На правах рукописи

Баранчиков Павел Алексеевич

АЛГОРИТМЫ ОРГАНИЗАЦИИ И МОДЕЛИ ОГРАНИЧЕНИЯ ДОСТУПА К ОТДЕЛЬНЫМ ЗАПИСЯМ ТАБЛИЦ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ

Специальность: 05.13.11 – «Математическое обеспечение вычислительных машин, комплексов и компьютерных сетей»

АВТОРЕФЕРАТ

диссертации на соискание ученой степени
кандидата технических наук

Рязань 2009

Работа выполнена в ГОУ ВПО «Рязанский государственный радиотехнический университет».

Научный руководитель:	доктор технических наук, профессор **Пылькин Александр Николаевич**
Официальные оппоненты:	доктор технических наук, профессор **Скворцов Сергей Владимирович** кандидат технических наук **Буланкин Валерий Борисович**
Ведущая организация:	Филиал ФГУП «ГНПРКЦ «ЦСКБ-Прогресс» ОКБ «СПЕКТР»

Защита диссертации состоится «23» декабря 2009 г. в 14 часов на заседании диссертационного совета Д 212.211.01 в ГОУ ВПО «Рязанский государственный радиотехнический университет» по адресу: 390005, г. Рязань, ул. Гагарина, 59/1.

С диссертацией можно ознакомиться в библиотеке ГОУ ВПО «Рязанский государственный радиотехнический университет».

Автореферат разослан «20» ноября 2009 г.

Ученый секретарь
диссертационного совета
кандидат технических наук, доцент Пржегорлинский В.Н.

ОБЩАЯ ХАРАКТЕРИСТИКА РАБОТЫ

Актуальность темы. Информационно-телекоммуникационные технологии интенсивно внедряются во все сферы человеческой деятельности.

Большинство современных систем управления базами данных (СУБД) предоставляют возможность ограничения доступа пользователей к объектам базы данных (БД), к которым относят таблицы, представления, пакеты, хранимые процедуры, последовательности и схемы. При этом вопросам ограничения доступа к отдельным записям таблиц БД не уделено должного внимания.

Единичные СУБД, такие как Линтер, предоставляют возможность установки меток безопасности на записи таблиц БД. Большинство промышленно эксплуатируемых СУБД не имеют подобных возможностей.

Использование в реляционных БД стандартного языка SQL позволяет производить одинаковые действия на различных СУБД, выполняя одинаково сформированные запросы. Таким образом, корректно спроектированные информационные системы сегодня могут быть портированы между несколькими СУБД.

Задачи ограничения доступа к отдельным записям возникают все чаще в связи с растущими потребностями в гибкости ограничения доступа в современных ИС, в том числе при использовании их для хранения информации, отнесенной к государственной или коммерческой тайне. На данный момент эти задачи решаются индивидуально, общих формализованных подходов к проектированию такого рода ограничений доступа не существует.

Степень разработанности темы. Вопросам ограничения доступа в БД уделяется достаточно много внимания в отечественной и зарубежной литературе. Значительный вклад в разработку методов ограничения доступа в СУБД внесли работы Гайдамакина Ф.М., Тарасюка М.В., Быкова Я.А., Yasunori I., Krishnamurtu M., Orset J.-M. Несмотря на большое количество работ по данной тематике, анализ литературы показал, что не в полной мере использованы возможности для решения рассматриваемых задач.

Цель работы — сокращение сроков проектирования схем БД при необходимости ограничения доступа к отдельным записям таблиц БД и повышение качества проектируемых схем за счет использования предложенных алгоритмов и способов ограничения доступа, осуществляющих ограничение по различным моделям доступа.

Для реализации этой цели должны быть решены следующие проблемы:

– анализ возможных способов ограничения доступа с выбором наиболее подходящего для конкретной задачи и совместимого с большинством реляционных БД;

– выбор средств реализации ограничения доступа, позволяющих реализовать ограничение доступа на большинстве СУБД.

Задачи исследования. Для поставленной цели диссертационной работы необходимо решить следующие задачи.

1. Адаптация моделей ограничения доступа, применяемых в файловых системах, для использования в реляционных БД.
2. Адаптация моделей ограничения доступа, применяемых в файловых системах, для использования при маскировке данных в реляционных БД.
3. Исследование проблем, связанных с введением ограничения доступа к записям в реляционных СУБД.
4. Разработка алгоритмов организации ограничения доступа к записям таблиц реляционных БД с использованием адаптированных моделей доступа.
5. Разработка алгоритмов организации маскировки записей в таблице реляционных БД с использованием адаптированных моделей доступа.
6. Разработка алгоритмов поддержания необходимого количества ложной информации в маскируемых таблицах.
7. Создание модели промышленной БД для оценки производительности предложенных алгоритмов.
8. Создание ПО автоматизированного тестирования производительности предложенных алгоритмов.

Методы исследования. Основные теоретические положения, выводы и экспериментальные результаты получены с использованием математического аппарата реляционной алгебры, булевой алгебры, теории вероятностей и теории множеств.

Публикации. По итогам исследований опубликовано 7 работ, в том числе 3 в журналах, рекомендованных ВАК, и 3 в материалах Всероссийских и Международных научно-технических конференций. В Федеральной службе по интеллектуальной собственности, патентам и товарным маркам (РОСПАТЕНТ) зарегистрирована 1 программа для ЭВМ (свидетельство № 2009614270).

Апробация работы. Результаты настоящей работы докладывались на 3 конференциях, в том числе на Всероссийской научно-технической конференции студентов, молодых ученых и специалистов «Новые информационные технологии в научных исследованиях и образовании», г. Рязань, 2008 г., 15-й Международной научно-технической конференции «Проблемы передачи и обработки информации в сетях и системах телекоммуникаций», Рязань, 2008 г., 34-й Всероссийской научно-технической конференции «Информационные и телекоммуникационные технологии», Рязань, 2009 г.

Научная новизна. В диссертационной работе произведена адаптация моделей ограничения доступа, применяемых в файловых системах, для использования при ограничении и маскировке записей в таблицах БД, что поз-

воляет существенно расширить возможности ограничения доступа в реляционных БД. Модели адаптированы для применения в СУБД, поддерживающих язык SQL, что позволяет использовать их в большинстве современных реляционных СУБД и не препятствует созданию приложений, свободно портируемых между СУБД. Произведен анализ проблем, возникающих при ограничении доступа к записям и маскировке записей таблиц БД, таких как конфликт ключей отношения и автоматическая генерация ложной информации для обеспечения правдоподобной маскировки записей.

Разработанные алгоритмы имеют допустимое для промышленного использования время обработки созданных запросов.

При проведении исследований в рамках диссертационной работы получены следующие новые научные результаты.

1. Адаптация кластеризационной, мандатной, функциональной и дискреционно-ролевой моделей ограничения доступа для использования их в СУБД.

2. Процедура ограничении доступа к записям таблиц реляционных БД, учитывающая конфликт ключей отношения.

3. Адаптация мандатной и функциональной моделей ограничений доступа для использования их при маскировке записей в СУБД.

4. Создание алгоритмов поддержания необходимой ложной информации для маскировки записей в БД.

5. Предложены алгоритмы организации ограничения доступа и маскировки данных на основе языка SQL.

Достоверность научных положений подтверждается:

– корректностью используемого математического аппарата реляционной алгебры, булевой алгебры, теории вероятностей и теории множеств.

– экспериментальным исследованием предложенных алгоритмов с использованием специально разработанной модели промышленной БД.

Практическая значимость работы. На основе полученных результатов автором предложены алгоритмы проектирования схем БД с ограничением доступа и маскировкой записей в таблицах реляционной БД. Разработанные алгоритмы позволяют как проектировать схемы новых БД, так и модернизировать существующие.

Разработанные алгоритмы показывают хорошую производительность для БД различного размера — от малых до промышленных.

Основные положения, выносимые на защиту:

1) адаптированы кластеризационная, мандатная, функциональная и дискреционно-ролевая модели ограничения доступа для использования в таблицах реляционных БД;

2) разработана процедура ограничения доступа к записям таблиц реляционных БД, учитывающая конфликт ключей отношения;

3) разработаны алгоритмы поддержания необходимой ложной информации при маскировке записей в таблицах реляционных БД.

Реализация и внедрение результатов работы. Результаты исследований внедрены:

– в ООО «ИНФОСТ-ПРОЕКТ» (г. Рязань) при разработке ПО автоматизации работы отдела технического надзора и ПО автоматизации ведения генерального плана предприятия;

– в ООО «Арго-Тур» (г. Рязань) при разработке ПО бронирования туристических путевок через Интернет;

– в учебном процессе ГОУ ВПО «Рязанский государственный радиотехнический университет» при обучении студентов специальностей 230101, 230104 и 230105;

– в Филиале ОАО «СО ЕЭС» Рязанское РДУ в ПО автоматизации ведения учета оборудования и программного обеспечения программно-аппаратного комплекса.

Структура и объем диссертации. Диссертация состоит из введения, четырех глав, заключения, списка литературы (77 источников), изложенных на 223 страницах, содержит 49 рисунков и 67 таблиц.

КРАТКОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность темы, определены цели и задачи исследований.

В первой главе описана краткая история развития ограничения доступа в реляционных СУБД (работы Э. Кодда, М. Стоунбрейкера и Д. Мейера).

Описаны существующие на данный момент способы организации ограничения доступа к данным внутри таблиц:

1) использование сервера приложений;

2) использование специального интерфейса, ограничивающего доступ;

3) создание отдельной таблицы для каждого класса записей сущности;

4) создание отдельного представления пользователя для каждого класса записей в сущности;

5) создание «динамического» представления пользователя, которое позволяет просматривать и редактировать только определенные записи (см рисунок 1).

Изложены основные модели ограничения доступа в файловых системах:

1) дискреционная (discretionary access control — DAC);

2) ролевая (role-based access control — RBAC);

3) мандатная (mandatory access control — MAC);

4) функциональная;
5) кластеризационная.

Приведено математическое описание этих моделей.

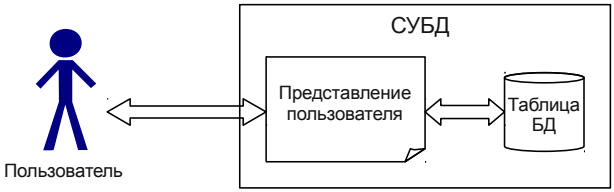

Рисунок 1 — Схема взаимодействия пользователя с БД через представление пользователя

Во второй главе рассматривается организация ограничения доступа к записям таблиц.

Назовем классом записей множество записей одного отношения, доступ к которым предоставлен одинаково.

Создадим представление пользователя, в котором выбираются только записи, к которым пользователь имеет доступ на основе специального предиката:

$$r_u = \sigma_{\beta(u)}(r). \quad (1)$$

Для реализации используются служебные таблицы с данными о доступе пользователей к классам записей. Пользователи, не имеющие доступа к данным таблицы, не получают доступ к соответствующему представлению.

Будем записывать предикат без параметра:

$$r_u = \sigma_{\beta}(r). \quad (2)$$

При ограничении доступа к записям таблиц БД возникает конфликт ключей отношения при добавлении записей в таблицу с ограничением доступа.

Пусть в r существует ключ K. Пусть k_k – набор конкретных значений атрибутов, входящих в ключ K. Обозначим через $r_j(k_k)$ i-ю запись, у которой значения ключевых атрибутов совпадают с k_k. Тогда возможна следующая ситуация:

$$\exists r_j(k_k),\ r_i(k_k) \notin r_u. \quad (3)$$

Тогда в r существует запись $r_i(k_k)$, идентифицируемая значением ключа k_k, к которой пользователь U не имеет доступа. Он не знает о ее существовании и может пытаться добавить r_j в r такую, что ее значения ключевых атрибутов совпадут со значениями r_i. После добавления записи пользователь имеет к ней доступ:

$$r_i(k_k) \notin r_u, r_j(k_k) \in r_u, \Rightarrow r_j \neq r_i. \quad (4)$$

r_i и r_j не совпадают по принадлежности к подмножествам записей r_u, видимых пользователю. r_i и r_j могут иметь различные значения атрибутов

кроме ключевых. Назовем такую ситуацию конфликтом ключей отношения (ККО).

Данная ситуация противоречит теории, определяющей ключи отношения как наборы атрибутов, однозначно определяющие все атрибуты отношения.

Предложено несколько основных вариантов разрешения ККО.

1. Блокировка изменений. Пользователю выдается сообщение о непредоставлении доступа. Блокировка изменений наиболее проста в реализации.

2. Изменение доступа к записи. Добавляемую запись относят к истинной или ложной по совпадению некоторых атрибутов. Если запись идентифицирована как верная, пользователю предоставляется доступ к ней.

Предлагается ввести служебное отношение *q* со схемой *Q*:

$$R=\{K\,A\,B\,C\,\ldots\};\,Q=\{K\,U\}. \tag{5}$$

q показывает, какие пользователи имеют доступ к соответствующим записям *r(R)*. Значение *U* идентифицирует пользователя, которому предоставлен доступ к записи, с ключом, совпадающим со значением ключа *K*. Выборка записей, доступных пользователю, будет в таком случае определена как

$$rw_u=\sigma_{F(U)}(r\bowtie q). \tag{6}$$

Здесь *F(U)* — функция, определяющая наличие доступа.

Выбор варианта обработки ККО следует производить индивидуально для каждой ситуации.

Ограничение доступа при использовании кластеризационной модели ограничения доступа. Пусть все множества записей r_{Ui}, к которым имеют доступ пользователи U_i, не пересекаются, а их объединение дает отношение *r*:

$$r_{u_1}\cap r_{u_2}\cap\ldots\cap r_{u_n}=\varnothing;\,r_{u_1}\cup r_{u_2}\cup\ldots\cup r_{u_n}=r. \tag{7}$$

Каждая запись принадлежит только одному из подмножеств r_u (кластеру). Такой доступ назовем кластеризационным.

Тогда в ключ *K* можно ввести атрибут *C*, определяющий, к какому множеству r_u принадлежит запись. Схема отношения примет вид:

$$R=\{A,B,\ldots\};\,R'=\{A,B,\ldots,C\}. \tag{8}$$

Назовем *C* идентификатором кластера. Множество записей, «видных» пользователю U_i, можно описать выборкой из видоизмененного отношения *r′*:

$$r_u=\sigma_{C=C_u}(r'), \tag{9}$$

где C_u — идентификатор кластера данного пользователя.

Ограничение доступа при использовании дискреционно-ролевой модели доступа. Модель доступа, в которой у объекта имеется владелец,

управляющий доступом к нему и имеющий полный доступ к объекту, называется дискреционной. Модель доступа, в которой к объекту предоставляется доступ по вхождению в определенную роль, называется ролевой моделью доступа. Дискреционно-ролевая модель ограничения доступа объединяет в себе обе модели.

Пусть имеется *r(R)*, доступ к его записям следует разграничить. Добавим атрибуты U, G, R_g, содержащие информацию о правах доступа к записям, и новая схема отношения R' будет выглядеть следующим образом:

$$R' = R \cup G R_g, \tag{10}$$

где G — группа пользователей, к которой относится эта запись; R_g — права доступа к записи для группы.

Права доступа могут располагаться следующими способами:

1) хранятся непосредственно в отношении (10);

2) выносятся в отдельное отношение s с ключом K_r;

3) вместе с указанием пользователя и группы выносятся в отношение s с ключом K_r, который входит в результирующее отношение r'.

Используем 3 отношения: *u(U)* — отношение пользователей, *g(G)* — отношение групп, g_u*(Gu)* — отношение связи пользователей и групп:

$$U = K_u N_u U_g R_d; G = K_g N_g; Gu = K_u K_g, \tag{11}$$

где K_u — ключ отношения пользователей; N_u — имя пользователя; U_g — группа пользователя по умолчанию; R_d — права доступа пользователя по умолчанию; K_g — ключ отношения групп; N_g — имя группы пользователей.

Функция *Gu (u, g)* определяет, входит ли пользователь u в группу g.

Введем предикат β для определения наличия доступа к записи:

$$\beta = (k_u = user) \vee (Gu(user, K_g) \wedge F(R_g)), \tag{12}$$

где k_u — идентификатор владельца записи; K_g — идентификатора группы; $F(R_g)$ — функция, определяющая, предоставлен ли группе доступ заданного типа; R_g — совокупность прав доступа, определяющая предоставление/непредоставление группе 4-х типов доступа.

Хранение привилегий в защищаемом отношении. Схема видоизмененного отношения *r'(R')* примет вид:

$$R' = R K_u K_g R_g. \tag{13}$$

При выборке записей, доступных пользователю на совершение одной операции, будет использоваться предикат β из (12):

$$r_u = \sigma_\beta (R' \bowtie U). \tag{14}$$

Понадобится *p(P)* с ключом K_p и атрибуты доступа группы и владельца. Тогда схема отношения r' примет вид:

$$R' = \{R K_p\}; \; P = \{K_p U G R_g\}. \tag{15}$$

Выборка записей, к которым пользователь имеет доступ, будет иметь вид:

$$r_u = \sigma_\beta(R' \bowtie U \bowtie P). \quad (16)$$

Ограничение доступа к записям при использовании мандатной модели ограничения доступа. Пользователям и записям сопоставлены значения мандатов. Пользователь имеет доступ к записи, если его мандат не ниже мандата записи. Отношение *r'(R')* представит измененное защищаемое отношение после расширения его служебным атрибутом:

$$R' = \{K_1 K_2 \ldots K_n A_1 A_2 \ldots A_n M_r\}, \quad (17)$$

где A_i — атрибуты *r(R)*; M_r — мандат записи; K_i — ключевые атрибуты *r(R)*.

Предикаты доступа пользователя будут иметь вид:

$$\beta_r = (M_r \le M_u);\ \beta_u = (M_r = M_u);\ \beta_d = (M_r = M_u). \quad (18)$$

Выборка данных r_u, доступных пользователю, будет иметь вид:

$$r_u = \sigma_{M_r \le M_u}(r'), \quad (19)$$

где M_u — мандат пользователя.

Чтобы скрыть мандаты записей, воспользуемся проекцией r_u на R:

$$r_u = \pi_R(\sigma_{M_r \le M_u}(r')). \quad (20)$$

Добавляемой записи присваивается заданный по умолчанию мандат. Так можно реализовать только мандатную модель ограничения доступа, описанную в (18), с ограничением мандата добавляемой записи, в то время как классическая модель ограничения доступа позволяет добавлять записи с мандатом, превышающим мандат пользователя.

Пусть *r(R)* — защищаемое отношение, а *s(S)* — его дочернее:

$$r(R) = \{K_r A_1 A_2 \ldots A_n\};\ s(R) = \{K_r K_s B_1 B_2 \ldots B_m\}, \quad (21)$$

где K_r — ключ отношения r; K_s — дополнительный атрибут, который совместно с K_r составляет ключ отношения s; B_i — неключевые атрибуты отношения s.

Функциональные зависимости атрибутов можно выразить:

$$K_r \to A_1 A_2 \ldots A_n,\ K_r K_s \to B_1 B_2 \ldots B_m. \quad (22)$$

При добавлении M_r в (17) ключ r не изменяется, функциональные зависимости в (22) также остаются без изменения.

Выборка данных из дочернего отношения s_u может осуществляться по критерию вхождения ключа родительской записи в выборку r_u:

$$s_u = \sigma_{K_r \in \pi_{Kr}(r_u)}(s). \quad (23)$$

Ограничение доступа при использовании функциональной модели ограничения доступа. Пусть все защищаемые объекты можно соотнести с одной или несколькими функциями, для выполнения которых необходим доступ к этим объектам. Обозначим такие функции через F. Пусть пользователи

имеют доступ к определенным функциям *Fs* из *F*. Обозначим функции, с которыми соотнесены защищаемые данные, через *Fo*:

$$F_s \subseteq F; F_o \subseteq F. \quad (24)$$

Пусть доступ к данным предоставляется только в том случае, когда набор функций пользователя *Fs* включает в себя набор функций *Fo*, с которыми соотнесены данные. Назовем такую модель доступа функциональной. Предикат наличия доступа будет иметь вид:

$$\beta = F_o \subseteq F_s. \quad (25)$$

Данные, к которым имеет доступ пользователь, можно обозначить так:

$$r_u = \sigma_{F_o \subseteq F_s}(r). \quad (26)$$

Назовем такие функции *F* областями доступа *AA*.

Пусть в системе имеется множество областей доступа *aa*. Обозначим набор областей, к которым имеет доступ пользователь *u*, через aa_u. Обозначим набор областей, необходимых для доступа к записи *j* отношения, через aa_{rj}:

$$aa_u \subseteq aa;\ aa_{r_j} \subseteq aa. \quad (27)$$

Структура данных в таком случае будет иметь следующий вид:

$$AA_u = \{U\ AA\};\ R' = \{K\ A_i A_2 \dots A_n\};\ AA_r = \{K\ AA\}. \quad (28)$$

Выборка истинных данных, доступных пользователю, будет иметь вид:

$$r_u = \sigma_{K \in \sigma_{U=user}(a_u \bowtie a_r)}(r'). \quad (29)$$

Разработанный алгоритм организации ограничения доступа приведен на рисунке 2.

В третьей главе рассматривается маскировка данных.

Пусть пользователю известно, что в отношении *r* хранится запись r_i. Пусть он не имеет к ней доступа. Пусть пользователь знает значения атрибутов $A_1 \dots A_n$ записи r_i, что позволяет ему судить о наличии r_i в отношении *r*. Обозначим остальные атрибуты, значения которых пользователю неизвестны и они подлежат скрытию, как $B_1 \dots B_n$.

Предоставим пользователю в результате выборки запись r_j:

$$r_i = (k_1 \dots k_n a_1 a_2 \dots a_n b_1 b_2 \dots b_n);\ r_j = (k_1 \dots k_n a_1 a_2 \dots a_n b'_1 b'_2 \dots b'_n), \quad (30)$$

где a_k — значение атрибута A_k в записи r_i; b_k — значение атрибута B_k в записи r_i, к которой пользователь не имеет доступа; b'_k — значение атрибута B_k в записи r_j, являющейся ложной; k_i — значения ключевых атрибутов K_i (далее по тексту — *K*).

Назовем эту технику маскировкой данных в отношении.

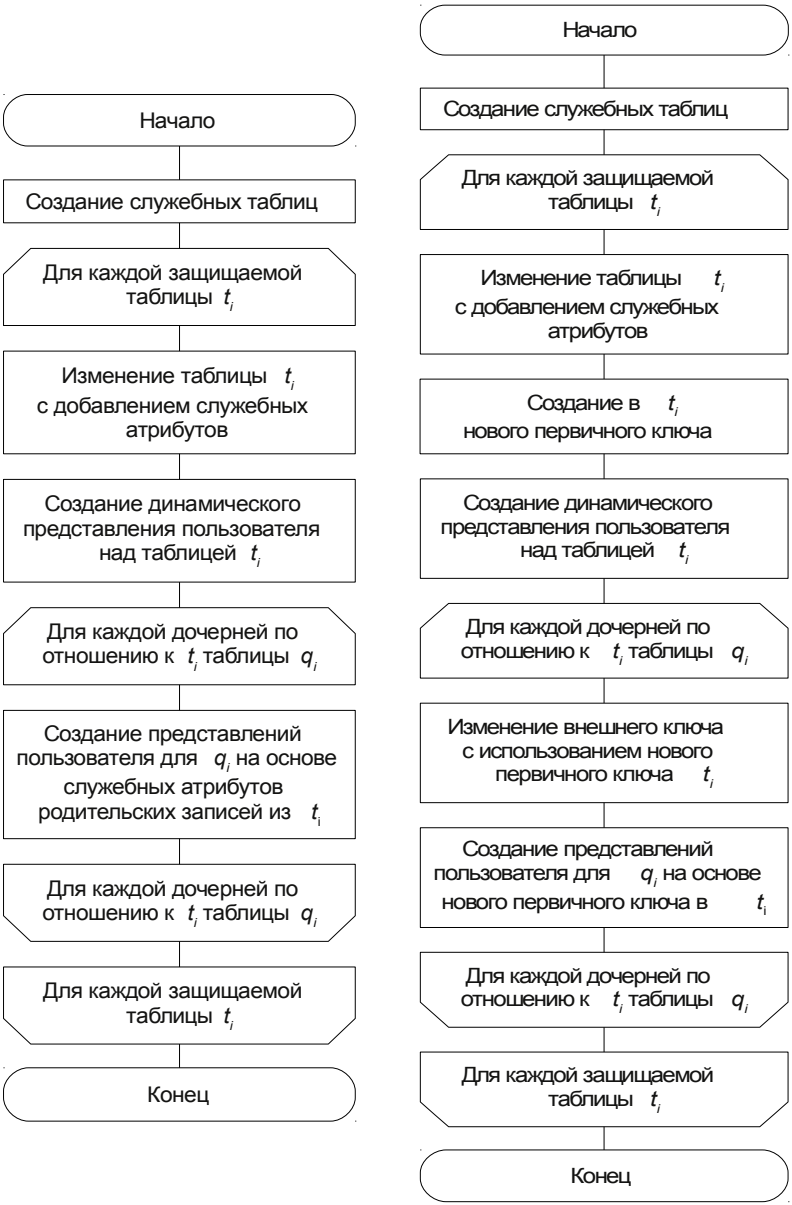

Рисунок 2 — Алгоритмы организации ограничения доступа (слева) и маскировки данных (справа)

Область применения маскировки данных в БД является более узкой по сравнению с ограничением доступа.

Алгоритм организации маскировки данных в записях таблиц реляционных БД приведен на рисунке 2.

Организация хранения истинной и ложной информации. Предложены 3 различных варианта организации хранения ложной информации.

1. Хранение всей информации в одном отношении.
2. Декомпозиция отношения на основе секретности атрибутов.
3. Хранение ложной информации в отдельном отношении.

Анализ вариантов показал преимущество первого варианта.

Изменения дочерних отношений. Для организации взаимосвязи с дочерними отношениями предлагается введение дополнительных CF-зависимостей.

Введем в защищаемое отношение идентификатор записи *Id*. Атрибут *Id* будет альтернативным ключом для *r′*:

$$R=\{K\,A_1\ldots A_n\};\ R'=\{K\,L\,Id\,A_1\ldots A_n\}, \tag{31}$$

где L — атрибут, определяющий доступ к записи в защищаемом отношении.

Пусть дочернее отношение имеет вид:

$$S=\{K_s B_1\ldots B_n K\}. \tag{32}$$

В *s(S)* вместо ключа *r* будет введенный служебный ключ отношения *r′*:

$$S'=\{K_s B_1\ldots B_n Id\}. \tag{33}$$

Выборка данных, доступных пользователю, будет иметь следующий вид:

$$\sigma_{\exists r'(Id)}(S'). \tag{34}$$

Выборка данных для пользователя должна содержать схему *S*:

$$s_u=\pi_S(s'\bowtie r_u). \tag{35}$$

Выборка данных из дочернего отношения является более трудоёмкой по сравнению с самим защищаемым отношением.

Мандатный доступ. Отношение *r′(R′)* представит измененное защищаемое отношение после расширения его служебным атрибутом:

$$R'=\{K_1 K_2\ldots K_n A_1 A_2\ldots A_n M_r\}. \tag{36}$$

Выборка данных r_{ua}, доступных пользователю, будет иметь вид:

$$r_{ua}=\sigma_{M_r\le M_u}(r'). \tag{37}$$

При реализации маскировки данных в мандатной модели доступа достаточно для каждой истинной записи r_{it} в защищаемом отношении создать одну ложную r_{if}, установив у нее минимальное значение мандата записи.

$$r_{it}=(k_1 k_2\ldots k_n a_1 a_2\ldots a_n m_i),\ r_{if}=(k_1 k_2\ldots k_n a'_1 a'_2\ldots a'_n m_{min}), \tag{38}$$

где a_1, a_2 ... a_n — атрибуты истинной записи r_{it}; a'_1, a'_2 ... a'_n — атрибуты ложной записи. Эти атрибуты скрывают (маскируют, подменяют) значения атрибутов a_1, a_2 ... a_n; k_1, k_2 ... k_n — ключевые атрибуты отношения, одинаковые в истинной и ложной записях, что позволяет пользователю принимать запись r_{if} за запись r_{it}; m_i — мандат истинной записи; m_{min} — минимальное допустимое значение мандата в ИС. В описанной ситуации ложная запись будет видна всем пользователям.

Чтобы исключить дублирование ключевых атрибутов, необходимо выбрать только одну из записей с заданным ключом, доступным пользователю. Мандат истиной записи всегда больше мандата ложной записи. Выбор записи осуществляется на основе максимального значения мандата:

$$r_u = \sigma_{m=max(m)}(\sigma_{m \le m_u}(r')). \quad (39)$$

Внешние ключи. Пусть дочернее отношение s будет иметь схему S:

$$S = \{K_{s1} K_{s2} \dots K_{sn} K_p A_{s1} A_{s2} \dots A_{sn}\}, \quad (40)$$

где A_{si} — неключевые атрибуты дочернего отношения $s(S)$; K_{si} — ключевые атрибуты дочернего отношения $s(S)$; K_p — атрибут, являющийся внешним ключом на родительское отношение r. После каскадного добавления мандата в дочернюю таблицу получим новую схему отношения $s'(S')$:

$$S' = \{K_{s1} K_{s2} \dots K_{sn} K_p A_{s1} A_{s2} \dots A_{sn} M_p\}, \quad (41)$$

где M_p — мандат родительской записи.

Функциональный доступ. Пусть в системе имеется множество областей доступа aa. Обозначим набор областей, к которым имеет доступ пользователь u, через aa_u. Обозначим набор областей, необходимых для доступа к записи j отношения через aa_{rj}.

$$aa_u \subseteq aa, aa_{r_j} \subseteq aa. \quad (42)$$

Пользователю предоставляется доступ к истинной записи, если его области доступа включают в себя области доступа, требуемые для записи:

$$aa_{r_j} \subseteq aa_u. \quad (43)$$

Для того чтобы полностью реализовать наличие записи в выборках всех пользователей, необходимо выбрать множество наборов aa_r, необходимых для доступа к конкретным атрибутам отношения r. В общем случае это будет множество всех подмножеств множества aa (булеан aa). Однако учитывая, что атрибуты отношения r имеют не все области доступа, многие области доступа оказываются невостребованными.

Структура данных в таком случае будет иметь следующий вид:

$$AA_u = \{U\ AA\},\ R = \{K\ A_1 A_2 \dots A_n\},\ AA_r = \{K\ AA\}. \quad (44)$$

Выборка истинных данных, доступных пользователю, будет иметь вид

$$\sigma_{K\in\sigma_{U=user}(a_u\bowtie a_r)}(r). \quad (45)$$

Организация хранения. Для реализации такого доступа требуется сложная связь между отношениями — «многие-ко-многим».

Все возможные варианты комбинаций областей доступа будут булеаном множества *aa*. Назовем его *AACS*.

$$AACS=P(AA). \quad (46)$$

Обозначим *AAC* как подмножество *aa*. Оно будет членом множества *AACS*. Описание прав доступа пользователей к записям с конкретным набором на записи, а также областей доступа, необходимых для доступа к записям, будет иметь вид:

$$AA_u=(AAC\ U),\ AA_r=(AAC\ K), \quad (47)$$

где AA_u и AA_r — соответственно схема введенного ранее отношения, описывающего набор областей доступа aa_u, доступных пользователям, и набор прав доступа aa_r, необходимых для доступа к записи с ключом *K*.

Фактически атрибут *AAC* обозначает множество областей доступа *AA*, которое в (30) задавалось набором записей в aa_u или aa_r. В этих отношениях ключами будут только идентификатор пользователя или записи соответственно.

Связь между aa_r и *r* будет иметь тип «один-к-одному», что позволяет опустить использование aa_r, а его атрибут *AAC* перенести в отношение *r′*.

Выборка доступных истинных записей примет вид:

$$r_{ua}=\sigma_{U=User\wedge aa_u.AAC\supseteq r'.AAC}(r'aa_u). \quad (48)$$

Возникает проблема хранения множеств *AAC* в *r′* и aa_u. В общем случае следует создать несколько отношений, характеризующих эти множества. aa_r будет отображать множества областей для записи защищенного отношения *r′*, а aa_u — множество областей доступа, предоставленных пользователю:

$$AA_r=(AA\ K'),AA_u=(AA\ U), \quad (49)$$

где *K′* — ключ записи защищенного отношения *r′*.

Для определения, имеет ли пользователь доступ к конкретной записи, введем предикат *AG*:

$$AG(k_c,u_c)=\sigma_{k_c=K'}(aa_r)\subset\sigma_{u_c=U}(aa_u). \quad (50)$$

Параметр k_c предиката *AG* идентифицирует ключ записи, доступ к которой проверяется. u_c идентифицирует субъект доступа. Возможно, что ключ отношения *r* вообще не входит в ключ отношения *r′*.

Отношение aa_r соответствует одному защищаемому отношению *r*. При защите других отношений следует либо создать дополнительные отношения с схемой AA_R, либо расширить отношение aa_r атрибутом, отражающим

принадлежность области доступа к определенному защищаемому отношению.

Введем функцию *M(k')*, которая будет определять количество областей доступа, относящихся к записи отношения *r'*, по значению ключа *k'*:

$$M(k')=|\sigma_{aa_r.k'=k'}(aa_r)|. \quad (51)$$

Поскольку в множестве комбинаций областей доступа *AAC* предусмотрены все их комбинации, то среди записей, удовлетворяющих *AG*, выбираются записи с максимальным количеством областей доступа записи. Выборка одной записи r_m со значением k_i ключа исходного отношения *r* будет выглядеть так:

$$r_m(k_i)=\sigma_{k=k_i \wedge AG(k',u)=true \wedge M(k')=max}(r'). \quad (52)$$

Строка *M(k')=max* обозначает выборку строки с максимальным значением M(k'). Эта выборка реализуется с помощью стандартного языка SQL.

Реализация предиката определения доступа. Реализация предиката *AG* на SQL довольно сложна, несмотря на простоту записи SQL-выражения, время выполнения ее будет велико.

```
select count(*) into count_r from aacr where kc=K' and aa not
in
       (select count(*) into count_u from aau where
uc=username)
return count=0;
```

Такой предикат можно определить без процедурного языка:

```
… where (select count(*) from aacr where    kc=K and aa not
in
 (select count(*) into count_u from aau where uc=username))...
```

Комплексное условие будет перемещено в секцию *where* запроса, а это значит, что в одном запросе будут выполняться множества подзапросов, вложенных друг в друга, что приведет к потерям производительности.

Использование побитной карты. Существуют приемы оптимизации подобных задач. Пусть мы имеем ИС, в которой количество областей доступа имеет верхний предел *N*. Тогда множество областей доступа можно будет обозначить одним *N*-битным числом, в котором каждой области доступа будет соответствовать определенный бит. Удобно использовать идентификаторы областей доступа для нумерации битов, которым соответствуют области доступа.

Здесь отпадает необходимость в использовании aa_r и aa_u. Вместо них в отношение *r'* будет введен атрибут, определяющий множество областей доступа aa_r в битовом формате. Аналогично поступим и с таблицей пользователей:

$$R'=\{K' \; AA_r \; A_1 \; A_2 \; ...\}, U'=\{U \; AA_u\}. \quad (53)$$

Предикат *AG* упростится с реляционного до логического:

$$AG(r_c, u_c) = ((aa_r(r_c) \rightarrow aa_u(u_c)) = 11..11_2). \quad (54)$$

Здесь используется логическая побитная импликация. Под записью $11...11_2$ следует понимать число в двоичной кодировке. Количество единиц соответствует количеству областей доступа или превосходит его и будет равно разрядности полей aa_r и aa_u.

Выражения $aa_r(r_c)$ и $aa_u(u_c)$ обозначают выборку соответствующего поля из отношения r' и таблицы пользовательского доступа w'.

Скорость вычисления количества областей доступа конкретной записи $M(k')$ будет существенно снижена по сравнению с предыдущим вариантом реализации.

При использовании побитной карты количество областей доступа ограничено разрядностью побитных карт. Это снижает масштабируемость системы контроля доступа.

Синхронизация данных. Возможно возникновение аномалий, при которых пользователю в результате выборки будут выдаваться дублирующиеся («лишние») записи, а некоторые необходимые записи могут отсутствовать.

Пусть существует отношение r со схемой R, которое в результате маскировки дополняется некоторым набором служебных атрибутов и ложных записей и имеет схему R'. Обозначим измененное отношение как r'.

Пусть непривилегированные пользователи не могут добавлять информацию в маскируемую таблицу. Тогда, чтобы обеспечить маскировку новых добавленных записей, необходимо создать соответствующие им ложные. Если ставится задача предоставления пользователям доступа на добавление записи в маскируемую таблицу, то для ее реализации необходимо производить синхронизацию данных в реальном времени в процессе добавления записей непривилегированными пользователями.

Пусть в r' добавлена запись r_t с ключом k_t, недоступная пользователю u. При выборке данных пользователю не будет предоставлена запись с ключом k_t, так как пользователь u не имеет доступа к записи r_t. Для реализации маскировки необходимо создать такую запись r_f в отношении r', чтобы значения ее ключевых атрибутов совпадали со значениями ключевых атрибутов записи r_t. Значения несекретных атрибутов при необходимости должны быть скопированы из r_t в r_f. Значения секретных атрибутов должны быть правдоподобны.

Точки входа в алгоритм. Необходимо реализовать автоматическое создание ложных записей r_f для обеспечения существования ложных записей для r_t в каждый момент времени. Для этого можно применять несколько методов:

– использование периодической генерации ложных записей;
– использование триггеров на добавление в отношение r';
– использование триггеров в представления пользователя над r';

– использование «виртуальных БД», таких как dbms_rls в Oracle.

Алгоритм проверки «лишних» записей. В отношении r' может быть несколько записей r_f. В таком случае необходимо создать записи r_{fi} так, чтобы каждому пользователю было «видно» одну и только одну запись r_{fi}. Для каждого алгоритма выбора ложной записи для конкретного пользователя построим алгоритм определения множества L меток выбора ложной информации *algF*.

Основной метод проверки наличия аномалий — перебор всех различных пользователей и/или пользовательских ролей и просмотр того, сколько записей r_{fi} будут присутствовать в выборке.

1. Для каждого пользователя БД.

1.1. Агрегативный запрос к БД на количество записей с одним значением ключа. Запрос имеет вид:

```
Select count(*),K
 from r' group by K having count(*) > 1
```

1.2. Если хотя бы один результат запроса больше 1 — для каждого ключа из полученной выборки, — удалить «лишние» ложные записи.

1.3. Если нет ни одной записи, в которой результат запроса будет большим 1, выход из алгоритма.

2. Перейти к пункту 1.

Проверка наличия в БД «лишних» записей является алгоритмом циклической структуры с постусловием. В нем есть функция удаления «лишних» записей, которая будет специфична в зависимости от модели доступа.

Алгоритм добавления «недостающих» ложных записей

1. Для каждого пользователя БД u_i.

1.1. Для каждой истинной записи r_j.

1.1.1. Существует ли запись с ключом $k(r_j)$ в выборке пользователя u_i.

1.1.2. Если нет, создать такую запись.

Для достижения баланса записей в выборках необходимо задать правила выборки ложных записей для удаления и правило генерации новой «ложной» записи при соответствующей необходимости.

Алгоритм генерации ложных записей.

1. Для всех вариантов уровней доступа AL (в данном примере — мандатов записи), при которых предоставляется доступ к некоторым атрибутам.

1.1. Создание новой записи.

1.2. Значения атрибутов, доступ к которым предоставлен для пользователей с уровнем доступа AL, копируются из истинной записи.

1.3. Значения атрибутов, доступ к которым не предоставлен для пользователей с уровнем доступа AL, копируются из записи с ложными данными.

1.4. Записи устанавливается уровень доступа AL.

В четвертой главе описывается построение программного стенда для испытания предложенных алгоритмов.

Цель испытаний — выявление наиболее подходящих запросов для реализации каждой модели доступа на каждой из выбранных СУБД. Оптимизация производится по критерию производительности.

Объектом испытаний являются запросы, осуществляющие ограничения доступа по различным моделями, и СУБД.

В силу различия в алгоритмах вычисления и оптимизации запросов в различных СУБД вычислительная сложность моделей может различаться.

Все представленные СУБД выполняют один запрос в одном потоке. Распараллеливание работы идет только в случае одновременных нескольких запросов. Один поток обязательно выполняется в одном ядре центрального процессора ЭВМ. Следовательно, для того чтобы СУБД были предоставлены максимальные ресурсы центрального процессора (ЦП), необходимо иметь ЦП с двумя или более ядрами. Тогда одно из ядер будет занято практически полностью (98-100%), а другое будет реализовывать внутренние нужды операционной системы (ОС).

Для 2-х и более ядер процессора необходимо следить за тем, чтобы в ОС не выполнялось более никаких вычислительно-сложных процессов и обслуживание ОС не перешло на ядро ЦП, в котором выполняется расчет запроса.

Для проведения испытаний наилучшим образом подходит выполнение всех запросов на ЭВМ, не занятой более ничем, кроме данного эксперимента.

Технические характеристики ЭВМ, использованной для тестирования: объем ЖМД — 250 Гб, объем ОЗУ — 4 Гб, частота процессора — 2200 МГц, количество ядер процессора — 2, ОС — Linux 2.6.27.21.

ПО для тестирования запросов. Для проведения такого рода опытов была составлена программа на языке Java, общая схема функционирования которой приведена на рисунке 3.

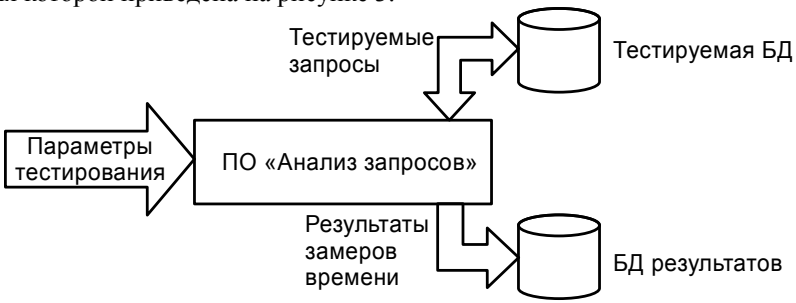

Рисунок 3 — Общая схема функционирования ПО тестирования запросов

Тестируемые запросы хранятся в БД. ПО должно последовательно в различных СУБД выполнять ряд исследуемых запросов и записывать в БД результаты запросов для последующей статистической обработки.

Для нейтрализации эффекта затрат времени на передачу данных по сети, логично рассматривать запросы только на вычисление количества записей. Независимо от оптимизатора запросов, работающего на тестируемой СУБД, для вычисления количества записей СУБД придется произвести все соединения и проверки в отношениях.

Для определения времени выполнения запросов производится фиксация отпечатка времени непосредственно перед запросом и после его выполнения.

Производительности запросов, измеренные с помощью данного ПО, приведены на рисунках 5 и 4.

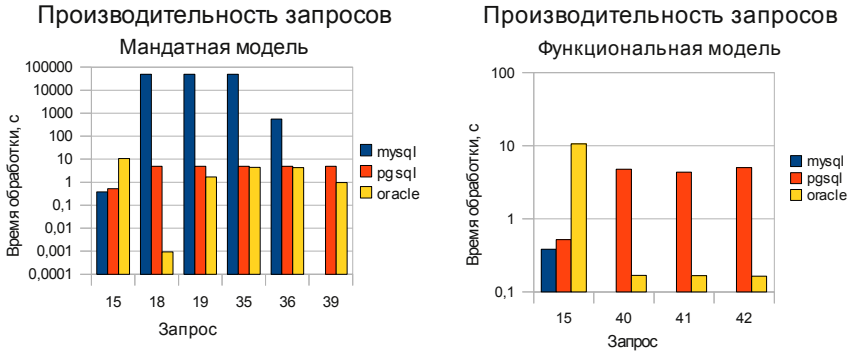

Рисунок 4 — Производительность маскировки записей для мандатной (слева) и функциональной (справа) моделей доступа

ОСНОВНЫЕ РЕЗУЛЬТАТЫ РАБОТЫ

1. Рассмотрены основные схемы ограничения пользовательского доступа к данным. Выделены доступ с ограничением на сервере БД и ограничением на сервере приложений как наиболее полно отвечающие требованиям к ограничению доступа. Обе схемы могут использоваться в зависимости от требований к системе.

2. Описана возникающая в процессе работы системы ограничения доступа ситуация, названная конфликтом ключей отношения, препятствующая использованию встроенных в СУБД механизмов ограничения целостности данных. Разработаны алгоритмы, позволяющие избежать конфликта и уменьшить трудоемкость проектирования схемы БД и администрирования ИС.

3. Разработаны алгоритмы и схемы БД для реализации ограничения доступа к записям таблиц БД с использованием кластеризационной, мандат-

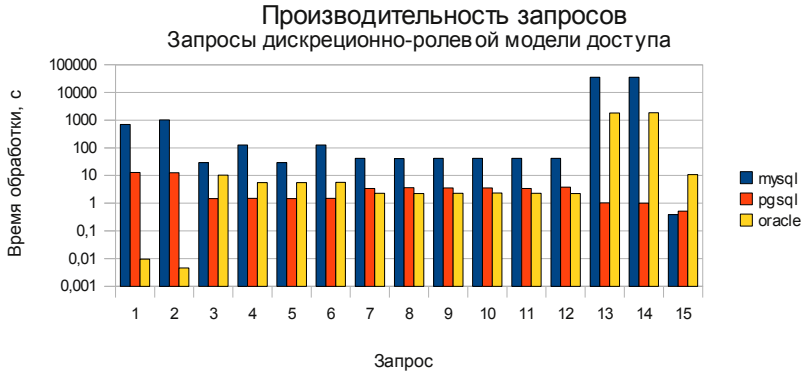

Рисунок 5 — Производительность ограничения доступа для дискреционно-ролевой модели

ной, функциональной и дискреционно-ролевой моделей доступа. Приведены отрицательные и положительные свойства различных вариантов реализации этих моделей доступа в БД путем изменения схемы БД: как изменения схемы защищаемого отношения, так и добавления новых служебных отношений, хранящих информацию о пользователях и правах доступа.

4. Сформулирована задача организации маскировки данных в таблицах БД. Уделено особое внимание «прозрачности» системы ограничения доступа для пользователей, что позволяет снизить вероятность атаки на информационную систему ввиду создания иллюзии предоставления доступа к засекреченной информации.

5. Рассмотрены основные варианты организации хранения истинной и ложной информации при маскировке данных. Проведенный анализ предложенных вариантов показал, что наиболее подходящим для промышленной эксплуатации является хранение истинной и ложной информации в одной таблице.

6. Разработаны схемы и алгоритмы маскировки данных при использовании мандатной и функциональной моделей доступа. Приведены предполагаемые достоинства и недостатки различных вариантов реализации доступа с использованием упомянутых моделей. Алгоритмы подразумевают создание дополнительных служебных таблиц в БД и изменение схемы защищаемых отношений.

7. Предложены конкретные варианты реализации ограничения доступа при использовании мандатной, дискреционно-ролевой и функциональной моделей на языке SQL, совместимые с многими СУБД. Все предложенные варианты были протестированы на СУБД mySQL, PostgreSQL и Oracle. Результаты статистической обработки времени выполнения конкретных реали-

заций запросов позволили сделать заключение о предпочтительности использования того или иного подхода для реализации заданной модели доступа на заданной СУБД.

8. Предложены конкретные варианты реализации маскировки данных при использовании мандатной и функциональной моделей ограничения доступа на языке SQL. В результате статистической обработки замеров производительности запросов выявлены наиболее подходящие по различным критериям оптимизации для различных СУБД.

9. Разработано ПО для тестирования производительности запросов. ПО позволяет тестировать запросы на различных СУБД, замеряя время выполнения запросов. Результаты измерений в дальнейшем подлежат статистической обработке средствами СУБД.

ПУБЛИКАЦИИ ПО ТЕМЕ ДИССЕРТАЦИИ

1. Баранчиков А.И., Баранчиков П.А. Организация доступа к записям таблиц в базах данных // Вестник РГРТА. Вып.20. — Рязань: РГРТА, 2007. С. 78-81.

2. Баранчиков А.И., Баранчиков П.А. Организация доступа к записям таблиц БД по аналогии с POSIX-совместимыми файловыми системами // Проблемы передачи и обработки информации в сетях и системах телекоммуникаций: Материалы 15-й Международной науч.-техн. конф. Часть 1. Рязань: РГРТУ, 2008. С. 13-16.

3. Баранчиков П.А. Разрешение конфликта ключей отношения при ограничении доступа к записям таблиц БД // Новые информационные технологии в научных исследованиях и образовании: материалы XIII Всероссийской научно-технической конференции студентов, молодых ученых и специалистов. Часть I. Рязань:РГРТУ, 2008. С. 121-123.

4. Баранчиков П.А. Конфликт ключей отношения и методы его решения // Вестник РГРТУ №4 (выпуск 26). Рязань, 2008. С. 66-69.

5. Баранчиков А.И., Баранчиков П.А. Проблема синхронизации истинной и ложной информации при маскировке данных в БД // Информационные и телекоммуникационные технологии: Материалы 34-ой всероссийской научно технической конференции. Часть 1. Рязань: РВВКУС, 2009, С. 381-382.

6. Баранчиков А.И., Баранчиков П.А. Практическая реализация дискреционно-ролевого доступа на чтение к записям БД // Вестник РГРТУ. №3 (выпуск 29). Рязань, 2009. С. 60-64.

7. Баранчиков П.А., Пылькин А.Н. Функциональная маскировка данных // Математическое и программное обеспечение вычислительнойых систем: Межвуз. сб. науч. тр. / под ред. Пылькина. М.:Горячая линия — Телеком, 2009. С. 137-144.

Баранчиков Павел Алексеевич

АЛГОРИТМЫ ОРГАНИЗАЦИИ И МОДЕЛИ ОГРАНИЧЕНИЯ ДОСТУПА К ОТДЕЛЬНЫМ ЗАПИСЯМ ТАБЛИЦ РЕЛЯЦИОННЫХ БАЗ ДАННЫХ

Автореферат
диссертации на соискание ученой степени
кандидата технических наук

Подписано в печать 16.11.2009. Формат бумаги 60× 80 1/16.
Бумага офисная. Печать трафаретная. Усл. печ. л. 1,25.
Уч.-изд. л. 1,25. Тираж 100 экз.

Редакционно-издательский центр
Рязанского государственного радиотехнического университета.
390005, г.Рязань, ул. Гагарина, 59/1.