

федеральное государственное автономное образовательное учреждение
высшего образования «Санкт-Петербургский национальный исследовательский
университет информационных технологий, механики и оптики»

На правах рукописи

Посевкин Руслан Владимирович

**Модели, методы и программные средства построения
естественно-языкового пользовательского интерфейса к базам данных**

05.13.11 - Математическое и программное обеспечение вычислительных
машин, комплексов и компьютерных сетей

Диссертация на соискание ученой степени
кандидата технических наук

Научный руководитель
д.т.н., доцент
Бессмертный Игорь Александрович

Санкт-Петербург – 2018

Оглавление

Введение	5
Глава 1. Обзор существующих пользовательских интерфейсов к базам данных.....	12
1.1 Анализ исследований в области обработки естественного языка и построения пользовательских интерфейсов к базам данных	12
1.2 Анализ существующих видов пользовательских интерфейсов к базам данных	15
1.3 Исследование естественно-языкового пользовательского интерфейса к базам данных	17
1.4 Анализ существующих реализаций естественно-языковых пользовательских интерфейсов	21
Выводы.....	26
Глава 2. Разработка метода построения естественно-языкового пользовательского интерфейса к базам данных	27
2.1 Разработка системы критериев качества естественно-языковых пользовательских интерфейсов	27
2.2 Анализ существующих методов обработки естественного языка	30
2.3 Разработка метода обработки естественно-языкового запроса с использованием шаблонов моделей предложений	35
Выводы.....	36
Глава 3. Разработка семантической модели базы данных и алгоритма ее автоматизированного формирования	38
3.1 Обоснование необходимости семантической модели базы данных. Требования к семантической модели базы данных	38
3.2 Структурный синтез семантической модели базы данных	41

3.3 Автоматизация процесса формирования семантической модели базы данных	43
3.3.1 Разработка алгоритма формирования семантической модели базы данных.....	43
3.3.2 Разработка программной системы автоматизированного формирования семантической модели базы данных	45
3.3.3 Разработка метода определения семантики типа связей между сущностями базы данных с использованием тезауруса.....	49
3.3.4 Разработка метода определения семантики сущностей базы данных на основе паттернов.....	52
3.3.5 Формирование таблицы проекций, обеспечивающей связь между терминами.....	54
Выводы.....	56
Глава 4. Разработка алгоритма построения запроса к базам данных на основе анализа естественно-языкового запроса.....	57
4.1 Использование семантической модели и К-представления для формирования запроса к базе данных	57
4.2 Разработка алгоритма формирования SQL-запроса на основе естественно-языкового запроса	58
4.3 Разработка функциональной модели естественно-языкового интерфейса	65
4.4 Обеспечение портируемости естественно-языкового интерфейса	67
Выводы.....	69
Глава 5. Экспериментальное исследование оценки качества разработанного естественно-языкового пользовательского интерфейса к базе данных.....	71
5.1 Оценка полноты, точности и F-меры естественно-языкового интерфейса	71

5.1.1. Постановка экспериментального исследования	71
5.1.2. Состав и организация тестового окружения	73
5.1.3. Анализ естественно-языковых запросов пользователя	83
5.1.4. Результаты экспериментального исследования	88
5.2 Оценка эффективности естественно-языкового интерфейса	92
5.2.1 Постановка экспериментального исследования	92
5.2.2 Состав и организация тестового окружения	92
5.2.3 Результаты экспериментального исследования	93
5.3 Оценка корректности работы естественно-языкового интерфейса, портированного на другой естественный язык	94
5.3.1. Постановка экспериментального исследования	94
5.3.2. Состав и организация тестового окружения	96
5.3.3. Результаты экспериментального исследования	96
Выводы	102
Заключение	103
Список литературы	106
Приложение 1. Свидетельство о регистрации объекта интеллектуальной собственности	115
Приложение 2. Акты о внедрении результатов диссертационного исследования	116
Приложение 3. Задание участника экспериментального исследования по оценке полноты, точности и F-меры естественно-языкового интерфейса ...	120
Приложение 4. Фрагменты исходного текста программы	126

Введение

Актуальность проблемы. Сегодня в мире сформировался класс программного обеспечения, где для взаимодействия человека с машиной наиболее эффективным подходом является использование естественно-языкового пользовательского интерфейса. На мобильных устройствах, таких как смартфоны и планшеты, в процессе взаимодействия с программами все чаще используются голосовые помощники, такие как Google Now, Siri, Cortana. Также в последнее время широкое распространение получили чат-боты, что позволяет пользователю взаимодействовать с программной системой с помощью привычного естественного языка, используемого в ежедневной коммуникации.

Естественно-языковой пользовательский интерфейс активно применяется в вопросно-ответных системах, где для получения ответа на вопрос пользователя требуется извлечение информации из связанной базы данных. При этом пользователю не требуется знать внутреннюю структуру базы данных и вручную формировать SQL-запросы. В данном контексте актуальна задача по преобразованию естественно-языкового запроса пользователя в запрос к базе данных на формальном языке.

Степень разработанности темы. Наибольший вклад в развитие обработки текстов, представленных на естественном языке, внесли Н. Хомский, А.А. Зализняк. Наибольший вклад в исследование естественно-языковых пользовательских интерфейсов внесли G. Hendrix, I. Androutsopoulos, T. Winograd, W. Woods, R. Kaplan. Над исследованием и разработкой естественно-языковых пользовательских интерфейсов на русском языке работали Ю.Д. Апресян, И.М. Богуславский, Е.И. Большакова, В.А. Жигалов, А.Е. Ермаков, Б.А. Кузнецов, М.Г. Мальковский, А.С. Нариньяни, Г.С. Осипов, Э.В. Попов, В.А. Фомичев и другие ученые.

Объектом исследования является пользовательский интерфейс к базе данных.

Предметом исследования являются методы преобразования пользовательского запроса на естественном языке в запрос к базе данных.

Целью исследования является повышение доступности информации, размещенной в базах данных для пользователя, не обладающего знаниями и навыками построения SQL-запросов.

Поставленная цель достигается решением следующих **задач**:

1. Анализ состояния проблемы и текущих исследований в области человеко-машинного взаимодействия.
2. Разработка семантической модели базы данных.
3. Разработка алгоритма и программной реализации автоматизированного формирования семантической модели базы данных.
4. Разработка алгоритма преобразования запроса пользователя на естественном языке в SQL-запрос.
5. Экспериментальное исследование работоспособности и эффективности разработанных методов и алгоритмов, реализующих естественно-языковой пользовательский интерфейс к базе данных.

Положения, выносимые на защиту:

1. **Метод построения естественно-языкового пользовательского интерфейса к базам данных.**
2. **Алгоритм формирования семантической модели базы данных.**
3. **Алгоритм построения запроса к базам данных на основе анализа текста, введенного пользователем на естественном языке.**
4. **Результаты экспериментального исследования по оценке качества работы разработанного естественно-языкового пользовательского интерфейса к базе данных.**

Научная новизна:

1. **Метод построения естественно-языкового пользовательского интерфейса к базам данных, отличающийся использованием шаблонов**

моделей предложений и обеспечивающий автоматическое извлечение данных без необходимости формирования пользователем SQL-запроса.

2. **Алгоритм формирования семантической модели базы данных,** отличающейся использованием таблицы проекций, обеспечивающей формирование запроса к базе данных с использованием терминов предметной области.

3. **Алгоритм построения запроса к базам данных на основе анализа текста, введенного пользователем на естественном языке,** отличающийся использованием семантической модели базы данных и позволяющий портировать естественно-языковой интерфейс на другие естественные языки и формальные языки запроса.

4. **Результаты экспериментального исследования по оценке качества работы разработанного естественно-языкового пользовательского интерфейса к базе данных,** подтверждающие достоверность полученных результатов.

Теоретическую и методическую основу исследования составляют методы прикладной лингвистики, теории баз данных, инженерии программного обеспечения. Методы исследования включают в себя эксперименты на тестовой базе данных в реляционной системе управления базами данных MySQL.

Достоверность и обоснованность результатов исследования подтверждается в результате сравнения разработанных методов и алгоритмов с существующими опубликованными материалами, а также внедрением полученных результатов.

Теоретическая значимость исследования обоснована тем, что экспериментальным путем подтверждена возможность преобразования запроса на естественном языке в SQL-запрос к базе данных.

Практическая значимость исследования подтверждается тем, что:

разработаны и внедрены в учебный процесс по дисциплине «Базы данных» на кафедре вычислительной техники Университета ИТМО, а также в

научно-исследовательскую деятельность международной научной лаборатории «Архитектура и методы проектирования встраиваемых систем и систем на кристалле» Университета ИТМО; полученные научные результаты в области построения естественно-языковых пользовательских интерфейсов к базам данных могут быть использованы в образовательных учреждениях, а также коммерциализированы в таких компаниях как Яндекс, Центр Речевых Технологий, Naumen, Terrasoft, Астерос, Центр программных решений, Линия24; создана модель эффективного применения разработанных методов к реализации естественно-языкового пользовательского интерфейса к базе данных; представлены методические рекомендации по дальнейшему совершенствованию методов и средств автоматизированного формирования семантической модели базы данных и естественно-языкового пользовательского интерфейса к базам данных.

Апробация результатов исследования. Основные положения диссертационной работы и результаты исследований докладывались на 12 всероссийских и международных конференциях, в том числе на международной студенческой научной конференции «Актуальные проблемы современной науки – новому поколению» (Ставрополь, 2015 г.), IV Всероссийском конгрессе молодых ученых (Санкт-Петербург, 2015 г.), XLIV научной и учебно-методической конференции (Санкт-Петербург, 2015 г.), 9-й и 10-й международных конференциях по приложениям в инфокоммуникационных технологиях AICT'15, AICT'16 (Ростов-на-Дону, 2015 г., Баку, Азербайджан, 2016 г.), III Международной научной конференции «Информационные технологии в науке, управлении, социальной сфере и медицине» (Томск, 2016 г.), XI всероссийской молодежной научно-практической конференции «Молодежные исследования и инициативы в науке, образовании, культуре, политике» (Биробиджан, 2016 г.), XVII международной научной конференции «Наука. Университет» (Новосибирск, 2016 г.), V всероссийской научно-практической конференции студентов, аспирантов и молодых ученых «Актуальные проблемы современной науки:

взгляд молодых» (Челябинск, 2016 г.), XVI международной конференции «Информатика: проблемы, методология, технологии» (Воронеж, 2016 г.), XLVI и XLVII научной и учебно-методической конференции Университета ИТМО (Санкт-Петербург, 2017 и 2018 гг.).

Публикации результатов исследования. По теме диссертации опубликовано шестнадцать работ, из них три статьи в журналах из перечня ВАК, 2 в изданиях, входящих в международную реферативную базы данных Scopus. Получено свидетельство о государственной регистрации программы для ЭВМ.

Личный вклад. Основные результаты, представленные в диссертации, получены лично автором. В статьях (Posevkin R., Bessmertny I., 2015), (Posevkin R., Bessmertny I., 2016), (Посевкин Р.В., Бессмертный И.А., 2016) постановка задач выполнена Бессмертным И.А, а разработка методов, алгоритмов и их практическая реализация выполнены автором. Постановка экспериментов диссертационного исследования выполнялась совместно с научным руководителем.

Объем и структура работы. Диссертационная работа изложена на 138 страницах, состоит из введения, пяти глав, содержащих 31 рисунок, 38 таблиц, заключения, приложений. Библиографический список включает 89 наименований.

Работа **соответствует паспорту специальности 05.13.11** - Математическое и программное обеспечение вычислительных машин, комплексов и компьютерных сетей – пункты «4. Системы управления базами данных и знаний» и «7. Человеко-машинные интерфейсы; модели, методы, алгоритмы и программные средства машинной графики, визуализации, обработки изображений, систем виртуальной реальности, мультимедийного общения».

Во введении определяется общее направление исследования, сформулированы цель и задачи, определяется научная новизна и практическая значимость работы.

В первой главе анализируется состояние проблемы и приводится обзор текущих исследований в области обработки естественного языка и построения пользовательских интерфейсов к базам данных. Также рассматриваются существующие виды пользовательских интерфейсов к базам данных, их преимущества, особенности и проблемы реализации. Проводится сравнительный анализ существующих реализаций естественно-языковых пользовательских интерфейсов к базам данных, на основе которого делается постановка задач диссертационного исследования.

Во второй главе рассматривается задача разработки метода построения естественно-языкового пользовательского интерфейса к базам данных. Анализируются существующие подходы и метод обработки естественного языка, используемых в естественно-языковых интерфейсах к базам данных. Разрабатывается система критериев качества естественно-языковых пользовательских интерфейсов к базам данных. Предлагается метод построения естественно-языкового пользовательского интерфейса к базам данных с использованием шаблонов моделей предложений, обеспечивающий автоматическое извлечение данных без необходимости формирования пользователем SQL-запроса.

В третьей главе описывается разработка семантической модели базы данных. На основе проведенного анализа обосновывается необходимость использования семантической модели базы данных в естественно-языковом пользовательском интерфейсе, формулируются требования к семантической модели. Также приводится описание разработанной программы автоматизированного формирования семантической модели базы данных.

Четвертая глава посвящена разработке алгоритма построения запроса к базам данных на основе анализа естественно-языкового запроса с использованием семантической модели базы данных и обеспечивающий портируемость естественно-языкового интерфейса.

Пятая глава содержит в себе результаты экспериментального исследования оценки качества разработанного естественно-языкового

пользовательского интерфейса на тестовой базе данных. Представленные результаты демонстрируют работоспособность разработанных методов и алгоритмов.

В **заключении** приводятся основные результаты и выводы по исследованию. В **приложениях** представлены свидетельство о регистрации программы для ЭВМ, акты о внедрении результатов исследования, задание участника экспериментального исследования, а также фрагменты исходного кода разработанной программы.

Глава 1. Обзор существующих пользовательских интерфейсов к базам данных

1.1 Анализ исследований в области обработки естественного языка и построения пользовательских интерфейсов к базам данных

Задача интеллектуальной обработки текстов и человеко-машинного взаимодействия является темой активных исследований начиная с 60-х годов XX века. Одна из наиболее значимых работ в области компьютерной лингвистики и обработки естественного языка – публикация Ноама Хомского «Синтаксические структуры» (Хомский Н., 1962). Положения, развитые в данной работе, до сих пор являются в компьютерной лингвистике доминирующими. В работе представлены идеи порождающих грамматик, основанных на правилах описания синтаксических структур. Работы Хомского послужили началом формирования системы грамматического описания на основе порождающих грамматик. В результате, на основе ряда грамматических правил возможно создать неограниченное множество предложений на естественном языке. Данное множество также включает в себя еще никогда и никем не высказанные предложения (Гуслякова А.В., 2016).

Значительный вклад в развитие советского и российского сегмента прикладной лингвистики внес А.А. Зализняк. В первой монографии (Зализняк А.А., 1967) приведено алгоритмическое описание склонения существительных, прилагательных, местоимений и числительных в русском языке в его письменной форме. В данной работе описаны важные теоретические проблемы морфологии, даны строгие определения лингвистических понятий, таких как «словоформа», «грамматическое значение» и прочие. В продолжении данной работы был опубликован грамматический словарь русского языка (Зализняк А.А., 1967), в котором для 100 тысяч слов русского языка представлена точная модель словоизменения. Данные знания легли в основу большинства современных программных

средств для обработки русскоязычного естественно-языкового текста, таких как парсер MyStem (Segalovich I.A. , 2003).

Наибольший вклад в исследование естественно-языковых пользовательских интерфейсов внесли G. Hendrix (Hendrix G.G., et al., 1978), I. Androutsopoulos (Androutsopoulos I., et al., 1995). Над исследованием и разработкой естественно-языковых пользовательских интерфейсов на русском языке работали Ю.Д. Апресян (Апресян Ю.Д., 1980), И.М. Богуславский (Богуславский И. М. и др., 2000), Е.И. Большакова (Большакова Е.И., 2014) (Большакова Е.И., и др., 2017), В.А. Жигалов (Жигалов В.А., и др., 2001) (Жигалов В.А., 2000), А.Е. Ермаков (Ермаков А. Е. и др., 2004) (Ермаков А. Е., 2002), Б.А. Кузнецов (Кузнецов Б.А. и др. , 2001), М.Г. Мальковский (Мальковский М. Г., 1985) (Мальковский М. Г. и др.), А.С. Нариньяни (Нариньяни А.С., 1979), Г.С. Осипов (Осипов Г. С. и др., 2005), Э.В. Попов (Попов Э.В., 1982), В.А. Фомичев (Фомичёв В. А. , 2007). и другие ученые.

Что касается практических реализаций в области человеко-машинного взаимодействия, то ключевыми разработками являются системы SHRDLU (Winograd T, 1972), LUNAR (Woods W.A., et al. , 1977), LIFER/LADDER (Hendrix G.G., et al., 1978).

SHRDLU – ранняя программная система понимания естественного языка, разработанная в Массачусетском технологическом институте в 1968-1970 годах. Программная система имитировала поведение робота, который манипулировал блоками на столе. Она могла управляться командами, сформированными с помощью обычных выражений английского языка, таких как «Pick up the green cube» («Возьми зеленый куб») и отвечать на вопросы вроде «How many blocks are in the box?» («Сколько блоков находится в коробке?»). В результате разработки данной системы была продемонстрирована возможность комбинирования синтаксиса, порождения выводов и семантики таким образом, что в итоге система начнет понимать естественный язык. SHRDLU обладала существенными ограничениями, так как могла управляться исключительно небольшим количеством предложений.

Область понимания ограничена лишь пространством блоков и только в настоящий момент времени. В результате, попытка расширения системы привела бы к снижению ее эффективности.

LUNAR (Woods W.A., et al., 1972) представлял собой естественно-языковой интерфейс базы данных. Работа системы была построена на использовании процедурной семантики и расширенной сети переходов. Система была способна обработать без ошибок 78% запросов, и до 90% после исправления ошибок. Однако, расширение системы было невозможно и LUNAR был неспособен обработать запросы, выходящие за пределы предметной области.

Система LIFER/LADDER представляет собой естественно-языковой пользовательский интерфейс, взаимодействующий с базой данных кораблей военно-морских сил США. Данный интерфейс в своей основе использует семантическую грамматику, где вместо синтаксических меток вида «глагол» или «существительное» применялись сущности вида «Корабль» или «Характеристика». Это означало, что система, как и SHRDLU, не могла быть расширена и была тесно привязана к предметной области. Тем не менее, в системе была предусмотрена возможность определения новых словарей, подсистема обработки незаконченного или неполного ввода.

Среди естественно-языковых пользовательских интерфейсов базы данных, предназначенных для обработки русскоязычных текстов можно отметить разработку А.С. Нариньяни ЗАПСИБ (ЗАПрос к Справочно-Информационной Базе) – конструктор лингвистических процессоров, который основан на семантически-ориентированном анализе в замкнутой предметной области (Нариньяни А.С., 1979). Система предоставляет пользователю возможность вести диалог посредством ограниченного подмножества естественного языка.

Естественным развитием проекта ЗАПСИБ является появление системы InterBase (Диненберг Ф.Г., и др., 1990). В системе используется семантически-ориентированный подход в обработке естественного языка. В дальнейшем

InterBase была доработана и реализована в виде коммерческого естественно-языкового интерфейса к базам данных InBASE (Жигалов В.А., и др., 2001).

1.2 Анализ существующих видов пользовательских интерфейсов к базам данных

Пользовательский интерфейс к базам данных включает в себя множество программных подсистем, которые осуществляют поиск, получение, обработку и просмотр информации, извлекаемой из базы данных.

Среди всех существующих видов пользовательских интерфейсов можно выделить следующие:

- Интерфейсы с использованием формального языка запросов;
- Интерфейсы с использованием запроса, формируемого посредством заполнения форм;
- Интерфейсы с графическим формированием запроса;
- Естественно-языковые интерфейсы.

При формировании запроса с помощью формального языка запросов, такого как язык SQL, пользователь должен знать синтаксис и правила формирования этого языка запросов. Также пользователь должен знать внутреннюю структуру конкретной базы данных, из которой происходит извлечение данных. В результате, при работе с подобным видом интерфейсов пользователю требуется обладать достаточно высоким уровнем квалификации. А столь высоким подобным уровнем квалификации обладают специалисты, которые занимаются проектированием программных систем.

Пользовательские интерфейсы, в которых запрос формируется с помощью заполнения форм, оказываются более дружелюбными для пользователя, чем формирование запросов с помощью формального языка запросов. При формировании запроса через заполнение форм пользователь видит некое множество параметров, а также множество доступных для поиска критериев и правил, а также список возможных значений полей. Наличие списка доступных значений позволяет минимизировать количество ошибки,

допущенных при формировании запроса. При использовании пользовательского интерфейса с формированием запроса с помощью заполнения форм возможные варианты запросов заранее сформированы разработчиком, а пользователю остается только заполнить недостающие значения. В результате, информация в интерфейсе может быть получена пользователем только в рамках нескольких типовых и заранее определенных разработчиком «срезов» всего объема информации, представленной в базе данных. Недостаток подобных пользовательских интерфейсов, как и в случае с интерфейсом с использованием формального языка запросов – у пользователя должен быть опыт работы с подобными интерфейсами. Также необходимо предварительно создать форму, что влечет за собой дополнительные трудозатраты разработчика при создании интерфейса.

Интерфейсы с графическим формированием запросов входят в состав ряда систем управления базами данных, как, например, Microsoft Access (Kasprzyk J., et. al., 1994). Преимуществом подобного решения является отсутствие необходимости у пользователя помнить названия полей, таблиц и команд формального языка запросов к базе данных. Помимо этого, для эффективной работы с интерфейсом с графическим формированием запросов пользователь должен обладать опытом и иметь представление о некоторых специфичных понятиях из области математики, а не предметной области (Vlachoudis V., et al, 2009). Например, термин связывания таблиц из реляционной алгебры. Еще одним препятствием для неподготовленного пользователя по работе с интерфейсом к базе данных с графическим формированием запросов выступает большое количество утомительных действий по работе с графическим интерфейсом. Фактически, пользовательский интерфейс позволяет создавать формальные запросы к базе данных в графическом виде (Жигалов В.А., 2000). При этом, как и в случае с интерфейсом с формальным языком запросов, пользователь должен иметь представление о внутренней структуре базы данных.

1.3 Исследование естественно-языкового пользовательского интерфейса к базам данных

Естественно-языковой пользовательский интерфейс представляет собой подмножество пользовательского интерфейса, который позволяет обрабатывать запросы, сформулированные на естественном языке. Для демонстрации пользователю найденной информации также возможно использование естественного языка в рамках естественно-языкового пользовательского интерфейса (Посевкин Р.В., 2018).

Естественно-языковые пользовательские интерфейсы включают в себя описание предметной области связанной базы данных. Данное описание находится выше логического уровня хранения данных и позволяет абстрагироваться от деталей внутренней реализации базы данных на структурном и содержательном уровнях (Жигалов В.А., 1997). Общая схема, демонстрирующая место естественного языка среди прочих уровней абстракций доступа к данным представлена на рисунке 1.1.

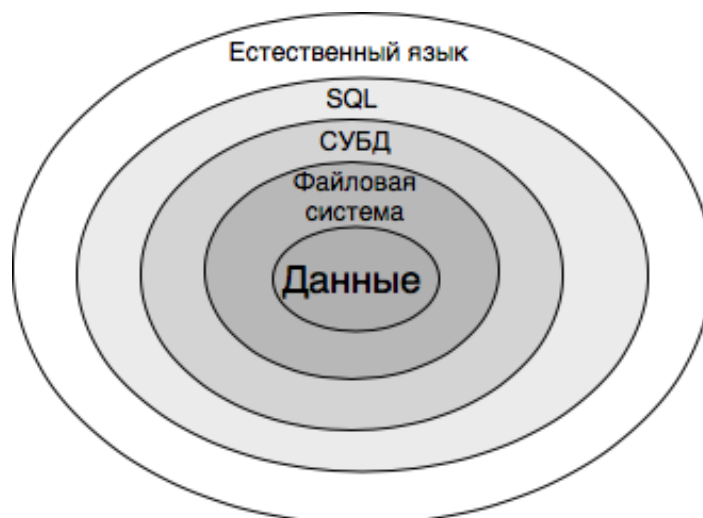


Рисунок 1.1 – Уровни абстракции доступа к данным

В качестве преимуществ естественно-языковых пользовательских интерфейсов можно отметить следующие аспекты:

- Большая гибкость. Один и тот же запрос пользователь может сформулировать различными способами. Естественно-языковой интерфейс способен обработать и интерпретировать различные формулировки

пользовательского запроса. В то время как интерфейсы с использованием запроса, формируемого посредством заполнения форм, графическим формированием запроса, а также с использованием формального языка запросов допускают только единственный вариант входных данных для одного набора выходных данных.

- Простота и удобство использования. Пользователю требуется меньше времени для изучения механики работы с интерфейсом. Помимо этого, отсутствует необходимость запоминания специального синтаксиса или терминов формального языка запросов.

- Возможность обеспечения поддержки пользовательской сессии. Процесс взаимодействия пользователя с естественно-языковым интерфейсом может быть реализован в виде диалога. При данном сценарии, система задает уточняющие вопросы пользователю. Уточняющие вопросы помогают сформировать лингвистический контекст основываясь на информации, полученной от пользователя в рамках предыдущего диалога.

- Скорость формирования запроса. Формирование запроса с применением естественного языка значительно быстрее, по сравнению с использованием интерактивных меню программной системы, составления запроса на формальном языке или написания отдельной компьютерной подпрограммы (Hendrix G.G., 1982).

- Специфичные вопросы. Пользовательские запросы, которые содержат отрицание или универсальную количественную оценку, могут быть достаточно просто сформулированы на естественном языке (Cohen P.R., 1992). При этом возможно возникновение сложностей при формировании подобных специфичных запросов в интерфейсах с графическим формированием запроса или в интерфейсах с использованием запроса, формируемого посредством заполнения форм (Androutsopoulos I., et al., 1995). В качестве примера естественно-языкового запроса, содержащего отрицания, может выступить запрос «В каком подразделении нет программистов?». Запрос, включающий

универсальную количественную оценку – «Какие отделы принимают участие в каждом проекте?»).

Среди проблем реализации естественно-языковых пользовательских интерфейсов стоит отметить следующие:

- **Неизвестность ограничений и всеобъемлемости данных.** При взаимодействии с естественно-языковым пользовательским интерфейсом для пользователя неочевидны все возможности и существующие ограничения. В качестве примера можно привести систему MASQUE (Auxerre P., 1986). Данная система способна корректно обработать запрос «Какие столицы у стран окружающих Балтийское море и окружающих Швецию?». Получив ответ, у пользователя создается ложное впечатление, что система способна обрабатывать любые комбинации словосочетаний внутри запроса. При этом, запрос «Какие столицы у стран окружающих Балтийское море и Швецию?» не может быть обработан системой (Dale R., et al., 2000). Интерфейсы с формальным языком запроса, с заполнением форм или графическим формированием запроса лишены подобного недостатка. В интерфейсах с формальным языком запросов синтаксис языка хорошо документирован и любой синтаксически верный запрос вернет ответ. В интерфейсах с заполнением форм или графическим формированием запроса пользователь обычно понимает какого рода запросы система сможет обработать. Пользователь сразу видит все доступные варианты, из которых возможно формирование запроса. В данном случае так же всегда будет возвращен ответ.

- **Лингвистические или концептуальные проблемы.** Естественно-языковой пользовательский интерфейс способен обрабатывать различные подмножества естественного языка. При этом возможна ситуация, когда запрос может быть интерпретирован различным образом, в то время как пользователь при формировании запроса об этом может не догадываться. В случае, если система не может обработать запрос, то для пользователя остается неочевидным с чем именно связана проблема. Проблема может быть связана с лингвистическими особенностями заданного запроса или концептуальной

проблемой построения запроса (Tennant H.R., et al., 1983). При наличии проблемы, связанной с лингвистическими особенностями заданного запроса, пользователь будет пытаться перефразировать запрос продолжая употреблять понятия, о которых у системы нет данных. Например, запрос о детях сотрудников, при том, что в базе данных нет информации о детях. Данная проблема связана с недостаточным лингвистическим покрытием предметной области. В другом случае пользователь может столкнуться с ситуацией, когда необходимо исправить концептуальную проблему с запросом. Интерфейс может предоставить пользователю подсказки по исправлению запроса. Например, упростить синтаксис запроса или перестать использовать неизвестные системе слова (Androutsopoulos I., et al., 1995).

- Необходимость предварительной настройки. Перед началом работы естественно-языковой пользовательский интерфейс должен быть предварительно настроен. В ряде систем управления базами данных, включающих в себя интерпретаторы формального языка запросов или включающих в себя реализацию пользовательских интерфейсов с использованием запроса, формируемого посредством заполнения форм, данная настройка не требуется.

- Излишняя свобода и неоднозначность естественного языка. Естественный язык достаточно многословный и многозначный, чтобы применять его в рамках человеко-машинного взаимодействия (Binot J.L., et al., 1991). Пользователям естественно-языкового пользовательского интерфейса требуется вводить длинный запрос, в то время как в графических интерфейсах запрос можно составить за несколько кликов мыши. Естественный язык многозначен, а интерфейсы с формальным языком запроса, с заполнением форм или графическим формированием запроса не допускают многозначного трактования.

- Произвольность в именовании сущностей в базе данных. Одни и те же поля и таблицы могут иметь различные названия в зависимости от разработчика базы данных. В результате, пользователь естественно-языкового

интерфейса и разработчик базы данных могут применять разную терминологию и понятия в отношении одних и тех же сущностей, хранящихся в базе данных.

- Завышенные ожидания пользователя от естественно-языкового пользовательского интерфейса. Так как пользователю до конца не известен и не понятен механизм обработки естественного языка, то пользователь может представлять интерфейс как некую интеллектуальную систему, обладающую здравым смыслом и позволяющую самостоятельно продуцировать новые знания на основе уже имеющихся данных (Hendrix G.G., 1982). Данная проблема не является актуальной для интерфейсов с формальным языком запроса, с заполнением форм или графическим формированием запроса, где возможности и ограничения системы более очевидны и наглядны для пользователя (Androutsopoulos I., et al., 1995).

1.4 Анализ существующих реализаций естественно-языковых пользовательских интерфейсов

Исследованием и разработкой естественно-языковых пользовательских интерфейсов активно занимаются ученые по всему миру, предлагая различные подходы. Традиционно, в компьютерной лингвистике наиболее развитыми являются разработки для английского языка. Данный факт обусловлен широким распространением и применением языка в мире. Тем не менее, естественно-языковые интерфейсы базы данных существуют и для русского языка. Сравнительный анализ существующих реализаций естественно-языковых пользовательских интерфейсов базы данных представлен в таблице 1.1.

Система ЗАПСИБ (Нариньяни А.С., 1979) – конструктор лингвистических процессоров, который основан на семантически-ориентированном анализе русскоязычного запроса. Система предоставляет пользователю возможность вести диалог посредством ограниченного подмножества естественного языка. Предметная область не расширяется.

Таблица 1.1 – Сравнительный анализ естественно-языковых пользовательских интерфейсов к базе данных

ЕЯ-интерфейс к БД	Язык	Расширение предметной области	Поддерж. более одной СУБД	Полнота используемых ЕЯ-конструкций	Поддерж. реляцион. СУБД	Поддерж. NoSQL СУБД
ЗАПСИБ	RU	–	+	–	+	–
InBASE	RU	+	+	+	+	–
SHRDLU	EN	–	–	–	–	–
English Query	EN	+	–	+	+	–
LUNAR	EN	–	–	+	+	–
LIFER / LADDER	EN	–	–	–	+	–
English Wizard	EN	+	+	+	+	–
NaLIR	EN	+	+	+	+	–
Sqlizer	EN	+	+	+	+	+/-
(Gadekar M. D. et al, 2015)	EN	+	–	–	–	+
(Никонов В.О., 2007)	RU	+	–	+	+	–
(Евдокимова И.С., 2004)	RU	+	+	–	+	–
(Правиков А. А., 2011)	RU	+	+	+	+	–

Коммерческий продукт InBASE (Жигалов В.А., и др., 2001). Система предполагает ограничения на используемые пользователем естественно-языковые конструкции. Расширение предметной области предполагается за

счет ручного формирования моделей. В результате работы системы формируется SQL-запрос к реляционной СУБД.

SHRDLU – ранняя программная система понимания естественного языка (Winograd T, 1972). Программная система не является в чистом виде естественно-языковым интерфейсом и имитирует поведение робота, манипулирующим блоками на столе. Она может управляться с помощью команд, сформированных из выражений английского языка. Тем не менее, система представлена в сравнительном анализе, так как ее реализация данной системы продемонстрировала способность понимать естественный язык. SHRDLU обладает существенными ограничениями и управляется с помощью небольшого количества предложений. Предметную область невозможно расширить. Функционал по формированию запросов к базе данных отсутствует.

Естественно-языковой пользовательский интерфейс English Query (Hamilton J. R., et al., 2001) основан на шаблонах и работает с запросами на английском языке. Система поставляется как часть Microsoft SQL Server и имеет существенное ограничение – работает только с данной СУБД. На рынке представлены бесплатные и платные версии Microsoft SQL Server.

LUNAR – естественно-языковой интерфейс базы данных, работающий с использованием процедурной семантики и расширенной сети переходов. Система была способна обработать без ошибок 78% запросов, и до 90% после исправления ошибок (Woods W.A., et al., 1977). Однако, расширение системы было невозможно и LUNAR был неспособен обработать запросы, выходящие за пределы предметной области (Woods W.A., et al., 1972). Система работает с базой данных, содержащей данные об анализах химических веществ, собранных космическим кораблем «Аполлон-11», совершившим посадку на поверхность Луны.

Система LIFER/LADDER (Hendrix G.G., et al., 1978) представляет собой естественно-языковой интерфейс к базе данных кораблей военно-морских сил США. Соответственно, обрабатывает текст на английском языке. В

семантической грамматике системы вместо синтаксических меток «глагол» и «существительное» использовались сущности «Корабль» или «Характеристика». Таким образом, система не могла быть расширена и была тесно привязана к предметной области. Также данный естественно-языковой интерфейс поддерживает запросы только к одной таблице, либо запросы к нескольким таблицам, но с достаточно простыми операциями объединения (Sathick K., et al., 2015).

Система English Wizard (Linguistic Technology, 1997) является коммерческой разработкой. По заявлениям разработчика, естественно-языковой интерфейс позволяет формировать SQL-запрос на основе запроса на английском языке без ограничения по используемым выражениям. Предметная область не ограничивается, при этом методика расширения предметной области не раскрывается.

Проект NaLIR (Li F., et al, 2014) обрабатывает сложные естественно-языковые запросы на английском языке и предполагает построение дерева зависимостей, использование правил и эвристик при разборе запроса. Система способна обладает высокой полнотой обрабатываемых естественно-языковых конструкций и может обработать сложные предложения. Расширение предметной области осуществляется в ручном режиме. В результате работы формируется SQL-запрос, направляемый к реляционной СУБД.

Естественно-языковой интерфейс к базе данных Sqlizer (Yaghmazadeh N., et al., 2017) использует методы машинного обучения для формирования SQL-запросов на основе естественно-языкового представления. Данная система является наиболее перспективной, среди представленных аналогов, работающих с английским языком. Предметная область системы может быть расширена в автоматизированном режиме. Существенных ограничений, касаемых полноты обрабатываемых естественно-языковых конструкций исследователями не заявлено. В результате работы система формирует SQL-запрос для реляционной СУБД. Однако разработчиками предусмотрена

возможность расширения системы для формирования запросов других видов, как например, запросов к NoSQL базам данных.

Система (Gadekar M. D. et al, 2015) содержит в себе ряд ограничений по обрабатываемым естественно-языковым конструкциям на английском языке. Тем не менее, данный интерфейс обеспечивает формирования запроса к NoSQL базе данных MondoDB. Расширение предметной области влечет за собой существенный объем ручной работы.

Работа (Никонов В.О., 2007) представляет собой диалоговую систему для ускорения работы экспертов при ответе на типовые запросы пользователей. Функциональность системы по обработке естественно-языкового русскоязычного пользовательского запроса может быть использована при реализации интерфейса к базе данных. Система реализована для применения в торговых предприятиях, занимающихся продажей алкогольной продукции и расширение предметной области затруднительно, так как несет в себе существенный объем ручной работы по формированию экспертами карты запросов.

Естественно-языковой интерфейс, представленный в работе (Евдокимова И.С., 2004), предназначен для работы с запросами на русском языке. Система работает только с реляционными базами данных, формируя в результате работы SQL-запрос, а также обладает рядом ограничений по отношению к содержимому естественно-языкового пользовательского запроса. Предусмотрено расширение предметной области, однако при интеграции с каждой базой данных потребуется значительное количество ручной работы по формированию необходимых для работы интерфейса данных.

Работа, описанная в (Правиков А. А., 2011), представляет собой естественно-языковой интерфейс рекомендательной системы для сайта с автомобильной тематикой. Система достаточно сильно связана с предметной областью, однако, в целом, допускает расширение предметной области. Тем не менее, для этого требуется достаточно большое количество ручных действий. Также, система может работать только с реляционными СУБД

допуская возможность построения исключительно SQL-запроса к базе данных.

В результате проведенного сравнительного анализа можно сделать вывод, что естественно-языковые пользовательские интерфейсы к базе данных, ориентированные для обработки русскоязычных запросов, отстают в своем развитии от англоязычных аналогов. Для устранения данного разрыва необходимо обеспечить интерфейс возможностью формирования и расширения предметной области в автоматизированном режиме, а также обеспечить возможность формирования запросов на других формальных языках запроса, как, например, запросы к NoSQL базам данных.

Выводы

В первой главе представлен обзор исследований в области обработки естественного языка и построения пользовательских интерфейсов к базам данных. Рассмотрены существующие виды пользовательских интерфейсов к базам данных, а также их преимущества, особенности и проблемы реализации.

В результате проведенного исследования выявлено противоречие, состоящее в том, что количество и разнообразие данных неуклонно возрастает, в то время как все чаще возникает необходимость доступа к данным для пользователя, не обладающего специальными знаниями и подготовкой для формирования запросов с использованием формального языка. Таким образом, задача разработки естественно-языкового пользовательского интерфейса к базам данных является актуальной. В результате проведенного сравнительного анализа существующих реализаций естественно-языковых пользовательских интерфейсов к базам данных обоснована целесообразность автоматизации возможности формирования и расширения предметной области, а также обеспечения возможности формирования запросов на других формальных языках запроса.

Глава 2. Разработка метода построения естественно-языкового пользовательского интерфейса к базам данных

2.1 Разработка системы критериев качества естественно-языковых пользовательских интерфейсов

В качестве основных критериев качества естественно-языковых пользовательских интерфейсов базы данных рассматриваются следующие факторы (Androutsopoulos I., et al., 1995):

1. Дружественность
2. Корректность
3. Гибкость
4. Полнота
5. Портитруемость
 - на другую предметную область;
 - на другой язык;
 - на другой формальный язык запроса.

Дружественным является интерфейс удобный и комфортный для работы пользователя. Также в данное понятие включается наличие возможности сообщения пользователю о проблемах с пониманием запроса в процессе его обработки. В подобной ситуации дружественный интерфейс должен как минимум сообщить о наличии проблемы, а лучше – предоставить обратную связь. Например, естественно-языковой пользовательский интерфейс может предложить варианты, как можно переформулировать запрос таким образом, чтобы он был успешно обработан, и пользователь смог найти ответ на свой вопрос.

Корректность – представляет собой свойство естественно-языкового интерфейса правильно распознать и верно интерпретировать суть запроса пользователя. В русскоязычных работах, посвященных исследованию естественно-языковых интерфейсах встречается упоминание данного критерия в качестве понятия «надежность». В частности, подобная терминология встречается в работах Житко (Житко В.А. и др. , 2011) и

Жигалова (Жигалов В.А., 1997). Для исключения путаницы с терминами из теории надежности систем принято решение об именовании данного критерия как «корректность». В данном случае важно, чтобы пользователь также корректно осуществлял формулирование своего намерения по получению интересующей информации из базы данных. Корректность естественно-языкового пользовательского интерфейса оценивается на двух этапах работы системы. Это этап обработки запроса на естественном языке, а также этап дальнейшего преобразования в запрос к базе данных на формальном языке.

Естественно-языковой пользовательский интерфейс способен обрабатывать некоторый объем пользовательских запросов. Величина объема правильно обрабатываемых и верно интерпретируемых пользовательских запросов представляет собой полноту естественно-языкового пользовательского интерфейса к базам данных. Полноту естественно-языкового пользовательского интерфейса можно вычислить с помощью выражения

$$R = \frac{N_{Correct}}{N_{All}},$$

где R (recall, полнота) – отношение количества корректно выданных результатов ($N_{Correct}$) к общему количеству возможных корректных результатов (N_{All}) (Николаева И.С., и др., 2016).

Естественно-языковой интерфейс является гибким в том случае, когда позволяет одинаково успешно обрабатывать различные группы запросов, которые имеют идентичную структуру. В данном случае речь идет о запросах, которые содержат в себе эллипсис, анафору (лексический повтор) или вложенные подзапросы. То есть более сложные для обработки типы запросов. Например, о существовании эллипсиса в запросе можно догадаться в большинстве случаев на основе анализа контекста. А это, в свою очередь, порождает дополнительный этап в обработке естественного языка. В то же время при наличии вложенного подзапроса в пользовательском запросе естественно-языковой интерфейс должен уметь декомпозировать запрос на

составные части. В дальнейшем каждый из подзапросов обрабатывается по отдельности.

Естественно-языковой пользовательский интерфейс, обладающий возможностью портируемости на другую предметную область, способен работать с терминами и понятиями, применяемыми в новой предметной области. При портировании на другую предметную область требуется формирование новой модели предметной области и новой модели базы данных. От разработчика естественно-языкового интерфейса может потребоваться доработка программного кода, от специалиста, отвечающего за инженерию знаний – построение схем для предметной области базы данных, а также описание новых концептов. При этом администратору базы данных требуется подготовить описание важных характеристик новой предметной области на уровне существующих таблиц и полей в базе данных.

Еще один важный критерий качества естественно-языкового интерфейса – портируемость на другой язык. Так как самым распространенным языком на настоящий момент является английский, то для этого языка проводится наибольшее количество исследований и разработано большинство естественно-языковых пользовательских интерфейсов. Возможность настройки интерфейса на другой естественный язык представляет собой актуальную задачу. Данный факт связан с активным использованием морфологических и синтаксических правил (Bessmertny I. A. et al. , 2016), специфичных для конкретного языка.

Другим показателем, характеризующим качественный естественно-языковой интерфейс, является портируемость на другой формальный язык запроса. В случае, когда естественно-языковой интерфейс в процессе своей работы формирует SQL-запрос к базе данных, то данный интерфейс может быть портирован на другую систему управления базами данных, которая поддерживает работу с SQL-запросами. В то же время при портировании естественно-языкового интерфейса для работы с NoSQL базами данных или XML-документами, то у естественно-языкового интерфейса достаточно

доработать или заменить только подсистему, отвечающую за формирование запроса на формальном языке. В таком случае в систему уже предварительно должен быть заложен гибкий механизм взаимодействия между внутренними подсистемами, который не привязан к конкретной реализации формального языка запроса и позволяет реализовать подобные доработки наименьшими усилиями (Жигалов В.А., 1997).

2.2 Анализ существующих методов обработки естественного языка

Сегодня существуют различные подходы к обработке естественного языка. Первый подход основан на синтаксическом анализе предложения. На данном этапе выделяются отношения синтаксических связей внутри предложения. Следующим этапом происходит определение типа предложения, а также главных и второстепенных членов предложения. На этом этапе используются лексические и синтаксические правила специфичные для анализируемого языка (Житко В.А. , 2012). Процесс синтаксического анализа предложения на естественном языке осуществляется в несколько этапов и последовательным образом. Таким образом, используются данные, полученные на предыдущем этапе морфологического анализа и в результате, происходит построение структуры предложения. Этапы морфологического и морфемного анализа в рамках обработки естественного языка происходят перед этапом синтаксического анализа. На этапе синтаксического анализа для каждого слова, входящего в анализируемое предложение, извлекаются отношения между сущностями предложения, которые формируют соответствия значениям грамматических категорий (Крайванова В.А., 2009).

Морфологические характеристики каждого слова определяются на этапе морфологического анализа. К таким характеристикам относятся часть речи, склонение, падеж и т.д. (Посевкин Р.В., Бессмертный И.А., 2016).

В зависимости от языка может быть различным как количество, так и вообще существование различных морфологических характеристик слова.

Также различаются и допустимые значения этих морфологических характеристик для разных языков. Тем не менее существуют общая часть характеристик, присутствующих во многих языках, например часть речи.

Существуют различные подходы, позволяющие провести морфологический анализ естественно-языкового текста на русском языке:

- вероятностный подход;
- «нечеткая» морфология;
- «четкая» морфология.

Подход, базирующийся на применении «четкой» морфологии наиболее распространен в процессе обработки естественно-языковых текстов на русском языке. Подход, с использованием «четкой» морфологии использует данные грамматического словаря Зализняка (Зализняк А.А., 1967), который содержит в себе основные словоформы русскоязычных слов, каждой из которых соответствует определенный код.

Также существуют правила, на основании которых для конкретного слова можно вывести все остальные возможные формы слова (Селезнев К., 2003). При проведении морфологического анализа с использованием «четкого» подхода, используется словарь, содержащий словоформы слов анализируемого языка и соответствующие им коды (Якубенко А.П., и др., 2006). В результате использования подобного словаря для каждой конкретной словоформы возможно получить ее морфологические характеристики.

Формирование подобного словаря на базе грамматического словаря Зализняка производится следующим образом. На первом этапе формирования словаря перебираются все содержащиеся в словаре слова, для каждого из которых формируется множество словоформ. Сформированные подобным образом словоформы вносятся в данный словарь и в будущем извлекаются оттуда при проведении морфологического анализа естественно-языкового текста на русском языке. Таким образом будут получены морфологические характеристики анализируемого слова (Посевкин Р.В., 2016).

Каждое слово в русском языке может быть декомпозировано на отдельные составные части. Такие составные части слова называются морфемами, в частности: приставка, окончание, корень, суффикс, основа. Подобное разбиение слова на составные части производится в рамках морфемного анализа. Для каждого слова русского языка, включенного в словарь морфем (Кузнецова А.И., и др., 1986) представлено разделение на отдельные части, но при этом не представлено описание того, какой это тип у каждой из частей. В результате, на основе информации, представленной в словаре морфем русского языка, невозможно сделать вывод какая из составных частей слова представляет собой приставку, а какая – корень слова (Тихонов А.Н., 2002).

Совокупность всех корней слов русского языка представляют собой неограниченное по объему множество. В то же время для таких морфем, как окончания, приставки и суффиксы вариативность возможных значений ограничена. Для русского языка четко определена последовательность нахождения морфем в рамках слова. В результате, слово начинается с приставки, за которой следует корень, завершающийся суффиксом и окончанием. При этом, в слове может отсутствовать как приставка, так и суффикс, а окончание может быть нулевым. В результате, возможно построение словаря, включающего декомпозицию слова русского языка на составные конструкции, а также описание каждой представленной морфемы на основе словаря морфем. Таким образом, разработанный словарь можно применить на этапе морфемного анализа слова.

При возникновении ситуации отсутствия анализируемого слова в морфемном словаре, возможно непосредственное провести анализ, ориентируясь как на рассмотренную ранее стандартную последовательность нахождения морфем в русскоязычном слове, так и на совокупность всех возможных приставок, суффиксов и окончаний, вариативность возможных значений которых ограничена (Елисеева О.Е., 2009).

Недостатком подхода, основанного исключительно на синтаксическом анализе, является отсутствие в результате разбора запроса информации о его смысле. Для устранения данной проблемы применяется подход, основанный на семантике. В рамках данного подхода включается этап семантического анализа текстового запроса. Таким образом, при обработке естественно-языкового пользовательского запроса последовательно выполняются морфологический, синтаксический и семантический анализ. Схема метода обработки естественно-языкового запроса пользователя представлена на рисунке 2.1.

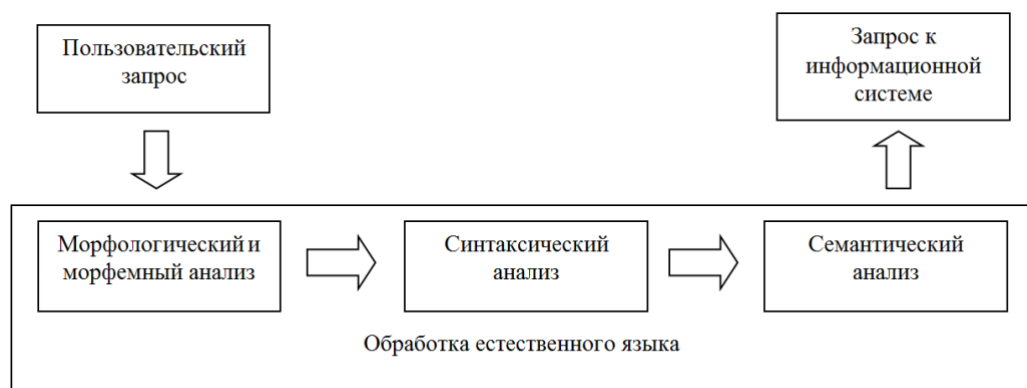


Рисунок 2.1 – Схема метода обработки естественно-языкового запроса, основанного на семантике

Семантический анализ представляет собой процесс обработки семантической сети, созданной на основе сведений о предметной области и языке, имеющимися в системе, а также результатов анализа запроса на предшествующих этапах обработки естественно-языкового русскоязычного текста. При построении семантической модели слова учитывается его многозначность, а сам смысл слова представляет собой результирующее множество различных интерпретаций, реализуемое в определенном контексте. На этапе семантического анализа выявляются отношения лингвистической сущности, а также сущностей, которые созданы для выявления отношений соответствия.

Также для решения проблемы неоднозначности при интерпретации смысла слова может быть использован тезаурус, который включает в себя сеть с узлами в виде слов и их смыслов. При этом в некоторых случаях

семантические конструкции могут содержать достаточное большое количество смыслов и их перебор может потребовать значительного времени (Gong Y., et al. , 2001). Также тезаурусы должны быть своевременно актуализированы, чтобы включать в себя вновь появляющиеся в современном обиходе слова – неологизмы. Результатом семантического анализа является результат разбора запроса, включающей в себя информацию о его смысловой оценке.

При реализации компоненты взаимодействия пользователя с программной системой естественно-языкового интерфейса можно использовать подмножество естественного языка, включающее в себя ограниченный набор грамматики и лексики. Несмотря на сокращение объема инструкций естественного языка, с помощью которых пользователь может формировать запрос к системе, значительного ухудшения функциональности и производительности интерфейса не происходит. Использование ограниченного подмножества естественного языка вместе с сохранением простоты восприятия не требует дополнительных трудозатрат по изучению со стороны пользователя. Преимуществом использования сокращенного языка является сокращение времени, требуемого для анализа и обработки естественного языка, а также разрешения проблемы неоднозначной интерпретации смыслов конструкций естественного языка, таких как омонимия (Deshpande A.K., et al., 2012).

Использование ограниченного естественного языка повлекло за собой возникновение еще одного подхода к обработке естественно-языкового текста – подход на основе моделей предложений. В данном случае возможные варианты пользовательских запросов к интерфейсу к базе данных формализуются в виде шаблонных конструкций (Deemter K.V., et al., 2005). В результате, при использовании данного метода происходит извлечение конструкций естественного языка, необходимых для формирования запроса к базе данных. Таким образом, этапы морфологического, морфемного и синтаксического анализа могут не потребоваться, при достаточной полноте

охвата естественно-языкового запроса шаблонами моделей предложений. Преимуществом данного подхода является более высокая скорость обработки запроса, по сравнению с полным анализом естественного языка. Однако, шаблонами моделей предложений должны охватывать все множество возможных запросов пользователей. Тем не менее, запросы к естественно-языковому интерфейсу представляют собой подмножество естественного языка, которое возможно формализовать с помощью ограниченного количества шаблонов.

2.3 Разработка метода обработки естественно-языкового запроса с использованием шаблонов моделей предложений

Для извлечения ключевых сущностей из естественно-языкового пользовательского запроса используются шаблоны модели предложений. В разработанном методе используются шаблоны двух видов:

1. Шаблоны выделения именованных сущностей.
2. Шаблоны общего уровня.

В процессе обработки естественно-языкового запроса выполняется выделение в тексте именованных сущностей и связанных с ними событий. Именованные сущности могут представлять собой имена персоналий, названия различных компаний, географические наименования. В настоящее время проводятся исследования (Daud A., et al., 2010), ставящие целью формализацию и формирование единого списка категорий именованных сущностей. Тем не менее, существует ряд достаточно важных сущностей, которые встречаются в большом количестве текстов, и которые также можно отнести к именованным сущностям, например (Большакова Е.И., и др., 2017):

- Время: с 10:00 до 19:00, 22:30;
- Даты: 01.04.1970, 60-е гг., 5 августа
- Адреса: ул. Ленина, д. 15, корп. 2, кв. 150;
- Денежные суммы и денежные единицы: \$50, JPY, руб.;
- Номера телефонов: +7 921 234-56-78; 8 (495) 231 56 34

- Профессии и должности: менеджер проектов, врач, начальник отдела.

Таблица 2.1 – Шаблоны моделей предложений общего уровня

Шаблон	Отношение	Конструкция формального языка запроса
<i>сколько [condition]</i>	Количество	COUNT([condition])
<i>в среднем [condition] //</i> <i>среднее количество[condition] //</i> <i>среднее значение[condition]</i>	Среднее количество	AVG([condition])
<i>сумма [numeric] //</i> <i>сколько всего [numeric]</i>	Сумма	SUM([numeric])
<i>от [numeric-1] до [numeric-2]</i>	Вхождение в интервал	BETWEEN [numeric-1] AND [numeric-2]
<i>существует ли [condition] //</i> <i>есть ли [condition]</i>	Существование	EXISTS([condition])
<i>[condition-1], который</i> <i>[condition-2]</i>	Вложенное условие	[condition-1] WHERE [condition-2]
<i>[condition-1] и [condition-2]</i>	Конъюнкция	[condition-1] AND [condition-2]
<i>[condition-1] или [condition-2]</i>	Дизъюнкция	[condition-1] OR [condition-2]
<i>не [condition]</i>	Отрицание	NOT([condition])

Шаблоны выделения именованных сущностей описывают синтаксическую структуру естественно-языкового текста, в рамках формализации которой приводится описание связей между словами предложения, а также ряда условий, которые накладываются на слова. Условия, задаваемые в рамках шаблонов, описывают морфологические, смысловые и семантические характеристики слова, которые, в свою очередь, могут быть выведены из других шаблонов. Если все определенные в шаблоне условия истинны, то для текста производится формализация шаблона

(Селезнев К., 2003). Шаблоны выделения именованных сущностей используются в ряде программных средств обработки естественного языка, таких как Томи́та-парсер (Большакова Е.И., 2014), RCO (Ермаков А.Е., и др., 2003), Alex (Толдова С.Ю., и др. , 2002). В рамках разрабатываемого естественно-языкового интерфейса для выделения именованных сущностей применяется Томи́та-парсер.

Шаблоны общего уровня применяются к предложению после применения шаблонов выделения именованных сущностей. Данный вид шаблонов позволяет выявить ключевые конструкции, необходимые для дальнейшего корректного формирования запроса на формальном языке. Шаблоны общего уровня позволяют идентифицировать такие отношения как *существование*, *количество*, *среднее количество*, *сумма* и другие. Примеры подобных шаблонов представлены в таблице 2.1.

Выводы

В результате проведенного исследования сформулированы критерии качества естественно-языковых пользовательских интерфейсов базы данных. Разработана метод построения естественно-языкового интерфейса, позволяющий автоматически извлекать данные из базы данных без необходимости ручного формирования пользователем SQL-запроса. В отличие от существующих решений, для обработки естественно-языкового запроса применяются шаблоны моделей предложений, что обеспечивает идентификацию отношений между ключевыми терминами естественного языка для построения запроса к базе данных.

Представленные в главе результаты опубликованы в (Посевкин Р.В., Бессмертный И.А., 2016), (Posevkin R., Bessmertny I., 2016), (Posevkin R., Bessmertny I., 2015).

Глава 3. Разработка семантической модели базы данных и алгоритма ее автоматизированного формирования

3.1 Обоснование необходимости семантической модели базы данных. Требования к семантической модели базы данных

В предыдущей главе разработан метод обработки естественно-языкового запроса, позволяющий выделить значимые сущности, на основе которых в дальнейшем может быть построен запрос к базе данных. Непосредственно после этапа выделения ключевых сущностей из естественно-языкового запроса возникает проблема несоответствия терминов естественно-языкового запроса (Бессмертный И.А. и др., 2017) сущностям даталогической модели базы данных.

Для решения данной проблемы применяются различные подходы. Для естественно-языковых интерфейсов к базе данных, обрабатывающих запросы на английском языке, задача устранения указанной коллизии несколько упрощается. В данном случае названия таблиц и полей базы данных представлены на английском языке, так же, как и естественно-языковой запрос пользователя. В результате, существуют подходы, в которых для сопоставления терминов пользовательского запроса и сущностей даталогической модели базы данных применяется расстояние Левенштейна (Ramesh S.H., et al., 2016). Подобный подход демонстрирует достаточно высокую эффективность при использовании в естественно-языковых интерфейсах к документо-ориентированным базам данных (Gadekar M. D. et al, 2015). Также существует подход, в котором проблема решается с использованием методов машинного обучения (Yaghmazadeh N., et al., 2017).

При реализации русскоязычных естественно-языковых интерфейсов к базе данных может применяться подход с предварительным формированием экспертом карты запросов (Никонов В.О., 2007). Также в работах (Евдокимова И.С., 2004) и (Правилов А.А., и др., 2010) предлагается предварительно формализовать знания об элементах базы данных.

Для решения проблемы несоответствия терминов естественно-языкового запроса сущностям даталогической модели базы данных в рамках данного исследования предлагается использовать семантическую модель базы данных. Содержимое семантической модели базы данных представлено на рисунке 3.1.

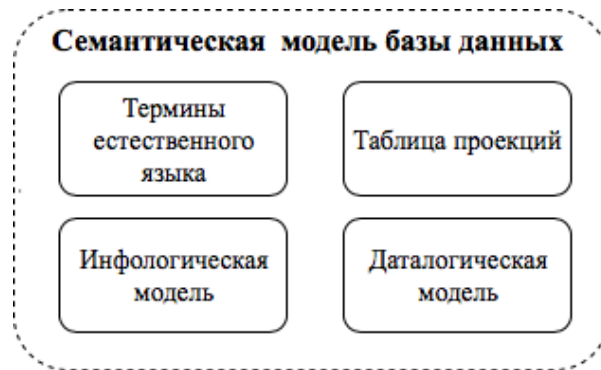


Рисунок 3.1 – Содержимое семантической модели базы данных

Семантическая модель базы данных состоит из следующих компонент:

- инфологическая модель базы данных;
- даталогическая модель базы данных;
- термины естественного языка;
- таблица проекций.

Инфологическая модель базы данных представляет собой описание предметной области. Основные элементы инфологических моделей – сущности, их атрибуты, связи между сущностями.

Даталогическая модель базы данных отражает логические взаимосвязи между элементами данных безотносительно их содержания и физической организации, разрабатывается с учётом реализации конкретной СУБД и конкретной предметной области на основе ее инфологической модели. Даталогическая модель включает в себя схемы отношений с указанием первичных ключей, а также связи между отношениями, реализованные посредством внешних ключей.

Термины естественного языка представляют собой множество слов, которые могут быть использованы пользователем при формировании запрос на естественном языке. Данное множество может быть сформировано вручную, а также расширено с помощью таблиц синонимов и тезаурусов.

Тезаурус представляет собой набор сведений, охватывающих определения и термины специальной области знаний или сферы деятельности. Тезаурус включает в себя описание семантических отношений, таких как синонимы, паронимы, антонимы и прочее. Также тезаурус помогает решить задачу по выявлению смысла не только с помощью определений, но и с помощью соотнесения слова с другими понятиями и группами понятий.

Таблица проекций включает в себя описание отношений между терминами естественного языка, инфологической и даталогических моделей базы данных.

Наличие семантической модели базы данных также позволяет обеспечить портируемость естественно-языкового пользовательского интерфейса на другую предметную область.

Таким образом, семантическая модель базы данных должна включать в себя следующие группы данных:

1. отношение тождества между термином естественно-языкового запроса и термином инфологической модели;
2. отношение подобия между термином-синонимом инфологической модели и термином инфологической модели базы данных;
3. отношение подобия между термином-синонимом естественно-языкового запроса и термином инфологической модели базы данных;
4. отношение номинации между термином естественно-языкового запроса и множеством терминов инфологической модели базы данных;
5. отношение агрегации между множеством терминов естественно-языкового запроса и термином инфологической модели базы данных;
6. отношение подобия между терминами инфологической и даталогической моделей базы данных (Евдокимова И.С., 2004).

Таким образом, K_{input} представляет собой множество терминов естественно-языкового пользовательского запроса, K_{info} – множество терминов инфологической модели базы данных, K_{data} – множество терминов даталогической модели базы данных. Задача семантической модели базы –

связать термин естественно-языкового запроса ($x \in K_{input}$) с термином даталогической модели ($z \in K_{data}$) посредством термина инфологической модели ($y \in K_{info}$). В результате, ранее представленные группы данных, включаемые в семантическую модель базы данных, могут быть представлены с помощью предикатов в следующем виде:

1. $\forall x K_{input}(x) \equiv \exists y K_{info}(y)$
2. $\forall y' P_{syn}(K_{info}(y), K_{info}(y')) \rightarrow \exists y K_{info}(y)$
3. $\forall x, x' P_{syn}(K_{input}(x), K_{input}(x')) \rightarrow \exists y K_{info}(y)$
4. $\exists x K_{input}(x) \rightarrow \exists y_1 \dots y_i K_{info}(\{y_1 \dots y_i\})$
5. $\exists x_1 \dots x_i K_{input}(\{y_1 \dots y_i\}) \rightarrow \exists y K_{info}(y)$
6. $\forall y K_{info}(y) \rightarrow \exists z K_{data}(z)$

Предикат P_{syn} отображает синонимичность терминов, входящих, в данном случае, в множество терминов естественно-языкового пользовательского запроса или инфологической модели базы данных.

3.2 Структурный синтез семантической модели базы данных

В семантической модели описываются различные сущности, сведения о которых содержатся в базе данных. Помимо этого, семантическая модель содержит отношения между этими сущностями. Данные отношения аналогичны связям в диаграммах сущность-связь (ER-диаграммы) (Вендров А.М., 1998). Пример данной диаграммы, описывающий отношения между подразделением и сотрудником представлен на рисунке 3.2.



Рисунок 3.2 – Диаграмма сущность-связь для описания отношения между сотрудником и подразделением

При этом отношение связи является двунаправленным. Связи между сущностями могут быть вида 1:1 (один к одному), 1:N (один ко многим), N:M (многие ко многим). При этом возможно наличие вычисляемых сущностей.

Значение вычисляемой сущности формируется на основе значений связанных сущностей (Nihalani N., et al., 2011). В качестве примера вычисляемой сущности можно привести сущность ФИО, которая формируется путем конкатенации значений сущностей «Фамилия», «Имя», «Отчество»:

$$\text{ФИО} = (\text{Фамилия: строка}, \text{Имя: строка}, \text{Отчество: строка}).$$

Также семантическая модель базы данных может содержать в себе последовательности данных, которые представляют собой список из упорядоченных элементов. Порядок внутри подобной последовательности может быть задан произвольным образом. Последовательности описывают определенную градацию и шкалы, которые не представлены в явном виде в базе данных. Например, последовательность, содержащая в себе иерархию уровней образования, может иметь значение для запросов вида «*сотрудники с образованием выше средней школы*».

Также семантическая модель базы данных может включать множества. Множества позволяют определять набор сущностей в рамках определенного класса. Множества позволяют упростить формирование составных и вложенных SQL-запросов с использованием семантической модели базы данных (Pan S., et al., 2007). Например, такое множество как «*Мужчины*» для базы данных, содержащих сведения о сотрудниках, может быть определено как «*подмножество множества Сотрудники, у которых пол мужской*» (Жигалов В.А., 2000). Данное множества можно формализовать в следующем виде:

$$\exists y: x \in E \cup s_x \in M$$

Создание подобного именованного набора может быть выполнено следующим образом:

```
create set Men
from Employee x
where x in Employees
and x.sex in Male
```

В естественно-языковых пользовательских запросах могут содержаться ограничения числовых значений для ряда параметров, таких как возраст сотрудника. Например, такие ограничения в естественно-языковом запросе могут быть сформулированы посредством выражения «возрастом от 25 до 40 лет». Выражения, которые позволяют сформировать ограничения, могут быть включены как в простые фразы, так и в более сложные естественно-языковые конструкции (Фомичев В.А., 2007). Для формулирования числовых ограничений в естественно-языковом запросе может использоваться логическая связка «отрицание», например: «*не старше пятидесяти лет*». Помимо этого, возможно неявное указание единиц измерения в рамках запроса пользователя – «*сотрудники от 20 до 30*». В данном примере система необходимо совершить автоматическую подстановку единицы измерения, соответствующей данным, хранимым в базе данных. В данном случае – годы. Подобные данные также возможно включить в семантическую модель базы данных.

3.3 Автоматизация процесса формирования семантической модели базы данных

3.3.1 Разработка алгоритма формирования семантической модели базы данных

В процессе разработки диалоговой системы перед разработчиком может возникнуть задача реализовать естественно-языковой пользовательский интерфейс к уже существующей и заполненной содержимым базе данных. В таком случае формирование семантической модели базы данных в ручном режиме требует значительного количества временных и трудовых затрат. В результате, стоимости разработки такой программной системы существенно увеличивается.

Для того, чтобы решить данную проблему, требуется разработка программной системы, позволяющей в автоматизированном режиме формировать семантическую модель для существующей и заполненной

содержимым базы данных. В самом общем виде данный процесс представлен на рисунке 3.3.

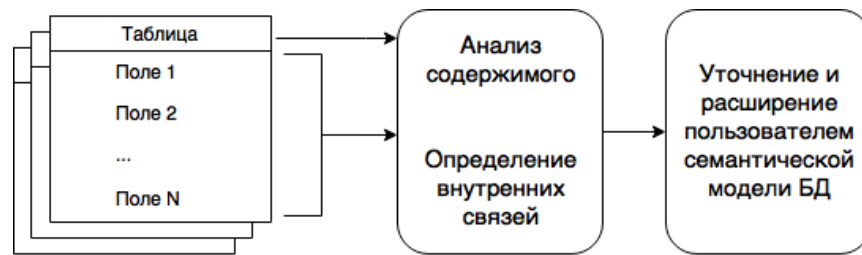


Рисунок 3.3 – Формирование семантической модели базы данных

На рисунке 3.4 представлена последовательность этапов, необходимых для успешного формирования семантической модели базы данных.

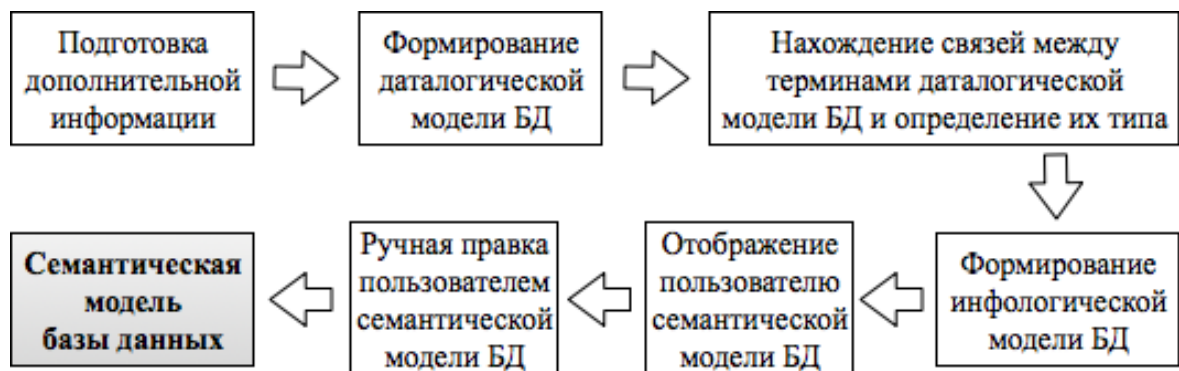


Рисунок 3.4 – Последовательность этапов формирования семантической модели базы данных

Таким образом, алгоритм формирования семантической модели базы данных можно формализовать в следующем виде:

1. Пользователь подготавливает источники дополнительных данных (тезаурусы, локализация), необходимые для анализа данных.
2. Извлекаются названия всех таблиц базы данных.
3. Для каждой таблицы извлекаются все поля.
4. На основе внешних ключей определяются связи между таблицами базы данных.
5. На основе анализа названий таблиц и их полей делается предположение о содержимом, определяется наличие и тип внутренних связей между сущностями базы данных.

6. Формируется инфологическая модель, добавляющая в семантическую модель знания о смысле и значении содержимого базы данных.

7. В ручном режиме пользователь дополняет и уточняет информацию в семантической модели базы данных.

Одна из возможных проблем, которая препятствует однозначному корректному определению и интерпретации содержимого базы данных, является многозначность естественного языка. При этом названия полей таблиц с одним и тем же содержимым могут различаться в зависимости от того, кто придумывал название для них. В решении этой проблемы может помочь использование тезаурусов, содержащих в себе сведения о всевозможных синонимах.

Также разработанный алгоритм формирования семантической модели может быть использован не только для начального формирования семантической модели, но и для ее дальнейшей актуализации, в случае нарушения целостности и консистентности данных семантической модели относительно содержимого базы данных.

3.3.2 Разработка программной системы автоматизированного формирования семантической модели базы данных

Разработана программная система, позволяющая в автоматизированном режиме сформировать семантическую модель базы данных. Алгоритм работы программы представлен на рисунке 3.5.

Перед началом работы с программной системой, пользователь может подключить источники дополнительных данных, которые будут использоваться системой в процессе анализа базы данных. В качестве такого источника данных может выступать тезаурус. Далее пользователь выбирает режим, в котором будет производиться подключение к базе данных. Программная система включает в себя механизм формирования тестовой базы данных. Подробная информация о структуре базы данных представлена в

разделе 5.1.2 диссертации. В качестве альтернативного варианта возможно подключение к уже существующей базе данных. Для этого пользователю потребуется ввести данные для подключения к базе данных, такие как наименование хоста, название БД, а также имя пользователя и пароль.



Рисунок 3.5 – Алгоритм работы программной системы автоматизированного формирования семантической модели базы данных

После успешного подключения к базе данных, пользователь также может указать дополнительные параметры для анализа базы данных. Например, локализацию данных, хранящихся в базе данных, чтобы при анализе применялись тезаурусы только для нужного языка. Далее происходит непосредственно анализ базы данных, после чего пользователь может

ознакомиться с получившейся семантической моделью базы данных. Если потребуются какие-либо изменения или дополнения, пользователь может внести самостоятельно необходимые правки.

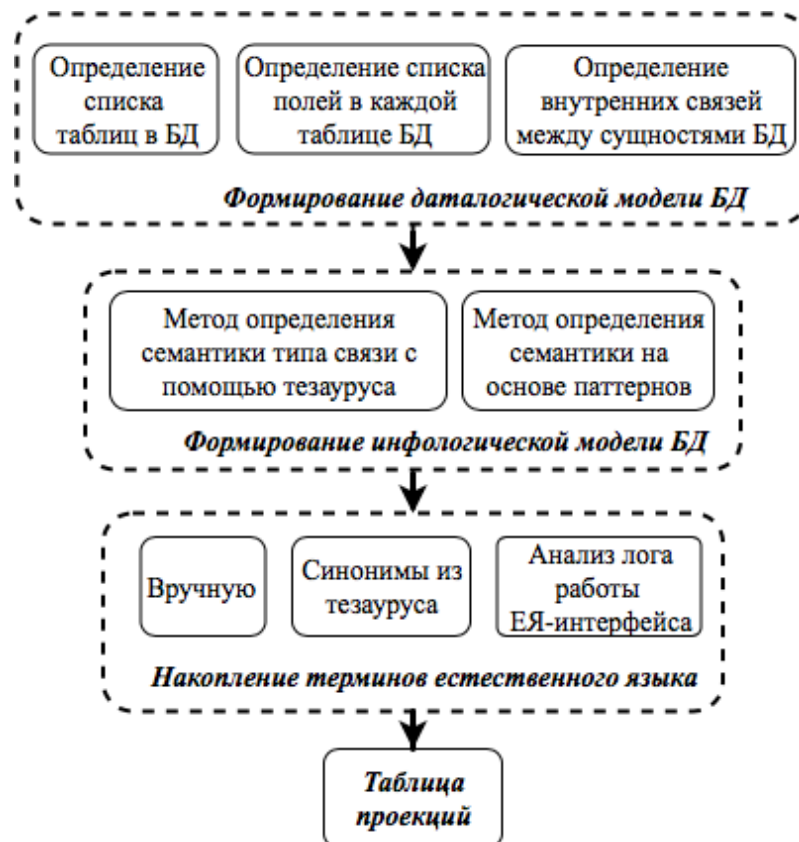


Рисунок 3.6 – Этапы анализа базы данных при формировании семантической модели

Шаг «Анализ БД» включает в себя несколько этапов анализа базы данных при формировании семантической модели, которые представлены на рисунке 3.6. Данный процесс состоит из следующих этапов:

1. Формирование даталогической модели базы данных.
2. Формирование инфологической модели базы данных.
3. Накопление терминов естественного языка.
4. Формирование таблицы проекций, отображающей соответствие терминов естественного языка, даталогической и инфологических моделей базы данных.

При этом, каждый из этих этапов включает в себя ряд независимых модулей. С целью дальнейшего совершенствования механизмов анализа базы данных для формирования семантической модели базы данных, список

модулей может быть расширен. Так как модули реализованы в виде независимых компонент, расширение списка правил и модулей не составляет труда и в системе предусмотрено дальнейшее масштабирование.

На этапе формирования даталогической модели базы данных определяется список всех таблиц, содержащихся в базе данных. Далее для каждой таблицы определяется список полей. После этого определяются внутренние связи между сущностями базы данных. Данные связи идентифицируются на основе внешних ключей таблиц. Также для идентификации возможных отношений между таблицами, не оформленных должным образом с помощью внешних ключей, применяются паттерны.

Используя набор паттернов, которые основаны на анализе именования таблиц и полей базы данных, возможно выделить внутренние связи между таблицами. В качестве примера такого паттерна можно привести шаблон, описывающий связь внешнего ключа и таблицы базы данных: *id_[tableName]* или *[tableName]_id*, где *[tableName]* – имя таблицы БД, с которой связан данный внешний ключ.

Рассмотрим данный процесс на примере, который изображен на рисунке 3.7. В данном примере представлена таблица базы данных *employee* – переменная отношения R_2 , и таблица *department*, которая является переменной отношения R_1 . Внешним ключом (FK, Foreign Key) таблицы *employee* является поле *id_department*. Значения данного поля должны совпадать со значениями переменной отношения R_1 . Переменная отношения R_1 является также потенциальным ключом (СК, Candidate Key). Поле *id* таблицы *department* является первичным ключом (ПК, Primary Key) и выступает в роли потенциального ключа. В результате, выполняются такие условия как:

1. В переменной отношения R_1 имеется потенциальный ключ СК такой, что СК и FK совпадают с точностью до переименования атрибутов.
2. В любой момент времени каждое значение FK в текущем значении R_2 идентично значению СК в некотором кортеже в текущем значении R_1 .

Таким образом, используя паттерн вида $id_tableName$ идентифицирована и извлечена взаимосвязь дочернего отношения R_2 (*employee*) по отношению к родительскому отношению R_1 (*department*).

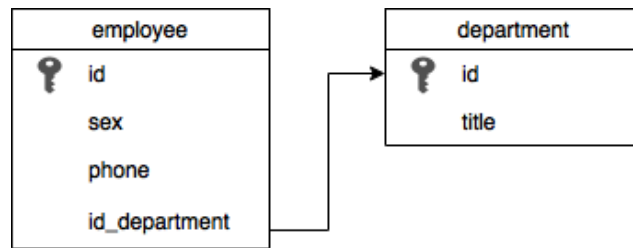


Рисунок 3.7 – Связь таблиц базы данных

В результате, в даталогической модели будет сохранено отношение связи между сущностями даталогической модели, имеющее следующий вид:

$$P_{link}(employee, department, id_department, id)$$

Этап формирования инфологической модели базы данных включает в себя метод определения семантики типа связи с помощью тезауруса, а также метод определения семантики на основе паттернов. Данные методы подробнее рассматриваются в разделах 3.3.3 и 3.3.4.

Далее при анализе базы данных следует этап накопления терминов естественного языка и формируется таблица проекций. Данный процесс рассматривается в разделе 3.3.5.

3.3.3 Разработка метода определения семантики типа связей между сущностями базы данных с использованием тезауруса

Семантическая модель базы данных представляет из себя важную составляющую естественно-языкового интерфейса к базе данных. Для решения проблемы несоответствия терминов естественно-языкового запроса сущностям даталогической модели базы данных необходимо формирование таблицы проекций. В частности, данная таблица позволяет произвести интерпретацию смыслового значения слов из естественно-языкового запроса пользователя в конкретные таблицы и поля базы данных, которые необходимо использовать в формальном запросе к базе данных. В результате, таблица

проекции позволяет обеспечить формирование запроса к базе данных с использованием терминов предметной области.

Для определения семантики данных, хранимых в том или ином поле таблицы базы данных, возможно использование тезауруса предметной области. Тезаурус представляет собой словарь, который включает в себя определения и термины конкретной области знаний (Браславский П.И. , 2003). В рамках этой предметной области содержится информация в базе данных. Тезаурус также содержит синонимы и антонимы, то есть семантические отношения между лексическими единицами. Помимо этого, тезаурус способен помочь в решении задачи по выявлению смысла не только с помощью определений, но и соотнеся слово с другими понятиями и группами понятий.

Подобный тезаурус должен быть сформирован заранее и подключен к естественно-языковому интерфейсу, для обеспечения возможности использования данных, хранящихся в нем, в процессе анализа и формирования семантической модели базы данных. В качестве примера подобного тезауруса могут быть использованы общие и свободно распространяемые решения, как например, WordNet (Miller G.A. , 1995) и EuroWordNet (Vossen P., 1997). WordNet содержит в себе состав и структуру лексической системы языка в целом, а не отдельной тематической области. WordNet представлен для следующих языков: английский, голландский, итальянский, испанский, немецкий, французский, чешский, эстонский. EuroWordNet описывает особенности лексических систем национальных языков и реализует в себе идею об объединении отдельных представлений в общую совокупность. Содержит в себе тезаурусы для европейских языков, таких как голландский, испанский, итальянский, немецкий, французский, чешский и эстонский (Азарова И.В., и др., 2003). Среди тезаурусов, поддерживающих русский язык можно отметить такие работы как RussNet (Azarova I. et al., 2002), YARN (Braslavski P., et al., 2014), Викисловарь (Krizhanovsky A. A., et al., 2013).

Также возможно применение собственного тезауруса. Собственный тезаурус может быть самостоятельно сформирован в автоматическом режиме,

например, на основе технической документации проекта или иных вспомогательных данных, служащих для описания предметной области и ключевых особенностей, связанных с данными, хранящимися в базе данных, для которой создается естественно-языковой пользовательский интерфейс.

Таблица 3.1 – Содержимое инфологической и даталогической модели базы данных

Термины инфологической модели БД	Термины даталогической модели БД
Сотрудник	employee
Отдел	department

В рассмотренном ранее примере, представленном на рисунке 3.2 и в таблице 3.1 представлена связь между сущностями *сотрудник* (*employee*) и *отдел* (*department*). Опираясь на информацию даталогической модели базы данных, сущности расположены в различных таблицах базы данных. При этом неизвестен тип связи между данными сущностями. Такую проблему возможно решить, путем предоставления пользователю возможности ручного задания вида связи непосредственно после этапа автоматического определения сущностей.

В ином случае, создается пользовательский тезаурус, который будет включать в себя типы связей между сущностями, хранящимися в базе данных. Такой тезаурус разрабатывается в организации, которая занимается разработкой, сопровождением и актуализацией базы данных. При реализации данного сценария, вид связи между сущностями *employee* и *department* представлен в виде следующего триплета:

< employee, работает в, department >

При этом, подключив общие тезаурусы, возможно дополнительно использовать эти данные для определения семантики сущностей, которые хранятся в указанных таблицах базы данных (Посевкин Р.В., 2018). Применяя данные тезаурусов, получим отношение, заданное таким триплетом как:

< сотрудник, работает в, отдел >

В результате, применив тезаурус к терминам даталогической модели, была успешно идентифицирована семантика типа связи между сущностями

инфологической модели. Данное отношение будет сохранено в инфологической модели базы данных:

$$P_{rel}(\text{Сотрудник}, \text{работает_в}, \text{Отдел})$$

3.3.4 Разработка метода определения семантики сущностей базы данных на основе паттернов

Для идентификации семантики сущностей, хранящихся в базе данных применяется набор методов, основанных на паттернах. Для идентификации семантики поля таблицы базы данных в автоматическом режиме осуществляется анализ содержимого данных, представленным в этом поле. Данный подход может применяться для идентификации данных, имеющих типичное представление и в силу своей общеупотребительности, широко используемых в различных базах данных, например:

- адрес электронной почты;
- номер телефона;
- автомобильный номер;
- серия и номер документов – паспорт, СНИЛС и т.п.;
- ФИО – сочетание фамилии, имени и отчества.

В результате проведения анализа содержимого поля, содержащего адрес электронной почты, возможно формализовать следующее правило. Данная запись представлена в виде строки, которая состоит из двух частей, разделенных символом «@»: $[prefix]@[postfix]$. В данном случае $[prefix]$ – текстовая строка, в то время как $[postfix]$ – доменное имя, имеющее многоуровневую структуру. В RFC 5322 (Resnick P.W., 2008) описан процесс проверки на наличие адреса электронной почты. Данный процесс может быть осуществлен посредством регулярного выражения, схема работы которого представлена на рисунке 3.8. Помимо этого, сейчас получают все большее распространение национальные домены первого уровня. В результате, $[prefix]$ и $[postfix]$ могут быть представлены не только на английском языке и регулярное выражение требует более расширенную версию в случае

необходимости поддержки символов национальных алфавитов (Посевкин Р.В., 2018).

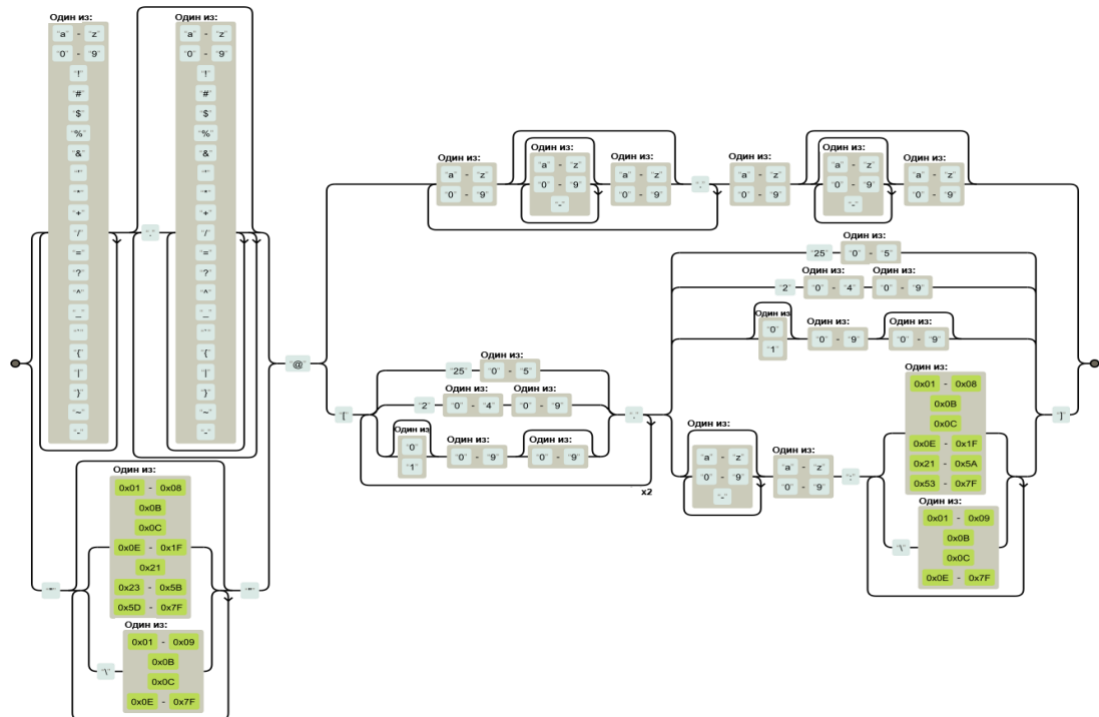


Рисунок 3.8 – Схема формирования регулярного выражения для адреса электронной почты

В процессе анализа содержимого поля базы данных в качестве дополнительного источника данных могут также использоваться текстовые корпуса и тезаурусы. Например, с помощью паттерна в поле таблицы базы данных была найдена комбинация Фамилия-Имя-Отчество. Применяв тезаурус, содержащий характерные признаки и наборы данных возможно подтвердить гипотезу по выявлению указанной смысловой конструкции.

Так, для русского языка сама процедура проверки предположения наличия ФИО в поле таблицы базы данных может быть инициирована в случае наличия в поле трех подряд идущих слов, начинающихся с заглавной буквы. Данный паттерн, описанный с помощью контекстно-свободных грамматик, имеет следующий вид:

$$\begin{array}{l} S \rightarrow A' A' A \\ A \rightarrow BC \mid B \\ B \rightarrow 'A' \mid 'B' / \dots / 'Y' \\ C \rightarrow CD \mid D \mid C' - 'C \\ D \rightarrow 'a' / 'b' / \dots \mid 'y' \end{array}$$

Также при использовании паттернов в процессе анализа семантики сущностей базы данных могут быть использованы источники дополнительных данных для совершенствования качества проводимого анализа. Таким дополнительным источником данных может выступить указание локали содержимого базы данных. Данная информация позволит сократить время обработки данных ввиду того, что в таком случае будут использоваться только те правила, тезаурусы и текстовые корпуса, которые релевантны указанному языку.

3.3.5 Формирование таблицы проекций, обеспечивающей связь между терминами

В семантической модели базы данных применяется таблица проекций, которая обеспечивает связь между терминами естественного языка, инфологической и даталогической моделей базы данных. Изначально ряд терминов естественного языка может быть включен в семантическую модель базы данных вручную. При этом для каждого включенного термина естественного языка с помощью тезаурусов определяются синонимы. В результате в семантическую модель включается термин, естественного языка, его синонимы, а также отношения синонимичности терминов естественного языка.

Например, в семантическую модель добавлен термин естественного языка:

«сотрудник»

С помощью тезауруса определен набор синонимов, которые также вносятся в семантическую модель:

«работник», «специалист»

При этом между всеми синонимами в семантической модели БД определяется соответствующие отношения:

$P_{syn}(\text{сотрудник}, \text{специалист})$

$P_{syn}(\text{сотрудник}, \text{работник})$

$P_{syn}(\text{работник}, \text{специалист})$

Далее в процессе работы естественно-языкового интерфейса анализируются данные, получаемые из логов работы интерфейса. В результате, определяются термины естественного языка, которые применяют пользователи при формировании запроса, но которые еще не были включены в семантическую модель базы данных. Данные термины естественного языка включаются в семантическую модель. Далее весь описанный выше процесс повторяется для вновь добавленных терминов.

В результате термины естественного языка, инфологической и даталогической моделей базы данных включаются в таблицу проекций. Также семантическая модель включает все выделенные ранее связи между терминами. Пример структуры семантической модели базы данных, включающий в себя таблицу проекций и отношения между терминами представлен на рисунке 3.9.

Таблица проекций		
Термины естественного языка	Термины инфологической модели БД	Термины даталогической модели БД
сотрудник	Сотрудник	employee
работник		
специалист		
департамент	Отдел	department
отдел		
мужчина	Сотрудник[:пол_мужской]	employee.sex = 'male'

Отношения между терминами

$P_{syn}(\text{сотрудник}, \text{специалист})$
 $P_{syn}(\text{сотрудник}, \text{работник})$
 $P_{syn}(\text{работник}, \text{специалист})$
 $P_{syn}(\text{департамент}, \text{отдел})$

$P_{link}(\text{employee}, \text{department}, \text{id_department}, \text{id})$

$P_{rel}(\text{Сотрудник}, \text{работает_в}, \text{Отдел})$

Рисунок 3.9 – Структура семантической модели базы данных

Выводы

На основе проведенного анализа исследований в смежных областях обосновывается необходимость использования семантической модели базы данных в естественно-языковом пользовательском интерфейсе. Применение семантической модели базы данных позволяет обеспечить формирование запроса к базе данных с использованием терминов предметной области.

Приведено описание процесса разработки семантической модели базы данных, сформулированы требования к семантической модели. В рамках исследования разработаны метод определения внутренних связей между сущностями базы данных; метод определения семантики типа связей между сущностями базы данных с использованием тезауруса; метод анализа семантики сущностей базы данных на основе паттернов. Также приводится описание разработанной программной системы автоматизированного формирования семантической модели базы данных. В отличие от известных решений, данные, необходимые для построения запроса к базе данных на основе естественно-языкового запроса формируются в автоматизированном режиме, что сокращает трудозатраты, необходимые для внедрения естественно-языкового пользовательского интерфейса.

Представленные в главе результаты опубликованы в (Посевкин Р.В., 2018) и (Посевкин Р.В., 2016).

Глава 4. Разработка алгоритма построения запроса к базам данных на основе анализа естественно-языкового запроса

4.1 Использование семантической модели и К-представления для формирования запроса к базе данных

В предыдущей главе разработана семантическая модель базы данных. Семантическая модель базы данных является важной компонентой естественно-языкового интерфейса к базе данных, отличается наличием таблицы проекций и обеспечивает формирование запроса к базе данных с использованием терминов предметной области.

В рамках этапа обработки запроса на естественном языке формируется семантическое представление запроса. При этом семантическое представление естественно-языкового запроса пользователя строится на основе данных семантической модели базы данных. Взаимосвязи терминов естественно-языкового запроса, семантической модели и внутреннего представления базы данных представлены на рисунке 4.1.



Рисунок 4.1 – Взаимосвязи терминов естественно-языкового запроса, семантической модели и внутреннего представления базы данных

Семантическое представление естественно-языкового запроса пользователя представляет собой выражение стандартного концептуального языка. В результате, формируется промежуточное К-представление запроса (Правилов А.А., и др., 2010). К-представление служит для связи между терминами естественно-языкового запроса и сущностями даталогической

модели базы данных. Следующим этапом промежуточное К-представление запроса преобразуется в SQL-запрос к базе данных. При этом преобразование К-выражения в SQL-запрос к базе данных осуществляется методом подстановки полученного выражения в заранее predetermined шаблон.

В результате, процесс выглядит следующим образом: *ЕЯ-запрос пользователя* → *К-представление запроса* → *SQL-запрос к БД* и в общем виде представлен на рисунке 4.2 (Посевкин Р.В., 2018).

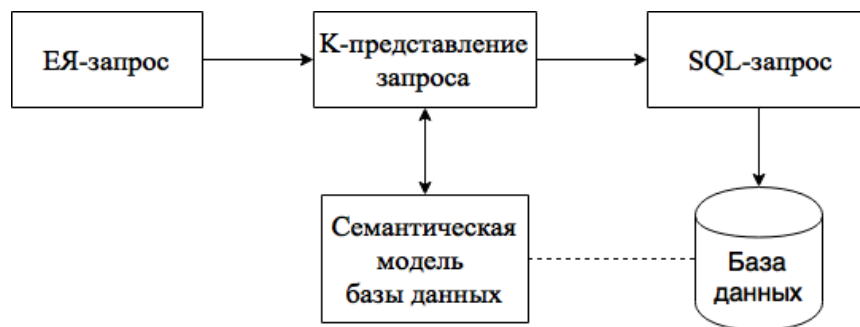


Рисунок 4.2 – Процесс обработки естественно-языкового запроса с использованием семантической модели базы данных

Использование К-представления позволяет сформировать описание запроса пользователя на основе его естественно-языкового представления, не привязываясь при этом к конкретным операторам языка запроса. В результате, на основе К-представления может быть сформирован не только SQL-запрос к реляционной СУБД, но и также запросы к другим источникам информации. Например, запрос к документно-ориентированной базе данных, чья структура запроса и используемый синтаксис отличаются от SQL-запроса.

4.2 Разработка алгоритма формирования SQL-запроса на основе естественно-языкового запроса

Промежуточное К-представление пользовательского запроса можно представить в виде объекта, с последующими свойствами этого объекта, в виде (*ключ, значение*). Рассмотрим пример – для естественно-языкового запроса «Список всех мужчин, работающих в компании» К-представление будет иметь вид:

все сотрудники(Пол, Мужской)*

Формальный запрос имеет следующий вид:

Запрашиваемый объект(запрос1, все сотрудники(Элемент, S1),
Описание1 (пол * (Элемент, S1) : y1, пол (y1, Мужской)))*

В данном случае, «сотрудники» – это предметная область, которая задается используемой базой данных, а именно – таблицей employee. Тогда как «пол» – это поле в таблице сотрудников. При этом соответствие поля в базе данных наименованию на естественном языке задано в семантической модели базы данных. Следующим этапом К-представление сопоставляется с набором шаблонов SQL-запросов. В результате, подобный естественно-языковой запрос будет преобразован в SQL-запрос вида:

*select * from employee where employee.sex='Male'*

Предложенный подход предполагает достаточно гибкую привязку к базе данных. Для связи естественно-языковых терминов и соответствующих полей таблиц (сущностей даталогической модели базы данных) используются таблицы проекций. Связь может быть прямая. Например, поле «Пол» соответствует полю «sex». Также связь может быть оформлена в виде вложенного запроса к базе данных.

Преимуществом предлагаемого подхода к обработке естественно-языковых запросов с использованием семантической модели базы данных является возможность обработки и формирования более комплексных SQL-запросов, которые содержат в себе вложенный подзапрос. Разработанный подход устраняет недостатки решений, представленных ранее в сравнительном обзоре в таблице 1.1, которые не обладают всей полнотой используемых естественно-языковых конструкций. В частности, недостатки подхода, предложенного в работе (Евдокимова И.С., 2004), в котором предлагается использовать ограниченное подмножество естественного языка для взаимодействия пользователя с естественно-языковым интерфейсом к базе данных. Комплексные запросы, содержащие в себе вложенный подзапрос,

встречаются довольно часто, в особенности в процессе связывания данных, извлекаемых из разных таблиц базы данных.

Рассмотрим пример такого запроса, представленного в таблице 4.1.

Таблица 4.1 – Этапы преобразования естественно-языкового пользовательского запроса в SQL-запрос к базе данных

Запрос на естественном языке	Телефоны всех мужчин отдела логистики
К-представление запроса	Запрашиваемый объект ((запрос1, все сотрудники*(Элемент, S1), Описание1 (пол * (Элемент, S1) : y1, пол (y1, Мужской)) ^ Описание2 (работа в отделе * (Элемент, S1) : y2, ID отдела (y2, S2))), (запрос2, все отделы*(Элемент, S2), Описание3 (ID * (Элемент, S2) : y2, название отдела (y2, Логистика))))
SQL-запрос	select phone from employee where employee.sex='male' and employee.id_department in (select id from department where department.title='logistics')

С учетом структуры хранения данных в базе данных, представленной на рисунке 4.1, пользовательский запрос можно декомпозировать на следующие этапы:

1. Выборка идентификаторов отделов, названия которых соответствует запрашиваемому.
2. Выборка сотрудников, пол которых мужской и которые работают в искомом отделе.
3. Получение номера телефона найденных сотрудников.

В данном примере для обработки запроса требуется извлечение данных из таблиц employee и department. Также в семантической модели базы данных должно быть указано, как связаны между собой эти две таблицы. А именно, то, что идентификатор employee.id_department соответствует идентификатору department.id. Помимо этого, семантическая модель базы данных включает в себя данные о синонимах и расположении соответствующих им данных в таблицах базы данных. Такие сведения содержатся в таблицах проекций. В результате, будет получено соответствие термина «телефон» сущности

employee.phone базы данных. При этом естественно-языковой термин «отдел» или «департамент» отображают необходимость поиска данных в таблице department базы данных. Учитывая тот факт, что далее в запросе следует термин «логистика» делается заключение, что необходимо осуществлять поиск по названию отдела. В соответствии с информацией из семантической модели базы данных название отдела соответствует полю department.title в базе данных. В данном случае также необходимо указание в семантической модели базы данных информации о соответствии понятия «логистика» значению «logistics».

Помимо этого, в семантическую модель базы данных может быть включена информация о сущностях, которые непосредственно не хранятся в базе данных, но могут быть сформированы на основе имеющихся данных. В данном примере это сущность «мужчина». Для текущей структуры базы данных – это сотрудники, имеющие пол «мужской»: employee.sex='male'. Подобные данные могут быть описаны в базе данных в виде общих табличных выражений (common table expressions, CTE) (Shalygina G., Novikov B., 2017). Также наличие общих табличных выражений позволяет формировать более сложные запросы к базе данных посредством естественно-языкового интерфейса.

Таким образом алгоритм формирования SQL-запроса на основе естественно-языкового вопроса пользователя представлен на рисунке 4.3 и может быть формализован в следующем виде:

1. Выделение именованных сущностей из естественно-языкового пользовательского запроса.
2. Выделение отношений между сущностями с помощью шаблонов моделей предложений.
3. Получение данных из семантической модели базы данных на основе терминов естественно-языкового запроса и таблицы проекций.
4. Формирование К-представления пользовательского запроса.

5. Подстановка данных даталогической модели в К-представление пользовательского запроса.
6. Формирование запроса на формальном языке к источнику данных на основе шаблона.

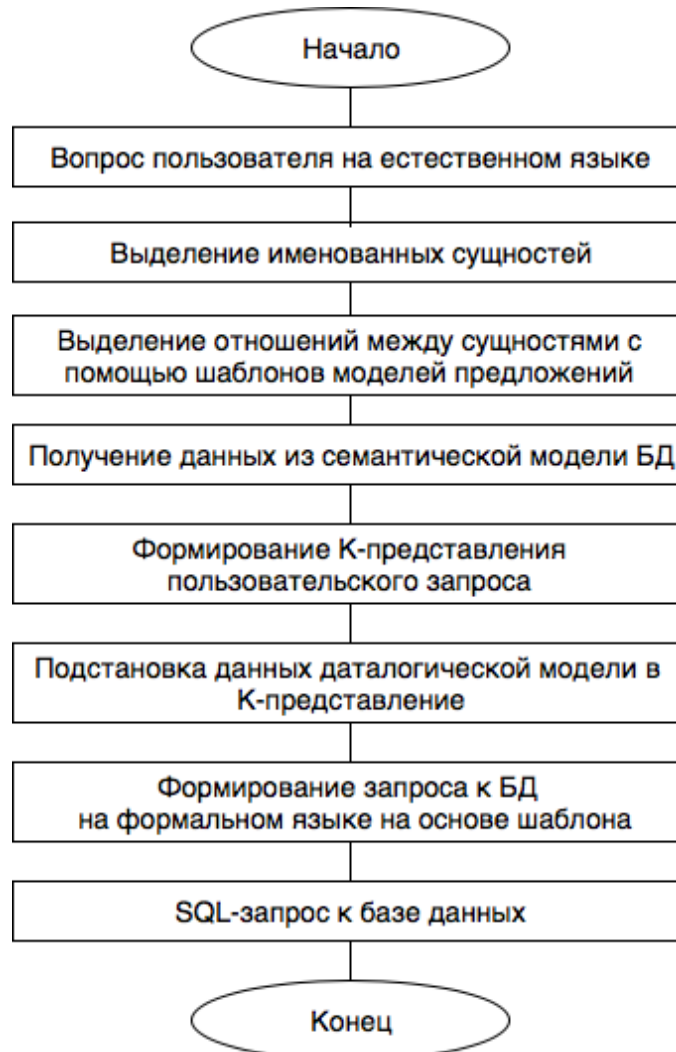


Рисунок 4.3 – Алгоритм формирования SQL-запроса на основе естественно-языкового вопроса пользователя

Рассмотрим поэтапно процесс преобразования запроса пользователя на естественном языке в SQL-запрос к базе данных в соответствии с предложенным алгоритмом. Вопрос пользователя на естественном языке имеет следующий вид:

Сколько неоплачиваемых отпусков у Иванова?

Из данного естественно-языкового вопроса пользователя по термину «Иванова» выделена именованная сущность *[NamedEntity: Фамилия]*. Также

на основе анализа термина естественного языка «Сколько», находящегося в начале предложения, выявлено P_{count} – отношение количества.

Далее производится получение данных из семантической модели базы данных. Данные, содержащиеся в семантической модели и относящиеся к данному примеру представлены на рисунке 4.4.

Таблица проекций		
Термины естественного языка	Термины инфологической модели БД	Термины даталогической модели БД
отпуск	Отпуск_сотрудника	vacation
отсутствие		
неоплачиваемый отпуск	Отпуск_сотрудника[::без_оплаты]	vacation.type = 'non-payable'
отпуск за свой счет		
за свой счет		
[NamedEntity: Фамилия]	Фамилия_сотрудника	employee.surname
сколько	P_{count}	COUNT(*)

Отношения между терминами

$P_{syn}(отпуск, отсутствие)$
 $P_{syn}(неоплачиваемый\ отпуск, отпуск\ за\ свой\ счет)$
 $P_{syn}(отпуск\ за\ свой\ счет, за\ свой\ счет)$
 $P_{syn}(за\ свой\ счет, неоплачиваемый\ отпуск)$

$P_{link}(vacation, employee, id_employee, id)$

Рисунок 4.4 – Данные, содержащиеся в семантической модели базы данных

На основе полученных из семантической модели данных, а также выделенных ранее сущностей и отношений формируется К-представление запроса пользователя в терминах инфологической модели:

$P_{count} (Отпуск_сотрудника[::без_оплаты], Фамилия_сотрудника[Иванов])$

Данное К-представление представляет собой связь между терминами естественного языка из вопроса пользователя с терминами инфологической модели в формализованном виде. Таким образом, К-представление

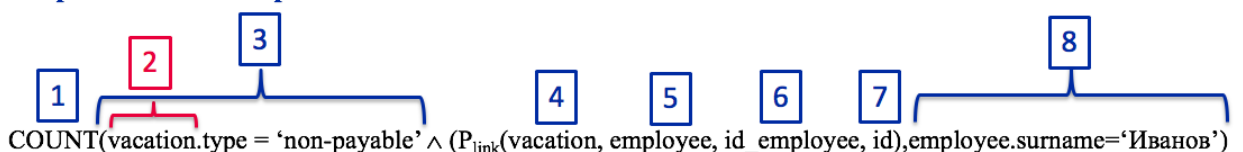
показывает, информацию о каких сущностях предметной области необходимо извлечь из базы данных. При этом, подробности реализации конкретной базы данных на этом этапе не учитываются, что позволяет обеспечить портируемость на другой формальный язык запросов. Например, для рассматриваемого примера отношению P_{count} соответствует термин даталогической модели БД $COUNT(*)$, ввиду использования языка SQL для построения запроса к реляционной СУБД. Если же необходимо использовать данный интерфейс для работы с документо-ориентированной базой данных, например, MongoDB, то К-представление в терминах инфологической модели БД будет иметь аналогичный вид. При этом отношению P_{count} будет соответствовать термин даталогической модели БД $\$count$. Таким образом, наличие К-представления позволяет осуществить портируемость естественно-языкового интерфейса на другой формальный язык запроса.

На этапе подстановки данных даталогической модели в К-представление формируется К-представление в терминах даталогической модели, имеющее следующий вид:

COUNT(vacation.type = 'non-payable' \wedge ($P_{link}(vacation, employee, id_employee, id)$, employee.surname='Иванов')

Далее осуществляется подстановка К-представления в соответствующий шаблон для формирования запроса на формальном языке. Данная процедура продемонстрирована на рисунке 4.5

К-представление запроса:


COUNT(vacation.type = 'non-payable' \wedge ($P_{link}(vacation, employee, id_employee, id)$, employee.surname='Иванов')

Шаблон запроса:

SELECT [1] (*) FROM [2] WHERE [3] AND [4] . [6] IN (SELECT [7] FROM [5] WHERE [8])

Рисунок 4.5 – Формирования запроса к базе данных на формальном языке на основе шаблона

Таким образом, сформирован SQL-запрос к базе данных:

SELECT COUNT() FROM vacation WHERE vacation.type='non-payable' AND vacation.id_employee IN (SELECT id FROM employee WHERE employee.surname='Иванов')*

В результате, на примере продемонстрирован процесс преобразования вопроса пользователя на естественном языке в SQL-запрос к базе данных.

4.3 Разработка функциональной модели естественно-языкового интерфейса

Разработана программная реализация естественно-языкового пользовательского интерфейса, включающая в себя ранее описанные алгоритмы и методы. Функциональная модель естественно-языкового интерфейса представлена на рисунке 4.6.

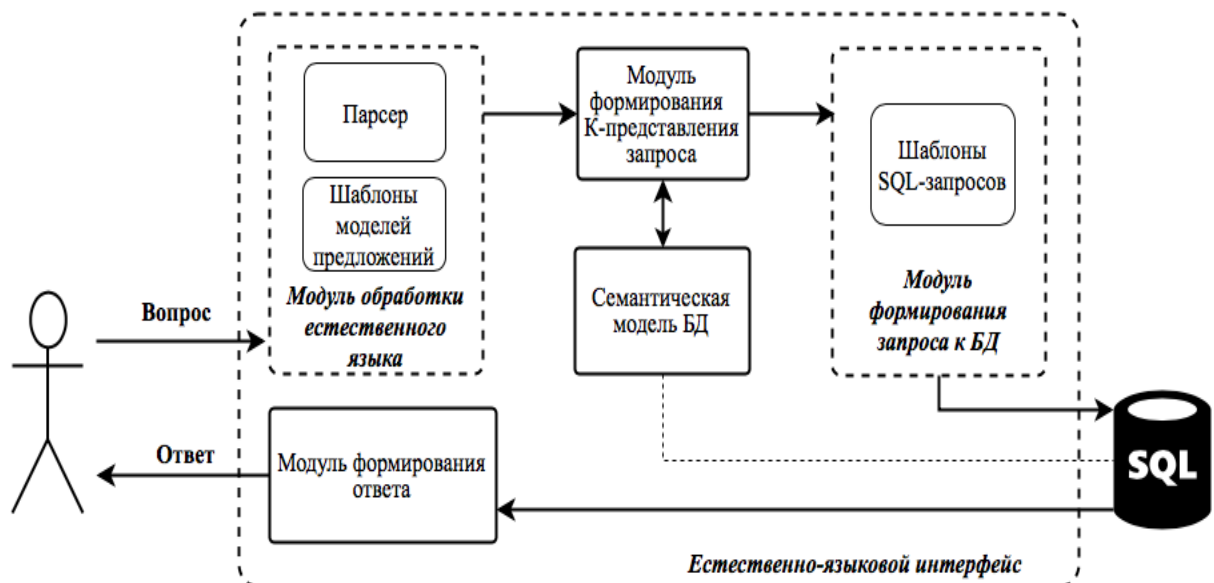


Рисунок 4.6 – Функциональная модель естественно-языкового интерфейса

Программная реализация естественно-языкового интерфейса к базам данных представляет собой клиент-серверное приложение. Клиентская часть реализована на HTML, CSS, JavaScript, с использованием интерфейсной библиотеки React (Fedosejev A., 2015). Серверная часть приложения реализована на Node.js (Tilkov S., Vinoski S., 2010). Для формирования структуры применяется каркас web-приложений (framework) Express.js (Mardan A., 2014). Приложение построено в соответствии с принципами,

заложенными в микросервисной архитектуре (Dragoni N., et al. , 2017). В результате, программная система состоит из набора небольших и легких модулей, что приводит к уменьшению связанности. Это позволяет гораздо проще добавлять новые модули, масштабировать систему в целом, а также изменять уже реализованный функционал приложения.

При работе с системой, пользователь задает вопрос путем текстового ввода в форму графического интерфейса. Далее пользовательский вопрос поступает для обработки на вход модуля обработки естественного языка. В свою очередь, модуль включает в себя парсер и шаблоны моделей предложений, с помощью которых производится обработка естественно-языкового пользовательского вопроса. После этапа обработки естественного языка данные поступают в модуль формирования К-представления запроса. На данном этапе из семантической модели базы данных выбираются необходимые данные на основе терминов естественно-языкового запроса и таблицы проекций. В данном случае ищется соответствие терминов естественно-языкового запроса сущностям даталогической модели базы данных. В результате формируется информация к каким полям и каким таблицам базы данных необходимо обратиться, чтобы получить данные для формирования ответа релевантного вопросу пользователя. Таким образом, на выход модуля поступает промежуточное К-представление пользовательского запроса. Эти данные поступают для обработки в модуль формирования запроса к базе данных. Модуль содержит в себе шаблоны запросов к базе данных на формальном языке. В случае реляционной системы управления базами данных, это шаблоны SQL-запросов. Здесь происходит выбор шаблона запроса на формальном языке на основе К-представления запроса и выделенных конструкций естественно-языкового запроса на основе шаблонов моделей предложений. После успешного подбора шаблона, происходит процесс формирования SQL-запроса к базе данных путем проецирования К-представления запроса на шаблон запроса. Сформированный SQL-запрос отправляется к базе данных. Если произошла ситуация, при которой

сформирован некорректный SQL-запрос, то в зависимости от настроек естественно-языкового интерфейса пользователю будет либо предложено переформулировать запрос на естественном языке, либо, будет выведен сформированный SQL-запрос, который пользователь может исправить вручную. На последнем этапе, результат обработки запроса базой данных направляется в модуль формирования ответа (Jurafsky D., 2000), где данные преобразовываются в удобочитаемый ответ и отображаются пользователю.

За счет отделения этапа формирования К-представления этапа формирования запроса с на основе шаблонов, появилась возможность формирования как SQL-запроса, так и запросов на других формальных языках, например, к документо-ориентированной БД. Таким образом, корректное формирование К-представления запроса, а также модульность компонент интерфейса обеспечивают возможность портирования интерфейса на другие естественные языки и формальные языки запроса.

4.4 Обеспечение портируемости естественно-языкового интерфейса

Ранее в разделе 2.1. в качестве критерия качества естественно-языкового пользовательского интерфейса к базам данных упоминалось свойство портируемости. Естественно-языковой пользовательский интерфейс может быть портирован следующим образом:

- на другой формальный язык запроса;
- на другую предметную область;
- на другой язык.

Ввиду того, что программная реализация естественно-языкового интерфейса выполнена с учетом принципов микросервисной архитектуры осуществить тот, или иной вид портируемости возможно путем внесения минимальных изменений в структуру естественно-языкового интерфейса.

При портировании естественно-языкового интерфейса **на другой формальный язык запроса** произойдет изменение самого источника данных.

Рассмотрим пример с портированием интерфейса для взаимодействия с документо-ориентированной системой управления базами данных. Реляционные СУБД обрабатывают SQL-запросы в качестве входной информации, в то время как запросы документо-ориентированных СУБД имеют иной формат. В данном случае потребуются изменения только в модуле формирования запроса к базе данных. Таким образом, подключив шаблоны запросов для документо-ориентированной СУБД вместо шаблонов SQL-запросов, естественно-языковой интерфейс к базе данных будет способен корректно выполнять свою работу. Если производится переход от реляционной СУБД к документо-ориентированной, содержащей аналогичные сведения, то потребуется также внести изменения в семантическую модель базы данных. Изменения связаны с различиями в структуре хранения данных внутри базы данных реляционного и документо-ориентированного типов. Процесс изменения данных в семантической модели БД также может быть автоматизирован. В результате, функциональная модель естественно-языкового интерфейса к реляционной базе данных, представленная ранее на рисунке 4.6, при портировании естественно-языкового интерфейса для работы с документо-ориентированной базой данных будет иметь вид, представленный на рисунке 4.7.

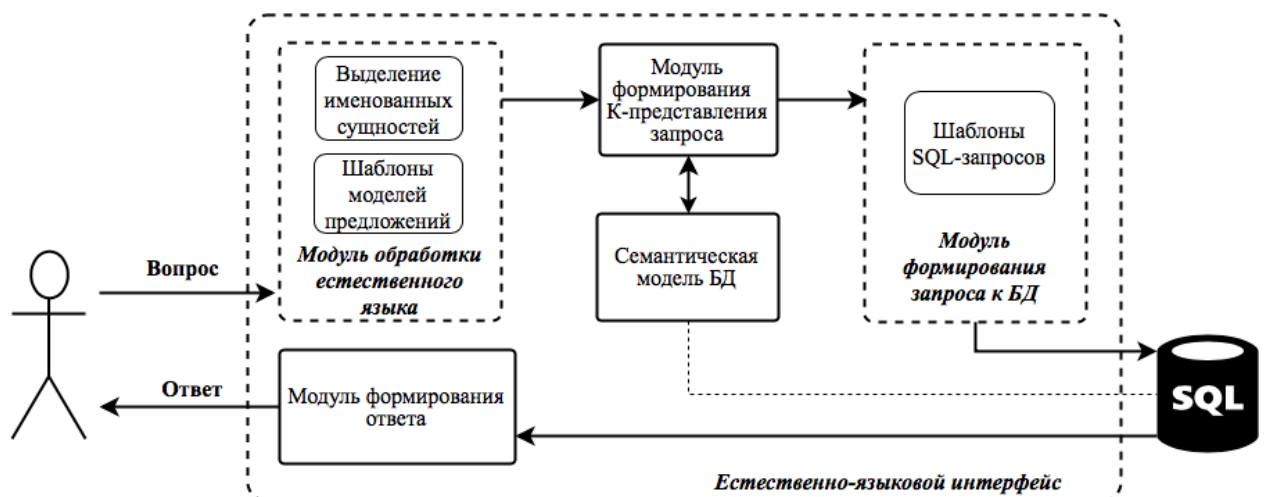


Рисунок 4.7 – Функциональная модель естественно-языкового интерфейса к документо-ориентированной базе данных

Осуществляя портирование разработанного естественно-языкового пользовательского интерфейса к базам данных **на другую предметную область**, существенных изменений в структуру интерфейса вносить не требуется. Единственным важным условием является необходимость формирования новой семантической модели для новой базы данных. Данный процесс осуществляется аналогичным образом, в автоматизированном режиме с помощью разработанной программной системы формирования семантической модели базы данных. Также с целью улучшения качества анализа новой базы данных возможно подключение новых тезаурусов, релевантных предметной области новой базы данных.

Для реализации процесса портирования естественно-языкового интерфейса к базам данных **на другой язык** потребуется внесение изменений в состав модуля обработки естественного языка. В первую очередь необходима замена шаблонов моделей предложений. Ввиду того, что структура предложений имеет специфику, присущую конкретному языку, шаблоны моделей предложений будут также различны. В случае, когда все возможные варианты пользовательских запросов могут быть охвачены исключительно шаблонами моделей предложений, компонента с парсером может быть исключена. В противном случае, если уже существующий парсер не поддерживает работы с новым естественным языком, необходимо его также заменить на парсер, поддерживающий новый язык. При использовании тезаурусов при формировании семантической модели базы данных, их необходимо отключить или обновить релевантными новому естественному языку. В случае, когда модуль формирования ответа не использует естественный язык для вывода информации пользователю, изменений в работу модуля вносить не требуется.

Выводы

В результате диссертационного исследования разработан алгоритм построения запроса к базам данных на основе анализа естественно-языкового

запроса с использованием семантической модели базы данных. При формировании запроса к базе данных строится К-представление запроса, а также используется информация из семантической модели базы данных.

Также разработана программная реализация естественно-языкового пользовательского интерфейса к базам данных. Данная реализация позволяет обеспечить портируемость интерфейса на другую предметную область. В отличие от известных решений, разработанный естественно-языковой интерфейс может быть портирован на другие естественные языки и формальные языки запроса за счет модульности программной системы и построения К-представления запроса. К-представление использует структуру данных, которая позволяет описывать связи между терминами естественного языка и сущностями даталогической модели базы данных без привязки к конструкциям и операторам конкретного формального языка запросов к базе данных.

Представленные в главе результаты опубликованы в (Посевкин Р.В., 2018).

Глава 5. Экспериментальное исследование оценки качества разработанного естественно-языкового пользовательского интерфейса к базе данных

5.1 Оценка полноты, точности и F-меры естественно-языкового интерфейса

5.1.1. Постановка экспериментального исследования

Проведено экспериментальное исследование по оценке полноты и точности разработанного естественно-языкового интерфейса к базе данных. В исследовании участвовало 82 человека, которым было предложено заполнить анкету. По результатам заполнения анкеты добровольцы были разделены на 2 группы в зависимости от наличия знаний в области построения SQL-запросов к базе данных. 34 человека в анкете указали, что обладают подобными навыками, 48 человек – не обладают.

Далее каждый из добровольцев получил список из 115 вопросов, ответы на которые требовалось найти с помощью разработанного естественно-языкового пользовательского интерфейса, взаимодействующего с тестовой базой данных. Задание участника экспериментального исследования представлено в приложении 3.

Таблица 5.1 – Категоризация запросов к базам данных

Категория	Описание
K1	Запрос не содержит агрегатных функций и оператор JOIN
K2	Запрос не содержит агрегатных функций, при этом производится объединение нескольких таблиц с помощью JOIN
K3	Запрос содержит агрегатные функции
K4	Запрос содержит вложенный подзапрос или SELF JOIN

Распределение запросов в зависимости от категории сложности представлено в таблице 5.2.

Таблица 5.2 – Распределение запросов в зависимости от категории сложности

Категория	Количество запросов	%
K1	10	9
K2	32	28
K3	35	30
K4	38	33
Всего	115	100

Под агрегатными функциями понимаются операторы SQL, такие как COUNT, SUM, MIN, MAX, AVG, которые вычисляют набор значений и возвращают одиночное значение. SELF JOIN представляет собой операцию объединения таблицы с ней самой таким образом, будто это две разные таблицы.

Если в процессе взаимодействия пользователя с естественно-языковым интерфейсом был сформирован некорректный запрос к базе данных, то для пользователей, которые обладают навыками формирования SQL-запросов выводится сформированный запрос и пользователь его самостоятельно исправляет. Пользователи, которые не обладают навыками формирования SQL-запросов, предоставляется возможность переформулировать запрос на естественном языке.

Для экспериментов была выбрана реляционная СУБД MySQL, которая является бесплатной и свободно распространяемой. Данный факт способствует достижению воспроизводимости данного эксперимента для других исследователей. В свою очередь, MySQL работает в рамках контейнера, создаваемого с помощью программного обеспечения Docker, которое позволяет запускать и управлять приложениями в среде виртуализации. Это также позволяет достичь воспроизводимости экспериментов, за счет отсутствия необходимости в дополнительной

настройке СУБД и окружения, а следовательно, минимизировать возможные ошибки при неверном конфигурировании системы.

Ввиду отсутствия в свободном доступе готовых естественно-языковых интерфейсов, описанных в работах (Правилов А. А., 2011) и (Никонов В.О., 2007), данные естественно-языковые интерфейсы были самостоятельно воспроизведены на основе методов и алгоритмов, представленных в данных работах. Также данные системы были адаптированы для использования с тестовой базой данных. Далее весь объем вопросов, сформированных добровольцами в рамках описанного эксперимента, был протестирован на этих естественно-языковых интерфейсах.

Произведена оценка корректности SQL-запросов, сформированных на основе естественно-языковых запросов. Также производилась оценка релевантности полученных ответов. После этого проведена оценка и сравнительный анализ показателей полноты, точности и безошибочности работы естественно-языкового интерфейса.

5.1.2. Состав и организация тестового окружения

В рамках экспериментального исследования была сформирована тестовая база данных. База данных работает под управлением свободной реляционной СУБД MySQL и запущена с помощью программной системы для автоматизации развёртывания и управления приложениями в среде виртуализации Docker. Общая конфигурация представлена в таблице 5.3.

Таблица 5.3 – Конфигурация тестовой базы данных

Средство контейнеризации	Docker
Версия программного обеспечения	18.03.0-ce
СУБД	MySQL
Версия СУБД	5.7.21
Общее количество таблиц	30
Общее количество полей таблиц	116

Тестовая база данных содержит в себе сведения о внутренней структуре организации. Структура базы данных представлена на рисунке 5.1.

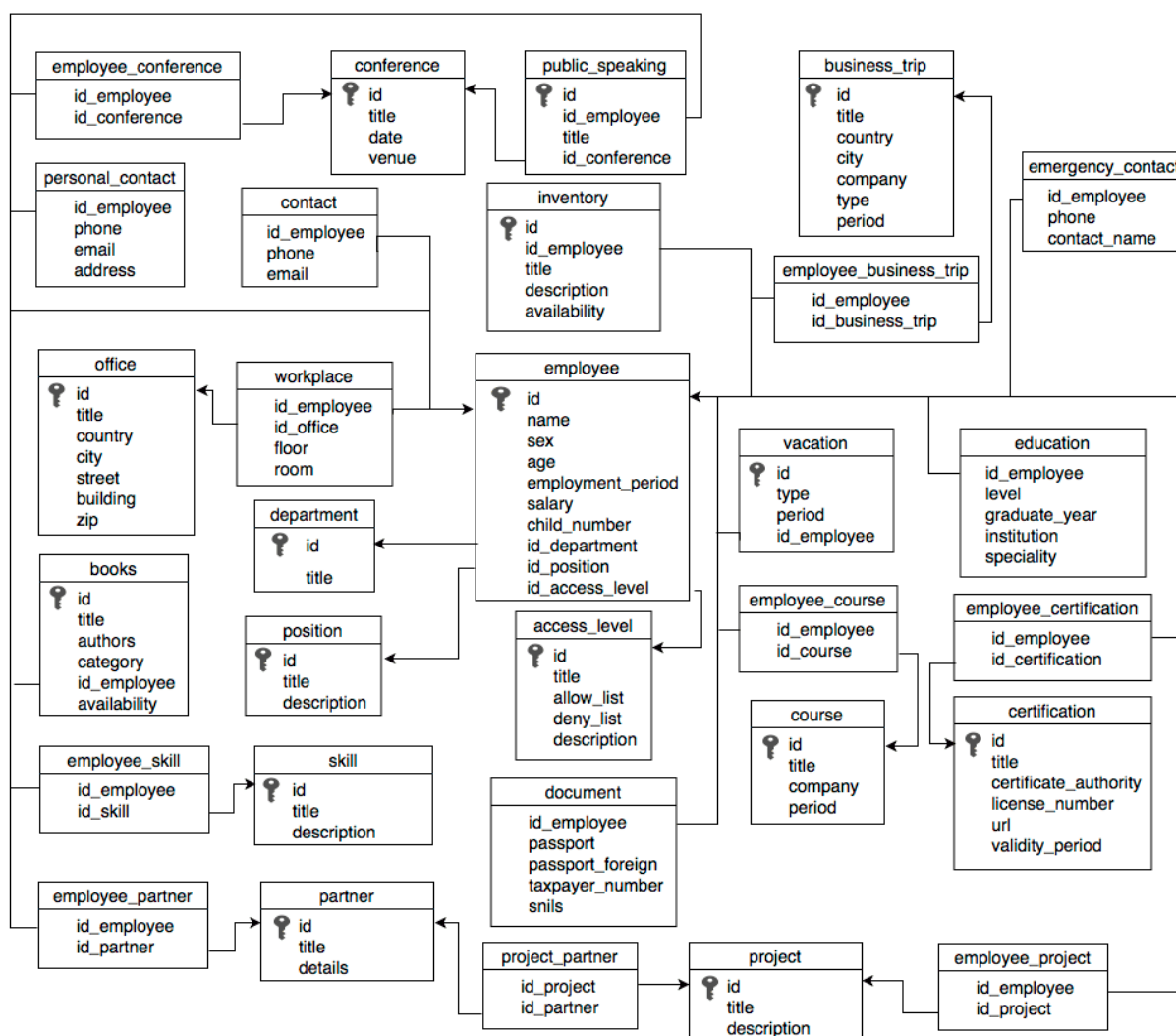


Рисунок 5.1 – Структура тестовой базы данных

Сведения о семантике внутреннего содержимого таблиц базы данных представлены в таблице 5.4.

Таблица 5.4 – Описание содержимого таблиц тестовой базы данных

№	Название таблицы	Семантика содержащихся данных
1	employee	Общие сведения о сотруднике
2	department	Перечень структурных отделов компании
3	contact	Контактная информация сотрудника
4	project	Перечень проектов компании
5	education	Сведения об образовании сотрудника
6	skill	Перечень навыков сотрудников
7	workplace	Сведения о рабочем месте конкретного сотрудника

8	office	Список офисов компании
9	document	Сведения о документах сотрудника
10	personal_contact	Персональные контактные данные сотрудника
11	position	Список должностей, существующих в организации
12	access_level	Список уровней доступа
13	certification	Сведения о документах, подтверждающих прохождение сертификации сотрудниками
14	business_trip	Список командировок сотрудников
15	vacation	Сведения об отпусках сотрудников
16	course	Список курсов повышения квалификации
17	emergency_contact	Экстренные контакты лиц, связанных с сотрудниками
18	inventory	Список оборудования компании
19	conference	Список конференций, в которых приняли участие сотрудники
20	public_speaking	Список публичных выступлений сотрудников
21	books	Список книг корпоративной библиотеки
22	partner	Список контрагентов, с которыми взаимодействует компания
23	employee_project	Связь сотрудника с проектом
24	project_partner	Связь проекта и контрагента
25	employee_partner	Связь сотрудника с контрагентом
26	employee_skill	Связь сотрудника и навыка
27	employee_certification	Связь сотрудника с сертификацией
28	employee_course	Связь сотрудника с курсом повышения квалификации

29	employee_business_trip	Связь сотрудника с командировкой
30	employee_conference	Связь сотрудника с конференцией

Состав полей таблиц, типы данных и описание содержимого приведены в таблицах 5.5 – 5.27.

Таблица 5.5 – Таблица employee: сведения о сотрудниках

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор сотрудника
name	varchar	ФИО сотрудника
sex	varchar	Пол сотрудника
age	int	Возраст сотрудника
employment_period	int	Трудовой стаж
salary	int	Размер заработной платы
child_number	int	Количество детей
id_department	int	Внешний ключ, отдел сотрудника
id_position	int	Внешний ключ, должность сотрудника
id_access_list	int	Внешний ключ, уровни доступа сотрудника

Таблица 5.6 – Таблица department: отделы компании

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор отдела
title	varchar	Название отдела

Таблица 5.7 – Таблица contact: контактная информация сотрудника

Название поля	Тип данных	Описание
---------------	------------	----------

id_employee	int	Внешний ключ, идентификатор сотрудника
phone	varchar	Номер телефона
email	varchar	Адрес электронной почты

Таблица 5.8 – Таблица project: проекты компании

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор проекта
title	varchar	Название проекта
description	varchar	Описание проекта

Таблица 5.9 – Таблица education: уровень образования сотрудника

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
level	varchar	Уровень образования
graduate_year	year	Год выпуска
institution	varchar	Наименование учебного заведения
speciality	varchar	Специальность

Таблица 5.10 – Таблица skill: перечень навыков сотрудников

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор навыка
title	varchar	Название навыка
description	varchar	Описание навыка

Таблица 5.11 – Таблица workplace: рабочее место сотрудника

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
id_office	varchar	Внешний ключ, идентификатор бизнес-центра
floor	int	Этаж
room	varchar	Комната

Таблица 5.12 – Таблица office: список офисов компании

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор офиса
title	varchar	Название офиса
country	varchar	Страна расположения
city	varchar	Город расположения
street	varchar	Улица
building	varchar	Строение
zip	varchar	Почтовый индекс

Таблица 5.13 – Таблица document: сведения о документах сотрудника

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
passport	int	Паспорт
passport_foreign	int	Заграничный паспорт
taxpayer_number	int	Номер налогоплательщика
snils	int	Страховой номер индивидуального лицевого счета

Таблица 5.14 – Таблица personal_contact: персональные контактные данные сотрудника

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
phone	varchar	Персональный номер телефона
email	varchar	Персональный адрес электронной почты
address	varchar	Адрес проживания

Таблица 5.15 – Таблица position: список должностей, существующих в организации

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор должности
title	varchar	Название должности
description	varchar	Описание должности

Таблица 5.16 – Таблица access_level: уровни доступа сотрудников

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор уровня доступа
title	varchar	Название уровня доступа
allow_list	enum	Список разрешенных действий
deny_list	enum	Список запрещенных действий
description	varchar	Описание уровня доступа

Таблица 5.17 – Таблица certification: сведения о документах, подтверждающих прохождение сертификации сотрудниками

Название поля	Тип данных	Описание
id	int	Первичный ключ,

		идентификатор сертификации
title	varchar	Название сертификации
certificate_authority	varchar	Название центра сертификации
license_number	varchar	Номер сертификата
url	varchar	URL-адрес проверки сертификата
validity_period	varchar	Срок действия

Таблица 5.18 – Таблица business_trip: список командировок сотрудников

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор командировки
title	varchar	Наименование командировки
country	varchar	Страна места командировки
city	varchar	Город места командировки
company	varchar	Компания
type	varchar	Вид командировки
period	varchar	Период командировки

Таблица 5.19 – Таблица vacation: сведения об отпусках сотрудников

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор отпуска
type	varchar	Категория отпуска
period	varchar	Период отпуска
id_employee	int	Внешний ключ, идентификатор сотрудника

Таблица 5.20 – Таблица course: список курсов повышения квалификации

Название поля	Тип данных	Описание
---------------	------------	----------

id	int	Первичный ключ, идентификатор курса
title	varchar	Название курса
company	varchar	Компания, проводящая курс
period	varchar	Период проведения курса

Таблица 5.21 – Таблица emergency_contact: экстренные контакты лиц, связанных с сотрудниками

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
phone	varchar	Номер телефона экстренного контактного лица
contact_name	varchar	Имя контактного лица

Таблица 5.22 – Таблица inventory: список оборудования компании

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор оборудования
id_employee	int	Внешний ключ, идентификатор сотрудника
title	varchar	Наименование оборудования
description	varchar	Спецификация оборудования
availability	boolean	Доступность для выдачи

Таблица 5.23 – Таблица conference: список конференций, в которых приняли участие сотрудники

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор конференции
title	varchar	Название конференции

date	varchar	Дата проведения
venue	varchar	Место проведения

Таблица 5.24 – Таблица public_speaking: список публичных выступлений сотрудников

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор публичного выступления
id_employee	int	Внешний ключ, идентификатор сотрудника
title	varchar	Название публичного выступления
id_conference	int	Внешний ключ, идентификатор конференции

Таблица 5.25 – Таблица books: список книг корпоративной библиотеки

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор книги
title	varchar	Название книги
authors	varchar	Авторы
category	varchar	Категория
id_employee	int	Внешний ключ, идентификатор сотрудника
availability	boolean	Доступность для выдачи

Таблица 5.26 – Таблица partner: список контрагентов, с которыми взаимодействует компания

Название поля	Тип данных	Описание
id	int	Первичный ключ, идентификатор

title	varchar	Наименование контрагента
details	varchar	Подробная информация

Таблица 5.27 – Таблица employee_project: связь сотрудника с проектом

Название поля	Тип данных	Описание
id_employee	int	Внешний ключ, идентификатор сотрудника
id_project	int	Внешний ключ, идентификатор проекта

Внутренняя организация таблиц базы данных под номерами 24 – 30 из таблицы 5.4 аналогична структуре таблицы «employee_project», представленной в таблице 5.27.

5.1.3. Анализ естественно-языковых запросов пользователя

В рамках экспериментального исследования пользователям требовалось получить ответы на вопросы. Для этого требовалось самостоятельно сформировать вопрос для поиска соответствующей информации с помощью естественно-языкового интерфейса. Если при взаимодействии пользователя с естественно-языковым интерфейсом к базе данных был сформирован некорректный запрос, то пользователю было необходимо переформулировать запрос. Пользователи, обладающие навыками формирования SQL-запросов, исправляли некорректно сформированный SQL-запрос. Результаты попыток исправления SQL-запроса представлены на рисунке 5.2.

Для пользователей, которые не обладают навыками формирования SQL-запросов, переформулировали непосредственно запрос на естественном языке. Результаты попыток исправления естественно-языкового запроса для пользователей, не знающих SQL отображены на рисунке 5.3.

Таким образом, 81% пользователей, столкнувшиеся с необходимостью переформулирования естественно-языкового запроса справились с этой задачей с двух попыток. При этом среди пользователей, знающих как

построить SQL-запрос, 92% справились с исправлением некорректного SQL-запрос с первой же попытки.

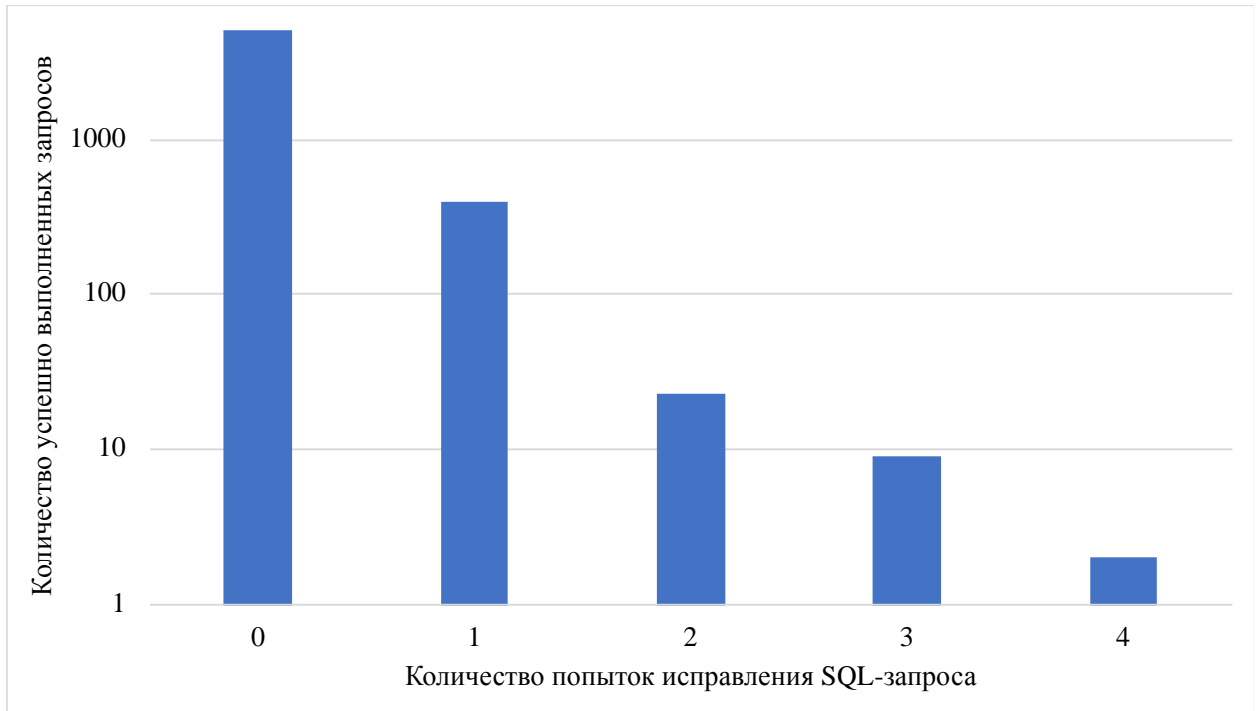


Рисунок 5.2 – Результаты попыток исправления SQL-запроса пользователями, знающими SQL

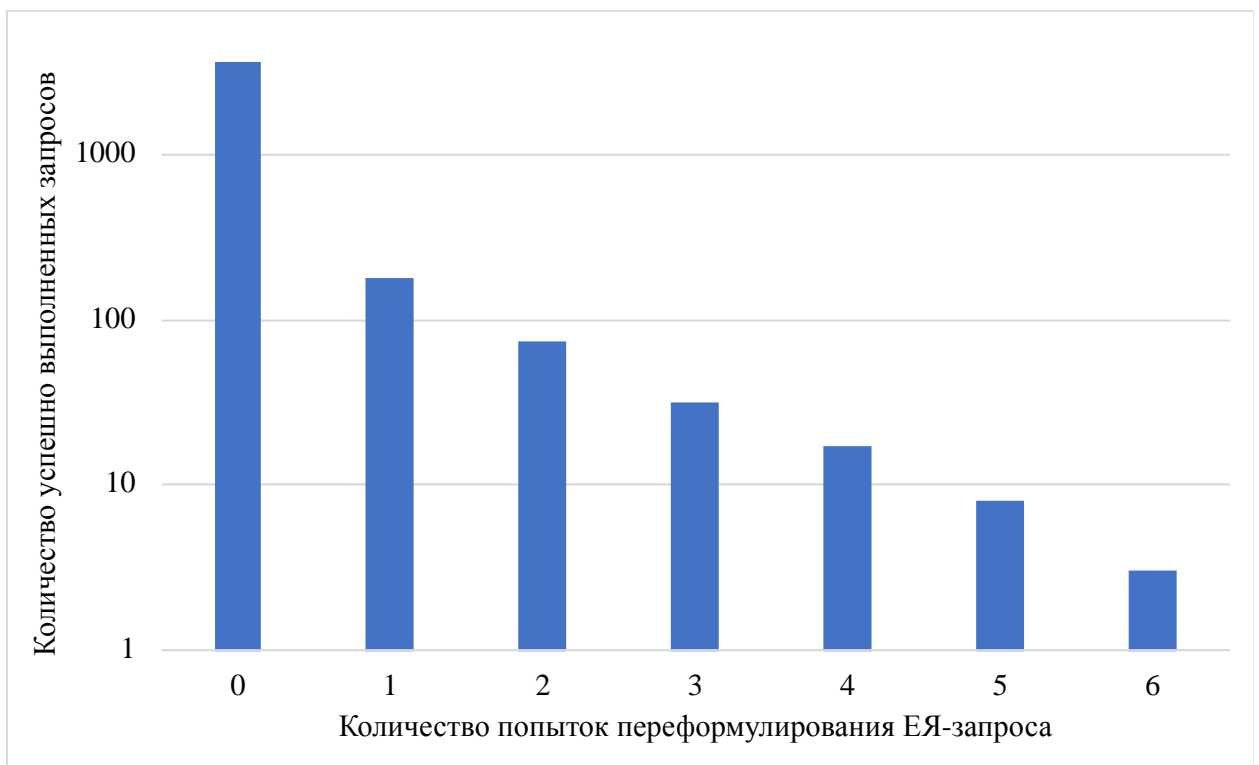


Рисунок 5.3 – Результаты попыток исправления естественно-языкового запроса пользователями, не знающими SQL

Примеры запросов, сформированных пользователями представлены в таблице 5.28.

Таблица 5.28 – Примеры естественно-языковых запросов пользователей к тестовой базе данных

Естественно-языковой запрос пользователя	Корректность обработки запроса	Релевантность полученного ответа
Сколько средний возраст сотрудников, у которых нет детей	Да	Да
Когда заканчивается сертификат 3522174	Да	Да
Сотрудники, работающие на одном этаже	Да	Нет
Сколько командировок было у Иванова	Да	Да
Какой принтер свободен	Да	Да
Какой программист получает меньше всех	Да	Нет
У кого из программистов минимальная зарплата	Да	Да
Кто выступал на SECR	Да	Да
Контакт близких Иванова на случай ЧС	Нет	Нет
Из какого отдела реже всех посещают конференции	Да	Да
Средний стаж работников с зарплатой больше ста тысяч	Да	Да
Сколько лет стажа у тех, кто получает больше 100 000	Да	Да
Какие сотрудники работают с одним партнером	Да	Нет
Среднее число навыков по должностям	Да	Да
Город, где было больше всего конференций	Да	Да

Рассмотрим несколько примеров некорректно обработанных запросов и случаев, когда мог быть получен нерелевантный ответ на вопрос. Для запроса «Сотрудники, работающие на одном этаже» был сформирован корректный SQL-запрос к базе данных. Однако, полученный ответ может быть нерелевантным ввиду многозначности фразы при интерпретации. Вероятно, пользователя интересовали сотрудники, работающие на одном этаже, при этом работающие в одном офисе. В этой ситуации данное ограничение также должно быть отражено в запросе к базе данных. Однако, если пользователю

необходимо получить статистику относительно всех существующих офисов, то полученный ответ окажется релевантным.

При обработке запроса *«Какой программист получает меньше всех»* неочевидно, необходимо осуществить поиск программиста с минимальной зарплатой среди всех сотрудников компании, или только среди программистов компании. В то же время, формулировка запроса *«У кого из программистов минимальная зарплата»* более строгая и устраняет данную неоднозначность при интерпретации запроса. В данном случае становится понятно, что выборка ограничена только программистами, а не всеми сотрудниками.

При обработке запроса вида *«Средний стаж работников с зарплатой больше ста тысяч»* важную роль играет выделение числительных для формирования количественных ограничений при формировании запроса к базе данных. В результате, данный запрос так же успешно обработан, как и запрос с ограничением, записанным с помощью цифр: *«Сколько лет стажа у тех, кто получает больше 100 000»*.

Однако, подобное выделение числительных и приведение их в числовой вид способно помешать корректной интерпретации запроса *«Какие сотрудники работают с одним партнером»*. Исходя из информации, что в задании, которое выдавалось пользователю требовалось получить ответ на вопрос «Информация о сотрудниках с общим партнером», термин «одним» следовало интерпретировать в виде условия *«один и тот же партнер»*.

Проблемы при обработке запроса *«Контакт близких Иванова на случай чс»* возникли ввиду отсутствия в семантической модели баз данных информации об аббревиатуре. Данная проблема может быть решена путем более полного наполнения семантической модели базы данных по результатам анализа логов работы естественно-языкового интерфейса. В данном случае, понятие «чс» – «чрезвычайная ситуация» должно быть включено в семантическую модель.

Также проблемы при интерпретации естественно-языкового запроса могут возникнуть в случаях, когда в запросе пользователь использует неявные

параметры. Например, в запросе «Какие работники много участвуют в конференциях?» параметр «много» является неявным. Данный критерий должен быть преобразован в более формализованный вид, например $\{N_{\text{конференций}} > 5\}$. Для осуществления подобного преобразования необходимо включение в семантическую модель информации, на основании которой неявные параметры, представленные в языковом запросе в качественном виде будут преобразованы в количественные ограничения, используемые при формировании запроса к базе данных.

В результате анализа работы естественно-языкового интерфейса по обработке запросов пользователей сформулированы рекомендации по обнаружению и исправлению некорректных запросов, которые могут быть применены как при усовершенствовании разработанного интерфейса, так и в альтернативных решениях. При вводе запроса, пользователь может допустить опечатки. В таких случаях при возникновении ошибки при обработке введенного запроса возможно исправить ошибку в автоматическом режиме с помощью программного средства обнаружения опечаток, такого как Яндекс.Спеллер (Чернова И. А., 2004).

В случае, когда при выполнении SQL-запроса в ответ была получена ошибка, возможно проведение анализа сообщения об ошибке для выявления полей таблицы базы данных, спровоцировавших ошибку. На основе этой информации пользователю отображается сообщение с конкретизацией части запроса, требующей исправления.

Также перед непосредственно отображением пользователю информации, полученной из базы данных, может быть произведен анализ количества полученных в ответе записей. Маловероятно, что пользователь целенаправленно формировал запрос, чтобы получить в ответ декартово произведение. В данном случае можно показать предупреждение о том, что количество записей очень большое и предложить пользователю переформулировать запрос. Альтернативным вариантом может быть

установка в конфигурации естественно-языкового интерфейса предельного количества записей, которое может получить пользователь в ответе.

5.1.4. Результаты экспериментального исследования

По итогам обработки естественно-языковых вопросов было проведено тестирование алгоритма формирования SQL-запроса к базе данных и релевантности полученного ответа.

Таблица 5.29 – Результаты экспериментального исследования по оценке полноты, точности и комбинированной F-меры естественно-языковых пользовательских интерфейсов

ЕЯ-интерфейс	Категория	# запросов	$ D_{retr} $	$ D_{rel} \cap D_{retr} $	Pr	Re	F-мера
Разработанный ЕЯ-интерфейс	K1	10	10	9	90	90	90
	K2	32	31	26	83.87	81.25	82.54
	K3	35	33	31	93.94	88.57	91.18
	K4	38	33	31	93.94	81.58	87.32
	Всего	115	107	97	90.65	84.35	87.39
(Никонов В.О., 2007)	K1	10	9	7	77.78	70	73.68
	K2	32	26	22	84.62	68.75	75.86
	K3	35	25	22	88	62.86	73.33
	K4	38	27	24	88.89	63.16	73.85
	Всего	115	87	75	86.21	65.22	74.26
(Правиков А. А., 2011)	K1	10	7	6	85.71	60	70.59
	K2	32	23	17	73.91	53.13	61.82
	K3	35	23	19	82.61	54.29	65.52
	K4	38	27	21	77.78	55.26	64.62
	Всего	115	80	63	78.75	54.78	64.62

В рамках эксперимента пользователями было сформировано 82 набора по $|D_{rel}| = 115$ вопросов. Была произведена оценка количества успешно

обработанных запросов $|D_{retr}|$. Под успешно обработанным запросом подразумевается синтаксически корректный SQL-запрос, сформированный на основе естественно-языкового запроса пользователя, успешно обработанный СУБД с первой же попытки. Также производился анализ релевантности полученных ответов методом экспертной оценки – $|D_{rel} \cap D_{retr}|$. Результаты экспериментального исследования представлены в таблице 5.29.

На основе полученных данных произведена оценка точности извлечения информации из базы данных $Pr = |D_{rel} \cap D_{retr}| / |D_{retr}|$. Таким образом, точность определяется отношением количества релевантных ответов к количеству успешно обработанных запросов. В результате, чем выше значение точности, тем более корректно естественно-языковой интерфейс способен преобразовать запрос на естественном языке в SQL-запрос к базе данных. Результаты оценки точности работы естественно-языковых интерфейсов в зависимости от категории сложности запроса представлены на рисунке 5.4.

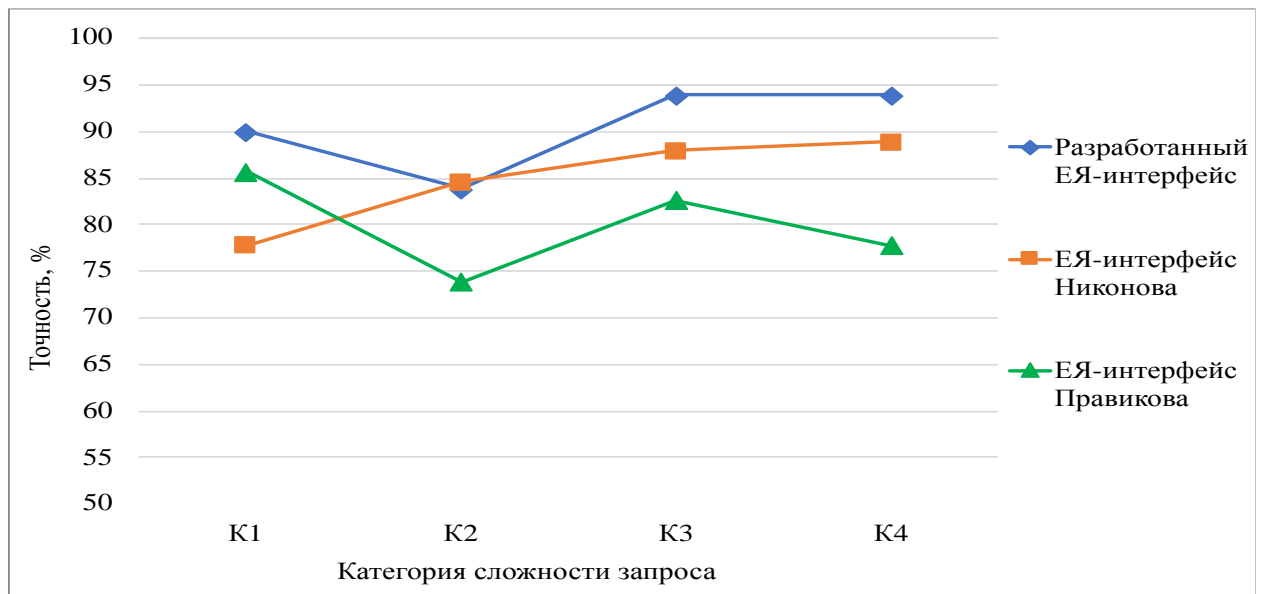


Рисунок 5.4 – Точность работы естественно-языковых интерфейсов в зависимости от категории сложности запроса

Естественно-языковой интерфейс способен обрабатывать определенный объем правильно интерпретируемых запросов пользователя. Объем понимаемых запросов характеризуется полнотой: $Re = |D_{rel} \cap D_{retr}| / |D_{rel}|$. Чем больше значение полноты естественно-языкового интерфейса, тем более

корректно интерфейс способен выделить ключевые сущности из естественно-языкового запроса. Также полнота показывает степень достаточности информации о семантике сущностей и связях между ними для формирования запроса к базе данных. Результаты оценки полноты естественно-языковых интерфейсов в зависимости от категории сложности запроса представлены на рисунке 5.5

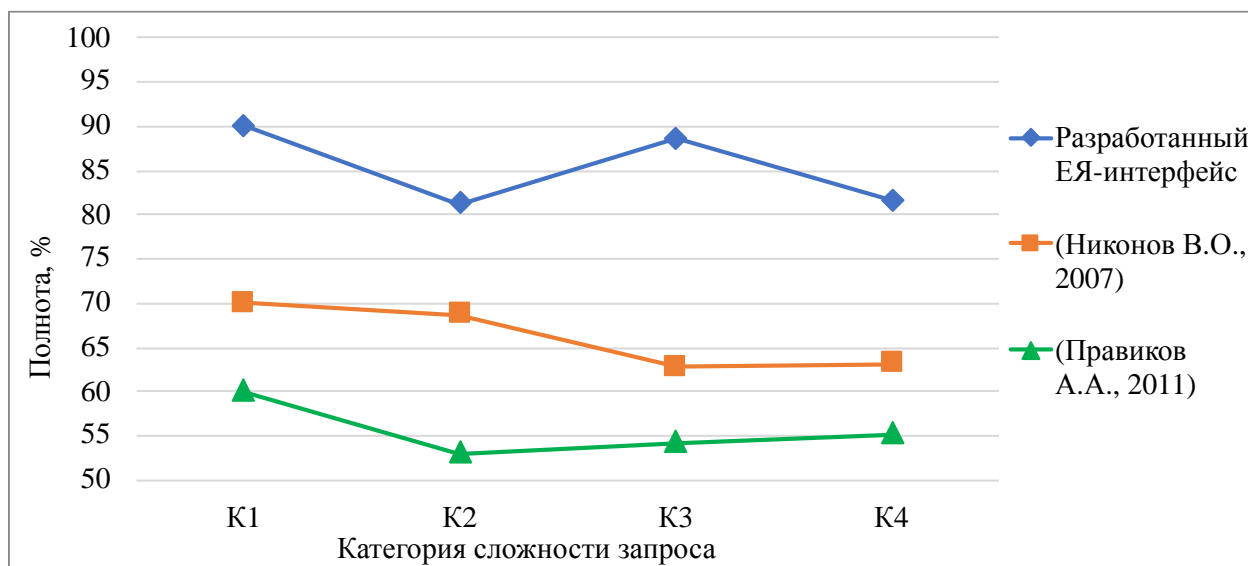


Рисунок 5.5 – Полнота естественно-языковых интерфейсов в зависимости от категории сложности запроса

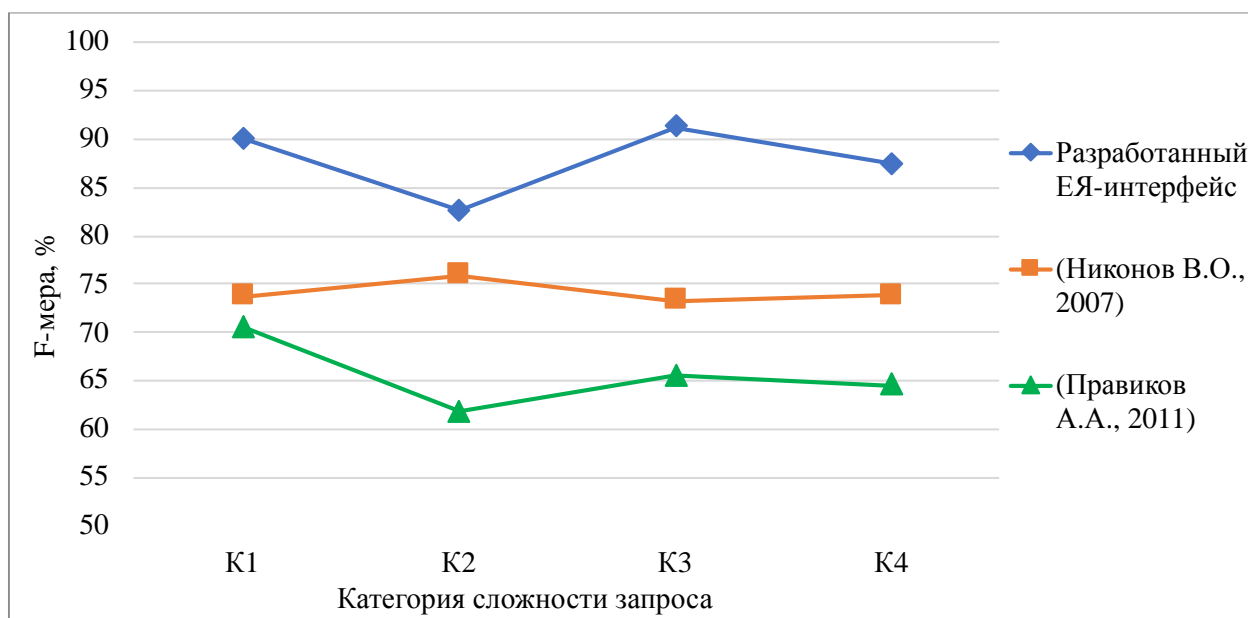


Рисунок 5.6 – F-мера естественно-языковых интерфейсов в зависимости от категории сложности запроса

Так как точность и полнота являются важными характеристиками для оценки работы естественно-языкового интерфейса, то для оценки этих

показателей в совокупности, используется комбинированная метрика F-мера, рассчитываемая как взвешенное гармоническое среднее основе точности и полноты: $F = 2PrRe/(Pr+Re)$. Результаты оценки F-меры естественно-языковых интерфейсов в зависимости от категории сложности запроса представлены на рисунке 5.6

Сравнительный анализ точности, полноты и F-меры естественно-языковых пользовательских интерфейсов представлен на рисунке 5.7.

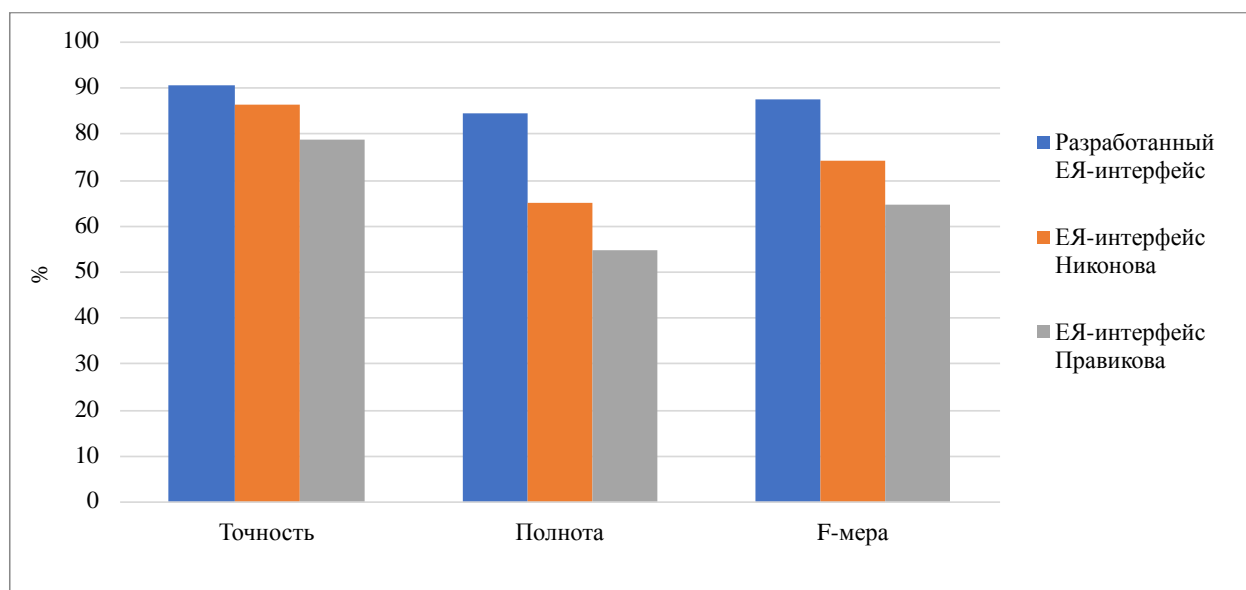


Рисунок 5.7 – Сравнительный анализ точности, полноты и F-меры естественно-языковых пользовательских интерфейсов

Результаты экспериментального исследования показали, что естественно-языковой интерфейс позволяет успешно обработать естественно-языковой запрос, построить на его основе корректный SQL-запрос к базе данных. В результате, пользователь получает возможность получить релевантный ответ на свой вопрос. Разработанный естественно-языковой интерфейс к базам данных продемонстрировал улучшение показателей корректности работы интерфейса. Эксперименты показали увеличение точности на 4.5%, полноты – на 19%, комплексной характеристики F-мера – на 13%, по сравнению с лучшими показателями среди альтернативных естественно-языковых интерфейсов (Никонов В.О., 2007) (Правиков А.А., и др., 2010), работающих с запросами на русском языке.

5.2 Оценка эффективности естественно-языкового интерфейса

5.2.1 Постановка экспериментального исследования

Предварительно было подготовлено 115 вопросов по предметной области тестовой базы данных. Далее добровольцу требовалось получить ответы на эти вопросы из базы данных с помощью:

- естественно-языкового интерфейса;
- формального языка запроса;
- интерфейса с графическим формированием запроса.

В качестве интерфейса с графическим формированием запросов использовались встроенные средства Microsoft Access. Для этого были предварительно импортированы данные из тестовой базы данных.

Для оценки эффективности естественно-языкового интерфейса применялся такой показатель как время, затрачиваемое пользователем для получения ответа от базы данных.

Так же, как и в эксперименте по оценке полноты, точности и F-меры естественно-языкового интерфейса, 82 добровольца стали участниками экспериментального исследования.

5.2.2 Состав и организация тестового окружения

Аналогично эксперименту по оценке полноты, точности и F-меры естественно-языкового интерфейса, исследование проведено на сформированной тестовой базе данных. Таким образом, состав и организация тестового окружения для естественно-языкового интерфейса представлены в разделе 5.1.2. При формировании запроса на формальном языке запроса (SQL) взаимодействие производилось с этой же базой данных. Для эксперимента с интерфейсом графического представления запроса данные из тестовой базы данных были импортированы в базу данных Microsoft Access.

5.2.3 Результаты экспериментального исследования

Проведен сравнительный анализ и тестовые испытания, демонстрирующие преимущества использования естественно-языкового пользовательского интерфейса при работе с базой данных. Критерием оценки эффективности использования интерфейса является время нахождения ответа на вопрос. В рамках эксперимента осуществлялся поиск ответов на заранее сформированные вопросы посредством взаимодействия с интерфейсом графического построения запросов, формального языка запросов (SQL) и естественно-языкового интерфейса к базе данных. Процесс получения ответа на вопрос, в свою очередь, состоит из этапов непосредственно пользовательского ввода, формирования SQL-запроса, а также исправления запроса в случае получения нерелевантных ответов или ошибки. При этом для формального языка запроса этап формирования SQL-запроса, фактически, совпадает с этапом пользовательского ввода. Результаты эксперимента по поиску ответов на вопросы с использованием различных видов интерфейсов к базе данных представлены в таблице 5.30.

Таблица 5.30– Время поиска ответов на вопросы с использованием различных видов интерфейсов к базе данных

	Интерфейс графического построения запросов	Формальный язык запроса	Естественно- языковой интерфейс
Длительность пользовательского ввода, с	30.34	18.84	8.78
Длительность формирования SQL-запроса, с	0.29	0	0.97
Длительность исправления запроса, с	3.23	1.59	2.56
Всего	33.86	20.43	12.31

Сравнительный анализ времени поиска ответа на вопросы в зависимости от используемого интерфейса взаимодействия с базой данных представлен на рисунке 5.8.

Использование естественно-языкового интерфейса позволяет сократить время получения ответа в 1.66 раза по сравнению с составлением вручную SQL-запроса и в 2.75 раза по сравнению с использованием интерфейса графического построения запроса. Таким образом, данные сравнительного анализа продемонстрировали эффективность предложенного решения по сравнению с альтернативными решениями.

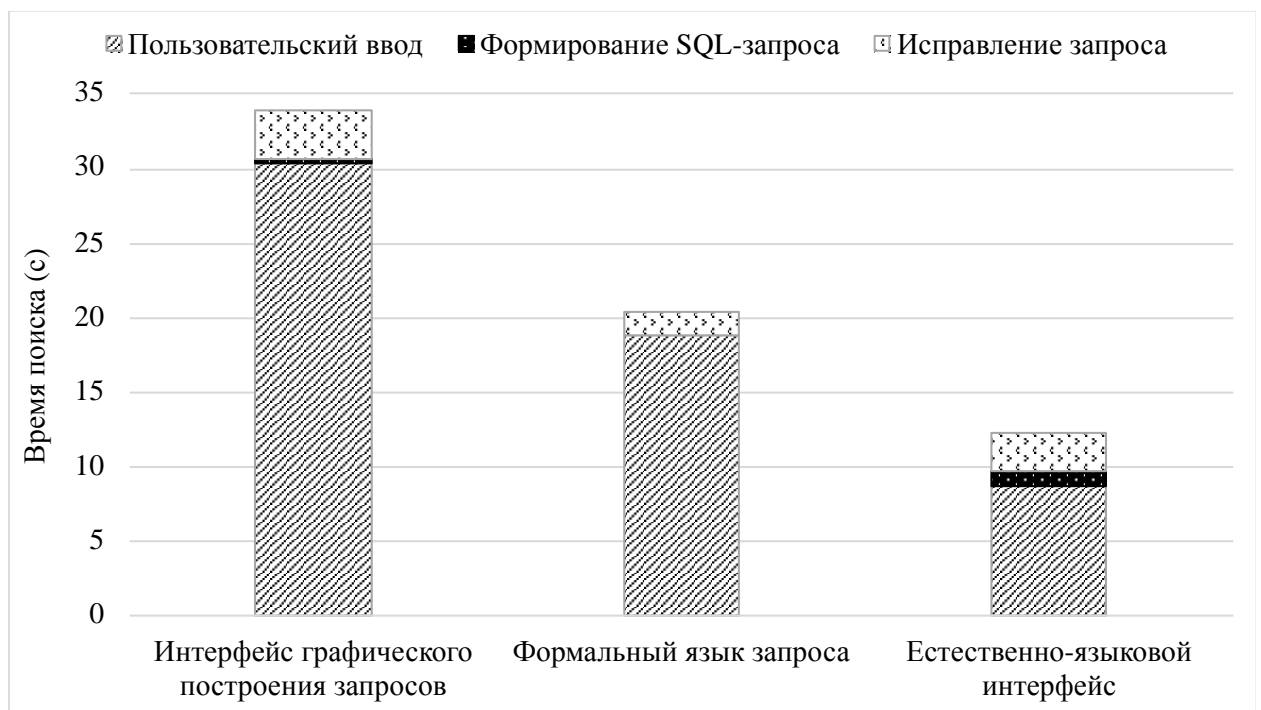


Рисунок 5.8 – Сравнительный анализ времени поиска ответа на вопросы в зависимости от используемого интерфейса взаимодействия с базой данных

5.3 Оценка корректности работы естественно-языкового интерфейса, портированного на другой естественный язык

5.3.1. Постановка экспериментального исследования

Для оценки корректности работы разработанного естественно-языкового интерфейса на другой естественный язык проведены эксперименты с использованием баз данных, содержащих в себе информацию на английском языке. Естественно-языковые интерфейсы, представленные в сравнительном

обзоре, недоступны для скачивания для воспроизведения экспериментов. Однако, для систем Sqlizer и Nalir доступны непосредственно результаты экспериментов. Таким образом, методика проведения данного эксперимента соответствует методике проведения эксперимента, осуществленного при разработке естественно-языкового пользовательского интерфейса к базам данных Sqlizer (Yaghmazadeh N., et al., 2017).

В рамках эксперимента разработанный естественно-языковой интерфейс был портирован для работы с данными, представленными на английском языке. В эксперименте используются общедоступные базы данных Microsoft Academic Search Database (MAS), база данных YELP, база данных IMDB. Для каждой из баз данных была сформирована семантическая модель.

Таблица 5.31 – Распределение запросов разных категорий в зависимости от базы данных

База данных	Категория	Количество запросов	%
MAS	K1	14	7
	K2	59	30
	K3	60	31
	K4	63	32
	Всего	196	100
IMDB	K1	18	13
	K2	69	53
	K3	27	21
	K4	17	13
	Всего	131	100
YELP	K1	8	6
	K2	49	38
	K3	51	40
	K4	20	16
	Всего	128	100

Были использованы 455 естественно-языковых запросов из эксперимента, описанного в (Yaghmazadeh N., et al., 2017). Из них, 196 запросов к базе данных MAS, 128 запросов к базе данных YELP, 131 запрос к базе данных IMDB. При этом все запросы разделены на 4 категории сложности, которые представлены в таблице 5.1. Распределение запросов разных категорий в зависимости от базы данных представлено в таблице 5.31.

Далее проводится сравнительный анализ результатов экспериментального исследования разработанного естественно-языкового интерфейса, портированного на другой естественный язык, а также интерфейсов Sqlizer и Nalir.

5.3.2. Состав и организация тестового окружения

Для экспериментального исследования разработанного естественно-языкового пользовательского интерфейса с учетом портирования на другой естественный язык использованы общедоступные базы данных – Microsoft Academic Search Database (MAS), база данных YELP, база данных IMDB.

База данных Microsoft Academic Search содержит в себе информацию об опубликованных научных статьях. YELP содержит в себе базу данных отзывов пользователей об услугах, предоставляемых коммерческими организациями. IMDB содержит в себе сведения о фильмах. Конфигурация баз данных представлена в таблице 5.32.

Таблица 5.32 – Конфигурация баз данных, содержащих информацию на английском языке

База данных	Размер	Количество таблиц	Количество столбцов
MAS	3.2 Гб	17	53
IMDB	1.3 Гб	16	65
YELP	2.0 Гб	7	38

5.3.3. Результаты экспериментального исследования

Результаты обработки запросов к базам данных MAS, IMDB, YELP, представленных на английском языке, с помощью разработанного

естественно-языкового интерфейса, портированного на другой естественный язык, а также интерфейсов Sqlizer и Nalir представлены в таблице 5.33. Так, в таблице отображено общее количество запросов в зависимости от категории сложности, а также используемой базы данных. Для каждого из интерфейсов представлено количество полученных релевантных ответов, а также их процентное соотношение от общего количества запросов данной категории.

Таблица 5.33 – Результаты обработки естественно-языковыми интерфейсами англоязычных запросов

БД	Категория	# запросов	Портированный ЕЯ-интерфейс		Sqlizer		Nalir	
			#	%	#	%	#	%
MAS	K1	14	13	92.9	12	85.7	11	78.6
	K2	59	56	94.9	52	88.1	39	66.1
	K3	60	52	86.7	49	81.7	34	56.7
	K4	63	55	87.3	45	71.4	30	47.6
	Всего	196	176	89.8	158	80.6	114	58.2
IMDB	K1	18	15	83.3	16	88.9	7	38.9
	K2	69	61	88.4	51	73.9	12	17.4
	K3	27	24	88.9	24	88.9	4	14.8
	K4	17	14	82.4	11	64.7	2	11.8
	Всего	131	114	87.0	102	77.9	25	19.1
YELP	K1	8	7	87.5	6	75.0	4	50.0
	K2	49	45	91.8	35	71.4	8	16.3
	K3	51	43	84.3	40	78.4	6	11.8
	K4	20	17	85.0	15	75.0	2	10.0
	Всего	128	112	87.5	96	75.0	20	15.6

В соответствии с методикой исследования, представленной в (Li F., et al, 2014), а также в дальнейшем использованной в (Yaghmazadeh N., et al., 2017), в качестве критерия корректности работы естественно-языкового интерфейса используется количество полученных релевантных ответов, выраженное в

процентах. Динамика изменения данного показателя в зависимости от категории сложности запроса, используемого естественно-языкового интерфейса и базы данных представлены на рисунках 5.9 – 5.11.

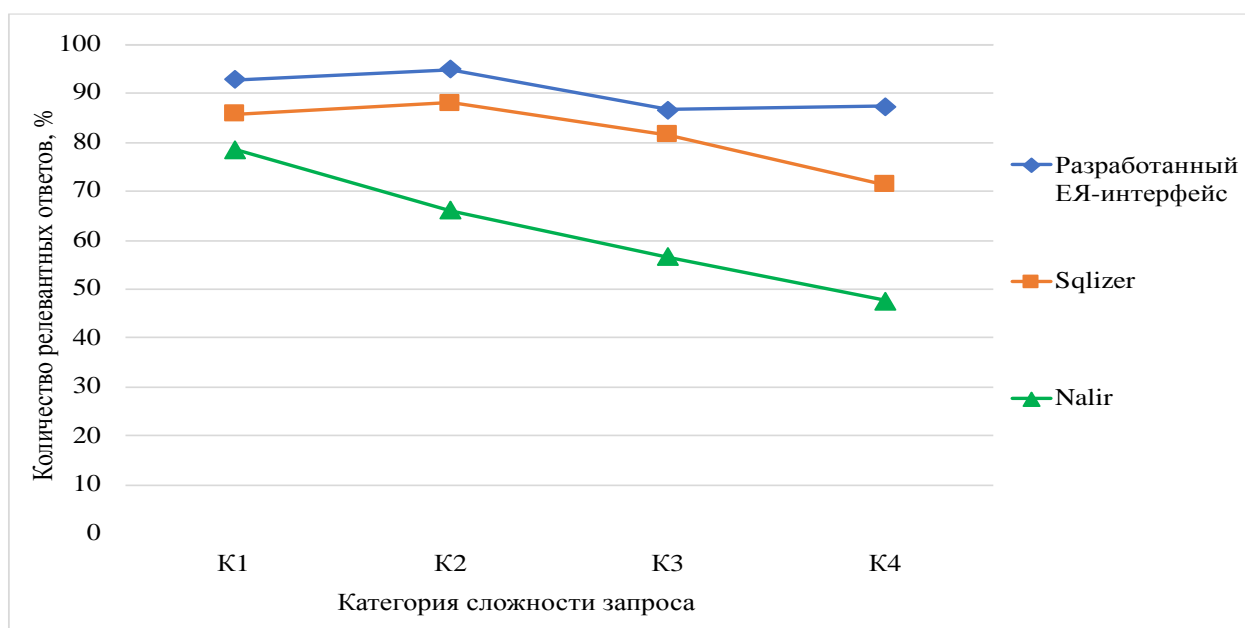


Рисунок 5.9 – Количество релевантных ответов, полученных при взаимодействии с базой данных MAS

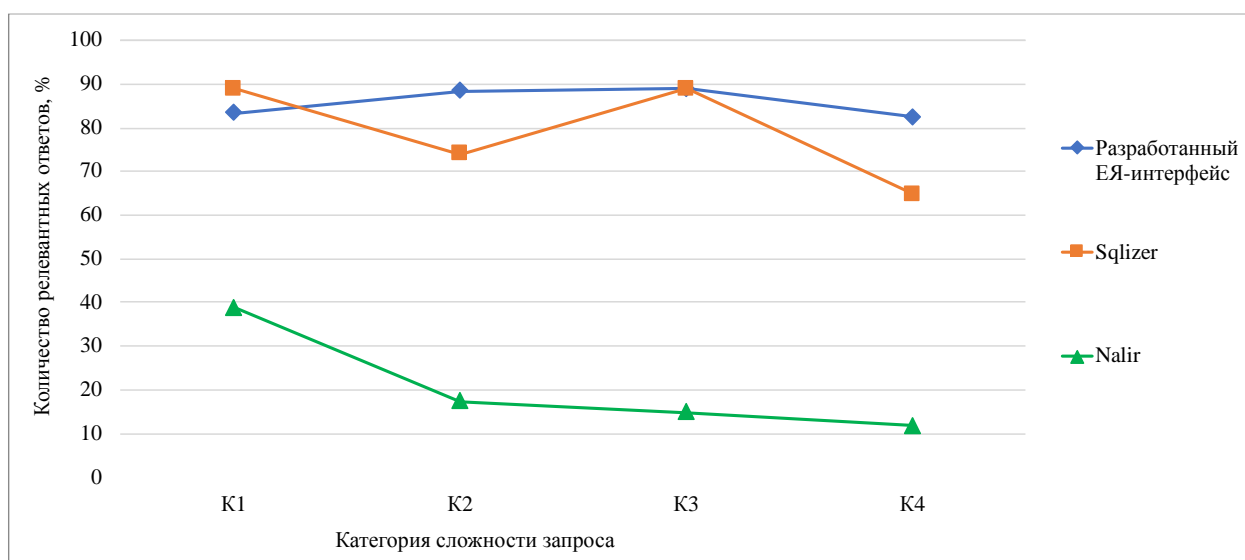


Рисунок 5.10 – Количество релевантных ответов, полученных при взаимодействии с базой данных IMDB

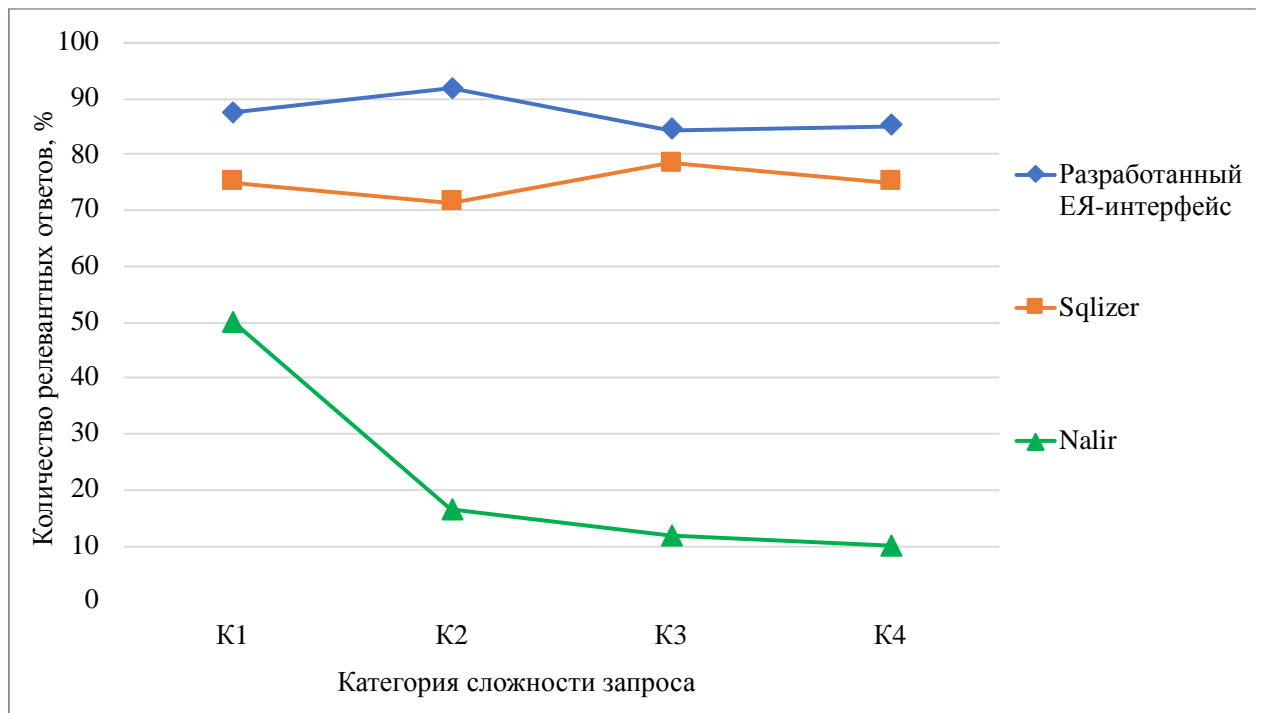


Рисунок 5.11 – Количество релевантных ответов, полученных при взаимодействии с базой данных YELP

На рисунке 5.12 представлены средние значения количества релевантных ответов для всех трех баз данных, а именно: 88% для разработанного естественно-языкового интерфейса, 78% – для Sqlizer, 31% – для Nalir.

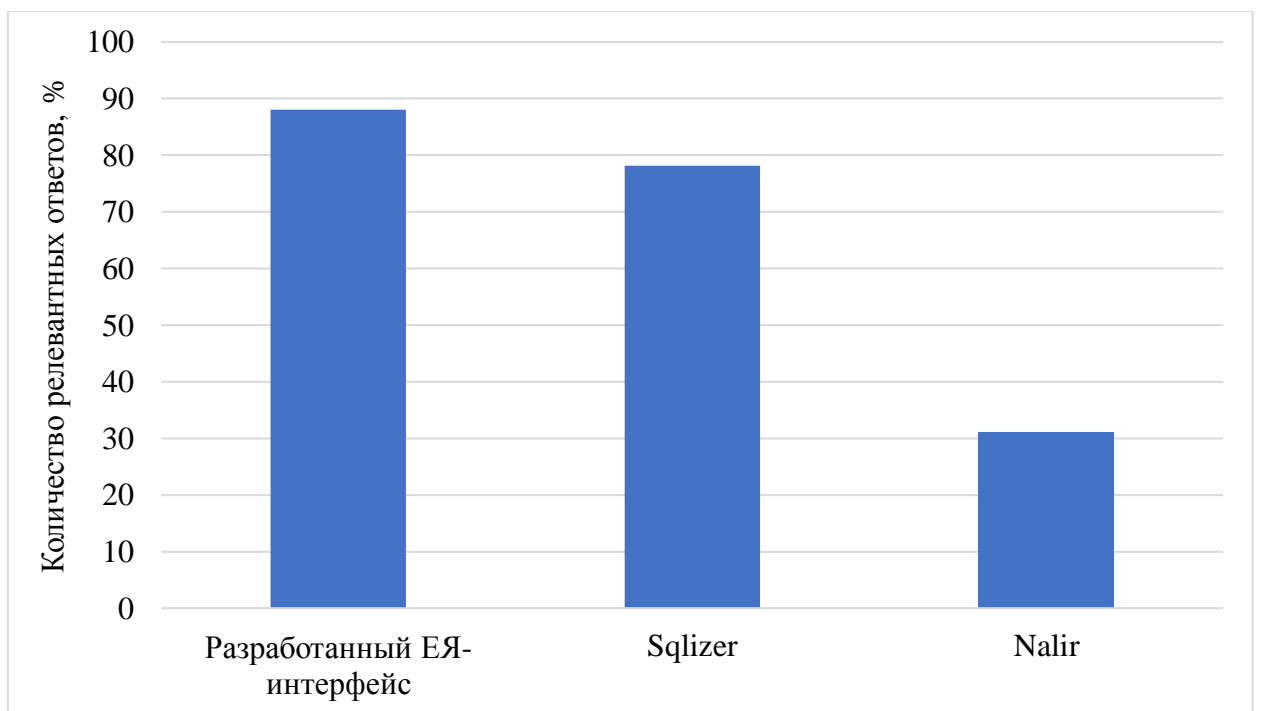


Рисунок 5.12 – Количество релевантных ответов, полученных при взаимодействии с базами данных MAS, IMDB и YELP

Количество релевантных ответов, по сути, характеризует полноту естественно-языкового интерфейса. Аналогично результатам, приведенным в главе 5.3.1, в таблице 5.34 приведены данные о количестве успешно обработанных запросов $|D_{retr}|$, количестве релевантных ответов $|D_{rel} \cap D_{retr}|$, а также точности Pr , полноте Re , и комплексной характеристике F -мере для запросов из каждой категории сложности для всех трех баз данных. Результаты F -меры разработанного естественно-языкового интерфейса, портированного на другой естественный язык, для каждой категории сложности запросов в зависимости от используемой базы данных представлены на рисунке

Таблица 5.34 – Полнота, точность и комбинированная F -мера разработанного естественно-языкового интерфейса, портированного на другой естественный язык

БД	Категория	# запросов	$ D_{retr} $	$ D_{rel} \cap D_{retr} $	Pr	Re	F -мера
MAS	K1	14	13	13	100	92.9	96.3
	K2	59	58	56	96.6	94.9	95.7
	K3	60	58	52	89.7	86.7	88.1
	K4	63	61	55	90.2	87.3	88.7
	Всего	196	190	176	92.6	89.8	91.2
IMDB	K1	18	16	15	93.8	83.3	88.2
	K2	69	67	61	91.0	88.4	89.7
	K3	27	26	24	92.3	88.9	90.6
	K4	17	16	14	87.5	82.4	84.8
	Всего	131	125	114	91.2	87.0	89.1
YELP	K1	8	8	7	87.5	87.5	87.5
	K2	49	47	45	95.7	91.8	93.8
	K3	51	49	43	87.8	84.3	86.0
	K4	20	19	17	89.5	85.0	87.2
	Всего	128	123	112	91.1	87.5	89.2

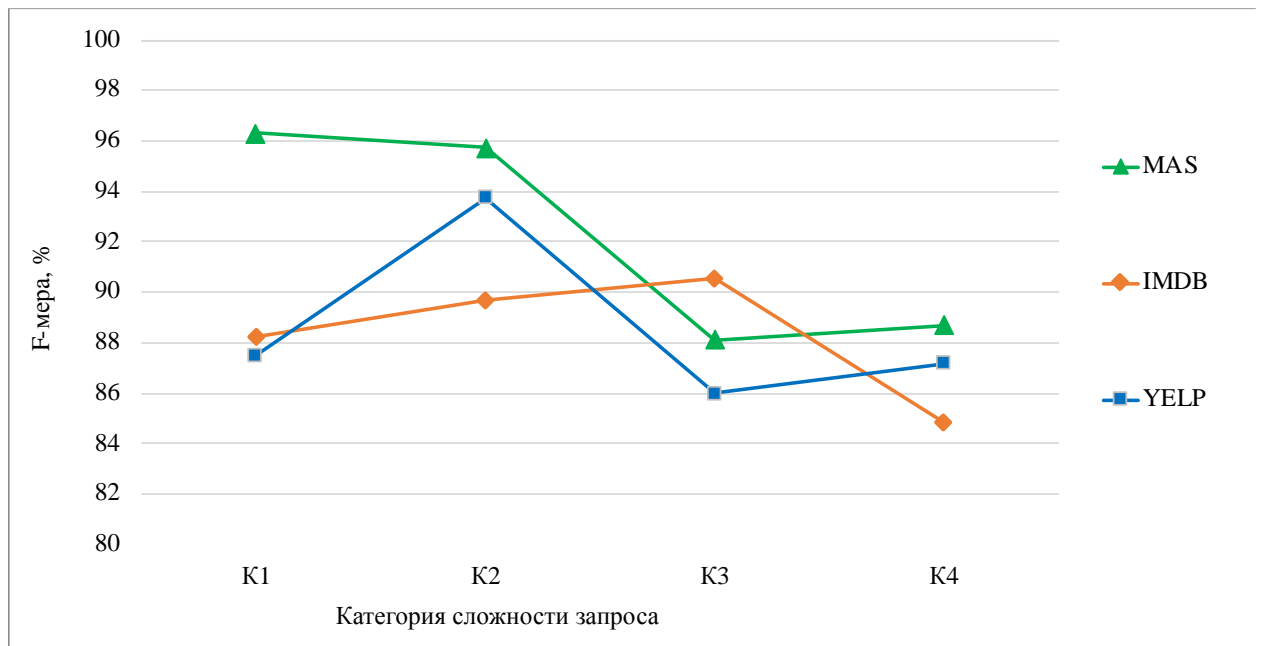


Рисунок 5.13 – F-мера разработанного естественно-языкового интерфейса, портированного на другой естественный язык, для каждой категории сложности запросов для баз данных MAS, IMDB, YELP

Естественно-языковые интерфейсы *Sqlizer* и *Nalir* обрабатывают пользовательские запросы исключая взаимодействие с пользователем и уточнение информации или исправление запроса. Разработанный естественно-языковой интерфейс для данного эксперимента аналогичным образом обрабатывал запросы исключая этап взаимодействия с пользователем. В то же время, системы *Sqlizer* и *Nalir* относятся к классу «database-agnostic» естественно-языковых интерфейсов, то есть семантическая информация получается системой в автоматическом режиме в процессе обработки запроса, что исключает необходимость в предварительной настройке работы системы для конкретной базы данных, однако приводит к более низкому показателю количества полученных релевантных ответов.

Таким образом, проведенные эксперименты продемонстрировали возможность портирования разработанного естественно-языкового интерфейса, а также увеличение в этом случае количества релевантных ответов на 10%, относительно аналогичных естественно-языковых интерфейсов (*Yaghmazadeh N., et al., 2017*) и (*Li F., et al, 2014*).

Выводы

Проведены экспериментальные исследования по оценке качества и эффективности разработанного естественно-языкового пользовательского интерфейса на тестовой базе данных. Результаты экспериментального исследования демонстрируют работоспособность разработанных методов и алгоритмов. Естественно-языковой интерфейс позволяет успешно обработать естественно-языковой запрос и построить на его основе корректный SQL-запрос к базе данных. В результате, пользователь получает возможность получить релевантный ответ на свой вопрос. Разработанный естественно-языковой интерфейс к базам данных продемонстрировал улучшение показателей корректности работы интерфейса. Эксперименты показали увеличение точности на 4.5%, полноты – на 19%, комплексной характеристики F-мера – на 13%, по сравнению с лучшими показателями среди альтернативных естественно-языковых интерфейсов (Никонов В.О., 2007) (Правилов А.А., и др., 2010), работающих с запросами на русском языке.

Помимо этого, использование естественно-языкового интерфейса позволяет сократить время получения ответа в 1.66 раза по сравнению с составлением вручную SQL-запроса и в 2.75 раза по сравнению с использованием интерфейса графического построения запроса. Таким образом, результаты экспериментов продемонстрировали эффективность предложенного решения по сравнению с альтернативными решениями.

Также проведены эксперименты по оценке корректности работы разработанного естественно-языкового интерфейса, портированного на другой естественный язык. В результате, эксперименты продемонстрировали как возможность портирования разработанного естественно-языкового интерфейса, так и увеличение объема корректно обработанных запросов на 10%, относительно альтернативных естественно-языковых интерфейсов Nalir и Sqlizer.

Представленные в главе результаты опубликованы в (Посевкин Р.В., 2018).

Заключение

В рамках диссертационной работы были получены следующие результаты:

1. Предложен метод построения естественно-языкового пользовательского интерфейса к базам данных, отличающийся использованием шаблонов моделей предложений и обеспечивающий автоматическое извлечение данных без необходимости формирования пользователем SQL-запроса.

2. Разработана семантическая модель базы данных и алгоритм ее автоматизированного формирования, в том числе:

метод определения семантики типа связей между сущностями базы данных с использованием тезауруса;

метод анализа семантики сущностей базы данных на основе паттернов;

программа, реализующая алгоритм автоматизированного формирования семантической модели базы данных. В отличие от известных решений, данные, необходимые для построения запроса к базе данных на основе естественно-языкового запроса формируются в автоматизированном режиме, что сокращает трудозатраты, необходимые для внедрения естественно-языкового пользовательского интерфейса.

3. Предложен алгоритм построения запроса к базам данных на основе анализа текста, введенного пользователем на естественном языке, позволяющий обеспечить портируемость естественно-языкового интерфейса на другие естественные языки и формальные языки запроса за счет использования семантической модели базы данных и К-представления естественно-языкового запроса при формировании запроса к базе данных. В результате, успешно решена проблема определения связей между терминами естественного языка и сущностями даталогической модели базы данных без привязки к конструкциям и операторам конкретного формального языка запросов к базе данных. В отличие от известных решений, разработанный интерфейс допускает возможность портирования на другие естественные и

формальные языки запроса. Данная возможность достигается за счет формирования К-представления запроса, а также модульности компонент интерфейса.

4. Выполнены экспериментальные исследования с использованием разработанных методов и алгоритмов, подтверждающие их работоспособность и эффективность. На основе сформированной тестовой базы данных проведена оценка полноты, точности и F-меры разработанного естественно-языкового интерфейса. Разработанный естественно-языковой интерфейс к базам данных продемонстрировал улучшение точности на 4.5%, полноты – на 19%, комплексной характеристики F-мера – на 13%, по сравнению с лучшими показателями среди альтернативных естественно-языковых интерфейсов.

Проведен сравнительный анализ и эксперименты, демонстрирующие эффективность использования естественно-языкового пользовательского интерфейса при работе с базой данных по сравнению с альтернативными видами интерфейсов. Использование естественно-языкового интерфейса позволяет сократить время получения ответа в 1.66 раза по сравнению с составлением вручную SQL-запроса и в 2.75 раза по сравнению с использованием интерфейса графического построения запроса.

Проведены эксперименты по оценке корректности работы разработанного естественно-языкового интерфейса, портированного на другой естественный язык. В результате, эксперименты продемонстрировали как возможность портирования разработанного естественно-языкового интерфейса, так и увеличение объема корректно обработанных запросов на 10%, относительно аналогичных естественно-языковых Sqlizer и Nalir

С помощью вычислительных экспериментов установлено, что разработанные алгоритмы позволяют получать релевантные ответы на вопросы, сформулированные на естественном языке, при этом пользователю не требуется знать внутреннюю структуру базы данных и вручную формировать SQL-запросы.

Таким образом, выполнены все задачи диссертации, что, в свою очередь, позволило достичь поставленной цели по повышению доступности информации, размещенной в базах данных для пользователя, не обладающего знаниями и навыками построения SQL-запросов.

В дальнейшем предлагается реализовать описанные в разделе 5.1.3 рекомендации по обнаружению и исправлению некорректных запросов, формируемых естественно-языковым интерфейсом на основе анализа пользовательского запроса. Перспективным направлением дальнейших исследований является минимизация ручных действий, необходимых для предварительной настройки естественно-языкового интерфейса, а также увеличение точности системы при обработке естественно-языковых запросов.

Список литературы

1. Азарова И.В., Митрофанова О.А., Синопальникова А.А. Компьютерный тезаурус русского языка типа WordNet // Труды Международной конференции Диалог. Компьютерная лингвистика и интеллектуальные технологии. – 2003. – С. 43-50.
2. Апресян Ю.Д. Типы информации для поверхностно-семантического компонента модели «Смысл-Текст» // Wiener slawistischer Almanach. Sonderband 1. Wien, 1980.
3. Бессмертный И.А., Нугуманова А.Б., Мансурова М.Е., Байбурин Е.М. Метод контрастного извлечения редких терминов из текстов на естественном языке // Научно-технический вестник информационных технологий, механики и оптики -2017. - Т. 17. - № 1(107). - С. 81-91
4. Богуславский И. М., Иомдин Л. Л., Крейдлин Л. Г., Фрид Н. Е., Сагалова, И. Л., Сизов В. Г. Модуль универсального сетевого языка (UNL) в составе системы ЭТАП-3 // Труды международной конференции Диалог. – 2000.
5. Большакова Е. И. Язык лексико-синтаксических шаблонов LSPL: опыт использования и пути развития // Программные системы и инструменты: Тематический сборник. – 2014. – № 15.
6. Большакова Е.И., Воронцов К.В., Ефремова Н.Э., Клышинский Э.С., Лукашевич Н.В., Сапин А.С. Автоматическая обработка текстов на естественном языке и анализ данных: учебное пособие – М.: Изд-во НИУ ВШЭ, 2017. – 269 с.
7. Браславский П. И. Тезаурус для расширения запросов к машинам поиска Интернета: структура и функции // Труды Международной конференции Диалог. Компьютерная лингвистика и интеллектуальные технологии. – 2003. – Т. 2003. – С. 95-100.
8. Вендров А. М. CASE-технологии. Современные методы и средства проектирования информационных систем // М.: Финансы и статистика. – 1998. – Т. 175. – №. 1.

9. Гуслякова А.В. Информационные технологии и лингвистика XXI века. – М:МПГУ, 2016. – 130 с.
10. Диненберг Ф. Г., Кучин С. И., Трапезников С. П. INTERBASE-система конструирования ЕЯ-интерфейса к dBASE III PLUS // Искусственный интеллект-90: Тр. – 1990. – С. 161-163.
11. Евдокимова И.С. Методы и алгоритмы трансляции естественно-языковых запросов к базе данных: Автореф. дис. ... кан. тех. наук //Красноярск, 2004 – 18 с.
12. Ермаков А. Е. Неполный синтаксический анализ текста в информационно-поисковых системах //Компьютерная лингвистика и интеллектуальные технологии: труды Международного семинара Диалог. – 2002. – №. 2.
13. Елисеева О.Е. Естественно-языковой интерфейс интеллектуальных систем: учебное пособие. – Минск: БГУИР, 2009. – 151 с.
14. Ермаков А. Е., Киселев С. Л., Плешко В. В. Поиск фактов в тексте естественного языка на основе сетевых описаний // Компьютерная лингвистика и интеллектуальные технологии: труды Международной конференции Диалог. – 2004. – С. 282-285.
15. Ермаков А. Е., Плешко В. В., Митюнин В. А. RCO Pattern Extractor: компонент выделения особых объектов в тексте // Сборник трудов XII Международной научной конференции «Информатизация и информационная безопасность правоохранительных органов» – Москва. – 2003. – С. 312-317.
16. Жигалов В. А. Основные характеристики и составляющие части ЕЯ-интерфейсов // Труды международного семинара Диалог-1997 по компьютерной лингвистике и ее приложениям. – Москва, 1997 [Электронный ресурс] – URL: <http://masters.donntu.org/2001/fvti/marchenko/diss/lib/art2.htm>
17. Жигалов В. А. Технология построения естественно-языковых интерфейсов к структурированным источникам данных: Автореф. дис. ... кан. тех. наук // Москва, 2000 – 25 с.

18. Жигалов В. А., Соколова Е. Г. InBASE: технология построения ЕЯ-интерфейсов к базам данных // Труды Международного семинара Диалог-2001 по компьютерной лингвистике и ее приложениям. – 2001. – С. 123-135.
19. Житко В.А. Пользовательский интерфейс интеллектуальных вопросно-ответных систему // NB: Кибернетика и программирование. — 2012. - № 1. - С.23-30.
20. Житко В. А. и др. Семантическая технология компонентного проектирования естественно-языкового интерфейса интеллектуальных вопросно-ответных систем. // Труды Международной научно-технической конференции Open Semantic Technology for Intelligent Systems (OSTIS) 2011 – 2011. — С. 395-408.
21. Зализняк А. А. Грамматический словарь русского языка. Словоизменение. – М.: Русский язык, 1977. – 879 с.
22. Зализняк А. А. Русское именное словоизменение. – М.: Наука, 1967 – 372 с.
23. Крайванова В. А. Модель естественно-языкового интерфейса для систем управления сложными техническими объектами и оценка эффективности алгоритмов на ее основе // Управление большими системами: сборник трудов. – 2009. – №. 26 – С. 158-177.
24. Кузнецов Б. А. и др. Обработка запросов на естественном языке-новое качество поиска в БД ВИНТИ // НТИ. Серия 2. – 2001. – №. 11. – С. 31.
25. Кузнецова А.И., Ефремова Т.Ф. Словарь морфем русского языка. – М. Русский язык, 1986. – 1136 с.
26. Мальковский М. Г. Диалог с системой искусственного интеллекта. – М:МГУ, 1985. – 232 с.
27. Мальковский М. Г., Шикин И. Ю. Нечеткий лингвистический интерфейс // Программирование. – 1998. – №. 4. – С. 50-61.
28. Нариньяни А.С. Лингвистические процессоры ЗАПСИБ (Часть 1 – задачи проекта). – Новосибирск, 1979 – 22 с. (Препринт/ВЦ СО АН СССР; №199).

29. Нариньяни А.С. Лингвистические процессоры ЗАПСИБ (Часть 2 – общая схема и основные модули). – Новосибирск, 1979 – 48 с. (Препринт/ВЦ СО АН СССР; №202).
30. Николаева И. С., Митренина О. В., Ландо Т. М. Прикладная и компьютерная лингвистика // М.: URSS. – 2016 – 315 с.
31. Никонов В. О. Диалоговая интеллектуальная система с естественно-языковым интерфейсом: Автореф. дис. ... кан. тех. наук // Краснодар, 2007 – 24 с.
32. Осипов Г. С. и др. Проблемы обеспечения точности и полноты поиска: Пути решения в интеллектуальной метапоисковой системе" Сириус" // Труды международной конференции Диалог. – 2005. – С. 390-395.
33. Попов Э. В. Общение с ЭВМ на естественном языке. – М.: Наука, 1982. – 360 с.
34. Посевкин Р.В. Применение семантической модели базы данных при реализации естественно-языкового пользовательского интерфейса // Научно-технический вестник информационных технологий, механики и оптики. – 2018. – Т. 18. – № 2. – С. 262–267.
35. Посевкин Р.В. Метод автоматизированного формирования семантической модели базы данных диалоговой системы // Программные продукты и системы. – 2018. – № 2. – С. 291–294.
36. Посевкин Р.В. Обработка естественного языка в процессе разработки пользовательского интерфейса // Сборник научных трудов III Международной научной конференции «Информационные технологии в науке, управлении, социальной сфере и медицине». – 2016. – С. 471–472.
37. Посевкин Р.В., Бессмертный И.А. Естественно-языковой пользовательский интерфейс диалоговой системы // Программные продукты и системы. – 2016. – № 3 – С. 5–9.
38. Правиков А. А. Разработка и применение метода формализации проектирования рекомендательных систем с естественно-языковым интерфейсом: канд. технических наук // Москва, 2011 – 160 с.

39. Правиков А.А., Фомичев В.А. Разработка рекомендательной системы с естественно-языковым интерфейсом на основе математических моделей семантических объектов // Бизнес-информатика. – 2010. – № 4. – С. 3–8.
40. Селезнев К. Обработка текстов на естественном языке // Открытые системы. – 2003. – Т. 12. [Электронный ресурс] – URL: <https://www.osp.ru/os/2003/12/183694/>
41. Тихонов А.Н. Морфемно-орфографический словарь. М.: АСТ: Астрель, 2002. — 704 с.
42. Толдова С. Ю. и др. Система Alex как средство для многоцелевой автоматизированной обработки текстов // Труды Международной конференции Диалог. Компьютерная лингвистика и интеллектуальные технологии. – М.: Наука, 2002. – Т.2 – С.192-208.
43. Фомичев В.А. Математические основы представления содержания посланий компьютерных интеллектуальных агентов. — М.: ГУ-ВШЭ, 2007. — 176 с.
44. Фомичёв В. А. Подход теории к-представления к формальному отображению содержания контрактов и протоколов переговоров в области электронной коммерции // Бизнес-информатика. – 2007. – №. 2.
45. Хомский Н. Синтаксические структуры. // Новое в лингвистике. Вып. II. — М., 1962. — С. 412-527.
46. Чернова И. Использование API Яндекс. спеллер. Готовые рецепты для веб-разработчика // Системный администратор. – 2014. – №. 5. – С. 54-55.
47. Якубенко А. П., Селезнев К. Е., Артемов М. А. Оптимальная организация морфологических словарей в поисковых системах // Вестник Воронежского государственного университета. Серия: Системный анализ и информационные технологии. – 2006. – №. 2. – С. 156-161.
48. Androutsopoulos I., Ritchie G. D., Thanisch P. Natural language interfaces to databases – an introduction // Natural language engineering. – 1995. – Т. 1. – №. 1. – С. 29-81.

49. Auxerre P. MASQUE Modular Answering System for Queries in English - Programmer's Manual. // Technical Report AIAI/SR/11, Artificial Intelligence Applications Institute – University of Edinburgh, March 1986.
50. Azarova I. et al. Russnet: Building a lexical database for the russian language // Proceedings of Workshop on Wordnet Structures and Standardisation and How this affect Wordnet Applications and Evaluation. Las Palmas. – 2002. – C. 60-64.
51. Bessmertny I. A. et al. Syntactic text analysis without a dictionary // Application of Information and Communication Technologies (AICT), 2016 IEEE 10th International Conference on. – IEEE, 2016. – C. 1-3.
52. Binot J. L. et al. Natural language interfaces: a new philosophy // SunExpert Magazine. – 1991. – T. 2. – №. 1. – C. 67-73.
53. Braslavski P., Ustalov D., Mukhin M. A spinning wheel for YARN: user interface for a crowdsourced thesaurus // Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics. – 2014. – C. 101-104.
54. Cohen P. R. The role of natural language in a multimodal interface // Proceedings of the 5th annual ACM symposium on User interface software and technology. – ACM, 1992. – C. 143-149.
55. Dale R., Moisl H., Somers H. Handbook of natural language processing. – CRC Press, 2000. – 962 C.
56. Daud A., Li J., Zhou L., Muhammad F. Knowledge discovery through directed probabilistic topic models: a survey // Frontiers of computer science in China. – 2010. – T. 4. – №. 2. – C. 280-301.
57. Deemter K. V., Theune M., Krahmer E. Real versus template-based natural language generation: A false opposition? // Computational Linguistics. – 2005. – T. 31. – №. 1. – C. 15-24.
58. Deshpande A. K., Devale P. R. Natural language query processing using probabilistic context free grammar // International Journal of Advances in Engineering & Technology. – 2012. – T. 3. – №. 2. – C. 568-572.

59. Dragoni N. et al. Microservices: yesterday, today, and tomorrow // Present and Ulterior Software Engineering. – Springer, Cham, 2017. – C. 195-216.
60. Fedosejev A. React. js Essentials. – Packt Publishing Ltd, 2015. – 187 C.
61. Gadekar M. D. et al. Natural Language (English) To MongoDB Interface // International Journal of Advanced Research in Computer Engineering & Technology. – 2015. – T. 4. – №. 3. – C. 1081-1083.
62. Gong Y., Liu X. Generic text summarization using relevance measure and latent semantic analysis // Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval. – ACM, 2001. – C. 19-25.
63. Hamilton J. R., Nayak T. K. Microsoft SQL server full-text search //IEEE Data Engineering Bulletin – 2001. – T. 24. – №. 4. – C. 7-10.
64. Hendrix G. G. Natural-language interface // Computational Linguistics. – 1982. – T. 8. – №. 2. – C. 56-61.
65. Hendrix G.G., Sacerdoti E.D., Sagalowicz D., Slocum J. Developing a natural language interface to complex data //ACM Transactions on Database Systems (TODS). – 1978. – T. 3. – №. 2. – C. 105-147.
66. Jurafsky D. Speech and language processing: An introduction to natural language processing // Computational linguistics, and speech recognition. – 2000. – 975 C.
67. Kacprzyk J., Zadrozny S. Fuzzy querying for microsoft access // Fuzzy Systems, 1994. IEEE World Congress on Computational Intelligence., Proceedings of the Third IEEE Conference on. – IEEE, 1994. – C. 167-171.
68. Krizhanovsky A. A., Smirnov A. V. An approach to automated construction of a general-purpose lexical ontology based on Wiktionary // Journal of Computer and Systems Sciences International. – 2013. – T. 52. – №. 2. – C. 215-225.

69. Li F., Jagadish H. V. NaLIR: an interactive natural language interface for querying relational databases // Proceedings of the 2014 ACM SIGMOD international conference on Management of data. – ACM, 2014. – C. 709-712.
70. Linguistic Technology. English Wizard – Dictionary Administrator's Guide. Linguistic Technology Corp., Littleton, MA, USA, 1997.
71. Mardan A. Express.js Guide: The Comprehensive Book on Express.js. – LeanPub, 2014. – 340 C.
72. Miller G. A. WordNet: a lexical database for English // Communications of the ACM. – 1995. – T. 38. – №. 11. – C. 39-41.
73. Nihalani N., Silakari S., Motwani M. Natural language interface for database: a brief review // International Journal of Computer Science Issues (IJCSI). – 2011. – T. 8. – №. 2. – C. 600-608.
74. Pan S., Shaw J. Natural Language Query Recommendation in Conversation Systems // IJCAI. – 2007. – C. 1701-1706.
75. Posevkin R., Bessmertny I. Translation of natural language queries to structured data sources // Application of Information and Communication Technologies (AICT), 2015 9th International Conference on, IET - 2015, pp. 57-59.
76. Posevkin R., Bessmertny I. Multilanguage natural user interface to database // 10th IEEE International Conference on Application of Information and Communication Technologies, AICT 2016 - Conference Proceedings, IET - 2016, pp. 304-306.
77. Ramesh S. H. et al. Towards Building A Domain Agnostic Natural Language Interface to Real-World Relational Databases // Proceedings of the 13th International Conference on Natural Language Processing. – 2016. – C. 305-314.
78. Resnick P.W. Internet message format. 2008. [Электронный ресурс] – URL: <http://www.ietf.org/rfc/rfc5322.txt>
79. Sathick K. J., Jaya A. Natural language to SQL generation for semantic knowledge extraction in social web sources // Indian Journal of Science and Technology. – 2015. – T. 8. – №. 1. – C. 1-10.

80. Shalygina G., Novikov B. Implementing Common Table Expressions for MariaDB // Second Conference on Software Engineering and Information Management (SEIM-2017)(full papers). – 2017. – C. 12-17.
81. Segalovich I.A. Fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine // MLMTA. – 2003. – C. 273-280.
82. Tennant H. R. et al. Menu-based natural language understanding // Proceedings of the 21st annual meeting on Association for Computational Linguistics. – Association for Computational Linguistics, 1983. – C. 151-158.
83. Tilkov S., Vinoski S. Node. js: Using JavaScript to build high-performance network programs // IEEE Internet Computing. – 2010. – T. 14. – №. 6. – C. 80-83.
84. Vlachoudis V. et al. FLAIR: a powerful but user friendly graphical interface for FLUKA // Proc. Int. Conf. on Mathematics, Computational Methods & Reactor Physics (M&C 2009), Saratoga Springs, New York. – 2009.
85. Vossen P. EuroWordNet: a multilingual database for information retrieval. – 1997. – C. 5-7.
86. Winograd T. Understanding Natural Language. // New York: Academic Press, 1972. – C. 1-191.
87. Woods W.A., Kaplan R.M. Lunar rocks in natural English: Explorations in natural language question answering // Linguistic structures processing. – 1977. – T. 5. – C. 521-569.
88. Woods W.A., Kaplan R.M., Nash-Webber B. The Lunar Science Natural Language Information System: Final Report. – Bolt, Beranek and Newman, Incorporated, 1972.
89. Yaghmazadeh N. et al. Sqlizer: Query synthesis from natural language // Proceedings of the ACM on Programming Languages. – 2017. – T. 1. – №. OOPSLA. – C. 63.

**Приложение 1. Свидетельство о регистрации объекта
интеллектуальной собственности**

РОССИЙСКАЯ ФЕДЕРАЦИЯ



СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2018615411

**«SemMoDB - Автоматизированная программная система
формирования семантической модели базы данных»**

Правообладатель: **Посевкин Руслан Владимирович (RU)**

Автор: **Посевкин Руслан Владимирович (RU)**

Заявка № **2018612495**
Дата поступления **19 марта 2018 г.**
Дата государственной регистрации
в Реестре программ для ЭВМ **08 мая 2018 г.**

Руководитель Федеральной службы
по интеллектуальной собственности

 **Г.П. Ивлиев**



Приложение 2. Акты о внедрении результатов диссертационного исследования

федеральное государственное автономное учреждение высшего образования
**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ
ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ**



УТВЕРЖДАЮ
Проректор Университета ИТМО
д.т.н., профессор
Никифоров В.О.
«__» _____ 2018г.

АКТ

**О внедрении в учебный процесс результатов диссертационной работы
Посевкина Руслана Владимировича «Модели, методы и программные
средства построения естественно-языкового пользовательского
интерфейса к базам данных» на соискание ученой степени кандидата
технических наук по специальности 05.13.11 «Математическое и
программное обеспечение вычислительных машин, комплексов
компьютерных сетей»**

Комиссия в составе

председателя	д.т.н., профессора, заведующего кафедрой вычислительной техники Алиева Тауфика Измаиловича
и членов	к.т.н., доцента кафедры вычислительной техники Дергачева Андрея Михайловича, ассистента кафедры вычислительной техники Клименкова Сергея Викторовича

составила настоящий акт о том, что результаты интеллектуальной деятельности аспиранта кафедры вычислительной техники Посевкина Руслана Владимировича, лежащие в основе его диссертационной работы, внедрены в учебный процесс на кафедре вычислительной техники Университета ИТМО для студентов, обучающихся по направлениям 09.04.01.- Информатика и вычислительная техника и 09.04.04 – Программная инженерия.

Внедрение заключается в следующем:

1. Посевкин Руслан Владимирович разработал программную систему
«SemMoDB – Автоматизированная программная система

формирования семантической модели базы данных», которая реализует в себе методы и средства автоматизированного формирования семантической модели базы данных для целей дальнейшей реализации естественно-языкового пользовательского интерфейса к базе данных. Данная программа защищена в качестве объекта интеллектуальной собственности (свидетельство о государственной регистрации программы для ЭВМ №2018615411 от 08.05.2018).

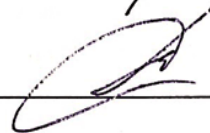
2. Указанная программа включена в учебно-методический комплекс дисциплины «Базы данных» в учебных планах бакалавриата по направлениям подготовки «Информатика и вычислительная техника» и «Программная инженерия».

Председатель комиссии

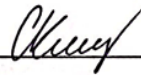


Алиев Т.И.

Члены комиссии



Дергачев А.М.



Клименков С.В.

федеральное государственное автономное учреждение высшего образования
**САНКТ-ПЕТЕРБУРГСКИЙ НАЦИОНАЛЬНЫЙ
 ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ ИНФОРМАЦИОННЫХ
 ТЕХНОЛОГИЙ, МЕХАНИКИ И ОПТИКИ**



УТВЕРЖДАЮ

Проректор Университета ИТМО

д.т.н., профессор

Никифоров В.О.

«__» ____ 2018г.

АКТ

**О внедрении в научно-исследовательскую деятельность результатов
 диссертационной работы Посевкина Руслана Владимировича «Модели,
 методы и программные средства построения естественно-языкового
 пользовательского интерфейса к базам данных» на соискание ученой
 степени кандидата технических наук по специальности 05.13.11
 «Математическое и программное обеспечение вычислительных машин,
 комплексов компьютерных сетей»**

Комиссия в составе

председателя	д.т.н., профессора, руководителя международной лаборатории «Архитектура и методы проектирования встраиваемых систем и систем на кристалле» Платунова Алексея Евгеньевича
и членов	к.т.н., сотрудника международной лаборатории «Архитектура и методы проектирования встраиваемых систем и систем на кристалле» Кустарева Павла Валерьевича, к.т.н., сотрудника международной лаборатории «Архитектура и методы проектирования встраиваемых систем и систем на кристалле» Быковского Сергея Вячеславовича


составила настоящий акт о том, что результаты интеллектуальной
 деятельности аспиранта кафедры вычислительной техники Посевкина
 Руслана Владимировича, лежащие в основе его диссертационной работы,
 внедрены в международную лабораторию «Архитектура и методы

проектирования встраиваемых систем и систем на кристалле» международного научного центра «Нелинейные и адаптивные системы управления» Университета ИТМО.

Внедрение заключается в следующем:


1. Посевкин Руслан Владимирович разработал программную систему «SemMoDB – Автоматизированная программная система формирования семантической модели базы данных», которая реализует в себе методы и средства автоматизированного формирования семантической модели базы данных для целей дальнейшей реализации естественно-языкового пользовательского интерфейса к базе данных. Данная программа защищена в качестве объекта интеллектуальной собственности (свидетельство о государственной регистрации программы для ЭВМ №2018615411 от 08.05.2018).
2. Указанная программа использована при выполнении научно исследовательской работы «Нелинейные и адаптивные системы управления» международной лабораторией «Архитектура и методы проектирования встраиваемых систем и систем на кристалле» (в составе Международного научного центра «Нелинейные и адаптивные системы управления»).

Председатель комиссии

 _____ А.Е.Платунов

Члены комиссии

 _____ П.В.Кустарев

 _____ С.В.Быковский

Приложение 3. Задание участника экспериментального исследования по оценке полноты, точности и F-меры естественно-языкового интерфейса

Вам предоставлен доступ к базе данных, содержащих сведения о внутренней структуре организации. Среди данных, представленных в базе данных можно отметить следующие:

- Сведения о сотруднике: ФИО, пол, возраст, стаж, размер зарплаты, количество детей, документы (СНИЛС, паспорт, ИНН);
- Отделы компании;
- Контактная информация сотрудника – личные и рабочие телефоны, адреса электронной почты, а также экстренные контакты;
- Проекты компании и связанные с ними партнеры;
- Образование сотрудников и их ключевые навыки;
- Офисы компании и расположение конкретных рабочих мест сотрудников;
- Должности сотрудников и уровни доступа;
- Командировки, сертификации, отпуска, участие в курсах повышения квалификации и конференциях, публичные выступления;
- Список оборудования и книг в корпоративной библиотеке.

С помощью естественно-языкового интерфейса необходимо получить ответы на следующие вопросы:

- 1) Информация о курсах, проводившихся весной
- 2) Дата проведения конференции SECR
- 3) Информация о свободных книгах
- 4) Информация о сотрудниках, работающих в одном отделе
- 5) Информация об офисе, расположенном на улице Ленина
- 6) Описание проекта ИСУ
- 7) Должностные обязанности программистов
- 8) Информация о курсах от одной организации
- 9) Узнать срок действия сертификата 3522174

- 10) Информация о курсах, проводимых компанией ABC Education
- 11) Информация о доступных принтерах
- 12) Информация о книгах, которые на руках у одного и того же сотрудника
- 13) Информация о всех проектах компании
- 14) Информация об экстренном контакте сотрудника Захарченко
- 15) Информация о сотрудниках с высшим образованием
- 16) Информация о сотрудниках с одинаковым уровнем доступа
- 17) Дата отпуска сотрудника Фирова
- 18) Информация о расположении отдела бухгалтерии
- 19) Информация об отделах, в которых есть женщины
- 20) Информация о сотруднике, у которого книга «Стратегия голубого океана»
- 21) Информация о книгах одного автора
- 22) Информация о сотрудниках, имеющих навык подготовки презентаций
- 23) Номер СНИЛС сотрудника Захарченко
- 24) Информация о сотрудниках, прошедших курс «Технический английский язык»
- 25) Информация о сотрудниках, у которых планируется командировка в Москву
- 26) Информация о проектах, над которыми работает сотрудник Теплов
- 27) Информация о книгах из одной категории
- 28) Информация о выступлениях сотрудника Теплова
- 29) Информация о сотрудниках, посетивших курсы в январе
- 30) Информация о командировках сотрудников отдела проектов
- 31) Информация о сотрудниках, проживающих по адресу Кронверкский проспект, дом 4
- 32) Адреса офисов, находящихся в одном городе

- 33) Информация об отпуске сотрудника Иванова
- 34) Информация о сотрудниках, берущих неоплачиваемый отпуск
- 35) Информация о сотрудниках, закончивших получение образования в 2015 году
- 36) Информация о сотрудниках, работающих на одном этаже
- 37) Информация о сотруднике по номеру телефона
- 38) Информация о нахождении кабинета 25
- 39) Информация о проектах, над которым работает партнер Сколково
- 40) Информация о сотрудниках, занимающих одинаковую должность
- 41) Информация об оборудовании принтер сотрудника Иванов
- 42) Срок окончания сертификата сотрудника Иванова
- 43) Информация о сотрудниках со сроком окончания сертификата в мае 2018 года
- 44) Информация о сотрудниках, сертификаты которых имеют одинаковую дату окончания
- 45) Информация о сотрудниках отдела бухгалтерии
- 46) Информация о сотрудниках, работающих в кабинете 25
- 47) Информация о сотрудниках с запрещенным доступом к ИСУ
- 48) Информация о сотрудниках, одновременно идущих в отпуск
- 49) Информация о сотрудниках офиса Экстрополис
- 50) Информация о сотрудниках, которые выступали на SECR
- 51) Информация о работе сотрудника Иванова
- 52) Информация о доступах сотрудников отдела бухгалтерии
- 53) Информация о сотрудниках одного года выпуска
- 54) Сотрудники, у которых нет информации о документах
- 55) Информация о среднем возрасте сотрудников
- 56) Информация о среднем стаже сотрудников отдела бухгалтерии
- 57) Информация о средней зарплате сотрудников по отделам
- 58) Информация о командировках, назначенных в одинаковую компанию

- 59) Информация о зарплате сотрудников отдела бухгалтерии
- 60) Информация о самой маленькой зарплате отдела разработки
- 61) Информация о самой высокой зарплате среди женщин
- 62) Информация о доступных книгах
- 63) Информация о доступном оборудовании
- 64) Информация о докладах сотрудников, выступающих более одного
раза
- 65) Информация о количестве стран, в которых расположены офисы
компании
- 66) Информация об отделах компании
- 67) Информация о количестве курсов, которые посетил сотрудник
Иванов
- 68) Информация о количестве партнеров по проектам
- 69) Количество проектов, над которым закреплен сотрудник Иванов
- 70) Информация о сотруднике с минимальным стажем
- 71) Отдел, в котором работает сотрудник с самым высоким стажем
- 72) Самый молодой сотрудник
- 73) Информация о всех молодых сотрудниках
- 74) Информация о конференциях, у которых совпадает дата
проведения
- 75) Информация о сотрудниках, которые в следующем году выходят
на пенсию
- 76) Количество командировок сотрудника Иванов
- 77) Количество командировок в г.Москва
- 78) Количество сотрудников, получивших образование в ИТМО
- 79) Количество офисов компании
- 80) Количество оборудования, которое занято одним конкретным
сотрудником (Иванов)
- 81) Информация о сотрудниках, работающих в одном кабинете
- 82) Количество отпусков сотрудника Иванова

- 83) Количество сотрудников с высшим образованием
- 84) Количество партнеров компании, которые представляют сферу «Медицина»
- 85) Информация о количестве сотрудников, которые являются многодетными родителями
- 86) Информация о средней зарплате сотрудников, которые являются многодетными родителями
- 87) Информация о сотрудниках, работающих над одинаковым проектом
- 88) Количество сотрудников, которые были в отпуске в мае
- 89) Узнать средний возраст сотрудников, заработная плата которых больше 40000
- 90) Узнать средний стаж сотрудников, заработная плата которых больше 100000
- 91) Информация о сотрудниках, которые были в этом году в отпуске более одного раза
- 92) Средний возраст сотрудников, в семьях которых нет детей
- 93) Информация о проектах, закрепленных за одним и тем же партнером
- 94) Сколько в компании работает женщин
- 95) Информация о количестве сотрудников, прошедших курс «Технический английский язык»
- 96) Информация о сотрудниках, работающих с одинаковым партнером
- 97) Узнать отдел, сотрудники которого реже других посещают конференции
- 98) Информация о проекте, над которым работает самое большое число сотрудников
- 99) Информация о проекте, по которому нет сотрудничества с партнерами

- 100) Среднее число книг, которые читают сотрудники отделов
- 101) Узнать сотрудника, который посетил наименьшее число курсов
- 102) Узнать, у кого из сотрудников наибольшее число публичных выступлений
- 103) Информация о сотрудниках-выпускниках одного и того же вуза
- 104) Узнать категорию, в которой представлено наименьшее число книг
- 105) Среднее число сотрудников в офисе
- 106) Узнать проект, сумма зарплат сотрудников которого наибольшая
- 107) Узнать среднее количество навыков сотрудников одной должности
- 108) Узнать конференцию, с самой большой посещаемостью сотрудников
- 109) Узнать город, в котором проводится наибольшее число конференций
- 110) Информация о сотруднике, который работает с наибольшим числом партнеров
- 111) Информация о сотрудниках с общим партнером
- 112) Узнать должность, сотрудники которой обладают наименьшим средним количеством навыков
- 113) Найти сертификаты, выданные одинаковой компанией
- 114) Узнать, какой отдел занимает наибольшее число этажей в офисе
- 115) Узнать отдел, в котором меньше всего сотрудников

Приложение 4. Фрагменты исходного текста программы

```

const express = require('express');
const app = express();
const mysql = require('mysql');
const bodyParser = require('body-parser');
const request = require('request');

const TEST_DB = {
  host: 'localhost',
  user: 'root',
  password: 'mysqlpw',
  multipleStatements: true,
  charset: 'utf8',
};

const TEST_DB_NAME = 'test';
let connection;
semanticModel = {
  projectionTable: []
};

const patternsList = [
  {
    infoValue: 'Номер телефона',
    regexp: /^(8|\+7)[\ - ]?( \(?\d{3}\) )?[\ - ]?[\d\ - ]{7,10}$/
  },
  {
    infoValue: 'Адрес электронной почты',
    regexp: /(?:[a-z0-9!#$%&'*/+=?^_`{|}~-]+(?:\. [a-z0-9!#$%&'*/+=?^_`{|}~-]+)*)|"(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21\x23-\x5b\x5d-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f]) *")@(?: (?:[a-z0-9](?:[a-z0-9-]*[a-z0-9])?\.)+[a-z0-9](?:[a-z0-9-]*[a-z0-9])?|\[(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)|[a-z0-9-]*[a-z0-9]:(?:[\x01-\x08\x0b\x0c\x0e-\x1f\x21-\x5a\x53-\x7f]|\\[\x01-\x09\x0b\x0c\x0e-\x7f])+) \)]) /
  },
  {
    infoValue: 'ФИО - Фамилия Имя Отчество',
  }
];

```

```

    regexp: /([А-ЯЁ][а-яё]+[\-\s]?){3,}/
  },
  {
    infoValue: 'URL-адрес',
    regexp: /^(https?|ftp|file):\/\/)?([\da-z\.-
]+)\.([a-z\.]{2,6})([\/\w \.-]*)*\/?$/
  },
  {
    infoValue: 'Время в 24-часовом формате',
    regexp: /^([01]?[0-9]|2[0-3]):[0-5][0-9]$/
  },
  {
    infoValue: 'Дата в формате ДД/ММ/ГГГГ',
    regexp: /^(0?[1-9]|[12][0-9]|3[01])([\/\-])(0?[1-
9]|1[012])\2([0-9][0-9][0-9][0-9])(([ -])([0-1]?[0-
9]|2[0-3]):[0-5]?[0-9]:[0-5]?[0-9])?$/
  },
  {
    infoValue: 'IP-адрес',
    regexp: /^(?:(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-
9]?)\.){3}(?:25[0-5]|2[0-4][0-9]|[01]?[0-9][0-9]?)$/
  }
];

app.use(function (req, res, next) {
  res.setHeader('Access-Control-Allow-Origin',
'http://localhost:3000');
  res.setHeader('Access-Control-Allow-Methods', 'GET,
POST, OPTIONS, PUT, PATCH, DELETE');
  res.setHeader('Access-Control-Allow-Headers', 'X-
Requested-With,content-type');
  res.setHeader('Access-Control-Allow-Credentials',
true);
  next();
});

app.use(bodyParser.urlencoded({
  extended: true,
}));
app.use(bodyParser.json());

```

```

app.get('/', function (req, res) {
  res.send('It works');
});

const _makeDBWithData = (connection) => {
  return new Promise((resolve, reject) => {
    _createDB({ name: TEST_DB_NAME, connection
  }).then(() => {
    _changeDBUser({ database: TEST_DB_NAME,
connection }).then(() => {
    _createTables(connection)
      .then(() => {
        _createLinks(connection);
      })
      .then(() => {
        _insertDataToEmployee(connection);
      })
      .then(() => {
        _insertDataToDepartment(connection);
      })
      .then(() => {
        _insertDataToContacts(connection);
      })
      .then(() => {
        console.log('Data added to database
successfully');
        resolve();
      })
    });
  });
});

const _createDB = (params) => {
  const { name, connection } = params;

  return new Promise((resolve, reject) => {
    const sql = `CREATE DATABASE IF NOT EXISTS
${name}`;

```



```

connection.query(sql, (err, result) => {
  if (err) {
    return reject(err);
  }

  console.log('Database created');
  resolve();
});
});
};

const _changeDBUser = (params) => {
  const { database, connection } = params;

  return new Promise((resolve, reject) => {
    connection.changeUser({ database }, (err, result)
=> {
      if (err) {
        return reject(err);
      }

      console.log('User changed successfully');
      resolve(result);
    });
  });
};

const _createTables = (connection) => {
  const createEmployee = `
    create table if not exists employee(
      id int primary key auto_increment,
      name varchar(255)not null,
      sex varchar(255)not null,
      id_department int
    ) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
  `;

  const createDepartment = `
    create table if not exists department(
      id int primary key auto_increment,
      title varchar(255)not null
    ) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
  `;

```

```

`;
const createContacts = `
  create table if not exists contacts (
    id_employee int,
    phone varchar(255) not null,
    email varchar(255) not null
  ) DEFAULT CHARSET=utf8 COLLATE=utf8_unicode_ci
`;

const sql = [
  createEmployee,
  createDepartment,
  createContacts,
].join(';');

return _sendQueryToDB({ sql, connection });
};

const _createLinks = (connection) => {
  const linkEmployeeDepartment = `
    ALTER TABLE employee ADD CONSTRAINT
    employee_department FOREIGN KEY (id_department)
    REFERENCES department(id)
  `;

  const sql = [
    linkEmployeeDepartment
  ].join(';');

  return _sendQueryToDB({ sql, connection });
};

const _insertDataToEmployee = (connection) => {
  const sql = 'insert into employee (id, name, sex,
id_department) values ?';

  const values = [
    [501, 'Алексей Иннокентьевич Репнин', 'Мужской',
1001],
    [502, 'Маргарита Борисовна Одоевская', 'Женский',
1002],

```

```

    [503, 'Святослав Васильевич Грибоедов', 'Мужской',
1003],
    [504, 'Любовь Владимировна Ходкевич', 'Женский',
1004],
    [505, 'Сергей Олегович Извольский', 'Мужской',
1005],
    [506, 'Елизавета Святославовна Бутурлина',
'Женский', 1006],
    [507, 'Руслан Ростиславович Вревский', 'Мужской',
1007],
    [508, 'Лидия Леонидовна Олсуфьева', 'Женский',
1001],
    [509, 'Матвей Аркадьевич Орлов', 'Мужской', 1002],
    [510, 'Таисия Филипповна Волконская', 'Женский',
1003],
    [511, 'Ростислав Никитич Оболянинов', 'Мужской',
1004],
    [512, 'Анна Павловна Пушина', 'Женский', 1005],
    [513, 'Филипп Данилович Скавронский', 'Мужской',
1006],
    [514, 'Марья Владиславовна Толстая', 'Женский',
1007],
    [515, 'Валентина Святославовна Рюмина', 'Женский',
1001],
    ];
    return _sendQueryToDB({ sql, values, connection });
};

const _insertDataToDepartment = (connection) => {
    const sql = 'insert into department (id, title)
values ?';

    const values = [
        [1001, 'Отдел информационных технологий'],
        [1002, 'Административно-хозяйственный отдел'],
        [1003, 'Финансовый отдел'],
        [1004, 'Юридический отдел'],
        [1005, 'Отдел по работе с персоналом'],
        [1006, 'Департамент по безопасности'],
        [1007, 'Департамент по стратегическим
коммуникациям'],

```

```

];

return _sendQueryToDB({ sql, values, connection });
};

const _insertDataToContacts = (connection) => {
  const sql = 'insert into contacts (id_employee,
phone, email) values ?';

  const values = [
    [501, '+79111234567', 'it@company.com'],
    [502, '+78121234567', 'aho@company.com'],
    [503, '88121234567', 'money@company.com'],
    [504, '89131234567', 'legal@company.com'],
    [505, '+18121234567', 'hr@company.com'],
    [506, '88132341556', 'security@company.com'],
    [507, '+78142345678', 'communication@company.com'],
    [508, '+78153456789', 'helo@company.com'],
    [509, '88011234567', 'example@yahoo.com'],
    [510, '+78161234577', 'private@gmail.com'],
    [511, '+278182345067', 'public@gmail.com'],
    [512, '+78021234567', 'mymailaddress@mail.com'],
    [513, '+78031234567', 'donotreply@yandex.com'],
    [514, '+78921234567', 'mail@yahoo.com'],
    [515, '+78941234567', 'anothermail@yahoo.com'],
  ];

  return _sendQueryToDB({ sql, values, connection });
};

const _sendQueryToDB = (params) => {
  const { sql, values=[], connection } = params;

  return new Promise((resolve, reject) => {
    connection.query(sql, [values], (err, result) => {
      if (err) {
        return reject(err);
      }

      resolve(result);
    });
  });
};

```

```

};

const _getDBTables = (params) => {
  const { database, connection } = params;
  const sql = 'SHOW TABLES';
  return new Promise((resolve, reject) => {
    connection.connect((error) => {
      if (error) {
        reject(error);
      }

      _changeDBUser({ database, connection }).then(()
=> {
        _sendQueryToDB({ connection, sql
}).then((result) => {
          const field = `Tables_in_${dbName}`;
          const tablesList = result.map((item) =>
item[field]);
          resolve(tablesList);
        });
      });
    });
  });
};

const _getTableFields = (params) => {
  const { database, table, connection } = params;
  return new Promise((resolve, reject) => {
    const sql = `SHOW COLUMNS FROM ${table}`;

    _sendQueryToDB({ connection, sql }).then((result)
=> {
      const fieldsList = result.map((item) =>
item.Field);
      resolve(fieldsList);
    });
  });
};

const _getLinks = (params) => {
  const { connection } = params;

```

```

return new Promise((resolve, reject) => {
  semanticModel.links = {};
  const sql = `
    SELECT
      TABLE_NAME, COLUMN_NAME, CONSTRAINT_NAME,
    REFERENCED_TABLE_NAME, REFERENCED_COLUMN_NAME
    FROM
      INFORMATION_SCHEMA.KEY_COLUMN_USAGE
    WHERE
      REFERENCED_TABLE_SCHEMA = '${TEST_DB_NAME}'
  `;

  _sendQueryToDB({ connection, sql }).then((result)
=> {
    result.map((item) => {
      const currentTableData =
semanticModel.links[item['TABLE_NAME']] || {};
      semanticModel.links[item['TABLE_NAME']] =
Object.assign({}, currentTableData, {
        [item['REFERENCED_TABLE_NAME']]: {
          rootKey: item['COLUMN_NAME'],
          nestedKey: item['REFERENCED_COLUMN_NAME'],
          rootType: 'foreign',
        }
      });

      const referencedTableData =
semanticModel.links[item['REFERENCED_TABLE_NAME']] ||
{};

      semanticModel.links[item['REFERENCED_TABLE_NAME']] =
Object.assign({}, referencedTableData, {
        [item['TABLE_NAME']]: {
          rootKey: item['REFERENCED_COLUMN_NAME'],
          nestedKey: item['COLUMN_NAME'],
          rootType: 'primary',
        }
      });
    });
    resolve();
  });
});

```

```

    });
};

const _analyzeFieldsSemantics = (params) => {
  const { connection, table } = params;
  return new Promise((resolve, reject) => {
    const sql = `SELECT * FROM ${table} LIMIT 1`;

    _sendQueryToDB({ connection, sql }).then((result)
=> {
      const values = Object.values(result[0]);
      const fields = Object.keys(result[0]);

      values.map((value, index) => {
        if (fields[index] === 'id') return;

        patternsList.forEach((pattern) => {
          if (pattern.regexp.test(value)) {
            const fieldName = fields[index];
            const datalogicItem =
`${table}.${fieldName}`;
            semanticModel.projectionTable.push([
              nl: [],
              info: pattern.infoValue,
              data: datalogicItem
            ]);
          }
        })
      });
      resolve();
    });
  });
};

const _addNlTerms = (options) => {
  const { index, word } = options;

  const nlList =
semanticModel.projectionTable[index].nl;
  nlList.push(word);
  const synonyms = _getSynonymsFromThesaurus(word);

```

```

synonyms.forEach((synonym) => {
  nlList.push(synonym);
});
};

const _getSynonymsFromThesaurus = (word) => {
  return new Promise((resolve, reject) => {
    const requestOptions = {
      key: API_KEY,
      lang: 'ru-ru',
      text: word
    };
    request({
      url:
'https://dictionary.yandex.net/api/v1/dicservice.json/lookup',
      method: 'GET',
      json: true,
      body: requestOptions
    }, function (error, response, body) {
      const synonyms = [];
      const syn = response.def[0].tr.map((item) =>
item.syn);
      syn.forEach((item) => {
        item.forEach((innerItem) => {
          synonyms.push(innerItem.text);
        })
      })

      function onlyUnique(value, index, self) {
        return self.indexOf(value) === index;
      }

      const synonymsList = synonyms.filter(onlyUnique);

      resolve(synonymsList);
    });
  });
};

app.post('/connect-db', function (req, res) {

```



```

let credentials;

if (req.body.isTest) {
  credentials = TEST_DB;
}

connection = mysql.createConnection(credentials);
connection.connect((error) => {
  if (error) {
    return;
  }

  _makeDBWithData(connection).catch((error) => {
    res.status(500).send({ error });
  }).then(() => {
    res.send({ status: 'ok' });
  });
});
});

app.post('/create-semantic-model', function (req, res)
{
  console.log('/create-semantic-model');
  const sql = 'SHOW TABLES';
  let credentials;

  if (req.body.isTest) {
    credentials = TEST_DB;
    dbName = TEST_DB_NAME;
  }

  connection = mysql.createConnection(credentials);
  const promisesList = [];
  _getDBTables({ connection, database: dbName
  }).then((tablesList) => {
    semanticModel.tables = tablesList;

    promisesList.push(_getLinks({ connection }));

    tablesList.forEach((item) => {

```

```

        promisesList.push(_getTableFields({ connection,
database: dbName, table: item }));
        _analyzeFieldsSemantics({ connection, database:
dbName, table: item });
    });
    Promise.all(promisesList).then((result) => {
        semanticModel.structure = {};
        for (let i = 0; i < result.length; i++) {
            const key = semanticModel.tables[i];
            semanticModel.structure[key] = result[i+1];
        }

        res.setHeader('Content-Type',
'application/json');
        res.send(JSON.stringify(semanticModel));
    });
});

app.listen(4545, function () {
    console.log('App listening on port 4545');
});

```