

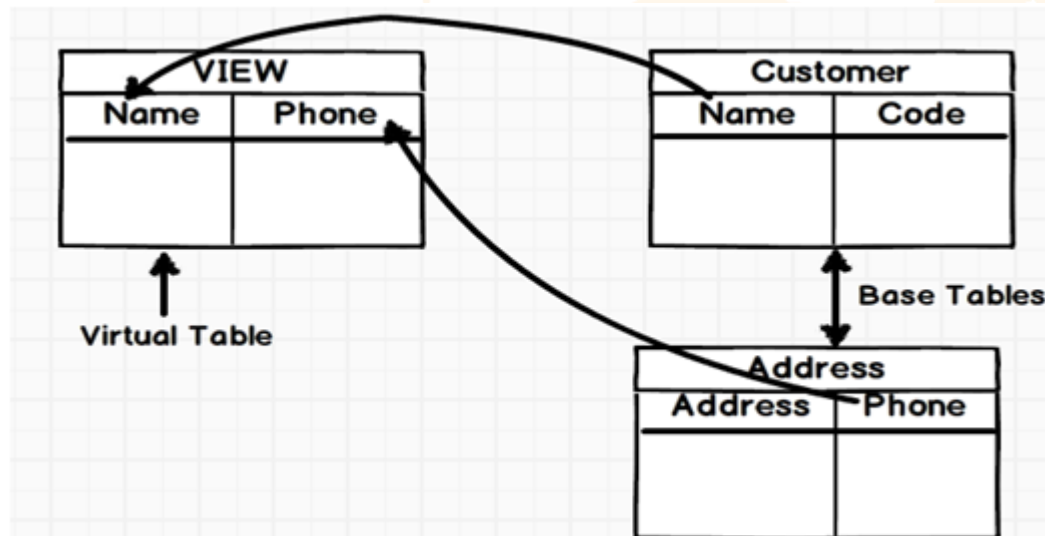
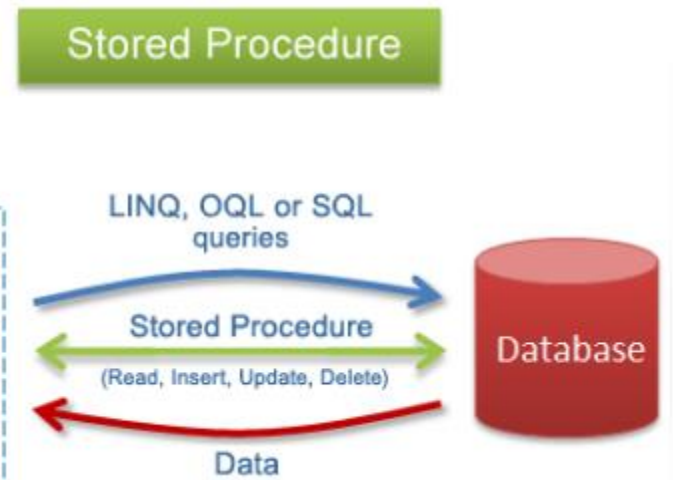
TRƯỜNG ĐÀO TẠO LẬP TRÌNH VIÊN VÀ QUẢN TRỊ MẠNG QUỐC TẾ BACHKHOA-APTECH

Bài 10

Views và Store Procedure

Tóm tắt

- Views
- Store Procedure



Tóm tắt

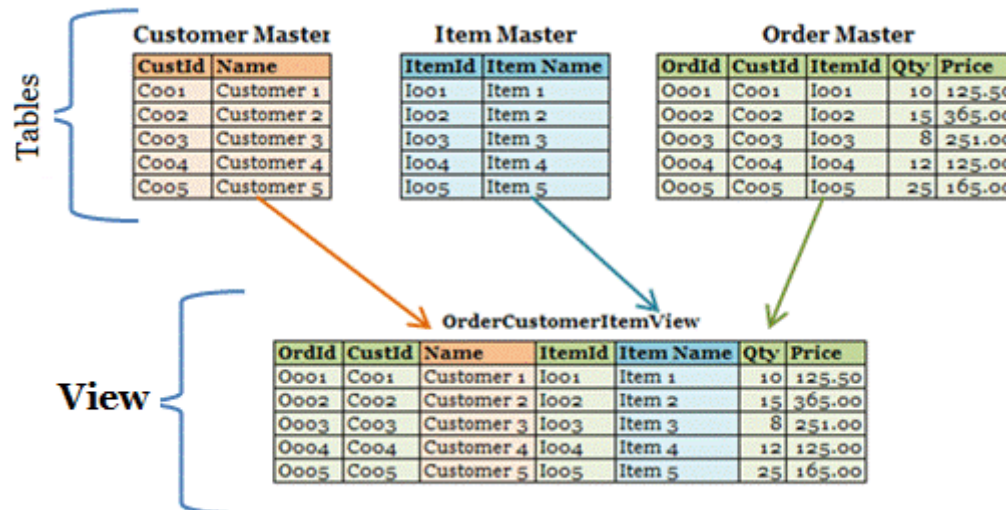
- Một csdl SQL Server có hai danh mục đối tượng chính:
 1. Các đối tượng **lưu trữ dữ liệu**.
 2. Các đối tượng **truy xuất, thao tác**, hoặc cung cấp truy xuất đến dữ liệu.
- **View** và thủ tục (**stored procedures**) thuộc về danh mục thứ hai.

Views

- View là một **bảng ảo**, được tạo ra bởi các cột được lấy từ **một hoặc nhiều bảng** (table) khác nhau
- Các bảng mà view được tạo từ đó được gọi là các **bảng cơ sở**.
- Những bảng này có thể trong cùng csdl hoặc từ các csdl khác.
- View cũng có thể bao gồm các cột **lấy từ một View khác** trong cùng csdl hoặc từ csdl khác.
- Một View có thể có tối đa **1024 cột**.

Views

- Dữ liệu bên trong view được lấy từ các bảng cơ sở, là các **bảng được tham chiếu** trong phần định nghĩa của view.
- Các dòng và các cột của view **được tạo động** khi view được tham chiếu.



Views

- Có thể tạo một view trong csdl hiện tại bằng việc dùng lệnh **CREATE VIEW**.
- Người dùng chỉ có thể tạo view với các **cột lấy từ bảng cơ sở hoặc từ các view** khác nếu người dùng có quyền truy cập đến các bảng và view đó.

Cú pháp:

```
CREATE VIEW <view_name>  
AS <select_statement>
```

Trong đó:

- **view_name**: chỉ ra tên của view.
- **select_statement**: lệnh SELECT định nghĩa view.

Views

Ví dụ:

```
USE QuanLyBanHang
```

```
Go
```

```
CREATE VIEW vwProductInfo AS
```

```
SELECT ProductId, ProductName, PriceInput,  
PriceOutput, [Status] FROM Product
```

```
GO
```

Tiền tố “vw” trong tên của View là **quy ước để ám chỉ** đây là view, phân biệt với các đối tượng khác.

Views

Gọi View:

```
SELECT * FROM vwProductInfo
```

Kết quả:

	ProductId	ProductName	PriceInput	PriceOutput	Status
1	1	Giày buộc giày công sở Sanvado	350000	400000	1
2	2	Giày nam buộc dây James Blanc	500000	600000	1
3	3	Giày da mềm Asos	350000	400000	1
4	4	Giày cao gót mũi nhọn	60000	650000	1
5	5	Giày cao gót đế nhá đá mùi hồ	350000	450000	1
6	6	Giày cao gót dây buộc Asos	550000	600000	1
7	7	Đép xỏ ngón nhiều màu - DT170	750000	800000	1
8	8	Đép quai ngang	300000	350000	1
9	9	Đép đế xuống hoa hồng quai trong	350000	400000	1
10	10	Đép cao su nam quai liền	630000	680000	1
11	11	xăng đan nam	700000	750000	1
12	12	Đép xỏ ngón kito thailand	800000	850000	1
13	13	Áo thun Nike cổ tròn	350000	400000	1

Views

Tạo View sử dụng từ khóa JOIN:

- Từ khóa JOIN được sử dụng để tạo view.
- Câu lệnh CREATE VIEW được sử dụng cùng với **từ khóa JOIN** để tạo một view sử dụng các cột từ nhiều bảng.

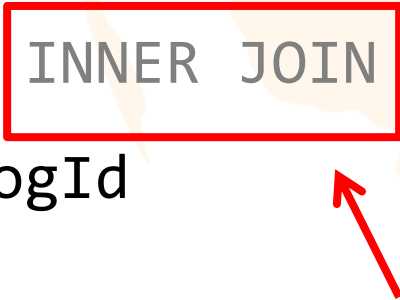
```
CREATE VIEW <view_name>  
AS  
SELECT * FROM table_name1  
JOIN table_name2  
ON table_name1.column_name = table_name2.column_name
```

Views

Ví dụ: tạo view hiển thị sản phẩm với tên danh mục rõ ràng.

-- QuanLyBanHang

```
CREATE VIEW vwProductDetails AS
SELECT p.ProductId, p.ProductName,
p.PriceOutput, c.CatelogName, p.[Status] FROM
Product p INNER JOIN Catelog c ON p.CatelogId
= c.CatelogId
GO
```



Views

Kết quả:

	ProductId	ProductName	PriceOutput	CatalogName	Status
1	1	Giày buộc giày công sở Sanvado	400000	Giày nam	1
2	2	Giày nam buộc dây James Blanc	600000	Giày nam	1
3	3	Giày da mềm Asos	400000	Giày nam	1
4	4	Giày cao gót mũi nhọn	650000	Giày nữ	1
5	5	Giày cao gót đế đinh đá mũi hờ	450000	Giày nữ	1
6	6	Giày cao gót dây buộc Asos	600000	Giày nữ	1
7	7	Dép xỏ ngón nhiều màu - DT170	800000	Dép nữ	1
8	8	Dép quai ngang	350000	Dép nữ	1
9	9	Dép đế xuồng hoa hồng quai tr...	400000	Dép nữ	1
10	10	Dép cao su nam quai liền	680000	Dép nam	1
11	11	xăng đan nam	750000	Dép nam	1
12	12	Dép xỏ ngón kito thailand	850000	Dép nam	1
13	13	Áo thun Nike cổ tròn	400000	Áo nam	1
14	14	Áo sơ mi nam ngắn tay họa tiết	400000	Áo nam	1

Views

Nguyên tắc và giới hạn:

- Một view được tạo bằng việc sử dụng câu lệnh **CREATE VIEW**.
- Tên view phải là **duy nhất**, **không thể trùng** với tên các bảng khác trong cùng lược đồ.
- View không thể tạo trên các **bảng tạm**.
- View không thể có **full-text index**.
- View **không chứa** định nghĩa **DEFAULT**.

Views

- Câu lệnh CREATE VIEW chỉ có thể bao gồm mệnh đề **ORDER BY** nếu như có từ khóa **TOP** được sử dụng.
- View không thể tham chiếu hơn **1024 cột**.
- Câu lệnh CREATE VIEW **không thể** bao gồm từ khóa **INTO** .
- Câu lệnh CREATE VIEW không thể kết hợp với các lệnh Transact-SQL khác trong cùng một khối (batch).

Views

Ví dụ: tạo View có sắp xếp sản phẩm theo tên, câu lệnh hợp lệ khi kết hợp với từ khóa TOP.

-- Tạo View đã sắp xếp và lấy ra 10 sản phẩm đầu tiên

```
CREATE VIEW vwSortedProductDetails AS
```

```
SELECT TOP 10 p.ProductId, p.ProductName,
```

```
p.PriceOutput, c.CatelogName FROM Product p INNER
```


```
JOIN Catelog c ON c.CatelogId = p.CatelogId
```

```
ORDER BY p.ProductName -- Sắp xếp theo tên
```



```
GO
```

Views

Kết quả:



	ProductId	ProductName	PriceOutput	CatelogName
1	16	Áo sơ mi nam dài tay VŨ TUẤN	450000	Áo nam
2	14	Áo sơ mi nam ngắn tay họa tiết	400000	Áo nam
3	15	Áo thun nam cổ tròn ECO JEA	300000	Áo nam
4	13	Áo thun Nike cổ tròn	400000	Áo nam
5	17	Áo voan cộc tay cao cổ	900000	Áo nữ
6	10	Dép cao su nam quai liền	680000	Dép nam
7	9	Dép đế xuồng hoa hồng quai trong	400000	Dép nữ
8	8	Dép quai ngang	350000	Dép nữ
9	12	Dép xỏ ngón kito thailand	850000	Dép nam
10	7	Dép xỏ ngón nhiều màu - DT170	800000	Dép nữ



Views

INSERT với VIEW:

Cột có thuộc tính IDENTITY: câu lệnh INSERT được sử dụng để thêm các dòng mới tới bảng hoặc view. Giá trị của cột được cung cấp tự động nếu:

- Cột có giá trị **mặc định** được chỉ ra.
- Cột có kiểu dữ liệu **timestamp**.
- Cột **chấp nhận** các giá trị **null**.
- Cột là cột **tính toán**.

Khi sử dụng câu lệnh INSERT trên view, nếu **vi phạm** (violated) bất kỳ **quy tắc** (rules), bản ghi sẽ **không được chèn**.

Views

Ví dụ: tạo 2 bảng **Employee_Personal_Details** và **Employee_Salary_Details** như sau:

Bảng Employee_Personal_Details		
Cột	Kiểu dữ liệu	Mô tả
EmpID	INT NOT NULL	ID
FirstName	varchar(30) NOT NULL	Họ
LAStName	varchar(30) NOT NULL	Tên
Address	varchar(30)	Địa chỉ

Views

Bảng Employee_Salary_Details		
Cột	Kiểu dữ liệu	Mô tả
EmpID	INT NOT NULL	ID
Designation	varchar(30)	Chỉ định
Salary	INT NOT NULL	Lương

2 bảng trên có mối tương quan là **EmpID** (mã nhân viên), để cho đơn giản trong ví dụ không thực hiện tạo khóa ngoại.

Views

Code SQL tạo bảng:

```
CREATE TABLE Employee_Personal_Details (  
    EmpID INT NOT NULL,  
    FirstName varchar(30) NOT NULL,  
    LAsTName varchar(30) NOT NULL,  
    Address varchar(30))  
GO
```

```
CREATE TABLE Employee_Salary_Details (  
    EmpID  
    INT NOT NULL,  
    Designation varchar(30),  
    Salary INT NOT NULL)  
GO
```

Views

Code SQL tạo View:

-- Tạo View

```
CREATE VIEW vwEmployee_Details AS
SELECT e1.EmpID, FirstName, LASTName,
Designation, Salary
FROM Employee_Personal_Details e1 JOIN
Employee_Salary_Details e2
ON e1.EmpID = e2.EmpID
```

Views

Thêm dữ liệu vào View bằng INSERT:

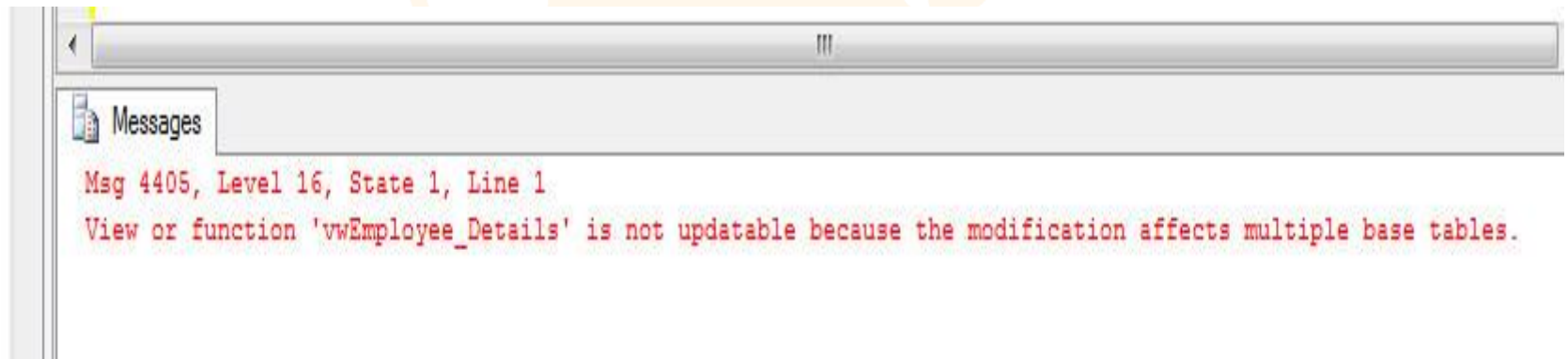
-- Thêm dữ liệu vào view

```
INSERT INTO vwEmployee_Details VALUES
```

```
(2, 'Nguyễn', 'Ái Quốc', 'Lãnh tụ vĩ đại', 16000)
```

Lỗi: do có ảnh hưởng tới cột **EmpID** ở bảng

Employee_Salary_Details (not null).



Views

UPDATE với VIEW:

- Câu lệnh UPDATE có thể được sử dụng để **thay đổi dữ liệu** trên view.
- Việc cập nhật trên view cũng sẽ **cập nhật đến các bảng** phía dưới.

Đoạn code dưới đây tạo bảng có tên Product_Details:

```
CREATE TABLE Product_Details(  
ProductID int, ProductName varchar(30), Rate  
money)
```

Views

Giả sử bảng đã có dữ liệu:

	ProductID	ProductName	Rate
1	5	DVD Writer	2250.00
2	4	DVD Writer	1250.00
3	6	DVD Writer	1250.00
4	2	External Hard Drive	4250.00
5	3	External Hard Drive	4250.00

Tiếp đó, tạo View dựa trên bảng Product_Details:

```
CREATE VIEW vwProduct_Details AS
```

```
SELECT ProductName, Rate FROM Product_Details
```

Views

Viết lệnh cập nhật dữ liệu trên View:

```
UPDATE vwProduct_Details
```

```
SET Rate=3000
```

```
WHERE ProductName='DVD Writer'
```

Kết quả: toàn bộ sản phẩm “**DVD Writer**” được cập nhật.

	ProductID	ProductName	Rate
1	5	DVD Writer	3000.00
2	4	DVD Writer	3000.00
3	6	DVD Writer	3000.00
4	2	External Hard Drive	4250.00
5	3	External Hard Drive	4250.00

Views

DELETE với VIEW:

- Có thể dùng câu lệnh DELETE để **xóa các dòng dữ liệu** từ view.
- Khi các dòng được xóa khỏi view, các **dòng tương ứng trong bảng cơ sở** cũng được xóa.

Ví dụ: xem xét view **vwCustDetails** liệt kê thông tin tài khoản của nhiều khách hàng khác nhau. Khi khách hàng đóng tài khoản, chi tiết của khách hàng cần được xóa.

Cú pháp:

```
DELETE FROM <view_name>  
WHERE <search_condition>
```

Views

- Giả sử một bảng có tên **Customer_Details**, và view có tên **vwCustDetails** được tạo dựa trên bảng này.
- Đoạn code sau đây được sử dụng để xóa các bản ghi có **CustID** là **C0004** khỏi view vwCustDetails.

Code:

```
DELETE FROM vwCustDetails WHERE CustID='C0004'
```

Views

Table and View Before Deletion

Table-dbo Customer_details				
CustID	AccNo	AccName	Date of Birth	City
C0001	1	Jane	02/02/1980	Topeka
C0002	2	Haris	05/12/1978	Lansing
C0003	3	Pitts	10/11/1985	Columbus
C0004	4	Monaliza	11/12/1980	San Francisco

vwCustDetails			
CustID	AccNo	AccName	City
C0001	1	Jane	Topeka
C0002	2	Haris	Lansing
C0003	3	Pitts	Columbus
C0004	4	Monaliza	Santa Maria

Xóa trên View

Delete

Table and View After Deletion

Table-dbo Customer_details				
CustID	AccNo	AccName	Date of Birth	City
C0001	1	Jane	02/02/1980	Topeka
C0002	2	Haris	05/12/1978	Lansing
C0003	3	Pitts	10/11/1985	Columbus

vwCustDetails			
CustID	AccNo	AccName	City
C0001	1	Jane	Topeka
C0002	2	Haris	Lansing
C0003	3	Pitts	Columbus

Tác động trên cả bảng cơ sở và View

Views

- Với nhu cầu thay đổi định nghĩa, một view có thể được chỉnh sửa bằng cách xóa đi sau đó tạo lại hoặc thực thi câu lệnh **ALTER VIEW**.
- Câu lệnh ALTER VIEW giúp chỉnh sửa view tồn tại mà **không cần phải phân quyền** hoặc thiết lập lại các thuộc tính khác của nó.
- View thường được chỉnh sửa khi người dùng **cần bổ sung thông tin hoặc thay đổi** các bảng cơ sở trong phần định nghĩa.

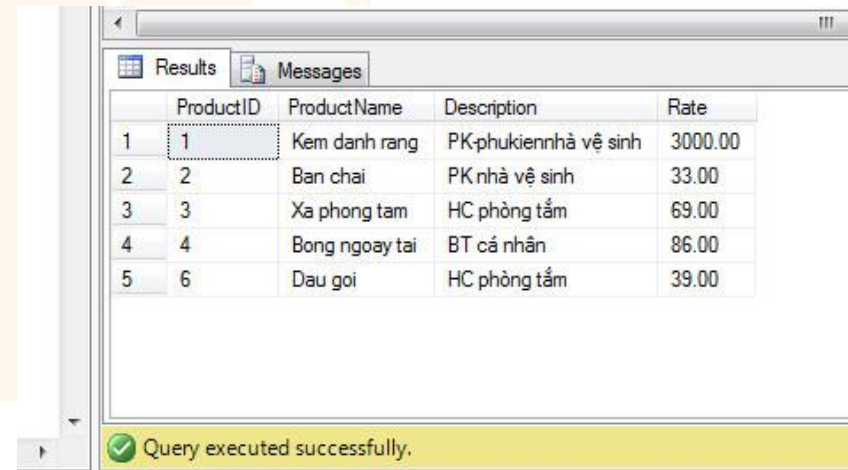
Views

Cú pháp:

```
ALTER VIEW <view_name>  
AS <select_statement>
```

-- **Chỉnh sửa View**

```
ALTER VIEW vwProduct_Full_Details AS  
SELECT ProductID, ProductName, [Description], Rate  
FROM Product_Details
```



	ProductID	ProductName	Description	Rate
1	1	Kem danh rang	PK-phukiennhà vệ sinh	3000.00
2	2	Bàn chải	PK nhà vệ sinh	33.00
3	3	Xà phòng tắm	HC phòng tắm	69.00
4	4	Bong ngoay tai	BT cá nhân	86.00
5	6	Dau goi	HC phòng tắm	39.00

Query executed successfully.

Views

- Một view có thể xóa khỏi csdl bằng lệnh **DROP VIEW**.
- Khi xóa view, dữ liệu trong **bảng cơ sở không bị ảnh hưởng** (vẫn còn).
- Định nghĩa của view và thông tin khác gắn kết với view **bị xóa khỏi system catalog**.
- Tất cả các **quyền hạn** gán cho view cũng được xóa.

Cú pháp:

```
DROP VIEW <view_name>
```

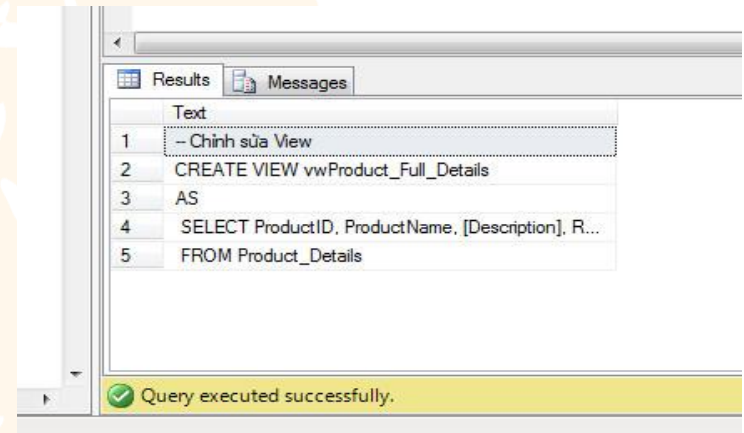
```
DROP VIEW vwProduct_Full_Details
```

Views

- Định nghĩa của view **giúp để hiểu dữ liệu** được lấy từ các bảng nguồn như thế nào.
- Thủ tục **sp_helptext** hiển thị thông tin có liên quan đến view khi tên của view được chỉ ra cho tham số của nó.

Cú pháp:

```
sp_helptext <view_name>
```



-- Xem mã nguồn SQL viết lên View

```
EXEC sp_helptext vwProduct_Full_Details
```

Views

Mệnh đề CHECK OPTION:

- Được sử dụng để đảm bảo **toàn vẹn miền giá trị** (domain integrity); Nó kiểm tra các **giá trị khi cập nhật trên view** phải thỏa mãn điều kiện được chỉ ra ở mệnh đề WHERE trong câu lệnh SELECT.
- Mệnh đề WITH CHECK OPTION **đảm bảo tất cả các câu lệnh chỉnh sửa** được thực thi đối với view phải **tuân thủ điều kiện** được thiết lập bên trong câu lệnh SELECT.

Views

Cú pháp:

```
CREATE VIEW <view_name>  
AS select_statement [ WITH CHECK OPTION ]
```

Trong đó:

WITH CHECK OPTION: chỉ ra rằng dữ liệu chỉnh sửa trong view tiếp tục thỏa mãn định nghĩa của view.

Ví dụ: xem xét tình huống với View hiển thị các sản phẩm có giá **> 400.000**, điều gì xảy ra nếu cập nhật giá sản phẩm có **id = 13** lên **500.000**?

Views

-- View với từ khóa With Check Option

```
CREATE VIEW vwProductInfo AS
```

```
SELECT ProductId, ProductName, PriceOutput, [Status] FROM  
Product WHERE PriceOutput <= 400000
```

```
WITH CHECK OPTION;
```

34

Results		Messages		
	ProductID	ProductName	PriceOutput	Status
1	1	Giày buộc giày công sở Sanvado	400000	1
2	3	Giày da mềm Asos	400000	1
3	8	Đép quai ngang	350000	1
4	9	Đép đế xuồng hoa hồng quai trong	400000	1
5	13	Áo thun Nike cổ tròn	400000	1
6	14	Áo sơ mi nam ngắn tay họa tiết	400000	1
7	15	Áo thun nam cổ tròn ECO JEA	300000	1

Views

-- Lệnh cập nhật.

```
UPDATE vwProductInfo set PriceOutput = 500000  
WHERE ProductId = 13
```

Gặp lỗi: (do tự phủ định điều kiện trên View)

Messages

Msg 550, Level 16, State 1, Line 149

The attempted insert or update failed because the target view either specifies
WITH CHECK OPTION or spans a view that specifies WITH CHECK OPTION and one or more
rows resulting from the operation did not qualify under the CHECK OPTION constraint.
The statement has been terminated.

Views

- Một view có thể được ràng buộc tới lược đồ của bảng cơ sở bằng tùy chọn **SCHEMABINDING**.
- Khi tùy chọn SCHEMABINDING được chỉ ra, các bảng cơ sở sẽ **không thể chỉnh sửa** được nữa, vì nó sẽ làm ảnh hưởng tới định nghĩa của view.
- Nếu muốn chỉnh sửa được bảng, trước tiên cần phải chỉnh sửa hoặc **xóa bỏ sự phụ thuộc** giữa view với bảng.
- Khi sử dụng tùy chọn SCHEMABINDING trong view, tên của các đối tượng được chỉ ra trong câu lệnh SELECT cần phải **kèm cùng với tên của lược đồ**.

Views

Cú pháp:

```
CREATE VIEW <view_name> WITH SCHEMABINDING  
AS <select_statement>
```

Ví dụ:

-- Tạo bảng

```
CREATE TABLE Customers(CustID INT,CustName varchar(50),Address  
varchar(60))  
GO
```

-- Tạo view

```
CREATE VIEW vwCustomers WITH SCHEMABINDING AS  
SELECT CustID, CustName, [Address] FROM dbo.Customers  
GO
```

Views

Chỉnh sửa bảng:

-- Chỉnh sửa bảng

ALTER TABLE Customers

ALTER COLUMN CustName VARCHAR(256)

GO



Messages

Msg 5074, Level 16, State 1, Line 108

The object 'vwCustomers' is dependent on column 'CustName'.

Msg 4922, Level 16, State 9, Line 108

ALTER TABLE ALTER COLUMN CustName failed because one or more objects access this column.

Views

Cập nhật thông tin mô tả cho View (do bảng cơ sở thay đổi):

- Thủ tục lưu **sp_refreshview** cập nhật metadata(thông tin mô tả) cho view.
- Nếu thủ tục sp_refreshview **không được thực thi**, metadata của view **không được cập nhật** để phản ánh(reflect) các sự thay đổi của các bảng cơ sở.
- Điều này dẫn đến việc **tạo ra kết quả không mong muốn**(unexpected) khi view được truy vấn.

Views

- Thủ tục lưu `sp_refreshview` được trả về giá trị mã là **0 nếu thực thi thành công**, hoặc trả về số **khác 0 trong trường hợp thực thi có lỗi**.

-- Refresh view

```
EXEC sp_refreshview vwCustomers
```


Store Procedure

- **Stored Procedure** là một **nhóm nhiều câu lệnh** Transact-SQL có vai trò như một khối mã lệnh thực hiện một tác vụ cụ thể, đã được biên dịch và lưu trữ trong SQL Server dưới một cái tên nào đó và **được xử lý như một đơn vị**.
- Một thủ tục cũng có thể tham chiếu tới phương thức .NET Framework Common Language Runtime (**CLR**).
- Stored procedure được dùng cho những **công việc được thực hiện lặp lại**.

Store Procedure

- Stored procedure có thể nhận nhiều giá trị qua các tham số đầu vào và trả về các giá trị kết quả qua các tham số ra.

Lợi ích:

Tăng cường bảo mật

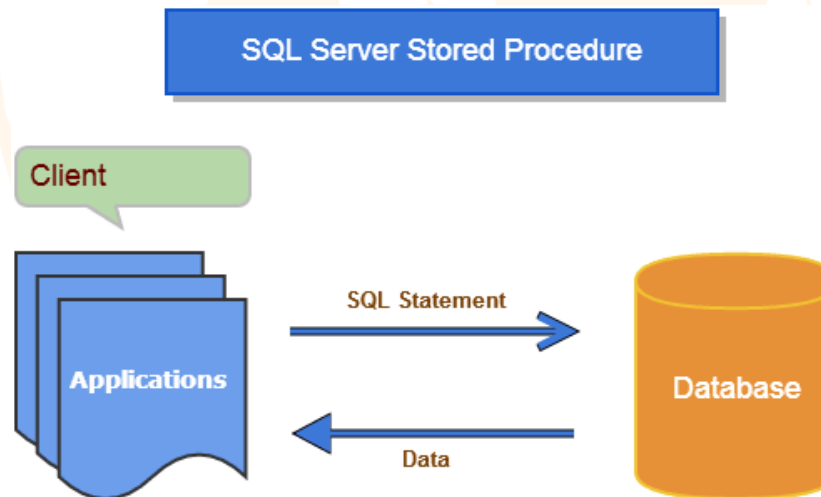
Biên dịch trước

Giảm băng thông giữa Client/Server

Tái sử dụng mã nguồn

Store Procedure

- Trong SQL Server, người dùng được phép tạo thủ tục lưu theo ý mình để thực hiện nhiều bài toán khác nhau.
- Cần phải có quyền **CREATE PROCEDURE** và **ALTER** để có thể tạo và sửa thủ tục.



<http://www.encodedna.com>

Store Procedure

```
CREATE { PROC | PROCEDURE } procedure_name  
[ { @parameter data_type } ]  
AS  
    <sql_statement>
```

Trong đó:

- **procedure_name**: chỉ ra tên của thủ tục.
- **@parameter**: chỉ ra tham số vào/ra của thủ tục.
- **data_type**: chỉ ra kiểu dữ liệu của tham số.
- **sql_statement**: là một hay nhiều câu lệnh Transact-SQL.

Store Procedure

Ví dụ:

-- Tạo thủ tục

```
CREATE PROCEDURE getProductInfor AS
```

```
SELECT top 10
```

```
ProductId, ProductName, PriceOutput, CatalogName
```

```
FROM Product inner join Catalog
```

```
ON Product.CatalogId = Catalog.CatalogId
```

Store Procedure

-- Thực thi thủ tục

EXEC getProductInfor

Results		Messages		
	ProductId	ProductName	PriceOutput	CatelogName
1	1	Giày buộc giày công sở Sanvado	400000	Giày nam
2	2	Giày nam buộc dây James Blanc	600000	Giày nam
3	3	Giày da mềm Asos	400000	Giày nam
4	4	Giày cao gót mũi nhọn	650000	Giày nữ
5	5	Giày cao gót đế nhá đá mũi hờ	450000	Giày nữ
6	6	Giày cao gót dây buộc Asos	600000	Giày nữ
7	7	Đép xỏ ngón nhiều màu - DT170	800000	Đép nữ
8	8	Đép quai ngang	350000	Đép nữ
9	9	Đép đế xuồng hoa hồng quai tr...	400000	Đép nữ
10	10	Đép cao su nam quai liền	680000	Đép nam

Store Procedure

Thủ tục có tham số

- Dữ liệu có thể được truyền từ chương trình gọi đến thủ tục được gọi bằng việc dùng các tham số.
- Tham số được chia thành hai loại
 1. **Tham số vào**: cho phép chương trình gọi **truyền** các giá trị tới một thủ tục. Các giá trị này được gán cho các biến đã được định nghĩa bên trong thủ tục.
 2. **Tham số ra**: cho phép thủ tục truyền các giá trị ngược **trở lại** chương trình gọi. Các giá trị này được gán cho các biến ở chương trình gọi.

Store Procedure

Tham số truyền vào:

- Các giá trị được truyền vào từ chương trình gọi tới thủ tục, và được nhận (lưu) vào các **tham số vào của thủ tục**.
- Tham số vào được **khai báo ngay lúc tạo** thủ tục.
- Các giá trị truyền cho tham số vào có thể là các hằng và cũng có thể là các biến.
- Các giá trị này được **truyền tại lúc gọi** thủ tục.
- Thủ tục sử dụng các giá trị này **để thực hiện** các công việc đã được chỉ ra.

Store Procedure

Tham số truyền vào:

```
CREATE PROCEDURE <procedure_name>  
@parameter <data_type>  
AS <sql_statement>
```

-- Tạo thủ tục có tham số truyền vào

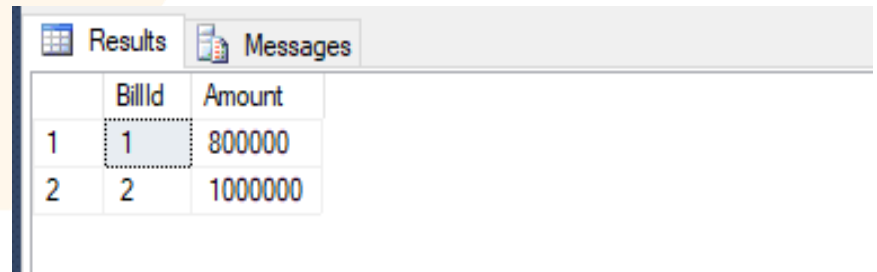
```
CREATE PROCEDURE getBillBuyCustomer
```

```
@id INT AS
```

```
SELECT BillId, Amount FROM Bill WHERE CustomerId = @id
```

-- Gọi PROCEDURE có tham số truyền vào

```
EXEC getBillBuyCustomer 1
```



	BillId	Amount
1	1	800000
2	2	1000000

Store Procedure

Tham số truyền ra:

- Tham số ra được **khai báo ngay lúc tạo** thủ tục.
- Để chỉ ra một tham số là tham số ra, từ khóa **OUTPUT** được sử dụng khi khai báo tham số.
- Cũng tương tự, câu lệnh gọi thủ tục cũng phải có biến được chỉ ra với từ khóa OUTPUT để **nhận kết quả từ thủ tục** được gọi.

```
EXECUTE <procedure_name> <parameters> OUTPUT
```

Store Procedure

-- Thủ tục có tham số trả về

```
CREATE PROCEDURE uspGetTotalSales
```

```
@name NVARCHAR(50),
```

```
@sum INT OUTPUT AS
```

```
SELECT @sum= SUM(BillId) FROM Bill b JOIN Customer c ON
```

```
b.CustomerId = c.CustomerId
```

```
WHERE c.CustomerName = @name
```

-- Thực thi thủ tục

```
DECLARE @sum INT;
```

```
EXEC totalBill 'Trinh Dinh Long', @sum OUTPUT;
```

```
PRINT 'So luong don hang: ' + convert(varchar(100),@sum);
```

Messages

So luong don hang: 3

Store Procedure

Đặc điểm của tham số OUTPUT:

Kiểu dữ liệu của tham số không thể là kiểu text và image

Câu lệnh gọi thủ tục phải chứa một biến để nhận giá trị trả về

Biến có thể sử dụng trong các câu lệnh Transact-SQL tiếp theo trong cùng lô(batch) hoặc lớp gọi thủ tục

Các tham số ra có thể là các giữ chỗ cho cursor (các biến kiểu cursor)

Store Procedure

Tạo thủ tục bằng công cụ SSMS:

1

- Mở **Object Explorer**.

2

- Trong **Object Explorer**, kết nối tới thể hiện(instance) của Database Engine.

3

- Sau khi kết nối thành công tới thể hiện(instance), hãy mở thể hiện (click và dấu + ở phần đầu của thể hiện).

4

- Mở **Databases** và mở csdl **AdventureWorks2014** database.

5

- Mở **Programmability**, click chuột phải **Stored Procedures**, và sau đó nhấn **New Stored Procedure**.

6

- Trên menu **Query**, nhấn **Specify Values for Template Parameters**. Hộp thoại **Specify Values for Template Parameters** sẽ hiển thị.

Store Procedure

Hộp thoại Specify Values for Template Parameters:

Parameter	Type	Value
Author		Name
Create Date		
Description		
Procedure_Name	sysname	ProcedureName
@Param1	sysname	@p1
Datatype_For_Param1		int
Default_Value_For_P...		0
@Param2	sysname	@p2
Datatype_For_Param2		int
Default_Value_For_P...		0

Tóm tắt bài học

- **View là một bảng ảo** cho phép chọn xem một hoặc nhiều cột từ bảng cơ sở. View được tạo bằng lệnh **CREATE VIEW**.
- Có thể **THÊM, SỬA, XÓA** dữ liệu từ View.
- Hàm thủ tục (**stored procedure**) là một nhóm các lệnh sql gói gọn trong một khối để **thực thi tác vụ cụ thể**.
- Hàm thủ tục có thể chứa tham số **VÀO** hoặc **RA**.

TRƯỜNG ĐÀO TẠO LẬP TRÌNH VIÊN VÀ QUẢN TRỊ MẠNG QUỐC TẾ BACHKHOA-APTECH

Thank for watching!

