INTRODUCERE IN SQL

Comenzile SQL sunt blocuri de interogare de baza. In particular, discutam declaratiile SQL folosite la:

- executia calculelor
- specificarea alternativa a capetelor de coloana
- concatenarea coloanelor
- sortarea rindurilor
- introducerea criteriilor de cautare.

Privire de ansamblu asupra SQL

Un sistem de management al bazei de date necesita un limbaj de interogare pentru a permite utilizatorului sa acceseze datele. SQL (limbaj de interogare structurata) este limbajul utilizat de majoritatea sistemelor de baze de date relationale.

Limbajul SQL a fost dezvoltat intr-un prototip de sistem de management a bazelor de date relationale - System R - de IBM la mijlocul anilor 1970. In 1979, Corporatia Oracle introduce prima implementare a SQL in varianta comerciala.

Trasaturi caracteristice SQL

- SQL este in limba engleza.
- SQL este un limbaj neprocedural: specifica ce informatii doresti, nu cum sa le obtii. Cu alte cuvinte SQL nu iti cere sa specifici metoda de acces la date. Toate cererile SQL folosesc **optimizarea cererilor** - o parte a RDBMS - pentru a determina rapid gasirea datelor specificate.
- La un moment dat, SQL proceseaza o singura inregistrare. Cea mai comuna forma a unui set de inregistrari este un tabel.
- SQL poate fi folosit de un sir de utilizatori incluzand DBA, programatori de aplicatii, personal de management si multe alte tipuri de utilizatori.
- SQL pune la dispozitie comenzi pentru o varietate de task-uri incluzand:
    - date interogate
    - inserarea, extragerea si stergerea rindurilor intr-un tabel.
    - crearea, modificarea si stergerea obiectelor de tip baza de date
    - controlul accesului la baza de date si la obiectele de tip baza de date.
    - garantarea consistentei bazei de date

La inceput sistemele de management a bazei de date au utilizat un limbaj separat pentru fiecare categorie de task-uri. SQL le-a unificat pe toate acestea intr-un singur limbaj.

SQL a devenit un limbaj standard industrial pentru bazele de date relationale . Institutul National American de Standarde (ANSI) a adoptat SQL ca limbaj standard pentru RDBMS in anul 1986. Organizatia Internationala de Standarde (ISO) a adoptat deasemenea SQL ca limbaj standard pentru RDBMS. Toate RDBMS-urile suporta unele

forme de SQL si toti vinzatorii de RDBMS se aliniaza la standardele ANSI.

Setul de comenzi SQL

Comanda si Descriere:

SELECT	este comanda cea mai utilizat ; este folosita pentru obtinerea informatiilor din bazele de date
INSERT UPDATE DELETE	aceste trei comenzi sunt utilizate pentru a introduce noi rinduri, pentru a actualiza rindurile existente si stergerea rindurilor nedorite din tabelele bazelor de date respective. (Ele sunt cunoscute in ansamblu ca DML sau comenzi ale limbajului de manevra a datelor)
CREATE ALTER DROP	aceste trei comenzi sunt utilizate dinamic pentru a crea, utiliza si sterge orice structura de date, de exemplu, tabele, expuneri, indecsi. (Ele sunt cunoscute sub numele colectiv DDL sau comenzi ale limbajelor de definire a datelor).
GRANT REVOKE	aceste doua comenzi sunt utilizate pentru a acorda sau a revoca drepturile de acces pentru bazele de date si structurile din **Oracle**.

Scrierea comenzilor SQL

Cand scriem comenzi SQL,este important sa ne reamintim cateva reguli simple pentru construirea unor declaratii valide care sunt si usor de citit si de editat:

- Comenzile SQL pot fi pe una sau mai multe linii.
- Clauzele sunt uzual plasate pe linii separate.
- Tabularea poate fi folosita.
- Cuvintele de comanda nu pot fi separate pe mai multe linii.
- Comenzile SQL nu sunt 'case sensitive'.
- O comanda SQL este introdusa la promptul SQL si liniile subsecventelor sunt numerotate de mediul de lucru.
- O singura declaratie poate fi considerata curenta cat timp ea este in buffer si poate fi executata plasand un punct si virgula(;) la sfarsitul ultimei clauze.

Fiecare din urmatoarele declaratii sunt valide:

```
select * from employees; -- utilizand * se listeaza toti angajatii

SELECT
*
FROM
EMPLOYEES
;

SELECT *
FROM EMPLOYEES;
```

Blocul de interogare de baza

In cea mai simpla forma trebuie sa contina:

1. O clauza SELECT, care listeaza coloanele pentru afisare astfel incat este esentiala o Proiectie.
2. O clauza FROM care specifica tabela implicata.

Pentru a lista toate numerele departamentelor, numele angajatilor si
codul managerilor in tabela EMP introduceti urmatoarele:

```
select department_id, first_name, last_name, manager_id
     from employees;
```

DEPARTMENT_ID	FIRST_NAME	LAST_NAME	MANAGER_ID
90	Steven	King	-
90	Neena	Kochhar	100
90	Lex	De Haan	100
10	Jennifer	Whalen	101
110	Shelley	Higgins	101
110	William	Gietz	205
80	Eleni	Zlotkey	100
80	Ellen	Abel	149
80	Jonathon	Taylor	149
-	Kimberely	Grant	149

De remarcat ca numele coloanelor sunt separate prin virgula.

```
     Este posibil sa selectam toate coloanele din tabela prin
specificarea
     unui asterisc ('*') dupa cuvantul SELECT.

          select * from employees;
```

Alte elemente in clauza SELECT

Este posibil sa se includa in clauza SELECT:

- Expresii aritmetice
- Alias-uri de coloane
- Coloane concatenate
- Literali

Toate aceste optiuni ajuta utilizatorul sa ceara date si sa le manevreze in functie de scopurile interogarii; de exemplu, executia calculelor, legarea coloanelor sau afisarea sirurilor de litere din text.

Expresii aritmetice

O expresie este o combinatie de una sau mai multe valori, operatori si functii care sa evalueaza la o valoare.

Expresiile aritmetice pot contine nume de coloane, valori numerice constante si operatori aritmetici:

```
     Operatori             Descriere
     ---------             ---------
        +                  adunare
        -                  scadere
        *                  inmultire
        /                  impartire
```

```
Pentru teste simple se poate folosi foarte usor tabela DUAL:
select 2*3 from dual;

       select first_name, last_name, salary *12
from employees;
```

FIRST_NAME	LAST_NAME	SALARY*12
Steven	King	288000
Neena	Kochhar	204000
Lex	De Haan	204000
Jennifer	Whalen	52800
Shelley	Higgins	144000
William	Gietz	99600
Eleni	Zlotkey	126000

Ellen	Abel	132000
Jonathon	Taylor	103200
Kimberely	Grant	84000

Daca expresia aritmetica contine mai mult decat un operator, prioritatile si ordinea sunt cele aritmetice cunoscute (de la stanga la dreapta pentru operatorii de aceeasi prioritate).

In urmatorul exemplu, inmultirea (250*12) este evaluata prima; apoi valoarea salariului este adunata la rezultatul multiplicarii(3000). Astfel pentru randul lui Stebem King avem 24000 + 3000=27 000.

```
select first_name, last_name, salary, salary + 250 *12
     from employees;
```

FIRST_NAME	LAST_NAME	SALARY	SALARY+250*12
Steven	King	24000	27000
Neena	Kochhar	17000	20000
Lex	De Haan	17000	20000
Jennifer	Whalen	4400	7400
Shelley	Higgins	12000	15000
William	Gietz	8300	11300
Eleni	Zlotkey	10500	13500
Ellen	Abel	11000	14000
Jonathon	Taylor	8600	11600
Kimberely	Grant	7000	10000

Parantezele pot fi utilizate pentru specificarea ordinii de executie a operatorilor, daca, de exemplu, adunarea e dorita inainte de inmultire:

Aliasuri de coloana

Cand se afiseaza rezultatul unei interogari, se utilizeaza numele coloanelor selectate. In multe cazuri acest nume poate fi criptic sau fara inteles.

Puteti schimba un titlu de coloana utilizand un 'ALIAS'.

Un alias de coloana da unei coloane un nume de titlu alternativ la iesire.

Specificati aliasul dupa coloana in lista selectata. Implicit, titlurile alias vor fi fortate la litere mari si nu pot contine blankuri, decat daca aliasul este inclus intre ghilimele(" ").

Pentru a afisa titlul de coloana ANSALAR pentru salariul anual insemnand SALARY*12, utilizati un alias de coloana:

```
select first_name, last_name, salary *12 ANSALAR
       from employees;
```

FIRST_NAME	LAST_NAME	ANSALAR
Steven	King	288000
Neena	Kochhar	204000
Lex	De Haan	204000
Jennifer	Whalen	52800

SAU

```
select first_name, last_name, salary *12 "SALAR ANUAL"
       from employees
```

FIRST_NAME	LAST_NAME	SALAR ANUAL
Steven	King	288000
Neena	Kochhar	204000
Lex	De Haan	204000
Jennifer	Whalen	52800

SAU

```
select first_name || '-' || last_name as nume from employees
```

NUME
Ellen-Abel
Curtis-Davies
Lex-De Haan
Bruce-Ernst

Nota:

Intr-o declaratie SQL, un alias de coloana poate fi utilizat numai in clauza SELECT.

Operatorul de concatenare

Operatorul de concatenare (||) permite coloanelor sa fie legate cu alte coloane, valorilor aritmetice sau sirurilor de caractere sa creeze un sir de caractere.

Pentru a combina first_name si last_name si sa se dea aliasul EMPLOYEE expresiei, introduceti:

select first_name || last_name from employees ;

FIRST_NAME||LAST_NAME
EllenAbel
CurtisDavies
LexDe Haan
BruceErnst

Literali

Un literal este orice caracter, expresie, numar inclus in clauza SELECT care nu este un nume de coloana sau un alias de coloana.

Un literal este reprezentat pe fiecare rand returnat la iesire.

Literalii de tip data calendaristica si caracter trebuie inchisi intre ghilimele simple ('); literalii de tip numar nu au nevoie de ghilimele simple (').

Urmatoarea declaratie contine 2 literali:

```
  select
       first_name || '-' || last_name as nume,
       'lucreaza in departamentul',
       department_id
from employees;
```

NUME	'LUCREAZAINDEPARTAMENTUL'	DEPARTMENT_ID
Steven-King	lucreaza in departamentul	90
Neena-Kochhar	lucreaza in departamentul	90
Lex-De Haan	lucreaza in departamentul	90

Tratarea valorilor nule

Daca unui rand ii lipseste o valoare pentru o anumita coloana, despre acea valoare se spune ca este nula.

O valoare nula este o valoare care este sau incorecta, sau necunoscuta, sau inaplicabila. O valoare nula nu este la fel ca 'zero'. Zero este un numar. Valoarea nula ocupa un octet in reprezentarea interna.

Valoarea nula este tratata corect de catre SQL.

Daca o valoare dintr-o expresie este nula atunci rezultatul este nul. In urmatoarea declaratie numai vanzatorii au un rezultat al salariului:

```
select first_name, COMMISSION_PCT
     from employees;
```

FIRST_NAME	COMMISSION_PCT
Steven	-
Neena	-
Lex	-
Jennifer	-
Shelley	-
William	-
Eleni	,2
Ellen	,3

Daca dorim sa obtinem un rezultat pentru toti angajatii, este necesar sa convertim valoarea nula la un numar. Functia **NVL** converteste o valoare nula la o valoare nenula.

Folositi functia NVL pentru a converti valoarea nula de la declaratia precedenta la 0.

```
          select first_name, salary + NVL(COMMISSION_PCT, 0) "Salariu
plus comision"  from employees;
```

FIRST_NAME	Salariu plus comision
Steven	24000
Neena	17000
Lex	17000
Jennifer	4400
Shelley	12000
William	8300
Eleni	10500,2
Ellen	11000,3

NVL asteapta doua argumente:

1. o expresie
2. o valoare nenula

De notat ca puteti folosi functia NVL pentru a converti un numar nul, data calendaristica sau sir de caractere la un alt numar, data calendaristica sau sir de caractere de acceasi lungime si de acelasi tip de date asteptate.

```
NVL(DATECOLUMN,'01-JAN-88')

NVL(NUMBERCOLUMN,9)

NVL(CHARCOLUMN,'MONDAY')
```

Prevenirea selectiei rindurilor duplicate

Pentru a lista toate numerele de departament din tabela EMP, introduceti:

```
select department_id from employees ;
```

DEPARTMENT_ID
90
90
90
10
110
110
80

Clauza DISTINCT

Pentru eliminarea valorilor duplicate din rezultat, includeti restrictia DISTINCT in comanda SELECT.

Pentru a elimina valorile duplicate afisate in exemplul urmator introduceti:

```
select distinct department_id from employees;
```

DEPARTMENT_ID
-
90
20

110
80
50
10

Coloane multiple pot fi specificate dupa restrictia DISTINCT si restrictia DISTINCT afecteaza toate coloanele selectate.

Pentru a afisa valorile distincte ale lui DEPTNO si JOB,introduceti:

```
select distinct department_id, job_id from employees;
```

DEPARTMENT_ID	JOB_ID
110	AC_ACCOUNT
90	AD_VP
80	SA_REP
50	ST_CLERK
110	AC_MGR
80	SA_MAN
50	ST_MAN
90	AD_PRES
60	IT_PROG
20	MK_MAN

Aceasta afiseaza o lista a **tuturor combinatiilor diferite de ocupatie si numere de departamente.**

De notat ca restrictia DISTINCT poate sa fie referita numai o singura data si trebuie sa urmeze imediat dupa cuvantul de comanda SELECT.

Clauza ORDER BY

In mod normal ordinea rindurilor intoarse in rezultatul unei cereri este nedefinita. Clauza ORDER BY poate fi utilizata pentru a sorta rindurile.

Daca o **folosim, clauza** ORDER BY trebuie sa fie intotdeauna ultima in declaratia SELECT.

Pentru a sorta dupa ENAME, introduceti:

```
select first_name, last_name
     from employees
     order by last_name;
```

FIRST_NAME	LAST_NAME
Ellen	Abel
Curtis	Davies
Lex	De Haan
Bruce	Ernst
Pat	Fay
William	Gietz

Ordonarea implicita a datelor

Ordinea sortarii implicite este **ascendenta**:

- Valorile numerice cele mai mici primele
- Valorile de tip date calendaristice cele mai mici primele.
- Valorile de tip caracter in ordinea alfabetica.

Inversarea ordinii implicite

Pentru a inversa aceasta ordine cuvintul de comanda DESC este specificat dupa numele coloanei in clauza ORDER BY.

Pentru a inversa ordinea coloanei HIREDATE, deci datele cele mai tirzii sa fie afisate primele,introduceti:

```
select first_name, last_name, hire_date
     from employees
     order by hire_date desc;
```

FIRST_NAME	LAST_NAME	HIRE_DATE
Eleni	Zlotkey	29-01-2000
Kevin	Mourgos	16-11-1999
Kimberely	Grant	24-05-1999
Diana	Lorentz	07-02-1999
Peter	Vargas	09-07-1998
Jonathon	Taylor	24-03-1998
Randall	Matos	15-03-1998

Ordonarea dupa mai multe coloane

Este posibil sa se ordoneze dupa mai multe coloane. Limita este numarul de coloane din tabela. In clauza ORDER BY se specifica coloanele pentru ordonat separate prin virgula. Daca una sau toate coloanele trebuie sa fie inversate specificati DESC dupa fiecare coloana.

Pentru a ordona dupa doua coloane si afisa in ordinea inversa a salariului, introduceti:

```
select distinct department_id, salary
     from employees
     order by department_id, salary desc;
```

DEPARTMENT_ID	SALARY
10	4400
20	13000
20	6000
50	5800
50	3500
50	3100
50	2600
50	2500

ORDER BY si valorile nule

In Oracle, valorile nule sunt afisate ultimele pentru secventele ascendente si sunt raportate primele cind rindurile sunt sortate in ordine descendenta.

Atentie:

Clauza ORDER BY este utilizata intr-o interogare cind se doreste sa se afiseze rindurile intr-o ordine specifica. Fara clauza ORDER BY randurile sunt returnate intr-o ordine convenita de ORACLE. Ordinea de afisare a rindurilor nu influenteaza ordinea interna a rindurilor asa cum sunt stocate in tabela.

Clauza WHERE

Clauza WHERE contine o conditie pe care rindurile trebuie sa o indeplineasca pentru a fi afisate.

Clauza WHERE, daca este folosita, trebuie sa urmeze clauzei FROM:

```
SELECT    coloane
FROM      tabela
WHERE     anumite conditii sunt indeplinite
```

Clauza WHERE poate compara valori in coloana, literali, expresii aritmetice sau functii.

Clauza WHERE asteapta trei elemente:

1. Un nume de coloana
2. Un operator de comparatie
3. Un nume de coloana, constanta sau lista de valori.

Operatorii de comparatie sunt utilizati in clauza WHERE si pot fi impartiti in doua categorii: logici si SQL.

Operatorii logici

Acesti operatori verifica urmatoarele conditii:

```
Operator  Semnificatie
--------  ------------
=         egal cu
>         mai mare decit
>=        mai mare sau egal
<         mai mic decit
<=        mai mic sau egal
```

Sirurile de caractere si datele calendaristice in clauza WHERE

Coloanele din ORACLE pot avea urmatoarele tipuri: caracter, numar sau data calendaristica.

Sirurile de caractere si datele calendaristice din clauza WHERE trebuie sa fie inchise in ghilimele simple ('). Sirurile de caractere trebuie sa se suprapuna cu valoarea coloanei daca nu, trebuie modificate de o functie.

Pentru a afisa toate detaliile despre angajatul cu numele King:

```
select * from employees
     where last_name = 'King';
```

Pentru a gasi toate numele de departamente cu numarul de departament mai mare ca 20, introduceti:

```
select department_name, department_id from departments
     where department_id >20;
```

Compararea valorilor dintr-o coloana cu valorile din alta

Puteti compara valorile dintr-o coloana cu valorile din alta coloana, sau cu o valoare constanta.

De exemplu, presupunem ca dorim sa obtinem acei angajati al caror comision este mai mare decat salariul lor:

```
select first_name , salary, COMMISSION_PCT from employees
     where COMMISSION_PCT >salary;
```

Operatori SQL

Sunt patru operatori SQL care opereaza pe toate tipurile de date:

```
Operatori SQL

Operator          Semnificatie
--------          ------------
BETWEEN..AND...   intre doua valori (inclusiv)
IN (list)         compara cu o lista de valori
LIKE              compara cu un model de tip caracter
IS NULL           este o valoare NULL
```

Operatorul BETWEEN

Realizeaza teste pentru valori intre, si inclusiv, o valoare minima si o valoare maxima.

Presupunind ca dorim sa vedem angajatii al caror salariu este intre 1000 si 2000:

```
select first_name , salary from employees
where salary between 1000 AND 5000;
```

FIRST_NAME	SALARY
Jennifer	4400
Trenna	3500
Curtis	3100

Randall	2600
Peter	2500
Diana	4200

De notat ca valorile specificate sunt inclusive si ca limita minima trebuie specificata prima.

Operatorul IN

Testeaza valorile dintr-o lista specificata.

Presupunem ca dorim sa gasim angajatii care au marca 100 sau 101 sau 102:

```
select first_name , employee_id from employees
     where employee_id IN (100, 101, 102);
```

FIRST_NAME	EMPLOYEE_ID
Lex	102
Steven	100
Neena	101

Daca se utilizeaza sir de caractere sau date calendaristice, acestea trebuie introduse intre ghilimele (' ').

Operatorul LIKE

Uneori nu se cunosc valorile exacte pe care le cautam. Utilizand operatorul LIKE este posibil sa selectam randurile care se potrivesc cu un model specificat. Operatia de potrivire a caracterelor poate fi asemanata cu o cautare 'wildcard'. Doua simboluri se pot utiliza la construirea sirului de cautare.

```
Simbol          Reprezentare
------          ------------
%                       orice secventa de zero sau mai multe
caractere
-               un singur caracter oarecare
```

Pentru a lista toti angajatii al caror nume incepe cu S, introduceti:

```
select first_name from employees
     where first_name like 'S%';
```

FIRST_NAME
Shelley
Steven

Operatorul IS NULL

Operatorul IS NULL face teste specifice pentru valorile NULL. Selectam angajatii fara commission:

```
select first_name, last_name, COMMISSION_PCT from employees
     where COMMISSION_PCT IS NULL;
```

FIRST_NAME	LAST_NAME	COMMISSION_PCT
Steven	King	-
Neena	Kochhar	-
Lex	De Haan	-
Jennifer	Whalen	-
Shelley	Higgins	-
William	Gietz	-
Kevin	Mourgos	-

Negarea expresiilor

```
Operator            Descriere
--------            ---------
NOT BETWEEN         nu se afla intre doua valori date
NOT IN              nu se afla intr-o lista data de valori
NOT LIKE            diferit de sirul
IS NOT NULL         nu este o valoare nula
```

Pentru a gasi angajatii ai caror salariu nu este intr-un interval, introduceti:

```
select first_name, salary
     from employees
     where salary NOT BETWEEN 24000 AND 30000;
```

FIRST_NAME	SALARY
Neena	17000
Lex	17000
Jennifer	4400
Shelley	12000
William	8300
Eleni	10500
Ellen	11000

Pentru a afla acei angajati a caror nume nu incepe cu M, introduceti:

```
select first_name
     from employees
```

```
where first_name NOT LIKE 'M%';
```

FIRST_NAME
Ellen
Curtis
Lex
Bruce
Pat
William
Kimberely
Shelley

Pentru a afla toti angajatii care au un manager (MGR),introduceti:

```
select first_name, manager_id
     from employees
     where manager_id IS NOT NULL;
```

FIRST_NAME	MANAGER_ID
Neena	100
Lex	100
Eleni	100
Kevin	100
Michael	100
Jennifer	101
Shelley	101

Nota:

Pentru ca o valoare NULL sa fie utilizata intr-o comparatie, atunci operatorul de comparatie trebuie sa fie IS NULL sau IS NOT NULL. Daca acesti operatori nu sunt uti lizati, atunci rezultatul este intotdeauna FALSE

De exemplu, COMMENT!=NULL este intotdeauna falsa. Rezultatul este fals deoarece o valoare NULL nu se poate compara cu o valoare diferita de NULL.

De notat ca o astfel de eroare nu este semnalata, rezultatul fiind intotdeauna fals.

Interogarea datelor cu conditii multiple

Operatorii AND sau OR pot fi utilizati pentru a compune expresii logice.

Rezultatul operatorului AND este adevarat numai daca ambele conditii sunt adevarate; rezultatul operatorului OR este adevarat daca cel putin una din conditii este adevarata.

In urmatoarele doua exemple, conditiile sunt aceleasi, dar operatorii difera.

```
Pentru a afla toti angajatii din departamentul 10 si
care castiga intre 1000 si 30000,introduceti:

      select first_name, department_id, salary
            from employees
            where department_id = 10
                  AND
                  salary Between 1000 AND 30000;
```

FIRST_NAME	DEPARTMENT_ID	SALARY
Jennifer	10	4400

Puteti combina AND sau OR in acceasi expresie logica. Cand AND sau OR apar in aceeasi clauza WHERE, toti operatorii AND sunt evaluati mai intai si apoi toti operatorii OR. Vom spune ca operatorii AND au o precedenta mai mare decat OR.

Deoarece AND are o precedenta mai mare decat OR urmatoarea declaratie SQL intoarce toti angajatii cu salarii sub 5000$ din departament 20 **si** pe toti din departamentul 90 indiferent de salariu:

```
      select first_name, department_id, salary
            from employees
            where salary < 7000
                  AND
                  department_id=20 OR department_id=90;
```

FIRST_NAME	DEPARTMENT_ID	SALARY
Steven	90	24000
Neena	90	17000
Lex	90	17000
Pat	20	6000

```
     Daca doriti sa selectati toti angajatii din departamentele 20 sau
90 care au salariu mai mic de 7000 puteti introduce:

      select first_name, department_id, salary
            from employees
            where salary < 7000
                  AND
                  (department_id=20 OR department_id=90);
```

FIRST_NAME	DEPARTMENT_ID	SALARY
Pat	20	6000

Parantezele specifica ordinea in care operatorii vor fi evaluati. In al doilea exemplu operatorul OR este evaluat inaintea operatorului AND.

TIPURI DE DATE CARACTER SI CONDITII

Tipurile de date caracter si conditii

Tipurile de baza ale datelor stocate intr-o tabela Oracle sunt: caracter, valoare numerica sau data calendaristica. De cate ori rezultatele unei conditii implica date de tip caracter, acestea pot varia in functie de tipul coloanei; ORACLE inzestreaza coloanele cu tipul CHAR pentru valori de lungime fixa si cu tipul VARCHAR2 pentru valori de lungime variabila.

Pentru coloanele cu tipul VARCHAR2, Oracle nu umple sirul de comparare si de aceea va face o potrivire exacta. In primul exemplu, doar un singur rand este intors pentru conditia:

```
WHERE NUME = 'SCOTT'
```

cand un alt rand stocat in coloana NUME are mai multe caractere decat sirul de comparat.

Pentru coloanele cu tipul CHAR, oricum, Oracle face umplere cand valorile coloanelor sunt initial stocate, facandu-le pe toate de aceeasi lungime.

Aceeasi conditie va intoarce ambele randuri pentru SCOTT, indiferent de cate spatii de sfarsit au fost adaugate cand valorile au fost stocate in tabela.

Oracle umple cu blancuri sirul de comparat in cel deal doilea caz si de aceea spatiile stocate sunt nesemnificative.

Precedenta operatorilor

Toti operatorii sunt aranjati intr-o ierarhie ceea ce le determina precedenta. Intr-o expresie operatiile sunt executate in ordinea precedentei lor.

Cand operatorii au precedenta egala atunci ei se evalueaza de la stanga la dreapta.

1. Toti operatorii de comparatie au precedenta egala: =, !=, <, >, <=, >=, BETWEEN...AND, IN, LIKE, IS NULL.
2. NOT (pentru a inversa rezultatul unei expresii logice. De ex: WHERE NOT (SAL > 2000))
3. AND

4. OR.

De fiecare data cand sunteti in dubiu, puteti sa utilizati parantezele pentru a clarifica semnificatia dorita si pentru a va asigura ca Oracle face ceea ce doriti.

SELECT-Sumar

Urmatoarele clauze sunt utilizate in comanda SELECT:

```
SELECT          [DISTINCT] [*,coloana alias],...]
FROM            tabela
WHERE           conditie(ii)
ORDER BY        [coloana,expr] [ASC/DESC];
```

SELECT
: selecteaza cel putin o coloana

Alias
: poate fi folosit pentru coloanele din lista selectata

*
: desemneza toate coloanele

DISTINCT
: poate fi utilizat pentru eliminarea duplicatelor

FROM Tabela
: desemneaza tabela din care provin coloanele

WHERE
: restrictioneaza cererea la randurile care indeplinesc o conditie.Poate contine valori de coloane,expresii si literali

AND/OR
: poate fi utilizat intr-o clauza WHERE pentru a construi conditii mai complexe. AND are prioritate peste OR.

()
: pot fi utilizate pentru a forta prioritatea

ORDER BY
: intotdeauna apare la sfarsit .Specifica ordinea de sortare.Una sau mai multe coloane pot fi specificate aici.

ASC
: ordinea ascendenta este ordinea de sortarå (implicita) si nu trebuie specificat.

DESC
: inverseaza ordinea de sortare implicita si trebuie specificat dupa un nume de coloana.

Clauzele pot fi introduse pe linii separate in buffer si tabelarea este utilizata pentru claritate si in editare.

Exercitii-Introducere in SQL

1. Selectati toate informatiile din tabela EMPLOYEES.

2. Listati toti angajatii care au salariul intre 10000 si 40000.

3. Listati numerele de departament si numele in ordinea numelor departamentelor folosind tabela departments.

4. Afisati toate tipurile de job-uri (tabela jobs).

5. Listati detaliile angajatilor din departamentele 10 si 20 in ordinea alfabetica a numelui.

6. Listati numele si salariul tuturor angajatilor din departamentul 20.

7. Afisati toti angajatii ai caror nume contine a sau b in interior.

8. Listati numele complet si job id pentru toti angajatii care au un manager.

9. Afiseaza numele si salariu plus comisionului ca renumeratie (alias) pentru toti angajatii.

10. Afiseaza toti salariatii care au fost angajati in anul 1994.

11. Afisati numele, salariul anual si comisionul pentru toti angajatii ai caror salariu lunar este mai mare decat 10 000. Iesirea va fi ordonata dupa salariu, cele mai mari primele. Daca doi sau mai multi angajati au acelasi salariu trebuie sortati dupa nume.