Data Models

- A Data Model in DBMS is the concept of tools that are developed to summarize the description of the database.
- It defines how the logical structure of a database is modeled.
- Data Models are fundamental entities to introduce abstraction in a DBMS.
- It defines how data is connected to each other and how they are processed and stored inside the system.
- A Data Model is collection of conceptual tools for describing:
  - Data
  - Data Relationships
  - Data Semantics
  - Consistency Constraints

Data Models

- Data models describes a set of concepts
  - Structure of a database
    - Elements and their data types
    - Records consisting of groups of elements
    - Relationships among the records
  - Operations for manipulating these structures
    - Retrievals and updates
  - Certain constraints that the database should obey

Categories of Data Models

- Conceptual (high-level) data models:
  - Provide concepts that are close to the way many users perceive data.
  - Entity Relationship (ER) Diagram comes under this model.
- Implementation (representational) data models:
  - Provide concepts that fall between the above two
  - It is used by many commercial DBMS implementations (e.g. relational data models)
- Physical (low-level, internal) data models:
  - Describe details of how data is stored in the computer.
  - These are usually specified in an ad-hoc manner through DBMS design

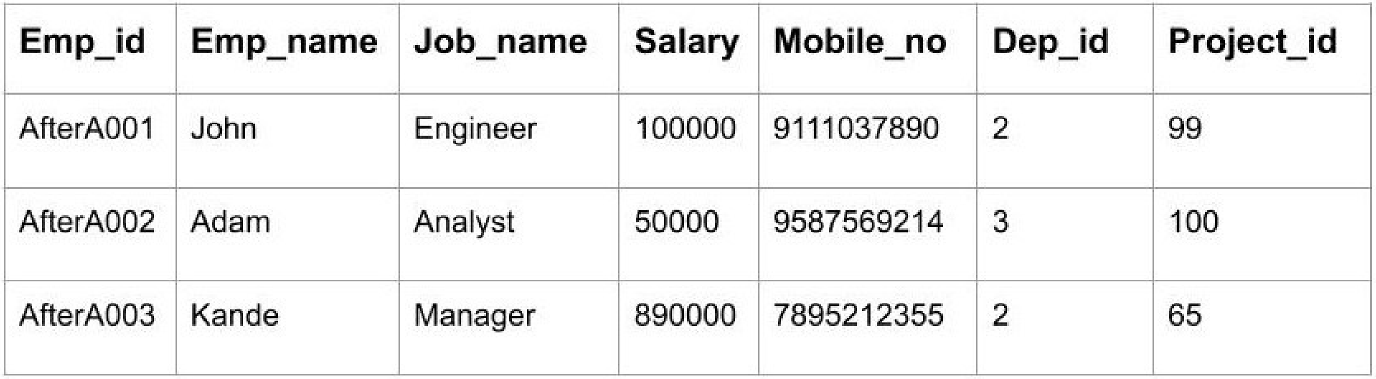
Types of Data Model

- Relational Model
- Entity-Relationship Model
- Object-based Data Model
- Hierarchical Model
- Network Model

Relational Model

- Relational Model is the most widely used model.
- In this model, the data is maintained in the form of a two-dimensional **tables** called ***Relation***.
- All the information is stored in the form of **rows and columns** where **columns** represents ***attributes*** and **row** represents ***records*** or ***tuples***.

- ***Features:***
  - ***Simple***
  - ***Scalable***
  - ***Structural Independence***

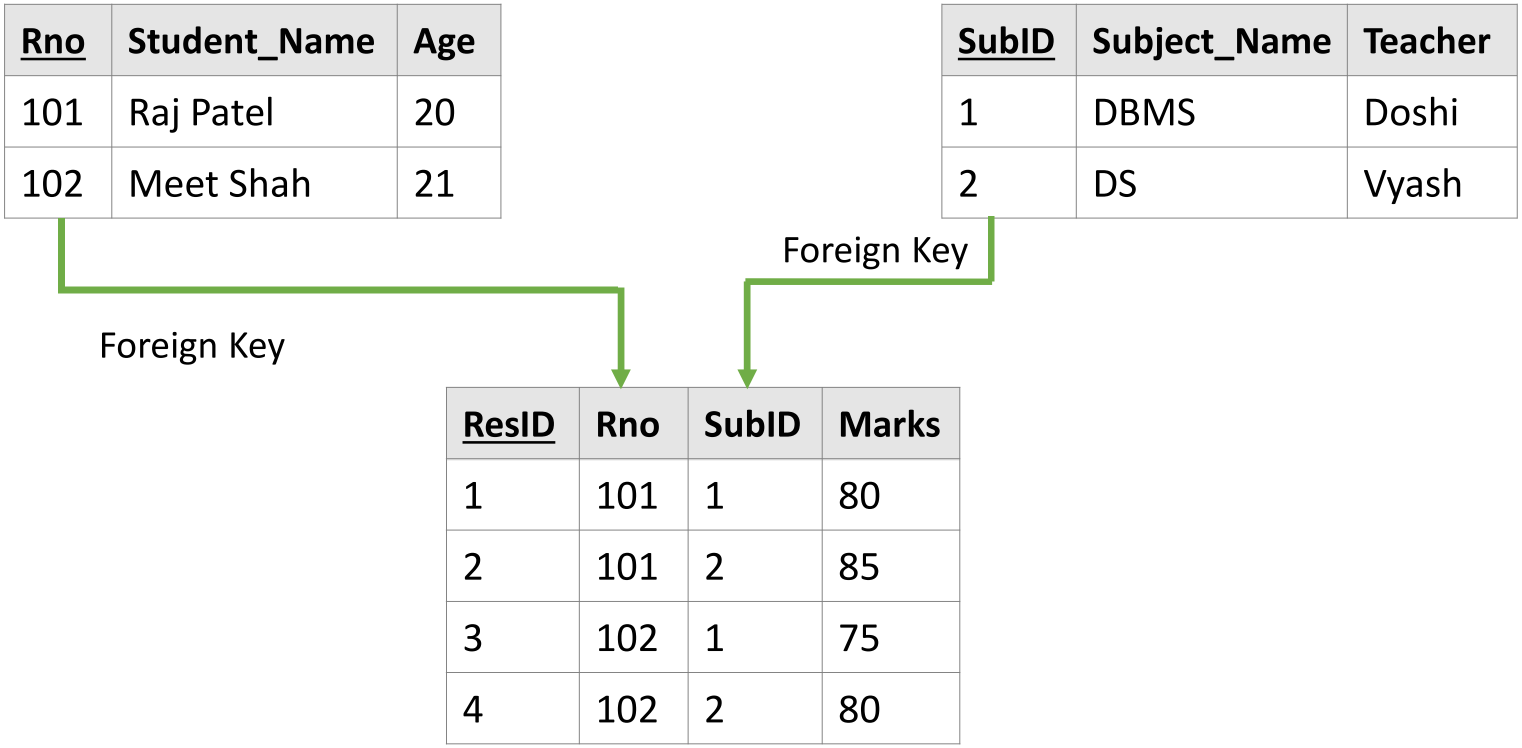


Emp_id	Emp_name	Job_name	Salary	Mobile_no	Dep_id	Project_id
AfterA001	John	Engineer	100000	9111037890	2	99
AfterA002	Adam	Analyst	50000	9587569214	3	100
AfterA003	Kande	Manager	890000	7895212355	2	65

EMPLOYEE TABLE

Relational Model

- In this model, **data is organized in two-dimensional tables** and the **relationship is maintained by storing a common attribute**.



Rno	Student_Name	Age
101	Raj Patel	20
102	Meet Shah	21

SubID	Subject_Name	Teacher
1	DBMS	Doshi
2	DS	Vyash

Foreign Key

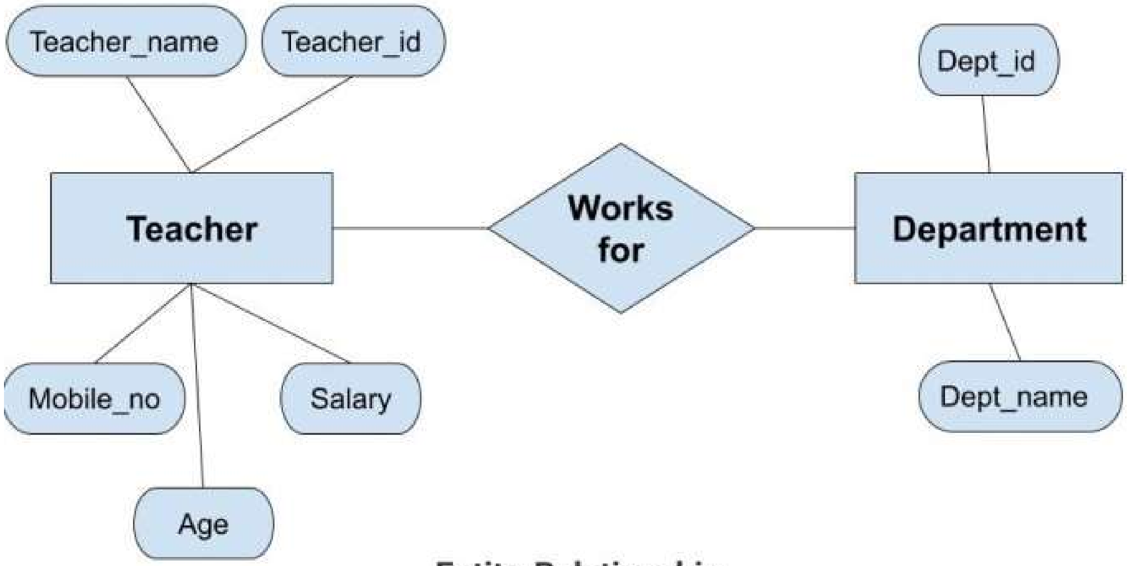
Foreign Key

ResID	Rno	SubID	Marks
1	101	1	80
2	101	2	85
3	102	1	75
4	102	2	80

E-R 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.
- 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.

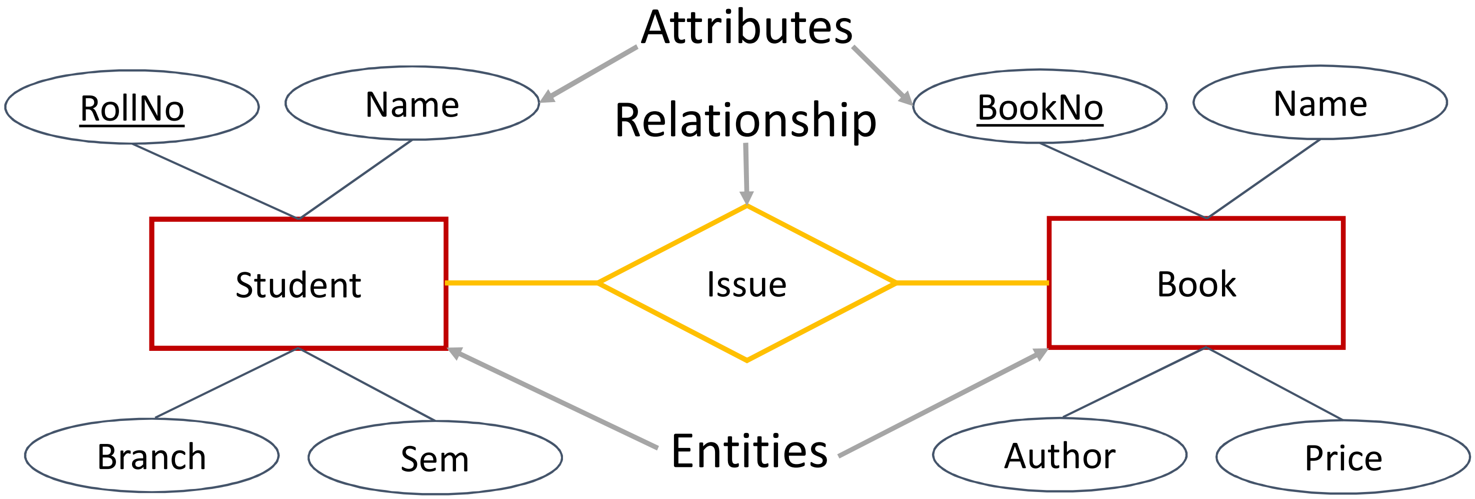
- ***Features:***
  - Graphical Representation for Better Understanding
  - Helps to Design Database



Entity-Relationship Model

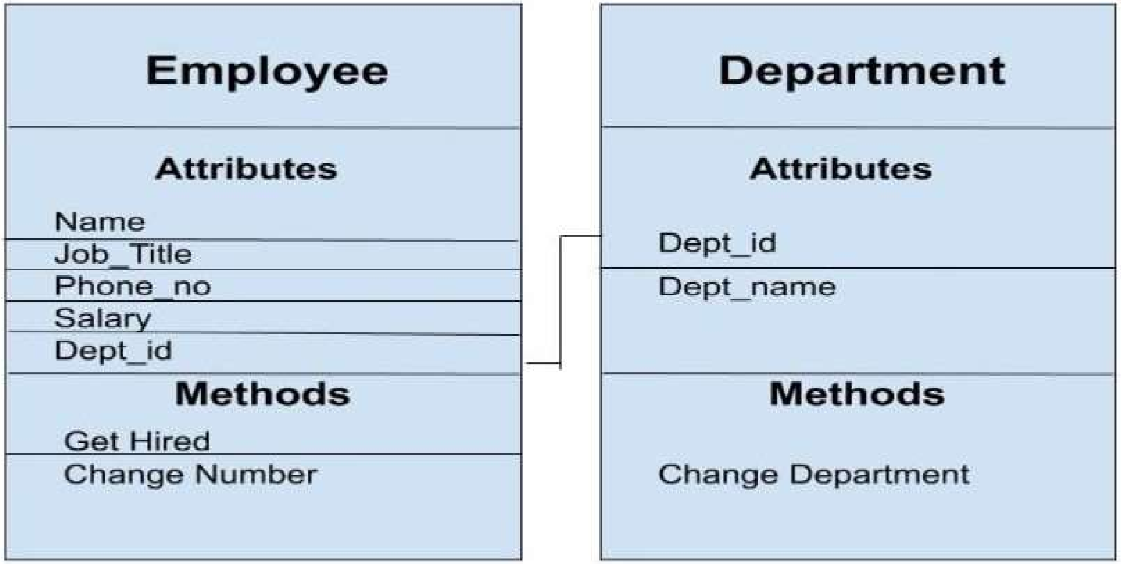
Entity-relationship Model

- In this database model, **relationships are created by dividing object of interest into entity** and **its characteristics into attributes**.



Object-Based Model

- Real-world problems are more closely represented through the object-based i.e. object-oriented data model.
- Both the data and relationship are present in a single structure known as an object.
- In this model, two are more objects are connected through links. We use this link to relate one object to other objects.



Employee

Attributes

Name
Job_Title
Phone_no
Salary
Dept_id

Methods

Get Hired
Change Number

Department

Attributes

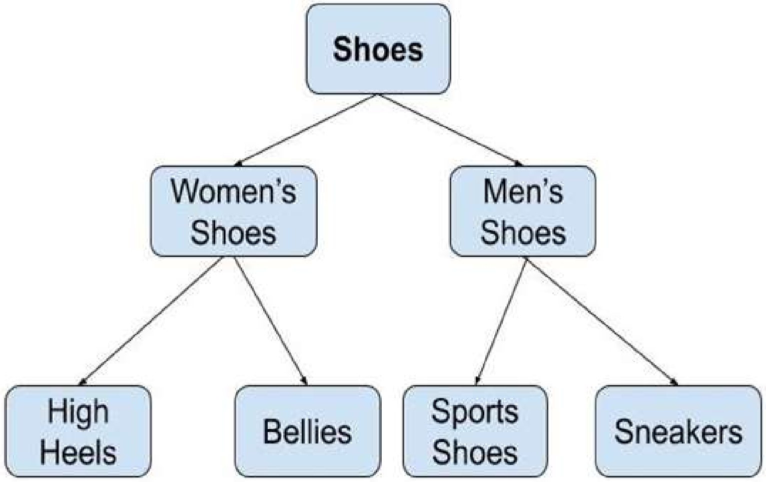
Dept_id
Dept_name

Methods

Change Department

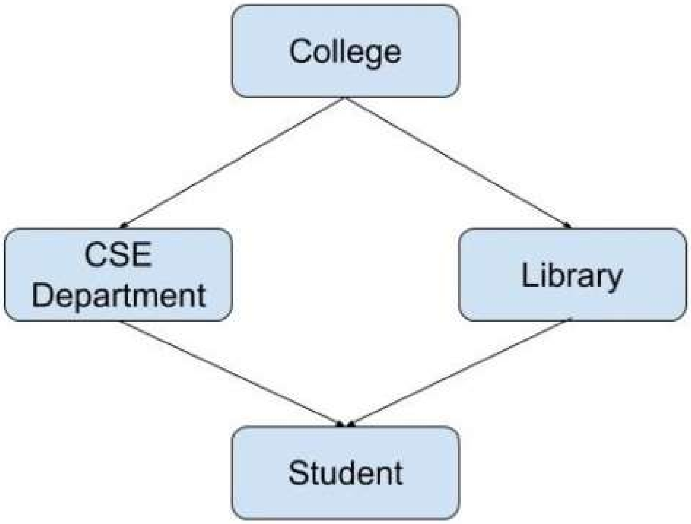
Object_Oriented_Model

Hierarchical and Network Model



Hierarchical Model

- Organizes the data in the hierarchical tree structure
- Hierarchy starts from the root and expands in the form of a tree
- ***Features:***
  - ***One-to-many relationship***
  - ***Parent-Child Relationship***
  - ***Deletion Problem***
  - ***Pointers***



Network Model

- Extension of the hierarchical model
- Record can have more than one parent.
- ***Features:***
  - Ability to Merge more Relationships
  - Many paths
  - Circular Linked List

Integrity Constraints

Integrity Constraints

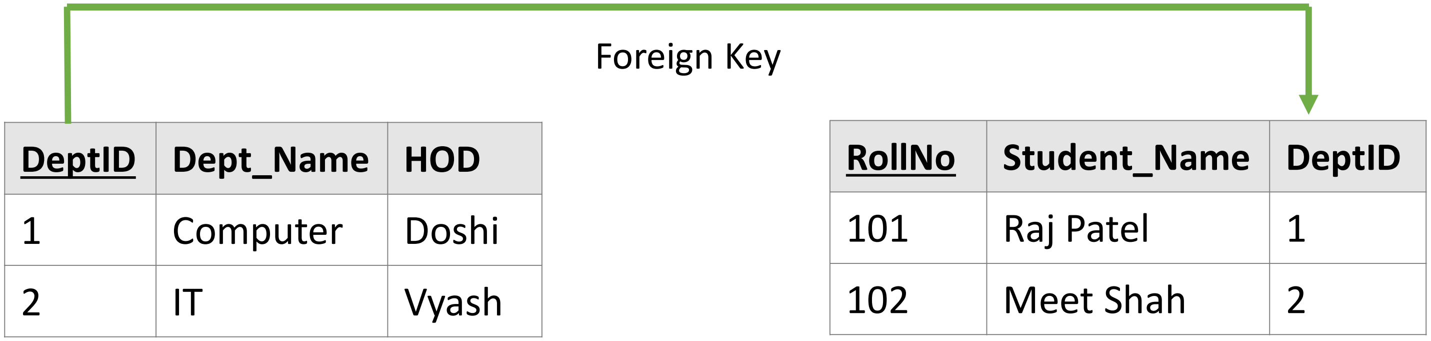
- Integrity constraints are a **set of rules**. It is used to **maintain the quality** of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to **guard against accidental damage** to the database.
- Various Integrity Constraints are:
  - Check
  - Not null
  - Unique
  - Primary key
  - Foreign key

Integrity Constraints

- Check
  - This constraint defines a business rule on a column. All the rows in that column must satisfy this rule.
  - Limits the data values of variables to a **specific set, range, or list of values**.
  - The constraint can be applied for a single column or a group of columns.
  - E.g. value of SPI should be between 0 to 10.
- Not null
  - This constraint ensures all rows in the table contain a definite value for the column which is specified as not null. Which means a **null value** is not allowed.
  - E.g. name column should have some value.
- Unique
  - This constraint ensures that a column or a group of columns in each row have a **distinct (unique)** value.
  - A column(s) can have a null value but the values cannot be duplicated.
  - E.g. “enrollmentno” column should have unique value.

Integrity Constraints

- Primary key
  - This constraint defines a column or combination of columns which uniquely identifies each row in the table.
  - Primary key = **Unique key + Not null**
  - E.g. enrollmentno column should have unique value as well as can't be null.
- Foreign key (referential integrity constraint)
  - A referential integrity constraint (foreign key) is specified between two tables.
  - In the referential integrity constraints, if a foreign key column in table 1 refers to the primary key column of table 2, then every value of the foreign key column in table 1 must be null or be available in primary key column of table 2.



Foreign Key

DeptID	**Dept_Name**	**HOD**
1	Computer	Doshi
2	IT	Vyash

RollNo	**Student_Name**	**DeptID**
101	Raj Patel	1
102	Meet Shah	2