Find the number of usernames under each real name

```
SELECT  UserName, Count(*)
FROM    Author
GROUP BY  Author.RealName
```

Find the number of articles written by each user, group by real names

```
SELECT  RealName, Count(*)
FROM    Author, Wrote
WHERE   Author.RealName = Wrote.UserName
GROUP BY  Author.RealName
```

Find the real names of the persons who wrote more than 4 articles

```
SELECT  RealName
FROM    Author AS A, Wrote AS W
WHERE   A.RealName = W.UserName
GROUP BY  A.RealName
HAVING  Count(*) >= 4
```

Player (#P, PName, Ranking, Age)
Court (#C, Type, Location)
Reserves (#P, #C, Hours, Date)

Get the names of the players who have reserved courts of 'Grass' type.

```
SELECT  PName
FROM    Player AS P, Reserves AS R
WHERE   P.#P = R.#P AND
        R.#C = (SELECT #C
                FROM   Court
                WHERE  Court.Type = 'Grass')
```

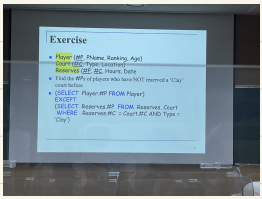
Find the #P of the players who have reserved 'Grass' courts but not 'Sand' courts.

```
(SELECT #P
 FROM   Reserves AS R, Court AS C
 WHERE  R.#C = C.#C AND
        C.Type = 'Grass')
 INTERSECT
(SELECT #P
 FROM   Reserves, Court
 WHERE  Reserves.#C = Court.#C AND
        C.Type = 'Sand')
```

Find the #Ps of players who have a ranking of 2 or who have reserved court with #C=5

```
(SELECT #P
 FROM   Player
 WHERE  Ranking = 2)
 UNION
(SELECT #P
 FROM   Reserves
 WHERE  #C = 5)
```

Find the #Ps of players who have NOT reserved a 'Clay' court before




```
SELECT  R.#P
FROM    Reserves AS R
WHERE   NOT EXIST ( SELECT *
                    FROM   Court AS C
                    WHERE  R.#C = C.#C AND
                           C.Type = 'Clay' )
```

Find #Ps of players who have reserved each court at least once

```
SELECT  R.#P AS #P
FROM    Reserves AS R
GROUP BY R.#P
HAVING  COUNT ( DISTINCT #C )
        = ( SELECT Count(*)
            FROM   Court )
```

6.1. How do the relations (tables) in SQL differ from the relations defined formally in Chapter 3? Discuss the other differences in terminology. Why does SQL allow duplicate tuples in a table or in a query result?

6.10. Specify the following queries in SQL on the COMPANY relational database schema shown in Figure 5.5. Show the result of each query if it is applied to the COMPANY database in Figure 5.6.

a. Retrieve the names of all employees in department 5 who work more than 10 hours per week on the ProductX project.

```
SELECT  E.Fname, Minit, Lname
FROM    EMPLOYEE AS E, WORKS_ON AS W
WHERE   E.Dno = 5 AND
        E.Ssn = W.ESSN AND
        W.Hours >= 10 AND
        W.Pno = (SELECT Pnumber AS Pno
                 FROM   PROJECT AS P
                 WHERE  P.Pname = 'ProductX')
```

b. List the names of all employees who have a dependent with the same first name as themselves.

```
SELECT  FName, Minit, Lname
FROM    EMPLOYEE AS E, DEPENDENT AS D
WHERE   E.Ssn = D.Essn AND
        E.Fname = D.Dependent_name
```

c. Find the names of all employees who are directly supervised by 'Franklin Wong'.

```
SELECT  FName, Minit, Lname
FROM    EMPLOYEE AS E1, EMPLOYEE AS E2
WHERE   E1.Super_ssn = E2.Ssn AND
        E2.Fname = 'Franklin' AND
        E2.Lname = 'Wong'
```

6.12. Specify the following queries in SQL on the database schema of Figure 1.2.

a. Retrieve the names of all senior students majoring in 'cs' (computer science).

```
SELECT  Name
FROM    STUDENT
WHERE   Major = 'CS' AND  Class = '4'
```

b. Retrieve the names of all courses taught by Professor King in 2007 and 2008.

```
SELECT  Course_name
FROM    COURSE AS C, SECTION AS S
WHERE   C.Course_number = S.Course_number AND
        S.Year = '07' OR S.Year = '08' AND
        S.Instructor = 'King'
```

c. For each section taught by Professor King, retrieve the course number, semester, year, and number of students who took the section.

```
SELECT  S.Course_number, S.Semester, S.Year, Count(*)
FROM    SECTION AS S, GRADE_REPORT AS G
WHERE   S.Section_identifier = G.Section_identifier
GROUP BY  S.Section_identifier
```

d. Retrieve the name and transcript of each senior student (Class = 4) majoring in CS. A transcript includes course name, course number, credit hours, semester, year, and grade for each course completed by the student.

```
SELECT  Name, Course_name, Course_number, Credit_hours, Semester,
        Year, Grade
FROM    (SELECT  Name, Grade, Section_identifier
         FROM    STUDENT AS SU,
                 GRADE_REPORT AS G
         WHERE   SU.Student_number = G.Student_number AND
                 SU.Class = 4 AND SU.Major = 'CS'
         GROUP BY  SU.Student_number) AS BASE,
        COURSE AS C, SECTION AS S
WHERE   BASE.Section_identifier = S.Section_identifier AND
        S.Course_number = C.Course_number
```

6.13. Write SQL update statements to do the following on the database schema shown in Figure 1.2.

a. Insert a new student, <'Johnson', 25, 1, 'Math'>, in the database.

```
INSERT INTO STUDENT
VALUES ('Johnson', 25, 1, 'Math')
```

b. Change the class of student 'Smith' to 2.

```
UPDATE STUDENT
SET    Class = 2
WHERE  Name = 'Smith'
```

c. Insert a new course, <'Knowledge Engineering', 'cs4390', 3, 'cs'>.

```
INSERT INTO COURSE
VALUES ('Knowledge Engineering', 'cs4390', 3, 'cs')
```

d. Delete the record for the student whose name is 'Smith' and whose student number is 17.

```
DELETE FROM STUDENT
WHERE  Name = 'Smith' AND
       Student_number = 17
```

Query 18. Retrieve the names of all employees who do not have supervisors.

```
SELECT  Name
FROM    EMPLOYEE
WHERE   Super_Ssn IS NULL
```

7.5. Specify the following queries on the database in Figure 5.5 in SQL. Show the query results if each query is applied to the database state in Figure 5.6.

a. For each department whose average employee salary is more than $30,000, retrieve the department name and the number of employees working for that department.

```
SELECT  Dname , Count(*)
FROM    DEPARTMENT AS D , EMPLOYEE AS E
WHERE   E.Dno = D.Dnumber
GROUP BY  E.Dno
HAVING  AVG(Salary) > 30000
```

b. Suppose that we want the number of *male* employees in each department making more than $30,000, rather than all employees (as in Exercise 7.5a). Can we specify this query in SQL? Why or why not?

```
SELECT  Dno, Count(*)
FROM    EMPLOYEE AS E
WHERE   E.Sex = 'M'
GROUP BY  Dno
Having  AVG(Salary) > 30000
```

7.6. Specify the following queries in SQL on the database schema in Figure 1.2.

a. Retrieve the names and major departments of all straight-A students (students who have a grade of A in all their courses).

```
SELECT  Name, Major
FROM    STUDENT AS S,
WHERE   NOT EXIST ( SELECT *
                    FROM GRADE_REPORT AS G
                    WHERE S.Snum = G.Snum
                      AND Grade ≠ 'A' )
```

b. Retrieve the names and major departments of all students who do not have a grade of A in any of their courses.

```
SELECT  Name, Major
FROM    STUDENT AS S
WHERE   NOT EXIST ( SELECT *
                    FROM GRADE_REPORT AS G
                    WHERE S.Student_Number = G.Snum
                      AND Grade = 'A' )
```

Count() 없이 query 동일!

7.7. In SQL, specify the following queries on the database in Figure 5.5 using the concept of nested queries and other concepts described in this chapter.

a. Retrieve the names of all employees who work in the department that has the employee with the highest salary among all employees.

```
SELECT  FName, LName
FROM    EMPLOYEE
GROUP BY  Dno
HAVING  MAX(Salary) = ( SELECT MAX(Salary)
                        FROM  EMPLOYEE )
```

b. Retrieve the names of all employees whose supervisor's supervisor has '888665555' for Ssn.

```
SELECT  E1.Fname, E1.Lname
FROM    EMPLOYEE E1
WHERE   E1.Super_Ssn IN ( SELECT  Ssn
                          FROM  EMPLOYEE AS E2
                          WHERE E2.Super_Ssn
                            = '888665555' )
```

c. Retrieve the names of employees who make at least $10,000 more than the employee who is paid the least in the company.

```
SELECT  FName, LName
FROM    EMPLOYEE AS E1
WHERE   E1.Salary >= 10000 + SELECT  MIN(Salary)
                             FROM  EMPLOYEE
```