Reporting Aggregate Data

Using the Group Functions

Objectives

After completing this lesson AND practicing,

• Describe the use of group functions

• Group data by using the GROUP BY clause

• Include or exclude grouped rows by using the

HAVING Clause

you will begin to understand the following

This lesson further addresses functions.

It focuses on obtaining summary information (such as averages) for groups of rows.

It discusses how to group rows in a table into smaller sets and

* how to specify search criteria for groups of rows.

What Are Group Functions

Group Functions 🡺 operate on sets of rows

🡺 to give one result per group

EXAMPLE:

SELECT AVG (salary)

FROM employees;

AVG(SALARY)

One result from a set of rows

8775

Single Row functions worked on single rows and returned 1 result per row

SELECT UPPER(last\_name) …

Each row selected changed the format of whatever last\_name was stored as to display in UPPER case

Group functions (multi-row functions)

🡪 Operate on sets of rows to give one result per group.

- These sets may comprise the entire table or the table split into groups

Types of Group Functions

AVG

COUNT

MAX

Group

Functions

MIN

1 result

SUM

OPTIONS you can use with the functions

AVG ( [distinct | ALL] EXPRE}) n )

- Average value of n, ignoring null values

COUNT ( { \* [distinct | ALL] })

- Number of rows where expr evaluates to something other than null

- count all selected rows using \* including duplicates and nulls unless use distinct

MAX([DISTINCT|ALL]expr)

- Maximum value of expr, ignoring null

MIN([DISTINCT|ALL]expr)

-Minimum value of expr, ignoring null values

SUM([DISTINCT|ALL]n)

- Sum values of n, ignoring null values

GROUP FUNCTION SYNTAX

PROBLEM: President wants to know data about salaries, such as AVERAGE, what the highest and lowest paid person's salary is and the company's total salary payout.

SELECT AVG (SALARY),

MAX (SALARY),

MIN (SALARY),

SUM (SALARY)

FROM EMPLOYEES;

AVG(SALARY) MAX(SALARY) MIN(SALARY) SUM(SALARY)

----------- ----------- ----------- -----------

8775 24000 2500 175500

Show the same for all Sales Reps

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

GUIDELINES

1

DISTINCT

- Makes the function consider only non-duplicate values

ALL

- Makes function consider every value

DEFAULT value is ALL and does not need to be specified

2

The DATA TYPES with the syntax ***expr*** argument may be

CHAR, VARCHAR2

NUMBER, DATE

3

All group functions ignore null values.

Using MIN, MAX function examples

You can use MIN and MAX for the following

- Numeric

- Character

- Date

This shows the most senior employees, the one working the longest and the most junior employee

SELECT MIN(HIRE\_DATE),

MAX (HIRE\_DATE)

FROM EMPLOYEES;

MIN(HIRE\_ MAX(HIRE\_

--------- ---------

Poor titles due partly to column width too small to show title

17-JUN-87 29-JAN-00

Applied to character columns

SELECT min (last\_name) as "Most Senior",

Max (last\_name) as "Newest Employee"

FROM employees;

Most Senior Newest Employee

------------------------- -------------------------

Abel Zlotkey

Using the Count Function

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

SELECT COUNT (\*)

FROM EMPLOYEES;

Using COUNT with an expression

SELECT COUNT (commission\_pct)

FROM EMPLOYEES

WHERE DEPARTMENT\_ID = 80;

NOTE:

Count supplies the number of row that satisfies the SELECT statement

Count with WHERE returns the number of rows

Adding an expression returns non-null values

Adding DISTINCT returns the number that are distinct from the number of non-nulls

DISTINCT Examples

How many departments are there in the employees table?

SELECT COUNT (DISTINCT department\_id)

FROM employees;

COUNT (DISTINCTDEPARTMENT\_ID)

----------------------------

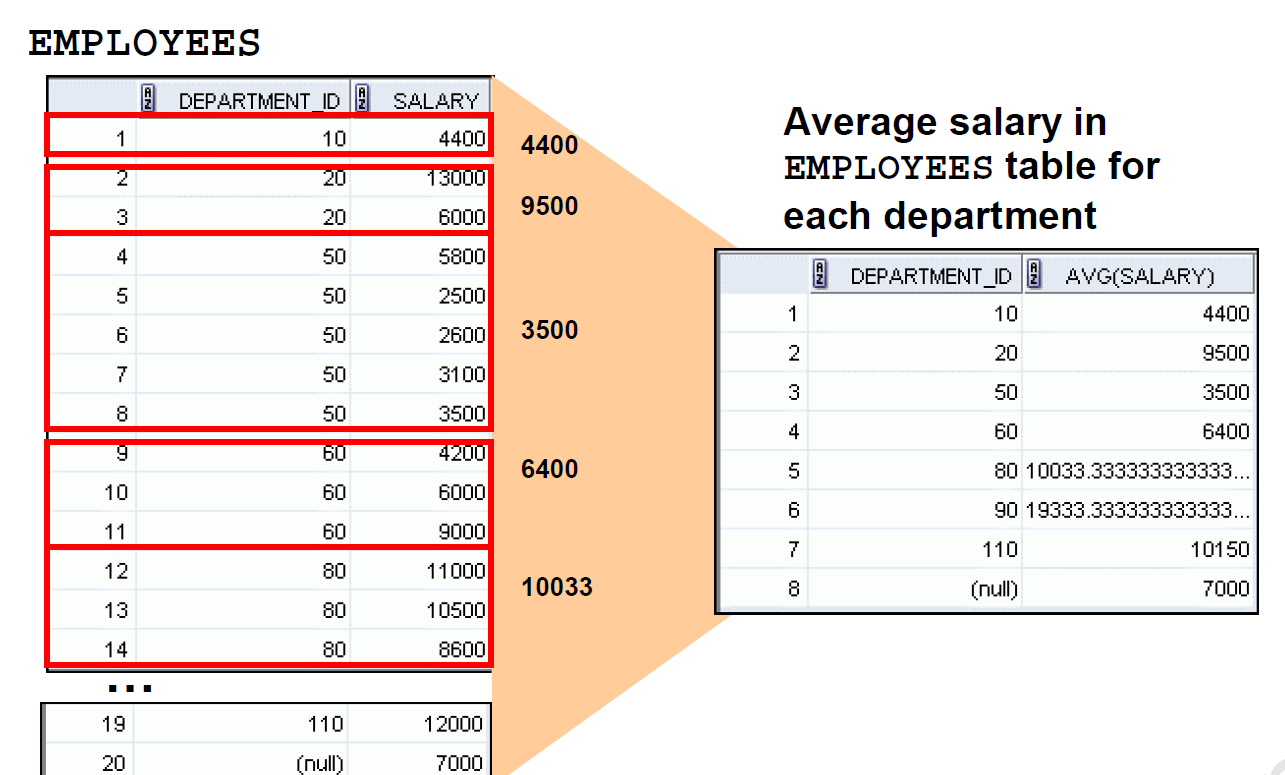
7

Groups of Data

All group functions have treated the table as one large group

Sometimes the information needs to be divide into smaller groups

Example: Average by department



GROUP BY

SELECT DEPARTMENT\_ID, AVG(SALARY)

FROM EMPLOYEES;

ERROR at line 1:

ORA-00937: not a single-group group function

The use of department\_id results in a row of output for each row in the employee table

The AVG wants a single result for the entire table. Can't display it sensibly

Introduces the GROUP BY to apply the group function by department\_id

SELECT DEPARTMENT\_ID, AVG(SALARY)

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID;

DEPARTMENT\_ID AVG(SALARY)

------------- -----------

7000

90 19333.3333

20 9500

110 10150

50 3500

80 10033.3333

60 6400

10 4400

8 rows selected

GROUP BY

The GROUP BY column does not need to be in the select

SELECT AVG(SALARY)

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID;

AVG(SALARY)

-----------

7000

19333.3333

Notice that the output is correct but is not very meaningful to the user

9500

10150

3500

10033.3333

6400

4400

GROUP BY often needs an ORDER BY

SELECT DEPARTMENT\_ID, AVG(SALARY)

FROM EMPLOYEES

GROUP BY DEPARTMENT\_ID

ORDER BY DEPARTMENT\_ID;

DEPARTMENT\_ID AVG(SALARY)

------------- -----------

10 4400

20 9500

50 3500

60 6400

80 10033.3333

90 19333.3333

110 10150

7000

Grouping by more than 1 column

Groups within groups

PROBLEM:

Display the total salary paid to each job title within each department

LOGIC

Group employee by department

Within department group job titles

Sum up that lower grouping

SELECT department\_id, job\_id, SUM(salary)

FROM employees

GROUP BY department\_id, job\_id;

DEPARTMENT\_ID JOB\_ID SUM(SALARY)

------------- ---------- -----------

110 AC\_ACCOUNT 8300

90 AD\_VP 34000

50 ST\_CLERK 11700

80 SA\_REP 19600

50 ST\_MAN 5800

80 SA\_MAN 10500

110 AC\_MGR 12000

90 AD\_PRES 24000

60 IT\_PROG 19200

20 MK\_MAN 13000

SA\_REP 7000

10 AD\_ASST 4400

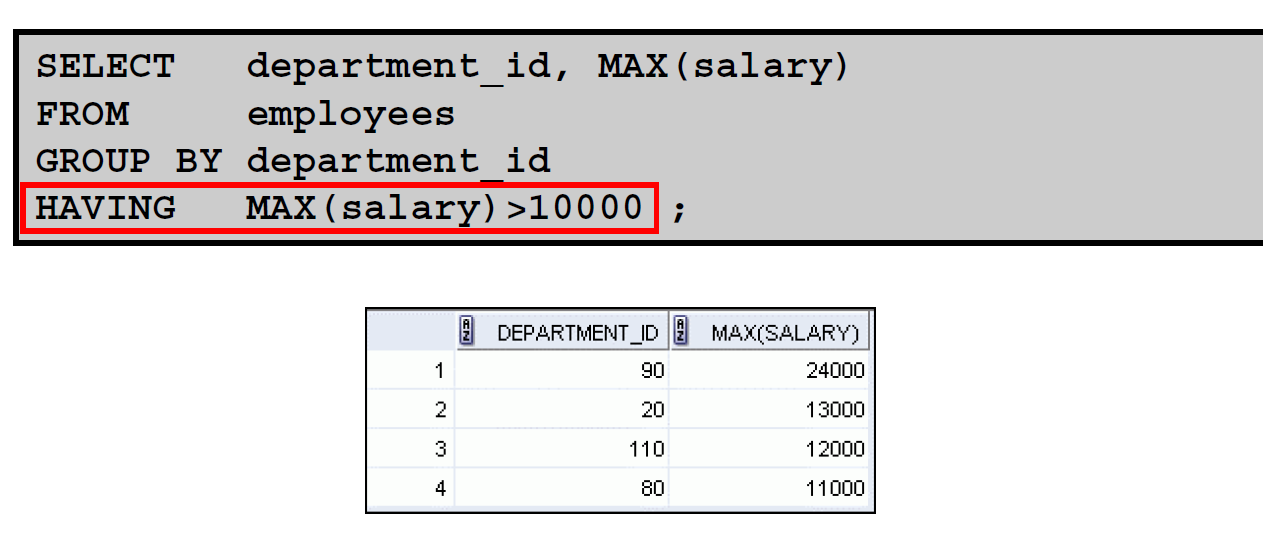
20 MK\_REP 6000

Restricting Which Groups to Show

🡪NOT by using the WHERE clause

🡪 HAVING clause

Find the maximum salary by department if maximum salary greater than 10,000



Again nicer if put in order

Add the ORDER BY clause

Nesting Group Functions

Display the maximum average salary

SELECT MAX(AVG(salary))

FROM employees

GROUP BY department\_id;

MAX(AVG(SALARY))

----------------

19333.3333