Universitatea Constantin Brâncuși din Târgu-Jiu
Automatică și Informatică Aplicată

# Baze de date

## Limbajul SQL

**Adrian Runceanu**

THE INFORMATION COMPANY

# Curs 7

# Limbajul SQL

# *Limbajul SQL*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# 6.2. 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**

# 6.2. 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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Cartesian Product and the Join Operations

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

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

Numele coloanei trebuie prefixat de numele tabelei în situaţiile când acelaşi 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) condiţia de join este omisă
2) condiţia de join nu este validă

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

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

# *Limbajul SQL*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# 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.*

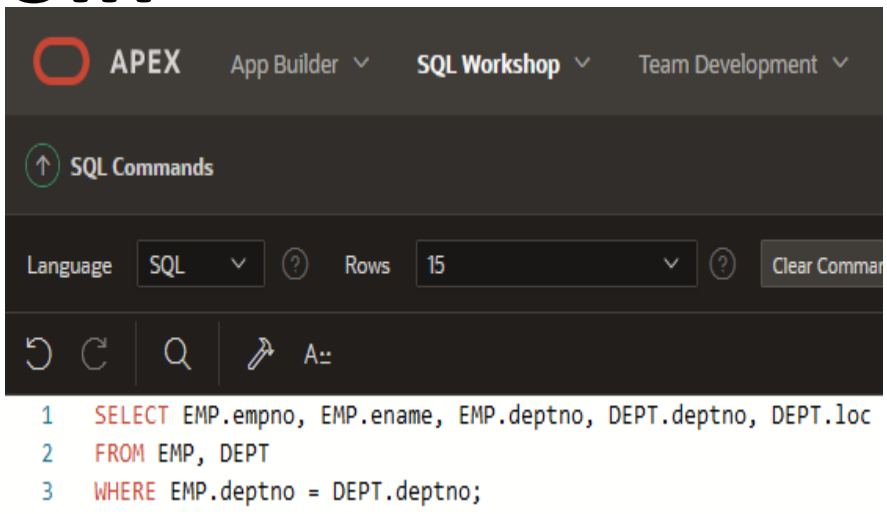
Exemplu:

```
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.*




Results | Explain | Describe | Saved SQL | History

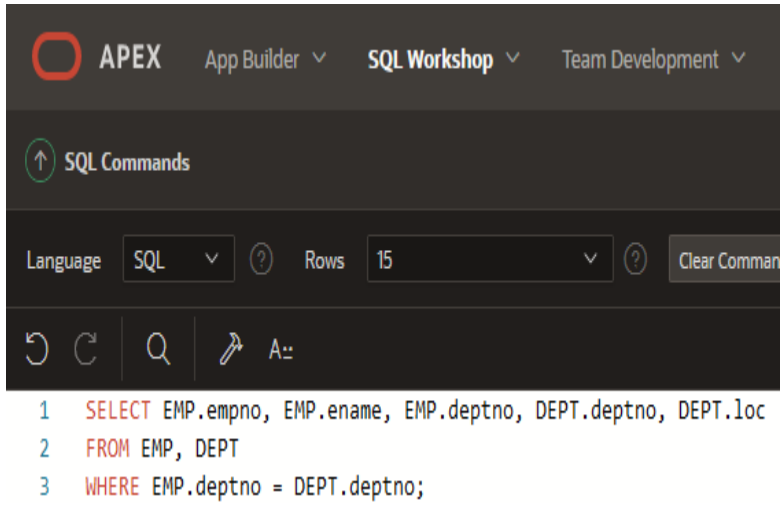
| EMPNO | ENAME | DEPTNO | DEPTNO | LOC |
|---|---|---|---|---|
| 7782 | CLARK | 10 | 10 | NEW YORK |
| 7934 | MILLER | 10 | 10 | NEW YORK |
| 7839 | KING | 10 | 10 | NEW YORK |
| 7902 | FORD | 20 | 20 | DALLAS |
| 7788 | SCOTT | 20 | 20 | DALLAS |
| 7566 | JONES | 20 | 20 | DALLAS |
| 7369 | SMITH | 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 |
| 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;




| EMPNO | ENAME | DEPTNO | DEPTNO | LOC |
|---|---|---|---|---|
| 7782 | 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 |
| 7369 | 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 |
| 7499 | ALLEN | 30 | 30 | CHICAGO |
| 7698 | BLAKE | 30 | 30 | CHICAGO |

# Equijoin

DEPT

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

Query | Count Rows | Insert Row | Load Data

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

Cheie străină (Foreign key)

Cheie primară (Primary key)

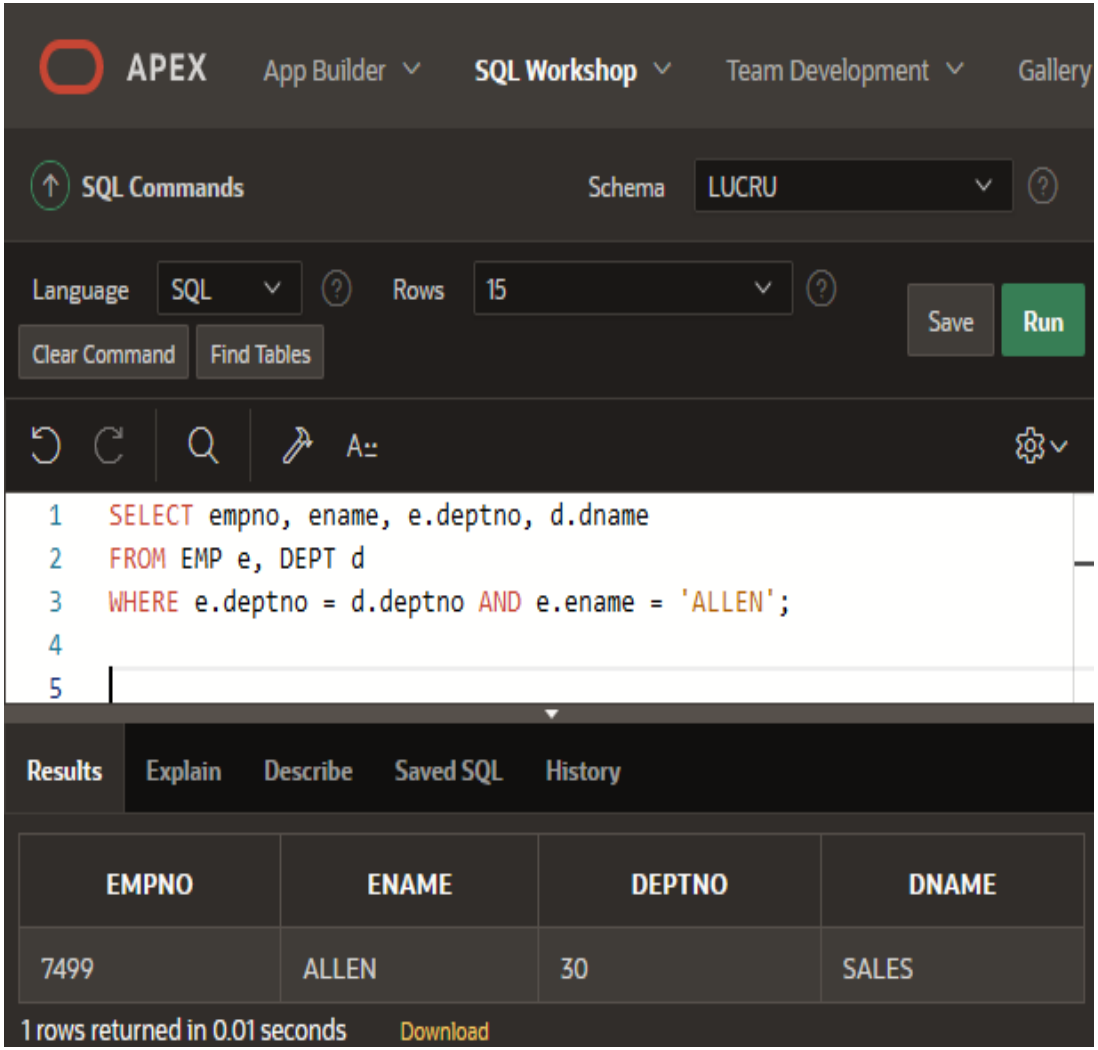
EMP

| EMPNO | ENAME | JOB | MGR | HIREDATE | SAL | COMM | DEPTNO |
|---|---|---|---|---|---|---|---|
| 7839 | KING | PRESIDENT | - | 11/17/1981 | 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 |

# Equijoin

**FOLOSIREA OPERATORULUI AND**

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



# Equijoin

## ALIAS-URI

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

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

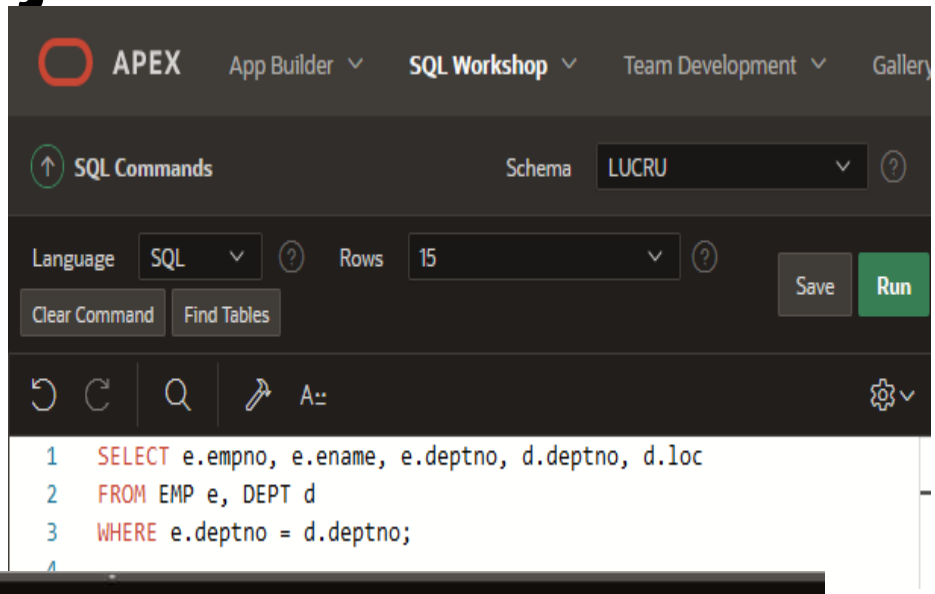
Se pot folosi alias-uri atât pentru coloane cât şi pentru tabele.

Dacă este precizat un alias pentru o tabelă în clauza **FROM**, atunci alias-ul respectiv trebuie să înlocuiască numele tabelei 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;
```




Results | Explain | Describe | Saved SQL | History

| EMPNO | ENAME | DEPTNO | DEPTNO | LOC |
|---|---|---|---|---|
| 7782 | 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 |
| 7369 | 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 |
| 7499 | ALLEN | 30 | 30 | CHICAGO |
| 7698 | BLAKE | 30 | 30 | CHICAGO |

14 rows returned in 0.01 seconds Download

# *Limbajul SQL*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# 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 situaţie se foloseşte **nonequijoin**-ul.

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

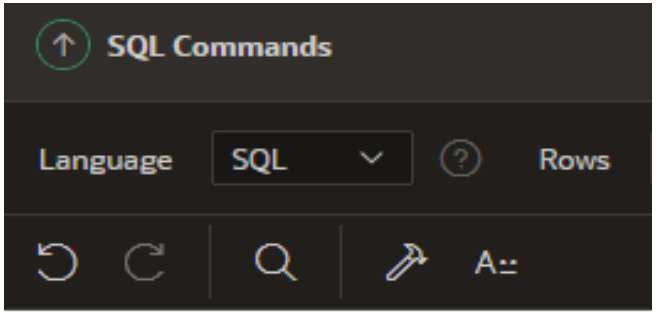
Se pot folosi operatorii:

**<=, >=, BETWEEN…AND**

# NONEQUIJOINS

Exemplu:

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




Results | Explain | Describe | Saved SQL | History

| 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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# 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 condiţiile 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 şi 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

**OBSERVAŢIE:**

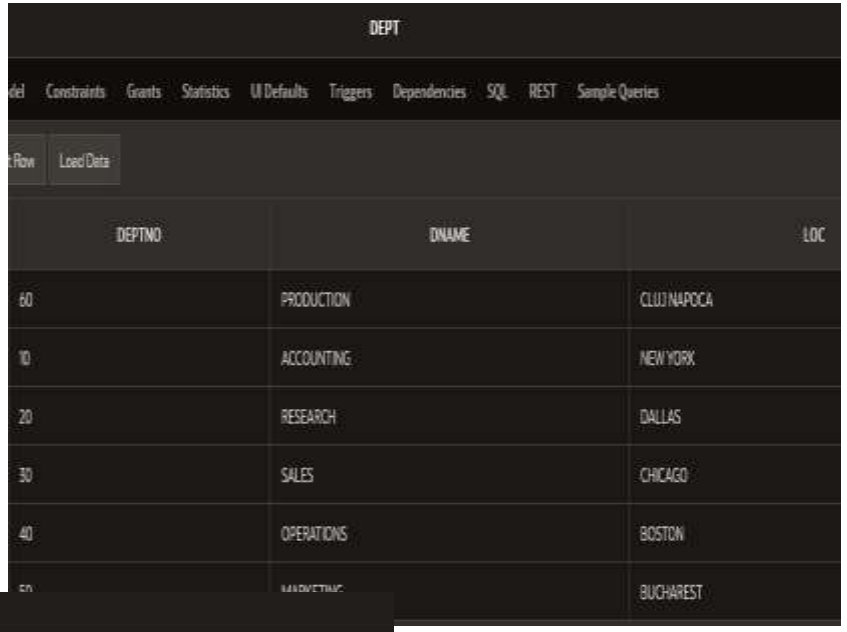
Un **outer join** nu poate folosi operatorul **IN** şi nu poate fi legat la altă condiţie prin operatorul **OR**.

```
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 angajaţi in departamentele 40, 50 şi 60.



DEPT

del Constraints Grants Statistics UI Defaults Triggers Dependencies SQL REST Sample Queries

t Row Load Data

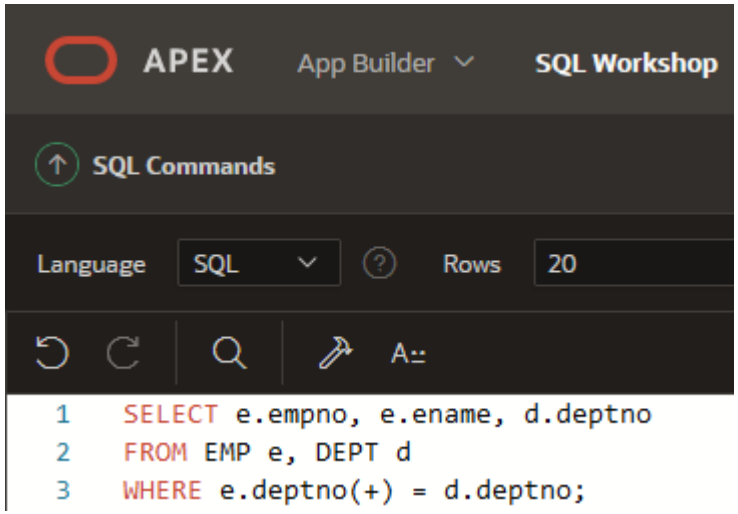
| DEPTNO | DNAME | LOC |
|---|---|---|
| 60 | PRODUCTION | CLUJ NAPOCA |
| 10 | ACCOUNTING | NEW YORK |
| 20 | RESEARCH | DALLAS |
| 30 | SALES | CHICAGO |
| 40 | OPERATIONS | BOSTON |
| 50 | MARKETING | BUCHAREST |

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;
```




Results | Explain | Describe | Saved SQL | History

| 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

**APLICAŢII:**

1) Creaţi o interogare care are ca rezultat afişarea numelui (nume) şi id-ul şi numele departamentului pentru angajaţi. Includeţi toţi angajaţii, chiar dacă nu sunt asignaţi unui departament.

2) Modificaţi interogarea din problema anterioară pentru a afişa toate id-urile departamentelor, chiar dacă nu au angajaţi asociaţi lor.

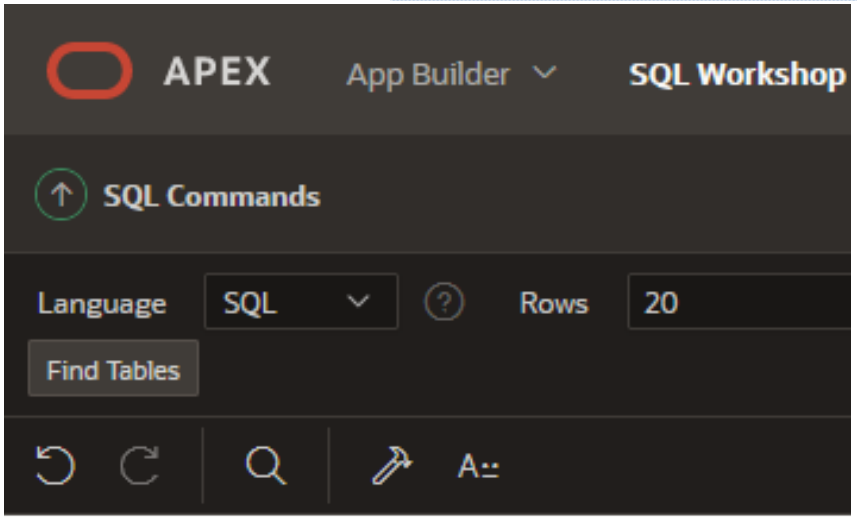
# Outer Joins

1) Creaţi o interogare care are ca rezultat afişarea numelui (nume) şi id-ul şi numele departamentului pentru angajaţi. Includeţi toţi angajaţii, chiar dacă nu sunt asignaţi unui departament.

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

# Outer Joins

1) Creaţi o interogare care are ca rezultat afişarea numelui (nume) şi id-ul şi numele departamentului pentru angajaţi. Includeţi toţi angajaţii, chiar dacă nu sunt asignaţi unui departament.




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

Results | Explain | Describe | Saved SQL | History

| 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 |

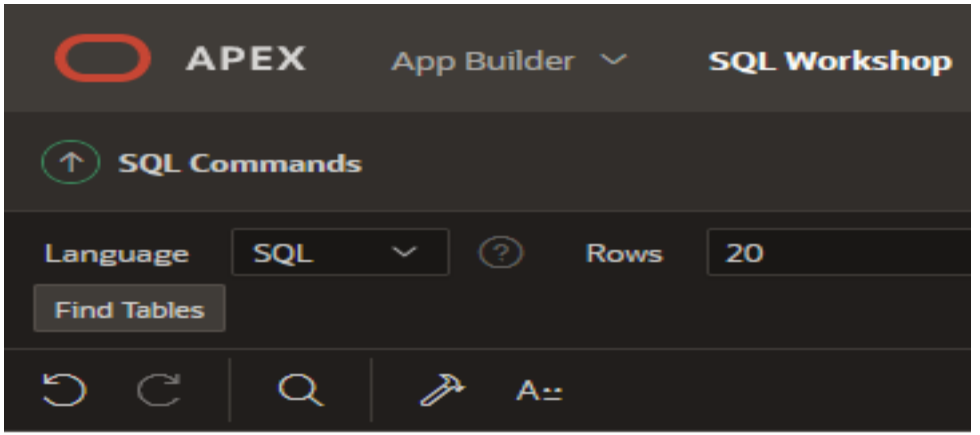
# Outer Joins

2) Modificaţi interogarea din problema anterioară pentru a afişa toate id-urile departamentelor, chiar dacă nu au angajaţi asociaţi lor.

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

# Outer Joins

2)Modificaţi interogarea din problema anterioară pentru a afişa toate id-urile departamentelor, chiar dacă nu au angajaţi asociaţi lor.




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

Results | Explain | Describe | Saved SQL | History

| 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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Self Joins

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

# Self Joins

- *Pentru a face join de la o tabelă la ea însăşi, tabelei îi sunt asociate 2 alias-uri.*

- Astfel, pentru baza de date, există în aparenţă 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

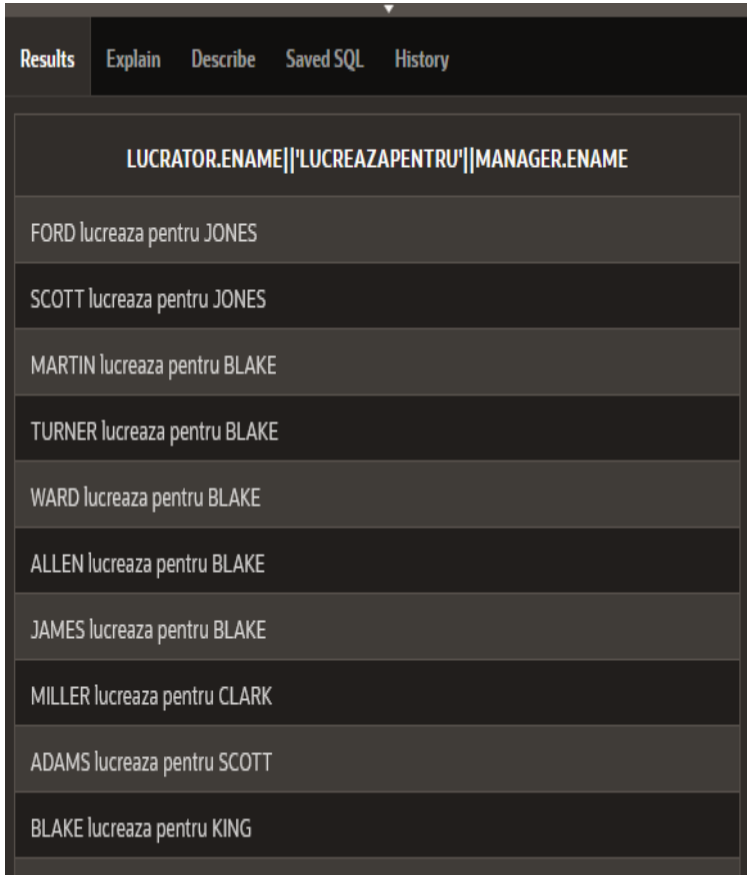
EMP

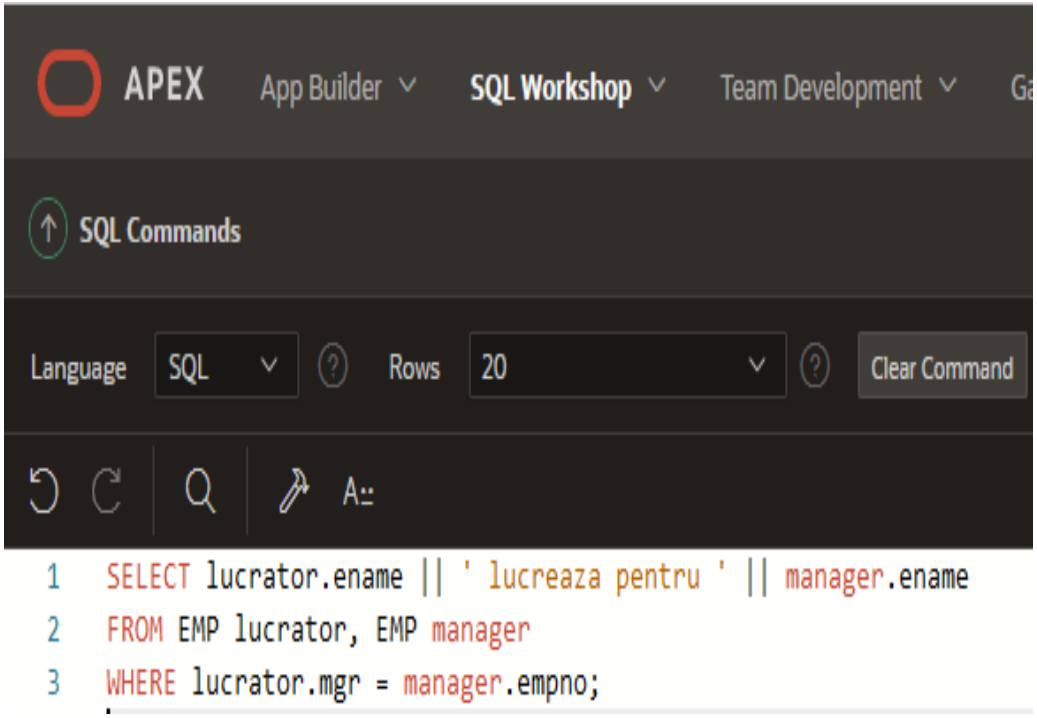
| EMPNO | ENAME |
|---|---|
| 7839 | KING |
| 7698 | BLAKE |
| 7782 | CLARK |
| 7566 | JONES |
| 7788 | SCOTT |
| 7902 | FORD |
| 7369 | SMITH |
| 7499 | ALLEN |
| 7521 | WARD |
| 7654 | MARTIN |
| 7844 | TURNER |
| 7876 | ADAMS |
| 7900 | JAMES |
| 7934 | MILLER |

# Self Joins

**Exemplu:**

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



# Self Joins

**APLICAŢII**

1) Afişaţi numele şi numărul pentru fiecare angajat împreună cu numele şi numărul managerului. Denumiţi coloanele: Angajat, Ang#, Manager şi Mgr#.

2) Modificaţi interogarea 1 pentru a afişa toţi angajaţii şi managerii lor chiar dacă un angajat nu are un manager. Ordonaţi lista obţinută alfabetic, după numele angajaţilor.

# Self Joins

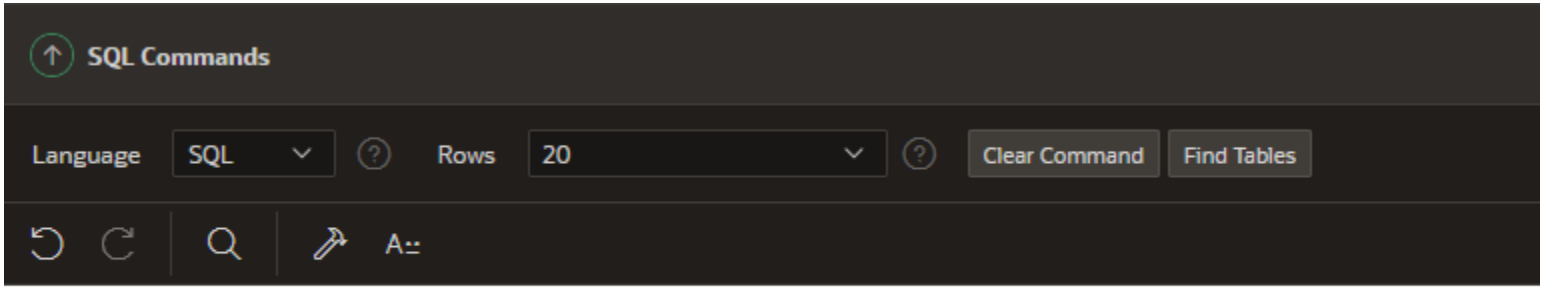
1) Afişaţi numele şi numărul pentru fiecare angajat împreună cu numele şi numărul managerului.
Denumiţi coloanele: Angajat, Ang#, Manager şi Mgr#.

```
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) Afişaţi numele şi numărul pentru fiecare angajat împreună cu numele şi numărul managerului.

Denumiţi coloanele: Angajat, Ang#, Manager şi Mgr#.




```
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;
```

Results | Explain | Describe | Saved SQL | History

| 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) Modificaţi interogarea 1 pentru a afişa toţi angajaţii şi managerii lor chiar dacă un angajat nu are un manager.

Ordonaţi lista obţinută alfabetic, după numele angajaţilor.

```
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

```
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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Cross Joins and Natural Joins

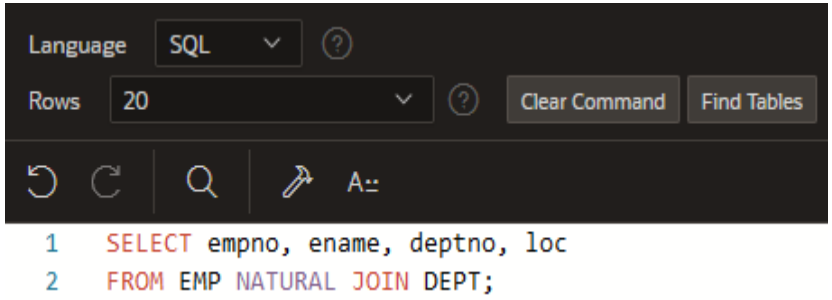
## NATURAL JOIN

- Este un ANSI/ISO SQL:1999 join echivalent cu equijoin-ul.
- *Se face pe baza tuturor coloanelor care au acelaşi nume din 2 tabele; coloanele respective trebuie să aibă acelaşi 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;




Results | Explain | Describe | Saved SQL | History

| | | | |
|---|---|---|---|
| 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

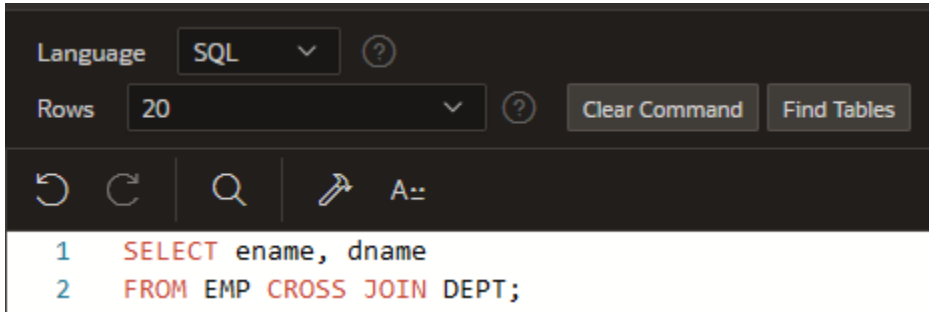
# Cross Joins and Natural Joins

## CROSS JOIN

Realizează produsul cartezian pentru două tabele.

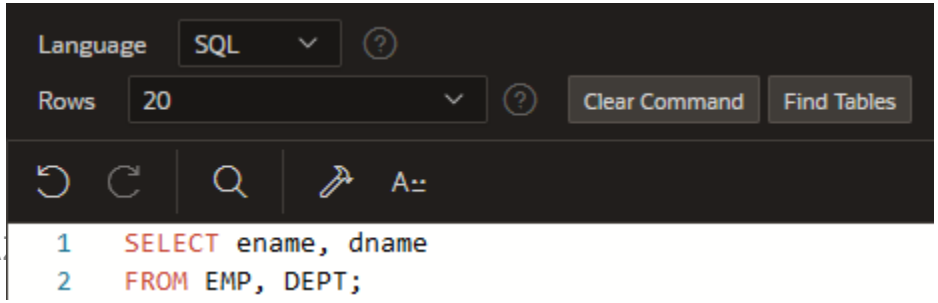
**Exemplu:**

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




este echivalentă cu instrucţiunea:

SELECT ename, dname
FROM EMP, DEPT;



# Cross Joins and Natural Joins

## APLICAŢII

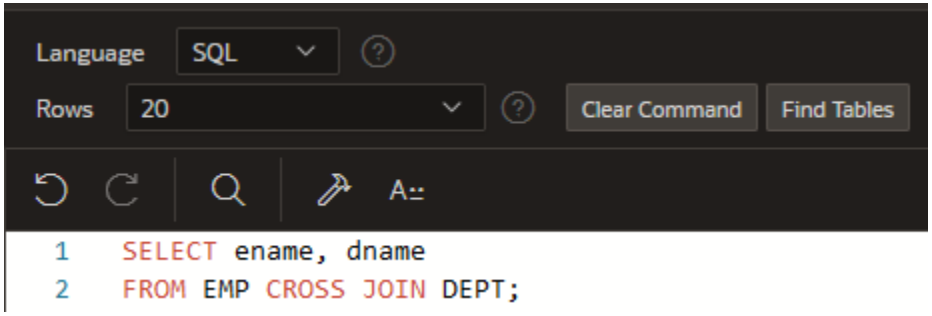
1) Creaţi un cross-join care afişează numele şi denumirea departmentului din tabelele **EMP** şi **DEPT**.

2) Creaţi o interogare care foloseşte un natural join pentru a pune în legătură tabelele **DEPT** şi **EMP**. Afişaţi id-ul, denumirea departamentului şi oraşul.

# Cross Joins and Natural Joins

1) Creaţi un cross-join care afişează numele şi denumirea departmentului din tabelele **EMP** şi **DEPT**.

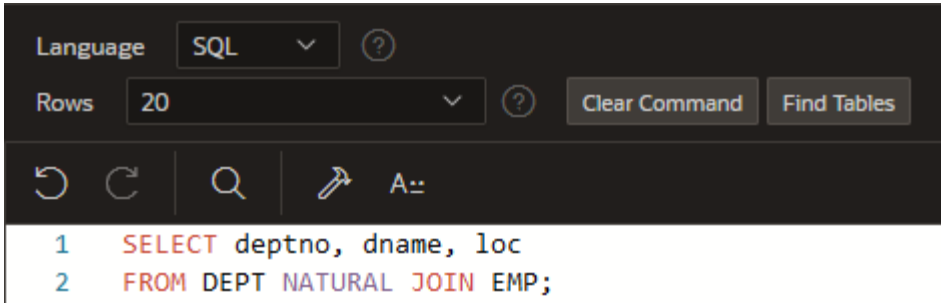
**SELECT** ename, dname
**FROM** EMP **CROSS JOIN** DEPT;



# Cross Joins and Natural Joins

2) Creaţi o interogare care foloseşte un natural join pentru a pune în legătură tabelele **DEPT** şi **EMP**. Afişaţi id-ul şi denumirea departamentului, oraşul.

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



# *Limbajul SQL*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Join Clauses

În această parte vom studia:

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

## 1. Clauza USING

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

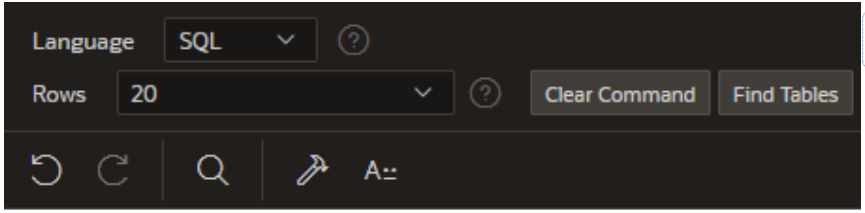
Pentru a evita această situaţie, clauza **JOIN** *se înlocuieşte* cu clauza **USING**.

# 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 instrucţiunea SELECT.

# Join Clauses




```
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);

Results | Explain | Describe | Saved SQL | History

| 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 folosiţi operatorii*: **<, >** sau BETWEEN, atunci nu putem folosi clauza USING.

În această situaţie se foloseşte clauza **ON**.

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

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

# Join Clauses

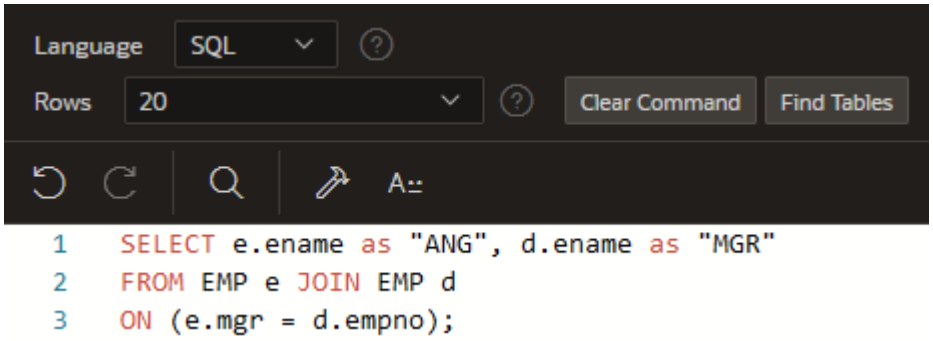
**Exemple:**

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 angajaţi care sunt şi manageri.

# Join Clauses

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




Results | Explain | Describe | Saved SQL | History

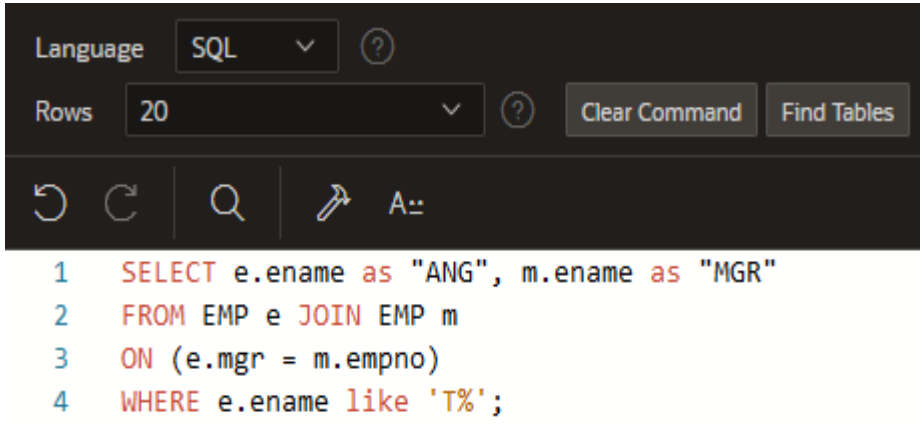
| ANG | MGR |
|---|---|
| FORD | JONES |
| SCOTT | JONES |
| MARTIN | BLAKE |
| TURNER | BLAKE |
| WARD | BLAKE |
| ALLEN | BLAKE |
| JAMES | BLAKE |
| MILLER | CLARK |
| ADAMS | SCOTT |
| BLAKE | KING |
| CLARK | KING |
| JONES | KING |
| SMITH | FORD |

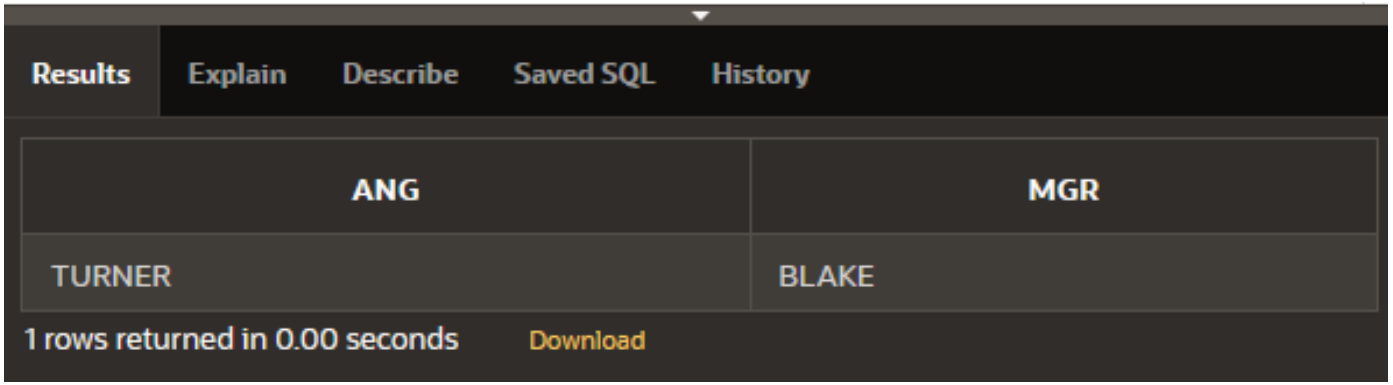
13 rows returned in 0.01 seconds Download

# 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%';
```



# *Limbajul SQL*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# 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)
```

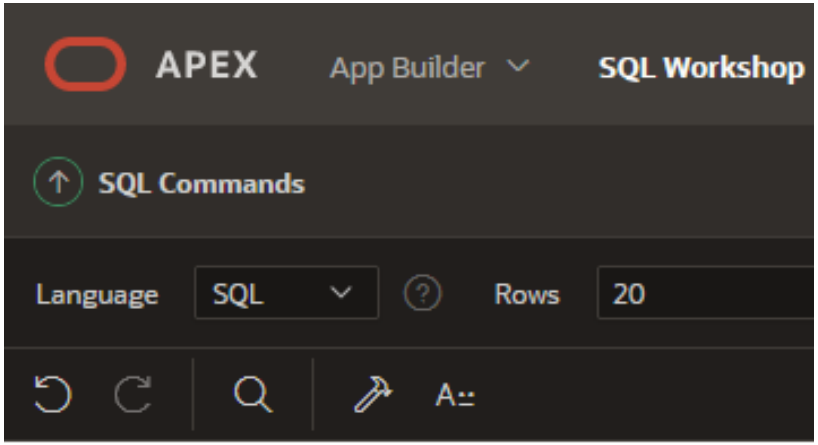
# Join Clauses

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

| Join-uri proprietate ORACLE | Join-uri ANSI/ISO SQL 1999 |
|---|---|
| **Produs cartezian** | Cross Join |
| **Equijoin** | 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) |
| **Non-equijoin** | Clauza ON |

# Join Clauses

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




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

Results | Explain | Describe | Saved SQL | History

| 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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Inner vs. Outer Joins

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

Atunci când *un join returnează atât liniile care se potrivesc cât şi cele care nu se potrivesc*, acesta se numeşte **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

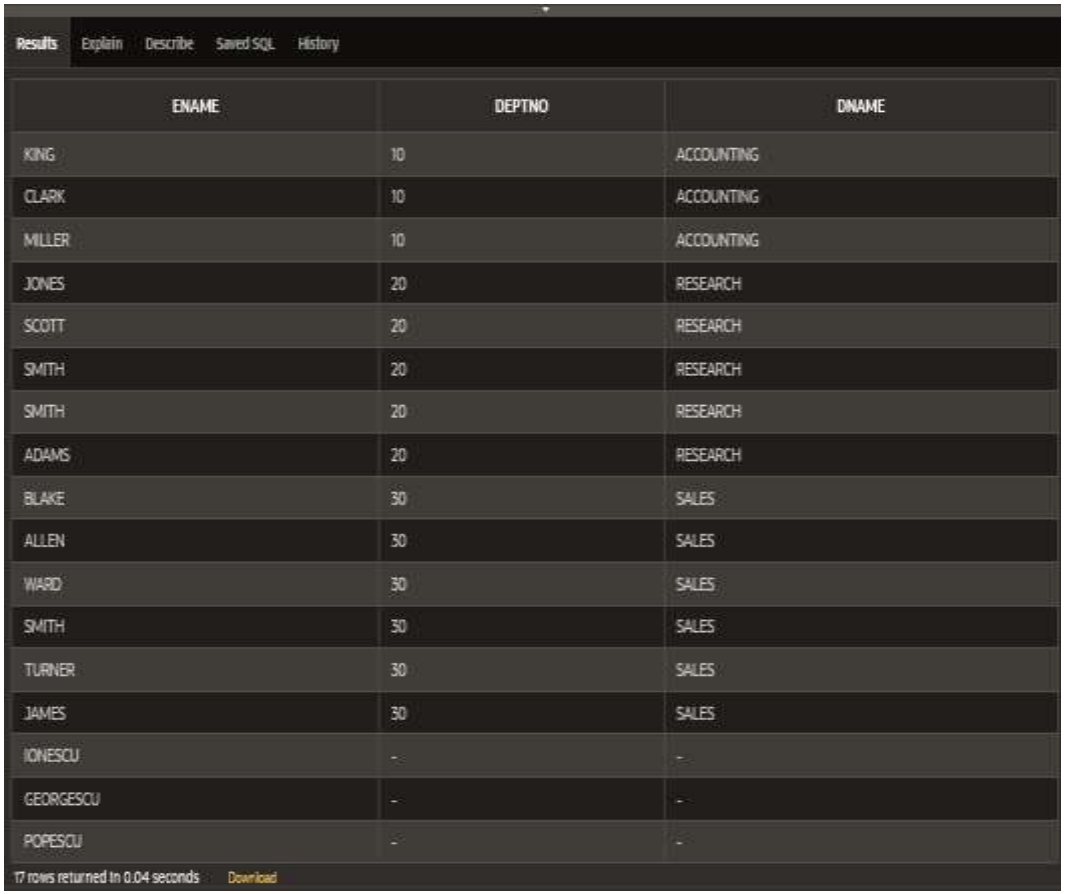
## 1. LEFT OUTER JOIN

Sunt afişaţi şi acei angajaţi 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

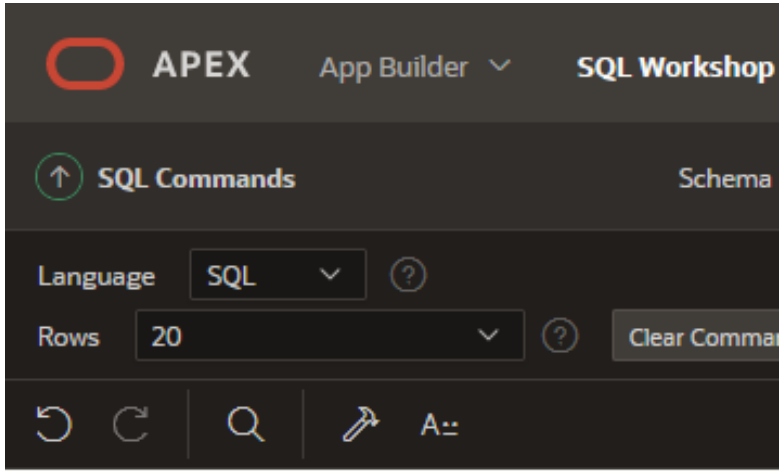
## 1. LEFT OUTER JOIN



Results | Explain | Describe | Saved SQL | History

| 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 afişate şi acele DEPT care nu au angajaţi.

```
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 afişate şi acele DEPT care nu au angajaţi.

```
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 afişate şi acele DEPT care nu au angajaţi.

Results | Explain | Describe | Saved SQL | History

| 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 |

SQL Commands

Language SQL

Rows 20

Clear Co

```
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 şi cele care nu se potrivesc din ambele tabele.

Spre deosebire de **outer join-ul** proprietatea Oracle, care nu permitea folosirea operatorului **(+)** în ambele părţi 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 şi cele care nu se potrivesc din ambele tabele.

Results | Explain | Describe | Saved SQL | History

| 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 |
| - | - | FINANCIAR |
| - | - | Support |
| - | - | PRODUCTION |

20 rows returned in 0.09 seconds Download

SQL Commands Schema

Language SQL

Rows 20 Clear Commar

```
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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Inner vs. Outer Joins

**APLICAŢII**

1) Afişaţi numele şi denumirea departamentului pentru toţi angajaţii, inclusiv pentru cei care nu sunt desemnaţi la nici un departament.
2) Afişaţi numele şi denumirea departamentului pentru toţi angajaţii, inclusiv acele departamente care nu au nici un angajat asociat.
3) Afişaţi numele şi denumirea departamentului pentru toţi EMP, inclusiv acele departamente care nu au nici un angajat asociat şi acei angajaţi care nu sunt desemnaţi nici unui departament.

# Inner vs. Outer Joins

1) Afişaţi numele şi denumirea departamentului pentru toţi angajaţii, inclusiv pentru cei care nu sunt desemnaţi 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

SQL Commands

Language SQL

Rows 20

```
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) Afişaţi numele şi denumirea departamentului pentru toţi angajaţii, 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

```
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) Afişaţi numele şi denumirea departamentului pentru toţi EMP, inclusiv acele departamente care nu au nici un angajat asociat şi acei angajaţi care nu sunt desemnaţi 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

```
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 APLICAŢII:**

1. Afişaţi numele şi numărul pentru angajaţi împreună cu numele şi numărul managerului. Denumiţi coloanele astfel: Angajat, Ang#, Manager, şi Mgr#.

2. Modificaţi rezultatul de la problema 1 astfel încât să fie afişaţi toţi angajaţii, inclusiv aceia care nu au manager. Ordonaţi rezultatele după numărul angajatului.

# Inner vs. Outer Joins

1. Afişaţi numele şi numărul pentru angajaţi împreună cu numele şi numărul managerului. Denumiţi coloanele astfel: Angajat, Ang#, Manager, şi 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

```
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 );
```

Results | Explain | Describe | Saved SQL | History

| 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. Modificaţi rezultatul de la problema 1 astfel încât să fie afişaţi toţi angajaţii, inclusiv aceia care nu au manager. Ordonaţi 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;
```

Results | Explain | Describe | Saved SQL | History

| 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*

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

### 6.2.1. JOIN-urile proprietatea ORACLE

#### 6.2.1.1. Cartesian Product

#### 6.2.1.2. Equijoin

#### 6.2.1.3. Non-equijoin

#### 6.2.1.4. Outer join

#### 6.2.1.5. Self join

### 6.2.2. JOIN-urile ANSI/ISO SQL99

#### 6.2.2.1. Cross join

#### 6.2.2.2. Natural join

#### 6.2.2.3. Using clause

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

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

### 6.2.3. Operatorii pe mulţimi

# Operatorii pe mulţimi

*Operatorii pe mulţimi combină rezultatele obţinute din două sau mai multe interogări.*

Cererile care conţin operatori pe mulţimi se numesc ***cereri compuse***.

Există patru operatori pe mulţimi:

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

# Operatorii pe mulţimi

- Toţi operatorii pe mulţimi au aceeaşi precedenţă.
- Dacă o instrucţiune ***SQL*** conţine mai mulţi operatori pe mulţimi, *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 mulţimi

- *Operatorul **UNION*** returnează toate liniile selectate de două cereri, eliminând duplicatele.

- Acest operator nu ignoră valorile *null* şi are precedenţă mai mică decât operatorul ***IN***.

# Operatorii pe mulţimi

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

# Operatorii pe mulţimi

- Operatorul ***INTERSECT*** returnează toate liniile comune cererilor asupra cărora se aplică.

- Acest operator nu ignoră valorile *null*.

# Operatorii pe mulţimi

- Operatorul ***MINUS*** determină liniile returnate de prima cerere care nu apar în rezultatul celei de-a doua cereri.

- Pentru ca operatorul *MINUS* să funcţioneze, este necesar ca toate coloanele din clauza *WHERE* să se afle şi în clauza *SELECT*.

# Operatorii pe mulţimi

***Observaţii:***

1. În mod implicit, pentru toţi operatorii cu excepţia lui *UNION ALL*, rezultatul este ordonat crescător după valorile primei coloane din clauza *SELECT*.

2. Pentru o cerere care utilizează operatori pe mulţimi, cu excepţia lui *UNION ALL*, *server*-ul ***Oracle*** elimină liniile duplicat.

# Operatorii pe mulţimi

1. În instrucţiunile *SELECT* asupra cărora se aplică operatori pe mulţimi, coloanele selectate trebuie să corespundă ca număr şi 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 mulţimi

**APLICAŢII:**

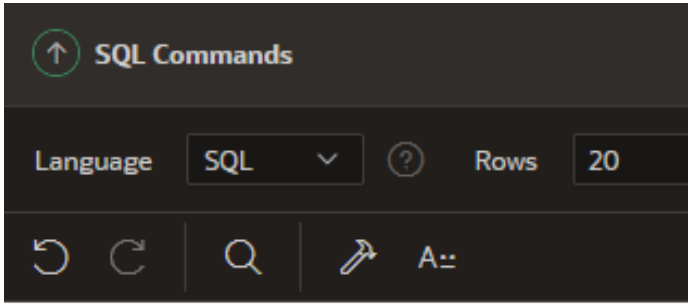
1. Să se afişeze: codurile departamentelor al căror nume conţine şirul “CC” sau în care lucrează angajaţi având codul job-ului “CLERK”.

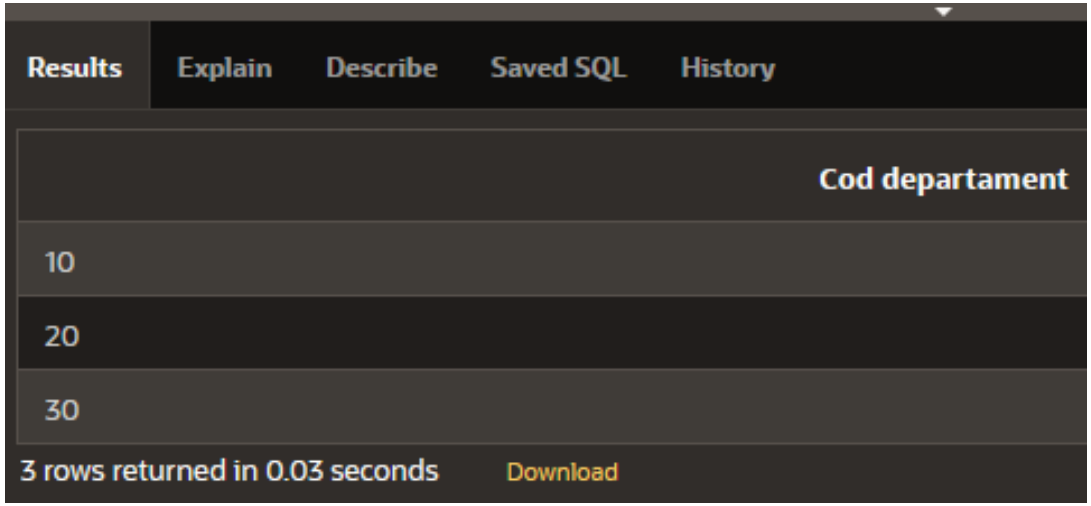
```
SELECT deptno "Cod departament"
FROM    EMP
WHERE  UPPER(job) = 'CLERK'
UNION
SELECT deptno
FROM    DEPT
WHERE LOWER(dname) LIKE '%CC%';
```

# Operatorii pe mulţimi

## APLICAŢII:

1. Să se afişeze: codurile departamentelor al căror nume conţine şirul “CC” sau în care lucrează angajaţi având codul job-ului “CLERK”.



# Operatorii pe mulţimi

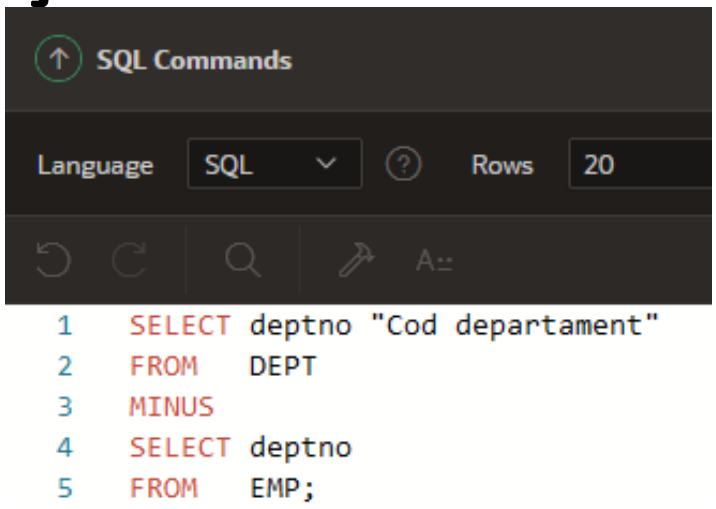
2. Să se obţină codurile departamentelor în care nu lucrează nimeni (nu este introdus nici un salariat în tabelul EMP).
   Se cer două soluţii.

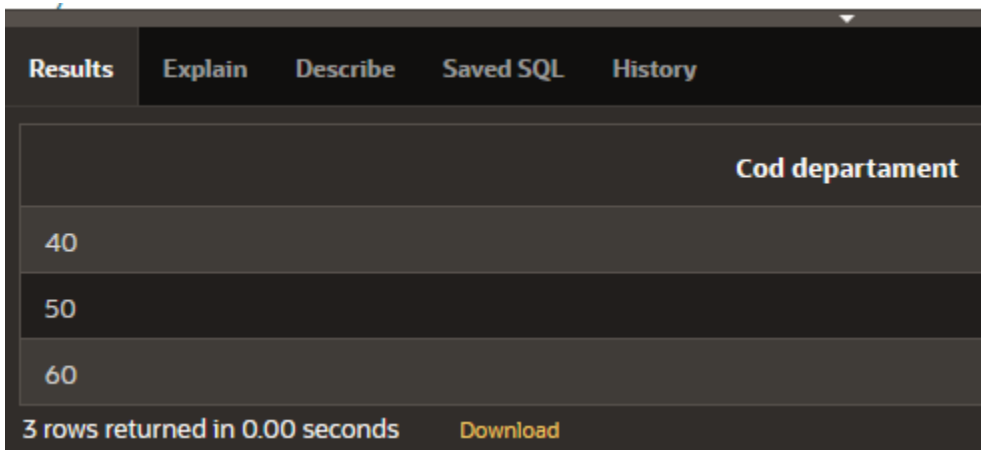
**Obs:** Operatorii pe mulţimi pot fi utilizaţi în subcereri.
Coloanele care apar în clauza WHERE a interogării trebuie să corespundă, ca număr şi tip de date, celor din clauza SELECT a subcererii.

# Operatorii pe mulţimi

Soluţia 1

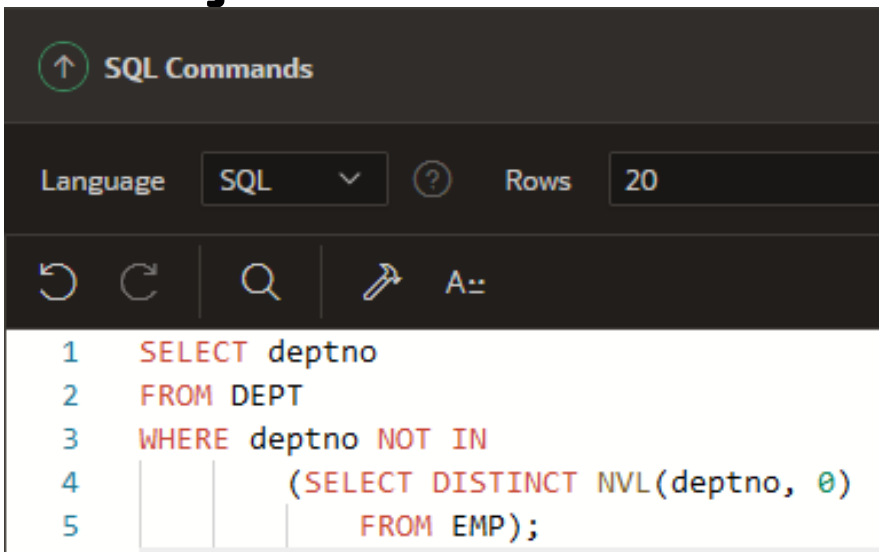
**SELECT** deptno "Cod departament"
**FROM** DEPT
**MINUS**
**SELECT** deptno
**FROM** EMP;



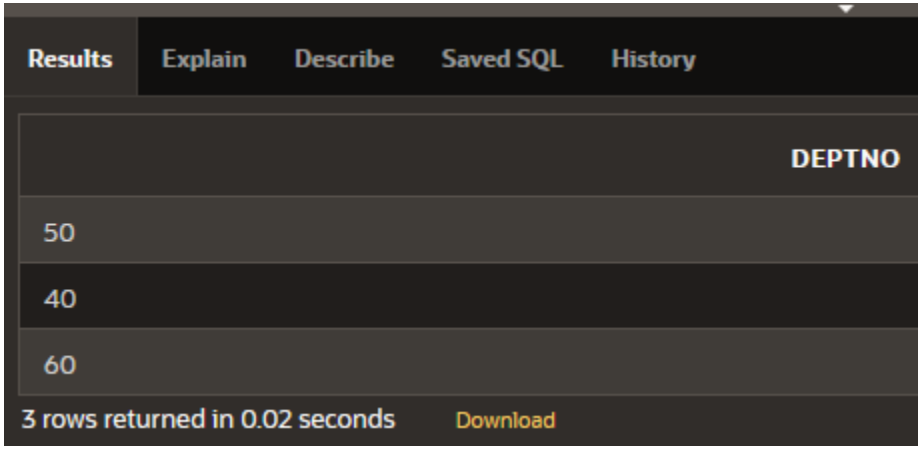



# Operatorii pe mulţimi

Soluţia 2




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



# Întrebări?