# **Задание 1**

Крупная строительная компания, которая также занимается проектированием и девелопментом, решила создать правильную архитектуру для работы с данными. Ниже представлены задачи, которые необходимо решить для каждой предметной области.

Какие типы СУБД, на ваш взгляд, лучше всего подойдут для решения этих задач и почему?

1.1. Бюджетирование проектов с дальнейшим формированием финансовых аналитических отчётов и прогнозирования рисков. СУБД должна гарантировать целостность и чёткую структуру данных.

1.1.\* Хеширование стало занимать длительно время, какое API можно использовать для ускорения работы?

1.2. Под каждый девелоперский проект создаётся отдельный лендинг, и все данные по лидам стекаются в CRM к маркетологам и менеджерам по продажам. Какой тип СУБД лучше использовать для лендингов и для CRM? СУБД должны быть гибкими и быстрыми.

1.2.\* Можно ли эту задачу закрыть одной СУБД? И если да, то какой именно СУБД и какой реализацией?

1.3. Отдел контроля качества решил создать базу по корпоративным нормам и правилам, обучающему материалу и так далее, сформированную согласно структуре компании. СУБД должна иметь простую и понятную структуру.

1.3.\* Можно ли под эту задачу использовать уже существующую СУБД из задач выше и если да, то как лучше это реализовать?

1.4. Департамент логистики нуждается в решении задач по быстрому формированию маршрутов доставки материалов по объектам и распределению курьеров по маршрутам с доставкой документов. СУБД должна уметь быстро работать со связями.

1.4.\* Можно ли к этой СУБД подключить отдел закупок или для них лучше сформировать свою СУБД в связке с СУБД логистов?

1.5.\* Можно ли все перечисленные выше задачи решить, используя одну СУБД? Если да, то какую именно?

*Приведите ответ в свободной форме.*

# **Решение 1**

1.1 SQL, например, Postgresql. Для хеширования - openssl или bcrypt.

1.2 NoSQL, например МongoDB.

1.3 Снова NoSQL, та же монго. Если что-то максимально простое, возможно redis?

1.4 Графовая СУБД, типа amazon neptune. Она пойдет и логистам и закупщикам.

1.5 Монго, полагаю. Но если речь идет про масштабный проект правильнее будет для каждой задачи использовать специализированный инструмент. Это более затратно, поскольку необходимо один в один синхронизировать и потом весь этот зоопарк поддерживать. Но это и более выгодно в плане производительности системы в целом.

# **Задание 2**

2.1. Пользователь пополняет баланс счёта телефона, распишите пошагово, какие действия должны произойти для того, чтобы транзакция завершилась успешно. Ориентируйтесь на шесть действий.

2.1.\* Какие действия должны произойти, если пополнение счёта телефона происходило бы через автоплатёж?

*Приведите ответ в свободной форме.*

# **Решение 2**

Инициирование транзакции:

Транзакция начинается с запроса пользователя на пополнение баланса телефона. В этот момент начинается новая транзакция.

Аутентификация пользователя:

Система аутентифицирует пользователя, проверяя его учетные данные. Этот шаг важен для обеспечения безопасности и предотвращения несанкционированных транзакций.

Проверка средств:

Происходит проверка, есть ли у пользователя достаточно средств на счете для пополнения баланса на указанную сумму. Это гарантирует выполнение условий целостности данных.

Резервирование средств:

Если проверка успешна, система временно резервирует сумму транзакции. Это предотвращает конфликты и устанавливает блокировку на изменение зарезервированных средств до завершения транзакции.

Завершение транзакции:

Происходит окончательное списание средств или изменение баланса. Этот шаг включает в себя обновление данных в базе данных, что подчиняется принципам атомарности.

Отправка уведомления:

Пользователю отправляется уведомление о успешной транзакции. Этот шаг важен для информирования пользователя и поддержания согласованности данных.

# **Задание 3**

3.1. Напишите пять преимуществ SQL-систем по отношению к NoSQL.

3.1.\* Какие, на ваш взгляд, преимущества у NewSQL систем перед SQL и NoSQL.

*Приведите ответ в свободной форме.*

# **Решение 3**

* 1. Структурированный язык запросов (SQL): SQL предоставляет мощный и удобный язык запросов, что упрощает извлечение данных из базы данных и выполнение сложных операций. Это особенно полезно при работе с связанными данными и сложными запросами.
* 2. Схема данных: SQL-базы данных используют строгую схему данных, что обеспечивает целостность данных и предотвращает некорректные записи. Это полезно в приложениях, где структура данных статична и должна быть хорошо определена.
* 3. Транзакции и ACID-свойства: SQL-системы обеспечивают поддержку транзакций и соблюдение ACID-свойств (Atomicity, Consistency, Isolation, Durability). Это гарантирует надежность и целостность данных в случае сбоев или ошибок.
* 4. Нормализация данных: SQL поддерживает нормализацию данных, что позволяет эффективно управлять повторяющейся информацией и минимизировать избыточность данных. Это особенно важно для поддержания структурной целостности в сложных системах.
* 5. Широкое использование и поддержка: SQL является стандартом для работы с реляционными базами данных, и существует множество различных реализаций SQL-систем. Это обеспечивает широкую поддержку, множество инструментов и богатый опыт разработчиков.

# **Задание 4**

Необходимо производить большое количество вычислений при работе с огромным количеством данных, под эту задачу выделено 1000 машин.

На основе какого критерия будете выбирать тип СУБД и какая модель *распределённых вычислений* здесь справится лучше всего и почему?

*Приведите ответ в свободной форме.*

# **Решение 4**

1. Характер данных. Если они структурированы - SQL с поддержкой горизонтального масштабирования, в противном случае NoSQL.
2. Типы операций. В зависимости от выбора операций будет меняться и тип бд. Например, если требуется много аналитических операций, то потребуется Spark или MapReduce.
3. Требования к транзакциям. Если есть требования к целостности данных, то стоит выбрать, например, Postgre.