Март, № 2 **Технические науки** **2013**

УДК 004657

ВЛАДИМИР БОРИСОВИЧ ЕФЛОВ
кандидат физико-математических наук, доцент кафедры механики строительного факультета, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
veflov@gmail.com

ЕВГЕНИЙ АЛЕКСАНДРОВИЧ ПИТУХИН
доктор технических наук, профессор кафедры прикладной математики и кибернетики математического факультета, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
eugene@petrsu.ru

ФЕДОР ВЛАДИМИРОВИЧ СТАФЕЕВ
аспирант кафедры прикладной математики и кибернетики математического факультета, Петрозаводский государственный университет (Петрозаводск, Российская Федерация)
fedor.stafeev@gmail.com

АНАЛИЗ ВЫПОЛНЕНИЯ ЗАПРОСОВ В СУБД H2*

Проведено исследование работы модулей выполнения запросов реляционных СУБД в зависимости от параметров запросов. Исследовалась реляционная СУБД H2 Database. Был использован свободно распространяемый продукт компании Oracle, реализующий виртуализацию аппаратного окружения. В экспериментальной БД было создано множество таблиц с одинаковой структурой, но с разным числом записей в таблице. В ходе основного эксперимента произведено измерение времени выполнения запросов в зависимости от относительного номера записи в таблицах и значений системной памяти. Для аппроксимации экспериментальных данных была выбрана кривая в виде обратной функции. Было отмечено существенное снижение времени выполнения после обработки первого запроса, а также скачкообразное снижение времени выполнения запросов при увеличении размера системной памяти в случае первого обращения к таблице. Для выяснения влияния объема кэша на производительность СУБД были проведены дополнительные измерения. Для определенных запросов при фиксированном значении системной памяти варьировался размер кэша от нулевого значения до дальнейших значений с шагом 512 Кб. При увеличении размера кэша был отмечен качественный скачок: уменьшение времени запроса до 25 раз. Данный факт свидетельствовал об использовании только системной памяти в ходе выполнения запросов. Учитывая ярко выраженную зависимость производительности H2 Database от технологии кэширования, увеличение объема данных, размещаемых в системной памяти, способно качественно уменьшить время выполнения запросов.

Ключевые слова: реляционная СУБД, выполнение запросов, СУБД H2

ВВЕДЕНИЕ

В условиях длительной эксплуатации информационных систем на основе реляционных СУБД, сопровождающейся накоплением пользовательских данных, актуальна проблема доступа и обработки хранимой информации. Разработке методик уменьшения времени доступа к данным, в частности оптимизации запросов, уделено особое место в публикациях последних десятилетий [2], [6], [7]. Нами проведено исследование работы модулей выполнения запросов реляционных СУБД в зависимости от их параметров. Объектом исследования была выбрана реляционная СУБД H2 Database, созданная и поддерживаемая сообществом разработчиков. СУБД выпускается под модифицированной лицензией MPL 1.1 (Mozilla Public License).

МЕТОДИКА ЭКСПЕРИМЕНТА

Для того чтобы обеспечить возможность варьировать параметры аппаратной платформы, был использован свободно распространяемый проприетарный вариант реализации технологии виртуализации компании Oracle. H2 Database реализована посредством технологии Java. В качестве ОС для гостевой системы и системы хоста использовался релиз FreeBSD 8.2. В экспериментальной БД было создано множество таблиц с одинаковой структурой с идентификаторами вида BOOKS_NNN, где NNN – число записей в таблице [3]. Структура таблиц имеет следующее описание согласно стандарту ANSI SQL [1]:

```
CREATE TABLE BOOKS_NNN (BOOKS_ID INTEGER DEFAULT NOT NULL, TITLE VARCHAR(500) DEFAULT NULL, AUTHOR VARCHAR(300) DEFAULT NULL, PRODUCERS VARCHAR(300) DEFAULT NULL, YEAR INTEGER DEFAULT NULL, ISBN BIGINT DEFAULT NULL, NUM_PAGES INTEGER DEFAULT NULL, PRICE DOUBLE DEFAULT NULL, PUBL_ID INTEGER DEFAULT NULL, SHOP_ID INTEGER DEFAULT NULL, PRIMARY KEY (BOOKS_ID));
```

В первой серии экспериментов использовались запросы типа SELECT по шаблону:

SELECT * FROM **tablename**
WHERE **field_name** = **value**,

где **tablename** – идентификатор таблиц вида BOOKS_NNN, **field_name** – идентификатор поля, **value** – варьируемое значение поля согласно его типу.

В качестве типов полей использовались: первичный ключ, целочисленный тип INT; целочисленный, тип INT; вещественные числа, тип DOUBLE; символьный, тип VARCHAR. Кроме того, использовались запросы с предикатом типа LIKE для символьного типа.

В таблицах BOOKS_NNN были созданы специальные записи с полями всех означенных типов со значениями первичного ключа из множества {0,05L; 0,1L…0,95L; L}, где L – число записей в таблице.

Всего в ходе эксперимента были проведены измерения для 10 значений системной памяти, 20 значений первичного ключа таблицы, 5 типов предикатов, по 10 измерений на каждую комбинацию параметров – всего 10 000 значений для каждой из таблиц. В эксперименте использовался персональный компьютер с архитектурой процессора x86 (AMD Sempron 2200+, 1500 MHz) и системной памятью в 512 Мб.

ПРОИЗВОДИТЕЛЬНОСТЬ

В ходе основного эксперимента произведено измерение времени выполнения запросов в зависимости от относительного номера записи в таблицах и значений системной памяти. Диапазон значений системной памяти виртуальной машины подбирался в ходе предварительных измерений и варьировался в диапазоне 112–240 Мб. В ходе основного эксперимента не изменялись параметры СУБД, выставленные по умолчанию разработчиками. В результате наблюдений был установлен ряд фактов.

Экспериментальная зависимость времени выполнения запросов типа SELECT с предикатом по полю первичного ключа от номера запроса имеет выраженный постоянный характер, хорошо описываемый средним значением.

Выполнение запросов проводилось пакетно при помощи предлагаемой разработчиками утилиты и разработанного для эксперимента скрипта. Пакет содержал полный список запросов. В данном случае, как следует из дальнейшего анализа, место конкретного запроса в общем списке имеет немаловажное значение.

В начале списка располагались запросы с предикатом по полю типа VARCHAR. По всем таблицам было отмечено существенное снижение времени выполнения после обработки первого запроса (начало графика на рис. 1). Результатом выполнений этих запросов является запись с минимальным номером кортежа.

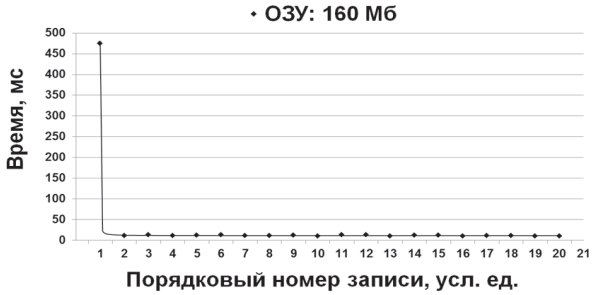

Рис. 1. Выполнение запросов вида «Select * FROM BOOKS_100000 WHERE BOOKS_10000.ISBN = …»

Объяснением этих наблюдений может служить следующее. H2, как и большинство современных СУБД [4], хранит часто используемые данные в системной памяти, обращение к которым занимает существенно меньшее время.

Также было отмечено скачкообразное снижение времени выполнения запросов при увеличении размера системной памяти в случае первого обращения к таблице данных (рис. 2), когда влияние механизма кэширования исключается.

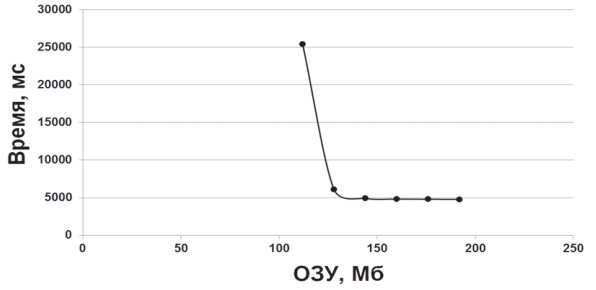

Рис. 2. Время выполнения запросов при первом обращении к таблице

Для аппроксимации экспериментальных данных была выбрана форма кривой (рис. 1–2) в виде обратной функции

$$T(s) = \frac{\gamma_1}{s - \gamma_2} + \gamma_3,$$

где $T(s)$ – время выполнения запроса, γ_1 – коэффициент пропорциональности, γ_2, γ_3 – вертикальная и горизонтальная оси гиперболы соответственно.

ИСПОЛЬЗОВАНИЕ КЭША

Для выяснения влияния объема кэша на производительность СУБД проведены дополнительные измерения. При фиксированном значении системной памяти и запросов к определенной таблице варьировался размер кэша, задаваемого в настройках подключения к СУБД. На рис. 3 представлены графики экспериментальных зависимостей. Производился скачкообразный рост кэша от нулевого размера до дальнейших значений с шагом 512 Кб. Из списка пакета запросов первый запрос выполнялся медленнее последующих. Качественный скачок – уменьшение времени запроса до 25 раз – отмечался при значении кэша в 28 160 Кб (рис. 4).

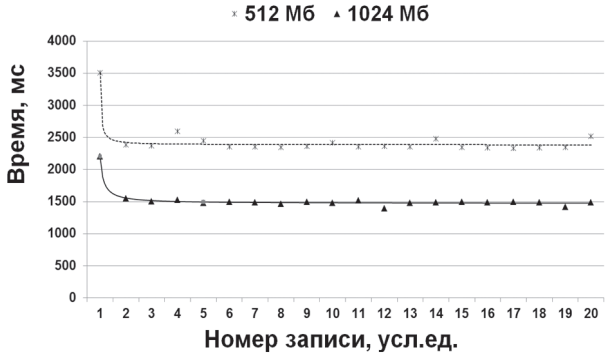

Рис. 3. Зависимость времени выполнения запросов при различных значениях размера кэшируемых данных

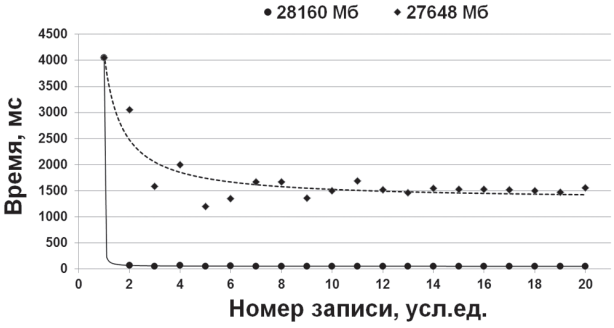

Рис. 4. Изменение зависимости времени выполнения запросов при увеличении размера кэшируемых данных

Среднее время выполнения запросов, следовавших в списке скрипта после первичного обращения к таблице, составляет, по оценке, всего около 1,5 % от среднего времени выполнения первичного запроса. Зависимость установившегося времени выполнения запросов от объема кэшируемых данных представлена на рис. 5.

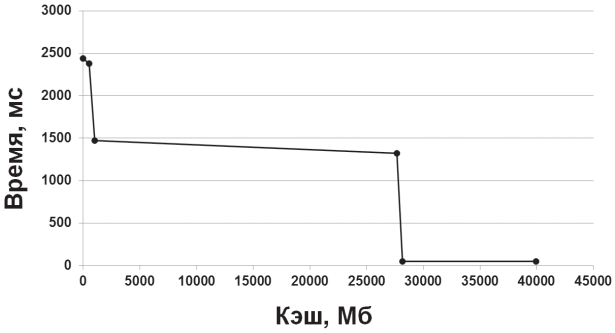

Рис. 5. Выполнение запросов при изменении размера кэша

Очевидно, СУБД не использовало вторичную память в этом случае. Все необходимые данные кэшировались в системной памяти. Необходимо отметить, что размер экспортированных в формат CSV файла данных запрашиваемой таблицы составил 5054 Кб.

ВЫВОДЫ

Учитывая ярко выраженную зависимость производительности H2 Database от технологии кэширования, увеличение объема данных, размещаемых в системной памяти, способно качественно уменьшить время выполнения запросов. Неслучайно в этой связи на протяжении последних двух десятилетий [5] активно происходит развитие направления в разработке СУБД – In-memory database, реляционная база данных, оптимизированная для работы с оперативной памятью и целиком размещаемая в ней.

* Работа выполнена при поддержке Программы стратегического развития ПетрГУ в рамках реализации комплекса мероприятий по развитию научно-исследовательской деятельности на 2012–2016 гг.

СПИСОК ЛИТЕРАТУРЫ

1. Грабер М. SQL. Справочное руководство. СПб.: Лори, 2006. 368 с.
2. Кузнецов С. Д. Методы оптимизации выполнения запросов в реляционных СУБД // Итоги науки и техники. Вычислительные науки. Т. 1. М.: ВИНИТИ, 1989. С. 76–153.
3. Стафеев Ф. В. Способы оценки эффективности оптимизаторов запросов в реляционных СУБД // Роль науки в устойчивом развитии общества: Сборник материалов 2-й междунар. науч.-практ. конф., 24–25 декабря 2010 г., г. Тамбов. Тамбов: Тамбовпринт, 2010. С. 150–151.
4. Ailamaki A., DeWitt D. J., Hill M. D., Wood D. A. and etc. DBMSs on a Modern Processor: Where Does Time Go?// VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases. Edinburgh, 1999. P. 266–277.
5. Deshmukh P. A. Review on Main Memory Database // International Journal of Computer & Communication Technology (IJCCT). 2011. Vol. 2 . №. 7. P. 54–58.
6. Ioannidis Y. E. Query optimization // ACM Computing Surveys (CSUR) Surveys Homepage archive. 1996. Vol. 28. Issue 1. P. 121–123.
7. Ouzzani M. Query Processing and Optimization on the Web // Distributed and Parallel Databases archive. 2004. Vol. 15. Issue 3. P. 187–218.

Yeflov V. B., Petrozavodsk State University (Petrozavodsk, Russian Federation)
Pitukhin E. A., Petrozavodsk State University (Petrozavodsk, Russian Federation)
Stafeev F. V., Petrozavodsk State University (Petrozavodsk, Russian Federation)

REVIEW OF QUERY EXECUTION IN DBMS H2

Research of query execution engines was carried out depending on the changes of query parameters. The relational DBMS H2 Database was investigated. Oracle Corporation's freeware was used to realize virtualization of hardware environment. We have cre-

ated experimental database consisting of many tables that have the same structure and different number of rows. In the course of the main experiment measurements of the query execution time depending on the relative row number in tables and values of the main memory were made. We decided on the use of the inverse function for approximation of measurement data. A significant decrease of the query execution time after first query execution was observed. A step like decrease of the query execution time under the increase of the main memory size was also noted. To determine the influence of cache value on the DBMS performance we have conducted an additional experiment. For certain queries with fixed values of main memories the cache memory was varied from null to further values with 512 Kb. A sharp decrease in the query execution time (up to 25 times) was registered in the case of cache memory increase. This observation testifies to the fact that the only memory used during query execution was the main memory. Considering strong marked dependence of the H2 Database performance from cache technology, the raise of the value of data in the main memory it is possible to reduce the query execution time qualitatively.

Key words: RDBMS, query execution, H2 Database

REFERENCES

1. G r a b e r M. *SQL. Spravochnoe rukovodstvo* [SQL. Reference Guide]. St. Petersburg, Lori Publ., 2006. 368 p.
2. K u z n e t s o v S. D. Methods of query execution optimization in relational DBMS [Metody optimizatsii vypolneniya zaprosov v relyatsionnykh SUBD]. *Itogi nauki i tekhniki. Vychislitel'nye nauki* [Results of science and techniques. Computational science]. Moscow, VINITI Publ., 1989. Vol. 1. P. 76–153.
3. S t a f e e v F. V. Estimation method of query optimizer performance in Relational DBMS [Sposoby otsenki effektivnosti optimizatorov zaprosov v relyatsionnykh SUBD]. *Sbornik materialov 2-y mezhdunarodnoy nauchno-prakticheskoy konferentsii* "*Rol' nauki v ustoychivom razvitii obshchestva"* [Proc. 2nd Int. Research and Practice "The Role of Science in Society Sustainable Development"]. Tambov, Tambovprint Publ., 2010. P. 150–151.
4. A i l a m a k i A., DeWitt D. J., Hill M. D., Wood D. A. and etc. DBMSs on a Modern Processor: Where Does Time Go?// VLDB'99, Proceedings of 25th International Conference on Very Large Data Bases. Edinburgh, 1999. P. 266–277.
5. D e s h m u k h P. A. Review on Main Memory Database // International Journal of Computer & Communication Technology (IJCCT). 2011. Vol. 2 . №. 7. P. 54–58.
6. I o a n n i d i s Y. E. Query optimization // ACM Computing Surveys (CSUR) Surveys Homepage archive. 1996. Vol. 28. Issue 1. P. 121–123.
7. O u z z a n i M. Query Processing and Optimization on the Web // Distributed and Parallel Databases archive. 2004. Vol. 15. Issue 3. P. 187–218.

Поступила в редакцию 04.12.2012