$$MZ = \bigcup_{i=1}^{i1} MZ_i . \qquad (32)$$

Проведенная группировка и формализация необходимых для проката заготовок в виде множеств мерных слябов позволяет переходить к решению второго этапа задачи планирования – разработке плана работы листопрокатного цеха и установлению очерёдности выполнения отдельных позиций портфеля заказов.

Научная новизна и практическая значимость. Научная новизна работы заключается в том, что предлагаемая методика формализации переменных позволяет заменить изучение связей между характеристиками объекта изучением отношений между множествами и может использоваться на первом этапе моделирования любых аналогичных объектов управления со сложной структурой. Формализация объекта в виде теоретико-множественных конструкций является основой для представления базы данных процесса и разработки алгоритмов оптимального планирования проката, которые позволят сократить производственные затраты и обеспечить выполнение наибольшего количества заказов.

Список литературы: 1.*Конвей Р.В., Максвелл В.Л., Миллер Л.В.* Теория расписаний. М.: Наука, 1975. 360с.**2.***Клименко В.М.* Технология прокатного производства. К.: Высшая школа, 1989. 311с.**3.***Бахтинов В.Б.* Технология прокатного производства. М.: Металлургия, 1983. 483с.

Поступила в редколлегию 29.08.2007

Криводубський Олег Александрович, канд.техн.наук, доцент кафедры прикладной математики и информатики Донецкого национального технического университета. Адрес: Украина, 83000, Донецк, ул.Дзержинского, 7, тел.345-44-27.

Косилов Сергей Алексеевич, соискатель кафедры прикладной математики и информатики Донецкого национального технического университета. Адрес: Украина, 83000, Донецк, ул. Дзержинского, 7, тел.345-44-27.

УДК 004.056

А.В. ТРУБИЦЫН

ИЕРАРХИЧЕСКАЯ МОДЕЛЬ ДОСТУПА К РЕСУРСАМ В ПОСТРЕЛЯЦИОННОЙ СУБД CACHE

Предлагается иерархическая модель раздельного доступа к данным на уровне элементов базы данных. Обсуждаются вопросы построения иерархических связей и приводятся примеры реализации на основе деревьев (глобалов) в постреляционной СУБД Cache.

1. Введение

Целью исследования является разработка модели дифференцированного доступа к ресурсам в постреляционной СУБД Cache на уровне записей одной БД.

Актуальность обусловлена тем, что для обеспечения требуемой политики безопасности в постреляционной СУБД Cache необходимо разделять доступ на уровне элементов базы данных. Однако возможности разделять доступ к данным в одной БД в СУБД Cache отсутствует. Предлагается иерархическая модель доступа к ресурсам, которая эффективно решает задачу разделения доступа.

При разработке автоматизированной системы сбора, хранения, обработки и отображения объявлений различного рода на основе СУБД Cache возникла следующая технологическая задача. Каждый клиент, который вносит данные в БД, должен иметь возможность выполнять CRUD (Create, Retrieve, Update, Delete) запросы только к своим данным. У каждого клиента есть свой менеджер, а у менеджера может быть несколько клиентов. Менеджер работает с данными своих клиентов, а значит должен иметь права на эти данные. Группой менеджеров управляет начальник отдела, обладающий правом доступа ко всей информации, которой обладают его менеджеры, а значит и к данным клиентам своих менеджеров (рис.1). Эта структура отвечает реальным отношениям на коммерческих предприятиях. Существуют клиенты, которые могут автоматически регистрироваться в системе через Интернет. Такие клиенты должны автоматически встраиваться в иерар-

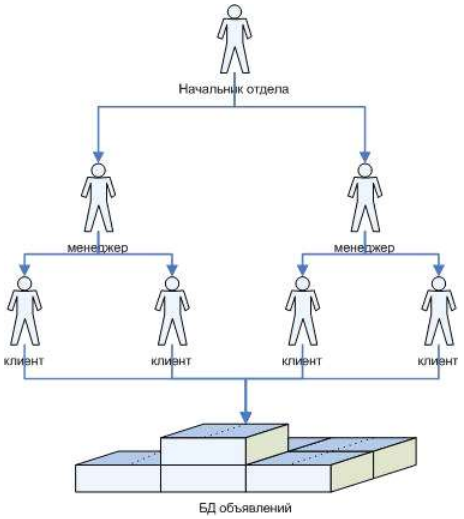

Рис.1. Иерархическая структура прав доступа к данным

хию менеджера. Выбор менеджера определяется его занятостью: новый зарегистрировавшийся клиент попадает к менеджеру, у которого меньше всего клиентов.

Описанная иерархическая структура бизнес-отношений среди персонала переносится и на возможность работы с данными в БД.

При реализации указанного механизма взаимодействия пользователей были поставлены такие требования: минимизация времени работы с подсистемой; простота в реализации; удобство хранения иерархии пользователей в БД.

СУБД Cache хранит данные в виде B*–деревьев (рис. 2). Дерево, которое имеет равное число уровней в каждом своём поддереве, называется сбалансированным (balanced tree). Характерным признаком B-дерева является минимизация среднего количества обращений к дисковым блокам при поиске требуемой записи данных. B-дерево, у которого каждый ключ указывает на блок данных, содержащий требуемую запись, называется B*-деревом. Оно делает возможным взаимнооднозначное соответствие между областью указателей и областью данных. B* - деревья состоят из следующих различных блоков: блока каталога на вершине, одного или нескольких блоков указателей, а также блоков данных, находящихся на нижнем уровне дерева и содержащих хранимую информацию [1,4].

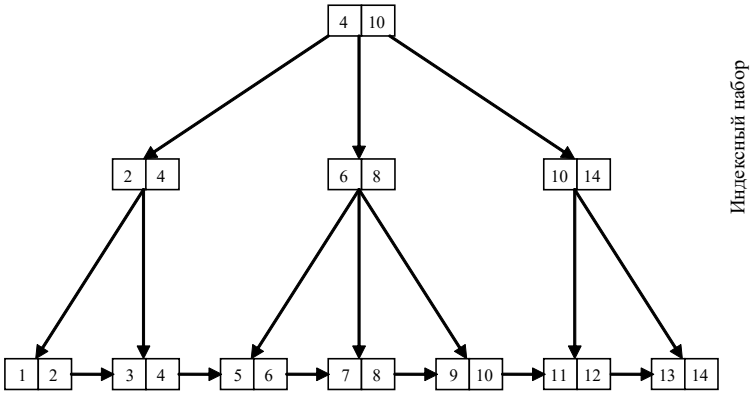

Рис. 2. Пример B* - дерева

Элементами оптимизации доступа через B*-деревья являются: сортировка нового элемента данных осуществляется в момент записи. B*-деревья сортируются постоянно и имеют высокую степень свободы реорганизации; блоки данных на нижнем уровне связаны друг с другом дополнительными указателями, так что последовательные операции с базой данных проводятся в оптимальном режиме, не покидая текущий уровень данных.

Пример бинарного дерева изображен на рис.2. Рассмотрим индексный набор. Верхняя запись содержит два значения 4 и 10. Следуя в левое поддерево, мы можем обратиться к записям, значения ключей которых меньше или равны 4; следуя в центральное поддерево, мы можем обратиться к записям, значения ключей которых больше 4 и меньше или равны 10; далее, следуя в правое поддерево, мы можем обратиться к записям, значения ключей которых больше 10.

2. Обсуждение модели

В настоящее время распространены следующие 2 подхода в управлении доступом к данным:

Дискреционное управление доступом (Discretionary access control, DAC) — разграничение доступа между пользователями и объектами данных в БД. Этот режим управления поддерживается стандартом SQL и реализован в большинстве современных СУБД.

Мандатное управление доступом (Mandatory Access Control, MAC) — это разграничение доступа пользователей к объектам данных, основанное на характеризуемой меткой конфиденциальности информации, которая содержится в объектах, и на официальном разрешении (допуске) субъектов обращаться к информации соответствующего уровня конфиденциальности [3]. Средства мандатной защиты предоставляются специальными СУБД и не реализованы в используемой СУБД Cache.

Стандартным решением разграничения доступа к данным на уровне файловой системы является следующий механизм (так разделяется доступ в операционных системах Linux, Windows), который легко реализуется на уровне БД. Пользователи разделяются на группы. Далее, каждой единице информации в БД соответствует специальный блок. Этот блок состоит из шести полей, которые имеют следующий смысл:

- первое поле – создатель имеет право читать содержимое;
- второе поле – создатель имеет право изменять содержимое;
- третье поле – группа имеет право читать содержимое;
- четвёртое поле – группа имеет право изменять содержимое;
- пятое поле – кто угодно имеет право читать содержимое;
- шестое поле – кто угодно имеет право изменять содержимое.

Этот механизм представляется не совсем удобным для реализации его в постреляционной БД Cache в виду хранения информации в деревьях и древовидной структуры отношений между пользователями.

Были сделаны **следующие предложения**:

1. Добавить к каждой единице информации специальный блок, который содержит ключ клиента, создавшего эту единицу информации.
2. Для каждого субъекта иерархии создать пользователя.
3. Иерархию пользователей хранить в виде дерева.
4. Для проверки доступа пользователя к данным использовать рекурсивные алгоритмы обхода дерева.

Суть политики безопасности данной задачи состоит в следующем: пользователь не может видеть данные вышестоящих по иерархии пользователей, но видит свои данные и данные всех своих подчиненных. Пользователь может изменять, удалять свои данные и данные всех подчиненных пользователей.

В СУБД Cache дерево называется глобалом. Группы пользователей заменяются на иерархическую структуру (см. рис.1). Для создания иерархии используется следующий синтаксис[1]:

```
S ^Hierarchy(“Менеджер_1”,”Клиент_1”) =”access”
S ^Hierarchy(“Менеджер_1”,”Клиент_2”) =”access”
S ^Hierarchy(“Менеджер_2”,”Клиент_3”) =”access”
S ^Hierarchy(“Менеджер_2”,”Клиент_4”) =”access”
S ^Hierarchy(“Начальник_Отдела_1”,”Менеджер_1”) =”access”
S ^Hierarchy(“Начальник_Отдела_1”,”Менеджер_2”) =”access”.
```

Пусть единицей информации в БД будет классифайд(объявление) “автомобиль”, тогда в Cache объявление со специальным блоком “owner”, регулирующим доступ, будет записано так:

```
S ^Adverts(1,”mark”)=”BMW”
S ^Adverts(1,”model”)=”512”
S ^Adverts(1,”year”)=”1998”
S ^Adverts(1,”price”)=”12000”
S ^Adverts(1,”owner”)=”Клиент_1”.
```

Такое объявление в соответствии с нашей иерархией могут просматривать следующие пользователи: “Клиент_1”, “Менеджер_1”, “Начальник_Отдела_1”.

В работе [5] обсуждалась подобная иерархическая модель, но моделировалась несколько другая бизнес-логика и реализовывалась в реляционной БД, а не в постреляционной БД.

3.Сравнение классической и иерархической модели

В классической модели при создании пользователя необходимо выполнить следующую последовательность из трёх действий:

– добавить пользователя в систему;

– добавить группу;

– в группу внести всех пользователей, которые имеют право просматривать данные добавленного пользователя.

Заметим, что нам придется создать группу для каждого пользователя, возможно, за исключением пользователей, которые находятся в вершине иерархического дерева. Например, при создании клиента, который находится на m-м уровне иерархического дерева, нам необходимо выполнить m действий по внесению в группу клиента того пользователя, который будет просматривать данные.

При регистрации пользователя в системе через Интернет или руками администратора пользователь автоматически или в ручном режиме назначается некоторому менеджеру. Далее в группу пользователя добавляются все пользователи группы менеджера.

При создании пользователя в иерархической модели необходимо выполнить следующие действия:

– добавить пользователя в систему;

– вставить в дерево иерархии.

Очевидно, что иерархическое дерево соответствует внутренней структуре хранения данных. Оно минимизирует количество операций, реализуется на понятном пользователю языке и обеспечивает необходимую безопасность данных. Как одно из следствий предложенной организации доступа к данным, администратором системы разделения доступа может быть любой сотрудник без специальной подготовки.

4.Выводы

В СУБД Cache возможно реализовать детализированный доступ к данным. Совместное использование детализированного и дискреционного доступа обеспечивает гибкость управления безопасностью в БД. Политика доступа может быть построена согласно любым правилам корпорации и учитывать сложные бизнес-правила.

Данный подход не слишком сильно усложняет структуру БД, но требует специальной подготовки администратора СУБД. Метод протестирован на СУБД Cache и использован при создании системы безопасности в автоматизированной системе управления объявлениями и рекламой для журналов “Мир Квартир”, “Автомагазин” (http://www.avtomagazine.ua) от фирмы AutoSoft.

Список литературы: 1. *Труб И. И.* СУБД Cache. Работа с объектами. М.: “Диалог-МИФИ”, 2006. 480 с. **2.** *Левитин Ананий В.* Алгоритмы: введение в разработку и анализ. М. “Вильямс”, 2006. 339 с. **3.** *Вирт Н.* Алгоритмы и структуры данных / Пер. с англ. 2-е изд. СПб: “Невский Диалект”, 2005. 352с. **4.** *Кормен Томас Х, Лейзерсон Чарльз И., Ривест Рональд Л., Штайн Клиффорд.* Алгоритмы: построение и анализ. М.: “Вильямс”, 2006. 1296с. **5.** *Петухова Н.* Метод обеспечения доступа к данным реляционных систем на уровне строк отношения // Transport and Telecommunication. Vol.4, N 1. 2003 , Рига, Латвия. С. 45–52. **6.** *Григорьев Ю.А., Ревунков Г.И.* Банки данных. М.: МГТУ им. Н. Э. Баумана, 2002. 320с. **7.** http://www.citforum.ru/database/articles/b-tree_indexes/.

Поступила в редколлегию 23.05.2007

Трубицын Андрей Владимирович, преподаватель ХНУРЭ. Научные интересы: проектирование программных продуктов; шаблоны проектирования и рефакторинг; функциональное программирование; постреляционные СУБД. Адрес: Украина, 61112, Харьков, пр. 50 лет ВЛКСМ, 96/153 к. 203, тел. 62-50-12