**Database – SQL**

**Relational Schema**

Database schema is another term for relational schema. A database schema is a set of relation schemas for a database as a whole. A database schema, often known as a relational schema, is a set of meta-data. The structure and limitations of data representations in a certain area are described by the database schema. A relational schema can be defined as a database blueprint that specifies how data is grouped into tables.

**Scenario**:

The MICE Cinema group is a small group of independent cinemas in Hertfordshire. They show what they regard as quality films and they usually start each showing of a film with a short talk about the film by the cinema manager.

There are a number of cinema venues in Hertfordshire. Each screen only shows one film an evening, although some of the venues have more than one screen with different capacities (number of seats) – therefore the venue can show more than one film in an evening. A specific film will be performed over a number of evenings on the same screen, the collective performances are known as showings.

For each performance of a film, the owners would like to know how much money was taken so that they can work out how much a film brings in overall for a showing, and collectively from the different cinemas in which it is performed, thus indicating which is the most popular film (and the least!).

Each cinema employs a group of staff with certain job types (Administrator, Receptionist,

Projectionist, Accountant, Cleaner etc.) Staff only work in one cinema. Some staff supervise other staff members. At its administrative center, not based at any cinema, MICE also has staff which help administer the group, such as Accountant, CEO, Central Advanced

Booking Sales Staff, etc.

**Dimensional Model:**

1. **Schema for Cinema:**

Cinema (Cinema ID, Cinema Name, Address, PhoneNo)

1. **Schema for Screen:**

Screen (Cinema ID\*, Screen ID, Capacity)

1. **Schema for Showing:**

Showing (Showing ID, Cinema ID\*, Screen ID\*, FilmID\*, Start Date, End Date)

1. **Schema for Film:**

Film (FilmID, Title, Director, ReleaseDate, RunningTime)

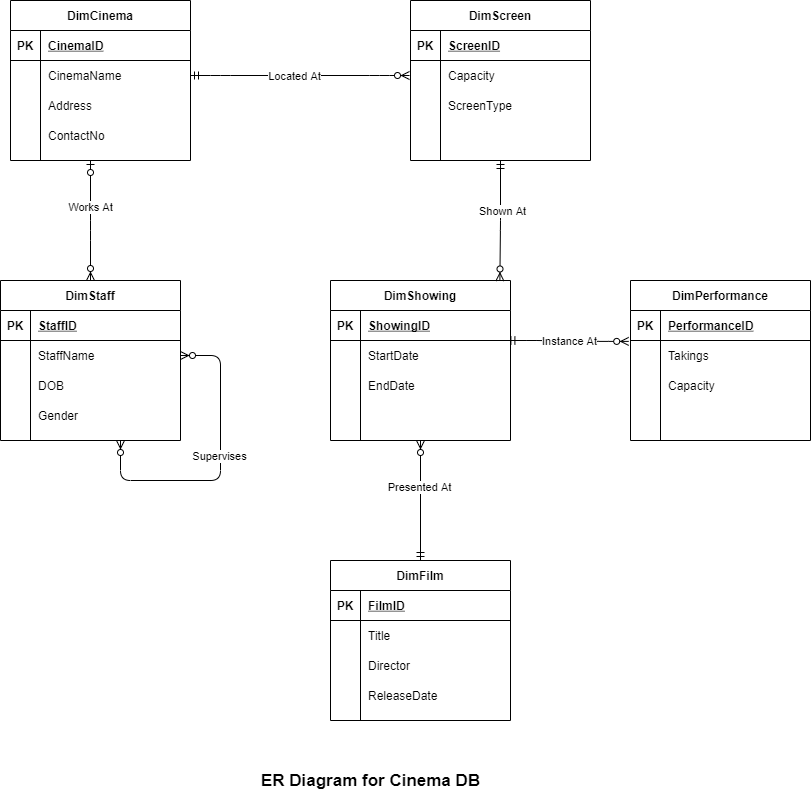
1. **Schema for Performance:**

Performance (Showing ID\*, PerformanceNo, Takings, AudienceSize)

1. **Schema for Staff:**

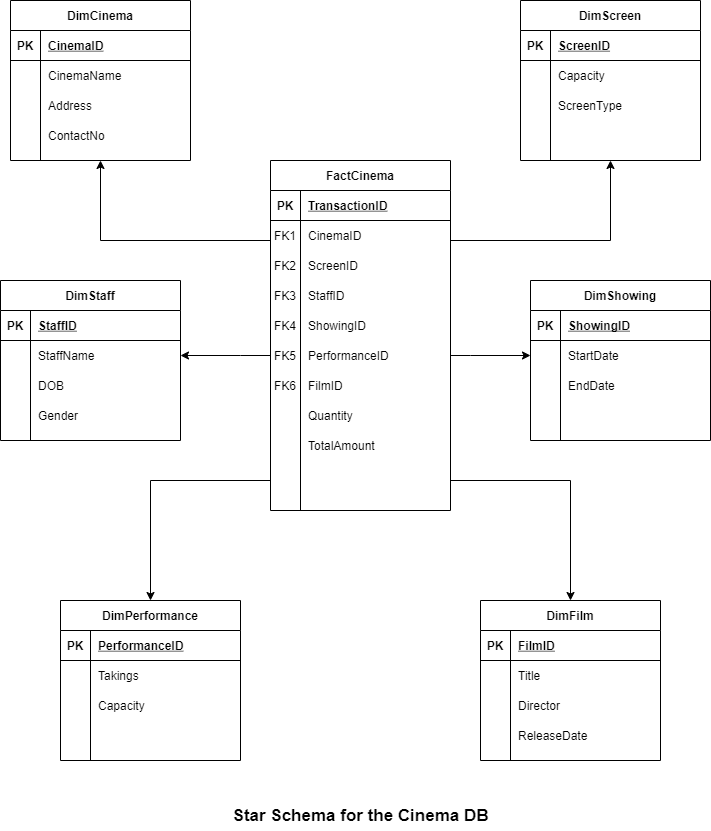
Staff (StaffID, FamilyName, GivenName, DoB, CinemaID\*, Supervisor\*)

**ER Diagram for the above Dimension Table:**



*Figure: ER Diagram for the above Dimension table*

**Star Schema for the above ER diagram:**



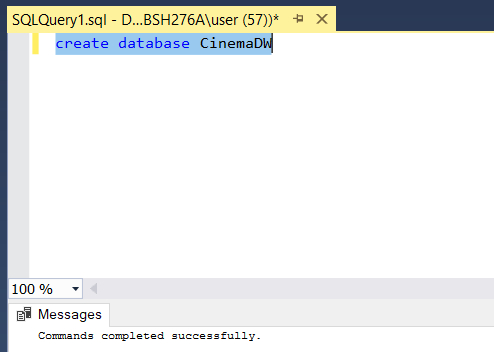
*Figure: Star Schema for the above mentioned ER Diagram*

**Creating the Table Structure as shown in the schema/diagram:**

Entity Relationship Model (ER Model) is a high-level data modelling model diagram. The ER model aids in the methodical analysis of data needs in order to create some well database. The ER Model is a representation of real-world things and their relationships.

Creating Database for Cinema:

create database CinemaDW



* Creating a Table for Cinema:

CODE:

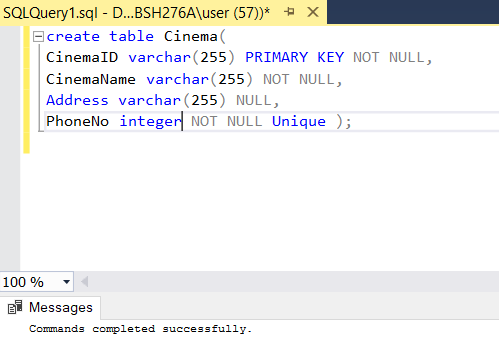
create table Cinema(

CinemaID varchar(255) PRIMARY KEY NOT NULL,

CinemaName varchar(255) NOT NULL,

Address varchar(255) NULL,

PhoneNo integer NOT NULL Unique );



*Figure: Table - Cinema*

ADDING VALUES TO THE TABLE CINEMA:

insert into Cinema(CinemaID, CinemaName, Address, PhoneNo)

values ('A23', 'Rock Cinema', 'Surrey', '89765676');

insert into Cinema(CinemaID, CinemaName, Address, PhoneNo)

values ('A289', 'Martin Cinema', 'London', '89734546');

insert into Cinema(CinemaID, CinemaName, Address, PhoneNo)

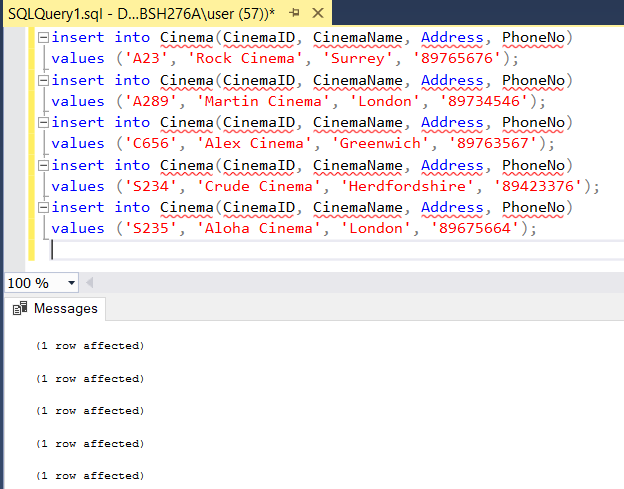
values ('C656', 'Alex Cinema', 'Greenwich', '89763567');

insert into Cinema(CinemaID, CinemaName, Address, PhoneNo)

values ('S234', 'Crude Cinema', 'Herdfordshire', '89423376');

insert into Cinema(CinemaID, CinemaName, Address, PhoneNo)

values ('S235', 'Aloha Cinema', 'London', '89675664');

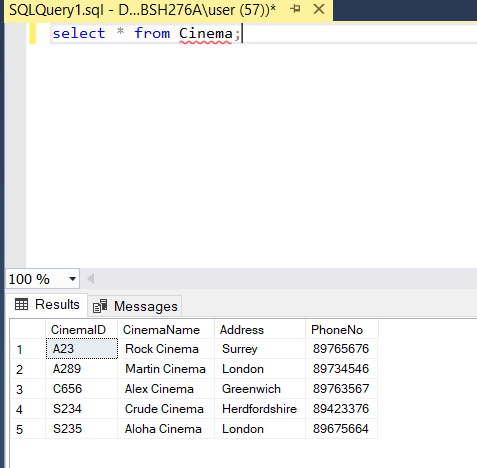


*Figure: Adding Values to the table Cinema*

**To view Table - Cinema:**

CODE:

select \* from Cinema;



*Figure: To View Table - Cinema*

* Creating a Table for Screen:

CODE:

create table Screen(

CinemaID varchar(255) NOT NULL,

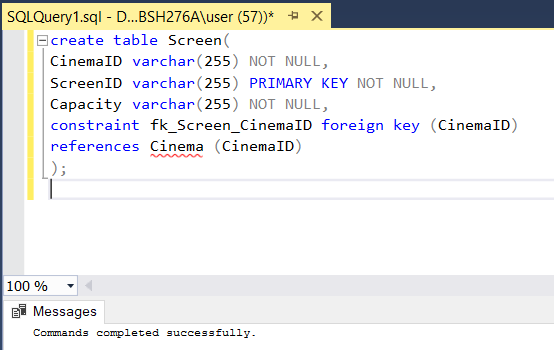
ScreenID varchar(255) PRIMARY KEY NOT NULL,

Capacity varchar(255) NOT NULL,

constraint fk\_Screen\_CinemaID foreign key (CinemaID)

references Cinema (CinemaID)

);



*Figure: Table - Screen*

ADDING VALUES TO THE TABLE SCREEN:

insert into Screen(CinemaID, ScreenID, Capacity)

values ('A23', 'A15', '200');

insert into Screen(CinemaID, ScreenID, Capacity)

values ('A289', 'B10', '220');

insert into Screen(CinemaID, ScreenID, Capacity)

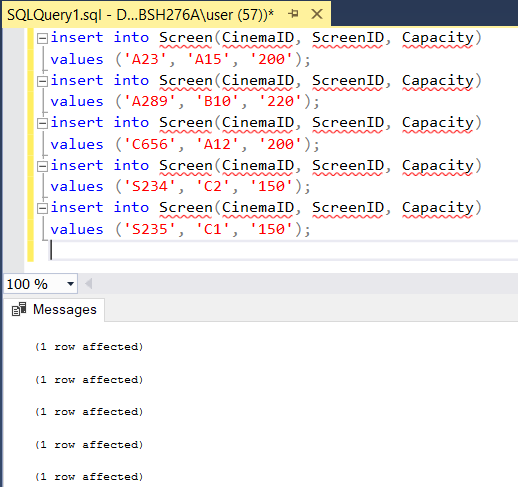
values ('C656', 'A12', '200');

insert into Screen(CinemaID, ScreenID, Capacity)

values ('S234', 'C2', '150');

insert into Screen(CinemaID, ScreenID, Capacity)

values ('S235', 'C1', '150');

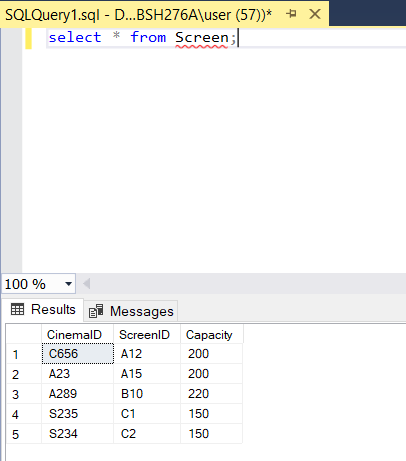


*Figure: Adding Value to the table - Screen*

To View Table Values:

Code:

select \* from Screen;



*Figure: To view Table Values of Screen*

* Creating a Table for Staff:

CODE:

create table Staff (

StaffID varchar(255) NOT NULL,

FamilyName varchar(255) NOT NULL,

GivenName varchar(255) NOT NULL,

DoB date NOT NULL,

CinemaID varchar(255) NOT NULL,

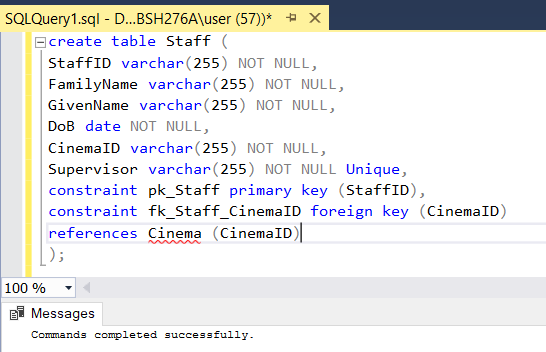
Supervisor varchar(255) NOT NULL Unique,

constraint pk\_Staff primary key (StaffID),

constraint fk\_Staff\_CinemaID foreign key (CinemaID)

references Cinema (CinemaID)

);



*Figure: Table - Staff*

Now using ALTER command to add Another Foreign Key - Supervisor to the table Staff

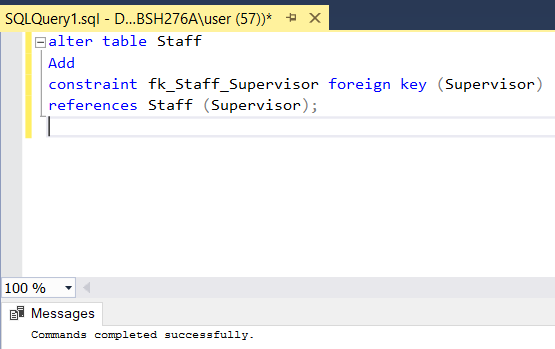
CODE:

alter table Staff

Add

constraint fk\_Staff\_Supervisor foreign key (Supervisor)

references Staff (Supervisor);



*Figure: Table - Staff (Alter)*

**ADDING VALUES TO TABLE STAFF:**

CODE:

insert into Staff(StaffID, FamilyName, GivenName, DoB, CinemaID, Supervisor)

values ('E23', 'Jack', 'James', '01-FEB-2012', 'A23', 'Alex');

insert into Staff(StaffID, FamilyName, GivenName, DoB, CinemaID, Supervisor)

values ('E53', 'Jane', 'Nord', '14-MAR-2012', 'A289', 'Aron');

insert into Staff(StaffID, FamilyName, GivenName, DoB, CinemaID, Supervisor)

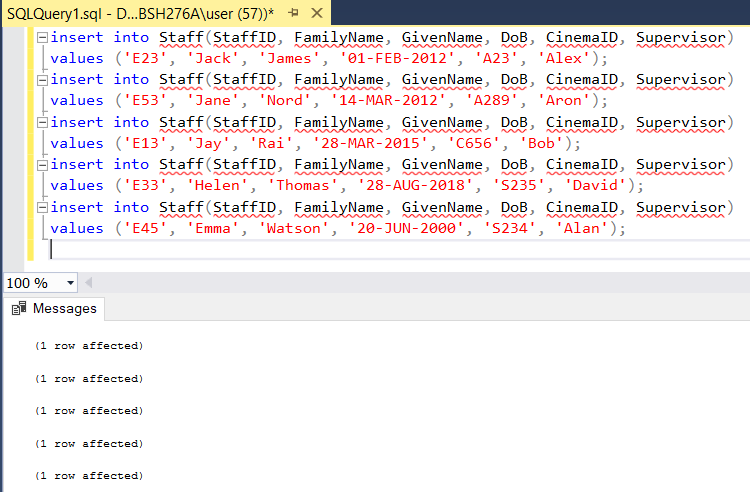
values ('E13', 'Jay', 'Rai', '28-MAR-2015', 'C656', 'Bob');

insert into Staff(StaffID, FamilyName, GivenName, DoB, CinemaID, Supervisor)

values ('E33', 'Helen', 'Thomas', '28-AUG-2018', 'S235', 'David');

insert into Staff(StaffID, FamilyName, GivenName, DoB, CinemaID, Supervisor)

values ('E45', 'Emma', 'Watson', '20-JUN-2000', 'S234', 'Alan');

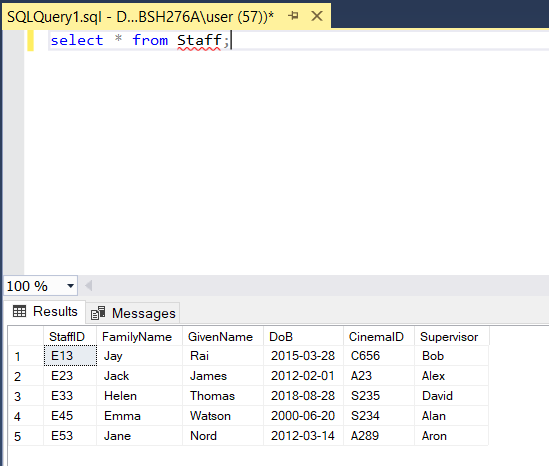


*Figure: Add values to the Staff Table*

To view Values of the Staff Table:

CODE:

select \* from Staff;



*Figure: To view Table values of Staff*

* **Creating a Table for Film:**

CODE:

create table Film(

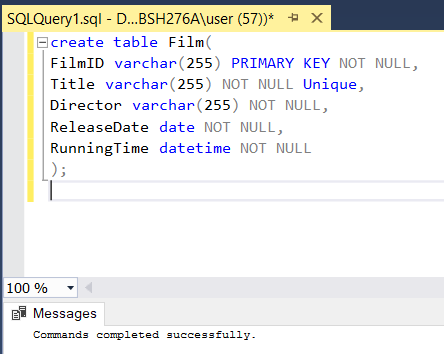
FilmID varchar(255) PRIMARY KEY NOT NULL,

Title varchar(255) NOT NULL Unique,

Director varchar(255) NOT NULL,

ReleaseDate date NOT NULL,

RunningTime datetime NOT NULL

); 

*Figure: Table - Film*

ADDING VALUES TO THE TABLE FILM:

CODE:

insert into Film(FilmID, Title, Director, ReleaseDate, RunningTime)

values ('V5646', 'Harry Potter & The Prisoner of Azkaban', 'J.K.Rowling', '01-FEB-2012', '01-FEB-2012 10:02:00');

insert into Film(FilmID, Title, Director, ReleaseDate, RunningTime)

values ('V7686', 'Spider Man', 'Sam Raimi', '14-MAR-2012', '14-MAR-2012 12:02:00');

insert into Film(FilmID, Title, Director, ReleaseDate, RunningTime)

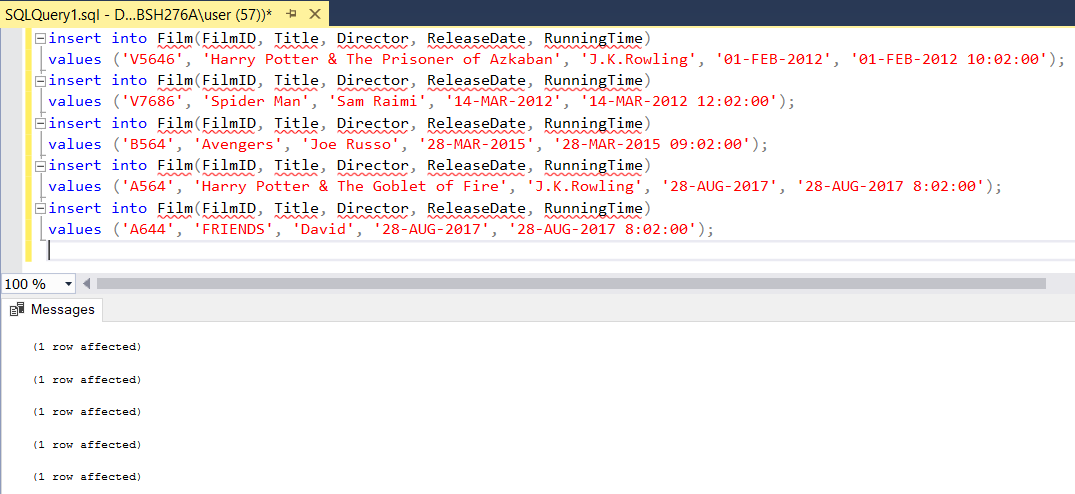
values ('B564', 'Avengers', 'Joe Russo', '28-MAR-2015', '28-MAR-2015 09:02:00');

insert into Film(FilmID, Title, Director, ReleaseDate, RunningTime)

values ('A564', 'Harry Potter & The Goblet of Fire', 'J.K.Rowling', '28-AUG-2017', '28-AUG-2017 8:02:00');

insert into Film(FilmID, Title, Director, ReleaseDate, RunningTime)

values ('A644', 'FRIENDS', 'David', '28-AUG-2017', '28-AUG-2017 8:02:00');

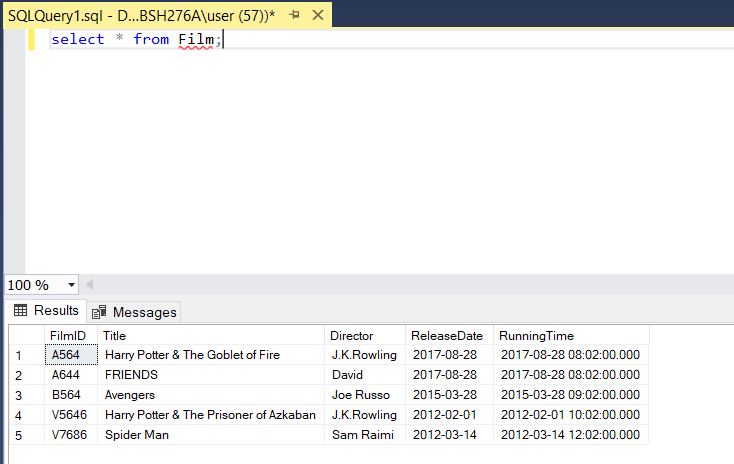


*Figure: Adding values to Table Film*

To View Table Values of Film:

CODE:

select \* from Film;



*Figure: To View Table Values of Film*

* **Creating a Table for Showing:**

CODE:

create table Showing(

ShowingID varchar(255) PRIMARY KEY NOT NULL,

CinemaID varchar(255) NOT NULL,

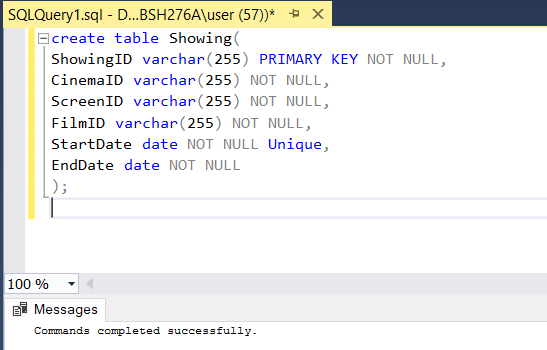
ScreenID varchar(255) NOT NULL,

FilmID varchar(255) NOT NULL,

StartDate date NOT NULL Unique,

EndDate date NOT NULL

);



*Figure: Table - Showing*

**ADDING VALUES TO SHOWING TABLE:**

CODE:

insert into Showing(ShowingID, CinemaID, ScreenID, FilmID, StartDate, EndDate)

values ('S233', 'A23', 'A15', 'V5646', '01-FEB-2012', '28-FEB-2012');

insert into Showing(ShowingID, CinemaID, ScreenID, FilmID, StartDate, EndDate)

values ('S454', 'A289', 'B10', 'V7686', '14-MAR-2012', '28-APRIL-2012');

insert into Showing(ShowingID, CinemaID, ScreenID, FilmID, StartDate, EndDate)

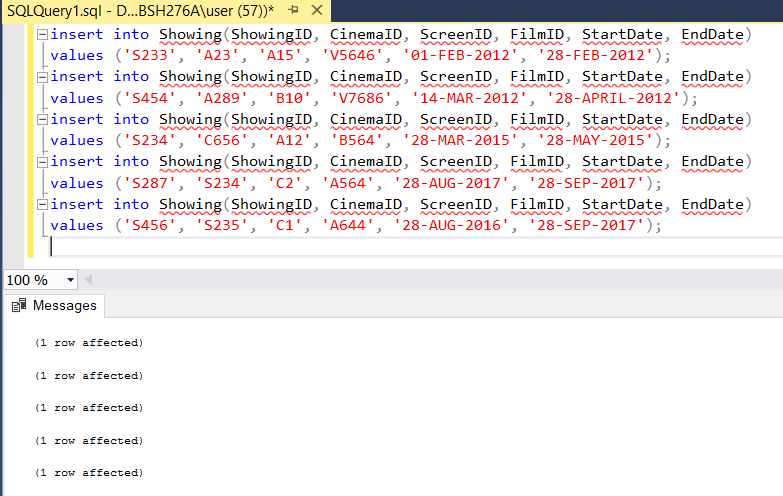
values ('S234', 'C656', 'A12', 'B564', '28-MAR-2015', '28-MAY-2015');

insert into Showing(ShowingID, CinemaID, ScreenID, FilmID, StartDate, EndDate)

values ('S287', 'S234', 'C2', 'A564', '28-AUG-2017', '28-SEP-2017');

insert into Showing(ShowingID, CinemaID, ScreenID, FilmID, StartDate, EndDate)

values ('S456', 'S235', 'C1', 'A644', '28-AUG-2016', '28-SEP-2017');

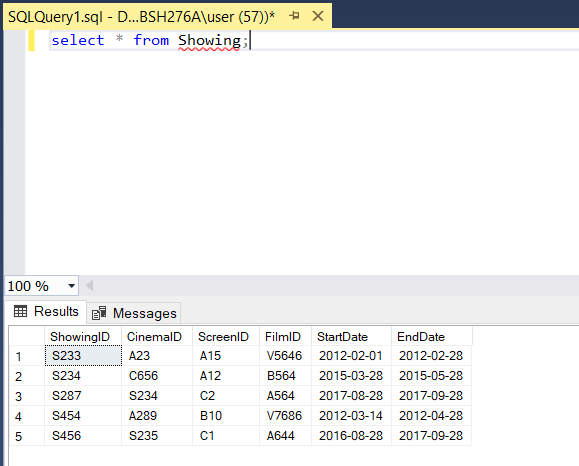


*Figure: Adding values to table Showing*

To view the values of the Table Showing:

CODE:

select \* from Showing;



*Figure: To view Values of the Table Showing*

* **Creating a Table for Performance:**

CODE:

create table Performance (

ShowingID varchar(255) NOT NULL,

PerformanceNo varchar(255) PRIMARY KEY NOT NULL,

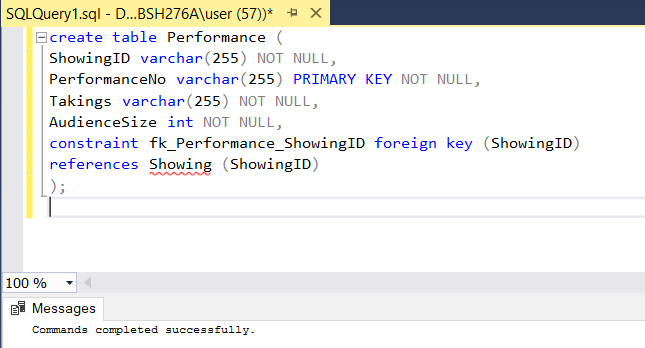
Takings varchar(255) NOT NULL,

AudienceSize int NOT NULL,

constraint fk\_Performance\_ShowingID foreign key (ShowingID)

references Showing (ShowingID)

);



*Figure: Table - Performance*

**ADDING VALUES TO THE TABLE PERFORMANCE:**

CODE:

insert into Performance(ShowingID, PerformanceNo, Takings, AudienceSize)

values ('S233', 'P5', '20', '200');

insert into Performance(ShowingID, PerformanceNo, Takings, AudienceSize)

values ('S454', 'P9', '15', '200');

insert into Performance(ShowingID, PerformanceNo, Takings, AudienceSize)

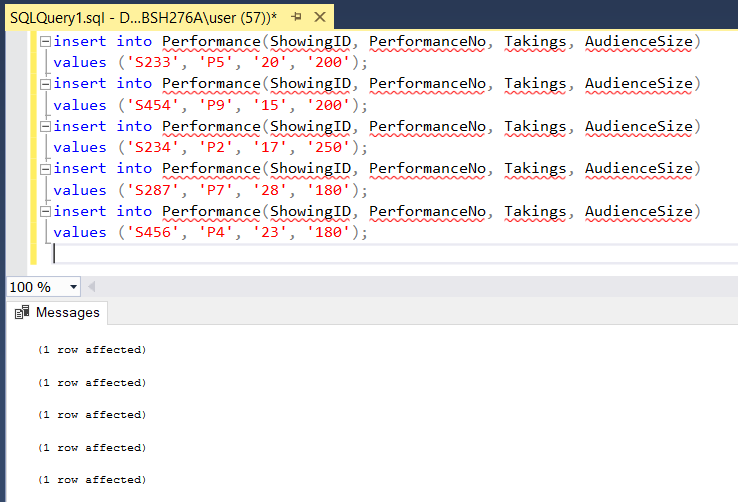
values ('S234', 'P2', '17', '250');

insert into Performance(ShowingID, PerformanceNo, Takings, AudienceSize)

values ('S287', 'P7', '28', '180');

insert into Performance(ShowingID, PerformanceNo, Takings, AudienceSize)

values ('S456', 'P4', '23', '180');

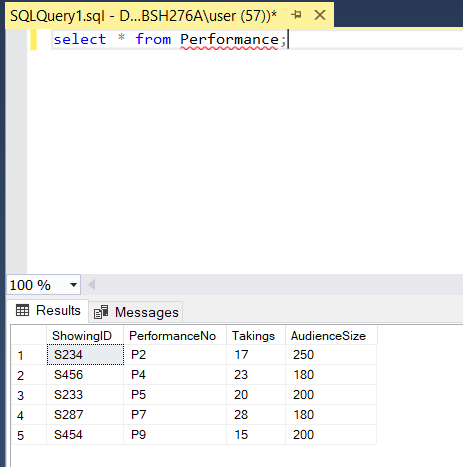


*Figure: Adding values to the Table Performance*

To View the values of the Table Performance:

CODE:

select \* from Performance;



*Figure: To View values of the Table Performance*

**Conclusion:**

Basically in this mini project I have tried to learn & implement a Database based on Cinema DB. Where I have understood how we can gather the requirements, how we can create ER diagrams, design star schema, design dimensional model & fact table. Hence, by using them I have tried to implement commands on creating database, tables, altering tables, using constraint to add foreign key. While doing so I have faced certain errors & bugs, which I did try to rectify to have a proper output.