BOOST.SPIRIT И РАЗБОР SQL SELECT

САРАНЧУК Дориан, ВИЗИР Иван, ТУРЧАК Даниил, МИХАЙЛЕНКО Евгений

Технический Университет Молдовы

***Аннотация**: Boost Spirit — набор библиотек, позволяющий создавать синтаксические анализаторы прямиком на C++, не обращаясь к сторонним средствам. Spirit был применён для разбора SQL в программе «SQLtoRA» для преобразования произвольных запросов в выражения реляционной алгебры. Цель продукта – упростить процесс изучения реляционной алгебры для начинающих специалистов.*

***Ключевые слова**: SQL, парсер, синтаксический анализ, дерево разбора, Boost Spirit.*

1. Введение

Spirit — набор библиотек C++ для синтаксического разбора, выполненный как встраиваемый предметно-ориентированный язык с использованием шаблонного мета-программирования. Целевая грамматика разбираемого текста создаётся целиком на языке C++, причём она свободно может смешиваться с другим кодом C++ и сразу же становится готовой к использованию. Анализаторы описываются декларативными выражениями грамматики целевого языка в виде, близком к форме Бэкуса-Наура (БНФ).

Язык SQL является очень гибким, поэтому выражения языка, запросы Select в частности, могут быть очень сложными, с сильной вложенностью подзапросов друг в друга, с интенсивным использованием условий выборки, операторов множеств, итд. Поэтому, в случае необходимости разбора этих выражений, не стоит изобретать велосипеды, а лучше изучить стандарт языка и его грамматику.

2. Spirit и грамматика языка

Грамматики языка SQL различных версий свободно доступны в Интернете на сайтах различных производителей систем управления базами данных (вне, собственно, документов стандартов), что, определённо, упрощает подготовку к созданию анализатора выражений. Доступность именно грамматик выражений является огромным преимуществом в описываемой теме, ибо, как указанно выше, анализаторы Spirit описываются именно как грамматики исследуемого языка, потому основная часть работ по созданию парсера сводится к переводу грамматики из графического или иного представления в выражения Spirit на языке C++. Например:

<query specification>::= SELECT [<set quantifier>] <select list> <table expression>

Boost.Spirit:

```
// где-то в объявлении класса грамматики
qi::rule<Iterator_t, skipGrammar> QuerySpec, SetQuantifier, SelectList, TableExpr;
// в конструкторе
QuerySpec = "SELECT">>-SetQuantifier>> SelectList>>TableExpr;
```

В описанном выше примере ясно прослеживается соответствие имён элементов правил и порядка их следования. Описание остальных элементов выражений Select — фильтров where, having, группировки отобранных кортежей, соединения и так далее, описываются таким же образом.

Результатом такой работы должен стать анализатор, способный распознать в тексте, поданном ему на вход, корректный запрос SQL Select, или же отрапортовать об ошибке, если во входном тексте допущена ошибка.

Подобные анализаторы несут мало смысла, обычно в качестве результата ожидают нечто в виде синтаксического дерева — структуры данных, сформированной в виде дерева, где каждый элемент соответствует какому-либо элементу проанализированного текста. При этом дерево должно быть удобным для различных манипуляций со стороны использующего анализатор программиста.

3. Абстрактное дерево разбора

В Boost.Spirit содержатся средства для автоматической генерации и последующего заполнения дерева, которое становится результатом деятельности анализатора наряду с указанием успешности разбора. Элементы дерева описываются программистом в виде структур (struct), элементами которых

могут быть как простые типы данных (int, char, double), строки (std::string), так и другие элементы описываемого дерева. Так как Spirit работает со своим внутренним представлением результатов анализа текста — tuple, так называемыми кортежами — то имена членов структур не имеют никакого значения, но важен порядок размещения этих элементов, а именно типов этих членов. Во время разбора текста кортежи, являющиеся результатами успешно пройденного правила, будут прозрачно преобразованы в структуры и помещены в дерево. Помимо этого, элементы структур должны быть объединены с помощью средств Spirit в комбинации, соответствующие условиям, комбинациям, повторениям и другим регулярным элементам правил грамматики анализатора. Например:

<simple table>::=<query specification>|<table value constructor>|<explicit table>
<query specification>::=SELECT [<set quantifier>]<select list> <table expression>

Соответствующие элементы дерева и их использование:

```
structsqlSimpleTable{boost::variant<sqlQuerySpec,sqlTableConstructor,sqlTable> value;};
stuct sqlQuerySpec { boost::optional<sqlSetQuantifier> SetQuantifier;
sqlSelectList SelectList;      sqlTableExpr TableExpr; };
qi::rule<Iterator_t, SimpleTable(), skipGrammar> SimpleTable;
qi::rule<Iterator_t, sqlQuerySpec(), skipGrammar> QuerySpec;
SimpleTable = QuerySpec | TableValConstr | ExplTable;
QuerySpec = "SELECT" >> -SetQuantifier >> SelectList >> TableExpr;
```

Последним шагом для описания элементов дерева является оборачивание каждой структуры, участвующей в дереве разбора, в макросы, создающие код для преобразования кортежей Spirit в структуры, описанные программистом.
В результате, чтобы разобрать текст парсером, достаточно передать в функцию qi::phrase_parse ссылку на корень дерева, ссылку на объект анализатора, ссылку на объект грамматики пропусков пробелов, а так же итератор строки, в которой содержится входной текст. После этого, если текст был верен, функция вернётся с успехом, а переданный в функцию корень дерева будет указывать на всё сформированное дерево целиком. Далее программист сможет выяснить список полей, список таблиц, условия выборки, способ и условие соединения таблиц и другие элементы запроса простым обходом дерева на языке C++.

4. Применение Spirit

Одной из возможных операций может быть преобразование запроса SQL Select в выражение на языке реляционной алгебры, которое может быть легко выполнено ввиду того, что грамматика выражений реляционной алгебры описывается во многом так же, как описываются запросы SQL.
В примеры программ, использующих Boost.Spirit для анализа кода, можно привести «boolmin» — программу для минимизации логических выражений, «CliPP» — генератор листингов кода C++ в вёрстке HTML, и 3D-движок «XEngine», использующий Spirit для разбора shading-языка.
Используя описанный выше подход, было реализовано приложение «SQLtoRA» [4], позволяющее выполнять проверку запросов SQL на верность, а так же преобразование некоторого их подмножества в выражения реляционной алгебры. Помимо этого, было реализовано обратное преобразование запросов, написанных на языке реляционной алгебры, в язык SQL. Приложение реализовано на основе Qt версии 4.8.1. с использованием библиотеки для генерации парсеров Boost Spirit.

Библиография

1. Jonathan Leffler. *BNF Grammar for ISO/IEC 9075:1992 - Database Language SQL (SQL-92).* URL: http://savage.net.au/SQL/sql-92.bnf.html
2. SQL Tutorial – W3School. URL: http://www.w3schools.com/sql/default.asp
3. Joel de Guzman, Hartmut Kaiser, *Boost Spirit 2.5.2.* URL: http://www.boost.org/doc/libs/1_52_0/libs/spirit/doc/html/index.html
4. Саранчук Дориан, Михайленко Евгений, Турчак Даниил, Визир Иван, Кулчу Татьяна. *Обучающая система для двунаправленного преобразования реляционных и SQL выражений.* Conferinţa Ştiinţifică a Colaboratorilor, Doctoranzilor şi Studenţilor UTM, 23 noiembrie 2013, UTM, Chişinău, Republica Moldova.