

SQL 3 : Nesting in Where and Having

Gary KL Tam

Department of Computer Science
Swansea University

SQL offers several constructs beyond relational algebra to allow users to write more powerful queries. In the previous lecture, we have studied a collection of constructs for statistical analysis. This lecture will introduce another collection for enhancing the functionality of WHERE and HAVING.

We will first discuss WHERE and then HAVING.

Recall that the where clause contains conditions of the form $A \text{ op } v$ where A is an attribute/value, op is an arithmetic operator (e.g., $<$), and v is another attribute/value.

Next, we will learn new conditions where

- v is an SQL statement
- op is an operator that compares a value to the result of v .

In

(A_1, \dots, A_n) in ([an SQL statement])

where each A_i is an attribute/value. (A_1, \dots, A_n) must obey the schema of the table T returned by the SQL statement.

The expression returns:

- true, if tuple (A_1, \dots, A_n) appears in T .
- false, otherwise.

The bracket of " (A_1, \dots, A_n) " can be omitted if $n = 1$.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where dept in (select dept from PROF where sal >= 10000)

Try:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where dept in (select dept from PROF where sal >= 10000)

Try:

select pid from PROF
 where dept in (CS)

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
where dept in (select dept from PROF where sal >= 10000)

Result:

pid
p1
p3
p6

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where (dept, rank) in (select dept, rank from PROF where sal >= 10000)

Try:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where (dept, rank) in (select dept, rank from PROF where sal >= 10000)

Try:

select pid from PROF
 where (dept, rank) in (CS, full)

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where (dept, rank) in (select dept, rank from PROF where sal >= 10000)

Result:

pid
p3
p6

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where (dept, sal) in (select dept, rank from PROF where sal >= 10000)

Error! (dept, sal) does not match the schema of the table returned by the nested SQL statement.

Not in

(A_1, \dots, A_n) not in ([an SQL statement])

where each A_i is an attribute/value. (A_1, \dots, A_n) must obey the schema of the table T returned by the SQL statement.

The expression returns:

- true, if tuple (A_1, \dots, A_n) does not appear in T .
- false, otherwise.

The bracket of " (A_1, \dots, A_n) " can be omitted if $n = 1$.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select pid from PROF
where (dept, rank) not in
(select dept, rank from PROF where sal >= 10000)
```

Result:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select pid from PROF
where (dept, rank) not in
(select dept, rank from PROF where sal >= 10000)
```

Result:

```
select pid from PROF
where (dept, rank) not in (CS, full)
```

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select pid from PROF
where (dept, rank) not in
(select dept, rank from PROF where sal >= 10000)
```

Result:

pid
p1
p2
p4
p5

Set Comparison 1

Some

$A > \text{some} ([\text{an SQL statement}])$

where A is an attribute/value, and must obey the schema of the table T returned by the SQL statement.

The expression returns:

- true, if A is greater than **at least one** tuple in T .
- false, otherwise.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > some (select sal from PROF where dept = 'CS')

Result:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > some (select sal from PROF where dept = 'CS')

Result:

select pid from PROF
 where sal > some (6000, 10000, 9000)

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > some (select sal from PROF where dept = 'CS')

Result:

pid
p2
p3
p5
p6

All

$A > \text{all} ([\text{an SQL statement}])$

where A is an attribute/value, and must obey the schema of the table T returned by the SQL statement.

The expression returns:

- true, if A is greater than **all** tuples in T .
- false, otherwise.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > all (select sal from PROF where dept = 'EE')

Try:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > all (select sal from PROF where dept = 'EE')

Try:

select pid from PROF
 where sal > all (8000, 5000, 8500)

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF
 where sal > all (select sal from PROF where dept = 'EE')

Result:

pid
p3
p6

The operator $>$ in “ $>$ some (all)” can be replaced with $<$, $<=$, $=$, $<>$ and $>=$.

The semantics in each case agrees with the literal meaning in English. For example, “ $<$ some” means “smaller than some element (in a table)”.

Next, we will learn two more useful conditions of the form:

op ([an SQL statement])

Exists

`exists ([an SQL statement])`

The expression returns:

- true, if the SQL statement returns a table with **at least one** tuple.
- false, otherwise.

`not exists ([an SQL statement])`

The expression returns:

- true, if the SQL statement returns an **empty** table.
- false, otherwise.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select pid from PROF as P
 where not exists
 (select sal from PROF
 where sal > P.sal)

Note

Observe how *P* is used in the nested query.

Result:

pid
 —
 p3

PROF

select pid from PROF as P
where not exists
(select sal from PROF
where sal > P.sal)

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

P

Transient:

select pid from PROF as P
where not exists
(select sal from PROF
where sal > 6000)

pid

PROF

select pid from PROF as P
where not exists
(select sal from PROF
where sal > P.sal)

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

P

Transient:

select pid from PROF as P
where not exists
(select sal from PROF
where sal > 8000)

pid

PROF

select pid from PROF as P
where not exists
(select sal from PROF
where sal > P.sal)

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

P

Transient:

select pid from PROF as P
where not exists
(select sal from PROF
where sal > 10000)

pid
p3

PROF

select pid from PROF as P
where not exists
(select sal from PROF
where sal > P.sal)

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

What is this query looking for in
plain English?

pid
p3

Next we extend the above discussion to HAVING.


```
select  $A_1, \dots, A_t, agg_1(B_1), \dots, agg_m(B_m)$   
from  $T_1, \dots, T_n$   
where  $P$   
group by  $C_1, \dots, C_g$   
having  $H$ 
```

where

- C_1, \dots, C_g are called **group-by attributes**.
- H is a **group predicate**.

The group predicate can contain conditions of the form:

$$\text{agg}(A) \text{ op (SQL statement)}$$

where

- *agg* is an aggregate function
- *op* can be
 - in, not in
 - < some/all
 - <= some/all
 - = some/all
 - <> some/all
 - > some/all
 - >= some/all
- The nested SQL statement must return a table of a single numeric column.

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select dept from PROF
group by dept
having avg(sal) >= some (select avg(sal) from PROF)
```

Transient:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select dept from PROF
group by dept
having avg(sal) >= some (select avg(sal) from PROF)
```

Transient:

```
select dept from PROF
group by dept
having avg(sal) >= some (7750)
```

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select dept from PROF
group by dept
having avg(sal) >= some (7750)
```

Result:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select dept from PROF
 group by dept
 having avg(sal) >= some (7750)

Result:

pid	name	dept	rank	sal	avg(sal)
p1	Adam	CS	asst	6000	8333.33 >= 7750?
p3	Calvin	CS	full	10000	
p6	Frank	CS	full	9000	
p2	Bob	EE	asso	8000	7166.67 >= 7750?
p4	Dorothy	EE	asst	5000	
p5	Emily	EE	asso	8500	

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select dept from PROF
group by dept
having avg(sal) >= some (select avg(sal) from PROF)

Result:

dept
cs

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select dept from PROF
group by dept
having avg(sal) >= some (select sal from PROF)
```

Try:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

```
select dept from PROF
group by dept
having avg(sal) >= some (select sal from PROF)
```

Try:

```
select dept from PROF
group by dept
having avg(sal) >= some (6000, 8000, 10000, 5000, 8500, 9000)
```

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select dept from PROF
 group by dept
 having avg(sal) >= some (6000, 8000, 10000, 5000, 8500, 9000)

Transient:

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select dept from PROF

group by dept

having avg(sal) >= some (6000, 8000, 10000, 5000, 8500, 9000)

Transient:

pid	name	dept	rank	sal	avg(sal)	
p1	Adam	CS	asst	6000	8333.33	>= some(6000, 8000 10000, 5000 8500, 9000) ?
p3	Calvin	CS	full	10000		
p6	Frank	CS	full	9000		
p2	Bob	EE	asso	8000	7166.67	>= some(6000, 8000 10000, 5000 8500, 9000) ?
p4	Dorothy	EE	asst	5000		
p5	Emily	EE	asso	8000		

PROF

pid	name	dept	rank	sal
p1	Adam	CS	asst	6000
p2	Bob	EE	asso	8000
p3	Calvin	CS	full	10000
p4	Dorothy	EE	asst	5000
p5	Emily	EE	asso	8500
p6	Frank	CS	full	9000

select dept from PROF
 group by dept
 having avg(sal) >= some (select sal from PROF)

Result:

dept
cs
ee