



विश्वजीवनामृतं ज्ञानम्

**ABV- INDIAN INSTITUTE OF INFORMATION TECHNOLOGY AND MANAGEMENT, GWALIOR**

# **DataBase Management System**

## **MINI PROJECT**

**Topic : Movie-Ticket-Booking-System**



# **TEAM MEMBERS -**

**Aman Kumar**

**2020IMT-007**

**Ansh Rusia**

**2020IMT-012**

**Shubhajeet Pradhan**

**2020IMT-097**

**Varun Kumar Tiwari**

**2020IMT-112**

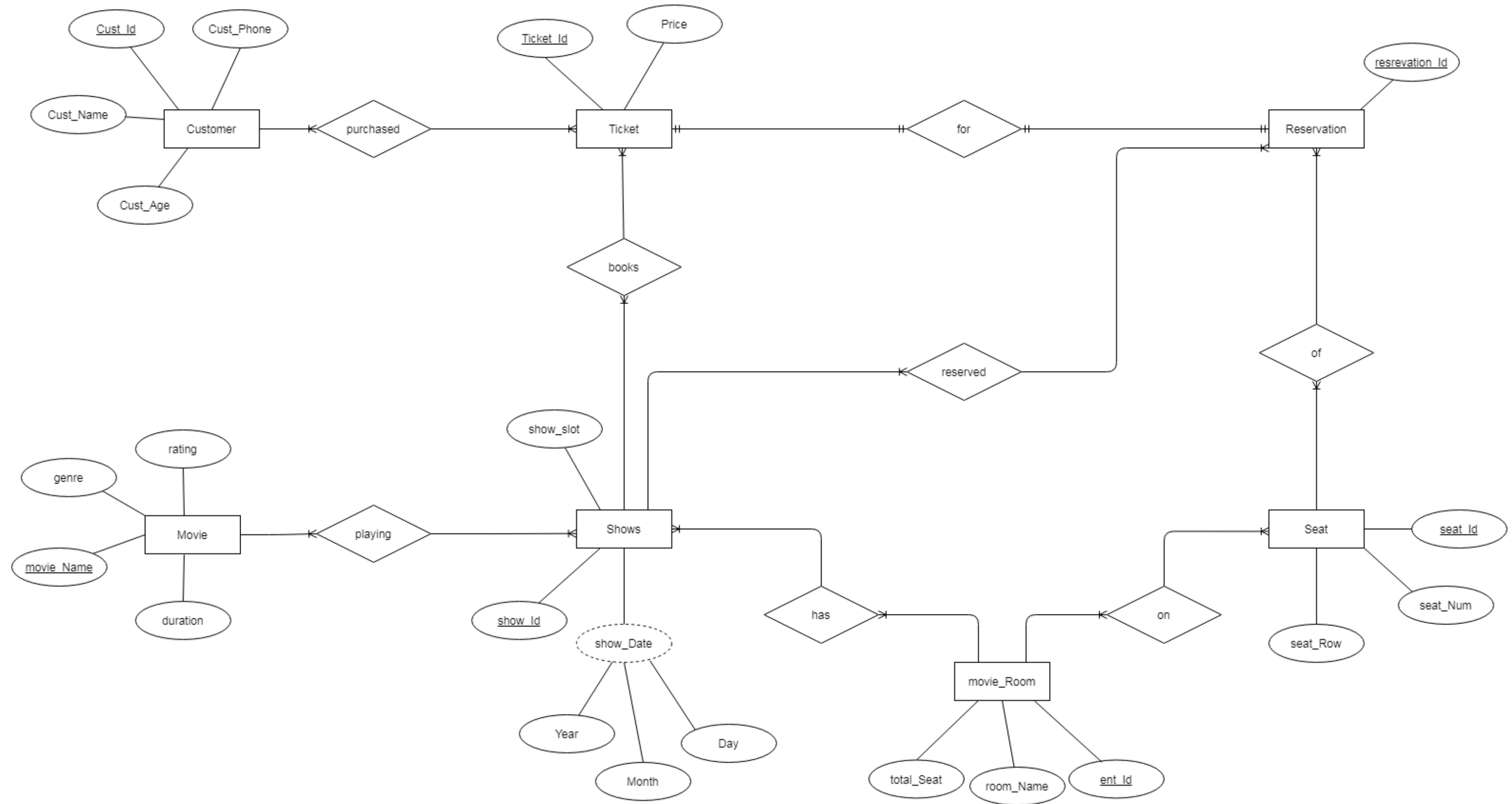
# Introduction

As the name suggests the movie ticket management system is a database management system for a multiplex. This Project aims to provide an insight of such Movie ticket management System. The database is designed to accommodate multiple theater rooms at same time to have a hassle free experience for the customer.

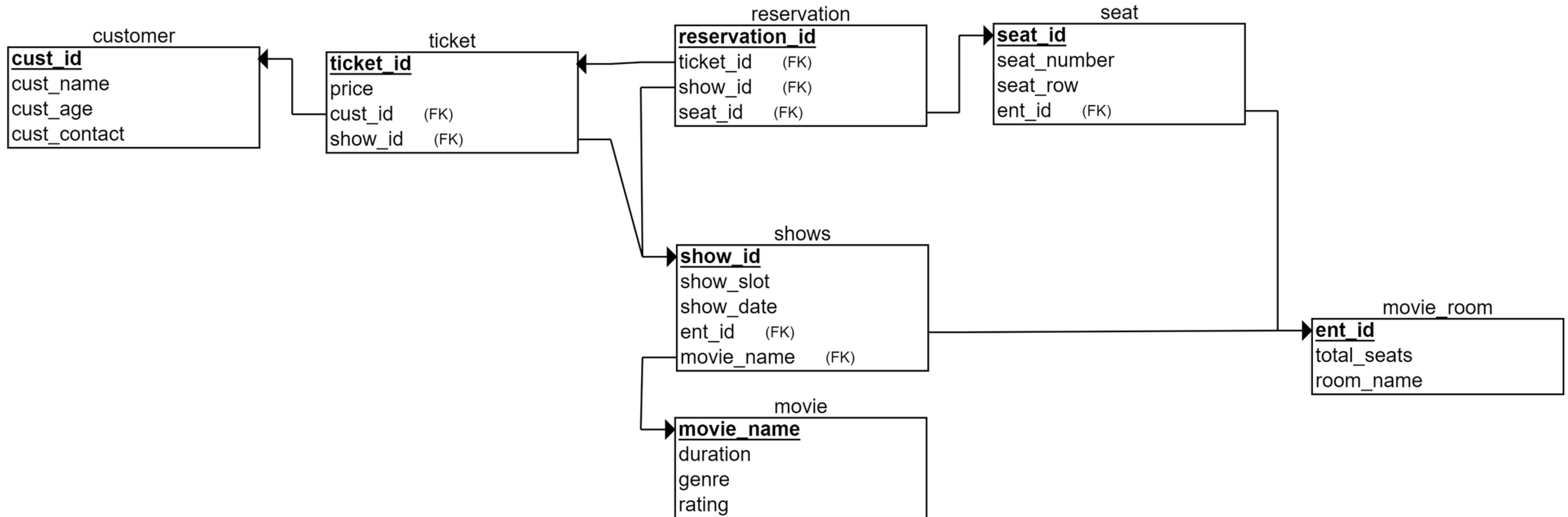
This management system is highly flexible and is well efficient for managing all information about the customer, movie and seats. The key focus is: well management of data and easy retrieval of information. Some key benefits of this projects are :-

Data consistency, Easy to handle, Easy data updating and Easy record keeping, Data redundancy can be avoided to some extent.

# Entity Relationship Model



# Relationship Schema



# FUNCTIONAL DEPENDENCIES

## Customer Entity

Functional Dependency = {cust\_id --> cust\_age, cust\_name, cust\_phone}  
Candidate key = cust\_id

## Movie Entity

Functional Dependency = { movie\_name --> room\_name total\_seats}  
Candidate key = movie\_name

## Movie Room Entity

Functional Dependency = { ent\_id --> room\_name total\_seats}  
Candidate key = ent\_id

## Reservation Entity

Functional Dependency = { reservation\_id --> ticket\_id, show\_id, seat\_id}  
Candidate key = reservation\_id

# FUNCTIONAL DEPENDENCIES

## Seat Entity

Functional Dependency = { seat\_id --> seat\_number, seat\_row, ent\_id }  
Candidate key = seat\_id

## Shows Entity

Functional Dependency = { show\_id --> show\_slot, show\_date, ent\_id, movie\_name }  
Candidate key = show\_id

## Ticket Entity

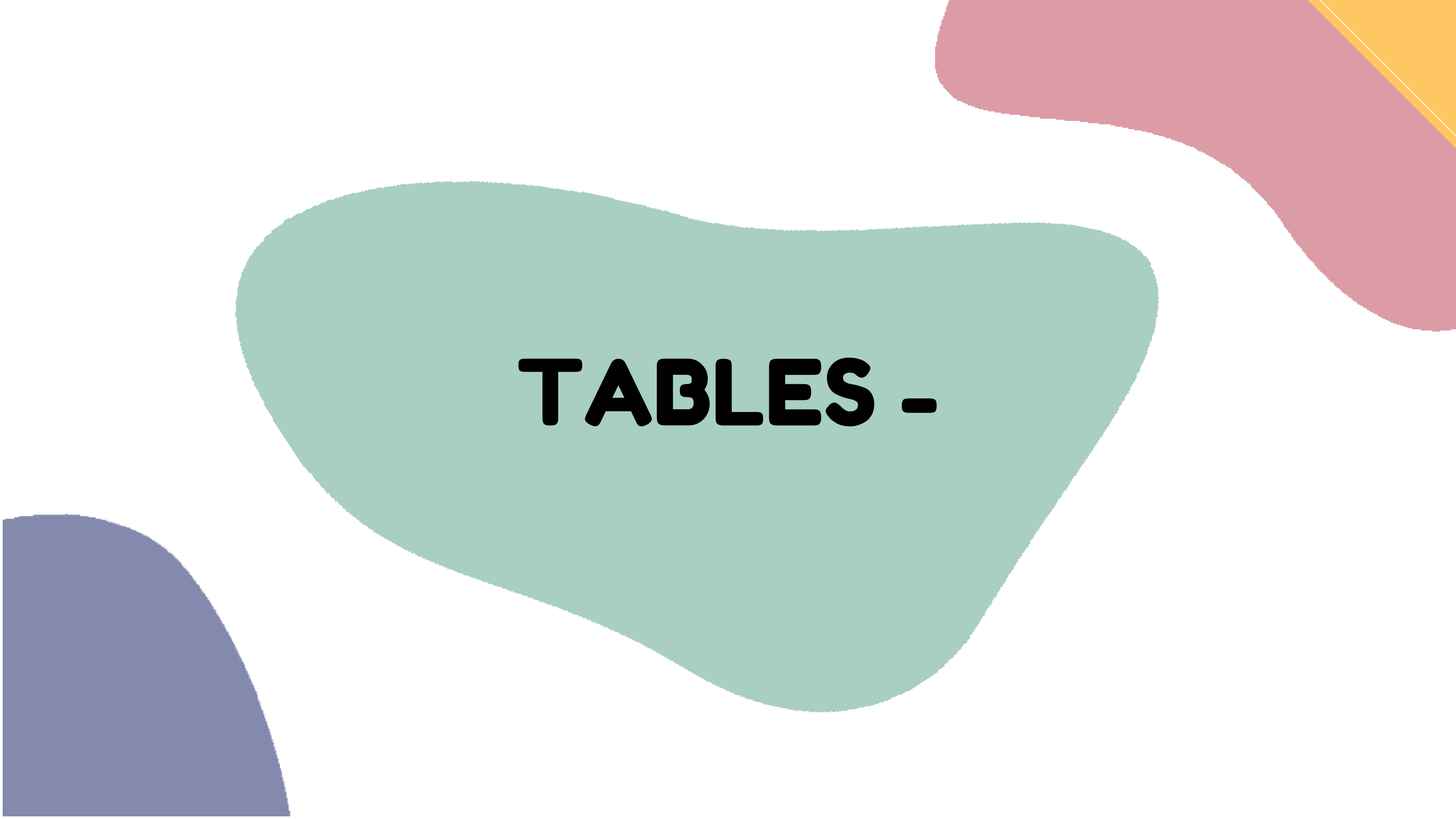
Functional Dependency = { ticket\_id --> price, cust\_id, show\_id }  
Candidate key = ticket\_id



# Normalisation

- 1 NF - In our management system, Customer, Ticket, Reservation, Seat, Movie Room, Shows and Movie has no attribute is multi-valued or composite attribute. Therefore, it is in First Normal Form.
- 2 NF - As there is no non-prime attribute defined by set of candidate key therefore, it is in Second Normal Form.
- 3 NF - In our entire relation there is no transitive dependency(i.e no non-prime attribute determining another non-prime attribute) so, it is in 3 NF also as well as every super key is present on the left side of the functional dependency so, it is in 3 NF.
- BCNF - Since, each and every functional dependency has super key on left therefore it is in Boyce–Codd Normal Form.





**TABLES -**

# Customer Table

```
CREATE TABLE Customer
(
    Cust_Id VARCHAR NOT NULL,
    Cust_Name VARCHAR NOT NULL,
    Cust_Age INT NOT NULL,
    Cust_Phone NUMERIC NOT NULL,
    PRIMARY KEY (cust_Id)
);
```

Cust_Id	Cust_Name	Cust_Age	Cust_Phone
P1	AMAN KUMAR	20	12345678
P2	ANSH RUSIA	20	23456789
P3	SHUBHAJEET PRADHAN	20	34567891
P4	VARUN KUMAR TIWARI	20	45678912

# Ticket Table

```
CREATE TABLE Ticket
(
    ticket_Id VARCHAR NOT NULL,
    price INT NOT NULL,
    Cust_Id VARCHAR NOT NULL,
    show_Id VARCHAR NOT NULL,
    PRIMARY KEY (ticket_Id),
    FOREIGN KEY (Cust_Id) REFERENCES Customer(Cust_Id),
    FOREIGN KEY (Show_Id) REFERENCES Shows(show_Id)
);
```

ticket_Id	price	Cust_Id	show_Id
TCK1	750	P1	SHW1
TCK2	300	P2	SHW2
TCK3	925	P3	SHW3
TCK4	1030	P4	SHW4

# Reservation Table

```
CREATE TABLE Reservation
(
  reservation_Id VARCHAR NOT NULL,
  ticket_Id VARCHAR NOT NULL,
  show_Id VARCHAR NOT NULL,
  seat_Id VARCHAR NOT NULL,
  PRIMARY KEY (reservation_Id),
  FOREIGN KEY (ticket_Id) REFERENCES Ticket(ticket_id),
  FOREIGN KEY (show_Id) REFERENCES Shows(show_id),
  FOREIGN KEY (seat_Id) REFERENCES Seat(seat_id)
);
```

! ticket_Id	price	Cust_Id	show_Id
TCK1	750	P1	SHW1
TCK2	300	P2	SHW2
TCK3	925	P3	SHW3
TCK4	1030	P4	SHW4

# Seat Table

```
CREATE TABLE seat
(
  seat_Id VARCHAR NOT NULL,
  seat_Number INT NOT NULL,
  seat_Row VARCHAR NOT NULL,
  ent_Id VARCHAR NOT NULL,
  PRIMARY KEY (seat_Id),
  FOREIGN KEY (ent_Id) REFERENCES movie_room(ent_Id)
);
```

seat_Id	seat_Number	seat_Row	ent_Id
E1S1	1	R1	ENT1
E1S2	2	R1	ENT1
E1S3	3	R2	ENT1
E1S4	4	R2	ENT1
E1S5	5	R3	ENT1
E2G1	1	R1	ENT2
E2G2	2	R1	ENT2
E2G3	3	R2	ENT2
E2G4	4	R2	ENT2
E2G5	5	R3	ENT2
E3E1	1	R1	ENT3
E3E2	2	R1	ENT3
E3E3	3	R2	ENT3
E3E4	4	R2	ENT3
E3E5	5	R3	ENT3

# Movie Room Table

```
CREATE TABLE movie_room
(
    ent_Id VARCHAR NOT NULL,
    total_Seats INT NOT NULL,
    room_Name VARCHAR NOT NULL,
    PRIMARY KEY (ent_Id)
);
```

ent_Id	total_Seats	room_Name
ENT1	5	Silver
ENT2	5	Gold
ENT3	5	Executive

# Show Table

```
CREATE TABLE shows
(
  show_Id VARCHAR NOT NULL,
  show_slot VARCHAR NOT NULL,
  show_Date DATE NOT NULL,
  ent_Id VARCHAR NOT NULL,
  movie_Name VARCHAR NOT NULL,
  PRIMARY KEY (show_Id),
  FOREIGN KEY (ent_Id) REFERENCES movie_room(ent_Id),
  FOREIGN KEY (movie_Name) REFERENCES movie(movie_Name)
);
```

! show_Id	show_slot	show_Date	ent_Id	movie_Name
SHW1	slotA	2021-09-07	ENT1	The Eternals
SHW2	slotB	2021-08-05	ENT2	Inception
SHW3	slotC	2021-06-11	ENT3	Iron-Man 2
SHW4	slotD	2021-06-23	ENT3	Iron-Man 2



# Movie Table

```
CREATE TABLE movie
(  
    movie_Name VARCHAR NOT NULL,  
    duration VARCHAR NOT NULL,  
    genre VARCHAR NOT NULL,  
    rating VARCHAR NOT NULL,  
    PRIMARY KEY (movie_Name)  
);
```

! movie_Name	duration	genre	rating
Inception	148	Thriller	5
Iron-Man 2	100	Sci-Fi	5
The Eternals	157	Sci-Fi	4



# **SQL Queries -**

1. Show all the details of people who booked movie for only "Executive Class".

```
SELECT cust_name, cust_phone, cust_age  
FROM customer NATURAL JOIN ticket NATURAL JOIN shows NATURAL JOIN movie_room  
WHERE room_name = 'Executive';
```

cust_name	cust_phone	cust_age
SHUBHAJEET PRADHAN	34567891	20
VARUN KUMAR TIWARI	45678912	20

(2 rows)

2. Show all the details of peoples who booked for a movie whose price is greater than 750.

```
SELECT cust_name, cust_phone, cust_age  
FROM customer NATURAL JOIN ticket  
WHERE price < 925;
```

cust_name	cust_phone	cust_age
AMAN KUMAR	12345678	20
ANSH RUSIA	23456789	20
(2 rows)		

**3. Show the age of all customers who are watching the movie "The Eternals".**

```
SELECT Cust_Age FROM Customer
WHERE Cust_Id =
(
  SELECT Cust_Id FROM Ticket
  WHERE show_id =
    (
      SELECT show_id FROM Shows
      WHERE movie_Name = 'The Eternals'
    )
);
```

```
cust_age
-----
          20
(1 row)
```

#### 4. Show seat id of customer whose name is "ANSH RUSIA".

```
SELECT seat_Id FROM Reservation
WHERE ticket_Id =
(
SELECT ticket_Id FROM ticket
WHERE Cust_Id =
(
SELECT Cust_Id FROM Customer
WHERE Cust_Name = 'ANSH RUSIA'
)
);
```

```
seat_id
-----
E1S2
(1 row)
```

## 5. Show all the Customers who is watching "Iron-Man 2" in "ENT 3"

```
SELECT cust_id, cust_name, cust_phone, cust_age, show_slot, price
FROM customer NATURAL JOIN ticket NATURAL JOIN shows
WHERE movie_name = 'Iron-Man 2' AND ent_id = 'ENT3';
```

cust_id	cust_name	cust_phone	cust_age	show_slot	price
P3	SHUBHAJEET PRADHAN	34567891	20	slotC	925
P4	VARUN KUMAR TIWARI	45678912	20	slotD	1030
(2 rows)					



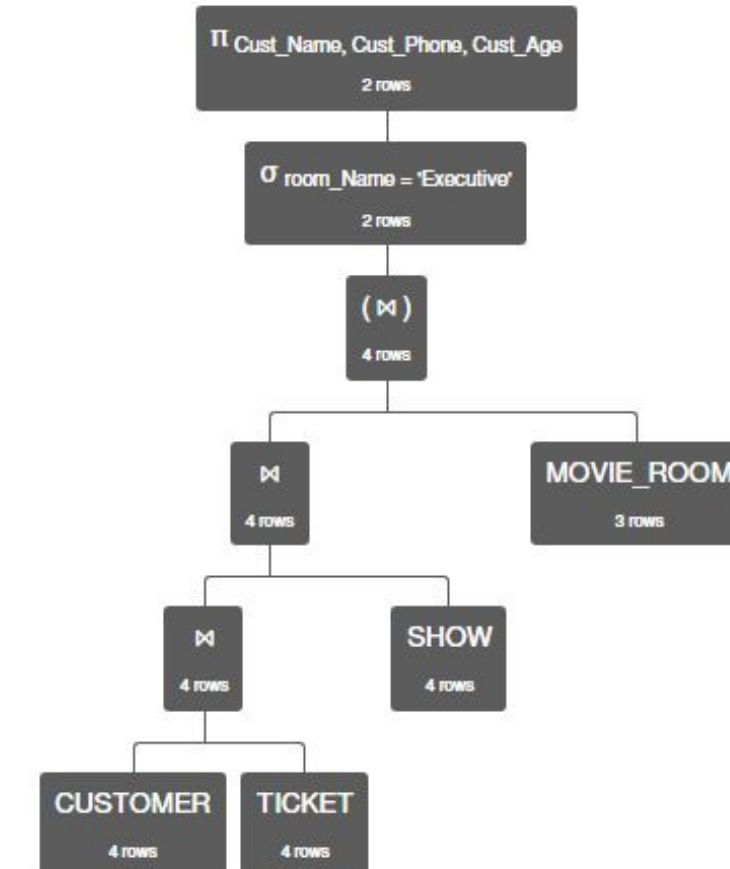


**RA Expression-**

## 1. Show all the details of people who booked movie for only "Executive Class".

```
 $\pi$  Cust_Name, Cust_Phone, Cust_Age (  $\sigma$  room_Name = 'Executive' ( CUSTOMER  $\bowtie$  TICKET  $\bowtie$  SHOW  $\bowtie$  MOVIE_ROOM ) )
```

CUSTOMER.Cust_Name	CUSTOMER.Cust_Phone	CUSTOMER.Cust_Age
'SHUBHAJEET PRADHAN'	34567891	20
'VARUN KUMAR TIWARI'	45678912	20



```
 $\pi$  Cust_Name, Cust_Phone, Cust_Age (  $\sigma$  room_Name = 'Executive' ( ( ( CUSTOMER  $\bowtie$  TICKET )  $\bowtie$  SHOW )  $\bowtie$  MOVIE_ROOM ) )
```

2. Show all the details of peoples who booked for a movie whose price is greater than 750.

```
 $\pi$  Cust_Name, Cust_Phone, Cust_Age (  $\sigma$  price < 925 ( CUSTOMER  $\bowtie$  TICKET ) )
```

CUSTOMER.Cust_Name	CUSTOMER.Cust_Phone	CUSTOMER.Cust_Age
'AMAN KUMAR'	12345678	20
'ANSH RUSIA'	23456789	20



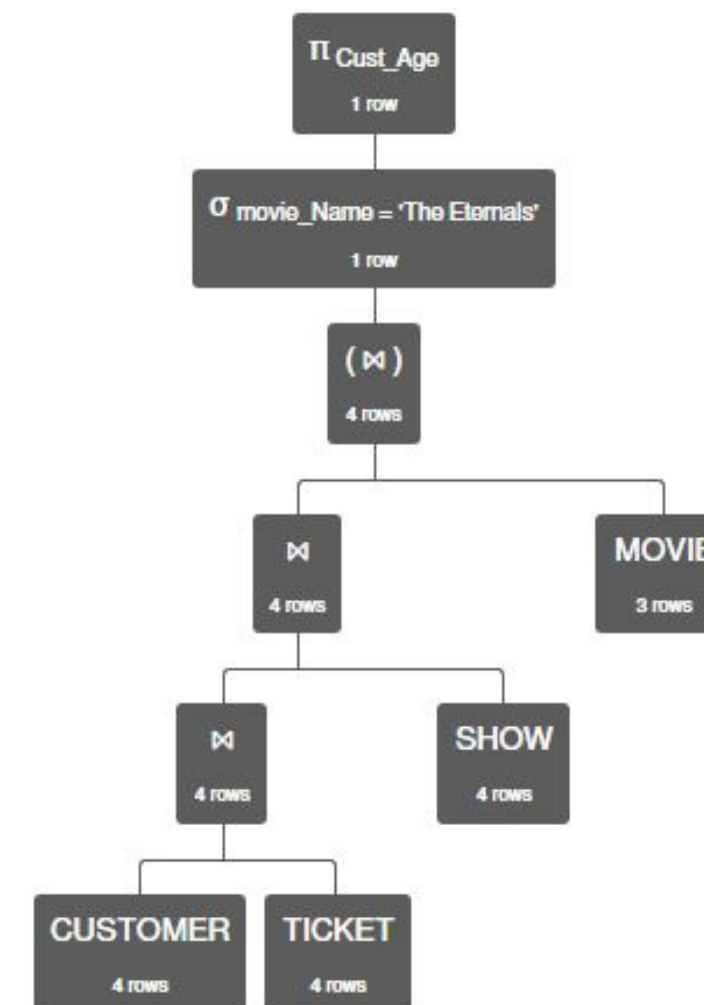
```
 $\pi$  Cust_Name, Cust_Phone, Cust_Age (  $\sigma$  price < 925 ( CUSTOMER  $\bowtie$  TICKET ) )
```

3. Show the age of all customers who are watching the movie "The Eternals".

$\pi_{Cust\_Age} (\sigma_{movie\_Name = 'The Eternals'} (CUSTOMER \bowtie TICKET \bowtie SHOW \bowtie MOVIE))$

CUSTOMER.Cust\_Age

20



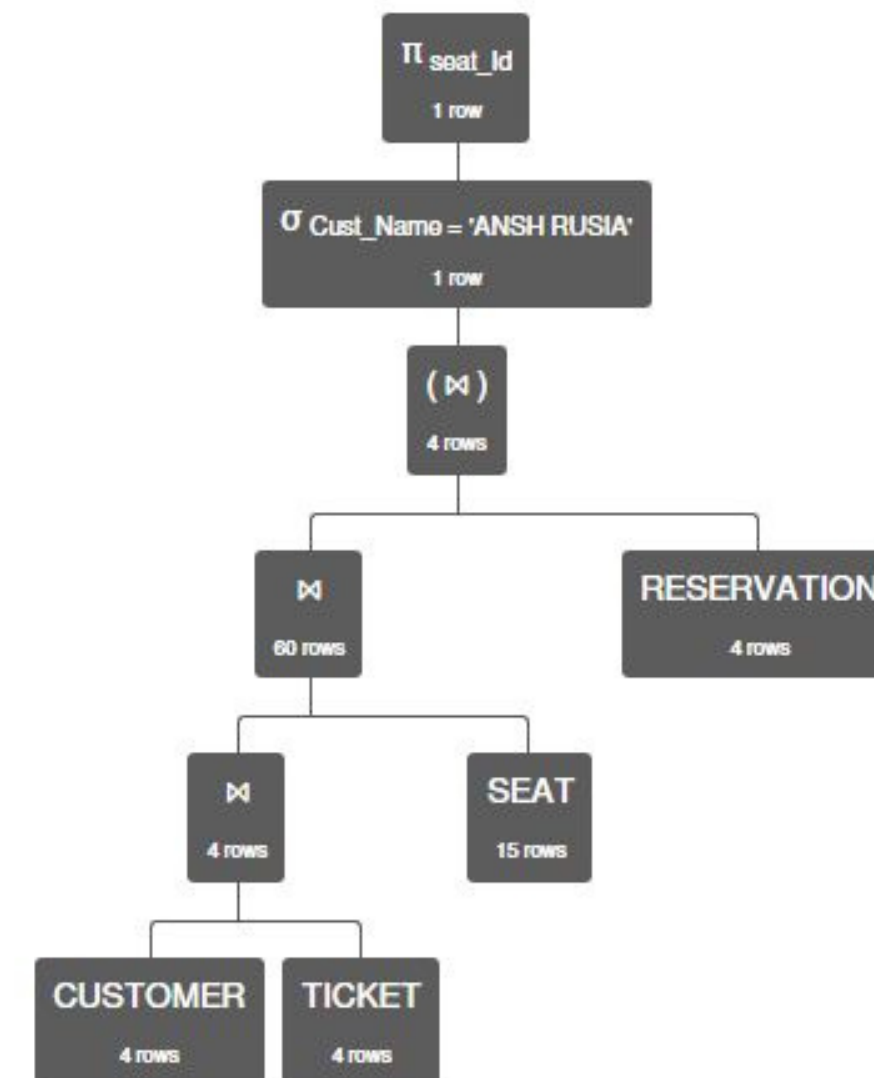
$\pi_{Cust\_Age} (\sigma_{movie\_Name = 'The Eternals'} (((CUSTOMER \bowtie TICKET) \bowtie SHOW) \bowtie MOVIE))$

4. Show seat id of customer whose name is "ANSH RUSIA".

```
 $\pi_{\text{seat\_Id}} (\sigma_{\text{Cust\_Name} = \text{'ANSH RUSIA'}} (\text{CUSTOMER} \bowtie \text{TICKET} \bowtie \text{SEAT} \bowtie \text{RESERVATION}))$ 
```

SEAT.seat\_Id

'E1S2'

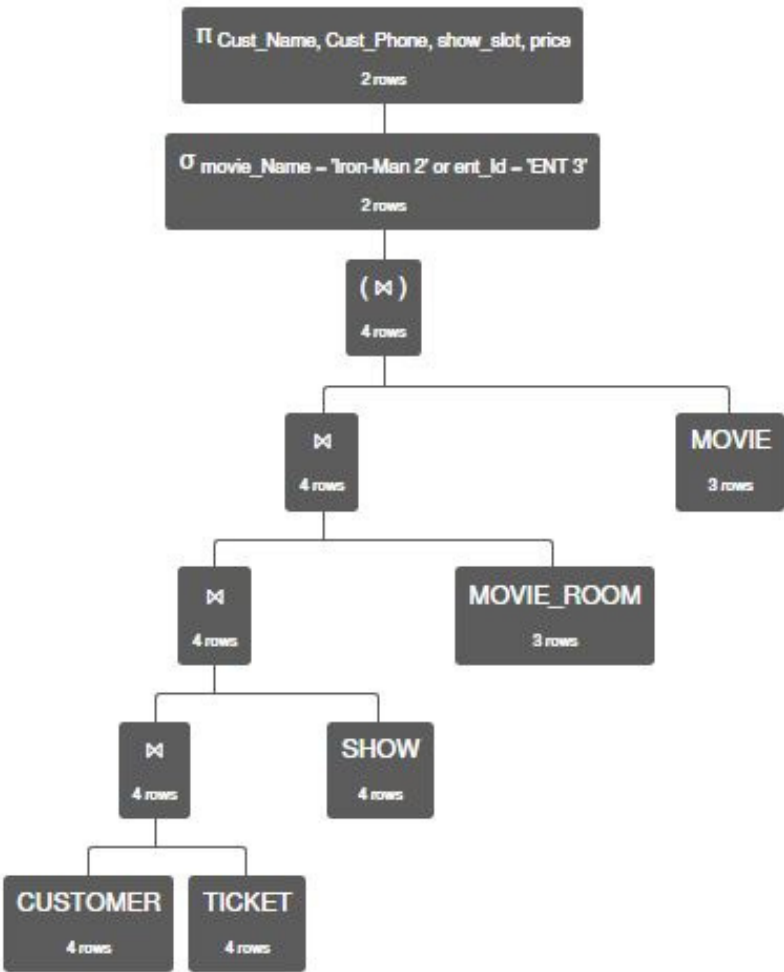


```
 $\pi_{\text{seat\_Id}} (\sigma_{\text{Cust\_Name} = \text{'ANSH RUSIA'}} (((\text{CUSTOMER} \bowtie \text{TICKET}) \bowtie \text{SEAT}) \bowtie \text{RESERVATION}))$ 
```

# 5. Show all the Customers who is watching "Iron-Man 2" in "ENT 3"

$\pi$  Cust\_Name, Cust\_Phone, show\_slot, price (  $\sigma$  movie\_Name = 'Iron-Man 2'  $\vee$  ent\_Id = 'ENT 3' (CUSTOMER  $\bowtie$  TICKET  $\bowtie$  SHOW  $\bowtie$  MOVIE\_ROOM  $\bowtie$  MOVIE))

CUSTOMER.Cust_Name	CUSTOMER.Cust_Phone	SHOW.show_slot	TICKET.price
'SHUBHAJEET PRADHAN'	34567891	'slotC'	925
'VARUN KUMAR TIWARI'	45678912	'slotD'	1030



$\pi$  Cust\_Name, Cust\_Phone, show\_slot, price (  $\sigma$  movie\_Name = 'Iron-Man 2' or ent\_Id = 'ENT 3' ( ( ( ( CUSTOMER  $\bowtie$  TICKET )  $\bowtie$  SHOW )  $\bowtie$  MOVIE\_ROOM )  $\bowtie$  MOVIE ) ) )