Chương 7: **CURSOR**

- Giới thiệu Cursor
- Các loại Cursor
- Làm việc với T-SQL Server Cursor

Giới thiệu

- **Cursor** là một đối tượng của CSDL mà nó hỗ trợ cho phép truy xuất và thao tác dữ liệu trong một tập kết quả (result set)
- **Dùng Cursor để:**
  - Định vị một dòng đặc biệt trong tập tin kết quả.
  - Truy xuất một dòng hoặc khối dòng bắt đầu từ vị trí của Cursor trong tập kết quả.
  - Cung cấp thao tác hiệu chỉnh dữ liệu cho các dòng tại vị trí của cursor trong tập kết quả.
  - Cung cấp các mức độ khác nhau của tính tường minh trong sự thay đổi được tạo bởi những người dùng khác đến tập kết quả.
  - Cung cấp việc truy cập dữ liệu trong tập kết quả cho các câu lệnh T-SQL trong script, Store procedure và Triggers.

Giới thiệu

- **Các thao tác cần thực hiện trong khi sử dụng Cursor trong SQL Server:**
  - Cursor cần phải khai báo và các thuộc tính của nó cũng cần được xác định.
  - Mở Cursor.
  - Phải duyệt (fetch) các dòng cần thiết từ Cursor.
  - Dữ liệu trong dòng hiện hành có thể được hiệu chỉnh khi cần thiết.
  - Tạm thời không dùng Cursor thì phải đóng Cursor lại.
  - Cursor cần phải được giải phóng khi không cần dùng nữa.

Các loại Cursor

- **T-SQL Sever cursor:** Cursor dựa trên câu lệnh khai báo cursor, được dùng chủ yếu trong các sript, store procudure, triggers. Nó được thi hành trên server và được quản lý bởi các câu lệnh T-SQL gửi từ client đến server.
- **API Server cursor (API-Application Program interface):** được thực thi trên server, được quản lý bởi một hàm cursor API và được cung cấp bởi hàm cursor API trong OLE DB, ODBC, và DB-Library.
- **Client Server:** SQL server ODBC driver, DB-Library DLL và ADO API DLL giúp thi hành cursor client một cách nội tại, cursor được thi hành bằng cách nắm giữ tập kết quả của client.

Các loại Cursor

Static :

- Tập kết quả của con trỏ loại Static được xây dựng trong temdb khi con trỏ được mở.
- Một Static con trỏ luôn luôn hiện tập kết quả giống như tập kết quả có được ngay sau khi con trỏ mở.
- Con trỏ không phản ánh bất kỳ sự thay đổi nào trong Database ngay cả khi những dòng dữ liệu có thể thay đổi được, các dòng mới được insert vào bởi các Transaction khác cũng vẫn không hiện lên mặc dù chúng thỏa điều kiện lọc dữ liệu.
- Thao tác Insert, Update, Delete đều không có tác dụng khi dùng static cursor.

Các loại Cursor

Keyset - driven:

- Thành viên và thứ tự các dòng trong một keyset – driven cursor là cố định khi cursor được mở. Con trỏ được điều khiển bởi một tập giá trị nhận dạng gọi là keyset. Keyset được xây dựng từ một tập các cột mà dùng để nhận dạng các dòng trong tập kết quả. Keyset được xây dựng trong temdb khi con trỏ được mở.
- Cho phép hiệu chỉnh (update, delete) dữ liệu trên cột không là keyset (bởi chủ cursor hay từ user khác) khi user duyệt thông qua con trỏ.
- Có thể thêm (insert) vào bảng nếu như cursor có thể chèn dữ liệu vào bảng.

Các loại Cursor

Dynamic: Trái ngược với Static cursor

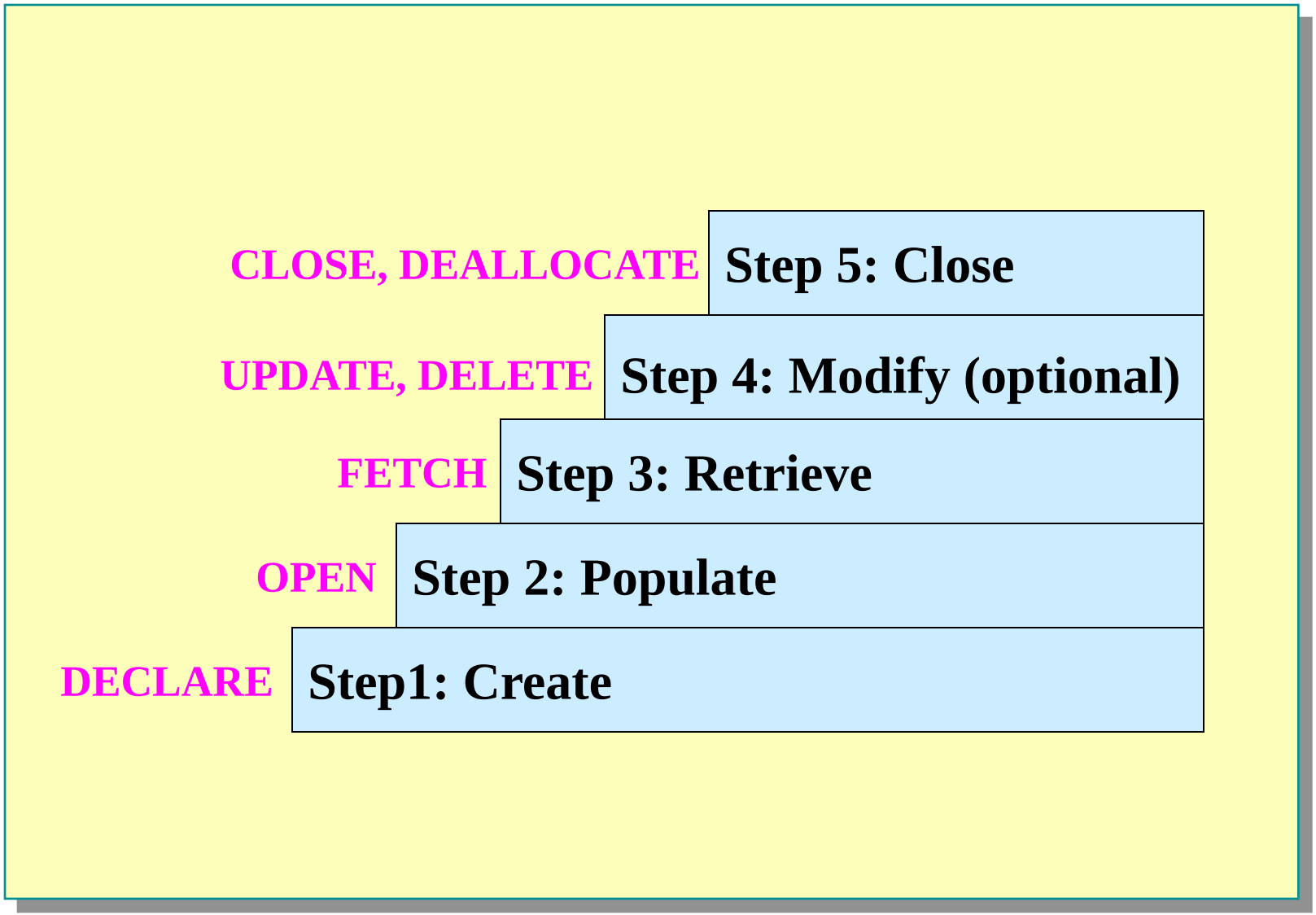
- Dynamic cursor phản ánh được tòan bộ sự thay đổi của các dòng dữ liệu trong tập kết quả khi duyệt con trỏ.
- Giá trị dữ liệu, thứ tự và các thành viên của các dòng trong tập tin kết quả có thể thay đổi ứng với mỗi lần duyệt con trỏ.
- Tất cả các lệnh Insert, Update, Delete của các user đều hữu hiệu thông qua con trỏ.
- Sự Update hữu hiệu ngay tức thời nếu chúng được Update thông qua con trỏ ứng với mẫu tin hiện thời, còn nếu Update bên ngoài con trỏ thì nó không hữu hiệu cho đến khi nó hoàn tất.

Các loại Cursor

Fast Forward only : Tương tự Dynamic nhưng nó chỉ duyệt con trỏ theo một chiều từ First đến Last

Cursor Type	Membership	Order	Values
Forward-only	**Dynamic**	**Dynamic**	**Dynamic**
Static	**Fixed**	**Fixed**	**Fixed**
Dynamic	**Dynamic**	**Dynamic**	**Dynamic**
Keyset-driven	**Fixed**	**Fixed**	**Dynamic**

Làm việc với T-SQL Server Cursor



Khai báo *Cursors*

- **Khai báo Cursors:**

  DECLARE cursor_name CURSOR

  [LOCAL | GLOBAL] [FORWARD_ONLY |SCROLL]

  [STATIC | KEYSET | DYNAMIC | FAST_FORWARD]

  [READ_ONLY | SCROLL_LOCKS | OPTIMISTIC]

  [TYPE_WARNING]

  FOR {select_statement}

  [FOR UPDATE [OF column_list] | FOR READ ONLY]

- **Mở Cursors:** OPEN cursor_name

Lấy mẫu tin và điều hướng Cursor

- **Lấy dữ liệu (Fetching Data):** Truy xuất từng dòng dữ liệu. Kiểm tra phạm vi con trỏ bằng @@Fetch_status

  FETCH [[NEXT | PRIOR | FIRST | LAST | ABSOLUTE n | RELATIVE n]] FROM cursor_name [INTO @variable_name [,...n]]

- **Đóng Cursor (Closing Cursors):**

  CLOSE cursor_name

- **Giải phóng Cursor (Deallocate Cursors):**

  DEALLOCATE cursor_name

Biến toàn cục với Cursor

- @@FETCH _STATUS: Trả về giá trị 1 khi con trỏ đã được dời đến quá cuối tập tin. Trả về 0 vẫn còn trong phạm vi của tập kết quả.

Return value	Description
0	FETCH statement was successful.
-1	FETCH statement failed or the row was beyond the result set.
-2	Row fetched is missing.

- @@CURSOR_ROWS: Trả về số dòng của Cursor đang mở.

Ví dụ về Cursors

```
USE Northwind

DECLARE KH CURSOR FOR

        SELECT companyname FROM customers WHERE
            city LIKE “L%“ ORDER BY  companyname

OPEN KH

FETCH NEXT FROM KH

WHILE @@FETCH_STATUS = 0

BEGIN

        FETCH NEXT FROM KH

END

CLOSE KH

DEALLOCATE KH
```

Ví dụ về Cursors

VD 2: Giả sử người quản lý cần một bảng báo cáo lịch sử khách hàng như sau:

Customer:ALFKI - Alfreds Futterkiste
- Order:10643 (Aug 25 1997)
- Order:10692 (Oct 3 1999)
- Order:10702 (Oct 13 1999)

Customer:ANATR - Ana Trujillo Emparedados y helados
- Order:10625 (Aug 8 1997)
- Order:10759 (Nov 28 1999)

Customer:ANTON - Antonio Moreno Taquería
- Order:10507 (Apr 15 1997)
- Order:10535 (May 13 1999)
- Order:10573 (Jun 19 1999)
- Order:10677 (Sep 22 1999)
- Order:10682 (Sep 25 1999)

Ví dụ về Cursors

Đoạn Batch thực hiện như sau:

```
DECLARE rpt CURSOR FOR
        SELECT c.CustomerID, c.CompanyName, o.OrderID,
        o.OrderDate
        FROM Customers c, Orders o
        WHERE c.CustomerID = o.CustomerID
        AND c.CustomerID LIKE 'A%'
        AND DatePart( year, o.OrderDate) = 1997
DECLARE @cid char( 8), @cname char( 40),
        @ordid char( 8), @orddt datetime, @old char( 8)
OPEN rpt
FETCH NEXT FROM rpt INTO @cid, @cname, @ordid,
@orddt
SELECT @old = ' ‘
```

Ví dụ về Cursors

```
WHILE @@fetch_status = 0
BEGIN
    IF @old = @cid
      BEGIN
            PRINT ' Order:' + rtrim( @ordid) + ' (' + cast(@orddt as
            CHAR( 10)) + ')'
      END
    ELSE
      BEGIN
            PRINT 'Customer:' + rtrim( @cid) + ' - ' +
            rtrim(@cname)
            PRINT ' Order:' + rtrim( @ordid) + ' (' + cast(@orddt as
            CHAR( 11)) + ')'
            SELECT @old = @cid
      END
FETCH NEXT FROM rpt INTO @cid, @cname, @ordid, @orddt
END
CLOSE rpt
DEALLOCATE rpt
```

Sửa chữa dữ liệu trong Cursor

- Sửa đổi dữ liệu trong Cursor

  UPDATE <*table_name*>

  SET <*column_name*> = <*expression*>

  WHERE CURRENT OF <*cursor_name*>

- Xóa dữ liệu trong Cursor

  DELETE <*table_name*>

  WHERE CURRENT OF <*cursor_name*>

Ví dụ về Cursors

- **Example:**

```
DECLARE MyCursor CURSOR FOR
SELECT c.CustomerID,c.Companyname,c.contactname,
            o.OrderID,o.OrderDate
FROM Customers c, Orders o WHERE c.CustomerID = o.CustomerID
FOR UPDATE
OPEN MyCursor
DECLARE @cid VARCHAR( 8), @c VARCHAR( 80), @o INT,
    @od DATETIME, @cn VARCHAR( 80)
FETCH NEXT FROM MyCursor INTO @cid, @c, @cn, @o, @od
SELECT @cid
BEGIN TRANSACTION
      UPDATE Customers SET CompanyName = 'q'
      WHERE CURRENT OF Mycursor
      DEALLOCATE MyCursor
      SELECT *  FROM Customers
ROLLBACK TRANSACTION
```

Ví dụ về Cursors

Example 3 :

```
CREATE TABLE t1(c1 INT PRIMARY KEY, c2 INT)
GO
CREATE TABLE t2(d1 INT PRIMARY KEY, d2 INT)
GO
INSERT INTO t1 VALUES (1, 10)
INSERT INTO t2 VALUES (1, 20)
INSERT INTO t2 VALUES (2, 30)
Go
DECLARE abc CURSOR LOCAL FOR
        SELECT * FROM t1
OPEN abc
FETCH abc
UPDATE t1 SET c2 = c2 + d2 FROM t2 WHERE CURRENT OF abc
GO
```