**Домашнее задание по теме «Выбор хранилища данных»**

1. **Анализ задачи выявил следующие особенности:**

5 млн. записей о продажах в месяц. Это 5000000 / 30 дней / 24 часа / 60 минут / 60 секунд -->> ~ 2 записи в секунду. Трафик записи не большой. Даже с учётом роста объёма данных в будущем. При этом следует учитывать то, что для совершения покупки (бронирования/заказа) покупатель может просмотреть не один десяток (а то и сотен) товаров! Соответственно, запросы на чтение в БД будут поступать гораздо чаще. Причём частота запросов на запись определена усреднённо (без учёта времени суток, праздников и т.д.), т.е. в период пиковых нагрузок частота запросов на чтение будет довольно приличной.

10 миллионов клиентов и 25 миллионов артикулов реализуемой номенклатуры (с учётом перспектив роста бизнеса) сразу накладывают ограничения на возможность применения СУБД с малым максимальным количеством строк в одной таблице.

Сложность предстоящих аналитических запросов обуславливает многочисленные соединения / объединения таблиц базы данных, возможно с использованием хранения информации в JSON-формате. Рост объёма в сфере электронной коммерции подразумевает расширение ассортимента (записей номенклатуры), клиентской базы (записей клиентов) и, как следствие, увеличение продаж (записей о продажах). Структуру таблиц и полей хранилища необходимо спрогнозировать с учётом возможного расширения реализуемой номенклатуры.

Анализ данных в режиме реального времени также ужесточает требования к скорости доступа к данным и их обработке.

В проекте прослеживается транзакционная зависимость в виде: заказ – оплата – отгрузка – доставка – получение. При этом процессы формирования заказа и оплаты являются самостоятельными транзакциями.

1. **Обоснование выбора хранилища.**

Для выбора хранилища данных необходимо более детально рассмотреть варианты СУБД, их преимущества и недостатки.

Топ-10 СУБД

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| СУБД | Тип | Преимущества | Недостатки | Основное использование |
| PostgreSQL | Реляционная | Очень надёжная и стабильная, поддержка большого количества типов записи информации, отсутствие ограничений по объёму базы и количеству записей | Сложности при настройке у новичков, повышенное потребление ресурсов. | Бэкенд, BigData, аналитические запросы. Самая популярная и универсальная СУБД. |
| MySQL | Реляционная | Простота установки и администрирования пользователей, высокая скорость работы | Ограничение – 50 млн. строк в таблице. | Бэкенд, небольшие корпоративные БД |
| Microsoft SQL Server | Реляционная | Надежная, быстрый доступ к данным, высокая производительность | Высокая стоимость, ограниченная масштабируемость, под Windows (в основном) | Приложения, интегрированные с технологиями Windows |
| SQLite | Реляционная | Компактная, простая | Файловая, ограниченный функционал и возможности | Маленькие проекты |
| MongoDB | NoSQL | Простой доступ к данным, простая совместимость с другими моделями данных, горизонтально масштабируемая, встроенное кэширование | Большое потребление памяти, нет связей между документами и коллекциями, плохая защищенность, сложный процесс перевода на другие языки запросов, медленное восстановление после аппаратных сбоев, не поддерживает сложные JOIN | BigData, датчики мониторинга, каталоги товаров электронной коммерции, данные без жёсткой структуры и связей |
| Oracle Database | Реляционная | Высокая производительность | Дороговизна, сложность. В 2022 году компания Oracle покинула российский рынок. | Купный бизнес, резервные копии других БД |
| DB2 | Реляционная | Расширенные возможности аналитики, хранение JSON, оптимизация запросов | Мало специалистов, высокая стоимость | Крупный бизнес и правительственные организации |
| Redis | NoSQL | Быстрая, допускает массовую обработку данных (до 1 ГБ для одной записи), встроенное кэширование | Полная зависимость от ОЗУ, нет поддержки языка запросов или объединений | Электронная коммерция, интернет вещей |
| Apache Cassandra | NoSQL | Безопасность данных, гибкость и внесение изменений | Требуются серьёзные вычислительные ресурсы | Центры обработки данных, электронная коммерция |
| ElasticSearch | NoSQL | Масштабируемая архитектура, быстрая обработка данных | Отсутствие мульти языковой поддержки, поддерживаются только форматы документа JSON, ограниченные инструменты мониторинга | Поисковая СУБД |

Надёжность и стабильность, поддержка большого количества типов записи информации, отсутствие ограничений по объёму базы и количеству записей, популярность (и, как следствие, более простая поддержка работающей системы) выделяют реляционную базу данных PostgreSQL в качестве основного претендента на ядро хранилища данных. По своим характеристикам она должна обеспечить требования по объёму хранимых данных, проведению аналитических запросов и масштабируемости (с учётом наличия ограничений на структуру данных в номенклатуре).

Однако, необходимость постоянного анализа данных в режиме реального времени для отслеживания заказов и складской аналитики (для минимизации возможности продажи отсутствующих или недоступных более товаров), а также сложность изменения больших по объёму данных таблиц в случае добавления / удаления новых полей ставит под сомнение возможность реализации схемы с использованием одной лишь СУБД PostgreSQL.

Не предназначенность СУБД колоночного типа для работы с транзакциями исключает данный тип из дальнейшего рассмотрения. К тому же в колоночном хранилище процесс записи проходит гораздо медленнее, чем даже в реляционной СУБД.

Базы данных NoSQL (NotOnlySQL) не должны следовать правилам ACID (атомарность, согласованность, изоляция и долговечность), поэтому их скорость работы превышает скорость работы реляционных СУБД. К тому же более гибкие в плане масштабирования, т.к. данные в них хранятся в JSON-подобном формате key-value. Однако у СУБД данного типа гораздо меньше возможностей в плане реализации сложных аналитических запросов к данным.

Исходя из вышеизложенного, предлагаю к реализации комбинированную схему.

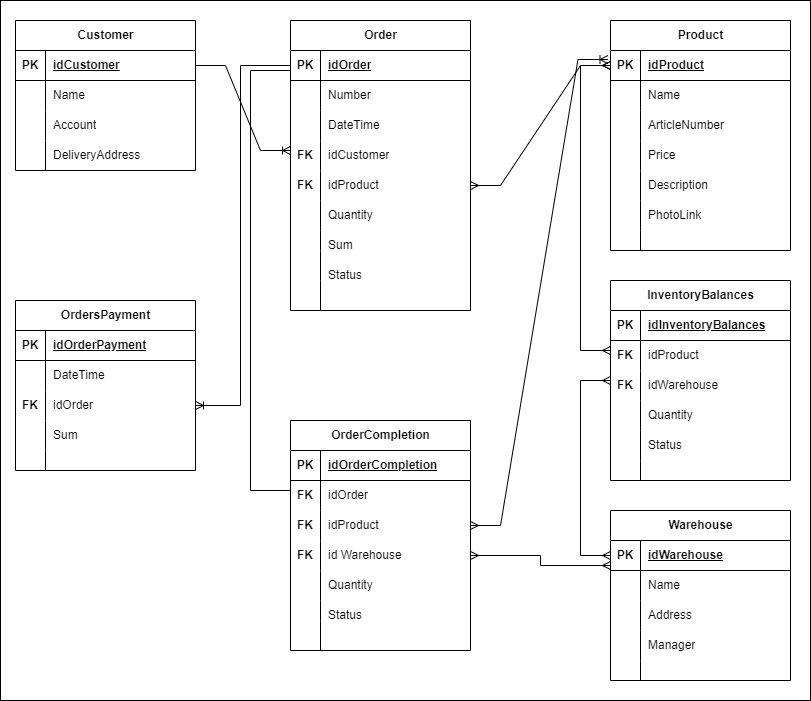
NoSQL база данных: транзакционные запросы, хранение информации о номенклатуре, клиентах и заказах. Из приведенных выше СУБД данного типа предлагаю остановить выбор на СУБД **Apache Cassandra**. В случае возможного ухода с рынка России данную СУБД можно частично заменить российским аналогом базы данных реального времени (I-DS/TSDB, TL.Solutions, ПО «Диспетчер», SIAD/SQL 6, ПАМ БДРВ, СУБД Линтер, Polymatica Analytics).

Реляционная база данных: аналитические запросы. С учётом всех преимуществ и недостатков, указанных выше, предлагаю остановить выбор на СУБД **PostgreSQL**. К тому же возможность работы с данными в JSON-формате должна упростить механизм обмена с NoSQL базой данных Apache Cassandra.

1. **Структура данных, которые будут храниться в хранилище**

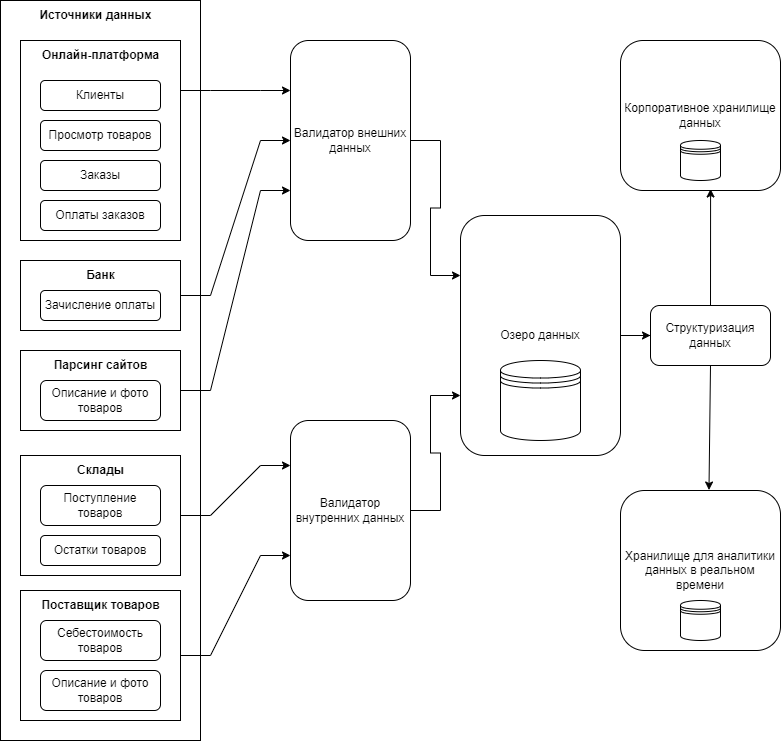
Сущности: Клиент (Customer), Заказ (Order), Товар (Product), Склад (Warehouse), Остатки запасов (InventoryBalances), Оплата заказа (OrdersPayment), Комплектация заказа (OrderCompletion).

Структура данных представлена в виде ER-диаграммы.



1. **Схема интеграции данных из разных источников в хранилище**

Источники данных: Онлайн-платформа (клиенты, просмотр товаров, заказы, оплаты заказов), Склады (поступление товаров, остатки товаров), Банк (зачисление оплаты), Парсинг сайтов (описание и фото товаров), Поставщик товаров (себестоимость и продажная цена товаров, описание и фото товаров).



1. **Настройка системы для обработки данных в реальном времени**

Данные (источники) –> предобработка (валидация) –> Data Lake –> структуризация и стандартизация под KHD -> KHD. Предполагаются следующие операции в режиме реального времени:

4.1. Поступление товара на склад. Товар может быть продуктом собственного производства и/или закупленным для реализации у стороннего контрагента. Информация о товаре поступает в озеро данных, проходит валидацию (проверяется наличие артикула, при необходимости создаётся новая запись в таблице Product, устанавливается цена товара), и записывается в КХД (по складам).

4.2. Просмотр товара клиентом. Посредством поисковой системы онлайн-платформы клиент направляет запрос о продукте в КХД. Из таблиц Product и InventoryBalances клиенту готовится ответ о товаре и товарных остатках на дату и время запроса.

4.3. Заказ товара клиентом. Посредством интерфейса онлайн-платформы клиент делает заказ товара. Данные заказа валидируются (проверяется наличие необходимого клиенту количества товара на складах), статусы соответствующих полей таблицы InventoryBalances меняются, формируется запись в таблице Order.

4.4. Оплата заказа. Посредством функционала онлайн-платформы от клиента поступают данные об оплате заказа. Данные валидируются (проверяются заказ, сумма и срок оплаты) и поступают в КХД.

4.5. Комплектация и отправка заказа. После поступления информации об оплате заказа, статус соответствующих записей в таблице InventoryBalances меняется, формируется новая запись в таблице OrderCompletion, заказ комплектуется (возможно, из разных складов) и отправляется клиенту по его адресу из таблицы Customer.

1. **Команда, которая должна реализовывать схему:**
   1. Архитектор данных (Data Architect) – 1,
   2. Инженер данных (Data Engineer) – 2 (по одному на каждую СУБД),
   3. Аналитик данных (Data analyst) – 1,
   4. Разработчик на ЯП (Software engineer) – 2 (по одному на код для каждой СУБД),
   5. Системный администратор - 1 (в случае реализации on-premise или colocation вариантов),
   6. Разработчик бизнес-аналитики (Business Intelligence (BI) Developer) – 1,
   7. Продукт-менеджер (Product-manager) – 1.
2. **Команда, которая должна поддерживать схему**
   1. Администратор баз данных (Database Administrator (DBA)) – 1 (постоянно),
   2. Специалист по данным (Data Scientist) - 1 (в случаях форс-мажора и/или модернизации схемы),
   3. Архитектор данных (Data Architect) - 1 (в случаях модернизации схемы),
   4. Системный администратор - 1 (в случае реализации on-premise или colocation вариантов).
3. **Команда Data-Science (укрупнённо):**
4. **Данные**
   1. Архитектор данных (Data Architect). Правила, политики, стандарты, модели, интеграция, доступность. Инструменты для реализации цели. Создаёт новые системы БД, разрабатывает способы повышения производительности и функциональности существующих систем и предоставляет доступ администраторам и аналитика БД. Отвечает за проектирование, реализацию и управление архитектуры данных организации.
   2. Инженер данных (Data Engineer). Получение, обработка и хранение данных. Реализация того, что спроектировал архитектор данных. Создаёт и поддерживает конвейеры данных, которые делают информацию доступной для специалистов по данным. Выполняет пакетную или оперативную обработку собранных и сохранённых данных.
   3. Менеджер данных (Data Manager). Отвечает за поток данных, процессы и координацию людей, где это уместно.
   4. Администратор баз данных (Database Administrator (DBA)). Обслуживание базы данных. Резервное копирование, безопасность, моделирование, доступ, миграции. Разрабатывает требования к БД, её проектирование, реализацию, эффективное использование и сопровождение, включая управление учётными записями пользователей БД и защиту от несанкционированного доступа.
5. **Обработка данных**
   1. Специалист по данным (Data Scientist). Определение бизнес-вопросов, преобразование данных, тренировка и настройка моделей ML, оценка результатов, отчёты, визуализация данных. Анализирует большие объёмы сложной и необработанной информации с целью нахождения закономерностей, которые будут способствовать развитию организаций и принятию важных бизнес-решений (статистика, визуализация, алгоритмы). Работает над предсказанием будущего поведения.
   2. Аналитик данных (Data analyst). Статистический анализ, отчёты, визуализация данных. Преобразует и манипулирует большими наборами данных для соответствия желаемому анализу компаний. Отвечает за выявление, понимание и интерпретацию любых закономерностей или тенденций, обнаруженных в сложных наборах данных. Помогает в процессе принятия решений. Работает с существующими данными.
   3. Статистик (Statistician). Собирает, анализирует и интерпретирует данные, что позволяет им выявлять тенденции и взаимосвязи, которые можно использовать для принятия организационных решений.
   4. Разработчик бизнес-аналитики (Business Intelligence (BI) Developer). Структурирование и представление полученных данных в понятном виде для менеджмента. Проектирует и разрабатывает методы, позволяющие бизнес-пользователям находить необходимую информацию для улучшения бизнеса.
6. **ML**
   1. Специалист по ML (ML Scientist). Исследует новые подходы к обработке данных и алгоритмы, которые будут применяться в адаптивных системах.
   2. Инженер по машинному обучению (Machine Learning Engineer). Специалист по данным, специализирующийся на ML. Автоматизация моделей ML, проектирование, разработка, тестирование ML-систем, тренировка и последующая поддержка модели ML. Разрабатывает системы ИИ.
   3. Инженер по исследованиям и разработкам ML (ML research and development engineer). Разработка новых алгоритмов ML.
7. **Связь с заказчиком и общественностью**
   1. Журналист данных (Data Journalist). Преобразование данных в захватывающие истории. Перенос выводимых данных в нужный контекст.
   2. Эксперт по предметам (Subject Matter Expert). Применение аналитических инсайтов на практике.
   3. Корпоративный архитектор (Enterprise Architect). Отвечает за согласование идей компании с правильными технологиями с целью оказания помощи в оптимальной реализации целей и идей.
   4. ML Ops. Отвечает за надёжное и эффективное развёртывание и поддержание моделей обучения на производстве.
   5. Продукт-менеджер (Product-manager). Отвечает за соответствие требований заказчика реальным возможностям команды.
8. **Frontend**
   1. Системный аналитик (System Analyst). Занимается переводом запросов заказчика к IT-системе и планированием автоматизации процессов.
   2. Архитектор приложения (Application Architect). Проектирует архитектуру приложений, отслеживает поведение приложений, используемых в бизнесе, и их взаимодействие друг с другом и пользователем.
   3. Разработчик на ЯП (Software engineer). Разрабатывает приложения, генерирующие данные, которые могут использоваться специалистами по данным.