ОПТИМИЗАЦИЯ ПРОИЗВОДИТЕЛЬНОСТИ ВЫПОЛНЕНИЯ ЗАПРОСОВ В MS SQL SERVER

ЛУКАШ Сергей

Технический Университет Молдовы

***Аннотация**: Статья посвящена способам оптимизации производительности выполнения запросов в MS SQL Server, в частном случае, путем использования индексации. В данной статье на примере показано, как с помощью плана запроса можно оценить текущую производительность запросов, а также, на реальном запросе выборки будет показано, как индексация влияет на производительность запроса.*

***Ключевые слова:** SQL-запрос, оптимизация, MS SQL Server, хранение данных, план выполнения, индексация.*

1. Понятие об SQL-запросе

SQL–запросы – это запросы, составляемые пользователями базы данных (БД) из последовательности SQL – инструкций. Запрос – это средство выборки, изменения и модификации необходимой информации в базе данных. SQL-инструкции задают набор операций над данными в БД для создания выходного набора необходимой информации.

Существуют следующие типы запросов: запросы на выборку, запросы на обновление, на добавление, на удаление, перекрестные запрос, запросы на создание таблиц и другие. Самым распространённым и чаще всего используемым является запрос на выборку данных. Такие запросы используются для отбора необходимой пользователю информации, содержащейся в таблицах.

2. Оптимизация

Для понятия «оптимизация» следует выделить три основных варианта, связанных с тематикой баз данных: общее понятие оптимизации, оптимизация производительности выполнения запроса и оптимизация запроса.

Наиболее общее понятие слова оптимизация объясняет ее следующим образом: это модификация системы с целью улучшения её эффективности. Данное понятие абстрактно отражает суть оптимизации – повышение эффективности и скорости работы системы.

Оптимизация производительности выполнения запроса – процесс поиска такого варианта исполнения запроса, при котором будет достигнута оптимальная производительность.

Оптимизация запроса являет собой изменение структуры запроса с целью максимизации скорости его выполнения.

Для оценки эффективности любой системы, в частности и для запроса, используются некоторые показатели. Для запросов, данные показатели могут быть найдены в плане выполнения запроса.

3. План выполнения и эффективность

План выполнения представляет собой последовательность операций, которые необходимо произвести над данными в реляционной системе управления базами данных (СУБД), чтобы получить результат SQL-запроса.

Для СУБД Microsoft SQL Server Management Studio план запроса может быть вызван с помощью кнопки «Показать предполагаемый план выполнения» или переключателя «Включить действительный план выполнения» на панели инструментов редактора запросов. Причем первый вариант не требует выполнения запроса. При активном переключателе, план будет выводится только в случае выполнения запроса. На рисунке 1, данные кнопка и переключатель обведены красными квадратами (кнопка подсвечена первым квадратом соответственно).

Рис.1. Способы вызова плана выполнения запроса

Полученный план выполнения, пример которого изображен на рисунке 2, позволяет оценить эффективность запроса. План читается справа налево, и представляет собой набор операций, производимых для получения результата запроса. Под каждой операцией находится название её типа

и её стоимость по отношению к всему пакету. Наведение курсора мыши на операцию, выводит на экран некоторые характеристики запроса, такие как тип физической и логической операции, количество прочитанных строк, фактическое количество строк, различные характеристики стоимости. Стоимость показывает приблизительные затраты на выполнение действий соответствующей операции. Эффективность запроса тем выше, чем меньше стоимость выполнения операции. Также, параметрами, характеризующими эффективность запроса, являются время CPU и затраченное время.

Время CPU и затраченное время не являются частью плана выполнения. Отображение данных характеристик включается двумя командами: SET STATISTICS IO ON и SET STATISTICS TIME ON. Время CPU показывает, сколько процессорного времени было затрачено на выполнение всего запроса, а затраченное время показывает время выполнения всего запроса. Как и в случае со стоимостью, тут действует принцип меньше-лучше, то есть меньше время – выше эффективность.

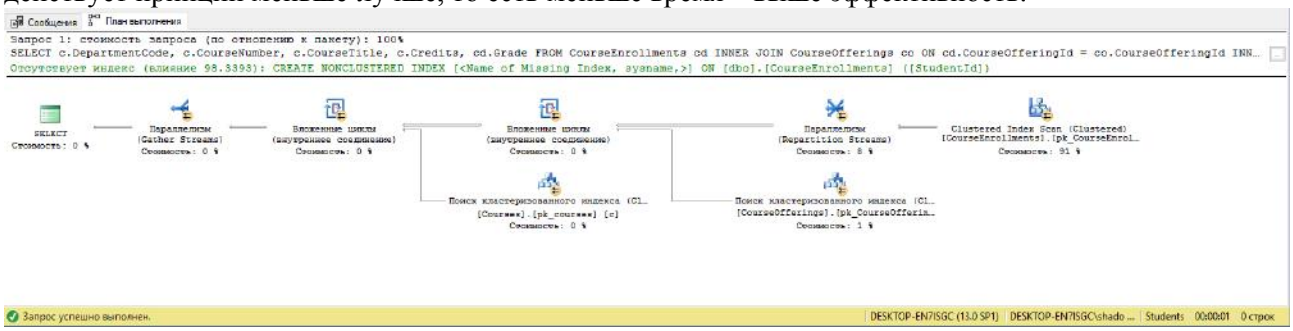

Рис. 2. План выполнения запроса

4. Хранение данных внутри MS SQL Server

Данные являются важнейшей частью БД. Для MS SQL Server характерно страничное хранение данных в так называемых «Структурах кластерных индексов» (англ. clustered index structure). Вид данной структуры изображен на рисунке 3. Данная структура, по сути, является сбалансированным деревом, разделяемым на три зоны: корневой узел (корень), промежуточный узел, узел данных (листья). Данные хранятся непосредственно в листьях в отсортированном виде, причем сортировка происходит по кластерному ключу.

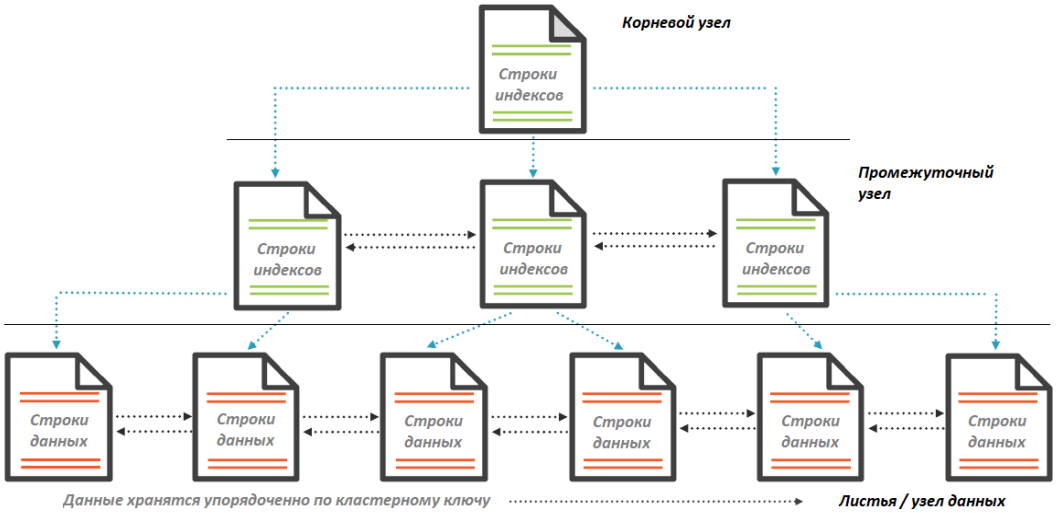

Рис. 3. Структура кластерных индексов

Корень предоставляет доступ ко всем страницам, к примеру, имея 1200 записей в БД, корень предоставлял бы доступ к записям 1-1200. Для облегчения задачи доступа, введен промежуточный узел, для элементов которого характерно, то что они предоставляют доступ только к определенной порции записей. Так, промежуточный узел может быть разделен на элементы с доступом к 1-400, 401-800, 801-1200 записям. Узел данных содержит листья - страницы, в которых непосредственно и хранится информацию. Для данного примера, каждая страница хранила бы записи с номерами 1-200, 201-400, 401-600, 601-800, 801-1000, 1001-1200 соответственно. Размер одной страницы может варьироваться от 4 до 8 килобайт, в зависимости от редакции MS SQL Server.

5. Способы оптимизации. Индексация

Для любой базы данных жизненно важным критерием является скорость обработки и выполнения запросов. В случае, если запрос выполняется медленно, следует прибегнуть к способам оптимизации производительности выполнения запросов. Их существует два:

- Оптимизация запросов,
- Использование индексации.

Оптимизация запроса, как было сказано выше, подразумевает реорганизацию и модификацию запроса, с целью увеличения его эффективности. Простыми словами: запрос нужно переписать так, чтобы он работал быстрее. В данной статье будет рассматриваться только индексация. Индексация – это процесс создания индексов к определенной таблице. Индекс представляет собой средство, помогающее ускорить поиск необходимых данных за счет физического или логического их упорядочивания. В основном, выделяют два типа индексов: кластерный и некластерный. Оба обладают следующими характеристиками:

- Представляет собой набор ссылок, упорядоченных по определенному столбцу таблицы;
- Это набор уникальных значений для таблицы с соответствующими ссылками на данные;
- Расположены и хранятся в самой таблице.

Кластерный индекс характеризуется тем, что при своем определении, он физически перестраивает местоположение данных. Благодаря этому операции, связанные с обращением к индексу и новым поиском нужной строки в таблице, отпадают – все необходимые данные находятся следом за текущей записью. Но данное свойство накладывает следующее ограничение: для одной таблицы может быть создан только один кластерный индекс. Это связано как раз с тем, что данные упорядочиваются физически.

В отличие от кластерных, некластерные индексы не перестраивает физическую структуру таблицы, а только организуют ссылки на соответствующие строки. Поэтому для таблицы может быть создано столько индексов, сколько необходимо пользователю.

6. Пример использования

Для примера использования, создадим запрос на выборку к реальной базе данных и выполним его сначала без применения индексации и с применением индексации. Имеется три таблицы: таблица CourseOfferings на 59 280 записей, таблица CourseEnrollments на 3 724 654 записей и таблица Courses на 180 записей. К данной БД был создан запрос, изображенный на рисунке 4.

```
SELECT c.DepartmentCode, c.CourseNumber, c.CourseTitle, c.Credits, cd.Grade
FROM CourseEnrollments cd
INNER JOIN CourseOfferings co
       ON cd.CourseOfferingId = co.CourseOfferingId
INNER JOIN Courses c
       ON co.DepartmentCode = c.DepartmentCode
       AND co.CourseNumber = c.CourseNumber
WHERE cd.StudentId = 29717;
```

Рис. 4. Вид тестового запроса к БД

Данный запрос возвращает информацию о курсах, на которые записан студент с идентификационным номером 29 717. Результирующие планы изображены на рисунке 5. Как видно по рисунку, наивысшая стоимость для одной операции – 91% - для выполненного запроса до применения индексации. После применения индексации, стоимость имеет чуть более распределенный характер. Так, наибольшая стоимость после индексации – 43%.

Рис.5. Вид планов выполнения до и после индексации

В таблице 1 сравнивается скорость выполнения запроса до и после применения индексации.

Таблица 1. Скорости выполнения запроса

Метрика	Без индексации	С индексацией
Операция доступа к данным	Сканирование кластерного индекса	Поиск по ключу
Стоимость выражения	12,013	0,089
Стоимость операции ввода/вывода	8,874	0,003
Время CPU (мс)	219	16
Затраченное время (мс)	195	63

Из данных таблицы видно, что индексация дает ощутимый прирост в производительности выполнения запроса. Нагрузка на процессор уменьшилась в 13.7 раз, а затраченное время в 3,1 раз.

7. Заключение

В современном мире, скорость обработки информации играет одну из первостепенных ролей. Но независимо от мощности компьютеров, плохо оптимизированные программы могут свести на нет все преимущества высокопроизводительных систем. Любая программа должна быть оптимизирована и эффективно выполнять свою задачу. Похожая ситуация происходит и с базами данных. В мире где огромное количество информации хранится в БД на серверах в интернете, скорость доступа к этим данным критична. Любой запрос следует по максимуму оптимизировать.

Одним из методов оптимизации, описанным в этой статье, является индексация. Индексация дает огромный прирост к производительности, но и здесь есть свои ограничения и замечания. Для часто обновляемых таблиц следует использовать как можно меньше индексов (при изменении записей происходит реорганизация индексов); если таблица содержит множество данных, изменения которых незначительны, то можно использовать столько индексов, сколько необходимо; индексацию не стоит использовать для небольших таблиц; уникальность значений в столбце влияет на производительность индекса; для составного индекса играет роль и порядок столбцов в индексе.

Литература

1. David Berry. *What Every Developer Should Know About SQL Server Performance*. [Электронный ресурс]. - режим доступа: https://www.pluralsight.com/courses/sql-server-performance-every-developer-should-know
2. Microsoft TechNet. *Оптимизация производительности выполнения запросов (SQL Server Compact).* [Электронный ресурс]. - режим доступа: https://technet.microsoft.com/ru-ru/library/ms172984(v=sql.110).aspx
3. Константин WINGUNT. *Оптимизация производительности SQL Server с использованием индексов.* [Электронный ресурс]. - режим доступа: https://habrahabr.ru/post/164717/
4. Антон Спирин. *14 вопросов об индексах в SQL Server, которые вы стеснялись задать.* [Электронный ресурс]. - режим доступа: https://habrahabr.ru/post/247373/
5. Константин Кузнецов. О, эти планы запросов. [Электронный ресурс]. - режим доступа: https://habrahabr.ru/post/98622/
6. ИНТУИТ. Основы SQL. [Электронный ресурс]. - режим доступа: http://www.intuit.ru/studies/courses/5/5/info