Introduction to Databases

Lecture 6

Data Manipulation Language - SELECT

Nga Nguyen

Planning

- Sub-query
- GROUP BY

Sub-query example

```
select label
from product
where price = (select price
                from product
                where lower(label) like '%mango nectar%');
```

Sub-query

- A sub-query may return:
  - a row containing :
    - one value
    - a tuple
  - a list of rows
  - nothing

Sub-query returning one value

```
SELECT name
FROM employee
WHERE post = (SELECT post
              FROM employee
              WHERE name = 'Martin') ;
```

- Return all employees who have the same post as Martin.
  Note : We consider that name is unique.
- Accept all operators : =, <, >, <=, >=, <>

Sub-query on several columns

```
SELECT name
FROM employee
WHERE (post, salary) =
        (    SELECT post, salary
             FROM employee
             WHERE name = 'Martin') ;
```

- Return all employees who have the same post and the same salary as Martin. *Note : We consider that name is unique.*

Sub-query returning a list

- …**WHERE** post **IN** (**SELECT post FROM ..**);
  - The value must be in the result of the sub-query
  - We can use **NOT IN**
- …**WHERE** number **> ALL (SELECT number FROM ..);**
  - The value must be greater than all the values given by the sub-query
- …**WHERE** post **> ANY** (**SELECT post FROM ..**);
  - The value must be greater than at least one value given by the sub-query

Sub-query returning a list

- Operator **IN**, **NOT IN**

- Simple operator **=, <>, <, >, <=, >=** followed by **ALL** or **ANY.**
  - = ANY is equivalent to IN
  - != ALL is equivalent to NOT IN

Sub-query returning a list

- Example

```
SELECT name
FROM customer
WHERE num_customer
IN (SELECT num_customer
   FROM corder
   WHERE odate = '05-JUN-98');
```

Aggregate functions
GROUP BY

Example 1 : compute the average degree for each vintage

Wine	vintage	year	degree	quantity
	CHABLIS	1977	10.9	100
	CHABLIS	1987	11.9	250
	VOLNAY	1977	10.8	400
	VOLNAY	1986	11.2	300
	MEDOC	1985	11.2	200

AVG	vintage	AVG(degree)
	CHABLIS	11.4
	VOLNAY	11.0
	MEDOC	11.2

Example 2 : compute the sum of quantity for each vintage

Wine	vintage	year	degree	quantity
	CHABLIS	1977	10.9	100
	CHABLIS	1987	11.9	250
	VOLNAY	1977	10.8	400
	VOLNAY	1986	11.2	300
	MEDOC	1985	11.2	200

SUM	vintage	SUM(quantity)
	CHABLIS	350
	VOLNAY	700
	MEDOC	200

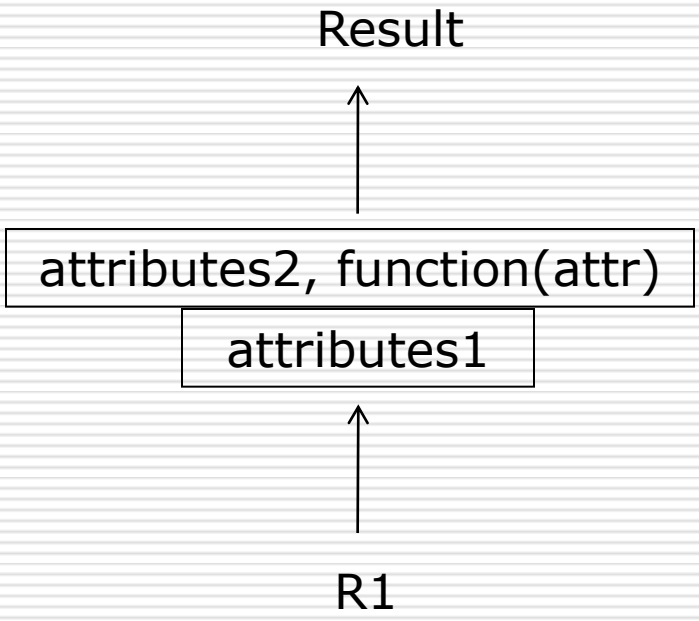
Aggregate functions

- Partition a table horizontally according to the values of a group of attributes called grouping factors
- Follow by a function applied on a column (SUM, MIN, MAX, AVG, COUNT, ...)

SQL GROUP BY syntax

```
SELECT
    <col1>, <col2>, <aggregate_function>(<col3>)
FROM
    ...
WHERE
    ...
GROUP BY <col1>, <col2>;
```

Algebraic tree



GROUP BY : example 1

- Compute the number of films realized by each film maker.

Film(num_film, *num_ind*, title, kind, fyear)

```
SELECT num_ind, COUNT(*)
FROM Film
GROUP BY num_ind;
```

GROUP BY : example 2

- Compute the number of dramatic films realized by each film maker.

Film(num_film, *num_ind*, title, kind, fyear)

```
SELECT num_ind, COUNT(*)
FROM Film
WHERE kind = 'Drama'
GROUP BY num_ind;
```

GROUP BY : HAVING clause

- The HAVING clause enables you to specify conditions that filter which group results appear in the final results.

```
SELECT num_customer, SUM(amount)
FROM corder
GROUP BY num_customer
HAVING SUM(amount) > 1000;
```

- *The WHERE clause places conditions on the selected columns, whereas the HAVING clause places conditions on groups created by the GROUP BY clause.*

WHERE or HAVING ?

- Question 1 : Compute for each cinema the number of projections whose date is prior to '2000-01-01'.

- Question 2 : In which cinemas the number of projections is greater than or equal to 4 ?

  Projection(*num_cine, num_film*, pdate)

WHERE or HAVING ?

- Answer 1 :

```
SELECT num_cine, COUNT(*)
FROM Projection
WHERE pdate < DATE '2000-01-01'
GROUP BY num_cine;
```

- Answer 2 :

```
SELECT num_cine, COUNT(*)
FROM Projection
GROUP BY num_cine
HAVING COUNT(*) >= 4;
```