Transact-SQL

Transact-SQL (**T-SQL**) is Microsoft's and Sybase's proprietary extension to the SQL (Structured Query Language) used to interact with relational databases. T-SQL expands on the SQL standard to include procedural programming, local variables, various support functions for string processing, date processing, mathematics, etc. and changes to the DELETE and UPDATE statements.

Transact-SQL is central to using Microsoft SQL Server. All applications that communicate with an instance of SQL Server do so by sending Transact-SQL statements to the server, regardless of the user interface of the application.

Stored procedures in SQL Server are executable server-side routines. The advantage of stored procedures is the ability to pass parameters.

Contents

Variables

Transact-SQL provides the following statements to declare and set local variables: DECLARE, SET and SELECT.

```
DECLARE @var1 NVARCHAR(30);
SET @var1 = 'Some Name';
SELECT @var1 = Name
  FROM Sales.Store
  WHERE CustomerID = 100;
```

Flow control

Keywords for flow control in Transact-SQL include BEGIN and END, BREAK, CONTINUE, GOTO, IF and ELSE, RETURN, WAITFOR, and WHILE.

`IF` and `ELSE` allow conditional execution. This batch statement will print "It is the weekend" if the current date is a weekend day, or "It is a weekday" if the current date is a weekday. (Note: This code assumes that Sunday is configured as the first day of the week in the `@@DATEFIRST` setting.)

```sql
IF DATEPART(dw, GETDATE()) = 7 OR DATEPART(dw, GETDATE()) = 1
    PRINT 'It is the weekend.';
ELSE
    PRINT 'It is a weekday.';
```

`BEGIN` and `END` mark a <u>block of statements</u>. If more than one statement is to be controlled by the conditional in the example above, we can use `BEGIN` and `END` like this:

```sql
IF DATEPART(dw, GETDATE()) = 7 OR DATEPART(dw, GETDATE()) = 1
BEGIN
    PRINT 'It is the weekend.';
    PRINT 'Get some rest on the weekend!';
END;
ELSE
BEGIN
    PRINT 'It is a weekday.';
    PRINT 'Get to work on a weekday!';
END;
```

`WAITFOR` will wait for a given amount of time, or until a particular time of day. The statement can be used for delays or to block execution until the set time.

`RETURN` is used to immediately return from a <u>stored procedure</u> or function.

`BREAK` ends the enclosing `WHILE` loop, while `CONTINUE` causes the next iteration of the loop to execute. An example of a `WHILE` loop is given below.

```sql
DECLARE @i INT;
SET @i = 0;

WHILE @i < 5
BEGIN
    PRINT 'Hello world.';
    SET @i = @i + 1;
END;
```

Changes to DELETE and UPDATE statements

In Transact-SQL, both the `DELETE` and `UPDATE` statements are enhanced to enable data from another table to be used in the operation, without needing a subquery:

- `DELETE` accepts joined tables in the `FROM` clause, similarly to `SELECT`. When this is done, the name or alias of which table in the join is to be deleted from is placed between `DELETE` and `FROM`.
- `UPDATE` allows a `FROM` clause to be added. The table to be updated can be either joined in the `FROM` clause and referenced by alias, or referenced only at the start of the statement as per standard SQL.

This example deletes all `users` who have been flagged with the 'Idle' flag.

```
DELETE u
  FROM users AS u
  INNER JOIN user_flags AS f
    ON u.id = f.id
    WHERE f.name = 'idle';
```

BULK INSERT

BULK is a Transact-SQL statement that implements a bulk data-loading process, inserting multiple rows into a table, reading data from an external sequential file. Use of `BULK INSERT` results in better performance than processes that issue individual `INSERT` statements for each row to be added. Additional details are available in MSDN (http://msdn2.microsoft.com/en-us/library/ms188365.aspx).

TRY CATCH

Beginning with SQL Server 2005,[1] Microsoft introduced additional `TRY CATCH` logic to support exception type behaviour. This behaviour enables developers to simplify their code and leave out `@@ERROR` checking after each SQL execution statement.

```
-- begin transaction
BEGIN TRAN;

BEGIN TRY
    -- execute each statement
    INSERT INTO MYTABLE(NAME) VALUES ('ABC');
    INSERT INTO MYTABLE(NAME) VALUES ('123');

    -- commit the transaction
    COMMIT TRAN;
END TRY
BEGIN CATCH
    -- roll back the transaction because of error
    ROLLBACK TRAN;
END CATCH;
```

See also

- Adaptive Server Enterprise (Sybase)
- PL/SQL (Oracle)
- PL/pgSQL (PostgreSQL)
- SQL/PSM (ISO standard)
- Sys.sysobjects
- Tabular Data Stream

References

1. "T-SQL Improvements in SQL Server 2012" (https://www.infoq.com/news/2012/03/T-SQL-20 12), Jonathan Allen on Mar 19, 2012, infoq.com

External links

- Sybase Transact-SQL User's Guide (http://infocenter.sybase.com/help/index.jsp?topic=/com.sybase.help.ase_15.0.sqlug/html/sqlug/title.htm)
- Transact-SQL Reference for SQL Server (https://docs.microsoft.com/en-us/sql/t-sql/language-reference) (MSDN)

  (2000 (http://msdn2.microsoft.com/en-us/library/aa260642(SQL.80).aspx), 2005 (http://msdn2.microsoft.com/en-us/library/ms189826.aspx), 2008 (http://msdn.microsoft.com/en-us/library/bb510741(SQL.100).aspx), 2012 (https://docs.microsoft.com/en-us/previous-versions/sql/sql-server-2012/bb510741(v=sql.110)))

- Transact-SQL examples (http://www.tsql.info)

Retrieved from "https://en.wikipedia.org/w/index.php?title=Transact-SQL&oldid=1047584731"