SQL – Data Manipulation Language

Carla Teixeira Lopes

Bases de Dados
Mestrado Integrado em Engenharia Informática e Computação, FEUP

Based on Jennifer Widom slides

Intersect operator

```
SELECT sID FROM Apply WHERE major='CS'
INTERSECT
SELECT sID FROM Apply WHERE major='EE';
```

sID
123
345

Some DBMS don't support the intersect operator

How to rewrite this query without the intersect operator?

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Intersect operator

```
SELECT A1.sID
FROM Apply A1, Apply A2
WHERE A1.sID = A2.sID AND A1.major = 'CS' AND A2.major = 'EE';
```

Duplicates are not eliminated now

sID
123
123
123
123
345

How to eliminate them?

```
SELECT DISTINCT A1.sID
FROM Apply A1, Apply A2
WHERE A1.sID = A2.sID AND A1.major = 'CS' AND A2.major = 'EE';
```

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Except operator

It's called *difference* in Relational Algebra

SELECT sID FROM Apply WHERE major='CS'
EXCEPT
SELECT sID FROM Apply WHERE major='EE';

sID
543
876
987

Some DBMS don't support the except operator

How to rewrite this query without the except operator?

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Except operator

```
SELECT DISTINCT A1.sID
FROM Apply A1, Apply A2
WHERE A1.sID=A2.sID AND A1.major='CS' AND A2.major<>'EE';
```

Finding one pair that satisfies this doesn't mean there's not another pair with the same student where he applied to 'CS' and 'EE'

This query is finding students who applied to 'CS'

sID
123
345
543
876
987

Not possible to rewrite the except query with the operators seen until now

Agenda

~~Introduction~~

~~Basic SQL Statement~~

~~Table Variables and Set Operators~~

Subqueries in WHERE clauses

Subqueries in FROM and SELECT clauses

The JOIN family of operators

Aggregation

Null values

Data Modification statements

Subqueries in WHERE

SELECT $A_1, A_2, \ldots, A_n$

FROM $R_1, R_2, \ldots, R_m$

WHERE condition → Expressions involving subqueries

Subqueries are nested SELECT statements

Subqueries generate sets that are used for comparison

A first example

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

```
SELECT  sID, sName
FROM    Student
WHERE   sID in
        (SELECT sID FROM Apply WHERE major='CS');
```

sID	sName
123	Amy
345	Craig
987	Helen
876	Irene
543	Craig

Subquery that finds the IDs of students who have applied to a major in CS

IDs and names of students who have applied to a major in CS at some college

How can we do this query without the subquery?

A first example

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

```
SELECT  DISTINCT Student.sID, sName
FROM    Student, Apply
WHERE   Student.sID = Apply.sID and major='CS';
```

sID	sName
123	Amy
345	Craig
987	Helen
876	Irene
543	Craig

Could we write sID instead of Student.sID?

Why is DISTINCT necessary here and not in the previous query?

A second example - duplicates

```
SELECT  sName
FROM    Student
WHERE   sID in
        (SELECT sID FROM Apply WHERE major='CS');
```

sName
Amy
Craig
Helen
Irene
Craig

Names of students who have applied to a major in CS at some college

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

A second example - duplicates

```
SELECT  DISTINCT sName
FROM    Student, Apply
WHERE   Student.sID = Apply.sID and major='CS';
```

sName
Amy
Craig
Helen
Irene

Why is the result different from the one in the previous query?

The two different Craigs turned in one result

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Third example - duplicates

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Can we list the GPA of students who have applied to a major in CS at some college using a subquery?

```
SELECT  GPA
FROM    Student
WHERE   sID in
            (SELECT sID FROM Apply WHERE major='CS');
```

GPA
3.9
3.5
3.7
3.9
3.4

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

Third example - duplicates

Can we list the GPA of students who have applied to a major in CS at some college **without** using a subquery?

No

```
SELECT  DISTINCT GPA
FROM    Student, Apply
WHERE   Student.sID = Apply.sID and major='CS';
```

GPA
3.9
3.5
3.7
3.4

The same problem as in previous example

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

Fourth example

Can we now write a query to list the students that have applied to a major in CS but have not applied to a major in EE, without using the except operator?

```
SELECT sID, sName
FROM Student
WHERE sID in (select sID from Apply where major='CS') AND
      sID not in (select sID from Apply where major='EE');
```


Equivalent

```
SELECT sID, sName
FROM Student
WHERE sID in (select sID from Apply where major='CS') AND
      not sID in (select sID from Apply where major='EE');
```

sID	sName
987	Helen
876	Irene
543	Craig

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

Exists operator

Checks whether a subquery is empty or not

Write a query that finds all colleges, such that there's some other college that is in the same state

cName	state	enr
Stanford	CA	15000
Berkeley	CA	36000
MIT	MA	10000
Cornell	NY	21000

cName	state
Stanford	CA
Berkeley	CA

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Exists operator

```
SELECT cName, state
FROM College C1
WHERE exists (select * from College C2
              where C2.state=C1.state);
```

Correlated reference

cName	state
Stanford	CA
Berkeley	CA
MIT	MA
Cornell	NY

→ What's the problem?

↓

Every college is in the same state as itself

Exists operator

```
SELECT cName, state
FROM College C1
WHERE exists (select * from College C2
              where C2.state=C1.state and C1.cName<>C2.cName);
```

cName	state
Stanford	CA
Berkeley	CA

College(<u>cName</u>, state, enr)

Student(<u>sID</u>, sName, GPA, sizeHS)

Apply(<u>sID</u>, <u>cName</u>, <u>major</u>, decision)

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Getting the largest value

Find the college that has the largest enrollment

```
SELECT cName
FROM College C1
WHERE not exists (select * from College C2
                  where C2.enr > C1.enr);
```

Find all colleges where there does not exist another college whose enrollment is higher than the first college

cName
Berkeley

To look for something that's the largest or the smallest

Getting the largest value – example 2

Find the student with the highest GPA

```
SELECT sName
FROM Student C1
WHERE not exists (select * from Student C2
                  where C2.GPA > C1.GPA);
```

sName
Amy
Doris
Irene
Amy

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

Getting the largest value – example 2

Can we find the student with the highest GPA without subqueries?

```
SELECT S1.sName, S1.GPA
FROM Student S1, Student S2
WHERE S1.GPA > S2.GPA;
```

Does this work?

Cannot do it using joins

Finds all students such that there is some other student whose GPA is lower, that is, all students except those who have the lowest GPA

College(<u>cName</u>, state, enr)

Student(<u>sID</u>, sName, GPA, sizeHS)

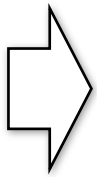
Apply(<u>sID</u>, <u>cName</u>, <u>major</u>, decision)

Alternatives to Intersect and Except

```
(SELECT R.A, R.B
 FROM   R)

INTERSECT

(SELECT S.A, S.B
 FROM   S)
```

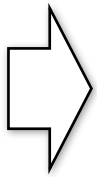


```
SELECT R.A, R.B
FROM   R
WHERE EXISTS(
            SELECT *
            FROM S
      WHERE R.A=S.A AND R.B=S.B)
```

```
(SELECT R.A, R.B
 FROM   R)

EXCEPT

(SELECT S.A, S.B
 FROM   S)
```



```
SELECT R.A, R.B
FROM   R
WHERE NOT EXISTS(
            SELECT *
      FROM S
      WHERE R.A=S.A AND R.B=S.B)
```

All operator

Checks whether the value has a certain relationship with **all** the results of the subquery

```
SELECT sName
FROM Student
WHERE GPA >= all (select GPA from Student);
```

sName
Amy
Doris
Irene
Amy

College(cName, state, enr)

Student(sID, sName, GPA, sizeHS)

Apply(sID, cName, major, decision)

All operator

```
SELECT sName
FROM Student S1
WHERE GPA > all (select GPA from Student S2
                 where S2.sID <> S1.sID);
```

sName

Does this work?

The query would be correct if we knew that every student's GPA was unique

College(<u>cName</u>, state, enr)

Student(<u>sID</u>, sName, GPA, sizeHS)

Apply(<u>sID</u>, <u>cName</u>, <u>major</u>, decision)

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Any operator

Checks whether the value has a certain relationship with **at least one** element of the results of the subquery

```
SELECT sName
FROM Student
WHERE not GPA < any (select GPA from Student);
```

$$\forall_{x \in X} \neg P(x) \equiv \neg \exists_{x \in X} P(x)$$

```
SELECT sName
FROM Student
WHERE GPA>= all (select GPA from Student);
```

sName
Amy
Doris
Irene
Amy

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Any operator – example 2

Find all students who are not from the smallest high school in the database

```
SELECT sID, sName, sizeHS
FROM Student
WHERE sizeHS > any (select sizeHS from Student);
```

SQLite does not support the any and all operators

Have to rewrite the queries using exists and not exists

How can we rewrite this query using exists?

sID	sName	HS
123	Amy	1000
234	Doris	1500
345	Irene	500
456	Amy	1000
567	Edward	2000
789	Gary	800
987	Helen	800
876	Irene	400
765	Jay	1500
654	Amy	1000
543	Craig	2000

Any operator – example 2

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

```
SELECT sID, sName, sizeHS
FROM Student S1
WHERE exists (select * from Student S2
                  where S2.sizeHS < S1.sizeHS);
```

An *any* or *all* query can always be written using *exists* or *not exists*

sID	sName	HS
123	Amy	1000
234	Doris	1500
345	Irene	500
456	Amy	1000
567	Edward	2000
789	Gary	800
987	Helen	800
876	Irene	400
765	Jay	1500
654	Amy	1000
543	Craig	2000

Any operator – example 3

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Can we rewrite the query that finds students who have applied to a major in CS and have not applied to a major in EE?

```
SELECT sID, sName
FROM Student
WHERE sID in (select sID from Apply where major='CS') AND
      sID not in (select sID from Apply where major='EE');
```

sID	sName
987	Helen
876	Irene
543	Craig

```
SELECT sID, sName
FROM Student
WHERE sID = any (select sID from Apply where major = 'CS')
AND sID <> any (select sID from Apply where major = 'EE');
```

Satisfied as long as there's anybody who applied to EE that is not the same as the student we're looking at

sID	sName
123	Amy
345	Craig
987	Helen
876	Irene
543	Craig

College(cName, state, enr)
Student(sID, sName, GPA, sizeHS)
Apply(sID, cName, major, decision)

Any operator – example 3

```
SELECT sID, sName
FROM Student
WHERE sID = any (select sID from Apply where major = 'CS')
AND sID <> any (select sID from Apply where major = 'EE');
```

sID	sName
123	Amy
345	Craig
987	Helen
876	Irene
543	Craig

How can we correct it?

```
SELECT sID, sName
FROM Student
WHERE sID = any (select sID from Apply where major = 'CS')
AND not sID = any (select sID from Apply where major = 'EE');
```

sID	sName
987	Helen
876	Irene
543	Craig

Kahoot time!

Any doubts?

Readings

Jeffrey Ullman, Jennifer Widom, A first course in Database Systems 3rd Edition

- Section 6.1 – Simple Queries in SQL
- Section 6.2 – Queries Involving More Than One Relation
- Section 6.3 - Subqueries
- Section 6.4 – Full-Relation Operations
- Section 6.5 – Database Modifications

Philip Greenspun, SQL for Web Nerds, http://philip.greenspun.com/sql/