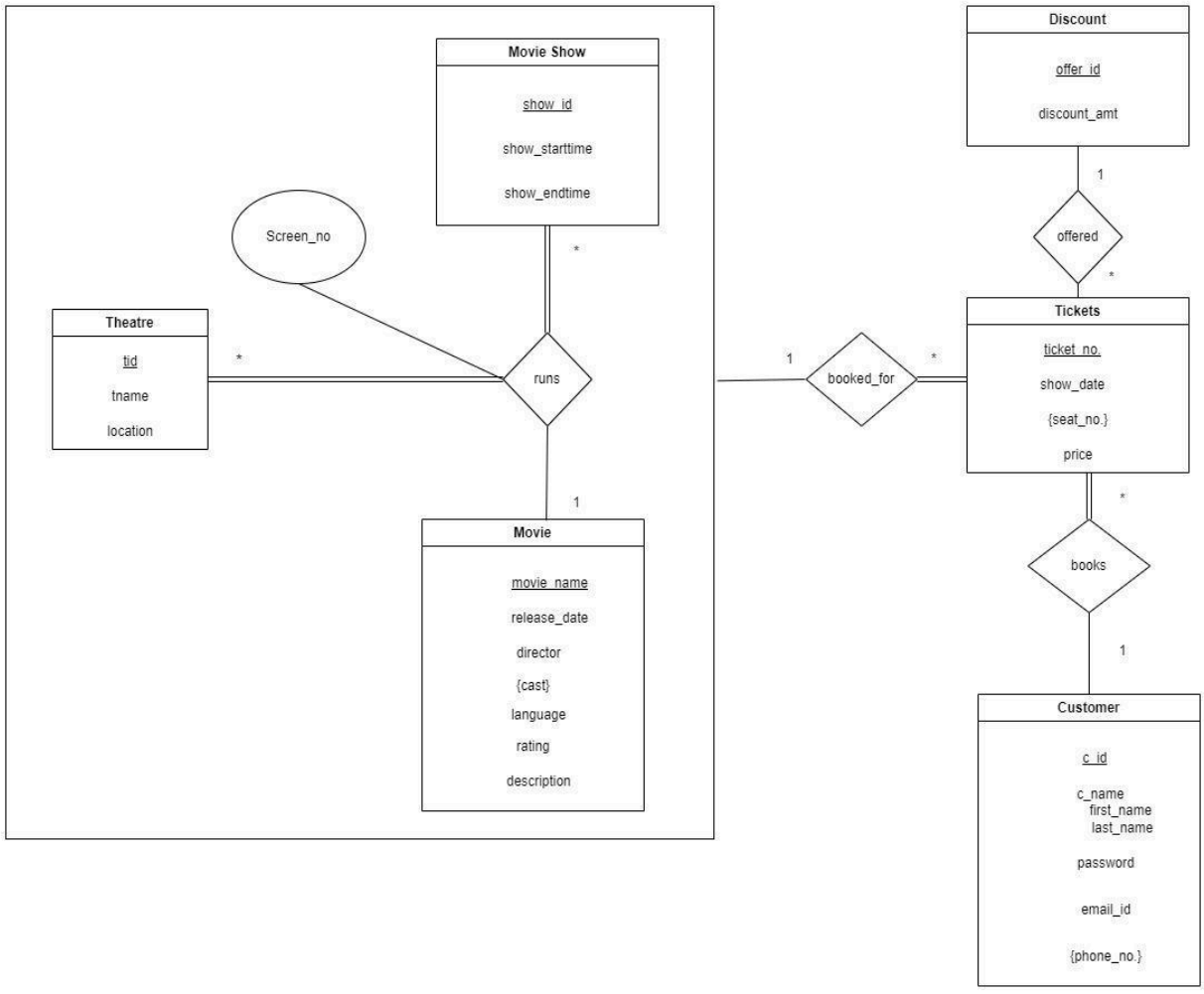
Movie Ticket Booking System

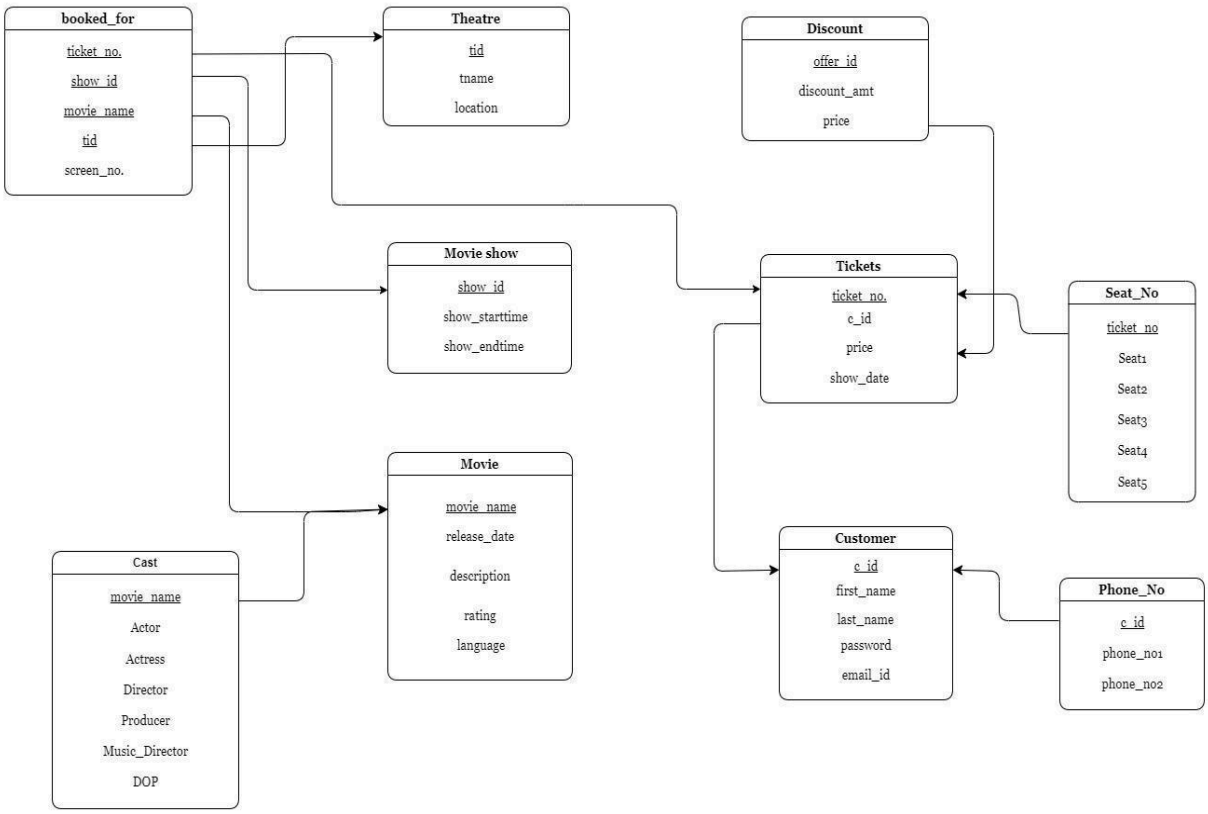
Project Review 2 - Group 10

AM.EN.U4AIE21114 - Annapoorna A K
AM.EN.U4AIE21130 - Gayathry M Wariyar
AM.EN.U4AIE21159 - Shreya Pulluri
AM.EN.U4AIE21165 - Varalakshmi M P
AM.EN.U4AIE21123 - CH. Sai Sriram Varma

1) Extended ER diagram



2) Schema diagram



3) Relational Schema

Booked_for (ticket_no, show_id, movie_name, tid, screen_no)

Theatre (tid, tname, location)

Discount (offer_id, discount_amt, price)

Movie_show (show_id, show_starttime, show_endtime)

Tickets (ticket_no, c_id, price, show_date)

Seat_No (ticket_no, Seat1, Seat2, Seat3, Seat4, Seat5)

Cast (movie_name, Actor, Actress, Director, Producer, Music Director, DOP)

Movie (movie_name, release_date, description, rating, language)

Customer (c_id, first_name, last_name, password, email_id)

Phone_No (c_id, phone_no1, phone_no2)

4) Universal Schema : (25 attributes)

Ticket_no. ,seat_no.,show_date, price, c_id, first_name, last_name, password, email_id, phone_no., discount_amt, **offer_id**, show_id, show_starttime, show_endtime, movie_name, screen_no. , tid, tname, location, release_date, description, rating, language, cast

5) Normalization

For the schema to be in 1NF we make all the attributes atomic -

a. Seat_no. → seat_no1, seat_no2, seat_no3, seat_no4, seat_no5
b. Phone_no → phone_no1, phone_no2
c. Cast → language, actor, actress, director, producer, music_director, DOP

1NF :

Ticket_no. ,seat_no1, seat_no2, seat_no3, seat_no4, seat_no5, show_date, price, c_id, first_name, last_name, password, email_id, phone_no1 ,phone_no2., discount_amt, **offer_id,** show_id, show_starttime, show_endtime, movie_name, screen_no. , tid, tname, location, release_date, description, rating, language, actor, actress, director, producer, music_director, DOP

Partial dependencies:

Attributes like seat_no1, seat_no2, seat_no3, seat_no4, seat_no5, show_date, price etc. are just dependent on **Ticket_no** not on both offer_id and Ticket_no.
Similarly discount_amt is dependent on only **offer_id**.

2NF :

Ticket (Ticket_no. ,seat_no1, seat_no2, seat_no3, seat_no4, seat_no5, show_date, price, c_id, first_name, last_name, password, email_id, phone_no1 ,phone_no2., show_id, show_starttime, show_endtime, movie_name, screen_no. , tid, tname, location, release_date, description, rating, language, actor, actress, director, producer, music_director, DOP)

Discount (offer_id, discount_amt)

After 2NF these are the Transitive Dependencies present:
c_id → first_name, last_name, password, email_id, phone_no1, phone_no2
show_id → show_starttime, show_endtime, screen_no, movie_name
tid → tname, location
movie_name → release_date, description, rating, language, actor, actress, director, producer, music_director, DOP

3NF :
Ticket (Ticket_no, c_id, seat_no1, seat_no2, seat_no3, seat_no4, seat_no5, price, show_date, show_id)

Customer (c_id, first_name, last_name, password, email_id, phone_no1 ,phone_no2)

Movie_show (show_id, show_starttime, show_endtime, tid, Screen_no, movie_name)

Movie (movie_name, release_date, description, rating, language, actor, actress, director, producer, music_director, DOP)

Theatre (tid, tname, location)

Discount (offer_id, discount_amt)

Ticket_Discount (Ticket_no, offer_id)

6) DDL Statements

```
-- Theatre table
CREATE TABLE Theatre (
    tid INT PRIMARY KEY,
    tname VARCHAR(100) NOT NULL,
    location VARCHAR(255) NOT NULL
);

-- Movie table
CREATE TABLE Movie (
    movie_name VARCHAR(255) PRIMARY KEY,
    release_date DATE NOT NULL,
    description TEXT,
    rating VARCHAR(10),
    language VARCHAR(50),
```

```
    actor VARCHAR(100),
    actress VARCHAR(100),
    director VARCHAR(100),
    producer VARCHAR(100),
    music_director VARCHAR(100),
    DOP VARCHAR(100)
);


-- Customer table
CREATE TABLE Customer (
    c_id SERIAL PRIMARY KEY,
    first_name VARCHAR(50) NOT NULL,
    last_name VARCHAR(50) NOT NULL,
    password VARCHAR(255) NOT NULL,
    email_id VARCHAR(100) UNIQUE NOT NULL,
    phone_no1 VARCHAR(20),
    phone_no2 VARCHAR(20)
);

-- Movie_Show table
CREATE TABLE Movie_Show (
    show_id INT PRIMARY KEY,
    tid INT,
    Screen_no INT NOT NULL,
    show_starttime TIME NOT NULL,
    show_endtime TIME NOT NULL,
    movie_name VARCHAR(255),
    FOREIGN KEY (tid) REFERENCES Theatre(tid),
    FOREIGN KEY (movie_name) REFERENCES Movie(movie_name));

-- Ticket table
CREATE TABLE Ticket (
    Ticket_no SERIAL PRIMARY KEY,
    c_id INT,
    seat_no1 INT NOT NULL,
    seat_no2 INT,
    seat_no3 INT,
    seat_no4 INT,
    seat_no5 INT,
    price DECIMAL(10, 2) NOT NULL,
    show_date DATE NOT NULL,
    show_id INT,
    FOREIGN KEY (c_id) REFERENCES Customer(c_id),
    FOREIGN KEY (show_id) REFERENCES Movie_Show(show_id));
```

```
-- Discount table
CREATE TABLE Discount (
    offer_id INT PRIMARY KEY,
    discount_amt DECIMAL(10, 2) NOT NULL
);


-- Ticket_Discount Table
CREATE TABLE Ticket_Discount (
    ticket_no INT,
    offer_id INT,
    PRIMARY KEY (ticket_no, offer_id),
    FOREIGN KEY (ticket_no) REFERENCES Ticket(ticket_no),
    FOREIGN KEY (offer_id) REFERENCES Discount(offer_id)
);
```

7) Sample queries:

i. Aggregate functions, Group by…having

Q. Find theatres with more than or equal to 3 shows.

```
SELECT t.tid, t.tname, COUNT(*) as show_count
FROM Theatre t
JOIN Movie_Show ms ON t.tid = ms.tid
GROUP BY t.tid, t.tname
HAVING COUNT(*) >= 3;
```

	tid [PK] integer	tname character varying (100)	show_count bigint
1	4	Sathyam Cinemas	3
2	6	Carnival Cinemas	3
3	2	INOX	3
4	7	PVR Icon	3
5	3	Prasads Multiplex	3
6	1	PVR Cinemas	4
7	5	Cinepolis	3

Q. Calculate ticket sales statistics per movie.

```
SELECT
    m.movie_name,
    COUNT(t.Ticket_no) as total_tickets_sold,
    SUM(t.price) as total_revenue,
    AVG(t.price) as average_ticket_price,
    MIN(t.price) as lowest_ticket_price,
    MAX(t.price) as highest_ticket_price
FROM
    Movie m
JOIN
    Movie_Show ms ON m.movie_name = ms.movie_name
JOIN
    Ticket t ON ms.show_id = t.show_id
GROUP BY
    m.movie_name
ORDER BY
    total_revenue DESC;
```

ii. Order by

Q. List movies ordered by release date (newest first)

```
SELECT movie_name, release_date
FROM Movie
ORDER BY release_date DESC;
```

	movie_name [PK] character varying (255)	release_date date
1	Pathaan	2023-01-25
2	Black Panther: Wakanda Forever	2022-11-11
3	Ponniyin Selvan: I	2022-09-30
4	Vikram	2022-06-03
5	Doctor Strange in the Multiverse of Madness	2022-05-06
6	KGF: Chapter 2	2022-04-14
7	RRR	2022-03-24

iii. Join, Outer Join

Q. List all theatres and their shows

```
SELECT t.tname, ms.show_id, ms.movie_name,
ms.show_starttime
FROM Theatre t
LEFT OUTER JOIN Movie_Show ms ON t.tid = ms.tid;
```

	tname character varying (100)	show_id integer	movie_name character varying (255)	show_starttime time without time zone
1	PVR Cinemas	1	Pathaan	14:30:00
2	INOX	2	RRR	18:00:00
3	Prasads Multiplex	3	Ponniyin Selvan: I	19:30:00
4	Sathyam Cinemas	4	Doctor Strange in the Multiverse of Madness	12:00:00
5	Cinepolis	5	KGF: Chapter 2	15:00:00
6	Carnival Cinemas	6	Vikram	20:00:00
7	PVR Icon	7	Black Panther: Wakanda Forever	11:00:00
8	PVR Cinemas	8	RRR	09:00:00
9	INOX	9	Ponniyin Selvan: I	13:00:00
10	Prasads Multiplex	10	KGF: Chapter 2	16:00:00
11	Sathyam Cinemas	11	Pathaan	19:00:00

Total rows: 24 of 24 Query complete 00:00:00.085

iv. Query having Boolean operators

Q. Find customers who have booked tickets for evening shows (after 6 PM) or morning shows (before 12 PM)

```
SELECT DISTINCT c.c_id, c.first_name, c.last_name
FROM Customer c
JOIN Ticket t ON c.c_id = t.c_id
JOIN Movie_Show ms ON t.show_id = ms.show_id
WHERE ms.show_starttime > '18:00:00' OR ms.show_starttime
< '12:00:00';
```

	c_id [PK] integer	first_name character varying (50)	last_name character varying (50)
1	1	Rahul	Sharma
2	2	Priya	Patel

Q. Find movies that are either in Tamil or Hindi, directed by a specific director.

```
SELECT movie_name, language, release_date, director, actor
FROM Movie
WHERE (language = 'Tamil' OR language = 'Hindi')
  AND (director = 'Mani Ratnam')
ORDER BY release_date DESC;
```

	movie_name [PK] character varying (255)	language character varying (50)	release_date date	director character varying (100)	actor character varying (100)
1	Ponniyin Selvan: I	Tamil	2022-09-30	Mani Ratnam	Vikram
2	Guru	Hindi	2007-01-12	Mani Ratnam	Abhishek Bachchan

v. Query having arithmetic operators

Q. Calculate the total revenue for each theatre

```
SELECT t.tid, t.tname, SUM(tk.price * (
   1+(CASE WHEN tk.seat_no2 IS NOT NULL THEN 1 ELSE 0 END)
+
     (CASE WHEN tk.seat_no3 IS NOT NULL THEN 1 ELSE 0 END)
+
     (CASE WHEN tk.seat_no4 IS NOT NULL THEN 1 ELSE 0 END)
+
     (CASE WHEN tk.seat_no5 IS NOT NULL THEN 1 ELSE 0
END))) as total_revenue
FROM Theatre t
JOIN Movie_Show ms ON t.tid = ms.tid
JOIN Ticket tk ON ms.show_id = tk.show_id
GROUP BY t.tid, t.tname;
```

	tid [PK] integer	tname character varying (100)	total_revenue numeric
1	6	Carnival Cinemas	2000.00
2	2	INOX	8400.00
3	3	Prasads Multiplex	3000.00
4	1	PVR Cinemas	4050.00
5	5	Cinepolis	600.00

vi. A search query using string operators

Q. Find movies with 'action' in their description

```
SELECT movie_name, description
FROM Movie
WHERE LOWER(description) LIKE '%action%'
```

	movie_name [PK] character varying (255)	description text
1	KGF: Chapter 2	Action-drama sequel
2	Pathaan	An action thriller featuring SRK
3	Vikram	Tamil action thriller

vii. Usage of to_char, extract

Q. Get monthly ticket sales report of each theatre

```
SELECT
    TO_CHAR(t.show_date, 'YYYY-MM') AS month,
    TO_CHAR(t.show_date, 'Month') AS month_name,
    EXTRACT(YEAR FROM t.show_date) AS year,
    th.tname AS theatre_name,
    SUM(t.price) AS total_revenue
FROM
    Ticket t
JOIN
    Movie_Show ms ON t.show_id = ms.show_id
JOIN
    Theatre th ON ms.tid = th.tid
GROUP BY
    TO_CHAR(t.show_date, 'YYYY-MM'),
    TO_CHAR(t.show_date, 'Month'),
    EXTRACT(YEAR FROM t.show_date),
    th.tname
ORDER BY
    year DESC, month DESC, total_revenue DESC;
```

	month text	month_name text	year numeric	theatre_name character varying (100)	total_revenue numeric
1	2024-06	June	2024	INOX	1900.00
2	2024-06	June	2024	Prasads Multiplex	1000.00
3	2024-06	June	2024	PVR Cinemas	700.00
4	2024-06	June	2024	PVR Icon	550.00
5	2024-06	June	2024	Carnival Cinemas	500.00
6	2024-06	June	2024	Cinepolis	300.00
7	2024-06	June	2024	Sathyam Cinemas	300.00
8	2024-05	May	2024	INOX	400.00
9	2024-05	May	2024	PVR Cinemas	300.00
10	2024-05	May	2024	Prasads Multiplex	250.00
11	2024-04	April	2024	PVR Cinemas	350.00
12	2024-04	April	2024	Prasads Multiplex	200.00
13	2024-03	March	2024	Prasads Multiplex	300.00
14	2024-03	March	2024	Cinepolis	300.00
15	2024-03	March	2024	PVR Cinemas	300.00

viii. Between, IN, Not between, Not IN

Q. Find tickets for shows between 2 PM and 6 PM, excluding specific theatres

```
SELECT t.Ticket_no, ms.show_starttime, th.tname
FROM Ticket t
JOIN Movie_Show ms ON t.show_id = ms.show_id
JOIN Theatre th ON ms.tid = th.tid
WHERE ms.show_starttime BETWEEN '14:00:00' AND '18:00:00'
  AND th.tid NOT IN (1, 3, 5);
```

	ticket_no integer	show_starttime time without time zone	tname character varying (100)
1	5	18:00:00	INOX
2	7	18:00:00	INOX
3	8	18:00:00	INOX
4	15	18:00:00	INOX
5	6	18:00:00	INOX

ix. Set operations

Q. Find actors in either 'Action' or 'Drama' movies.

```
SELECT actor, actress
FROM Movie
WHERE description LIKE '%action%'
UNION
SELECT actor, actress
FROM Movie
WHERE description LIKE '%drama%'
```

	actor character varying (100)	actress character varying (100)
1	Kamal Haasan	Fahadh Faasil
2	Shah Rukh Khan	Deepika Padukone
3	Vikram	Aishwarya Rai Bachchan
4	Yash	Srinidhi Shetty

Q. Find movie names that are played in theatres located in 'Mumbai' and have a release date after March 1st 2022.

```
SELECT MS.movie_name
FROM Movie_Show MS
JOIN Theatre T ON MS.tid = T.tid
WHERE T.location LIKE '%Mumbai%'
INTERSECT
SELECT movie_name
FROM Movie
WHERE release_date > '2022-03-01';
```

	movie_name character varying (255)
1	Doctor Strange in the Multiverse of Madness
2	KGF: Chapter 2
3	Black Panther: Wakanda Forever
4	RRR
5	Vikram
6	Ponniyin Selvan: I

x. Subquery using EXISTS / NOT EXISTS, ANY, ALL

Q. Find customers who have used all available discounts.

```
SELECT c.c_id, c.first_name, c.last_name
FROM Customer c
WHERE NOT EXISTS (
    SELECT d.offer_id
    FROM Discount d
    WHERE NOT EXISTS (
        SELECT 1
        FROM Ticket_Discount td
        WHERE td.ticket_no IN (
            SELECT t.Ticket_no
            FROM Ticket t
            WHERE t.c_id = c.c_id
        )
        AND td.offer_id = d.offer_id
    )
);
```

	c_id [PK] integer	first_name character varying (50)	last_name character varying (50)
1	1	Rahul	Sharma

Q. Find customers who have bought tickets for any movie with a rating greater than 4.0

```
SELECT c_id, first_name, last_name, email_id
FROM Customer
WHERE c_id = ANY (
    SELECT T.c_id
    FROM Ticket T
    JOIN Movie_Show MS ON T.show_id = MS.show_id
    JOIN Movie M ON MS.movie_name = M.movie_name
    WHERE M.rating::NUMERIC > 4.0
);
```

	c_id [PK] integer	first_name character varying (50)	last_name character varying (50)	email_id character varying (100)
1	1	Rahul	Sharma	rahul.sharma@email.com
2	2	Priya	Patel	priya.patel@email.com
3	3	Amit	Singh	amit.singh@email.com

8) Assumptions and Constraints:

a. Unique - Attribute like email_id should have unique values
b. Not Null - Attributes like show_endtime , show_starttime , password, theatre name, theatre location and customer name cannot be null values.
c. Assumptions - Seat_no1 cannot have a null value i.e. a minimum of one seat has to be booked.
d. Every seat for all the theatres has the same cost of Rs.150 per seat.

9) Attributes:

a. Ticket_no. : Auto-generated ticket numbers.
b. Seat_no. : Can be from A1 to A5. A user can book at most 5 tickets.
c. Show_date: Date of the movie show.
d. Price: Ticket Price
e. C_id: Customer id
f. First_name: First name of the customer
g. Last_name: Last name of the customer
h. Password: Password used by the customer to log in to the system.
i. Email_id: Email id used by the customer to log in to the system.
j. phone_no. : Phone number of the customer. At Most 2 phone numbers can be used.
k. Discount_amt: The amount being discounted from the price.
l. Offer_id: Offer code of the discount coupon being applied.
m. Show_id: id of the particular movie show.
n. Show_starttime: Starting time of the movie show.
o. Show_endtime: Ending time of the movie show.
p. Movie_name: Name of the movie.
q. Screen_no.: Screen number of a particular screen in the theatre.
r. Tid: Theatre id.
s. Tname: Theatre's name
t. Location: Location of the theatre.
u. Release_date: Release date of a movie
v. Description: Genre of the movie
w. Rating: Customer Ratings for a movie
x. Language: The language in which the movie will be screened.
y. Cast: Actors amd other technicians worked in a movie.