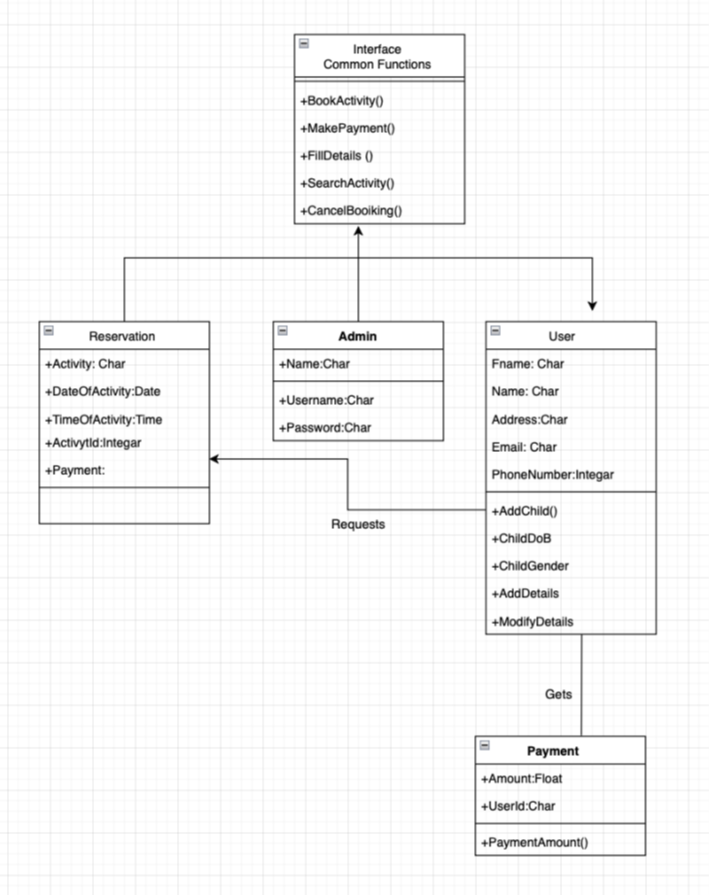
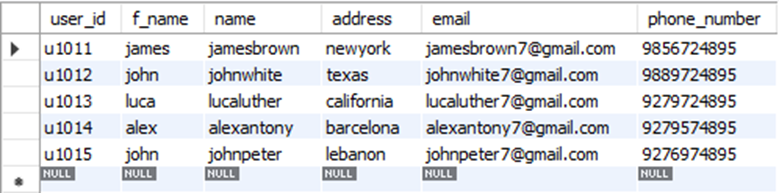
# Online Booking System

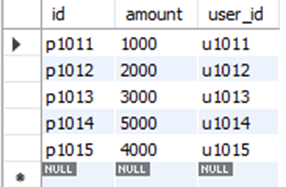


# Tables

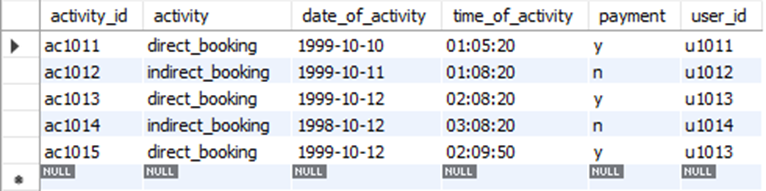
User table



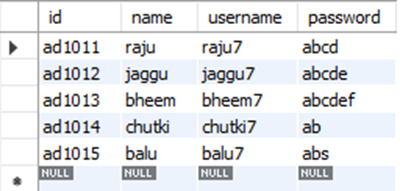
Payment table



Reservation table



Admin table



Create and insert queries

create table admin (

id varchar(20) unique not null,

name varchar(20) not null,

username varchar(20) not null,

password varchar(20) not null,

primary key (id)

);

create table user (

user\_id varchar(20) unique not null,

f\_name varchar(20) not null,

name varchar(20) not null,

address varchar(30) not null,

email varchar(30) not null,

phone\_number varchar(11) not null,

primary key (user\_id)

);

create table reservation (

activity\_id varchar(20) unique not null,

activity varchar(20) not null,

date\_of\_activity date not null,

time\_of\_activity time not null,

payment varchar(2) not null,

primary key (activity\_id),

user\_id varchar(20) not null,

foreign key (user\_id)

references user (user\_id)

);

create table payment (

id varchar(20) unique not null,

amount integer not null,

user\_id varchar(20) not null,

foreign key (user\_id)

references user (user\_id)

);

insert into admin values('ad1011', "raju", "raju7", "abcd"), ('ad1012', "jaggu", "jaggu7", "abcde"), ('ad1013', "bheem", "bheem7", "abcdef"), ('ad1014', "chutki", "chutki7", "ab"),

('ad1015', "balu", "balu7", "abs");

insert into user values('u1011', "james", "jamesbrown", "newyork", "jamesbrown7@gmail.com", "9856724895"), ('u1012', "john", "johnwhite", "texas", "johnwhite7@gmail.com", "9889724895"),

('u1013', "luca", "lucaluther", "california", "lucaluther7@gmail.com", "9279724895"), ('u1014', "alex", "alexantony", "barcelona", "alexantony7@gmail.com", "9279574895"),

('u1015', "john", "johnpeter", "lebanon", "johnpeter7@gmail.com", "9276974895");

insert into reservation values('ac1011', "direct\_booking", "1999-10-10", "01:05:20", "y", "u1011"), ('ac1012', "indirect\_booking", "1999-10-11", "01:08:20", "n", "u1012"),

('ac1013', "direct\_booking", "1999-10-12", "02:08:20", "y", "u1013"), ('ac1014', "indirect\_booking", "1998-10-12", "03:08:20", "n", "u1014"),

('ac1015', "direct\_booking", "1999-10-12", "02:09:50", "y", "u1013");

insert into payment values('p1011', 1000, "u1011"), ('p1012', 2000, "u1012"), ('p1013', 3000, "u1013"), ('p1014', 5000, "u1014"), ('p1015', 4000, "u1015");

# Two Triggers

Trigger to validate phone number:

DELIMITER //

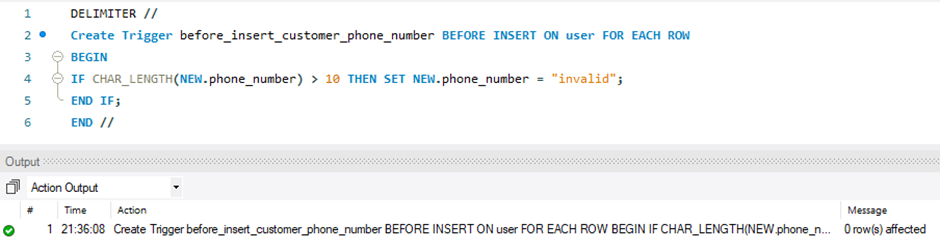
Create Trigger before\_insert\_user\_phone\_number BEFORE INSERT ON user FOR EACH ROW

BEGIN

IF CHAR\_LENGTH(NEW.phone\_number) > 10 THEN SET NEW.phone\_number = "invalid";

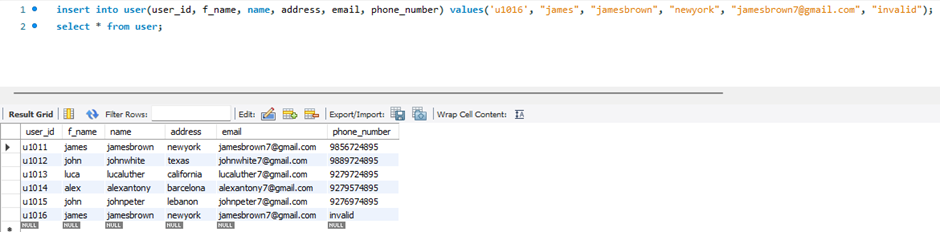
END IF;

END //



insert into user(user\_id, f\_name, name, address, email, phone\_number) values('u1016', "james", " jamesbrown", "newyork", "jamesbrown7@gmail.com", "invalid");

select \* from user;



Trigger to validate duplicate email

DELIMITER //

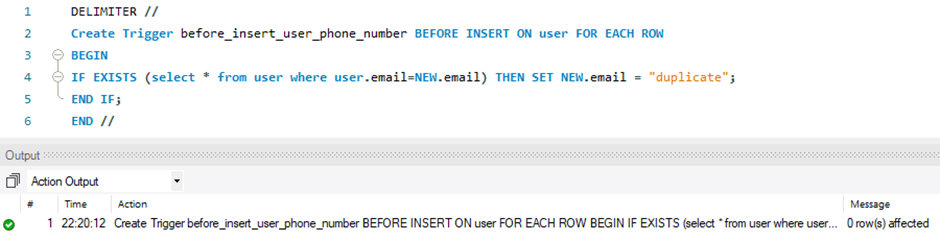
Create Trigger before\_insert\_user\_phone\_number BEFORE INSERT ON user FOR EACH ROW

BEGIN

IF EXISTS (select \* from user where user.email=NEW.email) THEN SET NEW.email = "duplicate";

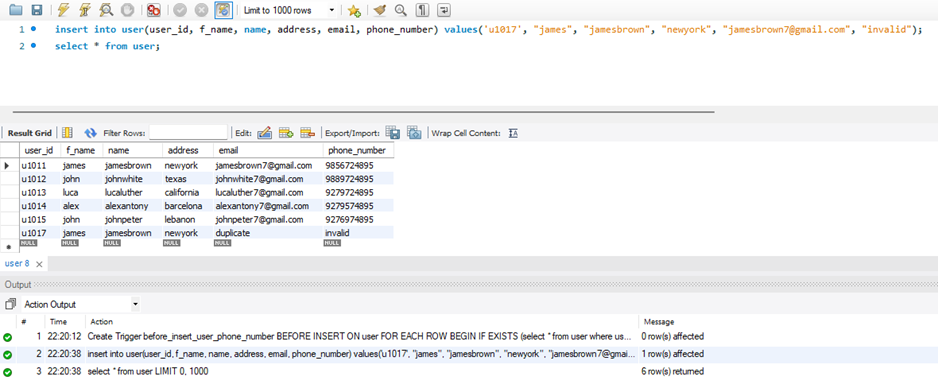
END IF;

END //



insert into user(user\_id, f\_name, name, address, email, phone\_number) values('u1017', "james", "jamesbrown", "newyork", "jamesbrown7@gmail.com", "invalid");

select \* from user;



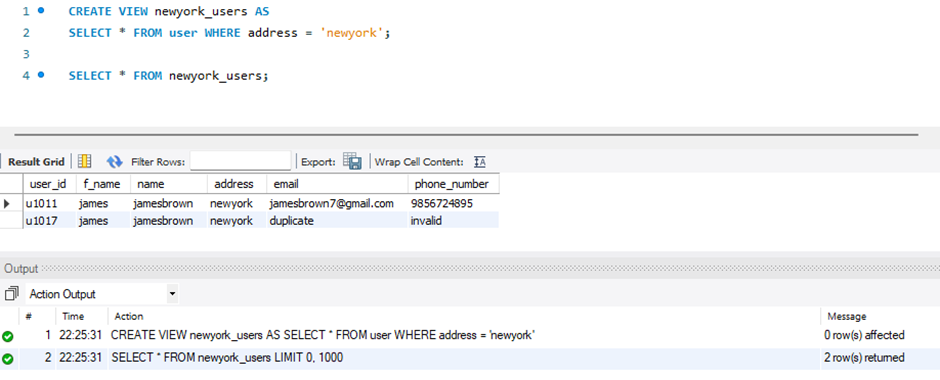
# Two Views

View for showing users from a particular location

CREATE VIEW newyork\_users AS

SELECT \* FROM user WHERE address = 'newyork';

SELECT \* FROM newyork\_users;

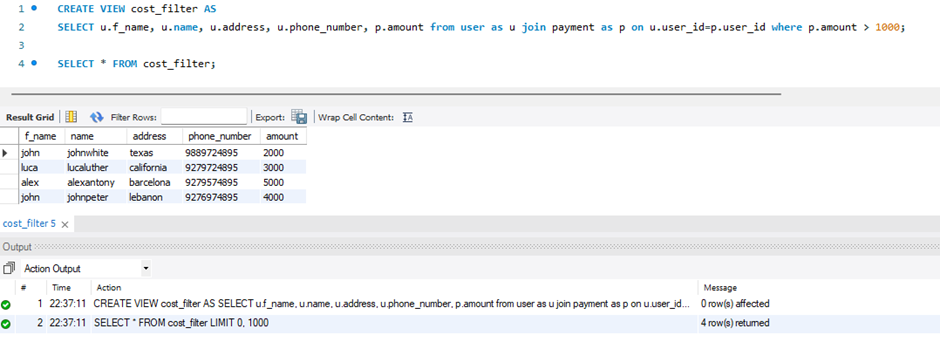


View to show customers with payment amount above a particular limit

CREATE VIEW cost\_filter AS

SELECT u.f\_name, u.name, u.address, u.phone\_number, p.amount from user as u join payment as p on u.user\_id=p.user\_id where p.amount > 1000;

SELECT \* FROM cost\_filter;



# Innovation

You usually come to a point when using software where you need assistance from the developers sooner or later. Nothing is more aggravating in those situations than not being able to acquire the assistance you expect or even assistance that was promised from the software house. Frequently, a thorough Help database is enough for you to immediately find the answer to issues with bookings, reservations, or capacity. But if that's not the case, you ought to be able to get in touch with the developer over the phone and online. One piece of advice: test the Support function as well when evaluating the booking software.

# References

‘MySQL Triggers’ (2020). Available at: <https://www.w3resource.com/mysql/mysql-triggers.php> (Accessed: 26 February 2020).

‘MySQL Views’ (2020). Available at: <https://www.w3schools.com/mysql/mysql_view.asp> (Accessed: 26 February 2020).