УДК 681.3

П.А. Баранчиков

КОНФЛИКТ КЛЮЧЕЙ ОТНОШЕНИЯ И МЕТОДЫ ЕГО РЕШЕНИЯ

Производится постановка проблемы возникновения конфликта ключей отношения при проектировании защищенных баз данных. Рассматриваются варианты решения конфликта в зависимости от предметной области и требований к разграничению доступа между пользователями. Приведены примеры реализации таких решений и краткое описание их достоинств и недостатков.

Ключевые слова: *базы данных, ограничение доступа, ключи отношения*

Введение. С развитием БД возникла необходимость разграничивать доступ к отдельным записям таблиц БД. Реализация такого доступа может быть весьма разнообразной [1], однако существует ряд общих проблем, возникающих при таком распределении доступа.

Постановка проблемы. Пусть существует БД с распределением доступа к отдельным записям таблиц. Обозначим совокупность записей отношения r, к которым имеет доступ пользователь U_1 на чтение, как ro_{u1}, а совокупность записей, к которым он имеет доступ на запись/чтение, – rw_{u1}. Очевидно, что

$$rw_u \subseteq ro_u. \quad (1)$$

В дальнейшем, будем рассматривать лишь множество записей r_u, к которому пользователь имеет доступ, без уточнения типа доступа, так как при разделении доступа на чтение и запись все преобразования производятся полностью аналогично.

Пусть в r существует ключ K. Пусть k_k – определенный 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:

$$\begin{gathered}\exists r_j(k_k) \\ r_i(k_k) \notin r_u.\end{gathered} \quad (2)$$

То есть в отношении r существует запись $r_i(k_k)$, идентифицируемая значением ключа k_k, но при этом пользователь U не имеет доступа к этой записи. Следовательно, он не знает, что такая запись в отношении существует и может произвести попытку добавления другой записи r_j в отношение r, такой что ее значения ключевых атрибутов совпадут со значениями записи r_i. После добавления новой записи пользователь имеет к ней доступ.

$$r_i(k_k) \notin r_u \quad (3)$$

Одновременно с этим:

$$\begin{gathered}r_j(k_k) \in r_u \\ \Rightarrow r_j \neq r_i.\end{gathered} \quad (4)$$

Здесь доказано, что r_i и r_j не совпадают по принадлежности к подмножествам записей, видимых пользователю r_u. Однако существует возможность того, что r_i и r_j имеют различные значения атрибутов и совпадают у них только значения ключевых атрибутов.

Иначе говоря, получается, что пользователь добавляет новую запись, отличающуюся от всех имеющихся записей в отношении, но ключ этой новой записи совпадает со значением ключа одной из уже существующих в таблице записей. Назовем такую ситуацию конфликтом ключей отношения (ККО).

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

Из определения ключа следует, что если у двух записей существует одно и то же значение ключа, то это одна и та же запись, однако в рассмотренной ситуации идентичность этих записей может быть поставлена под сомнение.

Также проблема конфликта ключей отношения осложняется тем, что при реализации ограничений целостности на уникальных ключах любая СУБД, поддерживающая уникальные ключи, выдаст ошибку неуникальности ключа при добавлении записи r_j.

Предложим методы, позволяющие избежать конфликт ключей отношения. Рассмотрим их поочередно.

Разбиение на пересекающиеся подмножества. Пусть все множества записей r_{Ui}, к которым имеют доступ пользователи U_i не

пересекаются, а их полное объединение дает полное отношение r:

$$r_{u_1} \cap r_{u_2} \cap \ldots \cap r_{u_n} = \varnothing$$
$$r_{u_1} \cup r_{u_2} \cup \ldots \cup r_{u_n} = r. \quad (5)$$

Таким образом, каждая запись принадлежит только одному из подмножеств r_u [2]. В такой ситуации возможно в ключ K отношения r ввести дополнительный атрибут C, определяющий, к какому множеству r_u принадлежит запись. Тогда схема отношения примет вид:

$$R = \{A, B, \ldots\}$$
$$R' = \{A, B, \ldots, C\}. \quad (6)$$

В таком случае множество записей, которые «видны» пользователю U_i, можно описать выборкой из видоизмененного отношения r' [1]:

$$r_{u_i} = \sigma_{C=C_i}(r'). \quad (7)$$

Для обобщения ситуации введем функцию $f(U, C)$, определяющую принадлежность записи множеству r_{Ui} в зависимости от пользователя U и специального атрибута C. Тогда в одном из вариантов функции будем иметь:

$$f(C) = \begin{cases} true, если C = C_i, \\ false, если C \neq C_i \end{cases} \quad (8)$$
$$r_{u_i} = \sigma_{f(U_i, C)}(r').$$

Очевидно, для реализации такого типа решения конфликта хорошо подходят представления пользователя, осуществляющие выборку с проверкой новых записей на принадлежность к множеству rw, которая в данном случае однозначно определяется выражением $C=C_i$.

Это решение хорошо для использования, если данные, введенные различными пользователями, имеют различный физический смысл. Например, если в отношении хранится информация о проведенных актах ревизии (см. таблицу 1), то различные подмножества записей будут соответствовать актам ревизий трубопроводов, актам ревизий аппаратов и сосудов и т.д. Правомерность такого разбиения на подмножества определяется тем, что инженеры одного сектора имеет доступ или к данным, относящимся к трубопроводам, или к данным, относящимся к аппаратам и сосудам.

Таблица 1

Код ревизии	Объект ревизии	Код сектора	Дата
1	Насос Н-1	1	05/08/2007
2	Насос Н-10	1	06/09/2007
3	Холодильник Х-3	2	07/05/2007
4	Теплообменник Т-5	2	09/07/2007
5	Сепаратор С-10	2	25/02/2008

Как видно из приведенного примера, все ревизии можно разделить на не пересекающиеся подмножества: ревизии машинного оборудования и ревизии аппаратов и сосудов. Особое свойство таких таблиц заключается в том, что инженерам нет производственной необходимости видеть и редактировать данные, введенные инженерами другого направления. Здесь функция принадлежности записи к подмножеству, доступному пользователю, определяется, как и было указано в формуле (1). В роли атрибута С будет выступать «Код сектора». Выборки записей для работников секторов надзора будут соответственно выглядеть так:

$$r_{маш.оборудование} = \sigma_{кодсектора=1}(r')$$
$$r_{аппаратысосуды} = \sigma_{кодсектора=2}(r') \quad (9)$$

Этот метод решения ККО довольно прост, однако сфера его применения не охватывает всех потенциальных задач, поставленных при проектировании схемы БД.

Блокировка изменений. При разрешении ККО самым простым методом является блокирование добавления записи, если такая запись уже есть. Если же записи нет, то система должна позволить (если это дополнительно не оговорено в ТЗ) пользователю добавить запись и предоставить далее пользователю доступ к изменению этой записи.

Пусть

$$r_i \notin r_u$$
$$r_j \in r_u. \quad (10)$$

В случае обнаружения ККО система (как правило, ограничение целостности в СУБД) запрещает добавление записи, инициированное пользователем. При использовании этого метода следует учитывать то, что если один пользователь ввел какие-либо данные, другой не

сможет их исправить и/или удалить. Как упомянуто выше, к добавленной пользователем записи должен быть предоставлен доступ пользователю, ее добавившему. Следовательно, должен существовать определенный алгоритм определения служебных атрибутов, отвечающих за доступ к записи.

Например, пусть доступ к записи определяется мандатным методом. В таком случае в отношение *r* добавим служебный атрибут *C*:

$$R = A_1 A_2 ... A_n$$
$$R' = A_1 A_2 ... A_n C. \quad (11)$$

Функция предоставления доступа к записи будет иметь вид:

$$f(C) = \begin{cases} true, если C \geq C_u, \\ false, если C < C_u. \end{cases} \quad (12)$$

Пусть в отделе кадров есть сотрудники, наделенные различными полномочиями при редактировании данных по сотрудникам предприятия. Тогда C_u будет определяться в зависимости от должности работника кадров. В качестве служебного атрибута *C*, введенного в (6), добавим в отношение *Сотрудники* атрибут *Доступ*, который будет определять уровень доступа:

Таблица 2

Код сотрудника	Должность	ФИО	Доступ
1	Ген. Директор	Иванов Иван Иванович	1
2	Гл. бухгалтер	Петров Петр Петрович	1
3	Фрезеровщик	Сидоров Сидор Сидорович	2
4	Грузчик	Николаев Николай Николаевич	2

Как видно из таблицы 2, рядовой сотрудник отдела кадров, имеющий уровень доступа 2, не имеет прав на просмотр и изменение данных по руководству предприятия. Начальник отдела кадров должен иметь уровень доступа 1, чтобы ему был предоставлен доступ к данным по всем работникам предприятия.

При добавлении нового сотрудника рядовым работником отдела кадров записи, соответствующей новому сотруднику, должен быть присвоен уровень доступа 2, чтобы этот работник мог видеть и редактировать занесенную им же запись. При добавлении сотрудника начальником отдела кадров новой записи таблицы должен быть присвоен уровень доступа 1, чтобы более никто не мог иметь доступа к этой записи.

Если рядовой работник отдела кадров введет информацию по генеральному директору Иванову Ивану Ивановичу, ему будет отказано в доступе, так как такая запись в таблице уже есть. Следует дополнительно отметить, что в приведенном примере можно добавить еще одного генерального директора с кодом 5 в отношение *Работники*. Для того чтобы упомянутый отказ в доступе произошел, необходимо ввести альтернативный ключ, например номер паспорта работника.

Изменение доступа к записи. Возможен вариант «разрешения» пользователю, добавляющему конфликтующую запись, просматривать информацию, введенную другим пользователем. Такое «разрешение» можно допустить на основании того, что пользователь, спровоцировавший ККО, имеет все данные о значениях ключевых атрибутов записи.

Для реализации такого подхода можно ввести служебное отношение *s* со схемой [3]:

$$R = \{KABC...\},$$
$$S = \{KU\}. \quad (13)$$

Отношение *s* показывает, какие пользователи имеют доступ к соответствующим записям отношения *r*. Значение атрибута *U* идентифицирует пользователя, которому предоставлен доступ к записи, с ключом, совпадающим со значением ключа *K*. В таком случае возможно и разделение доступа на чтение/запись:

$$R = \{KABC...\},$$
$$S = \{KUW\}. \quad (14)$$

В этом варианте отношение *s* расширено атрибутом *W*, указывающим, имеет ли пользователь *U* к записи *K* доступ на запись или только на чтение. Выборка записей, доступных пользователю, будет в таком случае определена как:

$$rw_u = \sigma_{F(U,W)}(r \triangleright\triangleleft s). \quad (15)$$

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

При «разрешении» доступа пользователя к данным также возможно выполнить некоторую дополнительную проверку. Так, например,

необходимо выбрать ту часть отношения, которую пользователь должен ввести в отношение для того, чтобы система предоставила этому пользователю доступ к добавляемой записи. В простейшем случае это будет ключ отношения. Но в более общем случае в этот набор атрибутов могут входить не только ключи.

Например, пусть отношение хранит информацию о клиентах: ФИО, номера паспортов и какие-то дополнительные рабочие данные. Можно реализовать предоставление доступа при правильном указании пользователем не только номера паспорта, но также даты выдачи паспорта и ФИО клиента. В случае совпадения всех этих данных можно заключить, что пользователь не просто перебирает все значения номеров паспортов (например, автоматически — атакуя систему «в лоб»), а владеет реальными данными по вводимому клиенту. Такие проверки можно осуществлять в триггерах на добавление записей в таблицы или хранимых процедурах.

Как видно из приведенных формул, реализация изменения доступа к записи осложняется вводом дополнительного отношения *s* в схему базы данных. Если же просто запретить пользователю редактировать записи с существующими ключами, создание этого отношения не обязательно. Если не производить дополнительных изменений в схему БД, СУБД просто запретит добавление записей, сославшись на дублирование ключа.

Следует отметить, что если не выдавать пользователю ошибки, то есть попытаться «скрыть» невозможность добавления записи, будет существовать возможность проверить успешность добавления косвенными методами, например, запросив выборку данных с сервера БД из таблицы/представления.

Заключение. Предложены подходы к решению проблемы возникновения конфликтов ключей отношения. Предложенные методы приемлемы для использования в промышленных БД, так как не противоречат основам теории реляционных баз данных и могут быть реализованы на подавляющем большинстве реляционных и объектно-реляционных СУБД. Методы могут применяться в различных ситуациях, в зависимости от требований к ограничению доступа пользователей и особенностей предметной области.

Библиографический список

1. *Баранчиков А.И., Баранчиков П.А.* Организация доступа к записям таблиц в базах данных ISBN 978-5-7722-0272-2/ Вестник РГРТА. Рязань, 2007 г. Вып.20.

2. *Мейер Д.* Теория реляционных баз данных. — М.:Мир, 1987. 608 с.

3. *Beeri, C., and Bernstein, P.A.* Computaional Problems Related to the Design of Normal Form Relational Schemas. ACM TODS 4»:1, March 1979, 30-59.