EXERCISE-8

Aggregating Data Using Group Functions

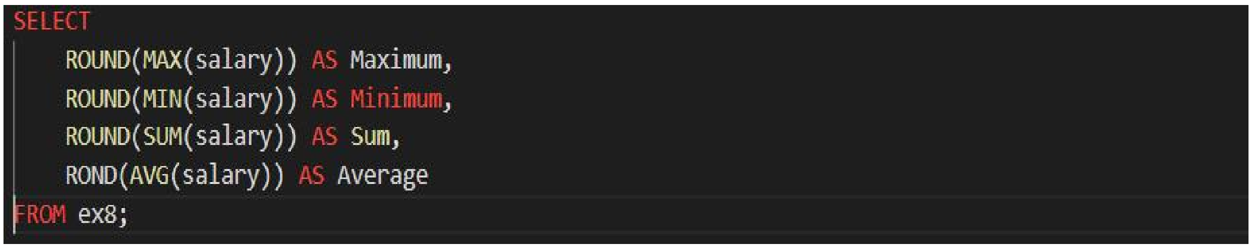
Find the Solution for the following: Determine the validity of the following three statements. Circle either True or False.

1. Group functions work across many rows to produce one result per group. [✓ True] / False
2. Group functions include nulls in calculations. True / [✓ False]
3. The WHERE clause restricts rows prior to inclusion in a group calculation. [✓ True] / False

The HR department needs the following reports:

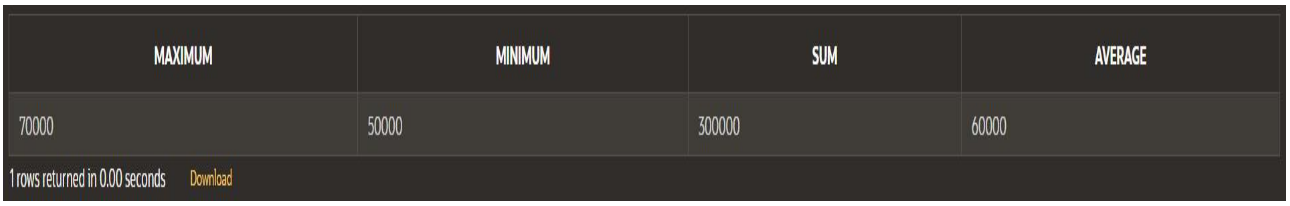
4. Find the highest, lowest, sum, and average salary of all employees. Label the columns Maximum, Minimum, Sum, and Average, respectively. Round your results to the nearest whole number.

QUERY :



```
SELECT
    ROUND(MAX(salary)) AS Maximum,
    ROUND(MIN(salary)) AS Minimum,
    ROUND(SUM(salary)) AS Sum,
    ROND(AVG(salary)) AS Average
FROM ex8;
```

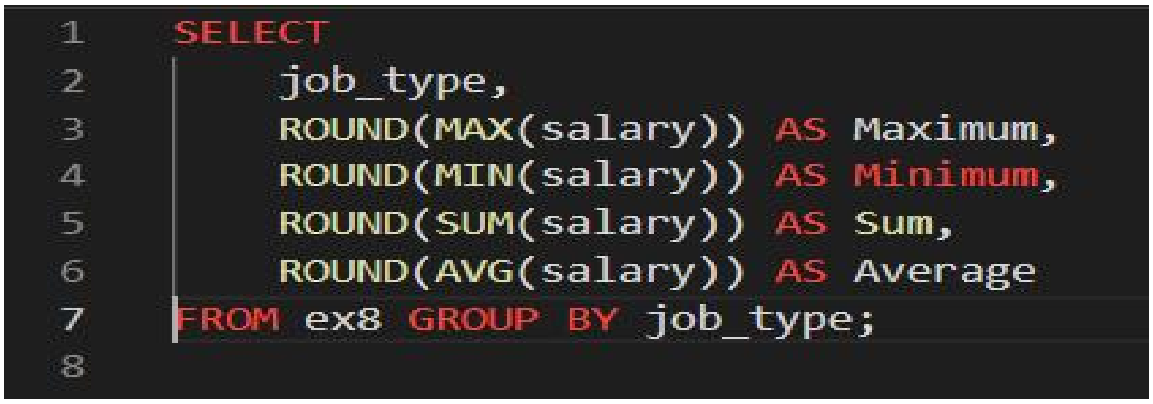
OUTPUT :



MAXIMUM	MINIMUM	SUM	AVERAGE
70000	50000	300000	60000

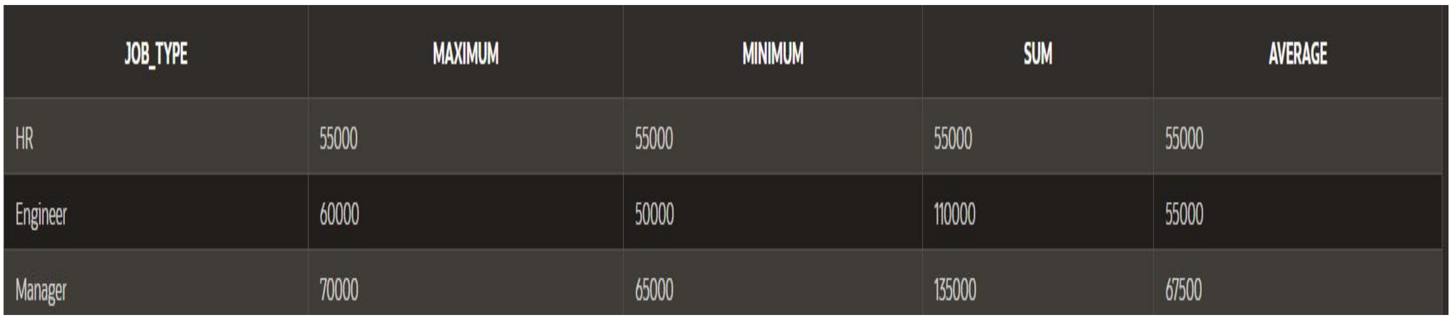
1 rows returned in 0.00 seconds   Download

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



```
SELECT
    job_type,
    ROUND(MAX(salary)) AS Maximum,
    ROUND(MIN(salary)) AS Minimum,
    ROUND(SUM(salary)) AS Sum,
    ROUND(AVG(salary)) AS Average
FROM ex8 GROUP BY job_type;
```

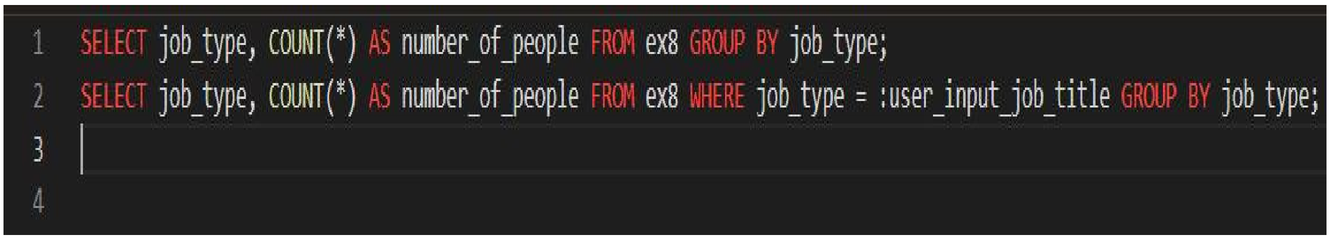
OUTPUT :



JOB_TYPE	MAXIMUM	MINIMUM	SUM	AVERAGE
HR	55000	55000	55000	55000
Engineer	60000	50000	110000	55000
Manager	70000	65000	135000	67500

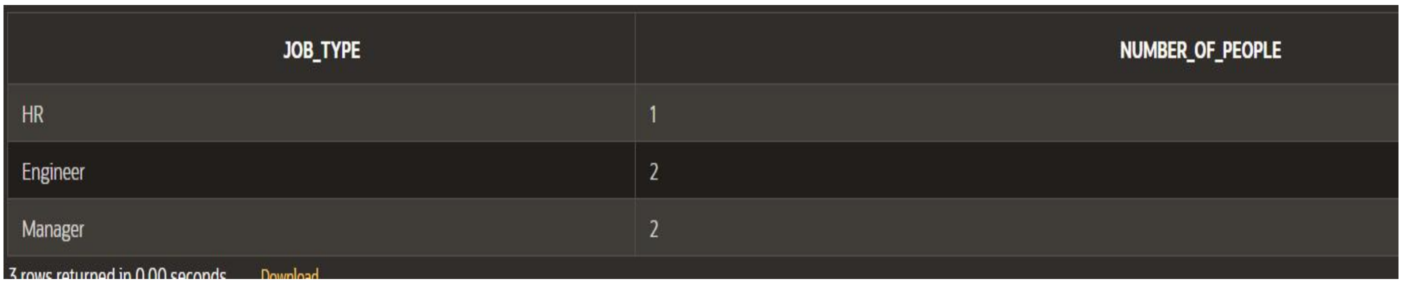
6. Write a query to display the number of people with the same job. Generalize the query so that the user in the HR department is prompted for a job title.

QUERY :



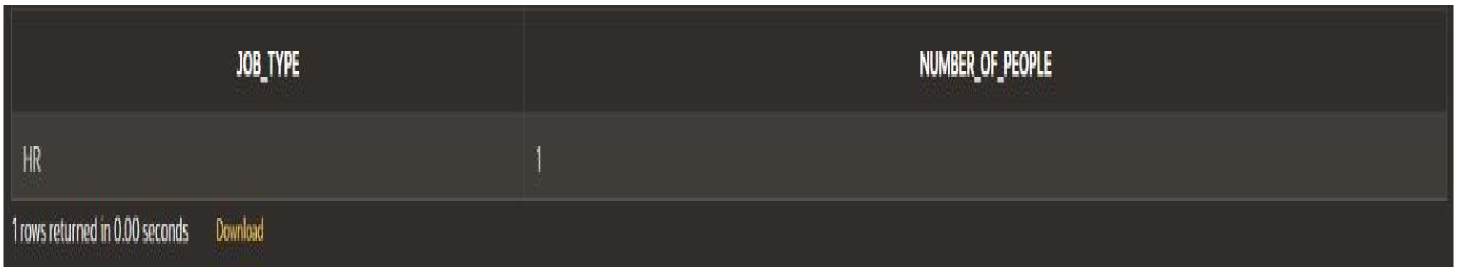
```
SELECT job_type, COUNT(*) AS number_of_people FROM ex8 GROUP BY job_type;
SELECT job_type, COUNT(*) AS number_of_people FROM ex8 WHERE job_type = :user_input_job_title GROUP BY job_type;
```

OUTPUT :



JOB_TYPE	NUMBER_OF_PEOPLE
HR	1
Engineer	2
Manager	2

3 rows returned in 0.00 seconds   Download

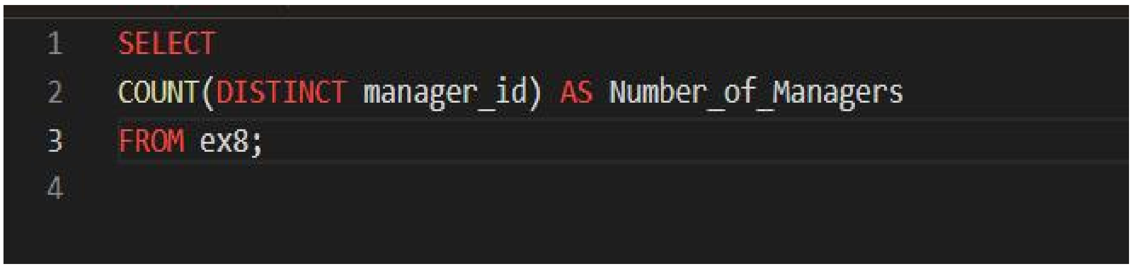


JOB_TYPE	NUMBER_OF_PEOPLE
HR	1

1 rows returned in 0.00 seconds   Download

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.

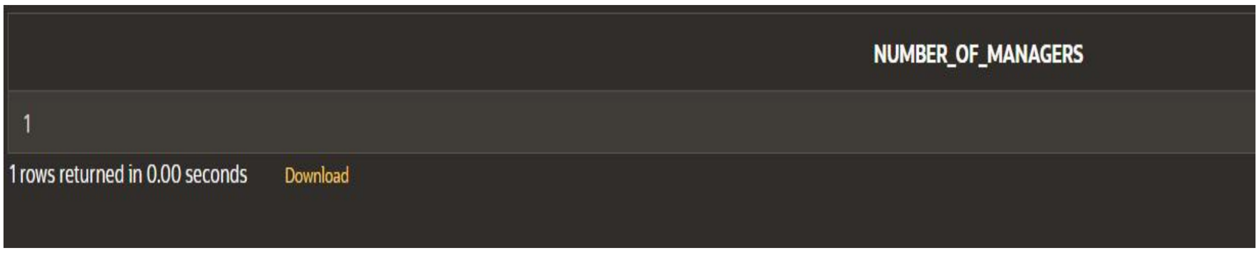
QUERY :



```
SELECT
COUNT(DISTINCT manager_id) AS Number_of_Managers
FROM ex8;

```

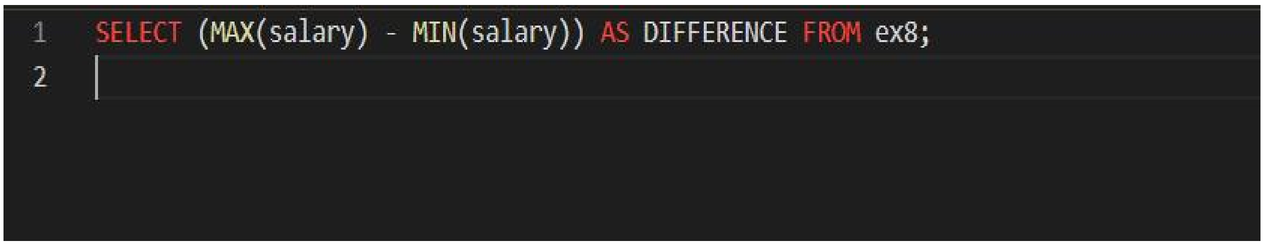
OUTPUT :



```
                                        NUMBER_OF_MANAGERS
1
1 rows returned in 0.00 seconds    Download
```

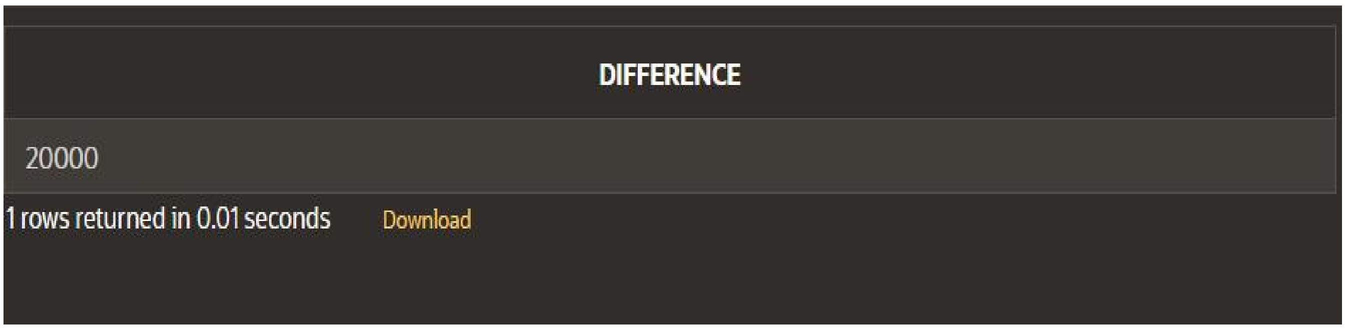
8. Find the difference between the highest and lowest salaries. Label the column DIFFERENCE

QUERY :



```sql
SELECT (MAX(salary) - MIN(salary)) AS DIFFERENCE FROM ex8;
```

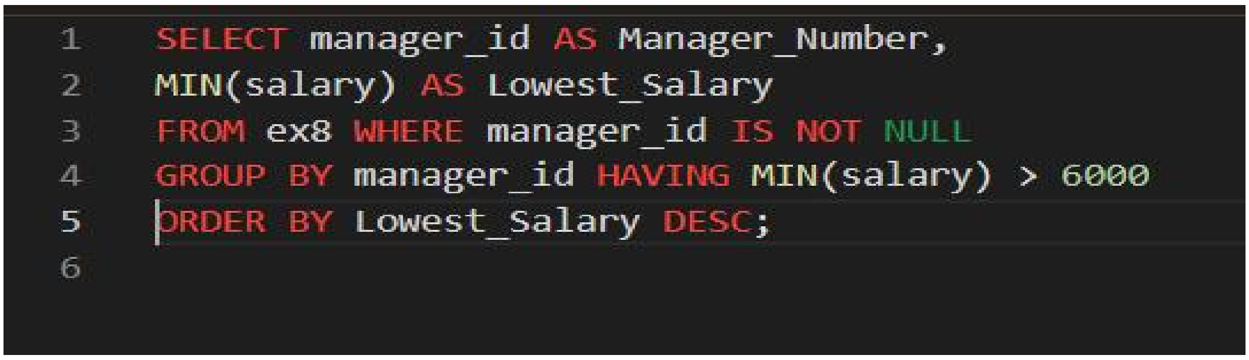
OUTPUT :



```
DIFFERENCE
20000
1 rows returned in 0.01 seconds     Download
```

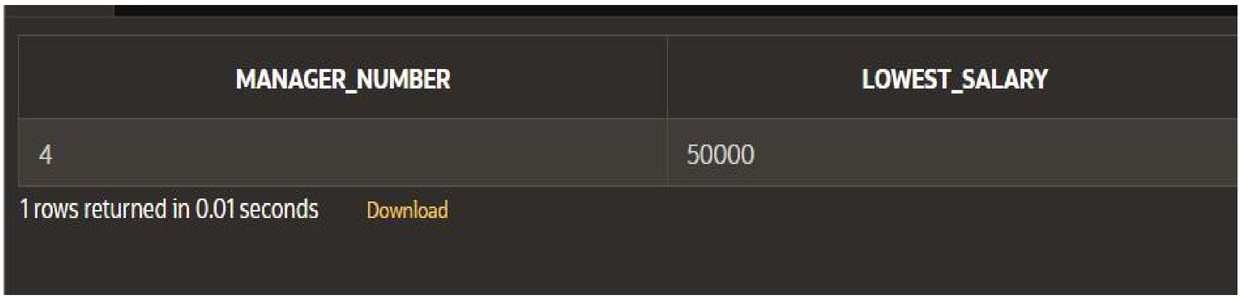
9. Create a report to display the manager number and the salary of the lowest-paid employee for that manager. Exclude anyone whose manager is not known. Exclude any groups where the minimum salary is $6,000 or less. Sort the output in descending order of salary.

QUERY :



```sql
SELECT manager_id AS Manager_Number,
MIN(salary) AS Lowest_Salary
FROM ex8 WHERE manager_id IS NOT NULL
GROUP BY manager_id HAVING MIN(salary) > 6000
ORDER BY Lowest_Salary DESC;
```

OUTPUT :

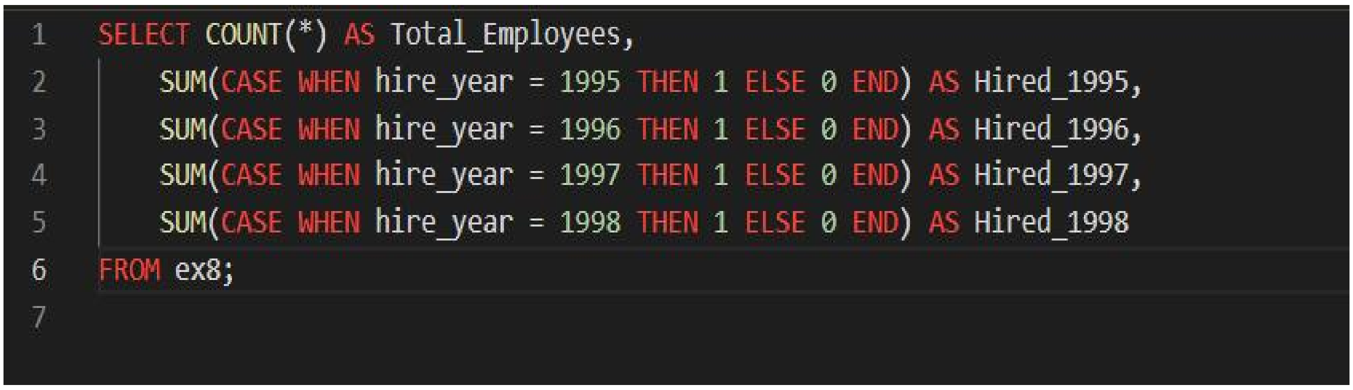


MANAGER_NUMBER	LOWEST_SALARY
4	50000

1 rows returned in 0.01 seconds Download

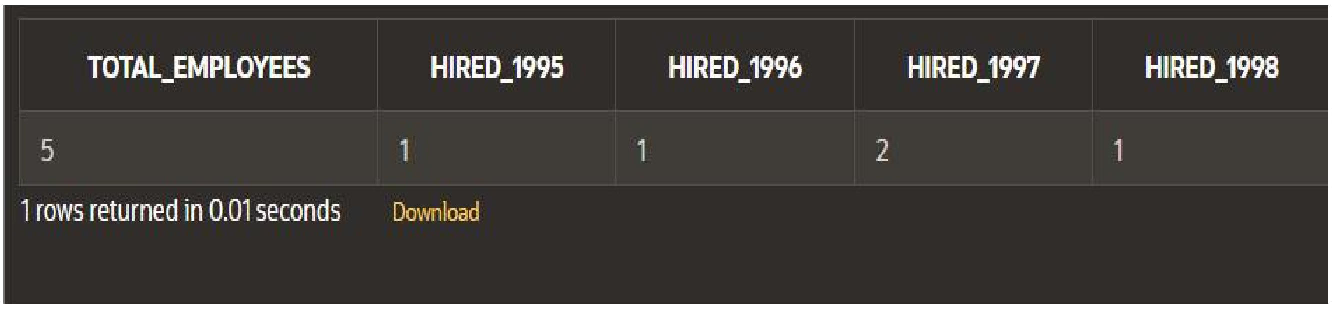
10. Create a query to display the total number of employees and, of that total, the number of employees hired in 1995, 1996, 1997, and 1998. Create appropriate column headings

QUERY :



```
SELECT COUNT(*) AS Total_Employees,
    SUM(CASE WHEN hire_year = 1995 THEN 1 ELSE 0 END) AS Hired_1995,
    SUM(CASE WHEN hire_year = 1996 THEN 1 ELSE 0 END) AS Hired_1996,
    SUM(CASE WHEN hire_year = 1997 THEN 1 ELSE 0 END) AS Hired_1997,
    SUM(CASE WHEN hire_year = 1998 THEN 1 ELSE 0 END) AS Hired_1998
FROM ex8;
```

OUTPUT :

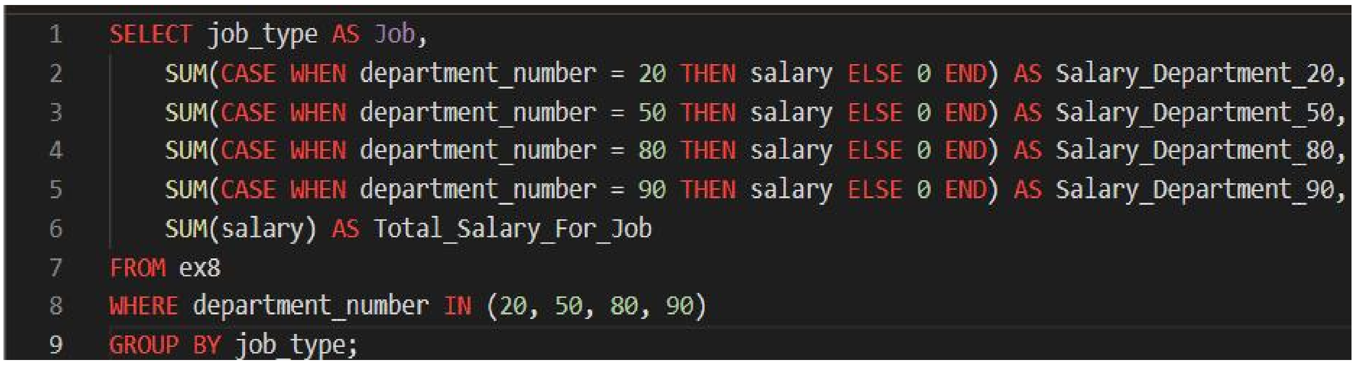


TOTAL_EMPLOYEES	HIRED_1995	HIRED_1996	HIRED_1997	HIRED_1998
5	1	1	2	1

1 rows returned in 0.01 seconds Download

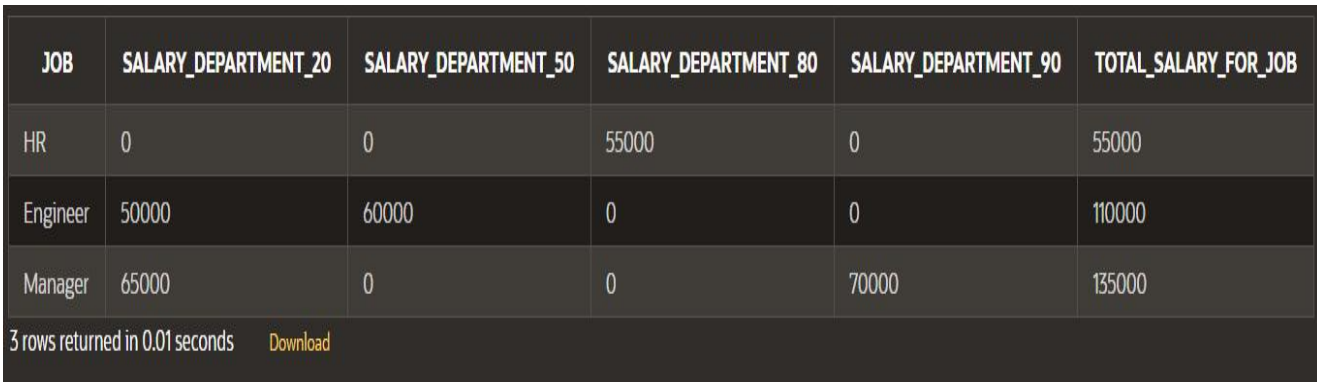
11. Create a matrix query to display the job, the salary for that job based on department number, and the total salary for that job, for departments 20, 50, 80, and 90, giving each column an appropriate heading.

QUERY :



```sql
SELECT job_type AS Job,
    SUM(CASE WHEN department_number = 20 THEN salary ELSE 0 END) AS Salary_Department_20,
    SUM(CASE WHEN department_number = 50 THEN salary ELSE 0 END) AS Salary_Department_50,
    SUM(CASE WHEN department_number = 80 THEN salary ELSE 0 END) AS Salary_Department_80,
    SUM(CASE WHEN department_number = 90 THEN salary ELSE 0 END) AS Salary_Department_90,
    SUM(salary) AS Total_Salary_For_Job
FROM ex8
WHERE department_number IN (20, 50, 80, 90)
GROUP BY job_type;
```

OUTPUT :

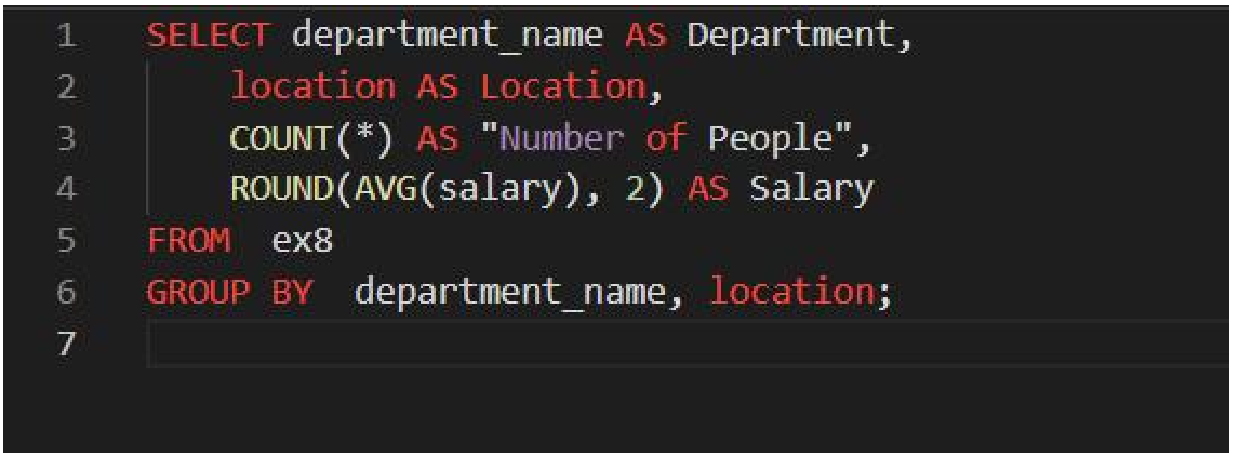


JOB	SALARY_DEPARTMENT_20	SALARY_DEPARTMENT_50	SALARY_DEPARTMENT_80	SALARY_DEPARTMENT_90	TOTAL_SALARY_FOR_JOB
HR	0	0	55000	0	55000
Engineer	50000	60000	0	0	110000
Manager	65000	0	0	70000	135000

3 rows returned in 0.01 seconds Download

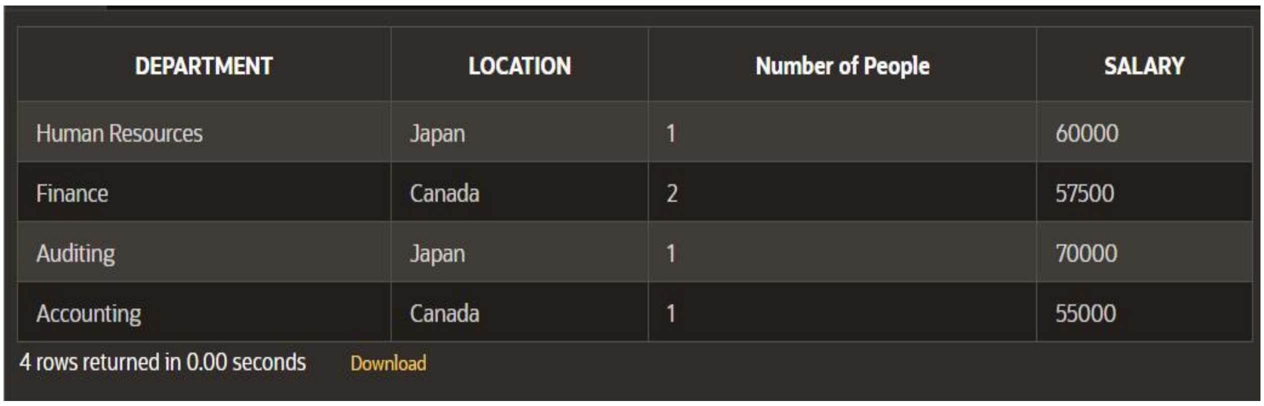
12. Write a query to display each department‘s name, location, number of employees, and the average salary for all the employees in that department. Label the column name-Location, Number of people, and salary respectively. Round the average salary to two decimal places

QUERY :



```sql
SELECT department_name AS Department,
    location AS Location,
    COUNT(*) AS "Number of People",
    ROUND(AVG(salary), 2) AS Salary
FROM  ex8
GROUP BY  department_name, location;
```

OUTPUT :



DEPARTMENT	LOCATION	Number of People	SALARY
Human Resources	Japan	1	60000
Finance	Canada	2	57500
Auditing	Japan	1	70000
Accounting	Canada	1	55000

4 rows returned in 0.00 seconds   Download