UNIVERSITY PARTNER

UNIVERSITY OF WOLVERHAMPTON

HERALD COLLEGE KATHMANDU

Database (4NCDB011)

C1: Project Report [Online Movie Booking System]

Student Id	: [2227486]
Student Name	: [Nayan Raj Khanal]
Group	: (L4CG4)
Module Leader	: [Deepson Shrestha]
Asst. Module Leader	: [Anmol Adhikari]
Tutor	: [Ishwor Luitel]
Cohort	: 6

Submitted on : <15-08-2022>

Contents:

1. INTRODUCTION: 1
1.1. The Basement: 1
1.2. Activities and Operations: 1
1.3. Business Rules: 1
1.4. Entities and Attributes: 3
1.4.1. Entity: 3
1.4.2. Attribute: 3
1.4.3. Tabular Demonstration: 3
1.5. ER Diagram: 6
1.5.1. ERD Online Movie Booking System: 6
1.6. Use Case Diagram: 7
2. NORMALIZATION: 8
2.1. Assumptions: 8
2.2. Normalization: 8
2.2.1. Un-Normalized Form (UNF): 9
2.2.2. First Normal Form (1NF): 10
2.2.3. Second Normal Form (2NF): 12
2.2.4. 3NF: 14
2.3. Post-Normalization ERD: 16
2.4. Data Dictionary: 17
3. IMPLEMENTATION: 22
3.1. Creation Of Tables: 22
3.2. Populating Data: 31
3.2.1. Commit: 31
4. QUERIES: 53
4.1. Execution: 53
4.1.1. List Fname using LIKE: 53
4.1.2. List all movie shows with rating of five: 54
SELECT Show_Title FROM movieshow WHERE Show_Stars='five'; 54
4.1.3. Listing all Cinema Halls by order of their descending Hall ID: 55
4.1.4. Limiting the values displayed of the Customer Table: 56
4.1.5. Counting records of Screen Type: 56
4.1.6. Update the data from the Administrator table: 57
4.1.7. Get all Customers last name in lowercase: 58
4.1.8. Get the Screen Quality after replacing ‘K’ with ‘HD’: 59

4.1.9. Get the admin’s last name and first name in a single column separated by a ‘_’. 60
4.1.10. Selecting rows using Distinct: 61
4.2. Dump File: 62
4.3. Drop Tables: 63
5. CONCLUSION: 64
5.1. Learning Experience: 64
5.2. Critical Evaluation: 65
5.3. Future Enhancements: 65

Table Of Figures:

Figure 1. The Basement 2
Figure 2. Admin 3
Figure 3. Payment 4
Figure 4. Booking Website 4
Figure 5. Movie Show 4
Figure 6. Customer 5
Figure 7. Schedule 5
Figure 8. Cinema Hall 5
Figure 9. Screen 5
Figure 10. ERD 6
Figure 11. Use Case Diagram 7
Figure 12. Post Normalization ERD 16
Figure 13. Admin Command 23
Figure 14. Admin Describe 23
Figure 15. Payment Command 24
Figure 16. Payment Describe 24
Figure 17. BookingWebsite Command 25
Figure 18. BookingWebsite Describe 25
Figure 19. MovieShow Command 26
Figure 20. MovieShow Describe 26
Figure 21. Customer Command 27
Figure 22. Customer Describe 27
Figure 23. Schedule Command 28
Figure 24. Schedule Describe 28
Figure 25. CinemaHall Command 29
Figure 26. CinemaHall Describe 29
Figure 27. Screen Command 30
Figure 28. Screen Describe 30
Figure 29. Inserting into admin table 34
Figure 30. Displaying Admin table 34
Figure 31. Inserting into payment table 37
Figure 32. Displaying Payment table 37
Figure 33. Inserting into bookingwebsite table 38
Figure 34. Displaying bookingwebsite table 38
Figure 35. Inserting into MovieShow table 41
Figure 36. Displaying Movie List Table 41
Figure 37. Inserting into Customer table 44
Figure 38. Displaying Customer Table 44
Figure 39. Inserting into Schedule table 46
Figure 40. Displaying Schedule Table 46
Figure 41. Inserting into Cinema Hall table 49
Figure 42. Displaying Cinema Hall Table 49
Figure 43. Inserting into screen table 52
Figure 44. Displaying Screen Table 52
Figure 45. Like Query 53
Figure 46. Like Output 53
Figure 47. Show_Title Query 54

Figure 48. Show_Title Output 54
Figure 49. Order Query...55
Figure 50. Order Output...55
Figure 51. Limit Query 56
Figure 52. Limit Output 56
Figure 53. Count Query 56
Figure 54. Count Output 56
Figure 55. Update Query..57
Figure 56. Update Output...57
Figure 57. Lower Query...58
Figure 58. Lower Output..58
Figure 59. Replace Query 59
Figure 60. Replace Output 59
Figure 61. Concat Query..60
Figure 62. Concat Output...60
Figure 63. Distinct Query...61
Figure 64. Distinct Output 61
Figure 65. Exporting 62

1. INTRODUCTION:

1.1. The Basement:

Established in 2001 by ‘Nayan Raj Khanal’, ‘The Basement’ has been one of the most renowned and top-rated movie hall company in the continent. It provides customer with top class service from quality and hygienic fast foods, friendly and helpful staffs to comfortable and clean seating hall etc. ‘The Basement” also provides customers with extra recreational activities before and after the shows such as shooting hoops, playing darts, bowling etc.

The founder of this movie hall company has a far-sighted vision which is provide the top-level entertainment inside and outside the movie room with affordable price so that all customers can have fun both internationally and nationally.

1.2. Activities and Operations:

The movie hall is accessible to the general public via an online movie booking website. The activities and operations performed by all the entities are as follows:
Admin MANAGES payment and booking website. Booking website RUNS movie show. Customer BOOKS show from the booking website. The show HAS a scheduled time. The movie show RUNS IN Cinema Hall. The cinema hall HAS numerous seats.

1.3. Business Rules:

The Administrator has the full control of the booking website. Admin adds/removes/updates movie list on a regular basis. The website is dynamic and user-friendly. It is easy to use for the customers who wishes to book a movie as per their convenience as the website and the hall are up and running 24/7. Once the ticket has been finalized, the customers can enjoy their movie in a A-tier movie hall which has numerous comfortable seats and various recreation activities.

There are a total of eight databases:

I. Administrator: The data it contains are ID, Name and Contact of Admin.

II. Payment: The data it contains are ID, Type, Date and Amount

III. Booking Website: The data it contains are URL, Name and Contact

IV. Movie Show: The data it contains are ID, Title, Stars and Description

V. Customer: The data it contains are ID, Name, Contact and Address

VI. Schedule: The data it contains are Time and Date

VII. Cinema Hall: The data it contains are ID, Name, Rating, Location and number of seats

VIII. Screen: The data it contains are Number, Quality and Size

The Basement' takes the safety of information of the customers very seriously. It only takes required information and proceeds the booking ensuring no leaks. The information from all the databases helps the company to improve its members, viewer experience and reach.

Figure 1. The Basement

Link: https://kathmandupost.com/money/2022/04/20/durbar-cinemax-opens-in-durbar-mall

1.4. Entities and Attributes:

1.4.1. Entity:

Entity is any type of object or event, real or abstract that we wish to store data. Entities are distinct, which means that each entity in a pair of entities possesses a characteristic that distinguishes it from the other entity.

1.4.2. Attribute:

An attribute is a property. It provides information about the instances in the database column. It has the following types:

I. Simple Attribute: It has one component

II. Composite Attribute: It has multiple components

III. Multivalued Attribute: It has more than one value

IV. Key Attribute: It uniquely identifies the item

V. Derived Attribute: It is derived from other attributes

1.4.3. Tabular Demonstration:

I. Administrator:

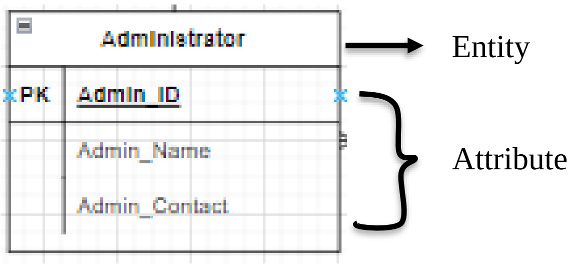
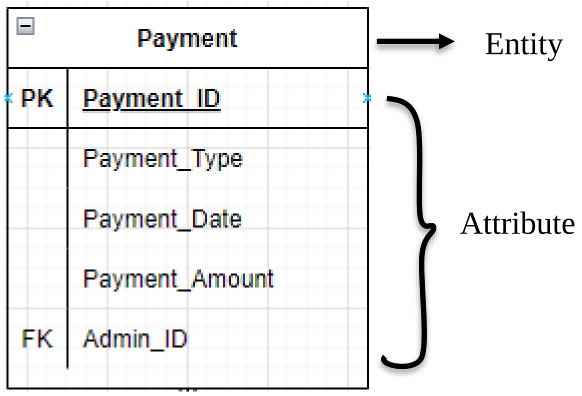


Figure 2. Admin

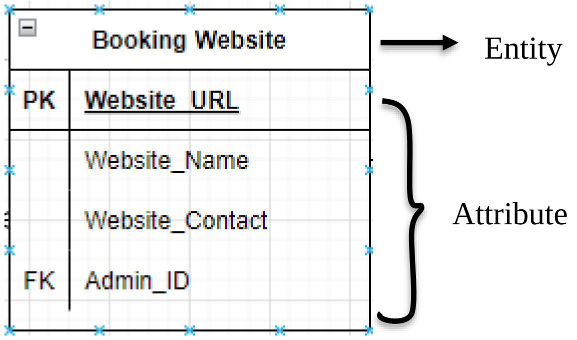
II. Payment:



	Payment
PK	Payment_ID
	Payment_Type
	Payment_Date
	Payment_Amount
FK	Admin_ID

Figure 3. Payment

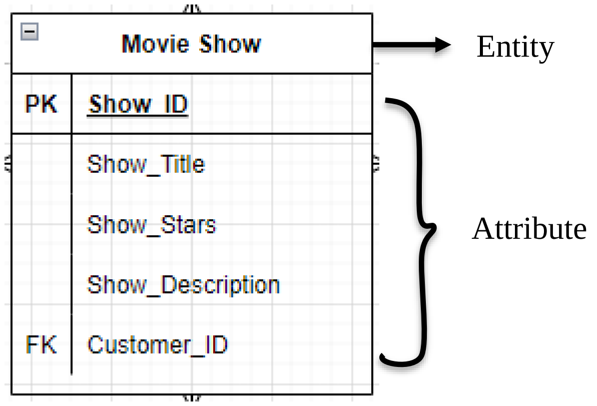
III. Booking Website:



	Booking Website
PK	Website_URL
	Website_Name
	Website_Contact
FK	Admin_ID

Figure 4. Booking Website

IV. Movie Show:



	Movie Show
PK	Show_ID
	Show_Title
	Show_Stars
	Show_Description
FK	Customer_ID

Figure 5. Movie Show

V. Customer:

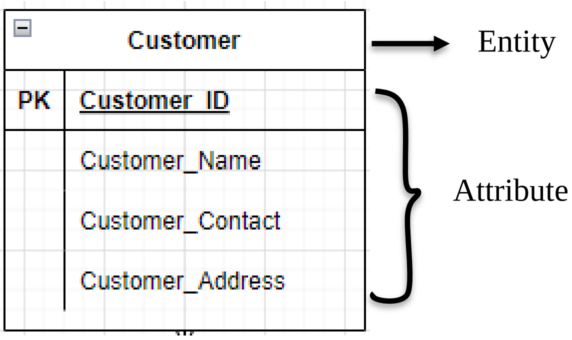


Figure 6. Customer

VI. Schedule:

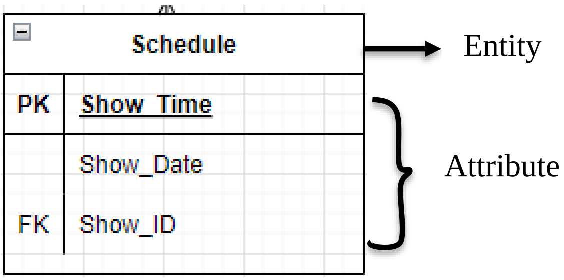


Figure 7. Schedule

VII. Cinema Hall:

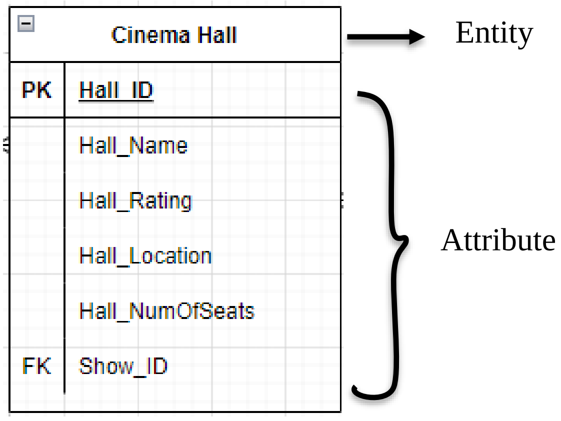


Figure 8. Cinema Hall

VIII. Screen:

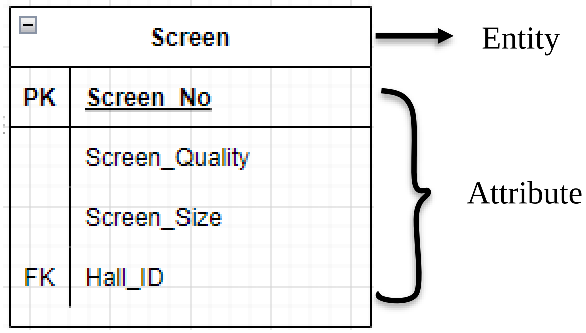


Figure 9. Screen

1.5. ER Diagram:

A particular sort of flowchart that shows how "entities" like people, things, or concepts link to one another inside a system is called an entity relationship (ER) diagram. Software engineering, business information systems, education, and research are the industries that use ER Diagrams the most to develop or debug databases.

1.5.1. ERD Online Movie Booking System:

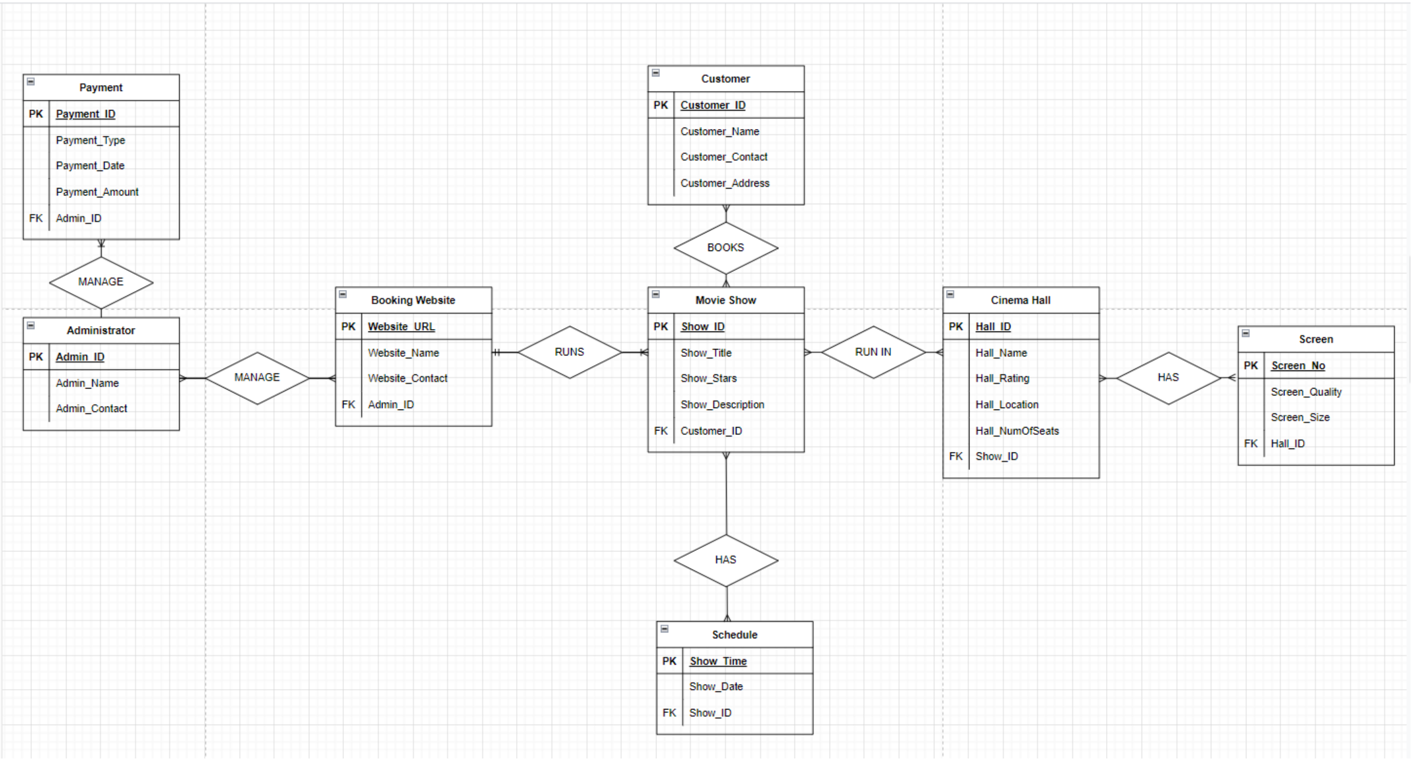


Figure 10. ERD

The initial ERD model of the Online Movie Booking System has minor flaws in its design, from the diagram one can observe that the system is not fully optimized, there are some unnecessary attributes which can be modified for better performance, reduced data inconsistency and redundancy. Secondly, there are also unnecessary entities which can be merged with other entities for clear visuals and better performance. Finally, last but not the least it is not visually appealing.

1.6. Use Case Diagram:

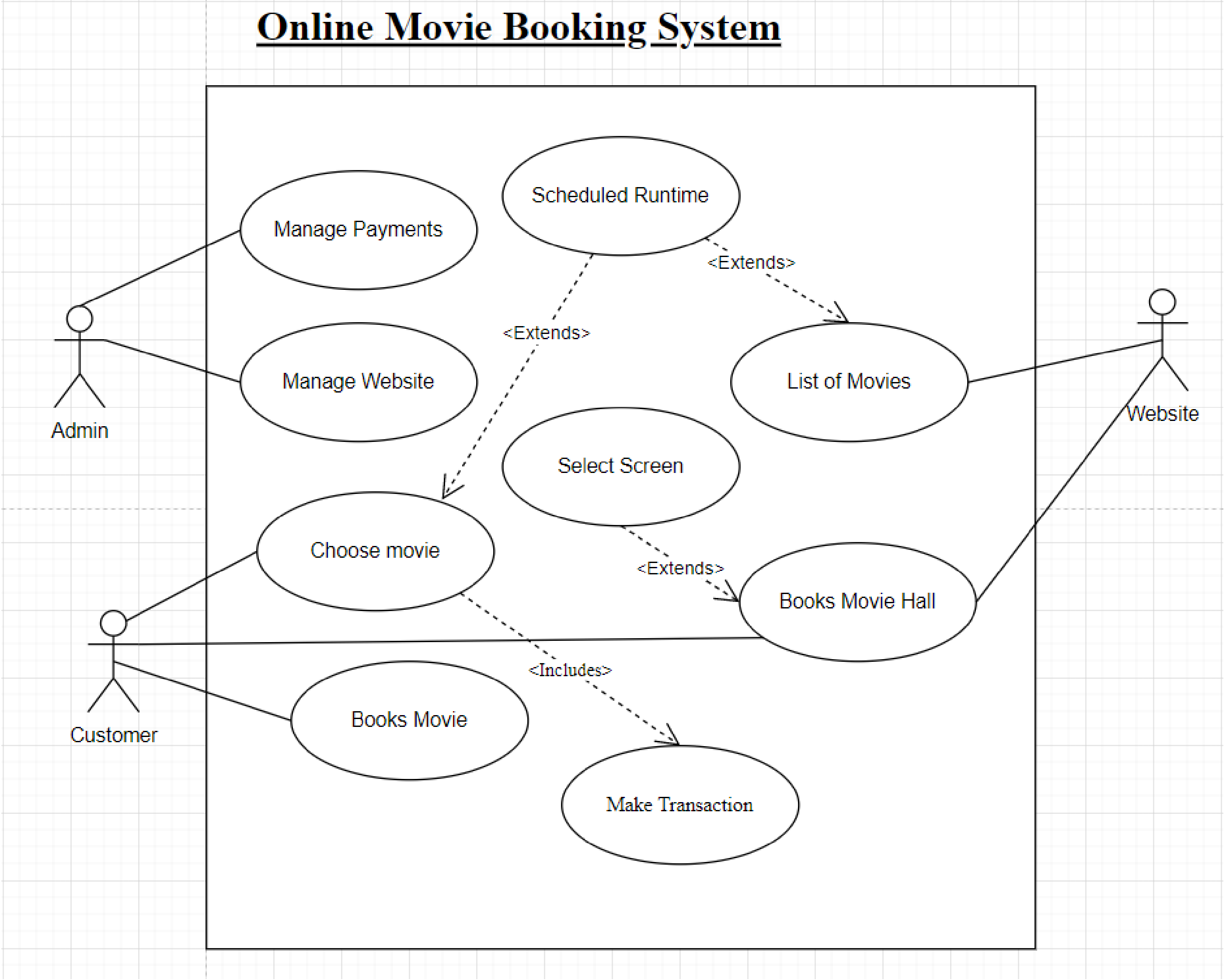


Figure 11. Use Case Diagram

2. NORMALIZATION:

2.1. Assumptions:

I. There can be many Admins looking over the system.

II. Payment is not only accepted via hand in cash but also other methods such as online pay, phone pay, cheque etc.

III. The website URL is permanent so is the name and the contact.

IV. There are many movies shows in the website to choose from for the customers. It is regularly updated by the Admin.

V. Customers can pay with any method by going to the booking website and selecting the show they want to watch.

VI. The show date is pre-available at the website for pre-booking.

VII. There are several cinema halls under the company which are international. Even though they share the same name, they differ by their location which is written at the end of the hall's name.

2.2. Normalization:

Normalization is a strategy used in database to reduce data redundancy and removing undesired traits like Insertion, Update, and Deletion Anomalies. Larger tables are split into smaller tables by normalization, which connect them together through relationships. By eliminating redundant data, normalization in SQL ensures that data is stored logically.

2.2.1. Un-Normalized Form (UNF):

Relation is un-normalized if it has not had any normalization rules applied to it and if it suffers from various anomalies. The tables already have some normalization rules applied to them hence, there is no need for UNF.

I. Administrator:

Administrator (Admin_ID (PK), Admin_Name, Admin_Contact)

II. Payment:

Payment (Payment_ID (PK), Payment_Type, Payment_Date, Payment_Amount, Admin_ID(FK))

III. Booking Website:

Booking Website (Website_URL (PK), Website_Name, Website_Contact, Admin_ID (FK))

IV. Movie Show:

Movie Show (Show_ID (PK), Show_Title, Show_Stars, Show_Description, Customer_ID (FK))

V. Customer:

Customer (Customer_ID (PK), Customer_Name, Customer_Contact, Customer_Address)

VI. Schedule:

Schedule (Show_Time, Show_Date, Show_ID (FK))

VII. Cinema Hall:

Cinema Hall (Hall_ID (PK), Hall_Name, Hall_Rating, Hall_Location, Hall_NumOfSeats, Show_ID (FK))

VIII. Screen:

Screen (Screen_No (PK), Screen_Quality, Screen_Size, Hall_ID (FK))

2.2.2. First Normal Form (1NF):

- All rows must be uniquely identified
- Each cell must only contain a single value
- Each value should non divisible

I. Administrator:

Administrator (Admin_ID (PK), Admin_FName, Admin_LName, Admin_Contact)

The Admin_Name is divided into sub-group, first name and last name, thus making all the values non-divisible and fulfilling all the necessary conditions.

II. Payment:

Payment (Payment_ID (PK), Payment_Type, Payment_Date, Payment_Amount, Admin_ID(FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

III. Booking Website:

Booking Website (Website_URL (PK), Website_Name, Website_Contact, Admin_ID (FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

IV. Movie Show:

Movie Show (Show_ID (PK), Show_Title, Show_Stars, Show_Description, Customer_ID (FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

V. Customer:

Customer (Customer_ID (PK), Customer_FName, Customer_LName Customer_Contact, Customer_Address)

The Customer_Name is divided into sub-group, first name and last name, thus making all the values non-divisible and fulfilling all the necessary conditions.

VI. Schedule:

Schedule (Show_Time, Show_Date, Show_ID (FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

VII. Cinema Hall:

Cinema Hall (Hall_ID (PK), Hall_Name, Hall_Rating, Hall_Location, Hall_NumOfSeats, Show_ID (FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

VIII. Screen:

Screen (Screen_No (PK), Screen_Quality, Screen_Size, Hall_ID (FK))

As all the rows are uniquely identified, each cell contains only a single value and each value are non-divisible thus fulfilling all the necessary conditions, hence no change done.

2.2.3. Second Normal Form (2NF):

- Database should be in 1NF
- There should not be any Partial Functional Dependencies

I. Administrator:

Administrator (Admin_ID (PK), Admin_FName, Admin_LName, Admin_Contact)

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

II. Payment:

Payment (Payment_ID (PK), Payment_Type, Payment_Date, Payment_Amount, Admin_ID(FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

III. Booking Website:

Booking Website (Website_URL (PK), Website_Name, Website_Contact, Admin_ID (FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

IV. Movie Show:

Movie Show (Show_ID (PK), Show_Title, Show_Stars, Show_Description, Customer_ID (FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

V. Customer:

Customer (Customer_ID (PK), Customer_FName, Customer_LName, Customer_Contact, Customer_Address)

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

VI. Schedule:

Schedule (Show_Time, Show_Date, Show_ID (FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

VII. Cinema Hall:

Cinema Hall (Hall_ID (PK), Hall_Name, Hall_Rating, Hall_Location, Hall_NumOfSeats, Show_ID (FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

VIII. Screen:

Screen (Screen_No (PK), Screen_Quality, Screen_Size, Hall_ID (FK))

Database is in 1NF and also there are is no Partial Functional Dependencies. Hence no change is done.

2.2.4. 3NF:

- Database should have gone through 1NF and 2NF normalization rules
- There should not be any Transitive Dependencies

I. Administrator:

Administrator (Admin_ID (PK), Admin_FName, Admin_LName, Admin_Contact)

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

II. Payment:

Payment (Payment_ID (PK), Payment_Type, Payment_Date, Payment_Amount, Admin_ID(FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

III. Booking Website:

Booking Website (Website_URL (PK), Website_Name, Website_Contact, Admin_ID (FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

IV. Movie Show:

Movie Show (Show_ID (PK), Show_Title, Show_Stars, Show_Description, Customer_ID (FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

V. Customer:

Customer (Customer_ID (PK), Customer_FName, Customer_LName, Customer_Contact, Customer_Address)

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

VI. Schedule:

Schedule (Show_Time, Show_Date, Show_ID (FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

VII. Cinema Hall:

Cinema Hall (Hall_ID (PK), Hall_Name, Hall_Rating, Hall_Location, Hall_NumOfSeats, Show_ID (FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

VIII. Screen:

Screen (Screen_No (PK), Screen_Quality, Screen_Size, Hall_ID (FK))

Database is in 1NF and 2NF also there are is no Transitive Dependencies. Hence no change is done.

2.3. Post-Normalization ERD:

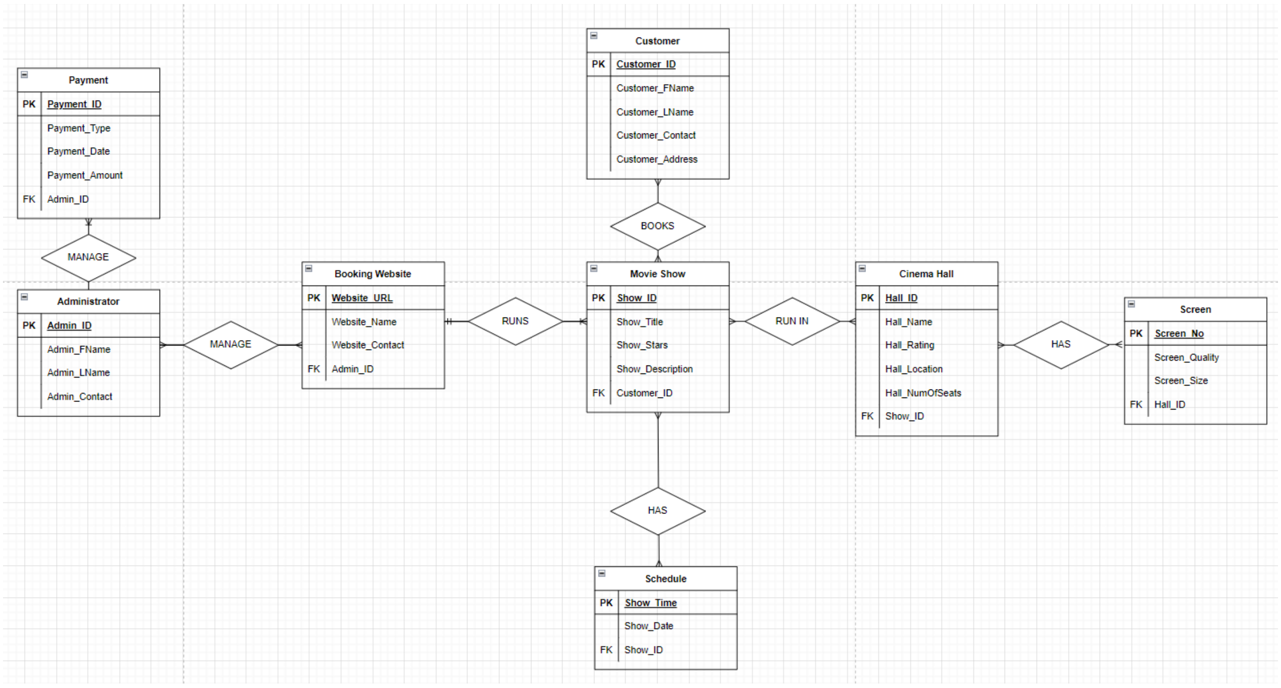


Figure 12. Post Normalization ERD

After normalization we can see that the tables are much more efficient. The normalization rules helped us reduce data redundancy and inconsistency. The diagram has also been much more informative and easier to grasp as each entity and attributes are properly categorized.

2.4. Data Dictionary:

Table	Attribute	Data Type	Size	Constraints	Description	Example
Administrator	Admin_ID	Varchar	10	PRIMARY KEY	Unique ID specific to a single Admin	0000000001
	Admin_FName	Varchar	15	NOT NULL	First name of the specific Admin	Nayan
	Admin_LName	Varchar	15	NOT NULL	Last name of the specific Admin	Khanal
	Admin_Contact	Varchar	10	NOT NULL	Contact detail of the specific Admin	9804302507

Table	Attribute	Data Type	Size	Constraints	Description	Example
Payment	Payment_ID	Varchar	10	PRIMARY KEY	Unique ID specific to the payment	0101010101
	Payment_Type	Varchar	10	NOT NULL	Method of transaction like Physical, Online or Scan	Esewa
	Payment_Date	Date	--------	NOT NULL	Date of the payment	2022-08-14
	Payment_Amount	Varchar	5	NOT NULL	Total amount paid for the show	300
	Admin_ID	-----------	--------	FOREIGN KEY	--------------	----------------

Table	Attribute	Data Type	Size	Constraints	Description	Example
Booking Website	Website_URL	Varchar	20	PRIMARY KEY	The URL of the company's website	thebasement.com
	Website_Name	Varchar	15	NOT NULL	Name of the Company	The Basement
	Website_Contact	Varchar	30	NOT NULL	Contact detail of the website	enquiries@thebasement.com
	Admin_ID	---------	-----	FOREIGN KEY	--------------	---------------------------------

Table	Attribute	Data Type	Size	Constraints	Description	Example
Movie Show	Show_ID	Varchar	10	PRIMARY KEY	The specific ID of the movie	1111111110
	Show_Title	Varchar	50	NOT NULL	Name of the movie	Red Dead Redemption 2
	Show_Stars	Varchar	4	NOT NULL	Total star rating out of five	Five
	Show_Description	Varchar	100	NOT NULL	Brief description of the movie without any spoilers	The movie is set in the old wild-west frontier. The protagonist is a man named Arthur Morgan. He is a feared gun-slinger. As the time progressed the wild west was slowly fading away and more modern civilized methods were being implemented. The plot embraces Arthur and his gang trying to turn away from this change. Can they succeed to escape the change or will they succumb to it or will they die trying.
	Customer_ID	---------	-----	FOREIGN KEY	-------------	--------------------------------

Table	Attribute	Data Type	Size	Constraints	Description	Example
Customer	Customer_ID	Varchar	10	PRIMARY KEY	The ID of the specific customer	0224450457
	Customer_FName	Varchar	15	NOT NULL	First Name of the Customer	John
	Customer_LName	Varchar	15	NOT NULL	Last Name of the Customer	Marston
	Customer_Contact	Varchar	30	NOT NULL	Contact detail of the customer	9811071477
	Customer_Address	Varchar	50	NOT NULL	Address of the customer	J-801, New Hanover, Saint Denis

Table	Attribute	Data Type	Size	Constraints	Description	Example
Schedule	Show_Time	Time	-----	PRIMARY KEY	The time of the broadcast	12:00:00
	Show_Date	Date	-----	NOT NULL	Date of the broadcast	2022-10-26
	Show_ID	---------	-----	FOREIGN KEY	-------------	---------------------------------

Table	Attribute	Data Type	Size	Constraints	Description	Example
Cinema Hall	Hall_ID	Varchar	10	PRIMARY KEY	The ID of the specific Hall	0222323092
	Hall_Name	Varchar	50	NOT NULL	Name of the Hall	The Basement, Romania
	Hall_Rating	Varchar	5	NOT NULL	Rating of the Hall out of ten	Ten
	Hall_Location	Varchar	50	NOT NULL	Location of the Hall	Lemoyne, Romania
	Hall_NumOfSeats	Varchar	3	NOT NULL	Number of seats available at the Hall	100
	Show_ID	---------	----	FOREIGN KEY	--------------	-------------------------------

Table	Attribute	Data Type	Size	Constraints	Description	Example
Screen	Screen_No	Varchar	10	PRIMARY KEY	The Serial Number of the Screen	AB12CD34EF
	Screen_Quality	Varchar	10	NOT NULL	The quality of the Screen	4K
	Screen_Size	Varchar	10	NOT NULL	The size of the Screen	65inch
	Hall_ID	---------	-----	FOREIGN KEY	--------------	---------------------------------

3. IMPLEMENTATION:

3.1. Creation Of Tables:

To create a table in a database, ‘CREATE TABLE’ command is used.

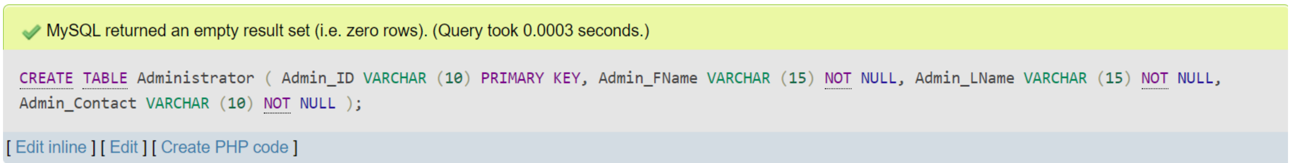
Syntax:

```
CREATE TABLE table_name (
    column1 datatype constraint,
    column2 datatype constraint,
    column3 datatype constraint,
   ....
);
```

Choosing a suitable table_name along with the columns it will house with the suitable datatypes are the only things needed. Along with these, user can also include constraints in their syntax which is optional. Constraints are used to enforce rules for data in the table.

I. Administrator:

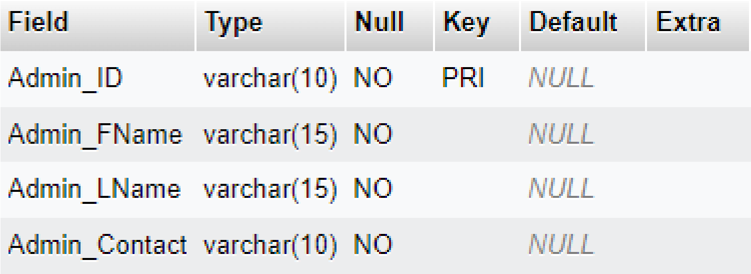
```
CREATE TABLE Administrator (
    Admin_ID VARCHAR (10) PRIMARY KEY,
    Admin_FName VARCHAR (15) NOT NULL,
    Admin_FName VARCHAR (15) NOT NULL,
    Admin_Contact VARCHAR (10) NOT NULL
);
```



MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
CREATE TABLE Administrator ( Admin_ID VARCHAR (10) PRIMARY KEY, Admin_FName VARCHAR (15) NOT NULL, Admin_LName VARCHAR (15) NOT NULL,
Admin_Contact VARCHAR (10) NOT NULL );
```

Figure 13. Admin Command

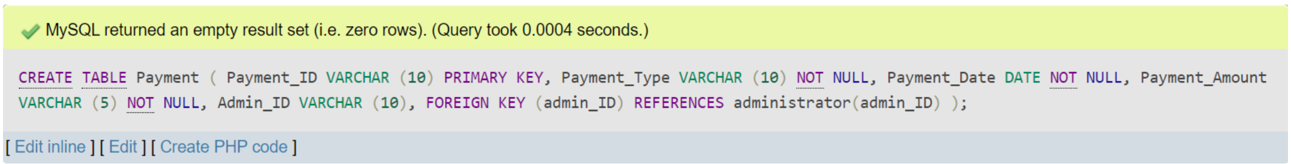


Field	Type	Null	Key	Default	Extra
Admin_ID	varchar(10)	NO	PRI	*NULL*	
Admin_FName	varchar(15)	NO		*NULL*	
Admin_LName	varchar(15)	NO		*NULL*	
Admin_Contact	varchar(10)	NO		*NULL*	

Figure 14. Admin Describe

II. Payment:

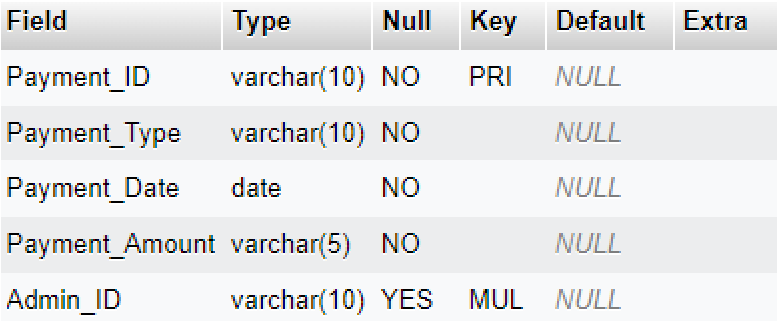
```
CREATE TABLE Payment (
   Payment_ID VARCHAR (10) PRIMARY KEY,
   Payment_Type VARCHAR (10) NOT NULL,
   Payment_Date DATE NOT NULL,
   Payment_Amount VARCHAR (5) NOT NULL,
   Admin_ID VARCHAR (10),
   FOREIGN KEY (admin_ID) REFERENCES administrator(admin_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)

```
CREATE TABLE Payment ( Payment_ID VARCHAR (10) PRIMARY KEY, Payment_Type VARCHAR (10) NOT NULL, Payment_Date DATE NOT NULL, Payment_Amount
VARCHAR (5) NOT NULL, Admin_ID VARCHAR (10), FOREIGN KEY (admin_ID) REFERENCES administrator(admin_ID) );
```

Figure 15. Payment Command

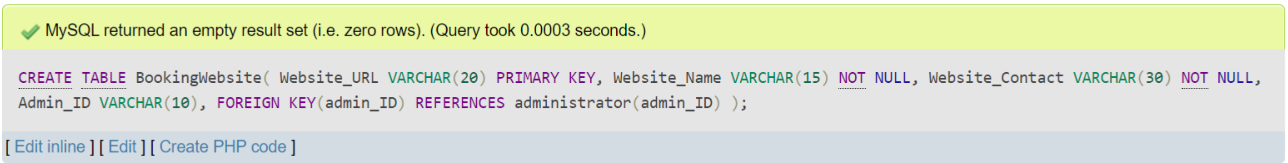


Field	Type	Null	Key	Default	Extra
Payment_ID	varchar(10)	NO	PRI	*NULL*	
Payment_Type	varchar(10)	NO		*NULL*	
Payment_Date	date	NO		*NULL*	
Payment_Amount	varchar(5)	NO		*NULL*	
Admin_ID	varchar(10)	YES	MUL	*NULL*	

Figure 16. Payment Describe

III. Booking Website:

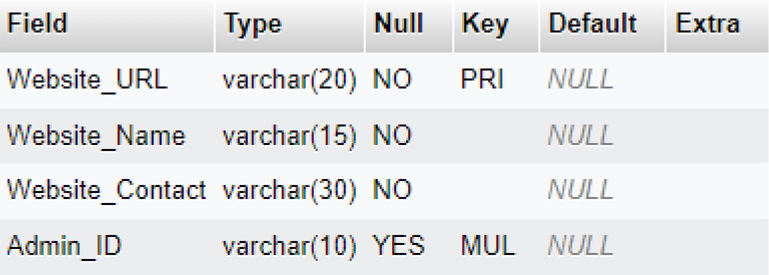
```
CREATE TABLE BookingWebsite (
   Website_URL VARCHAR (20) PRIMARY KEY,
   Website_Name VARCHAR (15) NOT NULL,
   Website_Contact VARCHAR (30) NOT NULL,
   Admin_ID VARCHAR (10),
   FOREIGN KEY (admin_ID) REFERENCES administrator(admin_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
CREATE TABLE BookingWebsite( Website_URL VARCHAR(20) PRIMARY KEY, Website_Name VARCHAR(15) NOT NULL, Website_Contact VARCHAR(30) NOT NULL,
Admin_ID VARCHAR(10), FOREIGN KEY(admin_ID) REFERENCES administrator(admin_ID) );
```

Figure 17. BookingWebsite Command

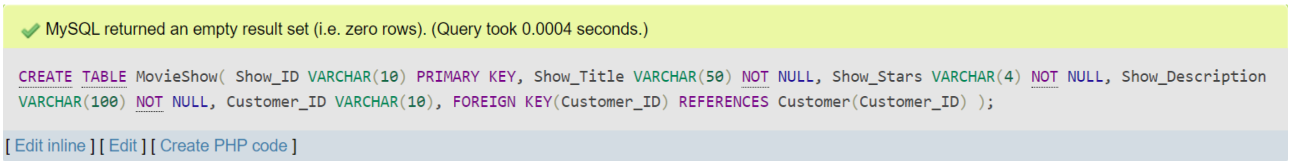


Field	Type	Null	Key	Default	Extra
Website_URL	varchar(20)	NO	PRI	*NULL*	
Website_Name	varchar(15)	NO		*NULL*	
Website_Contact	varchar(30)	NO		*NULL*	
Admin_ID	varchar(10)	YES	MUL	*NULL*	

Figure 18. BookingWebsite Describe

IV. Movie Show:

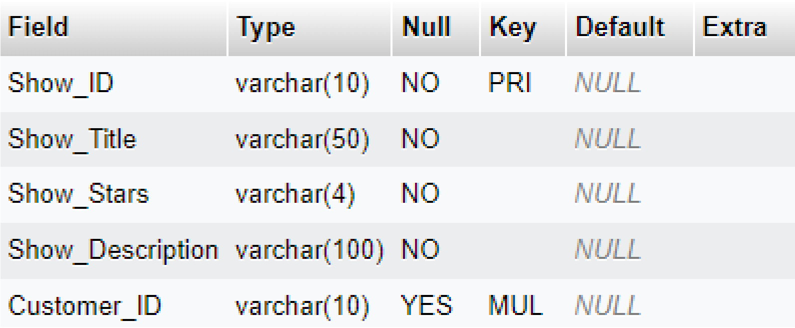
```
CREATE TABLE MovieShow (
    Show_ID VARCHAR (10) PRIMARY KEY,
    Show_Title VARCHAR (50) NOT NULL,
    Show_Stars VARCHAR (4) NOT NULL,
    Show_Description VARCHAR (100) NOT NULL,
    Customer_ID VARCHAR (10),
    FOREIGN KEY(Customer_ID) REFERENCES Customer (Customer_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0004 seconds.)

```
CREATE TABLE MovieShow( Show_ID VARCHAR(10) PRIMARY KEY, Show_Title VARCHAR(50) NOT NULL, Show_Stars VARCHAR(4) NOT NULL, Show_Description
VARCHAR(100) NOT NULL, Customer_ID VARCHAR(10), FOREIGN KEY(Customer_ID) REFERENCES Customer(Customer_ID) );
```

Figure 19. MovieShow Command

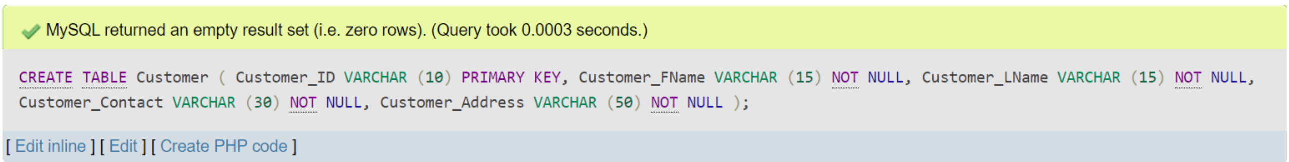


Field	Type	Null	Key	Default	Extra
Show_ID	varchar(10)	NO	PRI	*NULL*	
Show_Title	varchar(50)	NO		*NULL*	
Show_Stars	varchar(4)	NO		*NULL*	
Show_Description	varchar(100)	NO		*NULL*	
Customer_ID	varchar(10)	YES	MUL	*NULL*	

Figure 20. MovieShow Describe

V. Customer:

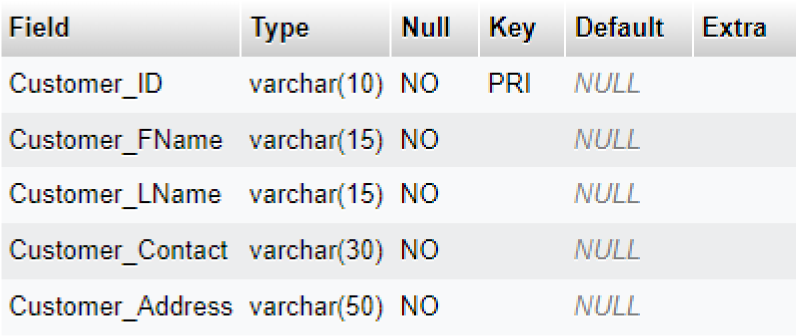
```
CREATE TABLE Customer (
   Customer_ID VARCHAR (10) PRIMARY KEY,
   Customer_FName VARCHAR (15) NOT NULL,
   Customer_LName VARCHAR (15) NOT NULL,
   Customer_Contact VARCHAR (30) NOT NULL,
   Customer_Address VARCHAR (50) NOT NULL
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
CREATE TABLE Customer ( Customer_ID VARCHAR (10) PRIMARY KEY, Customer_FName VARCHAR (15) NOT NULL, Customer_LName VARCHAR (15) NOT NULL,
Customer_Contact VARCHAR (30) NOT NULL, Customer_Address VARCHAR (50) NOT NULL );
```

Figure 21. Customer Command

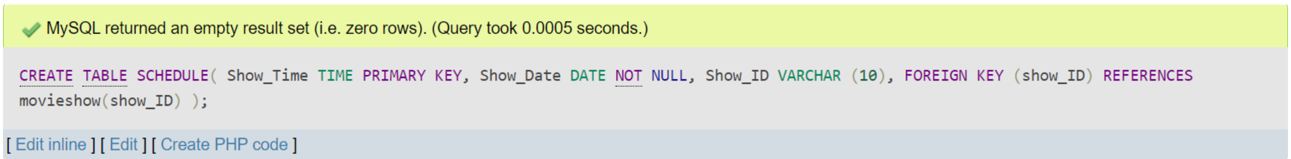


Field	Type	Null	Key	Default	Extra
Customer_ID	varchar(10)	NO	PRI	NULL	
Customer_FName	varchar(15)	NO		NULL	
Customer_LName	varchar(15)	NO		NULL	
Customer_Contact	varchar(30)	NO		NULL	
Customer_Address	varchar(50)	NO		NULL	

Figure 22. Customer Describe

VI. Schedule:

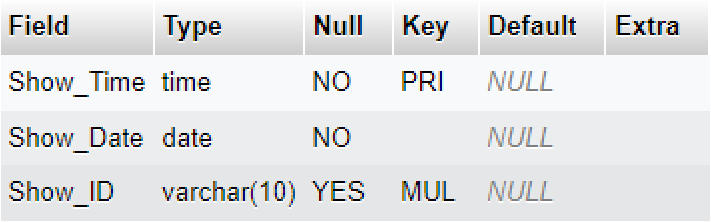
```
CREATE TABLE SCHEDULE (
    Show_Time TIME PRIMARY KEY,
    Show_Date DATE NOT NULL,
    Show_ID VARCHAR (10),
    FOREIGN KEY (show_ID) REFERENCES movieshow(show_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0005 seconds.)

```
CREATE TABLE SCHEDULE( Show_Time TIME PRIMARY KEY, Show_Date DATE NOT NULL, Show_ID VARCHAR (10), FOREIGN KEY (show_ID) REFERENCES
movieshow(show_ID) );
```

Figure 23. Schedule Command

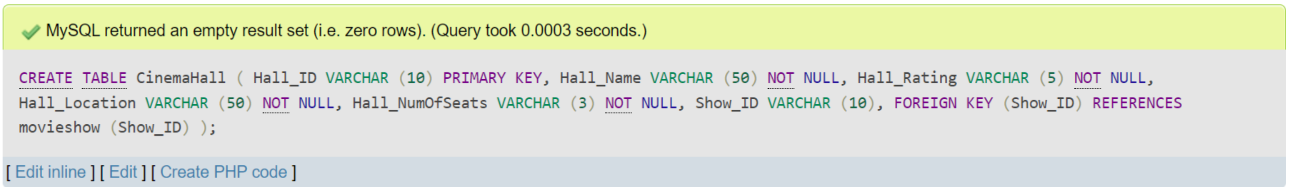


Field	Type	Null	Key	Default	Extra
Show_Time	time	NO	PRI	*NULL*	
Show_Date	date	NO		*NULL*	
Show_ID	varchar(10)	YES	MUL	*NULL*	

Figure 24. Schedule Describe

VII. Cinema Hall:

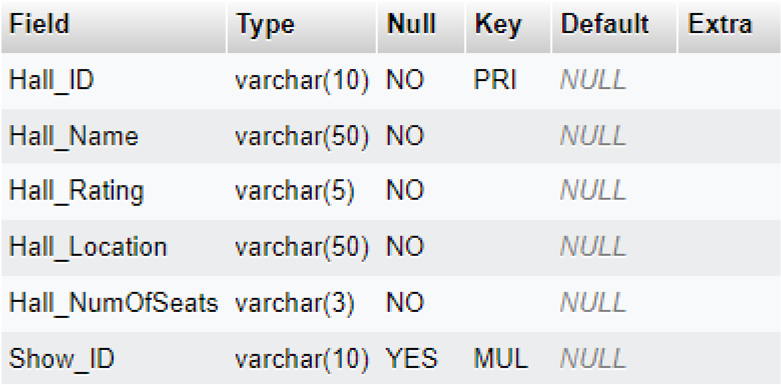
```
CREATE TABLE CinemaHall (
    Hall_ID VARCHAR (10) PRIMARY KEY,
    Hall_Name VARCHAR (50) NOT NULL,
    Hall_Rating VARCHAR (5) NOT NULL,
    Hall_Location VARCHAR (50) NOT NULL,
    Hall_NumOfSeats VARCHAR (3) NOT NULL,
    Show_ID VARCHAR (10),
    FOREIGN KEY (Show_ID) REFERENCES movieshow (Show_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
CREATE TABLE CinemaHall ( Hall_ID VARCHAR (10) PRIMARY KEY, Hall_Name VARCHAR (50) NOT NULL, Hall_Rating VARCHAR (5) NOT NULL,
Hall_Location VARCHAR (50) NOT NULL, Hall_NumOfSeats VARCHAR (3) NOT NULL, Show_ID VARCHAR (10), FOREIGN KEY (Show_ID) REFERENCES
movieshow (Show_ID) );
```

Figure 25. CinemaHall Command

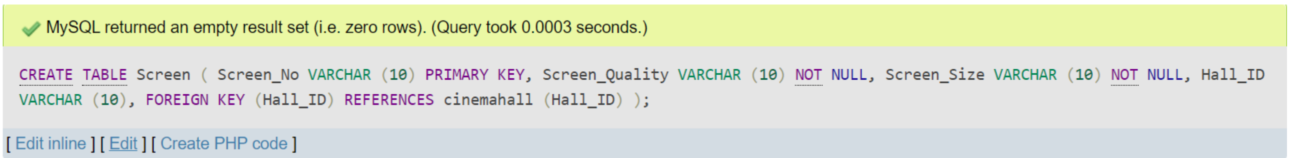


Field	Type	Null	Key	Default	Extra
Hall_ID	varchar(10)	NO	PRI	NULL	
Hall_Name	varchar(50)	NO		NULL	
Hall_Rating	varchar(5)	NO		NULL	
Hall_Location	varchar(50)	NO		NULL	
Hall_NumOfSeats	varchar(3)	NO		NULL	
Show_ID	varchar(10)	YES	MUL	NULL	

Figure 26. CinemaHall Describe

VIII. Screen:

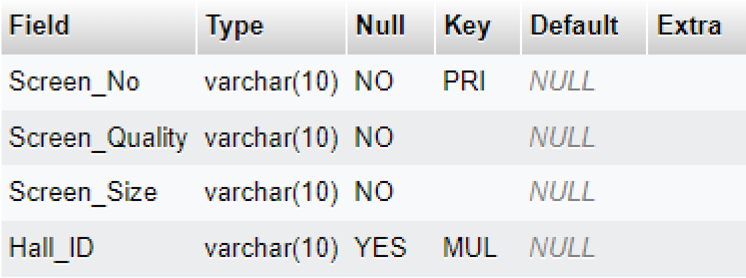
```
CREATE TABLE Screen (
    Screen_No VARCHAR (10) PRIMARY KEY,
    Screen_Quality VARCHAR (10) NOT NULL,
    Screen_Size VARCHAR (10) NOT NULL,
    Hall_ID VARCHAR (10),
    FOREIGN KEY (Hall_ID) REFERENCES cinemahall (Hall_ID)
);
```



✓ MySQL returned an empty result set (i.e. zero rows). (Query took 0.0003 seconds.)

```
CREATE TABLE Screen ( Screen_No VARCHAR (10) PRIMARY KEY, Screen_Quality VARCHAR (10) NOT NULL, Screen_Size VARCHAR (10) NOT NULL, Hall_ID
VARCHAR (10), FOREIGN KEY (Hall_ID) REFERENCES cinemahall (Hall_ID) );
```

Figure 27. Screen Command



Field	Type	Null	Key	Default	Extra
Screen_No	varchar(10)	NO	PRI	NULL	
Screen_Quality	varchar(10)	NO		NULL	
Screen_Size	varchar(10)	NO		NULL	
Hall_ID	varchar(10)	YES	MUL	NULL	

Figure 28. Screen Describe

3.2. Populating Data:

To populate data in the table, we can use “INSERT INTO” command.

Syntax:

INSERT INTO *table_name* (*column1*, *column2*, *column3*, ...)
VALUES (*value1*, *value2*, *value3*, ...);

Choosing the table_name then the respective column names and finally choosing what values to insert is all that is needed for this command to function.

3.2.1. Commit:

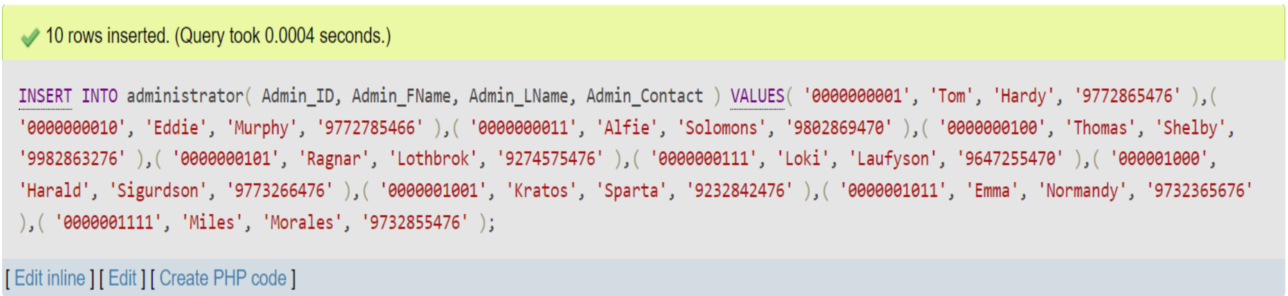
The transactional command used to save database changes triggered by a transaction is called COMMIT. All database transactions that have occurred since the last ROLLBACK or COMMIT command are saved when you use the COMMIT command.

The COMMIT command has the following syntax:

COMMIT;

I. Administrator:

```
INSERT INTO administrator (
   Admin_ID,
   Admin_FName,
   Admin_LName,
   Admin_Contact
)
VALUES (
   '0000000001',
   'Tom',
   'Hardy',
   '9772865476'
),(
   '0000000010',
   'Eddie',
   'Murphy',
   '9772785466'
),(
   '0000000011',
   'Alfie',
   'Solomons',
   '9802869470'
),(
   '0000000100',
   'Thomas',
   'Shelby',
   '9982863276'
),(
   '0000000101',
   'Ragnar',
   'Lothbrok',
   '9274575476'
),(
   '0000000111',
   'Loki',
   'Laufyson',
   '9647255470'
),(
   '000001000',
   'Harald',
   'Sigurdson',
   '9773266476'
),(
   '0000001001',
   'Kratos',
   'Sparta',
   '9232842476'
),(
   '0000001011',
   'Emma',
   'Normandy',
   '9732365676'
),(
   '0000001111',
   'Miles',
   'Morales',
   '9732855476'
);
```

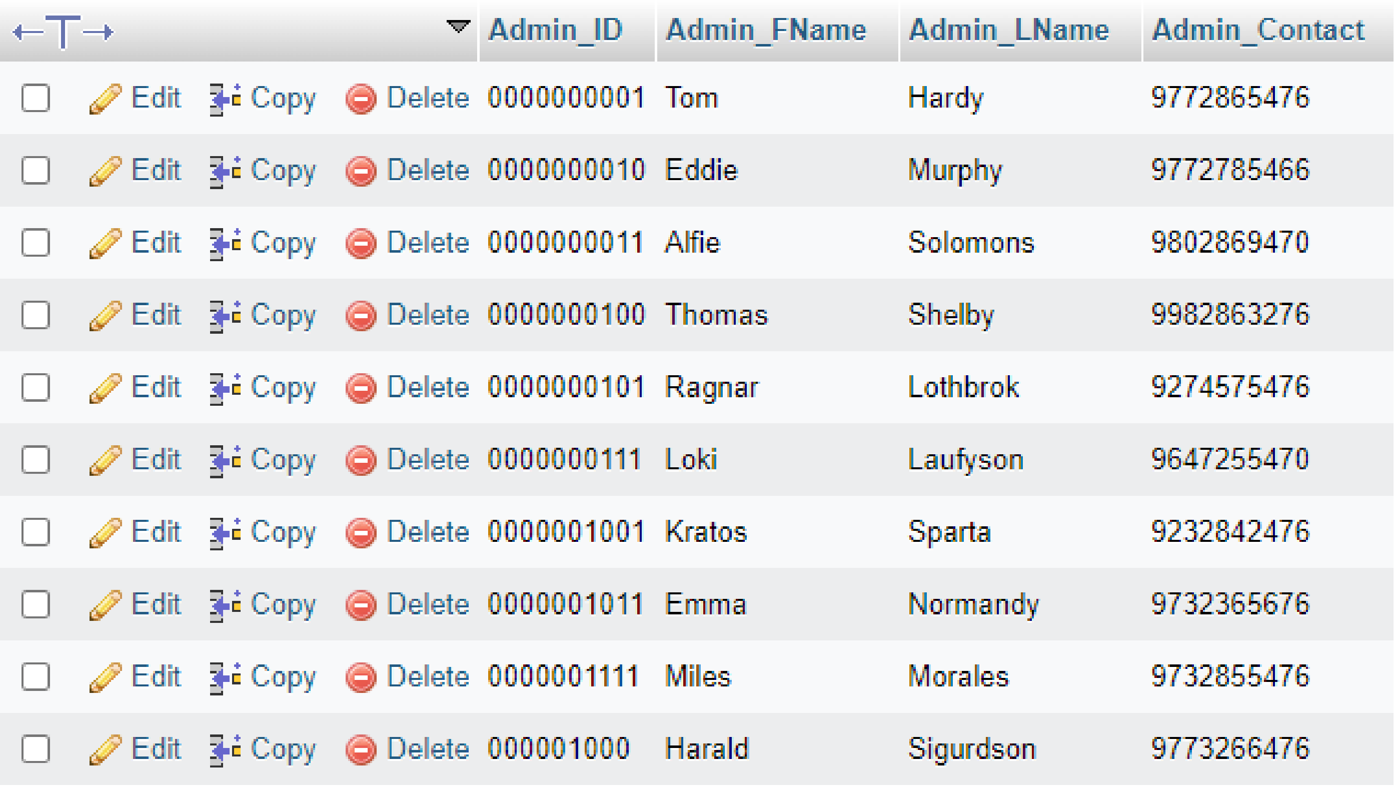


✓ 10 rows inserted. (Query took 0.0004 seconds.)

```
INSERT INTO administrator( Admin_ID, Admin_FName, Admin_LName, Admin_Contact ) VALUES( '0000000001', 'Tom', 'Hardy', '9772865476' ),(
'0000000010', 'Eddie', 'Murphy', '9772785466' ),( '0000000011', 'Alfie', 'Solomons', '9802869470' ),( '0000000100', 'Thomas', 'Shelby',
'9982863276' ),( '0000000101', 'Ragnar', 'Lothbrok', '9274575476' ),( '0000000111', 'Loki', 'Laufyson', '9647255470' ),( '000001000',
'Harald', 'Sigurdson', '9773266476' ),( '0000001001', 'Kratos', 'Sparta', '9232842476' ),( '0000001011', 'Emma', 'Normandy', '9732365676'
),( '0000001111', 'Miles', 'Morales', '9732855476' );
```

[Edit inline] [Edit] [Create PHP code]

Figure 29. Inserting into admin table

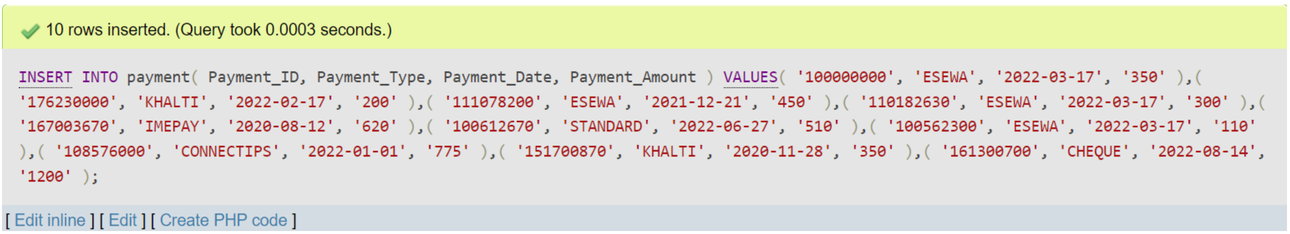


←T→				Admin_ID	Admin_FName	Admin_LName	Admin_Contact
□	Edit	Copy	Delete	0000000001	Tom	Hardy	9772865476
□	Edit	Copy	Delete	0000000010	Eddie	Murphy	9772785466
□	Edit	Copy	Delete	0000000011	Alfie	Solomons	9802869470
□	Edit	Copy	Delete	0000000100	Thomas	Shelby	9982863276
□	Edit	Copy	Delete	0000000101	Ragnar	Lothbrok	9274575476
□	Edit	Copy	Delete	0000000111	Loki	Laufyson	9647255470
□	Edit	Copy	Delete	0000001001	Kratos	Sparta	9232842476
□	Edit	Copy	Delete	0000001011	Emma	Normandy	9732365676
□	Edit	Copy	Delete	0000001111	Miles	Morales	9732855476
□	Edit	Copy	Delete	000001000	Harald	Sigurdson	9773266476

Figure 30. Displaying Admin table

II. Payment:

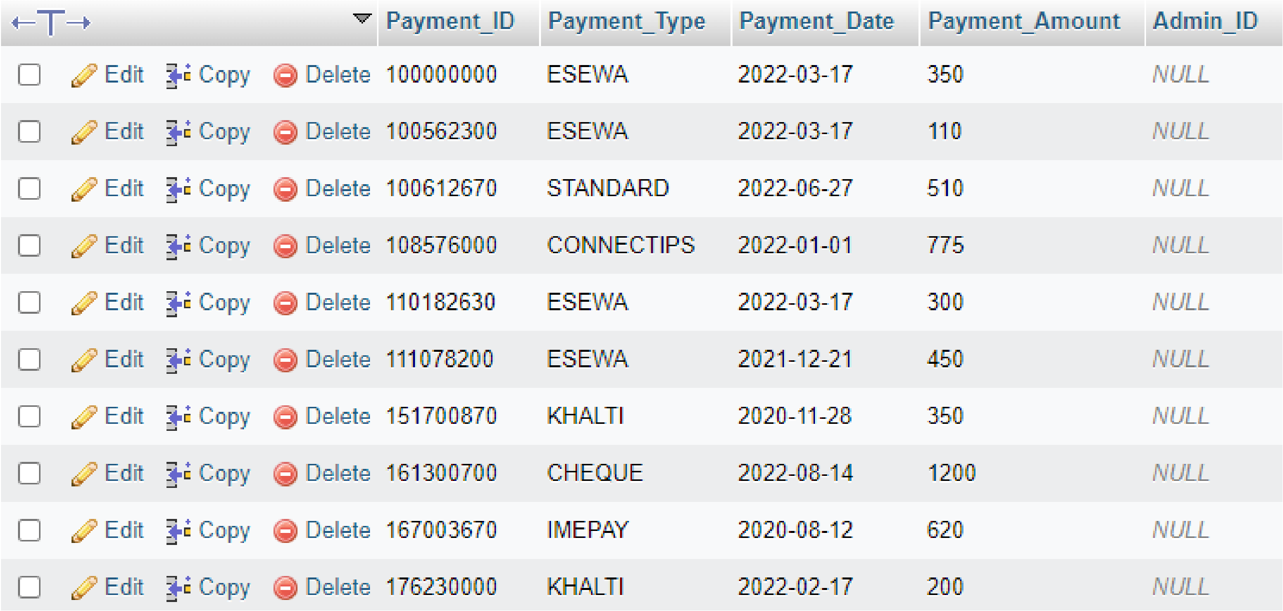
```
INSERT INTO payment (
    Payment_ID,
    Payment_Type,
    Payment_Date,
    Payment_Amount
)
VALUES (
    '100000000',
    'ESEWA',
    '2022-03-17',
    '350'
),(
    '176230000',
    'KHALTI',
    '2022-02-17',
    '200'
),(
    '111078200',
    'ESEWA',
    '2021-12-21',
    '450'
),(
    '110182630',
    'ESEWA',
    '2022-03-17',
    '300'
),(
    '167003670',
    'IMEPAY',
    '2020-08-12',
    '620'
),(
    '100612670',
    'STANDARD',
    '2022-06-27',
    '510'
),(
    '100562300',
    'ESEWA',
    '2022-03-17',
    '110'
),(
    '108576000',
    'CONNECTIPS',
    '2022-01-01',
    '775'
),(
    '151700870',
    'KHALTI',
    '2020-11-28',
    '350'
),(
    '161300700',
    'CHEQUE',
    '2022-08-14',
    '1200'
);
```



✓ 10 rows inserted. (Query took 0.0003 seconds.)

```
INSERT INTO payment( Payment_ID, Payment_Type, Payment_Date, Payment_Amount ) VALUES( '100000000', 'ESEWA', '2022-03-17', '350' ),(
'176230000', 'KHALTI', '2022-02-17', '200' ),( '111078200', 'ESEWA', '2021-12-21', '450' ),( '110182630', 'ESEWA', '2022-03-17', '300' ),(
'167003670', 'IMEPAY', '2020-08-12', '620' ),( '100612670', 'STANDARD', '2022-06-27', '510' ),( '100562300', 'ESEWA', '2022-03-17', '110'
),( '108576000', 'CONNECTIPS', '2022-01-01', '775' ),( '151700870', 'KHALTI', '2020-11-28', '350' ),( '161300700', 'CHEQUE', '2022-08-14',
'1200' );
```

Figure 31. Inserting into payment table

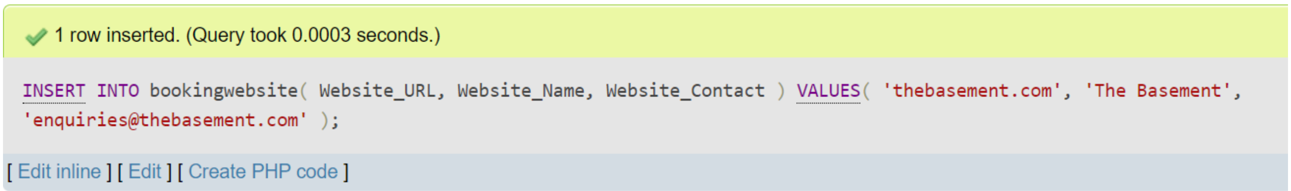


←T→	Payment_ID	Payment_Type	Payment_Date	Payment_Amount	Admin_ID
□ Edit Copy Delete	100000000	ESEWA	2022-03-17	350	NULL
□ Edit Copy Delete	100562300	ESEWA	2022-03-17	110	NULL
□ Edit Copy Delete	100612670	STANDARD	2022-06-27	510	NULL
□ Edit Copy Delete	108576000	CONNECTIPS	2022-01-01	775	NULL
□ Edit Copy Delete	110182630	ESEWA	2022-03-17	300	NULL
□ Edit Copy Delete	111078200	ESEWA	2021-12-21	450	NULL
□ Edit Copy Delete	151700870	KHALTI	2020-11-28	350	NULL
□ Edit Copy Delete	161300700	CHEQUE	2022-08-14	1200	NULL
□ Edit Copy Delete	167003670	IMEPAY	2020-08-12	620	NULL
□ Edit Copy Delete	176230000	KHALTI	2022-02-17	200	NULL

Figure 32. Displaying Payment table

III. Booking Website:

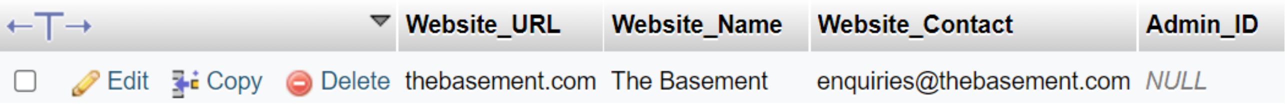
```
INSERT INTO bookingwebsite (
    Website_URL,
    Website_Name,
    Website_Contact
)
VALUES (
    'thebasement.com',
    'The Basement',
    'enquiries@thebasement.com'
);
```



✓ 1 row inserted. (Query took 0.0003 seconds.)

```
INSERT INTO bookingwebsite( Website_URL, Website_Name, Website_Contact ) VALUES( 'thebasement.com', 'The Basement',
'enquiries@thebasement.com' );
```

Figure 33. Inserting into bookingwebsite table



←T→	Website_URL	Website_Name	Website_Contact	Admin_ID
□ Edit Copy Delete	thebasement.com	The Basement	enquiries@thebasement.com	*NULL*

Figure 34. Displaying bookingwebsite table

As the assumption above in 2.1. this table does not change its values. All three URL, Name and Contact are static.

IV. Movie Show:

```
INSERT INTO movieshow (
    Show_ID,
    Show_Title,
    Show_Stars,
    Show_Description
)
VALUES (
    '4A327',
    'Anabelle',
    'Three',
    'John and Mia Form are attacked by members of a satanic cult that uses their old doll as a conduit to make their life miserable. This unleashes a string of paranormal events in the Forms residence'
),(
    'J817G',
    'Day Shift',
    'Two',
    'A hardworking dad out to provide for his daughter uses a boring pool-cleaning job as a front for his real gig: hunting and killing vampires.'
),(
    'FIN1L',
    'Uncharted',
    'Four',
    'Victor Sullivan recruits Nathan Drake to help him find the lost fortune of Ferdinand Magellan. However, they face competition from Santiago Moncada, who believes that the treasure belongs to him.'
),(
    '98YUP',
    'Sonic the Hedgehog 2',
    'Two',
    'After settling in Green Hills, Sonic is eager to prove that he has what it takes to be a true hero. His test comes when Dr. Robotnik returns with a new partner, Knuckles, in search of a mystical emerald that has the power to destroy civilizations. Sonic teams up with his own sidekick, Tails,
and together they embark on a globe-trotting journey to find the emerald before it falls into the wrong hands.'
),(
    'OK9A7',
    'The Batman',
    'Five',
    'Batman is called to intervene when the mayor of Gotham City is murdered. Soon, his investigation leads him to uncover a web of corruption, linked to his own dark past.'
),(
    'AWE0ME',
    'The Witcher: Nightmare of the Wolf',
    'Five',
    'Vesemir, a cocky young witcher, delights in slaying monsters for coin. When a dangerous new power rises on the Continent, Vesemir learns that some witchering jobs are about more than just money.'
),(
    'J7TY6',
    'Red Notice',
    'Four',
    'In the world of international crime, an Interpol agent attempts to hunt down and capture the worlds most wanted art thief.'
),(
    'KK98G',
    'Venom',
    'Five',
    'After a faulty interview with the Life Foundation ruins his career, former reporter Eddie Brocks life is in pieces. Six months later, he comes across the Life Foundation again, and he comes into contact with an alien symbiote and becomes Venom, a parasitic antihero.'
),(
    'PC78T',
    'Secret Headquarters',
    'Two',
```

'While hanging out after school, Charlie and his friends discover the headquarters of the worlds most powerful superhero hidden beneath his home. When villains attack, they must team up to defend the headquarters and save the world.'

),(

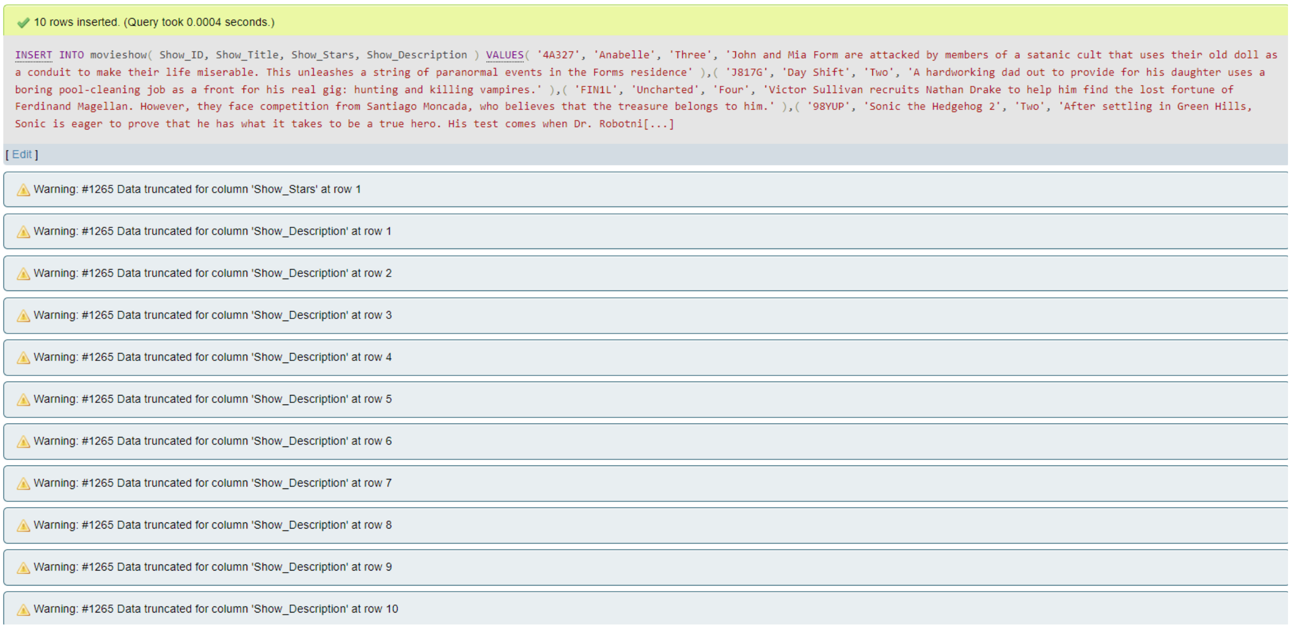
'H7TF5',

'The Grey Man',

'One',

'When the CIAs top asset -- his identity known to no one -- uncovers agency secrets, he triggers a global hunt by assassins set loose by his ex-colleague'

);

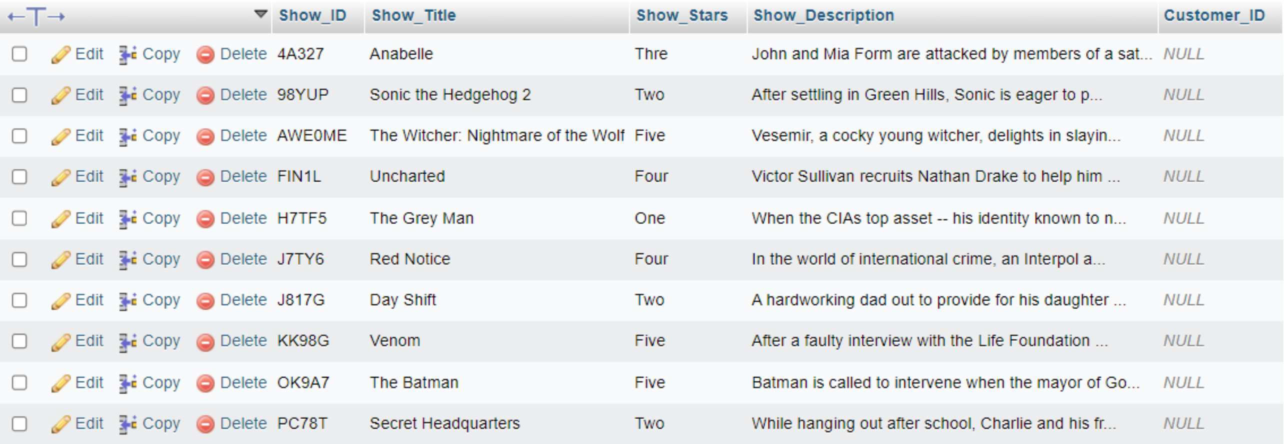


✓ 10 rows inserted. (Query took 0.0004 seconds.)

```
INSERT INTO movieshow( Show_ID, Show_Title, Show_Stars, Show_Description ) VALUES( '4A327', 'Anabelle', 'Three', 'John and Mia Form are attacked by members of a satanic cult that uses their old doll as a conduit to make their life miserable. This unleashes a string of paranormal events in the Forms residence' ),( 'J817G', 'Day Shift', 'Two', 'A hardworking dad out to provide for his daughter uses a boring pool-cleaning job as a front for his real gig: hunting and killing vampires.' ),( 'FIN1L', 'Uncharted', 'Four', 'Victor Sullivan recruits Nathan Drake to help him find the lost fortune of Ferdinand Magellan. However, they face competition from Santiago Moncada, who believes that the treasure belongs to him.' ),( '98YUP', 'Sonic the Hedgehog 2', 'Two', 'After settling in Green Hills, Sonic is eager to prove that he has what it takes to be a true hero. His test comes when Dr. Robotni[...]
```

```
Warning: #1265 Data truncated for column 'Show_Stars' at row 1
Warning: #1265 Data truncated for column 'Show_Description' at row 1
Warning: #1265 Data truncated for column 'Show_Description' at row 2
Warning: #1265 Data truncated for column 'Show_Description' at row 3
Warning: #1265 Data truncated for column 'Show_Description' at row 4
Warning: #1265 Data truncated for column 'Show_Description' at row 5
Warning: #1265 Data truncated for column 'Show_Description' at row 6
Warning: #1265 Data truncated for column 'Show_Description' at row 7
Warning: #1265 Data truncated for column 'Show_Description' at row 8
Warning: #1265 Data truncated for column 'Show_Description' at row 9
Warning: #1265 Data truncated for column 'Show_Description' at row 10
```

Figure 35. Inserting into MovieShow table

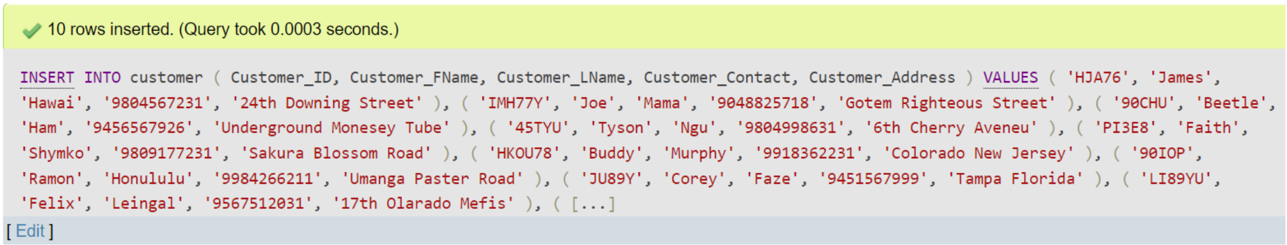


←T→				Show_ID	Show_Title	Show_Stars	Show_Description	Customer_ID
☐	Edit	Copy	Delete	4A327	Anabelle	Thre	John and Mia Form are attacked by members of a sat...	NULL
☐	Edit	Copy	Delete	98YUP	Sonic the Hedgehog 2	Two	After settling in Green Hills, Sonic is eager to p...	NULL
☐	Edit	Copy	Delete	AWE0ME	The Witcher: Nightmare of the Wolf	Five	Vesemir, a cocky young witcher, delights in slayin...	NULL
☐	Edit	Copy	Delete	FIN1L	Uncharted	Four	Victor Sullivan recruits Nathan Drake to help him ...	NULL
☐	Edit	Copy	Delete	H7TF5	The Grey Man	One	When the CIAs top asset -- his identity known to n...	NULL
☐	Edit	Copy	Delete	J7TY6	Red Notice	Four	In the world of international crime, an Interpol a...	NULL
☐	Edit	Copy	Delete	J817G	Day Shift	Two	A hardworking dad out to provide for his daughter ...	NULL
☐	Edit	Copy	Delete	KK98G	Venom	Five	After a faulty interview with the Life Foundation ...	NULL
☐	Edit	Copy	Delete	OK9A7	The Batman	Five	Batman is called to intervene when the mayor of Go...	NULL
☐	Edit	Copy	Delete	PC78T	Secret Headquarters	Two	While hanging out after school, Charlie and his fr...	NULL

Figure 36. Displaying Movie List Table

V. Customer:

```
INSERT INTO customer (
   Customer_ID,
   Customer_FName,
   Customer_LName,
   Customer_Contact,
   Customer_Address
)
VALUES (
   'HJA76',
   'James',
   'Hawai',
   '9804567231',
   '24th Downing Street'
),
(
   'IMH77Y',
   'Joe',
   'Mama',
   '9048825718',
   'Gotem Righteous Street'
),
(
   '90CHU',
   'Beetle',
   'Ham',
   '9456567926',
   'Underground Monesey Tube'
),
(
   '45TYU',
   'Tyson',
   'Ngu',
   '9804998631',
   '6th Cherry Aveneu'
),
(
   'PI3E8',
   'Faith',
   'Shymko',
   '9809177231',
   'Sakura Blossom Road'
),
(
   'HKOU78',
   'Buddy',
   'Murphy',
   '9918362231',
   'Colorado New Jersey'
),
(
   '90IOP',
   'Ramon',
   'Honululu',
   '9984266211',
   'Umanga Paster Road'
),
(
   'JU89Y',
   'Corey',
   'Faze',
   '9451567999',
   'Tampa Florida'
),
(
   'LI89YU',
   'Felix',
   'Leingal',
   '9567512031',
   '17th Olarado Mefis'
),
(
   'H45G6',
   'JJ',
   'Olutunje',
   '9809627231',
   'Howerd Stern Road'
);
```

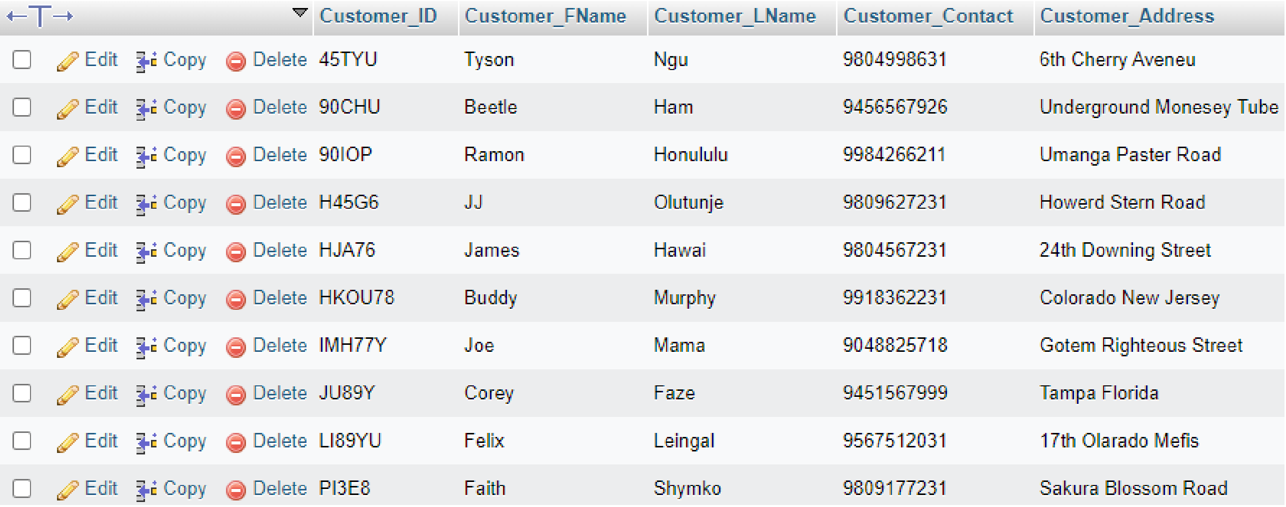


✓ 10 rows inserted. (Query took 0.0003 seconds.)

```
INSERT INTO customer ( Customer_ID, Customer_FName, Customer_LName, Customer_Contact, Customer_Address ) VALUES ( 'HJA76', 'James',
'Hawai', '9804567231', '24th Downing Street' ), ( 'IMH77Y', 'Joe', 'Mama', '9048825718', 'Gotem Righteous Street' ), ( '90CHU', 'Beetle',
'Ham', '9456567926', 'Underground Monesey Tube' ), ( '45TYU', 'Tyson', 'Ngu', '9804998631', '6th Cherry Aveneu' ), ( 'PI3E8', 'Faith',
'Shymko', '9809177231', 'Sakura Blossom Road' ), ( 'HKOU78', 'Buddy', 'Murphy', '9918362231', 'Colorado New Jersey' ), ( '90IOP',
'Ramon', 'Honululu', '9984266211', 'Umanga Paster Road' ), ( 'JU89Y', 'Corey', 'Faze', '9451567999', 'Tampa Florida' ), ( 'LI89YU',
'Felix', 'Leingal', '9567512031', '17th Olarado Mefis' ), ( [...]
```

[Edit]

Figure 37. Inserting into Customer table

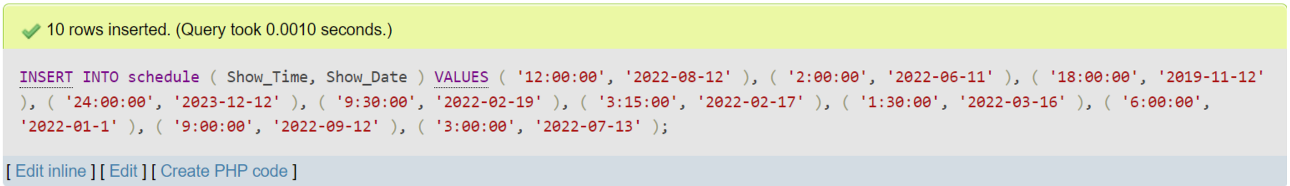


←T→				Customer_ID	Customer_FName	Customer_LName	Customer_Contact	Customer_Address
□	Edit	Copy	Delete	45TYU	Tyson	Ngu	9804998631	6th Cherry Aveneu
□	Edit	Copy	Delete	90CHU	Beetle	Ham	9456567926	Underground Monesey Tube
□	Edit	Copy	Delete	90IOP	Ramon	Honululu	9984266211	Umanga Paster Road
□	Edit	Copy	Delete	H45G6	JJ	Olutunje	9809627231	Howerd Stern Road
□	Edit	Copy	Delete	HJA76	James	Hawai	9804567231	24th Downing Street
□	Edit	Copy	Delete	HKOU78	Buddy	Murphy	9918362231	Colorado New Jersey
□	Edit	Copy	Delete	IMH77Y	Joe	Mama	9048825718	Gotem Righteous Street
□	Edit	Copy	Delete	JU89Y	Corey	Faze	9451567999	Tampa Florida
□	Edit	Copy	Delete	LI89YU	Felix	Leingal	9567512031	17th Olarado Mefis
□	Edit	Copy	Delete	PI3E8	Faith	Shymko	9809177231	Sakura Blossom Road

Figure 38. Displaying Customer Table

VI. Schedule:

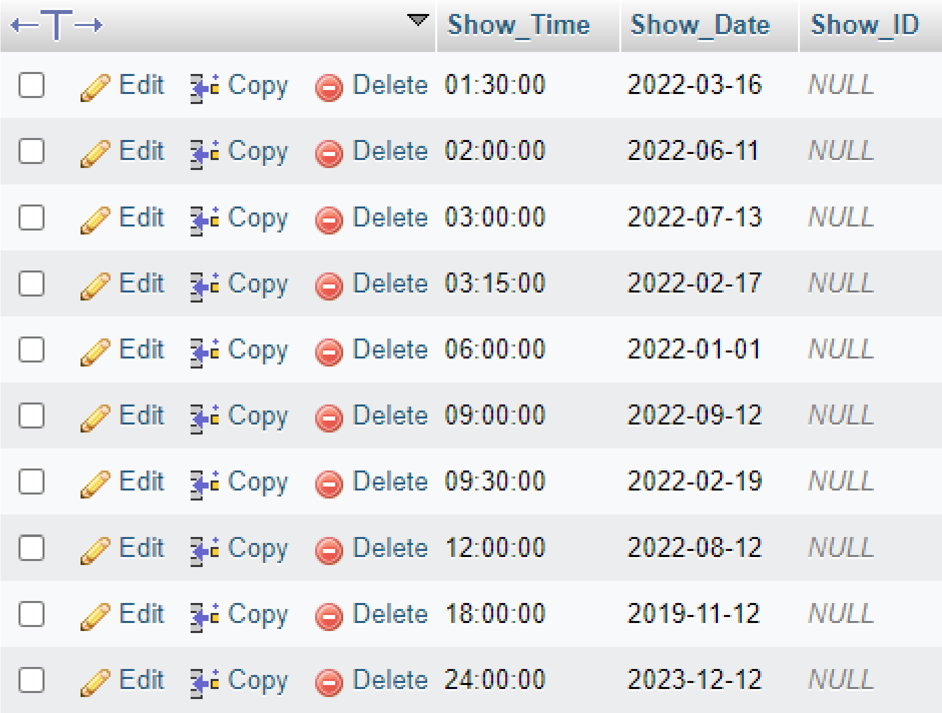
```
INSERT INTO schedule (
    Show_Time,
    Show_Date
)
VALUES (
    '12:00:00',
    '2022-08-12'
),
(
    '2:00:00',
    '2022-06-11'
),
(
    '18:00:00',
    '2019-11-12'
),
(
    '24:00:00',
    '2023-12-12'
),
(
    '9:30:00',
    '2022-02-19'
),
(
    '3:15:00',
    '2022-02-17'
),
(
    '1:30:00',
    '2022-03-16'
),
(
    '6:00:00',
    '2022-01-1'
),
(
    '9:00:00',
    '2022-09-12'
),
(
    '3:00:00',
    '2022-07-13'
);
```



✓ 10 rows inserted. (Query took 0.0010 seconds.)

```
INSERT INTO schedule ( Show_Time, Show_Date ) VALUES ( '12:00:00', '2022-08-12' ), ( '2:00:00', '2022-06-11' ), ( '18:00:00', '2019-11-12'
), ( '24:00:00', '2023-12-12' ), ( '9:30:00', '2022-02-19' ), ( '3:15:00', '2022-02-17' ), ( '1:30:00', '2022-03-16' ), ( '6:00:00',
'2022-01-1' ), ( '9:00:00', '2022-09-12' ), ( '3:00:00', '2022-07-13' );
```

Figure 39. Inserting into Schedule table

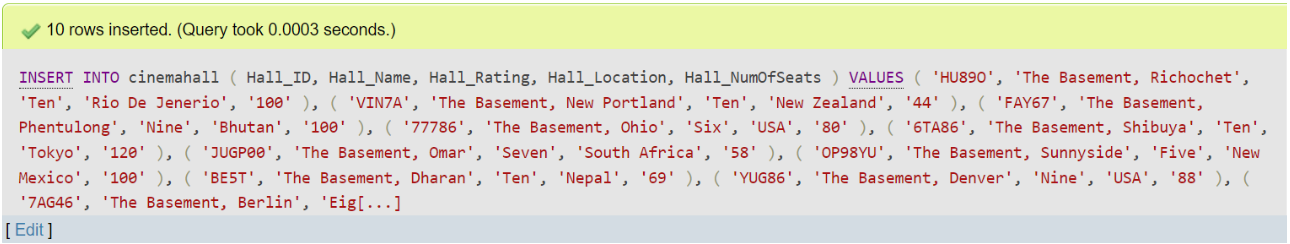


←T→	Show_Time	Show_Date	Show_ID
□ Edit Copy Delete	01:30:00	2022-03-16	NULL
□ Edit Copy Delete	02:00:00	2022-06-11	NULL
□ Edit Copy Delete	03:00:00	2022-07-13	NULL
□ Edit Copy Delete	03:15:00	2022-02-17	NULL
□ Edit Copy Delete	06:00:00	2022-01-01	NULL
□ Edit Copy Delete	09:00:00	2022-09-12	NULL
□ Edit Copy Delete	09:30:00	2022-02-19	NULL
□ Edit Copy Delete	12:00:00	2022-08-12	NULL
□ Edit Copy Delete	18:00:00	2019-11-12	NULL
□ Edit Copy Delete	24:00:00	2023-12-12	NULL

Figure 40. Displaying Schedule Table

VII. Cinema Hall:

```
INSERT INTO cinemahall (
    Hall_ID,
    Hall_Name,
    Hall_Rating,
    Hall_Location,
    Hall_NumOfSeats
)
VALUES (
    'HU89O',
    'The Basement, Richochet',
    'Ten',
    'Rio De Jenerio',
    '100'
),
(
    'VIN7A',
    'The Basement, New Portland',
    'Ten',
    'New Zealand',
    '44'
),
(
    'FAY67',
    'The Basement, Phentulong',
    'Nine',
    'Bhutan',
    '100'
),
(
    '77786',
    'The Basement, Ohio',
    'Six',
    'USA',
    '80'
),
(
    '6TA86',
    'The Basement, Shibuya',
    'Ten',
    'Tokyo',
    '120'
),
(
    'JUGP00',
    'The Basement, Omar',
    'Seven',
    'South Africa',
    '58'
),
(
    'OP98YU',
    'The Basement, Sunnyside',
    'Five',
    'New Mexico',
    '100'
),
(
    'BE5T',
    'The Basement, Dharan',
    'Ten',
    'Nepal',
    '69'
),
(
    'YUG86',
    'The Basement, Denver',
    'Nine',
    'USA',
    '88'
),
(
    '7AG46',
    'The Basement, Berlin',
    'Eight',
    'Germany',
    '100'
);
```

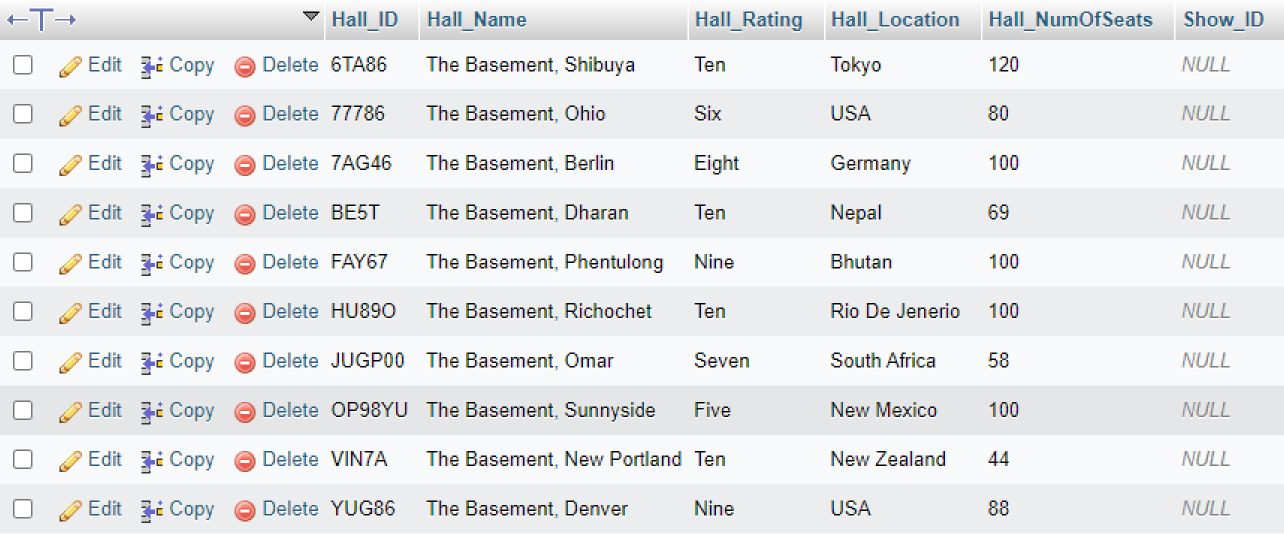


✓ 10 rows inserted. (Query took 0.0003 seconds.)

```
INSERT INTO cinemahall ( Hall_ID, Hall_Name, Hall_Rating, Hall_Location, Hall_NumOfSeats ) VALUES ( 'HU89O', 'The Basement, Richochet',
'Ten', 'Rio De Jenerio', '100' ), ( 'VIN7A', 'The Basement, New Portland', 'Ten', 'New Zealand', '44' ), ( 'FAY67', 'The Basement,
Phentulong', 'Nine', 'Bhutan', '100' ), ( '77786', 'The Basement, Ohio', 'Six', 'USA', '80' ), ( '6TA86', 'The Basement, Shibuya', 'Ten',
'Tokyo', '120' ), ( 'JUGP00', 'The Basement, Omar', 'Seven', 'South Africa', '58' ), ( 'OP98YU', 'The Basement, Sunnyside', 'Five', 'New
Mexico', '100' ), ( 'BE5T', 'The Basement, Dharan', 'Ten', 'Nepal', '69' ), ( 'YUG86', 'The Basement, Denver', 'Nine', 'USA', '88' ), (
'7AG46', 'The Basement, Berlin', 'Eig[...]
```

[Edit]

Figure 41. Inserting into Cinema Hall table

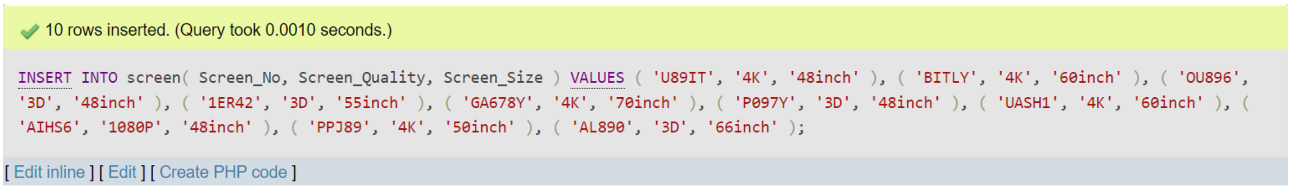


←T→				Hall_ID	Hall_Name	Hall_Rating	Hall_Location	Hall_NumOfSeats	Show_ID
□	Edit	Copy	Delete	6TA86	The Basement, Shibuya	Ten	Tokyo	120	*NULL*
□	Edit	Copy	Delete	77786	The Basement, Ohio	Six	USA	80	*NULL*
□	Edit	Copy	Delete	7AG46	The Basement, Berlin	Eight	Germany	100	*NULL*
□	Edit	Copy	Delete	BE5T	The Basement, Dharan	Ten	Nepal	69	*NULL*
□	Edit	Copy	Delete	FAY67	The Basement, Phentulong	Nine	Bhutan	100	*NULL*
□	Edit	Copy	Delete	HU89O	The Basement, Richochet	Ten	Rio De Jenerio	100	*NULL*
□	Edit	Copy	Delete	JUGP00	The Basement, Omar	Seven	South Africa	58	*NULL*
□	Edit	Copy	Delete	OP98YU	The Basement, Sunnyside	Five	New Mexico	100	*NULL*
□	Edit	Copy	Delete	VIN7A	The Basement, New Portland	Ten	New Zealand	44	*NULL*
□	Edit	Copy	Delete	YUG86	The Basement, Denver	Nine	USA	88	*NULL*

Figure 42. Displaying Cinema Hall Table

VIII. Screen:

```
INSERT INTO screen (
    Screen_No,
    Screen_Quality,
    Screen_Size
)
VALUES (
    'U89IT',
    '4K',
    '48inch'
),
(
    'BITLY',
    '4K',
    '60inch'
),
(
    'OU896',
    '3D',
    '48inch'
),
(
    '1ER42',
    '3D',
    '55inch'
),
(
    'GA678Y',
    '4K',
    '70inch'
),
(
    'P097Y',
    '3D',
    '48inch'
),
(
    'UASH1',
    '4K',
    '60inch'
),
(
    'AIHS6',
    '1080P',
    '48inch'
),
(
    'PPJ89',
    '4K',
    '50inch'
),
(
    'AL890',
    '3D',
    '66inch'
);
```

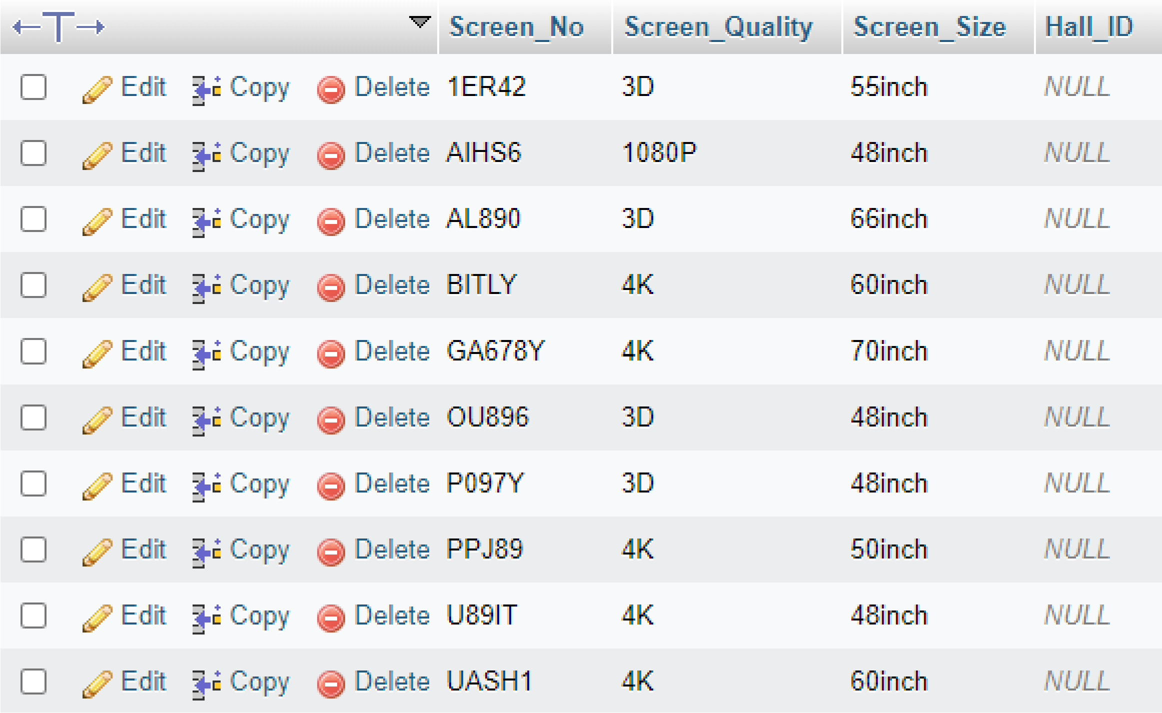


✓ 10 rows inserted. (Query took 0.0010 seconds.)

```
INSERT INTO screen( Screen_No, Screen_Quality, Screen_Size ) VALUES ( 'U89IT', '4K', '48inch' ), ( 'BITLY', '4K', '60inch' ), ( 'OU896',
'3D', '48inch' ), ( '1ER42', '3D', '55inch' ), ( 'GA678Y', '4K', '70inch' ), ( 'P097Y', '3D', '48inch' ), ( 'UASH1', '4K', '60inch' ), (
'AIHS6', '1080P', '48inch' ), ( 'PPJ89', '4K', '50inch' ), ( 'AL890', '3D', '66inch' );
```

[Edit inline] [Edit] [Create PHP code]

Figure 43. Inserting into screen table



←T→ ▼	Screen_No	Screen_Quality	Screen_Size	Hall_ID
□ Edit Copy Delete	1ER42	3D	55inch	NULL
□ Edit Copy Delete	AIHS6	1080P	48inch	NULL
□ Edit Copy Delete	AL890	3D	66inch	NULL
□ Edit Copy Delete	BITLY	4K	60inch	NULL
□ Edit Copy Delete	GA678Y	4K	70inch	NULL
□ Edit Copy Delete	OU896	3D	48inch	NULL
□ Edit Copy Delete	P097Y	3D	48inch	NULL
□ Edit Copy Delete	PPJ89	4K	50inch	NULL
□ Edit Copy Delete	U89IT	4K	48inch	NULL
□ Edit Copy Delete	UASH1	4K	60inch	NULL

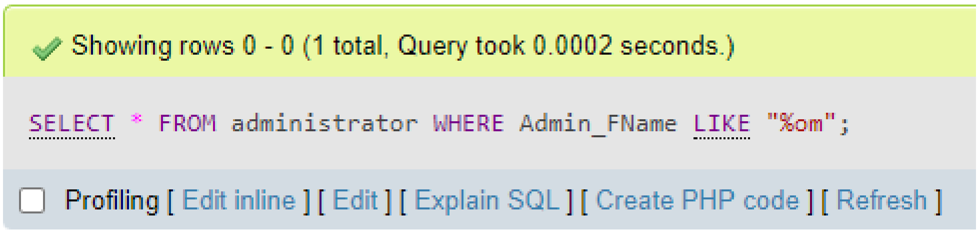
Figure 44. Displaying Screen Table

4. QUERIES:

4.1. Execution:

4.1.1. List Fname using LIKE:

SELECT * FROM administrator WHERE Admin_FName LIKE "%om";

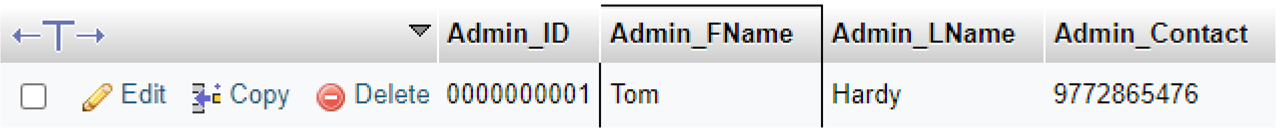


✓ Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)

```
SELECT * FROM administrator WHERE Admin_FName LIKE "%om";
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 45. Like Query



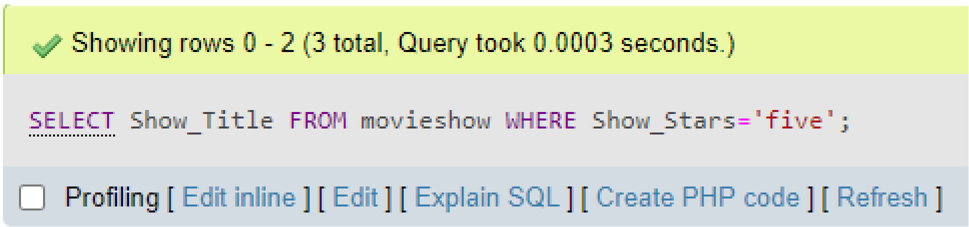
←T→	Admin_ID	Admin_FName	Admin_LName	Admin_Contact
□ Edit Copy Delete	0000000001	Tom	Hardy	9772865476

Figure 46. Like Output

In this query we selected all rows admin table whose first names has characters before ‘om’

4.1.2. List all movie shows with rating of five:

SELECT Show_Title FROM movieshow WHERE Show_Stars='five';

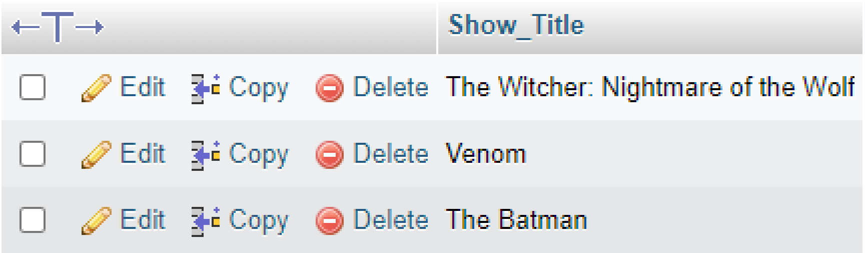


✓ Showing rows 0 - 2 (3 total, Query took 0.0003 seconds.)

```
SELECT Show_Title FROM movieshow WHERE Show_Stars='five';
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 47. Show_Title Query



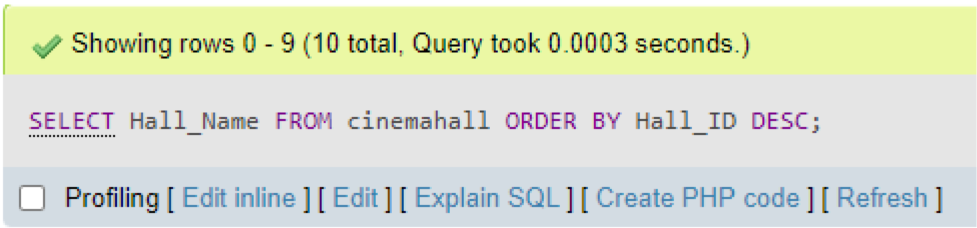
←T→				Show_Title
□	Edit	Copy	Delete	The Witcher: Nightmare of the Wolf
□	Edit	Copy	Delete	Venom
□	Edit	Copy	Delete	The Batman

Figure 48. Show_Title Output

In this query, all the show titles have been displayed which were given star rating of five.

4.1.3. Listing all Cinema Halls by order of their descending Hall ID:

SELECT Hall_Name FROM cinemahall ORDER BY Hall_ID DESC;

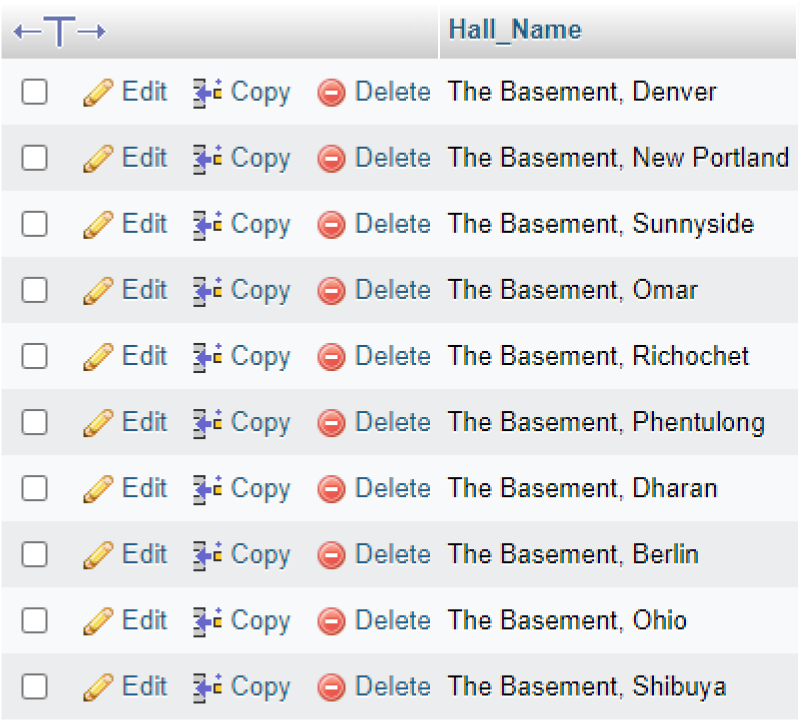


✓ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT Hall_Name FROM cinemahall ORDER BY Hall_ID DESC;
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 49. Order Query



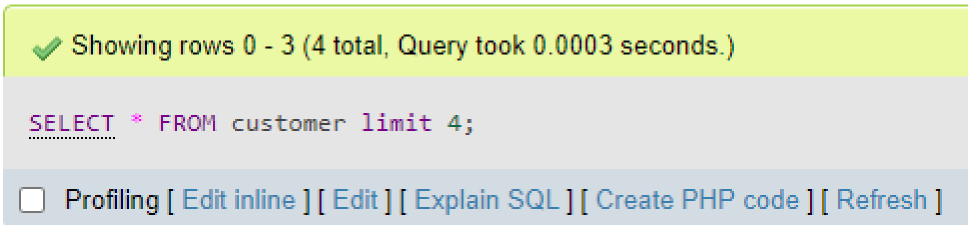
←T→				Hall_Name
☐	Edit	Copy	Delete	The Basement, Denver
☐	Edit	Copy	Delete	The Basement, New Portland
☐	Edit	Copy	Delete	The Basement, Sunnyside
☐	Edit	Copy	Delete	The Basement, Omar
☐	Edit	Copy	Delete	The Basement, Richochet
☐	Edit	Copy	Delete	The Basement, Phentulong
☐	Edit	Copy	Delete	The Basement, Dharan
☐	Edit	Copy	Delete	The Basement, Berlin
☐	Edit	Copy	Delete	The Basement, Ohio
☐	Edit	Copy	Delete	The Basement, Shibuya

Figure 50. Order Output

In this query, all the names of the hall have been ordered in a descending order by their Hall ID.

4.1.4. Limiting the values displayed of the Customer Table:

SELECT * FROM customer limit 4;

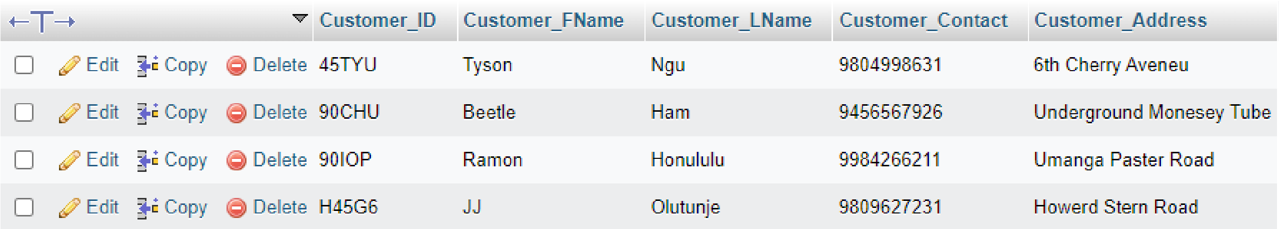


✓ Showing rows 0 - 3 (4 total, Query took 0.0003 seconds.)

```
SELECT * FROM customer limit 4;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 51. Limit Query



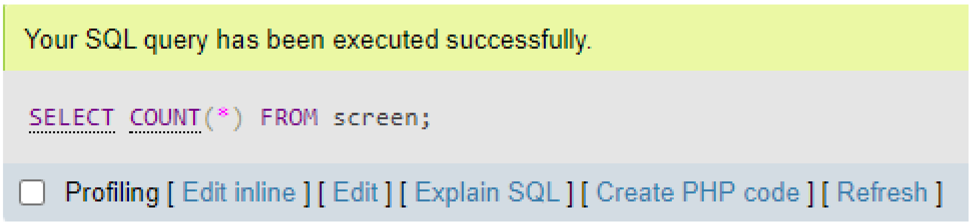
←T→	Customer_ID	Customer_FName	Customer_LName	Customer_Contact	Customer_Address
□ Edit Copy Delete	45TYU	Tyson	Ngu	9804998631	6th Cherry Aveneu
□ Edit Copy Delete	90CHU	Beetle	Ham	9456567926	Underground Monesey Tube
□ Edit Copy Delete	90IOP	Ramon	Honululu	9984266211	Umanga Paster Road
□ Edit Copy Delete	H45G6	JJ	Olutunje	9809627231	Howerd Stern Road

Figure 52. Limit Output

In this query, all the information of the customer table is displayed having maximum of 4 rows.

4.1.5. Counting records of Screen Type:

SELECT COUNT(*) FROM screen;

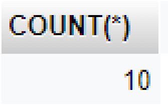


Your SQL query has been executed successfully.

```
SELECT COUNT(*) FROM screen;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 53. Count Query



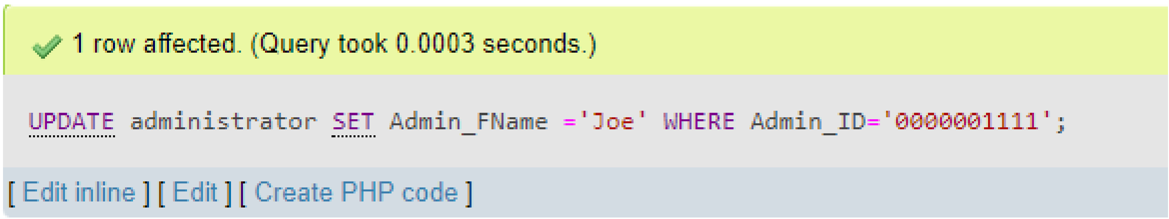
COUNT(*)
10

Figure 54. Count Output

In this query the count is used to return total number of records in rows in the table.

4.1.6. Update the data from the Administrator table:

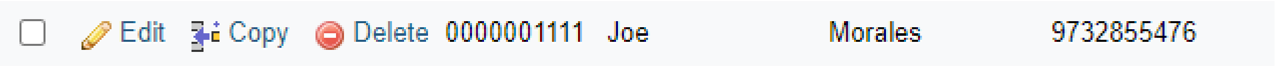
UPDATE administrator SET Admin_FName ='Joe' WHERE Admin_ID='0000001111';



✓ 1 row affected. (Query took 0.0003 seconds.)

```
UPDATE administrator SET Admin_FName ='Joe' WHERE Admin_ID='0000001111';
```

Figure 55. Update Query



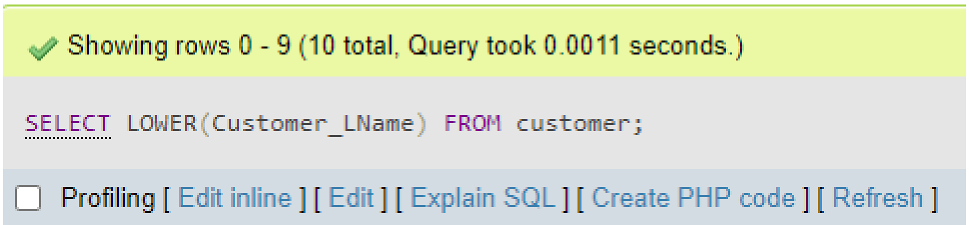
□	Edit	Copy	Delete	0000001111	Joe	Morales	9732855476

Figure 56. Update Output

In this query, the name of the administrator with the given admin_ID was changed to ‘Joe’

4.1.7. Get all Customers last name in lowercase:

SELECT LOWER(Customer_LName) FROM customer;

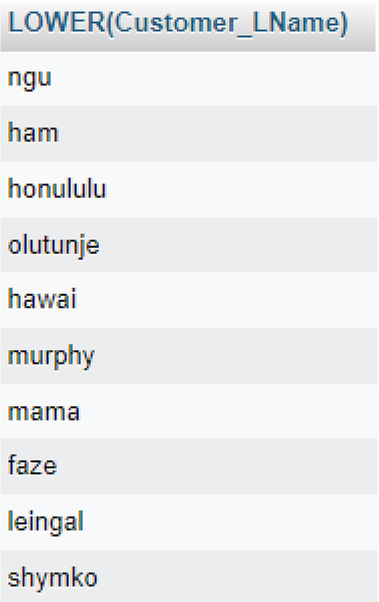


✓ Showing rows 0 - 9 (10 total, Query took 0.0011 seconds.)

```
SELECT LOWER(Customer_LName) FROM customer;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 57. Lower Query



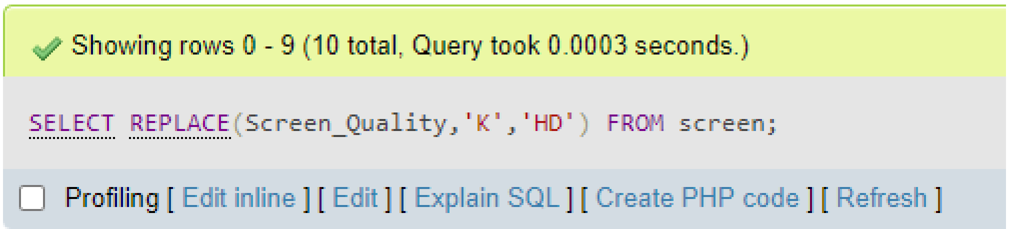
LOWER(Customer_LName)
ngu
ham
honululu
olutunje
hawai
murphy
mama
faze
leingal
shymko

Figure 58. Lower Output

In this query, all the last names of the customers have been displayed in lowercase.

4.1.8. Get the Screen Quality after replacing ‘K’ with ‘HD’:

SELECT REPLACE(Screen_Quality,'K','HD') FROM screen;

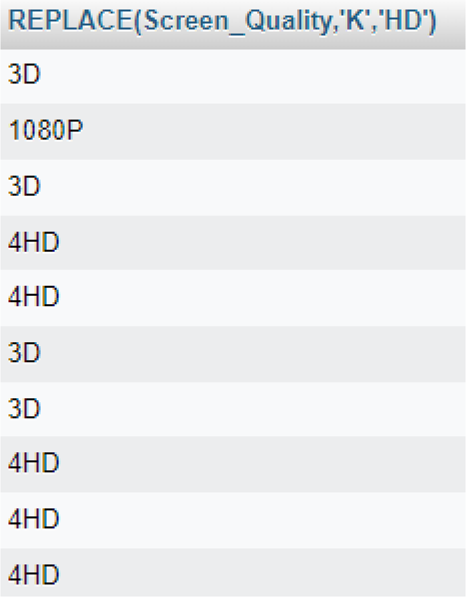


✓ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT REPLACE(Screen_Quality,'K','HD') FROM screen;
```

☐ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 59. Replace Query



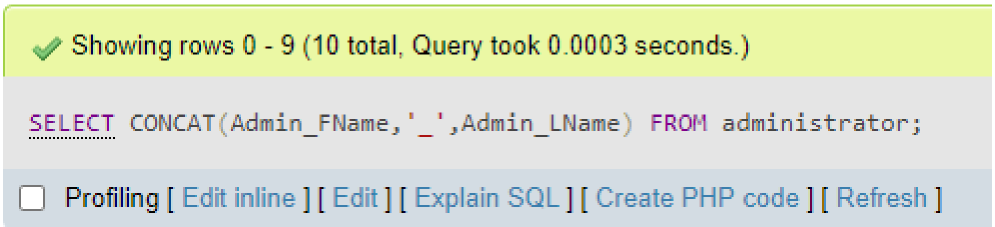
REPLACE(Screen_Quality,'K','HD')
3D
1080P
3D
4HD
4HD
3D
3D
4HD
4HD
4HD

Figure 60. Replace Output

In this query, we have replaced all the items having the letter ‘K’ with ‘HD’ in the Screen_Quality attribute.

4.1.9. Get the admin’s last name and first name in a single column separated by a ‘_’.

SELECT CONCAT(Admin_FName,'_',Admin_LName) FROM administrator;

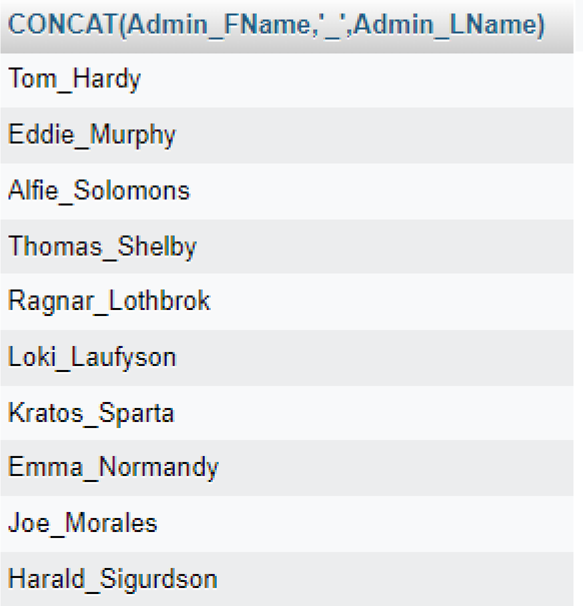


✓ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT CONCAT(Admin_FName,'_',Admin_LName) FROM administrator;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 61. Concat Query



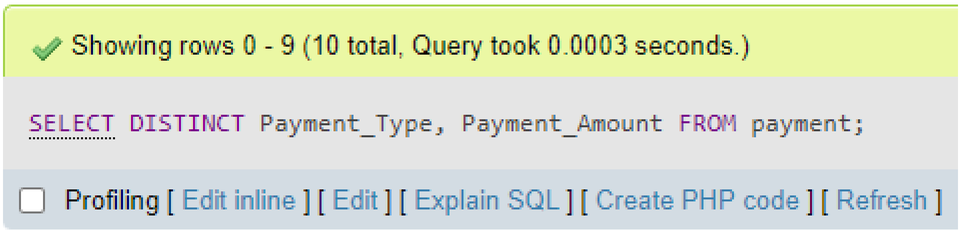
CONCAT(Admin_FName,'_',Admin_LName)
Tom_Hardy
Eddie_Murphy
Alfie_Solomons
Thomas_Shelby
Ragnar_Lothbrok
Loki_Laufyson
Kratos_Sparta
Emma_Normandy
Joe_Morales
Harald_Sigurdson

Figure 62. Concat Output

In this query we join the first name and the last name of the admin with a underscore in the middle.

4.1.10. Selecting rows using Distinct:

SELECT DISTINCT Payment_Type, Payment_Amount FROM payment;

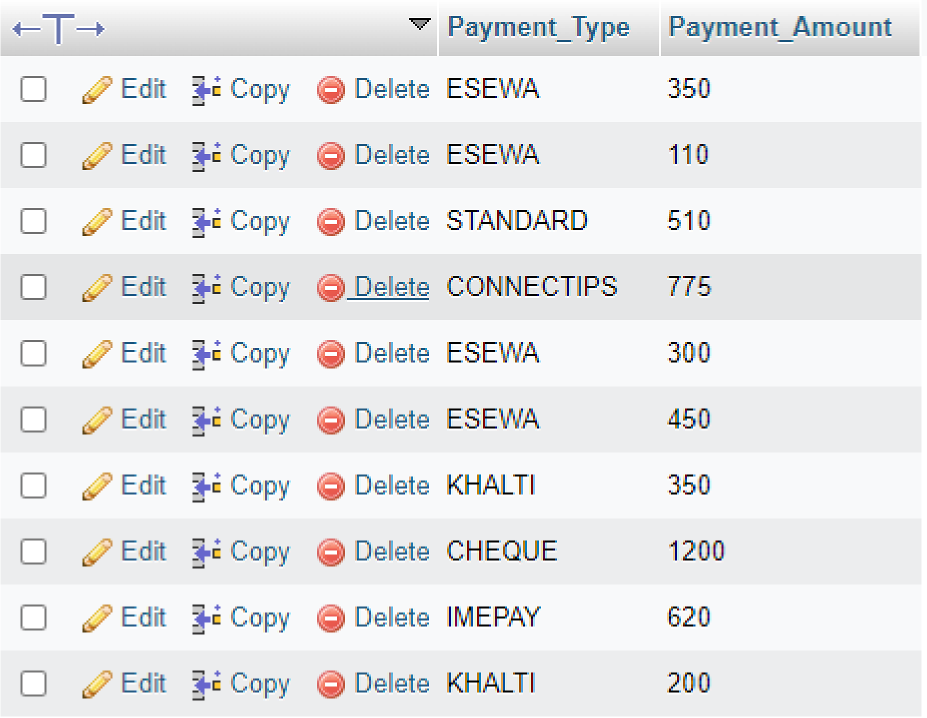


✓ Showing rows 0 - 9 (10 total, Query took 0.0003 seconds.)

```
SELECT DISTINCT Payment_Type, Payment_Amount FROM payment;
```

□ Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Figure 63. Distinct Query

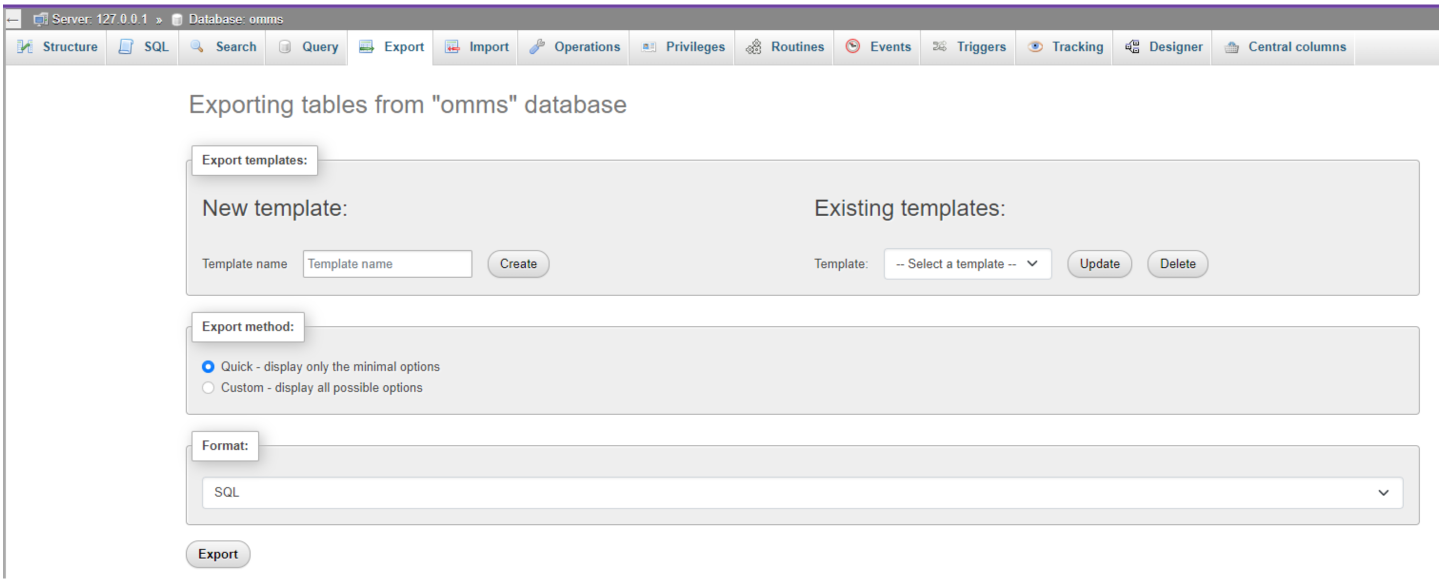


←T→	Payment_Type	Payment_Amount
□ Edit Copy Delete	ESEWA	350
□ Edit Copy Delete	ESEWA	110
□ Edit Copy Delete	STANDARD	510
□ Edit Copy Delete	CONNECTIPS	775
□ Edit Copy Delete	ESEWA	300
□ Edit Copy Delete	ESEWA	450
□ Edit Copy Delete	KHALTI	350
□ Edit Copy Delete	CHEQUE	1200
□ Edit Copy Delete	IMEPAY	620
□ Edit Copy Delete	KHALTI	200

Figure 64. Distinct Output

In this query, we have selected Distinct payment type and payment account.

4.2. Dump File:



Exporting tables from "omms" database

Export templates:

New template:

Template name: Template name [Create]

Existing templates:

Template: -- Select a template -- [Update] [Delete]

Export method:

● Quick - display only the minimal options
○ Custom - display all possible options

Format:

SQL

[Export]

Figure 65. Exporting

Google Drive Link for the file:
https://drive.google.com/drive/folders/1rQCDvEJ1JoytUk7q9onJqU79p9nzcMh5

4.3. Drop Tables:

I. Administrator:

DROP TABLE administrator;

II. Payment:

DROP TABLE payment;

III. BookingWebsite:

DROP TABLE bookingwebsite;

IV. MovieShow:

DROP TABLE movieshow;

V. Customer:

DROP TABLE customer;

VI. Schedule:

DROP TABLE schedule;

VII. CinemaHall:

DROP TABLE cinemahall;

VIII. Screen:

DROP TABLE Screen;

5. CONCLUSION:

5.1. Learning Experience:

I struggled a lot to understand the case scenarios given in the Google classroom when this assignment was first offered to us. The question was quite difficult for me to understand because I was unfamiliar with these Normalization techniques. Analysis and identification of entities and attributes was among the difficulties I encountered while working through different stages of my homework. It was difficult for a novice like me to complete the coursework because we had to standardize the data from UNF to 3NF. I made an effort to read, comprehend, and identify potential entities and attributes by reading the given scenario and the queries. My entities and characteristics were finally finished and normalized to 3NF after many efforts, research, and advice from my teachers. I also conducted some online research and looked through the slides that our lecturers and tutors had sent to me for this project. I generated tables after normalizing the data and added values in accordance with the specified queries. After the development of the tables and the insertion of the data, we had to query the tables, which was a little challenging for me. In order to address my questions, I therefore revised my class notes and conducted some research. With the help of reputable instructors, my issues were quickly resolved. My understanding of databases and SQL queries has increased as a result of this education. I learned about scenario analysis, defining entities and attributes, making ER diagrams, creating SQL tables, adding data to the tables, and practicing SQL queries. This training provided me with knowledge of the procedures, issues, and fixes encountered by various businesses when maintaining their database. Since I'm still learning, I encountered many challenging circumstances throughout this coursework, but after talking to my tutors and doing some research, these issues became insignificant. But without mistakes and challenges, learning would not be possible. As a result, this coursework has taught me how to deal with problems and find solutions. I had a fantastic and completely unique learning experience with this.

5.2. Critical Evaluation:

This assignment has given us a comprehensive look at how databases operate in all types of businesses, which has been a great learning opportunity. As a student, the "Databases" module is one of the most crucial ones because it teaches us about normalization, case study analysis, ER diagrams, and other concepts. I might soon need to manage enormous volumes of data because I'm a student of computing. I now have understanding of how data can be organized in a database and how to effectively store and retrieve data from a database thanks to this module. This lesson has assisted me in better understanding and using the concept of normalization, which allows for the efficient organization and maintenance of even large amounts of data in separate entities in order to reduce data anomalies, data redundancy, and speed up our processes. The "Databases" curriculum is connected to the "Software Engineering" module, which puts the theory of databases and ER diagrams into practice. I gained the ability to create ER diagrams from this module, which was incredibly useful for finishing the Software Engineering program's group assignments. I gained knowledge in this lesson about how to lessen data abnormalities and redundancy, which will benefit me in time-saving projects in the future. This project taught me a lot about transitive and partial dependencies, a crucial area for normalization. I've learned new skills from this lesson, and it has also made it easier for me to complete the assignments from other modules. I was able to easily discover and eliminate repetitive groups, partial dependencies, and transitive dependencies by studying this module. This is a really significant ability, and it will be very helpful for my profession in the future. From this program, I learned how to create, arrange, and maintain a database for any form of business.

5.3. Future Enhancements:

This was my first database designing job, it was close to the real deal itself. I tried my best to give it my all and, in the process, I think I have matured a lot and even grasped a lot. Having said that, I am still just a beginner and I am sure that if my work is looked over by my superiors, they will definitely find flaws but I am sure that I can take that as constructive criticism and work on myself and my skills to further improve myself. This database design contains my blood, sweat and tears for I have worked very hard for it. Though it still lacks more features, the UI/UX can be fine tuned and many other things I have planned for it will definitely come to pass for I am determined that I will work on this project till it becomes a worthy masterpiece, something that will help me take a step further in this competitive world of computers.