

# PLAN DE SEGURIDAD

## INTEGRANTES:

MARÍA LORENA ASCENSIÓN ANDRÉS M\_210885

DULCE ESMERALDA HERNÁNDEZ JUÁREZ M\_ 210542

EDGAR PÉREZ GARRIDO M\_210519

LUZ ADRIANA REYES GONZÁLEZ M\_ 210277

## ADMINISTRACIÓN DE BASES DE DATOS

**M.T.I. MARCO ANTONIO RAMÍREZ HERNÁNDEZ**

**ENERO - ABRIL 2024**

## INDICE

Objetivo General .....	3
Objetivos específicos .....	3
Políticas de acceso .....	3
Calendario de respaldos.....	6
Evaluación de respaldos.....	7

## **Objetivo General**

El presente documento tiene como objetivo garantizar la integridad, disponibilidad y confidencialidad de los datos sensibles del sistema en cuestión. Esto implica implementar procedimientos y tecnologías que aseguren la realización de copias de seguridad de manera regular y eficiente, así como la protección de esas copias contra pérdidas, alteraciones o accesos no autorizados, ya que el plan debe incluir políticas y protocolos de recuperación ante desastres para asegurar la continuidad del sistema en caso de incidentes graves.

## **Objetivos específicos**

- Proteger la información confidencial de los clientes y del gimnasio.
- Garantizar la disponibilidad de la webapp del gimnasio.
- Minimizar el impacto de un incidente en el proyecto.
- Permitir la recuperación rápida del proyecto en caso de un incidente.

## **El plan se basa en los siguientes principios:**

Confidencialidad: La información confidencial debe ser protegida contra el acceso no autorizado.

Integridad: La información y los sistemas deben ser precisos y completos.

Disponibilidad: La webapp del gimnasio debe estar disponible para los usuarios cuando la necesiten.

## **Políticas de acceso**

### **Roles**

Los roles creados en una base de datos son de suma importancia ya que son una herramienta poderosa para gestionar la seguridad, simplificar la administración de usuarios y garantizar el cumplimiento normativo, proporcionando un enfoque flexible y eficiente para controlar el acceso a los datos y funcionalidades de la base de datos.

Para este caso los roles de la base de datos son los siguientes:

- Cliente

Se refiere a la persona o entidad que adquiere productos o servicios de una empresa.

- Gerente

Se trata del individuo responsable de supervisar y dirigir las operaciones de una organización o de un equipo dentro de una empresa.

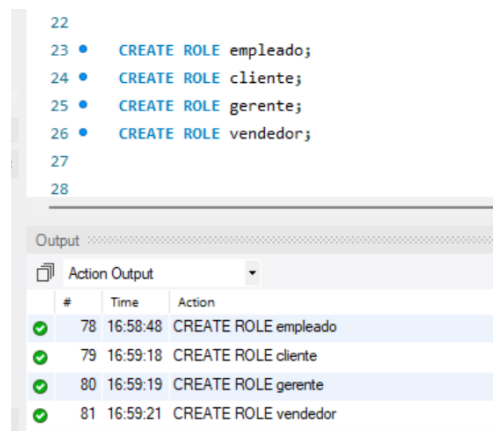
- Empleado

Es un individuo contratado por una empresa encargado de realizar tareas específicas en función de un contrato laboral o acuerdo del trabajo asignado.

- Vendedor

Sujeto que se dedica a la venta de productos o servicios a clientes potenciales en nombre de la empresa a la que presta servicios o ya sea por cuenta propia.

## Creación de roles en SQL




## Creación de roles en NOSQL

```
> db.createRole({
  role: "gerente",
  privileges: [
    { resource: { db: "M4DEL", collection: "comentarios" }, actions: ["find", "insert", "update", "remove"] }
  ],
  roles: []
})
< { ok: 1 }
```

## Usuarios

Los usuarios en una base de datos se encargan de realizar diversas tareas relacionadas con el acceso, manipulación y gestión de la información almacenada en la base de datos. Es por ello que es importante definir qué tipo de rol se asignará a cada uno.

## Creación de usuarios y asignación de roles en SQL



The image shows two screenshots of a SQL command window. The left screenshot displays four SQL commands to create users: 'lorena', 'dulce', 'edgar', and 'luz'. The right screenshot displays four SQL commands to grant roles to these users: 'empleado' to 'lorena', 'vendedor' to 'dulce', 'gerente' to 'edgar', and 'cliente' to 'luz'. Both screenshots include an 'Output' pane at the bottom showing the execution results of the commands.

```
28 • CREATE USER 'lorena'@'%' IDENTIFIED BY 'lorena123';
29 • CREATE USER 'dulce'@'%' IDENTIFIED BY 'dulce123';
30 • CREATE USER 'edgar'@'%' IDENTIFIED BY 'edgar123';
31 • CREATE USER 'luz'@'%' IDENTIFIED BY 'luz123';
32
```

#	Time	Action
82	17:03:28	CREATE USER 'lorena'@'%' IDENTIFIED BY 'lorena123'
83	17:03:29	CREATE USER 'dulce'@'%' IDENTIFIED BY 'dulce123'
84	17:03:31	CREATE USER 'edgar'@'%' IDENTIFIED BY 'edgar123'
85	17:03:32	CREATE USER 'luz'@'%' IDENTIFIED BY 'luz123'

```
27 • GRANT empleado TO 'lorena'@'%'
28 • GRANT vendedor TO 'dulce'@'%'
29 • GRANT gerente TO 'edgar'@'%'
30 • GRANT cliente TO 'luz'@'%'
31
```

#	Time	Action
95	17:29:03	GRANT empleado TO 'lorena'@'%'
96	17:29:04	GRANT vendedor TO 'dulce'@'%'
97	17:29:06	GRANT gerente TO 'edgar'@'%'
98	17:29:07	GRANT cliente TO 'luz'@'%'

## Creación de usuarios y asignación de roles en NOSQL

```
> db.createUser({
  user: "edgar",
  pwd: "edgar123",
  roles: ["gerente"]
})
< { ok: 1 }
```

## Privilegios

Los privilegios son esenciales en una base de datos para garantizar el control sobre los movimientos realizados en cada una de las tablas, así como la ética que cada usuario posee, ya que significa una prevención de errores y la responsabilidad de las acciones realizadas por los usuarios.

## Asignación de privilegios en SQL

```

38 • GRANT SELECT, INSERT, UPDATE ON m4del.* TO empleado;
39 • GRANT SELECT, INSERT, UPDATE ON m4del.* TO vendedor;
40 • GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, DROP ON m4del.* TO gerente;
41 • GRANT SELECT ON m4del.* TO cliente;
42
43 • FLUSH PRIVILEGES; /*para guardar los cambios*/
44

```

Output			
Action Output			
#	Time	Action	Message
99	17:30:32	GRANT SELECT, INSERT, UPDATE ON m4del.* TO empleado	0 row(s) affected
100	17:30:34	GRANT SELECT, INSERT, UPDATE ON m4del.* TO vendedor	0 row(s) affected
101	17:30:35	GRANT SELECT, INSERT, UPDATE, DELETE, CREATE, ALTER, DROP O...	0 row(s) affected
102	17:30:36	GRANT SELECT ON m4del.* TO cliente	0 row(s) affected
103	17:30:38	FLUSH PRIVILEGES	0 row(s) affected

## Asignación de privilegios en NOSQL

```

> db.createRole({
  role: "cliente",
  privileges: [
    { resource: { db: "M4DEL", collection: "comentarios" }, actions: ["find"] }
  ],
  roles: []
})
< { ok: 1 }
> db.createUser({
  user: "luz",
  pwd: "luz123",
  roles: ["cliente"]
})
< { ok: 1 }

```

## Calendario de respaldos

Nuestra UDN cuenta con las siguientes tablas, de las cuales somos los encargados:

### SQL

1. Productos
2. Detalle\_Productos
3. Detalle\_Pedidos
4. Pedidos
5. Promociones
6. .Detalle\_Promociones

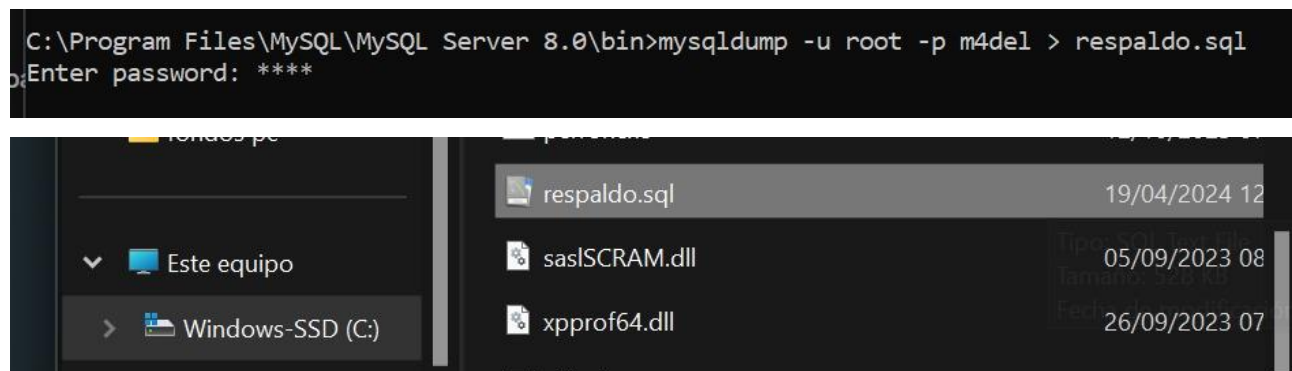
### NOSQL

1. Comentarios\_productos

Los datos se pueden salvaguardar de la siguiente manera:

- Copia completa de los datos de la plataforma (un día)
- Copias diarias sucesivas de los datos salvaguardando a disco solo las diferencias con el día anterior. De esta manera ahorramos espacio en disco evitando la copia completa de todos los datos a diario.

Los respaldos en MYSQL se pueden realizar como se muestra a continuación:



Las medidas anteriores serán aplicadas a las tablas de mayor importancia que contengan datos relevantes para el sistema como lo son:

1. Productos
2. Detalle\_Productos
3. Detalle\_Pedidos
4. Pedidos

Para las tablas restantes deberán realizarse copias de respaldo al menos semanalmente, salvo que en dicho periodo no se hubiera producido ninguna actualización de los datos.

Localización de los backups

- Discos duros.
- Memorias USB
- la nube (Internet).

## Evaluación de respaldos

Cada tipo de respaldo debería ser evaluado de forma periódica para asegurarse de que los datos se pueden leer. Es un hecho que algunas veces se realizan los respaldos que

son, de una forma u otra, ilegibles. La parte desafortunada en todo esto es que muchas veces esto se nota hasta que los datos se pierden y se deben restaurar desde el respaldo.

Las razones para estos pueden variar desde cambios en la alineación de los cabezales de la unidad de cinta, software de respaldo mal configurado o un error del operador. No importa la causa, sin las revisiones periódicas no se puede estar seguro de si está generando respaldos a partir de los cuales se puedan restaurar los datos más adelante.