Database Design & Applications

The Database Language - SQL Aggregation

Aggregating Data Using Group Functions

Objectives

- After completing this lesson, you should be able to do the following:
  - Identify the available group functions
  - Describe the use of group functions
  - Group data using the GROUP BY clause
  - Include or exclude grouped rows by using the
  - HAVING clause

What Are Group Functions?

- Group functions operate on sets of rows to give one result per group.

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
	7000
10	4400

...

20 rows selected.

The maximum salary in the `EMPLOYEES` table.

MAX(SALARY)
24000

Types of Group Functions

- AVG
- COUNT
- MAX
- MIN
- STDDEV
- SUM
- VARIANCE

Group Functions Syntax

```
SELECT      [column,] group_function(column), ...
FROM        table
[WHERE      condition]
[GROUP BY   column]
[ORDER BY   column];
```

AVG and SUM Functions

- You can use AVG and SUM for numeric data.

```
SELECT AVG(salary), MAX(salary),
       MIN(salary), SUM(salary)
FROM   employees
WHERE  job_id LIKE '%REP%';
```

AVG(SALARY)	MAX(SALARY)	MIN(SALARY)	SUM(SALARY)
8150	11000	6000	32600

MIN and MAX Functions

- You can use MIN and MAX for any data type.

```
SELECT MIN(hire_date), MAX(hire_date)
FROM   employees;
```

MIN(HIRE_	MAX(HIRE_
17-JUN-87	29-JAN-00

COUNT Function

COUNT(*) returns the number of rows in a table.

```
SELECT COUNT(*)
FROM   employees
WHERE  department_id = 50;
```

COUNT(*)
5

COUNT Function

- COUNT(*expr*) returns the number of rows with non-null values for the *expr*.
- Display the number of department values in the EMPLOYEES table, excluding the null values.

```
SELECT COUNT(commission_pct)
FROM   employees
WHERE  department_id = 80;
```

COUNT(COMMISSION_PCT)
3

DISTINCT Keyword

- COUNT(DISTINCT expr) returns the number of distinct non-null values of the *expr*.
- Display the number of distinct department values in the EMPLOYEES table.

```
SELECT COUNT(DISTINCT department_id)
FROM   employees;
```

COUNT(DISTINCTDEPARTMENT_ID)
7

Group Functions and Null Values

- Group functions ignore null values in the column.

```
SELECT AVG(commission_pct)
FROM employees;
```

AVG(COMMISSION_PCT)
.2125

Handling NULLs with Group Functions

- The ISNULL function forces group functions to include null values.

```
SELECT AVG(ISNULL(commission_pct,
0))   FROM employees;
```

AVG(NVL(COMMISSION_PCT,0))
.0425

Creating Groups of Data

For each department average salary in EMPLOYEE table

EMPLOYEES

DEPARTMENT_ID	SALARY	
10	4400	4400
20	13000	9500
20	6000	
50	5800	
50	3500	
50	3100	3500
50	2500	
50	2600	
60	9000	
60	6000	6400
60	4200	
80	10500	
80	8600	
80	11000	
90	24000	
90	17000	

...

20 rows selected.

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

GROUP BY Clause

```
SELECT        column,
FROM          group_function(column)   table
[WHERE        condition]
[GROUP BY group_by_expression]
 [ORDER BY column];
```

Divide rows in a table into smaller groups by using the GROUP BY clause.

GROUP BY Clause

All columns in the SELECT list that are not in group functions must be in the GROUP BY clause.

```
SELECT department_id, AVG(salary)
FROM   employees
GROUP BY department_id ;
```

DEPARTMENT_ID	AVG(SALARY)
10	4400
20	9500
50	3500
60	6400
80	10033.3333
90	19333.3333
110	10150
	7000

8 rows selected.

GROUP BY Clause

The GROUP BY column does not have to be in the SELECT list.

```
SELECTAVG(salary)
FROM   employees
GROUP BY department_id ;
```

AVG(SALARY)
4400
9500
3500
6400
10033.3333
19333.3333
10150
7000

Grouping by More Than One Column

EMPLOYEES

DEPARTMENT_ID	JOB_ID	SALARY
90	AD_PRES	24000
90	AD_VP	17000
90	AD_VP	17000
60	IT_PROG	9000
60	IT_PROG	6000
60	IT_PROG	4200
50	ST_MAN	5800
50	ST_CLERK	3500
50	ST_CLERK	3100
50	ST_CLERK	2600
50	ST_CLERK	2500
80	SA_MAN	10500
80	SA_REP	11000
80	SA_REP	8600

...

20	MK_REP	6000
110	AC_MGR	12000
110	AC_ACCOUNT	8300

20 rows selected.

Add up the salaries in the EMPLOYEES table for each job, grouped by department.

DEPARTMENT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

Grouping by More Than One Column

```
SELECT    department_id dept_id, job_id, SUM(salary)
FROM      employees
GROUP BY department_id, job_id ;
```

DEPT_ID	JOB_ID	SUM(SALARY)
10	AD_ASST	4400
20	MK_MAN	13000
20	MK_REP	6000
50	ST_CLERK	11700
50	ST_MAN	5800
60	IT_PROG	19200
80	SA_MAN	10500
80	SA_REP	19600
90	AD_PRES	24000
90	AD_VP	34000
110	AC_ACCOUNT	8300
110	AC_MGR	12000
	SA_REP	7000

13 rows selected.

Illegal Queries Using Group Functions

Any column or expression in the SELECT list that is not an aggregate function must be in the GROUP BY clause.

```
SELECT department_id,
COUNT(last_name)  FROM employees;
```

```
ERROR:
Column 'EMPLOYEES.department_id' is invalid in the
select list because it is not contained in either an
aggregate function or the GROUP BY clause.
```

Illegal Queries Using Group Functions

- You cannot use the WHERE clause to restrict groups.
- You use the HAVING clause to restrict groups.
- You cannot use group functions in the WHERE clause.

```
SELECT    department_id,
FROM      AVG(salary)   employees
WHERE     AVG(salary) > 8000
GROUP BY department_id;
```

```
WHERE  AVG(salary) > 8000
ERROR at line 3:
```

Cannot use the `WHERE` clause to restrict groups

Excluding Group Results

EMPLOYEES

DEPARTMENT_ID	SALARY
90	24000
90	17000
90	17000
60	9000
60	6000
60	4200
50	5800
50	3500
50	3100
50	2600
50	2500
80	10500
80	11000
80	8600
...	
20	6000
110	12000
110	8300

20 rows selected.

The maximum salary per department when it is greater than $10,000

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

Using HAVING Clause

Use the HAVING clause to restrict groups:

- Rows are grouped.
- The group function is applied.
- Groups matching the HAVING clause are displayed.

```
SELECT      column,
FROM        group_function  table
[WHERE      condition]
[GROUP BY   group_by_expression
[HAVING     group_condition]
[ORDER BY   column];
```

Using HAVING Clause

```
SELECT   department_id,
MAX(salary)  FROM  employees
GROUP BY department_id
HAVING MAX(salary)>10000   ;
```

DEPARTMENT_ID	MAX(SALARY)
20	13000
80	11000
90	24000
110	12000

Using HAVING Clause

```
SELECT   job_id, SUM(salary)
FROM     PAYROLL  employees
WHERE    job_id NOT LIKE '%REP%'
GROUP BY job_id
HAVINGSUM(salary) > 13000
ORDER BY SUM(salary);
```

JOB_ID	PAYROLL
IT_PROG	19200
AD_PRES	24000
AD_VP	34000

THANK YOU!