УДК 681.324

РАСШИРЕНИЯ РЕЛЯЦИОННОЙ МОДЕЛИ ДЛЯ ОБЕСПЕЧЕНИЯ БЕЗОПАСНОСТИ В БАЗАХ ДАННЫХ

А. В. Фомин,
аспирант
А. В. Бржезовский,
канд. техн. наук, доцент
Санкт-Петербургский государственный университет аэрокосмического приборостроения (СПбГУАП)

В статье предлагаются расширения реляционной модели для обеспечения безопасности при доступе к данным на основе мандатной модели безопасности Белла и ЛаПадула. В отличие от существующих моделей предлагаемая модель позволяет обеспечить ссылочную целостность в базах данных «стандартным» образом. Это достигается за счет введения свойства «Разделение», которое заключается в том, что исходная схема отношения реляционной модели должна быть разделена на две: ключевую схему отношений R', которая содержит уникальные первичные ключи, и схему данных R'' — собственно данные исходного отношения. Такой подход позволяет исключить дублирование ключей при явлении многозначной реализации.

Relational model extensions for database security are suggested. These models in contrast to existing models allow supplying with reference integrity by standard manner. This arrives by division original ralation scheme on two bounded scheme — «key scheme» and «data scheme». In key scheme there are apparent key of original relation. When polyinstantiation occurs in original relation scheme, in key scheme unique apparent keys keep. This allow to provide reference integrity on primary key of key scheme. For suggested concept entity integrity, null integrity, interinstance integrity, polyinstantiation integrity are proposed.

Введение

Одним из основных требований международных стандартов безопасности [1, 2] для информационных систем является реализация политики безопасности на основе формальных моделей безопасности. Особенно это требование важно для баз данных, которые являются хранилищем информации, в том числе конфиденциальной. Большинство современных систем управления базами данных позволяют поддерживать только дискреционную [3] и ролевую политики безопасности (ролевую модель можно рассматривать как подвид дискреционного доступа). Дискреционный доступ основан на предоставлении субъектам системы полномочий на выполнение операций над объектами системы (чтение, запись, модификация, удаление). Типичная реализация дискреционного контроля — списки прав доступа. При этом для каждого объекта системы (например, файла) указывается перечень субъектов, которые имеют (или не имеют) право на выполнение определенных операций над этим объектом. Но дискреционный контроль не обеспечивает гарантий по защите данных от несанкционированного доступа, поскольку не контролирует потоки информации. В частности он не решает проблему «троянского коня». Эти недостатки отсутствуют при организации мандатного управления доступом [4]. Поэтому для систем высокого класса защищенности требуется поддержка мандатной политики безопасности.

Основой мандатных моделей безопасности является модель Белла—ЛаПадула [4]. Система в модели Белла—ЛаПадула представляется в виде множества субъектов S, объектов O, $S \subseteq O$, прав доступа read и write. С точки зрения теории реляционных баз данных [5], под объектом может пониматься отношение, атрибут отношения, кортеж или даже отдельный элемент данных в рамках кортежа. Субъект представляет собой некий процесс, действующий от имени пользователя, желающего получить доступ к данным. Уровни безопасности задаются функцией F: $O \rightarrow L$, определяющей для каждого объекта и субъекта уровень безопасности из

множества уровней $L = \{c_1, c_2, \ldots, c_n\}$, разбивая множество сущностей системы на классы, в пределах которых их свойства, с точки зрения модели безопасности, являются эквивалентными. На множестве уровней безопасности определено частичное нестрогое отношение порядка для элементов этого множества, задаваемое оператором доминирования — «≤». Запись «$c_1 \le c_2$» обозначает, что уровень c_2 доминирует над уровнем c_1. Множество L в таком случае еще называют решеткой уровней безопасности. Контроль доступа осуществляется в зависимости от уровней безопасности взаимодействующих сторон на основании двух правил:

1) уполномоченное лицо (субъект) имеет право читать («read») только те документы, уровень безопасности которых не превышает его собственный;

2) уполномоченное лицо (субъект) имеет право заносить информацию («write») только в те документы, уровень безопасности которых не ниже его собственного.

Первое правило обеспечивает защиту информации, обрабатываемой более доверенными лицами, от доступа со стороны менее доверенных. Второе правило предотвращает утечку информации со стороны высокоуровневых участников процесса обработки информации к низкоуровневым.

В классической реляционной теории отсутствуют механизмы, позволяющие представить мандатное управление доступом к данным. Именно на поддержание этого вида доступа направлены описываемые ниже расширения.

Основные положения и существующие подходы

Реляционная модель данных была предложена Коддом в 1970 г. Согласно этой модели [6], данные хранятся в таблицах, называемых отношениями (R). Каждое отношение содержит одну или несколько колонок, называемых атрибутами. В некоторый момент времени отношение содержит в себе множество строк (которое может быть и пустым). Каждая строка называется кортежем. Количество кортежей в отношении меняется с течением времени в результате выполнения операций манипулирования данными, предусмотренных в реляционной алгебре. Набор атрибутов, однозначно идентифицирующих кортеж, называется возможным ключом отношения. Из всех возможных ключей для каждого отношения выбирается только один, который называется первичным ключом. В качестве примера рассмотрим отношение «Проекты» (табл. 1).

Это отношение состоит из трех атрибутов: «Код», «Наименование», «Описание». В отношении определены три кортежа, описывающие проекты некоторой строительной фирмы. Первичным ключом для данного отношения является атрибут «Код». В соответствии с мандатной политикой все данные должны иметь метку, определяющую уровень безопасности данных. Тогда будет возможно реализовать доступ к этим данным пользователей, имеющих определенный уровень безопасности. Например, пользователи, имеющие класс доступа «конфиденциально», могут «увидеть» данные с меткой уровня безопасности «конфиденциально», или ниже («общедоступно»). В связи с этим возникает необходимость в расширении стандартной реляционной модели метками безопасности (в дальнейшем такую модель будем называть расширенной реляционной моделью). Метки безопасности могут быть на разных уровнях детализации — от самих отношений до конкретных элементов данных в отдельном кортеже отношения. Рассмотрим уровень наибольшей детализации, когда доступ осуществляется к конкретным элементам данных (поэлементная классификация). В случае расширенной реляционной модели отношение, представленное в табл. 1, будет выглядеть так, как показано в табл. 2.

■ ***Таблица 1.*** *Отношение «Проекты»*

Код	Наименование	Описание
BZM00	Прометей	Строительство казарм
ТП18	Луна	Жилой дом
К678	Роза	Химический завод

■ ***Таблица 2.*** *Отношение «Проекты» с метками безопасности*

Код		Наименование		Описание	
BZM00	О	Прометей	К	Строительство казарм	К
ТП18	О	Луна	О	Жилой дом	О
К678	К	Роза	СК	Химический завод	СК

В табл. 2 для каждого кортежа определена метка безопасности, которая задает уровень безопасности данных на решетке уровней безопасности. В нашем примере решетка состоит из трех элементов, между которыми существует следующее отношение: О ≤ К ≤ СК. Собственно «О» — «общедоступно»; «К» — «конфиденциально»; «СК» — «строго конфиденциально». Таким образом, «О» — наименее конфиденциальная информация, «СК» — наиболее.

В соответствии с мандатной политикой доступа пользователи, имеющие различные полномочия, должны получить различное «представление» отношения. Например, пользователь, имеющий уровень полномочий «О», должен «увидеть» отношение из табл. 2, «отфильтрованное» в соответствии с его уровнем привилегий (табл. 3). Пользователь с полномочиями на доступ к элементам класса «К» увидит отношение, представленное в табл. 2 так, как это представлено в табл. 4. Пользователь с наивысшими полномочиями увидит все кортежи отношения (табл. 2).

■ ***Таблица 3.*** *Представление для пользователя с полномочиями «О»*

Код		Наименование		Описание	
BZM00	О	Null	О	Null	О
ТП18	О	Луна	О	Жилой дом	О

■ ***Таблица 4.*** *Представление для пользователя с полномочиями «К»*

Код		Наименование		Описание	
BZM00	О	Прометей	К	Строительство казарм	К
ТП18	О	Луна	О	Жилой дом	О
К678	К	Null	К	Null	К

Предположим, что пользователь с полномочиями «О» хочет вставить в отношение «Проекты» кортеж («К678», «Сакура», «Ресторан»). С точки зрения СУБД, реализующей реляционную модель, данная операция должна быть запрещена, так как кортеж с первичным ключом «К678» уже существует в отношении. Но, с точки зрения политики безопасности, отменить эту операцию нельзя, так как в противном случае возникает нарушение второго правила мандатной политики, когда уполномоченное лицо не может передавать информацию на уровень ниже его собственного. А при отмене вставки кортежа как раз и произойдет передача информации с более высокого уровня на более низкий о том, что кортеж с ключом «К678» уже существует и что информация о нем является конфиденциальной. Стандартным подходом в разрешении этой ситуации является допущение существования обоих кортежей с одинаковыми ключами в отношении. Это явление получило название «polyinstantiation» [7] («многозначность»). В этом случае, с точки зрения пользователя с полномочиями «СК», отношение должно выглядеть так, как представлено в табл. 5.

Следует отметить, что такая же ситуация возникнет и в случае модификации какого-либо отдельного элемента данных. Например, предположим, что пользователь с полномочиями «О» хочет модифицировать кортеж с первичным ключом «BZM00» с указанием наименования проекта — «Волна» и описанием — «Пристань». Требования политики безопасности повлекут создание нового кортежа, причем так, как показано в табл. 6.

Очевидно, что такой подход обладает целым рядом недостатков. Во-первых, нарушается основной принцип уникальности первичного ключа в отношении. Во-вторых, невозможно организовать ссылочную целостность, поскольку отсутствует уникальный первичный ключ в таблице, на которую необходимо ссылаться. Кроме того, существует проблема выборки данных с соединением кортежей. Рассмотрим, например, отношение «Сотрудники», которое состоит из атрибутов «Номер паспорта», «Фамилия» и «Проект» (табл. 7). Последний атрибут является «внешним ключом» на отношение «Проекты».

■ ***Таблица 5.*** *Иллюстрация явления «многозначности»*

Код		Наименование		Описание	
BZM00	О	Прометей	К	Строительство казарм	К
ТП18	О	Луна	О	Жилой дом	О
К678	К	Роза	СК	Химический завод	СК
К678	О	Сакура	О	Ресторан	О

■ ***Таблица 6.*** *Иллюстрация явления «многозначности»*

Код		Наименование		Описание	
BZM00	О	Волна	О	Пристань	О
BZM00	О	Прометей	К	Строительство казарм	К
ТП18	О	Луна	О	Жилой дом	О

Допустим, что мы можем построить запрос на объединение двух отношений (для упрощения рассматривается запрос на языке SQL):

```
Select * from Сотрудники, Проекты
Where Сотрудники.Проект = Проекты.Код
```

Условие «Сотрудники.Проект = Проекты.Код» включает также равенство соответствующих меток уровня безопасности. Это необходимо для получения корректного результата (иначе мы получим дублирование информации о филиалах). Результатом такого запроса будет только одна строчка, по которой сотрудник «Павлов» задействован в проекте «Сакура», а закрытая информация о том, что «Павлов» участвует в проекте «Роза», окажется потерянной даже для пользователей с уровнем привилегий «К» и «СК», имеющих полномочия на доступ к таким данным. По этой причине системы, построенные по принципу многозначности, производят соединение отношений не на уровне СУБД, а возлагают эту операцию на прикладную программу, которая выбирает данные из отдельных таблиц и обрабатывает их для получения корректных результатов [8].

В связи с этим представляется актуальной проблема создания таких расширений реляционной модели, которые бы позволили организовать ссылочную целостность, обеспечить корректную выборку данных с объединением таблиц и минимизировать потери от явления «polyinstantiation». Рассмотрим ее подробнее.

Расширения реляционной модели для обеспечения безопасности баз данных с организацией ссылочной целостности

Ссылочную целостность можно реализовать при условии существования уникального первичного ключа в отношении, на которое производится ссылка из другого отношения. Но, как уже упоминалось, при наличии явления множественности ключей это невозможно. Единственно возможным первичным ключом в таком отношении могла бы быть совокупность всех атрибутов, образующих «видимый» первичный ключ, и всех меток безопасности всех атрибутов отношения. Но на такой ключ ссылаться нет смысла, поскольку в противном случае мы получим

■ ***Таблица 7.*** *Отношения «Сотрудники»*

Номер паспорта		Фамилия		Проект	
1111111111	О	Павлов	О	К678	О

проблемы типа «отказа в обслуживании», когда изменить какую-либо запись (относительно атрибутов, не входящих в «видимый» первичный ключ) станет невозможным для разноуровневых пользователей по причине нарушения ссылочной целостности. Но и отказаться от явления «polyinstantiation» полностью также невозможно, поскольку это приводит обычно к нарушению политики безопасности, всевозможным ограничениям на модификацию данных или к блокировке работы системы при определенных условиях [9–11].

Ниже предлагается модель, позволяющая организовать ссылочную целостность при существовании многозначности.

Определение 1. Назовем исходной реляционной схемой отношения

$$R(A_1, C_1, A_2, C_2, \ldots, A_m, C_m, A_{m+1}, C_{m+1}, \ldots, A_n, C_n, TC),$$

где A_i — атрибут отношения, принимающий значения из домена $D_i = \mathrm{dom}(A_i)$; атрибуты $A_1, A_2, \ldots, A_m$ входят в «видимый» первичный ключ отношения со схемой R (под «видимым» первичным ключом понимается ключ, определенный пользователем); C_i — метка уровня безопасности атрибута A_i; TC — метка уровня безопасности кортежа; C_i, $TC \in L$ (L — множество уровней безопасности), $TC = C_{m+1} \bullet C_{m+2} \bullet \ldots \bullet C_n$ (оператор «•» обозначает наименьшую верхнюю границу для пары элементов) [4].

Определение 2. Внешним ключом в отношении $R1$, ссылающемся на отношение $R2$ (обозначим FK_{R2}), называется совокупность метки безопасности внешнего ключа (обозначим $C_{FK_{R2}}$), набора атрибутов отношения $R1$ и их меток безопасности, являющихся точной копией набора атрибутов и меток безопасности первичного ключа отношения $R2$:

$$FK_{R2} = C_{FK_{R2}} \cup \{A_{R1,i}, C_{R1,i}\};$$

$$A_{R1,i} \in FK_{R2}, C_{R1,i} \in FK_{R2} \Rightarrow A_{R1,i} \in R1, C_{R1,i} \in R1,$$

$$\exists A_{R2,j} \in R2, C_{R2,j} \in R2:$$

$$A_{R1,i} = A_{R2,j}, C_{R1,i} = C_{R2,j}, A_{R2,j} \in AK_{R2}.$$

Здесь AK_{R2} — «видимый» первичный ключ отношения $R2$ (определение см. ниже). Отношение может иметь несколько внешних ключей, каждый из которых имеет свои метки безопасности и свою совокупность атрибутов отношения; при этом $FK_{R2} \cap FK_{R3} \cap \ldots \cap FK_{Rn} = \varnothing$.

Определим ключевое свойство модели, которое позволяет организовать ссылочную целостность в базах данных, где поддерживается мандатная политика безопасности (в многоуровневых базах данных).

Свойство 1. Разделение. Для организации ссылочной целостности в многоуровневой реляционной модели исходная схема отношения R должна быть разделена на две схемы — R' и R'':

$$R'(A_1, C_1, A_2, C_2, \ldots, A_m, C_m);$$

$$R''(A_{m+1}, C_{m+1}, \ldots, A_n, C_n, FK_{R'}, TC).$$

Здесь $FK_{R'} = C_{FK_{R'}} \cup A_1 \cup C_1 \cup A_2 \cup C_2 \cup \ldots \cup A_m \cup \cup C_m$ — внешний ключ из отношения R'' на отношение R'. $C_{FK_{R'}}$ — метка безопасности внешнего ключа.

Если исходная схема отношения R содержит только атрибуты «первичного» ключа и их метки безопасности, то разделение производить не надо.

Таким образом, в отношении R' мы получим только определенные пользователем атрибуты первичного ключа отношения R и их метки безопасности, а в отношении R'' мы получим «информационное наполнение» отношения R и ссылку на соответствующий кортеж в отношении R'. Мы не можем представить в отношении R' только атрибуты «видимого» ключа без меток безопасности, так как в противном случае мы не могли бы скрыть конфиденциальные ключи. Но использование в отношении R' в качестве первичного ключа совокупности всех атрибутов и их меток безопасности позволит нам организовать ссылочную целостность, поскольку гарантируется уникальность такого ключа.

Рассмотрим пример. Представим отношения, показанные в табл. 6 и 7 в соответствии с предложенной концепцией.

Как видим, в отношении «Проекты′» (табл. 8) мы имеем уникальные ключи, что позволяет организовать нам ссылочную целостность на эти ключи из отношения «Сотрудники″» (табл. 11). В отношении «Сотрудникии″» внешним ключом является атрибут «Проект», который состоит из трех элементов. Первые два образуют собственно ссылку на ключ отношения «Проекты′». Третий представляет собой метку безопасности самой ссылки и необходим для корректного объединения при выборке данных из двух таблиц. Обозначим этот элемент $C_{FK_{\text{Проекты}'}}$. Для этого примера корректным выражением для выборки данных на языке SQL является следующее:

```
Select * from Сотрудники″, Проекты″
Where Сотрудники″.Проект = Проекты″.Код
and Сотрудники″. C_FK_Проекты′ = Проекты″.TC
```

■ ***Таблица 8.*** *Проекты′*

Код	
BZM00	O
ТП18	O

■ ***Таблица 9.*** *Проекты″*

Код		Наименование		Описание		TC
BZM00	O	Волна	O	Пристань	O	O
BZM00	O	Прометей	K	Строительство казарм	K	K
ТП18	O	Луна	O	Жилой дом	O	O

■ ***Таблица 10.*** *Сотрудники′*

Номер паспорта	
1111111111	O

■ ***Таблица 11.*** *Сотрудникии″*

Номер паспорта		Фамилия		Проект			TC
1111111111	O	Павлов	O	BZM00	O	K	K

В том случае, если бы ссылку создавал пользователь с полномочиями «О», мы получили бы в результате выполнения запроса объединение данных о сотруднике с несекретной информацией, по которой сотрудник Павлов участвует в проекте «Волна» по строительству пристани. В данном примере ссылку создавал пользователь с полномочиями «К». В результате выполнения запроса он получит конфиденциальную информацию о том, что Павлов работает над проектом «Прометей» и строит казармы. А пользователь с полномочиями «О» даже не «увидит» эту ссылку.

Определение 3. Экземпляры отношения.

1. Пусть r'' — отношение со схемой R''. Для каждого отношения r'' существует коллекция экземпляров отношения — по одному для каждого уровня безопасности $c \in L$. Каждый экземпляр r''_c отношения r'' представляет собой множество различных кортежей в форме $(a_1, c_1, a_2, c_2, \ldots, a_n, c_n, tc)$, где $a_i \in D_i$ или $a_i = \text{null}$; $a_i \neq \text{null} \Rightarrow c_i \leq c$, $i = 1, \ldots, n$; $c_1, c_2, \ldots, c_n \in L$; $tc = c_1 \bullet c_2 \bullet \ldots \bullet c_n$.

Метка уровня безопасности любого атрибута кортежа не может быть равной null, даже если атрибут $a_i = \text{null}$. Каждый экземпляр отношения служит для представления реальной сущности на некотором уровне безопасности c.

2. Пусть r' — отношение со схемой R'. Для каждого отношения r' существует коллекция экземпляров отношения — по одному для каждого уровня $c \in L$. Каждый экземпляр r'_c отношения r' представляет собой множество различных кортежей в форме $(a_1, c_1, a_2, c_2, \ldots, a_n, c_n)$, где $a_i \in D_i$, $c_i \leq c$. Каждый экземпляр r'_c служит для представления набора видимых первичных ключей на некотором уровне безопасности c.

Свойство 2. Видимые первичные ключи. Для схемы отношения R' «видимым» первичным ключом $AK_{R'}$ будет совокупность всех атрибутов. При этом для всех экземпляров r'_c отношения r' и кортежа $t \in r'_c$ *должны выполняться следующие требования:*

1) $t(A_i) \neq \text{null}$, $i = 1 \ldots n$, где n — количество атрибутов в отношении;

2) $t(C_i) = t(C_j)$.

Первое требование вытекает из требований стандартной реляционной модели и гарантирует, что не будет null-атрибутов для первичного ключа (в схему отношения R' в первичный ключ входят все атрибуты отношения). Второе требование гарантирует, что все атрибуты A_i схемы R' будут иметь один уровень безопасности. Это обеспечит видимость всего ключа целиком для некоторого уровня безопасности $c \in L$.

Следствие 1. Из свойства 2 и определения 2 вытекает, что для внешней ссылки $FK_{R'}$ из схемы отношения R'' на схему отношения R' выполняются следующие условия:

1) $A_i \in FK_{R'} \Rightarrow t(A_i) \neq \text{null}$;

2) $A_i, A_j \in FK_{R'} \Rightarrow t(C_i) = t(C_j)$.

Свойство 3. Сущностная целостность. Отношение r'' удовлетворяет требованию сущностной целостности, если для всех экземпляров r''_c отношения r'' и кортежа $t \in r''_c$ справедливо: $A_i \notin FK_{R'} \Rightarrow t(Ci) \geq t(C_{FK_{R'}})$.

Это свойство обеспечивает доминирование атрибутов отношения r'', не входящих в ссылку на первичный ключ отношения r', над атрибутами этой ссылки, что, в свою очередь, исключает возможность существования такого r''_c, в котором бы была не видна ссылка на отношение r'.

Свойство 4. null-целостность. Отношение r'' удовлетворяет null-целостности, если для каждого экземпляра отношения r''_c справедливы следующие условия:

1) для всех $t \in r''_c$, $t(A_i) = \text{null} \Rightarrow t(C_i) = t(C_{FK_{R'}})$;

2) будем говорить, что некоторый кортеж t поглощает кортеж s ($t, s \in r''_c$), если для каждого атрибута A_i справедливо, что $t(A_i, C_i) = s(A_i, C_i)$ или $t(A_i) \neq \text{null}$, а $s(A_i) = \text{null}$. Второе условие заключается в том, что r''_c не должно содержать двух различных кортежей, таких, что один поглощает другой.

Первое из этих условий обеспечивает сокрытие от менее привилегированных пользователей информации о возможном существовании данных на более высоких уровнях безопасности. Второе правило исключает «лишние» кортежи.

Свойство 5. Межэкземплярная целостность. Отношение r''_c удовлетворяет свойству межэкземплярной целостности, только если для всех уровней безопасности $c' \leq c$, c', $c \in L$ справедливо, что $r''_{c'} = \sigma(r''_c, c')$, где σ — функция фильтрации. Эта функция получает экземпляр $r''_{c'}$ из r''_c по следующим правилам:

1) для каждого кортежа $t \in r''_c$, $t(C_{AK}) \leq c'$ существует кортеж $t' \in r''_{c'}$, $t'(AK, C_{AK}) = t(AK, C_{AK})$ и для всех $A_i \notin AK$ выполняется:

$$t'(A_i, C_i) = \begin{cases} t(A_i, C_i), & \text{если } t(C_i) \geq c' \\ < \text{null}, t(C_{PK}) >, & \text{если нет;} \end{cases}$$

других кортежей в $r''_{c'}$ не существует;

2) для получения конечного результата $r''_{c'}$ приводится в соответствие с требованиями null-целостности.

Свойство межэкземплярной целостности инвариантно по отношению к отношениям со схемами R'' и R'.

Свойство 6. Целостность при множественности экземпляров первичного ключа (polyinstantiation integrity).

1. Схема отношения R'' удовлетворяет целостности при множественности экземпляров первичного ключа, если для всех $A_i \in R''$ выполняется: $FK_{R'}, C_i \rightarrow A_i$. Здесь символ «→» обозначает функциональную зависимость значения атрибута A_i от совокупности $FK_{R'}, C_i$. Это свойство определяет первичный ключ многоуровневой схемы отношения $R'' — PK_{R''}$.

Первичным ключом в схеме R'' является:

$$FK_{R'} \cup C_{1,R''} \cup C_{2,R''} \cup \ldots \cup C_{n,R''}.$$

2. Первичным ключом в схеме R' является:

$$AK \cup C_{1,R'} \cup C_{2,R'} \cup \ldots \cup C_{n,R'}.$$

Манипулирование данными в расширенной реляционной модели

Рассмотрим основные операции — «добавление данных», «изменение» и «удаление» в предложенной модели. Для наглядности будем использовать вместо реляционной алгебры нотацию языка SQL. При этом примем, что для пользователя, выполняющего операции манипулирования данными, разделение исходного отношения r со схемой R на два отношения r' и r'' (со схемами R' и R'' соответственно) может быть скрыто. Пользователь с некоторым уровнем полномочий $c \in L$ будет работать с совокупностью отношений r'_c и r''_c как с одним отношением, пользуясь представлением r_c: $r_c = r'_c : r''_c$.

Добавление данных. Операцию добавления данных пользователем, имеющим полномочия уровня $c \in L$, в общем случае можно представить следующим образом:

Insert

Into $r_c[(A_i[, A_j]...)]$

Values $(a_i[, a_j]...)$

Здесь квадратные скобки обозначают необязательные элементы, а многоточие — многократное повторение. Только атрибуты данных A_i могут быть явно определены. Метки безопасности C_i принимают значение c — уровня безопасности пользователя, от имени которого производится операция добавления.

Предположим, что существует кортеж t, такой, что $t(A_k) = a_k$, если Ak включен в список атрибутов, определяемых секцией «Into»; $t(A_k) = \text{null}$, если A_k не включен в список; $t(C_m) = c$, $m = 1 \ldots n$. В этом случае добавление данных возможно, только если выполняются следующие два условия:

1) $t(AK) \neq \text{null}$, где AK — все атрибуты, составляющие видимый первичный ключ кортежа t;

2) для всех кортежей u, $u \in r'_c$: $u(PK) \neq t(PK)$, где r'_c — экземпляр отношения r' для c-пользователя.

Само же добавление происходит по следующему алгоритму:

1) проверка возможности добавления данных;

2) добавление кортежа в отношение r' для хранения ключа новой записи;

3) добавление кортежа в отношение r'' и создание для него внешнего ключа на отношение r'.

Модификация данных. Выражение для модификации данных в некотором отношении r_c имеет следующую форму:

Update r_c

Set $A_i = s_i[, A_j = s_j]$ ...

[Where p]

Здесь s — скалярное выражение, определяющее значение некоего атрибута; p — предикат, определяющий те кортежи, которые подлежат модификации. Предикат может включать условия на метки безопасности в дополнение на условие к атрибутам данных. Но секция «Set» может иметь дело только с атрибутами данных. Измененные в результате выполнения запроса атрибуты будут иметь метки безопасности равные метке безопасности пользователя, осуществившего запрос.

В традиционной реляционной модели выполнение запроса на модификацию данных сводится к установлению для соответствующих атрибутов указанных в запросе значений. В расширенной модели процесс модификации отличается, поскольку необходимо предотвратить несанкционированные потоки данных.

Рассмотрим этот процесс. Примем S — множество кортежей, удовлетворяющих условию предиката p:

$$S = \{t \in r_c, t \text{ удовлетворяет } p\}.$$

Результат выполнения операции модификации можно представить как сумму результатов от модификации каждого кортежа в S. Модификацию кортежа t можно представить как замещение этого кортежа идентичным кортежем t' за исключением того, что атрибуты, указанные в секции «Set» операции модификации (обозначим их как «SET»), принимают новые значения:

$$r'(A_k, C_k) = \begin{cases} t(A_k, C_k), & A_k \notin SET \\ < s_k, c >, & A_k \in SET. \end{cases}$$

Для предотвращения утечки информации о модификации кортежа t к пользователям, имеющим меньший уровень привилегий, необходимо ввести дополнительный кортеж t''. Такая необходимость возникнет в случае: $\exists k$, $t(A_k) \neq \text{null}$, $t(C_k) < c$, где c — уровень полномочий пользователя, производящего модификацию данных. Кортеж t'' при этом должен содержать исходное значение атрибута A_k:

$$r''(A_k, C_k) = \begin{cases} t(A_k, C_k), & t(C_k) < c \\ < \text{null}, t(C_{PK}) >, & t(C_k) = c. \end{cases}$$

Таким образом, каждый кортеж t, $t \in S$ при модификации должен быть заменен кортежем t' и при необходимости кортежем t''. Полученный результат должен удовлетворять перечисленным выше ограничениям.

Рассмотрим, как отражается модификация данных «c»-пользователем на представлении отношения для более высокоуровневых пользователей. Здесь возможны две ситуации.

Первая связана с тем, что «c»-пользователь модифицирует некоторый кортеж u, в котором существует атрибут $u(A_k) \neq \text{null}$, $u(C_k) > c$. В силу межэкземплярной целостности «c»-пользователь видит этот кортеж в отфильтрованном виде, в котором кортежу u соответствует кортеж t: $t \in S$, $t(A_k) = \text{null}$, $t(C_k) = c$. Для предотвращения потери конфиденциальной информации заменить кортеж u кортежами t' и t'' так, как описано выше, но уже формируемые с точки зрения $c' > c$.

Вторая ситуация также связана со свойством межэкземплярной целостности. Предположим, что в секции «SET» существует атрибут A_k такой, что

$t(A_k) \neq$ null и $t(C_k) = c$. При модификации этого атрибута его значение в кортеже t будет заменено на новое — s_k. Теперь рассмотрим $r_{c'>c}$. В силу свойства межэкземплярной целостности может быть несколько кортежей $u \in r_{c'>c}$, которые имеют тот же самый первичный ключ, что и кортеж t ($u(AK, C_{AK}) = t(AK, C_{AK})$) и то же самое значение атрибута A_k ($u(A_k, C_k) = t(A_k, C_k)$). Для поддержания межэкземплярной целостности при множественности экземпляров первичного ключа необходимо во всех таких кортежах заменить значение атрибута A_k на новое — s_k. Имеем: $\forall A_k \in$ SET, $A_k \neq$ null $\Rightarrow \exists U = \{u \in r_{c'>c}: u(AK, C_{AK}) = t(AK, C_{AK}), u(A_k, C_k) = t(A_k, C_k)\} \Rightarrow$ заменяем кортеж u на новый кортеж u', идентичный кортежу u за исключением того, что $u'(A_k, C_k) = <s_k, c>$.

Удаление данных. Выражение для удаления данных имеет следующую основную форму:

Delete

From r_c

[Where p]

Здесь, как и в предыдущем случае, p — предикат, позволяющий идентифицировать те кортежи, которые подлежат удалению. В случае расширенной реляционной модели, даже если секция «Where» отсутствует, всегда добавляется одно условие: $t(TC) = c$, где c — уровень полномочий пользователя, от имени которого осуществляется удаление.

Рассмотрим простой пример. Предположим, что пользователь с полномочиями «О» хочет удалить запись о проекте с кодом «ТП18» из отношения «Проекты» (табл. 8, 9). Для этого он выполняет запрос «delete from „Проекты“ where Проекты.Код = = „ТП18“». СУБД должна в этом случае удалить кортеж (ТП18 О, Луна О, Жилой дом О) из отношения «Проекты″», а затем удалить кортеж (ТП18 О) из отношения «Проекты′». Тогда запись о проекте «Луна» будет полностью удалена из отношения «Проекты».

Но при удалении могут возникнуть и более сложные ситуации. Например, пользователь с уровнем полномочий «О» хочет удалить запись о проекте с кодом «BZM00» из отношения «Проекты» (табл. 8, 9). При этом он видит только одну запись (BZM00 О, Волна О, Пристань О). Вторую запись с несекретным ключом «BZM00» он не видит в силу требований null-целостности. Если пользователь удалит эту первую запись, то он не должен увидеть вторую запись. Иначе возникнет канал утечки о существовании скрытой информации о проекте «BZM00». Но эта вторая запись изначально даже не входила в представление для пользователя «О» уровня (r''_O).

В этом случае необходимо физически удалить кортеж (BZM00 О, Волна О, Пристань О) из отношения «Проекты″». А во втором кортеже с несекретным ключом «BZM00» повысить уровень безопасности ключа на один уровень (т. е. до уровня «К»). С точки зрения политики безопасности, это вполне допустимо, поскольку такая операция будет рассматриваться как запись данных с более низкого уровня на более высокий. Но, с точки зрения ссылочной целостности, это невозможно, пока не создан кортеж «BZM00 К» в отношении «Проекты′». Только после создания такой записи и повышения уровня безопасности ключевой связи в отношении «Проекты″» можно удалить кортеж «BZM00 О» в отношении «Проекты′». Теперь операцию удаления можно считать корректно совершенной.

Если бы кортеж (BZM00 О, Волна О, Пристань О) не существовал изначально в отношении «Проекты″», то пользователь уровня секретности «О» видел бы только второй кортеж (BZM00 О, null О, null О). Но просто так удалять такой кортеж тоже нельзя. Иначе будет потеряна конфиденциальная информация (BZM00 О, Прометей К, Строительство казарм К), которая доступна пользователю с уровнем секретности «К». Поэтому в данной ситуации надо создать кортеж «BZM00 К» в отношении «Проекты′» и затем повысить уровень секретности атрибута «Код» со значением «BZM00» в отношении «Проекты″». Только после этого можно удалить кортеж «BZM00 О» из отношения «Проекты′».

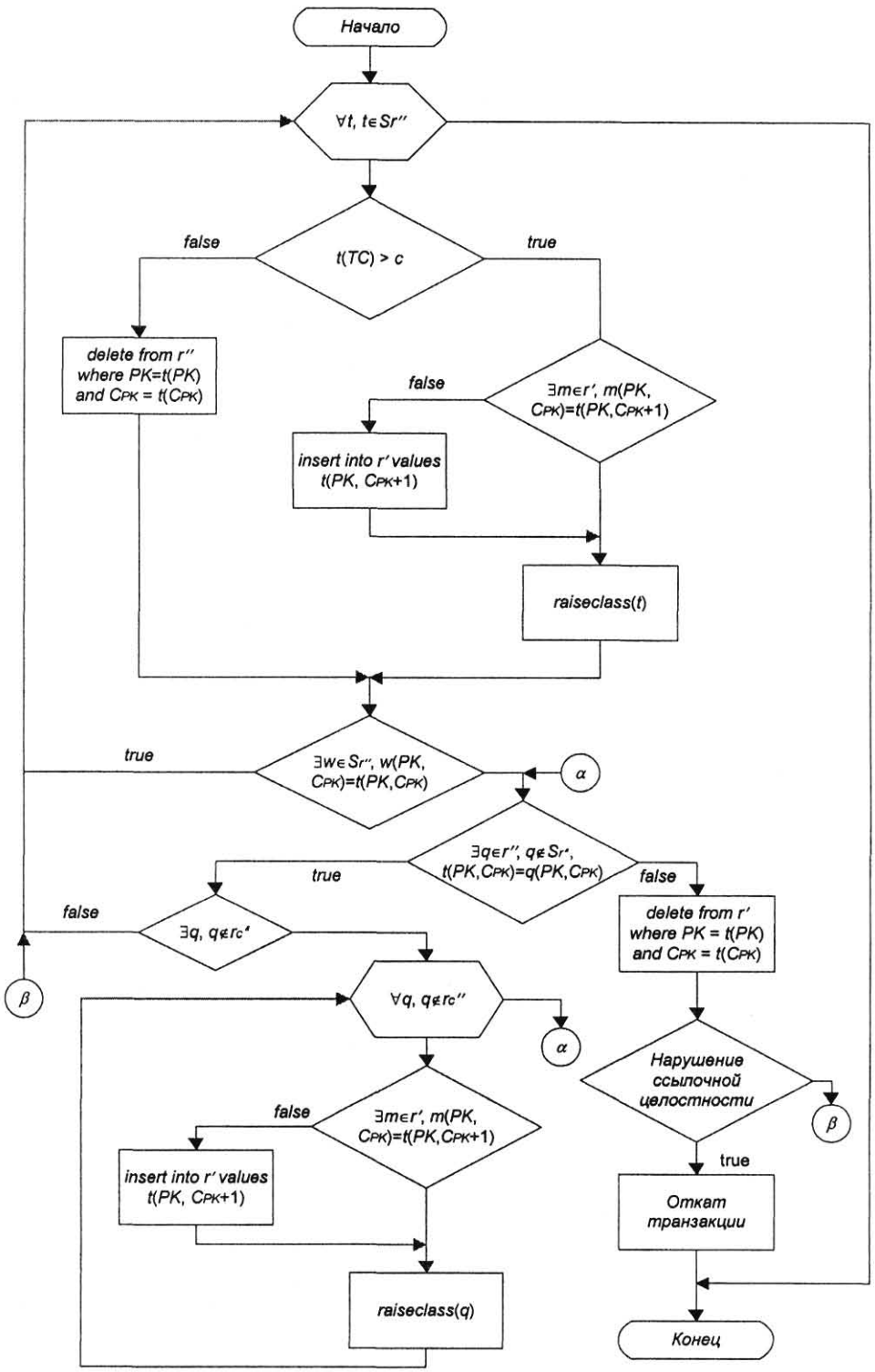
Еще один вариант удаления кортежа из отношения «Проекты»: допустим, что пользователь с полномочиями уровня «К» хочет удалить кортеж (BZM00 О, Прометей К, Строительство казарм К). В этом случае надо лишь физически удалить этот кортеж из отношения «Проекты″».

Следует отметить, что если с отношением «Проекты′» связаны другие отношения (собственно в нашем примере так и есть — имеется связь с отношением «Сотрудники»), то возникнет ситуация с нарушением ссылочной целостности при попытке удаления кортежа «BZM00 О». В этом случае процесс удаления должен быть отменен, а состояние базы данных должно вернуться в исходное — в то, которое было до начала удаления (откат транзакции).

Формально алгоритм удаления кортежа из отношения r в рассматриваемых расширениях реляционной модели представлен на рис. 1. Алгоритм учитывает все возможные случаи при удалении кортежа из отношения. В этом алгоритме для каждого кортежа из $S_{r''} = \{t \in r''_c, t$ удовлетворяет $p\}$ выполняются следующие действия, перечисленные ниже.

1. Проверяется метка безопасности удаляемого кортежа. Если она меньше или равна уровню полномочий пользователя, то производится физическое удаление кортежа из отношения r''. Если нет, то выполняется «алгоритм повышения уровня безопасности кортежа», который заключается не в физическом удалении кортежа из отношения, а в увеличении на один уровень секретности всех меток безопасности всех его атрибутов.

2. Проверяется существование среди удаляемых кортежей таких, в которых «видимый» первичный ключ и его метка безопасности идентичны «видимому» первичному ключу и метке безопасности кортежа, удаленного на шаге 1. Если такие кортежи есть, то они будут обработаны в следующих итерациях, если нет, то проверяется необходимость удаления первичного ключа из отношения r'.



■ ***Рис. 1.*** *Алгоритм удаления кортежа из отношения r' : r''*

3. Проверяется необходимость удаления ключа из отношения *r'*. Она заключается в проверке существования таких кортежей в отношении *r''*, которые не входят в число удаляемых кортежей, но у которых первичный ключ и его метка безопасности идентичны первичному ключу и метке безопасности кортежа, удаляемого на первом шаге. Если таких кортежей нет вообще, значит необходимо уда-

лить первичный ключ из отношения r'. Если такие кортежи есть, то надо проверить, есть ли необходимость в удалении первичного ключа.

4. Такая необходимость есть только в случае, когда среди найденных на шаге 3 кортежей есть такие, которые не относятся к множеству r_c. Это те кортежи, которые были вытеснены в силу требования null-целостности из представления отношения для пользователя с уровнем полномочий «*c*». Для таких кортежей запускается «алгоритм повышения уровня безопасности кортежа». После этого необходимо вернуться на шаг 3 для повторной проверки.

Алгоритм повышения уровня безопасности кортежа применяется для корректного «удаления» кортежей из отношения $r' : r''$. Алгоритм состоит из следующих шагов.

1. Проверяется в отношении r' существование ключа с меткой безопасности на уровень большей, чем метка безопасности обрабатываемого кортежа.

2. Если такого ключа не существует, необходимо создать его.

3. Для обрабатываемого кортежа выполняется функция повышения уровня безопасности кортежа raiseclass().

Функцию raiseclass можно представить следующим образом:

$$\text{raiseclass: } \forall\, i,\ i = 1...n,\ t(C_i) = c \Rightarrow$$
$$\Rightarrow t(C_i) = \text{nextclass}(t(C_i)).$$

Здесь n — количество атрибутов в схеме R; $t \in r$; функция nextclass() получает следующий уровень на решетке уровней безопасности.

Практическое использование

Многие функции, которые описаны в модели, должны возлагаться непосредственно на СУБД (например, функция фильтрации, повышения уровня безопасности, манипулирования данными). В связи с этим возникает вопрос о возможности применения модели в существующих СУБД, в стандартных реляционных базах данных.

Очевидно, что модель представима для существующих реляционных баз данных. Для этого необходимо рассматривать все элементы схем отношений R', R'' (и метки безопасности и атрибуты) как обычные атрибуты стандартной реляционной модели. Но доступ к данным в этом случае необходимо осуществлять только через хранимые процедуры, которые будут реализовывать описанную выше логику вместо СУБД. Доступ непосредственно к базе данных другими путями необходимо запретить. Это вполне можно осуществить средствами распределения полномочий современных СУБД.

Тем не менее разработка модели базы для поддержания предложенной нотации достаточно трудоемка. Необходимо не только продумать, как разделить таблицы, но и описать для каждой таблицы хранимую процедуру доступа к её данным. Поэтому для упрощения разработчику должны быть предоставлены специальные средства проектирования данных, которые могут выглядеть как расширения моделей «Сущность—Связь», и соответствующие средства для генерации физической структуры базы данных.

Заключение

В статье предложены расширения реляционной модели для обеспечения безопасности баз данных в соответствии с мандатной политикой доступа. Эти расширения, в отличие от существующих, позволяют организовать ссылочную целостность между отношениями. Разработаны алгоритмы по манипулированию данными в предлагаемой модели. Дальнейшие разработки будут направлены на создание модели для инфологического проектирования структуры базы данных на основе предложенных расширений.

Литература

1. **Federal** Criteria for Information Technology Security // National Institute of Standards and Technology & National Security Agency. — Version 1.0, 1992.
2. **Common** Criteria for Information Technology Security Evaluation // National Institute of Standard and Technology & National Security Agency (USA). Communication Security Establishment (Canada). Communications Electronics Security Group (United Kingdom). Bundesamt fur Sichereit in der Informationstechnik (Germany). Service Center de la Securite des Systemes d'Information (France). National Communications Security Agancy (Netherlands). — Version 2.1, 1999.
3. **Harrison M., Ruzzo W., Uhlman J.** Protection in operating systems // Communication of the ACM. — 1976. — Vol. 19. — N 8. — P. 461–471.
4. **Bell D. E., LaPadula L. J.** Secure Computer System: Unified Exposition and Multics Interpretation. MTR — 2997 Rev. 2. MITRE Corp., Bedford, Mass, 1976.
5. **Chen P. P-S.** The Entity-Relationship Modell. — Toward a Unified View of Data. ACM TODS 1:1. — March, 1976.
6. **Codd E. F.** A relational model of data for large shared data banks // Communication of the ACM. — 1970. — Vol.13. — N 6. — P. 377–387.
7. **Denning, Lunt T., Schell R.** A Multilevel Relational Data Model // IEEE Symp. Security and Privacy, 1987. — P. 220–234.
8. **Nelson, Doug, Chip Paradise**. Using Polyinstantiation to develop an MLS Application // Proc. IEEE 7th Annual Security Application Conf., 1991. — P. 12–22.
9. **Sushil J., Sandhu.** Enforcing Primary Key Requirements in Multilevel Relations // Proc. 4th RADC Workshop on Multilevel Database Security. Little Compton, 1991.
10. **Sandhu, Sushil J.** Honest Databases That Can Keep Secrets // Proc. 14th NIST-NCSC Nat'l Computer Security Conf. Washington, 1991. — P. 267–282.
11. **Sandhu, Ravi S., Sushil J.** Eliminating Polyinstantiation Securely // Computers and Security. — Vol. 11. — 1992. — P. 547–562.