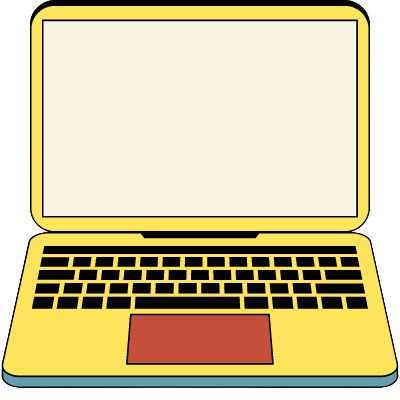
**Christian Millán Soria**

Finalizar instalación - Pila LAMP





- Curso: 2023-2024

- Clase: 2º DAW Tarde

- Profesor: David Hormigo Ramírez

- Módulo: Despliegue de Aplicaciones Web

- Fecha: 24/10/2023

**1. PHP** 3

**1.1. Instalación** 3

**1.2. Archivo default** 5

**1.3. Módulo de PHP** 7

**2. RDS** 8

**2.1. Creación** 8

**2.1.1. Método de creación** 9

**2.1.2. Motor de base de datos** 9

**2.1.3. Plantillas** 10

**2.1.4. Disponibilidad y durabilidad** 11

**2.1.5. Configuración** 11

**2.1.6. Configuración de la instancia** 12

**2.1.7. Almacenamiento** 13

**2.1.8. Conectividad** 13

**2.1.9. Autentificación de la base de datos** 14

**2.2. Conexión desde EC2** 15

**2.3. Creación del usuario “webuser”** 16

**2.4. Conexión desde máquina local** 19

**2.4.1. Connection Name** 20

**2.4.2. Connection Method** 20

**2.4.3. SSH Hostname** 21

**2.4.4. SSH Username** 21

**2.4.5. SSH Key File** 21

**2.4.6. MySQL Hostname** 21

**2.4.7. MySQL Server Port** 21

**2.4.8. Username** 22

**2.4.9. Password** 22

**2.5. Creación de la tabla “users”** 23

Esta práctica se basa en mostrar el proceso de instalación y configuración de los últimos elementos que nos faltan para terminar nuestra [**pila LAMP🔗**](https://aws.amazon.com/what-is/lamp-stack/#:~:text=A%20LAMP%20stack%20is%20a,and%20the%20programming%20language%2C%20PHP.): el lenguaje de programación en el que realizaremos las siguientes prácticas y la base de datos que alojará los elementos de nuestro entorno web.

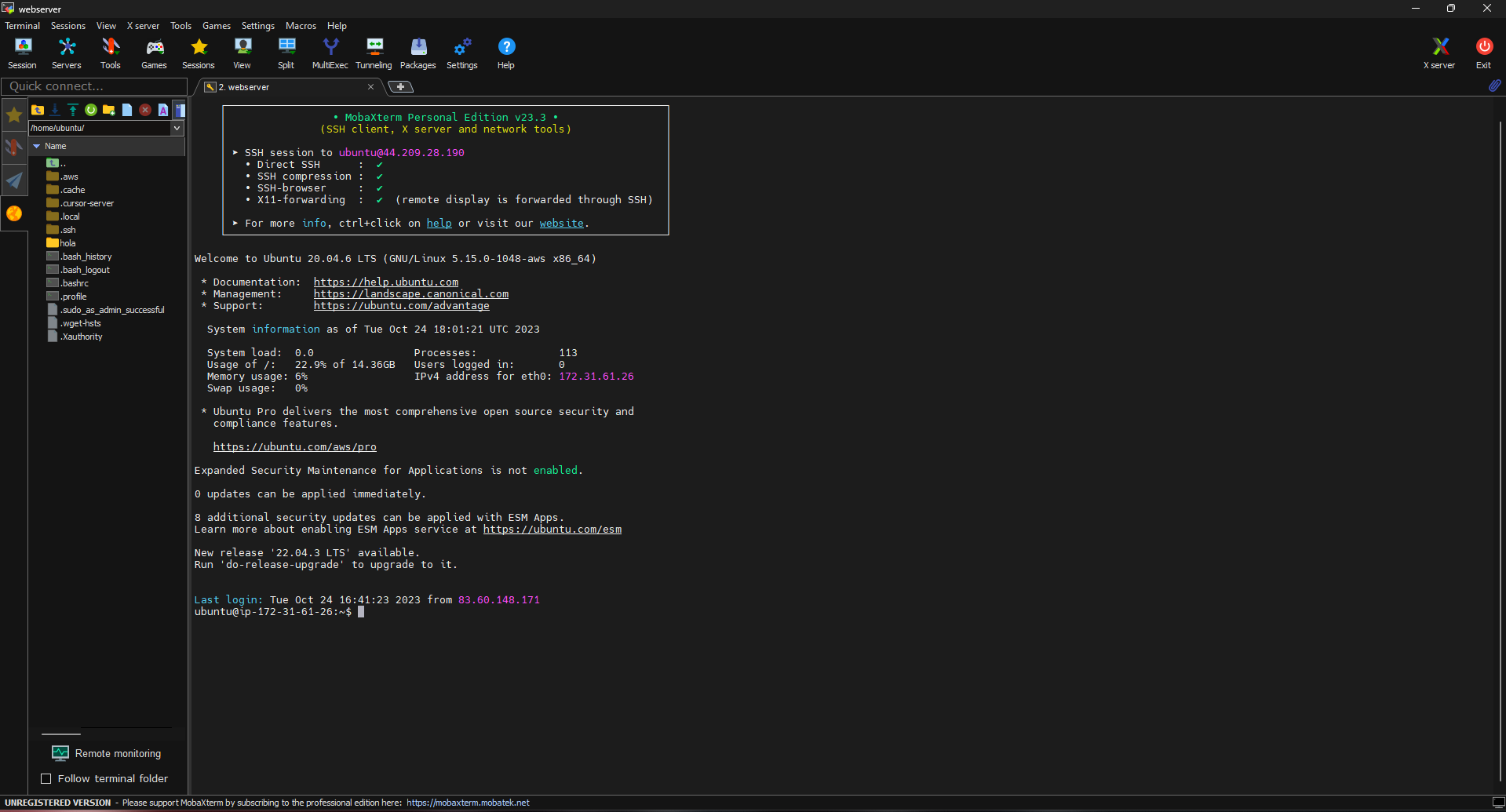
Vamos a empezar por el lenguaje de programación.

**1. PHP**

[**PHP🔗**](https://www.php.net/) es el lenguaje que utilizaremos. Es un lenguaje ampliamente utilizado en el desarrollo web y permite la creación de páginas dinámicas e interactivas.

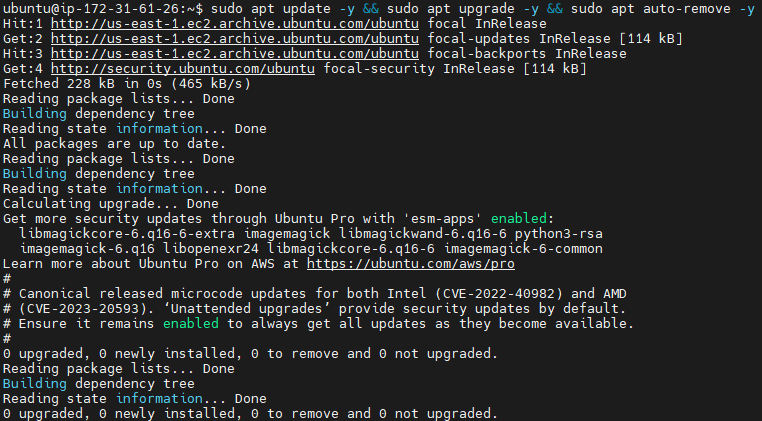
En prácticas anteriores hemos creado una instancia en AWS a la que solemos acceder mediante el protocolo SSH utilizando el par de claves que Amazon nos proporciona para nuestro usuario.

Repetimos el proceso de conexión y accedemos al terminal de comandos de nuestra máquina una vez más.



**1.1. Instalación**

El primer paso (como de costumbre; es una buena práctica) es actualizar la máquina para asegurarnos que los problemas no aparezcan por culpa de una versión menos estable: **sudo apt update -y && sudo apt upgrade -y && sudo apt auto-remove -y**.

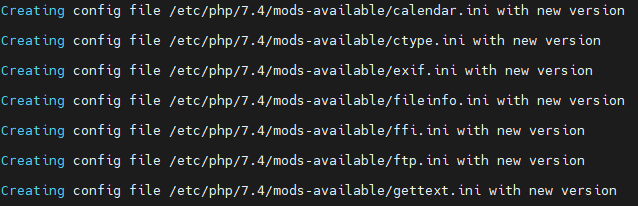


Siguiente paso: instalar PHP.

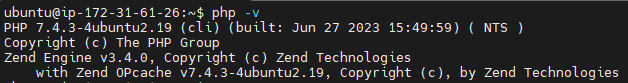
Como hemos visto anteriormente, el comando “sudo apt install …” es el que se utiliza en Ubuntu para instalar paquetes en la máquina localmente.

Esta ocasión no va a ser menos, así que utilizaremos el comando seguido del paquete que queremos obtener: [**sudo apt install php -y**🔗](https://ubuntu.com/server/docs/programming-php).

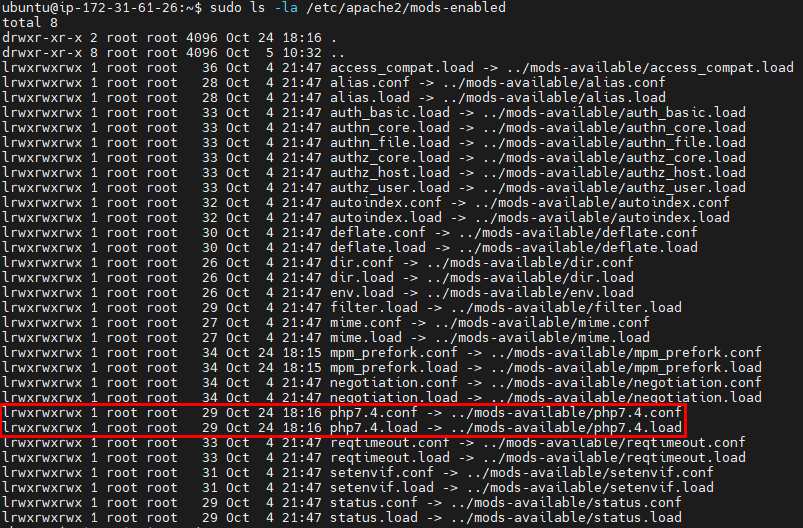
Dado que tenemos instalado Apache en nuestra máquina, se nos instalarán automáticamente los módulos necesarios para el servidor que tengan que ver con PHP, para que así este sea capaz de ejecutar el código PHP y no solo tomarlo como texto plano.



Ejecutando el comando “**php -v**” podemos comprobar que PHP se ha instalado correctamente, viendo así la versión que hemos descargado:



Mediante la utilización del siguiente comando podemos ver la lista de módulos habilitados en nuestra instancia de AWS: **sudo ls -la /etc/apache2/mods-enabled**.



**1.2. Archivo default**

Tras haber realizado el proceso de habilitación de los sitios de la [**práctica anterior🔗**](https://github.com/cmilsor245/daweb/blob/main/03-configuracion_hosts_virtuales/Configuraci%C3%B3n%20de%20Hosts%20Virtuales.pdf), si ingresamos la dirección IP pública (elástica) de nuestra instancia en la barra de búsqueda de un navegador web veremos que se nos abre uno de los archivos de dicha práctica.



Para esta práctica vamos a deshabilitar los sitios de la anterior y utilizaremos el sitio “**misitio**” como default para alojar una página web con formato PHP.

Lo primero es crear el archivo: **sudo nano /var/www/misitio/index.php**.

Introducimos el siguiente contenido:



\* No funciona si se abre la etiqueta de PHP con “**<?**” únicamente.

[“**phpinfo()**”🔗](https://www.php.net/manual/en/function.phpinfo.php) es una función de PHP que genera un contenido de información sobre el sistema en el que está instalado PHP.

Guardamos el archivo y salimos.

Vamos a proceder a deshabilitar los sitios mencionados anteriormente para poder ingresar únicamente la dirección IP pública de nuestra máquina en un navegador y que se muestre directamente la página de información de PHP:

**sudo a2dissite 404.conf && sudo a2dissite christianmillan.conf && sudo a2dissite turing.conf**

Recargamos Apache para aplicar los cambios: **sudo systemctl reload apache2**.

Ahora, al ingresar únicamente la dirección IP pública del servidor en un navegadore web, se muestra de nuevo la página con la que trabajamos en la práctica del sitio “misitio”.



Por defecto, Apache realiza una búsqueda de un archivo con extensión “**.html**” en el directorio establecido como predeterminado, y si no lo encuentra terminaría encontrando el archivo con extensión “**.php**” que acabamos de crear. El objetivo no es eliminar el archivo HTML, sino configurar Apache para que busque primero el archivo PHP, aun estando el archivo HTML en el mismo directorio.

Para hacer esto, utilizaremos el comando “**sudo nano /etc/apache2/mods-enabled/dir.conf**”. Este es el archivo de configuración que Apache utiliza para realizar dicha búsqueda.

El archivo viene de forma nativa de la siguiente manera:



Como se puede ver, el archivo HTML aparece el primero en la lista, por lo que es el primero que Apache intenta buscar.

Lo que vamos a hacer es cambiar el orden para que el archivo PHP aparezca el primero, eliminándolo de la cuarta posición y colocándolo en la primera:



Recargamos Apache: **sudo systemctl reload apache2**.

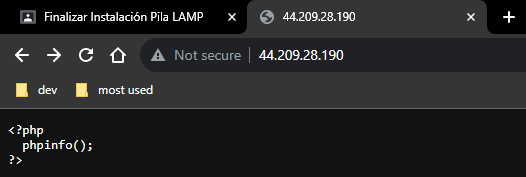
Con esto hemos hecho que Apache busque primero un archivo PHP, y no el HTML, por lo que ahora al recargar la página web podemos ver que el contenido ha cambiado al de dicho archivo.



**1.3. Módulo de PHP**

A modo de prueba, podemos deshabilitar el módulo principal de PHP en Apache para comprobar que, con este mismo archivo pero sin tener el módulo habilitado, la función no se ejecuta, sino que se toma como texto plano: **sudo a2dismod php7.4**.

Recargamos Apache y la página web.

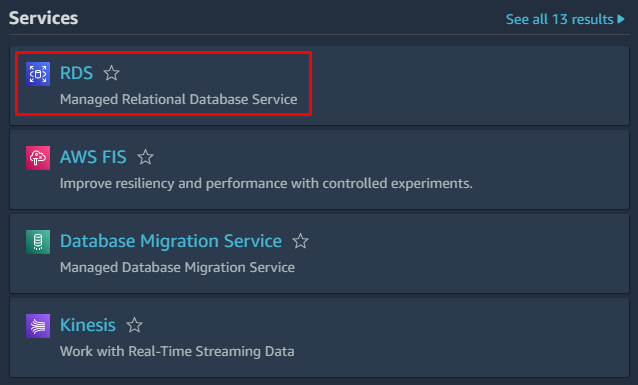


Para volverlo a habilitar: **sudo a2enmod php7.4**.

**2. RDS**

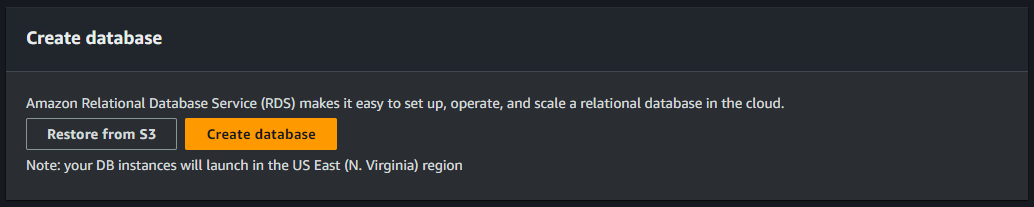
La segunda parte de esta práctica consiste en crear una base de datos mediante la utilización de [“**RDS**”🔗](https://aws.amazon.com/rds/?p=ft&c=db&z=3), un servicio que ofrece AWS para poder utilizar distintos motores de bases de datos de forma privada dentro de la propia red de AWS, sin estar expuesta a cualquier ataque externo.

Realizamos la búsqueda de “**rds**” en la barra de búsqueda del panel de control principal de nuestro laboratorio de AWS y entramos en su apartado.



Se nos presenta, como de costumbre, un menú donde podremos realizar distintas acciones relacionadas con RDS.

Hacemos clic en el botón “**Create database**”, en naranja.



**2.1. Creación**

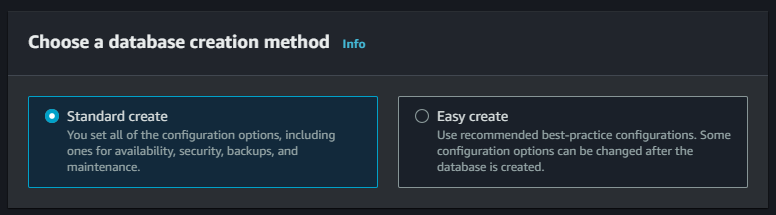
Aparece un menú de creación donde estableceremos la configuración básica/principal de la nueva base de datos para tenerla disponible para trabajar.

A continuación se mostrará cada apartado de la creación donde tendremos que cambiar alguna configuración para adaptar la base de datos a nuestras necesidades.

Si algún apartado no aparece, es que no se ha alterado de su forma predeterminada.

**2.1.1. Método de creación**

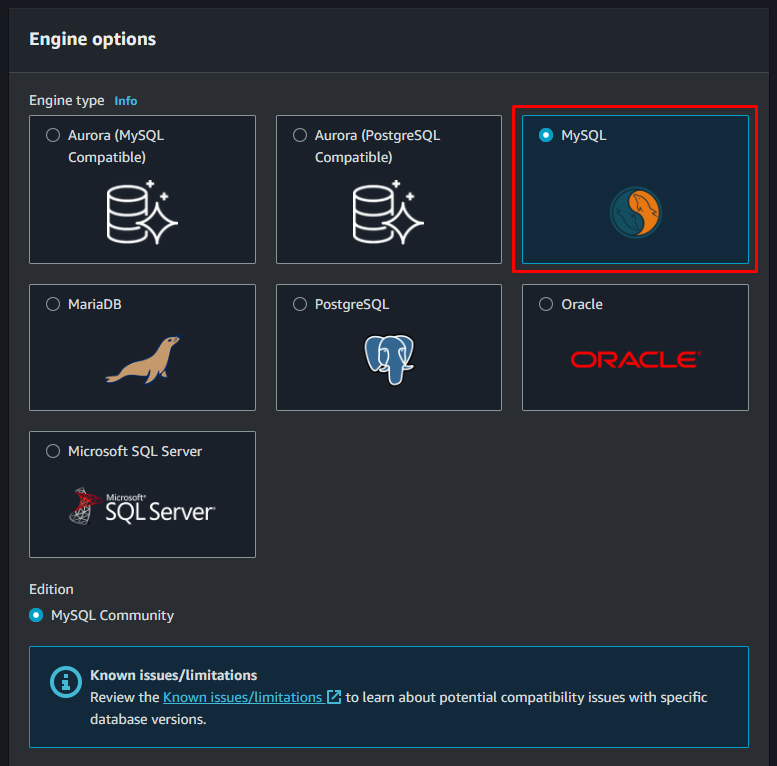
Elegiremos la opción de creación estándar, ya que nos interesa cambiar algunos apartados específicos.



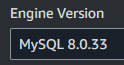
**2.1.2. Motor de base de datos**

Seleccionamos el motor de base de datos [“**MySQL**”🔗](https://www.mysql.com/), el cual es uno de los más usados en todo el mundo debido a su compatibilidad y estabilidad.

Dejamos la edición en la de comunidad.

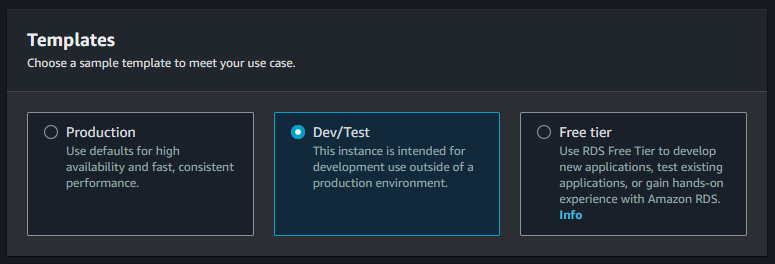


Escogeremos la versión [“**MySQL 8.0.33**”🔗](https://dev.mysql.com/doc/relnotes/mysql/8.0/en/news-8-0-33.html), que nos proporcionará una estabilidad confiable y un número bajo de bugs, tema del que ya hablamos en el apartado de selección de la imagen para la nueva instancia en la [**primera práctica🔗**](https://github.com/cmilsor245/daweb/blob/main/01-servidor_ubuntu_aws/Instalaci%C3%B3n%20de%20un%20servidor%20Ubuntu%20en%20AWS.pdf) del módulo.



**2.1.3. Plantillas**

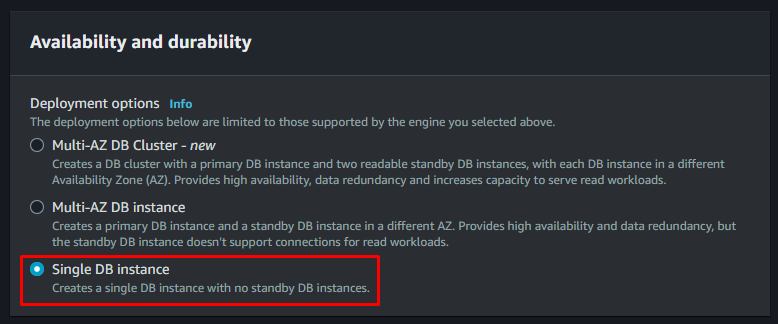
Elegimos la plantilla para desarrollo y pruebas, ya que tendrá una potencia adecuada para las prácticas que realizaremos con la base de datos.



**2.1.4. Disponibilidad y durabilidad**

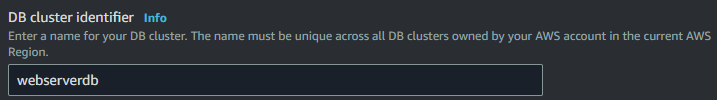
Es importante que escojamos un plan de disponibilidad de una única instancia de base de datos.

Esto es debido a que el saldo de nuestra cuenta de AWS es limitado (**$100**), y un plan de más instancias puede llegar a consumir demasiado y acabar con nuestro crédito muy rápidamente.

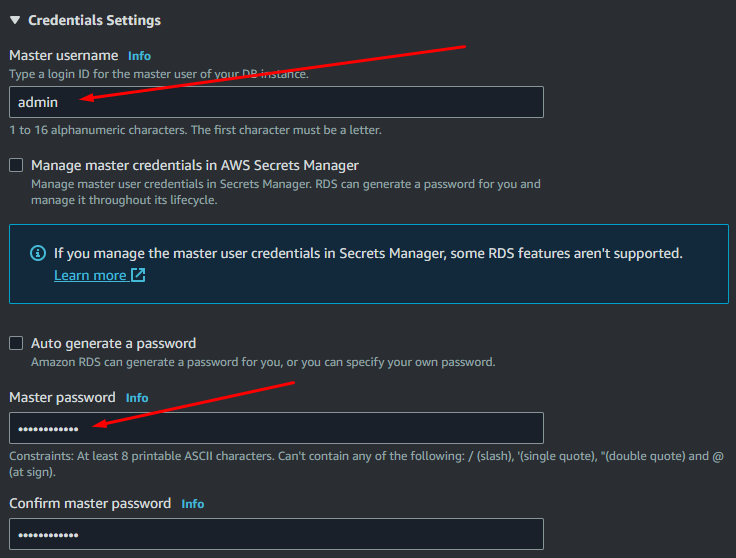


**2.1.5. Configuración**

Le damos un nombre a la nueva base de datos. Este nombre debe ser único entre el resto de bases de datos de nuestra cuenta de AWS.



Dejamos el nombre del usuario administrador por defecto en “**admin**” y establecemos una contraseña para este usuario en el apartado para la misma.

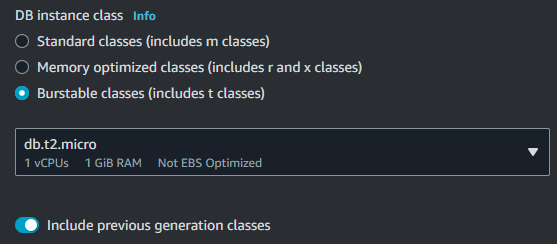


Otra forma de acceder sería activar la opción para generar la contraseña de forma aleatoria y automática “**Auto generate a password**”, aunque de esta forma la única vez que podemos visualizar dicha contraseña es mientras la base de datos se esta creando cuando hagamos clic en el botón para ello.

**2.1.6. Configuración de la instancia**

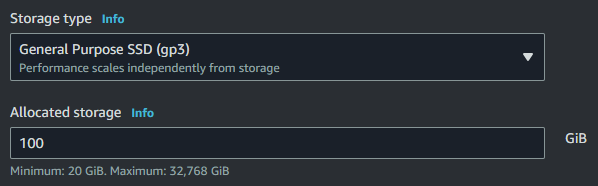
En este apartado, primero marcamos la opción “**Include previous generation classes**”, lo que nos permitirá seleccionar el tipo de instancia antiguo, que es uno de los que tenemos disponibles en el plan de nuestra cuenta de AWS.

Seleccionamos la tercera opción en la clase de instancia para optar a los tipos “**t**” y elegimos el tipo “**db.t2.micro**” en el desplegable.



**2.1.7. Almacenamiento**

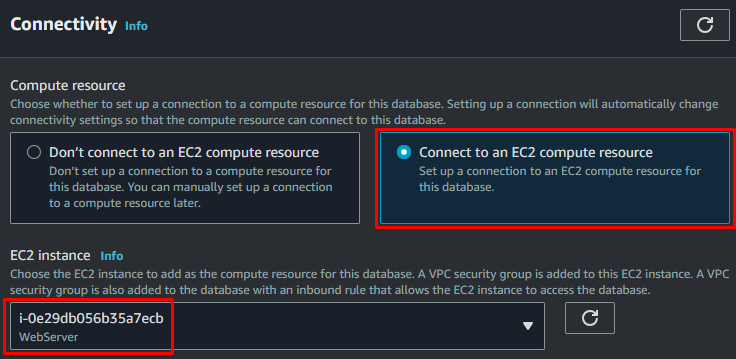
Cambiamos el tamaño por defecto de 200 GiB a **100 GiB.**



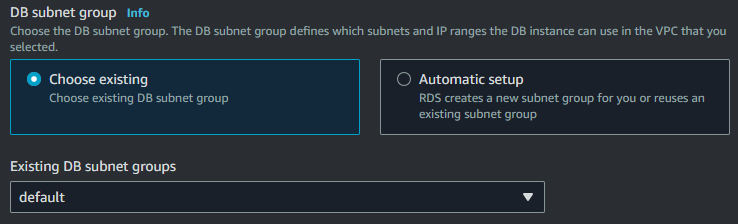
**2.1.8. Conectividad**

Aquí es donde establecemos que la nueva base de datos se conectará directamente con una instancia de nuestra cuenta de AWS, ya que estará integrada en la pila LAMP que queremos terminar de instalar en esta práctica.

Seleccionamos la opción para ello y la instancia a la que queremos que se conecte en el desplegable que aparece:

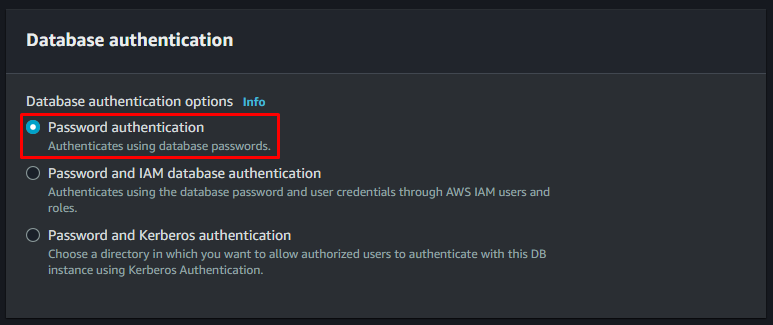


Además, elegimos que el grupo de seguridad sea uno existente para no crear uno nuevo automáticamente.

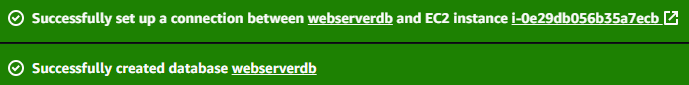


**2.1.9. Autenticación de la base de datos**

Dejamos la opción por defecto para realizar las conexiones mediante la utilización de la contraseña que escogimos anteriormente.



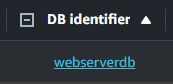
Por último, bajamos hasta abajo y hacemos clic en el botón “**Create database**”, en naranja.



**2.2. Conexión desde EC2**

Lo siguiente es establecer la conexión desde dentro de la red privada proporcionada por AWS.

Una vez se ha creado la base de datos por completo, entramos al resumen de la misma haciendo clic en su ID en el listado de bases de datos.



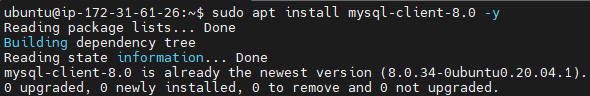
Aquí, al igual que con las instancias EC2, podemos ver una serie de elementos de la configuración de nuestra nueva base de datos.

En la parte izquierda, vemos un apartado llamado “**Endpoint & port**”. Este se refiere al hostname de nuestra base de datos, que vendría a ser similar a la dirección IP pública de una instancia EC2.

También podemos ver que el puerto que se utiliza por defecto es el **3306**.

Vamos a dirigirnos a la conexión por SSH con nuestra instancia de nuevo, al terminal de comandos.

Aquí ejecutaremos el comando “**sudo apt install mysql-client-8.0 -y**” para instalar el cliente necesario para realizar la conexión con la base de datos.



MySQL tiene una arquitectura de tipo [**cliente-servidor🔗**](https://www.techtarget.com/searchnetworking/definition/client-server#:~:text=What%20is%20client%2Dserver%3F,computing%20model%20used%20by%20mainframes.), lo que quiere decir que la base de datos se aloja en un servidor y para acceder a ella hace falta un cliente.

Una vez instalado con éxito, podemos pasar a ejecutar el siguiente comando:

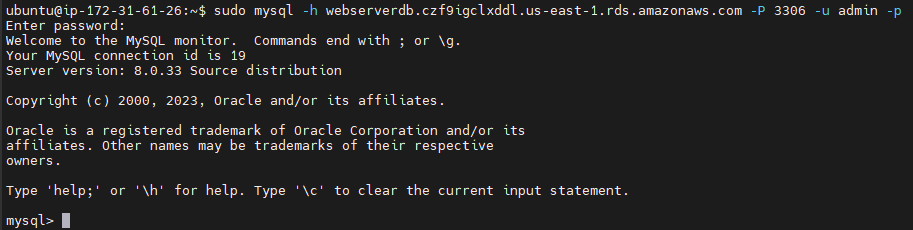
[**sudo mysql -h {host desde el resumen de la database en aws} -P 3306 -u admin -p🔗**](https://explainshell.com/explain?cmd=sudo+mysql+-h+-P+3306+-u+admin+-p)

\* Hacer clic en el comando para una explicación más detallada del mismo.

Al instalar el cliente se nos ha habilitado la opción de ejecutar comandos “**mysql**”.

Con este comando especificamos, primero el host al que nos queremos conectar, seguido del puerto que utilizará la conexión, el usuario con el que se llevará a cabo y por último solicitamos que se pida la contraseña del usuario para acceder a la base de datos.

En mi caso, el host es “**webserverdb.czf9igclxddl.us-east-1.rds.amazonaws.com**”.



**2.3. Creación del usuario “webuser”**

Como hemos visto, la conexión se ha realizado con el usuario “**admin**”, creado en el apartado de creación/configuración de la base de datos en AWS.

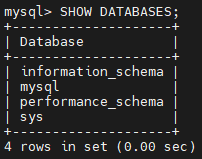
Esto es poco recomendable, ya que este usuario cuenta con demasiados permisos (todos ellos) sobre la base de datos completa: puede hacer absolutamente cualquier cosa con ella.

Vamos a proceder a crear un usuario llamado “**webuser**”, el cual contará únicamente con unos permisos concretos para que no pueda realizar cambios críticos en la base de datos.

Lo primero es crearlo, lo cual haremos con el comando “**CREATE USER 'webuser'@'%' IDENTIFIED BY 'webuser\_password';**”. [**Este comando**🔗](https://www.hostinger.com/tutorials/mysql/how-create-mysql-user-and-grant-permissions-command-line)se encarga de crear el nuevo usuario en el servidor localhost y establece su contraseña a “**webuser\_password**”.

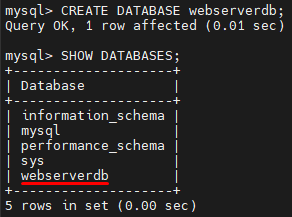


Como se puede comprobar con el comando “**SHOW DATABASES;**”, ahora mismo no contamos con una base de datos propia donde realizar prácticas/ejercicios, sino que existen una serie de base de datos que la creación del servidor RDS ha generado.



Lo que debemos hacer ahora es crear una base de datos para poder trabajar con ella y comprobar que la asignación de los permisos para el nuevo usuario “webuser” se ha ejecutado con éxito.

Utilizaremos el siguiente comando: **CREATE DATABASE webserverdb;**.



El siguiente paso es actualizar los permisos del mismo para que cumpla con lo establecido en la documentación de esta práctica.

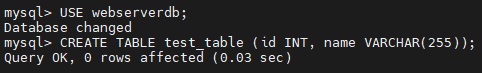


Utilizando el comando “**GRANT SELECT, INSERT, DELETE, UPDATE ON webserverdb.\* TO 'webuser'@'%';**” establecemos una lista de permisos específicos para nuestro nuevo usuario en la base de datos que hemos creado con anterioridad.

Actualizamos los permisos con “**FLUSH PRIVILEGES;**”.

A modo de prueba, vamos a crear una tabla con el usuario administrador desde este terminal de comandos y probaremos a realizar un SELECT con el nuevo usuario y a crear otra tabla con él, para asegurarnos de que puede realizar únicamente las acciones que le hemos concedido.

Utilizamos el comando “**USE webserverdb;**” y “**CREATE TABLE test\_table (id INT, name VARCHAR(255));**” con el usuario administrador para crear una nueva tabla.



Insertamos algunos datos aleatorios:

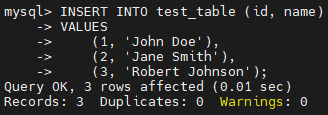
**INSERT INTO test\_table (id, name)**

**VALUES**

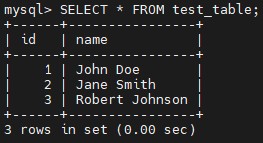
**(1, 'John Doe'),**

**(2, 'Jane Smith'),**

**(3, 'Robert Johnson');**



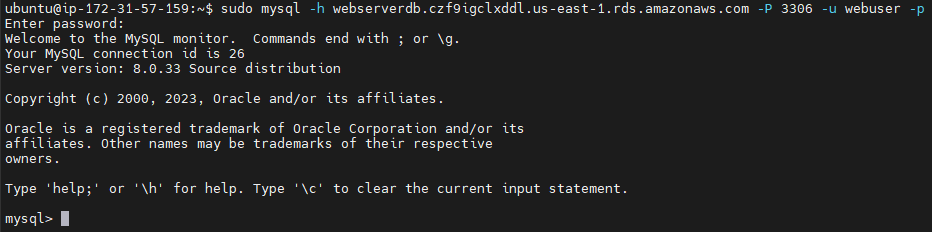
Comprobamos que los datos existen desde el usuario “admin”, que estamos seguros de que puede realizar cualquier acción en la base de datos.



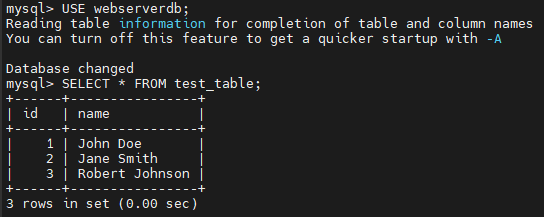
Ahora es turno de probar los permisos del nuevo usuario. Ejecutamos el comando “**exit**” para salir de la conexión con el servidor de base de datos.

Volvemos a establecer conexión con él, pero esta vez haciendo uso del login con el nuevo usuario:

**sudo mysql -h webserverdb.czf9igclxddl.us-east-1.rds.amazonaws.com -P 3306 -u webuser -p**



Probamos a hacer un simple **SELECT**, que entra en la lista de permisos de la práctica.



La petición se ejecuta sin problemas, pero, en cambio, si probamos, por ejemplo, a crear una nueva tabla con este usuario, se nos genera un error, ya que no cuenta con los permisos suficientes:



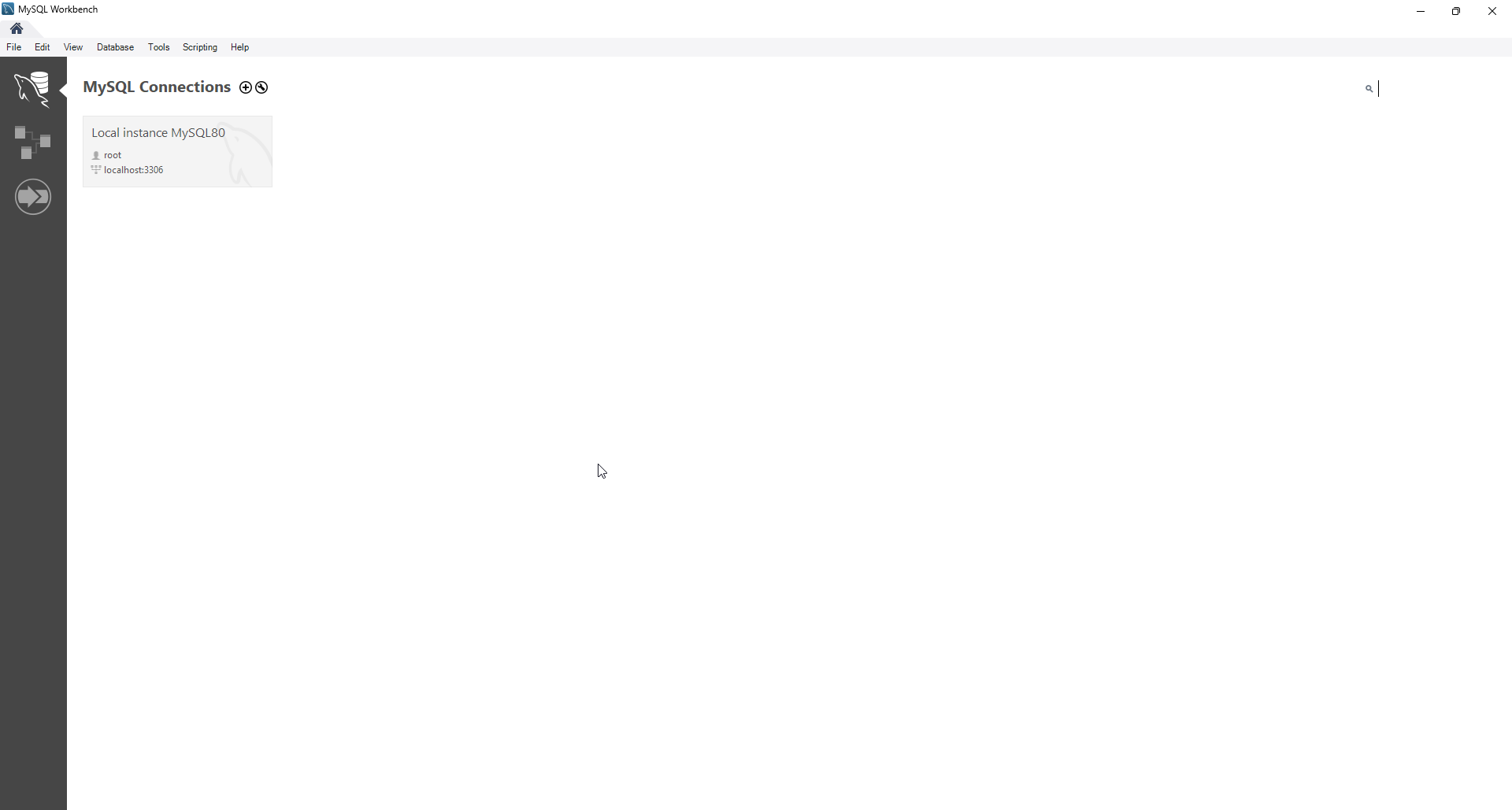
Esto ha dejado por comprobado que los permisos se han establecido correctamente y el nuevo usuario no puede realizar ninguna acción que no se le haya especificado previamente.

**2.4. Conexión desde máquina local**

El último paso de la tarea consiste en realizar una conexión similar a la anterior hecha desde EC2, pero esta vez desde la máquina local del alumno, mediante el uso de [“**MySQL Workbench**”🔗](https://www.mysql.com/products/workbench/).

\* Se presupone que MySQL Workbench ya está instalado.

Una vez entramos al programa, se nos presenta un home como el siguiente:



Haremos clic en el botón “**+**” al lado del título “**MySQL Connections**”.



Se abre un cuadro para realizar la conexión, un proceso similar al de crear conexiones en programas como PuTTy o MobaXterm, visto en [**anteriores prácticas🔗**](https://github.com/cmilsor245/daweb/blob/main/01-servidor_ubuntu_aws/Instalaci%C3%B3n%20de%20un%20servidor%20Ubuntu%20en%20AWS.pdf).

Seguiremos los siguientes pasos para rellenar cada campo necesario.

Esta conexión se va a realizar mediante el uso del protocolo SSH, ya que es la única forma que tenemos de acceder a nuestra instancia de AWS y, mediante ella, a la RDS nueva, ya que como vimos anteriormente, la red de las RDS de AWS es privada, por lo que es inaccesible desde el exterior (por seguridad), aunque un método es el siguiente:

**2.4.1. Connection Name**

Establecemos un nombre para la nueva conexión de MySQL Workbench que vamos a crear, así será accesible más adelante sin necesidad de repetir este método de nuevo.



**2.4.2. Connection Method**

Es importante que escojamos la opción [“**Standard TCP/IP over SSH**”🔗](http://download.nust.na/pub6/mysql/doc/workbench/en/wb-manage-db-connections-ssh.html#:~:text=The%20connection%20type%20allows%20MySQL,IP%20over%20an%20SSH%20tunnel.&text=In%20addition%20to%20a%20number,a%20number%20of%20specialized%20parameters.) en el desplegable que se nos presenta para este campo, ya que así utilizaremos de “**puente**” el protocolo SSH para realizar la conexión con el servidor de bases de datos estando en un equipo externo a la red privada de la misma.



**2.4.3. SSH Hostname**

Ingresamos la dirección IP pública de nuestra instancia en EC2/AWS.



**2.4.4. SSH Username**

Utilizaremos el usuario “**ubuntu**”, el mismo que utilizamos para las conexiones por SSH normales.



**2.4.5. SSH Key File**

Seleccionamos el archivo de **vockey** de nuestra cuenta de AWS que hemos descargado en anteriores prácticas. Necesitamos el archivo con extensión “**.pem**”.



**2.4.6. MySQL Hostname**

En este campo debemos introducir el punto de enlace que obtuvimos anteriormente desde el resumen de la nueva RDS creada (hace las veces de hostname de una instancia, pero para un servidor de bases de datos).



**2.4.7. MySQL Server Port**

Lo dejaremos en el predeterminado: **3306**, ya que es el que está utilizando nuestro servidor RDS.

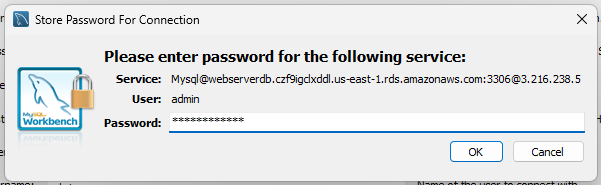
**2.4.8. Username**

Escogemos el usuario “**admin**”, que es el usuario administrador de nuestro servidor, ya que lo necesitaremos para crear una tabla más adelante.

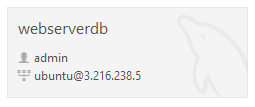


**2.4.9. Password**

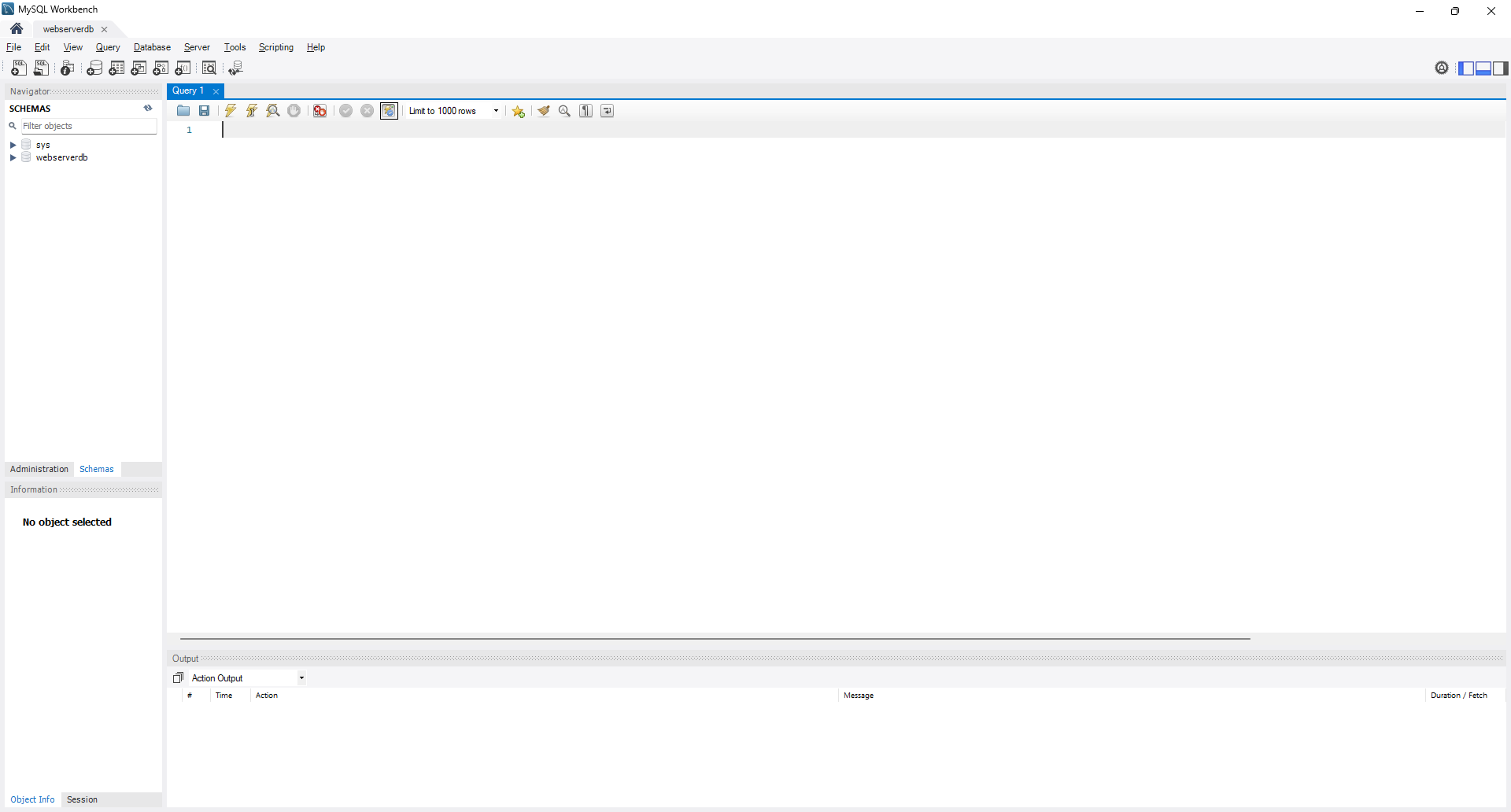
Clicamos en el botón “**Store in Vault…**” para escribir la contraseña que configuramos para el usuario administrador del RDS en el proceso de creación de la misma.



Hacemos clic en “**OK**” y se nos muestra el nuevo perfil de conexión en el panel de control de MySQL Workbench.



Hacemos clic en él (puede que se muestre un error de **fingerprint**, pero enseguida se realiza la conexión sin problema) y entramos al editor una vez la conexión se completa.



**2.5. Creación de la tabla “users”**

Para terminar, crearemos la tabla “**users**” desde el editor de MySQL Workbench con los campos apropiados.

Utilizaremos el siguiente comando:

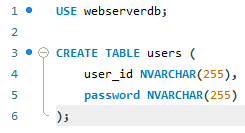
**CREATE TABLE users (**

**user\_id NVARCHAR(255),**

**password NVARCHAR(255)**

**);**

El tipo de dato de cada campo es el especificado en la documentación de la práctica.



Si hacemos un “**SELECT \* FROM users;**” podemos ver cómo se ha creado la tabla con los dos campos necesarios:

