

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

Example displays the job ID and total monthly salary for each job that has a total payroll exceeding \$13,000. The example excludes sales representatives and sorts the list by the total monthly salary.

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

Nesting Group Functions

Display the maximum average salary:

Group functions can be nested to a depth of two. The slide example displays the maximum average salary.

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

Summary

In this exercise, students should have learned how to:

- Use the group functions COUNT, MAX, MIN, and AVG
- Write queries that use the GROUP BY clause
- Write queries that use the HAVING clause

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

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 True

2. Group functions include nulls in calculations.
True/False False

3. The WHERE clause restricts rows prior to inclusion in a group calculation.
True/False True

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 MAX(salary) AS maximum MIN(salary)  
AS minimum, SUM(salary) AS sum, ROUND  
(AVG(salary)) AS Average FROM employees;
```

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

```
SELECT Job-id, min(salary), max(salary), sum(salary)
ROUND (avg(salary)) FROM employees GROUP BY Job-id
```

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 Job-id, COUNT(*) FROM employees
GROUP BY Job-id;
```

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
MANAGER" FROM employees;
```

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

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

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) FROM employees
WHERE manager-id IS NOT NULL GROUP BY manager-id
HAVING min(salary) > 6000 ORDER BY min(salary)
DESC
```

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" SUM(CASE WHEN TO
chr) (hire-date, 'YYYY') = '1995'
THEN, ELSE 0 END) AS "1995", SUM(CASE
WHEN TO - chr (hire-date, 'YYYY') = '1996'
```


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
THEN salary ELSE 0 END) AS "Dept 20",
SUM (CASE WHEN department_id = 50
THEN salary ELSE 0) AS "Dept 80"
```

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 AS "NAME", location,
1. City as "Number of People", COUNT (e.employee_id)
ROUND (AVG (e.salary), 2) AS "Salary" FROM
employees e, departments d, locations l WHERE
e.department_id = d.department_id AND d.location_id
= l.location_id GROUP BY department_name, l.city
```

Evaluation Procedure	Marks awarded
Query(5)	5
Execution (5)	5
Viva(5)	5
Total (15)	15
Faculty Signature	