

## مثال ساده پرداخت بانکی با استفاده از تراکنش و پروسیجر در مای اس کیو ال

عنوان:

ناصر نیازی

نویسنده:

۱۴:۲۵ ۱۳۹۳/۱۰/۳۰

تاریخ:

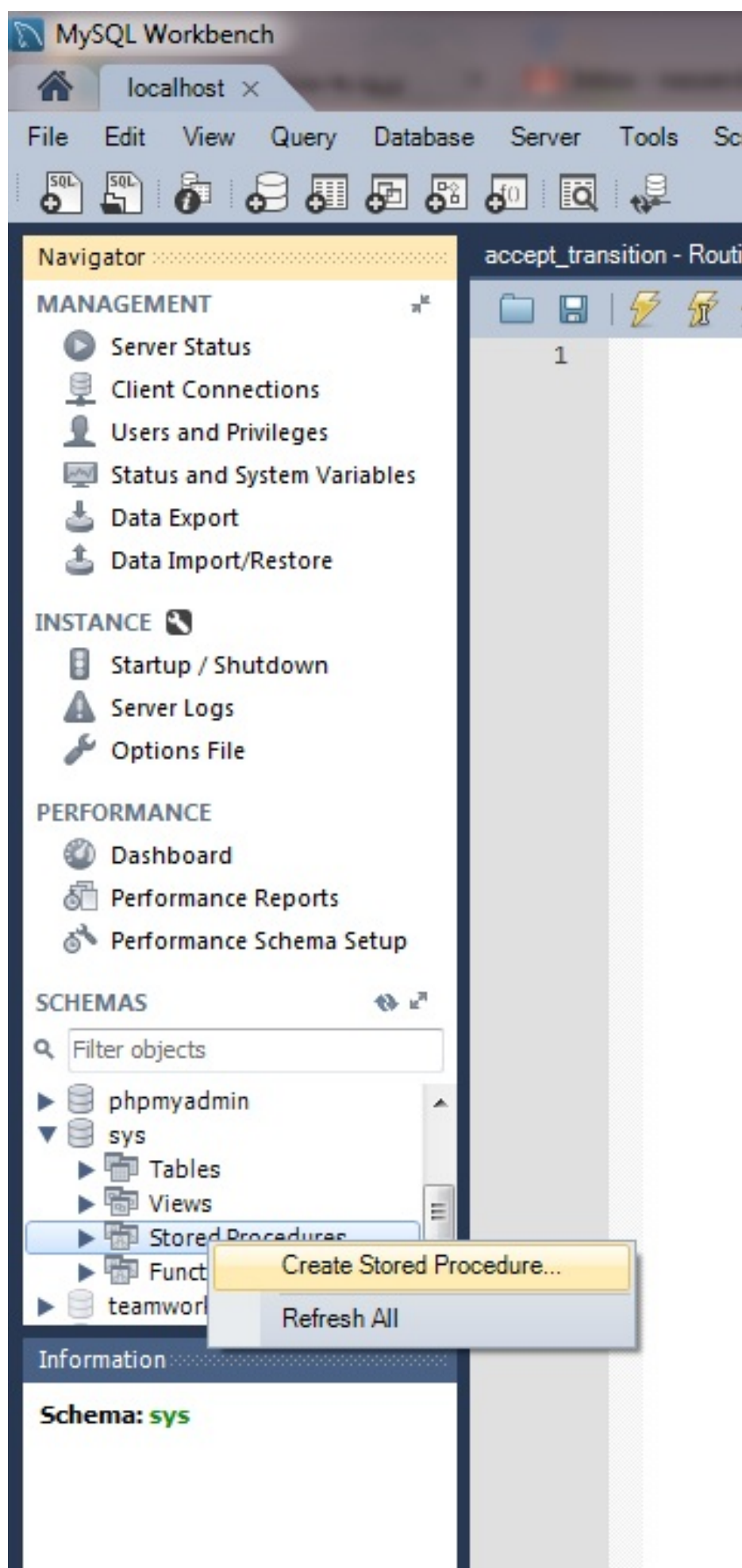
[www.dotnettips.info](http://www.dotnettips.info)

آدرس:

MySQL, transaction, Stored procedure

گروه‌ها:

برای انجام عملیاتی مثل عملیات حسابداری، نیاز به انجام پی در پی چندین دستور می‌باشد و در صورت انجام نشدن یکی از آنها، بقیه نیز نامعتبر خواهند بود که برای پیاده سازی این مکانیزم از تراکنش‌ها در بانک اطلاعاتی استفاده می‌شود. تراکنش‌ها معمولاً در بدنه‌ی توابع ذخیره شده روی بانک (stored procedure) پیاده سازی می‌شوند. برای تعریف یک پروسیجر در مای اس کیو ال من از برنامه‌ی MySQL Workbench به شکل زیر استفاده می‌کنم. البته می‌توان دستور ایجاد تابع را از روش‌های دیگر هم اجرا کرد.



در مای اس کیو ال برای تعریف یک تابع از ساختار زیر استفاده می‌کنیم :

```
DELIMITER $$

CREATE
    DEFINER=`user_name`@`host_name`|CURRENT_USER
    PROCEDURE `transition_name`(
        IN | OUT | INOUT `parameter_name` type(bigint,int , ...)
    )
    SQL SECURITY DEFINER| INVOKER
transition_name: BEGIN
#----procedure_body
END
```

نکات مربوط به تعریف :  
در قسمت

```
DEFINER=`user_name`@`host_name`|CURRENT_USER
```

کسی که تابع را تعریف کرده معرفی می‌شود. اگر شما برای انتقال دیتابیس از جایی به جای دیگر، از روش ایمپورت و اکسپورت استفاده کنید، اگر نام کاربری بانک شما متفاوت باشد، معمولاً این قسمت باعث خطا می‌شود؛ چون شما نمی‌توانید به نام فرد دیگری تابع بسازید. پیش فرض هم مقدار

```
CURRENT_USER
```

در نظر گرفته می‌شود که همان اسم کاربری و هاست شما است.  
نکته بعدی : قسمت

```
SQL SECURITY DEFINER| INVOKER
```

است که استفاده کننده از پروسیجر را مشخص می‌کند. مقدار DEFINER یعنی فقط تعریف کننده حق استفاده از این پروسیجر را دارد و مقدار INVOKER یعنی هر کسی حق استفاده از این تابع را دارد .  
برای شرح تراکنش، مثال پرداخت بانکی را شرح می‌دهیم:

```
DELIMITER $$

CREATE
    DEFINER=CURRENT_USER
    PROCEDURE `transition_pay`(
        #-----input value
        IN `pay_value` bigint,
        IN `admin_id` int,
        #-----result code
        OUT `result` bigint
    )
    SQL SECURITY INVOKER
transition_pay: BEGIN
DECLARE admin_credit DOUBLE DEFAULT 0;
SELECT `Credit`
INTO admin_credit
FROM `Admin`

WHERE `Admin_id` = admin_id
#----- transaction body
END
```

در قسمت بالا متغیری را تعریف کرده و آخرین میزان اعتبار ادمین را داخل آن قرار دادیم تا در قسمت تراکنش، مقدار پرداختی را به آن اضافه کنیم و دو باره ادمین را آپدیت کنیم. اگر بخواهیم به دلیلی قبل از رسیدن به تراکنش آن را کنسل کنیم، می‌توان از دستور LEAVE استفاده کرد:

مثال :

```
IF admin_id=0 THEN
set result = -1 ;
#exit procedure
LEAVE transition_pay;
END IF;
```

حال شروع تراکنش حالت ساده :

```
START TRANSACTION;
INSERT INTO
                `PayBalance` (`Value` , `Admin_id` )
                VALUES (pay_value, admin_id);

UPDATE `Admin`
SET `Credit`=admin_credit + pay_value
WHERE `admin_id`=admin_id;
COMMIT;
```

با پایان تراکنش، تمام مقادیر به درستی در بانک ذخیره می‌گردند. حال اگر بخواهیم به دلیلی داخل تراکنش آن را لغو کنیم از دستور ROLLBACK استفاده می‌کنیم.

مثال:

```
IF pay_value=0 THEN
set result = -1 ;
#roolback procedure
ROLLBACK ;
END IF;
```

برای اطمینان از اجرا شدن دستورات در مای اس کیو ال می‌توان از

```
SET autocommit = {0 | 1}
```

نیز استفاده کرد که مقدار پیش فرض آن یک است. یعنی هر دستوری بلافاصله اجرا شود. می‌توان قبل از دستوراتی که می‌خواهیم پی در پی اجرا شوند، یک بار آن را صفر و بعد از اجرای دستورات آنرا یک کنیم. نکته آخر اینکه با استفاده از زبان پی اچ پی هم می‌توان تراکنشی را شروع و تمام کرد و بین این دو دستورات مورد نظر را نوشت و همیشه وجود پروسیجر الزامی نیست.