NATIONAL INSTITUTE OF TECHNOLOGY, WARANGAL

Movie Tickets Management System

Database Management System Project

By:

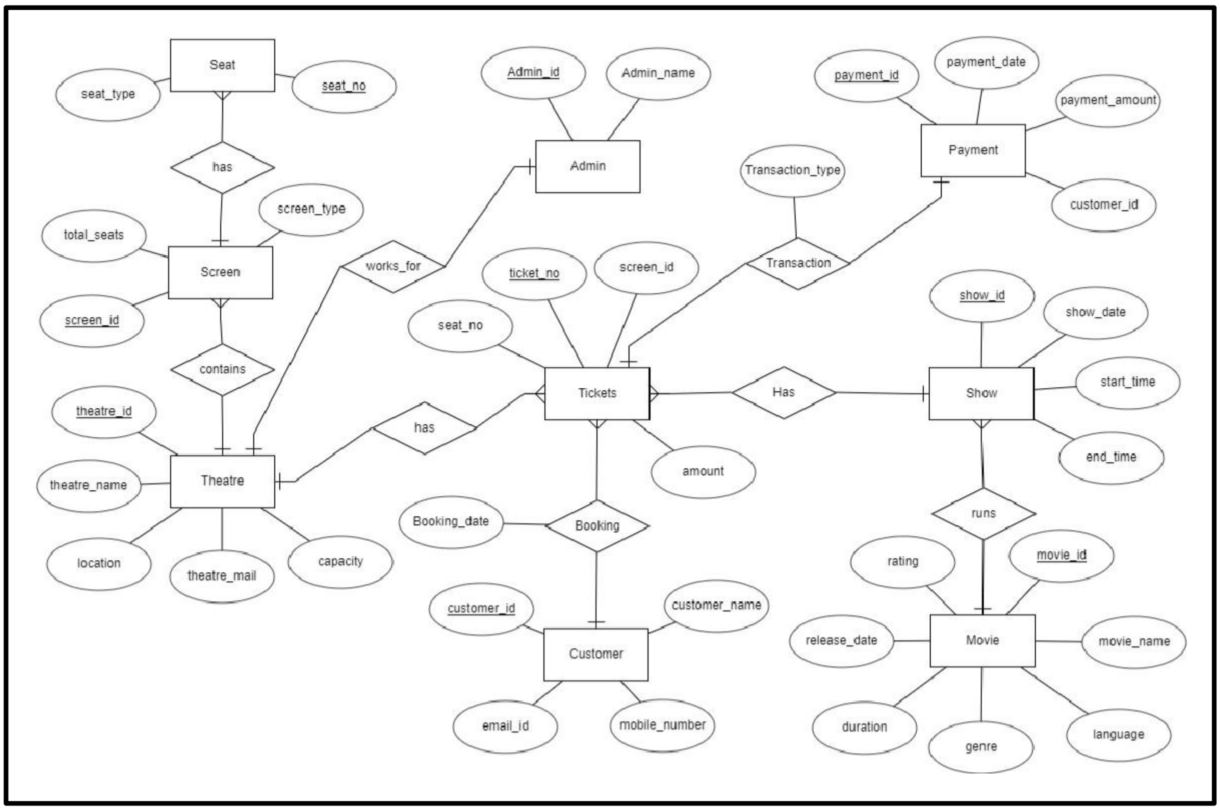
D.Shamitha 22EEB0B29

CONTENT:

Problem statement:

In this project ,we have decided to make a database about movie tickets and theatres by looking at the craze in the public towards Indian cinemas.
A movie ticket database is a collection of information regarding movie tickets that have been sold or reserved. It includes information such as the movie title, showtime, theatre location, ticket price, seat number, and customer information. This information can be used to manage the number of tickets sold , track attendance, payment transactions and analyse customer behaviour. The database can be stored in a computer system which reduces the work load on humans.

ENTITY RELATIONSHIP DIAGRAM:



RELATIONAL SCHEMA:

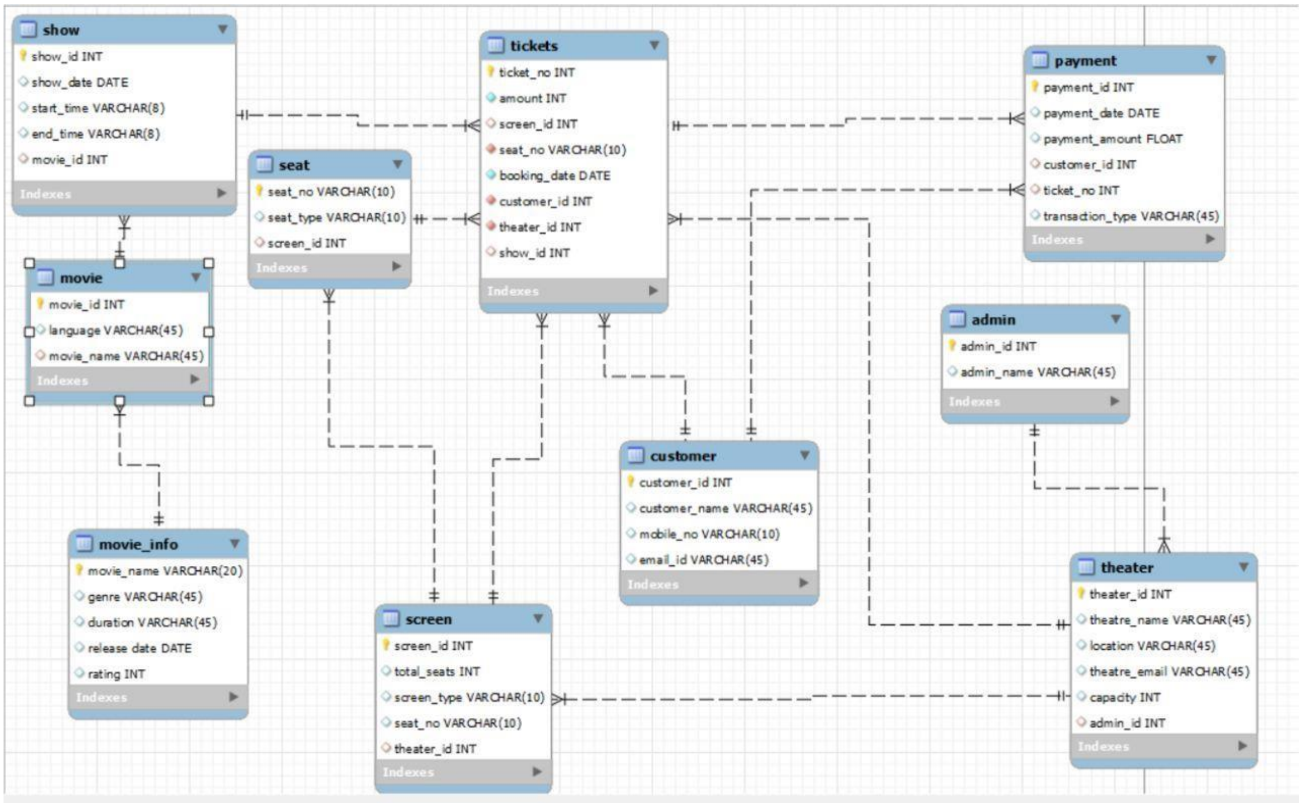


TABLE ASSUMPTIONS:

ticket:

Ticket entity holds complete information about tickets such as ticket_no,amount,booking_date,seat_no,customer_id,show_id,screen_id….etc.
Its Primary Key is ticket_no and it contains customer_id, show_id, seat_no, screen_id, theatre_id as foreign keys from customer, show1, seat, screen, theatre respectively.

customer:

Customer entity holds the information of the customer who bought the ticket.
Here Customer_id isthe unique identifier for each customer in the system.
It contains the attributes such as customer_id, customer_name, mobile_number, email_id.

show1:

show entity holds the information about the Show of the movie whose ticket has been purchased by the customer.
Here Show_id is primary key. It has attributes such as show_id, show_date, start_time, end_time of the movie and movie_id which is foreign key taking reference from Movie entity.

movie:

This entity holds the information about the movie which will be watched by the customer.
Its Primary Key is movie_id. It contains the attributes such as movie_id, movie_name, language, genre, duration, release_date, rating.

payment:

This entity holds the information about the Payment done by the Customer to buy the Movie tickets.
Its Primary key is Payment_id. Its Stores payment_id, payment_amount, transaction_type(Card,Cash,UPI), ticket_no which is foreign key references from Ticket entity.

admin:

This entity holds the information about Admin of the Theatre who sells the Tickets .Such as Admin_id and Admin_name. Here the Primary Key is admin_id.

theatre:

This entity holds the information of the theatre details where the movie is being screened. It contains the attributes such as theatre_id, theatre_name, location, capacity, theatre_mail, Admin_id which isforeign key referencing the Admin entity. Here Primary Key is Theatre_id.

screen:

This entity holds the information about each screen in Theatre.It contains the attributes such as Screen_id, Screen_type(3D,MAXXSCREEN,DOLBY ATMOS), total_seats available in the screen ,theatre_id which is foreign key references the theatre entity. Here the Primary key is screen_id.

seat:

This entity holds information about the seat of the theatre which has been allocated to the customer to watch the movie. It stores info such as Seat_no , Seat_type(recliner, deluxe, etc.) and screen_id which is foreign key referencing the Screen entity. Primary key of this entity is Seat_no.

<u>ASSUMPTIONS:</u>

1. we have assumed that one theatre has one admin itself and one admin belongs to one theatre itself.
2. we have assumed that one theatre sells many tickets but each ticket belongs to one theatre itself.
3. we have assumed that one customer can buy many tickets and each ticket is belongs to one customer itself.
4. we have assumed that one ticket has one payment itself and one payment belongs to one ticket.
5. we have assumed that one theatre has many screens but each screen belongs to one theatre itself.

6. we have assumed that one screen has many seats and each seat is belongs to only one screen itself.

7. we have assumed that one movie can be played in many shows but one show should plays only one movie.

8. we have assumed that one show has many tickets and one ticket belongs to one show itself.

Normal Forms:

1.FIRST NORMAL FORM(1NF): This is the most basic level of normalization.In 1NF,each table cell should contain only a single value,and each column should have unique name.The first normal form helps to eliminate duplicate data and simplify queries.

2.Second Normal Form(2NF):2NF eliminates redundant data requiring that each non-key attribute be dependent on the primary key.This means that each column should be directly related to the primary key,not to other columns.

3.Third Normal Form(3NF):3NF builds on 2NF by requiring that all non-key attributes are independent on each other.This means that each column should be directly related to the primary key,and not to any other columns in same table.

4.Boyce-codd Normal Form(BCNF):BCNF is a stricter form of 3NF that ensures that each determinant in a table is a super key.In other words,BCNF that each non-key attribute is dependent only on the candidate key.

Functional dependencies and Normalization:

1.Ticket:

Ticket_no->{amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id}

So here Ticket_no is primary key.

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,ticket_no and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.So There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. ticket_no to all other attributes so the table is in BCNF.

Customer:

Customer_id->{customer_name,email_id,mobile_number}

Hence the customer_id is primary key.

1NF:As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,customer_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. customer_id to all other attributes so the table is in BCNF.

Payment:

Payment_id->{payment_date,payment_amount,transaction_type,customer_id,ticket_no}

Ticket_no->{payment_id, payment_date,payment_amount,transaction_type,customer_id}

Hence the payment_id,ticket_no are candiadate keys.

Take payment_id as primary key.

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table the primary key is payment_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e.payment_id,ticket_no to all other attributes so the table is in BCNF.

Movie:

Movie_id->{movie_name,language,genre,duration,release_date,rating}

Movie_name->{genre, duration,release_date,rating}

Hence the movie_id is primary key.

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,movie_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In the above functional dependencies we can say that movie_name,genre,language,duration,release_date,rating are non prime attributes and movie_name is determining genre,language,duration,release_date,rating(non-prime->non-prime).Hence we can say it as transitive dependency.

Therfore this table is not in 3NF.To bring this table into 3NF we should do lossless decomposition

1)Movie:movie_id,language,movie_name.

2)Movie_Info:movie_name,genre,duration,release_date,rating.

Hence the Movie_id is primary key of the Movie table and

Movie_name is primary key of the Movie_info table.

Now in movie table all functional dependencies are from candidatekey(movie_id i.e prime attributes) to non prime attributes.Hence the Movie table is in 3NF

And in Movie_info table all functional dependencies are from candidatekey(movie_name i.e prime attribute) to non prime attributes.Hence the Movie_info table is in 3NF

BCNF: In all the above modified tables,only the superkeys are determining all other attributes.Hence we can say that the table is in BCNF.

Seat:

Seat_no->{seat_type,screen_id}

Hence seat_no is primary key

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,seat_no and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. seat_no to all other attributes so the table is in BCNF.

Screen:

Screen_id->{screen_type,total_seats,theatre_id}

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,screen _id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. screen_id to all other attributes so the table is in BCNF.

Admin:

Admin_id->{admin_name)

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e. admin_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. admin_id to all other attributes so the table is in BCNF.

Theatre:

Theatre_id->{theatre_name,location,capacity,theatre_name,admin_id}

Hence theatre_id is primary key

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,theatre_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. theatre_id to all other attributes so the table is in BCNF.

Show1:

Show_id->{start_time,end_time,show_date,movie_id)

Hence the Show_id is primary key.

1NF: As the table contains primary key and all the attributes are atomic attributes and there is no multivalued attributes so the table is in 1NF.

2NF: In this table there is only one primary key i.e,show_id and it is only single attribute so there is no partial dependency so the table is in 2NF.

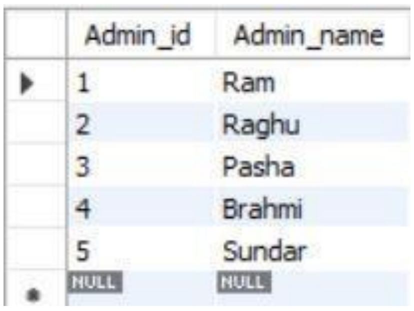
3NF: In this table all functional dependencies are from candidatekey(primeattribute) to non prime attributes.so There is no transitive dependency so the table is in 3NF.

BCNF: Here all Functional dependencies are from super key i.e. show_id to all other attributes so the table is in BCNF.

TABLES CREATION:

Admin:

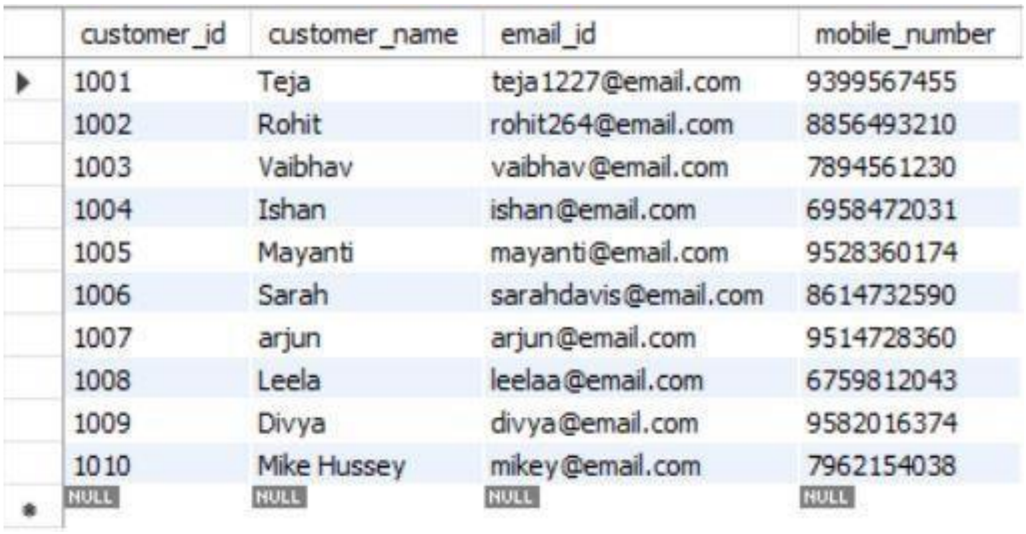
```
create table Admin(
Admin_id int primary key,
Admin_name varchar(50)
);
insert into admin values(1,'Ram');
insert into admin values(2,'Raghu');
insert into admin values(3,'Pasha');
insert into admin values(4,'Brahmi');
insert into admin values(5,'Sundar');
select * from admin;
```



Admin_id	Admin_name
1	Ram
2	Raghu
3	Pasha
4	Brahmi
5	Sundar
NULL	NULL

Customer:

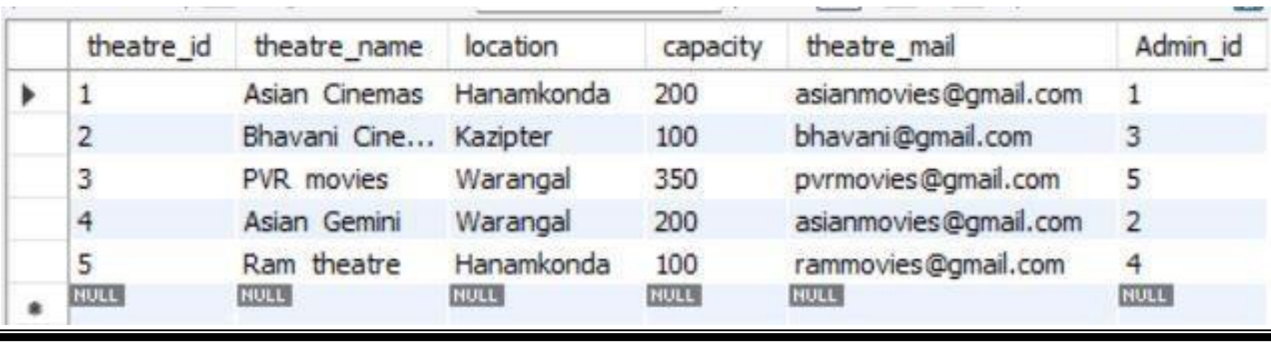
```
create table Customer(
customer_id int primary key,
customer_name varchar(50),
email_id varchar(100),
mobile_number varchar(10)
);
INSERT INTO customer VALUES (1001, 'Teja', 'teja1227@email.com',
'9399567455');
INSERT INTO customer VALUES (1002, 'Rohit', 'rohit264@email.com',
'8856493210');
INSERT INTO customer VALUES (1003, 'Vaibhav', 'vaibhav@email.com',
'7894561230');
INSERT INTO customer VALUES (1004, 'Ishan', 'ishan@email.com',
'6958472031');
INSERT INTO customer VALUES (1005, 'Mayanti', 'mayanti@email.com',
'9528360174');
INSERT INTO customer VALUES (1006, 'Sarah', 'sarahdavis@email.com',
'8614732590');
INSERT INTO customer VALUES (1007, 'arjun', 'arjun@email.com',
'9514728360');
INSERT INTO customer VALUES (1008, 'Leela', 'leelaa@email.com',
'6759812043');
INSERT INTO customer VALUES (1009, 'Divya', 'divya@email.com',
'9582016374');
INSERT INTO customer VALUES (1010, 'Mike Hussey', 'mikey@email.com',
'7962154038');
select * from Customer
```



	customer_id	customer_name	email_id	mobile_number
▶	1001	Teja	teja1227@email.com	9399567455
	1002	Rohit	rohit264@email.com	8856493210
	1003	Vaibhav	vaibhav@email.com	7894561230
	1004	Ishan	ishan@email.com	6958472031
	1005	Mayanti	mayanti@email.com	9528360174
	1006	Sarah	sarahdavis@email.com	8614732590
	1007	arjun	arjun@email.com	9514728360
	1008	Leela	leelaa@email.com	6759812043
	1009	Divya	divya@email.com	9582016374
	1010	Mike Hussey	mikey@email.com	7962154038
*	NULL	NULL	NULL	NULL

Theatre:

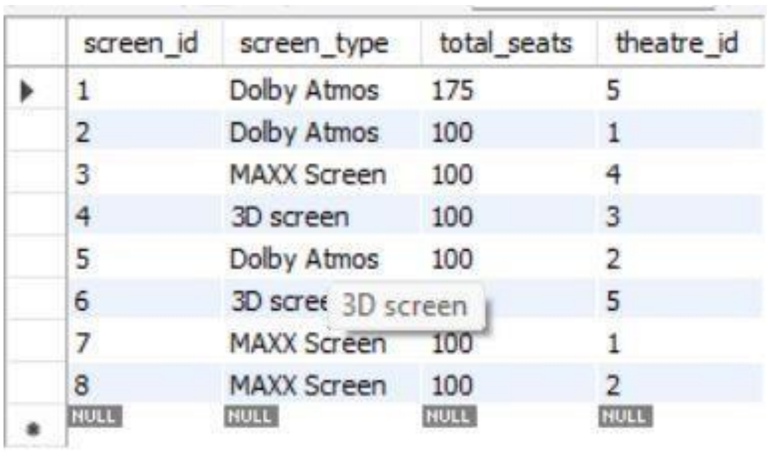
```
create table Theatre(
theatre_id int primary key,
theatre_name varchar(50),
location varchar(50),
capacity int,
theatre_mail varchar(100),
Admin_id int,
foreign key(Admin_id) references Admin(Admin_id)
);
insert into theatre values(1,'Asian
Cinemas','Hanamkonda',200,'asianmovies@gmail.com',1);
insert into theatre values(2,'Bhavani
Cinemas','Kazipter',100,'bhavani@gmail.com',3);
insert into theatre values(3,'PVR
movies','Warangal',350,'pvrmovies@gmail.com',5);
insert into theatre values(4,'Asian
Gemini','Warangal',200,'asianmovies@gmail.com',2);
insert into theatre values(5,'Ram
theatre','Hanamkonda',100,'rammovies@gmail.com',4);
select * from theatre
```



	theatre_id	theatre_name	location	capacity	theatre_mail	Admin_id
▶	1	Asian Cinemas	Hanamkonda	200	asianmovies@gmail.com	1
	2	Bhavani Cine...	Kazipter	100	bhavani@gmail.com	3
	3	PVR movies	Warangal	350	pvrmovies@gmail.com	5
	4	Asian Gemini	Warangal	200	asianmovies@gmail.com	2
	5	Ram theatre	Hanamkonda	100	rammovies@gmail.com	4
*	NULL	NULL	NULL	NULL	NULL	NULL

Screen:

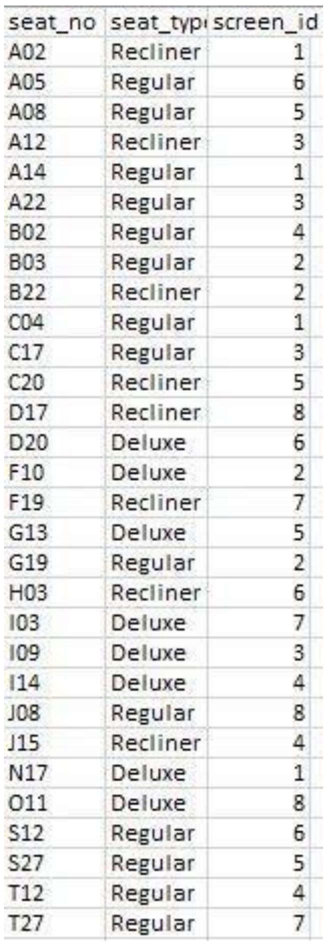
```
create table Screen(
screen_id int primary key,
screen_type varchar(50),
total_seats int,
theatre_id int,
foreign key(theatre_id) references Theatre(theatre_id)
);
insert into screen values(1,'Dolby Atmos',175,5);
insert into screen values(2,'Dolby Atmos',100,1);
insert into screen values(3,'MAXX Screen',100,4);
insert into screen values(4,'3D screen',100,3);
insert into screen values(5,'Dolby Atmos',100,2);
insert into screen values(6,'3D screen',175,5);
insert into screen values(7,'MAXX Screen',100,1);
insert into screen values(8,'MAXX Screen',100,2);
select * from Screen;
```



screen_id	screen_type	total_seats	theatre_id
1	Dolby Atmos	175	5
2	Dolby Atmos	100	1
3	MAXX Screen	100	4
4	3D screen	100	3
5	Dolby Atmos	100	2
6	3D scree 3D screen		5
7	MAXX Screen	100	1
8	MAXX Screen	100	2
NULL	NULL	NULL	NULL

Seat:

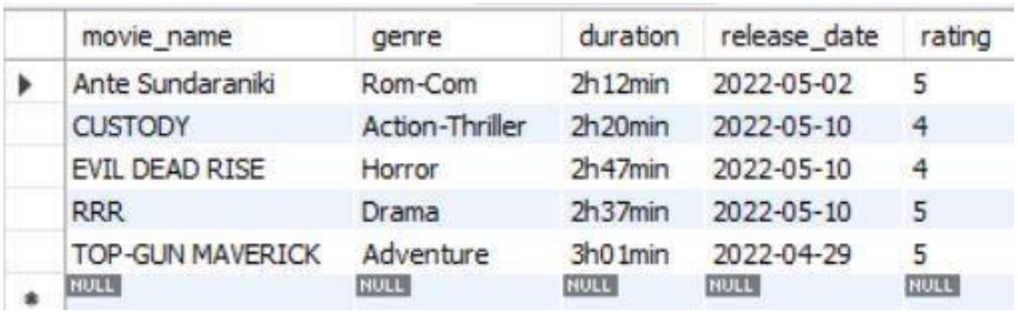
```
create table Seat(

seat_no varchar(30) primary key,

seat_type varchar(50),

screen_id int,

foreign key(screen_id) references Screen(screen_id)

);

insert into seat values('A02','Recliner',1);

insert into seat values('B22','Recliner',2);

insert into seat values('A12','Recliner',3);

insert into seat values('J15','Recliner',4);

insert into seat values('C20','Recliner',5);

insert into seat values('H03','Recliner',6);

insert into seat values('F19','Recliner',7);

insert into seat values('D17','Recliner',8);

insert into seat values('A14','Regular',1);

insert into seat values('B03','Regular',2);

insert into seat values('A22','Regular',3);

insert into seat values('T12','Regular',4);

insert into seat values('S27','Regular',5);

insert into seat values('S12','Regular',6);

insert into seat values('T27','Regular',7);

insert into seat values('J08','Regular',8);

insert into seat values('N17','Deluxe',1);

insert into seat values('F10','Deluxe',2);

insert into seat values('I09','Deluxe',3);

insert into seat values('I14','Deluxe',4);
insert into seat values('G13','Deluxe',5);
insert into seat values('D20','Deluxe',6);
insert into seat values('I03','Deluxe',7);
insert into seat values('O11','Deluxe',8);
insert into seat values('C04','Regular',1);
insert into seat values('G19','Regular',2);
insert into seat values('C17','Regular',3);
insert into seat values('B02','Regular',4);
insert into seat values('A08','Regular',5);
insert into seat values('A05','Regular',6);
select * from seat;
```



seat_no	seat_typ	screen_id
A02	Recliner	1
A05	Regular	6
A08	Regular	5
A12	Recliner	3
A14	Regular	1
A22	Regular	3
B02	Regular	4
B03	Regular	2
B22	Recliner	2
C04	Regular	1
C17	Regular	3
C20	Recliner	5
D17	Recliner	8
D20	Deluxe	6
F10	Deluxe	2
F19	Recliner	7
G13	Deluxe	5
G19	Regular	2
H03	Recliner	6
I03	Deluxe	7
I09	Deluxe	3
I14	Deluxe	4
J08	Regular	8
J15	Recliner	4
N17	Deluxe	1
O11	Deluxe	8
S12	Regular	6
S27	Regular	5
T12	Regular	4
T27	Regular	7

Movie_Info:

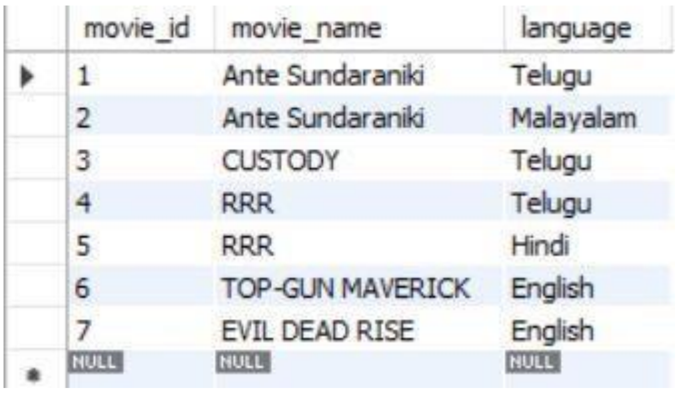
```
create table movie_info(
movie_name varchar(50) primary key,
genre varchar(20),
duration varchar(20),
release_date date,
rating INT
);
insert into movie_info values('Ante Sundaraniki','Rom-Com','2h12min','2022-05-02',4.9);
insert into movie_info values('CUSTODY','Action-Thriller','2h20min','2022-05-10',3.8);
insert into movie_info values('RRR','Drama','2h37min','2022-05-10',4.95);
insert into movie_info values('TOP-GUN MAVERICK','Adventure','3h01min','2022-04-29',4.5);
insert into movie_info values('EVIL DEAD RISE','Horror','2h47min','2022-05-10',4.1);
select * from movie_info;
```



	movie_name	genre	duration	release_date	rating
▶	Ante Sundaraniki	Rom-Com	2h12min	2022-05-02	5
	CUSTODY	Action-Thriller	2h20min	2022-05-10	4
	EVIL DEAD RISE	Horror	2h47min	2022-05-10	4
	RRR	Drama	2h37min	2022-05-10	5
	TOP-GUN MAVERICK	Adventure	3h01min	2022-04-29	5
*	NULL	NULL	NULL	NULL	NULL

Movie:

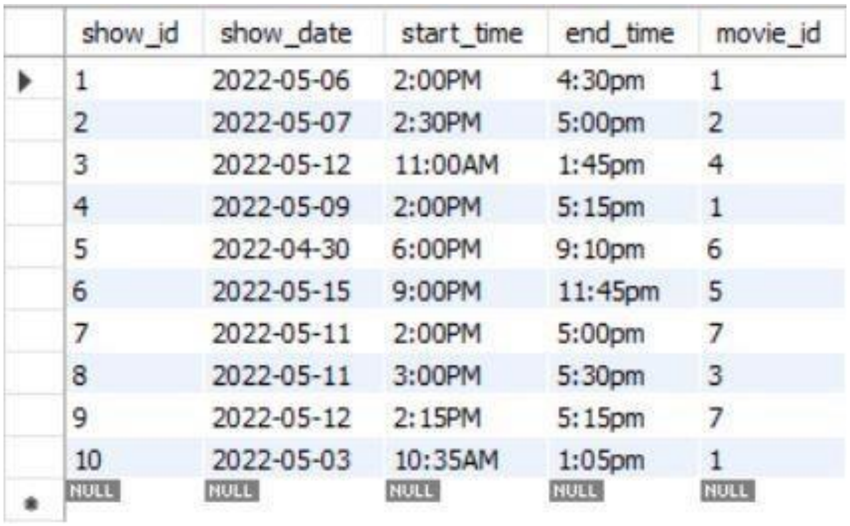
```
create table movie(
movie_id int primary key,
movie_name varchar(50),
language varchar(50),
foreign key (movie_name) references movie_info(movie_name)
);
insert into movie values(1,'Ante Sundaraniki','Telugu');
insert into movie values(2,'Ante Sundaraniki','Malayalam');
insert into movie values(3,'CUSTODY','Telugu');
insert into movie values(4,'RRR','Telugu');
insert into movie values(5,'RRR','Hindi');
insert into movie values(6,'TOP-GUN MAVERICK','English');
insert into movie values(7,'EVIL DEAD RISE','English');
select * from movie
```



movie_id	movie_name	language
1	Ante Sundaraniki	Telugu
2	Ante Sundaraniki	Malayalam
3	CUSTODY	Telugu
4	RRR	Telugu
5	RRR	Hindi
6	TOP-GUN MAVERICK	English
7	EVIL DEAD RISE	English
NULL	NULL	NULL

Show:

```
CREATE TABLE show1 (
show_id int primary key,
show_date date,
start_time varchar(20),
end_time varchar(20),
movie_id int,
foreign key(movie_id) references movie(movie_id)
);
insert into show1 values(1,'2022-05-06','2:00PM','4:30pm',1);
insert into show1 values(2,'2022-05-07','2:30PM','5:00pm',2);
insert into show1 values(3,'2022-05-12','11:00AM','1:45pm',4);
insert into show1 values(4,'2022-05-09','2:00PM','5:15pm',1);
insert into show1 values(5,'2022-04-30','6:00PM','9:10pm',6);
insert into show1 values(6,'2022-05-15','9:00PM','11:45pm',5);
insert into show1 values(7,'2022-05-11','2:00PM','5:00pm',7);
insert into show1 values(8,'2022-05-11','3:00PM','5:30pm',3);
insert into show1 values(9,'2022-05-12','2:15PM','5:15pm',7);
insert into show1 values(10,'2022-05-03','10:35AM','1:05pm',1);
select * from show1
```

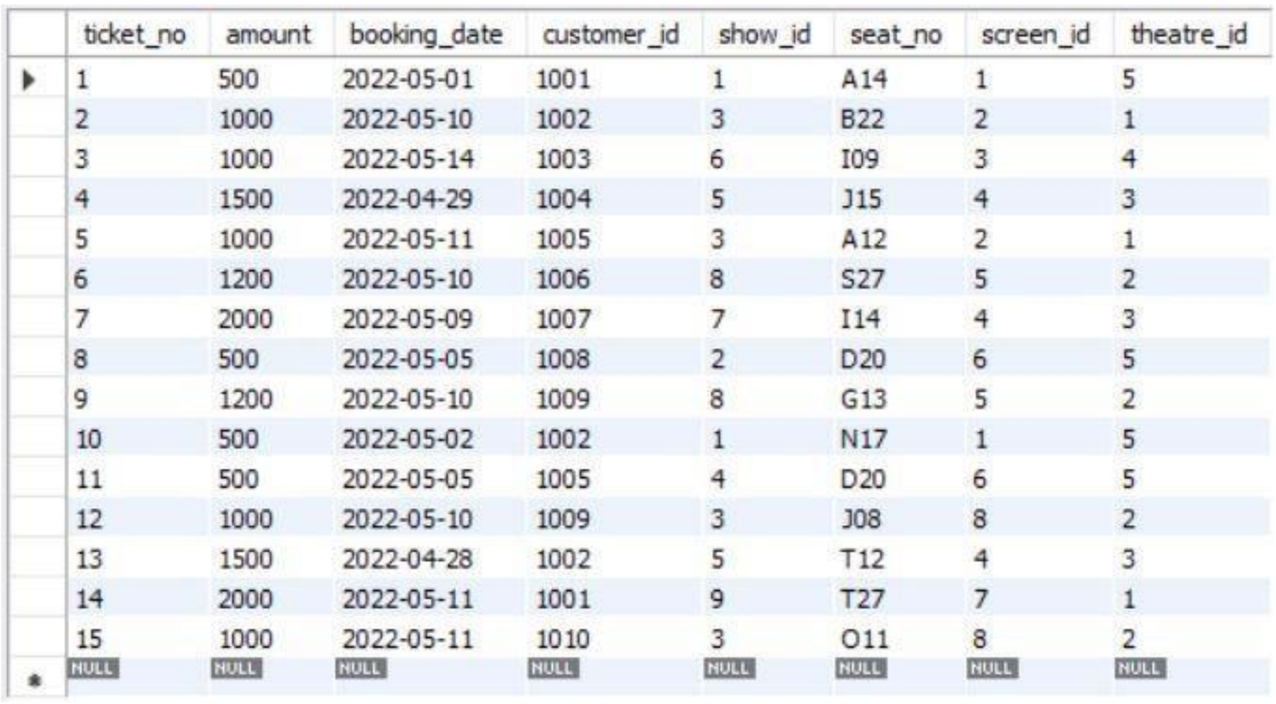


show_id	show_date	start_time	end_time	movie_id
1	2022-05-06	2:00PM	4:30pm	1
2	2022-05-07	2:30PM	5:00pm	2
3	2022-05-12	11:00AM	1:45pm	4
4	2022-05-09	2:00PM	5:15pm	1
5	2022-04-30	6:00PM	9:10pm	6
6	2022-05-15	9:00PM	11:45pm	5
7	2022-05-11	2:00PM	5:00pm	7
8	2022-05-11	3:00PM	5:30pm	3
9	2022-05-12	2:15PM	5:15pm	7
10	2022-05-03	10:35AM	1:05pm	1
NULL	NULL	NULL	NULL	NULL

Tickets:

```
create table Tickets(
ticket_no int(4) primary key AUTO_INCREMENT,
amount int,
booking_date date,
customer_id int,
show_id int,
seat_no varchar(30),
screen_id int,
theatre_id int,
foreign key(customer_id) references Customer(customer_id),
foreign key(show_id) references show1(show_id),
foreign key(seat_no) references Seat(seat_no),
foreign key(screen_id) references Screen(screen_id),
foreign key(theatre_id) references Theatre(theatre_id)
);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(500,'2022-05-01',1001,1,'A14',1,5);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1000,'2022-05-10',1002,3,'B22',2,1);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1000,'2022-05-14',1003,6,'I09',3,4);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1500,'2022-04-29',1004,5,'J15',4,3);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1000,'2022-05-11',1005,3,'A12',2,1);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1200,'2022-05-10',1006,8,'S27',5,2);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(2000,'2022-05-09',1007,7,'I14',4,3);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(500,'2022-05-05',1008,2,'D20',6,5);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1200,'2022-05-10',1009,8,'G13',5,2);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(500,'2022-05-02',1002,1,'N17',1,5);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(500,'2022-05-05',1005,4,'D20',6,5);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1000,'2022-05-10',1009,3,'J08',8,2);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1500,'2022-04-28',1002,5,'T12',4,3);
```

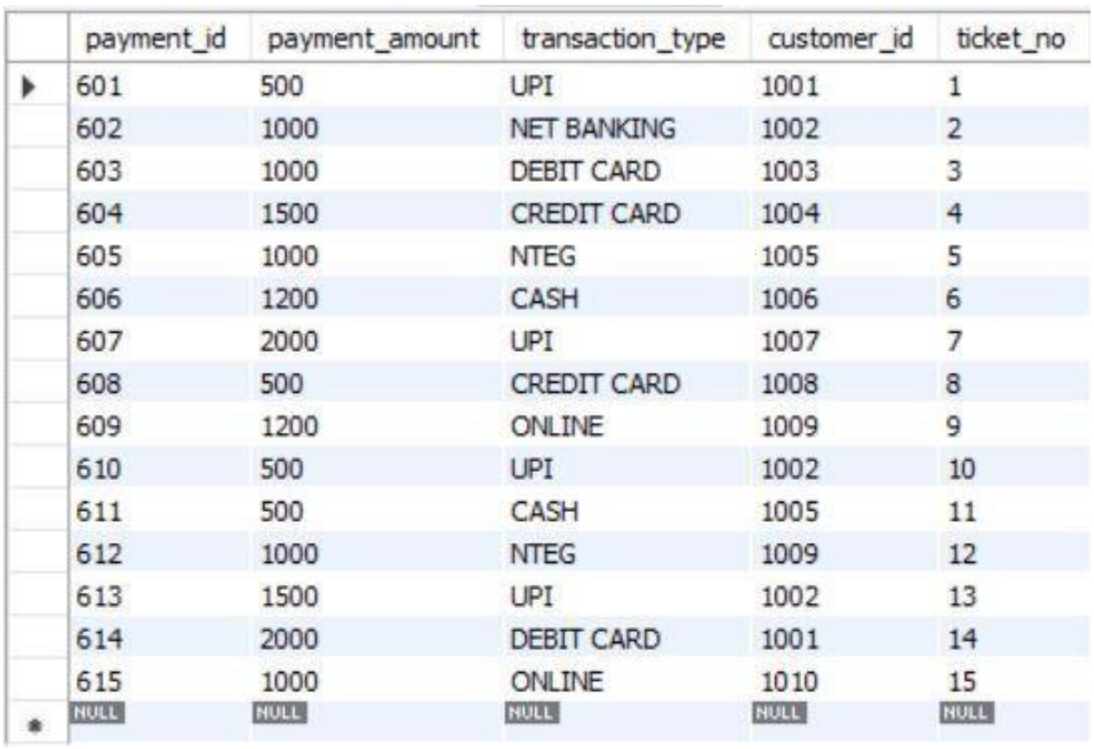
```
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(2000,'2022-05-11',1001,9,'T27',7,1);
insert into
tickets(amount,booking_date,customer_id,show_id,seat_no,screen_id,theatre_id
) values(1000,'2022-05-11',1010,3,'O11',8,2);
select * from Tickets;
```



ticket_no	amount	booking_date	customer_id	show_id	seat_no	screen_id	theatre_id
1	500	2022-05-01	1001	1	A14	1	5
2	1000	2022-05-10	1002	3	B22	2	1
3	1000	2022-05-14	1003	6	I09	3	4
4	1500	2022-04-29	1004	5	J15	4	3
5	1000	2022-05-11	1005	3	A12	2	1
6	1200	2022-05-10	1006	8	S27	5	2
7	2000	2022-05-09	1007	7	I14	4	3
8	500	2022-05-05	1008	2	D20	6	5
9	1200	2022-05-10	1009	8	G13	5	2
10	500	2022-05-02	1002	1	N17	1	5
11	500	2022-05-05	1005	4	D20	6	5
12	1000	2022-05-10	1009	3	J08	8	2
13	1500	2022-04-28	1002	5	T12	4	3
14	2000	2022-05-11	1001	9	T27	7	1
15	1000	2022-05-11	1010	3	O11	8	2
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Payment:

```
create table Payment(
payment_id int primary key,
payment_amount int,
transaction_type varchar(50),
customer_id int,
ticket_no int,
foreign key(customer_id) references Customer(customer_id),
foreign key (ticket_no) references Tickets(ticket_no)
);
insert into payment values(601,500,'UPI',1001,1);
insert into payment values(602,1000,'NET BANKING',1002,2);
insert into payment values(603,1000,'DEBIT CARD',1003,3);
insert into payment values(604,1500,'CREDIT CARD',1004,4);
insert into payment values(605,1000,'NTEG',1005,5);
insert into payment values(606,1200,'CASH',1006,6);
insert into payment values(607,2000,'UPI',1007,7);
insert into payment values(608,500,'CREDIT CARD',1008,8);
insert into payment values(609,1200,'ONLINE',1009,9);
insert into payment values(610,500,'UPI',1002,10);
insert into payment values(611,500,'CASH',1005,11);
insert into payment values(612,1000,'NTEG',1009,12);
insert into payment values(613,1500,'UPI',1002,13);
insert into payment values(614,2000,'DEBIT CARD',1001,14);
insert into payment values(615,1000,'ONLINE',1010,15);
select * from Payment
```

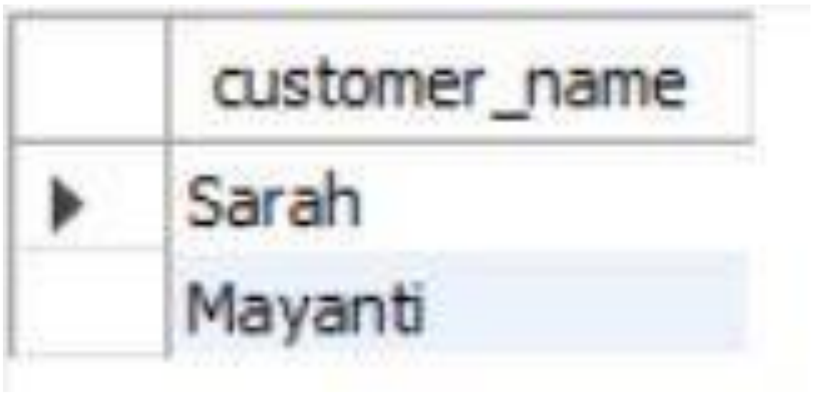


	payment_id	payment_amount	transaction_type	customer_id	ticket_no
▶	601	500	UPI	1001	1
	602	1000	NET BANKING	1002	2
	603	1000	DEBIT CARD	1003	3
	604	1500	CREDIT CARD	1004	4
	605	1000	NTEG	1005	5
	606	1200	CASH	1006	6
	607	2000	UPI	1007	7
	608	500	CREDIT CARD	1008	8
	609	1200	ONLINE	1009	9
	610	500	UPI	1002	10
	611	500	CASH	1005	11
	612	1000	NTEG	1009	12
	613	1500	UPI	1002	13
	614	2000	DEBIT CARD	1001	14
	615	1000	ONLINE	1010	15
*	NULL	NULL	NULL	NULL	NULL

SQL QUERIES:

1)Write a SQL query to find out customers who paid in cash for tickets.

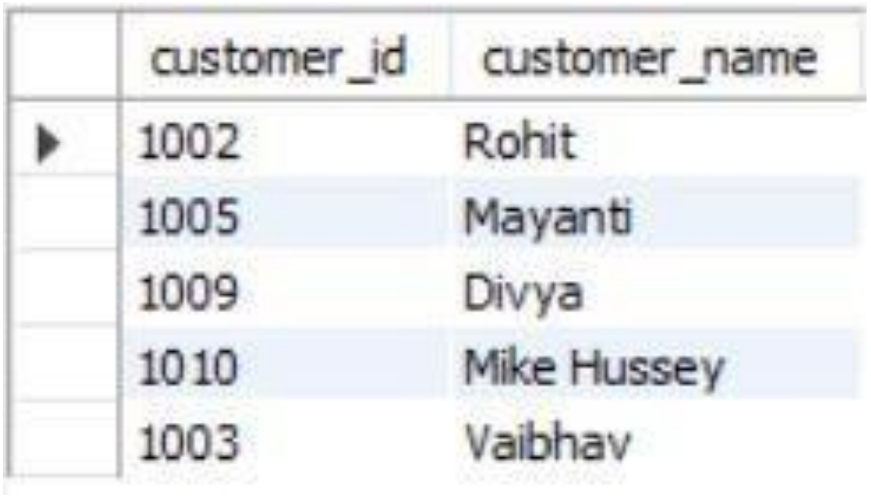
select distinct c.customer_name from customer c inner join payment p on c.customer_id=p.customer_id and p.transaction_type='CASH';



	customer_name
▶	Sarah
	Mayanti

2) Write a SQL query to find out which customer went to the movie RRR.

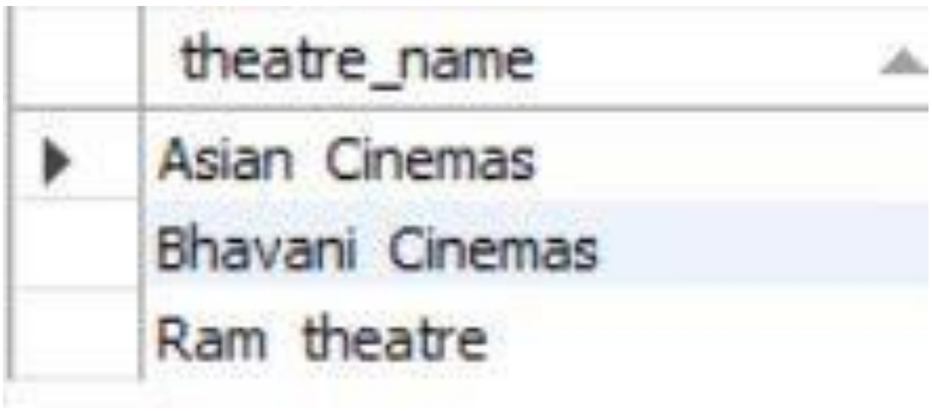
```
SELECT DISTINCT c.customer_id,c.customer_name
FROM customer c
inner join tickets t ON c.customer_id = t.customer_id
inner join show1 s ON t.show_id = s.show_id
inner join movie m ON s.movie_id = m.movie_id
WHERE m.movie_name = 'RRR';
```



customer_id	customer_name
1002	Rohit
1005	Mayanti
1009	Divya
1010	Mike Hussey
1003	Vaibhav

3) Write a SQL query to find out the theatres with more than one screen.

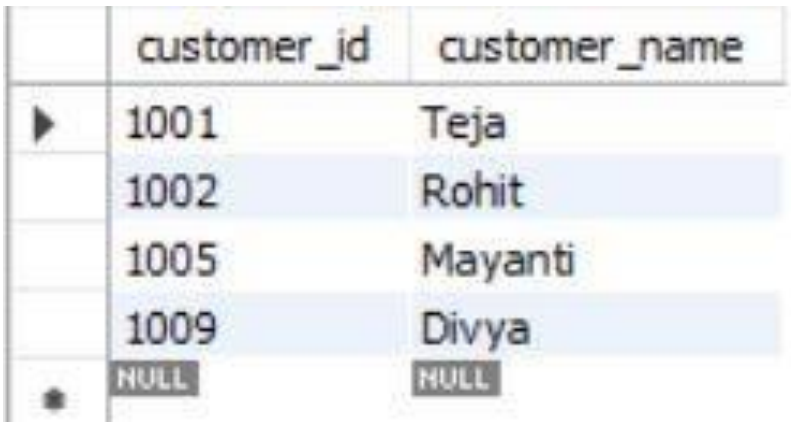
```
select theatre_name from theatre where theatre_id in(select distinct
theatre_id from screen    group by theatre_id    having count(screen_id)>1);
```



theatre_name
Asian Cinemas
Bhavani Cinemas
Ram theatre

4) Write a SQL query to find out which customer who bought more than 1 ticket.

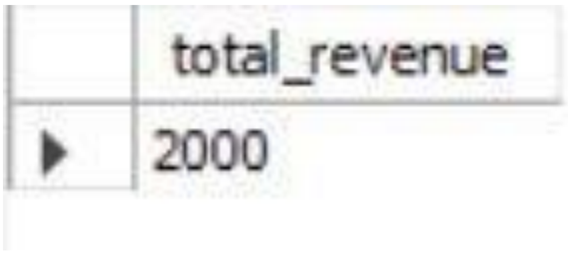
select customer_id, customer_name from customer where customer_id in(
select distinct customer_id from tickets group by customer_id having
count(ticket_no)>1);



	customer_id	customer_name
▶	1001	Teja
	1002	Rohit
	1005	Mayanti
	1009	Divya
*	NULL	NULL

5) Write a SQL query to find the total tickets revenue for the movie Ante Sundariniki.

SELECT SUM(t.amount) AS total_revenue FROM tickets t INNER JOIN show1 s ON t.show_id = s.show_id INNER JOIN movie m ON s.movie_id = m.movie_id and m.movie_name = 'Ante Sundaraniki';



	total_revenue
▶	2000

THANKYOU