Angelo Nichele SAE 3.01

Marwan Al-Masri

Maxence Krill

Guychel Babella

Livrable Triggers et Procédures DISGUISE'HUB



Table des matières

1 T	riggers	3
1.1	Code	3
1.2	Résultats	5
1.3	Tests	6
2 P	rocédures	8
2.1	Code	8
2.2	Résultats	9

1 Triggers

1.1 Code

```
CREATE TRIGGER check_qteStock
BEFORE INSERT ON Commander
FOR EACH ROW
BEGIN
   DECLARE v qteProduit DECIMAL(4);
   SELECT gteProduit INTO v gteProduit FROM Produit WHERE refProduit = NEW.refPro-
duit;
    IF NEW.qteCommandee <= 0 THEN</pre>
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'La quantité commandée doit être supérieure à 0.';
   ELSE
        IF NEW.qteCommandee > v_qteProduit THEN
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'La quantité commandée ne
peut pas être supérieure à la quantité en stock.';
        END IF;
    END IF;
END;
CREATE TRIGGER maj qteProduit
AFTER INSERT ON Commander
FOR EACH ROW
BEGIN
    DECLARE v_qteActuelle DECIMAL(4);
   SELECT qteProduit INTO v_qteActuelle
    FROM Produit
   WHERE refProduit = NEW.refProduit;
   UPDATE Produit
   SET qteProduit = v qteActuelle - NEW.qteCommandee
   WHERE refProduit = NEW.refProduit;
END;
```

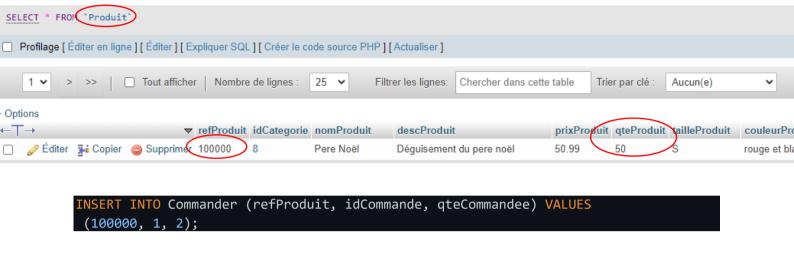
```
CREATE TRIGGER check dateExpCB
BEFORE INSERT ON Cartebleue
FOR EACH ROW
BEGIN
    IF NEW.dateExpCB <> DATE FORMAT(NEW.dateExpCB, '%Y-%m-%d') THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE TEXT = 'La date d''expiration de la carte bleue doit être au
format YYYY/MM/DD.';
   END IF;
END;
//
CREATE TRIGGER check_dateNaissanceClient
BEFORE INSERT ON Client
FOR EACH ROW
BEGIN
   IF NEW.dateNaissanceClient <> DATE FORMAT(NEW.dateNaissanceClient, '%Y-%m-%d')
THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE TEXT = 'La date de naissance du client doit être au format
YYYY/MM/DD.';
    END IF;
END;
CREATE TRIGGER check tailleProduit
BEFORE INSERT ON Produit
FOR EACH ROW
BEGIN
    IF NEW.tailleProduit NOT IN ('XS', 'S', 'M', 'L', 'XL', 'XXL') THEN
        SIGNAL SQLSTATE '45000'
        SET MESSAGE_TEXT = 'La taille du produit doit être XS, S, M, L, XL ou
XXL.';
   END IF;
END;
```

1.2 Résultats



1.3 Tests

Avant une insertion dans la table Commander :



Après une insertion valide dans la table Commander :



Après une insertion non valide dans la table Commander :

```
INSERT INTO Commander (refProduit, idCommande, qteCommandee) VALUES
  (100000, 1, 0);

INSERT INTO Commander (refProduit, idCommande, qteCommandee) VALUES
      (100000, 1, 0);

MySQL a répondu:  (3)

#1644 - La quantité commandée doit être supérieure à 0.
```

```
INSERT INTO Commander (refProduit, idCommande, qteCommandee) VALUES
  (100000, 1, 51);

INSERT INTO Commander (refProduit, idCommande, qteCommandee) VALUES
        (100000, 1, 51);

MySQL a répondu: ②

#1644 - La quantité commandée ne peut pas être supérieure à la quantité en stock.
```

Après une insertion non valide dans la table Produit :

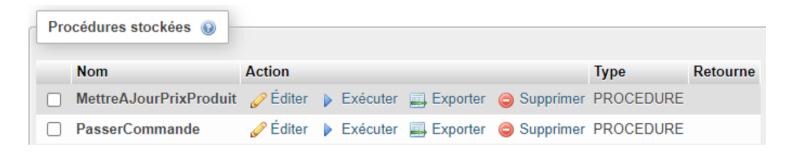
2 Procédures

2.1 Code

```
CREATE PROCEDURE PasserCommande(
    IN p idClient INT,
    IN p refProduit INT,
    IN p_qteCommandee INT,
    IN p idPaiement INT,
    IN p adrLivraison TEXT,
    IN p_codePostalLivraison DECIMAL(5),
    IN p_fraisLivraison DECIMAL(6,2)
BEGIN
    DECLARE v_qteProduit DECIMAL(4);
   SELECT qteProduit INTO v_qteProduit
    FROM Produit
   WHERE refProduit = p_refProduit;
    IF v_qteProduit IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le produit spécifié n''existe
pas.';
   ELSE
        IF p_qteCommandee <= 0 THEN</pre>
            SIGNAL SQLSTATE '45000' SET MESSAGE TEXT = 'La quantité commandée doit
être supérieure à 0.';
        ELSE
            IF p_qteCommandee > v_qteProduit THEN
                SIGNAL SQLSTATE '45000' SET MESSAGE TEXT = 'La quantité commandée
ne peut pas être supérieure à la quantité en stock.';
                INSERT INTO Commande (idClient, idPaiement, dateCommande, frai-
sLivraison, adrLivraison, codePostalLivraison, statutCommande)
                VALUES (p_idClient, p_idPaiement, NOW(), p_fraisLivraison,
p_adrLivraison, p_codePostalLivraison, 'En cours de préparation');
                SET @lastCommandeId = LAST_INSERT_ID();
                INSERT INTO Commander (refProduit, idCommande, qteCommandee)
                VALUES (p_refProduit, @lastCommandeId, p_qteCommandee);
            END IF;
        END IF;
   END IF;
END;
```

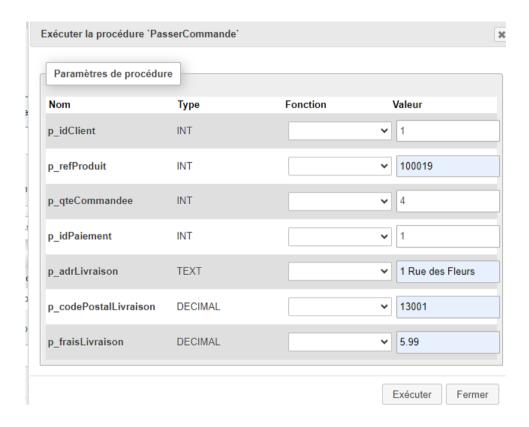
```
CREATE PROCEDURE MettreAJourPrixProduit(
    IN p_refProduit INT,
    IN p nouveauPrix DECIMAL(6,2)
BEGIN
    DECLARE v_qteProduit DECIMAL(4);
    SELECT qteProduit INTO v_qteProduit
    FROM Produit
    WHERE refProduit = p refProduit;
    IF v qteProduit IS NULL THEN
        SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le produit spécifié n''existe
pas.';
    ELSE
        IF p_nouveauPrix <= 0 THEN</pre>
            SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'Le nouveau prix doit être
supérieur à 0.';
        ELSE
            UPDATE Produit
            SET prixProduit = p nouveauPrix
            WHERE refProduit = p_refProduit;
        END IF;
    END IF;
END;
```

2.2 Résultats



2.3 Tests

Exécution de la procédure « PasserCommande » :



Mise à jour du stock dans la table Produit (Trigger maj_qteProduit) :

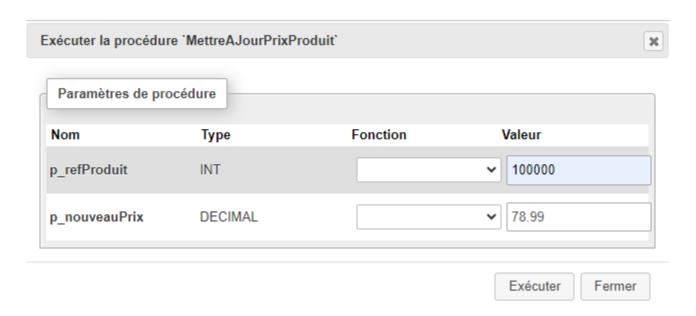
refProduit	idCategorie	nomProduit	descProduit	prixProduit	qteProduit	tailleProduit	couleurProduit
100019	12	Batman	Déguisement de Batman	55.99	96	XS	noir

Insertion dans la table Commande et Commander :



refProduit	id Commande	qteCommandee
100019	106	4

Exécution de la procédure « MiseAJourPrixProduit » :



UPDATE dans la table Produit :

refProduit	id Categorie	nomProduit	descProduit	prix Produit	qte Produit	tailleProduit	couleurProduit
100000	8	Pere Noël	Déguisement du pere noël	50.99	50	S	rouge et blanc

refProdui	t idCategorie	nomProduit	descProduit	prixProduit	qteProduit	tailleProduit	couleurProduit
100000	8	Pere Noël	Déguisement du pere noël	78.99	50	S	rouge et blanc