Baze de date-Anul 1
Laborator 8 SQL

Cereri ierarhice. Clauza WITH.

I. [Subcereri ierarhice]

- Clauzele ***START WITH*** şi ***CONNECT BY*** se utilizează în formularea cererilor ierarhice.
  - *START WITH* specifică o condiție care identifică liniile ce urmează să fie considerate ca rădăcini ale cererii ierarhice respective. Dacă se omite această clauză, sistemul *Oracle* utilizează toate liniile din tabel drept linii rădăcină.
  - *CONNECT BY* specifică o condiție care identifică relația dintre liniile „părinte“ și „copil“ ale ierarhiei. Condiția trebuie să conțină operatorul *PRIOR* pentru a face referință la linia „părinte“.
  - Operatorul *PRIOR* face referință la linia „părinte“. Plasarea acestui operator determină direcția interogării, dinspre „părinte“ spre „copil“ (*top-down*) sau invers (*bottom-up*). Traversarea *top-down*, respectiv *bottom-up* a arborelui se realizează prin specificări de forma următoare:

    Top-down: ***CONNECT BY PRIOR*** *cheie_parinte = cheie_copil*;
    Bottom-up: ***CONNECT BY PRIOR*** *cheie_copil = cheie_parinte*;

Obs: Operatorul *PRIOR* poate fi plasat în fața oricărui membru al condiției specificate în clauza *CONNECT BY*.
Obs: Liniile „părinte“ ale interogării sunt identificate prin clauza *START WITH*. Pentru a găsi liniile „copil“, *server*-ul evaluează expresia din dreptul operatorului *PRIOR* pentru linia „părinte“, şi cealaltă expresie pentru fiecare linie a tabelului. Înregistrările pentru care condiția este adevărată vor fi liniile „copil“. Spre deosebire de *START WITH*, în clauza *CONNECT BY* nu pot fi utilizate subcereri.

- Pseudocoloana ***LEVEL*** poate fi utilă într-o cerere ierarhică. Aceasta determină lungimea drumului de la rădăcină la un nod.

II. [Clauza WITH]

- Cu ajutorul clauzei *WITH* se poate defini un bloc de cerere înainte ca acesta să fie utilizat într-o interogare.
- Clauza permite reutilizarea aceluiași bloc de cerere într-o instrucțiune *SELECT* complexă. Acest lucru este util atunci când o cerere face referință de mai multe ori la același bloc de cerere, care conține operații *join çi* funcții agregat.

1. Să se obțină numele salariaților care lucrează într-un departament în care există cel puțin un angajat cu salariul egal cu salariul maxim din departamentul 30.
2. Utilizând clauza *WITH*, să se scrie o cerere care afişează numele departamentelor şi valoarea totală a salariilor din cadrul acestora. Se vor considera departamentele a căror valoare totală a salariilor este mai mare decât media valorilor totale ale salariilor pe departamente.

III. Operatorul *EXISTS*

- În instrucțiunile *SELECT* imbricate, este permisă utilizarea oricărui operator logic.
- Pentru a testa dacă valoarea recuperată de cererea externă există în mulțimea valorilor regăsite de cererea internă corelată, se poate utiliza operatorul *EXISTS*. Dacă subcererea returnează cel puțin o linie, operatorul returnează valoarea *TRUE*. În caz contrar, va fi returnată valoarea *FALSE*.
- Operatorul *EXISTS* asigură că nu mai este continuată căutarea în cererea internă după ce aceasta regăseşte o linie.

Exerciţii:

1. Să se afișeze codul, numele, data angajării, salariul şi managerul pentru:
   a) ierarhia arborescenta de sub De Haan.
   b) subalternii directi ai lui De Haan;
   c) subalternii care sunt cu 2 niveluri sub De Haan
2. Pentru fiecare linie din tabelul EMPLOYEES se va afișa o structura arborescentă în care va apărea angajatul, managerul său, managerul managerului etc. Coloanele afişate vor fi: codul angajatului, codul managerului, nivelul în ierarhie (LEVEL) și numele angajatului.
3. Pentru fiecare angajat să se determine numărul de subalterni (nu doar cei direcți).

Baze de date-Anul 2
Laborator 9 SQL

Operatorii ROLLUP şi CUBE. Clauza GROUPING SETS. Funcţia GROUPING. Operatorul DIVISION. Variabile de substituție.

I. [Operatorii ROLLUP şi CUBE. Clauza GROUPING SETS. Funcţia GROUPING.]

- **[Operatorul ROLLUP]** Acest operator furnizează valori **agregat** şi **superagregat** corespunzătoare expresiilor din clauza *GROUP BY*. Operatorul *ROLLUP* poate fi folosit pentru extragerea de statistici şi informaţii totalizatoare din mulţimile rezultate. Acest operator poate fi util la generarea de rapoarte, diagrame şi grafice.

Operatorul *ROLLUP* creează grupări prin deplasarea într-o singură direcţie, de la dreapta la stânga, de-a lungul listei de coloane specificate în clauza *GROUP BY*. Apoi, se aplică funcţia agregat acestor grupări. Dacă sunt specificate ***n* expresii** în operatorul *ROLLUP*, numărul de grupări generate va fi ***n* + 1**. Liniile care se bazează pe valoarea primelor *n* expresii se numesc linii obişnuite, iar celelalte se numesc linii superagregat.

Dacă în clauza *GROUP BY* sunt specificate *n* coloane, pentru a produce subtotaluri fără operatorul *ROLLUP* ar fi necesare *n* + 1 instrucţiuni *SELECT* conectate prin *UNION ALL*. Aceasta ar face execuţia cererii ineficientă pentru că fiecare instrucţiune *SELECT* determină accesarea tabelului. Operatorul *ROLLUP* determină rezultatele efectuând un singur acces la tabel şi este util atunci când sunt implicate multe coloane în producerea subtotalurilor.

Ilustrăm aplicarea acestui operator prin urmatorul exemplu.

Exemplu:

Pentru departamentele având codul mai mic decât 50, să se afişeze:

- pentru fiecare departament şi pentru fiecare an al angajării (corespunzător departamentului respectiv), valoarea totală a salariilor angajaţilor în acel an;
- valoarea totală a salariilor pe departamente (indiferent de anul angajării);
- valoarea totală a salariilor (indiferent de anul angajării şi de departament).

```
SELECT department_id, TO_CHAR(hire_date, 'yyyy'), SUM(salary)
FROM   employees
WHERE  department_id < 50
GROUP BY ROLLUP(department_id, TO_CHAR(hire_date, 'yyyy'));
```

Instrucţiunea precedentă va avea un rezultat de forma:

DEPARTMENT_ID	TO_CHAR(hire_date,'yyyy')	SUM(SALARY)
10	1987	4400
10		4400
20	1996	13000
20	1997	6000
20		19000
30	1994	11000
30	1995	3100
30	1997	5700
30	1998	2600
30	1999	2500
30		24900
40	1994	6500
40		6500
		54800

În rezultatul prezentat anterior se pot distinge 3 tipuri de linii.

- Prima linie reprezintă suma salariilor angajaţilor în 1987 din departamentul care are codul 10. În mod similar se interpretează liniile din rezultat care au toate coloanele completate.
- Linia a doua conţine valoarea totală a salariilor din departamentul al cărui cod este 10. La fel se interpretează toate liniile care se disting prin faptul că valoarea coloanei *TO_CHAR(hire_date, 'dd')* este *null*.
- Ultima linie conţine suma salariilor tuturor angajatiilor din departamentele al căror cod este mai mic decat 50. Întrucât această linie corespunde totalului general, ea conţine valoarea *null* pe toate coloanele, cu excepţia câmpului *SUM*(*salary*)*.*

- **[Operatorul CUBE]**

Operatorul *CUBE* grupează liniile selectate pe baza valorilor tuturor combinaţiilor posibile ale expresiilor specificate şi returnează câte o linie totalizatoare pentru fiecare grup. Acest operator este folosit pentru a produce mulţimi de rezultate care sunt utilizate în rapoarte. În vreme ce *ROLLUP* produce subtotalurile doar pentru o parte dintre combinaţiile posibile, operatorul *CUBE* produce subtotaluri pentru **toate combinaţiile posibile** de grupări specificate în clauza *GROUP BY*, precum şi un total general.

Dacă există ***n* coloane** sau expresii în clauza *GROUP BY*, vor exista **2^n combinaţii** posibile superagregat. Din punct de vedere matematic, aceste combinaţii formează un cub *n*-dimensional, de aici provenind numele operatorului. Pentru producerea de subtotaluri fără ajutorul operatorului *CUBE* ar fi necesare 2^n instrucţiuni *SELECT* conectate prin *UNION ALL*.

Exemplu:

Pentru departamentele având codul mai mic decât 50 să se afişeze:

- valoarea totală a salariilor corespunzătoare fiecărui an de angajare, din cadrul fiecărui departament;
- valoarea totală a salariilor din fiecare departament (indiferent de anul angajarii);
- valoarea totală a salariilor corespunzatoare fiecărui an de angajare (indiferent de departament);
- valoarea totală a salariilor (indiferent de departament si de anul angajarii).

```
SELECT department_id, TO_CHAR(hire_date, 'yyyy'), SUM(salary)
FROM   employees
WHERE  department_id < 50
GROUP BY CUBE(department_id, TO_CHAR(hire_date, 'yyyy'));
```

În plus faţă de rezultatul corespunzător operaţiei *ROLLUP*, operatorul *CUBE* va produce linii care reprezintă suma salariilor pentru fiecare an de angajare corespunzător unui departament având codul mai mic decât 50. Aceste linii se disting prin faptul că valoarea coloanei *department_id* este *null*.

➢ Pentru determinarea modului în care a fost obţinută o valoare totalizatoare cu ROLLUP sau CUBE, se utilizează funcţia:

- ***GROUPING***(expresie)
  Aceasta întoarce:
  - valoarea 0, dacă expresia a fost utilizată pentru calculul valorii agregat
  - valoarea 1, dacă expresia nu a fost utilizată.

- Dacă se doreşte obţinerea numai a anumitor grupări superagregat, acestea pot fi precizate prin intermediul clauzei :
  - ***GROUPING SETS*** ((expr_11, expr_12, …, expr_1n), (expr_21, expr_22, …expr_2m), …)

Exerciţii:

1. a) Să se afişeze numele departamentelor, titlurile job-urilor şi valoarea medie a salariilor, pentru:
- fiecare departament şi, în cadrul său pentru fiecare job;
- fiecare departament (indiferent de job);
- întreg tabelul.

b) Analog cu a), afişând şi o coloană care arată intervenţia coloanelor *department_name*, *job_title*, în obţinerea rezultatului.

2. a) Să se afişeze numele departamentelor, titlurile job-urilor şi valoarea medie a salariilor, pentru:
- fiecare departament şi, în cadrul său pentru fiecare job;
- fiecare departament (indiferent de job);
- fiecare job (indiferent de departament)
- întreg tabelul.

b) Cum intervin coloanele în obţinerea rezultatului? Să se afişeze 'Dep', dacă departamentul a intervenit în agregare, 'Job', dacă job-ul a intervenit în agregare, 'DepJob' dacă ambele au intervenit și 'Niciuna' daca nicio coloană nu a intervenit.

3. Să se afişeze numele departamentelor, numele job-urilor, codurile managerilor, maximul şi suma salariilor pentru:
- fiecare departament şi, în cadrul său, fiecare job;
- fiecare job şi, în cadrul său, pentru fiecare manager;
- întreg tabelul.