Cours sur les fonctionnalités pratiques de MariaDB

👤 Propriétaire	Ⓜ Marine
☰ Étiquettes	

Introduction

MariaDB est un système de gestion de base de données puissant et riche en fonctionnalités. Ce cours explore :

1. **Gestion des dates** (ex. : `DATEDIFF`, `TIMESTAMPDIFF`).
2. **Fonctions conditionnelles** (ex. : `IF`, `CASE`).
3. **Utilisation des boucles et procédures stockées** (ex. : `WHILE`, `LOOP`).
4. **Manipulation des chaînes de caractères** (ex. : `CONCAT`, `SUBSTRING`).

Chaque partie comprend une **explication**, des **exemples** et des **exercices pratiques**.

1. Gestion des dates

a. Fonction `DATEDIFF`

`DATEDIFF` calcule la différence en jours entre deux dates.

Syntaxe :

```
SELECT DATEDIFF(date1, date2) AS difference_in_days;
```

exemple :

```
SELECT DATEDIFF('2025-01-24', '2025-01-01') AS difference_in_
-- Résultat : 23 jours
```

b. Fonction `TIMESTAMPDIFF`

`TIMESTAMPDIFF` permet de calculer des différences dans d'autres unités (années, mois, heures, etc.).

Syntaxe :

```
SELECT TIMESTAMPDIFF(unit, date1, date2) AS difference;
```

exemple :

```
SELECT TIMESTAMPDIFF(YEAR, '2000-01-01', '2025-01-24') AS dif
-- Résultat : 25 ans
```

Exercice 1 : Comparaison de dates

1. Trouvez la différence en jours entre deux dates données.
2. Calculez combien de mois se sont écoulés entre `'2023-01-01'` et `'2025-01-01'`.

2. Fonctions conditionnelles

a. Fonction `IF`

La fonction `IF` permet d'exécuter une logique conditionnelle directement dans une requête SQL.

Syntaxe :

```
SELECT IF(condition, value_if_true, value_if_false);
```

Exemple :

```
SELECT IF(5 > 3, 'Vrai', 'Faux') AS result;
-- Résultat : 'Vrai'
```

b. Expression `CASE`

L'expression `CASE` offre une structure conditionnelle plus flexible.

Syntaxe :

```
SELECT
  CASE
    WHEN condition1 THEN result1
    WHEN condition2 THEN result2
    ELSE default_result
  END AS alias;
```

exemple :

```
SELECT
  CASE
    WHEN age < 18 THEN 'Mineur'
    WHEN age >= 18 THEN 'Majeur'
    ELSE 'Inconnu'
  END AS statut
FROM personnes;
```

Exercice 2 : Conditions

1. Créez une requête qui affiche "Adulte" si l'âge est supérieur ou égal à 18, sinon "Mineur".
2. Ajoutez une condition qui vérifie si un salaire dépasse 50 000 et affiche "Haut revenu" ou "Bas revenu".

3. Boucles et procédures stockées

a. Boucle `WHILE`

Les boucles permettent d'automatiser des tâches répétitives dans des procédures.

Syntaxe :

```
WHILE condition DO
    -- Instructions
END WHILE;
```

exemple :

```
CREATE PROCEDURE boucle_exemple()
BEGIN
    DECLARE compteur INT DEFAULT 1;
    WHILE compteur <= 5 DO
        SELECT CONCAT('Itération numéro : ', compteur);
        SET compteur = compteur + 1;
    END WHILE;
END;
//
DELIMITER ;
CALL boucle_exemple();
```

b. Boucle **LOOP**

Une boucle générique qui s'exécute jusqu'à une condition d'arrêt.

Syntaxe :

```
DELIMITER //
CREATE PROCEDURE boucle_loop()
BEGIN
    DECLARE compteur INT DEFAULT 1;
    boucle: LOOP
        IF compteur > 5 THEN
            LEAVE boucle;
        END IF;
        SELECT CONCAT('Itération numéro : ', compteur);
        SET compteur = compteur + 1;
    END LOOP boucle;
END;
```

```
//  
DELIMITER ;  
CALL boucle_loop();
```

Exercice 3 : Boucles

1. Créez une procédure qui affiche les nombres de 1 à 10.
2. Ajoutez une condition pour afficher "Pair" ou "Impair" pour chaque nombre.

4. Manipulation des chaînes de caractères

a. Fonction **CONCAT**

Permet de concaténer plusieurs chaînes.

Exemple :

```
SELECT CONCAT('Bonjour, ', 'monde !') AS message;  
-- Résultat : 'Bonjour, monde !'
```

b. Fonction **SUBSTRING**

Extrait une partie d'une chaîne.

Code :

```
SELECT SUBSTRING(string, start_position, length) AS result;
```

c. Fonction **REPLACE**

Remplace une partie d'une chaîne par une autre.

```
SELECT REPLACE('Bonjour, monde !', 'monde', 'MariaDB') AS mes  
-- Résultat : 'Bonjour, MariaDB !'
```

5. Intégration des concepts

Projet final

Créer une procédure qui combine les fonctionnalités apprises :

1. Calculer l'âge des utilisateurs à partir de leur date de naissance (avec `TIMESTAMPDIFF`).
2. Ajouter une condition pour indiquer si chaque utilisateur est "Mineur" ou "Majeur".
3. Concaténer leur prénom et nom pour afficher un message personnalisé (ex. : "Bonjour, Jean Dupont!").

Étapes :

1. Créez une table avec des noms, dates de naissance et âges.
2. Écrivez une requête qui utilise `TIMESTAMPDIFF` pour calculer les âges.
3. Utilisez `CASE` pour catégoriser les utilisateurs selon leur âge.
4. Ajoutez une concaténation des chaînes pour afficher un message.