BNF101 Base de Données Relationnelles

Lecture et extraction des données,

Requêtes Select avec jointures externes,

Requêtes Select avec sous-requêtes imbriquées,

Requêtes Select avec fonction d’agrégat,

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Dans les différents types de jointure présentés précédemment, toutes les lignes de la première table qui n'ont pas de lignes dans la seconde table et qui vérifient la condition de jointure, ne font pas partie du résultat final.

Si on souhaite faire apparaître cette ligne dans le résultat, on utilise la jointure externe.

Une jointure externe est une jointure qui favorise une table par rapport à une autre. Aussi, les lignes de la table propriétaire seront affichées même si la condition n'est pas réalisée.

Une jointure externe est explicitée par l'opérateur **LEFT OUTER JOIN** ou **RIGHT OUTER JOIN** placé entre les 2 tables concernées par la jointure.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Exemple :

Soit 2 tables A et B ayant une colonne en commun sous forme de clé étrangère ou non.

La syntaxe est la suivante :

SELECT <liste colonnes> FROM A LEFT OUTER JOIN B ON A.col_commune = B.col_commune;

SELECT <liste colonnes> FROM A RIGHT OUTER JOIN B ON A.col_commune = B.col_commune;

<liste colonnes> doit se comprendre comme col1,col2,…coln (ex : numcli,nomcli,…numcom…).

Col_commune doit se comprendre comme A.primary key = B.foreign key (ex : client.numcli=commande.numcli).

C ’est cette syntaxe qui est recommandée.

Remarque :

Dans le SGBD ORACLE il existe une ancienne syntaxe utilisant la clause WHERE.

SELECT <liste colonnes> FROM A, B WHERE A.col_commune = B.col_commune (+) ;

L’opérateur (+) est placé du côté de la table qui ne sera pas privilégiée en termes d’enregistrements non liés.

Cette syntaxe n’est pas recommandée et est surtout utilisé en cas de besoin de compatibilité avec des anciennes version d’ORACLE.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Remarque:

Le choix de la table A comme table gauche et de la table B comme table droite est arbitraire.

Cependant, dans le cas où la table B possède une clé étrangère qui référence la clé primaire de la table A, il est logique de choisir la table A comme table gauche.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Exercice :

Lister tous les clients qui ont passé ou non une ou plusieurs commandes.

Lister tous les clients qui n'ont pas passé de commandes.

Pour ces 2 requêtes, on affichera les colonnes suivantes :

Numéro du client, nom du client, le numéro de la commande, la date de la commande.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Correction:

Lister tous les **clients** qui ont passé ou non une ou plusieurs commandes :

SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A **LEFT OUTER JOIN** Commande B **ON** A.NumCli = B.NumCli;

Ou bien avec l'ancienne syntaxe :

SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A, Commande B **WHERE** A.NumCli **=** B.NumCli **(+)**;

L'opérateur (+) est place du côté de la table commande qui contient uniquement des enregistrement liés à un enregistrement la table client.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Résultat :

Ci-dessous la table de résultats avec les clients qui ont des commandes et avec 2 clients qui n'ont pas de commandes : les valeurs des colonnes NumCom et DateCom sont nulles « null ».

```
SQL> SELECT Client.NumCli, Client.NomCli, Commande.Numcom, Commande.Numcli, Commande.DateCom FROM Client LEFT OUTER JOIN
 Commande ON client.NumCli = commande.NumCli;

    NUMCLI NOMCLI                          NUMCOM     NUMCLI DATECOM
---------- ------------------------------ ---------- ---------- --------
         1 redford                                 2          1 15/10/06
         1 redford                                 3          1 18/09/18
         2 spacey                                  1          2 22/11/06
         3 roberts                                 4          3 17/11/18
         3 roberts                                 6          3 13/08/18
         4 benini
         5 maurante                                5          5 13/08/16
         6 pausini
         7 murphy                                  7          7 17/08/18
         8 Eastwood                                8          8 16/11/18
```

Dans la version 11g (version assez ancienne déployée sur le serveur delos du CNAM, les lignes des résultats sont indexées et triées par le numéro de client (numCli), clés primaires de la table CLIENT. Idem pour les jointures internes.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

```
SQL> SELECT Client.NumCli, Client.NomCli, Commande.Numcom, Commande.Numcli, Commande.DateCom FROM Client LEFT OUTER JOIN
 Commande ON Client.NumCli = Commande.NumCli;

    NUMCLI NOMCLI                          NUMCOM     NUMCLI DATECOM
---------- ------------------------------ ---------- ---------- --------
         2 SPACEY                                  1          2 22/11/06
         1 REDFORD                                 2          1 15/10/06
         1 REDFORD                                 3          1 18/09/18
         3 ROBERTS                                 4          3 17/11/18
         5 MAURANTE                                5          5 13/08/16
         3 ROBERTS                                 6          3 13/08/18
         7 MURPHY                                  7          7 17/08/18
         8 EASTWOOD                                8          8 16/11/18
         6 PAUSINI
         4 BENINI
```

Dans la version 21c d'ORACLE (version récente), les lignes des résultats sont triées et indexées sur les numéros de commande (numCom), clés primaires de la table COMMANDE.

Idem pour les jointures internes.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

```
bnf101=# SELECT Client.NumCli, Client.NomCli, Client.AdresseCli, Commande.Numcom, Commande.Numcli, Commande.DateCom FROM Client
 LEFT OUTER JOIN Commande ON Client.NumCli = Commande.NumCli;
 numcli |  nomcli  |       adressecli       | numcom | numcli |  datecom
--------+----------+------------------------+--------+--------+------------
      2 | spacey   | 7 rue vauquelin        |      1 |      2 | 2006-11-22
      1 | redford  | 8 rue de la paix       |      2 |      1 | 2006-10-15
      1 | redford  | 8 rue de la paix       |      3 |      1 | 2018-09-18
      3 | roberts  | 6 avenue foch          |      4 |      3 | 2018-11-17
      5 | maurante | 5 rue de florence      |      5 |      5 | 2016-08-13
      3 | roberts  | 6 avenue foch          |      6 |      3 | 2018-08-13
      7 | murphy   | 5 rue des vignes       |      7 |      7 | 2018-08-17
      8 | Eastwood | 10 rue mouffetard      |      8 |      8 | 2018-11-16
      6 | pausini  | 2 rue du nord          |        |        |
      4 | benini   | 40 avenue des gobelins |        |        |
```

Dans la version 9.5.25 de POSTGRESQL (version récente), les lignes des résultats sont triées et indexées sur les numéros de commande (numCom), clés primaires de la table COMMANDE.

Idem pour les jointures internes.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Correction:

Lister tous les **clients** qui n'ont pas passé de commandes :

1ère Forme :

SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A **LEFT OUTER JOIN** Commande B **ON** A.NumCli = B.NumCli **WHERE B.NumCom IS NULL**;

Il suffit de rajouter une condition "IS NULL" sur la valeur du numéro de commande.

Pour rappel, les client qui n'ont pas de commande ont un numéro de commande égal à null dans la table de résultats de la requête.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Résultat :

```
SQL>
SQL> SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A LEFT OUTER JOIN Commande B ON A.NumCli = B.NumCli WHERE B.NumCom IS NULL;

    NUMCLI NOMCLI                          NUMCOM DATECOM
---------- ------------------------------ ---------- ---------
         4 benini
         6 pausini
```

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Correction:

Lister tous les **clients** qui n'ont pas passé de commandes :

2ème forme :

(SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A **LEFT OUTER JOIN** Commande B **ON** A.NumCli = B.NumCli)

MINUS

(SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A **INNER JOIN** Commande B **ON** A.NumCli = B.NumCli);

L'opérateur MINUS permet d'extraire les résultats qui ne sont pas communs aux 2 requêtes.

Attention le nom de cet opérateur peut changer en fonction du SGBD :

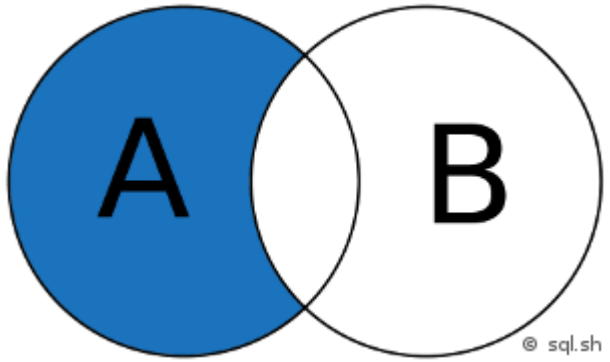
mySql et Oracle : MINUS.

Postgres : EXCEPT.

Remarque : cette requête n'est pas optimale en terme de performance.

LMD : SELECTION DES DONNEES – LES JOINTURES EXTERNES

Résultats :




```
SQL>
SQL> (SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A LEFT OUTER JOIN Commande B ON A.NumCli = B.NumCli)
MINUS
(SELECT A.NumCli, A.NomCli, B.NumCom, B.DateCom FROM Client A INNER JOIN Commande B ON A.NumCli = B.NumCli);
  2    3
    NUMCLI NOMCLI                          NUMCOM DATECOM
---------- ------------------------------ ---------- ---------
         4 benini
         6 pausini
```

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES

Consultation avec condition en sous-requête SQL permet de comparer une expression ou une colonne au résultat d'une autre requête select. Cette condition est dite condition de sous requête et les deux requêtes sont dites requêtes imbriquées.

Les formes générales des sous-requêtes suivent généralement le WHERE.

Elles se présentent comme:

WHERE expression opérateur_de_comparaison {**ALL/ANY/SOME**} (requête SELECT)

WHERE expression [**NOT**] **IN** (requête SELECT)

WHERE [**NOT**] **EXISTS** (requête SELECT)

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES

- **Remarques:**

Les sous-requêtes situées après les mots clefs IN,ALL,ANY et SOME doivent avoir le même nombre de colonnes que celui spécifié dans l'expression.

Ces sous-requêtes peuvent rendre plusieurs valeurs qui seront évaluées en fonction de l'opérateur de comparaison.

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES

Comparaison entre jointure interne et requête interne :

Liste de clients qui ont passé des commandes :

Requête avec une jointure interne :

SELECT A.NumCli, A.NomCli FROM Client A INNER JOIN Commande B ON A.NumCli = B.NumCli;

Requête avec une requête interne :

SELECT A.NumCli, A.NomCli FROM Client A WHERE NumCli IN (SELECT B.NumCli FROM Commande B);

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES : JOINTURE INTERNE

```
SQL> SELECT A.NumCli, A.NomCli FROM Client A INNER JOIN Commande B ON A.NumCli =
 B.NumCli;

    NUMCLI NOMCLI
---------- ------------------------------
         1 redford
         1 redford
         2 spacey
         3 roberts
         3 roberts
         5 maurante
         7 murphy
         8 Eastwood

8 ligne(s) selectionnee(s).
```

Dans la version 11g (version assez ancienne déployée sur le serveur delos du CNAM, les lignes des résultats sont indexées et triées par le numéro de client (numCli), clés primaires de la table CLIENT.

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES : SELECT IMBRIQUE

```
SQL> SELECT A.NumCli, A.NomCli FROM Client A WHERE NumCli IN (SELECT B.NumCli FR
OM Commande B);

    NUMCLI NOMCLI
---------- ------------------------------
         1 redford
         2 spacey
         3 roberts
         5 maurante
         7 murphy
         8 Eastwood

6 ligne(s) selectionnee(s).
```

Dans la version 11g (version assez ancienne déployée sur le serveur delos du CNAM, les lignes des résultats sont indexées et triées par le numéro de client (numCli), clés primaires de la table CLIENT.

Le résultat de la requête interne ramène 1 ligne de moins que celui de la jointure.

Voir le document BNF101_select_jointure_et_select_imbriqué.PDF

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES : SELECT IMBRIQUE

```
SQL> SELECT A.NumCli, A.NomCli FROM Client
  2  A INNER JOIN Commande B ON
  3  A.NumCli = B.NumCli;

    NUMCLI NOMCLI
---------- ------------------------------
         2 SPACEY
         1 REDFORD
         1 REDFORD
         3 ROBERTS
         5 MAURANTE
         3 ROBERTS
         7 MURPHY
         8 EASTWOOD

8 lignes sÚlectionnÚes.

SQL> SELECT A.NumCli, A.NomCli FROM Client A WHERE NumCli IN (SELECT B.NumCli FROM Commande B);

    NUMCLI NOMCLI
---------- ------------------------------
         2 SPACEY
         1 REDFORD
         3 ROBERTS
         5 MAURANTE
         7 MURPHY
         8 EASTWOOD
```

Dans la version 21c d'ORACLE (version récente), les lignes des résultats sont triées et indexées sur les numéros de commande (numCom), clés primaires de la table COMMANDE.

La colonne numcom n'est pas visible dans ces requêtes mais le tri se fait bien sur ce numéro numcom du côté de la table COMMANDE.

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES

Comparaison entre jointure interne et requête interne :

Retour sur l'exercice du début du cours : écrire une requête permettant de ramener les clients qui n'ont pas passé de commande.

3ème forme de requête avec une requête interne :

SELECT A.NumCli, A.NomCli FROM Client A WHERE NumCli NOT IN (SELECT B.NumCli FROM Commande B);

Remarque :

Cette requête est la plus efficace en terme de performances car le nombre de comparaisons est moins grand que pour les requêtes de la 1ère (jointure externe + filtre IS NULL sur le numCom) et de la 2ème forme (requête jointure externe moins requête jointure interne).

LMD : SELECTION DES DONNEES – LES REQUETES INTERNES

Ex :

ALL :

Si nous désirons lister les produits dont la quantité en stock est supérieure à toute quantité commandée on peut écrire:

SELECT Numpro, NomPro **FROM** Produit **WHERE** qteSto > **ALL**

(**select** qteCom **from** LigneCom);

IN: la condition est vraie si la comparaison est vraie pour une des valeurs retournées par la sous requête.

EXISTS: renvoie le booléen vrai ou faux selon le résultat de la sous-requête. Si l'évaluation de la sous-requête donne lieu à une ou plusieurs lignes, la valeur retournées est vraie.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

Les fonctions d'agrégat

SQL permet de consulter les contenus des attributs avec aussi l'application d'opérations arithmétiques et ce via un certain nombre de fonctions appelées fonctions d'agrégats dont nous citons quelques-unes.

- La fonction **COUNT**

Cette fonction a deux formes syntaxiques:

- **COUNT (*)**: donne le nombre de lignes satisfaisants la requête.
- **COUNT ([DISTINCT/ALL]** colonne**)**: donne le nombre de valeur de colonne satisfaisants la requête.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

Les fonctions d'agrégat

SUM ([DISTINCT/ALL] colonne**)** : donne la somme des valeurs de la colonne répondant à la requête. La colonne doit être de type numérique. L'option DISCTINCT somme les valeurs uniques de la colonne.

- La fonction **AVG**

AVG ([DISTINCT/ALL] colonne**)** : donne la moyenne des valeurs de la colonne répondant à la requête. La colonne doit être de type numérique. L'option DISCTINCT calcule la moyenne des valeurs distinctes de la colonne.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

- La fonction **MAX**

MAX ([DISTINCT/ALL] colonne**)** : donne le maximum relatif à la colonne.

- La fonction **MIN**

MIN ([DISTINCT/ALL] colonne**)** : donne le minimum relatif à la colonne.

- La fonction **STDDEV**

STDDEV ([DISTINCT/ALL] colonne**)** : donne l'écart type relatif à la colonne.

- La fonction **VARIANCE**

VARIANCE ([DISTINCT/ALL] colonne**)** : donne la variance relative à la colonne.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

- **Remarques**:
  - la colonne paramètre de ces fonctions, peut-elle aussi, être une expression de SUM, AVG, MAX, MIN, STDDEV.
  - la fonction COUNT comptabilise les occurrences ayant une valeur NULL, ce qui n'est pas le cas pour les autres fonctions.
  - d'autres fonctions sont aussi offertes par SQL relatives à la manipulation des chaines de caractères et de calcul simple.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

Le groupement

SQL permet de grouper des lignes de données ayant des valeurs communes via la clause GROUP BY associée à la clause SELECT et la syntaxe est:

SELECT nom_colonne_1,… , nom_colonne_j **FROM** nom_de table [**WHERE** condition] [**GROUP BY** liste de colonnes];

- **Remarques**:

La liste de sélection peut contenir uniquement les types des expressions suivantes:

- constantes
- fonctions d'agrégat
- expression identique à celle spécifiée dans la clause group by
- expression évaluant la même valeur pour toutes les lignes dans un groupe.

L'expressions de la clause GROUP BY peut contenir toute colonne faisant partie de la liste de tables de la clause FROM.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

Exemple de requête avec GROUP BY :

Lister les nombres de clients regroupés par ville :

Select count(*) AS NombreCli, ville FROM Client GROUP BY ville;

La syntaxe « AS NombreCli » permet d'affiche les nombres de clients dans une colonne de la table de résultats.

LMD : SELECTION DES DONNEES – LES FONCTIONS D'AGREGAT

```
SQL> Select count(*) AS NombreCli, ville FROM Client GROUP BY ville;

 NOMBRECLI VILLE
---------- ------------------------------
         5 paris
         1 marseille
         1 lyon
         1 lille
```