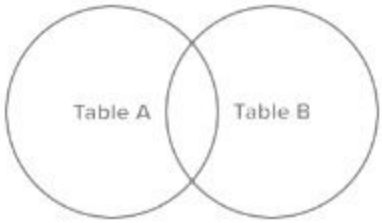
1. TEORIE

JOIN

Join-ul 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.

Vom exemplifica operația de **JOIN** pe baza a două tabele A și B:



Într-o instrucțiune SELECT care unește tabele prin operația de join, se recomandă ca numele coloanelor să fie precedate de numele sau alias-urile tabelelor pentru claritate și pentru îmbunătățirea timpului de acces la baza de date. Dacă același nume de coloană apare în mai mult de două tabele, atunci numele coloanei se prefixează **obligatoriu** cu numele sau alias-ul tabelului corespunzător.

Pentru a realiza un join între **n tabele**, va fi nevoie de cel puțin **n – 1 condiții de join**.

OBS: Un alias poate fi utilizat pentru a califica denumirea unei coloane. Calificarea unei coloane cu numele sau alias-ul tabelului se poate face :

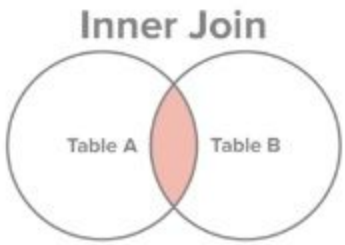
- opțional, pentru claritate și pentru îmbunătățirea timpului de acces la baza de date;
- obligatoriu, ori de câte ori există o ambiguitate privind sursa coloanei

Tipuri de join:

1. **Equijoin:**
   ➔ folosește operatorul egalitate pentru compararea valorilor
2. **Nonequijoin**
   ➔ condiția de join conține alți operatori decât operatorul egalitate

În cadrul acestora avem următoarele tipuri:

Inner join



➔ corespunde situației în care valorile de pe coloanele ce apar în condiția de join trebuie să fie egale

Există mai multe variante pentru a scrie operația de ***inner join***:

```
SELECT nume_coloane
FROM tabel1 [INNER] JOIN tabel2
      ON (tabel1.nume_coloana = tabel2.nume_coloana);
```

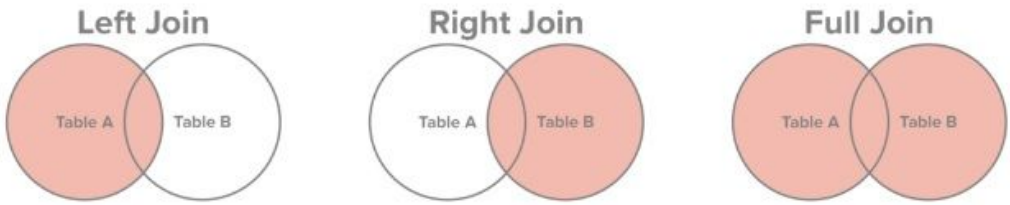
Dacă numele coloanelor din condiția de join din cele 2 tabele sunt identice se poate scrie:

```
SELECT nume_coloane
FROM tabel1 [INNER] JOIN tabel2 USING (nume_coloana);
```

Coloanele referite în clauza USING trebuie să nu conțină calificatori (să nu fie precedate de nume de tabele sau alias-uri) în nicio apariție a lor în instrucțiunea SQL.

Outer join: left, right, full

➔ un outer join este utilizat pentru a obține în rezultat şi înregistrările care nu satisfac condiția de join



LEFT JOIN: Sunt returnate toate rândurile tabelului din stânga cu corespondenţele din dreapta. Dacă nu există corespondent se completează câmpul respectiv cu valoarea *null*.

RIGHT JOIN: Sunt returnate toate rândurile tabelului din dreapta cu corespondenţele din stânga. Dacă nu există corespondent se completează câmpul respectiv cu valoarea *null*.

FULL JOIN: LEFT JOIN + RIGHT JOIN

Operatorul pentru outer join este semnul plus inclus între paranteze (+), care se plasează în acea parte a condiției de join care este deficitară în informație. Efectul acestui operator este de a uni liniile tabelului care nu este deficitar în informație şi cărora nu le corespunde nicio linie în celălalt tabel cu o linie cu valori null. Operatorul (+) poate fi plasat în orice parte a condiției de join, dar nu în ambele părți.

```
SELECT nume_coloane
FROM tabel1 LEFT | RIGHT | FULL JOIN tabel2
      ON (tabel1.nume_coloana = tabel2.nume_coloana);

SELECT nume_coloane
FROM tabel1 LEFT | RIGHT | FULL JOIN tabel2
      USING (nume_coloana);
```

Right join de mai sus folosind operatorul (+):

```
SELECT nume_coloane
FROM tabel1 JOIN tabel2
      ON (tabel1.nume_coloana (+) = tabel2.nume_coloana);
```

Left join de mai sus folosind operatorul (+):

```
SELECT nume_coloane
FROM tabel1 JOIN tabel2
      ON (tabel1.nume_coloana = tabel2.nume_coloana (+));
```

Natural join

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.

Coloanele având același nume în cele două tabele trebuie să nu fie precedate de numele sau alias-ul tabelului corespunzător.

```
SELECT nume_coloane
FROM tabel1 NATURAL JOIN tabel2;
```

Cross join

- ➔ face produs cartezian între tabele
- ➔ echivalent cu , in FROM

```
SELECT nume_coloane
FROM tabel1 CROSS JOIN tabel2;
```

Exemple:

Considerăm următoarele tabelele:

Customers:

customer_id	first_name	last_name	email	address	city	state	zip
1	George	Washington	gwashington@usa.gov	3200 Mt Vernon Hwy	Mount Vernon	VA	22121
2	John	Adams	jadams@usa.gov	1250 Hancock St	Quincy	MA	02169
3	Thomas	Jefferson	tjefferson@usa.gov	931 Thomas Jefferson Pkwy	Charlottesville	VA	22902
4	James	Madison	jmadison@usa.gov	11350 Constitution Hwy	Orange	VA	22960
5	James	Monroe	jmonroe@usa.gov	2050 James Monroe Parkway	Charlottesville	VA	22902

Orders:

order_id	order_date	amount	customer_id
1	07/04/1776	$234.56	1
2	03/14/1760	$78.50	3
3	05/23/1784	$124.00	2
4	09/03/1790	$65.50	3
5	07/21/1795	$25.50	10
6	11/27/1787	$14.40	9

customer_id este cheie primară în tabelul *Customers* şi cheie externă în tabelul *Orders*.

Inner Join

```
SELECT first_name, last_name, order_date, order_amount
FROM customers c JOIN orders o
      ON (c.customer_id = o.customer_id)
```

first_name	last_name	order_date	order_amount
George	Washington	07/4/1776	$234.56
John	Adams	05/23/1784	$124.00
Thomas	Jefferson	03/14/1760	$78.50
Thomas	Jefferson	09/03/1790	$65.50

Observăm că liniile pentru care nu s-a găsit o potrivire între tabele au fost ignorate (clienţii cu codurile 4 şi 5 din *Customers*, respectiv comenzile plasate de clienţii 9 şi 10).

Left Join

```
SELECT first_name, last_name, order_date, order_amount
FROM customers c LEFT JOIN orders o
      ON (c.customer_id = o.customer_id)
```

first_name	last_name	order_date	order_amount
George	Washington	07/04/1776	$234.56
John	Adams	05/23/1784	$124.00
Thomas	Jefferson	03/14/1760	$78.50
Thomas	Jefferson	09/03/1790	$65.50
James	Madison	NULL	NULL
James	Monroe	NULL	NULL

Observăm că au fost luate toate intrările din tabelul *Customers,* împreună cu perechile lor din *Orders,* iar pentru clienţii care nu au plasat comenzi s-a completat tabelul cu valoarea null.

Right Join

```
SELECT first_name, last_name, order_date, order_amount
FROM customers c RIGHT JOIN orders o
      ON (c.customer_id = o.customer_id)
```

first_name	last_name	order_date	order_amount
George	Washington	07/04/1776	$234.56
Thomas	Jefferson	03/14/1760	$78.50
John	Adams	05/23/1784	$124.00
Thomas	Jefferson	09/03/1790	$65.50
NULL	NULL	07/21/1795	$25.50
NULL	NULL	11/27/1787	$14.40

Observăm că au fost luate toate comenzile, iar pentru cele care nu au corespondent în tabelul clienţilor s-a completat tabelul cu valoarea null.

Full Join

```
SELECT first_name, last_name, order_date, order_amount
FROM customers c FULL JOIN orders o
      ON (c.customer_id = o.customer_id)
```

first_name	last_name	order_date	order_amount
George	Washington	07/04/1776	$234.56
Thomas	Jefferson	03/14/1760	$78.50
John	Adams	05/23/1784	$124.00
Thomas	Jefferson	09/03/1790	$65.50
NULL	NULL	07/21/1795	$25.50
NULL	NULL	11/27/1787	$14.40
James	Madison	NULL	NULL
James	Monroe	NULL	NULL

În acest caz au fost luaţi toţi clienţii şi toate comezile iar unde nu s-a putut face o potrivire s-a completat cu valoarea null.

Imagini şi tabele*:* *http://www.sql-join.com*

---

2. EXERCIŢII

1. Să se afişeze numele salariatului, codul şi numele departamentului pentru toți angajații care lucrează în departamente.
2. Să se afişeze numele salariatului, codul şi numele departamentului pentru toți angajații.
3. Să se listeze codurile şi denumirile job-urilor care există în departamentul 30.
4. Să se afişeze numele angajatului, numele departamentului şi oraşul pentru toți angajații care câştigă comision.
5. Să se afişeze numele salariatului şi numele departamentului pentru toți salariații care au litera A inclusă în nume.

6. Să se afişeze codul angajatului şi numele acestuia, împreună cu numele şi codul şefului său direct. Se vor eticheta coloanele Ang#, Angajat, Mgr#, Manager.
7. Să se modifice cererea anterioară pentru a afişa toți salariații, inclusiv cei care nu au şef. Să se folosească ambele metode învățate.
8. Scrieți o cerere care afişează numele angajatului, codul departamentului în care acesta lucrează şi numele colegilor săi de departament. Se vor eticheta coloanele corespunzător şi se vor sorta rezultatele după codul departamentului.
9. Creați o cerere prin care să se afişeze numele, codul job-ului, titlul job-ului, numele departamentului şi salariul angajaților. Se vor include şi angajaţii al căror departament nu este cunoscut.
10. Să se afişeze id-ul, codul departamentului, numele departamentului, numele şi job-ul tuturor angajaţilor din departamentele al căror nume conține şirul 'ti'. De asemenea, se va lista salariul angajaților, în formatul "$99,999.00". Rezultatul se va ordona alfabetic după numele departamentului, şi în cadrul acestuia, după numele angajaților.
11. Să se afişeze numele angajaţilor, numărul departamentului, numele departamentului, oraşul şi job-ul tuturor salariaţilor al căror departament este localizat în Oxford.
12. Să se afişeze numele salariaților şi numele departamentelor în care lucrează. Se vor afişa şi salariații care nu au asociat un departament.
13. Să se afişeze numele departamentelor şi numele salariaților care lucrează în ele. Se vor afişa şi departamentele care nu au salariați.
14. Să se afişeze numele angajaţilor impreună cu regiunile în care lucrează. Se vor eticheta coloanele corespunzător, iar numele va fi concatenat cu prenumele.