



## Le but du challenge Ma Petite Planète ?



Rassemble **tes camarades**  
pour participer  
au challenge !

Tou.te.s ensemble, relevez  
un **maximum de défis**  
BONUS pour la planète !

**Passez à l'action** en même  
temps que des milliers  
de personnes partout  
**dans le monde !**

Télécharge l'application !



### Inscrivez-vous



## Introduction

Les informations recherchées dans une base de données reposent sur des **patrons** toujours un peu identiques.

- Quels sont ces patrons?
- Comment les identifier?
- Comment construire la requête SQL qui répondra à la question?

## Attention!

Cette liste n'est pas exhaustive!

- au moins un
- aucun / jamais
- tous
- au moins deux

## Les requêtes de type au moins un

- Ce sont les plus courantes
- Exemples
  - Les clients qui ont commandé un téléphone
  - Les produits commandés par Dupont.

## Les requêtes de type au moins un

- Ce sont les plus courantes
- Exemples
  - Les clients qui ont commandé un téléphone  
Les clients qui ont commandé au moins un téléphone
  - Les produits commandés par Dupont.

## Les requêtes de type au moins un

- Ce sont les plus courantes
- Exemples
  - Les clients qui ont commandé un téléphone  
Les clients qui ont commandé au moins un téléphone
  - Les produits commandés par Dupont.  
Les produits qui ont été commandés au moins une fois par Dupont

## Les requêtes de type au moins un

### Traduction en SQL

- La **jointure naturelle**
  - ne garde que les lignes qui sont en correspondance
  - traduit bien la notion de **au moins un**
- **Exemple:** Les clients qui ont commandé un téléphone

```
select distinct idCli, nomCli, prenomCli
from CLIENT natural join COMMANDE natural join PRODUIT
where nomProd='Téléphone';
```

- Souvent le **distinct** est requis car la jointure peut dupliquer l'information (si le client a commandé plusieurs téléphones par ex.)

## Les requêtes de type au moins un

### Traduction en SQL

- **Le prédicat IN**
  - vérifie si une valeur se trouve (au moins une fois) dans une liste de valeurs
  - traduit bien la notion de **au moins un**
- **Exemple:** Les clients qui ont commandé un téléphone

```
select idCli, nomCli, prenomCli
from CLIENT
where idCli in ( /* la sous-requête retourne la liste */
  select idCli /* des numéros de client qui ont commandé un téléphone */
  from COMMANDE natural join PRODUIT
  where nomProd='Téléphone');
```

- Pour chaque client de la requête principale, on vérifie que son numéro fait partie des numéros de client ayant commandé un téléphone



## Les requêtes de type au moins un

### Traduction en SQL

- Le prédicat EXISTS
  - vérifie si une requête retourne au moins une valeur
  - traduit bien la notion de **au moins un**
- Exemple: Les clients qui ont commandé un téléphone

```
select idCli, nomCli, prenomCli
from CLIENT cl
where exists ( /* la sous-requête retourne la liste */
  select * /* des commande de téléphone du client de la requête princ. */
  from COMMANDE co natural join PRODUIT
  where nomProd='Téléphone' and cl.idCli=co.idCli);
```

- Pour chaque client de la requête principale, on vérifie que sa liste de commandes de téléphone contient au moins une valeur
- Il faut faire un lien entre la requête principale et la sous-requête

## Les requêtes de type au moins un

### Traduction en SQL

- **Comparaison ANY**
  - vérifie si la comparaison est vraie au moins une fois
  - traduit bien la notion de **au moins un**
- **Exemple:** Les clients qui ont commandé un téléphone

```
select idCli, nomCli, prenomCli
from CLIENT cl
where cl.idCli =ANY ( /* la sous-requête retourne la liste */
    select co.idCli /* des numeros de client qui ont commandé un téléphone */
    from COMMANDE co natural join PRODUIT
    where nomProd='Téléphone');
```

- Construction très similaire au **IN**

## Les requêtes de type aucun/jamais

- Exemples
  - Les clients qui n'ont pas commandé de clé USB
  - Les produits n'ont jamais été commandés par Dupont.

## Les requêtes de type aucun/jamais

- Exemples

- Les clients qui n'ont pas commandé de clé USB  
Les clients qui n'ont commandé aucune clé USB
- Les produits n'ont jamais été commandés par Dupont.

## Les requêtes de type aucun/jamais

- Exemples

- Les clients qui n'ont pas commandé de clé USB  
Les clients qui n'ont commandé aucune clé USB
- Les produits n'ont jamais été commandés par Dupont.  
Les produits qui n'ont été commandés aucune fois par Dupont

## Les requêtes de type aucun/jamais

### Traduction en SQL

- **Différence ensembliste**
  - **Idée**: On enlève à l'ensemble de tous les candidats potentiels les éléments qui répond à la requête **au moins un**
  - traduit bien la notion de **aucun**
- **Exemple**: Les clients qui n'ont pas commandé de clé USB

```
select idCli, nomCli, prenomCli /* Potentiellement tous le clients peuvent */
from CLIENT c1                /* répondre à la question */
except                        /* sans (MINUS) */
select idCli, nomCli, prenomCli ( /* les clients qui ont commandé */
from CLIENT natural join COMMANDER natural join PRODUIT /* au moins une */
where nomProd='Clé USB') /* clé USB */;
```

- Construction assez mathématique, peut être peu intuitive
- **EXCEPT** n'existe qu'à partir de la version 10.3 de MariaDB

## Les requêtes de type aucun/jamais

### Traduction en SQL

- Le prédicat **NOT IN**
  - vérifie si une valeur ne se trouve pas dans une liste de valeurs
  - traduit bien la notion de **aucun**
- **Exemple:** Les clients qui n'ont pas commandé de clé USB

```
select idCli, nomCli, prenomCli
from CLIENT
where idCli not in ( /* la sous-requête retourne la liste */
    select idCli /* des numéros de client qui ont commandé une clé USB */
    from COMMANDE natural join PRODUIT
    where nomProd='Clé USB');
```

- Pour chaque client de la requête principale, on vérifie que son numéro ne fait pas partie des numéros de client ayant commandé une clé USB

## Les requêtes de type aucun/jamais

### Traduction en SQL

- Le prédicat **NOT EXISTS**
  - vérifie si une requête retourne aucune valeur
  - traduit bien la notion de **aucun**
- **Exemple:** Les clients qui n'ont pas commandé de clé USB

```
select idCli, nomCli, prenomCli
from CLIENT cl
where not exists ( /* la sous-requête retourne la liste */
  select * /* des commandes de clés USB du client de la req. princ. */
  from COMMANDE co natural join PRODUIT
  where nomProd='Clé USB' and cl.idCli=co.idCli);
```

- Pour chaque client de la requête principale, on vérifie que sa liste de commande de clés USB ne contient aucune valeur
- Il faut faire un lien entre la requête principale et la sous-requête



## Les requêtes de type aucun/jamais

### Traduction en SQL

- **Comparaison ALL**
  - vérifie si la comparaison est vraie pour toutes les valeurs
  - traduit bien la notion de **aucun** si on l'utilise avec **!=**
- **Exemple:** Les clients qui n'ont pas commandé de clé USB

```
select idCli, nomCli, prenomCli
from CLIENT cl
where cl.idCli !=ALL ( /* la sous-requête retourne la liste */
    select co.idCli /* des numéros de clients qui ont commandé */
    from COMMANDE co natural join PRODUIT
    where nomProd='Clé USB'); /*une clé USB */
```

- Construction très similaire au **NOT IN**

## Les requêtes de type tous

- Exemples

- Les clients qui n'ont commandé que des clés USB
- Les produits n'ont été commandés que par Dupont.

⇒ On peut se ramener au cas **aucun/jamais** avec une négation dans la condition

⇒ Les constructions sont les mêmes: différence, NOT IN etc.

## Les requêtes de type tous

- Exemples

- Les clients qui n'ont commandé que des clés USB  
Les clients dont toutes les commandes sont des clés USB
- Les produits n'ont été commandés que par Dupont.

- ⇒ On peut se ramener au cas **aucun/jamais** avec une négation dans la condition
- ⇒ Les constructions sont les mêmes: différence, NOT IN etc.

## Les requêtes de type tous

- Exemples

- Les clients qui n'ont commandé que des clés USB

Les clients dont toutes les commandes sont des clés USB

- Les produits n'ont été commandés que par Dupont.

Les produits dont toutes les commandes ont été effectuées par Dupont

⇒ On peut se ramener au cas **aucun/jamais** avec une négation dans la condition

⇒ Les constructions sont les mêmes: différence, NOT IN etc.

## Les requêtes de type tous

- Exemples

- Les clients qui n'ont commandé que des clés USB  
Les clients dont toutes les commandes sont des clés USB  
les client dont aucune commande ne concerne pas une clé USB
- Les produits n'ont été commandés que par Dupont.  
Les produits dont toutes les commandes ont été effectuées par Dupont  
Les produits dont aucune commande n'a pas été effectuée par Dupont

⇒ On peut se ramener au cas aucun/jamais avec une négation dans la condition

⇒ Les constructions sont les mêmes: différence, NOT IN etc.

## Les requêtes de type tous

### Traduction en SQL

- Le prédicat **NOT IN**
- Exemple: Les clients qui n'ont commandé que des clés USB

```
select idCli, nomCli, prenomCli
from CLIENT
where idCli not in ( /* la sous-requête retourne la liste */
    select idCli /* des numéros de client qui ont commandé autre chose qu'une clé USB */
    from COMMANDE natural join PRODUIT
    where nomProd!='Clé USB');
```

## Les requêtes de type au moins 2

- Exemples

- Les clients qui ont commandé au moins deux produits différents
- Les produits n'ont été commandés au moins deux fois par Dupont.

- Il faut pouvoir accéder à 2 lignes différentes de la table COMMANDE

⇒ il va falloir 2 copies de cette table

## Les requêtes de type au moins 2

### Traduction en SQL

- **Exemple:** Les clients qui ont commandé au moins deux produits différents

```
select idCli, nomCli, prenomCli
from CLIENT cl
where cl.idCli in ( /* la sous-requête retourne la liste */
  select co1.idCli /* des numéros de clients qui apparaissent */
  from COMMANDE co1, COMMANDE co2 /* sur deux lignes différentes de COMMANDE */
  where co1.idCli=co2.idCli and co1.refProd!=co2.refProd);
```

- Que se passe-t-il si on veut exactement 2?
- Que se passe-t-il si on veut ceux qui ont effectué 100 commandes?



## Les requêtes de type au moins 2

### Traduction en SQL

- **Exemple:** Les clients qui ont commandé au moins deux produits différents

```
select idCli, nomCli, prenomCli
from CLIENT cl
where cl.idCli in ( /* la sous-requête retourne la liste */
    select co1.idCli /* des numéros de clients qui apparaissent */
    from COMMANDE co1, COMMANDE co2 /* sur deux lignes différentes de COMMANDE */
    where co1.idCli=co2.idCli and co1.refProd!=co2.refProd);
```

- Que se passe-t-il si on veut exactement 2?
- Que se passe-t-il si on veut ceux qui ont effectué 100 commandes?
- Pas très pratique  $\Rightarrow$  fonctions de comptage