CHƯƠNG 3: LẬP TRÌNH VỚI SQL

A. KIẾN THỨC CẦN NHỚ

1. Khai báo và sử dụng biến

Có 2 loại biến: Cục bộ và toàn cục

Biến cục bộ: là biến chỉ sử dụng trong đoạn chương trình khai báo nó như Query Batch, Stored Procedure, Function, chứa giá trị thuộc một kiểu nhất định.

Biến cục bộ được bắt đầu bằng 1 ký hiệu @

Khai báo:

DECLARE **<@***tên_biến*> <*Kiểu_dữ_liệu* **>, …**

Gán giá trị cho biến

SET @*tên_biến* = {*giá_trị*|*biến*|b*iểu_thức*|*SELECT …*}

Biến toàn cục: là biến được sử dụng bất kỳ đâu trong hệ thống. Trong SQL biến toàn cục là các biến hệ thống do SQL Server cung cấp.

SQL tự cập nhật giá trị cho các biến này, người sử dụng không thể gán giá trị trực tiếp cho biến này

Bản chất là 1 hàm (Function) và bắt đầu bằng ký tự @@

Một số biến toàn cục trong SQL

Tên biến	Ý nghĩa
@@ERROR	Mã số lỗi của câu lệnh T-SQL
@@FETCH_STATUS	Trạng thái truy cập Con trỏ: 0 nếu trạng thái truy cập thành công, -1 nếu không thành công

@@IDENTITY	Giá trị xác định (identity) được thêm vào
@@ROWCOUNT	Số lượng dòng của kết quả câu lệnh SQL
@@SERVERNAME	Tên của Server địa phương
@@TRANSCOUNT	Số lượng những giao dịch đang được mở
@@VERSION	Thông tin về phiên bản SQL Server đang dùng
@@CURSOR_ROWS	Số lượng các dòng dữ liệu của Con trỏ

2. Một số cấu trúc lệnh cơ bản

2.1. Cấu trúc IF…

Cú pháp:

IF *<điều kiện>*

Lệnh| Khối_lệnh 1

[***ELSE*** *Lệnh| Khối_lệnh*]

Khối lệnh là một hoặc nhiều lệnh nằm trong cặp từ khóa BEGIN…END

Giải thích cấu trúc

Kiểm tra điều kiện, nếu điều kiện đúng thì thực hiện khối lệnh 1, ngược lại thực hiện khối lệnh 2 và kết thúc.

2.2. Cấu trúc CASE

Cú pháp: Có hai dạng

Dạng 1:	**Dạng 2:**
CASE *Biểu_thức*	CASE
WHEN *Giá_trị 1 T*hen *kết_quả 1*	WHEN *<điều kiện 1>* THEN *kết_quả 1*
[WHEN Giá_trị 2 Then Kết_quả 2	WHEN *<điều kiện 2>* THEN *kết_quả 2*
[...n]	[...n]
[ELSE *kết_quả_khác*]	[ELSE *kết_quả_khác*]
END	END

Giải thích dạng 1:

Nếu biểu thức là giá trị 1 thì nhận kết quả 1 và kết thúc CASE, ngược lại nếu biểu thức là giá trị 2 thì nhận kết quả 2 và kết thúc CASE, … , ngoài ra thì nhận kết quả khác và kết thúc CASE.

Giải thích dạng 2:

Kiểm tra điều kiện, nếu điều kiện 1 đúng thì nhận kết quả 1 và kết thúc CASE, ngược lại nếu điều kiện 2 đúng thì nhận kết quả 2 và kết thúc CASE, …, ngoài ra nhận kết quả khác và kết thúc CASE.

2.3. Cấu trúc WHILE

Cú pháp

WHILE *<điều_kiện>*

BEGIN

Lệnh| Khối lệnh

[BREAK]

[CONTINUE]

END

Có thể thêm *Break* và *Continue* trong khối lệnh của while

BREAK: thoát khỏi vòng WHILE hiện hành.

CONTINUE : trở lại đầu vòng WHILE , bỏ qua các lệnh sau đó

Giải thích cấu trúc

Kiểm tra điều kiện, nếu điều kiện đúng thì thực hiện khối lệnh, tiếp tục kiểm tra điều kiện, cho đến khi nào điều kiện sai thì thoát khỏi WHILE.

Để vòng lặp không bị vô hạn thì trong nhóm lệnh phải có lệnh thay đổi điều kiện và sau một số lần lặp thì điều kiện sẽ sai và kết thúc WHILE.

3. THỦ TỤC (Stored Procedure)

Thủ tục là một đối tượng trong hệ quản trị CSDL bao gồm các câu lệnh SQL, chúng được kết hợp lại với nhau thành một khối lệnh, dùng để thực hiện một số công việc nào đó như cập nhật, thêm mới, xóa, hiển thị, tính toán và có thể trả về các giá trị.

Thủ tục hệ thống: là những thủ tục do SQL cung cấp (tự nghiên cứu System Stored Procedures) tên có tiếp đầu ngữ *sp_*

Thủ tục người dùng: do người dùng tạo ra, để dễ dàng phân biệt chúng ta quy định tên thủ tục có tiếp đầu ngữ *usp_*

Tạo thủ tục:

CREATE PROCEDURE *<Tên thủ tục>*

Danh sách tham số vào

[Danh sách tham số ra <Output>]

AS

<Đoạn chương trình xử lý>

[**RETURN** [*giá trị trả về*]]

GO

Lưu ý:

Lệnh RETURN được sử dụng để kết thúc thủ tục và trả về giá trị là một số. Giá trị mặc định là RETURN 0 nghĩa là công việc thành công, quy ước RETURN -1 công việc không thành công.

Lời gọi thủ tục

EXECUTE tên_thủ_tục [danh_sách_các_đối_số]

Số lượng các đối số cũng như thứ tự của chúng phải phù hợp với số lượng và thứ tự của các tham số khi định nghĩa thủ tục.

Chỉnh sửa thủ tục

Thay từ khóa CREATE trong lệnh tạo thủ tục bằng từ khóa ALTER.

Xóa thủ tục

DROP PROCEDURE <Tên thủ tục>

Mã hóa thủ tục

Thủ tục sẽ được mã hoá nếu tuỳ chọn WITH ENCRYPTION được chỉ định. Nếu thủ tục đã được mã hoá, ta không thể xem được nội dung của thủ tục.

Thêm từ khóa WITH ENCRYPTION trong lệnh ALTER thủ tục.

Biên dịch lại thủ tục

Khi người sử dụng làm thay đổi tới những index của bảng. Stored Procedures phải được biên dịch lại (recompiled) để chấp nhận những thay đổi đó.

Thêm từ khóa WITH RECOMPILE **trong lệnh ALTER thủ tục**

4. HÀM (Function)

Hàm là một đối tượng trong hệ quản trị CSDL, tương tự như thủ tục. Điểm khác biệt giữa hàm và thủ tục là hàm trả về một giá trị**.** Giá trị trả về có thể là một bảng có được từ một câu truy vấn**.**

Hàm hệ thống: System Function.

Hàm người dùng: Do người dùng tạo ra gồm 3 dạng:

- Scalar_valued Function

Giá trị trả về là kiểu dữ liệu cơ sở (int, varchar, float, datetime…)

- Table_valued Function:

Giá trị trả về là một Table có được từ một câu truy vấn

- Aggregate Function:

Giá trị trả về là một bảng mà dữ liệu có được nhờ tích lũy dần sau một chuỗi thao tác xử lý và insert

Tạo hàm:

CREATE FUNCTION tên_hàm ([danh_sách_tham_số vào])

RETURNS (kiểu_trả_về_của_hàm| Table)

AS

BEGIN

Các_câu_lệnh_của_hàm

RETURN {Giá trị | Biến | Biểu thức | *Câu lệnh truy vấn*}

END

5. CON TRỎ (Cursor)

Là một cấu trúc dữ liệu, ánh xạ đến một danh sách gồm các dòng dữ liệu từ một kết quả truy vấn (SELECT), cho phép duyệt tuần tự các dòng dữ liệu và đọc giá trị từng dòng trong danh sách kết quả.

Định nghĩa Con trỏ

DECLARE <Tên Con trỏ> CURSOR

FOR <Câu lệnh Truy vấn SELECT>

CON trỏ là cấu trúc toàn cục, duyệt theo một chiều từ đầu đế cuối, nội dung của Con trỏ có thể thay đổi.

Kiểm tra tình trạng Con trỏ:

Biến hệ thống @@FETCH_STATUS

Cho biết lệnh fetch vừa thực hiện có thành công hay không. Là cơ sở để biết đã duyệt đến cuối danh sách hay chưa

Nếu @@FETCH_STATUS =0 thì thành công, Con trỏ đang ở vị trí dòng thỏa mãn điều kiện trong kết quả truy vấn.

Nếu @@FETCH_STATUS <>0 thì KHÔNG thành công, Con trỏ đang ở vị trí vượt qua dòng cuối cùng kết quả truy vấn.

Các bước sử dụng Con trỏ trong lập trình

B1. Định nghĩa CURSOR từ một kết quả SELECT

DECLARE <tên Con trỏ> CURSOR FOR <Câu lệnh SELECT>

B2. Mở Cursor:

OPEN <Ten Con trỏ> , Con trỏ tham chiếu đến dòng 0

B3. Truy cập đến các bản ghi

FETCH NEXT FROM <Cursor_name> INTO <ds biến>

B4. Kểm tra có thành công không:

Nếu @FETCH_STATUS = 0 thì xử lý lệnh, quay lại B3

Nếu @FETCH_STATUS <> 0 thì sang B5

B5. Đóng Cursor:

CLOSE <Cursor_name>

B6. Xoá tham chiếu của Cursor:

DEALLOCATE <Cursor_name>

Sử dụng Con trỏ có thể đến vị trí một dòng nhất định trong tập kết quả.

6. Một số hàm cơ bản:

6.1. Các hàm toán học:

1. ABS(*x*) : Trị tuyệt đối của x.
2. SQRT(*x*) : Căn bậc hai của x.
3. SQUARE(x) : x bình phương.
4. POWER(y, x) : y lũy thừa x.
5. LOG(*x*) : Logarit của x.
6. EXP(*x*) : Hàm mũ cơ số e của x.
7. SIGN(*x*) : Lây dấu của số x (-1: x<0, 0: x=0, +1: x>0).
8. ROUND(*x,n*) : Làm tròn tới n số thập phân.
9. CEILING(x) : Số nguyên nhỏ nhất nhưng lớn hơn x.
10. FLOOR(X) : Số nguyên lớn nhất nhưng nhỏ hơn x.
11. ... và các hàm lượng giác: SIN, COS, TAN, ASIN, ACOS, ATAN

...

6.2. Các hàm xử lý chuỗi

1. ACSII(ch) : Mã ASCII của ký tự ch.
2. CHAR(n) : Ký tự có mã ASCII là n.
3. LOWER(str) : Trả vê chuỗi chữ thường.
4. UPPER(str) : Trả về chuỗi in hoa.
5. LTRIM(str) : Trả về chuỗi không có dấu cách bên trái.
6. RTRIM(str) : Trả về chuỗi không có dấu cách bên phải.
7. LEFT*(str,n)*: Lấy *n* ký tự phía trái của dãy *str*.
8. RIGHT(*str,n*): Lấy *n* ký tự phía phải của dãy *str*.

9. SUBSTRING(*str, start, n*): Lây *n* ký tự của dãy str kể từ vị trí start trong dãy.

10. REPLACE(str1, str2, str3): thay thế tất cả str2 trong str1 bằng str3.

11. STUFF(*str1, start, n, str2*): Thay thế n ký tự trong str1 từ vị trí start bằng chuỗi str2.

12. STR(x, len [, Dec]): Chuyển số x thành chuỗi.

6.3. Hàm xử lý ngày tháng

1. GETDATE(): Cho ngày tháng năm hiện tại.

2. DAY(*dd*): Cho số thứ tự ngày trong tháng của dd.

3. MONTH(*dd*): Cho số thứ tự tháng trong năm của dd.

4. YEAR(*dd*): Cho năm của biểu thức ngày *dd*.

6.4. Hàm chuyển đổi kiểu dữ liệu

1. CAST (biểu_thức AS kiểu_dữ_liệu)

Chuyển đổi giá trị của biểu thức sang kiểu được chỉ định.

2. CONVERT(kiểu_dữ_liệu, biểu_thức [,kiểu_chuyển_đổi])

Hàm có chức năng chuyển đổi giá trị của biểu thức sang kiểu dữ liệu được chỉ định. Tham số *kiểu_chuyển_đổi* là một giá trị số thường được sử dụng khi chuyển đổi giá trị kiểu ngày sang kiểu chuỗi nhằm qui định khuôn dạng dữ liệu được hiển thị và được qui định như sau:

Kiểu chuyển đổi

101	mm/dd/yy
102	yy.mm.dd
103	dd/mm/yy