Introduction to Structured and Unstructured Database Management Systems

Introduction to Databases

What is a Database?

- Definition: A database is an organized collection of data
- Types: Structured, Unstructured
- Purpose: Storing, managing, and retrieving data efficiently.

What is Structured Data?

- Data that follows a specific model or schema.
- - Organized in tables with rows and columns.
- - Easily searchable and processed.

What is Unstructured Data?

- Data that does not follow a specific format
- Examples: Text, video, images, social media posts.
- Difficult to search, process, or analyze without specific techniques.

What is Structured DBMS?

- A database system designed for structured data
- Uses tables with fixed schemas
- Examples: RDBMS

Features of Structured DBMS

- Data stored in tables with rows and columns
- Querying through SQL
- Easy backup, retrieval, and modification
- Examples: MySQL, Oracle, PostgreSQL.

Types of Structured Databases

- Relational Databases (RDBMS): Data is stored in tables
- Hierarchical Databases: Data stored in a tree-like structure

Examples of Structured DBMS

- MySQL
- PostgreSQL
- Oracle DB
- Microsoft SQL Server

What is Unstructured DBMS?

- A database system designed to manage unstructured data
- Data is not organized in a predefined manner
- Examples: NoSQL, document stores, multimedia storage systems.

Features of Unstructured DBMS

- No predefined schema
- Stores data in its raw format
- Flexible and scalable
- Can store multimedia data (images, audio, video)

Types of Unstructured Databases

- NoSQL Databases: MongoDB, Cassandra
- Document Stores: JSON, BSON

Examples of Unstructured DBMS

- MongoDB
- Cassandra
- Hadoop (for big data storage)

Advantages of Structured DBMS

- Easy to store and manage
- Consistent and reliable due to strict schemas
- Supports SQL queries and analytics
- High data integrity

Disadvantages of Structured DBMS

- Not suitable for large-scale unstructured data
- May require complex maintenance.

Advantages of Unstructured DBMS

- Scalable for big data and high-volume information
- Useful for complex data types like media files, logs, etc

Disadvantages of Unstructured DBMS (NoSQL)

- Lack of predefined structure can lead to inconsistencies
- Complex queries are more difficult to perform
- May require special processing tools and techniques.

Use Cases of Structured DBMS

- Financial systems (banking, transactions)
- Enterprise Resource Planning (ERP)
- Customer Relationship Management (CRM)
- Inventory management systems

Use Cases of Unstructured DBMS

- Big data applications (social media data)
- Content management systems (videos, audio)
- Real-time analytics (sensor data)

Structured DBMS - Real-World Example 1

- Banking System
- Description:- Stores transactional data, customer records, account information in relational tables
- Benefit:- consistency, and high performance.

Structured DBMS - Real-World Example 2

- E-commerce Platforms
- Description:- Stores products, customer orders, inventory in structured tables
- Benefit:-Efficient querying and reporting for sales analytics.

Unstructured DBMS - Real-World Example 1

- Social Media Platforms
- Description:- Stores user posts, images, videos in raw formats
- Benefit:- Flexibility to handle diverse types of data.

Structured Data Storage (RDBMS)

- How relational databases store data in tables
- Benefits:- Easy management, backup, and retrieval of transactional data.

Unstructured Data Storage (NoSQL)

- How NoSQL databases store non-relational, flexible data
- Benefits:-Highly scalable, flexible for big data and semi-structured data

Structured DBMS Querying

- Introduction to SQL: SELECT, INSERT, UPDATE, DELETE
-

Database Scalability

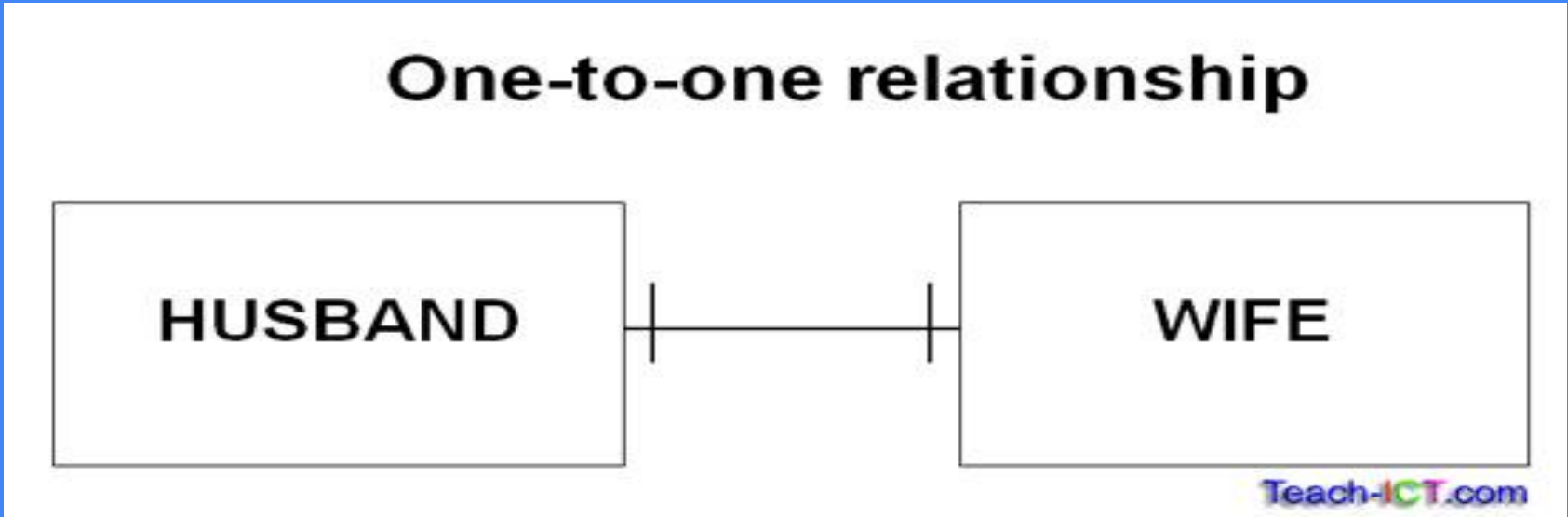
- Structured DBMS: Vertical scalability (upgrading hardware).
- Unstructured DBMS: Horizontal scalability (adding more servers).

Performance Comparison

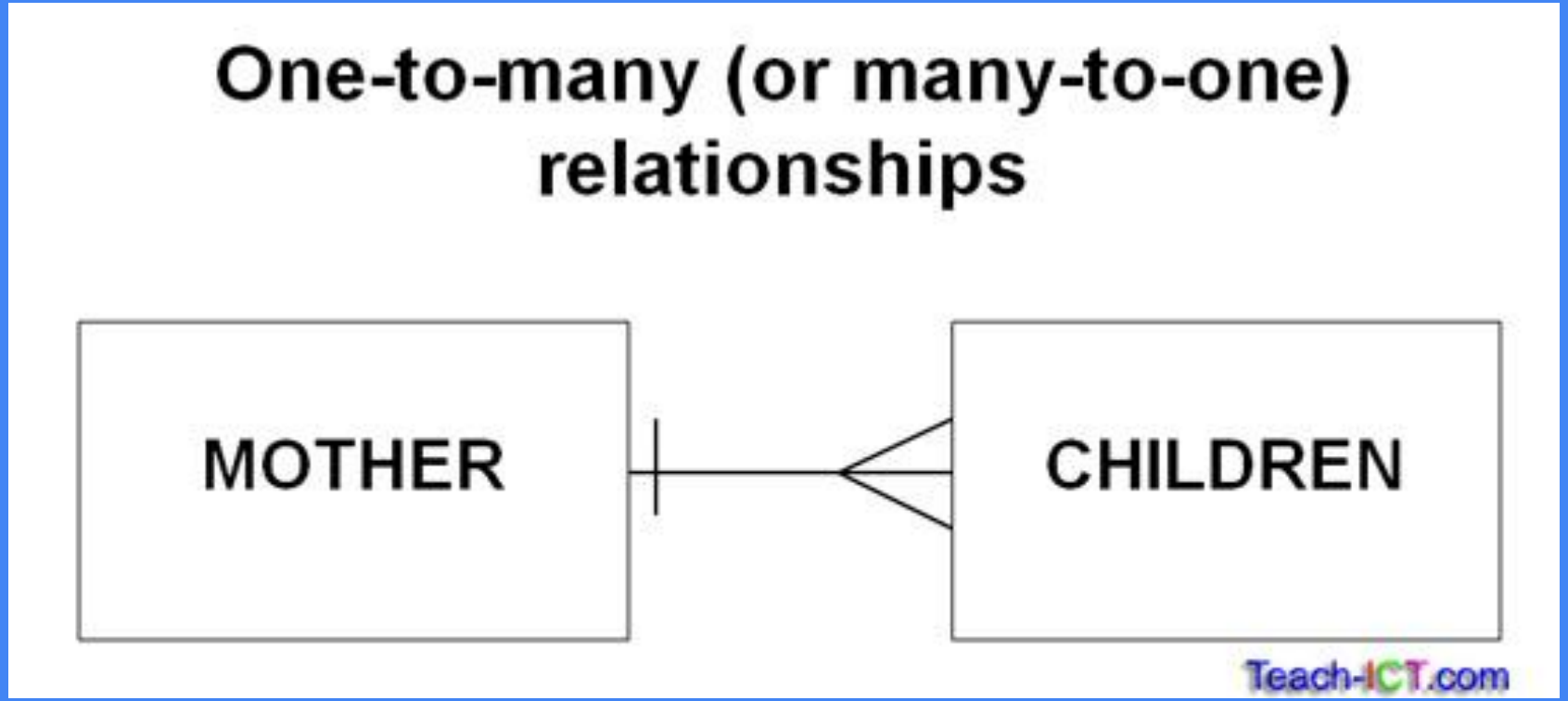
- Structured DBMS:- Faster for complex joins and queries with defined schema
- Unstructured DBMS:- Better for handling large datasets and unstructured queries.

Understanding Data Relationships

One-to-One Relationship



One-to-Many Relationship



Many-to-Many Relationship

CONSTRAINTS

- Def:-CONTSRAINTS ARE USED TO RESTRICTED UNWANTED(INVALID) DATA INTO TABLE
- ALL DATABASES ARE SUPPORTING THE FOLLOWING CONSTRAINT TYPES ARE
- UNIQUE:-TO RESTRICTED DUPLICATE VALUES BUT ACCEPTING NULLS INTO A COLUMN
- NOT NULL:-TO RESTRICTED NULLS BUT ACCEPTING DUPLICATE VALUES INTO A COLUMN

cont……..

- CHECK:-TO CHECK VALUES WITH USER DEFINED CONDITION BEFORE ACCEPTING VALUES INTO A COLUMN
- PRIMARY KEY:-TO RESTRICTED DUPLICATES & NULLS INTO A COLUMN
- A TABLE SHOULD HAVE ONLY "ONE PRIMARY KEY"

- FOREIGN KEY (REFERENCES KEY):-FOREIGN KEY IS USED TO ESTABLISH RELATIONSHIP BETWEEN TABLES
- DEFAULT:-IT A SPECIAL TYPE OF CONSTRAINT WHICH IS USED TO ASSIGN A USER DEFINE DEFAULT VALUE TO A COLUMN

SUB - LANGUAGES OF SQL:

1) DDL (DATA DEFINITION LANGUAGE):
    > CREATE, ALTER, RENAME, TRUNCATE, DROP
    >RECYCLEBIN, FLASHBACK, PURGE (LATEST FEATURES)
2) DML (DATA MANIPULATION LANGUAGE):
    > INSERT, UPDATE, DELETE
    > INSERT ALL, MERGE (NEW COMMANDS)
3) DQL / DRL (DATA QUERY / DATA RETRIVE LANGUAGE):
    > SELECT
4) TCL (TRANSACTION CONTROL LANGUAGE):
    > COMMIT, ROLLBACK, SAVEPOINT
5) DCL (DATA CONTROLL LANGUAGE):
    > GRANT, REVOKE

Aggregate Functions

- COUNT
- SUM
- AVG
- MIN
- MAX