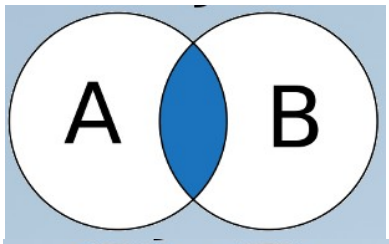
TYPE DE JOINTURES

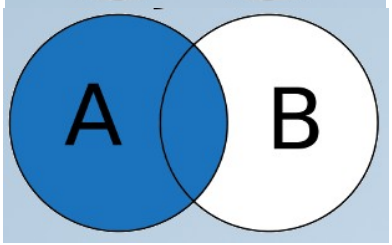
[INNER] JOIN
(jointure interne)

```
SELECT *
FROM A
[INNER] JOIN B ON A.key = B.key;
```



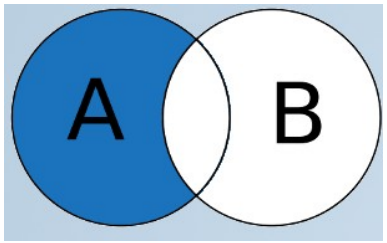
LEFT JOIN
(jointure externe)

```
SELECT *
FROM A
LEFT JOIN B ON A.key = B.key;
```



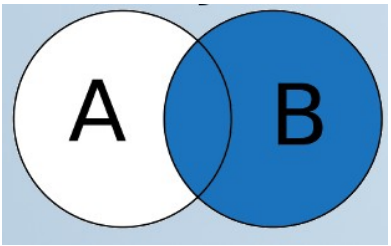
LEFT JOIN (sans l'intersection de B)
(jointure externe)

```
SELECT *
FROM A
LEFT JOIN B ON A.key = B.key
WHERE B.key IS NULL;
```



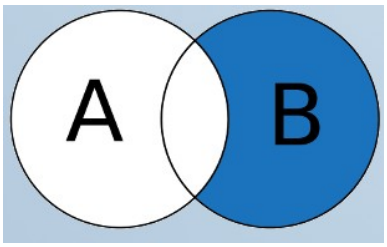
RIGHT JOIN
(jointure externe)

```
SELECT *
FROM A
RIGHT JOIN B ON A.key = B.key;
```



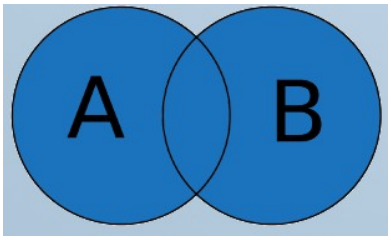
RIGHT JOIN (sans l'intersection de A)
(jointure externe)

```
SELECT *
FROM A
RIGHT JOIN B ON A.key = B.key
WHERE A.key IS NULL;
```



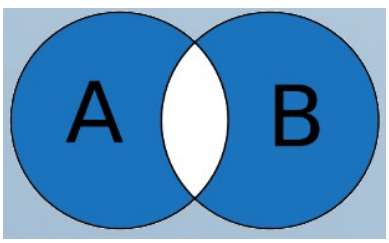
FULL JOIN
(jointure externe)

```
SELECT *
FROM A
FULL JOIN B ON A.key = B.key;
```



FULL JOIN (sans intersection)
(jointure externe)

```
SELECT *
FROM A
FULL JOIN B ON A.key = B.key
WHERE A.key IS NULL
OR B.key IS NULL;
```



Les jointures permettent de récupérer des données dans une base de données où les tables ont des relations entre elles.

Exemples avec les deux tables suivantes :

Table tomate :

Identifiant (PK)	masse	diamètre	couleur	nom_variete
1	230	100	rouge	Sweet Baby
2	100	70	jaune	Ananas
3	110	50	verte	Green Zebra
4	230	100	rouge	Roma
24	100	60	verte	Green Zebra
25	15	30	rouge	Sweet Baby
26	90	70	rouge	Roma

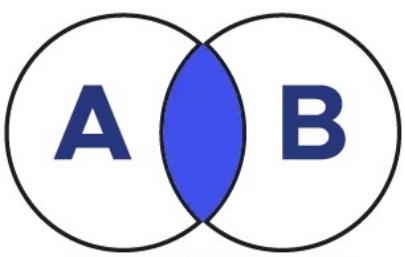
Table variete

Nom (PK)	prix_sachet	maturation	saveur
Green Zebra	4,75	tardive	sucrée
Noire de Crimée	4,9	précoce	sucrée
Reine Sainte-Marthe	4,5	mi-saison	acide
Roma	4,8	tardive	sucrée
Rose de Berne	4,5	tardive	sucrée
Supersteak	5,9	mi-tardive	cacahuète
Sweet Baby	6,95	précoce	sucrée

1. INNER JOIN

La jointure interne ou INNER JOIN permet de retourner les données quand la condition est vraie dans les deux tables.

Comme le montre le schéma, ce type de jointure va permettre de concaténer les tuples des 2 tables deux à deux si une condition est satisfaite. Cette condition peut être de tout type tant qu'elle retourne un booléen. Typiquement, cette condition sera l'égalité d'un attribut en commun. Il est



Intersection de deux ensembles

nécessaire de spécifier les attributs à utiliser pour effectuer la jointure. On utilise le mot clé **ON** suivi de l'égalité souhaitée après avoir renseigné les tables dans l'INNER JOIN.

C'est la requête qui fait perdre le plus d'informations puisqu'elle ne sélectionne que les lignes des deux tables où il y a des informations dans les deux tables.

```
SELECT identifiant, masse, diametre, couleur, nom, prix_sachet, maturation, saveur
FROM tomate AS T
INNER JOIN variete AS V
ON T.nom_variete = V.nom;
```

Résultat.

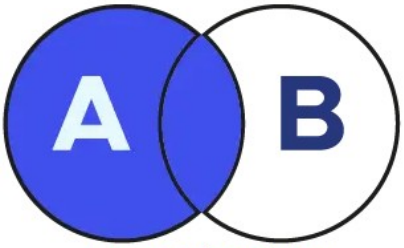
identifiant	masse	diametre	couleur	nom	prix_sachet	maturation	saveur
1	230	100	rouge	Sweet Baby	6,95	précoce	sucrée
3	110	50	verte	Green Zebra	4,75	tardive	sucrée
4	230	100	rouge	Roma	4,8	tardive	sucrée
24	100	60	verte	Green Zebra	4,75	tardive	sucrée
25	15	30	rouge	Sweet Baby	6,95	précoce	sucrée
26	90	70	rouge	Roma	4,8	tardive	sucrée

Ici en faisant une INNER JOIN entre les deux tables on obtient seulement les identifiants qui sont présents dans la table tomate et dans la table variete. La tomate Ananas est présent dans la table tomate mais pas dans la table variété tandis que les variétés Noire de Crimée, Reine Sainte-Marthe, Rose de Berne et Supersteak sont présentes dans la table variete mais pas dans la table tomate.

2. LEFT JOIN.

La jointure à gauche ou **LEFT** JOIN est une jointure entre 2 tables qui permet de retourner tous les enregistrements de la table de gauche même s'il n'y a pas de correspondance avec la table de droite. S'il n'y a pas de correspondance, les valeurs manquantes sont définies à NULL.

Comme pour le INNER JOIN, Il est nécessaire de spécifier les attributs à utiliser pour effectuer la jointure : On utilise le mot clé **ON** suivi de l'égalité souhaitée après avoir renseigné les tables dans le LEFT JOIN.



Jointure Gauche (Left Join)

Ici on fait une requête LEFT JOIN par rapport à la table tomate donc on affiche toutes les lignes lorsque le nom_variete est présent dans la table gauche donc dans la table tomate.

```
SELECT identifiant, masse, diametre, couleur, nom_variete, prix_sachet, maturation, saveur
FROM tomate AS T
LEFT JOIN variete AS V
ON T.nom_variete = V.nom;
```

Résultat.

identifiant	masse	diametre	couleur	nom_variete	prix_sachet	maturation	saveur
1	230	100	rouge	Sweet Baby	6,95	précoce	sucrée
2	100	70	jaune	Ananas			
3	110	50	verte	Green Zebra	4,75	tardive	sucrée
4	230	100	rouge	Roma	4,8	tardive	sucrée
24	100	60	verte	Green Zebra	4,75	tardive	sucrée
25	15	30	rouge	Sweet Baby	6,95	précoce	sucrée
26	90	70	rouge	Roma	4,8	tardive	sucrée

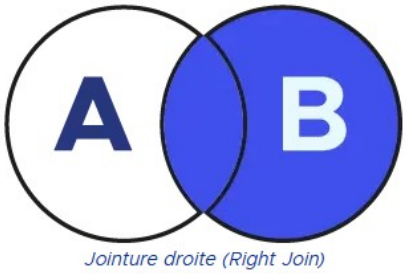
La tomate Ananas apparaît alors.

3. RIGHT JOIN.

La jointure à droite ou **RIGHT** JOIN est une jointure entre 2 tables qui permet de retourner tous les enregistrements de la table de droite même s'il n'y a pas de correspondance avec la table de gauche. S'il n'y a pas de correspondance, les valeurs manquantes sont définies à NULL.

Comme pour le INNER JOIN, ou le **LEFT** JOIN Il est nécessaire de spécifier les attributs à utiliser pour effectuer la jointure : on utilise le mot clé ON suivi de l'égalité souhaitée après avoir renseigné les tables dans le **RIGHT** JOIN.

Ici on fait un RIGHT JOIN par rapport à la table variete, donc on obtient comme résultat toutes les lignes de la table variete avec les lignes correspondantes de la table tomate lorsqu'il y a correspondance, c'est pourquoi la ligne correspondant à la tomate Ananas n'est pas présente.



Jointure droite (Right Join)

```
SELECT identifiant, masse, diametre, couleur, nom, prix_sachet, maturation, saveur
FROM tomate AS T
RIGHT JOIN variete AS V
ON T.nom_variete = V.nom;
```

Résultat.

identifiant	masse	diametre	couleur	nom	prix_sachet	maturation	saveur
1	230	100	rouge	Sweet Baby	6,95	précoce	sucrée
3	110	50	verte	Green Zebra	4,75	tardive	sucrée
4	230	100	rouge	Roma	4,8	tardive	sucrée
24	100	60	verte	Green Zebra	4,75	tardive	sucrée
25	15	30	rouge	Sweet Baby	6,95	précoce	sucrée
26	90	70	rouge	Roma	4,8	tardive	sucrée
				Noire de Crimée	4,9	précoce	sucrée
				Reine Sainte-Marthe	4,5	mi-saison	acide
				Rose de Berne	4,5	tardive	sucrée
				Supersteak	5,9	mi-tardive	cacahuète

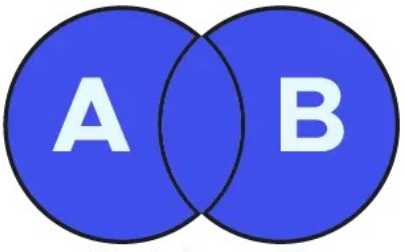
4. FULL JOIN.

La jointure FULL JOIN est une jointure entre 2 tables qui permet de retourner tous les enregistrements des deux tables même s'il n'y a pas de correspondance avec l'autre. S'il n'y a pas de correspondance, les valeurs manquantes sont définies à NULL. Cette jointure nécessite également l'attribut ON.

Ici peu importe s'il n'y a pas de lignes correspondantes dans l'une des deux tables, toutes les informations des deux tables sont présentes. C'est la jointure qui permet de garder le plus d'informations possibles.

Il n'y pas de FULL JOIN sur MySQL, il faut l'émuler.

```
SELECT * FROM tomate AS T
LEFT JOIN variete AS V
ON T.nom_variete = V.nom
UNION
SELECT * FROM tomate AS T
RIGHT JOIN variete AS V
```



Union de deux ensembles

```
ON T.nom_variete = V.nom;
```

5. NATURAL JOIN.

La jointure NATURAL JOIN est une jointure entre 2 tables qui permet de retourner les enregistrements des deux tables de façon "naturelle". Il faut qu'il y ait au moins une colonne avec le même nom dans les deux tables. Elle permet de faire une jointure naturelle en retournant les lignes où il y a des paires qui correspondent aux deux tables.

On renome la colonne nom_variete de la table tomate en nom. Ouvrir la base de données tomates2 pour tester la requête.

Avec NATURAL JOIN on ne spécifie pas le ON.

```
SELECT identifiant, masse, diametre, couleur, nom, prix_sachet, maturation, saveur
FROM tomate
NATURAL JOIN variete;
```

On obtient le même résultat avec :

```
SELECT identifiant, masse, diametre, couleur, T.nom, prix_sachet, maturation, saveur
FROM tomate T, variete V
WHERE T.nom=V.nom;
```

Résultat.

identifiant	masse	diametre	couleur	nom	prix_sachet	maturation	saveur
1	230	100	rouge	Sweet Baby	6,95	précoce	sucrée
3	110	50	verte	Green Zebra	4,75	tardive	sucrée
4	230	100	rouge	Roma	4,8	tardive	sucrée
24	100	60	verte	Green Zebra	4,75	tardive	sucrée
25	15	30	rouge	Sweet Baby	6,95	précoce	sucrée
26	90	70	rouge	Roma	4,8	tardive	sucrée

On voit à la sortie qu'il y a bien eu une jointure entre les deux tables avec "nom" car c'est la seule variable présente dans les deux tables.

Différence entre la NATURAL JOIN et la INNER JOIN ?

La INNER JOIN permet d'éviter la répétition de lignes égales, ce que ne permet pas la NATURAL JOIN. La jointure interne renvoie une table en fonction des données spécifiées dans le ON alors que la jointure naturelle renvoie une table en fonction d'une colonne avec le même nom et le même type dans les deux tables.

6. Remarques concernant la jointure interne.

1ère méthode : avec FROM et WHERE.

```
SELECT * FROM table_1, table_2
WHERE table_1.attribut = table_2.attribut;
```

Remarque : attribut doit avoir le même nom.

2nde méthode : avec JOIN et ON.

```
SELECT * FROM table_1
JOIN table_2 ON table_1.attribut = table_2.attribut ;
```

On peut permuter table1 et table2 :

```
SELECT * FROM table_2
JOIN table_1 ON table_2. attribut = table_1.attribut ;
```

Remarque : attribut doit avoir le même nom.

7. Jointure avec une table d'association.

Convention pour les clés :

- ➔ une clé primaire est en gras et est soulignée ;
- ➔ une clé étrangère est précédée d'un dièse et est en gras

On considère le schéma relationnel suivant :

table_1(**PK_1**, attribut_1, …, attribut_n)
table_2(**PK_2**, attribut_2, …, attribut_n)
table_asso(**#PK_1**, **#PK_2**)

On spécifie les trois tables dans le FROM. Dans le WHERE, on donne les deux conditions de jointure ainsi que les conditions supplémentaires.

```
SELECT * FROM table_1 T1, table_2 T2, table_assoc TA
WHERE TA.PK1=T1.PK1 AND TA.PK2=T2.PK2 AND condition(s);
```

8. Jointure avec trois tables.

```
SELECT attribut(s)
FROM (A JOIN B ON A.attribut1=B.attribut2) C          -- 1ère jointure --
JOIN D ON C.attribut3=D.attribut4                     -- 2ème jointure --
WHERE condition(s);
```