DATA ENGINEERING

The process of designing, building and scalling sytems that organize the data for analytics is called data engineering.

DATA WAREHOUSE:

A subject oriented

integrated

Time variant

Non-volatile

Collection of data in order to support management decision making .

- **Subject-oriented**
  Focuses on specific subjects like sales, customers, or products (not day-to-day operations).
- **Integrated**
  Collects and combines data from different sources into one consistent format.
- **Time-variant**
  Stores historical data (not just current data), so you can analyze trends over time.
- **Non-volatile**
  Once data is added, it is not changed or deleted. It's stable and used for analysis, not for frequent updates.

Need for Decision Support System (DSS) in Business

- In today's fast and competitive business world, decisions need to be made **quickly**.
- Managers can't take too long to think — they need tools to help them decide faster.
- So, companies use **Decision Support Systems (DSS)** — these systems provide information to help make smart and quick decisions.

How DSS Works (Architecture)

DSS has three main parts:

1. **Data Acquisition Layer**
    a. Collects raw data (from databases, documents, etc.).
2. **Analytical Engine**
    a. Uses models and expert systems to **analyze** the data and find useful insights.
3. **Graphical User Interface (GUI)**
    a. A user-friendly screen where managers can **see the results** and make decisions.

Choosing a Stock Portfolio: Structured vs Unstructured Data

- When selecting stocks, some parts are **structured**, like:
    - **Risk** (can be measured),
    - **Performance** (based on past data).
- But choosing which specific stocks to buy also involves **gut feeling** or **intuition** — this part is **unstructured** because it can't be measured easily.

Benefits of OLTP (Image 1)

1. Simplicity & Efficiency

- OLTP systems reduce paperwork and manual effort.
- For example, instead of using printed invoices and manual calculations, the system stores everything digitally and quickly.
- It helps businesses make faster and more accurate predictions about income and spending.

2. Data Integrity & Fast Query Processing

- OLTP systems keep data **accurate and consistent**, even when many users are working at the same time.
- They can **respond very quickly** when you ask for data or perform tasks—like checking a product's stock or making a sale.

Pitfalls of OLTP (Image 2)

1. Instant Updates Are Required

- OLTP systems must update information **immediately** (e.g., when a sale is made, stock levels should change instantly).
- This makes the system **more complex** to manage.

2. Not Suitable for Data Analysis

- The data collected is meant for operations, not for **analyzing trends or making business decisions**.
- For decision-making, businesses use **OLAP systems**.

3. Complex Queries Require Joins

- Even for simple tasks, you might need to get data from **multiple linked tables** using complex queries (called **joins**).
- This can make things a bit harder for those who are not technical.

Operational Data — *"What's happening now?"*

This is **raw, live data** directly generated by the business as it runs.

- 🔸 **Source:** Systems that run daily operations — like POS for sales, ERP for inventory, CRM for customer info
- 🔸 **Content:** Every tiny action—sales made, carts updated, items restocked, logins, transactions
- 🔸 **Nature:** Extremely **granular**, **fast-changing**, and tied to specific timestamps (e.g. "Order #5127 placed at 4:03 PM")
- 🔸 **Storage:** Usually stored in **relational tables** with lots of rows and keys

🍼 Example for your backend: A `sales_orders` table with fields like `order_id`, `product_id`, `timestamp`, `quantity` — used for order processing

📈 Business Data — *"What do the numbers mean?"*

This is **processed and analyzed** data that turns operations into **insights**.

- 🔸 **Source:** Comes from cleaning, aggregating, and transforming operational data
- 🔸 **Content:** KPIs, summaries, averages, forecasts, derived metrics
- 🔸 **Nature: Summarized** and **optimized** for reporting and dashboards — no noise, just meaning
- 🔸 **Storage:** Often lives in **data marts**, OLAP cubes, or dashboards

🧪 Example for your backend: A table showing **monthly revenue per product**, or **top-selling items last quarter**

🌍 External Data — *"What's going on outside?"*

This is any data not created by your company but **useful to enrich decision-making**.

- 🔸 **Source:** Public APIs (weather, news), paid data feeds (market reports), government sites, social media, etc.
- 🔸 **Content:** Anything from inflation rates, competitor prices, population stats, to trending hashtags
- 🔸 **Nature: Contextual**—not tied to transactions, but gives outside perspective
- 🔸 **Storage:** Often integrated via ETL pipelines or API fetches into staging areas in the warehouse

🧪 Example for your backend: Using **weather data** to see how rainy seasons affect online orders for umbrellas in Chennai 🌧️🛒

Where Business Data Comes From

Your **data store** contains two main categories:

1. Business Data

This is the actual information used by your company for analysis, decisions, and planning.

- ✅ It's extracted from:
  - **Operational databases** — like transaction logs, inventory systems, customer orders
  - **External sources** — anything outside your company that's useful, like:
    - 📈 Stock prices
    - 📊 Market indicators
    - 📣 Marketing info

        - 🦹‍♂️ Competitor data

💡 These external sources *enrich* your internal operational data by adding context, trends, and benchmarks.

2. Business Data Model

This is like the blueprint or schema that defines **how the business data is organized** inside the data store.

- 📐 It sets rules like:
    - What attributes each data entity should have
    - How tables relate to each other
    - Which constraints or keys keep everything consistent

🧩 Think of this as the **structure** your backend SQL schemas follow so that analysis tools can easily query and report on the data.

🔄 Flow Summary:

- Raw data (internal + external) goes into the data store
- It gets shaped by the **Business Data Model**
- Final result: Clean, structured data ready for insights and business intelligence tools

the **five main building blocks** of a **Decision Support System (DSS)** — which is a system used by businesses to help make smart decisions based on data.

🧱 1. Data Store Component

This is like the **warehouse** or **storage room** where all raw data sits — sales, customers, market stats, etc.

🧲 2. Data Extraction Component

Imagine tools that **pull out specific data** from the store. Like saying: “Show me last month’s sales only.”

🖌️ 3. Data Filtering Component

This cleans and **removes unnecessary data**. So you only see what matters, not everything.

🔍 4. End User Query Tool

This lets people **ask questions** like “What was our best-selling item last year?” — and the system finds answers.

📊 5. End User Presentation Tool

Finally, this part **displays the answers** in a readable format — graphs, tables, or reports you can understand quickly.

Step 1: Data Enters from Multiple Sources

- Sources like **budget, HR systems, project files, procurement records** feed data into the system.
- This data is raw and scattered — coming from different departments.

Step 2: Goes into the ODS (Operational Data Store)

- ODS acts like a **temporary storage and cleaning room**.
- Here, data is **merged, validated, and transformed** so it’s consistent and usable.
- Think of it like staging before a live show — everything gets organized backstage.

Step 3: Integrated Data Moves to the Data Warehouse

- Once cleaned and combined, the data goes to the main **data warehouse**.
- Now it's ready for reporting, analysis, and decision-making.

Step 4: Sent Back to Operational Systems

- The structured insights are sent to various roles:
  - 📊 **Executive Managers** use it for strategic decisions
  - 🛠️ **Program Managers** analyze project success
  - 📃 **Administrative Staff** work on scheduling, resources, etc.
  - 🧍 **Field Staff** and 🗣️ **Public Info teams** use insights for daily operations

🧠 Why It's Useful

- Instead of looking at **scattered, messy data**, the organization now sees **a single, refined view**.

- Decisions become **faster, smarter, and more data-driven**.

.

🧠 What Is a Data Mart?

A **data mart** is a **subset of a data warehouse** that focuses on a specific business area like sales, finance, HR, or inventory.

- 📦 Only contains data relevant to that department
- 🚀 Faster and easier to access compared to a full warehouse
- 🛠️ Often tailored to the types of questions users in that domain ask regularly

🔄 How It Fits in the Big Picture

Layer	Purpose	Example
Data Warehouse	Central, all-encompassing storage	All customer and sales data across India
Data Mart	Department-focused slice	Sales Mart for Chennai region only

🧪 Example for Backend Use

Imagine your ecommerce platform has a **Sales Data Mart**:

- It includes just `orders`, `products`, `sales_amount`, `region`, `sales_date`
- Marketing team can run queries like: “Show top-selling items in Tamil Nadu for last 3 months”

No need to dig through finance or HR data — everything in the mart is **sales-focused**!

🔄 OLAP (Online Analytical Processing)

OLAP is a technique used to **analyze complex data quickly and interactively**. It's perfect for turning huge volumes of data into meaningful business insights.

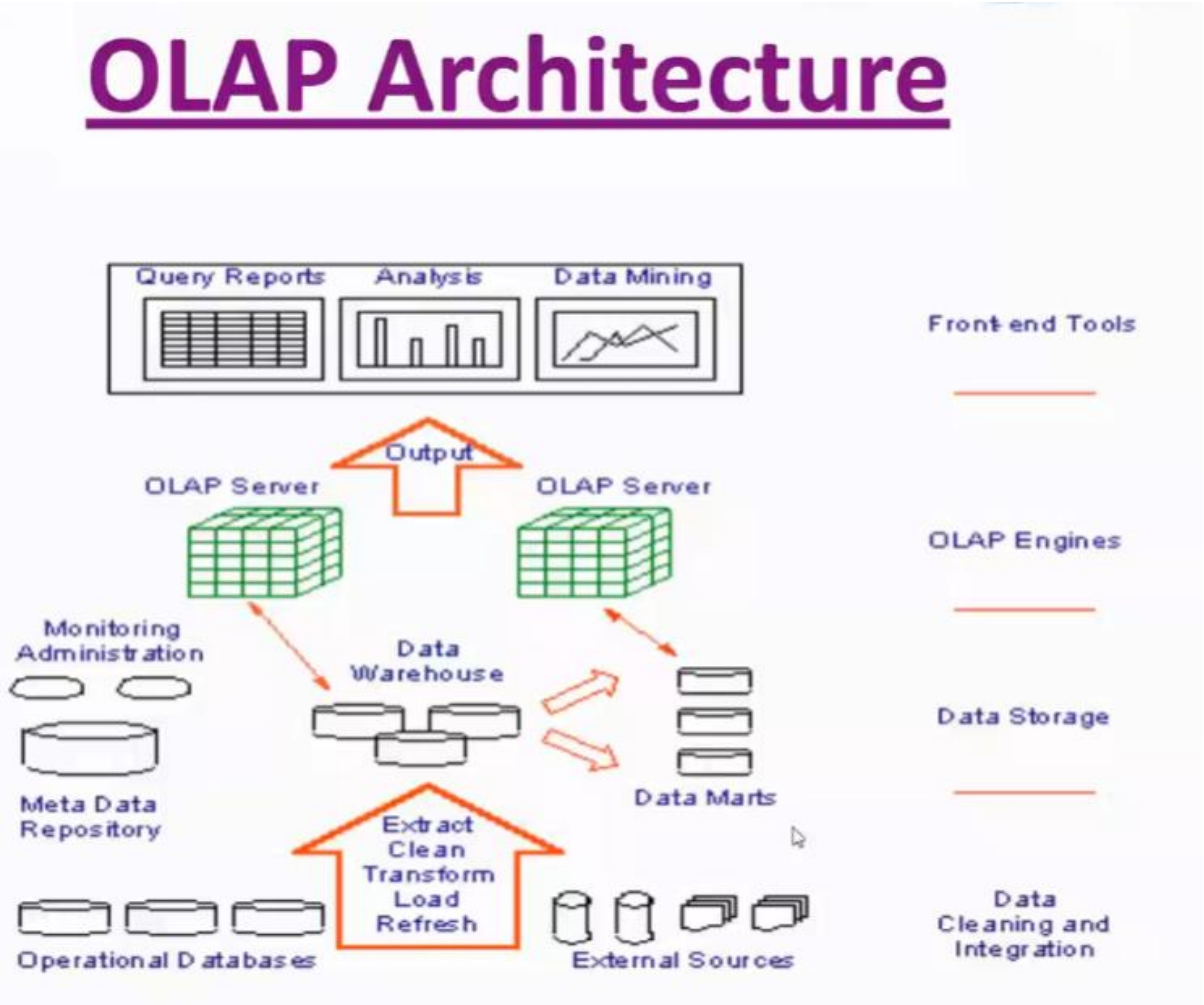
- 🧠 Used for: Decision support, trend analysis, forecasting, dashboards
- 🧊 Unlike OLTP (transactional systems), OLAP is built for **summarized, historical data**, not real-time transactions
- 🎯 Think: "Show me sales by region, category, and month — all in one view"

🎲 OLAP Cube — *The core structure of OLAP*

An OLAP Cube is a **multidimensional data structure** that lets users slice and dice data across many angles.

- 📚 Dimensions: Customer, Product, Time, Region — think of them like filters or grouping categories
- 📦 Measures: Numbers like Sales, Revenue, Quantity
- ⚙️ Supports operations like:
  - **Slice**: View data for one dimension (e.g., just February)
  - **Dice**: View a subcube (e.g., February + Electronics + Chennai)
  - **Drill down**: Zoom into finer details (e.g., Year → Month → Day)
  - **Roll-up**: Summarize (e.g., Day → Month → Quarter)

🧪 Example: Imagine a cube where you can instantly see **monthly sales by region and product category**, and zoom in/out across any axis.



What Is OLAP?

OLAP (Online Analytical Processing) is a system that helps you analyze large amounts of data quickly. It's used for business reports, dashboards, and decision-making.

Think of it like:

"I want to see sales by region, by month, and by product—all at once."

OLAP Architecture – Step-by-Step

1. **Data Sources**
    a. These are your raw data systems (like databases, files, or external sources).
2. **ETL Process**
    a. ETL stands for **Extract, Transform, Load**.
    b. It pulls data from sources, cleans it, and loads it into the warehouse.

3. **Data Warehouse**
    a. A central place where all cleaned and structured data is stored.
4. **Data Marts**
    a. Smaller, focused parts of the warehouse for specific departments (like sales or HR).
5. **OLAP Server**
    a. This is where the real analysis happens.
    b. It organizes data into cubes so you can slice, dice, drill down, and roll up.
6. **Front-End Tools**
    a. These are dashboards, reports, or apps where users view and interact with the data.

Why OLAP Is Useful

- Fast answers to complex questions
- Easy to explore data from different angles
- Supports decision-making with clear insights

What Is an OLAP Server?

An **OLAP Server** is a system that connects to a **data warehouse** and helps users analyze data easily. It organizes the data so users can:

- View summaries
- Drill down into details
- Slice and dice data by different dimensions (like time, region, product)

It's used in **Decision Support Systems (DSS)** to help businesses make informed decisions.

Types of OLAP Servers

1. ROLAP – Relational OLAP

- Uses **relational databases** (like SQL)
- Stores data in **tables**
- Uses **SQL queries** to analyze data
- Good for **large volumes** and **complex queries**
- Slower than MOLAP for some operations because it calculates data on the fly

Example use case: When you have huge datasets and need flexible querying.

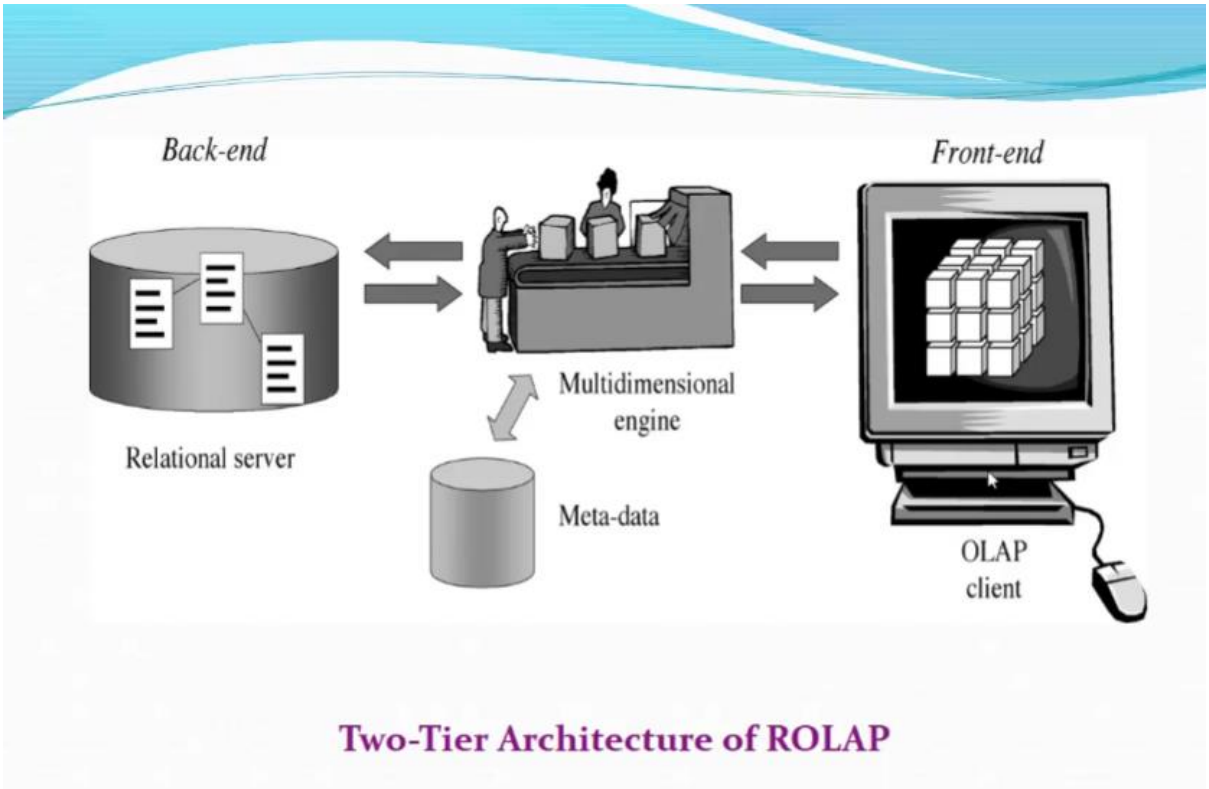
2. MOLAP – Multidimensional OLAP

- Uses **multidimensional databases**
- Stores data in **cubes**
- Pre-calculates summaries for fast access
- Very fast for **read-heavy** operations
- Limited flexibility compared to ROLAP

Example use case: When you need fast reports and dashboards with pre-aggregated data.

SUMMARY:

Feature	ROLAP	MOLAP
Storage	Relational tables	Multidimensional cubes
Speed	Slower (on-the-fly queries)	Faster (pre-aggregated data)
Flexibility	High	Moderate
Data Volume	Handles large data well	Best for summarized data
Query Language	SQL	Proprietary cube operations



Two Main Parts of ROLAP Architecture

1. Back-End (Where Data Lives and Gets Processed)

- **Relational Server**: This is your regular database (like MySQL or PostgreSQL) where all the raw data is stored.
- **Multidimensional Engine**: This engine takes the table data and organizes it so you can view it by dimensions like time, region, or product.
- **Metadata**: Extra information that helps the system understand how the data is structured and how to retrieve it efficiently.

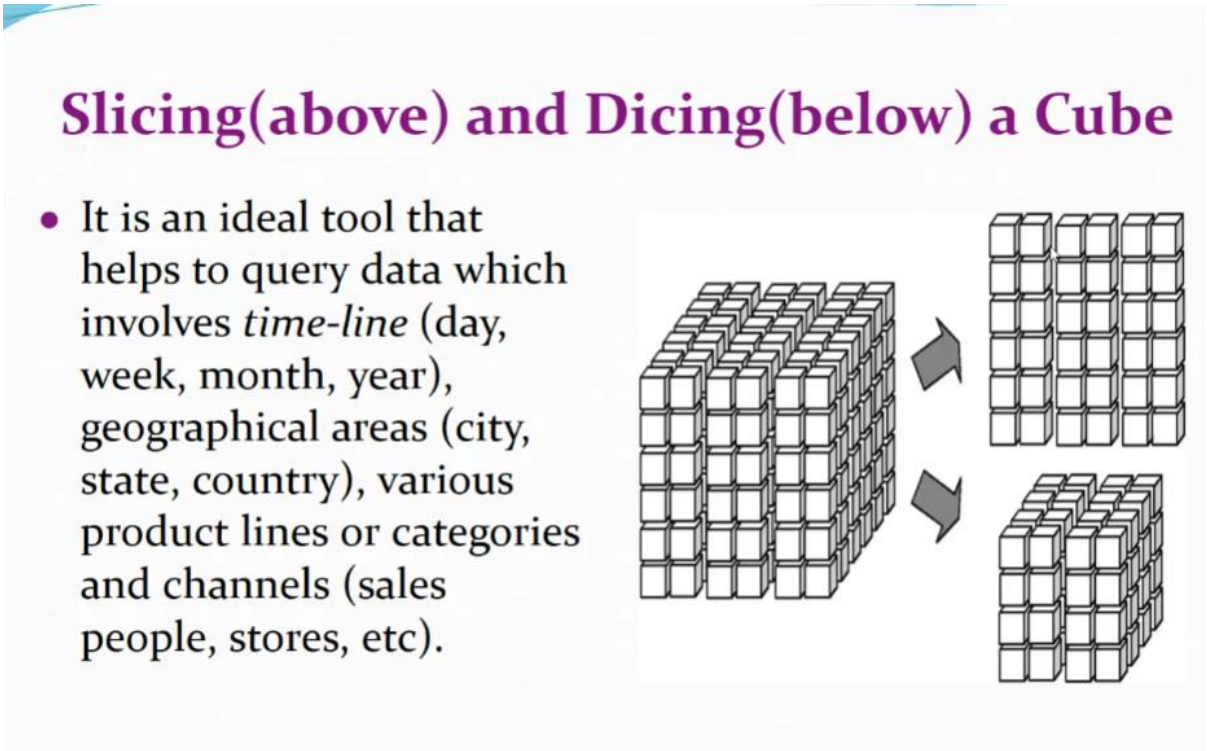
2. Front-End (Where Users See and Use the Data)

- **OLAP Client**: This is the tool or dashboard that users interact with. It shows data in a cube-like format and lets users ask questions like:
  - "Show me sales by month"
  - "Compare regions for this product"

Why It's Called Two-Tier

Because there are two layers:

- One for storing and processing data (back-end)
- One for displaying and analyzing data (front-end)



What Is a Data Cube?

A **data cube** is a way to organize data so you can look at it from different angles—like time, location, product, etc. It's used in business analysis to answer questions like:

- "What were the sales in Chennai last month?"
- "Which product sold best in 2022?"

What Is Slicing?

Slicing means taking one layer of the cube.

Example:

"Show me all sales for Chennai across all months."

You're looking at one region (Chennai) and ignoring other regions.

What Is Dicing?

Dicing means taking a smaller block from the cube.

Example:

“Show me sales for Chennai in January for electronics.”

You’re narrowing down by region, time, and product.

Why Use Slicing and Dicing?

They help you:

- Focus on specific data
- Compare different parts of your business
- Make better decisions by analyzing patterns

1. Information Processing

This is about organizing and showing data clearly. You use:

- Tables
- Charts
- Graphs

Example:

“Show me last month’s sales in a report.”

2. Analytical Processing

This is about exploring data from different angles. You use:

- Slicing (one layer of data)
- Dicing (small block of data)
- Drilling (zoom in or out)
- Pivoting (rearranging views)

Example:

“Compare sales by region and month.”

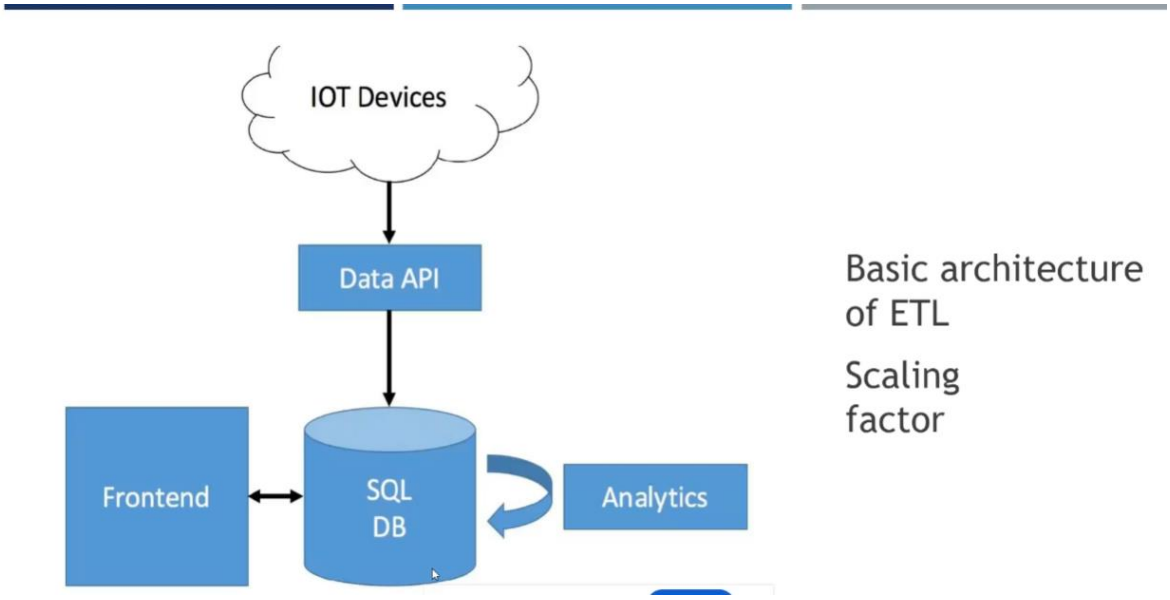
3. Data Mining

This is about finding hidden patterns or trends. You use:

- Prediction
- Classification
- Pattern discovery

Example:

“Which customers are likely to buy again?”



What Is ETL?

ETL stands for:

- **Extract**: Take data from a source (like IoT devices)
- **Transform**: Clean or change the data to make it useful
- **Load**: Put the data into a database

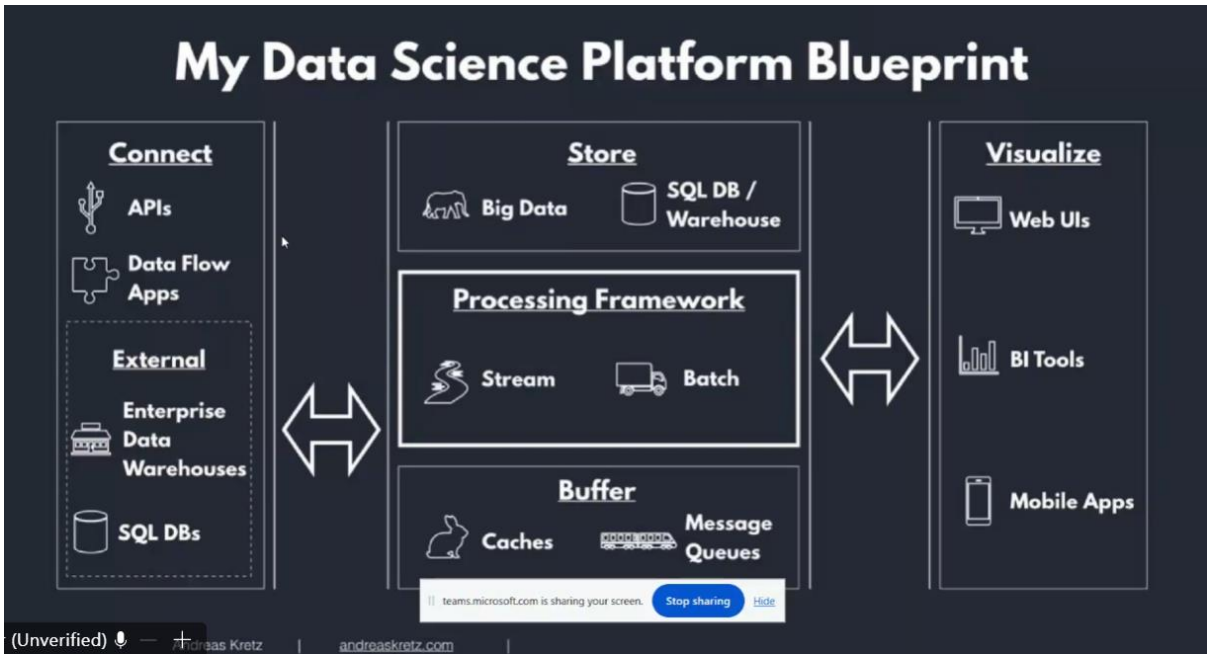
What the Image Shows (Step-by-Step)

1. **IoT Devices** These are smart devices that collect data (like sensors or machines).
2. **Data API** This is a tool that helps move the data from the devices to the database.
3. **SQL Database** This is where the data is stored in a structured format.

4. **Analytics** This part looks at the data and finds patterns or insights.
5. **Frontend** This is the app or dashboard that shows the data to users.

What Is Scaling Factor?

It means the system can grow—handle more devices, more data, and more users without breaking.



Overview

This architecture shows how a complete data science system works—from collecting data to showing insights to users. It's divided into five key layers:

1. **Connect**
2. **Store**
3. **Process**
4. **Buffer**
5. **Visualize**

Each layer plays a specific role in handling data efficiently.

1. Connect – Bringing Data In

This layer is responsible for collecting data from different sources:

- **APIs**: Interfaces that pull data from apps, websites, or devices.

- **Data Flow Apps**: Tools that automate data collection and movement.
- **Sources**: These can be:
    - Enterprise systems (like ERP or CRM)
    - SQL databases
    - External services (like weather data or social media)

The goal here is to make sure data enters the system reliably and continuously.

2. Store – Saving the Data

Once data is collected, it needs to be stored:

- **Big Data Storage**: For large volumes of unstructured data (like logs, images, or sensor data).
- **SQL Databases / Data Warehouses**: For structured data that fits into rows and columns.

This layer ensures data is organized and accessible for analysis.

3. Process – Making Data Useful

This layer transforms raw data into meaningful information:

- **Stream Processing**: Handles real-time data (e.g., live temperature readings).
- **Batch Processing**: Handles large datasets at scheduled intervals (e.g., daily sales reports).

Processing cleans, filters, and prepares data for analysis.

4. Buffer – Managing Flow and Speed

This layer helps control how fast and smoothly data moves:

- **Caches**: Temporary storage for quick access to frequently used data.
- **Message Queues**: Systems that hold data temporarily and send it to the next step when ready.

This prevents overload and ensures stability.

5. Visualize – Showing Results to Users

This is the final layer where users interact with the data:

- **Web UIs**: Dashboards and web apps for analysts and decision-makers.
- **BI Tools**: Tools like Tableau or Power BI for creating reports and charts.
- **Mobile Apps**: For accessing insights on the go.

This layer turns processed data into clear, actionable insights.

Summary

This architecture helps data scientists and engineers:

- Collect data from many sources
- Store it efficiently
- Process it for insights
- Manage flow and performance
- Present it to users in a useful way