ПРЕОБРАЗОВАНИЕ SQL ЗАПРОСОВ В ВЫРАЖЕНИЯ РЕЛЯЦИОННОЙ АЛГЕБРЫ

САВИЦКИЙ Максим, ЛЯЛИН Сергей
Coordonator: САРАНЧУК Дориан

Технический Университет Молдовы

Аннотация. *В работе приведены исследования и сравнения между реляционной алгеброй и языком запросов SQL. Приведены примеры некоторых преобразований выражений SQL в реляционную алгебру. Разработано приложение для автоматизации преобразований. Произведено сопоставление линейных последовательностей лексем естественного или формального языка с его формальной грамматикой.*

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

1. Введение

Существует более или менее универсальный способ манипуляции данными, поддерживаемый почти всеми серверными реляционными СУБД и большинством универсальных механизмов доступа к данным (в том числе при использовании их совместно с настольными СУБД). Этот способ — применение языка SQL (Structured Query Language — язык структурированных запросов). Самые главные особенности данного языка, это то, что с его помощью можно извлекать и суммировать данные, добавлять, удалять и модифицировать записи, защищать данные от несанкционированного доступа, создавать базы данных. [1]

Для преобразования выражения SQL в выражение реляционной алгебры создано приложение, которое принимает на входе выражение SQL, а на выходе выдаёт выражение реляционной алгебры. Данные команды могут быть обратно не конвертированными. К сожалению, не все запросы языка SQL можно преобразовать, а только те запросы, в которых отсутствует рекурсия.

2. Описание языка SQL

Structured Query Language представляет собой непроцедурный язык, используемый для управления данными реляционных СУБД. Термин «непроцедурный» означает, что на данном языке можно сформулировать, что нужно сделать с данными, но нельзя проинструктировать, как именно это следует сделать. Иными словами, в этом языке отсутствуют алгоритмические конструкции, такие как метки, операторы цикла, условные переходы и др.

Язык SQL был создан в начале 70-х годов в результате исследовательского проекта IBM, целью которого было создание языка манипуляции реляционными данными. Первоначально он назывался SEQUEL (Structured English Query Language), затем — SEQUEL/2, а затем — просто SQL. Официальный стандарт SQL был опубликован ANSI (American National Standards Institute — Национальный институт стандартизации, США) в 1986 году (это наиболее часто используемая ныне реализация SQL). Данный стандарт был расширен в 1989 и 1992 годах, поэтому последний стандарт SQL носит название SQL92. В настоящее время ведется работа над стандартом SQL3, содержащим некоторые объектно-ориентированные расширения [1].

3. Операторы языка SQL

Data Definition Language (DDL)

Data Definition Language содержит операторы, позволяющие создавать, изменять и уничтожать базы данных и объекты внутри них (отношения, представления и др.). Эти операторы перечислены в таблице 1.

Таблица 1. Операторы Data Definition Language

Оператор	Описание
CREATE TABLE	Применяется для добавления новой таблицы к базе данных
DROP TABLE	Применяется для удаления таблицы из базы данных
ALTER TABLE	Применяется для изменения структуры имеющейся таблицы
CREATE VIEW	Применяется для добавления нового представления к базе данных
DROP VIEW	Применяется для удаления представления из базы данных

Data Manipulation Language (DML)

Data Manipulation Language содержит операторы, позволяющие выбирать, добавлять, удалять и модифицировать данные. Операторы DML представлены в таблице 2.

Таблица 2. Операторы Data Manipulation Language

Оператор	Описание
SELECT	Применяется для выбора данных
INSERT	Применяется для добавления строк к таблице
DELETE	Применяется для удаления строк из таблицы
UPDATE	Применяется для изменения данных

Иногда оператор SELECT относят к отдельной категории, называемой Data Query Language (DQL).

4. Работа с операторами SQL

Выбор данных

Выбор данных представляет собой наиболее часто встречающуюся операцию, выполняемую с помощью SQL. Оператор SELECT — один из самых важных операторов этого языка, применяемый для выбора данных. Синтаксис этого оператора имеет следующий вид:

```
SELECT column-list
FROM table-list
[WHERE where-clause]
```

Операторы SELECT должны содержать слова SELECT и FROM; другие ключевые слова, такие как WHERE являются необязательными.

За ключевым словом SELECT следуют сведения о том, какие именно атрибуты необходимо включить в результирующие отношение. Знак звезды (*) после слова SELECT обозначает все атрибуты отношения.

Для выбора одного атрибута применяется следующий синтаксис:

```
SELECT Фамилия
```

Пример выбора нескольких атрибутов имеет вид:

```
SELECT Фамилия, Имя, Возврат
```

Если выбор данных осуществляется из нескольких отношений и при этом выбираются одноименные атрибуты из разных отношений, следует ссылаться на имена отношений для полной идентификации атрибутов, например:

```
SELECT студенты.Фамилия, преподаватели.Фамилия
```

Предложение FROM

Для указания имен отношений, из которых выбираются кортежи, применяется ключевое слово FROM, например:

```
SELECT * FROM студенты
```

Этот запрос возвратит все кортежи из отношения студенты.

Если в результирующем отношении нужны только атрибуты Имя и Фамилия, можно ввести следующее предложение SELECT:

```
SELECT Имя, Фамилия FROM студенты
```

Пример запроса к нескольким отношениям приведен ниже:

```
SELECT студенты.Фамилия, преподаватели.Фамилия
FROM студенты, преподаватели
```

Предложение WHERE

Для фильтрации результатов, возвращаемых оператором SELECT, можно использовать предложение WHERE, синтаксис которого имеет вид:

```
WHERE expression1 [{AND | OR} expression2 […]]
```

Например, вместо получения полного списка студентов можно ограничиться только теми из них, которых зовут Максим:

```
SELECT * FROM студенты
WHERE Имя = "Максим"
```

В предложении WHERE можно использовать различные выражения, например:

```
SELECT * FROM студенты
WHERE Оценка > 5 AND Оценка < 10
```

5. Описание грамматики языка SQL

Для преобразования выражений из SQL в реляционную алгебру создано приложение [2]. Приложение написано на язык C#, так как данный язык является одним из самых популярных языков и имеет большие функциональные возможности. Предпочтение C# было сделано также из-за наличия в нём библиотеки **Spart**, которая используется в проекте. На языке C++ есть схожая библиотека – **Boost.Spirit**. Во многом функционал этих библиотек схож, так как их основная задача - создание парсеров. Парсинг — это процесс сопоставления линейной последовательности лексем (слов) естественного или формального языка с его формальной грамматикой. Данные парсеры необходимы для проверки корректности записи выражения на языке SQL.

Библиотека **Spart** позволяет составлять грамматику и правила, которые в процессе могут быть легко изменены. Также данная библиотека даёт возможность следить за ходом операции, когда строка проходит через созданную грамматику. [3]

Ниже приведена формальная грамматика, где показывается, какой вид должна иметь входная команда на языке SQL.

$$Oper_{Select} \rightarrow SELECT\ (Atr_{add} | *)$$

(2)

$$Oper_{From} \rightarrow FROM\ Table$$
$$Oper_{Where} \rightarrow WHERE\ Expres$$
$$Oper_{Where} \rightarrow \varepsilon$$
$$Table \rightarrow table, Table$$
$$Table \rightarrow table$$
$$Atr_{add} \rightarrow attribute, Atr_{add} \mid attribute$$
$$Atr_{add} \rightarrow table.attribute, Atr_{add} \mid table.attribute$$
$$Expres \rightarrow Atr_{add}\ Operator\ Atr_{num} \mid \{AND|OR\}\ Expres$$
$$Atr_{num} \rightarrow Atr_{add} \mid const$$
$$Start \rightarrow Oper_{Select}\ Oper_{From}\ Oper_{Where}$$

6. Примеры переводов выражений SQL в реляционную алгебру

Исходя из грамматики, описанной выше, запрос на языке SQL может состоять, как и с выражением Where, так и без него. Также атрибуты могут быть записаны с идентификацией, то есть записать имя отношения перед именем атрибута. После ввода выражения производится пошаговая проверка (парсинг). Далее показаны несколько запросов и их переводы:

```
SELECT студенты.Фамилия, преподаватели.Фамилия FROM студенты,
преподаватели
```

После ввода выражения выполняется обработка атрибутов в команде Select. При обработке, приложение запоминает имена отношений, записанных в Select. Это необходимо для дальнейшего

перевода. Также здесь происходит проверка, чтобы имена атрибутов не начинались с не алфавитного символа.

После проверки части команды Select происходит обработка части команды From. Здесь тоже происходит проверка, чтобы имена отношения не начинались с не алфавитного символа. Также, если в операторе From находятся больше одного отношения, тогда необходимо между ними будет выполнить декартово произведение. После выполнения проверки всего выражения, приложение выдаст следующие выражение на языке реляционной алгебры:

$$\pi_{\text{Фамилия}}(\text{студенты}) \times \pi_{\text{Фамилия}}(\text{учителя})$$

В следующем запросе на языке SQL будет добавлена операция Where. В данном операторе атрибуты могут выражаться в виде атомарной или составной формулы, содержащей логические операторы AND, OR, NOT. Данный оператор выражается на языке реляционной алгебры с помощью оператора Селекция ($\sigma_{\text{Имя=Александр}}(\text{студенты})$).

```
SELECT Имя, Фамилия FROM студенты
WHERE Имя = "Максим"
```

Выражение на языке реляционной алгебры будет иметь следующий вид:

$$\sigma_{\text{Имя=Максим}}(\pi_{\text{Имя, Фамилия}}(\text{студенты}))$$

Заключение

Язык запросов РА не получил широкого распространения в современных технологиях СУБД. Языком практического применения в БД является SQL, однако знания РА наравне с SQL является необходимым для лучшего понимания сути операции над отношениями в реляционных БД. Перевод запросов на языке SQL в выражения на языке реляционной алгебры является не сложной задачей, но данных перевод необходим для того, чтобы лучше понимать, как происходят операции над отношениями. Сложности появились, когда необходимо было идентифицировать имена отношений перед именем атрибута в операторе Select. После обработки данные атрибуты должны быть выставлены в правильном порядке в нужных операциях проекции.

Литература

1. Алексей Федоров, Наталия Елманова. *Введение в базы данных. Часть 6. Введение в язык SQL.* URL: http://www.lib.csu.ru/dl/bases/prg/kompress/articles/2000_10_DBMS6/index.htm
2. Саранчук Дориан, Опря Дмитрий, Войков Александр. *Преобразование выражений алгебры А в различные языки запросов*. Conferinţa tehnico-ştiinţifică a colaboratorilor, doctoranzilor şi studenţilor, Universitatea Tehnică a Moldovei, 22 octombrie 2013, Chişinău, Republica Moldova.
3. Jonathan de Halleux. *Spart, a parser generator framework C#.* URL: http://www.codeproject.com/Articles/5676/Spart-a-parser-generator-framework-100-C