SQL

Ch 4 – SELECT Opérateurs et fonctions.

I. Introduction

Les requêtes permettent de récupérer des valeurs de colonnes de tables.
Il est également possible d'utiliser

- des calculs arithmétiques classiques
- ou des fonctions spécifiques

pour construire des réponses adaptées aux requêtes complexes.
Les valeurs ainsi calculées peuvent être utilisées

- comme valeurs renvoyées dans la liste des colonnes retournées (select)
- ou dans des expressions logiques pour limiter le nombre de lignes retournées (where)

Les fonctions sont mises en œuvre dans la plupart des SGBD ; il arrive cependant que des différences existent au niveau du nom d'une fonction, ou bien ses paramètres.
➔ IL EST INDISPENSABLE DE CONSULTER LA DOCUMENTATION DU SGBD SUR LEQUEL VOUS ALLEZ TRAVAILLER avant l'utilisation des fonctions.

II. Colonnes calculées et opérateurs arithmétiques

Les opérateurs arithmétiques utilisés en SQL sont identiques à ceux utilisés habituellement en programmation. Ils sont utilisés pour écrire des expressions arithmétiques basées sur les valeurs de colonnes et des valeurs littérales numériques.

A. Exemple

Exemple de la table des livres :
t_livres (id_livre, titre_livre, prix_base_livre)

On souhaite obtenir un catalogue avec les prix publics (calculées en fonction du prix de base majoré de 25%) :

```
SELECT id_livre, titre_livre, (prix_base_livre * 1.25)
    FROM t_livre ;
```

Il est souhaitable d'attribuer un nom aux colonnes calculées :

```
SELECT id_livre, titre_livre, (prix_base_livre * 1.25) AS
prixCatalogue
    FROM t_livre ;
```

B. Opérateurs

opérateur	Opération réalisée	priorité
+	addition	Moins prioritaire
-	soustraction	Moins prioritaire
/	division	prioritaire
*	multiplication	prioritaire
%	Modulo (reste entier après la division de 2 entiers)	prioritaire

Lorsque plusieurs opérateurs sont utilisés dans une expression arithmétique, la priorité des opérateurs revêt une grande importance (comme dans tout autre contexte de calcul)
La priorité des opérateurs détermine l'ordre selon lequel les calculs intermédiaires sont réalisés. Afin de lever toute ambiguïté, il est important d'utiliser des parenthèses pour déterminer les calculs intermédiaires (meilleure lisibilité des formules et assurance d'un calcul sûr).

Exemple de la table des livres :
On souhaite obtenir un catalogue avec les prix publics (calculées en fonction du prix de base auquel on ajoute un montant fixe de 1.5, le tout majoré de 25%) :

```
SELECT id_livre, titre_livre,
    prix_base_livre + 1.5 * 1.25 AS prixCatalogue
    FROM t_livre ;                          ➔ FAUX
```

Il est INDISPENSABLE ICI de préciser l'ordre dans lequel le calcul doit être réalisé : d'attribuer un nom aux colonnes calculées :

```
SELECT id_livre, titre_livre,
    (prix_base_livre + 1.5) * 1.25 AS prixCatalogue
    FROM t_livre ;        ➔ BON
```

III. Fonctions mathématiques

A. Exemple

Exemple de la table des livres :
t_livres (id_livre, titre_livre, prix_base_livre)

On souhaite obtenir un catalogue avec les prix publics (calculées en fonction du prix de base majoré de 25%) arrondi à 1 chiffre :

```
SELECT id_livre, titre_livre, ROUND((prix_base_livre * 1.25), 1) AS prixCatalogue
    FROM t_livre ;
```

B. Quelques fonctions mathématiques

fonction	Résultat renvoyé
ABS	Valeur absolue de la valeur spécifiée
ACOS, ASIN, ATAN	Angle exprimé en radians dont le cosinus, le sinus ou la tangente est fourni
COS, SIN, TAN, COT	cosinus, sinus ou tangente d'un angle exprimé en radians
CEILING	Plus petit entier supérieur ou égal à la valeur spécifiée.
DEGREES	Conversion de radians en degrés
EXP	Valeur exponentielle de la valeur spécifiée
FLOOR	**Le plus grand entier inférieur ou égal à la valeur spécifiée**
LOG	Logarithme naturel de la valeur spécifiée
LOG10	Logarithme en base 10 de la valeur spécifiée
PI	Valeur 3.1415927....
POWER (x, y)	Valeur de x élevée à la puissance y
RADIANS	Conversion de degrés en radians
RAND	Nombre aléatoire entre 0 et 1
ROUND(x, y)	**Arrondi de x avec y chiffres**
SIGN	Signe de la valeur spécifiée
SQRT	Racine carrée de la valeur spécifiée

Exemple de valeurs retournées :
Floor(123.45) ➔ 123
Floor(-123.45) ➔ -124

IV. Fonctions de dates

A. Exemple

Exemple de la table des livres :
t_paraitre (id_livre, date_parution)

On souhaite obtenir les livres parus au mois de juin (quelque soit l'année) :

```
SELECT id_livre
```



```
    FROM t_livre
    WHERE MONTH(date_parution) = 6 ;
```

A. Quelques fonctions de date

fonction	Valeur renvoyée
EXTRACT	Partie de date extraite
YEAR	Retourne l'année d'une date
MONTH	Retourne le mois d'une date
DAY	Retourne le jour dans le mois

V. Fonctions de textes

Les fonctions de chaînes de caractères permettent la manipulation des colonnes alphanumériques (types SQL : CHAR, VARCHAR)

A. Exemple

Exemple de la table des livres :
t_livres (id_livre, titre_livre, prix_base_livre)

On souhaite obtenir un catalogue les titres en majuscule :

```
SELECT UPPER(titre_livre) AS titre
    FROM t_livre ;
```

On souhaite obtenir une liste avec 'livre : ' et le titre du livre ::

```
SELECT "livre : " + UPPER(titre_livre) AS titre
    FROM t_livre ;
```

B. Quelques fonctions de manipulation de chaines

opérateur	signification
+ || CONCAT	Concaténation (dépend du SGBD)

fonction	Valeur renvoyée
LOWER	Convertit en minuscule
LTRIM	Supprime des espaces devant une colonne chaîne de caractères (à gauche : L=left)
REVERSE	Renvoie l'inverse
RIGHT(x,y)	Renvoie y caractères de x à partir de la droite
RTRIM	Supprime des espaces derrière une colonne chaîne de caractères (à droite : R=right)
SOUNDEX	Renvoie un code à 4 chiffres permettant d'évaluer la similitude entre 2 chaînes de caractères
SPACE	Revoie une chaîne composées d'espaces
STR	Conversion d'une donnée numérique en chaîne de caractères

SUBSTRING(x, y, z)	Renvoie z caractères de x à partir du caractère à la position y
UPPER	**Convertit en majuscule**

VI. Fonctions système et fonctions générales

A. Exemple

Exemple de la table des livres :

t_livres (id_livre, titre_livre, prix_base_livre)

On souhaite obtenir un catalogue des titres, avec la date actuelle :

```
SELECT titre_livre, CURRENT_DATE()
     FROM t_livre ;
```

Exemple de la table des emprunts de livres :

t_emprunt (id_emprunteur, #id_livre, dateEmprunt, dateRetour)

On souhaite produire une relance pour les dates de retour qui sont dépassées aujourd'hui (soit dateRetour < date d'aujourd'hui)

```
SELECT *, CURRENT_DATE()
     FROM t_emprunt
     WHERE dateRetour < CURRENT_DATE();
```

B. Liste

fonction	Valeur renvoyée
CURRENT_DATE	**Date courante**
CURRENT_TIME	Heure courante
CAST	Transtypage (changement de type)
COALESCE	Définir une valeur pour remplacer une valeur nulle

VII. Compléments

Bien que certaines fonctions soient « relativement » standard, il est indispensable de se rapprocher de la documentation du SGBD pour leur utilisation.
Voir une synthèse comparative des fonctions par SGBD : : http://sqlpro.developpez.com/cours/sqlaz/fonctions/

VIII. Exercices

Soient les définitions des tables **t_personne**, **t_chien**, **t_race**, **t_aliment** et **t_nourrir** :

- **t_personne** (nomPersonne, dateNaissancePersonne)
- **t_chien** (nomChien, #nomPersonne, poids, dateNaissanceChien, #nomRace)
- **t_race** (nomRace, prixDeBase)
- **t_aliment** (codeAliment, prixUnitaire)
- **t_nourrir** (#nomChien, #codeAliment, qteJour)

Les colonnes soulignés représentent les identifiants (clefs primaires), les colonnes préfixées par #, les clefs étrangères.

Pour chacune des questions, proposez la requête SQL qui permettra d'y répondre.

Question 1 : pour quelle quantité le chien « pollux » consommera t –il, de chaque aliment dont il se nourrit, en décembre ? (en décembre, il y a 31 jours...)

```
SELECT codeAliment, (qteJour * 31) AS quantiteDecembre
  FROM t_nourrir
  WHERE nomChien = "pollux";
```

On veut être sûr de trouver l'égalité avec 'pollux' (minuscule, pas d'espaces devant) : avant d'effectuer la comparaison, on supprime les espaces devant de 'nomChien' puis on converti en minuscules :

```
SELECT codeAliment, (qteJour * 31) AS quantiteDecembre
  FROM t_nourrir
  WHERE LOWER(LTRIM(nomChien)) = "pollux";
```

Question 2 : pour quelle valeur le chien « pollux » consommera t –il, de chaque aliment dont il se nourrit, en décembre ?

```
SELECT A.codeAliment, (qteJour * 31 * prixUnitaire) AS
valeurDecembre
  FROM t_nourrir A INNER JOIN t_aliment B
        ON A.codeAliment = B.codeAliment
  WHERE nomChien = "pollux";
```

Question 3+ : donnez la liste des chiens avec leur âge.
➔ On utilise le calcul de l'âge suivant : (Année courante – Année de naissance) + 1 : un chiot né le "01/06/2007" aura donc (2007 – 2007) + 1 = 1 an

```
SELECT nomChien, (((YEAR(CURRENT_DATE) –
YEAR(dateNaissanceChien)) + 1 ) AS ageDuChien
  FROM t_chien ;
```

Question 3 : le prix d'un chien est proportionnel à son age : il s'agit d'appliquer au prix de base le coefficient 1/age : donnez la liste des chiens avec leur prix.

```
SELECT nomChien, prixDeBase * 1 / (((YEAR(CURRENT_DATE) -
YEAR(dateNaissanceChien)) + 1 )) AS tarif
  FROM t_chien A INNER JOIN t_race B
        ON A.nomRace = B.nomRace ;
```

Question 4 : Quel montant devra dépenser chaque maître pour la nourriture de ses chiens au mois de décembre

```
SELECT D.nomPersonne, SUM((qteJour * 31 * prixUnitaire))
AS depenseDecembre
  FROM t_nourrir A INNER JOIN t_aliment B
        ON A.codeAliment = B.codeAliment
                  INNER JOIN t_chien C
        ON A.nomChien = C.nomChien
                  INNER JOIN t_personne D
        ON C.nomPersonne = D.nomPersonne
  GROUP BY nomPersonne;
```