

---

# Chương 5

## LẬP TRÌNH TRÊN SQL SERVER

---

# Mục tiêu

- Sau khi học xong chương này, SV có thể:
  - Biết các kiểu dữ liệu trên SQL Server
  - Nắm vững cú pháp khai báo, gán và xem giá trị của biến
  - Sử dụng biến hệ thống
  - Biết các toán tử trong SQL
  - Sử dụng các cấu trúc điều khiển
  - Sử dụng kiểu dữ liệu Cursor
  - Sử dụng các hàm thường dùng trong SQL

# Nội dung chi tiết

- Các kiểu dữ liệu trên SQL Server
- Khai báo biến
- Gán giá trị cho biến
- Xem giá trị hiện hành của biến
- Biến hệ thống
- Toán tử trong SQL
- Cấu trúc điều khiển
- Kiểu dữ liệu Cursor
- Các hàm thường dùng trong SQL
- Bài tập ứng dụng

# Các kiểu dữ liệu trên SQL Server

## ■ Kiểu số nguyên:

Kiểu dữ liệu	Từ	Tới
bigint	-9,223,372,036,854,775,808	9,223,372,036,854,775,807
int	-2,147,483,648	2,147,483,647
smallint	-32,768	32,767
tinyint	0	255
bit	0	1
decimal	$-10^{38} + 1$	$10^{38} - 1$
numeric	$-10^{38} + 1$	$10^{38} - 1$
money	-922,337,203,685,477.5808	+922,337,203,685,477.5807
smallmoney	-214,748.3648	+214,748.3647

## ■ Kiểu số thực:

Kiểu dữ liệu	Từ	Tới
float	-1.79E + 308	1.79E + 308
real	-3.40E + 38	3.40E + 38

# Các kiểu dữ liệu trên SQL Server

## ■ Kiểu ngày tháng:

Kiểu dữ liệu	Từ	Tới
<b>datetime</b> ( <i>độ chính xác là 3.33 mili giây</i> )	Jan 1, 1753	Dec 31, 9999
<b>Smalldatetime</b> ( <i>độ chính xác là 1 phút</i> )	Jan 1, 1900	Jun 6, 2079
<b>date</b>	Jan 1, 0001	Dec 31, 9999
<b>time</b> ( <i>chính xác tới số thập phân của giây</i> )	00:00:00.0000000	23:59:59.9999999

## ■ Kiểu chuỗi không chứa Unicode:

Kiểu dữ liệu	Mô tả
char(n)	độ dài cố định, tối đa là n ký tự ( $n \leq 8000$ )
varchar(n)	độ dài tùy biến, tối đa là n ký tự ( $n \leq 8000$ )
text	độ dài tùy biến, tối đa là 2.147.483.647 ký tự

# Các kiểu dữ liệu trên SQL Server

- Kiểu chuỗi có chứa Unicode:

Kiểu dữ liệu	Mô tả
nchar(n)	độ dài cố định, tối đa là n ký tự ( $n \leq 4000$ )
nvarchar(n)	độ dài tùy biến, tối đa là n ký tự ( $n \leq 4000$ )
ntext	độ dài tùy biến, tối đa là 1.073.741.823 ký tự

- Kiểu nhị phân:

Kiểu dữ liệu	Mô tả
binary(n)	độ dài cố định, tối đa là n byte ( $n \leq 8000$ )
varbinary(n)	độ dài tùy biến, tối đa là n byte ( $n \leq 8000$ )
image	độ dài tùy biến, tối đa là 2.147.483.647 byte

# Các kiểu dữ liệu trên SQL Server

- Kiểu dữ liệu khác:

Kiểu dữ liệu	Mô tả
sql_variant	lưu giữ các giá trị của các kiểu dữ liệu đa dạng được hỗ trợ bởi SQL Server (ngoại trừ text, ntext, và timestamp)
timestamp	lưu giữ một số duy nhất mà được cập nhật mỗi khi một hàng được cập nhật.
uniqueidentifier	lưu giữ một định danh chung (Globally Unique Identifier - GUID)
xml	lưu giữ dữ liệu XML, có thể lưu giữ XML trong một column hoặc một biến
cursor	tham chiếu tới một đối tượng con trỏ (Cursor)
table	lưu giữ một tập hợp kết quả để xử lý vào lần sau

# Khai báo biến

## ■ Biến cục bộ:

- Là biến do người lập trình khai báo, dùng để lưu trữ các giá trị tạm thời trong quá trình tính toán.
- Biến muốn sử dụng trong 1 batch (lô lệnh) phải được khai báo trước.
- Cú pháp khai báo biến:

**declare @Tên\_biến Kiểu\_dữ\_liệu**

- Trong đó: **Tên\_biến** luôn bắt đầu bằng ký tự **@**, không có khoảng trắng, ký tự đặc biệt, và không bắt đầu là số.
- VD:

```
declare @so_mon_thi int, @ho_ten_SV nvarchar(50)  
declare @ngay_sinh datetime
```



# Gán giá trị cho biến

- Gán giá trị cho biến cục bộ:

- Sử dụng lệnh **set** hoặc **select**:

**set @Tên\_biến = Giá\_trị / Hàm**

- VD:

*set @so\_mon\_thi = 5*

*set @ngay\_sinh = getdate()*

**select @Tên\_biến = Tên\_cột / Hàm  
from Tên\_bảng ...**

- VD:

*select @hocbongtb = avg(HOCBONG)*

*from SINHVIEN*

*where MAKH='TH'*

# Xem giá trị hiện hành của biến

- Lệnh **print**:

**print @Tên\_biến**

- VD1:

*print @so\_mon\_thi*

- Khi có kết hợp với chuỗi, phải đổi kiểu dữ liệu sang kiểu chuỗi bằng hàm **cast** hay **convert**

- VD2:

*print N'Học bổng trung bình là ' + cast(@hocbongtb as varchar(10))*

- Một biến chỉ có phạm vi hoạt động cục bộ trong một Batch (tập hợp nhiều lệnh kết thúc bằng lệnh **go**), Stored Procedure hay Trigger.

# Biến hệ thống

- Là biến (chỉ đọc) do SQL Server định sẵn.
- Cho biết trạng thái của hệ thống (phiên bản SQL Server, số dòng dữ liệu vừa được xử lý bởi câu lệnh, mã lỗi, số lượng kết nối, tình trạng cursor, ...)
- Tên bắt đầu bởi hai ký tự @@.
- Một số biến thường dùng:

Tên Biến	Ý nghĩa
<b>@@Error</b>	Số mã lỗi của câu lệnh thực hiện gần nhất. Khi một câu lệnh thực hiện thành công thì giá trị là 0
<b>@@RowCount</b>	Tổng số mẫu tin (dòng) được tác động của câu lệnh truy vấn gần nhất.
<b>@@Fetch_Status</b>	Trạng thái của việc đọc dữ liệu trong bảng theo cơ chế từng mẫu tin (cursor). Khi đọc dữ liệu của mẫu tin thành công thì giá trị là 0.

# Toán tử trong 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 trong SQL

## ■ Toán tử nối chuỗi:

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

- VD1: 

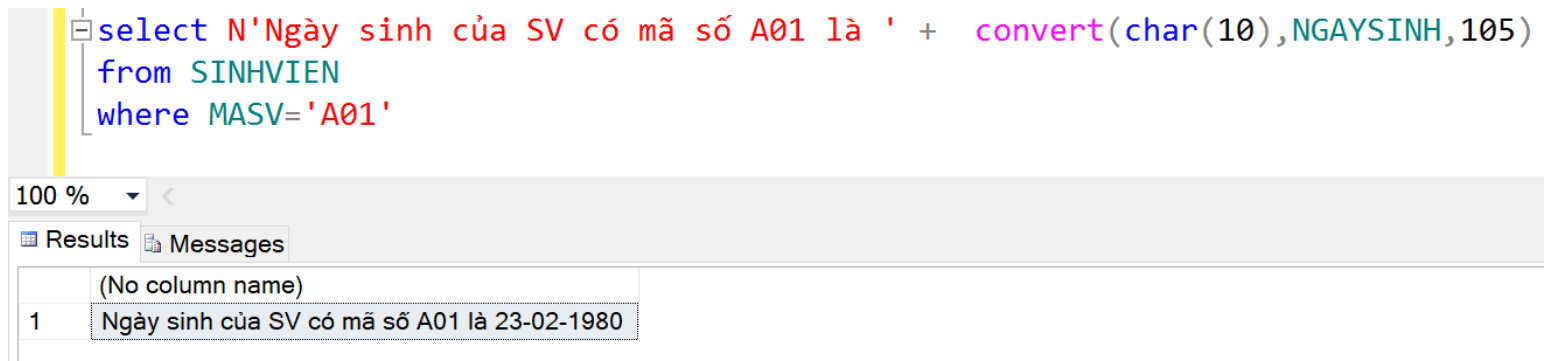
```
select 'Hello'+' '+'World!'
```



The screenshot shows the SQL Server query results for the query `select 'Hello'+' '+'World!'`. The results pane is set to 100% zoom. The 'Results' tab is active, showing a single row with the value 'Hello World!'. The column header is '(No column name)'.

	(No column name)
1	Hello World!

- VD2:



The screenshot shows the SQL Server query results for the query `select N'Ngày sinh của SV có mã số A01 là ' + convert(char(10),NGAYSINH,105) from SINHVIEN where MASV='A01'`. The results pane is set to 100% zoom. The 'Results' tab is active, showing a single row with the value 'Ngày sinh của SV có mã số A01 là 23-02-1980'. The column header is '(No column name)'.

	(No column name)
1	Ngày sinh của SV có mã số A01 là 23-02-1980

# Toán tử trong SQL

## ■ 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 trong SQL

## ■ Toán tử luận lý:

- Sử dụng các toán tử **and**, **or**, **not**.

- VD:

```
select *  
from SINHVIEN  
where (MAKH='TH' and HOCBONG>=50000) or (MAKH='AV' and HOCBONG>=100000)
```

100 % <

Results Messages

	MASV	HOSV	TENSV	PHAI	NGAYSINH	NOISINH	MAKH	HOCBONG
1	A01	Nguyễn Thu	Hải	0	1980-02-23 00:00:00.000	TP.HCM	AV	100000
2	A02	Trần Văn	Chính	1	1982-12-24 00:00:00.000	TP.HCM	TH	100000
3	A03	Lê Thu Bạch	Yến	0	1982-12-12 00:00:00.000	Hà Nội	AV	140000
4	B01	Trần Thanh	Mai	0	1981-12-20 00:00:00.000	Bến Tre	TH	200000
5	B03	Trần Thị	Thanh	0	1982-12-31 00:00:00.000	TP.HCM	TH	50000

# Cấu trúc điều khiển

- Cấu trúc rẽ nhánh if...else....:

```
if <biểu_thức_luận_lý>  
    <câu_lệnh_1> | <khối_lệnh_1>  
[ else  
    <câu_lệnh_2> | <khối_lệnh_2> ]
```

- VD:

```
if (select count(*) from SINHVIEN where NGAYSINH<'1975-4-30')>0  
begin  
    select MASV, HOSV, TENS, NGAYSINH, MAKH  
    from SINHVIEN  
    where NGAYSINH<'1975-4-30'  
end  
else  
    print N'Không có SV nào sinh trước ngày 30/4/1975'
```

100 % <

Messages

Không có SV nào sinh trước ngày 30/4/1975



# Cấu trúc điều khiển

- Cấu trúc if exists:

```
if exists (câu_lệnh_select)
    <câu_lệnh_1> | <khởi_lệnh_1>
[ else
    <câu_lệnh_2> | <khởi_lệnh_2> ]
```

- VD:

```
if exists(select * from SINHVIEN where NGAYSINH<'1975-4-30')
begin
    select MASV, HOSV, TENS, NGAYSINH, MAKH
    from SINHVIEN
    where NGAYSINH<'1975-4-30'
end
else
    print N'Không có SV nào sinh trước ngày 30/4/1975'
```

100 % <

Messages

Không có SV nào sinh trước ngày 30/4/1975

# Cấu trúc điều khiển

- Cấu trúc lặp while:

```
while <biểu_thức_luận_lý>  
begin  
    <khởi_lệnh_lặp>  
end
```

- Sử dụng lệnh **break** để thoát khỏi vòng lặp
- Sử dụng lệnh **continue** để bỏ qua các câu lệnh còn lại trong vòng lặp và thực hiện lần lặp kế tiếp.

# Cấu trúc lặp while

## ■ VD:

```
declare @i int, @dem int
set @i=100
set @dem=0
print N'Các số nguyên chia hết cho 4 từ 100=>150 là: '
while(@i<=150)
begin
    if @i%4=0
    begin
        print convert(char(5),@i)
        set @dem=@dem+1
    end
    set @i=@i+1
end
print N'Có '+cast(@dem as char(2))+N' số nguyên chia hết cho 4 từ 100=>150'
```

100 % <

Messages

Các số nguyên chia hết cho 4 từ 100=>150 là:

100

104

108

112

116

120

124

128

132

136

140

144

148

Có 13 số nguyên chia hết cho 4 từ 100=>150

# Kiểu dữ liệu Cursor

- Các lệnh của SQL Server đều thao tác trên nhiều mẫu tin (dòng dữ liệu) cùng lúc.
- **Cursor** là kiểu dữ liệu dùng để duyệt qua từng dòng dữ liệu trả về từ câu truy vấn select, cho phép thực hiện các xử lý khác nhau cho từng dòng dữ liệu cụ thể.
- Đặc điểm:
  - Cho phép thao tác lên từng dòng dữ liệu trả về từ câu lệnh select.
  - Do phải duyệt qua từng dòng dữ liệu nên thao tác xử lý chậm.

# Kiểu dữ liệu Cursor

## ■ Trình tự thực hiện để sử dụng cursor:

1. Khai báo cursor bằng lệnh **declare**
2. Mở cursor đã khai báo trước đó bằng lệnh **open**
3. Đọc và xử lý trên từng dòng dữ liệu trong cursor:
  - Sử dụng biến **@@Fetch\_status**
  - Các lệnh **fetch** và cấu trúc lặp **while**
4. Đóng cursor bằng lệnh **close** và giải phóng bộ nhớ bằng lệnh **deallocate**
  - Sau khi close, có thể mở lại
  - Lệnh **deallocate**: hủy cursor khỏi bộ nhớ

# Kiểu dữ liệu Cursor

- Khai báo con trỏ, trỏ đến một tập dữ liệu (kết quả của Select) bằng lệnh **declare**:

- VD: Có một tập dữ liệu từ câu lệnh Select như sau:

```
select MASV,HOSV,TENSV,MAKH from SINHVIEN
```

- Khai báo một con trỏ đặt tên là **cur\_sinhvien**:

```
declare cur_sinhvien cursor  
for select MASV,HOSV,TENSV,MAKH from SINHVIEN
```

- Mở con trỏ để bắt đầu quá trình đọc các dòng dữ liệu từ Cursor **cur\_sinhvien**:

```
open cur_sinhvien
```

- Khi cursor được mở, con trỏ sẽ trỏ tới dòng đầu tiên của tập dữ liệu, lúc này có thể đọc nội dung dòng đó bằng lệnh **fetch**:

# Kiểu dữ liệu Cursor

- Đọc nội dung dòng hiện tại, lưu vào 4 biến: @masv, @hosv, @tensv và @makh. Nếu đọc thành công (kiểm tra giá trị biến hệ thống @@fetch\_status = 0) thì dịch chuyển con trỏ tới dòng tiếp theo:

```
while @@fetch_status=0
begin
    --Xử lý dòng mới vừa đọc được
    select @tongmonthi=isnull(count(MAMH),0) from KETQUA where MASV=@masv
    print @masv+' '+@hosv+' '+@tensv+' '+@makh+' '+cast(@tongmonthi as varchar(3))
    --Thực hiện đọc tiếp các dòng kế
    fetch next from cur_sinhvien into @masv,@hosv,@tensv,@makh
end
```

- Sau khi xử lý xong, nếu không còn dùng đến Cursor, cần đóng lại và giải phóng các tài nguyên nó chiếm giữ:

```
close cur_sinhvien
deallocate cur_sinhvien
```

# Kiểu dữ liệu Cursor

■ VD:

```
--1. Khai báo biến cursor
declare cur_sinhvien cursor
for select MASV,HOSV,TENSV,MAKH from SINHVIEN
--2. Mở cursor
open cur_sinhvien
--3. Đọc và xử lý trên từng dòng dữ liệu trong cursor
declare @masv char(3),@hosv nvarchar(15),@tensv nvarchar(7),@makh char(2),@tongmonthi int
fetch next from cur_sinhvien into @masv,@hosv,@tensv,@makh
while @@fetch_status=0
begin
    --Xử lý dòng mới vừa đọc được
    select @tongmonthi=isnull(count(MAMH),0) from KETQUA where MASV=@masv
    print @masv+' '+@hosv+' '+@tensv+' '+@makh+' '+cast(@tongmonthi as varchar(3))
    --Thực hiện đọc tiếp các dòng kế
    fetch next from cur_sinhvien into @masv,@hosv,@tensv,@makh
end
--4. Đóng cursor và hủy cursor khỏi bộ nhớ
close cur_sinhvien
deallocate cur_sinhvien
```

100 % <

Messages

```
A01 Nguyễn Thu Hải AV 4
A02 Trần Văn Chính TH 1
A03 Lê Thu Bạch Yến AV 3
A04 Trần Anh Tuấn LS 2
A05 Trần Thanh Triều VL 0
B01 Trần Thanh Mai TH 2
B02 Trần Thị Thu Thủy TH 2
B03 Trần Thị Thanh TH 2
```



# Các hàm thường dùng trong SQL

## ■ Hàm chuyển đổi kiểu dữ liệu:

- Dùng để chuyển dữ liệu từ số, ngày sang chuỗi.
- Muốn nối các kiểu dữ liệu chuỗi, ngày, số lại với nhau thì phải chuyển tất cả thành kiểu chuỗi và dùng toán tử **+** để nối chuỗi.

### ❖ Hàm cast:

- Chuyển một kiểu dữ liệu sang kiểu bất kỳ.
- Cú pháp: **cast ( Biểu\_thức as Kiểu\_dữ\_liệu)**
- VD: `print cast (@tong as varchar(10))`

### ❖ Hàm convert:

- Chuyển kiểu ngày sang kiểu chuỗi.
- Cú pháp: **convert (Kiểu\_dữ\_liệu, Biểu\_thức [, Định\_dạng])**
- VD: `print convert (char(10),getdate(),105)`

100 %

Messages

03-07-2019

# Các hàm thường dùng trong SQL

- Hàm chuyển đổi kiểu dữ liệu:
  - Một số định dạng thường dùng:

Định dạng năm (YY)	Định dạng năm (YYYY)	Hiển thị dữ liệu
1	101	MM/dd/yyyy
3	103	dd/MM/yyyy
5	105	dd-MM-yyyy
12	112	yyyyMMdd
21	121	yyyy-MM-dd

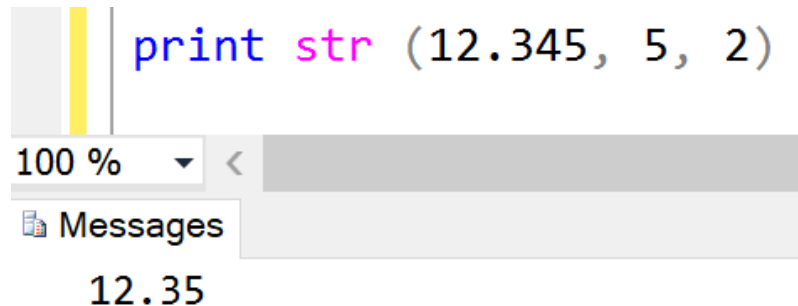
# Các hàm thường dùng trong SQL

## ■ Hàm chuyển đổi kiểu dữ liệu:

### ❖ Hàm str:

- Đổi 1 số thành chuỗi.
- Cú pháp: **str (Số\_thực, Số\_ký\_tự [, Số\_lẻ])**
- VD:

```
print str (12.345, 5, 2)
```



100 % <

Messages

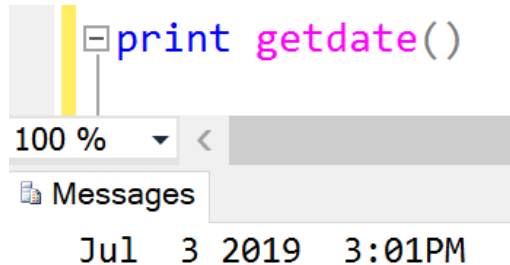
12.35

# Các hàm thường dùng trong SQL

## ■ Hàm ngày giờ:

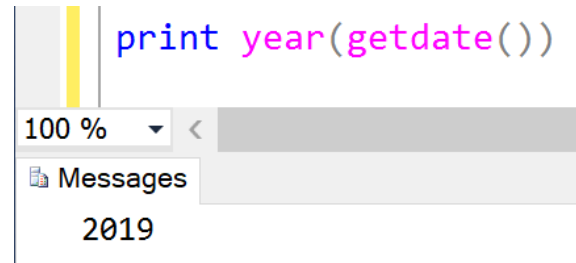
### ❖ Hàm **getdate()**:

- Trả về ngày, tháng, năm và giờ, phút, giây hiện hành.
- Cú pháp: **getdate()**
- VD:



### ❖ Hàm **day()**, **month()**, **year()**:

- Trả về ngày, tháng, năm.
- Cú pháp: **day()**; **month()**; **year()**
- VD:



# Các hàm thường dùng trong SQL

## ■ Hàm ngày giờ:

- Bảng mô tả các định dạng sử dụng trong các hàm thời gian:

Giá trị	Định dạng
Năm	yy, yyyy
Quý	qq, q
Tháng	mm, m
Ngày trong năm	dy, y
Ngày trong tuần	dw
Ngày trong tháng	dd, d
Tuần	wk, ww
Giờ	hh
Phút	mi, n
Giây	ss, s

# Các hàm thường dùng trong SQL

## ■ Hàm ngày giờ:

### ❖ Hàm DateAdd:

- Cộng một số vào giá trị ngày và trả về một giá trị ngày.
- Cú pháp: **DateAdd (Định dạng, Số, Ngày)**
- VD:

```
print N'Hôm nay là ngày: ' + convert (char(10),getdate(),105)
declare @ngaymoi datetime
set @ngaymoi = dateadd (mm, 5, getdate())
print N'Cộng thêm 5 tháng nữa là ngày: ' + convert(char(10), @ngaymoi, 105)
```

100 % <

Messages

Hôm nay là ngày: 03-07-2019

Cộng thêm 5 tháng nữa là ngày: 03-12-2019

### ❖ Hàm DateDiff:

- Trả về khoảng cách của hai ngày.
- Cú pháp: **DateDiff (Định\_dạng, ngày\_1, ngày\_2)**
- VD:

```
print N'Hôm nay là ngày: ' + convert (char(10),getdate(),105)
declare @ngaymoi datetime = '2019-11-07'
declare @sotuan int = datediff (wk, getdate(), @ngaymoi)
print N'Khoảng cách từ nay đến ngày ' + convert(char(10), @ngaymoi, 105) + N' là ' + cast(@sotuan as varchar(5)) + N' tuần'
```

100 % <

Messages

Hôm nay là ngày: 03-07-2019

Khoảng cách từ nay đến ngày 07-11-2019 là 18 tuần


# Các hàm thường dùng trong SQL

## ■ Hàm ngày giờ:

### ❖ Hàm DateName:

- Trả về chuỗi thời gian.
- Cú pháp: **DateName (Định dạng, Ngày)**
- VD:

```
print N'Today is ' + datename(dw, getdate())
```



100 % < Messages  
Today is Wednesday

### ❖ Hàm DatePart:

- Trả về một giá trị của ngày.
- Cú pháp: **DatePart (Định dạng, Ngày)**
- VD:

```
print N'Hôm nay là ngày: ' + convert(char(10), getdate(), 105)
declare @quy int = datepart(qq, getdate())
print N'Thuộc quý ' + cast(@quy as varchar(5)) + N' của năm ' + cast(year(getdate()) as char(4))
```



100 % < Messages  
Hôm nay là ngày: 03-07-2019  
Thuộc quý 3 của năm 2019

# Các hàm thường dùng trong SQL


## ■ Hàm xử lý số:

### ❖ Hàm Round:

- Làm tròn số.
- Cú pháp: **Round (số cần làm tròn, số chữ số thập phân)**

- VD:

```
print round (12.3456, 2)
```



The screenshot shows a code editor with a dropdown menu set to '100 %' and a 'Messages' panel. The output of the code is '12.3500'.



# Các hàm thường dùng trong SQL

## ■ Hàm xử lý chuỗi:

### ❖ Hàm Left, Right, Substring:

- Cắt chuỗi bên trái, phải và ở giữa.
- Cú pháp:

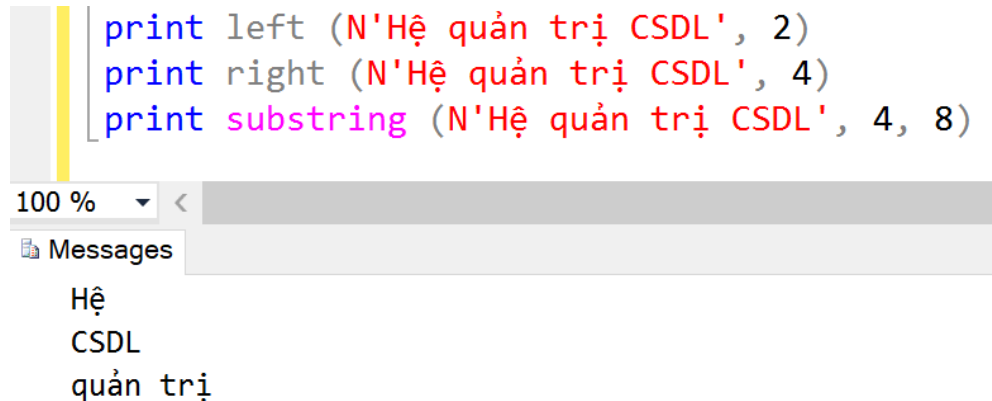
Left (chuỗi, số ký tự)

Right (chuỗi, số ký tự)

Substring (chuỗi, vị trí bắt đầu, số ký tự)

- VD:

```
print left (N'Hệ quản trị CSDL', 2)
print right (N'Hệ quản trị CSDL', 4)
print substring (N'Hệ quản trị CSDL', 4, 8)
```



The screenshot shows a SQL query execution window with a dropdown menu set to 100%. Below the query, there is a 'Messages' tab. The output of the query is displayed in a text area, showing the results of the Left, Right, and Substring functions applied to the string 'Hệ quản trị CSDL'.

Message
Hệ
CSDL
quản trị

# Các hàm thường dùng trong SQL

## ■ Hàm xử lý chuỗi:

### ❖ Hàm Upper, Lower:

- Chuyển đổi thành chữ hoa và chữ thường.
- Cú pháp:

Upper (chuỗi)

Lower (chuỗi)

- VD:

```
print upper(N'Hệ quản trị CSDL')  
print lower (N'Hệ quản trị CSDL')
```

100 % <

Messages

HỆ QUẢN TRỊ CSDL

hệ quản trị csdl

# Các hàm thường dùng trong SQL

## ■ Hàm xử lý chuỗi:

### ❖ Hàm LTrim, RTrim:

- Bỏ khoảng trắng bên trái, bên phải chuỗi.
- Cú pháp:

LTrim (chuỗi)

RTrim (chuỗi)

- VD:

```
print ltrim(N'   CSDL')
print rtrim(N'CSDL   ')
print ltrim(rtrim(N'   CSDL   '))
```

100 %

Messages

CSDL  
CSDL  
CSDL

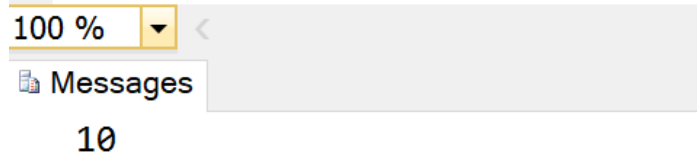
# Các hàm thường dùng trong SQL

## ■ Hàm xử lý chuỗi:

### ❖ Hàm Len:

- Trả về số ký tự trong chuỗi.
- Cú pháp: **Len (chuỗi)**

- VD: `print len(N'Hệ QT CSDL')`



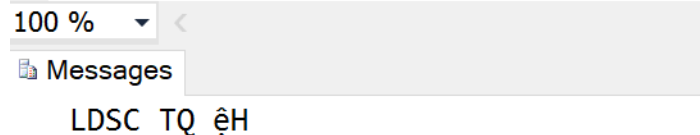
100 % < Messages  
10

### ❖ Hàm Reverse:

- Đảo chuỗi.
- Cú pháp:

#### **Reverse (chuỗi)**

- VD: `print reverse(N'Hệ QT CSDL')`



100 % < Messages  
LDSC TQ ệH

# Bài tập ứng dụng

## ■ Lập trình với CSDL Quản lý sinh viên:

### - Sử dụng cấu trúc điều khiển:

#### • Sử dụng cú pháp IF để thực hiện các yêu cầu sau:

1. Cho biết học bổng trung bình của SV khoa Tin Học là bao nhiêu? Nếu lớn hơn 100,000 thì in ra “không tăng học bổng”, ngược lại in ra “nên tăng học bổng”.
2. Sử dụng hàm DATENAME để tính xem có SV nào sinh vào ngày chủ nhật không? Nếu có thì in ra danh sách các SV đó, ngược lại thì in ra chuỗi “Không có SV nào sinh vào ngày Chủ Nhật”.
3. Hãy cho biết SV có mã số A01 đã thi bao nhiêu môn, nếu có thì in ra “SV A01 đã thi xxx môn”, ngược lại thì in ra “SV A01 chưa có kết quả thi”.
4. Hãy cho biết SV có mã số A01 đã thi đủ tất cả các môn chưa, nếu có thì in ra “SV A01 đã thi đủ tất cả các môn”, ngược lại thì in ra “SV A01 chưa thi đủ tất cả các môn”.
5. Hãy cho biết môn Vật lý nguyên tử đã SV thi chưa, nếu có thì in ra “Đã có SV thi môn Vật lý nguyên tử với điểm trung bình là xxx”, ngược lại thì in ra “Chưa có SV thi môn Vật lý nguyên tử”.

#### • Sử dụng cú pháp CASE để thực hiện các yêu cầu sau:

6. Liệt kê danh sách các SV có bổ sung thêm cột hiển thị thứ trong tuần (bằng tiếng Việt) của ngày sinh.

# Bài tập ứng dụng

## ■ Lập trình với CSDL Quản lý sinh viên:

### - Sử dụng cấu trúc điều khiển:

- Sử dụng cú pháp WHILE để thực hiện các yêu cầu sau:

7. Tính tổng các số nguyên từ 1 đến 100.
8. Tính tổng chẵn và tổng lẻ của các số nguyên từ 1 đến 100.
9. Tạo một bảng tên MONHOC\_1 có cấu trúc và dữ liệu dựa vào bảng MONHOC (chỉ lấy hai cột: MAMH, TENMH). Sau đó, sử dụng vòng lặp WHILE viết đoạn chương trình dùng để xóa từng dòng dữ liệu trong bảng MONHOC\_1 với điều kiện câu lệnh bên trong vòng lặp khi mỗi lần thực hiện chỉ được phép xóa một dòng dữ liệu trong bảng MONHOC\_1. Sau khi xóa một dòng thì thông báo ra màn hình nội dung “Đã xóa môn học ” + Tên môn học.

# Bài tập ứng dụng

## ■ Lập trình với CSDL Quản lý sinh viên:

### - Sử dụng đối tượng Cursor:

6. Duyệt cursor và xử lý hiển thị danh sách các SV gồm các thông tin: mã SV, họ tên SV, mã khoa, và có thêm cột tổng số môn thi.
7. Duyệt cursor và xử lý hiển thị danh sách các môn học có thêm cột Ghi chú, biết rằng nếu đã có SV thi thì in ra “Đã có xxx SV thi”, ngược lại thì in ra “Chưa có SV thi”.
8. Duyệt cursor và xử lý giảm học bổng của các SV theo các qui tắc sau:
  - Không giảm nếu  $\text{ĐTB} \geq 8.5$
  - Giảm 5% nếu  $7.5 \leq \text{ĐTB} < 8.5$
  - Giảm 10% nếu  $7 \leq \text{ĐTB} < 7.5$

# Câu hỏi trắc nghiệm

1. Chọn câu lệnh đúng để khai báo và gán giá trị cho biến:

- a. `set @ngay_hien_hanh = getdate()`
- b. `select @ngay_hien_hanh datetime = getdate()`
- c. `declare datetime @ngay_hien_hanh = getdate()`
- d. **`declare @ngay_hien_hanh datetime = getdate()`**



# Câu hỏi trắc nghiệm

## 2. Chọn phát biểu SAI về biến hệ thống:

- a. Biến hệ thống là biến chỉ đọc do SQL Server định sẵn.
- b. Không thể xem giá trị của biến hệ thống với lệnh print.
- c. Biến hệ thống cho biết trạng thái của hệ thống.
- d. Tên của biến hệ thống bắt đầu bằng hai ký tự @@.

# Câu hỏi trắc nghiệm

## 3. Chọn trình tự thực hiện ĐÚNG để sử dụng Cursor:

1. Mở cursor
2. Khai báo cursor
3. Đóng cursor và giải phóng bộ nhớ
4. Đọc và xử lý trên từng dòng dữ liệu trong cursor

- a. 1, 2, 3, 4
- b. 1, 2, 4, 3
- c. 2, 1, 4, 3
- d. 4, 3, 2, 1

# Câu hỏi trắc nghiệm

4. Chọn câu lệnh ĐÚNG để hiển thị thông tin ngày hiện hành:

- a. `print N'Hôm nay là ngày '+ convert(char(10),getdate(), 105)`
- b. `print N'Hôm nay là ngày '+ convert(getdate() as char(10))`
- c. `print N'Hôm nay là ngày '+ cast(char(10),getdate(), 105)`
- d. `print N'Hôm nay là ngày '+ getdate()`

# Câu hỏi trắc nghiệm

5. Câu lệnh sau sẽ cho kết quả gì?

```
print len('CSDL')
```

- a. 0
- b. 4
- c. C
- d. CS