3 главных вопроса про временные таблицы

Участников наших курсов часто интересуют внутренняя логика работы временных таблиц.

Какие критерии индексирования временных таблиц? Где они хранятся? Нужно ли в явном виде их удалять?

Ответы на эти вопросы мы рассмотрим в данной статье.

Также мы приведем случаи, когда **не нужно использовать временные таблицы**.

Где хранятся временные таблицы?

Начнем с того, что **временные таблицы** — это объекты СУБД, никаких временных таблиц на сервере 1С нет, не путайте пожалуйста, их с таблицами значений.

Под вопросом *«Где хранятся временные таблицы?»*, имеется ввиду физическое расположение, т.е. либо жесткий диск, либо оперативная память.

Вероятно, этот вопрос чаще других остается без ответа, либо ответы на него кардинально различаются.

Но все сходятся в том, что временные таблицы создаются и хранятся в базе *TempDB.*

Действительно, все временные таблицы относятся к базе данных *TempDB*, но это вовсе не значит, что они обязательно будут записываться на диск.

В компьютере нет железяки под названием *TempDB*, но в нем есть такие железяки как оперативная память и жесткий диск.

Правильный ответ на этот вопрос звучит так: **все временные таблицы по умолчанию создаются в оперативной памяти, а именно в буферном кэше.**

Конечно, есть исключения. Например, если таблица слишком большая, то сервер может принять решение сбросить ее на диск. Также возможна ситуация, когда сервер по каким-либо причинам решил отдать память под другие данные, тогда таблица тоже будет записана на диск.

Почему таблица создается именно в памяти? Тут все очевидно, дело в производительности, думаю не стоит объяснять, что чтение из оперативной памяти гораздо быстрее чтения с диска, даже если этот диск SSD.

Наверняка найдутся скептики, которые захотят проверить мои слова и это правильно. Все лгут, и все надо перепроверять, давайте этим и займемся.

Пишем следующий запрос в консоли:

```
ВЫБРАТЬ     1 КАК Поле1 ПОМЕСТИТЬ ВТ
```

Запускаем трассировку *SQL Profiler* с событием *SQL:BatchComplited,* выполняем запрос в консоли и получаем следующий текст *SQL* запроса:

```
INSERT INTO #tt1 (_Q_001_F_000) SELECT 1.0
```

Здесь мы видим только заполнение временной таблицы, т.к. код создания временных таблиц в нашей трассировке не отображается.

Чтобы понять, где создается временная таблица, необходимо понять откуда читаются данные, с диска или из памяти. Для этого используем показатель *physical reads* (количество физических чтений), т.е. сколько 8Кб страниц данных было прочитано с диска для выполнения запроса.

Чтобы получить значение этого показателя, необходимо выполнить создание и чтение временной таблицы в *Management Studio*.

Создаем новый запрос и пишем следующее:

```
create table #tt1 (_Q_001_F_000 int); -- создаем локальную временную таблицу tt1
INSERT INTO #tt1 (_Q_001_F_000) SELECT 1.0 – заполняем таблицу
set statistics io on; -- включаем вывод статистики ввода/вывода
select * from #tt1 -- читаем данные из таблицы
set statistics io off; -- выключаем вывод статистики
drop table #tt1 -- удаляем таблицу
```

После выполнения данного кода на закладке *«Сообщения»* получим следующий текст:

```
(строк обработано: 1)
Таблица «#tt1_____000000000066”.
Число просмотров 1, логических чтений 1, физических чтений 0,
упреждающих чтений 0, lob логических чтений 0, lob физических чтений 0,
lob упреждающих чтений 0.
```

Самое важное здесь то, что данные с диска не читались, т.к. число физических чтений равно 0, при этом есть 1 логическое чтение, т.е. данные были прочитаны только из памяти.

Отсюда можно сделать вывод, что временные таблицы в большинстве случаев создаются и хранятся в оперативной памяти, исключения уже описаны выше.

Надо ли индексировать временные таблицы?

На дисках ИТС, на экзамене 1С: Эксперт, да и я на своих курсах говорю, что нужно индексировать поля условий и соединений во временных таблицах.

Эта рекомендация настолько очевидна, что уже практически никто не подвергает ее сомнению.

Но как показывает практика, чаще всего индексы во временных таблицах не используются, либо используются, но запрос выполняется еще медленнее чем без них.

Скажу даже больше, для себя я сделал вывод, что польза от индексов на временных таблицах скорее исключение, чем правило.

Это не значит, что индексы не нужно использовать, это значит, что необходимо анализировать каждый конкретный случай и случаев, когда индекс не нужен, значительно больше.

Именно поэтому на курсах по оптимизации постоянно делается акцент на то, что с каждой проблемой надо разбираться отдельно, а не слепо выполнять рекомендации, не понимая почему эти рекомендации именно такие.

Давайте рассмотрим ситуацию с индексацией на примере.

Создадим временную таблицу с одним числовым полем и значениями от 1 до 1 млн.

Это можно сделать с помощью следующего пакетного запроса:

```
ВЫБРАТЬ 0 КАК Цифра

ПОМЕСТИТЬ ВТ_Цифры

ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 1 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 2 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 3 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 4 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 5 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 6 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 7 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 8 ОБЪЕДИНИТЬ ВСЕ ВЫБРАТЬ 9

; ////////////////////////////////////////////////////////////////////////////

ВЫБРАТЬ 100000 * Таб6.Цифра + 10000 * Таб5.Цифра + 1000 * Таб4.Цифра + 100 * Таб3.Цифра + 10 * Таб2.Цифра + Таб1.Цифра + 1 КАК Число

ПОМЕСТИТЬ ВТ_Числа ИЗ ВТ_Цифры КАК Таб1, ВТ_Цифры КАК Таб2, ВТ_Цифры КАК Таб3, ВТ_Цифры КАК Таб4, ВТ_Цифры КАК Таб5, ВТ_Цифры КАК Таб6

; ////////////////////////////////////////////////////////////////////////////

ВЫБРАТЬ ВТ_Числа.Число ИЗ ВТ_Числа КАК ВТ_Числа ГДЕ ВТ_Числа.Число = 777
```

Весь запрос выполняется в среднем за 1.2 секунды.

Если посмотреть трассировку *SQL Profiler*, то мы увидим следующее:

EventClass	Duration	Reads	RowCounts	TextData
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	0	0	0	SELECT spid, blocked FROM master..sysprocesses WHERE block...
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	0	24	10	INSERT INTO #tt1 (_Q_001_F_000) SELECT 0.0 UNION ALL SELEC...
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	1112	335360	1000000	INSERT INTO #tt2 (_Q_001_F_000) SELECT (((((((100000.0 * T...
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
RPC:Completed	107	2233	1	exec sp_executesql N'SELECT T1._Q_001_F_000 FROM #tt2 T1 W...
SQL:BatchCompleted	0	30	0	TRUNCATE TABLE #tt1
SQL:BatchCompleted	0	45	0	TRUNCATE TABLE #tt2
Trace Pause				
Trace Start				

Просмотр строк таблицы
[#tt2] [T1]
Стоимость: 100 %

На создание таблицы уходит 1.1 секунда и еще 0.1 секунда на сканирование всей таблицы, чтобы вернуть нам 1 строку.

Давайте посмотрим, что изменится, если добавить индекс в таблицу *ВТ_Числа*.

На моем компьютере запрос стал выполняться в среднем за 6 секунд.

EventClass	Duration	Reads	RowCounts	TextData
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	0	24	10	INSERT INTO #tt1 (_Q_001_F_000) SELECT 0.0 UNION ALL SELEC...
SQL:BatchCompleted	0	26	0	create index [TMPIND_0] on [#tt3] (_Q_001_F_000)
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	5307	442...	1000000	INSERT INTO #tt3 (_Q_001_F_000) SELECT (((((((100000.0 * T...
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
SQL:BatchCompleted	0	0	0	SELECT spid, blocked FROM master..sysprocesses WHERE block...
Showplan Statistics Profile				
Showplan XML Statistics P...				<ShowPlanXML xmlns="http://schemas.microsoft.com/sqlserver...
RPC:Completed	588	2057	2	exec sp_executesql N'SELECT T1._Q_001_F_000 FROM #tt3 T1 W...
SQL:BatchCompleted	0	30	0	TRUNCATE TABLE #tt1
SQL:BatchCompleted	0	49	0	DROP INDEX #tt3.TMPIND_0
SQL:BatchCompleted	0	45	0	TRUNCATE TABLE #tt3
Trace Pause				

Поиск в индексе (NonClustered)
[#tt3].[TMPIND_0] [T1]
Стоимость: 100 %

Время создания таблицы увеличилось с 1 секунды до 5.3, при этом даже поиск по индексу в таблице все равно происходит медленнее, чем сканирование 0.5 сек. против 0.1 без индекса. Единственное, в чем этот запрос выигрывает — немного меньше логических чтений, 2057 против 2233 при сканировании.

Используйте индексирование только в том случае, если вы видите от этого явный положительный эффект.

Надо ли явно удалять временные таблицы после создания?

Ответ будет зависеть от способа создания временной таблицы.

Если Вы используете временную таблицу только в одном пакетном запросе, то ваша таблица «живет» пока выполняется этот пакетный запрос. Это значит, что менеджер временных таблиц был создан не явно.

В данном случае *MS SQL* создает локальную временную таблицу с одной решеткой (#), например, *#tt1*.

Как только пакетный запрос завершается, неявный *МВТ* закрывается и автоматически последует команда *«Truncate table»*, которая удаляет созданную таблицу.

Если временная таблица проиндексирована, то сначала будет удален индекс и только потом таблица.

Пример можно посмотреть выше в трассировке.

В данном случае нет необходимости использовать команду *«УНИЧТОЖИТЬ»*, только если Вы не хотите создать в том же запросе новую таблицу с таким же именем, ну или считаете это хорошим стилем написания кода.

Здесь главное понимать, что таблица все равно будет удалена при завершении пакетного запроса.

Ситуация меняется если Вы явно используете *менеджер временных таблиц (МВТ)*, т.е. создаете соответствующий объект метаданных. В этом случае *MS SQL* создает глобальную временную таблицу с двумя решетками (##).

Такая таблица будет удалена в любом из следующих вариантов:

1. в запросе использована команда УНИЧТОЖИТЬ
2. вызван метод МенеджерВременныхТаблиц.Закрыть()

3. объект МенеджерВременныхТаблиц перестал существовать, например, завершилась работа процедуры/функции которая породила этот объект или пользователь закрыл программу

Если Вы используете объект МВТ, то временные таблицы рекомендуется удалять одним из первых 2х методов, как только в них отпала необходимость, иначе они будут висеть в памяти сервера СУБД, пока процедура/функция не закончит работу, что не есть хорошо. Если же у Вас процедура в которой был создан МВТ завершается как раз выполнением запроса, тогда конечно МВТ можно не удалять т.к. сработает 3 условие.

Подведем итог, если Вы не используете МВТ, то явно удалять временную таблицу не требуется, она будет удалена после завершения пакетного запроса.

Если явное объявление МВТ используется, то рекомендуется удалить таблицу вручную, например, в запросе командой *«Уничтожить»*, либо методом *МВТ.Закрыть()*.

Минусы временных таблиц

Идеальных инструментов не бывает, тем более в мире 1С.

Давайте рассмотрим, какие проблемы может принести активное использование временных таблиц.

Чрезмерное разрастание базы TempDB

Если Вы активно используете временные таблицы, то у Вас может довольно сильно разрастаться база *TempDB* и в один прекрасный день может занять все свободное место на диске. Размер *TempDB* автоматически только увеличивается, но не уменьшается. Внутри файла место может как заниматься, так и освобождаться, но сам размер файла только увеличивается.

Типичная ситуация, установили обновление, и через несколько дней *TempDB* раздуло до неимоверных размеров, а потом выясняется, что разработчики переписали запросы с использованием временных таблиц, причем таблицы эти внушительного размера и их много.

Для исправления ситуации необходимо выполнить следующие команды:

```
dbcc shrinkfile (tempdev, ЖелаемыйРазмерФайлаДанныхМб)
dbcc shrinkfile (templog, ЖелаемыйРазмерФайлаЛоговМб)
```

Чрезмерное упрощение запросов

Нельзя сказать, что это очень большой минус, но все же.

Часто временные таблицы используются как раз для упрощения сложных запросов, чтобы серверу было легче подобрать оптимальный план, и это правильно.

Но можно встретить и другую крайность, запрос, который вполне можно было бы написать без временных таблиц, он бы работал быстро и оптимально, все равно пишут через временные таблицы. Просто разработчику лень думать, как это можно сделать по-другому.

Если можно написать оптимальный запрос без использования временных таблиц, то лучше обойтись без них.

Запрос с оптимальным кодом без временных таблиц в любом случае будет работать быстрее и использовать меньше ресурсов, чем запрос с временными таблицами.

Заключение

После прочтения статьи у Вас может сложиться мнение, что я не советую использовать временные таблицы, но это не так. Напротив, я активно их использую для оптимизации запросов, но делаю это только после того, как удостоверюсь в их эффективности в данном конкретном случае.

Временные таблицы — это инструмент для решения определенных задач, а не волшебная таблетка на все случаи жизни.

Бурмистров Андрей

Дополнительные материалы

Все статьи проекта **Курсы-по-1С.рф:** **http://курсы-по-1с.рф/blog/articles/**

Курсы по оптимизации в 1С v.8

Учебный курс «Оптимизация производительности 1С:Предприятие 8 и подготовка к 1С:Эксперт по технологическим вопросам»

http://курсы-по-1с.рф/ускорение-1с/

Курс по «1С:Конвертации Данных» — профессиональная настройка правил обмена и типовые сценарии переноса данных

http://курсы-по-1с.рф/data-conv/