

Исследование Text2Report систем

Оглавление

1. Эволюция инструментов отчётности	2
2. Обзор text2report систем	4
2.1 Область применения.....	4
2.2 Сценарий использования пользователем	4
2.3 Типовая функциональная архитектура text2report систем.....	5
2.3.1 Схема функциональной архитектуры.....	5
2.3.2 Компоненты системы:	6
2.3 Пример text2report системы от команды RS GlowByte.....	7
2.3.1 Функциональные возможности.....	7
2.3.2 Элементы интерфейса	7
2.3.2 Модуль дообучения.....	9
2.3.3 Схема функциональной архитектуры.....	9
2.3.4 Схема технологической архитектуры	9
2.3.5 RAG (Retrieval-Augmented Generation)	10
3. Основные проблемы text2sql систем и пути их решения	11
3.1 Плохое понимание запроса пользователя	11
3.2 Неверный синтаксис сгенерированного SQL запроса	12
3.3 Проблемы с визуализацией результатов на графиках.....	13
4. Потенциал для участия GlowByte при внедрении text2report систем	13
4.1 Сбор и структурирование метаданных	13
4.1.1 Сбор схемы данных в специфическом формате	14
4.1.2 Сбор словаря бизнес-терминов.....	14
4.1.3 Сбор примеров SQL запросов	14
4.2 Непрерывная поддержка и развитие системы.....	15
4.2.1 Поддержка артефактов в актуальном состоянии	15
4.2.2 Near-realtime сбор обратной связи от пользователей.....	15
4.2.3 Тюнинг системы	15
4.3 Дополнительные направления создания ценности	16
4.3.1 Управление изменениями и обучение пользователей	16
4.3.2 Интеграция с существующим ландшафтом BI.....	16
4.3.3 Обеспечение безопасности и соответствия требованиям.....	16
Приложения:	16
Приложение-1: Системные промпты демонстрационной системы.....	16

Приложение-2: API Endpoints демонстрационной системы.....	20
Приложение-3: Функциональная архитектура демонстрационной системы	21
Приложение-4: Технологическая архитектура демонстрационной системы	22

1. Эволюция инструментов отчётности

Кратко рассмотрим основные шаги эволюции корпоративных систем отчётности:

Подход	Преимущества	Недостатки
Простые статические отчёты <i>Crystal Reports, Jasper Reports, статические Excel-отчеты</i> <i>Регламентированная отчётность, периодические отчеты с фиксированной структурой, отчеты для регуляторов, стандартные финансовые отчеты</i>	<p>Простота реализации и использования</p> <p>Стандартизированный формат представления данных</p> <p>Низкие требования к техническим компетенциям пользователей</p>	<p>Отсутствие гибкости: любое изменение требует вмешательства ИТ-специалистов</p> <p>Ограниченные возможности анализа и интерпретации данных</p>
BI системы и дашборды <i>Tableau, Power BI, QlikView, IBM Cognos, SAP BusinessObjects</i> <i>Оперативная аналитика, мониторинг KPI, анализ трендов, сравнительная аналитика, периодическая отчетность</i>	<p>Интерактивность и фильтрация данных в режиме реального времени</p> <p>Визуализация данных в удобном для восприятия формате</p> <p>Возможность незначительных модификаций без привлечения ИТ-специалистов</p>	<p>Высокая стоимость лицензий и внедрения</p> <p>Необходимость обучения персонала</p> <p>Ограниченная гибкость: создание принципиально новых отчетов требует специальных навыков</p> <p>Сложность настройки и поддержки</p>

<p>Self-service аналитика</p> <p><i>Tableau Desktop, Power BI Desktop, Alteryx, TIBCO Spotfire, EXCEL</i></p> <p><i>Специализированная аналитика на уровне отделов, исследование данных для принятия тактических решений, ad-hoc запросы, оперативный анализ изменяющихся бизнес-показателей</i></p>	<p>Значительное сокращение времени получения аналитики</p> <p>Снижение нагрузки на ИТ-отделы</p> <p>Адаптация отчетов под конкретные потребности пользователя</p> <p>Демократизация доступа к данным в организации</p>	<p>Требует определенных технических навыков от бизнес-пользователей</p> <p>Риск несогласованности методологии анализа в разных подразделениях</p> <p>Проблемы с качеством и достоверностью данных при отсутствии централизованного контроля</p> <p>Вопросы безопасности и конфиденциальности данных</p>
<p>Text2Report системы (на основе text2sql)</p> <p><i>LangChain/LangGraph, Databricks, Chat2DB, SQL Chat, Vanna, DataGrip, Outerbase</i></p> <p><i>Оперативный анализ данных для широкого круга бизнес-пользователей, поисковые запросы по корпоративным данным, исследовательская аналитика без привязки к заранее созданным отчетам, динамический анализ на основе новых бизнес-вопросов</i></p>	<p>Демократизация доступа к данным, минимальный порог входа для пользователей</p> <p>Максимальная скорость получения данных и инсайтов</p> <p>Возможность задавать нестандартные вопросы без предварительной подготовки отчетов</p> <p>Интерактивный характер взаимодействия с данными</p> <p>Согласованная методика расчёта показателей благодаря словарю бизнес-терминов</p>	<p>Зависимость от качества реализации LLM цепочек и контекста модели</p> <p>Потенциальные неточности в интерпретации запросов</p> <p>Ограничения в сложности формулируемых запросов</p> <p>Высокие требования к качеству исходных данных и их описанию</p>

Важно понимать, что каждый последующий этап не отменяет предыдущие, а дополняет их. Так, в современных организациях могут сосуществовать все четыре типа инструментов, решая разные задачи:

- **Статичные отчеты** остаются актуальными для регламентированной отчетности и формализованных бизнес-процессов
- **BI-дашборды** обеспечивают мониторинг ключевых показателей и стратегический анализ
- **Self-service аналитика** используется для тактических решений и специализированных исследований
- **Text2Report системы** обеспечивают максимальную гибкость и доступность аналитики для всех уровней персонала

Выбор конкретного инструмента должен определяться бизнес-задачами, зрелостью данных в организации и компетенциями пользователей. При этом тенденция очевидна — развитие идет в сторону демократизации доступа к данным, упрощения интерфейсов и сокращения времени от возникновения вопроса до получения аналитического ответа.

2. Обзор text2report систем

2.1 Область применения

Text2report системы представляют собой новейший этап в эволюции инструментов бизнес-аналитики, который существенно расширяет возможности взаимодействия бизнес-пользователей с данными. Эти системы особенно эффективны в следующих областях:

- **Оперативная аналитика:** когда требуется быстро получить ответы на нестандартные вопросы без ожидания разработки специализированных отчетов.
- **Исследовательский анализ данных:** для формулирования и проверки гипотез в режиме диалога с системой.
- **Непрограммирующие аналитики:** позволяет бизнес-пользователям без знания SQL получать данные из хранилищ.
- **Сценарии с частой сменой аналитических вопросов:** когда традиционные фиксированные дашборды не успевают за потребностями бизнеса.
- **Демократизация доступа к данным:** обеспечение возможности самостоятельной работы с данными для широкого круга сотрудников.

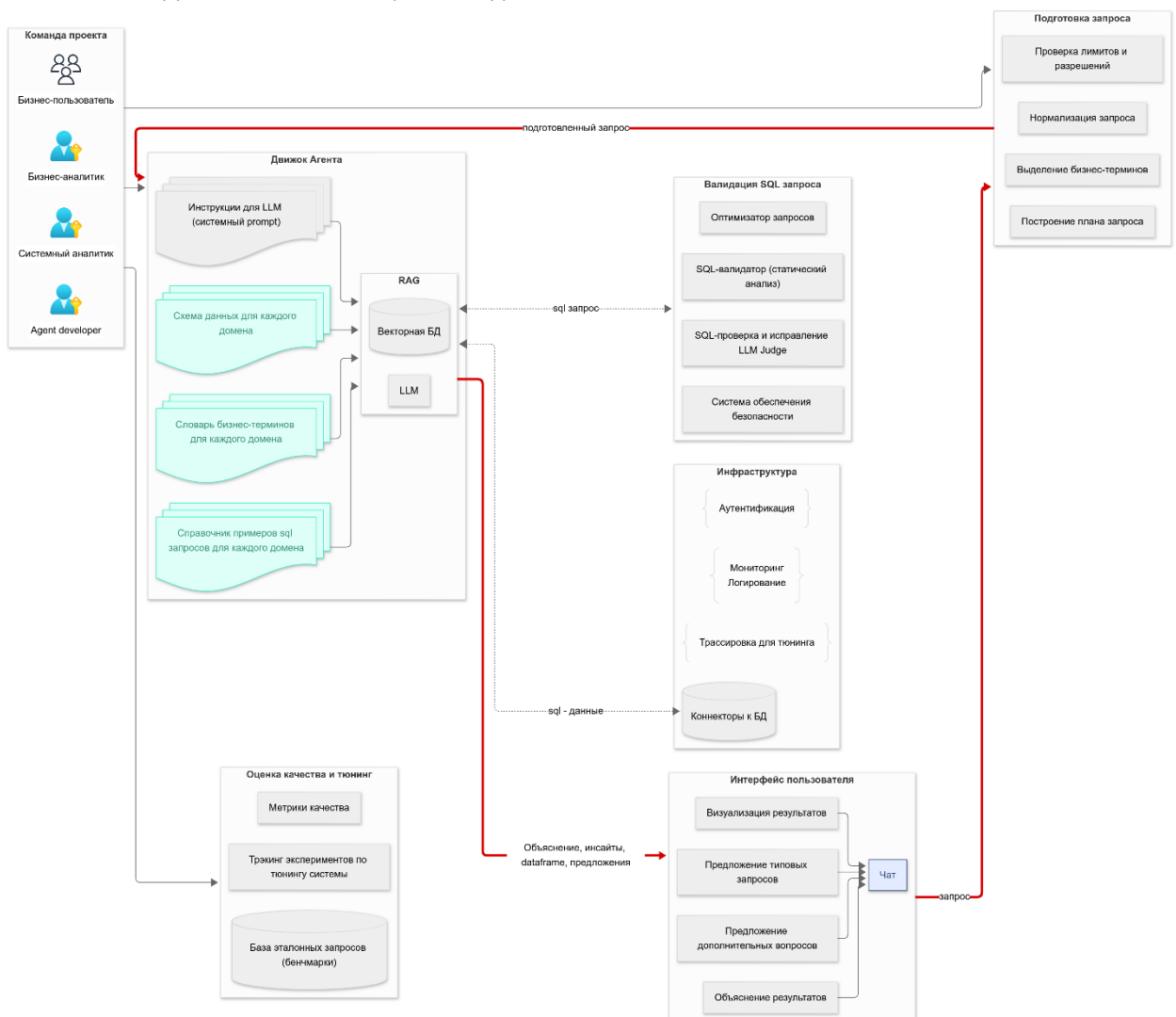
2.2 Сценарий использования пользователем

Типичный сценарий взаимодействия бизнес-пользователя с text2report системой выглядит следующим образом:



2.3 Типовая функциональная архитектура text2report систем

2.3.1 Схема функциональной архитектуры



2.3.2 Компоненты системы:

1. Интерфейс пользователя

- Обеспечивает взаимодействие пользователя с системой через чат
- Предлагает типовые запросы и дополнительные вопросы
- Отображает объяснения и визуализации результатов

2. Подготовка запроса

- Нормализует пользовательский запрос
- Выделяет бизнес-термины из запроса
- Проверяет лимиты и разрешения пользователя (предотвращение нецелевого использования LLM)
- Составляет план выполнения запроса

3. RAG

- Реализует механизм Retrieval Augmented Generation
- Включает векторную базу данных и языковую модель (LLM)
- Обогащает динамическую часть промпта к LLM релевантным контекстом

4. Движок Агента

- Преобразует естественный язык в SQL-запросы
- Содержит схемы данных, словари терминов и примеры запросов
- Формирует промпт для LLM (статичный и динамический компоненты)
- Является центральным компонентом системы

5. Валидация SQL запроса

- Выполняет статический анализ SQL-запросов
- Проверяет и исправляет запросы с помощью LLM Judge
- Оптимизирует запросы для более эффективного выполнения
- Контролирует доступ пользователей к данным в БД в соответствии с ролевой моделью

6. Инфраструктура

- Предоставляет коннекторы к различным базам данных
- Обеспечивает мониторинг и логирование системы
- Реализует трассировку для тюнинга системы
- Управляет аутентификацией пользователей

7. Оценка качества и тюнинг

- Отслеживает эксперименты по улучшению системы
- Поддерживает базу эталонных запросов для тестирования
- Измеряет и анализирует метрики качества работы системы

8. Команда проекта

- На этапе разработки и внедрения состоит из: бизнес-аналитика, системного аналитика, бизнес-пользователя и разработчика агента

2.3 Пример text2report системы от команды RS GlowByte

[Ссылка](#)

Область применения

- Пресейл
- Изучение text2report систем

2.3.1 Функциональные возможности

Рассматриваемый пример text2report системы, реализованный на базе специализированного opensource фреймворка Vanna AI, предоставляет следующие возможности:

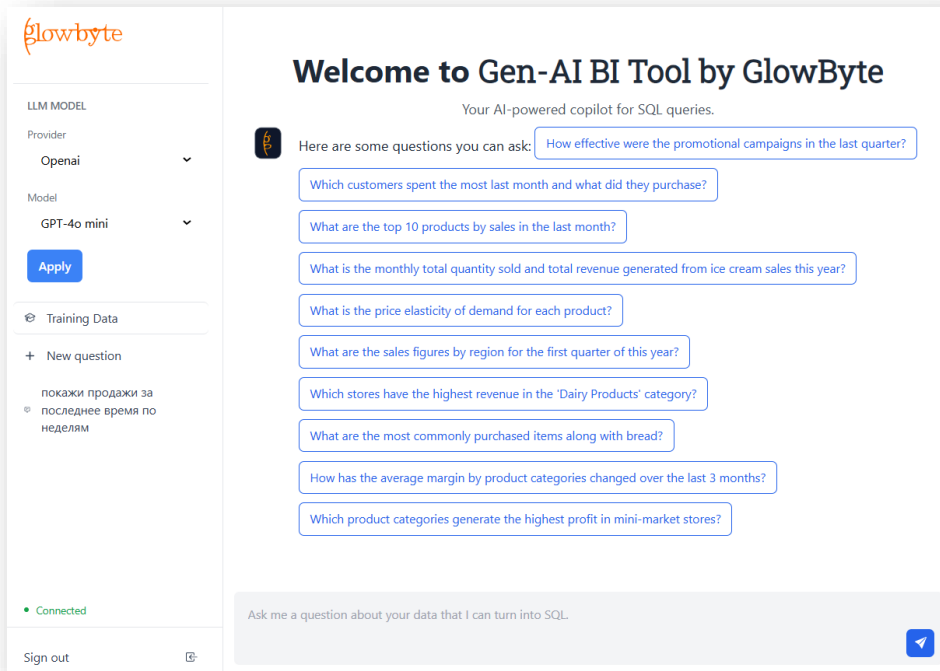
1. **Естественно-языковой интерфейс:** преобразование запросов на обычном языке в SQL-запросы, их исполнение и представление результатов в табличном виде.
2. **Интерактивные визуализации:** автоматическое создание графиков на основе результатов запросов.
3. **Контекстная осведомленность:** система "помнит" предыдущие запросы и использует их, если пользователь продолжает исследования в чате.
4. **Самообучение:** механизм улучшения качества ответов на основе обратной связи от пользователей.
5. **Многоязычность:** поддержка запросов на русском языке с правильной интерпретацией бизнес-терминологии.
6. **Многомодельность:** возможность выбора различных LLM для оптимальной работы системы.
7. **Гибкая настройка под предметную область:** механизмы обучения системы на специфике конкретного бизнеса.

2.3.2 Элементы интерфейса

2.3.2.1 Чат

Центральным элементом интерфейса является чат, который включает:

1. **Предложение типовых запросов:** система предлагает пользователю набор стартовых вопросов, основанных на структуре данных и типовых бизнес-задачах.
2. **Окно с запросом пользователя:** поле для ввода вопросов на естественном языке.
3. **Окно с SQL кодом:** опциональный элемент, который позволяет увидеть сгенерированный SQL-запрос, что полезно для отладки и обучения пользователей.
4. **Таблица с результатами:** структурированное представление данных, полученных в результате выполнения запроса.
5. **Сохранение результатов:** возможность сохранить таблицу с результатами в CSV файл.
6. **Визуализация результатов:** автоматически сгенерированные графики, использующие библиотеку Plotly для наглядного представления результатов анализа.
7. **Механизм обратной связи:** возможность указать, корректны ли результаты, и предложить правильный SQL-запрос для дообучения системы.
8. **Следующие вопросы:** система предлагает логически связанные вопросы для углубления анализа.



You: What are the top 10 products by sales in the last month?

AI:

```
SELECT p.product_name, SUM(s.quantity) AS total_quantity, SUM(s.total_revenue) AS total_revenue
FROM sales s
JOIN products p ON s.product_id = p.product_id
WHERE s.sale_date >= DATE_TRUNC('month', CURRENT_DATE) - INTERVAL '1 month'
AND s.sale_date < DATE_TRUNC('month', CURRENT_DATE)
GROUP BY p.product_name
ORDER BY total_quantity DESC
LIMIT 10;
```

PRODUCT_NAME	TOTAL_QUANTITY	TOTAL_REVENUE
Булочка с маком	10	3028
Блинчики с мясом	9	2180.700012207
Сыр Российский	8	711.840637207
Творог 9%	8	3063.0223999023
Молоко 2,5%	7	2776.9932861328
Филе куриное	7	2737.6628417969
Молоко 3,2%	5	1944.6999511719
Фарш говяжий	5	2221.6499023438
Колбаса вареная	5	256.3999938965
Чай черный	5	987

CSV Download

Top 10 Products by Sales in the Last Month

Were the results correct?

Yes No

Here are some followup questions you can ask:

- What are the top 10 products by total revenue in the last month?
- How many units of each product were sold in the last week?
- What are the top 5 products by sales quantity in the last month?
- What is the total revenue generated by each product in the last month?
- Which products had sales of more than 5 units in the last month?

2.3.2.2 Боковая панель

Боковая панель предоставляет доступ к настройкам и дополнительной функциональности:

- Выбор LLM:** возможность переключения между различными языковыми моделями (OpenAI, Anthropic, Mistral) и их вариациями для оптимизации качества и скорости ответов.

- **Новый запрос:** кнопка для начала нового диалога.
- **История запросов:** список ранее заданных вопросов для быстрого возврата к предыдущим результатам.
- **Модуль обучения:** переход в окно настройки системы на специфику конкретного бизнеса.

2.3.2 Модуль дообучения

Набор инструментов, позволяющий значительно улучшить работу системы за счёт предоставления дополнительного специфичного контекста:

- **DDL:** определение схемы данных в виде SQL-операторов CREATE TABLE и связей между таблицами.
- **Documentation:** словарь бизнес-терминов с указанием соответствующих таблиц, полей и формул расчета.
- **SQL:** примеры SQL-запросов и соответствующих текстовых вопросов, что критически важно для обучения системы генерации сложных запросов.

2.3.3 Схема функциональной архитектуры

Функциональная архитектура системы включает следующие основные блоки:

1. **Интерфейсный слой:** веб-интерфейс для взаимодействия с пользователем, реализованный с использованием современных веб-технологий.
2. **Аналитический движок:**
 - 2.1. Модуль обработки естественного языка
 - 2.2. Модуль генерации SQL
 - 2.3. Модуль выполнения запросов
 - 2.4. Модуль визуализации данных
3. **Слой обучения и настройки:**
 - 3.1. Модуль управления схемой данных
 - 3.2. Модуль управления бизнес-словарем
 - 3.3. Модуль управления примерами запросов
 - 3.4. Статичная часть промпта (системный промпт). *Полный текст системного промпта смотри в [Приложении-1](#).*
4. **Слой хранения:**
 - 4.1. База данных для хранения аналитических данных
 - 4.2. Векторное хранилище для RAG-компонента
 - 4.3. Кэш для оптимизации производительности

Схему функциональной архитектуры смотри в [Приложении-3](#)

2.3.4 Схема технологической архитектуры

С технической точки зрения система построена на основе следующего стека:

1. **Backend:** Flask (Python) для создания API-интерфейсов и бизнес-логики. *Полный список API endpoints смотри в [Приложении-2](#).*
2. **Frontend:** HTML, CSS, JavaScript.
3. **База данных:** DuckDB для быстрой аналитической обработки данных.
4. **Визуализация:** Plotly для создания интерактивных графиков.
5. **AI-компоненты:**

- 5.1. Языковые модели: поддержка OpenAI, Anthropic, Mistral
- 5.2. Embeddings модели: по-умолчанию из фреймворка Vanna.ai используется модель ll-MiniLM-L6-v2, но существует (и будет реализовано в будущем) возможность перепоределения embeddings моделей на другие, например OpenAI Embeddings API
- 5.3. Векторная база данных: ChromaDB для реализации RAG
- 5.4. Vanna AI как фреймворк для интеграции компонентов text2sql
6. [Исходники](#) проекта на GitHub (является форком проекта [Vanna-flask](#))

Схему технологической архитектуры смотри в [Приложении-4](#)

2.3.5 RAG (Retrieval-Augmented Generation)

2.3.5.1 Особенности реализации RAG

В системе реализован продвинутый подход к RAG, который выходит за рамки наивного решения:

- **Проблемы наивного RAG:** При наивном подходе весь обучающий датасет помещается в промпт вместе с запросом пользователя, что приводит к:
 - Понижению внимания LLM при относительно большом контексте
 - Низкой релевантности большей части контекста (потеря внимания LLM)
 - Повышенной стоимости запросов к API
 - Замедлению работы системы
- **Решение в системе:**
 - Использование векторных представлений для определения семантической близости
 - Выборка только релевантных фрагментов схемы данных, документации и примеров запросов
 - Динамическое формирование контекста для каждого запроса пользователя

2.3.5.2 Параметризация LLM

Система позволяет тонко настраивать работу языковых моделей:

Выбор модели: Поддерживаются современные топовые модели:

- OpenAI: GPT-4o, GPT-4o mini, o1-preview, o3-mini
- Anthropic: Claude 3.7 Sonnet. ★ В настоящий момент (апрель 2025г) показывает значительно-лучшие результаты в задаче text2sql с запросами на русском языке!
- Mistral: Codestral

Параметры генерации:

- Температура: контроль креативности vs. детерминированности ответов
- Максимальное количество токенов: ограничение объема генерируемого текста
- Системный промпт: детальные инструкции для модели по генерации качественного SQL

Специализированные промпты: система использует отдельные специализированные промпты для:

- Генерации SQL-запросов
- Создания визуализаций
- Формирования начальных и дополнительных вопросов

Примеры используемых промптов смотри в Приложении-1

2.3.5.3 Векторная база данных

Для эффективной работы RAG используется ChromaDB как векторная база данных. Это обеспечивает:

- Быструю семантическую индексацию обучающих данных
- Эффективный поиск релевантных фрагментов по вопросу пользователя
- Инкрементальное обновление базы знаний без переобучения всей системы

3. Основные проблемы text2sql систем и пути их решения

Несмотря на прогресс языковых моделей, внедрение text2sql систем в корпоративной среде сталкивается с рядом сложностей. Рассмотрим ключевые проблемы и эффективные методы их преодоления.

3.1 Плохое понимание запроса пользователя

Одной из фундаментальных проблем является корректная интерпретация LLM запроса пользователя на естественном языке в контексте конкретной предметной области и структуры данных компании.

Решения:

1. Точное описание схемы данных

Ключевым фактором является предоставление LLM полной и структурированной информации о схеме данных:

- Детальные DDL-описания всех таблиц с комментариями к полям
- Явное указание первичных и внешних ключей для понимания связей между таблицами
- Метаданные о типах данных и ограничениях
- Подробное текстовое описание полей

2. Подробный и полный словарь бизнес-терминов

Критически важным компонентом является создание и поддержка словаря бизнес-терминологии:

- Ясные определения бизнес-понятий с привязкой к конкретным таблицам и полям
- Точные формулы расчета для производных показателей
- Синонимы терминов, используемые в организации
- Иерархические отношения между понятиями для понимания их взаимосвязей
- Документирование бизнес-логики и ограничений
- Расшифровка бизнес-сленга компании \ отдела \ группы пользователей

3. База примеров SQL запросов

Эффективная text2sql система опирается на обширную базу примеров, особенно в части корректного построения сложных SQL запросов:

- Пары "вопрос на естественном языке - соответствующий SQL-запрос"
- Примеры сложных аналитических запросов, характерных для конкретного домена бизнеса
- Механизм постоянного пополнения через обратную связь от пользователей

4. Промежуточный план запроса

Внедрение дополнительного этапа планирования запроса повышает качество text2sql:

- Разбиение сложных запросов на логические шаги
- Формирование промежуточного плана перед генерацией SQL
- Возможность валидации плана человеком-экспертом или второй LLM
- Проверка соответствия плана бизнес-логике и структуре данных

5. Создание специализированного слоя данных

Хорошей практикой является создание отдельного специализированного слоя данных:

- Для каждого департамента и/или домена данных создать нормализованные витрины или view
- Словарь бизнес-терминов и справочник типовых sql-запросов привести в соответствие со специализированным слоем данных

3.2 Неверный синтаксис сгенерированного SQL запроса

Даже при правильном понимании запроса, LLM может генерировать SQL с синтаксическими ошибками или несовместимый с конкретным диалектом SQL используемой СУБД.

Решения:

1. Настройка системного промпта

Тщательная формулировка инструкций для LLM значительно улучшает качество генерации:

- Явное указание используемого диалекта SQL (DuckDB, PostgreSQL, T-SQL и т.д.)
- Перечисление специфических функций и синтаксических конструкций
- Список типичных ошибок с указанием правильных альтернатив
- Примеры корректного форматирования запросов

2. Анализ и предотвращение типовых ошибок

Систематический подход к устранению повторяющихся ошибок:

- Сбор и классификация типичных ошибок генерации
- Создание специфических инструкций в системном промпте для их предотвращения
- Регулярное обновление промпта на основе новых выявленных ошибок
- Создание библиотеки типовых паттернов для сложных запросов

3. Внедрение цикла синтаксической проверки

Автоматизированная проверка и исправление запросов:

- Интеграция SQL-линтеров для проверки синтаксиса перед выполнением
- Система автоматического исправления распространенных ошибок
- Валидация запроса против схемы базы данных

- Циклический процесс "генерация-проверка-исправление"

4. Fine-tuning языковых моделей

Возможно применение для обучения синтаксису диалекта SQL:

- Сбор датасета корректных запросов для конкретного SQL-диалекта
- Не для всех топовых LLM возможен fine-tuning

3.3 Проблемы с визуализацией результатов на графиках

Автоматическая визуализация результатов SQL-запросов обычно сталкивается с проблемами выбора подходящего типа графика и правильной подготовки данных.

Решения:

1. База примеров трансформаций датасетов для визуализации

Накопление опыта успешных визуализаций:

- Примеры преобразования данных для различных типов графиков
- Шаблоны предобработки данных для визуализации временных рядов, категориальных данных и др.
- Документирование лучших практик по типам бизнес-задач
- Связь между типами вопросов и оптимальными способами визуализации

2. Специализированный промпт для визуализации

Отдельная настройка инструкций для генерации визуализаций:

- Детальные рекомендации по выбору типа графика в зависимости от структуры данных
- Указания по форматированию (подписи осей, легенды, цветовые схемы)
- Правила трансформации данных для улучшения восприятия
- Специфические настройки для разных типов аналитических задач

3. Интерактивные корректировки

Предоставление пользователю возможности влиять на визуализацию:

- Интерфейс для выбора альтернативных типов графиков
- Управление параметрами визуализации (группировка, сортировка, фильтрация)
- Сохранение предпочтений пользователя для будущих запросов
- Обратная связь для улучшения автоматической визуализации

4. Потенциал для участия GlowByte при внедрении text2report систем

Внедрение text2report систем требует экспертизы в технологиях и бизнес-процессах. ИТ-интеграторы, специализирующиеся на хранилищах данных и бизнес-аналитике, например GlowByte, обладают необходимыми компетенциями для реализации таких проектов.

4.1 Сбор и структурирование метаданных

4.1.1 Сбор схемы данных в специфическом формате

Одним из фундаментальных этапов внедрения text2report систем является подготовка детальной схемы данных в формате, оптимальном для использования языковыми моделями:

- **Системные аналитики GlowByte**, обладающие глубоким пониманием архитектуры корпоративных хранилищ данных, могут:
 - Проводить аудит существующих структур данных и выявлять наиболее значимые для аналитических задач таблицы и связи
 - Оптимизировать представление схемы данных, фокусируясь на бизнес-значимых сущностях для каждого департамента и домена знаний
 - Разрабатывать DDL-описания с подробными комментариями, облегчающими понимание назначения и смысла каждого поля
 - Документировать взаимосвязи между таблицами с акцентом на бизнес-логику этих связей
 - Создавать специализированный слой данных и\или представления (views), упрощающие сложную структуру данных для конкретных департаментов и\или доменов
- **Преимущество GlowByte:** многолетний опыт работы с различными отраслевыми моделями данных позволяет быстро адаптировать подходы под специфику конкретной индустрии.

4.1.2 Сбор словаря бизнес-терминов

Создание качественного бизнес-гlossария требует тесного взаимодействия технических специалистов и экспертов предметной области:

- **Совместная работа бизнес-аналитиков и системных аналитиков GlowByte** позволяет:
 - Организовать серию структурированных интервью с предметными экспертами для выявления ключевой терминологии
 - Формализовать бизнес-термины, в том числе локальный бизнес-жаргон, с точной привязкой к таблицам и полям хранилища данных
 - Документировать формулы расчета показателей с учетом особенностей бизнес-логики
 - Выявлять и фиксировать синонимы терминов, используемые в разных подразделениях
 - Создавать иерархическую структуру понятий, отражающую взаимосвязи между бизнес-концепциями
- **Методология GlowByte:** структурированный подход к сбору и валидации бизнес-гlossария, включающий итеративные циклы уточнения с участием бизнес-пользователей.

4.1.3 Сбор примеров SQL запросов

Формирование качественной базы примеров запросов требует глубокой аналитической экспертизы:

- **Команда бизнес- и системных аналитиков GlowByte** может обеспечить:
 - Анализ исторических запросов к хранилищу данных для выявления типовых паттернов
 - Разработку эталонных SQL-запросов для наиболее востребованных бизнес-сценариев

- Сопоставление запросов с их описаниями на естественном языке в разных формулировках
 - Документирование сложных аналитических кейсов с подробными комментариями
- **Уникальная ценность GlowByte:** сочетание технической экспертизы в SQL с глубоким пониманием бизнес-потребностей в аналитике для конкретных отраслей.

4.2 Непрерывная поддержка и развитие системы

4.2.1 Поддержка артефактов в актуальном состоянии

Поддержание компонентов text2report системы в синхронизации с эволюционирующей бизнес-средой:

- **GlowByte может создать процессы непрерывного обновления:**
 - Автоматизированное отслеживание изменений в структуре хранилища данных
 - Периодические сессии ревизии бизнес-гlossария с предметными экспертами
 - Мониторинг новых аналитических потребностей бизнеса для пополнения базы примеров
 - Интеграция с процессами управления изменениями в ИТ-ландшафте организации
 - Разработка регламентов и процедур обновления различных компонентов системы
- **Проактивный подход GlowByte:** не только реагирование на изменения, но и прогнозирование будущих потребностей на основе отраслевой экспертизы.

4.2.2 Near-realtime сбор обратной связи от пользователей

Организация эффективного механизма пополнения базы знаний на основе реального использования системы и обратной связи от пользователей (в том числе через веб-интерфейс text2report системы):

GlowByte может обеспечить:

- Поддержка по обновлению базы примеров SQL запросов
 - Сбор, валидация и приоритизация обращений от пользователей
 - Разработка и тестирование SQL запросов под текстовые запросы пользователей
 - Добавление данных в базу знаний text2report системы

4.2.3 Тюнинг системы

Для того, чтобы система text2report постоянно находилась в актуальном состоянии, необходимо создавать бенчмарки и отслеживать метрики качества работы системы в целом, и её отдельных компонентов.

GlowByte может принять участие в:

- Создании сета эталонных sql-запросов для тестирования качества системы
- Разработке метрик качества работы системы и компонентов

4.3 Дополнительные направления создания ценности

4.3.1 Управление изменениями и обучение пользователей

Успешное внедрение новых аналитических инструментов предполагает шаги адаптации организации:

- **Команда GlowByte может предложить:**
 - Разработку стратегии внедрения text2report систем с учетом специфики корпоративной культуры
 - Создание программ обучения для различных категорий пользователей
 - Формирование сообщества внутренних чемпионов технологии
 - Разработку метрик эффективности использования и ROI
 - Сопровождение пользователей на начальных этапах для быстрого достижения успеха
- **Опыт GlowByte:** практические методики преодоления организационного сопротивления при внедрении инновационных аналитических инструментов.

4.3.2 Интеграция с существующим ландшафтом BI

Обеспечение гармоничного сосуществования text2report систем с традиционной инфраструктурой бизнес-аналитики:

- **Эксперты GlowByte могут разработать:**
 - Архитектуру интеграции с существующими BI-платформами
 - Единую модель безопасности и управления доступом
 - Механизмы совместного использования метаданных и результатов анализа
 - Стратегию миграции от традиционных отчетов к гибриднему подходу
 - Дорожную карту эволюции аналитических инструментов организации
- **Ценность подхода GlowByte:** фокус на прагматичные решения, позволяющие получать быстрые результаты при минимальном нарушении существующих процессов.

4.3.3 Обеспечение безопасности и соответствия требованиям

Внедрение систем на основе LLM требует особого внимания к вопросам информационной безопасности и комплаенса:

- **GlowByte может обеспечить:**
 - Оценку рисков использования LLM в корпоративной среде
 - Разработку архитектуры с учетом требований к обработке конфиденциальной информации
 - Консультации по соответствию регуляторным требованиям отрасли
 - Внедрение механизмов аудита и контроля использования системы
 - Настройку политик предотвращения утечки чувствительных данных
- **Экспертиза GlowByte:** глубокое понимание нормативных требований в различных отраслях и практический опыт их реализации в информационных системах.

Приложения:

Приложение-1: Системные промпты демонстрационной системы

Системный промпт для генерации SQL

SQL_SYSTEM_PROMPT = ""The user provides a question and you provide SQL. You will only respond with SQL code and not with any explanations.

Respond with pure SQL code only. Do not add backticks, do not use markdown syntax like ``sql. Do not include any explanations, headers, or formatting. Return ONLY the raw SQL query that can be directly executed by a database engine.

1. Используй только SQL синтаксис, совместимый с DuckDB
2. При запросах временных рядов, упорядочивай данные по времени
3. Всегда учитывай оптимизацию запросов
4. Если пользователь не указал конкретный период времени, используй последние данные
5. Если в запросе используются расчётные или агрегатные показатели, выведи также поля, которые используются в этих показателях
6. Используй только таблицы и поля, определенные в схеме
7. Возвращай ТОЛЬКО SQL запрос и ничего больше
8. КРИТИЧЕСКИ ВАЖНО! Для каждой таблицы и CTE используй понятные и уникальные псевдонимы:
 - Используй полные имена для псевдонимов таблиц (например, "sales AS sales" или "products AS products")
 - Если используешь короткие псевдонимы, всегда следи за их консистентностью во всем запросе
 - НИКОГДА не используй одинаковый псевдоним для разных таблиц или CTE
 - Если определил таблицу как "sales AS sales_data", то ВСЕГДА обращай к ее полям как "sales_data.field", а НЕ "s.field"
9. КРИТИЧЕСКИ ВАЖНО! *ВСЕГДА проверяй, что каждый столбец в SELECT, WHERE, GROUP BY, ORDER BY существует в соответствующей таблице с указанным псевдонимом
10. При использовании CTE (Common Table Expressions), убедись, что ты обращаешься только к столбцам, которые определены в этой CTE
11. Избегай двусмысленности при выборе столбцов с одинаковыми именами из разных таблиц - всегда указывай таблицу/псевдоним

ВАЖНО! *Если в запросе используются расчётные или агрегатные показатели, выведи также поля, которые используются в этих показателях*

Правила форматирования результатов в SQL запросе:

1. Для числовых полей с единицами измерения (например, "₽", "кв.м.") включай единицы измерения в заголовок столбца, а не в сами ячейки
2. ОБЯЗАТЕЛЬНО используй функцию ROUND() для всех числовых значений:
 - Для валютных значений (формат "currency") и FLOAT числовых значений всегда округляй до 2 знаков: ROUND(sum_column, 2)
 - Пример: SELECT ROUND(SUM(total_amount), 2) AS "Выручка, ₽" ...
3. !ВАЖНО!Для дат (даже если они обернуты в date_trunc()) используй соответствующие форматы, через функцию FORMAT или strftime:
 - Пример: strftime(sale_date, '%d.%m.%Y') AS "Дата продажи"
 - Пример: strftime(date_trunc('month', s.sale_date), '%Y-%m') as month
4. Для агрегатных функций и вычисляемых полей используй осмысленные названия:
 - Примеры: "Среднее значение", "Сумма продаж", "Прибыль, ₽"
5. Для вычисляемых полей с единицами измерения также включай единицы измерения в заголовок и округляй значения:
 - Пример: ROUND(SUM((s.unit_price - p.unit_cost) * s.quantity), 2) AS "Прибыль, ₽"

""

Системный промпт для генерации графиков

PLOTLY_SYSTEM_PROMPT = """You are an expert at generating Python code that uses Plotly to visualize data based on SQL results.

You will be given a question, SQL query, and information about the dataframe.

VISUALIZATION GUIDELINES:

1. SELECT THE BEST CHART TYPE:

- Bar charts: For comparing categories or discrete values
- Line charts: For time series or showing trends over a continuous variable
- Pie/Donut charts: Only for representing parts of a whole (limit to 5-7 segments max)
- Scatter plots: For showing correlation between two variables
- Heatmaps: For displaying relationships between three variables or matrices
- Bubble charts: When you need to show three dimensions of data
- Area charts: For cumulative totals over time
- Box plots: For distribution and outlier analysis

2. ENSURE CLARITY AND READABILITY:

- Use clear, descriptive title that answers the user's question
- Add appropriate axis labels with units where applicable
- Include a legend when multiple data series are present
- Sort data to highlight patterns (e.g., sort bars by value, not alphabetically)
- Limit the number of series on one chart (5-7 max)
- For text-heavy charts, rotate labels if needed to prevent overlap

3. COLOR USAGE:

- Use categorical color schemes for different categories
- Use sequential color schemes for numeric data
- Use contrasting colors for emphasis
- Make sure color choices work for color-blind users

4. FORMATTING AND ANNOTATIONS:

- Add hover tooltips with detailed information
- Round numbers appropriately (2 decimal places usually sufficient)
- Format large numbers with K, M, B suffixes
- Format percentages correctly
- Add annotations for important points
- Use grid lines sparingly

5. DATA TRANSFORMATION:

- Consider aggregating data if there are too many data points
- Calculate percentages or ratios when appropriate
- For time series, ensure proper date formatting
- Apply log scale for data with wide ranges
- Add trendlines when analyzing patterns

Only respond with executable Python code that creates a suitable Plotly figure. Do not include explanations, markdown formatting, or any text outside of the Python code.

"""

Системный промпт для генерации дополнительных вопросов

```
FOLLOWUP_QUESTIONS_SYSTEM_PROMPT = """You are an expert at generating follow-up questions based on SQL query results.
```

```
Given a question, SQL query, and dataframe, suggest 3-5 follow-up questions that would be logical next steps for analysis.
```

```
"""
```

```
# Системный промпт для генерации начальных вопросов
```

```
QUESTIONS_SYSTEM_PROMPT = """You are an expert at suggesting useful initial questions for SQL database exploration.
```

```
Based on general database knowledge, suggest 5-8 questions that would be good starting points for analysis.
```

```
"""
```

Приложение-2: API Endpoints демонстрационной системы

1. Генерация и выполнение запросов:

- 1.1. /api/v0/generate_questions (GET) - генерирует начальные вопросы для пользователя
- 1.2. /api/v0/generate_sql (GET) - преобразует текстовый запрос пользователя в SQL
- 1.3. /api/v0/run_sql (GET) - выполняет сгенерированный SQL-запрос
- 1.4. /api/v0/generate_plotly_figure (GET) - создает визуализацию на основе результатов запроса
- 1.5. /api/v0/generate_followup_questions (GET) - предлагает следующие вопросы на основе текущего запроса

2. Работа с результатами:

- 2.1. /api/v0/download_csv (GET) - позволяет скачать результаты запроса в формате CSV
- 2.2. /api/v0/load_question (GET) - загружает данные ранее выполненного запроса

3. Управление историей:

- 3.1. /api/v0/get_question_history (GET) - получает историю заданных вопросов

4. Управление обучением системы:

- 4.1. /api/v0/get_training_data (GET) - получает список обучающих данных
- 4.2. /api/v0/train (POST) - добавляет новые обучающие данные
- 4.3. /api/v0/remove_training_data (POST) - удаляет обучающие данные

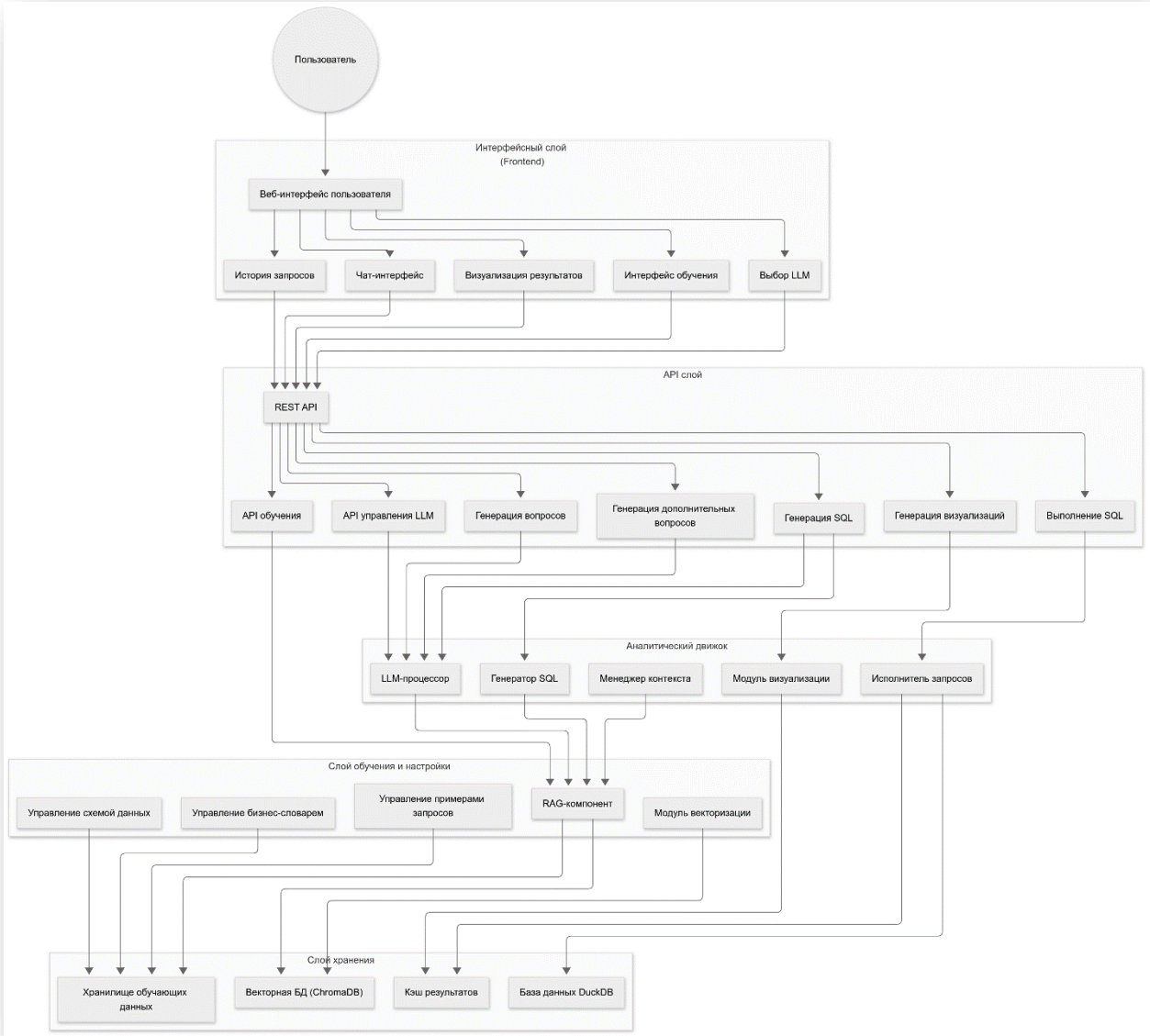
5. Управление LLM моделями:

- 5.1. /api/v0/get_available_models (GET) - получает список доступных языковых моделей
- 5.2. /api/v0/change_model (POST) - изменяет используемую языковую модель

6. Корневой маршрут:

- 6.1. / (GET) - возвращает главную страницу приложения

Приложение-3: Функциональная архитектура демонстрационной системы



Приложение-4: Технологическая архитектура демонстрационной системы

