Analytical Functions Exercise

1) Write a query to find the average salary by department **without** using the GROUP BY function. You have to use analytical function and do this.

2) Write a query to display the following information (Order by JOB_ID)

   - Employee ID
   - Job ID
   - Salary
   - Department ID
   - Average salary by Job ID
   - Average salary by Department ID
   - Average salary of all employees

3) Write a query to display the following information (Order by JOB_ID)

   - Employee ID
   - Job ID
   - Department ID
   - Salary
   - Average salary by Department ID
   - Difference between employee salary and Average salary by Department ID

4) Write a query to display the following information (Order by JOB_ID)

   - Job ID
   - Department ID
   - Number of managers by Job ID
   - Number of managers by Department ID
   - Total number of managers

5) Write a query to display the following information (Experiment with RANK, DENSE_RANK and ROW_NUMBER analytical functions)

   - Employee ID
   - Department ID
   - Salary
   - Salary Ranking

6) Write a query to display the following information

- Employee ID
- Department ID
- Salary
- Salary Ranking by Department ID

7) Write a query to display the following information

- Employee ID
- Department ID
- Salary
- Top Salary Ranking by Department ID
- Bottom Salary Ranking by Department ID

8) Write a query to display the following information, but display only top 3 and bottom 3 salaries.

- Employee ID
- Department ID
- Salary
- Top Salary Ranking by Department ID
- Bottom Salary Ranking by Department ID

9) Write a query to display the following information for the Department ID 30

- Employee ID
- Salary
- Next lowest salary (You need to use LEAD)
- Previous highest salary (You need to use LAG)

Answer 1

```
 select  DISTINCT
       d.department_name,
      avg(e.salary) over (partition by department_name) average_salary
from    departments d, employees  e
where d.department_id = e.department_id
```

Answer 2

```
select employee_id,
     job_id,
     salary,
     department_id,
     avg(salary) over (partition by job_id) job_avg,
     avg(salary) over (partition by department_id) deptno_avg,
     avg(salary) over () total_avg
from   employees
order by job_id
```

Answer 3

```
select employee_id,
     job_id,
     department_id,
     salary,
      avg(e.salary) over (partition by e.department_id)  Average_Salary_in_Dept,
      e.salary - avg(e.salary) over (partition by e.department_id)  Diff_With_Dept_Avg
```

```
from   employees e
order by job_id
```

Answer 4

```
select distinct job_id
,     department_id
,     count(distinct manager_id) over (partition by job_id) num_of_job_mgrs
,     count(distinct manager_id) over (partition by department_id) num_of_dept_mgrs
,     count(distinct manager_id) over () number_of_mgrs
from   employees
order by     job_id
```

Answer 5 (Observe the difference between RANK , DENSE_RANK and ROW_NUMBER)

```
select employee_id,
      department_id,
      salary,
      rank() over ( order by salary desc) sal_rank
 from employees

select employee_id,
      department_id,
      salary,
     dense_rank() over ( order by salary desc) sal_rank
from employees
```

```
select employee_id,
       department_id,
       salary,
      row_number() over ( order by salary desc) sal_rank
from employees
```

Answer 6 (Observe the difference between RANK and DENSE_RANK)

```
select employee_id,
       department_id,
       salary,
       rank() over ( partition by department_id order by salary desc) sal_rank
 from employees

select employee_id,
       department_id,
       salary,
      dense_rank() over (partition by department_id order by salary desc) sal_rank
from employees
```

Answer 7

```
select employee_id,
       department_id,
       salary,
       RANK() over (partition by department_id order by salary desc) sal_top_rank,
```

```
    RANK() over (partition by department_id order by salary asc) sal_bottom_rank
     from employees
```

Answer 8

```
SELECT employee_id, department_id, salary, sal_top_rank, sal_bottom_rank from
(
select employee_id,
    department_id,
    salary,
    RANK() over (partition by department_id order by salary desc) sal_top_rank,
    RANK() over (partition by department_id order by salary asc) sal_bottom_rank
     from employees
)
where sal_top_rank <=3 or sal_bottom_rank <=3
```

Answer 9

```
SELECT employee_id, salary,
LEAD(salary, 1, 0) OVER (ORDER BY salary DESC NULLS LAST) NEXT_LOWER_SAL,
LAG(salary, 1, 0) OVER (ORDER BY salary DESC NULLS LAST) PREV_HIGHER_SAL
FROM employees
where department_id = 30
ORDER BY salary DESC;
```