# LABORATOR 3 SQL

# CERERI MULTITABEL. SUBCERERI. STANDARDUL SQL 1999 (SQL3).

- Dacă în clauza FROM a unei comenzi SELECT apar mai multe tabele, atunci se realizează **produsul cartezian** al acestora
- Dacă este necesară obţinerea de informaţii corelate din mai multe tabele, atunci se utilizează **condiţii de join**.
- **Join-ul** este operaţia de regăsire a datelor din două sau mai multe tabele, pe baza valorilor comune ale unor coloane. Condiţiile de corelare utilizează de obicei coloanele cheie primară şi cheie externă.
- Pentru claritatea şi eficienţa accesului la baza de date se recomandă prefixarea numelor coloanelor cu numele tabelelor din care fac parte (*nume_tabel.nume_coloana*).
- De asemenea, există posibilitatea de a utiliza aliasuri pentru tabelele din clauza FROM şi utilizarea acestora în cadrul comenzii SELECT respective (*alias_tabel.nume_coloana*).
- Identificarea prin *nume_tabel.nume_coloana* sau *alias_tabel.nume_coloana* este obligatorie atunci când se face referinţă la o coloana ce apare în mai mult de un tabel din clauza FROM.

Tipuri de *join*:

- **equijoin** (se mai numeşte *inner join* sau *simple join*) – reprezintă compunerea a două tabele diferite după o condiţie ce conţine operatorul de egalitate;

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

- **nonequijoin** – reprezintă compunerea a două tabele după o condiţie ce nu conţine operatorul de egalitate;

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

- **outerjoin** – reprezintă compunerea externă a două tabele; tuplurile tabelului în dreptul căruia apare operatorul extern (+) sunt completate cu valori *null* dacă pentru acestea nu există niciun tuplu din celălalt tabel pentru care să fie îndeplinită condiţia de corelare; operatorul extern (+) poate fi plasat în orice parte a condiţiei de *join*, dar nu în ambele părţi ale acesteia;

  ***Observaţie***:
  - Join-ul a două tabele ce obţine doar liniile ce corespund condiţiei de join se numeşte INNER JOIN.

- Join-ul a două tabele ce obţine atât liniile ce corespund condiţiei de join cât şi cele care nu îndeplinesc această condiţie se numeşte OUTER JOIN. Liniile dintr-un tabel care nu au corespondent în celălalt tabel sunt completate cu valori *null*.

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

- **selfjoin** – reprezintă compunerea unui tabel cu el însuşi după o condiţie dată;

```
SELECT sef.last_name, angajat.last_name
FROM   employees sef, employees angajat
WHERE  sef.employee_id = angajat.manager_id
ORDER BY sef.last_name;
```

**1.** Afişaţi pentru fiecare angajat numele, codul şi numele departamentului în care lucrează.

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

**2.** Afişaţi numele angajatului, numele departamentului pentru toţi angajaţii care câştigă comision.

**3.** Afişaţi numele job-urile care există în departamentul 30.

**4.** Afişaţi numele, job-ul şi numele departamentului pentru toţi angajaţii care lucrează în Seattle.

```
SELECT last_name, job_id, department_name
FROM   employees e, departments d, locations s
WHERE  e.department_id = d.department_id
AND    d.location_id = s.location_id
AND    city = 'Seattle';
```

**5.** Afişaţi numele, salariul, data angajării şi numele departamentului pentru toţi programatorii (numele jobului este Programmer) care lucrează în America de Nord sau în America de Sud (numele regiunii este Americas).

**6.** Afişaţi numele salariaţilor şi numele departamentelor în care lucrează. Se vor afişa şi salariaţii al căror departament nu este cunoscut (*left outher join*).

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

**7.** Afişaţi numele departamentelor şi numele salariaţilor care lucrează în acestea. Se vor afişa şi departamentele care nu au salariaţi (*right outher join*).

**8.** Afişaţi numele departamentelor şi numele salariaţilor care lucrează în acestea. Se vor afişa şi salariaţii al căror departament nu este cunoscut, respectiv şi departamentele care nu au salariaţi (*full outher join*).
***Observaţie***: *full outer join = left outer join* UNION *right outer join*

**9.** Afişaţi numele, job-ul, numele departamentului, salariul şi grila de salarizare pentru toţi angajaţii.

**10.** Afişaţi codul angajatului şi numele acestuia, împreună cu numele şi codul şefului său direct. Etichetaţi coloanele CodAng, NumeAng, CodMgr, NumeMgr.

```
SELECT a.employee_id "CodAng", a.last_name "NumeAng",
       b.employee_id "CodMgr", b.last_name "NumeMgr"
FROM   employees a, employees b
WHERE  a.manager_id = b. employee_id;
```

**11.** Modificaţi cererea anterioară astfel încât să afişaţi toţi salariaţii, inclusiv pe cei care nu au şef.

**12.** Afişaţi numele salariatului şi data angajării împreună cu numele şi data angajării şefului direct pentru salariaţii care au fost angajaţi înaintea şefilor lor.

**13.** Pentru fiecare angajat din departamentele 20 şi 30 afişaţi numele, codul departamentului şi toţi colegii săi (salariaţii care lucrează în acelaşi departament cu el).

**14.** Afişaţi numele şi data angajării pentru salariaţii care au fost angajaţi după Fay.

```
SELECT last_name, hire_date
FROM   employees
WHERE  hire_date > (SELECT hire_date
                    FROM   employees
                    WHERE  last_name = 'Fay');
```

**15.** Rezolvaţi exerciţiul anterior utilizând join-uri.

```
SELECT a.last_name, a.hire_date
FROM   employees a, employees b
WHERE  UPPER(b.last_name)= 'FAY'
AND    a.hire_date>b.hire_date;
```

**16.** Scrieţi o cerere pentru a afişa numele şi salariul pentru toţi colegii (din acelaşi departament) lui Fay. Se va exclude Fay.

```
SELECT  last_name, salary
FROM    employees
WHERE   last_name <> 'Fay'
AND     department_id = (SELECT department_id
                         FROM employees
                         WHERE last_name = 'Fay');
```

**17.** Afişaţi numele şi salariul angajaţilor conduşi direct de Steven King.

```
SELECT last_name, salary
FROM   employees
WHERE  manager_id = (SELECT employee_id
                     FROM   employees
                     WHERE  UPPER(last_name) ='KING'
                     AND    UPPER(first_name) ='STEVEN');
```

**18.** Afişaţi numele şi job-ul tuturor angajaţilor din departamentul 'Sales'.

```
SELECT last_name, job_id
FROM   employees
WHERE  department_id = (SELECT department_id
                        FROM   departments
                        WHERE  department_name ='Sales');
```

**19.** Rezolvaţi exerciţiul anterior utilizând *join*-uri.

**20.** Afişaţi numele angajaţilor, numărul departamentului şi job-ul tuturor salariaţilor al căror departament este localizat în Seattle.

```
SELECT last_name, job_id, department_id
FROM   employees
WHERE  department_id IN
         (SELECT department_id
          FROM   departments
          WHERE  location_id = (SELECT location_id
                                FROM   locations
                                WHERE  city = 'Seattle'));
```

**21.** Rezolvaţi exerciţiul anterior utilizând *join*-uri.

**22.** Aflaţi dacă există angajaţi care nu lucrează în departamentul Sales, dar au aceleaşi câştiguri (salariu şi comision) ca şi un angajat din departamentul Sales.

```
SELECT last_name, salary, commission_pct, department_id
FROM   employees
WHERE (salary, commission_pct) IN
       (SELECT salary, commission_pct
        FROM   employees e, departments d
        WHERE  e.department_id = d.department_id
        AND    department_name = 'Sales')
        AND    department_id <> (SELECT department_id
                                 FROM   departments
                                 WHERE  department_name = 'Sales');
```

**23.** Scrieţi o cerere pentru a afişa angajaţii care câştigă mai mult decât oricare funcţionar. Sortaţi rezultatele după salariu, în ordine descrescătoare.

```
SELECT last_name, salary, job_id
FROM   employees
WHERE  salary > (SELECT MAX(salary)
                 FROM   employees
                 WHERE  job_id LIKE '%CLERK')
                 ORDER BY salary DESC;
```

**24.** Afişaţi salariaţii care au acelaşi manager ca şi angajatul având codul 140.

**25.** Afişaţi numele departamentelor care funcţionează în America.

**26.** Afişaţi numele angajatului, numele şefului său direct, respectiv numele şefului căruia i se subordonează şeful său direct.

**27.** Afişaţi codul, numele şi salariul tuturor angajaţilor care câştigă mai mult decât salariul mediu.

**28.** Afişaţi pentru fiecare salariat angajat în luna martie numele său, data angajării şi numele jobului.

**29.** Afişaţi pentru fiecare salariat al cărui câştig total lunar este mai mare decât 12000 numele său, câştigul total lunar şi numele departamentului în care lucrează.

**30.** Afişaţi pentru fiecare angajat codul său şi numele joburilor sale anterioare, precum şi intervalul de timp în care a lucrat pe jobul respectiv.

**31.** Modificaţi cererea de la punctul 30 astfel încât să se afişeze şi numele angajatului, respectiv codul jobului său curent.

**32.** Modificaţi cererea de la punctul 31 astfel încât să se afişeze şi numele jobului său curent.

**33.** Modificaţi cererea de la punctul 32 astfel încât să se afişeze informaţiile cerute doar pentru angajaţii care au lucrat în trecut pe acelaşi job pe care lucrează în prezent.

**34.** Modificaţi cererea de la punctul 33 astfel încât să se afişeze în plus numele departamentului în care a lucrat angajatul în trecut, respectiv numele departamentului în care lucrează în prezent.

**35.** Modificaţi cererea de la punctul 34 încât să se afişeze informaţiile cerute doar pentru angajaţii care au lucrat în trecut pe acelaşi job pe care lucrează în prezent, dar în departamente diferite.

**36.** Comparaţi sintaxa Oracle şi cu **standardul SQL3 (SQL 1999)**.

***Observaţie***: Sistemul Oracle oferă pentru join şi o sintaxă specifică, în conformitate cu standardul SQL3.

Sintaxa corespunzătoare standardului SQL3 este următoarea:

```
SELECT tabel_1.nume_coloană, tabel_2.nume_coloană
FROM   tabel_1
[CROSS JOIN tabel_2]
```

```
| [NATURAL JOIN tabel_2]
| [[INNER] JOIN tabel_2 USING (nume_coloană) ]
| [[INNER] JOIN tabel_2
        ON (tabel_1.nume_coloană = tabel_2.nume_coloană) ]
| [LEFT | RIGHT | FULL OUTER JOIN tabel_2
      USING (nume_coloană) |
      ON (tabel_1.nume_coloană = tabel_2.nume_coloană)];
```

Tipuri de *join*:

- **CROSS JOIN** – realizează produsul cartezian a două tabele;

```
SELECT a.region_name, b.region_name
FROM   regions a
CROSS JOIN regions b;

SELECT a.region_name, b.region_name
FROM   regions a, regions b;
```

- **NATURAL JOIN** – se bazează pe coloanele cu acelaşi nume din cele două tabele şi selectează liniile care au aceleaşi valori pentru aceste coloane;
  - Dacă tipurile de date ale coloanelor cu nume identice sunt diferite, va fi returnată o eroare.
  - Coloanele comune nu trebuie calificate cu aliasuri.
  - Oracle oferă posibilitatea completării automate a operaţiei de join.
  - Coloanele având acelaşi nume în cele două tabele trebuie să nu fie precedate de numele sau alias-ul tabelului corespunzător.

```
SELECT last_name, job_id, job_title
FROM   employees
NATURAL JOIN jobs;

SELECT last_name, e.job_id, job_title
FROM   employees e, jobs j
WHERE  e.job_id=j.job_id;
```

- **[INNER] JOIN … USING –** se bazează pe coloanele cu acelaşi nume din cele două tabele şi selectează liniile care au aceleaşi valori pentru aceste coloane;
  - Dacă cele două tabele conţin coloane cu acelaşi nume, dar cu tipurile de date diferite, atunci în loc de NATURAL JOIN se utilizează JOIN cu clauza USING.
  - Similar cu NATURAL JOIN, coloana de legătură nu trebuie calificată cu alias oriunde în cadrul cererii.

- Acest tip de JOIN se mai foloseşte atunci când cele două tabele au mai multe coloane cu nume comun, dar legătura dintre ele trebuie realizată doar utilizând câteva dintre acestea.

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

SELECT last_name, department_name, location_id
FROM   employees
INNER JOIN departments USING (department_id);

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

- **[INNER] JOIN … ON –** realizează compunerea a două tabele pe baza unei condiţii specificate;
  - Dacă se doreşte particularizarea, specificarea şi/sau evidenţierea clară a condiţiei de join se utilizează clauza ON. Aceasta separă condiţia de JOIN de alte condiţii, făcând codul mai lizibil.
  - Într-o înlănţuire de JOIN-uri acestea sunt evaluate de la stânga la dreapta.

```
SELECT department_name, city
FROM   departments d
JOIN locations l ON (d.location_id = l.location_id);

SELECT department_name, city
FROM   departments d
INNER JOIN locations l ON (d.location_id = l.location_id);

SELECT department_name, city
FROM   departments d, locations l
WHERE  d.location_id = l.location_id;
```

- **LEFT | RIGHT | FULL OUTER JOIN … USING | ON –** realizează compunerea externă a două tabele.
  - **left outer join**

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

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

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

- **right outer join**

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

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

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

- **full outer join**

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

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

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

**37.** Daţi o alternativă de rezolvare a exerciţiilor 25 - 35 implementând operaţia de join cu standardul SQL3.