

CS240

Database Management Systems

Radhika Sukapuram

What is a database ?

- › A collection of **inter-related data** that models some aspect of the real world

What is an example of a database ? Is there a database that you use every day ?

Examples

- › Data regarding academic aspects of IITG students
 - Courses
 - Instructors
 - Registration information of students
 - Grades of students
- › Web-based services (Amazon)
 - Online retailers: order tracking, customized recommendations
 - Online advertisements

Examples (contd.)

› Aadhaar

- Biometric information
- Phone numbers, address

› 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

Examples (contd.)

- › Airlines:

- reservations, schedules

- › Telecommunication:

- records of calls, texts and data usage
- generating monthly bills
- maintaining balances on prepaid calling cards

Databases are everywhere.

What are the common features of these databases ?

Common features

Contains data that is

- › Highly valuable
- › Relatively large
- › Accessed by multiple users and applications, often at the same time.

But is it sufficient to collect all this data ?

But is a database sufficient ?

- › How to read, write, modify data in a database ?
 - Buy from Amazon
 - Book a railway ticket
 - Withdraw money from a bank
 - Make a phone call !

How to do that efficiently and conveniently ?

Database Management System

- › We need a collection of programs
- › A Database Management System (DBMS) is a collection of interrelated data and a set of programs to access that data.

Can't we just store all data in a set of files ?

Data redundancy and inconsistency

Major: CS

Student roll no	Address	Phone No
2018-91	West Fort Road, Nagpur	9012345678

Minor: Mechanical

Student roll no	Address	Phone No
2018-91	West Fort Road, Nagpur	9012345678

› Change address of student 2018-91

-> Higher storage, access cost

-> Data inconsistency

Difficulty in accessing data

- › Find the phone numbers of all students who have CGPA > 8
- › What if there is no application program that calculates this ?
- › Suppose we additionally require that these students have A grades in their DBMS course ?

-> We need more responsive systems for general use

Data isolation

File 1 : CS
Majors

- In comma separated value format
- 2018-91, West Fort Road Nagpur, 9012345678

File 2 :
CGPA

- In XML
- <roll no> 2018-91
<grade>9.2</grade></roll no>

› Files in different formats

-> Writing new application programs is difficult

Integrity problems

- › Every bank account must maintain a minimum balance -
consistency constraint
 - > Developers must add new code to maintain this
 - > The constraint is buried in code, hard to add new constraints
- › No customer must have more than one account and every account must have an Aadhaar number associated with it

Atomicity problem

- › Leela withdraws Rs.1000 from one bank account and deposits it into her mother's account

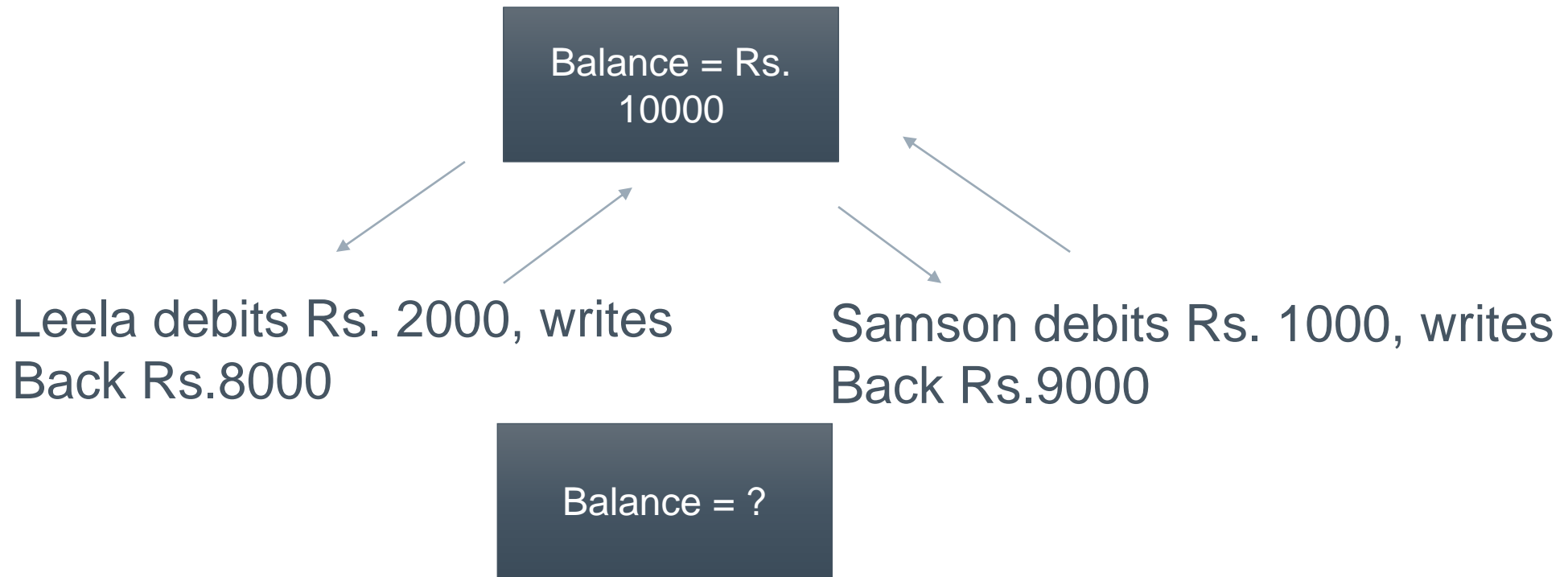
- › The program fails during execution

-> Money is withdrawn, but not deposited !

Funds transfer must be **atomic** – either it must happen in its entirety or not happen at all

Difficult to maintain atomicity in a conventional file system

Concurrent access anomalies



Difficult to provide supervision of programs to prevent this in a conventional file system

Security problems

- › If Aadhaar number is shared with a mobile company, the mobile company must not be able to access personal information
- › A user must not be able to view the accounts of any one else in a bank

-> Difficult to enforce security constraints in a conventional file system

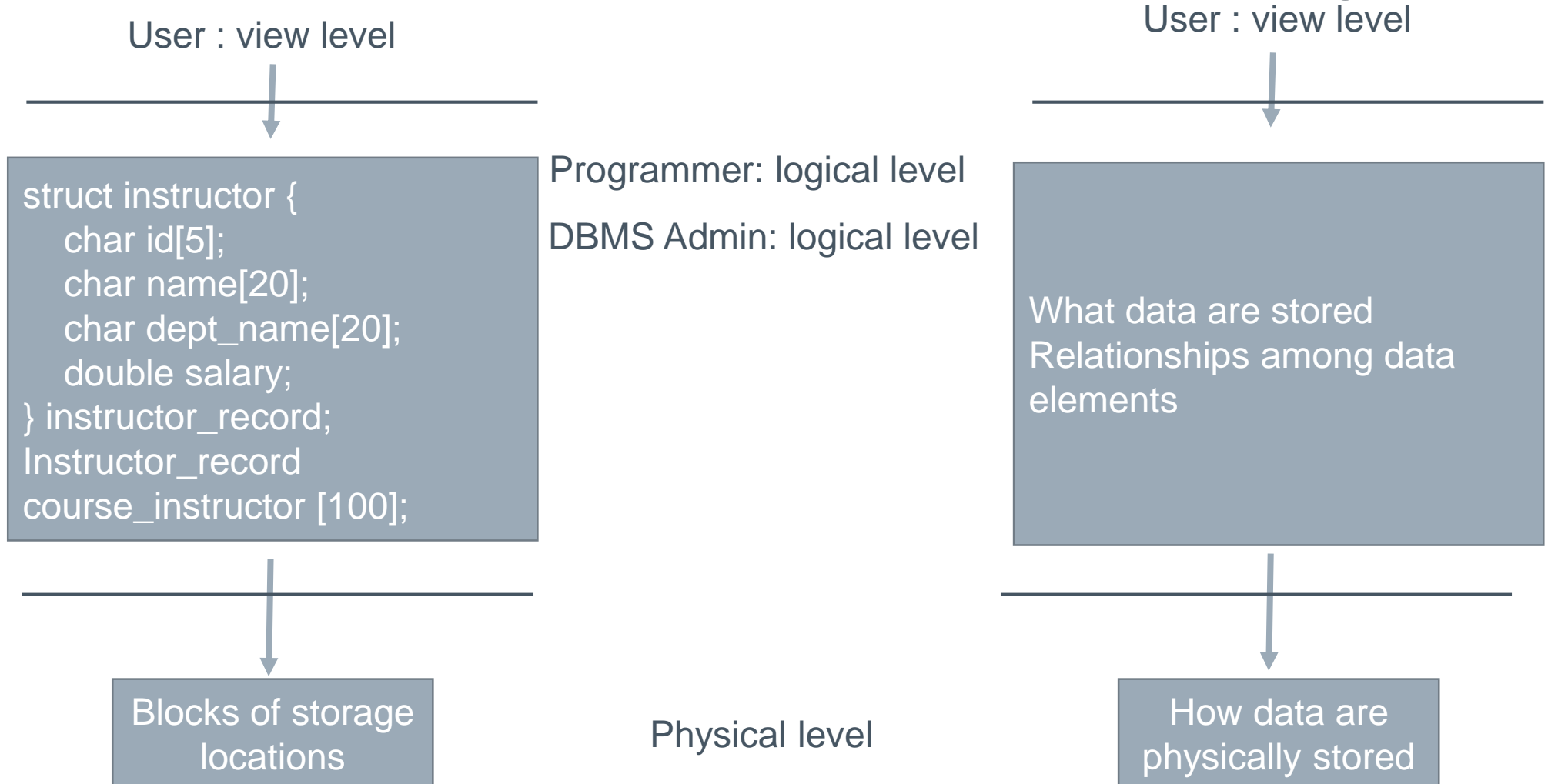
That is why we need a Database **Management System**

What are the common features of DBMSs?

Data Abstraction

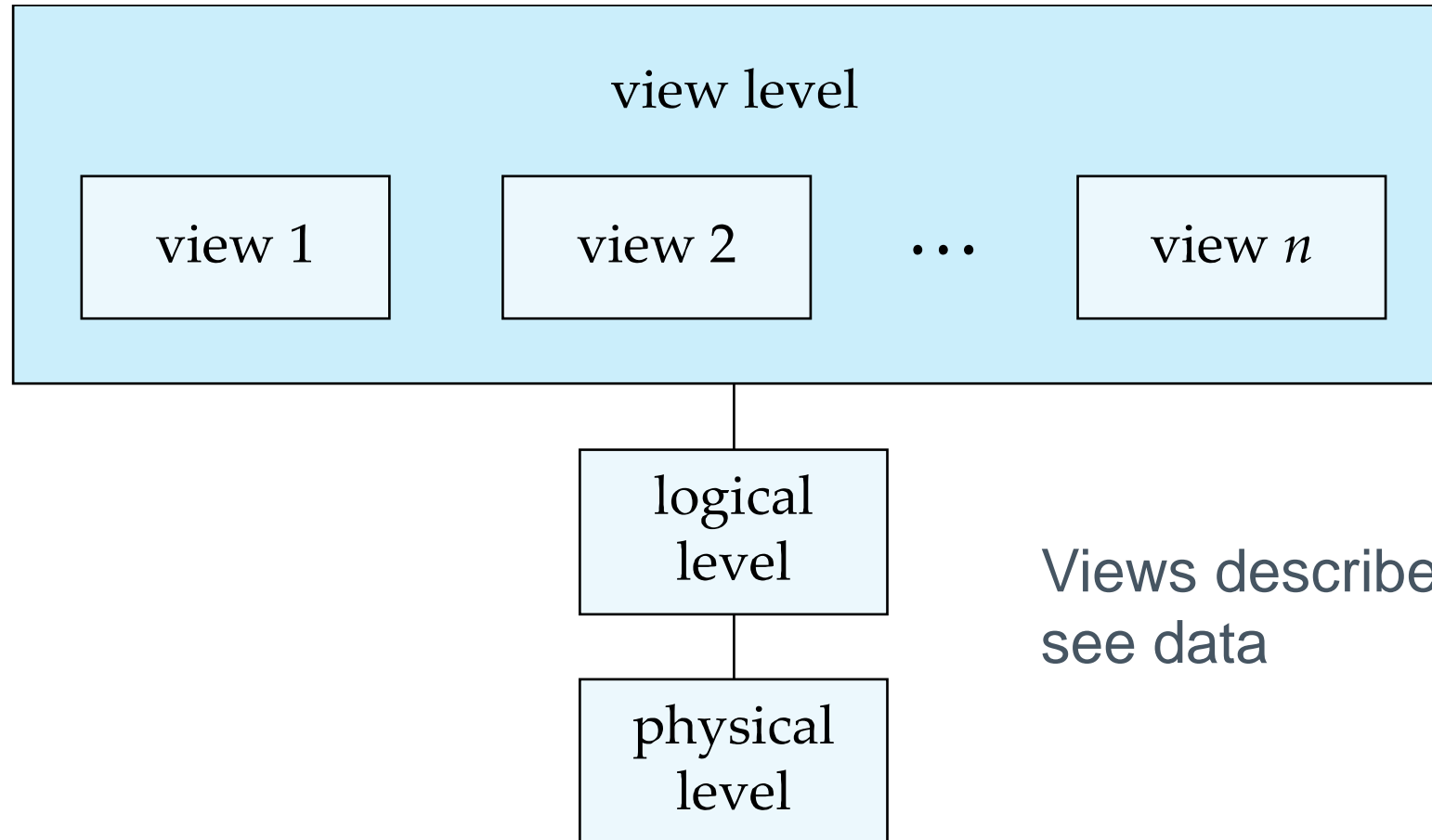
- › A DBMS provides users with an **abstract** view of its data – it hides details of how data is stored and maintained.
- › Other examples of abstractions ?

Data Abstraction : comparison with a program



Each level hides details from the level above and provides an interface

An architecture for a DBMS



Views describe how users see data

Physical data independence

- › Users at the logical level need not know the implementation of physical level structures
- › Provides protection from changes in physical structures

Instances and schemas

PROGRAMS

- › Type definitions and variable declarations
- › Values of variables at an instance

DBMS

- › Database schema : overall design of the database
- › Database instance: collection of information stored in a database at a particular moment

Schema

Subschema : describe different views of the database

Course_info
(cid:string,enrollment:integer)

view level

view 1

view 2

...

view n

logical
level

Logical schema : DB design at logical level

Students(sid: string, name: string,
login: string, age: integer, gpa:real)

Courses(cid: string, cname:string,
credits:integer)

Enrolled(sid:string, cid:string,
grade:string)

**Physical schema : DB design at
physical level**

**Relations stored as unordered
files.**

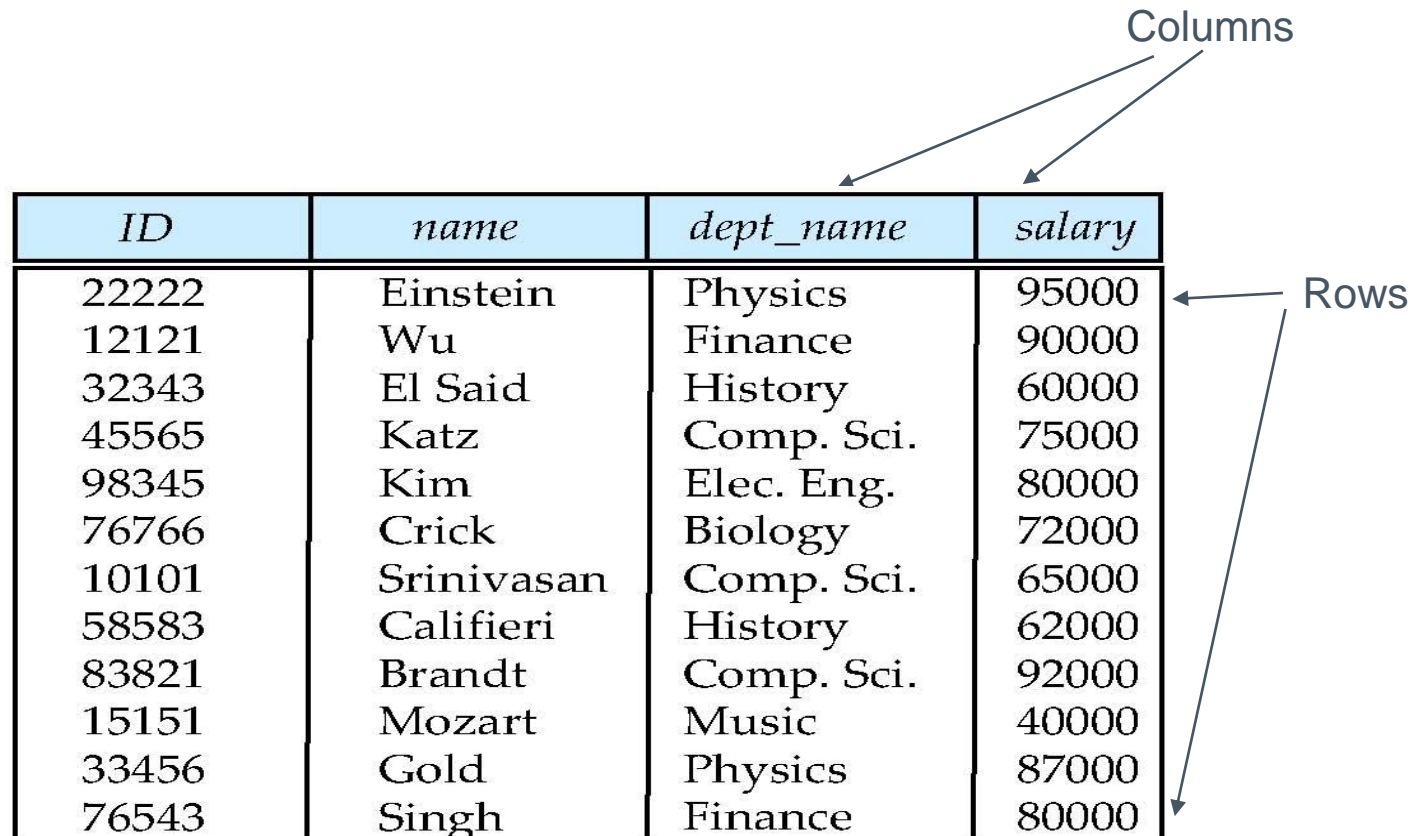
physical level

Data Model

- › A collection of tools that describe
 - Data (Values in Students, Courses etc.)
 - Data Relationships (Enrolled)
 - Data semantics (meaning and interpretation of data, what the data represents)
 - Consistency constraints (account balance must always be greater than 0)

Relational Model

- › All data are stored in tables (also called relations)



The diagram shows a table with four columns and ten rows. Two arrows labeled 'Columns' point to the 'dept_name' and 'salary' headers. Two arrows labeled 'Rows' point to the first and last data rows of the table.

<i>ID</i>	<i>name</i>	<i>dept_name</i>	<i>salary</i>
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

Relational Model

- › Database has several **records** of several **types**
- › Each table (also called **relation**) has records (2222, Einstein, Physics, 95000) of a particular type (instructor)
- › Each record type has a fixed number of fields, called **attributes** (ID, name, department name, salary)

The most widely used model !

Other data models

› Entity – Relationship model

- An entity is a “thing” in the real world that is distinguishable from other things
- Uses a collection of entities and their relationships

› Object-based data model

- Uses a richer type system compared to the relational model
- Extends the E-R model with object-oriented notions

› Semi-structured data model

- Permits specification of data where individual data items of the same type may have different sets of attributes
- Eg: Some items in a store may have an additional attribute called “unit of measure”

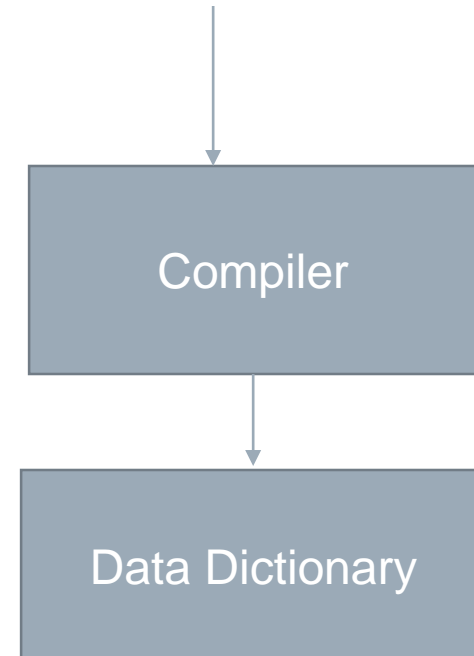
How can we create a database ?

We need a language!

Database Languages

- › **Data definition language**
DDL
- › To create schemas
- › To specify integrity constraints, which are checked every time a DB is updated

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



Some integrity constraints

- › Domain constraints : associating a domain of possible values with every attribute
- › Referential integrity : A *dept_name*, say “CSE, in a *course* record must appear in the *dept_name* attribute of the *department* relation

course	
Course id	Dept_name

department	
Dept_id	Dept_name

- A value that appears in one relation for a given set of attributes also appears in a certain set of attributes in another relation

Some integrity constraints contd.

› Assertions

- Any condition that the DB must always satisfy
- Domain constraints and referential integrity are special forms of assertions
- Many constraints cannot be expressed using these
- Eg: Every department must offer at least five courses every semester

Some integrity constraints contd.

› Authorization

Differentiates users on the type of access permitted to various data

Read authorization

Insert authorization

Update authorization

Delete authorization

May assign all, none or a combination of the above

Is a DDL sufficient ?

How to retrieve, insert, delete or modify data ?

Data Manipulation Languages

- › Retrieval, insertion, deletion and modification of information
- › Types of DMLs
 - Procedural DMLs : user must specify what data are needed and how to get that data
 - Declarative DMLs / Non-procedural DMLs: user must specify what data are needed and NOT how to get that data

Query

- › A statement requesting retrieval of information
- › Eg: Get the names of all students who are enrolled in the Winter semester of 2019
- › Query language: The portion of DML that involves information retrieval

SQL

- › A non-procedural language
- › Consists of both DDL and DML

How can an end user access a database ?

DML Query

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

DML Compiler

Low level
instructions

Query Evaluation
Engine

Application Programs

- › Non-procedural query languages such as SQL are not as powerful as a universal Turing machine.
- › SQL does not support actions such as input from users, output to displays, or communication over the network.
- › Such computations and actions must be written in a **host language**, such as C/C++, Java or Python, with embedded SQL queries that access the data in the database.
- › **Application programs** -- are programs that are used to interact with the database in this fashion.