

*Universitatea Constantin Brâncuși din Târgu Jiu*

*Facultatea: Inginerie*

*Program de conversie profesională a cadrelor didactice din învățământul preuniversitar: Informatică,  
Tehnologia Informației și a Comunicațiilor*

# Baze de date

## Limbajul SQL

**Teams: TIC - Baze de date-2022/2023**

**Adrian Runceanu**



THE **INFORMATION** COMPANY

# *Curs 6*

# *Limbajul SQL*

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

## 6. Cereri din mai multe tabele (**JOIN-uri**)

Scopul acestui curs este de a studia modalitatea preluării de date din mai multe tabele.

Pentru a putea realiza acest lucru, făcând legături între tabelele unei baze de date, putem folosi:

1. **JOIN**-urile proprietate **ORACLE**
2. **JOIN**-urile **ANSI/ISO SQL99**

## 6. Cereri din mai multe tabele (**JOIN-uri**)

Tipuri de **JOIN**-uri:

Join-uri proprietate ORACLE	Join-uri SQL 1999
Cartesian Product	Cross join
Equijoin	Natural join
Non-equijoin	Using clause
Outer join	Full (two sided) outer joins
Self join	Arbitrary join conditions for outer joins

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Cartesian Product and the Join Operations

**JOIN**-uri proprietatea **Oracle**

Pentru a prelua date din mai multe tabele, forma de bază a unei instrucțiuni **SELECT** constă în adăugarea unei **condiții de legătură (join)** în clauza **WHERE**

Numele coloanei trebuie prefixat de numele tabelului în situațiile când același nume de coloană apare în mai multe tabele.

# Cartesian Product and the Join Operations

```
SELECT table1.column, table2.column  
FROM table1, table2  
WHERE table1.column1 = table2.column2;
```



# Cartesian Product and the Join Operations

## PRODUSUL CARTEZIAN (CARTESIAN PRODUCT)

Presupune că *toate liniile din prima tabelă să fie unite (legate) cu toate liniile din tabela a doua.*

Se produce atunci când:

- 1) condiția de join este omisă
- 2) condiția de join nu este validă

Pentru a evita produsul cartezian trebuie adăugată o condiție de join validă.

Produsul cartezian generează foarte multe linii și este folosit foarte rar.

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Equijoin

## EQUIJOIN

Uneori denumit **simple join** sau **inner join**, *un equijoin este o legătură între tabele care combină linii ce au valori echivalente pentru coloanele specificate.*

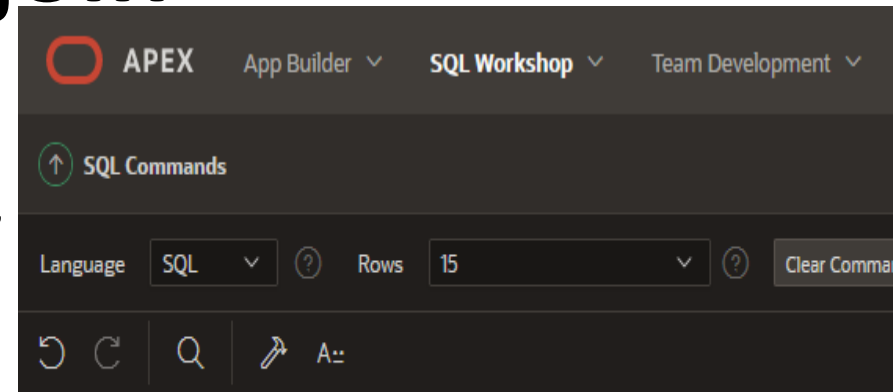
Exemplu:

```
SELECT EMP.empno, EMP.ename,  
        EMP.deptno, DEPT.deptno,  
        DEPT.loc  
FROM EMP, DEPT  
WHERE EMP.deptno = DEPT.deptno;
```

# Equijoin

## EQUIJOIN

Uneori denumit **simple join** sau **inner join**, *un equijoin este o legătură între tabele care combină linii ce au valori echivalente pentru coloanele specificate.*



```
1 SELECT EMP.empno, EMP.ename, EMP.deptno, DEPT.deptno, DEPT.loc
2 FROM EMP, DEPT
3 WHERE EMP.deptno = DEPT.deptno;
```

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7782	CLARK	10	10	NEW YORK
7954	MILLER	10	10	NEW YORK
7859	KING	10	10	NEW YORK
7902	FORD	20	20	DALLAS
7788	SCOTT	20	20	DALLAS
7566	JONES	20	20	DALLAS
7369	SMITH	20	20	DALLAS
7876	ADAMS	20	20	DALLAS
7521	WARD	30	30	CHICAGO
7854	MARTIN	30	30	CHICAGO
7844	TURNER	30	30	CHICAGO
7900	JAMES	30	30	CHICAGO
7499	ALLEN	30	30	CHICAGO
7698	BLAKE	30	30	CHICAGO

14 rows returned in 0.02 seconds [Download](#)

# Equijoin

Exemplu:

```
SELECT EMP.empno, EMP.ename,
       EMP.deptno, DEPT.deptno,
       DEPT.loc
FROM EMP, DEPT
WHERE EMP.deptno = DEPT.deptno;
```

APEX App Builder SQL Workshop Team Development

SQL Commands

Language SQL Rows 15 Clear Command

```
1 SELECT EMP.empno, EMP.ename, EMP.deptno, DEPT.deptno, DEPT.loc
2 FROM EMP, DEPT
3 WHERE EMP.deptno = DEPT.deptno;
```

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7780	CLARK	10	10	NEW YORK
7934	MILLER	10	10	NEW YORK
7839	KING	10	10	NEW YORK
7902	FORD	20	20	DALLAS
7788	SCOTT	20	20	DALLAS
7566	JONES	20	20	DALLAS
7569	SMITH	20	20	DALLAS
7836	ADAMS	20	20	DALLAS
7521	WARD	30	30	CHICAGO
7654	MARTIN	30	30	CHICAGO
7844	TURNER	30	30	CHICAGO
7900	JAMES	30	30	CHICAGO
7499	ALLEN	30	30	CHICAGO
7698	BLAKE	30	30	CHICAGO

14 rows returned in 0.00 seconds Download

# Equijoin

DEPT

Table Data Indexes Model Constraints Grants Statistics UI Defaults Triggers Dependencies SQL REST Sample Queries

Query Count Rows Insert Row Load Data

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON

Cheie  
străină  
(Foreign  
key)

Cheie  
primară  
(Primary  
key)

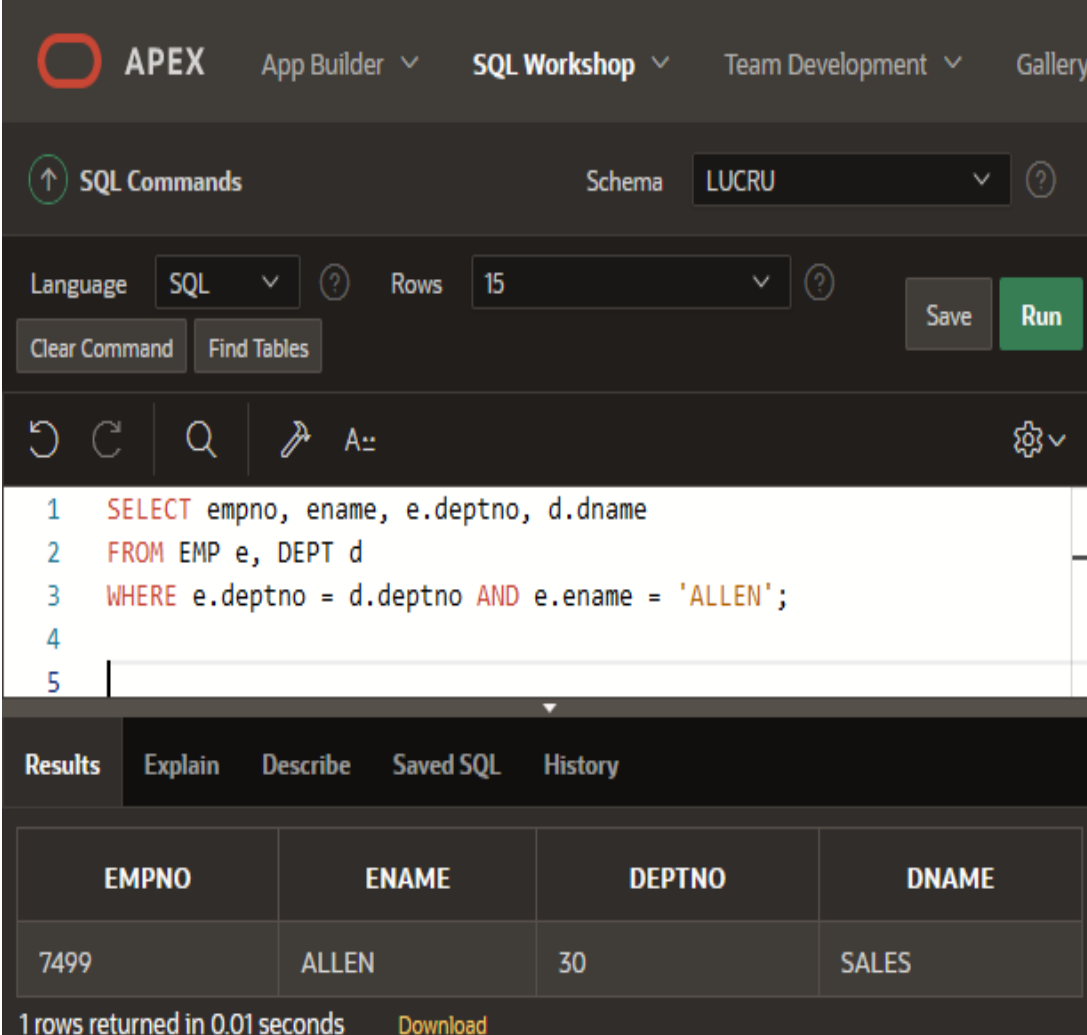
EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/07/1981	5000	-	10
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30
7782	CLARK	MANAGER	7839	06/09/1981	2450	-	10
7566	JONES	MANAGER	7839	04/02/1981	2975	-	20
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20
7369	SMITH	CLERK	7902	12/17/1980	800	-	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	09/08/1981	1500	0	30
7876	ADAMS	CLERK	7788	01/12/1983	1100	-	20
7900	JAMES	CLERK	7698	12/13/1981	950	-	30
7934	MILLER	CLERK	7782	01/23/1982	1300	-	10

# Equijoin

## FOLOSIREA OPERATORULUI **AND**

Ca și la interogările care folosesc o singură tabelă, se poate folosi operatorul **AND** pentru a restricționa liniile selectate.



The screenshot shows the Oracle APEX SQL Workshop interface. The top navigation bar includes 'APEX', 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. The 'SQL Commands' section is active, with the schema set to 'LUCRU'. The language is 'SQL' and the number of rows to display is '15'. The query being executed is:

```
1 SELECT empno, ename, e.deptno, d.dname
2 FROM EMP e, DEPT d
3 WHERE e.deptno = d.deptno AND e.ename = 'ALLEN';
4
5
```

The results are displayed in a table with the following columns: EMPNO, ENAME, DEPTNO, and DNAME. The results show one row for the employee ALLEN in the SALES department.

EMPNO	ENAME	DEPTNO	DNAME
7499	ALLEN	30	SALES

1 rows returned in 0.01 seconds [Download](#)

# Equijoin

## ALIAS-URI

Atunci când denumirile coloanelor și tabelelor sunt mari, devine incomod de lucrat cu acestea.

**Pentru a scurta denumirile respective, se folosesc alias-urile.**

Se pot folosi **alias**-uri atât pentru coloane cât și pentru tabele.

Dacă este precizat un alias pentru o tabelă în clauza **FROM**, atunci alias-ul respectiv trebuie să înlocuiască numele tabelii în clauza **SELECT**.



# Equijoin

Exemplu:

```
SELECT e.empno, e.ename,
       a.deptno, d.deptno, d.loc
FROM EMP e, DEPT d
WHERE e.deptno = d.deptno;
```

The screenshot shows the APEX SQL Workshop interface. At the top, there are navigation tabs for 'App Builder', 'SQL Workshop', 'Team Development', and 'Gallery'. Below this, the 'SQL Commands' section is visible, with the 'Schema' set to 'LUCRU'. The 'Language' is set to 'SQL' and 'Rows' is set to '15'. There are buttons for 'Clear Command', 'Find Tables', 'Save', and 'Run'. The SQL command is entered in the main text area:

```
1 SELECT e.empno, e.ename, e.deptno, d.deptno, d.loc
2 FROM EMP e, DEPT d
3 WHERE e.deptno = d.deptno;
4
```

The screenshot shows the 'Results' pane of the SQL Workshop. It displays a table with 14 rows and 5 columns: EMPNO, ENAME, DEPTNO, DEPTNO, and LOC. The data is as follows:

EMPNO	ENAME	DEPTNO	DEPTNO	LOC
7782	CLARK	10	10	NEW YORK
7784	MILLER	10	10	NEW YORK
7789	KING	10	10	NEW YORK
7902	FORD	20	20	DALLAS
7788	SCOTT	20	20	DALLAS
7566	JONES	20	20	DALLAS
7569	SMITH	20	20	DALLAS
7876	ADAMS	20	20	DALLAS
7521	WARD	30	30	CHICAGO
7654	MARTIN	30	30	CHICAGO
7844	TURNER	30	30	CHICAGO
7900	JAMES	30	30	CHICAGO
7400	ALLEN	30	30	CHICAGO
7698	BLAKE	30	30	CHICAGO

At the bottom of the results pane, it states '14 rows returned in 0.01 seconds' and provides a 'Download' link.

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# NONEQUIJOINS

Este posibil să dorim *să extragem date dintr-o tabelă ce nu are coloană corespondentă în cealaltă tabelă* (exemplu – când datele se înregistrează ca domenii de valori).

În această situație se folosește **nonequijoin**-ul.

În acest tip de join, deoarece nu există o potrivire exactă între 2 coloane din fiecare tabelă, nu se folosește operatorul de egalitate.

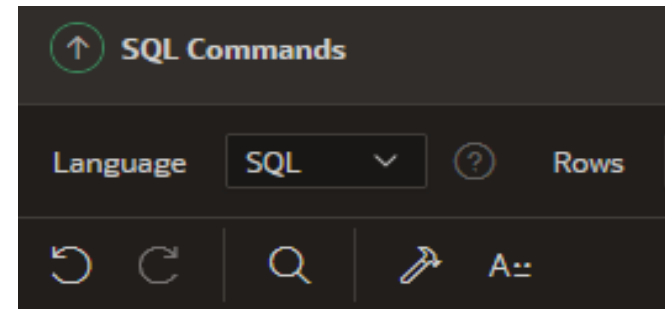
Se pot folosi operatorii:

**<=, >=, BETWEEN...AND**

# NONEQUIJOINS

Exemplu:

```
SELECT *
FROM EMP E, DEPT D
WHERE E.deptno > D.deptno
AND E.deptno IN (10, 30);
```



```
1 SELECT *
2 FROM EMP E, DEPT D
3 WHERE E.deptno > D.deptno
4 AND E.deptno IN (10, 30);
```

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO	DEPTNO	DNAME	LOC
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30	10	ACCOUNTING	NEW YORK
7844	TURNER	SALESMAN	7698	09/08/1981	1500	0	30	10	ACCOUNTING	NEW YORK
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30	10	ACCOUNTING	NEW YORK
7900	JAMES	CLERK	7698	12/03/1981	950	-	30	10	ACCOUNTING	NEW YORK
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30	10	ACCOUNTING	NEW YORK
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30	10	ACCOUNTING	NEW YORK
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30	20	RESEARCH	DALLAS
7844	TURNER	SALESMAN	7698	09/08/1981	1500	0	30	20	RESEARCH	DALLAS
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30	20	RESEARCH	DALLAS
7900	JAMES	CLERK	7698	12/03/1981	950	-	30	20	RESEARCH	DALLAS
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30	20	RESEARCH	DALLAS
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30	20	RESEARCH	DALLAS

12 rows returned in 0.01 seconds [Download](#)

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. Cartesian Product

#### 6.1.2. Equijoin

#### 6.1.3. Non-equijoin

#### 6.1.4. Outer join

#### 6.1.5. Self join

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. Cross join

#### 6.2.2. Natural join

#### 6.2.3. Using clause

#### 6.2.4. Full (two sided) outer joins

#### 6.2.5. Arbitrary join conditions for outer joins

### 6.3. Operatorii pe mulțimi

# Outer Joins

Join-urile studiate până în acest moment au avut ca rezultat linii care:

- 1. fie au avut o valoare care să corespundă în ambele tabele*
- 2. fie o valoare într-o tabelă se regăsea în intervalul dintre 2 valori ale celeilalte tabele*

Liniile care nu îndeplineau condițiile nu erau selectate.

# Outer Joins

Uneori, *dorim să extragem toate datele dintr-o tabelă, chiar dacă nu au valori care să se potrivească în cealaltă tabelă ( “missing data” )*.

Acest lucru se realizează folosind **outer join-ul**.

Operatorul pentru **outer join** este semnul plus pus între paranteze rotunde – **(+)**

# Outer Joins

Un **outer join** este folosit *pentru a vizualiza toate liniile care au valoare corespondentă în cealaltă tabelă și liniile dintr-o tabelă care nu au valoare corespondentă în cealaltă tabelă.*

*Pentru a indica tabela deficitară (care poate avea date lipsă – “missing data”), se pune operatorul (+) după numele coloanei din tabelă, în clauza **WHERE**.*



# Outer Joins

## OBSERVAȚIE:

Un **outer join** nu poate folosi operatorul **IN** și nu poate fi legat la altă condiție prin operatorul **OR**.

```
SELECT table1.column, table2.column  
FROM table1, table2  
WHERE table1.column(+) = table2.column
```

```
SELECT table1.column, table2.column  
FROM table1, table2  
WHERE table1.column = table2.column(+)
```

# Outer Joins

Nu există angajați in departamentele 40, 50 și 60.

DEPT		
DEPTNO	DNAME	LOC
60	PRODUCTION	CLUJNAPOCA
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
50	MARKETING	BUCHAREST

EMP

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT	-	11/17/1981	5000	-	10
7698	BLAKE	MANAGER	7839	05/01/1981	2850	-	30
7782	CLARK	MANAGER	7839	06/09/1981	2450	-	10
7566	JONES	MANAGER	7839	04/02/1981	2975	-	20
7788	SCOTT	ANALYST	7566	12/09/1982	3000	-	20
7902	FORD	ANALYST	7566	12/03/1981	3000	-	20
7369	SMITH	CLERK	7902	12/17/1980	800	-	20
7499	ALLEN	SALESMAN	7698	02/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	02/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	09/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	09/08/1981	1500	0	30
7876	ADAMS	CLERK	7788	01/12/1983	1100	-	20
7900	JAMES	CLERK	7698	12/03/1981	950	-	30
7934	MILLER	CLERK	7782	01/23/1982	1300	-	10

# Outer Joins

```
SELECT e.empno, a.ename,
       d.deptno
FROM EMP e, DEPT d
WHERE a.deptno(+) = d.deptno;
```

The screenshot shows the APEX SQL Workshop interface. At the top, there are tabs for 'APEX', 'App Builder', and 'SQL Workshop'. Below the tabs, there is a 'SQL Commands' section with an upward arrow icon. The 'Language' is set to 'SQL' and 'Rows' is set to '20'. The SQL command is displayed in a text area, and the execution results are shown below it.

```
1 SELECT e.empno, e.ename, d.deptno
2 FROM EMP e, DEPT d
3 WHERE e.deptno(+) = d.deptno;
```

EMPNO	ENAME	DEPTNO
7839	KING	10
7698	BLAKE	30
7782	CLARK	10
7566	JONES	20
7788	SCOTT	20
7902	FORD	20
7369	SMITH	20
7499	ALLEN	30
7521	WARD	30
7654	MARTIN	30
7844	TURNER	30
7876	ADAMS	20
7900	JAMES	30
7934	MILLER	10
-	-	50
-	-	40
-	-	60

17 rows returned in 0.01 seconds [Download](#)

# Outer Joins

## APLICAȚII:

- 1) Creați o interogare care are ca rezultat afișarea numelui (nume) și id-ul și numele departamentului pentru angajați. Includeți toți angajații, chiar dacă nu sunt asigurați unui departament.
- 2) Modificați interogarea din problema anterioară pentru a afișa toate id-urile departamentelor, chiar dacă nu au angajați asociați lor.

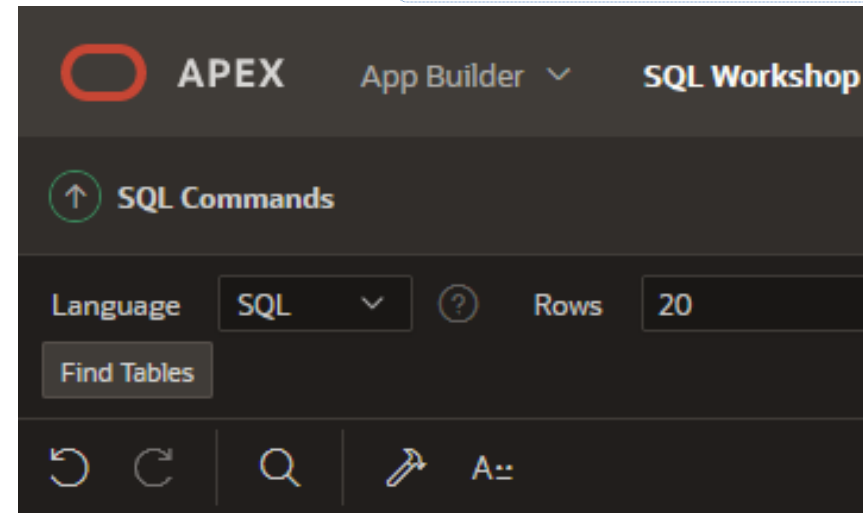
# Outer Joins

- 1) Creați o interogare care are ca rezultat afișarea numelui (nume) și id-ul și numele departamentului pentru angajați. Includeți toți angajații, chiar dacă nu sunt asigurați unui departament.

```
SELECT e.name, e.deptno, d.dname  
FROM EMP e, DEPT d  
WHERE e.deptno = d.deptno(+);
```

# Outer Joins

1) Creați o interogare care are ca rezultat afișarea numelui (nume) și id-ul și numele departamentului pentru angajați. Includeți toți angajații, chiar dacă nu sunt asignați unui departament.



```
1 SELECT e.ename, e.deptno, d.dname
2 FROM EMP e, DEPT d
3 WHERE e.deptno = d.deptno(+);
```

ENAME	DEPTNO	DNAME
KING	10	ACCOUNTING
CLARK	10	ACCOUNTING
MILLER	10	ACCOUNTING
JONES	20	RESEARCH
SCOTT	20	RESEARCH
FORD	20	RESEARCH
SMITH	20	RESEARCH
ADAMS	20	RESEARCH
BLAKE	30	SALES
ALLEN	30	SALES
WARD	30	SALES
MARTIN	30	SALES
TURNER	30	SALES
JAMES	30	SALES

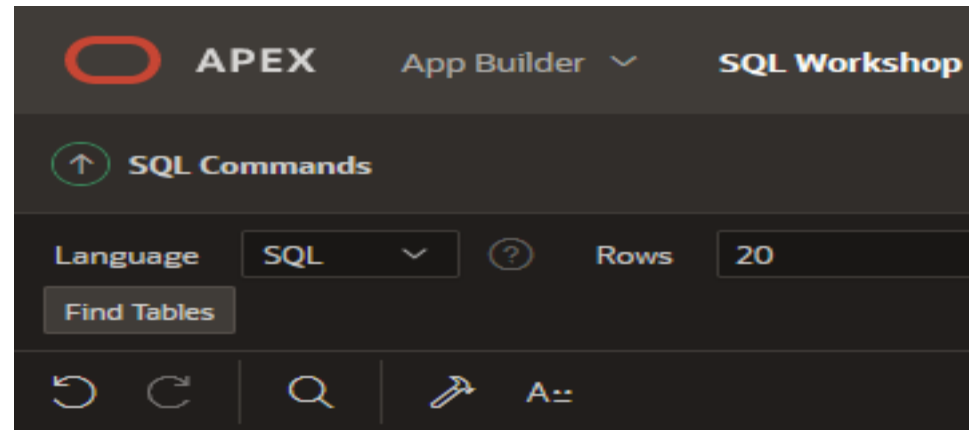
# Outer Joins

2) Modificați interogarea din problema anterioară pentru a afișa toate id-urile departamentelor, chiar dacă nu au angajați asociați lor.

```
SELECT e.ename, e.deptno, d.dname  
FROM EMP e, DEPT d  
WHERE e.deptno(+) = d.deptno;
```

# Outer Joins

2) Modificați interogarea din problema anterioară pentru a afișa toate id-urile departamentelor, chiar dacă nu au angajați asociați lor.



```

1 SELECT e.ename, e.deptno, d.dname
2 FROM EMP e, DEPT d
3 WHERE e.deptno(+) = d.deptno;

```

ENAME	DEPTNO	DNAME
CLARK	10	ACCOUNTING
MILLER	10	ACCOUNTING
KING	10	ACCOUNTING
FORD	20	RESEARCH
SCOTT	20	RESEARCH
JONES	20	RESEARCH
SMITH	20	RESEARCH
ADAMS	20	RESEARCH
WARD	30	SALES
MARTIN	30	SALES
TURNER	30	SALES
JAMES	30	SALES
ALLEN	30	SALES
BLAKE	30	SALES
-	-	OPERATIONS
-	-	MARKETING
-	-	PRODUCTION

17 rows returned in 0.01 seconds [Download](#)



# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Self Joins

- *În modelarea de date, uneori este necesar să punem în evidență o entitate în relație cu ea însăși.*
- Un exemplu este entitatea **angajat**.
- Un angajat poate fi și manager.
- Odată ce avem tabela EMP, devine necesară o relație specială numită **self join** (un join de la tabela EMP la ea însăși), pentru a afla numele managerului pentru fiecare angajat.

# Self Joins

- *Pentru a face join de la o tabelă la ea însăși, tablei îi sunt asociate 2 alias-uri.*
- Astfel, pentru baza de date, există în aparență 2 tabele.

# Self Joins

EMP		
EMPNO	ENAME	MGR
7839	KING	-
7698	BLAKE	7839
7782	CLARK	7839
7566	JONES	7839
7788	SCOTT	7566
7902	FORD	7566
7369	SMITH	7902
7499	ALLEN	7698
7521	WARD	7698
7654	MARTIN	7698
7844	TURNER	7698
7876	ADAMS	7788
7900	JAMES	7698
7934	MILLER	7782

MGR in  
tabela  
angajatilor  
are aceeasi  
valoare in  
EMPNO din  
tabela  
managerilor

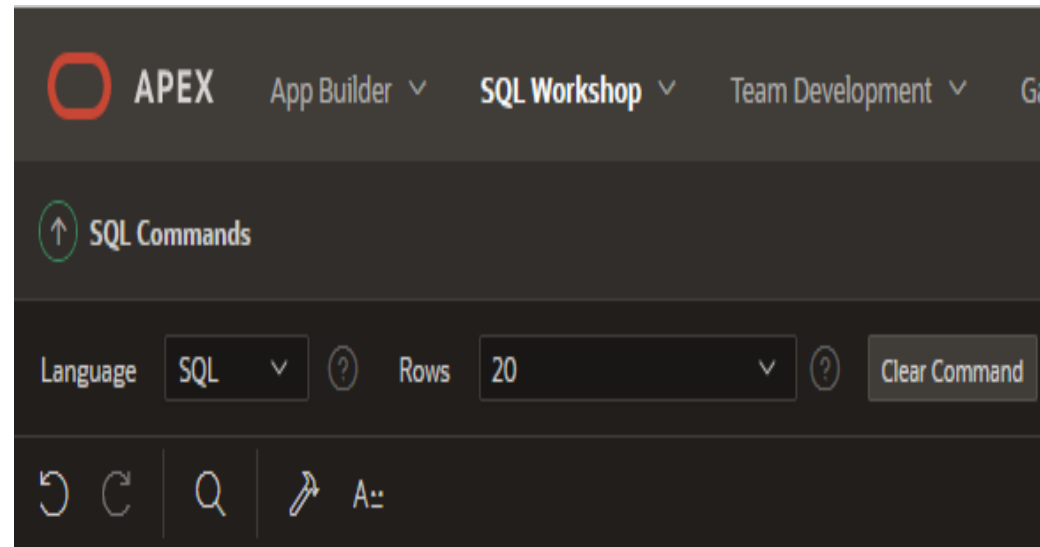
EMP	
EMPNO	ENAME
7839	KING
7698	BLAKE
7782	CLARK
7566	JONES
7788	SCOTT
7902	FORD
7369	SMITH
7499	ALLEN
7521	WARD
7654	MARTIN
7844	TURNER
7876	ADAMS
7900	JAMES
7934	MILLER

# Self Joins

## Exemplu:

```
SELECT lucrator.ename || ' lucreaza pentru ' || manager.ename
FROM EMP lucrator, EMP manager
WHERE lucrator.mgr = manager.empno;
```

Results	Explain	Describe	Saved SQL	History
LUCRATOR.ENAME  'LUCREAZAPENTRU'  MANAGER.ENAME				
FORD lucreaza pentru JONES				
SCOTT lucreaza pentru JONES				
MARTIN lucreaza pentru BLAKE				
TURNER lucreaza pentru BLAKE				
WARD lucreaza pentru BLAKE				
ALLEN lucreaza pentru BLAKE				
JAMES lucreaza pentru BLAKE				
MILLER lucreaza pentru CLARK				
ADAMS lucreaza pentru SCOTT				
BLAKE lucreaza pentru KING				



```
1 SELECT lucrator.ename || ' lucreaza pentru ' || manager.ename
2 FROM EMP lucrator, EMP manager
3 WHERE lucrator.mgr = manager.empno;
```

# Self Joins

## APLICAȚII

- 1) Afișați numele și numărul pentru fiecare angajat împreună cu numele și numărul managerului. Denumiți coloanele: Angajat, Ang#, Manager și Mgr#.
- 2) Modificați interogarea 1 pentru a afișa toți angajații și managerii lor chiar dacă un angajat nu are un manager. Ordonăți lista obținută alfabetic, după numele angajaților.

# Self Joins

1) Afișați numele și numărul pentru fiecare angajat împreună cu numele și numărul managerului. Denumiți coloanele: Angajat, Ang#, Manager și Mgr#.

```
SELECT    a.ename AS "Angajat", a.empno AS
            "Ang#", m.ename AS "Manager", m.empno AS
            "Mgr#"
FROM EMP a, EMP m
WHERE a.empno = m.empno;
```

# Self Joins

1) Afișați numele și numărul pentru fiecare angajat împreună cu numele și numărul managerului.

Denumiți coloanele: Angajat, Ang#, Manager și Mgr#.

```

↑ SQL Commands

Language SQL Rows 20 Clear Command Find Tables

↶ ↷ 🔍 📌 A::
  
```

```

1 SELECT a.ename AS "Angajat", a.empno AS "Ang#", m.ename AS "Manager", m.empno AS "Mgr#"
2 FROM EMP a, EMP m
3 WHERE a.empno = m.empno;
  
```

Angajat	Ang#	Manager	Mgr#
KING	7839	KING	7839
BLAKE	7698	BLAKE	7698
CLARK	7782	CLARK	7782
JONES	7566	JONES	7566
SCOTT	7788	SCOTT	7788
FORD	7902	FORD	7902
SMITH	7369	SMITH	7369
ALLEN	7499	ALLEN	7499
WARD	7521	WARD	7521
MARTIN	7654	MARTIN	7654
TURNER	7844	TURNER	7844
ADAMS	7876	ADAMS	7876
JAMES	7900	JAMES	7900
MILLER	7934	MILLER	7934

14 rows returned in 0.00 seconds [Download](#)



# Self Joins

2) Modificați interogarea 1 pentru a afișa toți angajații și managerii lor chiar dacă un angajat nu are un manager.

Ordonăți lista obținută alfabetic, după numele angajaților.

```
SELECT      a.ename AS "Angajat", a.empno AS  
            "Ang#", m.ename AS "Manager", m.empno AS  
            "Mgr#"  
FROM EMP a, EMP m  
WHERE a.empno = m.empno  
ORDER BY a.ename;
```

# Self Joins

↑ SQL Commands

Language SQL

Rows 20

Clear Command

Find Tables

↶ ↷ 🔍 📌 A::

```
1 SELECT a.ename AS "Angajat", a.empno AS "Ang#", m.ename AS "Manager", m.empno AS "Mgr#"
2 FROM EMP a, EMP m
3 WHERE a.empno = m.empno
4 ORDER BY a.ename;
```

Results	Explain	Describe	Saved SQL	History
Angajat	Ang#	Manager	Mgr#	
ADAMS	7876	ADAMS	7876	
ALLEN	7499	ALLEN	7499	
BLAKE	7698	BLAKE	7698	
CLARK	7782	CLARK	7782	
FORD	7902	FORD	7902	
JAMES	7900	JAMES	7900	
JONES	7566	JONES	7566	
KING	7839	KING	7839	
MARTIN	7654	MARTIN	7654	
MILLER	7934	MILLER	7934	
SCOTT	7788	SCOTT	7788	
SMITH	7369	SMITH	7369	
TURNER	7844	TURNER	7844	
WARD	7521	WARD	7521	

14 rows returned in 0.01 seconds [Download](#)

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Cross Joins and Natural Joins

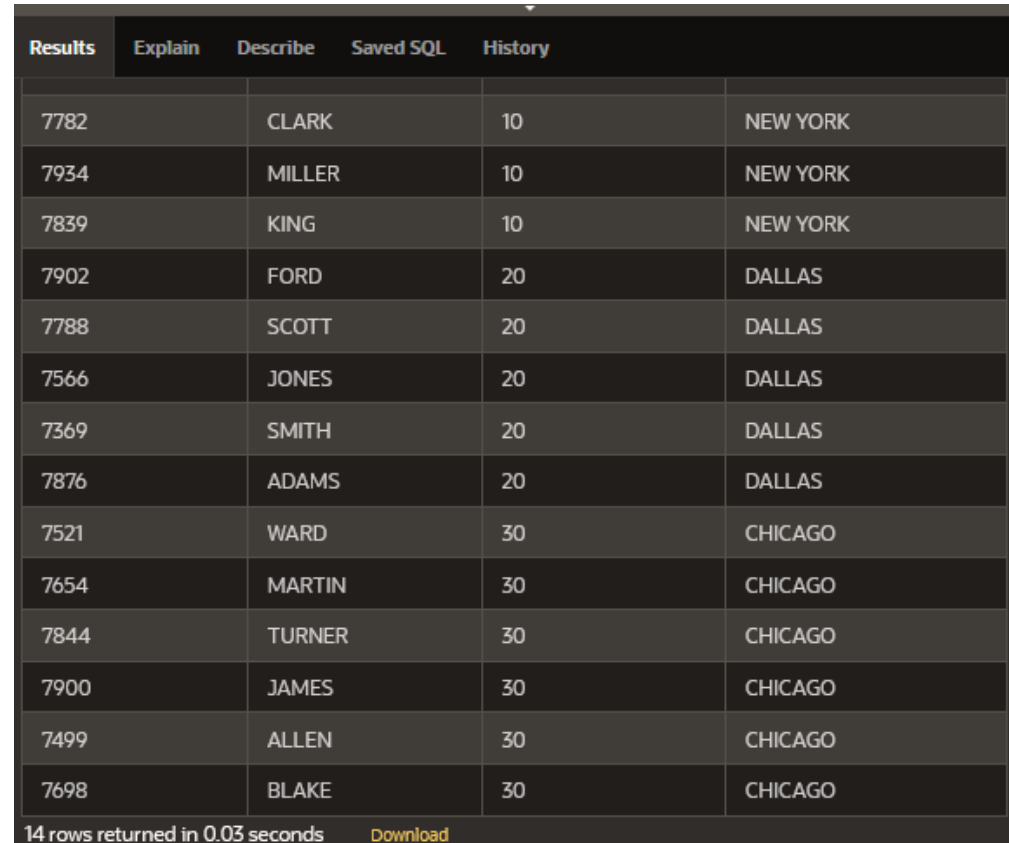
## NATURAL JOIN

- Este un **ANSI/ISO SQL:1999 join** echivalent cu **equijoin**-ul.
- *Se face pe baza tuturor coloanelor care au același nume din 2 tabele; coloanele respective trebuie să aibă același tip de date.*
- Sunt selectate liniile care au valori egale în toate coloanele corespondente.

# Cross Joins and Natural Joins

Exemplu:

```
SELECT empno, ename, deptno, loc  
FROM EMP NATURAL JOIN DEPT;
```



The screenshot shows a SQL query execution interface with a dark theme. The query is: `SELECT empno, ename, deptno, loc FROM EMP NATURAL JOIN DEPT;`. The results are displayed in a table with 14 rows. The columns are empno, ename, deptno, and loc. The results show a Cartesian product of the two tables, with 14 rows returned in 0.03 seconds.

empno	ename	deptno	loc
7782	CLARK	10	NEW YORK
7934	MILLER	10	NEW YORK
7839	KING	10	NEW YORK
7902	FORD	20	DALLAS
7788	SCOTT	20	DALLAS
7566	JONES	20	DALLAS
7369	SMITH	20	DALLAS
7876	ADAMS	20	DALLAS
7521	WARD	30	CHICAGO
7654	MARTIN	30	CHICAGO
7844	TURNER	30	CHICAGO
7900	JAMES	30	CHICAGO
7499	ALLEN	30	CHICAGO
7698	BLAKE	30	CHICAGO

14 rows returned in 0.03 seconds [Download](#)

# Cross Joins and Natural Joins

## CROSS JOIN

Realizează produsul cartezian pentru două  
tabele.

Exemplu:

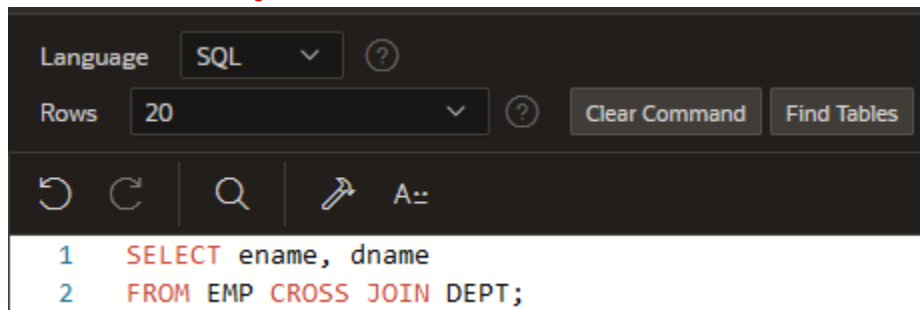
```
SELECT ename, dname
```

```
FROM EMP CROSS JOIN DEPT;
```

este echivalentă cu instrucțiunea:

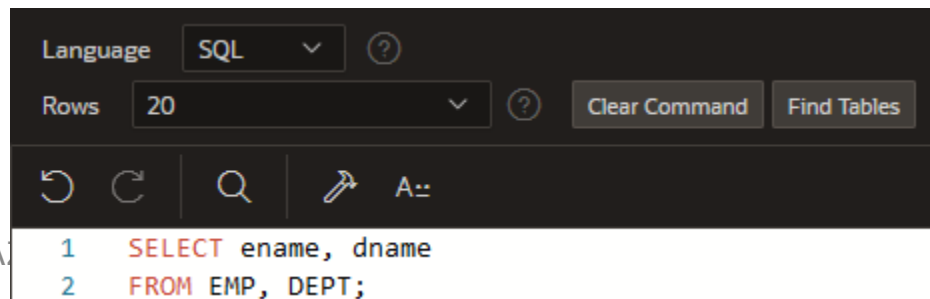
```
SELECT ename, dname
```

```
FROM EMP, DEPT;
```



The screenshot shows a SQL IDE interface with a dark theme. At the top, there is a 'Language' dropdown set to 'SQL' and a 'Rows' dropdown set to '20'. Below these are buttons for 'Clear Command' and 'Find Tables'. The main area contains a SQL query:

```
1 SELECT ename, dname
2 FROM EMP CROSS JOIN DEPT;
```



The screenshot shows a SQL IDE interface with a dark theme, identical to the one above. It displays the same SQL query but using natural join syntax:

```
1 SELECT ename, dname
2 FROM EMP, DEPT;
```

# Cross Joins and Natural Joins

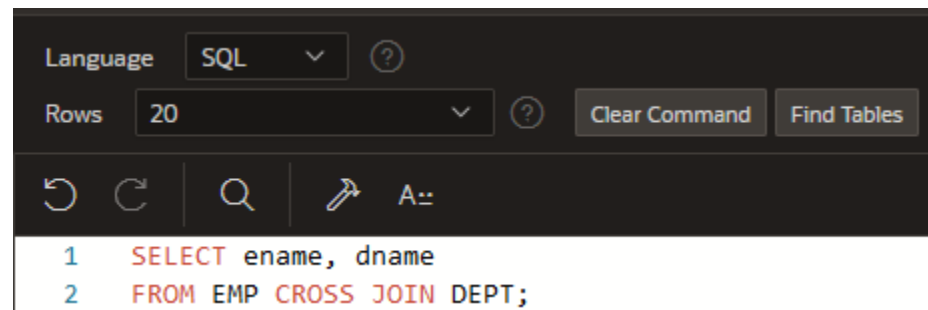
## APLICAȚII

- 1) Creați un **cross-join** care afișează numele și denumirea departmentului din tabelele **EMP** și **DEPT**.
- 2) Creați o interogare care folosește un **natural join** pentru a pune în legătură tabelele **DEPT** și **EMP**. Afișați id-ul, denumirea departamentului și orașul.

# Cross Joins and Natural Joins

- 1) Creați un **cross-join** care afișează numele și denumirea departmentului din tabelele **EMP** și **DEPT**.

```
SELECT ename, dname  
FROM EMP CROSS JOIN DEPT;
```



The screenshot shows a SQL IDE interface with a dark theme. At the top, there is a 'Language' dropdown menu set to 'SQL' and a 'Rows' dropdown menu set to '20'. Below these are buttons for 'Clear Command' and 'Find Tables'. The main area displays the SQL query:

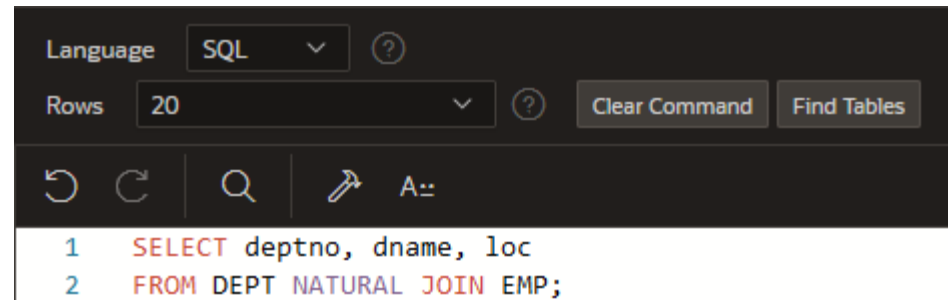
```
1 SELECT ename, dname  
2 FROM EMP CROSS JOIN DEPT;
```



## Cross Joins and Natural Joins

- 2) Creați o interogare care folosește un **natural join** pentru a pune în legătură tabelele **DEPT** și **EMP**. Afișați id-ul și denumirea departamentului, orașul.

```
SELECT deptno, dname, loc  
FROM DEPT  
NATURAL JOIN EMP;
```



The screenshot shows a SQL IDE interface with a dark theme. At the top, there is a 'Language' dropdown set to 'SQL' and a 'Rows' dropdown set to '20'. Below these are buttons for 'Clear Command' and 'Find Tables'. The main area shows the SQL query: `1 SELECT deptno, dname, loc` and `2 FROM DEPT NATURAL JOIN EMP;`. The query is highlighted in red and blue.

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Join Clauses

În această parte vom studia:

1. folosirea clauzelor **USING** și **ON**
2. realizarea unui join cu 3 tabele

## 1. Clauza **USING**

Într-un **natural join**, *dacă tabelele au coloane cu același nume dar tipuri diferite*, se produce eroare.

Pentru a evita această situație, clauza **JOIN** *se înlocuiește* cu clauza **USING**.

# Join Clauses

Clauza **USING** specifică coloanele care ar trebui folosite pentru pentru **equijoin**.

Coloana specificată în clauza **USING** nu trebuie să aibă nici un specificator (nume de tabela sau alias), în nici o parte din instrucțiunea **SELECT**.

# Join Clauses

```
Language SQL ?  
Rows 20 ? Clear Command Find Tables  
↶ ↷ 🔍 📌 A::
```

```
1 SELECT e.empno, e.ename, d.loc  
2 FROM EMP e  
3 JOIN DEPT d  
4 USING (deptno);
```

Exemplu:

**SELECT** e.empno, e.ename, d.loc  
**FROM** EMP e  
**JOIN** DEPT d  
**USING** (deptno);

EMPNO	ENAME	LOC
7782	CLARK	NEW YORK
7934	MILLER	NEW YORK
7839	KING	NEW YORK
7902	FORD	DALLAS
7788	SCOTT	DALLAS
7566	JONES	DALLAS
7369	SMITH	DALLAS
7876	ADAMS	DALLAS
7521	WARD	CHICAGO
7654	MARTIN	CHICAGO
7844	TURNER	CHICAGO
7900	JAMES	CHICAGO
7499	ALLEN	CHICAGO
7698	BLAKE	CHICAGO

14 rows returned in 0.01 seconds [Download](#)

# Join Clauses

## Clauza **ON**

Dacă coloanele folosite pentru **join** au *denumiri diferite* sau *dacă sunt folosiți operatorii*: **<**, **>** sau **BETWEEN**, atunci nu putem folosi clauza **USING**.

În această situație se folosește clauza **ON**.

Aceasta permite specificarea unei game variate de condiții pentru **join**-uri.

De asemenea, clauza **ON** ne permite să folosim **WHERE** pentru a restricționa linii dintr-o tabelă sau din ambele tabele.

# Join Clauses

## Exemple:

```
1) SELECT e.ename as "ANG", d.ename as "MGR"  
FROM EMP e JOIN EMP d  
ON (e.mgr = d.empno);
```

Se realizează un **self-join** pentru a selecta acei angajați care sunt și manageri.

# Join Clauses

1) **SELECT** e.ename as "ANG",  
 d.ename as "MGR"  
**FROM** EMP e **JOIN** EMP d  
**ON** (e.mgr = d.empno);

```
Language SQL ?
Rows 20 ? Clear Command Find Tables
? ? ? ? A::
```

```
1 SELECT e.ename as "ANG", d.ename as "MGR"
2 FROM EMP e JOIN EMP d
3 ON (e.mgr = d.empno);
```

ANG	MGR
FORD	JONES
SCOTT	JONES
MARTIN	BLAKE
TURNER	BLAKE
WARD	BLAKE
ALLEN	BLAKE
JAMES	BLAKE
MILLER	CLARK
ADAMS	SCOTT
BLAKE	KING
CLARK	KING
JONES	KING
SMITH	FORD

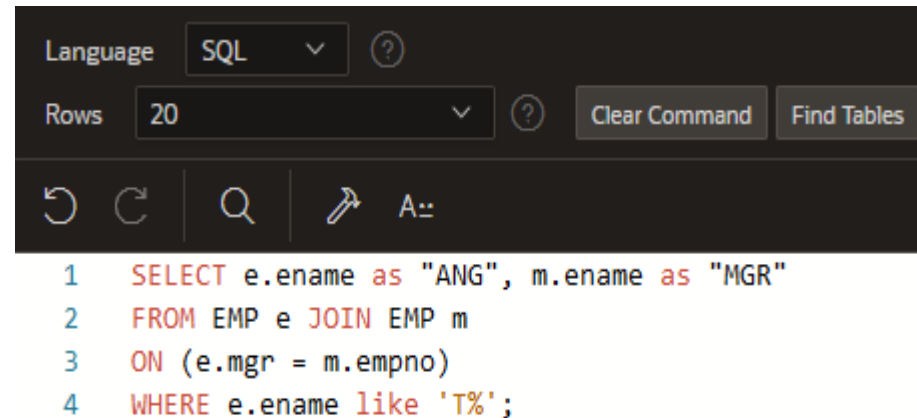
13 rows returned in 0.01 seconds [Download](#)



# Join Clauses

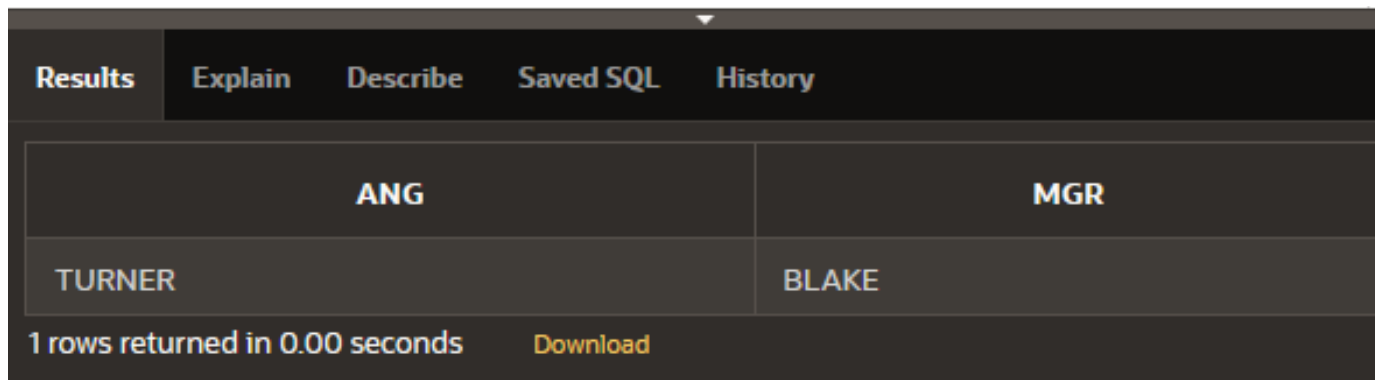
2) – folosirea clauzei **WHERE**

```
SELECT e.ename as "ANG", m.ename as "MGR"  
FROM EMP e JOIN EMP m  
ON (e.mgr = m.empno)  
WHERE e.ename like 'T%';
```



The screenshot shows a SQL IDE interface. At the top, there is a 'Language' dropdown set to 'SQL' and a 'Rows' dropdown set to '20'. There are buttons for 'Clear Command' and 'Find Tables'. Below the interface, the SQL query is displayed in a monospaced font, numbered 1 to 4.

```
1 SELECT e.ename as "ANG", m.ename as "MGR"  
2 FROM EMP e JOIN EMP m  
3 ON (e.mgr = m.empno)  
4 WHERE e.ename like 'T%';
```



The screenshot shows the 'Results' window of the SQL IDE. It has tabs for 'Results', 'Explain', 'Describe', 'Saved SQL', and 'History'. The 'Results' tab is active, showing a table with two columns: 'ANG' and 'MGR'. The table contains one row with the values 'TURNER' and 'BLAKE'. Below the table, it says '1 rows returned in 0.00 seconds' and there is a 'Download' button.

ANG	MGR
TURNER	BLAKE

1 rows returned in 0.00 seconds [Download](#)

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Join Clauses

## Realizarea unui join cu 3 tabele

Ambele clauze (**ON** si **USING**) se pot folosi pentru un astfel de join.

```
SELECT empno, loc, dname, city  
FROM EMP e JOIN DEPT d  
ON (d.deptno = e.deptno)  
JOIN locations l  
ON (d.loc = l.loc)
```

# Join Clauses

Comparatie intre join-urile proprietate **ORACLE** si join-urile **ANSI/ISO SQL 1999**

<b>Join-uri proprietate ORACLE</b>	<b>Join-uri ANSI/ISO SQL 1999</b>
<b>Produs cartezian</b>	Cross Join
<b>Equijoin</b>	Natural Join (daca coloanele de join au acelasi nume si acelasi tip de date)
	Clauza USING (daca coloanele de join au acelasi nume dar tipuri de date diferite)
	Clauza ON (daca coloanele au nume diferite)
<b>Non-equijoin</b>	Clauza ON

# Join Clauses

```
SELECT e.empno, e.ename, d.loc
FROM EMP e JOIN DEPT d
USING(deptno);
```

APEX App Builder SQL Workshop

SQL Commands

Language SQL Rows 20

⏪ ⏩ 🔍 📌 A::

```
1 SELECT e.empno, e.ename, d.loc
2 FROM EMP e JOIN DEPT d
3 USING(deptno);
```

EMPNO	ENAME	LOC
7782	CLARK	NEW YORK
7934	MILLER	NEW YORK
7839	KING	NEW YORK
7902	FORD	DALLAS
7788	SCOTT	DALLAS
7566	JONES	DALLAS
7369	SMITH	DALLAS
7876	ADAMS	DALLAS
7521	WARD	CHICAGO
7654	MARTIN	CHICAGO
7844	TURNER	CHICAGO
7900	JAMES	CHICAGO
7499	ALLEN	CHICAGO
7698	BLAKE	CHICAGO

14 rows returned in 0.00 seconds [Download](#)

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

## Inner vs. Outer Joins

În **ANSI-99 SQL**, *un join cu 2 sau mai multe tabele care returnează doar liniile care se potrivesc* se numește **inner join**.

Atunci când *un join returnează atât liniile care se potrivesc cât și cele care nu se potrivesc*, acesta se numește **outer join**.

Sunt trei tipuri de outer join în **ANSI/ISO SQL**:

- 1. LEFT OUTER JOIN**
- 2. RIGHT OUTER JOIN**
- 3. FULL OUTER JOIN**

# Inner vs. Outer Joins

## 1. LEFT OUTER JOIN

Sunt afișați și acei angajați care nu au desemnat un id\_dept (tabela DEPT este cea deficitară).

```
SELECT e.ename, e.deptno, d.dname  
FROM EMP e  
LEFT OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

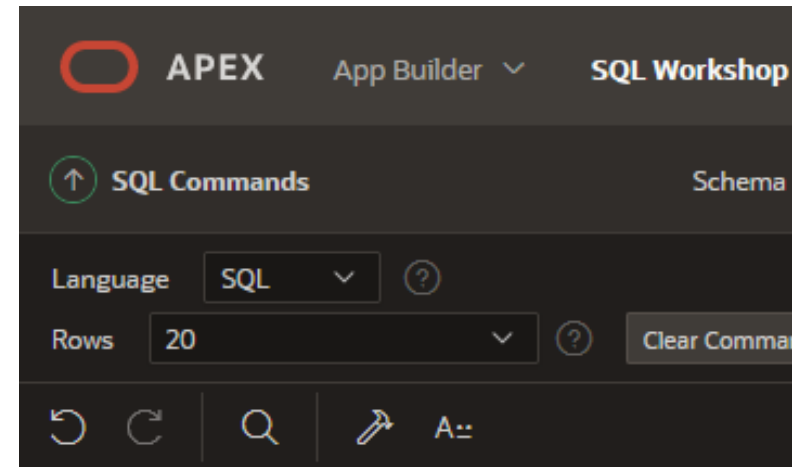


# Inner vs. Outer Joins

## 1. LEFT OUTER JOIN

ENAME	DEPTNO	DNAME
KING	10	ACCOUNTING
CLARK	10	ACCOUNTING
MILLER	10	ACCOUNTING
JONES	20	RESEARCH
SCOTT	20	RESEARCH
SMITH	20	RESEARCH
SMITH	20	RESEARCH
ADAMS	20	RESEARCH
BLAKE	30	SALES
ALLEN	30	SALES
WARD	30	SALES
SMITH	30	SALES
TURNER	30	SALES
JAMES	30	SALES
IONESCU	-	-
GEORGESCU	-	-
POPESCU	-	-

17 rows returned in 0.04 seconds [Download](#)



```

1  SELECT e.ename, e.deptno, d.dname
2  FROM EMP e
3  LEFT OUTER JOIN DEPT d
4  ON (e.deptno = d.deptno);

```

# Inner vs. Outer Joins

## 2. RIGHT OUTER JOIN

Sunt afișate și acele DEPT care nu au angajați.

```
SELECT e.ename, e.deptno, d.dname  
FROM EMP e  
RIGHT OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins

## 2. RIGHT OUTER JOIN

Sunt afișate și acele DEPT care nu au angajați.

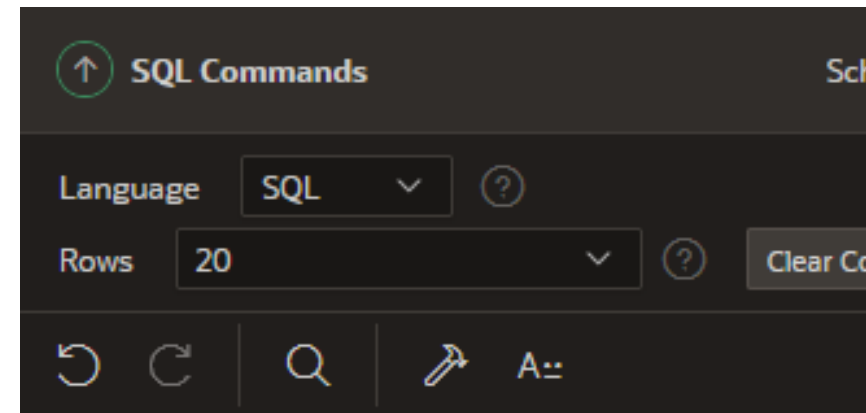
```
SELECT e.ename, e.deptno, d.dname  
FROM EMP e  
RIGHT OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins

## 2. RIGHT OUTER JOIN

Sunt afișate și acele DEPT care nu au angajați.

ENAME	DEPTNO	DNAME
CLARK	10	ACCOUNTING
MILLER	10	ACCOUNTING
KING	10	ACCOUNTING
FORD	20	RESEARCH
SCOTT	20	RESEARCH
JONES	20	RESEARCH
SMITH	20	RESEARCH
ADAMS	20	RESEARCH
WARD	30	SALES
MARTIN	30	SALES
TURNER	30	SALES
JAMES	30	SALES
ALLEN	30	SALES
BLAKE	30	SALES
-	-	OPERATIONS
-	-	MARKETING
-	-	PRODUCTION



```

1  SELECT e.ename, e.deptno, d.dname
2  FROM EMP e
3  RIGHT OUTER JOIN DEPT d
4  ON (e.deptno = d.deptno);

```

# Inner vs. Outer Joins

## 3. FULL OUTER JOIN

Returnează atât liniile care se potrivesc cât și cele care nu se potrivesc din ambele tabele.

Spre deosebire de **outer join-ul** proprietatea Oracle, care nu permitea folosirea operatorului **(+)** în ambele părți ale clauzei **WHERE**, **full outer join-ul** permite acest lucru.

```
SELECT e.ename, e.deptno, d.dname  
FROM EMP e  
FULL OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins

## 3. FULL OUTER JOIN

Returnează atât liniile care se potrivesc cât și cele care nu se potrivesc din ambele tabele.

ENAME	DEPTNO	DNAME
KING	10	ACCOUNTING
BLAKE	30	SALES
CLARK	10	ACCOUNTING
JONES	20	RESEARCH
SCOTT	20	RESEARCH
FORD	20	RESEARCH
SMITH	20	RESEARCH
ALLEN	30	SALES
WARD	30	SALES
MARTIN	30	SALES
TURNER	30	SALES
ADAMS	20	RESEARCH
JAMES	30	SALES
MILLER	10	ACCOUNTING
IONESCU	20	RESEARCH
-	-	MARKETING
-	-	OPERATIONS
-	-	FINANCIAL
-	-	Support
-	-	PRODUCTION

20 rows returned in 0.09 seconds [Download](#)

```

SQL Commands Schema
Language SQL
Rows 20 Clear Command
1 SELECT e.ename, e.deptno, d.dname
2 FROM EMP e
3 FULL OUTER JOIN DEPT d
4 ON (e.deptno = d.deptno);

```

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. **Cartesian Product**

#### 6.1.2. **Equijoin**

#### 6.1.3. **Non-equijoin**

#### 6.1.4. **Outer join**

#### 6.1.5. **Self join**

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. **Cross join**

#### 6.2.2. **Natural join**

#### 6.2.3. **Using clause**

#### 6.2.4. **Full (two sided) outer joins**

#### 6.2.5. **Arbitrary join conditions for outer joins**

### 6.3. Operatorii pe mulțimi

# Inner vs. Outer Joins

## APLICAȚII

- 1) Afișați numele și denumirea departamentului pentru toți angajații, inclusiv pentru cei care nu sunt desemnați la nici un departament.
- 2) Afișați numele și denumirea departamentului pentru toți angajații, inclusiv acele departamente care nu au nici un angajat asociat.
- 3) Afișați numele și denumirea departamentului pentru toți EMP, inclusiv acele departamente care nu au nici un angajat asociat și acei angajați care nu sunt desemnați nici unui departament.

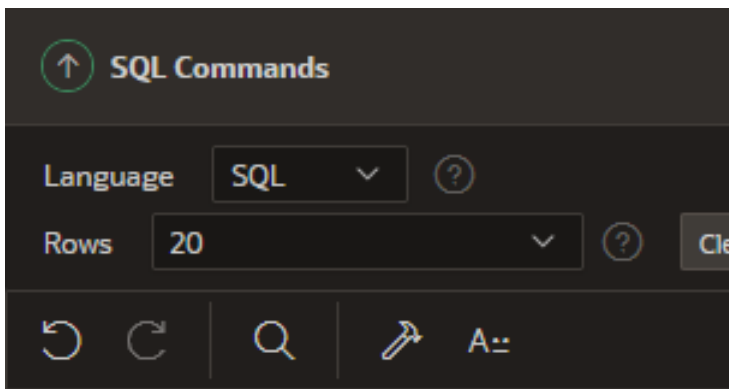


# Inner vs. Outer Joins

1) Afișați numele și denumirea departamentului pentru toți angajații, inclusiv pentru cei care nu sunt desemnați la nici un departament.

```
SELECT e.ename, e.job, d.dname  
FROM EMP e  
LEFT OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins



```

1  SELECT e.ename, e.job, d.dname
2  FROM EMP e
3  LEFT OUTER JOIN DEPT d
4  ON (e.deptno = d.deptno);

```

Results	Explain	Describe	Saved SQL	History
SCOTT				
			ANALYST	RESEARCH
FORD				
			ANALYST	RESEARCH
SMITH				
			CLERK	RESEARCH
ADAMS				
			CLERK	RESEARCH
IONESCU				
			SALESMAN	RESEARCH
BLAKE				
			MANAGER	SALES
ALLEN				
			SALESMAN	SALES
WARD				
			SALESMAN	SALES
MARTIN				
			SALESMAN	SALES
TURNER				
			SALESMAN	SALES
JAMES				
			CLERK	SALES

15 rows returned in 0.09 seconds [Download](#)

## Inner vs. Outer Joins

2) Afișați numele și denumirea departamentului pentru toți angajații, inclusiv acele departamente care nu au nici un angajat asociat.

```
SELECT e.ename, e.sal, d.dname  
FROM EMP e  
RIGHT OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins

↑ SQL Commands

Language

SQL

Rows

20

Clear



A::

```

1  SELECT e.ename, e.sal, d.dname
2  FROM EMP e
3  RIGHT OUTER JOIN DEPT d
4  ON (e.deptno = d.deptno);

```

Results	Explain	Describe	Saved SQL	History
MARTIN			1250	SALES
TURNER			1500	SALES
ADAMS			1100	RESEARCH
JAMES			950	SALES
MILLER			1300	ACCOUNTING
IONESCU			1000	RESEARCH
-			-	MARKETING
-			-	OPERATIONS
-			-	FINANCIAR
-			-	Support
-			-	PRODUCTION

20 rows returned in 0.01 seconds [Download](#)

## Inner vs. Outer Joins

3) Afișați numele și denumirea departamentului pentru toți EMP, inclusiv acele departamente care nu au nici un angajat asociat și acei angajați care nu sunt desemnați nici unui departament.

```
SELECT e.ename, e.job, d.dname  
FROM EMP e  
FULL OUTER JOIN DEPT d  
ON (e.deptno = d.deptno);
```

# Inner vs. Outer Joins

## SQL Commands

Language

SQL

Rows

20

Clear



A::

```

1  SELECT e.ename, e.job, d.dname
2  FROM EMP e
3  FULL OUTER JOIN DEPT d
4  ON (e.deptno = d.deptno);

```

Results	Explain	Describe	Saved SQL	History
MARTIN			SALESMAN	SALES
TURNER			SALESMAN	SALES
ADAMS			CLERK	RESEARCH
JAMES			CLERK	SALES
MILLER			CLERK	ACCOUNTING
IONESCU			SALESMAN	RESEARCH
-			-	MARKETING
-			-	OPERATIONS
-			-	FINANCIAR
-			-	Support
-			-	PRODUCTION

20 rows returned in 0.01 seconds [Download](#)

# Inner vs. Outer Joins

## ALTE APLICAȚII:

1. Afișați numele și numărul pentru angajați împreună cu numele și numărul managerului. Denumiți coloanele astfel: Angajat, Ang#, Manager, și Mgr#.
2. Modificați rezultatul de la problema 1 astfel încât să fie afișați toți angajații, inclusiv aceia care nu au manager. Ordonăți rezultatele după numărul angajatului.

## Inner vs. Outer Joins

1. Afișați numele și numărul pentru angajați împreună cu numele și numărul managerului. Denumiți coloanele astfel: Angajat, Ang#, Manager, și Mgr#.

```
SELECT      w.ename AS "Angajat",  
              w.empno AS "Ang#",  
              m.ename AS "Manager",  
              m.empno AS "Mgr#"  
FROM EMP w JOIN EMP m  
ON (w.mgr = m.empno );
```



# Inner vs. Outer Joins

SQL Commands Schema LUCRU

Language SQL Rows 20 Clear Command Find Tables Save

⏪ ⏩ 🔍 📌 A::

```

1 SELECT w.ename AS "Angajat", w.empno AS "Ang#", m.ename AS "Manager", m.empno AS "Mgr#"
2 FROM EMP w JOIN EMP m
3 ON (w.mgr = m.empno );

```

Angajat	Ang#	Manager	Mgr#
FORD	7902	JONES	7566
SCOTT	7788	JONES	7566
MARTIN	7654	BLAKE	7698
TURNER	7844	BLAKE	7698
WARD	7521	BLAKE	7698
ALLEN	7499	BLAKE	7698
JAMES	7900	BLAKE	7698
MILLER	7934	CLARK	7782
ADAMS	7876	SCOTT	7788
BLAKE	7698	KING	7839
CLARK	7782	KING	7839
JONES	7566	KING	7839
SMITH	7369	FORD	7902

13 rows returned in 0.01 seconds [Download](#)

## Inner vs. Outer Joins

2. Modificați rezultatul de la problema 1 astfel încât să fie afișați toți angajații, inclusiv aceia care nu au manager. Ordonați rezultatele după numărul angajatului.

```
SELECT      w.ename AS "Angajat",  
              w.empno AS "Ang#",  
              m.ename AS "Manager",  
              m.empno AS "Mgr#"  
FROM EMP w LEFT OUTER JOIN EMP m  
ON (w.mgr = m.empno );  
ORDER BY w.empno;
```

# Inner vs. Outer Joins

SQL Commands Schema LUCRU

Language SQL Rows 20 Clear Command Find Tables Save

```

1 SELECT w.ename AS "Angajat", w.empno AS "Ang#", m.ename AS "Manager", m.empno AS "Mgr#"
2 FROM EMP w LEFT OUTER JOIN EMP m
3 ON (w.mgr = m.empno )
4 ORDER BY w.empno;

```

Angajat	Ang#	Manager	Mgr#
SMITH	7369	FORD	7902
ALLEN	7499	BLAKE	7698
WARD	7521	BLAKE	7698
JONES	7566	KING	7839
MARTIN	7654	BLAKE	7698
BLAKE	7698	KING	7839
CLARK	7782	KING	7839
SCOTT	7788	JONES	7566
KING	7839	-	-
TURNER	7844	BLAKE	7698
ADAMS	7876	SCOTT	7788
JAMES	7900	BLAKE	7698
FORD	7902	JONES	7566
MILLER	7934	CLARK	7782

14 rows returned in 0.02 seconds [Download](#)

# *Limbajul SQL*

## 6. Cereri din mai multe tabele (**JOIN**-uri)

### 6.1. **JOIN**-urile proprietatea **ORACLE**

#### 6.1.1. Cartesian Product

#### 6.1.2. Equijoin

#### 6.1.3. Non-equijoin

#### 6.1.4. Outer join

#### 6.1.5. Self join

### 6.2. **JOIN**-urile **ANSI/ISO SQL99**

#### 6.2.1. Cross join

#### 6.2.2. Natural join

#### 6.2.3. Using clause

#### 6.2.4. Full (two sided) outer joins

#### 6.2.5. Arbitrary join conditions for outer joins

### 6.3. Operatorii pe mulțimi

# Operatorii pe mulțimi

*Operatorii pe mulțimi combină rezultatele obținute din două sau mai multe interogări.*

Cererile care conțin operatori pe mulțimi se numesc ***cereri compuse***.

Există patru operatori pe mulțimi:

1. **UNION**
2. **UNION ALL**
3. **INTERSECT**
4. **MINUS**

# Operatorii pe mulțimi

- Toți operatorii pe mulțimi au aceeași precedență.
- Dacă o instrucțiune **SQL** conține mai mulți operatori pe mulțimi, *server-ul Oracle* evaluează cererea de la stânga la dreapta (sau de sus în jos).
- Pentru a schimba această ordine de evaluare, se pot utiliza paranteze.

## Operatorii pe mulțimi

- Operatorul **UNION** returnează toate liniile selectate de două cereri, eliminând duplicatele.
- Acest operator nu ignoră valorile *null* și are precedență mai mică decât operatorul **IN**.

## Operatorii pe mulțimi

- Operatorul ***UNION ALL*** returnează toate liniile selectate de două cereri, fără a elimina duplicatele.
- Precizările făcute asupra operatorului ***UNION*** sunt valabile și în cazul operatorului ***UNION ALL***.
- În cererile asupra cărora se aplică ***UNION ALL*** nu poate fi utilizat cuvântul cheie ***DISTINCT***.



## Operatorii pe mulțimi

- Operatorul ***INTERSECT*** returnează toate liniile comune cererilor asupra cărora se aplică.
- Acest operator nu ignoră valorile *null*.

## Operatorii pe mulțimi

- Operatorul **MINUS** determină liniile returnate de prima cerere care nu apar în rezultatul celei de-a doua cereri.
- Pentru ca operatorul **MINUS** să funcționeze, este necesar ca toate coloanele din clauza **WHERE** să se afle și în clauza **SELECT**.

# Operatorii pe mulțimi

## **Observații:**

1. În mod implicit, pentru toți operatorii cu excepția lui **UNION ALL**, rezultatul este ordonat crescător după valorile primei coloane din clauza **SELECT**.
2. Pentru o cerere care utilizează operatori pe mulțimi, cu excepția lui **UNION ALL**, server-ul **Oracle** elimină liniile duplicat.

# Operatorii pe mulțimi

1. În instrucțiunile *SELECT* asupra cărora se aplică operatori pe mulțimi, coloanele selectate trebuie să corespundă ca număr și tip de date.
2. Nu este necesar ca numele coloanelor să fie identice.
3. Numele coloanelor din rezultat sunt determinate de numele care apar în clauza *SELECT* a primei cereri.

# Operatorii pe mulțimi

## APLICAȚII:

1. Să se afișeze: codurile departamentelor al căror nume conține șirul "CC" sau în care lucrează angajați având codul job-ului "CLERK".

```
SELECT deptno "Cod departament"
```

```
FROM EMP
```

```
WHERE UPPER(job) = 'CLERK'
```

```
UNION
```

```
SELECT deptno
```

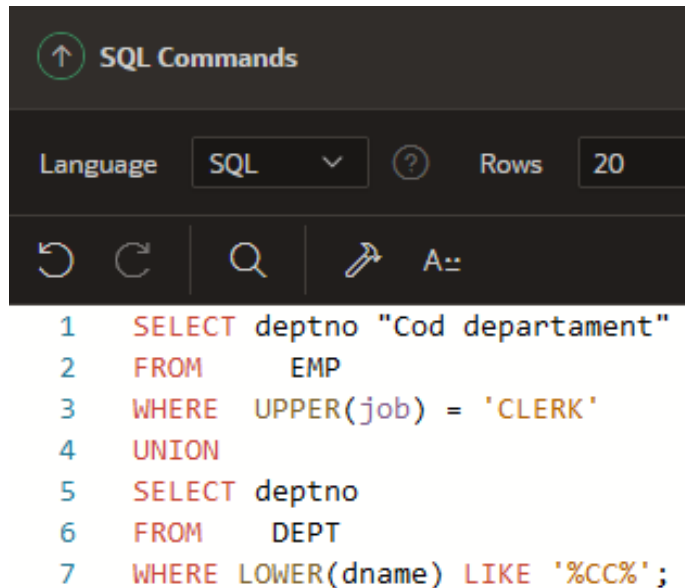
```
FROM DEPT
```

```
WHERE LOWER(dname) LIKE '%CC%';
```

# Operatorii pe mulțimi

## APLICAȚII:

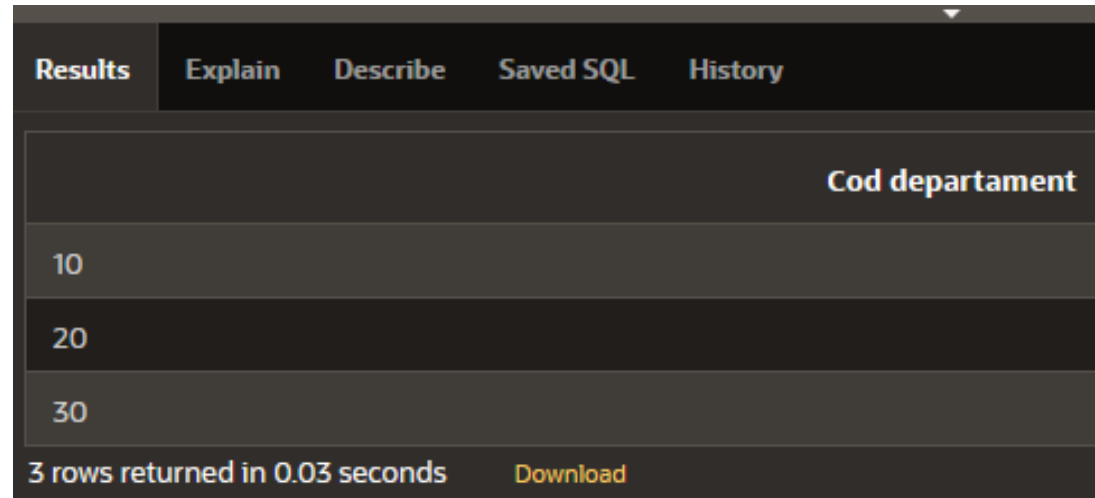
1. Să se afișeze: codurile departamentelor al căror nume conține șirul "CC" sau în care lucrează angajați având codul job-ului "CLERK".



SQL Commands

Language SQL Rows 20

```
1 SELECT deptno "Cod departament"  
2 FROM EMP  
3 WHERE UPPER(job) = 'CLERK'  
4 UNION  
5 SELECT deptno  
6 FROM DEPT  
7 WHERE LOWER(dname) LIKE '%CC%';
```



Results Explain Describe Saved SQL History

Cod departament
10
20
30

3 rows returned in 0.03 seconds [Download](#)

## Operatorii pe mulțimi

2. Să se obțină codurile departamentelor în care nu lucrează nimeni (nu este introdus nici un salariat în tabelul EMP).  
Se cer două soluții.

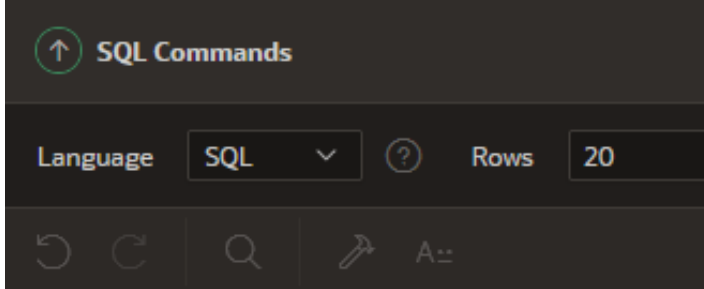
**Obs:** Operatorii pe mulțimi pot fi utilizați în subcereri.  
Coloanele care apar în clauza WHERE a interogării trebuie să corespundă, ca număr și tip de date, celor din clauza SELECT a subcererii.

# Operatorii pe mulțimi

Soluția 1

```

SELECT deptno "Cod departament"
FROM DEPT
MINUS
SELECT deptno
FROM EMP;
  
```

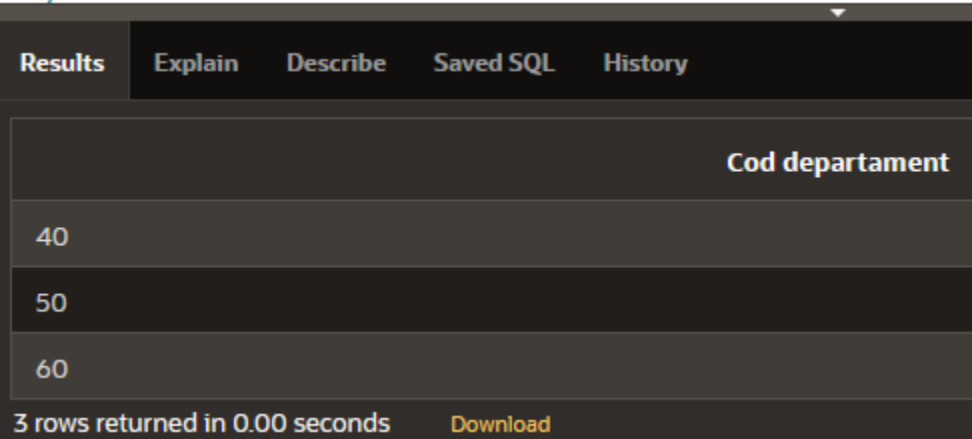


SQL Commands

Language SQL Rows 20

```

1  SELECT deptno "Cod departament"
2  FROM DEPT
3  MINUS
4  SELECT deptno
5  FROM EMP;
  
```



Results	Explain	Describe	Saved SQL	History
				Cod departament
40				
50				
60				

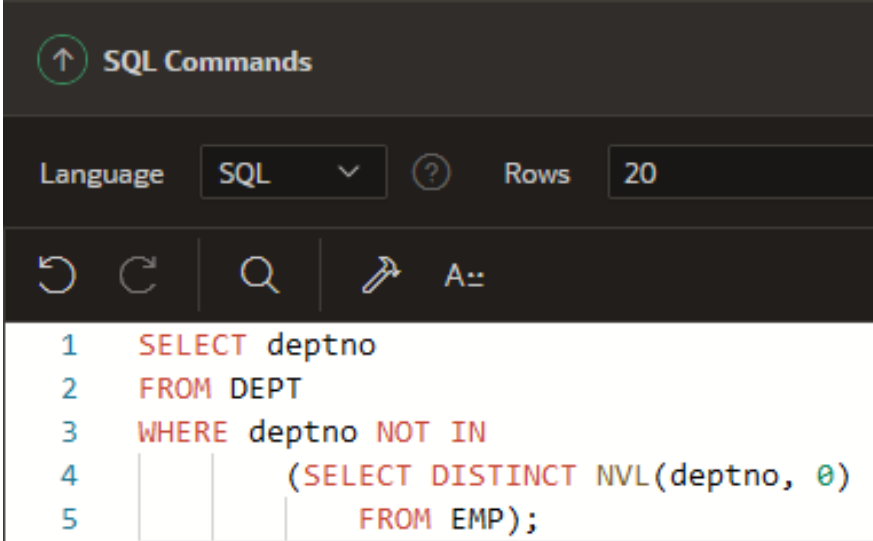
3 rows returned in 0.00 seconds [Download](#)



# Operatorii pe mulțimi

Soluția 2

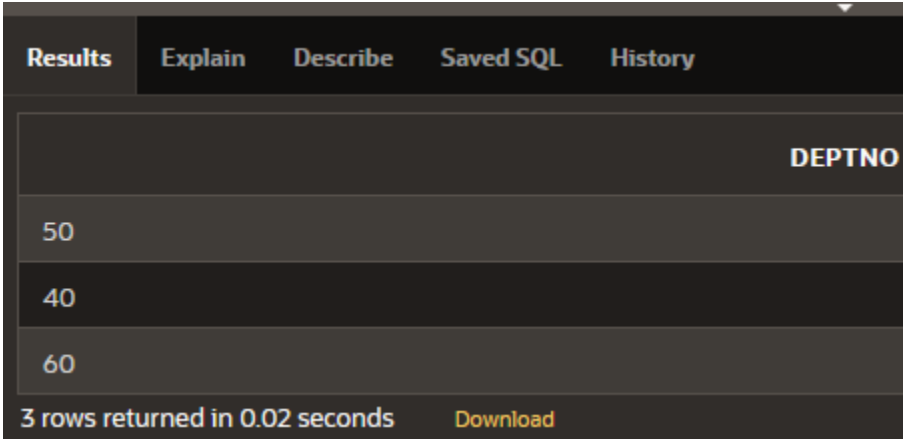
```
SELECT deptno  
FROM DEPT  
WHERE deptno NOT IN  
  ( SELECT DISTINCT NVL(deptno, 0)  
    FROM EMP );
```



SQL Commands

Language: SQL Rows: 20

```
1 SELECT deptno  
2 FROM DEPT  
3 WHERE deptno NOT IN  
4   (SELECT DISTINCT NVL(deptno, 0)  
5   FROM EMP);
```



Results Explain Describe Saved SQL History

DEPTNO
50
40
60

3 rows returned in 0.02 seconds [Download](#)

# Întrebări?