

SQL/

SQL-LMD Les requêtes complexes

association nationale
pour la formation professionnelle
des adultes

Vincent BOST



Le sous ensemble DML ou LMD permet de lire ou modifier le contenu des tables.

4 verbes expriment 4 formes de requêtes

- **Select** : requête extraction des données des tables
- **Insert** : requête insertion de nouvelles lignes dans une table
- **Update** : requête modification de lignes existantes
- **Delete** : requête suppression des lignes d'une table

Prise en compte des opérations CRUD

- **Create, Read, Update, Delete**

Requête d'extraction sur une seule table

Toutes les lignes sont retenues

- L'exécution de l'opération Select renvoie un jeu de résultats (Records Set ou Data Set) sous la forme d'une table (temporaire)

```
Select NCLI, NOM, LOCALITE  
From CLIENT;
```

NCLI	NOM	LOCALITE
B062	GOFFIN	Namur
B112	HANSENNE	Poitiers
B332	MONTI	Genève
B512	GILLET	Toulouse
C003	AVRON	Toulouse
C123	MERCIER	Namur
C400	FERARD	Poitiers
D063	MERCIER	Toulouse
F010	TOUSSAINT	Poitiers
F011	PONCELET	Toulouse
F400	JACOB	Bruxelles
K111	VANBIST	Lille
K729	NEUMAN	Toulouse
L422	FRANCK	Namur
S127	VANDERKA	Namur
S712	GUILLAUME	Paris

```
Select *  
From CLIENT;
```

*** = liste des colonnes**

```
Select NCLI, NOM  
From CLIENT  
Where LOCALITE = 'Toulouse';
```

NCLI	NOM
B512	GILLET
C003	AVRON
D063	MERCIER
F011	PONCELET
K729	NEUMAN

La clause Where exprime une condition sur les lignes retenues ou non dans le résultat

La clause Where respecte la syntaxe de l'expression d'une condition : inclusion d'opérateurs logiques

Utilisation de l'opérateur LIKE pour rechercher les éléments respectant un modèle

```
select NCLI
from CLIENT
where CAT like 'B_';
```

'_' = un caractère
quelconque

```
select NPRO
from PRODUIT
where LIBELLE like '%SAPIN%';
```

pattern

'%' = une chaîne
quelconque

Utilisation des caractères génériques %

Un **modèle** définit une famille de chaînes de caractères :

'B_' → 'B1'
 ' Bd'
 ' B '

'%SAPIN%' → 'PL. SAPIN 200x20x2'
 'Boite en SAPIN'
 'SAPIN VERNI'

'B_' ↗ 'xB'
 'B'
 'B12'

'%SAPIN%' ↗ 'boite S A P I N '

Il n'est pas possible
de comparer une
expression à null

Utiliser à cette fin
l'opérateur Is

```
Select NCLI  
From CLIENT  
Where CAT = null;
```

```
Select NCLI  
From CLIENT  
Where CAT is null;
```

NCLI
D063
K729

```
Select NCLI  
From CLIENT  
Where CAT is not null;
```

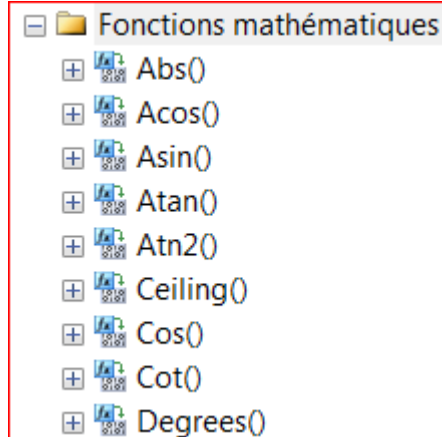
Il est possible de produire des valeurs à partir de calculs.

Les expressions peuvent mobiliser des constantes, faire référence à des colonnes et inclure des fonctions.

Restriction : types compatibles avec les opérations

```
Select NPRO as Produit, 0.21*PRIX*QSTOCK as Valeur_TVA
From  PRODUIT
Where QSTOCK > 500;
```

Produit	Valeur_TVA
CS264	67788
PA45	12789
PH222	37770.6
PS222	47397



Fonctions d'agrégation

- + Avg()
- + Binary_checksum()
- + Checksum()
- + Checksum_agg()
- + Count()
- + Count_big()
- + Grouping()
- + Grouping_Id()
- + Max()
- + Min()
- + Stdev()
- + Stdevp()
- + Sum()
- + Var()
- + Varp()

```
select 'Namur',avg(COMPTE) as Moyenne,  
       max(COMPTE)-min(COMPTE) as Ecart_max,  
       count(*) as Nombre  
from   CLIENT  
where  LOCALITE = 'Namur';
```

Namur	Moyenne	Ecart_max	Nombre
Namur	-2520	4580	4

```
select sum(QSTOCK*PRIX)  
from   PRODUIT  
where  QSTOCK > 500;
```



```
select count(NCLI) as Nombre,  
       count(NOM) as Noms,  
       count(LOCALITE) as Localités,  
       count(CAT) as Catégories  
from CLIENT;
```

Nombre	Noms	Localités	Catégories
16	16	16	14

```
select count(distinct NCLI) as Nombre,  
       count(distinct NOM) as Noms,  
       count(distinct LOCALITE) as Localités,  
       count(distinct CAT) as Catégories  
from CLIENT;
```

Nombre	Noms	Localités	Catégories
16	15	7	4

Prise en compte des valeurs distinctes

Les valeurs nulles ne sont pas retenues dans les calculs

La clause GROUP BY permet d'exprimer l'expression d'une agrégation selon un ou plusieurs domaines

La clause HAVING permet de restreindre le jeu de résultats aux valeurs résultantes d'une agrégation qui satisfont à la condition

```
select Localite, avg(COMPTE) as Moyenne,  
       count(*) as Nombre  
from   CLIENT  
group by Localite  
having avg(COMPTE) < 0
```

Une seule opération SELECT

Les lignes des différentes tables sont jointes en recourant le plus généralement aux jointures naturelles : Clé primaire – Clé étrangère

Nous distinguerons 4 opérateurs de jointures :

- La jointure interne (avec égalité) INNER JOIN
- La jointure externe gauche LEFT OUTER JOIN
- La jointure externe droite RIGHT OUTER JOIN
- La jointure externe complète FULL OUTER JOIN

Cette requête extrait les éléments des clients et des commandes pour les clients localisés en France

Uniquement les clients référencés dans la table des commandes

La jointure se fait entre la clé primaire de la table clients et la clé étrangère de la table commandes qui référence la table client

```
SELECT CustomerID, CompanyName, OrderDate, dbo.Orders.OrderID
FROM      Customers INNER JOIN  Orders
ON Customers.CustomerID = Orders.CustomerID
WHERE     (idPays = 'FR')
```

**Cette requête extrait les éléments des tables clients et commandes
Tous les clients référencés ou non dans la table des commandes
seront retenus**

**Pour ceux non référencés dans la table commande, les éléments de
la table commande vaudront null**

```
SELECT CustomerID, CompanyName, OrderDate, dbo.Orders.OrderID
FROM      Customers LEFT OUTER JOIN  Orders
ON Customers.CustomerID = Orders.CustomerID
WHERE     (idPays = 'FR')
```

Cette requête extrait les éléments des tables clients et commandes
Tous les clients référencés dans la table des commandes seront
retenus ainsi que les commandes sans référence client
Les contraintes d'intégrité référentielles définies ne peuvent
autoriser une commande sans référence de client

```
SELECT CustomerID, CompanyName, OrderDate, dbo.Orders.OrderID
FROM      Customers RIGHT OUTER JOIN  Orders
ON Customers.CustomerID = Orders.CustomerID
WHERE     (idPays = 'FR')
```

Cette requête réalise la jointure entre les éléments des tables clients et commandes puis joint l'ensemble de ce résultat aux éléments de la table Détails des commandes

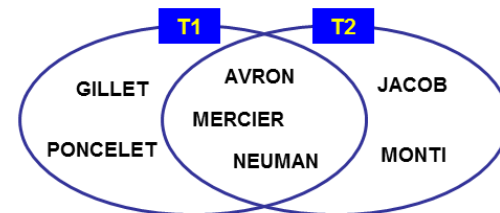
```
SELECT  CustomerID, CompanyName, OrderDate, OrderID, idPays,  
        Order_Details.ProductID, Order_Details.UnitPrice, Order_Details.Quantity  
FROM      dbo.Customers INNER JOIN  dbo.Orders  
        ON dbo.Customers.CustomerID = dbo.Orders.CustomerID  
        INNER JOIN  dbo.Order_Details ON dbo.Orders.OrderID =  
        dbo.Order_Details.OrderID  
WHERE     (dbo.Customers.idPays = 'FR')
```

Les ensembles doivent
avoir le même schéma

Union All permet de
conserver les doublons

T1
NOM
GILLET
AVRON
MERCIER
PONCELET
NEUMAN

T2
NOM
MONTI
NEUMAN
JACOB
MERCIER
AVRON



```
select NOM
from T1
union
select NOM
from T2
```

NOM
GILLET
AVRON
MERCIER
PONCELET
NEUMAN
MONTI
JACOB

```
select NOM
from T1
intersect
select NOM
from T2
```

NOM
AVRON
MERCIER
NEUMAN

```
select NOM
from T1
except
select NOM
from T2
```

NOM
GILLET
PONCELET


```
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'))));
```

→ les clients de Namur

→ les commandes des clients de Namur

→ les détails des commandes des clients de Namur

→ les produits référencés par les détails des commandes des clients de Namur

Utilisation du NOT IN pour préciser l'absence de relation

Intersection sur des ensembles comportant une seule colonne

```
select NCLI , NOMCLI  
from CLIENT  
where NCLI not in (select NCLI  
                   from commande);
```

Cette forme de requête est gourmande en ressources.

Référencie au sein de la requête interne une ligne de la requête externe

```
select NCLI, NOM, LOCALITE, COMPTE
from CLIENT as C
where COMPTE > (select avg(COMPTE)
                from CLIENT
                where LOCALITE = C.LOCALITE);
```

Sélection des clients dont le solde du compte est supérieur à la moyenne des comptes des clients de leur ville

A noter l'usage d'une deuxième instance de la même table avec un alias C

Sélection des produits qui ont fait l'objet d'une commande

La requête interne renvoie un ensemble de lignes (true) ou un ensemble vide (false)

```
select NPRO, LIBELLE
from   PRODUIT as P
where  exists (select *
               from   DETAIL
               where  NPRO = P.NPRO) ;
```

Extension de la clause SELECT


```
select O.OrderID, o.OrderDate, (select sum(Quantity * P.UnitPrice)
from [Order Details] OD inner join Products P
on OD.ProductID = P.ProductID
where OD.OrderID = O.OrderID) as MontantCdeActualise
from Orders O
```

expression retournant une
valeur

corrélation

Utilisation d'une table temporaire issue de l'exécution d'une opération de sélection

expression retournant une table



```
select MAX(MontantCdeActualise)
from (select sum(Quantity * P.UnitPrice) as MontantCdeActualise
from [Order Details] OD inner join Products P
on OD.ProductID = P.ProductID
Group by OrderID) as total |
```