Ex.No: 10 AGGREGATING DATA USING GROUP FUNCTION

DATE:25.09.2024

1. **Group functions work across many rows to produce one result per group.**
   True
   Explanation: Group functions, like `SUM`, `AVG`, `COUNT`, `MAX`, and `MIN`, aggregate data across multiple rows to produce a single result per group defined by the `GROUP BY` clause.
2. **Group functions include nulls in calculations.**
   False
   Explanation: Most group functions ignore `NULL` values in their calculations (e.g., `SUM` and `AVG` skip `NULL`s). However, `COUNT(*)` will count all rows, including those with `NULL` values.
3. **The WHERE clause restricts rows prior to inclusion in a group calculation.**
   True
   Explanation: The `WHERE` clause filters rows before grouping occurs. Only the rows that meet the `WHERE` condition are included in the group calculation.

The HR department needs the following reports:

4. Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number

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

MAXIMUM	MINIMUM	SUM	AVERAGE
7000	4500	28000	5600

1 rows returned in 0.00 seconds Download

5. Modify the above query to display the minimum, maximum, sum, and average salary for each job type.

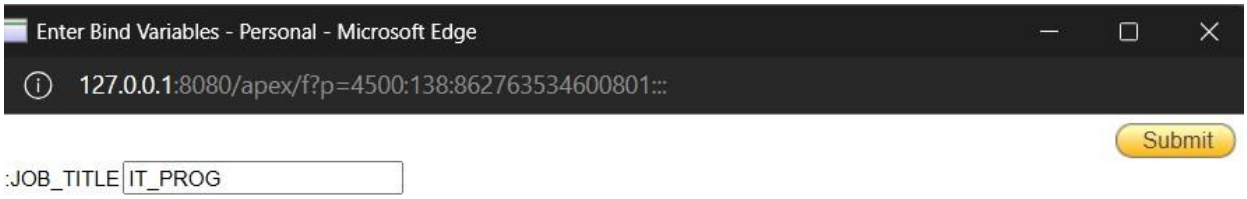
```
SELECT
```

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

JOB_ID	MAXIMUM	MINIMUM	SUM	AVERAGE
IT_PROG	5000	4500	9500	4750
AD_ASST	7000	7000	7000	7000
SA_MAN	5500	5500	5500	5500
HR_REP	6000	6000	6000	6000

6.Write a query to display the number of people with the same job. Generalize the query so that the user in the HR department is prompted for a job title.

```
SELECT
    COUNT(*) AS Number_of_People
FROM employees
WHERE job_id = :job_title;
```



NUMBER_OF_PEOPLE
2

7. Determine the number of managers without listing them. Label the column Number of Managers. Hint: Use the MANAGER_ID column to determine the number of Managers.

```
SELECT
    COUNT(DISTINCT manager_id) AS "Number of Managers"
FROM employees
```

```
WHERE manager_id IS NOT NULL;
```

Number of Managers
2

8. Find the difference between the highest and lowest salaries. Label the column DIFFERENCE.

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

DIFFERENCE
2500

9. Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

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

no data found

10. Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings.

```
SELECT
   COUNT(*) AS Total_Employees,
   SUM(CASE WHEN EXTRACT(YEAR FROM hire_date) = 1995 THEN 1 ELSE 0 END) AS
Hired_in_1995,
```

```
    SUM(CASE WHEN EXTRACT(YEAR FROM hire_date) = 1996 THEN 1 ELSE 0 END) AS
Hired_in_1996,
    SUM(CASE WHEN EXTRACT(YEAR FROM hire_date) = 1997 THEN 1 ELSE 0 END) AS
Hired_in_1997,
    SUM(CASE WHEN EXTRACT(YEAR FROM hire_date) = 1998 THEN 1 ELSE 0 END) AS
Hired_in_1998
FROM employees;
```

TOTAL_EMPLOYEES	HIRED_IN_1995	HIRED_IN_1996	HIRED_IN_1997	HIRED_IN_1998
5	2	1	1	1

11. Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

```
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;
```

TOTAL_EMPLOYEES	HIRED_IN_1995	HIRED_IN_1996	HIRED_IN_1997	HIRED_IN_1998
5	2	1	1	1

12.Write a query to display each department‘s name, location, number of employees, and the average salary for all the employees in that department. Label the column name-Location, Number of people, and salary respectively. Round the average salary to two decimal places.

```
SELECT
    d.department_name || '-' || l.location_id AS "name-Location",
    COUNT(e.employee_id) AS "Number of people",
    ROUND(AVG(e.salary), 2) AS "salary"
```

```
FROM departments d
JOIN employees e ON d.department_id = e.department_id
JOIN locations l ON d.location_id = l.location_id
GROUP BY d.department_name, l.location_id;
```

name-Location	Number of people	salary
Sales-1700	1	5500
IT-1500	2	4750
HR-1600	1	6000
Administration-1800	1	7000