Lecture 5 SQL-Advance

CS211 - Introduction to Database

Sample Database

- Student(sID char(4), sName char(10), gender char(1), age int, dID char(2), grade char(10))
- Dept(dID char(2), dName char(10), dean char(10))
- Course(cID char(3), cName char(12), hours int, credit int, iID char(3))
- Instructor(iID int, iName char(10), dID char(2), workload float)
- RC(sID char(8), cID char(3), score float)

Subquery

- To determine
  - if an element is in a set — (not) in
  - if a set is the subset of another one — Θ some / Θ all
  - If a set is empty — (not) exists
  - If there exist duplicate tuple in a set

Subquery – (not) in

Determine if the value of expression is in the set returned by a subquery

expression [not] in subquery

```
SELECT * FROM Student WHERE sName IN ('Amy', 'John', 'Ritch');
```

```
SELECT sID, sName FROM Student
WHERE  sID IN (SELECT sID FROM rc WHERE cID = '211');
```

- Find students(sID) who never took any course lectured by ‘Prabhu'

sID	sName	gender	age	dID	grade
4001	Amy	F	20	03	Sophomore
4002	Alice	F	19	04	Freshman
4003	Bob	M	20	03	Sophomore
4004	Cathy	F	18	04	Freshman
4005	John	M	21	03	Junior

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

iID	iName	dID	workload
101	Jeremy	03	2.5
102	Vadim	03	2
103	Prabhu	03	1.5
104	Liu Fang	03	1.5

cID	cName	hours	credit	iID
211	Database	56	3	104
225	C++	56	4	102
228	OS	56	3	103

- Find students(sID) who never took any course lectured by 'Prabhu '

```
SELECT sID FROM Student WHERE sID NOT IN (
   SELECT sID FROM rc r, course c, instructor i
   WHERE i.iName = 'Prabhu' AND i.iID = c.iID AND r.cID=c.cID ) ;
```

sID	sName	gender	age	dID	grade
4001	Amy	F	20	03	Sophomore
4002	Alice	F	19	04	Freshman
4003	Bob	M	20	03	Sophomore
4004	Cathy	F	18	04	Freshman
4005	John	M	21	03	Junior

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

iID	iName	dID	workload
101	Jeremy	03	2.5
102	Vadim	03	2
103	Prabhu	03	1.5
104	Liu Fang	03	1.5

cID	cName	hours	credit	iID
211	Database	56	3	104
225	C++	56	4	102
228	OS	56	3	103

Inner query & outer query

Outer Query

```
SELECT sID, sName
FROM Student
WHERE  sID in
        (SELECT sID
         FROM rc
         WHERE cID = '211');
```

Inner Query

Correlated query

Inner query uses values from the outer query

```
SELECT sID, sName
FROM Student
WHERE  sID in
        (SELECT sID
         FROM rc
         WHERE sID = Student.sID);
```

One way:
out→inner

Subquery – Θ some / Θ all

Determine if the value of expression satisfies the result of subquery

expression Θ some subquery (exist)

expression Θ all subquery (for all)

iID	iName	dID	workload
101	Jeremy	03	2.5
102	Vadim	03	2
103	Prabhu	03	1.5
104	Liu Fang	03	1.5

```
SELECT iName FROM Instructor
WHERE  workload <= ALL (SELECT workload FROM instructor);
```

{2.5 2 1.5 1.5}

- Find students(sID) whose score of '211' is not the top1.

- Find students(name) who fails all classes. (correlated)

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

- Find students(sID) whose score of '211' is not the top1.

```
SELECT sID FROM rc WHERE cID='211' AND score <
SOME ( SELECT r.score FROM rc r WHERE r.cID='211' ) ;
```

- Find students(sID) who fails all classes. (correlated)

```
SELECT sID FROM rc WHERE 60 > ALL (
   SELECT r.score FROM rc r WHERE r.sID=rc.sID ) ;
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

- Find the course that Amy got the lowest score (among all courses that Amy has taken)

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

- Find the course(cID) that Amy got the lowest score (among all courses that Amy has taken)

```
SELECT cID FROM rc, student
WHERE rc.sID = student.sID AND
        student.sName = 'Amy' AND
        rc.score <= ALL (
        SELECT r.score FROM rc r WHERE r.sID=rc.sID ) ;
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

expression in subquery **is equivalent to** expression =some subquery

```
SELECT sName FROM Student S
WHERE  sID IN (SELECT sID FROM rc
               WHERE sID=S.sID and cID='211');
```

```
SELECT sName FROM Student S
WHERE  sID = SOME (SELECT sID FROM rc
                   WHERE sID=S.sID and cID='211');
```

expression not in subquery **is not equivalent to** expression <> some subquery

expression not in subquery **is equivalent to** expression <> all subquery

```
SELECT sName FROM Student S
WHERE  sID NOT IN (SELECT sID FROM rc WHERE cID='211');
```

=

```
SELECT sName FROM Student S
WHERE  sID <> ALL (SELECT sID FROM rc WHERE cID='211');
```

≠

```
SELECT sName FROM Student S
WHERE  sID <> SOME (SELECT sID FROM rc WHERE cID='211');
```

Subquery – (not) exists

Determine if there exists tuples in the result of subquery

[not] exists subquery

```
SELECT sName FROM Student
WHERE NOT EXISTS (SELECT * FROM rc,course,instructor
                  WHERE rc.cID = course.cID and
                        course.iID = instructor.iID and
                        instructor.iName = 'Liu Fang' and
                         rc.sID = student.sID);
```

Another implementation

- except: to implement $R - S$

```
Select sname from student except
(select student.sname from rc, course, instructor,student
where rc.cid=course.cid and rc.sid = student.sID and
course.iid=instructor.iid and instructor.iname='liu fang');
```

Select expr|agfunc()

iID	iName	dID	workload
101	Jeremy	03	2.5
102	Vadim	03	2
103	Prabhu	03	1.5
104	Liu Fang	03	1.5

- Find difference of workload (<> 0) of any two instructors

```
SELECT i1.iName, i2.iName, i1.workload-i2.workload
FROM instructor i1, instructor i2
WHERE i1.workload > i2.workload ;
```

- Calculate the birth year for each student (this year 2021)

```
SELECT s.sID, s.sName, 2021-s.age+1 as birthYear
FROM Student s ;
```

```
SELECT abs(-3) ;
```

Aggregation Function

	Argument type	Return type	Description
Count	Any(can be *)	Numeric	Count of occurrences
Sum	Numeric	Numeric	Sum of arguments
Avg	Numeric	Numeric	Average of arguments
Max	Char or Numeric	Same as arg	Maximum value
Min	Char or Numeric	Same as arg	Minimum value

```
SELECT sum(workload) FROM instructor;
```

```
SELECT avg(score) FROM rc WHERE rc.cID='211';
```

- Calculate average score for each class

```
SELECT cID, avg(score)
FROM rc
GROUP BY cID
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

Select from (subquery)

- Calculate average over all students
  - Calculate average over all registered courses for each student
  - Calculate average over all students

```
SELECT avg(avgS)
FROM (SELECT avg(score) as avgS
        FROM rc
        GROUP BY rc.sID) avgStu;
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

- Return students(ID) who fails more than two classes

```
SELECT sID
FROM rc
WHERE score<60 and count(*)>2
GROUP BY sID
```

×

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

- Aggregation function does not appear in where-clause

Having

- Return students(ID) who fails more than two classes
- Must with GROUP BY

```
SELECT sID
FROM rc
WHERE score<60
GROUP BY sID HAVING count(*)>2
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

```
SELECT sID
FROM rc
WHERE score<60
HAVING count(*)>2 GROUP BY sID
```

×

- Find students(ID) who fails more than two classes, and calculate their average score on all courses

```
SELECT sID, avg(score)
FROM rc
WHERE sID in (SELECT sID
              FROM rc
              WHERE score<60
              GROUP BY sID
              HAVING count(*)>2)
GROUP BY sID
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

Union

- Return students(ID) takes 211 or 225

```
SELECT sID FROM rc WHERE cID='211'
UNION
SELECT sID FROM rc WHERE cID='225'
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

Union

- Customers(cID, cName,city)
- Agents(aID, aName,city)

- Find cities customers or agents is in

```
SELECT city FROM Customers
UNION
SELECT city FROM Agents
```

Intersect

- Return students(ID) takes both 211 and 225

```
SELECT sID FROM rc WHERE cID='211'
INTERSECT
SELECT sID FROM rc WHERE cID='225'
```

```
SELECT sID FROM rc
WHERE cID='211' AND
    sID IN (SELECT sID FROM rc WHERE cID='225')
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

Except

- Return students(ID) who doesn't take 211

```
SELECT sID FROM rc WHERE cID<>'211'
```

✕

```
SELECT DISTINCT sID FROM rc
EXCEPT
SELECT sID FROM rc WHERE cID='211'
```

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89

```
SELECT DISTINCT sID FROM rc
WHERE NOT EXISTS (
    SELECT * FROM rc r1 WHERE cID='211' and r1.sID=rc.sID)
```

NULL

- IS [NOT] NULL
- Return NULL for an arithmetic operation
- Return false or unknown for a comparison operation

- count(*) counts it
- however count(score) ignores it
- Other aggregation function ignores it

sID	cID	score
4001	211	87.5
4001	225	94
4001	228	78
4002	211	92
4003	225	65
4003	228	74
4004	211	88
4004	225	82.4
4005	225	86.7
4005	228	89
9999	999	*NULL*

join

- natural join
- [inner] join on <condition>
- left join on <condition>
- right join on <condition>
- full outer join on <condition>