Impact Factor:					
	ISRA (India) = 3.117	SIS (USA) = 0.912	ICV (Poland) = 6.630
	ISI (Dubai, UAE) = 0.829	РИНЦ (Russia) = 0.156	PIF (India) = 1.940
	GIF (Australia) = 0.564	ESJI (KZ) = 8.716	IBI (India) = 4.260
	JIF = 1.500	SJIF (Morocco) = 5.667	OAJI (USA) = 0.350

SOI: 1.1/TAS DOI: 10.15863/TAS

International Scientific Journal

Theoretical & Applied Science

p-ISSN: 2308-4944 (print) **e-ISSN:** 2409-0085 (online)

Year: 2019 **Issue:** 05 **Volume:** 73

Published: 29.05.2019 **http://T-Science.org**

SECTION 4. Computer science, computer engineering and automation.
UDC 004.

QR – Issue QR – Article

Oleg Yurievich Sabinin
Candidate of Engineering Sciences, Associate Professor
Peter the Great St. Petersburg Polytechnic University
olegsabinin@mail.ru

Nikita Vladimirovich Gorbatov
Student
Peter the Great St. Petersburg Polytechnic University
nik.gorbatov@gmail.com

DEVELOPMENT OF AN ALGORITHM FOR TRANSLATING NATURAL LANGUAGE SENTENCES INTO SQL QUERIES

Abstract: *This article discusses process of development an algorithm for translating natural language sentences into SQL queries*

Key words: *Database, Text to SQL.*
Language: *Russian*
Citation: Sabinin, O. Y., & Gorbatov, N. V. (2019). Development of an algorithm for translating natural language sentences into SQL queries. *ISJ Theoretical & Applied Science, 05 (73),* 414-418.
Soi: http://s-o-i.org/1.1/TAS-05-73-61 ***Doi***: https://dx.doi.org/10.15863/TAS.2019.05.73.61

РАЗРАБОТКА АЛГОРИТМА ТРАНСЛЯЦИИ ПРЕДЛОЖЕНИЙ ЕСТЕСТВЕННОГО ЯЗЫКА В ЗАПРОСЫ НА ЯЗЫКЕ SQL

Аннотация: *В данной статье рассматривается алгоритм транслятора естественного языка в запросы на языке SQL.*

Ключевые слова: *Базы данных, перевод текста в SQL.*

1 Introduction

С развитием технологий во многих сферах нашей жизни нам всё чаще приходится сталкиваться с различными информационными системами, для взаимодействия с которыми требуются определённые навыки и знания. Такие технологии проникают всё в большее количество сфер и, как следствие, возникает необходимость в повышении квалификации некоторых сотрудников, или же найма дополнительных, обладающих нужными навыками. В связи с этим возникает вопрос, можно ли найти менее затратный и автоматизированный способ решения данной проблемы?

Одна из самых очевидных идей – разработка «интерфейса» или прослойки, которая позволит людям без нужных навыков взаимодействовать, например, с СУБД [1].

В данной статье рассматривается разработка алгоритма трансляции предложения на естественном языке в запросы на языке SQL, анализируются подводные камни данной проблемы, а также указываются конкретные методы их решения.

2 Motivation

Самым лёгким способом взаимодействия с любым не знакомым сервисом для любого пользователя, вне зависимости от его навыков и образования, является взаимодействие через привычный пользователю язык, со знакомыми ему грамматическими конструкциями. Так как преобразовывать машинную логику в естественный язык существенно сложнее, чем проводить обратное преобразование (в связи с более широким спектром эмоциональной окраски и большого количества синонимов в естественном языке).

И именно данную функцию берёт на себя машинный формальный перевод. Больше всего

нас интересует первое появление интеллектуальной обработки текстов, которое берёт своё начало в 60х годах XX-го века, что в свою очередь, с увеличением вычислительной мощности в последующие года, привело к появлению нескольких базовых способов интерпретации естественного языка, таких как синтаксический и семантический анализ шаблонов. Синтаксический строится на основе разбора фразы с учётом частей предложений, семантический – использует информацию из предыдущего метода и дополняет его информацией из тезаурусов.

Базовые особенности перевода в SQL завязаны на его строгой структурированности и формализованности. Более строгая структурированность характеризуется малым количеством типовых команд и их форм.

3 Basic interpretation

Основную задачу перевода можно упростить до выявления типа команды, определения полей и дополнение их грамматическими конструкциями языка [2]. Но такой алгоритм применим только на базовом уровне, т. е. с запросами обычной структуры, без использования соединений или вложенных запросов. К тому же возникают сложности, если предложение на естественном языке плохо сформулировано, или содержит большое количество лишней информации. Такая проблема характерна не только для SQL, но и для большинства формализованных языков.

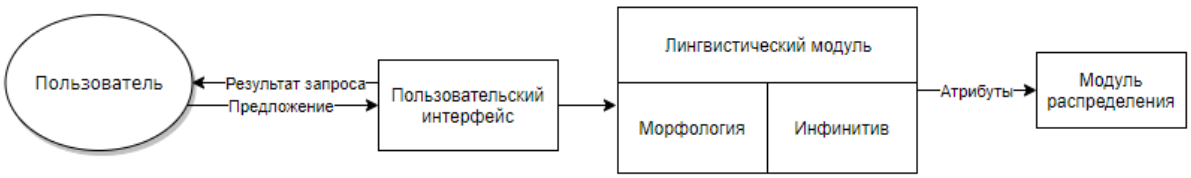

Рисунок 1 - Процедура разбора

Основная идея алгоритма состоит в первую очередь во всевозможной формализации конкретного предложения. Для того чтобы этого достичь следует избавиться от всевозможных словоформ определяя их инфинитив, но при этом сохраняя информацию о первоначальной формулировке. Изначальные формулировки могут помочь в определении логических взаимосвязей между операндами в предложении, так что знать их характеристики тоже полезно.

К определению инфинитивов есть несколько подходов. Один из самых старых и базовых – словарные алгоритмы, основанные на использовании больших и массивных словарей, и поиска по ним [3]. Очевидный минус их использования – довольно медленное выполнение и сильная зависимость от обновления словарей. Другой вариант – алгоритмы, основанные на морфологических конструкциях [4]. Оба подхода позволяют узнать как другие формы слова, так и некоторое количество вспомогательной информации. Но по причинам скорости работы и большого спектра возможностей, для решения нашей задачи логичнее использовать лингвистический анализатор основанный на втором методе. Также стоит отметить, что данный модуль будет необходим ещё и из-за большого синонимичного набора слов (одно и то же действие/объект можно обозначить по-разному) и каждое такое слово может быть не только в базовой форме.

После ввода команды от пользователя, пользовательский интерфейс передаёт её в лингвистический анализатор [5], задача которого определить важные атрибуты (такие как тип команды, объект, с которым требуется выполнить действие и различные параметры), после чего преобразовать их в базовую форму, а также заменить синонимами, если требуется. После этого из команды атрибутов и параметров (в изменённой форме) создаётся вектор, который передаётся в модуль распределения.

Кроме того, для решения поставленной задачи, нам потребуется ещё ряд дополнительной информации, не относящейся к конкретному введённому предложению. Во-первых - модуль взаимодействия с системой управления базами данных. Он используется для получения информации о схеме, с которой нам предстоит работать, для возможности подключения, отсылки запросов в базу данных, а также для анализа ошибок (отдельно хочется отметить, что планируется обрабатывать ошибки на стороне БД, а реагировать только на коды ошибок различными способами).

Во-вторых, небольшой модуль загрузки словарных данных. Дело в том, что для определения ключевых элементов команды нам потребуются различные словари синонимов. Также подобные словари потребуются для конвертации словесных формулировок условий и фильтров.

Стоит отдельно остановиться на том, что в зависимости от того, какой конкретно вид запроса нас интересует, процесс преобразования будет отличаться, и для разных типов запросов следует использовать разные сценарии транслятора. Для того, чтобы корректно переключаться между этими сценариями, следует проанализировать формализованную версию предложения. Этот процесс представлен модулем распределения (рис.1).

Задача модуля распределения – выяснить, какому из модулей передавать управление. Решение принимается следующим образом – мы имеем информацию о конкретной схеме в базе данных, полях, хранящихся в таблицах, ключах и ещё ряде параметров. По ним модуль распределения сможет определить тип команды. В случае если мы работаем с выборкой - хранится ли вся нужная нам информация в одной таблице, или же нам потребуется соединять несколько таблиц.

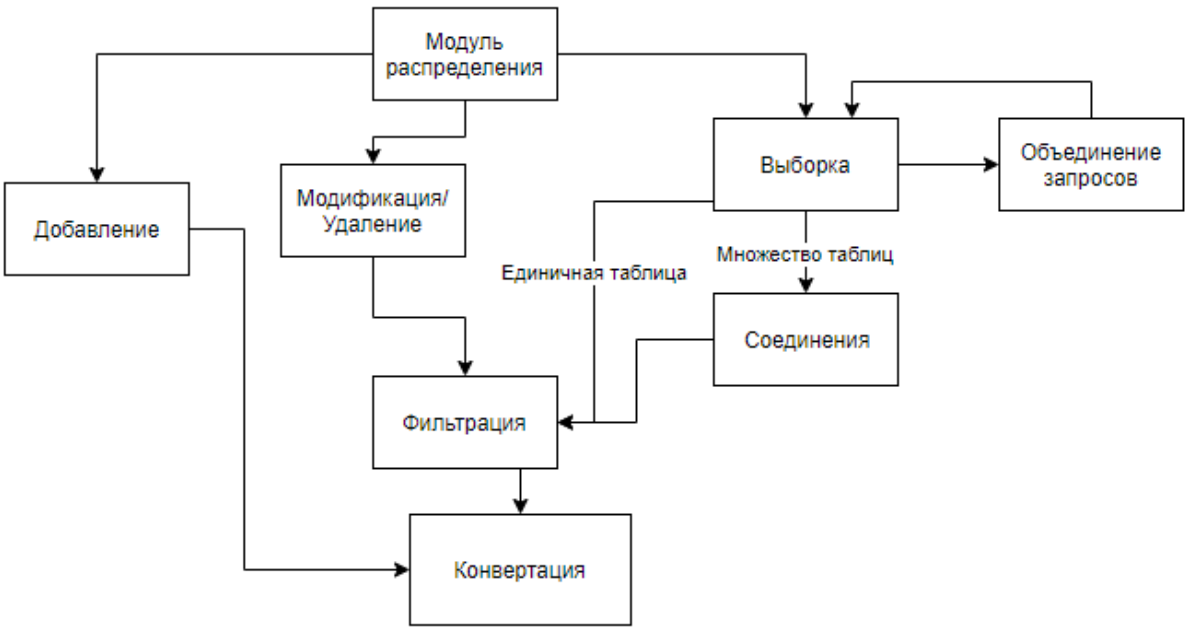

Рисунок 2 - Схема выбора вида запроса

4 Processing modules

Конкретно рассматривая некоторые сценарии разбора запросов (рис.2) для начала хочется обговорить механизм обработки запросов, содержащих в себе множественное соединение таблиц. При попытке перевода связанных запросов с использованием простых методов формализации возникают сложности. Требуется не только определить какую и откуда информацию нужно вывести, но и какую последовательность связей между таблицами нужно построить для создания запроса. Существует несколько способов решения данной проблемы. Один из давно известных – ручное составление словаря-схемы базы данных, с указанием полного пути соединения для каждых двух и более таблиц. Более современный подход – обучение нейронной сети, с целью создания преобразователя [6; 7; 8]. В данной работе хотелось бы взять лучшее из двух подходов, а именно отсутствие необходимости вручную корректировать решение для других схем базы данных, а также скорость решения конкретной задачи.

Идея такого решения довольно проста - из базы данных получаем набор таблиц, информацию о соединениях по ключам, и эту информацию пускаем в нейросеть, с целью получения полной карты соединений, а также весовой информации [9]. После получения полной карты мы можем использовать её для анализа и составления запроса с соединениями без нужды переделывать карту, пока структура схемы не будет изменена.

Ещё из возможных вариантов выборки стоит отметить запросы, состоящие из объединения двух компонентов, такие как union и minus. Фактически, работа с таким видом мало отличается от базового запроса выборки, за тем исключением, что благодаря позиционным атрибутам и некоторым ключевым словам, мы разделяем предложение на 2 части, анализируя их

как отдельные запросы, и затем объединяем их нужным оператором [10].

Один из самых часто используемых в разных типах запросов является модуль фильтрации, который будет использоваться как в самых разных видах выборок, так и в запросах модификации и удаления. Фильтрация основывается на целом ряде факторов - во-первых на информации о полях базы данных, во-вторых на логических и позиционных связях, в-третьих на морфологических характеристиках. Модуль фильтрации всегда анализирует обработанное предложение после других модулей для того, чтобы случайно не обработать данные, относящиеся к другим разделам.

Кроме различных запросов выборки, алгоритм должен уметь работать с запросами модификации. Основная сложность с ними состоит в определении конкретного действия, которое следует сделать с изменяемыми данными. Основываясь на морфологических характеристиках и ключевых словах, запрос может представлять собой как замену, так и модификацию с использованием текущих значений, с поддержкой преобразования текстовых операторов изменения и их преобразования в формализованный вид.

Более простым образом обстоит ситуация с обработкой запросов удаления, так как выявление базовой части запроса состоит из простого определения полей базы данных логически и позиционно связанных с командными маркерами. Оба предыдущих модуля, как и модуль обработки выборки, просто посылает обработанную информацию в модуль фильтрации для определения нужды анализа оставшейся части предложения, на предмет маркеров-фильтров.

Следующий модуль, обрабатывающий запросы о котором хотелось бы поговорить - модуль добавления данных. Он довольно прост алгоритмически и представляет собой последовательный анализатор, использующий позиционно-логические связки между объектами, которые можно отнести к полям схемы и словами, не относящимися к ключевым словам и полям, и преобразования этих связей в формальный вид. Единственная проблема с данным модулем представляется в следующем. Фактически нет никакой возможности проверить корректность введения пользователем информации если поля, в которые добавляется информация имеют, один тип. Как следствие, при работе с данным типом запросов на пользователя переносится наибольшая ответственность за корректность вводимых данных.

Последний модуль, это модуль конвертации, собирающий воедино все разобранные ранее конструкции и подставляющий их в конкретные команды языка SQL. После этого запрос исполняется с помощью модуля взаимодействия с системами управления базами данных и возвращается пользователю либо в виде результата, либо в виде ошибки (рис.1).

5 Conclusion

Данная статья описывает алгоритм трансляции естественного языка в запросы на языке SQL. Были рассмотрены требуемые для выполнения этой задачи способы формализации данных и основные этапы разбора (рис.1). Так же были обговорены и проанализированы способы и методы определения и трансляции конкретных вариаций запросов. (рис.2) Были перечислены логические модули алгоритма, отдельные этапы анализа и их важные особенности, которые помогут при реализации. Данная статья позволит в будущем разработать транслятор предложения естественного языка в запросы на языке SQL.

References:

1. (n.d.). «*Natural language interface for database: A brief review*» *Mrs Neelu Nihalani, Sanjay Silakari, and Mahesh Motwani*. Retrieved May 9, 2019, from https://pdfs.semanticscholar.org/00e2/6133551f152f4c8913c5442238d30a63ef79.pdf
2. Naihanova, L. V. (n.d.). *«methods and translation algorithms natural language inquiries to data base in SQL requests»* Retrieved April 5, 2019, from http://window.edu.ru/catalog/pdf2txt/669/18669/1120
3. Melnichuck, I. A. (1999). *«Practice of the theory of linguistic models "meaning-text"*, pp. 105-121.

4. (n.d.). Documentation API Yandex [online] Available at: tech.yandex.ru [Accessed 4 March 2019]
5. (n.d.). Documentation Yandex mystem Retrieved March 15, 2019, from https://tech.yandex.ru/mystem/doc/
6. (2016). «Vocabulary Selection Strategies for Neural Machine Translation» Gurvan L'Hostis 02.11.2016 Retrieved May 2, 2019, from https://arxiv.org/pdf/1610.00072.pdf
7. (2014). «Neural machine translation by jointly learning to align and translate» Bahdanau, D.; Cho, K.; and Bengio, Y. 2014.. ICLR 2015 Retrieved May 10, 2019, from https://arxiv.org/pdf/1409.0473.pdf
8. (2017). «An Encoder-Decoder Framework Translating Natural Language to Database Queries» Ruichu Cai, Boyan Xu , Xiaoyan Yang , Zhenjie Zhang , Zijian Li Advanced Digital Sciences Center, Illinois at Singapore Pte. Ltd, Singapore 2017. Retrieved April 5, 2019, from https://www.ijcai.org/proceedings/2018/0553.pdf
9. (2017). Natural Language Interface for Java Programming: Survey, Archana R. Shinde 10.2017 Retrieved April 20, 2019, from http://www.ijritcc.org/download/browse/Volume_5_Issues/November_17_Volume_5_Issue_11/1510221503_09-11-2017.pdf
10. (n.d.). «natural language to sql conversion system» anil m. Bhadgale, sanhita r. Gavas, meghana M. Patil & Pinki R. Goyal [online] Retrieved April 7, 2019, from https://s3.amazonaws.com/academia.edu.documents/31134370/17.NL_to_SQL_system-.full.pdf