Développement PHP
Partie 4 : Bases de données

DENIS LOEUILLET – IFA - 2017

Partie 4 : Bases de données

- Présentation des bases de données
- phpMyAdmin
- Lire les données
- Écrire des données
- Les fonctions SQL
- Les dates en SQL
- Les jointures entre tables

Partie 4 : Bases de données
Les fonctions SQL

1. Les fonctions scalaires
2. Les fonctions d'agrégat
3. GROUP BY et HAVING : le groupement de données
4. Résumé

Partie 4 : Bases de données
Les fonctions SQL

- SQL propose toute une série de fonctions !
  - Le langage SQL permet d'effectuer des calculs directement sur ses données à l'aide de fonctions toutes prêtes.
  - Celles-ci sont moins nombreuses qu'en PHP :
    - mais elles sont spécialement dédiées aux bases de données
    - très puissantes dans la pratique.
    - elles permettent par exemple :
      - de récupérer très simplement le prix moyen de l'ensemble des jeux de notre exemple,
      - de compter le nombre de jeux que possède chaque personne,
      - d'extraire le jeu le plus cher ou le moins cher,
      - etc.
    - Les fonctions se révèlent également indispensables lorsqu'on doit travailler avec des dates en SQL.

Partie 4 : Bases de données
Les fonctions SQL

- SQL propose toute une série de fonctions !
  - Les fonctions SQL peuvent être classées en deux catégories :
    - ✓ les fonctions scalaires :
      - elles agissent sur chaque entrée.
      - Par exemple, vous pouvez transformer en majuscules la valeur de chacune des entrées d'un champ ;
    - ✓ les fonctions d'agrégat :
      - lorsque vous utilisez ce type de fonctions, des calculs sont faits sur l'ensemble de la table pour retourner une valeur.
        - Par exemple, calculer la moyenne des prix retourne une valeur : le prix moyen.

Partie 4 : Bases de données
Les fonctions SQL

1. Les fonctions scalaires
   - Nous allons d'abord découvrir le mode d'emploi d'une fonction SQL de type scalaire :
     - la fonction UPPER.
   - Lorsque vous aurez appris à vous en servir, vous serez capables de faire de même avec toutes les autres fonctions scalaires.
   - Nous verrons ensuite une petite sélection de fonctions scalaires à connaître

Partie 4 : Bases de données
Les fonctions SQL : les fonctions scalaires

- Utiliser une fonction scalaire SQL
  - Pour nos exemples nous allons nous baser sur la table jeux_video
  - On écrit les noms des fonctions SQL en majuscules :
    - comme on le fait déjà pour la plupart des mots-clés comme SELECT, INSERT, etc.
    - Ce n'est pas une obligation mais plutôt une convention
  - Exemple :
    - la fonction UPPER() :
      - qui permet de convertir l'intégralité d'un champ en majuscules.
    - Supposons que nous souhaitions obtenir les noms de tous les jeux en majuscules ; voici comment on écrirait la requête SQL :

```
1  SELECT UPPER(nom) FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions scalaires

- Utiliser une fonction scalaire SQL

```
1  SELECT UPPER(nom) FROM jeux_video
```

  - Exemple :
    - ✓ La fonction UPPER est utilisée sur le champ nom.
    - ✓ On récupère ainsi tous les noms des jeux en majuscules.
    - ✓ Ça ne modifie pas le contenu de la table!!
    - ✓ La fonction UPPER modifie seulement la valeur envoyée à PHP.
      - ❖ On ne touche donc pas au contenu de la table.
    - ✓ Cela crée en fait un « champ virtuel » qui n'existe que le temps de la requête :
      - ❖ Il est conseillé de donner un nom à ce champ virtuel qui représente les noms des jeux en majuscules.
      - ❖ Il faut utiliser pour cela le mot-clé AS, comme ceci :

```
1  SELECT UPPER(nom) AS nom_maj FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions scalaires

- Utiliser une fonction scalaire SQL
  - Exemple :

```
1  SELECT UPPER(nom) AS nom_maj FROM jeux_video
```

    - ✓ On récupère les noms des jeux en majuscules via un champ virtuel appelé « nom_maj ».
    - ✓ Ce champ virtuel est appelé alias.
    - ✓ On peut s'en servir en PHP pour afficher les noms des jeux en majuscules :
      - ❖ Script upper.php
      - ❖ PHP ne récupère qu'un champ nommé nom_maj (même s'il n'existe pas dans la table).
      - ❖ En affichant le contenu de ce champ, on ne récupère que les noms des jeux en majuscules.
    - ✓ Pour récupérer le contenu des autres champs comme avant sans forcément leur appliquer une fonction : script upper_2.php

```
1  SELECT UPPER(nom) AS nom_maj, possesseur, console, prix FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions scalaires

- Présentation de quelques fonctions scalaires utiles
  - LOWER : convertir en minuscules

```
1  SELECT LOWER(nom) AS nom_min FROM jeux_video
```

    - ✓ Cette fois, le jeu « Sonic » sera renvoyé sous la forme « sonic » dans un champ nommé nom_min. (script lower.php)
  - LENGTH : compter le nombre de caractères

```
1  SELECT LENGTH(nom) AS longueur_nom FROM jeux_video
```

    - ✓ Pour « Sonic », on récupèrera donc la valeur 5 dans un champ longueur_nom.(script length.php)

Partie 4 : Bases de données
Les fonctions SQL : les fonctions scalaires

- Présentation de quelques fonctions scalaires utiles
  - ROUND : arrondir un nombre décimal (script round.php)

```
1  SELECT ROUND(prix, 2) AS prix_arrondi FROM jeux_video
```

    - ✓ La fonction ROUND() s'utilise sur des champs comportant des valeurs décimales.
    - ✓ deux paramètres :
      - ❖ le nom du champ à arrondir
      - ❖ le nombre de chiffres après la virgule que l'on souhaite obtenir.
    - ✓ Ainsi, si un jeu coûte 25,86999 euros, on obtiendra la valeur 25,87 euros dans un champ prix_arrondi.
  - Il existe beaucoup d'autres fonctions SQL du même type
    - ✓ La documentation de MySQL vous propose une liste bien plus complète de fonctions mathématiques (comme ROUND) et de fonctions sur les chaînes de caractères (comme UPPER).

Partie 4 : Bases de données
Les fonctions SQL

2. Les fonctions d'agrégat
   - Nous allons d'abord voir comment on utilise une fonction d'agrégat dans une requête SQL et comment on récupère le résultat en PHP,
   - puis je vous présenterai une sélection de fonctions à connaître.
   - il en existe bien d'autres que vous pourrez découvrir dans la documentation.
   - L'essentiel est de comprendre comment s'utilise ce type de fonctions :
     - vous pourrez ensuite appliquer à n'importe quelle autre fonction du même type.

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Utiliser une fonction d'agrégat SQL
  - Ces fonctions diffèrent assez des précédentes.
  - Plutôt que de modifier des valeurs une à une, elles font des opérations sur plusieurs entrées pour retourner une seule valeur.
  - Par exemple :
    - ✓ ROUND permettait d'arrondir chaque prix.
      - ❖ On récupérait autant d'entrées qu'il y en avait dans la table.
    - ✓ En revanche, une fonction d'agrégat comme AVG :
      - ❖ renvoie une seule entrée : la valeur moyenne de tous les prix.
  - Exemple :la fonction d'agrégat AVG :
    - ✓ Elle calcule la moyenne d'un champ contenant des nombres.
    - ✓ Utilisons-la sur le champ prix :

```
1  SELECT AVG(prix) AS prix_moyen FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Utiliser une fonction d'agrégat SQL
  - Exemple :la fonction d'agrégat AVG :

```
1  SELECT AVG(prix) AS prix_moyen FROM jeux_video
```

    - ✓ On donne un alias au résultat donné par la fonction : prix_moyen
    - ✓ La particularité, c'est que cette requête ne va retourner qu'une seule entrée :
      - ❖ le prix moyen de tous les jeux : prix_moyen
    - Exemple : script avg.php
      - ✓ Inutile de faire une boucle étant donné qu'on sait qu'on ne va récupérer qu'une seule entrée, puisqu'on utilise une fonction d'agrégat !!
      - ✓ On peut se permettre d'appeler fetch() une seule fois et en dehors d'une boucle étant donné qu'il n'y a qu'une seule entrée. Le code suivant est donc un peu plus adapté dans le cas présent : script avg_2.php
      - ✓ Ce code est plus simple et plus logique. On récupère la première et seule entrée avec fetch() et on affiche ce qu'elle contient, puis on ferme le curseur.

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- N'hésitez pas à filtrer
  - Exemple : script avg_where.php
    - ✓ le prix moyen des jeux appartenant à Patrick.

```
1  SELECT AVG(prix) AS prix_moyen FROM jeux_video WHERE possesseur='Patrick'
```

  - ❖ Le calcul de la moyenne ne sera fait que sur la liste des jeux qui appartiennent à Patrick.
  - ❖ On peut même combiner les conditions pour obtenir le prix moyen des jeux de Patrick qui se jouent à un seul joueur.

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Ne pas mélanger une fonction d'agrégat avec d'autres champs
  - ➢ On ne doit pas récupérer d'autres champs de la table quand on utilise une fonction d'agrégat, contrairement à tout à l'heure avec les fonctions scalaires.
    - ✓ En effet, quel sens cela aurait-il de faire :

```
1  SELECT AVG(prix) AS prix_moyen, nom FROM jeux_video
```

      - ❖ On récupérerait :
        - o d'un côté le prix moyen de tous les jeux
        - o de l'autre la liste des noms de tous les jeux
        - ➢ Il est impossible de représenter ceci dans un seul et même tableau.
      - ❖ Comme vous le savez, SQL renvoie les informations sous la forme d'un tableau.
        - o Or on ne peut pas représenter la moyenne des prix (qui tient en une seule entrée) en même temps que la liste des jeux.
          - Si on voulait obtenir ces deux informations il faudrait faire deux requêtes.

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Présentation de quelques fonctions d'agrégat utiles
  - AVG : calculer la moyenne
    - ✓ Cette fonction retourne la moyenne d'un champs contenant des nombres :

```
1  SELECT AVG(prix) AS prix_moyen FROM jeux_video
```

  - SUM : additionner les valeurs
    - ✓ Cette fonction permet d'additionner toutes les valeurs d'un champ : script sum.php

```
1  SELECT SUM(prix) AS prix_total FROM jeux_video WHERE possesseur='Patrick'
```

  - MAX : retourner la valeur maximale (script max.php)
    - ✓ Cette fonction analyse un champ et retourne la valeur maximale trouvée.

```
1  SELECT MAX(prix) AS prix_max FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Présentation de quelques fonctions d'agrégat utiles
  - MIN : retourner la valeur minimale (script min.php)
    - ✓ Cette fonction analyse un champ et retourne la valeur minimale trouvée :

```
1  SELECT MIN(prix) AS prix_min FROM jeux_video
```

  - COUNT : compter le nombre d'entrées (count.php)
    - ✓ Cette fonction permet de compter le nombre d'entrées.
    - ✓ Elle est très intéressante mais plus complexe.
    - ✓ On peut l'utiliser de plusieurs façons différentes :
      - ❖ L'utilisation la plus courante consiste à lui donner * en paramètre :

```
1  SELECT COUNT(*) AS nbjeux FROM jeux_video
```

        - On obtient le nombre total de jeux dans la table.

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Présentation de quelques fonctions d'agrégat utiles
  - COUNT : compter le nombre d'entrées
    - ✓ On peut filtrer avec une clause WHERE : script count_where.php

```
1  SELECT COUNT(*) AS nbjeux FROM jeux_video WHERE possesseur='Florent'
```

      - ❖ retourne le nombre de jeux appartenant à Florent
    - ✓ Il est possible de compter uniquement les entrées pour lesquelles l'un des champs n'est pas vide :
      - ❖ c'est-à-dire que le champs ne vaut pas NULL. (count_nonull.php)

```
1  SELECT COUNT(nbre_joueurs_max) AS nbjeux FROM jeux_video
```

Partie 4 : Bases de données
Les fonctions SQL : les fonctions d'agrégat

- Présentation de quelques fonctions d'agrégat utiles
  - COUNT : compter le nombre d'entrées
    - ✓ Enfin, il est possible de compter le nombre de valeurs distinctes sur un champ précis.
    - ✓ Par exemple dans la colonne « possesseur » :
      - ❖ Florent apparaît plusieurs fois, Patrick aussi, etc.
      - ❖ Mais combien y a t-il de personnes différentes dans la table ?
      - ❖ On peut le savoir en utilisant le mot-clé DISTINCT devant le nom du champ à analyser

```
1  SELECT COUNT(DISTINCT possesseur) AS nbpossesseurs FROM jeux_video
```

(script count_distinct.php)

Partie 4 : Bases de données
Les fonctions SQL

3. GROUP BY et HAVING : le groupement de données

- On a vu un peu plus tôt qu'on ne pouvait pas récupérer d'autres champs lorsqu'on utilisait une fonction d'agrégat.
  - Exemple :

```
1  SELECT AVG(prix) AS prix_moyen, console FROM jeux_video
```

  - ✓ Ça n'a pas de sens de récupérer le prix moyen de tous les jeux et le champ « console » à la fois :
    - Il n'y a qu'un seul prix moyen pour tous les jeux,
    - mais plusieurs consoles.
    - MySQL ne peut pas renvoyer un tableau correspondant à ces informations-là.

Partie 4 : Bases de données
Les fonctions SQL : le groupement de données

- GROUP BY : grouper des données
  - En revanche, ce qui pourrait avoir du sens, ce serait de demander le prix moyen des jeux pour chaque console !
  - Pour faire cela, on doit utiliser un nouveau mot-clé :
    - ✓ GROUP BY : signifie « grouper par ».
    - ✓ On utilise cette clause en combinaison d'une fonction d'agrégat (comme AVG) pour obtenir des informations intéressantes sur des groupes de données. (Script group_by.php)

```
1  SELECT AVG(prix) AS prix_moyen, console FROM jeux_video GROUP BY console
```

- ✓ Il faut utiliser GROUP BY en même temps qu'une fonction d'agrégat, sinon il ne sert à rien.
- ✓ Ici, on récupère le prix moyen et la console, et on choisit de grouper par console.
- ✓ on obtiendra la liste des différentes consoles de la table et le prix moyen des jeux de chaque plate-forme !

Partie 4 : Bases de données
Les fonctions SQL : le groupement de données

- GROUP BY : grouper des données

```
1  SELECT AVG(prix) AS prix_moyen, console FROM jeux_video GROUP BY console
```

- ✓ Cette fois les valeurs sont cohérentes ! On a la liste des consoles et le prix moyen des jeux associés.
- ✓ Exercice : (script group_by_valeur_totale.php)
  - ❖ essayez d'obtenir de la même façon la valeur totale des jeux que possède chaque personne.

prix_moyen	console
12.67	Dreamcast
5.00	Gameboy
47.50	GameCube

Partie 4 : Bases de données
Les fonctions SQL : le groupement de données

- HAVING : filtrer les données regroupées
  - HAVING est un peu l'équivalent de WHERE
  - mais il agit sur les données une fois qu'elles ont été regroupées.
  - C'est donc une façon de filtrer les données à la fin des opérations.(script HAVING.php)

```
1  SELECT AVG(prix) AS prix_moyen, console FROM jeux_video GROUP BY console HAVING prix_moyen <=
   10
```

    - ✓ on récupère uniquement la liste des consoles et leur prix moyen si ce prix moyen ne dépasse pas 10 euros.
  - HAVING ne doit s'utiliser que sur le résultat d'une fonction d'agrégat.
    - ✓ Voilà pourquoi on l'utilise ici sur prix_moyen et non sur console.

Partie 4 : Bases de données
Les fonctions SQL : le groupement de données

- HAVING : filtrer les données regroupées
  - ➢ Quelle est la différence entre WHERE et HAVING.
    - ✓ Les deux permettent de filtrer mais pas au même moment
      - ❖ WHERE agit en premier, avant le groupement des données,
      - ❖ HAVING agit en second, après le groupement des données.
      - ❖ On peut d'ailleurs très bien combiner les deux : (script having_where.php)

```
1  SELECT AVG(prix) AS prix_moyen, console FROM jeux_video WHERE possesseur='Patrick' GROUP BY
   console HAVING prix_moyen <= 10
```

      - ❖ Ici, on demande à récupérer le prix moyen par console de tous les jeux de Patrick (WHERE), à condition que le prix moyen des jeux de la console ne dépasse pas 10 euros (HAVING).

Partie 4 : Bases de données
Les fonctions SQL

3. Résumé

- MySQL permet d'exécuter certaines fonctions lui-même, sans avoir à passer par PHP. Ces fonctions modifient les données renvoyées.
- Il existe deux types de fonctions :
  - les fonctions scalaires : elles agissent sur chaque entrée récupérée. Elles permettent par exemple de convertir tout le contenu d'un champ en majuscules ou d'arrondir chacune des valeurs ;
  - les fonctions d'agrégat : elles effectuent des calculs sur plusieurs entrées pour retourner une et une seule valeur. Par exemple : calcul de la moyenne, somme des valeurs, comptage du nombre d'entrées…
- On peut donner un autre nom aux champs modifiés par les fonctions en créant des alias à l'aide du mot-clé AS.
- Lorsqu'on utilise une fonction d'agrégat, il est possible de regrouper des données avec GROUP BY.
- Après un groupement de données, on peut filtrer le résultat avec HAVING. Il ne faut pas le confondre avec WHERE qui filtre avant le groupement des données.