Universitatea Constantin Brâncuşi din Târgu Jiu
Facultatea: Inginerie
Program de conversie profesională a cadrelor didactice din învățământul preuniversitar: Informatică,Tehnologia Informației și a Comunicațiilor

Laborator 7

Cereri din mai multe tabele (JOIN-uri)

SQL Join este o instrucțiune care combină date din mai multe tabele. Fiecare comandă **SELECT** poate folosi una sau mai multe metode de **JOIN**. Standardul ANSI-SQL definește 5 tipuri de JOIN:

- INNER
- LEFT OUTER JOIN
- RIGHT OUTER JOIN
- FULL OUTER JOIN
- CROSS JOIN

Un caz special este **SELF JOIN** care face JOIN **pe o tabelă cu ea însăși**.

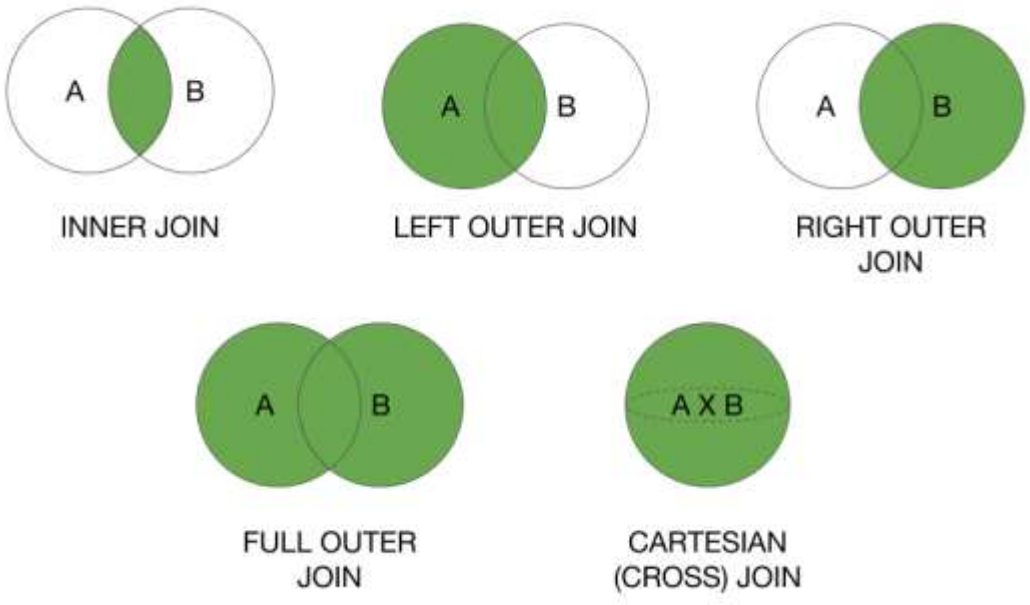


Image source: http://kirillpavlov.com/blog/2016/04/23/beyond-traditional-join-with-apache-spark

JOIN

Sintaxa pentru JOIN simplu este următoarea:

- **DISTINCT** – returnează doar o înregistrare în cazul în care comanda găsește liniile duplicate;
- **ALL** – returnează toate înregistrările simple și duplicate;
- **schema** – reprezintă schema de identificare a tabelei (view-ului);
- **column_alias** – este un nume alocat unei coloane(expresii) care va fi folosit în formatarea coloanei (numele care apare în antetul listei);
- **WHERE condition** – reprezintă o clauză (înlănțuire de condiții) care trebuie să fie îndeplinită în criteriul de selecție a înregistrărilor;
- **ORDER BY {expresion|position} [ASC|DESC]** – ordonează înregistrările selectate după coloanele din expresie sau în ordinea coloanelor (selectate în clauza SELECT) specificate prin poziție·

```
SELECT [DISTINCT|ALL] [[TABLE|table_alias].]{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias],
        [schema.]table2 [table2_alias]
WHERE
        {table1|table1_alias.}COLUMN = {tabel2|table2_alias}.COLUMN
[ORDER BY {expresion|POSITION} [ASC|DESC]]
```

CROSS JOIN (Produsul Cartezian)

- Produsul cartezian (Cross Join) a două tabele se obține prin concatenarea fiecărei linii dintr-o tabelă cu fiecare linie din cealaltă tabelă;
- Rezultatul este un număr de linii egal cu produsul cartezian dintre numărul de linii din fiecare tabela din clauza FROM. Această situație este mai puțin practică și se întâlnește, de regulă, când sunt puse greșit condițiile.
- Sintaxa CROSS JOIN este:

```
SELECT [DISTINCT|ALL] [[TABLE|table_alias].]{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias],
        [schema.]table2 [table2_alias]
[other clauses]
```

Alta sintaxa CROSS JOIN (recomandată pentru lucru cu toate bazele de date relaționale):

```
SELECT [DISTINCT|ALL] [[TABLE|table_alias].]{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias] CROSS JOIN
        [schema.]table2 [table2_alias]
[other clauses]
```

INNER JOIN

- INNER JOIN este cel mai folosit tip de Join;
- Este o operație care poate fi folosită cu siguranță într-o bază de date care impune integritatea referințelor sau în cazul în care câmpurile din constrângere nu sunt NULL;
- Inner Join are următoarele subtipuri:
- Equi-join
- Natural join
- Non Equi-join (Θ-Join)

Equi-join

- Dacă în condițiile de Join apar numai egalități, avem de-a face cu un EQUI-JOIN;
- Pentru a putea să realizăm un join pe mai multe coloane, este obligatoriu ca ele să conțină coloane de același tip cu date comune sau corelate;
- Folosiți aliasuri pentru tabele pentru a nu apărea ambiguități în momentul în care se referă coloanele pentru Sintaxa 1 și Sintaxa 2.

Sintaxa 1 pentru Equi-Join:

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias],
        [schema.]table2 [table2_alias]
WHERE
         {{table1|table1_alias}.}column_fromTable1 = {{table2|table2_alias}.}column_fromTable2
[other clauses]
```

Sintaxa 2 pentru Equi-Join (recomandată de ultimul standard SQL):

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias]
                [INNER] JOIN [schema.]table2 [table2_alias]
                ON      {{table1|table1_alias}.}column_fromTable1 =             {{table2|table2_ali
[other clauses]
```

Sintaxa 3 pentru Equi-Join (această sintaxă merge doar în Oracle):

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias]
                [INNER] JOIN [schema.]table2 [table2_alias]
                USING(COLUMN)
        [other clauses]
```

Natural Join

- Operația de join NATURAL JOIN este un tip de Equi-Join.
- Pentru acest tip de Join nu trebuie să se mai specifice condițiile de legătură, operația de egalitatea se face implicit pe toate coloanele care au același nume in cele două tabele.
- Rezultatul concatenării celor două tabele va afișa numai o coloană din perechea pe care formează condiția de egalitate.

Dacă folosiți Natural Join NU asociați aliasuri coloanelor care intră în componența condiției. Apare următoarea eroare: ORA-25155: column used in NATURAL join cannot have qualifier.

Sintaxa pentru NATURAL JOIN este următoarea:

```
SELECT [DISTINCT|ALL] [[TABLE|table_alias].]{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias]
                NATURAL JOIN [schema.]table2 [table2_alias]
[other clauses]
```

Outer Join

- Folosind Inner Join se selectează doar înregistrările care îndeplinesc condițiile din clauza WHERE.
- Apar situații când cererea trebuie să selecteze și înregistrările care nu îndeplinesc toate condițiile din clauză, în aceste cazuri se folosește OUTER JOIN (Join Extern).
- Outer Join are următoarele subtipuri:
- Left Outer Join
- Right Outer Join
- Full Outer Join

Left Outer Join

Fie două tabele A și B. Operația de Left Outer Join (A Left Outer Join B) returnează toate înregistrările din tabela A și, în cazul în care nu găsește o corespondență în tabela B va întoarce NULL, altfel va întoarce valorile selectate din tabela B.

Sintaxa Left Outer Join (merge doar în Oracle) este următoarea:

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias],
        [schema.]table2 [table2_alias]
WHERE
         {{table1|table1_alias}.}column_fromTable1 = {{table2|table2_alias}.}column_fromTable2 (+)
[other clauses]
```

Sintaxa Left Outer Join (recomandată de ultimul standard SQL, merge în toate SGBD-urile) este:

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias]
                LEFT OUTER JOIN [schema.]table2 [table2_alias]
                ON      {{table1|table1_alias}.}column_fromTable1 =             {{table2|table2_ali
[other clauses]
```

Right Outer Join

Fie două tabele A și B. Operația de Right Outer Join (A Right Outer Join B) returnează toate înregistrările din tabela B și, în cazul în care nu găsește o corespondență în tabela A va întoarce NULL, altfel va întoarce valorile selectate din tabela A.

Sintaxa Right Outer Join (merge doar în Oracle):

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias],
        [schema.]table2 [table2_alias]
WHERE
         {{table1|table1_alias}.}column_fromTable1(+) = {{table2|table2_alias}.}column_fromTable2
[other clauses]
```

Sintaxa Right Outer Join (recomandată de ultimul standard SQL, merge în toate SGBD-urile) este:

```
SELECT [DISTINCT|ALL] {{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
        [schema.]table1 [table1_alias]
                RIGHT OUTER JOIN [schema.]table2 [table2_alias]
                ON      {{table1|table1_alias}.}column_fromTable1 =             {{table2|table2_ali
[other clauses]
```

Full Outer Join

Fie două tabele A și B. Operația de Full Outer Join (A Full Outer Join B) returnează toate înregistrările din tabela A și B și, în cazul în care nu găsește o corespondență în tabela A sau B va întoarce NULL, altfel va întoarce valorile selectate din tabela A sau B.

Sintaxa Full Outer Join (recomandată de ultimul standard SQL, merge în toate SGBD-urile) este:

```
SELECT [DISTINCT|ALL]
{{TABLE|table_alias}.}{COLUMN|expression} [column_alias]
FROM
[schema.]table1 [table1_alias]
FULL [OUTER] JOIN [schema.]table2 [table2_alias]
ON
{{table1|table1_alias}.}column_fromTable1 =
{{table2|table2_alias}.}column_fromTable2
[other clauses]
```

Observații:

- Totdeauna semnul (+) se pune în dreptul tabelei deficitare de informație;
- Left Outer Join este invers simetrica cu Right Outer Join, adică, pentru tabelele A și B, A left outer join B = B right outer join A;
- Pentru a corela datele se pot folosi toți operatorii comparație și de negație, atât logici cât și SQL;
- Se pot combina operațiile de Join între ele.

Join Vertical

- Join-ul vertical este folosit pentru concatenarea rezultatelor mai multor comenzi SELECT și folosește operatorii UNION [ALL] (reuniune), INTERSECT (intersecția), MINUS (diferența);
- În acest caz se face Join după coloane de același tip, din acest motiv se numește Join Vertical;
- Coloanele selectate trebuie să fie de același tip (data type) când se folosesc operatorii UNION [ALL], INTERSECT, MINUS, chiar dacă au semnificații diferite;
- Folosind operatorul UNION ALL se selectează și înregistrările duplicate;
- Operatorul INTERSECT se folosește pentru a selecta înregistrările comune;

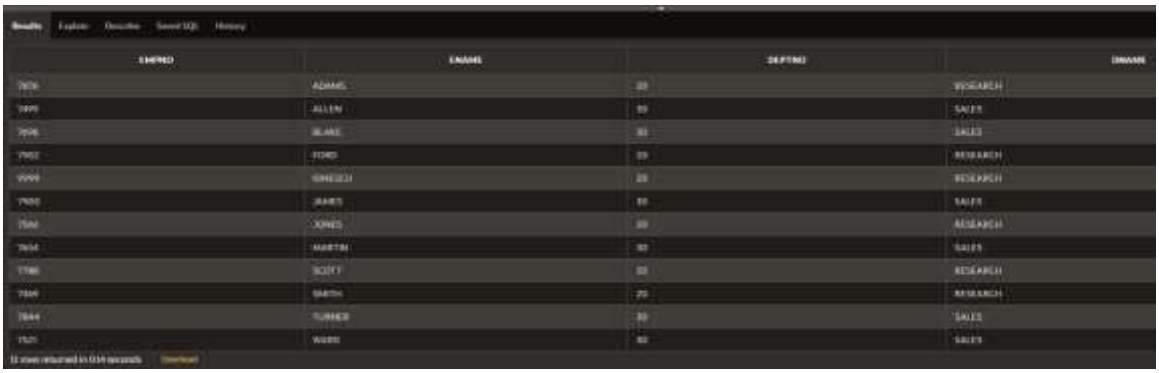
- Operatorul MINUS se folosește pentru a selecta doar înregistrările care nu se regăsesc în al doilea select.

Exercitii rezolvate:

1. Să se afişeze codul şi numele angajaţilor care lucrează în acelasi departament cu cel puţin un angajat al cărui nume conţine litera “t”. Se vor afişa, de asemenea, codul şi numele departamentului respectiv. Rezultatul va fi ordonat alfabetic după nume. Se vor da 2 soluţii pentru join (condiţie în clauza WHERE).

Solutie:

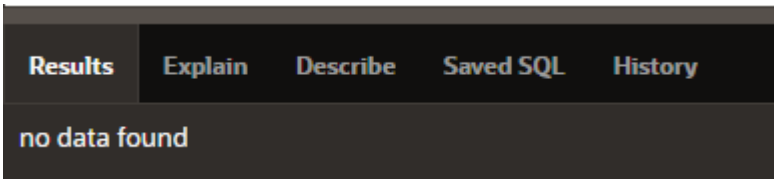
```
SELECT distinct e.empno, e.ename, e.deptno, d2.dname
FROM emp e JOIN emp d1 on (e.deptno = d1.deptno)
        JOIN dept d2 on (e.deptno = d2.deptno)
WHERE lower(d1.ename) like '%t%'
ORDER BY ename;
```



2. Sa se afiseze numele, salariul, titlul job-ului, şi oraşul în care lucrează angajatii condusi direct de KING.

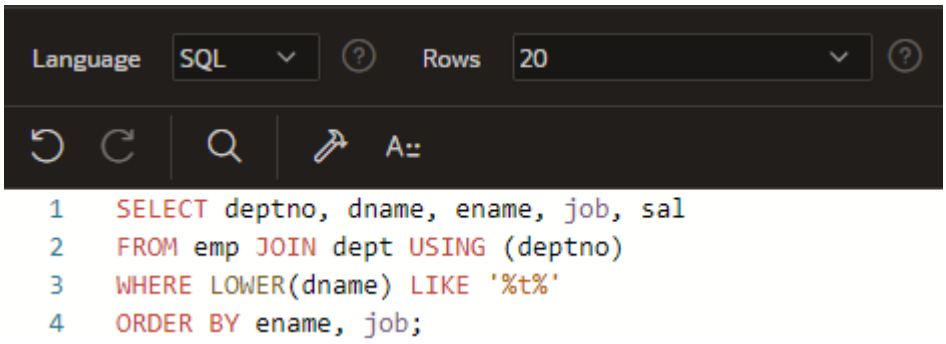
Solutie:

```
SELECT e1.empno, e1.sal, e2.ename, e2.job
FROM emp e1 JOIN emp e2 ON (e1.mgr = e2.mgr)
         JOIN dept d ON (e1.deptno = d.deptno)
WHERE LOWER(e2.ename) LIKE 'king';
```



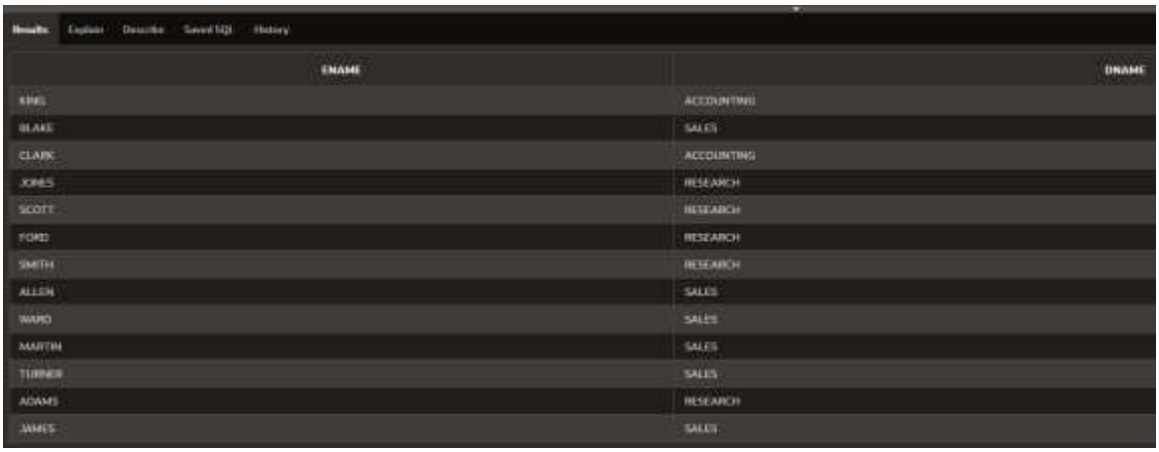
3. Afisati codul, numele si salariul tuturor angajatilor care castiga mai mult decat salariul mediu pentru job-ul corespunzător si lucreaza intr-un departament cu cel putin unul din angajatii al caror nume contine litera “t”.
Observatie: Salariul mediu pentru un job se va considera drept media aritmetică a valorilor minime şi maxime admise pentru acesta (media valorilor coloanelor min_salary şi max_salary).

```
SELECT deptno, dname, ename, job, sal
FROM emp JOIN dept USING (deptno)
WHERE LOWER(dname) LIKE '%t%'
ORDER BY ename, job;
```



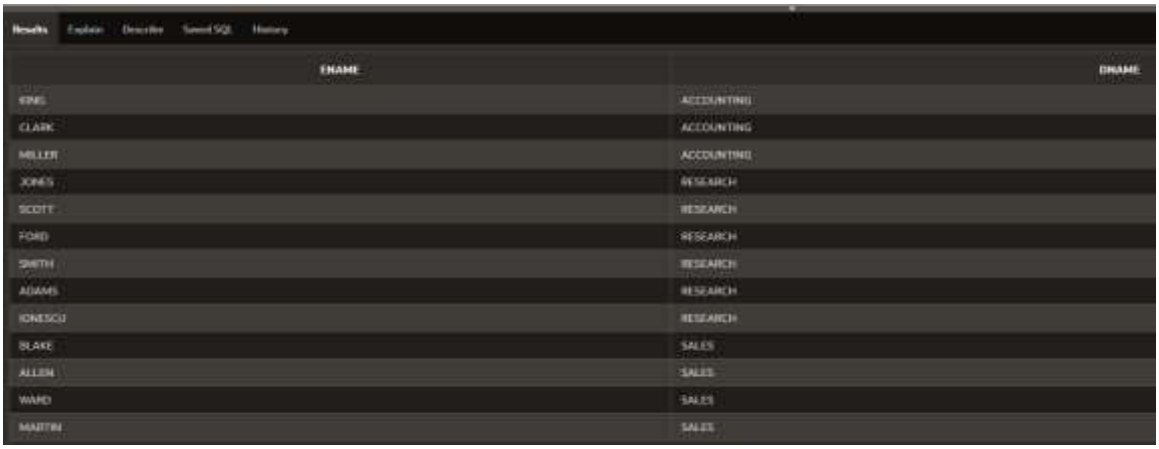
4. Să se afişeze numele salariaţilor şi numele departamentelor în care lucrează. Se vor afişa şi salariaţii care nu au asociat un departament. (**right outer join**).

```
SELECT ename, dname
FROM emp RIGHT JOIN dept USING (deptno);
```



5. Să se afişeze numele departamentelor şi numele salariaţilor care lucrează în ele. Se vor afişa şi departamentele care nu au salariaţi. (**left outer join**)

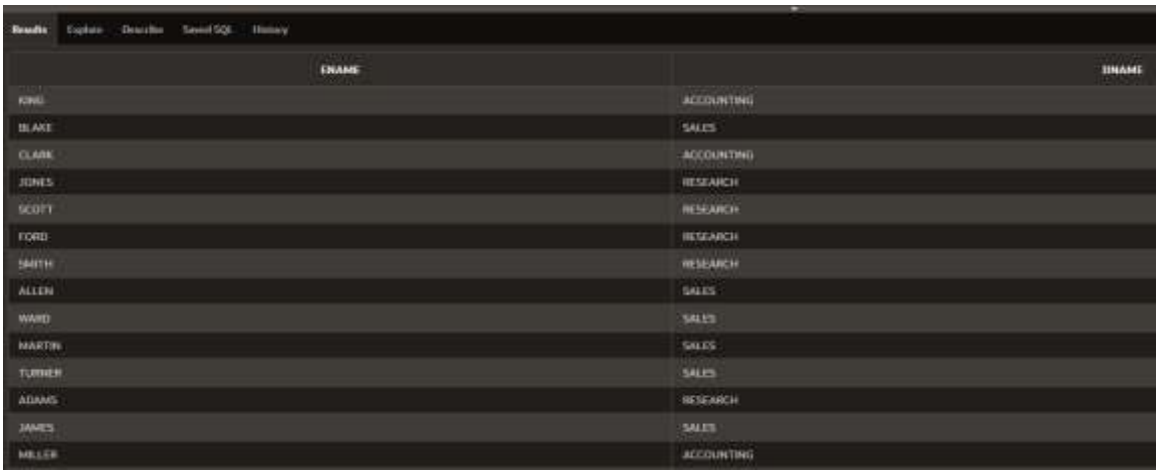
```
SELECT ename, dname
FROM emp LEFT JOIN dept USING (deptno);
```



ENAME	DNAME
KING	ACCOUNTING
CLARK	ACCOUNTING
MILLER	ACCOUNTING
JONES	RESEARCH
SCOTT	RESEARCH
FORD	RESEARCH
SMITH	RESEARCH
ADAMS	RESEARCH
IONESCU	RESEARCH
BLAKE	SALES
ALLEN	SALES
WARD	SALES
MARTIN	SALES

6. Cum se poate implementa **full outer join**?

```
SELECT ename, dname
FROM emp FULL JOIN dept USING (deptno);
```



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

7. Se cer codurile departamentelor al căror nume conţine şirul “ra” sau în care lucrează angajaţi având codul job-ului “clerk”. Cum este ordonat rezultatul?

```
select deptno
from dept
where lower(dname) like '%ra%'
union
select deptno
from emp
where lower(job)='clerk';
```

Sirul este ordonat.

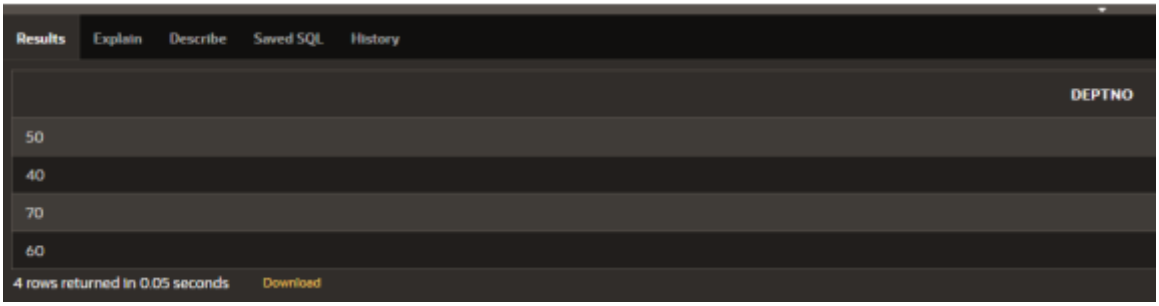
8. Ce se întâmplă dacă înlocuim UNION cu UNION ALL în exemplul anterior?

```
select deptno
from dept
where lower(dname) like '%ra%'
union ALL
select deptno
from emp
where lower(job)='clerk';
```

9. Sa se obtina codurile departamentelor in care nu lucreaza nimeni (nu este introdus nici un salariat in tabelul emp). Se cer două soluţii.

```
-- Solutia 1:
select deptno
from dept
minus
select deptno
from emp;

– Solutia 2 cu join:
select distinct deptno
from dept left join emp using (deptno) --linii deficitare in dept
where ename is null;
```



10. Sa se afiseze codul, numele departamentului si numarul de angajati care lucreaza in acel departament pentru:
a) departamentele in care lucreaza mai putin de 4 angajati;
b) departamentul care are numarul maxim de angajati.

```
select deptno, dname, count(empno)
from emp join dept using(deptno)
group by deptno, dname
having count(empno) < 4
UNION
select deptno, dname, count(empno)
from emp join dept using(deptno)
group by deptno, dname
having count(empno) =
                ( select MAX(count(empno))
                  from emp
                  group by deptno);
```

Probleme propuse spre rezolvare:

Se considera tabela **EMP** având următoarea structură:

Oracle APEX
WKSP_ORACLESTUDENTIAIA.EMP

Column Name	Data Type	Nullable	Default	Primary Key
EMPNO	NUMBER(4,0)	No		1
ENAME	VARCHAR2(50)	Yes		
JOB	VARCHAR2(50)	Yes		
MGR	NUMBER(4,0)	Yes		
HIREDATE	DATE	Yes		
SAL	NUMBER(7,2)	Yes		
COMM	NUMBER(7,2)	Yes		
DEPTNO	NUMBER(2,0)	Yes		

și avand următoarele valori introduse:

EMPNO	ENAME	JOB	MGR	HIREDATE	SAL	COMM	DEPTNO
7839	KING	PRESIDENT		11/17/1981	5000		10
7698	BLAKE	MANAGER	7839	5/1/1981	2850		30
7782	CLARK	MANAGER	7839	6/9/1981	2450		10
7566	JONES	MANAGER	7839	4/2/1981	2975		20
7788	SCOTT	ANALYST	7566	12/9/1982	3000		20
7902	FORD	ANALYST	7566	12/3/1981	3000		20
7369	SMITH	CLERK	7902	12/17/1980	800		20
7499	ALLEN	SALESMAN	7698	2/20/1981	1600	300	30
7521	WARD	SALESMAN	7698	2/22/1981	1250	500	30
7654	MARTIN	SALESMAN	7698	9/28/1981	1250	1400	30
7844	TURNER	SALESMAN	7698	9/8/1981	1500	0	30
7876	ADAMS	CLERK	7788	1/12/1983	1100		20
7900	JAMES	CLERK	7698	12/3/1981	950		30
7934	MILLER	CLERK	7782	1/23/1982	1300		10

Se considera tabela **DEPT** având următoarea structură:

Oracle APEX
WKSP_ORACLESTUDENTIAIA.DEPT

Column Name	Data Type	Nullable	Default	Primary Key
DEPTNO	NUMBER(2,0)	No		1
DNAME	VARCHAR2(50)	Yes		
LOC	VARCHAR2(50)	Yes		

și avand următoarele valori introduse:

DEPTNO	DNAME	LOC
10	ACCOUNTING	NEW YORK
20	RESEARCH	DALLAS
30	SALES	CHICAGO
40	OPERATIONS	BOSTON
15	Departament IT	123

1. Scrieți o interogare care sa afișeze toate departamentele inclusiv acele departamente fără niciun angajat. Se va afisa numele, job-ul, numărul departamentului și denumirea departamentului.

2. Scrieți o interogare care sa afișeze toate toți angajații care castiga mai puțin decat angajatul cu id-ul 7902. Se va afisa numele, job-ul și salariul.

3. Scrieți o interogare care sa afișeze toți angajații și managerii lor.

4. Scrieți o interogare care sa afișeze toți angajații care lucrează în același departament cu 'WARD'.

5. Scrieți o interogare care sa afișeze salariul mediu, și numărul de angajați care primesc comision din fiecare departament.

6. Să se listeze angajații din departamentele 10 și 30.

7. Să se selecteze toate funcțiile din departamentul 10 și 20.

8. Să se selecteze funcțiile care au primit același comision și se regăsesc în departamentele 10, 20,30.

9. Să se selecteze funcțiile care se găsesc în departamentul 10 dar nu se regăsesc în departamentul 30.

10. Scrieti o cerere care sa intoarca numele, codul si numele departamentului pentru toti angajatii.

11. Scrieti o cerere care sa aiba ca rezultat toate slujbele care se gasesc in departamentul 20.

12. Realizati o cerere care sa intoarca numele angajatului, numele departamentului, locatia si slujba tuturor celor care castiga un comision.

13. Scrieti o cerere care sa intoarca numele, numarul si numele departamentului si functia tuturor celor care lucreaza in CHICAGO.

14. Gasiti numele, functia, data angajarii si salariul angajatilor al caror salariu este superior celui mai mare salariu al vreunei persoane angajate dupa data de 05/06/1982.

15. Sa se afiseze numele fiecarui angajat, insotit de numele departamentului din care face parte, data angajarii lor.

16. Să se afișeze angajații care au salariul peste valoare medie a departamentului din care fac parte.

Bibliografie:

1. https://ocw.cs.pub.ro/courses/bd/laboratoare/05
2. https://www.w3resource.com/sql-exercises/joins-hr/sql-joins-hr-exercise-26.php
3. Cross Join: http://www.mysqltutorial.org/mysql-cross-join
4. Inner Join: http://www.mysqltutorial.org/mysql-inner-join.aspx
5. Self Join: http://www.mysqltutorial.org/mysql-self-join
6. Left Join: http://www.mysqltutorial.org/mysql-left-join.aspx
7. Right Join: http://www.mysqltutorial.org/mysql-right-join