УДК 681.3.016

СЕМАНТИКА НЕОПРЕДЕЛЕННЫХ ЗНАЧЕНИЙ В МОДЕЛЯХ БЕЗОПАСНОСТИ БАЗ ДАННЫХ

к.т.н. В.Я. Певнев, к.т.н. С.С. Танянский, к.т.н. Д.А. Руденко
(представил д.т.н., проф. Н.М. Зациркляный)

В статье рассмотрена многоуровневая модель защиты данных, основанная на идеи многозначности ключа базы данных. Предложен подход к использованию неопределенных значений в таблицах базы данных, позволяющий строить корректные запросы на языке манипулирования данными SQL.

Динамичность социальных процессов и возрастающий объем информации, необходимой для решения задач управления, требуют качественно нового подхода к организации информационного обеспечения управления и повышения его надежности.

При использовании современных информационных технологий возникает острая необходимость усовершенствования информационных систем и баз данных с целью повышения их безопасности, которая напрямую зависит от структуры, объема хранимых данных и количества пользователей. На самом элементарном уровне концепции обеспечения безопасности баз данных достаточно просты. Необходимо поддерживать два фундаментальных принципа: проверку полномочий и проверку подлинности. Проверка полномочий основана на том, что каждому пользователю или процессу информационной системы соответствует набор действий, которые он может выполнять по отношению к определенным объектам. Проверка подлинности означает достоверное подтверждение того, что пользователь или процесс, пытающийся выполнить санкционированное действие, действительно тот, за кого он себя выдает [1].

Если все пользователи, работающие в интерактивном режиме или запускающие пакетные приложения, достаточно надежны и имеют доступ к максимально закрытой информации, хранимой в системе, то сочетание средств проверки полномочий и проверки подлинности может быть вполне достаточным. Однако такая система оказывается неудовлетворительной, если необходимо организовать многоуровневую среду защиты информации. Многоуровневая защита означает, что в вычислительной системе хранится информация, относящаяся к разным классам секретности, и часть пользователей не имеют доступа к максимально секретному классу информации.

Примером подобной среды может быть вычислительная система учреждения, где в логически единой базе данных может содержаться

информация от полностью открытой до совершенно секретной. При этом степень благонадежности пользователей также варьируется от допуска только к несекретной информации до допуска к совершенно секретным данным. Таким образом, пользователь, имеющий низший статус благонадежности, может выполнять свою работу в системе, содержащей сверхсекретную информацию, но ни при каких обстоятельствах не должен быть допущен к ней.

Многоуровневая защита баз данных строится обычно на модели Белл-Ла-Падула, которая предназначена для управления активными процессами, запрашивающими доступ к информации, файлами, записями, полями или другими объектами данной информационной модели [2]. Обычно в модели Белл –Ла Падула используются два принципа ограничения доступа: простое свойство секретности и "✶-свойство"[3].

Хотя эти принципы сами по себе вполне применимы на практике, требование поддержки нескольких уровней защиты в пределах одной базе данных сопряжено с рядом проблем. Одну из них можно представить следующим примером.

Предположим, что персональные данные сотрудника хранятся в такой базе данных. "Настоящие" данные о его деятельности, которые соответствуют, например, борьбе с терроризмом в то же время могут иметь некоторое "прикрытие", согласно которому этот сотрудник является преподавателем военного университета. При этом в базе данных должна быть представлена и реальная информация, и прикрытие. Если расширить структуру таблицы, добавив свойство секретности не только для каждой строки, но и для каждого столбца, то можно столкнуться с новыми серьезными проблемами, так как каждый столбец в строке может принимать одно из нескольких возможных значений, в зависимости от классификации. Такое положение нарушает требование первой нормальной формы (1НФ), так как множественные значения могут рассматриваться как повторяющиеся группы (рисунок 1) [4].

Для решения проблемы поэлементной классификации воспользуемся свойством многозначности, когда в рамках одного отношения может существовать множество кортежей с одним и тем же значением ключа. При этом база данных маскирует значения, недоступные пользователю из соображений безопасности, и такое маскирование обычно осуществляется при помощи пустых значений представленных на рисунке 2.

Фундаментальные основы построения реляционных баз данных требуют, что бы каждый кортеж существовал в единственном экземпляре, поэтому может возникнуть ситуация записи неклассифицированных данные, содержащих как бы пустые значения. В связи с этим, в системах с многоуровневой защитой возникает проблема обработки таких кортежей, поскольку на самом деле некоторое значение пустое, а заполнить или изменить его нельзя.

В некоторых языках манипулирования данными, например в SQL, предусмотрены операторы поддержки безопасности, но они не обеспечивают

ФИО	Уровень доступа	Звание	Уровень доступа	Должность	Уровень доступа	Уровень доступа кортежа
Иванов И.И.	Н Н	Ст. л-т М-р	Н С	Преподаватель Спец.наз.	Н С	С
Петров П.П.	Н Н	К-н П\п	Н С	Нач.отд. Внешн. разведка	Н С	С

Нарушение 1НФ

Уровни доступа: Н – несекретно С – секретно **ФИО** ключ

Рис.1. Нарушение 1ФН при расширенной классификации секретности

ФИО	Уровень доступа	Звание	Уровень доступа	Должность	Уровень доступа	Уровень доступа кортежа
ванов И.И.	Н	Ст. л-т	Н С	Преподаватель	Н	С
Null	Н	М-р	С	Спец. наз.	С	С
Петров П.П.	Н	К-н	Н	Нач.отд.	Н	С
Null	Н	П\п	С	Внеш. разведка	С	С

Доступ к данным: “Несекретно”

(Вместо секретных данных пользователь видит пустые значения)

Рис.2. Использование неопределенных значений при многоуровневой защите данных

многоуровневой защиты. Возможность предоставлять привилегии доступа к определенным объектам, передавать другим пользователям пра-

ва на предоставление привилегий, отбирать предоставленные права - все это, так называемые, средства поддержки добровольного управления доступом [5]. Но при использовании многоуровневой защиты, этих возможностей недостаточно для обеспечения безопасности базы данных в совокупности с поддержкой актуальности и непротиворечивости данных. Проблема связана с необходимостью хранить в базе данных не только достоверную и полностью определенную информацию, но и такую, которая не определена в данный момент. Используя язык SQL для организации доступа (запросов) к базе данных, можно частично решить эту проблему.

В стандарте языка SQL выделяется специальное значение **Null**, которое считается автоматически входящим в любой встроенный тип данных и любой определенный пользователем домен [6]. Что имеется в виду под неопределенным значением в каждом конкретном случае, определяется семантикой предметной области. В языке фиксируется лишь следующее:

– если при вычислении арифметического выражения хотя бы одно из используемых значений есть **Null**, то и значение всего арифметического выражения есть **Null**;

– если в операциях сравнения **A Θ B (Θ = '=', '<', '>', '###', '###')** значение левого или правого операнда (или и того, и другого) есть **Null**, то истинностным значением операции сравнения является **Unknown** (неизвестно).

Таким образом, можно говорить, что в SQL используется в некотором смысле трехзначная логика. Таблицы истинности логических операций будут выглядеть следующим образом:

True AND Unknown = Unknown, **True OR Unknown = True,**
False AND Unknown = False, **False OR Unknown = Unknown,**
NOT(Unknown) = Unknown.

При этом, если логическое выражение представляет собой условие выборки строк из таблицы, то будут выбираться только те строки, для которых значение условия выборки вычисляется в **True**.

Использование пустого значения и соответственно трехзначной логики можно объяснить следующими причинами. Во-первых, наличие выделенного значения **Null** в некоторой степени решает проблему представления неполной информации. Во-вторых, трехзначная логика позволяет в ряде случаев не заботиться о наличии неопределенных значений при формулировке запросов. С другой стороны, для рассмотренного ранее примера, приведенные правила вычисления арифметических и логических выражений не являются однозначными.

Например, если предположить, что специальное значение **Null** обозначает полное отсутствие какого-либо значения в столбце соответству-

ющего кортежа, то в этом случае условие **(Null = Null) = True**, а **(A = Null) = False** для любого значения **A**, являющегося полностью определенным. Если обозначить через **Null** специальное значение, допускающее применение неопределенного значения, и считать, что диапазон возможных значений соответствующего столбца есть **(A, B)**, то, очевидно, что значение условия **(Null BETWEEN A AND B) = True**. При использовании неопределенных значений операции "больше" и "меньше" также вырабатывают логические значения **Unknown**, если значения одного или обоих операторов равны **Null**.

Таким образом, трехзначной логики в том виде, в котором она введена в SQL, оказывается достаточно в тех случаях, когда по смыслу неопределенное значение является запрещающим. Другими словами, если пользователя не интересует неполная информация, то автоматическая трактовка **Null** при вычислении арифметических выражений и выработка логического значения **Unknown** при вычислении логических выражений действительно позволяет коротко и выразительно формулировать запросы. Но для выборки неполной информации необходимо явно пользоваться **Null** как специальным значением.

Пусть **R** - некоторое отношение базы данных, содержащее неопределенные значения. Эквивалентность вида $R \gtrless R \equiv R$ (где $\gtrless$ - операция естественного соединения) корректная в двухзначной логике не является корректной в трехзначной логике. Таким образом, если запрос к таблицы (рисунок 2) имеет вид

$$T_1.\text{ФИО} = T_2.\text{ФИО} \ \text{AND} \ T_2.\text{Уровень доступа} = \text{H} \qquad (1)$$

и возможна ситуация **Unknown AND Unknown**, которая в результате дает значение **Unknown**.С другой стороны, по правилу транзитивности запрос можно преобразовать к виду

$$T_1.\text{ФИО} = T_2.\text{ФИО} \ \text{AND} \ T_2.\text{Уровень доступа} = \text{H} \ \text{AND} \ T_1.\text{Уровень доступа} = \text{H} \ , \qquad (2)$$

который может иметь вид **Unknown AND Unknown AND False**, в результате чего результат будет **False**, т.е. конкретное логическое значение.

Для примера, представленном на рисунке 2, в ответ на запрос "выдать имена сотрудников, выполняющих задание с кодом 'секретно'" пользователь с низшей категорией доступа не получит имя сотрудника, который выполняет "неопределенное" задание. Но, чтобы получить имена служащих, которые еще не выполняют никакого задания, необходимо воспользоваться структурой запроса (1) и специальным условием языка SQL - **"Is Null"**, представляющим собой, по существу, специальную форму операции сравнения для значения типа **Null**. Таким образом, для

различных категорий доступа могут генерироваться различные запросы и тем самым определяться различные результирующие значения.

В заключении следует отметить, что если классика реляционной теории говорит, что в любом отношении существуют хотя бы один возможный ключ, а первичный ключ выбирается из набора возможных ключей, то в SQL есть некоторые допущения. В соответствии с определением ключа, первичный ключ не может содержать неопределенных значений, тем не менее, ключи, объявляемые с использованием спецификации **Unique**, - могут содержать неопределенные значения. Следовательно, в таблице базы данных поддерживающей SQL может существовать возможный ключ и одновременно не существовать первичного ключа, что позволяет хранить в отношении дублирующие кортежи. В связи с этим, определение семантики баз данных с многоуровневой защитой требует точно определять, какие действия произведет, производимая операция над всеми затрагиваемыми ею объектами. Т.о., в общем случае, для поддержки многоуровневой защиты в условиях многозначности необходимо также расширить определение семантики операций манипулирования данными, доведя их до уровня, приемлемого для реализации в СУБД.

ЛИТЕРАТУРА

1. S.Castano, M.Fugini, G.Martella, P.Samarati. Database Security. – Addison - Wesley, 1995. – 422 p.
2. Емелин И.В., Эльгиян Р.В. Обеспечение многоуровневой защиты в информационных и вычислительных системах.– М.: ВНИИМИ, 1979. – 26 с.
3. S.Jajodia and R.Sandhu. Toward a Multilevel Secure Relational Data Model. – Proceedings of ACM SIGMOD, 1991. – 154 p.
4. Дедиков Э.А. Языковые средства реляционно - иерархических СУБД: Учеб. пособие. – К.: УМК ВО, 1988. – 132 с.
5. Саймон А. Стратегические технологии баз данных: менеджмент на 2000 год. – М. : Финансы и статистика, 2000. – 479 с.
6. Грофф Д., Вайнберг П. SQL: полное руководство. – BHV-Киев, 1999. – 608 с.

Поступила в редколлегию 30.5.2000