

CHƯƠNG 3

LẬP TRÌNH TRÊN SQL-SERVER





NỘI DUNG

3.1. Các kiểu DL trong SQL Server

3.2. Các hàm trong SQL Server

3.3 T-SQL Programming

3.4. Thủ tục và hàm người dung

3.5. Trigger

3.6. Transactions và Locks



3.1. CÁC KIỂU DL TRONG SQL SERVER



Kiểu dữ liệu trong SQL-SERVER

- ❖ Kiểu dữ liệu chỉ định kiểu của dữ liệu và dung lượng có thể lưu trữ của một đối tượng
- ❖ SQL Server hỗ trợ một số kiểu dữ liệu được cài đặt sẵn như sau:



Kiểu dữ liệu trong SQL-SERVER

Kiểu dữ liệu	Kích thước	Miền giá trị dữ liệu lưu trữ
> Các kiểu dữ liệu dạng số nguyên		
Int ✓	4 bytes	từ -2,147,483,648 đến +2,147,483,647
SmallInt `	2 bytes	từ -32768 đến +32767
TinyInt	1 byte	từ 0 đến 255
Bit	1 byte	0, 1 hoặc Null
> Các kiểu dữ liệu dạng số thập phân		
Decimal, Numeric	17 bytes	từ -10^{38} đến $+10^{38}$
> Các kiểu dữ liệu dạng số thực		
Float	8 bytes	từ $-1.79E+308$ đến $+1.79E+308$
Real	4 bytes	từ $-3.40E+38$ đến $+3.40E+38$



Kiểu dữ liệu trong SQL-SERVER

> Các kiểu dữ liệu dạng chuỗi có độ dài cố định

Char	N bytes	từ 1 đến 8000 ký tự, mỗi ký tự là một byte
------	---------	--

> Các kiểu dữ liệu dạng chuỗi có độ dài biến đổi

VarChar	N bytes	từ 1 đến 8000 ký tự, mỗi ký tự là 1 byte
---------	---------	--

> Các kiểu dữ liệu dạng chuỗi dùng font chữ Unicode

NChar	2*N bytes	từ 1 đến 4000 ký tự, mỗi ký tự là 2 bytes
-------	-----------	---

NVarChar	2*N bytes	từ 1 đến 4000 ký tự, mỗi ký tự là 2 bytes
----------	-----------	---



Kiểu dữ liệu trong SQL-SERVER

> Các kiểu dữ liệu dạng tiền tệ

Money	8 bytes	từ -922,337,203,685,477.5808 đến +922,337,203,685,477.5807
SmallMoney	4 bytes	từ -214,748.3648 đến + 214,748.3647

> Các kiểu dữ liệu dạng ngày và giờ

DateTime	8 bytes	từ 01/01/1753 đến 31/12/9999
SmallDateTime	4 bytes	từ 01/01/1900 đến 06/06/2079

> Các kiểu dữ liệu dạng chuỗi nhị phân (Binary String)

Binary	N bytes	từ 1 đến 8000 bytes
VarBinary	N bytes	từ 1 đến 8000 bytes
Image	N bytes	từ 1 đến 2,147,483,647 bytes



Biến trong T-SQL



Khái niệm

- Gói lệnh (Batch): tập các câu lệnh T-SQL liên tiếp nằm giữa 2 lệnh GO
 - Các lệnh trong một gói lệnh sẽ được gửi cùng lúc bởi ứng dụng đến SQL Server
 - SQL server sẽ thực hiện cùng lúc các lệnh trong cùng 1 batch



Khái niệm (tiếp)

- Biến trong T-SQL là một đối tượng có thể lưu trữ một giá trị dữ liệu.
- Có 2 loại biến:
 - Biến cục bộ
 - Biến toàn cục



Biến cục bộ

- ❖ Biến là một đối tượng để chứa dữ liệu
- ❖ Gọi là biến cục bộ vì phạm vi hoạt động của biến chỉ nằm trong một thủ tục, một hàm hoặc một lô có chứa lệnh khai báo biến đó.
- ❖ Một lô lệnh (batch) là các câu lệnh liên tiếp nhau và được kết thúc bằng từ khóa GO:
 - Những câu lệnh nằm trong cùng batch sẽ được xử lý cùng lúc
 - Lệnh GO dùng để gửi tín hiệu đến cho SQL SERVER biết đã kết thúc một batch job và yêu cầu thực thi.



Biến cục bộ (local variable)

- Biến cục bộ được tạo và dùng để lưu trữ các giá trị tạm thời trong phạm vi tính toán.
 - Biến phải có kiểu dữ liệu
 - Tên của biến phải bắt đầu với dấu '@'
 - Được khai báo bên trong một thủ tục, hàm, batch
 - Phạm vi hoạt động của biến từ vị trí khai báo đến khi kết thúc thủ tục, hàm hay batch



Khai báo biến cục bộ

- Cú pháp:

DECLARE @Tên_biến [**AS**] Kiểu_dữ_liệu [,...]

- Từ khóa 'AS' không bắt buộc

- Các biến cách nhau bởi dấu phẩy

- Ví dụ:

```
DECLARE @TongDiem AS Real, @DiemTB as Real  
DECLARE @NgayDatHang DATE
```



Khai báo biến cục bộ

■Cú pháp:

DECLARE @Tên_biến [**AS**] Kiểu_dữ_liệu [,...]

- Từ khóa 'AS' không bắt buộc
- Các biến cách nhau bởi dấu phẩy
- Các kiểu dữ liệu text, ntext hoặc image không được chấp nhận khi khai báo biến

■Ví dụ:

```
DECLARE @TongDiem AS Real, @DiemTB as Real  
DECLARE @NgayDatHang DATE
```



- ❖ Chú ý: kiểu dữ liệu text, ntext hoặc image không được chấp nhận khi khai báo biến
- ❖ Từ khóa SET hay SELECT được dùng để gán giá trị cho biến
- ❖ Cú pháp

`SET @tên_biến = giá_trị`

hoặc

`SELECT @tên_biến = giá_trị`

Ví dụ:

`SET @hoten=N'Nguyễn Thị Thái'`



Gán giá trị cho biến

- Bảng từ khóa SET hoặc bằng câu lệnh SELECT:

SET @Tên_biến = Giá_trị

SELECT @Tên_biến = Giá_trị

SELECT @Tên_biến = Tên_cột

FROM Tên_bảng

WHERE Điều_kiện



Gán giá trị cho biến (tiếp)

■ Ví dụ:

```
DECLARE @MaSVCanTim integer;  
SET @MaSVCanTim = 4;
```

```
DECLARE @hoten nvarchar(50);  
SELECT @hoten = HoTen  
from SinhVien  
where MaSV = @MaSVCanTim;
```



Xem giá trị hiện hành của biến

- Để hiển thị giá trị của biến:

PRINT @Tên_biến

PRINT @HoTen

- Khi hiển thị kết hợp với chuỗi, phải đổi kiểu dữ liệu sang kiểu chuỗi bằng hàm CAST hay CONVERT



Xem giá trị hiện hành của biến (tiếp)

```
DECLARE @NgaySinh date;  
SET @NgaySinh = '09/14/1983';  
PRINT N'Ngày sinh là ' + convert(char(12),@NgaySinh);  
  
DECLARE @MaxDiem as float;  
SELECT @MaxDiem = max(Diem)  
FROM KETQUA, SinhVien  
Where KETQUA.MaSV = SinhVien.MaSV  
and SinhVien.HoTen = N'Nguyễn Văn A';  
  
PRINT N'Điểm cao nhất là: ' + cast(@MaxDiem as char(4));
```



Phạm vi hoạt động của biến

- Một biến cục bộ chỉ có phạm vi hoạt động cục bộ trong một thủ tục, hàm, trigger hay batch.

```
DECLARE @MaxDiem as float;  
SELECT @MaxDiem = max(Diem)  
FROM KETQUA, SinhVien  
Where KETQUA.MaSV = SinhVien.MaSV  
and SinhVien.HoTen = N'Nguyễn Văn A';  
  
GO  
PRINT N'Điểm cao nhất là: ' + cast(@MaxDiem as char(4));  
GO
```

Lỗi vì chưa khai báo biến @MaxDiem trong batch



Biến toàn cục (Global Variables)

- Biến là biến được định nghĩa sẵn bởi hệ thống
 - Tên của biến phải bắt đầu với '@@'
 - Không thể gán giá trị cho biến toàn cục
 - Biến toàn cục không có kiểu
 - Ví dụ:
 - @@VERSION: phiên bản của SQL Server
- SELECT @@VERSION



Biến toàn cục (Global Variables)

- **@@SERVERNAME**: tên server
`SELECT @@ SERVERNAME`
- **@@ERROR**: trả về số thứ tự lỗi của lệnh thực thi sau cùng, nếu trả về 0 thì câu lệnh hoàn thành
- **@@ROWCOUNT**: trả về số dòng bị ảnh hưởng bởi lệnh thực thi gần nhất

```
SELECT * from SinhVien;
```

```
PRINT N'Số dòng tìm thấy là ' + convert(char(4),@@Rowcount);
```



3.2. HÀM TRONG SQL SERVER



Toán tử trong T-SQL



Toán tử số học

Ký hiệu	Ý nghĩa
+	Thực hiện phép cộng hai số
-	Thực hiện phép trừ hai số.
*	Thực hiện phép nhân hai số.
/	Thực hiện phép chia hai số.
%	Thực hiện phép chia lấy phần dư.



Toán tử nối chuỗi

- Sử dụng dấu '+' làm toán tử nối chuỗi

```
Declare @HoTen nvarchar(50)  
SET @HoTen = N'Nguyễn Văn A'  
PRINT N'Xin chào ' + @HoTen
```

```
Select N'Ngày sinh là: ' + cast(NgaySinh as char(12))  
From SinhVien  
Where Hoten = @HoTen
```



Toán tử so sánh

Ký hiệu	Ý nghĩa
=	Thực hiện phép so sánh bằng.
>	Thực hiện phép so sánh lớn hơn.
<	Thực hiện phép so sánh nhỏ hơn.
>=	Thực hiện phép so sánh lớn hơn hoặc bằng.
<=	Thực hiện phép so sánh nhỏ hơn hoặc bằng.
<>	Thực hiện phép so sánh khác.
!=	Thực hiện phép so sánh khác.
!>	Thực hiện phép so sánh không lớn hơn.
!<	Thực hiện phép so sánh không nhỏ hơn.



Toán tử logic

- Sử dụng các toán tử thông thường: AND, OR, NOT

```
Select HoTen, NgaySinh  
from SinhVien, KETQUA  
where SinhVien.MaSV = KETQUA.MaSV  
AND (KETQUA.Diem >=8 OR KETQUA.Diem <=5)
```



Thứ tự ưu tiên các toán tử

■ Từ cao đến thấp

Kiểu toán tử	Ký hiệu
Nhóm	()
Nhân, chia số học	*,/,%
Cộng trừ số học	- +
Nối chuỗi	+
Luận lý NOT	NOT
Luận lý AND	AND
Luận lý OR	OR



Các ký tự đại diện trong T-SQL



Các ký tự đại diện

Ký tự đại diện	Mô tả
_ (dấu gạch chân)	Một ký tự đơn
%	Chiều dài bất kỳ một chuỗi
[]	Một ký tự đơn trong phạm vi một cặp dấu ngoặc vuông
[^]	Nhiều ký tự đơn mà không nằm trong phạm vi cặp dấu ngoặc vuông



Các ký tự đại diện (tiếp)

Ký tự	Ví dụ
_ (dấu gạch chân)	<pre>select * from SinhVien where HoTen like N'_ương %'</pre>
%	<pre>select * from SinhVien where HoTen like N'Nguyễn%'</pre>
[]	<pre>select * from SinhVien where HoTen like N'[LN]%'</pre>
[^]	<pre>select * from SinhVien where HoTen like N'^L]%'</pre>



Các ký tự đại diện (tiếp)

Ví dụ: Tìm các sinh viên trong bảng SinhVien có HoTen chứa chữ cái đầu là L hoặc N, và chữ cái thứ 3 không phải là u

```
select * from SinhVien  
where HoTen like N'[NL]_[^u]%'
```



3.2. HÀM TRONG T-SQL



1. Hàm tổng hợp thống kê



Hàm tổng hợp thống kê

- Các hàm tập hợp (aggregate functions) tạo ra các giá trị tổng hợp cho kết quả truy vấn
 - SUM()
 - MIN()
 - MAX()
 - AVG()
 - COUNT()



Hàm tổng hợp thống kê

■ SUM([DISTINCT] Biểu_thức)

- Trả về tổng tất cả các giá trị của trường dữ liệu trong Biểu_thức
- Chỉ dùng được với dữ liệu kiểu số, bỏ qua giá trị NULL
- Có thể dùng DISTINCT với SUM để tính tổng cho các giá trị duy nhất của trường dữ liệu trong Biểu_thức

```
Select SUM(Distinct Diem)
From SinhVien,KETQUA
Where SinhVien.HoTen = N'Nguyễn Văn A'
And SinhVien.MaSV = KETQUA.MaSV
```



Hàm tổng hợp thống kê

■ **AVG([DISTINCT] Biểu_thức)**

- Trả về giá trị trung bình của tất cả các giá trị của trường dữ liệu trong Biểu_thức
- Chỉ dùng được với dữ liệu kiểu số, bỏ qua giá trị NULL

```
declare @DiemTB float
Select @DiemTB = AVG(Diem)
From SinhVien,KETQUA
Where SinhVien.HoTen = N'Nguyễn Văn A'
And     SinhVien.MaSV = KETQUA.MaSV
```



Hàm tổng hợp thống kê

■ COUNT([DISTINCT] Biểu_thức)

- Đếm các giá trị khác NULL trong biểu thức
- Có thể dùng với các trường số và ký tự
- Có thể dùng Count(*) để đếm tất cả các bản ghi

```
Select count(*)  
from SinhVien, KETQUA  
Where SinhVien.MaSV = KETQUA.MaSV  
AND SinhVien.HoTen = N'Nguyễn Văn A'
```



Hàm tổng hợp thống kê

- MAX(Biểu_thức)
 - Trả về giá trị lớn nhất trong biểu thức
 - Có thể dùng với các trường số, chuỗi và ngày tháng
 - Bỏ qua giá trị NULL

```
Select MAX(Diem)
from SinhVien, KETQUA
Where SinhVien.MaSV = KETQUA.MaSV
AND SinhVien.HoTen = N'Nguyễn Văn A'
```




Hàm tổng hợp thống kê

- MIN(Biểu_thức)
 - Trả về giá trị nhỏ nhất trong biểu thức
 - Có thể dùng với các trường số, chuỗi và ngày tháng
 - Bỏ qua giá trị NULL

```
Select MIN(NgaySinh)  
from SinhVien
```



2. HÀM XỬ LÝ CHUỖI



2. Các hàm xử lý chuỗi

- **ASCII()** : trả về giá trị mã ASCII của ký tự bên trái của chuỗi

```
Print ASCII('TOI')
```

```
Print ASCII('TO')
```

hai lệnh trên cùng trả về kết quả là mã 84

- **Char()** : chuyển đổi mã ASCII từ số nguyên sang dạng chuỗi

```
Print char(84)
```

trả về ký tự 'T'



2. Các hàm xử lý chuỗi (tiếp)

- **UPPER()** : chuyển đổi chuỗi sang kiểu chữ hoa

```
Print UPPER(N'Nguyễn Văn A')
```

Trả về:

NGUYỄN VĂN A

- **LOWER()** : chuyển đổi chuỗi sang kiểu chữ thường

```
PRINT LOWER(N'Nguyễn Văn A')
```

Trả về:

nguyễn văn a



2. Các hàm xử lý chuỗi (tiếp)

- **Len()** : trả về chiều dài của chuỗi

```
Print Len(N'Nguyễn Văn A')
```

Trả về:

12

- **CHARINDEX()** : trả về vị trí ký tự bắt đầu của chuỗi con trong chuỗi đang xét

```
Print CHARINDEX(N'Văn', N'Nguyễn Văn A')
```

Trả về:

8



2. Các hàm xử lý chuỗi (tiếp)

- **LTRIM()** : loại bỏ khoảng trắng bên trái của chuỗi

```
Print CharIndex(LTrim(N'   Văn'), N'Nguyễn Văn A')
```

Trả về:

- **RTRIM()** : loại bỏ khoảng trắng bên phải của chuỗi

```
Print RTrim(N'Nguyễn Văn A      ') + ' --'
```

Trả về:

Nguyễn Văn A--



2. Các hàm xử lý chuỗi (tiếp)

- **Left(chuỗi,n)** : trả về chuỗi bên trái tính từ đầu chuỗi cho đến vị trí thứ n

```
Print '--' + Left(N'Nguyễn Văn A',7) + '--'
```

Trả về: --Nguyễn --

- **Right(chuỗi,n)** : Trả về chuỗi bên phải tính từ cuối cho đến vị trí thứ n

```
Print '--' + Right(N'Nguyễn Văn A',6) + '--'
```

Trả về: -- Văn A--



2. Các hàm xử lý chuỗi (tiếp)

- **Ví dụ :** viết câu lệnh chọn riêng phần Họ từ trường HoTen của bảng SinhVien

`select LEFT(LTRIM(HoTen), CharIndex(' ', LTRIM(HoTen))-1)
from Sinhvien`

- **Ví dụ :**

```
declare @a nvarchar(50);  
set @a = N'    Nguyễn    Văn A'  
print '--' + LEFT(LTRIM(@a), CharIndex(' ', LTRIM(@a))-1) + '--'
```

Trả về

--Nguyễn--



3. HÀM XỬ LÝ VỚI THỜI GIAN



Các hàm ngày tháng

- ❖ **Getdate()**: cho biết ngày tháng hiện tại
- ❖ **Dateadd(datepart, number, date)**: để thêm ngày/tháng/năm vào date
- ❖ **Datediff(datepart, date1, date2)**: tính khoảng cách của date2-date1
- ❖ **Datename(datepart, date)**: cho biết tên tiếng Anh của ngày tháng



Các hàm xử lý thời gian

- **getDate()** : trả về ngày tháng năm của hệ thống

```
print N'Hôm nay là ngày: ' + cast(GETDATE() as char(20))
```

Trả về:

```
Hôm nay là ngày: Sep 18 2017 7:24AM
```



Các hàm xử lý thời gian (tiếp)

- **DatePart(tham_số, ngày)** : trả về một phần giá trị của một chuỗi dạng ngày tháng đầy đủ

Hàm DATEPART	Tham số
Year	yy, yyyy
Quarter	qq, q
Month	mm, m
Dayofyear	dy, y
Day	dd, d

Hàm DATEPART	Tham số
Week	wk, ww
Weekday	dw
Hour	hh
Minute	mi, n
Second	ss, s
Miliecond	ms



Các hàm xử lý thời gian (tiếp)

■ DatePart(tham_số, ngày) : ví dụ

```
print N'Hôm nay là ngày: ' + cast(datepart(dd,getdate()) as char(2))  
    + N' tháng ' + cast(datepart(mm,getdate()) as char(1))  
    + N' năm ' + cast(datepart(yyyy,getdate()) as char(4))
```

Trả về

Hôm nay là ngày: 18 tháng 9 năm 2017



Các hàm xử lý thời gian (tiếp)

- **DateDiff(tham_số, ngày_đầu, ngày_cuối)** : trả về số ngày trong khoảng thời gian giữa hai ngày

```
declare @ThoiHan as DateTime:  
set @ThoiHan = '04/09/2017'  
print N'Thời hạn đã qua: ' + cast(datediff(dd,@ThoiHan,getdate()) as char(3)) + N' ngày'  
print N'Thời hạn đã qua: ' + cast(datediff(m,@ThoiHan,getdate()) as char(1)) + N' tháng'
```

Trả về

Thời hạn đã qua: 162 ngày

Thời hạn đã qua: 5 tháng



Các hàm xử lý thời gian (tiếp)

- **Day(ngày)** : trả về ngày thứ mấy trong tháng
- **Month(ngày)** : trả về tháng thứ mấy trong năm
- **Year(ngày)** : trả về năm

```
print N'Hôm nay là ngày: ' + cast(day(getdate()) as char(2))  
      + N' tháng ' + cast(month(getdate()) as char(1))  
      + N' năm ' + cast(year(getdate()) as char(4))
```

Trả về

```
Hôm nay là ngày: 18 tháng 9 năm 2017
```



4. HÀM TOÁN HỌC



Các hàm toán học

- **square()** : trả về bình phương của một biểu thức
- **sqrt()** : trả về bình thường của một biểu thức

Ví dụ:

Print square(4)

Kết quả trả về như sau:

16

Ví dụ:

Print sqrt(4)

Kết quả trả về như sau:

2



Các hàm toán học (tiếp)

- **round()** : trả về số làm tròn của một biểu thức

```
print round(78.890766876,4)
print round(78.890766876,0)
print round(78.890766876,2)
print round(78.890766876,1)
```

- Kết quả lần lượt như sau

```
78.8908000000
79.0000000000
78.8900000000
78.9000000000
```



Các hàm chuyển đổi

- **cast(Biểu_thức as kiểu_dữ_liệu)** : trả về giá trị có kiểu dữ liệu theo định nghĩa

```
print cast(getdate() as varchar(11))  
print cast(getdate() as varchar(19))
```

Trả về

```
Sep 18 2017
```

```
Sep 18 2017 8:13AM
```



Các hàm chuyển đổi (tiếp)

- **convert(kiểu_dữ_liệu, biểu_thức)** : chuyển đổi giá trị có kiểu dữ liệu này sang kiểu dữ liệu khác nếu cho phép

Ví dụ:

Print convert(int, '12')

Kết quả trả về là số nguyên có giá trị như sau:

12



KHACH(MAKH, TENKH, THANHPHO, SODIENTHOAI)

NHA(MANHA, TENCHUNHA, DIACHI, GIATHUE)

HOPDONG(SOHD, MAKH, MANHA, NGAYBD, NGAYKT)

- a. Cho biết danh sách khách hàng đã có ít nhất 2 lần làm hợp đồng thuê nhà
- b. Hiển thị danh sách các nhà đang được thuê tính đến thời điểm hiện tại chưa đến hạn trả phòng.