

Instalación de Wordpress en una instancia EC2 de AWS

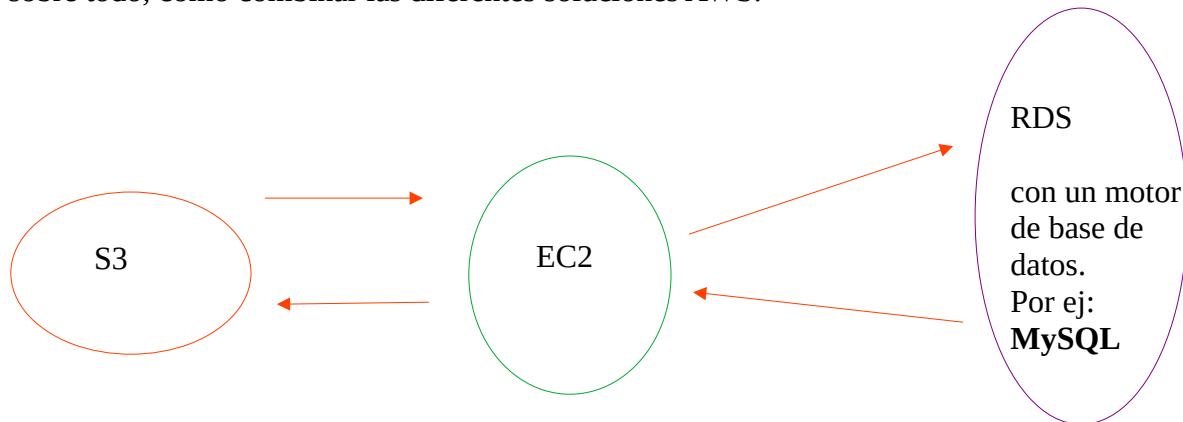
Vemos algunas de las indicaciones para instalar un “Wordpress” en una EC2, para empezar hay que hacer una serie de consideraciones, que en entornos productivos son más usuales:

- Wordpress es un CMS muy flexible que para funcionar se necesita de un Sistema Operativo que lo soporte, un motor de base de datos (MySQL o mariaDB, ambos son compatibles), Php, y unas cuantas dependencias más.
- Todo esto se puede instalar a mano en cualquier máquina compatible, pero dentro del propio AWS hay imágenes con todo ya preinstalado.
- Además hay diferentes métodos para conseguir dicho objetivo.

Por todo lo anterior, donde hay cientos de manuales y video tutoriales que se basan en todo eso. Hay una dolencia importante:

Son sólo una repetición de una instalación de una EC2 especializada. Donde todo está instalado, y configurado a gusto de alguien. Por lo tanto, si hay problemas → se desconoce su configuración, si hay vulnerabilidades → se desconoce su configuración, etc.

Aquí el objetivo es entender cómo funciona y se inter-relaciona sus tripas.
Y sobre todo, cómo combinar las diferentes soluciones AWS.



Necesitaremos los siguientes ingredientes:

- Un bucket **S3**, que usaremos como respaldo y almacenamiento de ficheros.
- Una **RDS** ya lanzada y configurada con el motor MySQL.
- Una instancia **EC2** (Debian en nuestro caso) que crearemos para este propósito específico, en la VPC que se creó con la RDS y que esté en la misma zona.

Necesitamos descargar una versión reciente de “WordPress”, en si sitio web: <https://wordpress.org/> terminado en org (ya que terminado en com en la página web de una empresa que aloja sitios web con esa tecnología, ya como servicio SaaS)

En <https://wordpress.org/download/> → descargamos la última versión.

Download and install it yourself

La propia web comenta las dependencias como PHP,...

For anyone comfortable getting their own hosting and domain.

[Download WordPress 6.8.1](#)

[Installation guide](#)

Recommend PHP 7.4 or greater and MySQL version 8.0 or MariaDB version 10.5 or greater.

Una vez descargado, lo pasamos a nuestro S3, después lo usaremos en la instancia.

miprojectonube

Objetos

Metadatos

Propiedades

Permisos

Objetos (2)



Copiar URI de S3



Copiar URL



Los objetos son las entidades fundamentales que se almacenan que otras personas obtengan acceso a sus objetos, tendrá que cc

Buscar objetos por prefijo

| Nombre

▲ | Tipo

| [HTML/](#)

Carpeta

| [wordpress-6.8.1.zip](#)

zip

Ahora vamos a crear una instancia EC2 ...

Toca coger información de la RDS sin despertarla hasta que sea necesario.

Bases de datos (2)

Filtrar por bases de datos

| Identificador de base de datos

▲ | Estado

▼ |

| [database-miprojectonube](#)

Detenc...

Y como indiqué, en la misma VPC que nuestra RDS y la misma subred.

database-miprojectonube

Identificador de base de datos	Estado	Rol	Motor
database-miprojectonube	Detención temporal	Instancia	MySQL Comm
CPU	Clase	Actividad actual	Región y AZ
-	db.t4g.micro		us-east-1a

Conectividad y seguridad Supervisión Registros y eventos Configuración Integracione

Conectividad y seguridad

Punto de enlace y puerto	Redes	Seguridad
Punto de enlace database-miprojectonube.	Zona de disponibilidad us-east-1a	Grupos de seguridad de la VPC seguridad-miprojectonube (sg-09fb4abe3bb4c71bd) Activo
Puerto 3306	VPC vpc-miprojectonube (vpc-0b3203b63d95bdaaa)	Accesible públicamente Sí

Tenemos el nombre de la VPC (que previamente le habíamos puesto nombre) y la zona (que coincide con una subred pública en nuestro caso)

Ahora creamos de EC2 con esos datos y compatible con el rol para ver S3 (ver el PDF de la amistad)

Y en la misma zona que tu RDS.

Lanzar una instancia Información

Amazon EC2 le permite crear máquinas virtuales, indican a continuación.

Nombre y etiquetas Información

Nombre

MiservidorWebProyectoNube

Una vez creada, con el rol que permite la interacción con las S3, realizamos las siguientes acciones en nuestra recién creada instancia EC2.

- Actualizar:
apt update
apt upgrade

(Suele ser útil hacerlo cuando llevas tiempo sin entrar.)

- Instalar Apache2
apt install apache2

Nota: Hay que retocar el Firewall de EC2, para que permita el tráfico web (HTML).

*Ahora instalamos el resto de las dependencias para lanzar “WordPress”, junto con nuestra RDS y servido a través de Apache en nuestra EC2.

```
apt install php libapache2-mod-php php-mysql awscli imagemagick  
php-imagick php-curl php-mbstring php-xml php-zip php-gd php-intl
```

```
root@ip-172-30-0-12:~#  
root@ip-172-30-0-12:~# apt install php libapache2-mod-php php-mysql awscli imagemagick php-imagick php-curl php-mbstring php-xml php-zip php-gd php-intl  
Reading package lists... Done  
Building dependency tree... Done  
Reading state information... Done
```

Podemos intentar conectar nuestra instancia EC2 a la RDS, ya que cuando instalemos “WordPress” este al ser llamado por Apache cuando nos conectemos con nuestro navegador, este estará por detrás interactuando con la base de datos y si no puede nuestra instancia EC2 verla, pues malo.

Y no se nos olvide instalar un cliente para conectar desde nuestra EC2 a la RDS
(que **ahora** debe estar despierta)

```
apt install default-mysql-client
```

Normalmente, en la instalaciones locales, se conecta así:

```
mysql -u administradora -p
```

En nuestro caso, la base de dato **no** es local, por lo tanto hay que poner su “**Punto de enlace**” que actúa como su URL, junto a el **-u** del usuario y **-p** (para que nos pida escribir la contraseña)

```
mysql -h database-miprojectonube.xx-1.rds.amazonaws.com -P 3306  
-u administradora -p
```

```
root@ip-172-30-0-12:/var/www/html#  
root@ip-172-30-0-12:/var/www/html# mysql -h database-miprojectonube.c-[REDACTED].rds.amazonaws.com -P 3306 -u administradora -p  
Enter password:  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MySQL connection id is 16  
Server version: 8.0.41 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)]> _
```

Si es capaz de verla, dará un mensaje parecido.

En caso contrario hay que **retocar** la seguridad de la RDS, para que permita conexiones desde la subred de nuestra EC2.

database-miprojectonube

Resumen

Identificador de base de datos
database-miprojectonube

Redes

Zona de disponibilidad
us-east-1a

VPC
vpc-miprojectonube

Seguridad

Grupos de seguridad de la VPC
seguridad-miprojectonube
(sg-09fb4abe3bb4c71bd)
 Activo

Podemos, ya que estamos conectados desde EC2 a la RDS, como cuando nos conectamos por HeidiSQL pero con un cliente en modo consola.
Probamos algunos comando ...

(ayuda)

```
MySQL [(none)]>
MySQL [(none)]> help

General information about MariaDB can be found at
http://mariadb.org

List of all client commands:
Note that all text commands must be first on line and end with ';'
?          (\?) Synonym for `help'.
charset   (\C) Switch to another charset. Might be needed for processing binlog with multi-byte charsets.
clear     (\c) Clear the current input statement.
connect   (\r) Reconnect to the server. Optional arguments are db and host.
delimiter (\d) Set statement delimiter.
edit      (\e) Edit command with $EDITOR.
ego       (\G) Send command to MariaDB server, display result vertically.
exit      (\q) Exit mysql. Same as quit.
go        (\g) Send command to MariaDB server.
help      (\h) Display this help.
nopager   (\n) Disable pager, print to stdout.
notee     (\t) Don't write into outfile.
nowarning (\w) Don't show warnings after every statement.
pager     (\P) Set PAGER [to_pager]. Print the query results via PAGER.
print    (\p) Print current command.
prompt   (\R) Change your mysql prompt.
quit     (\q) Quit mysql.
rehash   (\#) Rebuild completion hash.
sandbox  (\-) Disallow commands that access the file system (except \P without an argument and \e).
source   (\.) Execute an SQL script file. Takes a file name as an argument.
status   (\s) Get status information from the server.
system   (\!) Execute a system shell command.
tee      (\T) Set outfile [to_outfile]. Append everything into given outfile.
use     (\u) Use another database. Takes database name as argument.
warnings (\W) Show warnings after every statement.

For server side help, type 'help contents'
```

Nos da algunos comando que acepta (hay muchas más).

```
MySQL [(none)]> show databases;
+-----+
| Database           |
+-----+
| information_schema |
| miabasemiiiprojectonube |
| mysql              |
| performance_schema |
| sys                |
+-----+ MySQL [(none)]> SELECT User FROM mysql.user;
5 rows in set (0.010 sec)
+-----+
| User           |
+-----+
| administradora |
| rds_superuser_role |
| mysql.infoschema |
| mysql.session |
| mysql.sys |
| rdsadmin |
+-----+
6 rows in set (0.001 sec)

SELECT User
FROM mysql.user;

(para que muestre los usuarios
que tiene, deberá aparecer
root en vuestra caso,
yo le nombré administradora)
```

Y para salir

\q

```
MySQL [(none)]> \q  
Bye
```

Aquí, simplemente he probado que puedo ver la RDS desde mi instancia EC2 y ya de paso he curioseado por ella.

Puede hacerse todo lo que se hace con los clientes gráficos como HeidiSQL, pero con comandos. Para nosotros, ambos clientes sólo lo usamos para mirar como está, ya que interactuar directamente es otra asignatura.

Yo me conformo en el control del acceso, su conexión y que sea “WordPress” quién lo haga.

Volvamos a “WordPress” ...

Nos situamos en la carpeta donde mira Apache para instalar el zip que hemos pasado al S3.
(El S3 es como un bolsillo donde dejamos los archivos que ya usaremos)

```
cd /var/www/html/  
aws s3 cp s3://miproyectonube/wordpress-6.8.1.zip /var/www/html
```

donde tenemos el fichero descargado en el S3 al directorio donde mira Apache.

Descomprimimos el fichero

```
unzip wordpress-6.8.1.zip
```

(si no reconoce el comando, es que no está instalado **unzip**, pues se instala)
(apt install unzip)

Nos quedará una carpeta llamada `wordpress`, y alteramos los permisos ...

```
chown -R www-data:www-data wordpress/  
chmod -R 775 wordpress/
```

¡Ya está!

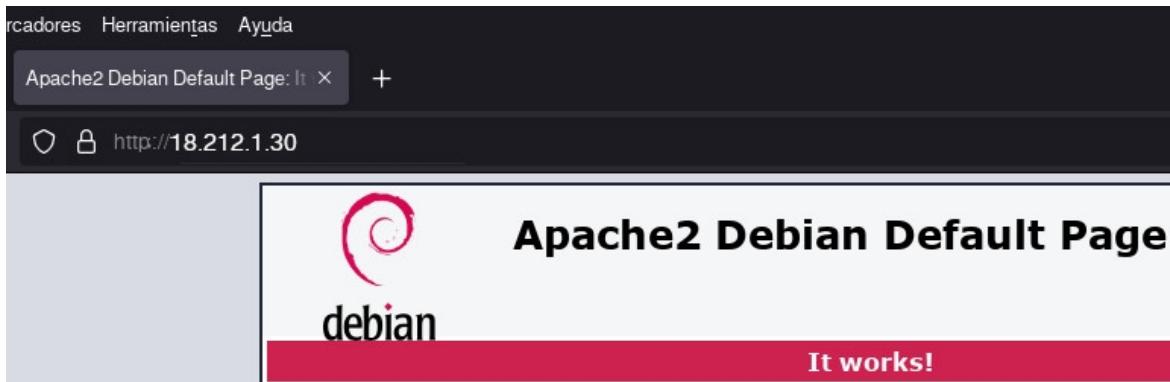
NOTA: en la ruta `/var/www/html/`
puede que tengas páginas web propias, o el “index.html”, aconsejo mover todo eso a otra carpeta

```
mv lo_que_sea /home/admin/
```

por ejemplo, para que no interactúe con “WordPress”

Ahora desde tu navegador, ponemos http:// (sin la “s”) seguido de la IP dinámica de tu EC2
http://xx.x.xxx.xx

aparece la página que esté en: “/var/www/html/index.html”



Si no has borrado tu “index.html” aparecerá la página web de bienvenida de Apache, en caso contrario, mostrará el directorio:

Index of /

Name	Last modified	Size	Description
wordpress/	2020-02-06 06:33	-	

Apache/2.4.62 (Debian) Server at 3.83.97.236 Port 80

Pinchas en el enlace, o hay un “index.html” escribes ...

http://xx.x.xxx.xx/wordpress/

A screenshot of a web browser window. The address bar shows "http://18.212.1.30/wordpress/". The page content includes a large "2.1" logo, the WordPress logo, and a sidebar with language selection options. The sidebar lists various languages including English (UK), English (Australia), English (New Zealand), English (Canada), Esperanto, Spanish of Costa Rica, Spanish of Peru, Spanish of Chile, Spanish of Venezuela, Spanish of Ecuador, Spanish of República Dominicana, Spanish of Uruguay, Spanish of Puerto Rico, Spanish of Guatemala, Spanish of México, Spanish of Argentina, Spanish of Colombia, Spanish, Estonian, Euskara, Persian (فارسی), Finnish, French of Canada, and French.

Aparecerá la página de inicio de la configuración de “WordPress”

Para aquí.
Ya sólo queda los últimos ajustes.

No configures nada todavía.
Hasta el siguiente paso.

OJO: Detiene las instancias y la RDS.