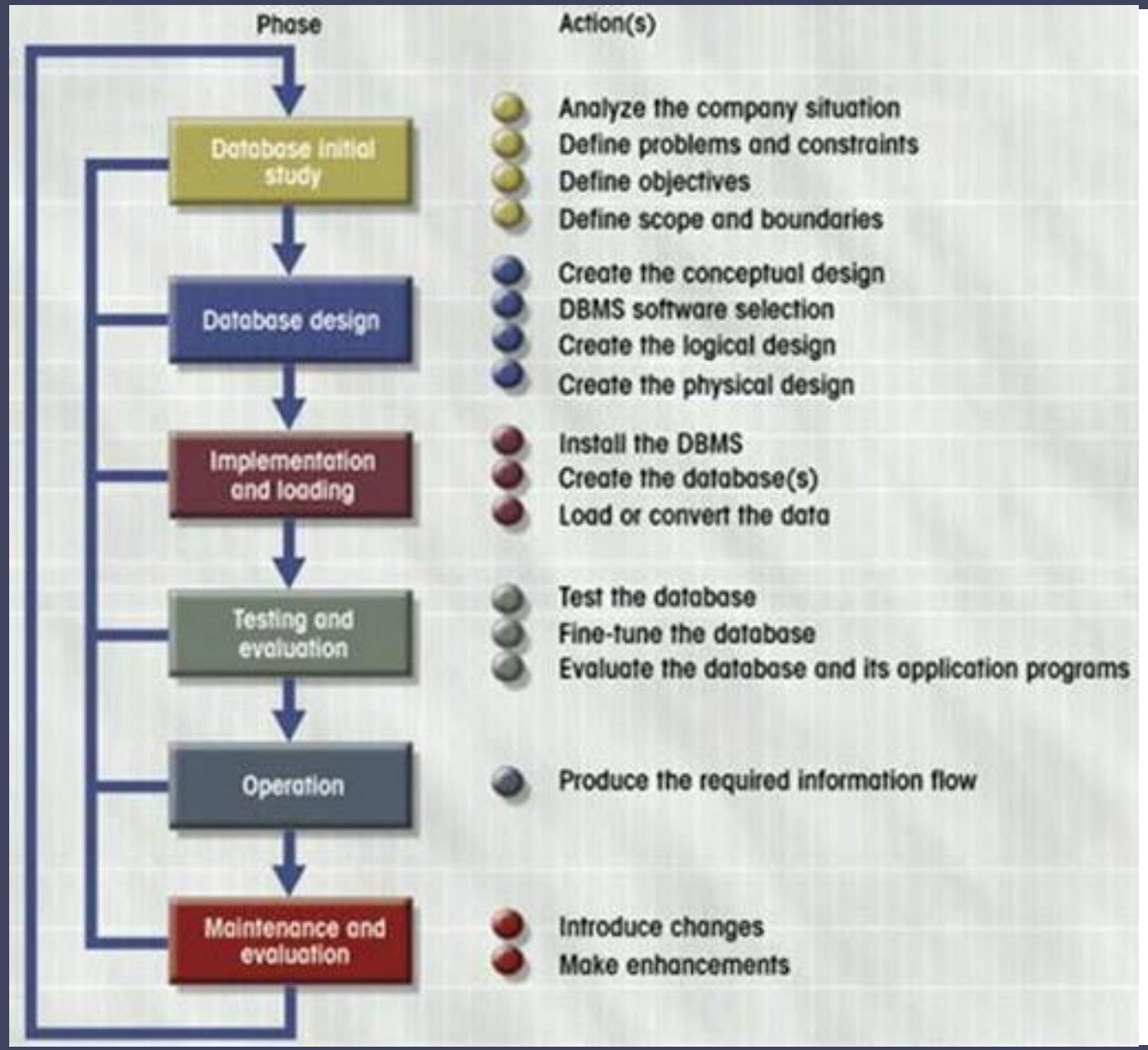
Database Systems

DR. MARIA SALEEMI

Database Development Cycle


Phase
Action(s)
Database initial study
Analyze the company situation
Define problems and constraints
Define objectives
Define scope and boundaries
Database design
Create the conceptual design
DBMS software selection
Create the logical design
Create the physical design
Implementation and loading
Install the DBMS
Create the database(s)
Load or convert the data
Testing and evaluation
Test the database
Fine-tune the database
Evaluate the database and its application programs
Operation
Produce the required information flow
Maintenance and evaluation
Introduce changes
Make enhancements

Phase I

Analysis

The analysis phase is where the stakeholders are interviewed, and any existing system is examined to identify problems, possibilities and constraints. The objectives and scope of the new system are determined.

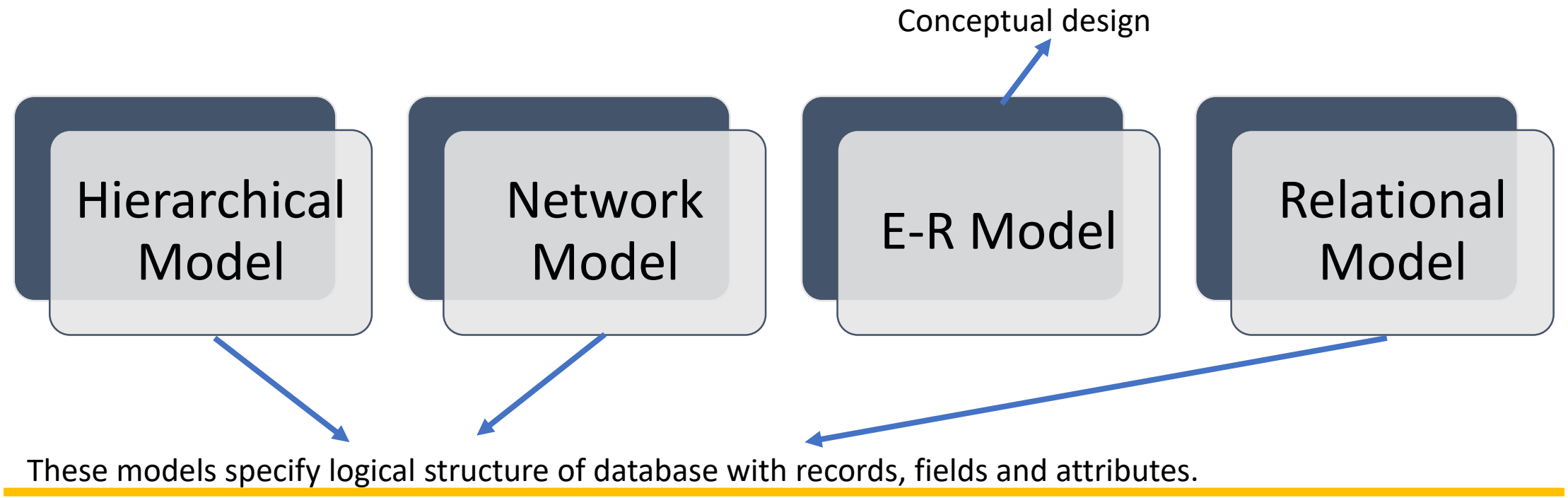
Phase II

Design

The design phase is where a conceptual design is created from the previously determined requirements, and a logical & physical design are created that will ready the database for implementation.

Database Design Models

Phase II: DESIGN
Database Design Models

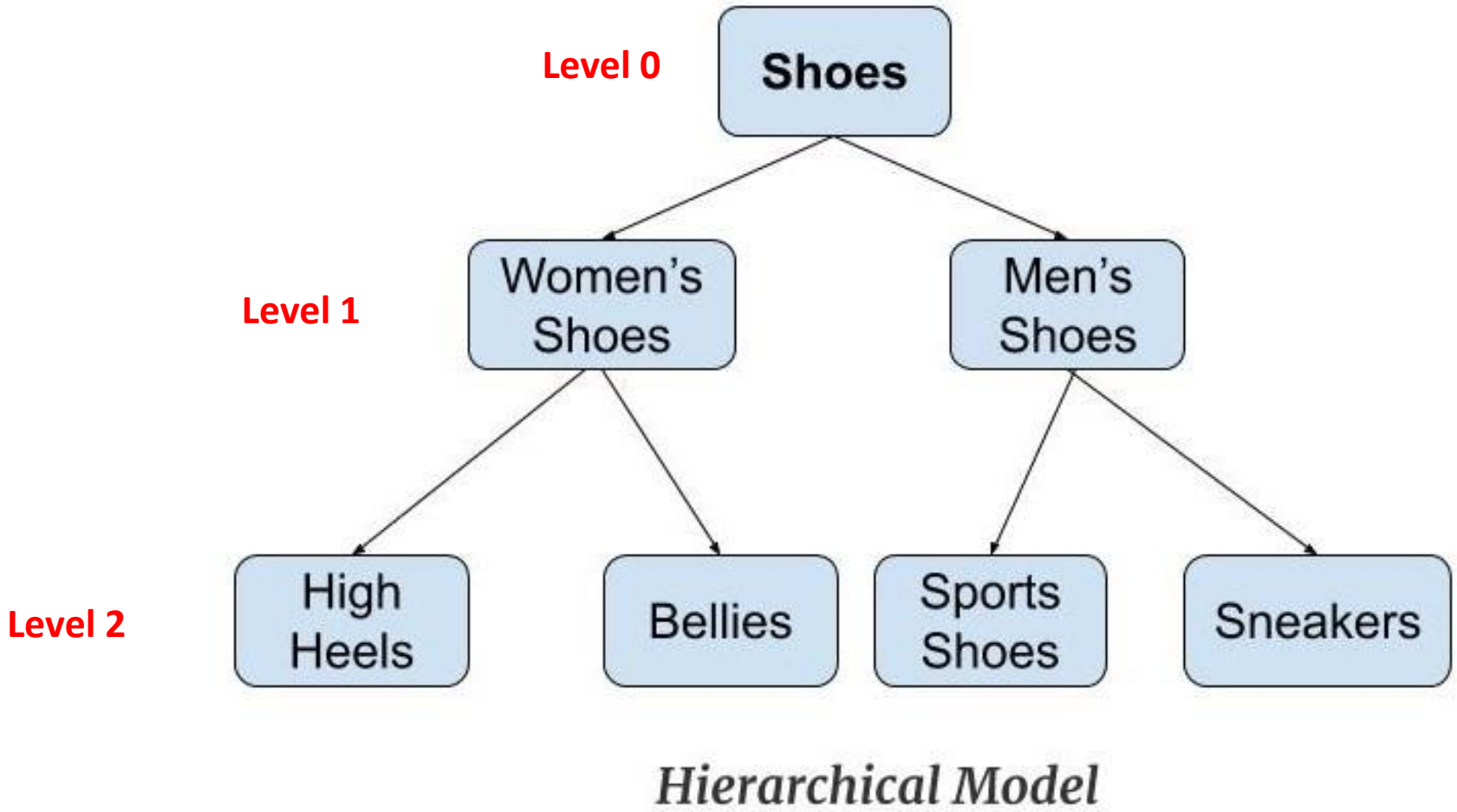


1. Hierarchical Model

In a Hierarchical database, model data is organized in a tree-like structure.

Data is Stored Hierarchically (top down or bottom up) format.

Data is represented using a parent-child relationship. In Hierarchical DBMS parent may have many children, but children have only one parent.



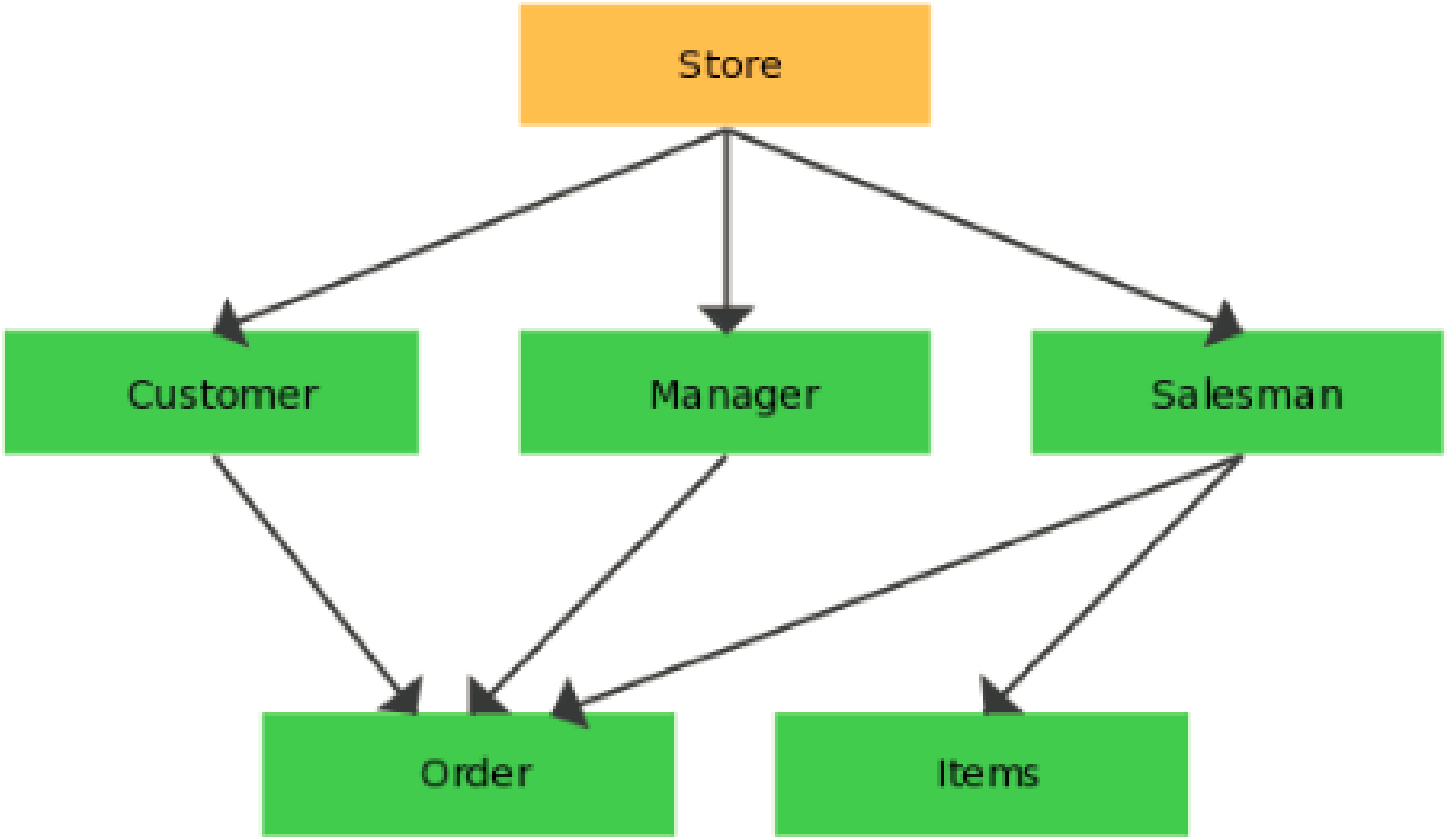
Hierarchical Model

2. Network Model

The network database model allows each child to have multiple parents.

It helps you to address the need to model more complex relationships like as the orders/parts many-to-many relationship.

In this model, entities are organized in a graph which can be accessed through several paths.


Store
Customer
Manager
Salesman
Order
Items

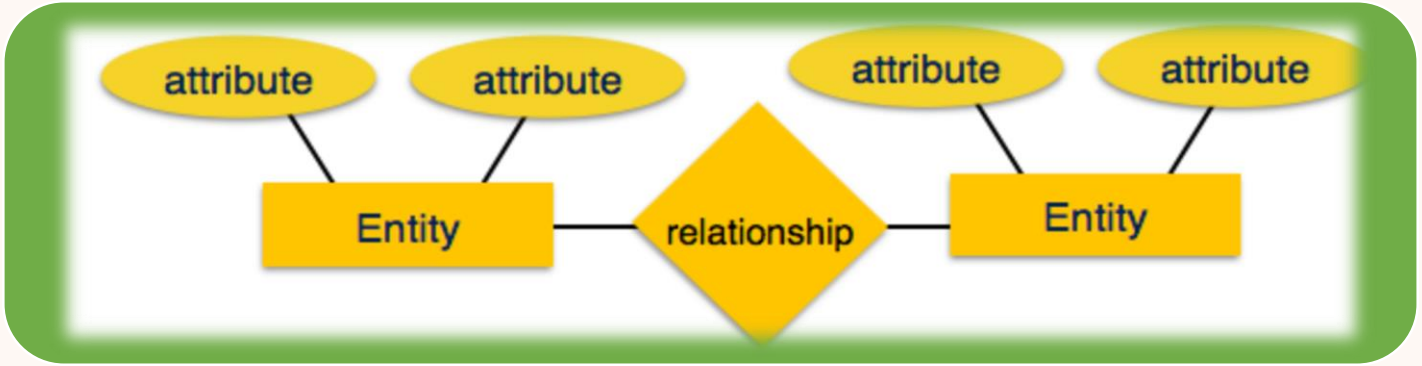
3. ER Model

Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints.

ER Model is best used for the conceptual design of a database. ER Model is based on:

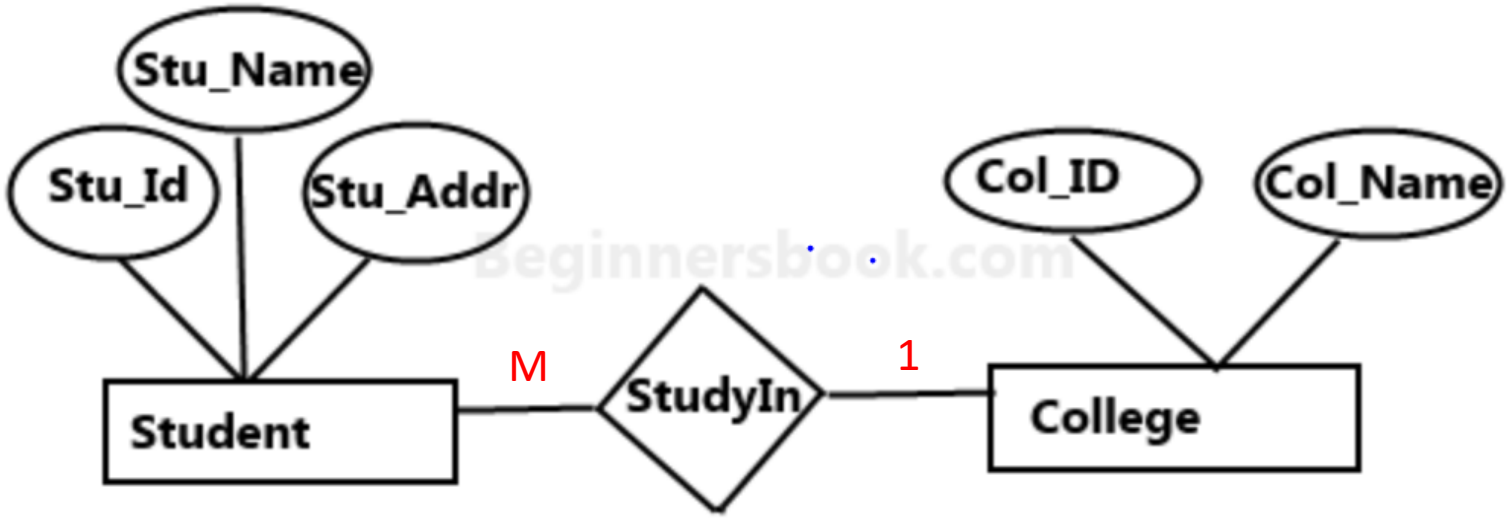
Entities and their *attributes.*

Relationships among entities.

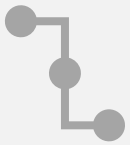


A simple ER Diagram:

In the following diagram we have two entities Student and College and their relationship. The relationship between Student and College is many to one as a college can have many students however a student cannot study in multiple colleges at the same time. Student entity has attributes such as Stu_Id, Stu_Name & Stu_Addr and College entity has attributes such as Col_ID & Col_Name.



Entity – An entity in an ER Model is a real-world entity having properties called **attributes**. Every **attribute** is defined by its set of values called **domain**. For example, in a school database, a student is considered as an entity. Student has various attributes like name, age, class, etc.



Relationship – The logical association among entities is called ***relationship***. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of association between two entities.

Mapping cardinalities –

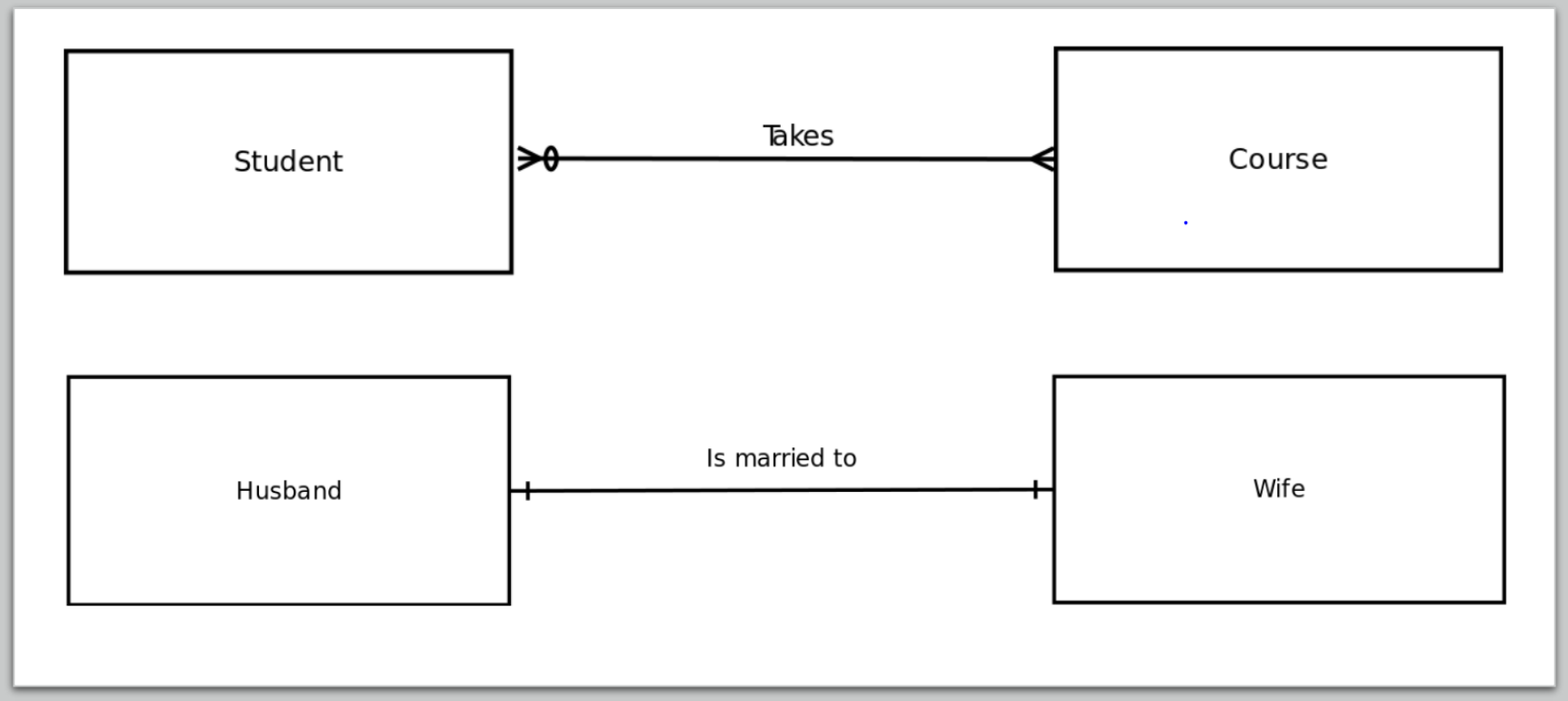
one to one

one to many

many to one

many to many

- Each student must have registered for at least one course, but a course does not necessarily have to have students registered. The student-to-course relationship is mandatory, and the course-to-student relationship is optional. (many to many)
- One husband *is married to* one wife and one wife to one husband (one to one)



4. Relational Model

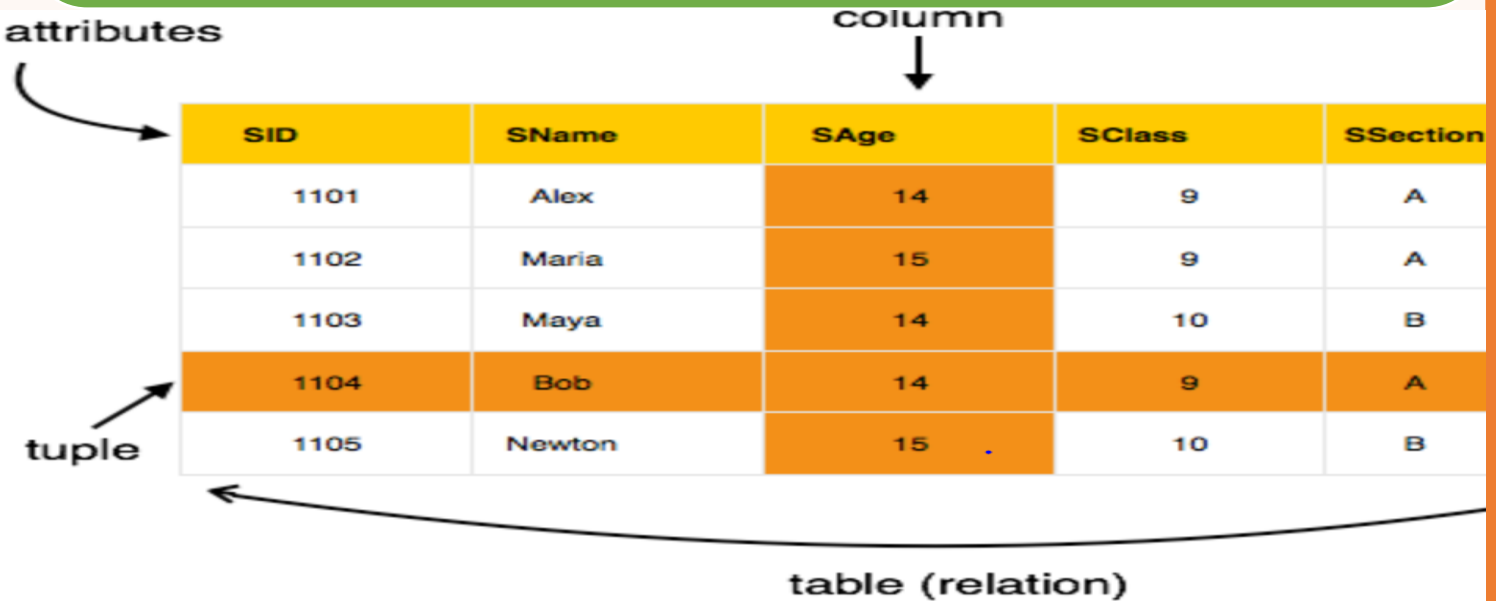
The most popular data model in DBMS is the Relational Model.

Data is stored in tables called **relations**.

Relations can be normalized.

Each row in a relation contains a unique value.

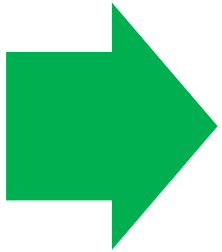
Each column in a relation contains values from a same domain



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

Phase III

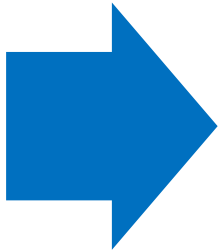
Implementation



The implementation phase is where the database management system (DBMS) is installed, the databases are created, and the data are loaded or imported.

Phase IV

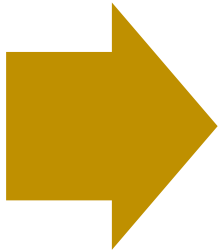
Testing



The testing phase is where the database is tested and fine-tuned, usually in conjunction with the associated applications.

Phase V

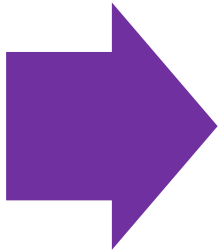
Operation



The operation phase is where the database is working normally, producing information for its users.

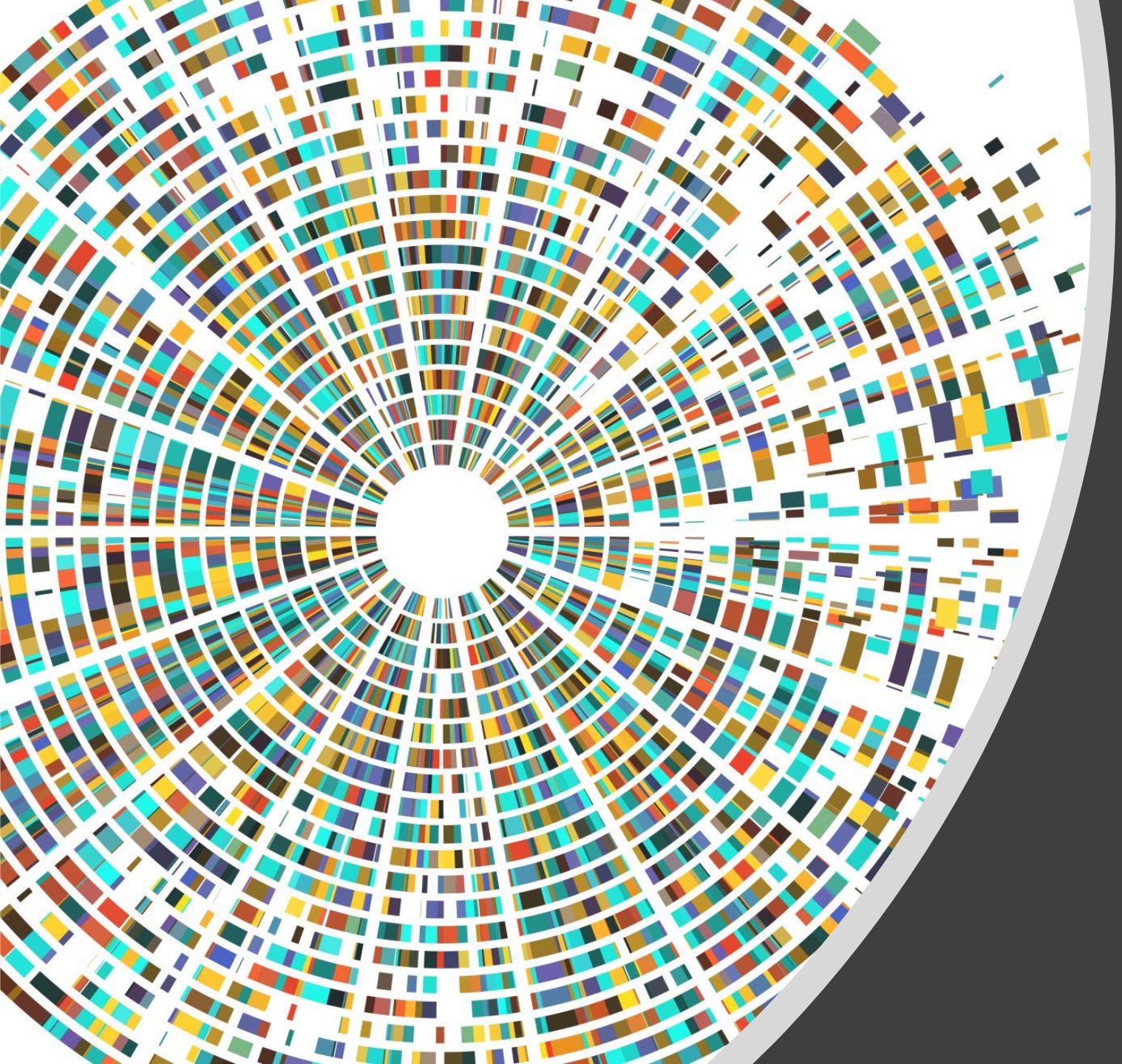
Phase VI

Maintenance



The maintenance phase is where changes are made to the database in response to new requirements or changed operating conditions.

Database Architectures



1 Tier Architecture
2 Tier Architecture
3 Tier Architecture

1 Tier Architecture

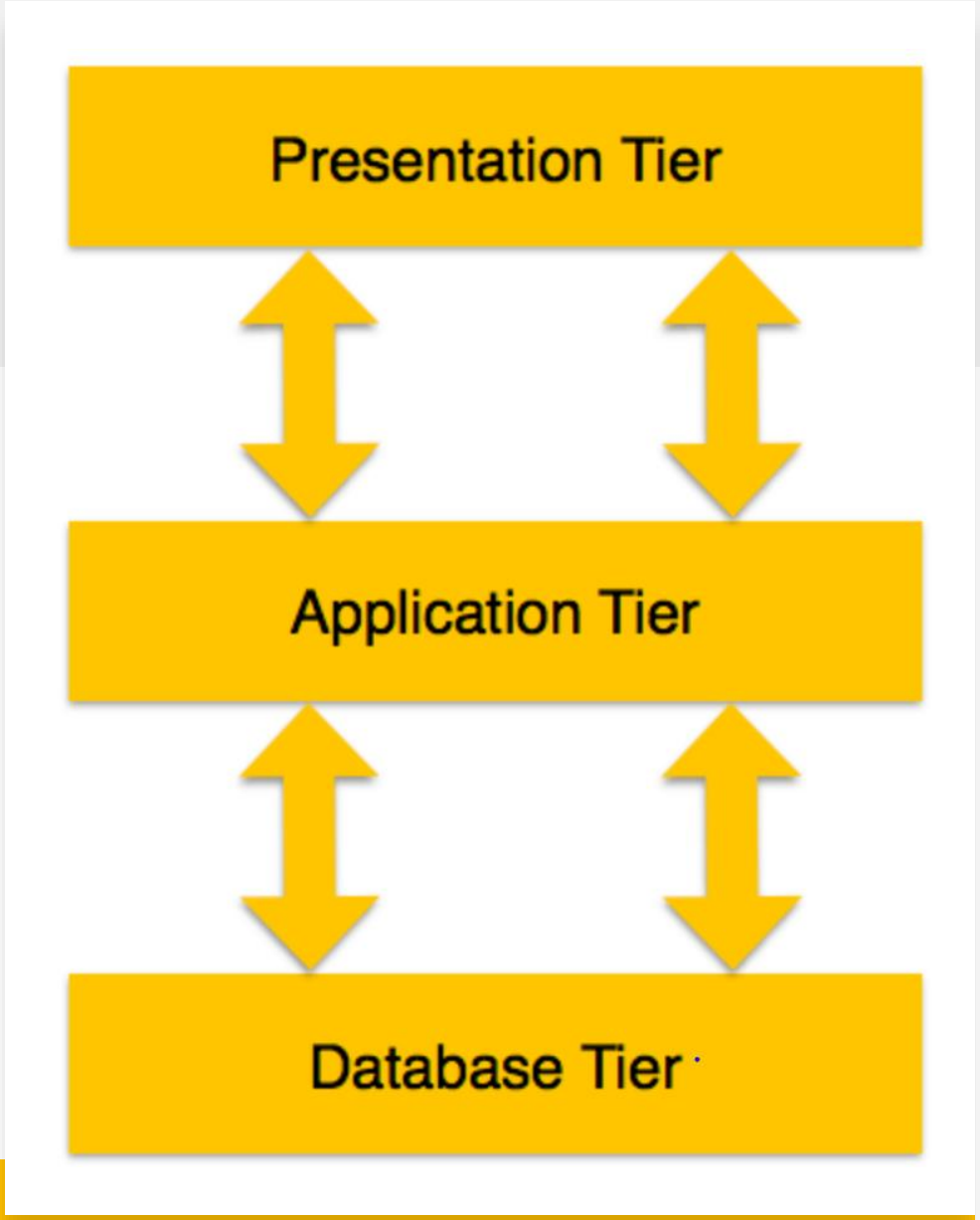
- the DBMS is the only entity where the user directly sits on the DBMS and uses it.
- Any changes done here will directly be done on the DBMS itself.
- It does not provide handy tools for end-users.
- Database designers and programmers normally prefer to use single-tier architecture.

2 Tier Architecture

- Programmers use 2-tier architecture where they access the DBMS by means of an application.
- Here the application tier is entirely independent of the database in terms of operation, design, and programming.

3 Tier Architecture

- A 3-tier architecture separates its tiers from each other based on the complexity of the users and how they use the data present in the database.
- It is the most widely used architecture to design a DBMS.



3- Tier Architecture

- **Database (Data) Tier** – At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.
- **Application (Middle) Tier** – At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
- **User (Presentation) Tier** – End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.