UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

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

Proiect nr. 154/323 cod SMIS – 4428 cofinanțat de prin Fondul European de Dezvoltare Regională “Investiții pentru viitorul dumneavoastră”.

Programul Operaţional Sectorial Creşterea Competitivităţii Economice - POS CCE

UNIUNEA EUROPEANĂ

GUVERNUL ROMÂNIEI

Instrumente Structurale
2007-2013

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

Baze de date

21. Interogări la mai multe tabele

Introducere

- Pentru extragerea datelor din mai multe tabele din baza de date, comanda SELECT foloseşte una sau mai multe metode de JOIN. Sintaxa pentru un join simplu este următoarea:
- **SELECT [DISTINCT,ALL] [table].column expr_alias**

  FROM [schema.table1] table1_alias, [schema.table2] table2_alias, ….

  WHERE table1_alias.colum= table2_alias.colum AND ….

  ORDER BY expresion(position)] [ASC,DESC]

Parametrii de interogare

- Parametrii comenzii au următoarea semnificație(cei din paranteze sunt opționali):
- ***DISTINCT*** - returnează numai o înregistrare, în cazul în care comanda găseşte rânduri duplicate;
- ***ALL*** – returnează toate înregistrările simple şi duplicate (selectează de asemenea toate coloanele tabelelor din clauza FROM);
- ***schema.table*** – reprezintă shema de identificare a tabelei(view-lui) specificată prin *user.table_name;*
- ***expr_alias*** – este un nume alocat unei expresii, care va fi folosit în formatarea coloanei (apare în antetul listei);
- ***table_alias*** – este un nume alocat unei tabele(view), care va fi folosit în cereri corelate;
- ***WHERE condition*** – reprezintă o clauză (înlănțuire de condiții), care trebuie să fie îndeplinită în criteriul de selecție a înregistrărilor;
- ***ORDER BY expresion(position)*** – ordonează înregistrările selectate după coloanele din expresie, sau în ordinea coloanelor specificate prin poziție;

Equi-join

- Dacă în condiția de join apar numai egalități, avem de-a face cu un *equi-join*. Pentru a putea să realizăm un join pe mai multe tabele, este obligatoriu ca ele să conțină coloane de acelaşi tip cu date comune sau corelate.
- Să luam un exemplu simplu, pentru a arăta cum funcționează un join. Pentru a lista angajații dintr-un departament, folosim tabelul *angajati,* însă în aceast tabel găsim asociat fiecărui angajat un id_dep, iar denumirea departamentului respectiv o găsim în tabelul *departamente,* deci se impune un join între cele două tabele, pentru ca în listă să apară şi denumirea departamentului.

Exemplu:

- SQL>SELECT a.id_dep ,b.den_dep departament,

  a.nume, a.functie

  FROM angajati a, departamente b

  WHERE a.id_dep=b.id_dep and a.id_dep=10;

- Se observă că au fost folosite aliasuri pentru tabele, pentru a nu crea ambiguitate, când referim coloane cu aceeaşi denumire.

Non Equi-join

- Când două sau mai multe tabele nu au coloane comune şi trebuie totuşi relaționate, avem un *non equi-join*;
- Relația dintre tabelele *angajati* şi *grila_salar* este un *non-equi-join*, în care nicio coloană din primul tabel nu corespunde direct cu o coloană din celălalt;
- Relația se obține folosind tot un operator (altul decât '**=**'), de exemplu *between*;
- Sunt permise folosirea de *equi-join* şi *non equi-join* într-o interogare complexă.

Exemple:

- Pentru a evalua gradul de salarizare al unui anajat, trebuie să consultăm grila de salarizare, pentru a identifica în ce plajă de salariu se situează salariul:
- SQL>SELECT a.nume, a.salariu, b.grad

  FROM angajati a, grila_salar b

  WHERE a.salariu BETWEEN b.nivel_inf AND b.nivel_sup

  AND a.id_dep=20;
- În exemplul anterior, dacă dorim să listăm şi departamentul, va trebui să facem join după 3 tabele:
- SQL>SELECT c.den_dep,a.nume, a.salariu, b.grad

  FROM angajati a, grila_salar b, departamente c

  WHERE a.salariu BETWEEN b.nivel_inf AND b.nivel_sup

  AND a.id_dep=c.id_dep AND a.id_dep=20;

Joinul unui tabel cu el însuşi

- Sunt situații când avem nevoie să extragem date corelate din acelaşi tabel. De exemplu, dacă dorim să afişăm care sunt şefii angajaților, trebuie să extragem din tabelul *angajati* şi numele şefului (id_sef).
- SQL>SELECT a.nume nume_ang,a.functie functie_ang,

  b.nume nume_sef,b.functie functie_sef

  FROM angajati a, angajati b

  WHERE a.id_sef=b.id_ang AND a.id_dep=10;

Produs cartezian

- Produsul cartezian a două tabele se obține prin concatenarea fiecărei linii dintr-o coloană cu fiecare linie din cealaltă, rezultând un număr de linii egal cu produsul numărului de linii din fiecare tabelă. Această situație este mai putin practică şi se întâlneşte, de regulă, când sunt puse greşit condiții în clauza *WHERE*.

Exemplu:

- SQL>SELECT nume ,functie ,den_dep

  FROM angajati , departamente

  WHERE functie='DIRECTOR';

Join extern

- Folosind equi-join, putem selecta toate înregistrările care îndeplinesc condițiile din clauza WHERE. Apar situații când cererea trebuie să selecteze şi înregistrări care nu îndeplinesc toate condițiile din clauză. Un exemplu ar fi să construim structura organizatorică a firmei selectând toate departamentele şi angajații care fac parte din fiecare departament. Există, în tabela departamente, departamentul 40 care nu are niciun angajat şi folosind equi-join acesta nu apare în listă. Pentru a depăşi situația, se foloseşte un *join extern* (+), aşa cum se vede în exemplul următor:
- SQL>SELECT a.id_dep,a.den_dep,b.nume,b.functie

  FROM departamente a, angajati b

  WHERE a.id_dep=b.id_dep(+);
- Putem să folosim şi operatorul *BETWEEN* într-un join extern. Dacă vrem să aflăm care angajați ies din grila de salarizare, prin dublarea salariilor, executăm următoarea cerere:
- SQL>SELECT c.den_dep,a.nume, a.salariu, b.grad FROM angajati a,

  grila_salar b, departamente c WHERE a.salariu*2 BETWEEN

  b.nivel_inf(+) AND b.nivel_sup(+) AND a.id_dep=c.id_dep ;

Join vertical

- Join-ul vertical este folosit pentru concatenarea rezultatelor mai multor comenzi *SELECT* şi foloseşte operatorii ***UNION*** (reuniune), ***INTERSECT*** (intersecție) si ***MINUS*** (diferența);
- În acest caz, join-ul se face după coloane de acelaşi tip, nu după rânduri, de aceea se mai numeşte şi vertical;
- Reuniunea se poate face pe coloane declarate de acelaşi tip (number,varchar,date), chiar dacă au semnificații diferite;
- Folosind operatorul *UNION ALL,* se selectează şi înregistrările duplicate;
- Operatorul *INTERSECT* este folosit pentru a selecta înregistrările comune;
- Operatorul *MINUS* este folosit pentru a selecta înregistrările necomune.

Exemple:

- SQL>SELECT id_dep,nume,functie,'are salariu' salar_comision,
  salariu sal_com FROM angajati WHERE id_dep=10
  UNION
  SELECT id_dep,nume,functie,'are comision ', comision
  FROM angajati WHERE id_dep=30;
- SQL>SELECT functie FROM angajati WHERE id_dep=10
  UNION ALL
  SELECT functie FROM angajati WHERE id_dep=20;
- SQL>SELECT functie, comision FROM angajati where id_dep=10
  INTERSECT
  SELECT functie,comision FROM angajati where id_dep=20;
- SQL> SELECT functie FROM angajati WHERE id_dep = 10
  MINUS
  SELECT functie FROM angajati WHERE id_dep = 30;