# Database Introduction

COSC 304 – Introduction to Database Systems

# What is a database?

A ***database*** is a collection of logically related data for a particular domain.

A ***database management system*** (***DBMS***) is software designed for the creation and management of databases.

- e.g. Oracle, DB2, Microsoft Access, MySQL, SQL Server, MongoDB

Bottom line: A ***database*** is the *data* stored and a ***database system*** is the *software* that manages the data.

# Databases in the Real-World

Databases are everywhere in the real-world even though you do not often interact with them directly.

- $50 billion dollar annual industry

Examples:

- Retailers manage their products and sales using a database.
  - Wal-Mart has one of the largest databases in the world!
- Online web sites such as Amazon, Apple, and Expedia track orders, shipments, and customers using databases.
- The university maintains all your registration information and marks in a database that is accessible over the Internet.

Can you think of other examples?

What data do you have?

# Developing without a Database

Without a database, applications use files to store data *persistently*. A ***file-based system*** has several problems: code and data duplication, high maintenance costs, and difficulty in supporting multiple users.

- There is no ***program-data independence*** separating the application from the data it is manipulating. If the data file changes, the code must be changed.

Example: You have been asked to build a simple inventory/sales program for a store. What challenges would you face without a DBMS?

# Data Independence and Abstraction

Databases provide ***data abstraction*** allowing the internal representation of the data to change without affecting programs that use the object through an external definition.

- The DBMS takes the description of the data and handles the low-level details of how to store it, retrieve it, and provide concurrent access to it.

# Database System Properties

A database system provides *efficient*, *convenient*, and *safe multi-user* storage and access to *massive* amounts of *persistent* data.

***Efficient*** - Able to handle large data sets and complex queries without searching all files and data items.

***Convenient*** - Easy to write queries to retrieve data.

***Safe*** - Protects data from system failures and hackers.

***Massive*** - Database sizes in gigabytes, terabytes and petabytes.

***Persistent*** - Data exists even if have a power failure.

***Multi-user*** - More than one user can access and update data at the same time while preserving consistency.

# Database Terminology

A ***data model*** is a collection of concepts that is used to describe the structure of a database. E.g. relational, XML, graphs, objects, JSON

- In the relational model, data is represented as tables and fields.

***Data Definition Language*** (***DDL***) allows the user to create data structures in the data model used by the database. A ***schema*** is a description of the structure of the database and is maintained and stored in the ***system catalog***. The schema is ***metadata***.

- A schema contains structures, names, and types of data stored.

Once a database has been created using DDL, the user accesses data using a ***Data Manipulation Language*** (***DML***).

- The DML allows for the insertion, modification, retrieval, and deletion of data.

SQL is a standard DDL and DML for the relational model.

# SQL DDL and DML Examples

Create a table to store data on products:

```
CREATE TABLE product(
    sku as VARCHAR(10),
    name as VARCHAR(40),
    desc as VARCHAR(50),
    inventory as INTEGER,
    PRIMARY KEY (sku));
```

Insert product into database:

```
INSERT INTO product VALUES ('1234','Soap','Ivory',100);
```

Retrieve all products where inventory < 10:

```
SELECT name, inventory FROM product WHERE inventory < 10;
```

# Database Properties Question

***Question:*** **True or False**: The data in a database is lost when the power to the computer is turned off.

**A)** true

**B)** false

# Database Abstraction Question

***Question:*** Defining how data is stored using DDL is similar to what in object-oriented programming?

**A)** Objects

**B)** Classes

**C)** Inheritance

**D)** Polymorphism

# DDL vs. DML Question

***Question:*** If you are querying data in a database, which language are you using:

**A)** DML

**B)** DDL

**C)** schemas

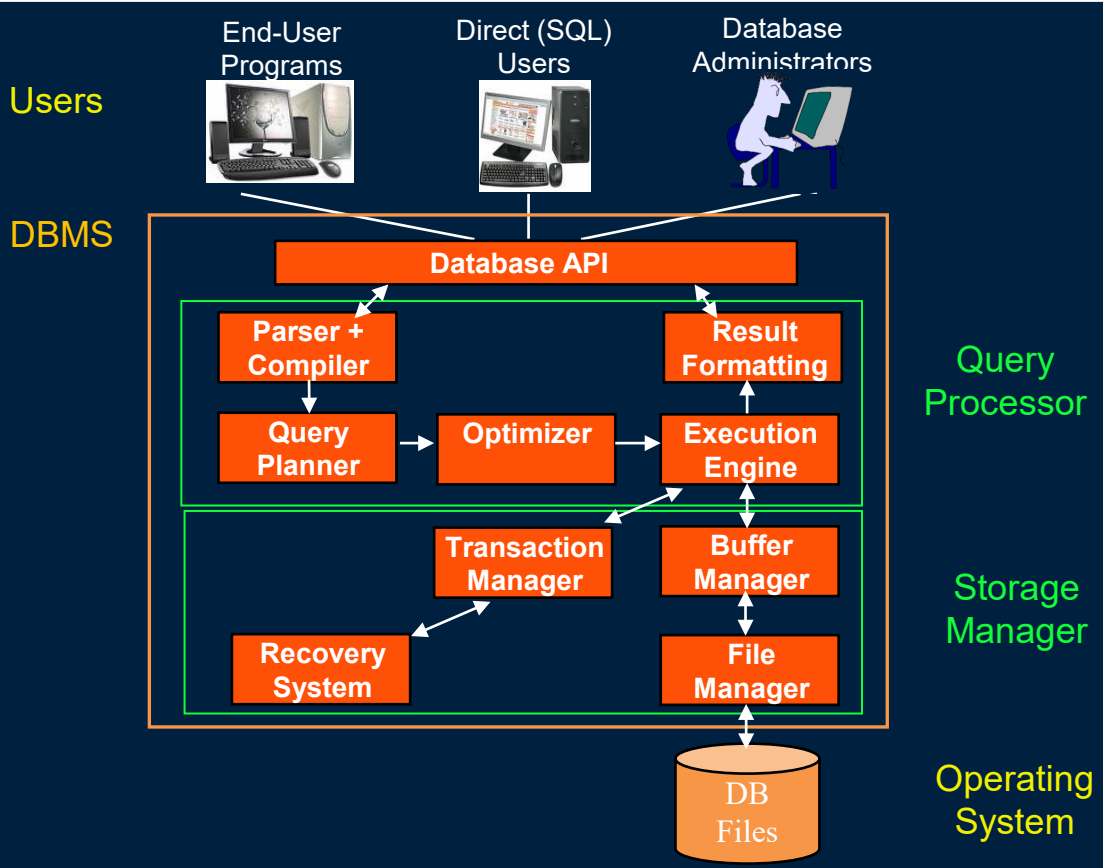
**D)** Java

# Components of a DBMS

A DBMS is a complicated software system containing many components:

- ***Query processor*** - translates user/application queries into low-level data manipulation actions.
  - Sub-components: query parser, query optimizer
- ***Storage manager*** - maintains storage information including memory allocation, buffer management, and file storage.
  - Sub-components: buffer manager, file manager
- ***Transaction manager*** - performs scheduling of operations and implements concurrency control algorithms.

# DBMS Architecture

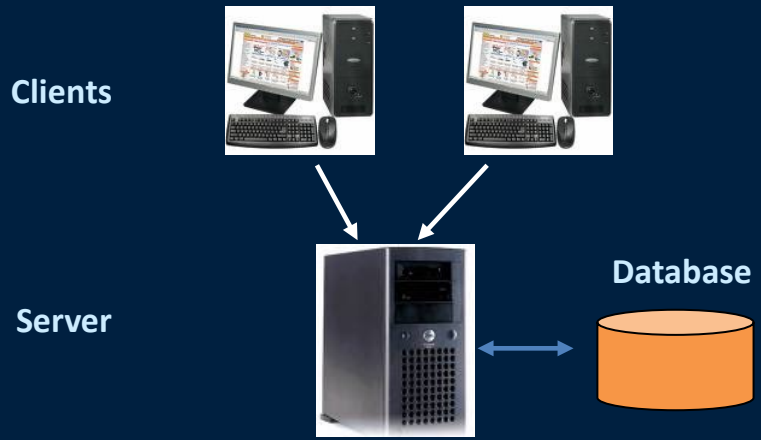


# Database Architectures

There are several different database architectures:

- **File-server (embedded) architecture** - files are shared but DBMS processing occurs at the clients (e.g. Microsoft Access or SQLite)
- **Two-Tier client-server architecture** - dedicated machine running DBMS accessed by clients (e.g. SQL Server)
- **Three-Tier client-server architecture** - DBMS is bottom tier, second tier is an application server containing business logic, top tier is clients (e.g. Web browser-Apache/Tomcat-Oracle)

# Two-Tier Client-Server Architecture

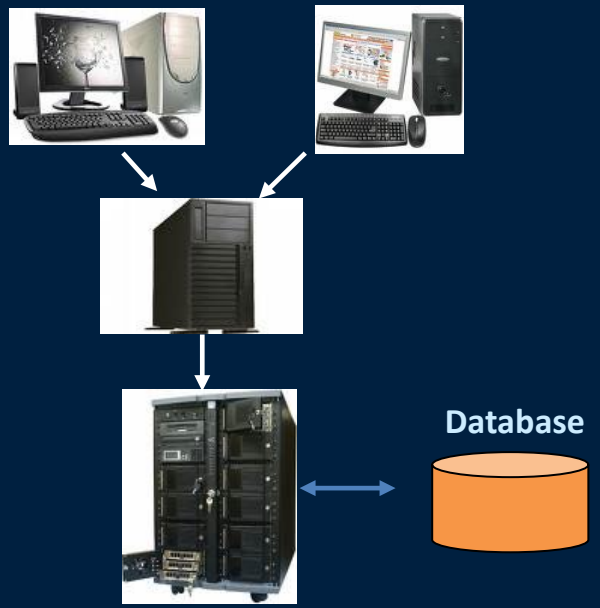



Tier 1: Client
- User Interface
- Business and Data Logic

Tier 2: Database Server
- Server-side validation
- Data storage/management

Advantages:
- Only one copy of DBMS software on dedicated machine.
- Increased performance.
- Reduced hardware and communication costs.
- Easier to maintain consistency and manage concurrency.

# Three-Tier Client-Server Architecture




Tier 1: Client (Web/mobile)
- User Interface

Tier 2: Application Server
- Business logic
- Data processing logic

Tier 3: Database Server
- Data validation
- Data storage/management

Advantages:
- Reduced client administration and cost using thin web clients.
- Easy to scale architecture and perform load balancing.

# Database People

There are several different types of database personnel:

- ***Database administrator*** (***DBA***) - responsible for installing, maintaining, and configuring the DBMS software.
- ***Data administrator*** (***DA***) - responsible for organizational policies on data creation, security, and planning.
- ***Database designer*** - defines and implements a schema for a database and associated applications.
  - ***Logical/Conceptual database designer*** - interacts with users to determine data requirements, constraints, and business rules.
  - ***Physical database designer*** - implements the logical design for a data model on a DBMS. Defines indexes, security, and constraints.
- ***DBMS developer*** - writes the DBMS software code.
- ***Application developer*** - writes code that uses the DBMS.
- ***User*** - uses the database directly or through applications.

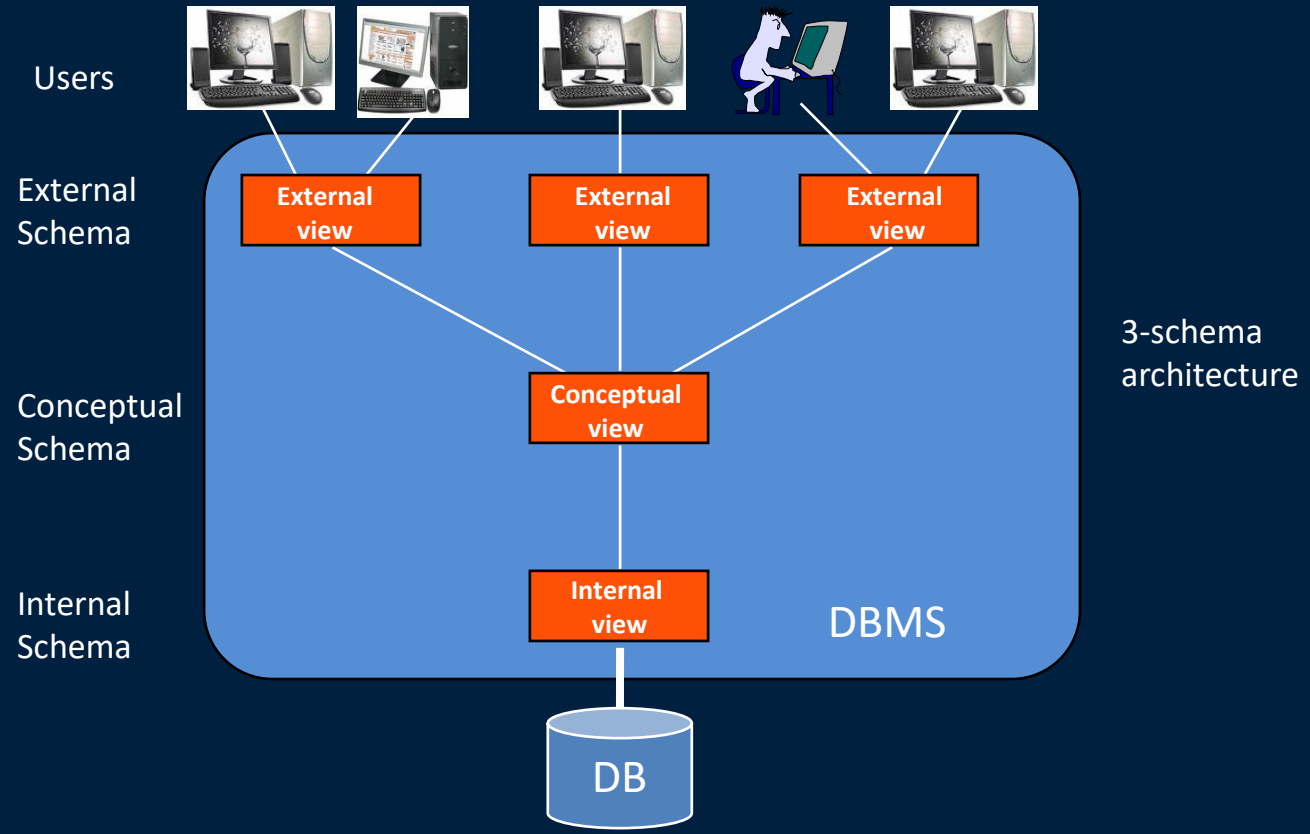
# ANSI/SPARC Architecture

One of the major advantages of database systems is data abstraction. Data abstraction is achieved by defining different views of the data. Each view isolates higher-level views from data representation details.

The ANSI/SPARC architecture consists of three views:

- ***Internal View*** - The physical representation of the database on the computer. *How* the data is stored.
- ***Conceptual View*** - The logical structure of the database that describes *what* data is stored and its relationships.
- ***External View*** - The *user's view* of the database that provides the part of the database relevant to the user.

# ANSI/SPARC Architecture



# Benefits of Three Schema Architecture

External Level:

- Each user can access the data, but have their own view of the data independent of other users.
  - *Logical data independence* - conceptual schema changes do not affect external views.

Conceptual Level:

- Single shared data representation for all applications and users which is independent of physical data storage.
  - Users do not have to understand physical data representation details.
  - The DBA can change the storage structures without affecting users or applications. *Physical data independence* - conceptual schema not affected by physical changes such as adding indexes or distributing data.

Internal (Physical) Level:

- Provides standard facilities for interacting with operating system for space allocation and file manipulation.

# Microsoft Access and the Three Schema Architecture

External Level:

- Microsoft Access does not call them views, but you can store queries and use the results in other queries (like a view).
  - External schema is the query (view) name and the attribute metadata.

Conceptual Level:

- All tables and field definitions are in the schema (accessible from the Tables tab).
  - Note that conceptual **schema** is not the data but the metadata.

Physical Level:

- Access represents all data in a single file whose layout it controls.
- The system processes this raw data file by knowing locations and offsets of relations and fields.

# ANSI/SPARC Architecture - Three Levels of Views

***Question:*** What are the three levels of views in the ANSI/SPARC architecture starting with the view closest to the user?

**A)** Internal, Conceptual, External

**B)** External, Internal, Conceptual

**C)** Internal, External, Conceptual

**D)** External, Conceptual, Internal

**E)** User, Logical, System

# ANSI/SPARC Architecture - Abstraction with Views

***Question:*** Assume you have a Java program accessing data stored in a file. Select ***one*** true statement.

**A)** The file organization is changed. The internal view is where this change is made.

**B)** A field is added to the database. The conceptual view is changed.

**C)** A user account has restricted access to the file. The external view must be changed.

**D)** More than one of the above

# Conclusion

A ***database*** is a collection of logically related data managed by a ***database management system*** (DBMS).

- Provides data independence and abstraction
- Data definition and manipulation languages (DDL and DML)

A DBMS has advantages over traditional file systems by supporting data independence and standard implementations for data management tasks.

The three schema architecture consists of external, conceptual, and internal schemas. Each view provides data abstraction and isolates the layer above from certain data manipulation details.

# Objectives

- Define: database, DBMS, schema, metadata
- Describe the features of a file-based system and some limitations inherent in that architecture.
- Define program-data independence/data abstraction and explain how it is achieved by databases but not by file systems.
- Define DDL and DML.  What is the difference?
- List some modules of a DBMS.
- List different people associated with a DBMS and their roles.
- Explain how a schema differs from data.
- Draw a diagram of the three schema architecture and explain what each level provides.  List benefits of the architecture.
- How does a schema provide data independence?
- Compare/contrast two-tier and three-tier architectures.


UBC
THE UNIVERSITY OF BRITISH COLUMBIA