Департамент образования и науки города Москвы

Государственное автономное образовательное учреждение высшего образования города Москвы

«Московский городской педагогический университет»

Институт цифрового образования

Департамент информатики, управления и технологий

ДИСЦИПЛИНА:

Распределенные системы

**Лабораторная работа 3.1**

**Проектирование архитектуры хранилища данных**

Выполнил(а): Шведова С.С., группа: АДЭУ-211

Преподаватель: Босенко Т.М.

Москва

2024

**Цель работы**: разработать архитектуру хранилища больших данных для заданного сценария использования.

**Задача:** создать архитектуру хранилища больших данных для компании, занимающейся автомобилями.

**Вариант 13.** Крупная автомобильная компания:

- Объем данных: 500 ТБ в год, рост 40% ежегодно.

- Скорость получения: до 5000 событий в секунду.

- Типы данных: 50% структурированные, 40% полуструктурированные, 10% неструктурированные.

- Требования к обработке: анализ данных с подключенных автомобилей, оптимизация производственных процессов.

- Доступность: 99.95%, время отклика <10 секунд.

- Безопасность: шифрование, соответствие 152-ФЗ и международным автомобильным стандартам.

**Алгоритм выполнения работы:**

* Объем данных: 500 ТБ в год.

- Рост: 40% ежегодно.

**1.2. Определить источники данных.**

Информация о поведении пользователей на сайте и в приложении, их транзакциях, поступающая со скоростью до 5000 событий в секунду.

**1.3. Выявить типы данных**

- Структурированные (данные из систем учета и CRM).

- Полуструктурированные (веб-сайты, мобильные приложения и камеры видеонаблюдения).

- Неструктурированные (отзывы и обратная связь клиентов, социальные сети).

* 1. **. Оценить объемы данных и скорость их поступления.**

- Объем данных: 500 ТБ в год.

- Скорость получения: до 5000 событий в секунду.

**1.5. Определить требования к аналитике и отчетности.**

- Анализ данных с подключенных автомобилей: еженедельно.

- Оптимизация производственных процессов: ежегодно.

**1.6. Доступность данных**

- Доступность: 99.95%, время отклика <10 секунд.

**Выбор модели хранилища данных**

Для выбранной компании подойдет модель Lakehouse, которая объединяет преимущества Data Lake и Data Warehouse, предоставляя возможность хранить разнообразные типы данных (структурированные, полуструктурированные, неструктурированные) и обеспечивая требуемый уровень доступности, безопасности и масштабируемости. Lakehouse поддерживает как аналитику в режиме реального времени, так и сложные аналитические запросы.

1. **Компоненты архитектуры**
   1. **Источники данных**

- Неструктурированные: отзывы и обратная связь клиентов, социальные сети.

- Полуструктурированные: веб-сайты, мобильные приложения и камеры видеонаблюдения.

- Структурированные: данные из систем учета и CRM.

**2.2. Слой сбора данных**

- Apache Kafka для обработки потоков данных.

- Пользовательские коннекторы для CRM.

- Logstash для сбора логов.

**2.3. Слой хранения данных**

- PostgreSQL для структурированных данных и метаданных.

- HDFS для работы с большими данными в экосистеме.

- MongoDB для хранения полуструктурированных данных.

**2.4. Слой обработки данных**

- Apache Spark является фреймворком для обработки и анализа больших объемов данных.

- Apache Flink является потоковым обработчиком данных, который может обрабатывать большие объемы событий с высокой скоростью.

- Microsoft Azure Stream Analytics для обработки данных в режиме реального времени с использованием облачных ресурсов.

**2.5. Слой аналитики и машинного обучения**

- Power BI для визуализации и дашбордов.

- Jupyter Notebook для интерактивной аналитики.

- TensorFlow - это открытая платформа для машинного обучения.

**2.6. Слой управления данными**

- Apache Ranger для контроля доступа и аудита.

- Apache Atlas для управления метаданными.

**2.7. Слой оркестрации и мониторинга**

- Kubernetes - программное обеспечение для оркестровки контейнеризированных приложений.

На рисунке 1 изображена архитектура хранения данных «Звезда» для процессов.

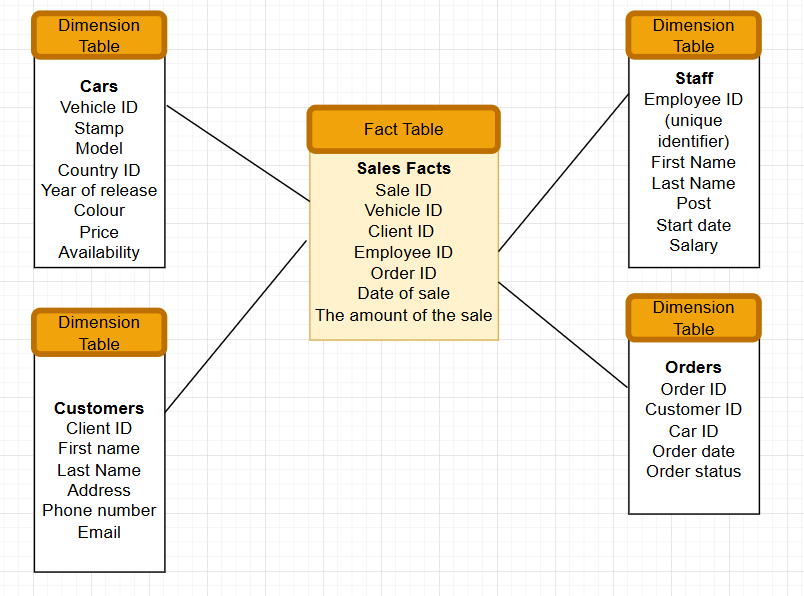


Рисунок 1. Архитектура хранения данных «Звезда»

На рисунке 2 изображена архитектура хранения данных «Снежинка» для самих данных.

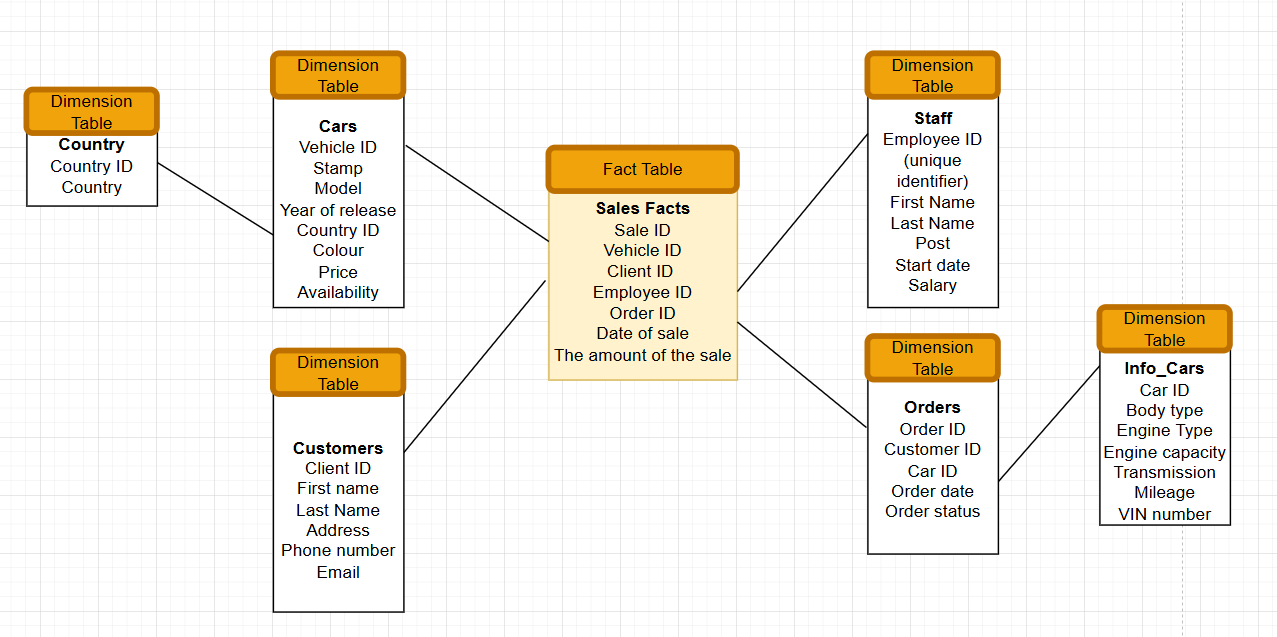


Рисунок 2. Архитектура хранения данных «Снежинка»

Больше приведенным требованиям отвечает архитектура хранения «снежинка», так как занимает меньшее дисковое пространство и имеет лучшую целостность данных.

**OLAP и OLTP**

- OLTP, как подход к обработке транзакций в реальном времени, необходим для автомобильной компании для обеспечения операционной деятельности, такой как обработка заказов на автомобили, запчасти, обслуживание клиентов, учет транзакций по продаже и обслуживанию автомобилей, управление запасами и т. д.

- OLAP, с другой стороны, предоставляет возможности для анализа данных. Для автомобильных компаний это может быть полезно для анализа продаж, прогнозирования спроса, управления запасами, анализа рыночных тенденций, оценки эффективности маркетинговых кампаний и улучшения стратегического планирования. Аналитика данных также может помочь в обнаружении дополнительных возможностей для роста и оптимизации бизнес-процессов.

Cхема OLAP и OLTP изображена на рисунке 3.

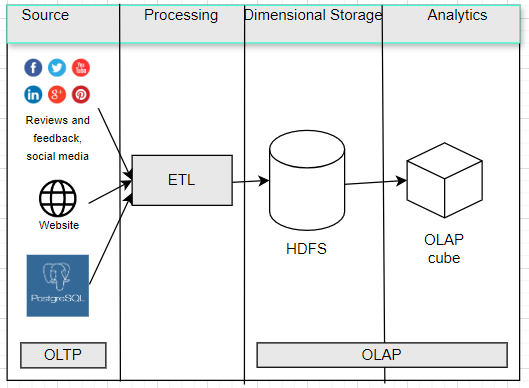


Рисунок 3. Cхема OLAP и OLTP

На рисунке 4 изображена схема архитектуры.

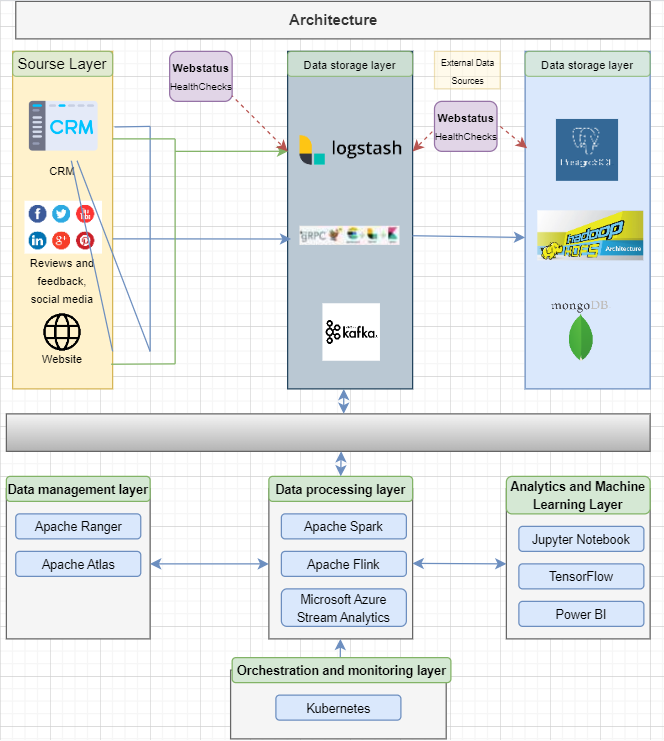


Рисунок 4. Схема архитектуры

**Процесс обработки данных**

1. Процесс обработки данных начинается с сбора информации из различных источников, включая социальные сети для неструктурированных данных, веб-сайты и мобильные приложения для полуструктурированных данных, и CRM системы для структурированных данных;
2. Пакетные задачи выполняются с помощью Spark;
3. Lля исследования данных и создания отчетов аналитики используют Jupyter Notebook и Power BI;
4. Модели машинного обучения обучаются на исторических данных (OLAP).

**Масштабирование и отказоустойчивость**

1. Apache Kafka для обработки потоков данных;
2. Пользовательские коннекторы для CRM;
3. Logstash для сбора логов.
4. Использование кластерной архитектуры Hadoop для горизонтального масштабирования.

**Безопасность**

1. Регулярное резервное копирование и план аварийного восстановления.
2. Шифрование данных с помощью HDFS Transparent Encryption.
3. Kubernetes обеспечивает безопасность путем реализации механизмов аутентификации, авторизации и управления доступом к ресурсам в кластере.

**Выводы:**

1. Созданная архитектура хранилища данных обеспечивает возможности обработки и хранения растущего объема разнообразной информации.

2. Благодаря HDFS архитектура является масштабируемой и отказоустойчивой.

3. Для хранения данных больше подходит «снежинка», так как занимает меньшее дисковое пространство и имеет лучшую целостность данных.

4. Архитектура может работать со структурированными, полуструктурированными и неструктурированными данными.