

TP MYSQL et BDD révisions

Notions abordées :

- le vocabulaire des bases de données et vocabulaire : SGBD, base de données, table, champ, dépendance fonctionnelle, clef primaire, clef étrangères, contrainte d'intégrité ;
- modélisation de base de données et dictionnaire des données ;
- modélisation de base de données et schéma relationnel : relation, attribut, relation un à plusieurs et relation plusieurs à plusieurs ;
- manipulation de base de données en SQL et gestion des droits d'accès en SQL.

1. Qu'est-ce qu'une base de données ?

1.1. Le vocabulaire

SGBD	Un SGBD (Système de Gestion de Bases de Données) est un logiciel qui permet de gérer des bases de données. Lorsqu'il gère des bases de données relationnelles, on parle de SGBDR (Système de Gestion de Bases de Données relationnelles).
Base de données	Une base de données est un ensemble organisé de données stocké sur un support informatiques. Une base de données permet en outre de stocker et de manipuler les
Table ou relation	Une base de données est divisée en tables ou relations ayant chacune un nom. Une table peut être vue comme un tableau à deux dimensions, constitué de lignes et de colonnes. Les colonnes sont appelées les champs ou les attributs . Les lignes sont également appelées tuples ou occurrences . Chaque champ correspond à une donnée élémentaire (=atomique) ayant un certain type.
Types de données	Chaque champ d'une table a un type. Les types de données courant sont : <ul style="list-style-type: none">- alphanumérique (=chaîne de caractères). On distingue souvent les chaînes dont le nombre de caractères est fixe ou variable. On peut communément imposer un nombre maximum de caractères ;- numérique. On distingue par exemple les sous-types suivants : nombre entier ou nombre décimal ;- date et heure. On distingue par les sous-types suivants : date seule, heure seule, date et heure. On notera qu'il existe de nombreuses façons de représenter les dates et/ou les heures (exemple : 12/01/2017, lundi 9 janvier

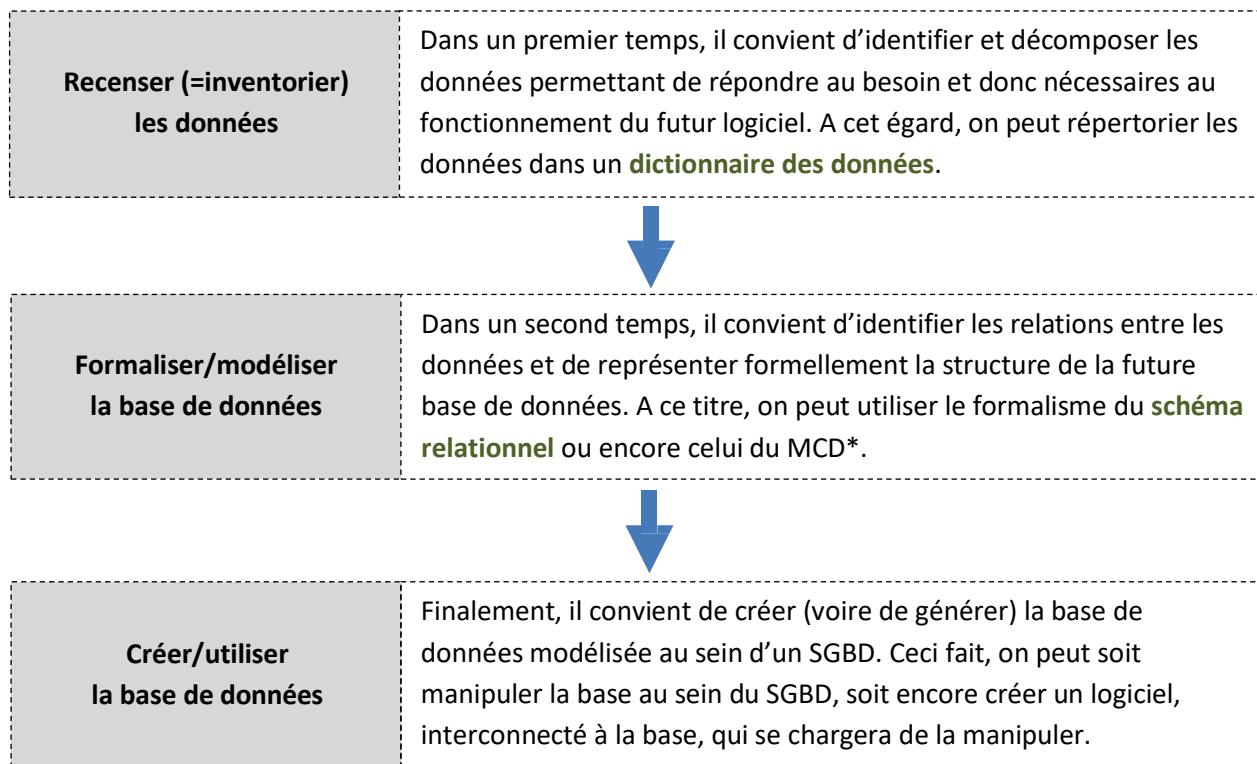
TP MYSQL et BDD révisions

Contrainte d'intégrité	Toutes les contraintes que doivent respecter les données sont appelées contraintes d'intégrité référentielle , ou plus simplement contraintes d'intégrité. Le respect du type de donnée est un exemple de contrainte
Clef primaire	Toute table ou relation doit avoir une clef primaire. La clef primaire est constituée d'un ou plusieurs champs. C'est ce champ ou ce sont ces champs qui sont qualifié de clef primaire. La clef primaire permet et doit permettre d'identifier de manière unique chaque ligne d'une table.
Clef étrangère	Une table ou relation peut avoir un champ qui fait office de lien vers une autre table, à savoir un champ faisant référence à la clef primaire d'une autre table. Un tel champ est appelé clef étrangère . De même qu'une clef primaire, une clef étrangère peut être constituée d'un ou plusieurs champs.
Dépendance fonctionnelle	Toute ligne d'une table est identifiable à la valeur de sa clef primaire. La valeur des autres champs de la ligne est en quelque sorte rattachée à la valeur de la clef primaire. On dit que ces autres champs sont en dépendance fonctionnelle avec la clef primaire, c'est-à-dire qu'ils lui sont rattachés.

L'on notera que ces concepts sont réinvestis et illustrés plus loin.

1.2. La conception

La conception d'une base de données passe typiquement par les étapes suivantes :



* Le MCD (Modèle Conceptuel des Données) est un schéma permettant de représenter graphiquement la structure d'une base de données. Nous n'en étudierons pas le formalisme.

TP MYSQL et BDD révisions

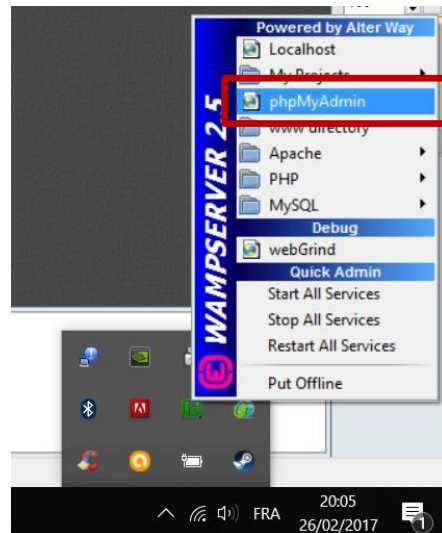
1.3. PhpMyAdmin et MySQL

De quoi s'agit-il ?

Nous citons un peu plus haut MySQL. **MySQL est un SGBD**, c'est-à-dire un logiciel qui permet de gérer et de manipuler des bases de données. Cependant, en soi, l'utilisation directe de MySQL n'est pas très conviviale...

Aussi, on peut utiliser le logiciel PhpMyAdmin. Pour l'obtenir, téléchargez par exemple le logiciel WampServer, lequel est disponible en téléchargement sur le site de l'éditeur.

PhpMyAdmin n'est pas un SGBD mais un logiciel qui permet de manipuler MySQL de manière plus conviviale, à savoir grâce à de charmantes interfaces web. Une fois WampServer démarré, vous pouvez accéder à PhpMyAdmin au moyen de votre navigateur favori à l'adresse : <http://localhost/phpmyadmin>.



A quoi cela ressemble-t-il ?

Cliquer ici pour créer une nouvelle base de données.

Cliquer sur la table pour en voir la structure ou encore le contenu.

Cliquer sur l'onglet « SQL » pour accéder à l'éditeur de requêtes. Cette interface est sans conteste la plus importante. En effet, vous pouvez tout y faire... pourvu que vous sachiez comment...

Le volet de gauche vous permet de visualiser et sélectionner les bases de données disponibles. Il vous permet également de voir les tables constituant chacune des bases de données. En l'occurrence, on voit 1 seule base de données (*db_gest_commandes*) et ses 4 tables.

A quoi ressemble une table ?

Afficher		Structure		SQL	Rechercher	Insérer
#	Nom	Type	Interclassement	Attributs	Null	Défaut
<input type="checkbox"/>	1 Code	varchar(7)	latin1_swedish_ci		Non	
<input type="checkbox"/>	2 Libelle	varchar(80)	latin1_swedish_ci		Non	Aucune
<input type="checkbox"/>	3 Prix	decimal(6,2)			Non	Aucune
<input type="checkbox"/>	4 Description	text	latin1_swedish_ci		Oui	NULL

A gauche, on voit la structure d'une table, la table « produit ». Par structure, on entend : les champs de la table, leur type, leur valeur par défaut, etc. Ici, la table est constituée de 4 champs, dont une clef primaire, soulignée, le champ « Code ». Regardons-en à présent le contenu, dans l'onglet « Afficher ».

TP MYSQL et BDD révisions

	Afficher	Structure	SQL	Rechercher	Insérer	Exporter	Importer
	←T→	Code	Libelle	Prix	Description		
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	ALPHA	Téléphone mobile Samsung Galaxy Alpha	199.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	CART1	Pack de cartouches encre HP 364 XL	59.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	CART2	Cartouche encre HP 364 Noire	19.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	CART3	Cartouche encre HP 364 Cyan	19.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	CART4	Cartouche encre HP 364 Magenta	19.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	CART5	Cartouche encre HP 364 Jaune	19.99	NULL			
<input type="checkbox"/> Modifier <input type="checkbox"/> Copier <input type="checkbox"/> Effacer	HTCONE	Téléphone mobile HTC ONE	179.50	NULL			

Avec cette image, on constate :

- Qu'une table prend la forme d'un tableau structuré où chaque champ correspond à une colonne ;
- Qu'une table a une ou plusieurs colonnes correspondant à la clef primaire. En l'occurrence, il s'agit de la colonne « Code » ;
- Que **la clef primaire doit permettre d'identifier chaque ligne de manière unique**. Dans l'exemple ci-dessus, on ne peut avoir deux produits avec le même code. Il s'agit d'une **contrainte d'intégrité**, c'est-à-dire une règle qui doit être vérifiée pour que les données restent cohérentes ;
- Que les champs « Libelle », « Prix » et « Description » sont en **dépendance fonctionnelle** avec le champ « Code ». Qu'est-ce que cela signifie ? Cela signifie par exemple que « Cartouche encre HP 364 Magenta » est le libellé du produit dont le code est « CART5 ». En d'autres termes, **les valeurs sur une même ligne sont rattachées à la valeur de la clef primaire de la ligne**.

Que peut-on faire avec PhpMyAdmin ?

Avec PhpMyAdmin, on peut en outre demander à MySQL d'exécuter des requêtes SQL. Nous étudierons plus loin les principales requêtes SQL standards. Quoiqu'il en soit, une requête SQL permet d'effectuer une opération sur une base de données : créer une table, insérer une ligne, consulter et/ou calculer des données, etc.

PhpMyAdmin permet de plus de visualiser de manière plutôt conviviale le résultat de ces fameuses requêtes.

Afficher

Structure

SQL

Rechercher

Montrer zone SQL

⚠ La sélection courante ne contient pas de colonne unique. Les gr

Votre requête SQL a été exécutée avec succès

```
SELECT count(code) AS "Nombre de produits" FROM produit
```

+ Options

Nombre de produits
20

Requête SQL : récupère le nombre de produits.

Résultat de la requête : 20 produits

TP MYSQL et BDD révisions

2. Comment modéliser une base de données ?

2.1. Le dictionnaire des données

Le **dictionnaire des données** est une représentation tabulaire de la structure d'une base de données. Il consiste à énumérer tous les champs de toutes les tables d'une base de données. Il prend par exemple la forme suivante :

Champ	Type	Longueur	Vide ?	Par défaut	Description
uneChaine	chaîne	10	oui		Chaîne de car. à 10 car. maximum
unEntier	entier	1	non	0	Entier compris entre -127 et 128
unDecimal	décimal	5,2	non	0.00	Décimal à 5 chiffres, 2 après la virgule
...

Champ : nom du champ, à savoir dénomination de la donnée à stocker ;

Type : type de donnée du champ (chaîne de caractères, entier, décimal, date, heure, etc.) ;

Longueur : pour une chaîne de caractères, longueur maximale de la chaîne ; pour entier, en général, le nombre d'octets sur lequel est codé l'entier ; pour un nombre décimal, le nombre de chiffres total suivi du nombre de chiffres après la virgule ;

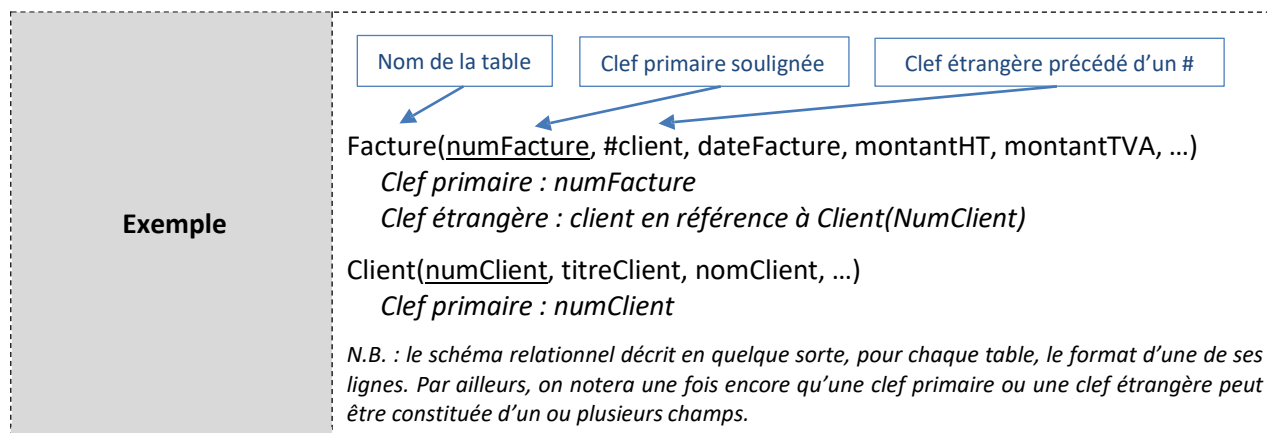
Vide : indique si le champ peut être vide, c'est-à-dire avoir la valeur *null* ;

Par défaut : valeur que le champ prend par défaut ;

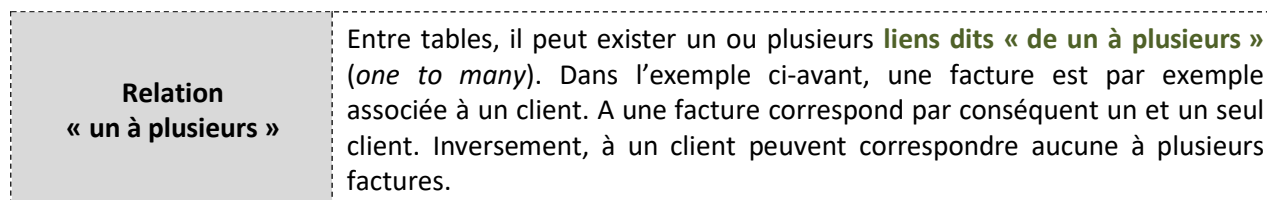
Description : précise la signification du champ à stocker.

2.2. Le schéma relationnel

Le **schéma relationnel** est une représentation qui permet de mettre en évidence les relations entre les données. Ce schéma a une syntaxe, un formalisme, qui lui est propre. Et vous devez le connaître. Dans un schéma relationnel, les tables sont appelées « **relations** » et les champs appelés « **attributs** ».



Les relations entre tables sont dictées par les clefs primaires et clefs étrangères.



TP MYSQL et BDD révisions

Exemple de relation « un à plusieurs » :

Table « Facture »					Table « Client »		
numFacture	client	dateFacture	montantHT	...	numClient	nomClient	...
1	1	05/01/2017	500,00		1	Nestle	
2	2	05/01/2017	750,00		2	Vinci	
3	2	06/01/2017	2500,00	
4	1	06/01/2017	600,00				
...			

On comprend que le « 2 » figurant le champ « client » (clef étrangère) de la table champ renvoie vers le client « 2 » de la table « Client », c'est-à-dire Vinci.

Remarque ! Au regard de l'exemple ci-dessus, on comprendra que :

- Une facture a 1 et 1 seul client. Toutefois, si l'on accepte que le champ « client » puisse être vide, c'est-à-dire qu'il puisse prendre la valeur NULL, alors une facture pourra avoir 0 ou 1 client ;
- Un client peut avoir 0 à plusieurs factures. Et il a autant de factures que de lignes où son « numClient » apparaît dans la colonne « client » de la table « Facture » ;
- **Une clef étrangère fait toujours référence à une clef primaire.**

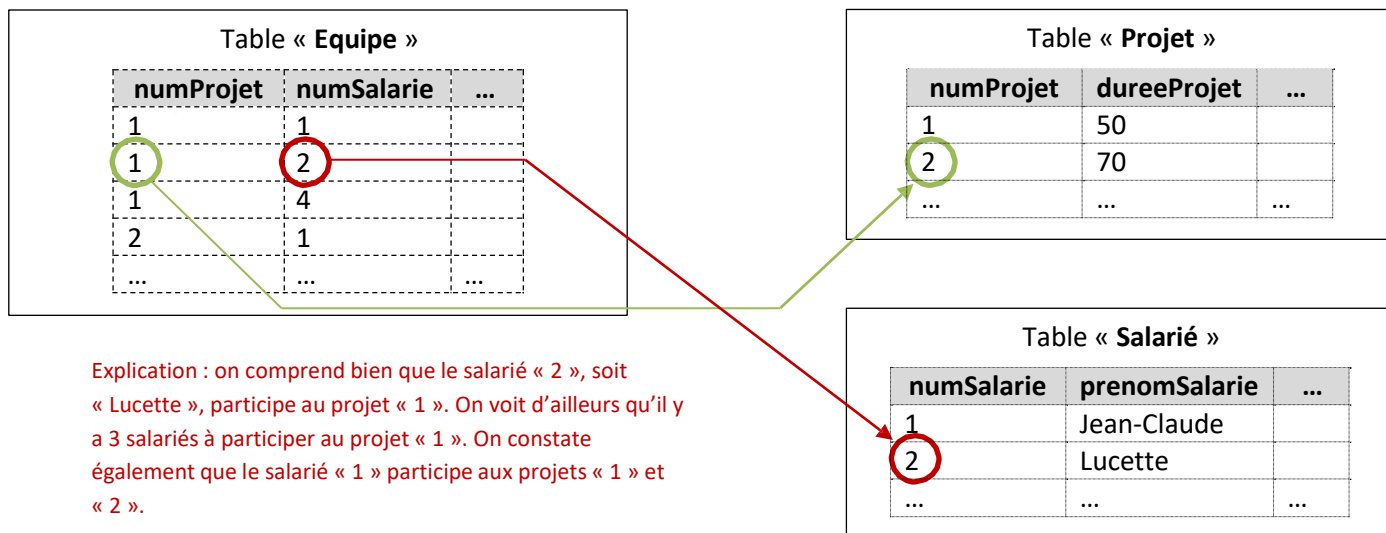
Relation « plusieurs à plusieurs »	Entre tables, il peut également exister un ou plusieurs liens dits « de plusieurs à plusieurs » (<i>many to many</i>). Par exemple, un salarié peut travailler sur plusieurs projets. Inversement, plusieurs salariés peuvent travailler sur le même projet. On obtiendra un schéma relationnel de la forme suivante :
	Salarie(<u>numSalarie</u> , prenomSalarie, nomSalarie, ...) Clef primaire : numSalarie
	Equipe(<u>#salarie</u> , <u>#projet</u>) Clef primaire : salarie, projet Clefs étrangères : - salarie en référence au champ numSalarie de la relation Salarie - projet en référence au champ numProjet de la relation Projet
	Projet(<u>numProjet</u> , dureeProjet, prixProjet, ...) Clef primaire : numProjet
N.B. : dans le cadre d'une relation « plusieurs à plusieurs », l'on a toujours une table « centrale » avec une clef primaire composée. Ici, la table centrale est « Equipe », qui fait la relation entre « Salarié » et « Projet ».	

Remarque ! Au regard de l'exemple fourni ci-après, on comprendra que :

- Plusieurs salariés peuvent participer à un même un projet et un salarié peut participer à plusieurs projets, d'où l'appellation « relation de plusieurs à plusieurs » ;
- Que la table centrale « Equipe » stocke la composition des équipes (couples Salarié + Projet). La clef primaire est ainsi constituée de deux champs, l'un identifiant le projet, l'autre identifiant le salarié.

TP MYSQL et BDD révisions

Exemple de relation « plusieurs à plusieurs » :



3. Comment manipuler une base de données ?

Important ! Ci-après figurent divers exemples de requêtes. Pour les tester :

- Ouvrir le fichier « SQL » joint au présent cours au moyen de NotePad++ ;
- Copier le contenu du fichier dans l'onglet « SQL » de PhpMyAdmin, puis cliquer sur « Exécuter » ;
- Vous pouvez désormais tester vos requêtes sur la base « db_gest_commande » nouvellement créée.

3.1. Le langage SQL

SQL est un langage qui permet entre autres :

- De créer, modifier et supprimer des tables d'une base de données préalablement sélectionnée. On parle de langage de définition des données ;
- De consulter, insérer, modifier et supprimer des lignes des tables d'une base de données préalablement sélectionnée. On parle alors de langage de manipulation des données.

3.2. Les types de données

Nous l'avons déjà vu, chaque champ d'une table a un type. Voici quelques-uns des types disponibles sur les SGBDR, dont MySQL :

Type	Type SQL	Description
Alphanumérique	CHAR(n)	Chaîne de caractères de longueur fixes (n caractères)
	VARCHAR(n)	Chaîne de caractères de longueur variable (n car. max)
Numérique	INTEGER	Entier de -2^{31} à $2^{31}-1$ (entier sur 32 bits)
	FLOAT	Décimal (parties entière et décimale séparée par un point)
	DECIMAL(n[,d])	Décimal à n chiffres dont d décimales (d est facultatif)
Date/heure	DATE	Date sous la forme AAAA-MM-JJ
	TIME	Heure sous la forme hh:mm:ss.ns
	TIMESTAMP	Date et heure
Booléen	BOOLEAN	0 ou False pour FAUX, 1 ou True pour VRAI

TP MYSQL et BDD révisions

3.3. La création de table

Créer une table consiste à préciser le nom de la table, à lister les champs de la table avec leur type, puis à préciser la clef primaire et les éventuelles clefs étrangères.

Exemple	Par exemple, pour créer deux tables, Client et Commande, on pourra utiliser les requêtes SQL suivantes :
	<div><div>CREATE TABLE Client(Num INTEGER auto_increment, Civilete VARCHAR(3) NOT NULL, Prenom VARCHAR(30) NOT NULL, Nom VARCHAR(30) NOT NULL, Adresse VARCHAR(80) NOT NULL, CodePostal VARCHAR(5) NOT NULL, Ville VARCHAR(30) NOT NULL, PRIMARY KEY (Num));</div><div>CREATE TABLE Commande(Num INTEGER auto_increment, NumClient INTEGER NOT NULL, Emission DATE NOT NULL, PRIMARY KEY (Num), FOREIGN KEY (NumClient) REFERENCES Client(Num));</div></div> <p><u>Commentaire :</u></p> <ul style="list-style-type: none">○ Les champs « <i>Num</i> » étant ici les clefs primaires de chacune des deux tables, ils sont NOT NULL par convention et il n'est pas utile de le préciser ;○ « <i>auto_increment</i> » permet de préciser que l'entier est un compteur qui augmentera tout seul à chaque insertion de ligne (=enregistrement) dans la table ;○ Le champ « <i>NumClient</i> » est et doit être du même type que la clef primaire à laquelle il fait référence.

3.4. La modification et la suppression de table

Suppression d'une table	DROP TABLE nomTable ;
Ajout d'un champ	ALTER TABLE nomTable ADD champAjouté type[, ...] ;
Ajout d'une clef primaire	ALTER TABLE nomTable ADD PRIMARY KEY (nomChamp1[, ...]) ;
Ajout d'une clef étrangère	ALTER TABLE nomTable ADD FOREIGN KEY (nomChamp1[, ...]) REFERENCES nomTablePointée(nomChampPointé1[, ...]) ;
Suppression d'un champ	ALTER TABLE nomTable DROP champSupprimé[, ...] ;
Suppression de la clef primaire	ALTER TABLE nomTable DROP PRIMARY KEY ;
Suppression d'une clef étrangère	ALTER TABLE nomTable DROP CONSTRAINT nomContrainte ; <i>Commentaire : lorsqu'une clef étrangère est créée, un nom de contrainte lui est attribué. Pour la supprimer cette clef étrangère, il faut utiliser ce nom.</i>

3.5. L'insertion, la modification et la suppression de données

On rappelle qu'on appelle « enregistrement » les lignes d'une table. A cet égard, le langage SQL permet d'insérer de nouveaux enregistrements, de modifier des enregistrements existant mais encore de supprimer des enregistrements. Le format des requêtes SQL est le suivant :

TP MYSQL et BDD révisions

Insertion	INSERT INTO nomTable[(champ1, champ2, ...)] VALUES (valeur1, valeur2, ...), -- 1 ^{ère} ligne ajoutée (valeur1, valeur2, ...), -- 2 ^{ème} ligne ajoutée ... (valeur1, valeur2, ...); -- dernière ligne ajoutée
Modification	UPDATE nomTable SET champModifié = nouvelleValeur [, ...] [WHERE condition1 AND/OR condition2 ...];
Suppression	DELETE FROM nomTable [WHERE condition1 AND/OR condition2 ...];

Bien entendu, rien ne vaut quelques exemples :

Ajout d'un nouveau client
INSERT INTO Client(Civile, Nom, Prenom, Adresse, CodePostal, Ville, Email) VALUES ("M.", "Washington", "Denzel", "Av. de Wagram", "75000", "PARIS", "dw@gmail.com"); Autre possibilité : INSERT INTO Client VALUES (NULL, "M.", "Denzel", "Washington", "Av. de Wagram", "75000", "PARIS", "dw@gmail.com"); <i>Commentaire : si on décide de ne pas préciser la liste des champs, les valeurs doivent être précisée dans l'ordre des champs de la table.</i>
Mise à jour du prix du produit n°1
UPDATE Produit SET Prix = 75.32 WHERE Num = 1
Augmentation de 10% du prix des produits
UPDATE Produit SET Prix = Prix * 1.1
Suppression du produit n°5
DELETE FROM Produit WHERE Num = 5
Suppression des produits dont le prix excède 1000€
DELETE FROM Produit WHERE Prix > 1000
Suppression des clients dont le prénom commence par la letter B
DELETE FROM Produit WHERE Prenom LIKE "B%" <i>Commentaire : le symbole « % » dans la chaîne « B% » permet d'indiquer que le prénom peut commencer par la lettre « B » suivie de n'importe quels caractères. L'opérateur de comparaison « LIKE » ressemble à un « = » mais permet de plus d'effectuer ce genre de comparaisons approximatives.</i>

3.6. La récupération de données

Le cœur du SQL, pourrait-on dire, réside dans les requêtes que nous allons présentement étudier : les requêtes SELECT. On parle de requêtes de **projection**, et souvent, à tort, de sélections.

Ces requête SQL permettent en quelque sorte d'afficher/retourner des données. Elles ont la syntaxe suivante :

TP MYSQL et BDD révisions

Projection	<p>SELECT champProjeté1 [, ...] -- projection de champs</p> <p>FROM table1 [, ...] -- tables utilisées</p> <p>WHERE condition1 [AND/OR condition2 ...] -- restrictions</p> <p>GROUP BY champRegroupement1 [, ...] -- regroupements sur les champs HAVING</p> <p>condition1 [AND/OR condition2 ...] -- restrictions après regroupements ORDER BY</p> <p>champTri ASC/DESC [, ...] -- tri croissant/décroissant sur les champs</p> <p><i>Commentaire : les clauses SQL (SELECT, FROM, etc.) doivent, si elles figurent dans la requête, figurer dans cet ordre. Seules les clauses SELECT et FROM sont obligatoires (en fait, seule la clause SELECT l'est réellement).</i></p>
Restrictions	<p>Les critères (≈conditions) de restriction la clause WHERE (resp. HAVING) permettent de limiter les lignes retourner. Ces critères permettent de ne retourner que les lignes vérifiant les conditions précisées dans le WHERE (resp. HAVING). Il s'agit de conditions de la forme : expression1 operateur expression2.</p> <p>Les opérateurs disponibles sont :</p> <ul style="list-style-type: none"> ○ les opérateurs de comparaison habituels : =, >, <, >=, <=, <> ; ○ les opérateurs propres au SQL : <ul style="list-style-type: none"> - expr BETWEEN valeur1 AND valeur2 : champ/expression comprise entre 2 valeurs - expr LIKE chaine : sorte d'égalité permettant d'utiliser les caractère « % » (n'importe quels caractères) et « _ » (n'importe quel caractère) pour préciser des conditions telles que : champ/expression commençant par..., contenant... ou terminant par... - expr IN (valeur1, valeur2[, ...]) : champ/expression faisant partie de l'une des valeurs listées.

Comme de coutumes, quelques exemples vaudront mieux que de longs discours :

Liste de tous les produits (tous les champs sont projetés)
SELECT * FROM Produit
Liste des produits (code et libellé) dont le prix est supérieur à 1000€
SELECT code, libelle FROM Produit WHERE Prix > 1000
Liste des produits dont le prix est compris entre 1000€ et 2000€
SELECT * FROM Produit WHERE Prix BETWEEN 1000 AND 2000
Commandes (numéro) passées le 13 mars 2016
SELECT num FROM Commande WHERE Emission = "2016-03-13"
Commandes passées le 13 ou 14 mars 2016
SELECT * FROM Commande WHERE Emission = "2016-03-13" OR Emission = "2016-03-14"
SELECT * FROM Commande WHERE Emission BETWEEN "2016-03-13" AND Emission = "2016-03-14"
SELECT * FROM Commande WHERE Emission IN ("2016-03-13", "2016-03-14")
Listes des produits par ordre alphabétique sur les libellés
SELECT * FROM Produit ORDER BY Libelle ASC
SELECT * FROM Produit ORDER BY Libelle -- le tri est croissant par défaut
Listes des commandes de l'année 2016
SELECT * FROM Commande WHERE YEAR(Emission) = 2016 ORDER BY Libelle ASC
SELECT * FROM Produit ORDER BY Libelle -- le tri est croissant par défaut

TP MYSQL et BDD révisions

3.7. Les jointures

Les jointures interviennent lorsque l'on souhaite récupérer des données en provenance de plusieurs tables. La jointure permet en quelque sorte de préciser au SGBD comment mettre en relation les données. S'agissant de « mettre en relation » les données, on comprendra qu'il s'agit de mettre en relation la clef étrangère d'une table avec la clef primaire d'une autre.

Exemple de jointure :

Liste de toutes les commandes avec le prénom et le nom du client.

```
SELECT Commande.*, Client.Prenom, Client.Nom
FROM Commande, Client
WHERE Commande.NumClient = Client.Num ;
```

Commentaire :

- Cette requête (avec jointure) comporte deux tables dans le *FROM* ;
- L'égalité *Commande.NumClient = Client.Num* permet d'effectuer la fameuse jointure ;
- *Commande.** permet de projeter tous les champs de la table *Commande* ;
- Les champs sont préfixés par le nom de la table suivi d'un point pour éviter les ambiguïtés et bien savoir de quelle table provient chaque champ.

Table « Client »

Num	Civilite	Prenom	Nom	Adresse	CodePostal	Ville	Email
1	M.	Jean-Claude	DUS	28 Rue de Bronzés	78000	Rambouillet	NULL
2	M.	Seigneur	ARAGORN	37 Avenue de la Terre du Milieu	75000	PANAME	aragorn@outlook.fr
3	M.	Agent	SMITH	1 Boulevard de la Matrice	45000	ORLEANS	neo@gmail.com
4	M.	Philippe	Pozo di Borgo	2 Rue des intouchables	78000	TRAPPES	NULL
5	M.	Daniel	Radcliffe	7 Rue des Avada Kedavra	45600	ST JEAN LE BLANC	wingardium@leviosa.com

Table « Commande »

Num	NumClient	Emission
1	1	2016-12-15
2	1	2016-12-30
3	1	2017-01-03
4	2	2017-01-07
5	3	2017-01-12
6	5	2017-01-17

Résultat de la requête exemple

Num	NumClient	Emission	Prenom	Nom
1	1	2016-12-15	Jean-Claude	DUS
2	1	2016-12-30	Jean-Claude	DUS
3	1	2017-01-03	Jean-Claude	DUS
4	2	2017-01-07	Seigneur	ARAGORN
5	3	2017-01-12	Agent	SMITH
6	5	2017-01-17	Daniel	Radcliffe

Jointure

Remarque ! Au regard de l'exemple ci-dessus, on retiendra bien que :

- Une jointure implique l'utilisation d'au moins deux tables ;
- Lorsqu'une requête fait intervenir deux tables ayant respectivement X et Y colonnes, on peut afficher jusqu'à X+Y colonnes. Bref, lorsque l'on a plusieurs tables (clause FROM), on peut projeter (clause SELECT) les champs des deux tables ;
- La jointure permet de mettre en relation les lignes des tables, c'est-à-dire de mettre en vis-à-vis les lignes qui vont ensemble.

TP MYSQL et BDD révisions

3.8. Les agrégats

Les agrégats permettent de projeter des données calculées et ainsi de répondre à des questions comme : quel est le nombre de ... ? Quelle est la moyenne de ... ? Etc. Les agrégats figurent dans la clauses SELECT :

COUNT(expression)	Permet de compter un nombre d'enregistrements
SUM(expression)	Permet de calculer la somme de valeurs numériques
MIN(expression)	Permet de trouver le minimum parmi des valeurs numériques
MAX(expression)	Permet de trouver le maximum parmi des valeurs numériques
AVG(expression)	Permet de calculer la moyenne de valeurs numériques

Exemples :

Afficher le nombre de produits ou quel est le nombre de produit ?
SELECT count(Num) FROM Produit ;
Afficher le prix du produit le plus cher ou quel est le prix du produit le plus cher ?
SELECT max(Prix) FROM Produit ;

Les deux exemples précédents ne retournent qu'une ligne. La clause GROUP BY permet de répondre à des questions plus complexes en regroupant des enregistrements ensemble. Les agrégats sont alors calculés par groupes d'enregistrements. La clause HAVING se comporte comme la clause WHERE à ceci près qu'elle permet d'effectuer une restriction en fonction des agrégats, à savoir une fois que ces derniers ont été calculés.

Exemples :

Quel est le nombre de commande par année ?
SELECT count(Num) FROM Commande GROUP BY YEAR(Emission) ;
Quel est le nombre de lignes par commande ?
SELECT count(Num) FROM Ligne GROUP BY NumCommande ;
Afficher le nombre de produits commandés de chaque commande.
SELECT sum(quantite) FROM Ligne GROUP BY NumCommande ;
Afficher les années où le nombre de commande a été supérieur à 1000.
SELECT Year(Emission) FROM Commande GROUP BY YEAR(Emission) HAVING count(Num) > 1000 ;

Remarque ! Il est absolument **interdit d'utiliser des agrégats dans la clause WHERE**.

3.9. Les sous-requêtes

Les requêtes peuvent elles-mêmes utiliser des requêtes, imbriquées dans la requête principale. Les requêtes imbriquées sont qualifiées de sous-requêtes. En particulier, les sous-requêtes sont particulièrement utilisées pour effectuer des restrictions avancées.

TP MYSQL et BDD révisions

Exemples :

Commandes passées par des clients de STRASBOURG	
Avec jointure : SELECT Commande.* FROM Commande, Client WHERE Commande.NumClient = Client.Num AND Client.Ville = "STRASBOURG" ;	Avec sous-requête : SELECT * FROM Commande WHERE Commande.NumClient IN (SELECT Num FROM Client WHERE Ville = "STRASBOURG");
Produits dont le prix est supérieur à la moyenne	
SELECT * FROM Produit WHERE Prix > (SELECT AVG(Prix) FROM Produit);	
Commande(s) dont la date est supérieure à celle de toutes les autres (=dernières commandes)	
SELECT * FROM Commande WHERE Emission > ALL (SELECT Emission FROM Commande);	

Remarque : il faut bien souvent essayer de décomposer un problème en sous-problèmes plus simples. En ce sens, une sous-requête répond à un sous-problème plus simple.

4. Comment restreindre les accès à une base de données ?

Très souvent, les bases de données sont partagées entre plusieurs utilisateurs. Pour des raisons de sécurité, il convient de restreindre les droits des utilisateurs de sorte qu'ils ne puissent effectuer que les manipulations dont ils ont besoin.

Dans le cadre de la conception d'un logiciel, l'on tâchera ainsi d'identifier les profils d'utilisateurs, c'est-à-dire de déterminer les groupes d'utilisateurs ayant les mêmes besoins (exemple : développeur, administrateur et utilisateur final). Ceci fait, l'on déterminera les droits d'accès de chacun des groupes.

A cet égard, le SQL permet de configurer les droits d'accès des utilisateurs d'une base de données de sorte qu'un utilisateur puisse seulement :

- Consulter des enregistrements (**SELECT**) ;
- Et/ou ajouter des enregistrements (**INSERT**) ;
- Et/ou mettre à jour des enregistrement (**UPDATE**) ;
- Et/ou supprimer des enregistrement (**DELETE**) ;
- Et/ou créer des tables/vues (**CREATE**) ;

TP MYSQL et BDD révisions

- Et/ou modifier des tables/vues (**ALTER**) ;
- Et/ou supprimer des tables/vues (**DROP**).

Ajout d'un droit d'accès	G R A
Révocation d'un droit d'accès	REVOKE droit1 [, ...] ON T_CHAMBRE FROM nomUtilisateur [, ...] ;

Exemples :

Autorise le profil « CLIENT » à consulter et à ajouter des commandes
GRANT SELECT, INSERT ON Commande TO CLIENT ;
Révoque l'autorisation accordée au profil « CLIENT » d'ajouter des commandes
REVOKE INSERT ON Commande TO CLIENT ;

Le langage SQL	Le SQL est le langage que vous utiliserez cette année pour créer (langage de définition de données) et manipuler (langage de manipulation des données) les données de bases de données.
-----------------------	---

Au travers de cet exercice, nous considérerons la base de données de « gestion des commandes » dont le schéma relationnel est le suivant :

Client(id_client, Civilite, Prenom, Nom, Adresse, CodePostal, Ville, Pays)

Clef primaire : id_client

Commande(id_commande, #id_Client, Emission)

Clef primaire : id_commande

Clef étrangère : id_client en référence au champ id_client de la relation Client

Autre : Emission correspond à la date de la commande

Produit(id_produit, Libelle, Prix)

Clef primaire : id_produit

Autre : le champ Prix est un nombre décimal à 6 chiffres, dont 2 après la « virgule ».

Ligne(id_ligne, #id_Commande, #id_Produit, Quantite)

Clef primaire : #id_Commande, #id_produit

Clefs étrangères :

-# id_Commande en référence au champ id_commande de la relation Commande ;

-# id_Produit en référence au champ id_produit de la relation Produit.

TP MYSQL et BDD révisions

Questions :

- 1.1.** Quelles sont les tables ?
- 1.2.** Quelle est la signification de la relation Ligne ?

Exercice 2 : création de tables

Les types de données	Nous l'avons vu, chaque champ d'une table a un type. Les différents types disponibles sur les SGBDR (dont MySQL, SGBD que nous utiliserons) sont entre autres les suivants :		
	Type	Type SQL	Description
	Alphanumérique	CHAR(n)	Chaîne de caractères de longueur fixes (n caractères)
		VARCHAR(n)	Chaîne de caractères de longueur variable (n car. max)
	Numérique	INTEGER	Entier de -2^{31} à $2^{31}-1$ (entier sur 32 bits)
		FLOAT	Décimal (parties entière et décimale séparée par un point)
		DECIMAL(n[,d])	Décimal à n chiffres dont d décimales (d est facultatif)
	Date/heure	DATE	Date sous la forme AAAA-MM-JJ
		TIME	Heure sous la forme hh:mm:ss.ss
TIMESTAMP		Date et heure	
Booléen	BOOLEAN	0 ou False pour FAUX, 1 ou True pour VRAI	
La création de tables	Créer une table consiste à préciser le nom de la table, à lister les champs de la table avec leur type, puis à préciser la clef primaire et les éventuelles clefs étrangères.		
	Par exemple, pour créer les tables Client et Commande, on utilisera les requêtes SQL suivantes :		
	CREATE TABLE Client(CREATE TABLE Commande(
	<u>Commentaire :</u>		
	<ul style="list-style-type: none">- les champs id_XXXX étant ici les clefs primaires de chacune des deux tables, ils sont NOT NULL par convention et il n'est pas utile de le préciser ;- auto_increment permet de préciser que l'entier est un compteur qui augmentera tout seul à chaque insertion de ligne (=enregistrement) dans la table ;- le champ id_XXX est et doit être du même type que la clef primaire à laquelle il fait référence.		

Questions :

- 2.1.** En vous inspirant du cours, écrire le script de création de tables (Uwamp, Wamp, SQLite, ...) ex :« `db_gest_commande_debut`».
- 2.2.** Que se passerait-il si la champ NumClient pouvait être NULL ?

Exercice 3 : modification de tables

TP MYSQL et BDD révisions

Une fois créée, la structure de la table peut encore être modifiée, voire la table peut être supprimée. Pour ce faire, on utilisera les requêtes SQL suivantes :

Suppression d'une table	DROP TABLE nomTable ;
Ajout d'un champ	ALTER TABLE nomTable ADD champAjouté type[, ...] ;
Ajout d'une clef primaire	ALTER TABLE nomTable ADD PRIMARY KEY (nomChamp1[, ...]) ;
Ajout d'une clef étrangère	ALTER TABLE nomTable ADD FOREIGN KEY (nomChamp1[, ...]) REFERENCES nomTablePointée(nomChampPointé1[, ...]) ;
Suppression d'un champ	ALTER TABLE nomTable DROP champSupprimé[, ...] ;
Suppression de la clef primaire	ALTER TABLE nomTable DROP PRIMARY KEY ;
Suppression d'une clef étrangère	ALTER TABLE nomTable DROP CONSTRAINT nomContrainte ; <i>Commentaire : lorsqu'une clef étrangère est créée, un nom de contrainte lui est attribué. Pour la supprimer cette clef étrangère, il faut utiliser ce nom.</i>

Questions :

- 3.1.** En vous aidant du tableau ci-dessus, rédiger la requête SQL permettant d'ajouter un champ Email à la table Client. Mettre à jour (en rouge) le schéma relationnel.
- 3.2.** On souhaite à présent pouvoir ajouter une description complète de chaque produit. Proposer une évolution de la base de données et rédiger la requête SQL permettant cette évolution. Enfin, mettre à jour (en rouge) le schéma relationnel.
- 3.3.** Après création de la base de données, on s'aperçoit que la mise en place de la clef étrangère de la table Commande a été omise. Rédiger la requête SQL permettant de corriger cette erreur.
- 3.4.** La société utilisant cette base de données n'a que des clients français. Dès lors, il apparaît que le champ Pays de la table Client est inutile. Ecrire le requête SQL permettant de le supprimer. Mettre à jour le schéma relationnel (en rouge toujours).

TP MYSQL et BDD révisions

Exercice 4 : insertion, modification et suppression d'enregistrements

On rappelle qu'on appelle « enregistrement » les lignes d'une table. A cet égard, le langage SQL permet d'insérer de nouveaux enregistrements, de modifier des enregistrements existant mais encore de supprimer des enregistrements. Le format des requêtes SQL est le suivant :

Insertion	INSERT INTO nomTable[(champ1, champ2, ...)] VALUES (valeur1, valeur2, ...), -- 1 ^{ère} ligne ajoutée (valeur1, valeur2, ...), -- 2 ^{ème} ligne ajoutée ... (valeur1, valeur2, ...); -- dernière ligne ajoutée
Modification	UPDATE nomTable SET champModifié = nouvelleValeur [, ...] [WHERE condition1 AND/OR condition2 ...];
Suppression	DELETE FROM nomTable [WHERE condition1 AND/OR condition2 ...];

Bien entendu, rien ne vaut quelques exemples :

Ajout d'un nouveau client
INSERT INTO Client(Civilite, Nom, Prenom, Adresse, CodePostal, Ville, Email) VALUES ("M.", "Washington", "Denzel", "Av. de Wagram", "75000", "PARIS", "dw@gmail.com"); Autre possibilité : INSERT INTO Client VALUES (NULL, "M.", "Denzel", "Washington", "Av. de Wagram", "75000", "PARIS", "dw@gmail.com"); <i>Commentaire : si on décide de ne pas préciser la liste des champs, les valeurs doivent être précisée dans l'ordre des champs de la table.</i>
Mise à jour du prix du produit n°1
UPDATE Produit SET Prix = 75.32 WHERE Num = 1
Augmentation de 10% du prix des produits
UPDATE Produit SET Prix = Prix * 1.1
Suppression du produit n°5
DELETE FROM Produit WHERE Num = 5
Suppression des produits dont le prix excède 1000€
DELETE FROM Produit WHERE Prix > 1000
Suppression des clients dont le prénom commence par la letter B
DELETE FROM Produit WHERE Prenom LIKE "B%" <i>Commentaire : le symbole « % » dans la chaîne « B% » permet d'indiquer que le prénom peut commencer par la lettre « B » suivie de n'importe quels caractères. L'opérateur de comparaison « LIKE » ressemble à un « = » mais permet de plus d'effectuer ce genre de comparaisons approximatives.</i>

TP MYSQL et BDD révisions

Questions :

- 4.1. Rédiger la requête qui permet de vous ajouter en tant que client.
- 4.2. Rédiger la requête qui permet d'ajouter deux ordinateurs, un « Ordinateur Asus trop bien » à 1150€ ainsi qu'un « Ordinateur AlienWare un peu cher » à 2500€.
- 4.3. Rédiger la requête permettant de supprimer les clients des Yvelines (78).
- 4.4. Rédiger la requête permettant de modifier le client Grégoire LEMARCHE dont le nom a été mal saisi (LEMACHIN a été saisi au lieu de LEMARCHE).
- 4.5. Rédiger la requête permettant de supprimer les commandes antérieures aux 31/12/2016. Pourquoi cette requête échouera ?

Exercice 5 : récupérer/projeter des données

Le cœur du SQL, pourrait-on dire, réside dans les requêtes que nous allons étudier dans cette cinquième partie : les requêtes SELECT. On parle de requêtes de projection, et souvent, à tort, de sélections. Ces requêtes SQL permettent en quelque sorte d'afficher/retourner des données. Elles ont la syntaxe suivante :

Projection	SELECT champProjeté1 [, ...] FROM table1 [, ...] WHERE condition1 [AND/OR condition2 ...] GROUP BY champRegroupement1 [, ...] HAVING condition1 [AND/OR condition2 ...] ORDER BY champTri ASC/DESC [, ...] <u>Commentaire :</u> - les clauses SQL (SELECT , FROM , WHERE , etc.) doivent, si elles figurent dans la requête, figurer dans cet ordre ; - seules les clauses SELECT et FROM sont obligatoires (en fait, seule la clause SELECT l'est réellement).	-- projection de champs -- tables utilisées -- restrictions -- regroupements sur les champs -- restrictions après regroupements -- tri croissant/décroissant sur les champs
Restrictions	Les critères (≈conditions) de restriction la clause WHERE (resp. HAVING) permettent de limiter les lignes retourner. Ces critères permettent de ne retourner que les lignes vérifiant les conditions précisées dans le WHERE (resp. HAVING). Il s'agit de conditions de la forme : expression1 operateur expression2. Les opérateurs disponibles sont : ○ les opérateurs de comparaison habituels : =, >, <, >=, <=, <> ; ○ les opérateurs propres au SQL : - expr BETWEEN valeur1 AND valeur2 : champ/expression comprise entre 2 valeurs - expr LIKE chaîne : sorte d'égalité permettant d'utiliser les caractère « % » (n'importe quels caractères) et « _ » (n'importe quel caractère) pour préciser des conditions telles que : champ/expression commençant par..., contenant... ou terminant par... - expr IN (valeur1, valeur2[, ...]) : champ/expression faisant partie de l'une des valeurs listées.	

Comme de coutumes, quelques exemples vaudront mieux que de longs discours :

TP MYSQL et BDD révisions

Liste de tous les produits (tous les champs sont projetés)

SELECT * FROM Produit

Liste des produits (code et libellé) dont le prix est supérieur à 1000€

SELECT code, libelle **FROM** Produit **WHERE** Prix > 1000

Liste des produits dont le prix est compris entre 1000€ et 2000€

SELECT * FROM Produit **WHERE** Prix **BETWEEN** 1000 **AND** 2000

Commandes (numéro) passées le 13 mars 2016

SELECT num **FROM** Commande **WHERE** Emission = "2016-03-13"

Commandes passées le 13 ou 14 mars 2016

SELECT * FROM Commande **WHERE** Emission = "2016-03-13" **OR** Emission = "2016-03-14"

SELECT * FROM Commande **WHERE** Emission **BETWEEN** "2016-03-13" **AND** Emission = "2016-03-14"

SELECT * FROM Commande **WHERE** Emission **IN** ("2016-03-13", "2016-03-14")

Listes des produits par ordre alphabétique sur les libellés

SELECT * FROM Produit **ORDER BY** Libelle ASC

SELECT * FROM Produit **ORDER BY** Libelle -- le tri est croissant par défaut

Listes des commandes de l'année 2016

SELECT * FROM Commande **WHERE** YEAR(Emission) = 2016 **ORDER BY** Libelle ASC

SELECT * FROM Produit **ORDER BY** Libelle -- le tri est croissant par défaut

Questions :

- 5.1. Récupérer le détail de la commande 58.
- 5.2. Lister les produits par ordre décroissant de prix.
- 5.3. Lister les commandes de l'année 2015 par date décroissante.
- 5.4. Lister les commandes du mois de janvier 2016.
- 5.5. Lister les commandes ordonnées par années croissantes.

TP MYSQL et BDD révisions

Exercice 6 : les jointures

Les jointures interviennent lorsque l'on souhaite récupérer des données en provenance de plusieurs tables. La jointure permet en quelque sorte de préciser au SGBD comment mettre en relation les données. S'agissant de « mettre en relation » les données, on comprendra qu'il s'agit de mettre en relation la clef étrangère d'une table avec la clef primaire d'une autre.

Questions :

- 6.1. Récupérer le détail de la commande 58 avec l'adresse e-mail du client qui a passé la commande.
- 6.2. Récupérer le détail de la commande 35 avec ses lignes de commandes.
- 6.3. Récupérer la ligne de commande 2 de la commande 5 avec le libellé et le prix du produit commandé.
- 6.4. Récupérer le détail de la commande 47 avec ses lignes de commandes, et pour chaque ligne de commande le libellé et le prix du produit commandé.
- 6.5. Compléter la requête 6.4. afin d'afficher (calcul) le prix de chaque lignes de la commande 47.

Exercice 7 : les agrégats

Les agrégats permettent de projeter des données calculées et ainsi de répondre à des questions comme : quel est le nombre de ... ? Quelle est la moyenne de ... ? Etc. Les agrégats figurent dans la clauses SELECT :

COUNT(expression)	Permet de compter un nombre d'enregistrements
SUM(expression)	Permet de calculer la somme de valeurs numériques
MIN(expression)	Permet de trouver le minimum parmi des valeurs numériques
MAX(expression)	Permet de trouver le maximum parmi des valeurs numériques
AVG(expression)	Permet de calculer la moyenne de valeurs numériques

TP MYSQL et BDD révisions

Exemples :

Afficher le nombre de produits ou quel est le nombre de produit ?
SELECT count(Num) FROM Produit ;
Afficher le prix du produit le plus cher ou quel est le prix du produit le plus cher ?
SELECT max(Prix) FROM Produit ;

Les deux exemples précédents ne retournent qu'une ligne. La clause GROUP BY permet de répondre à des questions plus complexes en regroupant des enregistrements ensemble. Les agrégats sont alors calculés par groupes d'enregistrements. La clause HAVING se comporte comme la clause WHERE à ceci près qu'elle permet d'effectuer une restriction en fonction des agrégats, à savoir une fois que ces derniers ont été calculés.

Exemples :

Quel est le nombre de commande par année ?
SELECT count(Num) FROM Commande GROUP BY YEAR(Emission) ;
Quel est le nombre de lignes par commande ?
SELECT count(Num) FROM Ligne GROUP BY NumCommande ;
Afficher le nombre de produits commandés de chaque commande.
SELECT sum(quantite) FROM Ligne GROUP BY NumCommande ;
Afficher les années où le nombre de commande a été supérieur à 1000.
SELECT Year(Emission) FROM Commande GROUP BY YEAR(Emission) HAVING count(Num) > 1000 ;

Questions :

- 7.1. Quel est le nombre de clients ?
- 7.2. Afficher le nombre de clients par ville trié par nom de ville croissant.
- 7.3. Quel est le nombre de commandes par mois et par année ?
- 7.4. Quel est le nombre de commandes par ville ?
- 7.5. Quel est le prix moyen d'un produit ?
- 7.6. Quel est le prix total de la commande 58 ?
- 7.7. Quel est le nombre de commandes comptabilisant moins de 5 lignes ?
- 7.8. Quel est le chiffre d'affaires réalisé sur l'année 2016 ?
- 7.9. Quel est le chiffre d'affaires réalisé par client ?

Exercice 8 : les sous-requêtes

Les requêtes peuvent elles-mêmes utiliser des requêtes, imbriquées dans la requête principale. Les requêtes imbriquées sont qualifiées de sous-requêtes. En particulier, les sous-requêtes sont particulièrement utilisées pour effectuer des restrictions avancées.

TP MYSQL et BDD révisions

Exemples :

Commandes passées par des clients de STRASBOURG	
Avec jointure : SELECT Commande.* FROM Commande, Client WHERE Commande.NumClient = Client.Num AND Client.Ville = "STRASBOURG" ;	Avec sous-requête : SELECT * FROM Commande WHERE Commande.NumClient IN (SELECT Num FROM Client WHERE Ville = "STRASBOURG");
Produits dont le prix est supérieur à la moyenne	
SELECT * FROM Produit WHERE Prix > (SELECT AVG(Prix) FROM Produit);	
Commande(s) dont la date est supérieure à celle de toutes les autres (=dernières commandes)	
SELECT * FROM Commande WHERE Emission > ALL (SELECT Emission FROM Commande);	

Remarque : il faut bien souvent essayer de décomposer un problème en sous-problèmes plus simples. En ce sens, une sous-requête répond à un sous-problème plus simple.

Questions :

- 8.1. Quel est le produit le moins cher ? Quel est le plus cher ?
- 8.2. Quel est ou quels sont les clients à avoir passé le plus de commandes ?

TP MYSQL et BDD révisions

Exercice 9 : les droits sur les bases de données

Très souvent, les bases de données sont partagées entre plusieurs utilisateurs. Pour des raisons de sécurité, il convient de restreindre les droits des utilisateurs de sorte qu'ils ne puissent effectuer que les manipulations dont ils ont besoin. Les utilisateurs sont typiquement regroupés par profils d'utilisateurs ayant les mêmes besoins (exemple : développeur, administrateur et utilisateur final). Finalement, le SQL permet de configurer les droits d'accès des utilisateurs d'une base de données de sorte qu'un utilisateur puisse seulement :

- consulter des enregistrements (**SELECT**) ;
- et/ou ajouter des enregistrements (**INSERT**) ;
- et/ou mettre à jour des enregistrement (**UPDATE**) ;
- et/ou supprimer des enregistrement (**DELETE**) ;
- et/ou créer des tables/vues (**CREATE**) ;
- et/ou modifier des tables/vues (**ALTER**) ;
- et/ou supprimer des tables/vues (**DROP**).

Ajout d'un droit d'accès	GRANT droit1 [, ...] ON nomTable TO nomUtilisateur [, ...] ;
Révocation d'un droit d'accès	REVOKE droit1 [, ...] ON T_CHAMBRE FROM nomUtilisateur [, ...] ;

Exemples :

Autorise le profil « CLIENT » à consulter et à ajouter des commandes
GRANT SELECT, INSERT ON Commande TO CLIENT ;
Révoque l'autorisation accordée au profil « CLIENT » d'ajouter des commandes
REVOKE INSERT ON Commande TO CLIENT ;

Questions :

La base de données est interconnectée à un logiciel (un portail extranet) au moyen duquel les clients passent leurs commandes et peuvent consulter les commandes qu'ils ont passées. C'est la secrétaire qui enregistre les clients. Pour ce faire, elle consulte la liste des clients puis ajoute un nouveau client. Sur demande expresse, il arrive à celle-ci de supprimer les clients. Ce sont en revanche les commerciaux qui gèrent le catalogue de produits. Finalement, seule le responsable commercial peut modifier ou supprimer une commande passée.

9.1. Identifier les profils d'utilisateurs.

9.2. Pour chaque profil d'utilisateurs, déterminer les droits d'accès dont il a besoin.

9.3. Rédiger les requêtes SQL permettant d'octroyer les droits d'accès nécessaires à un profil d'utilisateur (au choix).