📝 Bases de données / SQL

Introduction aux bases de données

Les bases de données sont des magasins qui permettent de stocker des informations et permettent notamment d'effectuer différentes opérations de lecture et d'écriture.

Elles sont très efficaces pour retrouver des informations qui y sont stockées. Vous pouvez interroger une base de données, par exemple : "Combien d'utilisateurs sont inscrits sur mon site web ?" ou "Quel article dans mon blog contient tel ou tel mot-clé ?"

La force des bases de données réside également dans leurs possibilités de lier les différentes informations qui y sont préalablement stockées.

Présentation du langage SQL

Le SQL (acronyme de Structured Query Language) est un langage informatique permettant d'interagir avec des bases de données relationnelles. Ce langage est normalisé depuis 1986.

Exemples de requêtes :

- Lecture / Recherche

```
SELECT * FROM ma_table WHERE name = "jean";
```

- Mise à jour

```
UPDATE ma_table SET champ1 = "updated value" WHERE id = 5;
```

- Suppression

```
DELETE FROM ma_table WHERE id = 12;
```

Les serveurs web locaux

Contneu

MySQL et phpMyAdmin

MySQL

MySQL est un SGBDR (Système de Gestion de Bases de Données Relationnelles). C'est l'un des systèmes les plus populaires au monde et est très utilisé dans le monde du web.

Comme son nom l'indique, MySQL utilise le langage SQL pour effectuer des requêtes.

phpMyAdmin

phpMyAdmin (souvent abrégé PMA) est une application web de gestion pour MySQL. Il est écrit en PHP et permet de gérer les bases de données dans une interface visuelle simple et intuitive.

Il est important de noter que phpMyAdmin n'est qu'une application. Il est tout à fait possible de gérer MySQL avec d'autres applications similaires ou en ligne de commande...

Modélisation et conception d'une base de données

Les tables

Au sein d'une base de données, une table est un ensemble de données organisées sous forme de tableau. Par exemple, une table pourrait être comparée à une feuille dans un fichier excel.

On trouve ainsi dans une table :

- plusieurs colonnes : les champs
- plusieurs lignes : les données

Il est possible de créer plusieurs tables SQL dans une même base de données. Ainsi, nous pourrons avoir une table users pour les membres, articles pour les billets d'un blog, etc.

Chaque table doit être nommée rigoureusement afin d'éviter les incohérences et d'avoir une maintenabilité aisée. De plus, il ne ne peut y avoir deux tables ayant le même nom dans une même base de données, puisque les requêtes SQL s'effectuent sur une table de données.

Les types de données

Dans une base de données, on trouve plusieurs tables. Celles-ci se composent des colonnes servant à contenir chacune un type de données précis.

Par exemple, une table users contiendra probablement les colonnes :

- id (clé primaire pour l'auto-incrémentation)
- username (pseudo)
- email

- birthdate (date de naissance)
- gender (sexe, homme ou femme)

En fonction de l'information stockée, le type de la colonne ne sera pas la même, pour des raisons de clarté, de facilité, mais également d'optimisation.

Les types de données les plus courants sont :

- VARCHAR : permet de stocker un texte ayant pour longueur maximale 255 caractères.
- TEXT : permet de stocker un texte ayant pour longueur maximale 65 535 caractères
- INT : permet de stocker un nombre positif ou négatif sans décimales (maxi 4 294 967 295)
- FLOAT : permet de stocker un nombre positif ou négatif avec décimales
- DATE : permet de stocker une date au format YYYY-MM-DD
- DATETIME : permet de stocker une date et une heure au format YYYY-MM-DD HH:MM:SS
- ENUM : permet de stocker une liste de choix, très ressemblant à un array en PHP

Il existe toutefois de nombreux autres types de données. Par exemple pour les valeurs numériques : TINYINT, SMALLINT, MEDIUMINT et BIGINT qui vont permettre d'augmenter le nombre maximum.

Pour des raisons d'optimisation et de performance, il est nécessaire de choisir les bons types de données et de limiter leurs tailles possibles.

Administration d'une base de données

Les utilisateurs

MySQL dispose d'un utilisateur par défaut, root. Cet utilisateur est un super administrateur qui peut effectuer toutes les opérations qu'il souhaite. Lors de la mise en ligne d'un site web, il est dangereux d'accéder à une base de données via l'utilisateur root.

Ainsi, il est possible de créer plusieurs utilisateurs avec différents privilèges pour chacun. Le privilège, par exemple, de stocker, de lire ou de modifier des données. Ces privilèges peuvent exister sur les bases de données ou sur les tables.

Chaque utilisateur est ensuite authentifié à l'aide d'un mot de passe.

Les différents SGBD

Un Système de Gestion de Bases de Données (abrégé SGBD) est un logiciel permettant d'introduire des données, de mettre à jour et d'accéder aux données stockées dans une base.

Le modèle relationnel consiste à stocker l'information dans des tables très précisément définies par leur schéma (leurs différentes colonnes, clés primaires, clés étrangères). Cela permet de ne pas stocker l'information plusieurs fois, et de pouvoir facilement consolider les données avec des requêtes SQL et des jointures.

On distingue deux principaux SGBD, à savoir :

- Les SGBDR : Système de Gestion de Base de Données Relationnelles. Par exemple, MySQL en est un.
- Les NoSQL : Not Only SQL. Cela ressemble fortement à un grand tableau PHP. Il n'y plus de colonnes, on parle alors de bases de données clé-valeur

On trouvera dans les SGBDR des logiciels tels que :

- MySQL / MariaDB
- PostgreSQL
- Microsoft SQL Server
- Oracle Database

Les SGBD NoSQL les plus courants sont :

- MongoDB
- Redis
- CouchDB
- Cassandra

Clés primaires

Dans une base de données, une clé primaire permet d'assurer l'unicité de chaque ligne de la table. Chaque table de votre base de données doit contenir une colonne de clés primaires. C'est donc un identifiant unique. La clé primaire, souvent de type numérique et nommée id, sera auto-incrémentée.

Ces clés primaires permettent une sélection précise pour de la lecture (SELECT), de la modification (UPDATE) ou suppression (DELETE) de données au sein d'une table.

Importer et exporter des données

L'import et l'export de données d'une base est une chose importante. Par exemple, pour pouvoir faire des sauvegardes, mais également afin de pouvoir travailler en équipe et donc de bénéficier de la même structure d'une base de données. Sous MySQL, il existe plusieurs manières pour importer/exporter des données

phpMyAdmin offre une interface intuitive et pratique. Il est donc possible d'exporter des données en cliquant sur le bouton du même nom.

Plusieurs options sont alors disponibles comme :

- Exporter uniquement la structure de la base
- Exporter la structure et les données

De plus, une case peut être cochée afin de créer la base de données directement lors d'un import du fichier.

L'import de données se fera à partir d'un fichier préalablement exporté. Le fichier peut être éventuellement compressé pour les bases importantes. Toutefois, il faudra vérifier dans la configuration de PHP la taille maximale d'upload de fichier.

MySQL offre la possibilité d'importer/exporter des données en ligne de commandes. Bien que moins intuitif, cela peut s'avérer très utile si la base de données est de taille importante.

Pour exporter une base :

```
mysql -h host -u user -ppass nom_de_la_base > fichier_dump.sql
```

Le fichier SQL fichier_dump.sql contiendra alors l'export de la base de données.

Pour importer une base :

```
mysql -h host -u user -ppass nom_de_la_base < fichier_dump.sql
```

On notera le sens différent du chevron entre l'import < et l'export >.

Les relations

MySQL est un Système de Gestion de Bases de Données Relationnelles (abrégé SGBDR). Il est donc capable de gérer des relations entre différentes tables.

Il est nécessaire que les deux tables aient un champ en commun, en principe, c'est le champ qui permet d'identifier un élément de la table principale de manière unique (la clé primaire).

Par exemple, on pourra relier une table articles contenant les articles d'un blog et une liste d'utilisateurs users. Ainsi, dans la table articles on trouvera une colonne id_user permettant d'effectuer la relation avec la table users.

CRUD

L'acronyme informatique anglais CRUD (pour Create, Read, Update, Delete) désigne les quatre opérations de base pour le stockage et la manipulation d'informations dans une base de données.

- Create : Insérer des nouveaux enregistrements (écriture)
- Read : Récupérer des enregistrements (lecture)
- Update : Mettre à jour des enregistrements (écriture)
- Delete : Supprimer des enregistrements (écriture)

Les opérateurs correspondent chacun à un type de requête SQL :

Create

```
INSERT INTO user (firstname, lastname, email) VALUES ('John', 'Doe', 'john.doe@mail.com');
```

-- Syntaxe alternative

```
INSERT INTO user SET firstname = 'John', lastname = 'Doe', email = 'john.doe@mail.com';
```

La syntaxe alternative ne permet pas d'insérer plusieurs séries de valeurs en une seule requête, contrairement à la syntaxe de base.

Exemple :

```
INSERT INTO user (firstname, lastname, email) VALUES

('John', 'Doe', 'john.doe@mail.com'),

('Bob', 'Arctor', 'bob.arctor@mail.com');
```

Read

```
SELECT firstname, lastname FROM user;
```

Update

```
UPDATE user SET firstname = 'John', lastname = 'Doe' WHERE id = 123;
```

Delete

```
DELETE FROM user WHERE email = '';
```

Clauses SQL de base

SELECT / FROM

Permet de lire de l'information depuis une table.

La clause SELECT précise la liste des colonnes à remonter ou * pour remonter toutes les colonnes de la table.

La clause FROM précise la table sur laquelle doit être effectuée la sélection.

On peut utiliser des fonctions SQL dans la clause SELECT (c.f. fonctions SQL).

```
SELECT colonne1, colonne2 FROM ma_table;
```

```
SELECT * FROM ma_table;
```

```
SELECT COUNT(id) FROM ma_table;
```

Les colonnes peuvent être remontées avec des noms personnalisés Exemple :

```
SELECT colonne1, colonne2 FROM ma_table;
```

```
SELECT * FROM ma_table;
```

WHERE

Réservé aux requêtes SELECT, UPDATE et DELETE, permet de filtrer la sélection ou les lignes à mettre à jour ou à supprimer.

`SELECT firstname, lastname FROM user WHERE age >= 18;` -- Remonte le prénom et le nom de tous les utilisateurs de 18 ans ou plus

`UPDATE contact SET newsletter = 0 WHERE email = '';` -- Mets à jour le champ newsletter de tous les contacts avec un email vide

`DELETE FROM post WHERE archive = 1;` -- Supprime tous les posts dont la colonne archive vaut 1

On peut combiner plusieurs clauses WHERE en les séparant par l'opérateur AND et/ou OR. La gestion des combinatoires et des priorités est complétée avec des parenthèses.

`SELECT firstname, lastname, email FROM user WHERE email != '' AND newsletter = 1 AND (country = 'FR' OR country = 'BE' OR country = 'CH') ;` -- Remonte tous les utilisateurs dont l'email n'est pas vide, qui ont accepté de recevoir la newsletter et qui résident en France OU en Belgique OU en Suisse

ORDER BY

Réservé aux requêtes SELECT, permet de trier la liste des résultats de la requête, en ordre croissant (ASC) ou décroissant (DESC).

Par défaut si on ne précise pas l'ordre de tri, c'est un tri croissant qui est effectué. On peut combiner plusieurs clauses de tri en les séparant avec des virgules.

`SELECT * FROM movies ORDER BY release_date DESC;` // Remonte tous les films triés par date de sortie décroissante

`SELECT * FROM movies ORDER BY RAND();` // Remonte tous les films dans un ordre pseudo aléatoire

`SELECT * FROM movies ORDER BY lastname ASC, firstname ASC; //` Remonte tous les films triés par ordre alphabétique sur le nom de famille, et à nom égal par ordre alphabétique croissant sur le prénom

LIMIT

Réservé aux requêtes SELECT, permet de restreindre la sélection en précisant un nombre de lignes, et éventuellement un point de départ, permettant entre autres de gérer une pagination.

`SELECT * FROM movies ORDER BY RAND() LIMIT 3;` // Remonte 3 films aléatoires

`SELECT * FROM movies LIMIT 50, 10;` // Remonte 50 films à partir de la ligne 50

INSERT INTO

Permet d'insérer une ligne dans une table en précisant les colonnes à remplir avec leurs valeurs respectives.

`INSERT INTO user (firstname, lastname, email) VALUES ('John', 'Doe', 'john.doe@mail.com');` // Insère une ligne dans la table user

`INSERT INTO user SET firstname = 'John', lastname = 'Doe', email = 'john.doe@mail.com';` // Syntaxe alternative, même résultat que la requête précédente

UPDATE

Permet de modifier une ou plusieurs ligne(s) existante(s) d'une table.

`UPDATE contact SET newsletter = 1, update_date = NOW() WHERE id = 42;` // Modifie la colonne newsletter et update_date de l'utilisateur 42

`UPDATE movies SET rank = rank + 1 WHERE id = 123;` // Modifie le champ rank en l'incrémentant de 1 pour l'identifiant 123

`UPDATE user SET newsletter = 0 WHERE email = '';` // Modifie la colonne newsletter pour tous les user avec une colonne email vide

ATTENTION : Une requête UPDATE sans clause WHERE affectera toutes les lignes de la table

DELETE

Permet de supprimer une ou plusieurs ligne(s) depuis une table.

`DELETE FROM user WHERE id = 42;` // Supprime la ligne de la table user dont l'identifiant est 42

`DELETE FROM mail WHERE deleted = 1;` // Supprime tous les lignes de la table mail dont la colonne deleted vaut 1

ATTENTION : Une requête DELETE sans clause WHERE affectera toutes les lignes de la table

Fonctions SQL de base

COUNT

Permet de compter le nombre de lignes remontées par une requête.

`SELECT COUNT(id) FROM user;` // Remonte le nombre de lignes de la table user

MIN/MAX

Permet de retourner la plus petite/grande valeur d'une colonne.

`SELECT MIN(year) as min_year, MAX(year) as max_year FROM movies;` // Renvoie l'année du film le plus ancien et l'année du film le plus récent

AVG

Calcule et renvoit la moyenne de la colonne scores pour toutes les lignes

```
SELECT AVG(scores) FROM nom_de_la_table;
```

SUM

Permet de faire la somme de tous les valeurs d'une colonne.

`SELECT SUM(scores) FROM games;` -- Calcule et renvoit la somme des valeurs de la colonne scores

`SELECT SUM(price) FROM products;` -- Même chose avec des prix de produits

NOW

Renvoie la date et l'heure courante

`UDPATE user SET last_connexion = NOW();` // Modifie la date de dernière connexion au format "Y-m-d H:i:s"

Les erreurs SQL

Différents types d'erreurs peuvent survenir lors de la connexion à la base de données MySQL, et durant l'exécution des requêtes SQL.

Les erreurs les plus fréquentes à la connexion

- [2002] php_network_getaddresses: getaddrinfo failed: Hôte inconnu
- [2002] Une tentative de connexion a échoué car le parti connecté n'a pas répondu convenablement au-delà d'une certaine durée ou une connexion établie a échoué car l'hôte de connexion n'a pas répondu.

Le paramètre host du DSN semble incorrect ou le serveur ne répond pas. Le paramètre host doit respecter les formats suivants : 127.0.0.1, localhost, 12.34.56.78, http://www.domain.com.

- [1045] Access denied for user 'root'@'localhost' (using password: YES)

L'utilisateur root n'est pas autorisé à se connecter à la base de données, ou le mot de passe est incorrect.

- [1049] Unknown database 'db_name'

Le nom de la base de données est incorrect ou l'utilisateur connecté n'a pas les droits de lecture sur cette base de données.

Les erreurs SQL les plus fréquentes

- [42000] Syntax error or access violation [1064] You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'SELECT FROM'...

La requête comporte une erreur de syntaxe, il faut vérifier les points suivants :

    - Le nom et l'ordre des instructions sont corrects
    - Il ne manque aucune virgule, parenthèse ou opérateur (=, !=, ...)
    - Aucun mot clé réservé à SQL n'est utilisé comme nom de table ou de colonne (select, from, match, against...etc) sans être échappé avec des `

- [42S02] Base table or view not found [1146] Table 'db_name.table2' doesn't exist

La table sur laquelle porte la requête est incorrect ou n'existe pas, il faut vérifier sa présence et son nom exact dans la base de données

- [42S22] Column not found: 1054 Unknown column 'tittle' in 'field list'

Le nom d'une colonne utilisée dans la requête est incorrect ou n'existe pas, il faut vérifier sa présence et son nom exact dans la base de données