Principles of Database Systems

Introduction to SQL(4)

Aggregate Functions
(聚集函数)

Aggregate Functions

- Aggregate functions are functions that take a collection (a set or multiset) of values as input and return a single value.
  - Average: **avg**
  - Minimum: **min**
  - Maximum: **max**
  - Total : **sum**
  - Count: **count**

Basic Aggregation

- Example: "Find the average salary of instructors in the Computer Science department."
  - **select avg** (salary)
    from instructor
    where dept_name= 'Comp. Sci.';

  - a meaningful name can be given to the result attribute by using the **as clause**
  - Retaining duplicates is important in computing an average.

Basic Aggregation

- If we do want to eliminate duplicates, we use the keyword **distinct** in the aggregate expression.

  – E.g. “Find the total number of instructors who teach a course in the Spring 2010 semester.”

```
select count (distinct ID)
from teaches
where semester = 'Spring' and year = 2010;
```

Basic Aggregation

- We use the aggregate function **count** frequently to count the number of tuples in a relation.
- The notation for this function in SQL is **count (*)**

```
select count (*)
from course;
```

- SQL does not allow the use of **distinct** with **count (*)**. It is legal to use **distinct** with **max** and **min**, even though the result does not change.

Aggregation with Null Values

- All aggregate operations except count(*) ignore tuples with null values on the aggregated attributes (除了count(*)外所有的聚集函数都忽略输入集合中的空值)

- What if collection has only null values?
  - count returns 0
  - all other aggregates return null

Aggregation with Grouping(分组)

- Apply the aggregate function not only to **a single set of tuples**, but also to **a group of sets of tuples**
  - E.g. “Find the average salary in each department.”

Aggregation with Grouping(分组)

- apply the aggregate function not only to **a single set of tuples**, but also to **a group of sets of tuples**
  - E.g. "Find the average salary in each department."

ID	*name*	*dept_name*	*salary*
76766	Crick	Biology	72000
45565	Katz	Comp. Sci.	75000
10101	Srinivasan	Comp. Sci.	65000
83821	Brandt	Comp. Sci.	92000
98345	Kim	Elec. Eng.	80000
12121	Wu	Finance	90000
76543	Singh	Finance	80000
32343	El Said	History	60000
58583	Califieri	History	62000
15151	Mozart	Music	40000
33456	Gold	Physics	87000
22222	Einstein	Physics	95000

dept_name	*avg_salary*
Biology	72000
Comp. Sci.	77333
Elec. Eng.	80000
Finance	85000
History	61000
Music	40000
Physics	91000

Aggregation with Grouping

- apply the aggregate function not only to **a single set of tuples**, but also to **a group of sets of tuples**

  – E.g. “Find the average salary in each department.”

  select dept_name, **avg** (salary) **as** avg_salary
  from instructor
  group by dept_name;

  – Note: departments with no instructor will not appear in result

Aggregation with Grouping

- Attributes in **select clause** outside of aggregate functions must appear in **group by** list (出现在select子句中但没有被聚集的属性必须出现在group by子句中)
- ```
  /* erroneous query */
  select dept_name, ID, avg (salary)
  from instructor
  group by dept_name;
  ```
  ```

The Having Clause

- Try: Find the names and average salaries of all departments whose **average salary** is greater than 42000

- Can we use the where clause to satisfy the query?

The Having Clause

- Try: Find the names and average salaries of all departments whose **average salary** is greater than 42000

- This query requires restriction on tuple groups rather than tuples, so the where clause is useless in the case. Instead, **having clause** should be used to filter(过滤) tuple groups.

The Having Clause

- Try: Find the names and average salaries of all departments whose **average salary** is greater than 42000

  select dept_name, **avg** (salary)
  from instructor
  group by dept_name
  having avg (salary) > 42000;

- Note: predicates in the **having clause** are applied **after** the formation of groups whereas predicates in the **where clause** are applied **before** forming groups (用在having中的谓词在形成分组后才起作用，而where子句中的谓词在分组前起作用)

The Having Clause

- Try: “For each course section offered in 2009, find the average total credits (tot_cred) of all students enrolled in the section, if the section had at least 2 students.”

Database Used

classroom(*building*, *room_number*, *capacity*)
department(*dept_name*, *building*, *budget*)
course(*course_id*, *title*, *dept_name*, *credits*)
instructor(*ID*, *name*, *dept_name*, *salary*)
section(*course_id*, *sec_id*, *semester*, *year*, *building*, *room_number*, *time_slot_id*)
teaches(*ID*, *course_id*, *sec_id*, *semester*, *year*)
student(*ID*, *name*, *dept_name*, *tot_cred*)
takes(*ID*, *course_id*, *sec_id*, *semester*, *year*, *grade*)
advisor(*s_ID*, *i_ID*)
time_slot(*time_slot_id*, *day*, *start_time*, *end_time*)
prereq(*course_id*, *prereq_id*)

The Having Clause

- Try: “For each course section offered in 2009, find the average total credits (tot_cred) of all students enrolled in the section, if the section had at least 2 students.”

```
select course_id, semester, year, sec_id, avg (tot_cred)
from takes natural join student
where year = 2009
group by course_id, semester, year, sec_id
having count (ID) >= 2;
```

SQL query

- A typical SQL query has the form:

  select $A_1, A_2, \ldots, A_n$
  from $r_1, r_2, \ldots, r_m$
  where P
  group by $A_1, A_{2.....}$
  having P
  order by A_1, A_2

① FROM ② WHERE ③ GROUP BY ④ HAVING ⑤ SELECT ⑥ ORDER BY

Section Review

- Aggregate Functions
  - Basic Aggregation
    - max, min, avg, sum, count
  - Aggregation with Grouping
  - The Having Clause
  - Aggregation with Null Values