عنوان: **استفاده از SQLDom برای آنالیز عبارات T-SQL**
نویسنده: وحید نصیری
تاریخ: ۱۳۹۱/۱۰/۰۶ ۲۲:۱۰
آدرس: www.dotnettips.info
برچسب‌ها: SQL Server, T-SQL

به همراه بسته Features pack اس کیوال سرور 2012، دو بسته SqlDom.msi نیز وجود دارند (نسخه‌های X86 و X64). این بسته حاوی اسمبلی Microsoft.SqlServer.TransactSql.ScriptDom.dll می‌باشد که نهایتا در آدرس Program Files\Microsoft SQL Server\110\SDK\Assemblies کپی خواهد شد.
به کمک آن می‌توان عبارات پیچیده T-SQL را Parse و آنالیز کرد. البته باید در نظر داشت هرچند این بسته جهت SQL Server 2012 ارائه شده اما این اسمبلی با نگارش‌های 2005 به بعد اس کیوال سرور کاملا سازگار است و اساسا نیازی هم به SQL Server ندارد. در ادامه مروری خواهیم داشت بر نحوه استفاده از آن.

یافتن کوئری‌های * Select در بین انبوهی از اسکریپت‌ها به کمک SQLDom

در مورد مضرات کوئری‌های * select پیشتر مطلبی را در این سایت خوانده‌اید. در ادامه قصد داریم به کمک امکانات اسمبلی Microsoft.SqlServer.TransactSql.ScriptDom.dll، تعدادی عبارت T-SQL را آنالیز کرده و مشخص کنیم که آیا حاوی * select هستند یا خیر. کد کامل آن‌را در ذیل مشاهده می‌کنید:

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Text;
using Microsoft.SqlServer.TransactSql.ScriptDom;

namespace DbCop
{
    // Microsoft® SQL Server® 2012 Transact-SQL ScriptDom
    // SQL Server 2012 managed parser, Supports SQL Server 2005+
    // SQLDom.msi (redist x86/x64)
    // http://www.microsoft.com/en-us/download/details.aspx?id=29065
    // X86: http://go.microsoft.com/fwlink/?LinkID=239634&clcid=0x409
    // X64: http://go.microsoft.com/fwlink/?LinkID=239635&clcid=0x409
    // Program Files\Microsoft SQL
Server\110\SDK\Assemblies\Microsoft.SqlServer.TransactSql.ScriptDom.dll

    class Program
    {
        static void Main()
        {
            const string tSql = @"
                -- select * in PROCEDURE
                CREATE PROCEDURE dbo.SelectStarTest
                AS
                SELECT * FROM dbo.tbl1
                go

                -- select * in PROCEDURE with TableVar
                Create PRocedure SelectAll
                AS
                Declare @X table(Id integer)
                Select * from @x
                go

                -- select * in PROCEDURE with ctex
                CREATE PROCEDURE dbo.SelectAllCte
                AS
                WITH ctex
                AS (
                SELECT * FROM sys.objects
                )
                SELECT * FROM ctex
                go

                -- normal select *
                select * from tbl1;
                select * from dbo.tbl2;
            ";
```

```
            IList<ParseError> errors;
            TSqlScript sqlFragment;
            using (var reader = new StringReader(tSql))
            {
                var parser = new TSql110Parser(initialQuotedIdentifiers: true);
                sqlFragment = (TSqlScript)parser.Parse(reader, out errors);
            }

            if (errors != null && errors.Any())
            {
                var sb = new StringBuilder();
                foreach (var error in errors)
                    sb.AppendLine(error.Message);

                throw new InvalidOperationException(sb.ToString());
            }

            var i = 0;
            foreach (var batch in sqlFragment.Batches)
            {
                Console.WriteLine("Batch: {0}, Statement(s): {1}", ++i, batch.Statements.Count);
                foreach (var statement in batch.Statements)
                {
                    processStatement(statement);
                }
                Console.WriteLine();
            }

            Console.WriteLine("\nPress a key...");
            Console.Read();
        }

        private static void processStatement(TSqlStatement statement)
        {
            var createProcedureStatement = statement as CreateProcedureStatement;
            if (createProcedureStatement != null)
            {
                var statementList = createProcedureStatement.StatementList;
                foreach (var procedureStatement in statementList.Statements)
                {
                    processStatement(procedureStatement);
                }
            }

            var selectStatement = statement as SelectStatement;
            if (selectStatement != null)
            {
                var query = selectStatement.QueryExpression;
                var selectElements = ((QuerySpecification)query).SelectElements;
                foreach (var selectElement in selectElements)
                {
                    var expression = selectElement as SelectStarExpression;
                    if (expression == null) continue;
                    Console.WriteLine(
                        "`Select *` detected @StartOffset:{0}, Line:{1}, T-SQL: {2}",
                        expression.StartOffset,
                        expression.StartLine,
                        statementToString(selectStatement));
                }
            }
        }

        private static string statementToString(TSqlFragment selectStatement)
        {
            var text = new StringBuilder();
            for (var i = selectStatement.FirstTokenIndex; i <= selectStatement.LastTokenIndex; i++)
            {
                text.Append(selectStatement.ScriptTokenStream[i].Text);
            }
            return text.ToString();
        }
    }
}
```

توضیحات:

پس از نصب SQLDom.msi، ارجاعی را به اسمبلی زیر اضافه نمائید تا بتوانید کد فوق را کامپایل کنید:

Program Files\Microsoft SQL Server\110\SDK\Assemblies\Microsoft.SqlServer.TransactSql.ScriptDom.dll

کار با ایجاد وهله‌ای از TSql110Parser شروع می‌شود. متد Parse آن، آرگومانی از نوع TextReader را قبول می‌کند. برای مثال با استفاده از StringReader می‌توان محتوای یک متغیر رشته‌ای را به آن ارسال کرد و یا توسط StreamReader یک فایل sql را.
پس از فراخوانی متد Parse، بهتر است بررسی شود که آیا عبارت T-SQL دریافتی معتبر بوده است یا خیر. اینکار را توسط لیستی از ParseErrorهای دریافتی می‌توان انجام داد.
خروجی متد Parse، حاوی یک سری Batch آنالیز شده است. هر عبارت Go در اینجا یک Batch را تشکیل می‌دهد. سپس در داخل هر batch به دنبال batch.Statements خواهیم گشت تا بتوان به عبارات T-SQL آن‌ها دسترسی یافت.
در ادامه کار اصلی توسط متد processStatement صورت می‌گیرد. عبارات دریافتی، در حالت کلی از نوع TSqlStatement هستند اما در اصل می‌توانند یکی از مشتقات آن نیز باشند. در اینجا فقط دو مورد CreateProcedureStatement و SelectStatement بررسی شده‌اند (مطابق رشته tSql ابتدای مثال). هر دو عبارت، از کلاس TSqlStatement مشتق شده‌اند.
در متد processStatement عبارات select معمولی و همچنین آن‌هایی که داخل رویه‌های ذخیره شده تعریف شده‌اند، استخراج شده و در نهایت بررسی می‌شوند که آیا از نوع SelectStarExpression هستند یا خیر (همان * select صورت مساله).
خروجی مثال فوق به شرح زیر است:

```
Batch: 1, Statement(s): 1
`Select *` detected @StartOffset:140, Line:5, T-SQL: SELECT * FROM dbo.tbl1

Batch: 2, Statement(s): 1
`Select *` detected @StartOffset:368, Line:12, T-SQL: Select * from @x

Batch: 3, Statement(s): 1
`Select *` detected @StartOffset:659, Line:22, T-SQL: WITH ctex
                AS (
                SELECT * FROM sys.objects
                )
                SELECT * FROM ctex

Batch: 4, Statement(s): 2
`Select *` detected @StartOffset:753, Line:26, T-SQL: select * from tbl1;
`Select *` detected @StartOffset:791, Line:27, T-SQL: select * from dbo.tbl2;
```