Chapter 2: Database System Concepts and Architecture

Outline:

2.1 Data Models and their Categories

2.2 Schemas Versus Instances

2.3 DBMS Architecture and Data Independence
- Three-Schema Architecture
- Data Independence

2.4 Database Languages

2.5 Database Users
- System Analysts
- Database Designers
- Application Developers
- Database Administrators
- End Users

2.1 Data Models and their Categories

- Database approach provides some level of ***data abstraction*** by hiding details of data storage that are not needed by most database users.
- A ***data model***: is a collection of conceptual tools for describing data, data relationships, data semantics, and constraints.
- Data models include a set of basic operations for specifying retrievals and updates (insert, delete, and modify) on the database.

Categories of Data Models

1. ***Conceptual (high-level, semantic) data models:***
    - Provide concepts that are close to the way many users perceive data. (Ex: Entity Relationship Model (ER Model)).
        - ER model of real world consists of a collection of basic objects, called entities, and collection of relationships among these objects.
            - Entities: Employee, Department, Project…
            - Relationships between entities: An employee ***works for*** a department. the relationship set ***works for*** associates employees with departments.

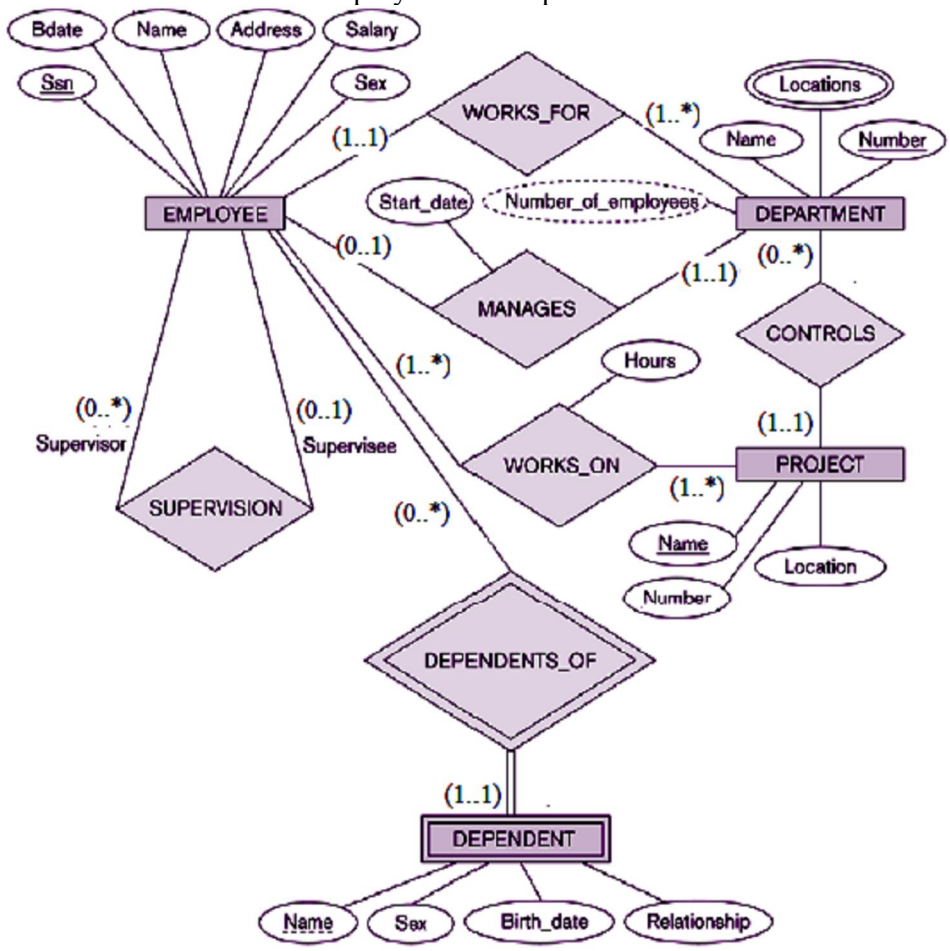
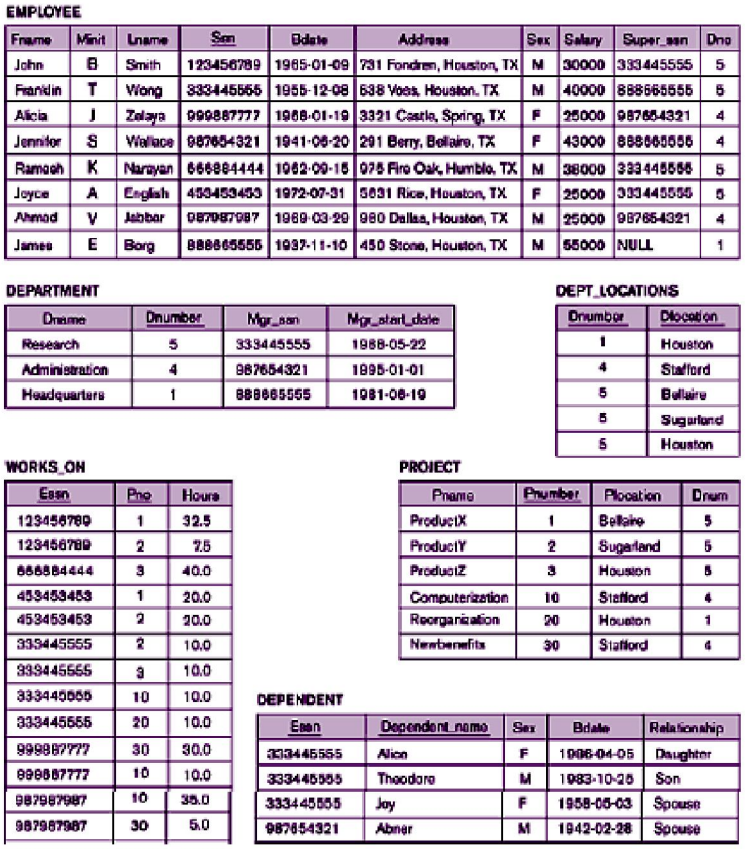


Figure 2.1: ER Model

2. ***Representational (Implementation) data models:***

   - Provide concepts that may be understood by end users but that are not too different from the way data is organized within the computer. (Ex: *relational data* models used in many commercial systems).
     - The relational model uses a collection of tables to represent both data and relationship among those data.
     - Each table has multiple columns, and each column has a unique name.



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse

Figure 2.2: Relational Model

3. ***Physical (low-level, internal) data models:***

   - Provide concepts that describe details of how data is stored in the computer.

2.2 Schemas versus Instance

- A database is divided into ***schemas*** and ***Instances.***

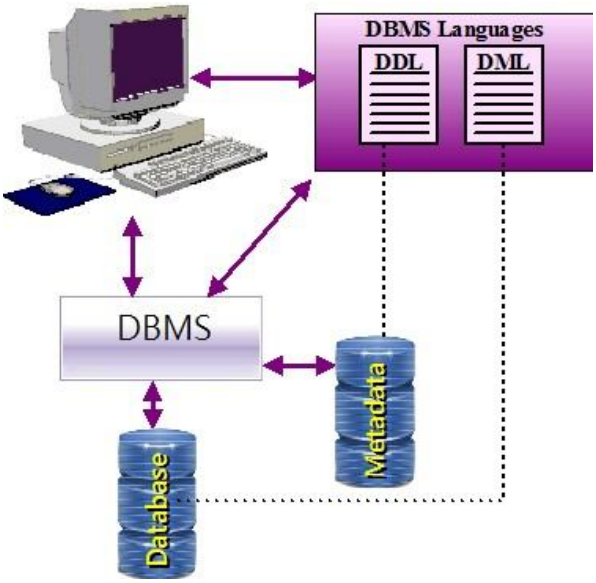


Figure 2.3: Database System

Schema – the logical structure of the database

Ex: the database consists of information about a set of customers and accounts and the relationship between them.

- The ***description*** of a database
- This is specified during database design
- Schemas are changed infrequently
- When we define a new database, we specify the database schema only to the DBMS using DDL language
- The schema is sometimes called the ***intension***,or ***metadata***
- Schema is analogous to type information of a variable in a program
- Schemas are generally stored in a data dictionary (catalog)

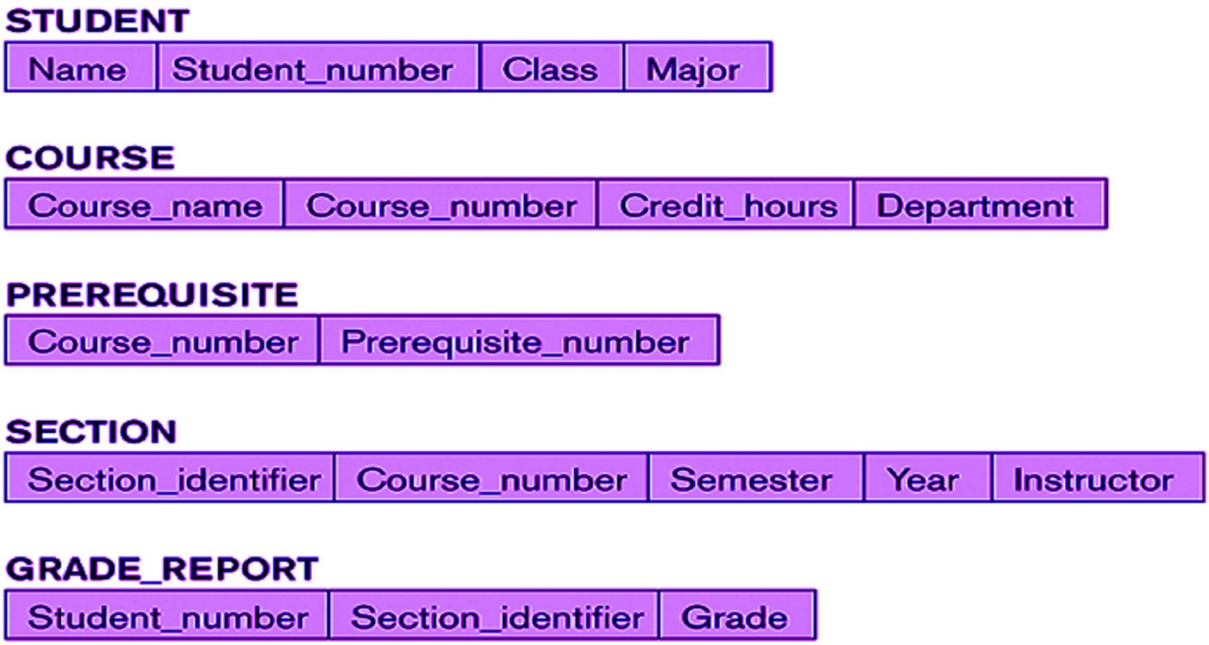


Figure 2.4: An example of the relational schema diagram

Instance – The actual data stored in a database at a *particular moment in time*

- May change quite frequently; change every time data is inserted, deleted, or modified using DML language.
- When we define a new database, we specify the database schema only to the DBMS (the current state of the database is the *empty state* with no data).
- The DBMS is partly responsible for ensuring that every state of the database is a ***valid state*** – that is, a state that satisfies the structure and constraints specified in the schema.
- The instance sometimes called the ***extension*** of schema**, facts**, **states,** *or* **snapshot**.
- Instance is analogous to the value of a variable in a program

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	04	King
92	CS1310	Fall	04	Anderson
102	CS3320	Spring	05	Knuth
112	MATH2410	Fall	05	Chang
119	CS1310	Fall	05	Anderson
135	CS3380	Fall	05	Stone

GRADE_REPORT

Student_number	Section_identifier	Grade
17	112	B
17	119	C
8	85	A
8	92	A
8	102	B
8	135	A

PREREQUISITE

Course_number	Prerequisite_number
CS3380	CS3320
CS3380	MATH2410
CS3320	CS1310

Figure 2.5: An example of the relational instances.

2.3 DBMS Architecture and Data Independence

- ***The Three-Schema Architecture***
    - The goal of the *three-schema architecture* is to separate the user applications and the physical database.
    - It proposed to support DBMS characteristics of:
        - Insulation of programs and data (program-data and program-operation independence).
        - Support of multiple user views of the data.
        - Use of a catalog to store the database description (schema).

- Database systems have several schemas, partitioned according to the levels of abstraction.
- Schemas can be defined at the following three levels:
  1. The *internal level* (physical) has an *internal schema* that describes the physical storage structure of the database. The internal schema uses a physical data model and describes the complete details of data storage and access paths for the database.
  2. The *conceptual level* (logical) has a *conceptual scheme* is a high-level description of the whole database. The conceptual schema hides the details of physical storage structures and concentrates on describing entities, data types, relationships, user operations and constraints. A high-level data model or an implementation data model can be used at this level.
  3. The *external* or *view level* include a number of *external schemas* or *user views* that describe the view of different user groups. High-level data model or an implementation data model can be used at this level.

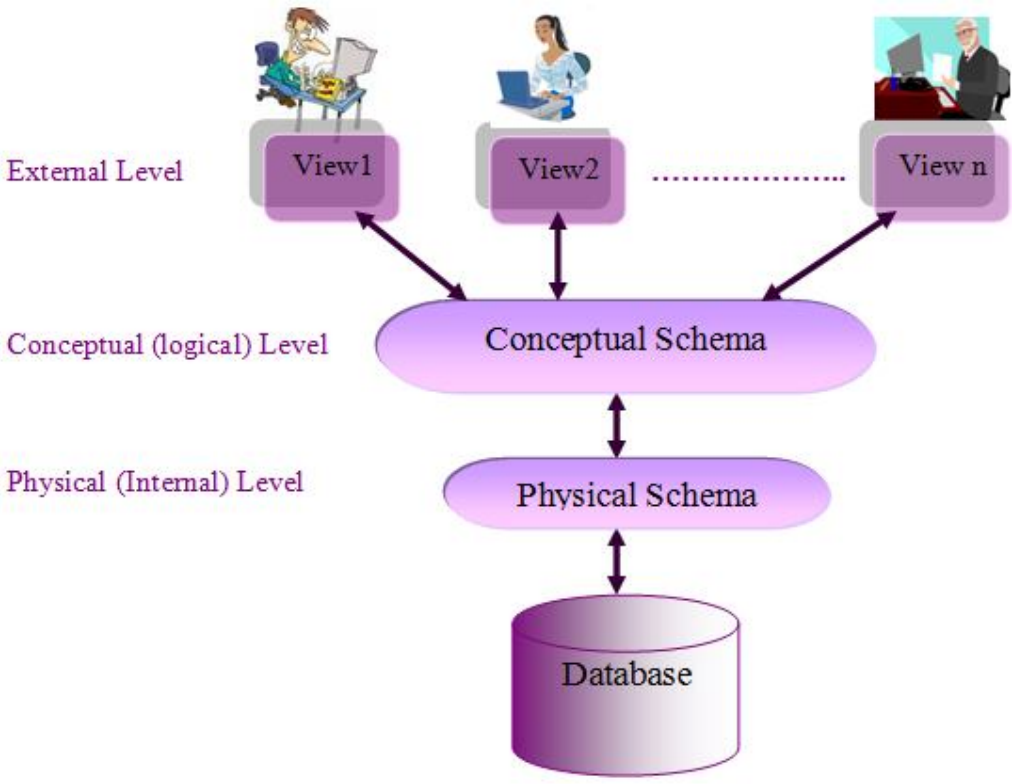


Figure 2.6: Three Schema Architecture.

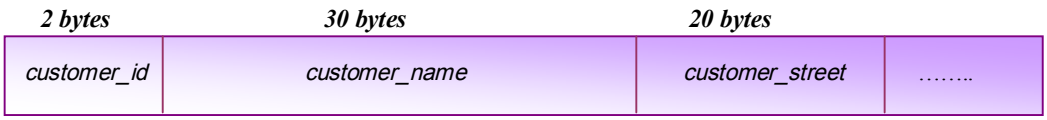
- ***Example (Bank DB)***:
  - a banking may have several record types, including:

- Customer, with customer-name, social-security…. fields
- Account, with account-name, balance ….. fields
- Employee, with employee-name, salary …. fields

- In a C++ language, we may declare a Customer record as follows:

```
struct customer
{
 int customer_id;
char customer_name[30];
char customer_street[20];
……..
};
```

- ***At internal level***, a customer record can be described as a block of consecutive storage locations (for example, words or bytes).



- The language compiler hides this level of detail from programmers, but DBA may be aware of certain details of the physical organization of the data.

- ***At the conceptual level***, each such record is described by a type definition, and the interrelationship among these record types is defined.,as in the previous C++ code segment. Programmers work at this level of abstraction and DBA usually work at this level of abstraction.

- ***At the external or view level***, users see a set of application programs that hide details of the data types. In addition to hiding details of the conceptual level of the database, the views also provide a security mechanism to prevent users from accessing parts of the database. For example, tellers in a bank see only that part of the database that has information on customer accounts; they cannot access information concerning salaries of employees.

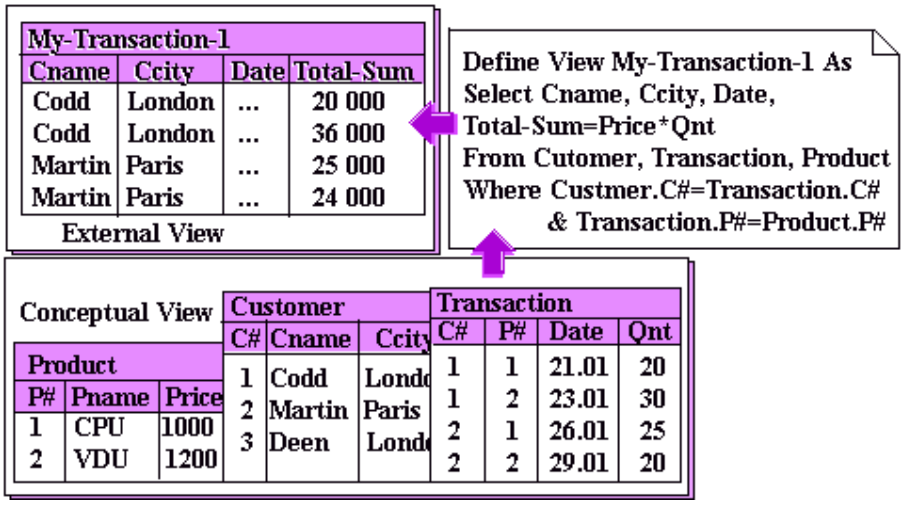


Figure 2.7: External and conceptual views

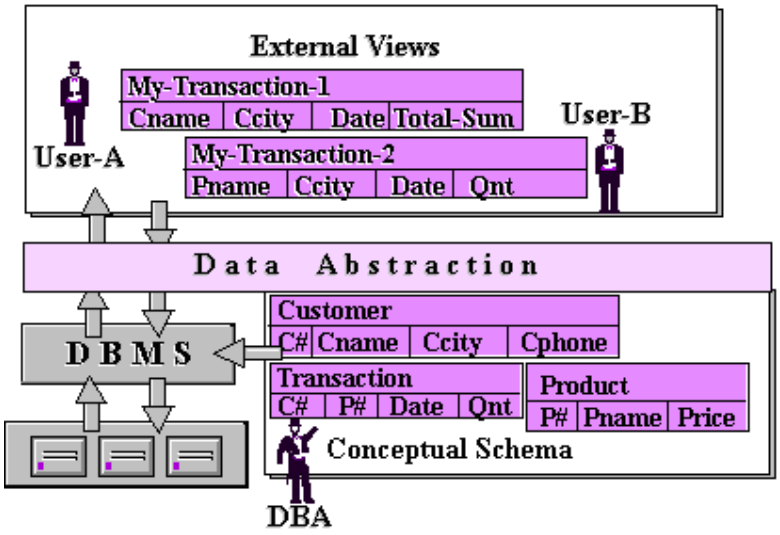


Figure 2.8: Multiple Views of Data

- ***Data Independence***
    - The three-schema architecture can be used to explain the concept of data independence, which can be defined as the capacity to change the schema at one level of a database system without having to change the schema at next higher level.

- We can define two types of data independence:
  1. *Logical data independence* - the capacity to change the conceptual scheme without having to change external schemas or application programs.
     - Change the conceptual schema to expand the database (by adding a record type or data item), or to reduce the database (by removing a record type or data item). In the latter case, external schemas that refer only to the remaining data should not be affected.
  2. *Physical data independence* - the capacity to change the internal scheme without having to change the conceptual (or external) schemas. Changes to the internal schema may be needed because some physical files had to be reorganized. for example, reorganize physical files to improve performance.

- *For example*:

  UNIVERSITY Conceptual Schema
  STUDENT (Name, Student Number, Class, Major)
  COURSE (Course Name, Course Number, Credit, Dept)
  PREREQUISITE (Course Number, Prerequisite Number)
  SECTION (Section Id, Course Number, Semester, Year, Instructor)
  GRADE_REPORT(Student Number, Section Id , Grade)

  UNIVERSITY External Schema
  TRANSCRIPT(Student Name, Course Number, Grade, Semester, Year, Section Id)
  derived from STUDENT, SECTION, GRADE_REPORT

  PREREQUISITES(Course Name, Course Number, Prerequisites)
  derived from PREREQUISITE, COURSE

2.4 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.
- DDL and DML not two separate languages; instead they simply form parts of a single db language, such as SQL language.

1. A Data Definition Language (DDL)

- A language that is used to define database schemas.
- The DDL statement is used to identify description of the schema construct and store the schema description in the DBMS catalog (data dictionary).
  - A data dictionary contains metadata (i.e., data about data).
  - The schema of a table is an example of metadata.
- DDL example:

  Create table Student (StNo number(14),name varchar(20),Bdate date);

  - When create a new table, it also update a special set of tables (data dictionary).

- DDL used by the DBA and by database designer to define the conceptual schema.

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

Figure 2.10: An example of a database catalog for the database.

2. A Data Manipulation Language (DML)

- A DML language is used to manipulate data.
  - Retrieve information stored in the db.
  - Insert new information into the db.
  - Delete information from the db.
  - Modify (Update) information stored in the db.

- There are two classes of DMLs:
  - ***A high-level or nonprocedural DML***:
    - User specifies what data is required without specifying how to get those data.
    - High-level DMLs, such as SQL, can specify and retrieve many records in a single DML statement (called set-at-a-time or set-oriented).
    - It can be embedded in a host programming language (DML with Java, VB, C++, etc..), or it can be used as a stand-alone query language.
    - Also called **declarative** languages.
  - ***A low-level or procedural DML:***
    - user specifies what data is required and how to get those data
    - This type of DML typically retrieves individual records or objects from the database (called record-at-a-time or record-oriented) such as PL/SQL.

3. ***A data control language (DCL):***
   - Several commands for access control, DBMS administration, etc

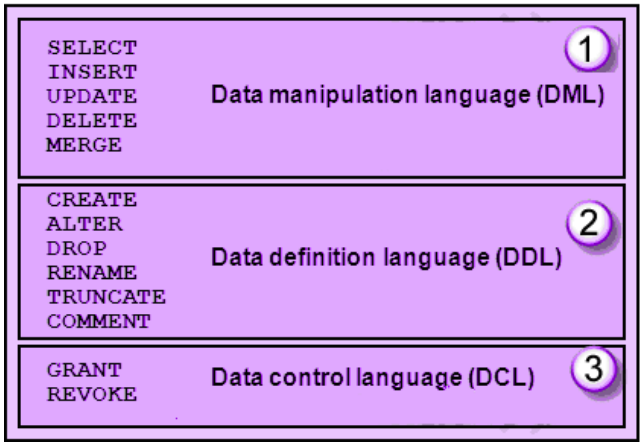


Figure 2.11: Database Languages.

2.5 Database Users

1. ***System Analysts***
   - communicate with each prospective database user group in order to understand its
     - information needs
     - processing needs
   - develop a specification of each user group's information and processing needs
   - develop a specification integrating the information and processing needs of the user groups
   - document the specification

2. ***Database Designers***
   - choose appropriate structures to:
     - represent the information specified by the system analysts
     - store the information in a normalized manner in order to guarantee integrity and consistency of data
   - DBA and database designer define the schema
   - document the database design

3. ***Application Developers (Programmer)***
   - implement the database design
   - implement the application programs to meet the program specifications
   - test and debug the database implementation and the application programs
   - document the database implementation and the application programs

   - Note: Application programmers implement these specifications and programs. Such analysts and programmers nowadays called *software engineers*

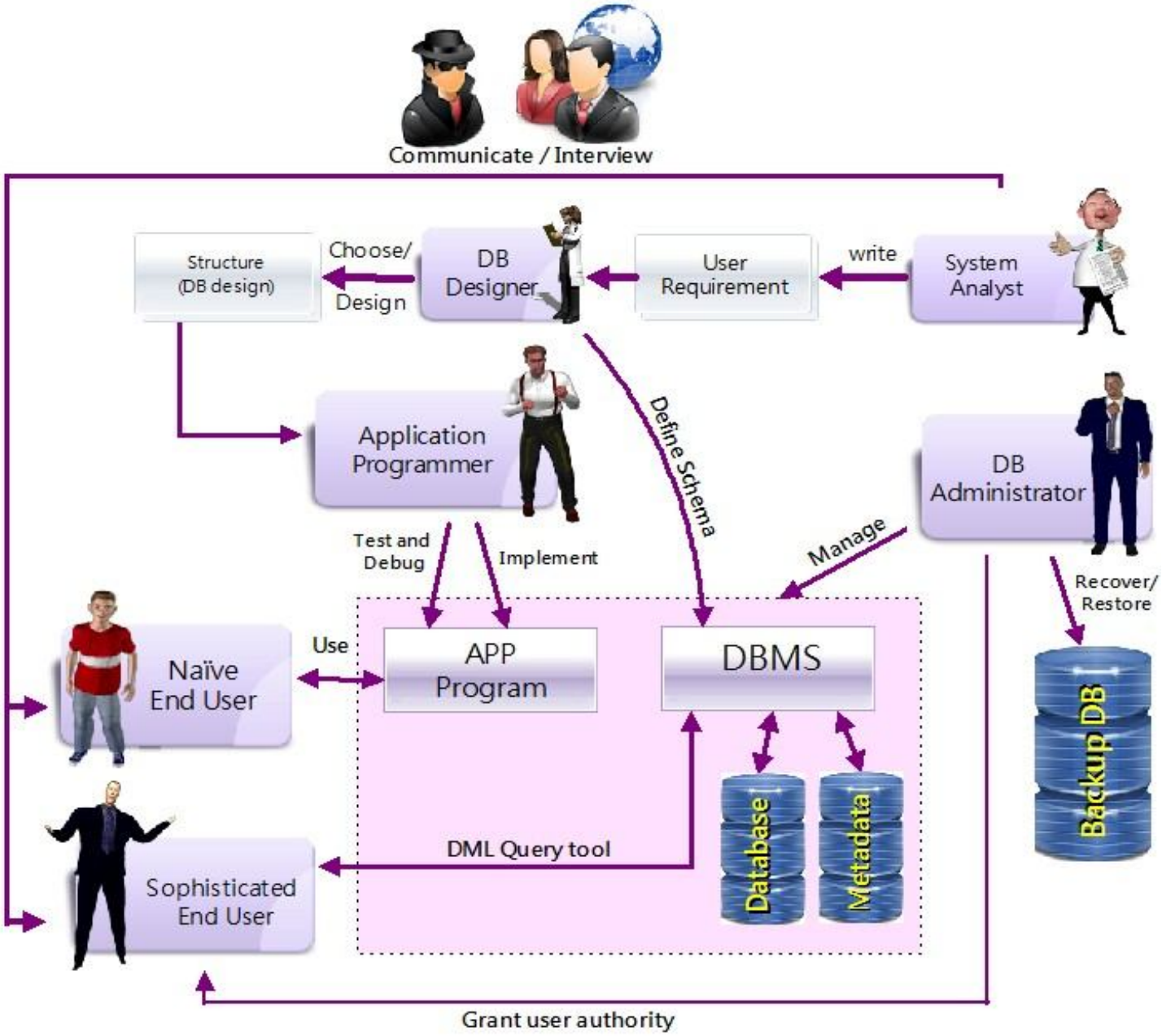
4. ***Database Administrators***
   - One of the main reasons for using DBMSs is to have central control of both the data and the programs that access those data. A person who has such central control over the system is called a **database administrator** (**DBA**). The functions of a DBA include:
     - **Schema definition**. The DBA creates the original database schema by executing a set of data definition statements in the DDL.
     - **Storage structure and access-method definition**.
     - **Schema and physical-organization modification**. The DBA carries out changes to the schema and physical organization to reflect the changing needs of the Organization, or to alter the physical organization to improve performance.
     - **Granting of authorization for data access**. By granting different types of authorization, the database administrator can regulate which parts of the database various users can access. The authorization information is kept in a special system structure that the database system consults whenever someone attempts to access the data in the system.
     - **Routine maintenance**. Examples of the database administrator's routine maintenance activities are:

- Periodically backing up the database, either onto tapes or onto remote servers, to prevent loss of data in case of disasters such as flooding.
- Ensuring that enough free disk space is available for normal operations, and upgrading disk space as required.
- Monitoring jobs running on the database and ensuring that performance is not degraded by very expensive tasks submitted by some users.

5. ***End Users***
   - People whose jobs require access to the database for querying, updating, and generating

   1. **Sophisticated users –** interact with the system without writing programs. Instead,they form their requests in a database query language. They submit each such query to a query processor, whose function is to break down DML statements into instructions that the storage manager understands.

   2. **Naïve users** – invoke one of the permanent application programs that have been written previously. Ex: people accessing database over the web, bank tellers, clerical staff


Communicate / Interview
Structure (DB design)
Choose/ Design
DB Designer
User Requirement
write
System Analyst
Application Programmer
Define Schema
DB Administrator
Test and Debug
Implement
Manage
Recover/ Restore
Naïve End User
Use
APP Program
DBMS
Backup DB
Database
Metadata
DML Query tool
Sophisticated End User
Grant user authority