Формирование системы SQL-запросов для отображения объектного пользовательского представления предметной области в базу данных

Дрождин В.В., Зинченко Р.Е.

Пензенский государственный педагогический университ им. В.Г. Белинского

Создание автоматизированных информационных систем (АИС), способных на основе объектного внешнего (пользовательского) представления предметной области (ПО) автоматически формировать базу данных (БД) и поддерживать корректное отображение объектов ПО в БД на основе системы базовых SQL-запросов, является актуальной задачей. Решине этой задачи позволит в полном объеме реализовать независимость пользовательского представления ПО от организации данных в БД.

В данной работе рассмотрим создание системы базовых SQL-запросов для исходно заданного пользовательского представления ПО.

В качестве примера ПО будем использовать информацию о студентах, содержащуюся в объектах двух типов: «человек» и «студент». При этом пользовательское представление ПО будет иметь следующий вид:

студент (человек, №_зачетки, группа, специальность, факультет)

человек (паспорт, фамилия, имя, отчество, дата_рождения, адрес)

Свойство (атрибут) «человек» в объекте «студент» является ссылкой на объект «человек» и указывает, что все свойства объекта «человек» включаются в объект «студент».

Исходную БД, соответствующую представлению ПО, можно сформировать в форме универсального отношения U со схемой:

U (паспорт, фамилия, имя, отчество, дата_рождения, адрес, №_зачетки, группа, специальность, факультет).

Для простоты можно считать, что все атрибуты имеют текстовый формат, т.к. определение типов данных для атрибутов не входит в круг вопросов данной работы. Ограничением в универсальном отношении является только отсутствие дубликатных кортежей.

При этом система базовых SQL-запросов, отображающих пользовательское представление объектов ПО в БД, формируется тривиально на основе операции проекции. Поэтому получим следующую систему базовых SQL-запросов:

1) человек:

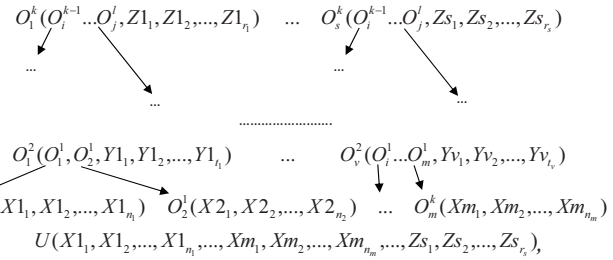
```
Select паспорт, фамилия, имя, отче-
  ство, дата_рождения, адрес
From U
```

2) студент:

```
Select *
From U
```

Рассмотрим решение этой задачи для общего случая.

Пусть имеется пользовательское объектное представление ПО вида:



где O_i^j – i-й компонент уровня j, включающий в себя хотя бы один объект уровня $j-1$ и, возможно, объекты более низких уровней l, а также собственные атрибуты.

Тогда универсальное отношение U будет иметь схему:

$$U(X1_1, X1_2, ..., X1_{n_1}, ..., Xm_1, Xm_2, ..., Xm_{n_m}, ..., Zs_1, Zs_2, ..., Zs_{r_s}).$$

Система базовых SQL-запросов примет вид:

1) Объект O_1^1:

```
Select X1_1, X1_2, ..., X1_n1
From U
...
```

2) Объект O_1^2:

```
Select
  X1_1, X1_2, ..., X1_n1, X2_1, X2_2, ..., X2_n1, Y1_1, Y1_2, ..., Y1_t1
From U
...
```

3) Объект O_s^k:

```
Select Xi_1,..., Xj_1, Yi_2,..., Yj_2, Zs_1, Zs_2,..., Zs_rs,
From U
```

Создание универсального отношения осуществляется командой:

```
Create table U
(
A1 Text,
A2 Text,
...
AN Text
)
```

где N – общее количество атрибутов в универсальном отношении и каждый атрибут A_i^j сопоставлен атрибуту объекта пользовательского представления ПО.

Хотя организация БД в форме универсального отношения имеет право на существование, однако будет требовать очень больших затрат на ведение

Табл. 1.

Фамилия	Имя	Отчество	Дата рождения	Адрес	Паспорт	№ зачетки	Группа	Специальность	Факультет
Андреев	Иван	Семенович	02.11.1982	Пенза	9811				
Николаев	Петр	Петрович	13.02.1982	Пенза	7314				
Андреев	Илья	Семенович	22.03.1981	Тамбов	2538				
Куликов	Петр	Иванович	28.03.1982	Пенза	1033				
Семенов	Кирилл	Иванович	22.03.1981	Пенза	8588				
Козлов	Петр	Викторович	15.11.1982	Кузнецк	4031				
Денисов	Николай	Андреевич	03.08.1982	Пенза	5781				
Алексеев	Семен	Анатольевич	28.04.1981	Кузнецк	3781				
Иванов	Иван	Иванович	07.08.1981	Пенза	6114				
					9811	5	ПЭ1	ПЭ	ФЭМИ
					7314	2	ПЭ2	ПЭ	ФЭМИ
					2538	6	МИ1	МИ	ФМФ
					1033	7	МИ1	МИ	ФМФ
					8588	4	ПЭ2	ПЭ	ФЭМИ
					4031	10	ПЭ1	ПЭ	ФЭМИ
					5781	8	МИ2	МИ	ФМФ
					3781	1	МИ2	МИ	ФМФ
					6114	3	МП1	МП	ФЭМИ
					9811	9	МИ1	МИ	ФМФ

и иметь слишком низкую надежность вследствие возникновения коллизий. Следовательно, необходима разработка БД более точно моделирующей ПО. Создание такой БД целесообразно осуществить на основе обучения АИС.

Будем считать, что АИС создает универсальное отношение в качестве учебной (временной) БД и просит пользователя дать примеры гипотетических (возможных) объектов ПО. Предположим, пользователь ввел следующие данные (таблица 1).

Тогда, используя алгоритм выявления функциональных зависимостей, предложенный Д. Мейером[1], АИС для заданного набора данных определяем следующий минимальный набор зависимостей *F*:

специальность → факультет

группа → специальность

№_зачетки → паспорт, группа

паспорт → фамилия, имя, отчество, дата_рождения, адрес

С помощью алгоритма проектирования БД в 3НФ [1] АИС разрабатывает схему БД вида:

R1 (специальность, факультет)

R2 (группа, специальность)

R3 (паспорт, фамилия, имя, отчество, дата_рождения, адрес)

R4 (№_зачетки, паспорт, группа)

Одиночным подчеркиванием отмечены первичные ключи отношений.

Для устранения дублирования данных в каждом отношении создадим атрибут ID с уникальными числовыми значениями, который будет использоваться в качестве первичного ключа отношения, а все внешние ключи заменим на ID_Ri. Тогда получаем следующую схему БД:

R1 (ID, специальность, факультет)

R2 (ID, группа, ID_R1)

R3 (ID, паспорт, фамилия, имя, отчество, дата_рождения, адрес)

R4 (ID, № зачетки, ID_R3, ID_R2)

Двойным подчеркиванием отмечены возможные ключи отношений.

Учитывая, что отношение R3 содержит всю информацию об объекте ПО «человек», а отношение R4 позволяет с помощью операции естественного соединения собрать все данные об объекте ПО «студент», переименуем эти отношения соответственно в «человек» и «студент». Объекты, хранящиеся в отношениях R1 и R2, не имеют интерпретации в терминах ПО и представляют более «тонкую» (точную) структуру ПО, хотя, возможно, сложившуюся в данном состоянии ПО (заданном примере), о которой пользователь не имеет никакого представления или такая «тонкая» структура ПО не требуется ему для решения предполагаемых (конкретных) задач.

Создание разработанной БД осуществляется набором операторов:

```
Create table R1
(
ID Integer Constraint R1ID Primary key,
специальность Text,
факультет Text
)
Create table R2
(
ID Integer Constraint R2ID Primary key,
группа Text,
ID_R1 Integer Constraint A_ID_R1
   references R1,
Constraint unique группа
```

1 Мейер Д. Теория реляционных баз данных. – М.: Мир, 1987. – 608 с.

```
)
Create table Человек
(
ID Integer Constraint ЧеловекID Primary
   key,
паспорт Text,
фамилия Text,
имя Text,
отчество Text,
дата_рождения Text,
адрес Text,
Constraint unique паспорт
)
Create table Студент
(
ID Integer Constraint СтудентID Primary
   key,
№_зачетки Text,
ID_R2 Integer Constraint A_ID_R2
   references R2,
ID_R3 Integer Constraint A_ID_R3
   references R3,
Constraint unique №_зачетки
)
```

В общем случае оператор создания отношения БД имеет следующий вид:

```
Create table Ri
(
ID Integer Constraint RiID Primary key,
ID_Rj Integer Constraint A_ID_Rj[_γ]
   references Rj,
…
Ak Text,
…
Constraint unique Ak1[,Ak2,…]
)
```

где γ – параметр, означающий номер ссылки на отношение R_j и обязательный в случае наличия нескольких ссылок на одно отношение.

На основе соответствия атрибутов новой схемы БД и универсального отношения и использования операции естественного соединения отношений (Join) получаем следующую систему базовых SQL-запросов:

1) человек:

```
Select *
From Человек
```

2) студент:

```
Select *
From R1 Join R2 Join Студент
  Join Человек
```

В общем случае SQL-запрос, позволяющий получить информацию об объекте ПО, имеет вид:

```
Select *
From Ri Join Rj Join …
```

Теперь АИС может удалить универсальное отношение и создать БД, адекватную ПО и обеспечивающую решение информационных задач с высокой эффективностью.

Предложенный метод формирования БД без проектирования и генерация системы SQL-запросов, отображающей внешнее представление ПО в БД, позволяет пользователям, решающим практически полезные задачи, самостоятельно создавать необходимые АИС. Это существенно расширит использование АИС в различных областях деятельности человека при минимальных затратах на их создание и эксплуатацию.