Data Models and Database Languages

by

Dr. Pratik Roy

Database Systems

- 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
- Database systems are used to manage collections of data that are:
  - Highly valuable
  - Relatively large
  - Accessed by multiple users and applications, often at the same time.
- A modern database system is a complex software system whose task is to manage a large, complex collection of data.

Database Applications Examples

- **Enterprise Information**
  - **Sales:** customers, products, purchases
  - **Accounting:** payments, receipts, assets
  - **Human Resources:** Information about employees, salaries, payroll taxes.
- **Manufacturing:** For management of the supply chain and for tracking production of items in factories, inventories of items in warehouses and stores, and orders for items.
- **Banking and Finance**
  - customer information, accounts, loans, and banking transactions.
  - Credit card transactions
  - **Finance:** sales and purchases of financial instruments (e.g., stocks and bonds; storing real-time market data
- **Universities:** For student information, course registrations, and grades.
- **Airlines:** For reservations and schedule information.
- **Telecommunication:** records of calls, texts, and data usage, generating monthly bills, maintaining balances on prepaid calling cards.

Database Applications Examples

- **Web-based services**
  - **Social-media:** For keeping records of users, connections between users (such as friend/follows information), posts made by users, rating/like information about posts, etc.
  - **Online retailers:** order tracking, customized recommendations
  - **Online advertisements**
- **Document databases:** For maintaining collections of new articles, patents, published research papers, etc.
- **Navigation systems:** For maintaining the locations of varies places of interest along with the exact routes of roads, train systems, buses, etc.

View of Data

- ❑A database system is a collection of interrelated data and a set of programs that allow users to access and modify these data.
- ❑A major purpose of a database system is to provide users with an abstract view of the data. That is, the system hides certain details of how the data are stored and maintained.
- ❑**Data models**
  - A collection of conceptual tools for describing data, data relationships, data semantics, and consistency constraints.

Data Models

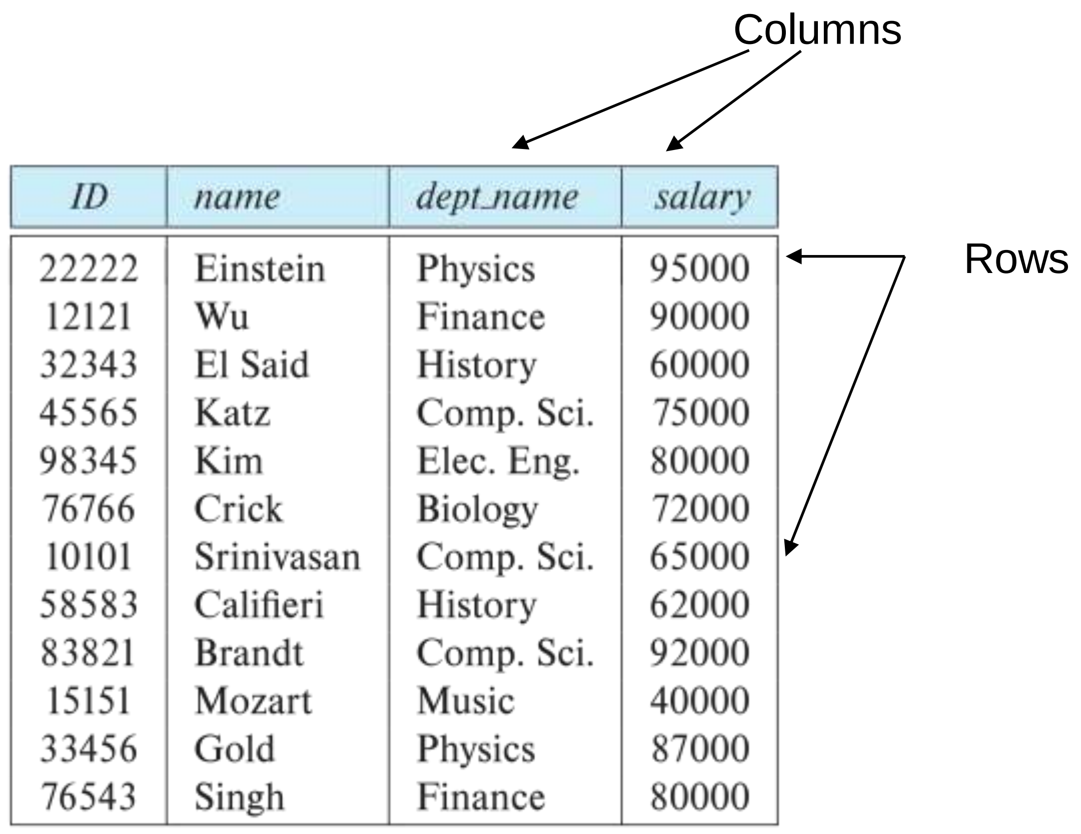
- A collection of tools for describing
  - Data
  - Data relationships
  - Data semantics
  - Data constraints
- **Relational Model:** The relational model uses a collection of tables to represent both data and the relationships among those data. Each table has multiple columns, and each column has a unique name. Tables are also known as relations. The relational model is an example of a record-based model. Each record type defines a fixed number of fields, or attributes. The columns of the table correspond to the attributes of the record type.
- **Entity-Relationship Model:** The entity-relationship (E-R) data model uses a collection of basic objects, called entities, and relationships among these objects. An entity is a “thing” or “object” in the real world that is distinguishable from other objects. The entity-relationship model is widely used in database design.

Data Models

- **Semi-structured Data Model:** The semi-structured data model permits the specification of data where individual data items of the same type may have different sets of attributes. This is in contrast to the data models mentioned earlier, where every data item of a particular type must have the same set of attributes. JSON (JavaScript Object Notation) and Extensible Markup Language (XML) are widely used semi-structured data representations.
- **Object-Based Data Model:** Object-oriented programming (Java, C++, or C#) has become the dominant software-development methodology. This led initially to the development of a distinct object-oriented data model, but today the concept of objects is well integrated into relational databases. Standards exist to store objects in relational tables. This can be seen as extending the relational model with notions of encapsulation, methods, and object identity.
- Other older models:
  - **Network model**
  - **Hierarchical model**

Relational Model

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


Columns
Rows

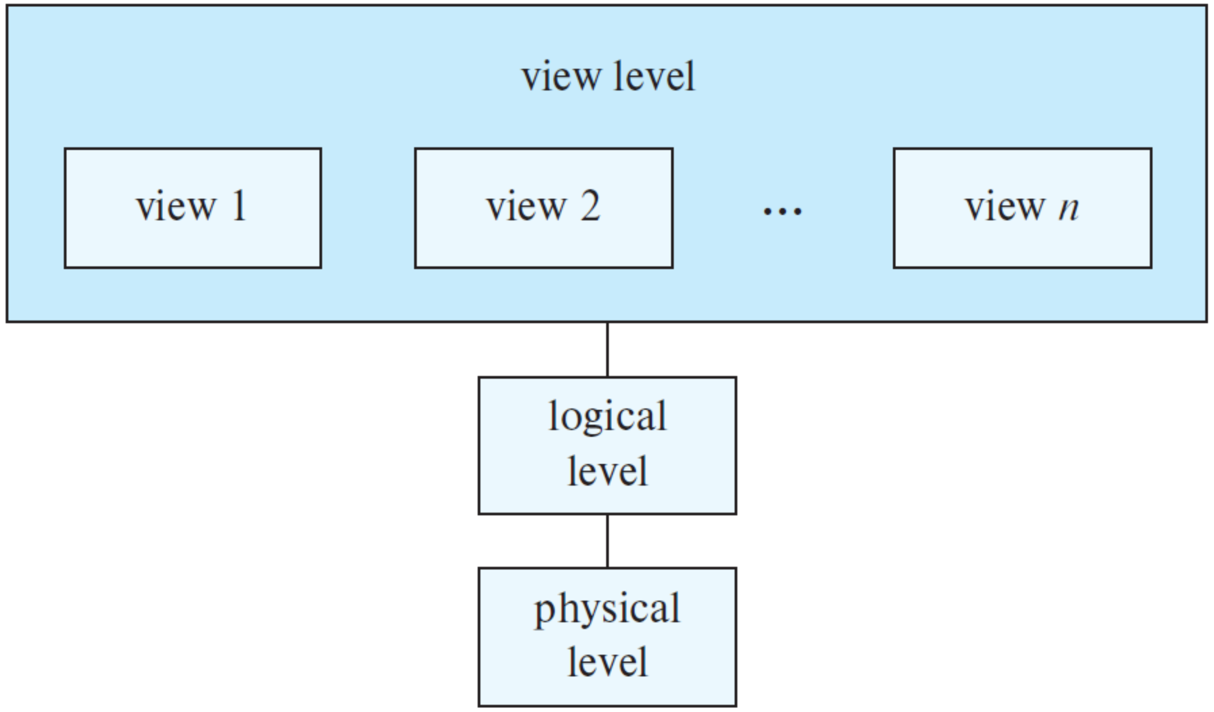
ID	*name*	*dept_name*	*salary*
22222	Einstein	Physics	95000
12121	Wu	Finance	90000
32343	El Said	History	60000
45565	Katz	Comp. Sci.	75000
98345	Kim	Elec. Eng.	80000
76766	Crick	Biology	72000
10101	Srinivasan	Comp. Sci.	65000
58583	Califieri	History	62000
83821	Brandt	Comp. Sci.	92000
15151	Mozart	Music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

(a) The *instructor* table

Ted Codd
Turing Award 1981

Data Abstraction

- ❑Hide the complexity of data structures to represent data in the database from users through several levels of data abstraction.
- ❑**Levels of Abstraction**



Levels of Abstraction

- **Physical level:** The lowest level of abstraction describes ***how*** the data are actually stored. The physical level describes complex low-level data structures in detail.
- **Logical level:** The next-higher level of abstraction describes ***what*** data are stored in the database, and what relationships exist among those data.
- **View level:** The highest level of abstraction describes only part of the entire database. Even though the logical level uses simpler structures, complexity remains because of the variety of information stored in a large database. Many users of the database system do not need all this information; instead, they need to access only a part of the database. The view level of abstraction exists to simplify their interaction with the system. The system may provide many views for the same database.

Levels of Abstraction

- Views can also hide information for security purposes: In addition to hiding details of the logical level of the database, the views also provide a security mechanism to prevent users from accessing certain parts of the database. For example, clerks in the university registrar office can see only that part of the database that has information about students; they cannot access information about salaries of instructors.

Instances and Schemas

- Databases change over time as information is inserted and deleted.
- A database schema corresponds to the variable declarations (along with associated type definitions) in a program. Each variable has a particular value at a given instant. The values of the variables in a program at a point in time correspond to an instance of a database schema.
- The overall design of the database is called the database **schema**.
  - Analogous to type information of a variable in a program.
- The collection of information (actual content) stored in the database at a particular moment is called an **instance** of the database.
  - Analogous to the value of a variable.
- Database systems have several schemas, partitioned according to the levels of abstraction.

Instances and Schemas

- ❑ **Physical schema** – the overall physical structure of the database
  - Describes the database design at the physical level.
- ❑ **Logical Schema** – the overall logical structure of the database
  - Describes the database design at the logical level.
- ❑ A database may also have several schemas at the view level, sometimes called subschemas, that describe different views of the database.

Physical Data Independence

- The ability to modify the physical schema without changing the logical schema.
- Applications depend on the logical schema.
- The physical schema is hidden beneath the logical schema and can usually be changed easily without affecting application programs.
- Application programs are said to exhibit physical data independence if they do not depend on the physical schema and thus need not be rewritten if the physical schema changes.
- In general, the interfaces between the various levels and components should be well defined so that changes in some parts do not seriously influence others.

Database Languages

- ❑A database system provides a **data-definition language (DDL)** to specify the database schema and a **data-manipulation language (DML)** to express database queries and updates.
- ❑They form parts of a single database language: The SQL

Data Definition Language (DDL)

- We specify a database schema by a set of definitions expressed by a special language called a data-definition language (DDL).
- Specification notation for defining the database schema.
- Data values stored in the database must satisfy certain consistency constraints. For example, suppose the university requires that the account balance of a department must never be negative. The DDL provides facilities to specify such constraints. The database system checks these constraints every time the database is updated.
  - **Domain Constraints:** A domain of possible values must be associated with every attribute (for example, integer types, character types, date/time types). Declaring an attribute to be of a particular domain acts as a constraint on the values that it can take. Domain constraints are the most elementary form of integrity constraint. They are tested easily by the system whenever a new data item is entered into the database.

Data Definition Language (DDL)

- **Referential Integrity:** There are cases where we wish to ensure that a value that appears in one relation for a given set of attributes also appears in a certain set of attributes in another relation (referential integrity). For example, the department listed for each course must be one that actually exists in the university. More precisely, the *dept_name* value in a *course* record must appear in the *dept_name* attribute of some record of the department relation.

- **Authorization:** We may want to differentiate among the users as far as the type of access they are permitted on various data values in the database. These differentiations are expressed in terms of authorization: **read authorization**, which allows reading, but not modification of data; **insert authorization**, which allows insertion of new data, but not modification of existing data; **update authorization**, which allows modification, but not deletion of data; and **delete authorization**, which allows deletion of data. We may assign the user all, none, or a combination of these types of authorization.

Data Definition Language (DDL)

- The output of the DDL is placed in the **data dictionary**, which contains **metadata**—that is, data about data.
- DDL compiler generates a set of table templates stored in a data dictionary.
- **Example:**

```
create table instructor (
    ID              char(5),
    name            varchar(20),
    dept_name       varchar(20),
    salary          double(8,2));
```

- 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 updating the data organized by the appropriate data model. The types of access are:
  - Retrieval of information stored in the database
  - Insertion of new information into the database
  - Deletion of information from the database
  - Modification of information stored in the database.
- There are basically two types of data-manipulation language:
  - **Procedural DMLs -** require a user to specify *what* data are needed and *how* to get those data.
  - **Declarative DMLs (Nonprocedural DMLs) -** require a user to specify *what* data are needed *without* specifying how to get those data.
- Declarative DMLs are usually easier to learn and use than are procedural DMLs.

Data Manipulation Language (DML)

- A **query** is a statement requesting the retrieval of information.
- The portion of a DML that involves information retrieval is called a **query language**.

SQL

- SQL is nonprocedural. A query takes as input one or several tables and always returns a single table.
- Example to find all instructors in Comp. Sci. dept:

```
select name
from instructor
where dept_name = 'Comp. Sci.';
```

- The query specifies that those rows from the table *instructor* where the *dept_name* is Comp. Sci. must be retrieved, and the name attribute of these rows must be displayed. The result of executing this query is a table with a single column labeled *name* and a set of rows, each of which contains the name of an instructor whose *dept_name* is Comp. Sci.

SQL

- Queries may involve information from more than one table.
- The following query finds the instructor ID and department name of all instructors associated with a department with a budget of more than Rs. 95,000.

```
select instructor.ID, department.dept_name
from instructor, department
where instructor.dept_name= department.dept_name and
department.budget > 95000;
```

- SQL is NOT a Turing machine equivalent language.
- To be able to compute complex functions SQL is usually embedded in some higher-level language.