УДК 004.652.4, 004.652.6

С.С. Танянский [1], Ю.А. Мальков [2]

[1]ХНУРЭ, г. Харьков, Украина, tanyansky_ss@yahoo.com;
[2] ХНУРЭ, г. Харьков, Украина, malkov@smtp.ru

ОТОБРАЖЕНИЕ ЭЛЕМЕНТОВ РЕЛЯЦИОННОЙ МОДЕЛИ ДАННЫХ В ЭЛЕМЕНТЫ ДЕДУКТИВНОЙ МОДЕЛИ

Рассматривается соответствие понятий реляционной и дедуктивной моделей данных. Представлены отображения компонентов реляционной модели в соответствующие компоненты дедуктивной модели данных. Рассмотрены конструкции логического программирования, описывающие основные операции над данными реляционной модели.

БАЗЫ ДАННЫХ, РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ, ДЕДУКТИВНАЯ МОДЕЛЬ ДАННЫХ, DATALOG, ЛОГИЧЕСКОЕ ПРОГРАММИРОВАНИЕ, ДЕДУКТИВНЫЕ БАЗЫ ДАННЫХ

Введение

На сегодняшний день реляционная модель является стандартом для баз данных. Широкое распространение она получила из-за своей простоты и декларативности. Ведущие производители СУБД, такие как Oracle, Microsoft, IBM, DB2 выпускают именно реляционные системы.

Под моделью данных подразумевают механизм хранения и обработки данных, а также язык, обеспечивающий взаимодействие между пользователем и базой данных.

Несмотря на свою широкую распространенность, реляционная модель не идеальна и имеет ряд ограничений, одним из которых является слабая выразительная мощность стандартного для реляционной системы языка SQL. Так как в нем отсутствует рекурсия, транзитивное замыкание не может быть определено без использования внешнего процедурного языка.

Другим существенным ограничением являются строгие требования реляционных систем к типам данных, что является недостатком при использовании неоднородных источников данных.

Также к ограничениям реляционной модели можно отнести отсутствие возможности работы с неопределенной информацией.

Следовательно, для решения описанных выше задач является целесообразным использование другой модели данных. Наиболее подходящей, с точки зрения описанных недостатков, является дедуктивная модель представления данных. В отличие от реляционных систем, дедуктивные базы данных (ДБД) используют доказательно-теоретическое представление данных вместо модельно-теоретичского.

Доказательно-теоретический подход к базам данных рассматривает базу данных как комбинацию данных (аксиом) и набор теорем, которые должны быть выведены из аксиом. Выполнение запроса к дедуктивной базе данных является процессом доказательства теоремы.

В отличие от модельно-теоретического, доказательно-теоретическое представление позволяет описывать конструкции базы данных (базовые данные, запросы, ограничения целостности) единообразно – при помощи общего языка. Это позволяет создавать более понятные и унифицированные интерфейсы.

На данный момент в литературе не существует четкого определения соответствий понятий реляционной и дедуктивной моделей данных. В связи с этим, целью данной работы является определение соответствия компонентов реляционной и дедуктивной модели данных, рассмотрение языковых конструкций логического программирования, используемых в ДБД, выражение стандартных операций реляционной алгебры в терминах логического программирования.

2. Основные компоненты реляционной и дедуктивной моделей данных

2.1. Реляционная модель

В реляционной модели база данных рассматривается как набор явных именованных переменных-отношений, каждое из которых содержит явный набор кортежей и явный набор ограничений целостности.

Согласно Дейту [1] реляционная модель состоит из трех частей:

$$M_R = < S, IC, O >, \quad (1)$$

где S – структура данных; IC – ограничения целостности и O – средства манипулирование данными.

Структурная часть включает данные и описание объектов, рассматриваемых реляционной моделью:

$$S = < D, R > \quad (2)$$

Постулируется [2], что единственной структурой данных, используемой в реляционной модели, являются нормализованные n-арные отношения. Каждое отношение r характеризуется схемой $R(A_1, A_2, \ldots, A_n)$, состоящей из множества имен атрибутов $A_1, A_2, \ldots, A_n$, каждому из которых ставится в соответствие множество D_i, называемое доменом атрибута A_i.

Целостная часть описывает ограничения, которые должны выполняться для любых отношений в любых реляционных базах данных. Ограничения целостности реляционной базы данных (*IC*) можно определить формулой:

$$IC =< B, F >, \quad (3)$$

где B – бизнес-правила – ограничения, которые зависят от семантики элементов домена, а F– функциональные зависимости.

Бизнес-правила ограничивают значения атрибута отношения, например, рост человека не может быть равен 100 метрам, а функциональные зависимости определяют зависимость между соответствующими атрибутами.

Манипуляционная часть реляционной модели описывает способы определения и манипулирования данными и выражается формулой:

$$O =< DDL, DML >, \quad (4)$$

где DDL – язык определения данных (ЯОД), а DML – язык манипулирования данными (ЯМД).

2.2. Дедуктивная модель данных

Дедуктивная модель данных интерпретирует базу данных как множество предложений логики первого порядка, а под выполнением запроса или удовлетворением ограничения здесь рассматривается доказательство того, что некоторая логическая формула является логическим следствием из базовых данных.

Дедуктивная модель данных представлена формулой:

$$M_D =< EDB, P >, \quad (5)$$

где EDB – множество данных (экстенсионал), именуемых как факты, и P – множество правил вывода, являющихся логической программой.

Под *логической программой P* подразумевают конечное множество правил вывода вида:

$$p \leftarrow p_1, p_2, \ldots, p_m, not\ p_{m+1}, \ldots, not\ p_n, \quad (6)$$

где $m,n \geq 0$, а p и p_i – литералы: p называют головой правила, множество литералов p_i образует тело правила, а символом $\leftarrow$ обозначена операция импликации. Приведенная запись интерпретируется следующим образом: «Если истинно $p_1, p_2, \ldots, p_m, not\ p_{m+1}, \ldots, not\ p_n$, то истинно p».

Как видно из формулы (5), дедуктивная модель, в отличие от реляционной, обладает дополнительными возможностями, реализуемыми с помощью машины вывода логического программирования. Этот механизм, используя алгоритмы логического вывода, такие как обратный вывод и резолюция [3], способен продуцировать новые факты на основе применения правил вывода к фактам, заданным в *EDB*.

Машину вывода ДБД можно представить как функцию f:

$$EDB' = f(EDB), \quad (7)$$

где EDB – множество фактов экстенсионала; f – функция продуцирования фактов и EDB' – множество фактов, полученных вследствие логического вывода.

Логическую программу P в контексте баз данных можно представить формулой:

$$P =< IDB, IC >, \quad (8)$$

где IDB – интенсионал – часть дедуктивной БД, состоящая из множества правил вывода, а IC множество ограничений целостности.

В ДБД представление и манипулирование данными осуществляется при помощи специальных конструкций, называемых дизъюнктами (9). Дизъюнкт является конечным списком литералов вида $p(a_1, a_2, \ldots, a_n)$ или отрицаний литералов $-p(a_1, a_2, \ldots, a_n)$. Приведенные литералы являются предикатными функциями, возвращающими множество значений {0,1} (или «ложь» и «истина») и определены на множестве фактов *EDB*:

$$p(a_1, a_2, \ldots, a_n) \leftarrow p_1(a_1, a_2, \ldots, a_k), \ldots, p_m(a_1, a_2, \ldots, a_n) \quad (9)$$

Любое выражение, включая данные (факты), правила вывода, ограничения целостности и запросы, определяется при помощи дизъюнктов. В соответствии с (9) рассмотрим несколько типов дизъюнктов.

– Факты. Если в дизъюнкте будет отсутствовать тело, а все значения аргументов предиката p будут константами, то он будет представлять основную аксиому, т.е. утверждение, которое однозначно является истинным.

– Правила вывода имеют вид (9) и могут рассматриваться как дедуктивные аксиомы, которые дают определение предиката в голове правила в терминах, представленных в теле правила.

– Ограничения целостности выражаются дизъюнктами, в которых отсутствует голова.

– Запросы или цели – это дизъюнкты, тело которых содержит предикатный символ, определяющий множество фактов, над которым будет осуществляться запрос, а голова состоит из знака “?”.

Интерпретацией логической программы P называют комбинацию пространства рассуждений, отображения индивидуальных констант из этой программы на объекты в этом пространстве и набора заданных значений для предикатов и функций, содержащихся в P.

Если дизъюнкт p удовлетворяется в интерпретации τ – то говорят, что τ является моделью для p. Аналогично, если множество дизъюнктов P удовлетворяется в интерпретации τ, говорят, что τ является моделью для P.

Поскольку логическая программа может допускать несколько интерпретаций, то, следовательно, она может иметь более одной модели. Таким обра-

зом, с теоретико-доказательственной точки зрения база данных в общем случае может иметь несколько различных моделей, в отличие от модельно-теоретической, где модель всегда одна.

После введения базовых понятий логического программирования, применяемого в дедуктивных базах данных, рассмотрим соответствие основных компонентов реляционной и дедуктивной модели.

3. Отображение компонентов реляционной модели данных в компоненты дедуктивной

Рассмотрение приведенных моделей данных показало, что реляционная модель данных (1) состоит из компонентов, описывающих структуру данных (2), целостную часть (3), представленную ограничениями целостности, и языков определения и манипулирования данными (4).

В то же время дедуктивная модель данных представляется в виде множества фактов *EDB*, и логической программы (5), состоящей из множеств правил вывода и ограничений целостности (8).

Проведенный выше анализ моделей данных позволяет сделать вывод о возможности отображения составных частей реляционной модели данных в компоненты дедуктивной (рис. 1).

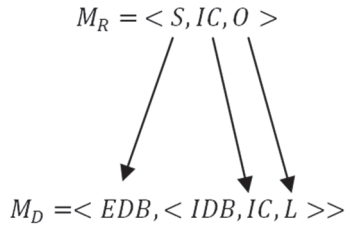

Рис. 1. Отображение компонентов реляционной модели данных в компоненты дедуктивной

Для структурной части реляционной модели можно построить отображение в множество фактов экстенсионала *EDB* дедуктивной модели:

$$\rho : S^{M_R} \to EDB^{M_D}. \qquad (10)$$

Множество ограничений целостности реляционной модели можно представить множеством правил вывода логической программы, являющейся частью дедуктивной модели:

$$\beta : IC^{M_R} \to IC^{M_D}. \qquad (11)$$

Манипуляционную часть реляционной модели данных можно выразить с помощью эквивалентных логических формул и конструкций логического программирования:

$$\theta : O^{M_R} \to L^{M_D}. \qquad (12)$$

Рассмотрим более подробно отображение каждого компонента реляционной модели.

3.1. Отображение структурной части

Структурная часть реляционной модели определяет отношение как средство хранения данных. В то же время предикат можно представить как функцию, описывающую реляционное отношение: если значения аргументов предиката являются кортежем некоторого отношения, то предикат будет возвращать истинное значение.

Определение: Пусть *r* – реляционное отношение со схемой $R(A_1, A_2, \ldots, A_n)$, которое содержит конечное множество кортежей $\{t_1, t_2, \ldots, t_k\}$. Каждый кортеж $t_i(A_1, A_2, \ldots, A_n)$ представляется в виде литерала $p_i(a_1, a_2, \ldots, a_n)$, где каждый атрибут A_j кортежа t_i отображается в аргумент a_j предиката p_i:

$$\phi : A_j^{t_i} \to a_j^{p_i}. \qquad (13)$$

Поскольку предикат является функцией, вышесказанное можно переформулировать следующим образом: предикат p_i будет принимать значения “истина” лишь в том случае, если его аргументы будут отображением значений некоторого кортежа t_i из отношения *r*.

Рассмотрим пример: пусть дано отношение *r* (рис.2), состоящее из двух кортежей $\{a_1, b_1, c_1\}$ и $\{a_1, b_2, c_2\}$. Представим данное отношение в терминах дедуктивной модели с помощью предиката *p*. Тогда предикат *p* будет возвращать значение “истина” лишь при подстановке в качестве его аргументов $\{a_1, b_1, c_1\}$ и $\{a_1, b_2, c_2\}$.

A	B	C
a_1	b_1	c_1
a_1	b_2	c_2

$p(a_1, b_1, c_1).$
$p(a_1, b_2, c_2).$

Рис. 2. Отображение реляционного отношения в множество фактов

Кортеж отношения *r*, представленный в виде *n*-арного предикатного символа $p(a_1, a_2, \ldots, a_n)$, где *n* арность исходного отношения *r*, а a_i – константа, называют *фактом*. Множество всех фактов дедуктивной базы данных принято называть *экстенсионалом*.

В реляционной модели данных понятие отношения непосредственно связано с понятием домена [2]. В связи с этим рассмотрим соответствующие домену понятия дедуктивной модели и построим отображение.

Определение: пусть $D = \{D_1, D_2, \ldots, D_n\}$ – множество доменов, где $D_i = \{d_1^i, d_2^i, \ldots, d_k^i\}$, где d_j^i - значение домена $(1 < j < k)$. Определим предикат $p(a_1, a_2, \ldots, a_n)$ и множество значений $a_j^i = \{a_1^i, a_2^i, \ldots, a_k^i\}$ его аргумента a_i. Тогда, можно построить отображение γ множества элементов домена D_i в множество значений аргумента a_i:

$$\gamma : d_j^i \to a_j^i. \qquad (14)$$

Представим систему отображений структурной части реляционной модели в экстенсионал дедуктивной при помощи диаграммы (рис. 3).

Пусть S^{M_R} – множество отношений, определенных в реляционной модели M_R, а EDB^{M_D} –

множество фактов экстенсионала M_D. Пространство допустимых состояний, выразимых в M_R и M_D, обозначим, соответственно B_R и B_D. Тогда ρ – отображение вида (10), $\omega: B_R \to B_D$ – отображение пространства состояний M_R в пространство состояний M_D, а μ^{Def} – семантическая функция модели языка определения данных.

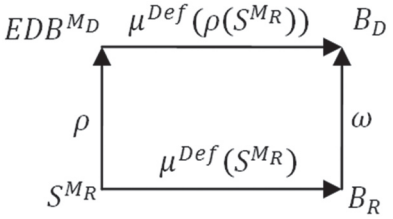

Рис. 3. Диаграмма отображения компонент M_R в M_D

3.2. Отображение манипуляционной части

Как было сказано выше, в основе дедуктивной модели лежит доказательно-теоретический подход к базам данных, расширяющих классическое понимание базы данных за счет применения логического программирования.

Наиболее распространенным языком, применяемым в ДБД, является Datalog.

Семантика Datalog определена на алфавитах $Var, Const\ 8\ Pred$, обозначающих переменные, константы и предикаты.

Переменные обозначают строками буквенно-цифровых символов конечной длины, начинающихся с прописной буквы. Следует отметить, что существует специальная переменная «_», называемая анонимной. Ею обозначают любую неименованную переменную. Анонимные переменные используют для отсечения, некоторых аргументов предиката, что аналогично операции проекции в реляционной алгебре.

Константой является текстовая строка или число. Например: “строковая константа” или “12”.

Предикаты обозначают строками алфавитно-цифровых символов конечной длины, которые начинаются со строчной буквы. Как говорилось выше, предикаты являются булевыми функциями над множествами данных.

К функциям Datalog, помимо формулирования дедуктивных правил вывода, относят описание запросов. Для выражения запросов, называемых в логическом программировании целями, на языке Datalog используют дизъюнкты вида (15). Голова цели состоит из символа “?”, а тело – из предиката, над которым выполняют запрос и условий отбора:

$$?\leftarrow p(A_1, A_2, \ldots, A_k, \ldots, A_n), A_k\, \Omega C, \qquad (15)$$

где, p – предикат, определенный на множестве фактов EDB^{M_D}; A_i – переменные, связанные с значениями аргументов предиката; $A_k\, \Omega C$ условие выборки; Ω – множество логических функций сравнения $(>,<,=,\neq)$; C – константа. Следовательно, результатом запроса будут являться все факты p, у которых аргумент A_k будет отвечать заданному условию отбора.

Цели, как и любые дизъюнкты, не содержат свободных переменных. Т.е. все появляющиеся в цели переменные связаны квантором всеобщности. Из этого следует, что переменная, появляющаяся в некоторой цели G, имеет смысл только в этой цели и ее значение не распространяется на другие составляющие логической программы.

Синтаксис логического программирования основан на использовании операций булевой алгебры, к которым относятся отрицание, импликация, конъюнкция и дизъюнкция. В Datalog для выражения булевых операций используются символы и соглашения, определенные в табл. 1.

Таблица 1

Выражение булевых операций в Datalog

Дизъюнкция	«;»
Конъюнкции	«,»
Отрицание	«not»
Импликация	« ← »

Синтаксис языка Datalog является более выразительным чем, языки реляционных систем, и все операции реляционной алгебры и реляционного исчисления можно выразить с помощью эквивалентных логические операций и конструкций логического программирования. Рассмотрим их подробно (табл. 2).

Таблица 2

Соответствие формул Datalog операциям реляционной алгебры

Операция реляционной алгебры	Обозначение	Равнозначное выражение на Datalog
Объединение	$R \cup S$	$q(X_1, X_2, \ldots, X_n) \leftarrow r(X_1, X_2, \ldots, X_n).$ $q(X_1, X_2, \ldots, X_n) \leftarrow s(X_1, X_2, \ldots, X_n).$
Пересечение	$R \cap S$	$q(X_1, X_2, \ldots, X_n) \leftarrow r(X_1, X_2, \ldots, X_n), s(X_1, X_2, \ldots, X_n).$
Разность	$R - S$	$q(X_1, X_2, \ldots, X_n) \leftarrow r(X_1, X_2, \ldots, X_n), not\ s(X_1, X_2, \ldots, X_n).$
Декартово произведение	$R \times S$	$q(X_1, X_2, \ldots, X_{k+s}) \leftarrow r(X_1, X_2, \ldots, X_k), s(X_{k+1}, X_{k+2}, \ldots, X_{k+s}).$
Селекция	$\sigma_C(R)$	$q(X_1, X_2, \ldots, X_k \ldots, X_n) \leftarrow r(X_1, X_2 \ldots, X_k, \ldots, X_n), X_k > C.$
Проекция	$\pi_{X,Y,\ldots,Z}(R)$	$q(X_1, X_2, \ldots, X_k) \leftarrow r(X_1, X_2, \ldots, X_k, \_, \ldots, \_).$

Операции соединения и деления являются зависимыми операциями, и следовательно могут быть выражены с помощь рассмотренных операций разности и декартова произведения.

Представленные выражения на Datalog позволяют выражать все операции манипулирования данными, используемые в реляционной модели. В связи с этим можно построить диаграмму отображения языка манипулирования данными реляционной модели в дизъюнкты Datalog (рис. 4).

Пусть O^{M_R} – множество операторов языка манипулирования данными модели M_R, L^{M_D} – множество рассмотренных выражений на Datalog, представляющих ЯМД модели M_D, эквивалентных операциям реляционной алгебры. Тогда θ – отображение вида (12), $\omega: B_D \to B_R$ – отображение пространства состояний M_R в пространство состояний M_D, а μ^{Man} - семантическая функция модели языка манипулирования данными.

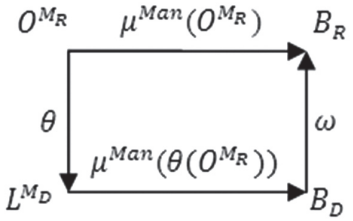

Рис. 4. Диаграмма отображения операторов ЯМД

3.3. Отображение целостной части

Под ограничениями целостности подразумевают соответствие имеющейся в базе данных информации её внутренней логике, структуре и всем явно заданным правилам. Каждое правило, налагающее некоторое ограничение на возможное состояние базы данных, называется ограничением целостности.

Ограничения целостности, используемые в реляционной модели, можно классифицировать следующим образом [4]:

1. Первичные ключи.
2. Ограничения ссылочной целостности.
3. Ограничения области значений.

В общем случае ограничения целостности на Datalog выражают при помощи правил вида (16). Голова такого правила является пустой, а тело состоит из предиката, который нарушает целостность базы при истинности условия:

$$\leftarrow p(A_1, A_2, \ldots, A_k, \ldots, A_n), A_k\, \Omega C, \qquad (16)$$

где p – предикат, определенный на множестве фактов EDB^{M_D}; A_i – переменные, связанные с значениями аргументов предиката; $A_k\, \Omega C$ условие; C – константа. Из приведенного выражения следует, что целостность базы данных будет нарушена, если найдется хоть один факт p, у которого аргумент A_k будет отвечать заданному условию.

Первичным ключом называют атрибут или множество атрибутов, уникальным образом идентифицирующих объект в его классе. Никакие два объекта класса не могут совпадать по своим значениям для каждого множества атрибутов, формирующих ключ. Применительно к ДБД, приведенное определение означает, что во множестве EDB^{M_D} не должно существовать двух фактов с одинаковыми значениями аргументов, определяющих уникальность факта.

Для представления первичных ключей в дедуктивной модели данных вводят правила вида:

$$unique\, key(p, \gamma), \qquad (17)$$

где p – имя ограничиваемого ключом предиката и γ подмножество аргументов p, определяющих состав ключа. Подмножество аргументов γ задают в виде перечисления позиций аргументов, разделенных запятой. Например: *unique key*(p, [1,3]) .

Ограничение ссылочной целостности в реляционной модели заключается в том, что каждому кортежу отношения, содержащему внешний ключ, должен соответствовать кортеж в другом отношении, содержащем первичный ключ. Отношение, содержащее внешний ключ, называют дочерним, а первичный – родительским.

В дедуктивной модели ограничение ссылочной целостности в общем виде представляют правилом вида:

$$\leftarrow p_1(A_1, \ldots, A_k, \_\ldots, \_), not\ p_2(A_1, \ldots, A_k, \_, \ldots, \_), \qquad (18)$$

где p_1 – предикат, на который накладывается условие ссылочной целостности; p_2 – родительский предикат для p_1; $A_1, A_2, \ldots, A_k$ – атрибуты, входящие в состав внешнего ключа.

Ограничения области значений требуют, чтобы значение атрибута реляционного отношения выбиралось из определенного множества значений или находилось в определенных границах. Ограничения такого вида можно представить правилами вида (19, 20):

$$\leftarrow p(A_1, \ldots, A_k, \ldots, A_n), not\ A_k = C_1, \ldots, not\ A_k = C_m, \qquad (19)$$

где p – предикат; A_k – аргумент предиката, значение которого, не входящее в множество значений $\{C_1, C_2, \ldots, C_m\}$, нарушает целостность БД,

$$\leftarrow p(A_1, A_2, \ldots, A_k, \ldots, A_n), A_k < C_1, C_2 < A_k, \qquad (20)$$

где p – предикат; A_k – аргумент предиката, значение которого, выходящее за пределы диапазона $\{C_1, C_2\}$, нарушает целостность БД.

Кроме представленных выше ограничений целостности, накладываемых на факты *EDB*, дедуктивная система позволяет ограничивать вывод фактов, полученных в результате логического вывода (EDB'). Следовательно, ограничения целостности в дедуктивной модели можно представить формулой:

$$IC =< IC^{EDB}, IC^{EDB'} >, \qquad (21)$$

где IC^{EDB} – ограничения целостности, накладываемые на множество фактов EDB, а $IC^{EDB'}$ – ограничения целостности, накладываемые на множество продуцируемых фактов EDB'. Поскольку в реляционной модели отсутствует механизм целостности хранимых запросов, механизмы, аналогичные $IC^{EDB'}$, в ней не представлены.

4. Использование дедуктивных баз данных в задачах интеграции реляционных систем

Достаточно часто реляционные системы используют для хранения табличных документов с нарушением требований реляционной модели. Это связано с ошибками и недоработками при проектировании, к которым можно отнести отсутствие нормализации и различную степень абстракции данных. Интеграция таких систем является достаточно сложной задачей и требует более мощных и выразительных средств, нежели реляционные языки.

Для решения этой задачи предлагается использовать аппарат дедуктивных баз данных, значительно расширяющий операционную спецификацию реляционных систем. Как было показано выше, все основные структурные компоненты реляционной модели выразимы средствами логического программирования, следовательно, их можно однозначно отобразить в дизъюнкты ДБД. Применение дедуктивных правил вывода к фактам, представляющим различные неоднородные системы баз данных, позволит представить их в виде единого виртуального множества данных, над которым можно осуществлять запросы. Общую схему работы предлагаемого метода представим на рис. 5.

4.1. Архитектура системы интеграции неоднородных баз данных на основе ДБД

Как было показано выше, любую реляционную базу данных можно представить в виде логической программы *P* и множества фактов *EDB*. Следовательно, множество интегрируемых систем баз данных $\{DB_1, DB_2, \ldots, DB_n\}$ можно представить как совокупность множеств фактов $\overline{EDB} = \{EDB_1, EDB_2, \ldots, EDB_n\}$ и логической программы $\overline{P}$, включающей правила, заданные в $P_1, P_2, \ldots, P_n$.

Поскольку, интегрируемые базы данных зачастую имеют различную степень нормализации и абстракции данных, а также могут содержать нарушения требований реляционной модели, необходимо задать механизм приведения локальных структур данных к глобальной виртуальной схеме. Это достаточно просто реализовать при помощи правил вывода вида (6).

Рассмотрим пример. Пусть даны две базы данных, представленные отношениями R_1 и R_2, R_3 (рис. 6). Представим их с помощью предикатов p_1, p_2, p_3 соответственно.

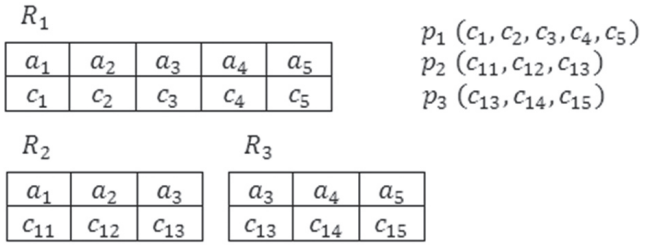

Рис. 6. Отношения R_1, R_2, R_3 и их представление в виде фактов p_1, p_2, p_3

Поскольку предикаты p_1, p_2, p_3 обладают различным количеством аргументов, определим правила вывода, которые приведут описанные предикаты к глобальной схеме. Глобальную схему обозначим при помощи предиката *p*:

$$p(X_1, X_2, X_3, X_4, X_5) \leftarrow p_1(X_1, X_2, X_3, X_4, X_5).$$
$$p(X_1, X_2, X_3, X_4, X_5) \leftarrow p_2(X_1, X_2, X_3), p_3(X_3, X_4, X_5). \qquad (22)$$

Подробнее правила преобразования различных вариантов неоднородных структур к глобальной схеме рассмотрены в [8].

Рассмотрим схему осуществления запроса над глобальной схемой (рис. 7).

Рис. 5. Схема интеграции реляционных СУБД на основе ДБД

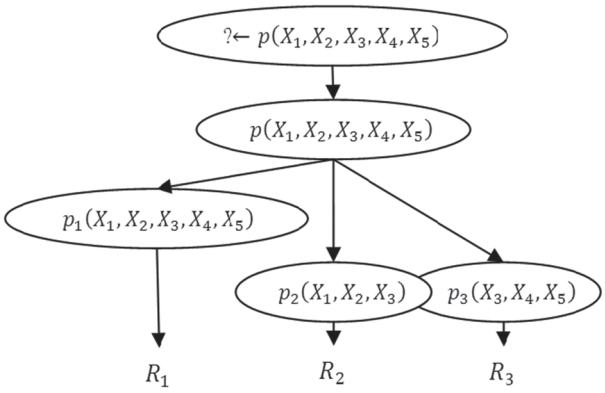

Рис. 7. Схема осуществления запроса над глобальной схемой $\overline{EDB}$

При осуществлении запроса над глобальной схемой $\overline{EDB}$ машина вывода Datalog, используя алгоритмы нисходящего вычисления [3], конструирует дерево доказательства запроса, начиная с предиката заданного в глобальной схеме и заканчивая на нижнем уровне, содержащем факты локальных экстенсионалов.

Выводы

В силу исторически сложившихся обстоятельств реляционная модель данных является доминирующей на рынке современных систем управления базами данных. Рассмотренные недостатки не позволяют решать ряд определенных задач из-за слабой выразительности реляционной модели. В то же время дедуктивная модель имеет более мощный манипуляционный компонент за счет использования логического программирования. Описанный в работе ряд отображений компонентов реляционной модели в элементы дедуктивной позволяет утверждать о возможности однозначной трансляции одной модели в другую. Представление реляционных систем в дедуктивном виде позволит решить ряд проблем, связанных с интеграцией неоднородных баз данных в силу большей выразительной мощности Datalog и отсутствия жестких требований к типизации данных.

Полученные результаты дают основания для дальнейшего изучения механизма отображения ограничений целостности, заданных в локальных системах, в ограничения целостности глобальной схемы. Поскольку ограничения целостности, описывающие локальную схему, не всегда допустимы для глобальной схемы, а их удовлетворение может привести к потере данных.

Список литературы: 1. *Дейт, К.Дж.* Введение в системы баз данных [Текст]: пер. с англ. – М.: Вильямс, 2006. – 1071 с. **2.** *Кодд, Э.Ф.* Реляционная модель данных для больших совместно используемых банков данных [Текст] / Э.Ф. Кодд // Системы управления базами данных. – 1995. – № 1. – С. 145-160. **3.** *Чери, С.* Логическое программирование и базы данных [Текст] : пер. с англ. / С. Чери, Г. Готлоб, Л._Танка.– М.: Мир, 1992. – 352 с. **4.** *Ульман, Дж. Д.* Введение в системы баз данных [Текст]: пер. с англ. – М.: Лори, 2000. – 374 с. **5.** *Калиниченко, Л.А.* Методы и средства интеграции неоднородных баз данных [Текст] / Л.А. Калиниченко. – М.: Наука, 1983. – 424 с. **6.** *Zaniolo, C.* Key Constraints and Monotonic Aggregates in Deductive Databases [Текст] / C. Zaniolo // Computational Logic: Logic Programming and Beyond. –2002. – С. 109-134. **7.** *Гарсиа-Молина Г.* Системы баз данных. Полный курс [Текст]: пер. с англ. / Г. Гарсиа-Молина, Дж. Ульман, Дж. Уидом.– М.: Вильямс, 2003. – 1088 с. **8.** *Танянский, С.С.* Организация запросов к распределенным данным средствами логического программирования [Текст] / С.С. Танянский, Ю.А. Мальков // Бионика интеллекта: науч.-техн. журнал. – 2010. – №1(72). С. 118-121.

Поступила в редколлегию 14.09.2011

УДК 004.652.4, 004.652.6

Відображення елементів реляційної моделі даних в елементи дедуктивної моделі / С.С. Танянський, Ю.А. Мальков // Біоніка інтелекту: наук.-техн. журнал. – 2011. – № 3 (77). – С. 136-142.

В статті розглянуто відображення компонентів реляційної моделі в компоненти дедуктивної, що дає підстави для побудови системи інтеграції з локальнонезалежними даними, заснованої на використанні логічного програмування та дедуктивного представлення бази даних.

Іл.: 7. Бібліогр.: 8 найм.

UDK 004.652.4, 004.652.6

Mapping relational model items to deductive model items / S.S. Tanyansky, Y.A. Malkov // Bionics of Intelligense: Sci. Mag. – 2011. – № 3 (77). – P. 136-142.

This article was focused on mapping items of relational model to deductive model items. This mapping provides a basis for constructing integration system with locally independent data, which based on application of logic programming and deductive presentation of databases.

Fig.: 7. Ref.: 8 items.