- Db2, MySQL (phpMyAdmin), PostgreSQL (pgAdmin)

Module 1. Fundamental Concept

Module 2. Hands-on

Module 3. MySQL & PostgreSQL

with phpMyAdmin with pgAdmin

Module 4. Course Assignment

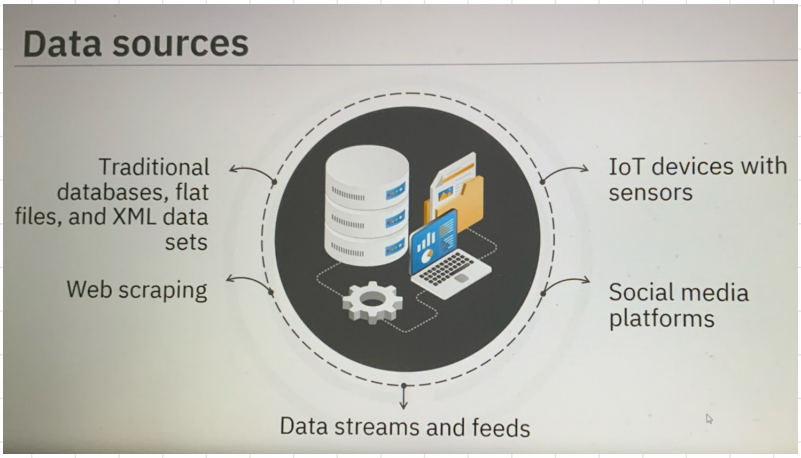
database design, Creating ERDs, final project

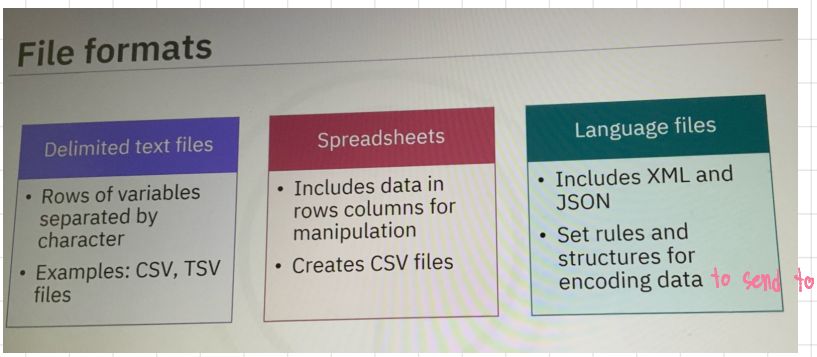
mod.1:
Review Data Fundamental

- Structured data
  - Highly organize
  - Follow a predefined format
  - table : rows, columns
  - strict schema
  - easy retrieval
  - Ex: Spreadsheets, SQL database, online forms (stored in record)
- unstructure data
  - Lacks a specific format
  - Does not conform to rules
  - challenge to process and Analise

  Ex: text file (free form doc.), media file, web possess multimedia, social media
- Semi-structured data
  - Possesses some organizational data
  - Does not follow strict tabular
  - Hierarchical structures or tags to organize information
  - balance flexibility and structure

  Ex. Json file, XML doc., Email:[name, address]: struc. [content]: unstruc.





Data repositories

Store, manage, and organize data

Categories include:
- Relational databases
- Non-relational databases

Offer a structured framework for retrieval and administration

Relational databases

Includes structured data stored in related tables

Along with supporting systems known as RDBMS

Minimizes data duplication with links

Example: IBM DB2, Microsoft SQL Server, Oracle, MySQL

Relational databases

support database activity : customer transaction, HR, workflow

Designed for OLTP systems

Stores high volume of operational data

Ensures transactional integrity

Scope of relational database

OLAP:

rela., non-rela, Data warehouse, Data lake, Big Data

Include various storage solutions

Focuses on querying and analyzing large data sets

Example:
- Sourcing data from CRM for generating sales

Non-relational databases

Offer flexibility in handling diverse and unstructured data

Examples include MongoDB, Cassandra, and Redis

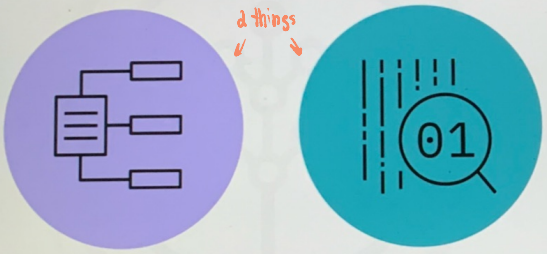
MongoDB:

Stores semi-structured or unstructured data

Information and Data model

Data management

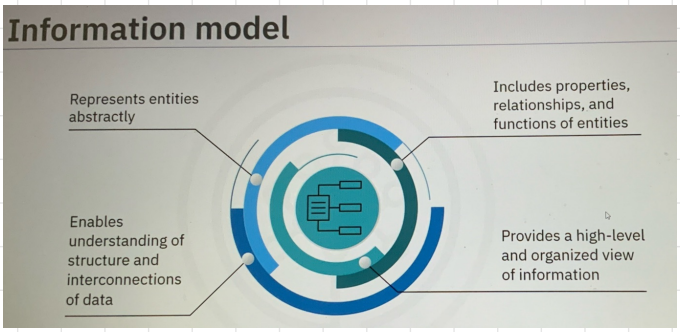
Key concepts in data organization:



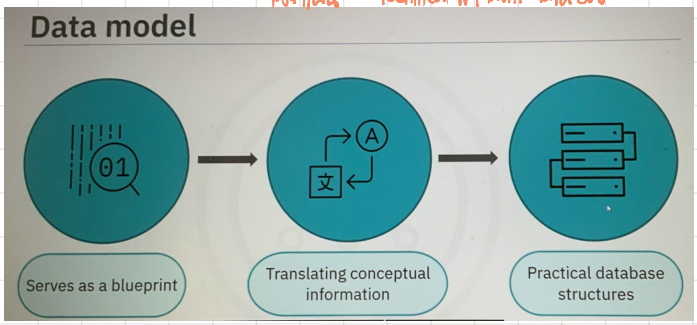
Information Models

Data Models

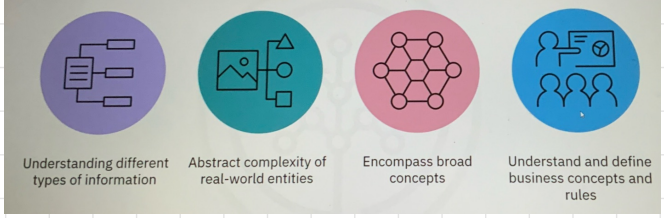
Information model



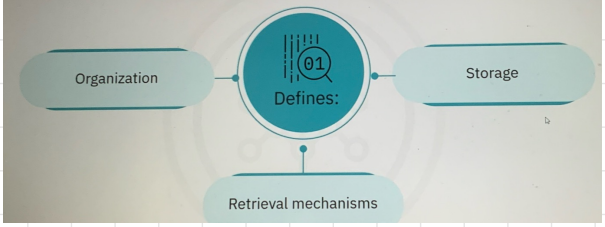
Data model



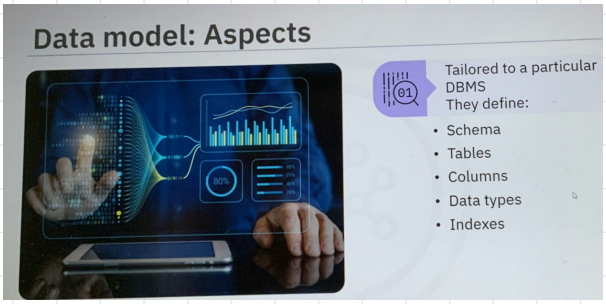
Information model: Aspects



Data model

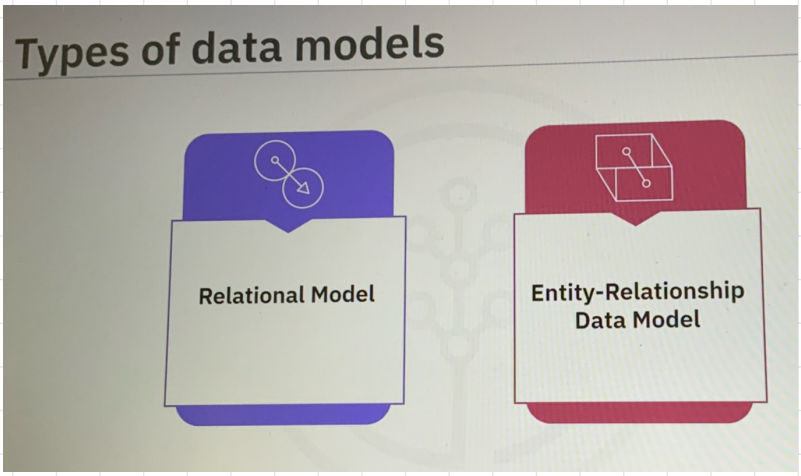


Data model: Aspects

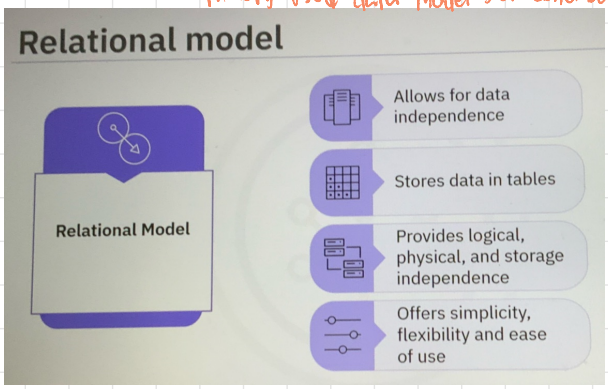


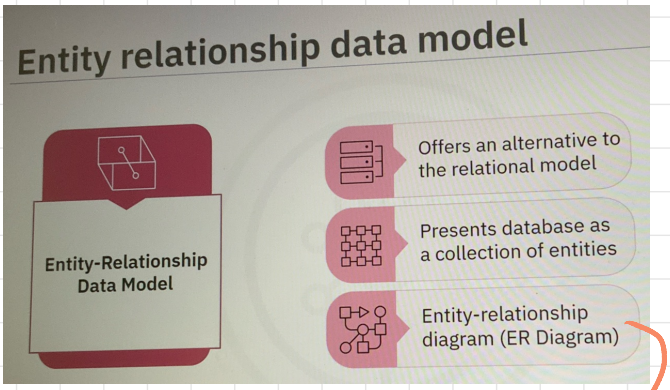
Differences: Information and data models

Differences	Information Models	Data Models
Level of Detail	Less detailed	Specifies storage and manipulation
Purpose	Understand business concepts	Technical implementation for storage and querying
Usage	Business analysis and agreement	Database system design and development
System Development	Information model first	The data model created later

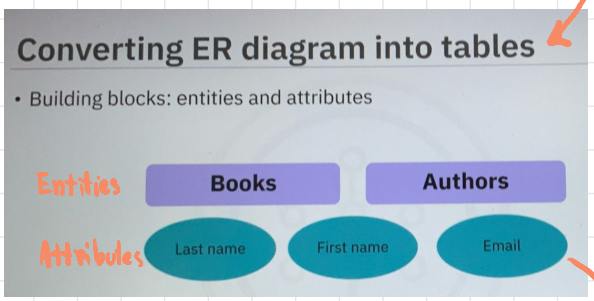


widely used data model for databases





convert

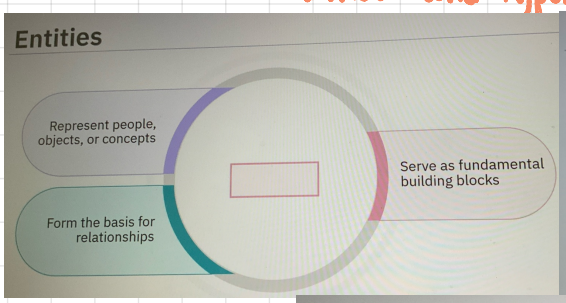


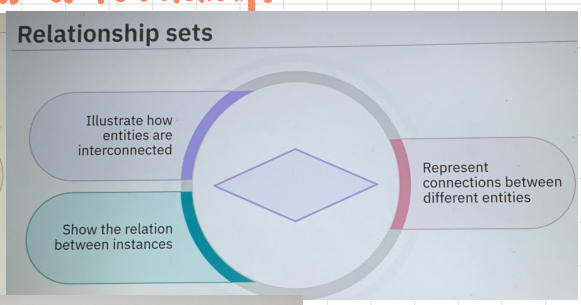
Entities become to table

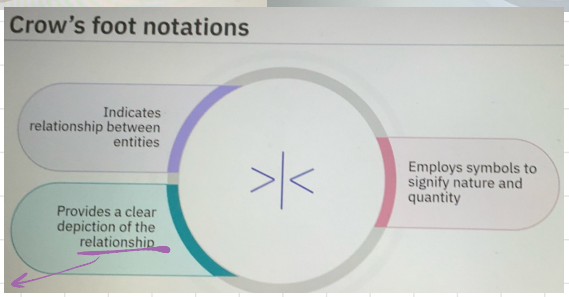
Att → column

Extend this

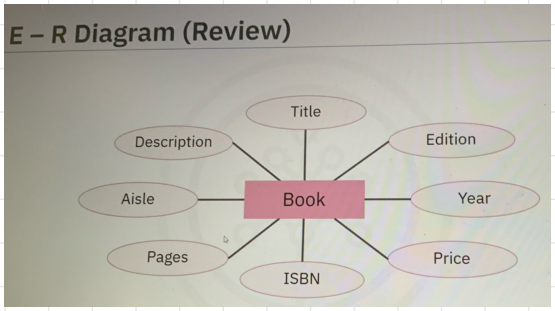
ERDs and Types of Relationships

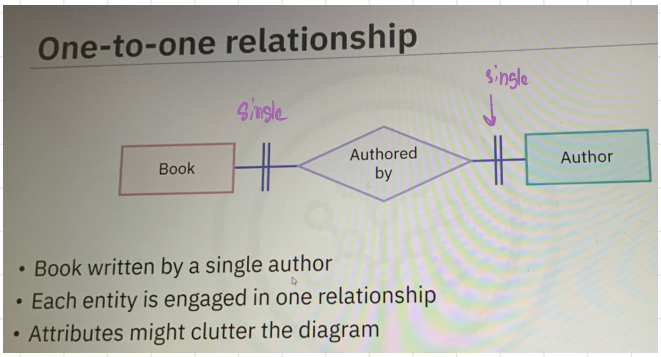


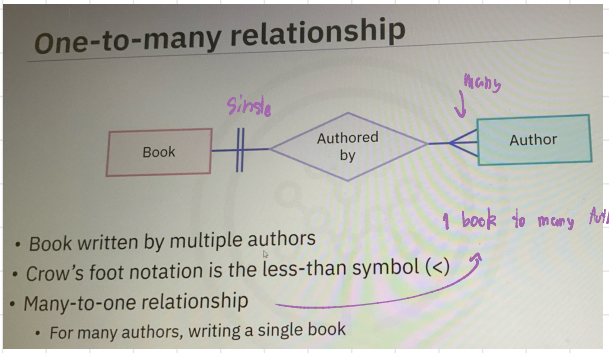


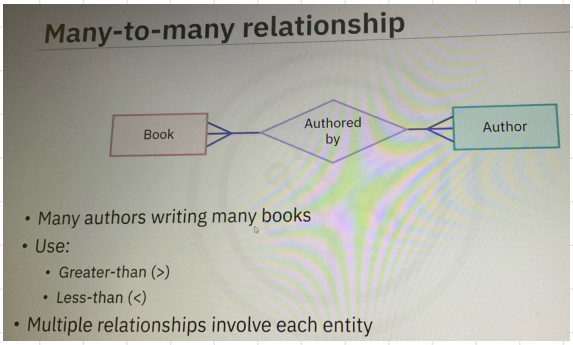


one-to-one, one-to-many, many-to-many

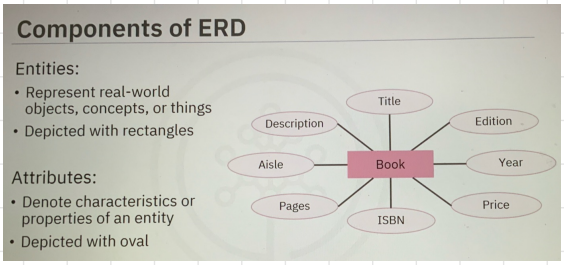


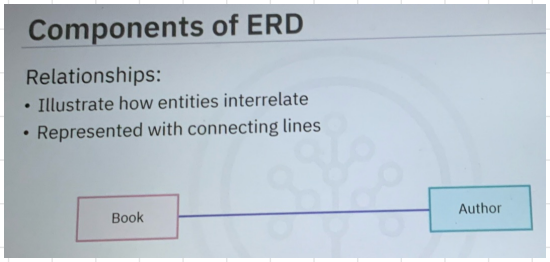


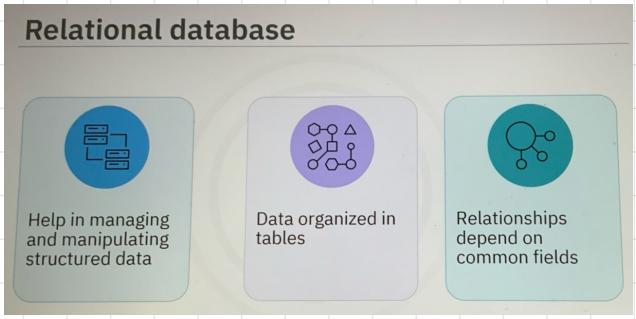


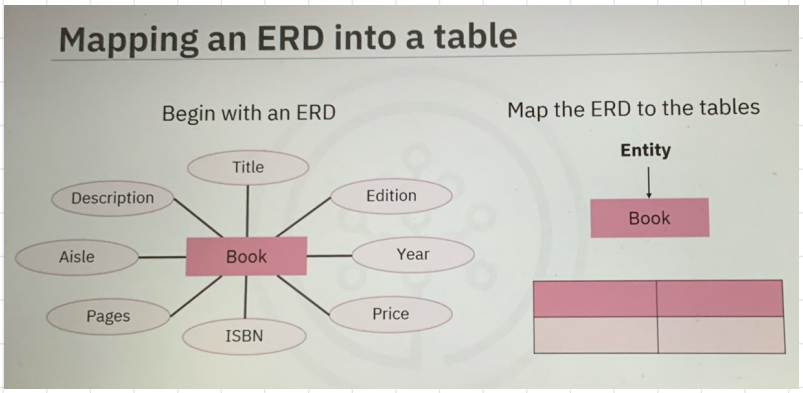


Mapping Entities to Table









Example: Entity Author

Author_ID	Last name	First name	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	CA
A4	Shame	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	ip@univ.com	Transilvania	RO

Best practices

- Primary key designation
- Data validation
- Default values
- Using views
- Concurrency control

Primary key designation:
- Uniquely identify each book in the database
- Example: BookID

Data validation:
- Ensure accuracy and consistency of data
- Involves checks for data types, ranges, and formats
- Example: Published year only accepts numerical values

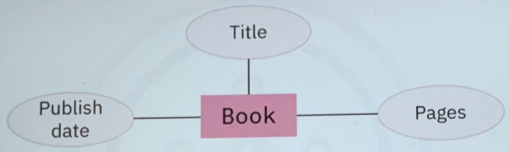
Default values:
- Streamline data entry
- Enhance records
- Example: Setting the default value to "Unknown"

Use of views:
- Present a customized and simplified perspective of data
- Example: View of Book and Author together

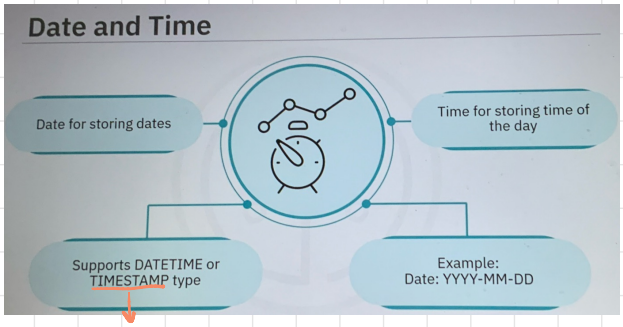
Concurrency control:
- Manage access and modifications to the database
- Prevents data inconsistency and conflicts
- For example, include a "LastModified" timestamp

Data Type

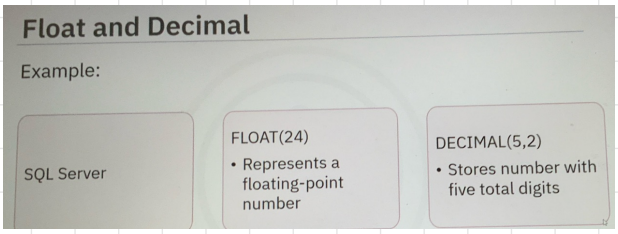
Database table

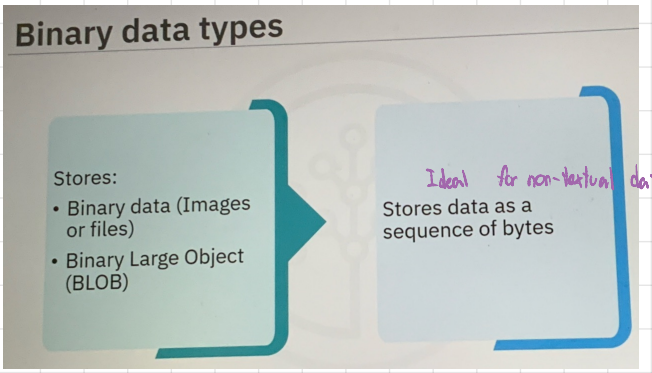


Title	Publish date	Pages
Lean Software Development	2003-05-18	240
SQL Cookbook	2005-12-01	595
Scientific Computing	2001-07-17	576



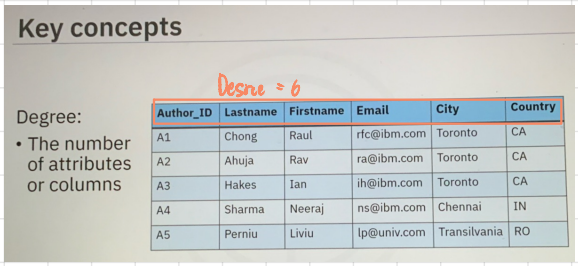
include Date and time





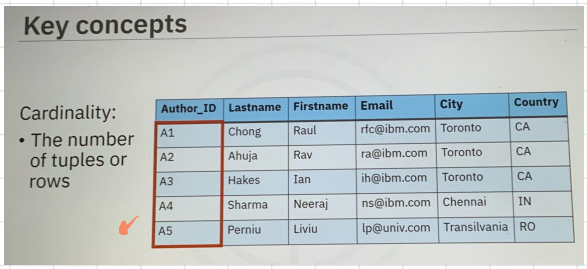
Ideal for non-textual data

Relational Data



Degree = 6

Author_ID	Lastname	Firstname	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	CA
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@univ.com	Transilvania	RO



Author_ID	Lastname	Firstname	Email	City	Country
A1	Chong	Raul	rfc@ibm.com	Toronto	CA
A2	Ahuja	Rav	ra@ibm.com	Toronto	CA
A3	Hakes	Ian	ih@ibm.com	Toronto	CA
A4	Sharma	Neeraj	ns@ibm.com	Chennai	IN
A5	Perniu	Liviu	lp@univ.com	Transilvania	RO

Cardinality = 5

Database Architecture

Deployment topology

Arrangement or configuration of:
- Hardware
- Software
- Network components

Depends on factors:
- Scalability
- Performance
- Reliability
- Nature of the application

Common deployment topologies

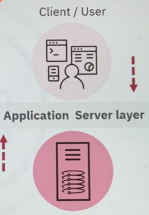
- Single-tier architecture
- Client-server architecture or two-tier database architecture
- Three-tier architecture
- Cloud-based

Single-tier architecture

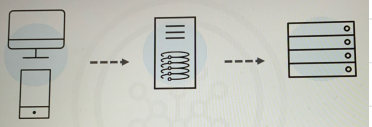
- Deploys all components on a single server
- Operates within the same environment

Client-server architecture

- Also referred to as two-tier architecture
- Divides the application into two distinct layers
  - Client layer: Responsible for user interface
  - Server layer: Manages the application logic
- The remote server hosts the database
- Users access it from client systems



Three-tier architecture



- Introduces middle tier between client and server

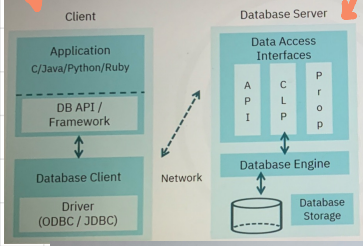
Three-tier architecture emergence

Security concern:
- Prevent unauthorized data access or modification

Performance:
- Avoid overloading the server with unnecessary traffic

Maintainability:
- Avoid making changes to data except for administrators

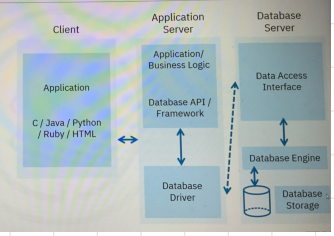
Two-tier database overview



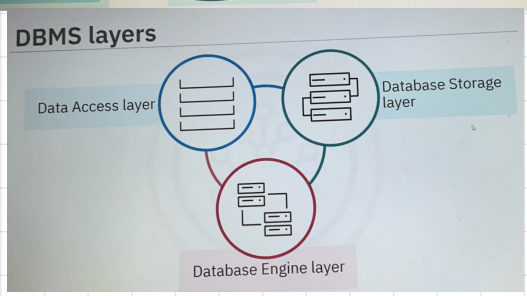
- Server and application operate in separate tiers
- Application connects using language-dependent interface
- Interface communicates with the server through API

Three-tier database architecture

- Application and business layer in separate tiers
- Presentation layer:
  - Serves as UI
  - Accessible through various platforms
- Application server:
  - Client connects over the network
  - Encapsulates the application and business logic
  - Communicates with database server through API/driver



DBMS layers



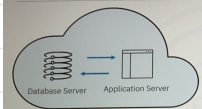
?

Cloud deployment

- Involves residing the database within a cloud
- Offers many advantages inherent to cloud-based services
- Eliminates the need to download database software locally

ongoing ma

Cloud deployment overview



- Are easily accessible to users with internet
- Involves communication within the cloud environment
- Suitable for:
  - Development
  - Testing
  - Full-scale production environments

Communication in database interface

Database Interface:
- Communicates via Database Client or API

Data Access Layer:
- Server includes various client interfaces
- Examples: JDBC, ODBC, CLP, vendor-specific interfaces

Database Server Engine:
- Compiles queries
- Retrieves, processes data, returns result set

Distributed Architecture and Clustered Databases

Distributed architectures

- Helpful for large-scale workloads
- Support high availability or scalability
- Involve clusters of machines interconnected through a network

Types of database architecture

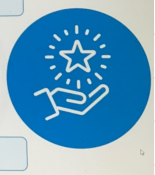
- Shared disk architecture
- Shared nothing architecture
- Combination and specialized architectures

Shared disk architecture

- Multiple database servers processing workloads concurrently
- Server establishes a connection to shared storage
- High-speed communication between servers

Shared disk architecture: Advantages

- Ensures effective workload distribution
- Ensures scalability as demand grows
- Reroutes clients during server failures
- Maintains high availability



Shared nothing architecture

- Utilizes either replication or partitioning techniques
- Allows distribution of client workloads
- Promotes parallel processing
- Enhances fault tolerance by rerouting clients

Combination and specialized architectures

- Employ a combination of:
  - Shared disk
  - Shared nothing
  - Replication or partitioning techniques
- Integrate specialized hardware components

Common techniques

- Help in managing data and optimizing performance
- Common techniques include:

Database replication

Database partitioning and sharding

Database replication

- Involves copying changes to replicas
- Distributes the client workload across servers

Replica resides in the same location, we call it a High Availability (HA) replica

Database partitioning and sharding

- Involves partitioning tables with substantial data
- Each shard possesses its compute resources
- Increased parallel processing and improved overall performance