ЛАБОРАТОРНА РОБОТА №3

Схема бази даних

1. Таблиця CUSTOMER (Клієнт)

- **Призначення:** Зберігання інформації про клієнтів кінотеатру
- **Стовпці:**
    - customer_id (SERIAL) - первинний ключ, автоінкремент
    - name (VARCHAR(100)) - ім'я клієнта, NOT NULL
    - email (VARCHAR(100)) - електронна пошта, NOT NULL, UNIQUE
    - phone_number (VARCHAR(15)) - номер телефону, NOT NULL, UNIQUE

2. Таблиця HALL (Зал)

- **Призначення:** Інформація про кінозали
- **Стовпці:**
    - hall_id (SERIAL) - первинний ключ, автоінкремент
    - name (VARCHAR(50)) - назва залу, NOT NULL, UNIQUE
    - total_seats (INT) - загальна кількість місць, NOT NULL, CHECK > 0

3. Таблиця MOVIE (Фільм)

- **Призначення:** Каталог фільмів
- **Стовпці:**
    - movie_id (SERIAL) - первинний ключ, автоінкремент
    - title (VARCHAR(200)) - назва фільму, NOT NULL
    - release_date (DATE) - дата виходу, NOT NULL
    - duration (INT) - тривалість у хвилинах, NOT NULL, CHECK > 0
    - genre (VARCHAR(50)) - жанр, NOT NULL

4. Таблиця SEAT (Місце)

- **Призначення:** Фізичні місця в залах
- **Стовпці:**
    - seat_id (INT) - NOT NULL
    - hall_id (INT) - зовнішній ключ до HALL, NOT NULL
    - row_number (INT) - номер ряду, NOT NULL, CHECK > 0

- seat_number (VARCHAR(10)) - номер місця, NOT NULL

- **Ключі:**
  - Первинний ключ: композитний (seat_id, hall_id)
  - UNIQUE обмеження: (hall_id, seat_number)

5. Таблиця SESSION (Сеанс)

- **Призначення:** Розклад показів фільмів
- **Стовпці:**
  - session_id (SERIAL) - первинний ключ, автоінкремент
  - movie_id (INT) - зовнішній ключ до MOVIE, NOT NULL
  - hall_id (INT) - зовнішній ключ до HALL, NOT NULL
  - session_time (TIMESTAMP) - дата та час сеансу, NOT NULL
- **Ключі:**
  - UNIQUE обмеження: (hall_id, session_time) - запобігає накладанню сеансів

6. Таблиця BOOKING (Бронювання)

- **Призначення:** Замовлення квитків
- **Стовпці:**
  - booking_id (SERIAL) - первинний ключ, автоінкремент
  - customer_id (INT) - зовнішній ключ до CUSTOMER, NOT NULL
  - booking_date (DATE) - дата бронювання, NOT NULL
  - total_amount (NUMERIC(8,2)) - загальна сума, NOT NULL, CHECK >= 0

7. Таблиця TICKET (Квиток)

- **Призначення:** Індивідуальні квитки
- **Стовпці:**
  - ticket_id (SERIAL) - первинний ключ, автоінкремент
  - booking_id (INT) - зовнішній ключ до BOOKING, NOT NULL
  - session_id (INT) - зовнішній ключ до SESSION, NOT NULL
  - hall_id (INT) - частина зовнішнього ключа до SEAT, NOT NULL
  - seat_id (INT) - частина зовнішнього ключа до SEAT, NOT NULL

- price (NUMERIC(5,2)) - ціна квитка, NOT NULL, CHECK > 0

- **Ключі:**
  - Зовнішній ключ: композитний (seat_id, hall_id) до SEAT
  - UNIQUE обмеження: (session_id, seat_id, hall_id) - запобігає подвійному продажу

1.2. Важливі обмеження та припущення

Обмеження цілісності даних:

1. **UNIQUE обмеження:**
   - Кожен клієнт має унікальні email та телефон
   - Назви залів унікальні
   - Комбінація зал+час сеансу унікальна (немає накладання сеансів)
   - Комбінація сеанс+місце+зал унікальна (кожне місце продається один раз на сеанс)
2. **CHECK обмеження:**
   - Кількість місць у залі > 0
   - Тривалість фільму > 0
   - Ціна квитка > 0
   - Сума бронювання >= 0
3. **FOREIGN KEY обмеження з політиками видалення:**
   - ON DELETE CASCADE: Видалення залу → автоматичне видалення місць
   - ON DELETE RESTRICT: Запобігає видаленню записів з залежностями
   - ON DELETE CASCADE: Видалення бронювання → видалення всіх його квитків

Бізнес-припущення:

1. Один клієнт може мати багато бронювань
2. Одне бронювання може містити декілька квитків
3. Один фільм може мати багато сеансів
4. Один зал може приймати багато сеансів (в різний час)
5. Одне фізичне місце може бути продане на різні сеанси, але не на один сеанс двічі

Приклади виконнання запитів:

SELECT * FROM TICKET WHERE price = 125.00;

ticket_id [PK] integer	booking_id integer	session_id integer	hall_id integer	seat_id integer	price numeric (5,2)
1	1	1	1	1	125.00
2	1	1	1	2	125.00
5	3	3	1	3	125.00

INSERT INTO CUSTOMER (name, email, phone_number)

VALUES ('Іван Коваленко', 'ivan.kovalenko@example.com', '380661112233');

SELECT * FROM CUSTOMER WHERE name = 'Іван Коваленко';

customer_id [PK] integer	name character varying (100)	email character varying (100)	phone_number character varying (15)
4	Іван Коваленко	ivan.kovalenko@example.c...	380661112233

SELECT * FROM CUSTOMER WHERE name = 'Олександр Дудник'; UPDATE CUSTOMER SET email = 'oleksandr.new@example.com' WHERE name = 'Олександр Дудник'; SELECT * FROM CUSTOMER WHERE name = 'Олександр Дудник';

customer_id [PK] integer	name character varying (100)	email character varying (100)	phone_number character varying (15)
1	Олександр Дудник	oleksandr.new@example.c...	380971112233

SELECT * FROM TICKET WHERE ticket_id = 6;

DELETE FROM TICKET WHERE ticket_id = 6;

SELECT * FROM TICKET WHERE ticket_id = 6;

ticket_id [PK] integer	booking_id integer	session_id integer	hall_id integer	seat_id integer	price numeric (5,2)

-- Підрахунок записів у кожній таблиці

SELECT 'CUSTOMER' as таблиця, COUNT(*) as записів FROM CUSTOMER

UNION ALL SELECT 'HALL', COUNT(*) FROM HALL

UNION ALL SELECT 'MOVIE', COUNT(*) FROM MOVIE

UNION ALL SELECT 'SEAT', COUNT(*) FROM SEAT

UNION ALL SELECT 'SESSION', COUNT(*) FROM SESSION

UNION ALL SELECT 'BOOKING', COUNT(*) FROM BOOKING

UNION ALL SELECT 'TICKET', COUNT(*) FROM TICKET;

ticket_id,booking_id,session_id,hall_id,seat_id,price

1,1,1,1,1,125.00

2,1,1,1,2,125.00

3,2,2,2,1,150.25

4,2,2,2,2,150.25

5,3,3,1,3,125.00

-- Перегляд основних змін

SELECT '=== CUSTOMER ===' as info; SELECT * FROM CUSTOMER;

SELECT '=== MOVIE ===' as info; SELECT * FROM MOVIE;

SELECT '=== HALL ===' as info; SELECT * FROM HALL;

SELECT '=== SESSION ===' as info; SELECT * FROM SESSION;

SELECT '=== BOOKING ===' as info; SELECT * FROM BOOKING;

SELECT '=== TICKET ===' as info; SELECT * FROM TICKET;

CUSTOMER,4

HALL,3

MOVIE,3

SEAT,7

SESSION,3

BOOKING,3

TICKET,5

ВИСНОВКИ

1. **База даних успішно пройшла тестування OLTP-операцій:**
    - **Всі типи DML-операцій (SELECT, INSERT, UPDATE, DELETE) виконуються коректно**
    - **Обмеження цілісності даних працюють належним чином**
    - **Схема підтримує всі необхідні бізнес-процеси кінотеатру**
2. **Дотримано вимог лабораторної роботи:**
    - **Кожна таблиця містить мінімум 3-5 записів**
    - **Виконано різноманітні SELECT запити з фільтрацією**
    - **Продемонстровано роботу INSERT, UPDATE, DELETE з перевірками**
    - **Показано роботу обмежень FOREIGN KEY на прикладі помилки**
3. **Схема бази даних є повною та цілісною:**
    - **Усі необхідні сутності представлені**
    - **Встановлені правильні зв'язки між таблицями**
    - **Реалізовані всі бізнес-правила через обмеження**