Leçon 4 : Les procédures stockées

AYIKPA KACOUTCHY JEAN : Enseignant - Chercheur

Table des matières

1- Généralité des procédures stockées

I

Définition : 1.1- Définition

Une procédure stockée est *une série d'instructions SQL* désignée *par un nom*.

Lorsque l'on crée *une procédure stockée*,on l'enregistre dans la base de données que l'on utilise, au même titre qu'une table.

Une fois la procédure créée, il est possible *d'appeler celle-ci, par son nom. Les instructions de la procédure sont alors exécutées.*

Contrairement aux requêtes préparées, qui ne sont gardées en mémoire que pour la session courante, les procédures stockées sont, comme leur nom l'indique, stockées de manière durable, et font bien partie intégrante de la base de données dans laquelle elles sont enregistrées.

1.2- Intérêt des procédures stockées

- *Les procédures stockées permettent de réduire les allers-retours entre le client et le serveur MySQL*. En effet, si l'on englobe en une seule procédure un processus demandant l'exécution de plusieurs requêtes, le client ne communique qu'une seule fois avec le serveur (pour demander l'exécution de la procédure) pour exécuter la totalité du traitement. Cela permet donc un certain gain en performance.
- *Elles permettent également de sécuriser une base de données*. Par exemple, il est possible de restreindre les droits des utilisateurs de façon à ce qu'ils puissent uniquement exécuter des procédures. Finis les DELETE dangereux ou les UPDATE inconsidérés. Chaque requête exécutée par les utilisateurs est créée et contrôlée par l'administrateur de la base de données par l'intermédiaire des procédures stockées.
- *Elles permettent ensuite de s'assurer qu'un traitement est toujours exécuté de la même manière, quelle que soit l'application/le client qui le lance*. Il arrive par exemple qu'une même base de données soit exploitée par plusieurs applications, lesquelles peuvent être écrites avec différents langages. Si on laisse chaque application avoir son propre code pour un même traitement, il est possible que des différences apparaissent (distraction, mauvaise communication, erreur ou autre). Par contre, si chaque application appelle la même procédure stockée, ce risque disparaît.

Application 1 :

Exercice

[solution n°1 p.9*]*

Un procédure stockée est :

- une série d'instructions SQL désignée par un nom.
- une exploitation visuelle des données
- une accélération de la mise à jour des données

2- Traitement des procédures stockées

III

2.1- Création d'une procédure stockée

Syntaxe :

Cas 1 : Procédure avec une seule requête

CREATE PROCEDURE nom_procedure (Liste des paramètres)

corps de la procédure;

Cas 2 : Procédure avec un bloc d'instructions

DELIMITER |

CREATE PROCEDURE nom_procedure (Liste des paramètres)

BEGIN

Bloc d'instructions ;

END|

CREATE PROCEDURE :il s'agit de la commande à exécuter pour créer une procédure.

Liste des paramètres : après le nom de la procédure viennent des parenthèses. Celles-ci sont obligatoires ! À l'intérieur de ces parenthèses, on définit les éventuels paramètres de la procédure. Ces paramètres sont des variables qui pourront être utilisées par la procédure.

Corps de la procédure : c'est là que l'on met le contenu de la procédure, ce qui va être exécuté lorsqu'on lance la procédure. Cela peut être soit une seule requête, soit un bloc d'instructions.

DELIMITER | : Un délimiteur est tout simplement (par défaut), le caractère *;*. C'est-à-dire le caractère qui permet de délimiter les instructions, mais pour une procédure avec un bloc d'instructions il faut changer le délimiteur de la fin de procédure pour éviter une erreur. Dans notre cas nous utilisons le délimiteur |.

Exemple :

Soit la table suivant :

Client(idcli,nomcli,prencli,sexecli,datenaiscli)

Créer une procédure qui affiche la liste des clients

```
1)
CREATE PROCEDURE Liste_clients()
SELECT idcli,nomcli,prencli,sexecli,datenaiscli FROM Client;

2)
DELIMITER |
CREATE PROCEDURE Liste_clients()
BEGIN
SELECT idcli,nomcli,prencli,sexecli,datenaiscli FROM Client;
END |
```

2.2- Appel d'une procédure stockée

Pour appeler une procédure stockée, il faut utiliser le mot-clé *CALL,* suivi du nom de la procédure appelée.

Exemple :

Soit la table suivant :

Client(idcli,nomcli,prencli,sexecli,datenaiscli)

Créer une procédure qui affiche la liste des client, puis appelé la procédure

```
1)
CREATE PROCEDURE Liste_clients()
SELECT idcli,nomcli,prencli,sexecli,datenaiscli FROM Client;
//************************** Appel de la procédure
CALL Liste_clients();

2)
DELIMITER |
CREATE PROCEDURE Liste_clients()
BEGIN
SELECT idcli,nomcli,prencli,sexecli,datenaiscli FROM Client;
END |
//************************** Appel de la procédure
CALL Liste_clients()| //Utiliser le délimiteur créé

```

2.3- Paramètre d'une procédure stockée

Un paramètre peut être de trois sens différents : *entrant (IN), sortant (OUT), ou les deux (INOUT).*

- *IN* : c'est un paramètre "entrant". C'est-à-dire qu'il s'agit d'un paramètre dont la valeur est fournie à la procédure stockée. Cette valeur sera utilisée pendant la procédure (pour un calcul ou une sélection, par exemple).
- *OUT :* il s'agit d'un paramètre "sortant", dont la valeur sera établie au cours de la procédure et qui pourra ensuite être utilisé en dehors de cette procédure.
- *INOUT* : un tel paramètre sera utilisé pendant la procédure, verra éventuellement sa valeur modifiée par celle-ci, et sera ensuite utilisable en dehors.

Lorsque vous allez déclarer un paramètre au sein de votre procédure stockée il vous faudra le faire en 3 étapes :

- *Son sens* (entrant, sortant ou les deux, si aucun sens n'est donné IN sera le sens par défaut)
- *Son nom* (obligatoire afin de pouvoir appeler votre paramètre à la suite d'une instruction WHERE par exemple)
- *Son type* (INT, VARCHAR (10), etc.)

Exemple : Donner la liste de tous les clients qui on le nom BLON

```
1
2 DELIMITER |
3 CREATE PROCEDURE Liste_clients(IN nom varchar(20))
4 BEGIN
5 SELECT idcli,nomcli,prencli,sexecli,datenaiscli
6 FROM Client
7 where nomcli = nom;
8 END |
9 //************************* Appel de la procédure
10 Cas 1 :
11 CALL Liste_clients('BLON')|
12
13 Cas 2 :
14 SET @nom = 'BLON'
15 CALL Liste_clients(@nom)|
16
17
18
```

2.4- Suppression d'une procédure stockée

Pour supprimer une procédure, on utilise DROP.

DROP PROCEDURE Nom_procédure ;

Exemple :

Soit la table suivant :

Client(idcli,nomcli,prencli,sexecli,datenaiscli)

Supprimer la procédure liste_client

```
1 DROP PROCEDURE liste_client;
```

Application 2 :

IV

Exercice

[solution n°2 p.9]

Énoncé :

Soit la table suivante :

LIVRE(NUMLIV,NOMAUT ,TITRELIV,GENRELIV,PRIXLIV)

- Créer une procédure stockée qui permet de donner la liste des livres supérieurs à 2000. La valeur de la condition doit être passer en paramètre dans une variable mont
- Appel la procédure par la suite avec son paramètre.

NB : Respecter l'ordre de création des propriétés

Solution :

Liste_livre( )

NUMLIV,NOMAUT ,TITRELIV,GENRELIV,PRIXLIV

FROM LIVRE

=2000

Liste_livre( )

Solutions des exercices

> **Solution** n°1

Exercice p. 4

Un procédure stockée est :

- ◉ une série d'instructions SQL désignée par un nom.
- ○ une exploitation visuelle des données
- ○ une accélération de la mise à jour des données

> **Solution** n°2

Exercice p. 8

Énoncé :

Soit la table suivante :

LIVRE(NUMLIV,NOMAUT ,TITRELIV,GENRELIV,PRIXLIV)

- Créer une procédure stockée qui permet de donner la liste des livres supérieurs à 2000. La valeur de la condition doit être passer en paramètre dans une variable mont
- Appel la procédure par la suite avec son paramètre.

NB : Respecter l'ordre de création des propriétés

Solution :

```
DELIMITER |
CREATE PROCEDURE Liste_livre(IN mont int)
BEGIN
SELECT NUMLIV,NOMAUT ,TITRELIV,GENRELIV,PRIXLIV
FROM LIVRE
where PRIXLIV=mont;
END |
SET @mont=2000
CALL Liste_livre(@mont)|
```