Формирование гиперкубового представления данных со списочными компонентами

С.В. Зыкин

Аннотация. В статье рассмотрена проблема автоматизации формирования гиперкубового представления данных из исходной реляционной базы данных. При этом предполагается, что в ячейке гиперкуба может присутствовать несколько однородных значений (список), используемых при анализе данных. Предлагаемая в данной статье технология является развитием традиционных OLAP-технологий.

Ключевые слова: гиперкуб, реляционная база данных, OLAP-технология.

Введение

Обработка и анализ накопленной информации является актуальной для многих исследовательских и прикладных задач. Наиболее популярным способом ее решения в настоящее время является технология оперативной аналитической обработки данных OLAP (online analytical processing) и Data Mining. Основу OLAP-технологии составляет построение гиперкубового (многомерного) представления данных. Проблема автоматизации анализа данных актуальна как для пользователей больших корпоративных информационных систем, так и для пользователей сравнительно небольших баз данных, поскольку одни и те же данные приходится многократно реорганизовывать вручную, чтобы обработать их тем или иным алгоритмом поиска закономерностей.

В работах, посвященных OLAP, значительное внимание уделяется исследованию свойств моделей гиперкубов [1-3] и операциям их преобразования [2,4] с целью получения представления, необходимого для анализа данных. Особое внимание уделяется построению иерархий в размерностях [2,3,5-7], что позволяет гарантировать корректность операций агрегации данных. В работах [3,5,7] рассматриваются нормальные формы для многомерных моделей данных, которые позволяют контролировать значения NULL в иерархиях размерностей.

В данной статье предполагается, что основой аналитической работы пользователя является формирование различных гиперкубов из исходного реляционного представления данных. Это необходимо при выявлении скрытых закономерностей в данных и проведения анализа данных, не предусмотренного при проектировании складов данных. Следовательно, основное внимание необходимо акцентировать на сокращении времени формирования схемы и представления нового гиперкуба. Формирование представления гиперкуба должно быть выполнено автоматически алгоритмами в соответствии с выбранной схемой.

Далее будем предполагать следующую технологическую последовательность обработки данных:

- исходные данные должны быть представлены в реляционном нормализованном ви-

[1] Работа поддержана проектом РФФИ № 09-07-00059-а

де [8, 9], к ним должен обеспечиваться доступ по технологии OLTP (online transaction processing);

- пользовательское представление данных в виде гиперкубов, реализующее технологию OLAP, обеспечивается инструментарием, преобразующим исходные данные в необходимый на данный момент гиперкуб;
- гиперкубовое представление данных далее используется в специализированном пакете программ для классификации, кластеризации, прогнозирования и т.д.

Существенные затраты времени для установления соответствия между реализацией БД и реализацией гиперкуба в данной работе предлагается сократить за счет автоматизации этого процесса с использованием свойств схемы исходной операционной базы данных.

1. Постановка задачи

В работе [10] для автоматизации формирования гиперкуба предлагается использовать последовательность преобразований:

$$RRD \Rightarrow TJ \Rightarrow GC,$$

где ***RRD*** – реляционное представление данных, ***TJ*** – таблица соединений, ***GC*** – гиперкуб (далее соответствующие модели будут формально определены). В данном случае ***RRD*** – представление исходной модели данных, а ***GC*** – целевой модели. Наличие промежуточного представления ***TJ*** позволяет динамически управлять содержимым гиперкуба за счет определения контекстных ограничений и логических ограничений на данные.

В настоящей работе предусматривается возможность формирования списка однородных значений в отдельной ячейке гиперкуба без их агрегирования, поскольку списки таких значений используются во многих алгоритмах анализа данных.

Рассмотрим формализацию задачи. Пусть задана схема базы данных $\mathfrak{R}=\{R_1, R_2, ..., R_k\}$, полученная в результате нормализации отношений [8, 9]. Отношения R_i определены на множестве атрибутов $U=\{A_1, A_2, ..., A_n\}$. Пусть $[R_i]$ – схема отношения, множество атрибутов, на которых определено отношение R_i. Предположим, что схема $\mathfrak{R}$ является редуцированной [9], т.е. не существует двух отношений таких, что $[R_i]\subseteq[R_j]$, при $i\neq j$. Кортеж $t[X]$ – совокупность значений атрибутов $A_j\in X\subseteq[R_i]$, заданных в кортеже $t\in R_i$. Неопределенное значение NULL атрибута A_j в кортеже t: $t[A_j]$=NULL не равно любому другому значению, в том числе другому неопределенному значению.

Гиперкуб будем задавать в виде совокупности размерностей: $\{D_1,D_2,\ldots,D_d\}$, где D_l – множество расширенных имен атрибутов: $R_i.A_j$, $A_j\in[R_i]$, M – множество мер, также заданных в виде расширенных имен атрибутов. Значения D_l являются значениями координат гиперкуба, значения M будут располагаться в рабочей области гиперкуба. Для каждой размерности задается ограничение в виде логической формулы F_l, которая формируется из элементарных условий, связанных операциями конъюнкции, дизъюнкции и отрицания. Элементарные условия заданы на атрибутах отношений: $R_i.A_j\theta R_{i'}.A_{j'}$ либо $R_i.A_j\theta<const>$, $\theta\in\{\geq, \leq, >, <, =, \neq\}$, $<const>$ – константа либо неопределенное значение, задаваемое пользователем при активизации приложения. Причем, атрибуты в выражении F_l могут не принадлежать ни размерностям, ни мерам, а ограничение будет задаваться опосредованно, за счет контекстов. Способ формирования множеств отношений, из которых будут получены значения размерностей и мер гиперкуба, обсуждается далее.

В данной работе предлагается отказаться от необходимости выполнения функциональной зависимости [3,5,7,10]:

$$D_1D_2\ldots D_d\rightarrow M, \qquad (1.1)$$

что позволит иметь в одной ячейке гиперкуба несколько значений (список) атрибута $R_i.A_j\in M$. Рассмотрим примеры представлений данных, для которых такое предположение является актуальным.

Пример 1. Рассмотрим фрагмент учебного плана в Вузе [11]. Задано множество атрибутов: A_1 – Специальность, A_2 – Предмет, A_3 – Семестр, A_4 – Количество часов, A_5 – Вид занятия, A_6 – Контроль успеваемости, A_7 – Номер специальности, A_8 – Номер предмета. На предложенном множестве атрибутов существуют следующие зависимости: $DEP=\{A_7\rightarrow A_1,\ A_8\rightarrow A_2,\ A_7A_8A_3A_5\rightarrow A_4,\ A_7A_8A_3\twoheadrightarrow A_5(A_6)\}$ (три функциональных и одна многозначная). По правилам построения нормальных форм [8,9] будет полу-

чена следующая схема базы данных: Специальности = $R_1(\underline{A_7}, A_1)$, Предметы = $R_2(\underline{A_8}, A_2)$, Учебная нагрузка = $R_3(\underline{A_7}, \underline{A_8}, \underline{A_3}, \underline{A_5}, A_4)$, Контроль = $R_4(\underline{A_7}, \underline{A_8}, \underline{A_3}, \underline{A_6})$, где подчеркнуты ключевые атрибуты отношений. Одно из возможных представлений гиперкуба приведено в Табл. 1.

В Табл.1 атрибуты размерностей представлены жирным шрифтом, атрибуты мер – курсивом, значения атрибутов – обычным шрифтом.

Схема гиперкуба в Табл. 1 может быть представлена в следующем виде:

$$\{R_1.A_1\}\times\{R_3.A_3\{R_2.A_2\{R_3.A_5(R_3.A_4)\}(R_4.A_6)\}\}, \quad (1.2)$$

где $D_1=\{R_1.A_1\}$ и $D_2=\{R_3.A_3, R_2.A_2, R_3.A_5\}$ – размерности, $M=\{R_3.A_4, R_4.A_6\}$ – меры. Логическое ограничение: $F=(R_4.A_3=2\wedge(R_2.A_2=$'Математика' $\vee R_2.A_2=$'Физика'$))$.

В литературе, посвященной моделированию гиперкубов [3,5,7,10], меры рассматриваются отдельно от размерностей, поскольку предполагается, что иерархии в каждой размерности имеют по одному терминальному уровню. В Табл.1 размерность D_2 имеет два терминальных уровня $R_2.A_2$ и $R_3.A_5$, так как у каждого из них есть собственная сопоставленная мера, хотя между ними установлено иерархическое подчинение. Если бы меры не были сопоставлены размерностям, то возникла бы неоднозначность в расположении значений мер. Таким образом, схема (1.2) задает иерархию для размерности D_2, в которой терминальными уровнями фактически являются меры. Кроме того, меры должны быть сопоставлены только атрибутам одной размерности, поскольку в противном случае в одной ячейке таблицы надо было бы сопоставить разнородные значения мер, либо дополнительно дробить ячейки в рабочей области гиперкуба.

Табл.1 может иметь другую схему, например:

$$\{R_1.A_1\{R_2.A_2\{R_3.A_5(R_3.A_4)\}(R_4.A_6)\}\}\times\{R_3.A_3\}.$$

Структура таблицы при этом изменится, а содержание останется прежним.

Пример 2. Рассмотрим фрагмент базы данных лечебного заведения [12]. Задано множество атрибутов: A_1 – № пациента, A_2 – ФИО пациента, A_3 – № показателя, A_4 – показатель, A_5 – значение показателя, A_6 – № дня получения показателя, A_7 – группа пациентов. На предложенном множестве атрибутов существуют следующие зависимости: $DEP=\{A_1\rightarrow A_2A_7,\ A_3\rightarrow A_4,\ A_1A_3A_6\rightarrow A_5\}$. По правилам построения нормальных форм будет получена следующая схема базы данных: Пациенты = $R_1(\underline{A_1}, A_2, A_7)$, Перечень анализов = $R_2(\underline{A_3}, A_4)$, Результаты анализов = $R_3(\underline{A_1}, \underline{A_3}, \underline{A_6}, A_5)$. Одно из возможных представлений гиперкуба в Табл. 2.

Схема гиперкуба имеет вид:

$$\{R_3.A_6\}\times\{R_2.A_4\{R_1.A_7(R_3.A_5)\}\}, \quad (1.3)$$

где $D_1=\{R_3.A_6\}$ и $D_2=\{R_2.A_4, R_1.A_7\}$ – размерности, $M=\{R_3.A_5\}$ – мера. Логическое ограничение: $F=(R_2.A_4=$'Креатинин'$\vee R_2.A_2=$'Белок'$\vee R_2.A_2=$ 'Билирубин'$)\wedge(R_1.A_7=2\vee R_1.A_7=3)$. Списками значений Табл. 2 удобно пользоваться при анализе данных с использованием критерия Уилкоксона,

Табл. 1. Фрагмент учебного плана в вузе

Семестр	2							
Предмет	Математика				Физика			
Вид занятия	Лекции	Практика	лаб.раб.		Лекции	практика	лаб.раб.	
Специальность	*Количество часов*	*Количество часов*	*Количество часов*	*Контроль успеваемости*	*Количество часов*	*Количество часов*	*Количество часов*	*Контроль успеваемости*
История	36	36	18	зач.,экз.	18	18	18	зач.
Филология	18	18	18	зач.	36	36	18	зач.,экз.
Правоведение	48	48	24	зач.,экз.	48	48	24	зач.,экз.

Табл. 2. Сводная таблица анализов пациентов

Показатель	Креатинин		Белок		Билирубин	
Группа пациентов	2	3	2	3	2	3
№ дня получения показателя	*Значение показателя*	*Значение показателя*	*Значение показателя*	*Значение показателя*	*Значение показателя*	*Значение показателя*
1	61,97,78,101 ...	64,104,69,49 ...	82,70,67,69 ...	70,64,80,74 ...	14.2,17.8,18.84,44.3 ...	19.5,16.8,8.6,19.5 ...
2	63,102,83,113 ...	71,108,71,32 ...	64,58,68,61 ...	55,57,54,62 ...	34.7,15.4,96.5,64.9 ...	36.8,19.5,32.4,73.9 ...
3	59,59,87,79 ...	71,110,75,51 ...	68,62,58,59 ...	55,65,70,65 ...	19.5,17.8,83.78,114.3 ...	24.9,12.3,15.8,30.3 ...

алгоритмами дискриминантного анализа и т.п. В Табл. 2 значения показателей могут совпадать друг с другом. Это не означает, что они дублируются при формировании гиперкуба, эти значения получены у разных пациентов. Следовательно, при формировании гиперкуба необходимо различать дублированные значения мер, полученные вследствие способа формирования представления, от совпадающих значений мер, соответствующих различным объектам базы данных.

При формировании представлений в примерах 1 и 2 использовались так называемые контексты и таблица соединения, определение и способ формирования которых рассматриваются далее. В данной статье предлагается не ограничивать пользователя в выборе структуры гиперкуба, а только подсказывать ему корректные решения по формированию гиперкуба при заданных мерах и размерностях.

2. Формирование контекстов

2.1. Контекст приложения

Пусть ***DEP*** – множество зависимостей на отношениях из ***R***. Рассмотрим базовые зависимости, используемые при проектировании схемы БД [8,9,13]. Пусть ***DEP*** – множество зависимостей – функциональных (ФЗ), многозначных (МЗ), включения (ВЗ), соединения (СЗ), определенных на множестве атрибутов ***U*** и множестве отношений $\mathfrak{R}$. Пусть ***R*** – отношение, определенное на множестве атрибутов ***U*** (универсальное реляционное отношение).

Определение 2.1 (ФЗ). Пусть ***X*** и ***Y*** – некоторые подмножества из множества атрибутов ***U***. Будем говорить, что ***X*** функционально определяет ***Y***: $X \to Y$, если в любой реализации ***R*** не могут присутствовать два кортежа $t,u \in R$, такие что $t[X]=u[X]$ и $t[Y] \neq u[Y]$.

Пусть заданы множества атрибутов $X \subseteq U$, $Y \subseteq U$ и $X \cap Y = \varnothing$, $Z = R \setminus (X \cup Y)$.

Определение 2.2 (МЗ). Множество ***X*** мультиопределяет множество ***Y*** в контексте ***Z***: $X \twoheadrightarrow Y(Z)$ (многозначная зависимость), если для произвольной реализации ***R*** существует два кортежа $t_1,t_2 \in R$ таких, что $t_1[X]=t_2[X]$, то существует кортеж t_3, для которого выполнено:

$$t_3[X]=t_1[X],\ t_3[Y]=t_1[Y],\ t_3[Z]=t_2[Z],$$

в силу симметрии существует кортеж t_4:

$$t_4[X]=t_1[X],\ t_4[Y]=t_2[Y],\ t_4[Z]=t_1[Z].$$

Определение 2.3 (ЗС). Отношение $R(V_1, V_2, ..., V_p)$ удовлетворяет зависимости соединения $*(V_1, V_2, ..., V_p)$ тогда и только тогда, когда ***R*** удовлетворяет свойству соединения без потерь информации (СБПИ):

$$R=R[V_1] \bowtie R[V_2] \bowtie \ldots \bowtie R[V_p],$$

где $\bowtie$ – операция естественного соединения, $R[V_i]$ – проекция отношения ***R*** по атрибутам V_i.

Заметим, что многозначная зависимость является частным случаем зависимости соединения, а функциональная зависимость является частным случаем многозначной зависимости [8,9].

Формальным основанием для установления связей на схеме базы данных являются зависимости включения [14]:

Определение 2.4 (ЗВ). Пусть $R_i[A_1, \ldots, A_m]$ и $R_j[B_1, ..., B_p]$ – схемы отношений (не обязательно различные), $V \subseteq \{A_1, \ldots, A_m\}$ и $W \subseteq \{B_1, ..., B_p\}$, $|V|=|W|$, тогда объект $R_i[V] \subseteq R_j[W]$ называется зависимостью включения, если $R_i[V] \subseteq R_j[W]$, где $|V|$ – мощность множества ***V***.

Если выполнено условие $V=W$, то такой вид ЗВ называется типизированными (typed) [15, 16]. Это дополнительное ограничение вполне согласуется с общепринятым свойством связей на схеме БД: связи отражают количественное соотнесение кортежей в отношениях и не обладают какой-либо семантикой. Необходимость связывания различных по смыслу атрибутов является следствием ошибок при проектировании схемы БД:

- потеряна какая-либо ФЗ для связываемых атрибутов,
- неверно определена семантика атрибутов, как в примере 1.1 [16], результирующее множество зависимостей содержит ЗВ, установленную для атрибутов-синонимов.

Далее связи на схеме БД будем считать типизированными.

Пусть $C=\{R_1, R_2, ..., R_m\}$ – произвольное подмножество отношений реляционной БД.

Определение 2.5. Зависимость $dep_j \in DEP$ будем считать реализованной на ***C***, если операция дополнения, удаления или модификации кортежа

в произвольном отношении $R_i \in C$ будет заблокирована организационно-техническими средствами при нарушении dep_j.

Под организационными средствами подразумевается способ проектирования схемы БД с указанием ограничений целостности на данные, под техническими – возможности системы управления базами данных (СУБД) по поддержке этих ограничений целостности.

Определение 2.6. Множество C будем называть контекстом, если оно удовлетворяет свойству СБПИ на зависимостях DEP, реализованных в C.

Замечание. В основе контекста лежит операция естественного соединения, которая собирает из различных отношений БД связанные друг с другом по значению данные. Затем эти данные (кортежи) участвуют в формировании новых структур, естественным образом дополняя и ограничивая друг друга, что делает уместным использование термина "контекст" для совокупности таких значений.

Первоначальный выбор размерностей и мер гиперкуба предлагается сделать в расширенном виде: $R_i.A_j$, где R_i – наименование отношения из исходной реляционной БД, и A_j – наименование атрибута в этом отношении. Таким образом будет задано начальное множество отношений $R^0=\{R^0_1, R^0_2, ..., R^0_q\}$, участвующее в обязательном порядке сначала в формировании таблицы соединения, а потом – гиперкуба.

Совокупность отношений, по которым строится гиперкуб, должна удовлетворять свойству СБПИ [17], поскольку лишние кортежи в промежуточном представлении данных дают лишние значения в рабочей области гиперкуба. Следовательно, дальнейшая задача состоит в дополнении множества R^0 отношениями из $\Re$, чтобы результирующее множество удовлетворяло свойству СБПИ на множестве зависимостей, то есть являлось контекстом. В общем случае таких вариантов дополнения существует несколько. Каждый из вариантов (контекстов) имеет свою смысловую нагрузку, поэтому окончательный выбор контекста может выполнить только пользователь. Задача алгоритма заключается в последовательной генерации контекстов без зацикливания. Для сокращения количества перебираемых вариантов при формировании контекстов, ближайших к множеству R^0, предлагается сделать этот перебор направленным.

Сформулируем критерии, которые позволят сделать перебор отношений направленным.

- Замыкание первичного ключа нового отношения R_i совпадает со всем множеством атрибутов в выбранных отношениях. Такое отношение получает приоритет 3.
- Для отношения R_i выполнено условие существования связи, соответствующей ЗВ $R_i[X] \subseteq R_j[X]$ с уже выбранными отношениями R_j, где множество атрибутов X является первичным ключом отношения R_j. Такое отношение получает приоритет 2.
- Дополняемое отношение R_i должно иметь непустое пересечение с уже выбранными отношениями. В противном случае в алгоритме проверки свойства СБПИ невозможно будет сделать ни одной подстановки. Такое отношение получает приоритет 1. Остальные отношения получают приоритет 0.
- Формируемый контекст не должен содержать лишних отношений, наличие которых обусловлено только порядком присоединения отношений к контексту в алгоритме.

Перечисленные критерии увеличивают вероятность более быстрого достижения результата.

Введем обозначения.

Пусть $R^1=\{R^1_1, R^1_2, ..., R^1_p\}$ – множество отношений, не входящих в исходное множество R^0: $R^1=\Re \setminus R^0$.

U^0 – множество атрибутов, на котором определены отношения из R^0: $U^0=[R^0_1] \cup [R^0_2] \cup \ldots \cup [R^0_q]$.

DEP^0 – множество зависимостей, реализованных на отношениях из R^0.

Рассмотрим схему алгоритма, удовлетворяющего сформулированным критериям.

1. Производится подсчет весов для отношений R^1, и их упорядочение по убыванию весов.

2. Формируются сочетания без повторений из отношений R^1, сначала по одному, затем по два и так далее. Сочетания начинаются с наименьших значенийи и далее последовательно увеличиваются, например, сочетания по два элемента: $(1,2)$, $(1,3)$,..., $(2,3)$.... Текущее сочетание отношений совместно с R^0 проверяется на выполнение свойства СБПИ. Если свойство

выполнено, то полученное сочетание дополняется к множеству контекстов.

3. В процессе выполнения алгоритма пользователю предлагается выбрать нужный контекст приложения.

Замечание. Такая схема алгоритма на ближайших итерациях находит дополнительные отношения, с наибольшей вероятностью образующие наименьший контекст с исходным множеством отношений R^0. В примерах 1 и 2 все отношения исходных баз данных являются контекстами приложений. Не трудно убедиться, что они удовлетворяют свойству СБПИ.

Далее контекст приложения будем обозначать C_0.

В существующих OLAP-системах, в том числе в "Microsoft Analysis Services" и "ORACLE Analytic Workspace Manager", свойство СБПИ не анализируется, что является основной причиной ошибок при формировании представлений гиперкубов.

Определение 2.7. Множество атрибутов KM_j будем называть ключом атрибута меры $R_i.A_j \in M$ в контексте C_0, если $KM_j \subseteq [C_0]$, зависимость $KM_j \rightarrow R_i.A_j$ выводима в FD^0, и не существует выводимой в FD^0 зависимости $Y \rightarrow R_i.A_j$, где $Y \subset KM_j$ и FD^0 множество реализованных в C_0 функциональных зависимостей. Пусть $KM = KM_1 \cup KM_2 \cup \ldots \cup KM_h$, где $h = |M|$.

После того как сформирована схема GC (с установлением иерархий в размерностях и присоединением мер к атрибутам одной из размерностей), простейшим решением задачи формирования гиперкуба по контексту приложения $C_0 = \{R^0_1, R^0_2, \ldots, R^0_m\}$ $(m \geq q)$ при $F = \varnothing$ является выполнение операции естественного соединения отношений для формирования промежуточного представления TJ:

$$TJ = R^0_1[V_1] \bowtie R^0_2[V_2] \bowtie \ldots \bowtie R^0_m[V_m], \quad (2.1)$$

где V_i – множество атрибутов $A_j \in [R^0_i]$, для которых выполнено: либо существует размерность D_l такая, что $R^0_i.A_j \in D_l$, либо $R^0_i.A_j \in M$, либо $R^0_i.A_j \in KM$; либо $R^0_i.A_j$ атрибут, который участвует в операции естественного соединения с другими отношениями контекста (определяется зависимостями соединения). Далее значения координат формируются в виде проекций по соответствующим атрибутам $TJ[D_l]$, с необходимой сортировкой кортежей каждой размерности в соответствии с иерархией. Завершается построение GC присваиванием значений мер M в рабочей области GC: для каждого кортежа $t \in TJ$ на пересечении значений координат $t[D_l]$, $l = 1,\ldots,d$, ставится значение $t[A_j]$, $R^0_i.A_j \in M$ (в данном случае R^0_i будет произвольным). Во всех остальных ячейках GC ставится значение NULL. Проблема дублированных значений в ячейках гиперкуба и остальные детали реализации технологии будут далее рассмотрены.

Предложенная процедура формирования GC не решает следующие проблемы:

- если какому-либо набору значений размерности не соответствует ни одного значения меры, то эти значения размерностей не появятся в реализации гиперкуба (по свойству операции естественного соединения). Однако отсутствие значений мер также является предметом анализа данных.

- ограничения на значения размерностей могут быть заданы опосредованно – через значения на связанные кортежи в отношениях, которые не входят в контекст приложения. Тогда эти отношения должны образовывать отдельный контекст с отношениями для размерностей.

- для некоторых размерностей необходимо иметь декартово произведение исходных отношений (все возможные комбинации атрибутов одной размерности). Тогда эти отношения не должны дополняться другими отношениями для получения контекста.

2.2. Управление размерностями

Резюмируя сформулированные требования к представлению размерностей в GC, получим, что в нем должны быть представлены следующие компоненты:

- контексты размерностей,
- псевдоконтексты.

Рассмотрим эти компоненты.

1. Отдельные контексты формируются для размерностей, которые должны быть связаны по значениям с множеством отношений, не совпадающим с C_0. Допустим, в примере 1 не для всех предметов заполнены сведения об учебной нагрузке, но в представлении требуется наличие всех предметов. Тогда размерность $D_1 = \{R_1.A_1\}$

должна быть сформирована по отдельному контексту $\{R_1\}$. В другом случае на размерность накладывается ограничение в виде логической формулы, в которую входят атрибуты отношений, не принадлежащих контексту C_0. В этом случае совокупность отношений, атрибуты которых выбраны в качестве компонентов размерности и компонентов логической формулы, дополняется новыми отношениями до необходимого контекста по алгоритму, схема которого предложена в предыдущем разделе.

2. Псевдоконтексты размерностей формируются аналогичным способом, за исключением того, что ограничения могут задаваться только на атрибутах уже выбранных отношений и это множество отношений не дополняется другими отношениями до контекста. В примерах 1 и 2 нет размерностей, сформированных по псевдоконтекстам. Однако в примере 1 альтернативное представление данных со схемой

$$\{R_1.A_1\{R_2.A_2\}\}\times\{R_3.A_5(R_3.A_4)\}$$

имеет размерность, сформированную по псевдоконтексту из отношений R_1 и R_2.

Если какая-либо размерность должна являться частью контекста приложения, то для этой размерности отдельный контекст не формируется. В представлении TJ, сформированном по формуле (2.1), все размерности являются частью контекста приложения.

Заметим, что для контекста приложения нельзя задавать отдельное ограничение, поскольку оно фактически будет являться ограничением на меры. Тогда в результирующем представлении будет содержаться либо пустая ячейка, либо ячейка с неполным списком значений. Это может неправильно воспринято пользователем - значения размерностей присутствуют в представлении, а соответствующего значения меры нет, тогда как это значение есть в БД, но оно "отфильтровано" ограничением.

Логическая формула для всего представления данных будет иметь вид: $F=F_1\wedge F_2\wedge\ldots\wedge F_d$, где формула F_l соответствует размерности D_l. Если для какой-либо размерности логическая формула не задана, то вместо нее в общую формулу подставляется значение *TRUE*.

Сформированное множество всех контекстов и псевдоконтекстов обозначим $C=\{C_0, C_1, \ldots, C_p\}$ (переменная p получила новое значение). Заметим, что дополнительные контексты и псевдоконтексты необходимы, прежде всего, для управления содержимым размерностей GC.

3. Формирование гиперкубового представления

Рассмотрим алгоритм формирования гиперкуба GC. Пусть размерности упорядочены так, что размерность D_s соответствует контексту C_s, $s=1, 2,\ldots, p$.

Шаги $s=1, 2,\ldots, p$. На каждом шаге формируется представление размерностей D'_s, для которых имеются отдельные контексты:

$$TJ_s=\sigma_{Fs}(R^s_1[V^s_1]\bowtie R^s_2[V^s_2]\bowtie\ldots\bowtie R^s_{m(s)}[V^s_{m(s)}])[D_s], \quad (3.1)$$

где $C_s=\{R^s_1, R^s_2,\ldots, R^s_{m(s)}\}$, атрибуты V^s_i отношений R^s_i выбираются по правилам, аналогичным формуле (2.1), $m(s)$ – элемент массива, содержащего количество отношений в каждом контексте. Каждое представление TJ_s целесообразно сразу преобразовать в иерархическое представление в соответствии с заданной схемой. В каждом узле иерархии будут содержаться значения атрибута текущего уровня, имеющие одно общее значение атрибута предыдущего уровня. Значения внутри узла иерархии сортируются, и каждое из них, кроме листьев, имеет указатели на узлы – потомки (потомков у одного значения может быть несколько). Значения атрибутов в листьях снабжаются пустыми указателями переменной длины, которые позднее будут заполнены ссылками на ячейки со списками значений атрибутов мер. При формировании иерархии размерностей неопределенные значения атрибутов будем считать совпадающими, Это позволит объединить значения атрибутов мер, если соответствующие им значения размерностей являются неопределенными. Например, в иерархии адресов {Страна{Область{Город{Район {Населенный пункт}}}}} для жителей города атрибут 'Населенный пункт' будет неопределен, а для сельских жителей неопределенным будет атрибут 'Город'. Если считать неопределенные значения атрибутов различными, то значения атрибутов мер для жителей одного района города будут распределены по различным ячейкам, аналогичная ситуация будет и для жителей од-

ного населенного пункта в сельской местности. Обе эти ситуации являются ошибкой.

Шаг *p+1.* На этом шаге формируется представление контекста приложения:

$$TJ=\sigma_F(R^0_1[V^0_1]\bowtie R^0_2[V^0_2]\bowtie\ldots\bowtie R^0_{m(0)}[V^0_{m(0)}]), \quad (3.1)$$

где $C_0=\{R^0_1, R^0_2,\ldots, R^0_{m(0)}\}$, атрибуты V^0_i отношений R^0_i выбираются по правилам, аналогичным формуле (2.1), ***m(0)*** –количество отношений в контексте приложения ($m(0)=m$). Выражения F_l в формуле F, в которых есть неопределенные элементарные условия, заменяются значением *TRUE*. Следовательно, формула F задает более слабые ограничения, чем на предыдущих шагах, и в представлении TJ будет больше кортежей, чем это необходимо для сопоставления со значениями сформированных размерностей. Если $p=0$, то в представлении TJ нет лишних кортежей.

Шаг *p+2.* Для каждого кортежа $t\in TJ$ проверяется наличие соответствующих значений в иерархиях размерностей (идентифицирующий путь), что аналогично условию: существует кортеж $u\in TJ_s$ такой, что $t[D'_s]=u[D'_s]$, $s=1, 2,\ldots, p$. Если это условие выполнено, то кортеж t используется для формирования иерархий D'_s, $s=p+1,\ldots, d$, для которых отсутствуют отдельные контексты. Если соответствующий путь отсутствует в иерархии, то он формируется по правилам, аналогичным предыдущим шагам.

Каждое значение атрибута $A_j\in M$ текущего кортежа t дополняется к соответствующей ячейке со списком, если оно не дублировано. Ячейка существует, если все идентифицированные пути в иерархиях имеют совпадающий указатель на нее. В противном случае создается новая ячейка, в нее дополняется значение атрибута A_j и каждому идентифицированному пути в иерархиях дополняется указатель на эту ячейку.

Конец алгоритма.

Для завершения обсуждения алгоритма необходимо определить признак дублирования значения атрибута меры в ячейке. Отражением семантики сформированного контекста приложения является следующее определение.

Определение 3.1. Значение атрибута меры $t[A_j]$, где $A_j\in M$, для текущего кортежа $t\in TJ$ дублирует значение $t'[A_j]$, $t'\in TJ$, если:

1) $t[A_j]=t'[A_j]$,
2) $t[D_s]=t'[D_s]$, $s=1, 2,\ldots, d$,
3) $t[KM_j]=t'[KM_j]$.

Замечание 1. Условия 1 и 2 определения 3.1 проверяются непосредственно по текущим значениям кортежа t, размерностей D_s и значениям атрибута A_j, уже записанным в текущую ячейку. Для проверки условия 3 необходимо знать значения ключевых атрибутов KM_j ранее просмотренного кортежа t'. Наиболее экономичным и технологичным решением данной проблемы является временное хранение номера кортежа t' совместно со значением атрибута A_j в ячейке. Тогда условие 3 можно проверить, сравнивая два кортежа в представлении TJ. Альтернативным решением этой задачи является хранение номера кортежа исходного отношения, который использовался для формирования TJ, однако стандартный доступ к данным с помощью языка SQL такой возможности не предоставляет.

Замечание 2. Если в контексте приложения для атрибута меры A_j отсутствует ключ, то по аксиоме рефлексивности [8] ключом будет сам атрибут. Следовательно, в ячейках для этого атрибута не может быть совпадающих значений.

Заключение

Предложенная модель многомерных данных является обобщением известных моделей, прежде всего за счет снятия ограничения (1.1). Технология ориентирована на работу аналитика, где не требуется быстрая (за доли секунд) реакция системы на запросы, поскольку в большинстве случаев аналитик должен вдумчиво выполнить различные виды анализа над различными представлениями данных с участием ИТ-специалиста. Технология предназначена, прежде всего, для ИТ-специалиста, чтобы он мог за приемлемое время подготовить для обработки необходимое представление данных. Хотя, после некоторого обучения, аналитик самостоятельно может освоить данную технологию.

Основным методологическим принципом в данной работе является то, что операционная база данных должна удовлетворять принципам независимости, неизбыточности, непротиворечивости и т.д. Эта база данных является ядром приложений для множества пользователей, а не

только отдельно взятого аналитика. Автору данной статьи неоднократно приходилось принимать участие в обработке медицинской информации. Этот опыт показывает, что основное время тратится на реорганизацию одних и тех же данных для их последующей обработки тем или иным алгоритмом анализа. Наибольшей эффективности в работе аналитика можно достичь, если выполнены следующие принципы:

1. операционная база данных нормализована и не зависит от конкретных приложений;

2. имеется эффективный инструмент для формирования различных пользовательских представлений данных;

3. имеется функционально развитый пакет программ для аналитической обработки данных, интерфейс которого согласован с инструментом пункта 2.

Литература

1. Vassiliadis P., Sellis T. A survey of logical models for OLAP databases// SIGMOD Rec., V. 28, No 4, 1999. P. 64-69.
2. Pedersen T.B., Jensen C.S., Dyreson C.E. A foundation for capturing and querying complex multidimensional data// Inf. Syst., V. 26, No. 5, 2001. P. 383-423.
3. Lechtenborger J., Vossen G. Multidimensional normal forms for data warehouse design// Inf. Syst., V. 28, No. 5, 2003. P. 415-434.
4. Li H.-G., Yu H., Agrawal D., Abbadi A.E. Progressive ranking of range aggregates// Data & Knowledge Engineering, V. 63, No. 1, 2007. P. 4-25.
5. Lehner W., Albrecht J., Wedekind H. Normal forms for multidimensional databases// Proceedings of the Tenth International Conference on Scientific and Statistical Database Management, 1998. P. 63-72.
6. Giorgini, P., Rizzi, S., Garzetti, M. Goal-oriented requirement analysis for data warehouse design// In Proceedings of the 8th ACM international Workshop on Data Warehousing and OLAP: DOLAP '05, 2005. P. 47-56.
7. Mazon J., Trujillo J., Lechtenborger J. Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms. Data Knowl. Eng. V. 63, No. 3. 2007. P. 725-751.
8. Ульман Дж. Основы систем баз данных// - М.: Финансы и статистика, 1983. - 334 с.
9. Мейер Д. Теория реляционных баз данных// - М.: Мир, 1987. - 608 с.
10. Зыкин С.В. Формирование гиперкубического представления реляционной базы данных // Программирование. - 2006. - № 6. - С. 348 - 354.
11. Кукин А.В., Зыкин С.В. Построение математической модели учебного процесса для долгосрочного планирования// Омск: ОмГУ. В сб. Математические структуры и моделирование, 2002, Вып. 10. С. - 77-86.
12. Полуянов А.Н., Зыкин С.В. Технология подготовки и анализа данных для построения шкалы оценки печеночной недостаточности// Материалы Всероссийской конференции с международным участием «Знания –Онтологии – Теории» (ЗОНТ–09), Новосибирск, 2009, Т. 1. С. - 236-242.
13. Дейт К. Введение в системы баз данных. – М.: Диалектика, 1998. – 782 с.
14. Casanova M., Fagin R., Papadimitriou C. Inclusion Dependencies and Their Interaction with Functional Dependencies// Journal of Computer and System Sciences. 1984. No 28(1). P. 29 - 59.
15. Missaoui R., Godin R. The Implication Problem for Inclusion Dependencies: A Graph Approach// SIGMOD Record. 1990. V 19, No 1. P. 36 - 40.
16. Levene M., Vincent M.W. Justification for Inclusion Dependency Normal Form// IEEE Transactions on Knowledge and Data Engineering. 2000. V 12, No 2. P. 281 - 291.
17. Miller L., Nila S. Data Warehouse Modeler: A CASE Tool for Warehouse Design// Thirty-First Annual Hawaii International Conference on System Sciences. 1998. V 6. P 42 - 48.

Зыкин Сергей Владимирович. Заведующий лабораторией Омского филиала Института математики СО РАН. Окончил Алтайский политехнический институт в 1981 году. Доктор технических наук, профессор. Автор 51 печатной работы. Область научных интересов: теория проектирования баз данных, межмодельные отображения в базах данных, технологии аналитической обработки данных. E-mail: zykin@ofim.oscsbras.ru.