СОДЕРЖАНИЕ

ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

В настоящем отчете применяют следующие термины, сокращения и определения с соответствующими определениями.

OLAP	
API	
ETL	
BI-система	
БД	
NoSQL-хранилищами	
batch processing	
stream processing	
OLAP	
MOLAP	
ROLAP	
HOLAP	
MapReduce	
SQL	
PostgreSQL	
MySQL	
Oracle	
СУБД	

ВВЕДЕНИЕ

В современном мире финансовая аналитика требует обработки больших объемов данных, поступающих из различных источников. Для эффективного анализа и принятия решений необходимо агрегировать данные, сводя их к структурированным наборам, удобным для последующей обработки. В данной главе рассматриваются особенности систем агрегирования данных, их архитектурные принципы и функциональные возможности.

1 Исследование предметной области

В современных финансовых организациях и крупных компаниях система мотивации сотрудников часто включает премирование на основе KPI, объемов продаж, выполнения планов и других показателей. Ручной расчет таких премий трудоемок, подвержен ошибкам и не позволяет оперативно анализировать данные в различных разрезах (по отделам, регионам, временным периодам).

На протяжении последних лет различные исследователи и компании предлагали решения в области агрегирования данных и аналитической обработки. Однако существующие подходы либо ориентированы на обработку данных в пакетном режиме, что замедляет расчет различных данных, том числе расчета премий, в реальном времени, либо требуют значительных вычислительных ресурсов. Исследование направлено на разработку системы, способной эффективно обрабатывать финансовые данные в режиме оперативной аналитики, сочетая высокую скорость обработки, точность расчетов и адаптивность к изменяющимся условиям.

1.1 Исследование особенностей систем агрегирования данных

1.1.1 Определение и назначение систем агрегирования данных

Системы агрегирования данных представляют собой программные и аппаратные комплексы, предназначенные для сбора, обработки, обобщения и хранения информации из разнородных источников. Эти системы применяются для консолидации данных и их подготовки к аналитической обработке.

Основные задачи агрегирования данных:

- Объединение данных из различных источников (базы данных, API, файловые хранилища);
- Очистка и трансформация данных;
- Поддержка процессов ETL (Extract, Transform, Load);
- Обеспечение оперативного доступа к агрегированным данным;
- Подготовка данных для аналитических и BI-систем.

1.1.2 Классификация систем агрегирования данных

1.1.2.1 По типу источников данных

Реляционные базы данных (SQL-based) – традиционные базы данных, такие как PostgreSQL, MySQL, Oracle, которые хорошо подходят для структурированных данных и обеспечивают мощные средства запросов и агрегирования.

NoSQL-хранилища – базы данных вроде MongoDB, Cassandra, предназначенные для обработки неструктурированных и полуструктурированных данных, часто применяются для потоковой аналитики и масштабируемых решений.

Потоковые данные (Stream Data) – данные, поступающие в реальном времени из сенсоров, логов, API-интерфейсов (Kafka, Apache Flink). Используются для построения предсказательных моделей и аналитики в реальном времени.

1.1.2.2 По способу обработки данных

Пакетная обработка (Batch Processing) – применяется, когда анализ проводится на больших объемах данных, но в режиме периодического обновления. Подходит для исторического анализа, но не всегда удовлетворяет требованиям оперативной аналитики.

Потоковая обработка (Stream Processing) – позволяет анализировать данные в реальном времени. Применяется для финансовой аналитики, где требуется оперативное реагирование.

Гибридные подходы – сочетают пакетную и потоковую обработку, позволяя обрабатывать данные как в реальном времени, так и ретроспективно. Этот вариант предпочтителен в задачах финансовой аналитики, так как он обеспечивает баланс между производительностью и точностью.

1.1.2.3 По архитектуре

Централизованные системы – все данные собираются и обрабатываются в одном хранилище. Такой подход удобен для небольших объемов данных, но плохо масштабируется.

Распределенные системы – данные обрабатываются на множестве узлов, что повышает отказоустойчивость и масштабируемость. Такие системы, как Hadoop, Spark и ClickHouse, позволяют работать с большими объемами данных и обеспечивают высокую производительность.

Выбор конкретного класса системы агрегирования данных зависит от требований к обработке данных, скорости аналитики и потребностей бизнеса. В рамках исследования предпочтение отдается распределенной системе с гибридной моделью обработки, так как она позволяет анализировать финансовые данные как в реальном времени, так и с учетом исторических трендов.

1.1.3 Архитектурные принципы систем агрегирования данных

Современные системы агрегирования данных строятся на основе следующих архитектурных подходов:

- Многоуровневые архитектуры, включающие уровни сбора, обработки и хранения данных.
- Микросервисный подход, обеспечивающий гибкость и масштабируемость.
- Использование облачных технологий, позволяющее динамически изменять ресурсы под нагрузку.

1.2 Исследование методов выполнения аналитических запросов

1.2.1 Введение в аналитические запросы

Аналитические запросы предназначены для обработки больших объемов данных с целью выявления закономерностей, трендов и аномалий. Они широко используются в финансовой аналитике для расчета премий, оценки рисков и прогнозирования.

1.2.2 Основные методы выполнения аналитических запросов

1.2.2.1 OLAP (Online Analytical Processing)

OLAP-технология предназначена для многомерного анализа данных и позволяет выполнять сложные аналитические запросы.

Основные типы OLAP:

- MOLAP (Multidimensional OLAP) — хранение данных в многомерных кубах.

- ROLAP (Relational OLAP) — хранение данных в реляционных таблицах, обработка с помощью SQL-запросов.
- HOLAP (Hybrid OLAP) — гибридный подход, сочетающий MOLAP и ROLAP.

1.2.2.2 MapReduce

Метод MapReduce позволяет обрабатывать большие объемы данных параллельно на распределенных системах. Этот метод эффективен для работы с неструктурированными и полуструктурированными данными.

MapReduce не удовлетворяет требованиям оперативной, низкозадержанной аналитики, необходимой для расчета премий в реальном времени. Он может быть использован лишь для фоновых, ночных пакетных расчетов — но не как основа системы, ориентированной на актуальные бизнес-задачи.

1.2.2.3 SQL и NoSQL-подходы

SQL-методы: Используются в традиционных реляционных БД (PostgreSQL, MySQL, Oracle) для аналитических запросов с агрегацией (SUM, AVG, COUNT, GROUP BY).

NoSQL-методы: Используются в документоориентированных, графовых и других БД (MongoDB, Cassandra) для обработки данных в реальном времени.

Для данной работы выбираем SQL-методы, так как для аналитики необходимы аналитические запросы с агрегацией.

1.2.2.4 Индексы и материализованные представления

Для ускорения аналитических запросов применяются индексы (B-деревья, Bitmap-индексы) и материализованные представления, хранящие предварительно рассчитанные результаты запросов.

1.3 Исследование особенностей работы оперативной аналитической обработки

1.3.1 Определение оперативной аналитической обработки (OLAP)

Оперативная аналитическая обработка (OLAP) представляет собой технологию, обеспечивающую быстрый доступ к агрегированным данным в многомерных структурах. OLAP используется в системах финансовой

аналитики для мгновенного расчета показателей, таких как премии, рентабельность и финансовые риски.

1.3.2 Основные принципы OLAP

Многомерность данных: Данные организованы в виде кубов с различными измерениями (время, категория, география).

Агрегация: Данные сводятся к обобщенным показателям, что снижает объем вычислений.

Оптимизация хранения: Используются специальные структуры данных для быстрого доступа.

1.3.3 Технологии оперативной аналитической обработки

Среди наиболее распространенных технологий OLAP можно выделить:

Apache Druid: Высокопроизводительная аналитическая база данных для работы с потоковыми и историческими данными.

ClickHouse: Колонночная СУБД с высокой скоростью выполнения аналитических запросов.

Microsoft Analysis Services: Инструмент для работы с OLAP-кубами в экосистеме Microsoft.

1.3.4 Преимущества и недостатки OLAP

Преимущества:

- Высокая скорость выполнения запросов за счет предварительной агрегации данных.
- Возможность многомерного анализа данных.
- Поддержка сложных аналитических вычислений.

Недостатки:

- Высокие затраты на вычислительные ресурсы.
- Ограниченная гибкость по сравнению с транзакционными базами данных.
- Сложность настройки и поддержки.

1.3.5 Системы, использующие OLAP

Oracle OLAP – интегрирован в Oracle Database, обеспечивает мощные аналитические функции для обработки многомерных данных.

Минусы: высокая стоимость лицензирования, сложность настройки и администрирования, требует значительных вычислительных ресурсов.

Microsoft Analysis Services (SSAS) – мощный инструмент для построения OLAP-кубов и выполнения аналитических запросов.

Минусы: ограниченная поддержка неструктурированных данных, зависимость от экосистемы Microsoft, требует профессиональной настройки.

IBM Cognos TM1 – высокопроизводительная платформа для финансового моделирования и планирования.

Минусы: высокая стоимость, сложность интеграции с внешними системами, требует квалифицированных специалистов.

SAP BW/4HANA – корпоративное хранилище данных с OLAP-функциями, работает в in-memory режиме.

Минусы: высокая стоимость лицензий и инфраструктуры, требует больших объемов оперативной памяти, сложность администрирования.

Amazon Redshift – облачное хранилище данных с поддержкой OLAP-запросов и масштабируемой аналитикой.

Минусы: высокая стоимость при больших объемах данных, зависимость от AWS-инфраструктуры, задержки при обработке сложных многомерных запросов.

Каждая из этих систем обладает своими характеристиками и подходами. Например, Oracle OLAP интегрирован в реляционную СУБД и поддерживает сложные аналитические запросы, а Amazon Redshift предлагает горизонтальное масштабирование и облачную доступность. Microsoft SSAS ориентирован на удобство работы с Excel и Power BI, а SAP BW/4HANA обеспечивает высокую скорость обработки за счет технологии in-memory.

1.3.6 Заключение

Для реализации системы агрегирования данных, ориентированной на оперативную финансовую аналитику и расчет премий, наиболее рациональным выбором является использование системы управления базами данных ClickHouse. Эта технология представляет собой колоночную СУБД,

оптимизированную для высокоскоростной аналитической обработки больших объемов данных. В отличие от традиционных строковых СУБД, ClickHouse обеспечивает быструю агрегацию и фильтрацию по нужным измерениям, что критично для построения OLAP-запросов и финансовой отчетности. Среди основных преимуществ системы — поддержка многомерного анализа (OLAP), высокая масштабируемость, сжатие данных и возможность обрабатывать миллиарды строк в секунды. Кроме того, ClickHouse легко интегрируется с BI-средствами и имеет низкие требования к инфраструктуре по сравнению с корпоративными решениями вроде SAP BW или Oracle OLAP. Таким образом, с учетом требований к скорости, надежности и эффективности при работе с финансовыми данными в реальном времени, ClickHouse представляет собой оптимальную технологическую основу для разработки данной системы.

2 Исследование подхода к OLAP технологии

Каждая из этих систем обладает своими характеристиками и подходами. Например, Oracle OLAP интегрирован в реляционную СУБД и поддерживает сложные аналитические запросы, а Amazon Redshift предлагает горизонтальное масштабирование и облачную доступность. Microsoft SSAS ориентирован на удобство работы с Excel и Power BI, а SAP BW/4HANA обеспечивает высокую скорость обработки за счет технологии in-memory.

2.1 Анализ и сравнение существующих подходов

OLAP (Online Analytical Processing) — это технология многомерной аналитики, предназначенная для оперативного анализа больших объемов данных по различным измерениям (время, регионы, клиенты, продукты и т.д.). Она позволяет выполнять сложные аналитические запросы с минимальной задержкой и широко используется в бизнес-аналитике, финансовом моделировании, логистике, телекоммуникациях и других отраслях.

2.1.1 ROLAP (Relational OLAP)

ROLAP работает поверх реляционных баз данных и использует SQL-запросы для выполнения аналитических операций. Данные хранятся в

обычных таблицах, но дополнительно создаются агрегаты и представления (views), оптимизированные под OLAP-запросы.

Агрегация происходит во время выполнения запроса (on-the-fly). Используются индексы, материализованные представления, партиционирование. Возможна работа с большими объемами исходных данных без предварительной агрегации.

Преимущества:

- Гибкость.
- Масштабируемость.
- Использование стандартного SQL.

Можно легко изменять структуру запросов и измерения. Хорошо работает с современными колоночными СУБД. Облегчает интеграцию и обучение персонала

Недостатки:

- Производительность зависит от оптимизации запросов.
- Высокая нагрузка.

Зависит от оптимизации запросов. Высокая нагрузка на СУБД при больших объемах данных.

Используется в таких СУБД: ClickHouse, PostgreSQL (с расширениями), Amazon Redshift, Google BigQuery.

Применяется в финансовых платформах «Тинькофф Аналитика», BI-системах Tableau и Metabase.

2.1.2 MOLAP (Multidimensional OLAP)

MOLAP использует специально подготовленные многомерные структуры (кубы), в которых данные агрегируются и хранятся заранее. Эти кубы представляют собой нативное OLAP-хранилище, не зависящее от реляционной модели.

Данные проходят процесс ETL и загружаются в многомерные кубы. Кубы агрегируют значения по измерениям заранее (pre-aggregation). Запросы к кубам выполняются почти мгновенно, так как обращаются к уже рассчитанным данным.

Преимущества:

- Очень высокая скорость отклика.
- Эффективность для предсказуемых сценариев анализа.

Недостатки:

- Ограниченная гибкость: изменение структуры куба требует перестроения.
- Трудности масштабирования при больших объемах данных.

Используется в таких система: Microsoft SQL Server Analysis Services (SSAS), IBM Cognos, Oracle Essbase.

Используется в крупных корпорациях с фиксированными аналитическими сценариями (например, отчетность по кварталам).

2.1.3 HOLAP (Hybrid OLAP)

HOLAP — это гибридный подход, который совмещает в себе преимущества ROLAP и MOLAP. Детальные данные хранятся в реляционной базе (ROLAP), а агрегированные — в кубах (MOLAP).

Кубы содержат агрегаты для часто используемых запросов. Подробные значения запрашиваются из реляционной базы. Система определяет, откуда извлекать данные в зависимости от запроса.

Преимущества:

- Компромисс между гибкостью и производительностью.
- Возможность обработки больших объемов без потери скорости.

Недостатки:

- Сложность реализации и поддержки.
- Необходимость синхронизации кубов и таблиц.

Используется в: SAP BW, Microsoft SSAS (в гибридном режиме), IBM Cognos TM1. Применяется в крупных банках, ритейле, логистике.

Таблица 1

Подход	**Хранение данных**	**Скорость агрегаций**	**Гибкость**	**Масштабируемость**	**Недостатки**
ROLAP	Реляционные таблицы	Средняя	Высокая	Высокая	Медленная агрегация, нагрузка на СУБД

MOLAP	Многомерные кубы	Очень высокая	Низкая	Средняя	Требует подготовки, ограничен по объему
HOLAP	Комбинированное	Высокая	Средняя	Средняя	Сложная реализация, дублирование логики

После детального анализа архитектур OLAP-технологий видно, что ROLAP обеспечивает необходимую гибкость и масштабируемость, особенно в условиях постоянно меняющейся финансовой отчетности. Несмотря на то, что MOLAP превосходит по скорости при фиксированных сценариях, его ограниченность в динамической бизнес-среде делает его менее подходящим. HOLAP представляет интерес как компромисс, однако его сложность и ресурсоемкость внедрения не оправданы в рамках данной задачи. Поэтому в данной работе выбран **MOLAP-подход** с реализацией на базе **ClickHouse** — колоночной СУБД, предоставляющей быстрый отклик, нативную поддержку аналитических функций и совместимость с SQL-запросами. Для улучшения MOLAP-подхода изменим способ хранения уже агрегированных данных для того чтобы увеличить скорость выдачи данных и уменьшить объем хранимых агрегированных данных.

2.2 Разработка алгоритма для системы агрегирования данных

Основной идеей формирование куба, заключается в том, чтобы формировать каждый последующий разворот куба мы будет на основе предыдущего, так идя он начальной таблице, мы постепенно будет ее увеличивать, добавляя сгруппированные значения и из полученной таблице будем рекурсивно группировать её по каждой иерархии аналитика (рис. 1).

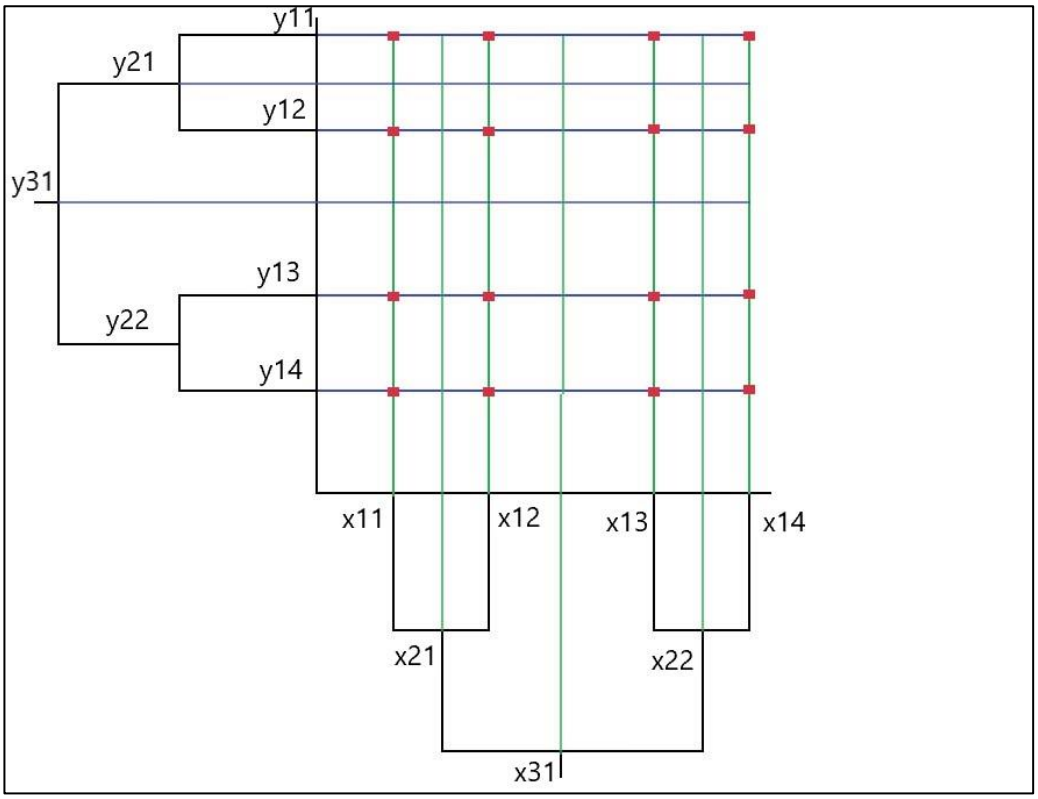

Рис. 1. Агрегация в двумерной плоскости [разработано автором]

На рис. 1 представлена первая агрегация, при которой формируются значения на листьях. Добавляя полученные значения к основным, мы можем делать группировку следующего уровня иерархии аналитика. Нам не приходится каждый раз рассчитывать куб до нужного уровня, поскольку при каждой агрегации уровня иерархии аналитика, сформированные данные уже будут, но используя больше памяти при каждой последующей агрегации над кубом. Данную проблему можно решить тем, что каждый столбец данных считать отдельно, последовательно. Таким образом если у нас в изначальных столбцах данных находятся 100 столбцов, мы соберем куб 100 раз для каждого столбца, следовательно, мы уменьшим объем потребляемой памяти при формировании куба, но скорость формировании всего куба увеличиться.

2.2.1 Как формируются индексы

С точки зрения математики для того чтобы хранить плоскость на прямой необходимо каждой точки дать свой уникальный индекс (номер), таким образов если у нас многомерная плоскость, то для каждой из плоскостей нужно присвоить свой индекс. Сформированный куб это и есть многомерная

пространство плоскостей. Каждый индекс будет состоять из индексов аналитика.

Для того чтобы сформировать индекс аналитика, нужно подготовить таблицу, в которой они будут храниться. Данная таблица будет состоять из уникальных значений каждого уровня иерархии аналитика, в которой в столбце «Имя» будут находится имена всех уникальных значений аналитики, а в столбце «Индекс» число для каждого уникального значения в столбце «Имя».

Перед подстановкой индексов, нужно отредактировать с агрегированные данные. Для этого нужно создать столбы с названиями всех аналитиков и подставить первое не пустое значение из названий столбцов иерархии аналитика. Таки образов в итоговой таблицу окажется количество столбцов равное количеству аналитиков.

Далее необходимо заменить значения в столбцах на значение в таблице с индексами и соединить каждую строку в одну ячейку. Таким образом получаться индексы для каждого значения OLAP-куба.