseña para que pueda administrarla.

☐ Generar contraseña automáticamente
Amazon RDS puede generar una contraseña en su nombre, o bien puede especificar su propia contraseña.

Contraseña maestra | Información

Seguridad de la contraseña | **Fuerte**

Restricciones mínimas: al menos 8 caracteres ASCII imprimibles. No puede contener ninguno de los siguientes símbolos: / * @

Confirmar la contraseña maestra | Información

En el almacenamiento asignado le daremos 20 GB.

Almacenamiento

Tipo de almacenamiento | Información
Los volúmenes de almacenamiento SSD de IOPS aprovisionadas (io2) ya están disponibles.

SSD de uso general (gp2)
Rendimiento de referencia determinado por el tamaño del volumen

Almacenamiento asignado | Información

20 GiB

El valor de almacenamiento asignado debe ser de 20 GiB a 6144 GiB

Una vez todo configurado le damos a la opción de “Crear base de datos”.

► **Configuración adicional**
Opciones de base de datos, cifrado activado, copia de seguridad activado, retroceder desactivado, mantenimiento, Registros de CloudWatch, eliminar protección desactivado.

Usted es responsable de asegurarse de que dispone de todos los derechos necesarios para cualquier producto o servicio de terceros que utilice con los servicios de AWS.

Cancelar **Crear base de datos**

Nos mostrará el mensaje que se creó la base de datos correctamente.

Se ha creado correctamente la base de datos **rds-avila** [Ver detalles de conexión](#) ✕

Puede utilizar la configuración de rds-avila para simplificar la configuración de complementos de base de datos sugeridos mientras terminamos de crear su base de datos.

Notificaciones 0 0 1 4 0

Bases de datos (1) ☒ Recursos del grupo [Modificar](#) [Acciones](#) **Crear base de datos**

Filtrar por bases de datos

Identificador de base de datos	Estado	Rol	Motor	Región ...	Tamaño
rds-avila	Backin...	Instancia	MySQL Co...	us-east-1d	db.t4g.m

2. Creación de instancia EC2 Linux e instalación de Wordpress

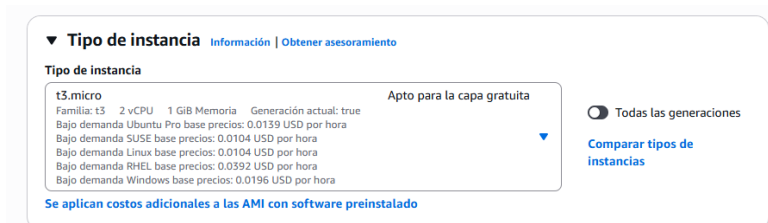
En la consola de AWS, buscamos instancias y le damos a la opción Lanzar instancia.



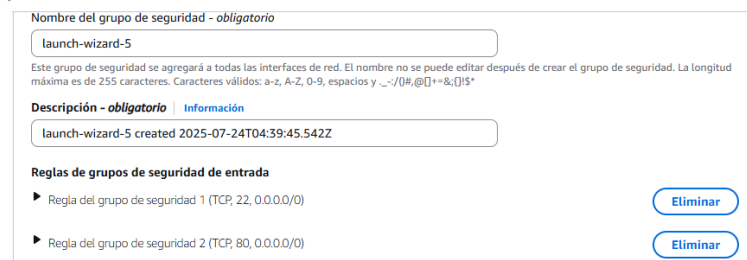
Elegimos la AMI de Amazon Linux.



Elegimos la instancia t3.micro.



En los grupos de seguridad vamos agregar la regla de seguridad SSH (22) y HTTP (80).



Una vez todo configurado le damos a “Lanzar instancia”.

Número de instancias

Información

1

Imagen de software (AMI)

Amazon Linux 2023 AMI 2023.8.2...[más información](#)

ami-0cbb2c6a1bb2ad653

Tipo de servidor virtual (tipo de instancia)

t3.micro

Firewall (grupo de seguridad)

Nuevo grupo de seguridad

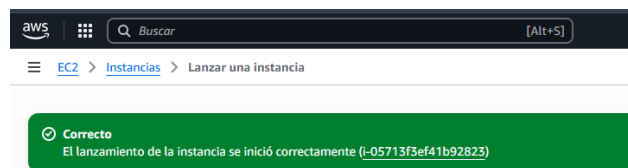
Almacenamiento (volúmenes)

1 x 8 GiB (SSD) x 1

Cancelar

Lanzar instancia

Una vez lanzada, nos mostrará el mensaje que el lanzamiento se inicio correctamente



Luego seleccionamos la instancia y le damos a conectar para que nos de la IP pública.

Instancias (1/3)

Información

Conectar

	Name	ID de la instancia	Estado de la i...
<input type="checkbox"/>	ServidorDesafio	i-0fa08ef6b1e08267b	Detenida
<input checked="" type="checkbox"/>	ServidorLuis	i-05713f34f41b92823	En ejecución
<input type="checkbox"/>	servidor linux	i-0c1f806b3eec5eff5	Detenida

En putty nos conectamos como SSH usando la ip publica de la instancia C2.

[illegible]

Con el comando **sudo dnf update -y** actualizamos los paquetes

```
ec2-user@ip-172-31-33-26-~  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf update -y  
Amazon Linux 2023 Kernel Livepatch repository 188 kB/s | 18 kB 00:00  
=====  
WARNING:  
A newer release of "Amazon Linux" is available.  
  
Available Versions:  
  
Version 2023.8.20250721:  
Run the following command to upgrade to 2023.8.20250721:  
  
dnf upgrade --releasever=2023.8.20250721  
  
Release notes:  
https://docs.aws.amazon.com/linux/al2023/release-notes/relnotes-2023.8.20250721.html  
=====  
Dependencies resolved.  
Nothing to do.  
Complete!  
[ec2-user@ip-172-31-33-26 ~]$
```

Con el comando **sudo dnf install httpd -y** instalamos Apache que será nuestro servidor web

```
ec2-user@ip-172-31-33-26-~  
Complete!  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install httpd -y  
Last metadata expiration check: 0:00:45 ago on Thu Jul 24 05:01:51 2025.  
Dependencies resolved.  
=====  
Package Arch 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  
-----
```

Con el comando **sudo dnf install php php-mysqlnd php-fpm php-xml php-json php-cli -y** instalamos PHP

```
ec2-user@ip-172-31-33-26-~  
Complete!  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install php php-mysqlnd php-fpm php-xml p  
hp-json php-cli -y  
Last metadata expiration check: 0:01:52 ago on Thu Jul 24 05:01:51 2025.  
Dependencies resolved.  
=====  
Package Arch Version Repository Size  
-----  
Installing:  
php8.4 x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 17 k  
php8.4-fpm x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 2.0 M  
php8.4-mysqlnd x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 156 k  
Installing dependencies:  
libsodium x86_64 1.0.18-4.amzn2023 amazonlinux 176 k  
libxml2 x86_64 1.1.43-1.amzn2023.0.1 amazonlinux 183 k  
nginx-filesystem noarch 1:1.28.0-1.amzn2023.0.1 amazonlinux 9.5 k  
php8.4-cli x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 3.8 M  
php8.4-common x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 799 k  
php8.4-pdo x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 100 k  
php8.4-process x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 52 k  
php8.4-xml x86_64 8.4.8-1.amzn2023.0.1 amazonlinux 973 k  
Installing weak dependencies:  
-----
```

Con el comando **sudo dnf install -y mariadb105-server** instalamos MariaDB.

```
ec2-user@ip-172-31-33-26 ~$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install -y mariadb105-server  
Last metadata expiration check: 0:02:37 ago on Thu Jul 24 05:01:51 2025.  
Dependencies resolved.  


| Package                  | Arch   | Version                    | Repository  | Size  |
|--------------------------|--------|----------------------------|-------------|-------|
| Installing:              |        |                            |             |       |
| mariadb105-server        | x86_64 | 3:10.5.29-1.amzn2023.0.1   | amazonlinux | 10 M  |
| Installing dependencies: |        |                            |             |       |
| mariadb-connector-c      | x86_64 | 3.3.10-1.amzn2023.0.1      | amazonlinux | 211 k |
| mariadb-connector-c-odbc | noarch | 3.3.10-1.amzn2023.0.1      | amazonlinux | 9.9 k |
| mariadb105               | x86_64 | 3:10.5.29-1.amzn2023.0.1   | amazonlinux | 1.5 M |
| mariadb105-common        | x86_64 | 3:10.5.29-1.amzn2023.0.1   | amazonlinux | 28 k  |
| mariadb105-errmsg        | x86_64 | 3:10.5.29-1.amzn2023.0.1   | amazonlinux | 212 k |
| mysql-libs               | noarch | 1.0.4-2.amzn2023.0.3       | amazonlinux | 36 k  |
| perl-B                   | x86_64 | 1.80-477.amzn2023.0.7      | amazonlinux | 177 k |
| perl-DBD-MariaDB         | x86_64 | 1.22-1.amzn2023.0.4        | amazonlinux | 153 k |
| perl-DBI                 | x86_64 | 1.643-7.amzn2023.0.3       | amazonlinux | 700 k |
| perl-Data-Dumper         | x86_64 | 2.174-460.amzn2023.0.2     | amazonlinux | 55 k  |
| perl-File-Copy           | noarch | 2.34-477.amzn2023.0.7      | amazonlinux | 20 k  |
| perl-FileHandle          | noarch | 2.03-477.amzn2023.0.7      | amazonlinux | 15 k  |
| perl-Math-BigInt         | noarch | 1:1.9998.39-2.amzn2023.0.2 | amazonlinux | 202 k |


```

Con el comando **sudo dnf install wget -y** instalamos wget (sirve para instalar paquetes).

```
ec2-user@ip-172-31-33-26 ~$  
Tasks: 177 (limit: 1057)  
Memory: 17.9M  
CPU: 42ms  
CGroup: /system.slice/httpd.service  
├─27917 /usr/sbin/httpd -DFOREGROUND  
├─27924 /usr/sbin/httpd -DFOREGROUND  
├─27925 /usr/sbin/httpd -DFOREGROUND  
├─27926 /usr/sbin/httpd -DFOREGROUND  
└─27927 /usr/sbin/httpd -DFOREGROUND  
  
Jul 24 05:05:59 ip-172-31-33-26.ec2.internal systemd[1]: Starting httpd.service:  
Jul 24 05:05:59 ip-172-31-33-26.ec2.internal systemd[1]: Started httpd.service:  
Jul 24 05:05:59 ip-172-31-33-26.ec2.internal httpd[27917]: Server configured, 12  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install wget -y  
Last metadata expiration check: 0:11:26 ago on Thu Jul 24 05:01:51 2025.  
Package wget-1.21.3-1.amzn2023.0.4.x86_64 is already installed.  
Dependencies resolved.  
Nothing to do.  
Complete!  
[ec2-user@ip-172-31-33-26 ~]$
```

Con el comando **wget <https://wordpress.org/latest.tar.gz>** podemos descargar WordPress.

```
ec2-user@ip-172-31-33-26 ~$  
Jul 24 05:05:59 ip-172-31-33-26.ec2.internal httpd[27917]: Server configured, 12  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install wget -y  
Last metadata expiration check: 0:11:26 ago on Thu Jul 24 05:01:51 2025.  
Package wget-1.21.3-1.amzn2023.0.4.x86_64 is already installed.  
Dependencies resolved.  
Nothing to do.  
Complete!  
[ec2-user@ip-172-31-33-26 ~]$ wget https://wordpress.org/latest.tar.gz  
--2025-07-24 05:14:08-- https://wordpress.org/latest.tar.gz  
Resolving wordpress.org (wordpress.org)... 198.143.164.252  
Connecting to wordpress.org (wordpress.org)[198.143.164.252]:443... connected.  
HTTP request sent, awaiting response... 200 OK  
Length: 26925441 (26M) [application/octet-stream]  
Saving to: 'latest.tar.gz'  
  
latest.tar.gz 100%[=====] 25.68M 78.0MB/s in 0.3s  
2025-07-24 05:14:09 (78.0 MB/s) - 'latest.tar.gz' saved [26925441/26925441]  
[ec2-user@ip-172-31-33-26 ~]$
```

Con el comando **tar -xvzf latest.tar.gz** extraemos los archivos

```
wp-includes/widgets/class-wp-widget-media-video.php
8,631 100% 73.29KB/s 0:00:00 (xfr#3221, to-chk=9/3574)
wp-includes/widgets/class-wp-widget-media-image.php
15,367 100% 129.37KB/s 0:00:00 (xfr#3222, to-chk=8/3574)
wp-includes/widgets/class-wp-widget-meta.php
4,091 100% 34.44KB/s 0:00:00 (xfr#3223, to-chk=7/3574)
wp-includes/widgets/class-wp-widget-pages.php
5,720 100% 48.19KB/s 0:00:00 (xfr#3224, to-chk=6/3574)
wp-includes/widgets/class-wp-widget-recent-comments.php
7,060 100% 59.44KB/s 0:00:00 (xfr#3225, to-chk=5/3574)
wp-includes/widgets/class-wp-widget-recent-posts.php
5,938 100% 49.99KB/s 0:00:00 (xfr#3226, to-chk=4/3574)
wp-includes/widgets/class-wp-widget-rss.php
5,244 100% 44.19KB/s 0:00:00 (xfr#3227, to-chk=3/3574)
wp-includes/widgets/class-wp-widget-search.php
2,724 100% 22.93KB/s 0:00:00 (xfr#3228, to-chk=2/3574)
wp-includes/widgets/class-wp-widget-tag-cloud.php
6,778 100% 57.06KB/s 0:00:00 (xfr#3229, to-chk=1/3574)
wp-includes/widgets/class-wp-widget-text.php
21,311 100% 177.88KB/s 0:00:00 (xfr#3230, to-chk=0/3574)

sent 75,179,811 bytes received 63,569 bytes 0/0 bytes/sec
total size is 74,932,879 speedup is 1.00
[ec2-user@ip-172-31-33-26 ~]$
```

Con el comando **sudo rsync -avP wordpress/ /var/www/html/** movemos los archivos al directorio web de Apache:

[illegible]

Con el comando **sudo chown -R apache:apache /var/www/html/*** ajustamos los permisos.

```
ec2-user@ip-172-31-33-26 ~  
login as: ec2-user  
Authenticating with public key "desafionuevo"  
  
##### Amazon Linux 2023  
#####  
#####  
#####  
##### https://aws.amazon.com/linux/amazon-linux-2023  
#####  
##### V ->  
#####  
#####  
#####  
Last login: Thu Jul 24 05:11:55 2025 from 190.5.43.131  
[ec2-user@ip-172-31-33-26 ~]$  
[ec2-user@ip-172-31-33-26 ~]$ sudo rsync -avP wordpress /var/www/html/  
sending incremental file list  
  
sent 93,110 bytes received 367 bytes 62,318.00 bytes/sec  
total size is 74,932,879 speedup is 801.62  
[ec2-user@ip-172-31-33-26 ~]$ sudo chown -R apacheapache /var/www/html/*  
[ec2-user@ip-172-31-33-26 ~]$
```

Con el comando **sudo wget** <https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm> descargamos el archivo RPM del cliente MySQL.

```
ec2-user@ip-172-31-33-26 ~$ sudo systemctl restart httpd
[ec2-user@ip-172-31-33-26 ~]$ sudo wget https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm
--2025-07-24 05:24:23-- https://dev.mysql.com/get/mysql80-community-release-el9-1.noarch.rpm
Resolving dev.mysql.com (dev.mysql.com)... 23.49.176.249, 2600:1408:5400:4a7::2e31, 2600:1408:5400:4b3::2e31
Connecting to dev.mysql.com (dev.mysql.com)|23.49.176.249|:443... connected.
HTTP request sent, awaiting response... 302 Moved Temporarily
Location: https://repo.mysql.com/mysql80-community-release-el9-1.noarch.rpm [following]
--2025-07-24 05:24:24-- https://repo.mysql.com/mysql80-community-release-el9-1.noarch.rpm
Resolving repo.mysql.com (repo.mysql.com)... 184.25.42.21, 2600:1408:c400:1692::1d68, 2600:1408:c400:169d::1d68
Connecting to repo.mysql.com (repo.mysql.com)|184.25.42.21|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 10534 (10K) [application/x-redhat-package-manager]
Saving to: 'mysql80-community-release-el9-1.noarch.rpm'

mysql80-community-r 100%[=====] 10.29K --.-KB/s in 0s

2025-07-24 05:24:24 (295 MB/s) - 'mysql80-community-release-el9-1.noarch.rpm' saved [10534/10534]
```

Con el comando **sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y** vamos a instalar el archivo RPM:

```
ec2-user@ip-172-31-33-26 ~$ sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y
Last login: Thu Jul 24 05:21:24 2025 from 190.5.43.131
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install mysql80-community-release-el9-1.noarch.rpm -y
Last metadata expiration check: 0:24:30 ago on Thu Jul 24 05:01:51 2025.
Dependencies resolved.

Package Arch Version Repository Size
Installing:
mysql80-community-release noarch el9-1 @commandline 10 k

Transaction Summary
Install 1 Package
Total size: 10 k
```

Con el comando **sudo rpm --import** <https://repo.mysql.com/RPM-GPG-KEY-mysql-2023> importamos la clave pública de MySQL:

```
ec2-user@ip-172-31-33-26 ~$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023
Installed:
mysql80-community-release-el9-1.noarch

Complete!
[ec2-user@ip-172-31-33-26 ~]$ sudo rpm --import https://repo.mysql.com/RPM-GPG-KEY-mysql-2023
[ec2-user@ip-172-31-33-26 ~]$ sudo dnf install mysql-community-client -y
MySQL 8.0 Community Server 50 MB/s | 2.7 MB 00:00
MySQL Connectors Community 5.5 MB/s | 90 kB 00:00
MySQL Tools Community 39 MB/s | 1.2 MB 00:00
Dependencies resolved.

Package Arch Version Repository Size
Installing:
mysql-community-client x86_64 8.0.43-1.el9 mysql80-community 3.3 M
Installing dependencies:
mysql-community-client-plugins x86_64 8.0.43-1.el9 mysql80-community 1.4 M
mysql-community-common x86_64 8.0.43-1.el9 mysql80-community 556 k
mysql-community-libs x86_64 8.0.43-1.el9 mysql80-community 1.5 M
```


3. Configuración de Amazon RDS en la instancia EC2

Nos conectamos a la base de datos RDS utilizando el Endpoint, con el siguiente comando **mysql -h <rds-endpoint> -u admin -p**

```
ec2-user@ip-172-31-33-26:~$ mysql -h rds-avila.cmtwmmugehzy.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 46
Server version: 8.0.41 Source distribution

Copyright (c) 2000, 2019, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]>
```

Vamos a crear la database de wordpress, el usuario y la contraseña

```
MySQL [(none)]> CREATE DATABASE wordpress;
Query OK, 1 row affected (0.015 sec)

MySQL [(none)]> CREATE USER 'wpluis'@'%' IDENTIFIED BY 'desafio2025';
Query OK, 0 rows affected (0.014 sec)

MySQL [(none)]> GRANT ALL PRIVILEGES ON wordpress.* TO 'wpluis'@'%';
Query OK, 0 rows affected (0.007 sec)

MySQL [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.008 sec)

MySQL [(none)]>
```

Nos dirigimos a la carpeta de Wordpress y editamos el archivo de configuración

```
ec2-user@ip-172-31-33-26:/var/www/html$ mysql -h rds-avila.cmtwmmugehzy.us-east-1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MySQL connection id is 47
Server version: 8.0.41 Source distribution

Copyright (c) 2000, 2019, Oracle, MariaDB Corporation Ab and others.

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

MySQL [(none)]> CREATE DATABASE wordpress;
Query OK, 1 row affected (0.015 sec)

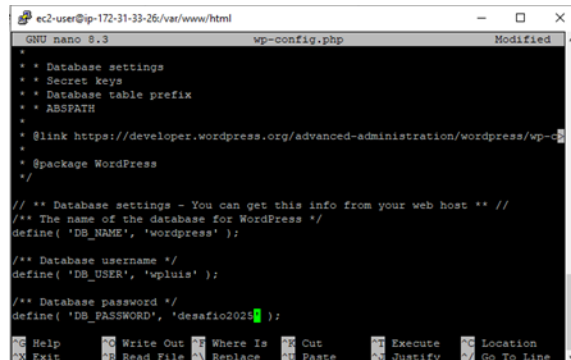
MySQL [(none)]> CREATE USER 'wpluis'@'%' IDENTIFIED BY 'desafio2025';
Query OK, 0 rows affected (0.014 sec)

MySQL [(none)]> GRANT ALL PRIVILEGES ON wordpress.* TO 'wpluis'@'%';
Query OK, 0 rows affected (0.007 sec)

MySQL [(none)]> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.008 sec)

MySQL [(none)]> EXIT;
Bye
[ec2-user@ip-172-31-33-26 ~]$ cd /var/www/html
[ec2-user@ip-172-31-33-26 html]$ sudo nano wp-config.php
```

Una vez dentro del archivo, tenemos que modificar las siguientes líneas con los datos de la base de datos RDS.



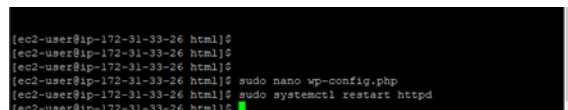
```
ec2-user@ip-172-31-33-26:~$ nano /var/www/html/wp-config.php
GNU nano 3.3 wp-config.php Modified
/*
 * Database settings
 * Secret keys
 * Database table prefix
 * ABSPATH
 */
* @link https://developer.wordpress.org/advanced-administration/wordpress/wp-config.php
*
* @package WordPress
*/

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

/** Database username */
define( 'DB_USER', 'wpuser' );

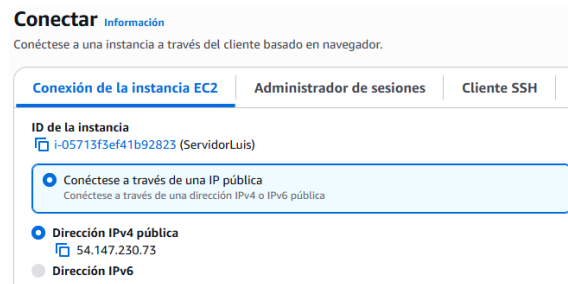
/** Database password */
define( 'DB_PASSWORD', 'desafio2025' );
```

Reiniciamos el servidor apache.



```
ec2-user@ip-172-31-33-26:~$ sudo nano wp-config.php
ec2-user@ip-172-31-33-26:~$ sudo systemctl restart httpd
```

Con la IP publica de la instancia, vamos a acceder al sitio de WordPress.



Conectar Información

Conéctese a una instancia a través del cliente basado en navegador.

Conexión de la instancia EC2 | Administrador de sesiones | Cliente SSH

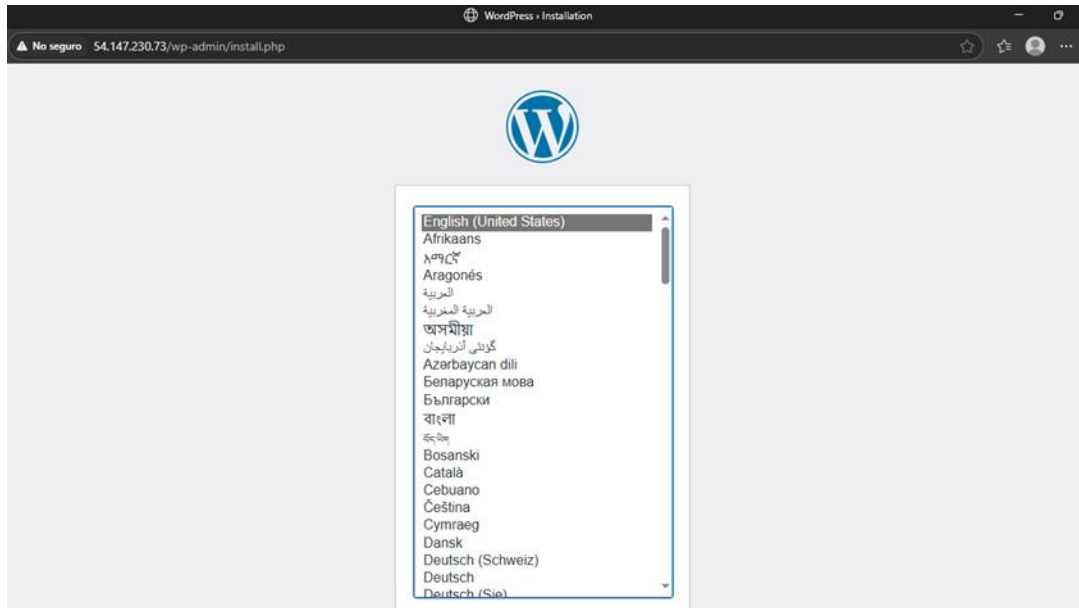
ID de la instancia
i-05713f3ef41b92823 (ServidorLuis)

Conéctese a través de una IP pública
Conéctese a través de una dirección IPv4 o IPv6 pública

Dirección IPv4 pública
54.147.230.73

Dirección IPv6

Vemos que instalamos y configuramos WordPress en la instancia EC2, conectándolo exitosamente a una base de datos en Amazon RDS.



REFLEXIÓN

Instalar WordPress en EC2 y conectarlo a una base de datos en Amazon RDS te proporciona un entorno de producción robusto, con un rendimiento superior, seguridad avanzada y escalabilidad fácil. Esta configuración es ideal tanto para sitios web pequeños como para aquellos que planean escalar a medida que crecen, todo mientras optimizas los costos y el tiempo de administración.