МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра «Прикладная математика и информатика»
(Наименование учебного структурного подразделения)

09.04.03 Прикладная информатика
(код и наименование направления подготовки, специальности)

Технология бизнес-анализа
(направленность (профиль) / специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (МАГИСТЕРСКАЯ ДИССЕРТАЦИЯ)

на тему Моделирование системы анализа данных на основе технологии OLAP с помощью программного обеспечения с открытым исходным кодом

Обучающийся Д.О. Щербакова
(Инициалы Фамилия) (личная подпись)

Научный руководитель канд. техн. наук, доцент, О.В. Аникина
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Содержание

Введение

В результате наложенных на Российскую Федерацию санкций некоторые крупные иностранные разработчики программного обеспечения и IT-корпорации теперь недоступны на территории страны. Среди них Tableau, Qlik, Microsoft. Большое количество отечественных предприятий производило аналитику данных на базе решений этих компаний, предоставляющих наиболее популярные решения в области анализа и визуализации данных. В настоящее время многим отечественным предприятиям приходится искать новые сервисы для своей инфраструктуры и перестраивать бизнес-процессы.

Сегодня компании накапливают большое количество данных. Эти данные необходимо анализировать, чтобы не отставать от конкурентов в эффективности менеджерских решений. Анализ данных позволяет принимать эффективные решения, так как они основываются на информации, полученной в прошлом. Это позволяет уменьшить непредсказуемость в будущем.

Однако накапливание большого количества данных приводит к тому, что для их обработки необходимо задействовать больше ресурсов. Для того, чтобы польза от накапливания данных превышала издержки на их обслуживание, были разработаны информационные аналитические системы, которые получили название «Системы поддержки принятия решений». В архитектуру системы принятия решений в общем случае входят транзакционная база данных, «подсистема хранения и подсистема анализа» [2, с. 16]. Система анализа в классической системе поддержки принятия решений строится на базе технологии многомерного анализа данных OLAP (Online Analytical Processing).

На протяжении нескольких последних лет в области аналитической обработки данных наблюдается тенденция рассматривать технологию OLAP как устаревшую и уходящую в прошлое. Однако в контексте данного

исследования предполагается, что с использованием современного программного обеспечения возможно эффективно проводить анализ данных на основе технологии многомерного анализа данных OLAP.

В текущей ситуации компании, которые используют системы анализа данных на основе технологии многомерного анализа данных OLAP, задумываются о способах проводить многомерный анализ данных без использования популярных иностранных платных решений. Так как сервисы для анализа данных на основе технологии многомерного анализа данных, представленные на отечественном рынке, часто являются дорогостоящими, решения с открытым исходным кодом (open-source) могли бы стать альтернативой для многих компаний.

Актуальность исследования определяется ростом потребности в комплексной оценке OLAP-систем для анализа данных с открытым исходным кодом в условиях ограничений на использование зарубежного программного обеспечения.

Проблема, которую могла бы решить данная научно-исследовательская работа, — это отсутствие разработанных методик для всесторонней оценки производительности и бизнес-эффективности OLAP-систем, которые учитывают оценку устойчивости системы, а также необходимость провести сравнение эффективности и производительности инструментов анализа данных на основе технологии многомерного анализа данных OLAP с открытым исходным кодом. Данная проблема широко не изучена, так как до недавнего времени российским компаниям не приходилось существовать в условиях столь жестких санкций и, как следствие, рынок технологических инструментов никогда не был настолько ограниченным. В новых рыночных условиях поиск оптимального технологического решения становится актуальной проблемой для многих предприятий.

Гипотеза исследования: системы анализа данных на основе OLAP-продуктов с открытым исходным кодом могут конкурировать в эффективности с коммерческими OLAP-продуктами.

Объект исследования — это системы анализа данных с открытым исходным кодом, поддерживающие технологию многомерного анализа данных OLAP.

Предметом исследования является оценка эффективности систем, поддерживающих технологию многомерного анализа данных OLAP.

Целью работы является комплексное сравнение эффективности OLAP-систем.

Для достижения поставленной цели были сформулированы и решены следующие задачи:

– рассмотрение теоретических аспектов систем интерактивной аналитической обработки данных;

– обзор существующих методов и инструментов многомерного анализа данных OLAP, включая как коммерческие, так и открытые решения, с оценкой их функциональности и производительности;

– исследование актуальности применения технологии многомерного анализа данных OLAP;

– анализ деятельности предприятия для выявления проблемных процессов, требующих аналитического вмешательства. Сбор требований и выработка рекомендаций по функциональности и структуре будущей системы OLAP;

– моделирование и реализация системы OLAP;

– разработка методики оценки эффективности OLAP-систем. Оценка эффективности и функциональности разработанной системы с точки зрения удовлетворения требований бизнеса и возможности проведения аналитических исследований.

Научная новизна заключается в разработке методики комплексной оценки производительности и бизнес-эффективности OLAP-систем с включением нагрузочного тестирования, что является малоизученным аспектом.

Теоретическая значимость исследования заключается в развитии и углублении знаний о методах оценки производительности OLAP-систем и сравнении характеристик средств анализа данных, предоставляющих возможность применения OLAP-технологии.

Практическая значимость исследования состоит в том, что результаты данного исследования предоставляют полезные рекомендации относительно использования эффективной и экономически выгодной альтернативы коммерческим OLAP-системам компаниям, сталкивающимся с необходимостью замены ранее используемых иностранных решений в области анализа данных.

В процессе исследования были использованы следующие методы: аналитический метод, метод экспертной оценки, метод моделирования.

На защиту выносятся:

- Оценка эффективности инструментов OLAP с открытым исходным кодом на основе проведенного эксперимента.

- Разработанная методика оценки OLAP-систем с включением оценки устойчивости.

- Результаты апробации гипотезы исследования, подтверждающие конкурентоспособность инструментов OLAP с открытым исходным кодом.

Работа состоит из введения, четырех разделов, заключения и списка литературы. Объем работы составляет 93 страницы и содержит 43 рисунка, 6 таблиц. Список литературы включает 67 наименований источников.

Во введении обосновывается актуальность выбранной темы исследования, определяется объект, предмет и цель исследования, выдвигается гипотеза и формулируются задачи работы, рассматриваются научная новизна исследования.

В первом разделе описываются теоретические аспекты систем интерактивной аналитической обработки данных.

Во втором разделе проведен анализ существующих методов и инструментов многомерного анализа данных OLAP, включая как

коммерческие, так и открытые решения. Проводится исследование актуальности применения технологии многомерного анализа данных OLAP.

В третьем разделе проводится анализ деятельности предприятия для выявления проблемных процессов, требующих аналитического вмешательства, и производится моделирование будущей системы анализа данных.

В четвертом разделе производится реализация системы OLAP, разработка методики эффективности OLAP-систем, оценка эффективности и функциональности разработанной системы с точки зрения удовлетворения требований бизнеса и возможности проведения аналитических исследований.

В заключении представлены основные результаты поставленных задач и полученные выводы.

Основные результаты исследования представлены в публикациях:

- Щербакова Д.О. Обзор основных программных продуктов для построения систем анализа данных по технологии OLAP: сборник статей по материалам CCCV Международной заочной научно-практической конференции «Молодой исследователь: вызовы и перспективы» (Москва, Апрель 2023 г.);

- Щербакова Д.О. Оптимизация процесса работы с данными с помощью инструмента многомерного анализа данных Atoti: сборник статей по материалам Международной заочной научно-практической конференции «Молодой исследователь: вызовы и перспективы» (Москва, Декабрь 2023 г.).

1 Теоретические аспекты систем онлайн аналитической обработки данных

1.1 Реляционные базы данных

Глобальное увеличение количества данных привело к тому, что компании столкнулись с необходимостью их хранить таким образом, чтобы их можно было в любой момент времени запросить и наглядно представить, если это понадобится.

Транзакционные базы данных (OLTP) используются для хранения транзакций в реляционной форме. Реляционная форма предполагает наличие в данных реляций (отношений, связей). Такая форма хранения интуитивно понятна пользователя. Отношения позволяют четко определять, как субъекты анализа связаны между собой.

Таблицы, которыми представлена реляционная форма хранения, имеют интуитивно понятный вид и тем самым дают пользователю возможность производить такие действия, как вставка, обновление, удаление данных с помощью простых шаблонных действий, для изучения которых не требуется специальных знаний. Наглядная табличная форма помогает быстро убедиться в правильности или неправильности произведенных действий.

Запросы в таких базах данных оформляются на языке SQL (язык структурированных запросов), удобство и простота которого также вносит свой вклад в популярность реляционных баз данных. [10, с. 28].

На рисунке 1 представлен процесс обработки запроса в реляционной базе данных.

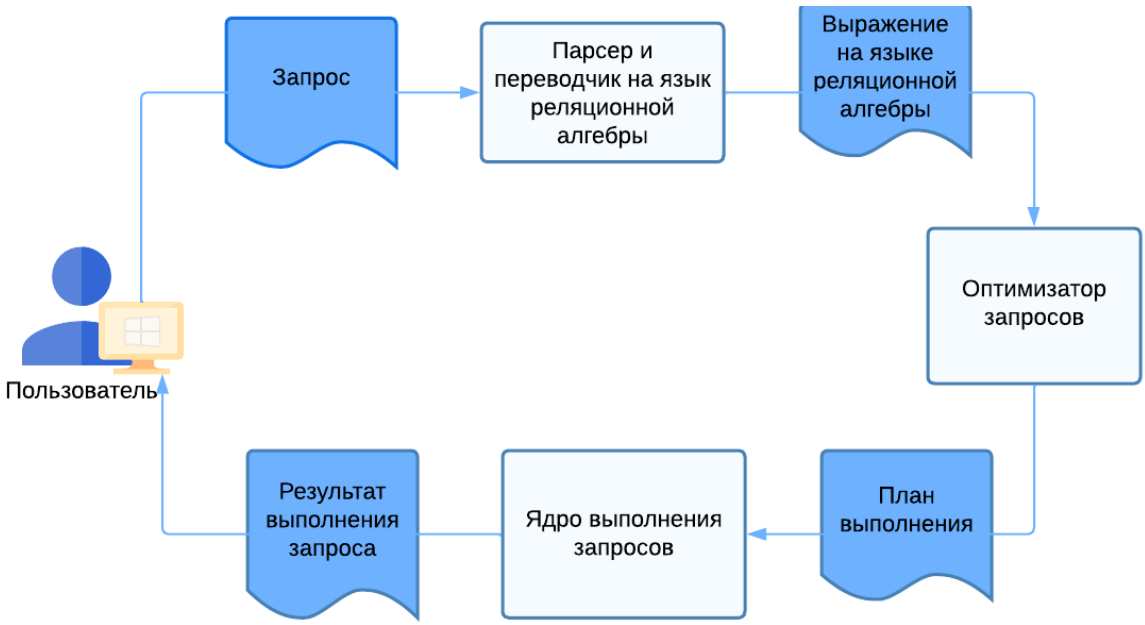

Рисунок 1 – Процесс обработки запроса к реляционной базе данных

Когда данные запрашиваются из базы, запрос сначала анализируется (проверка на соответствие синтаксическим правилам SQL). После этого база данных находит способ извлечь данные наиболее эффективно и составляет план выполнения, согласно которому обрабатывает данные и возвращает их пользователю в том виде, в котором он запрашивал средствами SQL.

В транзакционной базе данных данные хранятся таким образом, чтобы соответствовать концепции нормализации. Концепция нормализации в рамках реляционной модели данных была сформулирована Э. Ф. Коддом: «В нормализованном отношении все неключевые атрибуты функционально зависят от ключа отношения» [41]. Это означает, что данные, описывающие характеристики основных данных, в основной таблице должны быть закодированы в виде некоторого уникального для данного атрибута ключа.

На рисунке 2 представлены таблицы, содержащие один и тот же набор информации, хранение которой реализовано в разных формах: нормальной и ненормализованной. Как можно увидеть из рисунка 2, в основной таблице нормальной формы хранения нет никакой содержательной информации. Она содержит в себе внешние ключи, и только их сочетание позволяет

определить, какой именно набор изделий кодирует каждая отдельная строка в таблице. Нормальная форма не является наглядной и интуитивно понятной, так как, чтобы получить информацию, которая несет некоторый смысл для пользователя, ему придется по ключам искать в других таблицах содержательную информацию, которые они кодирует. В примере на рисунке 2 по ключам мы можем определить цвет и материал изделия. В ненормализованной форме это наглядно представлено в одной таблице.

Несмотря на то, что на первый взгляд нормальная форма хранения неудобна пользователю, это позволяет не хранить данные в базе в избыточном виде, так как ключ имеет малую размерность и требует мало ресурсов хранения.

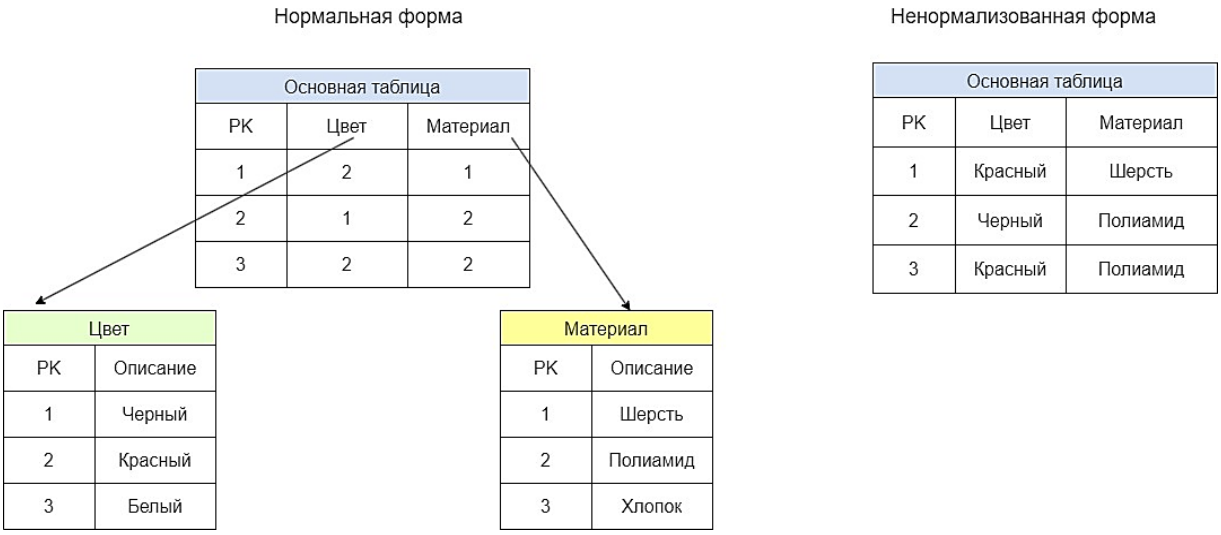

Рисунок 2 – Разница в структуре хранения в нормальной и ненормализованной форме

К плюсам нормальной формы хранения также относится то, что такая форма позволяет уменьшить число ошибок на основе человеческого фактора. Пользователю, добавляющему данные в такую базу, позволено внести данные, которые ограничены разработанной ранее структурой. То есть, набор возможных значений ограничен списком предопределенных ключей. В таком

случае ошибка при вводе намного менее вероятна, поэтому это позволяет сохранить целостность данных.

При этом, такая сложная структура хранения, а именно распределение данных по некоторому числу таблиц, сказывается негативно на скорости выполнения запросов.

Это обусловлено тем, что для выполнения запроса на извлечение какой-либо содержательной информации, содержащей не ключи, а текстовые характеристики, необходимо совершить объединение связанных таблиц [37, с. 2].

Операция объединения в реляционной базе данных – это действие, которого аналитики стараются избегать, так как она является одной из самых затратных по ресурсам.

Для эффективной работы с большими данными необходим «глубокий анализ в различных разрезах, выявление закономерностей, прогноз вероятных будущих данных» [33].

Как пишет А.Ю. Копова: «аналитик обеспечивает полноту сбора данных о внешнем рынке, отвечает за итоговую интерпретацию результатов автоматизированного процесса обработки данных, отвечает за представленный в системе контент» [16]. Из этого следует, что аналитическая работа подразумевает выполнение запросов для извлечения содержательной информации.

Очевидно, что реляционный формат хранения не приспособлен для проведения анализа данных, потому что такие базы являются слишком медлительными с точки зрения получения информации. При этом, они являются лучшим выбором для хранения данных, которые постоянно добавляются или обновляются, так как операции записи и изменения в этих таблицах не требуют много ресурсов.

Также часто встречается ситуация, при которой аналитику необходимо предоставлять частые отчеты (например, ежедневные), так как данные часто обновляются, и аналитику важны актуальные тенденции. Такая ситуация

требует использования одновременно двух бах данных: OLTP и хранилище данных.

1.2 Хранилище данных

Концепция хранилища данных возникла в середине 1980-х гг., когда появилась необходимость анализировать большие объемы информации и подготавливать управленческую отчетность.

Понятие хранилища данных было впервые определено Б. Инмоном: «хранилище данных – это предметно-ориентированная, интегрированная, неизменяемая и поддерживающая хронологию электронная коллекция данных для обеспечения процесса принятия решений» [49].

Согласно определению Р. Кимбалла: «хранилище данных — это система, которая извлекает, очищает, сопоставляет и передает исходные данные в хранилище размерных данных, а затем поддерживает и реализует запросы и анализ с целью принятия решений» [53].

На рисунке 3 показана типичная архитектура хранилища данных. Типичное хранилище включает в себя OLAP-сервер, как это показано на рисунке 3 [44].

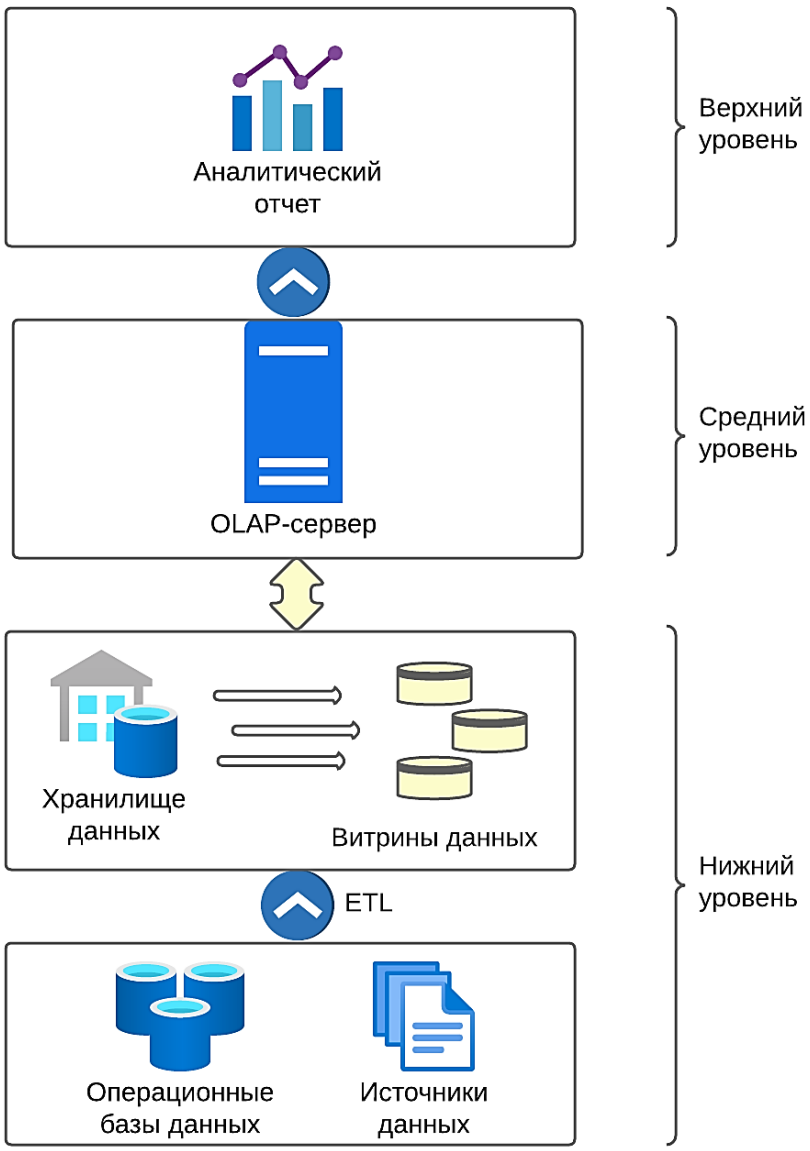

Рисунок 3 – Архитектура хранилища данных с включением OLAP-сервера

1.3 Технология OLAP

Впервые термин «OLAP» был введен Э. Ф. Коддом. По определению Кодда: «OLAP – технология оперативной аналитической обработки данных, заключающаяся в подготовке агрегированной информации на основе больших массивов данных» [40].

Как видно из данного определения, OLAP позволяет хранить и запрашивать предрассчитанные данные. Как указывают Е.А. Берсенева, А.А. Седов: «с точки зрения конечного пользователя, суть OLAP-технологии

состоит в том, что данные ему предоставляются в динамической таблице, автоматически суммирующей их в различных разрезах и позволяющей интерактивно управлять как вычислениями, так и формой отчета» [3]. Это решает проблему реляционной формы хранения, которая была обозначена выше: низкая скорость запросов при наличии большого числа отношений в базе данных.

Технология OLAP своим базовым условием ставит необходимость хранения в многомерной форме, в виде так называемого куба данных. Пример такой кубической структуры хранения представлен на рисунке 4.

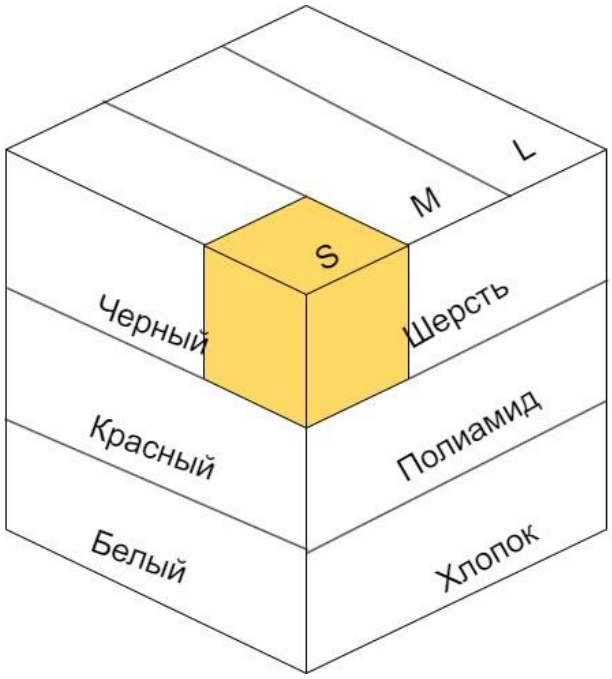

Рисунок 4 – Пример многомерного представления данных

В кубе данные агрегированы по всем разрезам, которые задаются при его построении. Например, в кубе на рисунке 4 были заданы измерения куба: размер, материал и цвет. Для любого сочетания этих измерений в кубе хранятся предрассчитаные агрегированные количественные данные. Форма агрегации данных также может быть задана пользователем. Так, можно получить данные о средней цене изделий с определенным цветом и размере,

о сумме издержек на изделия с определенным материалом и цветом и так далее.

На рисунке 4 желтым выделено пересечение измерений, для которого можно быстро получить агрегированные количественные данные об изделиях из шерсти черного цвета и размера «S». К примеру, можно сравнить среднюю прибыль от таких изделий с прибылью от изделий с другими параметрами, и этот анализ позволит принять информированное решение о том, какие изделия производить в будущем.

При этом скорость доступа к данным будет намного быстрее, чем при получении этих данных из реляционной формы хранения. Куб обычно требует редкого обновления, и это способствует тому, что аналитику не придется сталкиваться с перегрузкой сервера базы данных [11, с. 2].

Высокая скорость запросов и отсутствие перебоев в работе сервера базы данных объясняет востребованность многомерных кубов данных для компаний, у которых возникает необходимость работы с большими данными.

При этом, технология OLAP имеет ряд слабых сторон [15]:

- каждое обновление данных в кубе приводит к необходимости обновления данного куба. Так как перестроение куба данных – это часто очень ресурсоемкая задача, это может стать очевидным недостатком. Как будет показано далее в данной работе, часть поставщиков программного обеспечения встроили в свои продукты функции, позволяющие не обновлять куб полностью при внесении некоторых изменений его в структуру;

- для проведения анализа обычно необходимо выделять общие тенденции, тренды. При этом точность данных может пострадать. Это упрощает данные. При некорректном использовании технологии упрощение может привести к ситуации, когда некоторые важные характеристики данных будут представлены неверно.

1.4 Хранилище данных с использованием OLAP-сервера

При разработке хранилища данных важной задачей становится определение формы хранения в данном хранилище.

Согласно А. Федорову и Н. Елмановой: «Для моделирования хранилища данных были введены концепции таблицы фактов и таблицы измерений. В таблице фактов хранятся накопленные количественные величины, в то время как в таблице измерений — качественные, описательные данные. Ключевым правилом разделения данных является следующее: в таблице фактов содержится то, что анализируется, а в таблице измерений — то, относительно чего выполняется анализ» [30, с. 22].

Из этого следует, что в кубе данных на рисунке 4 размер, цвет, материал – это измерения. Эта информация должна храниться в таблицах измерений. Ссылки на эту информацию хранятся в таблице фактов, которая, помимо этого, содержит количественные данные (например, о ценах, выручке, издержках).

Есть две основных модели хранилища данных: схемы хранения «звезда» и «снежинка». Данные схемы обязаны своим названиям их формам, которые схематично представлены на рисунках 5-6.

В схеме «звезда» таблица фактов должна содержать все внешние ключи, которые есть в базе данных. Все эти внешние ключи являются первичными ключами для таблиц измерений. Важная особенность данной схемы – это то, что таблицы измерений не представляет самостоятельную ценность для аналитика, так как не содержит никакой содержательной информации.

Примерная структура схемы "звезда" представлена на рисунке 5.

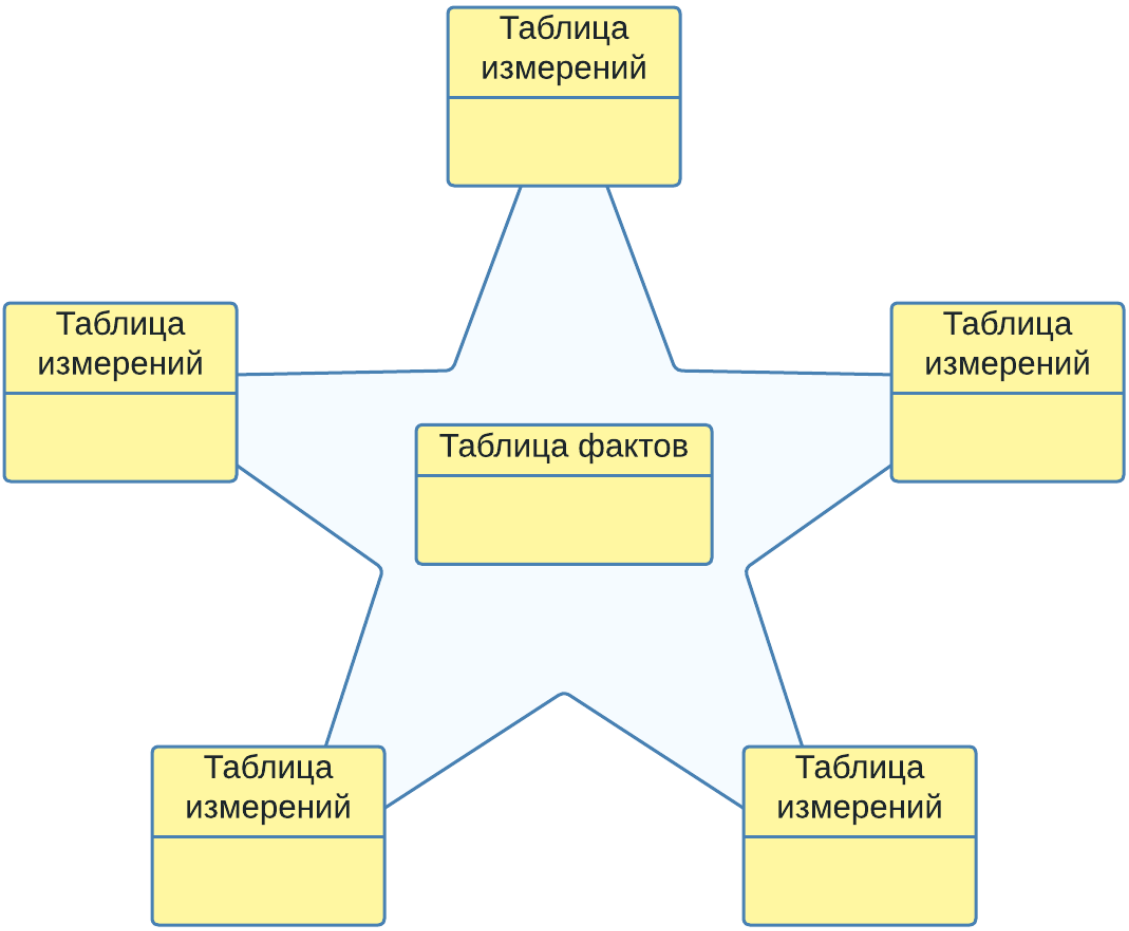

Рисунок 5 – Пример структуры хранилища данных, построенного по схеме «звезда»

В отличие от схемы «звезда», схема «снежинка» состоит из трех типов таблиц: таблицы фактов, таблиц измерений и таблиц вложенных измерений (субизмерений). Таблицы измерений (все или несколько) содержат ссылки на таблицы вложенных измерений. Как пишет М.З. Кармани: «из-за наличия таблиц вложенных измерений в схеме «снежинка» вводится концепция иерархии» [48].

На рисунке 6 показана примерная структура схемы «снежинка».

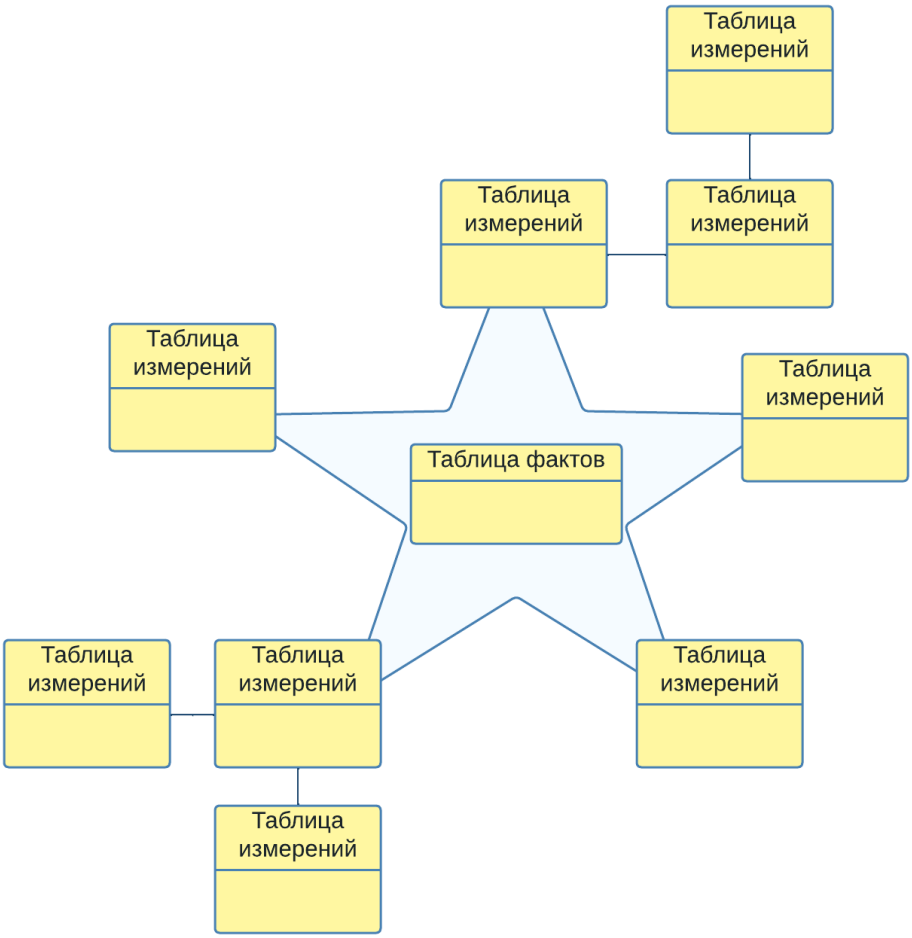

Рисунок 6 – Пример структуры хранилища данных, построенного по схеме «снежинка»

Производительность запросов в схеме «звезда» ниже, чем в ненормализованной форме хранения, но выше, чем в схеме «снежинка» за счет того, что в схеме «звезда» требуется меньшее количество объединений [38, с. 2].

Сложная структура связей в схеме «созвездие» обуславливает низкую производительность запросов при такой схеме хранения. При этом углубление иерархии измерений снижает скорость запросов значительнее, чем горизонтальное расширение измерений [51].

Однако если данные имеют потенциал для дальнейшей нормализации, схема «звезда» является более затратной по требуемым ресурсам хранения, так как данные в такой схеме находятся в ненормализованной форме.

При выборе схемы необходимо учитывать размер таблиц измерений. Если измерения представляют собой широкие таблицы, то предпочтительнее будет схема «снежинка». Но следует учитывать, что схема «снежинка» будет требовательнее к ресурсам для больших объемов данных [45].

Схема «созвездие» («Galaxy scheme», галактика) является расширением схемы звезды. Таблиц фактов в такой схеме больше одной, и эти таблицы связаны между собой общими измерениями. Название такой схемы обусловлено тем, что ее можно определить как соседство нескольких звезд («созвездие») или созвездий («галактика») [62].

На рисунке 7 представлена примерная структура схемы «созвездие».

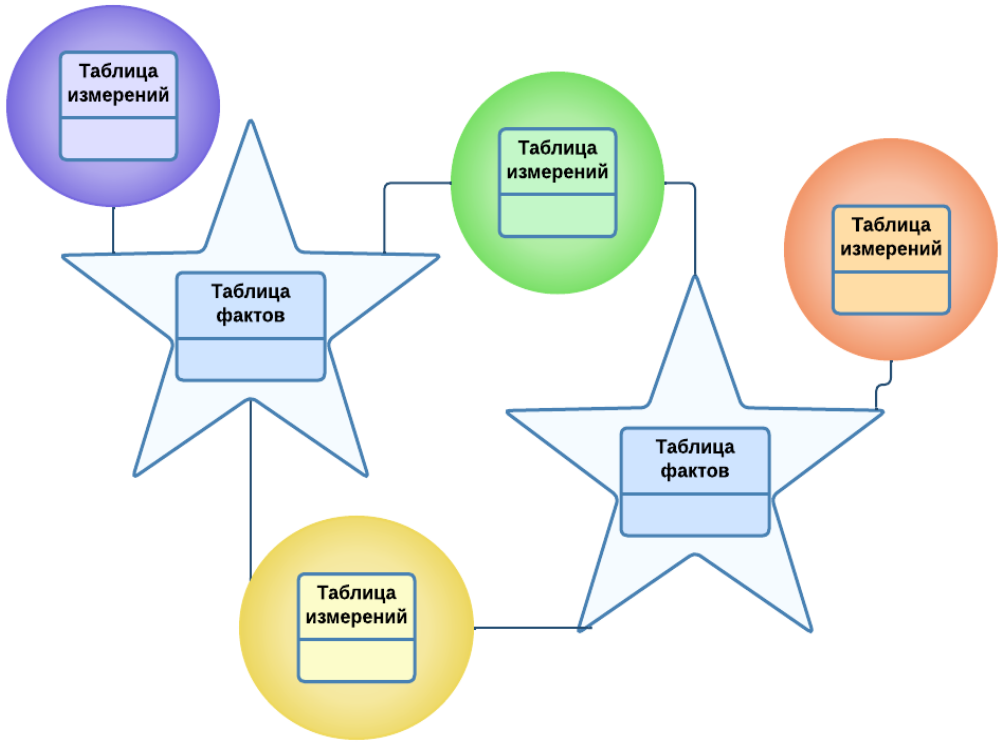

Рисунок 7 – Пример структуры хранилища данных, построенного по схеме «созвездие»

Эта схема является нестандартным выбором, однако в ряде случаев она применима. Иногда бизнес накапливает разноплановые количественные данные, которые хранятся в разных таблицах для того, чтобы их удобнее было анализировать, разграничивать доступ к таблицам, делать отдельные

таблицы менее тяжеловесными. Зачастую эти таблицы фактов являются таблицами с большим числом строк. Объединение этих таблиц в одну широкую таблицу, как этого требуют схема «звезда» или «снежинка», является нерациональным.

В таблице 1 представлено сравнение схемы «созвездие» со схемами «звезда» и «снежинка».

Таблица 1 – Классификация моделей данных для OLAP-хранилища [45]

Характеристики для сравнения	Схема "звезда"	Схема "снежинка"	Схема "созвездие"
Связи в схеме	Одна таблица фактов соединена с несколькими таблицами измерений	Одна таблица фактов соединена с несколькими таблицами измерений, которые соединены с таблицами субизмерений	Несколько таблиц фактов соединены с таблицами измерений
Нормализация данных	Денормализованная форма данных	Нормализованная форма данных	Нормализованная форма данных
Количество измерений	Несколько таблиц измерений относятся к одной таблице фактов	Несколько таблиц измерений относятся к одной таблице фактов и к субизмерениям	Несколько таблиц измерений относятся к нескольким таблицам фактов
Избыточность данных	Есть	Нет	Нет
Производительность	Малое количество внешних ключей за счет отсутствия внешних связей в измерениях приводит высокой производительности	Большое количество внешних ключей приводит низкой производительности	Несколько таблиц фактов и сложные связи между ними и измерениями приводят к низкой производительности

Продолжение таблицы 1

Характеристики для сравнения	Схема "звезда"	Схема "снежинка"	Схема "созвездие"
Сложность	Простая для понимания и интерпретации	Более сложная, чем схема "звезда"	Используется для сложных структур данных, сложна для интерпретации
Ограничения	Можно использовать только одну таблицу фактов, нельзя использовать субизмерения	Можно использовать только одну таблицу фактов	Нельзя использовать субизмерения
Использование памяти	Из-за избыточности данных требует большое количество памяти	Из-за отсутствия избыточности в данных требует меньше места на диске	Из-за отсутствия избыточности в данных требует меньше места на диске

1.5 Алгебраические операции OLAP

Алгебраические операции OLAP составлены из атомарных операторов. Атомарность в данном случае означает, что эти основные операторы не могут быть выражены какой-либо комбинацией других операторов. Анализ работ по данной теме показал, что пять категорий атомарных операторов считаются основными для алгебры многомерных структур [66].

- SELECT – это оператор, который позволяет пользователю выбрать определенные значения куба.

- DRILLING – это вид операций, который включает в себя операторы ROLL UP и DRILL DOWN. Они позволяют перемещаться вглубь иерархий куба и наоборот. Выбор более высокого уровня детализации осуществляется оператором ROLL UP. К примеру, этой командой можно перейти от размерности измерений «Округа» к размерности «Район». Выбор более

глубокого погружения по иерархии осуществляется оператором DRILL DOWN, что позволяет в противоположность к предыдущему примеру, поменять размерность от «Района» к «Округу».

- ROTATE – это оператор вращения куба. Вращением куба можно выбирать другие измерения куба для анализа. FROTATE – это оператор для вращения количественных данных (фактов).

- Оператор ADD позволяет изменить пользовательские меры (кастомизированные вычисления, заданные пользователем). Меры можно добавлять командой ADDM или удалять командой DELM.

- Оператор агрегирования AGGREGATE используется для определения того, как агрегированные значения будут выводиться в качестве итогового значения по строкам и столбцам сводных таблиц.

Схематичное изображение данных операций представлено на рисунке 8.

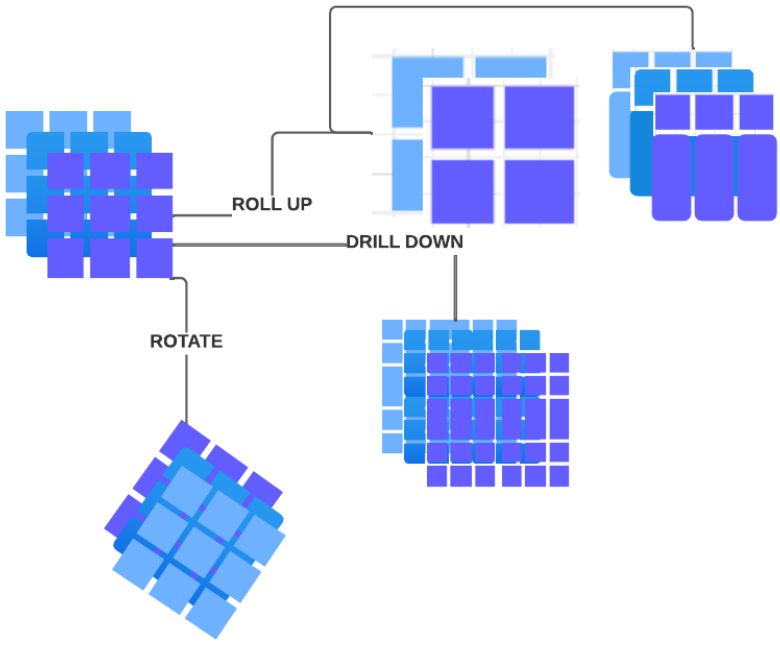

Рисунок 8 – Основные алгебраические операции OLAP

Алгебра для схемы «созвездие» может отличаться от классических схем. Ф. Рават, Р. Турнье выделяют 4 алгебраические операции для данной схемы, которые отличаются от традиционных операций OLAP.

− Операция фокусировки используется для фокусирования данных на несколько осей.

− Операция выбора подмножества используется для уменьшения области анализируемых данных.

− Чтобы воспользоваться преимуществами иерархически упорядоченных параметров, пользователю потребуются операции изменения уровня детализации анализируемых данных.

− Чтобы изменить критерии анализа, необходима операция вращения анализируемого объекта вокруг других осей анализа или вращения осей анализа вокруг разных объектов. [63]

В некоторых моделях авторы указывают на возможность согласованной обработки параметров [35, 39, 48].

1.6 Требования к OLAP-системам. FASMI

В 1993 г. Э. Ф. Кодд и его коллеги разработали ряд правил-рекомендаций для технологии OLAP.

Очевидно, что ключевым свойством OLAP-системы является многомерность. Это также важное свойство для анализа данных, так как рассмотрение данных с точки зрения разных измерений позволяет найти скрытые зависимости в данных.

Такая система должна быть встроена в общую архитектуру системы поддержки принятия решений и предоставлять понятный интерфейс. Система должна оцениваться пользователями как легкая в использовании и доступная для неопытных пользователей. Клиент-серверная архитектура должна обеспечивать простое подключение различных клиентов к серверу.

Манипулирование данными должно быть неограниченным по числу используемых измерений.

Такая система должна обеспечивать целостность данных, при этом предоставляя возможность пользователям пользоваться кубом совместно.

Система должна быть стабильной, не перегружаться из-за увеличения числа измерений. Все измерения в системе должны быть одинаковыми по важности и возможностям. В системе должен быть реализована простая интеграция с многими источниками данных.

Размерность и уровни агрегации должны быть неограниченными. Система должна поддерживать практически любое количество измерений данных, и каждое из этих общих измерений должно предоставлять пользователю возможность определять почти неограниченное число уровней агрегации. [42]

В 1995 г. на основе требований, изложенных Э. Ф. Коддом, Н. Пендсом и Р. Критом был сформулирован тест FASMI, включающий требования к приложениям для многомерного анализа. На рисунке 9 представлены требования теста из работы Н. Пендса [59].

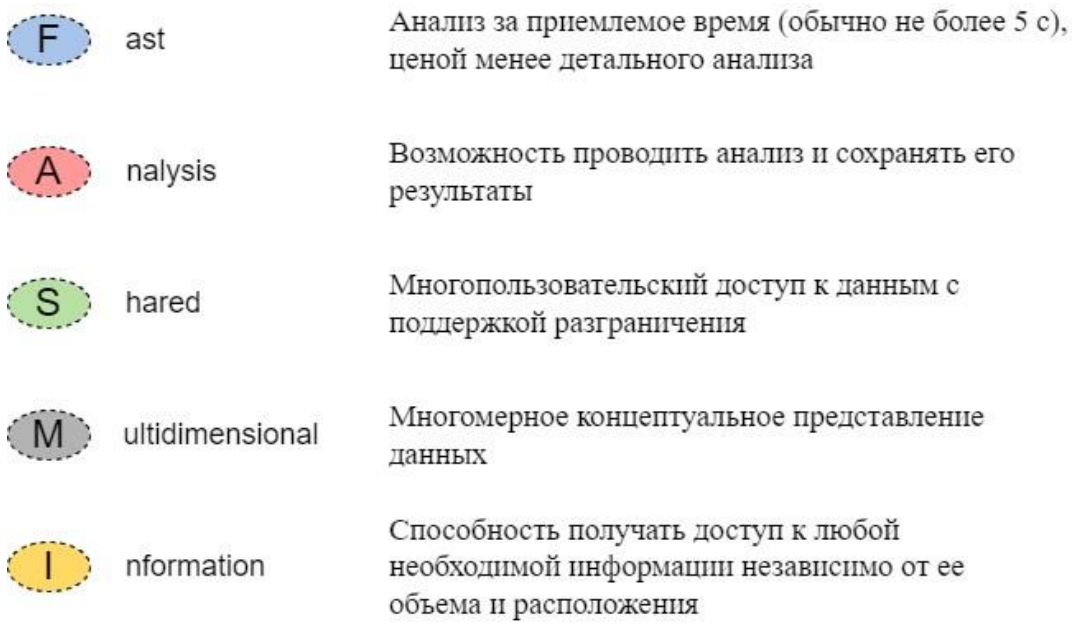

Рисунок 9 – Требования FASMI

1.7 Классификация OLAP-систем

Сервер OLAP может функционировать разными методами:

- в качестве системы управления реляционными базами данных, которая переводит операции над многомерными данными в стандартные операции реляционной модели;

- в качестве системы, использующей многомерную модель хранения, что позволяет работать напрямую с многомерными данными без использования реляционных таблиц [44].

В. Кириллов, Г. Громов указывают, что есть три основных типа OLAP-систем:

– реляционные (ROLAP): данные в виде совокупности отношений. ROLAP представлен в большинстве BI-систем. Здесь нет предварительно определенной схемы куба, следовательно, ROLAP ассоциирован с большими вычислительными нагрузками;

– многомерные (MOLAP): данные предварительно агрегированы, и запросы к ним являются заранее предопределёнными. Такие данные организованы в виде упорядоченных многомерных массивов, которые представляют собой гиперкубы данных;

– смешанные (HOLAP): сочетание инструментов, реализующих реляционную и многомерную модель данных. В этом случае в агрегированные данные хранятся в структуре MOLAP, но только до определенной степени детализации [13].

Основные различия в типах OLAP-систем представлены на рисунке 10 и таблице 2.

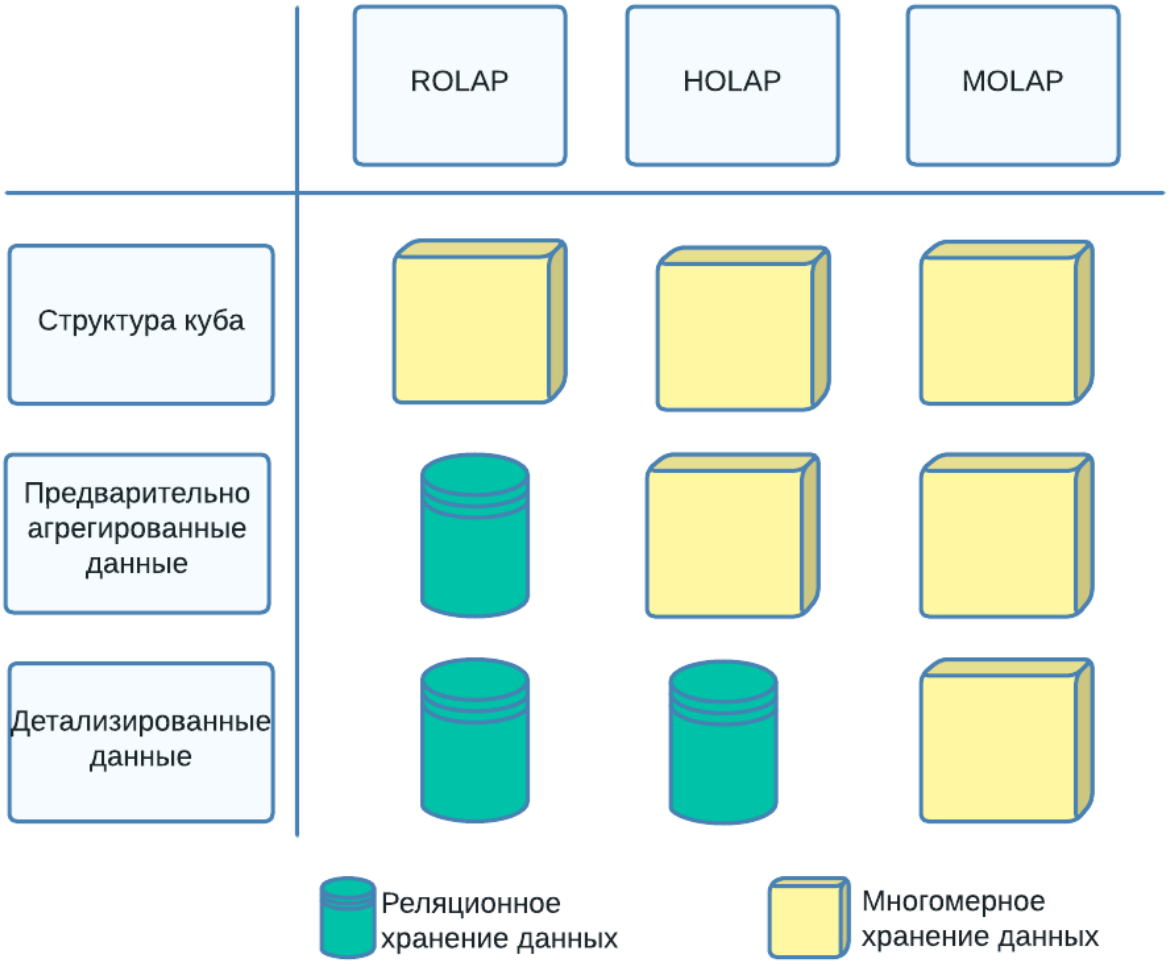

Рисунок 10 – Организация хранения данных в разных типах OLAP-систем

Таблица 2 – Классификация OLAP-систем [47, с. 6]

Характеристика	ROLAP	MOLAP	HOLAP
Масштабируемость	Высокая степень масштабируемости	Низкая степень масштабируемости	Средняя степень масштабируемости
Использование памяти	Наименьший возможный объем памяти	Наибольший возможный объем памяти	Средний объем памяти

Продолжение таблицы 2

Характеристика	ROLAP	MOLAP	HOLAP
Производительность	Система не оптимизирована для выполнения сложных многотабличных запросов. Следствием этого является низкая производительность	Система оптимизирована для быстрого расчёта агрегатных показателей.	Система обеспечивает более высокую скорость при получении ответов на сложные многотабличные запросы, чем ROLAP-система, но ниже, чем MOLAP-система
Скорость обработки запросов	Низкая скорость	Высокая скорость	Средняя скорость
Уровень защиты информации и разграничения прав доступа	Высокий уровень защиты информации	Низкий уровень защиты информации	Средний уровень защиты информации

На сегодняшний день помимо традиционных форм OLAP на рынке присутствуют альтернативные подходы.

Многие компании вместо традиционного подхода выбирают технологию In-memory OLAP. Такой подход может быть полезен компаниям, которые хотели бы найти компромисс между традиционными подходами, так как принципиальная новизна In-memory OLAP состоит в том, что данные агрегируются и хранятся в оперативной памяти. При этом, система оптимизирована для хранения в оперативной памяти, и это приводит к максимально эффективному ее использованию в этом качестве.

Для реализации этой технологии используется промежуточная система баз данных, обрабатывающая запросы, которая «хранится в оперативной памяти компьютера, что позволяет избежать задержек, связанных с обращениями к диску» [20].

А.Н. Андреев относит к плюсам подхода In-memory OLAP то, что «не существует угрозы «взрыва» данных, так как в кубе не сохраняются предварительно вычисленные значения» [1].

Как указывает С. Кирюшин, к минусам In-memory OLAP относится «ограниченность хранения и обработки куба данных объемом основной памяти» [14];

Основные различия в архитектуре OLAP-систем представлены на рисунке 11.

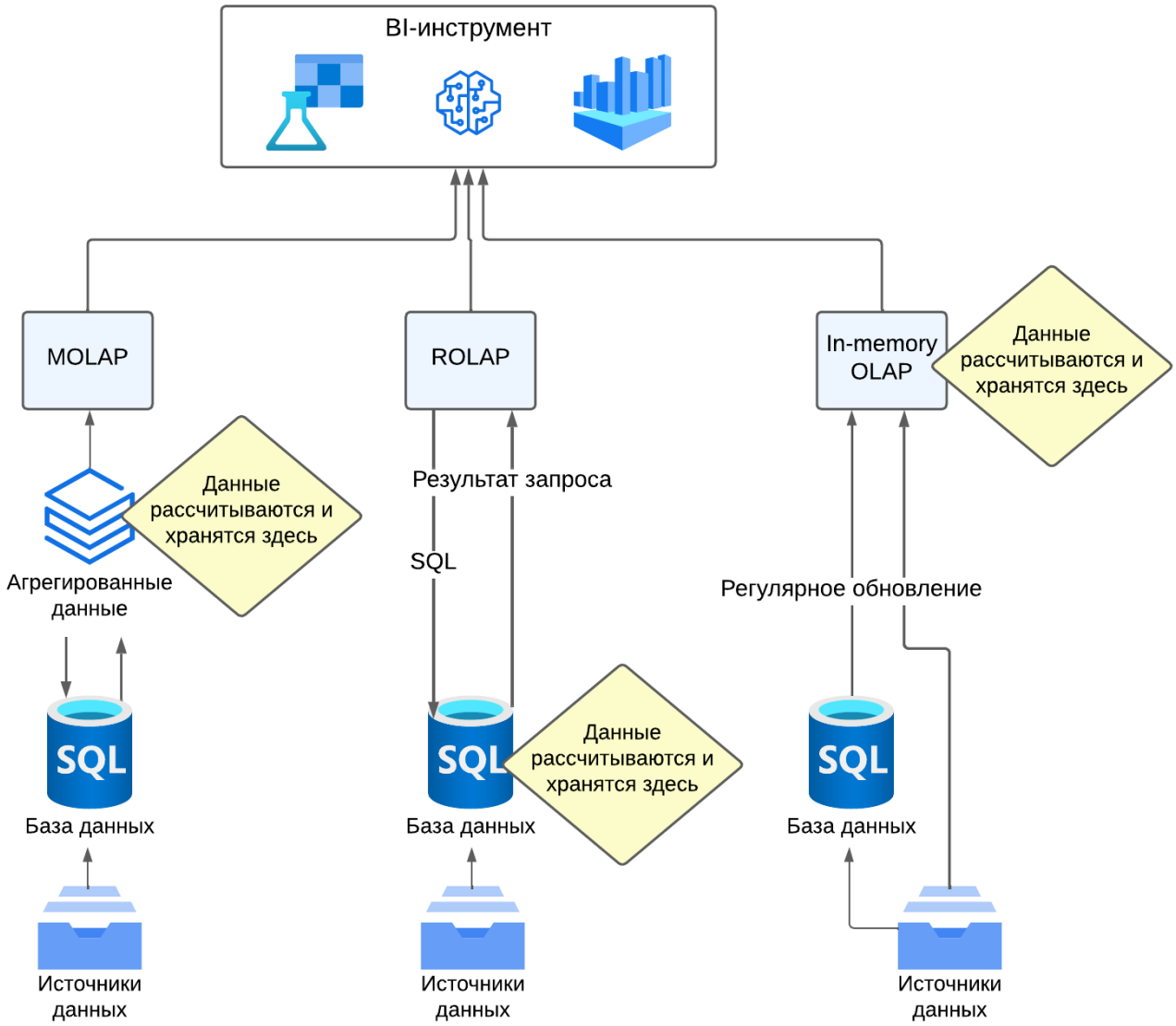

Рисунок 11 – Различия в архитектуре OLAP-систем

Выводы по первому разделу

В ходе анализа работ отечественных и зарубежных авторов были сформулированы следующие выводы.

Аналитическая работа подразумевает выполнение запросов для извлечения содержательной информации, и это обуславливает популярность технологии OLAP, которая позволяет сделать скорость запросов максимальной.

Куб требует редкого обновления, и это способствует тому, что аналитику не придется сталкиваться с перегрузкой сервера базы данных. Если компании необходимо постоянно обновлять данные для аналитики, то технология OLAP может быть неприменима, так как перестроение куба данных – это часто очень ресурсоемкая задача.

Среди недостатков технологии OLAP можно выделить высокие временные затраты на реализацию системы, необходимость больших объемов памяти и упрощение структуры данных.

2 Обзор основных методологий и программных продуктов для построения систем анализа данных по технологии OLAP

2.1 Обзор функциональных возможностей наиболее популярных программных продуктов для анализа данных на основе технологии OLAP

На рынке существует множество различных производителей OLAP-систем. Согласно отчету компании Gartner на январь 2022 года, самыми востребованными зарубежными продуктами OLAP с закрытым исходным кодом были Microsoft Analysis Services, Qlik, Oracle Essbase, IBM Cognos TM1 [17, с. 180]. Характеристики этих продуктов представлены в таблице 3.

Таблица 3 - Характеристики иностранных OLAP-продуктов с закрытым исходным кодом [32].

OLAP-продукт	Операционная система		Тип хранения				Инструменты визуализации
	Windows	Linux	MOLAP	ROLAP	HOLAP	In-Memory OLAP	
Microsoft Analysis Services	Да	Нет	Да	Да	Да	Да	Microsoft Excel, Microsoft Power BI
Essbase	Да	Да	Да	Да	Да	Нет	Oracle Analytics Cloud, Narrative Reporting, Tableau
IBM Cognos TM1	Да	Да	Да	Нет	Нет	Да	IBM Cognos Insight, IBM Performance Modeler, IBM Cognos Cafe Cognos BI, TM1 Perspectives for Excel

Продолжение таблицы 3

OLAP-продукт	Операционная система		Тип хранения				Инструменты визуализации
	Windows	Linux	MOLAP	ROLAP	HOLAP	In-Memory OLAP	
Qlik	Да	Да	Нет	Нет	Нет	Да	QlikView

Microsoft Analysis Services интегрирован в Microsoft SQL Server и использует различные архитектуры хранения данных, однако основной упор делается на MOLAP. Одной из ключевых функций этого сервиса является возможность работы с несколькими связанными таблицами фактов без необходимости их объединения в одну таблицу (схема «созвездие»). Как указывает С.М. Сарсимбаева, эта система позволяет «проектировать, создавать и управлять многомерными структурами, которые содержат подробные статистические данные из нескольких источников данных» [27, с. 2].

Oracle Essbase представляет собой высокопроизводительный движок MOLAP. Создание кубов Essbase может происходить автоматически на основе бизнес-модели BI, и их интеграция в семантический уровень способствует повышению производительности рабочих нагрузок BI. Сервер Essbase обеспечивает расширенные возможности многопользовательского чтения и записи, включая обновление данных и пересчет. Бизнес-пользователи могут передавать данные обратно на сервер и выполнять их пересчет с использованием сценариев вычислений [57].

IBM Cognos TM1 использует многомерную базу данных в форме кубов с возможностью обратной записи в исходные базы данных. Механизм TM1 позволяет проводить сложные вычисления непосредственно в памяти на стороне клиента, что обеспечивает быструю обработку данных и масштабируемость [4].

Среди основных функций платформы Qlik можно выделить поддержку совместной работы в онлайн-режиме в защищенной среде и мощные возможности визуализации данных, включая создание дашбордов [61].

Основной чертой QlikView является применение ассоциативной архитектуры с обработкой данных в оперативной памяти (In-memory OLAP). Ассоциативный анализ предполагает объединение таблиц из различных источников данных, автоматическое обнаружение совпадающих полей и установление ассоциативных связей между ними [9].

На фоне санкций перечисленные выше компании объявили об уходе из РФ. Вследствие этого, компаниям необходимо найти продукты, доступные в стране и обладающие похожей функциональностью.

2.2 Обзор функциональных возможностей программных продуктов для анализа данных на основе технологии OLAP, представленных на российском рынке

Среди российских систем, предоставляющих возможность многомерного анализа данных, наиболее популярными являются Visary BI, Форсайт, Visiology, Polymatica [28]. Характеристики наиболее популярных российских OLAP-продуктов представлены в таблице 4.

Таблица 4 - Характеристики российских OLAP-продуктов [32]

		Visary BI	Форсайт	Visiology	Polymatica
Операционная система	Windows	Да	Да	Да	Нет
	Linux	Да	Да	Да	Да

Продолжение таблицы 4

		Visary BI	Форсайт	Visiology	Polymatica
Тип хранения	MOLAP	Нет	Нет	Нет	Нет
	ROLAP	Да	Да	Да	Нет
	HOLAP	Нет	Нет	Да	Нет
Тип хранения	In-Memory OLAP	Нет	Нет	Да	Да
Совместимые инструменты визуализации		Visary Dashboards, Microsoft Excel	Форсайт Аналитические панели, Microsoft Excel	Visiology Dashboards, Microsoft Excel	Polymatica Dashboards Microsoft Excel

Visary BI — это веб-ориентированное low-code решение. Low-code — это методология, которая позволяет производить программные продукты с минимальным использованием традиционного кодирования. Visary BI легко настраивается под требования пользователя через гибкие инструменты настройки и учитывает отраслевую специфику. Система обеспечивает корректную работу с отечественными операционными системами и системами управления базами данных. Visary BI поддерживает работу с неограниченным количеством источников данных. Есть ETL-коннектор, возможность построить OLAP-куб [8].

Форсайт — это платформа, которая предоставляет широкие возможности по подготовке и анализу данных, а также обеспечивает стабильную работу [31]. Однако эта система не является дружелюбной к неопытному пользователю из-за сложности внутренней логики построения аналитических отчетов. Из преимуществ системы можно отметить наличие репозиториев, которые позволяют создавать структурированную систему каталогов, папок и файлов, что обеспечивает организацию удобного доступа к элементам аналитических отчетов. Файлами могут являться отчёты

различных видов, объекты, выполняющие функции ETL, и другие виды объектов [7].

Visiology — аналитическая платформа, предназначенная для построения информационно-аналитических систем. Ее преимуществами являются широкий набор настроек, широкий набор настроек для формирования и отправки регламентных отчетов с возможностью включения данных из нескольких дашбордов, есть возможность использовать в диаграммах данные, рассчитанные на лету, простое, интуитивное освоение системы [67].

Polymatica - аналитическая платформа, объединяющая три продукта: Polymatica Analytics — инструмента для работы с большими объемами данных, Polymatica Dashboards — конструктор информационных панелей и отчетов и Polymatica ML — отвечающий за создание моделей машинного обучения и управление их жизненным циклом. В платформе используется подход «Мультисфера» — технология, позволяющая оперативно обрабатывать большие объемы данных с использованием многомерных моделей в системе Polymatica. Недостатком платформы является отсутствие встроенного ETL-инструмента. Достоинствами являются интуитивный интерфейс и собственный OLAP-сервер с возможностью обработки больших объемов данных в режиме реального времени [60].

2.3 Обзор функциональных возможностей программных продуктов для анализа данных на основе технологии OLAP с открытым исходным кодом

Поскольку «доступность продуктов с открытым исходным кодом не зависит от экономической и политической ситуации, их внедрение может быть наиболее рациональным решением для многих российских компаний» [32].

В таблице 5 представлены характеристик наиболее известных open-source OLAP-продуктов [65].

Таблица 5 - Характеристики иностранных OLAP-продуктов с открытым исходным кодом [32]

OLAP-продукт	Операционная система		Тип хранения				Инструменты визуализации
	Windows	Linux	MOLAP	ROLAP	HOLAP	In-Memory OLAP	
Apache Kylin	Нет	Да	Да	Нет	Нет	Да	Superset, Zeppelin, Tableau, Qlik, Redash, Microsoft Excel
Jedox Palo	Да	Да	Да	Нет	Нет	Да	Microsoft Excel, Qlik, Tableau, Jedox Web, Power BI
Mondrian	Да	Да	Нет	Да	Нет	Нет	Jpivot
Atoti	Да	Да	Нет	Нет	Нет	Да	Jupyter Lab, Jupyter Notebook, Microsoft Excel
Cubes	Да	Да	Нет	Да	Нет	Да	CubesViewer
Clickhouse	Нет	Да	Да	Нет	Нет	Нет	Microsoft Excel, DataLens, Tableau

Apache Kylin – это MOLAP сервер, позволяющий быстро обработать большие массивы данных путем построения схемы «звезда», построения OLAP-куба, запуска запросов и получения результатов через API [18].

Jedox Palo – это MOLAP сервер [50]. Palo рассчитан прежде всего на взаимодействие с Excel. Большинство задач, таких как создание измерений и кубов, выполняются с помощью плагина для Excel. Palo Eclipse Client позволяет проектировать и просматривать измерения и кубы в платформонезависимой среде Eclipse Framework [54].

Pentaho от Mondrian – это ROLAP-сервис, архитектура которого состоит из нескольких уровней. Уровень, который в Pentaho называется базовым, необходим для хранения агрегированных данных. На уровне измерений выполняются запросы к этим данных. Уровень хранения – реляционная база данных. То, как отображаются данные, определяется уровнем представления [56, 12].

Atoti – это сервис-библиотека Python. Она позволяет создавать многомерные кубы по технологии In-memory OLAP. Куб моделируется и отображается в среде Python, там же данные можно анализировать и визуализировать [36].

Cubes – это фреймворк для Python, который предоставляет возможности для построения многомерных кубов по технологии ROLAP. При этом Cubes не предоставляет возможность анализировать и визуализировать данные, для этих задач поставляется утилита CubesViewer. Это веб-инструмент, который может использоваться для построения различных графиков [43].

ClickHouse – это NoSQL база данных, которая относится к типу колоночных баз данных. Это высокопроизводительная база данных, которая позволяет без построения многомерных кубов, производить анализ по технологии OLAP. Этому способствует колоночный формат хранения данных, который является очень эффективным, если нет необходимости часто обновлять данные.

По результатам исследования, проведенного в данном разделе, были сделаны следующие выводы.

Российские OLAP-сервисы широко представлены на рынке и обладают большой функциональностью. При этом в большинстве своем они являются платными продуктами. Кроме того, стоимость лицензии чаще всего делает эти продукты представителями верхнего ценового сегмента, что делает их недоступными для небольших компаний.

Многие OLAP-продукты, которые поставляются бесплатно, также предоставляют широкий набор функциональных возможностей для анализа данных. Для большинства небольших компаний функций данных продуктов будет достаточно. Среди них есть продукты, которые поддерживают различные типы хранения данных, включая MOLAP, ROLAP и In-memory OLAP. Это позволит выбрать подходящий продукт в зависимости от требований проекта.

Учитывая тот факт, что открытый исходный код делает эти продукты доступными независимо от политической и экономической ситуации, они могли бы стать подходящим решением для многих компаний.

2.4 Роль OLAP в современном мире

В последнее десятилетие технология OLAP часто подвергается критике. Эта технология многими экспертами признается «устаревшей и уходящей в прошлое» [64]. Однако упадок OLAP — это спорное утверждение, так как это во многом является вопросом правильного применения терминологии.

Около 30 лет назад аналитики данных задумались о том, что использовать реляционные базы данных для больших рабочих нагрузок – это неэффективный подход. Ситуация усугублялась тем фактом, что в те времена компьютеры не были мощными. Это приводило к тому, что подавляющему большинству бизнес-пользователей приходилось использовать относительно небольшие объемы памяти для выполнения рабочих нагрузок BI. Таким образом, первые практики BI остановились на общем подходе извлечения из

реляционной базы данных необходимых данных и помещения их в эффективную структуру данных в памяти для манипулирования.

Но в ситуациях, когда объем данных превышает доступную память компьютера, этот способ не подходит. Поэтому ранние BI-системы решали проблему агрегированием данных и кэшированием агрегированных данных во вложенном массиве. Так появилась технология многомерных структур, ставшая в последующем доминирующим методом хранения данных для их аналитики.

Недостатками классического OLAP-куба являются [21]:

- невозможность работы в реальном времени;
- обновление куба занимает много времени и требует участия специалистов из разных областей;
- необходимость в больших аппаратных ресурсах;
- чувствительность к нарушениям целостности данных.

Р. Турнье пишет: «ограничения, накладываемые OLAP-структурой, делают необходимостью для компаний поддерживать сложный ETL-процесс» [66]. Обработка данных для извлечения их из OLTP-систем и загрузка в кубы является сложным процессом, требующим специальных знаний. Например, используется моделирование размерностей Кимбалла [53], концепция разработки хранилища данных Инмона [49], гибридный подход Data Vault, объединивший достоинства схемы «звезда» и 3 нормальной формы, предложенный Дэном Линстедтом в 2000 г. [55, 26].

В последнее десятилетие взрывной рост количества данных и рост производительности вычислительных мощностей привел к развитию ряда решений, которые могут заменять классический OLAP-подход.

Что касается OLAP как концепции, то многие из упомянутых выше недостатков в большей степени связаны с применением именно тяжеловесных кубов, а не с самой концепцией OLAP. OLAP, по своей сути, направлен на оптимизацию возможностей изучения многомерных данных, предварительное формирование и вычисление соответствующих показателей

и размерностей для быстрого получения бизнес-информации [5]. Традиционная система хранения данных, такая как SQL, с трудом справляется с этим без большой работы узких специалистов.

На сегодняшний день ключевыми стратегиями реализации онлайн аналитической обработки данных, помимо традиционного OLAP-куба, основанного на реляционных базах данных, являются применение In-memory OLAP баз данных и технологии NoSQL, в частности, колоночных баз данных [58].

In-memory OLAP и NoSQL базы данных могут использоваться для онлайн аналитической обработки данных. Они могут продуктивно взаимодействовать с большими объемами данных и высокой плотностью данных. В частности, колоночные базы данных имеют архитектуру, которая оптимизирована для аналитических операций.

In-memory OLAP базы данных позволяют полностью или частично хранить данные в оперативной памяти. Это приводит к повышенной производительности в сравнении с базами данных, оптимизированными для дискового хранения, поскольку доступ к оперативной памяти более быстрый, а внутренние алгоритмы оптимизации проще. Согласно С.Д. Кузнецову, «хранение всех данных (или, по крайней мере, горячих данных) в основной памяти обеспечивает более высокую скорость обработки транзакций, что частично сглаживает негативные эффекты одновременно выполняемых аналитических запросов» [19].

Модель NoSQL возникла в ответ на потребность в эффективной обработке очень больших объемов данных. Технологии NoSQL, в частности колоночные базы данных, ориентированы на горизонтальное масштабирование и работу с данными, либо недостаточно структурированными, либо постоянно изменяющимися [52, с. 19]. Это нереляционные базы данных, не использующие SQL или использующие аналоги SQL.

В колоночных базах данных данные находятся в ячейках, сгруппированных в колонки, а не в строки данных. Чтение и запись происходит с использованием колонок, а не строк.

А.Н. Петрова пишет: «Преимущества хранения в столбцах заключаются в быстром поиске, доступе и агрегации данных. Реляционные базы данных хранят каждую строку как непрерывную запись на диске. Разные строки хранятся в разных местах на диске, в то время как колоночные базы данных хранят все ячейки, относящиеся к колонке, как непрерывную запись, что делает операции поиск и доступа быстрее» [24].

Различия в архитектуре реляционных и колоночных баз данных представлены на рисунке 12.

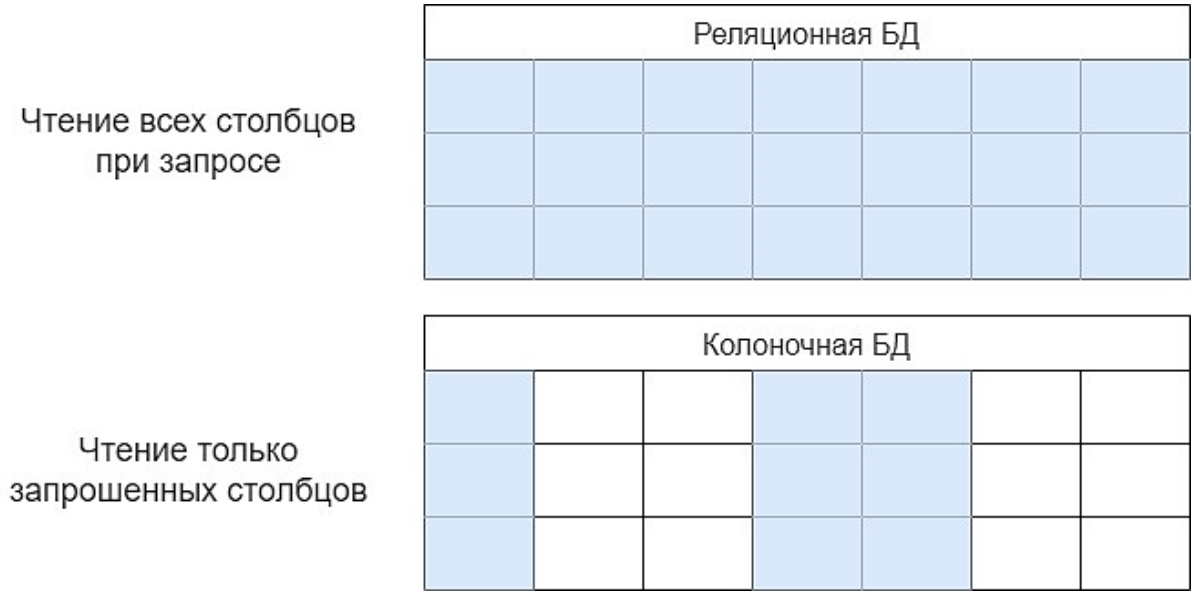

Рисунок 12 – Различия в архитектуре реляционных и колоночных баз данных

В то время как OLAP-кубы требуют загрузки в куб множества интересующих измерений, колоночные базы данных позволяют выполнять аналогичные рабочие нагрузки OLAP с одинаково высоким уровнем производительности без необходимости создавать новые кубы. Другими

словами, это база данных SQL, которая идеально подходит для рабочих нагрузок OLAP.

Это обусловлено также тем, что колоночные базы данных сжимаются лучше, чем реляционные базы данных. Похожие фрагменты данных хранятся в памяти намного эффективнее, чем разные фрагменты информации. Поскольку колоночные базы данных хранят столбцы данных, то есть значения с идентичными типами и сходными значениями, это обеспечивает более эффективное сжатие. В целом такое сжатие означает, что при выполнении агрегирующего запроса в память может быть загружено больше данных, что, в свою очередь, приводит к более быстрому выполнению общих запросов [22].

Преимущества хранения данных в колоночной форме заключаются в быстром поиске, доступе и агрегации данных. Поскольку колоночные базы данных эффективно сжимают похожие фрагменты данных в памяти, операции агрегации выполняются более быстро. Этот подход также позволяет освободить дополнительное пространство для сортировки и индексации данных внутри столбцов.

Конечным результатом всех этих свойств является то, что колоночные базы данных могут обеспечивать производительность, подобную OLAP-кубу, без необходимости явного проектирования и сборки кубов.

Однако производительность обновления в колоночной базе данных очень низкая [6]. Это приводит к тому, что многие современные колоночные базы данных ограничивают возможность обновлять данные после сохранения.

Также к недостаткам NoSQL баз данных можно отнести смягчение жестких требований к транзакциям, присутствующих в реляционных базах данных [29]. Основные принципы ACID гарантируют, что целостность и согласованность данных в реляционных хранилищах будет обеспечена даже при сбоях. Поскольку для ряда задач в строгом следовании ACID нет необходимости, NoSQL-базы чаще всего предлагают компромисс и

принципы BASE, которые являются менее строгими и позволяют достичь более высокой производительности [34, 25]. Отличия подходов представлены на рисунке 13.

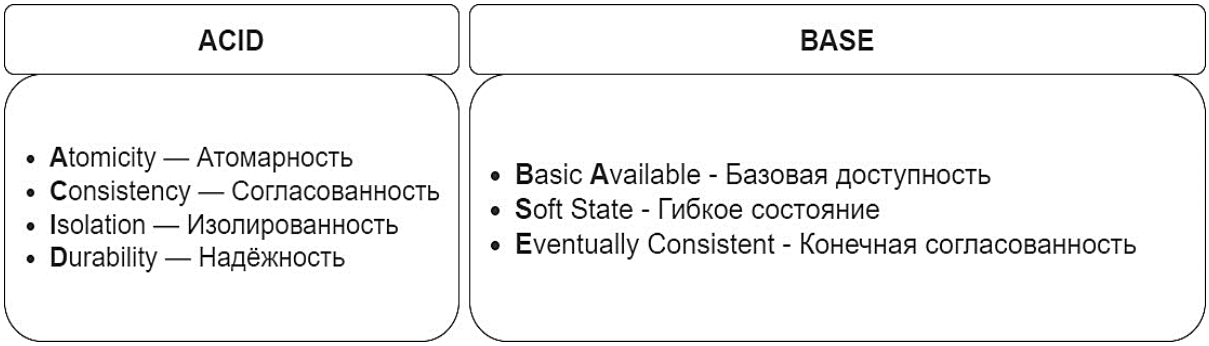

Рисунок 13 – Отличия требований к SQL и NoSQL базам данных

Выводы по второму разделу

В области аналитической обработки данных наблюдается сдвиг в сторону новых технологий, отдаляющий классический OLAP-подход на второй план.

Недостатки традиционного OLAP, такие как ограничения работы в реальном времени и сложность изменения структуры куба, делают его менее привлекательным в современном мире больших данных.

Новые подходы, такие как In-memory OLAP базы данных и технологии NoSQL, предоставляют более эффективные методы обработки данных, обеспечивая более гибкий и масштабируемый анализ.

Эти технологии позволяют работать с большими объемами данных и обеспечивают более высокую производительность по сравнению с традиционными методами.

3 Анализ бизнес-процессов компании

3.1 Характеристика деятельности предприятия. Анализ проблемных процессов в компании

ООО «Майндсет» занимается разработкой аналитических отчетов по заказу других компаний. Команда аналитиков и инженеров разрабатывает аналитические отчеты, отражающие ключевые метрики и показатели эффективности бизнеса клиентов, а также маркетинговые исследования на основе открытых данных. Отчеты помогают компаниям принимать обоснованные стратегические решения на основе данных.

Основные рабочие процессы компании:

- заказ и сбор требований. Этот процесс включает в себя взаимодействие с клиентами для понимания их потребностей и требований к аналитическим отчетам;

- анализ данных. Этот процесс включает в себя сбор, очистку, обработку и анализ данных для создания аналитических отчетов;

- разработка и создание отчетов. Этот процесс включает в себя разработку аналитических отчетов в соответствии с требованиями клиента.

В данный момент компания собирает данные из открытых источников в базу данных PostgreSQL. Затем данные проходят первичную обработку на языке SQL. Обработанные данные передаются в среду разработки с установленным языком программирования Python, где производится разведочный анализ данных и разрабатываются аналитические отчеты. Функциональная модель, отображающая структуру и функции системы процесса работы с данными, изображена на рисунках 14, 15.

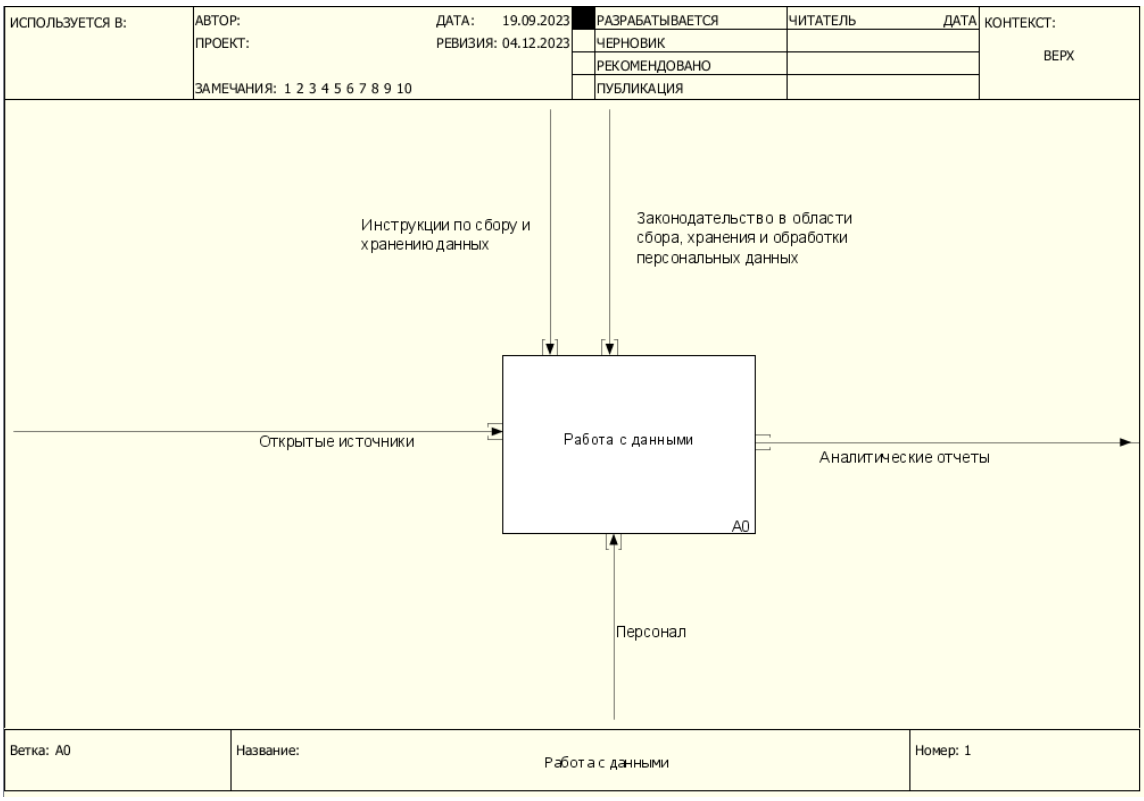

Рисунок 14 - Контекстная диаграмма IDEF0 «КАК ЕСТЬ» процесса работы с данными

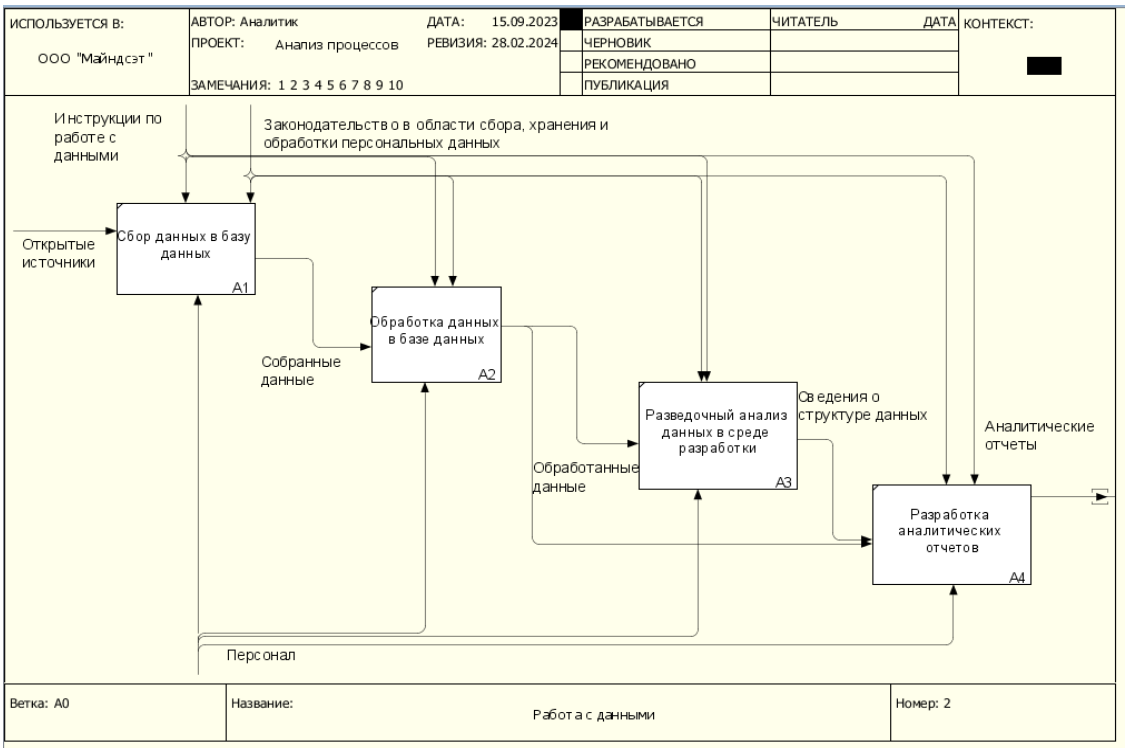

Рисунок 15 - Декомпозиция 1-го уровня контекстной диаграммы IDEF0 «КАК ЕСТЬ» процесса работы с данными

На рисунке 16 представлена кросс-функциональная карта, подробно описывающая обязанности подпроцессов в процессе.

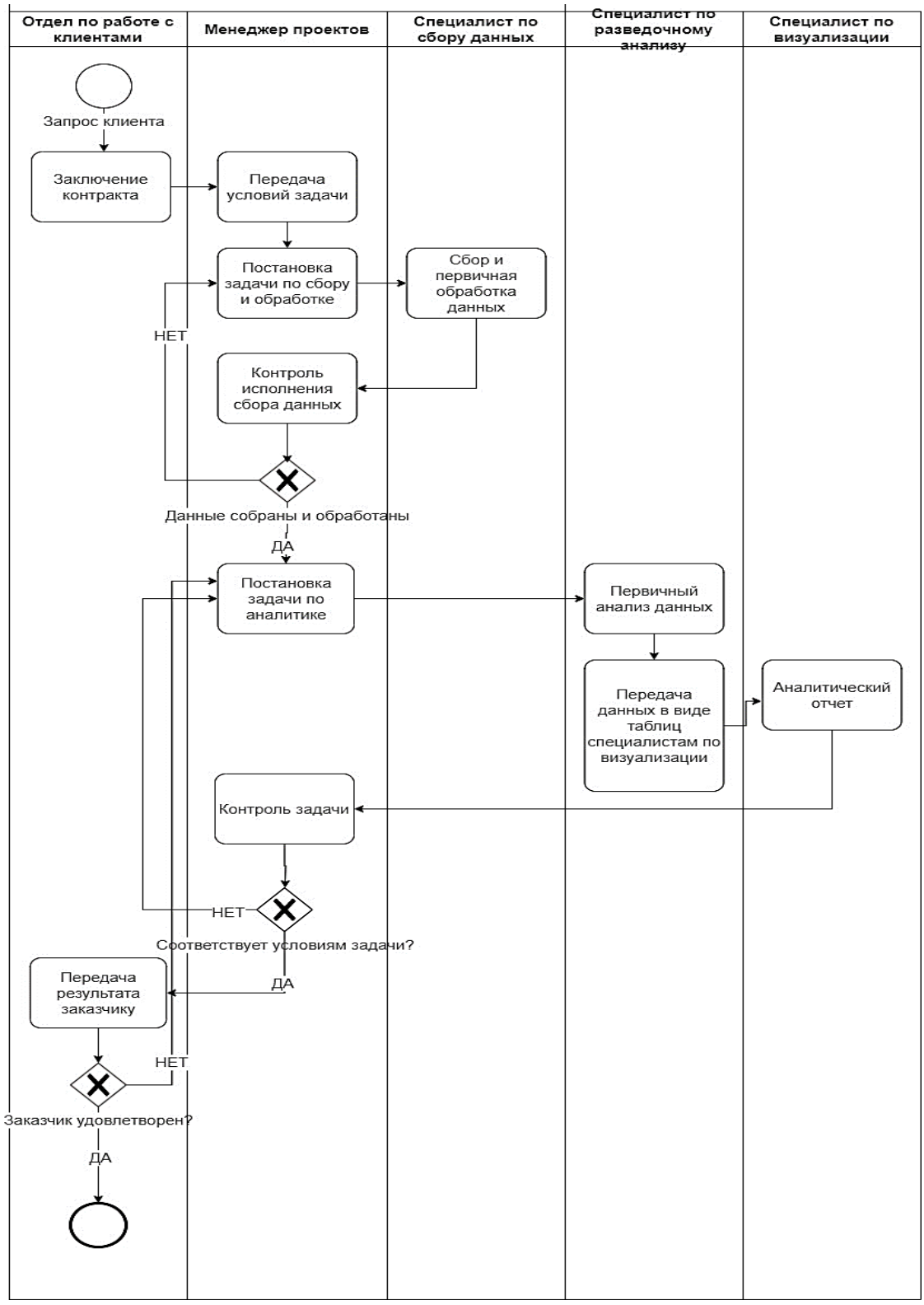

Рисунок 16 - Диаграмма плавательных дорожек «КАК ЕСТЬ» процесса работы с данными

Для достижения цели оптимизации процесса сбора и хранения данных и автоматизации задач при работе с данными компании необходимо сделать процесс обработки данных более простым и стандартизированным. В данный момент большие массивы данных не могут быть переданы для составления отчетов аналитикам в том виде, в котором они хранятся в базе данных. Предварительно эти данные необходимо обрабатывать средствами языков SQL и Python. Эти процессы требует больших временных ресурсов и специальных навыков.

Ключевые точки улучшения в работе с большими данными в данной компании включают:

– оптимизация ресурсов и автоматизация задач. Оптимизация ресурсов и автоматизация задач позволяют сократить затраты на управление и анализ данных, что помогает достичь цели снижения издержек и увеличения прибыли;

– обучение персонала. Успешное обучение персонала в области работы с данными и современными технологиями анализа данных помогает компании достичь цели повышения профессионального уровня сотрудников.

На настоящий момент в компании отлажены процессы автоматизированного сбора данных. Однако методы, используемые специалистами компании для хранения и обработки данных, можно расценивать как устаревшие. Учитывая развитие Big Data, компаниям необходимо внедрять эффективные ETL-процессы, которые автоматически обрабатывают и структурируют данные перед передачей в аналитическую среду. Улучшенная обработка данных повышает качество и надежность аналитических отчетов.

Возможной точкой улучшения для ООО «Майндсет» можно считать оптимизацию процесса сбора и хранения данных, автоматизацию задач при работе с данными.

3.2 Сбор требований. Выработка рекомендаций

Сбор требований осуществлялся на основе использования метода интервьюирования. В процессе интервьюирования были выявлены следующие основные требования, которым должна отвечать система:

– пользовательский интерфейс должен предоставлять возможности для создания, настройки и выполнения аналитических запросов к данным;

– система должна предоставлять возможность агрегировать и суммировать данные для быстрого выполнения аналитических запросов;

– пользователям должны быть доступны гибкие возможности создания отчетов и визуализации данных, включая диаграммы, графики, таблицы;

– система должна обеспечивать высокую производительность при выполнении аналитических запросов даже с большими объемами данных;

– интерфейс пользователя должен быть интуитивно понятным и легким в использовании для широкого круга пользователей;

– решение должно быть продуктом с открытым с исходным кодом для того, чтобы компания могла снизить политический риск.

Полный список требований представлен в Приложении А.

Основные данные, с которыми работает компания, — это данные по рынку страхования Российской Федерации из открытых источников с сайта Центрального банка России [23].

Основной задачей исследования является моделирование системы анализа указанных данных, которые могут быть использованы страховыми компаниями для анализа общих показателей рынка, последующего прогнозирования и использование результатов для принятия эффективных решений.

Данные по рынку страхования Российской Федерации за период с 2016 г. по 2 кв. 2023 г. содержат информацию о нескольких накопленных

количественных величинах (о премиях, выплатах и договорах), а также описывающих их качественных данных (справочники). Из этого следует, что предпочтительно использовать схему «звезда», как это показано на диаграмме классов, представленной рисунке 17.

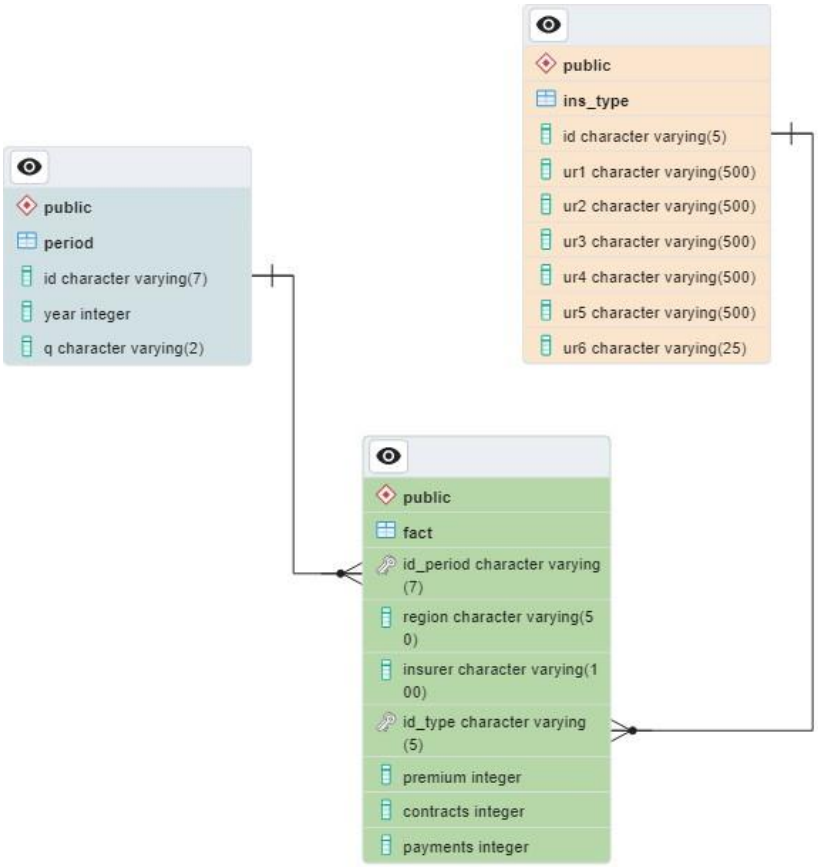

Рисунок 17 - Диаграмма классов OLAP-хранилища

В данной структуре данных таблица фактов «fact» содержит информацию о страховщике (поле «insurer»); регионе, где зафиксирован страховой факт (поле «region»); периоде, когда зафиксирован страховой факт (поле «period»); виде страхования (поле «id_type»); сумме премий, собранных страховщиком (поле «premium»); числе заключенных договоров (поле «contracts»); выплатах, произведенных страхователям (поле «payments»).

Таблица фактов связана с измерениями period, в котором данные о периоде разбиты на годы и кварталы, и ins_type, в котором расшифрована

информация о виде страхования. Данные о видах страхования обладают иерархической структурой, как показано на рисунке 18.

На рисунке 18 представлена не полная иерархия, в реальности она имеет 5 уровней.

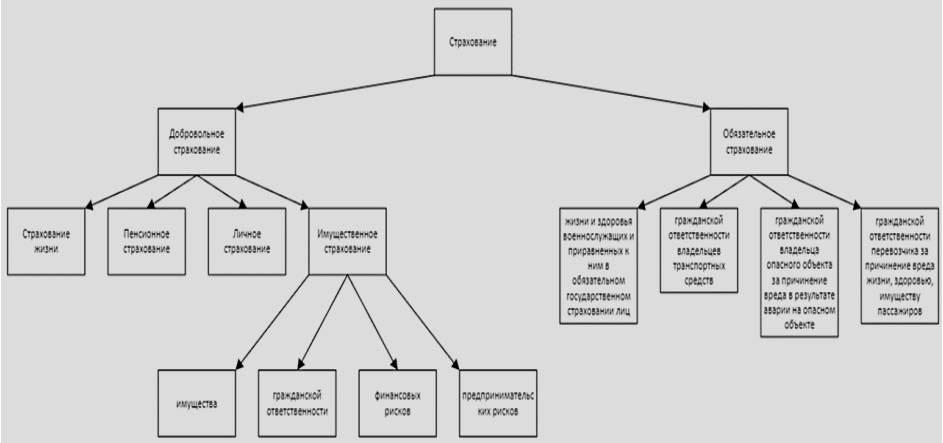

Рисунок 18 – Иерархическая структура видов страхования

Структура данных требует, чтобы решение поддерживало создание и обработку иерархических структур.

На основе проведенных исследований, анализа требований компании и данных, которые компания планирует анализировать, была выработана рекомендация рассмотреть в качестве альтернативы внедренным технологиям использование технологии многомерного анализа данных OLAP.

Базы данных OLTP, которую сейчас использует компания для анализа данных, оптимизирована для обработки операций транзакционной обработки, а не для аналитических запросов.

Так как связанные данные хранятся в нескольких таблицах, анализ требует использование медленных, неэффективных и сложных для составления SQL-запросов с использованием SQL-команды JOIN, которую используют, чтобы объединять строки таблиц на основе связующего столбца между ним.

Проведение анализа с помощью SQL-команды JOIN в OLTP-базе может привести к избыточному использованию ресурсов сервера и снижению производительности для других операций транзакционной обработки.

При этом, OLAP позволит анализировать сводные данные по различным измерениям, таким как регион, период и вид страхования, и осуществлять сводные расчеты для получения ценной информации. Учитывая иерархическую структуру данных о видах страхования, OLAP может позволить быстро обрабатывать и анализировать такие данные, позволяя пользователям проводить анализ на разных уровнях детализации.

Однако классические OLAP-решения обладают рядом недостатков. На сегодняшний день ключевыми стратегиями реализации онлайн аналитической обработки данных являются применение In-memory OLAP баз данных и технологии NoSQL, в частности, колоночных баз данных.

Колоночные базы данных медленно работают на запись и обновление данных. Так как для компании важно иметь возможность обновлять данные для аналитики, вариант построения системы аналитики на основе колоночных баз данных был отклонен.

Наиболее предпочтительным вариантом для компании представляется использование In-memory OLAP-продукта Atoti, так как он является современным решением с открытым исходным кодом. Atoti предоставляет гибкую модель данных с возможностью быстрого создания и изменения структуры кубов данных без необходимости перезагрузки данных. Atoti также поддерживает создание вычисляемых мер и иерархии.

Atoti предполагает работу в Python, что соответствует специализации тех работников, которые сейчас производят анализ в компании. Однако Atoti, при этом, убирает этап разворачивания куба в отдельном инструменте. По сути, Atoti позволяет пройти весь этап анализа данных, задействуя только один инструмент. Это позволяет ограничить набор требований к сотрудникам только знанием языка программирования Python.

Так как в данный момент сотрудникам необходимо знание языка запросов SQL на очень высоком уровне для того, чтобы собирать из большого объема данных объекты, пригодные для анализа, это могло бы стать существенным упрощением работы и позволило бы сократить затраты на специалистов с узкой специализацией. Также, в отличие от классического OLAP-куба, куб в Atoti не предполагает использования языка многомерных выражений MDX.

Выводы по третьему разделу

Для оптимизации процесса работы с данными компании необходимо сделать процесс обработки данных более простым и стандартизированным.

Важные направления улучшения включают эффективный сбор и хранение данных, оптимизацию ресурсов и автоматизацию задач, а также обучение персонала. Компании следует обратить внимание на современные методы обработки и анализа данных. В целом, улучшение процесса работы с данными поможет компании повысить эффективность и качество предоставляемых услуг, а также добиться конкурентного преимущества на рынке аналитических услуг.

В результате анализа требований компании к системе анализа данных и особенностей данных, с которыми она работает, рекомендуется рассмотреть внедрение технологии многомерного анализа данных OLAP. Это обусловлено необходимостью обеспечить высокую производительность при выполнении аналитических запросов даже с большими объемами данных, а также гибкими возможностями создания отчетов и визуализации данных. Наиболее предпочтительным вариантом для компании представляется использование In-memory OLAP-продукта Atoti, так как он соответствует требованиям по производительности, гибкой модели данных и открытому исходному коду.

4 Реализация системы OLAP в Atoti

4.1 Моделирование системы OLAP в Atoti

На основе проведенных исследований была выработана рекомендация рассмотреть в качестве альтернативы внедренным технологиям использование технологии многомерного анализа данных, предоставляемую Atoti.

Платформа Atoti – это библиотека с открытым исходным кодом, в которой реализована возможность интегрировать полноценный OLAP-куб в среду Python. Это технология, позволяющая напрямую запрашивать данные, создавать отчеты и получать аналитические данные «на лету». Специалисты по визуализации данных с помощью Excel или приложения Atoti могут обращаться к OLAP-кубам и получать многомерные сечения в виде сводных таблиц.

На рисунке 19 представлено сравнение архитектуры классического хранилища данных Microsoft Analysis Services и архитектуры Atoti [36].

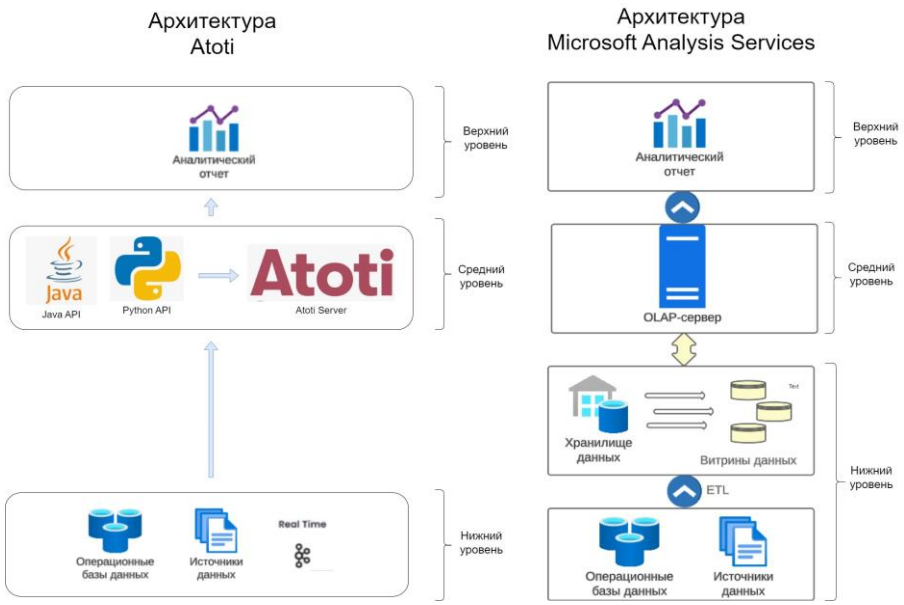

Рисунок 19 - Сравнение архитектуры Microsoft Analysis Services и архитектуры Atoti

После внедрения решения появляется возможность устранить этап предварительного анализа и обработки данных, что сделает процесс работы с данными менее длительным и позволит значительно уменьшить число ошибок на основе человеческого фактора. На рисунках 20, 21 показано, что система аналитики с включением классического OLAP-куба требует участия большего количества специалистов.

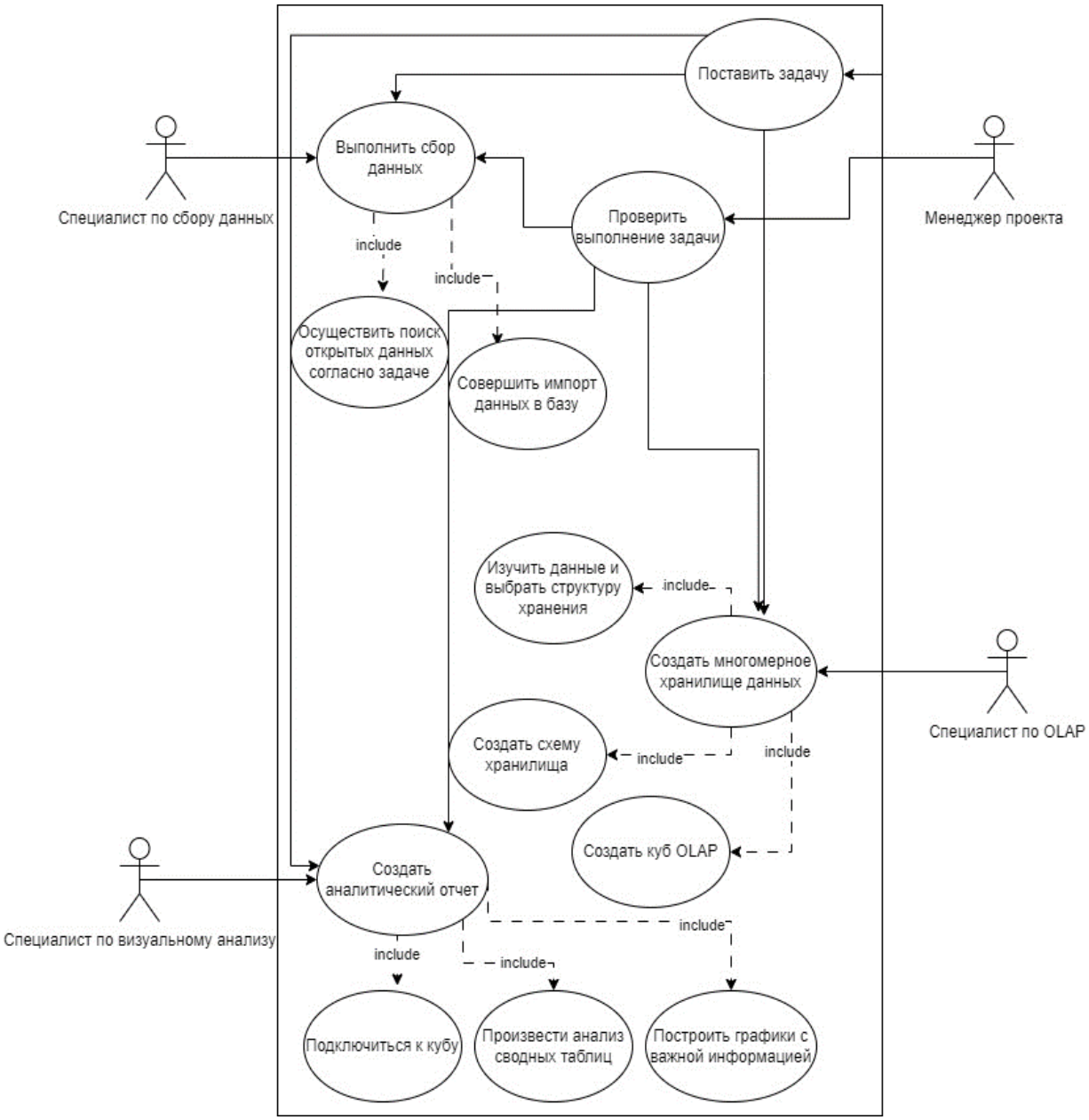

Рисунок 20 – Диаграмма вариантов использования системы с включением Microsoft Analysis Services

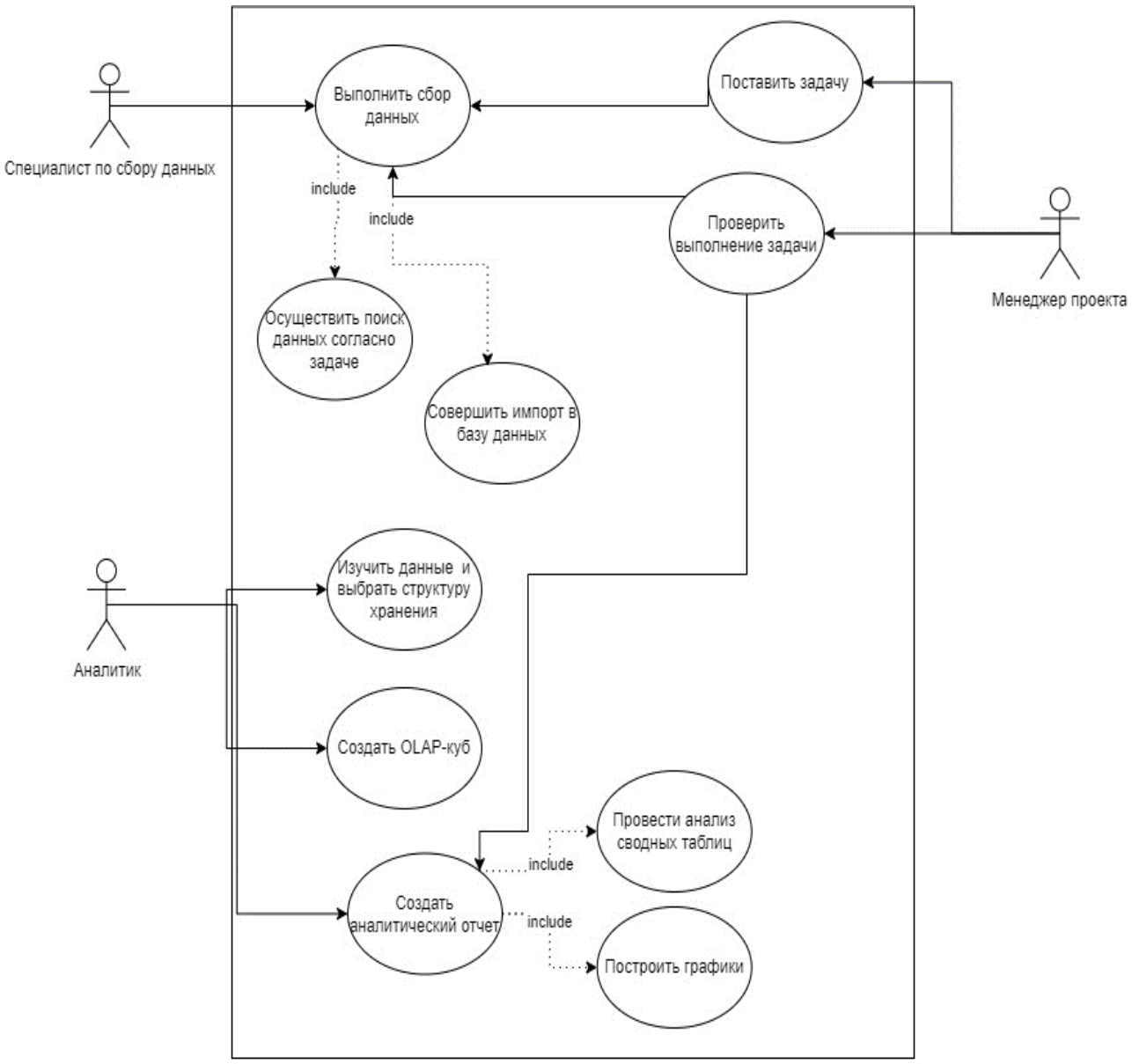

Рисунок 21 – Диаграмма вариантов использования системы с включением Atoti

Таким образом, предложенный подход предоставляет возможность также сократить затраты на работу узких специалистов.

4.2 Реализация системы

Сервис Atoti позволяет собирать, анализировать и визуализировать данные в среде программирования Python, а именно в ноутбуках Jupyter. Jupyter — это интерактивная среда для работы с кодом. В ней можно

создавать блоки кода, которые можно исполнять по отдельности и визуализировать результаты.

Jupyter поддерживает несколько языков программирования, включая Python, R и Julia. Ноутбуки можно интегрировать с различными инструментами и источниками данных.

Также в Jupyter можно не только отображать код, но и текстовые вставки, картинки, визуализировать графики с помощью специализированных библиотек.

В библиотеку Atoti встроен модуль, позволяющий интегрироваться с популярной библиотекой Pandas. В частности, данные Atoti могут хранится в структуре Pandas DataFrame, которая является общеприменимой для работы с табличными и многомерными данными в среде программирования Python.

Для построения системы анализа данных на основе Atoti в Jupyter необходимо установить соответствующие библиотеки.

Поскольку данные компании хранятся в базе данных PostgreSQL, для их загрузки в ноутбук необходимо использовать JDBC-драйвер, позволяющий получать данные из базы данных. В методе atoti.session.read_sql() библиотеки Atoti необходимо прописать строку подключения к базе данных. Этот метод возвращает данные в виде Pandas DataFrame.

Код на рисунке 22 читает данные из базы данных PostgreSQL (три разных таблицы: таблица фактов и две таблицы измерений) в Atoti, используя метод read_sql() сессии Atoti.

```
fact = session.read_sql(
    """SELECT id_period, region, insurer, id_type, premium, contracts, payments
    FROM public.fact""",
    url="jdbc:postgresql://host:5432/db?user=user&password=password",
    table_name="fact"
)

ins_type = session.read_sql(
    """SELECT * FROM public.ins_type""",
    url="jdbc:postgresql://host:5432/db?user=user&password=password",
    table_name="ins_type"
)

period = session.read_sql(
    """SELECT * FROM public.period""",
    url="jdbc:postgresql://host:5432/db?user=user&password=password",
    table_name="period"
)
```

Рисунок 22 – Получение данных из базы данных PostgreSQL в Atoti

В этих запросах метод в качестве аргумента передается SQL-запрос для выборки данных из таблицы. В частности, запрос для получения данных для таблицы фактов выбирает столбцы id_period, region, insurer, id_type, premium, contracts и payments из таблицы public.fact в базе данных PostgreSQL. table_name — это имя таблицы, которое будет использоваться для создания временной таблицы в Atoti для хранения данных.

Ключевая особенность Atoti как BI-сервиса – это возможность создания многомерного куба данных. При этом, числовые столбцы будет определены как меры, категориальные столбцы - как измерения куба. На рисунке 23 показан процесс присоединения измерений к таблице фактов (.join()), после чего, куб можно визуализировать с помощью метода cube.schema.

```
fact.join(ins_type, fact['id'] == ins_type['id'])
fact.join(period, fact["period"] == period["id"])
```

```
cube = session.create_cube(fact)
```

```
cube.schema
```

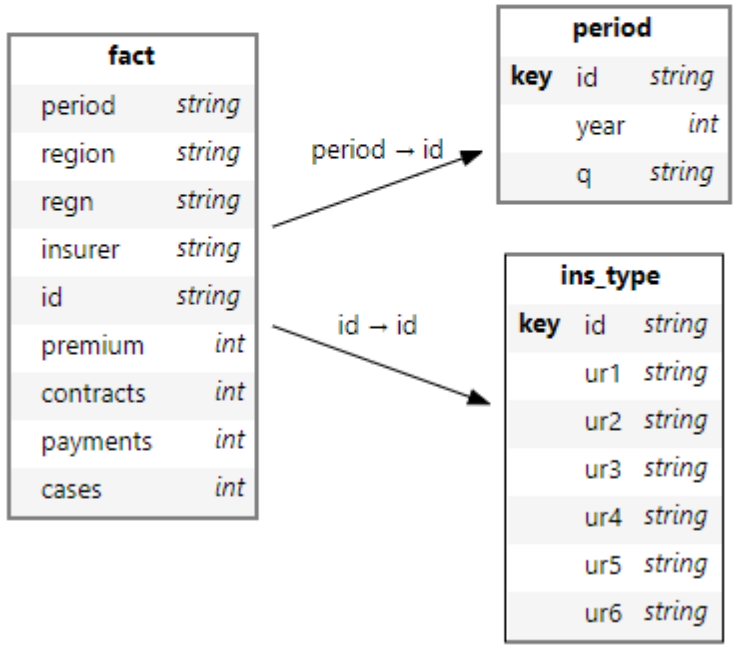

Рисунок 23 – Создание куба данных в Atoti

Как видно из рисунка 23, создание куба в Atoti не требует написания сложного кода. Создатели продукта придерживаются концепции создания прототипа, то есть, куб быстро создается и собирается, а затем обогащается дополнительными вычислениями и агрегатами. Это позволяет протестировать большое количество вариаций куба. Внесение изменений в куб не является сложной и трудоемкой задачей.

Преимущество Atoti перед классическим OLAP-подходом заключается в том, что параметры куба можно изменять после сборки куба данных: добавлять вычисляемые меры и изменять элементы. Этот подход позволяет изменять куб в реальном времени. Так, в среде JupyterLab можно настроить иерархии для измерений уже после того, как куб был собран, как это показано на рисунке 24. После этого действия куб можно не пересобирать, изменения уже внесены в его структуру.

```
h["ins_type"] = {
    "ur1": ins_type["ur1"],
    "ur2": ins_type["ur2"],
    "ur3": ins_type["ur3"],
    "ur4": ins_type["ur4"],
    "ur5": ins_type["ur5"],
    "ur6": ins_type["ur6"],
}
```

```
h["period_hier"] = {
    "year": period["year"],
    "quater": period["q"]
}
```

Рисунок 24 – Изменений измерений в Atoti

В Atoti также доступно вычисление пользовательских мер, в том числе сложных мер с использованием встроенных функций Atoti, таких как ранжирование (atoti.rank()) и агрегирование (atoti.total()), как это показано на рисунке 25. Агрегатные функции, встроенные в Atoti, позволяют произвести сложные вычисления без необходимости использовать сложные конструкции. При этом, настройка пользовательских мер также не требует ручного обновления куба данных.

```
m["Сумма премий млн руб."] = m["premium.SUM"]/1000
m["Сумма выплат млн руб."] = m["payments.SUM"]/1000
m['Число полисов шт'] = m['contracts.SUM']
```

```
m["Ранг по премиям"] = tt.rank(m["premium.SUM"], h['insurer'], ascending=False)
```

```
m["Средняя премия руб."] = m['premium.SUM']/m['contracts.SUM']*1000
```

```
m['Доля страховой компании'] = m["premium.SUM"]/tt.total(m["premium.SUM"], h['insurer'])
```

Рисунок 25 – Пример вычисляемых меры в Atoti

Явным преимуществом Atoti является отсутствие необходимости использовать в работе язык запросов и многомерные выражения. На рисунке 26 показано, как многомерные выражения формируются автоматически при работе в интерактивной среде Python.

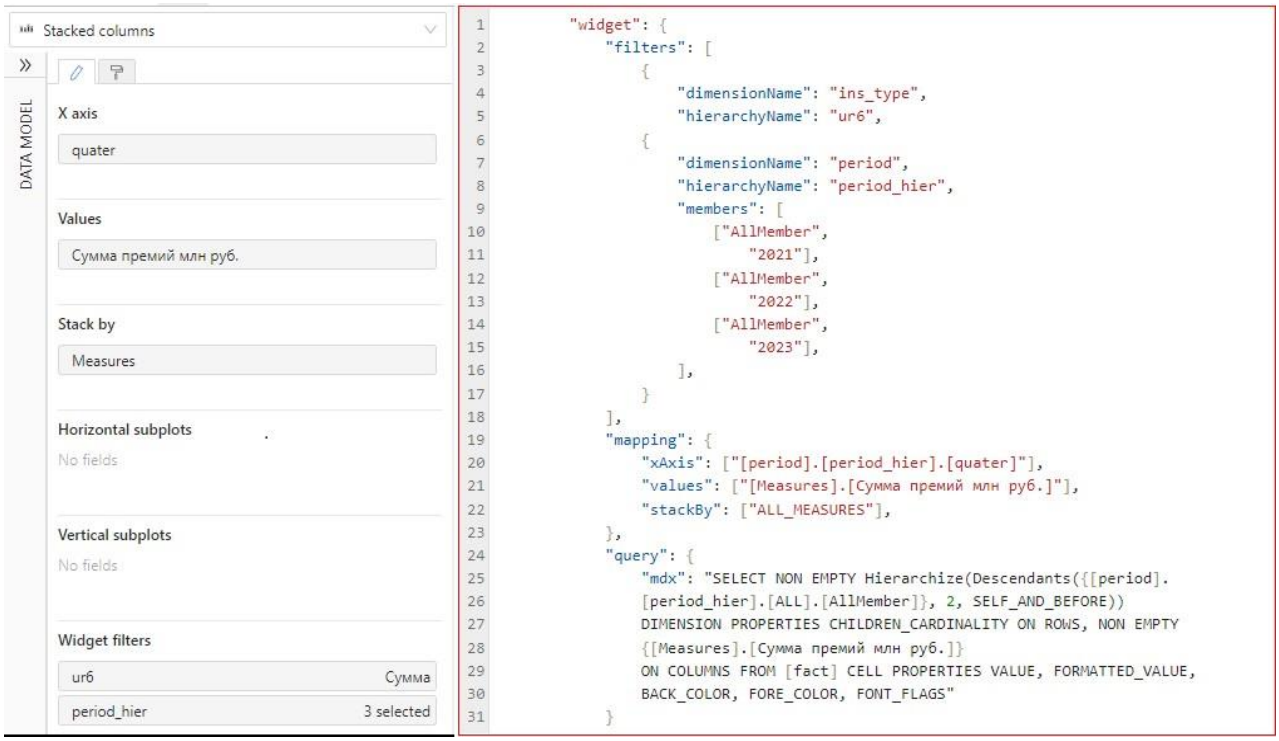

Рисунок 26 – Автоматическое формирование MDX-запроса в Atoti

Atoti предоставляет различные возможности визуализации данных прямо в JupyterLab, что позволяет аналитикам и разработчикам проводить анализ данных непосредственно в интерактивной среде. На рисунке 27 представлен график, сформированный на основе drag-and-drop запроса, то есть путем перетаскивания и отпускания элементов интерфейса для создания запроса данных. Такой способ работы позволяет пользователям интуитивно выбирать и анализировать данные, не используя язык запросов напрямую.

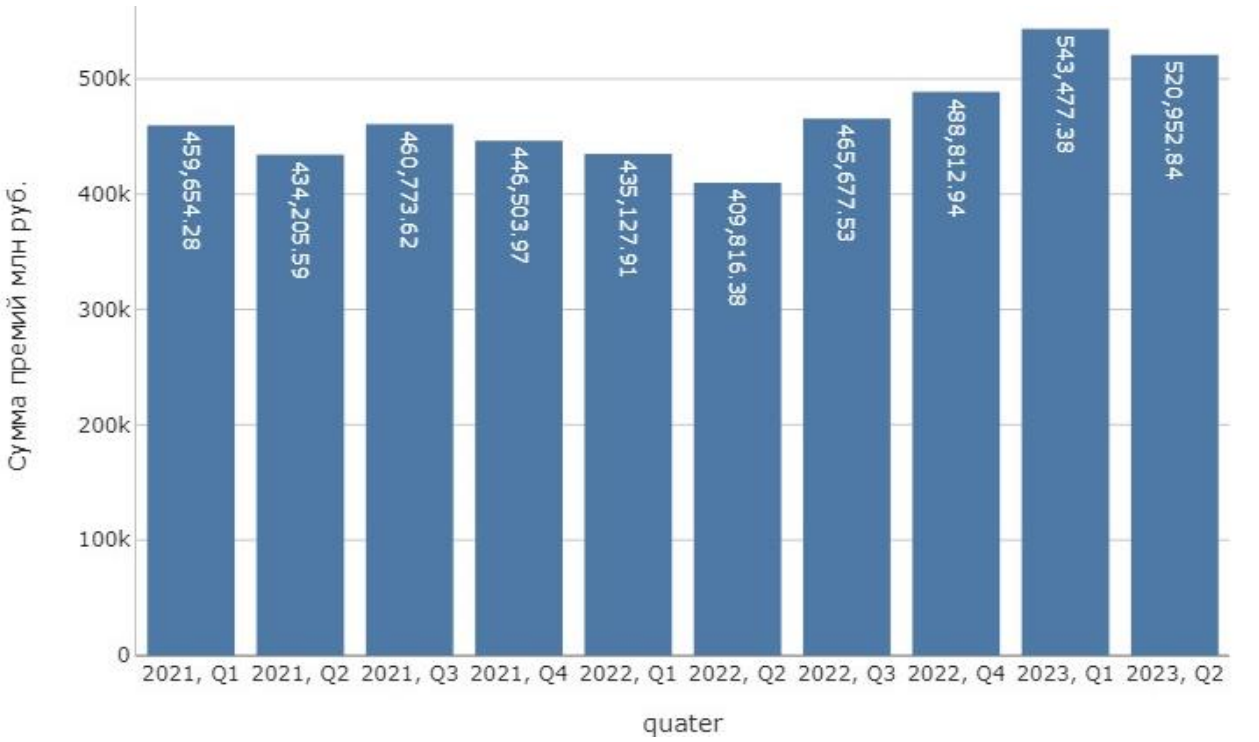

Рисунок 27 – Визуализация запроса в Atoti

Приложение для визуализации Atoti является удобным инструментом с большим количеством функций, позволяющим строить графики без использования кода. На рисунках ниже представлены основные функции, востребованные аналитиками.

На рисунке 28 можно увидеть, что в Atoti есть поддержка сложных вычисляемых фильтров. На графике будут отфильтрованы компании, которые не входят в топ-10 компаний по сумме выручки в указанный период.

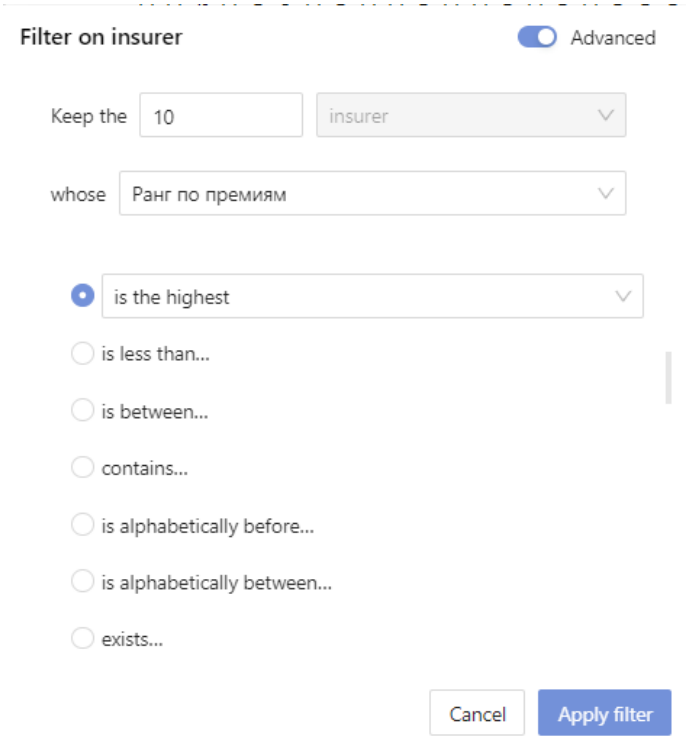

Рисунок 28 – Использование сложных фильтров

На рисунке 29 показана возможность условного форматирования ячеек: более прибыльные компании будут окрашены зеленым, остальные – красным.

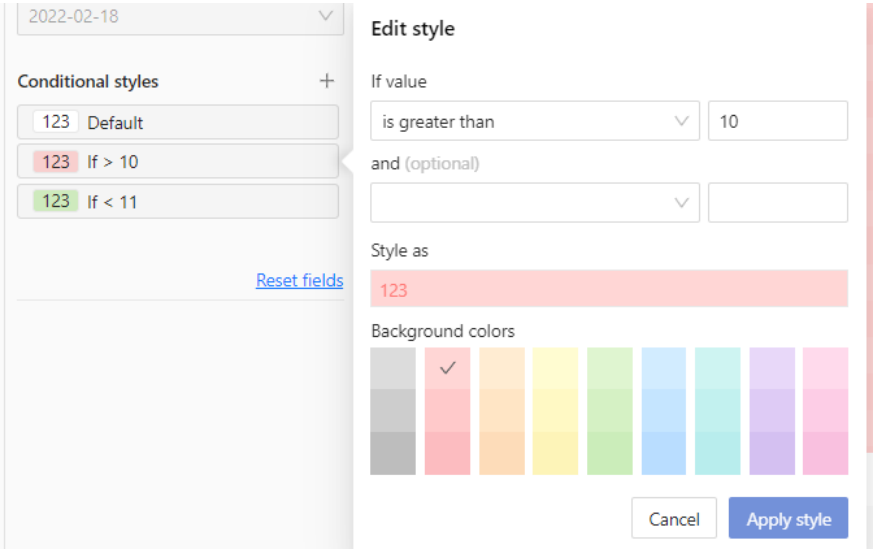

Рисунок 29 – Возможность настройки условного форматирования ячеек

На рисунке 30 представлены возможности кастомизации графика: форматирование названий осей, легенды, подписи данных, цветов элементов графика.

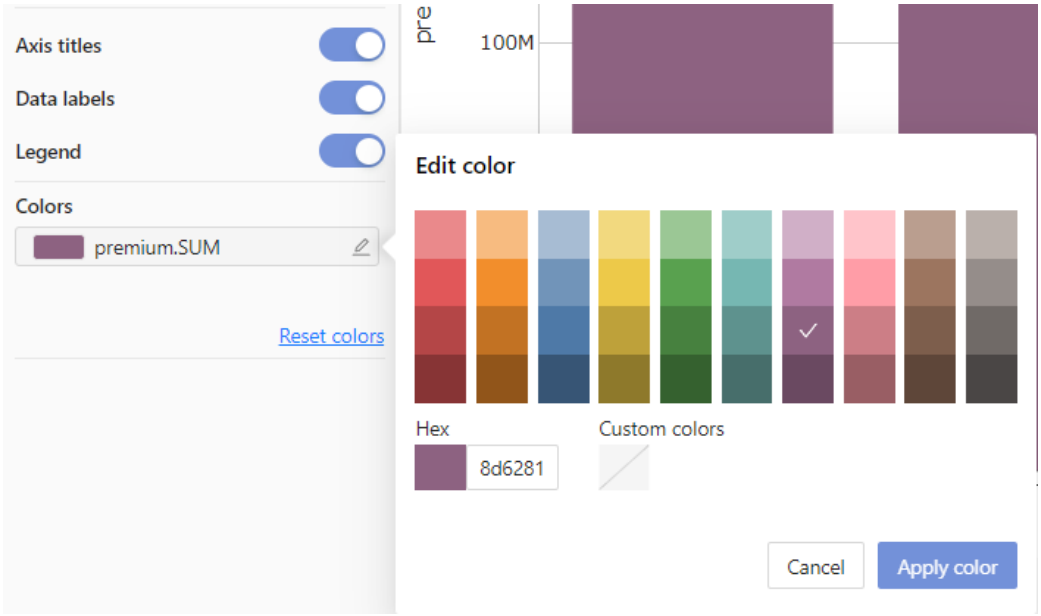

Рисунок 30 – Возможности кастомизации графиков

4.3 Оценка эффективности решения

4.3.1 Методика оценки эффективности

Для произведения оценки эффективности внедренного решения была разработана методика всесторонней оценки производительности и бизнес-эффективности систем анализа данных. Она учитывает следующие аспекты:

Критерии оценки производительности:

- сравнение времени отклика систем на стандартный набор запросов;
- оценка надежности системы при длительной нагрузке.

Критерии оценки функциональности:

- сравнение времени отклика систем на стандартный набор запросов;
- анализ способностей интеграции системы с другими инструментами;
- оценка интуитивности интерфейса системы для пользователя.

Критерии оценки бизнес-эффективности:

- анализ затрат на внедрение и поддержку системы;
- исследование влияния системы на скорость и качество бизнес-процессов.

Методы для проведения оценивания по данной методике:

- разработка и применение стандартных бизнес-сценариев для имитации реальной рабочей нагрузки;
- проведение опросов среди пользователей для оценки удобства использования и влияния на бизнес.

Эта методика представляет собой комплексный подход к оценке OLAP-систем. Она учитывает как технические аспекты, так и их воздействие на бизнес. Важность такого подхода особенно ощутима для малых компаний, где каждое решение должно быть обосновано и приносить реальную пользу. Мы рассматриваем не только скорость и надежность системы, но и то, как она влияет на работу компании в целом.

Сценарии тестирования, которые приближены к реальным рабочим условиям, станут реальным подтверждением эффективности решения. Это позволит подтвердить практическую применимость в повседневных задачах компании.

Согласно методике, необходимо сравнить производительность запросов и провести нагрузочное тестирование в Atoti, реляционной таблице и классическом OLAP-кубу Microsoft Analysis Services.

4.3.2 Сборка OLAP-куба в Visual Studio

В целях сравнения эффективности внедренного решения с классическими подходами к анализу многомерных данных создадим OLAP-куб на основе имеющихся данных в Microsoft Analysis Services с помощью Visual Studio. Visual Studio - это интегрированная среда разработки от Microsoft, предназначенная для создания различных типов приложений, включая проекты многомерных данных.

Структура куба в Visual Studio представлена на рисунке 31.

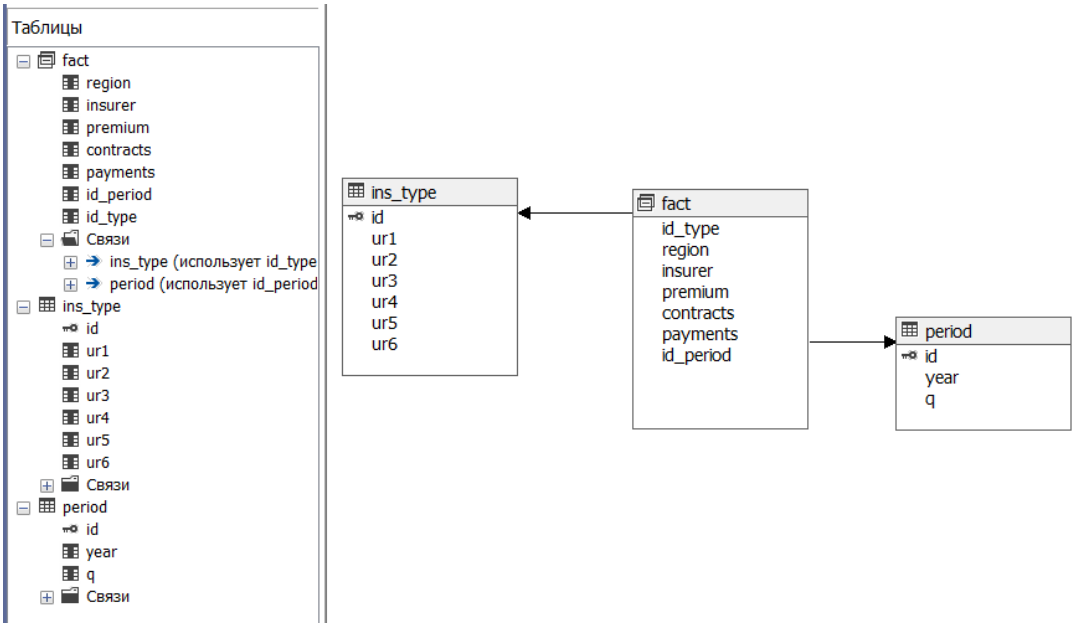

Рисунок 31 – Структура OLAP-куба в Visual Studio

Analysis Services также предоставляет возможность определения иерархий, как показано на рисунке 32, и вычисление пользовательских мер, как показано на рисунке 33.

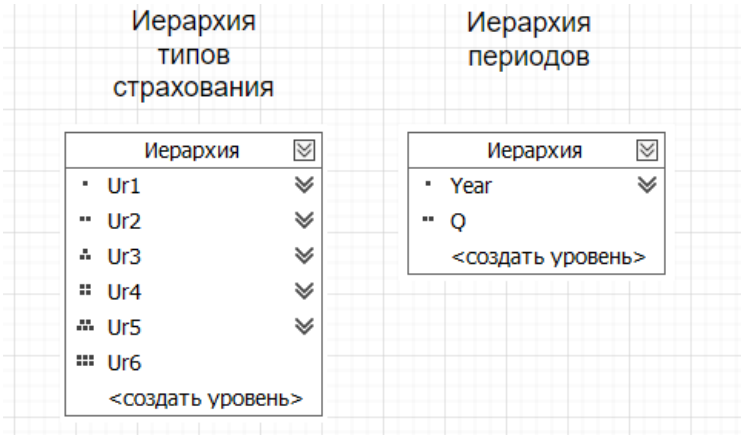

Рисунок 32 – Определение иерархий в Visual Studio

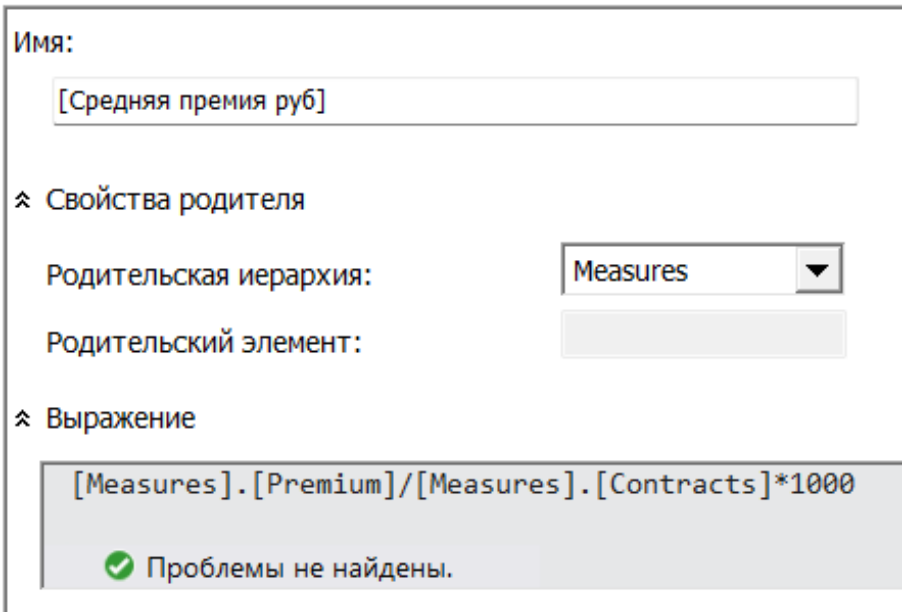

Рисунок 33 – Расчет вычисляемых мер в Visual Studio

4.3.3 Тестирование производительности запросов

Запрос, используемый для тестирования производительности, выполняет агрегацию суммы собранных премий по периодам (годам-кварталам) для видов страхования, относящихся к категории "Сумма" за определенные годы (2021, 2022, 2023). Эта метрика может быть полезна для анализа динамики суммы премий и принятия стратегических решений на основе этих данных.

CPU time – это время, которое процессор фактически затрачивает на выполнение кода программы. Оно измеряется в процессорных тактах или циклах и показывает общее количество времени, которое процессор был занят выполнением кода программы.

На рисунках 34, 35, 36 представлен данный запрос и его CPU time в исследуемых продуктах.

```
EXPLAIN ANALYZE
SELECT fact.period, SUM(premium) FROM fact
JOIN period
on fact.period = period.id
JOIN ins_type
on fact.id = ins_type.id
WHERE period.year in (2021, 2022, 2023) and ins_type.Ur6 = 'Сумма'
GROUP BY fact.period
```

1	2021-Q1	459654274
2	2021-Q2	434205606
3	2021-Q3	460773626
4	2021-Q4	446503982
5	2022-Q1	435127915
6	2022-Q2	409816397
7	2022-Q3	465677522
8	2022-Q4	488812939
9	2023-Q1	543477395
10	2023-Q2	520952844

Execution Time: 715.875 ms

Рисунок 34 – CPU выполнения запроса в базе данных PostgreSQL, выраженное в мс

ObjectName	Error	ClientProcessID	SPID	CPUTime
			6050	0
			6050	
			6050	0
			6050	
		316	4199	
			4199	0
			4199	0
		316	4199	0

```
SELECT
    { [Measures].[Premium] } ON COLUMNS,
    { [Period].[Id].members } ON ROWS
FROM [CBR2 1 1]
WHERE ( [Ins Type].[Ur6].[Сумма],
          { [Period].[Year].[2021],
            [Period].[Year].[2022],
                [Period].[Year].[2023] } )
```

	Premium
All	370035204
2021-Q1	459654274
2021-Q2	434205606
2021-Q3	460773626
2021-Q4	446503982
2022-Q1	435127915
2022-Q2	409816397
2022-Q3	465677522
2022-Q4	488812939
2023-Q1	543477395
2023-Q2	520952844

Рисунок 35 – CPU выполнения запроса в Microsoft Analysis Services, выраженное в мс

```
from time import process_time

t1_start = process_time()/1000

cube.query(
          cube.measures['Сумма премий млн руб.'],
          levels=[lvl['quater']],
          filter=(lvl['year'].isin('2021','2022','2023')) &
                 (lvl['ins_type', 'ur6', 'ur6'] == 'Сумма')
          )

t1_stop = process_time()/1000

print("CPU в миллисекундах:", t1_stop-t1_start)
```

Сумма премий млн руб.

year	quater	
2021	**Q1**	459,654.28
	Q2	434,205.59
	Q3	460,773.62
	Q4	446,503.97
2022	**Q1**	435,127.91
	Q2	409,816.38
	Q3	465,677.53
	Q4	488,812.94
2023	**Q1**	543,477.38
	Q2	520,952.84

```
CPU в миллисекундах: 1.5625000000000014e-05
```

Рисунок 36 – CPU выполнения запроса в Atoti, выраженное в мс

Как можно видеть из рисунков 34, 35, 36, использование OLAP позволяет выполнить указанный запрос намного быстрее. Это происходит за счет того, что данные в OLAP-кубе предварительно агрегируются и для различных комбинаций измерений.

4.3.4 Нагрузочное тестирование решений

Для проведения нагрузочного тестирования был разработан бизнес-сценарий, направленный на анализ эффективности страховых продуктов. Бизнес-сценарий представлен в Приложении Б.

Для проведения нагрузочного тестирования на основе разработанного бизнес-сценария использованы реальные рабочие процессы компании, включая создание тестовых сценариев, которые имитируют типичные операции, такие как запросы на получение данных, и использование данных, которые отражают реальные объемы операций страховой компании, для оценки способности системы справляться с высокой нагрузкой.

Для проведения тестирования используется специализированное приложение JMeter, которое позволяет проверить пропускную способность и стабильность системы.

Как указано на рисунке 37, тест-план в JMeter включает:

- использование SQL-запросов, описанных в бизнес-сценарии, которые позволяют оценить различные аспекты работы системы;
- моделирование работы 60 пользователей, которые одновременно отправляют запросы к системе;
- моделирование периода постепенного увеличения нагрузки. Оно составляет 60 с, что означает, что все 60 пользователей начнут отправлять запросы в течение одной минуты. Каждый из пользователей повторяет набор запросов 5 раз.

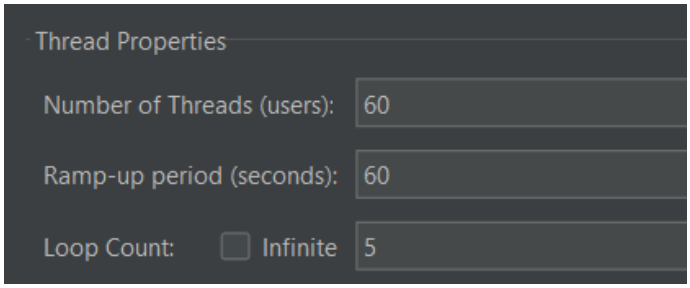

Рисунок 37 – Элемент тест-плана в JMeter

Тест-план позволяет имитировать реальную нагрузку на систему и анализировать её способность справляться с высокой конкурентностью и интенсивностью операций.

На рисунках 38, 39, 40 представлены отчеты с агрегированными данными о проведении тестирования. Пункт «Throughput» — это описание пропускной способности, выраженной в запросах в с, для каждого запроса и в целом для теста в строке итогов «Total».

Label ↑	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received KB/sec	Sent KB/sec	Avg. Bytes
Avg Contracts	70	1198	505	2716	572,60	0,00%	1,2/sec	6,29	0,00	5256,0
Max Premium	48	906	403	1947	383,23	0,00%	48,2/min	0,29	0,00	374,0
Measures by Period	66	1455	613	3572	726,42	0,00%	1,1/sec	0,24	0,00	227,0
Payments by InsType	65	1498	541	3770	827,92	0,00%	1,0/sec	1,77	0,00	1756,0
Property Contracts	51	894	432	2041	391,00	0,00%	51,2/min	0,04	0,00	48,0
TOTAL	300	1221	403	3770	673,38	0,00%	4,8/sec	8,02	0,00	1724,8

Рисунок 38 – Оценка пропускной способности для нагрузки в Postgres

Label ↑	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughput	Received K...	Sent KB/...	Avg. Byt...
Avg Contracts	50	5	3	11	1,98	0,00%	52,6/min	19,22	0,79	22443,0
Max Premium	68	5	3	23	2,68	0,00%	1,2/sec	18,54	1,10	16477,0
Measures by Period	60	5	3	13	1,93	0,00%	1,0/sec	10,33	1,03	10408,0
Payments by InsType	54	5	3	15	2,31	0,00%	54,9/min	12,09	0,82	13520,0
Property Contracts	68	5	3	13	1,86	0,00%	1,2/sec	17,64	1,06	15146,0
TOTAL	300	5	3	23	2,20	0,00%	5,1/sec	76,55	4,73	15423,6

Рисунок 39 – Оценка пропускной способности
для нагрузки в Microsoft Analysis Services

Label ↑	# Samples	Average	Min	Max	Std. Dev.	Error %	Throughp...	Received K...	Sent KB/s...	Avg. Bytes
Avg Contracts	76	26	3	1175	146,14	0,00%	1,3/sec	17,83	1,10	14093,0
Max Premium	48	4	2	9	1,79	0,00%	50,1/min	0,78	0,66	952,0
Measures by Period	69	4	2	9	1,66	0,00%	1,2/sec	4,34	0,95	3776,0
Payments by InsType	44	4	2	10	1,85	0,00%	46,3/min	2,78	0,59	3692,0
Property Contracts	63	9	2	362	44,75	0,00%	1,1/sec	3,96	0,83	3666,0
TOTAL	300	11	2	1175	76,96	0,00%	5,1/sec	29,29	4,06	5902,4

Рисунок 40 – Оценка пропускной способности для нагрузки в Atoti

На рисунках 41, 42, 43 представлены графики изменения ожидания отклика системы при увеличении нагрузки для каждого запроса.

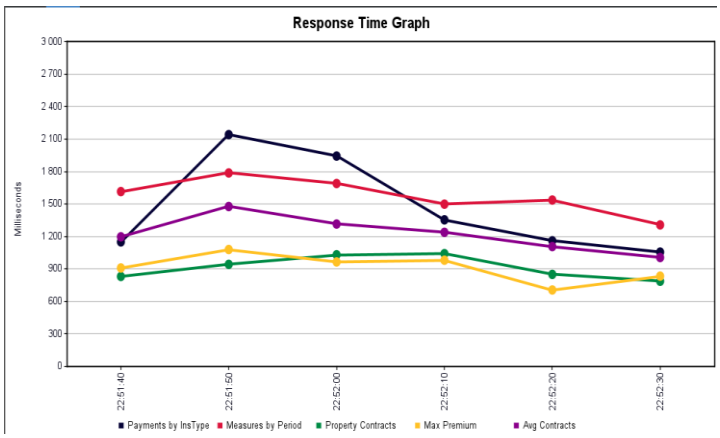

Рисунок 41 – Время отклика системы в Postgres, выраженное в мс

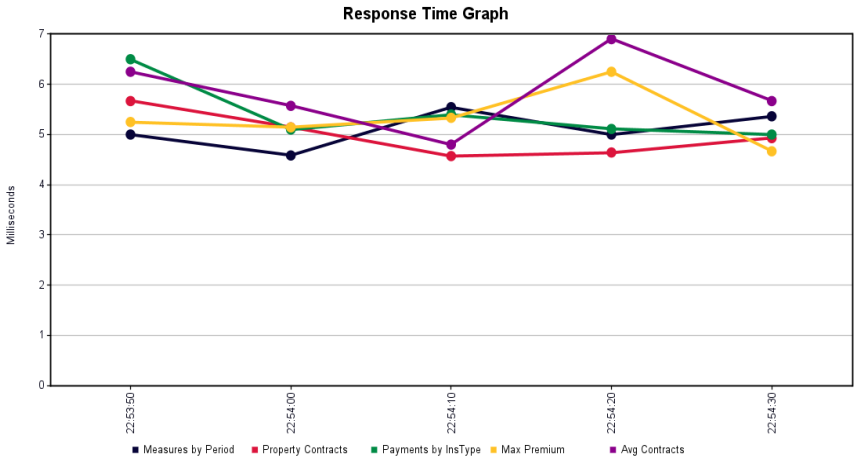

Рисунок 42 – Время отклика системы в Microsoft Analysis Services, выраженное в мс

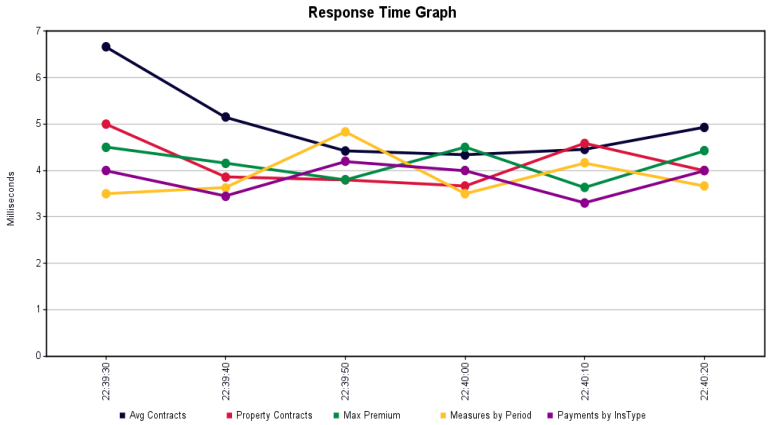

Рисунок 43 – Время отклика системы в Atoti, выраженное в мс

Пропускная способность Postgres ниже, чем в продуктах многомерного анализа данных, а ожидание отклика в разы ниже, чем у конкурентов. Время отклика в Atoti достаточно стабильное, в то время как в Microsoft Analysis Services при увеличении нагрузки наблюдается значительное увеличение времени получения ответа от системы.

4.3.5 Сравнительная оценка решений

В таблице 6 представлено финальное сравнение продуктов. Оценка проводилась по следующим параметрам:

- производительность запросов. То, насколько быстро система может обрабатывать запросы на анализ данных, является базовым фактором для возможности проводить анализ данных эффективно. Это самый важный параметр для компании, так как низкая скорость запросов, а соответственно и подготовки отчета, делает компанию менее конкурентноспособной на рынке аналитических услуг;

- устойчивость системы. То, насколько долго система способна справляться с большими нагрузками, определяет эффективность работы компании в части возможности выполнения срочных заказов;

- поддержка быстрого изменения измерений и агрегаций. Этот параметр также является важным, так как зачастую потребности бизнеса меняются очень быстро. Для того, чтобы эффективно реагировать на изменения, необходимо их вовремя уловить. Поддержка быстрых изменений в структуре отчета может существенно увеличить эффективность анализа;

- иерархическая структура. Это параметр также является важным, так как данные компании содержат иерархии. Прохождение по уровням иерархии может помочь сделать отчет более информативным и помогает избежать ошибок за счет того, что аналитик лучше понимает структуру данных;

- пользовательские вычисления. Если пользователи могут создавать свои собственные меры, это сделает отчеты более кастомизированными для

конкретного бизнеса. Без этого, отчет получится шаблонным, что не прибавляет ему ценности на рынке аналитических услуг;

- интуитивный интерфейс. Это параметр не самый приоритетный, так как опытные специалисты способны производить анализ без визуальных подсказок со стороны системы. Однако это может увеличить скорость подготовки анализа менее опытными специалистами;

- визуализация данных. Необходимость визуализации данных отчета в сторонних системах делает процесс анализа более длительным. Быстрая визуализация позволяет оценить данные на лету и, если это необходимо, быстро внести изменения в запрос;

- скорость развертывания и обновления. Это один из ключевых факторов эффективности системы анализа данных, особенно, если внесение изменений в вычисляемые элементы возможно только с последующим обновлением куба. Если куб разворачивается длительное время, это сделает систему анализа данных этой компании неповоротливой машиной. Сотрудники компании будут вынуждены выбирать между качеством анализа и его скоростью;

- масштабируемость. Этот параметр позволяет оценить, насколько эффективно компания сможет обрабатывать данные, сели их количество будет постоянно расти;

- интеграция с другими системами позволяет упростить работу с данными, так как нет необходимости заниматься миграцией данных из одной системы в другую. Интеграция предполагает удобный способ получить функциональность нескольких продуктов в одном сервисе без значительных временных затрат.

Таблица 6 – Сравнение эффективности систем анализа данных

Критерий	Реляционная форма хранения	Analysis Services	Atoti
Производительность запросов	Низкая из-за необходимости выполнения множества сложных SQL-запросов и операций объединения данных	Высокая благодаря предварительно вычисленным агрегатам и кешированию промежуточных результатов	Высокая благодаря предварительно вычисленным агрегатам и кешированию промежуточных результатов
Устойчивость системы	Пропускная способность низкая, время отклика длительное, достаточно стабильное	Пропускная способность высокая, время отклика низкое, нестабильное	Пропускная способность высокая, время отклика низкое, стабильное
Поддержка быстрого изменения измерений и агрегаций	Возможность ограничена структурой таблиц и стандартными функциями SQL	Средняя из-за ограничений на количество измерений и сложностью добавления новых атрибутов	Высокая благодаря гибкой модели данных и возможности быстро изменять структуру кубов
Иерархическая структура	Не применимо	Высокая, поддержка многомерных иерархий данных	Высокая, легко создавать иерархии из разных измерений и атрибутов
Пользовательские вычисления	Не применимо	Высокая, пользователи могут создавать пользовательские меры и показатели	Высокая, возможность гибко настраивать вычисляемые меры и показатели
Интуитивный интерфейс	Не применимо	Высокая, интуитивный интерфейс анализа данных	Высокая, простой и интуитивно понятный пользовательский интерфейс

Продолжение таблицы 6

Критерий	Реляционная форма хранения	Analysis Services	Atoti
Визуализация данных	Не применимо	Высокая, поддержка различных инструментов визуализации	Высокая, разнообразные инструменты визуализации данных
Скорость развертывания и обновления	Не применимо	Низкая, требуется пересоздание куба при изменении структуры или данных	Высокая, быстрое создание и изменение структуры кубов без перезагрузки данных
Масштабируемость	Низкая, ограничена объемом и структурой базы данных	Средняя, ограничена производительностью сервера и памятью	Высокая, хорошая масштабируемость за счет оптимизированной работы с памятью
Поддержка различных типов данных и источников	Низкая, ограничена типами данных и форматами файлов	Средняя, поддержка основных типов данных и баз данных	Высокая, широкая поддержка различных типов данных и источников

Выводы по четвертому разделу

В контексте требований, предъявляемых к системе анализа данных компанией, Atoti является лучшим выбором по сравнению с реляционной формой хранения и Microsoft Analysis Services.

Atoti обеспечивает высокую производительность запросов благодаря предрассчитанным данным и кешированию промежуточных результатов.

Atoti обладает более высокой устойчивостью, чем конкуренты.

Система Atoti позволяет создавать иерархические структуры измерений.

Пользователи могут создавать пользовательские вычисления и меры в Atoti, что делает систему более гибкой и настраиваемой под конкретные потребности бизнеса.

Интуитивный интерфейс и разнообразные инструменты визуализации данных делают работу с Atoti простой и удобной для широкого круга пользователей.

Atoti обеспечивает высокую скорость развертывания и обновления кубов, что позволяет оперативно адаптировать аналитические решения к изменяющимся потребностям бизнеса. Этот подход позволяет «не только изменять куб в реальном времени, но и уйти от необходимости использовать разные инструменты для создания кубов данных, анализа данных и визуализации данных» [5].

Высокая масштабируемость Atoti обеспечивает стабильную производительность даже при работе с большими объемами данных.

Atoti обладает широкой поддержкой источников, что позволяет пользователям анализировать данные из различных источников и форматов.

Также следует отметить, что Atoti является продуктом с открытым исходным кодом, что позволяет снизить политический риск и обеспечивает гибкость в развитии и поддержке системы.

Заключение

В ходе исследования была сформулирована гипотеза исследования: системы анализа данных на основе OLAP-продуктов с открытым исходным кодом могут конкурировать в эффективности с коммерческими OLAP-продуктами.

Для подтверждения этой гипотезы были решены следующие задачи:

– рассмотрены теоретические аспектов систем интерактивной аналитической обработки данных OLAP. По результатам выполнения данной задачи было выяснено, что одной из главных причин широкого использования технологии OLAP в компаниях, работающих с большими объемами данных, является высокая скорость обработки запросов, что является ключевым фактором для аналитических целей;

– проведен обзор существующих методов и инструментов многомерного анализа данных OLAP, включая как коммерческие, так и открытые решения, с оценкой их функциональности и производительности. По результатам выполнения данной задачи было выяснено, что на рынке много OLAP-продуктов, которые поставляются бесплатно, и при этом имеют широкий набор функциональных возможностей для анализа данных. Учитывая тот факт, что открытый исходный код делает эти продукты доступными независимо от политической и экономической ситуации, они могли бы стать подходящим решением для многих компаний;

– исследовалась актуальность применения технологии многомерного анализа данных OLAP. По результатам анализа было выяснено, что роль классического OLAP-подхода сегодня не так велика. Классические методы уступают место более эффективным методам обработки данных, которые обеспечивают более гибкий и масштабируемый анализ. Эти технологии позволяют работать с большими объемами данных и обеспечивают более высокую производительность по сравнению с традиционными методами;

– был произведен анализ деятельности предприятия для выявления проблемных процессов, требующих аналитического вмешательства. В результате анализа требований компании к системе анализа данных и особенностей данных, с которыми она работает, была выработана рекомендация рассмотреть внедрение технологии многомерного анализа данных OLAP в виде инструмента Atoti. Это обусловлено необходимостью обеспечить высокую производительность при выполнении аналитических запросов даже с большими объемами данных;

– моделирование и реализация системы OLAP. Система была смоделирована успешно;

– разработана методика оценки эффективности OLAP-систем анализа данных с включением нагрузочного тестирования. Произведена оценка эффективности и функциональности разработанной системы с точки зрения удовлетворения требований бизнеса и возможности проведения аналитических исследований. Тестирование показало, что внедренная система не уступает по своим возможностям классическому OLAP-кубу и технологии, включающей использование OLTP-таблиц для анализа данных, которая использовалась в компании до этого. Кроме того, внедренная система является более предпочтительной с точки зрения ряда параметров, таких как скорость изменений в системе и интуитивность интерфейса.

По результатам работы были выявлены следующие ключевые моменты.

– технология OLAP остается актуальной несмотря на то, что сегодня появляются более высокопроизводительные базы данных, которые позволяют не хранить предрассчитанные данные. Однако такие базы применимы не для всех структур и особенностей данных. OLAP-системы являются по-прежнему востребованными, благодаря тому, что они модифицируются, чтобы оставаться актуальными;

– включение нагрузочного тестирования в методику оценки эффективности OLAP-систем показало его значимость для определения

реальной производительности и устойчивости систем в условиях интенсивной эксплуатации;

– с использованием современных OLAP-инструментов с открытым исходным кодом, таких как Atoti, можно эффективно проводить анализ данных на основе технологии OLAP. Такие сервисы могут позволить отечественным компаниям легче и с меньшими издержками производить трансформацию систем анализа данных в текущих политических условиях.

Таким образом, исследование подтвердило гипотезу о том, что моделирование систем анализа данных на основе OLAP-продуктов с открытым исходным кодом может конкурировать по эффективности с коммерческими продуктами.

Полученные экспериментальные данные демонстрируют высокую эффективность и производительность системы анализа данных на основе OLAP в сравнении с альтернативными методами.

Таким образом, результаты данного исследования предоставляют полезные рекомендации относительно использования эффективной и экономически выгодной альтернативы коммерческим OLAP-системам компаниям, сталкивающимся с необходимостью замены ранее используемых иностранных решений в области анализа данных. Разработанная методика оценки позволяет определить наиболее подходящие системы для конкретных бизнес-потребностей, учитывая устойчивость системы. Это может помочь компаниям сократить расходы на аналитическую инфраструктуру и повысить скорость и эффективность принятия решений.

Список используемой литературы и используемых источников

1. Андреев А. Н. Классификация OLAP-систем вида xOLAP //. Рязанский государственный радиотехнический университет. 2010.
2. Барсегян А. А., Куприянов М. С., Степаненко В. В., Холод И. И. Технологии анализа данных: Data Mining, Visual Mining, Text Mining, OLAP // Спб.: БХВ–Петербург, 2017. – 384 с.
3. Берсенева Е. А., Седов А. А. Создание автоматизированной системы формирования аналитической отчетности в городской клинической больнице с использованием OLAP-технологии // Врач и информационные технологии. 2010. №4.
4. Бобрышева Г. В., Пудовкина Е. М. Анализ многомерных данных в режиме реального времени с помощью IBM Cognos TM1 // Новые информационные технологии и системы: сборник научных статей XII Международной научно-технической конференции, Пенза, 17–19 ноября 2015 года. – Пенза: Пензенский государственный университет, 2015. - С. 210-215
5. Галиев С. А. Традиционная архитектура Data warehouse / С. А. Галиев // Теоретический и практический потенциал современной науки : Сборник научных статей. Том Ч. VI. – Москва : Издательство "Перо", 2020. – С. 113-116. – EDN WHMGDJ.
6. Демидов Э.Д., Сидоренко Я. М., Мацепура А. М., Петрова А. Н. Сравнения колоночных баз данных // Наука, инновации и технологии: от идей к внедрению : Материалы II Международной научно-практической конференции молодых ученых, Комсомольск-на-Амуре, 14–18 ноября 2022 года / Редколлегия: А.В. Космынин (отв. ред.) [и др.]. Том 1. – Комсомольск-на-Амуре: Комсомольский-на-Амуре государственный университет, 2022. – С. 199-202.
7. Долгачева Е. Л., Косюк Е. Ю., Попова Д. Л., Русаков А. М. Подбор инструментальных средств анализа текстов для создания

информационно-аналитической системы // Научный альманах. – 2020. – № 11-2(73). – С. 27-38

8. Ильяшенко В. М. Обзор российских систем бизнес-аналитики: вызовы и возможности // Развитие науки и практики в глобально меняющемся мире в условиях рисков: Сборник материалов X Международной научно-практической конференции, Москва, 25 апреля 2022 года / Редколлегия: Л.К. Гуриева, З.Ш. Бабаева [и др.]. – Москва: Общество с ограниченной ответственностью "ИРОК", ИП Овчинников Михаил Артурович (Типография Алеф). 2022

9. Калинина Е. Ю. Автоматизация процедур обработки неструктурированных массивов данных в целях маркетинговых исследований рынка судостроения: маг. дис. 19.00.05 – СПб., 2017. – 48 с.

10. Карпова И. П. Базы данных. Учебное пособие. – Московский государственный институт электроники и математики (Технический университет). – М.: 2009

11. Картавая И. И. Многомерный анализ данных на OLAP-технологии // Современное состояние, проблемы и перспективы развития отраслевой науки: Материалы Всероссийской конференции с международным участием, Москва, 23–24 ноября 2017 года. – Москва: Издательство "Перо", 2017. – С. 477-481

12. Каширин И. Ю., Семченков С. Ю. Интерактивная аналитическая обработка данных в современных OLAP-системах // Бизнес-информатика. 2009. №2

13. Кириллов В., Громов Г. Введение в реляционные базы данных. СПб.: БХВ-Петербург, 2016. 464 с.

14. Кирюшин С. Разработка стратегии цифровой трансформации // Учебник 4CDTO: «О цифровизации и цифровой трансформации». Гл. 4. 1-е изд. М.: 2020

15. Козикова П. В. КРАТКИЙ ОБЗОР OLAP ТЕХНОЛОГИИ // Материалы VII Международной студенческой научной конференции «Студенческий научный форум». - 2015

16. Копова А.Ю. Бизнес-аналитика: современный инструментарий, тенденции развития // Труды Санкт-Петербургского государственного института культуры. – 2015 - 5 с.

17. Короткая Н. В. Современные платформы Bi: возможности и инструменты // Системы управления, информационные технологии и математическое моделирование: Материалы IV Всероссийской научно-практической конференции с международным участием. В 2-х томах, Омск, 19 мая 2022 года / Отв. редактор В.Н. Задорожный. Том I. – Омск: Омский государственный технический университет, 2022. – С. 179-184.

18. Костенко Е. А. OLAP-система как эффективный инструмент стратегического управления // Стратегическое развитие социально-экономических систем в регионе: инновационный подход: Материалы VIII международной научно-практической конференции, Владимир, 06 июня 2022 года – Владимир: Издательско-полиграфическая компания "Транзит-ИКС". 2022

19. Кузнецов С. Д., Велихов П. Е., Фу Ц. Аналитики в реальном времени, гибридная транзакционная/аналитическая обработка, управление данными в основной памяти и энергозависимая память // Труды ИСП РАН. 2021. №3.

20. Миронов А А., Мордвинов В. А., Скуратов А. К. Семантико-энтропийное управление OLAP и модели интеграции xOLAP в SemanticNET (ONTONET) // Информатизация образования и науки №2, 2009. С. 21-30.

21. Михайлов М. В., Коломеец М. В., Булгаков М. В., Чечулин А. А. Исследование и определение основных достоинств и недостатков существующих типов хранилищ данных и анализ их применения // Технические науки – от теории к практике. 2016. №11 (59)

22. Мытников А. Н. Анализ, проектирование и разработка высокопроизводительной системы для формирования аналитических запросов и хранения больших объемов данных на базе столбцовой СУБД Yandex Clickhouse // Информационные технологии. Проблемы и решения. – 2020. – № 3(12). – С. 99-105. – EDN WIEAJX.

23. Обзор ключевых показателей деятельности страховщиков Retrieved from https://www.cbr.ru/insurance/reporting_stat/ (Дата обращения 28.02.2024)

24. Петрова А. Н. Колоночная модель данных // Наука, инновации и технологии: от идей к внедрению: Материалы II Международной научно-практической конференции молодых ученых, Комсомольск-на-Амуре, 14–18 ноября 2022 года / Редколлегия: А.В. Космынин (отв. ред.) [и др.]. Том 1.

25. Пигарев, М. Р. Методики разработки SQL и NOSQL баз данных, их преимущества и отличия / М. Р. Пигарев, Е. А. Иванова // Цифровизация экономики: направления, методы, инструменты : Сборник материалов IV всероссийской научно-практической конференции, Краснодар, 17–21 января 2022 года. – Краснодар: Кубанский государственный аграрный университет имени И.Т. Трубилина, 2022. – С. 283-286.

26. Попов М. И., Лазарев Ф. Б. Data Vault в качестве архитектуры хранилища данных банковской организации // Auditorium. 2022. №2 (34)

27. Сарсимбаева С. М. Разработка гибридных OLAP систем многомерного анализа данных на основе Microsoft analysis services // Вестник Алматинского университета энергетики и связи. – 2019. – № 1(44). – С. 79-85.

28. Системы BI в России: обзор TAdviser [Электронный ресурс]. 2021 URL: https://www.tadviser.ru/index.php/BI?cache=no&ptype=otr (Дата обращения 27.11.2022)

29. Хабр [Электронный ресурс] / ClickHouse: очень быстро и очень удобно. Режим доступа: https://habr.com/ru/post/322724/, свободный (дата обращения 20.11.2023 г).

30. Федоров А., Елманова Н. Введение в OLAP-технологии Microsoft, Москва: Диалог-МИФИ, 2002

31. Форсайт. Аналитическая платформа. Документация на продукт https://help.fsight.ru/ru/first_topic.htmFAQ. [Электронный ресурс]. 2022 (Дата обращения 25.11.2022)

32. Щербакова Д.О. Обзор основных программных продуктов для построения систем анализа данных по технологии OLAP: сборник статей по материалам CCCV Международной заочной научно-практической конференции «Молодой исследователь: вызовы и перспективы» (Москва, Апрель 2023 г.).

33. Щербакова Д.О. Оптимизация процесса работы с данными с помощью инструмента многомерного анализа данных Atoti: сборник статей по материалам Международной заочной научно-практической конференции «Молодой исследователь: вызовы и перспективы» (Москва, Декабрь 2023 г.)

34. Якубова В. Д. Сравнительный анализ реляционных и не реляционных баз данных в управлении / В. Д. Якубова Студенческие научные исследования: сборник статей V Международной научно-практической конференции, Пенза, 10 марта 2021 года. – Пенза: "Наука и Просвещение" (ИП Гуляев Г.Ю.), 2021. – С. 90-94.

35. Agrawal R., Gupta A., Sarawagi S.: Modeling Multidimensional Databases. Int. Conf. on Data Engineering (ICDE), pp.232–243, 1997

36. Atoti. Atoti Documentation: FAQ. [Электронный ресурс]. 2022 Retrieved from https://docs.atoti.io/latest/getting_started/tutorial/tutorial.html (Дата обращения 27.11.2022)

37. Balke W., Homoceanu S. Data Warehousing & Data Mining - C4 // Institut für Informationssysteme Technische Universität Braunschweig, 2009

38. Benjelloun M. El, Amin E. Impact of using Snowflake Schema and Bitmap Index on Data Warehouse Querying // Int. J. Comput. Appl., vol. 180, no. 15, 2018. pp. 33–35

39. Cabibbo L., Torlone R.: A Systematic Approach to Multidimensional Databases. 5th Italian Symposium on Advanced Database Systems (SEBD), pp.361–377, 1997.

40. Codd E. F., Codd S. B., Salley C. T. Providing OLAP (on-line analytical processing) to user-analysts: An IT mandate. Technical report // Wiley, 1993

41. Codd E. F. Further Normalization of the Data Base Relational Model // Research Report / RJ / IBM / San Jose, California RJ909, 1971

42. Codd E. F. A Relational Model of Data for Large Shared Data Banks // Communications of the ACM Volume 13 Issue, June 1970, pp 377–387.

43. CubesViewer. CubesViewer Documentation: FAQ. [Электронный ресурс] 2022 URL: https://github.com/jjmontesl/cubesviewer/blob/master/doc/guide/index.md (Дата обращения 27.11.2022)

44. Data Warehouse Architecture: Traditional vs. Cloud: [Электронный ресурс]. 2019 URL: https://panoply.io/data-warehouse-guide/data-warehouse-architecture-traditional-vs-cloud/ (Дата обращения 23.11.2022)

45. Dageville B. The Snowflake Elastic Data Warehouse // SIGMOD/PODS' 16 June 26 - July 01, 2016, San Francisco, CA, USA, 2016

46. GeeksForGeeks: Difference between Star Schema and Fact Constellation Schema [Электронный ресурс] 2023 URL: https://www.geeksforgeeks.org/difference-between-star-schema-and-fact-constellation-schema (Дата обращения 13.05.2023)

47. Grabova O., Darmont J., Chauchat J-H, Zolotaryova I. Business Intelligence for Small and Middle-Sized Entreprises // ACM SIGMOD Rec., vol. 39, no. 2, pp. 39–50, 2010

48. Gyssen M., Lakshmanan L. V. S.: A Foundation for Multi-Dimensional Databases. 23rd Int. Conf. on Very Large Data Bases (VLDB), pp.106–115, 1997

49. Inmon W. H., Building the Data Warehouse // Wiley, 2005 -517 с.

50. Jedox Palo Knowlegement Base [Электронный ресурс]. 2022 URL: https://knowledgebase.jedox.com/jedox/planning/palo-olap-functions.html (Дата обращения 25.11.2022)

51. Karmani M. Z., Mustafa G., Sarwar N., Wajid S. A Review of Star Schema and Snowflakes Schema. - 2020, 13 с.

52. Khan W., Khan K., Teerath, Cheng Z. & Raj, Kislay, Roy, Arunabha M., Luo B. SQL and NoSQL Databases Software architectures performance analysis and assessments -- A Systematic Literature review. 10.48550/arXiv.2209.06977.

53. Kimball R., Ross M., The Data Warehouse Toolkit: The Complete Guide to Dimensional Modeling // Wiley, 2002

54. Lapa, Joaquim & Bernardino, Jorge & Figueiredo Ana. A comparative analysis of open source business intelligence platforms // ACM International Conference Proceeding Series. – 2014

55. Linstedt D. (December 2010). Super Charge your Data Warehouse. Dan Linstedt. ISBN 978-0-9866757-1-3

56. Mondrian Pentaho Documentation [Электронный ресурс]. 2022 URL: https://mondrian.pentaho.com/documentation/faq.php (Дата обращения 25.11.2022)

57. OLAP’s promising path forward [Электронный ресурс]. 2021 Retrieved from https://medium.com/atoti/olaps-promising-path-forward-f83cbda9c375 (Дата обращения 14.11.2023)

58. Ouali, Nahla, Tmar, Mohamed, Gargouri, Faiez. A data-centric approach to manage business processes // Computing 98, 2015

59. Pendse N. The OLAP Report: Succeeding with On-line Analytical Processing // Business Intelligence, 1995

60. Polymatica Analytics [Электронный ресурс]. 2022 URL: https://www.polymatica.ru/polymatica-analytics (Дата обращения 27.11.2022)

61. Qlik - FAQ [Электронный ресурс]. 2022 URL:https://www.qlik.com/ru-ru (Дата обращения 25.11.2022)

62. Ravat F., Teste O., Tournier R., Zurfluh G. Designing and Implementing OLAP Systems from XML Documents. 10.1007/978-0-387-87431-9_15.

63. Ravat F., Teste O., Tournier R., Zurfluh G. A Conceptual Model for Multidimensional Analysis of Documents. 4801. 550-565. 10.1007/978-3-540-75563-0_37

64. The Rise and Fall of the OLAP Cube [Электронный ресурс]. 2020 Retrieved from https://www.holistics.io/blog/the-rise-and-fall-of-the-olap-cube (Дата обращения 09.11.2023)

65. Thomsen C., Pedersen T. A Survey of Open Source Tools for Business Intelligence // International Journal of Data Warehousing and Mining. – 2005

66. Tournier R.: OLAP model, algebra and graphic language for multidimensional databases. Scientific Report n° IRIT/RR—2007-6–FR, IRIT, Université Paul Sabatier (Toulouse 3), France

67. Visiology Docs [Электронный ресурс]. 2022 URL: https://ru.visiology.su/products/why-visiology (Дата обращения 27.11.2022)

Приложение А

Спецификация требований к программному обеспечению

Назначение

Эта спецификация требований описывает функциональные и нефункциональные требования для системы анализа данных. Этот документ предназначен для команды, которая будет реализовывать и проверять корректность работы системы.

ОБЩЕЕ ОПИСАНИЕ

Описание продукта

Система анализа данных — это система, которая позволит пользователям:

1. Создавать многомерные кубы данных для анализа множества измерений и показателей.
2. Выполнять аналитические запросы к данным и получать соответствующие результаты.

Доступ к системе анализа данных может осуществляться только авторизованными пользователями, связанными с аналитическими процессами в организации.

1. Функциональные требования:

1. Интерфейс анализа данных: Пользовательский интерфейс должен предоставлять возможности для создания, настройки и выполнения аналитических запросов к данным.
2. Создание кубов данных: Система должна позволять пользователям создавать многомерные кубы данных для анализа множества измерений и показателей.
3. Агрегирование и суммирование: Система должна предоставлять возможность агрегировать и суммировать данные для быстрого выполнения аналитических запросов.

4. Поддержка различных типов данных: Приложение должно поддерживать различные типы данных, включая числовые, текстовые, даты и другие.

5. Интеграция с источниками данных: Система должна позволять интегрировать данные из различных источников, таких как базы данных, файлы CSV и другие.

6. Пользовательские отчеты и визуализация: Пользователям должны быть доступны гибкие возможности создания отчетов и визуализации данных, включая диаграммы, графики, таблицы и др.

2. Нефункциональные требования:

1. Производительность: Система должна обеспечивать высокую производительность при выполнении аналитических запросов даже с большими объемами данных.

2. Масштабируемость: Приложение должно быть легко масштабируемым для обработки роста объемов данных и пользовательской нагрузки.

3. Безопасность: Система должна обеспечивать аутентификацию, авторизацию и контроль доступа к данным для обеспечения безопасности информации.

4. Надежность: Приложение должно быть надежным и иметь механизмы для обнаружения и восстановления от сбоев.

5. Удобство использования: Интерфейс пользователя должен быть интуитивно понятным и легким в использовании для широкого круга пользователей.

3. Интеграция и взаимодействие:

1. API и интеграция: Система должна предоставлять API для интеграции с другими приложениями и инструментами анализа данных.

2. Экспорт данных: Пользователи должны иметь возможность экспортировать данные и результаты анализа для дальнейшего использования.

4. Требования к производительности:

1. Время выполнения запросов: Система должна обеспечивать быстрое выполнение аналитических запросов, включая запросы к многомерным кубам данных.

2. Отклик системы: Приложение должно иметь низкий временной отклик для пользовательских запросов и операций.

5. Требования к поддержке:

1. Поддержка и обновления: Поставщик системы должен обеспечивать регулярные обновления, поддержку и обслуживание продукта.

Приложение Б

Бизнес-сценарий для страховой компании

Целью данного бизнес-сценария является повышение эффективности страховых продуктов и улучшение удовлетворенности клиентов.

Этапы реализации:

1. Выявление трендов в собираемых премий за последние три года. Оценка того, какие периоды являются наиболее прибыльными.

Запрос на языке SQL:

2. Оценка того, как изменяется интерес к добровольному страхованию имущества в течение года. Определение кварталов с наибольшим и наименьшим спросом.

3. Выявление регионов с наибольшим и наименьшим количеством заключённых договоров для дальнейшей локализации маркетинговых усилий.

4. Идентификация страховщиков с наивысшими показателями в сумме премий для возможного перераспределения ресурсов.

5. Оценка того, какие виды страхования требуют наибольших выплат и могут нуждаться в пересмотре условий или цен.

Результатом выполнения сценария будут являться:

1. Определение наиболее и наименее прибыльных сегментов бизнеса.

2. Разработка стратегий по оптимизации страховых продуктов и управлению рисками.

3. Планирование бюджета и ресурсов на основе данных о производительности продуктов.