In the Lecture Series Introduction to Database Systems

SQL

Presented by Stéphane Bressan

Nested Queries: IN

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

There can be nested queries using ``IN'', ``=ANY'', ``>ALL'' etc.

outer query

```
SELECT s.name
FROM student s
WHERE email IN
```

Inner query

```
    (SELECT l.owner
    FROM loan l
    WHERE l.returned  > '2010-03-04'
    AND l.borrower = 'anniechapman1991@yahoo.com');
```

name
JENNY HUNT
QIN WEI

Nested Queries: ANY

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

```
SELECT s.name
FROM student s
WHERE s.email =  ANY (SELECT l.owner
   FROM loan l
   WHERE l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com');
```

No ``ANY'' in SQLite

"=" without "ANY" is just comparing with one (or with the first result in SQLite).

``= ANY'' is the same as ``IN''.

Nested Queries: EXISTS

- Is there a book more expensive than 30$ in the catalog?

```
SELECT 'YES'
   FROM catalog c
   WHERE c.price > 30;
```

YES
YES

- Is there a book more expensive than 100$ in the catalog?

```
SELECT 'YES'
   FROM catalog c
   WHERE c.price > 100;
```

No result

- Let us make things more complicated than needed by using EXISTS. This will help us understand how EXISTS works.

Nested Queries: EXISTS

- Is there a book more expensive than 30$ in the catalog?

```
SELECT 'YES'
FROM catalog c1
WHERE EXISTS (SELECT *
   FROM catalog c2
   WHERE c2.price > 30);
```

```
YES
YES
YES
YES
YES
```

EXISTS is true if and only if the inner query has some results. If it has no results, exists is false.

- Is there a book more expensive than 100$ in the catalog?

```
SELECT 'YES'
FROM catalog c1
WHERE EXISTS (SELECT *
   FROM catalog c2
   WHERE c2.price > 100);
```

```
No result
```

- Change `c2.price` into `c1.price` and see what happens…

Nested Queries: EXISTS, Correlated Subqueries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

```
SELECT s.name
FROM student s
WHERE EXISTS(SELECT *
   FROM loan l
   WHERE s.email = l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com');
```

EXISTS is true if and only if the inner query has some results. If it has no results, exists is false.

`s.email` in the inner query is an attribute of the table `student s` in the outer query. The inner query is correlated to the outer query.

name
JENNY HUNT
QIN WEI

Nested Queries (Scope of Correlation)

- An attribute of an outer query can only be used within the SELECT and WHERE clauses of the query in which its relation is declared (FROM clause) and within inner queries (subqueries) queries.

- Attributes of the inner-queries cannot be used in the outer-queries!

Nested Queries: Un-nesting

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

Nested queries can sometimes be rewritten as simple queries. However, the simple query may behave slightly differently with respect to duplicates.

```
SELECT s.name
FROM loan l, student s
WHERE s.email=l.owner
   AND l.returned  > '2010-03-04'
   AND l.borrower = 'anniechapman1991@yahoo.com';
```

name
JENNY HUNT
JENNY HUNT
JENNY HUNT
QIN WEI
QIN WEI

Nested Queries

Find the names of the students from whom anniechapman1991@yahoo.com borrowed a book and returned it after 2010-03-04.

We could express this query with a simple query (but some duplication) and with nested queries using IN, = ANY and EXISTS (with correlation).

Nested Queries: Correlated Subqueries

Find the names of the students in Annie Chapman's (anniechapman1991@yahoo.com) faculty from whom she borrowed a book.

```
SELECT s1.name
FROM student s1
WHERE s1.email IN
       (SELECT l.owner
       FROM loan l, student s2
       WHERE
       l.borrower = 'anniechapman1991@yahoo.com'
       AND l.borrower=s2.email
       AND s1.faculty=s2.faculty);
```

The inner query is correlated with the outer query. The inner query depends on the faculty of the student in the inner query. The inner query cannot be executed alone.

Error: no such column: s1.faculty

name
JENNY HUNT

Nested Queries: ALL

Find the names of the students who were the last ones to return the books that they borrowed

```
SELECT s.name
FROM student s, loan l1
WHERE l1.borrower=s.email
   AND l1.returned >= ALL
       (SELECT l2.returned
       FROM loan l2);
```

"ALL" allows to compare, not to one, but to every result of the subquery.

"ALL" adds **expressive power**: the query above cannot be expressed as a simple (not nested) query.

"ALL" is not supported in SQLite.

Nested Queries: <> ALL

Find the different students from do not own a book

```
SELECT s.email
FROM student s
WHERE s.email  <> ALL (SELECT c.owner
   FROM  copy c);
```

“ALL” is not supported in SQLite.

The nested queries using “<>ALL’’ cannot be rewritten into simple queries.

They add **expressive power**.

“<>ALL’’ is strictly the same as “NOT IN”.

Nested Queries: NOT IN

Find the different students from do not own a book

```
SELECT s.email
FROM student s
WHERE s.email NOT IN (SELECT c.owner
   FROM copy c);
```

The nested queries using “NOT IN” cannot be rewritten into simple queries.

They add **expressive power**.

``<> ALL'' is the same as ``NOT IN''.

email
angjiayi1990@hotmail.com
anniechapman1991@yahoo.com
chnghuiling1992@gmail.com
choyyiting1992@hotmail.com
davidchapman1989@msn.com
dennisbeckham1989@msn.com
…

Nested Queries: NOT EXISTS

Find the different students from whom anniechapman1991@yahoo.com never borrowed

```
SELECT s.email
FROM student s
WHERE NOT EXISTS (SELECT *
   FROM copy c
   WHERE s.email = c.owner);
```

The inner query is correlated to the outer query.

The nested queries using “NOT EXISTS” cannot be rewritten into simple queries.

They add **expressive power**.

“NOT EXISTS” is slightly more general than “NOT IN” but generally equivalent, although it requires correlation.

email
angjiayi1990@hotmail.com
anniechapman1991@yahoo.com
chnghuiling1992@gmail.com
choyyiting1992@hotmail.com
davidchapman1989@msn.com
dennisbeckham1989@msn.com
…

Nested Queries: in the HAVING Clause

We can also use nested queries in the “HAVING” clause in conjunction with “GROUP BY” and aggregate functions.

Find the email of the students who own strictly more copies of books than davidchapman1989@msn.com

```
SELECT l1.borrower
FROM loan l1
GROUP BY l1.borrower
HAVING COUNT(*)>
       (SELECT COUNT(*)
       FROM loan l2
       WHERE l2.borrower='davidchapman1989@msn.com');
```

borrower
anniechapman1991@yahoo.com
dennispalmer1992@yahoo.com

Nested Queries

- There can be multiple nested queries and multiple levels of nested queries
- Nested queries can appear in the WHERE but also the HAVING clauses
- Although it can be done, we shall **never** use nested queries in the SELECT and WHERE clauses unless it is absolutely necessary…

In Summary

- Nested queries are introduced by "IN", "NOT IN", "=ANY", ">ANY", "EXISTS", "NOT EXISTS" etc.
- Nested queries can be used to improve the modularity and readability of queries.
- Nested queries can be used to express queries that could not otherwise be expressed with simple queries.
- Nested queries do not necessarily increase the expressive power.
- Some nested queries are equivalent to queries without nesting. However, with such constructs as “NOT IN” and “NOT EXISTS” the expressive power of SQL is increased.
- Nested queries can be used in the "WHERE“ clause and the "HAVING" clause. Recent versions of SQL allow them in the " SELECT " and " FROM” clauses. We do not allow this!
- There can be multiple levels of nesting.
- Inner queries can be correlated to their outer query. That is attributes of the outer query may be used in the inner queries at any level.

ALGEBRA

UNION, INTERSECT AND EXCEPT

Union

Find all the information about students in the computer science department or in the information systems department.

```
SELECT *
FROM student T
WHERE T.department='CS'
UNION
SELECT *
FROM student T
WHERE T.department='IS';
```

Can you write the same query without UNION?

name	email	year	faculty	department	graduate
ANG JIA YI	angjiayi1990@hotmail.com	8/1/2009	School of Computing	CS	
DAVID CHAPMAN	davidchapman1989@msn.com	1/1/2008	School of Computing	CS	
DENNIS BECKHAM	dennisbeckham1989@msn.com	8/1/2010	School of Computing	IS	
DING WEI XIANG	dingweixiang1990@yahoo.com	1/1/2010	School of Computing	IS	
HUANG XUANTI	huangxuanti1992@msn.com	8/1/2007	School of Computing	IS	
IRIS BROWN	irisbrown1992@hotmail.com	8/1/2008	School of Computing	CS	
...					

Intersection

Find the emails of students in the computer science department owning a book with ISBN14 ‘978-0684801520’.

```
SELECT T1.email
FROM student T1
WHERE T1.department='CS'
INTERSECT
SELECT T2.owner AS email
FROM copy T2
WHERE T2.book='978-0684801520';
```

Can you write the same query without INTERSECT?

email
liushaojun2010@msn.com

(Non-Symmetric) Difference

Find the mails of students in the computer science department except those owning a book with ISBN14 ‘978-0684801520’.

```
SELECT T1.email
FROM student T1
WHERE T1.department='CS'
EXCEPT
SELECT T2.owner AS email
FROM copy T2
WHERE T2.book='978-0684801520';
```

Oracle uses ``MINUS’’

Can you write the same query without EXCEPT?

T1.email
angjiayi1990@hotmail.com
davidchapman1989@msn.com
irisbrown1992@hotmail.com
liuzhencai1990@msn.com
ngookaiting1991@yahoo.com
ngyanfen2010@msn.com
ngyongming2011@yahoo.com
qinyiyang2010@hotmail.com
qinyuwei2011@hotmail.com
siowcaokhoa1991@msn.com

JOINS

AVOID THEM IF YOU CAN…

UNION, INTERSECT and EXCEPT

- UNION, INTERSECT and EXCEPT remove duplicates.

Join

Find the emails of students owning a book with ISBN14 ‘978-0262033848’.

```
SELECT T1.email
FROM student T1, copy T2
WHERE T2.owner=T1.email
AND T2.book='978-0684801520';
```

email
liushaojun2010@msn.com
tayweiguo1989@msn.com

Inner Join

Find the emails of students owning a book with ISBN14 ‘978-0262033848’

```
SELECT T1.email
FROM student T1 INNER JOIN copy T2
ON T2.owner=T1.email
WHERE T2.book='978-0684801520';
```

Why would one want to do that?

Left Outer Join

Find the names of the students and the titles of the books they own. If a student does not own any book, print a NULL value.

```
SELECT T1.name,T2.book
FROM student T1, copy T2
WHERE T1.email=T2.owner
UNION
SELECT T3.name, CAST(NULL AS CHAR(14)) AS book
FROM student T3
WHERE NOT EXISTS (SELECT * FROM copy T4
                  WHERE T3.email=T4.owner);
```

T1.name	T2.book
ANG JIA YI	
ANNIE CHAPMAN	
CHNG HUI LING	
CHOY YI TING	
DAVID CHAPMAN	
DAVID HALL	978-0321474049
DENNIS BECKHAM	
...	

Left Outer Join

Find the names of the students and the different titles of the books they own. If a student does not own any book, print a NULL value.

```
SELECT DISTINCT T1.name, T2.book
FROM student T1 LEFT OUTER JOIN copy T2
ON T1.email=T2.owner;
```

name	book
GOH HUI YING	978-1449389673
TAY WEI GUO	978-0684801520
PENG JIAYUAN	
HUANG ZHANPENG	978-1594487712
ZHENG ZHEMIN	
...	

```
SELECT DISTINCT T1.name, T2.book
FROM student T1 LEFT OUTER JOIN
copy T2
ON T1.email=T2.owner
ORDER BY T1.name;
```

T1.name	T2.book
ANG JIA YI	
ANNIE CHAPMAN	
CHNG HUI LING	
CHOY YI TING	
DAVID CHAPMAN	
DAVID HALL	978-0321474049
DENNIS BECKHAM	
...	

Right Outer Join

Find the title of books and the emails of their owner. If a book does not have an owner , print a NULL value.

```
Select T2.title, T1.owner
FROM copy T1 RIGHT OUTER JOIN book T2
ON T1.book=T2.ISBN14;
```

Error: RIGHT and FULL OUTER JOINs are not currently supported

Full Outer Join

A full outer join will pad both the left and right relations with null values.

Create a `table table(a, b, c)` with 2 or 3 rows.

```
Select DISTINCT T1.a, T2.c
FROM table T1 FULL OUTER JOIN table T2
ON T1.b=T2.b
```

Error: RIGHT and FULL OUTER JOINs are not currently supported

Other Join

- (EQUI) JOIN
- NATURAL JOIN USING
- CROSS JOIN

You will learn more about UNION, INTERSECT, EXCEPT and JOINs in the Relational Algebra lecture.

Summary

1. FROM (JOIN)
2. WHERE
3. GROUP BY
4. HAVING
5. ORDER BY
6. SELECT
7. UNION, INTERSECT, EXCEPT

Credits

The content of this lecture is based on chapter 5 of the book "Introduction to database Systems"

By

S. Bressan and B. Catania, McGraw Hill publisher

Clipart and media are licensed from Microsoft Office Online Clipart and Media