Homework Assignment 2

Due: 11:59PM April 7, 2023

1. Fill in the blanks.
(a) The (Data-definition language (DDL)) provides commands for defining relation schemas, deleting relations, and modifying relation schemas.
(b) The (Data-manipulation language (DML)) provides the ability to query information from the database and to insert tuples into, delete tuples from, and modify tuples in the database.
(c) The primary key attributes are required to be (nonnull) and (unique).
(d) The (foreign-key) specifies that the values of attributes for any record in the relation must correspond to values of the primary key attributes of some tuple in another relation.
(e) Subqueries that return only one tuple containing a single attribute are called (scalar subqueries).
(f) The (order by) clause causes the records in the result of a query to appear in sorted order.
(g) The (with) clause provides a way of defining a temporary relation whose definition is available only to the query in which the clause occurs.

2. The SQL LIKE operator is case sensitive (in most systems), but the LOWER() function on strings can be used to perform case-insensitive matching. Show how to write a query that finds departments whose names contain the string “sci” as a substring, regardless of the case.

```
WHERE LOWER(Department) LIKE '%sci%'
```

3. Show that, in SQL, <> ALL is identical to NOT IN

The <> ALL operator returns true if the value on the left side of the operator is not equal to any value returned by the subquery on the right side. Similarly, the NOT IN operator returns true if the value on the left side is not existed in the set of values returned by the subquery on the right side. Since, “not equal to any value” and “not existed in the set of values” are the equivalent propositions, these two operators are identical.

4. List two reasons why null values might be introduced into the database

One of the reasons why null values might be introduced is to denote absence of value in the database. For example, in the student relation, new students can be added but their department has not been assigned yet. In this example, null value may be useful to denote the absence of department which is a missing information.
Another reason is to represent the inapplicability of the attribute. For example, in a database containing a table of people with a column for spouse name, a null value can be used in this column for people who are not married yet.

5. Consider the relational database of Figure 3.19, where the primary keys are underlined. Given an expression in SQL for each of the following queries.

a.
```
SELECT E.ID
FROM employee AS E, works AS W
WHERE E.ID = W.ID AND W.comp_name ≠ 'First Bank Corporation';
```

b.
```
SELECT E.ID E.name E.city
FROM employee AS E,
    (SELECT (*) FROM works
        WHERE comp_name ≠ 'First Bank Corporation'
          AND salary > 10000) AS W
WHERE E.ID = W.ID;
```

c.
```
SELECT E.ID
FROM employee AS E, works AS W
WHERE E.ID = W.ID
 AND W.salary > SELECT MAX(salary) FROM works
                WHERE company_name = 'small Bank corporation'
```

d.
```
R1 = SELECT (DISTINCT city)
     FROM company
     WHERE company_name = 'small Bank corporation';

SELECT company_name FROM R1, company
WHERE company.city = some (R1);
```

e.
```
R1 = SELECT company_name, COUNT(ID) AS count
     FROM works GROUP BY company_name;

SELECT company_name FROM R1
where count ≥ MAX(count);
```

```
f. R1 = SELECT comp_name , AVG (salary) AS avg
       FROM works GROUP BY comp_name ;

   SELECT comp_name FROM R1
   WHERE avg > (SELECT a FROM R1
                WHERE comp_name
                  = 'First Bank Corporation');
```

```
g. UPDATE employee SET city = 'Newtown'
   WHERE ID = 12345
```

```
h. SELECT ID, person_name
   FROM employee AS E, works AS W, company AS C
   WHERE E.ID = W.ID
   AND   W.company_name = C.company_name
   AND   E.city = C.city;
```

```
i. SELECT ID, person_name FROM employee, works
   WHERE employee.ID = works.ID
         works.salary > AVG (works.salary) ;
```

```
j. SELECT company_name FROM (
       SELECT company_name, SUM(salary) AS sum
       FROM works GROUP BY company_name
       HAVING sum = MIN (sum)) ;
```

```
k. UPDATE works SET salary = salary * 1.1
   WHERE company_name = 'First Bank Corporation' ;
```

```
l. DELETE FROM works
   WHERE company_name = 'Small Bank Corporation' ;
```

6. Find the answers to the following questions and provide the SQL queries showing how you find them. All queries should be complete to obtain the listed answers solely by themselves

a

Answer:

	`count(t.course_id)`
1	7

SQL Query to obtain your answer:

```
use university;
select count(t.course_id)
from teaches as t, teaches as s
where t.semester = "Fall" and
      s.semester = "Spring" and
      t.course_id = s.course_id;
```

b

Answer:

	`count(distinct (title))`
1	133

SQL Query to obtain your answer:

```
use university;
select count(distinct (title)) from course;
```

c

Answer:

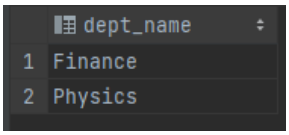
	dept_name	avg_montly_salary
1	Cybernetics	8029

SQL Query to obtain your answer:

```
use university;
select dept_name, round(avg(salary)/12) as avg_montly_salary
from instructor
group by dept_name
having dept_name = "Cybernetics";
```

d

Answer:



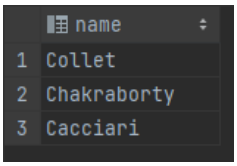
	dept_name
1	Finance
2	Physics

SQL Query to obtain your answer:

```
use university;
select dept_name
from department
where budget > (select budget
                from department
                where dept_name = "Psychology")
Order By dept_name;
```

e

Answer:



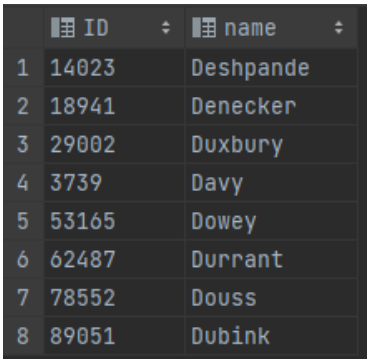
	name
1	Collet
2	Chakraborty
3	Cacciari

SQL Query to obtain your answer:

```
use university;
select name
from student
where dept_name = "Geology" and
      name like "C%";
```

f

Answer:



	ID	name
1	14023	Deshpande
2	18941	Denecker
3	29002	Duxbury
4	3739	Davy
5	53165	Dowey
6	62487	Durrant
7	78552	Douss
8	89051	Dubink

SQL Query to obtain your answer:

```
use university;
select s.ID, s.name
from student as s
where s.dept_name = "History" and
      s.name like "D%" and
      (select count(t.course_id)
       from takes as t, course as c
       where t.ID = s.ID and
             c.title like "%Music%" and
             t.course_id = c.course_id < 5);
```

g

Answer:

	ID	name	dept_name	tot_cred
1	94998	Krishnakumar	Physics	81

SQL Query to obtain your answer:

```
use university;
select *
from student
where length(name) > 11 and
      (dept_name = "Physics" Or dept_name = "Comp. Sci.");
```

h

Answer:

	`count(ID)`
1	99

SQL Query to obtain your answer:

```
use university;
select count(ID)
from student
where dept_name = "Comp. Sci." and
      tot_cred > (select min(tot_cred)
                  from student
                  where dept_name = "English");
```

i

Answer:

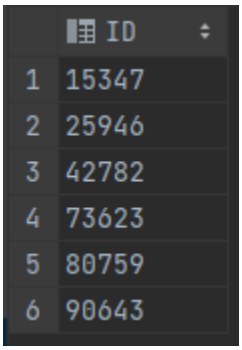
	building
1	Saucon
2	Stabler
3	Taylor
4	Whitman

SQL Query to obtain your answer:

```
use university;
select b.building
from (select building, sum(capacity) as sum
      from classroom group by building having sum > 100) as b;
```

j

Answer:



	ID
1	15347
2	25946
3	42782
4	73623
5	80759
6	90643

SQL Query to obtain your answer:

```
use university;
select i.ID from instructor as i
            where exists(select * from teaches as t
                                  where t.ID = i.ID and
                                        year < 2003) and
                  not exists(select * from teaches as t
                                      where t.ID = i.ID and
                                            year >= 2003);
```

k

Query:

```
use university;
select dept_name, count(ID) as num_students
from student group by dept_name
             order by num_students desc;
```

l

Answer:

```
use university;
select * from course
         where title in
               (select title from course group by title
                      having count(title) = 1);
```

7. Find the answers to the following questions and provide the SQL queries showing how you find them. All queries should be complete to obtain the listed answers solely by themselves

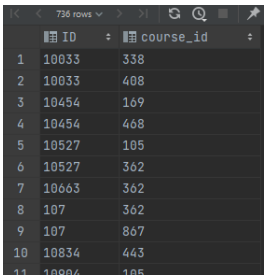
A

Query:

```
use university;
select course_id, sec_id, semester, year, count(ID) as num_student
from takes group by course_id, sec_id, semester, year;
```

B

Answer:



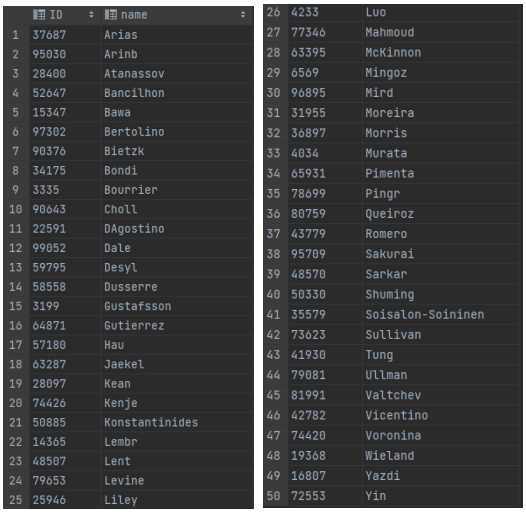
	ID	course_id
1	10033	338
2	10033	408
3	10454	169
4	10454	468
5	10527	105
6	10527	362
7	10663	362
8	107	362
9	107	867
10	10834	443

SQL Query to obtain your answer:

```
use university;
select ID, course_id from takes group by ID, course_id having COUNT(*) >= 2;
```

C

Answer:



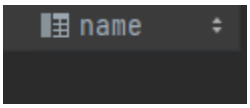
	ID	name
1	37687	Arias
2	95030	Arinb
3	28400	Atanassov
4	52647	Bancilhon
5	15347	Bawa
6	97302	Bertolino
7	90376	Bietzk
8	34175	Bondi
9	3335	Bourrier
10	90643	Choll
11	22591	DAgostino
12	99052	Dale
13	59795	Desyl
14	58558	Dusserre
15	3199	Gustafsson
16	64871	Gutierrez
17	57180	Hau
18	63287	Jaekel
19	28097	Kean
20	74426	Kenje
21	50885	Konstantinides
22	14365	Lembr
23	48507	Lent
24	79653	Levine
25	25946	Liley
26	4233	Luo
27	77346	Mahmoud
28	63395	McKinnon
29	6569	Mingoz
30	96895	Mird
31	31955	Moreira
32	36897	Morris
33	4034	Murata
34	65931	Pimenta
35	78699	Pingr
36	80759	Queiroz
37	43779	Romero
38	95709	Sakurai
39	48570	Sarkar
40	50330	Shuming
41	35579	Soisalon-Soininen
42	73623	Sullivan
43	41930	Tung
44	79081	Ullman
45	81991	Valtchev
46	42782	Vicentino
47	74420	Voronina
48	19368	Wieland
49	16807	Yazdi
50	72553	Yin

SQL Query to obtain your answer:

```
use university;
select i.ID, i.name from instructor as i where i.ID not in (select teach.ID
from teaches as teach, takes as take
    where teach.course_id = take.course_id and
          teach.sec_id = take.sec_id and
          teach.semester = take.year and
          teach.year = take.year and
          take.grade like "%A%")
    order by i.name;
```

D

Answer:



SQL Query to obtain your answer:

```
use university;
select name
from instructor as i
where not exists
    ((select course_id
      from course as c
      where c.dept_name = i.dept_name)
    except
        (select distinct course_id
        from teaches as t
        where t.ID = i.ID))
order by name desc;
```

E

The result of the query would not be zero if the instructor table contains any NULL values in the salary column. This is because the AVG function ignores NULL values when calculating the average, while the SUM and COUNT functions do not.

For example, consider an instance of the instructor table with the following data:

ID	**name**	**dept_name**	**salary**
1	John	CS	80000
2	Jane	CS	NULL

In this case, the result of the query would be calculated as follows:

The AVG(salary) would return 80000, because it only considers the non-NULL salary value for John.

The SUM(salary) would return 80000, because it sums all salary values including NULLs (which are treated as 0).

The COUNT(*) would return 2, because it counts the number of records regardless of whether they contain NULLs.

Therefore, the result of the query would be 80000 - (80000/2) = 40000, which is not zero.