عنوان مقاله: یافتن UDFهایی که موجب کاهش کارایی SQL Server می شوند

نویسنده مقاله: سید محمد حسینی

تاریخ انتشار: دی ماه ۱۳۹۴

منبع: https://nikamooz.com/find-udf-that-sql-server-performance-will-be-decreased

مقدمه

چند وقت پیش، در حالی که مشغول کار بر روی بهینه سازی کارایی بانک اطلاعاتی یکی از سیستم های شرکت بودم، با یک SP مواجه شدم که در آن عملیاتی نظیر عملیات زیر انجام می شد:

- یک SELECT که نتیجه آن در یک جدول موقت قرار می گرفت
- سپس یک INSERT که داده ها را از جدول موقت می خواند
- برخی دستکاری داده ها
- و در انتها درج داده ها در یک جدول دیگر جهت اهداف گزارش گیری

پس از بررسی بیشتر و تجزیه و تحلیل این فرآیند، دریافتم که مشکل واقعی، استفاده گسترده از توابع-اسکالر- تعریف شده کاربر (UDF) می باشد. با شناسایی این مشکل و بازسازی SP مربوطه، توانستیم مدت زمان اجرا را از ۹۰ دقیقه تا کمتر از ۵ دقیقه کاهش دهیم. این مثال می تواند نشان از اهمیت شناسایی و رفع مشکلات ناشی از کارایی UDFها باشد.

در ادامه برخی از مسائل ناشی از UDFها و برخی از پرسوجوهایی که می توان توسط آنها محل استفاده از UDFها و همچنین تأثیرات آنها بر کارایی را بدست آورد، بررسی می شود. بنابراین در انتها شما می دانید که در کجای کد باید بهینه سازی را جهت بهبود کارایی انجام دهید.

چرا توابع اسکالر SQL Server بد هستند؟

توابع SQL Server به دو دسته تقسیم می شوند:

- توابع اسکالر(Scalar Functions)
- توابعی که خروجی آنها جدول است(Table Valued Functions-نیک آموز TVFs)

اگرچه توابع برای کپسوله سازی و ساده سازی کد بسیار مفید هستند، ولی چندین مشکل در رابطه با کارایی UDFها، به خصوص توابع اسکالر وجود دارد. برخی از این مشکلات عبارتند از:

- توابع اسکالر مانند کرسر(cursor) عمل می کنند، به ازای هر سطر از پرسوجو که پردازش می شود، اجرا می شوند. این موضوع با توجه به رشد پرسوجو، به صورت نمایی موجب کاهش کارایی می شود.
- توابع اسکالر اجازه نمی دهند پرسوجو از منافع پردازش موازی(Parallelism) بهره مند شود، بنابراین اجرای پرسوجو را محدود به یک پردازنده می کنند.
- توابع اسکالر منجر به تخمین نادرست و کمتر از حد مطلوب هزینه ها در پلان اجرایی می شود.

توابع اسکالر به عنوان گلوگاه های پنهان(Hidden Bottlenecks) هستند، زیرا درک تاثیر آنها بر کارایی، با نگاه کردن به یک پلان اجرایی که در آن UDFها مشکل ساز هستند، بسیار دشوار است.
در اینجا فقط چند تکنیک جهت یافتن توابع اسکالر و میزان تاثیر آنها بر کارایی بررسی شده است. مطمئنا با بررسی های بیشتر در اینترنت می توان راهکارهای بهتر و بیشتری را جستجو نمود.

چگونه میزان تاثیر توابع اسکالر SQL Server بر کارایی را اندازه گیری کنیم؟

هنگامی که قصد ارزیابی کارایی تاثیر توابع اسکالر را داریم، به عنوان یک DBA باید برخی موارد را بدانیم:

- لیست اسامی توابع مورد استفاده و میزان/تعداد دفعات فراخوانی این توابع چقدر است؟
- تأثیر این توابع بر کارایی چیست – به عبارت دیگر، زمان و منابع مورد نیاز برای اجرای این توابع چقدر است؟

همانطور که قبلا نیز گفته شد، پلان اجرایی SQL برای تخمین کارایی توابع اسکالر ابزار مناسبی نیست، زیرا در هنگام استفاده از پرسوجوهای بزرگ میزان اصلی تاثیرات را منعکس نمی کند.
در ادامه چندین راه جهت بررسی و ارزیابی کارایی UDFها ارائه شده است که دارای قابلیت ها و محدودیت هایی می باشند.
استفاده از DMVها جهت به دست آوردن پلان اجرایی UDF به جای پلان SQL Server
این کوئری آمارهای مربوط به کارایی را برای توابعی که اخیرا فراخوانی شده اند نشان می دهد(نظیر تعداد اجراها، متوسط زمان سپری شده، متوسط خواندن های منطقی/فیزیکی، و غیره). توابعی که دارای تعداد دفعات اجرای بیشتری هستند، همچنین آنهایی که تعداد عملیات خواندن آنها بیشتر است بهترین کاندید برای بهینه سازی هستند. از نظر من این بهترین روش برای ارزیابی کارایی UDFها می باشد. البته این روش دارای محدودیت هاییی هم هست- تنها توابعی را که به داده ها دسترسی داشته باشند به دست خواهد آورد.
این کوئری تمام انواع UDFها، از جمله UDFهای اسکالر را باز می گرداند.

```
USE [TempStockAsset۴]
SELECT TOP ۱۰۰ DB_NAME() AS [DATABASE],
QS.total_worker_time / ۱۰۰۰۰۰۰AS TotalWorkerTime,
QS.total_elapsed_time/ ۱۰۰۰۰۰۰AS TotalElapsedTime_Sec,
QS.total_elapsed_time/(۱۰۰۰۰۰۰*qs.execution_count) AS [avg_elapsed_time_Sec],
QS.execution_count,
QS.total_logical_reads/QS.execution_count AS Avg_logical_reads,
QS.max_logical_writes, ST.text AS ParentQueryText,
SUBSTRING(ST.[text],QS.statement_start_offset/۲+۱,
(CASE
WHEN QS.statement_end_offset = -۱
THEN LEN(CONVERT(nvarchar(MAX), ST.[text])) * ۲
ELSE QS.statement_end_offset
END - QS.statement_start_offset)/۲) AS [Query Text] ,
QP.query_plan ,
O.type_desc
FROM sys.dm_exec_query_stats QS
CROSS APPLY sys.dm_exec_sql_text(QS.sql_handle) AS ST
CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) QP
LEFT JOIN Sys.objects O
ON O.object_id =St.objectid WHERE O.type_desc LIKE '%Function%'
ORDER BY qs.total_worker_time DESC;
```

استفاده ازDMV ها جهت به دست آوردن پلان اجرایی کوئری با ارجاع به UDF به جای پلانSQL Server

کوئری زیر، پلان های اجرایی SQL را جهت یافتن عبارت "ComputeScalar" که نشان دهنده عملیات رخ داده توسط توابع اسکالر هستند را به همراه آمار مربوط به کارایی این کوئری های را نمایش می دهد. البته این کوئری میزان منابع مورد استفاده توسط توابع را نمایش نمی دهد.

```
;WITH XMLNAMESPACES
(
DEFAULT 'http://schemas.microsoft.com/sqlserver/۲۰۰۴/۰۷/showplan'
)
, GetQueriesWithUDF AS
(
SELECT DISTINCT udf.value('(@FunctionName)[۱]','varchar(۱۰۰)') As UDF_Name,
RIGHT(udf.value('(@FunctionName)[۱]','varchar(۱۰۰)'),
LEN(udf.value('(@FunctionName)[۱]','varchar(۱۰۰)'))-
CHARINDEX('.',udf.value('(@FunctionName)[۱]','varchar(۱۰۰)'),
```

```
CHARINDEX('.',udf.value('(@FunctionName)[۱]',
'varchar(۱۰۰)'),۱)+۱)) As Stripped_UDF_Name,
QS.execution_count,
CONVERT(float,cs.value('(.//RelOp/@EstimateRows)[۱]',
'varchar(۱۰۰)')) As EstimatedRows,
QS.total_elapsed_time/۱۰۰۰۰۰۰ As TotalElapsedTime_Sec,
QS.max_elapsed_time/۱۰۰۰۰۰۰ As max_elapsed_time_Sec,
QS.total_elapsed_time/(۱۰۰۰۰۰۰*qs.execution_count) AS [avg_elapsed_time_Sec],
QS.total_logical_reads,
QS.total_logical_reads/qs.execution_count As avg_logical_reads,
SUBSTRING(ST.[text],QS.statement_start_offset/۲+۱,
(CASE
WHEN QS.statement_end_offset = -۱
THEN LEN(CONVERT(nvarchar(max), ST.[text])) * ۲
ELSE QS.statement_end_offset
END - QS.statement_start_offset)/۲) AS [Query_Text],
ST.text As Parent_Query
FROM sys.dm_exec_query_stats QS
CROSS APPLY sys.dm_exec_sql_text (QS.sql_handle ) ST
CROSS APPLY sys.dm_exec_query_plan(qs.plan_handle) AS qp
CROSS APPLY qp.query_plan.nodes('.//ComputeScalar') AS CompCsalar(cs)
CROSS APPLY cs.nodes('.//UserDefinedFunction[@FunctionName]') AS fn(udf)
WHERE DB_Name(qp.dbid) not in ('msdb','master','tempdb')
)
SELECT *
FROM GetQueriesWithUDF
WHERE Query_text LIKE '%'+Stripped_UDF_Name+'%'
;ORDER BY EstimatedRows DESC OPTION(MAXDOP ۱, RECOMPILE)
```

استفاده از SQL Server Extended Events جهت به دست آوردن تعداد UDFها

این کوئری یک سشن Extended Events جهت به دست آوردن رویدادهای module_end که بر اساس نوع شیء تابع فیلتر شده اند را ایجاد می کند. باید توجه داشت که پیش از اجرای این اسکریپت باید نام بانک اطلاعاتی مورد نظر شما با AdventureWorks۲۰۱۴ جایگزین گردد. جهت به حداقل رساندن تأثیر تریس بر روی سرور، در این مثال فقط تعداد کل اجراها به ازای هر UDF جمع آوری می شود. با این حال، توصیه می شود این اسکریپت را بر روی سرور اصلی اجرا نکنید. همچنین این روش با SQL ۲۰۱۲ و نسخه های بعد به خوبی کار می کند، اما ممکن است با نسخه های قدیمی تر به درستی کار نکند.

```
 class="xml" style="font-family:monospace;"> CREATE EVENT SESSION [UDF_Trace]>
ON SERVER
)ADD EVENT sqlserver.module_end
ACTION(sqlserver.database_name,sqlserver.sql_text)
('WHERE ([sqlserver].[equal_i_sql_ansi_string]([object_type],'FN
(('AND [sqlserver].[database_name]=N'AdventureWorks۲۰۱۴
)ADD TARGET package۰.histogram
,'SET filtering_event_name=N'sqlserver.module_end
((source=N'object_name',source_type=(۰
,WITH (MAX_MEMORY=۴۰۹۶ KB,EVENT_RETENTION_MODE=ALLOW_SINGLE_EVENT_LOSS
,MAX_DISPATCH_LATENCY=۳۰ SECONDS,MAX_EVENT_SIZE=۰ KB
(MEMORY_PARTITION_MODE=NONE,TRACK_CAUSALITY=OFF,STARTUP_STATE=OFF
GO
ALTER EVENT SESSION [UDF_Trace] ON SERVER STATE=START
<GO
```

استفاده از تریس سمت سرور در SQL Server جهت به دست آوردن اطلاعاتی در مورد اجرای UDFها

این کوئری یک تریس سمت سرور ایجاد می کند که رویداد SP:Completed را جمع آوری کرده و فیلتر آن بر اساس ObjectType=۲۰۰۳۸ می باشد. توجه داشته باشید که این تریس در صورت اجرا بر روی سرور اصلی و وجود تعداد زیادی رویداد، می تواند به طور جدی مشکلات کارایی زیادی برای سرور ایجاد نماید.

در اینجا از تریس سمت سرور بجای تریس های برنامه Profiler استفاده شده است تا حجم بار کاری سرور کاهش داده شود. در صورتی که با تریس های سمت سرور آشنایی ندارید، این تریس ها در پس زمینه اجرا شده و می توان آنها را توسط دستورات TSQL اجرا/متوقف نمود. نتایج به دست آمده نیز توسط برنامه SQL Profiler یا دستورات TSQL قابل بررسی هستند.

کوئری مورد نظر در زیر ارائه شده است.

```
-- Create a Queue
declare @rc int
declare @TraceID int
declare @maxfilesize bigint,@FilePath nvarchar(۵۰۰)=N'C:\Data\UDFTrace'
set @maxfilesize = ۵
-- Please replace the text InsertFileNameHere, with an appropriate
-- filename prefixed by a path, e.g., c:\MyFolder\MyTrace. The .trc extension
-- will be appended to the filename automatically. If you are writing from
-- remote server to local drive, please use UNC path and make sure server has
-- write access to your network share
```

```
exec @rc = sp_trace_create @TraceID output, ۰, @FilePath, @maxfilesize, NULL
if (@rc != ۰) goto error
-- Client side File and Table cannot be scripted
-- Set the events
declare @on bit
set @on = ۱
exec sp_trace_setevent @TraceID, ۱۰, ۱۲, @on
exec sp_trace_setevent @TraceID, ۱۰, ۱۳, @on
exec sp_trace_setevent @TraceID, ۱۰, ۱۴, @on
exec sp_trace_setevent @TraceID, ۱۰, ۳۴, @on
exec sp_trace_setevent @TraceID, ۱۰, ۳۵, @on
exec sp_trace_setevent @TraceID, ۴۵, ۱۲, @on
exec sp_trace_setevent @TraceID, ۴۵, ۱۳, @on
exec sp_trace_setevent @TraceID, ۴۵, ۱۴, @on
exec sp_trace_setevent @TraceID, ۴۵, ۲۸, @on
exec sp_trace_setevent @TraceID, ۴۵, ۳۴, @on
exec sp_trace_setevent @TraceID, ۴۵, ۳۵, @on
-- Set the Filters
declare @intfilter int
declare @bigintfilter bigint
set @intfilter = ۲۰۰۳۸
exec sp_trace_setfilter @TraceID, ۲۸, ۰, ۰, @intfilter
set @intfilter = NULL
exec sp_trace_setfilter @TraceID, ۲۸, ۰, ۱, @intfilter
-- Set the trace status to start
exec sp_trace_setstatus @TraceID, ۱
-- display trace id for future references
select TraceID=@TraceID
goto finish
error:
select ErrorCode=@rc
finish:
go
```

جهت اجرای این کوئری باید ابتدا مسیر دلخواهی را برای متغییر @FilePath مشخص کرده و سپس کوئری را اجرا نمود. پس از اجرای کوئری، IDاتولید شده را باید جهت استفاده های بعدی یادداشت نمود. مثلا برای متوقف کردن تریس به این ID نیاز می باشد. پس از اتمام عملیات و جمع آوری اطلاعات مورد نظر، IDمربوطه را در کوئری ریز جایگزین @TraceID کرده و برای متوقف کردن تریس این کوئری را اجرا نمایید.

```
class="xml" style="font-family:monospace;">exec sp_trace_setstatus @TraceId, ۰
exec sp_trace_setstatus @TraceId, ۲
```

همچنین می توان جهت بررسی اطلاعات جمع آوری شده توسط تریس، از کوئری زیر استفاده نمود(مسیر مشخص شده باید با مسیر مورد نظر شما جایگزین گردد)

```
SELECT databasename,objectname,sum(duration)
FROM fn_trace_gettable('c:\data\UDFTrace.trc',default)
GROUP BY databasename,objectname
```