Высшая школа юриспруденции
и администрирования

Институт юридического
менеджмента

Москва
2025

Хранение и получение данных

Гурьянов Марк Владимирович
«Архитектор программного обеспечения АО «ГК «Визионерс»

Ценность данных

Данные — это актив, а не инфраструктура. Их ценность сравнима с нефтью: при правильной обработке они дают мощный эффект для бизнеса.

Как нефть нуждается в переработке, так и данные. Если их просто хранить, они ничего не дадут. Но если навести порядок, понять, какие из них важны и как их использовать — можно получить серьёзную выгоду: от оптимизации процессов до новых источников дохода.

> *Data is the new oil*

Цитата была сказан Клайв Хамби в 2006 году

На ваш взгляд, умело ли компания, где вы работаете управляется с данными?

Почему и главное зачем?

”

Управленец, не понимающий принципы работы с данными — как водитель, не знающий, где взять топливо, куда его залить и зачем оно вообще необходимо.

“

Данные — не инфраструктура, а актив

Ответьте себе на вопрос: «*Кто работает с данными в вашей компании?*»

Если никто — значит, данные никому не принадлежат.

Если нет владельца — нет ответственности, нет качества, нет движения.

Снова к истории:

В 15 веке европейские колонисты обменивали золото индейцев на разные изделия, так как те не осознавали ценность этого ресурса.

DataDome

Платит за сбор данных о поведении пользователей

Яндекс.Задания

Оплачивает подготовку данных для обучения моделей

Работа с данными "по старинке"

Типичные проблемы:

Расхождения в отчетности
Требуют ручной проверки и сверки

Дублирование информации
Справочники отличаются в разных отделах

Решения на основе догадок
Отсутствие объективных данных для принятия решений

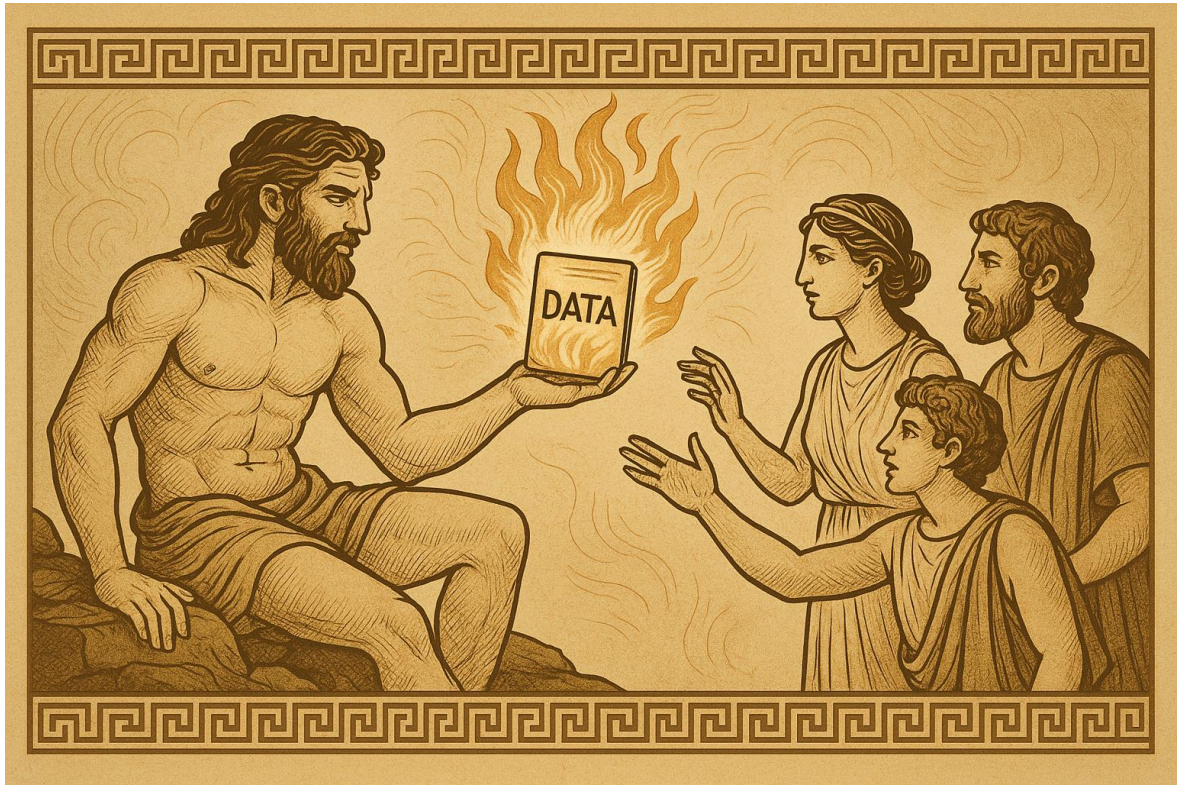
Затруднения с современными решениями
Невозможность или высокая стоимость внедрения

Длительный поиск и подготовка информации
Потеря времени на рутинные операции

Это лишь верхушка айсберга проблем, с которыми сталкиваются компании, не имеющие системного подхода к работе с данными.

Мифы и барьеры

- "Это сложно, дорого и долго"
- "Это задача IT"
- " У нас всё уже есть "
- " Главное — собрать всё"
- "Мы не IT-компания — нам не нужно это всерьёз"



Миф о том, как Прометей дал людям не огонь, а материалы этой лекции

Как меняется ваша роль?

"Современный управленец должен быть не просто потребителем данных, но и активным участником процесса их сбора, обработки и анализа."

Ориентируется в данных

Знает, какие данные есть в компании, как они хранятся и откуда берутся

Требует данные

Использует данные как основу для принятия решений, а не полагается только на интуицию

Создаёт культуру

Формирует культуру ответственности за качество данных в своей команде

Общается на одном языке с IT

Общается на одном языке с техническими специалистами об архитектуре, метриках и аналитике

Эволюция хранения данных (1970-1999)

1970-1979

Зарождение реляционной модели

1970
Статья Эдгара Кодда о реляционной модели

1974-1979
IBM разрабатывает System R

1975
Появление концепции ACID

1970-е
Разработка SQL в IBM

1980-1989

Коммерциализация реляционных БД

1979
Oracle выпускает первую коммерческую SQL СУБД

1982
IBM DB2

1986-1987
SQL становится стандартом ANSI и ISO

Конец 1980-х
Появление OLTP

1990-1999

Интернет-революция

1995
MySQL для веб-приложений

1996
PostgreSQL как объектно-реляционная реляционная СУБД

Середина 1990-х
Data Warehousing

1998
Основание Google

Каждая волна была ответом на конкретные технологические и бизнес-вызовы своего времени.

Эволюция хранения данных (2000-2018)

2000-2006

Кризис масштабирования

2003
Google File System (GFS)

2004
Google MapReduce - революция в распределенной обработке

2005
Apache Hadoop

2006
"Data is the new oil" (Клайв Хамби)

2007-2012

NoSQL революция

2007-2009
Amazon DynamoDB, CouchDB, MongoDB, Redis

2008
Cassandra (Facebook), Google App Engine

2011
Apache Kafka (LinkedIn), начало Apache Spark

2012
Facebook достигает 1 млрд пользователей

2013-2018

Облачная трансформация

2013
Snowflake как cloud-native платформа

2014
Apache Spark, Amazon Redshift

2016-2017
Google BigQuery с ML, Azure Cosmos DB, Kubernetes

2018
GDPR влияет на архитектуру данных

Этот период характеризуется переходом от монолитных решений к распределенным системам и облачным сервисам.

Эволюция хранения данных (2019-2024)

Эра Data Lakehouse и AI-революции

2019
Databricks представляет концепцию Data Lakehouse

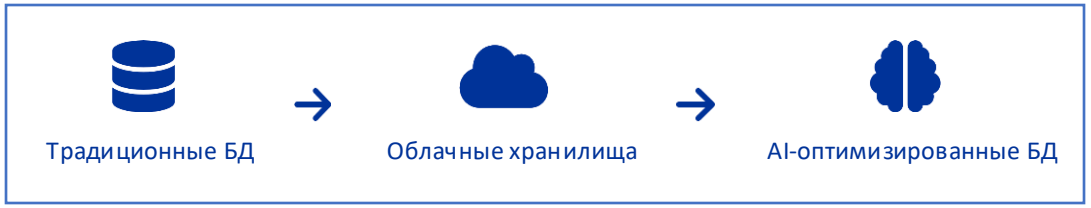
2020
Пандемия ускоряет цифровизацию, Snowflake IPO ($3.4 млрд)

2021
dbt революционизирует ELT, развитие modern data stack

2022
ChatGPT создает спрос на векторные БД

2023
LLM и RAG архитектуры, PostgreSQL pgvector

2024
Массовое внедрение векторных БД, real-time OLAP



Данные становятся основой для AI-приложений, требуя новых специализированных решений.

Типы и источники данных

Типы данных:

Структурированные

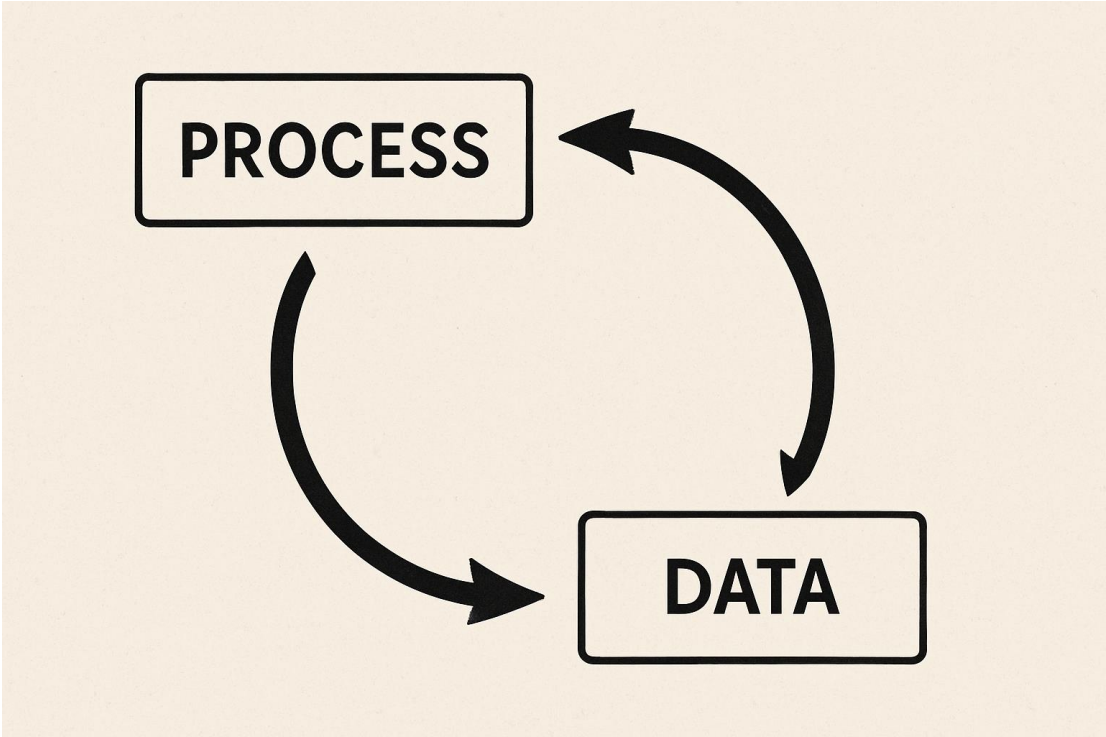
Данные с четкой схемой и фиксированной структурой (таблицы, реляционные БД)

Полуструктурированные

Данные с гибкой схемой (JSON, XML, NoSQL документы)

Неструктурированные

Данные без определенной структуры (текст, изображения, видео, аудио)



Структурированные данные

Что это такое:

- Чёткая схема
- Фиксированная структура: строки и столбцы
- Заданные типы значений: дата, число, текст, логический признак

Примеры:

- Таблица заказов: ID заказа, дата, сумма, статус
- Справочник клиентов: ФИО, дата регистрации, номер телефона
- Бухгалтерские проводки: Счёт дебета, счёт кредита, сумма, дата

Преимущества для бизнеса:

- Легко хранятся в базах данных
- Просты для анализа и отчетности
- Поддерживают строгую валидацию

Пример:

ID заказа	Дата	Клиент	Сумма	Статус
1001	01.06.2025	Иванов И.И.	5000 ₽	Оплачен
1002	02.06.2025	Петров П.П.	7500 ₽	В обработке

Полуструктурированные данные

Что это такое:

- Гибкая схема данных
- Иерархическая структура
- Самоописываемые данные с тегами или метками

Примеры:

- JSON-документы (конфигурации, API-ответы)
- XML-файлы (SOAP API, конфигурации)
- Документы в NoSQL базах данных (MongoDB)

Преимущества для бизнеса:

- Гибкость при изменении требований
- Лучшая масштабируемость
- Быстрая разработка и итерации

Пример:

```
{
 "id": 1001,
 "customer": {
  "name": "Иванов Иван",
  "email": "ivanov@example.com",
  "phone": "+7 (999) 123-4567"
 }
}
```

```
<party type="PHYSICAL" sourceSystem="AL" rawId="2">
rawId="2">
  <field name=“name">Олег </field>
  <field name="birthdate">02.01.1980</field>
  <attribute type="PHONE" rawId="AL.2.PH.1">
    <field name="type">MOBILE</field>
    <field name="number">+7 916 1234567</field>
1234567</field>
  </attribute>
</party>
```

```
spring:
 config:
  use-legacy-processing: true
 application:
  name: catalog
 version:
  major: 2
```

Неструктурированные данные

Что это такое:

- Данные без определённой структуры
- Не имеют предопределённой модели данных
- Сложно анализировать стандартными методами

Примеры:

- Текстовые документы, электронные письма
- Изображения, фотографии, сканы документов
- Видео и аудиофайлы
- Сообщения в социальных сетях

Преимущества для бизнеса:

- Требуют специальных методов обработки (ML, AI)
- Сложности с хранением и индексацией
- Содержат скрытые инсайты и ценную информацию
- Составляют до 80% всех корпоративных данных

Пример:

Текстовые документы
Отчеты, статьи, договоры

Изображения
Фотографии, сканы, графики

Медиафайлы
Видео, аудиозаписи, презентации

Источники данных

Внутренние источники

Транзакционные системы

- ERP-системы
- CRM-системы
- Кассовые системы
- Складской учет

Операционные системы

- Системы управления процессами
- Системы документооборота
- Учетные системы

Корпоративные приложения

- Корпоративная почта
- Мессенджеры
- Планировщики

Внешние источники

Открытые данные

- Государственные порталы
- Статистические службы
- Открытые датасеты

Социальные медиа

- Социальные сети
- Видеоплатформы
- Форумы и обсуждения

Партнерские данные

- Данные поставщиков
- Логистические партнеры
- Платежные системы

IoT и сенсоры

- Промышленные датчики
- Мобильные устройства
- Умные устройства

Онлайн источники

данные, поступающие в реальном времени

Офлайн источники

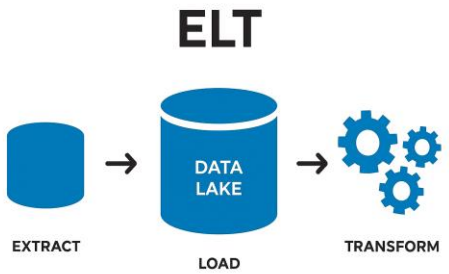
данные, которые поступают периодически, пакетами

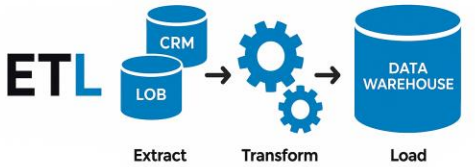
Архитектуры интеграции данных

ETL / ELT

Extract, Transform, Load / Extract, Load, Transform

- ETL: преобразование перед загрузкой
- ELT: преобразование после загрузки
- Для пакетной обработки данных

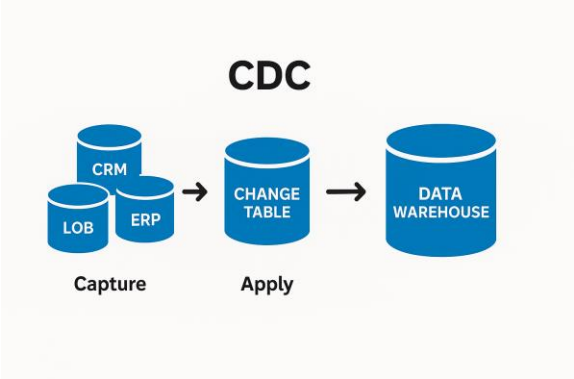




CDC (Change Data Capture)

Отслеживание изменений в данных

- Захват только изменившихся данных
- Минимальная нагрузка на источник
- Почти реальное время



OLTP и OLAP

OLTP

Системы обработки транзакций в реальном времени

- Поддерживают повседневные операции бизнеса
- Обрабатывают множество коротких транзакций
- Требуют высокой доступности и надежности
- Фокус на целостности данных (ACID)

«OLTP — это операционный пульс компании»

OLAP

Системы аналитической обработки в реальном времени

- Поддерживают принятие решений
- Многомерный анализ данных (кубы)
- Сложные агрегации и вычисления
- Фокус на производительности запросов

«OLAP — это аналитический мозг компании»

Характеристика	OLTP (Online Transaction Processing)	OLAP (Online Analytical Processing)
Назначение	Обработка транзакций	Аналитика и отчетность
Тип операций	Вставка, обновление, удаление	Сложные запросы и агрегации
Объем данных	Небольшие транзакции	Большие объемы данных
Скорость	Миллисекунды	Секунды/минуты
Нормализация	Высокая (3NF)	Низкая (денормализация)
Историчность	Текущие данные	Исторические данные
Оптимизация	Для записи	Для чтения
Примеры систем	ERP, CRM, банковские системы	BI, Data Warehouse, отчетность

Типы систем управления базами данных

Основные категории СУБД

Реляционные СУБД
Данные в виде таблиц со связями
PostgreSQL, MySQL, Oracle, SQL Server

NoSQL СУБД
Нереляционные модели данных
MongoDB, Cassandra, Redis, Neo4j

NewSQL
Сочетание SQL и масштабируемости NoSQL
Google Spanner, CockroachDB, YDB

Критерии выбора СУБД

Модель данных
Структура данных и связи между ними

Масштабируемость
Вертикальная vs горизонтальная, объемы данных

Производительность
Скорость чтения/записи, задержки, пропускная способность

Согласованность
ACID vs BASE, CAP-теорема

Доступность
Отказоустойчивость, репликация, восстановление

Экосистема
Инструменты, поддержка, сообщество, документация

Эволюция подхода к СУБД

1970-е: Реляционная модель

Эдгар Кодд предложил реляционную модель данных

Появление SQL, IBM System R, Oracle

1980-1990-е: Доминирование РСУБД

Стандартизация SQL, развитие коммерческих СУБД

Oracle, DB2, SQL Server, PostgreSQL, MySQL

2000-е: Кризис масштабирования

Рост объемов данных, проблемы с горизонтальным масштабированием

Появление распределенных систем, шардинг

2009-2015: NoSQL революция

Отказ от ACID в пользу BASE, CAP-теорема

MongoDB, Cassandra, Redis, Neo4j, Elasticsearch

2015-2020: NewSQL и полиглот

Возвращение к транзакционности с сохранением масштабируемости

Google Spanner, CockroachDB, полиглот-персистентность

2020+: Эра AI и векторных БД

Интеграция с ИИ, векторные вычисления, RAG-архитектуры

Pinecone, Weaviate, Milvus, Qdrant, ChromaDB

Ключевые тенденции

От монолита к распределенным системам

Переход от вертикального к горизонтальному масштабированию для работы с большими объемами данных

От жесткой схемы к гибкости

Эволюция от строгих схем к schema-on-read и гибким моделям данных

От ACID к BASE и обратно

Поиск баланса между согласованностью и доступностью в распределенных системах

От универсальности к специализации

Развитие специализированных СУБД для конкретных типов данных и задач

От локальных к облачным

Переход к DBaaS (Database-as-a-Service) и полностью управляемым решениям

NoSQL: Нереляционные базы данных

Что такое NoSQL?

NoSQL (Not Only SQL) — класс систем управления базами данных, которые отличаются от традиционных реляционных СУБД отказом от использования табличной схемы данных и языка SQL.

Ключевые характеристики:

- Гибкие схемы данных (schema-less)
- Горизонтальная масштабируемость
- Высокая производительность для специфических задач
- Распределенная архитектура
- Отказ от строгой согласованности (ACID) в пользу доступности (BASE)

Основные типы NoSQL СУБД

Документные (Document)

Хранение иерархических структур данных (JSON, BSON)

MongoDB, Couchbase, Firestore

Контент-системы, каталоги, профили пользователей

Ключ-значение (Key-Value)

Простейшая модель данных — пары ключ-значение

Redis, DynamoDB, Riak

Кэширование, сессии, счетчики, очереди

Колоночные (Column-family)

Данные хранятся по столбцам, а не по строкам

Cassandra, HBase, ScyllaDB

Аналитика, временные ряды, большие объемы данных

Графовые (Graph)

Хранение связей между объектами в виде графа

Neo4j, JanusGraph, ArangoDB

Социальные сети, рекомендации, сложные связи

Векторные (Vector)

Хранение и поиск векторных представлений данных

Pinecone, Weaviate, Qdrant, Milvus

Поиск по схожести, ИИ-приложения, RAG-системы

Документные СУБД

Название	Описание
MongoDB	Самая популярная документная СУБД. Простая, гибкая, поддерживает вложенные структуры.
Couchbase	Совмещает документы с возможностями кэширования. Подходит для real-time приложений.
Amazon DocumentDB	Облачная альтернатива MongoDB от Amazon, совместимая по API.
RavenDB	Поддерживает ACID-транзакции, ориентирована на .NET-разработку.
Firebase Realtime Database / Firestore	Документные БД от Google для мобильных и веб-приложений.

Документные СУБД — один из самых популярных подтипов NoSQL.

Они хранят данные в виде документов, чаще всего в формате JSON, BSON или XML. Каждый документ — это самостоятельная запись, содержащая набор полей и значений. В отличие от реляционных баз, здесь нет жёстко заданной схемы: каждый документ может иметь свою уникальную структуру, даже внутри одной коллекции.

Документы могут быть вложенными (то есть содержать другие документы), что удобно для хранения сложных объектов, таких как «профиль клиента» или «карточка товара».

Это позволяет очень гибко работать с разнородными и изменяющимися данными, не нарушая общую систему хранения.

Колоночные СУБД

Название	Описание
ClickHouse	Российская колоночная СУБД, ориентирована на высокопроизводительную аналитику.
Amazon Redshift	Облачное хранилище от AWS, активно используется для аналитики.
Apache Druid	Заточен под real-time аналитику и интерактивные панели мониторинга.
Apache Pinot	Используется в LinkedIn, оптимизирован для низкой задержки запросов.
Vertica	Коммерческая колоночная БД от Micro Focus. Высокая скорость агрегаций.

Колоночные базы данных хранят данные не построчно, как реляционные СУБД, а по столбцам. Это означает, что значения одного и того же поля (например, "сумма покупки") хранятся рядом, а не распределены по разным строкам.

Такая модель сильно ускоряет аналитические запросы, когда нужно обработать и агрегировать один или несколько столбцов по огромному объёму данных. Именно поэтому колоночные СУБД часто применяются в системах бизнес-аналитики и отчётности.

СУБД ключ-значение

Название	Описание
Redis	Лёгкое и очень быстрое хранилище в памяти, часто используется как кэш.
Amazon DynamoDB	Облачное масштабируемое Key-Value хранилище от AWS.
Riak KV	Распределённая СУБД с высокой отказоустойчивостью.
Aerospike	Высокопроизводительное решение для real-time аналитики.
LevelDB / RocksDB	Встраиваемые локальные key-value базы от Google / Meta.

Key-Value хранилища — это наиболее простая модель баз данных, в которой каждая запись состоит всего из двух частей:

ключа (уникального идентификатора) и значения (данных, привязанных к этому ключу).

Такой подход обеспечивает максимально быструю запись и чтение, особенно при огромных объёмах и высокой нагрузке. Благодаря своей простоте Key-Value базы легко масштабируются и часто используются в распределённых системах и real-time сценариях.

Они не поддерживают сложных связей, запросов или аналитики — их задача проста: “сохрани и отдай по ключу”.

Графовые СУБД

Название	Описание
Neo4j	Самая известная графовая СУБД, с визуальным интерфейсом и удобным языком Cypher.
OrientDB	Комбинирует графовую и документную модели, подходит для гибридных задач.
Amazon Neptune	Облачное графовое хранилище, оптимизировано под масштабирование.
ArangoDB	Мульти-модельная СУБД: поддерживает графы, документы и key-value.
Microsoft Azure Cosmos DB	Поддерживает графовые запросы через Gremlin.

Графовые базы данных — это тип СУБД, в которых информация хранится не в виде таблиц, а в виде узлов и связей между ними.

Узел (node) — это объект (например, человек, компания, продукт),

Связь (edge) — это то, что их соединяет (например, “друг”, “партнёр”, “купил”).

Такая модель позволяет очень быстро анализировать сложные сети и отношения, что невозможно или крайне затратно на реляционных СУБД.

Графовые базы не просто хранят данные, а делают сами связи частью данных — со своими типами, весами, направлениями и контекстом.

Векторные СУБД

Название	Описание
Qdrant	Российская высокопроизводительная векторная СУБД с REST API.
Milvus	Open-source хранилище с поддержкой масштабирования и гибкой индексации.
Pinecone	Облачная векторная база, рассчитанная на поиск по большим коллекциям.
Weaviate	Векторное хранилище с возможностью расширения семантического поиска.
Faiss	Библиотека от Meta для локального векторного поиска, применяется в СУБД.

Векторные базы данных предназначены для хранения и поиска по векторным представлениям данных — числовым наборам, описывающим объекты в многомерном пространстве.

Каждый вектор — это, по сути, цифровой “отпечаток” объекта, будь то текст, изображение, аудиофрагмент или профиль пользователя. Такие отпечатки можно сравнивать между собой: чем ближе вектора — тем более схожи объекты.

Главная задача векторных СУБД — быстрый поиск по схожести, а не по точному совпадению.

Это позволяет находить близкие по смыслу записи, даже если формально они не совпадают.

In-Memory СУБД

Название	Описание
Redis	Лидер в области In-Memory. Часто используется как кэш, брокер сообщений и счётчик.
Memcached	Очень лёгкая система для кэширования запросов и данных.
SAP HANA	Корпоративная платформа, ориентированная на аналитику в памяти.
VoltDB	Расчётная СУБД с поддержкой транзакций и аналитики в режиме реального времени.
Tarantool	Российская In-Memory СУБД с возможностью скриптов и расширений.

In-Memory базы данных — это системы, которые хранят данные не на жёстком диске, а в оперативной памяти (RAM).

Благодаря этому операции чтения и записи происходят в десятки и сотни раз быстрее, чем в классических СУБД, работающих с диском.

Такие базы особенно полезны там, где важны минимальные задержки — например, в высоконагруженных веб-системах, финансовых приложениях, игровых и телеком-сервисах.

Некоторые In-Memory СУБД поддерживают долговременное хранение на диске “на всякий случай”, но основная работа — всегда “в памяти”.

NewSQL

Название	Описание
CockroachDB	Распределённая SQL-база с высокой отказоустойчивостью.
Google Spanner	Масштабируемая SQL-база от Google с глобальной синхронизацией.
TiDB	Китайская NewSQL-база, сочетающая транзакции и аналитику.
YDB	Российская распределённая СУБД с поддержкой SQL-подобного языка YQL.
VoltDB	Вариант in-memory NewSQL, рассчитанный на real-time транзакции.

NewSQL — это новое поколение систем управления базами данных, которые сохраняют реляционную модель и язык SQL, но при этом решают те задачи, с которыми классические СУБД не справляются: масштабирование, отказоустойчивость, обработка больших объёмов.

NewSQL совмещают надежность и предсказуемость реляционной архитектуры с гибкостью и производительностью, присущей NoSQL. Эти СУБД ориентированы на нагруженные системы, которым требуются транзакции и аналитика в одном решении.

Data Warehouse (DWH)

Что это такое

Централизованная система для сбора, хранения и анализа данных из всех внутренних систем компании:

- **Единая версия правды** для всей организации
- **Очищенные и согласованные** данные
- **Историчность** - аналитика по периодам
- **Интеграция с BI** - визуализация и отчеты

«DWH — это фундамент для зрелого подхода к аналитике»

Аналогия

Представьте сеть магазинов с разными кассами и форматами отчетов. DWH — это центральный офис, где:

- Отчеты приведены к единому виду
- Данные проверены и синхронизированы
- Руководители получают информацию из единого источника

Преимущества

✓ Централизация данных
✓ Чистота и согласованность
✓ Историчность
✓ Прозрачность и доверие

Ограничения

✗ Инерционность
✗ Задержка обновления
✗ Сложность запуска
✗ Ограниченная гибкость

Когда использовать

- Стандартизация аналитики и отчетности
- Объединение данных из разных систем
- Важны согласованные KPI и контроль изменений
- Приоритет — надежность и структурированность

Data Lake

Что это такое

Архитектура для массового накопления данных в сыром виде, без предварительной структуризации:

- **Хранение без потерь** - данные сохраняются "как есть"
- **Гибкость** - любые типы данных
- **Масштабируемость** - легко увеличивается под растущие объемы
- **Низкая стоимость** - хранения (особенно в облаке)

«Сначала собери всё. Потом реши, что с этим делать»

Аналогия

Если DWH — это оформленный архив с каталогами, то Data Lake — огромный склад, куда можно складировать любые данные "про запас":

- Структурированные (таблицы)
- Полуструктурированные (JSON, XML)
- Неструктурированные (изображения, видео, тексты)

Преимущества

✓ Гибкость форматов
✓ Масштабируемость
✓ Низкая стоимость
✓ Хранение без потерь

Ограничения

✗ Нет "единой версии правды"
✗ Сложность доступа
✗ Требует инженерной поддержки

Когда использовать

- Быстрый рост объема данных
- Разнотипные и неструктурированные данные
- Сохранение всех "цифровых следов"
- Сбор данных для будущего анализа

Data Lakehouse

Что это такое

Гибридная архитектура, объединяющая лучшие качества Data Warehouse и Data Lake:

- **Гибкость Data Lake** - хранение любых типов данных
- **Структура DWH** - метаданные, схемы, ACID-транзакции
- **Производительность** - оптимизация для аналитики
- **Версионность** - отслеживание изменений данных

«Lakehouse — это попытка получить лучшее из обоих миров»

Аналогия

Представьте современный загородный дом (Lakehouse):

- Просторный и вместительный как склад (Data Lake)
- Но с четкой планировкой и организацией (Data Warehouse)
- Можно хранить что угодно, но при этом легко найти нужное

Преимущества

✓ Единая платформа для всех данных
✓ Снижение дублирования
✓ Упрощение инфраструктуры
✓ Поддержка ML и BI

Ограничения

✗ Относительно новая концепция
✗ Сложность миграции
✗ Требует специалистов

Примеры платформ

Databricks
Создатели концепции

Delta Lake
Открытый формат

Apache Iceberg
Табличный формат

Apache Hudi
Инкрементальные обновления

Data Mesh

Что это такое

Децентрализованный подход к управлению данными, основанный на доменной организации:

- **Домены** - данные принадлежат бизнес-подразделениям
- **Данные как продукт** - с SLA, документацией, поддержкой
- **Самообслуживание** - стандартизированные интерфейсы
- **Федеративное управление** - общие стандарты и политики

«Data Mesh — это организационный подход, а не технология»

Аналогия

Представьте рынок вместо централизованного супермаркета:

- Каждый продавец (домен) отвечает за свои товары (данные)
- Единые правила торговли (стандарты)
- Покупатели могут напрямую взаимодействовать с продавцами
- Каждый продавец заинтересован в качестве своего товара

Преимущества

✓ Масштабируемость организации
✓ Ответственность за качество
✓ Снижение бюрократии
✓ Гибкость и автономность

Ограничения

✗ Требует культурных изменений
✗ Сложность внедрения
✗ Риск несогласованности
✗ Дублирование усилий

Когда использовать

- Крупные организации с множеством доменов
- Проблемы с централизованным подходом
- Потребность в гибкости и автономии команд
- Готовность к организационным изменениям

Data Fabric

Что это такое

Интегрированная архитектура данных с единым слоем управления и доступа:

- **Единая сеть** - связывает все источники данных
- **Автоматизация** - интеграция, каталогизация, управление
- **Обнаружение** - поиск данных по всей организации
- **Безопасность** - единые политики доступа

«Data Fabric — это "умная ткань", связывающая все данные организации»

Аналогия

Представьте "умный город" с единой системой управления:

- Все районы (источники данных) связаны единой инфраструктурой
- Центральная система знает, где что находится
- Автоматизированная маршрутизация и доставка
- Единая система безопасности и контроля доступа

Преимущества

✓ Единый доступ к данным
✓ Снижение дублирования
✓ Автоматизация процессов
✓ Целостное управление

Ограничения

✗ Сложность внедрения
✗ Высокая стоимость
✗ Требует зрелости процессов
✗ Зависимость от вендора

Когда использовать

- Множество разрозненных источников данных
- Гибридная и мульти-облачная среда
- Потребность в едином управлении
- Высокие требования к безопасности

Сравнение архитектур хранения данных

Характеристика	Data Warehouse	Data Lake	Data Lakehouse	Data Mesh	Data Fabric
Структура данных	Структурированные	Любые	Любые	Любые	Любые
Схема данных	Schema-on-write	Schema-on-read	Гибридная	По доменам	Метаданные
Организация	Централизованная	Централизованная	Централизованная	Децентрализованная	Распределенная
Стоимость	Высокая	Низкая	Средняя	Средняя	Высокая
Качество данных	Высокое	Низкое	Среднее	Зависит от домена	Высокое
Гибкость	Низкая	Высокая	Высокая	Высокая	Средняя
Скорость внедрения	Медленная	Быстрая	Средняя	Медленная	Медленная

Тенденции развития

- Движение к гибридным решениям (Lakehouse)
- Усиление роли метаданных и каталогов
- Автоматизация управления данными
- Интеграция с ИИ и ML-инструментами

Выбор архитектуры

Зависит от:

- Зрелости организации
- Типов данных и задач
- Бюджета и ресурсов
- Организационной структуры

Импортозамещение СУБД

Преимущества российских решений

- Соответствие требованиям регуляторов
- Локальная техническая поддержка
- Независимость от санкций
- Учет российской специфики

Вызовы импортозамещения

- Функциональные ограничения
- Переобучение персонала
- Миграция существующих данных
- Интеграция с другими системами

SQL (Реляционные СУБД)

Название	Происхождение	Разработчик	Сертификация
Postgres Pro	Форк PostgreSQL	Postgres Professional	ФСТЭК, ФСБ
Jatoba	Форк PostgreSQL	Газинформсервис	ФСТЭК
Tantor	Форк PostgreSQL	ТАНТОР Лабс	ФСТЭК
Proxima DB	Форк PostgreSQL	OrionSoft	ФСТЭК
Ред База Данных	Форк Firebird	РЕД СОФТ	ФСТЭК

SQL (Реляционные СУБД)

Название	Происхождение	Разработчик	Сертификация
Pangolin DB	Собственная разработка	СберТех	—
«Квант- Гибрид»	Форк PostgreSQL	КВАНТОМ	ФСТЭК
Arenadata Postgres	Форк PostgreSQL	Arenadata	—
SoQoL	Собственная разработка	РЕЛЭКС	ФСТЭК
Linter	Собственная разработка	НИП ИВК	ФСТЭК, ФСБ

NoSQL и NewSQL

Название	Тип СУБД	Происхождение	Разработчик	Сертификация
ClickHouse	Колоночные СУБД	Собственная разработка	ClickHouse Inc. / Yandex	—
Qdrant	Векторные СУБД	Собственная разработка	Qdrant Team (Россия)	—
Tarantool	In-Memory СУБД	Собственная разработка	VK (ex-Mail.ru Group)	—
YDB	NewSQL	Собственная разработка	VK (ex-Mail.ru Group)	—

Векторные базы данных и ИИ

Что такое векторизация данных

Преобразование любых данных в числовые векторы:

- **Текст**
- **Изображение**
- **Аудио**
- **Пользователь**

«Векторы — это универсальный язык для ИИ, позволяющий сравнивать несравнимое»

Принцип работы

Семантически близкие объекты располагаются рядом в многомерном пространстве. Например, векторы слов "автомобиль" и "машина" будут ближе друг к другу, чем к вектору слова "яблоко".

RAG-архитектура

Retrieval Augmented Generation — современный подход к ИИ:

1. Запрос пользователя преобразуется в вектор
2. Векторная БД находит похожие векторы (документы)
3. Найденные документы передаются в LLM-модель
4. Модель генерирует ответ на основе контекста

Преимущества

✓ Актуальность информации
✓ Снижение галлюцинаций
✓ Прозрачность источников
✓ Работа с корпоративными данными

Применения

- Корпоративные чат-боты
- Поиск по документации
- Персонализированные рекомендации
- Анализ клиентского опыта

Кейсы

Вопросы


ВЫСШАЯ • ШКОЛА • ЭКОНОМИКИ •