09 - Requêtes imbriquées (non corrélées)

dimanche 14 décembre 2025 13:49

Le mot clé IN (inclusion)

↳ Liaison à l'aide d'un opérateur ensembliste.

↳ Exemple :
```
SELECT s.SNAME
FROM S s
WHERE s.STATUS IN (10,20,50);
```
→ Syntaxe : < valeur > IN < ensemble >

↳ Exemple :
```
SELECT *
FROM S s
WHERE s.CITY IN ( SELECT s.CITY
                  FROM S s
                  WHERE s.CITY = 'London');
```
= requête imbriquée

⚠ On peut nier l'inclusion avec NOT IN

Quantificateurs ANY et ALL

→ ANY : permet de vérifier si au moins une valeur de la liste satisfait la condition.

→ ALL : permet de vérifier si toutes les valeurs de la liste satisfont la condition.

Syntaxe :
< valeur > < comparateur > ANY/ALL < ensemble >

↳ Exemple :
```
SELECT p.NP
FROM P p
WHERE p.Poids <= ALL ( SELECT p2.poids
                       FROM P p2);
```

Requêtes imbriquées avec liaison naturelle :

→ Le produit le + léger

```
SELECT p.NP
FROM P p
WHERE p.Poids = ( SELECT MIN(p2.Poids)
                  FROM P p2);
```

un élément — un élément par un ensemble

Liaison (naturelle) avec un opérateur de comparaison

Opérations ensemblistes :

Union de deux ensembles :
↳ Mots clés UNION et UNION ALL

Intersection de deux ensembles :
↳ Mot clé INTERSECT ⇒ requête avec IN (ou EXISTS)

Différence de deux ensembles :
↳ Mot clé EXCEPT ⇒ requête avec NOT IN (ou NOT EXIST)

Union :

↳ SQL permet de combiner des requêtes à l'aide des opérateurs ensembliste Union et Union All.
↳ Ces opérateurs fonctionnent sur des ensembles compatibles, c'est à dire des relations ayant les mêmes colonnes.

Union et Union All :

→ Union fait l'union de deux ensemble et supprime les doublons.

↳ Exemple :
```
SELECT product_id
FROM order_id
UNION
SELECT product_id
FROM inventaire;
```

→ Union All garde les doublons.

↳ Exemple :
```
SELECT location_id
FROM location
UNION ALL
SELECT location_id
FROM department;
```

Intersection :

↳ Intersect renvoie l'intersection de deux ensembles.

↳ Exemple :
```
SELECT product_id
FROM inventaires
Intersect
SELECT product_id
FROM order_items;
```

⇒ Il est possible de s'en passer à l'aide de requêtes imbriquées.

```
SELECT product_id
FROM inventaires
WHERE product_id IN ( SELECT product_id
                      FROM order_items);
```

⇒ Les produits de la table inventaires dont l'identifiant est dans l'ensemble des identifiants des produits de la table order_items.

Différence

↳ EXCEPT renvoie la différence entre deux ensembles.

↳ Exemple :
```
SELECT product_id
FROM inventaires
EXCEPT
SELECT product_id
FROM order_items;
```

⇒ Il est possible de s'en passer à l'aide de requêtes imbriquées.

```
SELECT product_id
FROM inventaires
WHERE product_id NOT IN (SELECT product_id
                         FROM order_items);
```

⇒ Les produits de la table inventaires dont l'identifiant n'est pas dans l'ensemble des identifiants de produits de la table order_items.