

# Persistence des données I

## DON2

Denis Boigelot, Geneviève Cuvelier, Selim Rexhep, Yannick Voglaire



Haute École Bruxelles-Brabant  
École Supérieure d'Informatique

Année académique 2020 / 2021

## Plan du cours

- 0 – Présentation
- 1 – Introduction
- 2 – Dépendance fonctionnelle
- 3 – Schéma conceptuel
- 4 – Projection et sélection
- 5 – Jointure
- 6 – Agrégat
- 7 – Sous-requête
- 8 – Fichiers

## 8 – Sous-requête

- 1 Principe
- 2 Condition par sous-requête
- 3 Sous-requête corrélée
- 4 Quantificateurs ensemblistes
- 5 Table courante
- 6 Exercices

# Principe

Quels sont les numéros des clients habitants à Namur ?

```
SELECT ncli  
      FROM client  
      WHERE localite = ' Namur' ;
```

ncli
B062
C123
L422
Z127

# Principe

Quels sont les numéros des clients habitants à Namur ?

```
SELECT ncli  
      FROM client  
      WHERE localite = ' Namur' ;
```

ncli
B062
C123
L422
Z127

Les numéros et dates de commandes des clients habitants à Namur :

```
SELECT ncom, datecom  
      FROM commande  
      WHERE ncli IN ('C123',' Z127',' B062',' L422') ;
```

Ne marche qu'une fois

# Principe

Quels sont les numéros des clients habitants à Namur ?

```
SELECT ncli  
  FROM client  
  WHERE localite = ' Namur' ;
```

Les numéros et dates de commandes des clients habitants à Namur :

```
SELECT ncom, datecom  
  FROM commande  
  WHERE ncli IN (SELECT ncli  
                  FROM client  
                  WHERE localite = 'Namur');
```

Marche toujours !

# Principe

Remarque : certains select imbriqués peuvent s'écrire au moyen d'une jointure. La requête du slide précédent s'écrirait par exemple :

```
SELECT d.ncom, d.datecom  
      FROM commande d  
      JOIN client c ON c.ncli = d.ncli  
      WHERE c.localite = 'Namur' ;
```

Nous verrons cependant que ce n'est pas toujours le cas ...

# Principe

```

SELECT *
FROM produit
WHERE npro IN (SELECT npro
FROM detail
WHERE ncom IN (SELECT ncom
FROM commande
WHERE ncli IN (SELECT ncli
FROM client
WHERE localite = 'Namur')1
)2
)3
)4

```

1. Les clients de Namur

2. Les commandes des clients de Namur

3. Les détails des commandes des clients de Namur

4. Les produits des détails des commandes des clients de Namur



# Principe

L'idée générale des **selects imbriqués** est que le select interne peut être, soit :

- ◇ une table ou,
- ◇ une colonne ou,
- ◇ une valeur,

et doit être incluse dans le select externe où on attend respectivement une table, une colonne ou une valeur.

# IN

Une condition IN (sous-requête) correspond le plus souvent à une condition d'association.

```
SELECT *  
FROM T  
WHERE ct IN (SELECT cs  
                  FROM S  
                  WHERE <condition>);
```

La requête ci-dessus répond à la question :

Quels sont les **T** associés à des **S** ?

# IN

Symétrie sur les champs des conditions d'associations

Quelles sont les commandes qui sont associées à un client habitant à Namur ?

```
SELECT *  
  FROM commande  
 WHERE ncli IN (SELECT ncli  
                  FROM client  
                  WHERE localite = 'Namur');
```

Quelles sont les clients associés à une commande passée le 12-9-2017 ?

```
SELECT *  
  FROM client  
 WHERE ncli IN (SELECT DISTINCT ncli  
                  FROM commande  
                  WHERE datecom = '12-9-2017');
```

## IN

Quelles sont les commandes qui ne spécifient pas le produit PA60 ? (= qui ne sont pas associées à un détail spécifiant PA60)

```
SELECT ncom, datecom, ncli
FROM commande
WHERE ncom NOT IN (SELECT DISTINCT ncom
                     FROM detail
                     WHERE npro = 'PA60');
```

ok !

```
SELECT ncom, datecom, ncli
FROM commande
WHERE ncom IN (SELECT DISTINCT ncom
                 FROM detail
                 WHERE npro != 'PA60');
```

Ne répond pas à la question !

## Références multiples

Références multiples à une même table et sous-requête corrélée

```
SELECT ncli, nom, localite, compte  
  FROM client AS c  
  WHERE compte > (SELECT AVG(compte)  
                   FROM client  
                   WHERE localite = c.localite);
```

Pour chaque ligne trouvée par le select externe, le select interne est réévalué (de façon à ne considérer que les clients dont la localité correspond à celle de la ligne du select externe).

## Références multiples

Condition d'association quantifiée

```
SELECT ncom, datecom, ncli  
  FROM commande AS c  
  WHERE (SELECT COUNT(*)  
        FROM detail  
        WHERE ncom = c.ncom) >= 3;
```

## Références multiples

Null = null ?

```
SELECT ncli  
  FROM client  
  WHERE cat IN (SELECT cat  
                  FROM client  
                  WHERE ncli = 'D063');
```

**no data found...**

Réflexion : Donnez la sémantique de cette requête.

## EXISTS - NOT EXISTS

Le prédicat **EXISTS(E)**, où E est une sous-requête, est vrai si l'ensemble désigné par E est non vide

Quels sont les produits pour lesquels il existe au moins un détail ?

```
SELECT nrpo, libelle  
      FROM produit AS p  
      WHERE EXISTS(SELECT *  
                   FROM detail  
                   WHERE npro = p.npro);
```

Le prédicat **NOT EXISTS(E)**, est vrai si l'ensemble désigné par E est vide



# ALL - ANY

Quelles sont les commandes qui spécifient la plus petite quantité de PA60 ?

```
SELECT DISTINCT ncom
FROM detail
WHERE qcom <= ALL (SELECT qcom
                     FROM detail
                     WHERE nrpo = 'PA60')

AND nrpo = 'PA60';
```

Variante :

```
SELECT DISTINCT ncom
FROM detail
WHERE qcom = (SELECT MIN(qcom)
              FROM detail
              WHERE nrpo = 'PA60')

AND nrpo = 'PA60'
```

# ALL - ANY

Quelles sont les commandes qui ne spécifient **pas** la plus petite quantité de PA60 ?

```
SELECT DISTINCT ncom
FROM detail
WHERE qcom > ANY (SELECT qcom
                    FROM detail
                    WHERE npro = 'PA60')
AND npro = 'PA60';
```

Variante :

```
SELECT DISTINCT ncom
FROM detail
WHERE qcom > (SELECT MIN(qcom)
              FROM detail
              WHERE npro = 'PA60')
AND npro = 'PA60'
```

# IN - EXISTS

Deux expressions d'une condition d'association

Plus concise :

```
SELECT *  
  FROM client  
  WHERE ncli IN (SELECT DISTINCT ncli  
                  FROM commande  
                  WHERE datecom = '12-09-2009');
```

Plus explicite :

```
SELECT *  
  FROM client c  
  WHERE EXISTS (SELECT *  
                FROM commande m  
                WHERE m.ncli = c.ncli  
                AND datecom = '12-09-2009');
```

## Select imbriqué et jointure

Remarquons que la requête précédente peut à nouveau s'exprimer au moyen d'une jointure :

```
SELECT *  
  FROM client c  
    JOIN commande ON client.ncli = commande.ncli  
  WHERE datecom = '12-09-2009' ;
```

# Table courante

Quelle est la nature du retour d'une requête ?

## Table courante

Le retour d'une requête peut être utilisé comme une table au sein du **FROM**

```
SELECT *  
  FROM commande JOIN (SELECT *  
                        FROM client  
                        WHERE localite = 'Namur') AS c  
  ON commande.ncli = c.ncli ;
```

Peut s'écrire plus simplement :

```
SELECT *  
  FROM commande JOIN client  
    ON commande.ncli = client.ncli  
 WHERE localite = 'Namur' ;
```

## Sous-requêtes : exercices

Donnez la sémantique :

```
SELECT npro  
  FROM produit p  
  WHERE libelle LIKE '%SAPIN%'  
        AND EXISTS (SELECT *  
                     FROM detail  
                     WHERE npro = p.npro) ;
```

```
SELECT DISTINCT localite  
  FROM client  
  WHERE ncli IN (SELECT ncli  
                 FROM commande  
                 WHERE ncom IN (SELECT ncom  
                               FROM detail  
                               WHERE npro = 'CS464')) ;
```

## Sous-requêtes : exercices

Donnez la sémantique :

```
SELECT npro, libelle
FROM produit
WHERE npro NOT IN (SELECT npro
                    FROM detail
                    WHERE ncom IN (SELECT ncom
                                FROM commande
                                WHERE datecom LIKE '%2008%'));
```

```
SELECT ncli, nom
FROM client c
WHERE compte <= ALL (SELECT compte
                      FROM client cl JOIN commande cmd
                      ON cl.ncli = cmd.ncli
                      WHERE c.cat = cl.cat);
```