DEPARTMENT OF CS & IT

DATA PREPARATION AND VISUALIZATION

UNIT -2

2.1 What is Data Preparation?

Data preparation is the process of converting raw, unstructured data into a clean, structured, and standardized format for analysis. It involves cleansing, transforming, and validating data to ensure it meets organizational quality standards. This process is essential for making data usable and reliable for decision-making.

Though the data preparation process can be long and involved, it's essential in making data usable. Without proper preparation, there is no guarantee that the data your organization ingests will be accurate, complete, accessible, or reliable.

2.2 Why is Data Preparation Important?

Data preparation is crucial because it ensures that the data used for analytics is accurate, consistent, and reliable, which directly affects the quality of insights and business decisions.

1. **Ensures Data Accuracy & Consistency**
   - Raw data often contains missing values, inaccuracies, and format mismatches.
   - Preparation corrects these issues so analytics tools work with clean and trustworthy data.
2. **Improves Data Quality**
   - By removing errors, handling outliers, and standardizing formats, data becomes more usable and meaningful.
   - Enhances the validity of analysis results.
3. **Combines and Consolidates Data Sets**
   - Different sources often have different formats or structures.
   - Data prep merges and reconciles these variations to create a single, cohesive dataset.
4. **Enriches and Optimizes Data**
   - Blends internal and external data (e.g., sales data with market trends).
   - Creates new features (e.g., derived columns).
   - Balances data sets to avoid bias (especially in machine learning).
5. **Supports Business Decision-Making**
   - Makes sure analytics deliver actionable insights, not misleading outputs.
   - Helps businesses make data-driven decisions with confidence.
6. **Enables Self-Service Analytics**
   - Curated, prepped data allows business users to explore and analyze data without IT help.
   - Boosts productivity and decision-making across teams.

Data preparation is the foundation of effective analytics. It ensures data is clean, consistent, and enriched so that insights are valid, actionable, and aligned with business goals.

2.3 Steps in the Data Preparation Process

Data preparation is a systematic, multi-step process used to clean, structure, and organize raw data for analysis. While specific approaches may vary, the core steps typically include the following:

I. Data Collection

The first step in data preparation is acquiring the data. You need to know what data you need, what data is available, and how to gather that data.

Goal: Gather relevant data from various sources such as:

1. **Operational systems (ERP, CRM)**
    - **ERP System**: Pulling sales and inventory data from SAP ERP.
        - **Scenario:** A retail company uses SAP ERP to manage sales, inventory, and purchasing. The business analyst wants to prepare a monthly sales and stock availability report.
        - **Example: Data Extracted from SAP ERP:**

Product ID	Product Name	Units Sold	Stock Remaining	Sales Date
P101	Laptop 15"	120	50	2025-07-01
P102	Mouse USB	300	200	2025-07-01
P103	Keyboard	150	80	2025-07-01

        - **Usage:**
            - Sales Data → Used to track revenue and sales trends.
            - Inventory Data → Helps forecast reordering needs.
    - **CRM:** Extracting customer interaction logs from Salesforce.
        - **Scenario:** A service company uses Salesforce CRM to store customer communication history. The marketing team wants to analyze which customers are most engaged.
        - **Example: Data Extracted from Salesforce:**

Customer ID	Name	Interaction Type	Date	Notes
C001	Rohini S	Email	2025-07-02	Sent promotional offer
C002	Anil Kumar	Call	2025-07-03	Discussed product demo
C003	Meena R	Chat	2025-07-05	Asked about service pricing

        - **Usage:**
            - **Interaction Logs** → Identify active customers for targeted campaigns.
            - **Customer Notes** → Help sales teams personalize follow-ups.
2. **Data warehouses and data lakes**
    - **Data Lakes:** Importing raw clickstream data from AWS S3.
        - **Scenario:** An e-commerce website tracks every click, scroll, and page visit of users to understand browsing patterns. These events (clickstream data) are stored as raw JSON or CSV files in AWS S3 — this storage acts as a data lake.
        - **Example Raw Clickstream File (JSON) stored in S3:**

```
[
  {
   "user_id": "U123",
   "session_id": "S5678",
   "event_type": "page_view",
   "page_url": "/products/laptop",
   "timestamp": "2025-08-13T10:15:32Z",
   "device": "mobile",
   "location": "India"
  },
{
  "user_id": "U123",
  "session_id": "S5678",
  "event_type": "click",
  "element_id": "add_to_cart_button",
  "page_url": "/products/laptop",
  "timestamp": "2025-08-13T10:15:50Z",
  "device": "mobile",
  "location": "India"
 }
]
```

- **Usage:**
  - **Raw Storage** – S3 keeps the unprocessed clickstream logs exactly as they were captured.
  - **Analytics** – Data engineers later load these logs into AWS Athena or Redshift for analysis.
  - **Insights** – Identify which product pages get the most clicks, or detect where users drop off before purchase.

3. **Cloud databases and external sources**
   - **Cloud Databases – Retrieving Real-Time User Activity from Firebase or Google BigQuery**
     - **Scenario:** A mobile gaming app uses Firebase to store player activity logs in Cloud Firestore. The data is also connected to Google BigQuery for large-scale analysis.
     - **Example Data from Firebase / BigQuery:**

user_id	session_id	event_type	level	score	timestamp
U101	S5001	level_start	1	0	2025-08-13T08:15:32Z
U101	S5001	level_finish	1	120	2025-08-13T08:20:05Z
U102	S5002	purchase	NULL	500	2025-08-13T08:25:42Z

     - **Usage:**
       - **Real-Time Tracking:** Monitor player engagement and purchase patterns.
       - **Analytics:** Use BigQuery SQL queries to find the average score per player or detect churn risk**.**
   - **External Sources – Using APIs to Collect Weather Data from Open Weather Map to Correlate with Sales**
     - **Scenario:** A cold-drink company wants to see if hot weather increases sales. They use the OpenWeatherMap API to fetch daily weather data for each city where they operate.
     - **Example Weather API Response (JSON):**

```
{
 "city": "Chennai",
 "date": "2025-08-13",
 "temperature": 35,
 "weather": "clear sky",
 "humidity": 60
}
```

date	city	product	units_sold

2025-08-13	Chennai	Cola 500ml	1200
2025-08-13	Delhi	Cola 500ml	800

- **Correlation Use Case:**
    - Join sales data with weather data on date + city.
    - Analyze whether sales increase when temperature > 30°C.

Key Points:

- Ensure data aligns with the business goals and analysis objectives.
- Collaboration between data scientists, engineers, BI teams, and end users is essential at this stage.

II. Data Discovery and Profiling

Understanding the data's structure, content, and potential issues by exploring the datasets and identifying key attributes and extracts common patterns and relationships in the data. This stage marks the beginning of data quality management. You should thoroughly examine the data to identify any missing values, outliers, and inconsistencies.
Goal: Understand the content, structure, and quality of the collected data.

Includes:

- **Data profiling:** Analysing statistics (mean, max, nulls, etc.), data types, patterns.
    - Calculating null values in the "Customer Email" column.
    - Checking data types: "**Order Date**" should be a datetime field, not text.
    - Finding duplicates in "**Customer ID**".
- **Identifying:**
    - Relationships between columns/tables
        - Foreign key relationship between **Orders.customer_id** and **Customers.id**.
    - Missing values, anomalies, and duplicates
        - A product's price showing as ₹0 or an age listed as 200.
    - Data quality issues

Note: ***Data profiling*** *is different from* ***data mining.***
Data profiling focuses on understanding data structure and quality, while **data mining** focuses on finding patterns and insights**.**

Example:

Dataset Before Profiling

Customers Table

customer_id	name	email	age
C001	Rohini S	rohini@email.com	28
C002	Anil Kumar	**NULL**	200
C003	Meena R	meena@email.com	32
C003	Meena R	meena@email.com	32

Orders Table

order_id	customer_id	order_date	product	price
O1001	C001	"2025-08-01"	Laptop	50000
O1002	C002	"08-05-2025"	Mouse	0

O1003	C003	"2025/08/07"	Keyboard	2500

Data Profiling Findings:

1. **Calculating Null Values**
   - **"Customer Email"** column has 1 missing value (C002).
2. **Checking Data Types**
   - **"Order Date"** column has inconsistent formats:
     - "2025-08-01" (ISO format)
     - "08-05-2025" (MM-DD-YYYY)
     - "2025/08/07" (YYYY/MM/DD)
       → Should be standardized to a datetime type.
3. **Finding Duplicates**
   - **"Customer ID"** C003 appears twice in the Customers table with identical details → needs deduplication.
4. **Identifying Relationships**
   - Orders.customer_id should match an ID in Customers.customer_id.
     ✅ All match in this example, but profiling confirms referential integrity.
5. **Detecting Anomalies**
   - "Price" of Mouse = ₹0 → possible data entry error.
   - "Age" of C002 = 200 → unrealistic, likely incorrect input.

III. Data Cleansing (or Cleaning)

Any data with issues identified in the discovery and profiling stage should be separated from the presumably higher-quality data ingested. At this point, attempts should be made to cleanse the dirty data, using a variety of techniques

Goal: Fix data quality issues identified during profiling.

Common tasks:

- Fix Inaccurate Entries: Change "Untied States" to "United States".
- Handle Missing Data: Fill missing "City" based on ZIP Code.
- Deduplication: Remove repeated customer entries with same name, email.
- Standardization: Convert date formats (e.g., DD-MM-YYYY → YYYY-MM-DD).
- Remove or fix faulty/inaccurate data
- Tools Used: OpenRefine, Excel, Python (Pandas), Talend.

Best Practices:

- Define internal data quality standards
- Create a data cleansing plan
- Train staff in cleansing techniques
- Continuously monitor data quality

Example:

Before Cleaning – Raw Customer Data

customer_id	name	email	city	zip_code	country	join_date
C001	Rohini S	rohini@email.com	Chennai	600001	Untied States	15-08-2025

C002	Anil Kumar	anil@email.com	**NULL**	560002	United States	12-08-2025
C003	Meena R	meena@email.com	Bangalore	560002	United States	08-08-2025
C003	Meena R	meena@email.com	Bangalore	560002	United States	08-08-2025

Cleansing Actions Applied

1. **Fix Inaccurate Entry** → "Untied States" → "United States" for C001.
2. **Handle Missing Data** → Fill C002's "City" as "Bangalore" based on zip_code = 560002.
3. **Deduplication** → Remove duplicate row for C003.
4. **Standardization** → Convert "join_date" format from DD-MM-YYYY to YYYY-MM-DD.

After Cleaning – Processed Customer Data

customer_id	name	email	city	zip_code	country	join_date
C001	Rohini S	rohini@email.com	Chennai	600001	United States	2025-08-15
C002	Anil Kumar	anil@email.com	Bangalore	560002	United States	2025-08-12
C003	Meena R	meena@email.com	Bangalore	560002	United States	2025-08-08

IV. Data Structuring

Data from a variety of sources may come in numerous formats. Some will be structured, some unstructured. Standardizing data structure is important for future access. All ingested data must eventually conform to your organization's standard data structure, which means analysing that data's original structure and mapping it to your standard structure.

Goal: Model and organize the data into usable formats.

Tasks:

- Convert raw or unstructured data (e.g., CSV, JSON, logs) into structured formats like relational tables.
- Organize the data to meet the needs of BI and analytics tools.

Before Structuring – Raw Nested JSON (Unstructured Data)

```
[
  {
   "customer_id": "C001",
   "name": "Rohini S",
   "email": "rohini@email.com",
   "orders": [
     {
      "order_id": "O1001",
      "date": "2025-08-10",
      "items": [
        { "product_id": "P101", "product_name": "Laptop", "price": 50000 },
        { "product_id": "P102", "product_name": "Mouse", "price": 500 }
      ]
     }
   ]
  }
]
```

Structuring Process

1. **From JSON to Table:** Load JSON into Python and convert it into a flat tabular format.

2. **Flatten Nested Data:** Separate into **Customers**, **Orders**, and **Products** relational tables.

After Structuring – Relational Tables

Customers Table

customer_id	name	email
C001	Rohini S	rohini@email.com

Orders Table

order_id	customer_id	date
O1001	C001	2025-08-10

Products Table

order_id	product_id	product_name	price
O1001	P101	Laptop	50000
O1001	P102	Mouse	500

This makes the data ready for SQL queries, BI tools, and analytics instead of staying locked in complex JSON.

V. Data Transformation and Enrichment

Transforming data enriches it, providing additional insights beyond that contained in the raw data itself and contributing to better decision-making.

Goal: Transform and enhance the data to make it more useful.

Data Transformation:

- Modify the structure/values for usability:
  - Create derived fields (e.g., total = price × quantity)
  - Aggregate or filter data (e.g., Remove inactive users from the user dataset.)
  - Normalize or encode values (e.g., Convert income levels into categories (e.g., Low, Medium, High).

Example

Before Transformation – Sales Data (Internal)

order_id	customer_id	price	quantity	status
O1001	C001	50000	1	Completed
O1002	C002	2000	2	Cancelled
O1003	C003	1500	3	Completed

Data Transformation Steps

1. Create Derived Field – > total = price × quantity
2. Filter Data – Keep only status = Completed
3. Normalize Values – Convert total into spending categories:
   - Low: < ₹5,000
   - Medium: ₹5,000 – ₹20,000
   - High: > ₹20,000

After Transformation

order_id	customer_id	price	quantity	total	spending_category
O1001	C001	50000	1	50000	High
O1003	C003	1500	3	4500	Low

Data Enrichment:

- Add additional context:
  - Merge internal and external data sources (e.g., Append demographic data from census API to internal customer database.)
  - Add metadata (e.g., Tag fields with source system and update timestamp.) or calculated fields (e.g., Add "Customer Lifetime Value" based on transaction history.)
  - Improve data usefulness and completeness

Example:

Data Enrichment Steps

1. **Merge External Data** – Append demographic info from a **Census API**:
   - Age group, region, income bracket.
2. **Add Metadata** – Tag each row with:
   - source_system = "ERP"
   - update_timestamp = current date/time
3. **Add Calculated Field** – Calculate **Customer Lifetime Value (CLV)** from historical purchases.

After Enrichment

order_id	customer_id	total	spending_category	age_group	region	CLV	source_system	update_timestamp
O1001	C001	50000	High	25-34	Chennai	120000	ERP	2025-08-13 10:30:00
O1003	C003	4500	Low	35-44	Bangalore	25000	ERP	2025-08-13 10:30:00

VI. Data Validation and Publishing

Goal: Ensure the data is ready for use and made accessible to users.

Steps:

- Run automated validation checks for accuracy, consistency, and completeness.
  - E.g., Ensure no nulls in mandatory fields like Product ID.
  - E.g., Check if all postal codes match known country-specific formats.
- Store the final data in:
  - Data warehouses
  - Data lakes
  - Analytics platforms or BI tools
- Make data available to:
  - Business analysts
  - Data scientists
  - Executives
  - Other users (via dashboards, datasets, APIs)

2.4 Common Data Cleaning Techniques

Data cleaning is a critical part of the data preparation process. It ensures that data is accurate, consistent, and ready for analysis by handling errors, inconsistencies, and irrelevant information.

2.4.1. Handling Missing Values

Causes:

- Data entry mistakes
- Incomplete forms or surveys
- System errors or transmission failures.

Techniques:

1. **Imputation:**

   Imputation is the process of **replacing missing or incomplete data with substituted values** so the dataset becomes complete and usable for analysis.

   - The substituted values can be **simple** (like mean, median, mode) or **advanced** (like predictions from regression or machine learning models).
   - **Goal:** Preserve as much information as possible and avoid bias caused by missing data.

     Example:

     If the "Age" column has missing entries, you might replace them with the **average age** of all other records, or **predict** them using related features like education level and income.

   A. **Replace missing values with:**

      a. **Mean, median, or mode (for numerical data) -** This is a simple imputation technique where missing numerical values are replaced with a single representative value from the available data.

      - **Mean imputation**: Replace missing values with the average of the existing values.
      - **Median imputation:** Replace missing values with the middle value when data is sorted (less sensitive to outliers).
      - **Mode imputation**: Replace missing values with the most frequent value (usually for categorical or discrete numeric data).

      Example:

      Suppose we have ages of customers:

Customer ID	**Age**
101	25
102	—
103	28
104	27

      - **Mean:** (25 + 28 + 27) ÷ 3 = **26.67** → Replace missing age with **26.67.**
      - **Median:** Middle value of (25, 27, 28) is **27** → Replace missing age with **27.**
      - **Mode:** If the most frequent age is 27, replace missing age with **27.**

      a. Mean/Median/Mode Imputation:

      b. **Most frequent category (for categorical data)**

      For missing categorical values, we replace them with the mode — the category that occurs most often in the dataset.

      - This works well when the missing data is small in proportion and the most frequent category is representative of the dataset.

Example:
Suppose we have a dataset of customers and their preferred payment methods:

Customer ID	Payment Method
201	Credit Card
202	—
203	Cash
204	Credit Card
205	Credit Card

Step 1: Find the most frequent category:

- "Credit Card" appears **3 times**
- "Cash" appears **1 time**

Step 2: Replace missing values with "Credit Card":

Customer ID	Payment Method
201	Credit Card
202	Credit Card
203	Cash
204	Credit Card
205	Credit Card

2. **Advanced methods:**

A. **Regression imputation** - A missing data handling technique where a statistical model (such as linear regression) is built using the non-missing records, and the missing values are predicted based on other related variables. (Use the relationships between variables to make the best guess for the missing value.)

B. Multiple imputation - An **advanced statistical method** for handling missing data that replaces each missing value with **several different plausible values** based on predictions from a model, creating multiple complete datasets. (Each dataset is analysed separately, and the results are **combined** to account for uncertainty and reduce bias.)

Example:
Dataset (Before Handling Missing Data)

customer_id	age	education	income
C001	28	Bachelor's	₹35,000
C002	40	Master's	₹60,000
C003	35	Bachelor's	Missing
C004	50	High School	₹25,000
C005	30	Master's	Missing

Regression Imputation

- **Goal:** Use a regression model to predict missing income values using **age** and **education** as predictors.
- **Example:** For C003, the model might learn that *Bachelor's degree + Age 35* usually corresponds to ~₹38,000.
- Predicted:
  - C003 → ₹38,000
  - C005 → ₹55,000

Multiple Imputation

- **Goal:** Create several versions of the dataset with different plausible values for the missing entries, then combine results for analysis.
- Example: For C003's missing income, the method may generate:
    - Imputation 1: ₹37,500
    - Imputation 2: ₹38,200
    - Imputation 3: ₹39,000
- Final estimate is the **statistical combination** (mean or pooled analysis) → ₹38,233.

Final Dataset (After Imputation)

customer_id	**age**	**education**	**income**
C001	28	Bachelor's	₹35,000
C002	40	Master's	₹60,000
C003	35	Bachelor's	₹38,233
C004	50	High School	₹25,000
C005	30	Master's	₹55,000

3. **Deletion:**
   A method for handling missing data by **removing** entire rows (records) or columns (features) that contain missing values.
    - **Row deletion**: Removes the whole record if any required value is missing.
    - **Column deletion**: Removes the whole column if too many values are missing or the column is not essential.
    - **Risk:** Can cause loss of important data, especially if the missingness is not random.

Example 1 – Row Deletion:
Dataset before cleaning:

ID	**Name**	**Age**	**City**
1	Rahul	28	Mumbai
2	Priya	—	Delhi
3	Arjun	31	—
4	Meera	25	Chennai

Step: Remove rows with missing values:

ID	Name	Age	City
1	Rahul	28	Mumbai
4	Meera	25	Chennai

Example 2 – Column Deletion:
If a dataset has a "Middle Name" column with **90% missing values**, it can be dropped entirely since it adds little value

2.4.2. Removing Duplicates

Removing duplicates is the process of identifying and eliminating redundant records from a dataset so that each entry is unique.

Why it's important:

- Prevents inflated counts in reports.
- Avoids biased analysis in statistics and machine learning.
- Improves data quality and storage efficiency**.**

How to do it:

1. Identify duplicates using a unique key (e.g., Customer_ID, or a combination like Name + Date).
2. Remove duplicates using:
    - Excel → *Data* → *Remove Duplicates*.

- Python (Pandas) → df.drop_duplicates().
- SQL → Use DISTINCT or ROW_NUMBER () with filtering.

Example:

Before Cleaning:

Order_ID	Customer	Date	Amount
1	Rahul	2025-08-01	500
2	Priya	2025-08-02	700
1	Rahul	2025-08-01	500

After Cleaning:

Order_ID	Customer	Date	Amount
1	Rahul	2025-08-01	500
2	Priya	2025-08-02	700

2.4.3. Data Type Conversion (Datacasting)

Data type conversion, also called datacasting, is the process of changing the data type of a value or column to match the required format for analysis or storage.

Why it's important:

- Ensures data consistency across systems.
- Prevents calculation errors (e.g., treating numbers as text).
- Allows correct sorting, filtering, and aggregation.

Examples:

- Converting "2025-08-13" from text → datetime for date calculations.
- Changing "100" from string → integer for mathematical operations.
- Casting a decimal 10.5 → integer (result = 10).
- Boolean values from text ("Yes"/"No")

2.4.4. Outlier Detection and Treatment

Outlier detection and treatment is the process of identifying data points that deviate significantly from the rest of the dataset and deciding how to handle them so they do not distort analysis, statistics, or predictive models.

- Outlier Detection → Finding unusual values that are much higher or lower than most of the data.
- Outlier Treatment → Removing, adjusting, transforming, or separately analyzing these values to improve data quality and model accuracy.

Example:

If most students scored between 60–90 on an exam but one student scored 5, that score is an outlier and may need investigation or special handling.

Techniques for Detection:

Example Dataset:

1. Statistical Methods:

- **Z-score**:
  Values beyond ±3 standard deviations

$$Z = \frac{(x - \text{mean})}{\text{std deviation}}$$

-
-

Flag values where `|Z| > 3`.

Example: In a dataset of exam scores (mean = 70, SD = 5), a score of 90 gives Z=4 → outlier.

- **IQR (Interquartile Range)**:

$$\text{IQR} = Q3 - Q1$$

Outliers are:

$$x < Q1 - 1.5 \times IQR \quad \text{or} \quad x > Q3 + 1.5 \times IQR$$

Example: If Q1 = 20, Q3 = 40 → IQR = 20 → Values below -10 or above 70 are outliers.

2. Machine Learning Methods:

- **Clustering (e.g., K-Means)**: Detect data points far from cluster centroids
- **Anomaly Detection Algorithms**: Isolation Forest, One-Class SVM, etc.

Treatment Options:

- Remove outliers (if caused by errors or irrelevant to analysis).
- Cap or Winsorize (replace extreme values with a limit).
- Transform values (e.g., log or square root transformation to reduce skew).
- Analyze separately (if the outliers carry important insights, e.g., fraud cases).

2.5. Historical Perspective – How Did We Get Here?

2.5.1. Early Days of Data Preparation (Before 1980s–90s)

- **Fully Manual Workflows**
  - In the early days (before the 1980s–90s), data preparation involved **manual inspection, correction, and formatting**.
  - Analysts worked directly with **paper reports**, handwritten logs, or small digital files stored on floppy disks.
- **Tools of the Time**
  - Primary tools were **spreadsheets** (like Lotus 1-2-3, early Excel), basic database programs (dBASE), and text editors.
  - Sorting, filtering, and cleaning were done **row by row**, often without automation.
- **Data Sources Were Limited**
  - Data mostly came from **transaction records, surveys, or experiments** — not the complex streams we see today.
  - Volume was typically in **kilobytes to a few megabytes**, so manual handling was possible.
- **Error Handling**
  - Detecting errors relied heavily on **human observation** rather than statistical checks.
  - Cross-checking was done by **double-entry verification** or **manual reconciliation** with source documents.
- **Time and Labor Intensive**
  - Projects could take **weeks** just to prepare a dataset for analysis.
  - Data cleaning was considered a **secondary task**, often underfunded and undervalued compared to the analysis itself.

- **Skill Requirements**
    - Analysts needed **deep domain knowledge** because many errors could only be spotted with contextual understanding (e.g., knowing that a sales figure was unrealistic).
- **Limitations**
    - Manual methods were prone to **human error**.
    - Lack of automation meant **repetition of work** for recurring datasets.
    - Scaling to larger datasets was nearly impossible.

2.5.2. Growth of Data and Rise of ETL Processes (1990s–2000s)

Trigger for Change

- **Enterprise systems**, the **internet**, and **digital records** increased **data volume** and **complexity** exponentially.
- Companies started integrating multiple systems — each with different formats.

ETL (Extract, Transform, Load)

- Became the **standardized workflow** for data integration.
- **Extract**: Pull data from multiple sources — databases, flat files, APIs.
- **Transform**: Cleaning, standardizing, validating, and enriching data.
- **Load**: Store into a **data warehouse** for reporting and analytics.

Impact

- Improved **data consistency**.
- Enabled **centralized storage** (data warehouses like Teradata, Oracle).
- Large-scale **business intelligence** systems became possible.

Limitations

- ETL processes were **batch-oriented** — not real-time.
- Complex ETL pipelines required **specialized engineers**.

2.5.3. Tool-Based Automation and Languages (2000s–2010s)

Automation Shift

- Rise of **Excel macros**, **SQL queries**, **R scripts**, and later **Python** for:
    - Data cleaning
    - Statistical transformations
    - Automation of repetitive workflows

Advantages

- Reduced human effort.
- Enhanced **repeatability** and **traceability**.
- More **complex transformations** possible.
- Enabled collaboration between IT and analysts.

Popular Tools

- **Excel**: Pivot tables, formulas, macros.
    - For early data manipulation and spreadsheet-based analysis.
    - Quick calculations, charts, small-scale data cleaning.

- **SQL**: Direct database querying and cleaning. (For querying, joining, filtering, and transforming structured data in databases)
- **R**: Statistical computing. (For statistical analysis and advanced data manipulation)
- **Python**: Pandas, NumPy for data manipulation. (Flexible scripting, automation, and integration with ML.)

Limitations

- Still required **technical skills**.
- Code had to be maintained and documented.

2.5.4. Modern Era: Speed, Automation, and Self-Service (2010s–Present)

Key Changes

- **Real-time data streams** from IoT, social media, and online transactions.
- Explosion of **cloud computing** — scalable infrastructure without large capital cost.

Self-Service Data Prep

- Tools like **Tableau Prep**, **Alteryx**, **Trifacta** allow **non-technical users** to clean and prepare data visually.
- Machine learning aids:
    - **Automated join recommendations**
    - **Anomaly detection**
    - **Auto-filling missing values**

Cloud Advantages

- **Real-time collaboration** across teams.
- **Scalable processing** for large datasets.
- Integration with AI/ML pipelines.

Example Use Case

- Retail company integrates **point-of-sale data**, **weather data**, and **social media sentiment** in real time for demand forecasting.

Limitations

- **Cost** of cloud services.
- **Data privacy** and security concerns.
- Dependency on vendor ecosystems.

Evolution of Data Preparation

Era	**Data Size**	**Tools**	**Speed**	**Automation**	**Skills Needed**
Pre-1980s–90s	KB–MB	Paper, Lotus 1-2-3, dBASE	Very slow	None	Domain expertise
1990s–2000s (ETL)	MB–GB	ETL tools, SQL	Moderate	Partial	Technical + domain
2000s–2010s	GB–TB	Excel, SQL, R, Python	Faster	High	Technical scripting
2010s–Present	TB–PB	Tableau Prep, Alteryx, ML tools	Real-time	Very high	Low to moderate

2.6. What is Self-Service Data Preparation?

Self-service data preparation is the process where **non-technical users** (such as business analysts, marketing managers, or operations staff) can **independently clean, merge, transform, and prepare data** for analysis **without heavy reliance on IT teams or data engineers**.
These solutions typically offer **visual, drag-and-drop interfaces** that make complex data tasks **intuitive and quick**.

Why It Emerged

- Traditional Bottlenecks
  - In older workflows, analysts had to request IT teams for data extraction or cleaning.
  - This created delays of days or even weeks before data was ready for analysis.
- Need for Real-Time Decision Making
  - Business environments have shifted to **faster, more competitive** decision cycles.
  - Analysts needed **instant access** to clean data for quick reports, dashboards, and insights.
- Advancements in Tools
  - Tools like Tableau Prep, Alteryx, and Trifacta introduced visual workflows and automation.
  - Enabled drag-and-drop data cleaning, real-time profiling, and simplified dataset merging.

Key Features

Feature	**Description**
Data Connectivity	Directly connect to multiple sources (Excel, SQL databases, APIs, cloud storage, CRM systems).
Data Profiling	Automatically detect errors, duplicates, and anomalies. Show data quality metrics visually.
Data Cleaning	Remove errors, fix inconsistent values, fill or replace blanks, normalize formats.
Merging & Joining	Combine datasets with point-and-click joins instead of SQL code.
Data Transformation	Reshape datasets (pivot/unpivot, split columns, aggregate values) with a visual interface.
Reusable Workflows	Save data preparation steps as templates for future use.
Automation	Schedule recurring data prep tasks without manual intervention.

```
 ┌─────────────────────┐
 │   Data Sources      │
 │ (Excel, SQL, APIs,  │
 │   Cloud, CRM)       │
 └─────────────────────┘
          │
          │
          ▼
┌───────────────────────┐
│   Connect & Import    │
│ (Drag & Drop Tools)   │
└───────────────────────┘
          │
          │
          ▼
┌───────────────────────┐
│   Data Profiling      │
│(Detect errors, gaps,  │
│ duplicates, patterns) │
└───────────────────────┘
          │
          │
          ▼
┌───────────────────────┐
│   Data Cleaning       │
│ (Fill blanks, fix     │
│ formats, remove dup.) │
└───────────────────────┘
          │
          │
          ▼
┌───────────────────────┐
│ Merge & Transform     │
│ (Join datasets,       │
│ reshape, aggregate)   │
└───────────────────────┘
          │
          │
          ▼
┌───────────────────────┐
│   Analysis & Output   │
│ (Dashboards, reports, │
│   export cleaned data)│
```

Benefits

- **Faster Decision-Making**
  - No waiting for IT — analysts prepare and analyze data immediately.
- **Empowers Analysts**
  - Business teams gain control over their own data.
- **Improves Agility**
  - Supports real-time, on-demand data analysis for changing business conditions.
- **Reduces IT Workload**
  - IT teams focus on infrastructure, security, and governance rather than day-to-day cleaning.

Who Uses It?

- **Business Analysts –** For dashboards, KPIs, and ad-hoc analysis.
- **Data Scientists –** For quick preprocessing before advanced modeling.
- **Marketing Teams –** To merge campaign data, leads, and sales performance.
- **Finance Teams** – To consolidate budgets, expenses, and forecasting data.
- **HR Teams –** To analyze employee performance, attendance, and payroll.

Example Scenario

A retail company's marketing manager needs to compare last month's sales with social media engagement.

- Traditional way: Wait for IT to fetch and clean data → Delay in campaign adjustments.
- Self-service way: Manager directly connects to sales database + social media API, merges datasets visually, cleans missing values, and creates a dashboard in minutes.

Traditional vs Self-Service Data Preparation

Aspect	**Traditional**	**Self-Service**
Who Prepares Data	IT/Data Engineers	Business Analysts & Non-technical users
Speed	Slow (days/weeks)	Fast (minutes/hours)
Skills Required	SQL, scripting	Basic tool navigation
Flexibility	Low	High
IT Workload	Heavy	Reduced

2.7. Introduction to Data Preparation Tools

Modern **data preparation tools** help convert raw, messy data into clean, structured formats suitable for analysis. They differ in **interface**, **functionality**, and **deployment model**, catering to different types of users — from business analysts to data scientists.

A. Types of Tools

1. Programming / Scripting Tools

- **Examples:**
  - **Python:** Pandas, NumPy, PySpark.
  - **R**: dplyr, tidyr, data.table.
- **Purpose:** Allow fine-grained control over data cleaning, transformation, and integration via code.
- **Advantages:**
  - Highly flexible and customizable.
  - Suitable for automation and large-scale batch processing.
  - Enables complex logic that GUI tools may not handle.
- **Best for:**
  Data scientists, data engineers, and technical analysts who are comfortable with programming.
- **Example Use Case:**
  A data scientist uses Python Pandas to merge millions of rows from multiple CSV files, fill missing values, and compute derived features for a machine learning model.

2. Visual Interface Tools

- **Examples:** Tableau Prep, Alteryx, Microsoft Power Query.
- **Purpose:** Prepare data using graphical, drag-and-drop workflows instead of code.
- **Advantages:**
  - Easy for non-programmers.
  - Quick prototyping and exploration.

    - Visual workflows make it easier to explain to stakeholders.
- **Best for:**
  Business analysts, marketing teams, HR analysts, or anyone needing quick data preparation **without coding.**
- **Example Use Case:**
  A marketing analyst uses Tableau Prep to clean campaign performance data and merge it with CRM data before visualization.

B. Standalone vs. Integrated Tools

1. Standalone Tools

- **Examples:** Trifacta, OpenRefine.
- **Purpose:** Dedicated solely to data cleaning, wrangling, and transformation**.**
- **Advantages:**
    - Strong specialized features for data prep.
    - Can connect to diverse sources.
- **Limitations:**
    - Often require exporting the prepared data into another tool for visualization/analysis.
- **Example Use Case:**
  OpenRefine is used to clean inconsistent product names before exporting the dataset into Tableau for dashboarding.

2. Integrated Tools

- **Examples:** Power BI + Power Query, Tableau + Tableau Prep.
- **Purpose:** Combine data preparation + visualization/reporting in one platform.
- **Advantages:**
    - End-to-end workflow from data ingestion → transformation → visualization.
    - Good for real-time analytics.
- **Example Use Case:**
  Power BI with Power Query pulls in sales data, transforms it, and builds a live sales dashboard — all within one environment.

C. Deployment Options

1. Cloud-Based Tools

- **Examples:** AWS Glue, Google Cloud Dataprep (by Trifacta).
- **Advantages:**
    - Scalable for big data.
    - Accessible from anywhere.
    - Automatic updates.
- **Best for:** Distributed teams, large-scale processing, remote collaboration**.**
- **Example:** AWS Glue processes terabytes of customer logs for a global e-commerce site.

2. On-Premises Tools

- **Examples:** OpenRefine, Alteryx (desktop version).
- **Advantages:**
    - Full control over data storage (good for security compliance).
    - No internet dependency.
- **Limitations:** Less flexible for remote teams.
- **Example:** A bank uses Alteryx locally to ensure sensitive financial data stays inside their secure network.

3. Hybrid Tools

- **Example:** Azure Data Factory.

- **Advantages:**
    - Mix of cloud and local processing.
    - Useful during cloud migration.
- **Example:** A healthcare company processes sensitive patient data locally but pushes aggregated statistics to the cloud for analytics.

D. Machine Learning in Data Preparation

- **Why it's important:**
  ML can automate repetitive tasks and improve data quality without heavy manual effort.
- **Features:**
    - **Automatic detection of:**
        - Data types (e.g., date, currency, category).
        - Anomalies (e.g., outliers, invalid entries).
    - **AI-powered suggestions for:**
        - Data transformations (e.g., column splits, merges).
        - Missing value handling (e.g., mean imputation, ML-based prediction).
- **Example:**
  In Trifacta, if two datasets have similar column names and patterns, it suggests the best join method automatically.
- **Benefit:** Reduces preparation time and improves accuracy.

Category	Examples	Best For	Interface Type
Programming	Python (Pandas), R (dplyr)	Data scientists, engineers	Code-based
Visual Interface	Tableau Prep, Alteryx	Business analysts	Drag-and-drop
Standalone	OpenRefine, Trifacta	Deep data cleaning	Mixed
Integrated	Power BI, Tableau	All-in-one workflows	Mixed
Cloud	AWS Glue, Google Dataprep	Big data, remote teams	Web-based
On-Premises	Alteryx Desktop, OpenRefine	Secure local processing	Local install
Hybrid	Azure Data Factory	Cloud + local integration	Mixed

2.8. Users of Data Preparation Tools

Modern data preparation tools are used by various professionals depending on their role, skills, and objectives. These users differ in their technical expertise, the type of data they work with, and their end goals.

1. Data Scientists

- **Role:** Build predictive models, perform advanced statistical analyses, and extract insights from complex datasets.
- **Use of Tools:**
    - Automate repetitive cleaning and transformation tasks.
    - Use scripting tools (Python, R) for flexible, customized data wrangling.
    - Prepare large datasets for machine learning or AI models.
- **Preferred Tools:** Python (Pandas, NumPy), R (dplyr, tidyr), Trifacta, Alteryx.

2. Data Engineers

- **Role:** Design, build, and maintain data pipelines and infrastructure.
- **Use of Tools:**
    - Develop ETL (Extract, Transform, Load) workflows.

    - Integrate multiple data sources into a central warehouse.
    - Optimize data storage formats for scalability and performance.
- **Preferred Tools:** Apache Spark, Talend, AWS Glue, Azure Data Factory, SQL-based ETL systems.

3. Business Analysts

- **Role:** Translate data into actionable business insights and reports.
- **Use of Tools:**
    - Clean and merge datasets to build dashboards and KPI reports.
    - Use visual, drag-and-drop tools for quick preparation without coding.
    - Validate and profile data before analysis.
- **Preferred Tools:** Tableau Prep, Power Query, Alteryx.

4. Data Analysts

- **Role:** Perform exploratory data analysis (EDA) and generate visualizations or summaries.
- **Use of Tools:**
    - Filter, sort, and reshape datasets for deeper analysis.
    - Create aggregated datasets for reporting.
    - Work with both code-based (SQL, Python) and visual tools.
- **Preferred Tools:** SQL, Python (Pandas), Power BI, Excel with Power Query.

5. Information Workers

- **Role:** Non-technical users (e.g., HR, Marketing, Finance staff) who occasionally work with data.
- **Use of Tools:**
    - Prepare small to medium datasets for departmental reporting.
    - Use self-service data prep platforms to clean and transform data without IT dependency.
- **Preferred Tools:** Excel, Google Sheets, Tableau Prep, Zoho Analytics.

✅ **Key Point:**
While data scientists and engineers prefer code-based and automated tools for scalability, business analysts, data analysts, and information workers often rely on self-service and visual tools for faster, non-technical preparation.

2.9. Data Preparation in Analytics Architecture
A. Role in Analytics Architecture
Data preparation is a critical stage in the analytics architecture that sits between data ingestion and analysis/modeling.
It ensures that raw data from diverse sources is cleaned, transformed, and structured to meet the specific requirements of analytical processes.
Position in Architecture:

1. **Data Sources:** Databases, APIs, IoT devices, logs, spreadsheets.
2. **Data Ingestion Layer:** Collects and imports data into storage systems (data lake, data warehouse).
3. **Data Preparation Layer:**
    - Cleanses, integrates, transforms, and formats data.
    - Handles missing values, duplicates, inconsistent formats.
4. **Analytics Layer:**
    - Data visualization, reporting, predictive modeling, machine learning.
5. **Decision Layer:**
    - Uses insights for strategic and operational decisions.

Why It's Important in the Architecture:

- Prevents “garbage in, garbage out” errors in analysis.
- Standardizes data across sources for compatibility.
- Saves analyst time by automating repetitive cleaning tasks.

B. Data Preparation & Analytics Life Cycle

The data analytics life cycle consists of multiple phases that guide analysts through data collection, processing, and deriving insights. Each phase is critical for ensuring accurate, valuable, and actionable results.

1. Discovery

- **What happens:**
  - Identify the business problem or question.
  - Understand data availability, constraints, and potential value.
  - Explore raw data at a high level to see patterns, gaps, and opportunities.
- **Example:**
  A retail company wants to predict customer churn.
  During discovery, the analyst checks existing customer purchase history, demographics, and service usage logs**.**

2. Data Preparation

- **What happens:**
  - Collect, clean, and transform data so it’s ready for analysis.
  - Remove duplicates, handle missing values, correct data types, and merge multiple datasets.
- **Example:**
  The retail company removes inactive accounts, fills missing age values using median, standardizes date formats, and merges sales and customer data.

3. Model Planning

- **What happens:**
  - Decide which algorithms, techniques, and tools to use.
  - Define training/testing data splits.
  - Select metrics for evaluation (e.g., accuracy, recall).
- **Example:**
  The company decides to test Logistic Regression and Random Forest for churn prediction and will measure success with F1-score.

4. Model Building

- **What happens:**
  - Train the chosen models using prepared data.
  - Tune hyperparameters to improve performance.
  - Test the model with unseen data.
- **Example:**
  The retail company trains a Random Forest model on 70% of data and tests it on the remaining 30%, finding that Random Forest performs better than Logistic Regression.

5. Communicate Results

- **What happens:**
  - Present findings through reports, dashboards, or visualizations.
  - Translate technical results into actionable business insights.

- **Example:**
  The analyst creates a dashboard showing that high churn risk is linked to fewer purchases and low engagement in loyalty programs.

6. Operationalize

- **What happens:**
  - Deploy the model into production for real-time or batch predictions.
  - Monitor and maintain model performance over time.
- **Example:**
  The churn prediction model is integrated into the CRM system, automatically flagging customers likely to churn so the marketing team can send retention offers.

Key Point:
This process is iterative — after deployment, feedback and new data may require going back to earlier stages (especially Discovery and Data Preparation) to refine the model.

2.9.1. Continuous Exploration and Discovery

- **Meaning:**
  Analysts and data scientists **keep exploring data at every stage** instead of doing it only in the beginning.
  New patterns, anomalies, or opportunities often emerge as work progresses.
- **Why important:**
  Initial assumptions might be wrong, and new insights can appear after preparing or modeling data.
- **Example:**
  In a healthcare project predicting patient readmission, the team starts by exploring patient demographics and medical history. Later, during model building, they discover that **previous hospital visits** is a stronger predictor than originally thought. They return to the data to extract more detailed visit patterns.

2.9.2. Iterative and Adaptive

- **Meaning:**
  The process is **cyclical** — after completing one stage, you might loop back to previous stages to refine data, update models, or adjust goals.
  The approach adapts to **new findings, changing requirements, or updated data**.
- **Why important:**
  Real-world data projects rarely work perfectly the first time — models need tuning, and business needs evolve.
- **Example:**
  In a retail churn prediction project, after deploying the model, feedback from the marketing team reveals that **seasonal factors** also influence churn. The data team adapts by adding seasonal features and retraining the model.

✅ **In short:**

- **Continuous exploration** = Keep analyzing and questioning data throughout the process.
- **Iterative** = Repeat steps until results are optimized.
- **Adaptive** = Adjust the process when new information or challenges arise.