

PROYECTO FINAL DE BASE DE DATOS

1.Introducción

La realización del presente proyecto **es voluntaria** y permitirá reforzar los conocimientos adquiridos durante el curso, así como subir la nota final del módulo (hasta 1 punto), siempre y cuando el módulo esté previamente superado (se han superado todos los resultados de aprendizaje asociados al módulo)

2. Base de datos

En este proyecto se desarrollarán 2 bases de datos (MySQL y MongoDB) y 1 API REST para interactuar con la base de datos MySQL. Respecto a las bases de datos se deben seguir las siguientes directrices:

- Las bases de datos tendrán un mínimo de 3 tablas/colecciones
- Debe identificarse al menos una operación de tipo transaccional (se necesita realizar operaciones INSERT, UPDATE, DELETE sobre varias tablas/colecciones de forma atómica)
- Todos los identificadores serán autonuméricos en la base de datos MySQL
- Se definirán todas las claves únicas que sean necesarias para que no se permitan duplicados en determinadas columnas/atributos
- En el script de creación de la base de datos MongoDB se definirá un esquema de validación para cada colección, donde se indiquen los atributos y tipos de datos.

El sistema de información puede elegirse libremente. No obstante, se adjuntan a continuación ideas:

Sistema de información	Descripción corta	Transacción clave
Marketplace de segunda mano	Tienda online de compraventa de productos entre particulares	Los productos tienen un campo estado (disponible, vendido) que se actualiza cuando se crea una operación de compraventa
Red social con monedas virtuales	Red social donde se premian los likes en las publicaciones con monedas virtuales	Se dispone de un acumulador de likes en la tabla/colección de usuarios. Cuando este acumulador pasa a un nuevo millar (de 999 a 1000, de 19999 a 20000...) se incrementa en 1 las monedas del usuario. Cuando este acumulador baja de millar (de 1000 a 999, de 20000 a 19999...) se decrementa en 1 las monedas del usuario

Gestión de torneos de deportes	Gestor de torneos de algún tipo de deporte, por ejemplo: fútbol o baloncesto. Se almacenarán equipos, partidos y jugadores.	Los resultados de los partidos incrementan puntos en los equipos cuando se empata o se gana
Préstamo de libros	Sistema de gestión de préstamos de libros	Los libros tienen un campo estado (disponible, prestado) que se actualiza cuando se crea un préstamo
Reservas de eventos	Plataforma para reservar entradas de museos, conciertos...	Cuando se crea una reserva se descuenta del aforo del evento correspondiente las entradas reservadas

3. Formato de la memoria

Se entregará una memoria realizada mediante Google Docs o Microsoft Word con el siguiente formato:

Tipo de letra: Arial o similar de 12 puntos.
Interlineado: 1.15
Alineación: justificación completa.
Se incluirá un espacio después de cada párrafo.
Todas las páginas menos la portada deben ir numeradas.

4. Contenidos de la memoria

Los apartados a incluir en la memoria serán los siguientes:

PORTADA

ÍNDICE

1.INTRODUCCIÓN

Describir brevemente el sistema de información del proyecto, indicando de forma general los datos que se van a almacenar. Hay que indicar que se implementan dos soluciones alternativas (mysql y mongodb).

2. TECNOLOGÍAS

2.1. MySQL

Incluye una descripción breve del sistema gestor de base de datos MySQL, sus características principales y su logo

2.2. MongoDB

Incluye una descripción breve del sistema gestor de base de datos MongoDB, sus características principales y su logo

3. DISEÑO DE LA BASE DE DATOS

3.1. Modelo de datos MySQL

Incluye un diagrama de base de datos relacional con notación Crow's Foot

3.2. Modelo de datos MongoDB

Incluye un diagrama de base de datos MongoDB con notación Crow's Foot

4. DESARROLLO DE LA BASE DE DATOS

4.1. Script SQL

Incluye un script SQL que cree la base de datos, las tablas, las claves ajenas y que al menos inserte 5 filas en cada una de las tablas

4.2. Consultas SQL

Añade 5 enunciados de consultas variadas (una tabla, varias tablas, group by, subconsultas) y sus soluciones

4.3. Script MongoDB

Incluye un script para ejecutar desde la consola de MongoDB Compass que cree la base de datos (comando use), cree cada colección con un esquema de validación (db.createCollection) y que al menos inserte 5 filas en cada una de las colecciones

Para el desarrollo del script de MongoDB utiliza como referencia el script proporcionado en la tarea: A7. MONGODB – Consultas

4.4. Consultas MongoDB

Añade 5 enunciados de consultas variadas y sus soluciones

5. API REST

5.1. API REST sobre MySQL en local

Desarrolla una api REST para realizar operaciones CRUD sobre las diferentes tablas de la base de datos MySQL.

Crea por cada tabla de la base de datos un video donde se vea que puedes recuperar y almacenar datos utilizando la api REST correspondiente. Utiliza el complemento Thunder Client de VS, realiza pruebas de GET, POST, PUT y DELETE.

Sube los videos a Internet (por ejemplo, Google Drive) y genera enlaces públicos.

Añade en este apartado los enlaces.

Para realizar este apartado b  sate en los 4 ficheros adjuntos en Aules que implementan una API Rest sin utilizar ning  n framework:

- config.php (sirve el fichero que ya ten  as de pr  cticas anteriores)
- cliente-pdo.php (este fichero ha sufrido cambios)
- api.utils.php (este fichero es nuevo)
- api/usuarios.php (este fichero es nuevo, ponlo dentro de una carpeta api)

Se trata de un c  digo muy b  sico y que tiene gran margen de mejora (por ejemplo, a  adir validaciones, separar el c  digo en varios ficheros: modelo, controlador, enrutador...)

ACCIONES:

1.Crea una tabla usuarios con la misma estructura que la mostrada a continuaci  n:

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G
id_usuario	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>
nombre	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
apellidos	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
email	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
password	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
rol	ENUM('admin', 'user')	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

2. Ajusta la configuraci  n de Nginx (sudo nano /etc/nginx/sites-available/default) para que reescriba este tipo de URLs:

- `http://localhost/api/usuarios ==> http://localhost/api/usuarios.php`
- `http://localhost/api/usuarios/5 ==> http://localhost/api/usuarios.php?id=5`

```

josemaria@josemaria-VirtualBox: /etc/nginx/sites-available
Archivo Editar Ver Buscar Terminal Ayuda
GNU nano 6.2 default *

# Maneja la ruta /api/usuarios
location /api/usuarios {
    rewrite ^/api/usuarios$ /api/usuarios.php last;
}

# Maneja la ruta /api/usuarios/{id}
location /api/usuarios/ {
    rewrite ^/api/usuarios/([0-9]+)$ /api/usuarios.php?id=$1 last;
}

# pass PHP scripts to FastCGI server
#
location ~ \.php$ {
    include snippets/fastcgi-php.conf;

```

Para que se aplique la configuración reinicia Nginx: **sudo service nginx restart**

3. Ajusta en el plugin “Open PHP/HTML/JS In Browser” la ruta raíz donde se encuentran las páginas web. Con ello se consigue que al darle a abrir una página (por ejemplo usuarios.php que está en la carpeta api) se genere una url de este tipo: `http://direccion_ip/api/usuarios.php`

Open-PHP-HTML-JS-in-browser: Document Root Folder

Base directory of your pages to serve from `http://localhost` domain (eg. `C:\xampp\htdocs\`, `/var/www/`, etc.). Autodetect by default

Open-PHP-HTML-JS-in-browser: Alternative Document Root Folders

Alternative base directories of your pages to serve from `http://localhost` domain

[Add Item](#)

Open-PHP-HTML-JS-in-browser: Url To Open

Url scheme to open in browser (`http://localhost` or `file:///` or custom)

Open-PHP-HTML-JS-in-browser: Custom Host

Custom localhost hostname with port number (eg. `localhost:8888`, `localhost:1234`)

Open-PHP-HTML-JS-in-browser: Custom Url To Open

Custom url to open in browser (eg. `http://${host}/${relativeDirnameDocumentRoot}/${fileBasename}`)

4. Comprueba que el API Rest permite hacer operaciones CRUD sobre la tabla de usuarios.

5. Dentro de la carpeta api y utilizando usuarios.php como plantilla crea un archivo php (api rest) para cada una de las tablas de tu base de datos.

5.2. API REST sobre MySQL en AWS

Desarrolla una api REST en AWS para realizar operaciones CRUD sobre las diferentes tablas de una base de datos MySQL también ubicada en la nube AWS.

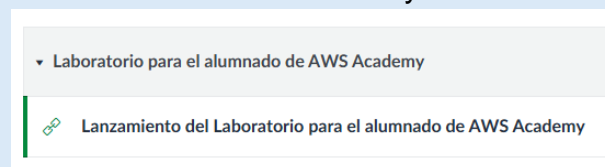
Crea por cada tabla de la base de datos un video donde se vea que puedes recuperar y almacenar datos utilizando la api REST correspondiente. Utiliza el complemento Thunder Client de VS, realiza pruebas de GET, POST, PUT y DELETE.

Sube los videos a Internet (por ejemplo, Google Drive) y genera enlaces públicos.

Añade en este apartado los enlaces.

Para crear la máquina virtual en AWS:

1. Entra en AWS ACADEMY con el acceso enviado por el profesor vía email
2. En el curso Learner Lab entra a Módulos → Lanzamiento del Laboratorio para el alumnado de AWS Academy



3. Realiza el tutorial “Amazon_EC2_Linux.pdf” (el paso 18 y los siguientes ignóralos) teniendo en cuenta el siguiente cambio: ELIGE EL SISTEMA OPERATIVO UBUNTU

4. Abre una consola y accede a la carpeta donde está descargado el fichero labsuser.pem

5. Conéctate por ssh a la máquina virtual Ubuntu mediante este comando.
`ssh -i labsuser.pem ubuntu@<IP_elástica>`

Para instalar NGINX en AWS:

- 1.- Ejecuta los siguientes comandos:

```
sudo apt update
sudo apt install nginx
sudo apt install php-fpm
```

2.- Comprueba en la ruta /var/run/php/ el archivo .sock disponible. En mi caso dispongo del archivo php8.3-fpm.sock

```
ubuntu@ip-172-31-32-6:~$ cd /var/run/php
ubuntu@ip-172-31-32-6:/var/run/php$ ls
php-fpm.sock  php8.3-fpm.pid  php8.3-fpm.sock
ubuntu@ip-172-31-32-6:/var/run/php$ |
```

3.- Modifica el archivo de configuración de NGINX (sudo nano/etc/nginx/sites-available/default) realizando los siguientes cambios:

-Añade en la lista index.php

-Quita 4 almohadillas y pon la versión correcta del archivo .sock

```
root /var/www/html;

# Add index.php to the list if you are using PHP
index index.html index.htm index.nginx-debian.html index.php;

server_name _;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    try_files $uri $uri/ =404;
}

# pass PHP scripts to FastCGI server
#
location ~ /\.php$ {
    include snippets/fastcgi-php.conf;
    #
    # With php-fpm (or other unix sockets):
    fastcgi_pass unix:/run/php/php8.3-fpm.sock;
    # With php-cgi (or other tcp sockets):
    fastcgi_pass 127.0.0.1:9000;
}
```

4.- Instala el soporte para acceso a bases de datos MySQL:

sudo apt install php-mysql

5.- Configura el servicio NGINX para que arranque automáticamente:

sudo systemctl enable nginx

6.- Reinicia el servicio NGINX y comprueba el estado del servicio:

sudo service nginx restart

sudo service nginx status

7.- Crea un fichero index.html sencillo (por ejemplo un simple h1) en /var/www/html y comprueba que a través de la IP elástica puedes acceder a la web

← → ↻ ⚠ No es seguro http://54.226.214.249

hola

Para instalar MySQL en AWS:

1.- Ejecuta los siguientes comandos:

```
sudo apt update
```

```
sudo apt install mysql-server
```

```
sudo mysql
```

TEN ESPECIAL CUIDADO CON ESTE COMANDO. HAY UN ; AL FINAL

```
ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql_native_password BY 'Password2025###@@';
```

2.- Edita el fichero mysqld.cnf (`sudo nano /etc/mysql/mysql.conf.d/mysqld.cnf`)
Cambia bind-address (127.0.0.1 → 0.0.0.0)

3.- Ejecuta estos comandos:

```
sudo mysql -u root -p (te pedirá el password...)
```

```
UPDATE mysql.user SET Host='%' WHERE User='root' AND Host='localhost';  
FLUSH PRIVILEGES;
```

4.- Configura el servicio MySQL para que arranque automáticamente:

```
sudo systemctl enable mysql
```

5.- Reinicia el servicio MySQL y comprueba el estado del servicio:

```
sudo service mysql restart
```

```
sudo service mysql status
```

6.- En el panel de control de AWS accede al grupo de seguridad asociado a la instancia y añade una regla de entrada nueva (abrir puerto 3306):

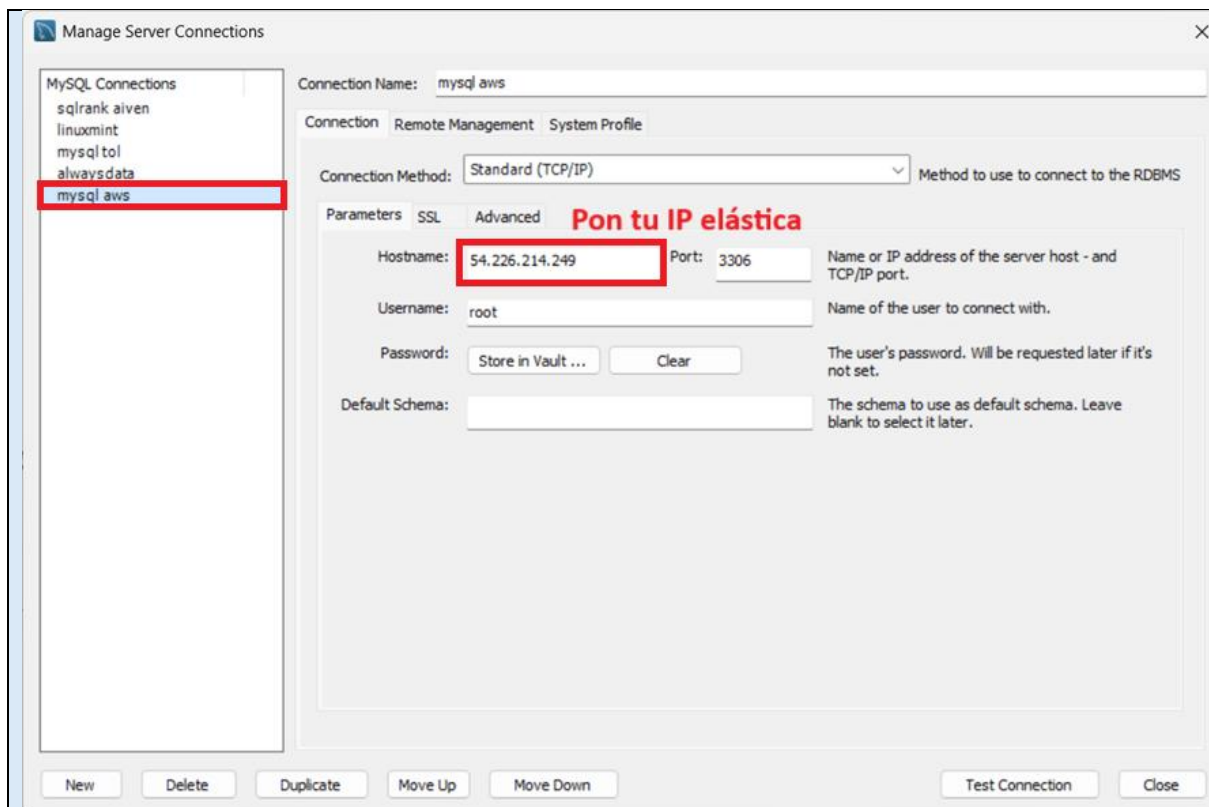
Editar reglas de entrada [Información](#)

Las reglas de entrada controlan el tráfico entrante que puede llegar a la instancia.

Reglas de entrada Información	Tipo Información	Protocolo Información	Intervalo de puertos Información	Origen Información
sgr-05c052400f91aed33	SSH	TCP	22	Perso... <input type="text" value="0.0.0.0"/>
sgr-09d1723597ec4ed95	HTTP	TCP	80	Perso... <input type="text" value="0.0.0.0"/>
sgr-0735490458b991bd2	MYSQL/Aurora	TCP	3306	Perso... <input type="text" value="0.0.0.0"/>

[Agregar regla](#)

7.- Crea una nueva conexión en MySQL Workbench (MySQL AWS por ejemplo) donde tengas en hostname tu IP elástica:



8.- Comprueba que puedes conectarte con MySQL Workbench correctamente al servidor de MySQL instalado en la nube AWS.

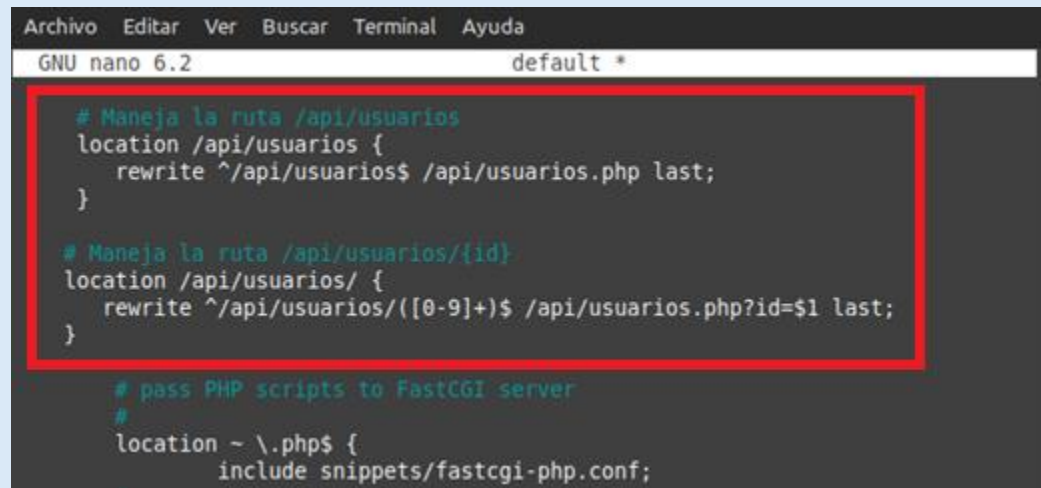
9.- Ejecuta el script SQL para que se genere la base de datos en MySQL

Para desplegar la API REST en AWS:

1. Copia los archivos php creados para la API REST local en la máquina virtual AWS utilizando un terminal y el comando scp. Este comando permite copiar archivos a una máquina remota por ssh. Busca en Internet como hacerlo. El destino de los archivos puede ser en principio /home/ubuntu/
2. Ahora haz una conexión ssh a la máquina virtual en AWS y verifica que los archivos están en /home/ubuntu/
3. Mediante la conexión ssh accede a /var/www/html y crea una carpeta api (mkdir)
4. Copia los archivos de /home/ubuntu a /var/www/html o /var/www/html/api según corresponda.
5. Ajusta en config.php database, password y host (debe ser 127.0.0.1)

6. Ajusta la configuración de Nginx (sudo nano /etc/nginx/sites-available/default) para que reescriba este tipo de URLs:

- <http://localhost/api/usuarios> ==> <http://localhost/api/usuarios.php>
- <http://localhost/api/usuarios/5> ==> <http://localhost/api/usuarios.php?id=5>

A screenshot of the GNU nano 6.2 text editor. The title bar shows 'GNU nano 6.2' and 'default *'. The editor content shows Nginx configuration code. A red rectangle highlights two location blocks. The first block is for '/api/usuarios' and the second is for '/api/usuarios/'. Both use the 'rewrite' directive to redirect to PHP scripts. Below the highlighted blocks, there is a comment about passing PHP scripts to the FastCGI server and a corresponding location block for '\.php\$' which includes 'snippets/fastcgi-php.conf'.

```
Archivo  Editar  Ver  Buscar  Terminal  Ayuda
GNU nano 6.2                                default *

# Maneja la ruta /api/usuarios
location /api/usuarios {
    rewrite ^/api/usuarios$ /api/usuarios.php last;
}

# Maneja la ruta /api/usuarios/{id}
location /api/usuarios/ {
    rewrite ^/api/usuarios/([0-9]+)$ /api/usuarios.php?id=$1 last;
}

# pass PHP scripts to FastCGI server
#
location ~ \.php$ {
    include snippets/fastcgi-php.conf;
```

Para que se aplique la configuración reinicia Nginx: **sudo service nginx restart**

7. Comprueba que la API REST funciona y crea los videos correspondientes. Ten en cuenta que debes acceder a URLs similares a estas: http://ip_elastica/api/usuarios, http://ip_elastica/api/productos

6. CONCLUSIONES

7. REFERENCIAS BIBLIOGRÁFICAS

Incluye en formato APA los recursos web utilizados en la realización del proyecto