UNIVERSITI TEKNOLOGI MALAYSIA

FACULTY OF COMPUTING, UTMJB

SEMESTER 1, SESSION 2023/2024

GROUP PROJECT

DATABASE

Phase 3 (P3) – Database Logical Design & SQL (20%)

SECD 2523 : DATABASE

SECTION 03

LECTURER'S NAME:

DR. IZYAN IZZATI BINTI KAMSANI

GROUP NAME:

TECHNO HUNTER

GROUP MEMBERS:

NAME	MATRIC NUMBER
1. MOHAMAD REZQI BIN MUHARAM	B23CS0006
2. MUHAMAD IDHAM BIN MOHAMAD RAZALI	B23CS0007
3. MUHAMMAD HABIEL WAFI BIN ZAIRI	B23CS0009
4. NUR FIRZANAH BINTI SHAMSHUL AZAM	B23CS0013
5. NUR BALQIS BINTI MOHD NASIR	B23CS0067

Table Of Contents

1.0 Introduction

In the rapidly evolving landscape of the entertainment industry, advancements in technology are reshaping how audiences experience movies. Recognizing the growing demand for a seamless cinema booking platform, this proposal outlines the strategic introduction of an Online Cinema Booking System. To achieve this, the conceptual Entity-Relationship Diagram (ERD) will undergo a transformation into a Logical ERD, involving the removal of non-relational features, derivation of a relational schema, and normalization up to BCNF. The resulting Logical ERD will then be depicted, and the data dictionary will be updated to reflect the refined data structure. Subsequently, the logical model will be precise examination validated against the system's transaction requirements, ensuring its alignment with practical needs and interface design, thus laying the foundation for an efficient and robust Online Cinema Booking System.

2.0 Overview Of Project

In response to the changing landscape of the entertainment industry, we propose the development of an Online Cinema Booking System. This project aims to simplify the movie-ticketing process and improve the overall cinema-going experience. We will transform a conceptual Entity-Relationship Diagram into a Logical ERD, derive a relational schema, and perform normalization. The key deliverables include a finalized Logical ERD, an updated data dictionary, and a validated system meeting transaction requirement. The project's significance lies in meeting the growing demand for a user-friendly cinema booking platform, with a dedicated team ensuring a streamlined and efficient solution.

The Online Cinema Booking System project addresses the need for a modernized and user-centric approach to movie-ticketing, aligning with the expectations of today's audiences. With a clear timeline and a focused scope, the project is poised to deliver a cutting-edge solution that enhances the cinema booking process and provides an improved experience for patrons.

3.0 Database Conceptual Design

Database conceptual design is the initial stage in the database design process where the overall structure and organization of the database are defined at an important level. It involves creating an abstract representation of the data and the relationships between different data entities without getting into the details of how the data will be physically stored or implemented in a specific database management system (DBMS). Moreover, the conceptual design phase is often a conceptual schema or an entity-relationship diagram (ERD) that visually represents the entities, relationships, and constraints in the system. This conceptual schema provides a foundation for the subsequent phases of database design, where the focus shifts to defining the detailed structure and implementation of the database based on the conceptual model.

3.1 Updated Business Rule

1. **Customer** - Each customer must have a unique CustomerID.
    - Each customer can have only one membership, linked through the MembershipID.
    - Only one receipt was received by one customer.
2. **Membership** - Each membership is associated with a unique MembershipID.
    - Each membership can has many customers.
3. **Receipts** – Each receipts information entry has a unique ReceiptsID. Payment information is linked to a specific customer through CustomerID.
4. **Movie** - Each movie has a unique MovieID.
    - One movie can have many showtime.
    - A movie can be showed in many halls.
5. **Booking -** Each booking has a unique BookingID.
    - Each booking can be made by only one customers.
    - Each booking can only book one movie at a time.
6. **Hall** - Each hall has a unique HallNumber. A hall may be played by one movie

3.2 Conceptual Entity-Relationship Diagram

A Conceptual Entity-Relationship Diagram (ERD) is a visual representation that illustrates the high-level relationships and entities within a system or a business domain. It's a fundamental step in the database design process and is used to capture the essential concepts and their associations in a clear and understandable way.

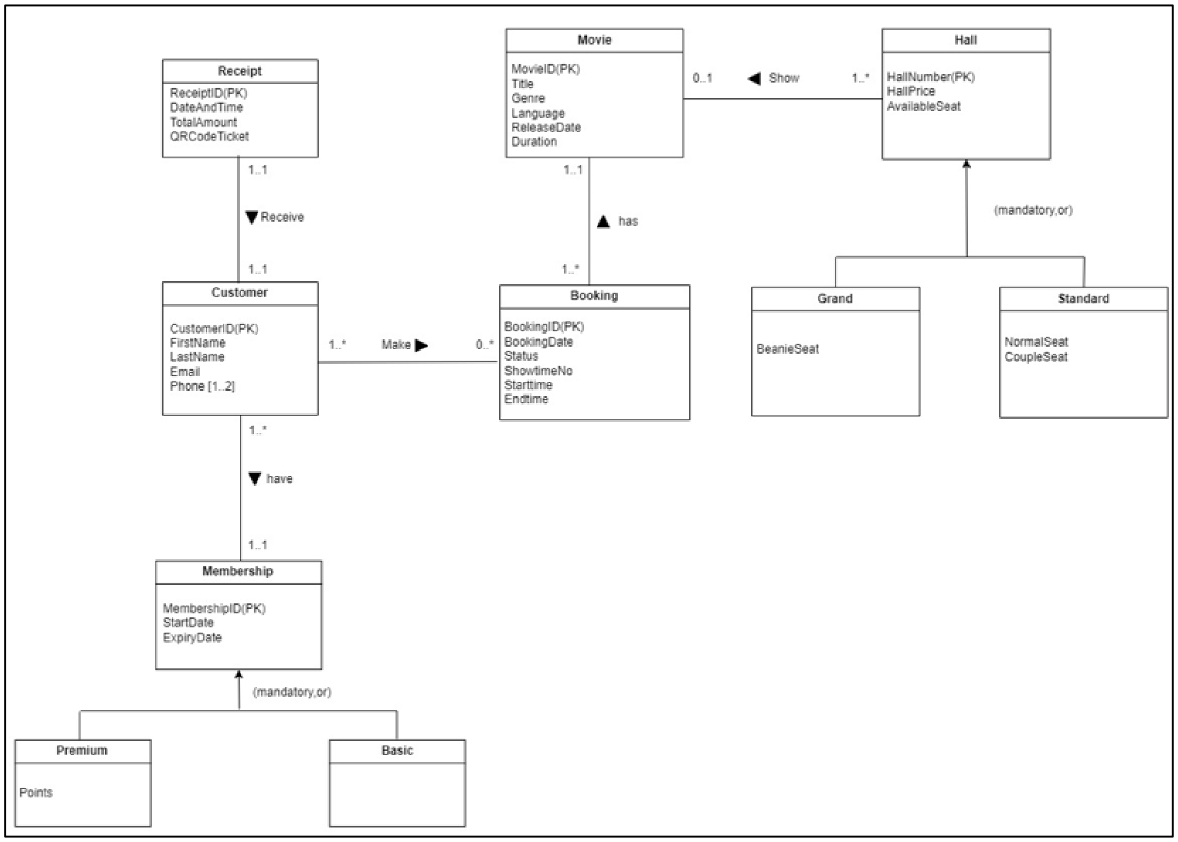


Figure 1: Conceptual ERD for The Cinema Booking System

4.0 Database Logical Design

Database logical design involves transforming a conceptual data model into a logical structure that can be implemented using a specific database management system (DBMS). The purpose is to create relations for the logical data model that represent the entities, relationships, and attributes identified in the conceptual data model (conceptual ERD- cERD).

4.1 Logical Entity-Relationship Diagram

A Logical Entity-Relationship Diagram (ERD) is a visual representation of the data model that represents the logical structure of a database system. It's a diagrammatic tool used during the logical design phase to illustrate the entities, attributes, and relationships in a database. The logical ERD provides a detailed view of how data is organized and related without specifying the physical implementation details.

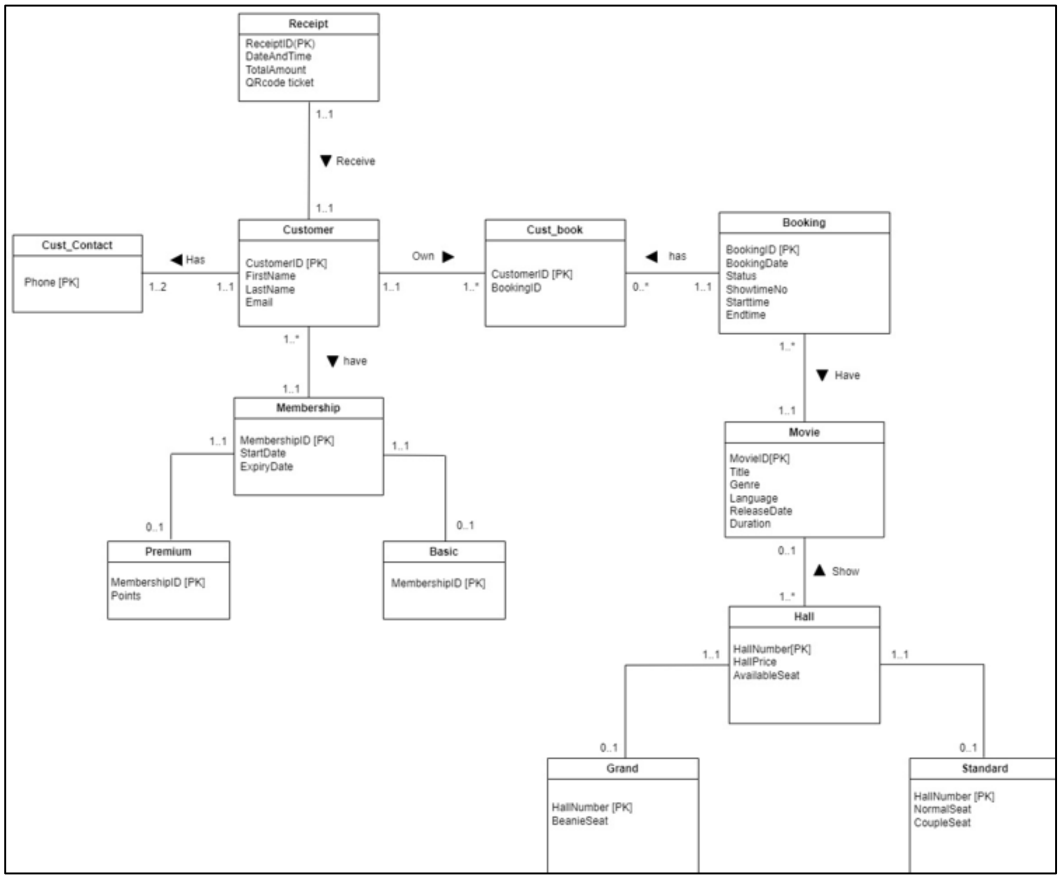


Figure 2: Logical ERD for The Cinema Booking System

4.1.1 Logical Design Relational Schemas

Step 1 – Strong Entity

1) Receipt (ReceiptID, DateAndTime, TotalAmount, QRCodeTicket)
   Primary Key : ReceiptID
2) Customer (CustomerID, FirstName, LastName, Email)
   Primary Key : CustomerID
3) Membership (MembershipID, StartDate, ExpiryDate)
   Primary Key : MemberhipID
4) Movie (MovieID, Title, Genre, Language, ReleaseDate, Duration)
   Primary Key : MovieID
5) Booking (BookingID, BookingDate, Status, ShowtimeNo, StartTime, Endtime)
   Primary Key : BookingID
6) Hall (HallNumber, HallPrice, AvailableSeat)
   Primary Key: HallNumber

Step 3 – One To Many

1) Relationship: Customer have Membership

   Parent → Membership

   Child → Customer

   - Membership (MembershipID, StartDate, ExpiryDate)
   - Cust_Member (CustomerID, FirstName, LastName, Email, Phone, *MembershipID*)
     Primary Key : MembershipID, CustomerID
     Foreign Key : MembershipID references Membership (MembershipID)

2) Relationship : Booking have Movie

   Parent → Movie

   Child → Booking

- Movie (<u>MovieID</u>, Title, Genre, Language, ReleaseDate, Duration)
- Booking (<u>BookingID</u>, BookingDate, Status, ShowtimeNo, StartTime, EndTime, *MovieID*)
  Primary Key : BookingID
  Foreign Key : MovieID references Movie (MovieID)

3) Relationship : Hall show Movie

   Parent → Movie

   Child → Hall

   - Movie (<u>MovieID</u>, Title, Genre, Language, ReleaseDate, Duration)
   - Hall (<u>HallNumber</u>, HallPrice, AvailableSeat, *MovieID*)
     Primary Key : MovieID, HallNumber
     Foreign Key : MovieID references Movie (MovieID)

Step 4 – One To One

- Cust_Receipt (*CustomerID*, FirstName, LastName, Email, Phone, *ReceiptID*, DateAndTime, TotalAmount, <u>QRCodeTicket</u>)
  Primary Key : QRCodeTicket
  Foreign Key : CustomerID references Customer (CustomerID)
  QRCodeTicket references Receipt (QRCode Ticket)

Step 5 – Superclass/Subclass

Superclass → Membership
Subclass → Premium, Basic

- Premium (<u>MembershipID</u>, StartDate, ExpiryDate, Points)
- Basic (<u>MembershipID</u>, StartDate, ExpiryDate)
  Primary Key : MembershipID

Superclass → Hall

Subclass → Grand, Standard

- Grand (<u>HallNumber</u>, HallPrice, AvailableSeat, BeanieSeat)
- Standard (<u>HallNumber</u>, HallPrice, AvailableSeat, NormalSeat, CoupleSeat)
  Primary Key : HallNumber

Step 6 – Many To Many

- Cust_Booking (CustomerID, BookingID)
  Primary Key : CustomerID, BookingID

Step 7 – Multi-valued

- Customer (CustomerID, FirstName, LastName, Email)
- Cust_Contact (*CustomerID*, Phone)
  Primary Key : CustomerID, Phone
  Foreign Key : CustomerID references Customer (CustomerID)

4.1.2 Finalize Logical Design Relational Schemas

1. Membership (MembershipID, StartDate, ExpiryDate)
2. Cust_Member (CustomerID, FirstName, LastName, Email, Phone, *MembershipID*)
3. Movie (MovieID, Title, Genre, Language, ReleaseDate, Duration)
4. Booking (BookingID, BookingDate, Status, ShowtimeNo, StartTime, EndTime, *MovieID*)
5. Hall (HallNumber, HallPrice, AvailableSeat, *MovieID*)
6. Cust_Receipt (QRCodeTicket, *CustomerID*, FirstName, LastName, Email, Phone, *ReceiptID*, DateAndTime, TotalAmount)
7. Premium (MembershipID, StartDate, ExpiryDate, Points)
8. Basic (MembershipID, StartDate, ExpiryDate)
9. Grand (HallNumber, HallPrice, AvailableSeat, BeanieSeat)
10. Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, CoupleSeat)
11. Cust_Booking (CustomerID, BookingID)
12. Customer (CustomerID, FirstName, LastName, Email)
13. Cust_Contact (Phone, *CustomerID*)

**REMARKS : Underline is Primary Key
: *Italic is* Foreign Key

4.2 Updated Data Dictionary

Membership

Field Name	Data Type	Field Length	Constraint	Description	Example
MembershipID	NUMBER	10	Primary Key	Customer membership id or unique key that auto generated	0001
StartDate	DATE	-	Not Null	Start date for the membership	2008-11-11
ExpiryDate	DATE	-	Not Null	Expiry date for the membership	2008-12-11

Cust_Member

Field Name	Data Type	Field Length	Constraint	Description	Example
CustomerID	NUMBER	10	Primary Key	Customer id or unique key that auto generated	1234
FirstName	VARCHAR2	20	Not Null	First name of customer	Abu
LastName	VARCHAR2	20	Not Null	Last name of customer	Hakiem
Email	VARCHAR2	30	Not Null	Email id for customer	Abu12@yahoo.com
Phone	VARCHAR2	12	Not Null	Landline or phone number	01234567
MembershipID	NUMBER	10	Foreign Key	Customer membership id or unique key that auto generated	0001

<u>Movie</u>

Field Name	Data Type	Field Length	Constraint	Description	Example
MovieID	NUMBER	10	Primary Key	Movie id or unique key that auto generated	002
Title	VARCHAR2	30	Not Null	Movie title	Upin & Ipin Kembara
Genre	VARCHAR2	15	Not Null	Movie genre	Horror
Language	VARCHAR2	15	Not Null	Movie language	English
ReleaseDate	DATE	-	Not Null	Release date for movie	2008-12-11
Duration	VARCHAR2	20	Not Null	Duration for movie	1 Hour 45 Minutes

Booking

Field Name	**Data Type**	**Field Length**	**Constraint**	**Description**	**Example**
BookingID	NUMBER	10	Primary Key	Customer booking id	0091
BookingDate	DATE	-	Not Null	Date of booking	2008-11-11
Status	VARCHAR2	50	Not Null	If status available, showtime will display on the page	Available or unavailable
ShowTimeNo	NUMBER	2	Not null	Linked to start time and end time	1
StartTime	TIMESTAMP	-	Not Null	Show start time	2008-11-11 13:23:44
EndTime	TIMESTAMP	-	Not Null	Show end time	2008-11-11 13:23:44
MovieID	NUMBER	10	Foreign Key	Movie id or unique key that auto generated	002

Hall

Field Name	**Data Type**	**Field Length**	**Constraint**	**Description**	**Example**
HallNumber	VARCHAR2	3	Primary Key	Movie hall number	A01
HallPrice	DECIMAL	6,2	Not Null	Price for ticket according to hall type	21.90
AvailableSeats	NUMBER	10	Not Null	Available seats for the movie	5
MovieID	NUMBER	10	Foreign Key	Movie id or unique key that auto generated	002

Cust_Receipt

Field Name	Data Type	Field Length	Constraint	Description	Example
CustomerID	NUMBER	10	Foreign Key	Customer id or unique key that auto generated	1234
FirstName	VARCHAR2	20	Not Null	First name of customer	Abu
LastName	VARCHAR2	20	Not Null	Last name of customer	Hakiem
Email	VARCHAR2	30	Not Null	Email id for customer	Abu12@yahoo.com
Phone	NUMBER	12	Not Null	Landline or phone number	01234567
ReceiptID	VARCHAR2	10	Foreign Key	Customer receipt id	A01MY220
DateAndTime	TIMESTAMP	-	Not Null	Date and time for receipt	2008-11-11 13:23:44
TotalAmount	DECIMAL	6,2	Not Null	Total amount of payment	90
QRCodeTicket	NUMBER	11	Primary Key	Code ticket after made payment	32436899450

Premium

Field Name	Data Type	Field Length	Constraint	Description	Example
MembershipID	NUMBER	10	Primary Key	Customer membership id or unique key that auto generated	0001
StartDate	DATE	-	Not Null	Start date for the membership	2008-11-11
ExpiryDate	DATE	-	Not Null	Expiry date for the membership	2008-12-11
Points	NUMBER	5	Not Null	Points for members when buy online ticket	10000

Basic

Field Name	Data Type	Field Length	Constraint	Description	Example
MembershipId	NUMBER	10	Primary Key	Customer membership id or unique key that auto generated	0001
StartDate	DATE	-	Not Null	Start date for the membership	2008-11-11
ExpiryDate	DATE	-	Not Null	Expiry date for the membership	2008-12-11

Grand

Field Name	Data Type	Field Length	Constraint	Description	Example
HallNumber	VARCHAR2	3	Primary Key	Movie hall number	1
HallPrice	DECIMAL	6,2	Not Null	Price for ticket according to hall type	21.90

AvailableSeats	NUMBER	10	Not Null	Available seats for the movie	5
BeanieSeat	VARCHAR2	5	Not Null	Seat Type	E1

Standard

Field Name	Data Type	Field Length	Constraint	Description	Example
HallNumber	VARCHAR2	3	Primary Key	Movie hall number	0101
HallPrice	DOUBLE	6,2	Not Null	Price for ticket according to hall type	21.90
AvailableSeats	NUMBER	10	Not Null	Available seats for the movie	5
NormalSeat	VARCHAR2	5	Not Null	Seat type	G6
CoupleSeat	VARCHAR2	5	Not Null	Seat type	J8

Cust_Booking

Field Name	Data Type	Field Length	Constraint	Description	Example
CustomerID	NUMBER	10	Primary Key	Customer id or unique key that auto generated	1234
BookingID	NUMBER	10	Primary Key	Customer booking id	0091

Customer

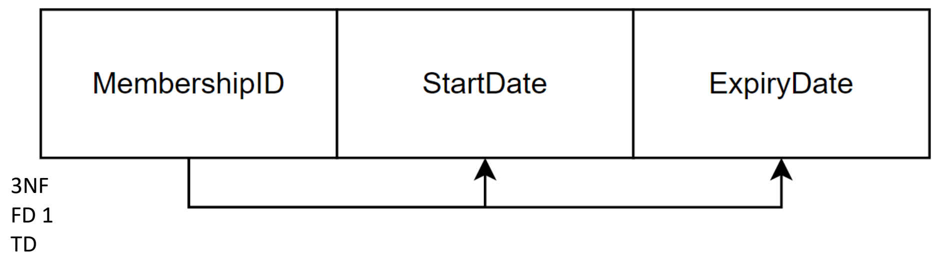
Field Name	Data Type	Field Length	Constraint	Description	Example
CustomerID	NUMBER	10	Primary Key	Customer id or unique key that auto generated	1234
FirstName	VARCHAR2	20	Not Null	First name of customer	Abu
LastName	VARCHAR2	20	Not Null	Last name of customer	Hakiem
Email	VARCHAR2	30	Not Null	Email id for customer	Abu12@yahoo.com

Cust_Contact

Field Name	Data Type	Field Length	Constraint	Description	Example
CustomerID	NUMBER	10	Foreign Key	Customer id or unique key that auto generated	1234
Phone	VARCHAR2	12	Primary Key	Landline or phone number	01234567

4.3 Normalization

MembershipID

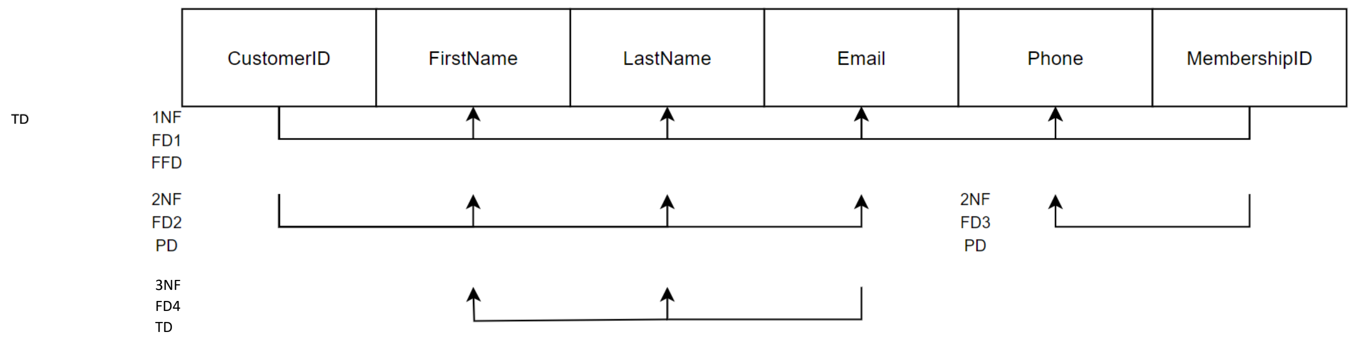


3NF

Membership (MembershipID, StartDate, ExpiryDate)

PK: MembershipID

Customer_Member



1NF

Customer_Member (CustomerID, FirstName, LastName, Email, Phone, MembershipID)

PK: CustomerID, MembershipID

2NF

CustomerInfo (CustomerID, FirstName, LastName, Email)

PK: CustomerID

MemberInfo(MembershipID, Phone)

PK: MembershipID

Customer_Member (CustomerID, MembershipID)

PK: CustomerID, MembershipID

FK: CustomerID references CustomerInfo(CustomerID)

FK: MembershipID references MemberInfo(MembershipID)

3NF

EmailInfo(Email, FirstName, LastName)

PK: Email

CustomerInfo(CustomerID. Email)

PK: CustomerID

MemberInfo(MembershipID, Phone)

PK: MembershipID

Customer_Member (CustomerID, MembershipID, Email)

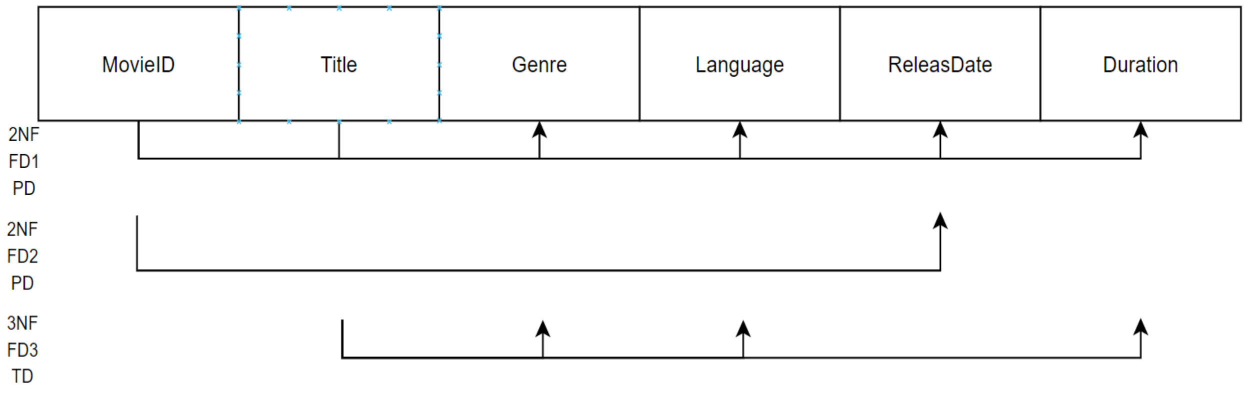
PK: CustomerID, MembershipID, Email

FK: Email reference EmailInfo (Email)

FK: CustomerID reference CustomerInfo (CustomerID)

FK: MembershipID reference MemberInfo (MembershipID)

Movie



2NF

Movie (MovieID, Title, Genre, Language, ReleasDate, Duration)

PK: MovieID

Movie (MovieID)

PK: MovieID

FK: MovieID reference Movie (MovieID)

3NF

MovieRelease (MovieID, ReleasDate)

PK: MovieID

MovieDetails (Title, Genre, Language, Duration)

PK: Title

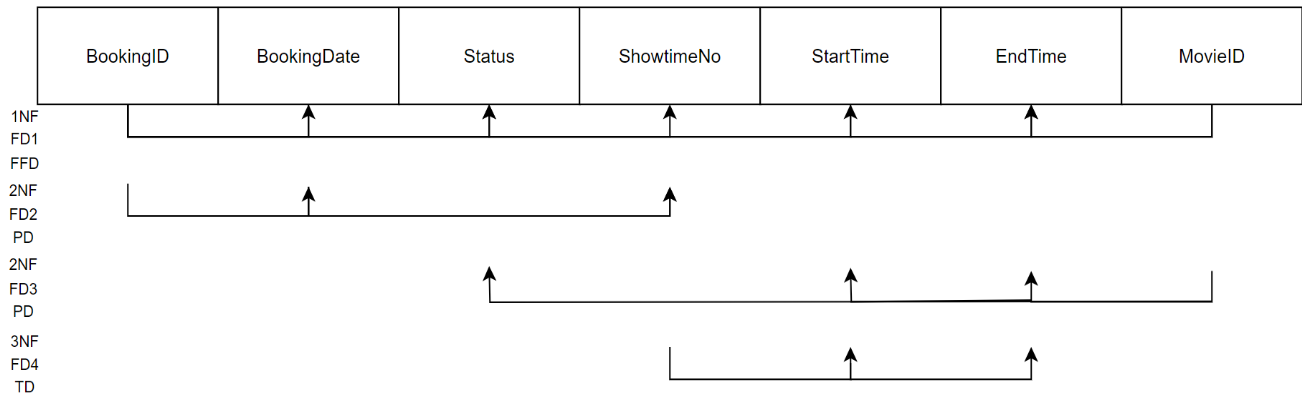
Movie (MovieID, Title)

PK: MovieID, Title

FK: MovieID reference MovieRealease (MovieID)

FK: Title reference MovieDetails (Title)

Booking



1NF

Booking (BookingID, BookingDate, Status, ShowTimeNo, StartTime,EndTime,MovieID)

PK: BookingID, MovieID

2NF

BookInfo (BookingID, BookingDate, ShowTimeNo)

PK: BookingID

MovieInfo (MovieID, Status, StartTime, EndTime)

PK: MovieID

Booking (BookingID, MovieID)

PK: BookingID, MovieID

FK: BookingID references BookInfo(BookID)

FK: MovieID references MovieInfo(MovieID)

3NF

Showtime (ShowTimeNo, StartTime, EndTime)

PK: ShowTimeNo

MovieInfo (MovieID, Status)

PK: MovieID

BookInfo (BookingID, BookingDate, ShowTimeNo)

PK: BookingID

Booking (BookingID, MovieID, ShowTimeNo)

PK: BookingID, MovieID, ShowTimeNo

FK : ShowtimeNo references Showtime (ShowtimeNo)

Booking (BookingID, MovieID, ShowTimeNo)

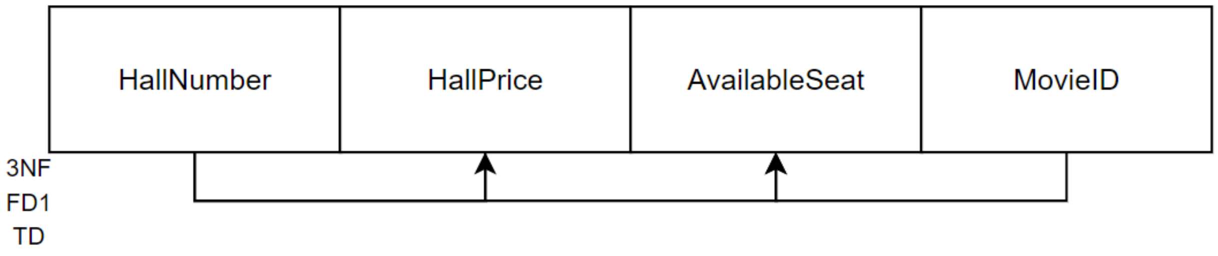
PK: BookingID, MovieID, ShowTimeNo

FK: ShowTimeNo reference Showtime(ShowTimeNo)

FK: MovieID reference MovieInfo(MovieID)

FK: BookingID reference BookInfo(BookingID)

Hall

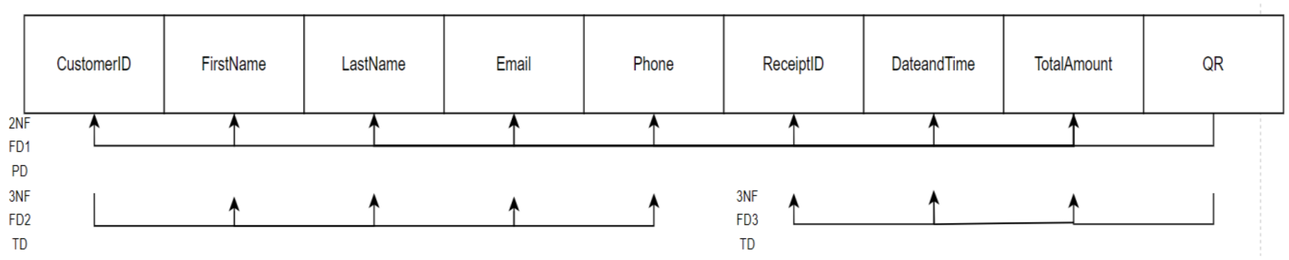


3NF

Hall (HallNumber, HallPrice, AvailableSeat, MovieID)

PK: HallNumber

Cust_Receipt



<u>2NF</u>

Cust_Receipt (<u>QRCodeTicket</u>, CustomerID, FirstName, LastName, Email, Phone, ReceiptID, DateandTime,TotalAmount)

PK: QRCodeTicket

<u>3NF</u>

Cust_Paid(<u>CustomerID</u>, FirstName, LastName, Email, Phone)

PK: CustomerID

Receipt (<u>QRCodeTicket</u>, ReceiptID, DateandTime, TotalAmount)

PK: QRCodeTicket

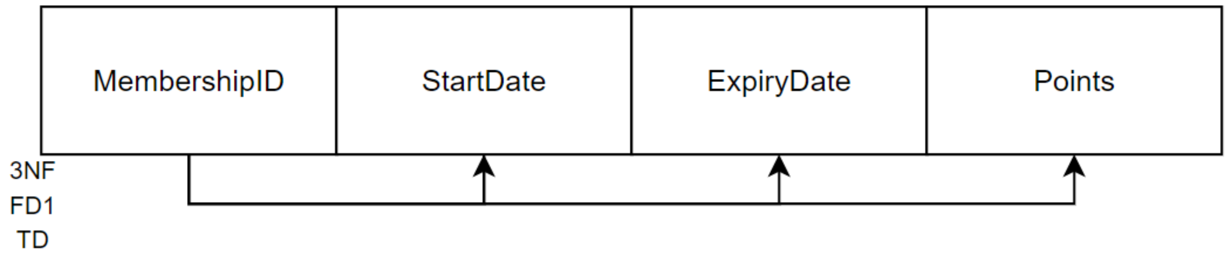
Cust_Receipt (<u>CustomerID,</u> <u>QRCodeTicket</u>)

PK: CustomerID, QRCodeTicket

FK: CustomerID reference Customer(CustomerID)

FK: QRCodeTicket reference Receipt(QRCodeTicket)

Premium

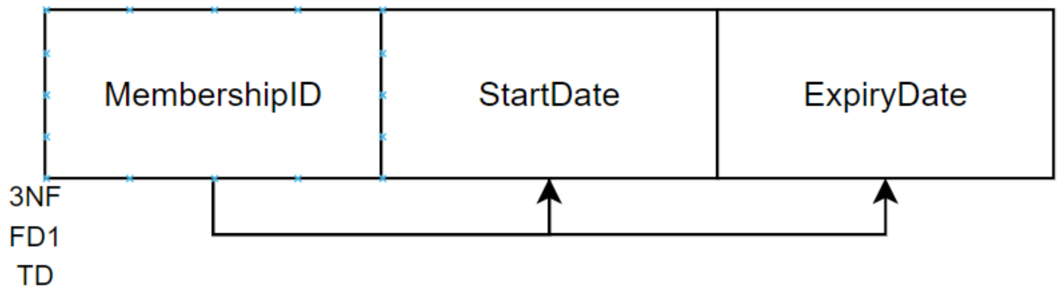


<u>3NF</u>

Premium (<u>MembershipID</u>, StartDate, ExpiryDate, Points)

PK: MembershipID

Basic

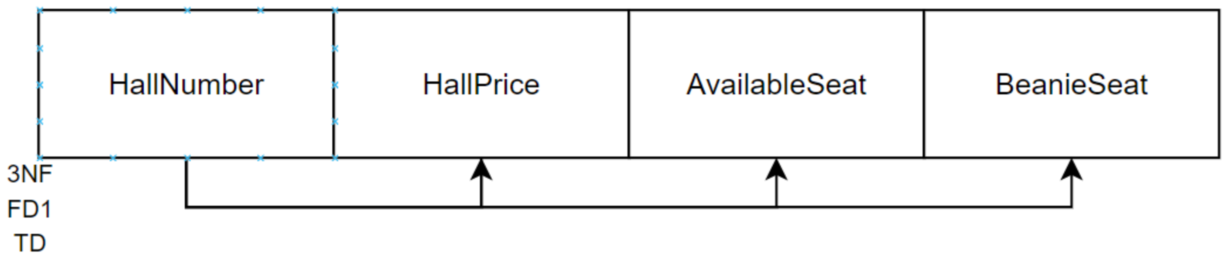


<u>3NF</u>

Basic (<u>MembershipID,</u> StartDate, ExpiryDate)

PK: MembershipID

Grand

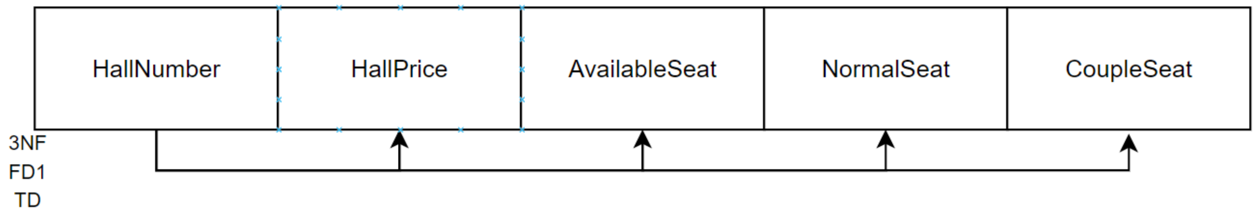


<u>3NF</u>

Grand (<u>HallNumber</u>, Hallprice, AvailableSeat, BeanieSeat)

PK: HallNumber

Standard

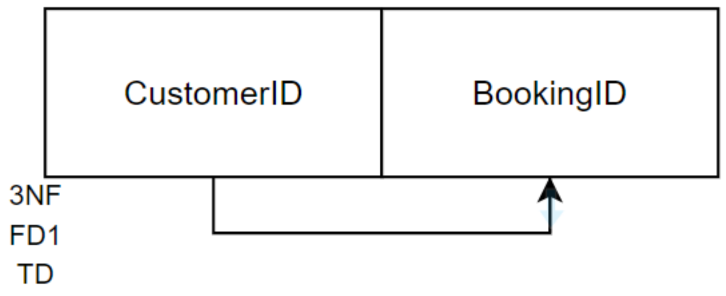


<u>3NF</u>

Standard (<u>HallNumber</u>, HallPrice, AvailableSeat, NormalSeat, CoupleSeat)

PK: HallNumber

Cust_booking

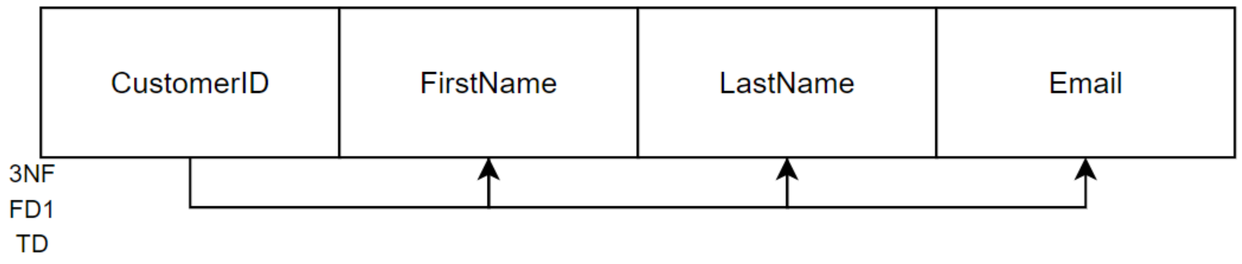


<u>3NF</u>

Cust_booking (<u>CustomerID,</u> BookingID)

PK: CustomerID

Customer

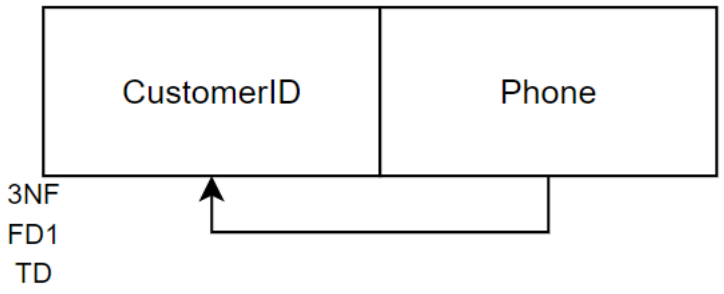


<u>3NF</u>

Customer (<u>CustomerID</u>, FirstName, LastName, Email)

PK: CustomerID

Cust_contact



<u>3NF</u>

Cust_contact (<u>Phone,</u> CustomerID)

PK: Phone

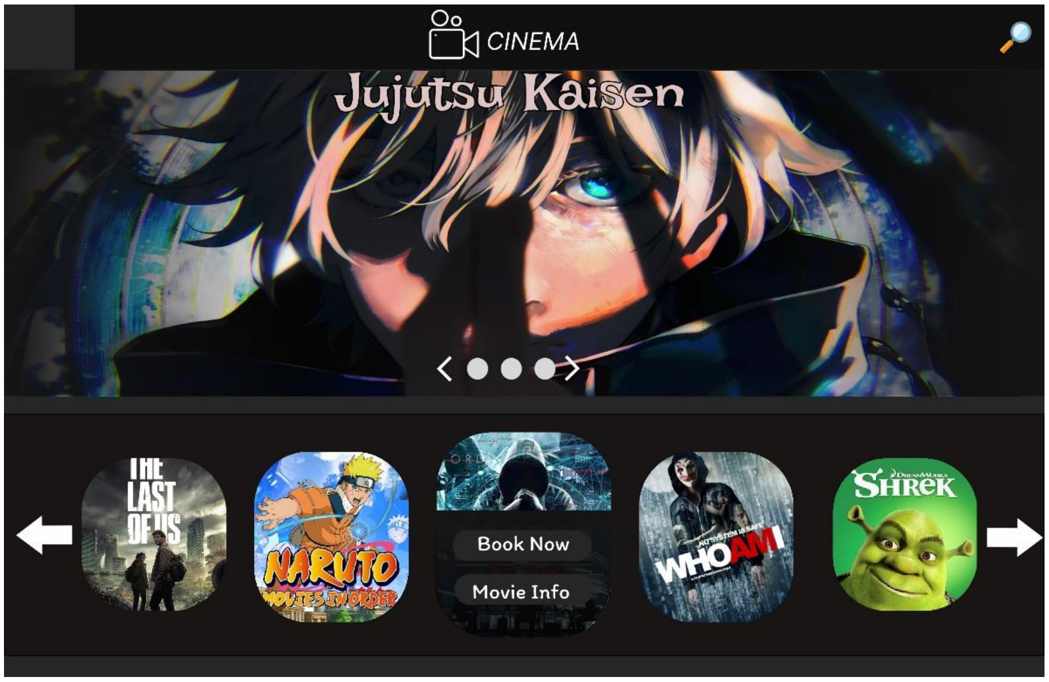
5.0 Relational Database Schemas (After Normalization)

1) Membership (MembershipID, StartDate, ExpiryDate)
2) Email_Info(Email, FirstName, LastName)
3) Cust_Info(CustomerID, Email)
4) MemberInfo(MembershipID, Phone)
5) Cust_Member (CustomerID, MembershipID, Email)
6) MovieRelease (MovieID, ReleaseDate)
7) MovieDetails (Title, Genre, Language, Duration)
8) Movie (MovieID, Title)
9) Showtime (ShowTimeNo, StartTime, EndTime)
10) MovieInfo (MovieID, Status)
11) BookInfo (BookingID, BookingDate, ShowTimeNo)
12) Booking (BookingID, MovieID, ShowTimeNo)
13) Hall (HallNumber, HallPrice, AvailableSeat, MovieID)
14) Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
15) Receipt (QRCodeTicket, ReceiptID, DateandTime, TotalAmount)
16) Cust_Receipt (CustomerID, QRCodeTicket)
17) Premium (MembershipID, StartDate, ExpiryDate, Points)
18) Basic (MembershipID, StartDate, ExpiryDate)
19) Grand (HallNumber, HallPrice, AvailableSeat, BeanieSeat)
20) Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, CoupleSeat)
21) Cust_booking (CustomerID, BookingID)
22) Customer (CustomerID, FirstName, LastName, Email)
23) Cust_contact (Phone, CustomerID)

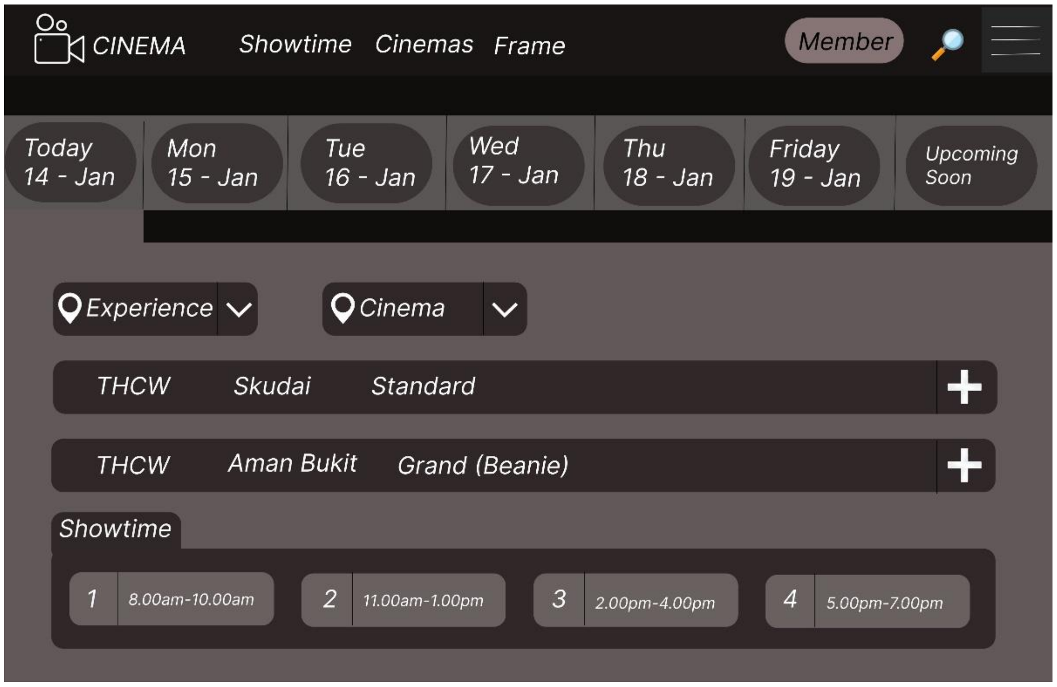
**REMARKS : Underline is Primary Key

6.0 Mock UI Pages

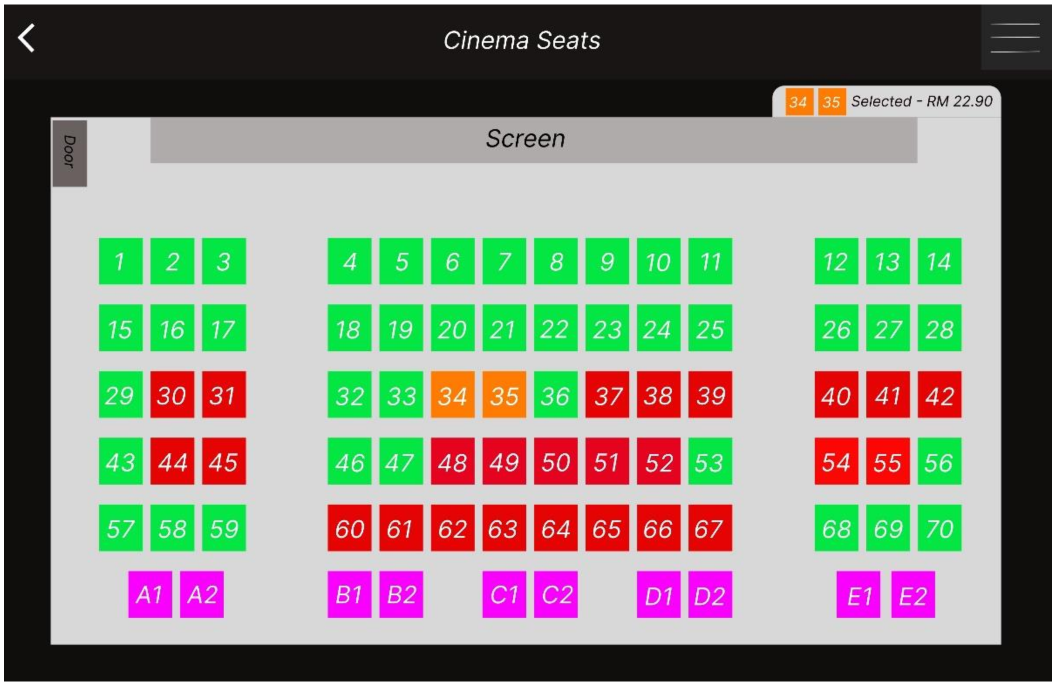
Home Page



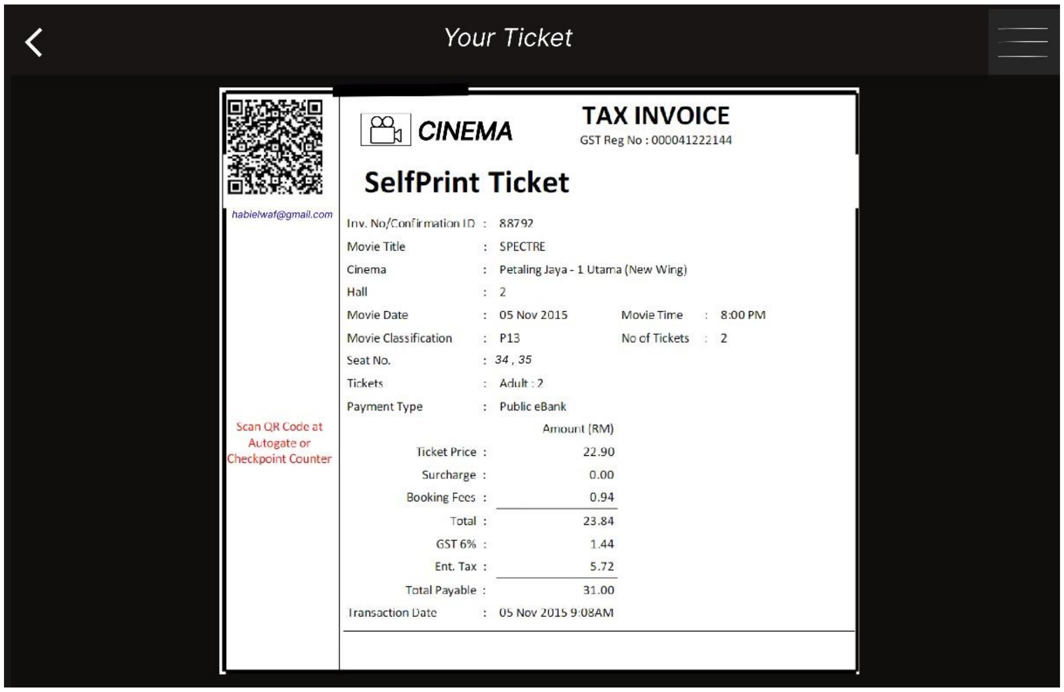
Booking Page



Seat Choosing Page



Receipt Page



CINEMA

TAX INVOICE

GST Reg No : 000041222144

SelfPrint Ticket

habielwaf@gmail.com

Inv. No/Confirmation ID	: 88792			
Movie Title	: SPECTRE			
Cinema	: Petaling Jaya - 1 Utama (New Wing)			
Hall	: 2			
Movie Date	: 05 Nov 2015	Movie Time	:	8:00 PM
Movie Classification	: P13	No of Tickets	:	2
Seat No.	: 34 , 35			
Tickets	: Adult : 2			
Payment Type	: Public eBank			

	Amount (RM)
Ticket Price :	22.90
Surcharge :	0.00
Booking Fees :	0.94
Total :	23.84
GST 6% :	1.44
Ent. Tax :	5.72
Total Payable :	31.00

Transaction Date : 05 Nov 2015 9:08AM

Scan QR Code at Autogate or Checkpoint Counter

7.0 SQL Statements (DDL & DML)

DDL

1) Create Table – Membership

```
CREATE TABLE Membership (
    MembershipID NUMBER(10) PRIMARY KEY,
    StartDate DATE,
    ExpiryDate DATE
);
```

2) Create Table – Email_Info

```
CREATE TABLE Email_Info (
    Email VARCHAR2(30) PRIMARY KEY,
    FirstName VARCHAR2(20),
    LastName VARCHAR2(20)
);
```

3) Create Table – Cust_Info

```
CREATE TABLE Cust_Info (
    CustomerID NUMBER(10) PRIMARY KEY,
    Email VARCHAR2(30) REFERENCES Email_Info(Email)
);
```

4) Create Table – MemberInfo

```
CREATE TABLE MemberInfo (
    MembershipID NUMBER(10) PRIMARY KEY
      REFERENCES Membership(MembershipID),
    Phone VARCHAR2(12)
);
```

5) Create Table – Cust_Member

```
CREATE TABLE Cust_Member (
    CustomerID NUMBER(10) REFERENCES Cust_Info(CustomerID),
    MembershipID NUMBER(10) REFERENCES Membership(MembershipID),
    Email VARCHAR2(30) REFERENCES Email_Info(Email),
    PRIMARY KEY (CustomerID, MembershipID)
);
```

6) Create Table – MovieRelease

```sql
CREATE TABLE MovieRelease (
    MovieID NUMBER(10) PRIMARY KEY,
    ReleaseDate DATE
);
```

7) Create Table – MovieDetails

```sql
CREATE TABLE MovieDetails (
    Title VARCHAR2(30),
    Genre VARCHAR2(15),
    Language VARCHAR2(15),
    Duration VARCHAR2(20),
    PRIMARY KEY (Title)
);
```

8) Create Table – Movie

```sql
CREATE TABLE Movie (
    MovieID NUMBER(10) PRIMARY KEY REFERENCES MovieRelease(MovieID),
    Title VARCHAR2(30)
);
```

9) Create Table – Showtime

```sql
CREATE TABLE Showtime (
    ShowTimeNo NUMBER(2) PRIMARY KEY,
    StartTime TIMESTAMP,
    EndTime TIMESTAMP
);
```

10) Create Table – MovieInfo

```sql
CREATE TABLE MovieInfo (
    MovieID NUMBER(10) PRIMARY KEY REFERENCES Movie(MovieID),
    Status VARCHAR2(50)
);
```

11) Create Table – BookInfo

```sql
CREATE TABLE BookInfo (
    BookingID NUMBER(10) PRIMARY KEY,
    BookingDate DATE,
    ShowTimeNo NUMBER(2) REFERENCES Showtime(ShowTimeNo)
);
```

12) Create Table – Booking

```sql
CREATE TABLE Booking (
    BookingID NUMBER(10) PRIMARY KEY REFERENCES BookInfo(BookingID),
    MovieID NUMBER(10) REFERENCES Movie(MovieID),
    ShowTimeNo NUMBER(2) REFERENCES Showtime(ShowTimeNo)
);
```

13) Create Table – Hall

```sql
CREATE TABLE Hall (
    HallNumber VARCHAR2(3) PRIMARY KEY,
    HallPrice DECIMAL(6,2),
    AvailableSeat NUMBER(10),
    MovieID NUMBER(10) REFERENCES Movie(MovieID)
);
```

14) Create Table – Cust_Paid

```sql
CREATE TABLE Cust_Paid (
    CustomerID NUMBER(10) PRIMARY KEY REFERENCES Cust_Info(CustomerID),
    FirstName VARCHAR2(20),
    LastName VARCHAR2(20),
    Email VARCHAR2(30) REFERENCES Email_Info(Email),
    Phone VARCHAR2(12)
);
```

15) Create Table – Receipt

```sql
CREATE TABLE Receipt (
    QRCodeTicket NUMBER(11) PRIMARY KEY,
    ReceiptID NUMBER(10),
    DateandTime TIMESTAMP,
    TotalAmount DECIMAL(6,2)
);
```

16) Create Table – Cust_Receipt

```sql
CREATE TABLE Cust_Receipt (
    CustomerID NUMBER(10) REFERENCES Cust_Info(CustomerID),
    QRCodeTicket NUMBER(11) REFERENCES Receipt(QRCodeTicket),
    PRIMARY KEY (CustomerID, QRCodeTicket)
);
```

17) Create Table – Premium

```sql
CREATE TABLE Premium (
    MembershipID NUMBER(10) PRIMARY KEY
      REFERENCES Membership(MembershipID),
    StartDate DATE,
    ExpiryDate DATE,
    Points NUMBER(5)
);
```

18) Create Table – Basic

```sql
CREATE TABLE Basic (
    MembershipID NUMBER(10) PRIMARY KEY
      REFERENCES Membership(MembershipID),
    StartDate DATE,
    ExpiryDate DATE
);
```

19) Create Table – Grand

```sql
CREATE TABLE Grand (
    HallNumber VARCHAR2(3) PRIMARY KEY,
    HallPrice DECIMAL(6,2),
    AvailableSeat NUMBER(10),
    BeanieSeat VARCHAR2(5)
);
```

20) Create Table – Standard

```sql
CREATE TABLE Standard (
    HallNumber VARCHAR2(3) PRIMARY KEY,
    HallPrice DECIMAL(6,2),
    AvailableSeat NUMBER(10),
    NormalSeat VARCHAR2(5),
    CoupleSeat VARCHAR2(5)
);
```

21) Create Table – Cust_Booking

```sql
CREATE TABLE Cust_Booking (
    CustomerID NUMBER(10) REFERENCES Cust_Info(CustomerID),
    BookingID NUMBER(10) REFERENCES BookInfo(BookingID),
    PRIMARY KEY (CustomerID, BookingID)
);
```

22) Create Table – Customer

```
CREATE TABLE Customer (
    CustomerID NUMBER   (10) PRIMARY KEY,
    FirstName VARCHAR2(20),
    LastName VARCHAR2(20),
    Email VARCHAR2(30) REFERENCES Email_Info(Email)
);
```

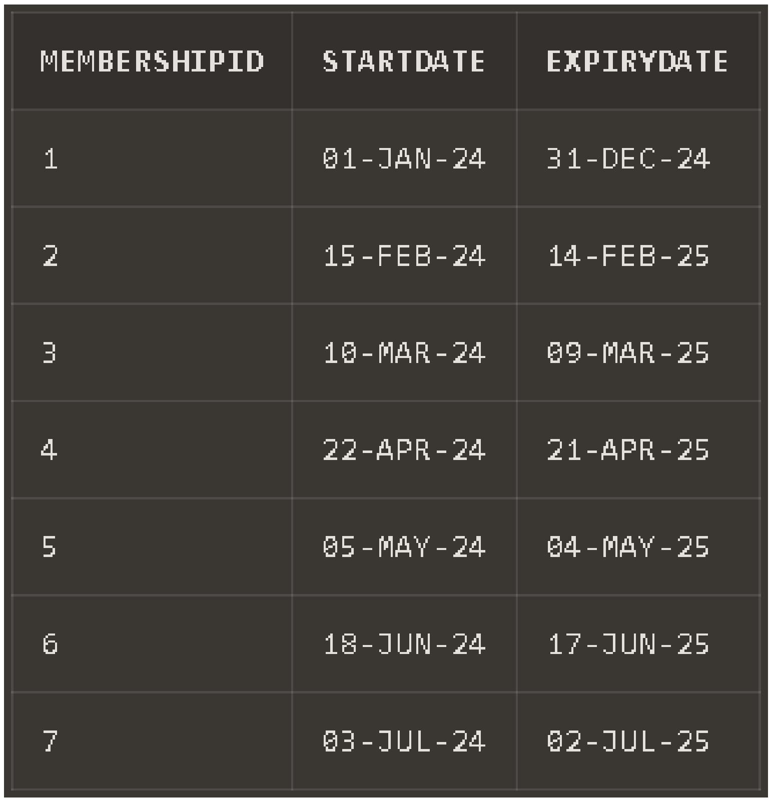
23) Create Table – Cust_Contact

```
CREATE TABLE Cust_contact (
    Phone VARCHAR2(12) PRIMARY KEY,
    CustomerID NUMBER REFERENCES Customer(CustomerID)
);
```

DML

1) Insert Data – Membership

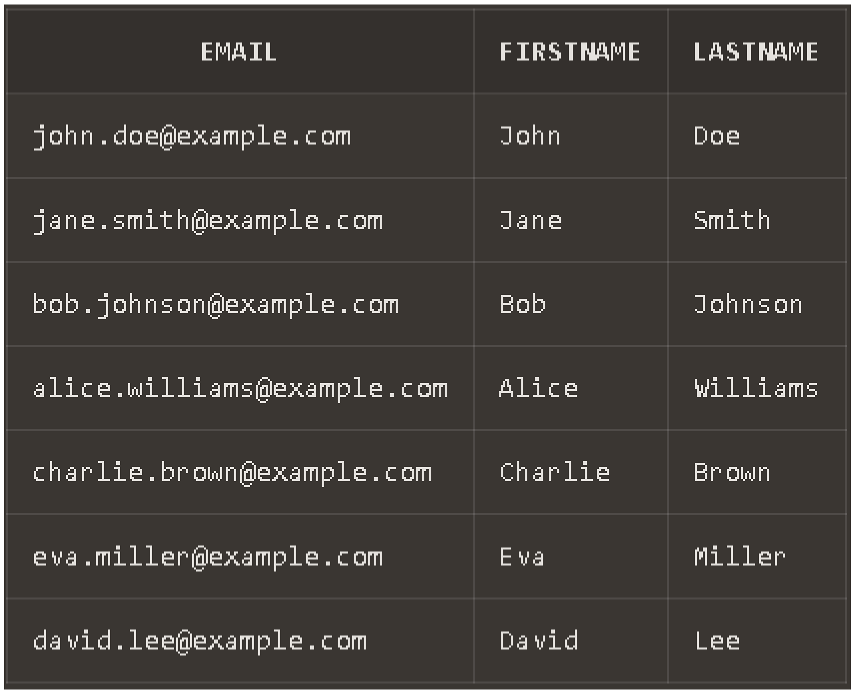
```
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00001', DATE '2024-01-01', DATE '2024-12-31');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00002', DATE '2024-02-15', DATE '2025-02-14');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00003', DATE '2024-03-10', DATE '2025-03-09');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00004', DATE '2024-04-22', DATE '2025-04-21');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00005', DATE '2024-05-05', DATE '2025-05-04');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00006', DATE '2024-06-18', DATE '2025-06-17');
INSERT INTO Membership (MembershipID, StartDate, ExpiryDate) VALUES
('00007', DATE '2024-07-03', DATE '2025-07-02');
```



MEMBERSHIPID	STARTDATE	EXPIRYDATE
1	01-JAN-24	31-DEC-24
2	15-FEB-24	14-FEB-25
3	10-MAR-24	09-MAR-25
4	22-APR-24	21-APR-25
5	05-MAY-24	04-MAY-25
6	18-JUN-24	17-JUN-25
7	03-JUL-24	02-JUL-25

2) Insert Data – Email_Info

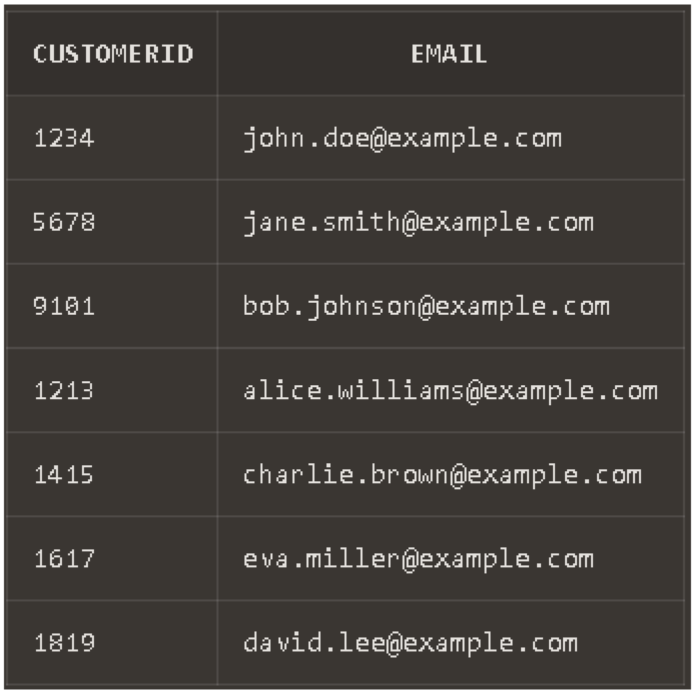
```
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('john.doe@example.com', 'John', 'Doe');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('jane.smith@example.com', 'Jane', 'Smith');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('bob.johnson@example.com', 'Bob', 'Johnson');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('alice.williams@example.com', 'Alice', 'Williams');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('charlie.brown@example.com', 'Charlie', 'Brown');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('eva.miller@example.com', 'Eva', 'Miller');
INSERT INTO Email_Info (Email, FirstName, LastName) VALUES
('david.lee@example.com', 'David', 'Lee');
```



EMAIL	FIRSTNAME	LASTNAME
john.doe@example.com	John	Doe
jane.smith@example.com	Jane	Smith
bob.johnson@example.com	Bob	Johnson
alice.williams@example.com	Alice	Williams
charlie.brown@example.com	Charlie	Brown
eva.miller@example.com	Eva	Miller
david.lee@example.com	David	Lee

3) Insert Data – Cust_Info

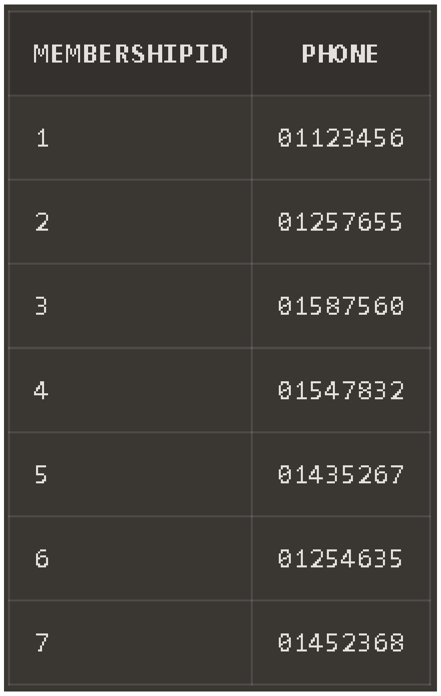
```sql
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(1234, 'john.doe@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(5678, 'jane.smith@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(9101, 'bob.johnson@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(1213, 'alice.williams@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(1415, 'charlie.brown@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(1617, 'eva.miller@example.com');
INSERT INTO Cust_Info (CustomerID, Email) VALUES
(1819, 'david.lee@example.com');
```



CUSTOMERID	EMAIL
1234	john.doe@example.com
5678	jane.smith@example.com
9101	bob.johnson@example.com
1213	alice.williams@example.com
1415	charlie.brown@example.com
1617	eva.miller@example.com
1819	david.lee@example.com

4) Insert Data – MemberInfo

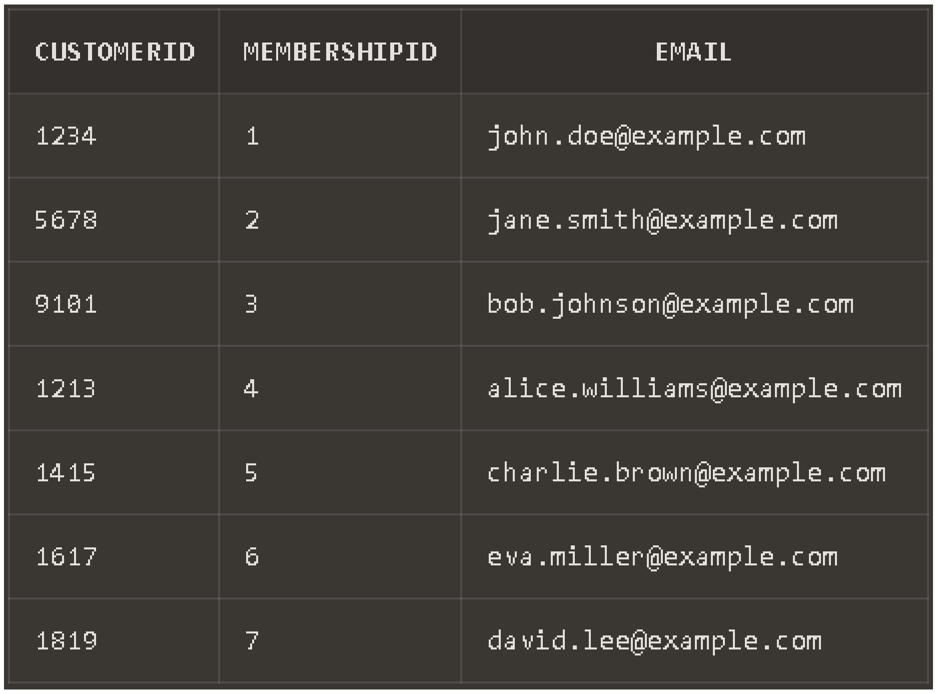
```
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00001, '01123456');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00002, '01257655');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00003, '01587560');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00004, '01547832');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00005, '01435267');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00006, '01254635');
INSERT INTO MemberInfo (MembershipID, Phone) VALUES
(00007, '01452368');
```



MEMBERSHIPID	PHONE
1	01123456
2	01257655
3	01587560
4	01547832
5	01435267
6	01254635
7	01452368

5) Insert Data – Cust_Member

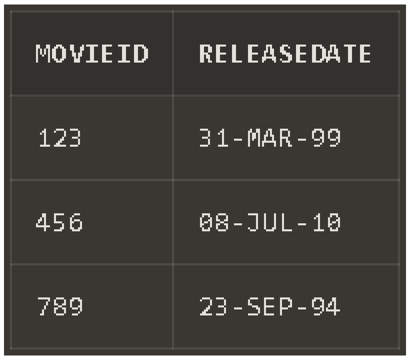
```sql
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(1234, 00001, 'john.doe@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(5678, 00002, 'jane.smith@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(9101, 00003, 'bob.johnson@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(1213, 00004, 'alice.williams@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(1415, 00005, 'charlie.brown@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(1617, 00006, 'eva.miller@example.com');
INSERT INTO Cust_Member (CustomerID, MembershipID, Email) VALUES
(1819, 00007, 'david.lee@example.com');
```



CUSTOMERID	MEMBERSHIPID	EMAIL
1234	1	john.doe@example.com
5678	2	jane.smith@example.com
9101	3	bob.johnson@example.com
1213	4	alice.williams@example.com
1415	5	charlie.brown@example.com
1617	6	eva.miller@example.com
1819	7	david.lee@example.com

6) Insert Data – MovieRelease

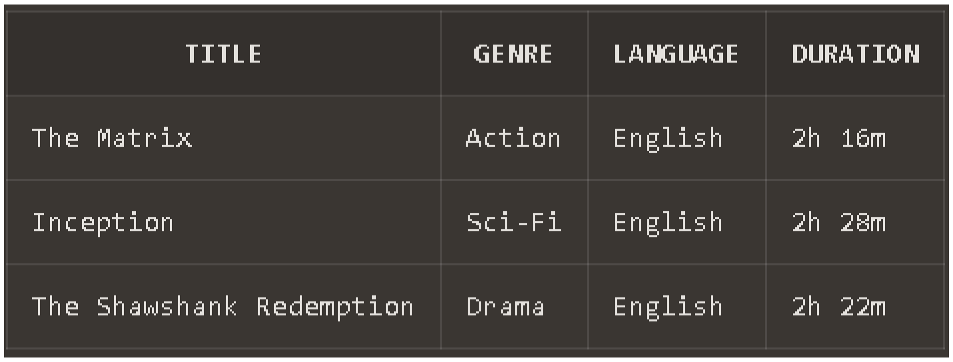
```
INSERT INTO MovieRelease (MovieID, ReleaseDate) VALUES
(123, DATE '1999-03-31');
INSERT INTO MovieRelease (MovieID, ReleaseDate) VALUES
(456, DATE '2010-07-08');
INSERT INTO MovieRelease (MovieID, ReleaseDate) VALUES
(789, DATE '1994-09-23');
```



MOVIEID	RELEASEDATE
123	31-MAR-99
456	08-JUL-10
789	23-SEP-94

7) Insert Data – MovieDetails

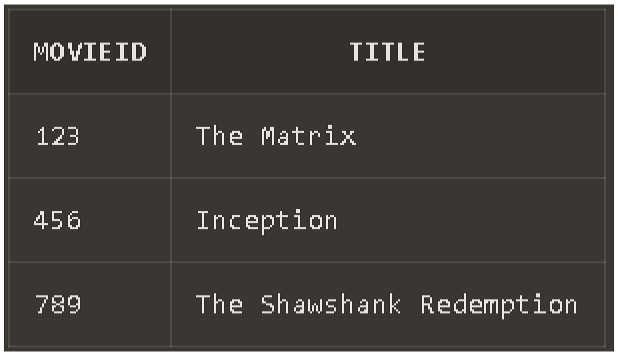
```
INSERT INTO MovieDetails (Title, Genre, Language, Duration) VALUES
('The Matrix', 'Action', 'English', '2h 16m');
INSERT INTO MovieDetails (Title, Genre, Language, Duration) VALUES
('Inception', 'Sci-Fi', 'English', '2h 28m');
INSERT INTO MovieDetails (Title, Genre, Language, Duration) VALUES
('The Shawshank Redemption', 'Drama', 'English', '2h 22m');
```



TITLE	GENRE	LANGUAGE	DURATION
The Matrix	Action	English	2h 16m
Inception	Sci-Fi	English	2h 28m
The Shawshank Redemption	Drama	English	2h 22m

8) Insert Data – Movie

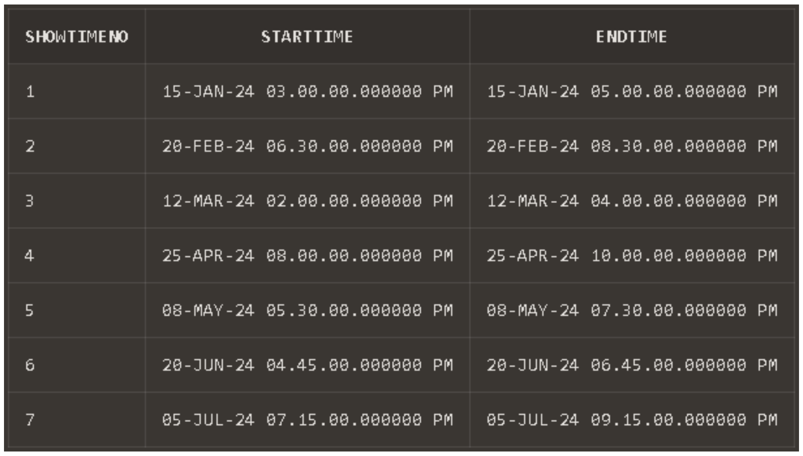
```
INSERT INTO Movie (MovieID, Title) VALUES
(123, 'The Matrix');
INSERT INTO Movie (MovieID, Title) VALUES
(456, 'Inception');
INSERT INTO Movie (MovieID, Title) VALUES
(789, 'The Shawshank Redemption');
```



MOVIEID	TITLE
123	The Matrix
456	Inception
789	The Shawshank Redemption

9) Insert Data – Showtime

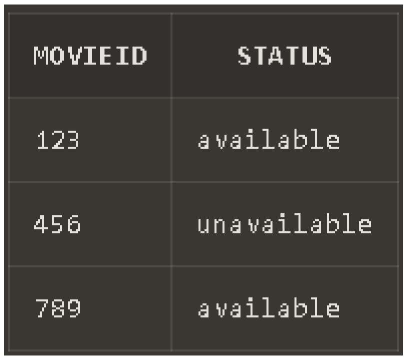
```sql
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(1, TIMESTAMP '2024-01-15 15:00:00', TIMESTAMP '2024-01-15 17:00:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(2, TIMESTAMP '2024-02-20 18:30:00', TIMESTAMP '2024-02-20 20:30:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(3, TIMESTAMP '2024-03-12 14:00:00', TIMESTAMP '2024-03-12 16:00:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(4, TIMESTAMP '2024-04-25 20:00:00', TIMESTAMP '2024-04-25 22:00:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(5, TIMESTAMP '2024-05-08 17:30:00', TIMESTAMP '2024-05-08 19:30:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(6, TIMESTAMP '2024-06-20 16:45:00', TIMESTAMP '2024-06-20 18:45:00');
INSERT INTO Showtime (ShowTimeNo, StartTime, EndTime) VALUES
(7, TIMESTAMP '2024-07-05 19:15:00', TIMESTAMP '2024-07-05 21:15:00');
```



SHOWTIMENO	STARTTIME	ENDTIME
1	15-JAN-24 03.00.00.000000 PM	15-JAN-24 05.00.00.000000 PM
2	20-FEB-24 06.30.00.000000 PM	20-FEB-24 08.30.00.000000 PM
3	12-MAR-24 02.00.00.000000 PM	12-MAR-24 04.00.00.000000 PM
4	25-APR-24 08.00.00.000000 PM	25-APR-24 10.00.00.000000 PM
5	08-MAY-24 05.30.00.000000 PM	08-MAY-24 07.30.00.000000 PM
6	20-JUN-24 04.45.00.000000 PM	20-JUN-24 06.45.00.000000 PM
7	05-JUL-24 07.15.00.000000 PM	05-JUL-24 09.15.00.000000 PM

10) Insert Data – MovieInfo

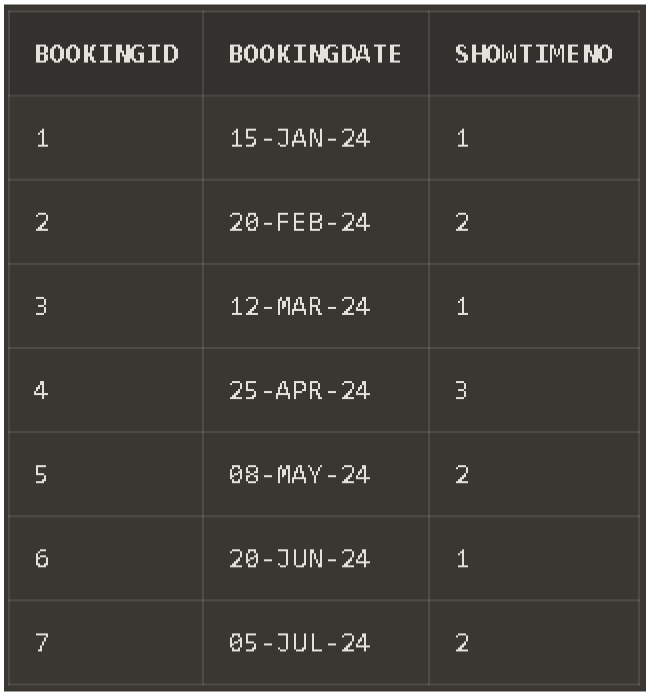
```
INSERT INTO MovieInfo (MovieID, Status) VALUES
(123, 'available');
INSERT INTO MovieInfo (MovieID, Status) VALUES
(456, 'unavailable');
INSERT INTO MovieInfo (MovieID, Status) VALUES
(789, 'available');
```



MOVIEID	STATUS
123	available
456	unavailable
789	available

11) Insert Data – BookInfo

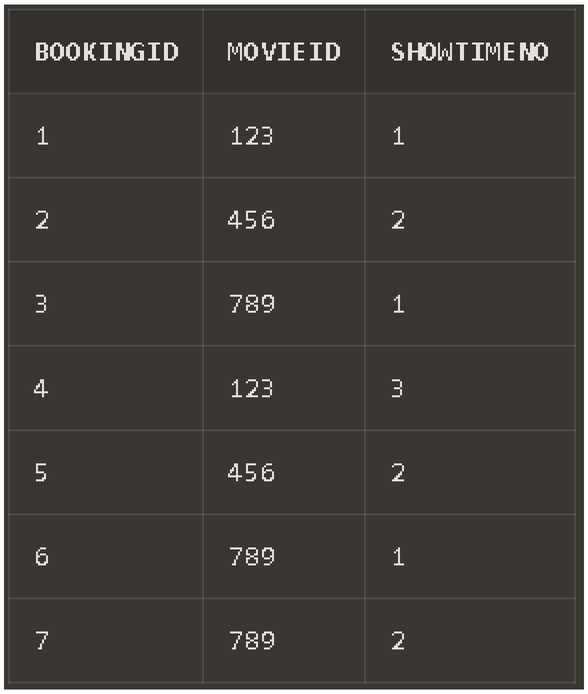
```sql
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(1, DATE '2024-01-15', 1);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(2, DATE '2024-02-20', 2);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(3, DATE '2024-03-12', 1);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(4, DATE '2024-04-25', 3);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(5, DATE '2024-05-08', 2);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(6, DATE '2024-06-20', 1);
INSERT INTO BookInfo (BookingID, BookingDate, ShowTimeNo) VALUES
(7, DATE '2024-07-05', 2);
```



BOOKINGID	BOOKINGDATE	SHOWTIMENO
1	15-JAN-24	1
2	20-FEB-24	2
3	12-MAR-24	1
4	25-APR-24	3
5	08-MAY-24	2
6	20-JUN-24	1
7	05-JUL-24	2

12) Insert Data – Booking

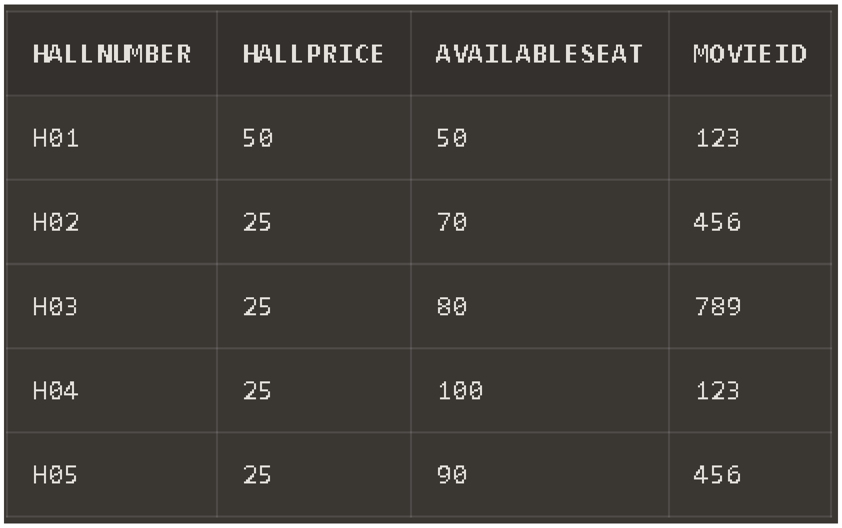
```
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(1, 123, 1);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(2, 456, 2);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(3, 789, 1);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(4, 123, 3);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(5, 456, 2);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(6, 789, 1);
INSERT INTO Booking (BookingID, MovieID, ShowTimeNo) VALUES
(7, 789, 2);
```



BOOKINGID	MOVIEID	SHOWTIMENO
1	123	1
2	456	2
3	789	1
4	123	3
5	456	2
6	789	1
7	789	2

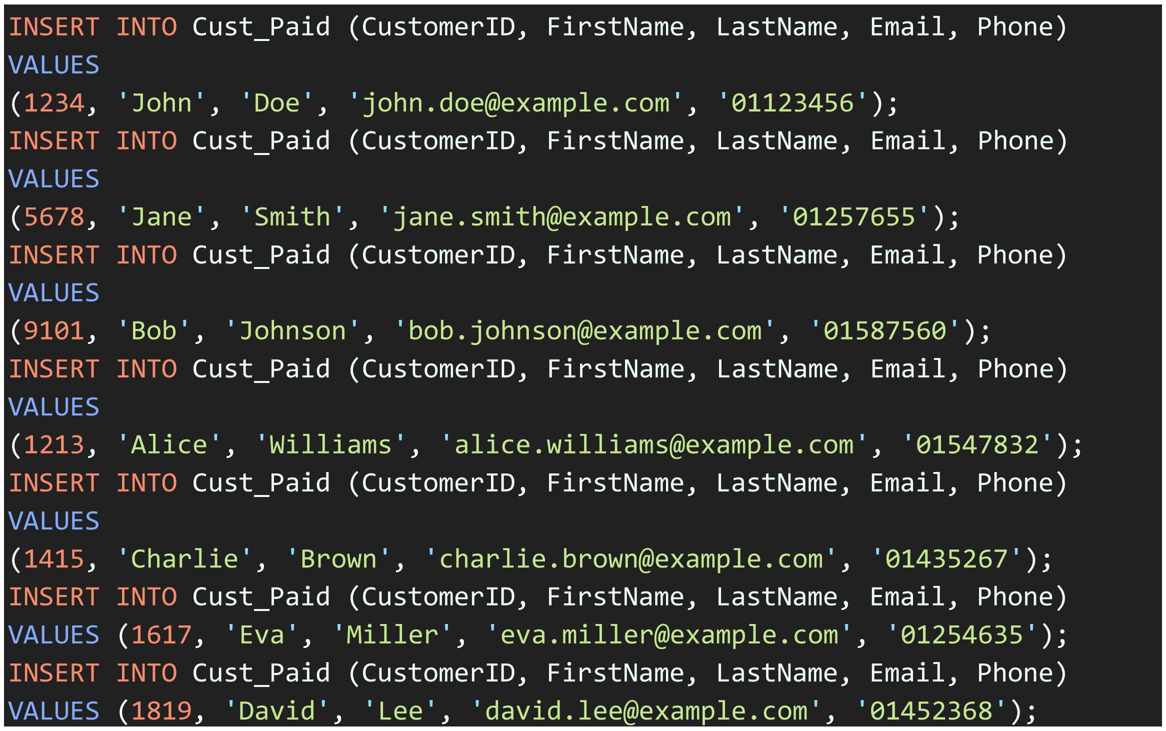
13) Insert Data – Hall

```
INSERT INTO Hall (HallNumber, HallPrice, AvailableSeat, MovieID) VALUES
('H01', 50.00, 50, 123);
INSERT INTO Hall (HallNumber, HallPrice, AvailableSeat, MovieID) VALUES
('H02', 25.00, 70, 456);
INSERT INTO Hall (HallNumber, HallPrice, AvailableSeat, MovieID) VALUES
('H03', 25.00, 80, 789);
INSERT INTO Hall (HallNumber, HallPrice, AvailableSeat, MovieID) VALUES
('H04', 25.00, 100, 123);
INSERT INTO Hall (HallNumber, HallPrice, AvailableSeat, MovieID) VALUES
('H05', 25.00, 90, 456);
```

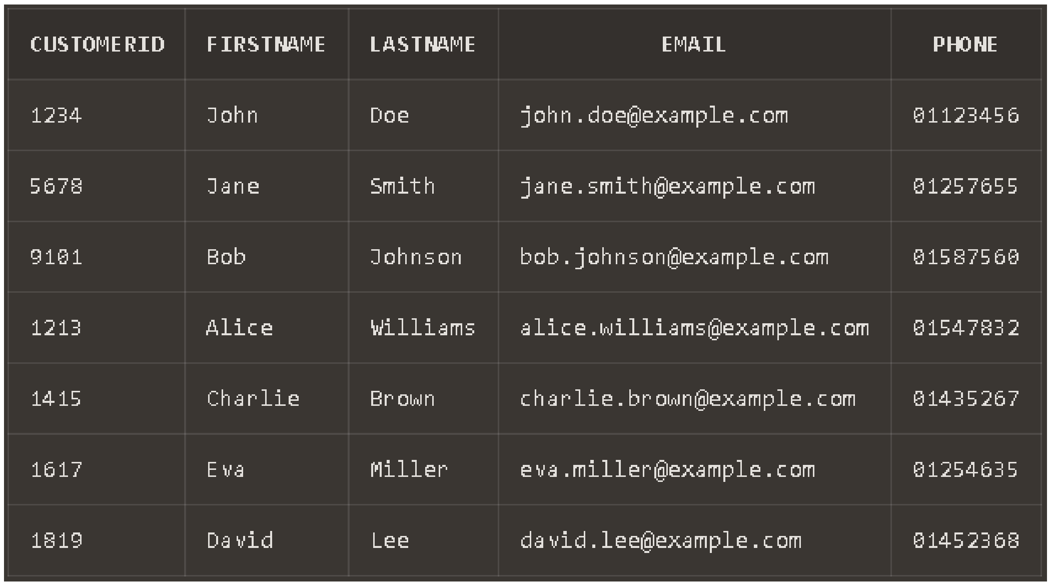


HALLNUMBER	HALLPRICE	AVAILABLESEAT	MOVIEID
H01	50	50	123
H02	25	70	456
H03	25	80	789
H04	25	100	123
H05	25	90	456

14) Insert Data – Cust_Paid



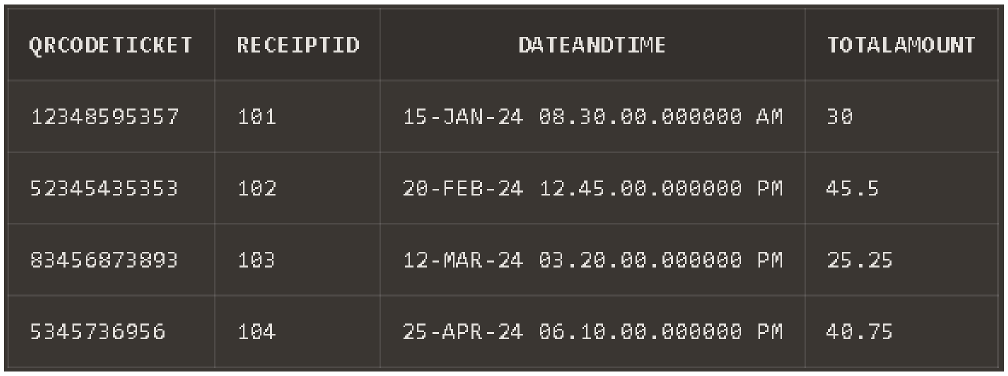
```sql
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES
(1234, 'John', 'Doe', 'john.doe@example.com', '01123456');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES
(5678, 'Jane', 'Smith', 'jane.smith@example.com', '01257655');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES
(9101, 'Bob', 'Johnson', 'bob.johnson@example.com', '01587560');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES
(1213, 'Alice', 'Williams', 'alice.williams@example.com', '01547832');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES
(1415, 'Charlie', 'Brown', 'charlie.brown@example.com', '01435267');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES (1617, 'Eva', 'Miller', 'eva.miller@example.com', '01254635');
INSERT INTO Cust_Paid (CustomerID, FirstName, LastName, Email, Phone)
VALUES (1819, 'David', 'Lee', 'david.lee@example.com', '01452368');
```



CUSTOMERID	FIRSTNAME	LASTNAME	EMAIL	PHONE
1234	John	Doe	john.doe@example.com	01123456
5678	Jane	Smith	jane.smith@example.com	01257655
9101	Bob	Johnson	bob.johnson@example.com	01587560
1213	Alice	Williams	alice.williams@example.com	01547832
1415	Charlie	Brown	charlie.brown@example.com	01435267
1617	Eva	Miller	eva.miller@example.com	01254635
1819	David	Lee	david.lee@example.com	01452368

15) Insert Data – Receipt

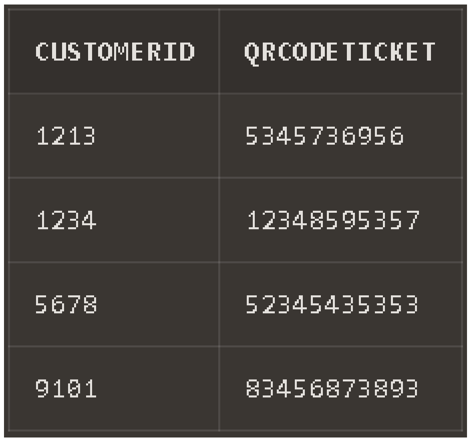
```sql
INSERT INTO Receipt (ReceiptID, DateAndTime, TotalAmount, QRCodeTicket)
VALUES (101, TO_TIMESTAMP('2024-01-15 08:30:00', 'YYYY-MM-DD HH24:MI:SS'),
30.00, '12348595357');
INSERT INTO Receipt (ReceiptID, DateAndTime, TotalAmount, QRCodeTicket)
VALUES (102, TO_TIMESTAMP('2024-02-20 12:45:00', 'YYYY-MM-DD HH24:MI:SS'),
45.50, '52345435353');
INSERT INTO Receipt (ReceiptID, DateAndTime, TotalAmount, QRCodeTicket)
VALUES (103, TO_TIMESTAMP('2024-03-12 15:20:00', 'YYYY-MM-DD HH24:MI:SS'),
25.25, '83456873893');
INSERT INTO Receipt (ReceiptID, DateAndTime, TotalAmount, QRCodeTicket)
VALUES (104, TO_TIMESTAMP('2024-04-25 18:10:00', 'YYYY-MM-DD HH24:MI:SS'),
40.75, '05345736956');
```



QRCODETICKET	RECEIPTID	DATEANDTIME	TOTALAMOUNT
12348595357	101	15-JAN-24 08.30.00.000000 AM	30
52345435353	102	20-FEB-24 12.45.00.000000 PM	45.5
83456873893	103	12-MAR-24 03.20.00.000000 PM	25.25
5345736956	104	25-APR-24 06.10.00.000000 PM	40.75

16) Insert Data – Cust_Receipt

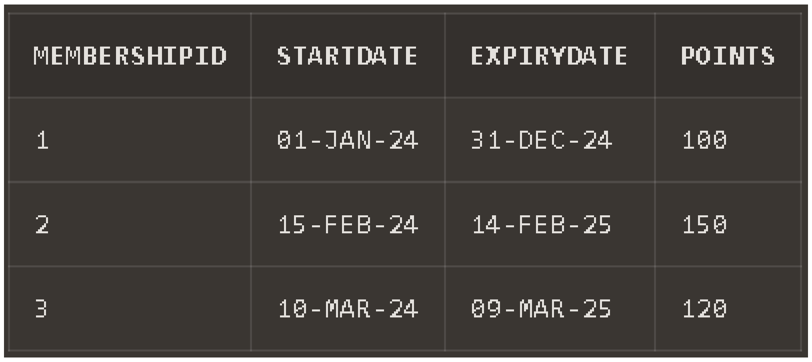
```
INSERT INTO Cust_Receipt (CustomerID, QRCodeTicket)
VALUES (1234, '12348595357');
INSERT INTO Cust_Receipt (CustomerID, QRCodeTicket)
VALUES (5678, '52345435353');
INSERT INTO Cust_Receipt (CustomerID, QRCodeTicket)
VALUES (9101, '83456873893');
INSERT INTO Cust_Receipt (CustomerID, QRCodeTicket)
VALUES (1213, '05345736956');
```



CUSTOMERID	QRCODETICKET
1213	5345736956
1234	12348595357
5678	52345435353
9101	83456873893

17) Insert Data – Premium

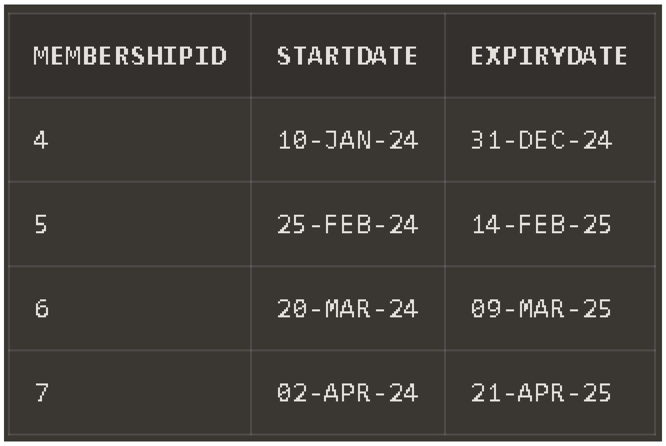
```
INSERT INTO Premium (MembershipID, StartDate, ExpiryDate, Points) VALUES
(00001, TO_DATE('2024-01-01', 'YYYY-MM-DD'), TO_DATE('2024-12-31', 'YYYY-
MM-DD'), 100);
INSERT INTO Premium (MembershipID, StartDate, ExpiryDate, Points) VALUES
(00002, TO_DATE('2024-02-15', 'YYYY-MM-DD'), TO_DATE('2025-02-14', 'YYYY-
MM-DD'), 150);
INSERT INTO Premium (MembershipID, StartDate, ExpiryDate, Points) VALUES
(00003, TO_DATE('2024-03-10', 'YYYY-MM-DD'), TO_DATE('2025-03-09', 'YYYY-
MM-DD'), 120);
```



MEMBERSHIPID	STARTDATE	EXPIRYDATE	POINTS
1	01-JAN-24	31-DEC-24	100
2	15-FEB-24	14-FEB-25	150
3	10-MAR-24	09-MAR-25	120

18) Insert Data – Basic

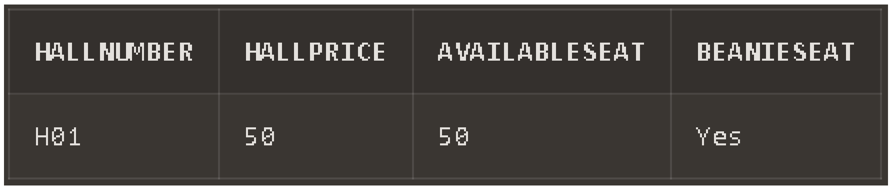
```
INSERT INTO Basic (MembershipID, StartDate, ExpiryDate) VALUES
(00004, TO_DATE('2024-01-10', 'YYYY-MM-DD'), TO_DATE('2024-12-31', 'YYYY-
MM-DD'));
INSERT INTO Basic (MembershipID, StartDate, ExpiryDate) VALUES
(00005, TO_DATE('2024-02-25', 'YYYY-MM-DD'), TO_DATE('2025-02-14', 'YYYY
MM-DD'));
INSERT INTO Basic (MembershipID, StartDate, ExpiryDate) VALUES
(00006, TO_DATE('2024-03-20', 'YYYY-MM-DD'), TO_DATE('2025-03-09', 'YYYY-
MM-DD'));
INSERT INTO Basic (MembershipID, StartDate, ExpiryDate) VALUES
(00007, TO_DATE('2024-04-02', 'YYYY-MM-DD'), TO_DATE('2025-04-21', 'YYYY-
MM-DD'));
```



MEMBERSHIPID	STARTDATE	EXPIRYDATE
4	10-JAN-24	31-DEC-24
5	25-FEB-24	14-FEB-25
6	20-MAR-24	09-MAR-25
7	02-APR-24	21-APR-25

19) Insert Data – Grand

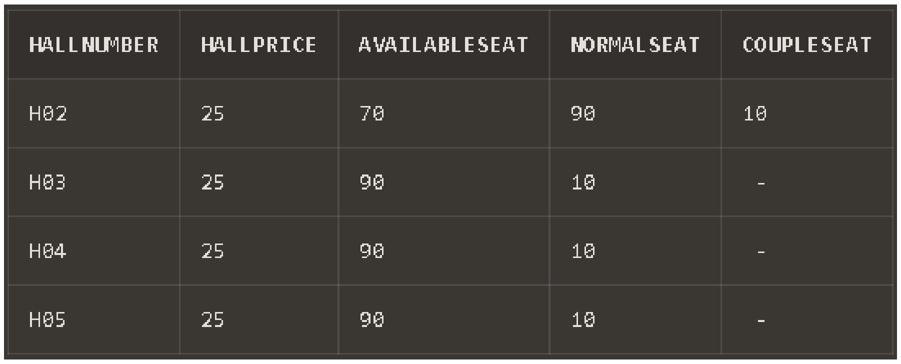
```
INSERT INTO Grand (HallNumber, HallPrice, AvailableSeat, BeanieSeat)
VALUES ('H01', 50.00, 50, 'Yes');
```



HALLNUMBER	HALLPRICE	AVAILABLESEAT	BEANIESEAT
H01	50	50	Yes

20) Insert Data – Standard

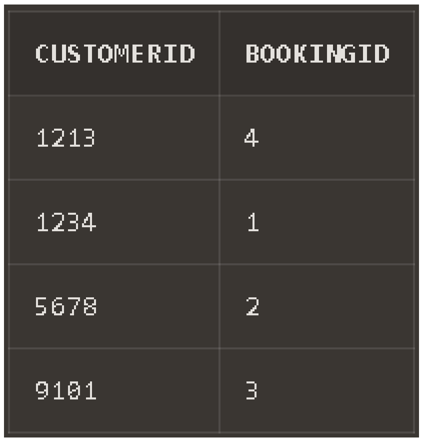
```
INSERT INTO Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, 
CoupleSeat) VALUES ('H02', 25.00, 70, 90, 10);
INSERT INTO Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, 
CoupleSeat) VALUES ('H03', 25.00, 90, 10, NULL);
INSERT INTO Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, 
CoupleSeat) VALUES ('H04', 25.00, 90, 10, NULL);
INSERT INTO Standard (HallNumber, HallPrice, AvailableSeat, NormalSeat, 
CoupleSeat) VALUES ('H05', 25.00, 90, 10, NULL);
```



HALLNUMBER	HALLPRICE	AVAILABLESEAT	NORMALSEAT	COUPLESEAT
H02	25	70	90	10
H03	25	90	10	-
H04	25	90	10	-
H05	25	90	10	-

21) Insert Data – Cust_Booking

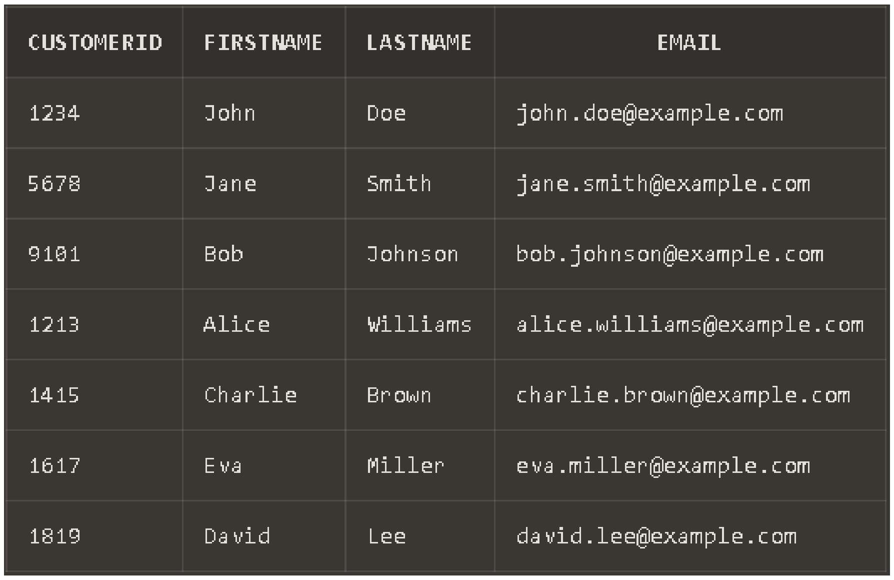
```
INSERT INTO Cust_Booking (CustomerID, BookingID) VALUES (1234, 1);
INSERT INTO Cust_Booking (CustomerID, BookingID) VALUES (5678, 2);
INSERT INTO Cust_Booking (CustomerID, BookingID) VALUES (9101, 3);
INSERT INTO Cust_Booking (CustomerID, BookingID) VALUES (1213, 4);
```



CUSTOMERID	BOOKINGID
1213	4
1234	1
5678	2
9101	3

22) Insert Data – Customer

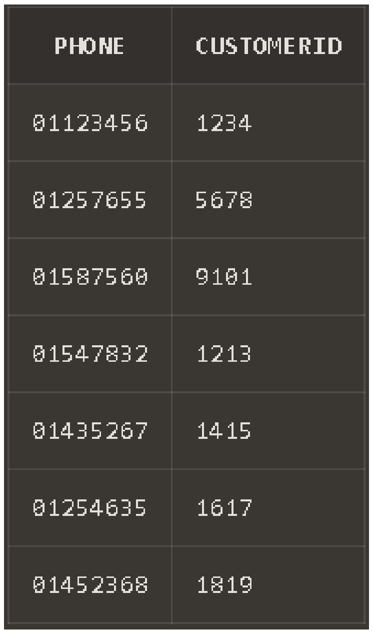
```sql
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (1234, 'John', 'Doe', 'john.doe@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (5678, 'Jane', 'Smith', 'jane.smith@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (9101, 'Bob', 'Johnson', 'bob.johnson@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (1213, 'Alice', 'Williams', 'alice.williams@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (1415, 'Charlie', 'Brown', 'charlie.brown@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (1617, 'Eva', 'Miller', 'eva.miller@example.com');
INSERT INTO Customer (CustomerID, FirstName, LastName, Email)
VALUES (1819, 'David', 'Lee', 'david.lee@example.com');
```



CUSTOMERID	FIRSTNAME	LASTNAME	EMAIL
1234	John	Doe	john.doe@example.com
5678	Jane	Smith	jane.smith@example.com
9101	Bob	Johnson	bob.johnson@example.com
1213	Alice	Williams	alice.williams@example.com
1415	Charlie	Brown	charlie.brown@example.com
1617	Eva	Miller	eva.miller@example.com
1819	David	Lee	david.lee@example.com

23) Insert Data – Cust_Contact

```
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (1234, '01123456');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (5678, '01257655');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (9101, '01587560');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (1213, '01547832');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (1415, '01435267');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (1617, '01254635');
INSERT INTO Cust_Contact (CustomerID, Phone) VALUES (1819, '01452368');
```



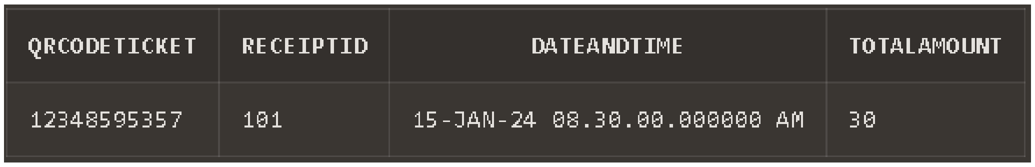
PHONE	CUSTOMERID
01123456	1234
01257655	5678
01587560	9101
01547832	1213
01435267	1415
01254635	1617
01452368	1819

8.0 Queries

First Query

- Display receipt where the receipt id is 101.

```
SELECT *
FROM Receipt
WHERE ReceiptID = 101;
```

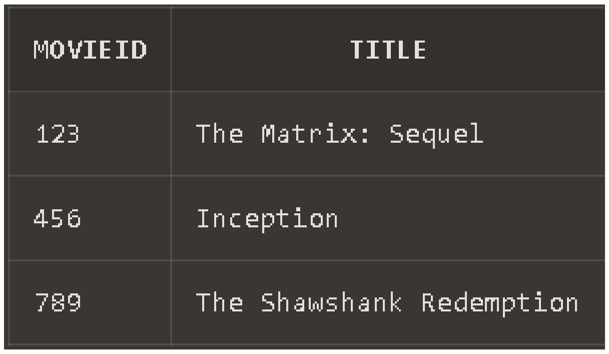


QRCODETICKET	RECEIPTID	DATEANDTIME	TOTALAMOUNT
12348595357	101	15-JAN-24 08.30.00.000000 AM	30

Second Query

- Update movie title of a movie with the title of 'The Matrix' to 'The Sequel' and display movieid and title.

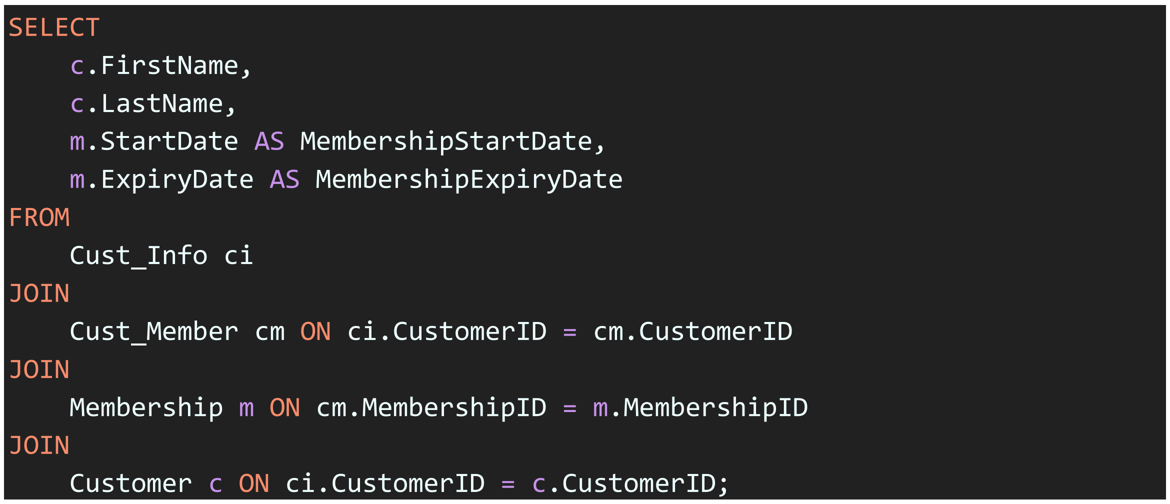
```
UPDATE Movie
SET Title = 'The Matrix: Sequel'
WHERE Title = 'The Matrix';

SELECT * FROM Movie;
```



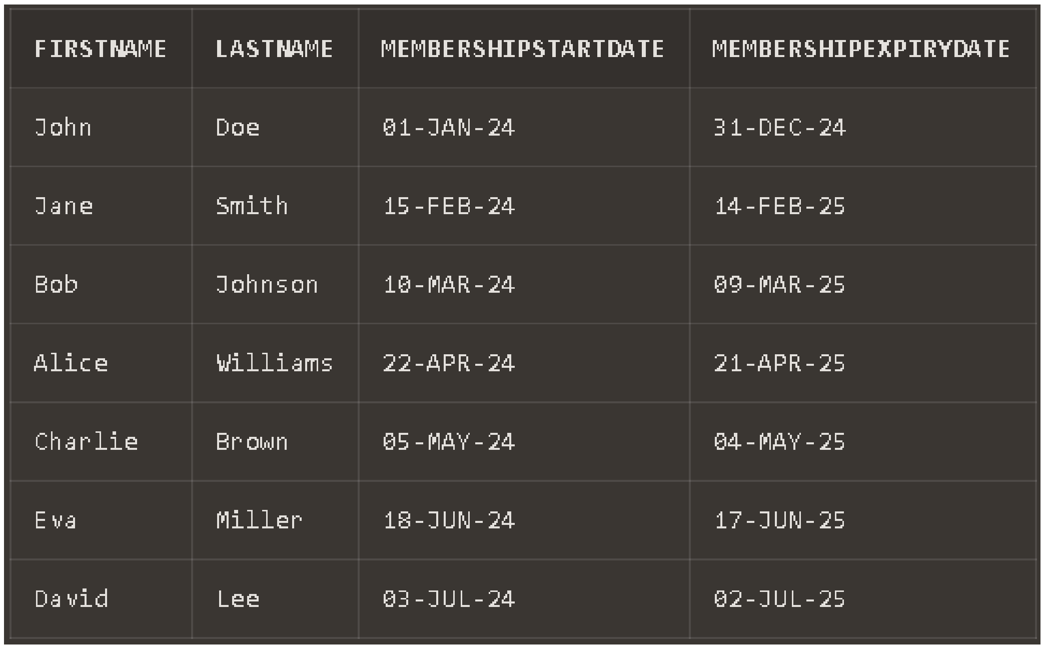
MOVIEID	TITLE
123	The Matrix: Sequel
456	Inception
789	The Shawshank Redemption

Third Query

- Display customer first name and last name, and their membership start date and expiry date.



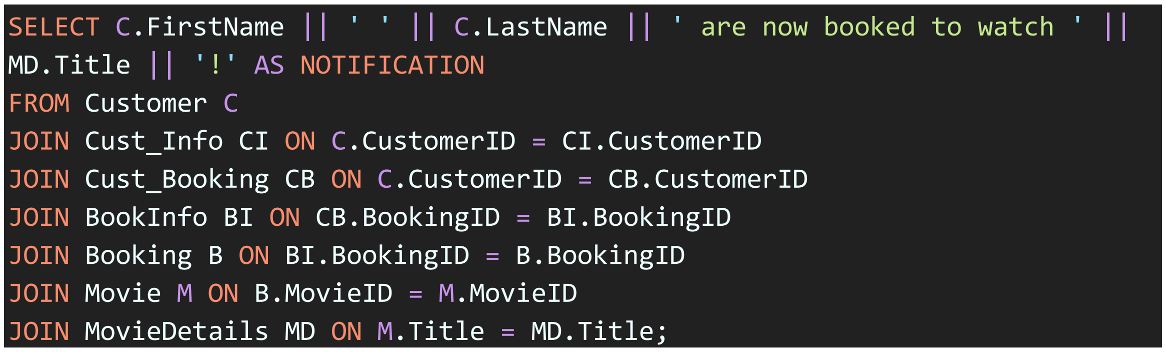
```sql
SELECT
    c.FirstName,
    c.LastName,
    m.StartDate AS MembershipStartDate,
    m.ExpiryDate AS MembershipExpiryDate
FROM
    Cust_Info ci
JOIN
    Cust_Member cm ON ci.CustomerID = cm.CustomerID
JOIN
    Membership m ON cm.MembershipID = m.MembershipID
JOIN
    Customer c ON ci.CustomerID = c.CustomerID;
```



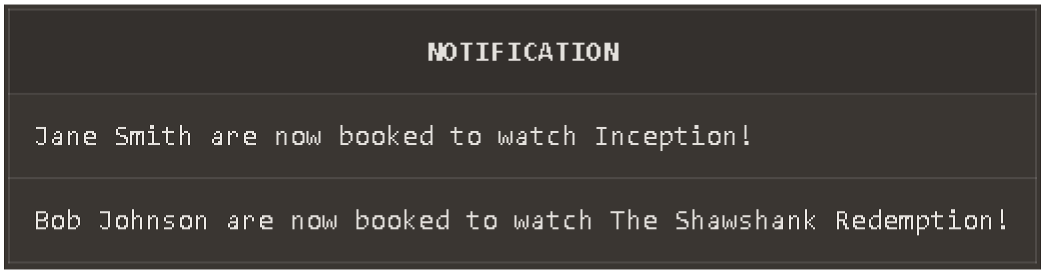
FIRSTNAME	LASTNAME	MEMBERSHIPSTARTDATE	MEMBERSHIPEXPIRYDATE
John	Doe	01-JAN-24	31-DEC-24
Jane	Smith	15-FEB-24	14-FEB-25
Bob	Johnson	10-MAR-24	09-MAR-25
Alice	Williams	22-APR-24	21-APR-25
Charlie	Brown	05-MAY-24	04-MAY-25
Eva	Miller	18-JUN-24	17-JUN-25
David	Lee	03-JUL-24	02-JUL-25

Fourth Query

- Display customer first name and last name, and the title of the movie they are watching in a sentence form of "FirstName LastName you are now booked to watch MovieTitle!".



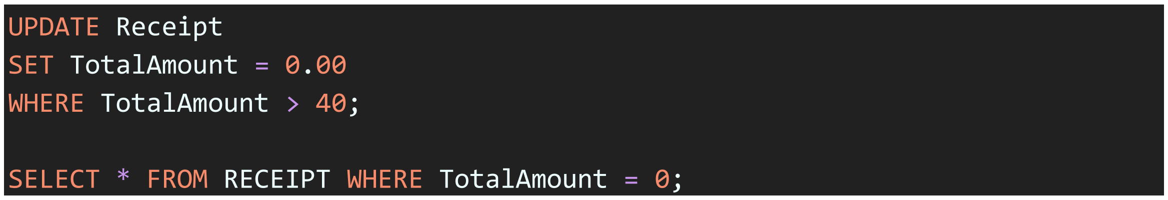
```sql
SELECT C.FirstName || ' ' || C.LastName || ' are now booked to watch ' ||
MD.Title || '!' AS NOTIFICATION
FROM Customer C
JOIN Cust_Info CI ON C.CustomerID = CI.CustomerID
JOIN Cust_Booking CB ON C.CustomerID = CB.CustomerID
JOIN BookInfo BI ON CB.BookingID = BI.BookingID
JOIN Booking B ON BI.BookingID = B.BookingID
JOIN Movie M ON B.MovieID = M.MovieID
JOIN MovieDetails MD ON M.Title = MD.Title;
```



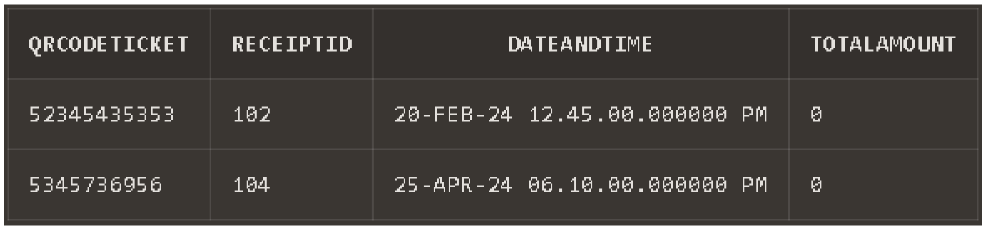
NOTIFICATION
Jane Smith are now booked to watch Inception!
Bob Johnson are now booked to watch The Shawshank Redemption!

Fifth Query

- Update total price to 0.00 where total amount is more than 40. Display all info from receipt.



```sql
UPDATE Receipt
SET TotalAmount = 0.00
WHERE TotalAmount > 40;

SELECT * FROM RECEIPT WHERE TotalAmount = 0;
```



QRCODETICKET	RECEIPTID	DATEANDTIME	TOTALAMOUNT
52345435353	102	20-FEB-24 12.45.00.000000 PM	0
5345736956	104	25-APR-24 06.10.00.000000 PM	0

9.0 Summary

In the closing stage, Phase 3 focuses on transforming a conceptual Entity-Relationship Diagram (ERD) into logical ERD which results in forming relational schema and normalization processes. The end results sought after are the implementation of completed final Logical ERD, an updated data dictionary and a set up validated system that meets. The main objective of this project is to focus on the need for a modern cinema scheduling software, with emphasis placed on easy navigation and enhanced customer experience.

This crucial project aims to accommodate the needs of an advanced cinema reservation system, appreciating that continuous process improvement coupled with heightened user satisfaction is essential. This driving force is the dedication of a collaborative team to create an innovative, efficient solution. The plan is aligned with the dynamism of audience expectations in this fast changing technological and user's preference landscape. Through innovation and user-centric design, the project intends to deliver a solution that not only meets but goes beyond what is needed regarding today's standards and requirements of the audience.

By carefully studying all the details of cinemas booking domain and aiming to build a system that meets present needs but also predicts and responds constantly, in dependence on future trends. It is predicted that the result will be a strong and stable cinema booking solution, which not only modernizes current infrastructure but also creates ground for further innovation to react to various technological adjustments as well as users' new demands.