

Sept 8

Cereri la mai multe tabele

FROM lista tabele

[WHERE conditie JOIN]Produs Cartezian dacă lipsește cond. JOIN
↓

nu este suportat de majoritatea tipurilor de BD

ex:

1) SELECT Nume, Prenume FROM Angajat, Departament WHERE Dep-MR = 5

↓
cond de tip
select în
acest caz

2)

SELECT Nume, Prenume FROM Angajat, Departament WHERE CNP = Manager and Dep-Nume = "Cercetare"
↑ ↓
(cond JOIN simplă)

Alias la tabele

În unele cazuri cereri select pot fi anumite \Rightarrow pt a specifica condițiile necesare trebuie utilizat un indicator la tabela.SELECT Angajat.Nume, Angajat.Prenume FROM Angajat,
Departament WHERE Angajat.CNP = Departament.Manager and
Dep-Nume = "Cercetare"↓
utilizarea este optională
calificatoruluiex: să se obțină numele și prenumele angaj., prenume și
numele și pren. superviz. acestora.SELECT A.Nume, A.Prenume, B.Nume, Numes, B.Prenume
Prenume AS FROM Angajat A, B
WHERE
(Angajat B)

A.CNP = B.CNP

Dacă la o tabelă s-a definit un alias explicit acesta tb. utilizat pt calificarea tuturor cāmpurilor din tabela resp.

ex: Presup. că dorim să obținem Nume, Pren. angaj. care lucrează la proiecte ce sunt coord. de Departamente din care Angaj. face parte.



Join cu trei tabele

SELECT FROM Angajat A, Proiect B, Lucrează L
A.Nume, A.Prenume

WHERE A.Cnp = L.Cnp and A.depnr = B.depnr and
B.PNr = C.PNr

Dacă la clauza FROM avem N tabele multile care nu suportă produs cartezian necesită N-1 cond. JOIN minimu.

OR \Leftrightarrow union (op. union din Algebra Relat.)



dacă se utiliză JOIN și OR

JOIN ADITIONAL (outer Join)

Se mai numește Join extern pt că sunt incluse și înreg. unei sau mai multor tabele ce nu îndepl. cond. Join, dar combinate cu înreg nule.

Ace particularitate se găsește în implementarea ORACLE până în prezent. Si (peste și (\exists) și alte posibilități de Join extern cāmpul tabeliei dia care conține de Join neîndepl. det. adăug. de înreg cu val NULL nu are sufixul (+)

ex: să se devină Nume, Pren. angaj și cel al supervisorilor, dar rezultatul să includă și pe cei care nu au supervisor

SELECT A.Nume, A.Prenume, B.Nume, B.Prenume PS
FROM Angajat A, Angajat B WHERE A.Cnp(+)=B.Cnp

↓
se includ și angaj car.
nu au supervisor

OBS: 1) F. important la ce cămpuri se adaugă inclusarea (-)
2) nu se poate specifica $A.Cnp(+) = B.Cnp(+)$ pt struct
de mai sus până la ver 9i.

Alte forme de JOIN incluse în standardul după 9i

① CROSS JOIN:

SELECT [DISTINCT] lista_exprești FROM Tabel1 CROSS JOIN
Tabel2 [WHERE] condiții]

↳ condiții select (setul combinat
cu Prod Cart.)
condiții JOIN

Se poate aplica numai pt 2 tabele.

ex: să se det. numele și pren angaj și numele Dep din care
acestia fac parte.

SELECT Nume, Prenume, Dep_Nume FROM Angajat CROSS JOIN
Departament WHERE Dep_Nr = D_Nr AND D_Nr IN (2,5)
(daca se remuntră la cond where => un produs cartezian)

② JOIN ... USING

Op. de JOIN între 2 tabele cu cond de JOIN de egalitate
intre cămpurile specif. la clauza USING, cămpuri se același nume în cele 2 tabele.

Angajat(Nume, Prenume, ..., D-Nr, ...)

Departament(Nume, D-loc, ..., D-Nr, ...)

```
SELECT A.Nume, A.Prenume, B.Nume AS Dep_Nume  
FROM Angajat A, Departament B JOIN using Departament E  
USING (D_Nr)
```

↓
lista cămpurilor cu nume egale
nu permite utiliz. aliasurilor în cond pt using
nu permite scrierea expresiilor
 Mai multe cămpuri using (Nume, D_Nr) (\Rightarrow) $\begin{cases} A.Nume = \\ B.Nume \end{cases}$
nu este permis scrierea WHERE
 $\begin{cases} A.D_Nr = \\ B.D_Nr \end{cases}$
operătorul

OBS: Join using nu permite scrierea ~~clauzelor de tip~~
SELECT ; ↓
op join + eventual op Project

③ JOIN...ON

Op de join între 2 tabele, structura este:

```
SELECT [DISTINCT] lista expresii FROM Tabela1 JOIN Tabelă2  
ON (condiții join)
```

ex:

1) SELECT Nume, Prenume, Nume NumeM, Prenume Preznt
FROM Angajat JOIN Departament ON (Dep_Nr = D_Nr
AND Manager = cnp)
(afisează angaj. care sunt manageri)

se pot scrie nu
numai condiție de
egalitate

2) numele și pren angaj. care lucrează la un proiect mai
mult de 20 ore.

④ NATURAL JOIN

Op de join cu egalit. cămpurilor cu același nume din:
tabele invocate.

```
SELECT [DISTINCT] lista expresii FROM Tabela1 NATURAL  
JOIN TABELA2
```

SELECT Name, Prenume, D-Nume FROM Angajat NATURAL
JOIN Departament

OBS:

SELECT * FROM Angajat NATURAL JOIN Departament

SELECT * FROM Angajat, Departament WHERE D-Nr = Dep-Nr

rez are n coloane ((7) 2 coloane identice D-Nr, Dep-Nr)

rez are n-1 coloane fata de Natural JOIN

Outer JOIN:

OUTER JOIN

SELECT [DISTINCT] lista expresii FROM Tabela1] LEFT { Tabela
ON (conditie) } RIGHT
ALL }

- 1) LEFT → inreg cu val. NULLE provin din tabela 2
- 2) RIGHT → inreg cu val. nule din tabela 1
- 3) ALL → este echiv. cu reuniunea a 2 op JOIN: LEFT și
RIGHT.

ORACLE: suportă atât cu cheie ALL, cât și FULL

OBS:

1) Alias-ul unei tabele are efect local (nu se salvează în BD
are efect doar în interog. curentă)

2) Tb. nu respectă restricțiile privit. la def. numelui tabeliei ex.
at nu depăș. 30 caract, nu fie num. rezervat; aliasul poate
cuprinde blanc ⇒ se va scrie între "", situație în care
ORACLE tratează alias-ul case sensitive.

SELECT "Angajat".Name, "Angajat".Prenume, D-Nume
FROM Angajat "Angajat", Departament
WHERE "Angajat".Dep-Nr = D-Nr

dacă apare "angajat", ... apare eroare este Case Sensitive

↓
dacă nu se folosesc "" nu este case sensitive, iar folosirea
LJ implică folosirea "".

La orice JOIN se poate utiliza clauza ORDER BY (viz. curs 6)

Functii statistice

Op. cu functii statistice se pot include intr-o cerere la o tabela sau la mai multe tabele

MIN, MAX, AVG, COUNT, SUM

STD DEV, COR



corelatie

→ in ORACLE (?) in plus

standard deviation

→ nu are efect

[MIN ([ALL | DISTINCT] nume camp)]

[MAX ([ALL | DISTINCT] nume camp)]



neprecizarea atributului → implicit ALL

OBS: functiile min, max nu se aplică pe câmpurile cu val. nu indiferent de precizia atributului; dacă se prez. DISTINCT atunci MIN, MAX nu sunt afectate dacă nu s-ar prezisa. Pot fi aplicate numai pe câmpuri ce au tipuri de date ce pot fi ordonate.

COUNT (*) → numără toate înreg. stabelei indif. dacă avem (ALL) val. nule sau nu

COUNT ([ALL | DISTINCT] nume camp) → nr. înreg ce au val. diferit de nulla pt câmpul precizat dacă nu se prez. atribut sau se prez. ALL, resp. va numără înreg care au val. dif. de null dacă se prez. DISTINCT și val. distincte (val. distincte și dif. de null vor fi numărate → DISTINCT)

SUM ([ALL | DISTINCT]) nume camp)

AVG ([ALL | DISTINCT]) nume camp)

Sunt posibile numai pt câmpuri cu val. numerice, dacă prez. DISTINCT → numai pt înreg cu valori distincte, iar dacă nu, se aplică numai înreg cu val. nule.

Nu întotdeauna sum, count (=) avg.

: salariu min, max , nr angaj companiei

OBS: SELECT MIN(salariu), MAX(salariu), COUNT(cnp) FROM Angajat

1) COUNT(CNP) = COUNT(*) pt că cnp = cheie primară

2) utilizarea în această formă a fct. va produce o tabelă cu nr. ang. → combinațiile tabelei sunt formate de numele combinatoriu la care s-a aplicat fct. prefixat de numele fct, altfel se utilizează alias.

3) Fct. se utiliz. în conjuncție cu o clauză specif. GROUP BY dacă se dorește ca fct. să se aplique doar pt angaj care indep. cond. de grupare.

ex: să se devă pt fiecare depart. nr angaj și salariul mediu pe departament.

SELECT D-Nr, COUNT(cnp), AVG(salariu) FROM Angajat GROUP BY D-Nr
⇒ { Dep 1 nr angaj salariu mediu
 Dep 2 -/- -/-
 :
 Depn }

ex: nr de angaj pt fiecare dep care au același salariu

SELECT D-Nr, Salariu, COUNT(*) FROM Angajat GROUP BY D-Nr, Salariu

(dacă nu apare Salariu la select → D-Nr apare de atâtea ori câte salariu dist (7))

SELECT D-Nr, MIN(Salariu), MAX(Salariu) FROM Angajat GROUP BY D-Nr

→ dacă se utiliză MIN, MAX ⇒ dau ordinea implicită în re sunt ordonate după funcții ⇒ dacă se dorește ordonarea după D-Nr se adaugă ORDER BY D-Nr.

Condiții impuse asupra funcțiilor

Condițiile referit la rezultat nu se pot apăsa în clauza WHERE → în medale BD se utiliză clauza HAVING se pot pune condiții pt. rezultate.

Ex: să se doată numele și prenumele angajatilor care sumă orelor săpt. aleșă. La proiecte este mai multă decât 40, lista angajatorilor fiind ordonată crescător după suma orelor.

```
SELECT Nume, Prenume, SUM(ore) FROM Angajat, Lucreaza  
WHERE Cnp = cnp GROUP BY cnp HAVING SUM(ore) < 40  
ORDER BY SUM(ore) DESC
```

→ dacă nu se doarează să apară în rezultat suma orelor ⇒ nu se trage la SELECT SUM(ore)

→ sau SUM(ore) AS Sumăore ⇒ singurul loc unde se poate utiliza alias doar la ORDER BY (alias de col.) nu se pot utiliza în HAVING etc.

Ex: să se obțină numele și prenumele angajatorilor care lucrează la proiecte une ore mai multe decât media orelor aleșă la proiecte de toti angajatorii companiei.

```
SELECT Nume, Prenume  
FROM Angajat, Lucreaza WHERE cnp = cnp GROUP BY cnp  
HAVING SUM(ore) > (SELECT AVG(ore) FROM Lucreaza)
```

rezultatul subcereri

din toată
compania

→ nu întotdeauna se pot pune astfel de condiții

Ex: să se doată proiectele coord. de dep. la care Popescu este manager pt care suma orelor tuturor angajatorilor care lucrează la proiectele resp. > 200.

SELECT A.P_Nume, SUM(ore)
 FROM Angajat B, Proiect A, Departament C, Lucreaza D
 WHERE B.Nume = "Popescu" and B.cnup = C.manager and
 C.D_Nr = A.D_Nr and A.P_Nr = D.P_Nr
 GROUP BY D.P_Nr
 HAVING SUM(ore) > 200
 ORDER BY SUM(ore) % ordonarea s-ar face crescator

OBS: 1) este posibil sa cererea sa contina val NULL la ORDER BY => fiecare baza de date si-a stabilit o strategie in vedere ordonarii pt val NULL (ex: ORACLE: NULL val se va mai multe 2) Toate cererile SQL presupun o sa parcurgerea a tabelui => altfel apar cereri corelate (mai multe parcurgeri a tabelui)
 ex: sa se obtin numele si prenumele angajatilor care lucreaza la proiecte cu nr 2, cel putin la 3 proiecte si la fiecare dintre acestea nr de ore > 10.

SELECT A.Nume, A.Prenume
 FROM Angajat A, Proiect P, Lucreaza L,
 WHERE P.D_Nr = 2 and P.P_Nr = L.P_Nr and L.cnup = A.cnup
 GROUP BY L.cnup
 and L.ore > 10
 HAVING COUNT(L.cnup) > 3

3) daca nu se foloseste clauza GROUP BY si apare in cerere HAVING => rez executiei poate avea o sa intreg daca cond de HAVING este indepl., sau o tabela vidala daca cond este FALSE.

ex: sa se obtina nr angajati conponuici, nr > 1000

SELECT COUNT(*) FROM Angajat HAVING COUNT(*) > 1000
 ↓

se obtine o tabela cu o sa intreg in coloana count
 → daca nr > 1000 => intreg continut nr. angajat
 → altfel, vides

Exercițiu:

- ① Sa se găsească angaj. comp. și numele dep. clinc. care fac parte care lucrează mai mult de 10 ore la proiecte la care lucrează și managerul dep. sau.
- ② Sa se det. angaj. ce au cel puțin 2 pers. în echip., care lucrează la proiecte uneori de ore > 20 (suma orelor la toate pr.).
- ③ At se det. proiect. ce au beneficiarul B la care lucrează c. puțin 5 angajați.
- ④ At se det. numele și pren. angaj., numele și pren. supr. acestora, precum și suma nr de ore desf. de angaj.-la toate proiectele la care lucrează ei.

OBS: 1) la clauza HAVING se pot efectua și operații (atunci de apăr condiții)

2) ! SUBCERERI (intorc o val., o tabelă cu o coloană - un singur, o tabelă)