-- LABORATOR 2

18. Să se afişeze numele angajatului, numele departamentului şi

id-ul locației pentru toţi angajaţii care câştigă comision.

```
SELECT last_name, department_name, location_id
FROM employees e, departments d
WHERE e.department_id = d.department_id AND commission_pct IS NOT NULL;

-- Cum putem afisa si angajatii care nu au departament?
SELECT last_name, department_name, location_id
FROM employees e, departments d
WHERE e.department_id = d.department_id (+) AND commission_pct IS NOT NULL;

SELECT * FROM locations; -- location_id este cheie primara deci avem o lista de locatii UNICE

SELECT * FROM departments; -- location_id este cheie externa
                           --deci avem locatii in care se afla departamente
```

19. Să se afişeze numele, titlul job-ului şi denumirea departamentului pentru toţi angajaţii care lucrează în Oxford.

```
SELECT last_name, job_title, department_name, city, l.location_id
FROM employees e, jobs j, departments d, locations l
WHERE e.job_id = j.job_id and e.department_id = d.department_id
    and d.location_id = l.location_id
    and UPPER(city) = 'OXFORD';
```

20. Să se afişeze codul angajatului şi numele acestuia, împreună cu numele şi codul şefului său direct. Se vor eticheta coloanele Ang#, Angajat, Mgr#, Manager.

```
SELECT ang.employee_id Ang#, ang.last_name Angajat, sef.employee_id Mgr#, sef.last_name
Manager
FROM employees ang, employees sef
WHERE ang.manager_id = sef.employee_id;

SELECT * FROM employees;
```

21. Să se modifice cererea anterioară pentru a afişa toţi salariaţii, inclusiv cei care nu au şef.

```
SELECT ang.employee_id Ang#, ang.last_name Angajat, sef.employee_id Mgr#, sef.last_name
Manager
FROM employees ang, employees sef
WHERE ang.manager_id = sef.employee_id (+);
```

24. Să se afişeze numele şi data angajării pentru salariaţii care au fost angajaţi după Gates.

```
SELECT ang.last_name NumeAng, ang.hire_date DataAng,
        gates.last_name NumeGates, gates.hire_date DataGates
FROM employees ang, employees gates
WHERE ang.hire_date > gates.hire_date
         and initcap(gates.last_name) = 'Gates';

--SAU:
SELECT ang.last_name "Nume Ang", ang.hire_date "Data Ang",
        gates.last_name "Nume Gates", gates.hire_date "Data Gates"
FROM employees ang, employees gates
WHERE ang.hire_date - gates.hire_date > 0
          and initcap(gates.last_name) = 'Gates';
```

25. Să se afişeze numele salariatului şi data angajării împreună cu numele şi

data angajării şefului direct pentru salariaţii care au fost angajaţi înaintea şefilor lor.

Se vor eticheta coloanele Angajat, Data_ang, Manager si Data_mgr.

```
SELECT ang.last_name Angajat, ang.hire_date Data_Ang, m.last_name Manager, m.hire_date
Data_mgr
FROM employees ang, employees m
WHERE ang.manager_id = m.employee_id AND ang.hire_date < m.hire_date;
```

-- LABORATOR 3

-- explicatii JOIN (continuare laborator 2)

-- Join-ul este operaţia de regăsire a datelor din două sau mai multe tabele,

-- pe baza valorilor comune ale unor coloane. De obicei, aceste coloane reprezintă cheia primară,

-- respectiv cheia externă a tabelelor.

-- Reamintim că pentru a realiza un join între n tabele, va fi nevoie de cel puţin n – 1 condiţii de join

-- Tipuri de JOIN:

--Nonequijoin – condiţia de join conţine alţi operatori decât operatorul de egalitate.

--Exemplu Nonequijoin:

```
SELECT * FROM job_grades;
```

```
SELECT last_name, salary, grade_level, lowest_sal, highest_sal
FROM employees, job_grades
WHERE salary BETWEEN lowest_sal AND highest_sal;
```

-- Inner join (equijoin, join simplu)

-- corespunde situaţiei în care valorile de pe coloanele ce apar în condiţia de join trebuie să fie egale

--EXEMPLE (folosind atat join-ul in WHERE cat si cel din standardul SQL3):

-- I. Condiția de Join este scrisă în clauza WHERE a instrucțiunii SELECT

```
SELECT employee_id, d.department_id, last_name, department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id;
```

--JOIN IN FROM (standardul SQL3) - folosind ON

```
SELECT employee_id, d.department_id, last_name, department_name
FROM employees e JOIN departments d ON (e.department_id = d.department_id);
```

-- JOIN IN FROM (standardul SQL3) - folosind USING

-- USING SE UTILIZEAZA dacă există coloane având acelasi nume

-- in acest caz coloanele referite nu trebuie sa contina calificatori

-- adica sa nu fie precedate de nume de tabele sau alias-uri

```
SELECT employee_id, department_id, last_name, department_name
FROM employees JOIN departments USING (department_id);
```

-- OUTER JOIN

-- pentru a afisa si angajatii care nu au departament se utilizeaza

-- simbolul (+) in partea deficitara de informatie

-- deficit de informatie -> angajati FARA departament

```
SELECT employee_id, d.department_id, last_name, department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id (+);
```

-- In cazul standardului SQL3 se utilizeaza LEFT, RIGHT şi FULL OUTER JOIN

-- dorim sa afisam si angajatiti care nu au departament (la fel ca in exemplul anterior)

-- Cum gandim? -> dorim sa afisam TOTI angajatii chiar daca au sau nu departament

-- TOTI angajatii adica tot ce este in tabelul EMPLOYEES

```
SELECT employee_id, d.department_id, last_name, department_name
FROM employees e LEFT JOIN departments d ON (e.department_id = d.department_id);
```

-- => returneaza toate inregistrarile din tabelul EMPLOYEES si numai acele inregistrări din -- DEPARTMENTS care indeplinesc conditia

-- afisam TOATE departamentele chiar daca au sau nu angajati

```
SELECT employee_id, d.department_id, last_name, department_name
FROM employees e RIGHT JOIN departments d ON (e.department_id = d.department_id);
```

1. Scrieți o cerere pentru a se afisa numele, luna (în litere) şi anul angajării

pentru toți salariații din acelasi departament cu Gates,

al căror nume conţine litera “a”. Se va exclude Gates.

```
SELECT ang.last_name NumeAng, TO_CHAR(ang.hire_date, 'month-yyyy') "Luna si an",
      ang.department_id AngDep, gates.last_name NumeGates, gates.department_id DepGates
FROM employees ang JOIN employees gates ON (ang.department_id = gates.department_id)
    and initcap(gates.last_name) = 'Gates'
    and lower(ang.last_name) like '%a%'
    and initcap(ang.last_name) != 'Gates';
```

5. Cum se poate implementa full outer join?

```
-- inner join -> 106 linii returnate => 106 angajati care lucreaza in departamente
SELECT last_name, department_name
FROM employees e JOIN departments d ON (e.department_id = d.department_id);


-- RIGHT JOIN (106 elemente comune + 16 departamente fara angajati)
-- afisam intersectia -> angajatii care lucreaza in departamente
-- impreuna cu departamentele in care nu lucreaza angajati
-- in final o sa obtinem TOATE departamentele chiar daca au sau nu angajati
SELECT last_name, department_name
FROM employees e RIGHT JOIN departments d ON (e.department_id = d.department_id);

-- LEFT JOIN (106 elemente comune + angajatii fara departament -> 1 angajat)
-- afisam intersectia -> angajatii care lucreaza in departamente
-- cat si angajatii care nu au departament
-- in final afisam TOTI angajatii chiar daca au sau nu departament
SELECT last_name, department_name
FROM employees e LEFT JOIN departments d ON (e.department_id = d.department_id);

--left + right = 106 elemente comune + 16 departamente fara ang + 1 anag fara depart = 123

-- full join
-- 123 rezultate
SELECT last_name, department_name
FROM employees e FULL JOIN departments d ON (e.department_id = d.department_id);
```

```
-- UNION
-- observam ca union returneaza 121 de randuri, fata de full join care returneaza 123
-- deoarece UNION -> returneaza elementele comune si necomune o singura data

SELECT last_name, department_name
FROM employees e RIGHT JOIN departments d ON (e.department_id = d.department_id)

UNION

SELECT last_name, department_name
FROM employees e LEFT JOIN departments d ON (e.department_id = d.department_id);


--Cum putem afisa toate elementele (123) la fel ca in cazul lui FULL JOIN?

SELECT employee_id, last_name, department_name
FROM employees e RIGHT JOIN departments d ON (e.department_id = d.department_id)

UNION

SELECT employee_id, last_name, department_name
FROM employees e LEFT JOIN departments d ON (e.department_id = d.department_id);
```