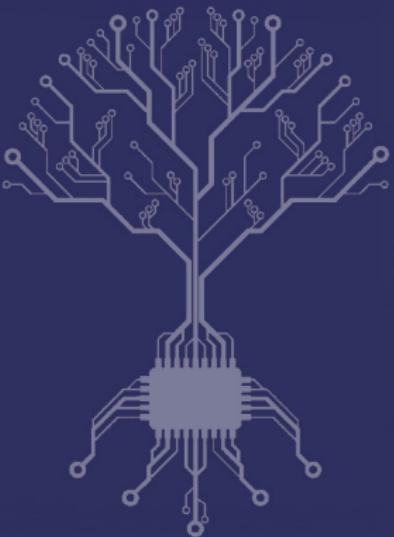




وزارة التربية الوطنية
والتعليم الأولي والرياضة
+٣٩٠٣٤٠٢٨٧٤ | ٢٠٢٤
لـ ٣٠٣٢٨ لـ ٣٠٣٢٧ لـ ٣٠٣٢٦

BASE DE DONNÉES (3)

INTERROGATION (SUITE)



JOINTURE

Joindre deux tables, qui sont en relation à l'aide d'une clé primaire et d'une clé étrangère, revient à créer une nouvelle table temporaire qui combine les lignes qui ont des valeurs égales dans la colonne commune des deux tables.

Syntaxe

```
SELECT ... FROM Table1 JOIN Table2 ON Table1.colonne1=Table2.colonne2;
```

Ou bien

```
SELECT ... FROM Table1, Table2 WHERE Table1.colonne1=Table2.colonne2;
```

Exemple

■ Soit deux tables :

- ▶ *student(idstudent, firstname, lastname, field, average, #idschool)*
- ▶ *school(idschool, name, city)*

<i>idstudent</i>	<i>firstname</i>	<i>lastname</i>	<i>field</i>	<i>average</i>	<i>idschool</i>
251423530	Ahmad	Bachir	Economics	15	CP0012
251423561	Omar	Alaoui	Economics	14	CP0012
251677001	Fatima	Mohammadi	Mathematics	19	CP0461



<i>idschool</i>	<i>name</i>	<i>city</i>
CP0012	Omar Alkhiyyam	Rabat
CP0461	Charif Alidrissi	Taza

==

<i>idstudent</i>	<i>firstname</i>	<i>lastname</i>	<i>field</i>	<i>average</i>	<i>idschool</i>	<i>name</i>	<i>city</i>
251423530	Ahmad	Bachir	Economics	15	CP0012	Omar Alkhiyyam	Rabat
251423561	Omar	Alaoui	Economics	14	CP0012	Omar Alkhiyyam	Rabat
251677001	Fatima	Mohammadi	Mathematics	19	CP0461	Charif Alidrissi	Taza

Exemple

- Pour afficher les moyennes des étudiants dont l'école est '*Omar Alkhayyam*' une jointure est nécessaire car la moyenne de l'étudiant et le nom de l'école se trouvent dans des tables différentes :
 - ▶ `SELECT student.average FROM student JOIN school ON student.idschool=school.idschool WHERE school.name = 'Omar Alkhayyam' ;`
 - ▶ ou `SELECT student.average FROM student, school WHERE student.idschool=school.idschool AND school.name = 'Omar Alkhayyam' ;`

REQUÊTE IMBRIQUÉE

Une requête imbriquée est une sous-requête exécutée à l'intérieur d'une autre requête SQL. Elle remplace souvent une constante au sein d'une clause WHERE ou HAVING.

Les sous-requêtes sont souvent utilisées avec l'instruction SELECT. Toutefois, on peut également les utiliser dans une instruction INSERT, UPDATE, DELETE ou dans une autre sous-requête.

Syntaxe

SELECT . . . FROM . . . WHERE . . . opérateur (SELECT . . . FROM . . .);

Opérateur peut être :

- Si la requête renvoie un seul résultat :
 - ▶ Un opérateur simple : $=, <, >, !=, <=, >=$
- Si elle renvoie plusieurs résultats :
 - ▶ IN
NOT IN
 - ▶ EXISTS (renvoie *vrai* si la sous requête renvoie au moins une ligne)
NOT EXISTS (renvoie *vrai* si la sous requête ne renvoie aucune ligne)

Exemples

- Afficher les étudiants dont la note est supérieure ou égale à la moyenne des notes de tous les étudiants :
 - ▶ `SELECT * FROM student WHERE average >= (SELECT AVG(average) FROM student);`
- Afficher l'étudiant qui a obtenu la meilleure note :
 - ▶ `SELECT * FROM student WHERE average = (SELECT MAX (average) FROM student);`

Exemple

■ Soit les tables :

- ▶ student (idstudent, firstname, lastname, field, average, city)
- ▶ book (ISBN, title, author)
- ▶ borrow (#idstudent, #ISBN, date)

■ Afficher les étudiants qui ont emprunté un livre :

- ▶ SELECT student.idstudent, firstname, lastname FROM student WHERE idstudent IN (SELECT idstudent FROM borrow) ;

ou bien

- ▶ SELECT student.idstudent, firstname, lastname FROM student WHERE EXISTS (SELECT * FROM borrow WHERE student.idstudent=borrow.idstudent) ;