

Tut 7Algebra relationalăOperatori

- tradicionali pe multimi:
 - UNION, INTERSECT, PRODUCT, DIFFERENCE

relationali speciali:

- PROJECT, SELECT, JOIN, DIVISION

param, multimi

- ① Operatorul PROJECT ($\text{PROJECT}(R, \overline{A_1, \dots, A_m})$)
- ↳ elibera anumite attribute ale unei relații producând o nouă multime, pe verticală "a acesteia.
(selectarea deoră anumite coloane ale unei tabl.)

Exemplu

Să se obțină o listă care conține numele, prenumele și numărul de telefon al angajaților.

Algebra relatională

Rezultat = $\text{PROJECT}(\text{EMPLOYEES}, \text{FIRST-NAME}, \text{LAST-NAME}, \text{PHONE-NUMBER})$

SQL

Select FIRST-NAME, LAST-NAME, PHONE-NUMBER
FROM EMPLOYEES;

- ② Operatorul SELECT (SELECT(R, condition))

Selectarea datele din relație care îndeplinește o condiție dată.

Exemplu

Să se obțină informații complete despre angajat care lucra la jobul cu codul AC-ACCOUNT

Algebra relatională

Rezultat = $\text{SELECT}(\text{EMPLOYEES}, \text{job-id} = 'AC-ACCOUNT')$

SQL

SELECT
FROM EMPLOYEES
WHERE job-id = 'AC-ACCOUNT';

- ③ Join

NATURAL JOIN

JOIN

SEMI JOIN

OUTER JOIN

NATURAL JOIN (JOIN(R,S))

↳ presupune un form între 2 relații prin egalitatea
atributelor cu aceeași nume.

Exemplu

Să se obțină toate informațiile despre departamente
și locațiile unde se găsesc.

Algebra relatională

rezultat = JOIN(DEPARTMENTS, LOCATIONS)

SQL

```
SELECT *  
FROM DEPARTMENTS NATURAL JOIN LOCATIONS
```

A-JOIN (JOIN(R, S, condition))

↳ presupune realizarea unui JOIN între 2 relații
cu o condiție

Exemplu

Să se obțină toate info despre departamentele și
angajații care lucrează în ele, dar doar pebra
cei care au salariul < 20000.

Algebra relatională

rezultat = JOIN(EMPLOYEES, DEPARTMENTS, SALARY < 20000)

SQL

```
SELECT *  
FROM EMPLOYEES JOIN DEPARTMENTS USING(department_id)  
WHERE SALARY < 20000;
```

○ Semi-Join ($\text{SEMIJOIN}(R, S)$)
 ↳ presupune un form între 2 relații de concuranță
 atribuibile unei anumite relații participante.

Exemplu

Afișati numele, prenumele și telefonul angajatorilor cu job-urile
 al căror salariu maxim < 60000

Algebra relațională

$R_1 = \text{JOIN}(\text{EMPLOYEES}, \text{JOBS}, \text{MAX-SALARY} < 60000)$
 Rezultat = $\text{PROJECT}(R_1, \text{FIRST-NAME}, \text{LAST-NAME}, \text{PHONE-NUMBER})$

SQL

`SELECT FIRST-NAME, LAST-NAME, PHONE-NUMBER
 FROM EMPLOYEES JOIN JOBS USING (JOB-ID)
 WHERE MAX-SALARY < 60000`

○ Outer-Join

Left-Join
 Right-Join
 Full-Join

Left-Join

Afișați numele, prenumele, numele departamentelor din care
 face parte toți angajații.

Algebra relațională:

$R_1 = \text{PROJECT}(\text{EMPLOYEES}, \text{FIRST-NAME}, \text{LAST-NAME}, \text{EMAIL},$
 $\text{DEPARTMENT-ID})$

$S = \text{PROJECT}(\text{DEPARTMENTS}, \text{DEPARTMENT-ID}, \text{DEPARTMENT-NAME})$
 Rezultat = LEFT OUTER JOIN (R_1, S)

RIGHT JOIN

Afiseaza numele angajatilor si detalii despre joburile din stoc

Algebra relationala:

$R = \text{PROJECT}(\text{EMPLOYEES}, \text{FIRST_NAME})$

$\text{REZULTAT} = \text{RIGHT OUTER JOIN}(R, \text{JOB_HISTORY})$

SQL

```
SELECT DISTINCT FIRST_NAME  
FROM EMPLOYEES RIGHT JOIN JOB_HISTORY USING  
(EMPLOYEE_ID);
```

Full JOIN

Să se afifice numele departamentelor, orasele și adresa locației în care se află acestea; să se afifice toate locurile departamentele

Algebra relationala

$R = \text{PROJECT}(\text{DEPARTMENTS}, \text{DEPARTMENT_NAME})$

$S = \text{PROJECT}(\text{LOCATIONS}, \text{CITY}, \text{STREET_ADDRESS})$

$\text{REZULTAT} = \text{FULL OUTER JOIN}(R, S)$

SQL

SELECT DEPARTMENT-NAME, CITY, STREET-ADDRESS

FROM DEPARTMENTS FULL OUTER JOIN LOCATIONS

ON DEPARTMENTS.LOCATION-ID = LOCATIONS.LOCATION-ID

(4) UNION (UNION(R,S))

↳ reunirea a 2 relatiilor

Afișați numele & prenumele angajaților care au salariul < 5000
care lucresă în departamentul Shipping

Alg. rel
 $R = \text{SELECT}(\text{EMPLOYEES}, \text{SALARY} < 5000);$

$S_1 = \text{JOIN}(\text{EMPLOYEES}, \text{DEPARTMENTS});$

$S_2 = \text{SELECT}(S_1, \text{DEPARTMENT-NAME} = 'Shipping');$

$S_3 = \text{PROJECT}(S_2, \text{FIRST-NAME}, \text{LAST-NAME});$

$R_1 = \text{PROJECT}(R, \text{---}, \text{---});$

Rezultat = $\text{UNION}(R_1, S_3);$

SQL

```
SELECT FIRST-NAME, LAST-NAME
FROM EMPLOYEES
WHERE salary < 5000
```

UNION

```
SELECT FIRST-NAME, LAST-NAME
FROM EMPLOYEES
JOIN DEPARTMENTS USING (department-id)
WHERE department-name = 'Shipping'
```

(5) DIFFERENCE (DIFFERENCE(R,S))

Afișați ~~numele departamenteelor cu id-ul dep~~ care au tot > 200 dar
care nu au angajați.

Algoritm relational

$S = \text{PROJECT}(\text{DEPARTMENTS}, \text{DEPARTMENT-NAME}, \text{department-id});$

$S_1 = \text{SELECT}(S, \text{department-id} > 200);$

$R = \text{JOIN}(\text{EMPLOYEES}, \text{DEPARTMENTS});$

$R_1 = \text{PROJECT}(R, \text{DEPARTMENT-NAME}, \text{department-id});$

Rezultat = $\text{DIFFERENCE}(S_1, R_1);$

SQL
SELECT department-name, department-id
FROM departments
WHERE department-id > 200

MINUS
SELECT department-name, department-id
FROM employees
JOIN departments using(department-id);

① INTERSECT
Se va afisa toturile angajatorilor care lucrau la foizul 'Accountant' si care nu erau in departamentul cu id-ul 100.

Algebra relationala

$R = \text{SELECT}(\text{EMPLOYEES}, \text{DEPARTMENT-ID} = 100);$
 $R_1 = \text{PROJECT}(R, \text{EMPLOYEE-ID});$
 $S = \text{JOIN}(\text{EMPLOYEES}, \text{JOBS}),$
 $S_1 = \text{SELECT}(S, \text{JOB-TITLE} = 'Accountant');$
 $S_2 = \text{PROJECT}(S, \text{EMPLOYEE-ID});$
Resultat = INTERSECT(R_1, S_2);

SQL

SELECT employee-id
FROM employees
JOIN jobs ON (employees.job-id = jobs.job-id)
WHERE job-title = 'Accountant'

INTERSECT
select employee-id
from employees
where department-id = 100;

(1) PRODUCT

↳ produs cartesian dintre 2 relata

Realizat, produsul cartesian dintre multimea
id-urilor departamente și multimea id-urilor locațiilor

$R_1 = \text{PROJECT}(\text{DEPARTMENTS}, \text{DEPARTMENT-ID});$
 $S = \text{---} (\text{LOCATIONS}, \text{LOCATION-ID});$
rezultat = PRODUCT(R_1, S)

SQL

SELECT d.department_id, l.location_id
FROM DEPARTMENTS d, LOCATIONS l;