Australian
National
University

Introduction to Database Systems – Part I

General Concepts

What is a Database?

Assignment Project Exam Help

https://powcoder.com

- Have you designed a database?

Add WeChat powcoder

- Have you worked with a database?

Definition of Databases

- A **database** is a collection of **related data**.

- Implicit properties:

  - represents some **aspects of the real world**;

  - a **logically coherent collection** of data;

  - designed and built for a **specific purpose**.

Examples (Huge):

Amazon: – It has 244 million active customers, over 60 million items occupying many terabytes of data (clothing, sports, videos, office products).

YouTube: – Over 1.3 billion users, 300 hours of videos added every minute, average of one billion mobile YouTube views per day

What is a Database Management System?

- A **database management system** (DBMS) is a collection of programs that enable users to create and maintain a database.

- It is a general-purpose software system that facilitates the process of
  - **defining:** specifying data types, structures and constraints;
  - **constructing:** storing data on some storage medium;
  - **manipulating:** retrieving and manipulating data;
  - **sharing:** using data by multiple users/programs simultaneously.

- Well-known relational DBMSs include Oracle, IBM DB2, Microsoft's Access, Microsoft's SQL Server, MySQL, postgreSQL, etc.
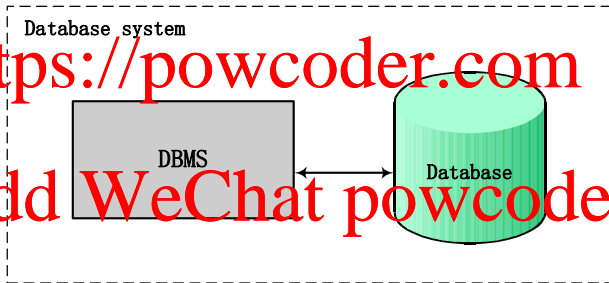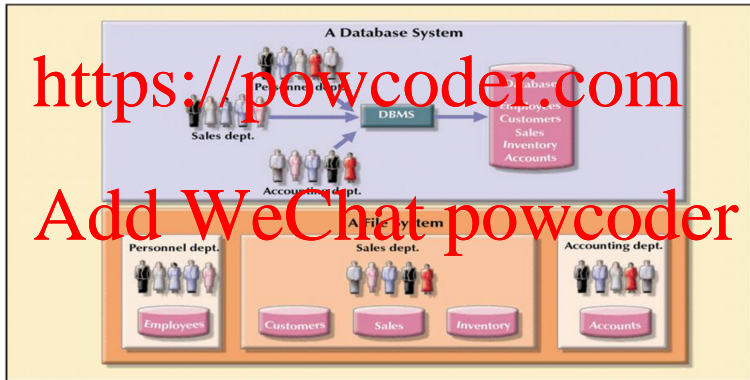
What is a Database System?

- A **database system** is part of information systems dealing with data retrieval and manipulation.
- It often refers to a DBMS plus a database.

- Main services a database system provides:
  - answer queries efficiently;
  - execute updates efficiently.

Why is a Database System Needed?

- Database system: an integrated collection of logically related data
- File system: many separate and unrelated files

Database Systems: Design, Implementation, & Management, 6th Edition, Rob & Coronel
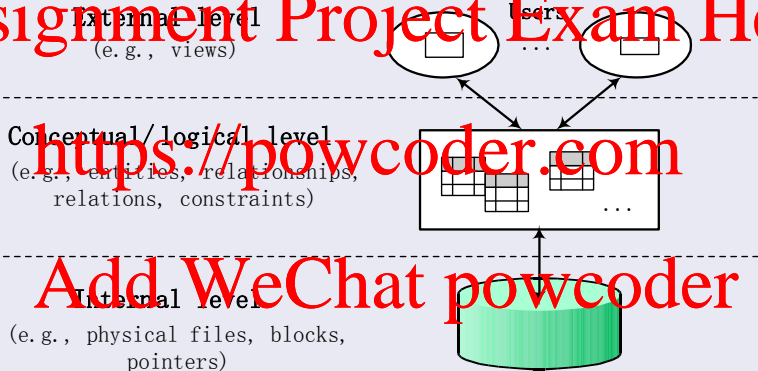
Why is a Database System Needed?

- Advantages of using a database system:

  - **Data redundancy:** Data redundancy is controlled to ensure consistency and save the storage space.

  - **Data integrity:** Some integrity constraints can be enforced automatically by the DBMS.

  - **Data security:** Since the data is managed centrally, the DBMS ensures that the database access is through an authorized channel.

In addition to the above, the database system also facilitates the following:

Concurrent transactions; backup and recovery services; data independence; etc.

Three-level ANSI/SPARC Architecture

External level
(e.g., views)

Conceptual/logical level
(e.g., entities, relationships,
relations, constraints)

Internal level
(e.g., physical files, blocks,
pointers)

- **Note:** schemas at the three levels are *descriptions* of data; the stored data *actually* exists at the internal level (i.e., physical level) only.

Three-level ANSI/SPARC Architecture

- **External Schema**
  - perspective of the user / application
  - describes restructured parts of the database used in applications

- **Conceptual (Logical) Schema**
  - perspective of a community of users
  - describes what data is stored in the database and relationships among data (independent from their physical storage structures)

- **Internal Schema**
  - perspective of the implementation / system realization
  - describes how data is stored in the database (e.g., physical storage structures).

Derived Principles – Data Independence

- **Logical data independence**: change the conceptual/logical schemas without having to change external schemas or application programs

  Example: If adding or removing entities, external schemas that refer only to the remaining data should not be affected.

- **Physical data independence**: change the internal schemas without having to change the conceptual/logical schemas

  Example: If physical files were reorganised, we should not have to change the conceptual/logical schemas.
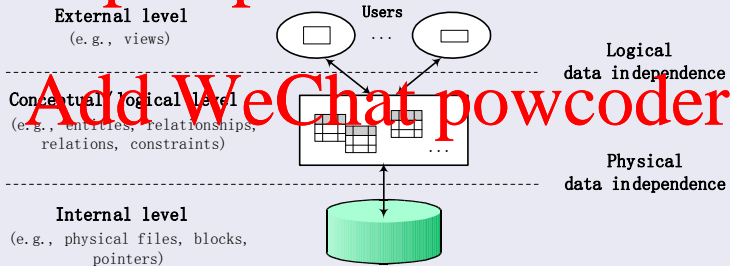
- **Key idea:** When the schema is changed at some level,
  - the schema at the next higher level remains unchanged;
  - only the *mapping* between two levels is changed.

External level
  (e.g., views)

Users
  ...

Logical
data independence

Conceptual level
(e.g., entities, relationships,
  relations, constraints)

  ...

Physical
data independence

Internal level
(e.g., physical files, blocks,
      pointers)

Historical Remarks I/II

- **Hierarchical Databases**

  - Oldest data model (1960s);

  - SABRE, a collaboration between IBM and American Airlines;

- **Network Databases**

  - Extension of hierarchical databases, from tree to network (late 1960s);

- **Relational Databases**

  - Edgar F. Codd,
    A Relational Model of Data for Large Shared Data Banks

  - System R and SQL

Historical Remarks II/II

- **Object-Oriented Databases**

  - Driven by object-oriented programming languages (1980s);

  - Designed to store and share complex, structured objects.

- **XML Databases**

  - XML is emerged as the standard for Web data exchange (1990s);

  - Suitable to sparse data, deeply nested data and mixed content.

- **NoSQL Databases**

  - Recent development in industry (since 2009);

  - We will discuss NoSQL databases at the end of this course.