УДК 004.043
DOI 10.21685/2072-3059-2018-1-3

*И. В. Бельченко, Р.А. Дьяченко*

# МЕТОДИКА ПОВЫШЕНИЯ ПРОИЗВОДИТЕЛЬНОСТИ ИНФОРМАЦИОННОЙ СИСТЕМЫ ЗА СЧЕТ ОПТИМАЛЬНОЙ РЕСТРУКТУРИЗАЦИИ ДАННЫХ

**Аннотация**.

*Актуальность и цели.* Производительность информационной системы – это один из главных показателей ее эффективности. Большинство существующих информационных систем используют реляционные базы данных для хранения информации. Проектирование базы данных для информационной системы заключается в последовательной разработке концептуальной, логической и физической модели. Эта последовательность проектирования доказала свою эффективность в случае, когда требования к базе данных, продиктованные предметной областью, строго формализованы, нагрузка на сервер баз данных прогнозируема. Но существуют случаи, когда информационных систем, работающих с базой данных, несколько. Каждая из них имеет свои требования, иногда противоречивые, к информации, которая должна быть предоставлена базой данных за минимально возможное время. В таких случаях задачи прогнозирования нагрузки, выделения наиболее ресурсоемких запросов требуют применения методов системного анализа. Целью работы является теоретическое обоснование методики повышения производительности информационной системы за счет оптимальной реструктуризации данных.

*Материалы и методы.* Получение оптимального разбиения исследуемой табличной структуры на дочерние начинается с системного анализа способов хранения данных системой управления базой данных (СУБД). Для формализации предметной области выделены параметры и множества, влияющие на скорость обработки запросов на чтение информации к исследуемой таблице базы данных. Рассмотрен существующий подход к хранению строк данных таблиц в виде блоков на дисковом накопителе. Сформулирована задача оптимизации количества блоков данных, необходимых для обработки группы запросов на чтение информации. Предложена целевая функция, введены структурные ограничения. Предложен метод поиска субоптимального решения. Описан алгоритм методики.

*Выводы*. Предложен подход к нахождению субоптимального разбиения исследуемой табличной структуры на дочерние. Предложенная методика особенно актуальна для таблиц баз данных, содержащих большой набор строк. Полученные результаты могут быть использованы при проектировании отечественных СУБД. Дальнейшие исследования в этой области связаны с разработкой методик поиска оптимальных разбиений табличных структур баз данных без необходимости введения ограничений на максимально возможное число дочерних таблиц.

**Ключевые слова**: система поддержки принятия решений, оптимизация, структуры данных, базы данных, системный анализ.

*I. V. Bel'chenko, R. A. D'yachenko*

# METHODOLOGY OF IMPROVING THE PRODUCTIVITY OF THE INFORMATION SYSTEM THROUGH THE OPTIMUM DATA RESTRUCTURING

**Abstract**.

*Backgrounds*. The performance of the information system is one of the main indicators of its effectiveness. Most existing information systems and software complexes use relational databases to store information. Designing a database (DB) for an information system consists in the sequential development of a conceptual, logical and physical model. This design sequence has proven effective when the database requirements dictated by the subject area are strictly formalized, the load on the database server is predictable. But there are cases when the information systems working with the database are several. Each of them has its own information requirements, sometimes contradictory, which must be provided by the database in the shortest possible time. In such cases, the task of forecasting the load, allocating the most resource-intensive queries requires the use of methods of system analysis. The purpose of the work is the theoretical justification of the methodology for increasing the productivity of the information system through optimal data restructuring.

*Materials and methods*. Getting the optimal partition of the studied table structure into children begins with a system analysis of the methods of data storage by the database management system (DBMS). To formalize the domain, the parameters and sets that affect the processing speed of requests for reading information to the database table being researched are highlighted. An existing approach to storing rows of these tables as blocks on a disk drive is considered. The problem of optimization of the number of data blocks necessary for processing a group of requests for reading information is formulated. The objective function is proposed, structural limitations are introduced. A method for finding a suboptimal solution is proposed. The algorithm of the technique is described.

*Conclusions*. An approach is proposed for finding a suboptimal partition of the investigated table structure into its children. The proposed technique is especially relevant for database tables containing a large set of strings. The obtained results can be used in the design of domestic DBMS. Further research in this area is related to the development of techniques for searching for optimal partitioning of database tables without the need to introduce restrictions on the maximum possible number of child tables.

**Key words**: decision support system, optimization, data structures, databases, system analysis.

## Введение

Большинство многопользовательских информационных web-систем выделяются такими требованиями, как оперативное взаимодействие с пользователем [1]. Эффективное исполнение данного требования зависит не только от аппаратной составляющей, включающей в себя серверное оборудование и линии связи, но и от реализации программных компонентов, среди которых программное приложение, реализованное с применением web-технологий, и система управления базой данных (СУБД). В статье рассмотрена методика повышения производительности информационной системы за счет увеличения скорости выполнения запросов на чтение информации базы данных. От структуры данных, способах ее физического размещения на жестких дисках зависит количество обращений к дисковым накопителям, которые сопровождаются соответствующими прерываниями и задержками по времени [2].

Важным понятием при рассмотрении вопроса физической организации баз данных является понятие блока. Блок – это минимальный адресуемый элемент внешней памяти, с помощью которого осуществляется обмен ин-

формацией между оперативной и внешней памятью. Запись и чтение блоков осуществляется через буферную часть оперативной памяти. Для организации каждого файла базы данных, в зависимости от его размера во внешней памяти, выделяется от одного до *N* блоков, где размещаются записи. В одном блоке могут разместиться все записи или в нескольких блоках – одна запись, либо в одном блоке – одна запись; от этого будет зависеть время считывания и записи элементов файла. Записи в блоках размещаются плотно, без промежутков, последовательно одна за другой. В блоке часть памяти отводится под служебную информацию: относительный адрес свободных участков памяти, указатели на следующий блок и т.д. Для хранения поступающих данных, которые должны размещаться в одном блоке, заполненном уже полностью, выделяется дополнительный блок памяти в области переполнения записи, организованной в виде одного блока, где записи связываются указателями в одну цепь.

Таким образом, на скорость поиска влияют: объем блока в байтах, объем файла, количество записей в блоке файла, количество записей в блоке индекса, количество блоков в файле, доля резервной части блока, число полей в записи, размер записи в байтах [2].

## 1. Постановка задачи вертикальной реструктуризации табличных структур данных

Процесс построения оптимальной модели данных информационной системы включает оптимальное распределение таблиц базы данных по блокам на дисковом накопителе. Основным критерием оптимизации модели данных информационной системы является минимальный размер строки таблицы реляционной базы данных, позволяющий в одном блоке хранить больше данных и, как следствие, минимизировать количество операций чтения блоков данных с жесткого диска при выполнении запросов к базе данных. Это достигается за счет уменьшения объема данных, побочно участвующих в запросе [3]. Традиционная схема процесса считывания блоков данных для выполнения запроса на считывание данных представлена на рис. 1.

В рамках методики предлагается разделить таблицы базы данных на несколько сущностей, связанных отношением один к одному. В соответствии с принципами блочного хранения данных в СУБД каждая таблица будет храниться в отдельном наборе блоков. При выполнении запроса на чтение информации СУБД считывает блоки данных с жесткого диска в оперативную память каждой таблицы, атрибуты которой участвуют в запросе. Схема процесса считывания блока данных для выполнения запроса на считывание данных с использованием методики разделения таблиц представлена на рис. 2.

Задача повышения производительности информационной системы сводится к поиску оптимального разделения табличных структур базы данных с учетом конкретной группы запросов на чтение информации, выявленной статистически в рамках жизненного цикла БД.

## 2. Оптимизация табличных структур данных информационной системы

Для формализации задачи рассмотрим множества и параметры, влияющие на скорость обработки запросов на чтение информации к исследуемой таблице базы данных.

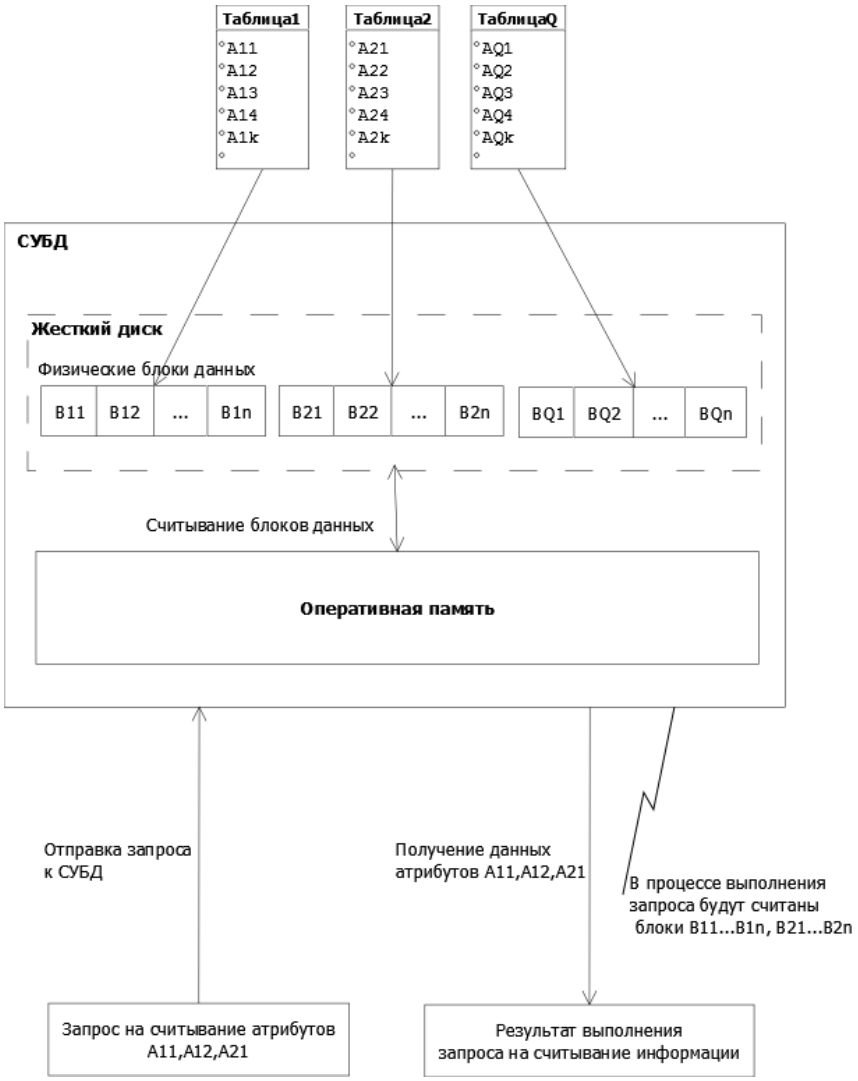


Рис. 1. Традиционная схема процесса считывания блоков данных для выполнения запроса на считывание данных

1. Целочисленный параметр $TS$, равный количеству атрибутов в исследуемой таблице.

2. Вектор типов данных $DBT = \{dbt_{idbt} \,|\, idbt = \overline{1, ndbt}\}$, которые поддерживаются конкретной выбранной СУБД. Элемент вектора – занимаемый элементом типа размер данных в байтах памяти.

3. Набор атрибутов (столбцов таблицы) $TA$, заданный бинарной матрицей, элемент которой $ta_{ita,jta}$ равен единице, если столбец $ita$ таблицы имеет тип $jta$, $ita = 1,\ldots,TS, jta = 1,\ldots,ndbt$.

4. Множество, представляющее группу запросов $Q = \{q_{iq} \,|\, iq = \overline{1, nq}\}$ на чтение информации из таблицы базы данных, элемент множества – кортеж из двух элементов $q_{iq} = \{SFQ_{iq}, QA_{iq}\}$, где $SFQ_{iq}$ числовой параметр, равный

частоте появления запроса за выбранный период времени, $QA_{iq} = \{qa_{iqa} \,|\, iqa = \overline{1, TS}\}$ – бинарный вектор, размерность которого равна количеству атрибутов таблицы *TS*; $qa_{iqa} = 1$, если атрибут таблицы *TA* участвует в запросе, и 0 в противном случае; *nq* – количество запросов в статистической выборке, выявленной в рамках жизненного цикла БД.

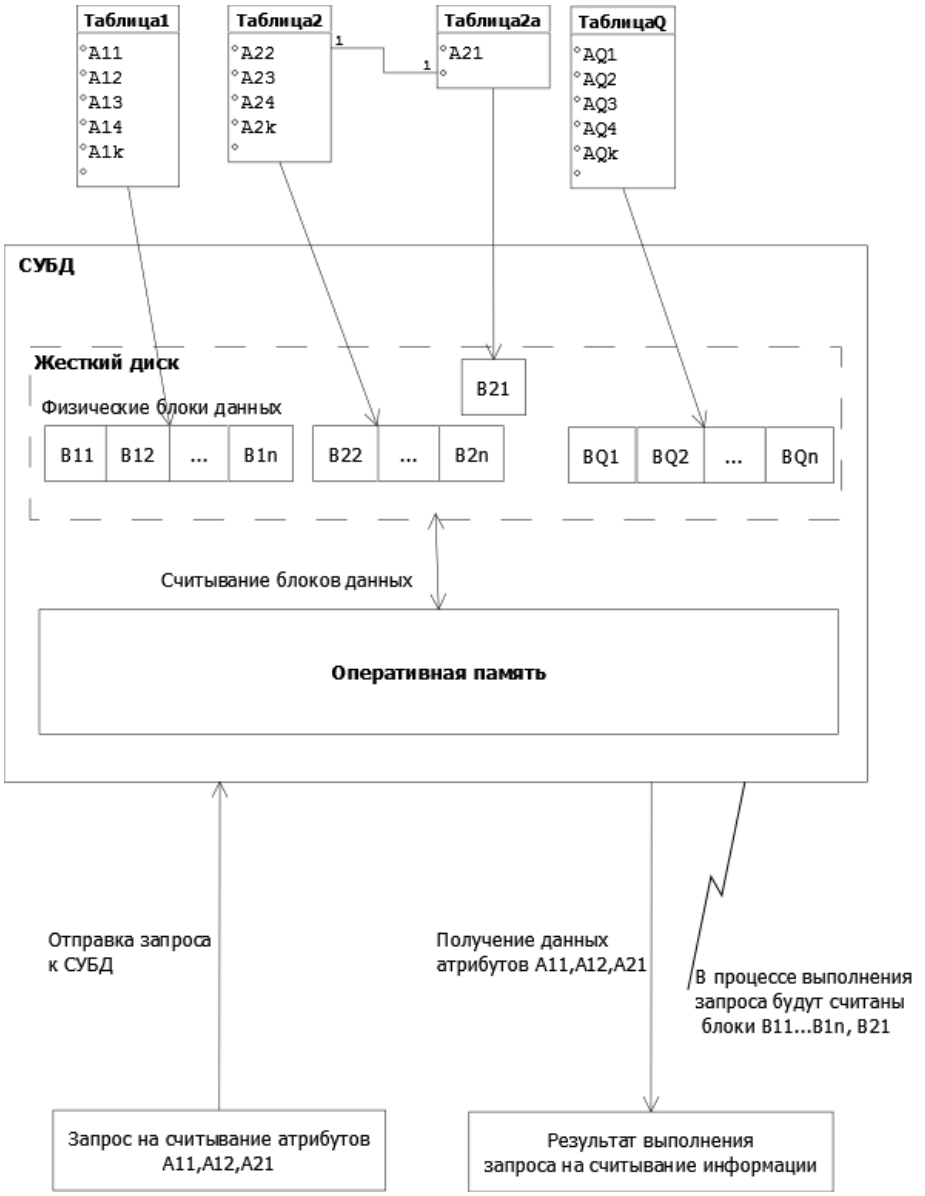


Рис. 2. Схема процесса считывания блока данных для выполнения запроса на считывание данных с использованием методики разделения таблиц

5. Множество индексов, характеризующихся набором полей таблицы, по которым построен индекс $IN = \{in_{iin} \,|\, iin = \overline{1, nin}\}$. Элемент множества

$in_{iin} = \{in_{iin,jin} \,|\, jin = \overline{1,TS}\}$ – бинарный вектор, размерность которого равна количеству атрибутов таблицы $TS$; $in_{iin,jin} = 1$, если атрибут $jin$ таблицы $TA$ участвует в индексе $in_{iin}$, и 0 в противном случае.

6. Хранимые процедуры и функции $PF = \{pf_{ipf} \,|\, ipf = \overline{1,npf}\}$, характеризующиеся набором полей, используемых в теле хранимой процедуры или функции. Элемент множества $pf_{ipf} = \{pf_{ipf,jpf} \,|\, jpf = \overline{1,TS}\}$ – бинарный вектор, размерность которого равна количеству атрибутов таблицы $TS$, $pf_{ipf,jpf} = 1$, если атрибут $jpf$ таблицы $TA$ участвует в теле хранимой процедуры или функции $pf_{ipf}$, и 0 в противном случае.

7. Множество триггеров базы данных $TG = \{tg_{itg} \,|\, itg = \overline{1,ntg}\}$, характеризующихся набором полей таблицы, используемых в теле триггера. Элемент множества $tg_{itg} = \{tg_{itg,jtg} \,|\, jtg = \overline{1,TS}\}$ – бинарный вектор, размерность которого равна количеству атрибутов таблицы $TS$; $tg_{itg,jtg} = 1$, если атрибут $jtg$ таблицы $TA$ участвует в теле триггера $tg_{itg}$, и 0 в противном случае.

### 3. Анализ влияния количества физических блоков данных, используемых таблицей базы данных, на общее время выполнения группы запросов к ней

Множество запросов $Q$ к рассматриваемой таблице обрабатывается СУБД за время $T(Q,TA,DBT)$. Временные затраты $T(Q,TA,DBT)$ можно представить в виде суммы временных затрат на чтение блоков данных таблиц $T_h(Q,TA,DBT)$, участвующих в запросах $Q$, и остальных временных затрат $T_o(Q,TA,DBT)$, к которым относятся временные затраты на выполнение плана обработки запроса, на передачу информации и т.д.:

$$T(Q,TA,DBT) = T_h(Q,TA,DBT) + T_o(Q,TA,DBT).$$

В рамках методики предлагается уменьшить слагаемое, влияющее на общее время выполнения запроса $T_h(Q,TA,DBT)$. Временные затраты $T_h(Q,TA,DBT)$ в общем виде зависят от количества операций чтения блоков данных таблиц с жесткого диска. Пусть временная задержка, связанная с считыванием одного блока данных, равна $T_b$, тогда:

$$T_h(Q) = \left( \sum_{iq}^{nq} B\left(q_{iq}, TA, DBT\right) \right) \cdot T_b,$$

где $B\left(q_{iq}, TA, DBT\right)$ – число информационных блоков, которые необходимо считать с жесткого диска в кэш СУБД для дальнейшего выполнения запроса $q_{iq}$ к таблице, заданной бинарной матрицей $TA$. Кэш СУБД находится

в оперативной памяти вычислительного устройства; $T_b$ – временная задержка, связанная со считыванием одного блока данных.

Функция $B\left(q_{iq},TA,DBT\right)$ вычисляется как отношение:

$$B\left(q_{iq},TA,DBT\right)=\frac{RC\cdot RS(q_{iq},TA,DBT)}{DB},$$

где $RC$ – количество строк в рассматриваемой таблице;

$$RS(q_{iq},TA,DBT)=RSS(q_{iq},TA,DBT)+RST(q_{iq},TA,DBT),$$

$RS(q_{iq},TA,DBT)$ – величина, характеризующая дисковое пространство, занимаемое одной строкой таблицы в байтах; $RSS(q_{iq},TA,DBT)$ – количество памяти, занимаемое служебными отметками СУБД для строки, считываемое при выполнении запроса $q_{iq}$; $RST(q_{iq},TA,DBT)$ – количество памяти, занимаемое атрибутами таблицы в строке, считываемое при выполнении запроса $q_{iq}$; $DB$ – фиксированный размер блока данных выбранной СУБД. В большинстве СУБД он равен 8 Кб.

Параметры $RC$ и $DB$ остаются неизменными.

Так как временную задержку $T_b$, связанную со считыванием одного блока данных, допускается считать постоянной величиной, на сумму временных затрат на чтение блоков данных таблицы $TA$ – $T_h(Q,TA,DBT)$ влияет количество блоков, необходимое для считывания, которое вычисляется как функция $F\left(Q,TA,DBT\right)$:

$$F\left(Q,TA,DBT\right)=\sum_{iq}^{nq}B\left(q_{iq},TA,DBT\right).$$

Подставим в $F\left(Q,TA,DBT\right)$, формулу функции $B\left(q_{iq},TA,DBT\right)$. Функция, определяющая количество блоков, необходимых для считывания с жесткого диска в оперативную память при выполнении множества запросов $Q$ к рассматриваемой таблице *TA*, имеет вид

$$F\left(Q,TA,DBT\right)=\sum_{iq}^{nq}\left(\frac{RC*RS(q_{iq},TA,DBT)}{DB}\right).$$

### 4. Методика уменьшения количества физических блоков данных, используемых СУБД для выполнения группы запросов к таблице, за счет ее оптимального разделения на дочерние таблицы

В рамках методики предлагается разделить рассматриваемую таблицу на $NB\in[1;TS]$ дочерних таблиц, связанных с родительской отношением один к одному, 1:1.

Введем следующую переменную:

$$x_{ij} = \begin{cases} 1, & \text{если } j\text{-атрибут необходимо выделить в } i\text{-ю таблицу,} \\ 0 & \text{в противном случае.} \end{cases}$$

Переменная представляют собой бинарную матрицу для таблицы реляционной базы данных размерностью $TS \times TS$, где $TS$ – количество атрибутов таблицы. Строки матрицы соответствуют таблицам, на которые разбивается родительская таблица, а столбцы соответствуют их атрибутам. Графический пример переменной, соответствующей разбиению таблицы на три дочерних, представлен на рис. 3.

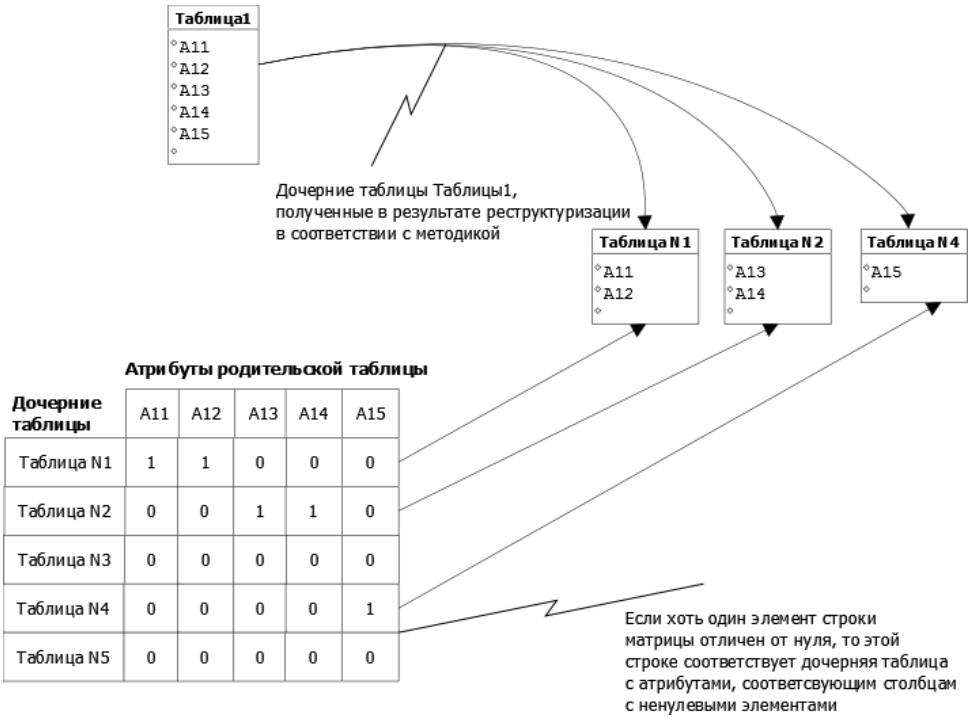


Рис. 3. Графический пример переменной, соответствующей разбиению таблицы на три дочерних

Количество блоков $BM(Q, RC, DB, DBT, TA, X)$, которое необходимо считать с жесткого диска для выполнения множества запросов $Q$ к таблице $TA$, вычисляется как функция, равная сумме блоков, которые необходимо считать с жесткого диска для выполнения множества запросов $Q$ к каждой из дочерних таблиц. Максимальное количество дочерних таблиц равно числу атрибутов родительской таблицы $TA$ и равно $NB$:

$$BM(Q, RC, DB, DBT, TA, X) =$$

$$= \sum_{iq=1}^{nq} \left\lceil \frac{RC \cdot RSM(DBT, TA, X_1) \cdot FQ(q_{iq}, X_1)}{DB} \right\rceil + \ldots$$

$$\ldots + \sum_{iq=1}^{nq} \left\lceil \frac{RC \cdot RSM\left(DBT, TA, X_{irs}\right) \cdot FQ(q_{iq}, X_{irs})}{DB} \right\rceil + \ldots$$

$$\ldots + \sum_{iq=1}^{nq} \left\lceil \frac{RC \cdot RSM\left(DBT, TA, X_{nb}\right) \cdot FQ(q_{iq}, X_{nb})}{DB} \right\rceil,$$

где

$$RSM\left(DBT, TA, X_{irs}\right) = \left( \sum_{j}^{TS} \left[ x_{irs,j} \cdot \left( \sum_{idbt}^{ndbt} DBT_{idbt} \cdot TA_j \right) \right] + RDS\left(DBT, TA, X_{irs}\right) \right),$$

$$F\left(q_{iq}, X_{irs}\right) = \begin{cases} 1, \text{ если } \sum_{u}^{TS} (X_{irs})_u \cdot (q_{iq}(QA))_u > 0, \\ 0 \text{ в противном случае,} \end{cases}$$

$$irs = 1, \ldots, nb, \;\; j = 1, \ldots, TS, \;\; idbt = 1, \ldots, ndbt.$$

Параметры $RC$ и $DB$ являются постоянными; $RSM$ – функция, характеризующая количество байт информации, занимаемое одной строкой дочерней таблицы $irs$; $RDS\left(DBT, TA, X_{irs}\right)$ – функция, характеризующая дисковое пространство, занимаемое служебными отметками СУБД в строке дочерней таблицы $irs$ в байтах.

Следовательно, задача повышения производительности системы сводится к поиску такого разделения таблицы на дочерние, при котором сумма блоков, которые необходимо считать в КЭШ СУБД для выполнения множества запросов $Q$, минимальна.

Целевая функция:

$$\min_{x_{irs,j}} \sum_{iq,irs} \left\lceil \frac{\left( \left( \sum_{j}^{TS} \left[ x_{irs,j} \left( \sum_{idbt}^{ndbt} dbt_{idbt}\, ta_j \right) \right] + RDS\left(DBT, TA, x_{irs}\right) \right) RC\, FQ\left(q_{iq}, x_{irs}\right) \right\rceil}{DB},$$

где

$$F\left(q_{iq}, x_{irs}\right) = \begin{cases} 1, \text{ если } \sum_{u}^{TS} (x_{irs})_u \cdot (q_{iq}(QA))_u > 0, \\ 0 \text{ в противном случае,} \end{cases}$$

$$irs = 1, \ldots, nb, \;\; j = 1, \ldots, TS, \;\; idbt = 1, \ldots, ndbt.$$

При структурных ограничениях:

1. Каждый атрибут родительской таблицы может присутствовать только в одной дочерней таблице:

$$\sum_{m_1} x_{m_1,i_1} = 1,\ m_1 = 1,\ldots,TS, i_1 = 1,\ldots,TS\,.$$

2. Атрибуты таблицы, используемые при построении индексов, должны принадлежать хотя бы одной дочерней таблице:

$$\forall ix,\ ix = 1,\ldots,|IN| : \prod_{m_2}^{TS}\left[\sum_{i_2}^{TS}\left(x_{m_2,i_2}\cdot in_{ix,i_2} - in_{ix,i_2}\right)\right] = 0,$$

$$m_2 = 1,\ldots,TS, i_2 = 1,\ldots,TS.$$

3. Атрибуты таблицы, используемые в теле хранимых процедур или функций, должны принадлежать хотя бы одной дочерней таблице:

$$\forall px,\ px = 1,\ldots,|PF| : \prod_{m_3}^{TS}\left[\sum_{i_3}^{TS}\left(x_{m_3,i_3}\cdot pf_{px,i_3} - pf_{px,i_3}\right)\right] = 0,$$

$$m_3 = 1,\ldots,TS, i_3 = 1,\ldots,TS.$$

4. Атрибуты таблицы, используемые в работе триггеров исследуемой таблицы, должны принадлежать хотя бы одной дочерней таблице:

$$\forall tx,\ tx = 1,\ldots,|TG| : \prod_{m_4}^{TS}\left[\sum_{i_4}^{TS}\left(x_{m_4,i_4} * tg_{tx,i_4} - tg_{tx,i_4}\right)\right] = 0,$$

$$m_4 = 1,\ldots,TS, i_4 = 1,\ldots,TS.$$

5. Отношение количества физических блоков данных, необходимого для хранения данных рассматриваемой таблицы до применения к количеству блоков, необходимого для хранения данных в полученных после применения методики дочерних таблицах, не должно превышать заданного параметра $TSIZE$, $TSIZE \in (0;1]$:

$$TSIZE = \frac{\sum_{iq}^{nq}\frac{RC\cdot RS(q_{iq},TA,DBT)}{DB}}{\left(\left(\sum_{j}^{TS}\left[x_{irs,j}\left(\sum_{idbt}^{ndbt} dbt_{idbt}\cdot ta_j\right)\right] + RDS\left(DBT,TA,x_{irs}\right)\right)\frac{RC\cdot FQ\left(q_{iq},x_{irs}\right)}{DB}\right)}.$$

## 5. Нахождение оптимального разделения таблицы на дочерние для выполнения группы запросов путем полного перебора множества допустимых решений

Целевая функция нелинейна, а также нелинейны ограничения. Переменная $X$ – бинарная матрица размерностью $TS\cdot TS$. Представим переменную в виде машинного слова длиной $TS\cdot TS$. Следовательно, количество возможных комбинаций переменной определяется как $2^{TS\cdot TS}$. Исходя из это-

го, задача обладает экспоненциальной сложностью и является *NP*-трудной. Система из $N$ переменных, каждая из которых может принимать $K$ возможных состояний, может иметь $K^N$ возможных состояний. Анализ такой системы требует обработки как минимум $K^N$ бит информации. Задача становится трансвычислительной, если $K^N > 10^{93}$. Данная задача становится трансвычислительной при количестве атрибутов $TS = 17$, следовательно, отсутствуют способы решения методом полного перебора или грубой силы [4].

В статье представлен способ уменьшения количества возможных комбинаций переменной за счет ограничения максимально возможного числа дочерних таблиц, на которые производится разбиение. Для этого необходимо уменьшить число строк в бинарной матрице $X$. Это приведет к ограничению количества дочерних таблиц, на которые может быть разделена исследуемая таблица. Такой подход позволит найти субоптимальное разделение родительской таблицы $TA$ на дочерние для выполнения группы запросов $Q$ в рамках максимально возможного количества дочерних таблиц, и оптимальное для введенного ограничения на количество дочерних таблиц. Данный подход возможен, если в исследуемой таблице число атрибутов невелико. Это подходит для таблиц-справочников, которые описаны небольшим числом атрибутов.

## 6. Алгоритм методики повышения производительности информационной системы за счет оптимальной реструктуризации данных

Алгоритм применения методики состоит из следующих этапов:

– на первом этапе определяются множества, которые необходимы для дальнейшей оптимизации информационной системы;

– на втором этапе определяются необходимые параметры оптимизации информационной системы;

– на третьем этапе формализуется целевая функция минимизации;

– на четвертом этапе выявляются все структурные ограничения для целевой функции;

– на пятом этапе выполняется поиск субоптимального значения переменной на основе множеств параметров, ограничений и целевой функции оптимизации;

– на шестом этапе получаем оптимальную табличную структуру, повышающую производительность информационной системы.

## Заключение

В результате проведенного исследования была сформулирована проблема повышения производительности информационной системы за счет реструктуризации табличных структур данных. Получено ее описание в теоретико-множественном представлении. Сформулирована целевая функция и ограничения. Предложен подход к нахождению субоптимального разбиения исследуемой табличной структуры на дочерние. Предложенная методика особенно актуальна для таблиц БД, содержащих большой набор строк. В таких табличных структурах обычно хранят данные транзакционные системы – системы обработки транзакций в реальном времени (OLTP) [5, 6].

Полученные результаты могут быть использованы при проектировании отечественных СУБД. Дальнейшие исследования в этой области связаны с разработкой методик поиска оптимальных разбиений табличных структур БД без необходимости введения ограничений на максимально возможное число дочерних таблиц.

### *Библиографический список*


1. **Богданова, А. В.** Повышение качества образовательного процесса за счет внедрения системы «Электронное расписание» в учебной организации / А. В. Богданова, Р. А. Дьяченко, И. В. Бельченко // Политематический сетевой электронный научный журнал Кубанского государственного аграрного университета. – 2016. – № 117. – С. 873–885.
2. **Эмблер, С. В.** Рефакторинг баз данных: эволюционное проектирование : пер. с англ. / Скотт В. Эмблер, Прамодкумар Дж. Садаладж. – М. : Вильямс, 2007. – 672 с.
3. **Тоу, Д.** Настройка SQL. Для профессионалов / Д. Тоу. – СПб. : Питер, 2004. – 333 с.
4. **Чигаркина, Е. И.** Базы данных : учеб. пособие / Е. И. Чигарина. – Самара : Изд-во СГАУ, 2015. – 208 с.
5. **Atroshchenko, V. A.** Development and research of statistical methods and optimization algorithms of search for solutions in intelligence automated systems / V. A. Atroshchenko, V. Ye. Belchenko, I. V. Belchenko, R. A. Dyachenko // International journal of pharmacy and technology. – 2016. – Vol. 8, № 2. – P. 14137–14149.
6. **George, J.** Klir. Facets of Systems Science / J. George. – N.Y., Springer, 1991. – 664 c.


### *References*


1. Bogdanova A. V., D'yachenko R. A., Bel'chenko I. V. *Politematicheskiy setevoy elektronnyy nauchnyy zhurnal Kubanskogo gosudarstvennogo agrarnogo universiteta* [Polytechnical network electronic scientific journal of Kuban State Agrarian University]. 2016, no. 117, pp. 873–885.
2. Embler S. V., Pramodkumar Dzh. Sadaladzh *Refaktoring baz dannykh: evolyutsionnoe proektirovanie: per. s angl.* [Database refactoring: evolutionary design: translated from English]. Moscow: Vil'yams, 2007, 672 p.
3. Tou D. *Nastroyka SQL. Dlya professionalov* [SQL configuration. For professionals]. Saint-Petersburg: Piter, 2004, 333 p.
4. Chigarkina E. I. *Bazy dannykh: ucheb. posobie* [Databases: scientific manuals]. Samara: Izd-vo SGAU, 2015, 208 p.
5. Atroshchenko V. A., Belchenko V. Ye., Belchenko I. V., Dyachenko R. A. *International journal of pharmacy and technology*. 2016, vol. 8, no. 2, pp. 14137–14149.
6. George J. Klir. *Facets of Systems Science*. New York, Springer, 1991, 664 p.



***Бельченко Илья Владимирович***
аспирант, Кубанский государственный технологический университет (Россия, г. Краснодар, ул. Московская, 2)

E-mail: ilur@mail.ru

***Bel'chenko Il'ya Vladimirovich***
Postgraduate student, Kuban State Technological University
(2 Moskovskaya street, Krasnodar, Russia)

***Дьяченко Роман Александрович***
доктор технических наук, доцент, директор института компьютерных систем и информационной безопасности, Кубанский государственный технологический университет (Россия, г. Краснодар, ул. Московская, 2)

E-mail: emessage@rambler.ru

***D'yachenko Roman Aleksandrovich***
Doctor of engineering sciences, associate professor, director of the Institute of Computer Systems and Information Security, Kuban State Technological University (2 Moskovskaya street, Krasnodar, Russia)

---

УДК 004.043

**Бельченко, И. В.**

**Методика повышения производительности информационной системы за счет оптимальной реструктуризации данных** / И. В. Бельченко, Р.А. Дьяченко // Известия высших учебных заведений. Поволжский регион. Технические науки. – 2018. – № 1 (45). – С. 26–38. – DOI 10.21685/2072-3059-2018-1-3.