УДК 004.434

ВИКОРИСТАННЯ ПРОГРАМНОЇ СИСТЕМИ ANTLR ДЛЯ СТВОРЕННЯ МОВИ РЕЛЯЦІЙНОЇ АЛГЕБРИ

Фісун Микола, Кандиба Ігор

Чорноморський національний університету імені Петра Могили

Анотація

Розглянуто спосіб побудови компілятора мови реляційної алгебри як предметно-орієнтованої мови для використання в навчальному процесі при опануванні курсу з організації баз даних. Пропонується використати засоби генератора парсерів ANTRL, крос-платформової мови Python з подальшою передачею отриманого коду в SQL на виконання СКБД: SQLite або MySQL.

Abstract

This article is devoted to research the method of constructing a compiler for relational algebra as a domain-specific language. This language can be used in the database organization course. It is proposed to use the generators parsers ANTRL and cross platform programming language Python for generating SQL code for DBMS: SQLite and MySQL.

Вступ

Предметно-орієнтовані мови (ПОМ) відомі вже дуже давно. На зорі становлення мов програмування загального призначення вони розроблялися для програмування вузького кола завдань, щоб користувач комп'ютера, який добре володіє моделями предметної області і не дуже підготовлений як програміст, міг запрограмувати свої завдання. Однак цим мовам не судилося стати помітною частиною в IT-індустрії. Це пояснюється створенням спеціалізованих бібліотек в середовищі мов програмування загального призначення, можливостями графічного інтерфейсу і підвищенням програмістської підготовленості фахівців. в предметних областях. Тому зараз серед багатьох і багатьох предметно-орієнтованих мов залишилися такі, що охоплюють широке коло специфічних задач. Як приклад можна навести такі мови моделювання, як GPSS, DYNAMO та ін. Але в останні роки спостерігається підвищення інтересу до предметно орієнтованих мов [1], де вони англійською називаються Domain-Specific Languages (DSL). Вони стають популярними з двох основних причин. Перше – це підвищення продуктивності праці розробників, оскільки програмний код на ПОМ може бути в кілька разів коротше програмного коду на мові програмування загального призначення. Друге – поліпшення зв'язку з фахівцями в предметній області. Правильно вибрана мова може зробити складний блок коду істотно простіше і для програмування, і для розуміння, що підвищує продуктивність працюючих з ним. Вона спрощує спілкування розробника програмного забезпечення (ПЗ) з фахівцями в предметній області. Текст програми на ПОМ виступає одночасно і як виконуване ПЗ, і як опис проблеми, яке вузькі фахівці в цій галузі знань можуть прочитати, щоб зрозуміти, як їх уявлення виглядає в програмному представлені.

Мета дослідження

В даному дослідженні в якості такої ПОМ обрано мову реляційної алгебри (REAL), яка, разом з реляційним численням, є математичним базисом мови SQL. Звичайно, вона ні в якому разі не замислюється як конкурент SQL. REAL буде використовуватися в навчальному процесі в курсі з основ баз даних для глибшого розуміння реляційної моделі даних. Вихідним мовою відповідного компілятора буде мова SQL, програма на якої буде виконуватися в програмному середовищі обраної СКБД.

Результати дослідження

Оператори мови REAL діляться на два класи: створення і ведення реляційних таблиць, які схожі з відповідними операторами мови SQL, і оператори маніпулювання таблицями, тобто оператори самої реляційної алгебри. Останні включають таки ключові слова як INTERSECT, UNION, MINUS, INNER/LEFT/RIGHT JOIN, DIVIDE BY, WHERE та ін., що запропоновані в [2]. В мові передбачено пріоритетність операцій і використання дужок, тобто є можливість записувати вирази, що робить процес компіляції не простим. Наведемо приклад одного з можливих виразів маніпулювання з таблицями реляційної БД.

(((TA,BLE1 **minus** TA,BLE2) **where** TA,BLE1.ATR1 = “good” **and** TABLE2.ATR4 > 120) **union** TA,BLE3) **[**ATR1, ATR2, ATR3, ATR4**]**;

Послідовність виконання операторів наступна. Спочатку виконується операція віднімання (*minus*), над отриманим відношенням виконується операція вибірки (*where*), потім виконується операція об’єднання (*union*) цього відношення з відношенням/таблицею TA,BLE3 і останньою операцією є операція проекції (*[…]*) отриманого після операції union відношення по атрибутах <ATR1, ATR2, ATR3, ATR4>. При цьому під час семантичного аналізу компілятор, зокрема, має перевіряти на сумісність відношень перед виконанням операцій *minus* і *union*.

Розглядалися кілька підходів для реалізації мови REAL. В результаті обрано генератор парсерів ANTLR (ANother Tool for Language Recognition) [3]. Він підтримує можливість роботи з багатьма сучасними мовами програмування: Python, C++, Java та ін.. Результат роботи даного інструменту є класи аналізаторів на мові обраній користувачем.

Побудова мови з використання інструменту ANTLR вимагає написання спеціалізованого файлу з набором граматичних правил, які будуть перетворені у лексичний та синтаксичний аналізатори. Перетворення створених правил у програмний код здійснюється застосунком на мові Java, що входить до складу пакету ANTLR: java org.antlr.v4.Tool-Dlanguage=Python2 REALM_Union.g4 [4]. ANTLR підтримує можливість відображення синтаксичного дерева з використанням графічного інтерфейсу, як показано нижче на рисунку.

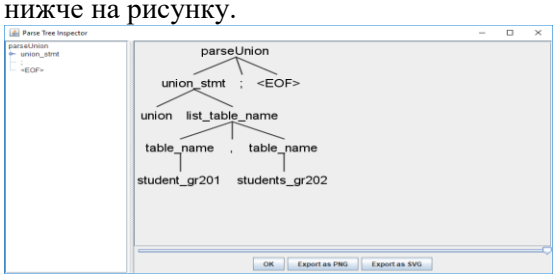

Рисунок – Синтаксичне дерево запиту Union

Виклик даного інструменту здійснюється також за допомогою java: java org.antlr.v4.gui.TestRig REAL_Union parseUnion -gui input.txt, де input.txt – файл, що містить запит на мові REAL, у даному випадку синтаксис запиту Union з переліком таблиць через кому.

Список використаних джерел:

1. Фаулер Мартин. Предметно-ориентированные языки программирования: Пер. с англ. М. : ООО ‘‘И.Д. Вильямс’’, 2011. – 576 с.

2. Дейт К. Дж. Введение в системы баз данных. - Киев*Москва: Диалектика, 1998. - 787 с.

3. Теренс Парр. The Definitive ANTLR 4 Reference. - Pragmatic Bookshelf, 2011. – 322 с.

4. Документація інструментів ANTLR[Електронний ресурс]. – Режим доступу : URL : https://github.com/antlr/antlr4/blob/master/doc/python-target.md – Загол. з екрану/