Как всё начиналось

В 60-х годах прошлого столетия появилась необходимость в надежной модели хранения и обработки данных. В первую очередь эти данные генерировались банками и финансовыми организациями. В то время не существовало единых стандартов работы с данными и моделями, да и работа как таковая заключалась в ручном упорядочении и организации хранящейся информации.

У банков худо-бедно получалось записывать информацию о транзакциях в виде файлов в заранее подготовленную структуру. У каждой организации было собственное понимание того, как все это должно выглядеть и работать. Не было таких понятий, как консистентность (англ. data consistency), целостности данных (англ. data integrity). В файлах часто встречались дубликаты данных клиентов и их транзакций, которые необходимо было каким-то образом уточнять и приводить в порядок, делалось это в основном вручную. В целом все проблемы того времени в отношении работы с данными можно разделить на несколько основных видов:

- Представление структуры в каждом файле было различным.
- Необходимо было согласовывать данные в разных файлах, чтобы обеспечить непротиворечивость информации.
- Сложность разработки и поддержки приложений, работающих с конкретными данными, и их обновления при изменении структуры файла.

По сути, здесь мы видим антипаттерн «чистой архитектуры», который был описан Робертом Мартином (Robert C. Martin).

Следует отметить, что были попытки создания моделей, позволяющих навести порядок в данных и их обработке. Одна из таких попыток — **иерархическая модель**, в которой данные были организованы в виде древовидной структуры. Иерархическая модель была востребованной, но не гибкой. В ней каждая запись могла иметь только одного «предка», даже если отдельные записи могли иметь несколько «потомков». Из-за этого базы данных представляли только отношения «один к одному» или «один ко многим». Невозможность реализации отношения «многие ко многим» могла привести к проблемам при работе с данными и усложняла модель. Более того, вопросы консистентности данных и отсутствия дублирования информации здесь вообще не стояли. Первая иерархическая СУБД называлась IMS от IBM.

На помощь иерархической пришла **сетевая модель данных**, и уже новая концепция реализовала отношение «многие ко многим». Данный подход был предложен как спецификация модели CODASYL в рамках рабочей группы DBTG (Data Base Task Group).

Но всё это модели, которые сложно было поддерживать. Упростить задачу сбора и обработки данных смог Франк Кодд (Edgar F. Codd). Его фундаментальная работа привела к появлению реляционных баз данных, которые нужны практически всем отраслям. Кодд предложил язык Alpha для управления реляционными данными. Коллеги Кодда из IBM — Дональд Чемберлен (Donald Chamberlin) и Рэймонд Бойс (Raymond Boyce) — создали один из языков под

влиянием работы Кодда. Они назвали свой язык SEQUEL (Structured English Query Language), но изменили название на SQL из-за существующего товарного знака.

Появление реляционных БД и их эволюция

Активное развитие технологий БД началось примерно в 1970 году, когда Кодд опубликовал свою работу, послужившую основой для создания реляционной модели данных. Среди достоинств этой модели стоит выделить:

- Отсутствие дублирования данных.
- Исключение ряда ошибок и аномалий данных, которые есть в других моделях.
- Все данные представлены как факты, хранящиеся в виде отношений (relations) со столбцами (attributes) и строками (tuples).

Одной из первых РСУБД можно назвать dBase-II от компании Ashton-Tate, которая выпустила свой продукт в 1979 году. Она смогла поставить на рынок около 100 000 копий своего продукта. В итоге эта база данных стала самой популярной среди всех существовавших в то время продуктов. К слову, компанию Ashton-Tate позже приобрела фирма Borland. В целом реляционной СУБД этот продукт можно было назвать лишь с очень большой натяжкой.

Но начало было положено — и другие компании стали представлять свои продукты. Так, например, появились Oracle, Ingress и Focus.

К 1980 году сформировалось архитектурное (высокоуровневое) и инженерное (низкоуровневое) понимание того, как должна функционировать РСУБД. И пришло решение о введении стандарта (SQL Standard ISO and ANSI).

Развитие баз данных усилилось после распространения локальных сетей, а затем и сети глобальной. Сеть позволяла совместно использовать оборудование. Также пользователи хотели совместно (параллельно) работать с РСУБД, что ускорило развитие многопользовательских приложений для локальных сетей. Была создана клиент-серверная архитектура обработки данных.

С 90-х годов технология стала более дружелюбной к пользователю. По словам маркетологов, в СУБД теперь мог разобраться любой человек. Конечно, это преувеличение, но в целом направление эволюции понятно.

В это же время стали появляться первые онлайн-сервисы (например, отправка цветов, подарков, открыток, ведение блогов и т. п.). Большинство этих сервисов стали (и продолжают) работать в связке PHP + MySQL.

Распространение сотовой связи и сотовых телефонов с 1996 года привело к созданию специализированных баз данных для обработки информации в мобильном устройстве. Сейчас они эволюционировали в отдельный стек баз данных (In-Memory) и применяются либо в качестве кэша данных, находясь перед основной базой данных, либо в качестве Warm / Hot логического слоя, используясь в Lambda / Kappa архитектуре.

По моему мнению, в 2000-х произошла своеобразная эволюция, когда нереляционная модель была интегрирована в реляционную базу данных (я говорю про интеграцию XML-формата). Можно было определять модель в модели (по сути, описать любую модель можно в одном столбце таблицы). Примерами технической реализации такого «фрактального моделирования» были Oracle Nested Tables, а также тип XML, а в последствии JSON. Такие модели назвали Post-Relational Models, и можно сказать, что начали появляться зачатки работы с noSQL-моделями формата «ключ-значение / ключ-документ».

Значение и ценность данных стал постепенно осознавать бизнес, в результате чего возросла необходимость в соответствующих специалистах. Технологии БД стали использовать далеко не только для OLTP и OLAP-трафиков, но и для глубоких исследований в данных (поиск аномалий, корреляций, использование статистического аппарата и т. д.).

С 2006 года начал работать облачный сервис Amazon AWS, по словам его представителей, сейчас он насчитывает уже свыше 20 000 частных Data Lakes, построенных внутри облака.

Ну а сейчас роль баз данных возросла еще больше. Ведь данные генерирует любое умное устройство, а их становится всё больше. Это уже не только телевизоры или смартфоны, но и зубные щетки и даже чайники (IOT-трафик, не путать с Index-Organized Tables! ;)

Правда, с увеличением объема данных стало больше и узконаправленных баз данных, которые специализируются на работе с разными типами информации и моделями.

Реляционных СУБД не так много (по сравнению с нереляционными базами данных), но архитектура каждой из них уникальна. У каждой есть свои плюсы и минусы. Также можно отметить, что в мире не существует РСУБД, которая полностью бы описывала математическую реляционную модель Франка Кодда, кроме одной с именем Rel и языком Tutorial-D. Почему это так? Неужели сложность в технической реализации реляционной теории? Нет, конечно. По моему мнению, на самом деле все проще: бизнес неявно диктует свои условия реализации хранения и обработки данных. Давайте вспомним некоторые основные свойства отношений (relations) в рамках реляционной теории и сравним их с реальной жизнью.

1. **“All tuples are unique“**. Это значит, что необходимо хранить один факт о свершившимся событии из реального мира. Данное утверждение также поддерживает определение простейшего уникального ключа для отношения, который должен включать весь набор атрибутов отношения.

Реляционная теория	Реальн
$R(A, B, C) =$ 	SELECT A,B,C FROM R;
Недопустимо хранить дублирование данных	Мы можем создать д числе дубликаты

2. **“The order of the lines is irrelevant”**. Зачем нам вообще вводить такое понятие, как сортировка