UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

Platformă de e-learning și curriculă e-content pentru învățământul superior tehnic

Baze de date 1

5. SQL - Data Manipulation Language (1)

STUD

MATR	NUME	AN	GRUPA	DATAN	LOC	TUTOR	PUNCTAJ	CODS
1456	GEORGE	4	1141A	12-MAR-82	BUCURESTI		2890	11
1325	VASILE	2	1122A	05-OCT-84	PITESTI	1456	390	11
1645	MARIA	3	1131B	17-JUN-83	PLOIESTI		1400	11
3145	ION	1	2112B	24-JAN-85	PLOIESTI	3251	1670	21
2146	STANCA	4	2141A	15-MAY-82	BUCURESTI		620	21
3251	ALEX	5	2153B	07-NOV-81	BRASOV		1570	21
2215	ELENA	2	2122A	29-AUG-84	BUCURESTI	2146	890	21
4311	ADRIAN	3	2431A	31-JUL-83	BUCURESTI		450	24
3514	FLOREA	5	2452B	03-FEB-81	BRASOV		3230	24
1925	OANA	2	2421A	20-DEC-84	BUCURESTI	4311	760	24
2101	MARIUS	1	2412B	02-SEP-85	PITESTI	3514	310	24
4705	VOICU	2	2421B	19-APR-84	BRASOV	4311	1290	24

SPEC si BURSA

CODS	NUME	DOMENIU
11	MATEMATICA	STIINTE EXACTE
21	GEOGRAFIE	UMANIST
24	ISTORIE	UMANIST

TIP	PMIN	PMAX	SUMA
FARA BURSA	0	399	
BURSA SOCIALA	400	899	100
BURSA DE STUDIU	900	1799	150
BURSA DE MERIT	1800	2499	200
BURSA DE EXCEPTIE	2500	9999	300

SINTAXA SELECT

```
SELECT [DISTINCT] lista_de_expresii
FROM nume_tabela
WHERE conditie_linie
       -- clauza optionala
ORDER BY criterii_sortare_rezultat;
       -- clauza optionala
```

EFECT

- Se parcurg rând pe rând liniile tabelei specificate pe clauza FROM.
- Din fiecare linie conţinând date pentru care condiţia aflată pe clauza WHERE este adevărată va rezulta o linie în rezultatul cererii. În cazul în care WHERE lipseşte, toate liniile tabelei FROM vor avea o linie corespondentă în rezultatul cererii.
- Linia de rezultat este compusă pe baza listei de expresii aflată pe clauza SELECT.

EFECT

- Dacă există cuvântul cheie DISTINCT, din rezultat se elimină liniile duplicat.
- Înainte de a trimite rezultatul, serverul îl sortează în funcţie de criteriile specificate de clauza ORDER BY.
- În cazul în care ORDER BY lipseşte, liniile din rezultat sunt într-o ordine independentă de conţinutul lor sau de ordinea în care ele au fost adăugate în tabelă.

REZULTAT

- Numărul coloanelor din rezultat este egal cu numărul expresiilor din lista aflată pe clauza SELECT. Aceste expresii dau şi numele coloanelor din rezultat.
- În lipsa clauzei DISTINCT, numărul de linii din rezultat este egal cu numărul liniilor din tabelă care îndeplinesc condiţia WHERE sau, când clauza respectivă lipseşte, cu numărul total de linii din tabelă.

REZULTAT

- Evaluarea valorii de adevăr a condiţiei din WHERE se face doar pe baza datelor aflate pe linia respectivă.
- Deoarece parcurgerea liniilor specificată de o cerere SELECT se face după un plan de execuţie generat de server, folosirea clauzei ORDER BY este obligatorie în cazul în care se doreşte un rezultat sortat după anumite criterii.

LISTA SELECT

- Nume de coloane sau *

```
SELECT NUME, DOMENIU
FROM SPEC;

SELECT *
FROM STUD;
```

LISTA SELECT

- Constante:

```
SELECT 'Specializarea ', NUME,
            ' infiintata in ', 1995
FROM SPEC
```

LISTA SELECT

- Expresii aritmetice:

```
SELECT TIP, SUMA, (SUMA+20)*1.1
FROM BURSA;
```

Functia NVL (MySQL: IFNULL)

```
SELECT TIP, SUMA,
            NVL((SUMA+20)*1.1, 0)
FROM BURSA;
```

LISTA SELECT

- Expresii concatenate:

```
SELECT 'Specializarea '|| NUME ||
            ' are codul ', CODS
FROM SPEC;
```

Cu valori nule:

```
SELECT TIP, ' are valoarea ' || SUMA ||
            '.Lei'
FROM BURSA;
```

LISTA SELECT

Alias de coloana:

- Nu poate fi mai lung de 30 de caractere.
- Incepe cu o literă, conţine decât litere, cifre, _, # şi $ sau e pus intre ghilimele (tot max. 30 caractere intre ghilimele).
- Între ghilimele literele mici sunt considerate diferite de literele mari.
- Nu poate fi folosit decât în cererea curentă. Sistemul nu stochează în baza de date sau altundeva aceste nume alternative.
- Nu poate fi folosit în alte clauze ale cererii (doar in SELECT si ORDER BY).

LISTA SELECT

Alias de coloana:

```
SELECT TIP AS "Tip bursa",
                ' are valoarea ' || SUMA ||
                '.Lei' AS Descriere
FROM BURSA;
```

```
Tip bursa            DESCRIERE
-------------------- --------------------------
FARA BURSA            are valoarea .Lei
BURSA SOCIALA         are valoarea 100.Lei
. . . . . . . . . . . . . . . . . . . . . . . .
```

LISTA SELECT

DISTINCT: Elimina liniile duplicat din rezultat:

```
SELECT CODS
FROM STUD;

SELECT DISTINCT CODS
FROM STUD;

SELECT DISTINCT CODS, AN
FROM STUD;
```

CLAUZA WHERE

Sintaxa: WHERE expresie_logica

Exemplu:

```
SELECT NUME, GRUPA, CODS
FROM STUD
WHERE AN = 4;
```

CLAUZA WHERE

Operatori de comparatie:

Operator	Semnificație
=	Egal
>	Mai mare
>=	Mai mare sau egal
<	Mai mic
<=	Mai mic sau egal
<>	Diferit
!=	Diferit
^=	Diferit

CLAUZA WHERE

Conditii compuse (AND, OR, NOT) si paranteze

- AN=2 AND PUNCTAJ>500 OR CODS=11
- AN=2 AND (PUNCTAJ>500 OR CODS=11)

CLAUZA WHERE

Operatorul BETWEEN:

Sintaxa:

expresie BETWEEN valoare_minima AND valoare_maxima

Exemplu:

```
SELECT NUME, AN, PUNCTAJ
FROM STUD
WHERE PUNCTAJ BETWEEN 2000 AND 4000;
```

CLAUZA WHERE

BETWEEN: Alte exemple

```
SELECT NUME, AN, PUNCTAJ
FROM STUD
WHERE PUNCTAJ + 100 BETWEEN TUTOR -
   2000 AND TUTOR + 1000;

SELECT NUME, LOC, DATAN
FROM STUD
WHERE LOC BETWEEN 'A' AND 'L' AND
      DATAN BETWEEN '1-JAN-82' AND '31-DEC-
   82';
```

CLAUZA WHERE

Operatorul IN:

Sintaxa:

expresie IN (val_1, val_2, ..., val_n)

Exemple:

```
SELECT NUME, AN, DATAN
FROM STUD
WHERE TUTOR IN (1456, 2146);
```

IN ignora valorile nule din lista:

```
SELECT NUME, AN, GRUPA, TUTOR
FROM STUD
WHERE TUTOR IN (NULL, 1456, 2146);
```

CLAUZA WHERE

NOT IN intoarce fals daca lista contine valori nule:

```
SELECT NUME, AN, GRUPA, TUTOR
FROM STUD
WHERE TUTOR NOT IN (NULL, 1456, 2146);
```

IN este operator derivat:

```
SELECT NUME, AN, DATAN
FROM STUD
WHERE TUTOR=1456 OR TUTOR=2146;
```

CLAUZA WHERE

Operatorul IN. Alte exemple:

```
SELECT NUME, PUNCTAJ, CODS
FROM STUD
WHERE PUNCTAJ + 10 IN (CODS*30+70,
   CODS*200+700);

SELECT NUME, LOC, DATAN
FROM STUD
WHERE LOC IN ('BUCURESTI', 'PLOIESTI') OR
      DATAN IN ('02-SEP-85', '19-APR-84', '29-
   AUG-84');
```

CLAUZA WHERE

Operatorul LIKE:

Sintaxa:

expresie LIKE 'SABLON' [ESCAPE 'caracter']

Caractere de inlocuire in sablon:

Caracter	**Semnificație**
_	Orice caracter
%	Orice şir de caractere, inclusiv şirul vid

CLAUZA WHERE

Operatorul LIKE: Exemple

```
SELECT NUME, AN, GRUPA
FROM STUD
WHERE NUME LIKE 'A%';

SELECT NUME, GRUPA
FROM STUD
WHERE NUME LIKE '____';

SELECT NUME, DOMENIU
FROM SPEC
WHERE DOMENIU LIKE '% %';
```

CLAUZA WHERE

Operatorul LIKE: Alte exemple:

```
SELECT NUME, DOMENIU
FROM SPEC
WHERE NUME LIKE '%A%I_';

SELECT NUME||'_'||DOMENIU AS
   NUMESIDOMENIU
FROM SPEC
WHERE NUME||'_'||DOMENIU LIKE '%\_U%'
   ESCAPE '\'
```

CLAUZA WHERE

Operatorul LIKE pentru numere, siruri, date:

```
SELECT NUME, DATAN, PUNCTAJ
FROM STUD
WHERE DATAN LIKE '%84' AND PUNCTAJ LIKE '%9_`
```

Valorile nule nu sunt considerate sirul vid

```
SELECT NUME, TUTOR
FROM STUD
WHERE TUTOR LIKE '%' OR
      TUTOR NOT LIKE '%';
```

Sablocul se poate obtine dintr-o expresie

```
SELECT NUME, 'A' || '%' || TUTOR AS SABLON
FROM STUD
WHERE NUME LIKE 'A' || '%' || TUTOR;
```

CLAUZA WHERE

Operatorul LIKE va considera literele mici si mari diferite in sablon (Oracle):

```
SELECT NUME, DATAN
FROM STUD
WHERE DATAN LIKE '%oct%';

SELECT NUME, DATAN
FROM STUD
WHERE DATAN LIKE '%OCT%';
```

CLAUZA WHERE

Operatorul IS NULL:
Sintaxa:
expresie IS NULL -- iar negata este:
expresie IS NOT NULL

Valorile nule nu se pot compara cu =, <>:

```
SELECT NUME, TUTOR
FROM STUD
WHERE TUTOR = NULL; -- fals mereu
```

```
SELECT NUME, TUTOR
FROM STUD
WHERE TUTOR <> NULL; -- fals mereu
```

CLAUZA WHERE

Operatorul IS NULL: Exemple:

```
SELECT NUME, TUTOR
FROM STUD
WHERE TUTOR IS NULL;
```

```
SELECT NUME, TUTOR
FROM STUD
WHERE TUTOR IS NOT NULL;
```

CLAUZA ORDER BY

Sintaxa:

ORDER BY criteriu1 [DESC] [,criteriu2 [DESC]...]

Cuvântul cheie opţional DESC (de la englezescul ***descending***) specifică inversarea ordinii de sortare implicite pentru criteriul respectiv (ordinea ascendentă, crescătoare) astfel încât sortarea se face descendent (descrescător).

EFECT

- În cazul în care ORDER BY conţine mai multe criterii de sortare, ele nu sunt echivalente ci se iau în considerare în ordinea specificată:
- Se sortează rezultatul după primul criteriu
- Pentru valori egale pentru primul criteriu se ia în considerare al doilea criteriu
- Pentru valori egale pentru primele două criterii se ia în considerare al treilea criteriu, s.a.m.d.

ORDER BY – coloane din rezultat

```
SELECT NUME, DOMENIU, CODS
FROM SPEC
ORDER BY NUME;

SELECT NUME, AN, GRUPA, DATAN,
   CODS
FROM STUD
ORDER BY AN DESC, NUME
```

ORDER BY – alias de coloana

```
SELECT NUME, PUNCTAJ,
  (PUNCTAJ+20)*1.1 PMARIT
FROM STUD
WHERE CODS=11
ORDER BY PMARIT;
```

ORDER BY – expresii (coloane si aliasuri)

```
SELECT NUME, PUNCTAJ, (PUNCTAJ+20)*1.1
   PMARIT
FROM STUD
WHERE CODS=11
ORDER BY (PUNCTAJ+20)*1.1;

SELECT NUME, PUNCTAJ, (PUNCTAJ+20)*1.1
   PMARIT
FROM STUD
WHERE CODS=11
ORDER BY PUNCTAJ-PMARIT;
```

ORDER BY – coloane care nu apar in rezultat

```
SELECT NUME, AN, GRUPA
FROM STUD
WHERE AN=2
ORDER BY LOC DESC, (PUNCTAJ/10);
```

ORDER BY – coloane care nu apar in rezultat (1)

```
SELECT MATR, NUME, AN
FROM STUD
ORDER BY 3 DESC, 2;

SELECT MATR, NUME, AN
FROM STUD
ORDER BY 3 DESC, NUME;
```

ORDER BY – coloane care nu apar in rezultat (2)

Numarul de coloana nu se poate da printr-o expresie:

```
SELECT MATR, NUME, AN
FROM STUD
ORDER BY 2+1 DESC, NUME;
```

ORDER BY – Valori nule (1)

Sunt considerate mai mari decat orice valoare (Oracle):

```
SELECT TIP, SUMA
FROM BURSA
ORDER BY SUMA
```

ORDER BY – Valori nule (2)

Rezultat:

```
TIP                          SUMA
-------------------- -----
BURSA SOCIALA                100
BURSA DE STUDIU              150
BURSA DE MERIT               200
BURSA DE EXCEPTIE            300
FARA BURSA
```

Cereri SELECT pe mai multe tabele

- În foarte multe cazuri se doreşte ca un acelaşi rezultat să conţină date care sunt stocate în două sau mai multe tabele din baza de date, ca în exemplele următoare:
- Lista cu nume studenţi şi denumiri specializări. Tabelele care conţin aceste date sunt STUD (numele studentului) şi SPEC (denumirea specializării).
- Numele studenţilor (din tabela STUD) şi tipul burselor acestora (din tabela BURSA).
- Numele studentului, denumirea specializării şi cuantumul bursei. În acest caz sunt implicate toate cele trei tabele ale bazei de date de test.

JOIN

- Operaţia care permite astfel de regăsiri se numeşte ***join*** (termen preluat din limba engleză) şi este realizată prin intermediul unei cereri SELECT având următoarele caracteristici:
- În clauza FROM este specificată nu doar o singură tabelă ci o listă de tabele.
- În clauza WHERE există o condiţie care să coreleze liniile tabelelor din lista FROM (condiţie numită şi ***condiţie de join***).

SINTAXA

```
SELECT [DISTINCT] lista_de_expresii
  FROM lista_de_tabele
  WHERE conditie_de_join
            AND conditii_suplimentare
. . .
-- alte clauze: GROUP BY, HAVING, ORDER
  BY
```

OBSERVATII

- Atunci când condiţia de join lipseşte, fiecare linie a unei tabele din lista FROM este concatenată cu fiecare linie a celorlalte tabele, obţinându-se de fapt ***produsul cartezian*** al acestora.
- Dacă în condiţia de join apar numai egalităţi operaţia este numită şi ***echijoin***. În celelalte cazuri avem un ***non-echijoin***.
- În lista de tabele care participă la join o tabelă poate să apară repetat. O astfel de operaţie este numită şi ***joinul unei tabele cu ea însăşi*** sau ***self-join***.

OBSERVATII (2)

- În cazul în care o linie a unei tabele nu se corelează prin condiţia de join cu nici o linie din celelalte tabele ea nu va participa la formarea rezultatului. Se poate însă cere ca aceasta să fie luată în considerare pentru rezultat, rezultând aşa numitul ***join extern*** (în engleză ***outer join***).
- În versiunile anterioare sintaxa joinului în Oracle era diferită de standardul ANSI. Începând cu versiunea Oracle 9i au fost introduse în limbaj şi tipurile de join din standardul SQL:1999 (SQL-3) printre care ***cross-join, join natural*** şi mai multe ***variante de join extern.***

PRODUS CARTEZIAN

Cererea:

```
SELECT *
FROM STUD, SPEC;
```

va returna un rezultat având 12 coloane şi 36 de linii formate din concatenarea fiecărei linii din STUD cu fiecare linie din SPEC.

De asemenea, cererea:

```
SELECT DATAN, DOMENIU
FROM STUD, SPEC
WHERE LOC='BUCURESTI';
```

are un rezultat conţinând 15 linii

ECHIJOIN

```
SELECT MATR, STUD.NUME, STUD.CODS,
  SPEC.NUME
FROM STUD, SPEC
WHERE STUD.CODS = SPEC.CODS AND
DOMENIU='STIINTE EXACTE`
```

- Este marcata conditia de join. Restul conditiei este suplimentar

NON-ECHIJOIN

```
SELECT NUME, AN, TIP, SUMA
FROM STUD, BURSA
WHERE PUNCTAJ BETWEEN PMIN AND PMAX
        AND CODS=11
```

- Este marcata conditia de join

ALIAS DE TABELA

```
SELECT S.NUME, S.CODS, "SP
  STUD".NUME, TIP
FROM STUD S, SPEC "SP STUD", BURSA
WHERE S.CODS = "SP STUD".CODS AND
  S.PUNCTAJ BETWEEN PMIN AND PMAX
```

- Aliasul trebuie sa indeplineasca anumite conditii (Oracle):

ALIAS DE TABELA (2)

- Nu poate fi mai lung de 30 de caractere.
- În cazul în care nu începe cu o literă sau când conţine alte caractere dacât litere, cifre, _, # şi $ trebuie pus între ghilimele (de exemplu atunci când conţine spaţii).
- În cazul unui alias între ghilimele, dimensiunea maximă de 30 de caractere nu include ghilimelele.

ALIAS DE TABELA (3)

- Între ghilimele literele mici sunt considerate diferite de literele mari.
- Nu poate fi folosit decât în cererea curentă. Sistemul nu stochează în baza de date sau altundeva aceste nume alternative.
- În cazul în care o tabelă are asociat un alias prin clauza FROM acesta poate şi trebuie să fie folosit pentru prefixare în toate celelalte clauze ale cererii.

ALIAS DE TABELA (4)

- În cazul unui alias între ghilimele literele mici sunt considerate diferite de literele mari. Din această cauză dacă prima linie a cererii anterioare este:

```
SELECT S.NUME, S.CODS, "Sp Stud".NUME,
    TIP
```

- se obţine eroare

ALIAS DE TABELA (4)

- Dacă pentru o tabelă a fost definit un alias numele tabelei nu mai poate fi folosit pentru prefixare. În cazul cererii anterioare, prima linie nu mai poate fi rescrisă astfel:

```
SELECT S.NUME, STUD.CODS,"Sp Stud".NUME,
    TIP
```

JOIN – cont.

- În cazul în care condiţia de join nu este completă rezultatul va conţine un produs cartezian între joinurile existente şi restul tabelelor.
- Exemplu: în cererea următoare lipseşte condiţia de join care leagă tabela BURSA de celelalte tabele.

```
SELECT S.NUME, S.CODS, "SP STUD".NUME,
    TIP
FROM STUD S, SPEC "SP STUD", BURSA
WHERE S.CODS = "SP STUD".CODS;
```

- În acest caz rezultatul obţinut este produsul cartezian între BURSA şi joinul lui STUD cu SPEC şi va avea 60 de linii (5 linii din BURSA * 12 linii ale joinului STUD cu SPEC).

JOIN – cont.

1. În cazul general al unui join pe ***N*** tabele, condiţia de join este compusă din ***N - 1*** subcondiţii conectate prin AND care relaţionează întreg ansamblul de tabele. Altfel spus, dacă se construieşte un graf al condiţiei în care nodurile sunt tabele şi arcele subcondiţii de join care leagă două tabele atunci acest graf trebuie să fie conex.

JOINUL UNEI TABELE CU EA INSASI

- Este obligatorie folosirea aliasurilor:

```
SELECT S.NUME "NUME STUD",
       S.AN "AN STUD",
       T.NUME "NUME TUTOR",
       T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.TUTOR=T.MATR
```

ALT EXEMPLU

- Studenti care au acelasi tutor:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR;
```

- INCORECT! Apar si cupluri cu acelasi student

ALT EXEMPLU - cont

- Alta varianta:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR
      AND S1.MATR <> S2.MATR
```

- INCORECT! Apar ambele cupluri XY si YX

ALT EXEMPLU - cont

- Varianta corecta:

```
SELECT S1.NUME "STUDENT 1",
       S2.NUME "STUDENT 2",
       S2.TUTOR "TUTOR"
FROM STUD S1, STUD S2
WHERE S1.TUTOR=S2.TUTOR
      AND S1.MATR > S2.MATR
```

JOIN EXTERN (Oracle)

- Apar si studentii fara tutor:

```
SELECT S.NUME "NUME STUD", S.AN "AN
  STUD", T.NUME "NUME TUTOR",
  T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.CODS = 11 AND
      S.TUTOR=T.MATR(+)
```

JOIN EXTERN (Oracle)

- Rezultat:

NUME STUD	AN STUD	NUME TUTOR	AN TUTOR
VASILE	2	GEORGE	4
GEORGE	4		
MARIA	3		

JOIN EXTERN (Oracle)

- Daca schimbam plusul apar studentii care nu sunt tutori:

```
SELECT S.NUME "NUME STUD", S.AN "AN
  STUD", T.NUME "NUME TUTOR",
  T.AN "AN TUTOR"
FROM STUD S, STUD T
WHERE S.CODS = 11 AND
      S.TUTOR(+)=T.MATR
```

JOIN EXTERN FARA =

- Joinul extern se poate folosi şi în cazul în care condiţia nu este de egalitate. Se pot folosi operatorii <, <=, >, >= sau <>. Un exemplu în acest sens este următorul: cererea

```
SELECT S1.NUME "STUDENT 1",
  S1.PUNCTAJ "PUNCTAJ 1",
   S2.NUME "STUDENT 2",
   S2.PUNCTAJ "PUNCTAJ 2"
FROM STUD S1, STUD S2
WHERE S1.PUNCTAJ > S2.PUNCTAJ;
```

JOIN EXTERN FARA = (cont.)

- Asa apar toate perechile (66) plus inca o linie cu studentul avand cel mai mare punctaj:

```
SELECT S1.NUME "STUDENT 1",
  S1.PUNCTAJ "PUNCTAJ 1",
   S2.NUME "STUDENT 2",
   S2.PUNCTAJ "PUNCTAJ 2"
FROM STUD S1, STUD S2
WHERE S1.PUNCTAJ (+) > S2.PUNCTAJ;
```

OBSERVATIE

- Marcajul de join extern se poate folosi şi atunci când condiţia de join este compusă, cu excepţia cazului în care se foloseşte sau logic (OR) sau operatorul de incluziune IN urmat de o listă care conţine mai mult de o valoare.
- Joinul extern se poate folosi şi în conjuncţie cu operatorii specifici SQL

OBSERVATIE

- In Oracle o condiţie de tipul:

```
WHERE S.TUTOR(+)=T.MATR(+)
```

- va produce eroarea: *ORA-01468: a predicate may reference only one outer-joined table.*

Bibliografie

1. **Hector Garcia-Molina, Jeffrey D. Ullman, Jennifer D. Widom**: *Database Systems: The Complete Book*, Prentice-Hall, Englewood Cliffs, NJ, 2002.
2. **F. Rădulescu** : *Oracle SQL, PL/SQL*, Editura Printech, ISBN 973-718-203-02005