Programmation Web Dynamique 2

10. Requêtes complexes (fin)

Groupes et fonctions d’agrégation

- Un groupe est un sous-ensemble de lignes d’une table qui possède les mêmes valeurs pour certaines colonnes.
  - Ex : Les films dans la table film qui possèdent le même réalisateur ont tous la même valeur dans la colonne RealisateurFilm
  - Ex : Les articles dans une table Articles qui ont été écrit à la même date possèdent la même valeur dans la colonne DateCreation
- On peut définir des groupes pour les utiliser avec des fonctions d’agrégation, qui nous permettront d’effectuer des totals, des moyennes, des écarts-types, ou de trouver les maximums et les minimums de ces groupes.

Création d'un groupe

- On crée un groupe dans une requête en utilisant la clause GROUP BY, que l'on ajoute après notre clause WHERE.
- Cette clause regroupera toutes les rangées qui possède les mêmes valeurs dans les colonnes spécifiées.
- On l'utilise comme suit :
  - SELECT nomrealisateur FROM realisateurs JOIN films ON films.realisateur = realisateurs.idrealisateur WHERE films.annee = 1990 GROUP BY nomrealisateur;
    - Cette requête trouvera tous les réalisateurs qui ont réalisé un film sorti en 1990, et vous n'obtiendrez pas de doublons, puisque les rangées possédant la même valeur dans nomrealisateur sont GROUPÉES ensemble par la clause GROUP BY.

Fonctions d’agrégation

- **AVG :** Calcule la moyenne des valeurs de l’expression envoyée en paramètres (généralement une colonne).
- **COUNT :** Dénombre le nombre de lignes du groupe. Si une expression est envoyée en paramètres, on ne compte que les lignes pour lesquelles cette expression n’est pas NULL. On peut lui mettre * en paramètres.
- **MAX :** Retourne la plus petite des valeurs de l’expression envoyée en paramètres.
- **MIN :** Retourne la plus grande des valeurs de l’expression envoyée en paramètres.
- **STDDEV :** Calcule l’écart-type des valeurs de l’expression envoyée en paramètres.
- **SUM :** Calcule la somme des valeurs de l’expression envoyée en paramètres.
- **VARIANCE :** Calcule la variance des valeurs de l’expression envoyée en paramètres.

Exemples :

- SELECT nomrealisateur, COUNT(*) AS NombreFilms FROM realisateurs JOIN films ON films.realisateur = realisateurs.idrealisateur WHERE films.annee = 1990 GROUP BY nomrealisateur;
  - Retourne le nom du réalisateur mais aussi le nombre de films qu'il a réalisé lors de cette année!
- Avec une table Equipes contenant un nom d'équipe et une table Joueurs contenant une rangée par joueur (nom, prénom, salaire):
  - SELECT NomEquipe, SUM(salaireJoueur) AS MasseSalariale FROM Joueurs JOIN Equipes ON Joueurs.idequipe = equipe.id GROUP BY Joueurs.idequipe;
  - Retourne le nom de l'équipe et la masse salariale de cette équipe en faisant la somme du salaire de tous les joueurs qui sont dans la même équipe. Les joueurs sont groupés ensemble par équipe à cause de la clause GROUP BY.

La clause HAVING

- La clause HAVING permet d'effectuer une restriction sur les lignes groupées par un GROUP BY, à la manière de la clause WHERE vu précédemment. Cependant, la clause HAVING ne peut s'appliquer que sur les résultats de fonctions d'agrégations.
- Il est possible d'avoir à la fois une clause WHERE et une clause HAVING.
  - Dans ce cas, la clause WHERE définit quelles rangées seront sélectionnées pour être mises dans les rangées groupées.
  - Une fois les groupes générés à partir des rangées sélectionnées et de la clause GROUP BY, on applique la clause HAVING sur les groupes pour en enlever les éléments que l'on ne veut pas.
  - Ex : SELECT nomrealisateur, COUNT(*) AS NombreFilms FROM realisateurs JOIN films ON films.realisateur = realisateurs.idrealisateur WHERE films.annee = 1990 GROUP BY nomrealisateur HAVING COUNT(*) > 1;
    - Ne sélectionne que les réalisateurs qui ont réalisé PLUS D'UN film en 1990!