

FP055-Introduccion a **las bases de datos.**

AA5. Sentencias DCL Y TCL.

Contenido

Caso teórico.	3
Caso 1.	3
Caso 2.	3
Caso 3.	4
Caso 4.	5
Caso 5.	5
Caso 6.	7
TRANSACCIONES.	7
Caso 7.	7
Caso 8.	8
Caso 9.	8
Caso 10.	9
Caso 11.	9
Caso 12.	10
Caso 13.	12
Caso Practico.	15
Caso 1 - 2.	16
Caso 3.	17
Caso 4.	18
Caso 5.	18
Transacciones.	19
Caso 6.	19
Caso 7.	20
Bibliografía.	20
Caso 1.	20
Eliminar y comprobar privilegios.	20
DROP USER.	20
Caso 5.	20
Caso 8.	20
Caso 9.	20
Caso 10.	20
Caso 11 - 12.	20

Caso teórico.

Caso 1.

Al instalar un servidor MySQL, se crea un usuario denominado `root@localhost`. ¿A qué se refiere este usuario?

El usuario **root@localhost** en un servidor MySQL es una cuenta de usuario predeterminada que se crea automáticamente durante la instalación del servidor MySQL. Este usuario tiene una significación especial y propósitos específicos en la administración de la base de datos MySQL:

Usuario root: Es el usuario administrador por defecto en MySQL. Tiene acceso completo a todas las bases de datos y tablas en el servidor MySQL.

El usuario **root** puede realizar cualquier operación, incluyendo la creación y eliminación de bases de datos, la modificación de esquemas de tablas, la gestión de otros usuarios y sus permisos, y más.

@localhost: La parte **@localhost** especifica el host desde el cual el usuario **root** puede conectarse al servidor MySQL. En este caso, **localhost** indica que el usuario **root** puede conectarse solo desde la misma máquina en la que está instalado el servidor MySQL. Esto significa que no se puede acceder a esta cuenta de usuario **root** de manera remota (a través de la red); solo se permite el acceso local.

Caso 2.

Rellena la siguiente tabla explicando lo que significa cada sección de una instrucción MySQL para crear un nuevo usuario. Explica las partes que conforman un usuario.

CREATE USER	1
'administrador'	2
@'10.0.2.%'	3
IDENTIFIED BY 'adm123';	4

La instrucción **CREATE USER** en MySQL se utiliza para crear un nuevo usuario en la base de datos. Esta instrucción puede desglosarse en varias partes, cada una con un propósito específico. Vamos a detallar cada sección de la instrucción dada como ejemplo:

CREATE USER: Este es el comando inicial que le indica a MySQL que vas a crear un nuevo usuario. Es la declaración estándar para iniciar la creación de un usuario.

'administrador': Este es el nombre del usuario que se está creando. Este nombre es el que el usuario utilizará para iniciar sesión en el servidor MySQL. Los nombres de usuario en MySQL son strings y, por lo tanto, se colocan entre comillas simples.

@'10.0.2.%': Esta parte especifica el host o la red desde donde el usuario podrá conectarse al servidor MySQL. En este caso, cualquier dirección IP que comience con **10.0.2.** podrá ser utilizada para conectarse como **administrador**. El **%** actúa como un comodín, lo que significa que cualquier dirección IP que encaje con **10.0.2.X** será aceptada. Si se usara **@'localhost'**, significaría que el usuario solo puede conectarse desde la misma máquina donde se ejecuta el servidor MySQL.

IDENTIFIED BY 'adm123': Esta sección establece la contraseña para el nuevo usuario. Aquí, **adm123** es la contraseña asignada al usuario. La contraseña también es un string y se coloca entre comillas simples. Este es un elemento crítico para la seguridad del usuario; asegurarse de que la contraseña sea fuerte y segura es esencial.

¿Cuál instrucción MySQL es la utilizada para eliminar un usuario?

Para eliminar un usuario en MySQL, se utiliza la instrucción **DROP USER**. Esta instrucción elimina completamente un usuario existente del servidor MySQL, incluyendo todos sus privilegios. La sintaxis básica para eliminar un usuario es:

```
DROP USER 'nombre_de_usuario'@'host';
```

Donde **nombre_de_usuario** es el nombre del usuario que deseas eliminar y **host** especifica desde dónde puede conectarse ese usuario. Por ejemplo, si quieres eliminar un usuario llamado **ejemplo** que puede conectarse desde cualquier host, usarías:

```
DROP USER 'ejemplo'@'%';
```

Es importante tener en cuenta que esta acción es irreversible y debería usarse con precaución, ya que una vez que se elimina un usuario, todos sus privilegios y configuraciones también se eliminan.

Caso 3.

¿Qué instrucciones se utilizan para otorgar y eliminar privilegios a un usuario de la Base de Datos?

Para gestionar los privilegios de un usuario en una base de datos MySQL, se utilizan principalmente dos instrucciones: **GRANT** y **REVOKE**.

Otorgar Privilegios (GRANT): La instrucción **GRANT** se utiliza para otorgar privilegios específicos a un usuario. La sintaxis general es:

```
GRANT tipo_de_privilegio ON base_de_datos.tabla TO 'nombre_de_usuario'@'host' IDENTIFIED BY 'contraseña';
```

tipo_de_privilegio: El tipo de privilegio que deseas otorgar (como SELECT, INSERT, UPDATE, DELETE, etc.).

base_de_datos.tabla: Especifica la base de datos y/o tabla sobre la cual se otorgan los privilegios.

nombre_de_usuario@host: El nombre del usuario y el host desde donde puede conectarse.

IDENTIFIED BY 'contraseña': Opcionalmente se puede asignar o cambiar la contraseña del usuario.

Ejemplo:

```
GRANT SELECT, INSERT ON miBaseDeDatos.* TO 'usuarioEjemplo'@'localhost';
```

Eliminar Privilegios (REVOKE): La instrucción **REVOKE** se utiliza para eliminar privilegios específicos de un usuario.

Su sintaxis general es:

```
REVOKE tipo_de_privilegio ON base_de_datos.tabla FROM 'nombre_de_usuario'@'host';
```

tipo_de_privilegio, **base_de_datos.tabla**, y **nombre_de_usuario@host** tienen el mismo significado que en la instrucción **GRANT**.

Ejemplo:

```
REVOKE INSERT ON miBaseDeDatos FROM 'usuarioEjemplo'@'localhost';
```

Estas instrucciones permiten un control detallado sobre los accesos y operaciones que cada usuario puede realizar en la base de datos. Es importante utilizarlas con cuidado para mantener la seguridad y la integridad de la base de datos.

Caso 4.

Al otorgar privilegios a un usuario podemos incluir al final de la sentencia, la cláusula **WITH GRANT OPTION**. ¿Para qué se utiliza esta cláusula?

La cláusula **WITH GRANT OPTION** en MySQL se utiliza al **otorgar privilegios a un usuario y permite que dicho usuario, a su vez, pueda otorgar los mismos privilegios que ha recibido a otros usuarios**. Es una forma de delegar la capacidad de administración de privilegios, extendiendo la cadena de permisos más allá del administrador original o del usuario que otorga los privilegios.

Por ejemplo, si otorgas a un usuario privilegios sobre una base de datos con **WITH GRANT OPTION**, este usuario podrá luego otorgar esos mismos privilegios a otros usuarios. La sintaxis general sería:

GRANT tipo_de_privilegio **ON** base_de_datos.tabla **TO**

'nombre_de_usuario'@'host' **WITH GRANT OPTION**;

Esta característica debe usarse con precaución, ya que otorga un nivel significativo de control y puede afectar la seguridad de la base de datos.

Caso 5.

Existen cuatro niveles de privilegios en MySQL. Explica a qué se refieren y rellena la siguiente tabla indicando a qué se refieren siguiendo el ejemplo de las dos primeras líneas:

Los privilegios en MySQL se pueden otorgar en diferentes niveles, cada uno con un ámbito de aplicación específico. Estos niveles son:

Nivel Global: Afecta a todas las bases de datos y tablas del servidor MySQL.

Nivel de Base de Datos (BBDD): Aplica a todas las tablas y rutinas almacenadas dentro de una base de datos específica.

Nivel de Tablas: Se aplica a operaciones específicas en tablas seleccionadas.

Nivel de Rutina Almacenada (Stored Routine): Afecta a las rutinas almacenadas (procedimientos y funciones) específicas.

A continuación, rellenaré la tabla con la descripción de algunos privilegios comunes:

PRIVILEGIOS	Descripción	GLOBAL	BBDD	TABLAS	STORED ROUTINE
ALL [PRIVILEGES]	Otorga todos los privilegios en el nivel específico excepto GRANT OPTION	X	X	X	X
ALTER	Permite cambiar la estructura de las tablas. Dependiendo del nivel donde se presente puede requerir además los privilegios CREATE, INSERT, ALTER o DROP	X	X	X	
ALTER ROUTINE	Permite modificar o eliminar una rutina almacenada				X
CREATE	Permite crear nuevas tablas o bases de datos	X	X		
CREATE ROUTINE	Permite crear rutinas almacenadas		X		
CREATE TABLESPACE	Permite crear tablespaces	X			

PRIVILEGIOS	Descripción	GLOBAL	BBDD	TABLAS	STORED ROUTINE
CREATE TEMPORARY TABLES	Permite crear tablas temporales	X	X		
CREATE USER	Permite crear nuevos usuarios y asignar privilegios	X			
CREATE VIEW	Permite crear vistas	X	X	X	
DELETE	Permite eliminar registros de tablas	X	X	X	
DROP	Permite eliminar bases de datos, tablas y vistas	X	X	X	
EVENT	Permite crear, modificar o eliminar eventos	X	X		
EXECUTE	Permite ejecutar rutinas almacenadas		X		X
FILE	Permite leer y escribir archivos en el servidor	X			
GRANT OPTION	Permite otorgar o revocar privilegios que el usuario posee	X	X	X	X
INDEX	Permite crear o eliminar índices	X	X	X	
INSERT	Permite insertar registros en tablas	X	X	X	
LOCK TABLES	Permite usar el comando LOCK TABLES en tablas	X	X		
PROCESS	Permite ver todos los procesos en ejecución	X			
PROXY	Permite actuar en nombre de otro usuario	X			
REFERENCES	Permite crear claves foráneas	X	X	X	
RELOAD	Permite ejecutar comandos FLUSH	X			
REPLICATION CLIENT	Permite a los servidores esclavos pedir información al maestro	X			
REPLICATION SLAVE	Permite al servidor actuar como esclavo en la replicación	X			
SELECT	Permite leer datos de tablas	X	X	X	
SHOW DATABASES	Permite ver todas las bases de datos	X			
SHOW VIEW	Permite ver las definiciones de las vistas	X	X	X	
SHUTDOWN	Permite apagar el servidor	X			
SUPER	Permite ejecutar varias acciones administrativas	X			

PRIVILEGIOS	Descripción	GLOBAL	BBDD	TABLAS	STORED ROUTINE
TRIGGER	Permite crear o eliminar triggers		X	X	
UPDATE	Permite modificar registros en tablas	X	X	X	
USAGE	Sinónimo de “ningún privilegio”	X	X	X	X

Hay que destacar que algunos privilegios solo tienen sentido en ciertos niveles. Por ejemplo, el privilegio **CREATE TABLESPACE** solo es relevante a nivel global, mientras que **ALTER ROUTINE** solo aplica a rutinas almacenadas específicas.

Caso 6.

¿Qué instrucción se utiliza para mostrar los privilegios de un usuario?

Para mostrar los privilegios de un usuario en MySQL, puedes utilizar la instrucción **SHOW GRANTS**. Esta instrucción se utiliza para listar los privilegios otorgados a un usuario específico. La sintaxis general es la siguiente:

```
SHOW GRANTS FOR 'usuario'@'host';
```

Donde **usuario** es el nombre del usuario y **host** es el host desde el cual el usuario se conecta. Por ejemplo, si deseas ver los privilegios otorgados al usuario **admin** que se conecta desde el host **localhost**, usarías:

```
SHOW GRANTS FOR 'admin'@'localhost';
```

Esto mostrará una lista de todas las sentencias **GRANT** que han sido ejecutadas para ese usuario, detallando sus privilegios en la base de datos.

TRANSACCIONES.

Caso 7.

¿Qué es una transacción? ¿Cuál es el objetivo de la misma?

Una transacción es una secuencia de operaciones que se realiza como una unidad de trabajo. Las transacciones son un concepto fundamental para garantizar la integridad y consistencia de los datos en una base de datos. El objetivo principal de una transacción es asegurar que una serie de operaciones se complete con éxito como un todo. Si alguna de las operaciones dentro de la transacción falla, entonces toda la transacción se revierte, lo que significa que el estado de la base de datos vuelve al punto anterior al inicio de la transacción.

Las características clave de las transacciones son descritas por el acrónimo ACID:

Atomicidad: Garantiza que todas las operaciones dentro de la transacción se ejecuten por completo o ninguna lo haga. Si una operación falla, toda la transacción se revierte.

Consistencia: Asegura que la transacción no violará ninguna restricción de integridad de la base de datos. La transacción lleva la base de datos de un estado consistente a otro también consistente.

Aislamiento: Determina cómo y cuándo los cambios realizados por una transacción son visibles para otras transacciones. El aislamiento ayuda a prevenir conflictos entre transacciones concurrentes.

Durabilidad: Una vez que una transacción se ha comprometido, sus efectos son permanentes en la base de datos, incluso en caso de fallos del sistema.

Caso 8.

En 1970, Jim Gray definió las propiedades que necesitaba tener una transacción confiable. Más tarde, en 1983, Andreas Reuter y Theo Härder crearon el término "ACID" para describir estas 4 propiedades. Rellena la siguiente tabla con la definición de cada propiedad.

Las propiedades ACID son fundamentales para los sistemas transaccionales y se definen de la siguiente manera:

Atomicidad: Una transacción que actúa sobre varias piezas de información solo se completa si todas las piezas se guardan con éxito. Aquí se aplica el principio de "todo o nada" a la transacción.

Consistencia: Los datos guardados no pueden violar la integridad de la base de datos.

Aislamiento: Ninguna otra transacción tiene lugar y afecta a la transacción en cuestión. Esto previene las "colisiones en el aire".

Durabilidad: Los fallos del sistema o los reinicios no afectan a las transacciones comprometidas

¿Cuál es un ejemplo de propiedades ACID?

Podría ser útil examinar las propiedades de la base de datos ACID y sus conceptos mediante un ejemplo.

Para nuestro ejemplo, considere una transacción bancaria en la que está retirando dinero de una cuenta corriente para depositarlo en su cuenta de ahorros. Se registran varias operaciones:

- Retiro de dinero de la cuenta corriente
- Depósito de fondos a ahorros
- Pista de auditoría de transacciones.

¿Cómo ayudaría ACID en esta situación? Repasemos las cuatro propiedades una por una para entenderlas y ver dónde ayudan al banco a mantener registros precisos:

- Queremos una **Transacción Atómica**. El proceso de la OMS para mover dinero de la cuenta corriente a la cuenta de ahorros no puede completarse, a menos que las tres operaciones se completen con éxito. Un fallo en cualquiera, cancela toda la operación.
- Nuestro objetivo es mantener la **coherencia**, asegurándonos de que tanto la transferencia *como* la pista de auditoría estén completas.
- Al utilizar **Isolation**, nos aseguramos de que otras transacciones bancarias no puedan actualizar nuestras cuentas hasta que se complete la operación.
- Mantener la operación **duradera** garantiza que el DBMS no "perderá" las transacciones comprometidas o las guardará. Por ejemplo, si el servidor sufre una pérdida de energía, los datos confirmados no se perderán.

Caso 9.

Los pasos siguientes son los pasos que se utilizan para crear una transacción y siempre son los mismos en cualquier DBMS. Rellena la siguiente tabla indicando las sentencias a utilizar para poder llevarlos a cabo

Iniciar la transacción:

Sentencia SQL: **START TRANSACTION;** o **BEGIN;**

Confirmar la transacción (si todo está correcto):

Sentencia SQL: **COMMIT;**

Deshacer la transacción y dejar la base de datos como estaba antes de iniciarla (en caso de error):

Sentencia SQL: **ROLLBACK;**


```
START TRANSACTION
    [transaction_characteristic [, transaction_characteristic] ...]

transaction_characteristic: {
    WITH CONSISTENT SNAPSHOT
  | READ WRITE
  | READ ONLY
}

BEGIN [WORK]
COMMIT [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
ROLLBACK [WORK] [AND [NO] CHAIN] [[NO] RELEASE]
SET autocommit = {0 | 1}
```

Caso 10.

¿Para qué se utiliza la sentencia AUTOCOMMIT?

La sentencia AUTOCOMMIT en MySQL se utiliza para desactivar o activar el modo de autocommit predeterminado para la sesión actual. Por defecto, MySQL opera con el modo autocommit activado, lo que significa que cuando no se está dentro de una transacción, cada declaración se considera atómica, como si estuviera rodeada por START TRANSACTION y COMMIT. Cuando el autocommit está activado, cada sentencia se considera una transacción completa y se confirma tan pronto como termina.

```
$> mysql test
```

```
mysql> CREATE TABLE customer (a INT, b CHAR (20), INDEX (a));
Query OK, 0 rows affected (0.00 sec)
mysql> -- Do a transaction with autocommit turned on.
mysql> START TRANSACTION;
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO customer VALUES (10, 'Heikki');
Query OK, 1 row affected (0.00 sec)
mysql> COMMIT;
Query OK, 0 rows affected (0.00 sec)
mysql> -- Do another transaction with autocommit turned off.
mysql> SET autocommit=0;
Query OK, 0 rows affected (0.00 sec)
mysql> INSERT INTO customer VALUES (15, 'John');
Query OK, 1 row affected (0.00 sec)
mysql> INSERT INTO customer VALUES (20, 'Paul');
Query OK, 1 row affected (0.00 sec)
mysql> DELETE FROM customer WHERE b = 'Heikki';
Query OK, 1 row affected (0.00 sec)
mysql> -- Now we undo those last 2 inserts and the delete.
mysql> ROLLBACK;
Query OK, 0 rows affected (0.00 sec)
mysql> SELECT * FROM customer;
+-----+
| a | b |
+-----+
| 10 | Heikki |
+-----+
1 row in set (0.00 sec)
mysql>
```

Caso 11.

¿Qué es un SAVEPOINT y cuándo se utiliza esta sentencia?

El comando **SAVEPOINT** en MySQL se utiliza para dividir o romper una transacción en múltiples unidades, ofreciendo la posibilidad de revertir la transacción hasta un punto especificado. Esto significa que con un punto de guardado (**SAVEPOINT**), se puede revertir parte de una transacción en lugar de la transacción completa. Al establecer un **SAVEPOINT** con un nombre especificado, si ya existe un punto de guardado con ese nombre, el antiguo se eliminará.

Los pasos para usar puntos de guardado en MySQL incluyen iniciar una transacción con la declaración **START TRANSACTION**, establecer un **SAVEPOINT** con el comando **SAVEPOINT identificador**, donde el **identificador** es un nombre que eliges

para el punto de guardado, y luego realizar cambios en la base de datos dentro de la transacción.

13.3.4 Sentencias SAVEPOINT, ROLLBACK TO SAVEPOINT y RELEASE SAVEPOINT

```
SAVEPOINT identifier
ROLLBACK [WORK] TO [SAVEPOINT] identifier
RELEASE SAVEPOINT identifier
```

InnoDB soporta las sentencias SQL **SAVEPOINT**, **ROLLBACK TO PUNTO DE GUARDADO**, **LANZAMIENTO SAVEPOINT** y la palabra clave opcional para **ROLLBACK**, **WORK**.

La instrucción **SAVEPOINT** establece un Punto de salvaguarda de transacción con nombre con un nombre de **identificador**. Si la transacción actual tiene un punto de guardado con el mismo nombre, se elimina el punto de guardado anterior y se establece uno nuevo.

La **REVERSIÓN A SAVEPOINT** revierte una transacción en la instrucción Punto de salvaguarda sin finalizar la transacción. Modificaciones que la transacción actual realizada en las filas después del punto de salvaguarda fue set se deshacen en la reversión, pero *no* libera los bloqueos de fila que se almacenaron en memoria después del punto de guardado. (Para una nueva fila insertada, el bloqueo la información es transportada por el ID de transacción almacenado en la fila; La cerradura no se almacena por separado en la memoria. En este caso, la fila El bloqueo se libera en el botón Deshacer). Puntos de guardado que se establecieron en un momento posterior tiempo que el punto de guardado con nombre se eliminan. InnoDB

Si el **ROLLBACK A SAVEPOINT** devuelve el siguiente error, significa que no existe ningún punto de retorno con el nombre especificado:

```
ERROR 1305 (42000): SAVEPOINT identifier does not exist
```

Caso 12.

¿Qué instrucción se utiliza para regresar a un **SAVEPOINT** sin necesidad de eliminar toda la transacción?

LEY DE POLÍTICA DE PROTECCIÓN DE DATOS (LOPD).

Para regresar a un **SAVEPOINT** en una transacción sin necesidad de descartar toda la transacción, se utiliza la sentencia SQL **ROLLBACK TO SAVEPOINT [nombre_del_savepoint]**. Esta sentencia permite deshacer todas las operaciones realizadas después del establecimiento del **SAVEPOINT** especificado, manteniendo intactas las realizadas antes de este punto.

Por ejemplo, si has creado un **SAVEPOINT** llamado **sp1**, puedes regresar a él con la sentencia:

```
ROLLBACK TO SAVEPOINT sp1
```

Esta funcionalidad es útil para manejar errores y situaciones inesperadas en transacciones complejas, permitiendo un control más granular sobre las operaciones y la capacidad de revertir parcialmente los cambios sin afectar toda la transacción.

Example

```
START TRANSACTION;

-- Perform some SQL operations

SAVEPOINT my_savepoint;

-- Perform more SQL operations

-- If an error occurs, you can roll back to the savepoint
ROLLBACK TO my_savepoint;

-- Continue with the transaction or commit
COMMIT;
```

In this example, if an error occurs after the **SAVEPOINT** is set, you can use the **ROLLBACK TO** statement to revert the transaction to the state of the **SAVEPOINT**. This way, you avoid rolling back the entire transaction and only undo the changes made after the **SAVEPOINT**.

A partir de la información la conseguí con ChatGPT:

En cuanto a ejemplos que cumplan con la Ley de Política de Protección de Datos (LOPD), aquí hay algunos escenarios:

Consentimiento explícito para recopilación de datos: Antes de recopilar datos personales de usuarios, se debe obtener su consentimiento explícito y claro. Por ejemplo, un formulario de registro en un sitio web debe incluir una casilla de verificación (no premarcada) que indique que el usuario está de acuerdo con la política de privacidad y el tratamiento de sus datos.

Derecho a ser olvidado: Los usuarios tienen derecho a solicitar que sus datos personales sean eliminados de la base de datos de una empresa. Por ejemplo, un usuario puede pedir a una red social que elimine su perfil y toda la información asociada.

Acceso y rectificación de datos: Los usuarios tienen derecho a acceder a sus datos personales almacenados por una empresa y a solicitar correcciones si estos son incorrectos. Por ejemplo, un banco debe permitir a los clientes revisar y actualizar su información personal.

Protección de datos desde el diseño: Las empresas deben incorporar medidas de protección de datos en el diseño de sus sistemas y procesos. Esto podría incluir el uso de cifrado para proteger los datos de los usuarios y garantizar que sólo el personal autorizado tenga acceso a ellos.

Notificación de brechas de seguridad: En caso de una brecha de seguridad que pueda comprometer datos personales, la empresa debe notificar a las autoridades competentes y a los afectados lo antes posible. Por ejemplo, si una tienda en línea sufre un hackeo que expone datos de clientes, debe informar a estos y a la agencia de protección de datos correspondiente.

Estos ejemplos reflejan cómo las empresas y organizaciones deben manejar los datos personales de acuerdo con la LOPD y otras regulaciones similares como el RGPD (Reglamento General de Protección de Datos) en la Unión Europea.

En cuanto a ejemplos que cumplan con la Ley de Política de Protección de Datos (LOPD), aquí hay algunos escenarios:

Consentimiento explícito para recopilación de datos: Antes de recopilar datos personales de usuarios, se debe obtener su consentimiento explícito y claro. Por ejemplo, un formulario de registro en un sitio web debe incluir una casilla de verificación (no premarcada) que indique que el usuario está de acuerdo con la política de privacidad y el tratamiento de sus datos.

Derecho a ser olvidado: Los usuarios tienen derecho a solicitar que sus datos personales sean eliminados de la base de datos de una empresa. Por ejemplo, un usuario puede pedir a una red social que elimine su perfil y toda la información asociada.

Acceso y rectificación de datos: Los usuarios tienen derecho a acceder a sus datos personales almacenados por una empresa y a solicitar correcciones si estos son incorrectos. Por ejemplo, un banco debe permitir a los clientes revisar y actualizar su información personal.

Protección de datos desde el diseño: Las empresas deben incorporar medidas de protección de datos en el diseño de sus sistemas y procesos. Esto podría incluir el uso de cifrado para proteger los datos de los usuarios y garantizar que sólo el personal autorizado tenga acceso a ellos.

Notificación de brechas de seguridad: En caso de una brecha de seguridad que pueda comprometer datos personales, la empresa debe notificar a las autoridades

competentes y a los afectados lo antes posible. Por ejemplo, si una tienda en línea sufre un hackeo que expone datos de clientes, debe informar a estos y a la agencia de protección de datos correspondiente.

Caso 13.

A partir de la normativa legal sobre Protección de Datos de Carácter Personal, indicar hasta qué punto se ve afectado el coWorking y qué medidas debe tomar.

A partir de la información la conseguí con ChatGPT:

La Ley Orgánica de Protección de Datos Personales y Garantía de los Derechos Digitales (LOPD-GDD) y el Reglamento General de Protección de Datos (RGPD) de la Unión Europea son los principales marcos normativos que regulan la protección de datos personales. Para un coWorking, como cualquier otra entidad que maneja datos personales, estas regulaciones son muy relevantes. A continuación, se describen los aspectos en los que el coWorking se ve afectado y las medidas que debe tomar:

- **Consentimiento Informado:** El coWorking debe obtener el consentimiento explícito de los socios, empleados y clientes para el tratamiento de sus datos personales. Este consentimiento debe ser claro, específico y basado en una declaración o una clara acción afirmativa.
- **Derecho a la Información:** Las personas cuyos datos sean recolectados deben ser informadas sobre quién recopila los datos (el coWorking), con qué fin, durante cuánto tiempo se almacenarán y a quién se pueden divulgar.
- **Derechos de los Interesados:** Los socios y clientes tienen derecho a acceder a sus datos personales, rectificarlos si son incorrectos, suprimirlos, oponerse a su tratamiento y solicitar la limitación o portabilidad de sus datos.
- **Protección de Datos desde el Diseño y por Defecto:** El coWorking debe implementar medidas técnicas y organizativas adecuadas para garantizar que se cumple la normativa de protección de datos, como cifrado, anonimización de datos, entre otros.
- **Registro de Actividades de Tratamiento:** Es necesario llevar un registro detallado de todas las actividades de tratamiento de datos personales, incluyendo la finalidad del tratamiento, categorías de datos tratados y destinatarios de los datos.
- **Notificación de Brechas de Seguridad:** En caso de una violación de datos personales, el coWorking debe notificar a la autoridad competente y, en ciertos casos, a las personas afectadas, en un plazo máximo de 72 horas tras haberse percatado de la brecha.
- **Evaluación de Impacto sobre la Protección de Datos (EIPD):** Para procesos que puedan resultar en un alto riesgo para los derechos y libertades de las personas, como un uso intensivo de datos personales, se requiere realizar una EIPD.
- **Delegado de Protección de Datos (DPD):** Dependiendo del volumen y tipo de datos tratados, puede ser necesario designar un DPD que supervise el cumplimiento de la normativa de protección de datos.
- **Formación y Concienciación:** Es esencial que todo el personal del coWorking reciba formación regular sobre la importancia de la protección de datos y cómo deben ser tratados correctamente.

- **Contratos con Terceros:** Si el coWorking comparte datos con terceros (como proveedores de servicios), debe asegurarse de que estos también cumplen con la normativa de protección de datos.

Estas medidas no solo son obligatorias desde un punto de vista legal, sino que también contribuyen a fomentar la confianza de los socios y clientes en la gestión de sus datos personales por parte del coWorking.

La ley y el artículo correspondiente para cada uno de los puntos mencionados anteriormente, basándome en la Ley Orgánica de Protección de Datos Personales y Garantía de los Derechos Digitales (LOPD-GDD) de España y el Reglamento General de Protección de Datos (RGPD) de la Unión Europea:

- **Consentimiento Informado:**
 - RGPD: Artículo 7 (Condiciones para el consentimiento):
Establece las condiciones bajo las cuales el consentimiento es considerado válido como base legal para el tratamiento de datos personales. Esto incluye la necesidad de que el consentimiento sea libre, específico, informado e inequívoco.
 - LOPD-GDD: Artículo 6 (Consentimiento del afectado):
Regula los aspectos específicos relacionados con el consentimiento en el contexto español, complementando el RGPD.
- **Derecho a la Información:**
 - RGPD: Artículos 12-14 (Transparencia e información):
Establecen los requisitos para proporcionar información a los sujetos de datos sobre el tratamiento de sus datos personales, incluyendo los detalles del responsable del tratamiento, los propósitos del tratamiento, y los derechos de los sujetos de datos.
 - LOPD-GDD: Artículo 13 (Derechos de los interesados):
Detalla los derechos de acceso, rectificación, supresión, y otros relacionados con los datos personales de los sujetos de datos.
- **Derechos de los Interesados:**
 - RGPD: Artículos 15-22 (Derechos de acceso, rectificación, supresión, limitación del tratamiento, portabilidad y oposición):
Incluyen derechos como el acceso, rectificación, supresión (derecho al olvido), limitación del tratamiento, portabilidad de los datos y derecho de oposición.
 - LOPD-GDD: Artículo 13 y siguientes (Derechos de los interesados)
- **Protección de Datos desde el Diseño y por Defecto:**
 - RGPD: Artículo 25 (Protección de datos desde el diseño y por defecto)
Incluyen derechos como el acceso, rectificación, supresión (derecho al olvido), limitación del tratamiento, portabilidad de los datos y derecho de oposición.
- **Registro de Actividades de Tratamiento:**
 - RGPD: Artículo 30 (Registro de actividades de tratamiento):
Exige que los responsables y encargados del tratamiento mantengan un registro de las actividades de tratamiento bajo su responsabilidad.
- **Notificación de Brechas de Seguridad:**
 - RGPD: Artículo 33 y 34 (Notificación de una violación de seguridad de datos personales a la autoridad de control y a los interesados):
Establecen la obligación de notificar a la autoridad de control y, en

ciertos casos, a los sujetos de datos afectados, acerca de brechas de seguridad que puedan suponer un riesgo para los derechos y libertades de las personas naturales.

- **Evaluación de Impacto sobre la Protección de Datos (EIPD):**
 - RGPD: Artículo 35 (Evaluación de impacto sobre la protección de datos):
Requiere que se realice una evaluación de impacto en relación con el tratamiento de datos personales que sea probable que resulte en un alto riesgo para los derechos y libertades de los sujetos de datos.
- **Delegado de Protección de Datos (DPD):**
 - RGPD: Artículos 37-39 (Designación, posición y tareas del delegado de Protección de Datos):
Establece las condiciones para la designación de un DPD, sus funciones y su posición dentro de la organización.
 - LOPD-GDD: Artículo 34 (delegado de protección de datos):
- Detalla los requisitos específicos para la figura del DPD en el contexto de la legislación española.
- **Formación y Concienciación:**
No hay un artículo específico, pero es una práctica recomendada y se infiere de la obligación general de asegurar el tratamiento adecuado de los datos personales (RGPD: Artículo 24, Responsabilidad del responsable del tratamiento)
- **Contratos con Terceros:**
 - RGPD: Artículo 28 (Encargado del tratamiento):
Establece las obligaciones y los requisitos para los encargados del tratamiento, incluyendo la necesidad de tener un contrato o acto jurídico que regule la relación entre el responsable y el encargado del tratamiento.

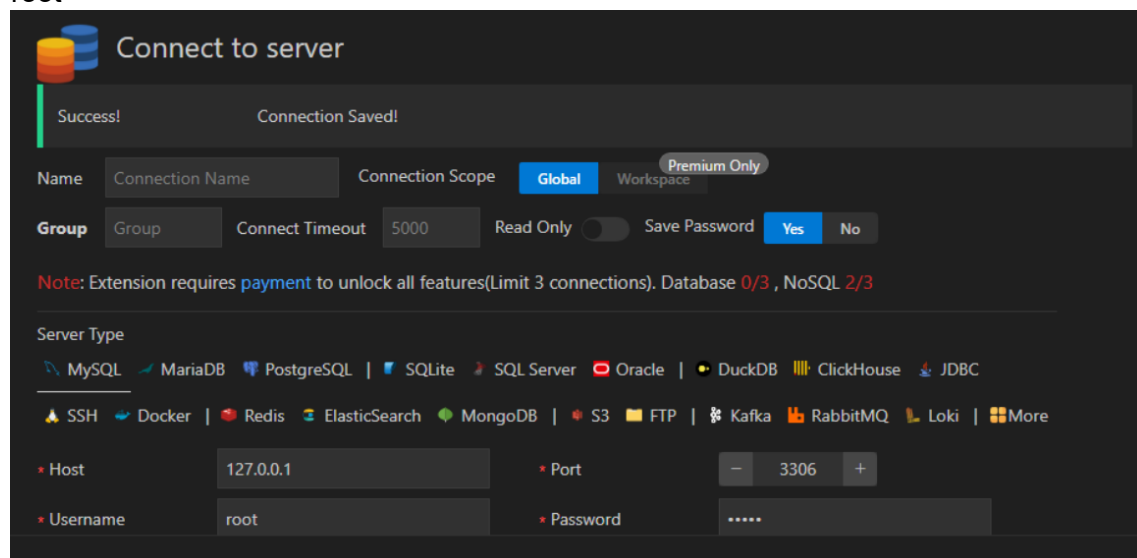
Es importante tener en cuenta que la LOPD-GDD adapta y complementa el RGPD en el contexto español, por lo que ambos deben leerse de forma conjunta para obtener una comprensión completa de los requisitos legales en España.

Caso Practico.

Para hacer los siguientes ejercicios. Lo hice desde mi usuario DDiaz, que ya había creado, desde el inicio del curso.



Me di cuenta de que mi usuario no tenía permisos root, entonces me conecté desde el root



Caso 1 - 2.

```

1  -- Active: 1704456366099@@127.0.0.1@3306@cowkng MySQL
2  -----CASO 1-----
3  -----
4  --Crear Usuario CoWorkAdmin
5  --% permite que el usuario desde cualquier HOST
6  -- PSS: DDiaz
7  > Execute | JSON
8  CREATE USER 'CoWorkAdmin'@'%' IDENTIFIED BY 'DDiaz';
9  -- Otorgamos privilegios
10 > Execute | JSON
11 GRANT CREATE USER ON *.* TO 'CoWorkAdmin'@'%';
12 > Execute | JSON
13 GRANT ALL PRIVILEGES ON *.* TO 'CoWorkAdmin'@'%' WITH GRANT OPTION;
14 --Aplicamos los cambios generados
15 > Execute | JSON
16 FLUSH PRIVILEGES;
17 -----
18 --Me he equivocado y le he dado privilegio para entrar en todas las bases de datos
19 --Tenemos que quitarle los privilegios y luego darle los privilegios de forma adecuada.
20 > Execute | JSON
21 REVOKE ALL PRIVILEGES ON *.* FROM 'CoWorkAdmin'@'%';
22 -----
23 -----CASO 1-----
24 -----
25 -----
26 -----
27 -----
28 -----
29 -----
30 -----
31 -----
32 -----
33 -----
34 -----
35 -----
36 -----
37 -----
38 -----
39 -----
40 -----
41 -----
42 -----
43 -----
44 -----
45 -----
46 -----
47 -----
48 -----
49 -----
50 -----
51 -----
52 -----
53 -----
54 -----
55 -----
56 -----
57 -----
58 -----
59 -----
60 -----
61 -----
62 -----
63 -----
64 -----
65 -----
66 -----
67 -----
68 -----
69 -----
70 -----
71 -----
72 -----
73 -----
74 -----
75 -----
76 -----
77 -----
78 -----
79 -----
80 -----
81 -----
82 -----
83 -----
84 -----
85 -----
86 -----
87 -----
88 -----
89 -----
90 -----
91 -----
92 -----
93 -----
94 -----
95 -----
96 -----
97 -----
98 -----
99 -----
100 -----
  
```

mysql.user X

```

13 -----
14 -----
15 --Me he equivocado y le he dado privilegio para entrar en todas las bases de datos
16 --Tenemos que quitarle los privilegios y luego darle los privilegios de forma adecuada.
17 > Execute | JSON
18 REVOKE ALL PRIVILEGES ON *.* FROM 'CoWorkAdmin'@'%';
19 > Execute | JSON
20 GRANT ALL PRIVILEGES ON Cowkng.* TO 'CoWorkAdmin'@'%' WITH GRANT OPTION;
21 > Execute | JSON
22 FLUSH PRIVILEGES;
23 -----
24 --Para comprobar que el usuario se ha generado de forma correcta
25 > Execute | JSON
26 SELECT user, host FROM mysql.user WHERE user = 'CoWorkAdmin'; 5ms
27 --Mostramos los privilegios que tiene el usuario
28 > Execute | JSON
29 SHOW GRANTS FOR 'CoWorkAdmin'@'%';
30 -----
31 -----CASO 1-----
32 -----
33 -----
34 -----
35 -----
36 -----
37 -----
38 -----
39 -----
40 -----
41 -----
42 -----
43 -----
44 -----
45 -----
46 -----
47 -----
48 -----
49 -----
50 -----
51 -----
52 -----
53 -----
54 -----
55 -----
56 -----
57 -----
58 -----
59 -----
60 -----
61 -----
62 -----
63 -----
64 -----
65 -----
66 -----
67 -----
68 -----
69 -----
70 -----
71 -----
72 -----
73 -----
74 -----
75 -----
76 -----
77 -----
78 -----
79 -----
80 -----
81 -----
82 -----
83 -----
84 -----
85 -----
86 -----
87 -----
88 -----
89 -----
90 -----
91 -----
92 -----
93 -----
94 -----
95 -----
96 -----
97 -----
98 -----
99 -----
100 -----
  
```

mysql.user X

Search results

Cost: 8ms < 1 > Total 1

	user	host
1	CoWorkAdmin	%

```

23 -----
24 -----CASO 1-----
25 -----
26 -----
27 -----
28 -----
29 -----
30 -----
31 -----
32 -----
33 -----
34 -----
35 -----
36 -----
37 -----
38 -----
39 -----
40 -----
41 -----
42 -----
43 -----
44 -----
45 -----
46 -----
47 -----
48 -----
49 -----
50 -----
51 -----
52 -----
53 -----
54 -----
55 -----
56 -----
57 -----
58 -----
59 -----
60 -----
61 -----
62 -----
63 -----
64 -----
65 -----
66 -----
67 -----
68 -----
69 -----
70 -----
71 -----
72 -----
73 -----
74 -----
75 -----
76 -----
77 -----
78 -----
79 -----
80 -----
81 -----
82 -----
83 -----
84 -----
85 -----
86 -----
87 -----
88 -----
89 -----
90 -----
91 -----
92 -----
93 -----
94 -----
95 -----
96 -----
97 -----
98 -----
99 -----
100 -----
  
```

Result X

Search results

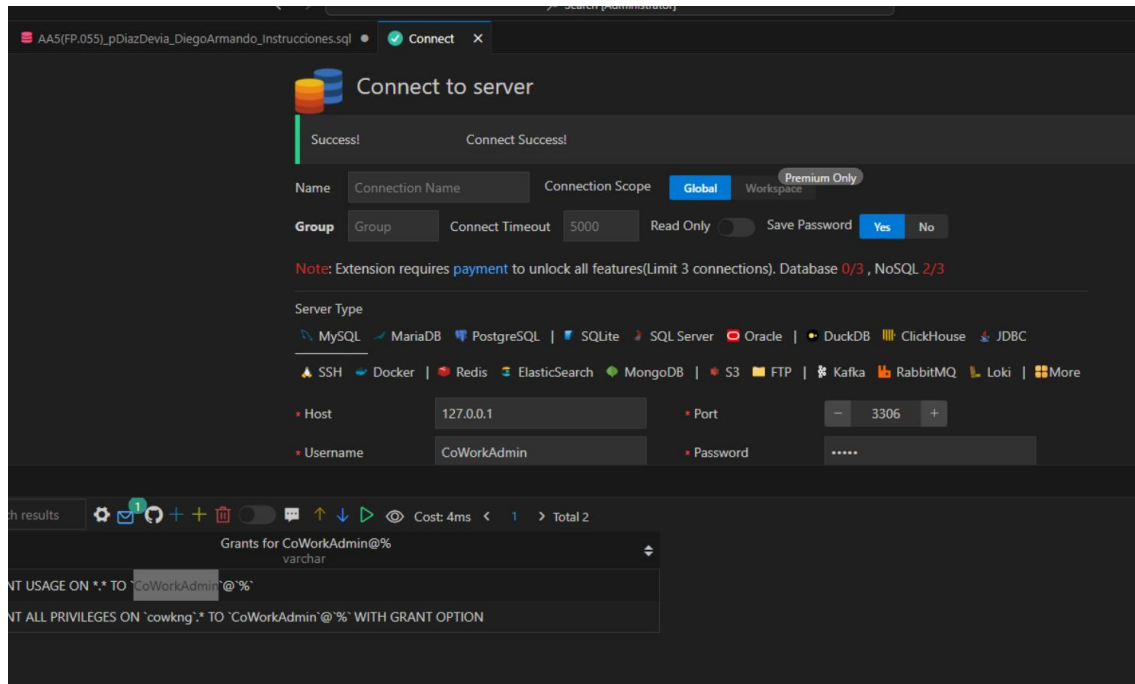
Cost: 4ms < 1 > Total 2

Grants for CoWorkAdmin@%

	Grants for CoWorkAdmin@%
1	GRANT USAGE ON *.* TO 'CoWorkAdmin'@'%'
2	GRANT ALL PRIVILEGES ON `cowkng`.* TO 'CoWorkAdmin'@'%' WITH GRANT OPTION

Caso 3.

Esto lo he hecho de esta manera.



Y al intentarlo a través de comando, este comando no funciona:

mysql -h 127.0.0.1 -u CoWorkAdmin -p

mysql: Es el comando principal que invoca el cliente de línea de comandos de MySQL.

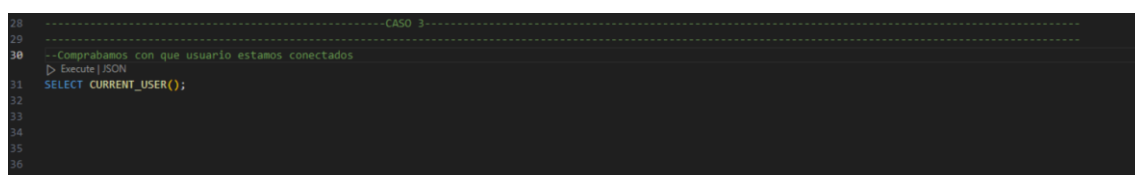
-h: Seguido por la dirección del host (hostname o dirección IP) donde se está ejecutando el servidor de bases de datos MySQL. En tu caso, **127.0.0.1** representa el host local, es decir, tu propia computadora.

-u: Seguido por el nombre del usuario para la autenticación en el servidor de bases de datos MySQL. En este caso, **CoWorkAdmin** es el nombre de usuario que se utilizará para iniciar sesión.

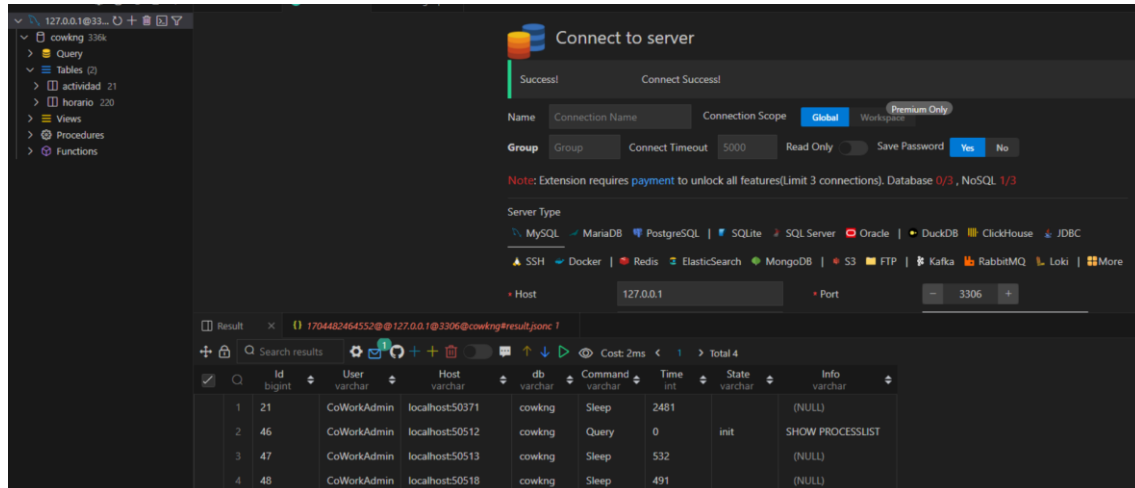
-p: Indica que se proporcionará una contraseña para el usuario especificado con **-u**. Si se coloca inmediatamente una contraseña después de **-p** (sin espacio), esa es la contraseña que se utilizará. Sin embargo, es más seguro omitir la contraseña aquí y en su lugar, ingresarla cuando el sistema la pida después de ejecutar el comando.

-D: Especifica la base de datos con la que conectarse directamente tras el inicio de sesión. En este caso, deberías seguirlo con **Cowkng**, que es el nombre de la base de datos a la que **CoWorkAdmin** tiene acceso.

No funciona ya que estamos trabajando desde Visual Studio, la conexión se hace desde la interfaz gráfica.



Caso 4.



SELECT: Permite al usuario **Supervisor** leer las tablas, lo cual es necesario para ver las actividades y horarios existentes.

INSERT: Permite al usuario **Supervisor** añadir nuevas entradas, lo cual es necesario para crear nuevas actividades y programar horarios.

UPDATE: Permite al usuario **Supervisor** modificar entradas existentes, lo cual es necesario para actualizar información sobre actividades y horarios.

DELETE: Permite al usuario **Supervisor** eliminar entradas, lo cual puede ser necesario para eliminar actividades obsoletas o cambiar horarios.

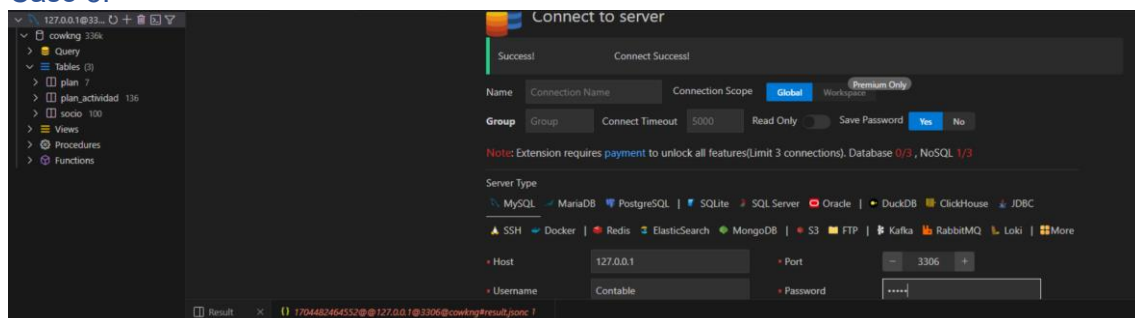
Justificación de los permisos:

Seguridad y Restricción: Estamos otorgando solo los permisos necesarios para las tablas específicas que el **Supervisor** necesita manejar, siguiendo el principio de privilegio mínimo necesario.

Funcionalidad: El **Supervisor** debe ser capaz de manejar completamente las actividades y horarios, lo cual incluye agregar, editar y eliminar entradas en estas tablas.

Control de Acceso: No se otorgan permisos de administración de base de datos como **ALTER** o **DROP**, ni acceso a otras tablas o bases de datos para evitar cambios estructurales o acceso a información no relacionada con sus tareas.

Caso 5.



Justificación de los permisos para el usuario Contable:

El carácter **'%'** en el host permite al contable conectarse desde cualquier dirección, lo que es necesario para trabajar tanto desde la oficina como de forma remota.

Se conceden permisos **SELECT**, **INSERT**, **UPDATE**, y **DELETE** en las tablas relevantes para que el contable pueda realizar todas las operaciones CRUD (Crear,

Leer, Actualizar, Eliminar) necesarias para gestionar la información de planes y suscripciones.

No se otorgan permisos **ALTER** o **DROP** para evitar que el contable pueda hacer cambios estructurales en la base de datos o eliminar tablas, lo que podría afectar la integridad de los datos.

El usuario **Contable** no recibe privilegios administrativos o acceso a otras tablas que contengan información sensible o que no estén relacionadas con sus responsabilidades.

Transacciones.

Caso 6.

```

76 INSERT INTO empresa (nif, empresa, telefono, persona_contacto, email, fecha_inicio_convenio)
77 VALUES ('5487628', 'LosBitMasters', '01548859665', 'Diego', 'Diego@losbitmasters.com', '2023-07-01');
78 -- Asumiendo que ya conoces el NIF de la empresa y que el nuevo socio es un socio corporativo
79 -- Ejecuta | JSON
80 -- Si ambas inserciones fueron exitosas, confirmar los cambios
81 -- Ejecuta | JSON
82 COMMIT;
83
84 --Vamos a verificar que se han creado de forma correcta
85 --Verificar la inserción en la tabla socio.
86 -- Ejecuta | JSON
87 SELECT * FROM socio WHERE documento_identidad = 'DocX64542D'; 6ms
88

```

id_socio	documento_identidad	nombre	apellido1	apellido2	sexo	fecha_nacimiento	id_plan	fecha_alta	activo	tarjeta_acceso	telefono_contacto	email	codigo	
1	105	DocX64542D	ElCabalero	Mascara	Dorada	M	1986-02-11	4	2024-01-11	1	(NULL)	1205488888	Elgringito@gringoland.com	52048

```

86 -- Ejecuta | JSON
87 SELECT * FROM socio WHERE documento_identidad = 'DocX64542D';
88 --Verificar la inserción en la Tabla corporativo:
89 -- Ejecuta | JSON
90 SELECT * FROM corporativo WHERE id_socio = (SELECT id_socio FROM socio WHERE documento_identidad = 'DocX64542D'); 2ms
91

```

id_socio	nif
1	105

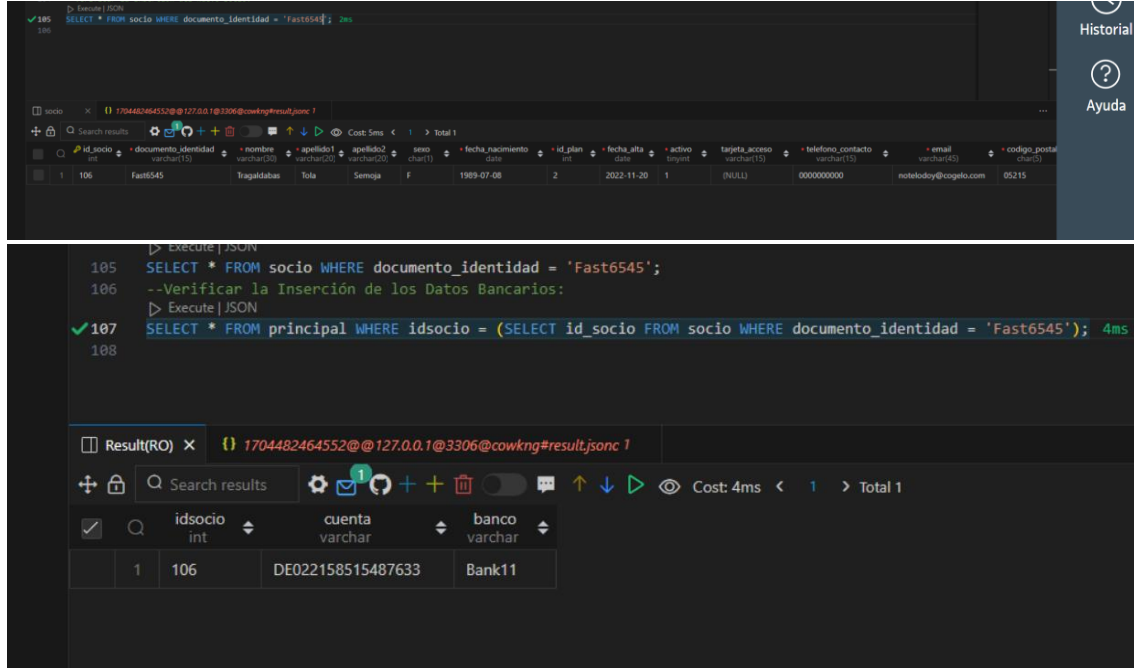
```

90 -- Ejecuta | JSON
91 SELECT * FROM empresa WHERE nif = '5487628'; 3ms
92

```

nif	empresa	telefono	persona_contacto	email	fecha_inicio_convenio	fecha_fin_convenio	
1	5487628	LosBitMasters	01548859665	Diego	Diego@losbitmasters.com	2023-07-01	(NULL)

Caso 7.



```

105 SELECT * FROM socio WHERE documento_identidad = 'Fast6545';
106 --Verificar la Inserción de los Datos Bancarios:
107 SELECT * FROM principal WHERE idsocio = (SELECT id_socio FROM socio WHERE documento_identidad = 'Fast6545');
108

```

idsocio	cuenta	banco
106	DE022158515487633	Bank11

Bibliografía.

Caso 1.

[Web1.](#)

Eliminar y comprobar privilegios.

[Web 1](#), [Web2](#), [Web3](#).

DROP USER.

[Web 1.](#)

Caso 5.

[Web 1](#), [Tutorial1\(1:30\)](#), [Tutorial2\(lista completa\)](#), [Tutorial3](#).

Caso 8.

[Web1.](#)

Caso 9.

[Web1.](#)

Caso 10.

[Web1](#), [Web2](#).

Caso 11 - 12.

[Web1](#), [Web2](#), [Web3](#).