Data Base Management System

Unit - I

What is Database Management System

- A Database Management System (DBMS) is software designed to store, retrieve, define, and manage data in a database.

OR

- It is a Software for creating, storing, maintaining, and accessing database files
- Makes using databases more efficient

- 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

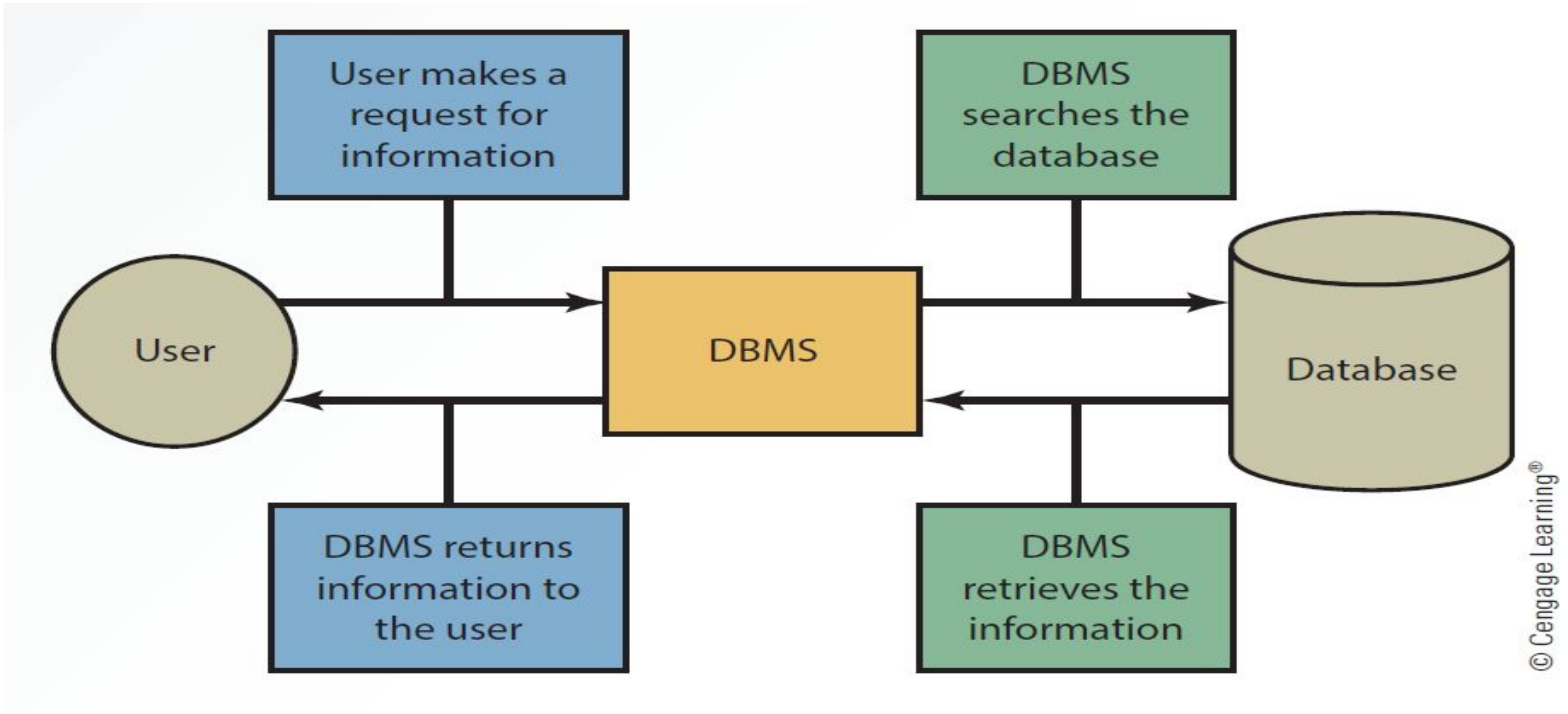
Basic Definitions

- **Data**: Known facts that can be recorded and have an implicit meaning.
- **Database**: A collection of related data.
- **Mini-world**: Some part of the real world about which data is stored in a database. For example, student grades and transcripts at a university.
- **Database Management System (DBMS)**: A software package / system to facilitate the creation and maintenance of a computerized database.
- **Database System**: The DBMS software together with the data itself. Sometimes, the applications are also included.

Components of DBMS

- **Hardware:** Can range from a PC to a network of computers.
- **Software:** DBMS, operating system, network software (if necessary) and also the application programs.
- **Data:** Used by the organization and a description of this data called the schema.
- **People:** Includes database designers, DBAs, application programmers, and end-users.
- **Procedure:** Instructions and rules that should be applied to the design and use of the database and DBMS.

Interaction Between the User, DBMS and Database


User makes a request for information
DBMS searches the database
User
DBMS
Database
DBMS returns information to the user
DBMS retrieves the information
© Cengage Learning®

- **Concerns of DBMS:**

  Integrity

  - Data integrity in the database is the correctness, consistency and completeness of data. Data integrity is enforced using the following three integrity constraints:
    - **Entity Integrity -** This is related to the concept of primary keys. All tables should have their own primary keys which should uniquely identify a row and not be NULL.
    - **Referential Integrity -** This is related to the concept of foreign keys. A foreign key is a key of a relation that is referred in another relation.
    - **Domain Integrity -** This means that there should be a defined domain for all the columns in a database.

  Consistency

  Redundancy

  Security

Advantage of DBMS over File Processing System

- **File System**

In a typical file processing system, each and every subsystem of the information system will have its own set of files.

The typical file-oriented system is supported by a conventional operating system.

Permanent records are stored in various files and a number of different application programs are written to extract records from and add records to the appropriate files.

Drawbacks of using file systems to store data

- Data redundancy and inconsistency
- Difficulty in accessing data
- Concurrent access anomalies
- Security problem
- Integrity problems
- Data isolation

Advantages of DBMS over file system

Redundancy can be reduced:

- A major difficulty was that many applications used their own special files of data. Thus, some data items were common to several applications.
- Database systems can eliminate data redundancy, since all applications share a common pool of data.

Inconsistency can be avoided:

- Since the same information can be present at multiple files in file systems, data needs to be updated at all the files whenever any changes in data occur.
- For example, a changed customer address may be reflected in personal information file, but not in savings account file. By having a centralized database, most of this can be avoided.

Reduced programming effort:

- Database provides a separation between programs and data, so that programs can be somewhat independent of the details of data definition.
- By providing access to a pool of shared data and by supporting powerful data manipulating languages, database systems eliminate a large amount initial and maintenance programming.

Security can be enforced:

- Not every user should be able to access all the data.
- For example, in a banking system, payroll personnel need only the information about various bank employees. They do not need access to customer accounts. Since, in the file systems, application programs are added to systems in ad-hoc manner, it is difficult to enforce security constraints.

Integrity can be maintained:

- The data values stored in the database must satisfy certain types of consistency constraints.
- For example, the balance of a bank account may never fall below a particular amount. In file system, these constraints are enforced by adding appropriate code in various application programs. However, when new constraints are added, it is difficult to change the programs to enforce them.

Applications of DBMS

There are different fields where a database management system is utilized. Following are a few applications which utilize the information base administration framework.

Railway Reservation System

In the rail route reservation framework, the information base is needed to store the record or information of ticket appointments, status about train's appearance, and flight. Additionally, if trains get late, individuals become acquainted with it through the information base update.

Library Management System

There are loads of books in the library so; it is difficult to store the record of the relative multitude of books in a register or duplicate. Along these lines, the data set administration framework (DBMS) is utilized to keep up all the data identified with the name of the book, issue date, accessibility of the book, and its writer.

Banking
Database the executive's framework is utilized to store the exchange data of the client in the information base.

Education Sector

Presently, assessments are led online by numerous schools and colleges. They deal with all assessment information through the data set administration framework (DBMS). In spite of that understudy's enlistments subtleties, grades, courses, expense, participation, results, and so forth all the data is put away in the information base.

Credit card exchanges

The database Management framework is utilized for buying on charge cards and age of month to month proclamations.

Social Media Sites

We all utilization of online media sites to associate with companions and to impart our perspectives to the world. Every day, many people group pursue these online media accounts like Facebook, Twitter, and Google in addition to. By the utilization of the data set administration framework, all the data of clients are put away in the information base and, we become ready to interface with others.

Broadcast communications

Without DBMS any media transmission organization can't think. The Database the executive's framework is fundamental for these organizations to store the call subtleties and month to month postpaid bills in the information base.

Online Shopping

These days, web-based shopping has become a major pattern. Nobody needs to visit the shop and burn through their time. Everybody needs to shop through web based shopping sites, (for example, Amazon, Flipkart, Snapdeal) from home. So all the items are sold and added uniquely with the assistance of the information base administration framework (DBMS). Receipt charges, installments, buy data these are finished with the assistance of DBMS.

Human Resource Management

Big firms or organizations have numerous specialists or representatives working under them. They store data about worker's compensation, assessment, and work with the assistance of an information base administration framework (DBMS).

Airline Reservation System

This framework is equivalent to the railroad reservation framework. This framework additionally utilizes an information base administration framework to store the records of flight takeoff, appearance, and defer status.

Purpose of database system

- **Data redundancy and inconsistency**
  - Same information may be duplicated in several places.
  - All copies may not be updated properly.

- **Difficulty in** new program to carry out each new task

- **Data isolation**
  - Data in different formats.
  - Difficult to write new application programs.
  - files and formats

- **Security problems**
  - Every user of the system should be able to access only the data they are permitted to see.
  - E.g. payroll people only handle employee records, and cannot see customer accounts; tellers only access account data and cannot see payroll data.
  - Difficult to enforce this with application programs.

- **Integrity problems**
  - Data may be required to satisfy constraints.
  - E.g. no account balance below Rs.2500.
  - Again, difficult to enforce or to change constraints with the file-processing approach.

Advantage of DBMS

- Controlling Redundancy
- Sharing of Data
- Data Consistency
- Integration of Data
- Integration Constraints
- Data Security
- Report Writers
- Control Over Concurrency
- Backup and Recovery Procedures
- Data Independence

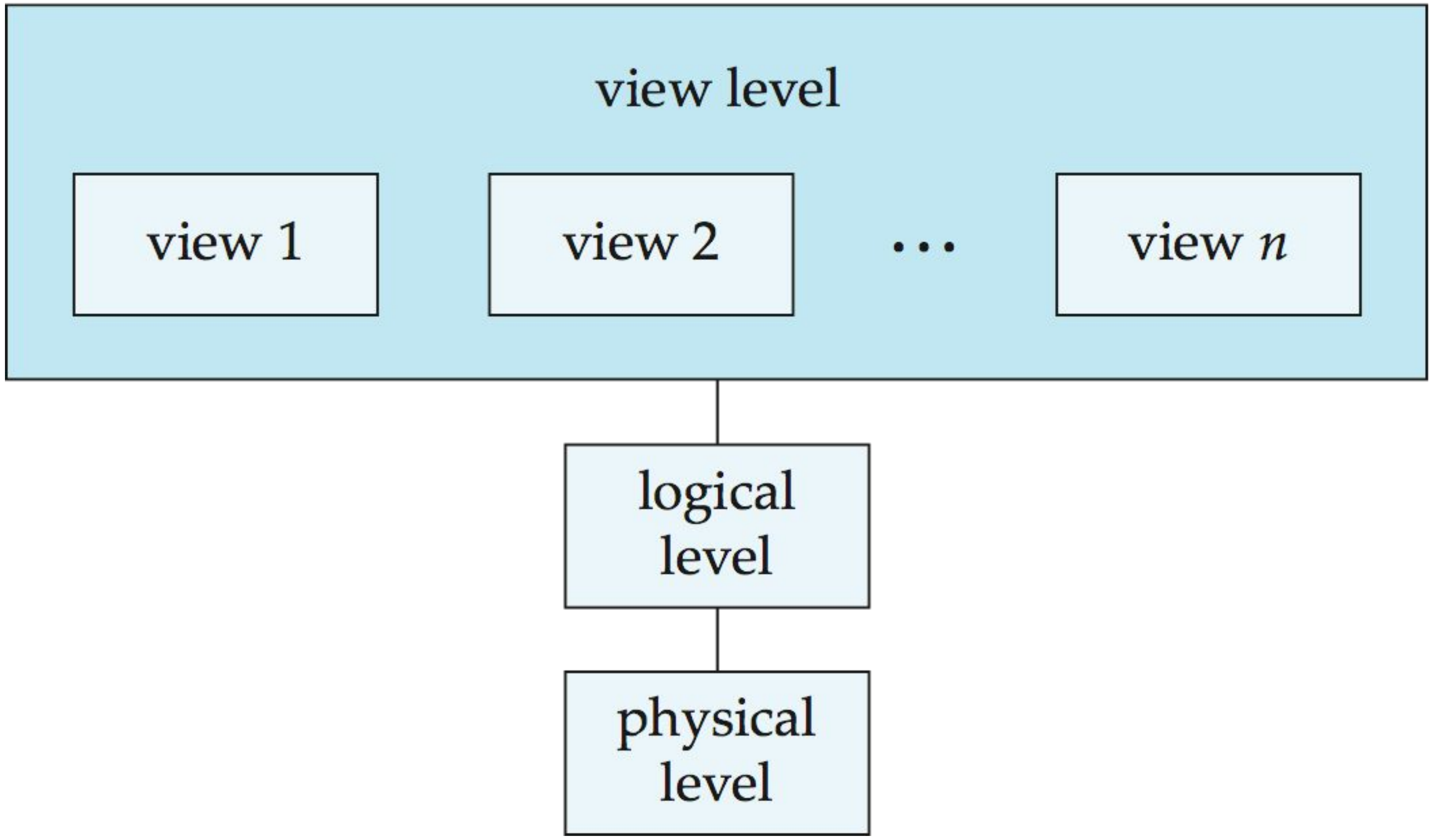
Disadvantage of DBMS

- Cost of Hardware and Software
- Cost of Data Conversion
- Cost of Staff Training
- Appointing Technical Staff
- Database Damage

Views of data

- 1. DATA ABSTRACTION
- 2. INSTANCES & SCHEMAS
- 3. DATA MODELS

1. DATA ABSTRACTION



- **Physical Level**

The lowest level of abstraction describes how a system actually stores data. The physical level describes complex low- level data structures in detail.

- **Logical Level**

The next higher level of abstraction describes what data the database stores, and what relationships exist among those data.

The logical level thus describes an entire database in terms of a small number of relatively simple structures.

Although implementation of the simple structures at the logical level may involve complex physical level structures, the user of the logical level does not need to be aware of this complexity. This referred to as physical data independence.

- **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 a 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.

2. Instances and Schemas

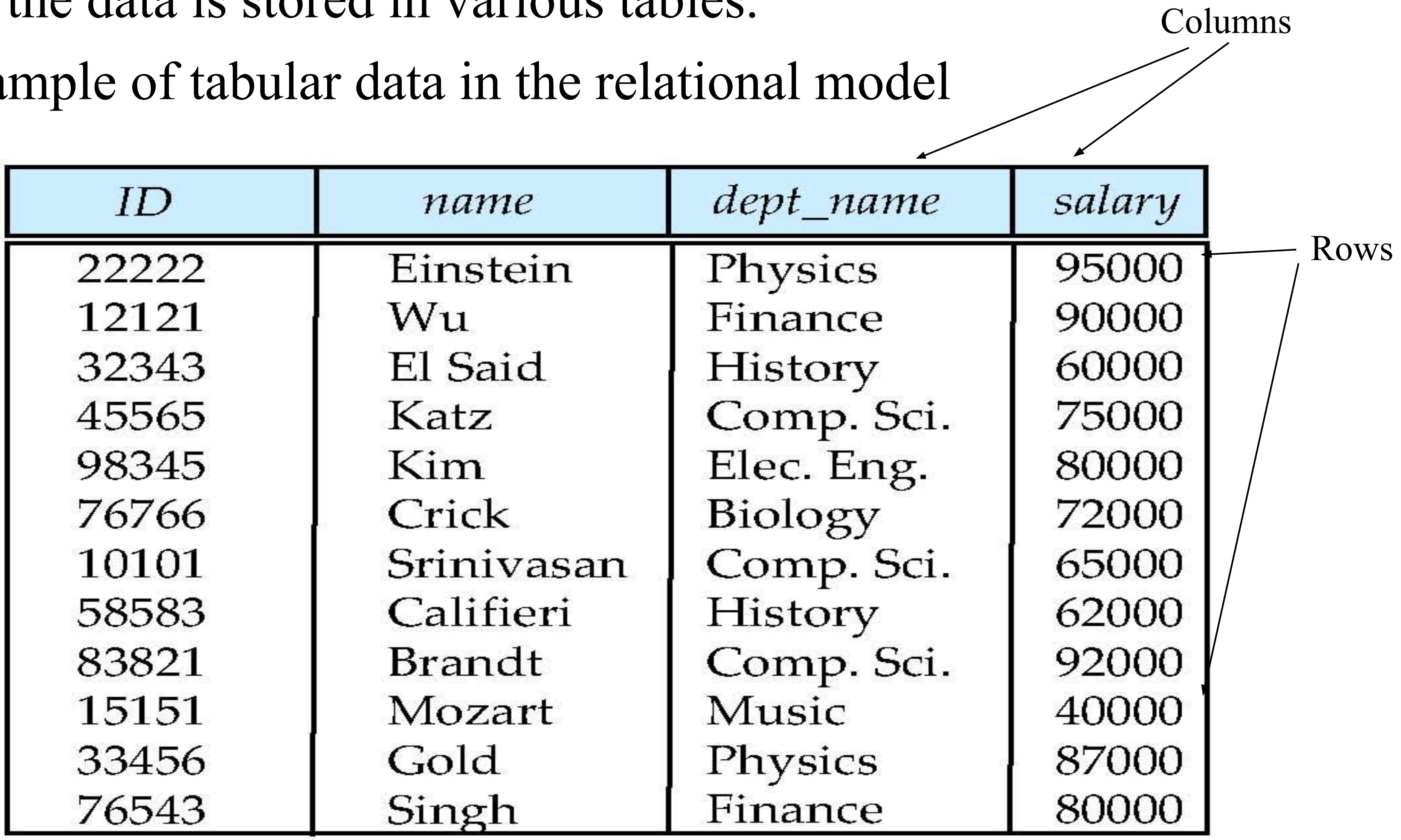
- Similar to types and variables in programming languages
- **Logical Schema** – the overall logical structure of the database
  - Example: The database consists of information about a set of customers and accounts in a bank and the relationship between them
    - 4 Analogous to type information of a variable in a program
- **Physical schema**– the overall physical structure of the database
- **Instance** – the actual content of the database at a particular point in time
  - Analogous to the value of a variable
- **Physical Data Independence** – the ability to modify the physical schema without changing the logical schema
  - Applications depend on the logical schema
  - 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.

3. 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)
- Semi structured data model (XML)
- 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

A Sample Relational Database

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

dept_name	*building*	*budget*
Comp. Sci.	Taylor	100000
Biology	Watson	90000
Elec. Eng.	Taylor	85000
Music	Packard	80000
Finance	Painter	120000
History	Painter	50000
Physics	Watson	70000

(b) The *department* table

Data Definition Language (DDL)

- Specification notation for defining the database schema

  Example: **create table** *instructor* (
  ```
                   ID          char(5),
                   name        varchar(20),
                   dept_name   varchar(20),
                   salary      numeric(8,2))
  ```
- DDL compiler generates a set of table templates stored in a ***data dictionary***
- 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
- 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

Design Approaches

- Need to come up with a methodology to ensure that each of the relations in the database is "good"

- Two ways of doing so:

  - Entity Relationship Model (Chapter 7)
    - Models an enterprise as a collection of *entities* and *relationships*
    - Represented diagrammatically by an *entity-relationship diagram:*

  - Normalization Theory (Chapter 8)
    - Formalize what designs are bad, and test for them

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

Database Engine

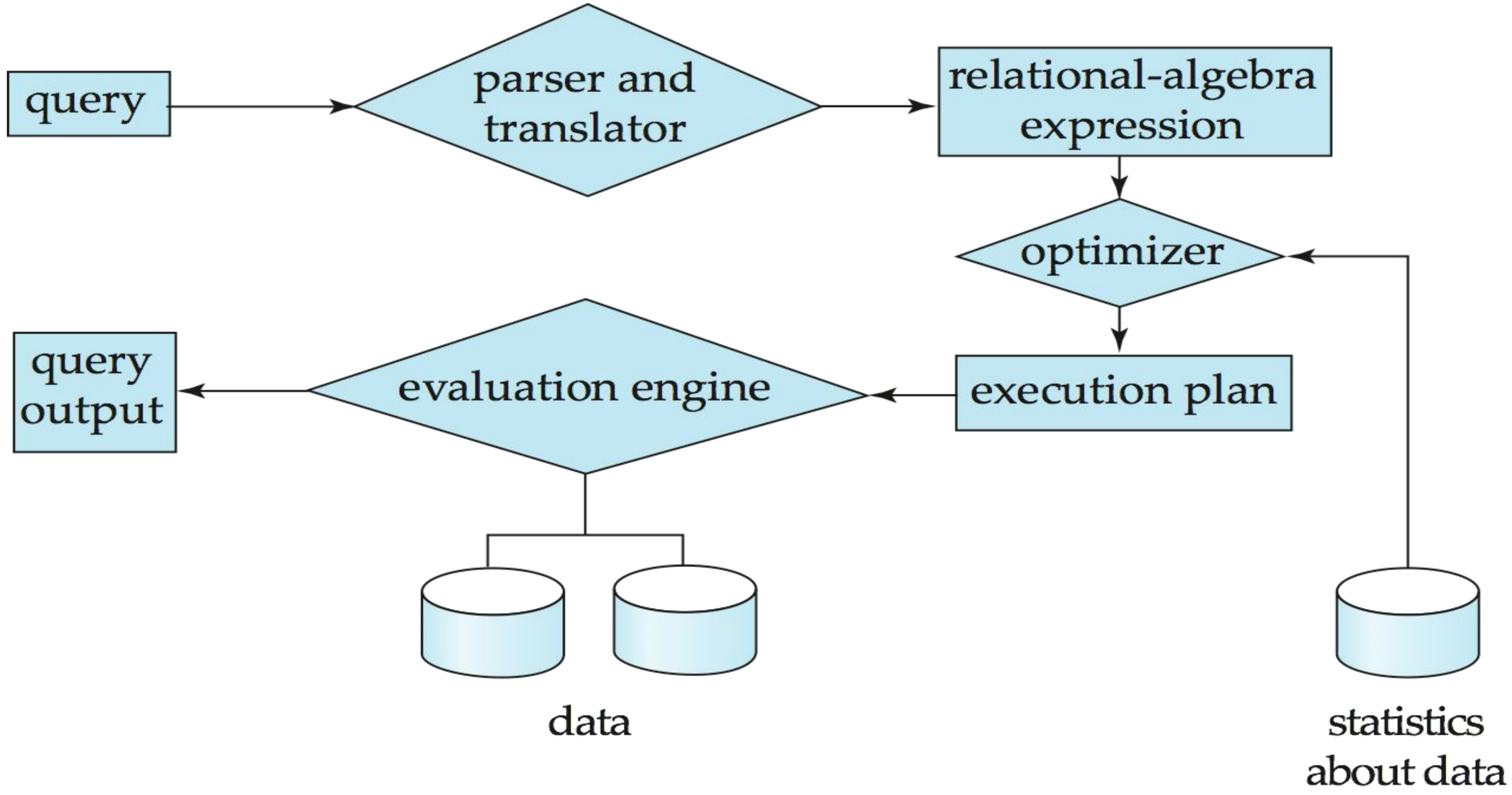
- Storage manager
- Query processing
- Transaction manager

Storage Management

- **Storage manager** is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
    - Interaction with the OS file manager
    - Efficient storing, retrieving and updating of data
- Issues:
    - Storage access
    - File organization
    - Indexing and hashing

Query Processing

1. Parsing and translation
2. Optimization
3. Evaluation



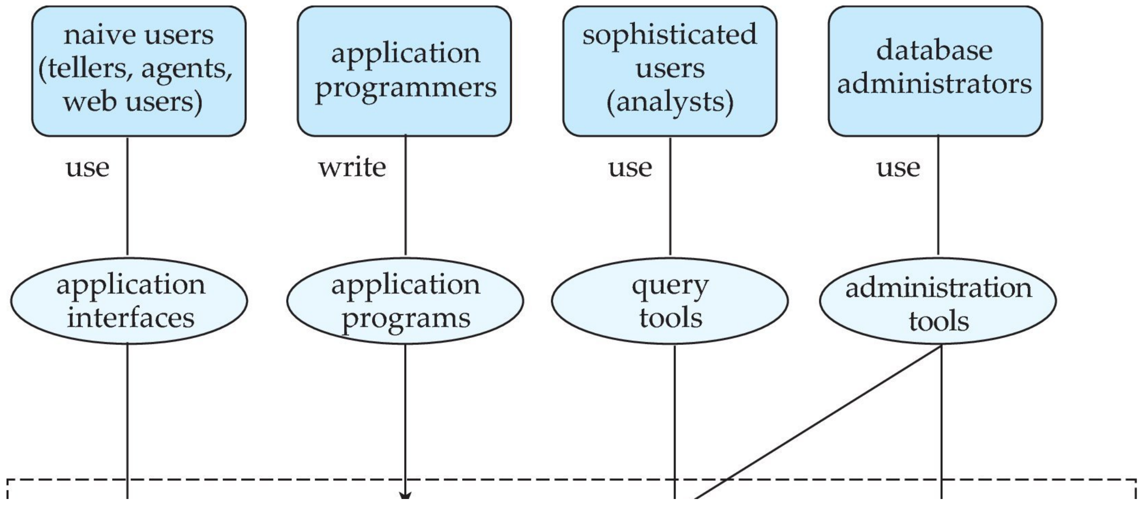
Query Processing (Cont.)

- Alternative ways of evaluating a given query
    - Equivalent expressions
    - Different algorithms for each operation
- Cost difference between a good and a bad way of evaluating a query can be enormous
- Need to estimate the cost of operations
    - Depends critically on statistical information about relations which the database must maintain
    - Need to estimate statistics for intermediate results to compute cost of complex expressions

Transaction Management

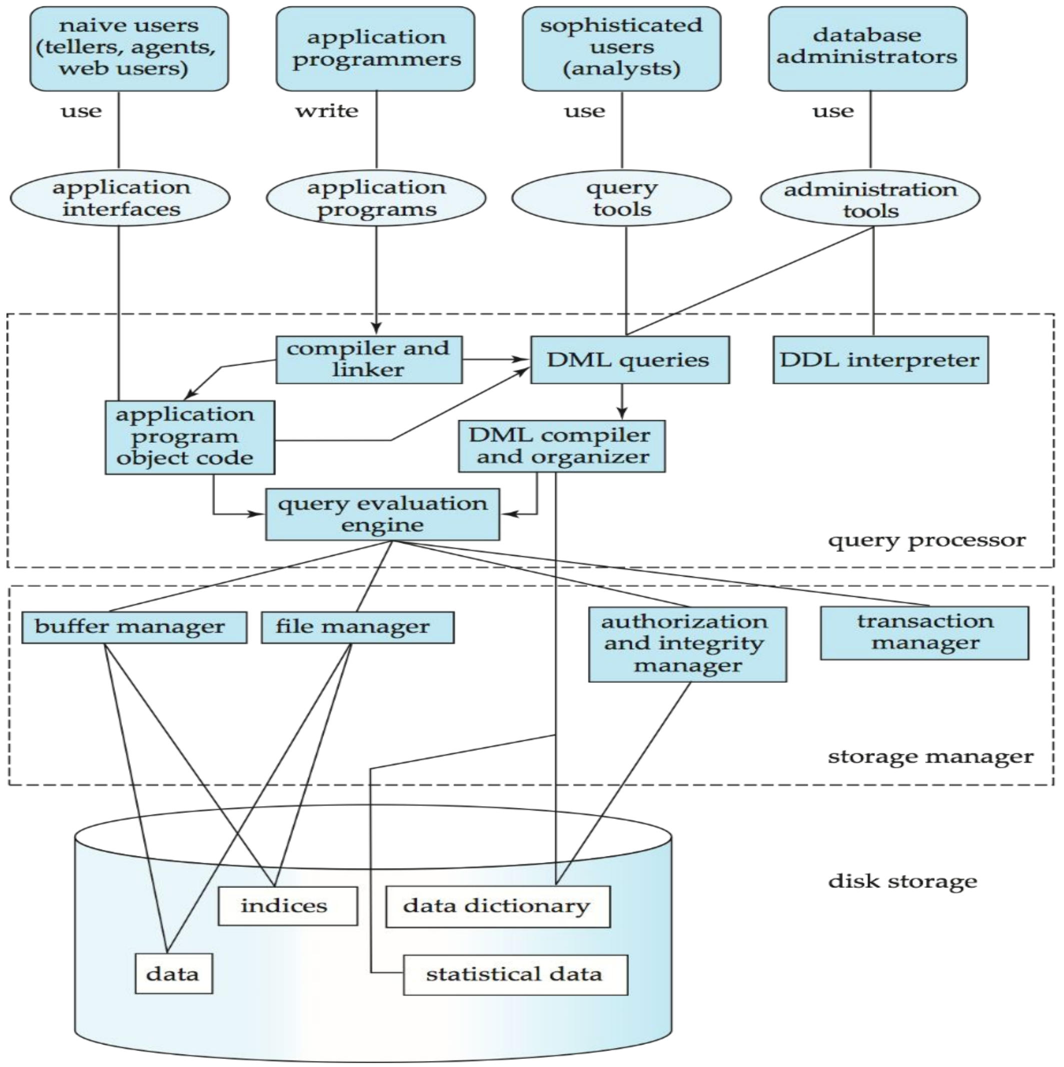
- What if the system fails?
- What if more than one user is concurrently updating the same data?
- A **transaction** is a collection of operations that performs a single logical function in a database application
- **Transaction-management component** ensures that the database remains in a consistent (correct) state despite system failures (e.g., power failures and operating system crashes) and transaction failures.
- **Concurrency-control manager** controls the interaction among the concurrent transactions, to ensure the consistency of the database.

Database Users and Administrators



Database

Database System Internals



Database Architecture

The architecture of a database systems is greatly influenced by the underlying computer system on which the database is running:

- Centralized
- Client-server
- Parallel (multi-processor)
- Distributed

Database Users

Users are differentiated by the way they expect to interact with the system

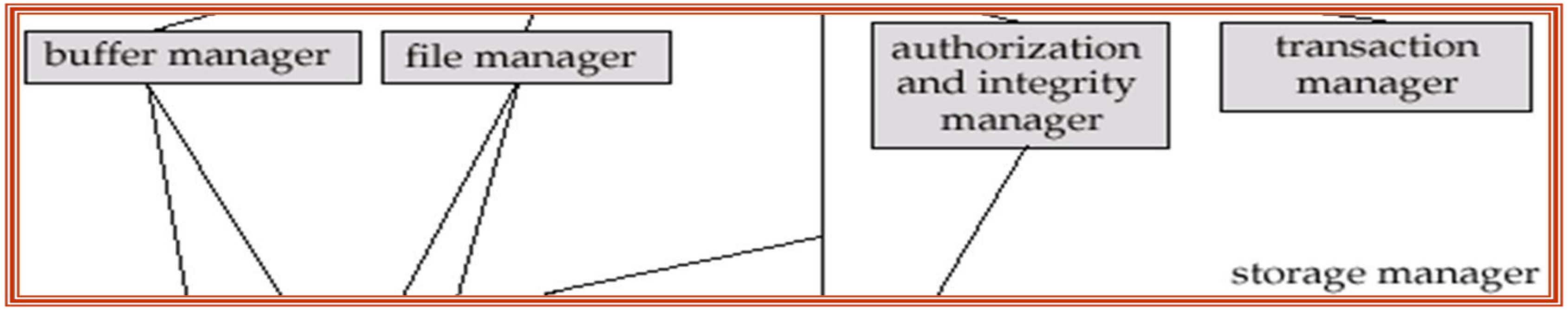
- **Application programmers** – interact with system through DML calls
- **Sophisticated users** – form requests in a database query language
- **Specialized users** – write specialized database applications that do not fit into the traditional data processing framework
- **Naïve users** – invoke one of the permanent application programs that have been written previously

E.g. people accessing database over the web, bank tellers, clerical staff

Storage Management

- Storage manager
  is a program module that provides the interface between the low-level data stored in the database and the application programs and queries submitted to the system.
- The storage manager is responsible to the following tasks:
  interaction with the file manager efficient storing, retrieving and updating of data

Storage Manager Components



The storage manager components include:

- **Authorization and integrity manager,**
  which tests for the satisfaction of integrity
- **Transaction manager,**
  which ensures that the database remains in a consistent
- **File manager,**
  which manages the allocation of space on disk storage and the
- **Buffer manager,**
  which is responsible for fetching data from disk storage into main memory

Authorization and integrity manager, which tests for the satisfaction of integrity
constraints and checks the authority of users to access data.

Transaction manager, which ensures that the database remains in a consistent
(correct) state despite system failures, and that concurrent transaction executions proceed without conflicting.

File manager, which manages the allocation of space on disk storage and the
data structures used to represent information stored on disk.

Buffer manager, which is responsible for fetching data from disk storage into
main memory, and deciding what data to cache in main memory. The buffer manager is a critical part of the database system, since it enables the database to handle data sizes that are much larger than the size of main memory.

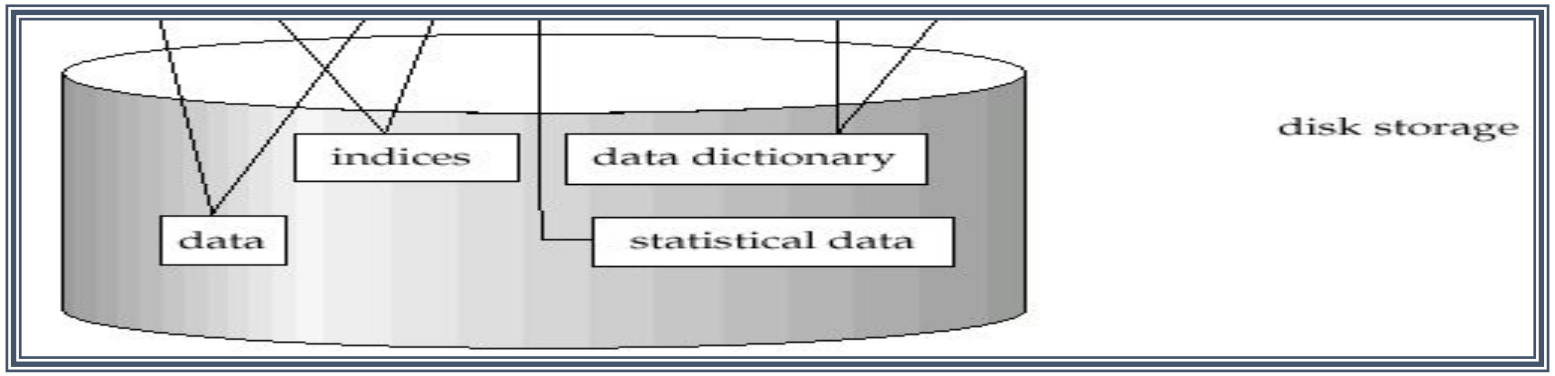
Storage Manager

The storage manager implements several data structures as part of the physical system implementation:

Data files, which store the database itself**.**

Data dictionary, which stores metadata about the structure of the database, in particular the schema of the database.

Indices, which provide fast access to data items that hold particular values**.**



Query Processor

The query processor components include
***DDL interpreter**, which interprets DDL statements and records the definitions* in the data dictionary.
***DML compiler**, which translates DML statements in a query language into an* evaluation plan consisting of low-level instructions that the query evaluation engine understands.
A query can usually be translated into any of a number of alternative evaluation plans that all give the same result.
The DML compiler also performs query optimization, that is, it picks the lowest cost evaluation plan from among the alternatives.
***Query evaluation engine**, which executes low-level instructions generated by* the DML compiler.

Data Independence- Database Schemas

- **Database Schema**: The *description* of a database. Includes descriptions of the database structure and the constraints that should hold on the database.
- **Schema Diagram**: A diagrammatic display of (some aspects of) a database schema.

Database Schema Vs. Database State

- **Database State:** Refers to the content of a database at a moment in time.
- **Initial Database State:** Refers to the database when it is loaded
- **Valid State:** A state that satisfies the structure and constraints of the database.

Three-Schema Architecture

- Defines DBMS schemas at *three levels*:
  - **Internal schema** at the internal level to describe physical storage structures and access paths. Typically uses a *physical* data model.
  - **Conceptual schema** at the conceptual level to describe the structure and constraints for the *whole* database for a community of users. Uses a *conceptual* or an *implementation* data model.
  - **External schemas** at the external level to describe the various user views. Usually uses the same data model as the conceptual level.

Data Independence

When a schema at a lower level is changed, only the **mappings** between this schema and higher-level schemas need to be changed in a DBMS that fully supports data independence. The higher-level schemas themselves are *unchanged*. Hence, the application programs need not be changed since they refer to the external schemas.

Logical Data Independence

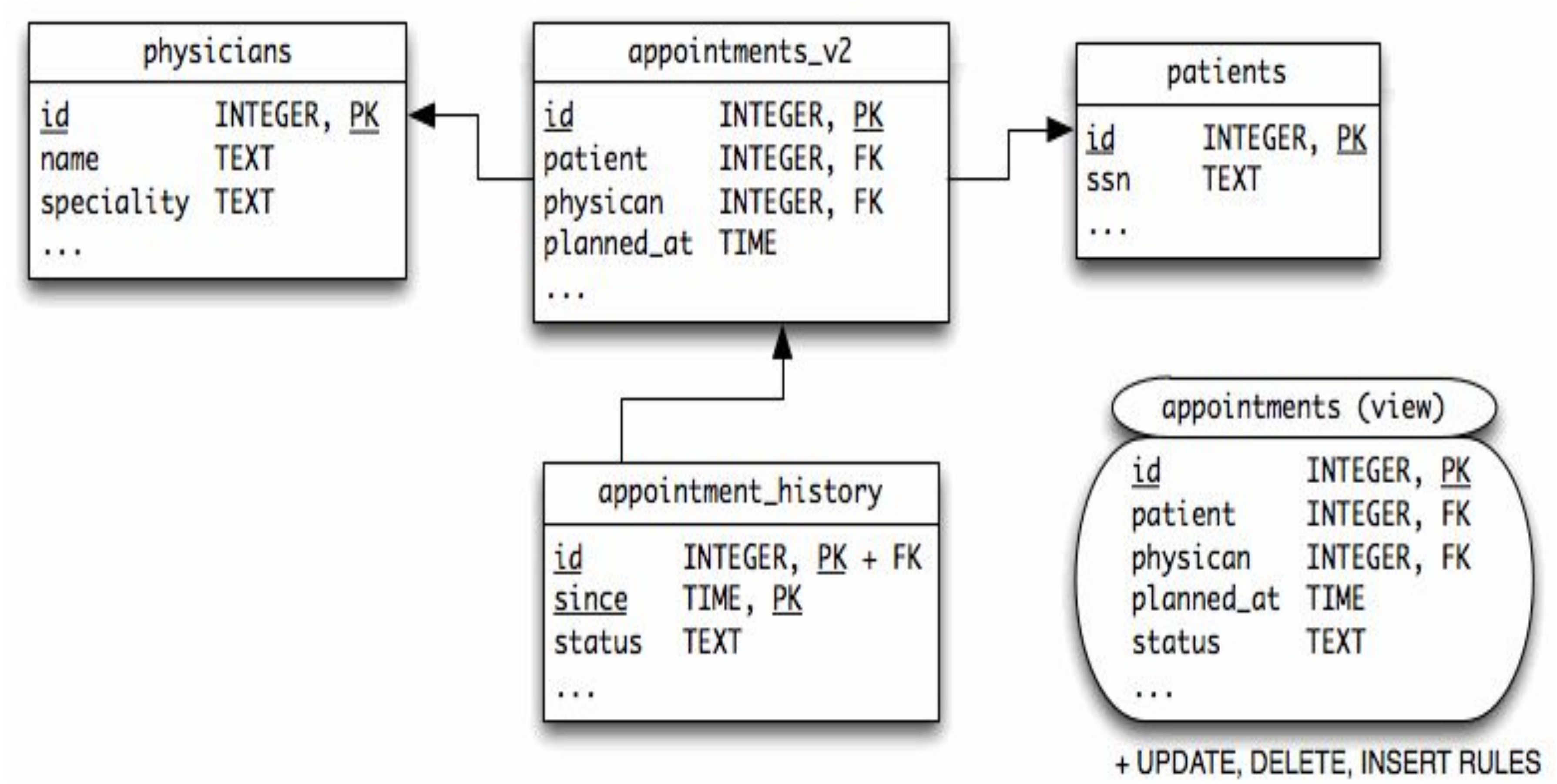
The ability to change the logical schema without changing the external schema or application programs is called as Logical Data Independence.

OR

The ability to change the logical schema without having to change the external schema.

Examples

The addition or removal of new entities, attributes, or relationships to the conceptual schema should be possible without having to change existing external schemas or having to rewrite existing application programs.


physicians
id INTEGER, PK
name TEXT
speciality TEXT
...
appointments_v2
id INTEGER, PK
patient INTEGER, FK
physican INTEGER, FK
planned_at TIME
...
patients
id INTEGER, PK
ssn TEXT
...
appointment_history
id INTEGER, PK + FK
since TIME, PK
status TEXT
...
appointments (view)
id INTEGER, PK
patient INTEGER, FK
physican INTEGER, FK
planned_at TIME
status TEXT
...
+ UPDATE, DELETE, INSERT RULES

Physical Data Independence

- The ability to change the physical schema without changing the logical schema is called as Physical Data Independence. Changes in the physical schema may include.
- Using new storage devices.
- Using different data structures.
- Switching from one access method to another.
- Using different file organizations or storage structures.
- Modifying indexes.

Examples

A change to the internal schema, such as using different file organization or storage structures, storage devices, or indexing strategy, should be possible without having to change the conceptual or external schemas.

Summary :

- **Logical Data Independence**: The capacity to change the conceptual schema without having to change the external schemas and their application programs.
- **Physical Data Independence**: The capacity to change the internal schema without having to change the conceptual schema.

The Evolution of Data Models

- Flat Files:
  - Earlier, punched cards technology was used to store data – later, files.
  - But the files have no as such advantage, rather have several limitations.

Pros and Cons of Flat Files

- Advantages
- Limitations
- Various access methods , e.g., sequential, indexed, random
- Requires extensive programming in third-generation language such as COBOL, BASIC.
- Separation and isolation: Each program maintains its own set of data, users of one program may not be aware of holding or blocking by other programs that are being used somewhere else, by another user.
- Duplication of data – same data is held by different programs, thus, wastes space and resources.
- High maintenance costs such as ensuing data consistency and controlling access Sharing granularity is very coarse.
- Weak security.

Hierarchical Data Model:

- In this model, files are related in a parent/child manner, with each child file having at most one parent file.
- Prominent hierarchical database model was IBM’s first DBMS called IMS (Information Management System).

Pros and Cons of Hierarchical Data Model:

- Advantages
- Limitations
- Efficient searching.
- Complex implementation
- Less redundant data.
- Difficult to manage and lack of standards, can’t easily handle many-many relationships.
- Data independence.
- Lacks structural independence.
- Database security and integrity.

Network Data Model:

- Early 1960s, Charles Bachmann developed first DBMS at Honeywell, Integrated Data Store (IDS)
- It standardized in 1971 by the CODASYL group (Conference on Data Systems Languages).
- In Network data model, files are related as owners and members, similar to the common network model except that each member file can have more than one owner.
- Network data model identified the following three database components:
- Network schema—database organization[structure]
- Sub-schema—views of database per user
- Data management language — at low level , procedural
- Prominent network database model was CODASYL DBTG model where as IDMS was the most popular network DBMS.

Pros and Cons of Network Data Model

- Advantages
- Limitations
- Ability to handle more relationship types
- System complexity and difficult to design and maintain
- Ease of data access
- Lack of structural independence as data access method is navigational.
- Data Integrity
- Data Independence

Relational Database Model

- The relational database model was conceived by E. F. Codd in 1970.
- It can be defined using the following two terminologies:
  - Instance – a table with rows or columns.
  - Schema – specifies the structure (name of relation, name and type of each column)
- The model is based on branches of mathematics called set theory and predicate logic.

Pros & Cons of Relational Database Model

- Advantages:
  - Ease of use because of rows and columns.
  - Flexibility with respect to manipulation by operators such as project and join.
  - Precision due to the manipulation of the relations between the tables ensures that there is no ambiguity.
- Disadvantages :
  - Machine Performance in case of large number of joins.
  - Physical Storage Consumption
  - Slow extraction of meaning from data

Object Oriented Database Model:

- It supports the modeling and creation of the data as objects.
- **Pros and Cons of Object Oriented Database Model**:
  - Advantages
  - Limitations
  - Can efficiently manage a large number of different data types.
  - Switching an existing database to OODBMS requires an entire change from scratch.
  - Objects with complex behaviors are easy to handle using inheritance and polymorphism etc.
  - An OODBMS is typically tied to a specific programming language and an API; this reduces its flexibility.
  - Reduces the large number of relations by creating objects.
  - Ad-hoc queries are difficult to implement as one cannot join two classes as one can join two tables in RDBMS.
  - Therefore, queries depend upon the design of the system.
  - Creates problems when deleting data in bulk.

Object Relational Database Model:

- Object relational databases span the object and relational concepts.
- Advantages
- Limitations
- Large storage capacity
- The architecture of the object relational model is not appropriate for web applications.
- High access speed