DATABASE SYSTEM CONCEPTS AND ARCHITECTURE

Lecture 2

50

A Map Is a Model of Reality

51

Data Abstraction

- A fundamental characteristic of a database approach is that it provides some level of data abstraction.
  - Data abstraction generally refers to the suppression of details of data organization and storage.
  - Highlights essential features for improved understanding of data.

52

Why Use Models?

- Models can be useful when we want to **examine or manage part** of the real world
- The costs of using a model are often considerably lower than the costs of using or experimenting with the real world itself
- Examples:
  - airplane simulator
  - nuclear power plant simulator
  - flood warning system
  - model of a heat reservoir
  - map

53

Why use Models? Intent of Models

- A model is a **means of communication**
  - However, Users of a model must have a certain amount of knowledge in common
- A model **emphasizes selected aspects**
- A model is described in some language/ standardized notation
- **A model can be erroneous**

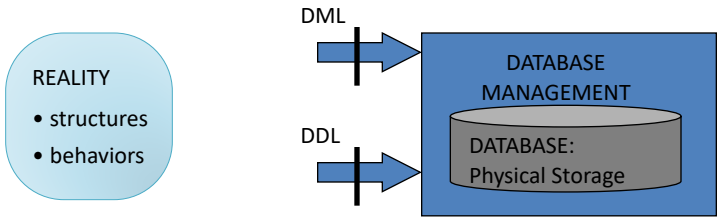
54

Formal Definitions: Data Models

- **Data Model**: A set of concepts to describe the *structure* of a database, and certain *constrains (behaviors)* that the database should obey.
- **Data Model Operations**: Operations for specifying database retrievals and updates by referring to the concepts of the data model.
  - Operations on the data model may include *basic operations* and *user-specific operations*.

55

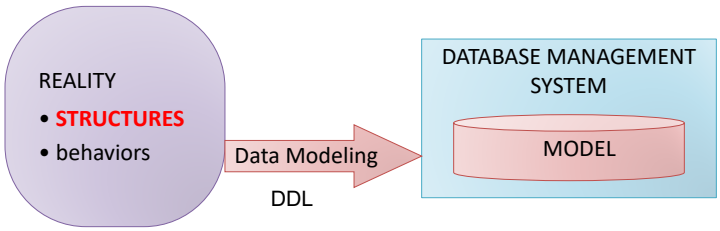
Models of Reality



- A **database** is a model of **structures** that mimic reality
- The **use of a database** reflect **behaviors** of reality
- A **database management system** is a **software system** which supports the **definition and use of a database**
  - **DDL: Data Definition Language**
  - **DML: Data Manipulation Language**

56

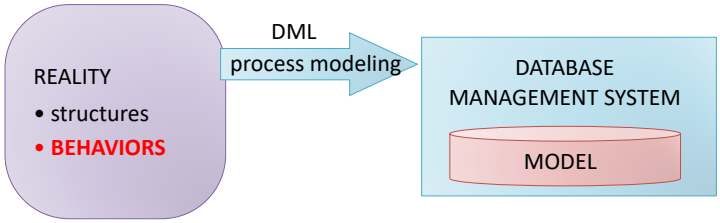
Data Modeling



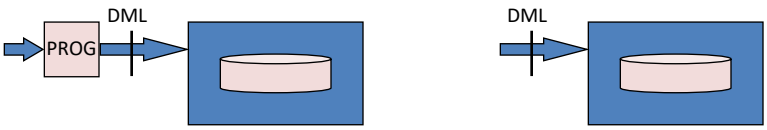
- The **model** represents a perception of **structures** of reality
- The **data modeling** process is to fix a **perception of structures of reality** and **represent this perception**
- In the data modeling process, we **select** aspects, and we **abstract**

57

Process Modeling



- The **use of the model** reflects **processes of reality**
- Processes may be represented by programs with embedded database **queries and updates**
  - Note: processes may be represented by ad-hoc database queries and updates at run-time



58

Summary: Database Design

- The purpose of database design is to create a database which
  - Is a model of structures of reality
  - Ensures concept formation through abstraction
  - Supports queries and updates modeling processes of reality
  - **Runs efficiently**

59

DATABASE SCHEMAS AND ARCHITECTURES

Lecture 2

60

Database Schemas Vs State

- **Database Schema**: The *description* of a database. Includes descriptions of the database **structure** and the **constraints (behaviors)** that should hold on the database.
  - **Schema Diagram**: A diagrammatic display of (some aspects of) a database schema.
  - **Schema Construct**: A component of the schema or an object within the schema, e.g., STUDENT, COURSE.
- **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.

61

Database Schema Vs. Database State

- **Distinction**
  - The **database schema** changes *very infrequently*. The **database state** changes *every time the database is updated.*
  - **Schema** is also called **intension**, whereas **State** is called **extension**.

62

Database Schema Vs State

STUDENT

Name	Student_number	Class	Major

COURSE

Course_name	Course_number	Credit_hours	Department

PREREQUISITE

Course_number	Prerequisite_number

SECTION

Section_identifier	Course_number	Semester	Year	Instructor

GRADE_REPORT

Student_number	Section_identifier	Grade

SCHEMA

STUDENT

Name	Student_number	Class	Major
Smith	17	1	CS
Brown	8	2	CS

COURSE

Course_name	Course_number	Credit_hours	Department
Intro to Computer Science	CS1310	4	CS
Data Structures	CS3320	4	CS
Discrete Mathematics	MATH2410	3	MATH
Database	CS3380	3	CS

SECTION

Section_identifier	Course_number	Semester	Year	Instructor
85	MATH2410	Fall	07	King
92	CS1310	Fall	07	Anderson
102	CS3320	Spring	08	Knuth
112	MATH2410	Fall	08	Chang
119	CS1310	Fall	08	Anderson
135	CS3380	Fall	08	Stone

STATE

63

Database Schema Vs State

RELATIONS

Relation_name	No_of_columns
STUDENT	4
COURSE	4
SECTION	5
GRADE_REPORT	3
PREREQUISITE	2

COLUMNS

Column_name	Data_type	Belongs_to_relation
Name	Character (30)	STUDENT
Student_number	Character (4)	STUDENT
Class	Integer (1)	STUDENT
Major	Major_type	STUDENT
Course_name	Character (10)	COURSE
Course_number	XXXXNNNN	COURSE
….	….	…..
….	….	…..
….	….	…..
Prerequisite_number	XXXXNNNN	PREREQUISITE

META DATA

64

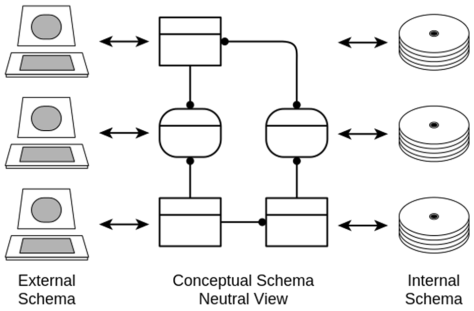
The ACID properties of a DBMS (while working)

1. **Atomicity** requires that each transaction be "all or nothing".
2. **Consistency:** The consistency property ensures that any transaction will bring the database from one valid state to another.
3. **Isolation:** The isolation property ensures that the concurrent execution of transactions results in a system state that would be obtained if transactions were executed serially.
4. **Durability:** The durability property ensures that once a transaction has been committed, it will remain so, even in the event of power loss, crashes, or errors.

65

Three-Schema Architecture and Data Independence

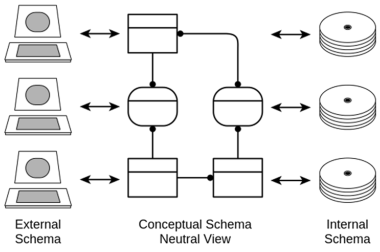
- Proposed to support DBMS characteristics of:
  - **Program-data independence**.
  - Support of **multiple views** of the data.



66

Three-Schema Architecture

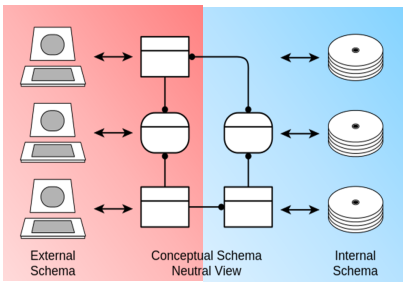
- **Internal schema** at the internal level to describe physical storage structures and access paths.
- **Conceptual schema** at the conceptual level to describe the structure and constraints for the *whole* database for a community of users
- **External schemas** at the external level to describe the various user views.



68

Data Independence

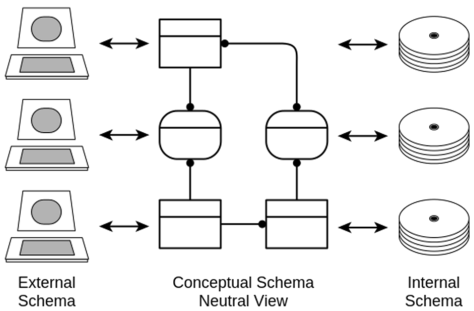
- **Logical Data Independence**: The capacity to change the **conceptual schema** without having to change the **external schemas** and their application programs.
- **Physical Data Independence**: The capacity to change the **internal schema** without having to change the **conceptual schema**.



69

Three-Schema Architecture

- **Mappings** among schema levels are needed to transform requests and data.
  - Programs refer to an external schema, which are mapped by the DBMS to the internal schema for execution.



70

Consequence of Data Independence

- When a schema at the internal level is changed, only the **mappings** between the internal schema and conceptual schemas need to be changed in a DBMS that fully supports data independence.
  - Similarly, mappings between external and conceptual schemas duly change
- The External schemas themselves are *unchanged*.
- Hence, the application programs need not be changed.

71

Summary: Categories of Data Models

1. **Conceptual** (**high-level**, **logical**) **data models:** Provide concepts that are close to the way many **users perceive data**. (Also called **entity-based** or **object-based** data models.)
2. **Physical** (**low-level**, **internal**) **data models:** Provide concepts that describe details of **how data is stored** in the computer.
3. **Implementation** (**representational**) **data models:** Provide concepts that fall between the above two, balancing user views with some computer storage details.

72

ABSTRACTIONS AND WHEN TO USE A DBMS

73

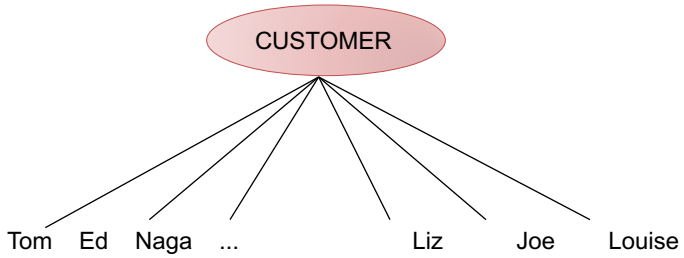
Abstraction: Database Design

- It is very important that the **language** (notations) used for data representation supports abstraction, namely
  1. Classification
  2. Aggregation
  3. Generalization

74

1. Abstraction: Classification

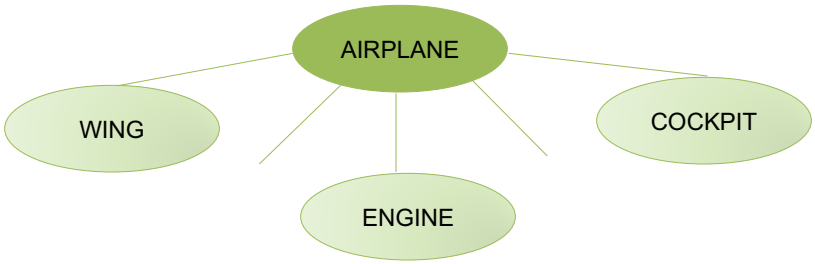
- In a classification we form a concept in a way which allows us to decide whether a given phenomena is a **member of the** concept.



75

2. Abstraction: Aggregation (Part of)

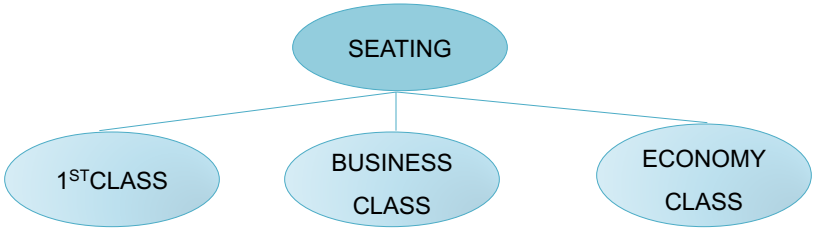
- In an aggregation we form a concept from existing concepts. The phenomena that are members of the new concept's extension are composed of phenomena from **the extensions of the existing concepts**



76

3. Abstraction: Generalization

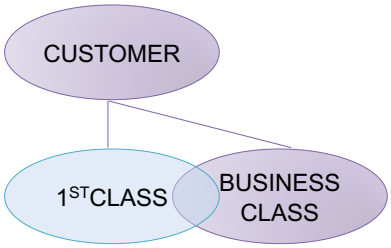
- In a generalization we form a new concept by emphasizing common aspects of existing concepts, **leaving out special aspects**



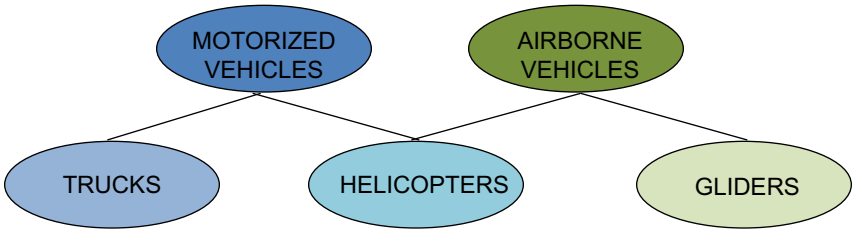
77

3. Generalization (cont.)

Sub-concepts may overlap



Sub-concepts may have multiple super-concepts



78

When to use a DBMS?

1. **Persistent storage of data**
   1. Sharing of data
   2. Data documentation
   3. Backup and Recovery
2. **Centralized control of data**
   1. Control of redundancy
   2. Control of consistency and integrity
3. **Multiple user support**
   1. Control of Access and Security

79

Features of a DBMS

- **Persistent storage**
  - Programs that update, that query, and manipulate data.
  - Data continues to live long after program finishes.
- **Efficient:** do not search entire database to answer a query.
  - **Integrity** refers to the accessed **data** being correct.
- **Convenient:** allow users to query the data as easily as possible
  - **Consistency** refers to the accessed **data** being available.
- **Secure, concurrent, and atomic access**
  - Allow multiple users to access database simultaneously
  - Allow a user access to only authorized data.
  - Provide some guarantee to reliability against system failures.

80

Do not use a DBMS when

- The initial investment in hardware, software, and training is too high.
- Data and applications are simple and stable.
- Real-time requirements cannot be met by the DBMS.
- Multiple user access is not needed.
- The generality a DBMS provides is not needed.
- The overhead for security, concurrency control, and recovery is too high.

81