Homework Assignment 2, by 22000546 Yeeun Lee

1.
(a) Data-definition language
(b) Data-manipulation language
(c) nonnull, unique
(d) foreign key
(e) scalar subqueries
(f) order by
(g) with

2.
SELECT dept_name FROM department WHERE LOWER(dept_name) LIKE '%sci%'

3.
In the 'a' (<>ALL / NOT IN) 'B' operation, <> ALL means that 'a' is different from all the elements of 'B'. NOT IN means 'a' is not included in 'B'. It is the same as saying that 'a' is different from all elements of B, so both operations are identical.

4.
It is to express the case where the value does not exist and the case where the value exists but is unknown.

5.
(a)
SELECT ID FROM works WHERE company_name <> 'First Bank Corporation';

(b)
SELECT A.ID, person_name, city FROM employee AS A, works AS B WHERE A.ID=B.ID AND company_name='First Bank Corporation' AND salary > 10000;

(c)
SELECT ID FROM works WHERE salary > ALL (SELECT salary FROM works WHERE company_name='Small Bank Corporation');

(d)
SELECT company_name FROM company AS A WHERE NOT EXISTS
(SELECT B.city FROM company AS B WHERE B.company_name='Small Bank Corporation'
AND B.city NOT IN (SELECT C.city FROM company AS C

```
        WHERE A.company_name=C.company_name)
);
```

(e)

```
SELECT company_name FROM works GROUP BY company_name HAVING COUNT(ID) >= ALL (SELECT COUNT(ID) FROM works GROUP BY company_name);
```

(f)

```
SELECT company_name FROM works GROUP BY company_name HAVING AVG(salary) > (SELECT AVG(salary) FROM works WHERE company_name='First Bank Corporation');
```

(g)

```
UPDATE employee SET city='Newtown' WHERE ID=12345;
```

(h)

```
SELECT A.ID, A.person_name FROM employee AS A, works AS B, company AS C WHERE A.ID=B.ID AND B.company_name=C.company_name AND A.city=C.city;
```

(i)

```
WITH avg_salary(company_name, avg_sal) AS (
        SELECT company_name, AVG(salary) FROM works GROUP BY company_name
)
SELECT ID, person_name FROM employee AS A, works AS B, avg_salary AS C WHERE A.ID=B.ID AND B.company_name=C.company_name AND B.salary > C.avg_sal;
```

(j)

```
SELECT company_name FROM works GROUP BY company_name HAVING SUM(salary) <= ALL (SELECT SUM(salary) FROM works GROUP BY company_name);
```

(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.

(a)

COUNT(course_id)
51
49

SELECT COUNT(course_id) FROM section GROUP BY semester;

(b)

COUNT(DISTINCT title)
133

SELECT COUNT(DISTINCT title) FROM course;

(c)

ROUND(AVG(salary/12), 2)
8028.88

SELECT ROUND(AVG(salary/12), 2) FROM instructor WHERE dept_name='Cybernetics';

(d)

dept_name
Finance
Physics

SELECT dept_name FROM department WHERE budget > (SELECT budget FROM department WHERE dept_name='Psychology') ORDER BY dept_name;

(e)

name
Collet
Chakraborty
Cacciari

SELECT name FROM student WHERE dept_name='Geology' AND name LIKE 'C%';

(f)

ID	name
14023	Deshpande
18941	Denecker
29002	Duxbury
3739	Davy
53165	Dowey
62487	Durrant
78552	Douss
89051	Dubink

SELECT A.ID, A.name FROM student AS A WHERE A.dept_name='History' AND name LIKE 'D%' AND A.ID NOT IN (SELECT ID FROM takes AS B, course AS C WHERE B.course_id=C.course_id AND dept_name='Music' GROUP BY ID HAVING COUNT(B.course_id) >= 5);

(g)

ID	name	dept_name	tot_cred
94998	Krishnakumar	Physics	81

SELECT * FROM student WHERE (dept_name='Physics' OR dept_name='Comp. Sci.') AND name LIKE '__________%';

(h)

COUNT(*)
99

SELECT COUNT(*) FROM student WHERE dept_name='Comp. Sci.' and tot_cred > SOME (SELECT tot_cred FROM student WHERE dept_name='English');

(i)

building
Saucon
Stabler
Taylor
Whitman

SELECT building FROM classroom GROUP BY building HAVING SUM(capacity) > 100;

(j)

ID
15347
25946
42782
73623
80759
90643

SELECT A.ID FROM instructor AS A, teaches AS B WHERE A.ID=B.ID AND
EXISTS (SELECT C.ID FROM teaches AS C WHERE C.year <= 2003 AND A.ID=C.ID)
AND NOT EXISTS (SELECT C.ID FROM teaches AS C WHERE C.year > 2003 AND A.ID=C.ID);

(k)

SELECT dept_name, COUNT(*) AS num_students FROM student GROUP BY dept_name ORDER BY num_students DESC;

(l)

WHERE title in (SELECT title FROM course GROUP BY title HAVING COUNT(title)=1)

7.

(a)

SELECT DISTINCT course_id, sec_id, semester, year, COUNT(ID) AS num_students FROM takes GROUP BY course_id, sec_id, semester, year;

(b)

course_id	ID
362	16480
362	16969
362	27236
362	39925
362	39978
362	44881
362	49611
362	5414
362	69581
362	9993

SELECT DISTINCT course_id, ID FROM takes GROUP BY ID, course_id HAVING COUNT(course_id) >= 3 ORDER BY course_id;

(c)

ID	name
37687	Arias
95030	Arinb
52647	Bancilhon
97302	Bertolino
59795	Desyl
58558	Dusserre
64871	Gutierrez
57180	Hau
74426	Kenje
50885	Konstantinides
79653	Levine
63395	McKinnon
96895	Mird
31955	Moreira
4034	Murata
78699	Pingr
35579	Soisalon-Soi…
16807	Yazdi
72553	Yin

SELECT A.ID, name FROM instructor AS A WHERE A.ID NOT IN (SELECT B.ID FROM teaches AS B, takes AS C WHERE B.course_id=C.course_id AND grade='A ') ORDER BY name;

(d)

name

SELECT A.name FROM instructor AS A WHERE NOT EXISTS
(SELECT course_id FROM course AS B WHERE A.dept_name=B.dept_name
AND course_id NOT IN (SELECT C.course_id FROM teaches AS C WHERE A.ID=C.ID))
ORDER BY A.name DESC;

(e)

For example, assume that there are values of 100, 50, 73, and NULL for 'salary' in the *instructor* table. At this time, the average obtained using the AVG() function is 74.333 and the average obtained using the (SUM(salary)/COUNT(*)) is 55.7500. In this case, the result of 'SELECT AVG(salary) – (SUM(salary)/COUNT(*)) FROM instructor;' is 18.5833. This difference occurs because the AVG() function ignores NULL value when calculating the average. In other words, in the above example, AVG() calculates only the mean for three non-null values, but the mean obtained by adding all the values and dividing by the total number of values calculates the mean for four values, so the result value of the query is not zero.