Même tables que le Cours 1 Ce que l'on veut

id_produit	desc_prod	prix	id_mag	quant
100	tabouret	34	1	10
100	tabouret	34	2	7
120	chaise	55	1	5
...	...	...	...	...

1er essai

```
SELECT * FROM produit, strock;;
```

Resutat

id_produit	desc_prod	prix	id_produit	id_mag	quant
100	tabouret	34	*100*	1	10
120	chaise	55		1	5
130	...	...	...		...
-------	-----	----	---		----
100	tabouret	34	101	5	15
120	chaise	55		5	5
130	...	...	...		...
...	...	...	...		...

On obtient le produit cartésien, c'est à dire toutes les combinaisons d'une ligne de `produit` et d'une ligne de `stock`.

Le produit cartesien sert **rarement**.

On ne regarde que les lignes ou les 2 `id_produit` sont identiques

```
SELECT * FROM produit, stock WHERE produit.id_produit = stock.id_produit
```

Remarque

`produit.id_produit` désigne la colonne `id_produit` de la table `produit`

On a obtenu une ***jointure*** naturelle. On parle de jointure quand on oblige les valeurs de certaine colonne à être égales entre elles (ou à respecter certaines relations). C'est une jointure ***naturelle*** quand on force l'égalité de colonne qui on le même sense (et le même nom).

On elimine une des deux colonnes `id_produit`

```
SELECT produit.id_produit,desc_produit,prix,id_magasin, quant
FROM produit, stock
WHERE produit.id_produit = stock.id_produit;
```

id_produit	desc_prod	prix	id_mag	quant
100	tabouret	34	1	10
100	tabouret	34	2	7
120	chaise	55	1	5
...	...	...	...	...

1. Application

- Obtenir les produits en stock dant le magasin 1
- On les identifiants description et le stock

```
SELECT produit.id_produit, desc_produit, quant
FROM produit, stock
WHERE produit.id_produit = stock.id_produit AND id_magasin = 1
```

- Donnez les produits ()id_produit, desc_produit) en trop grande quantité (>= 10) dans des magasin situé à Lyon (quant, nom_magasin)

```
SELECT p.id_produit, desc_produit, quant, nom_magasin
FROM produit AS p, strock AS s, magasin AS m
WHERE p.id_produit = s.id_produit AND s.id_magasin = m.m.id_magasin AND
quant >=10 AND adresse = 'Lyon';
```

1.1. Remarque

`produit AS p` est un alias On peut aussi ecrire : `produit p`

2. Jointure non naturelles

Les couples de magasins qui sont dans la même villes

```
SELECT m1.id_magasin,m1.nom_magasin, m2.id_magasin, m2.nom_magasin
FROM magasin m1, magasin m2
WHERE m1.adresse = m2.adresse AND m1.id_magasin < m2.id_magasin
```

`m1.id_magasin < m2.id_magasin` pour eviter

```
magasin 5, magasin 2
magasin 2, magasin 5
```

3. Opérations ensemblistes

- ***EX :*** Identifiant, description, prix des chaises et des tabourets

```
SELECT *
FROM produit
WHERE desc_produit = 'chaise'
UNION
SELECT *
FROM produit
WHERE desc_produit = 'tabouret';
```

Remarque générales sur les opérations ensemblistes : - Les 2 requêtes doivent donner le même nombre de colonne et leurs types doit correspondre. les noms des colonnes résultat sont ceux de la premieres requête.

Le résultat d'une opération ensembliste est sans doublon

3.1. 3 Opérations ensemblistes

```
UNION INTERSECTION EXCEPT (MINUS dans d'autre SQL)
```

```
SELECT id_produit
FROM produit
WHERE desc_produit = 'tabouret'
    INTERSECT
SELECT  id_produit
FROM stock s, magasin m
WHERE s.id_magasin = m.id_magasin AND adresse = 'Lyon';
```

On obtien les `id_produit` des tabouret en stock à Lyon

- ***EX :*** On veux les `id_produit` des produits qui ne sont en stock dans aucun magasin

```
SELECT id_produit
    FROM produit
EXCEPT
SELECT id_produit
    FROM stock;
```

- ***EX :*** On veux les couples (`id_produit, id_mag`) pour lesquels le produit n'est pas en stock dans le magasin.

```
SELECT id_produit, id_magasin // produit cartesien
    FROM produit, magasin
EXCEPT
SELECT id_produit, id_magasin // les couple (id_produit, id_mag )
FROM stock // ou produit est en stock dans le magasin
```