



**ABV-INDIAN INSTITUTE OF INFORMATION  
TECHNOLOGY AND MANAGEMENT  
GWALIOR-474 015**

**DATABASE MANAGEMENT SYSTEM  
MINI PROJECT**

**Movie-Ticket-Booking-System *by***

**AMAN KUMAR**

(2020IMT-007)

**ANSH RUSIA**

(2020IMT-012)

**SHUBHAJEET PRADHAN**

(2020IMT-097)

**VARUN KUMAR TIWARI**

(2020IMT-112)

*under the guidance and supervision of*

**Dr. DEBANJAN SADHYA ( Asstt. Professor)**

# Contents

<b>1</b>	<b>Introduction</b>	<b>3</b>
1.1	Description of the movie ticket booking system . . . . .	3
1.2	Benefits of our Database design . . . . .	3
<b>2</b>	<b>Schema of the Databases</b>	<b>4</b>
<b>3</b>	<b>ER diagram</b>	<b>4</b>
<b>4</b>	<b>Tables</b>	<b>5</b>
4.1	Customer Table . . . . .	5
4.1.1	Query . . . . .	5
4.1.2	Output Table . . . . .	5
4.2	Movie Table . . . . .	5
4.2.1	Query . . . . .	5
4.2.2	Output Table . . . . .	5
4.3	Movie Room Table . . . . .	6
4.3.1	Query . . . . .	6
4.3.2	Output Table . . . . .	6
4.4	Reservation Table . . . . .	6
4.4.1	Query . . . . .	6
4.4.2	Output Table . . . . .	6
4.5	Seat Table . . . . .	7
4.5.1	Query . . . . .	7
4.5.2	Output Table . . . . .	7
4.6	Shows Table . . . . .	8
4.6.1	Query . . . . .	8
4.6.2	Output Table . . . . .	8
4.7	Ticket Table . . . . .	8
4.7.1	Query . . . . .	8
4.7.2	Output Table . . . . .	8
<b>5</b>	<b>Normalisation</b>	<b>9</b>
5.1	Customer Table . . . . .	9
5.2	Movie Table . . . . .	9
5.3	Movie Room Table . . . . .	10
5.4	Reservation Table . . . . .	10
5.5	Seat Table . . . . .	11
5.6	Shows Table . . . . .	11
5.7	Ticket Table . . . . .	12
<b>6</b>	<b>Queries and Results</b>	<b>13</b>
6.1	Show all the details of people who booked movie for only "Executive Class".	13
6.1.1	Relational Algebra Expression . . . . .	13
6.1.2	WorkFlow . . . . .	13

6.1.3	R.A. Result Table . . . . .	14
6.1.4	SQL Query . . . . .	14
6.1.5	SQL Result Table . . . . .	14
6.2	Show all the details of peoples who booked for a movie whose price is less than 925. . . . .	15
6.2.1	Relational Algebra Expression . . . . .	15
6.2.2	WorkFlow . . . . .	15
6.2.3	R.A. Result Table . . . . .	16
6.2.4	SQL Query . . . . .	16
6.2.5	SQL Result Table . . . . .	16
6.3	Show the age of all customers who are watching the movie "The Eternals".	17
6.3.1	Relational Algebra Expression . . . . .	17
6.3.2	WorkFlow . . . . .	17
6.3.3	R.A. Result Table . . . . .	18
6.3.4	SQL Query . . . . .	18
6.3.5	SQL Result Table . . . . .	18
6.4	Show seat id of customer whose name is "ANSH RUSIA". . . . .	19
6.4.1	Relational Algebra Expression . . . . .	19
6.4.2	WorkFlow . . . . .	19
6.4.3	R.A. Result Table . . . . .	20
6.4.4	SQL Query . . . . .	20
6.4.5	SQL Result Table . . . . .	20
6.5	Show all the Customers who is watching "Iron-Man 2" in "ENT 3" . . .	21
6.5.1	Relational Algebra Expression . . . . .	21
6.5.2	WorkFlow . . . . .	21
6.5.3	R.A. Result Table . . . . .	22
6.5.4	SQL Query . . . . .	22
6.5.5	Result Table . . . . .	22

# 1 Introduction

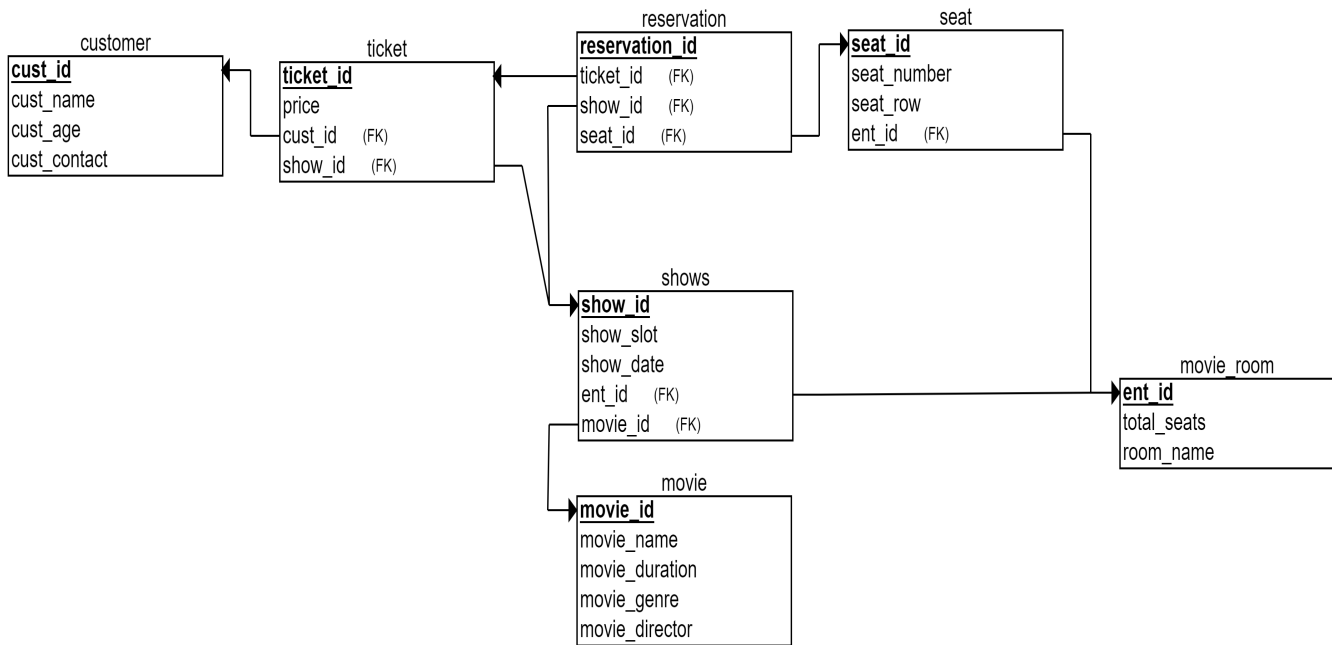
## 1.1 Description of the movie ticket booking system

- As the name suggests the movie ticket management system is a database management system for a multiplex. The database is designed to accommodate multiple theater rooms at same time to have a hassle free experience for the customer and the staff.
- The project is highly flexible and is well efficient for managing all information about the customer. The key focus is: well management of data and easy retrieval of information. The speed and accuracy should be maintained in a proper way.
- Due to faster output of data the system becomes efficient. Their is no manual searching of files and hence loss of data due to human error is less.

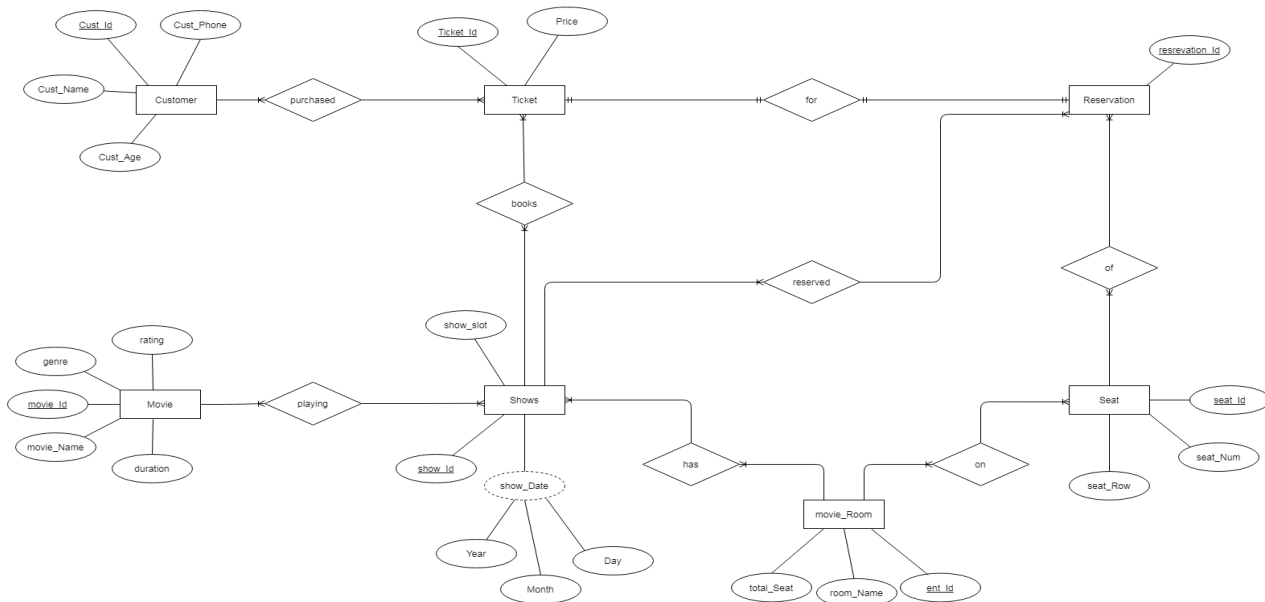
## 1.2 Benefits of our Database design

- Less human error
- Strength and strain of manual labour can be reduced
- Security
- Data redundancy can be avoided to some extent
- Data consistency
- Easy to handle
- Easy data updating
- Easy record keeping

## 2 Schema of the Databases



## 3 ER diagram



## 4 Tables

### 4.1 Customer Table

#### 4.1.1 Query

```
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)
);
```

#### 4.1.2 Output Table

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

### 4.2 Movie Table

#### 4.2.1 Query

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

#### 4.2.2 Output Table

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

## 4.3 Movie Room Table

### 4.3.1 Query

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

### 4.3.2 Output Table

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

## 4.4 Reservation Table

### 4.4.1 Query

```
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)
);
```

### 4.4.2 Output Table

! reservation_Id	ticket_Id	show_Id	seat_id
RE01	TCK1	SHW1	E2G4
RE02	TCK2	SHW2	E1S2
RE03	TCK3	SHW3	E3E5
RE04	TCK4	SHW4	E3E5

## 4.5 Seat Table

### 4.5.1 Query

```
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)
);
```

### 4.5.2 Output Table

seat_Id	seat_Num	seat_Row	ent_Id
E1S1	1	R1	E1
E1S2	2	R1	E1
E1S3	3	R2	E1
E1S4	4	R2	E1
E1S5	5	R3	E1
E2G1	1	R1	E2
E2G2	2	R1	E2
E2G3	3	R2	E2
E2G4	4	R2	E2
E2G5	5	R3	E2
E3E1	1	R1	E3
E3E2	2	R1	E3
E3E3	3	R2	E3
E3E4	4	R2	E3
E3E5	5	R3	E3



## 4.6 Shows Table

### 4.6.1 Query

```
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)
);
```

### 4.6.2 Output Table

! 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

## 4.7 Ticket Table

### 4.7.1 Query

```
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)
);
```

### 4.7.2 Output Table

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

## 5 Normalisation

### 5.1 Customer Table

Functional dependencies in this table are:

F.D. = {cust\_id  $\rightarrow$  cust\_age cust\_name cust\_phone}

Candidate key = cust\_id

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- {cust\_id  $\rightarrow$  cust\_name, cust\_age, cust\_phone} Since the left side of the FD is the super key, the relation follows BCNF.

### 5.2 Movie Table

Functional dependencies in this table are :

F.D. = { movie\_name  $\rightarrow$  room\_name total\_seats}

Candidate key = movie\_name

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- {movie\_name  $\rightarrow$  duration, genre, rating} Since the left side of the FD is the super key, the relation follows BCNF.

### 5.3 Movie Room Table

Functional dependencies in this table are :

F.D. = {  $\text{ent\_id} \rightarrow \text{room\_name} \text{ total\_seats}$  }

Candidate key =  $\text{ent\_id}$

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- { $\text{ent\_id} \rightarrow \text{total\_seats}, \text{room\_name}$ } Since the left side of the FD is the super key, the relation follows BCNF.

### 5.4 Reservation Table

Functional dependencies in this table are :

F.D. = {  $\text{reservation\_id} \rightarrow \text{ticket\_id}, \text{show\_id}, \text{seat\_id}$  }

Candidate key =  $\text{reservation\_id}$

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- { $\text{reservation\_id} \rightarrow \text{ticket\_id}, \text{show\_id}, \text{seat\_id}$ } Since the left side of the FD is the super key, the relation follows BCNF.

## 5.5 Seat Table

Functional dependencies in this table are :

F.D. = { seat\_id  $\rightarrow$  seat\_number, seat\_row, ent\_id }

Candidate key = seat\_id

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- {seat\_id  $\rightarrow$  seat\_number, seat\_row, ent\_id} Since the left side of the FD is the super key, the relation follows BCNF.

## 5.6 Shows Table

Functional dependencies in this table are :

F.D. = { show\_id  $\rightarrow$  show\_slot, show\_date, ent\_id, movie\_name }

Candidate key = show\_id

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- {show\_id  $\rightarrow$  show\_slot, show\_date, ent\_id, movie\_id} Since the left side of the FD is the super key, the relation follows BCNF

## 5.7 Ticket Table

Functional dependencies in this table are :

F.D. = { ticket\_id  $\rightarrow$  price, cust\_id, show\_id }

Candidate key = ticket\_id

- No multivalued attribute, hence the relation is in 1NF.
- Since there is only one attribute in the candidate key, all the non-key attributes are fully functional dependent on the primary key, and hence the relation is in 2NF form.
- Since there are no transitive dependencies present (no non-prime attribute derives other non-prime attributes), the relation is in 3NF.
- {ticket\_id  $\rightarrow$  price, cust\_id, show\_id} Since the left side of the FD is the super key, the relation follows BCNF.

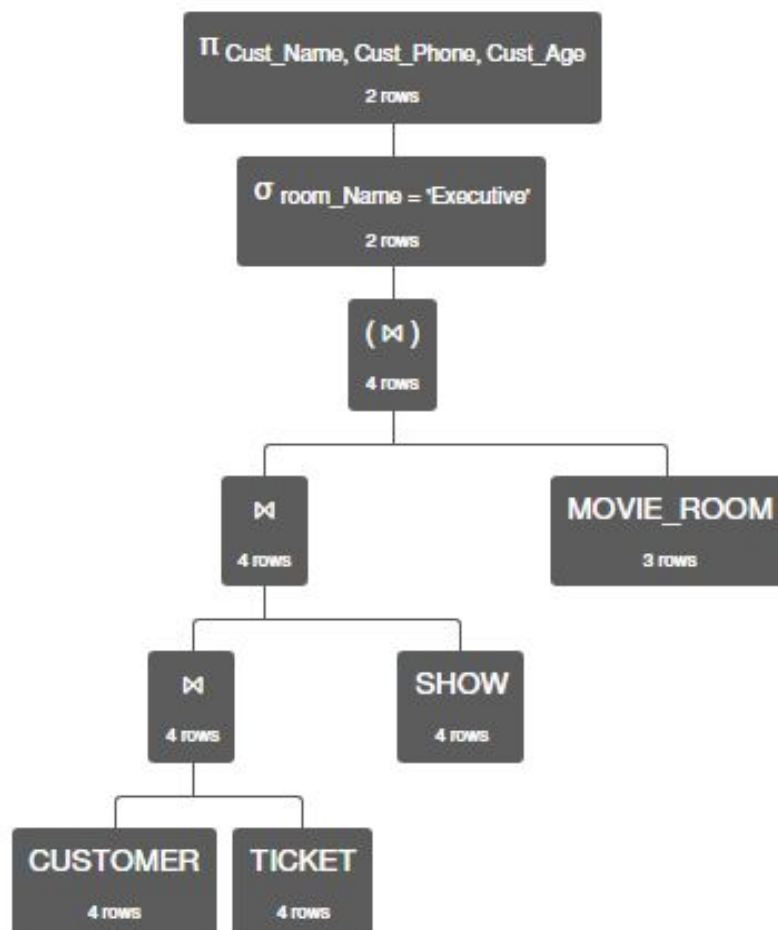
## 6 Queries and Results

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

### 6.1.1 Relational Algebra Expression

$\pi_{Cust\_Name, Cust\_Phone, Cust\_Age} (\sigma_{room\_Name = 'Executive'} (CUSTOMER \bowtie TICKET \bowtie SHOW \bowtie MOVIE\_ROOM))$

### 6.1.2 WorkFlow



$\pi_{Cust\_Name, Cust\_Phone, Cust\_Age} (\sigma_{room\_Name = 'Executive'} (((CUSTOMER \bowtie TICKET) \bowtie SHOW) \bowtie MOVIE\_ROOM))$

### 6.1.3 R.A. Result Table

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

### 6.1.4 SQL Query

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

### 6.1.5 SQL Result Table

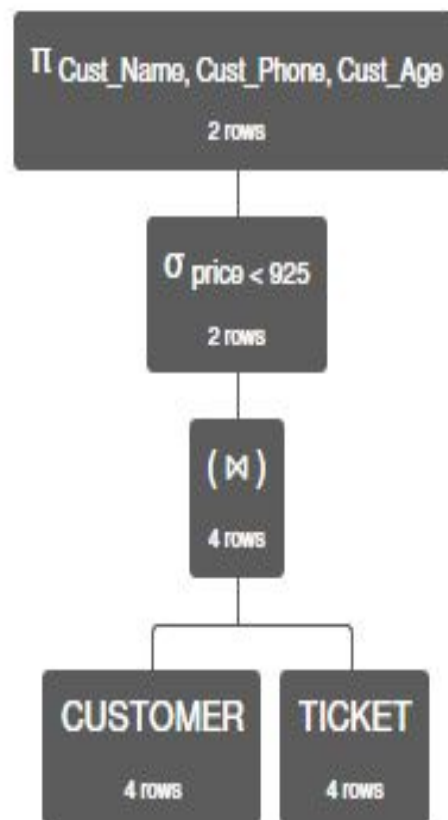
cust_name	cust_phone	cust_age
SHUBHAJEET PRADHAN	34567891	20
VARUN KUMAR TIWARI	45678912	20
(2 rows)		

6.2 Show all the details of peoples who booked for a movie whose price is less than 925.

### 6.2.1 Relational Algebra Expression

$\pi_{\text{Cust\_Name, Cust\_Phone, Cust\_Age}} (\sigma_{\text{price} < 925} (\text{CUSTOMER} \bowtie \text{TICKET}))$

### 6.2.2 WorkFlow



$\pi_{\text{Cust\_Name, Cust\_Phone, Cust\_Age}} (\sigma_{\text{price} < 925} (\text{CUSTOMER} \bowtie \text{TICKET}))$



### 6.2.3 R.A. Result Table

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

### 6.2.4 SQL Query

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

### 6.2.5 SQL Result Table

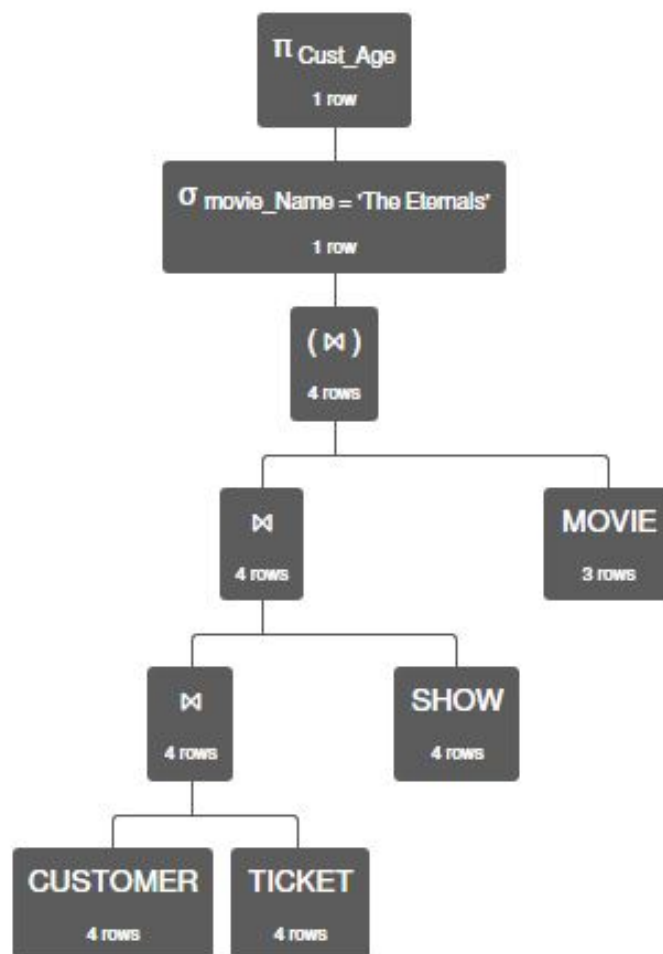
cust_name	cust_phone	cust_age
AMAN KUMAR	12345678	20
ANSH RUSIA	23456789	20
(2 rows)		

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

### 6.3.1 Relational Algebra Expression

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

### 6.3.2 WorkFlow



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

### 6.3.3 R.A. Result Table

CUSTOMER.Cust_Age
20

### 6.3.4 SQL Query

```
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'
    )
);
```

### 6.3.5 SQL Result Table

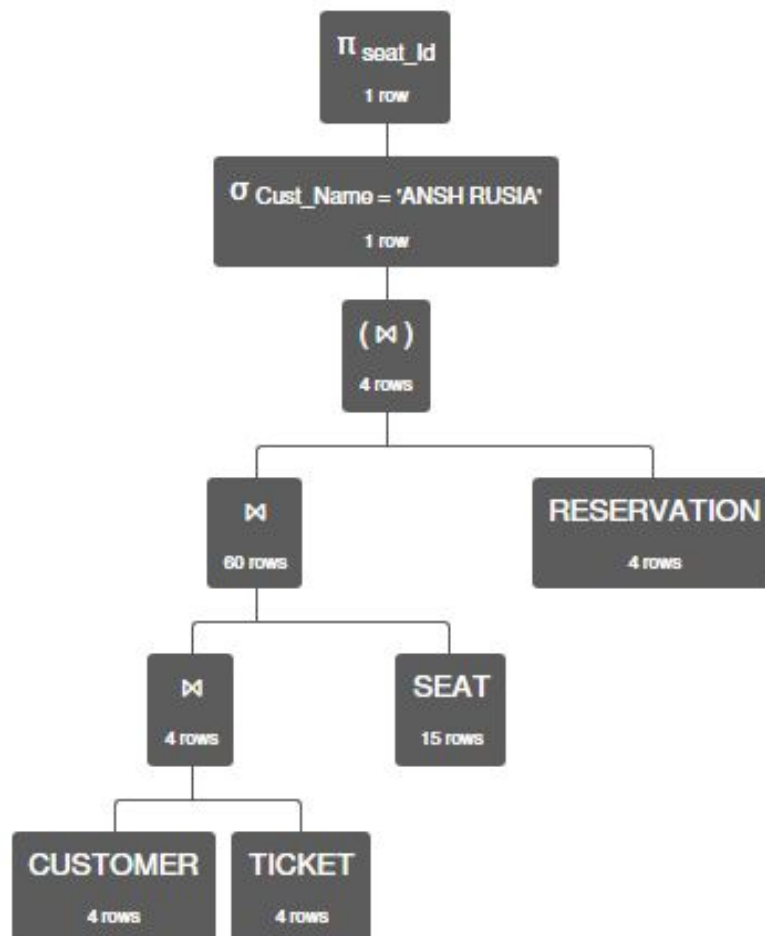
cust_age
20
(1 row)

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

#### 6.4.1 Relational Algebra Expression

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

#### 6.4.2 WorkFlow



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

#### 6.4.3 R.A. Result Table

SEAT.seat_Id
'E1S2'

#### 6.4.4 SQL Query

```
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'
    )
);
```

#### 6.4.5 SQL Result Table

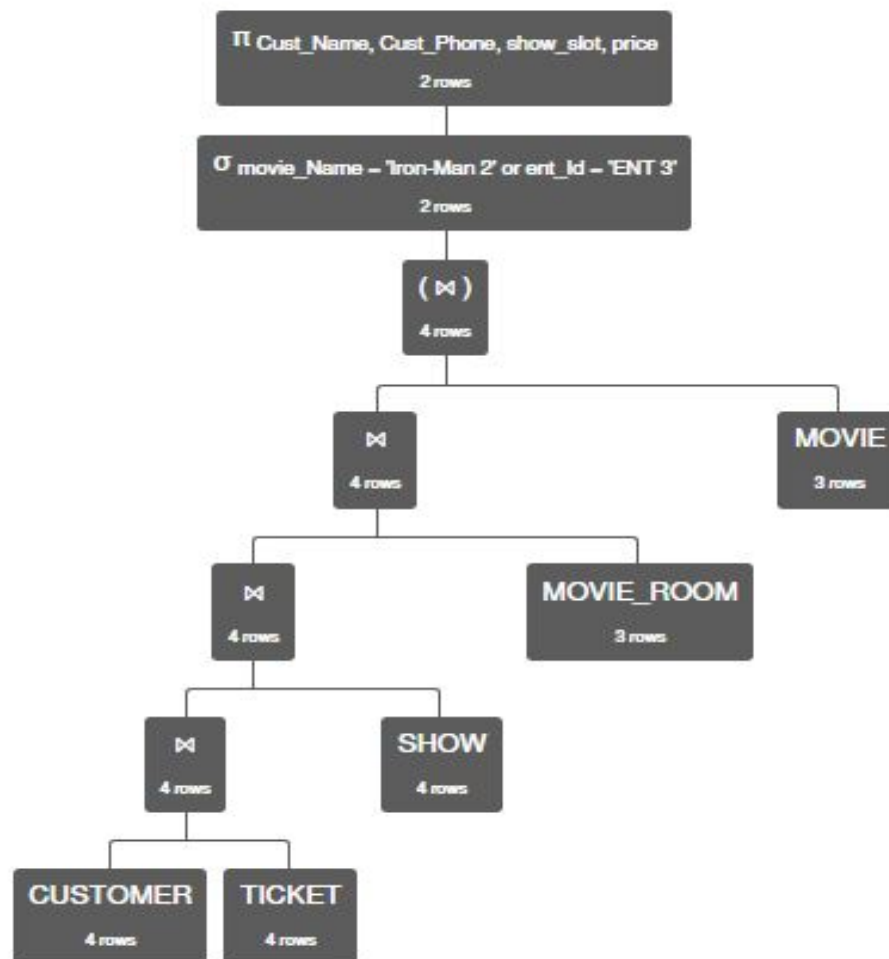
seat_id
E1S2
(1 row)

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

### 6.5.1 Relational Algebra Expression

```
 $\pi$  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))
```

### 6.5.2 WorkFlow



```
 $\pi$  Cust_Name, Cust_Phone, show_slot, price (  $\sigma$  movie_Name = 'Iron-Man 2' or ent_Id = 'ENT 3' ( ( ( (
```

---

### 6.5.3 R.A. Result Table

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

### 6.5.4 SQL Query

```
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';
```

### 6.5.5 Result Table

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)