

Práctica Final de Bases de Datos: Creación de un negocio innovador

Blanca Lozano Collado

Índice:

1. Revisión Bibliográfica

- a. Idea de negocio innovador
- b. Tipo de datos
- c. Importancia de la Base de Datos
- d. Qué SGBD utiliza
- e. Tecnologías adicionales

2. Creación del diagrama ER

- a. Al menos 7 tablas de entidades propias, no tablas intermedias
- b. Relaciones complejas (uno a muchos, muchos a muchos, etc.)
- c. Claves primarias y foráneas correctamente definidas.
- d. Restricciones de integridad adecuadas.
- e. Cardinalidades justificadas.

3. Implementación en MySQL

- a. Creación de las tablas con sus respectivas relaciones.
- b. Inserción de al menos dos registros por tabla.

4. Manipulación y modificación de datos

- a. 30 consultas SELECT en las que se filtre las tablas, necesario que hayan JOINS y al menos 3 subconsultas. La complejidad de las mismas se valorará este punto en la nota .
- b. Procedimiento almacenado con una función

1. Revisión Bibliográfica

1.1 - La idea de negocio innovador.

La idea de negocio para realizar este trabajo consiste en una plataforma de alquiler de ropa para diversas ocasiones como bodas, graduaciones, entre otros eventos. Su objetivo es ofrecer la posibilidad de alquilar prendas de vestir para cualquier ocasión, ya que muchas veces compramos trajes o vestidos que al final solo nos acabamos poniendo una vez. Con esta idea tenemos en cuenta factores como el tipo de evento, las preferencias personales del usuario y se ofrece una alternativa económica al consumo de las prendas de vestir de alta calidad.

1.2 - Qué tipo de datos se manejarán.

- **Clientes:** En esta tabla se almacenará información personal de cada cliente como: nombre, email, teléfono, ciudad, dirección, tipo de suscripción (gratuita o premium) y fecha de registro en la plataforma.
- **Prendas:** Cada prenda de la plataforma incluirá información como: nombre, tipo, talla, color, marca, stock y estado actual de la prenda (disponible o no disponible)
- **Alquileres:** Se registra información sobre el alquiler de las prendas incluyendo: fecha de inicio y fin del alquiler.
- **Evento:** Cada alquiler está vinculado a un evento del cual se almacena: tipo de evento, fecha, ubicación, clima esperado y dress code.
- **Recomendaciones:** Almacena información como: motivo de recomendación, fecha y estado (pendiente, aceptada o rechazada). Esta tabla además depende de una suscripción premium por parte del usuario.
- **Estilos:** Tabla que recoge los estilos asociados a un evento, incluye información como la descripción y la categoría del estilo.
- **Estilistas:** Expertos que asisten a los usuarios con la recomendación de prendas. De estos se almacenará información como: nombre, email, teléfono y especialidad.

1.3 - Cuál es la importancia de una base de datos en este contexto

Para la implementación de esta plataforma una base de datos bien estructurada es **fundamental**. Permite ofrecer un servicio personalizado, controlar el stock y el estado de las prendas, gestionar alquileres, automatizar procesos, garantizar la seguridad de los datos y mejorar la eficiencia general del sistema.

1.4 - Qué Sistema Gestor de Bases de Datos (SGBD) utiliza y por qué

Vamos a utilizar **MySQL**, un sistema de gestión de bases de datos relacional que permite organizar los datos en tablas relacionadas entre sí, facilitando la integridad y eficiencia en el manejo de los datos.

1.5 - Tecnologías adicionales que empleamos en la implementación

Vamos a trabajar usando tecnologías como **XAMPP** como entorno de desarrollo local y **MySQL Workbench** para el diseño del diagrama ER y ejecución de consultas de forma visual. Además, para dibujar el diagrama ER he utilizado mi IPad.

2. Creación del diagrama ER

Un cliente puede realizar tantos alquileres como desee. Si el cliente es premium, puede acceder (de forma opcional) al sistema de recomendaciones y puede solicitar tantas como quiera, así como hacer sesiones de recomendación conjunta junto con otros usuarios de la plataforma. Estas recomendaciones son atendidas por estilistas, profesionales que ayudan a elegir prendas. Dichas recomendaciones pueden o no derivar en un alquiler, donde el cliente decide alquilar una prenda para un evento y estilo específico.

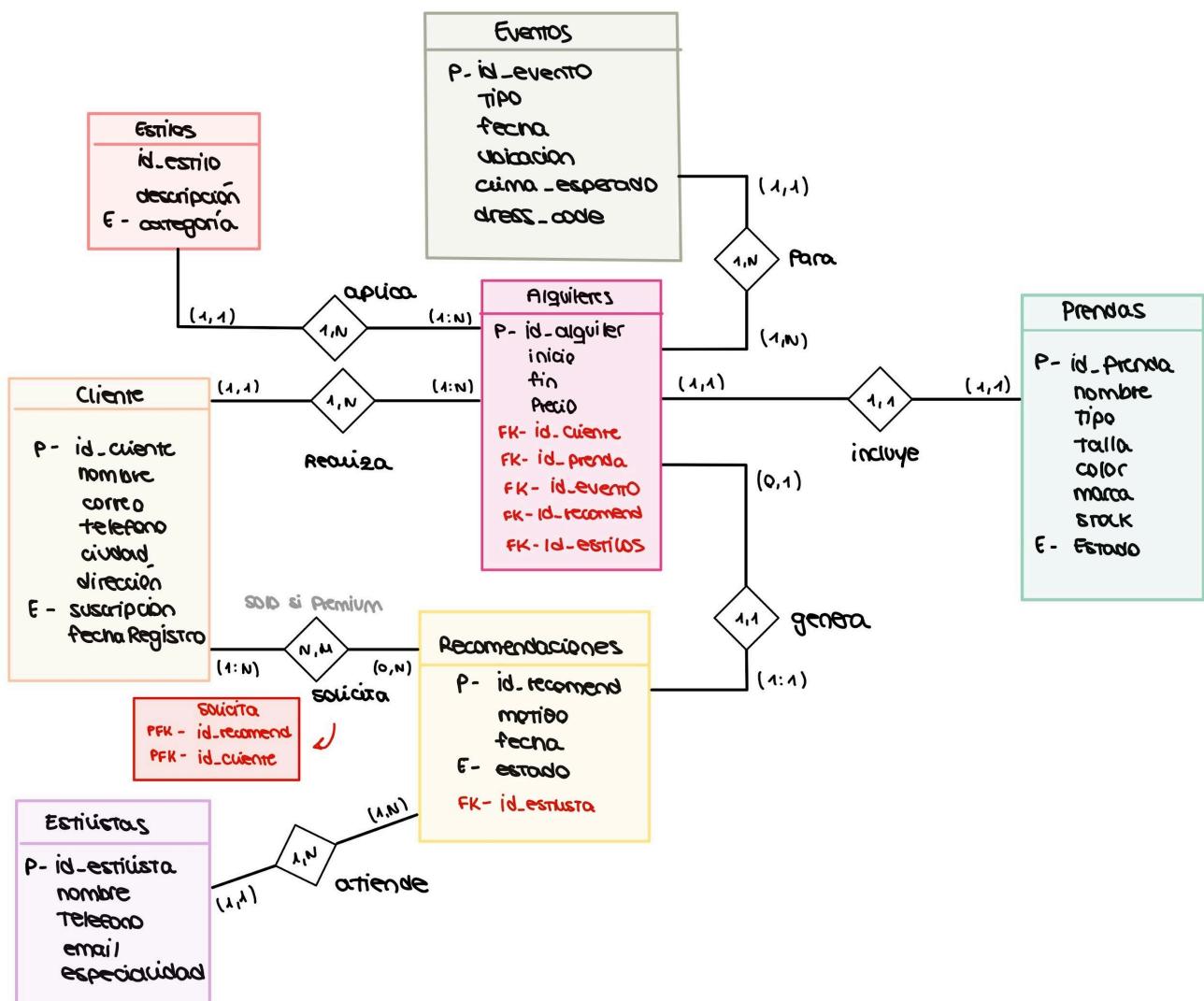
Cada evento puede estar vinculado a múltiples alquileres si varios clientes alquilan prendas para la misma ocasión.

Cada prenda sólo puede ser alquilada una vez por usuario y solamente una prenda por alquiler.

Cada estilo representa una estética general (formal, elegante..) y puede aplicarse a muchos alquileres.

Los estilistas no están ligados directamente con los alquileres, sino que solamente asesoran las recomendaciones donde asisten los clientes premium que quieren una experiencia personalizada de la plataforma.

El diagrama Entidad-Relación siguiendo dichas pautas, tiene la siguiente forma:



3. Implementación en MySQL

3.1 - Creación de las tablas

El código implementado para dicho diagrama ER en MySQL tiene la siguiente forma:

```
1 • CREATE DATABASE Proyecto;
2 • Use Proyecto;
3
4 • CREATE TABLE Clientes (
5     id_cliente INT AUTO_INCREMENT PRIMARY KEY,
6     nombre VARCHAR(100),
7     correo VARCHAR(100) UNIQUE,
8     telefono CHAR(9),
9     ciudad VARCHAR(100),
10    direccion VARCHAR(250),
11    suscripcion ENUM('gratuito', 'premium') DEFAULT 'gratuito',
12    fecha_registro DATETIME DEFAULT CURRENT_TIMESTAMP
13 );
14
15 • CREATE TABLE Estilos (
16     id_estilo INT AUTO_INCREMENT PRIMARY KEY,
17     descripcion VARCHAR(500),
18     categoria VARCHAR(100)
19 );
20
21 • CREATE TABLE Eventos (
22     id_evento INT AUTO_INCREMENT PRIMARY KEY,
23     tipo VARCHAR(100),
24     fecha DATE,
25     ubicacion VARCHAR(255),
26     clima Esperado VARCHAR(100),
27     dress_code VARCHAR(100)
28 );
29
30 • CREATE TABLE Estilistas(
31     id_estilista INT AUTO_INCREMENT PRIMARY KEY,
32     nombre VARCHAR(100),
33     telefono CHAR(9),
34     correo VARCHAR(100) UNIQUE,
35     especialidad VARCHAR(255)
36 );
37
38 • CREATE TABLE Prendas(
39     id_prenda INT AUTO_INCREMENT PRIMARY KEY,
40     nombre VARCHAR(200),
41     tipo VARCHAR(100),
42     talla CHAR(1),
43     color VARCHAR(100),
44     marca VARCHAR(200),
45     stock INT,
46     estado ENUM('disponible','no disponible') DEFAULT 'disponible'
47 );
48
49 • CREATE TABLE Recomendaciones (
50     id_recomendacion INT AUTO_INCREMENT PRIMARY KEY,
51     motivo VARCHAR(200),
52     fecha DATE,
53     estado ENUM('aceptada', 'solicitada','rechazada'),
54     id_estilista INT,
55     FOREIGN KEY(id_estilista) REFERENCES Estilistas(id_estilista)
56 );
```

```
58 • CREATE TABLE Alquileres (
59     id_alquiler INT AUTO_INCREMENT PRIMARY KEY,
60     inicio DATE,
61     fin DATE,
62     id_cliente INT,
63     id_prenda INT,
64     id_evento INT,
65     id_recomendacion INT,
66     id_estilo INT,
67     FOREIGN KEY(id_cliente) REFERENCES Clientes(id_cliente),
68     FOREIGN KEY(id_prenda) REFERENCES Prendas(id_prenda),
69     FOREIGN KEY(id_evento) REFERENCES Eventos(id_evento),
70     FOREIGN KEY(id_recomendacion) REFERENCES Recomendaciones(id_recomendacion),
71     FOREIGN KEY(id_estilo) REFERENCES Estilos(id_estilo)
72 );
73
74 • CREATE TABLE Solicitud (
75     id_cliente INT,
76     id_recomendacion INT,
77     PRIMARY KEY(id_cliente, id_recomendacion),
78     FOREIGN KEY(id_cliente) REFERENCES Clientes(id_cliente),
79     FOREIGN KEY(id_recomendacion) REFERENCES Recomendaciones(id_recomendacion)
80 );
```

3.2 - Inserción de registros a las tablas

CLIENTES:

```
INSERT INTO Clientes(nombre, correo, telefono, ciudad, direccion, suscripcion)
VALUES
('Ana Garcia', 'anag@gmail.com', '654256178', 'Madrid', 'Calle Sol 45', 'premium'),
('Marta Perez', 'martap@gmail.com', '632647510', 'Valencia', 'Avenida Mar 12', 'gratuito');
```

	id_cliente	nombre	correo	telefono	ciudad	direccion	suscripcion	fecha_registro
▶	1	Ana Garcia	anag@gmail.com	654256178	Madrid	Calle Sol 45	premium	2025-05-22 08:06:27
2	Marta Perez	martap@gmail.com	HULL	632647510	Valencia	Avenida Mar 12	gratuito	2025-05-22 08:06:27
*	HULL	HULL	HULL	HULL	HULL	HULL	HULL	HULL

PRENDAS:

```
INSERT INTO Prendas(nombre, tipo, talla, color, marca, stock, estado)
VALUES
('Vestido largo azul', 'Vestido', 'M', 'Azul', 'Zara', 2, 'disponible'),
('Traje gris', 'Traje', 'L', 'Gris', 'Massimo Dutti', 2, 'disponible');
```

	id_prenda	nombre	tipo	talla	color	marca	stock	estado
▶	1	Vestido largo azul	Vestido	M	Azul	Zara	2	disponible
2	Traje gris	Traje	L	Gris	Massimo Dutti	2	disponible	HULL
*	HULL	HULL	HULL	HULL	HULL	HULL	HULL	HULL

ESTILISTAS:

```
INSERT INTO Estilistas(nombre, correo, telefono, especialidad)
VALUES
('Rodrigo Sanchez', 'rodrigos@gmail.com', '645309730', 'Bodas'),
('Laura Gallo', 'laurag@gmail.com', '678539860', 'Graduaciones');
```

	id_estilista	nombre	correo	telefono	especialidad
▶	1	Rodrigo Sanchez	rodrigos@gmail.com	645309730	Bodas
2	Laura Gallo	laurag@gmail.com	HULL	678539860	Graduaciones
*	HULL	HULL	HULL	HULL	HULL

EVENTOS:

```
INSERT INTO Eventos(tipo, fecha, ubicacion, clima Esperado, dress_code)
VALUES
('Boda', '2025-06-15', 'Sevilla', 'Soleado', 'Formal'),
('Graduación', '2025-06-09', 'Barcelona', 'Nublado', 'Semi-formal');
```

	id_evento	tipo	fecha	ubicacion	clima Esperado	dress_code
▶	1	Boda	2025-06-15	Sevilla	Soleado	Formal
2	Graduación	HULL	2025-06-09	Barcelona	Nublado	Semi-formal
*	HULL	HULL	HULL	HULL	HULL	HULL

RECOMENDACIONES:

```
INSERT INTO Recomendaciones(motivo, fecha, estado, id_estilista)
VALUES
('Recomendación para una boda en verano', '2025-07-12', 'solicitada', 1),
('Look para evento formal nocturno', '2025-05-05', 'aceptada', 2);
```

	id_recomendacion	motivo	fecha	estado	id_estilista
▶	1	Recomendación para una boda en verano	2025-07-12	solicitada	1
2	Look para evento formal nocturno	HULL	2025-05-05	aceptada	2
*	HULL	HULL	HULL	HULL	HULL

ALQUILERES

```
INSERT INTO Alquileres(inicio, fin, id_cliente, id_prenda, id_evento, id_recomendacion, id_estilo)
VALUES
('2025-06-10', '2025-06-17', 1, 1, 1, 1, 1),
('2025-06-28', '2025-07-03', 2, 2, 2, 2, 2);
```

	id_alquiler	inicio	fin	id_cliente	id_prenda	id_evento	id_recomendacion	id_estilo
▶	1	2025-06-10	2025-06-17	1	1	1	1	1
2	2025-06-28	2025-07-03	HULL	2	2	2	HULL	2
*	HULL	HULL	HULL	HULL	HULL	HULL	HULL	HULL

SOLICITA

```
INSERT INTO Solicitud(id_cliente, id_recomendacion)
VALUES
(1,1),
(1,2);
```

	id_cliente	id_recomendacion
▶	1	1
2	1	2
*	HULL	HULL

ESTILOS:

```
INSERT INTO Estilos(descripcion, categoria)
VALUES
('Eventos formales', 'formal'),
('Eventos casuales', 'casual');
```

	id_estilo	descripcion	categoria
▶	1	Eventos formales	formal
2	Eventos casuales	casual	HULL
*	HULL	HULL	HULL

4. Manipulación y modificación de datos

4.1 - 30 consultas SELECT (necesario que hayan JOINS) y al menos 3 subconsultas.

1. Mostrar todos los clientes con suscripción premium

```
SELECT nombre FROM Clientes  
WHERE suscripcion = 'premium';
```

2. Mostrar prendas disponibles en stock

```
SELECT *  
FROM Prendas  
WHERE estado = 'disponible' AND stock > 0;
```

3. Mostrar los clientes con más de un alquiler

```
SELECT clientes.nombre, count(alquileres.id_alquiler) AS total_alquileres  
FROM Clientes  
JOIN Alquileres ON Clientes.id_cliente = Alquileres.id_cliente  
GROUP BY Clientes.id_cliente  
HAVING(total_alquileres) > 1;
```

4. Mostrar la información completa de los alquileres

```
SELECT *  
from Alquileres;
```

5. Mostrar los alquileres terminados

```
SELECT *  
FROM Alquileres  
WHERE fin < curdate();
```

6. Mostrar los eventos que hayan generado más de 3 alquileres

```
SELECT e.tipo, count(a.id_alquiler) as total_alquileres  
FROM Eventos e  
JOIN Alquileres a ON e.id_evento = a.id_evento  
HAVING (total_alquileres) > 3;
```

7. Mostrar los clientes que hayan alquilado una prenda talla 'M'

```
SELECT c.nombre, p.talla  
FROM Clientes c  
JOIN Alquileres a ON c.id_cliente = a.id_cliente  
JOIN Prendas p ON a.id_prenda = p.id_prenda  
WHERE p.talla = 'M';
```

8. Mostrar prendas que nunca han sido alquiladas

```
SELECT *  
FROM PRENDAS  
WHERE id_prenda NOT IN (  
    SELECT id_prenda  
    FROM Alquileres  
)
```

9. Mostrar eventos que no tienen alquileres asociados

```
SELECT * FROM EVENTOS  
WHERE id_evento NOT IN(  
    SELECT id_evento  
    FROM Alquileres  
)
```

10. Mostrar clientes que han solicitado recomendaciones y estas hayan sido aceptadas

```
SELECT c.nombre, c.suscripcion, r.estado AS recomendacion  
FROM Clientes c  
JOIN Solicita s ON c.id_cliente = s.id_cliente  
JOIN Recomendaciones r ON r.id_recomendacion = s.id_recomendacion  
WHERE r.estado = 'aceptada' AND c.suscripcion = 'premium';
```

11. Mostrar el nombre del cliente y el número total de alquileres realizados

```
SELECT c.nombre, COUNT(id_alquiler) AS total_alquileres  
FROM Clientes c  
JOIN Alquileres a ON c.id_cliente = a.id_cliente;
```

12. Mostrar los clientes registrados en 2025

```
SELECT *  
FROM Clientes  
WHERE YEAR(fecha_registro) = 2025;
```

13. Mostrar todos los eventos cuyo dress code sea formal

```
SELECT *  
FROM Eventos  
WHERE dress_code = 'Formal';
```

14. Mostrar el número de prendas disponibles por talla

```
SELECT talla, count(*) AS cantidad
FROM PRENDAS
WHERE estado = 'disponible' AND stock > 0
GROUP BY talla;
```

15. Mostrar el número de clientes registrados

```
SELECT count(*) AS total_clientes
FROM Clientes;
```

16. Mostrar los estilistas cuya especialidad sean las bodas

```
SELECT *
FROM Estilistas
WHERE especialidad = 'Bodas';
```

17. Mostrar el promedio de la duración de un alquiler (en días)

```
SELECT AVG(datediff(fin,inicio)) duracion_promedia
FROM Alquileres;
```

18. Mostrar los clientes y la cantidad de recomendaciones de cada uno

```
SELECT c.nombre, count(s.id_recomendacion) AS total_recomendaciones
FROM Clientes c
JOIN Sigue s ON c.id_cliente = s.id_cliente
GROUP BY c.id_cliente;
```

19. Mostrar los eventos que ocurren en junio de 2025

```
SELECT *
FROM Eventos
WHERE MONTH(fecha) = 6 AND YEAR(fecha) = 2025;
```

20. Mostrar los eventos cuyo clima esperado sea Nublado

```
SELECT *
FROM Eventos
WHERE clima Esperado = 'Nublado';
```

21. Mostrar la ciudad y la cantidad de clientes que viven en ella

```
SELECT ciudad, count(*) as total_clientes
FROM Clientes
GROUP BY ciudad;
```

22. Mostrar los clientes que hayan alquilado prendas de la marca 'Zara' y la cantidad de prendas de esa marca

```
SELECT c.nombre, p.marca, count(p.id_prenda) as total_prendas
FROM Clientes c
JOIN Alquileres a ON a.id_cliente = c.id_cliente
JOIN Prendas p ON p.id_prenda = a.id_prenda
WHERE p.marca = 'Zara';
```

23. Mostrar los estilos utilizados en los alquileres

```
SELECT DISTINCT e.descripcion
FROM Estilos e
JOIN Alquileres a ON a.id_estilo = e.id_estilo;
```

24. Mostrar eventos con el total de prendas alquiladas de cada uno

```
SELECT e.tipo, count(a.id_prenda) as total_prendas
FROM Eventos e
JOIN Alquileres a ON a.id_evento = e.id_evento
GROUP BY e.id_evento;
```

25. Mostrar clientes que nunca hayan realizado un alquiler

```
SELECT *
FROM Clientes
WHERE id_cliente NOT IN (
    SELECT id_cliente
    FROM Alquileres
);
```

26. Mostrar los estilistas y la cantidad de recomendaciones que ha emitido cada uno

```
SELECT e.nombre, count(r.id_recomendacion) as total_recomendaciones
FROM Estilistas e
JOIN Recomendaciones r ON e.id_estilista = r.id_estilista
GROUP BY e.nombre;
```

27. Mostrar todos los estilistas registrados y su especialidad

```
SELECT nombre, especialidad  
FROM Estilistas;
```

28. Mostrar todas las prendas en la talla 'L'

```
SELECT *  
FROM Prendas  
WHERE talla = 'L';
```

29. Mostrar todas las recomendaciones aceptadas

```
SELECT *  
FROM Recomendaciones  
WHERE estado = 'aceptada';
```

30. Mostrar todas las graduaciones programadas para junio 2025

```
SELECT *  
FROM EVENTOS  
WHERE tipo = 'Graduación' AND MONTH(fecha) = 6 AND YEAR(fecha) = 2025;
```

4.2 - Creación de un procedimiento almacenado

```
124      DELIMITER //  
125  
126  • CREATE PROCEDURE RegistrarCliente(  
127      IN nombre_param VARCHAR(100),  
128      IN correo_param VARCHAR(100),  
129      IN telefono_param CHAR(9),  
130      IN ciudad_param VARCHAR(100),  
131      IN direccion_param VARCHAR(250),  
132      IN suscripcion_param ENUM('gratuito', 'premium')  
133  )  
134  BEGIN  
135  
136  IF EXISTS (  
137      SELECT *  
138      FROM Clientes  
139      WHERE correo = correo_param  
140  ) THEN  
141      signal sqlstate '45000'  
142      set message_text = 'El correo ya existe';  
143  END IF;  
144  
145  INSERT INTO Clientes(nombre, correo, telefono, ciudad, direccion, suscripcion)  
146  VALUES(nombre_param, correo_param, telefono_param, ciudad_param, direccion_param, suscripcion_param);  
147  END //  
148  
149  DELIMITER ;  
150  
151  • CALL RegistrarCliente(  
152      'Carlos Ruiz',  
153      'carlosr@gmail.com',  
154      '612345678',  
155      'Sevilla',  
156      'Calle Mayor 20',  
157      'gratuito'  
158  );
```