# Popcorns to Pixels

## Database Management Project

SUBMITTED TO DR. HARGEETKAUR MA'AM

# Lights, Camera, Data!

## Introduction

In this project we made a step stone in revolutionizing the film industry. As we created a DBMS based system on Online Movie Ticket Booking such that it allows the users to Book their tickets of their favourite movies on their own ease as well connivence

# About Us!

## Meet Our Team!

- Team Leader
- Jainil Doshi
- 21BCP226

- Team Member
- Jugal Chhatriwala
- 21BCP240

- Team Member
- Rashmi Patel
- 21BCP231

# Objectives

1. To develop a database management system that can efficiently handle the booking and sales of movie tickets.
2. To create a user-friendly interface that allows users to easily search for and book movie tickets online.
3. To provide real-time updates on movie showtimes and availability of tickets, keeping users informed and reducing the risk of overbooking or double-booking.
4. To improve the overall experience of moviegoers by reducing wait times and enabling faster access to tickets.5
5. To allow businesses to customize their ticket prices and promotions,

# Functionalites

**Even we add several funtionalites that makes the user experinece much smoother as well comfortable too!**

1.Name, Location, Number of Screen

2. Price , Availability of the Ticket

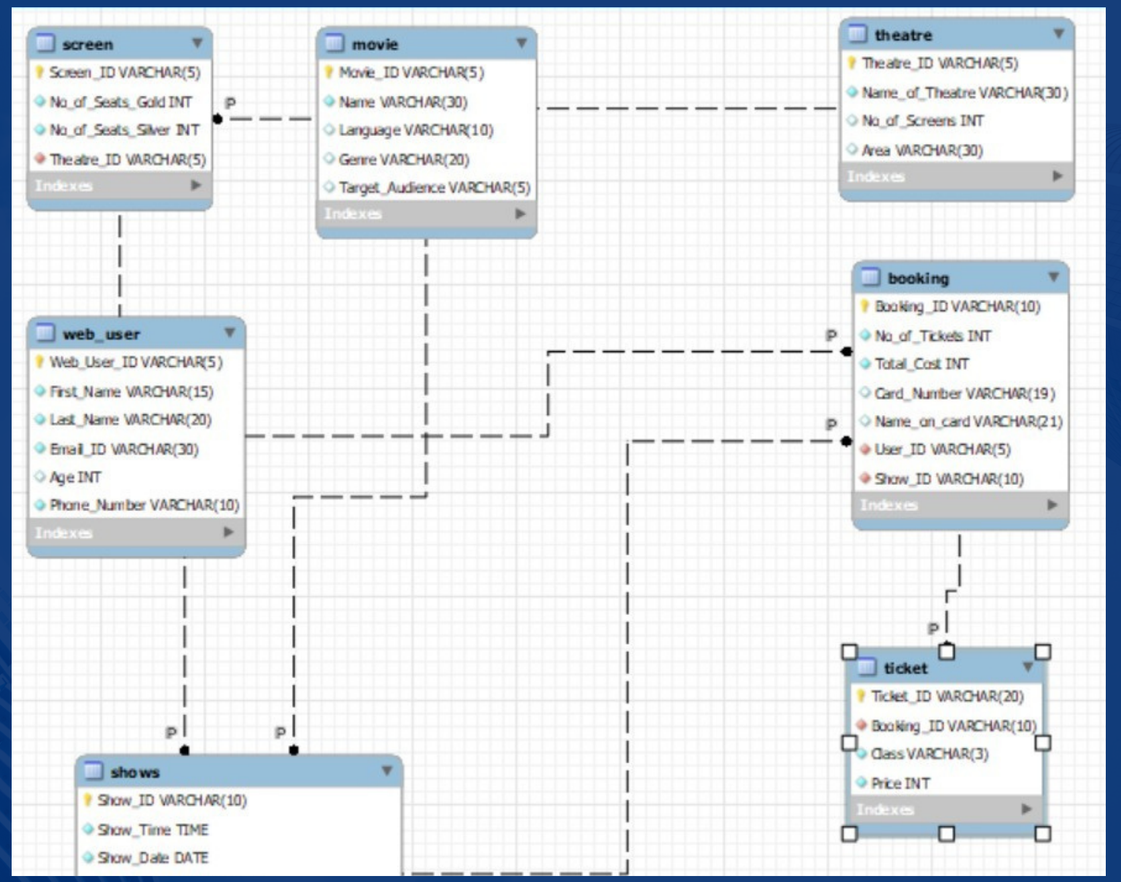
3. Genre, Starcast, Directors, Reviews, Language of the Movie

## We too collect information from user such as :

Customer Information, Contact Details, Age for their better Personalization and Recommendations for Movies and Shows.

# Entity-Relational Diagram

## (Before Normalization)



# Normalization

## 1NF

First Normal Form (1NF): All tables are already in 1NF as they have atomic values and no repeating groups. However, In This Web_User table, the Phone_number is the seperate attribute but still in the same itself table. So we need to create a seperate table for it and link it with Web_user table via foreign key.

# 2NF

To make the tables 2NF, we need to ensure that each table has a single purpose or theme and does not contain any partial dependencies.
Looking at the existing tables, we can see that Web_User has a partial dependency, as the Phone_Number attribute only depends on the Web_User_ID. Therefore, we need to split it into two separate tables.
In the Booking table, there is a partial dependency of Show_ID on User_ID, as Show_ID is dependent on User_ID as well as Booking_ID. Therefore, we need to create a separate table for this dependency.
And later on we can link this easily via using foreign key.

# 3NF

A relation is in third normal form, if there is no transitive dependency for non-prime attributes as well as it is in second normal form.
To bring it to 3NF we need to split 2 Table Movie and Booking into Movie_Details and Booking_Details.
New Tables:-

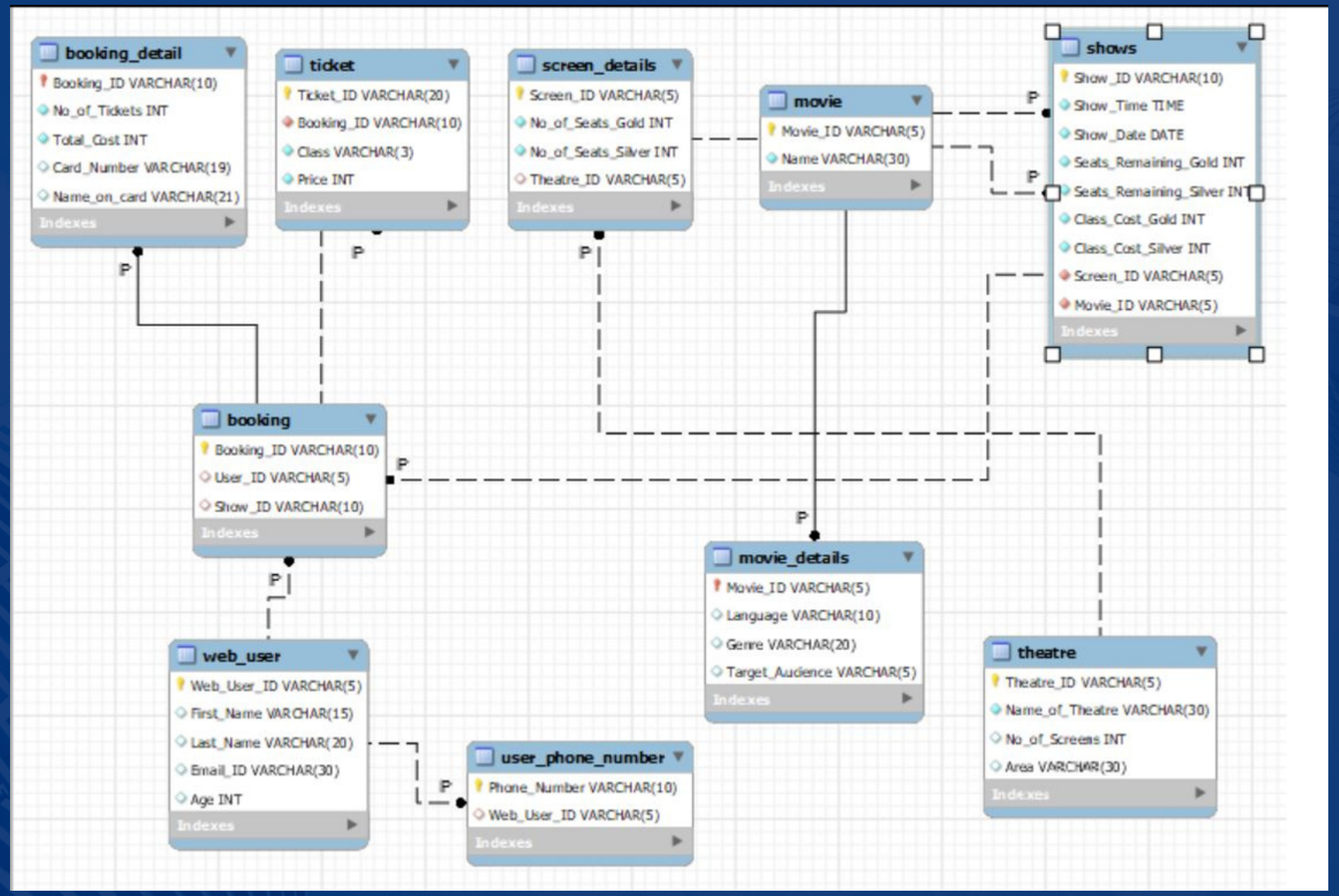
- movie_Details(Language, Genre, Target_Audience)
- booking_details(booking_id,no_of_tickets,total_cost,card_number,name_on_card)

# BCNF

It is the abbreviated form of Boyce–Codd Normal Form.
All tables are already in BCNF as they have no non-trivial dependencies between attributes.
LHS must be candidate or Super  Key.

# Entity-Relational Diagram

## (After Normalization)



# Relational Model

- Web_User(user_id,First_Name,Last_Name,Email_ID, Age)
  - This table stores information about the users' such as their user_id, Name, Email and Age.
- Theatre(Theatre_ID, Name_Theatre, No_Screens, Area)
  - This table stores information about the Theatre such as their Theatre_id, Name of the Theatre, Number of Screen and It's Area.
- Screen(Screen_ID, Seats_Gold, Seats_Silver, Theatre_Id)
  - This table stores information about the Screen and Type of Seat Availability.

- Movie (Name)
  - This table stores information about the current ongoing movies that are casting on the respective screens
- Movies_details(Movie_ID, Language, Genre, Target_Audience)
  - This table stores information about the language, genre and the target_Audience of the Movies
- Show_details(Show_id, Show_Date,Remaining seats,  Cost, Screen_id,Movie_id)
  - This table stores information about the Show time , id, Date and the Remaining Seats in the Gold and Silver Seats and  it s cost.

- Booking_details (Booking_id, tickets, Total_cost, Card_number, Name)
  - This table stores information about the Booking  System and Payment Methods as well the name on the card and its number.
- Ticket( Ticket_id, Booking_id, Class, Price)
  - This table stores information about Ticket scheme and its price and its class
- Booking(Booking_id, user_id, show_id
  - This table stores the information about the booking section
- user_phone_number(phone_number, web_user_id)
  - This table contains the information about the user's phone number and contains user_id

# SQL QUERIES

```
CREATE TABLE Theatre (
    Theatre_ID VARCHAR(5) PRIMARY KEY,
    Name_of_Theatre VARCHAR(30) NOT NULL,
    No_of_Screens INT,
    Area VARCHAR(30)

CREATE TABLE Screen_Details (
   Screen_ID VARCHAR(5) PRIMARY KEY,
   No_of_Seats_Gold INT NOT NULL,
   No_of_Seats_Silver INT NOT NULL,
   Theatre_ID VARCHAR(5),
   FOREIGN KEY (Theatre_ID) REFERENCES Theatre(Theatre_ID)
);
CREATE TABLE Movie_Details (
   Movie_ID VARCHAR(5) PRIMARY KEY,
   Language VARCHAR(10),
   Genre VARCHAR(20),
   Target_Audience VARCHAR(5),
   FOREIGN KEY (Movie_ID) REFERENCES Movie(Movie_ID)
);
```

```
CREATE TABLE Shows (
 Show_ID VARCHAR(10) PRIMARY KEY,
 Show_Time TIME NOT NULL,
 Show_Date DATE NOT NULL,
 Seats_Remaining_Gold INT NOT NULL CHECK
(Seats_Remaining_Gold >= 0),
 Seats_Remaining_Silver INT NOT NULL CHECK
(Seats_Remaining_Silver >= 0),
 Class_Cost_Gold INT NOT NULL,
 Class_Cost_Silver INT NOT NULL,
 Screen_ID VARCHAR(5) NOT NULL,
 Movie_ID VARCHAR(5) NOT NULL,
 FOREIGN KEY (Screen_ID) REFERENCES
Screen_Details(Screen_ID) ,
 FOREIGN KEY (Movie_ID) REFERENCES Movie(Movie_ID)
);
```

```
CREATE TABLE Ticket (
 Ticket_ID VARCHAR(20) PRIMARY KEY,
 Booking_ID VARCHAR(10) NOT NULL,
 Class VARCHAR(3) NOT NULL,
 Price INT NOT NULL,
 FOREIGN KEY (Booking_ID) REFERENCES
Booking(Booking_ID)
);

CREATE TABLE Booking_Detail (
   Booking_ID VARCHAR(10) PRIMARY KEY,
   No_of_Tickets INT NOT NULL,
   Total_Cost INT NOT NULL,
   Card_Number VARCHAR(19),
   Name_on_card VARCHAR(21),
      FOREIGN KEY (Booking_ID) REFERENCES
Booking(Booking_ID)
);
```

```
CREATE TABLE Booking (
   Booking_ID VARCHAR(10) PRIMARY KEY,
   User_ID VARCHAR(5),
   Show_ID VARCHAR(10),
   FOREIGN KEY (User_ID) REFERENCES
Web_User(Web_User_ID) ,
   FOREIGN KEY (Show_ID) REFERENCES
Shows(Show_ID)
);
CREATE TABLE User_Phone_Number (
   Phone_Number VARCHAR(10) PRIMARY KEY,
   Web_User_ID VARCHAR(5),
   FOREIGN KEY (Web_User_ID) REFERENCES
Web_User(Web_User_ID)
);
```

# Populating the Tables

```
INSERT INTO Theatre (Theatre_ID,
Name_of_Theatre, No_of_Screens, Area)
VALUES
   ('T001', 'PVR Phoenix', 6, 'Ahemdabad'),
   ('T002', 'Conplex', 8, 'Vadodra'),
   ('T003', 'Cinepolis ', 4, 'Bhopal'),
   ('T004', 'Rajhans', 5, 'Sciencecity'),
   ('T005', 'INOX Forum', 6, 'Rajkot');

-- INSERT INTO Screen_Details
(Screen_ID, No_of_Seats_Gold,
No_of_Seats_Silver, Theatre_ID)
VALUES
   ('S001', 100, 200, 'T001'),
   ('S002', 150, 250, 'T002'),
   ('S003', 120, 180, 'T003'),
   ('S004', 80, 150, 'T004'),
   ('S005', 90, 170, 'T005');
```

```
INSERT INTO Movie_Details (Movie_ID, Language, Genre,
Target_Audience)
VALUES
   ('M001', 'English', 'Action', 'PG-13'),
   ('M002', 'English', 'Drama', 'R'),
   ('M003', 'Hindi', 'Epic', 'U/A'),
   ('M004', 'Hindi', 'Romance', 'U'),
   ('M005', 'English', 'Animation', 'U');

INSERT INTO Booking_Detail (Booking_ID, No_of_Tickets, Total_Cost,
Card_Number, Name_on_card)
VALUES
('B001', 2, 600, '1234567890123456', 'Rashmi Panchal'),
('B002', 1, 200, '2345678901234567', 'Jugal Chhatrivala'),
('B003', 3, 900, '3456789012345678', 'Jainil Doshi'),
('B004', 4, 1200, '4567890123456789', 'Amit Patel'),
('B005', 2, 600, '5678901234567890', 'Sara Shah');
```

# THANK YOU

*For Giving Your Time*