SQL Database Assignment Report

Title: Cinema Theater Database Project Report

Student Name: Pavani Oruganti

Student ID: 24082686

1.Introduction

This project uses SQLite to model a relational database for a movie theater system. The database was created to hold client information, movies, theaters, showtimes, and tickets. With several tables, realistic randomized data, and all four types of data—nominal, ordinal, interval, and ratio—it satisfies the assignment's requirements.
The objective was to develop a schema that could simulate the operation of a genuine movie theatre while also including restrictions and foreign keys to guarantee data integrity.

2.Generation of Data

Python's faker package, which creates plausible names, emails, cities, and dates, was used to generate the data. Randomization was used for characteristics like:
• Age range (12–80 years)
• The cost of tickets (£5.00–£20.00)
• Film durations (80–180 minutes)
• Theaters with 100–300 seats
• Levels of membership (Bronze, Silver, Gold, Platinum)

Deliberate duplicates and missing values were included in order to satisfy the realism requirements:
• Overlapping seat numbers and duplicate client names.
• Some customers' emails are missing.
This guarantees that the dataset captures flaws found in the real world.

3. Textual Representation of Database Schema

There are five tables in the schema:

Code

```
Customers(
    customer_id PK,
    name,
    email UNIQUE,
    membership_level CHECK(Bronze, Silver, Gold, Platinum),
    age CHECK(age >= 0)
)


Movies(
    movie_id PK,
    title,
    genre,
```

duration_minutes CHECK(duration_minutes > 0),
release_year
)

Theaters(
theater_id PK,
name,
capacity CHECK(capacity > 0),
location
)

Showtimes(
showtime_id PK,
movie_id FK → Movies.movie_id,
theater_id FK → Theaters.theater_id,
show_date,
start_time
)

Tickets(
ticket_id PK,
customer_id FK → Customers.customer_id,
showtime_id FK → Showtimes.showtime_id,
seat_number,
price CHECK(price >= 0)
)

Relationships

- **Customers → Tickets** (one-to-many)
- **Movies → Showtimes** (one-to-many)
- **Theaters → Showtimes** (one-to-many)

- **Showtimes → Tickets** (one-to-many)

4. Explanation of Tables

- The data was divided into several tables using the following normalization principles:
- Steer clear of duplications (e.g., movie information is saved once in Movies).
- Foreign keys enforce legitimate references, ensuring integrity.
- Permit scalability: It's simple to add a new film, theater, or patron.

To put logical rules on the data, CHECK, NOT NULL, and UNIQUE constraints were applied. For example, ticket prices ≥ 0, age ≥ 0, and membership levels are four categories.

5. Ethical and Data Privacy Considerations

- Only synthetic data: No actual clients or private information was used.
- Anonymization: Faker-generated names and emails guarantee privacy.
- Data realism without sensitivity: Despite being realistic, the dataset omits sensitive information such as bank records or medical histories.
- Ethical Use: There will be no commercial exploitation of the database; it will only be used for educational reasons.

6. Conclusion

- The development of a realistic, multi-table SQLite database is demonstrated effectively by this project.
- All four forms of data are used.
- The application of constraints and foreign keys.
- The creation of more than a thousand rows of realistic but randomized data.
- Relationships are explained by textual schema representation.

The movie theater database will offer a solid foundation for honing SQL queries, data extraction, and even additional analysis of association rules or consumer behavior insights.