**Movie Ticketing Platform Database Design**

This document outlines a normalized database schema designed to support a movie ticketing platform similar to BookMyShow. The primary goal of this design is to ensure that the data structure is clear, human-readable, and maintains data integrity by following proper normalization practices.

**1. Key Entities and Their Purpose**

**Theatres**

This table stores information about cinema venues.

* **theatre\_id**: Unique identifier
* **name**: Name of the theatre
* **location**: Physical address or area

**Movies**

Represents movies that can be shown at any theatre.

* **movie\_id**: Unique identifier
* **title**: Movie name
* **language**: Language the movie is in

**Formats**

Describes the screening format for a movie (like 2D or 3D).

* **format\_id**: Unique identifier
* **format\_name**: Name of the format

**Shows**

A "show" is a specific combination of a movie, format, and theatre on a given date. Multiple shows can occur for the same movie on the same day.

* **show\_id**: Unique identifier
* **theatre\_id**: Refers to the theatre hosting the show
* **movie\_id**: Refers to the movie being shown
* **format\_id**: Refers to the format
* **show\_date**: Date when the show occurs

**ShowTimes**

Represents the exact times when a show is scheduled on a specific date.

* **show\_time\_id**: Unique identifier
* **show\_id**: Refers to the show
* **start\_time**: Time when the show starts

**2. SQL Table Structures**

Below are SQL scripts to create the tables mentioned above.

CREATE TABLE theatres (

theatre\_id INT PRIMARY KEY,

name VARCHAR(100) NOT NULL,

location VARCHAR(255)

);

CREATE TABLE movies (

movie\_id INT PRIMARY KEY,

title VARCHAR(200) NOT NULL,

language VARCHAR(100) NOT NULL

);

CREATE TABLE formats (

format\_id INT PRIMARY KEY,

format\_name VARCHAR(50) NOT NULL

);

CREATE TABLE shows (

show\_id INT PRIMARY KEY,

theatre\_id INT NOT NULL,

movie\_id INT NOT NULL,

format\_id INT NOT NULL,

show\_date DATE NOT NULL,

FOREIGN KEY (theatre\_id) REFERENCES theatres(theatre\_id),

FOREIGN KEY (movie\_id) REFERENCES movies(movie\_id),

FOREIGN KEY (format\_id) REFERENCES formats(format\_id)

);

CREATE TABLE show\_times (

show\_time\_id INT PRIMARY KEY,

show\_id INT NOT NULL,

start\_time TIME NOT NULL,

FOREIGN KEY (show\_id) REFERENCES shows(show\_id)

);

**3. Sample Data**

Let’s look at some example entries to help understand how the system works.

-- Theatres

INSERT INTO theatres VALUES (1, 'PVR Cinemas', 'Mumbai');

INSERT INTO theatres VALUES (2, 'Cinepolis', 'Delhi');

-- Movies

INSERT INTO movies VALUES (1, 'Inception', 'English');

INSERT INTO movies VALUES (2, 'Spirited Away', 'Japanese');

INSERT INTO movies VALUES (3, '3 Idiots', 'Hindi');

-- Formats

INSERT INTO formats VALUES (1, '2D');

INSERT INTO formats VALUES (2, '3D');

-- Shows

INSERT INTO shows VALUES (1, 1, 1, 1, '2025-05-27');

INSERT INTO shows VALUES (2, 1, 2, 2, '2025-05-27');

INSERT INTO shows VALUES (3, 2, 3, 1, '2025-05-28');

-- Show Times

INSERT INTO show\_times VALUES (1, 1, '10:00');

INSERT INTO show\_times VALUES (2, 1, '14:30');

INSERT INTO show\_times VALUES (3, 1, '19:00');

INSERT INTO show\_times VALUES (4, 2, '11:00');

INSERT INTO show\_times VALUES (5, 2, '15:45');

INSERT INTO show\_times VALUES (6, 3, '13:00');

INSERT INTO show\_times VALUES (7, 3, '18:30');

**4. Query to Get Shows on a Specific Date and Theatre**

Suppose you want to list all the movies playing at "PVR Cinemas" on May 27, 2025. Here’s the SQL query:

SELECT

m.title AS Movie,

m.language AS Language,

f.format\_name AS Format,

st.start\_time AS ShowTime

FROM shows s

JOIN theatres t ON s.theatre\_id = t.theatre\_id

JOIN movies m ON s.movie\_id = m.movie\_id

JOIN formats f ON s.format\_id = f.format\_id

JOIN show\_times st ON st.show\_id = s.show\_id

WHERE t.name = 'PVR Cinemas'

AND s.show\_date = '2025-05-27'

ORDER BY st.start\_time;

This query joins relevant tables and filters results based on theatre name and show date, giving a clear list of shows and their times in chronological order.