# GAZİ UNIVERSITY

# FACULTY OF ENGINEERING
# DEPARTMENT OF INDUSTRIAL ENGINEERING

Lecturer : Dr Ercan Ezin

IE326-DATABASE MANAGEMENT SYSTEMS

WEEK 2: DATA AND DATABASE DESIGN

# AGENDA

- DATA AND AINFORMATION
- DATA MODELS
- DATABASE DESIGN

# DATA & INFORMATION

# Data and Information


The term **DATA** consists of unprocessed facts that are used in the production of information and are suitable for making sense of. These facts can be numeric, alphanumeric or symbols, or they can be graphical.

**INFORMATION** is the result obtained by analyzing and processing the data in order to make it meaningful to support the decision-making process.

**Data is the source of information.**

# Information & Information Systems

Informatics refers to the processes that includes all the applied operations in order to obtain information. Some of these processes are;

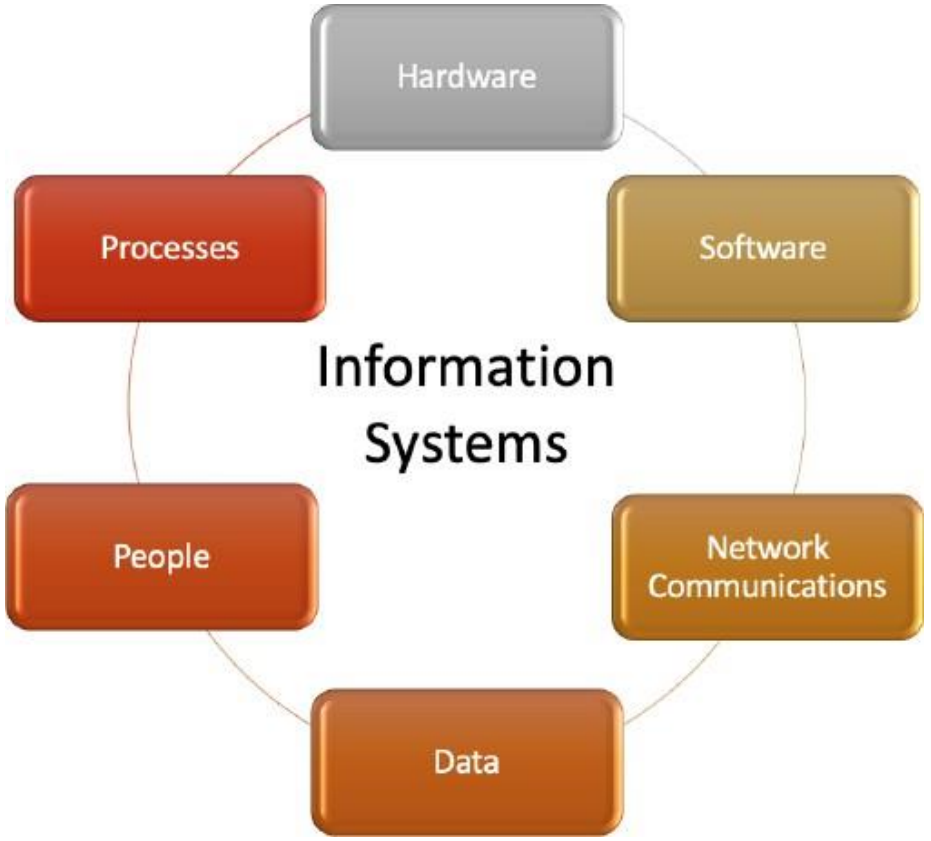
- association
- classification
- calculation
- correction
- summarization
- arrangement

- In a rough definition, a structure which performs the information process called an information system .
- It is an "*information production system*" that collects, processes, stores and reports the information necessary for its users to make a decision from different sources.

# Information & Information Systems

The information system can be:

- Manual
- Computer based.



# Information & Information Systems

- **Data** is the *most important atomic component* of the information system.
- The process of storing data in information systems is done using databases.
- In order to use a database management system with *appropriate features*, it is of "great importance" to design an effective database that will support the characteristics of the information in terms of form, time and content.


# Database Management Systems

- **Database management systems (DBMS)** are computer applications that store databases and ensure that the stored databases are managed and operated within certain order and restrictions.

# Database Management Systems

## Some of functions of DBMS are:

- Data sharing,
- Conceptual data organization,
- User and rights management,
- Backup and recover,
- Many components for information production,
- Data distribution,
- Restriction definition and application.

# Database Management Systems

Some of *the most important DBMS* currently used are listed below:

- Microsoft SQL Server
- Oracle IBM DB2
- MySQL (Open source, some paid)
- Sybase Firebird (Open source)
- PostgreSQL (Open source)

One of the biggest advantages of DBMSs is that they allow users and application developers to deal with the conceptual representation of the data without needing to deal directly with the data itself. This structure also brings many advantages in terms of data security.

# DATA MODELS

# Data Models

- Data models define the data and database structure based on entities,
  attributes of entities and relationships between entities.
- The used data model will answer the following questions regarding the data to be kept in the system:
  - How the data will be grouped and kept,
  - Which groups will be related to each other,
  - How to connect one group to another ,
  - How will be defined the characteristics of the relations between the groups (i.e. the student's grade in the course),
  - What will be the restrictions on data processing?

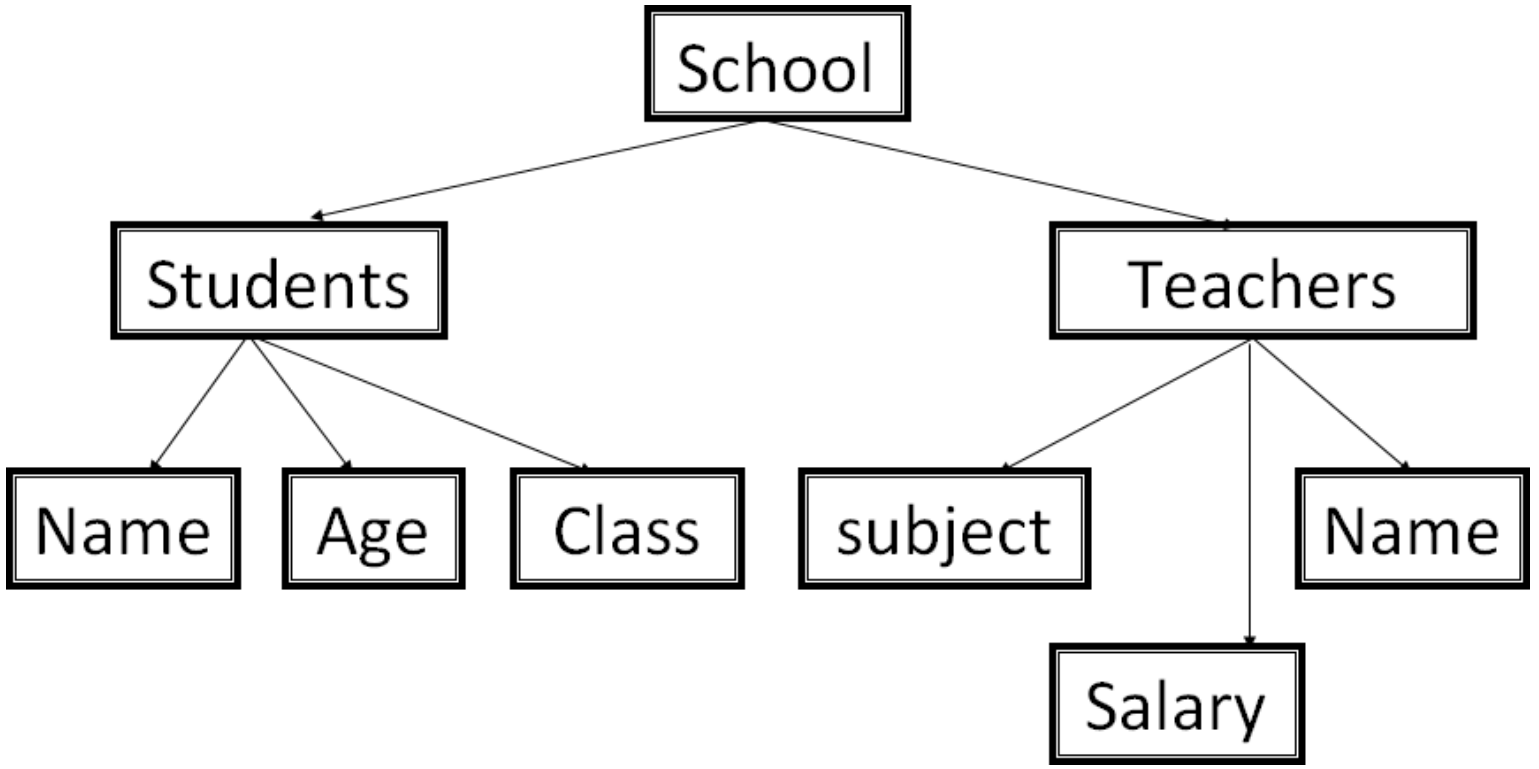
# Data Models – Flat Model

- The ***Flat Model*** was the first method used to store data in a computer.
- Each data item is stored on a disc sequentially in one large file.
- If you want to ***locate a particular item***, you must start searching process from the ***beginning*** and each item is checked one by one sequentially until the searched value is ***matched***.
- Drawbacks:
  - ❖ Data duplication, very poor security, slow operation when retrieving data, non-support for different data types, memory allocation problem, concurrent data sharing is not allowed, insecure mechanism, etc.

# Data Models – Hierarchical Model

- Hierarchical Model was the first DBMS model.
- This model organizes the data in a hierarchical tree structure. The hierarchy starts from the *root* which has stem data and then it becomes more extensive in a form of tree adding *child node* to the parent node.
- This model easily represents some of the real-world relationships like food recipes, sitemap of a website etc.

# Data Models – Hierarchical Model



# Features of a Hierarchical Model

- **One-to-many relationship:** The example in previous slide is organized in a tree-like structure where the one-to-many relationship is between the datatypes. Also, there can be only one path from parent to any node. For example, if we want to go to the node **Salary** we only have one path to reach there i.e. through **Teachers** node.
- **Parent-Child Relationship:** Each child node has a parent node but a parent node can have more than one child node. Multiple parents are not allowed.
- **Deletion Problem:** If a parent node is deleted then the child node is automatically deleted.
- **Pointers:** Pointers are used to connect the parent node to child node and to navigate between the stored data. In the previous example the **School** node points to the two other nodes **Students** node and **Teachers** node.

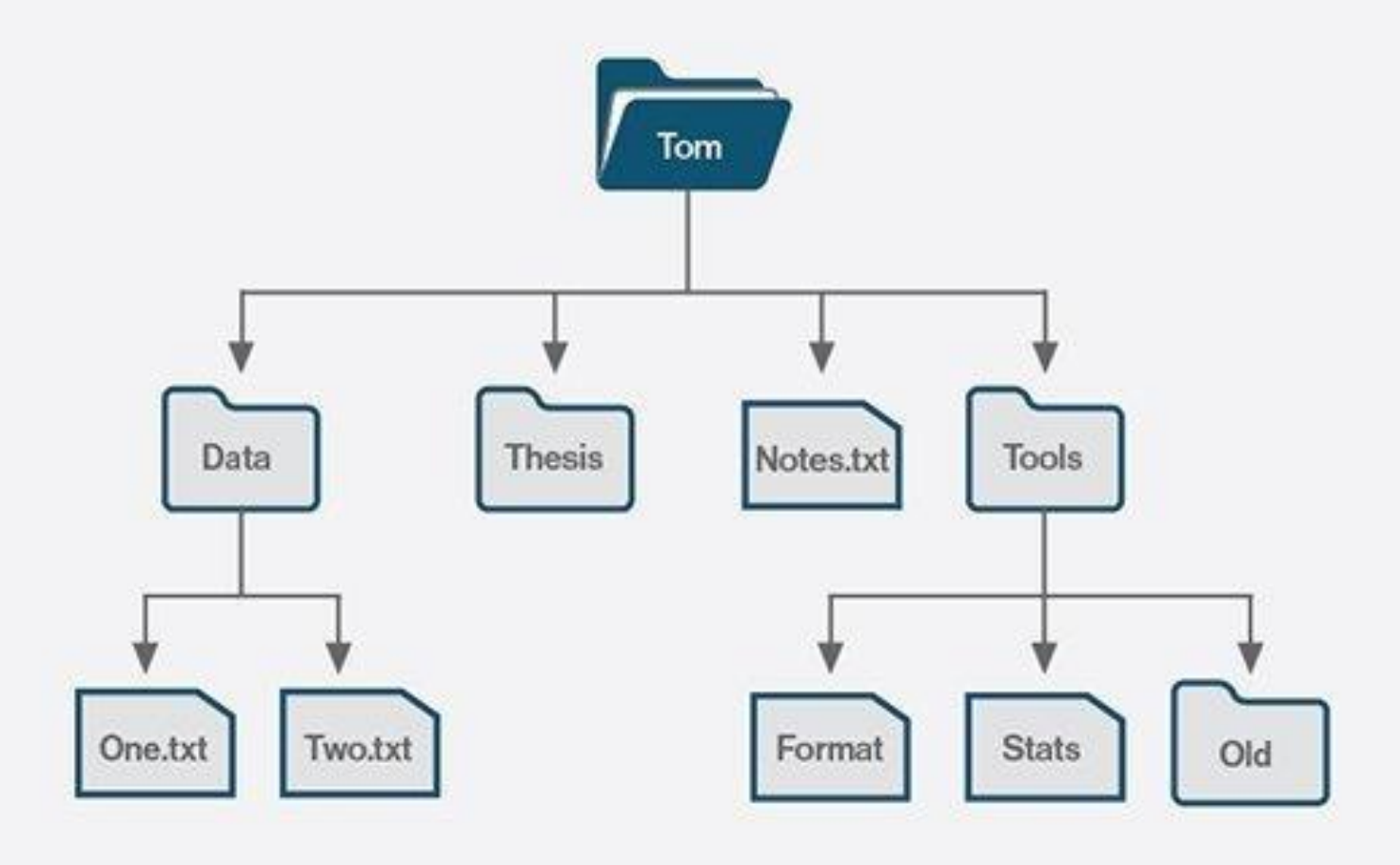
# Pros-Cons of Hierarchical Model

- Advantages of Hierarchical Model:
  - Very simple and fast to traverse through a tree-like structure.
  - Any change in the parent node is automatically reflected in the child node so, the integrity of data is maintained.

- Disadvantages of Hierarchical Model:
  - Complex relationships are not supported.
  - if we have some complex relationship where a child node needs to have two parent node, we would not able to use this model as the child node does not support multiple parents
  - If a parent node is deleted then the child node is automatically deleted.

# Example of Hierarchical Model

COMPUTER FILE SYSTEM IS HIERARCHICAL.

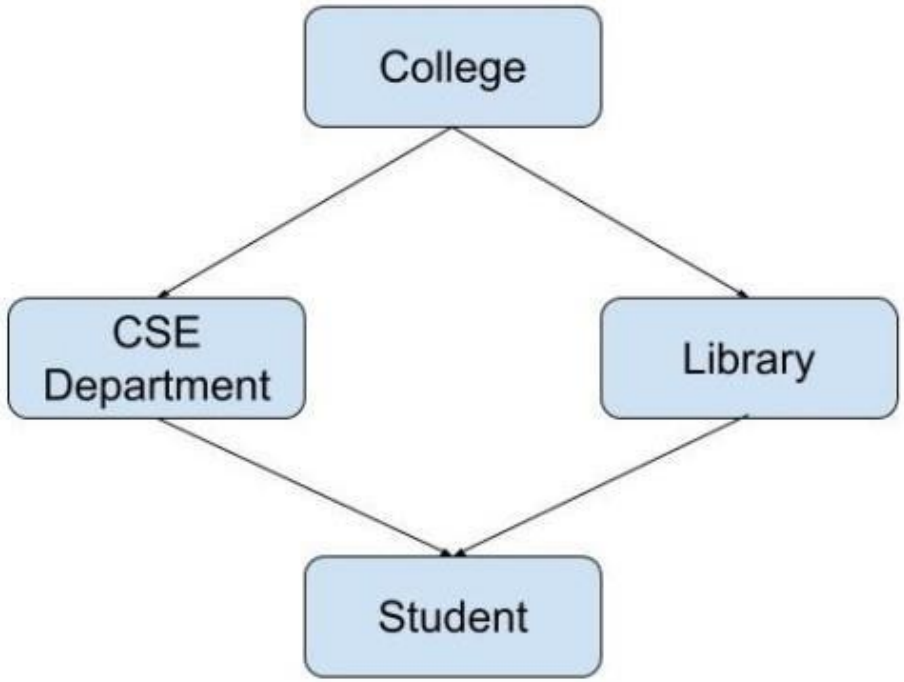


# Example of Hierarchical Model

```
- <customers>
  - <customer>
      <name>Griffin, Stewie</name>
      <address>31 Spooner Street, Quohog</address>
    - <orders>
      - <order number="120791-A">
          <shipper>UPS</shipper>
          <shipdate>2006-08-23</shipdate>
        </order>
      - <order number="120791-B">
          <shipper>Fedex</shipper>
          <shipdate>2006-08-23</shipdate>
        </order>
      </orders>
  </customer>
  - <customer>
      <name>Taronga, Leela</name>
      <address>1 Planet Express Plaza, NNY</address>
    - <orders>
      - <order number="120788">
          <shipper>Fedex</shipper>
          <shipdate>2006-08-23</shipdate>
        </order>
      - <order number="120844">
          <shipper>UPS</shipper>
          <shipdate>2006-08-24</shipdate>
        </order>
      </orders>
  </customer>
  - <customer>
      <name>Rubble, Barney</name>
      <address>93 Pebble Lane, Bedrock</address>
    - <orders>
      - <order number="120459">
          <shipper>Fedex</shipper>
          <shipdate>2006-08-21</shipdate>
        </order>
      </orders>
  </customer>
</customers>
```

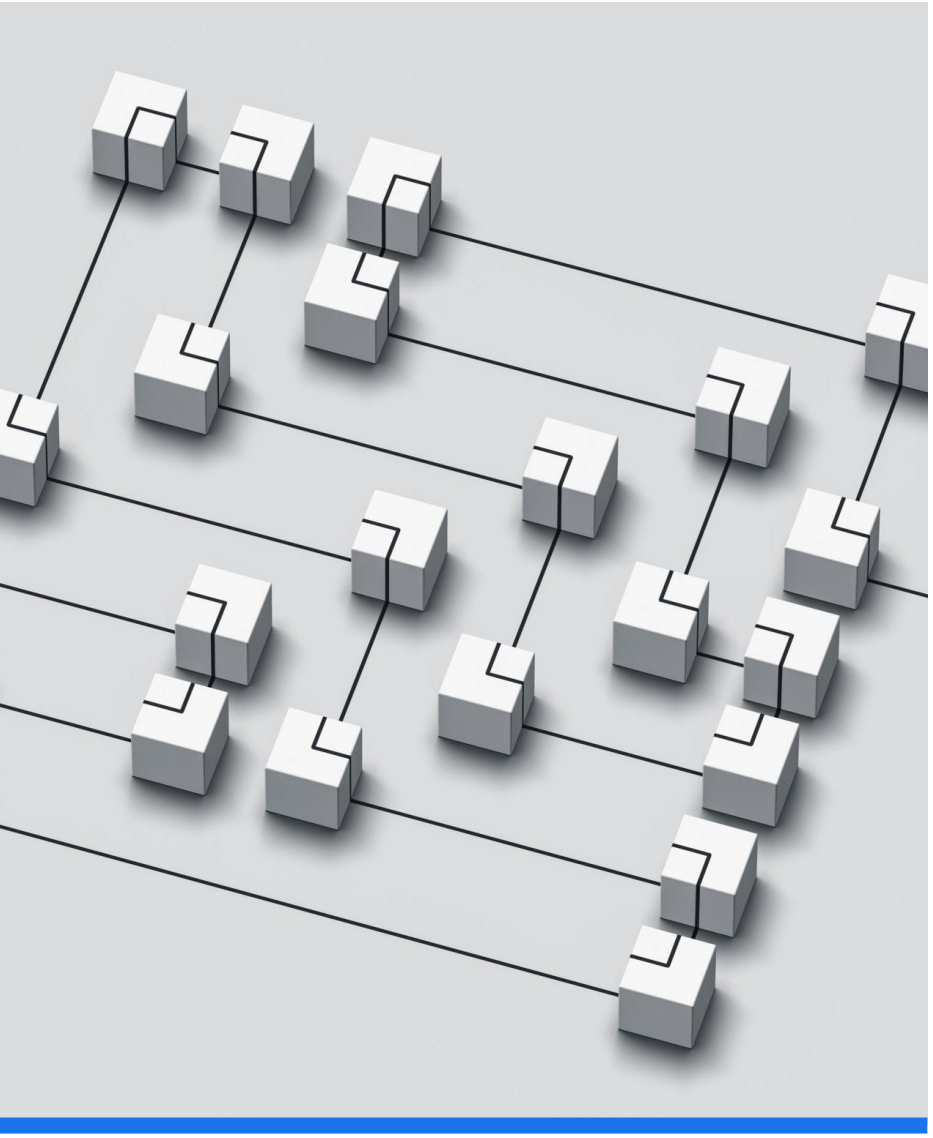
XML structures are hierarchical.

# Data Models – Network Model



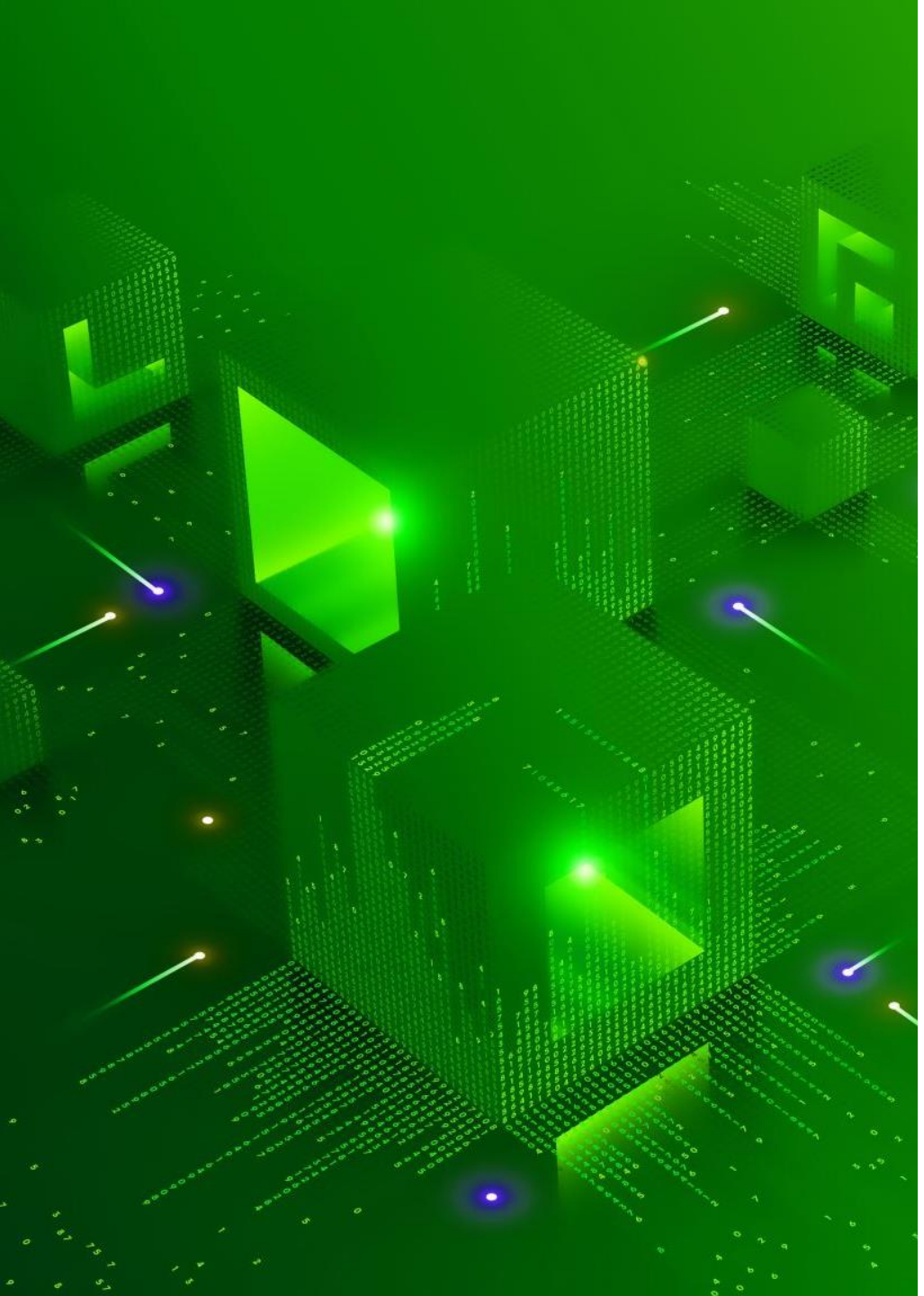

*Network Model*

- 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.



# Data Models – Object-Oriented Model

- A database with an object-oriented data model is called an object database. And the model used here is called Object Oriented Data Model (OODM)
- The main reason for the emergence of this model is the need to save and share complex objects (objects derived from class) when necessary.
- Like as in any object-oriented programming language, the classes are defined and the objects of this class are derived and stored in the database.



# Data Models – Object-Oriented Model

- OODM uses the same model as object-oriented programming
  •languages.
- It allows some basic features of object-oriented programming to identify and manipulate the data in database management systems.
- It makes the database objects look like a programming language
  •objects.

# Data Models – Object-Oriented Model

The real-world problems can be represented more closely using the object-oriented data model(OODM).

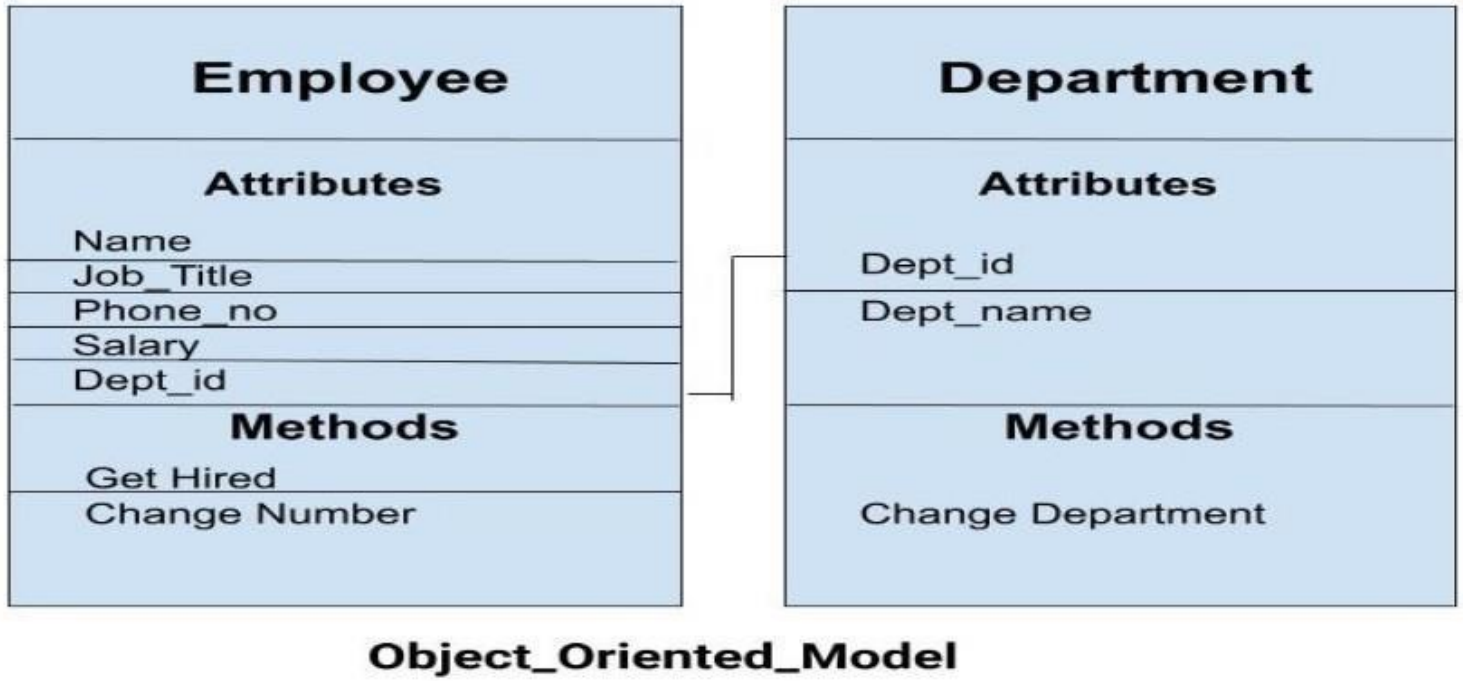
With this model, both the data and relationship are presented in a single structure known as an object.

Audio, video, images, etc. can be stored in a database which was *not possible* in the relational model

In an OODM model, two are more objects are connected through links. This link is used to relate one object to other objects.

# Data Models – Object-Oriented Model

AN EXAMPLE OF OODBMS.




Object_Oriented_Model

# Data Models - Relational Model

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 row and columns.

The basic building blocks of the structure of a relational model are tables.

So, in the relational model the tables are connected with relations.

# Data Models – Relational Model

- Example: Employee table is shown below.

| 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**

# Advantages of Relational Model

- Simple: This model is simpler compared to the network and hierarchical model.
- Scalable: This model can be easily scaled we can add as many rows and columns as we want.
- Structural Independence: We can make any changes to the database structure without changing the way data is accessed.
  - If any change is applied to the database structure without affecting the capability to DBMS and the data is still can be reached without an interruption.
  - So, it can be said that the structural independence has been achieved.

# Disadvantages of Relational Model

- Hardware Overheads: This model requires more powerful hardware computers and data storage devices to overcome complexities and making things easier for the user.
- Bad Design: The relational model is very easy to design and use, so the users don't need to know how the data is stored. However, this ease of design can cause the database to slow down later with database growth.
- But all these disadvantages are minor as compared to the advantages of the relational model. These problems can be avoided with the help of proper implementation and organization.

# Data Models – Entity-Relationship Model

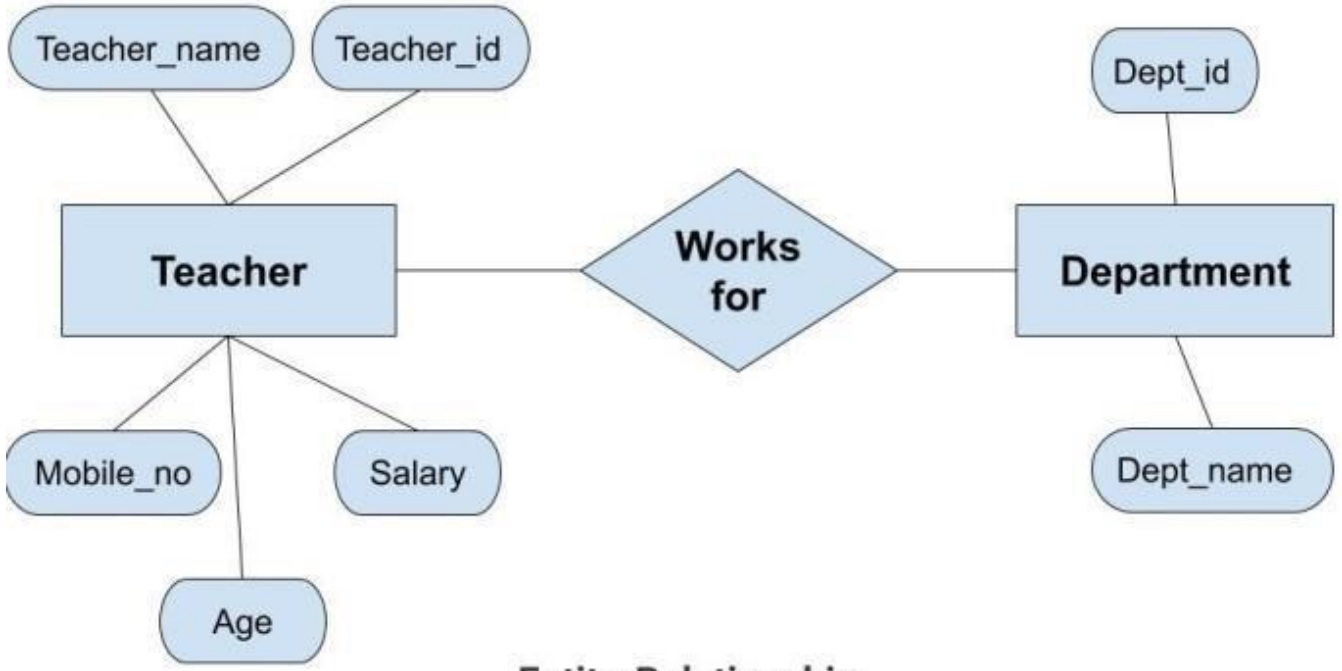
- Entity-Relationship Model or simply ER Model is a high-level data model diagram.
- In this model, the real-world problem is presented in the pictorial form to make it easy for the users to understand.
- It is also very easy for the developers to understand the system by just looking at the ER diagram.
- ER diagram is used as a visual tool to represent an ER Model.

# Data Models – Entity-Relationship Model

ER diagram has the following three components:

- Entities: Entity is a real-world thing. It can be a person, place, or even a concept. Example: Teachers, Students, Course, Building, Department, etc. are some of the entities of a School Management System.
- Attributes: An entity contains a real-world property called attribute. This is the characteristics of an attribute. Example: The entity teacher has an attribute like teacher id, salary, age, etc.
- Relationship: Relationship tells how two attributes are related. Example: Teacher works for a department.

# Data Models – Entity-Relationship Model




Entity-Relationship Model

- In the left side diagram, the entities are Teacher and Department.
- The attributes of Teacher entity are *Teacher_Name*, *Teacher_id*, *Age*, *Salary*, *Mobile_Number*.
- The attributes of entity Department entity are *Dept_id*, *Dept_name*.
- The two entities are connected using the relationship.
- Here, each teacher works for a department.

# Features of Entity-Relationship Model

- Graphical Representation: It is very easy and simple to understand so it can be used by the developers to communicate with the stakeholders.
- ER Diagram: ER diagram is used as a visual tool for representing the model.
- Database Design: This model helps the database designers to build the database.

# Advantages of Entity-Relationship Model

- ***Simple:*** Conceptually ER Model is very easy to build. If we know the relationship between the attributes and the entities, we can easily build the ER Diagram for the model.
- ***Effective Communication Tool***: This model is used widely by the database designers for communicating their ideas.
- ***Easy Conversion to any Model***: This model matches well with the relational model and can be easily converted to a relational model by converting the ER model to a table. It also can be converted to any other model such as network model, hierarchical model etc.

# Disadvantages of Entity-Relationship Model

- ***No industry standard for notation:*** There is no industry standard for developing an ER model. So, a developer may use notations that can not be understandable by the other developers.
- ***Hidden information:*** Some information might be lost or hidden in the ER model. Since it is a high-level view, there is a possibility that some information will be hidden.

# Relational DB Model – Concepts

Each row contains different but the same type of data.

Each column contains data corresponding to an attribute of the entities.

Since entities are related to each other by attributes, it is called Relational Model.

| Student ID | First Name | Last Name | Email | Major | Faculty |
|---|---|---|---|---|---|
| 200120 | Kate | West | kwest@email.com | Music | Arts |
| 200121 | Julie | McLain | jmclain@email.com | Finance | Business |
| 200122 | Tom | Erlich | terlich@email.com | Sculpture | Arts |
| 200123 | Mark | Smith | msmith@email.com | Biology | Science |
| 200124 | Jen | Foster | jfoster@email.com | Physics | Science |
| 200125 | Matt | Knight | mknight@email.com | Finance | Business |
| 200126 | Karen | Weaver | kweaver@email.com | Music | Arts |
| 200127 | John | Smith | jsmith@email.com | Sculpture | Arts |
| 200128 | Allison | Page | apage@email.com | History | Humanities |
| 200129 | Craig | Cambell | ccambell@email.com | Music | Arts |
| 200130 | Steve | Edwards | sedwards@email.com | Biology | Science |
| 200131 | Mike | Williams | mwilliams@email.com | Linguistics | Humanities |
| 200132 | Jane | Reid | jreid@email.com | Music | Arts |

# Must Have Features of Relationships

- The relationship has a singular name.
- Attributes are in a simple attribute structure.
- The relationship has at least one key.
- Attributes have singular names.
- The order of attributes is **unimportant**.
- Row order is **unimportant**.

# Features of Relationships

**Student**

| Student ID | First Name | Last Name | Email | Major | Faculty |
|---|---|---|---|---|---|
| 200120 | Kate | West | kwest@email.com | Music | Arts |
| 200121 | Julie | McLain | jmclain@email.com | Finance | Business |
| 200122 | Tom | Erlich | terlich@email.com | Sculpture | Arts |
| 200123 | Mark | Smith | msmith@email.com | Biology | Science |
| 200124 | Jen | Foster | jfoster@email.com | Physics | Science |
| 200125 | Matt | Knight | mknight@email.com | Finance | Business |
| 200126 | Karen | Weaver | kweaver@email.com | Music | Arts |
| 200127 | John | Smith | jsmith@email.com | Sculpture | Arts |
| 200128 | Allison | Page | apage@email.com | History | Humanities |
| 200129 | Craig | Cambell | ccambell@email.com | Music | Arts |
| 200130 | Steve | Edwards | sedwards@email.com | Biology | Science |
| 200131 | Mike | Williams | mwilliams@email.com | Linguistics | Humanities |
| 200132 | Jane | Reid | jreid@email.com | Music | Arts |

R="Student", attributes of the relation:
A1="StudentID",
A2="First_Name",
A3="Last_Name",
A4="Email",
A5="Major",
A6="Faculty"
The representation of the R relationship containing n attributes as R(A1, A2, …, An) is called the Relation Schema.

Student (Student_ID, First_Name, Last_Name, Email, Major,Faculty)

# Database Schema for RDBMS

1

A relationship schema may also include detailed information such as the data types of attributes (domain) and some other integrity constraints.

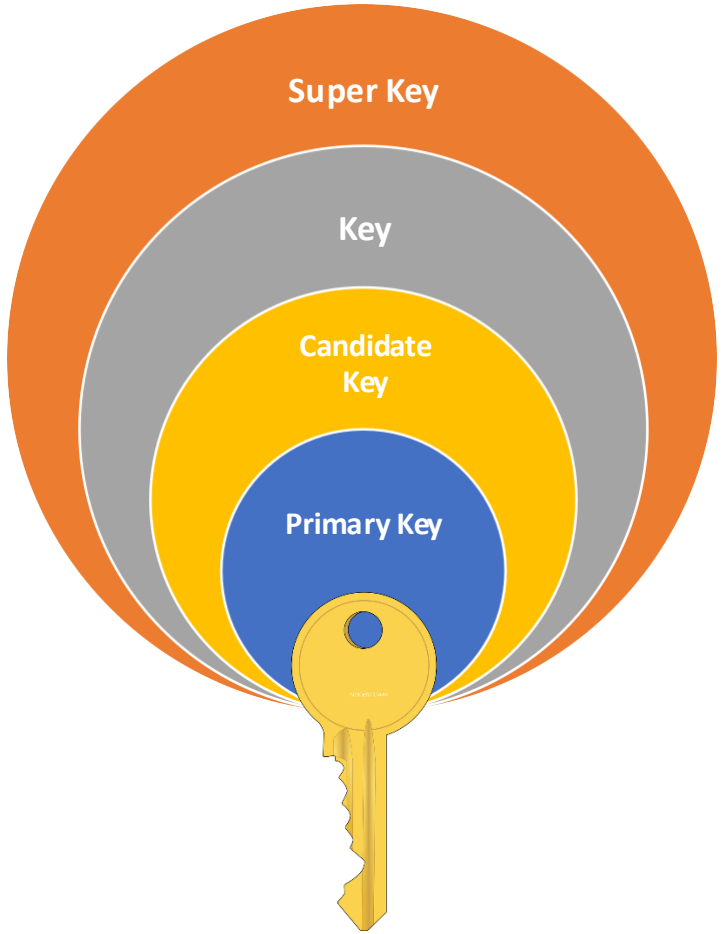
2

Student (studentID *integer primary key*, First_Name *character[20]*, Last_Name *character[20]*, Email *character[20]*, Major *character[20]*, Faculty *character[20]*, )

3

The set of relationship diagrams of ALL relationships in the database is called the Database Relationship Schema.

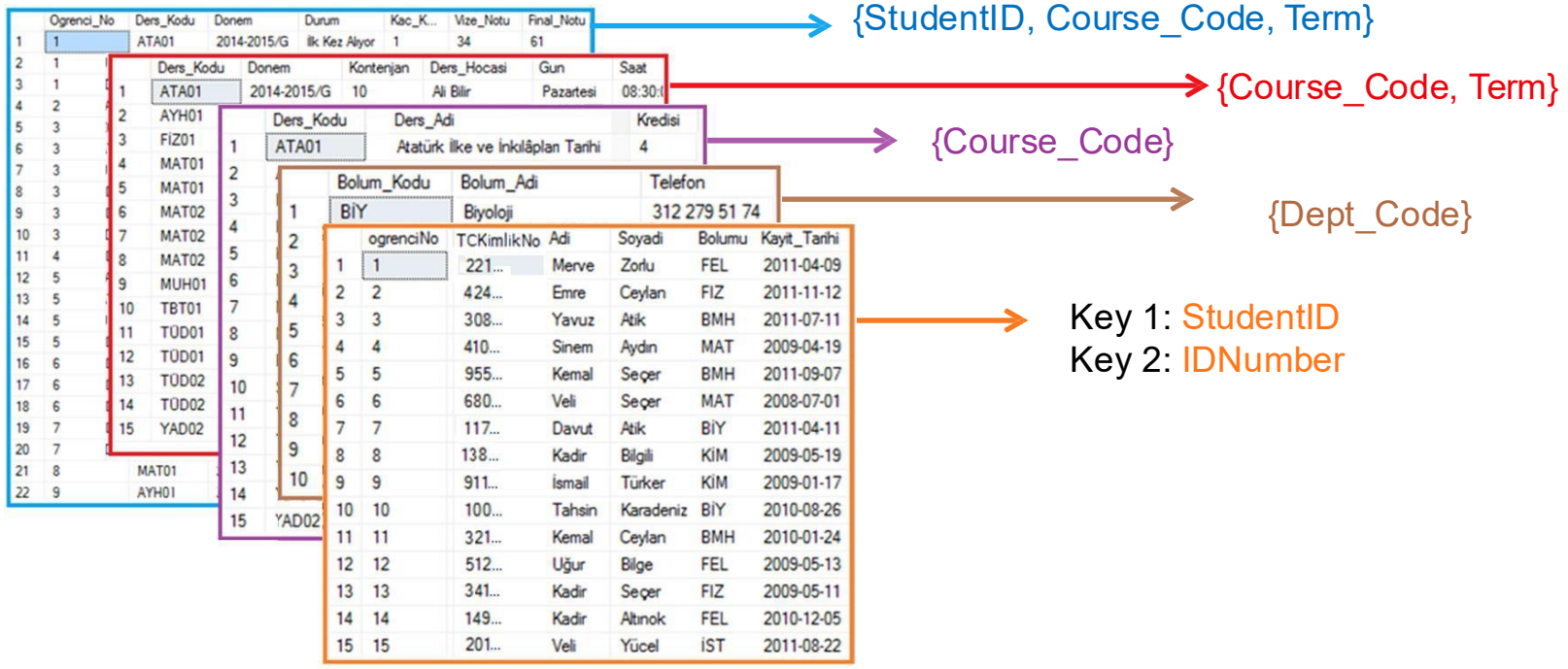
# Key Concept



# Key Concept

## KEY

A subgroup of attributes that make all the entities in the relationship different from each other and contain the least number of attributes is called a Key. There may be more than one.



# Super Key

## SUPER KEY

A set of attributes belonging to a relationship whose subset is a key is called a Super Key.

| Student ID | First Name | Last Name | Email | Major | Faculty |
|---|---|---|---|---|---|
| 200120 | Kate | West | kwest@email.com | Music | Arts |
| 200121 | Julie | McLain | jmclain@email.com | Finance | Business |
| 200122 | Tom | Erlich | terlich@email.com | Sculpture | Arts |
| 200123 | Mark | Smith | msmith@email.com | Biology | Science |
| 200124 | Jen | Foster | jfoster@email.com | Physics | Science |
| 200125 | Matt | Knight | mknight@email.com | Finance | Business |
| 200126 | Karen | Weaver | kweaver@email.com | Music | Arts |
| 200127 | John | Smith | jsmith@email.com | Sculpture | Arts |
| 200128 | Allison | Page | apage@email.com | History | Humanities |
| 200129 | Craig | Cambell | ccambell@email.com | Music | Arts |
| 200130 | Steve | Edwards | sedwards@email.com | Biology | Science |
| 200131 | Mike | Williams | mwilliams@email.com | Linguistics | Humanities |
| 200132 | Jane | Reid | jreid@email.com | Music | Arts |

Super key 1
Student_ID

Super key 2
{Student_ID, First_Name, Last_Name}

Super key 3
{Student_ID, Major}

Super key 4
{Student_ID, Major, Faculty}

# Candidate Key – Primary Key

- If there is more than one key in a relationship, each of key is called a **Candidate Key**.
- One of the candidate keys selected to express the relationship is called the **Primary Key**.

Primary Keys

| StudentId | firstName | lastName | courseId |
|---|---|---|---|
| L0002345 | Jim | Black | C002 |
| L0001254 | James | Harradine | A004 |
| L0002349 | Amanda | Holland | C002 |
| L0001198 | Simon | McCloud | S042 |
| L0023487 | Peter | Murray | P301 |
| L0018453 | Anne | Norris | S042 |

Candidate Keys

| StudentId | firstName | lastName | courseId |
|---|---|---|---|
| L0002345 | Jim | Black | C002 |
| L0001254 | James | Harradine | A004 |
| L0002349 | Amanda | Holland | C002 |
| L0001198 | Simon | McCloud | S042 |
| L0023487 | Peter | Murray | P301 |
| L0018453 | Anne | Norris | S042 |

# Foreign Key

## FOREIGN KEY

A field or set of fields in a relationship that **uniquely indicates a row** of the same or another relationship.



The primary key of the indicated table is used.

| Purpose of usage |
| --- |
| Linking a group of variables (record, tuple) with the same or different relationships |

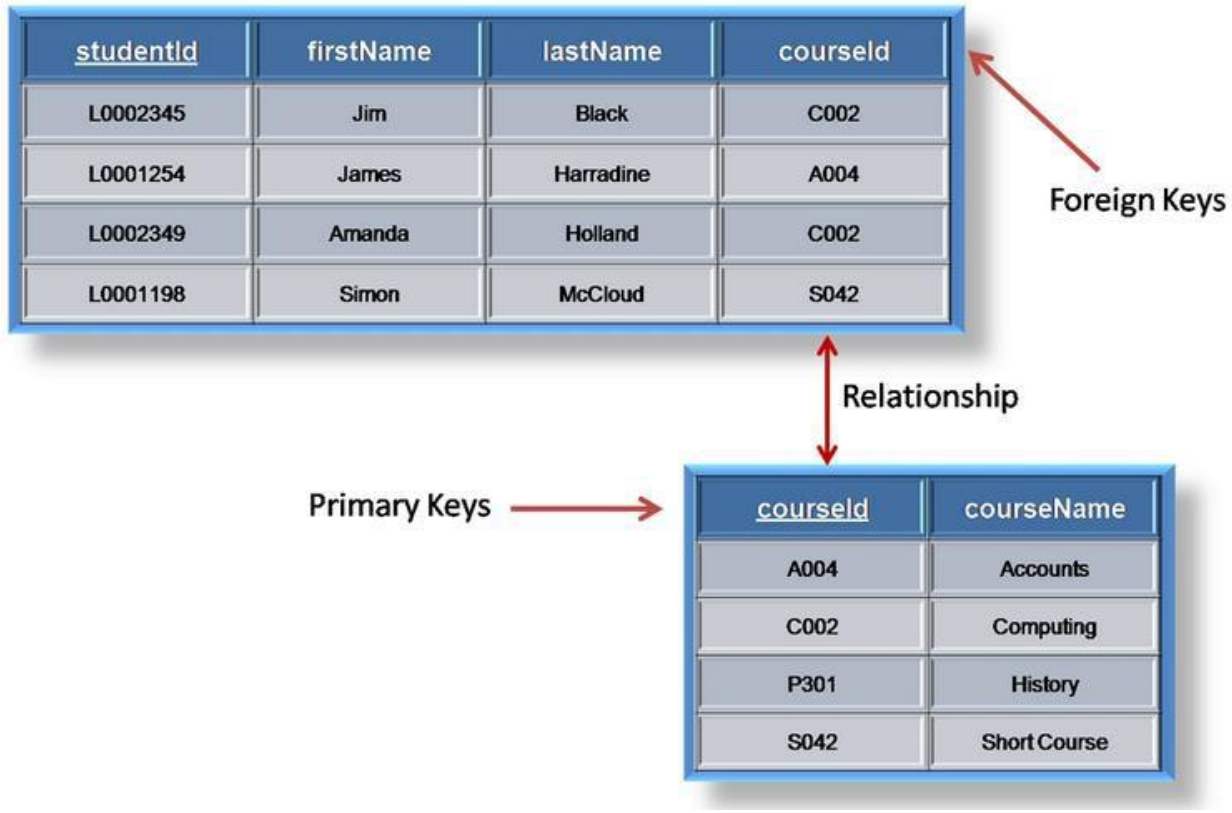
# Foreign Key

Some situations that require the use of foreign keys:

- For Student and Department relations: In which department does the student study?
- For Product and Machine relationships: On which machine was the product manufactured?
- For Personnel and Department relations: In which department does the personnel work?
- For Invoice and Customer relations: Which customer does the invoice belong to?
- For Customer and Address relationships: What is the Customer's shipping address?
- For Course and Department relations: Which department does the course belong to?
- For Citizen and Citizen relations: Who is the Citizen Kane's father?

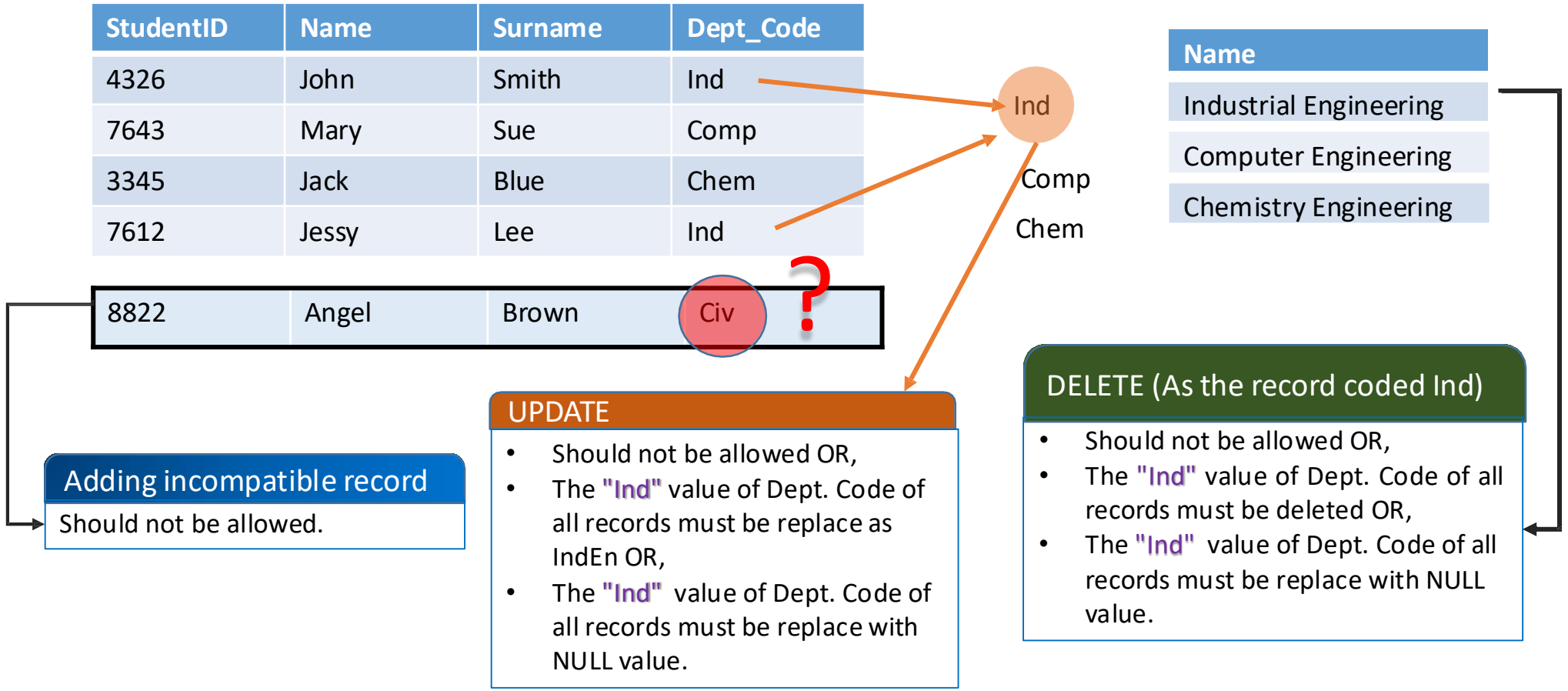
# Foreign Key

- The attribute named as "**courseid**" in the student table is a foreign key which the attribute used to access detailed information about the course that the student is studying.
- The course name which the student is studying is accessed from the course entity that contains the **foreign key value** named **courseId** in the student entity in its **primary key** named **courseId**.



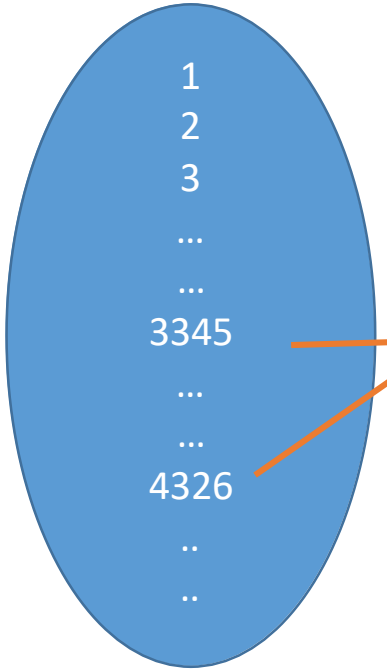
# Foreign Key

## Maintaining Database Consistency on **Foreign Keys.**



# Domain

Attribute Domain



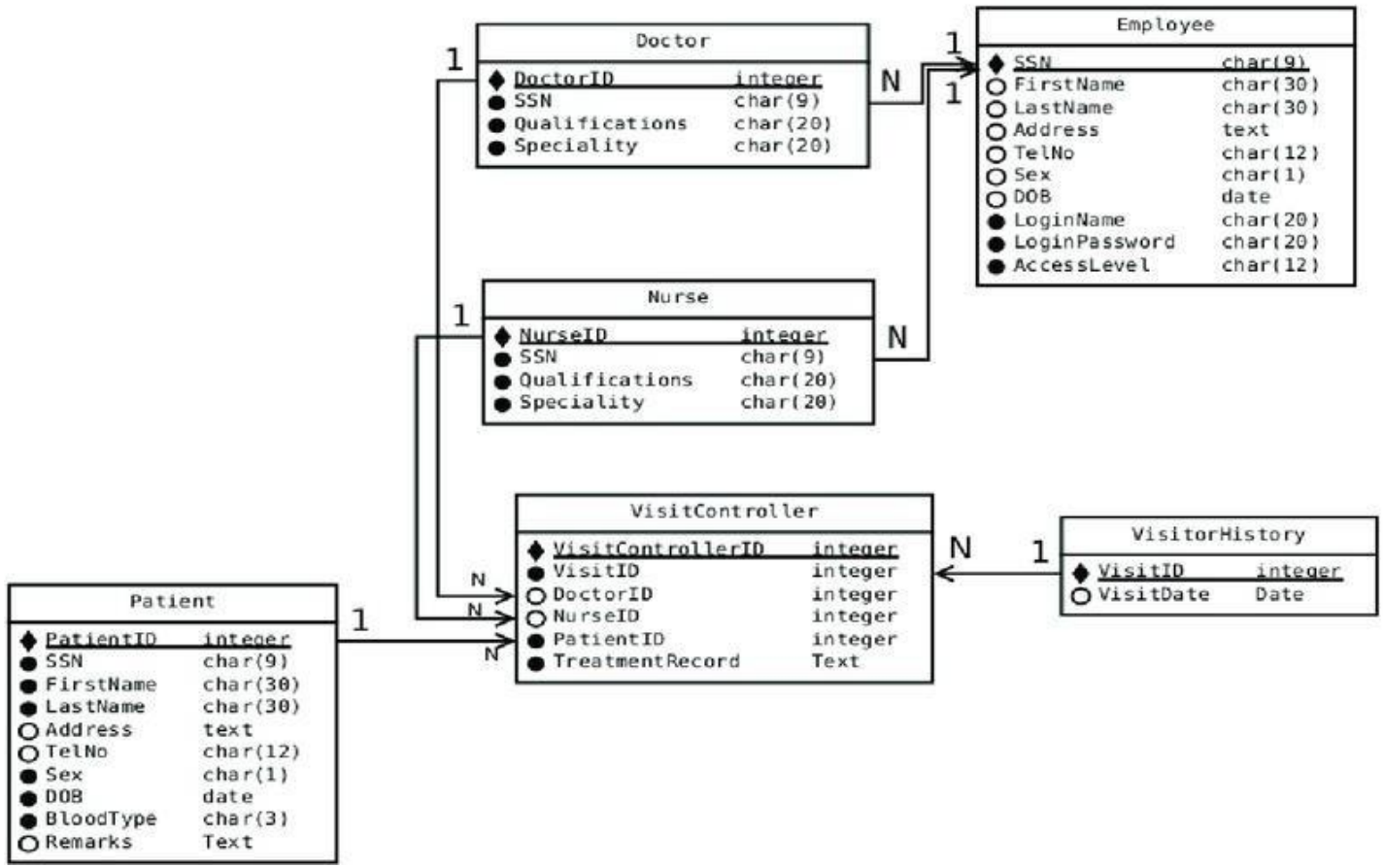

The set of all values matching the attribute.

| StudentID | Name | Surname | Dept_Code |
|---|---|---|---|
| 4326 | John | Smith | Ind |
| 7643 | Mary | Sue | Comp |
| 3345 | Jack | Blue | Chem |
| 7612 | Jessy | Lee | Ind |

Student(StudentID *integer* **key,** Name stirng[20], Surname string[20], Dept_Code string [10]

Domain: A set that contains all information of character type up 1 to 10 in length.

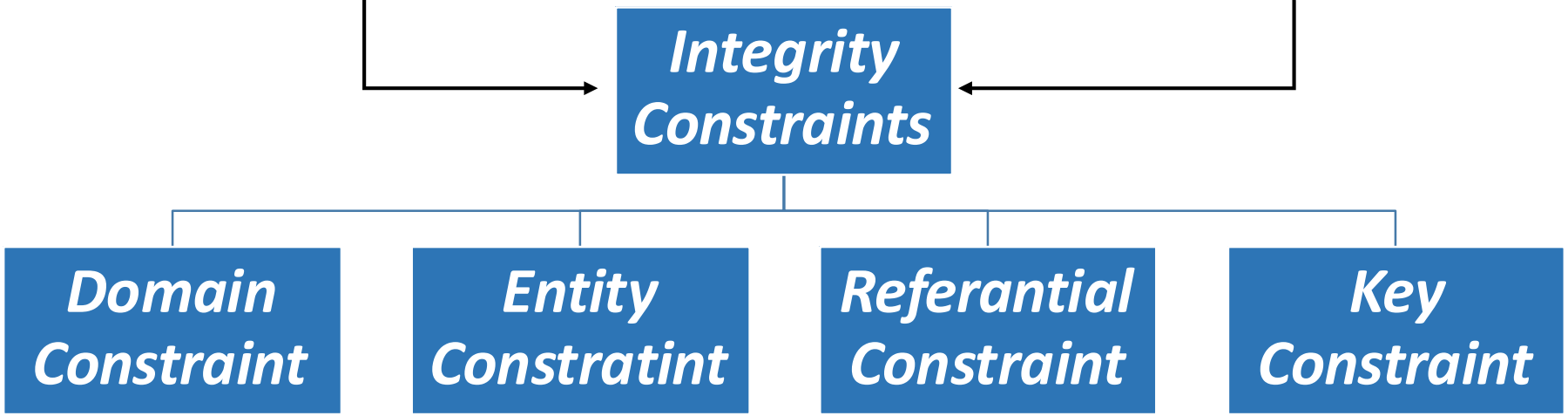
# Database Schema



# Integrity Constraints

Q: When will it happen?
A: While creating a relationship

Q: When is it checked?
A: When changing instance relation.
(Insert, Update, Delete)



# Domain Constraint

- Domain constraint can be defined as the definition of a valid set of values for an attribute.
- The data type of domain includes string, character, integer, time, date, currency, etc. The value of the attribute must be available in the corresponding domain.

| ID | NAME | SEMENSTER | AGE |
|---|---|---|---|
| 1000 | Tom | 1st | 17 |
| 1001 | Johnson | 2nd | 24 |
| 1002 | Leonardo | 5th | 21 |
| 1003 | Kate | 3rd | 19 |
| 1004 | Morgan | 8th | A |

Not allowed. Because AGE is an integer attribute

# Entity Constraint

- The entity constraint states that primary key value can't be null.
- This is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.
- A table can contain a null value other than the primary key field.

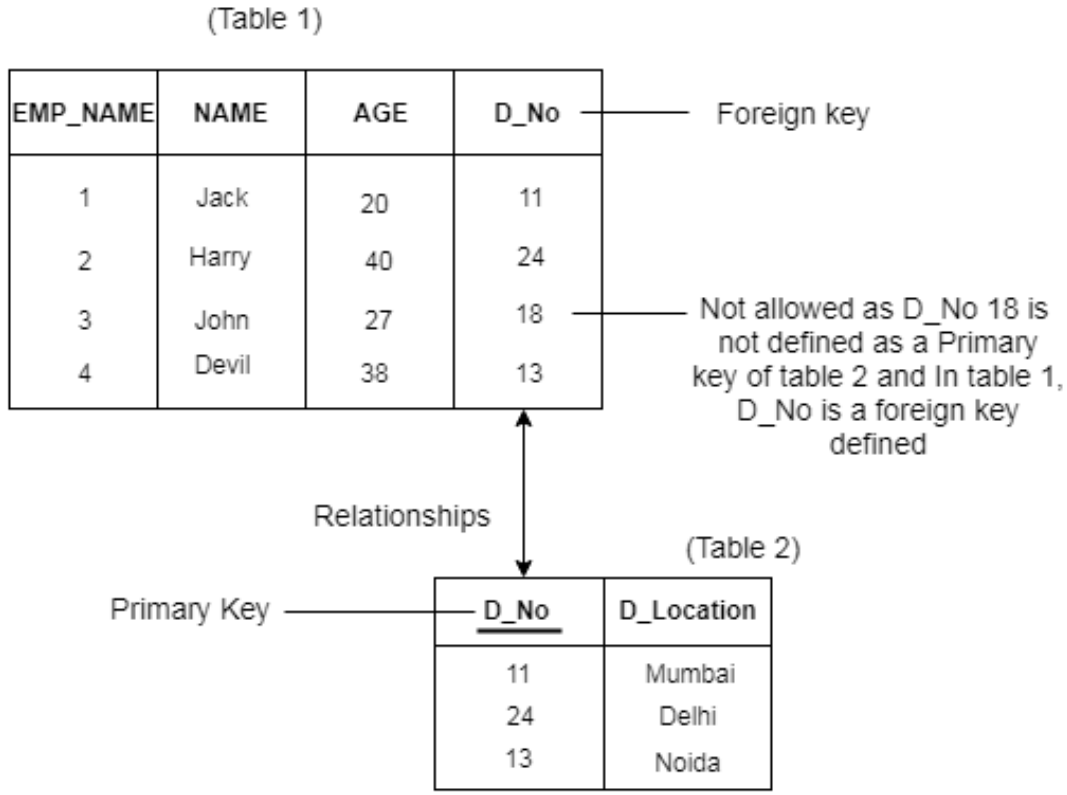
**EMPLOYEE**

| EMP_ID | EMP_NAME | SALARY |
|---|---|---|
| 123 | Jack | 30000 |
| 142 | Harry | 60000 |
| 164 | John | 20000 |
| | Jackson | 27000 |

Not allowed as primary key can't contain a NULL value

# Referential Constraint

- A referential constraint is specified between two tables.
- In the Referential integrity constraints, if a foreign key in Table 1 refers to the Primary Key of Table 2, then every value of the Foreign Key in Table 1 must be null or be available in Table 2.



# Key Constraint

- Key Constraint: The rules that are to be followed while entering data into columns of the database table.
- The key constraint ensure that the data entered in the columns by the user falls within the criteria specified by the condition.

| ID | NAME | SEMENSTER | AGE |
|---|---|---|---|
| 1000 | Tom | 1st | 17 |
| 1001 | Johnson | 2nd | 24 |
| 1002 | Leonardo | 5th | 21 |
| 1003 | Kate | 3rd | 19 |
| 1002 | Morgan | 8th | 22 |

Not allowed. Because all row must be unique

# Key Constraint

- Types of key constraint in DBMS:

- **UNIQUE** : provides a unique/distinct values to specified columns.
- **NOT NULL:** ensures that the specified column doesn't contain a NULL value.
- **DEFAULT:** provides a default value to a column if none is specified.
- **CHECK:** checks for the predefined conditions before inserting the data inside the table.

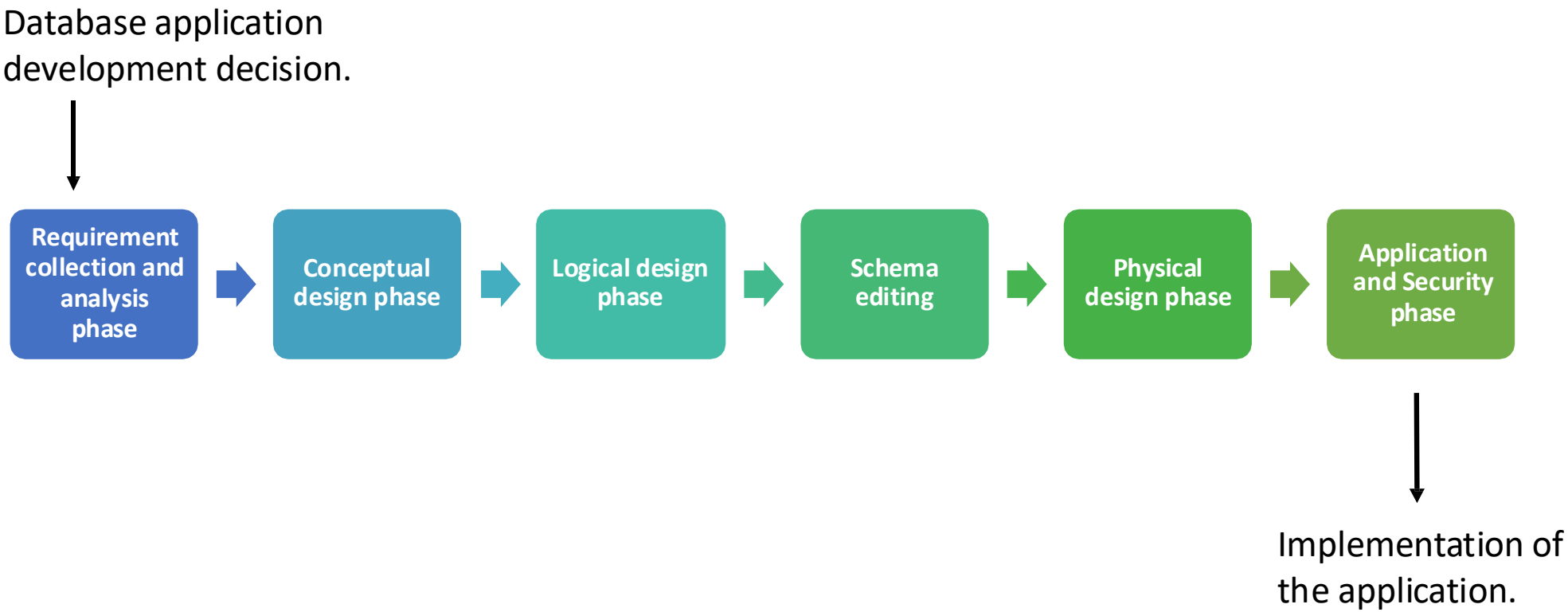
# DATABASE DESIGN

# WHAT IS NEEDED FOR DATABASE DESIGN?

- To develop database applications in a successful manner:

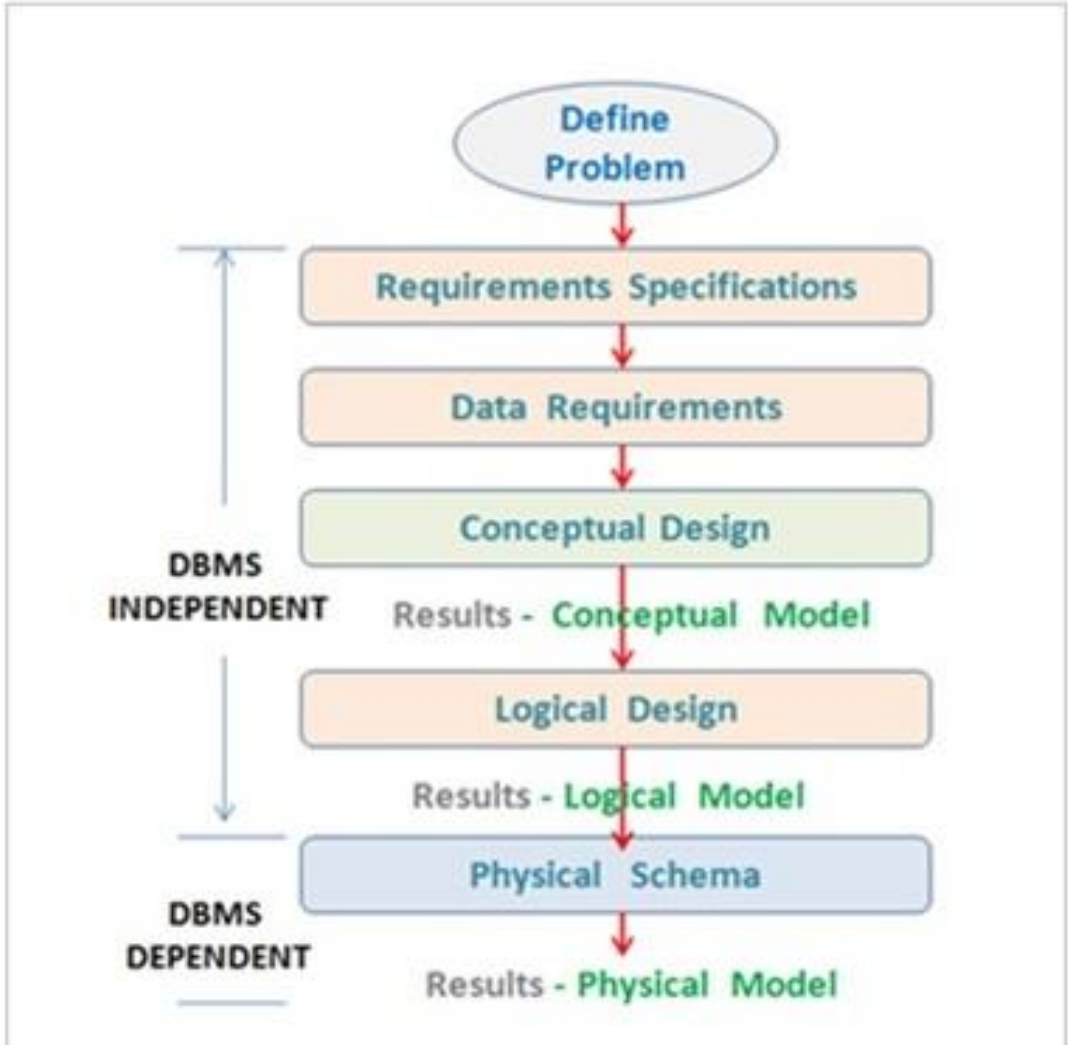
- ☑ ***Collecting required resources***
- ☑ ***Structured approach to project management.***
- ☑ ***User training and user involvement.***
- ☑ ***Team references.***
- ☑ ***Rigorous technical specification.***
- ☑ ***Similar software advantages and disadvantages.***
- ☑ ***Product delivery and decision-making method.***
- ☑ ***Senior management support.***

# The Process of Database Design



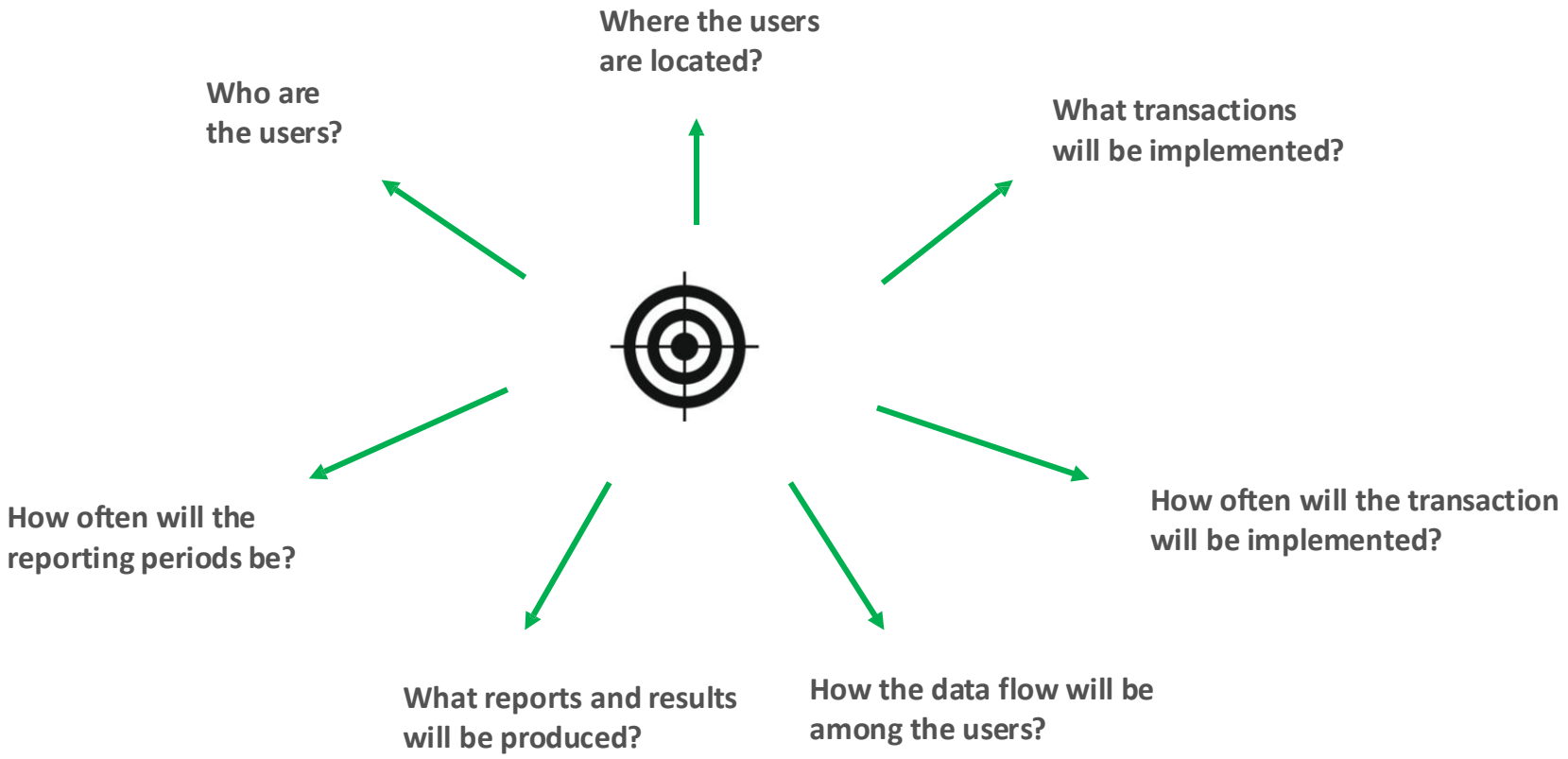
# The Process of Database Design-2

Pay attention to DBMS-Dependent and DBMS-independent steps.



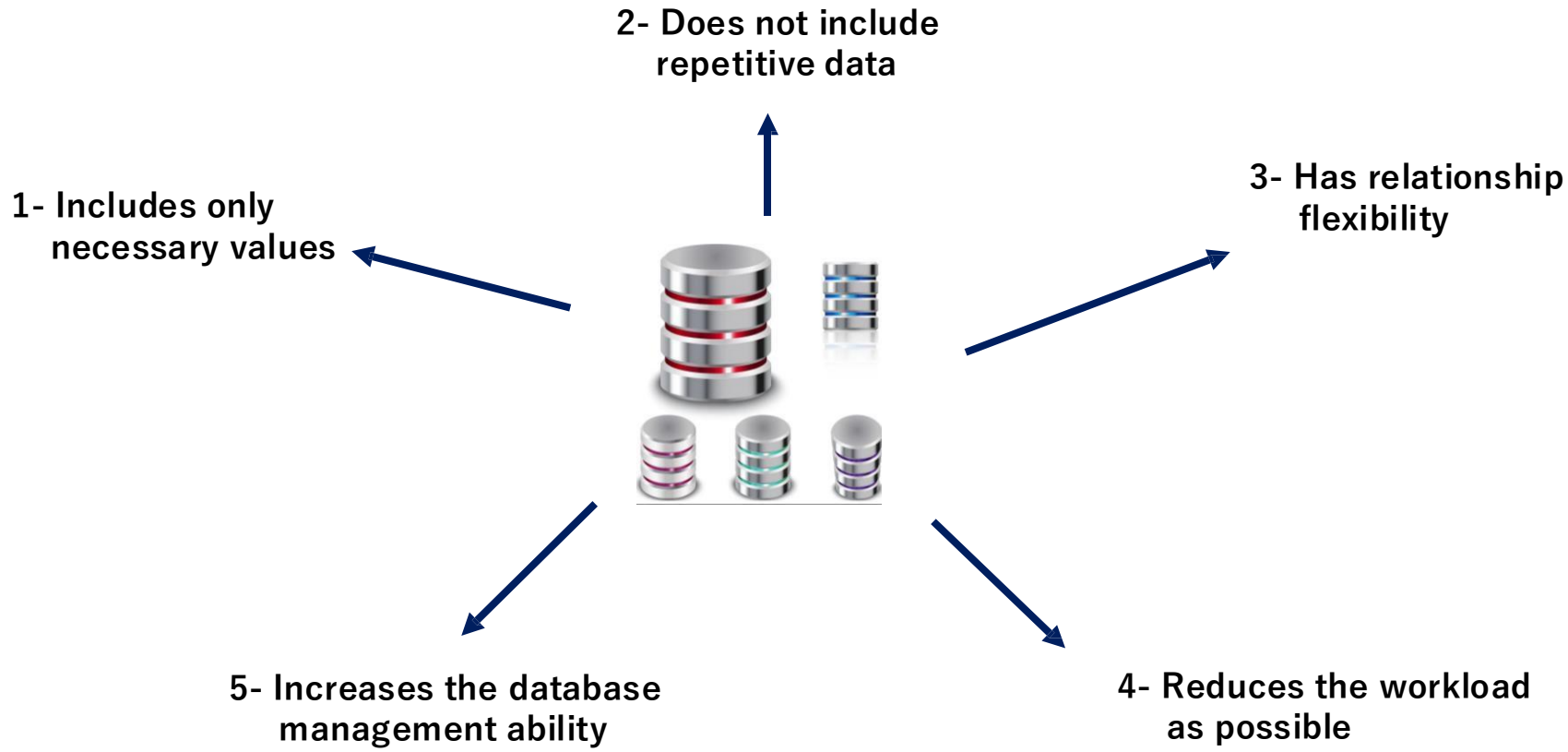
# Requirement Analysis

- The prerequisite for developing a quality DB application is to determine the application purpose with sufficient scope and clarity.



# Requirement Analysis

**Dimensions that improve database design quality.**



# 1. Determination of Requirements

•**Importance of Definition:** A correct solution cannot be generated from an incorrectly defined problem; learning by trial-and-error is the most expensive method.

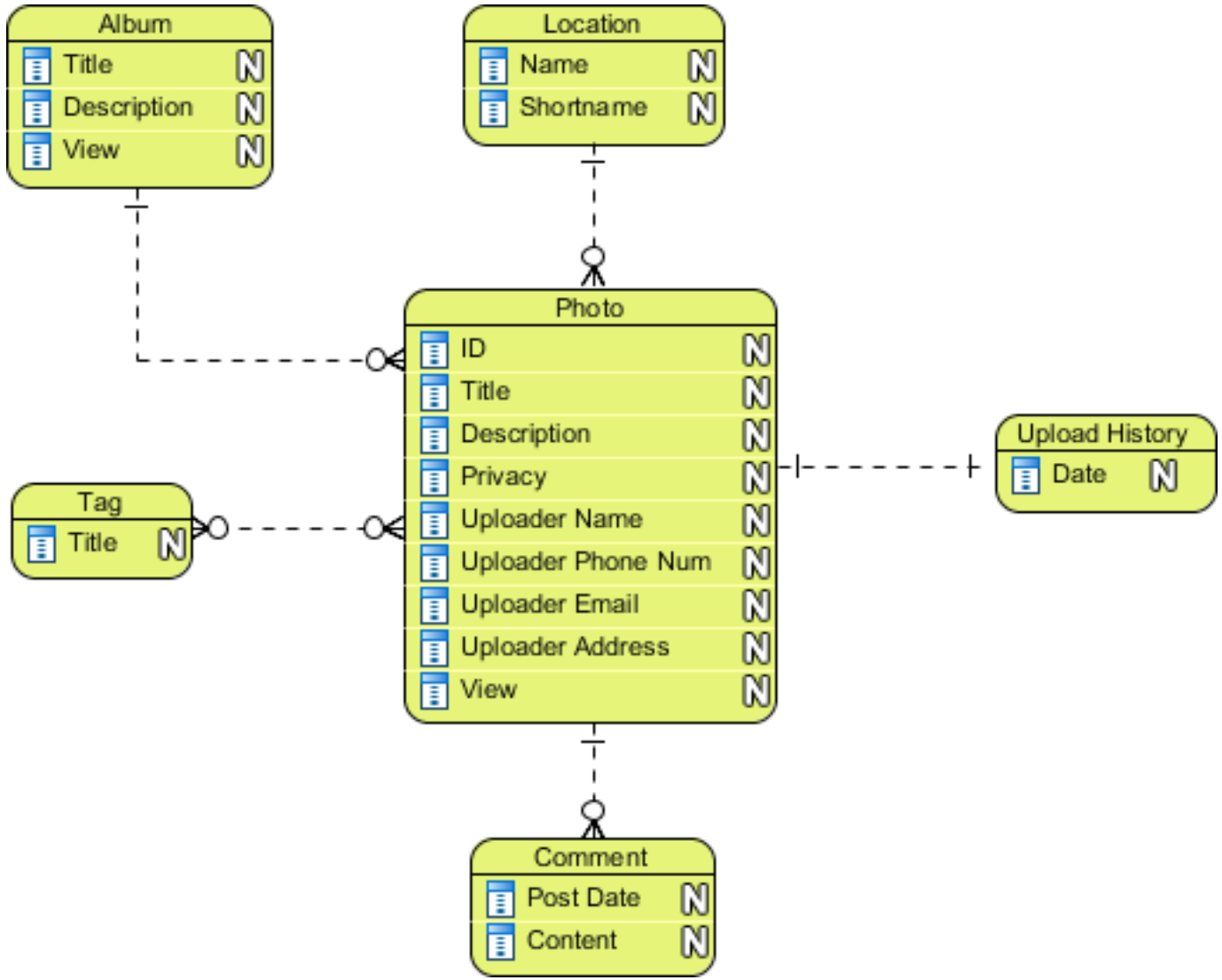
•**Team Formation:** A project team consisting of technical experts, end-users, and managers must be established at this stage.

•**Scope Consensus:** The team must agree on key elements: system users, data inputs, operational processes, outputs, and required reports.

•**Planning & Success:** A project schedule must be created, and specific success factors for all activities must be clearly defined.

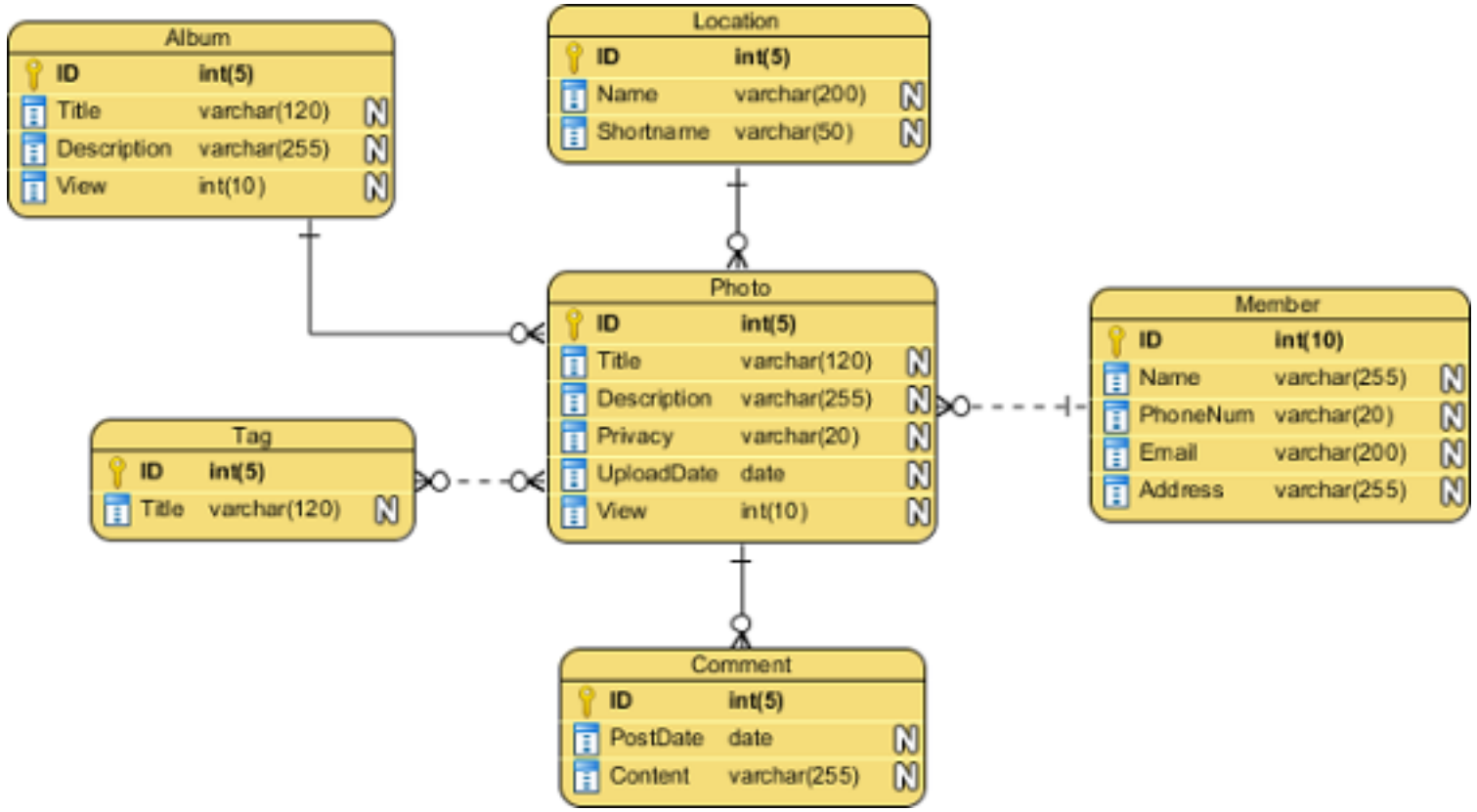
# Conceptual Design

- It is the first stage the transformation of requirements into a working DB.
- It should be a concise design free from details.
- Data that will be stored in DB, relations between them, restrictions etc. should be expressed conceptually.
- The conceptual design should be capable of transforming into the data model to be used.



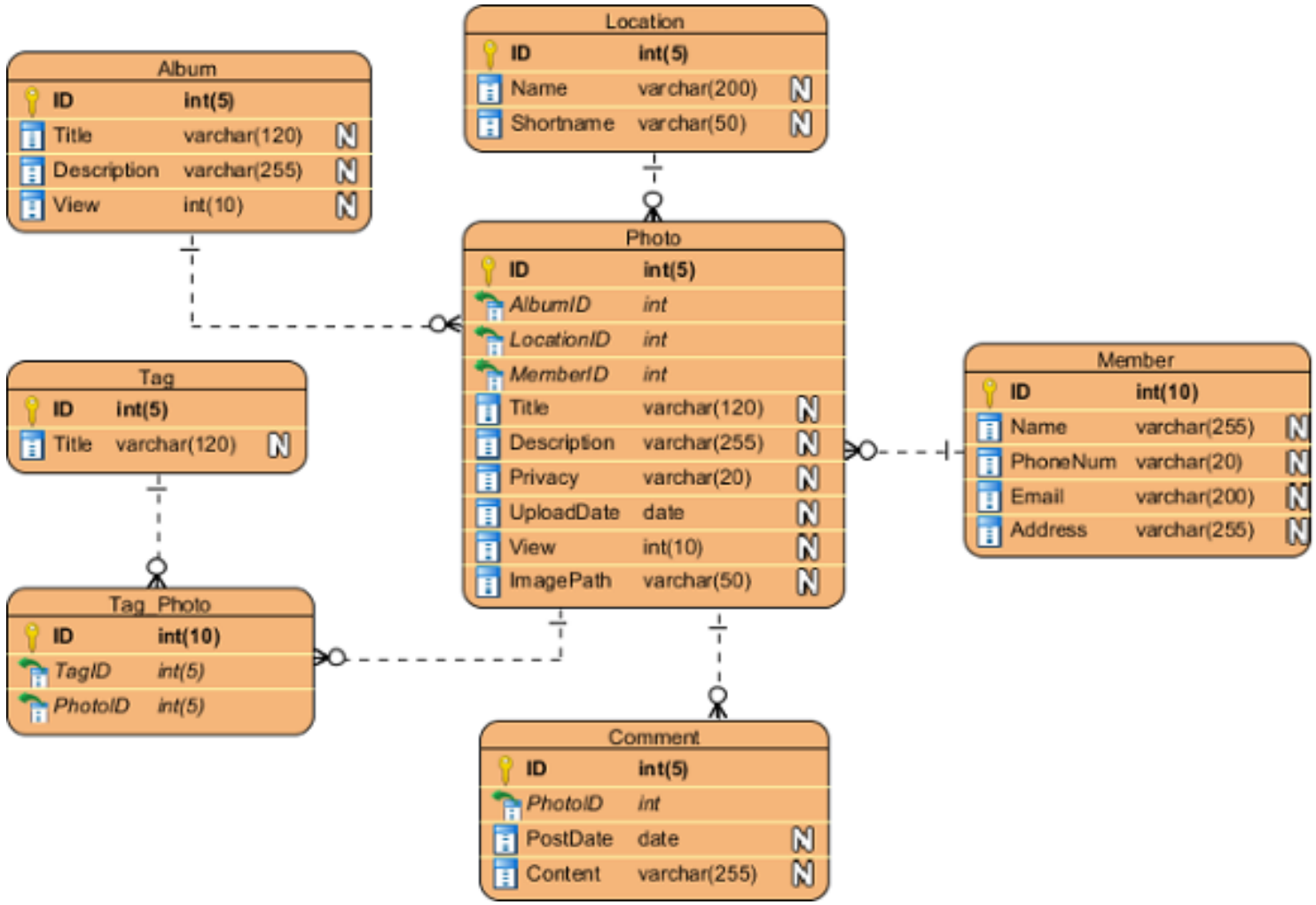
# Logical Design

- A conceptual design can be converted into a database schemas in a structure, that is suitable for the data model to be used.
- The logical design should define a database that can work flawlessly in the DBMS tool to be used.



# Physical Design

- It focuses on performance and space utilization.
- For this purpose, the features and components that the database should physically carry are determined and they are physically created.
- Such as; attribute design, physical register design, index design, buffer-oriented decisions, transaction design, etc.



# Schema Editing

- By considering every elements in the logical design, possible problems are determined and the design is revised to eliminate possible conflicts in the future.
- **Normalization** is the process of organizing data in a database. It includes creating tables and establishing relationships between those tables according to rules designed both to protect the data and to make the database more flexible by eliminating redundancy and inconsistent dependency. Normalization method can be used for schema editing in relational DBMSs.
- The normalization method rearranges the relations to bring some desired attributes to the design.

# Application and Security Design

## Includes some subjects such as below:

- Units that will use the application,
- Processes to be run by the units,
- Process design,
- Inputs and outputs of processes,
- Inter-process relations,
- Determining the DB objects that each process can and can't reach etc.

# THE END

CHECK OUT COURSE WEBSITE: HTTPS://GAZI-END-MUH.GITHUB.IO

GOT ANY QUESTIONS LATER?

SEND ME AN EMAIL: DR.ERCAN.EZIN@GMAIL.COM