Modern Storages and Data Warehousing
Week 1 - Intro

Попов Илья, i.popov@hse.ru

Давайте знакомиться

- Попов Илья
- Закончил ПМИ ФКН в 2021
- Закончил ФТиАД в 2023
- Работал 4 года в OZON, из них 2 - руководил аналитикой маркетплейса
- С осени 2022 года руковожу разработкой аналитической инфраструктуры в Яндекс.Беспилотниках
- Email: i.popov@hse.ru / contact@popov.tech
- Telegram: @mgcrp

Организационные моменты

Важная информация:

〉 GitHub курса: https://github.com/mgcrp/hse_dwh_course_2023

〉 Telegram курса: https://t.me/+n-2dgcnojw4yOTli

〉 02.09-21.10 (с перерывом 16.09-23.09) - по субботам

〉 07.11 - 19.12 - по вторникам

Курс состоит из:

〉 13 занятий - лекции и семинары

〉 4 домашних задания

GitHub

Telegram

Содержание курса

01 | Введение в Data Engineering

02 | Файловые хранилища. S3 / HDFS

03 | Hadoop-экосистема

04 | Apache Spark

05 | Data Warehousing. Data Vault / Anchor Modeling

06 | Очереди и потоки данных: Apache Kafka / Spark Streaming

07 | MPP. ClickHouse / Vertica / Greenplum

08 | Транспорты данных: CDC / Debezium

09 | Планы запросов. Отличия запросов к реляционным БД / MPP / Spark

10 | ETL / ELT. Правила проектирования раздел

11 | BI-инструменты: Metabase / Superset / Grafana

12 | Инфраструктура данных в облаках

+ Новые вызовы DE: Data Governance, Data Quality, Data Lineage, Process Lineage

Формула оценки

Формула оценки:

$О_{итог} = 0.25\ О_{ДЗ\ 1} + 0.25\ О_{ДЗ\ 2} + 0.25\ О_{ДЗ\ 3} + 0.25\ О_{ДЗ\ 4}$

Примерное содержание ДЗ:

〉 ДЗ №1

Настроить репликацию данных между хостами PostgreSQL в Docker

Создать автоматически партицируемую таблицу в PostgreSQL, написать запросы для нескольких сценариев использования

〉 ДЗ №2:

Настроить трансфер из PostgreSQL в MPP / HDFS (по выбору) в Docker

Разложить БД на якорную модель / Data Vault (в зависимости от выбранного хранилища), написать брокеры для пополнения DDS

〉 ДЗ №3

Собрать из полученных данных в DWH несколько витрин с помощью dbt

Реализовать регламентное обновление витрин с помощью AirFlow в Docker

〉 ДЗ №4:

Добавить в полученную Docker-compose инсталляцию BI-инструмент по выбору (Metabase, Superset, Grafana)

Построить по полученной витрине данных дэшборд

P.S.: Каждое следующее задание, как вы видите, связано с предыдущим. Приступить к новому, не сделав прошлое, не получится.

Что будет по итогу

〉 После курса вы будете понимать, чем разные хранилища данных отличаются друг от друга и как они работают

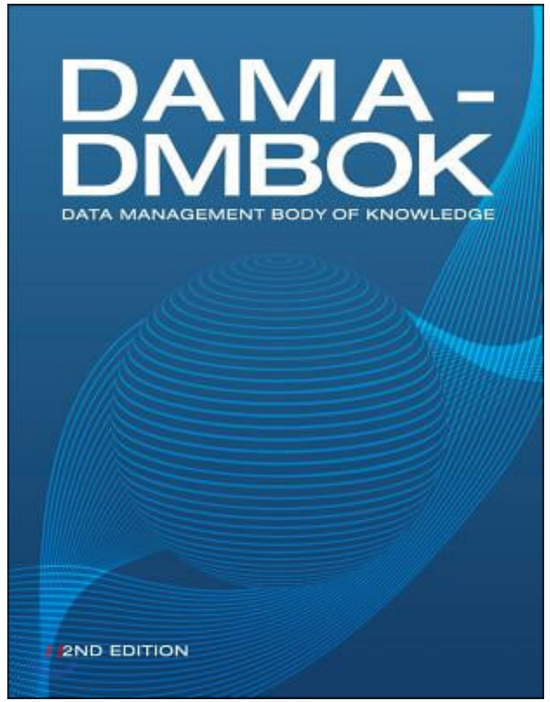
〉 Поймете, как выбрать хранилище под задачи вашего проекта

〉 Сможете разговорить с дата-инженерами на одном языке, будете лучше понимать, как ставить им задачи

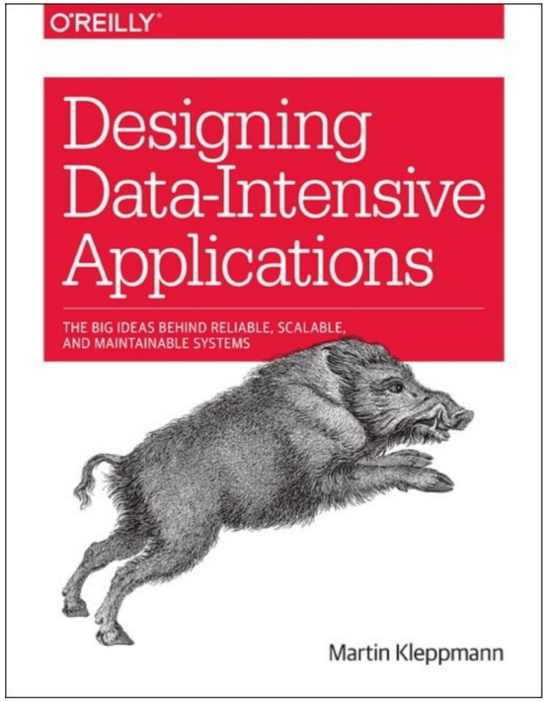
〉 Научитесь писать эффективные запросы к разным системам обработки данных

〉 *Бонус: курс поможет с прохождением интервью на все Data-позиции*

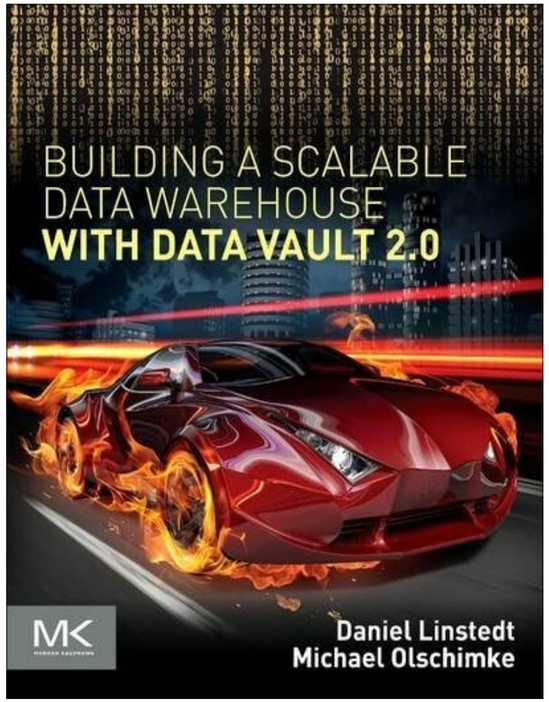
Что почитать на досуге



DAMA DMBOK (Data Management Body of Knowledge)



Martin Kleppmann - Designing Data-Intensive Applications



Dan Linstedt - Building a Scalable Data Warehouse with Data Vault 2.0

Если хочется еще контента

- ШАД - Data Warehousing
- ШАД - Алгоритмы во внешней памяти
- ШАД - Алгоритмы для работы с большими объемами данных
- ШАД - Базы данных
- Clickhouse Academy - Clickhouse Training - https://learn.clickhouse.com/
- Vertica Academy - Vertica Essentials - https://academy.vertica.com/course/essentials10x
- Data Engineering Zoom Camp - https://dezoomcamp.streamlit.app

Введение в Data Engineering

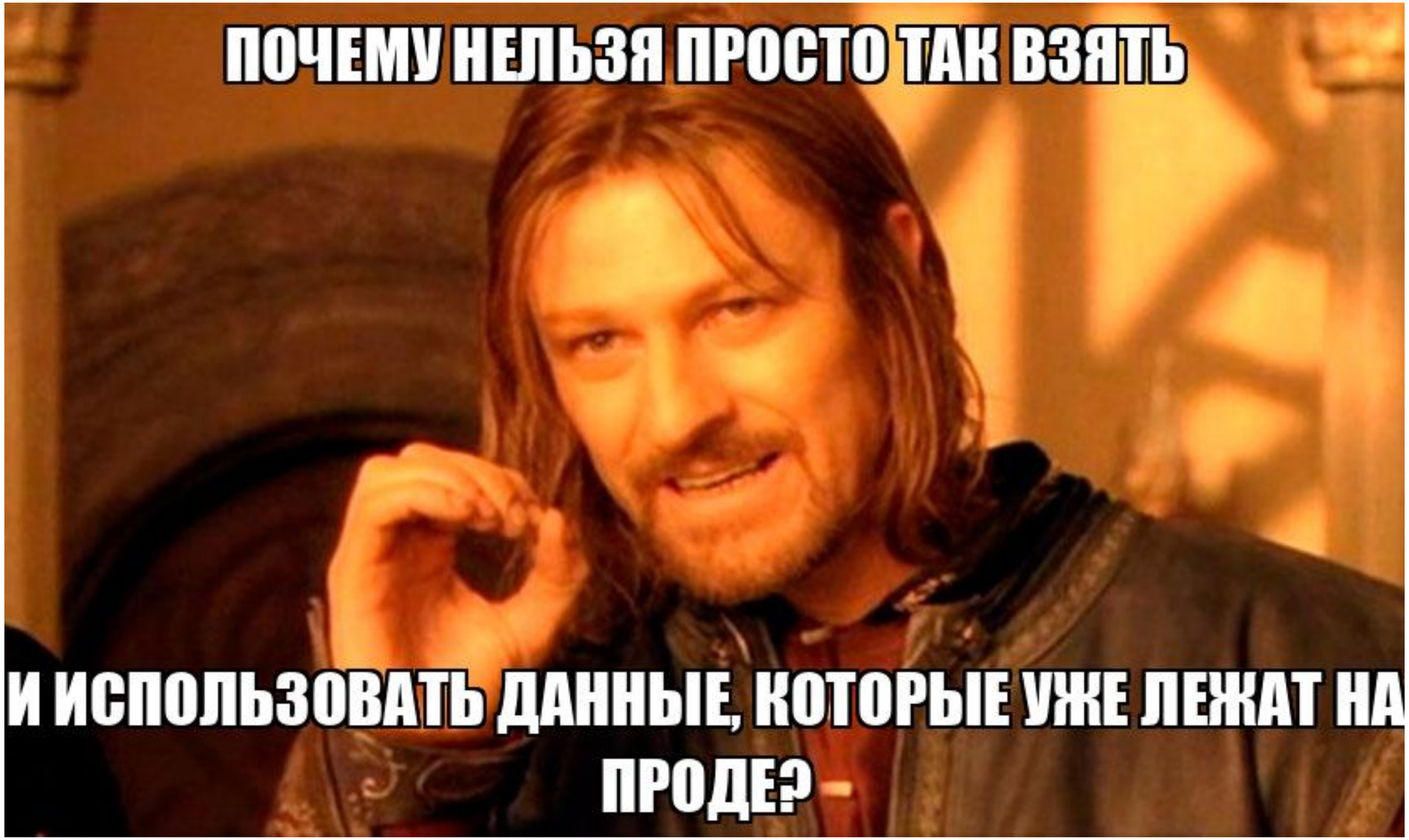
Зачем компании обрабатывать данные?

Система поддержки принятия решений (СППР) (англ. Decision Support System, DSS) — компьютерная автоматизированная система, целью которой является помощь людям, принимающим решение в сложных условиях для полного и объективного анализа предметной деятельности.

Чем занимается СППР:

- Ввод данных
- Хранение данных
- Анализ данных

Но…


ПОЧЕМУ НЕЛЬЗЯ ПРОСТО ТАК ВЗЯТЬ
И ИСПОЛЬЗОВАТЬ ДАННЫЕ, КОТОРЫЕ УЖЕ ЛЕЖАТ НА ПРОДЕ?

Почему нельзя просто использовать данные, которые уже где-то лежат?

Определимся с базовыми понятиями

Аннотация:
Современный бизнес нуждается в введении data-driven культуры, что подразумевает принятие решений на основе регулярно обновляющихся данных. В случае гостиничного бизнеса оптимальным решением является разработка автоматизированного пайплайна, который бы хранил, обрабатывал и анализировал регулярно обновляющиеся данные. Для такой задачи потребовались инструменты анализа временных рядов (модели ARIMA, ARIMAX), инструменты для работы с большим количеством признаков и декомпозированием (Prophet, Random Forest, LightGBM), а также разработка всей backend- части (БД PostgreSQL и СУБД DBeaver) и её автоматизация (Cron). Также было произведено обучение модели чувствительности спроса к ценовым параметрам среднего тарифа и объема траффика на сайте компании (IRF), и была проведена кросс-валидация на различных временных периодах.
В итоге выбраны модель LightGBM для применения в разрезе каждой пары отеля и типа номеров, которые в совокупности показали наилучшую точность прогнозов по данным кросс-валидации. Эти модели автоматически размещаются в настроенные БД, обновляясь ежедневно, и имеют вид, готовый к использованию бизнес единицами.

👆🏻Такие вещи на курсе будут сразу караться минус баллом

〉 **СУБД** - Система Управления Базами Данных - непосредственно движок, который осуществляет работу БД. Пример: PostgreSQL, MySQL, MS SQL Server, Oracle SQL и т.д.

〉 **БД** - непосредственно База Данных. Структура из таблиц и данных. Пример: база данных сервиса, база данных интернет магазина и т.д.

〉 **IDE** - интерфейс для доступа к данным и администрированию БД. Пример: Data Grip, DBeaver, Microsoft SQL Server Studio и т.д.

OLTP-хранилища

〉 **OLTP (Online Transaction Processing)** - система оперативной обработки информации (транзакций). Заточены под быструю обработку INSERT, UPDATE, DELETE операций.

〉 **Транзакция** - некоторый набор операций над базой данных, который рассматривается как единое целое. Транзакцией может являться несколько операций, но они обязательно выполняются все вместе, часть операций отдельно выполниться не может.

〉 **Главное требование к OLTP-системам** - быстрое обслуживание относительно простых запросов большого числа пользователей, при этом время выполнение запроса не должно превышать (микро-, милли-)секунд.

OLAP-хранилища

- **OLAP (Online Analytical Processing)** имеет дело с историческими или архивными данными. OLAP характеризуется относительно низким объемом отдельных запросов.
- **Главное требование к OLAP-системам** - переваривать тяжелые аналитические запросы с большим объемом данных. SLA на OLAP-системы менее строгий - от нескольких минут до нескольких часов.

Холодные хранилища

〉 **Холодное хранилище** - самое медленное из трех. Задача такого хранилища - дешево хранить большой объем данных, доступ к котором осуществляется редко. Пример такого хранилища - s3; Основная идея схожа с идеей OLAP, но здесь еще менее строгие требования к SLA на обработку данных.

〉 **Главное требование к холодным хранилищам** - максимально дешево хранить большой объем данных.

Сравнение хранилищ

Характеристика	OLTP	OLAP	Холодное
Степень детализации	Хранение детализированных данных	Хранение детализированных и обобщенных (агрегированных) данных	Хранение произвольных данных
Формат хранения	Зависит от архитектуры сервиса	Единый, согласованный, структурированный	Пережатый
Допущение избыточности	Максимальная нормализация, сложные структуры	Контролируемая избыточность	Избыточность не контролируется

Сравнение хранилищ

Характеристика	OLTP	OLAP	Холодное
Управление данными	Добавление \ удаление \ изменение в любое время	Периодическое добавление данных	Периодическая архивация данных
Количество хранимых данных	Все данные, необходимые для работы сервиса	Все данные, необходимые для аналитики	Все архивные данные
Характер запросов к данным	Регламентные запросы с бэкенда	Произвольные аналитические запросы	Произвольные запросы над архивными данными

Вывод - OLAP / OLTP / холодные хранилища не взаимозаменяемы, нужно понимать различия систем друг от друга и уметь пользоваться всеми.

Что мы строим? Классическое определение хранилища данных.

> **Хранилище данных** — предметно-ориентированный, интегрированный, неизменчивый, поддерживающий хронологию набор данных, организованный для целей поддержки принятия решений.

〉 **Предметная ориентированность** — для принятия решений требуется некоторая строго определенная совокупность данных, которая и поступает из БД в хранилище данных, второстепенные ненужные атрибуты отсеиваются.

〉 **Интегрированность** — устранение несоответствий внутри данных.

〉 **Временная привязка** — оперативные системы охватывают небольшой интервал времени, что достигается за счет периодического архивирования данных.

〉 **Неизменчивость** — модификация данных не производится, поскольку может привести к нарушению их целостности.

Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало

Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало
〉 1992 - Bill Inmon “Building the Data Warehouse”
〉 1996 - Ralph Kimball “The Data Warehouse Toolkit”

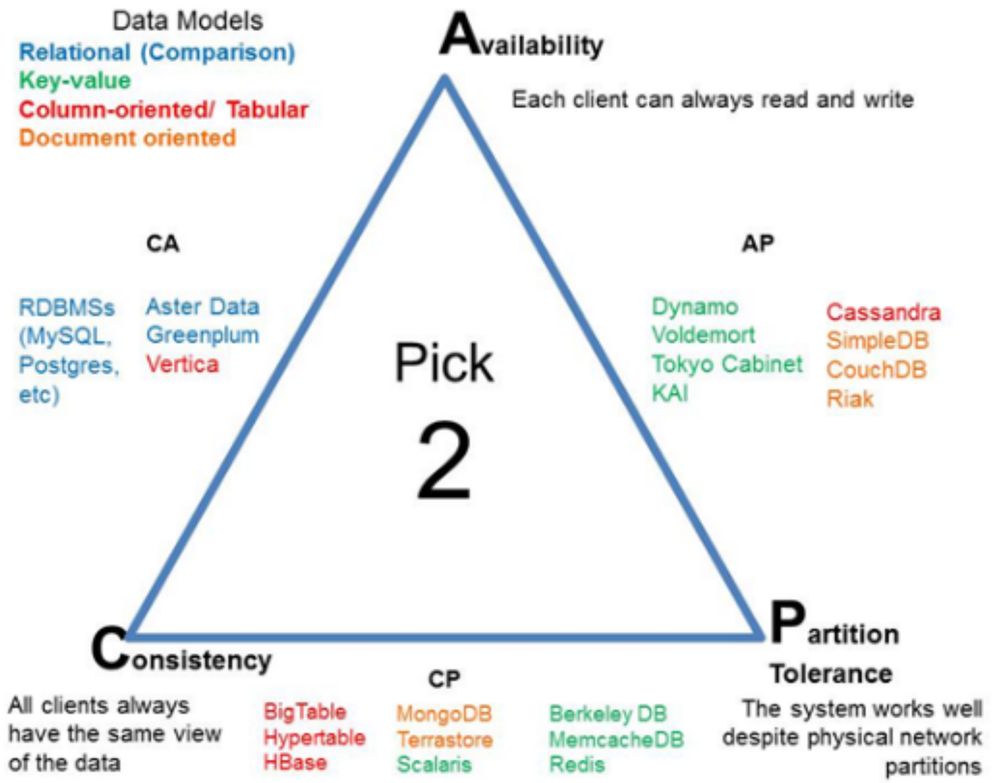
Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало
〉 1992 - Bill Inmon “Building the Data Warehouse”
〉 1996 - Ralph Kimball “The Data Warehouse Toolkit”
〉 1998 - NoSQL

Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало
〉 1992 - Bill Inmon “Building the Data Warehouse”
〉 1996 - Ralph Kimball “The Data Warehouse Toolkit”
〉 1998 - NoSQL
〉 2002 - CAP теорема

CAP-теорема

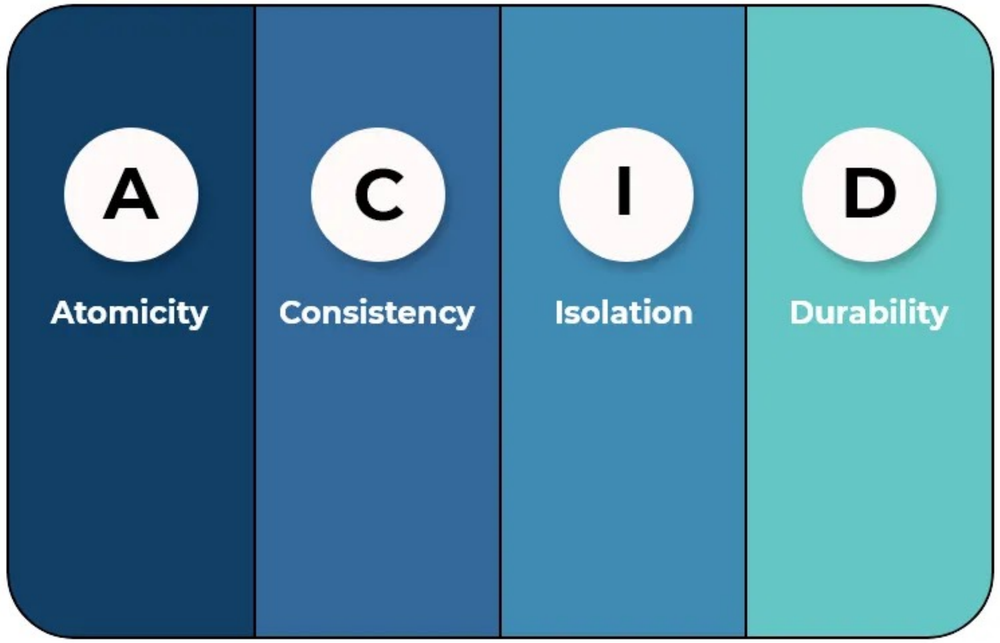


- C - **Consistency** - согласованность. Каждое чтение дает последнюю запись;
- A - **Availability** - доступность. Каждый не упавший узел всегда успешно выполняет запросы;
- P - **Partition Tolerance** - устойчивость к распределению. Если между узлами нет связи, они продолжают работать независимо друг от друга.
- Можно выбрать только 2 из 3;

Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало
〉 1992 - Bill Inmon “Building the Data Warehouse”
〉 1996 - Ralph Kimball “The Data Warehouse Toolkit”
〉 1998 - NoSQL
〉 2002 - CAP теорема
〉 2007 - Michael Stonebraker ”The end of an architectural era: it's time for a complete rewrite”
〉 2008 - NewSQL. Масштабируемость от NoSQL + ACID от реляционных СУБД

ACID-принцип



- A - Atomicity - атомарность. Транзакция обрабатывается либо целиком, либо никак.
- C - Consistency - согласованность. Транзакция не нарушает согласованности данных после своего исполнения.
- I - Isolation - изолированность. Параллельные транзакции не влияют на работу друг друга.
- D - Durability - стойкость. Сбой системы не должен влиять на успешно завершенные транзакции.

Лирическое отступление

〉 Первые БД в современном понимании появились в 1970-х
〉 До начала 1990-х реляционные СУБД плавно развивались, возможностей их архитектуры хватало
〉 1992 - Bill Inmon “Building the Data Warehouse”
〉 1996 - Ralph Kimball “The Data Warehouse Toolkit”
〉 1998 - NoSQL
〉 2002 - CAP теорема
〉 2007 - Michael Stonebraker ”The end of an architectural era: it's time for a complete rewrite”
〉 2010-е - Big Data

Big Data

Определение VVV:

〉 **Volume** (объем) - объем хранилища в смысле HDD

〉 **Velocity** (скорость) - скорость прироста данных в хранилище и скорость обработки данных

〉 **Variety** (разнообразность) - одновременная обработка разных типов структурированной и полу-структурированной информации

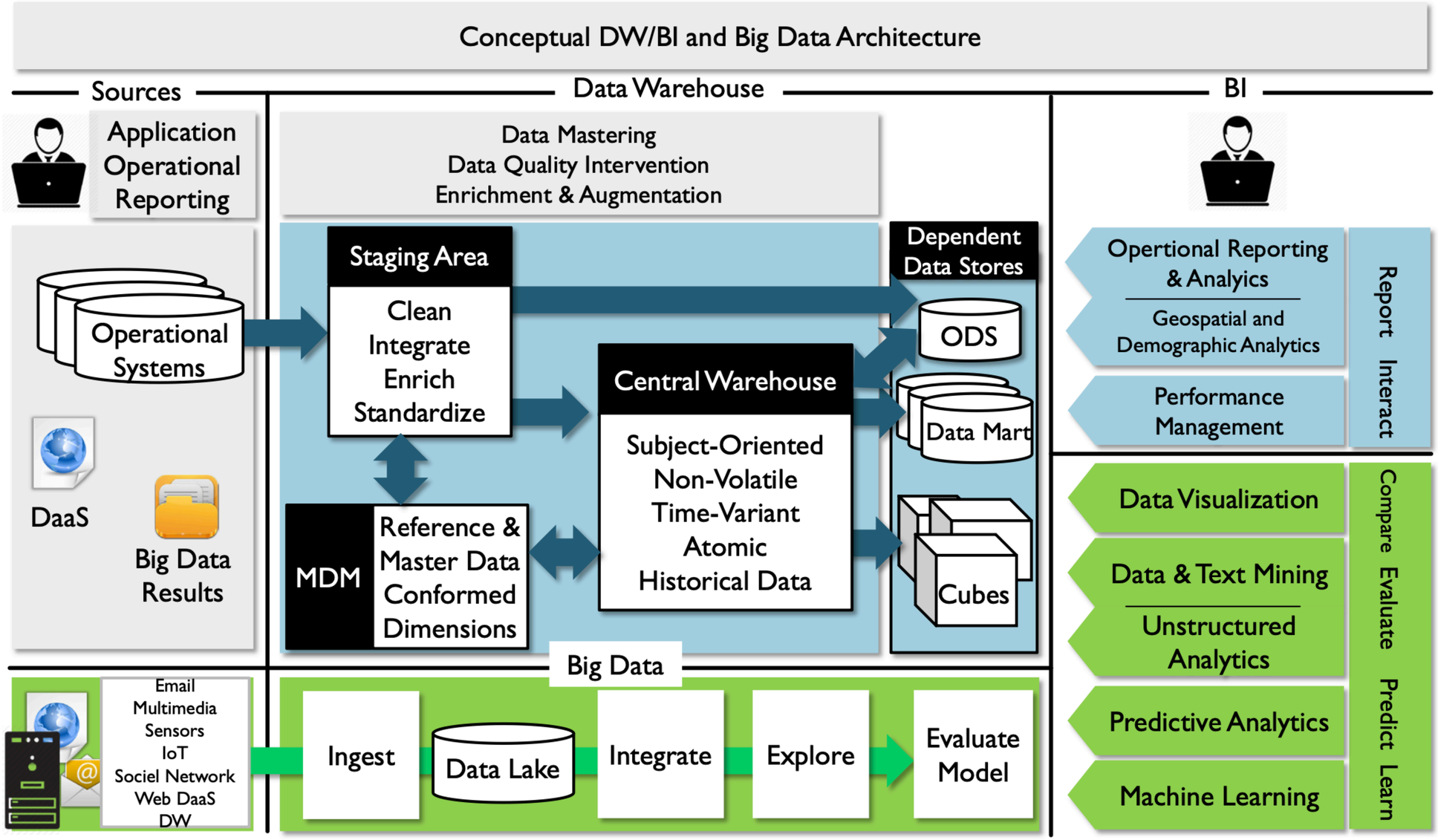
Определение +VV:

〉 **Veracity** (доверенность) - достоверность, целостность и согласованность данных, возможность доверять результатам вычислений

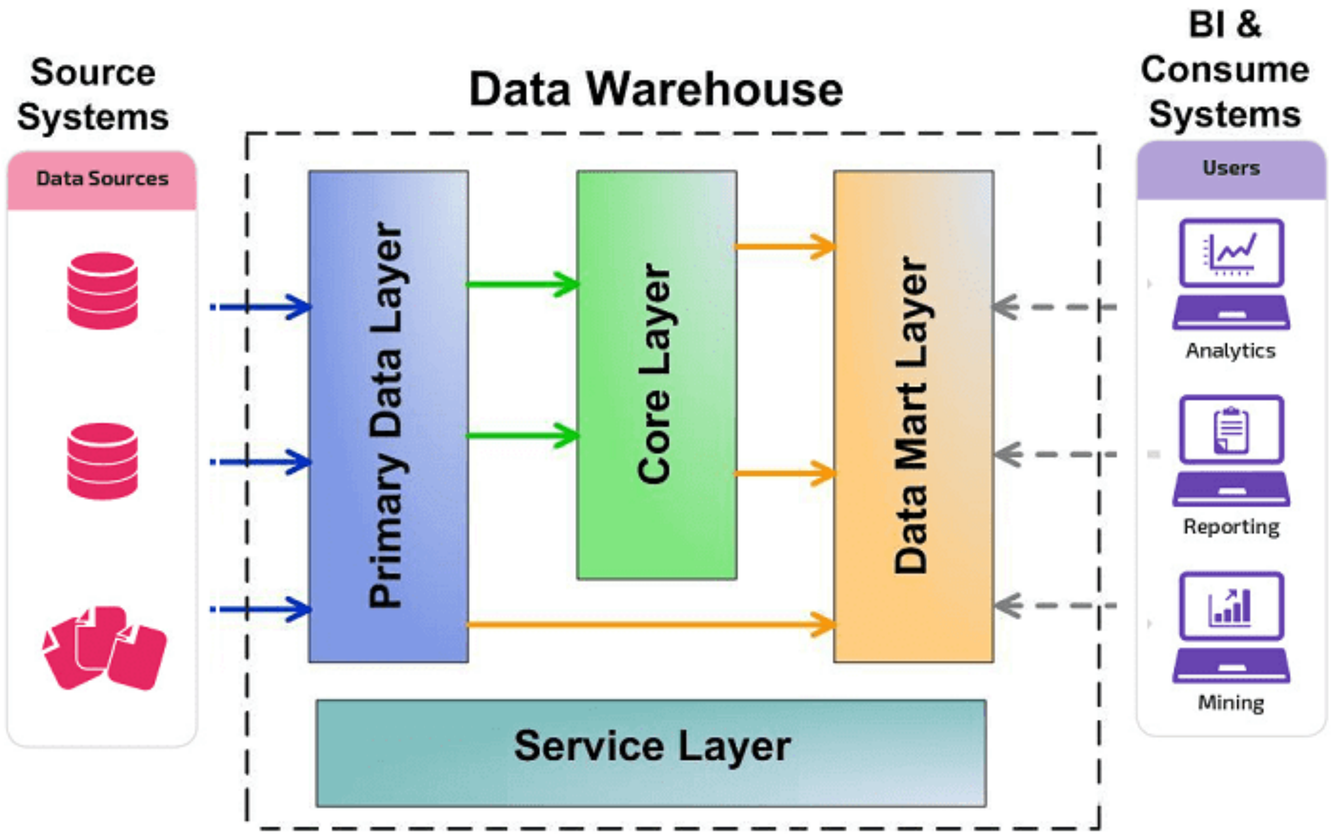
〉 **Value** (ценность) - ценность, экономическая целесообразность хранения и обработки такого объема данных

Устройство корпоративного хранилища данных

Хранилище компании



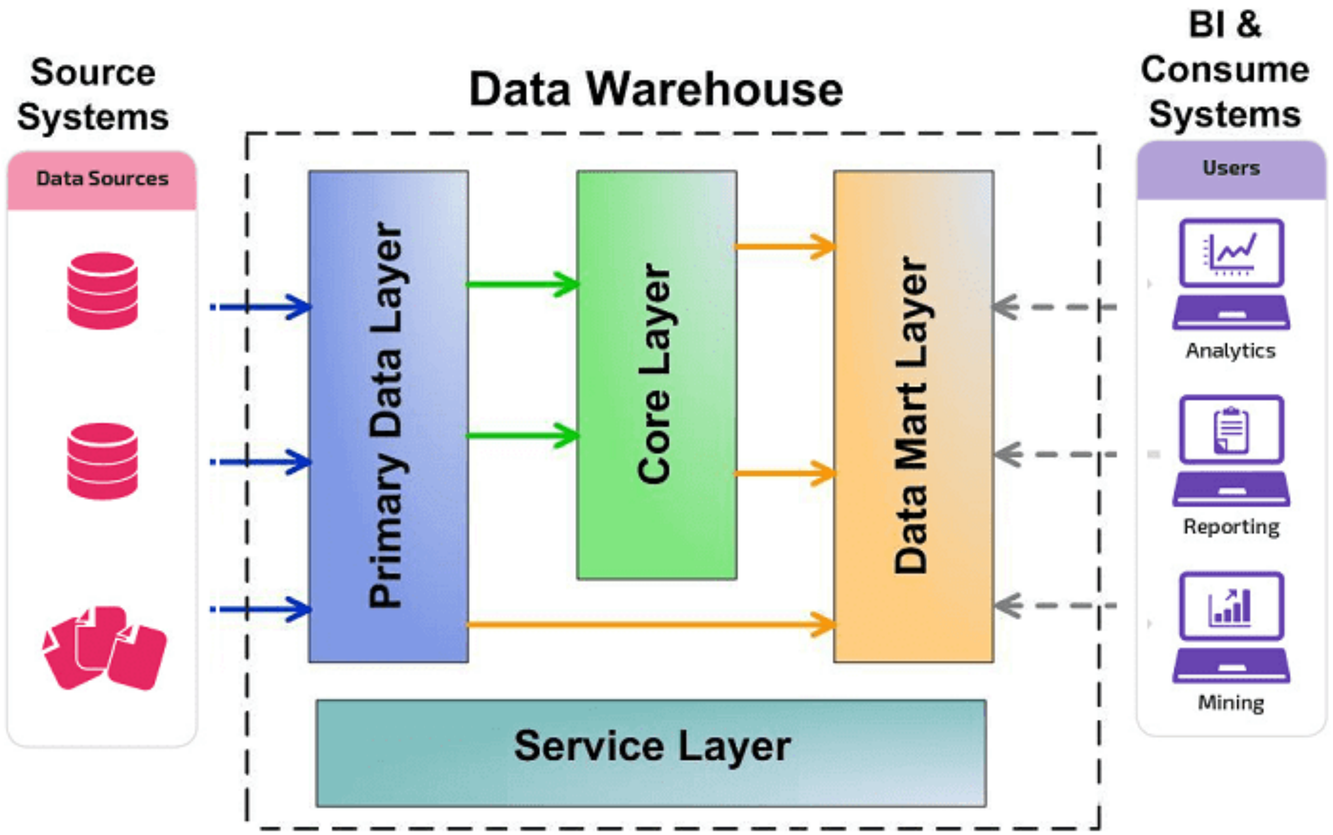
Теперь говорим про DWH



Уровневая архитектура DWH - средство борьбы со сложностью системы.

Каждый последующий уровень абстрагирован от сложностей внутренней реализации предыдущих слоёв.

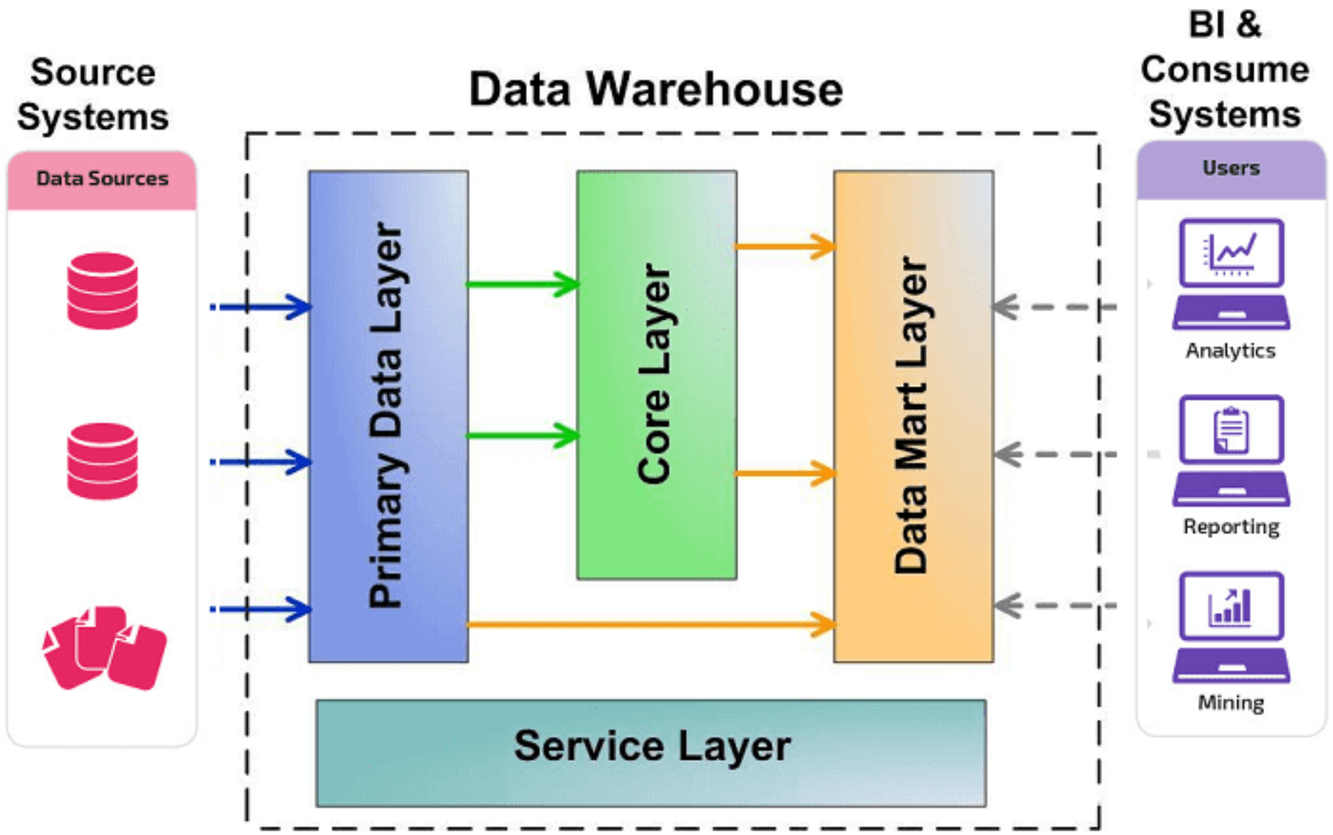
Теперь говорим про DWH



〉 **Primary Data Layer**, он же **Raw**-слой, он же **Staging** - привозим данные в наш OLAP, чтобы с ними было удобнее работать;

На этом слое происходит абстрагирование следующих слоев хранилища от физического устройства источников данных, способов их сбора и методов выделения изменений.

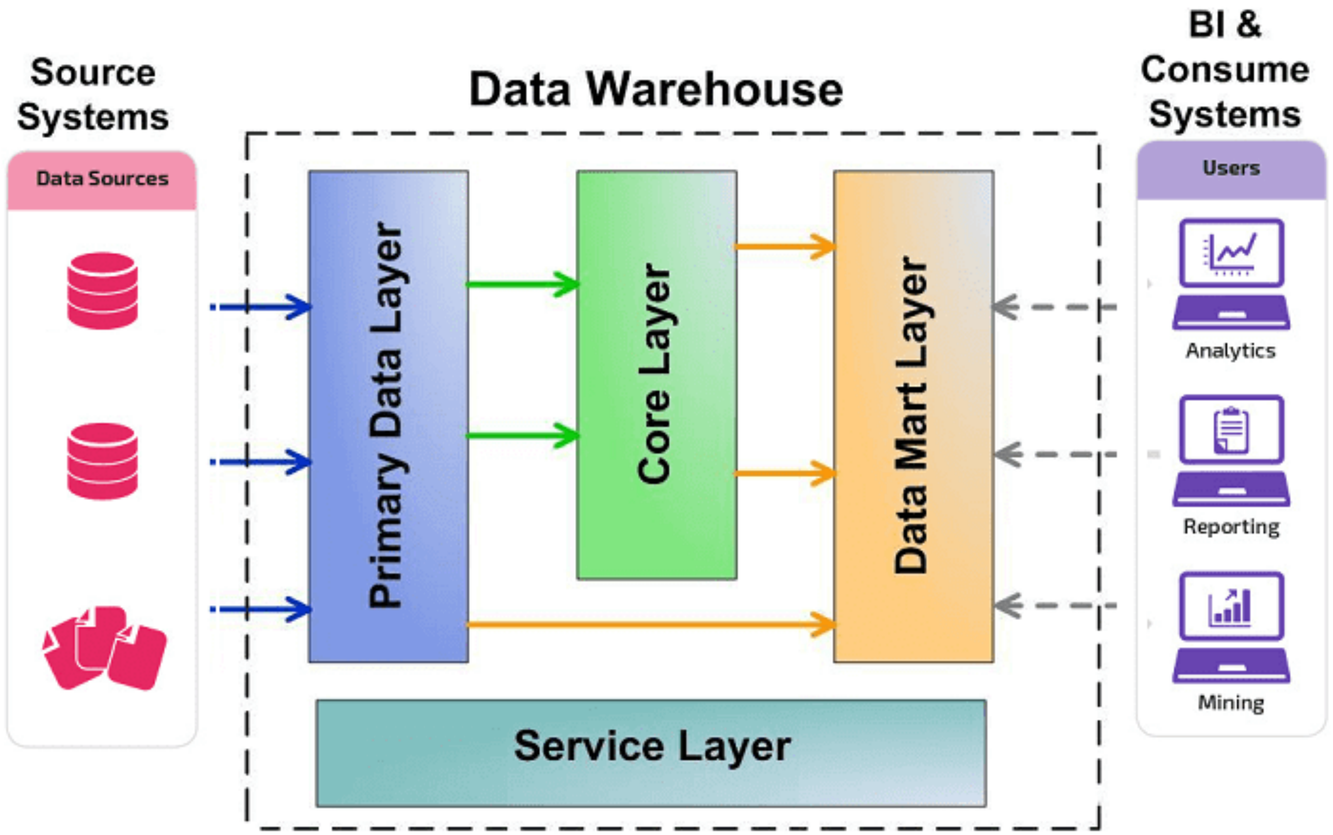
Теперь говорим про DWH



〉 **Core Data Layer**, он же **Detailed Data Store** - центральный слой, в котором происходит консолидация данных из разных источников, приводя их к единым структурам и ключам.

Здесь происходит основная работа с качеством данных и происходят трансформации, чтобы абстрагировать потребителей от особенностей логического устройства источников данных и необходимости их взаимного сопоставления.

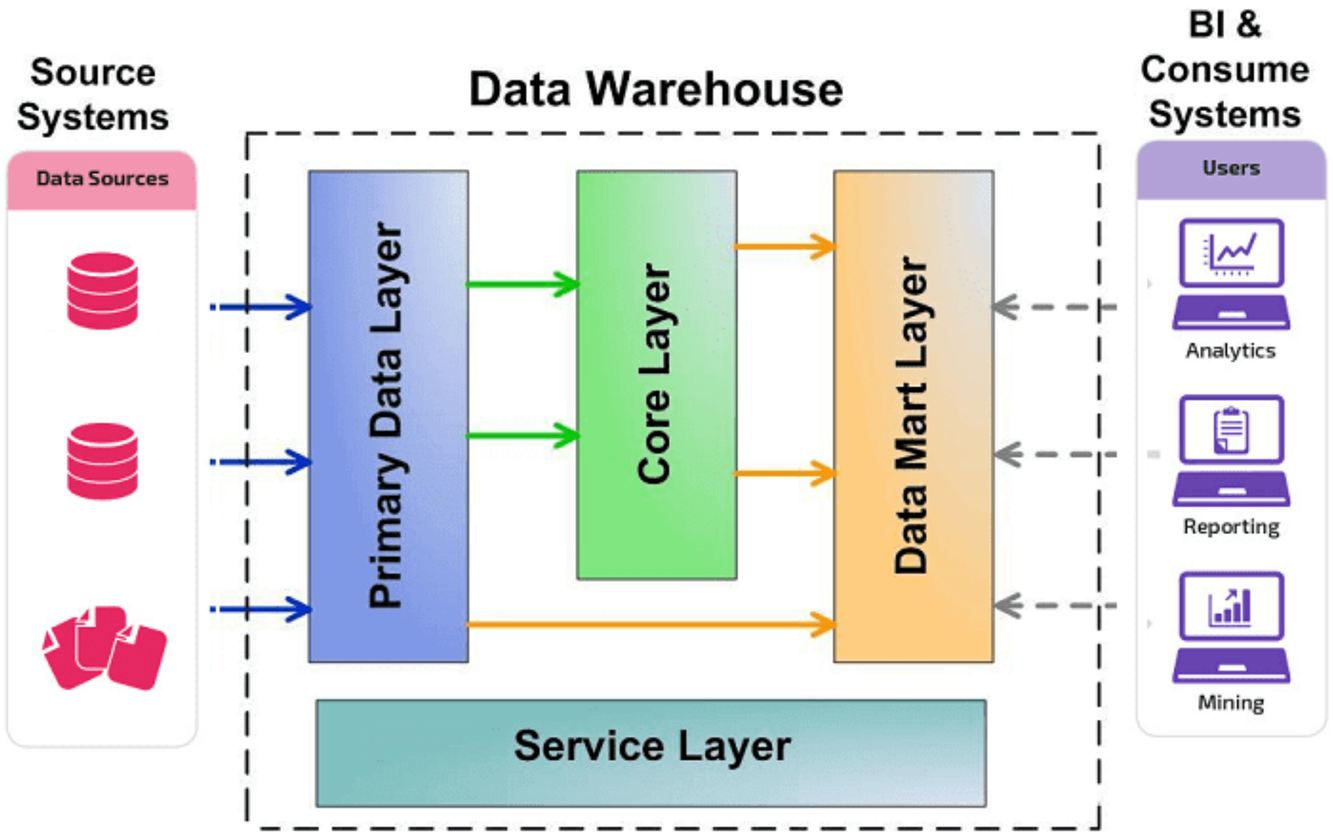
Теперь говорим про DWH



〉 **Data Mart Layer**, он же **Common Data Marts** - слой, где данные преобразуются к структурам, удобным для анализа и использования в BI или других системах-потребителях.

Этот слой может включать в себя не только общие витрины данных, собранные DE, но и специализированные, собранные DA под какую-то конкретную задачу.

Теперь говорим про DWH

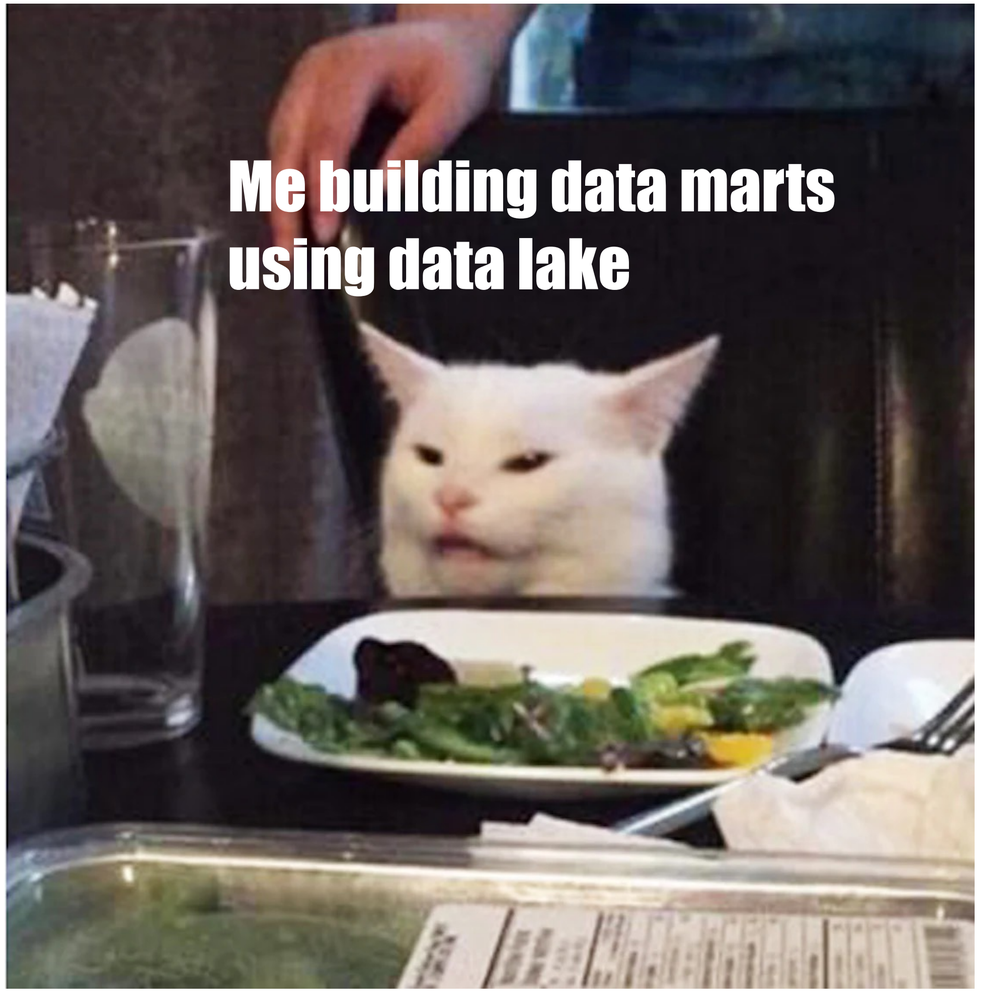


К этому часто добавляют:

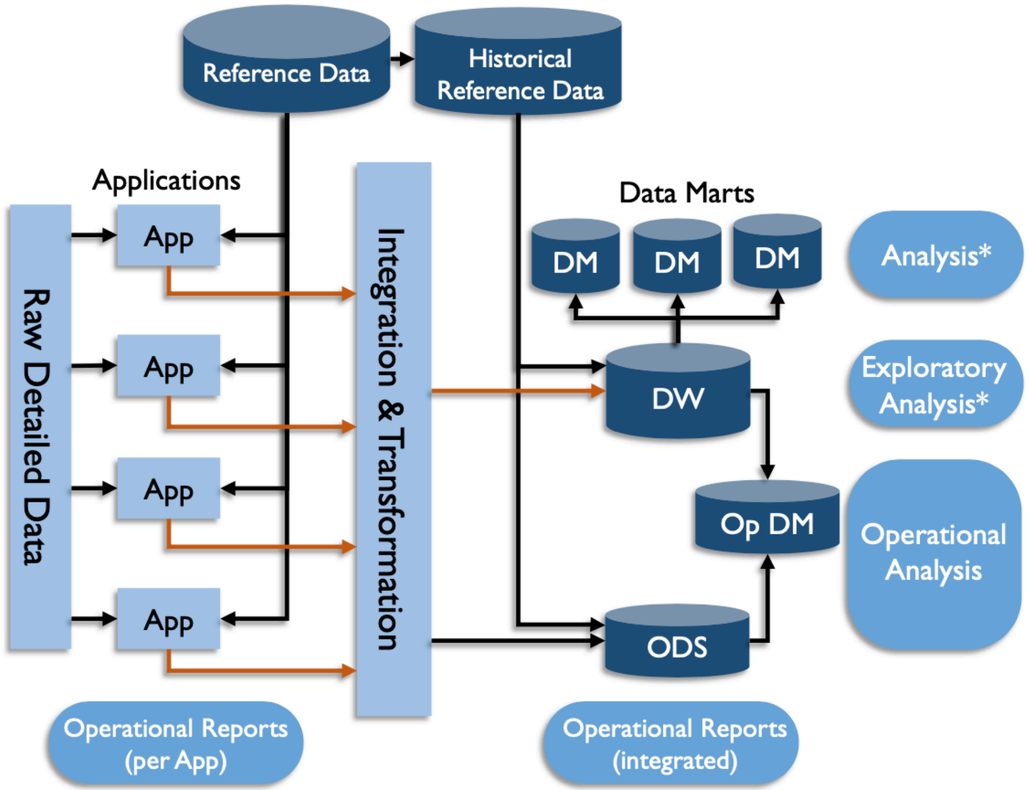
〉 **Operational Data Store** - слой, в который realtime реплицируются данные для операционной аналитики;

〉 **Presentation Layer** - специфичные витрины под отчеты/BI-инструменты.

Два подхода к проектированию DWH



DWH по Инмону



DWH по Инмону это:

〉 Проектирование ХД модели “сверху вниз”.

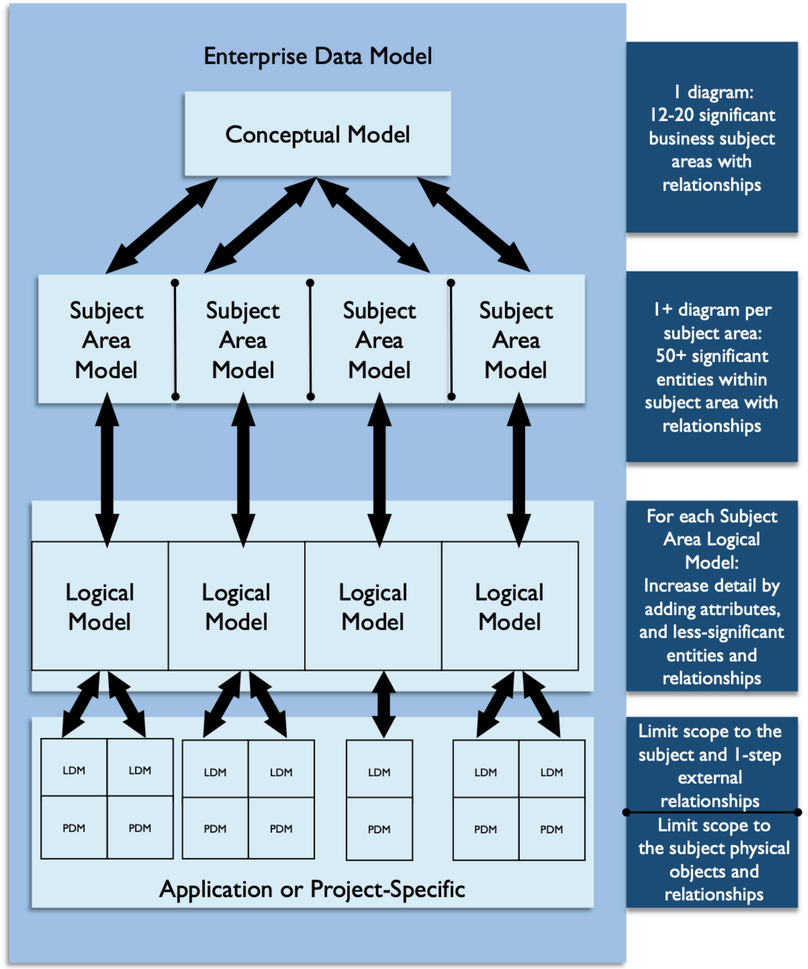
〉 Тщательный анализ бизнеса в целом. Выявляются бизнес-области, в них - ключевые бизнес-сущности, затем - их характеристики (атрибуты) и связи между ними.

〉 Строительство хранилища не сразу, а по частям

〉 Высокая степень нормализации детального слоя

Большой и целостный проект всей компании, а не механическое перекладывание JSON

DWH по Инмону



Плюсы

〉 «Единая версия правды»

〉 Отсутствие противоречивости в данных

〉 Отсутствие избыточности упрощает ETL и уменьшает вероятность коллизий в данных

〉 Детальный слой содержит проекцию бизнес-процессов

〉 Легко поддерживать при увеличении количества источников

Минусы

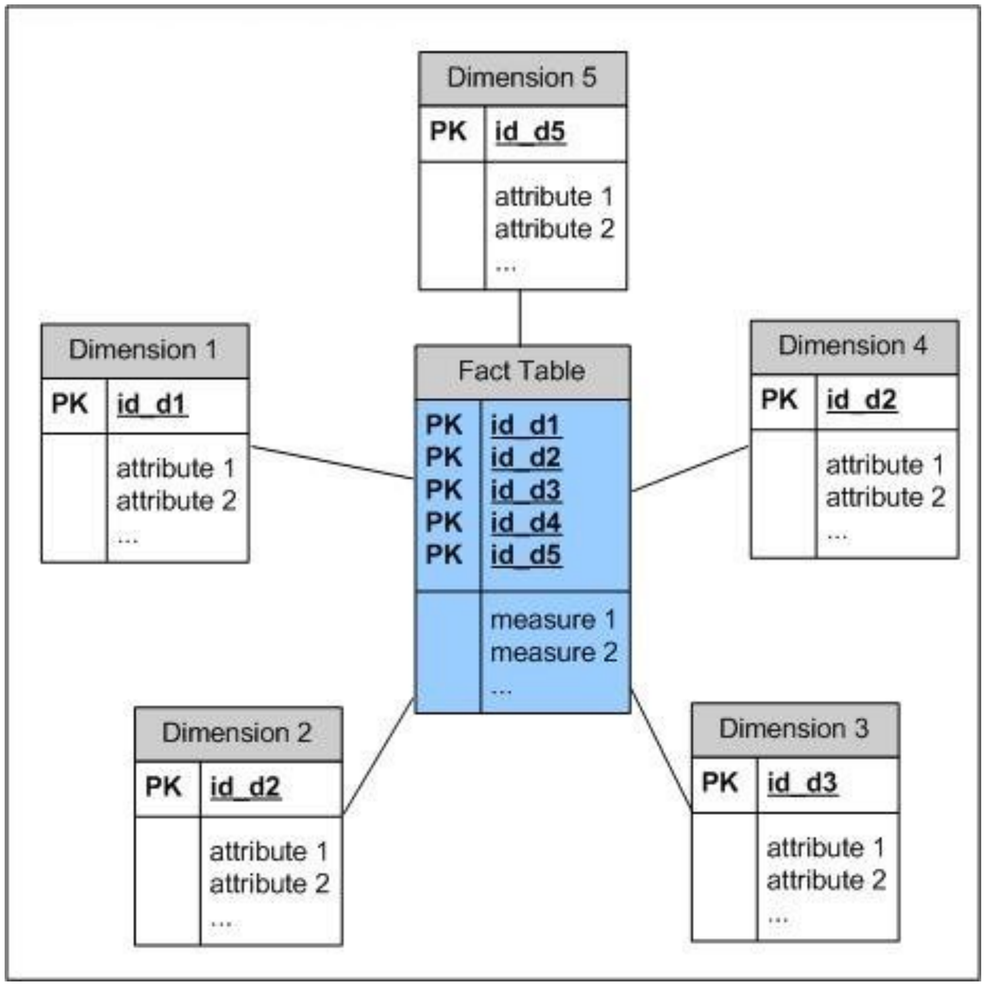
〉 Сложная в проектировании, нужна высококлассная команда

〉 Сложно для аналитиков, много джоинов

〉 Долгая в реализации на первоначальном этапе анализа бизнеса

〉 Дорого

DWH по Кимбаллу



DWH по Кимбаллу - копия транзакционных данных, специально структурированных для запроса и анализа в виде витрин данных.

Хранилище по Кимбаллу можно назвать коллекцией витрин данных (отчетов).

Проектирование снизу вверх:

〉 Анализ потребностей – узнаем, какие отчеты нужны

〉 Анализ источников – узнаем, в каких источниках есть данные

〉 Проектируем витрину под конкретного потребителя

〉 Первичные данные из источников преобразуются в витрины

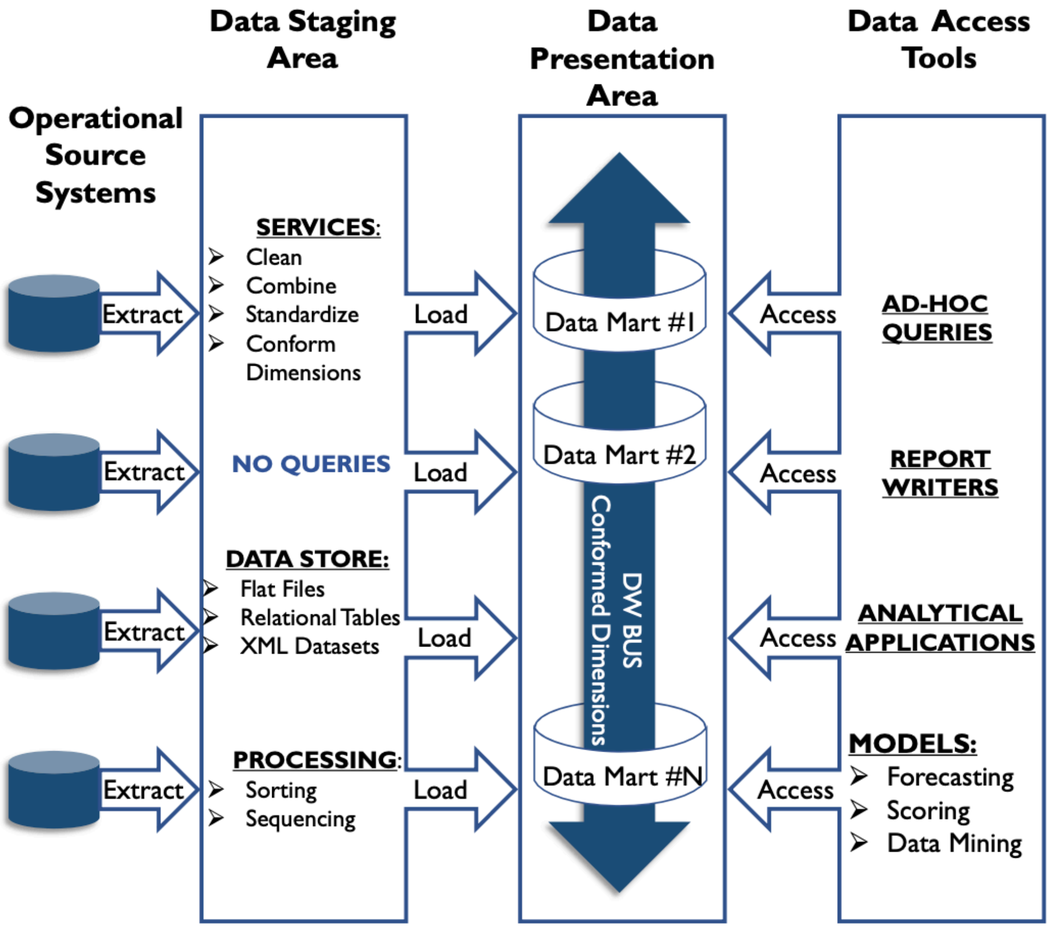
Особенности:

〉 Схема “Звезда” (факты + измерения)

〉 Нет DDS

〉 Много разных витрин, связанных между собой

DWH по Кимбаллу



Плюсы

〉 Быстрый эффект

〉 Достаточно поэтапного анализа бизнес-областей

〉 Не требует высококвалифицированных специалистов (на старте)

Минусы

〉 Высокая стоимость поддержки новых источников

〉 Отсутствие стандартизации показателей (в каждой витрине может быть свой алгоритм)

Data Lake - частный случай DWH по Кимбаллу

Озеро данных (Data Lake) - хранилище большого объема неструктурированных данных, собранных или генерированных одной компанией.

В озеро данных поступают все данные, которые собирает компания, без предварительной очистки и подготовки.

Особенности Data Lake:

〉 Хранятся все данные, в т.ч. и «бесполезные»

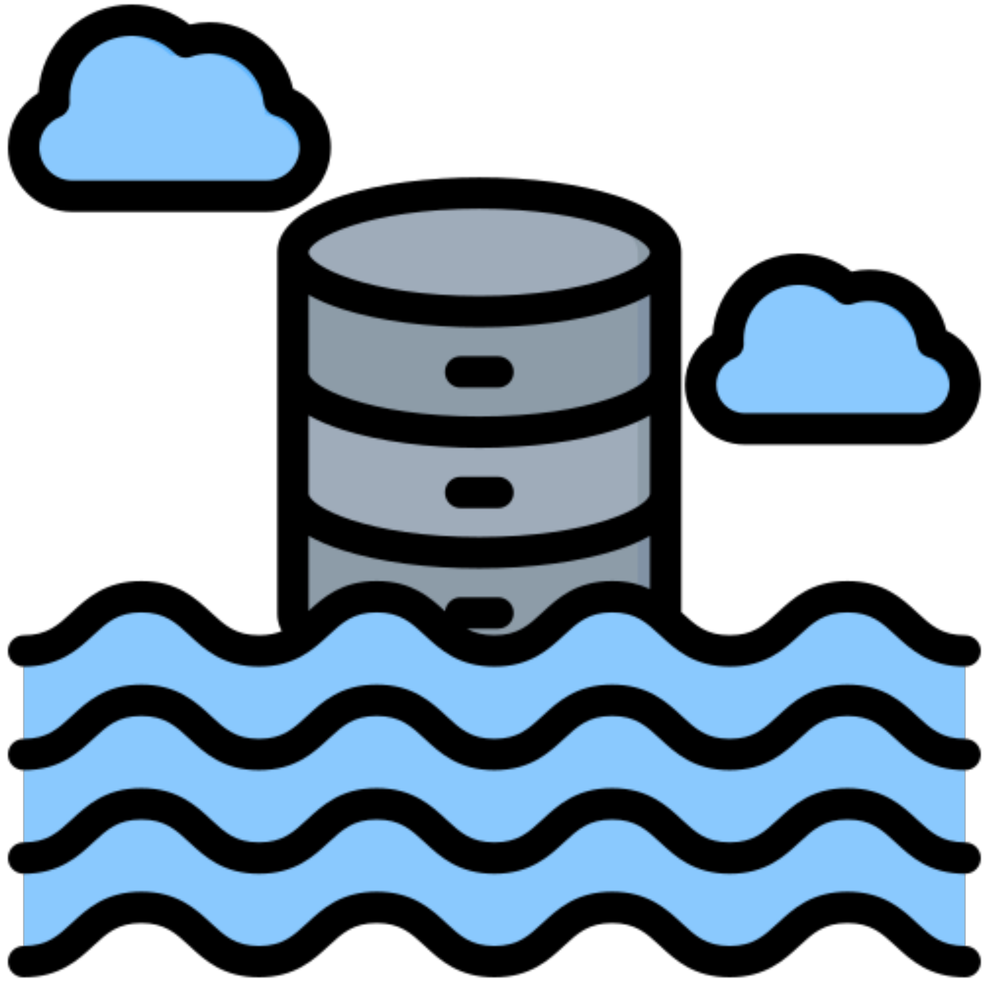
〉 Структурированные, полу-структурированные и неструктурированные разнородные данные любых форматов

〉 Высокая гибкость, которая позволяет в процессе эксплуатации добавлять новые типы и структуры данных

〉 Необходима дополнительная обработка данных для их практического использования из-за отсутствия четкой структуры

〉 Дешевле DWH с точки зрения проектирования

Data Lake - плюсы



- **Масштабируемость** – распределенная файловая система позволяет по мере необходимости подключить новые машины или узлы без изменения структуры хранилища или сложной перенастройки
- **Экономичность** – Data Lake можно построить на базе свободного ПО Apache Hadoop, без дорогостоящих лицензий и дорогих серверов, масштабируя систему под свои затраты
- **Универсальность** – большие объемы разнородных данных могут использоваться практически для любой исследовательской задачи – от прогнозирования спроса до выявления пользовательских предпочтений или влияния погоды на качество продукции
- **Быстрота запуска** – накопленные объемы Data Lake позволяют быстро проверить очередную модель, не тратя время и инженерные ресурсы на сбор информации из разных источников

Data Lake - минусы

Озеро умеет свойство превращаться в болото - **Data Swamp**

〉 Низкое качество данных ввиду отсутствия контроля при их загрузке, простоты этого процесса и дешевизны хранения;

〉 Сложность определения ценности данных:

- Big Data предполагает важность любой информации
- Но если бизнесу быстро нужны какие-то данные, об этом, как правило, известно заранее.
- такую информацию логично сразу загружать в DWH или витрину

Data Mesh

Data Mesh - модное молодежное направление в DWH.

〉 Поделим данные на домены, домены друг от друга независимы.

〉 Данные - интерфейс к сервису.

〉 Относимся к данным как к продукту.

〉 Дата инженеры сидят внутри продуктовых команд.

Очень полезная статья - https://habr.com/ru/post/495670/

Обработка данных

ETL и ELT

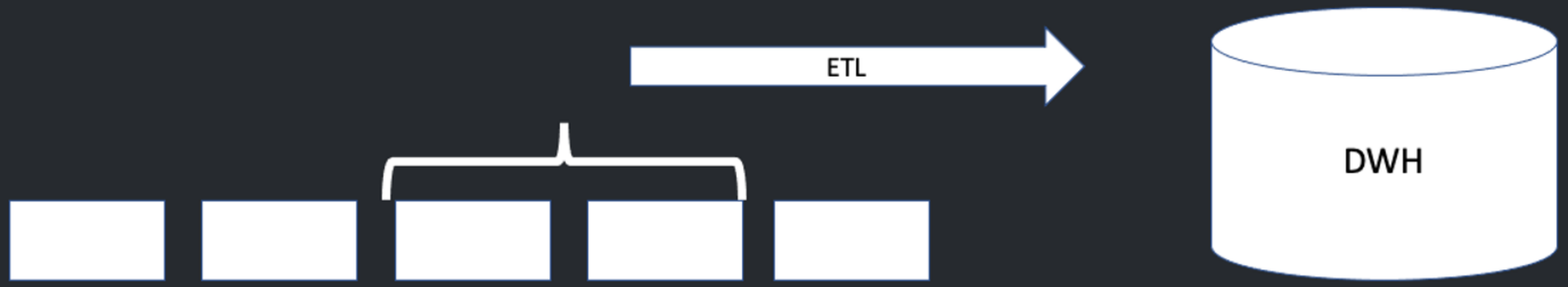
ETL - это процесс преобразования данных, который состоит из:

〉 **Извлечение данных (Extraction - E)** - из одного или нескольких источников и подготовка их к преобразованию (загрузка в промежуточную область, проверка данных на соответствие спецификациям и возможность последующей загрузки в ХД);

〉 **Трансформация данных (Transform - T)** - преобразование форматов и кодировки, агрегация и очистка;

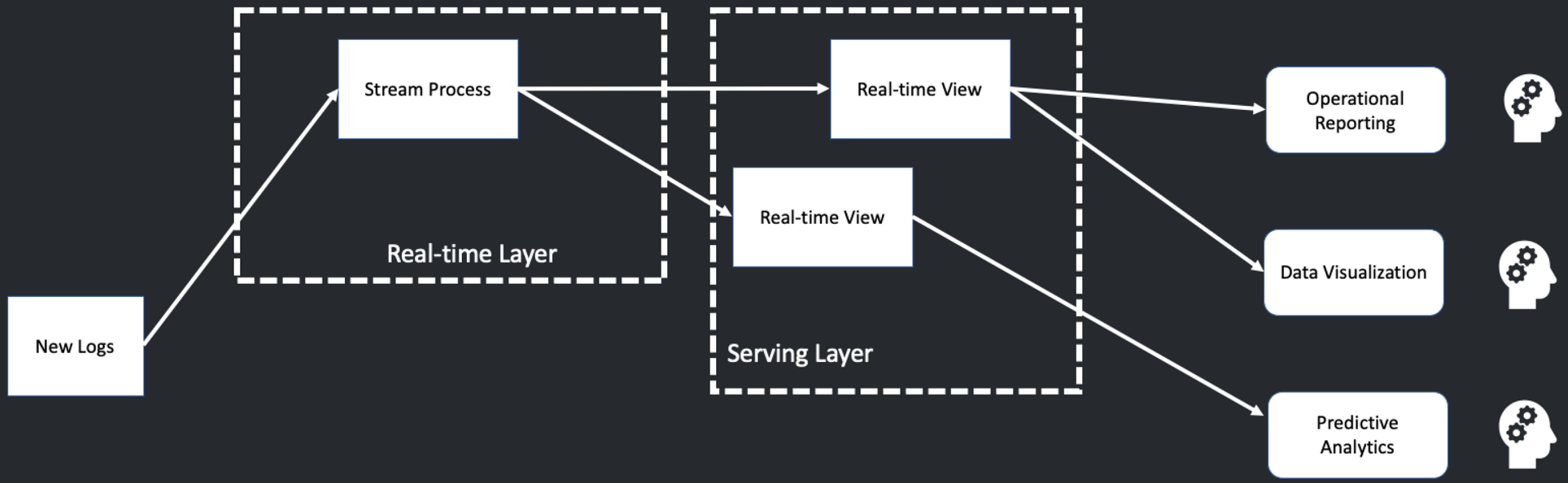
〉 **Загрузка данных (Load - L)** - запись преобразованных данных, включая информацию о структуре их представления (метаданные) в необходимую систему хранения или витрину данных.

Batching

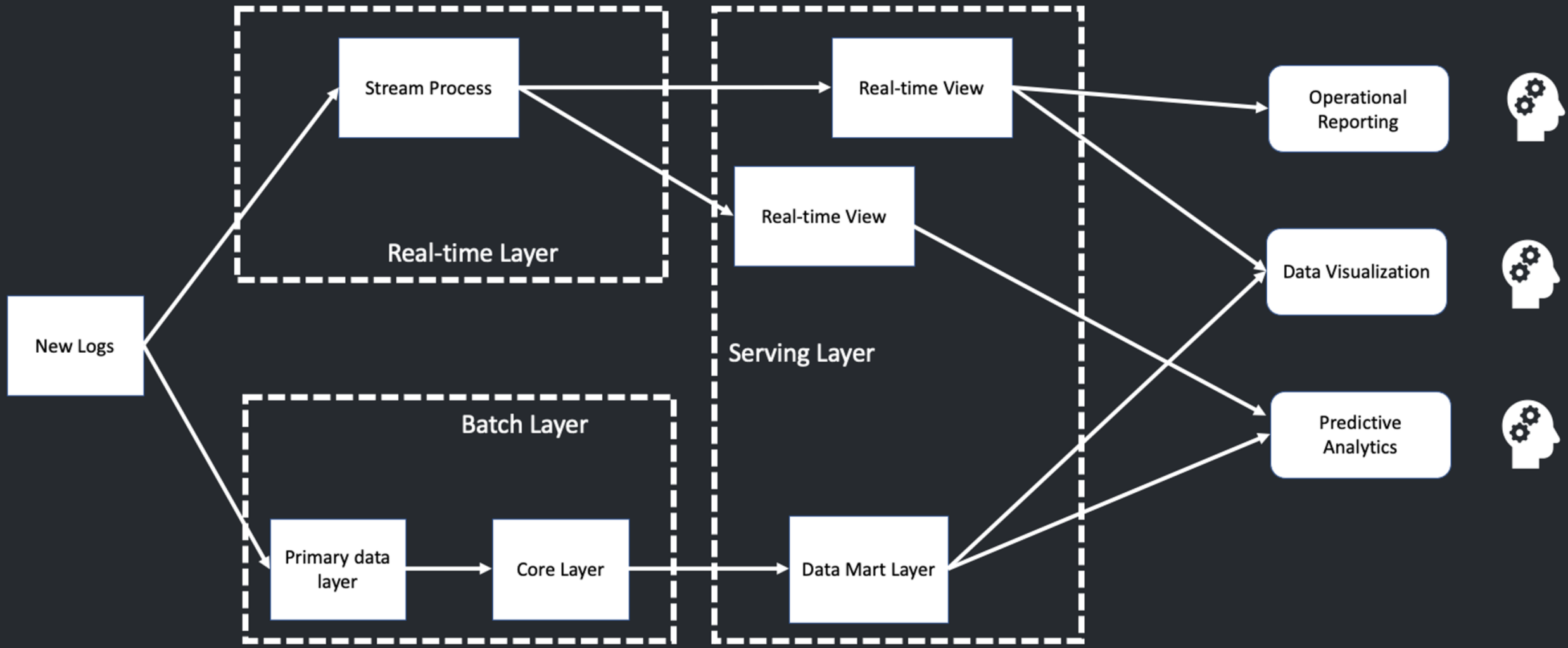


Streaming

Kappa


Stream Process
Real-time Layer
New Logs
Real-time View
Real-time View
Serving Layer
Operational Reporting
Data Visualization
Predictive Analytics

Lambda


Stream Process
Real-time Layer
New Logs
Batch Layer
Primary data layer
Core Layer
Real-time View
Real-time View
Serving Layer
Data Mart Layer
Operational Reporting
Data Visualization
Predictive Analytics

Перерыв