



Desarrollo de una Aplicación Java con Acceso a Base a Datos PostgreSQL “Planificación de cartelera para un cine”

Integrantes:

Antonia Paredes (21194240121)

Leandro Matamoros (20978573021)

Docente: Ania Cravero

Asignatura: Bases de datos

Temuco, 24 de junio de 2024

Índice

1. Introducción.....	3
2. Caso de estudio.....	4
3. Diseño de la base de datos.....	5
4. Implementación de la base de datos.....	7
5. Implementación de CRUD en java.....	9
6. Manejo de concurrencia y transacciones.....	13
7. Conclusión.....	15

1. Introducción

Una base de datos bien estructurada es un pilar fundamental para simplificar numerosos procesos transaccionales en cualquier organización. Desde el registro eficiente de ventas hasta la gestión óptima de reservas y la planificación detallada de horarios, una adecuada gestión de datos no solo mejora la operatividad, sino que también optimiza recursos y tiempos.

Para evaluar la eficacia de una aplicación con acceso a base de datos, se examinó un caso específico: la compleja planificación de proyecciones en un cine. Este escenario implica la asignación precisa de horarios y salas a las diversas películas disponibles.

La propuesta de esta aplicación busca mejorar y simplificar la planificación de las proyecciones cinematográficas, optimizando el uso de los recursos disponibles y maximizando la eficiencia operativa del cine.

2. Caso de estudio

Un sistema de gestión de cartelera es esencial para administrar eficazmente la programación de películas en un cine.

El caso de estudio se centra en diseñar y desarrollar una base de datos para dicho cine. La base de datos contendrá información sobre las películas, las salas de cine y los horarios de proyección.

Cada película será caracterizada por atributos como título, género, duración y clasificación. Las salas de cine serán identificadas mediante un número único, y se registrarán las capacidades de asientos de cada una.

Para los horarios de proyección (función), se registrará la fecha y hora de inicio de la película junto con la sala en la que se llevará a cabo la proyección.

El objetivo principal será la planificación de funciones (Transacción), por lo que se desarrollará una aplicación que facilite el manejo CRUD (Crear, Leer, Actualizar y Eliminar) de la tabla asociada a estas funciones. Esto permitirá una gestión eficiente y organizada de la programación de películas conforme a las necesidades del cine.

3. Diseño de la base de datos

Para el desarrollo de la base de datos, se identificaron 4 principales tablas/entidades, y sus respectivos atributos:

1. Películas: En esta tabla se almacenan todas las películas disponibles para proyectar, y sus datos relevantes, tales como:
 - Id_película (PK)
 - Título
 - Género
 - Duración
 - Clasificación (FK)
2. Clasificaciones: Las diferentes categorías de clasificación de una película, según la necesidad del cine, pueden usarse por ejemplo las clasificaciones MPAA, BBFC o LSF, los atributos de esta tabla son:
 - Clasificación (PK)
 - Edad mínima
 - Restricción
3. Salas De Cine: Se registran las salas existentes en el cine y su capacidad (aforo).
 - Id_sala (PK)
 - Capacidad
4. Horarios De Proyección (Funciones): Representa la 'transacción' de la base de datos, es la planificación de una función a proyectar, se incluyen los siguientes atributos:
 - Id_función (PK)
 - Id_película (FK)
 - Id_sala (FK)
 - Hora de inicio

Para visualizar el diseño de la base de datos de mejor manera, se realiza el siguiente diagrama Entidad-Relación (ER):

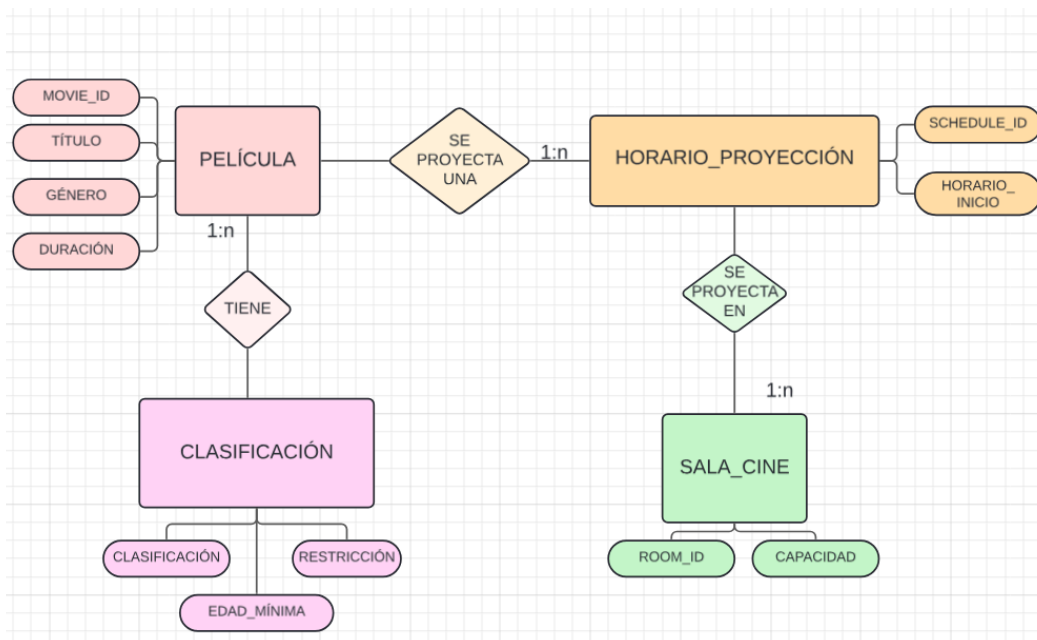
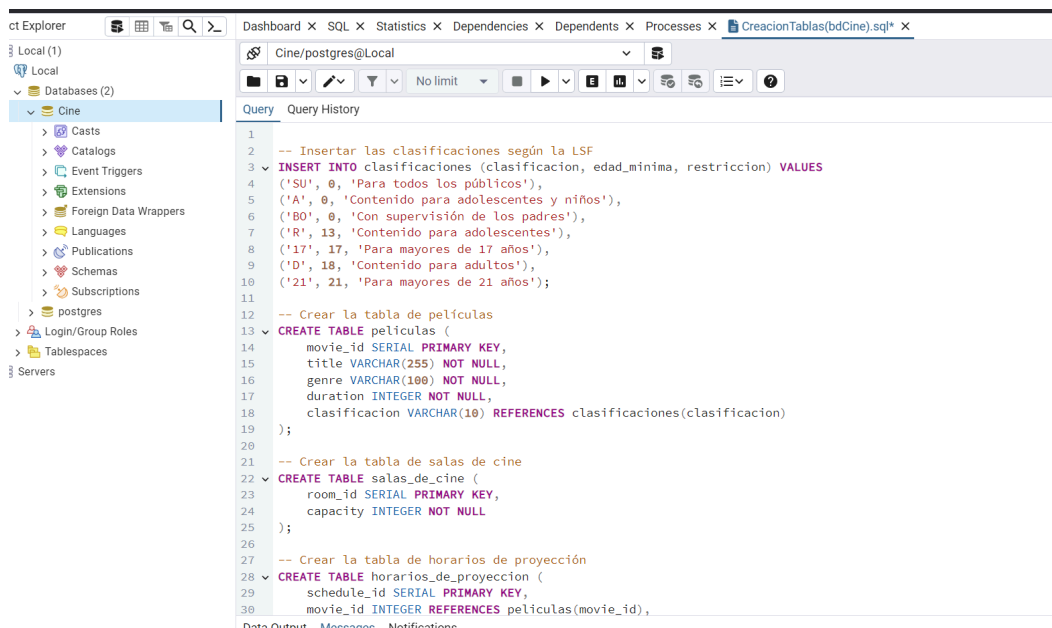


Figura n°1: 'Diagrama ER' (Elaboración propia)

4. Implementación de la base de datos

Se implementa la base de datos PostgreSQL mediante un script SQL, además se insertan datos de muestra para hacer posterior uso de la base de datos.

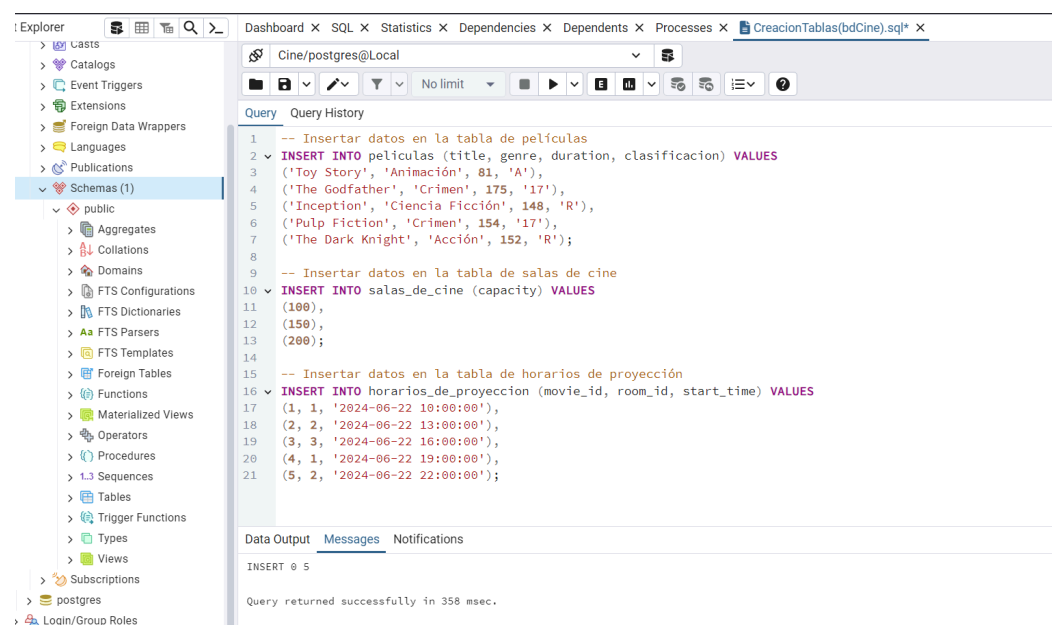
(Todo el script SQL utilizado se encuentra en el GitHub anexado al final del informe).



The screenshot shows the pgAdmin interface with the 'Cine' database selected in the left sidebar. The main pane displays a SQL script for creating tables and inserting data. The script includes comments in Spanish and SQL commands for creating three tables: 'clasificaciones', 'peliculas', and 'salas_de_cine', along with inserting sample data into them.

```
1
2
3 -- Insertar las clasificaciones según la LSF
4 INSERT INTO clasificaciones (clasificacion, edad_minima, restriccion) VALUES
5 ('SU', 0, 'Para todos los públicos'),
6 ('A', 0, 'Contenido para adolescentes y niños'),
7 ('B0', 0, 'Con supervisión de los padres'),
8 ('R', 13, 'Contenido para adolescentes'),
9 ('17', 17, 'Para mayores de 17 años'),
10 ('D', 18, 'Contenido para adultos'),
11 ('21', 21, 'Para mayores de 21 años');
12
13 -- Crear la tabla de películas
14 CREATE TABLE peliculas (
15     movie_id SERIAL PRIMARY KEY,
16     title VARCHAR(255) NOT NULL,
17     genre VARCHAR(100) NOT NULL,
18     duration INTEGER NOT NULL,
19     clasificacion VARCHAR(10) REFERENCES clasificaciones(clasificacion)
20 );
21
22 -- Crear la tabla de salas de cine
23 CREATE TABLE salas_de_cine (
24     room_id SERIAL PRIMARY KEY,
25     capacity INTEGER NOT NULL
26 );
27
28 -- Crear la tabla de horarios de proyección
29 CREATE TABLE horarios_de_proyeccion (
30     schedule_id SERIAL PRIMARY KEY,
31     movie_id INTEGER REFERENCES peliculas(movie_id),
32     room_id INTEGER REFERENCES salas_de_cine(room_id),
33     start_time TIME NOT NULL
34 );
```

Figura n°2: 'Creación de tablas en la base de datos'



The screenshot shows the pgAdmin interface with the 'Cine' database selected. The main pane displays a SQL script for inserting sample data into the tables created in the previous figure. The script includes comments in Spanish and SQL commands for inserting data into 'peliculas', 'salas_de_cine', and 'horarios_de_proyeccion'.

```
1
2 -- Insertar datos en la tabla de películas
3 INSERT INTO peliculas (title, genre, duration, clasificacion) VALUES
4 ('Toy Story', 'Animación', 81, 'A'),
5 ('The Godfather', 'Crimen', 175, '17'),
6 ('Inception', 'Ciencia Ficción', 148, 'R'),
7 ('Pulp Fiction', 'Crimen', 154, '17'),
8 ('The Dark Knight', 'Acción', 152, 'R');
9
10 -- Insertar datos en la tabla de salas de cine
11 INSERT INTO salas_de_cine (capacity) VALUES
12 (100),
13 (150),
14 (200);
15
16 -- Insertar datos en la tabla de horarios de proyección
17 INSERT INTO horarios_de_proyeccion (movie_id, room_id, start_time) VALUES
18 (1, 1, '2024-06-22 10:00:00'),
19 (2, 2, '2024-06-22 13:00:00'),
20 (3, 3, '2024-06-22 16:00:00'),
21 (4, 1, '2024-06-22 19:00:00'),
22 (5, 2, '2024-06-22 22:00:00');
```

Below the script, the 'Data Output' tab shows the results of the execution: 'INSERT 0 5' and 'Query returned successfully in 358 msec.'

Figura n°3: 'Inserción de datos de referencia'

Query

Query History

1

SELECT

2

hp.schedule_id,

3

p.title AS movie_title,

4

p.genre AS movie_genre,

5

p.duration AS movie_duration,

6

c.clasificacion AS movie_clasificacion,

7

c.restriccion AS movie_restriccion,

8

sc.room_id,

9

sc.capacity AS room_capacity,

10

hp.start_time

11

FROM

12

horarios_de_proyeccion hp

13

JOIN

14

peliculas p ON hp.movie_id = p.movie_id

15

JOIN

16

clasificaciones c ON p.clasificacion = c.clasificacion

17

JOIN

18

salas_de_cine sc ON hp.room_id = sc.room_id

19

ORDER BY

20

hp.start_time;

Scratch Pad

Data Output

Messages

Notifications

schedule_id

integer

movie_title

character varying (255)

movie_genre

character varying (100)

movie_duration

integer

movie_clasificacion

character varying (10)

movie_restriccion

character varying (255)

room_id

integer

room_capacity

integer

start_time

timestamp without time zone

1

1

Toy Story

Animación

81

A

Contenido para adolescentes y niños

1

100

2024-06-22 10:00:00

2

2

The Godfather

Crimen

175

17

Para mayores de 17 años

2

150

2024-06-22 13:00:00

3

3

Inception

Ciencia Ficción

148

R

Contenido para adolescentes

3

200

2024-06-22 16:00:00

Figura n°4: 'Visualización de Funciones creadas'

Vemos que luego de ejecutar todas las consultas vemos que las funciones de las películas se muestran correctamente con todos los datos

5. Implementación de CRUD en java

Se implementó una aplicación CRUD en java, que permite gestionar los horarios de la cartelera del cine, pudiendo crear, modificar, eliminar y ver las diferentes funciones disponibles.

A continuación se muestra el menú de la aplicación, seguido de los diferentes ejemplos de uso de esta.

```
Cine MOCKITA
/\_/\
( o.o )
> ^ <
---MENU---
1. Crear función
2. Ver cartelera
3. Actualizar función
4. Eliminar función
5. Salir
```

Figura nº5 : 'Menú principal'

```
Cine MOCKITA
/\_/\
( o.o )
> ^ <
---MENU---
1. Crear función
2. Ver cartelera
3. Actualizar función
4. Eliminar función
5. Salir
Elige una opción: 2
ID Función: 1 | Película: Toy Story | Sala: 1 | Fecha y hora: 2024-06-22 10:00:00.0
ID Función: 2 | Película: The Godfather | Sala: 2 | Fecha y hora: 2024-06-22 13:00:00.0
ID Función: 3 | Película: Inception | Sala: 3 | Fecha y hora: 2024-06-22 16:00:00.0
ID Función: 4 | Película: Pulp Fiction | Sala: 1 | Fecha y hora: 2024-06-22 19:00:00.0
ID Función: 5 | Película: The Dark Knight | Sala: 2 | Fecha y hora: 2024-06-22 22:00:00.0
Cine MOCKITA
```

Figura nº6 : 'Opción número 2: Ver cartelera'

Comparando las figuras 4 y 6, se puede comprobar que los resultados son fieles a la base de datos con la que se está trabajando, demostrando que la conexión del programa java es correcta.

```
---MENU---
1. Crear función
2. Ver cartelera
3. Actualizar función
4. Eliminar función
5. Salir
Elige una opción: 1

Películas disponibles:
1 - Toy Story
2 - The Godfather
3 - Inception
4 - Pulp Fiction
5 - The Dark Knight

Ingrese el ID de la película: 5

Salas disponibles:
1 - Capacidad: 100
2 - Capacidad: 150
3 - Capacidad: 200

Ingrese el ID de la sala: 2
Fecha y hora de inicio (YYYY-MM-DD HH:MM:SS): 2024-12-12 20:50:00
Función creada exitosamente.
```

Figura nº7: ‘Opción número 1 : Crear función’
(Se agrega una programación a la cartelera de cine)

```
---MENU---
1. Crear función
2. Ver cartelera
3. Actualizar función
4. Eliminar función
5. Salir
Elige una opción: 2
ID Función: 1 | Película: Toy Story | Sala: 1 | Fecha y hora: 2024-06-22 10:00:00.0
ID Función: 2 | Película: The Godfather | Sala: 2 | Fecha y hora: 2024-06-22 13:00:00.0
ID Función: 3 | Película: Inception | Sala: 3 | Fecha y hora: 2024-06-22 16:00:00.0
ID Función: 4 | Película: Pulp Fiction | Sala: 1 | Fecha y hora: 2024-06-22 19:00:00.0
ID Función: 5 | Película: The Dark Knight | Sala: 2 | Fecha y hora: 2024-06-22 22:00:00.0
ID Función: 6 | Película: The Dark Knight | Sala: 2 | Fecha y hora: 2024-12-12 20:50:00.0
```

Figura nº8: ‘Se comprueba que la función fue agregada con éxito’

schedule_id integer	movie_title character varying (255)	movie_genre character varying (100)	movie_duration integer	movie_clasificacion character varying (10)	movie_restriccion character varying (255)	room_id integer
1	Toy Story	Animación	81	A	Contenido para adolescentes y niños	1
2	The Godfather	Crimen	175	17	Para mayores de 17 años	2
3	Inception	Ciencia Ficción	148	R	Contenido para adolescentes	3
4	Pulp Fiction	Crimen	154	17	Para mayores de 17 años	1
5	The Dark Knight	Acción	152	R	Contenido para adolescentes	2
6	The Dark Knight	Acción	152	R	Contenido para adolescentes	2

Figura nº9: ‘Se comprueba que la base de datos fue modificada’

```
Elige una opción: 3

Funciones disponibles:
ID Función: 1, Película: Toy Story, Sala ID: 1, Fecha y hora: 2024-06-22 10:00:00.0
ID Función: 2, Película: The Godfather, Sala ID: 2, Fecha y hora: 2024-06-22 13:00:00.0
ID Función: 3, Película: Inception, Sala ID: 3, Fecha y hora: 2024-06-22 16:00:00.0
ID Función: 4, Película: Pulp Fiction, Sala ID: 1, Fecha y hora: 2024-06-22 19:00:00.0
ID Función: 5, Película: The Dark Knight, Sala ID: 2, Fecha y hora: 2024-06-22 22:00:00.0
ID Función: 6, Película: The Dark Knight, Sala ID: 2, Fecha y hora: 2024-12-12 20:50:00.0

ID de la función a actualizar: 6

Películas disponibles:
1 - Toy Story
2 - The Godfather
3 - Inception
4 - Pulp Fiction
5 - The Dark Knight

Nuevo ID de película: 1

Salas disponibles:
1 - Capacidad: 100
2 - Capacidad: 150
3 - Capacidad: 200

Nuevo ID de sala: 2
Nueva fecha y hora (YYYY-MM-DD HH:MM:SS): 2024-12-12 20:50:00
Función actualizada exitosamente.
```

Figura nº10: ‘Opción número 3: Actualizar función’
(Modifica una función existente)

schedule_id integer	movie_title character varying (255)	movie_genre character varying (100)	movie_duration integer	movie_clasificacion character varying (10)	movie_restriccion character varying (255)	room_id integer	room_capacity integer	start_time timestamp without time zone
1	Toy Story	Animación	81	A	Contenido para adolescentes y niños	1	100	2024-06-22 10:00:00
2	The Godfather	Crimen	175	17	Para mayores de 17 años	2	150	2024-06-22 13:00:00
3	Inception	Ciencia Ficción	148	R	Contenido para adolescentes	3	200	2024-06-22 16:00:00
4	Pulp Fiction	Crimen	154	17	Para mayores de 17 años	1	100	2024-06-22 19:00:00
5	The Dark Knight	Acción	152	R	Contenido para adolescentes	2	150	2024-06-22 22:00:00
6	Toy Story	Animación	81	A	Contenido para adolescentes y niños	2	150	2024-12-12 20:50:00

Figura nº11: “Se observa el cambio en la base de datos”

```

Elige una opción: 4

Funciones disponibles:
ID Función: 1, Película: Toy Story, Sala ID: 1, Fecha y hora: 2024-06-22 10:00:00.0
ID Función: 2, Película: The Godfather, Sala ID: 2, Fecha y hora: 2024-06-22 13:00:00.0
ID Función: 3, Película: Inception, Sala ID: 3, Fecha y hora: 2024-06-22 16:00:00.0
ID Función: 4, Película: Pulp Fiction, Sala ID: 1, Fecha y hora: 2024-06-22 19:00:00.0
ID Función: 5, Película: The Dark Knight, Sala ID: 2, Fecha y hora: 2024-06-22 22:00:00.0
ID Función: 6, Película: Toy Story, Sala ID: 2, Fecha y hora: 2024-12-12 20:50:00.0

ID de la función a eliminar: 6
Función eliminada exitosamente.

```

Figura nº12: ‘Opción número 4: Eliminar función’

schedule_id integer	movie_title character varying (255)	movie_genre character varying (100)	movie_duration integer	movie_clasificacion character varying (10)	movie_restriccion character varying (255)	room_id integer	room_capacity integer	start_time timestamp without time zone
1	Toy Story	Animación	81	A	Contenido para adolescentes y niños	1	100	2024-06-22 10:00:00
2	The Godfather	Crimen	175	17	Para mayores de 17 años	2	150	2024-06-22 13:00:00
3	Inception	Ciencia Ficción	148	R	Contenido para adolescentes	3	200	2024-06-22 16:00:00
4	Pulp Fiction	Crimen	154	17	Para mayores de 17 años	1	100	2024-06-22 19:00:00
5	The Dark Knight	Acción	152	R	Contenido para adolescentes	2	150	2024-06-22 22:00:00

Figura nº13: ‘Se observa la base de datos’
(Se comprueba que la función fue eliminada con éxito)

```

Cine MOCKITA
 /\_/\
( o.o )
 > ^ <
---MENU---
1. Crear función
2. Ver cartelera
3. Actualizar función
4. Eliminar función
5. Salir
Elige una opción: 5
Conexión cerrada. Adiós!

```

Figura nº14: ‘Opción nº5: Salir’
(Finaliza la ejecución de la aplicación)

6. Manejo de concurrencia y transacciones

Control de concurrencia por bloqueo: Se utiliza el bloqueo de filas (row-level locking) con la cláusula "FOR UPDATE" en las operaciones de inserción, actualización y eliminación. Esto previene que múltiples transacciones modifiquen los mismos datos simultáneamente.

Este mecanismo asegura que no se creen funciones duplicadas o conflictivas en caso de que múltiples usuarios intenten crear funciones simultáneamente para la misma película, sala y horario.

```
String lockSql = "SELECT * FROM horarios_de_proyeccion WHERE movie_id = ? AND room_id = ? AND start_time = ? FOR UPDATE";
try (PreparedStatement lockStmt = conn.prepareStatement(lockSql)) {
    lockStmt.setInt( parameterIndex: 1, movieId);
    lockStmt.setInt( parameterIndex: 2, roomId);
    lockStmt.setTimestamp( parameterIndex: 3, Timestamp.valueOf(startTime));
    ResultSet rs = lockStmt.executeQuery();
}
```

Figura n°15: 'Control de concurrencia para crear una función'

El control de concurrencia está presente no solo para crear una función, sino también para modificar y eliminar una de ellas.

Adicionalmente, se implementa el manejo de transacciones como un mecanismo complementario para el control de concurrencia:

1. Inicio de transacción: Se utiliza `conn.setAutoCommit(false)` para iniciar una transacción manual.

```
conn.setAutoCommit(false);
```

Figura n°16: 'Línea de código'

2. Operaciones atómicas: Todas las operaciones dentro de la transacción se agrupan como una unidad lógica de trabajo.
3. Finalización de transacción: Se usa `conn.commit()` para confirmar los cambios o `conn.rollback()` para revertirlos en caso de error.

```
        conn.commit();
    } catch (SQLException e) {
        try {
            conn.rollback();
        } catch (SQLException rollbackEx) {
            System.out.println("Error al hacer rollback: " + rollbackEx.getMessage());
        }
        System.out.println("Error al crear la función: " + e.getMessage());
    }
```

Figura n°17: 'Manejo de excepciones'

Este enfoque transaccional contribuye al manejo de concurrencia de las siguientes maneras:

1. Asegura la consistencia de los datos al completar todas las operaciones relacionadas como una unidad atómica.
2. Proporciona aislamiento, evitando que otras transacciones vean cambios parciales o inconsistentes.
3. Permite la resolución de conflictos mediante rollback y reintento en caso de problemas de concurrencia.
4. Garantiza la durabilidad de los cambios una vez confirmada la transacción.
5. Facilita el manejo de errores, permitiendo revertir cambios en caso de fallos durante la ejecución.

La combinación de bloqueos a nivel de fila y el manejo de transacciones proporciona un robusto sistema de control de concurrencia, asegurando la integridad de los datos en un entorno multiusuario.

7. Conclusión

El desarrollo de una aplicación para la gestión de la cartelera cinematográfica utilizando Java (API JDBC) y PostgreSQL ha sido un ejercicio integral que ha permitido aplicar y aprender conceptos fundamentales de bases de datos.

La implementación del sistema de gestión de funciones ha proporcionado una plataforma robusta para la creación, actualización, eliminación y visualización de horarios de proyección (CRUD), facilitando así una planificación detallada y eficiente de las películas disponibles.

Además, la integración de mecanismos de control de concurrencia mediante el uso de bloqueos de filas y transacciones ha asegurado la integridad de los datos y evitado conflictos en situaciones de acceso concurrente, garantizando la consistencia de la información en todo momento para los usuarios.

En resumen, este proyecto ha proporcionado una experiencia práctica invaluable en el diseño y desarrollo de sistemas de gestión de bases de datos aplicados a situaciones del mundo real, destacando la importancia de la planificación y ejecución precisa en la implementación de soluciones tecnológicas que mejoran significativamente la operatividad y eficiencia de las organizaciones.

ANEXOS:

- Link de acceso al repositorio del proyecto, donde se encuentran el script SQL y el programa Java elaborado:
<https://github.com/LeandroEsteban/TareaAccesoDatos> .