Restricţionarea şi sortarea datelor obţinute în urma interogării unei baze de date

1. Obiective:

În urma interogării unei baze de date poate apare necesitatea reducerii numărului de linii afişate sau specificării ordinii în care să fie afişate datele. Acest referat prezintă regulile SQL folosite pentru realizarea acestor acţiuni, noţiunile prezentate fiind însoţite de numeroase exemple.

După parcurgerea acestui referat studentul va dispune de cunoştinţele necesare efectuării următoarelor operaţii:

- **limitarea numărului înregistrărilor returnate de o interogare;**
- **sortarea înregistrărilor returnate de o interogare.**

2. Limitarea înregistrărilor folosind o selecţie:

EMP

EMPNO	ENAME	JOB	…	DEPTNO
7839	KING	PRESIDENT		10
7698	BLAKE	MANAGER		30
7782	CLARK	MANAGER		10
7566	JONES	MANAGER		20
…				

"… returnează toţi angajaţii din departamentul 10"

EMP

EMPNO	ENAME	JOB	…	DEPTNO
7839	KING	PRESIDENT		10
7782	CLARK	MANAGER		10
7934	MILLER	CLERK		10

Să presupunem, conform exemplului de mai sus, că se doreşte afişarea tuturor angajaţilor din departamentul 10 (prezintă interes doar setul de linii care au valoarea 10 în coloana DEPTNO). Această metodă de restricţionare reprezintă baza clauzei **`WHERE`** în SQL.

2.1 Limitarea liniilor selectate

Se poate restricţiona numărul de linii returnate de o interogare folosind clauze `WHERE`. O clauză `WHERE` conţine o condiţie ce trebuie îndeplinită de fiecare linie din rezultat şi urmează imediat după clauza `FROM`.

```
SELECT [DISTINCT] {*, coloana [alias], …}
FROM tabel
[WHERE conditie];
```

unde:

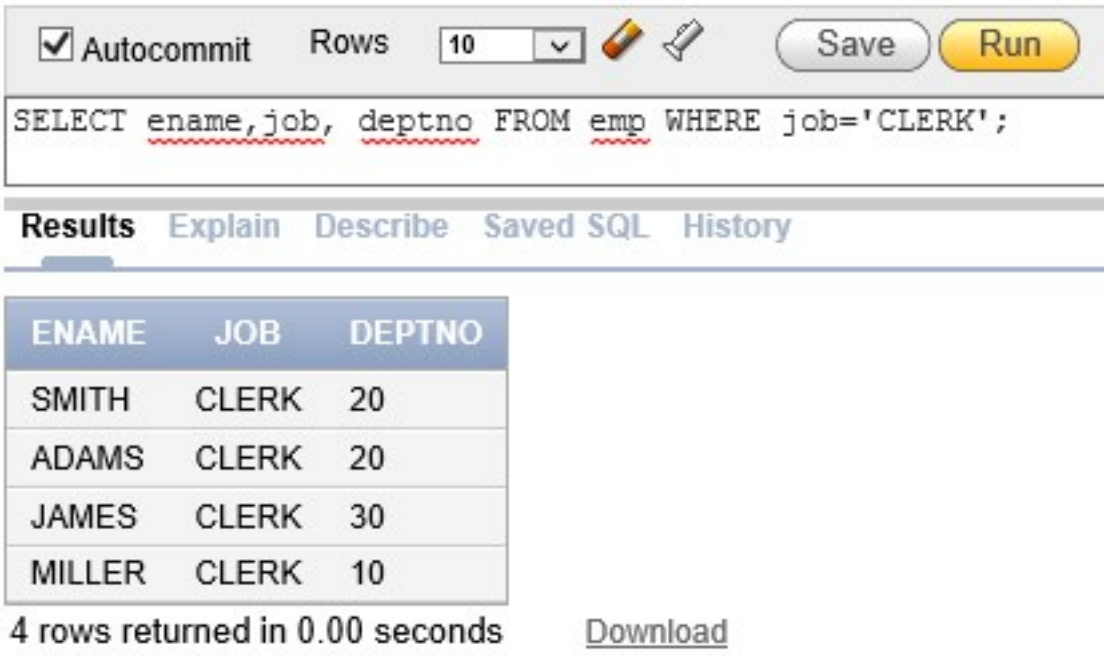
`WHERE`	restricţionează interogarea la liniile ce îndeplinesc condiţia *conditie*.
conditie	reprezintă condiţia ce trebuie satisfăcută de fiecare înregistrare ce apare în rezultat; este compusă din nume de coloane, expresii, constante şi operatori de comparaţie.

Clauza `WHERE` poate compara valorile din coloane, valori literale, expresii aritmetice sau funcţii, fiind compusă din trei elemente:

- numele coloanei
- operatorul de comparaţie
- nume de coloană, constantă sau listă de valori

2.1.1. Folosirea clauzei `WHERE`

```
SELECT  ename, job, deptno FROM  emp WHERE  job='CLERK';
```



Exemplul anterior afişează toţi angajaţii care au funcţia CLERK.

Şiruri de caractere şi date calendaristice

Şirurile de caractere şi datele calendaristice utilizate în clauza `WHERE` trebuie incluse între apostrofuri (‘ ‘). Toate căutările la nivel de caracter sunt *case-sensitive* (i.e. se face distincţie între litere mici şi majuscule). Datele calendaristice sunt memorate de Oracle în formatul secol, an, luna, zi, ore, minute şi secunde. Afişarea implicită a datei este DD-MON-YY.

Notă: valorile numerice nu trebuie incluse între apostrofuri.

Operatori de comparaţie

Operatorii de comparaţie (=, >, >=, <, <=, <>) sunt folosiţi în condiţii care compară două expresii. Utilizarea lor în clauza `WHERE` respectă următorul format:

```
WHERE expresie operator valoare
```

Exemple:

```
… WHERE hiredate='01-JAN-95'
… WHERE sal>=1500
… WHERE ename='SMITH'
```

Alţi operatori de comparaţie ce pot fi utilizaţi într-o clauză `WHERE` sunt prezentaţi în tabelul următor:

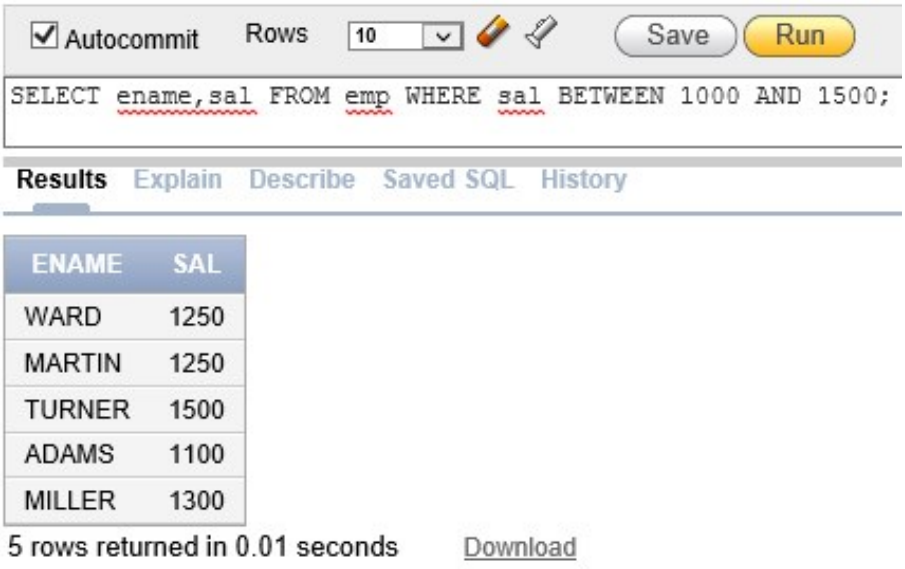
Operator	Semnificaţie
BETWEEN … AND …	Între două valori (inclusiv)
IN (listă)	Potriveşte orice valoare din listă
LIKE	Potriveşte un tip de caracter
IS NULL	Este valoare null

2.1.1.1 Operatorul BETWEEN

Operatorul BETWEEN se utilizează pentru selectarea valorilor dintr-un interval.

```
SELECT ename, sal FROM scott.emp WHERE sal BETWEEN 1000 AND 1500;
```

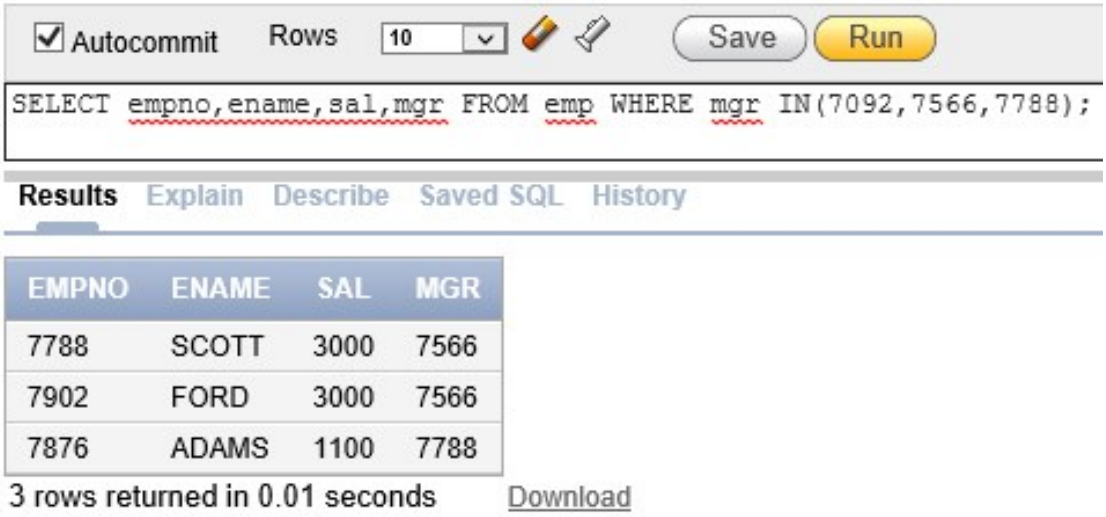
Limita inferioară **Limita superioară**



2.1.1.2 Operatorul IN

Operatorul IN este utilizat pentru căutare într-o listă de valori. El poate fi utilizat cu orice tip de dată.

```
SELECT  empno, ename, sal, mgr FROM emp WHERE  mgr IN (7902, 7566, 7788);
```



Următorul exemplu returnează câte o linie din tabelul `emp` pentru fiecare angajat al cărui nume este inclus în lista de nume din clauza `WHERE`.

```
SELECT  empno, ename, mgr, deptno FROM emp WHERE  ename IN ('FORD', 'ALLEN');
```

Notă: dacă în listă sunt folosite caractere sau date calendaristice, acestea trebuie incluse între apostrofuri (‘ ‘).

2.1.1.3 Operatorul LIKE

```
SELECT  ename FROM  emp WHERE  ename LIKE 'S%';
```

Nu întotdeauna se cunoaşte valoarea exactă pe baza căreia se va efectua căutarea. Instrucţiunea `SELECT` permite selectarea liniilor care corespund unui tipar de caractere cu ajutorul operatorului `LIKE`. Operaţia de potrivire după un tipar de caractere este referită drept căutare cu caractere *wildcard*. Pentru construirea şirurilor de căutare pot fi utilizate două simboluri:

Simbol	Descriere
%	Reprezintă orice secvenţă de caractere
_ (underscore)	Reprezintă un singur caracter

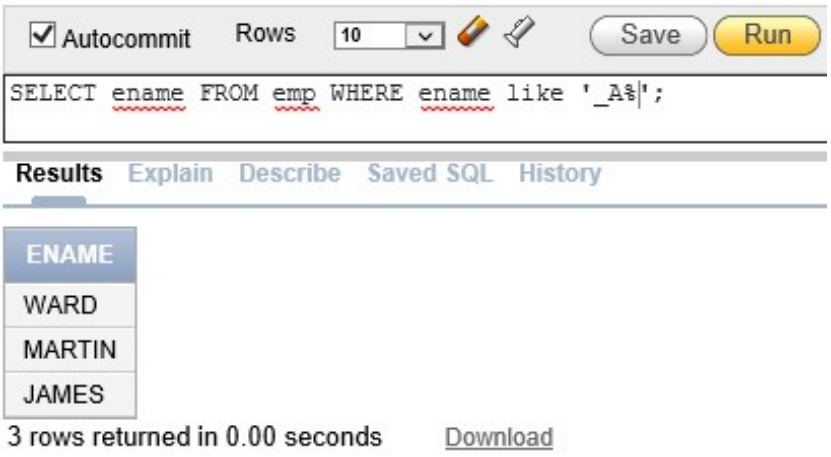
Instrucţiunea `SELECT` de mai sus returnează numele angajaţilor din tabelul `emp` al căror nume începe cu “S”. Numele care încep cu “s” nu vor fi returnate.

În anumite cazuri, operatorul `LIKE` poate fi utilizat în locul operatorului `BETWEEN`. Următorul exemplu afişează numele şi data angajării tuturor angajaţilor a căror angajare s-a făcut între ianuarie 1981 şi decembrie 1981.

```
SELECT  ename, hiredate FROM emp WHERE  hiredate LIKE '%81';
```

- Pot fi combinate diferite tipuri de potriviri pe caracter

```
SELECT  ename FROM emp WHERE ename LIKE '_A%':
```



- Dacă se doreşte căutarea caracterelor '%' sau '_' se va folosi opţiunea `ESCAPE`, care precizează de fapt caracterul care va fi utilizat drept Escape.

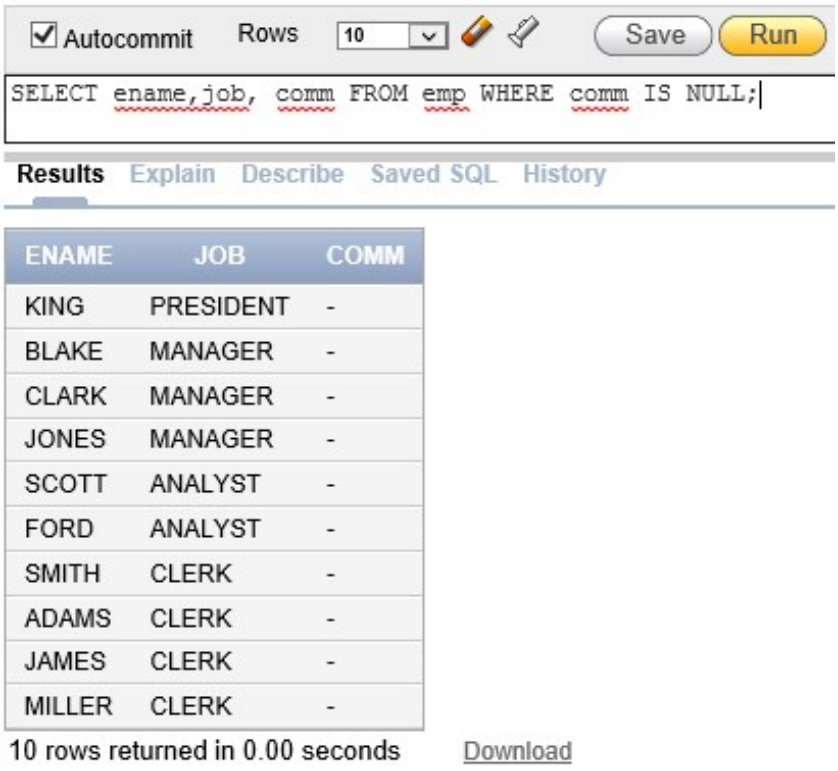
Următoarea instrucţiune SELECT afişează numele tuturor angajaţilor al căror nume conţine secvenţa de caractere "A_S".

```
SELECT  ename FROM  emp WHERE  ename LIKE '%A\_S' ESCAPE '\';
```

2.1.1.4 Operatorul IS NULL

Operatorul **`IS NULL`** este utilizat pentru căutarea valorilor null. Deoarece valoarea null are semnificaţia unei valori indisponibile, neatribuite, necunoscute sau neaplicabile ea nu poate apare în cadrul unei operaţii de comparaţie, deoarece ar conduce la un rezultat null. De exemplu, pentru a afişa numele, funcţia şi comisionul tuturor angajaţilor care nu au dreptul la comision se va folosi următoarea instrucţiune `SELECT`:

```
SELECT  ename, job, comm FROM  emp WHERE  comm IS NULL;
```



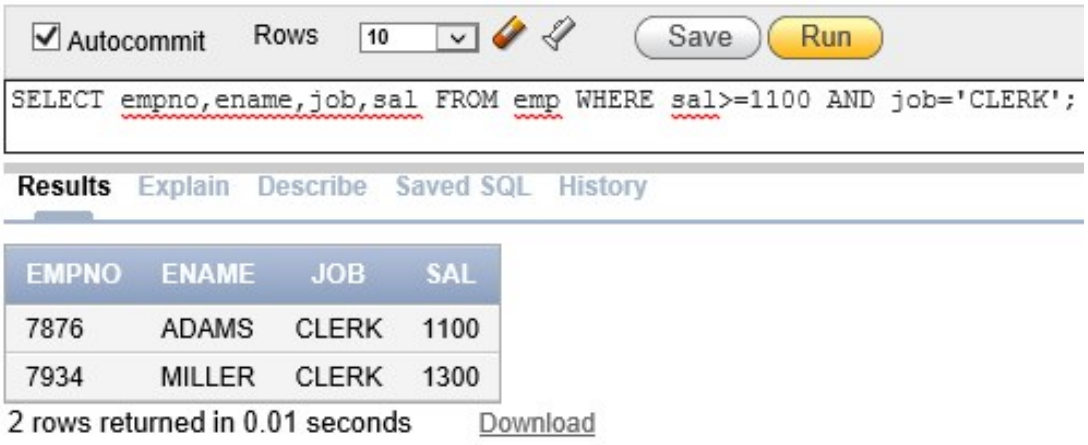
2.1.1.5 Operatori logici (AND, OR, NOT)

Un operator logic combină două componente de tip condiţie pentru a produce un singur rezultat bazat pe acestea sau inversează rezultatul unei singure condiţii. În SQL sunt disponibili trei operatori logici: AND, OR şi NOT.

Operator	Comentariu
AND	Returnează TRUE dacă ambele componente ale condiţiei sunt adevărate
OR	Returnează TRUE dacă una din componentele condiţiei este adevărată
NOT	Returnează TRUE dacă respectiva condiţie este falsă

Folosirea operatorului AND

```
SELECT empno, ename, job, sal FROM emp WHERE sal>=1100 AND job='CLERK';
```

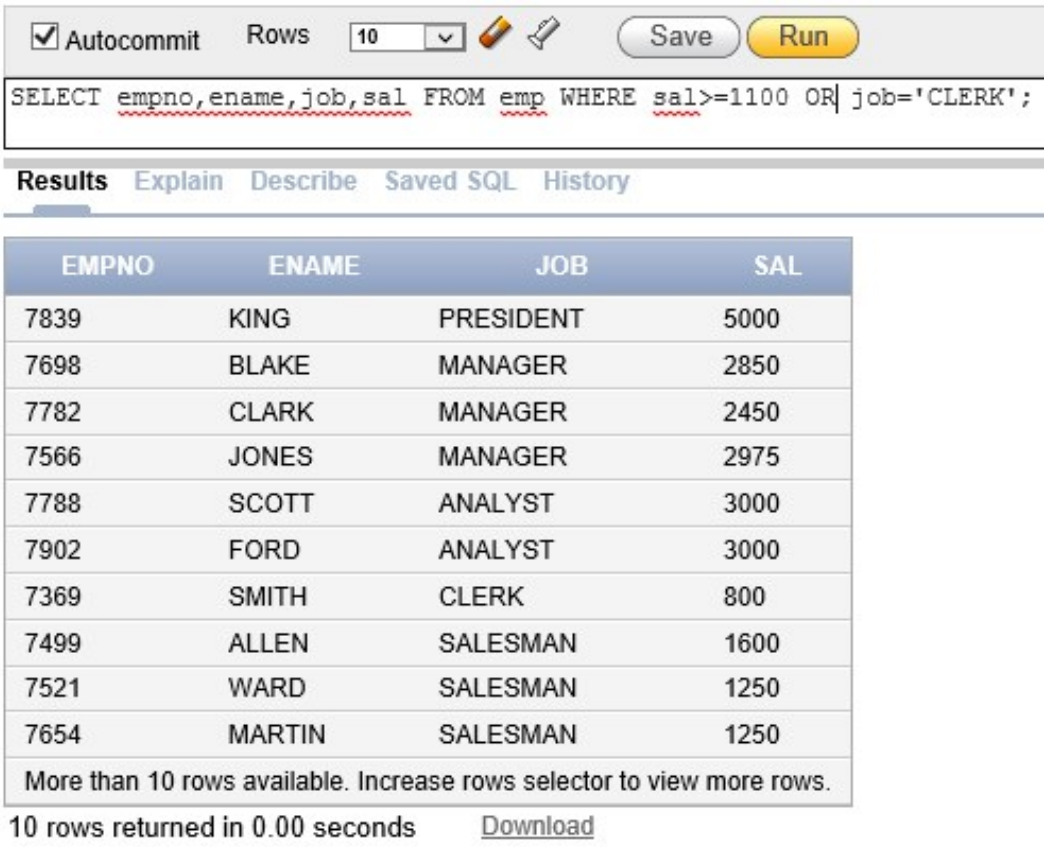


În exemplul de mai sus ambele condiţii trebuie să fie adevărate pentru a fi selectată o înregistrare. De aceea, un angajat care are funcţia CLERK ***şi*** câştigă mai mult de $1100 va fi selectat.

Notă: - toate căutarile de tip caracter sunt *case-sensitive.*
- şirurile de caracter trebuie incluse între apostrofuri (‘ ‘).

Folosirea operatorului OR

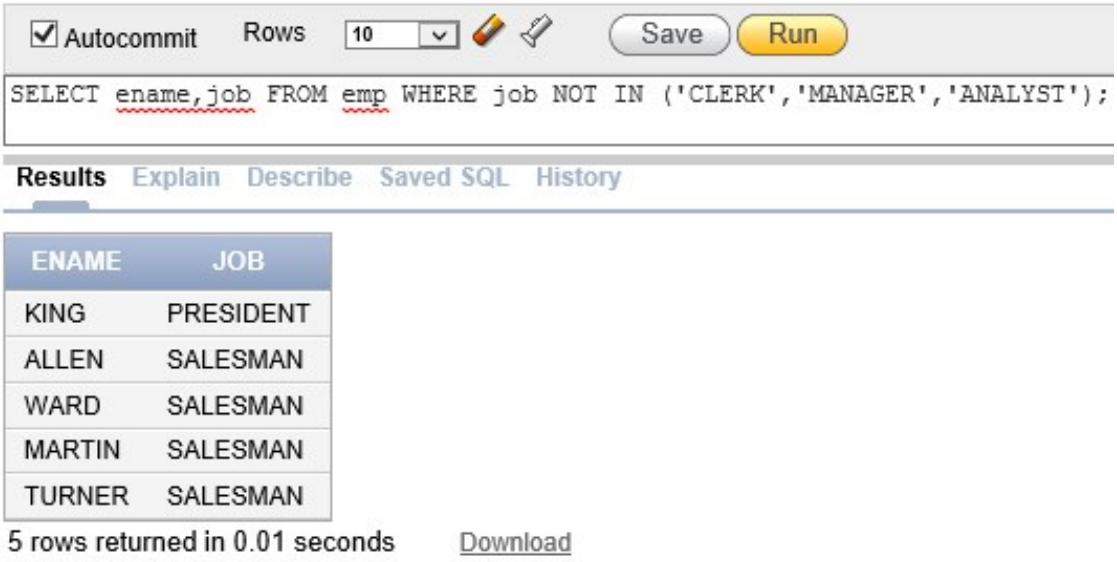
```
SELECT empno, ename, job, sal FROM emp WHERE sal>=1100 OR job='CLERK';
```



În exemplul de mai sus, vor fi selectate înregistrările care îndeplinesc cel puţin o condiţie: fie `sal>=1100`, fie `job='CLERK'`.

Folosirea operatorului NOT

```
SELECT  ename, job FROM emp WHERE job NOT IN ('CLERK', 'MANAGER',
'ANALYST');
```



În exemplul de mai sus este afişat numele şi funcţia tuturor angajaţilor a căror funcţie ***nu este*** CLERK, MANAGER sau ANALYST.

Notă: operatorul NOT poate fi combinat şi cu alţi operatori SQL, cum ar fi `BETWEEN`, `LIKE` şi `IS NULL`.

```
… WHERE job NOT IN ('CLERK', 'ANALYST')
… WHERE sal NOT BETWEEN 1000 AND 1500
… WHERE comm IS NOT NULL
```

Precedenţa operatorilor

Prioritate	Operator
1 – maximă	Toţi operatorii de comparaţie
2	NOT
3	AND
4 – minimă	OR

Notă: aceste reguli pot fi încălcate folosind paranteze.

2.1.2 Clauza ORDER BY

Liniile returnate de o interogare sunt afişate într-o ordine oarecare. Pentru sortarea liniilor se utilizează clauza `ORDER BY`, care, dacă este folosită, trebuie să apară ultima în instrucţiunea SELECT. Sortarea se poate face dupa o coloană, o expresie sau după un alias de coloană.

Sintaxă:

```
SELECT  expresie
FROM  tabel
[WHERE  conditie]
[ORDER BY  {coloana, expresie}  [ASC|DESC];
```

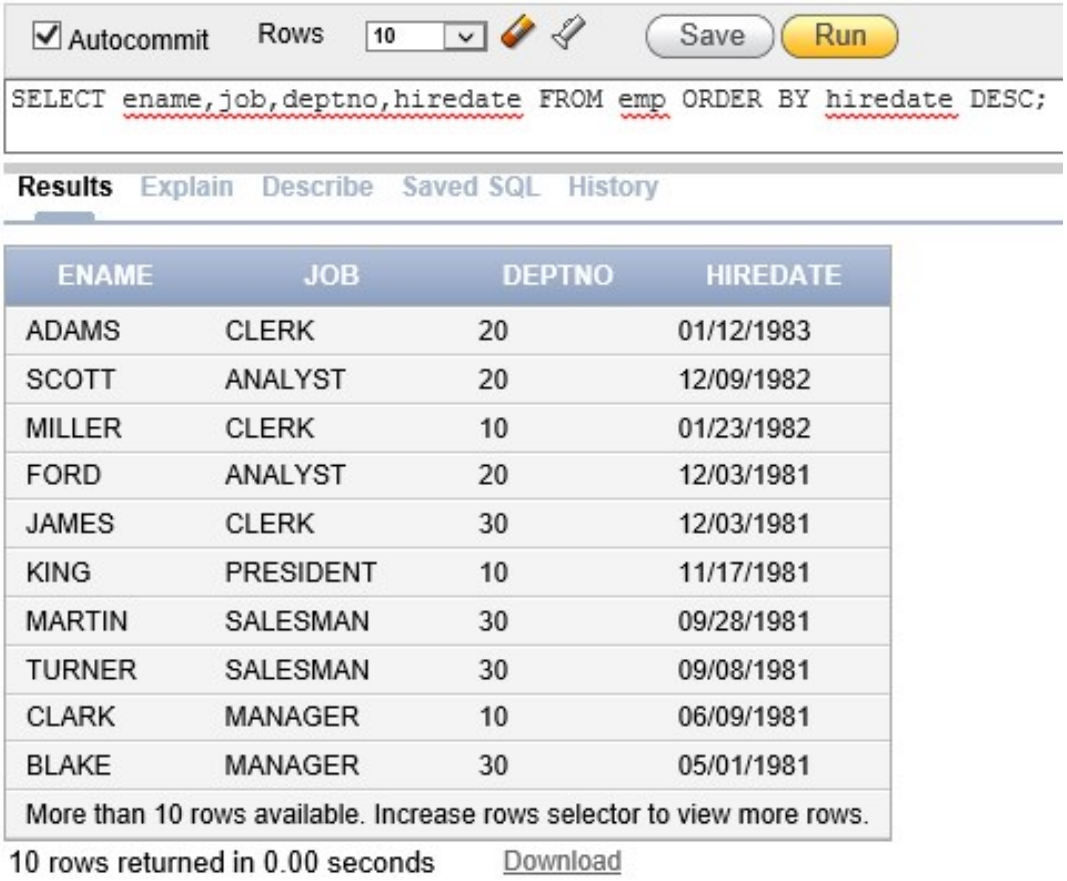
unde:

`ORDER BY`	specifică ordinea în care sunt afişate liniile.
`ASC`	ordonează liniile ascendent – implicit.
`DESC`	ordonează liniile descendent.

Dacă nu este folosită clauza `ORDER BY` ordinea sortării este nedefinită şi Serverul Oracle poate afişa liniile în ordine diferită pentru două interogări identice.

SELECT-ul următor afişează rezultatele interogării ordonate descrescător după coloana *hiredate*.

```
SELECT  ename, job, deptno, hiredate FROM emp ORDER BY hiredate DESC;
```



Ordinea implicită a sortării datelor este cea ascendentă:

- valorile numerice sunt afişate începând cu cea mai mică valoare – de exemplu 1- 999.
- datele sunt afişate începând cu cea mai recentă – de exemplu 01-JAN-92 înaintea lui 01-JAN-95.
- valorile tip caracter sunt afişate în ordine alfabetică – de exemplu A înaintea lui Z.
- valorile null sunt afişate ultimele în cazul sortărilor ascendente si primele pentru cele descendente.

Următoarea instrucţiune `SELECT` ordonează liniile după aliasul coloanei `sal*12` (de tip expresie aritmetică).

```
SELECT  empno, ename, sal*12 annsal FROM emp ORDER BY  annsal;
```

În următorul exemplu liniile sunt sortate după coloanele `deptno` şi `sal`, iar liniile având aceeaşi valoare pentru `deptno` fiind sortate descendent după `sal`.

```
SQL> SELECT  ename, deptno, sal
  2  FROM  scott.emp
  3  ORDER BY  deptno, sal DESC;
```

Notă: se poate face sortare şi după o coloană care nu este în lista clauzei `SELECT`.

Probleme

1. Afişati numele şi salariul angajaţilor din tabelul `emp` care câştigă mai mult de $2850. Salvaţi instrucţiunea SQL în fişierul p1.sql şi apoi rulaţi-l.
2. Modificaţi p1.sql astfel încât să afişaţi numele şi salariul tuturor angajaţilor ale căror salarii nu intră în intervalul $1500 - $2850. Salvaţi instrucţiunea în fişierul p2.sql şi apoi rulaţi din nou interogarea.
3. Afişaţi numele şi numerele de departament ale angajaţilor care lucrează în departamentele 10, respectiv 30, ordonaţi alfabetic după nume.
4. Modificaţi fişierul p2.sql şi listaţi numele şi salariul angajaţilor care câştigă mai mult de $1500 şi lucrează în departamentul 10 sau 30. Redenumiţi coloanele din rezultat Angajat şi Salar Lunar. Salvaţi modificările în fişierul p4.sql şi apoi rulaţi-l.
5. Afişaţi numele şi funcţia pentru angajaţii care nu au manager.
6. Afişaţi numele, salariul şi comisionul pentru toţi angajaţii care au comision. Sortaţi datele în ordine descendentă după salariu şi comision.
7. Afişaţi numele angajatilor care conţin 2 caractere ‘L’ consecutive în numele lor şi îndeplinesc următoarea condiţie: lucrează în departamentul 30 sau au manager cu marca 7782.
8. Modificaţi p4.sql şi afişaţi numele, salariul şi comisionul pentru toţi angajaţii care au comisionul mai mare decât salariul mărit cu 10%. Salvaţi modificările în fisierul p8.sql şi apoi rulaţi-l.
9. Afişaţi numele, funcţia şi salariul angajaţilor ce au funcţia Clerk sau Analyst şi al căror salariu nu este de $1000, $3000 sau $5000.
10. Afişaţi numele, funcţia şi data angajării persoanelor angajate între 10 februarie 1981 şi 1 mai 1981. Ordonaţi înregistrările returnate de interogare crescător după data angajării.