Проект 1: Каталог большого маркетплейса

Одной очень перспективной компании по разработке ПО поступил заказ, в рамках которого необходимо разработать базу данных, представляющую каталог маркетплейса *«nozO»* — онлайн-платформы для продажи и покупки товаров. Заказчик не предъявил никаких ограничений на используемые технологии, однако предъявил некоторые требования к разрабатываемой системе:

- База данных должна уметь хранить огромное множество различных видов товаров.
- У каждого вида товара существуют свои характеристики и описание, однако некоторые представители вида товаров могут содержать дополнительные поля-характеристики, которые могут быть уникальными для конкретного товара.
- В дальнейшем могут появляться новые виды товаров, и, наоборот, товаров конкретного вида может вовсе не стать.
- База данных должна обеспечивать быстрый поиск по названиям товаров, категориям, характеристикам и их значениям, а также поддерживать фильтрацию и сортировку результатов поиска по нескольким параметрам.

Проанализировав требования заказчика, командой разработчиков было принято четкое решение **НЕ** использовать реляционную базу данных в этом проекте. Вот причины, которые озвучил технический руководитель команды разработки:

- Сложность поддержки актуальной структуры реляционной базы данных для хранения всех возможных видов товаров, их характеристик.
- Отсутствие гибкости в работе с динамическими и уникальными характеристиками товаров.
- Неудобство SQL для ответов на запросы покупателей, содержащие различные фильтры и ключевые слова. Сложность обработки вводимых пользователем данных (например, нужно учитывать возможные ошибки в написании).

Проект 2: Слады большого маркетплейса

После успешного внедрения базы данных каталога товаров дела маркетплейса *«nozO»* пошли в гору! У компании открываются множество огромных складов по всей стране, а вместе с ними появляется большая ответственность перед продавцами/покупателями за сохранность товара, отсутствие задержек и потеряшек. *«nozO»* снова обратилась к одной очень перспективной компании по разработке ПО, чтобы та разработала для них базу данных по учету складов.

На встрече заказчик предъявил несколько требований:

- Сразу несколько складов должны работать с одной базой данных.
- База данных должна иметь четкую структуру, строго определенную схему.
- Должен вестись не только учет товаров и запасов, но и их перемещение между складами. При этом не должно возникать никаких проблем, связанных с консистентностью данных: ни одна операция не должна остаться выполненной лишь на половину.
- Должна быть возможность легко генерировать отчеты о состоянии запасов, перемещениях и общих логистических операциях.

На этот раз командой разработчиков было решено реализовывать реляционную базу данных по нескольким причинам:

- Реляционная модель позволяет легко отразить взаимосвязи между сущностями (складами, товарами с их уникальными номерами, поставщиками) с помощью таблиц и связей между ними.
- Механизмы транзакций (ACID) в реляционных базах данных гарантируют, что каждая операция (например, перемещение товара между складами) будет выполнена полностью или не выполнена вовсе, обеспечивая целостность данных.
- Язык SQL позволяет быстро проводить запросы для поддержания статистики, подсчета недостач и так далее.