

Rapport : les curseurs en MySQL

Module : Base de Données MySQL

Nom: Aya ouazzani akkar hammoudi

SOMMAIRE :

- 1) Le contexte et les objectifs pédagogiques
- 2) Le code SQL complet
- 3) Des captures d'écran prouvant l'exécution des requêtes
- 4) Une analyse des résultats

1) Le contexte:

Cet atelier s'inscrit dans le cadre de la gestion budgétaire pour des projets. L'objectif est de pratiquer l'utilisation des curseurs en MySQL afin de parcourir ligne par ligne les données d'une table, effectuer des calculs, mettre à jour les informations et générer des alertes.

les objectifs pédagogiques:

Créer et exécuter une procédure stockée utilisant un **curseur**.

Calculer automatiquement le **taux d'utilisation du budget** pour chaque projet.

Mettre à jour le **statut** des projets selon leur consommation.

Générer des **alertes dynamiques** en cas de dépassement.

Tracer toutes les opérations dans un **historique** pour consultation future.

Créer une **fonction personnalisée réutilisable** pour calculer le taux d'un projet spécifique.

2) Le code SQL :

Étape 1 – Création de la base et des tables

-- Création de la base

```
1 CREATE DATABASE IF NOT EXISTS gestion_projets;
2 USE gestion_projets;
```

-- Table principale

```

1 CREATE TABLE projets (
2     id_projet INT AUTO_INCREMENT PRIMARY KEY,
3     nom VARCHAR(100),
4     budget DECIMAL(10,2),
5     depenses DECIMAL(10,2),
6     taux_utilisation DECIMAL(5,2),
7     statut VARCHAR(20)
8 );
9
10 -- Table secondaire : alertes
11 CREATE TABLE alertes (
12     id_alerte INT AUTO_INCREMENT PRIMARY KEY,
13     id_projet INT,
14     message VARCHAR(255),
15     date_alerte DATETIME
16 );
17 -- Table historique : pour tracer les calculs
18 CREATE TABLE historique (
19     id INT AUTO_INCREMENT PRIMARY KEY,
20     id_projet INT,
21     taux DECIMAL(5,2),
22     date_calcul DATETIME
23 );
24

```

Étape 2 – Insertion de données de test :

```

1 INSERT INTO projets (nom, budget, depenses) VALUES
2 ('Projet Alpha', 10000, 8500),
3 ('Projet Beta', 5000, 6000),
4 ('Projet Gamma', 8000, 2000);
5 SELECT * FROM projets;

```

Étape 3 – Création de la procédure principale avec curseur :

```

1 DELIMITER //
2
3 CREATE PROCEDURE calculer_taux_utilisation(IN taux_seuil DECIMAL(5,2))
4 BEGIN
5     -- Déclaration des variables
6     DECLARE v_id INT;
7     DECLARE v_budget DECIMAL(10,2);
8     DECLARE v_depenses DECIMAL(10,2);
9     DECLARE v_taux DECIMAL(5,2);
10    DECLARE fin BOOLEAN DEFAULT FALSE;
11
12    -- Déclaration du curseur
13    DECLARE cur_projets CURSOR FOR
14        SELECT id_projet, budget, depenses FROM projets;

```

```

16    -- Gestion de la fin du curseur
17    DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin = TRUE;
18
19    -- Ouverture du curseur
20    OPEN cur_projets;
21
22    -- Boucle de lecture
23    boucle_projets: LOOP
24        FETCH cur_projets INTO v_id, v_budget, v_depenses;
25
26        -- Condition de fin
27        IF fin THEN
28            LEAVE boucle_projets;
29        END IF;
30
31        -- Calcul du taux d'utilisation
32        SET v_taux = (v_depenses / v_budget) * 100;
33
34        -- Mise à jour selon le taux
35        IF v_taux > taux_seuil THEN
36            UPDATE projets
37                SET taux_utilisation = v_taux, statut = 'Depassement'
38                WHERE id_projet = v_id;
39
40            -- Création d'une alerte
41            INSERT INTO alertes (id_projet, message, date_alerte)
42                VALUES (v_id, CONCAT('Depassement du seuil ', taux_seuil, ' % → ', v_taux, ' %'), NOW());
43        ELSE
44            UPDATE projets
45                SET taux_utilisation = v_taux, statut = 'Normal'
46                WHERE id_projet = v_id;
47        END IF;
48
49        -- Enregistrement dans l'historique
50        INSERT INTO historique (id_projet, taux, date_calcul)
51            VALUES (v_id, v_taux, NOW());
52    END LOOP;
53
54    -- Fermeture du curseur
55    CLOSE cur_projets;
56 END //
57
58 DELIMITER ;
59

```

Étape 4 – Exécution et tests :

```

1 CALL calculer_taux_utilisation(90);
2 SELECT * FROM projets;
3 SELECT * FROM alertes;
4 SELECT * FROM historique;
5

```

Étape 6 – Créer une fonction réutilisable :

```

1 DELIMITER //
2
3 CREATE FUNCTION calcul_taux(id_p INT)
4 RETURNS DECIMAL(5,2)
5 DETERMINISTIC
6 BEGIN
7     DECLARE v_budget DECIMAL(10,2);
8     DECLARE v_depenses DECIMAL(10,2);
9     DECLARE v_taux DECIMAL(5,2);
10
11    SELECT budget, depenses INTO v_budget, v_depenses
12    FROM projets WHERE id_projet = id_p;
13
14    IF v_budget > 0 THEN
15        SET v_taux = (v_depenses / v_budget) * 100;
16    END IF;
17
18    RETURN v_taux;
19 END //
20
21 DELIMITER ;
22

```

Test :

```

1 SELECT nom, calcul_taux(id_projet) AS taux_calcule
2 FROM projets;

```

Étape 7 :

Nettoyage automatique :

```

1 DELIMITER //
2
3 CREATE PROCEDURE nettoyer_alertes()
4 BEGIN
5     DELETE FROM alertes
6     WHERE date_alerte < DATE_SUB(NOW(), INTERVAL 30 DAY);
7 END //
8
9 DELIMITER ;

```

Créer une vue globale :

```

1 CREATE VIEW vue_projets_alertes AS
2 SELECT p.nom, p.budget, p.depenses, p.taux_utilisation, p.statut,
3 a.message, a.date_alerte
4 FROM projets p
5 LEFT JOIN alertes a ON p.id_projet = a.id_projet;

```

Affichage global :

```

1 SELECT * FROM vue_projets_alertes;

```

3) Des captures d'écran prouvant l'exécution des requêtes :

Étape 1 – Création de la base et des tables

-- Crédation de la base

-- Table principale

```
✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0007 seconde(s)).  
  
CREATE TABLE projets ( id_projet INT AUTO_INCREMENT PRIMARY KEY, nom VARCHAR(100), budget DECIMAL(10,2), depenses DECIMAL(10,2), taux_utilisation DECIMAL(5,2), statut VARCHAR(20) );  
  
[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]  
  
✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0007 seconde(s)).  
  
-- Table secondaire : alertes CREATE TABLE alertes ( id_alerte INT AUTO_INCREMENT PRIMARY KEY, id_projet INT, message VARCHAR(255), date_alerte DATETIME );  
  
[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]  
  
✓ MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0008 seconde(s)).  
  
-- Table historique : pour tracer les calculs CREATE TABLE historique ( id INT AUTO_INCREMENT PRIMARY KEY, id_projet INT, taux DECIMAL(5,2), date_calcul DATETIME );  
  
[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]
```

Étape 2 – Insertion de données de test :

```
✓ 3 lignes insérées.  
Identifiant de la ligne insérée : 3 (traitement en 0,0010 seconde(s)).  
  
INSERT INTO projets (nom, budget, depenses) VALUES ('Projet Alpha', 10000, 8500), ('Projet Beta', 5000, 6000), ('Projet Gamma', 8000, 2000);  
  
[ Éditer en ligne ] [ Éditer ] [ Créer le code source PHP ]  
  
✓ Affichage des lignes 0 - 2 (total de 3, traitement en 0,0008 seconde(s)).  
  
SELECT * FROM projets;  
  
□ Tout afficher | Nombre de lignes : 25 ▾ Filtrer les lignes: Chercher dans cette table Trier par clé : Aucun(e) ▾  
  
Options supplémentaires  
  


|                              | id_projet | nom          | budget   | depenses | taux_utilisation | statut |
|------------------------------|-----------|--------------|----------|----------|------------------|--------|
| □  Éditer  Copier  Supprimer | 1         | Projet Alpha | 10000.00 | 8500.00  | NULL             | NULL   |
| □  Éditer  Copier  Supprimer | 2         | Projet Beta  | 5000.00  | 6000.00  | NULL             | NULL   |
| □  Éditer  Copier  Supprimer | 3         | Projet Gamma | 8000.00  | 2000.00  | NULL             | NULL   |

  


↑ □ Tout cocher Avec la sélection :  Éditer  Copier  Supprimer  Exporter

  
□ Tout afficher | Nombre de lignes : 25 ▾ Filtrer les lignes: Chercher dans cette table Trier par clé : Aucun(e) ▾
```

Étape 3 – Crédation de la procédure principale avec curseur :

```
CREATE PROCEDURE calculer_taux_utilisation(IN taux_seuil DECIMAL(5,2)) BEGIN -- Déclaration des variables DECLARE v_id INT; DECLARE v_budget DECIMAL(10,2); DECLARE v_depenses DECIMAL(10,2); DECLARE v_taux DECIMAL(5,2); DECLARE fin BOOLEAN DEFAULT FALSE; -- Déclaration du curseur DECLARE cur_projets CURSOR FOR SELECT id_projet, budget, depenses FROM projets; -- Gestion de la fin du curseur DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin = TRUE; -- Ouverture du curseur OPEN cur_projets; -- Boucle de lecture boucle_projets: Déclaration du curseur DECLARE cur_projets CURSOR FOR SELECT id_projet, budget, depenses FROM projets; -- Gestion de la fin du curseur DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin = TRUE; -- Ouverture du curseur OPEN cur_projets; -- Boucle de lecture boucle_projets: LOOP FETCH cur_projets INTO v_id, v_budget, v_depenses; -- Condition de fin IF fin THEN LEAVE boucle_projets; END IF; -- Calcul du taux d'utilisation SET v_taux = (v_depenses / v_budget) * 100; -- Mise à jour selon le taux IF v_taux > taux_seuil THEN UPDATE projets SET tau [...]
```

Étape 4 – Exécution et tests :

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0304 seconde(s).)

```
CALL calculer_taux_utilisation(90);
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0007 seconde(s).)

```
SELECT * FROM projets;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

Tout afficher | Nombre de lignes : 25 | Filtrer les lignes : Chercher dans cette table | Trier par clé : Aucun(e) |

Options supplémentaires

	id_projet	nom	budget	depenses	taux_utilisation	statut
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	1	Projet Alpha	10000.00	8500.00	85.00	Normal
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	2	Projet Beta	5000.00	6000.00	120.00	Dépassement
<input type="checkbox"/> Éditer <input type="checkbox"/> Copier <input type="checkbox"/> Supprimer	3	Projet Gamma	8000.00	2000.00	25.00	Normal

Étape 5 – Comprendre le rôle du curseur:

Élément	Rôle	Exemple SQL
CURSOR	Permet de lire chaque ligne du résultat d'une requête une par une	DECLARE cur_projets CURSOR FOR SELECT id_projet, budget, depenses FROM projets;
FETCH	Récupère les valeurs de la ligne courante dans des variables	FETCH cur_projets INTO v_id, v_budget, v_depenses;
HANDLER	Gère la fin du curseur pour éviter les erreurs quand il n'y a plus de lignes	DECLARE CONTINUE HANDLER FOR NOT FOUND SET fin = TRUE;
LOOP	Exécute les opérations pour chaque ligne	boucle_projets: LOOP ... END LOOP;
LEAVE	Quitte la boucle proprement lorsque toutes les lignes ont été lues	IF fin THEN LEAVE boucle_projets; END IF;

Étape 6 – Créer une fonction réutilisable :

Afficher la zone SQL

MySQL a retourné un résultat vide (c'est à dire aucune ligne). (traitement en 0,0237 seconde(s).)

```
CREATE FUNCTION calcul_taux(id_p INT) RETURNS DECIMAL(5,2) DETERMINISTIC BEGIN DECLARE v_budget DECIMAL(10,2); DECLARE v_depenses DECIMAL(10,2); DECLARE v_taux DECIMAL(5,2); SELECT budget, depenses INTO v_budget, v_depenses FROM projets WHERE id_projet = id_p; IF v_budget > 0 THEN SET v_taux = (v_depenses / v_budget) * 100; END IF; RETURN v_taux; END;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Test :

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0028 seconde(s).)

```
SELECT nom, calcul_taux(id_projet) AS taux_calcule FROM projets;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table | Trier par clé : Aucun(e) |

Options supplémentaires

	nom	taux_calcule
<input type="checkbox"/>	Éditer Copier Supprimer Projet Alpha	85.00
<input type="checkbox"/>	Éditer Copier Supprimer Projet Beta	120.00
<input type="checkbox"/>	Éditer Copier Supprimer Projet Gamma	25.00

↑ Tout cocher Avec la sélection : Éditer Copier Supprimer Exporter

Étape 7 :

Nettoyage automatique :

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0028 seconde(s).)

```
CREATE PROCEDURE nettoyer_alertes() BEGIN DELETE FROM alertes WHERE date_alerte < DATE_SUB(NOW(), INTERVAL 30 DAY); END;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Créer une vue globale :

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0011 seconde(s).)

```
CREATE VIEW vue_projets_alertes AS SELECT p.nom, p.budget, p.depenses, p.taux_utilisation, p.statut, a.message, a.date_alerte FROM projets p LEFT JOIN alertes a ON p.id_projet = a.id_projet;
```

[Éditer en ligne] [Éditer] [Créer le code source PHP]

Affichage global :

Affichage des lignes 0 - 2 (total de 3, traitement en 0,0017 seconde(s).)

```
SELECT * FROM vue_projets_alertes;
```

Profilage [Éditer en ligne] [Éditer] [Expliquer SQL] [Créer le code source PHP] [Actualiser]

Tout afficher | Nombre de lignes : 25 | Filtrer les lignes: Chercher dans cette table |

Options supplémentaires

	nom	budget	depenses	taux_utilisation	statut	message	date_alerte
<input type="checkbox"/>	Éditer Copier Supprimer Projet Beta	5000.00	6000.00	120.00	Depassement	Depassement du seuil 90.00 % → 120.00 %	2025-10-30 20:52:25
<input type="checkbox"/>	Éditer Copier Supprimer Projet Alpha	10000.00	8500.00	85.00	Normal	NULL	NULL
<input type="checkbox"/>	Éditer Copier Supprimer Projet Gamma	8000.00	2000.00	25.00	Normal	NULL	NULL

↑ Tout cocher Avec la sélection : Éditer Copier Supprimer Exporter

4) Une analyse des résultats:

- Projet Alpha** : 85% → statut Normal, aucune alerte.
- Projet Beta** : 120% → dépassement du seuil 90%, statut Depassement, alerte créée.
- Projet Gamma** : 25% → statut Normal, aucune alerte.

