

LABORATOARE 3 – 4

BAZE DE DATE

LAB: OBIIECTIVE

- Interogari multi-relatie
- Operatori pe multimi
- Subcereri nesincronizate (necorelate)

Exercitii recapitulative

- a) Afisati si comentati structura tabelii DEPARTMENTS
- b) Afisati codul departamentului, denumirea departamentului si codul managerului pentru departamentele avand codul locatiei in multimea {1500,1700,1800} si a caror denumire incepe cu litera 'A'. Rezultatul va fi sortat dupa codul managerului
- c) Afisati data curenta in formatul zi/luna/an. Etichetati coloana rezultat "Astazi".

Exercitii recapitulative - Raspuns

a) Afisati si comentati structura tabelii DEPARTMENTS

DESCRIBE DEPARTMENTS;

Name	Null?	Type
DEPARTMENT_ID	NOT NULL	NUMBER(4)
DEPARTMENT_NAME	NOT NULL	VARCHAR2(30)
MANAGER_ID		NUMBER(6)
LOCATION_ID		NUMBER(4)

b) Afisati codul departamentului, denumirea departamentului si codul managerului pentru departamentele avand codul locatiei in multimea {1500,1700,1800} si a caror denumire incepe cu litera 'A'. Rezultatul va fi sortat dupa codul managerului.

**SELECT DEPARTMENT_ID,DEPARTMENT_NAME,MANAGER_ID
FROM DEPARTMENTS
WHERE LOCATION_ID IN (1500,1700,1800)
AND DEPARTMENT_NAME LIKE 'A%'
ORDER BY MANAGER_ID;**

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID
10	Administration	200
110	Accounting	205

c) Afisati data curenta in formatul zi/luna/an. Etichetati coloana rezultat "Astazi".

**SELECT TO_CHAR(SYSDATE,'DD/MONTH/YYYY') AS "ASTAZI"
FROM DUAL;**

ASTAZI
01/FEBRUARY /2013

Interogari multi-relatie

Interogare multi-relatie = cerere care regaseste date din mai multe tabele.

Operatorii algebrei relationale:

- produs cartezian
- join
- operatori pe multimi (reuniune, diferenta, intersectie)

Interogari multi-relatie :

PRODUS CARTEZIAN

Produsul cartezian – generează toate perechile posibile de tupluri, primul element al perechii fiind luat din prima relație, iar cel de-al doilea element din cealaltă relație

MEDIC			SECTIE		MEDIC × SECTIE				
COD	NUME	ID_SECTIE	ID	DENUMIRE	COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie	100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie	100	Anton	1	2	Chirurgie
120	Bart	1	3	Ortopedie	100	Anton	1	3	Ortopedie
130	Caro	3	1	Neurologie	110	Arsene	1	1	Neurologie
140	Craiu	3	2	Chirurgie	110	Arsene	1	2	Chirurgie
			3	Ortopedie	110	Arsene	1	3	Ortopedie
					120	Bart	1	1	Neurologie
					120	Bart	1	2	Chirurgie
					120	Bart	1	3	Ortopedie
					130	Caro	3	1	Neurologie
					130	Caro	3	2	Chirurgie
					130	Caro	3	3	Ortopedie
					140	Craiu	3	1	Neurologie
					140	Craiu	3	2	Chirurgie
					140	Craiu	3	3	Ortopedie

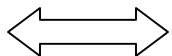
Interogari multi-relatie :

PRODUS CARTEZIAN

Produsul cartezian – generează toate perechile posibile de tupluri, primul element al perechii fiind luat din prima relație, iar cel de-al doilea element din cealaltă relație

MEDIC			SECTIE		MEDIC x SECTIE				
COD	NUME	ID_SECTIE	ID	DENUMIRE	COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie	100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie	100	Anton	1	2	Chirurgie
120	Bart	1	3	Ortopedie	100	Anton	1	3	Ortopedie
130	Caro	3	1	Neurologie	110	Arsene	1	1	Neurologie
140	Craiu	3	2	Chirurgie	110	Arsene	1	2	Chirurgie
			3	Ortopedie	110	Arsene	1	3	Ortopedie
					120	Bart	1	1	Neurologie
					120	Bart	1	2	Chirurgie
					120	Bart	1	3	Ortopedie
					130	Caro	3	1	Neurologie
					130	Caro	3	2	Chirurgie
					130	Caro	3	3	Ortopedie
					140	Craiu	3	1	Neurologie
					140	Craiu	3	2	Chirurgie
					140	Craiu	3	3	Ortopedie

SELECT *
FROM MEDIC, SECTIE;



SELECT *
FROM MEDIC CROSS JOIN SECTIE;

Interogari multi-relatie :

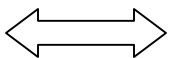
PRODUS CARTEZIAN

In cadrul produsului cartezian, cand realizam proiectie pe anumite coloane, se pot obtine linii duplicate in rezultatul final. Retinem: duplicatele NU sunt eliminate.

Exemplu:

MEDIC			SECTIE		MEDIC × SECTIE				
COD	NUME	ID_SECTIE	ID	DENUMIRE	COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie	100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie	100	Anton	1	2	Chirurgie
120	Bart	1	3	Ortopedie	100	Anton	1	3	Ortopedie
130	Caro	3			110	Arsene	1	1	Neurologie
140	Craiu	3			110	Arsene	1	2	Chirurgie
					110	Arsene	1	3	Ortopedie
					120	Bart	1	1	Neurologie
					120	Bart	1	2	Chirurgie
					120	Bart	1	3	Ortopedie
					130	Caro	3	1	Neurologie
					130	Caro	3	2	Chirurgie
					130	Caro	3	3	Ortopedie
					140	Craiu	3	1	Neurologie
					140	Craiu	3	2	Chirurgie
					140	Craiu	3	3	Ortopedie

**SELECT ID_SECTIE, ID
FROM MEDIC, SECTIE;**



**SELECT ID_SECTIE, ID
FROM MEDIC CROSS JOIN SECTIE;**

Exercitii – set 1

d) Se cunosc:

Cardinalitate (Employees)=107

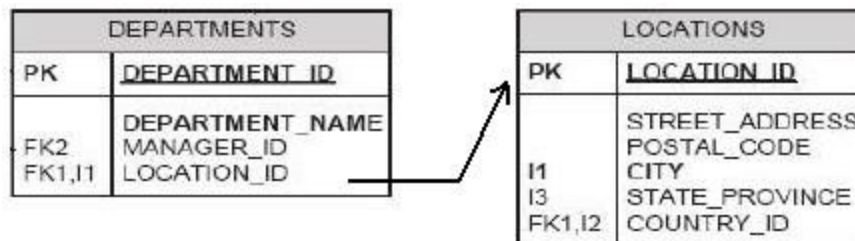
Cardinalitate (Departments) =27

Pot exista mai multi angajati cu acelasi salariu in acelasi departament.

Putem cunoaste exact care va fi cardinalitatea rezultatului urmatoarei cereri?

```
SELECT salary, department_name  
FROM employees, departments;
```

e) Sa se obtina lista tuturor posibilitatilor de amplasare a departamentelor firmei in orase.



Exercitii – set 1 – Rasp.

d) Se cunosc:

Cardinalitate (Employees)=107

Cardinalitate (Departments) =27

Pot exista mai multi angajati cu acelasi salariu in acelasi departament.

Putem cunoaste exact care va fi cardinalitatea rezultatului urmatoarei cereri?

SELECT salary, department_name

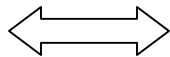
FROM employees, departments;

Da, pentru ca duplicatele sunt pastrate: $107 * 27 = 2889$ linii

e) Sa se obtina lista tuturor posibilitatilor de amplasare a departamentelor firmei in orase.

SELECT department_id, city

FROM departments, locations;



SELECT department_id, city

FROM departments CROSS JOIN locations;

Interogari multi-relatie : JOIN

Join este operatorul de compunere din algebra relationala, ce extrage tupluri din mai multe relații **corelate**.

Join este operația de regăsire a datelor din două sau mai multe tabele, pe baza valorilor comune ale unor coloane. De obicei, aceste coloane reprezintă **cheia primară**, respectiv **cheia externă** a tabelelor.

Recapitulare: cheie primara? cheie externa?
cele 3 reguli de integritate structurala?

Recapitulare – cheile dintr-o tabela

Cheia primara este o mulțime minimală de attribute ale căror valori identifică unic un tuplu într-o relație.

Legatura intre 2 tabele este realizata prin intermediul **cheii externe**.

Cheia externa dintr-o tabela refera cheia primara din cealalta tabela.

Modelul relațional respectă **3 reguli de integritate structurală**.

Regula 1 – unicitatea cheii:

Cheia primară trebuie să fie **unică și minimală**.

Regula 2 – integritatea entității:

Atributele cheii primare trebuie să fie **diferite de valoarea null**.

Regula 3 – integritatea referirii.

O cheie externă trebuie să fie **ori null în întregime, ori să corespundă unei valori a cheii primare asociate**.

Interogari multi-relatie : JOIN

Operatorul JOIN combină produsul cartezian, selecția și proiecția.

- Θ -JOIN (THETA-JOIN)
 - EQUIJOIN
 - € NATURAL JOIN
 - NONEQUIJOIN
- SELF JOIN
- OUTER JOIN (compunere externa)
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN

Interogari multi-relatie : THETA-JOIN

Operatorul θ -JOIN combină tupluri din două relații cu condiția ca valorile atributelor specificate să satisfacă o anumită condiție specificată explicit în cadrul operației.

$$\text{JOIN}(R, S, \text{condiție}) = \sigma_{\text{condiție}} (R \times S)$$

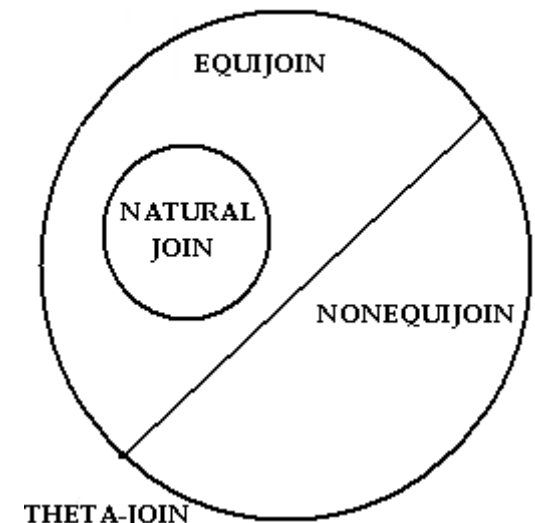
Condiție testează egalitate de coloane?

DA \Rightarrow **equijoin**

Egalitate pe toate coloanele ce
au același nume în cele 2 tabele?

DA \Rightarrow natural join

NU \Rightarrow nonequijoin

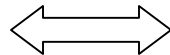


Interogari multi-relatie : EQUIJOIN

$JOIN(MEDIC, SECTIE, MEDIC.ID_SECTIE=SECTIE.ID)$

$= \sigma_{MEDIC.ID_SECTIE=SECTIE.ID} (MEDIC \times SECTIE)$

SELECT *
FROM MEDIC, SECTIE
WHERE MEDIC.ID_SECTIE=SECTIE.ID;



SELECT *
FROM MEDIC JOIN SECTIE
ON (MEDIC.ID_SECTIE=SECTIE.ID);

MEDIC			SECTIE	
COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie
120	Bart	1	3	Ortopedie
130	Caro	3	1	Neurologie
140	Craiu	3	2	Chirurgie
			3	Ortopedie

X

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
100	Anton	1	2	Chirurgie
100	Anton	1	3	Ortopedie
110	Arsene	1	1	Neurologie
110	Arsene	1	2	Chirurgie
110	Arsene	1	3	Ortopedie
120	Bart	1	1	Neurologie
120	Bart	1	2	Chirurgie
120	Bart	1	3	Ortopedie
130	Caro	3	1	Neurologie
130	Caro	3	2	Chirurgie
130	Caro	3	3	Ortopedie
140	Craiu	3	1	Neurologie
140	Craiu	3	2	Chirurgie
140	Craiu	3	3	Ortopedie

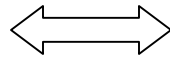
σ

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie

Interogari multi-relatie: EQUIJOIN

In cadrul produsului EQUIJOIN, cand realizam proiectie pe anumite coloane, se pot obtine linii duplicate in rezultatul final. Retinem: duplicatele NU sunt eliminate.

**SELECT MEDIC.ID_SECTIE,SECTIE.ID
FROM MEDIC, SECTIE
WHERE MEDIC.ID_SECTIE=SECTIE.ID;**



**SELECT MEDIC.ID_SECTIE,SECTIE.ID
FROM MEDIC JOIN SECTIE
ON (MEDIC.ID_SECTIE=SECTIE.ID);**

MEDIC			SECTIE	
COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie
120	Bart	1	3	Ortopedie
130	Caro	3		
140	Craiu	3		



COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie



MEDIC.ID_SECTIE, SECTIE.ID

ID_SECTIE	ID
1	1
1	1
1	1
3	3
3	3

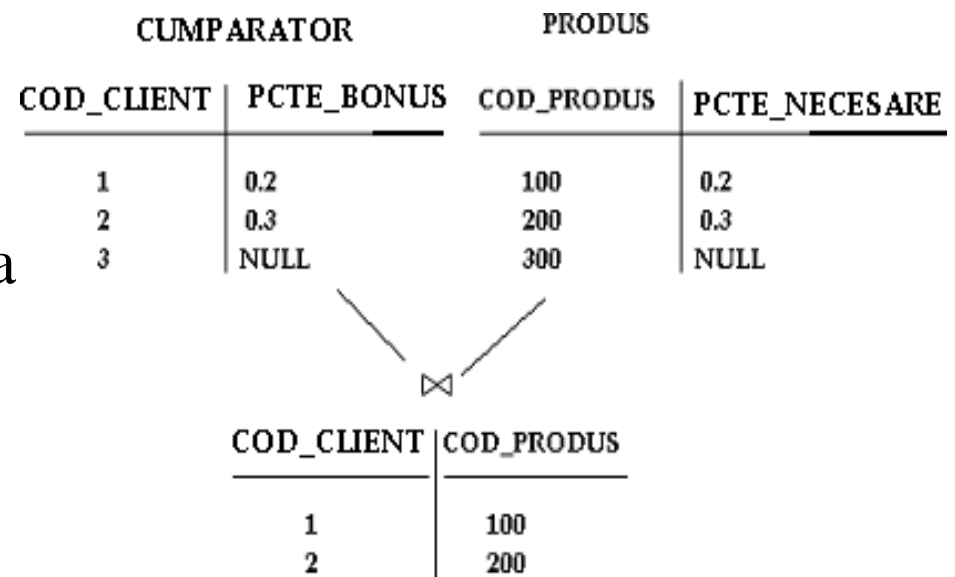
Interogari multi-relatie: EQUIJOIN

In cadrul produsului EQUIJOIN, daca se ajunge la **compararea a doua valori NULL**, aflate in coloanele de join, atunci evaluarea conditiei va fi NULL si in consecinta tuplul **NU va fi inclus in rezultatul final**.

Exemplu:

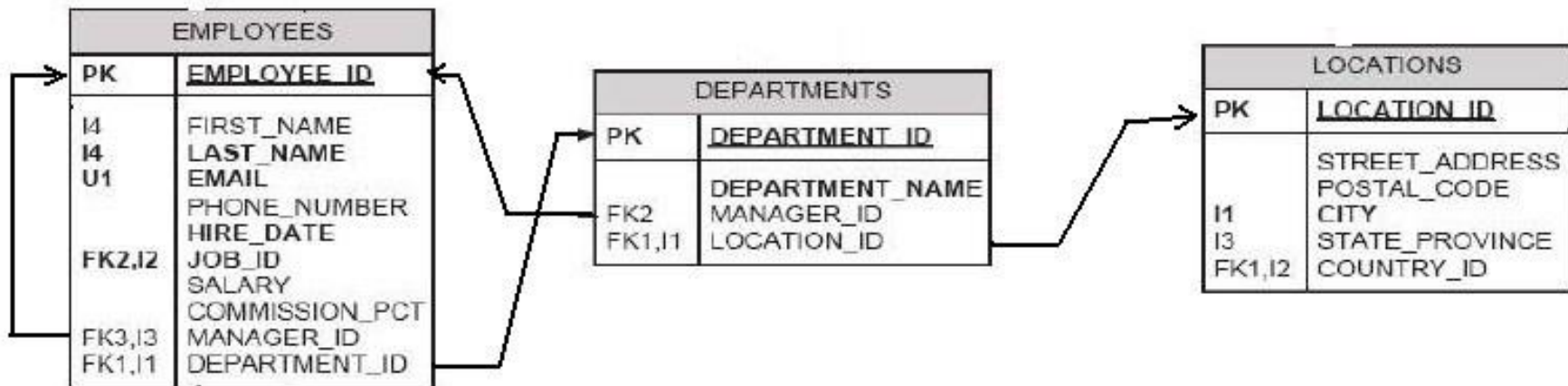
Clientii au posibilitatea de a obtine un singur produs pe baza punctelor bonus de fidelitate.

```
SELECT c.cod_client, p.cod_produ  
FROM PRODUS p, CLIENT c  
WHERE p.pcte_necesare=c.pcte_bonus;
```



Exercitii EQUIJOIN– set 2

- f) Să se afișeze pentru toți angajații: numele salariatului, codul departamentului și numele departamentului din care face parte.
- g) Să se listeze job-urile distincte care există în departamentul 30.
- h) Să se afișeze numele, job-ul, codul și numele departamentului pentru toți angajații care lucrează în Oxford.



Exercitii EQUIJOIN– set 2 – Rasp.

f) Să se afișeze pentru toți angajații: numele salariatului, codul departamentului și numele departamentului din care face parte.

Gresit:

SELECT last_name, department_id, department_name

FROM EMPLOYEES, DEPARTMENTS

WHERE EMPLOYEES.department_id=DEPARTMENTS.department_id;

ERROR at line 1:
ORA-00918: column ambiguously defined

Exercitii EQUIJOIN– set 2 – Rasp.

f) Să se afișeze pentru toți angajații: numele salariatului, codul departamentului și numele departamentului din care face parte.

Gresit:

SELECT last_name, department_id, department_name

FROM EMPLOYEES, DEPARTMENTS

WHERE EMPLOYEES.department_id=DEPARTMENTS.department_id;

ERROR at line 1:
ORA-00918: column ambiguously defined

Retinem: Coloanele care apar cu acelasi nume in mai multe tabele din join trebuie prefixate cu numele (sau cu alias-ul) tabelei din care se doresc extrase datele coloanei. Restul coloanelor, pentru care nu exista ambiguitate de ce tabela apartin, pot fi lasate neprefixate.

Exercitii EQUIJOIN– set 2 – Rasp.

f) Să se afișeze pentru toți angajații: numele salariatului, codul departamentului și numele departamentului din care face parte.

Gresit:

SELECT last_name, department_id, department_name

FROM EMPLOYEES, DEPARTMENTS

WHERE EMPLOYEES.department_id=DEPARTMENTS.department_id;

ERROR at line 1:
ORA-00918: column ambiguously defined

Retinem: Coloanele care apar cu acelasi nume in mai multe tabele din join trebuie prefixate cu numele (sau cu alias-ul) tabelei din care se doresc extrase datele coloanei. Restul coloanelor, pentru care nu exista ambiguitate de ce tabela apartin, pot fi lasate neprefixate.

Corect:

SELECT last_name, DEPARTMENTS.department_id, department_name

FROM EMPLOYEES, DEPARTMENTS

WHERE EMPLOYEES.department_id=DEPARTMENTS.department_id;

Exercitii EQUIJOIN – set 2 – Rasp.

f) Să se afișeze pentru toți angajații: numele salariatului, codul departamentului și numele departamentului din care face parte.

Corect (de pe pagina anterioara):

```
SELECT last_name, DEPARTMENTS.department_id, department_name  
FROM EMPLOYEES, DEPARTMENTS  
WHERE EMPLOYEES.department_id=DEPARTMENTS.department_id;
```

O buna regula: folosirea alias-urilor pentru tabele. Usureaza scrierea, cresc lizibilitatea codului.

Intrucat clauza FROM se executa prima => alias-urile tabelor sunt vizibile in toate clauzele cererii.

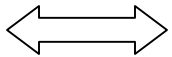
```
SELECT e.last_name, d.department_id, d.department_name  
FROM EMPLOYEES e, DEPARTMENTS d  
WHERE e.department_id=d.department_id;
```

Exercitii EQUIJOIN – set 2 – Rasp.

f) Să se afișeze pentru toți angajații: numele salariatului, codul și numele departamentului din care face parte.

Corect cu alias-uri la tabele (de pe pagina anterioara):

```
SELECT e.last_name, d.department_id, d.department_name  
FROM EMPLOYEES e, DEPARTMENTS d  
WHERE e.department_id=d.department_id;
```



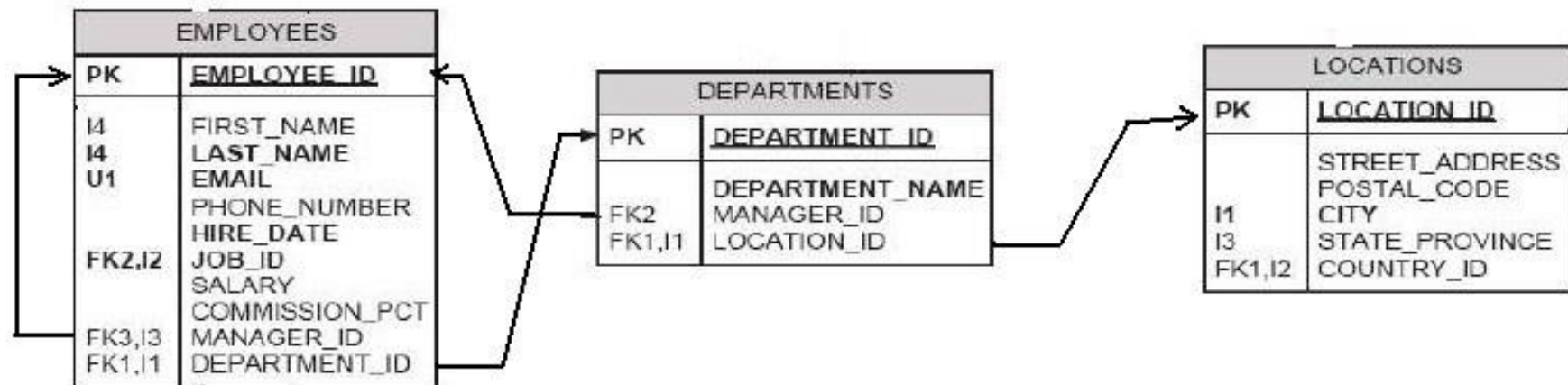
```
SELECT e.last_name, d.department_id, d.department_name  
FROM EMPLOYEES e JOIN DEPARTMENTS d  
ON (e.department_id=d.department_id);
```

Exercitii EQUIJOIN– set 2

g) Să se listeze job-urile distincte care există în departamentul 30.

(in clauza where putem testa si alte conditii, pe langa conditia de join)

h) Să se afișeze numele, job-ul, codul și numele departamentului pentru toți angajații care lucrează în Oxford.

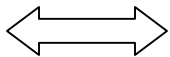
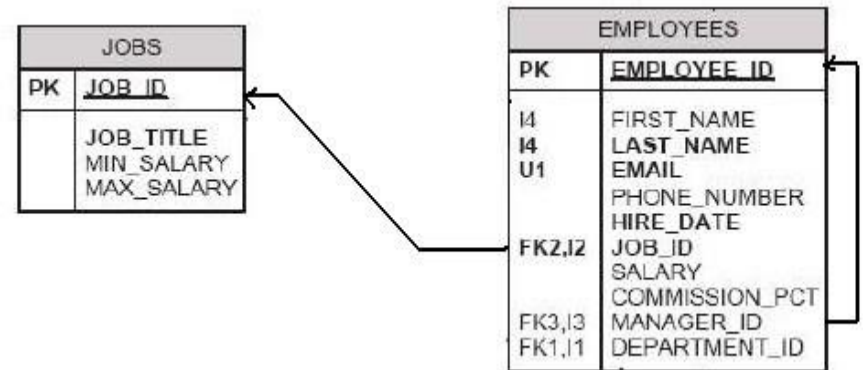


Exercitii EQUIJOIN – set 2 – Rasp.

g) Să se listeze job-urile distincte, cod job si denumire job, care există în departamentul 30.

(in clauza where putem testa si alte conditii, pe langa conditia de join)

```
SELECT DISTINCT j.job_title, j.job_id  
FROM JOBS j, EMPLOYEES e  
WHERE j.job_id=e.job_id  
      AND e.department_id=30;
```



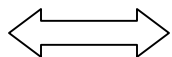
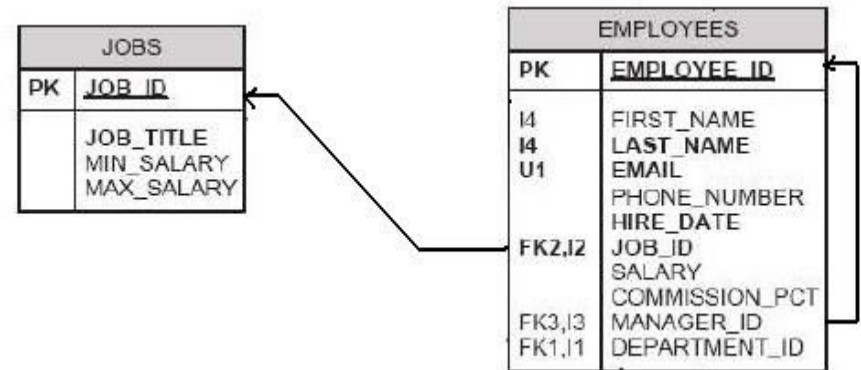
```
SELECT DISTINCT j.job_title, j.job_id  
FROM JOBS j JOIN EMPLOYEES e  
      ON (j.job_id=e.job_id)  
WHERE e.department_id=30;
```

Exercitii EQUIJOIN – set 2 – Rasp.

g) Să se listeze job-urile distincte, cod job si denumire job, care există în departamentul 30.

(in clauza where putem testa si alte conditii, pe langa conditia de join)

```
SELECT DISTINCT j.job_title, j.job_id  
FROM JOBS j, EMPLOYEES e  
WHERE j.job_id=e.job_id  
AND e.department_id=30;
```



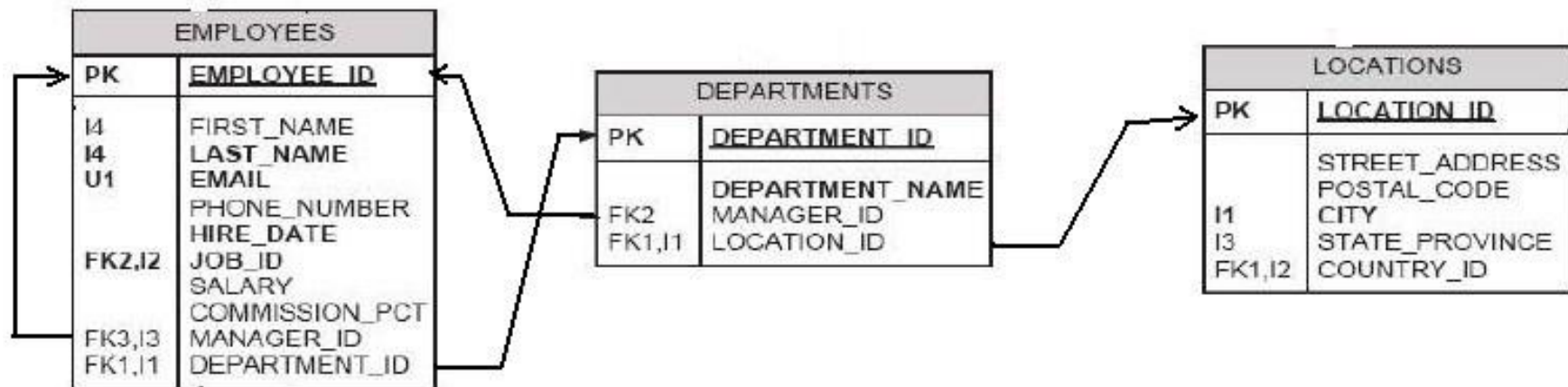
```
SELECT DISTINCT j.job_title, j.job_id  
FROM JOBS j JOIN EMPLOYEES e  
ON (j.job_id=e.job_id)  
WHERE e.department_id=30;
```

Retinem: numai conditia de join poate fi plasata in clauza FROM dupa ON, restul conditiilor de selectie raman in WHERE!

Exercitii EQUIJOIN– set 2

h) Să se afișeze numele, job-ul, codul și numele departamentului pentru toți angajații care lucrează în Oxford.

(3 tabele sursa EMPLOYEES, DEPARTMENTS, LOCATIONS
=> $3-1=2$ conditii de join)

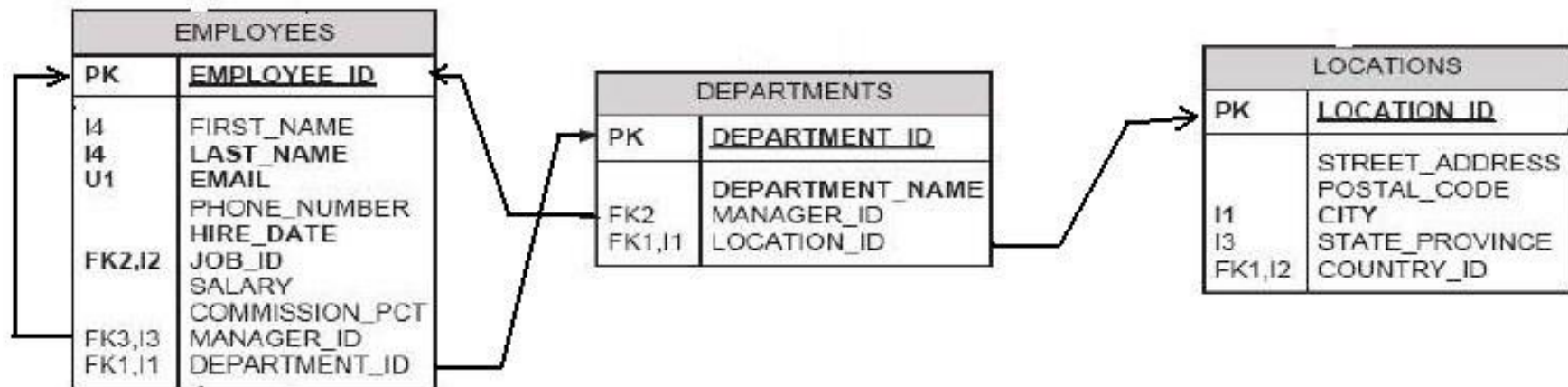


Exercitii EQUIJOIN – set 2 – Rasp.

h) Să se afișeze numele, job-ul, codul departamentului și numele departamentului pentru toți angajații care lucrează în orasul Oxford.

(3 tabele sursa EMPLOYEES, DEPARTMENTS, LOCATIONS => 3-1=2 conditii de join):

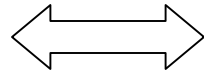
```
SELECT e.last_name, e.job_id, d.department_id, d.department_name
FROM EMPLOYEES e, DEPARTMENTS d, LOCATIONS l
WHERE e.department_id=d.department_id
      AND d.location_id=l.location_id
      AND l.city='Oxford';
```



Exercitii EQUIJOIN – set 2 – Rasp.

- h) Să se afișeze numele, job-ul, codul departamentului și numele departamentului pentru toți angajații care lucrează în orasul Oxford.

```
SELECT e.last_name, e.job_id, d.department_id, d.department_name
FROM EMPLOYEES e, DEPARTMENTS d, LOCATIONS l
WHERE e.department_id=d.department_id
      AND d.location_id=l.location_id
      AND l.city='Oxford';
```



```
SELECT e.last_name, e.job_id, d.department_id, d.department_name
FROM EMPLOYEES e JOIN DEPARTMENTS d
                ON (e.department_id=d.department_id)
                JOIN LOCATIONS l
                ON (d.location_id=l.location_id)
WHERE l.city='Oxford';
```

Remarca: in scrierea SQL3, in clauza ON pot sa apara tabele deja declarate anterior in cadrul clauzei FROM. Parsarea si verificarea corectitudinii clauzei FROM se face de la stanga la dreapta.

Exercitiu EQUIJOIN : INDIVIDUAL

- i) Creați o cerere prin care să se afișeze numele angajatului, codul job-ului, titlul job-ului, numele departamentului și salariul angajaților având salariu sub 2000 \$ sau care au fost angajați în anul 1987.



Exercitiu EQUIJOIN : INDIVIDUAL -Rasp

- i) Creați o cerere prin care să se afișeze numele angajatului, codul job-ului, titlul job-ului, numele departamentului și salariul angajaților având salariu sub 2000 \$ sau care au fost angajați în anul 1987.

```
SELECT e.last_name,j.job_id,j.job_title, d.department_name  
FROM JOBS j,EMPLOYEES e, DEPARTMENTS d  
WHERE j.job_id=e.job_id  
      AND e.department_id=d.department_id  
      AND (e.salary<2000  
           OR to_char(e.hire_date,'YYYY')='1987');
```

Interogari multi-relatie : JOIN

Operatorul JOIN combină produsul cartezian, selecția și proiecția.

- Θ -JOIN (THETA-JOIN)
 - ✓ EQUIJOIN
 - € NATURAL JOIN
 - NONEQUIJOIN
- SELF JOIN
- OUTER JOIN (compunere externa)
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN

Interogari multi-relatie : NATURAL JOIN

Presupune existența unor *coloane având același nume în ambele tabele*.

Clauza determină selectarea liniilor din cele două tabele, care au *valori egale în aceste coloane*.

Dacă tipurile de date ale coloanelor cu nume identice sunt diferite, va fi returnată o eroare!

Intrebare: Date tabelele fiind MEDIC si SECTIE putem calcula rezultatul unui NATURAL JOIN intre aceste tabele?

MEDIC			SECTIE	
COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	2	Chirurgie
120	Bart	1	3	Ortopedie
130	Caro	3		
140	Craiu	3		

Interogari multi-relatie : NATURAL JOIN

Presupune existența unor *coloane având același nume în ambele tabele*.

Clauza determină selectarea liniilor din cele două tabele, care au *valori egale în aceste coloane*.

Dacă tipurile de date ale coloanelor cu nume identice sunt diferite, va fi returnată o eroare!

Intrebare: Date tabelele fiind MEDIC si SECTIE putem calcula rezultatul unui NATURAL JOIN intre aceste tabele?

MEDIC		
COD	NUME	ID_SECTIE
100	Anton	1
110	Arsene	1
120	Bart	1
130	Caro	3
140	Craiu	3

SECTIE	
ID	DENUMIRE
1	Neurologie
2	Chirurgie
3	Ortopedie

Nu, pentru ca nu au coloane cu acelasi nume in ambele tabele!

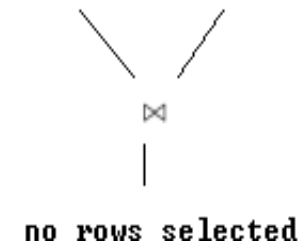
Interogari multi-relatie : NATURAL JOIN

Consideram tabelele MEDIC si SECTIE2.

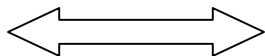
Clauza NATURAL JOIN are urmatorul efect : se va testa implicit egalitatea între **toate perechile de coloane care au acelasi denumire in ambele tabele.**

COD	NUME	ID_SECTIE
100	Anton	1
110	Arsene	1
120	Bart	1
130	Caro	3
140	Craiu	3

ID_SECTIE	NUME
1	Neurologie
2	Chirurgie
3	Ortopedie



```
SELECT MEDIC.id_sectie,  
SECTIE2.num     FROM MEDIC ,  
SECTIE2  
WHERE  
    MEDIC.id_sectie=SECTIE2.id_sectie  
    AND MEDIC.num     = SECTIE2.num;
```

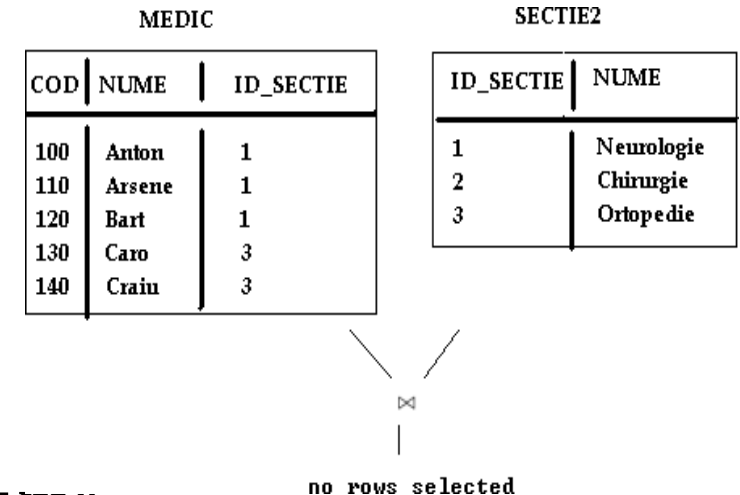


```
SELECT id_sectie, num  
FROM MEDIC NATURAL JOIN SECTIE2;
```

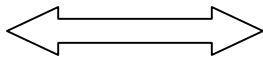
Interogari multi-relatie : NATURAL JOIN

Consideram tabelele MEDIC si SECTIE2.

Clauza NATURAL JOIN are urmatorul efect : se va testa implicit egalitatea intre toate perechile de coloane care au aceeasi denumire in ambele tabele.



```
SELECT MEDIC.id_sectie, SECTIE2.numa  
FROM MEDIC , SECTIE2  
WHERE MEDIC.id_sectie=SECTIE2.id_sectie  
AND MEDIC.numa = SECTIE2.numa;
```



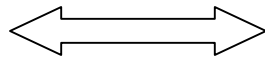
```
SELECT id_sectie, nume  
FROM MEDIC NATURAL JOIN SECTIE2;
```

Remarca: coloanele comune de join sa nu fie prefixate niciunde in cerere!

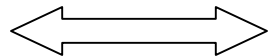
Interogari multi-relatie : NATURAL JOIN

Ce optiuni avem la dispozitie cand mai multe coloane au aceeasi denumire intre 2 tabele, dar vrem join doar pe un subset dintre acestea?

**SELECT m.id_sectie, s2.num
FROM MEDIC m, SECTIE2 s2
WHERE m.id_sectie=s2.id_sectie;**



**SELECT m.id_sectie, s2.num
FROM MEDIC m JOIN SECTIE2 s2
ON (m.id_sectie=s2.id_sectie);**



**SELECT id_sectie, s2.num
FROM MEDIC m JOIN SECTIE2 s2 USING (id_sectie);**

MEDIC

COD	NUME	ID_SECTIE
100	Anton	1
110	Arsene	1
120	Bart	1
130	Caro	3
140	Craiu	3

SECTIE2

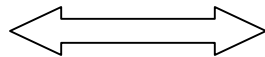
ID_SECTIE	NUME
1	Neurologie
2	Chirurgie
3	Ortopedie

ID_SECTIE	NUME
1	Neurologie
1	Neurologie
1	Neurologie
3	Ortopedie
3	Ortopedie

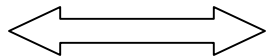
Interogari multi-relatie : NATURAL JOIN

Ce optiuni avem la dispozitie cand mai multe coloane au aceeasi denumire intre 2 tabele, dar vrem join doar pe un subset dintre acestea?

```
SELECT m.id_sectie, s2.numa  
FROM MEDIC m, SECTIE2 s2  
WHERE m.id_sectie=s2.id_sectie;
```



```
SELECT m.id_sectie, s2.numa  
FROM MEDIC m JOIN SECTIE2 s2  
ON (m.id_sectie=s2.id_sectie);
```



```
SELECT id_sectie, s2.numa  
FROM MEDIC m JOIN SECTIE2 s2 USING (id_sectie);
```

Remarca: campul comun de join sa nu fie prefixat niciunde in cerere!

Interogari multi-relatie : NATURAL JOIN

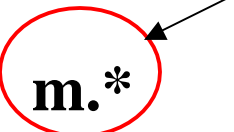
Remarca:

Atunci cand avem coloane cu aceeasi denumire in doua tabele si folosim clauza *tabela1 NATURAL JOIN tabela2*

sau clauza *tabela1 JOIN tabela2 USING (coli,colj,colk)*

nu avem voie sa prefixam niciunde in cerere coloanele de join.

Din acest motiv, cererea urmatoare este **gresita**:

 **SELECT** **m.***
FROM **MEDIC** **m** **JOIN** **SECTIE2** **s2** **USING** (**id_sectie**);

ERROR at line 1:
ORA-25154: column part of USING clause cannot have qualifier

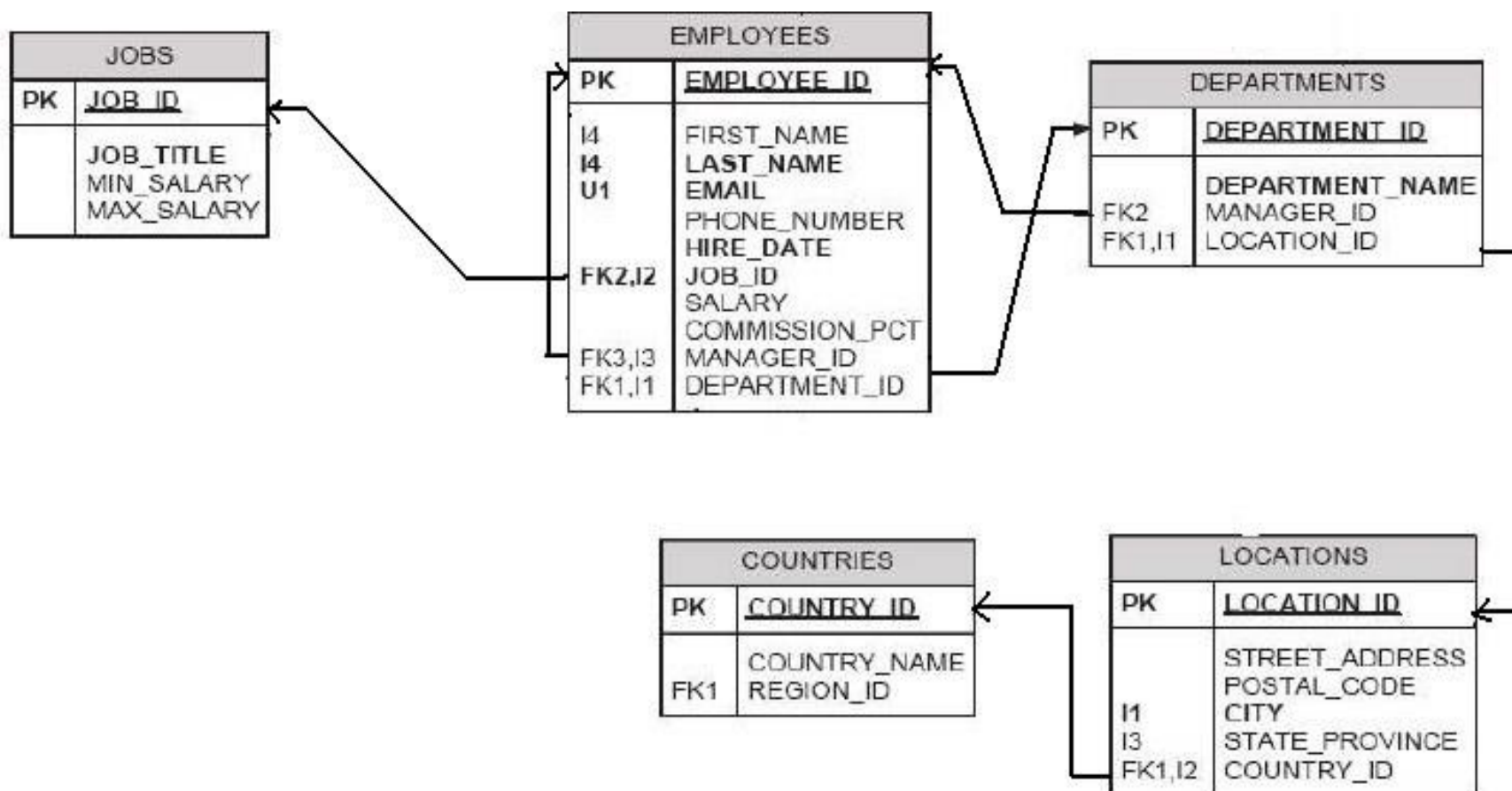
Deoarece prin expandare, ar fi: **SELECT** m.cod,m.nume,**m.id_sectie**

O alta consecinta a imposibilitatii prefixarii coloanelor comune de join este ca **in rezultatul final coloanele comune pot apare doar o singura data.**

Exercitii NATURAL JOIN – set 3

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, orașul și țara în care lucrează angajatii condusi direct de managerul cu cod 100.



Exercitii NATURAL JOIN – set 3 -Rasp

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, oraşul şi ţara în care lucrează angajatii condusi direct de managerul cu cod 100.

Varianta 1)

```
SELECT e.last_name,e.salary,j.job_title,l.city,c.country_name
FROM JOBS j, EMPLOYEES e, DEPARTMENTS d, LOCATIONS l, COUNTRIES c
WHERE j.job_id=e.job_id
      AND e.department_id=d.department_id
      AND d.location_id=l.location_id
      AND l.country_id=c.country_id
      AND e.manager_id=100;
```

Exercitii NATURAL JOIN – set 3 -Rasp

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, oraşul şi ţara în care lucrează angajatii condusi direct de managerul cu cod 100.

Varianta 2)

```
SELECT e.last_name,e.salary,j.job_title,l.city,c.country_name
FROM JOBS j JOIN EMPLOYEES e
      ON (j.job_id=e.job_id)
JOIN DEPARTMENTS d
      ON (e.department_id=d.department_id)
JOIN LOCATIONS l
      ON (d.location_id=l.location_id)
JOIN COUNTRIES c
      ON (l.country_id=c.country_id)
WHERE e.manager_id=100;
```

Exercitii NATURAL JOIN – set 3 -Rasp

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, oraşul şi ţara în care lucrează angajatii condusi direct de managerul cu cod 100.

Varianta 3) Posibila, pentru ca exista coloane cu nume comun intre tabele, 2 cate 2

```
SELECT e.last_name,e.salary,j.job_title,l.city,c.country_name
FROM JOBS j JOIN EMPLOYEES e USING (job_id)
      JOIN DEPARTMENTS d USING (department_id)
      JOIN LOCATIONS l USING (location_id)
      JOIN COUNTRIES c USING (country_id)
WHERE   e.manager_id=100;
```

Exercitii NATURAL JOIN – set 3 -Rasp

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, oraşul şi ţara în care lucrează angajatii condusi direct de managerul cu cod 100.

Varianta x) Posibila, pentru ca exista coloane cu nume comun intre tabele, 2 cate 2

SELECT e.last_name,e.salary,j.job_title,l.city,c.country_name

FROM JOBS j NATURAL JOIN EMPLOYEES e

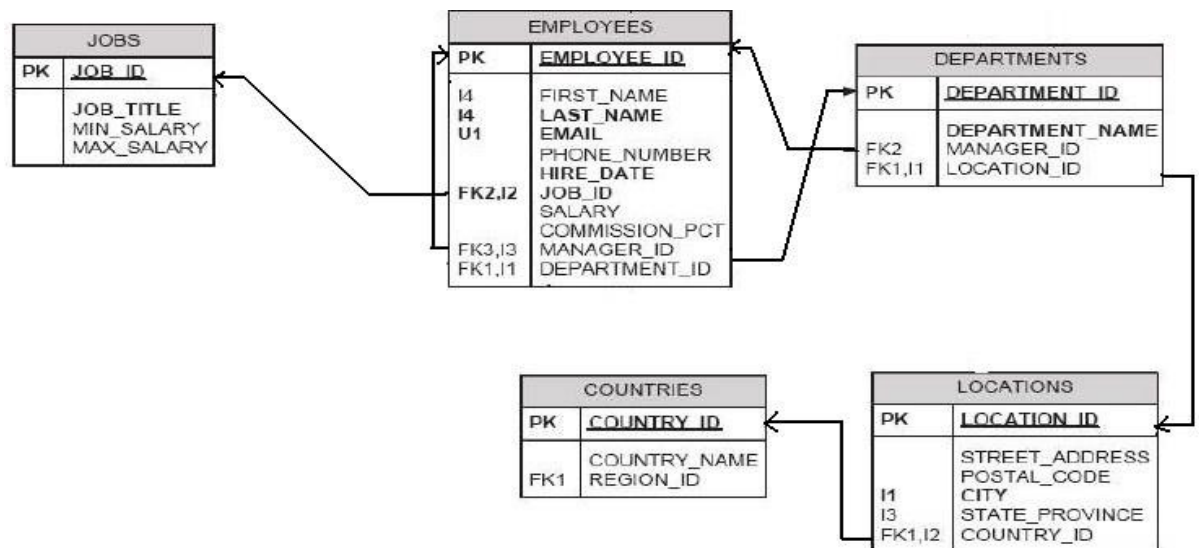
NATURAL JOIN DEPARTMENTS d

NATURAL JOIN LOCATIONS l

NATURAL JOIN COUNTRIES c

WHERE e.manager_id=100;

Care este problema?



Exercitii NATURAL JOIN – set 3 -Rasp

j) Scrieti in 3 forme echivalente pentru rezolvarea urmatorului enunt:

Sa se afiseze numele, salariul, titlul job-ului, oraşul şi ţara în care lucrează angajatii condusi direct de managerul cu cod 100.

Varianta x) Posibila, pentru ca exista coloane cu nume comun intre tabele, 2 cate 2

SELECT e.last_name,e.salary,j.job_title,l.city,c.country_name

FROM JOBS j NATURAL JOIN EMPLOYEES e

NATURAL JOIN DEPARTMENTS d

NATURAL JOIN LOCATIONS l

NATURAL JOIN COUNTRIES c

WHERE e.manager_id=100;

Care este problema?

Coloanele cu denumire comuna

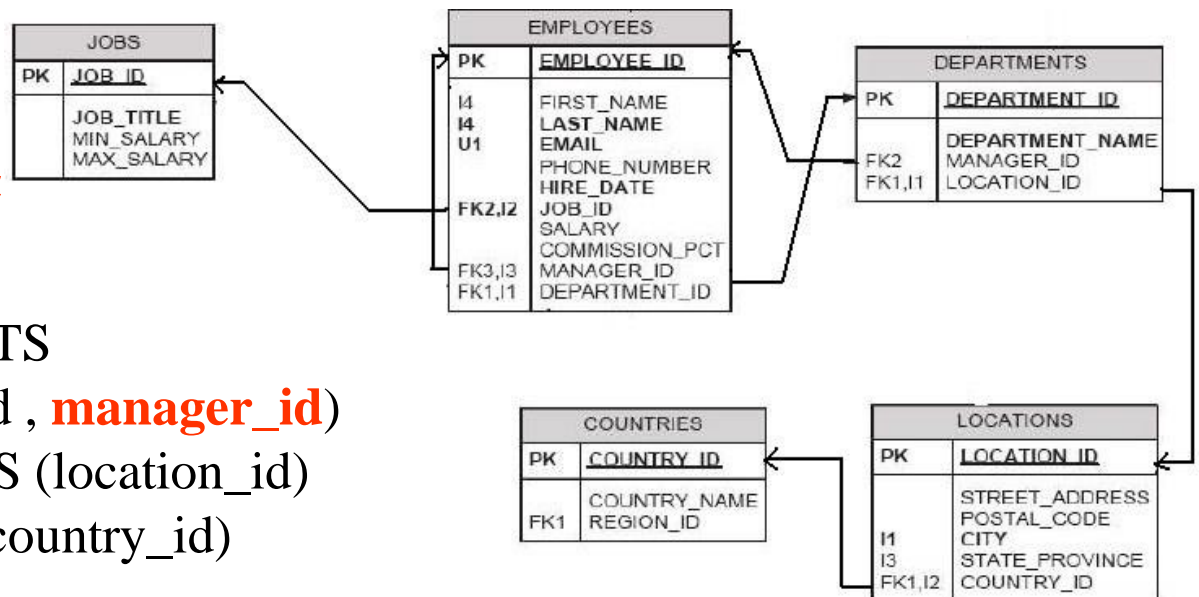
JOBS – EMPLOYEES (job_id)

EMPLOYEES – DEPARTMENTS

(department_id , **manager_id**)

DEPARTMENTS –LOCATIONS (location_id)

LOCATIONS – COUNTRIES (country_id)



Interogari multi-relatie : JOIN

Operatorul JOIN combină produsul cartezian, selecția și proiecția.

- Θ -JOIN (THETA-JOIN)
 - ✓ EQUIJOIN
 - ✓ NATURAL JOIN
 - NONEQUIJOIN
- SELF JOIN
- OUTER JOIN (compunere externa)
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN

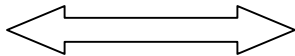
Interogari multi-relatie : NONEQUIJOIN

Condiția de join conține alți operatori decât operatorul egalitate.

Insa nu uitam ca este un join => date din mai multe tabele ce sunt comparate.

Exercitiu: k) Sa se gaseasca toate titlurile de job-uri care au maximul salariului mai mic decat salariul angajatului cu cod 100.

```
SELECT j.job_title  
FROM JOBS j, EMPLOYEES e  
WHERE j.max_salary<e.salary  
      AND e.employee_id=100;
```



```
SELECT j.job_title  
FROM JOBS j JOIN EMPLOYEES e  
      ON (j.max_salary<e.salary)  
WHERE e.employee_id=100;
```

Interogari multi-relatie : JOIN

Operatorul JOIN combină produsul cartezian, selecția și proiecția.

- Θ -JOIN (THETA-JOIN)
 - ✓ EQUIJOIN
 - ✓ NATURAL JOIN
 - ✓ NONEQUIJOIN
- SELF JOIN
- OUTER JOIN (compunere externa)
 - LEFT OUTER JOIN
 - RIGHT OUTER JOIN
 - FULL OUTER JOIN

Interogari multi-relatie : SELF JOIN

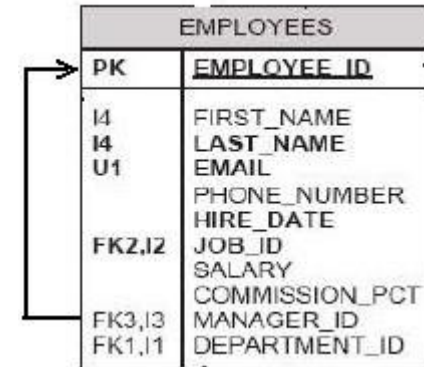
Reprezinta join-ul unui tabel cu el însuși.

Exercitiu:

1) Self join de tip nonequijoin

Să se afișeze numele și data angajării pentru salariații care au fost angajați după *Gates*

```
SELECT E.last_name, E.hire_date
FROM EMPLOYEES E, EMPLOYEES E2
WHERE E2.last_name='Gates'
      AND E.hire_date>E2.hire_date;
```



Interogari multi-relatie : SELF JOIN

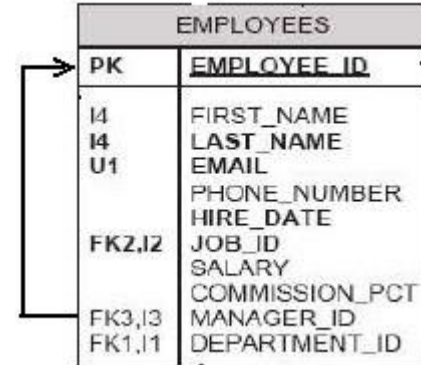
Reprezinta join-ul unui tabel cu el însuși.

Exercitiu:

1) Self join de tip nonequijoin

Să se afișeze numele și data angajării pentru salariații care au fost angajați după *Gates*

```
SELECT E.last_name, E.hire_date  
FROM EMPLOYEES E, EMPLOYEES E2 ←  
WHERE E2.last_name='Gates'  
AND E.hire_date>E2.hire_date;
```



EMPLOYEES	
PK	EMPLOYEE_ID
I4	FIRST_NAME
I4	LAST_NAME
U1	EMAIL
	PHONE_NUMBER
	HIRE_DATE
FK2,I2	JOB_ID
	SALARY
	COMMISSION_PCT
FK3,I3	MANAGER_ID
FK1,I1	DEPARTMENT_ID

Aceeasi tabela este scrisa de 2 ori in clauza FROM, dar are neaparat alias-uri de tabela diferite!

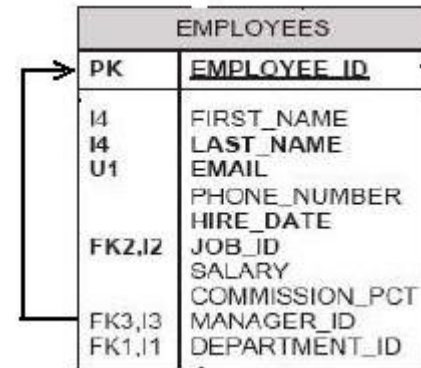
Interogari multi-relatie : SELF JOIN

Reprezinta join-ul unui tabel cu el însuși.

Exercitiu:

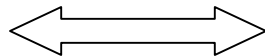
1) Self join de tip nonequijoin

Să se afișeze numele și data angajării pentru salariații care au fost angajați după *Gates*



EMPLOYEES	
PK	EMPLOYEE_ID
I4	FIRST_NAME
I4	LAST_NAME
U1	EMAIL
	PHONE_NUMBER
	HIRE_DATE
FK2,I2	JOB_ID
	SALARY
	COMMISSION_PCT
FK3,I3	MANAGER_ID
FK1,I1	DEPARTMENT_ID

```
SELECT E.last_name, E.hire_date
FROM EMPLOYEES E, EMPLOYEES E2
WHERE E2.last_name='Gates'
      AND E.hire_date>E2.hire_date;
```



```
SELECT E.last_name, E.hire_date
FROM EMPLOYEES E JOIN EMPLOYEES E2
      ON (E.hire_date>E2.hire_date)
WHERE E2.last_name='Gates';
```

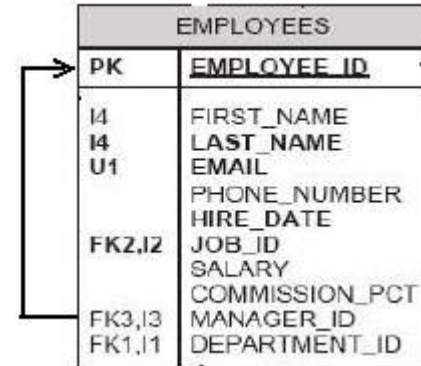
Interogari multi-relatie : SELF JOIN

Reprezinta join-ul unui tabel cu el însuși.

Exercitiu:

m) Self join de tip equijoin

Scriti o cerere pentru a se afisa numele, luna (în litere) și anul angajării pentru toti salariatii din acelasi departament cu Gates, al căror nume conține litera “a”.



The diagram shows the EMPLOYEES table with columns: EMPLOYEE_ID (PK), FIRST_NAME, LAST_NAME, EMAIL, PHONE_NUMBER, HIRE_DATE, JOB_ID, SALARY, COMMISSION_PCT, MANAGER_ID, and DEPARTMENT_ID. A self-join relationship is indicated by an arrow from the PK column to the MANAGER_ID column.

EMPLOYEES	
PK	EMPLOYEE_ID
I4	FIRST_NAME
I4	LAST_NAME
U1	EMAIL
	PHONE_NUMBER
	HIRE_DATE
FK2,I2	JOB_ID
	SALARY
	COMMISSION_PCT
FK3,I3	MANAGER_ID
FK1,I1	DEPARTMENT_ID

```
SELECT e.last_name, to_char(e.hire_date,'MONTH') as "luna",  
       to_char(e.hire_date,'YYYY') as "anul"
```

```
FROM employees e, employees e2
```

```
WHERE e2.last_name='Gates'
```

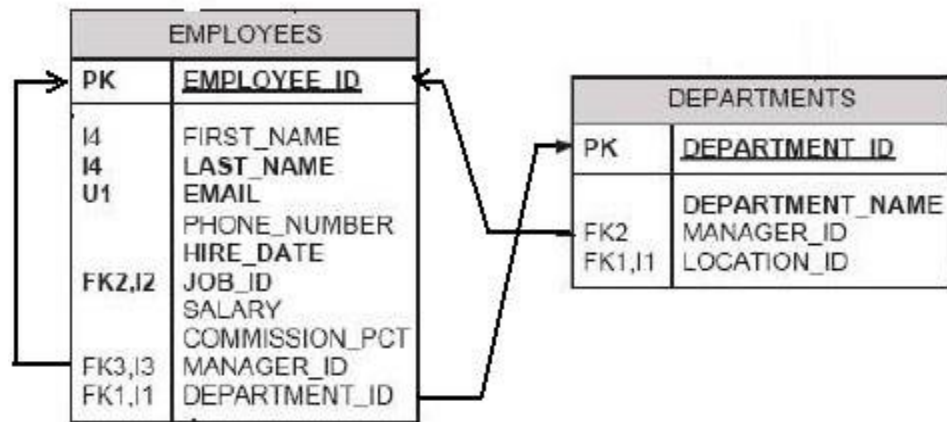
```
AND e.department_id=e2.department_id
```

```
AND e.last_name LIKE '%a%';
```

Exercitiu SELF JOIN – set 4 :

INDIVIDUAL

n) Sa se afiseze codul și numele angajaților care lucrează în același departament cu cel puțin un angajat al cărui nume conține litera “t”.

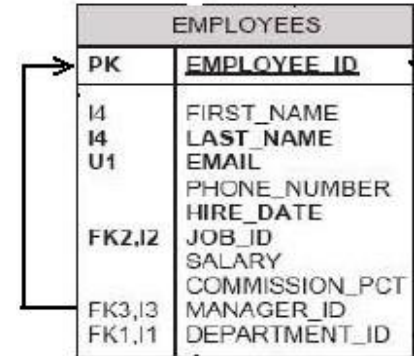


o) Extindeti exercitiul n) ca sa se afiseze codul și numele **si denumirea departamentului** angajaților care lucrează în același departament cu cel puțin un angajat al cărui nume conține litera “t”.

Exercitiu SELF JOIN –set 4 - Rasp

n) Sa se afiseze codul și numele angajaților care lucrează în același departament cu cel puțin un angajat al cărui nume conține litera “t”.

```
SELECT DISTINCT E.employee_id, E.last_name
FROM EMPLOYEES E, EMPLOYEES E2
WHERE E.department_id= E2.department_id
      AND lower(E2.last_name) LIKE '%t%'
      AND e.employee_id != e2.employee_id;
```



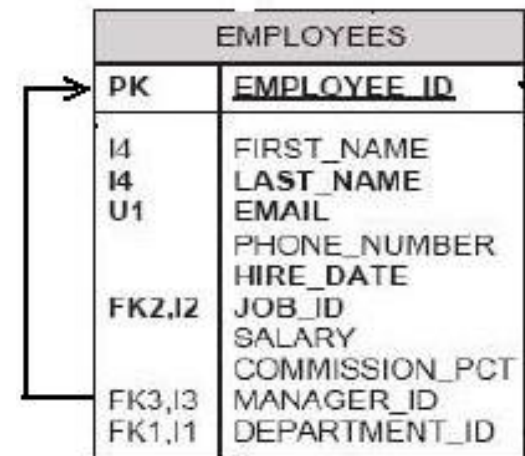
o) Extindeți exercitiul n) ca să se afișeze codul și numele și denumirea departamentului angajaților care lucrează în același departament cu cel puțin un angajat al cărui nume conține litera “t”.

```
SELECT DISTINCT E.employee_id, E.last_name, D.department_name
FROM EMPLOYEES E, EMPLOYEES E2, DEPARTMENTS D
WHERE E.department_id= E2.department_id
      AND E.department_id=D.department_id
      AND lower(E2.last_name) LIKE '%t%'
      AND e.employee_id != e2.employee_id;
```

Exercitiu fara calculator

Ce efect are comanda urmatoare?

```
SELECT *  
FROM EMPLOYEES e NATURAL JOIN  
EMPLOYEES e2;
```



Interogari multi-relatie : JOIN

Operatorul JOIN combină produsul cartezian, selecția și proiecția.

✓ Θ -JOIN (THETA-JOIN)

- ✓ EQUIJOIN
- ✓ NATURAL JOIN
- ✓ NONEQUIJOIN

✓ SELF JOIN

• OUTER JOIN (compunere externa)

- LEFT OUTER JOIN
- RIGHT OUTER JOIN
- FULL OUTER JOIN

Interogari multi-relatie: OUTER JOIN

Dacă o linie nu satisface condiția de *join*, atunci linia respectivă nu va apare în rezultatul cererii. Pentru a evita această pierdere, în algebra relațională a fost introdus operatorul *outer-join*.

Un *outer-join* (join extern) este reprezentat prin operatorul (+) care este plasat în clauza *WHERE* după numele tabelului ale cărei linii trebuie să nu se piardă din rezultatul cererii.

Semnul (+) poate fi plasat în oricare parte a condiției din clauza *WHERE*, însă nu în ambele părți.

Efectul operatorului (+) este că se generează valori *NULL* pentru coloanele tabelului lângă care apare scris, ori de câte ori tabelul nu are nici o linie care să poată fi reunită cu o linie din celălalt tabel.

Interogari multi-relatie: OUTER JOIN

Tipuri de outer-join $R \bowtie S$:

- **Left outer join** - pe langa tuplurile din equijoin, ia si tuplurile din R care nu au corespondent in S.
- **Right outer join** –pe langa tuplurile din equijoin, ia si tuplurile din S care nu au corespondent in R
- **Full outer join** –pe langa tuplurile din equijoin, ia si tuplurile din R care nu au corespondent in S, si tuplurile din S care nu au corespondent in R.

#FULL OUTER = LEFT OUTER **UNION** RIGHT OUTER
JOIN JOIN JOIN

Interogari multi-relatie: OUTER JOIN

Exemplu: Sa se obtina informatii despre medici si sectii, chiar si despre sectiile care momentan nu au medici.

MEDIC

COD	NUME	ID_SECTIE
100	Anton	1
110	Arsene	1
120	Bart	1
130	Caro	3
140	Craiu	3

SECTIE

ID	DENUMIRE
1	Neurologie
2	Chirurgie
3	Ortopedie

EQUIJOIN

```
SELECT *
FROM MEDIC M, SECTIE S
WHERE M.ID_SECTIE=S.ID;
```

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie

RIGHT OUTER JOIN

```
SELECT *
FROM MEDIC M, SECTIE S
WHERE M.ID_SECTIE(+)=S.ID;
```

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie
NULL	NULL	NULL	2	Chirurgie

Interogari multi-relatie: OUTER JOIN

Exemplu: Sa se obtina informatii despre medici si sectii, chiar si despre sectiile care momentan nu au medici.

MEDIC		
COD	NUME	ID_SECTIE
100	Anton	1
110	Arsene	1
120	Bart	1
130	Caro	3
140	Craiu	3

SECTIE	
ID	DENUMIRE
1	Neurologie
2	Chirurgie
3	Ortopedie

Este un right outer join.

EQUIJOIN

```
SELECT *
FROM MEDIC M, SECTIE S
WHERE M.ID_SECTIE=S.ID;
```

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie

Cum il putem transforma in left outer join?

RIGHT OUTER JOIN

```
SELECT *
FROM MEDIC M, SECTIE S
WHERE M.ID_SECTIE(+) = S.ID;
```

COD	NUME	ID_SECTIE	ID	DENUMIRE
100	Anton	1	1	Neurologie
110	Arsene	1	1	Neurologie
120	Bart	1	1	Neurologie
130	Caro	3	3	Ortopedie
140	Craiu	3	3	Ortopedie
NULL	NULL	NULL	2	Chirurgie

Interogari multi-relatie: OUTER JOIN

Exemplu: Sa se obtina informatii despre medici si sectii, chiar si despre sectiile care momentan nu au medici.

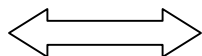
Retinem:

(+) se pune acolo unde vrem sa completeze cu NULL

(+) se pune in **partea opusa fata de unde luam “chiar daca ...”**

Urmatoarele scrieri sunt echivalente:

```
SELECT *  
FROM MEDIC M,SECTIE S  
WHERE M.ID_SECTIE(+)=S.ID;
```



```
SELECT *  
FROM MEDIC M RIGHT OUTER JOIN SECTIE S  
ON (M.ID_SECTIE=S.ID);
```

Exercitii OUTER JOIN – set 5

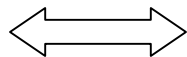
- p) Să se afișeze numele salariaților și numele departamentelor în care lucrează. Se vor afișa și salariații care nu au asociat un departament.
- q) Să se afișeze numele departamentelor și numele salariaților care lucrează în ele. Se vor afișa și departamentele care nu au salariați.
- r) Scrieti o cerere care afiseaza departamentele, chiar și cele fără funcționari, și funcționarii, chiar și cei care nu sunt asigurați nici unui departament.

Exercitii OUTER JOIN – set 5 - Rasp

p) Să se afișeze numele salariaților și numele departamentelor în care lucrează. Se vor afișa și salariații care nu au asociat un departament

REZULTAT=RIGHT_OUTER_JOIN(DEPARTMENTS, EMPLOYEES)

```
SELECT last_name, department_name  
FROM departments d, employees e  
WHERE d.department_id(+) = e.department_id;
```



```
SELECT last_name, department_name  
FROM departments d RIGHT OUTER JOIN employees e  
ON d.department_id = e.department_id;
```

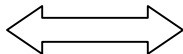
RIGHT
OUTER
JOIN

Exercitii OUTER JOIN – set 5 - Rasp

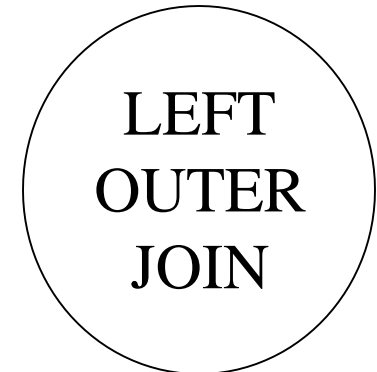
q) Să se afișeze numele departamentelor și numele salariaților care lucrează în ele. Se vor afișa și departamentele care nu au salariați.

REZULTAT=LEFT_OUTER_JOIN(DEPARTMENTS, EMPLOYEES)

```
SELECT department_name, last_name  
FROM   departments d, employees e  
WHERE d.department_id = e.department_id(+);
```



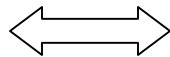
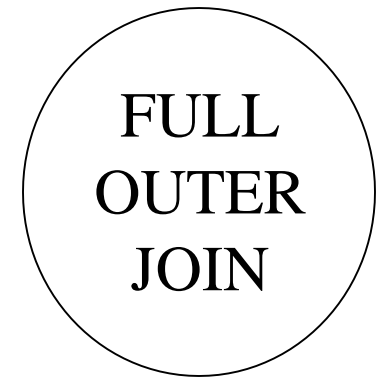
```
SELECT last_name, department_name  
FROM   departments d LEFT OUTER JOIN employees e  
      ON d.department_id = e.department_id;
```



Exercitii OUTER JOIN – set 5 -Rasp

r) Scrieti o cerere care afiseaza departamentele, chiar și cele fără funcționari, și funcționarii, chiar și cei care nu sunt asigurați nici unui departament

```
SELECT employee_id, last_name, department_name
FROM employees e, departments d
WHERE e.department_id = d.department_id(+)
UNION
SELECT employee_id, last_name, department_name
FROM employees e, departments d
WHERE e.department_id(+) = d.department_id;
```



```
SELECT DISTINCT employee_id, last_name, department_name
FROM employees e FULL OUTER JOIN departments d
ON e.department_id = d.department_id;
```

Interogari multi-relatie: OUTER JOIN

Topici avansate:

STAR-SCHEMA OUTER JOIN



Se cer info despre joburi(chiar vacante), angajati si departamente (chiar fara angajati).

```
SELECT J.JOB_ID,E.EMPLOYEE_ID,D.DEPARTMENT_ID
FROM JOBS J, EMPLOYEES E, DEPARTMENTS D
WHERE J.JOB_ID=E.JOB_ID(+)
AND D.DEPARTMENT_ID=E.DEPARTMENT_ID(+);
```

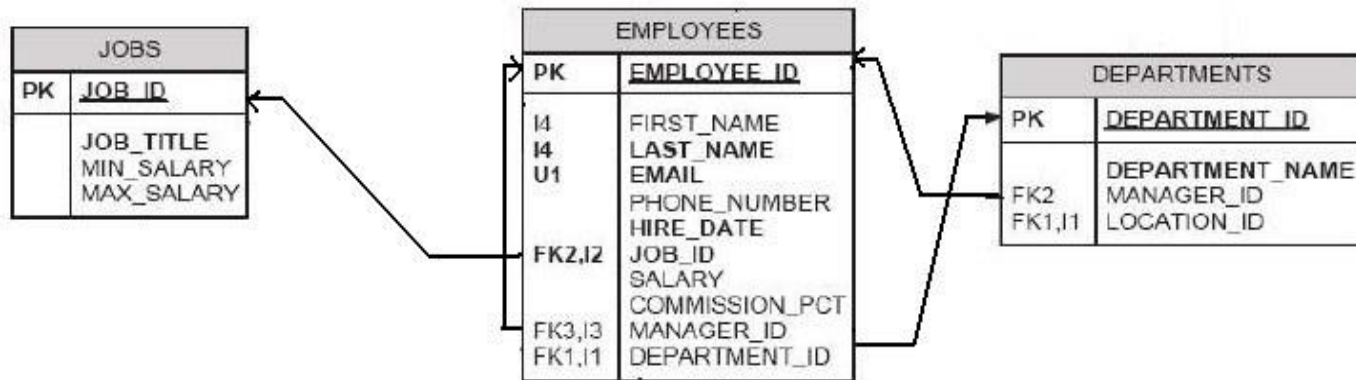
```
ERROR at line 3:
ORA-01417: a table may be outer joined to at most one other table
```

EROARE!

Interogari multi-relatie: OUTER JOIN

Topici avansate:

STAR-SCHEMA OUTER JOIN



Se cer info despre joburi(chiar vacante), angajati si departamente (chiar fara angajati).

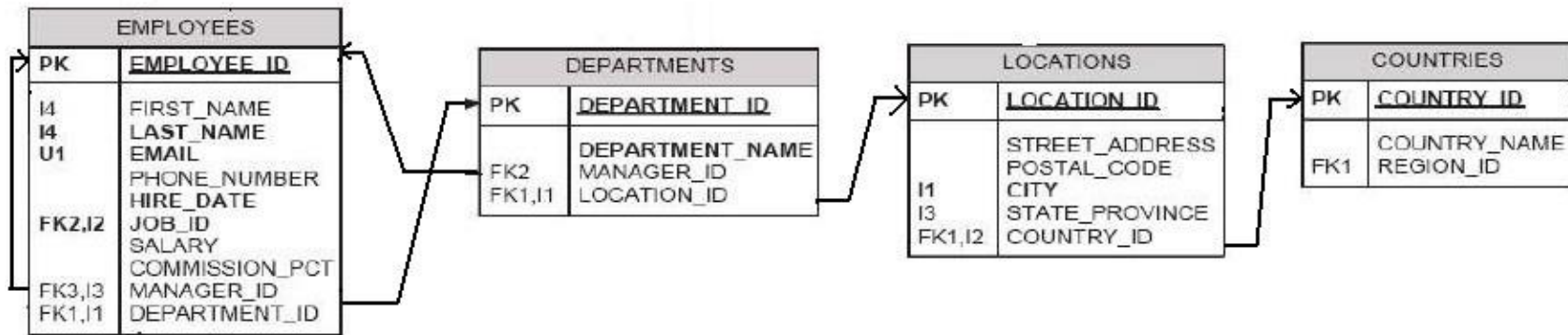
Solutia usoara este folosind subcereri:

```
SELECT aux.JOB_ID,aux.EMPLOYEE_ID,D.DEPARTMENT_ID
FROM (select J.JOB_ID,E.EMPLOYEE_ID,E.DEPARTMENT_ID
      from JOBS J LEFT OUTER JOIN EMPLOYEES E
            ON (J.JOB_ID=E.JOB_ID)) aux
FULL OUTER JOIN DEPARTMENTS D
      ON (D.DEPARTMENT_ID=aux.DEPARTMENT_ID);
```

Interogari multi-relatie: OUTER JOIN

Topici avansate:

INCOMPLETE JOIN TRAIL – este o eroare cand avem un lant de join-uri intr-o schema ierarhica si punem outer join partial pe lant



Sa se afiseze codurile de tara, locatie, departament, angajat inclusiv pt tarile fara locatii

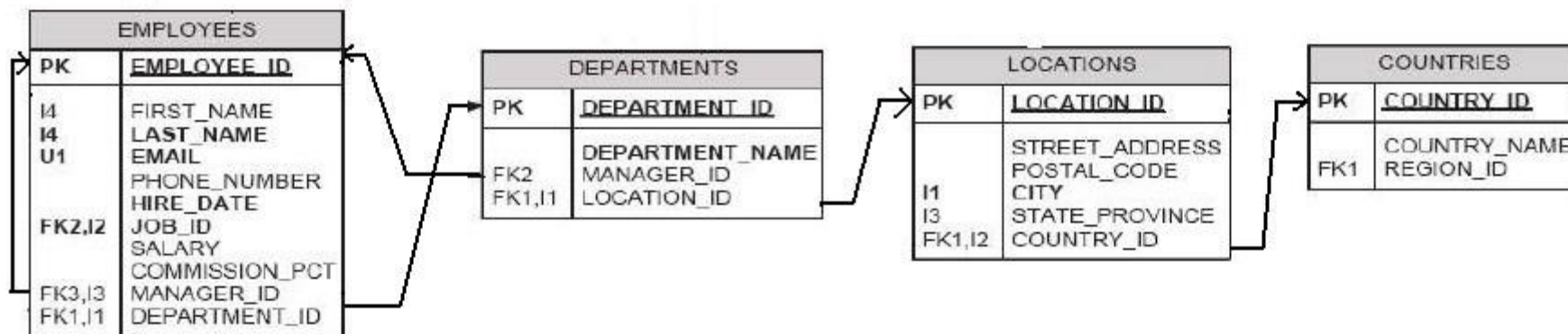
```
SELECT E.EMPLOYEE_ID, D.DEPARTMENT_ID, L.LOCATION_ID, C.COUNTRY_ID
FROM EMPLOYEES E, DEPARTMENTS D, LOCATIONS L, COUNTRIES C WHERE
L.COUNTRY_ID(+)=C.COUNTRY_ID
AND D.LOCATION_ID=L.LOCATION_ID
AND E.DEPARTMENT_ID=D.DEPARTMENT_ID;
```

GRESIT!

Interogari multi-relatie: OUTER JOIN

Topici avansate:

INCOMPLETE JOIN TRAIL – este o eroare cand avem un lant de join-uri intr-o schema ierarhica si punem outer join partial pe lant



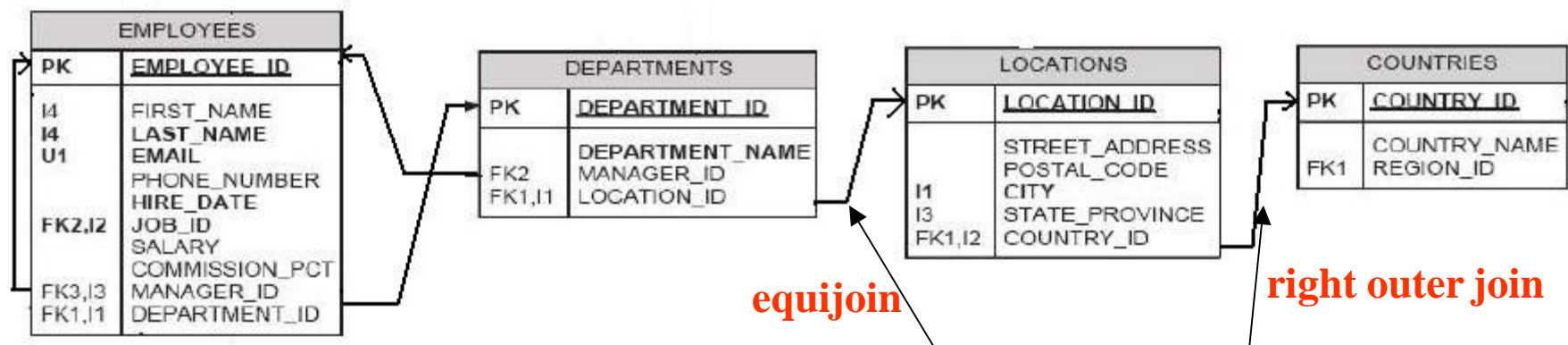
Corect, trebuie propagat outer join pe tot lantul de join-uri:

```
SELECT E.EMPLOYEE_ID,D.DEPARTMENT_ID,L.LOCATION_ID,C.COUNTRY_ID
FROM EMPLOYEES E, DEPARTMENTS D, LOCATIONS L, COUNTRIES C WHERE
L.COUNTRY_ID(+)=C.COUNTRY_ID
AND D.LOCATION_ID(+)=L.LOCATION_ID
AND E.DEPARTMENT_ID(+)=D.DEPARTMENT_ID;
```

Interogari multi-relatie: OUTER JOIN

Topici avansate:

COMBINAREA EQUIJOINS SI OUTER JOINS - cand intr-o cerere avem si outer join si equijoins, este bine sa le demarcam clar prin subcereri in clauza FROM (view-uri inline)



Numele tarilor(chiar daca nu au locatii), iar in cadrul tarilor codul locatiilor si codul departamentelor din locatii numai pentru locatiile care au departamente.

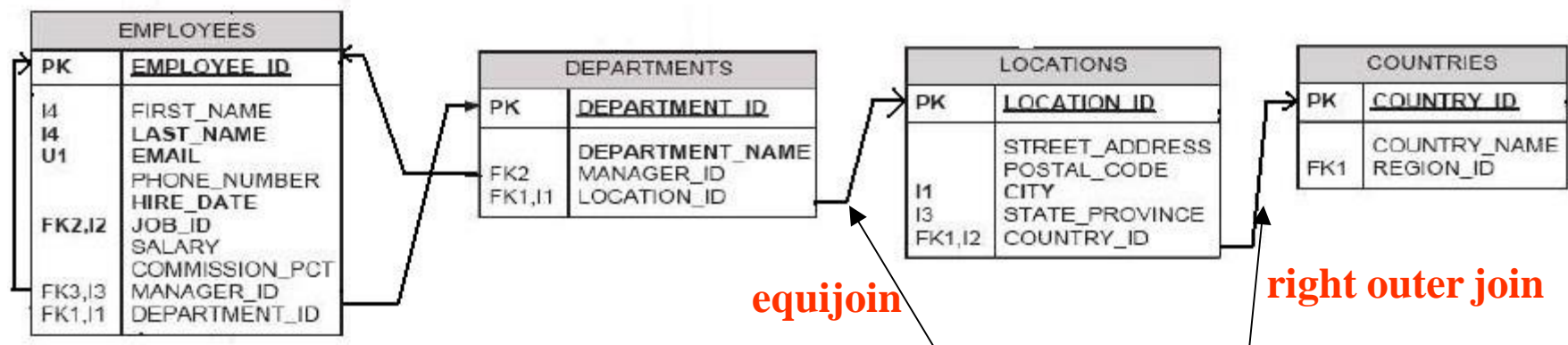
```
SELECT C.COUNTRY_ID,L.LOCATION_ID,D.DEPARTMENT_ID
FROM DEPARTMENTS D, LOCATIONS L, COUNTRIES C
WHERE L.COUNTRY_ID(+)=C.COUNTRY_ID
AND D.LOCATION_ID=L.LOCATION_ID;
```

INCOMPLETE
JOIN TRAIL!

Interogari multi-relatie: OUTER JOIN

Topici avansate:

COMBINAREA EQUIJOINS SI OUTER JOINS - cand intr-o cerere avem si outer join si equijoins, este bine sa le demarcam clar prin subcereri in clauza FROM (view-uri inline)



Solutia usoara este folosind SQL3, in care delimitam prin paranteze ordinea:

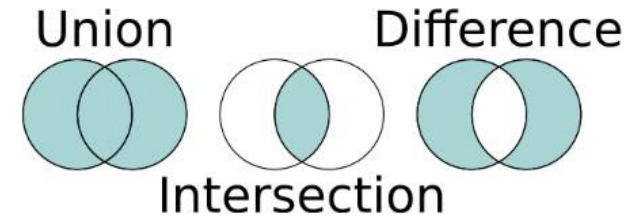
```
SELECT C.COUNTRY_ID,L.LOCATION_ID,D.DEPARTMENT_ID
FROM ( DEPARTMENTS D JOIN LOCATIONS L
      ON (D.LOCATION_ID=L.LOCATION_ID) )
  RIGHT OUTER JOIN COUNTRIES C
    ON (L.COUNTRY_ID=C.COUNTRY_ID);
```

LAB: OBIIECTIVE

✓ Interogari multi-relatie

- Operatori pe multimi
- Subcereri nesincronizate (necorelate)

Operatori pe multimi



Operatorul UNION

Reuniunea a două relații R și S este mulțimea tuplurilor aparținând fie lui R, fie lui S, fie ambelor relații.

Varianta UNION ALL nu elimina duplicatele.

Operatorul DIFFERENCE (in Oracle implemetat **MINUS**)

Diferența a două relații R și S este mulțimea tuplurilor care aparțin lui R, dar nu aparțin lui S. Diferența este o operație binară necomutativă care permite obținerea tuplurilor ce apar numai într-o relație.

Operatorul INTERSECT

Intersecția a două relații R și S este mulțimea tuplurilor care aparțin și lui R și lui S.

Operatori pe mulțimi

Comenzile SELECT, care intervin în cereri ce conțin operatori pe mulțimi, trebuie să satisfacă anumite condiții:

- toate comenzile SELECT trebuie să aibă același număr de coloane;
- opțiunea DISTINCT este implicită** (excepție UNION ALL);
- numele coloanelor sunt cele din prima comandă SELECT;
- dimensiunea coloanei implicit este cea mai mare dintre cele două coloane;
- sunt admise combinații de forma:

SELECT1 UNION SELECT2 INTERSECT SELECT3 și ordinea de execuție este de la stânga la dreapta;

SELECT1 UNION (SELECT2 INTERSECT SELECT3) și ordinea este dată de paranteze.

Exercitii Operatori pe multimi – set 6 -Rasp

- s) Se cer codurile departamentelor al căror nume conține șirul “re” **sau** în care lucrează angajați având codul job-ului “SA_REP”.
- t) Sa se obtina codurile departamentelor in care nu lucreaza nimeni (nu este introdus nici un salariat in tabelul employees).
- u) Se cer codurile departamentelor al căror nume conține șirul “re” **și** în care lucrează angajați având codul job-ului “HR_REP”.

Exercitii Operatori pe multimi – set 6- Rasp

- s) Se cer codurile departamentelor al căror nume conține șirul “re” sau în care lucrează angajați având codul job-ului “SA_REP”.

```
SELECT department_id "Cod departament"
```

```
FROM employees
```

```
WHERE job_id='SA_REP'
```

UNION

```
SELECT department_id
```

```
FROM departments
```

```
WHERE department_name LIKE '%re%';
```

- t) Sa se obtina codurile departamentelor in care nu lucreaza nimeni (nu este introdus nici un salariat in tabelul employees).

```
SELECT department_id "Cod departament"
```

```
FROM departments
```

MINUS

```
SELECT department_id
```

```
FROM employees;
```

Exercitii Operatori pe multimi – set 6 -Rasp

u) Se cer codurile departamentelor al căror nume conține șirul “re” și în care lucrează angajați având codul job-ului “HR_REP”.

```
SELECT department_id "Cod departament"  
FROM employees  
WHERE job_id='HR_REP'  
INTERSECT  
SELECT department_id  
FROM departments  
WHERE department_name LIKE '%re%';
```

LAB : OBIECTIVE

- ✓ Interogari multi-relatie
- ✓ Operatori pe multimi
- Subcereri nesincronizate (necorelate)

Subcereri nesincronizate (necorelate)

O subcerere este o comandă SELECT încapsulată într-o clauză a altei instrucțiuni SQL, numită instrucțiune „părinte“. Utilizând subcereri, se pot construi interogări complexe pe baza unor instrucțiuni simple. Subcererile mai sunt numite instrucțiuni SELECT imbricate sau interioare.

Subcererea returnează o valoare care este utilizată de către instrucțiunea „părinte“. Utilizarea unei subcereri este echivalentă cu efectuarea a două cereri secvențiale și utilizarea rezultatului cererii interne ca valoare de căutare în cererea externă (principală).

Subcereri nesincronizate (necorelate)

Astazi studiem subcererile nesincronizate (necorelate) cu cererea parinte.

```
SELECT   lista_select  
FROM     nume_tabel  
WHERE     expresie operator (SELECT lista_select  
                                FROM   nume_tabel);
```

- cererea internă este executată prima și determină o valoare (sau o mulțime de valori);
- cererea externă se execută o singură dată, utilizând valorile returnate de cererea internă.

Daca subcererea intoarce o singura valoare, se va folosi in cererea parinte unul din operatorii =, <>, >, <, <=, >=

Daca subcererea intoarce o multime de valori, se va folosi in cererea parinte unul din operatorii IN, NOT IN, ANY, ALL.

Exercitii Subcereri necorelate – set 7

- v) Folosind subcereri, să se afișeze numele și data angajării pentru salariații care au fost angajați după Gates. În firma un singur angajat se numeste Gates.

- w) Folosind subcereri, scrieți o cerere pentru a afișa numele și salariul pentru toți colegii (din același departament) lui King. În firma exista doi angajati cu numele King, insa in departamente diferite.

Exercitii Subcereri necorelate – set 7 - Rasp

- v) Folosind subcereri, să se afișeze numele și data angajării pentru salariații care au fost angajați după Gates. În firma un singur angajat se numeste Gates.

```
SELECT last_name, hire_date  
FROM EMPLOYEES  
WHERE hire_date > ( SELECT hire_date  
                     FROM EMPLOYEES  
                     WHERE last_name='Gates');
```

- w) Folosind subcereri, scrieți o cerere pentru a afișa numele și salariul pentru toți colegii (din același departament) lui King. În firma există doi angajați cu numele King, însă în departamente diferite.

```
SELECT last_name, salary  
FROM EMPLOYEES  
WHERE department_id IN ( SELECT department_id  
                        FROM EMPLOYEES  
                        WHERE last_name='King');
```

DEPARTMENT_ID

90
80



Subcereri nesincronizate (necorelate)

Daca subcererea intoarce o multime de valori, se va folosi in cererea parinte unul din operatorii IN, NOT IN, ANY, ALL.

```
SELECT lista_select  
FROM nume_tabel  
WHERE expresie operator (SELECT lista_select  
FROM nume_tabel);
```

WHERE col1 = ANY (SELECT ...) \iff WHERE col1 IN (SELECT ...)

WHERE col1 > ANY (SELECT ...) \iff mai mare ca minimul;

WHERE col1 < ANY (SELECT ...) \iff mai mic ca maximul;

WHERE col1 > ALL (SELECT ...) \iff mai mare ca maximul;

WHERE col1 < ALL (SELECT ...) \iff mai mic ca minimul;

WHERE col 1 != ALL (SELECT ...) \iff WHERE col1 NOT IN (SELECT ...)

Exercitii Subcereri necorelate – set 8

- x) Scrieti o cerere pentru a afisa angajatii care castiga mai mult decat oricare functionar (codul de job contine şirul “CLERK”). Sortati rezultatele dupa salariu, in ordine descrescatoare.

Exercitii Subcereri necorelate – set 8- Rasp

- x) Scrieti o cerere pentru a afisa angajatii care castiga mai mult decat oricare functionar (job-ul conține șirul “CLERK”). Sortati rezultatele dupa salariu, in ordine descrescatoare.


```
SELECT last_name, salary
FROM employees
WHERE salary > ALL(SELECT salary
                    FROM employees
                    WHERE job_id LIKE '%CLERK%')
ORDER BY 2 DESC;
```

Ce rezultat este returnat dacă se înlocuiește “ALL” cu “ANY”?

Exercitii Subcereri necorelate – set 8- Rasp

- x) Scrieti o cerere pentru a afisa angajatii care castiga mai mult decat oricare functionar (job-ul conține șirul “CLERK”). Sortati rezultatele dupa salariu, in ordine descrescatoare.

```
SELECT last_name, salary
FROM employees
WHERE salary > ANY(SELECT salary
                    FROM employees
                    WHERE job_id LIKE '%CLERK%')
ORDER BY 2 DESC;
```



Ce rezultat este returnat dacă se înlocuiește “ALL” cu “ANY”?
Angajatii care castiga mai mult decat cel mai prost platit clerk.