# DATABASE MANAGEMENT SYSTEM

PRESENTED BY
V ELAVARASAN
INFORMATION TECHNOLOGY

# DATABASE MANAGEMENT SYSTEM (DBMS)

- DBMS contains information about a particular enterprise
  - Collection of interrelated data
  - Set of programs to access the data
  - An environment that is both *convenient* and *efficient* to use
- Databases can be very large.
- Databases touch all aspects of our lives

# DATABASE APPLICATIONS

- Database Applications:
  - Banking: transactions
  - Airlines: reservations, schedules
  - Universities:  registration, grades
  - Sales: customers, products, purchases
  - Online retailers: order tracking, customized recommendations
  - Manufacturing: production, inventory, orders, supply chain
  - Human resources: employee records, salaries, tax deductions

# DRAWBACKS OF USING FILE SYSTEMS TO STORE DATA

- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become “buried” in program code rather than

    being stated explicitly
  - Hard to add new constraints or change existing ones

# DRAWBACKS OF USING FILE SYSTEMS TO STORE DATA (CONT.)

- Atomicity of updates
  - Failures may leave database in an inconsistent state with partial updates carried out
  - Example:Transfer of funds from one account to another should either complete or not happen at all
- Concurrent access by multiple users
  - Concurrent access needed for performance
  - Uncontrolled concurrent accesses can lead to inconsistencies

# LEVELS OF ABSTRACTION

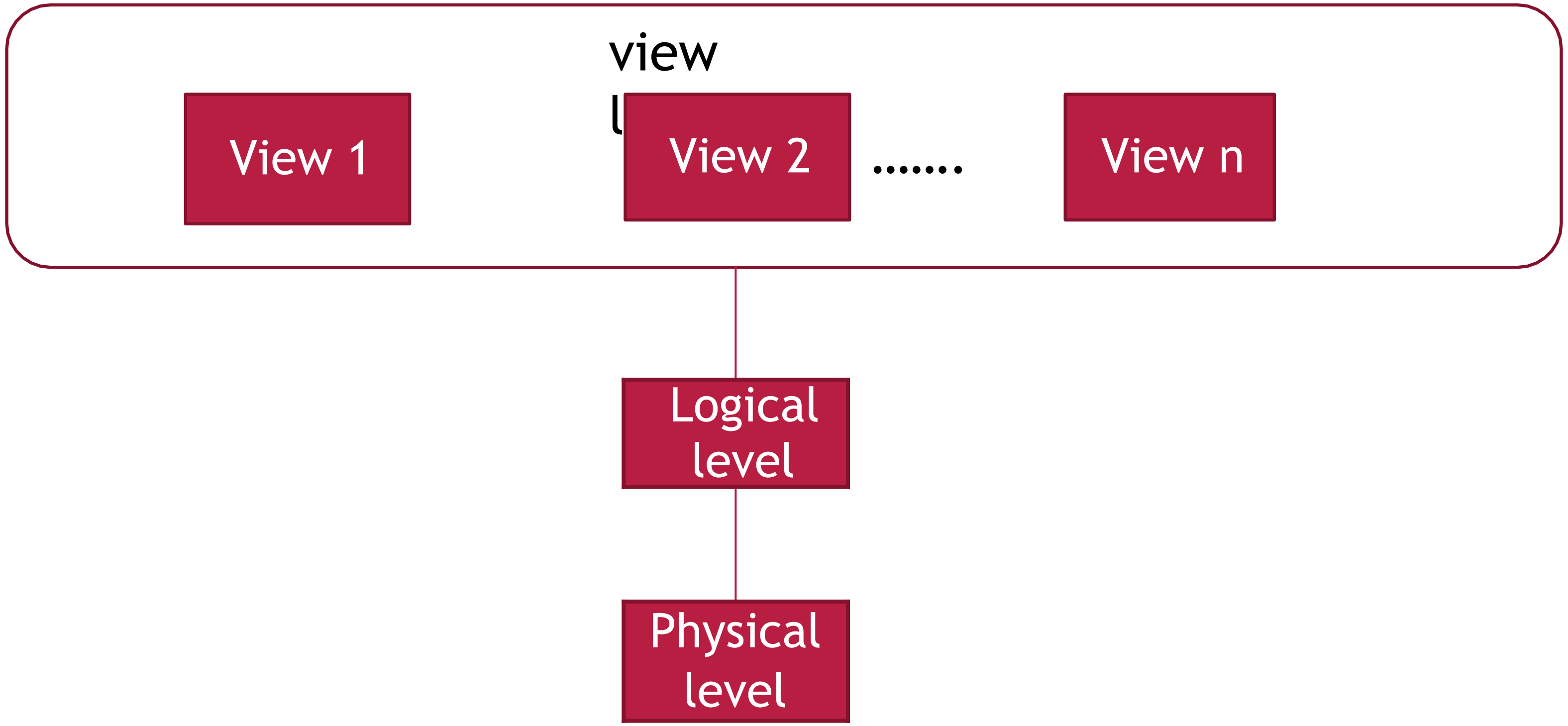
- **Physical level:** describes how a record (e.g., instructor) is stored.
- **Logical level:** describes data stored in database, and the relationships among the data.

  **type** *instructor* = **record**

  *ID* : string;
  *name* : string; *deptName* : string; *salary* : integer;

  end;
- **View level:** application programs hide details of data types. Views can also hide information (such as an employee's salary) for security purposes.

# ARCHITECTURE FOR A DATABASE SYSTEM



# DATA MODELS

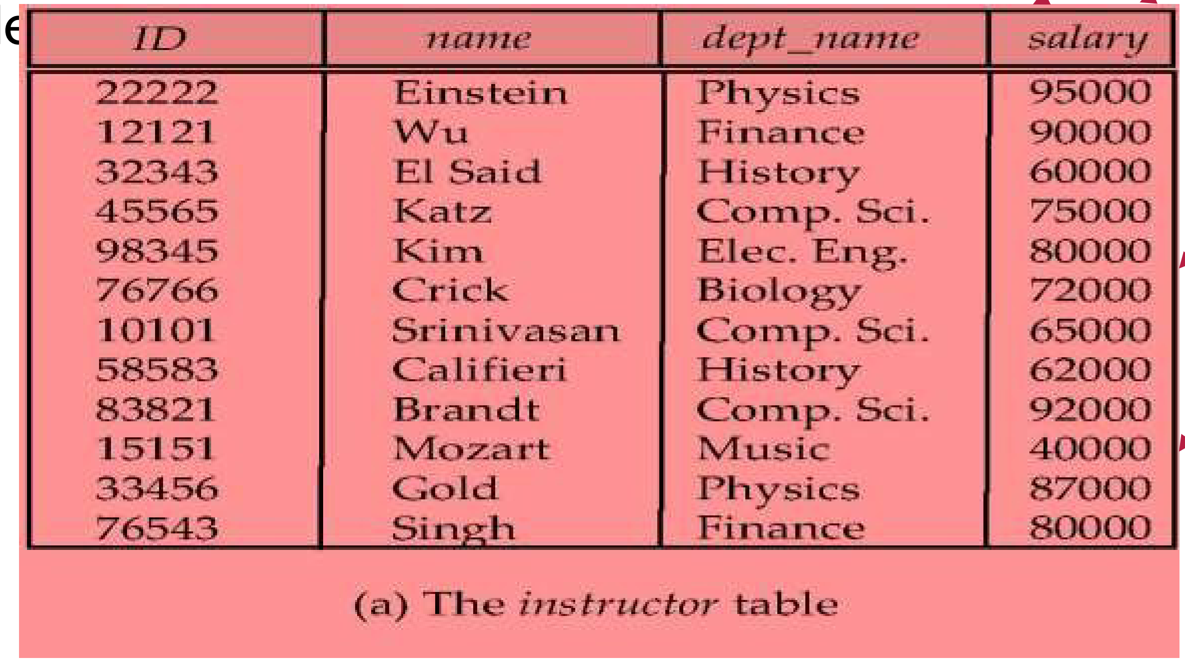
- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- Relational model
- Entity-Relationship data model (mainly for database design)
- Object-based data models (Object-oriented and Object-relational)
- Semistructured data model (XML)

# RELATIONAL MODEL

- All the data is stored in various tables.
- Example of tabular data in the relational mode
-

columns

rows



| *ID* | *name* | *dept_name* | *salary* |
|---|---|---|---|
| 22222 | Einstein | Physics | 95000 |
| 12121 | Wu | Finance | 90000 |
| 32343 | El Said | History | 60000 |
| 45565 | Katz | Comp. Sci. | 75000 |
| 98345 | Kim | Elec. Eng. | 80000 |
| 76766 | Crick | Biology | 72000 |
| 10101 | Srinivasan | Comp. Sci. | 65000 |
| 58583 | Califieri | History | 62000 |
| 83821 | Brandt | Comp. Sci. | 92000 |
| 15151 | Mozart | Music | 40000 |
| 33456 | Gold | Physics | 87000 |
| 76543 | Singh | Finance | 80000 |

(a) The *instructor* table

# DATA DEFINITION LANGUAGE (DDL)

- Specification notation for defining the database schema Example:

```
create table instructor (
ID              char(5),
nam            varchar(20)
edept_name ,  varchar(20),
salary     numeric(8,2))
```

- DDL compiler generates a set of table templates stored in a ***data dictionary***

# DATA DEFINITION LANGUAGE (DDL)

- Data dictionary contains metadata (i.e., data about data)
    - Database schema
    - Integrity constraints
        - Primary key (ID uniquely identifies instructors)
    - Authorization
        - Who can access what

# DATA MANIPULATION LANGUAGE (DML)

- Language for accessing and manipulating the data organized by the appropriate data model
  - DML also known as query language
- Two classes of languages
  - **Pure** – used for proving properties about computational power and for optimization
    - Relational Algebra
    - Tuple relational calculus
    - Domain relational calculus
  - **Commercial** – used in commercial systems
    - SQL is the most widely used commercial language

# SQL

- The most widely used commercial language
- SQL is NOT a Turing machine equivalent language
- SQL is NOT a Turing machine equivalent language
- To be able to compute complex functions SQL is usually embedded in some higher-level language
- Application programs generally access databases through one of
  - Language extensions to allow embedded SQL
  - Application program interface (e.g., ODBC/JDBC) which allow SQL queries to be sent to a
    database

# DATABASE DESIGN

- The process of designing the general structure of the database:
- Logical Design – Deciding on the database schema. Database design requires that we find a "good" collection of relation schemas.
  - Business decision – What attributes should we record in the database?
  - Computer Science decision – What relation schemas should we have and how should the attributes be distributed among the various relation schemas?
- Physical Design – Deciding on the physical layout of the database

# OBJECT-RELATIONAL DATA MODELS

- Relational model: flat,“atomic” values
- Object Relational Data Models
  - Extend the relational data model by including object orientation and constructs to deal with added data types.
  - Allow attributes of tuples to have complex types, including non-atomic values such as nested relations.
  - Preserve relational foundations, in particular the declarative access to data, while extending modeling power.
  - Provide upward compatibility with existing relational languages.

# XML: EXTENSIBLE MARKUP LANGUAGE

- Defined by the WWW Consortium (W3C)
- Originally intended as a document markup language not a database language
- The ability to specify new tags, and to create nested tag structures made XML a great

  way to exchange **data**, not just documents
- XML has become the basis for all new generation data interchange formats.
- A wide variety of tools is available for parsing, browsing and querying XML documents/data