Lập trình Website tĩnh

Views – Stored Procedures - Trigger

Ths. Vũ Duy Khương

Nội dung

1. View trong MySQL
2. Stored Procedure trong MySQL
3. Trigger
4. Bài thực hành

© Copyright 2021 Ths. Vũ Duy Khương

View trong MySQL

- View (Khung Hình) có tác dụng giống như tạo một Table ảo với các Fields và Records mà ta có thể tự định nghĩa và khác hoàn tòa với Table gốc.
- View có đầy đủ các tính chất của một Table ➔ Có thể truy vấn, delete và tạo mới View thông qua một câu lệnh đơn giản.
- Nếu table đổi dữ liệu thì mặc nhiên View cũng sẽ thay đổi theo. ➔ không mất nhiều công sức để cập nhật lại dữ liệu

© Copyright 2021 Ths. Vũ Duy Khương

View trong MySQL

- Ví dụ:

Giả sử 1 trang cần hiển thị 10 tin mới nhất, như vậy ta sẽ truy vấn lấy 10 tin và sắp xếp giảm dần theo ID.
Nhưng trong SQL nó sẽ duyệt toàn bộ bảng rồi mới trả về kết quả và điều này làm cho truy vấn trở nên chậm chạm.

Để giải quyết nó thì ta sẽ tạo một View gồm 10 tin mới nhất và lúc hiển thị ra chỉ cần lấy trong View nên tốc độ sẽ nhanh hơn rất nhiều lần.

© Copyright 2021 Ths. Vũ Duy Khương

Ưu điểm sử dụng View trong MySQL

- Đơn giản hóa các câu truy vấn phức tạp bởi vì một View được tạo ra bởi một câu truy vấn SQL
- Che giấu đi sự phức tạp của mô hình dữ liệu trong hệ thống
- **View** giúp giới hạn dữ liệu cho người sử dụng
- **View** giúp tăng tính bảo mật hơn bởi vì View chi đọc mà không ghi được (Read Only) nên việc hacker tấn công cập nhật dữ liệu là điều không thể.

© Copyright 2021 Ths. Vũ Duy Khương

Ưu điểm sử dụng View trong MySQL

- **View** cho phép tăng hoặc giảm các Fields tùy theo yêu cầu sử dụng bởi vì nó được tạo từ một câu truy vấn SELECT nên bạn có thể JOIN nhiều Table lại với nhau và lưu vào View
- **View** tăng khả năng phát triển lại ứng dụng hoặc tương thích với nhiều ứng dụng chạy chung một CSDL

© Copyright 2021 Ths. Vũ Duy Khương

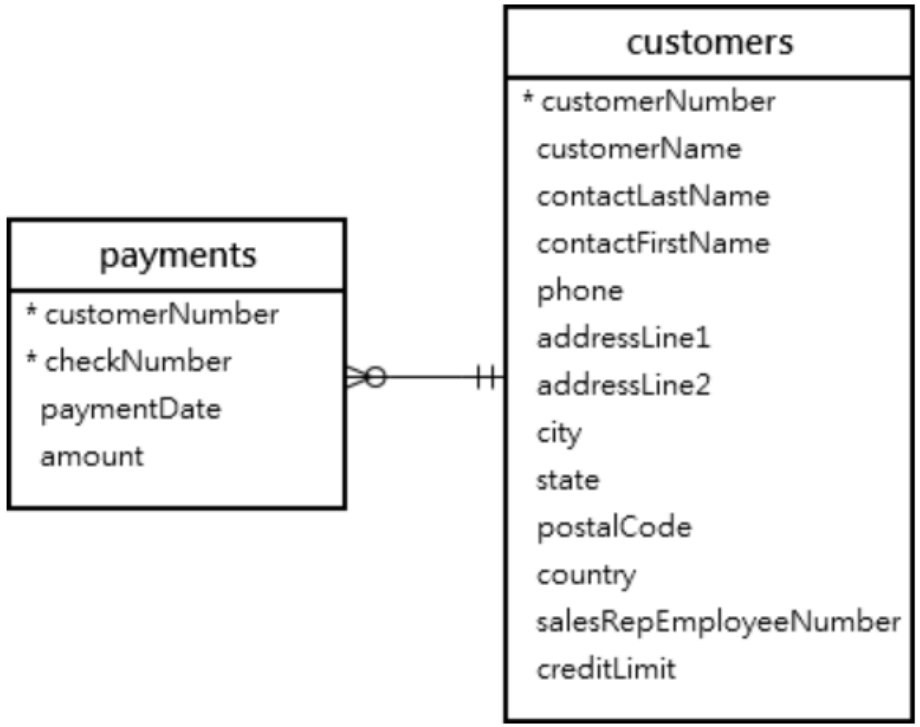
Nhược điểm sử dụng View trong MySQL

- Truy vấn trong **View** có thể sẽ chậm hơn trong table
- Phụ thuộc vào Table gốc, nếu Table gốc thay đổi cấu trúc thì đòi hỏi View cũng phải thiết kế lại cho phù hợp

© Copyright 2021 Ths. Vũ Duy Khương

Đặt vấn đề View

- Bài toán: Giả sử ta có 2 table gồm:
  - **payments**: Lưu lịch sử thanh toán của khách hàng
  - **customers**: Lưu danh sách khách hàng



© Copyright 2021 Ths. Vũ Duy Khương

Đặt vấn đề View

- Câu truy vấn lấy ra tất cả khách hàng và thông tin lịch sử thanh toán của khách hàng

```
SELECT
    customerName,
    checkNumber,
    paymentDate,
    amount
FROM
    customers
INNER JOIN
    payments USING (customerNumber);
```

© Copyright 2021 Ths. Vũ Duy Khương

Đặt vấn đề View

- Kết quả:

customerName	checkNumber	paymentDate	amount
Atelier graphique	HQ336336	2004-10-19	6066.78
Atelier graphique	JM555205	2003-06-05	14571.44
Atelier graphique	OM314933	2004-12-18	1676.14
Signal Gift Stores	BO864823	2004-12-17	14191.12
Signal Gift Stores	HQ55022	2003-06-06	32641.98
Signal Gift Stores	ND748579	2004-08-20	33347.88
Australian Collectors, Co.	GG31455	2003-05-20	45864.03
Australian Collectors, Co.	MA765515	2004-12-15	82261.22
Australian Collectors, Co.	NP603840	2003-05-31	7565.08
Australian Collectors, Co.	NR27552	2004-03-10	44894.74
La Rochelle Gifts	DB933704	2004-11-14	19501.82
La Rochelle Gifts	LN373447	2004-08-08	47924.19

Nhưng 1 ngày nào đó cần lấy lại thông tin này ➔ Viết lại câu truy vấn

© Copyright 2021 Ths. Vũ Duy Khương

Đặt vấn đề View

➔ Sử dụng View trong MySQL

Cú pháp:

```
CREATE VIEW view_name AS
SELECT statement
```

Ví dụ:

```
CREATE VIEW customerPayments
AS
SELECT
    customerName,
    checkNumber,
    paymentDate,
    amount
FROM
    customers
INNER JOIN
    payments USING (customerNumber);
```

© Copyright 2021 Ths. Vũ Duy Khương

Đặt vấn đề View

➔ Sau khi chạy câu truy vấn này thì trên hệ thống database xuất hiện thêm một View có tên là **customerPayments**

➔ Ở những lần sau nếu bạn muốn lấy dữ liệu thì chỉ việc chạy câu SQL này.

SELECT * ***FROM*** *customerPayments;*

Kết luận: Một View sẽ được lưu trữ trên ổ đĩa vật lý nên thực chất nó cũng là một table, vì vậy bạn có thể thực hiện truy vấn với câu Select. Tuy nhiên với lệnh Delete hoặc Update thì không thực hiện trên View được

© Copyright 2021 Ths. Vũ Duy Khương

Tại sao nên sử dụng View

- **Đơn giản hóa truy vấn phức tạp**

  View giúp đơn giản hóa những câu truy vấn phức tạp cho những lần sử dụng sau, bở khi dữ liệu ở bảng chính thay đổi thì trong View cũng sẽ được thay đổi theo nên vấn đề đồng bộ dữ liệu rất chính xác.
- **Làm giảm độ phức tạp tính toán**
- **Phân quyền và bảo mật**

  Không xem được Table, chỉ xem được View. Vì là một table nên ta có thể giới hạn phân quyền cho table một cách dễ dàng

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

➢ Lệnh Create View

Cú pháp:

CREATE [OR REPLACE] VIEW [db_name.]view_name [(column_list)]
AS
  select-statement;

Ví dụ: Table film from DB sakila

```
create view filmView as
        select
                title,
                 sum(length) length_total
          from
                film
          group by film_id
          order by length_total desc;
```

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

➢ Để kiểm tra View đã xuất hiện chưa, ta sử dụng lệnh

SHOW TABLES;

Tables_in_sakila
film_category
film_list
film_text
filmview
inventory

➢ Để xem chi tiết đâu là table và đâu là view thì sử dụng lệnh sau:

SHOW FULL TABLES;

Tables_in_sakila	Table_type
film_list	VIEW
film_text	BASE TABLE
filmview	VIEW
inventory	BASE TABLE
language	BASE TABLE
nicer_but_slow...	VIEW
payment	BASE TABLE
rental	BASE TABLE

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

➢ Ví dụ: Tạo mới một View từ một View khác

Giả sử cần lấy các bộ film có length_total >= 185

```
Create View lengthMax As
        Select
              title
       from
             filmView
       where
              length_total >=185
```

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

- **Lệnh Drop View** : dùng để xóa một View bất kì ra khỏi database trong MySQL.
- Cú pháp:

  DROP VIEW [IF EXISTS] view_name;
- Trong đó:
  - Từ khóa DROP VIEW xóa một view nào đó.
  - Từ khóa IF EXISTS có thể có hoặc không. Nếu có thì sẽ không bị lỗi khi view không tồn tai, ngược lại sẽ bị lỗi.
- Để xóa nhiều view cùng một lúc thì ta sử dụng cú pháp sau:

  DROP VIEW [IF EXISTS] view_name1 [,view_name2]...;

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

➢ **Lệnh Drop View** :

Ví dụ:

Drop view if exists lengthMax;

© Copyright 2021 Ths. Vũ Duy Khương

Các lệnh trong View

➢ **Lệnh đổi tên View** :

Cú pháp:

```
RENAME TABLE original_view_name
TO new_view_name;
```

© Copyright 2021 Ths. Vũ Duy Khương

Function & Stored Procedure

- **Function**: Là đoạn chương trình kịch bản (programming scripts) với các câu lệnh SQL nhúng (embedded SQL) được lưu dưới dạng đã được biên dịch và thi hành thực tiếp bởi MySQL server.
- **Stored Procedure**: cho phép lưu trữ các logic ứng dụng trên CSDL.

- **Cú pháp:**

```
DELIMITER $$
CREATE PROCEDURE procedureName()
BEGIN
    /*Xu ly*/
END; $$
DELIMITER ;
```

© Copyright 2021 Ths. Vũ Duy Khương

Function & Stored Procedure

➢ Trong đó:

- DELIMITER $$ dùng để phân cách bộ nhớ lưu trữ thủ tục Cache và mở ra một ô lưu trữ mới. Đây là cú pháp nên bắt buộc
- CREATE PROCEDURE procedureName() dùng để khai báo tạo một Procedure mới, trong đó procedureName chính là tên thủ tục còn hai từ đầu là từ khóa.
- BEGINvà END; $$ dùng để khai báo bắt đầu của Procedure và kết thúc Procedure
- Cuối cùng là đóng lại ô lưu trữ DELIMITER ;

© Copyright 2021 Ths. Vũ Duy Khương

Function & Stored Procedure

- Gọi Procedure:

```
CALL storeName();
```

Ví dụ:

```
DELIMITER $$
CREATE PROCEDURE GetAllProducts()
BEGIN
    /*Xu ly*/
END; $$
DELIMITER ;

CALL GetAllProducts();
```

© Copyright 2021 Ths. Vũ Duy Khương

Function & Stored Procedure

- Xem Procedure:

```
SHOW PROCEDURE status;
```

- Sửa Procedure:

Ví dụ:

```
DELIMITER $$

DROP PROCEDURE IF EXISTS `GetAllProducts`$$

CREATE PROCEDURE `GetAllProducts`()
BEGIN
  SELECT *  FROM products;
END$$

DELIMITER ;
```

© Copyright 2021 Ths. Vũ Duy Khương

Biến trong Stored Procedure

- Cú pháp:

  DECLARE variable_name datatype(size) **DEFAULT** default_value

- Trong đó:

- **DECLARE**: là từ khóa tạo biến
- **variable_name** là tên biến
- **datatype(size)** là kiểu dữ liệu của biến và kích thước của nó
- **DEFAULT default_value**: là gán giá trị mặc định cho biến

Ví dụ:

DECLARE product_title **VARCHAR**(255) **DEFAULT** 'No Name';

© Copyright 2021 Ths. Vũ Duy Khương

Phạm vi hoạt động của biến trong Stored Procedure

- Một biến bên trong phần thân của Procedure (giữa BEGIN và END) thì đó ta gọi là biến cục bộ của Procedure

```
1  DELIMITER $$
2  DROP PROCEDURE IF EXISTS tinhTong $$
3  CREATE PROCEDURE tinhTong()
4  BEGIN
5      DECLARE a INT (11) DEFAULT 0;
6      DECLARE b INT (11) DEFAULT 0;
7      DECLARE tong INT (11) DEFAULT 0;
8
9      SET a = 200;
10     SET b = 300;
11     SET tong = a + b;
12
13     SELECT tong;
14
15 END; $$
16 DELIMITER;
```

call tinhTong ➔ 500

© Copyright 2021 Ths. Vũ Duy Khương

If else trong Stored Procedure

- Mệnh đề if cho phép bạn tạo luồng xử lý rẻ nhánh, nếu đúng thì thực thì và ngược lại mệnh đề sai thì nó sẽ không thực thi.

```
1  IF if_expression THEN
2      commands
3     ELSEIF elseif_expression THEN
4      commands
5     ELSE
6      commands
7  END IF;
```

© Copyright 2021 Ths. Vũ Duy Khương

If else trong Stored Procedure

- Ví dụ: Tạo bảng member như dưới

```sql
CREATE TABLE IF NOT EXISTS `members` (
  `us_id` INT(11) NOT NULL AUTO_INCREMENT,
  `us_username` VARCHAR(30) COLLATE utf8_unicode_ci DEFAULT NULL,
  `us_password` VARCHAR(32) COLLATE utf8_unicode_ci DEFAULT NULL,
  `us_level` TINYINT(1) DEFAULT '0',
  PRIMARY KEY (`us_id`)
) ENGINE=INNODB  DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci AUTO_INCREMENT=4 ;

--
-- Contenu de la table `members`
--

INSERT INTO `members` (`us_id`, `us_username`, `us_password`, `us_level`) VALUES
(1, 'admin', '57e34a1be668ebd6e40d430806beb099', 1),
(2, 'member', '57e34a1be668ebd6e40d430806beb099', 2),
(3, 'banded', '57e34a1be668ebd6e40d430806beb099', 0);
```

- Trong bảng này ta cần chú ý đến field us_level như sau:
  - Nếu us_level = 0 => tài khoản bị khóa
  - Nếu us_level = 1 => admin
  - Nếu us_level = 2 => member

© Copyright 2021 Ths. Vũ Duy Khương

If else trong Stored Procedure

➢ Bài giải:

```
DELIMITER $$

DROP PROCEDURE IF EXISTS `checkLogin`$$

CREATE PROCEDURE `checkLogin`(
    IN input_username VARCHAR(255),
    IN input_password VARCHAR(255),
    OUT result VARCHAR(255)
)
BEGIN
    /*Bien flag luu tru level. Mac dinh la -1*/
    DECLARE flag INT(11) DEFAULT -1;

    /*Thuc hien truy van gan level vao bien flag*/
    SELECT us_level INTO flag FROM members
    WHERE us_username = input_username AND us_password = MD5(input_password);

    /*Sau khi thuc hien lenh select nay ma ko co du lieu thi
      luc nay flag se khong thay doi. Chinh vi the neu flag = -1 tuc la sai thong tin
    */
    IF (flag <= 0) THEN
            SET result = 'Thong tin dang nhap sai';
        ELSEIF (flag = 0) THEN
            SET result = 'Tai khoan bi khoa';
        ELSEIF (flag = 1) THEN
            SET result = 'Tai khoan admin';
        ELSE
            SET result = 'Tai khoan member';
    END IF;
END$$

DELIMITER ;
```

➢ Sử dụng:

```
CALL checkLogin('admin', 'vancuong', @result);
SELECT @result;

-- hoặc

CALL checkLogin('member', 'vancuong', @result);
SELECT @result;

-- hoặc

CALL checkLogin('banded', 'vancuong', @result);
SELECT @result;
```

© Copyright 2021 Ths. Vũ Duy Khương

Case trong Stored Procedure

➢ Cú pháp:

```
CASE  case_expression
   WHEN when_expression_1 THEN commands
   WHEN when_expression_2 THEN commands
   ...
   ELSE commands
END CASE;
```

Kết quả:

```
CALL docSo(1); -- MOT
CALL docSo(2); -- HAI
CALL docSo(3); -- BA
CALL docSo(4); -- BON
CALL docSo(5); -- NAM
CALL docSo(6); -- SAU
CALL docSo(7); -- BAY
CALL docSo(8); -- TAM
CALL docSo(9); -- CHIN
CALL docSo('tum lum'); -- KHONG TIM THAY
```

➢Ví dụ:

```
DELIMITER $$

DROP PROCEDURE IF EXISTS `docSo`$$

CREATE PROCEDURE `docSo`(IN a INT(11))
BEGIN

    DECLARE message VARCHAR(255);

    CASE a
        WHEN 0 THEN
            SET message = 'KHONG';
        WHEN 1 THEN
            SET message = 'MOT';
        WHEN 2 THEN
            SET message = 'HAI';
        WHEN 3 THEN
            SET message = 'BA';
        WHEN 4 THEN
            SET message = 'BON';
        WHEN 5 THEN
            SET message = 'NAM';
        WHEN 6 THEN
            SET message = 'SAU';
        WHEN 7 THEN
            SET message = 'BAY';
        WHEN 8 THEN
            SET message = 'TAM';
        WHEN 9 THEN
            SET message = 'CHIN';
        ELSE
            SET message = 'KHONG TIM THAY';

    END CASE;

    SELECT message;

END$$

DELIMITER ;
```

© Copyright 2021 Ths. Vũ Duy Khương

Vòng lặp While trong Stored Procedure

➢ Cú pháp:

```
WHILE expression DO
   Statements
END WHILE;
```

➢Ví dụ:

```
DELIMITER $$

DROP PROCEDURE IF EXISTS loopWhile$$

CREATE PROCEDURE loopWhile(
    IN a INT(11),
    IN b INT(11)
)
BEGIN
    -- Chuoi in ra man hinh--
        DECLARE str  VARCHAR(255) DEFAULT '';

        WHILE (a <= b) DO
        SET  str = CONCAT(str,a,',');
                SET  a = a + 1;
        END WHILE;

        SELECT str;
END$$
DELIMITER ;
```

➢Kết quả:

CALL loopWhile(1,10);

© Copyright 2021 Ths. Vũ Duy Khương

Ưu điểm Stored Procedure

- Để tăng hiệu xuất xử lý của ứng dụng
- Giúp giảm thời gian giao tiếp giữa các ứng dụng với hệ quản trị MYSQL
- Giúp các ứng dụng nhìn minh bạch hơn, nghĩa là khi ta định nghĩa các thao tác xử lý vào một Stored thì công việc của các ngôn ngữ lập trình khác chỉ quan tâm đến tên thủ tục, các tham số truyền vào chứ không cần biết nó thực hiện như thế nào

© Copyright 2021 Ths. Vũ Duy Khương

Nhược điểm Stored Procedure

- Tốn bộ nhớ
- Thực hiện quá nhiều xử lý trong mỗi thủ tục thì CPU làm việc nặng hơn
- Khó phát triển ứng dụng

© Copyright 2021 Ths. Vũ Duy Khương

Trigger

- **Định nghĩa**: Triggers là quá trình tự động thi hành các lệnhSQL hoặc hàm/thủ tục sau hoặc trước các lệnh INSERT, UPDATE, hoặc DELETE.
- Các ứng dụng có thể bao gồm: lưu lại thay đổi hoặc cập nhật dữ liệu các bảng khác.
- Trigger chạy sau mỗi câu lệnh cập nhật bảng do đó có thể thêm tải với CSDL

© Copyright 2021 Ths. Vũ Duy Khương

Trigger

➢ Cú pháp:

```
1  CREATE TRIGGER trigger_name
2  {BEFORE | AFTER} {INSERT | UPDATE| DELETE }
3  ON table_name FOR EACH ROW
4  trigger_body;
```

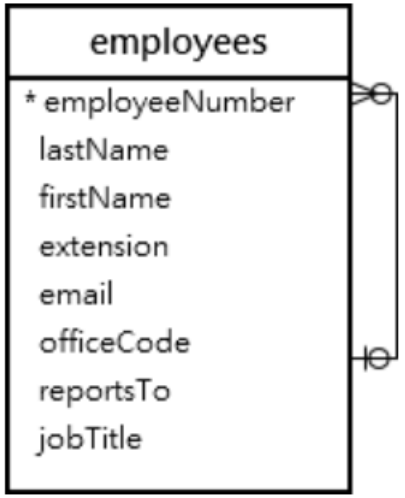
➢ Trong đó:

- trigger_name là tên của trigger mà bạn muốn đặt.
- INSERT | UPDATE| DELETE: Mỗi trigger sẽ được gắn cho một trong ba hành động này.
- BEFORE | AFTER: Nếu bạn chọn Before thì trigger sẽ chạy trước khi hành động xảy ra, ngược lại After là sau hành động xảy ra.
- ON table_name là table sẽ được gắn trigger này.
- FOR EACH ROW là sẽ duyệt qua từng row.

© Copyright 2021 Ths. Vũ Duy Khương

Trigger

➢ Ví dụ:



Step 1: tạo một table employees có cấu trúc như sau.

```
CREATE TABLE employees_audit (
    id INT AUTO_INCREMENT PRIMARY KEY,
    employeeNumber INT NOT NULL,
    lastname VARCHAR(50) NOT NULL,
    changedat DATETIME DEFAULT NULL,
    action VARCHAR(50) DEFAULT NULL
);
```

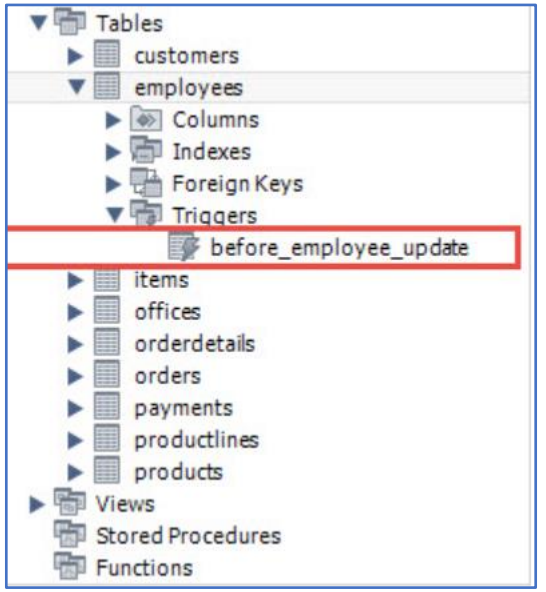
Step 2: Tạo một Trigger như sau

```
EATE TRIGGER before_employee_update
    BEFORE UPDATE ON employees
    FOR EACH ROW
 INSERT INTO employees_audit
 SET action = 'update',
     employeeNumber = OLD.employeeNumber,
     lastname = OLD.lastname,
     changedat = NOW();
```

© Copyright 2021 Ths. Vũ Duy Khương

Trigger

- Xem Trigger tồn tại chưa:
  SHOW TRIGGERS;



- Cập nhập 1 row bất kỳ

```
1  UPDATE employees
2  SET
3      lastName = 'Phan'
4  WHERE
5      employeeNumber = 1056;
```

- Drop Trigger:

```
1  DROP TRIGGER [IF EXISTS] [schema_name.]trigger_name;
```

© Copyright 2021 Ths. Vũ Duy Khương

Bai Tap

Bài 1 : Tạo csdl hong_hoa, có 2 bảng là mặt hàng và nhật ký bán hàng như sau:

```
CREATE TABLE mathang
(
    mahang    NVARCHAR(5)   PRIMARY KEY,
    tenhang   NVARCHAR(50)  NOT NULL,
    soluong   INT
);
CREATE TABLE   nhatkybanhang
(
    stt INT  PRIMARY KEY,
    ngay DATETIME ,
    nguoimua  VARCHAR(30),
    mahang VARCHAR(5),
    soluong   INT,
    giaban    INT
);
```

© Copyright 2021 Ths. Vũ Duy Khương

Bai Tap

Môt tả:
- Bảng mặt hàng mathang: sẽ lưu trữ các mặt hàng có trong kho, với mã hàn, tên hàng,số lượng trong kho.

- Khi 1 sản phẩm được bán t sẽ ghi nó vào bảng nhatkybanhang với mã hàng, tên hàng, stt....bằng câu lênh:
insert into nhatkybanhang (stt, ngay, nguoimua, mahang, soluong, giaban) values (1 ,'2012-5-20','Mr ngoc', 'H1',1 ,7000);

- Và lúc này trường số lượng bên bảng mathang sẽ tự động giảm xuống. bằng giá trị số lượng ban đầu trừ đi số lượng bán đi. và đồng thời cập nhật vào bảng mathang.

© Copyright 2021 Ths. Vũ Duy Khương

Bai Tap

1. Tạo Trigger

```
delimiter $$
create trigger after_insert_nkbh
after insert on nhatkybanhang
for each row
begin
 UPDATE mathang
 SET soluong = soluong - inserted.soluong
 where mahang = inserted.mahang;

end $$
delimiter ;
```

© Copyright 2021 Ths. Vũ Duy Khương

Bai Tap

Bài 2: Tạo 1 procedure CheckDiem() với điểm nhập vào và hiển thị ra phân loại học sinh: Kém(0 – 5), Trung Bình[5 – 6), Trung Bình Khá[6-7), Khá [7-8) , Giỏi [8-9), Xuất sắc [9 – 10]

Bài 3: Tạo 1 procedure Check Thứ Trong tuần sử dụng Case

© Copyright 2021 Ths. Vũ Duy Khương

Q&A

© Copyright 2021 Ths. Vũ Duy Khương


FPT Software
THANK YOU