

HUMAN **booster**



VOTRE SOLUTION COMPETENCE

Bases de données relationnelles

Conception • Utilisation

Administration • Optimisation

Procédures stockées

(Stored Procedures ou « Procs Stock »)



Qu'est-ce que c'est ?

- Les procédures stockées sont des fonctions « coté serveur de base de données »
- Elles sont développées « sur mesure » pour accomplir un traitement
- Elles font pleinement partie de la structure de la base de données
- Elles peuvent être créées, modifiées et supprimées à l'aide de **CREATE**, **ALTER** et **DROP**
- Elles peuvent être exécutées :

→ Exemple d'appel : **CALL** NomProcédure;
ou **EXECUTE** NomProcédure;

À quoi ça sert ?

- **Le but d'une procédure stockée est d'exécuter un traitement personnalisé :**
- **Il peut s'agir d'un traitement :**
 - **de lecture (exemple : export statistiques complexe)**
 - **d'insertion (exemple : facturation de fin de mois)**
 - **de mise à jour (exemple : augmentation tarifaire sélective)**
 - **de toute autre nature (exemple : envois de mail, sauvegardes, purges, etc.)**
- **Les procédures stockées permettent aussi de garder en mémoire des requêtes « support » utilisés a des fins de monitoring**

À quoi ça ressemble ?



```
DELIMITER |
```

```
CREATE PROCEDURE NomProcedure()
```

```
BEGIN
```

```
-- (traitement);
```

```
END |
```

Puisque la partie traitement de la procédure utilise le point-virgule, il est nécessaire de changer le délimiteur pour marquer la fin de la procédure

```
DELIMITER |
```

```
CREATE PROCEDURE NomProcedure(INT parametre)
```

```
BEGIN
```

```
-- (traitement utilisant le paramètre);
```

```
END |
```

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
INSERT INTO Entreprise(nom)
VALUES ('Un nom...');
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



WHILE

DO -- On boucle

```
INSERT INTO Entreprise(nom)
VALUES ('Un nom...');
```

END WHILE;

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @i = 0;
```

```
WHILE @i < 40 DO -- On boucle 40 fois
```

```
INSERT INTO Entreprise(nom)  
VALUES ('Un nom...');
```

```
SET @i = @i + 1;  
END WHILE;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40;
```

```
SET @i = 0;
```

```
WHILE @i < @maxi DO -- On boucle 40 fois
```

```
INSERT INTO Entreprise(nom)
```

```
VALUES ('Un nom...');
```

```
SET @i = @i + 1;
```

```
END WHILE;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN

```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
```

```
SET @i = 0;
```

```
WHILE @i < @maxi DO -- On boucle autant de fois que demandé
```

```
INSERT INTO Entreprise(nom)
```

```
VALUES ('Un nom...');
```

```
SET @i = @i + 1;
```

```
END WHILE;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
```

```
SET @i = 0;
```

```
WHILE @i < @maxi DO -- On boucle autant de fois que demandé
```

```
    SET @nomTemp = -- Comment faire une chaine aléatoire ?
```

```
INSERT INTO Entreprise(nom)
```

```
VALUES (@nomTemp);
```

```
SET @i = @i + 1;
```

```
END WHILE;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 99999999999999), 20, 36); -- Chaine aléatoire

    INSERT INTO Entreprise(nom)
    VALUES (@nomTemp);

    SET @i = @i + 1;
END WHILE;
```

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 99999999999999), 20, 36); -- Chaine aléatoire
```

```
INSERT INTO Entreprise(nom)
VALUES (@nomTemp);
```

```
SET @i = @i + 1;
END WHILE;
```

```
SELECT CONCAT(@i, ' entreprises ajoutées') AS Resultat;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); -- Chaine aléatoire
    SET @nbSalariesTemp = 1 + RAND() * 249; -- Nombre aléatoire entre 1 et 250
```

```
INSERT INTO Entreprise(nom, nbSalaries)
VALUES (@nomTemp, @nbSalariesTemp);
```

```
SET @i = @i + 1;
END WHILE;
```

```
SELECT CONCAT(@i, ' entreprises ajoutées') AS Resultat;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); -- Chaine aléatoire
    SET @nbSalariesTemp = 1 + RAND() * 249; -- Nombre aléatoire entre 1 et 250
    SET @estPubliqueTemp = IF(RAND() < 0.10, 1, 0); -- Est publique dans 10 % des cas

    INSERT INTO Entreprise(nom, nbSalaries, estPublique)
    VALUES (@nomTemp, @nbSalariesTemp, @estPubliqueTemp);

    SET @i = @i + 1;
END WHILE;

SELECT CONCAT(@i, ' entreprises ajoutées') AS Resultat;

END
```


Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); -- Chaine aléatoire
    SET @nbSalariesTemp = 1 + RAND() * 249; -- Nombre aléatoire entre 1 et 250
    SET @estPubliqueTemp = IF(RAND() < 0.10, 1, 0); -- Est publique dans 10 % des cas
    SET @dateFondaTemp = CURDATE() - INTERVAL (RAND() * 300) DAY; -- Date aléatoire

    INSERT INTO Entreprise(nom, nbSalaries, estPublique, dateFondation)
    VALUES (@nomTemp, @nbSalariesTemp, @estPubliqueTemp, @dateFondaTemp);

    SET @i = @i + 1;
END WHILE;

SELECT CONCAT(@i, ' entreprises ajoutées') AS Resultat;
```

END

Pas à pas : insertion de 40 à 60 données aléatoires dans une table « Entreprise »

BEGIN



```
SET @maxi = 40 + RAND() * 20; -- Nombre aléatoire entre 40 et 60 (60 - 40 = 20)
SET @i = 0;

WHILE @i < @maxi DO -- On boucle autant de fois que demandé
    SET @nomTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); -- Chaine aléatoire
    SET @nbSalariesTemp = 1 + RAND() * 249; -- Nombre aléatoire entre 1 et 250
    SET @estPubliqueTemp = IF(RAND() < 0.10, 1, 0); -- Est publique dans 10 % des cas
    SET @dateFondaTemp = CURDATE() - INTERVAL (RAND() * 300) DAY; -- Date aléatoire
    SET @idDptTemp = (SELECT idDepartement FROM Departement
                      ORDER BY RAND() LIMIT 1); -- ID de département aléatoire

    INSERT INTO Entreprise(nom, nbSalaries, estPublique, dateFondation, idDepartement)
    VALUES (@nomTemp, @nbSalariesTemp, @estPubliqueTemp, @dateFondaTemp, @idDptTemp);

    SET @i = @i + 1;
END WHILE;

SELECT CONCAT(@i, ' entreprises ajoutées') AS Resultat;
```

Comment les créer dans PhpMyAdmin ?

1. Cliquez sur votre base de données

2. En haut à droite, cliquez sur "Plus", puis sur "Procédures stockées"

3. Cliquez sur "Ajouter une procédure"

4. Nommez votre procédure

5. Supprimez le paramètre

6. Copiez-collez le code-source de la procédure

7. Cliquez ici

8. La procédure est créée

9. Appelez-la en cliquant ici une fois

10. La procédure a fonctionné

Détails

Nom de la procédure: AddQuelqueChose

Type: PROCEDURE

Paramètres: Direction: IN, Nom: , Type: INT, Taille/Valeurs*: , Options: Supprimer

Ajouter un paramètre

Définition

```
1 BEGIN
2 DECLARE numeroSiretTemp VARCHAR(50);
3 DECLARE raisonSocialeTemp VARCHAR(50);
4 DECLARE anneeTemp INT;
5 DECLARE estAutoentrepreneurTemp BOOLEAN;
6 DECLARE idMetierTemp INT;
7 DECLARE maxi INT;
8 DECLARE i INT;
9 SET @maxi = 100 + RAND() * 50; -- Nombre aléatoire compris entre 100 et 150
10 SET @i = 0;
11 BEGIN
12 WHILE @i < @maxi DO
13 SET @numeroSiretTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); --
14 Le numéro de Siret doit être une suite de caractères aléatoire
15 SET @raisonSocialeTemp = CONV(FLOOR(RAND() * 999999999999999), 20, 36); --
```

Est déterministe ☐

Créateur

Type de sécurité DEFINER

Accès aux NO SQL

Exécuter **Fermer**

✓ La procédure 'AddQuelqueChose' a été créée.

CREATE PROCEDURE `AddQuelqueChose`() NOT DETERMINISTIC NO SQL SQL SECURITY DEFINER
DECLARE anneeTemp INT; DECLARE estAutoentrepreneurTemp BOOLEAN; DECLARE idMetierTemp INT; DE
@i = 0; BEGIN WHILE @i < @maxi DO SET @numeroSiretTemp = CONV(FLOOR(RAND() * 99999
CONV(FLOOR(RAND() * 999999999999999), 20, 36); -- La raison sociale doit être une suite de carac
comprise entre 1900 et 2019 SET @estAutoentrepreneurTemp = IF(RAND() < 0.35, 1, 0); -- Le spéc

Procédures stockées

AddQuelqueChose Éditer Exécuter Exporter Supprimer PROCEDURE

✓ La requête SQL a été exécutée avec succès.
1 ligne a été affectée par la dernière instruction de la procédure.

CALL `AddQuelqueChose`();

Résultats de l'exécution de la procédure 'AddQuelqueChose'

Résultat

123 ajoutés

Avantages

- Les procédures stockées permettent de limiter les échanges entre le client et le serveur (seul le nom de la procédure transite : pas le texte complet de la requête)
- Elles évitent au serveur d'interpréter la requête à chaque fois car elle est précompilée au moment de sa création
- Elles sécurisent la base de données en limitant l'accès direct aux tables
- Elles permettent d'uniformiser les traitements (ainsi, si plusieurs instance d'applis utilisent la même BDD, elles peuvent utiliser la même procédure stockée afin d'effectuer un calcul d'un montant TVA correctement arrondi, par exemple)

Limites

- En déportant le traitement sur le serveur de base de données, elles en augmentent la charge de travail
- Elles font s'éloigner du rôle de « stockage » des données et empiètent sur le rôle de « logique » et des « traitements métiers »

HUMAN **booster**

●● VOTRE SOLUTION COMPETENCE

Tél. 04 73 24 93 11 – contact@humanbooster.com
www.humanbooster.com

●● Clermont-Ferrand ●● Montpellier ●● Lyon