УДК 004.4'22

Автоматизация формирования табличных приложений

Зыкин С. В.¹

*Федеральное государственное бюджетное учреждение науки Институт математики
им. С.Л. Соболева Сибирского Отделения Российской академии наук
630090 Россия, г. Новосибирск, просп. Акад. Коптюга, 4*

e-mail: szykin@mail.ru

получена 20 декабря 2012

Ключевые слова: табличные приложения, реляционная база данных, аналитическая обработка данных, соединение без потерь

Рассмотрены проблемы автоматизации формирования интерфейса между таблицей и реляционной базой данных. Формализована постановка задачи и представлено описание существующих подходов к формированию представлений данных на примере широко распространенных CASE-инструментов. Предложено определение промежуточного представления данных в виде таблицы соединений, которая используется для обеспечения корректности формирования представления данных и также необходима для прямого и обратного преобразования данных. На основе свойства соединения без потери информации и реализованных зависимостей вводится понятие и способ формирования контекстов приложения и ограничений на данные. Рассмотренный материал далее используется для построения обратного преобразования данных из табличного представления в реляционное. На основе использования свойств связей на схеме базы данных устанавливается частичный порядок над отношениями, вводится ограничение ацикличности на допустимые схемы баз данных. Полученные результаты далее используются при анализе принципов формирования обратного преобразования данных, рассмотрена схема алгоритма такого преобразования.

Введение

Формирование табличных представлений данных используется во множестве приложений, работающих с базами данных (БД). Наиболее популярными в настоящее время является технология оперативной аналитической обработки данных OLAP (online analytical processing). В основе OLAP-технологии лежит построение гиперкубического (многомерная таблица) представления данных. В большинстве случаев предполагается, что такое представление данных должно быть постоянно хранимым

¹Работа выполнена при поддержке Минобрнауки РФ, соглашение 14.B37.21.1123

и периодически обновляемым из БД (технология MOLAP). Основным аргументом в пользу такого дублирования данных выдвигается требование минимального времени отклика системы на запросы. Возможности пользователей в анализе данных при этом ограничиваются сформированным представлением. Другой подход заключается в преобразовании схемы исходной БД в “звезду” или “снежинку” (технология ROLAP), что позволяет динамически формировать представления данных. Однако при этом нарушается принцип независимости данных, в частности независимость схемы БД от места и способа использования данных.

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 большинстве работ предполагается постоянное хранение гиперкубических представлений для обеспечения наибольшей скорости обработки данных. При этом считается, что схемы гиперкубов являются неизменными и заранее хорошо продуманы.

В данной статье предполагается, что основой работы пользователя является необходимость формирования новых представлений данных из исходной реляционной БД. Это необходимо как при проведении визуального анализа, так и при выявлении скрытых закономерностей в данных с использованием статистического, дискриминантного и тому подобных видов анализа. Кроме того, сформированное представление может быть использовано для редактирования данных, что является более удобным, чем редактирование в БД. Предусмотреть все эти функции при проектировании складов данных не реально. В этом случае основное внимание необходимо акцентировать на сокращении времени формирования схемы нового представления данных и его автоматическом формировании по технологии, представленной в данной работе.

Далее будем предполагать следующую последовательность обработки данных:

1. Исходные данные должны быть представлены в реляционном нормализованном виде, и к ним обеспечивается доступ по технологии OLTP (online transaction processing);
2. Пользовательское представление данных в виде таблиц, реализующее технологию OLAP, обеспечивается инструментарием, преобразующим исходные данные в необходимую на данный момент таблицу.
3. Табличное представление данных далее используется в специализированном пакете программ для классификации, кластеризации, прогнозирования и т.д. Кроме того, оно используется для дополнения новых данных, необходимых для проведения дальнейшего анализа.

1. Постановка задачи

Пусть задана БД $\mathfrak{R} = \{R_1, R_2, \dots, R_k\}$, полученная в результате нормализации отношений [8, 9]. Отношения R_i заданы на множестве атрибутов $U = \{A_1, A_2, \dots, A_n\}$. Пусть $[R_i]$ – схема отношения, множество атрибутов, на которых определено отно-

шение R_i . Предположим, что 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 $t \in R_i$.

Для демонстрации предлагаемых решений рассмотрим пример БД.

Пример 1. Пусть множество U задано следующим набором атрибутов: A_1 – ФИО студента, A_2 – Предмет, A_3 – Код группы, A_4 – Специальность, A_5 – Количество часов, A_6 – Оценка, A_7 – № студента, A_8 – № группы, A_9 – № предмета, A_{10} – № специальности, A_{11} – Занятия, A_{12} – Семестр, A_{13} – Балл аттестации, A_{14} – ФИО преподавателя, A_{15} – № аудитории, A_{16} – День недели, A_{17} – Время начала занятия, A_{18} – № преподавателя, A_{19} – № дня недели, A_{20} – № занятия, A_{21} – Вид аттестации. Схема БД $\mathfrak{R}$ имеет следующий вид:

Студенты = $R_1(\underline{A_7}, \underline{A_8}, A_1)$,

Список групп = $R_2(\underline{A_8}, A_3, A_{10})$,

Предметы = $R_3(\underline{A_9}, A_2)$,

Специальности = $R_4(\underline{A_{10}}, A_4)$,

Учебная нагрузка = $R_5(\underline{A_9}, \underline{A_{10}}, \underline{A_{11}}, A_{12}, A_5)$,

Оценки = $R_6(\underline{A_7}, \underline{A_8}, \underline{A_9}, A_6)$,

Аттестация = $R_7(\underline{A_7}, \underline{A_8}, \underline{A_9}, \underline{A_{21}}, A_{13})$,

Преподаватели = $R_8(\underline{A_{18}}, A_{14})$,

Неделя = $R_9(\underline{A_{19}}, A_{16})$,

Преподаватели = $R_{10}(\underline{A_{20}}, A_{17})$,

Расписание = $R_{11}(\underline{A_8}, \underline{A_{19}}, \underline{A_{20}}, A_9, A_{15}, A_{18})$,

где подчеркнуты атрибуты первичных ключей отношений.

На рисунке 1 показаны связи между отношениями на схеме БД типа “один ко многим” (стрелка находится со стороны “ко многим”). Каждая связь имеет обозначение в виде набора атрибутов, по которым она установлена. На рисунке 1 показаны все связи, реализующие ссылочные ограничения целостности (внешние ключи). В рассматриваемой технологии имеется ограничение: множество отношений, по которым строится представление данных, не должно быть связано циклически (существует частичный порядок). Поскольку связи на схеме являются направленными, то на схеме БД в примере 1 нет ни одного множества отношений, которые были бы связаны циклически. Самый простой пример циклических связей дает множество отношений, для атрибутов которых установлены циклические функциональные зависимости, что на практике встречается крайне редко. Поэтому указанное ограничение не оказывает существенного влияния на применимость рассматриваемой технологии.

Для формирования табличного представления данных определим множества атрибутов: X – общая размерность (располагается по строкам), $Y_1, Y_2, \dots, Y_N$ – отдельные размерности (все располагаются по столбцам). В рабочей области таблицы располагаются меры $Z_1, Z_2, \dots, Z_N$. Мера Z_1 соответствует размерностям (X, Y_1) , мера Z_2 соответствует размерностям (X, Y_2) и т.д. Естественными являются ограничения: $X \cap Y_i = \emptyset$, $(X \cup Y_i) \cap Z_i = \emptyset$, $i = \overline{1, N}$. Такое представление данных будем называть композиционной таблицей CT : композиция размерностей Y_i . В таблице 1 представлен пример композиционной таблицы, где полужирным шрифтом представлены атрибуты размерностей, атрибуты мер – курсивом, значения атрибутов – обычным шрифтом.

Рис.1. Связи на схеме БД

Таблица 1. Фрагмент учебного плана

Семестр	2					
Предмет	Математика			Физика		
Занятия	Лекции	Практ.	Контроль	Лекции	Практ.	Контроль
Специальность	Кол-во часов	Кол-во часов	успеваемости	Кол-во часов	Кол-во часов	успеваемости
История	36	36	зач.	18	18	зач.
Филология	18	18	зач.	36	36	экз.
Право	48	48	экз.	48	24	зач.

Схема данных в таблице 1 следующая: $X = \{\text{Специальность}\}$, $Y_1 = \{\text{Семестр, Предмет, Занятия}\}$, $Y_2 = \{\text{Семестр, Предмет}\}$. Общие атрибуты в размерностях Y_1 и Y_2 объединены в представлении таблицы. Однако каждая из размерностей имеет собственную сопоставленную меру: размерностям (X, Y_1) сопоставлена мера $\{\text{Кол-во часов}\}$, а размерностям (X, Y_2) – мера $\{\text{Контроль успеваемости}\}$. В литературе, посвященной моделированию гиперкубов [3, 5, 7, 10], все меры сопоставлены всем размерностям, что вполне приемлемо для представления гиперкуба в целом и выполнения над ним операций типа свертки. В примере если атрибут $\{\text{Кол-во часов}\}$ в

кортеже гиперкуба имеет определенное значение, то атрибут {Контроль успеваемости} – неопределенное и наоборот. Эти неопределенности будут присутствовать при построении проекции гиперкуба на плоскость, что существенно затрудняет визуальный анализ данных и требует специальных действий с неопределенностями при выборке данных для последующего глубокого анализа. Следовательно, представление данных в виде композиционной таблицы является более предпочтительным, кроме того, оно не нуждается в постоянном хранении, а может быть сформировано динамически из БД в том виде, в котором это требуется для текущего анализа данных. Однако проблема корректности представления данных является актуальной как для гиперкубического представления, так и для композиционной таблицы и требует отдельного решения.

Для демонстрации преимущества предлагаемого подхода рассмотрим еще один пример представления данных (таблица 2).

Таблица 2. Расписание занятий

Код группы		М-210			М-220		
День недели	Время начала занятия	Предмет	ФИО преподавателя	№ аудитории	Предмет	ФИО преподавателя	№ аудитории
Понедельник	8:00	Физика	Уваров В.А.	1-130			
	9:45	Алгебра	Чернов О.Г.	1-130	Физика	Уваров В.А.	1-340
	11:30						
	...						
Вторник	8:00						
	9:45				Алгебра	Чернов О.Г.	1-130
	11:30						
	...						

Раздельное формирование размерностей в таблице 2 позволяет показать отсутствующие значения мер, которые могут быть изменены и записаны в БД (шаг 9 технологии далее будет рассмотрен). Если размерности формировать совместно с мерами, то значения размерностей, которым не соответствует ни одного значения меры, исчезнут из представления. Заметим, что отсутствие каких-либо данных в БД само по себе является предметом предварительного анализа.

Таким образом, целью данной статьи является описание реализации технологии корректного преобразования данных из реляционной БД в композиционную таблицу и обратного преобразования после редактирования данных в композиционной таблице:

$$\mathfrak{R} \Leftrightarrow CT$$

2. Описание технологии

Для автоматизации построения композиционной таблицы предлагается следующая последовательность формирования ее представления и работы с ней:

Шаг 1. Пользователь из списка атрибутов БД формирует множества атрибутов: размерности $X, Y_1, Y_2, \dots, Y_N$ и соответствующие им меры $Z_1, Z_2, \dots, Z_N$.

Шаг 2. Формирование иерархий размерностей для множеств атрибутов $X, Y_1, Y_2, \dots, Y_N$.

Шаг 3. По шаблону задаются логические ограничения на размерности F_0 для X , F_1 для Y_1 , и так далее, F_N для Y_N . В ограничении F_i могут присутствовать атрибуты отношений, в которых нет атрибутов размерности. В этом случае все отношения для размерности должны образовывать контекст, а ограничение F_i будет контекстным.

Шаг 4. Формирование контекстов и псевдоконтекстов размерностей $C_0, C_1, \dots, C_N$. Некоторые из них могут быть объявлены пустыми. Формирование контекста приложения C^* . Контекст приложения содержит в себе контексты размерностей и отношения, в которых содержатся атрибуты мер.

Шаг 5. Формирование реализации c^* контекста приложения C^* , содержащего отношения контекстов размерностей и отношения, атрибуты которых содержатся в логических формулах F_i .

Шаг 6. Формирование реализаций c_i контекстов и псевдоконтекстов C_i для размерностей $X, Y_1, Y_2, \dots, Y_N$ с сортировкой кортежей в соответствии с иерархией.

Шаг 7. Формирование реализации (представления) композиционной таблицы CT (заполнение значений мер на соответствующих местах таблицы).

Шаг 8. Анализ данных в представлении композиционной таблицы и их редактирование при необходимости.

Шаг 9. Запись отредактированных данных в БД.

Пользователь вручную выполняет шаги 1, 3 и 8, и осуществляет выбор предложенных вариантов в шагах 2, 4 и 5. Все остальные операции выполняются автоматически. Заметим, что в предложенной последовательности шагов формирования композиционной таблицы отсутствует необходимость каким-либо образом модифицировать схему исходной БД, что делает возможным реализовать все принципы проектирования БД [8, 9], в том числе самый важный – принцип независимости данных.

Рассмотрим важные детали реализации перечисленных шагов технологии.

Выбор атрибутов на первом шаге осуществляется за счет вызова системных функций исполняющей среды: получить список таблиц БД и получить список атрибутов таблицы. При этом пользователю сначала предлагается выбрать таблицу, затем атрибут из нее. Таким образом, множества X, Y_i и Z_i задаются в виде списков расширенных имен атрибутов: $R_i.A_j$.

На шаге 2 иерархии формируются автоматически по правилам, рассмотренным в работе [11], и пользователю предлагается только их модифицировать при необходимости. Основная идея состоит в том, что на верхние уровни иерархии помещаются атрибуты с меньшим количеством значений, что снижает количество дублированных значений на нижних уровнях иерархии. При этом используются свойства функциональных и многозначных зависимостей между атрибутами иерархии. Такой подход предпочтительнее построений иерархий, основанных только на функ-

циональных зависимостях [3, 5, 7], поскольку делает представление данных более компактным (за счет меньшего количества дублированных значений в иерархии и, как следствие, пустых ячеек для мер в рабочей области) и выдает результат даже в случае отсутствия каких-либо зависимостей между атрибутами иерархии. Однако в [11] не предусмотрено объединение верхних уровней иерархий для размерностей Y_i .

Для формирования логических ограничений на размерности в шаге 3 используются традиционные шаблоны. Сначала формируются конъюнктивные ограничения на атрибуты размерностей в виде элементарных предикатов, аналогичных предикатам языка SQL. Для каждого атрибута в предикатах указывается, допустимо или нет наличие в нем неопределенного значения. Это дополнительное условие позволяет избавиться от значений UNKNOWN: если допускается неопределенное значение для атрибута, то соответствующий элементарный предикат заменяется значением TRUE, если в текущем кортеже атрибут имеет значение NULL, и заменяется значением FALSE в противном случае. Далее формируются дизъюнкции из конъюнктивных групп, что в результате служит логическим ограничением на размерность.

Технологическим ограничением на шаге 3 является запрет на использование атрибута в логическом выражении, если он использован в качестве меры. Основанием для такого запрета является возможность наличия значений размерностей в представлении композиционной таблицы и отсутствия соответствующих значений мер, хотя они есть в БД, но соответствующие кортежи отфильтрованы при формировании представления. Это может служить причиной для неверного анализа результатов.

Исследование контекстов при формировании многомерного представления данных рассмотрено в работе [12]. Кратко рассмотрим понятие контекста, поскольку контексты позволяют управлять содержимым размерностей и, как следствие, содержимым самой композиционной таблицы.

Пусть $C = \{R'_1, R'_2, \dots, R'_m\}$ – произвольное подмножество отношений реляционной БД и DEP множество реализованных зависимостей в БД.

Определение 1. 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 зависимости 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. Функциональные зависимости реализуются первичным ключом и уникальным индексом по набору атрибутов в отношении. Зависимости включения – внешним ключом. Многозначные зависимости и зависимости соединения реализуются за счет декомпозиции при проектировании схемы БД.

Определение 2. Множество C будем называть контекстом, если оно удовлетворяет свойству соединения без потери информации (СБПИ) [8] 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 выполняется в расширенном виде: $R_i.A_j$, где R_i – наименование отношения, A_j – наименование атрибута в этом отношении. Таким образом будет задано начальное множество отношений для контекста $R^0 = \{R_1^0, R_2^0, \dots, R_m^0\}$.

На шаге 4 совокупность отношений C^* должна удовлетворять свойству СБПИ [13], поскольку лишние кортежи в промежуточном представлении данных дают лишние значения в рабочей области композиционной таблицы. Следовательно, дальнейшая задача состоит в дополнении множества R^0 отношениями из $\mathfrak{A}$, 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 заикливания. Для сокращения количества перебираемых вариантов при формировании контекстов, ближайших к множеству R^0 , предлагается сделать этот перебор направленным. Кроме того, расширение до контекста некоторых C_i позволит использовать ограничение операции естественного соединения. Например, в сводной ведомости для учебной группы студентов надо показать только те предметы, которые изучались в этой группе, а не все предметы из БД. Дополнение к C_i отношения с учебной нагрузкой решает эту проблему. Использование только тех значений в размерностях, которым сопоставлены меры, в данном случае оценки, может дать неверный результат: если по какому-либо предмету пока отсутствуют оценки, то этот предмет исчезнет из результирующего представления.

В работе [12] рассмотрены критерии, по которым упорядочиваются отношения, не вошедшие в R^0 , и предложен алгоритм направленного перебора отношений, который на ближайших итерациях находит дополнительные отношения, с наибольшей вероятностью образующие наименьший контекст с исходным множеством отношений R^0 . При этом последовательно будут сгенерированы все допустимые контексты, пользователю надо будет только выбрать подходящий контекст.

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 3. Множество C' будем называть псевдоконтекстом, если оно используется только для размерностей и для него не требуется выполнение свойства СБПИ.

Замечание. Псевдоконтексты формируются пользователем вручную, ограничения для них задаются только на атрибутах отношений C' и они не дополняются другими отношениями до контекста. Использование псевдоконтекста в размерности дает все допустимые комбинации значений атрибутов, в том числе те, которым не сопоставлены значения мер. Пустые ячейки в рабочей области таблицы, появившиеся за счет использования контекста или псевдоконтекста в размерности, могут быть использованы для ввода новых значений мер.

Формирование реализации контекста приложения c^* и реализаций размерностей

c_i на шагах 5 и 6 осуществляется по следующему правилу:

$$c^* = \sigma_{F^*}(R_1^*[V_1^*] \bowtie R_2^*[V_2^*] \bowtie \dots \bowtie R_s^*[V_s^*]), \quad (1)$$

$$c_i = \sigma_{F_i}(R_1^i[V_1^i] \bowtie R_2^i[V_2^i] \bowtie \dots \bowtie R_{m(i)}^i[V_{m(i)}^i]), \quad (2)$$

где $C_i = \{R_1^i, R_2^i, \dots, R_{m(i)}^i\}$ в (2) контексты и псевдоконтексты, $i = \overline{0, N}$, а $C^* = \{R_1^*, R_2^*, \dots, R_s^*\}$ в (1) контекст, V_j^i – минимально необходимый набор атрибутов отношения R_j^i для выполнения операции естественного соединения ($\bowtie$) и для формирования размерностей и мер, $m(i)$ – количество отношений в контексте, σ_F – операция селекции.

Некоторые из C_i могут быть предварительно объявлены пустыми. Тогда для них представление формируется в виде проекции от c^* . В этом случае в соответствующей размерности будут присутствовать только те значения, которым сопоставлены значения мер.

С учетом структуры композиционной таблицы очевидно, что логическое ограничение на кортежи из c^* в (1) имеет следующий вид:

$$F^* = F_0 \wedge (F_1 \vee F_2 \vee \dots \vee F_N).$$

После того, как сформированы таблицы c^* и c_i , в композиционную таблицу CT на шаге 7 сначала загружаются размерности c_i . Затем последовательно просматриваются кортежи таблицы c^* , содержащие размерности и меры, и значения мер из текущего кортежа расставляются в ячейки рабочей области CT , соответствующие значениям размерностей в текущем кортеже.

Ограничения на редактирование данных в шаге 8 определяются возможностями алгоритма в шаге 9. В основном они касаются ключевых атрибутов $\mathfrak{K}$. Неключевые атрибуты редактируются без ограничений: дополнение, удаление, модификация значений, вставка новых строк и столбцов в композиционной таблице.

Наиболее сложным в алгоритмическом плане является шаг 9. В работе [14] рассмотрено условие ацикличности для совокупности отношений, основанное на ссылочных ограничениях целостности (связях), приведено сравнение с условием ацикличности ассоциированного гиперграфа [9]. Показано, что условие ацикличности на связях является более слабым. Кроме того, разработаны алгоритмы построения связей и удаления избыточных связей. Выполнение условия ацикличности позволяет установить частичный порядок для отношений контекста приложения C^* . В работе [15] представлен алгоритм редактирования данных в БД с использованием мультитабличных представлений, условием корректного выполнения которого является наличие частичного порядка на отношениях, из которых сформирована мультитаблица. Этот алгоритм положен в основу шага 9.

Идея алгоритма заключается в последовательном просмотре отношений контекста C^* , начиная с минимальных до финальных отношений в соответствии с частичным порядком. При этом каждому измененному значению ставится в соответствие два кортежа: $t_1 \in c^*$, из которого было сформировано это значение, и t_2 – новый кортеж, сформированный по значениям атрибутов X , Y_i и Z_i , с проверкой условия $F(t_2) \neq \text{FALSE}$. Значения некоторых атрибутов в t_2 будут не определены. Кортеж t_1 будет отсутствовать, если были добавлены новая строка или столбец в CT . При

последовательном передвижении от минимальных отношений к финальным, в соответствии с частичным порядком, будут восстановлены неопределенные значения в t_2 и заменены старые значения из кортежа t_1 на новые значения из кортежа t_2 .

3. Пример реализации технологии

Рассмотрим порядок формирования представления данных на примере “Расписания занятий” (таблица 2).

На первом шаге вручную определяется набор атрибутов, образующих меры и размерности: $X = \{R_9.A_{16}, R_{10}.A_{17}\}$, $Y_1 = \{R_2.A_3\}$, $Z_1 = \{R_3.A_2, R_8.A_{14}, R_{11}.A_{15}\}$.

На втором шаге автоматически определяется иерархия размерности X . Поскольку обычно количество занятий в день больше, чем дней в неделе, то атрибут $R_9.A_{16}$ автоматически становится корнем иерархии, а атрибут $R_{10}.A_{17}$ его потомком. Если в учебном заведении обратная ситуация, то пользователю надо будет вручную переставить эти атрибуты.

На третьем шаге для сформированных размерностей задаются ограничения. Одно ограничение соответствует одной размерности. Формирование ограничения начинается с выбора атрибута, как и на первом шаге. Затем предлагается выбрать операцию сравнения: больше, меньше, равно и т.д. для числовых атрибутов и даты. Для символьных атрибутов выбираются свои операции: “содержит”, “начинается с” и т.д. Затем вводится значение для выполнения операции сравнения. Вместо значения можно ввести знак “?”, тогда значение будет запрошено при активации готового приложения. Далее для атрибута надо определить возможность наличия неопределенного значения. Затем у пользователя запрашивается необходимость ввода дополнительного или альтернативного условия, которое определяется аналогичным способом.

На четвертом шаге формируются контексты. Размерности X соответствует начальное множество отношений: $\{R_9, R_{10}\}$. Пользователь указывает, что это множество является псевдоконтекстом (в представлении нужны все комбинации значений этих атрибутов) и далее оно не дополняется другими отношениями до контекста. Для размерности Y_1 начальным множеством отношений является $\{R_2\}$, оно удовлетворяет свойству СБПИ и не требует дополнения других отношений. Хотя пользователь может дополнить к множеству отношение R_1 , что позволит иметь в расписании только те группы, в которых не менее одного студента. Формирование контекста приложения начинается с объединения контекстов размерностей. Далее к нему дополняются отношения, содержащие атрибуты мер. В результате для таблицы 2 будет получено множество: $\{R_2, R_3, R_8, R_9, R_{10}, R_{11}\}$. Поскольку отношение R_{11} содержит суперключ для выделенных отношений, то свойство СБПИ автоматически выполнено и это множество является контекстом (дополнять его другими отношениями не надо).

Шаги 5–7 выполняются автоматически с использованием формул (1) и (2). В результате будет сформировано представление, аналогичное таблице 2.

Допустим, что на шаге 8 пользователь заменил занятие у группы М-210 в понедельник в 8:00 на предмет “История”, который ведет преподаватель “Орлов А.С.”. Выбором из соответствующих списков заменяется предмет “Физика” на предмет “История” и преподаватель “Уваров В.А.” заменяется на преподавателя “Орлов А.С.”.

Последовательность отношений, в соответствии с частичным порядком, будет следующей: $R_2, R_3, R_8, R_9, R_{10}, R_{11}$, в которой финальным отношением будет R_{11} . Схема (заголовок) промежуточного представления:

$c^* = (A_2, A_3, A_{16}, A_{17}, A_{15}, A_{14}, A_8, A_9, A_{18}, A_{19}, A_{20})$.

Соответствующий кортеж t_1 будет иметь вид:

$t_1 = (\text{Физика}, \text{М-210}, \text{Понедельник}, 8:00, 1-130, \text{Уваров В.А.}, 1, 2, 1, 1, 1)$.

Начальные значения в кортеже t_2 будут следующими:

$t_2 = (\text{История}, \text{М-210}, \text{Понедельник}, 8:00, 1-130, \text{Орлов А.С.}, -, -, -, -, -)$.

После поиска в отношении R_2 кортеж t_2 будет иметь вид:

$t_2 = (\text{История}, \text{М-210}, \text{Понедельник}, 8:00, 1-130, \text{Орлов А.С.}, 1, -, -, -, -)$.

Последовательно просматривая отношения R_3, R_8, R_9, R_{10} , будем иметь:

$t_2 = (\text{История}, \text{М-210}, \text{Понедельник}, 8:00, 1-130, \text{Орлов А.С.}, 1, 3, 1, 1, 2)$.

Для завершения операции в отношении R_{11} соответствующие значения кортежа t_1 заменяем на значения кортежа t_2 .

4. Заключение

Разработанные алгоритмы реализованы в виде программного обеспечения с использованием средств Microsoft Office. На встроенном языке Access разработаны две служебные формы: Мастер-форма и Шаблон-форма, которые могут импортироваться в любую БД, созданную посредством Access. В Мастер-форме реализованы функции настройки приложения, а в Шаблон-форме реализованы универсальные алгоритмы построения преобразований, рассмотренные ранее.

Настройка приложения начинается с того, что в существующую БД Access импортируются две формы: Мастер-форма и Шаблон-форма. Затем активизируется Мастер-форма и задаются необходимые значения параметров в соответствии с шагами 1 – 4 рассмотренной технологии. При формировании логического ограничения на шаге 3 некоторые значения в логическом условии могут остаться неопределенными, они впоследствии будут запрошены у пользователя при старте приложения. Это позволит не размножать однотипные приложения, а создать одно настраиваемое приложение. При завершении работы Мастер-формы по Шаблон-форме создается пользовательская форма в текущей БД Access. Служебная информация для формы сохраняется во вспомогательной таблице БД Access. В начале работы пользователь активизирует разработанную для него форму, при необходимости определяет значения в логическом выражении и нажимает кнопку “Передать данные”. При этом в пользовательской форме автоматически выполняются шаги 5 – 7 рассмотренной технологии. Перед выполнением шага 7 запускается Excel. Представление данных формируется в рабочей области таблицы Excel. После работы с таблицей пользователь нажимает кнопку “Записать данные”, расположенную в форме Access и все изменения будут записаны в БД. Excel-файл закрывается без сохранения.

Шаги 1 – 7 рассмотренной технологии были реализованы Редреевым П.Г. в среде Delphi. В качестве источника данных был выбран файл формата mdb и интерфейс ADODB. Такая компоновка средств разработки приводит к достаточно длительному процессу формирования композиционной таблицы. Но основной причиной отказа от этой разработки явилось отсутствие в таблице Delphi средств анализа данных, аналогичных программе Excel. Дополнительным аргументом для выбора среды раз-

работки было то, что Excel является замкнутым по отношению к взаимодействию с другим программным обеспечением. В идеале инструментарий должен быть независимой программой, реализованной, например, в среде C++ и устанавливающей взаимодействие, с одной стороны, с Excel, с другой стороны – имеющей настраиваемый SQL-интерфейс с произвольной СУБД. К сожалению, Excel не допускает подобного с собой обращения, более того, присутствующие в документации средства по управлению другими программами из Excel просто не работают. Поэтому и был выбран подход, описанный в данной статье.

При разработке технологии предполагалось, что схема используемой БД будет корректно построена. То есть схема будет удовлетворять одной из финальных нормальных форм (не ниже третьей), удовлетворять свойству СБПИ и сохранять зависимости. Прежде всего, это используется при проверке условий существования преобразований. Мастер-форма для этого считывает исходную информацию из схемы. Если схема построена неправильно или модифицирована под специфические потребности, например ROLAP, то это существенно ограничит возможности системы по генерации приложений.

Список литературы

1. *Vassiliadis P., Sellis T.* A survey of logical models for OLAP databases // SIGMOD Rec. 1996. V. 28, No 4. P. 64–69. DOI: 10.1145/344816.344869
2. *Pedersen T.B., Jensen C.S., Dyreson C.E.* A foundation for capturing and querying complex multidimensional data // Inf. Syst. 2001. V. 26, No 5. P. 383–423. DOI: 10.1016/S0306-4379(01)00023-0
3. *Lechtenborger J., Vossen G.* Multidimensional normal forms for data warehouse design // Inf. Syst. 2003. V. 28, No 5. P. 415–434. DOI: 10.1016/S0306-4379(02)00024-8
4. *Li H.-G., Yu H., Agrawal D., Abbadi A.E.* Progressive ranking of range aggregates // Data & Knowledge Engineering. 2007. V. 63. No 1. 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. Capri: IEEE Computer Society. 1998. P. 63–72.
6. *Giorgini, P., Rizzi, S., Garzetti, M.* Goal-oriented requirement analysis for data warehouse design // Proceedings of the 8th ACM international Workshop on Data Warehousing and OLAP: DOLAP'05. Bremen: ACM. 2005. P. 47–56.
7. *Mazon J., Trujillo J., Lechtenborger J.* Reconciling requirement-driven data warehouses with data sources via multidimensional normal forms // Data & Knowledge Engineering. 2007. V. 63, No 3. P. 725–751. DOI: 10.1016/j.datak.2007.04.004
8. *Ullman J.* Principles of database systems // Computer Science Press, 1980. 379 p.
9. *Maier D.* The theory of relational databases // Computer Science Press, 1983. 637 p.
10. *Зыкин С.В.* Формирование гиперкубического представления реляционной базы данных // Программирование. 2006. № 6. С. 71–80. (English transl.: *Zykin S. V.* Formation of Hypercube Representation of Relational Database // Programming and Computer Software, 2006, V. 32, No 6. P. 348–354.)

11. *Редреев П.Г.* Построение иерархий в многомерных моделях данных // Известия Саратовского университета. Серия Математика. Механика. Информатика. 2009. Т. 9. № 4, ч. 1. С. 84–87. (*Redreev P.G.* Postroenie ierarhij v mnogomernyh modeljah dannyh // Izvestija Saratovskogo universiteta. Serija Matematika. Mehanika. Informatika. 2009. T. 9. No 4. ch. 1. S. 84–87 [in Russian].)
12. *Зыкин С.В.* Формирование гиперкубического представления данных со списочными компонентами // Информационные технологии и вычислительные системы. 2010. № 4. С. 38–46. (*Zykin S.V.* Formirovanie giperkubicheskogo predstavlenija dannyh so spisochnymi komponentami // Informacionnye tehnologii i vychislitel'nye sistemy. 2010. No 4. S. 38–46 [in Russian].)
13. *Miller L., Nila S.* Data Warehouse Modeler: A CASE Tool for Warehouse Design // Thirty-First Annual Hawaii International Conference on System Sciences. Kohala Coast: IEEE Computer Society. 1998. V 6. P. 42 – 48.
14. *Зыкин С.В.* Автоматизация формирования связей на схеме базы данных // Вестник Омского университета. 2010. № 4. С. 150–155. (*Zykin S.V.* Avtomatizacija formirovanija svjazej na sheme bazy dannyh // Vestnik Omskogo universiteta. 2010. No 4. S. 150–155 [in Russian].)
15. *Зыкин С.В.* Редактирование базы данных с использованием табличных представлений // Вестник Омского университета. 2012. № 2. С. 181–187. (*Zykin S.V.* Redaktirovanie bazy dannyh s ispol'zovaniem tablichnyh predstavlenij // Vestnik Omskogo universiteta. 2012. No 2. S. 181–187. [in Russian].)

Automation of Tabular Application Formation

Zykin S. V.

Sobolev Institute of Mathematics, Acad. Koptug avenue, 4, Novosibirsk, 630090, Russia

Keywords: tabular applications, relational database, online analytical processing, lossless join

The paper considers automation problems of the interface formation between a table and a relational database. The task description is formalized and the description of the existing approaches to formation of data representations on an example of widely widespread CASE-tools is submitted. The definition of intermediate data representation as a "join table" is offered, which is used for maintenance of correctness of data representation formation, and also is necessary for direct and inverse data transformations. On the basis of lossless join property and realized dependencies, the concept and a way of context formation of the application and restrictions is introduced. The considered material is further used for constructing an inverse data transformation from tabular presentation into a relational one. On the basis of relationships properties on a database scheme, the partial order on the relations is established, and the restriction of acyclic databases schemes is introduced. The received results are further used at the analysis of principles of formation of inverse data transformation, and the basic details of such a transformation algorithm are considered.

Сведения об авторе:

Зыкин Сергей Владимирович,

Омский филиал Учреждения Российской академии наук
Института математики им. С.Л. Соболева Сибирского Отделения РАН,
доктор техн. наук, профессор