



TRƯỜNG ĐẠI HỌC THỦ DẦU MỘT
KHOA KỸ THUẬT CÔNG NGHỆ



HỆ QUẢN TRỊ CSDL

Chương 5

TRIGGER AND TRANSACTION



Nội dung

❖ Trigger

- Giới thiệu Trigger
- Phân loại Trigger
- Tạo Trigger

❖ Transaction




TRIGGER



Giới thiệu

- ❖ Trigger là một loại stored procedure đặc biệt được thực thi một cách tự động khi có một sự kiện thay đổi dữ liệu xảy ra như Update, Insert hay Delete.
- ❖ Trigger được dùng để đảm bảo ràng buộc dữ liệu, tính nhất quán, hoặc thực hiện các quy tắc dữ liệu phức tạp.
- ❖ Trigger không sử dụng hai đặc tính của thủ tục là tham số và giá trị trả về.
- ❖ Một Trigger được định nghĩa trên 1 bảng nhưng các xử lý trong nó có thể liên quan đến nhiều bảng.




Giới thiệu (tt)

- ❖ Ưu điểm:
 - Sử dụng trigger để kiểm tra tính toàn vẹn của csdl.
 - Trigger có thể bắt lỗi logic nghiệp vụ (business logic) ở mức CSDL.
 - Có thể dùng trigger là một cách khác để thay thế việc thực hiện những công việc hẹn giờ theo lịch.
 - Trigger rất hiệu quả khi được sử dụng để kiểm soát những thay đổi của dữ liệu trong bảng.
- ❖ Nhược điểm
 - Trigger chỉ là một phần mở rộng của việc kiểm tra tính hợp lệ của dữ liệu chứ không thay thế được hoàn toàn công việc này.
 - Trigger hoạt động ngầm ở trong CSDL, không hiển thị ở tầng giao diện. Do đó, khó chỉ ra được điều gì xảy ra ở tầng CSDL.
 - Trigger thực hiện các update lên bảng dữ liệu vì thế nó làm gia tăng lượng công việc lên CSDL và làm cho hệ thống chạy chậm lại.

Bài giảng Hệ quản trị CSDL

5



Một số ứng dụng

- ❖ Kiểm tra ràng buộc toàn vẹn dữ liệu
- ❖ Kiểm soát dữ liệu hiện tại khi có thay đổi đến giá trị của mẫu tin trong bảng
- ❖ Kiểm tra dữ liệu nhập vào có phù hợp với mối quan hệ giữa các bảng không
- ❖ Kiểm tra quá trình xóa bản ghi

6



Phân loại

Có 3 loại chính

- ❖ INSERT Trigger
- ❖ UPDATE Trigger
- ❖ DELETE Trigger

7



Tạo Trigger

- ❖ Tạo Trigger bằng Enterprise Manager
- ❖ Tạo Trigger bằng T_SQL

Cú pháp:

Create Trigger <Tên Trigger>


On <Tên Bảng/Tên View>

For |After|Instead of [DELETE, INSERT, UPDATE]

AS

<Câu lệnh SQL>

8



Tạo Trigger


Cú pháp:

Create Trigger <Tên Trigger>
On <Tên Bảng/Tên View>
For |After|Instead of [DELETE, INSERT, UPDATE]
AS <Câu lệnh SQL>

Trong đó:

- **For |After:** Trigger được gọi sau khi thao tác Insert/Update/Delete tương ứng đã được thực hiện thành công
 - Các dòng mới được thêm vào đồng thời chứa trong bảng dữ liệu và bảng Inserted
 - Các dòng bị xóa chỉ nằm trong bảng Deleted
- Có thể xử lý quay lui các thao tác đã thực hiện bằng lệnh roll transaction

9



Tạo Trigger

Cú pháp:

Create Trigger <Tên Trigger>
On <Tên Bảng/Tên View>
For |After|Instead of [DELETE, INSERT, UPDATE]
AS <Câu lệnh SQL>

Trong đó:

- **Instead of:** Trigger được gọi thay cho thao tác Insert/Update/Delete tương ứng
 - Các dòng mới được thêm vào chỉ chứa bảng Inserted
 - Các dòng bị xóa đồng thời chứa trong bảng dữ liệu và bảng Deleted

10



Tạo Trigger

Lưu ý:

- ❖ **Inserted** và **Deleted** là hai bảng cục bộ trong bộ nhớ chính có cấu trúc giống bảng mà Trigger định nghĩa trên đó và chỉ tồn tại trong thời gian Trigger xử lý.
- ❖ **Inserted** và **Deleted** chứa hình ảnh của dữ liệu trước và sau khi cập nhật.
- ❖ Dữ liệu ở trong bảng sẽ không bị ảnh hưởng bởi phép toán cập nhật nếu nó không có trong bảng **Inserted** và **Deleted**
- ❖ **Inserted**: chứa các dòng vừa mới được Insert/update vào bảng
- ❖ **Deleted**: Chứa các dòng vừa mới bị xóa khỏi bảng bởi thao tác update/delete

11



Tạo Trigger

❖ Nội dung của 2 bảng Inserted và Deleted

Kiểu Trigger	Inserted Table	Deleted Table
UPDATE	Lưu trữ bản sao của các bản ghi được cập nhật khi câu lệnh kết thúc	Lưu trữ những bản ghi trước khi cập nhật
DELETE	Không sử dụng	Lưu trữ những bản ghi bị xóa
INSERT	Lưu trữ những bản sao của những bản ghi được thêm.	Không sử dụng

12



INSERT Trigger

- ❖ Thực hiện bất cứ khi nào có sự thêm dữ liệu vào bảng
- ❖ Cách thực hiện của INSERT trigger:
 - Thêm dữ liệu vào Inserted table.
 - Kiểm tra dữ liệu trong Inserted table, để xác định xem nó có hợp lệ không.
 - Nếu hợp lệ thì thêm dữ liệu từ Inserted table vào trigger table.

13



INSERT Trigger (tt)

- ❖ Ví dụ 1: Tạo Trigger cho phép khi thêm mới một bản ghi trong bảng CHITIETDATHANG thì trường TonKho trong bảng MATHANG thay đổi tương ứng.

```
CREATE TRIGGER ThemChiTietDatHang
ON CHITIETDATHANG
FOR INSERT
AS
UPDATE MATHANG
SET TonKho=TonKho - inserted.SoLuong
FROM MATHANG
INNER JOIN inserted ON
MATHANG.MaHang=inserted.MaHang
```

14



INSERT Trigger (tt)

❖ Cách khác:

```
CREATE TRIGGER ThemChiTietDatHang
ON CHITIETDATHANG
FOR INSERT
AS
BEGIN
    Declare @ma nvarchar(50)
    Declare @s int
    set @ma = (SELECT MaHang FROM inserted)
    set @s = (SELECT SoLuong FROM inserted)
    UPDATE MATHANG
    SET TonKho=TonKho - @s
    WHERE MaHang=@ma
END
```



INSERT Trigger (tt)

- ❖ Ví dụ 2: Tạo trigger cho phép khi thêm mới một bản ghi trong bảng CHITIETDATHANG thì có kiểm tra số lượng nhập vào trong bảng CHITIETDATHANG phải \leq giá trị tại trường TonKho trong bảng MATHANG nếu không thì thông báo lỗi.

```
CREATE TRIGGER KTChiTietDatHang ON CHITIETDATHANG
FOR INSERT
```

```
AS
```

```
IF (SELECT TonKho from MATHANG INNER JOIN inserted ON
MATHANG.MaHang=inserted.MaHang) - (SELECT SoLuong
from inserted) <0
```

```
BEGIN
```

```
    PRINT N'Hàng trong kho không đủ cung cấp'
```

```
    ROLLBACK TRAN
```

```
END
```




```

ALTER TRIGGER ThemChiTietDatHang
ON CHITIETDATHANG
FOR INSERT
AS
BEGIN
    Declare @ma nvarchar(50)
    Declare @s int, @tk int
    set @ma = (SELECT MaHang FROM inserted)
    set @s = (SELECT SoLuong FROM inserted)
    set @tk = (SELECT tonkho FROM MATHANG WHERE MaHang
= @MA)
    IF @tk - @s < 0
        BEGIN
            RAISERROR ('Hàng trong kho không đủ cung cấp', 1, 2)
            ROLLBACK TRAN
        END
    ELSE
        BEGIN
            UPDATE MATHANG
            SET TonKho = TonKho - @s
            WHERE MaHang = @ma
        END
    END
END

```

17



INSERT Trigger (tt)

Ví dụ 3:

Create Trigger T_KTThem2 **ON** Nhacc **FOR INSERT**

AS

```

Begin
    declare @t char(20)
    declare @count int
    set @t=(select tencc from inserted)
    set @count=(select COUNT(*) from NhaCC
where TenCC=@t)
    If @count>1
        Begin
            print ('trung ten nha cung cap')
            rollback tran
        End
    Return
End

```

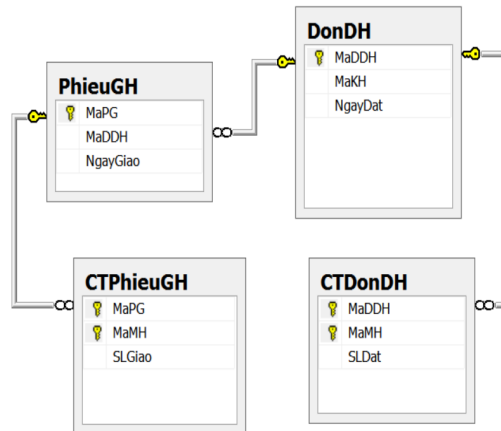
18



INSERT Trigger (tt)

❖ Ví dụ 4

- **Yêu cầu:** Viết Trigger để kiểm tra việc thêm dữ liệu vào bảng PhieuGH theo yêu cầu Ngày giao hàng phải sau ngày đặt hàng và không trễ quá 1 tháng kể từ ngày đặt hàng.



19



INSERT Trigger (tt)

Ví dụ 4 (tt)

Create trigger HD_PGH ON PhieuGH

After Insert

As

If exists (Select * From Inserted I, DonDH D Where
I.MaDDH = D.MaDDH And (D.NgayDat > I.NgayGiao
Or Datediff(Month, D.NgayDat, I.NgayGiao) > 1))

BEGIN

Print (N'Ngày giao hàng không hợp lệ')

Rollback transaction

END

Vì vi phạm RBTV nên phải hủy bỏ mọi thay đổi.
trở về trạng thái ban đầu khi chưa Insert.

20



UPDATE Trigger

- ❖ Thực hiện bất cứ khi nào có sự cập nhật dữ liệu trong bảng.
- ❖ Cách thực hiện của UPDATE trigger:
 - Chuyển những dòng dữ liệu cũ (trước khi cập nhật) vào Deleted table.
 - Thêm những dòng có giá trị mới vào Inserted table, và Trigger table.
 - Kiểm tra lại giá trị ở trong Deleted và Inserted tables nếu có bất cứ yêu cầu liên quan nào.

21



UPDATE Trigger

Ví dụ 1: Khi nhập thêm hoặc cập nhật thì sẽ kiểm tra nếu số lượng mặt hàng đã >10 thì thông báo không cho nhập thêm.

Create trigger T_KTSLMatHang On chuyenhang
For INSERT, UPDATE

AS

Begin

declare @ms char(5) = (select macc from inserted)

declare @d int

set @d=(select COUNT(*) from chuyenhang where

Macc=@ms)

if @0

begin

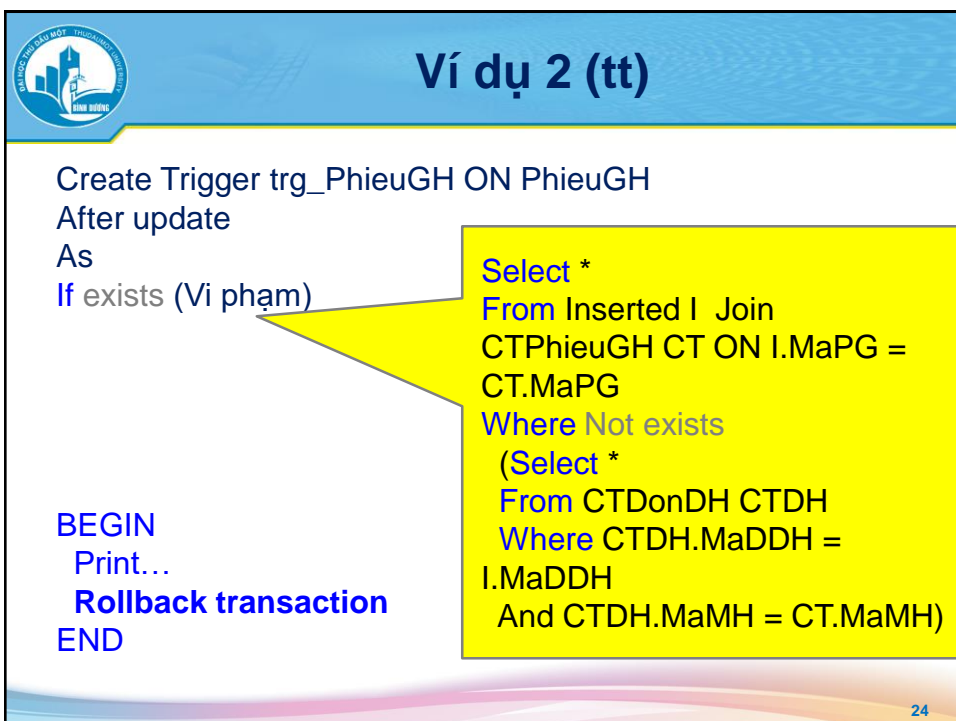
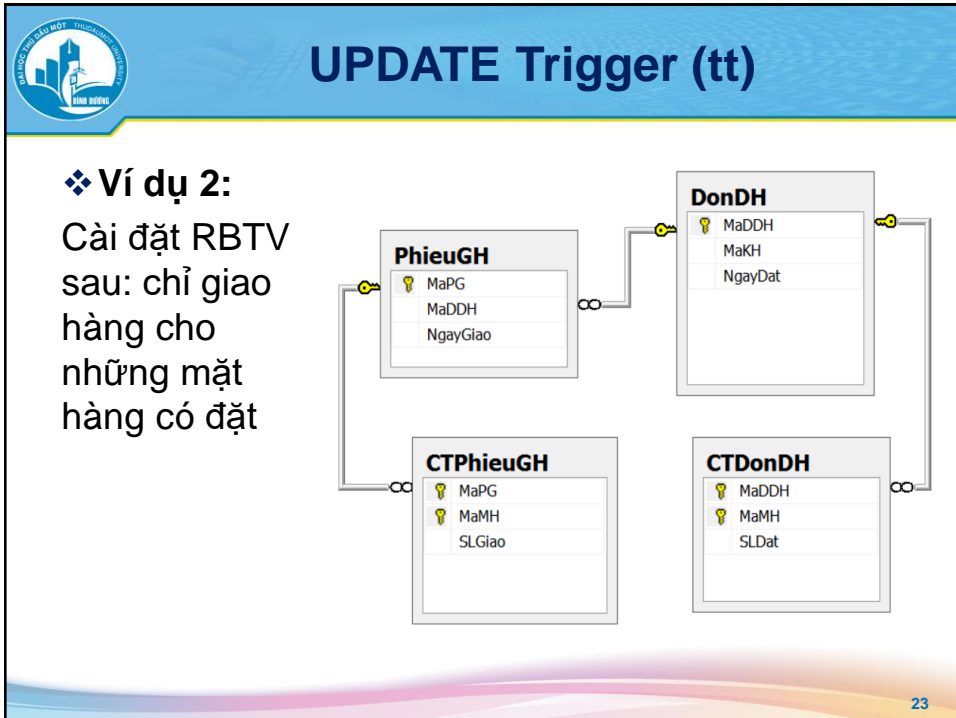
print (N'số lượng hàng của nhà cung cấp này đã đủ')

rollback tran

end

End

22





Các loại trigger

❖ SQL Server có hai loại Trigger

➤ Trigger thông thường: AFTER (FOR) Trigger

- ✓ Chạy sau các hành động kiểm tra dữ liệu của các Rule, Constraint
- ✓ Dữ liệu đã bị tạm thời thay đổi trong bảng

➤ INSTEAD OF Trigger

- ✓ Chạy trước các hành động kiểm tra dữ liệu
- ✓ Dữ liệu chưa hề bị thay đổi
- ✓ Có thể thay thế hành động cập nhật dữ liệu bằng các hành động khác



Xóa trigger

❖ Xóa trigger bằng Enterprise Manager

❖ Xóa bằng T_SQL

DROP TRIGGER <Tên_trigger>



Sửa nội dung trigger

- ❖ Sửa nội dung bằng Enterprise Manager
- ❖ Sửa nội dung bằng T_SQL

```
ALTER    TRIGGER <Tên_Trigger> ON <Tên_bảng>
FOR INSERT [, UPDATE [,DELETE ]]
AS
[DECLARE    Biến_cục_bộ]
    Các_lệnh
```



Trigger lồng nhau

- ❖ Trigger có thể lồng nhau
 - Hành động cập nhật (INSERT, UPDATE, DELETE) → Trigger → Cập nhật bảng khác → Trigger trên bảng tương ứng
- ❖ Số cấp lồng tối đa 32 cấp
- ❖ Cấu hình cho phép/không cho phép trigger lồng nhau: (Mặc định trigger được phép lồng)
 - EXEC sp_configure 'nested triggers', [0 | 1]



Trigger lồng nhau

❖ Ví dụ:

➤ Trigger lồng nhau:

- ✓ Cho các bảng dữ liệu A(A1,A2) B(B1,B2) C(C1,C2)
- ✓ Trong đó các cột A1,A2,B1,B2,C1,C2 có kiểu dữ liệu là int.

➤ Viết Trigger cho hành động insert 1 bản ghi vào bảng A sẽ insert bản ghi đó vào bảng B.

➤ Viết Trigger cho hành động insert 1 bản ghi vào bảng B sẽ insert bản ghi đó vào bảng C.

➔ Như vậy khi thêm 1 bản ghi vào bảng A sẽ gọi Trigger thứ 1 để thêm dữ liệu vào bảng B. Khi đó trigger thứ 2 được tự động gọi.



Trigger lồng nhau

```
CREATE TRIGGER TG1 ON A FOR INSERT AS
BEGIN
    DECLARE @A1 INT SET @A1=(SELECT A1 FROM INSERTED)
    DECLARE @A2 INT SET @A2=(SELECT A2 FROM INSERTED)
    INSERT INTO B(B1,B2) VALUES(@A1,@A2)
END
```

```
CREATE TRIGGER TG2 ON B FOR INSERT AS
BEGIN
    DECLARE @B1 INT SET @B1=(SELECT B1 FROM INSERTED)
    DECLARE @B2 INT SET @B2=(SELECT B2 FROM INSERTED)
    INSERT INTO C(C1,C2) VALUES(@B1,@B2)
END
```



TRANSACTIONS



Giới thiệu

- ❖ Một giao tác (transaction) là chuỗi một hoặc nhiều câu lệnh SQL được kết hợp lại với nhau thành một khối công việc.
- ❖ Các câu lệnh SQL xuất hiện trong giao tác thường có mối quan hệ tương đối mật thiết với nhau và thực hiện các thao tác độc lập.
- ❖ Việc kết hợp các câu lệnh lại với nhau trong một giao tác nhằm đảm bảo tính toàn vẹn dữ liệu và khả năng phục hồi dữ liệu.
- ❖ Trong một giao tác, các câu lệnh có thể độc lập với nhau nhưng tất cả các câu lệnh trong một giao tác đòi hỏi hoặc phải thực thi trọn vẹn hoặc không một câu lệnh nào được thực thi.



Giới thiệu (tt)

- ❖ Giao tác được chia làm 2 loại: Tường minh và không tường minh.
- ❖ Giao tác không tường minh:
 - Mỗi câu lệnh coi như một transaction: INSERT, UPDATE, DELETE
 - Sau khi thực hiện lệnh, các thay đổi dữ liệu sẽ được cập nhật ngay vào CSDL.

33



Giới thiệu (tt)

- ❖ Giao tác tường minh: là giao tác phải được khai báo bằng từ khóa bắt đầu:
 - **Begin Transaction [Tran_name]**
 - **Các câu lệnh theo sau thuộc vào giao tác đã khai báo.**
 - **Kết thúc giao tác:**
 - ✓ Quá trình thực hiện lệnh nếu bị lỗi có thể hủy bỏ giao tác bằng lệnh **Rollback Tran**
 - Dữ liệu nếu có thay đổi trong giao tác sẽ bị hủy bỏ về trạng thái ban đầu khi chưa thực hiện giao tác
 - ✓ Khi giao tác đã hoàn tất, kết thúc chuyển tác bằng **Commit Tran**
 - Dữ liệu thay đổi sẽ được lưu lại

34



Mô hình giao tác trong SQL

- ❖ Giao tác SQL được định nghĩa dựa trên các câu lệnh xử lý giao tác sau đây:
 - **BEGIN TRAN[SACTION] [Tran name]**: Bắt đầu một giao tác
 - **SAVE TRAN[SACTION] <Mark name>**: Đánh dấu một vị trí trong giao tác (gọi là điểm đánh dấu).
 - **ROLLBACK TRAN[SACTION] [<mark name>]**: Quay lui trở lại đầu giao tác hoặc một điểm đánh dấu trước đó trong giao tác.
 - **COMMIT TRAN[SACTION] [Tran name]**: Đánh dấu điểm kết thúc một giao tác. Khi câu lệnh này thực thi cũng có nghĩa là giao tác đã thực hiện thành công.

35



Ví dụ 1: Giao tác hoàn tất

```
SELECT COUNT(*) AS N'Tổng số mặt hàng trước khi thêm' FROM MATHANG
BEGIN TRAN
INSERT INTO MATHANG VALUES('AS01',N'Laptop Asus A250','DQV', 'DT',
1000, N'Cái', 13000000)
SELECT COUNT(*) AS N'Tổng số mặt hàng sau khi thêm trong giao tác'
FROM MATHANG
COMMIT TRAN
SELECT COUNT(*) AS N'Tổng số mặt hàng hiện tại' FROM MATHANG
```

Kết quả:

```
Tổng số mặt hàng trước khi thêm
23
Tổng số mặt hàng sau khi thêm trong giao tác
24
Tổng số mặt hàng hiện có
24
```

36



Ví dụ 2: Giao tác bị hủy

```
SELECT COUNT(*) AS N'Tổng số mặt hàng trước khi thêm' FROM MATHANG
BEGIN TRAN
INSERT INTO MATHANG VALUES('AS01',N'Laptop Asus A250','DQV', 'DT',
1000, N'Cái', 13000000)
SELECT COUNT(*) AS N'Tổng số mặt hàng sau khi thêm trong giao tác'
FROM MATHANG
ROLLBACK TRAN
SELECT COUNT(*) AS N'Tổng số mặt hàng hiện tại' FROM MATHANG
```

Kết quả:

```
Tổng số mặt hàng trước khi thêm
23
Tổng số mặt hàng sau khi thêm trong giao tác
24
Tổng số mặt hàng hiện có
23
```

37



Ví dụ 3: Giao tác thành công nhưng bị bỏ qua lệnh ở giữa

```
BEGIN TRANSACTION giaotac3
UPDATE diemthi SET diemlan2=0 WHERE diemlan2 IS
NULL
SAVE TRANSACTION a
UPDATE monhoc SET sodvht=4 WHERE sodvht=3
ROLLBACK TRANSACTION a
UPDATE monhoc SET sodvht=2 WHERE sodvht=3
COMMIT TRANSACTION giaotac3
```

38



Ví dụ 4: Giao tác bị hủy

BEGIN TRANSACTION giaotac4

UPDATE diemthi SET diemlan2=0 WHERE diemlan2 IS NULL

SAVE TRANSACTION a

UPDATE monhoc SET sodvht=4 WHERE sodvht=3

ROLLBACK TRANSACTION giaotac4

UPDATE monhoc SET sodvht=2 WHERE sodvht=3

COMMIT TRANSACTION giaotac4

39



Ví dụ 5: Giao tác lồng nhau

CREATE PROC sp_TranEx(@a INT,@b INT)

AS

BEGIN

BEGIN TRANSACTION T1

IF NOT EXISTS (SELECT * FROM T WHERE A=@A)

INSERT INTO T VALUES(@A,@B)

IF NOT EXISTS (SELECT * FROM T WHERE A=@A+1)

INSERT INTO T VALUES(@A+1,@B+1)

COMMIT TRANSACTION T1

END

BEGIN TRANSACTION

EXECUTE sp_tranex 20,40

SAVE TRANSACTION a

EXECUTE sp_tranex 30,60

ROLLBACK TRANSACTION a

EXECUTE sp_tranex 40,80

COMMIT TRANSACTION

	A	B
	20	40
	21	41
	40	80
	41	81

40



Tài liệu tham khảo

- ❖ Slide Bài giảng Hệ quản trị Cơ sở dữ liệu, khoa CNTT Đại học Thủ Dầu Một
- ❖ Mike Chapple, Microsoft SQL Server 2012 for Dummies, Wiley, 2013.
- ❖ Tạ Thị Thu Phượng, Hệ quản trị cơ sở dữ liệu (Bài giảng tóm tắt), Đại học Đà Lạt.

Bài giảng Hệ quản trị CSDL

41



Phone: 0650. 3834930

Website: www.fit.tdmu.edu.vn