# Baze de date, IESC – Îndrumar de Laborator

## Capitolul 4 – Interogarea bazelor de date

### Instrucţiunea SELECT

Instrucţiunea SELECT este instrucţiunea de interogare a bazei de date din SQL2, prin care regăsim informaţiile dorite din una sau mai multe tabele [FI]. Această instrucţiune are următoarea sintaxă generală:

SELECT [DISTINCT] lista_coloane
[FROM lista_tabele]
[WHERE condiție]
[clauze_secundare];

Ordinea apariției clauzelor din comanda Select este cea specificată în sintaxă (Where apare după lista tabelelor, iar alte clauze secundare apar după condiția logică).

Reamintim că elementele între paranteze drepte sunt opţionale, cu menţiunea că în Oracle clauza FROM este obligatorie, chiar dacă nu specificăm o tabelă reală, ci dorim să afişăm rezultatul unor calcule matematice, de exemplu:

SELECT 4*5 + 3 FROM DUAL;

Sau afişarea datei curente:

SELECT CURRENT_DATE FROM DUAL;

Sau a datei şi a timpului:

SELECT CURRENT_TIMESTAMP FROM DUAL;

Sau, mai complicat, utilizând o funcţie de conversie text cu specificarea formatului:

SELECT TO_CHAR(SYSDATE, 'MM-DD-YYYY HH24:MI:SS') "Data si ora curenta"

FROM DUAL;

Tabela DUAL este o tabelă virtuală specială (cu rol de variabilă auxiliară) care permite lucrul cu SELECT ca şi când am afişa valoarea unei variabile sau constante.

*Restricţia*

Operaţia de restricţie (sau selecţie) permite decuparea pe orizontală a conţinutului unei tabele, după o anumită condiţie specificată în clauza WHERE a instrucţiunii SELECT. Restricţia presupune că afişăm toate atributele (coloanele) tabelei.

Exemplu (testaţi direct):

- Afişaţi toţi angajaţii (din tabela Employees) care au salariul mai mare decât 5000:

SELECT * FROM Employees WHERE Salary > 5000;

**Exerciţiu:**

- Afişaţi toţi angajaţii care au salariul între 7000 şi 10000;
  *Notă:* aici se poate folosi mai elegant operatorul *between … and ….*
- Afişaţi toţi angajaţii care au prenumele *John*;
- Afişaţi toţi angajaţii care au fost angajaţi după 01-DEC-2007;
- Afişaţi toţi programatorii (job IT_PROG), apoi dintre aceştia, pe cei care câştigă mai mult de 5000;

- Afişaţi într-o singură instrucţiune Select doar angajaţii care au funcţia de Stock Clerk (ST_CLERK) şi pe cei care au funcţia de Marketing Manager (MK_MAN).

*Câteva funcţii speciale pentru lucrul cu şiruri de caractere*

- Funcţiile lower() şi upper() transformă şirurile de caractere în minuscule respectiv majuscule;

- Caracterul subliniere (underscore _ ) ţine locul unui singur caracter, iar caracterul % (procent) ţine locul la zero, mai multe, sau oricâte caractere;

*Exemplu:* afişarea tuturor persoanelor ale căror prenume încep cu litera *A* (sau *a*):

SELECT * FROM Employees WHERE upper(first_name) LIKE 'A%' ;

Există multe alte funcţii pentru şiruri de caractere, precum: initcap(), substr(), length() etc. pe care nu le detaliem în acest îndrumar.

**Exerciţiu:**

- Afişaţi persoanele care au caracterul ‚a' pe a treia poziţie în Prenume;

- Afişaţi toate persoanele care au în Job_ID cuvântul ‚clerk'.

*Proiecţia*

Proiecţia este operaţia din algebra relaţională care permite o decupare pe verticală a unei tabele, adică alegerea doar a unor coloane din aceasta.

Exemplu de proiecţie:

SELECT DISTINCT first_name, last_name FROM Employees;

*Alias-ul unei coloane:*

Reprezintă afişarea unei coloane cu alt nume decât cel din tabelă. De exemplu, putem relua instrucţiunea de mai sus afişând numele atributelor în limba română; testaţi direct:

SELECT DISTINCT first_name **as** Prenume, last_name **as** Nume FROM Employees ;

Sau, echivalent (AS fiind opţional în Oracle):

SELECT DISTINCT first_name Prenume, last_name Nume FROM Employees ;

De asemenea, putem folosi *operatorul de concatenare a şirurilor de caractere,* || , pentru a obţine afişări mai personalizate, precum (testaţi direct):

SELECT 'Angajatul ' || first_name || ' ' || last_name || ' are salariul ' || salary AS
"Informatii persoane" FROM Employees ;

Sau:

SELECT first_name || ' ' || last_name AS "Nume si Prenume", Salary Salariu FROM Employees;

Observaţie: Alias-ul „Informatii persoane" a fost scris cu ghilimele pentru a nu returna o eroare. Puteam folosi şi alias-ul Informatii_persoane.

Observaţie: Putem limita numărul de linii returnat de o comandă SELECT utilizând funcţia ROWNUM, ca în exemplul de mai jos, în care se afişează doar primele 3 linii:

SELECT first_name || ' ' || last_name AS "Nume si Prenume", Salary Salariu FROM Employees where ROWNUM<=3;

*Sortarea liniilor tabelei folosind clauza ORDER BY*

Putem ordona liniile afişate de instrucţiunea SELECT adăugând clauza ORDER BY. De exemplu, pentru a afişa toţi angajaţii în ordinea crescătoare a salariului putem executa comanda:

SELECT First_name, Last_name, Salary FROM Employees ORDER BY Salary;

Sortarea crescătoare este cea prestabilită. Dacă dorim o sortare descrescătoare putem adăuga opţiunea DESC după cum urmează:

SELECT First_name, Last_name, Salary FROM Employees ORDER BY Salary DESC;

Coloana după care se face ordonarea (Salary în exemplul de mai sus) nu trebuie neapărat să facă parte din proiecţie (din coloanele afişate). De asemenea, valoarea NULL este considerată cea mai mare valoare.

**Exerciţiu:** Afişaţi lista angajaţilor din Employees în ordinea crescătoare a numelui, apoi a prenumelui (angajaţii cu acelaşi nume să fie ordonaţi ascendent după prenume).

*Crearea unei tabele folosind instrucţiunea SELECT*

Prin execuţia unei instrucţiuni SELECT se poate obţine o tabelă nouă, conţinând rezultatul comenzii SELECT. Astfel, pentru a crea o tabelă „Programatori" preluând din tabela „Employees" numele şi prenumele angajaţilor cu Job_ID „IT_PROG" putem folosi instrucţiunea următoare:

```
CREATE TABLE Programatori AS
SELECT First_name Prenume, Last_Name Nume FROM  Employees
WHERE  Job_id = 'IT_PROG';
```

Observaţi utilizarea alias-urilor de coloană; verificaţi efectul instrucţiunii de mai sus prin afişarea conţinutului tabelei Programatori, şi a structurii acesteia (DESCRIBE Programatori;).

Ce observaţi referitor la tipul de date al coloanelor din noua tabelă?

*Definirea cu SELECT a unor coloane noi*

O facilitate a limbajului SQL este aceea că permite definirea (prin intermediul instrucţiunii SELECT) a unor coloane noi, calculate prin expresii conţinând alte nume de coloane sau constante.

De exemplu, dacă dorim afişarea unei coloane „Salariu net" lunar pentru angajaţii din tabela Employees putem utiliza codul următor:

SELECT First_name, Last_name, Salary as "Salariu brut", *0.7*Salary as "Salariu net"*

FROM Employees;

Atenţie: această nouă coloană nu poate fi utilizată ca atare în clauza where, ci condiţia poate fi impusă de exemplu: WHERE  0.7*Salary > 1000.

**Exerciţii:**

- Modificaţi instrucţiunea de mai sus astfel încât în loc de Salariu net să apară o coloană denumită „Propunere marire salariu", care să conţină un salariu brut majorat cu 20%, doar pentru toate cele 6 categorii de manageri. Verificare: tabela rezultat ar trebui să conţină 12 linii.

- Afişaţi pentru toţi angajaţii „salariul pe zi" folosind un alias de coloană. Notă: Puteţi utiliza funcţia ROUND(valoare, nr_zecimale);

Notă: Limbajul SQL include mai multe funcţii matematice, precum: valoare absolută ABS(*n*), restul împărţirii lui *m* la *n* MOD(*m*,*n*), rădăcină pătrată SQRT(*m*), funcţia putere POWER(*m*,*n*) etc.

*Funcţii agregat*

Funcţiile agregat, sau funcţiile de grup, permit obţinerea de date statistice din mai multe linii ale tabelei. Cele mai importante sunt:

- COUNT
  Întoarce numărul de înregistrări – COUNT(*) sau numărul de valori diferite de NULL – COUNT(expresie)
- AVG
  Calculează media aritmetică.

- MIN
  Întoarce valoarea minimă a unei expresii.
- MAX
  Întoarce valoarea maximă a unei expresii.
- SUM
  Calculează suma valorilor.

Exemple:

SELECT COUNT(*) FROM Employees; -- afişează numărul de linii din tabelă

Afişarea salariului minim, maxim şi mediu din toată tabela Employees:

SELECT MIN(Salary) Salariul_min, MAX(Salary) Salariul_max, ROUND(AVG(Salary),2) Salariul_mediu FROM Employees;

**Exerciţiu:** Afişaţi salariul minim, maxim şi mediu pentru programatorii din tabela Employees.

*Clauza GROUP BY*

Clauza GROUP BY se foloseşte pentru a grupa liniile unei tabele pe baza unor criterii în scopul calculării de valori statistice pentru fiecare grup în parte. Rezultatul obţinut va conţine câte o linie pentru fiecare grup identificat. [FR]

Exemplu:

Care este salariul mediu pe fiecare categorie de angajaţi din tabela Employees?

SELECT Job_ID, ROUND(AVG(Salary)) Salariul_mediu FROM Employees GROUP BY Job_ID;

Observaţii:

- GROUP BY se foloseşte doar atunci când avem nevoie de funcţii agregat. Dacă omitem clauza group by, funcțiile agregat se vor aplica la toate liniile tabelei.
- Coloana (sau coloanele) după care se realizează gruparea pot să nu apară în tabela rezultat (pot să nu fie specificate între SELECT și FROM).
- Nu se poate preciza in group by un alias de coloană.
- Numele de coloane specificate în SELECT trebuie să fie compatibile cu clauza de grupare. Astfel, următorul exemplu este **incorect**:

  SELECT First_name, AVG(Salary) FROM Employees

  GROUP BY Department_ID;

  Și va returna o eroare: ”not a GROUP BY expression”, deoarece coloana First_name conține valori diferite pentru aceeași valoare a ”Department_ID”.

*Clauza HAVING*

În cazul în care folosim clauza GROUP BY, nu putem folosi clauza WHERE. Clauza WHERE se înlocuieşte în această situaţie cu clauza HAVING, care are acelaşi efect ca şi WHERE, doar că este folosită împreună cu GROUP BY.

Exemplu:

Care sunt categoriile de angajaţi cu un salariu minim de cel puţin 7000?

```
SELECT Job_ID, MIN(Salary) Salariul_minim  FROM Employees
        GROUP BY   Job_ID
        HAVING  MIN(Salary) > 7000;
```

*Subinterogări*

Subinterogările sunt folosite pentru a returna anumite date către interogarea principală, şi sunt executate înaintea acesteia.

Exemplu: Să se găsească salariaţii din fiecare departament, care au salariul minim în departamentul respectiv [IP]:

```
SELECT First_name, Last_name, Salary, Department_ID
FROM Employees
WHERE (Department_ID, Salary) IN (SELECT Department_ID, MIN(Salary) FROM Employees
GROUP BY Department_ID);
```

Notă: Observaţi folosirea perechii (Department_ID, Salary) pentru a identifica persoanele care au salariul minim pe fiecare departament, şi a operatorului IN.

**Exerciţiu:** afişaţi (utilizând o sub-interogare) toţi angajaţii care lucrează în acelaşi departament cu angajatul cu numele Karen PARTNERS.

*Joncţiuni*

Joncţiunea este operaţia din algebra relaţională prin care putem combina liniile din două tabele într-o singură tabelă. Cel mai frecvent joncţiunea este implementată impunând egalitatea valorii cheii străine cu valoarea cheii primare (sau secundare) referite.

Exemplu: Afişaţi toţi angajaţii din tabela Employees împreună cu numele departamentului în care lucrează fiecare:

```
SELECT First_name, Last_name, Department_name   FROM  Employees, Departments
        WHERE  Employees.Department_ID = Departments.Department_ID;
```

## Probleme propuse

1) Afişaţi toţi angajaţii din tabela Employees care lucrează în departamentul cu numele ‚Sales'.
2) Afişaţi numărul de angajaţi care lucrează în departamentul ‚Sales'.
3) Afişaţi pentru fiecare angajat: Prenumele, Numele şi numele complet al job-ului (de exemplu: „Programmer".
4) Scrieţi şi rulaţi instrucţiunea SQL pentru a afişa într-un tabel cu 2 coloane, *Numele departamentului* şi *Număr de angajaţi*, pe câte o linie câţi angajaţi lucrează în fiecare departament. Notă: ar trebui să obţineţi un tabel cu 11 linii.
5) Scrieţi şi rulaţi instrucţiunea SQL pentru interogarea: „Care sunt numele departamentelor care au sediul în oraşul Seattle?".
6) Scrieţi şi rulaţi instrucţiunea SQL pentru interogarea: „Care sunt numele departamentelor care au sediul pe strada Arthur St?". Notă: folosiţi operatorul LIKE, deoarece adresa nu conţine doar denumirea străzii, ci şi numărul. De asemenea, puteţi utiliza funcţiile *upper* sau *lower* pentru a nu depinde de felul în care a fost introdusă adresa (litere mari sau mici).
7) Scrieţi şi rulaţi instrucţiunea SQL pentru interogarea: „Care sunt prenumele şi numele angajaţilor care lucrează în SUA sau Canada?". Afişaţi şi numele ţării, şi ordonaţi rezultatul după această coloană.