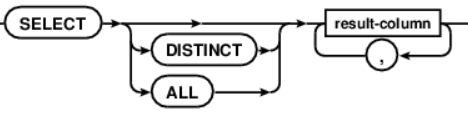
SQL : Langage de Manipulation de Données

Table des matières

[Notations utilisées lors de l'écriture des ordres SQL 1](#_Toc524078307)

[I. La projection. 2](#_Toc524078308)

[I.A. Choix des colonnes. 2](#_Toc524078309)

[ 2](#_Toc524078310)

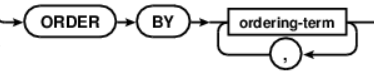
[ 2](#_Toc524078311)

[I.B. Titre de colonne 3](#_Toc524078312)

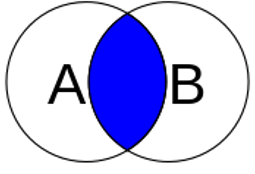
[II. Sélection ou restriction : Clause WHERE. 4](#_Toc524078313)

[Description of where_clause.gif follows 4](#_Toc524078314)

[III. Tri du résultat : Clause ORDER BY. 6](#_Toc524078315)

[ 6](#_Toc524078316)

[IV. Jointure interne. 7](#_Toc524078317)

 [IV.A. Définition. 7](#_Toc524078318)

[IV.B. Exemple 7](#_Toc524078319)

[IV.C. Combinaison de la jointure avec la projection. 9](#_Toc524078320)

[IV.D. Combinaison de la jointure avec la restriction. 9](#_Toc524078321)

[IV.E. Synonymes ou alias. 9](#_Toc524078322)

[IV.F. Jointure naturelle 10](#_Toc524078323)

[IV.G. Auto-jointure 10](#_Toc524078324)

[IV.H. Autres jointures. 11](#_Toc524078325)

[IV.H.1°) Jointure Externe. 11](#_Toc524078326)

[V. Les expressions et les fonctions 12](#_Toc524078327)

[V.A. Les fonctions agrégatives 12](#_Toc524078328)

[V.B. Les expressions arithmétiques. 13](#_Toc524078329)

[V.C. Les fonctions prédéfinies. 13](#_Toc524078330)

[VI. Union 14](#_Toc524078331)

[VII. Intersection et différence 15](#_Toc524078332)

[VII.A. L’intersection. 15](#_Toc524078333)

[VII.B. La différence. 15](#_Toc524078334)

## Notations utilisées lors de l'écriture des ordres SQL

* Les mots réservés SQL sont imprimés en lettres majuscules.
* Les noms symboliques ou expressions spécifiques à écrire par le programmeur sont imprimées en caractères minuscules.
* Les expressions facultatives sont éditées entre crochets [].
* Lorsqu'on peut choisir entre plusieurs expressions mais que, au moins l'une d'elles est obligatoire, elles sont regroupées entre accolades {}.
* L'alternative entre expressions est symbolisée par |.

# La projection.

## Choix des colonnes.

La projection consiste à extraire les colonnes spécifiées d'une table puis éventuellement élimine les lignes en double.

Une projection s'exprime à l'aide du langage SQL par la clause:

SELECT [ALL|DISTINCT] {[colonne1 [, colonne2, …]] | \* }

FROM nomTable;

ALL est l'option par défaut: elle affiche toutes les lignes (même les doublons).

|  |  |
| --- | --- |
| **\*** |  |

L’exemple suivant fournit toutes les lignes et toutes les colonnes (\*) de la table *produit* dans la base de données *Commande :*

SELECT \* FROM produit;

L’exemple suivant fournit toutes les lignes et uniquement un sous-ensemble des colonnes (idProduit, nom du produit et Prix Unitaire) de la table produits dans la base de données.

SELECT idProduit, nomProd, Prix

FROM produit ;

SQL n'élimine pas les doublons à moins que ceci soit explicitement demandé par le **mot DISTINCT**. On obtient ici la liste des codes des clients ayant déjà commandé **sans doublon.**

SELECT **DISTINCT** idCli

FROM commande;

Cette requête montre **une utilisation inutile** de DISTINCT car on n’aura jamais deux lignes de la table fournisseur ayant même numéro et même nom.

SELECT DISTINCT idFournisseur, nomFour

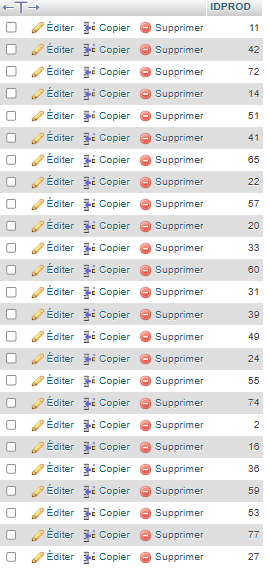
FROM fournisseur;

Application:



##### Liste des produits commandés (identifiant).

Select DISTINCT IDPROD from detailcde ;



##### Affichez les contacts des fournisseurs.

Select NOMCONTACT from fournisseur ;



##### Liste des clients avec toutes leurs informations

Select \* from client ;



##### Liste des expéditeurs (numéros) responsables de la livraison d’une commande

Select DISTINCT IDEXPEDITEUR from commande ;



## Titre de colonne

Lorsque les résultats de la requête sont affichés, le nom donné à une colonne lors de sa création est utilisé comme titre par défaut.

#### Renommer une colonne

Pour obtenir des résultats plus lisibles, vous pouvez spécifier un autre titre de colonne en choisissant *nom\_de\_colonne* AS *titre\_de\_colonne* au lieu d'utiliser le nom de colonne dans la liste de sélection.

Par exemple, pour remplacer nomproduit par Désignation dans la requête précédente:

SELECT nomproduit AS '*Désignation*', idProduit AS ‘RefProduit’

FROM produit ;

Le résultat de l’instruction apparaît comme suit :

|  |  |
| --- | --- |
| Désignation | RefProduit |
| Tourtière | 54 |
| Pâté chinois | 55 |
| Ravioli Angelo | 57 |
| Escargots de Bourgogne | 58 |
| Sirop d'érable | 61 |
| Tarte au sucre | 62 |

#### Colonne calculée.

Vous pouvez utiliser des opérateurs arithmétiques pour effectuer des calculs impliquant constantes et/ou colonnes. Les opérateurs autorisés sont +, -, \*, /.

Exemple :

SELECT idProduit, nomProduit, Prix\*1.196 AS 'PuTTC'

FROM produit;

Application.



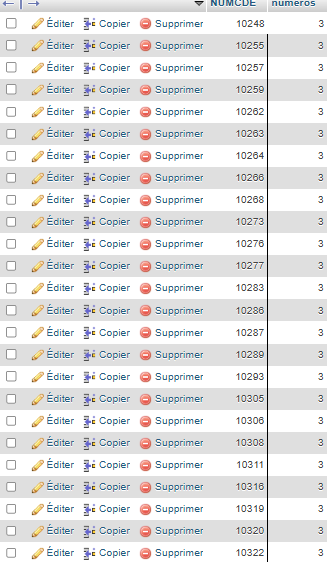
##### Liste des catégories.

Select NOMCAT as categorie from CATEGORIE



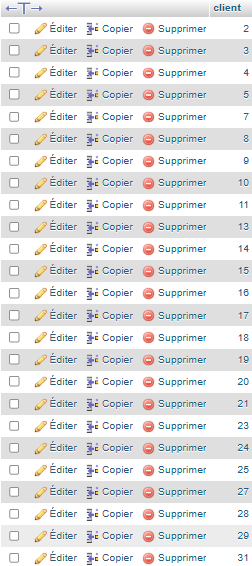
##### Liste des commandes (numéros) avec l’expéditeur n°3 (indiquez l’identifiant).

Select DISTINCT NUMCDE, IDEXPEDITEUR as numéros FROM commande WHERE IDEXPEDITEUR = 3



##### Liste des clients (id) ayant passé une commande

Select distinct IDLCI as client from commande



# Sélection ou restriction : Clause WHERE.

# Description of where_clause.gif follows

La clause WHERE spécifie les critères des lignes à extraire. La syntaxe est la suivante:

SELECT \*

FROM NomTable

WHERE condition;

Cet exemple fournit la liste des produits de la catégorie 3:

SELECT \*

FROM Produits

WHERE idCat=3;

La combinaison d'une restriction avec une projection s'effectue en remplaçant \* par la liste des colonnes à projeter.

La même requête avec l'affichage du nom du produit et le prix unitaire s'écrit:

SELECT nomProd, Prix

FROM Produit

WHERE codecatégorie=3;

De manière plus générale, la condition suivant la clause WHERE peut inclure :

 Des opérateurs de comparaison (tels que =, < >, <, >, <=, >=)

WHERE remise > 0.1

 Des intervalles (BETWEEN et NOT BETWEEN)

WHERE remise BETWEEN 0.1 AND 0.2

 Des listes (IN, NOT IN)

WHERE nomCategorie IN ('HIFI', 'INFO', 'VIDEO')

 Des concordances avec des modèles (LIKE et NOT LIKE) valables pour les CHAINES DE CARACTERES pas les DATES!

WHERE CpFour LIKE '75%'

Vous pouvez utiliser les caractères Joker (génériques) suivants:

% remplace de 0 à n caractères

\_: remplace un caractère et un seul.

 Des tests pour vérifier si les champs sont renseignés ou non (IS NULL et IS NOT NULL)

WHERE Remise IS NULL

 Des combinaisons de critères (avec AND, OR)

WHERE remise >0.1 AND remise<0.2

 Les opérateurs ANY et ALL (utilisés dans les sous-requêtes)

Cet exemple montre toutes les informations contenues dans la table Produitoù le *Prix de vente d’un* produit vendu est supérieur à 20€.

SELECT \*

FROM Produit

WHERE Prix > 20;

Cet exemple fournit uniquement les lignes pour lesquelles un contact fournisseur a été renseigné :

SELECT \*

FROM Fournisseur

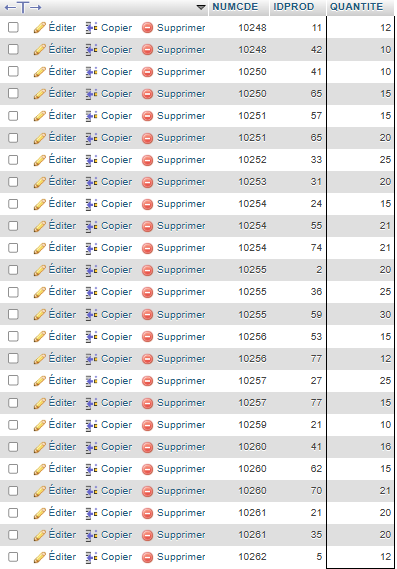
WHERE nomContact IS NOT NULL;

Application.



##### Liste des lignes de commandes avec quantité entre 10 et 30

##### Select \* from detailcde where QUANTITE BEETWEEN 10 and 30



##### Liste des clients (Nom, cp, ville) qui habitent une ville dont le nom commence par B ainsi que ceux dont le nom commence par T.

Select

##### Liste des produits (référence, nom) vendus par le fournisseur n°12

##### Liste des commandes passées en septembre 2018

# Tri du résultat : Clause ORDER BY.

## 

La clause ORDER BY trie les résultats d'une requête sur une ou plusieurs colonnes (maximum 16). Un tri peut être effectué dans un ordre croissant (ASC) ou décroissant (DESC). Si l'ordre de tri n'est pas spécifié, le tri par défaut est croissant.

La requête suivante renvoie des résultats triés sur la colonne NumCde:

SELECT \*

FROM DetailCde

WHERE quantite > 10

ORDER BY NumCde;

Si vous spécifiez plusieurs colonnes dans la clause ORDER BY, les tris sont imbriqués.

L'instruction suivante trie les lignes de la table *DetailCde*, d'abord par date de commande en ordre décroissant (les plus récentes en premier), ensuite par ordre croissant sur les numéros de commande. Les valeurs NULL sont triées en premier lieu:

SELECT numCde, idCli, dateCde,

FROM DetailCde

WHERE quantite > 10

ORDER BY dateCde DESC, NumCde;

Vous pouvez utiliser le numéro de position d'une colonne dans la liste de sélection à la place du nom de colonne. Vous pouvez mélanger des noms de colonne et des numéros de la liste de sélection.

L'instruction suivante fournit le même résultat que la précédente :

SELECT numCde, idCli, dateCde,

FROM DetailCde

WHERE quantite > 10

ORDER BY 3 DESC, 1;

**Application de tris.**



##### Liste des commandes à livrer en septembre 2018 triées en commençant par les commandes à livrer en premier.

##### Liste des produits (Nom du Produit,Reference du Produit) classés par fournisseurs ( NumFour décroissant). Pour chaque fournisseur, ils seront classés par référence de produit.

##### Liste des clients de France du département Loiret (le code postal commence par 45) triés par ville.

# Jointure interne – INNER JOIN

## Définition.

La jointure interne est l’opération qui consiste à rapprocher ligne à ligne des informations issues de deux ou plusieurs tables.

Une instruction de jointure interne:

 utilise une colonne dans chaque table;

 compare les valeurs de ces colonnes ligne par ligne;

 forme de nouvelles lignes en combinant les lignes qui contiennent les valeurs retenues dans la comparaison.

Bien que la comparaison soit souvent basée sur un critère d'égalité (comparaison de valeurs identiques), on pourra utiliser des opérateurs <, <=, >, >=, <>.

## Exemple

La requête suivante affiche les informations sur chaque commande et le client l'ayant effectué.

SELECT \*

FROM client

INNER JOIN commande

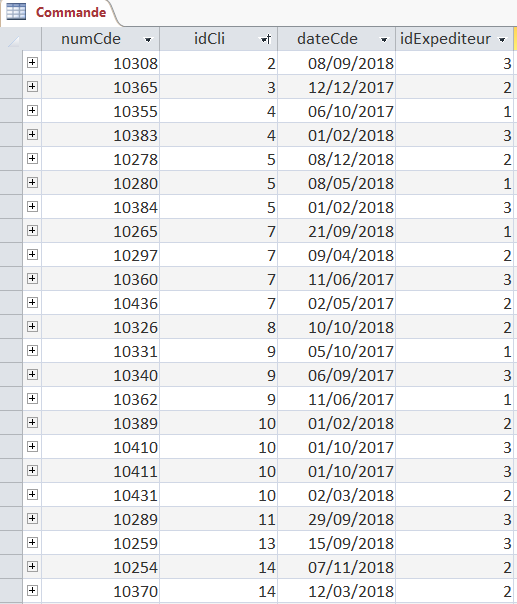
ON commande.idCli=client.idClient;

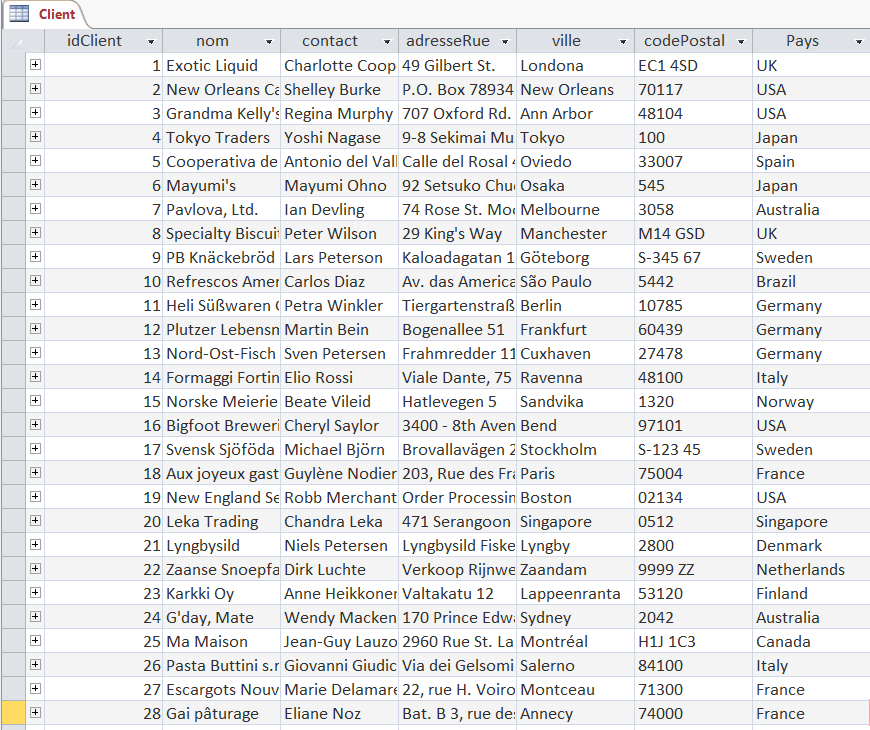
Remarques :

INNER est facultatif, on peut simplement écrire **JOIN**.

Le préfixage de la colonne CodeClient par le nom de table est obligatoire car cette colonne a la même appellation dans les tables Clients et Commandes.

Tables d'origine





Résultat de la requête:

Informations sur le client ayant effectué la commande (Commande.idCli=Client.idClient

client.idClient

commande.idCli



Informations sur chaque commande

Remarque: les clients n'ayant réalisé aucune commande n'apparaissent pas (ils sont éliminés par la jointure interne).

## Combinaison de la jointure avec la projection.

Lorsqu'une requête extrait certaines informations contenues dans des tables différentes, il faut une jointure pour récupérer les informations demandées et une projection pour ne garder que certaines informations.

Par exemple, pour extraire les produits (idProduit, NomProduit) classés par nom de fournisseurs en indiquant le numéro du fournisseur, il faut exécuter l'instruction suivante :

SELECT idProduit, NomProd, fournisseur.idFournisseur, NomFour

FROM produit

INNER JOIN fournisseur

ON produit.idfournisseur=fournisseur.idFournisseur

ORDER BY 4;



Remarque: préfixage de la colonne NumFour.

## Combinaison de la jointure avec la restriction.

Par exemple, pour extraire les noms des fournisseurs (triés par ordre alphabétique) des produits de nom de catégorie='Photo', il faut exécuter l'instruction suivante :

SELECT DISTINCT nomFour

FROM produit

INNER JOIN categorie

ON categorie.idCategorie=produit.idCat

INNER JOIN fournisseur

ON fournisseur.idFournisseur=produit.idFournisseur

WHERE nomCate='Seafood'

ORDER BY 1;

Notez que les colonnes jointes n'ont pas besoin de figurer dans la liste de sélection (elles n'apparaissent donc pas dans les résultats).

## Synonymes ou alias.

Vous pouvez définir des synonymes pour les tables. Dans ce cas, le nom complet de la table ne peut pas être utilisé si un synonyme a été défini. Pour utiliser un synonyme, spécifiez le nom d'objet, un espace et ensuite le synonyme, comme ci-dessous :

SELECT idProduit, NomProduit, f.idFournisseur, NomFour

FROM produit p

INNER JOIN fournisseur f

ON p.idfournisseur=f.idFournisseur;

**Application sur les jointures internes.**



##### Liste des commandes (commencez par les plus récentes) qui contiennent le produit n°15.

##### Liste des fournisseurs (toutes leurs informations) du produit n°15.

##### Noms et ville des clients qui ont commandé le produit n°15 en 2018.

##### Liste des fournisseurs des produits de la catégorie Condiments (nom de la cat.) triés par nom.

## Jointure naturelle

Elle permet d'éviter de devoir spécifier le critère de jointure. Le compilateur SQL se charge de rechercher dans les 2 tables les champs qui ont un nom identique.

SELECT \*

FROM produit

NATURAL JOIN fournisseur;

équivaut donc à :

SELECT \*

FROM produit

JOIN fournisseur

ON produit.numFournisseur = fournisseur.numFournisseur;

## Auto-jointure

#### Définition

Une *auto-jointure* sert à comparer des valeurs à l'intérieur de colonnes d'une même table. Par exemple, vous pouvez utiliser une auto-jointure pour déterminer les produits ayant le même fournisseur que le produit n°15.

Comme cette requête effectue une jointure de la table *produits* avec elle-même, la table joue deux rôles. Pour distinguer ces rôles, vous pouvez temporairement (et arbitrairement) donner deux noms d'emprunt à cette table  p*1* et p*2*. Ces noms d'emprunt qualifient les noms de colonne pour la suite de la requête. Voici l'exemple d'instruction d'auto-jointure répondant au besoin exprimé:

SELECT p1.idProduit, p1.description  
FROM produit p1

INNER JOIN produit p2

ON p2.idFournisseur=p1.idFournisseur

WHERE p2.idProduit=’15’

AND p1.idProduit<>’15’;

Le dernier critère permet d'éliminer l'affichage du produit n°15 qui a bien sûr le même fournisseur que lui-même.

**Applications : autojointures**



##### Liste des commandes livrées par le même expéditeur que la commande n°14.

##### Liste des produits ayant la même catégorie que le produit 14.

## Autres jointures.

### Jointure Externe.

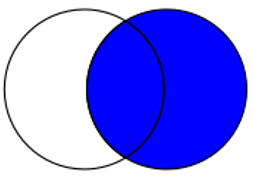
Les jointures externes sont plus "complètes" que les jointures internes. En plus de sélectionner l'ensemble des lignes des 2 tables respectant le critère de jointure comme le font les jointures internes, elles sélectionnent les données d'une des tables qui ne respectent pas le critère de jointure.

#### LEFT JOIN

SELECT \*

FROM TableGauche

LEFT JOIN TableDroite



|  |  |
| --- | --- |
| ON critère de jointure |  |

Le **LEFT JOIN** (ou LEFT OUTER JOIN) implique que l'on sélectionne toutes les lignes respectant le critère de jointure, puis on ajoute toutes les lignes de la table "**TableGauche**" qui ont été rejetées car elles ne respectaient pas le critère de jointure.

Appliquons le LEFT JOIN à l'exemple utilisé pour les jointures internes :

SELECT \*

FROM client

LEFT JOIN commande

ON commande.idCli=client.idClient;

**Résultat**: tous les enregistrements de la table de gauche (Client) apparaissent y compris ceux qui n’ont rien commandé.



Remarque: Cette jointure sera très pratique pour afficher tous les clients avec leur nombre de commandes (y compris ceux n'ayant aucune commande, éléminés par la jointure classique)

#### RIGHT JOIN

Implique que l'on sélectionne toutes les lignes respectant le critère de jointure, puis on ajoute toutes les lignes de la table "**TableDroite**" qui ont été rejetées car elles ne respectaient pas le critère de jointure.

# Les expressions et les fonctions

Une expression est une combinaison de variables, de constantes, de fonctions au moyen d'opérateurs.

Une fonction est une routine (sous-programme) utilisant des arguments (paramètres) qui peuvent être des variables, des constantes, des expressions ou des fonctions.

**Les expressions et les fonctions peuvent être utilisées dans les clauses SELECT, WHERE, ORDER BY, GROUP BY, HAVING**.

## C:\Users\estelle\AppData\Local\Microsoft\Windows\INetCache\Content.MSO\60B88451.tmp Les fonctions agrégatives

Elles permettent d'effectuer des **calculs verticaux.** Par exemple, sur l'ensemble ou un sous-ensemble des valeurs d'une colonne.

Les fonctions disponibles sont:

**SUM** : permet d'effectuer la somme des valeurs d'une colonne numérique,

**AVG** : permet d'effectuer la moyenne des valeurs d'une colonne numérique,

**MAX**: permet de rechercher la valeur maximale d'une colonne numérique,

**MIN:** permet de rechercher la valeur minimale d'une colonne numérique,

**COUNT**: permet de compter le nombre de valeurs prises par une colonne

pour compter le nombre de lignes sélectionnées, on peut utiliser l'argument \* ( COUNT(\*))

pour compter le nombre de valeurs différentes prises par une colonne, il faut indiquer l'argument **DISTINCT** suivi du nom de la colonne.

Exemple: Calcul du prix moyen des produits:

SELECT AVG(Prix)  
FROM Produit;

**Applications:**



##### Afficher le prix le plus élevé pour le produit dont l’identifiant est '15'.

##### Quantité moyenne commandée par le client n°14.

##### Nbre de clients ayant commandé en janvier 2018.

##### Afficher le nombre de commandes passées par le client N°15

## Les expressions arithmétiques.

Il est possible d'effectuer dans une requête SQL, des calculs horizontaux (=sur des lignes) en utilisant les opérateurs : +, - , \* , / aussi bien dans la clause SELECT que dans un WHERE (expression d'une condition sur des valeurs calculées). Les arguments des opérateurs sont des noms de colonnes de type numérique et ou des constantes.

Exemple : Quelles sont les produits dont le Prix TTC est supérieur à 100 euros?

SELECT \*  
FROM Produit  
WHERE prix\*1.196>100;

## Les fonctions prédéfinies.

Pour SQL server:

* GETDATE() pour obtenir la date système,
* MONTH(date) pour obtenir le mois d'une date,
* YEAR(date) pour obtenir l'année,
* Différences entre deux dates: d1-d2 : donne le nombre de jours entre d1 et d2
* ROUND(n) pour arrondir un nombre.

Pour MySQL

* CURDATE() et CURRENT\_DATE()
* MONTH(date) pour obtenir le mois d'une date,
* YEAR(date) pour obtenir l'année,
* DATEDIFF(date1, date2) :calcule le nombre de jours entre la date1 et la date2.
* ROUND(nb,2) pour arrondir un nombre réel à 2 décimales.

Les fonctions ROUND s'utilisent différemment selon le SGBDR: voir la documentation en ligne.

**Application:**



##### Liste des produits commandés en 2018

##### Liste des commandes dont la date de commande est hier

# Union

Associe les résultats de plusieurs requêtes en un ensemble de résultats constitué de toutes les lignes appartenant aux requêtes de l'union.

Cette requête affiche l'ensemble des noms des clients et des fournisseurs.

SELECT Nom FROM Client

UNION

SELECT NomFour FROM Fournisseur;

Application : liste des clients (idClient, Nom habitant Tokyo ainsi que ceux habitant Melbourne.



Toutes les listes de sélection des instructions doivent contenir le même nombre d'expressions (noms de colonne, expressions arithmétiques et fonctions d'agrégation). Par exemple, l'instruction suivante n'est pas valide car la première liste de sélection est plus longue que la seconde:

SELECT \* FROM Client

UNION

SELECT NomFour FROM Fournisseur;

# Intersection et différence

## L’intersection.

L'opérateur d'intersection est INTERSECT. L’intersection compare les résultats de plusieurs requêtes et retourne uniquement les lignes communes à toutes les requêtes de l'intersection

Exemple: Liste des commandes (Numéro et date) comportant des produits de catégorie 2 et de catégorie 3.

SELECT c.NumCde, DateCde

Question: la requête suivante donne-t-elle le même résultat?

SELECT c.NumCde, DateCde

FROM Commandes c

INNER JOIN DetailCommande d

ON c.NumCde=d.NumCde

INNER JOIN Produit p

ON d.idProduit=p.idProduit

WHERE idCat=2 and idCat=3;

Non, car aucun produit n'appartient aux deux catégories à la fois

FROM Commande c

INNER JOIN DetailCommande d

ON c.NumCde=d.NumCde

INNER JOIN Produit p

ON d.idProduit=p.idProduit  
WHERE idCat=2  
**INTERSECT**

SELECT c.NumCde, DateCde

FROM Commande c

INNER JOIN DetailCommandes d

ON c.NumCde=d.NumCde

INNER JOIN Produit p

ON d.idProduit=p.idProduit  
WHERE idCat=3;

## La différence.

L'opérateur de différence est MINUS. Il permet de retourner les lignes présentes dans le résultat de la première requête et absentes du résultat de la deuxième requête.

Exemple 1: Liste des commandes (Numéro et date) comportant des produits de la catégorie 2 sans comporter des produits de la catégorie 3.

SELECT c.NumCde, DateCde

FROM Commande c

INNER JOIN DetailCommande d

ON c.NumCde=d.NumCde

INNER JOIN Produit p

ON d.idProduit=p.idProduit  
WHERE idCat=2  
**MINUS**

SELECT c.NumCde, DateCde

FROM Commande c

INNER JOIN DetailCommande d

ON c.NumCde=d.NumCde

INNER JOIN Produit p

ON d.idProduit=p.idProduit  
WHERE idCat=3;

Remarque:

* Les opérateurs INTERSECT et MINUS ne sont ni disponibles sous Transact-SQL (langage SQL de SQLServer) , ni sous MySQL: vous utiliserez des sous-requêtes avec IN ou NOT IN.