CS2102 Database Systems

Semester 1 2019/2020

Tutorial 03 (*Selected Answers*)

Quiz

1. Consider the following relational database, where the primary key of each table is shown underlined:
    - **Pizzas** (pizza)
    - **Customers** (cname, area)
    - **Restaurants** (rname, area)
    - **Contains** (pizza, ingredient)
    - **Sells** (rname, pizza, price)
    - **Likes** (cname, pizza)

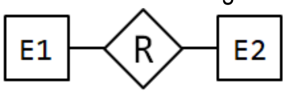
    Pizzas indicates all the pizzas of interest. **Customers** indicates the name and location of each customer. **Restaurants** indicates the name and location of each restaurant. **Contains** indicates the ingredients used in each pizza. **Sells** indicates the pizzas sold by restaurants and their prices. **Likes** indicates the pizzas that customers like.

    The following are all the foreign key constraints on the database schema:
    - `Contains.pizza` is a foreign key that refers to `Pizzas.pizza`
    - `Sells.rname` is a foreign key that refers to `Restaurants.name`
    - `Sells.pizza` is a foreign key that refers to `Pizzas.pizza`
    - `Likes.cname` is a foreign key that refers to `Customers.pizza`
    - `Likes.pizza` is a foreign key that refers to `Pizzas.pizza`

    Write SQL query to answer each of the following questions. Remove duplicate records from all query results.

    a) Find all pizzas that is sold by some restaurants; exclude any pizza that is not sold by any restaurants.
    b) Find all customers that likes ‘Marinara’.
    c) Find all area where ‘Marinara’ is being sold.

2. Consider the following ER diagram where there are 5 entries in `E1` and 5 entries in `E2`.

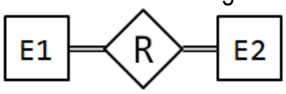


    What is the maximum and minimum number of entries in `R`?

3. Consider the following ER diagram where there are 5 entries in `E1` and 5 entries in `E2`.

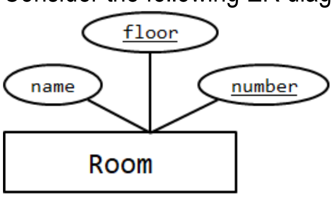
    What is the maximum and minimum number of entries in `R`?

4. Consider the following ER diagram where there are 5 entries in `E1` and 5 entries in `E2`.



    What is the maximum and minimum number of entries in `R`?

5. Consider the following ER diagram. What are the primary keys of `Room`?



6. Consider a schema `R(a:numeric, b:varchar(2), c:integer)`. Which of the following insertion will fail?

    a) `INSERT INTO R VALUES (1, ‘A’, 3);`
    b) `INSERT INTO R (b,c,a) VALUES (1, ‘A’, 3);`
    c) `INSERT INTO R (c,b,a) VALUES (1.0, ‘A’, 3);`
    d) `INSERT INTO R VALUES (1.0, ‘A’, 3);`
    e) `INSERT INTO R (b,c) VALUES (‘A’, 3);`

Tutorial Questions *[Discussion: 7(a), 7(b), 7(c), 8(a), 8(b)]*

7. Consider the following relational database, where the primary key of each table is shown underlined:
   - **Pizzas** (<u>pizza</u>)
   - **Customers** (<u>cname</u>, area)
   - **Restaurants** (<u>rname</u>, area)
   - **Contains** (<u>pizza, ingredient</u>)
   - **Sells** (<u>rname, pizza</u>, price)
   - **Likes** (<u>cname, pizza</u>)

   Pizzas indicates all the pizzas of interest. **Customers** indicates the name and location of each customer. **Restaurants** indicates the name and location of each restaurant. **Contains** indicates the ingredients used in each pizza. **Sells** indicates the pizzas sold by restaurants and their prices. **Likes** indicates the pizzas that customers like.

   The following are all the foreign key constraints on the database schema:
   - `Contains.pizza` is a foreign key that refers to `Pizzas.pizza`
   - `Sells.rname` is a foreign key that refers to `Restaurants.name`
   - `Sells.pizza` is a foreign key that refers to `Pizzas.pizza`
   - `Likes.cname` is a foreign key that refers to `Customers.pizza`
   - `Likes.pizza` is a foreign key that refers to `Pizzas.pizza`

   Write SQL query to answer each of the following questions. Remove duplicate records from all query results.

   a) Find all area where 'Homer' can go to find at least one of the pizza that he likes.

   b) Suppose that the database contains an additional relation `Dislikes(`<u>`cname,pizza`</u>`)` which indicates the pizzas that customers do not like. The database also satisfies the following constraint: for every customer $c \in \pi_{\text{cname}}(\text{Customers})$ and for every pizza $p \in \pi_{\text{pizza}}(\text{Contains})$, either $(c, p) \in$ Likes or $(c, p) \in$ Dislikes (*in other words, you know the likes and dislikes of every customers with respect to all pizzas, and they cannot both like and dislike a pizza*). Given this database, find all customer pairs (C_1, C_2) such that C_1 likes some pizza that C_2 does not like.

   c) Find all customer-restaurant pairs (C,R) where C and R are both located in the same area, and C likes some pizza that is sold by R.

Solution:

a) <u>SQL Query from</u> $\pi_{\text{R.area}}\left(\sigma_{\text{R.rname=S.rname} \wedge \text{S.pizza=L.pizza} \wedge \text{L.cname='Homer'}}(\rho_{\text{R}}(\text{Restaurants}) \times \rho_{\text{S}}(\text{Sells}) \times \rho_{\text{L}}(\text{Likes}))\right)$

```
SELECT DISTINCT R.area
FROM   Restaurants R, Sells S, Likes L
WHERE  R.rname = S.rname
  AND  S.pizza = L.pizza
  AND  L.cname = 'Homer';
```

b) <u>SQL Query from</u> $\pi_{\text{L.cname,D.cname}}\left(\sigma_{\text{L.pizza=D.pizza}}(\rho_{\text{L}}(\text{Likes}) \times \rho_{\text{D}}(\text{Dislikes}))\right)$

```
SELECT DISTINCT L.cname, D.cname
FROM   Likes L, Dislikes D
WHERE  L.pizza = D.pizza;
```

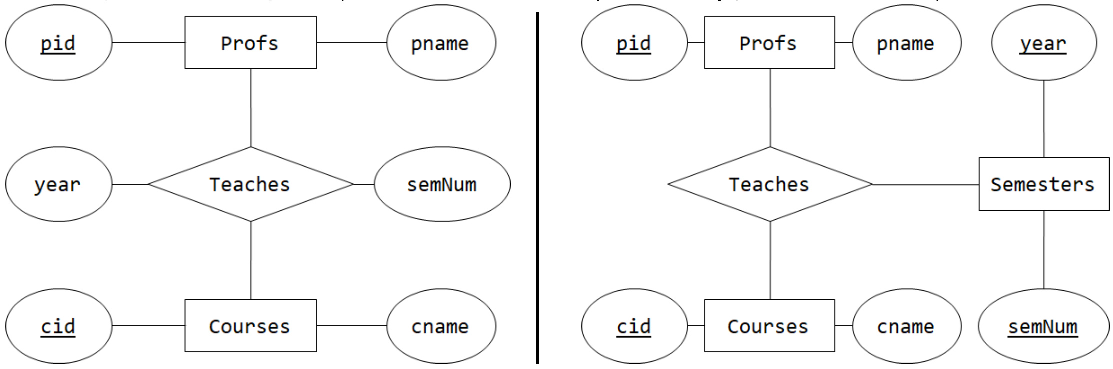
c) <u>SQL Query from</u>

$$\pi_{\text{C.cname,R.rname}}\left(\sigma_{\text{S.pizza=L.pizza} \wedge \text{R.rname=S.rname} \wedge \text{C.cname=L.cname} \wedge \text{C.area=R.area}}(\rho_{\text{R}}(\text{Restaurants}) \times \rho_{\text{C}}(\text{Customers}) \times \rho_{\text{S}}(\text{Sells}) \times \rho_{\text{L}}(\text{Likes}))\right)$$

```
SELECT DISTINCT C.cname, R.rname
FROM   Customers C, Restaurants R, Sells S, Likes L
WHERE  S.pizza = L.pizza AND R.rname = S.rname
  AND  C.cname = L.cname AND C.area  = R.area;
```

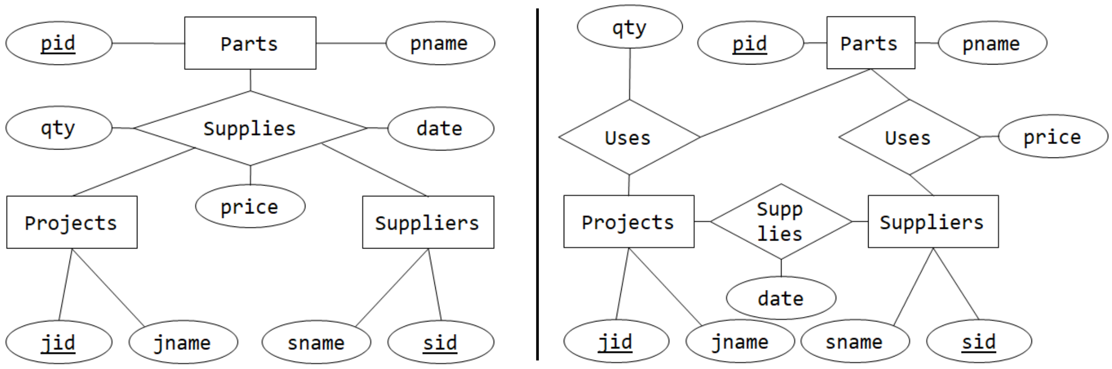
8. For each of the applications below, we have two different ER designs. Discuss whether the designs are equivalent in the sense of capturing the same semantics and constraints of the application.

   a) This application is about the courses (with identifier `cid` and name `cname`) taught by professors (with identifier `pid` and name `pname`) in different semesters (identified by `year` and `semNum`).



Design A **Design B**

   b) This application is about the parts (with identifier `pid` and name `pname`) supplied by suppliers (with identifier `sid` and name `sname`) to projects (with identifier `jid` and name `jname`), where `price` represents the unit price of part, `qty` represents the quantity of a part, and `date` represents the date of transaction.



Design A **Design B**

Solution:

a) <u>Design A</u>

Each professor-course `(pid,cid)` pair can participate at most once in the `Teaches` relationship set.

<u>Design B</u>

The pair `(pid,cid)` can participate multiple times in `Teaches` for different semesters. This is done by modeling semester as an entity set.

b) <u>Design A</u>

If `(S,P,J)` is in `Supplies` relationship set, then Supplier `S` supplies part `P` to project `J`. The value of each of the relationship attributes `date`, `qty`, and `price` depends on the `(S,P,J)` triples. For example, supplier `S` could be selling part `P` to project `J` for *\$10* each, but the same supplier `S` could be selling the same part `P` to another project `J2` for *\$15* each.

<u>Design B</u>

The value of attributes `date`, `qty`, and `price` depends only on a pair of entities. The price of a part `P` sold by supplier `S` is fixed for all projects. Even if `(S,P)` is in `Sells` relationship set (*supplier* `S` *sells part* `P`), `(S,J)` is in `Supplies` relationship set (*supplier* `S` *supplies to project* `J`), and `(J,P)` is in `Uses` relationship set (*project* `J` *uses parts* `P`), does not necessarily mean that supplier `S` sells part `P` to project `J`.