Write your answers in the box below only. Do not write on the back or outside the box.

Database Systems — CSci 4380
Midterm Exam #2
November 1, 2018

RCS ID: ______________ @rpi.edu **Name:** ____________________

RIN # : ______________

Rules. The exam is 110 minutes for a total of 100 points. Open book and notes. Do not use any electronic tools including your computer, phone or tablet. Work alone. You **cannot** talk to anyone in class, or share notes or thoughts.

Question 1. Write the following queries using **SQL** for the data model below. The data model is described in detail in the back of the exam.

Similar to Homeworks #4 and #5, your solutions should be a single SQL expression. However, you can use any SQL construct that you wish. Make sure that your queries are easily readable and syntactically correct.

Users(email, password, name, street, state, city, zip, country, ccno)
Houses(id, label, description, street, state, city, zip, country, price, owneremail,avgrating)
HouseAmenities(houseid, amenity)
Trips(id, fromdate, todate, isconfirmed, totalprice, customeremail, houseid)
Reviews(tripid, isclean, isgoodvalue, isgoodrenter, reviewtext)
Messages(id, senderemail, receiveremail, messagetext, sentdatetime, prevmid,tripid)

(a) (10 points) Return the id of all trips that either started or ended in 2017 or the trip had an associated message that was sent in 2017. (Hint: remember `extract (year from datevalue)`).

(b) (10 points) Find and return the name, state, city of all users who stayed in the same house as customer in two different trips with start dates at least 90 days apart and reviewed the earlier trip with `isclean = 1`.

(c) (12 points) Return the name of amenities that are listed for at least three houses in New York state such that there is a confirmed trip in the database for each house. These amenities should not appear in a house listed in any other state.

(d) (12 points) Find all unconfirmed trips for a house in Troy, New York that were supposed to start on '10/31/2018'. For each trip, return the house id, house label, email of the owner as well as number of messages sent by the owner for this trip.

(e) (10 points) Delete all reviews entered by users who are also owners of houses. Such users should have no business reviewing others.

Question 2 (14 points). Write a trigger that activates after a tuple is inserted into the `Reviews` table. Recall that each review is for a trip and each trip is for a specific house.

The trigger updates the tuple in relation `Houses` for this house and sets the average rating attribute (`Houses.avgrating`) to the average of the `Reviews.isgoodvalue` values for all trips involving this house.

Complete the trigger description given below for simplicity. You are free to use any pl/pgsql type procedural code here.

```
CREATE FUNCTION avgreview_f () RETURNS trigger AS $$
DECLARE




BEGIN




























RETURN NEW;
END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER avgreview_trigger AFTER INSERT ON reviews
FOR EACH ROW EXECUTE PROCEDURE avgreview_f();
```

Question 3 (a-10 points). Create a view called `housestats` that returns for each house in the `Houses` table the following attributes: `Houses.id, Houses.label, Houses.state, Houses.city, numtrips (total number of confirmed trips), numdaysrented=(total number of days the house is rented by the confirmed trips)` by completing the following expression:

```
create view housestats(id, label, state, city, numtrips, numdaysrented) as
select
```

Question 3 (b-10 points). Use your view from Question 3(a) above to find the most popular houses in Troy, New York based on `numtrips`. For these houses, return all the attributes in the view `housestats`.

Question 4 (12 points). You are given the following table definitions and table contents.

```
CREATE TABLE ROLES(id INT PRIMARY KEY, name VARCHAR(10) NOT NULL, type VARCHAR(10)) ;
CREATE TABLE SHOWS(id INT PRIMARY KEY, title VARCHAR(100) NOT NULL);
CREATE TABLE APPEARS(id INT PRIMARY KEY, sid INT, rid INT NOT NULL
       , FOREIGN KEY (sid) REFERENCES SHOWS(id) ON DELETE CASCADE ON UPDATE SET NULL
       , FOREIGN KEY (rid) REFERENCES ROLES(id) ON DELETE CASCADE ON UPDATE SET NULL);
```

Roles

id	name	type
1	Rick	Rick
2	Morty	Morty
3	PickleRick	Rick
4	Evil Rick	Rick
5	Rick Morty	Morty
6	Jaguar	

Shows

id	title
5	Meeseeks and Destroy
12	A Rickle in Time
24	Pickle Rick
28	The Ricklantis Mixup

Appears

id	sid	rid
1	5	1
2	5	2
3	24	1
4	24	2
5	24	3
6	24	6
7	28	1
8	28	2
9	28	5

For each operation below, describe which rows from which tables are changed/deleted and why (or why not). Assume each operation operates on the table contents listed above (hence each part is independent).

(a) `DELETE FROM roles WHERE name = ’Jaguar’ ;`

(b) `UPDATE shows SET id = 24 WHERE name LIKE ’%Rick%’;`

(c) `UPDATE roles SET id = 8 WHERE id = 5;`

(d) `DELETE FROM shows WHERE NOT EXISTS (SELECT * FROM appears WHERE sid=shows.id);`

Use this page for scratch work only. Do not share your solutions or any drafts of your solutions with anyone.

Data model to be used in Exam #2

This is a data model based on the E-R problem from Homework#3, storing information about house rentals in a system similar to Airbnb. It is simplified in various ways to make it easier to use in an exam. Please read carefully. The keys of each relation are underlined.

Users(email, password, name, street, state, city, zip, country, ccno)

Stores information about all the users in the system, renters and customers including credit card number (ccno).

Houses(id, label, description, street, state, city, zip, country, price, owneremail, avgrating)

Each house is listed by an owner and we store the email of the owner for each house. The daily price of a house is fixed for simplicity. Average rating is the average rating value for this house.

HouseAmenities(houseid, amenity)

This relation lists the various amenities a house has such as 'free wifi', 'garage parking', etc,

Trips(id, fromdate, todate, isconfirmed, totalprice, customeremail, houseid)

Each trip has a start and end date, whether it is confirmed or not (true/false value) and a total price for the trip. Each trip is by a user (customer), so his/her email is stored. This is the person paying for the trip. Each trip is for a specific house, so the id of the house is also stored. (The renter can be deduced from the owner of the house).

Reviews(tripid, isclean, isgoodvalue, isgoodrenter, reviewtext)

We will only store the reviews of the houses in this database. There can only be one review for each trip, so we will use tripid as the key. Each review is for a specific house and its renter (stored in other relations) and contains 1-5 star values for isclean, isgoodvalue, isgoodrenter as well as a review text.

Messages(id, senderemail, receiveremail, messagetext, sentdatetime, prevmid, tripid)

Finally, we store messages from one user (sender) to another user (receiver) together with the text and datetime it was sent. The attribute `prevmid` can be null if this is an original email or store the id of the message that this message was sent in response to. Each message is for a specific trip.