

Database Management System (BCSC-1003)

Topic: **Introduction to DBMS**



Er. Priya Pandey

Assistant Professor, Dept. of CEA, GLA University,

Mathura

Content



- Difference between Schema and Instance in DBMS
- Data Independence
- Database Models
- DBMS Vs RDBMS

Difference between Schema and Instance in DBMS

Difference between Schema and Instance in DBMS



Instances:

- Instances are the collection of information stored at a particular moment.
- The instances can be changed by certain CRUD (Create, Read, Update, Delete) operations as like addition, deletion of data.

Example –

Let's say a table TEACHER in our database whose name is SCHOOL, suppose the table has 50 records so the instance of the database has 50 records for now and tomorrow we are going to add another fifty records so tomorrow the instance have total 100 records. This is called an instance.

Difference between Schema and Instance in DBMS



Schema :

- Schema is the overall description of the database.
- The basic structure of how the data will be stored in the database is called schema.

Example –

Let's say a table named TEACHER in our database name SCHOOL, the TEACHER table require the name, doj, dob in the table so we design a structure as :

TEACHER

name: varchar doj: date dob: date

Above given is the schema of the table TEACHER.

Data Independence

Data Independence



- Data independence can be explained using the three-schema architecture.
- Data independence refers characteristic of being able to modify the schema at one level of the database system without altering the schema at the next higher level.
- There are two types of data independence:
 - (1) Logical Data Independence and
 - (2) Physical Data Independence

Data Independence

Logical Data Independence:

- Logical data independence refers characteristic of being able to change the conceptual schema without having to change the external schema.
- Logical data independence is used to separate the external level from the conceptual view.
- If we do any changes in the conceptual view of the data, then the user view of the data would not be affected.
- Logical data independence occurs at the user interface level.

Data Independence

Physical Data Independence:

- Physical data independence can be defined as the capacity to change the internal schema without having to change the conceptual schema.
- If we do any changes in the storage size of the database system server, then the conceptual structure of the database will not be affected.
- Physical data independence is used to separate conceptual levels from the internal levels.
- Physical data independence occurs at the logical interface level.

Database Models

Database Models

- Data Model gives us an idea that how the final system will look like after its complete implementation.
- It defines the data elements and the relationships between the data elements.
- Data Models are used to show how data is stored, connected, accessed and updated in the DBMS.
- Some of the Data Models in DBMS are:

Hierarchical Model, Network Model, Entity-Relationship Model, Relational Model, Flat Data Model, Object-Oriented Data Model, Context Data Model, Associative Data Model, Object-Relational Data Model, Semi-Structured Data Model etc.

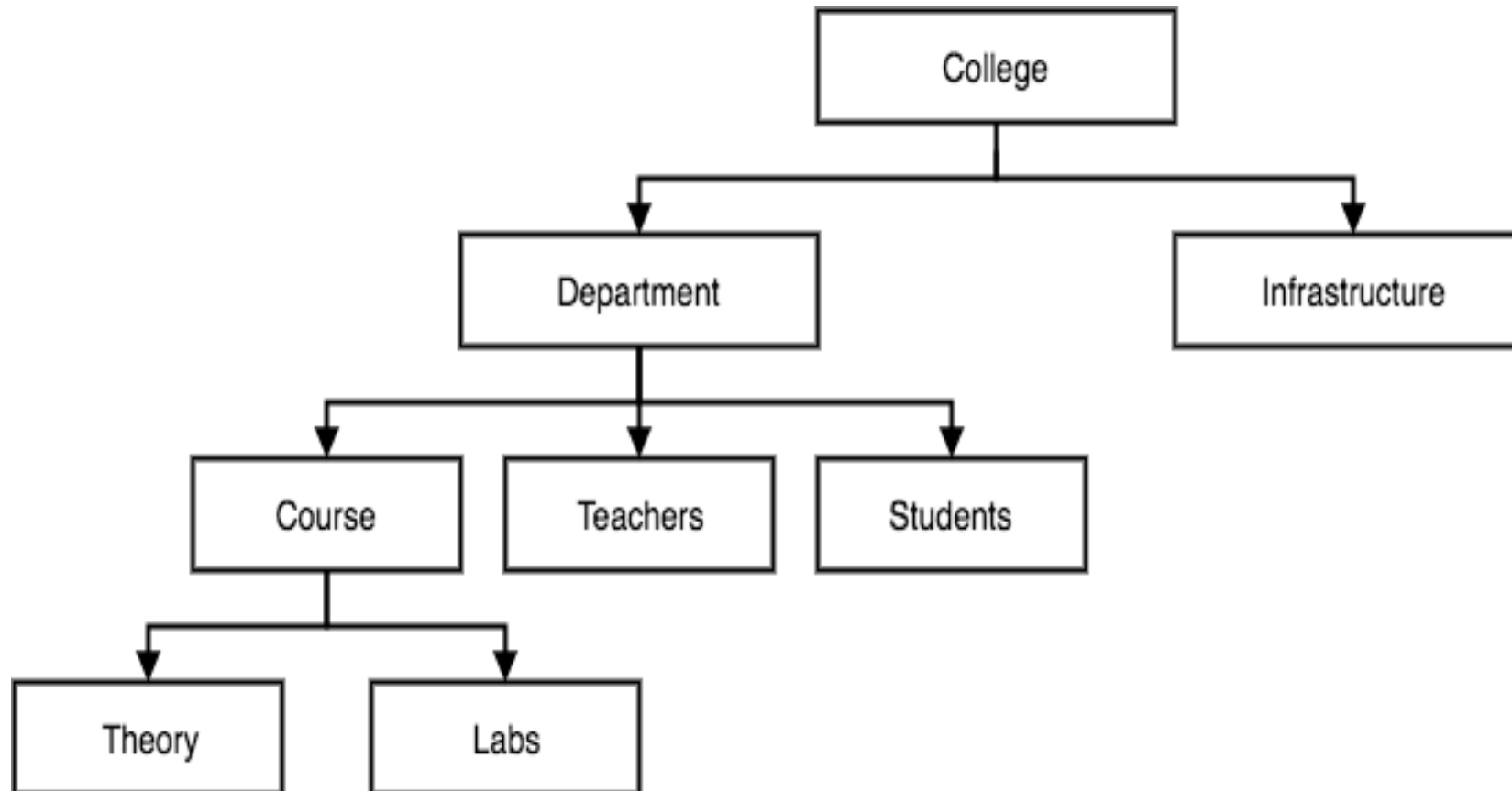
Database Models

Hierarchical Model:

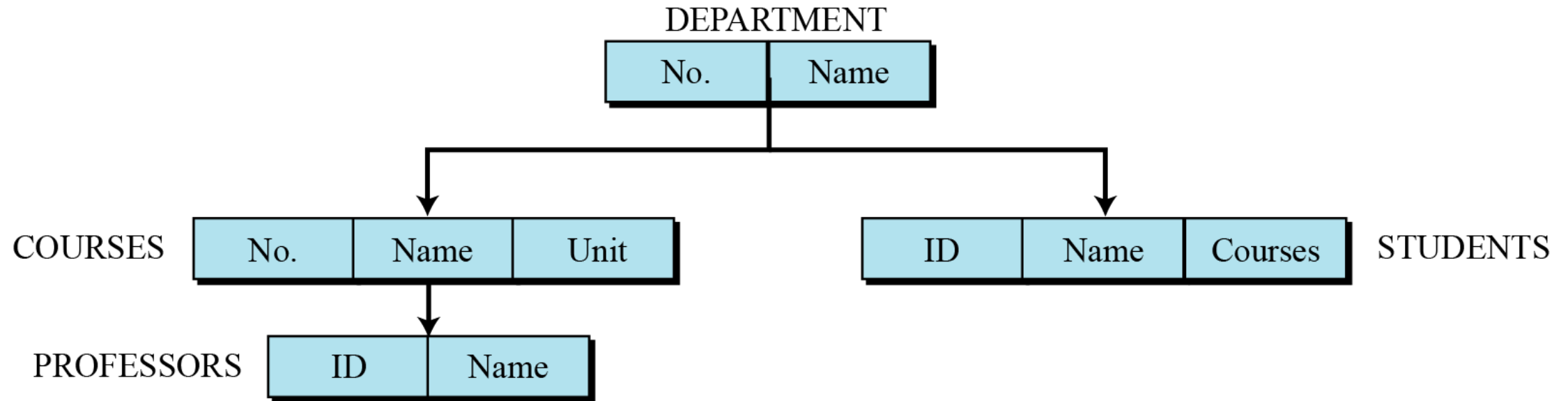
- Hierarchical Model was the first DBMS model.
- This model organizes the data in the hierarchical tree structure.
- The hierarchy starts from the root which has root data and then it expands in the form of a tree adding child node to the parent node.
- This model easily represents some of the real-world relationships like hierarchy of any organization, sitoman

Database Models

Example:



An example of the hierarchical model representing a university



Database Models

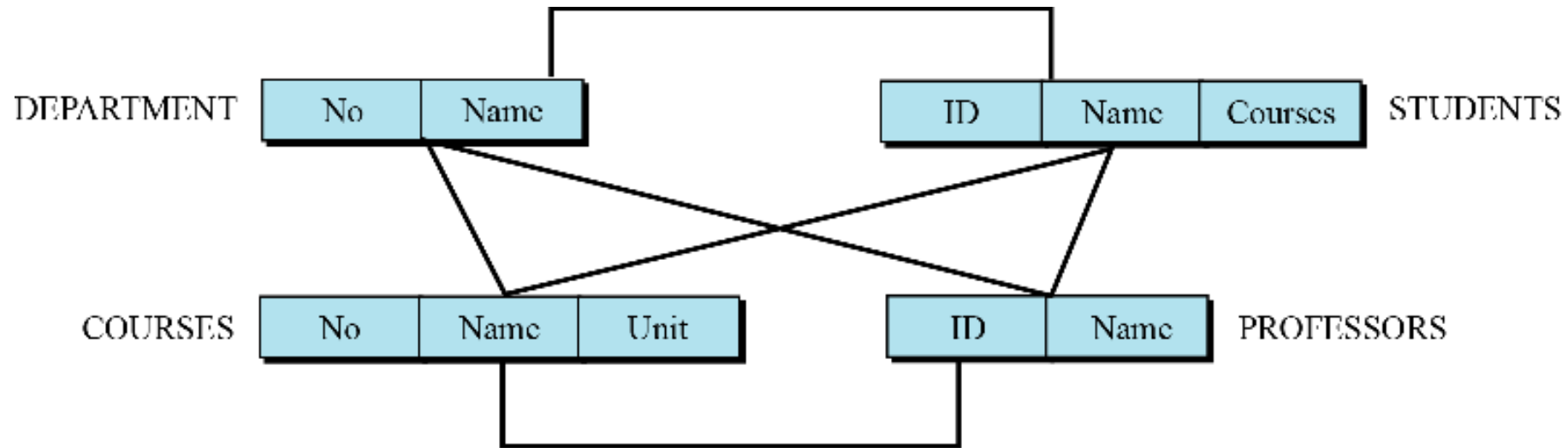


Network Model

- This model is an extension of the hierarchical model.
- It was the most popular model before the relational model.
- This model is the same as the hierarchical model, the only difference is that a record can have more than one parent.
- It replaces the hierarchical tree with a graph.

Network database model

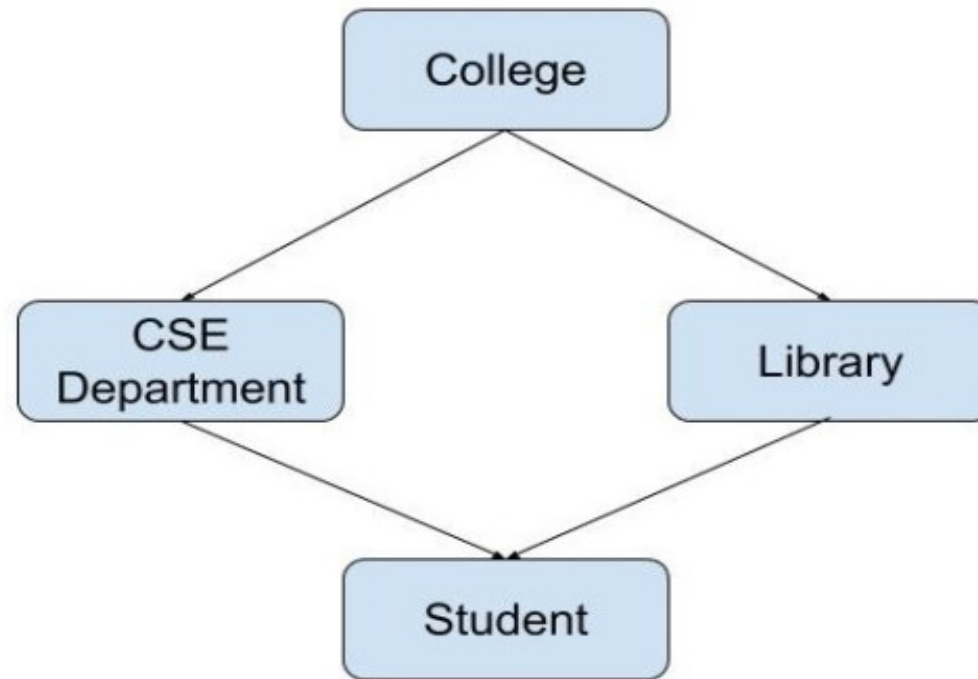
In the network model, the entities are organized in a graph, in which some entities can be accessed through several paths (Figure).



An example of the network model representing a university

Database Models

Example: In the example below we can see that node student has two parents i.e. CSE Department and Library. This was earlier not possible in the hierarchical model.



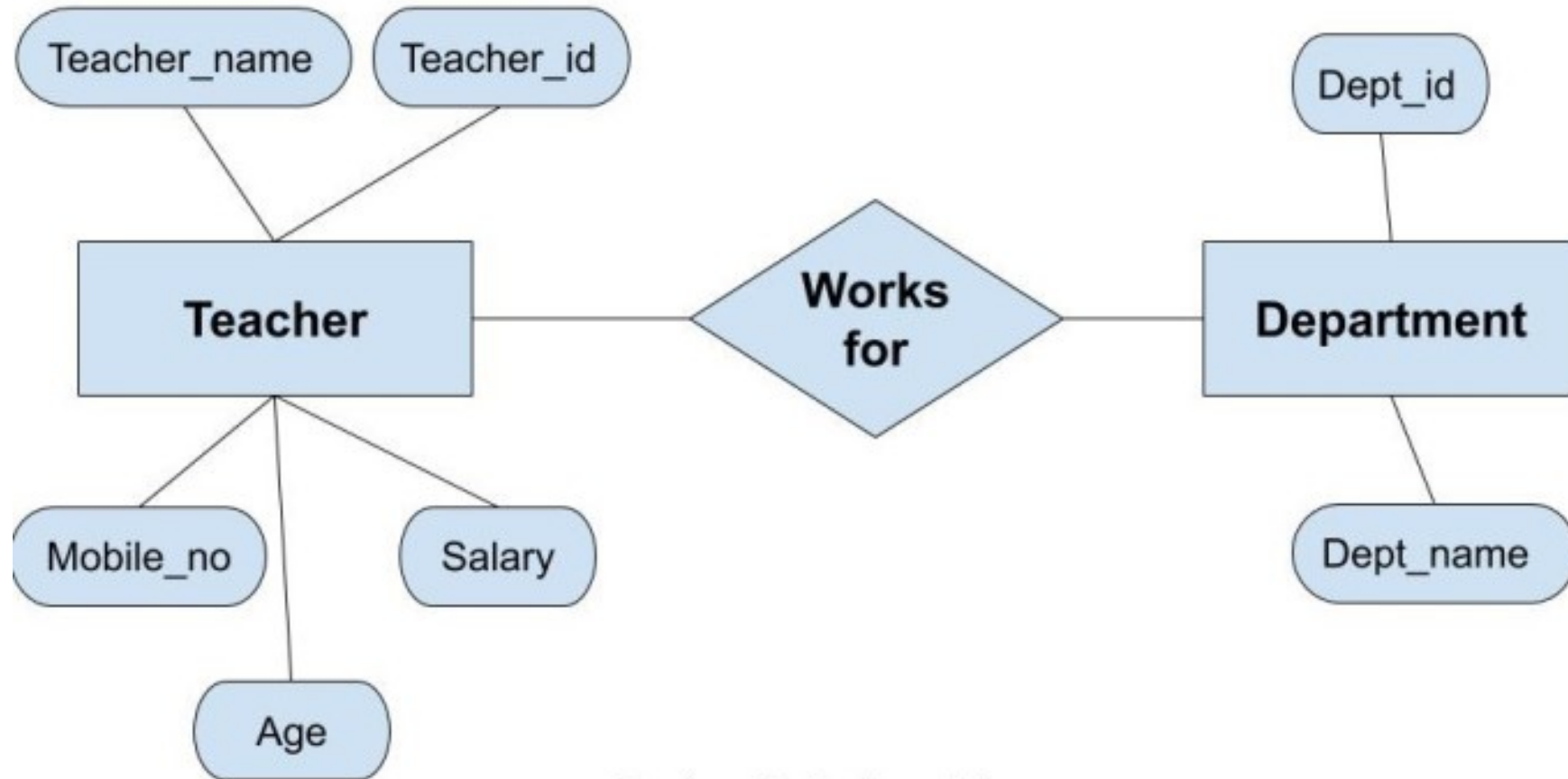
Network Model

Database Models

Entity-Relationship Model

- Entity-Relationship Model or simply ER Model is a high-level data model diagram.
- In this model, we represent the real-world problem in the pictorial form to make it easy for the stakeholders to understand.
- It is also very easy for the developers to understand the system by just looking at the ER diagram.
- We use the ER diagram as a visual tool to represent an ER Model.
- ER diagram has the following three components: Entity, Attribute and Relationship.

Database Models



Entity-Relationship
Model

Database Models

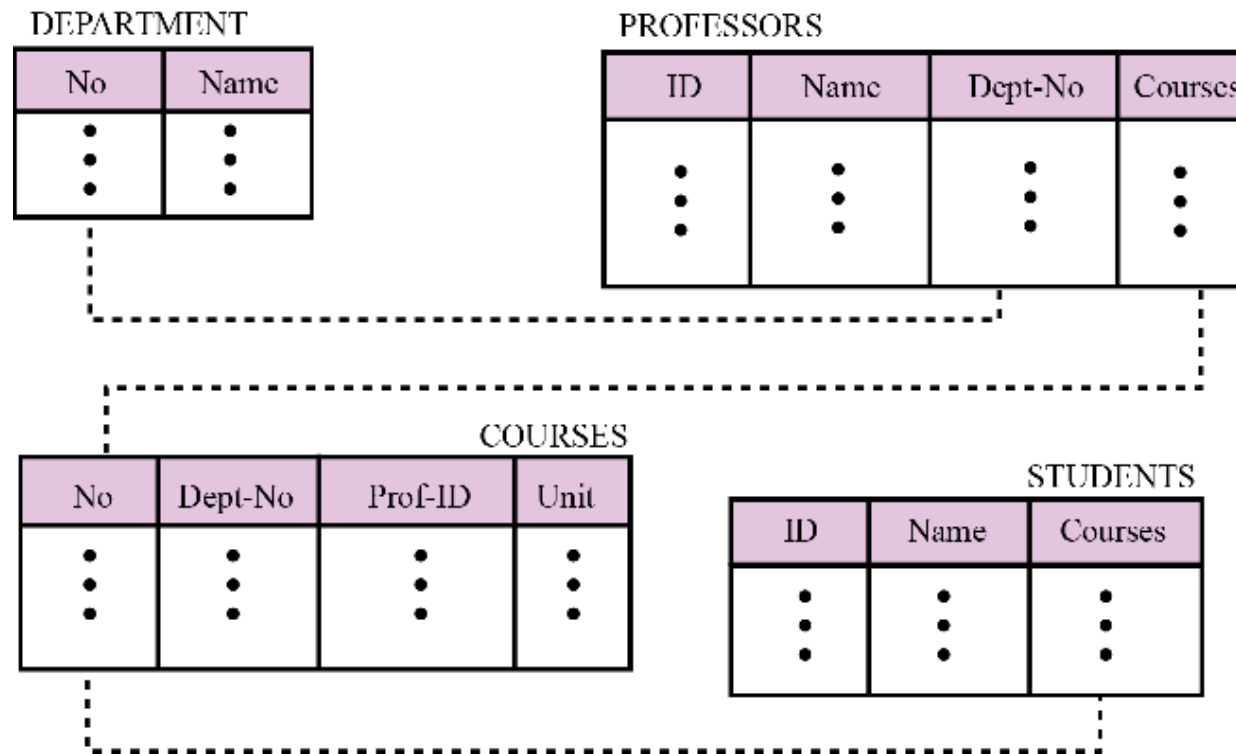


Relational Model

- This model was introduced by Dr. E. F. Codd in 1970.
- Relational Model is the most widely used model.
- In this model, the data is maintained in the form of a two-dimensional table.
- All the information is stored in the form of rows and columns.
- The basic structure of a relational model is tables.
- So, the tables are also called relations in the relational model.

Relational database model

In the relational model, data is organized in two-dimensional tables called relations. The tables or relations are, however, related to each other.



An example of the relational model representing a university

Database Models

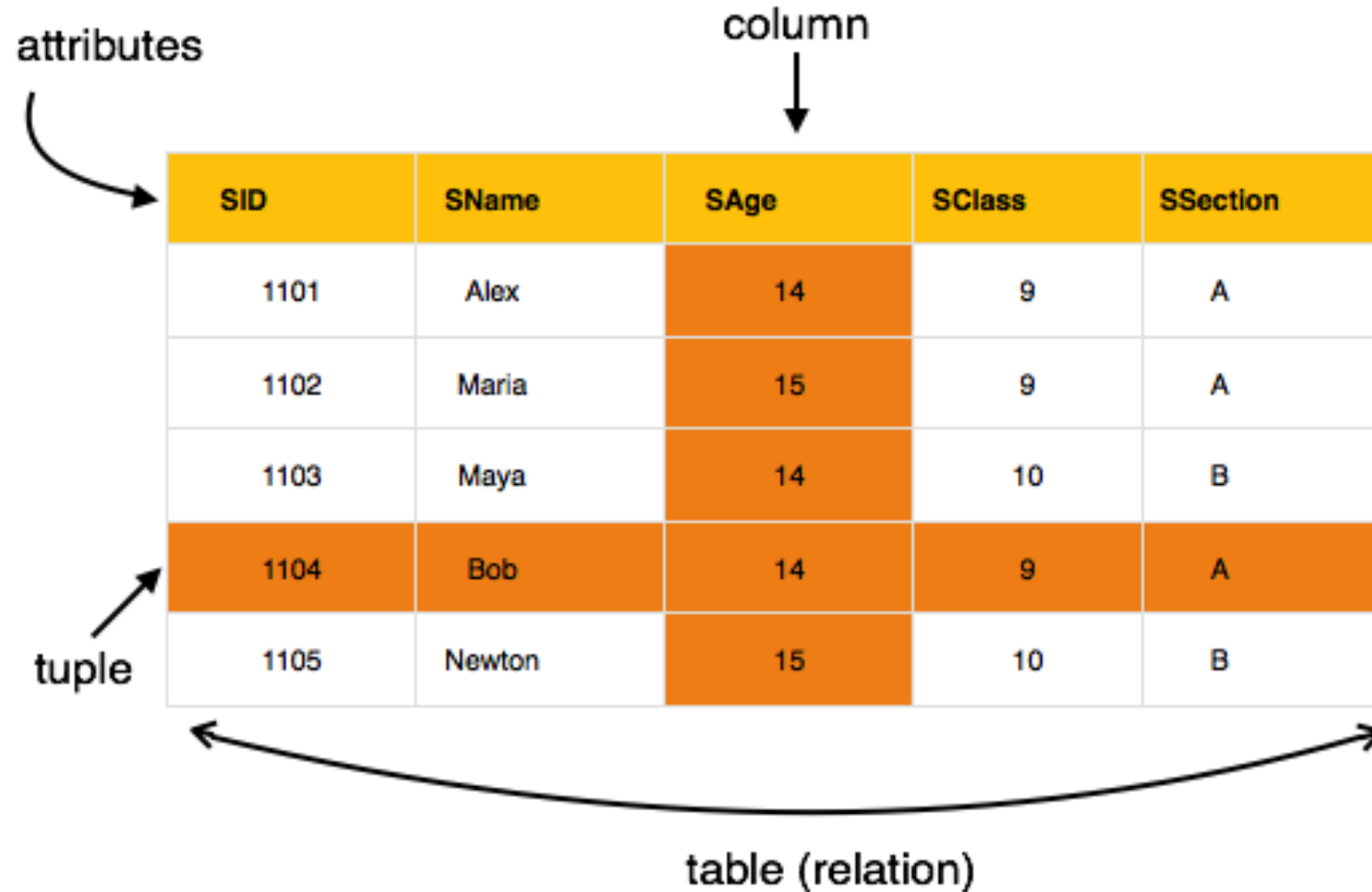
attributes

column

SID	SName	SAge	SClass	SSection
1101	Alex	14	9	A
1102	Maria	15	9	A
1103	Maya	14	10	B
1104	Bob	14	9	A
1105	Newton	15	10	B

tuple

table (relation)



Object Oriented Model

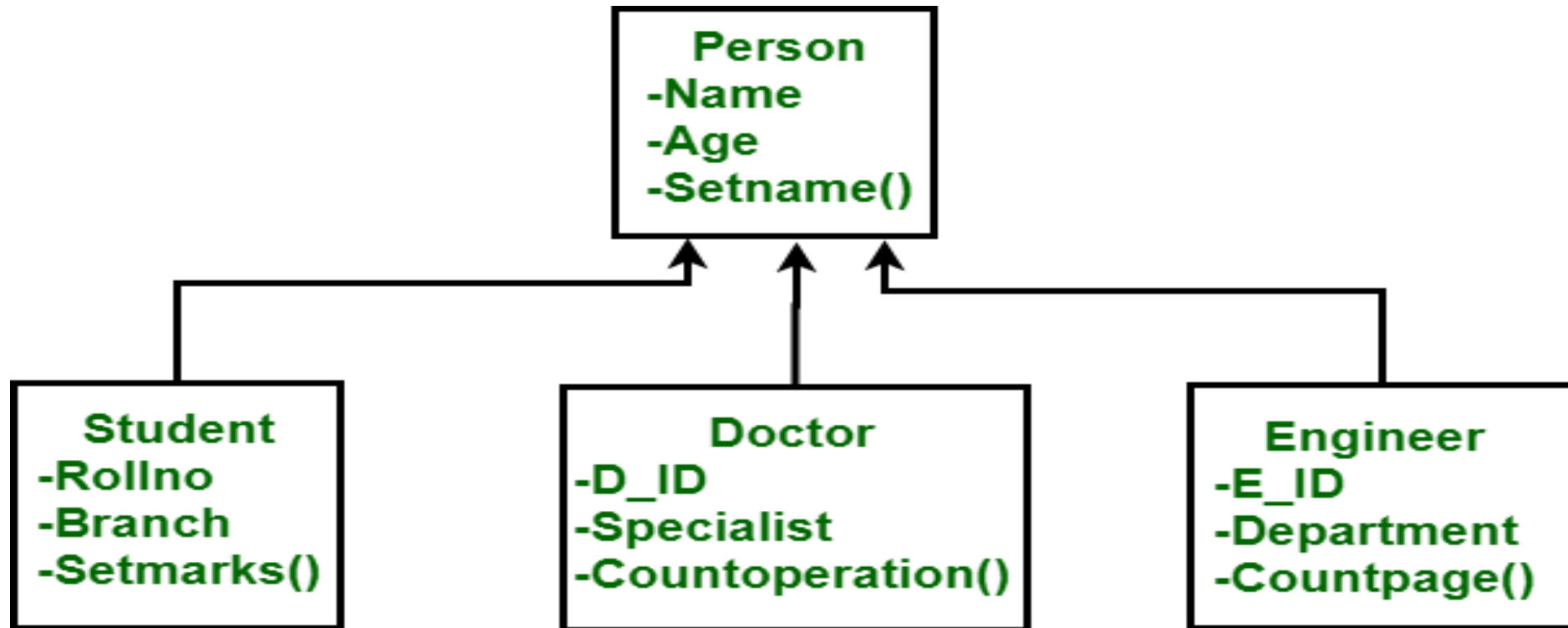
- This is extended ER model
- An object is identified as a real life identity that captures distinguishable properties and behavior.
- Based on data abstraction.
- A data model that is closely similar to the real world was required to reflect the complex real world challenges.
- Represents data using class diagram and methods to represents class, method, object and inheritance.
- Object-based model=E-R model+OO features
- Object-relational data model=object-based model+Relational model

Object Oriented Data Model :

- Data and their relationship are contained in a single structure known as a "object" in an object-oriented data model.
- In this, real world problems are represented as objects with different attributes. All objects have multiple relationships between them. Basically, it is combination of Object Oriented programming and Relational Database Model

Object Oriented Data Model = Combination of Object Oriented Programming + Relational database model

Components of Object Oriented Data Model :



Description of the figure–

Objects:An object is an abstraction of a real world entity or we can say it is an instance of class. Objects encapsulates data and code into a single unit which provide data abstraction by hiding the implementation details from the user. For example: Instances of student, doctor, engineer in the previous figure.

Attribute –

An attribute describes the properties of object. For example: Object is STUDENT and its attribute are Roll no, Branch, Setmarks() in the Student class.

Description

Methods –

Method represents the behavior of an object. Basically, it represents the real-world action. For example: Finding a STUDENT marks in previous figure as Setmarks().

Class –

A class is a collection of similar objects with shared structure i.e. attributes and behavior i.e. methods. An object is an instance of class. For example: Person, Student, Doctor, Engineer in above figure.

In this example, students refers to class and S1, S2 are the objects of class which can be created in main function.

```
class student
{
    char Name[20];
    int roll_no;
    --
    --
    public:
    void search();
    void update();
}
```

Encapsulation

- It refers to the building of data, along with the methods that operate on that data, into a single unit.
- There are three basic techniques to encapsulate data in object programming as data members, methods and classes which can be encapsulated.

Example: The bag contains different stuffs like pen, pencil, notebook etc within it, in order to get any stuff we need to open that bag.

Encapsulation and abstraction

- Abstraction is the method of hiding th unwanted information whereas encapsulationis a method to hide the data in a single entity or unit along with a method to protect information from outside.

Inheritance –

By using inheritance, new class can inherit the attributes and methods of the old class i.e. base class. For example: as classes Student, Doctor and Engineer are inherited from the base class Person.

Advantages:

- Codes can be reused due to inheritance.
- Easily understandable.
- Cost of maintenance can be reduced due to reusability of attributes and functions because of inheritance.

Disadvantages:

- It is not properly developed so not accepted by users easily.

**DBMS
RDBMS**

Vs

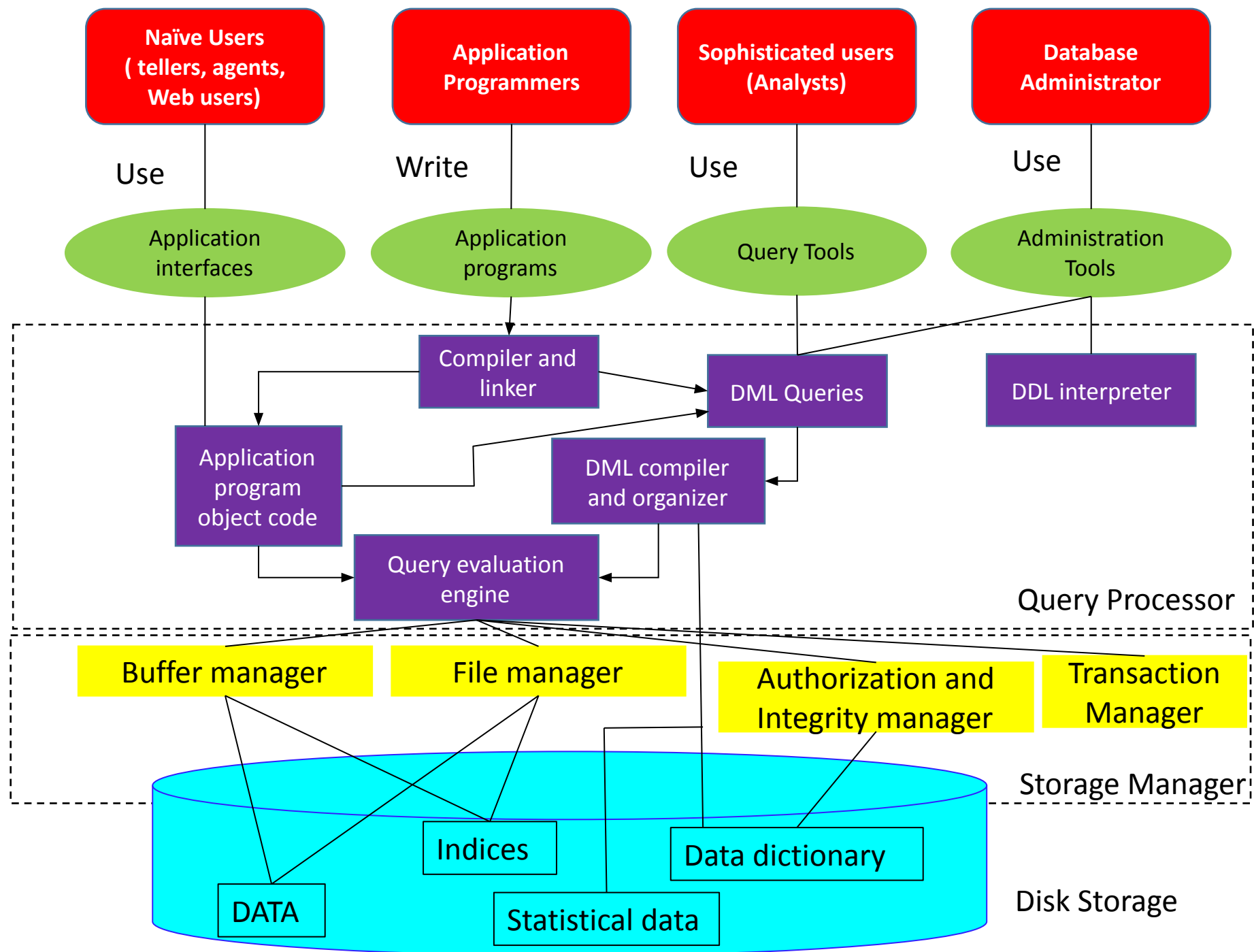
DBMS Vs RDBMS

The below table demonstrates the main difference between RDBMS and DBMS:

DBMS	RDBMS
DBMS stores data as a file.	Data is stored in the form of tables.
Data elements need to access individually.	Multiple data elements can be accessed at the same time.
No relationship between data.	Data is stored in the form of tables, which are related to each other.
It deals with small quantity of data.	It deals with large amount of data.
Data redundancy is common in this model.	Keys and indexes do not allow Data redundancy.
It supports single user.	It supports multiple users.
The data in a DBMS is subject to low security levels with regards to data manipulation.	There exists multiple levels of data security in a RDBMS.
Examples: XML, Window Registry, etc.	Examples: MySQL, PostgreSQL, SQL Server, Oracle, Microsoft Access etc.

Overall Database Structure

Overall database management structure



Database Users

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

- Application programmers – interact with system through DML calls, writes application programs
- Sophisticated users – form requests in a database query language eg analysts
- 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

Database Administrator (DBA)

- Coordinates all the activities of the database system; the database administrator has a good understanding of the enterprise's information resources and needs.
- Database administrator's duties include:
 - Schema definition- by executing statements in DDL
 - Storage structure and access method definition
 - Schema and physical organization modification
 - Granting user authority to access the database
 - Specifying integrity constraints eg. Primary key
 - Back up data
 - Ensure enough disk space for smooth performing of DB
 - Monitors jobs running on DB

Storage Manager

- Provide interface between low level data in database and application program.
- Components:
 - Authorization & integrity Manager: define role and responsibility for users, and provide Integrity checks.
 - Transaction manager: ensure DB consistency in concurrent access
 - File Manager: manage allocation of disk space and Data structure used for storing data
 - Buffer Manager: decide what data to *cache*

Disk Storage

- Data Files: which stores database itself
- Data Dictionary: stores metadata about the structure of the database
- Indices: provide fast access to data items

Query Processor

- DDL Interpreter: which interprets DDL statements and records the definitions in the data dictionary
- DML Compiler: translate DML statements to evaluation plan. Also perform query optimization. (*pick the lowest cost evaluation plan from among the alternatives*)
- Query Evaluation Engine: executes low level instructions generated by DML compiler.

References



- Korth, Silbertz and Sudarshan (1998), “Database Concepts”, 5th Edition, TMH.
- Elmasri and Navathe (2010), “Fundamentals of Database Systems”, 6th Edition, Addison Wesley.
- Date C J,” An Introduction to Database Systems”, 8th Edition, Addison Wesley.
- M. Tamer Oezsu, Patrick Valduriez (2011). “Principles of Distributed Database Systems”, 2nd Edition, Prentice Hall.
- <https://www.geeksforgeeks.org/difference-between-schema-and-instance-in-dbms/> last accessed on 10 December 2021.
- <https://www.javatpoint.com/dbms-data-independence/> last accessed on 10 December 2021.

*Thank
you*

