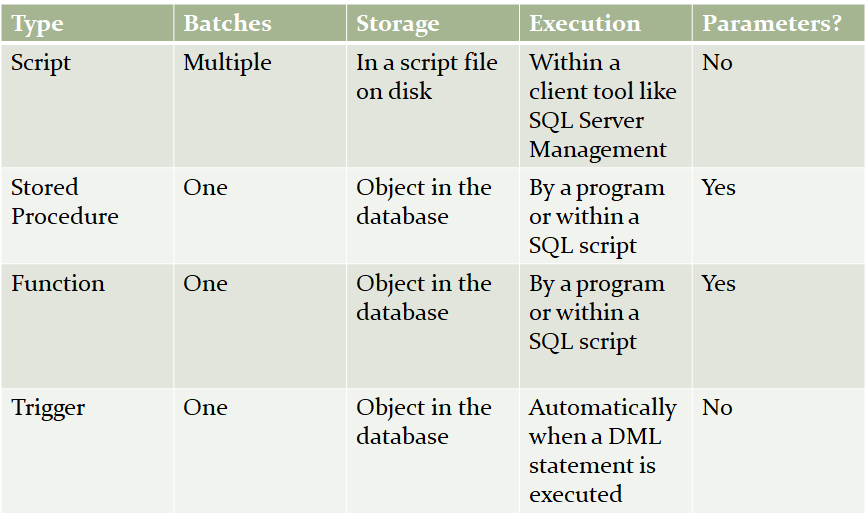
**Procedure**



**Description**

A stored procedure is an executable database object that contains SQL statements.

Stored procedures are precompiled.

Application programmers and end users don’t need to know the structure of the database or SQL.

Ability to restrict control access to the database.

Execute procedures using the EXEC command.

**Syntax Cont**

The EXECUTE\_AS\_clause allows users to execute the stored procedure with permissions specified by the clause.

To create a temporary stored procedure prefix the procedure with # for local, and ## for global; these procedures only exist while the current session is open.

You can use parameters to pass one or more value from the calling program to the stored procedure.

The name of a stored procedure is typically prepended with ‘sp.’

Execution: EXEC spProcdure\_name

**Example:**

CREATE PROC spInvTotal1

@DateVar smalldatetime,

@InvTotal money OUTPUT

AS

SELECT @InvTotal = SUM(InvoiceTotal)

FROM Invoices

WHERE InvoiceDate >= @DateVar;

CREATE PROC spInvTotal3

@InvTotal money OUTPUT,

@DateVar smalldatetime = NULL,

@VendorVar varchar(40) = '%'

AS

IF @DateVar IS NULL

SELECT @DateVar = MIN(InvoiceDate) FROM Invoices;

SELECT @InvTotal = SUM(InvoiceTotal)

FROM Invoices JOIN Vendors ON Invoices.VendorID = Vendors.VendorID

WHERE (InvoiceDate >= @DateVar) AND (VendorName LIKE @VendorVar);

**Return Example:**

CREATE PROC spInvCount @DateVar smalldatetime = NULL,

@VendorVar varchar(40) = ‘%’

AS

IF @DateVar IS NULL

SELECT @DateVar = Min(InvoiceDate) FROM Invoices;

DECLARE @InvCount int;

SELECT @InvCount = COUNT(InvoiceID)

FROM Invoices JOIN Vendors ON Invoices.VendorID = Vendors.VendorID

WHERE (InvoiceDate >= @DateVar) AND (VendorName LIKE @VendorVar);

RETURN @InvCount;

**User-Defined Functions**

A User-Defined Function (UDF or ‘ a function’) is an executable database object that contains SQL statements.

The name can be up to 128 characters

Typically prefixed with the letters ‘fn’

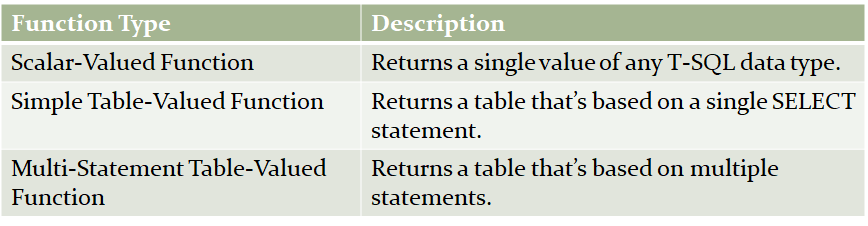
Functions Always return a value

Scalar-Valued Function returns a single value of any T-SQL data type

Table-Valued Function returns an entire table

If based on a single SELECT statement -> ‘simple table valued function’

If based on multiple SQL statements -> ‘multi-statement table-valued function’

****

**Function vs Procedure**

Function must return single value

only input parameter

no permanent changes

no try catch

Procedure can return zero, one, multiple values

input and output parameters

not with select having where

can use try catch

**Trigger**

A procedure which is executed after specific action queries

Action queries: essentially (but not always) DML statements.

Executed on a table or view.

Cannot trigger directly, pass parameters to a trigger, or expect a trigger to return a value back.

Within triggers you refer to the Inserted and Deleted tables created by the system.

FOR ANSI-standard Executes the trigger’s action query after the original action query

AFTER Introduced in SQL Server 2000 Same as FOR

INSTEAD OF Executes the trigger’s action query instead of the original action query

**After Trigger**

A table can have multiple AFTER triggers

A view can’t have an AFTER trigger

If there is an error in the original query the AFTER action query is never executed

**EVENTDATA**

When a trigger is created for a DDL statement the EVENTDATA() function becomes available.

EVENTDATA() returns an XML document that contains all information about the event causing the trigger to fire.

DECLARE @EventData xml;

SET @EventData = EVENTDATA();