**Задание**:

1. Выберите одно или несколько хранилищ данных (реляционная база данных, NoSQL, колоночное хранилище и т. д.), которая наилучшим образом соответствуют требованиям компании.

2. Обоснуй свой выбор, предоставив аргументы, почему данное хранилище данных является оптимальным для конкретных требований.

3. Опишите структуру данных, которые будут храниться в выбранном хранилище (таблицы, колонки и связи).

4. Предложи схему интеграции данных из разных источников в хранилище.

5. Опиши, как бы вы настроили систему для обработки данных в реальном времени.

**Решение**:

**Общая концепция.**

Для таких требований компании в сфере e-commerce предлагаю рассмотреть решения на основе облачных технологий, таких как облачные системы управления базами данных (DBaaS) и облачные аналитические платформы. Например, использование системы управления данными вроде Google Cloud Platform, Amazon Web Services или Microsoft Azure позволит обеспечить масштабируемость и обработку больших объемов данных. Также стоит рассмотреть инструменты для реализации аналитических запросов, такие как Apache Hadoop, Apache Spark или аналогичные решения, которые обеспечат высокую производительность и возможность работы с большими объемами данных.

**Вопрос 1 + Вопрос 2.**

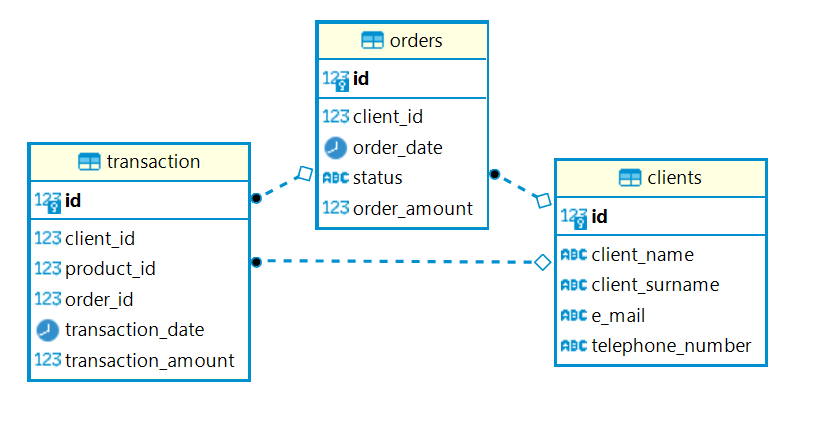
Предлагается не ограничиваться выбором одной архитектуры БД, так как возможно наличие разнородной информации в e-commerce компании. Для эффективного управления большими объемами данных и обеспечения масштабируемости с ростом нагрузки рекомендуется комбинация реляционных баз данных и NoSQL баз данных.

1. Реляционные базы данных (например, PostgreSQL) обеспечат структурированное хранение данных о клиентах, инвентаре и транзакциях, что позволит обрабатывать транзакционные операции и обеспечивать целостность данных. Это по своей природе структурированная информация.
2. NoSQL базы данных, такие как MongoDB или Cassandra, обеспечат гибкость и масштабируемость для обработки и хранения больших объемов неструктурированных данных, таких как данные клиентских взаимодействий, логи веб-серверов, данные о складе, информация о товарах и данные маркетинговых кампаний. Это по своей природе неструктурированная информация, которая может содержать данные вплоть до картинок товаров.

Комбинация реляционных и NoSQL баз данных позволит обеспечить эффективное хранение и обработку разнообразных данных, а также обеспечит возможность проведения сложных аналитических запросов для выявления тенденций продаж и сегментации клиентов (например, с применением Apache Hadoop или Apache Spark)

**Вопрос 3.**

Конечная структура данных будет зависеть от специфических требований и бизнес-процессов компании. Однако, сформируем синтетический пример для компании e-commerce, используя комбинацию реляционной базы данных и NoSQL базы данных.



**create** **table** clients

(

id **integer** **primary** **key**,

client\_name **text**,

client\_surname **text**,

e\_mail **text**,

telephone\_number **text**

);

**create** **table** orders

(

id **integer** **primary** **key**,

client\_id **integer**,

order\_date **date**,

status **text**,

order\_amount **integer**,

**foreign** **key** (client\_id) **references** clients(id)

);

**create** **table** **transaction**

(

id **integer** **primary** **key**,

client\_id **integer**,

product\_id **integer**,

order\_id **integer**,

transaction\_date **date**,

transaction\_amount **real**,

**foreign** **key** (client\_id) **references** clients(id),

**foreign** **key** (order\_id) **references** orders(id)

)

*Коллекция "Товары" в NoSQL базе данных (например, MongoDB):*

{

"\_id": ObjectId("61fb50e985b014f16c725e94"),

"название": "Футболка",

"цена": 20,

"остаток": 100

},

{

"\_id": ObjectId("61fb50e985b014f16c725e95"),

"название": "Джинсы",

"цена": 50,

"остаток": 50

}

*Коллекция "Логи взаимодействий клиентов":*

{

"client\_id": "123456",

"Действие": "Просмотр страницы товара",

"Дата и время": "2023-10-30 14:30:00",

"Дополнительная информация": "..."

}

*Коллекция "Данные маркетинговых кампаний":*

{

"company\_id": "789012",

"Тип кампании": "Электронная рассылка",

"Дата начала": "2023-10-01",

"Дата окончания": "2023-10-31",

"Статистика": { "Открытия": 100, "Клики": 20, "Конверсии": 5 }

}

Такая структура данных обеспечит эффективное хранение и управление данными о клиентах, продуктах, транзакциях, а также о взаимодействиях клиентов и маркетинговых кампаниях.

**Вопрос 4.**

Потребуется интеграция из онлайн-платформы продаж, системы учета клиентов и системы учета склада. Для этого необходимо создать механизмы для сбора, преобразования и загрузки данных (ETL). Для эффективности можно использовать несколько шагов:

***Определение источников данных***. Определить основные источники данных, такие как онлайн-платформа продаж, системы управления клиентами (CRM), системы управления товаром (склад) и маркетинговые инструменты (Директ, например).

***Создание схемы данных.*** Создать схему данных для каждого источника данных, определив ключевые поля и связи между данными. Это поможет понять, как данные будут сопоставляться и интегрироваться и в каком формате.

***Использование ETL-процесса.*** Используем процесс для интеграции данных. Извлекаем данные из разных источников, преобразуем их в соответствии с целевыми схемами данных и загружаем в реляционную и NoSQL базы данных.

***Регулярное обновление данных***. Потребуется настроить автоматизированный процесс обновления данных, чтобы регулярно обновлять информацию из разных источников. Это обеспечит актуальность данных и минимизирует возможные задержки.

***Установка мониторинга.*** Установим систему мониторинга для отслеживания процесса интеграции данных. Мониторинг поможет обнаруживать возможные ошибки или проблемы в процессе обновления данных, что позволит оперативно реагировать на них. Для обеспечения аналитической обработки данных можно связать наше хранилище с инструментами аналитики, такими как Apache Spark или другими BI-системами, для проведения анализа данных и отчетности.

***Резервное копирование данных***. Организуем регулярное создание резервных копий данных, чтобы избежать потери информации в случае сбоев или ошибок в процессе интеграции. Это обеспечит сохранность данных и защитит их от потенциальных угроз.

**Вопрос 5.**

Для обработки данных в реальном времени в e-commerce компании можно использовать следующие шаги:

***Выбор подходящего инструмента.*** Потребуется первоначально выбрать подходящий инструмент для обработки данных в реальном времени, такой как Apache Kafka, Apache Storm, Apache Flink или аналогичные технологии, которые обеспечат эффективную обработку потоков данных.

***Разработка архитектуры потоковых данных.*** Разработка архитектуры потоковых данных, определив источники данных, потоки обработки и конечные цели обработки данных. Это поможет оптимизировать процесс обработки данных и снизить задержки.

***Установка системы сбора данных.*** Настраивается система сбора данных для непрерывного мониторинга и сбора данных о заказах, клиентах и складской информации. Это обеспечит непрерывный поток данных для последующей обработки. В том числе устанавливается система мониторинга оповещений для отслеживания производительности и надежности системы обработки данных в реальном времени, что поможет обнаружить проблемы и сбои в работе системы и оперативно на них реагировать.

***Настройка обработки данных в реальном времени***. Установка шаблонов обработки данных, фильтры и трансформации, необходимых для обработки потоков данных.

***Тестирование и оптимизация производительности.*** Проводим тщательное тестирование системы обработки данных в реальном времени для определения узких мест и возможных улучшений производительности. По итогам тестирования оптимизируем процессы обработки данных для повышения эффективности системы.

**Архитектура. Мое предложение.**

Для компании e-commerce, которая собирает миллионы записей о продажах, клиентах и инвентаре ежедневно, а также требует возможности проведения сложных аналитических запросов и обработки данных в реальном времени, предлагаю следующую архитектуру приложения:

1. Фронтенд:
   * Интерфейс онлайн-платформы для клиентов (JS).
   * Панель администратора для управления данными, продуктами и заказами.
2. Бэкенд:
   * Обработка заказов и транзакций.
   * Управление клиентскими данными и товарами.
   * Сбор данных о клиентах, заказах, транзакциях.
   * Система обработки данных в реальном времени (фильтры).
3. Хранилище данных:
   * Реляционная база данных (например, PostgreSQL) для структурированных данных о клиентах, продуктах, заказах и транзакциях.
   * NoSQL база данных (например, MongoDB) для хранения неструктурированных данных о товарах и маркетинговых кампаниях.
4. Система аналитики:
   * Apache Hadoop или Spark для сложных аналитических запросов и обработки больших объемов данных.
   * Инструменты бизнес-аналитики для выявления тенденций продаж, сегментации клиентов и оценки эффективности маркетинговых кампаний.
5. Система обработки данных в реальном времени:
   * Использование Apache Kafka, Apache Storm, или аналогичных технологий для обработки данных в реальном времени, мониторинга заказов и складской информации.
6. Инфраструктура в облаке:
   * Использование облачных технологий, таких как Google Cloud Platform, Amazon Web Services или Microsoft Azure, для обеспечения масштабируемости и надежности системы.

Такая архитектура приложения обеспечит управление данными, высокую производительность, а также обработку данных в реальном времени для мониторинга заказов и складской информации.