



Tarea de Evaluación 3t

Alejandro Garcia-Mauriño
Salas



Parte de Base de Datos (explicación para presentación)

1. Selección del SGBD (Sistema de Gestión de Base de Datos)

Para este proyecto he elegido MySQL como sistema de gestión de base de datos. Es un sistema muy utilizado, gratuito, fácil de integrar con Java (a través de JDBC), y suficiente para un proyecto de biblioteca como este.

2. Diseño del Esquema Relacional

Creamos una tabla principal llamada libros, normalizada para representar los libros de una biblioteca. Incluye los siguientes campos:

```
1 CREATE TABLE libros (  
2   Id VARCHAR(10) PRIMARY KEY,  
3   Título VARCHAR(100),  
4   Autor VARCHAR(100),  
5   Editorial VARCHAR(100),  
6   Año INT,  
7   Disponible BOOLEAN  
8 );
```

Limpiar Formato Obtener consulta almacenada automáticamente

☐ Vincular parámetros

Guardar esta consulta en favoritos:

Delimitador ; ☐ Mostrar esta consulta otra vez ☐ Mantener la caja de texto con la consulta ☐ Deshacer («rollback») al finalizar ☒ Habilite la revisión de las claves for

3. Inserción de Datos de Ejemplo (DML)



Para comprobar que la base de datos funciona correctamente y que las consultas devuelven datos reales, he insertado ejemplos de libros en la tabla libros utilizando sentencias INSERT. He añadido 7 libros de diferentes géneros como ciencia ficción, terror, aventuras, clásicos y fantasía.

```
1 INSERT INTO libros (Id, Titulo, Autor, Editorial, Año, Disponible) VALUES
2 ('L001', 'Cien Años de Soledad', 'Gabriel García Márquez', 'Sudamericana', 1967, TRUE),
3 ('L002', '1984', 'George Orwell', 'Secker & Warburg', 1949, TRUE),
4 ('L003', 'El Principito', 'Antoine de Saint-Exupéry', 'Reynal & Hitchcock', 1943, FALSE),
5 ('L004', 'It', 'Stephen King', 'Viking Press', 1986, TRUE),
6 ('L005', 'Dune', 'Frank Herbert', 'Chilton Books', 1965, TRUE),
7 ('L006', 'Frankenstein', 'Mary Shelley', 'Lackington, Hughes', 1818, FALSE),
8 ('L007', 'Harry Potter y la piedra filosofal', 'J.K. Rowling', 'Bloomsbury', 1997, TRUE);|
```

Limpiar Formato Obtener consulta almacenada automáticamente

☐ Vincular parámetros

Guardar esta consulta en favoritos:

Delimitador ☐ Mostrar esta consulta otra vez ☐ Mantener la caja de texto con la consulta ☐ Deshacer («rollback») al finalizar ☒ Habilite la revisión

4. Procedimientos Almacenados (versión simplificada para presentación)

Hemos creado dos procedimientos que sirven para gestionar si un libro está disponible o no, sin repetir código.

Prestar libro



```
1 DELIMITER //  
2 CREATE PROCEDURE prestar_libro(IN libro_id VARCHAR(10))  
3 BEGIN  
4     UPDATE libros  
5     SET Disponible = FALSE  
6     WHERE Id = libro_id;  
7 END //  
8 DELIMITER ;  
9
```

Detalles

Nombre de rutina

Tipo

PROCEDURE

[Cambiar a FUNCTION](#)

Parámetros

Dirección	Nombre	Tipo	Longitud/Valores
IN	libro_id	VARCHAR	10

[Agregar parámetro](#)

[Eliminar último parámetro](#)

Definición

```
BEGIN  
    UPDATE libros  
    SET Disponible = TRUE  
    WHERE Id = libro_id;  
END
```

Devolver libro



```
1 CREATE PROCEDURE devolver_libro(IN libro_id VARCHAR(10))
2 BEGIN
3     UPDATE libros
4     SET Disponible = TRUE
5     WHERE Id = libro_id;
6 END;
7
```

Nombre de rutina

prestar_libro

Tipo

PROCEDURE

Cambiar a FUNCTION

Parámetros

Dirección	Nombre	Tipo
↑ IN ▼	libro_id	VARCHAR

Agregar parámetro

Eliminar último parámetro

Definición

```
BEGIN
    UPDATE libros
    SET Disponible = FALSE
    WHERE Id = libro_id;
END
```

Es determinístico

☐



Ajustar privilegios

☒



Procedimientos realizados:

Rutinas

☐ Seleccionar todo  Exportar  Eliminar

	Nombre	Tipo	Retorna	
<input type="checkbox"/>	devolver_libro	PROCEDURE		 Editar  Ejecutar  Exportar  Eliminar
<input type="checkbox"/>	prestar_libro	PROCEDURE		 Editar  Ejecutar  Exportar  Eliminar

5. Disparadores (Triggers)

Creemos un trigger que guarda un historial cada vez que se elimina un libro:

Tabla para guardar historial:

```
1 CREATE TABLE historial_eliminaciones (  
2     Id VARCHAR(10),  
3     Titulo VARCHAR(100),  
4     FechaEliminacion TIMESTAMP DEFAULT CURRENT_TIMESTAMP  
5 );
```

Aquí se puede ver la tabla creada con las especificaciones que le decimos que son Id,Titulo, FechaEliminación.



✓ MySQL ha devuelto un conjunto de valores vacío (es decir: cero columnas). (La consulta tardó 0,0002 segundos.)

```
SELECT * FROM `historial_eliminaciones`
```

☐ Perfilando [[Editar en línea](#)] [[Editar](#)] [[Explicar SQL](#)] [[Crear código PHP](#)] [[Actualizar](#)]

Id	Titulo	FechaEliminacion
----	--------	------------------

Operaciones sobre los resultados de la consulta

Crear vista

Guardar esta consulta en favoritos

Etiqueta: ☐ Permitir que todo usuario pueda acceder a este favorito

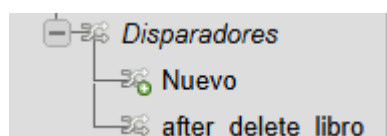
[Guardar esta consulta en favoritos](#)

Trigger asociado con la tabla creada:

```
1 DELIMITER //
2 CREATE TRIGGER after_delete_libro
3 AFTER DELETE ON libros
4 FOR EACH ROW
5 BEGIN
6     INSERT INTO historial_eliminaciones (Id, Titulo)
7     VALUES (OLD.Id, OLD.Titulo);
8 END //
9 DELIMITER ;
```

Esto permite conservar un registro de los libros eliminados.

Aquí se puede ver el Trigger creado correctamente:





Disparadores

<input type="checkbox"/>	Seleccionar todo	Exportar	Eliminar
Nombre	Tiempo	Evento	
<input type="checkbox"/>	after_delete_libro	AFTER DELETE	Editar Exportar Eliminar

Aquí se puede editar el Trigger:

Editar disparador

Detalles

Nombre del disparador	after_delete_libro
Tabla	libros
Tiempo	AFTER
Evento	DELETE
Definición	<pre>BEGIN INSERT INTO historial_eliminaciones (Id, Titulo) VALUES (OLD.Id, OLD.Titulo); END</pre>
Definidor	root@localhost

6. Modelado de Datos Complejos

Como MySQL no permite tipos objetos, representamos relaciones complejas usando una tabla generos, donde un libro puede tener varios géneros:



```
1 CREATE TABLE generos (  
2     IdLibro VARCHAR(10),  
3     Genero VARCHAR(50),  
4     FOREIGN KEY (IdLibro) REFERENCES libros(Id)  
5 );
```

Nombre del índice: [?](#)

PRIMARY

Opción de índice: [?](#)

PRIMARY

Opciones avanzadas

Columna
<div><div></div><div>Id [varchar(30)]</div></div>

☐ Agregar 1 columna(s) al índice

[Continuar](#) [Previsualizar SQL](#)

Ejemplo de inserción de géneros:

```
1 INSERT INTO generos (IdLibro, Genero) VALUES  
2 ('L001', 'Realismo Mágico'),  
3 ('L002', 'Distopía'),  
4 ('L004', 'Terror'),  
5 ('L004', 'Ciencia Ficción'),  
6 ('L005', 'Ciencia Ficción'),  
7 ('L006', 'Terror'),  
8 ('L007', 'Ciencia Ficción');
```



Esto representa una relación de uno a muchos entre libros y géneros.

Podemos visualizar esta tabla desde phpMyAdmin con esta consulta:

```
1 SELECT * FROM generos;
```

✓ Mostrando filas 0 - 6 (total de 7, La consulta tardó 0,0002 segundos.)

SELECT * FROM generos;

☐ Perfilando [[Editar en línea](#)] [[Editar](#)] [[Explicar SQL](#)] [[Crear código PHP](#)] [[Actualizar](#)]

☐ Mostrar todo | Número de filas: 25 ▼ | Filtrar filas: | Ordenar según la clave:

Opciones extra

IdLibro	Genero
L001	Realismo Mágico
L002	Distopía
L004	Terror
L004	Ciencia Ficción
L005	Ciencia Ficción
L006	Terror
L007	Ciencia Ficción