Title: Aggregating and Data Using Group Function

Author: S. Santhosh Kumar

Exercises and Solutions

1. Group Functions Work Across Many Rows to Produce One Result per Group.

True/False:
Answer: True

2. Group Functions Include Nulls in Calculations.

True/False:
Answer: False

3. The `WHERE` Clause Restricts Rows Prior to Inclusion in a Group Calculation.

True/False:
Answer: True

4. Find the Highest, Lowest, Sum, and Average Salary of All Employees

```
SELECT ROUND(MAX(salary)) AS "Maximum",
       ROUND(MIN(salary)) AS "Minimum",
       ROUND(SUM(salary)) AS "Sum",
       ROUND(AVG(salary)) AS "Average"
FROM employees;
```

Expected Output:

Maximum	Minimum	Sum	Average
24000	3000	100000	5000

5. Display the Minimum, Maximum, Sum, and Average Salary for Each Job Type

```
SELECT job_id,
       ROUND(MIN(salary)) AS "Minimum",
       ROUND(MAX(salary)) AS "Maximum",
       ROUND(SUM(salary)) AS "Sum",
       ROUND(AVG(salary)) AS "Average"
FROM employees
GROUP BY job_id;
```

Expected Output:

job_id	Minimum	Maximum	Sum	Average
IT_PROG	4000	12000	50000	8000

6. Display the Number of People with the Same Job

```
SELECT COUNT(*) AS "Number of People"
FROM employees
WHERE job_id = '&job_title';
```

Expected Output:

Number of People
5

7. Determine the Number of Managers

```
SELECT COUNT(DISTINCT manager_id) AS "Number of Managers"
FROM employees
WHERE manager_id IS NOT NULL;
```

Expected Output:

Number of Managers
8

8. Find the Difference Between the Highest and Lowest Salaries

```
SELECT (MAX(salary) - MIN(salary)) AS "DIFFERENCE"
FROM employees;
```

Expected Output:

DIFFERENCE
21000

9. Display the Manager Number and the Salary of the Lowest-Paid Employee

```
SELECT manager_id, MIN(salary) AS "Minimum Salary"
FROM employees
WHERE manager_id IS NOT NULL
GROUP BY manager_id
HAVING MIN(salary) > 6000
ORDER BY MIN(salary) DESC;
```

Expected Output:

manager_id	Minimum Salary
100	8000
101	7500

10. Display the Total Number of Employees and the Number Hired Each Year

```
SELECT COUNT(*) AS "Total Employees",
       SUM(CASE WHEN TO_CHAR(hire_date, 'YYYY') = '1995' THEN 1 ELSE 0 END)
AS "Hired in 1995",
       SUM(CASE WHEN TO_CHAR(hire_date, 'YYYY') = '1996' THEN 1 ELSE 0 END)
AS "Hired in 1996",
       SUM(CASE WHEN TO_CHAR(hire_date, 'YYYY') = '1997' THEN 1 ELSE 0 END)
AS "Hired in 1997",
       SUM(CASE WHEN TO_CHAR(hire_date, 'YYYY') = '1998' THEN 1 ELSE 0 END)
AS "Hired in 1998"
FROM employees;
```

Expected Output:

Total Employees	Hired in 1995	Hired in 1996	Hired in 1997	Hired in 1998
107	5	10	8	9

11. Matrix Query to Display Job, Salary by Department, and Total Salary

```
SELECT job_id,
       SUM(CASE WHEN department_id = 20 THEN salary ELSE 0 END) AS "Dept 20
Salary",
       SUM(CASE WHEN department_id = 50 THEN salary ELSE 0 END) AS "Dept 50
Salary",
       SUM(CASE WHEN department_id = 80 THEN salary ELSE 0 END) AS "Dept 80
Salary",
       SUM(CASE WHEN department_id = 90 THEN salary ELSE 0 END) AS "Dept 90
Salary",
       SUM(salary) AS "Total Salary"
FROM employees
WHERE department_id IN (20, 50, 80, 90)
GROUP BY job_id;
```

Expected Output:

job_id	Dept 20 Salary	Dept 50 Salary	Dept 80 Salary	Dept 90 Salary	Total Salary
IT_PROG	10000	20000	30000	40000	100000

12. Display Each Department's Name, Location, Number of Employees, and Average Salary

```
SELECT d.department_name || ', ' || l.city AS "Location",
       COUNT(e.employee_id) AS "Number of People",
       ROUND(AVG(e.salary), 2) AS "Average Salary"
FROM employees e
JOIN departments d ON e.department_id = d.department_id
JOIN locations l ON d.location_id = l.location_id
GROUP BY d.department_name, l.city;
```

Expected Output:

Location	Number of People	Average Salary
IT, New York	25	8000.50
HR, Los Angeles	15	7000.75