BOOTSTRAP MYSQL

Ou comment réussir ses examens MySQL et my_cinema en un bootstrap?

Introduction

- **MySQL** est un langage de gestion de bases de données **SQL**
- **Phpmyadmin** est une interface graphique de gestion de bases de données SQL intégrant MySQL
- Une base de donnée est un ensemble de données formatées en **bases**, **tables**, **champs** et **enregistrements**
- Par exemple excel est une base de donnée formatant les données en fichiers, onglets, colonnes, lignes et possède son langage propre et une interface graphique pour ces données

Les actions de bases

- En tant que développeur il vous faudra apprendre à :
  - Insérer des enregistrements en base de donnée
  - Sélectionner des enregistrements en base de donnée
  - Mettre à jour des enregistrements en base de donnée
  - Supprimer des enregistrements en base de donnée

- Dans un produit web, ces quatre actions sont souvent associés à une donnée spécifique, c'est ce que l'on appelle un **CRUD** (create, read, update, delete)
- Par exemple, on peut parler du CRUD user

INSERT

- INSERT INTO film(titre, resum) VALUES('Moi la W@C', 'Indisponible')
- Ici, on dit à SQL d' « insérer dans la table film, et respectivement dans les champs titre et resum, les valeurs 'Moi la W@C' et 'Indisponible' »

SELECT

- SELECT titre, resum FROM film WHERE titre LIKE « %In the% »
- Ici, on dit à SQL de « sélectionner le titre et le résumé de la table film où le titre ressemble à 'In the' »

UPDATE

- UPDATE film SET titre = ‘La Coding’ WHERE titre = ‘Moi la W@C’
- Ici, on dit à SQL de « mettre à jour la table film de façon à ce que le titre deviennent ‘coding’ où les enregistrements ont pour titre ‘Moi la W@C’ »

DELETE

- DELETE FROM film WHERE titre = 'Moi la W@C'
- Ici, on dit à SQL de « supprimer les enregistrements de la table film où le titre est 'Moi la W@C' »

WHERE 1

- **WHERE** est suivi d'une **condition**
- Cette commande permet de sélectionner les enregistrements d'une BDD qui respectent cette condition
- Ex : SELECT * FROM film WHERE titre LIKE '%In the%'
  - Sélectionne tout les champs dans la table film où le titre ressemble à 'In the'
  - Ici le titre ressemble à 'In the' est la condition de la requête définie par la commande WHERE

WHERE 2

- La commande WHERE à le même comportement que le IF de PHP, vous pouvez donc stacker les conditions grâce aux commander **AND** et **OR** :
  - SELECT * FROM film WHERE titre LIKE '%In the%' OR (id => 100 AND id < 200)
    - Sélectionne tout les champs dans la table film où le titre ressemble à 'In the' ou dont l'id est compris entre 100 et 199

Les options d'affichage

- **ORDER BY** permet de trier les enregistrements d'une requête SQL et prends une option ASC ou DESC
  - SELECT * FROM film ORDER BY titre ASC
- **AS** permet de renommer un enregistrement dans les résultats d'une requête
  - SELECT date_début_affiche AS 'date de sortie' FROM film
- **LIMIT** limite le nombre d'enregistrements maximums retournés par une requête SQL
  - SELECT * FROM film ORDER BY titre ASC LIMIT 10

Les fonctions SQL

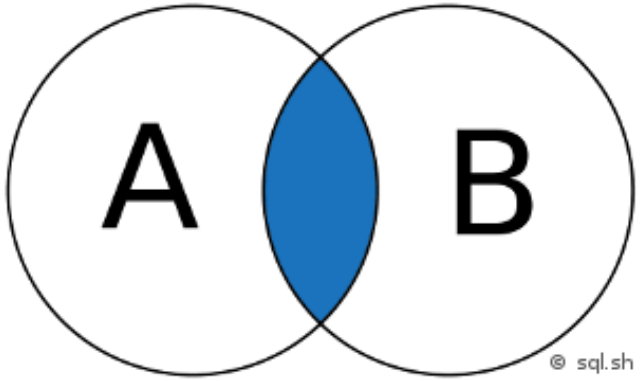
- Les fonctions SQL ont le même comportement que les fonctions PHP
- Quelques exemples :
  - **AVG**() est une fonction d'agrégation qui permet de faire la moyenne d'un champ numérique d'un ensemble d'enregistrements
    - SELECT AVG(nbr_siege) FROM salle
  - **UPPER**() est une fonction de chaîne de caractère qui permet de mettre un champ en majuscule
    - SELECT UPPER(titre) FROM film
  - **NOW**() est une fonction de dates et d'heures qui permet d'obtenir l'heure courante, elle est très pratique pour insérer l'heure d'un enregistrement
    - INSERT INTO abonnement(nom, date_début) VALUES('50%', NOW())

Les jointures

- Les jointures SQL permettent **d'associer plusieurs tables** dans une même requête
- En général, elles permettent d'associer les enregistrements de deux tables différentes en associant l'égalité des valeurs d'un champ d'une première table par rapport à la valeur des champs d'une seconde table
- Il est possible lors de la conception de BDD de mettre en place des **relations** entre les tables qui nécessitent l'utilisation de jointures

INNER JOIN

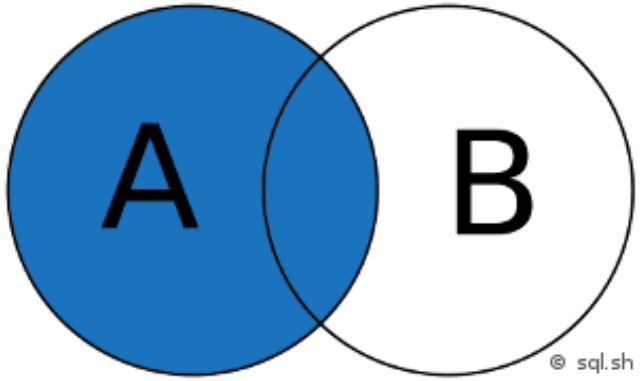
- Jointure interne pour retourner les enregistrements quand la **condition est vraie** dans les deux tables
  - SELECT * FROM A INNER JOIN B ON A.key = B.key



  - SELECT * FROM film INNER JOIN distrib ON film.id_distrib = distrib.id_distrib

LEFT JOIN

- Jointure externe pour retourner tous les enregistrements de la table de gauche **même si la condition n'est pas vérifiée** dans l'autre table
  - SELECT * FROM A LEFT JOIN B ON A.key = B.key



  - SELECT * FROM film INNER JOIN distrib ON film.id_distrib = distrib.id_distrib

Exercices - CRUD

1. Ajouter la salle 42 – Infinite au 4ème étage de votre cinéma. Cette salle contient 1200 places.
2. Affichez toutes les salles de votre cinéma par nombre de sièges.
3. Une extension du 4ème étage permet de gagner 25 places dans la salle infinite. Mettez à jour votre enregistrement.
4. Un dégât des eaux vous force à fermer la salle infinite, agissez en conséquence.

Exercices - SELECT

1. Affichez les salles du RDC et les salles du 3ème étage.
2. Affichez les salles qui sont comprises entre le RDC et le deuxième étage, et les salles qui ont plus de 200 sièges.
3. Affichez toutes les salles dont le nom commence par « p » et finit par « et » sans respecter la casse.

Exercices – les fonctions

1. Faites la moyenne de la durée des films dans votre cinema.
2. Comptez le nombre de salles de votre cinéma.
3. Affichez le nom de la salle qui contient le plus de sièges dans votre cinéma.
4. Affichez la longueur du plus grand nom de salle de votre cinéma.

Exercices - Les jointures

1. Affichez le nom du distributeur des 50 premiers films classés par ordre alphabétique.
   a. Avec un INNER JOIN.
   b. Avec un LEFT JOIN.
2. Que voyez-vous ?
3. Pourquoi ?

Conclusion

- MySQL est un langage de programmation simple
- Il obéit grosso modo aux même impératifs que PHP

- SQL est une manière de structurer ses données pour y avoir facilement accès de manière sécurisée
- Mais SQL a ses limites :
  - **Moteur de recherche**
  - **Big data**