CS 480: DATABASE SYSTEMS – Summer 2020

Homework 3: RA, SQL, and Relational Normalization Theory – 100 pts

Due on: Saturday, July 25

No late submissions accepted

Type your answers in the space provided in this pdf file, or in the word document and upload it to Gradescope. Don't forget to mark the answers for each question when submitting the homework.

Handwritten answers are NOT acceptable. Only submissions through Gradescope before the deadline will be graded.

Name: Ivan Madrid

NetID: imadri2

Part A (60 pts): Queries

Consider the following database:

Person (Pname, Street, City, Cname, Salary)
Company (Cname, City, Sales)
Branch (Cname, City)

The *Person* relation provides the information about a person including the name, address (street and city), the company for which the person works, and his/her salary. The *Company* table provides the city in which the company is headquartered plus the company's total sales. The *Branch* relation provides the cities in which the company has branches (might be different than HQ's city).

1. (30 pts) Write expressions in SQL, to retrieve answers to the following queries:
   a. Find the names of all companies that have a branch in every city where *Amazon* also has a branch.

```
select Cname
from branch
where City exist in all (Select city from branch where Cname = 'Amazon')
```

b. Find pairs consisting of the name of a company and the total salaries paid to employees of that company, for those companies that are headquartered in Chicago and have at least a hundred employees.

```
Select c.Cname, sum(p.salary) as Total
from comapany c
inner join person p
on c.Cname =p.Cname
where c.city = 'Chicago'
group by c.Cname having count (p.name) >= 100
```

c. For every city, find the companies that have the highest number of employees living in that city, and output triples consisting of the city name, company name, and the number of employees of that company living in that city i.e., (City, Cname, N).

```
Select p.City, p.Cname, p.count(p.name) as N
from person p
group by p.Cname
```

Consider the following car dealer database:

Customer (SSN, name, gender, city)
Sales (VID, SSN, price, sales_year)
Vehicle (VID, make, model, manufacture_year)

The customer relation gives the social security number (SSN), name, gender and the city of residence of customers. The Sales relation contains information about purchases of vehicles. Each of its tuple contains the VID of the vehicle purchased, the SSN of the customer (i.e.,purchaser), the price and the year of the purchase. The Vehicle relation contains information about all the vehicles such as the VID, make, model and the year of manufacture.

2. (30 pts) Write expressions in SQL, to retrieve answers to the following queries:
    a. Find the make and model of the least popular vehicle (The vehicle bought by the lowest number of customers).

```
select v.make, v.model
from vehicle v
join sales s
on v.VID = s.VID
group by s.VID
order by count (*) asc limit 1
```

    b. Find the names of all customers who paid a price higher than the average sale price of the same vehicles (same make, model and year).

```
select c.name
from customer c
join sales s
on s.ssn = c.ssn
where s.price > (select avg(s.price)
                 from sales s2
                 where s2.vid = s.vid)
```

c. Find the **vehicle** (particular VID) with the highest sale price for each maker.

```
select s.vid, max(s.price
from sales s
inner join vehicle v
on s.vid = v.vid
group by v.make)
```

Part B (40 pts):

3. (20 pts) Consider the schema (R, F), where R=ABCDE, F= {C→EA, E→D, DB→C, AC→E}
   a. (5 pts) Find all the keys of the relation, and for each one, prove it is indeed a key.

DB+={BDCEA}
CB+ = {BCEAD}
BE+= {BEDCA}

   b. (10 pts) Decompose the schema into a set of BCNF schemas. Is this decomposition lossless? Is it dependency-preserving? Explain your answer.

C->E
C->A
E->D
DB->C
AC->E

c. (5 pts) What is the attribute closure of EAB?

EAB+ = {EABDC}

4. (20 pts) Consider the following set *F* of functional dependencies on the relation schema *r* (*A*, *B*, *C*, *D*, *E*, *F*):

$$A \rightarrow BCD$$
$$BC \rightarrow DE$$
$$B \rightarrow D$$
$$D \rightarrow A$$

a. (5 pts) Compute *B*+.

B->BD
BD->ABD
ABD->ABCD
ABCD->ABCDE
B+ = ABCDE

b. (5 pts) Prove that *AF* is a superkey of *r*.

AF->BCD
AFBCD->AFBCDE

c. (10 pts) Give a BCNF decomposition of *r* using the original set of functional dependencies.

r1(A,B,C,D)
r2(A,E)
r3(A,E)