ПРЕОБРАЗОВАНИЕ ВЫРАЖЕНИЙ АЛГЕБРЫ A В РАЗЛИЧНЫЕ ЯЗЫКИ ЗАПРОСОВ

ОПРЯ Дмитрий, ВОЙКОВ Александр
Coordonator: САРАНЧУК Дориан

Технический Университет Молдовы

Аннотация. *В работе приведено исследование Алгебры A, ее представление в виде двоичного дерева, а также преобразование исходного двоичного дерева в различные языки запросов. Алгебра A рассмотрена и с теоретической и с практической точки зрения, причем упор сделан на практическое применение алгебры. Рассмотрены различные операции, а также их аналоги на других языках запросов. Также представлено приложение, которое моделирует некоторую базу данных, выполняет автоматическое преобразование запросов, а также выводит их результат на основе созданной базы данных.*

Ключевые слова: *Алгебра A, реляционная алгебра, алгебра Кодда, язык запросов SQL, бинарное дерево, отношение, атрибут.*

1. Введение

Базисом предложенной Кристофером Дейтом и Хью Дарвеном Алгебры A являются операции реляционного отрицания (дополнения), реляционной конъюнкции (или дизъюнкции) и проекции (удаления атрибута). Реляционные аналоги логических операций определяются в терминах отношений на основе обычных теоретико-множественных операций и позволяют выражать напрямую операции пересечения, декартова произведения, естественного соединения, объединения отношений и т. д. Путем комбинирования базовых операций выражаются операции переименования атрибутов, соединения общего вида, взятия разности отношений. Алгебра A позволяет лучше осознать логические основы реляционной модели, хотя, безусловно, является в меньшей степени ориентированной на практическое применение, чем алгебра Кодда.

Нельзя не упомянуть еще и о том, что алгебра Кодда в действительности не является алгеброй отношений в математическом смысле, поскольку ее операции применимы не ко всем отношениям. В отличие от этого Алгебра A – это «настоящая» алгебра, в которой отсутствуют какие-либо ограничения на операнды операций [1].

2. Краткое описание Алгебры A

В исходный базовый набор операций входят операции реляционного дополнения <not>, удаления атрибута <remove>, переименования атрибута <rename>, реляционной конъюнкции <and> и реляционной дизъюнкции <or>.

Поскольку некоторые базовые операции Алгебры A имеют названия обычных логических операций, чтобы избежать путаницы, имена реляционных операций берутся в угловые скобки: <not>, <and>, <or> и т. д.

Операция <not> производит дополнение s заданного отношения *r*. Заголовком *s* является заголовок *r*. Тело *s* включает все кортежи, соответствующие этому заголовку и не входящие в тело *r*.

Пусть *s* обозначает результат операции *r* <remove> A. Операция <remove> производит отношение *s*, формируемое путем удаления указанного атрибута A из заданного отношения *r*. Операция эквивалентна взятию проекции *r* на все атрибуты, кроме A. Заголовок *s* получается теоретико-множественным вычитанием из заголовка *r* множества из одного элемента. Тело *s* состоит из таких кортежей, которые соответствуют заголовку *s*, причем каждый из них является подмножеством некоторого кортежа тела отношения r.

Пусть *s* обозначает результат операции *r* <rename> (A, B). Операция <rename> производит отношение *s*, которое отличается от заданного отношения *r* только именем одного его атрибута, которое изменяется с A на B.

Пусть *s* обозначает результат операции r_1 <and> r_2. Операция <and> является реляционной конъюнкцией, в некоторых случаях выдающей в результате отношение *rs*, ранее называвшееся естественным соединением двух заданных отношений r_1 и r_2.

Пусть *s* обозначает результат операции r_1 <or> r_2. Операция <or> является реляционной дизъюнкцией и обобщением того, что ранее называлось объединением. Заголовок *s* есть объединение

заголовков r_1 и r_2. Тело s состоит из всех кортежей, соответствующих заголовку s и являющихся надмножеством либо некоторого кортежа из тела r_1, либо некоторого кортежа из тела r_2. [1].

3. Преобразование базовых операций Алгебры А в запросы SQL, алгебры Кодда и реляционной алгебры

Используя построенное двоичное дерево можно выполнить преобразования исходной строки, написанной на языке Алгебры А (АА), в строку, которая является запросом: SQL, алгебры Кодда (АК), реляционной алгебры (РА). Здесь представлены примеры таких преобразований, где используются все пять операций АА, описанных выше. Вначале записана исходная строка, а ниже представлены ее интерпретации в других языках. Для данного примера используется два простых отношения *r*(A, B), *s*(A, D, C).

```
AA: <not> r
PA:(π{A}(r_domain) x π{B}(r_domain)) \ r
AK:(PROJECT r_domain {A} JOIN PROJECT r_domain {B}) MINUS r
SQL:
SELECT DISTINCT temp_0.A AS A, temp_1.B AS B
FROM r_domain as temp_0, r_domain as temp_1
EXCEPT SELECT DISTINCT A, B
FROM r;

AA: s <remove> A
PA: π{D, C}(s)
AK: PROJECT s {D, C}
SQL:
SELECT DISTINCT s.D AS D, s.C AS C
FROM s;

AA: s <rename> (C, X)
PA: s
AK: s RENAME {C, X}
SQL:
AS SELECT DISTINCT s.A AS A, s.D AS D, s.C AS X
FROM s;

AA: s <and> r
PA: s [><] r
AK: s NATURAL JOIN r
SQL:
SELECT DISTINCT *
FROM s  NATURAL JOIN r;

AA: s <or> r
PA: s >< r
AK: s FULL OUTER JOIN r
SQL:
SELECT DISTINCT temp_0.A AS A, temp_1.D AS D,
temp_2.C AS C, temp_4.B AS B
FROM
s_domain AS temp_0, s_domain AS temp_1, s_domain AS temp_2,
r_domain AS temp_3, r_domain AS temp_4,
s AS temp_5, r AS temp_6
WHERE (temp_0.A = temp_3.A)
AND
((temp_0.A = temp_5.A AND temp_1.D = temp_5.D
AND temp_2.C = temp_5.C ) OR
(temp_3.A = temp_6.A AND temp_4.B = temp_6.B ));
```

4. Программная реализация

Описанный выше алгоритм получил практическое применение в проекте «Simple converter». Конечной целью данного проекта была разработка программного обеспечения, которое позволяло бы произвести преобразование выражения на языке реляционной алгебры или Алгебры А в эквивалентное выражение (или ряд выражений) в терминах SQL, и наоборот. Первоочередной задачей встал выбор реляционной СУБД, которая бы соответствовала следующим требованиям:

- встраиваемое ядро;
- имплементация ADO.Net адаптера;
- компактность;
- максимальная приближенность синтаксиса к оригинальному языку SQL.

Всем требованиям соответствовала СУБД SQLite.

SQLite — компактная встраиваемая реляционная база данных. Слово «встраиваемый» означает, что SQLite не использует парадигму клиент-сервер, то есть ядро SQLite не является отдельно работающим процессом, с которым взаимодействует программа, а предоставляет библиотеку, с которой программа компонуется и движок становится составной частью программы. SQLite хранит всю базу данных (включая определения, таблицы, индексы и данные) в единственном стандартном файле на том компьютере, на котором исполняется программа. Простота реализации достигается за счёт того, что перед началом исполнения транзакции записи весь файл, хранящий базу данных, блокируется; Несколько процессов или потоков могут одновременно без каких-либо проблем читать данные из одной базы. Благодаря архитектуре ядра возможно использовать SQLite как на встраиваемых системах, так и на выделенных машинах с гигабайтными массивами данных. [3].

Основная идея заключалась в том, что конечным преобразованием считается выражение в терминах SQL–запрос, который и будет исполняться. Другими словами, после конвертации выражения на языке Алгебры А или реляционной алгебры в некий SQL-запрос, он будет исполнен, а его результат будет представлен пользователю. При этом, по возможности, выражение на языке Алгебры А будет конвертировано в выражение на языке реляционной алгебры и наоборот.

Ниже представлены основные окна приложения с демонстрационным примером (рис. 1, 2).

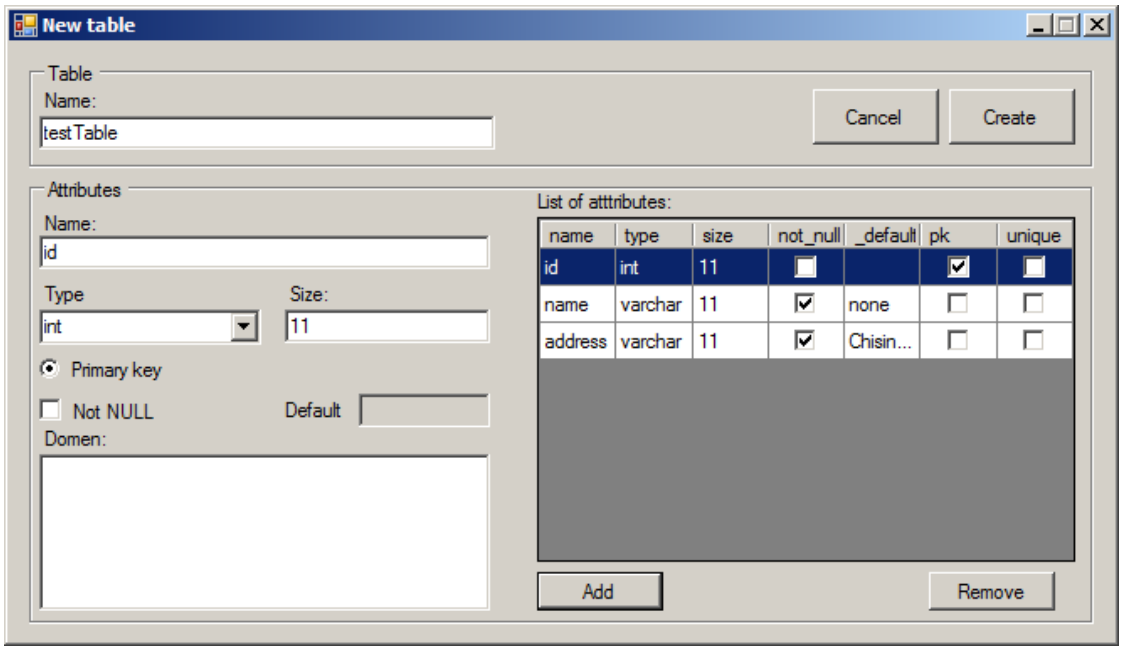

Рис. 1 —Окно создания отношений

Вначале создается несколько отношений, для которых задаются атрибуты, различные свойства, а также домены для атрибутов. Перечисление элементов домена необходимо для корректной работы запросов Алгебры А. Надо отметить, что реализована мгновенная валидация данных при потере фокуса — это проверки на валидность имени таблицы и атрибута.В текстовое поле Алгебры А было введено выражение «<not> r <and> <not> s» и нажата кнопка «Convert». После этого в текстовых полях SQL и RA были отображены эквивалентные выражения в терминах SQL и реляционной алгебры, соответственно. В верхней части окна показан результат исполнения выражения конвертированного в SQL-скрипт.

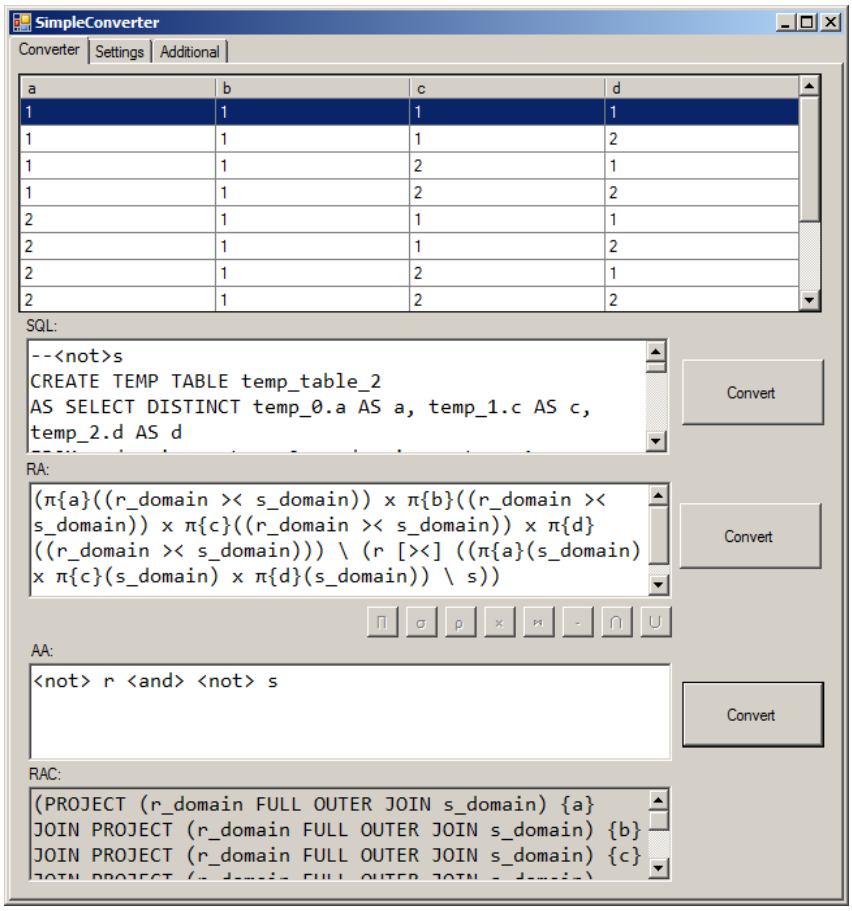

Рис. 2 — Основное окно приложения

Надо также отметить, что приложение обладает примитивным CRUD интерфейсом для таблиц, а также предоставляет возможность просмотреть список таблиц и их атрибутов.

Заключение

Алгебра A была разработана Кристофером Дейтом и Хью Дарвеном для того, чтобы отдалиться от операций над множествами. В результате этих работ была создана новая алгебра, которая включает в себя минимальный, но достаточный набор операций над отношениями. Она оказалась не слишком удобной, как видно на примерах выше, но, тем не менее, ее язык достаточно элегантен и красив.

Практический смысл Алгебры A остается под большим вопросом. В большинстве случаев, при рассмотрении запросов, необходимо основываться на понятии «домен», а именно конкретное перечисление элементов, входящих в домен данного атрибута. Например, очевидно, что для натуральных чисел невозможно перечислить элементы домена.

Также в Алгебре A отсутствует выборка определенных кортежей или отдельных элементов по некоторому условию. С учетом данных факторов, ее практическая ценность резко падает, а интерес к ней становится сугубо теоретическим.

Библиография

1. Codd. E.F. *Relational Completeness of Data Base Sublanguages*. Data Base Systems, Courant Computer Science Symposia Series 6. — Englewood Cliffs, N.J.: Prentice-Hall, 1972.
2. Томас Х. Кормен, Чарльз И. Лейзерсон, Рональд Л. Ривест, Клиффорд Штайн. *Алгоритмы: построение и анализ*, 3-е издание, — М.: «Вильямс», 2013.
3. Джон Ливерт. *SQLite*. URL: http://ru.wikipedia.org/wiki/SQLite.