(a) List the data of all departments.

```
SELECT * FROM DEPARTMENTS;
```

(b) List last names, hired date and salary of all employees who work for NO.20 department.

```
SELECT last_name, hire_date, salary
FROM   employees
WHERE  DEPARTMENT_ID=30;
```

Display the average, highest, lowest and sum of the monthly salaries and the number of employees for each department.

```
SELECT   department_id, AVG(salary),
MAX(salary), MIN(salary),SUM(salary), COUNT(*)
FROM     employees
GROUP BY department_id ;
```

Retrieve the name, email and salary of all employees who work for the 'Marketing' department.

```
SELECT last_name, job_id, EMAIL, salary
FROM   employees E JOIN DEPARTMENTS D
ON   E.DEPARTMENT_ID = D.DEPARTMENT_ID
WHERE D.DEPARTMENT_NAME = 'Marketing';
```

Display the employee last name and department name for all employees who have an ‘S’ in their last name.

```
SELECT last_name, department_name
FROM   employees E JOIN DEPARTMENTS D
ON   E.DEPARTMENT_ID = D.DEPARTMENT_ID
WHERE last_NAME like '%S%';
```

Following problem (d) display the department name.

```
SELECT   department_name, AVG(salary),
MAX(salary), MIN(salary),SUM(salary), COUNT(*)
FROM     employees E JOIN DEPARTMENTS D
ON   E.DEPARTMENT_ID = D.DEPARTMENT_ID
GROUP BY department_name ;
```

Find the names of employees and their respective managers.

```
SELECT E.last_name, M.last_name
FROM   employees E JOIN employees M
ON   E.MAANGER_ID = M.EMPLOYEE_ID;
```

Retrieve the name of employees who have minimum salary in their department.

```
SELECT last_name, job_id, salary
FROM   employees
WHERE  (department_id,salary) IN
        (SELECT department_id, MIN(salary)
         FROM   employees
         GROUP BY department_id);
```

```
SELECT last_name, salary, job_id
FROM   employees outer
WHERE  salary = (SELECT MIN(salary)
                 FROM   employees
                 WHERE  department_id =
                  outer.department_id) ;
```

Show the last name, job, salary, and department name of those employees who earn commission. Sort the data by salary in descending order.

```
SELECT last_name, job_id, salary, department_name
FROM   employees E JOIN DEPARTMENTS D
ON   E.DEPARTMENT_ID = D.DEPARTMENT_ID
WHERE E.COMMISSION_PCT IS NOT NULL
ORDER BY salary DESC;
```

Write a query to display the top 3 earners in the EMPLOYEES table. Display their last names, salaries and department name.

```
SELECT rownum RANK, last_name, salary, department_name
FROM
(SELECT department_name, last_name, salary
  from employees E join departments D
  on E.department_id = D.department_id
   order by salary desc)
where rownum < =3;
```

(k) Write a query to display last names of employees who earn less than average salary in their department.

```
SELECT last_name, salary, department_id
FROM   employees outer
WHERE  salary < (SELECT AVG(salary)
                 FROM   employees
                 WHERE  department_id =
                        outer.department_id) ;
```

(I) Find all departments that do not have any employees

```
SELECT department_id, department_name
FROM departments d
WHERE NOT EXISTS (SELECT 'X'
                  FROM   employees
                  WHERE  department_id
                         = d.department_id);
```

(m) Use a correlated subquery to delete only those rows from the EMPLOYEES table that also exist in the JOB_HISTORY table.

```
DELETE FROM employees E
WHERE employee_id =
            (SELECT employee_id
             FROM   emp_history
             WHERE  employee_id = E.employee_id);
```

(n) Write a query to display the last names of employees who have one or more coworkers in their departments with later hire date but higher salaries.

```
SELECT        last_name
FROM          employees outer
WHERE EXISTS     (SELECT 'X'
                  FROM        employees inner
                  WHERE       inner.department_id =
                              outer.department_id
                  AND   inner.hire_date > outer.hire_date
                  AND   inner.salary > outer.salary);
```

Increase 10 % salary of all employees who belong to 'IT' department.

```
UPDATE employees
SET    salary = (1+0.1) * salary
WHERE department_id =
       (SELECT department_id
         FROM   departments d
         WHERE  department_name  = 'IT');
```

Insert < 70, 'Public Relations', 100, 1700> into departments.

```
INSERT INTO departments(department_id,
    department_name, manager_id, location_id)
VALUES (70, 'Public Relations', 100, 1700);
```

Delete the EMPLOYEE tuple with employee_id is 206.

```
DELETE FROM employees
 WHERE  employee_id = 206;
```

Modify the SALARY attribute of the employees 205 tuple with employee_id is 103.

```
UPDATE employees
SET    salary=    (SELECT   salary
                  FROM      EMPLOYEES
                  WHERE     employee_id = 103
WHERE employee_id=205 ;
```

Show the department name, locations, names, job titles, and salaries of employees who work at Taipei City.

```
SELECT department_name, city||street_address , job_title
FROM   employees e
JOIN   departments d
ON     d.department_id = e.department_id
JOIN   locations l
ON     d.location_id = l.location_id
JOIN   JOBS j
ON     e.job_id = j.job_id
WHERE  l.CITY= 'Taipei' ;
```

Show the department number, department name, and number of employees working in each department that: **Includes fewer than 3 employees.**

```
SELECT D.department_id, department_name, COUNT(*)
NOofDept
FROM   employees E join departments D
on E.department_id = D.department_id
HAVING 2 >= COUNT(*)
GROUP BY D.department_id, department_name;
```

Has the highest number of employees.

```
SELECT D.department_id DEPT_ID, department_name,
COUNT(*) NOofDept
FROM   employees E join departments D
on E.department_id = D.department_id
GROUP BY D.department_id, department_name
HAVING  COUNT(*)= (SELECT MAX(COUNT(*)) FROM
employees group by department_id);
```