# SQL Metadata Parser

Author: László Dobos  
Data: 06/22/2011

## Introduction

This tool is intended for parsing SQL scripts with XML comments so metadata information for databases can be created automatically from the script. Comment structure follows those of C# xml comments with a few modifications that makes writing comment easier. These modifications include the tool’s ability to figure out the context of certain comment tags, for example, automatic look-up of table names and column names is supported.

## Basics

SQL Server syntax uses – (double dash) for single line comments and /\* \*/ for multi-line comments. SQL XML comments can only be in single line comments and the double dash must be followed by the so called magic token, a / (slash). Comments not starting with --/ are not considered XML comments and are omitted.

## Tables

Tables can have various description fields and well as per-column descriptions. The following script demonstrates the usage.

CREATE TABLE dbo.TestTable (

---------------------------------------------------------------

--/ <summary>Table used for test</summary>

--/ <remarks>

--/ One can write remarks here.

--/ </remarks>

-------------------------------------------------------

ID INT NOT NULL, --/ <column>unique ID number</column>

raBore FLOAT NOT NULL,--/ <column unit="deg" content="pos.ra">right ascension</column>

decBore FLOAT NOT NULL --/ <column unit="deg" content="pos.dec">declination</column>

)

All comment lines referring to the table must be inside the parentheses of the CREATE TABLE command. Tags supported within a table description are: summary, remarks, column. The name of the table and the names of the columns will be inferred from the context. Column comments must be in the same line as the column names and the line must begin with the name of the column.

## Stored procedures

The following script demonstrates the usage of stored procedure comments.

CREATE PROC dbo.SomeProc (

--------------------------------------------------------------

--/ <summary>Multiplication</summary>

--/ <remarks>

--/ A simple stored procedure

--/ </remarks>

--------------------------------------------------------------

@param1 int, --/ <param>First value</param>

@param2 float --/ <param>Second column</param>

)

AS

SELECT @param1 \* @param2

Although the SQL syntax does not require, comments referring to the stored procedure must be enclosed in parentheses following the name of the procedure. Tags supported inside a stored procedure are: summary, remarks, param. Parameter columns must be in the same line as the parameter name, the line must start with the parameter name, and only one parameter per line is supported.

## Scalar functions

Comments for scalar functions are pretty much the same as for stored procedure except a tag referring to the return value is added as demonstrated with the following code snippet.

CREATE FUNCTION dbo.SomeFunc (

--------------------------------------------------------------

--/ <summary>Multiplication</summary>

--/ <remarks>

--/ A simple user-defined function

--/ </remarks>

--/ <returns>The product of the values</returns>

--------------------------------------------------------------

@param1 int, --/ <param>First value</param>

@param2 float --/ <param>Second column</param>

)

RETURNS float

AS

BEGIN

RETURN @param1 \* @param2

END

The returns tag must also be inside the parentheses of the function. Restrictions for the parameter comments are the same as for stored procedures.

## Table-valued functions

Basically the same as scalar functions, except comments are allowed for the return table columns too, as it is demonstrated in the following snippet.

CREATE FUNCTION dbo.TestFunction2 (

--/ <summary>Number sequence</summary>

--/ <remarks>

--/ A simple user-defined function

--/ </remarks>

--/ <returns>The numbers</returns>

--------------------------------------------------------------

@param1 int, --/ <param>This is a parameter</param>

@param2 int --/ <param>Another parameter</param>

)

RETURNS

@res TABLE

(

col int --/ <column>Results table column</column>

)

AS

BEGIN

DECLARE @i int = @param1;

WHILE (@i <= @param2)

BEGIN

INSERT @res (col) VALUES (@i);

SET @i = @i + 1

END

RETURN

END

For column comments, the same restrictions apply as before: they must be between the two prentheses of the table definition following the RETURNS keyword, must be in the same line as the column definition and the column name must be the first word in the line.

## Enums

Although SQL Server has no concept for constants or enumerations, this metadata toolkit provides a way of describing predefined values of columns.