DBMS Unit-II

By:
Mani Butwall
Asst. Prof. (CSE)

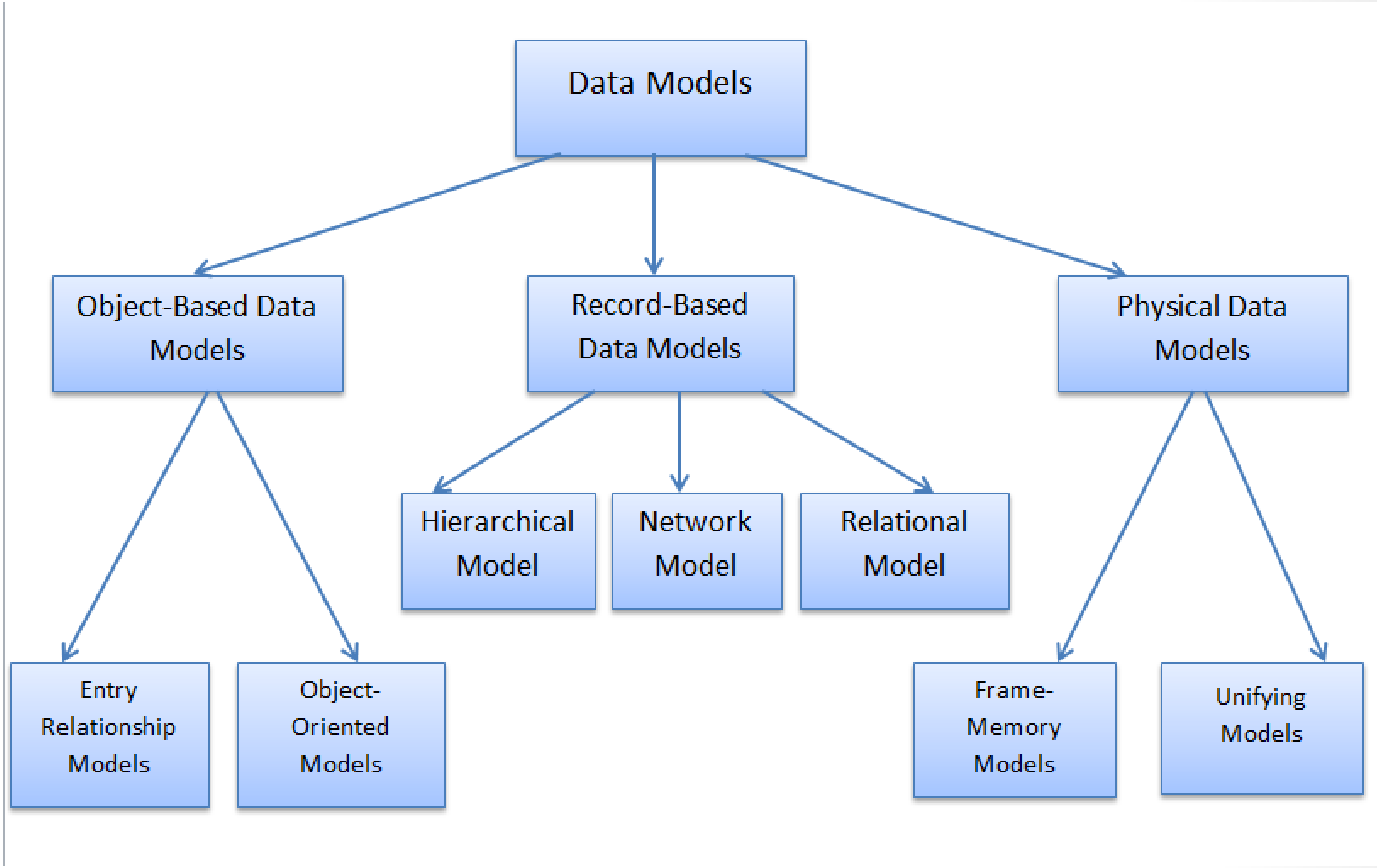
Contents

- Data models
- Entity-relationship model
- Network model
- Relational Model
- Object oriented data models
- Integrity constraints
- Data manipulation operations

Data Model

- A Database model defines the logical design and structure of a database and defines how data will be stored, accessed and updated in a database management system. While the **Relational Model** is the most widely used database model, there are other models too:

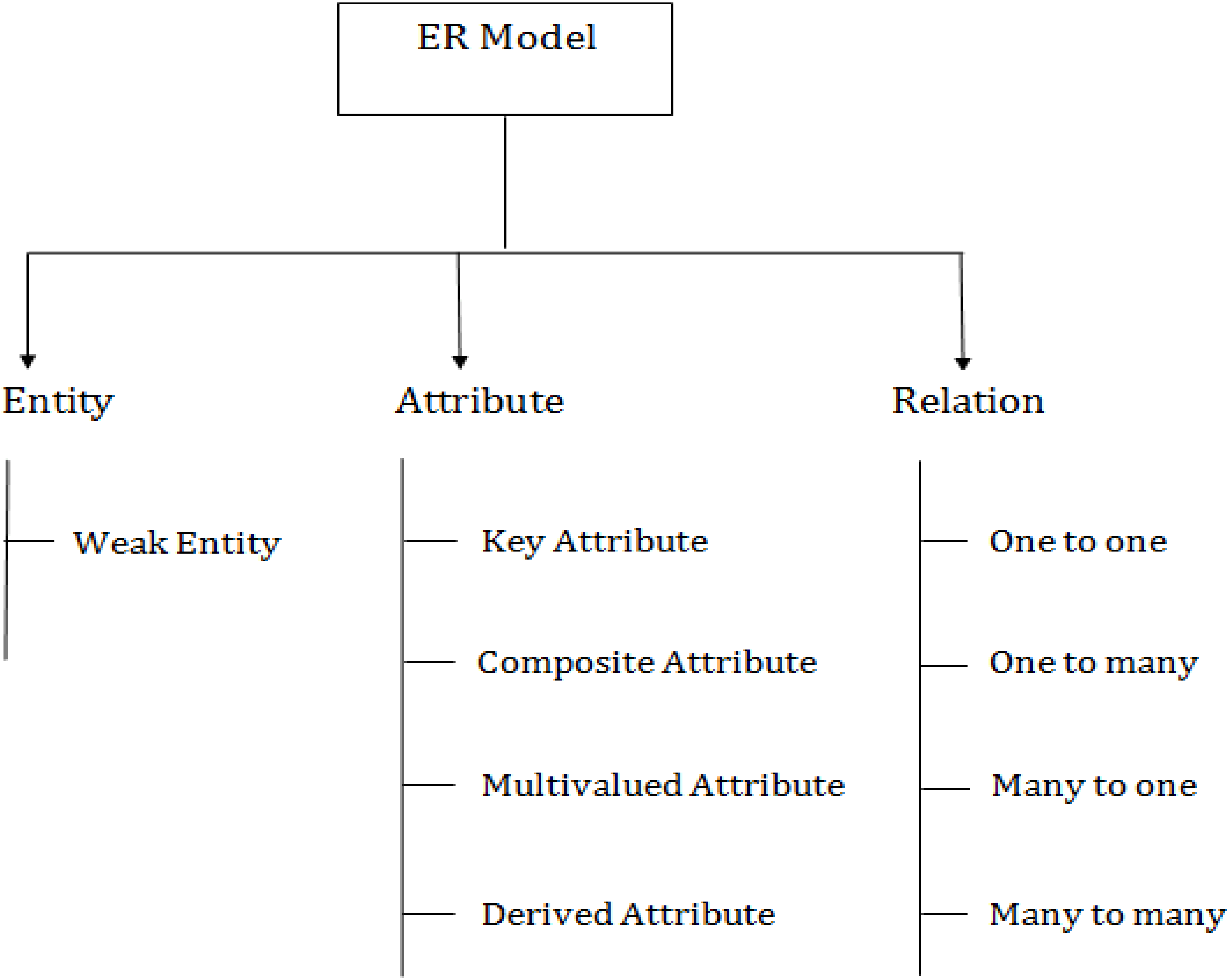
1. Hierarchical Model
2. Network Model
3. Entity-relationship Model
4. Relational Model



Data Models
Object-Based Data Models
Record-Based Data Models
Physical Data Models
Hierarchical Model
Network Model
Relational Model
Entry Relationship Models
Object-Oriented Models
Frame-Memory Models
Unifying Models

ER Model

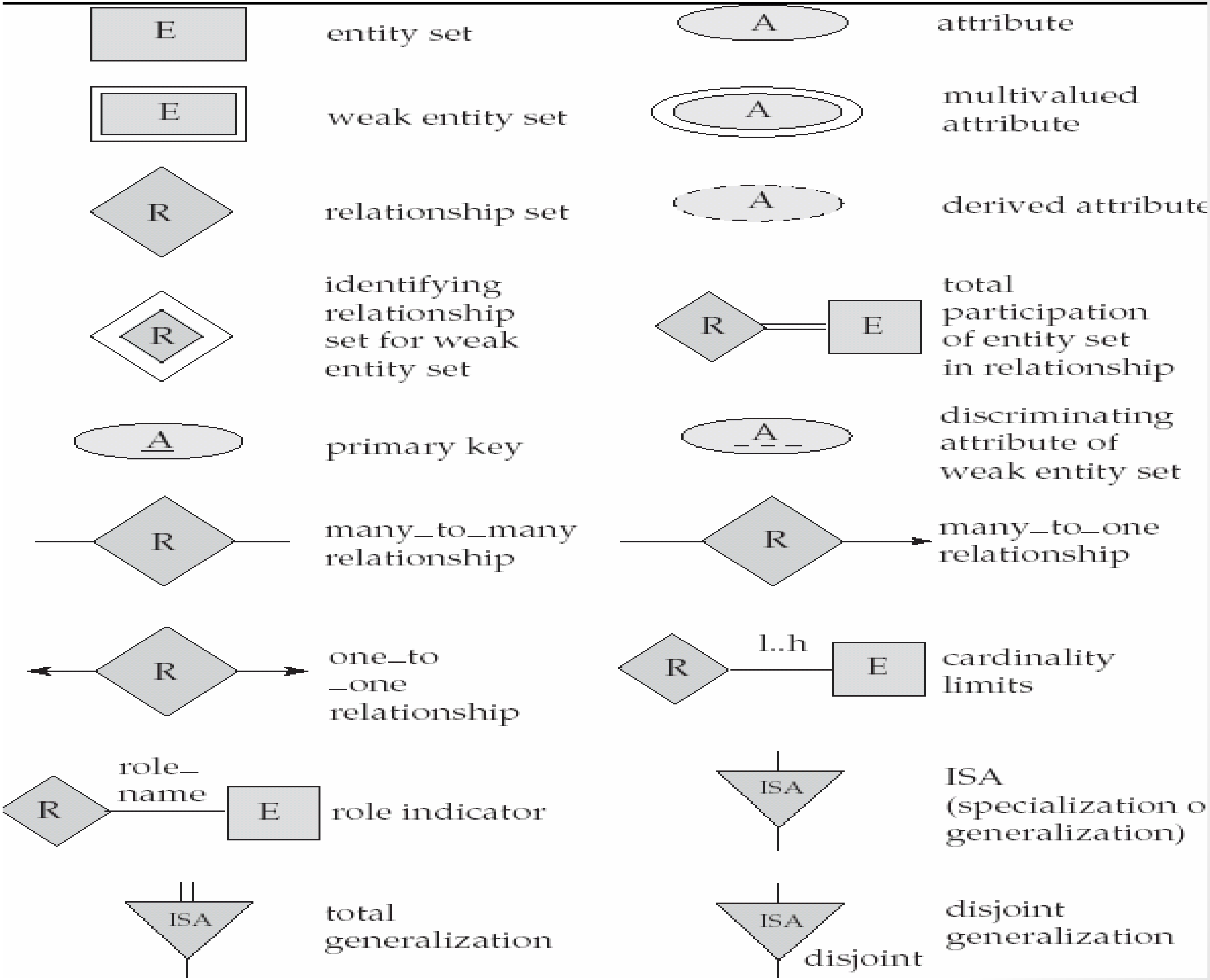
- An **Entity–relationship model (ER model)** describes the structure of a database with the help of a diagram, which is known as **Entity Relationship Diagram (ER Diagram)**. An ER model is a design or blueprint of a database that can later be implemented as a database.
- The main components of E-R model are: **entity set and relationship set.**
- An ER diagram shows the relationship among entity sets. ER diagram shows the complete logical structure of a database.



- ER Model
  - Entity
    - Weak Entity
  - Attribute
    - Key Attribute
    - Composite Attribute
    - Multivalued Attribute
    - Derived Attribute
  - Relation
    - One to one
    - One to many
    - Many to one
    - Many to many

Symbols Used

1. Rectangles represent entity sets.
2. Diamonds represent relationship sets.
3. Lines link attributes to entity sets and entity sets to relationship sets.
4. Ellipses represent attributes
5. Double ellipses represent multivalued attributes.
6. Dashed ellipses denote derived attributes.
7. Underline indicates primary key attributes



entity set

attribute

weak entity set

multivalued attribute

relationship set

derived attribute

identifying relationship set for weak entity set

total participation of entity set in relationship

primary key

discriminating attribute of weak entity set

many_to_many relationship

many_to_one relationship

one_to _one relationship

cardinality limits

role indicator

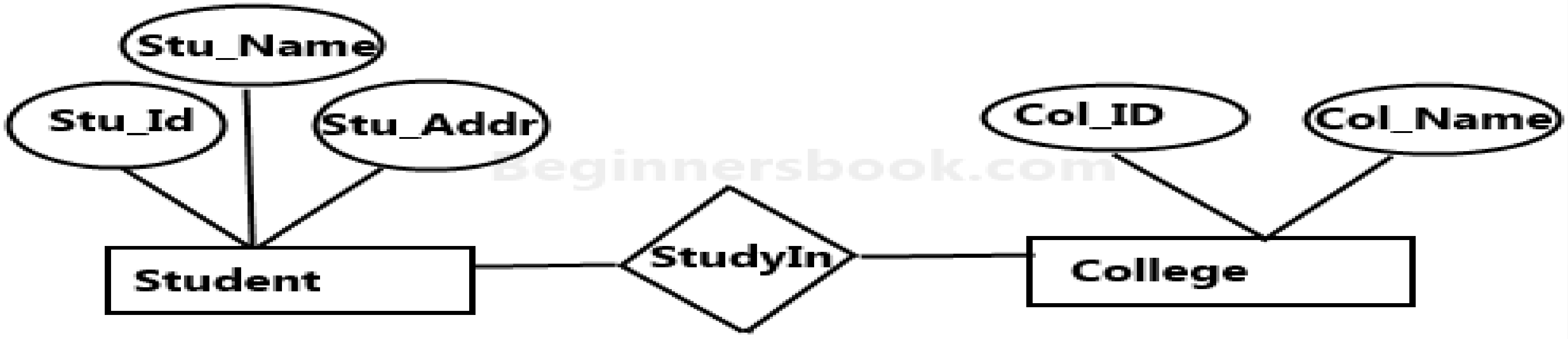
ISA (specialization o generalization)

total generalization

disjoint generalization

Entity

- An **entity** is an object that exists and is distinguishable from other objects.
- An entity is represented as rectangle in an ER diagram. For example: In the following ER diagram we have two entities Student and College and these two entities have many to one relationship as many students study in a single college.



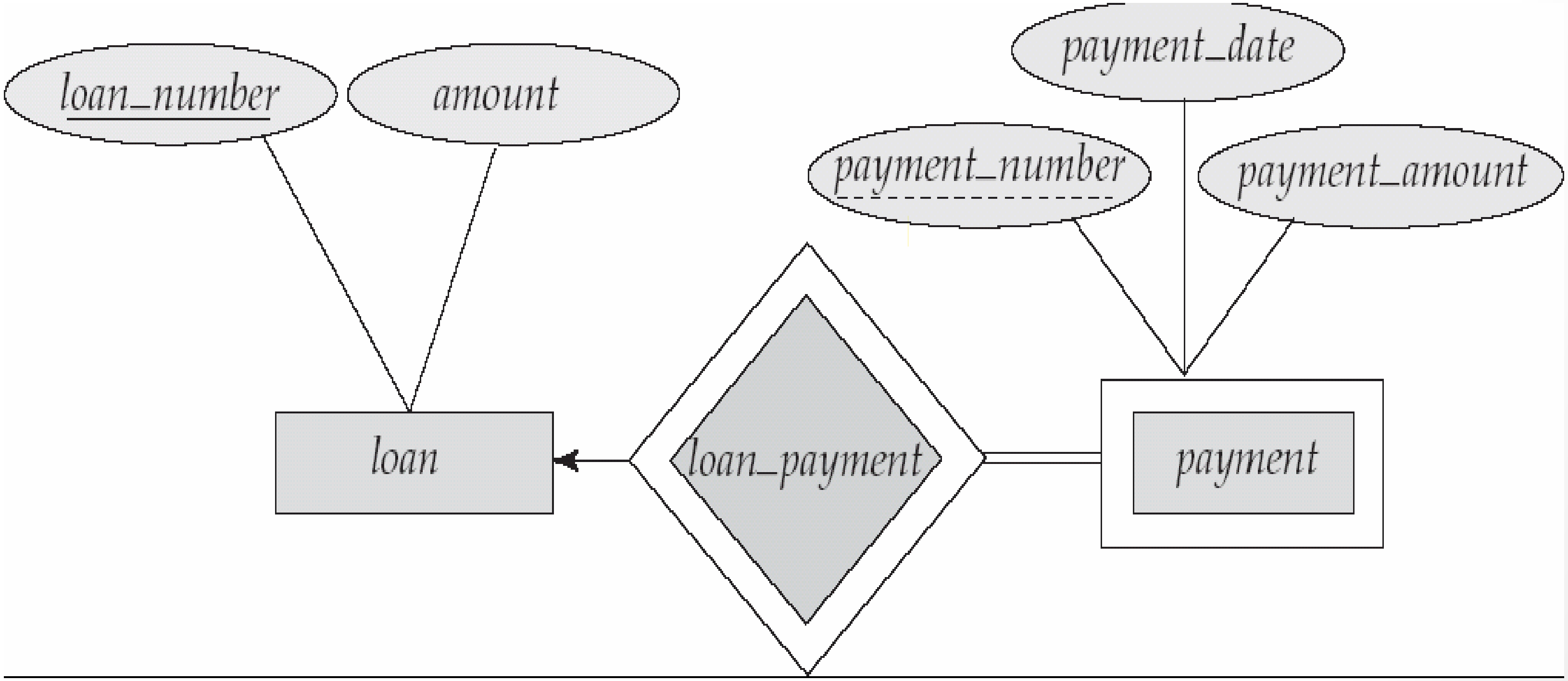
Sample E-R Diagram

- **Examples of entities:**

1. **Person:** Employee, Student, Patient
2. **Place:** Store, Building
3. **Object:** Machine, product, and Car
4. **Event:** Sale, Registration, Renewal
5. **Concept:** Account, Course

Weak Entity Set

- An entity that cannot be uniquely identified by its own attributes and relies on the relationship with other entity is called weak entity. The weak entity is represented by a double rectangle. For example – a payment number can not be uniquely identified unless it is associated with other entity like loan or deposit.



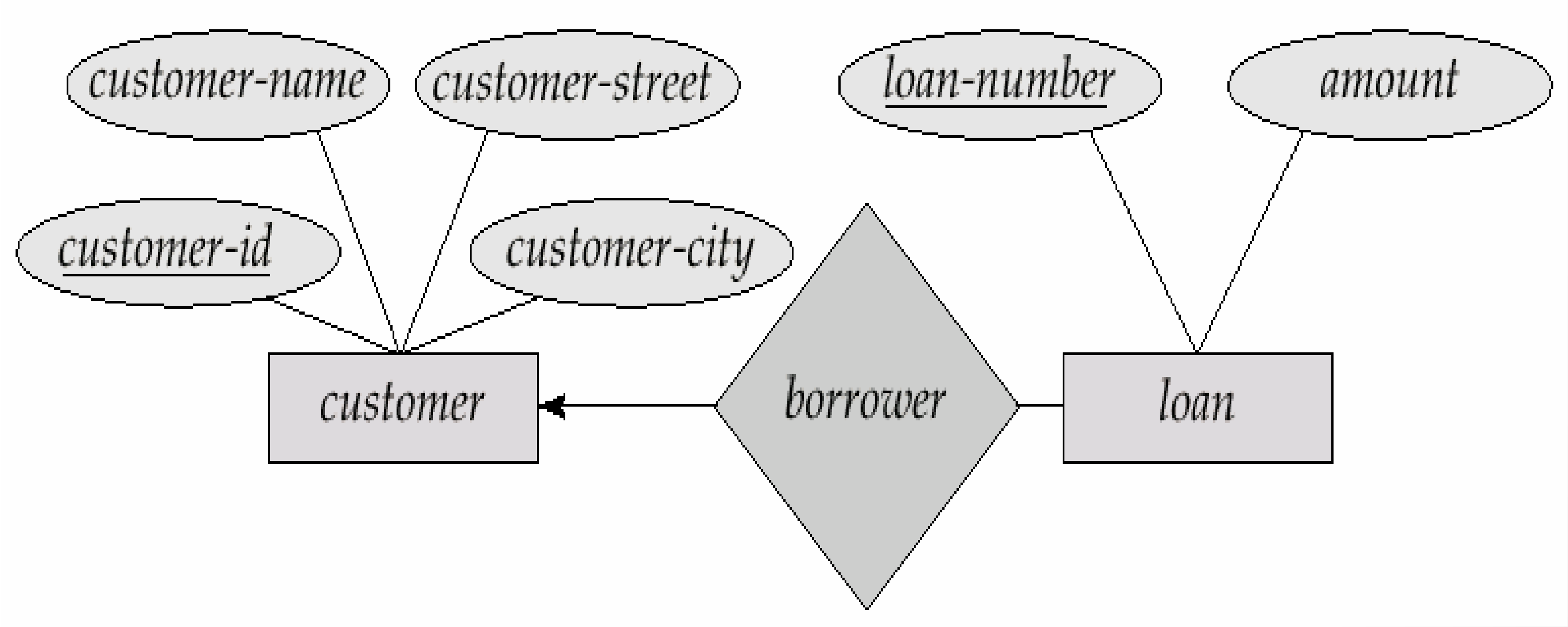
Attribute

- An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:

1. Key attribute
2. Composite attribute
3. Multivalued attribute
4. Derived attribute

Key Attribute

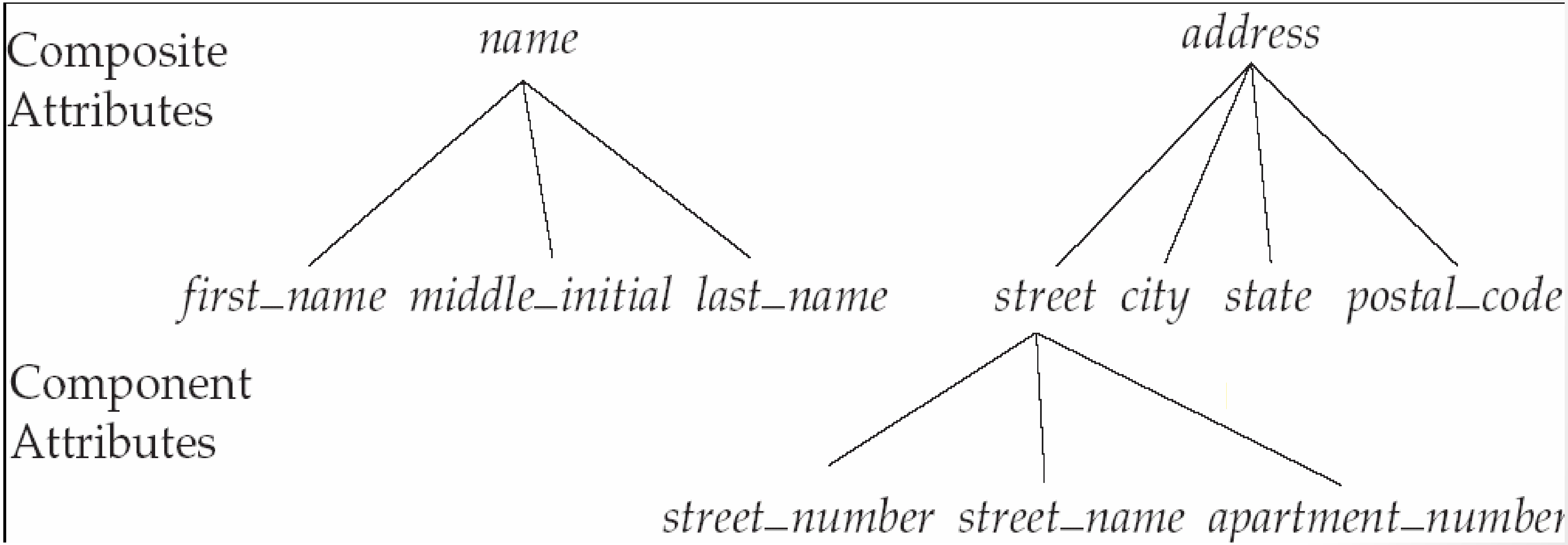
- A key attribute can uniquely identify an entity from an entity set. For example, student roll number can uniquely identify a student from a set of students. Key attribute is represented by oval same as other attributes however the **text of key attribute is underlined**.



Mani Butwall, CSE ITM University

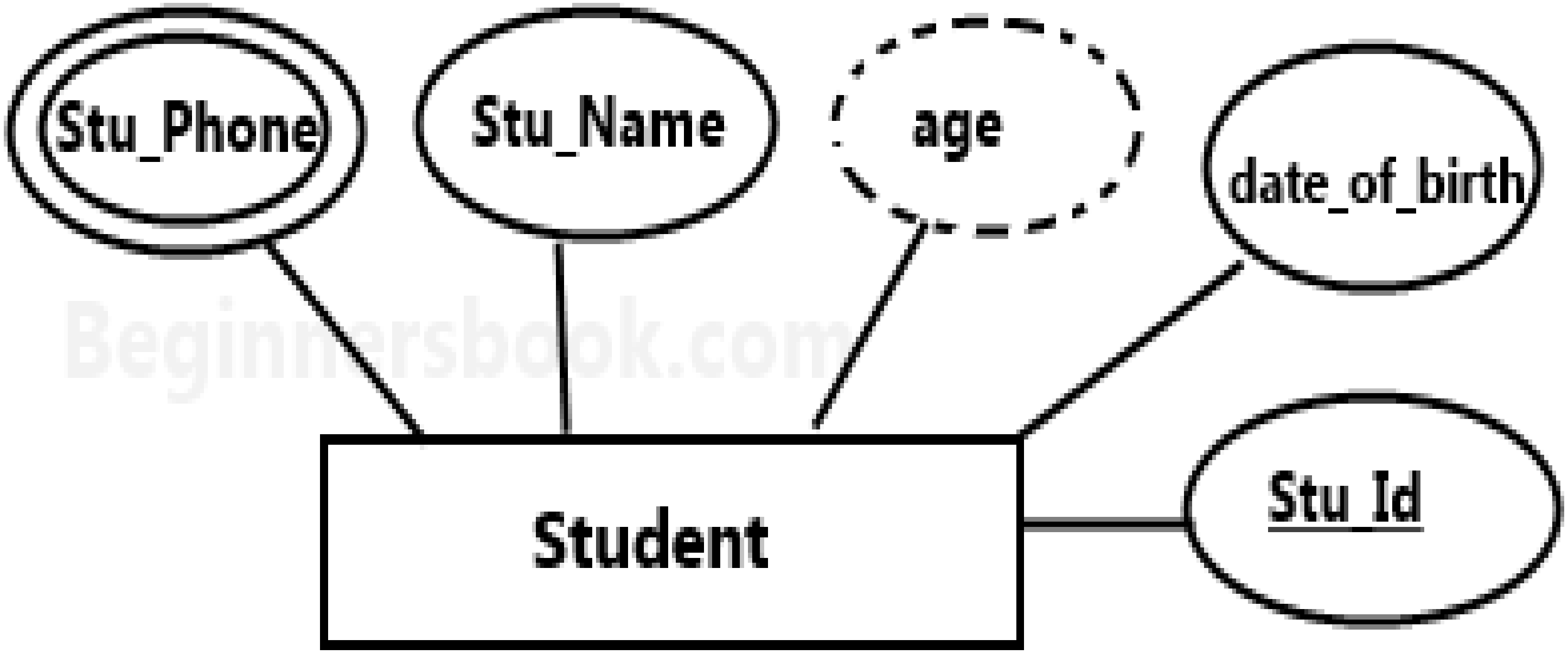
Composite Attribute

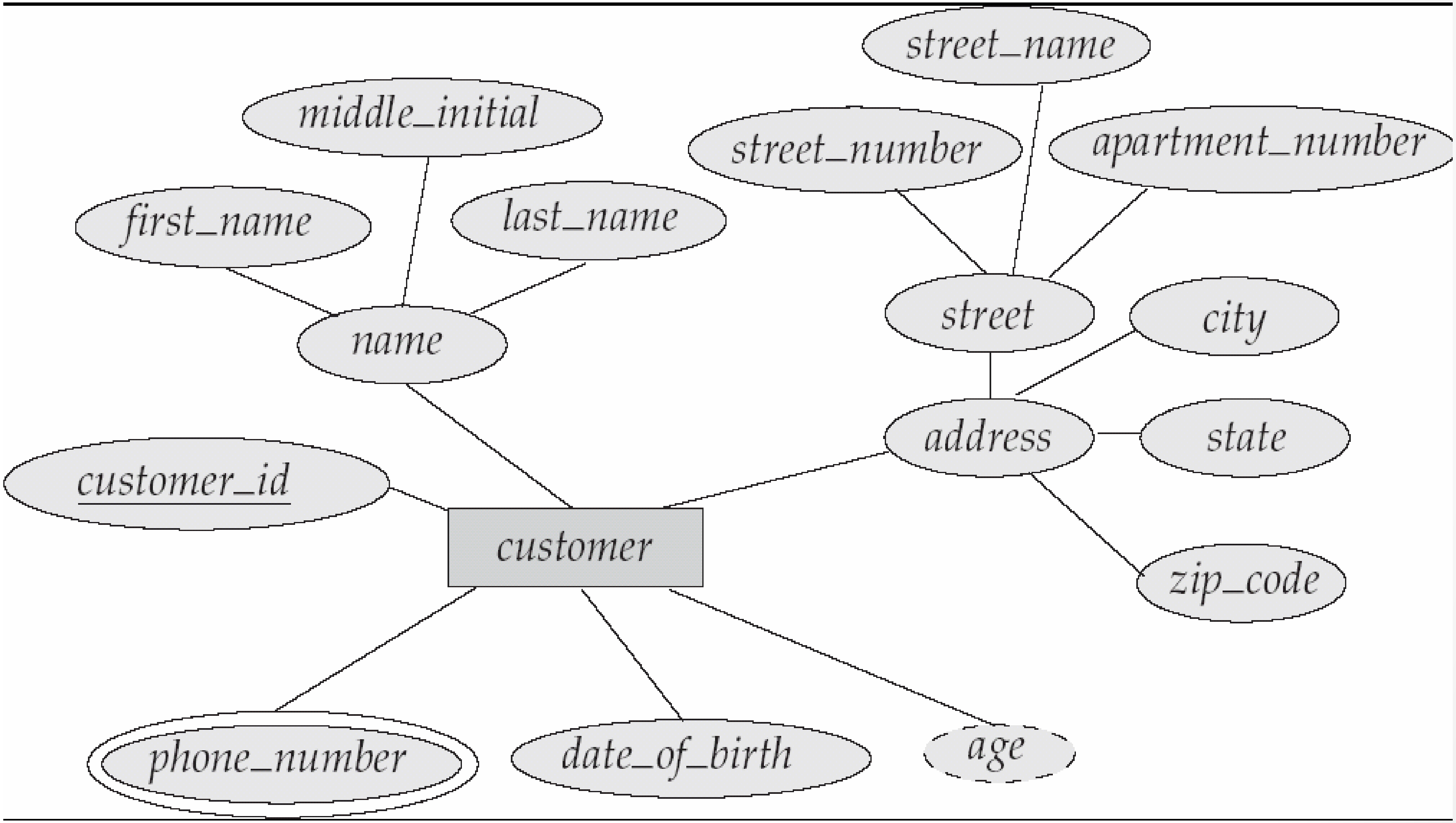
- An attribute that is a combination of other attributes is known as composite attribute. For example, In student entity, the student address is a composite attribute as an address is composed of other attributes such as pin code, state, country.



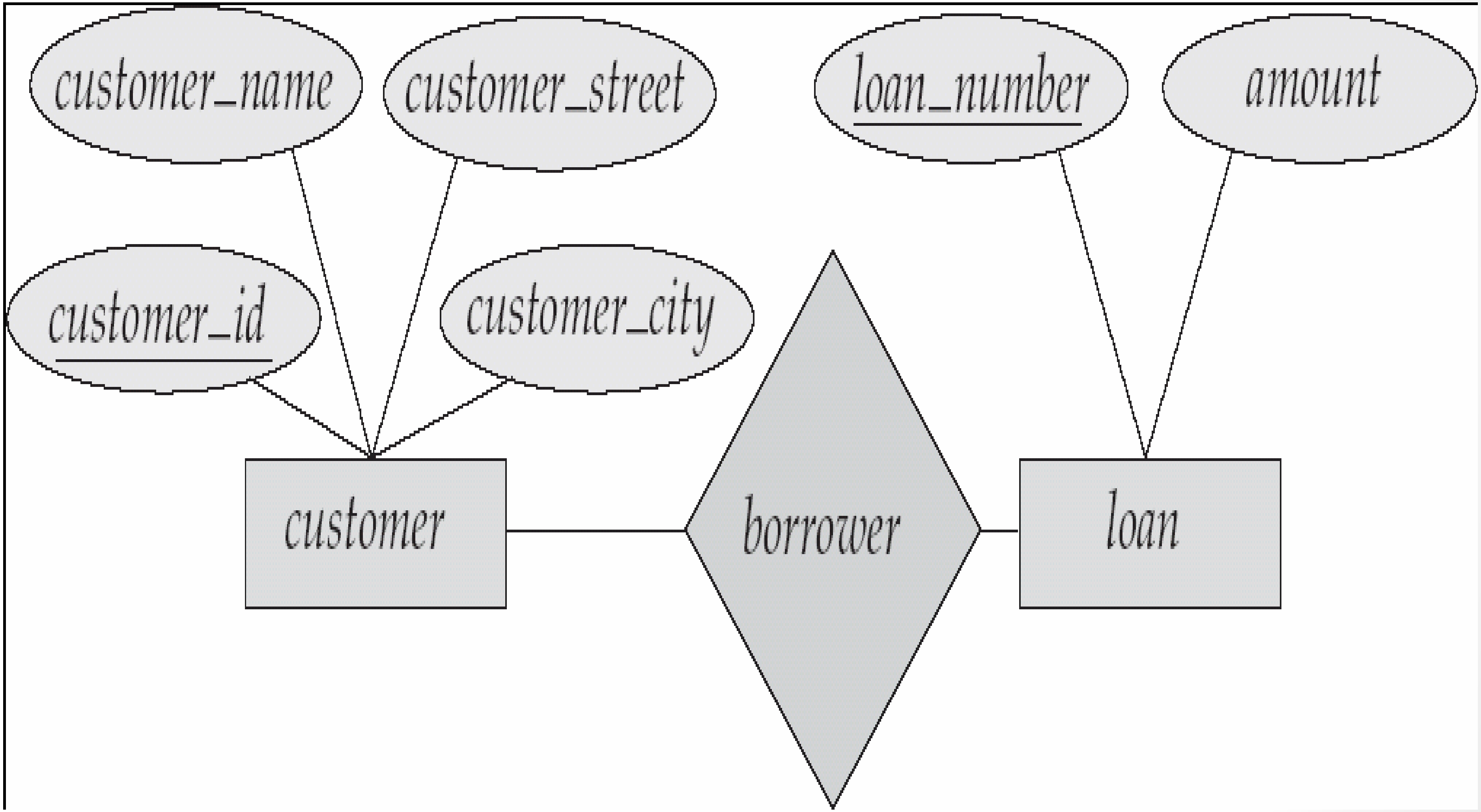
Multivalued and Derived Attribute

- An attribute that can hold multiple values is known as multivalued attribute. It is represented with **double ovals** in an ER Diagram. For example – A person can have more than one phone number.
- A derived attribute is one whose value is dynamic and derived from another attribute. It is represented by **dashed oval** in an ER Diagram. For example – Person age is a derived attribute as it changes over time and can be derived from another attribute (Date of birth).so the phone number attribute is multivalued.


Stu_Phone
Stu_Name
age
date_of_birth
Stu_Id
Student
Beginnersbook.com


street_name
middle_initial
street_number
apartment_number
first_name
last_name
street
city
name
address
state
customer_id
customer
zip_code
phone_number
date_of_birth
age

Example

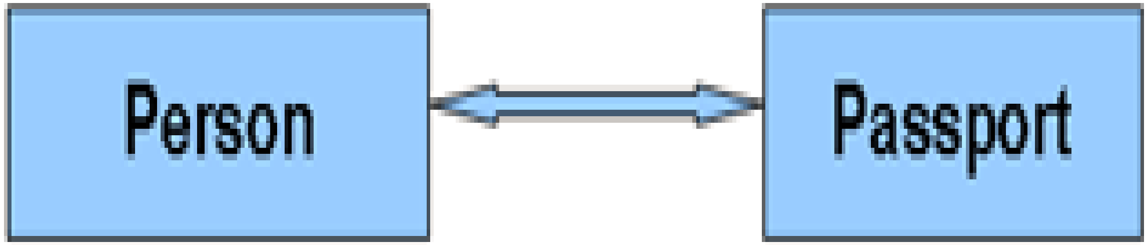

customer_name
customer_street
loan_number
amount
customer_id
customer_city
customer
borrower
loan

Relationship

- A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities. There are four types of relationships:
  1. One to One
  2. One to Many
  3. Many to One
  4. Many to Many

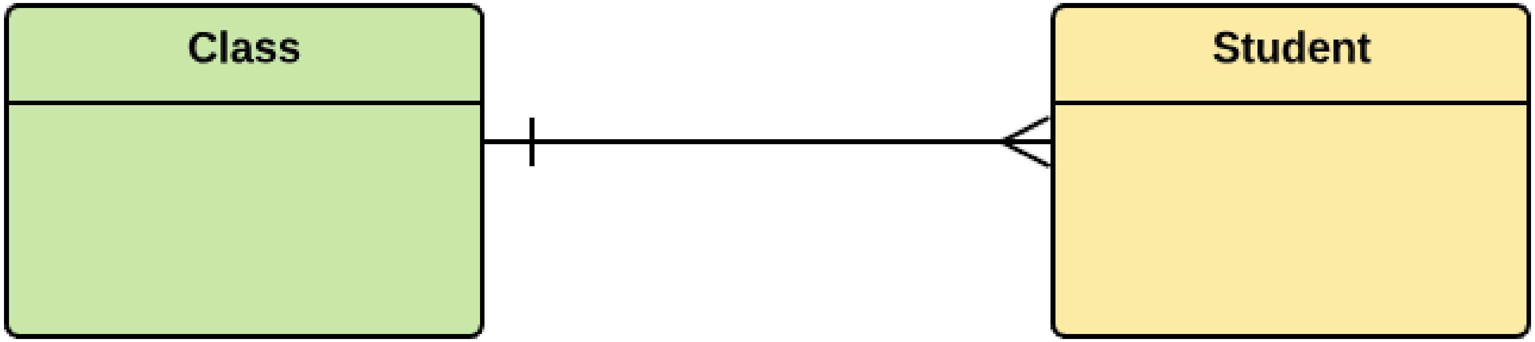
One to One Relationship

- When a single instance of an entity is associated with a single instance of another entity then it is called one to one relationship. For example, a person has only one passport and a passport is given to one person.



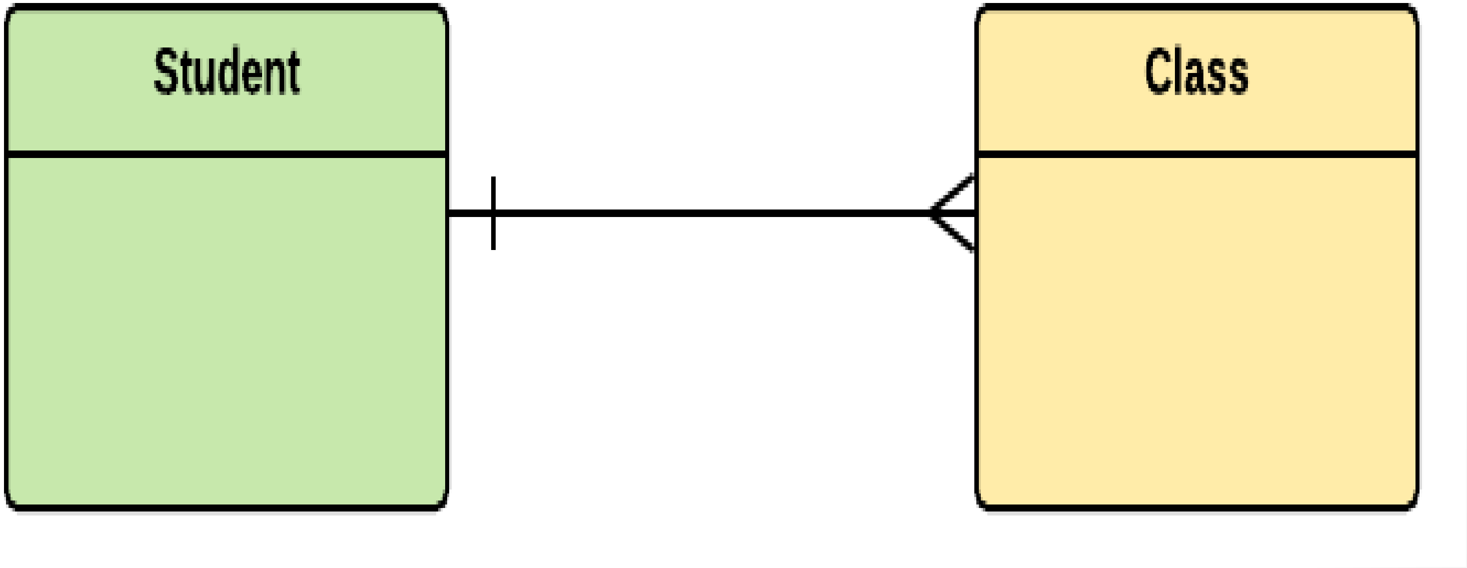
One to Many Relationship

- One entity from entity set X can be associated with multiple entities of entity set Y, but an entity from entity set Y can be associated with at least one entity.
- For example, one class is consisting of multiple students.



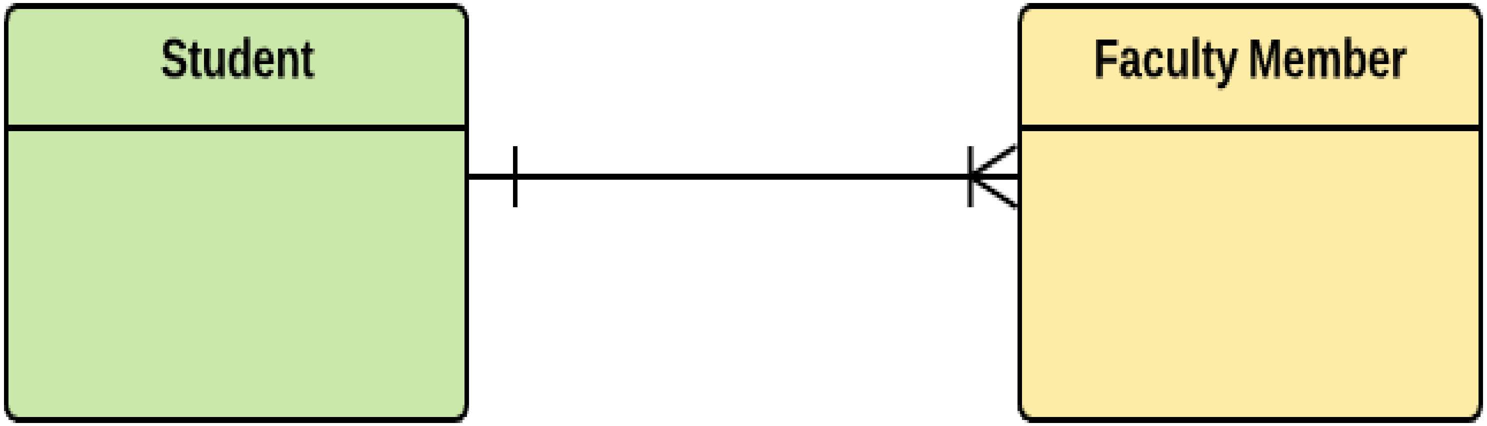
Many to One Relationship

- More than one entity from entity set X can be associated with at most one entity of entity set Y. However, an entity from entity set Y may or may not be associated with more than one entity from entity set X.
- For example, many students belong to the same class.

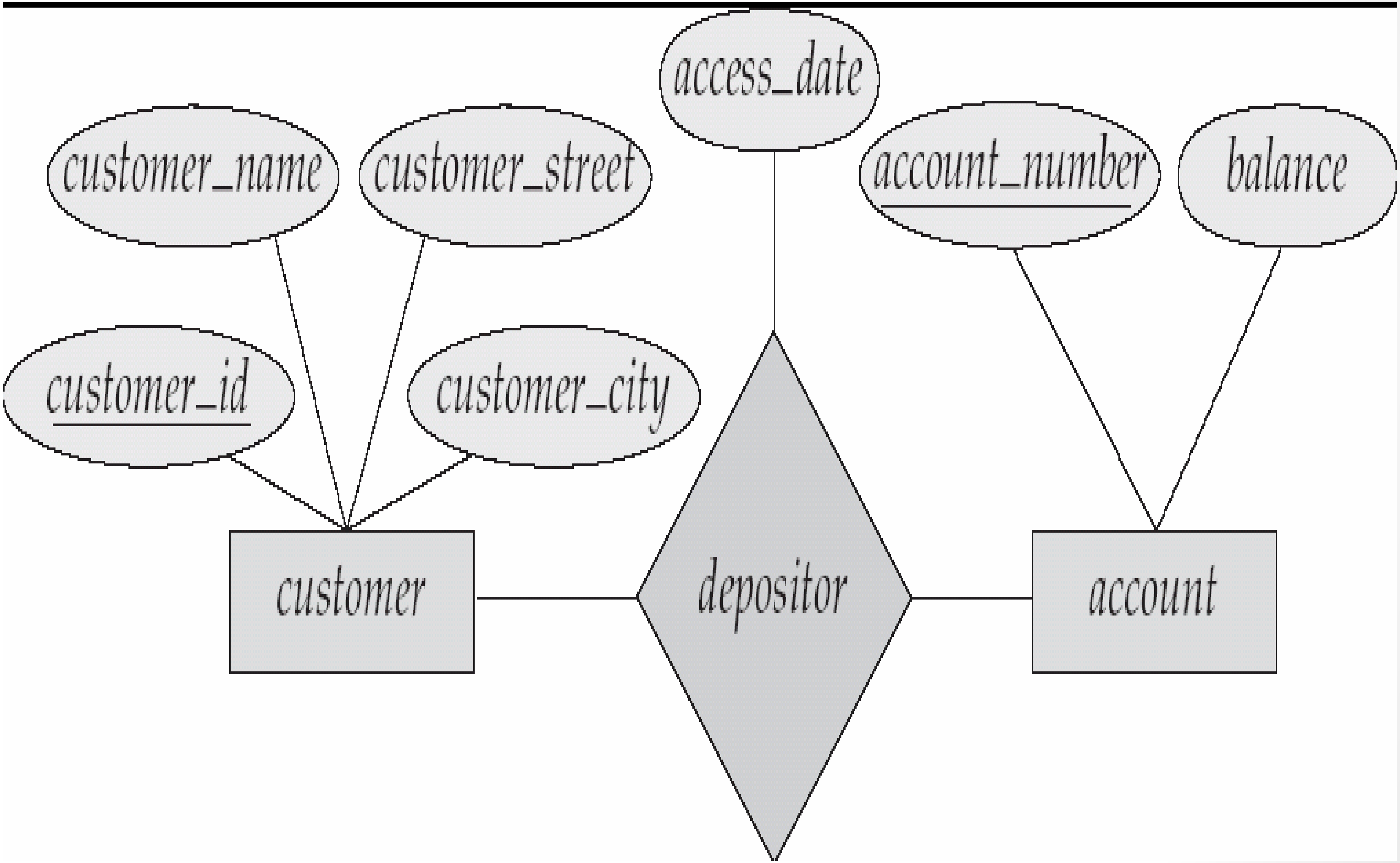


Many to Many Relationship

- One entity from X can be associated with more than one entity from Y and vice versa.
- For example, Students as a group are associated with multiple faculty members, and faculty members can be associated with multiple students.



Relationship Sets with Attributes

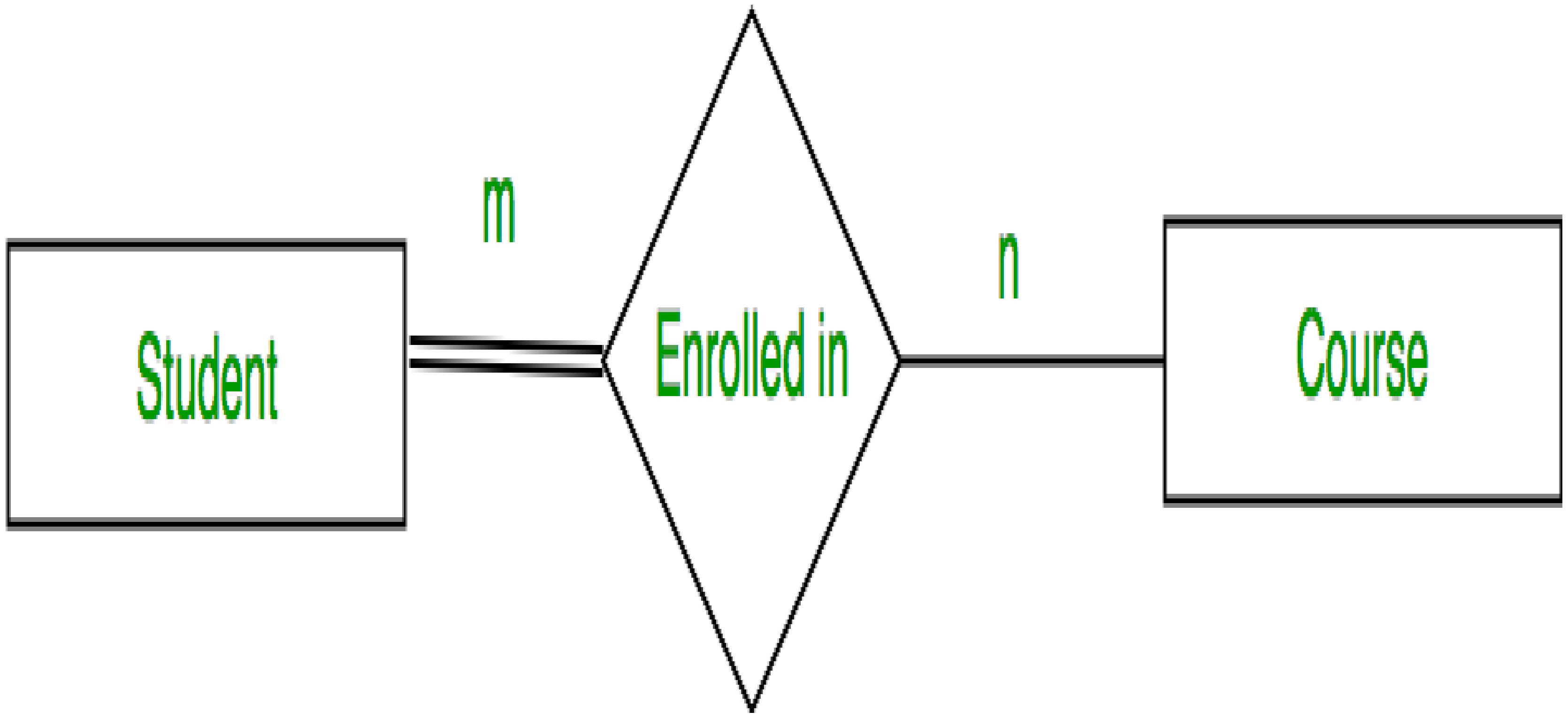


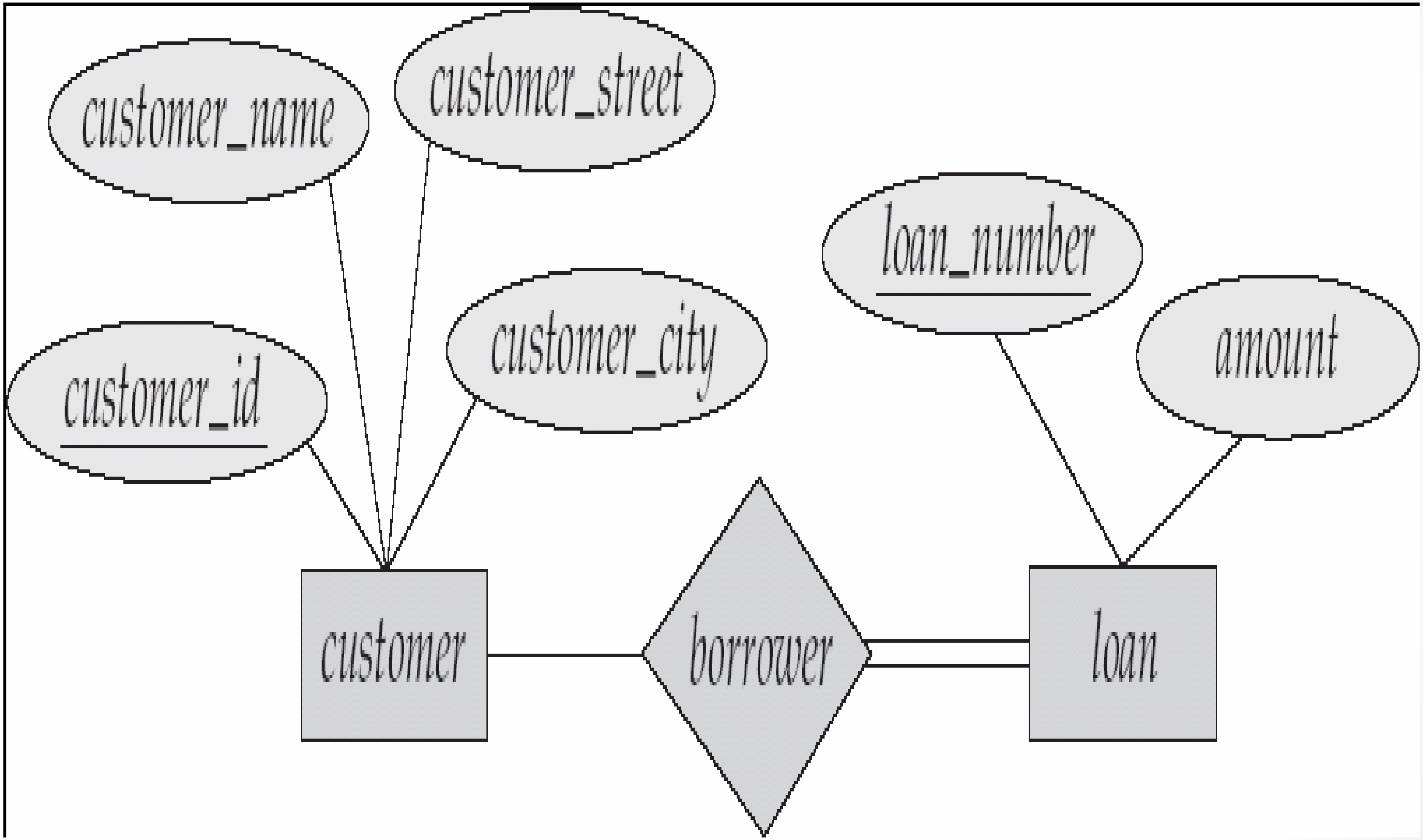
Participation of an Entity Set in a Relationship Set

Participation Constraint is applied on the entity participating in the relationship set.

1. **Total Participation** – Each entity in the entity set **must participate** in the relationship. If each student must enroll in a course, the participation of student will be total. Total participation is shown by double line in ER diagram.

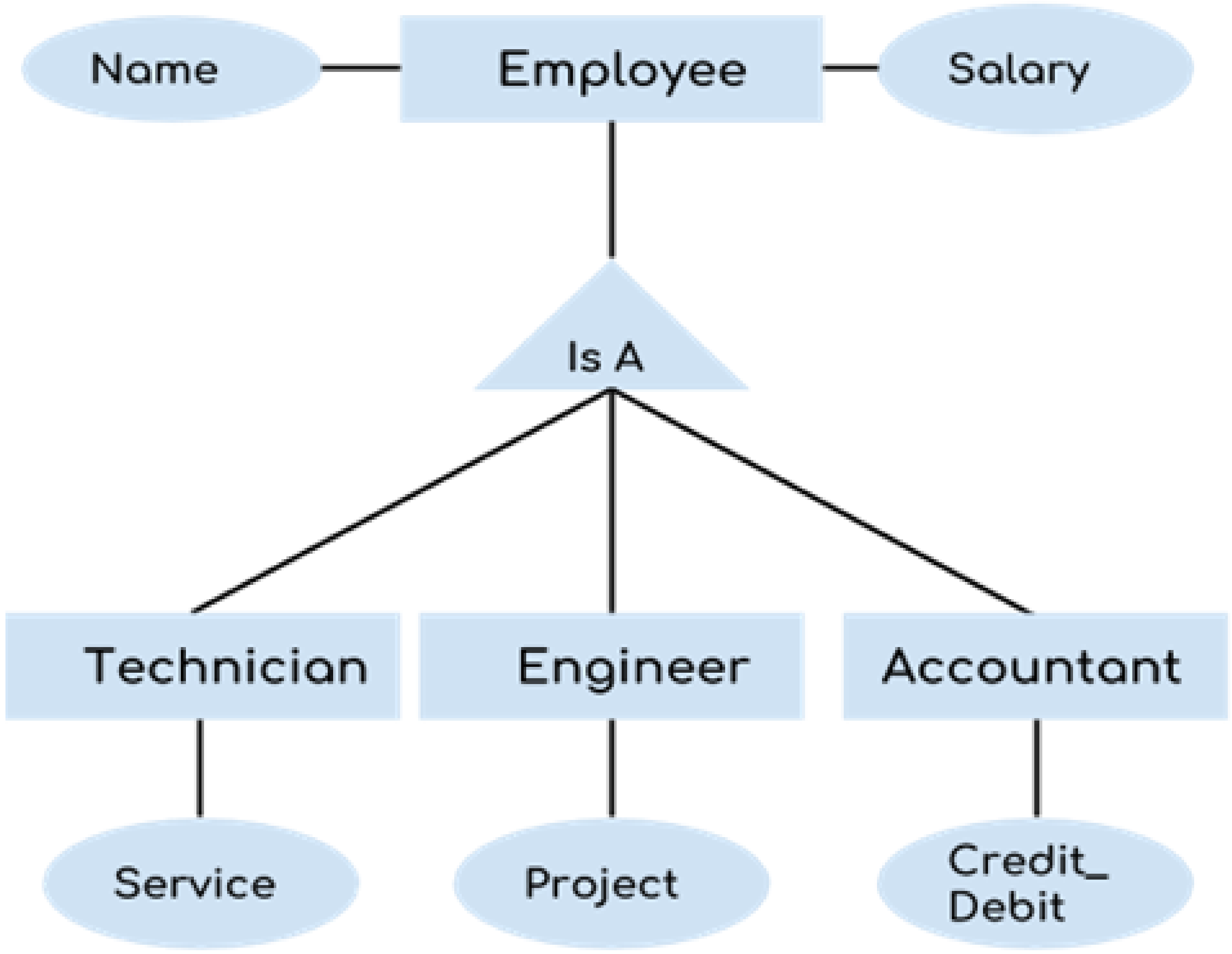
2. **Partial Participation** – The entity in the entity set **may or may NOT participate** in the relationship. If some courses are not enrolled by any of the student, the participation of course will be partial. The diagram depicts the 'Enrolled in' relationship set with Student Entity set having total participation and Course Entity set having partial participation.


Student
m
Enrolled in
n
Course


customer_name
customer_street
customer_id
customer_city
loan_number
amount
customer
borrower
loan

Specialization

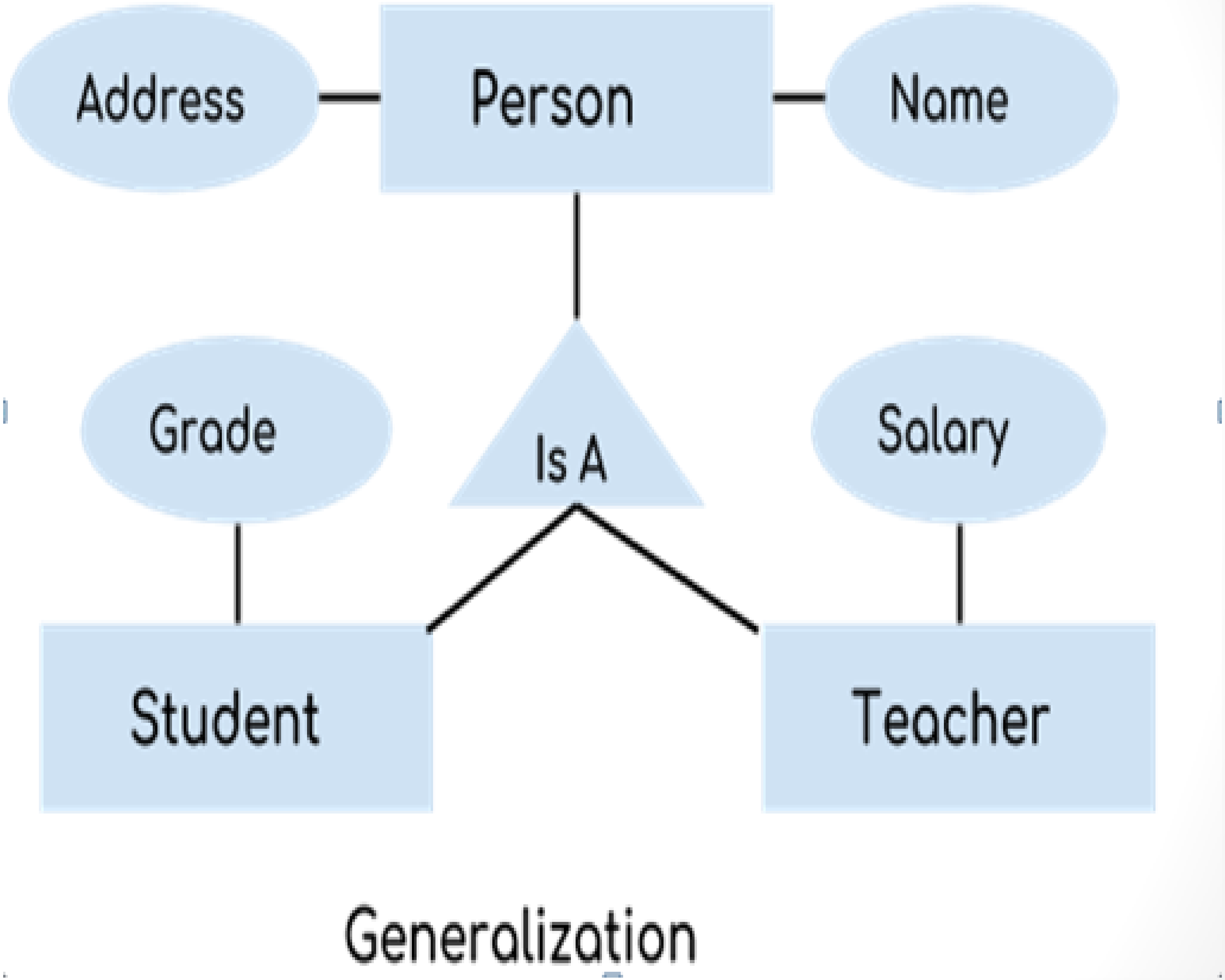
- **Specialization** is a top-down approach
- In this a higher-level entity is divided into multiple *specialized* lower-level entities.
- In addition to sharing the attributes of the higher-level entity, these lower-level entities have *specific* attributes of their own.
- Specialization is usually used to find subsets of an entity that has a few different or additional attributes.



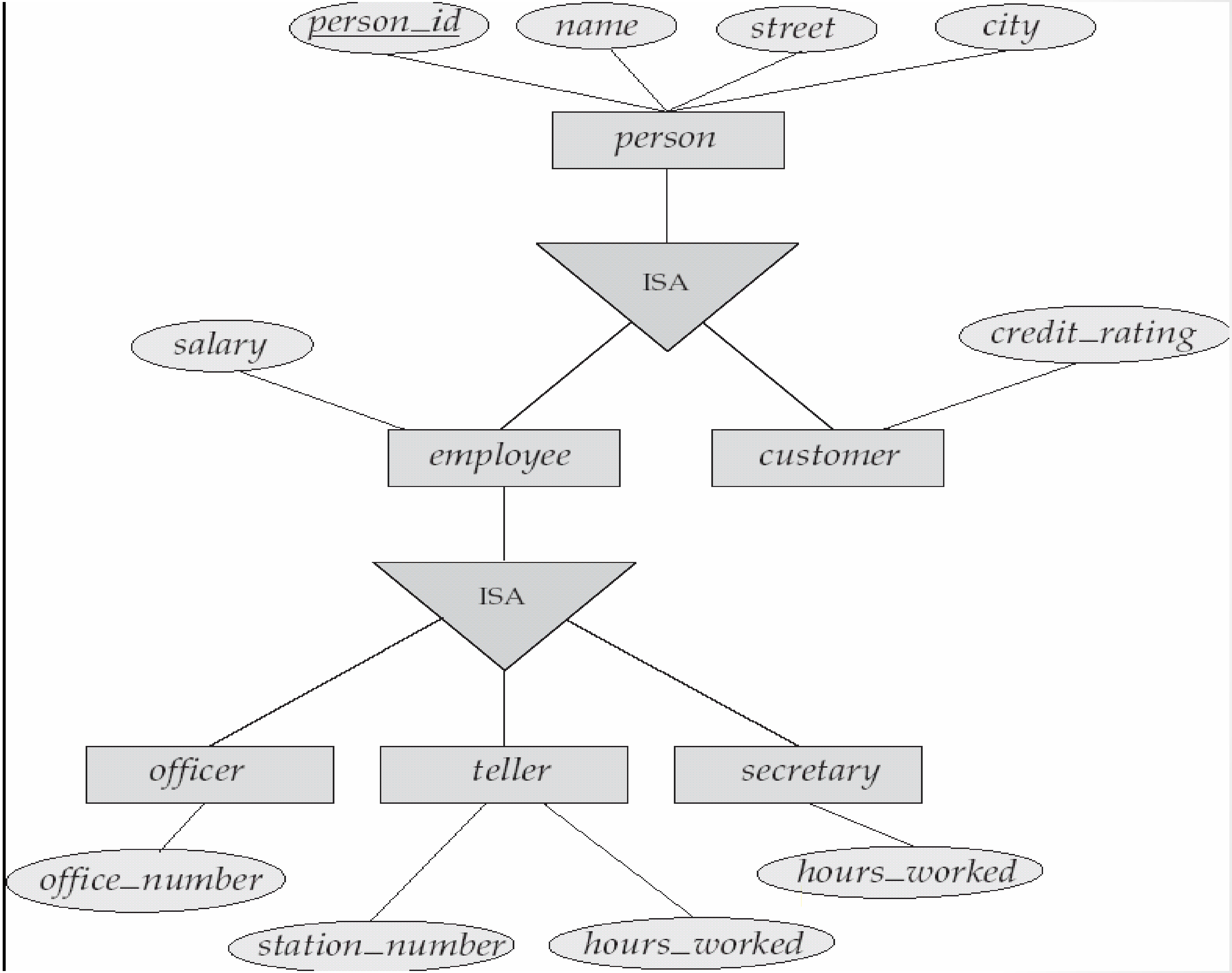
Specialization

Generalization

- Generalization is a bottom-up approach
- In this multiple lower-level entities are combined to form a single higher-level entity.
- Generalization is usually used to find common attributes among entities to form a generalized entity.
- It can also be thought of as the opposite of specialization.
- Specialization and generalization are simple inversions of each other; they are represented in an E-R diagram in the same way.

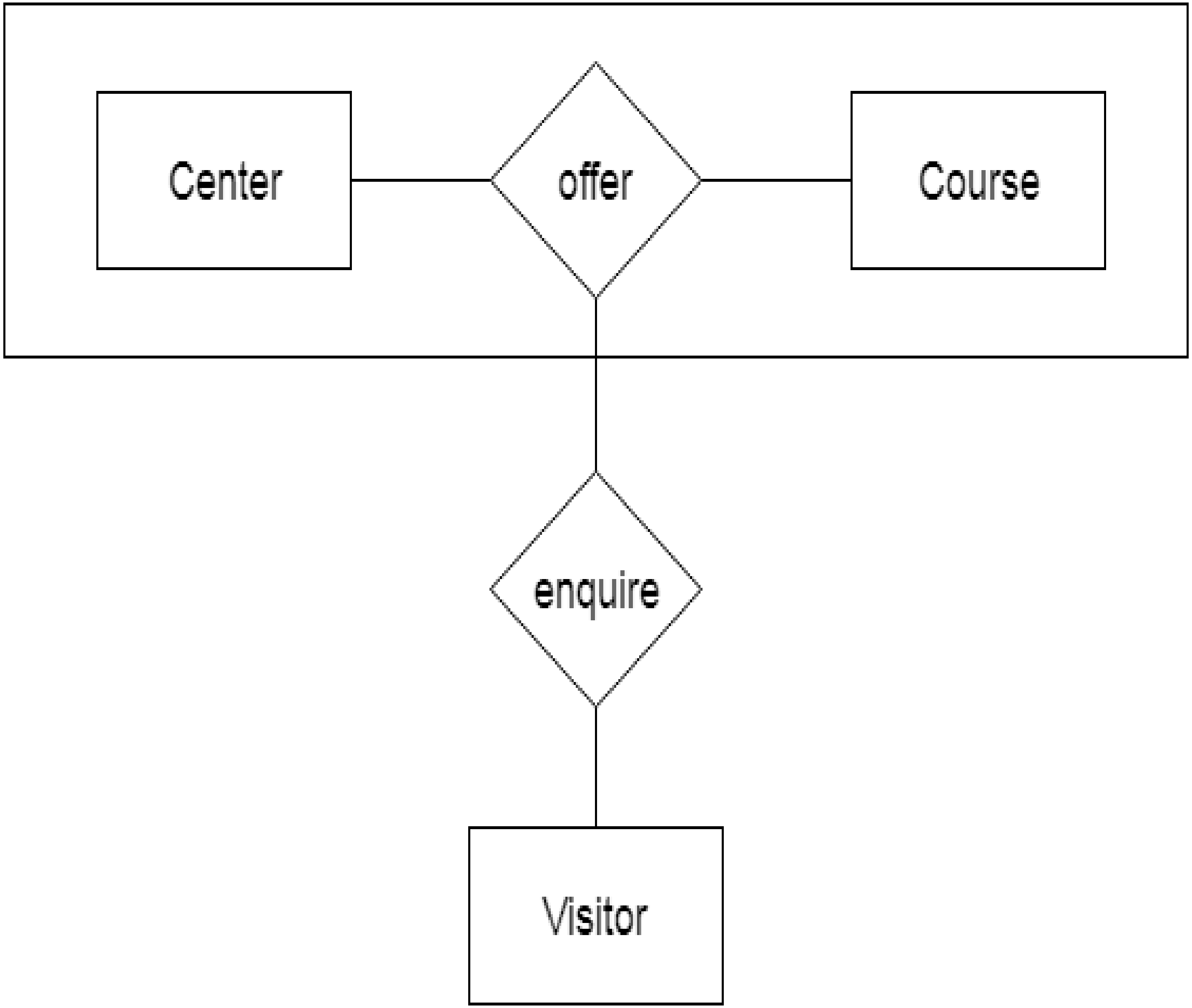


Generalization

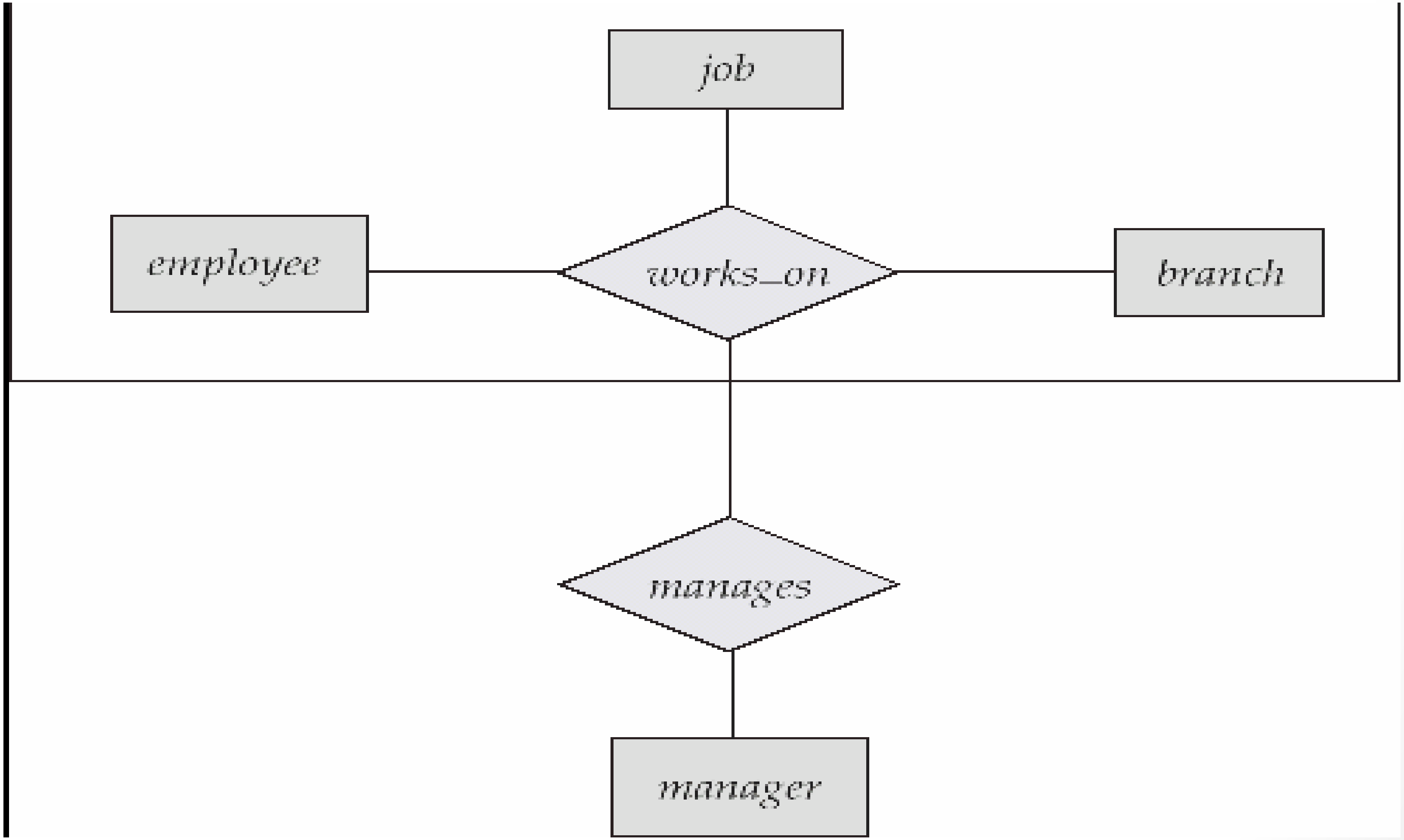

person_id
name
street
city
person
ISA
salary
credit_rating
employee
customer
ISA
officer
teller
secretary
office_number
station_number
hours_worked
hours_worked

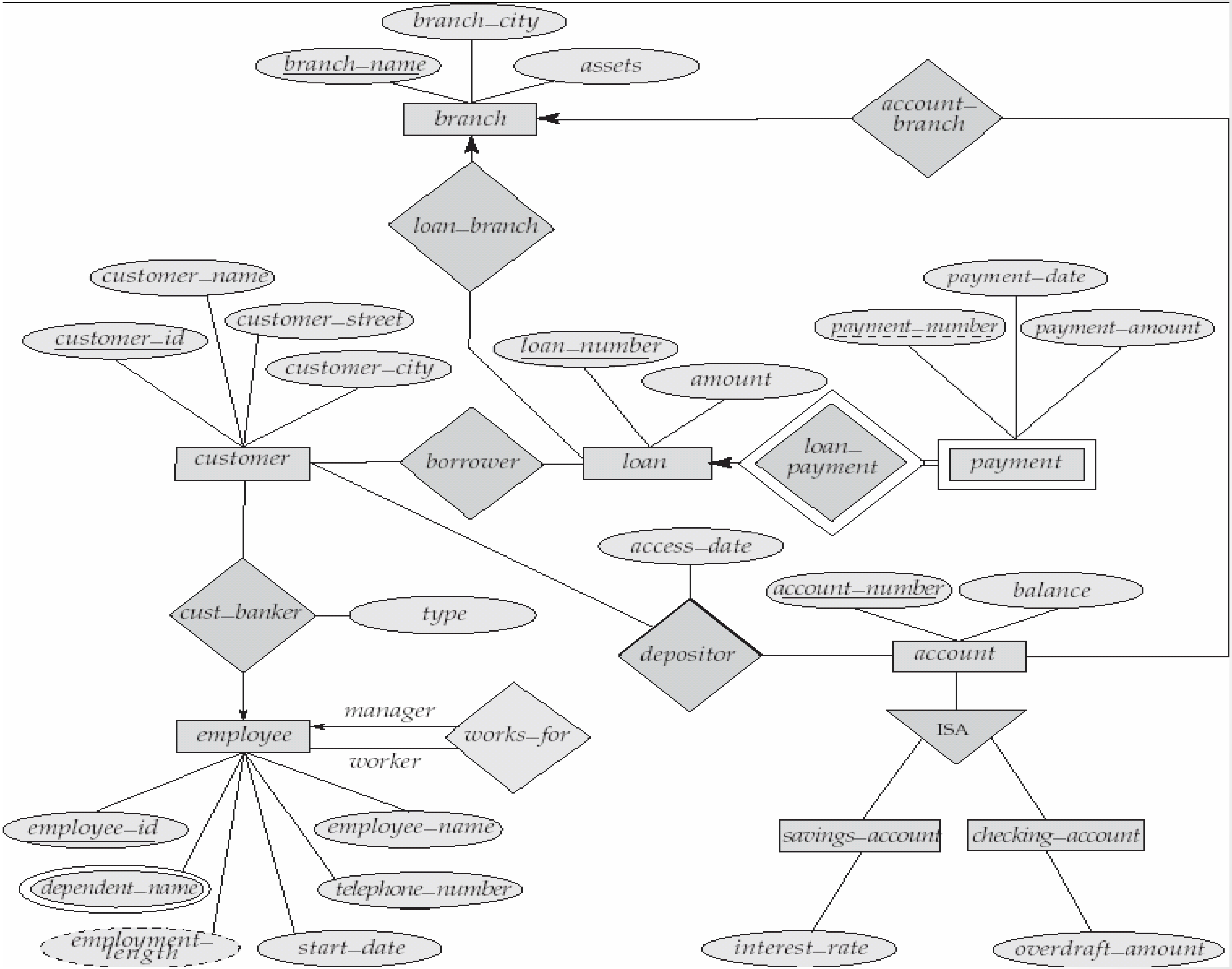
Aggregation

- **Aggregation** is a process in which a single entity alone is not able to make sense in a relationship so the relationship of two entities acts as one entity.
- Aggregation in DBMS (Database Management System) is a process of combining two or more entities to form a more meaningful new entity.
- **For example:** Center entity offers the Course entity act as a single entity in the relationship which is in a relationship with another entity visitor. In the real world, if a visitor visits a coaching center then he will never enquiry about the Course only or just about the Center instead he will ask the enquiry about both.


Center
offer
Course
enquire
Visitor

- Suppose managers want to manage for tasks performed by an employee at a branch




branch_city
branch_name
assets
branch
account–branch
loan_branch
customer_name
customer_street
customer_id
customer_city
loan_number
amount
payment_date
payment_number
payment_amount
customer
borrower
loan
loan–payment
payment
access_date
account_number
balance
cust_banker
type
depositor
account
manager
employee
works_for
worker
ISA
employee_id
employee_name
dependent_name
telephone_number
employment–length
start_date
savings_account
checking_account
interest_rate
overdraft_amount

Advantages of ER Model

1. **Conceptually it is very simple:** ER model is very simple because if we know relationship between entities and attributes, then we can easily draw an ER diagram.
2. **Better visual representation:** ER model is a diagrammatic representation of any logical structure of database. By seeing ER diagram, we can easily understand relationship among entities and relationship.
3. **Effective communication tool:** It is an effective communication tool for database designer.
4. **Highly integrated with relational model:** ER model can be easily converted into relational model by simply converting ER model into tables.
5. **Easy conversion to any data model:** ER model can be easily converted into another data model like hierarchical data model, network data model and so on.

Disadvantages of ER Model

1. **Limited constraints and specification**
2. **Loss of information content:** Some information be lost or hidden in ER model
3. **Limited relationship representation:** ER model represents limited relationship as compared to another data models like relational model etc.
4. **No representation of data manipulation:** It is difficult to show data manipulation in ER model.
5. **Popular for high level design**: ER model is very popular for designing high level design
6. **No industry standard for notation**

Relational Model

- **RELATIONAL MODEL (RM)** represents the database as a collection of relations.
- Proposed by E. F. Codd in 1970.
- A relation is nothing but a table of values.
- Every row in the table represents a collection of related data values. These rows in the table denote a real-world entity or relationship.
- The table name and column names are helpful to interpret the meaning of values in each row. The data are represented as a set of relations. However, the physical storage of the data is independent of the way the data are logically organized.
- Some popular Relational Database management systems are:

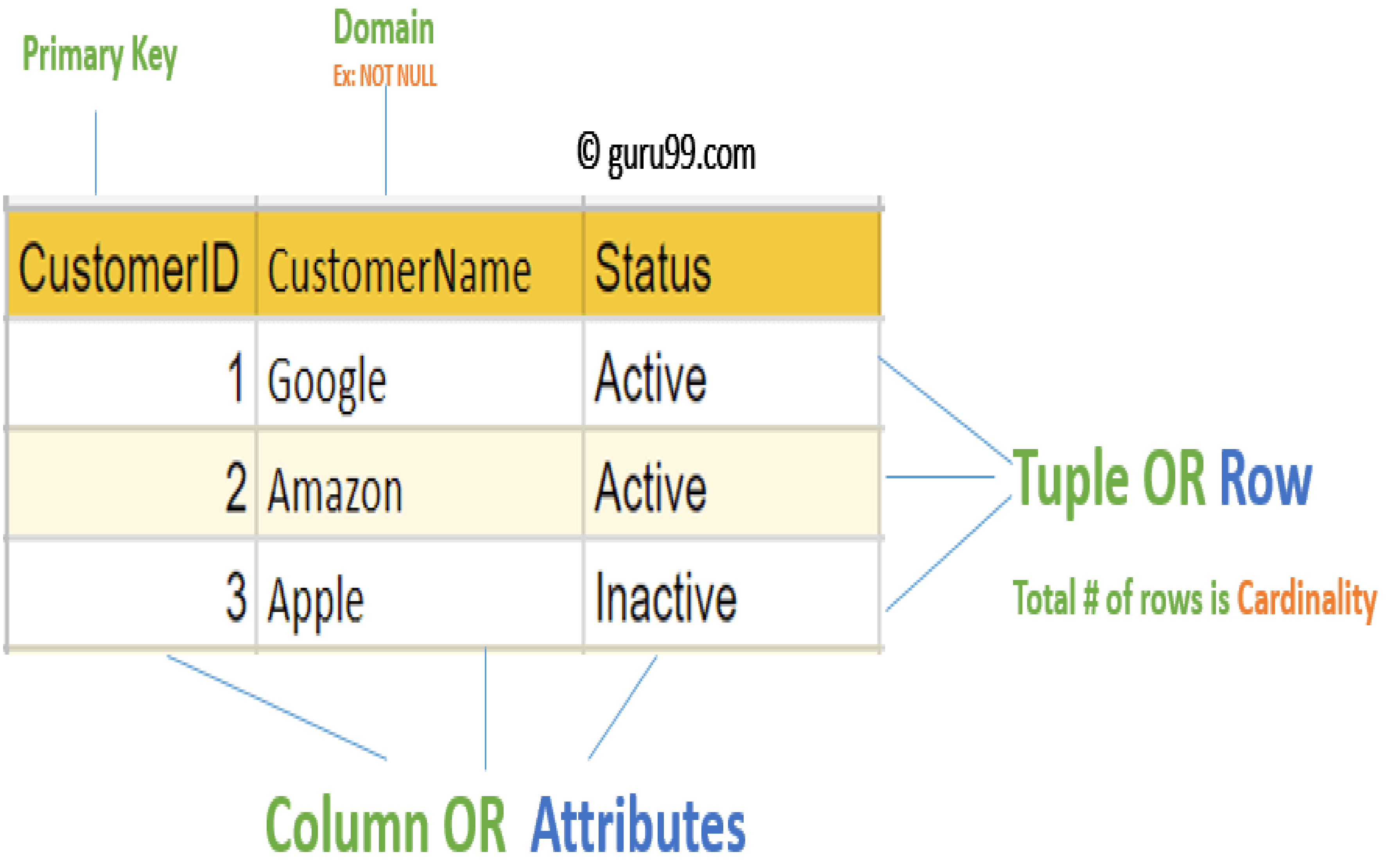
1. DB2 and Informix Dynamic Server - IBM
2. Oracle and RDB – Oracle
3. SQL Server and Access - Microsoft

Relational Model Concepts

- **Attribute:** Each column in a Table. Attributes are the properties which define a relation. e.g., Student_Rollno, NAME,etc.
- **Tables** – In the Relational model the, relations are saved in the table format. It is stored along with its entities. A table has two properties rows and columns. Rows represent records and columns represent attributes.
- **Tuple** – It is nothing but a single row of a table, which contains a single record.
- **Relation Schema:** A relation schema represents the name of the relation with its attributes.
- **Degree:** The total number of attributes which in the relation is called the degree of the relation.

- **Cardinality:** Total number of rows present in the Table.
- **Relation key** - Every row has one, two or multiple attributes, which is called relation key.
- **Attribute domain** – Every attribute has some pre-defined value and scope which is known as attribute domain

Table also called Relation



Primary Key

Domain

Ex: NOT NULL

© guru99.com

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

Tuple OR Row

Total # of rows is Cardinality

Column OR Attributes

Total # of column is Degree

Operations in Relational Model

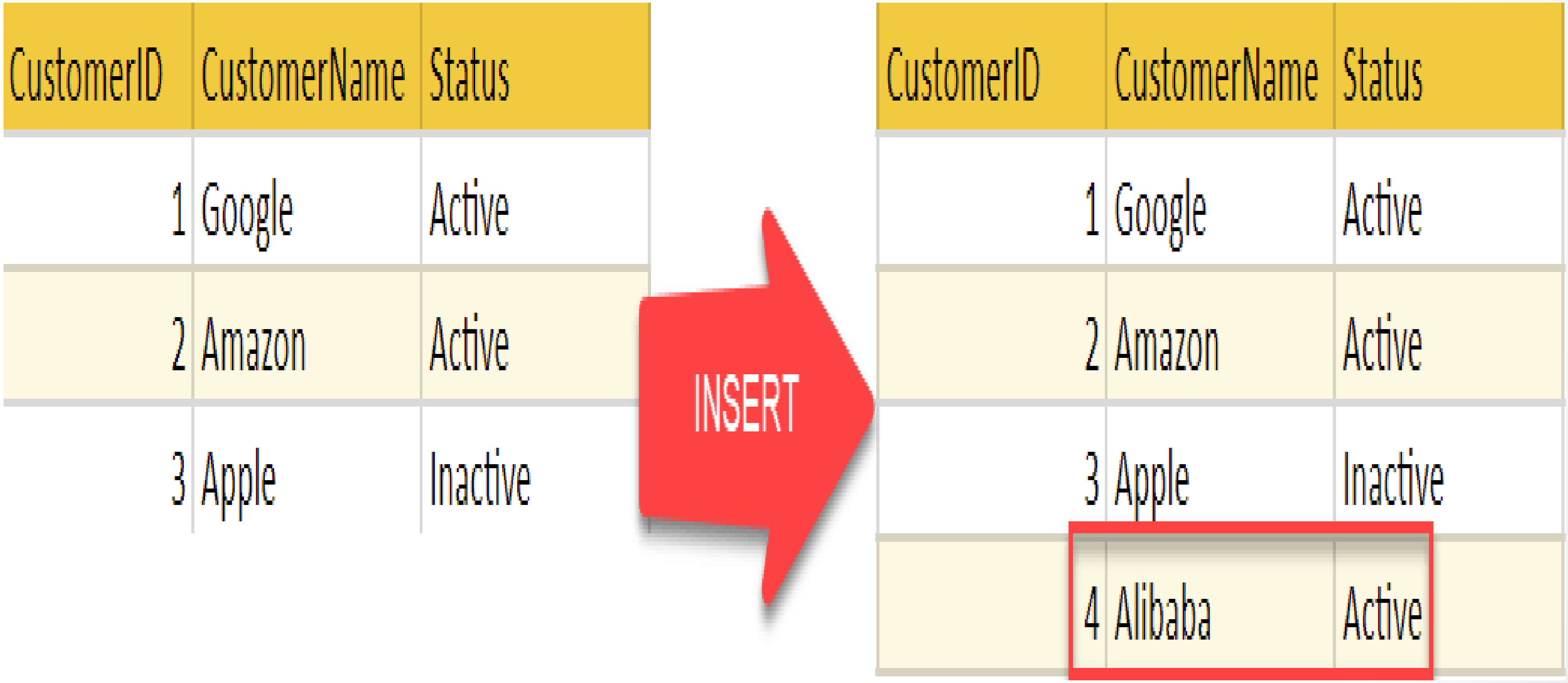
- Four basic update operations performed on relational database model are
  - Insert, update, delete and select.

1. Insert is used to insert data into the relation
2. Delete is used to delete tuples from the table.
3. Modify allows you to change the values of some attributes in existing tuples.
4. Select allows you to choose a specific range of data.

- Whenever one of these operations are applied, integrity constraints specified on the relational database schema must never be violated.

Insert Operation

- The insert operation gives values of the attribute for a new tuple which should be inserted into a relation.

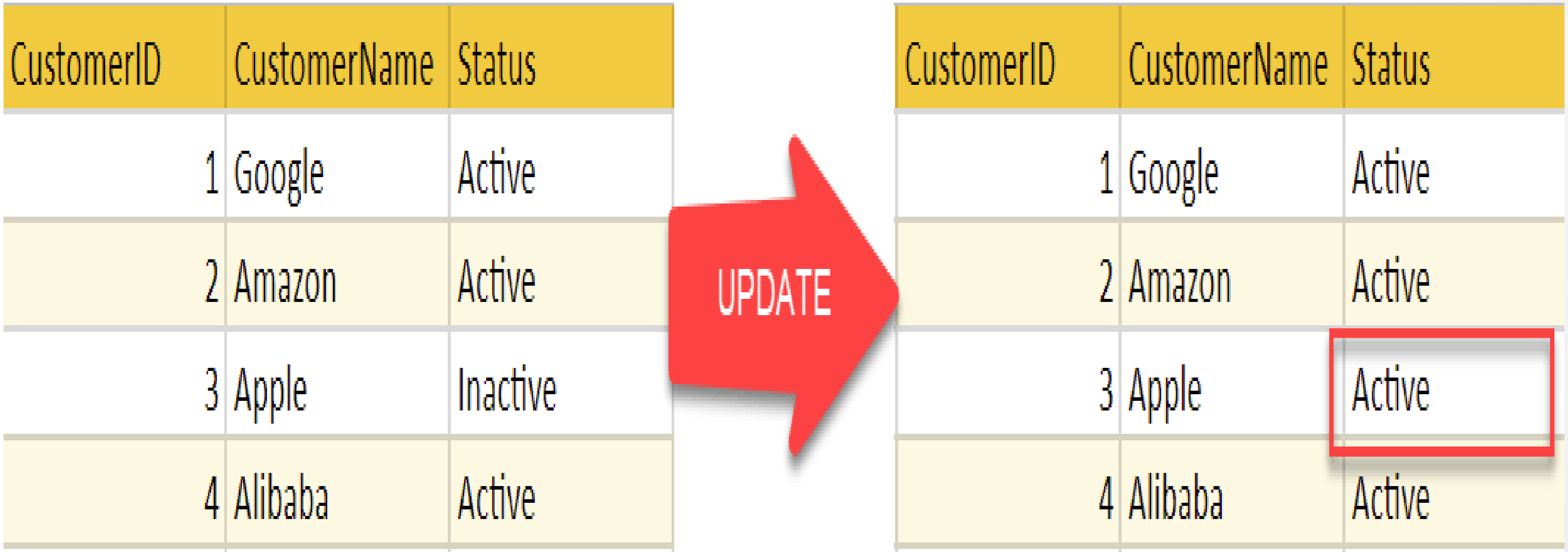


CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

Update Operation

- You can see that in the below-given relation table CustomerName= 'Apple' is updated from Inactive to Active.



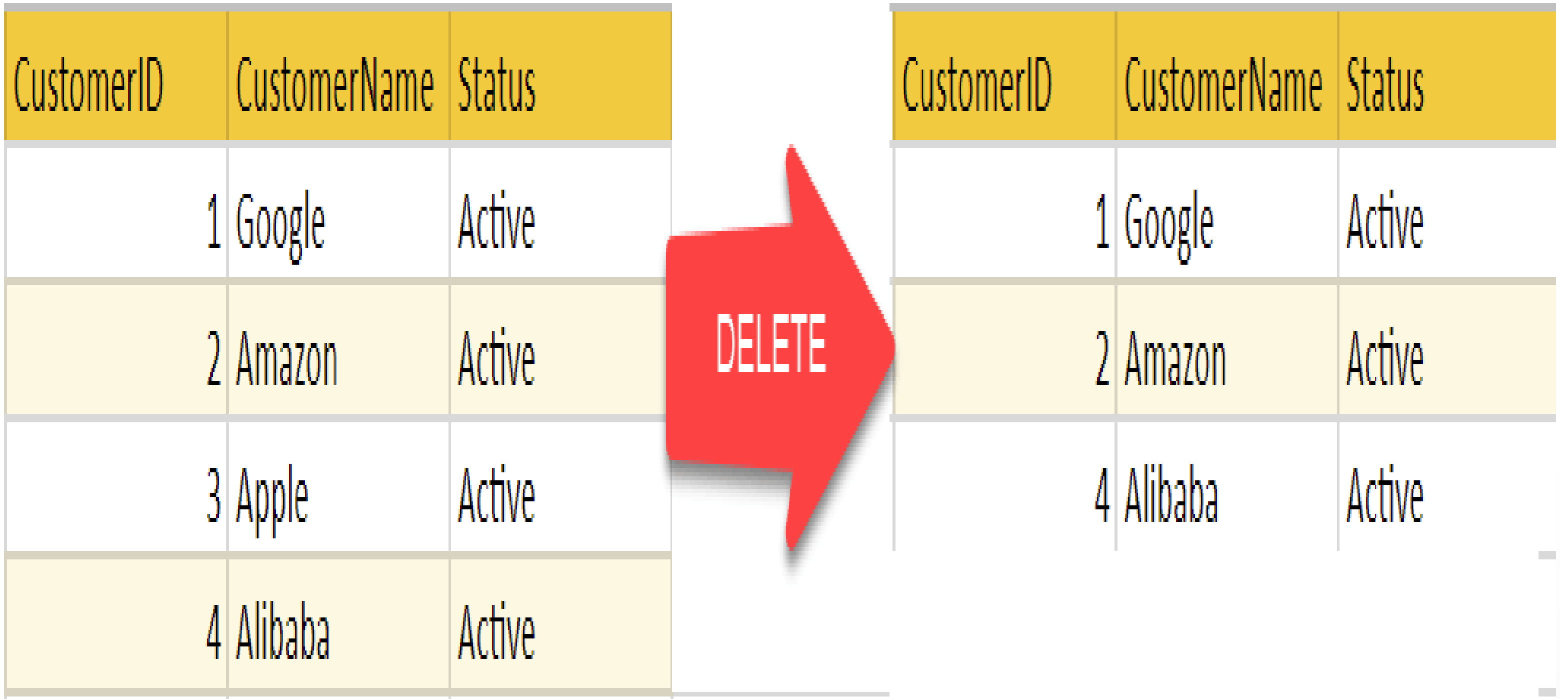
CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Inactive
4	Alibaba	Active

UPDATE

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

Delete Operation

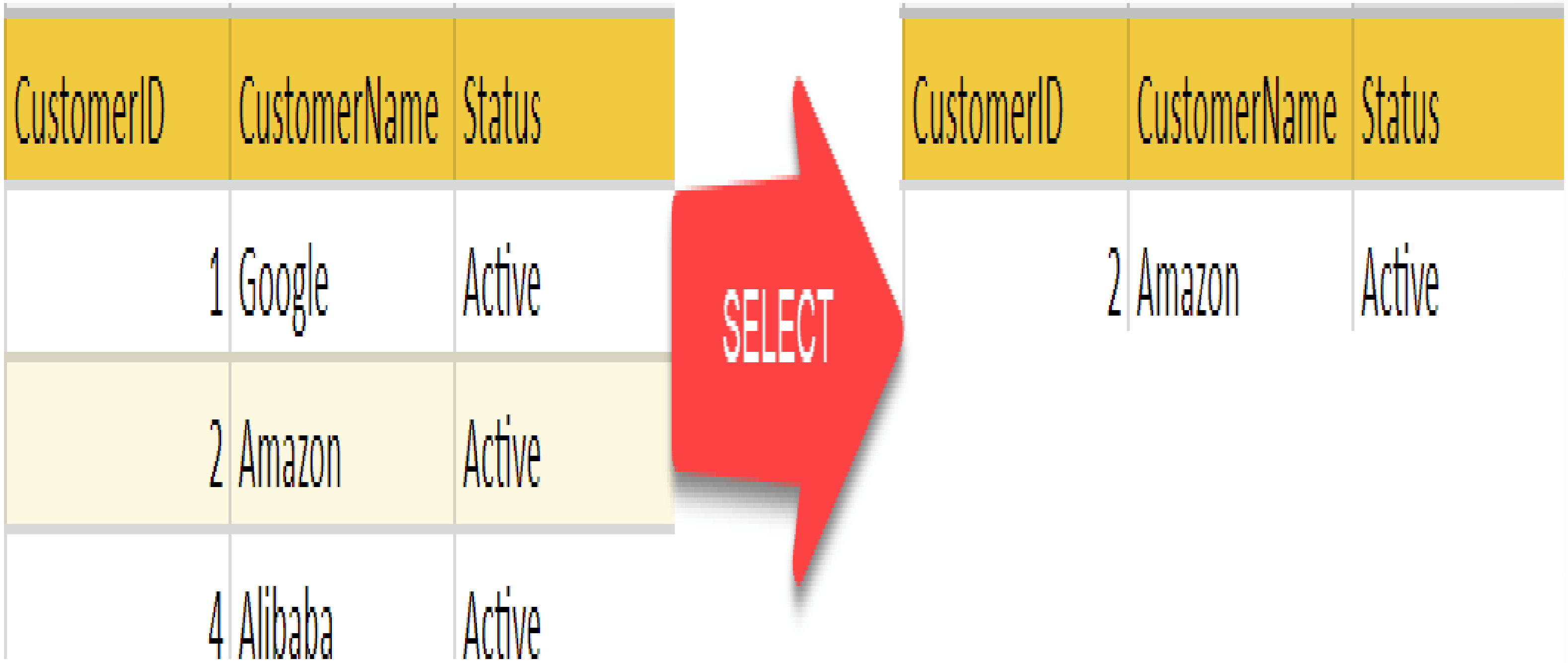
- To specify deletion, a condition on the attributes of the relation selects the tuple to be deleted.
-



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
3	Apple	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

Select Operation



CustomerID	CustomerName	Status
1	Google	Active
2	Amazon	Active
4	Alibaba	Active

CustomerID	CustomerName	Status
2	Amazon	Active

Best Practices for creating a Relational Model

1. Data need to be represented as a collection of relations
2. Each relation should be depicted clearly in the table
3. Rows should contain data about instances of an entity
4. Columns must contain data about attributes of the entity
5. Cells of the table should hold a single value
6. Each column should be given a unique name
7. No two rows can be identical
8. The values of an attribute should be from the same domain

Advantages of Relational Model

1. **Simplicity**: A relational data model is simpler than the hierarchical and network model.
2. **Structural Independence**: The relational database is only concerned with data and not with a structure. This can improve the performance of the model.
3. **Easy to use**: The relational model is easy as tables consisting of rows and columns is quite natural and simple to understand
4. **Query capability**: It makes possible for a high-level query language like SQL to avoid complex database navigation.
5. **Data independence**: The structure of a database can be changed without having to change any application.
6. **Scalable**: Regarding a number of records, or rows, and the number of fields, a database should be enlarged to enhance its usability.

Disadvantages

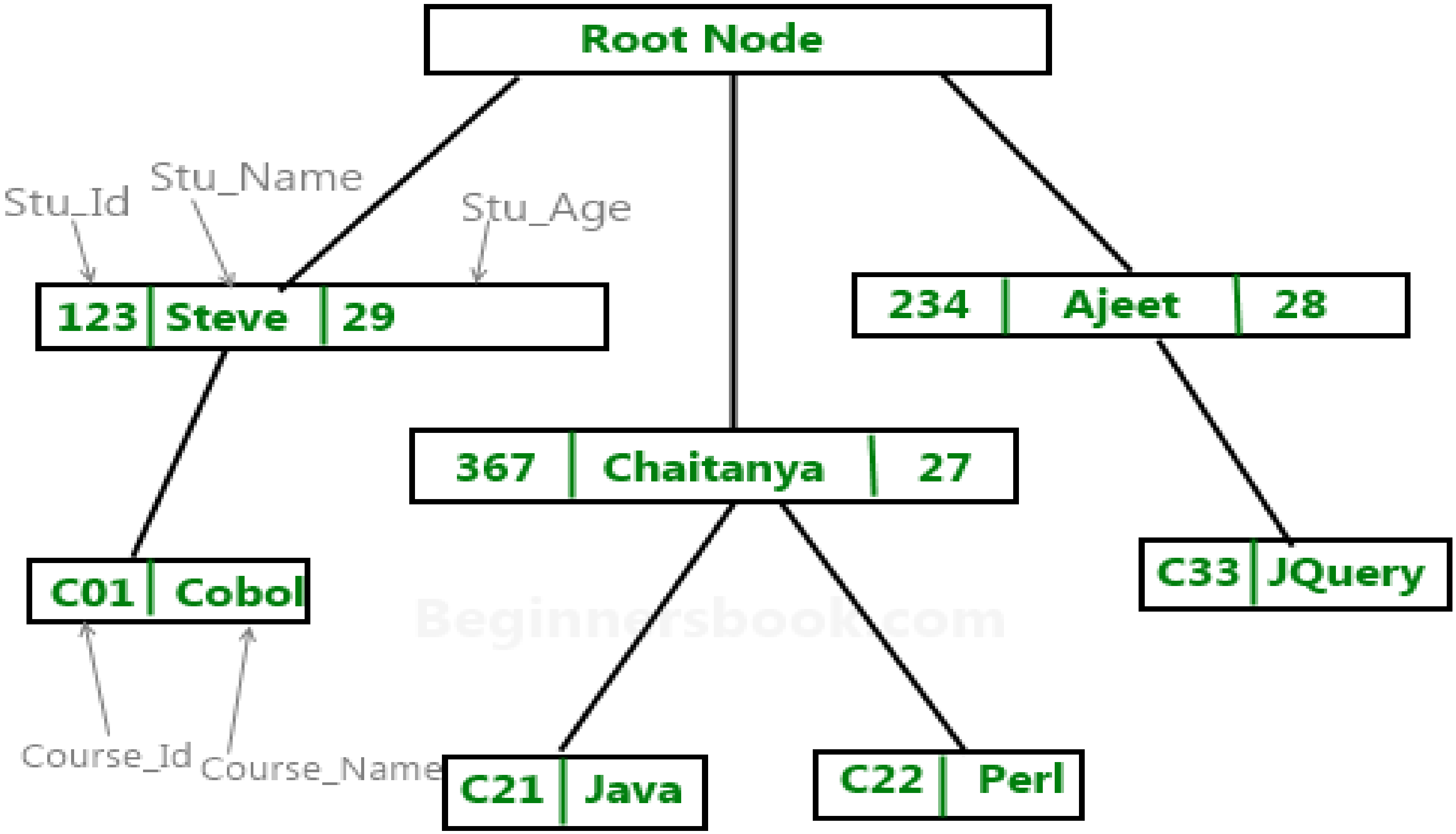
1. Few relational databases have limits on field lengths which can't be exceeded.
2. Relational databases can sometimes become complex as the amount of data grows, and the relations between pieces of data become more complicated.
3. Complex relational database systems may lead to isolated databases where the information cannot be shared from one system to another.

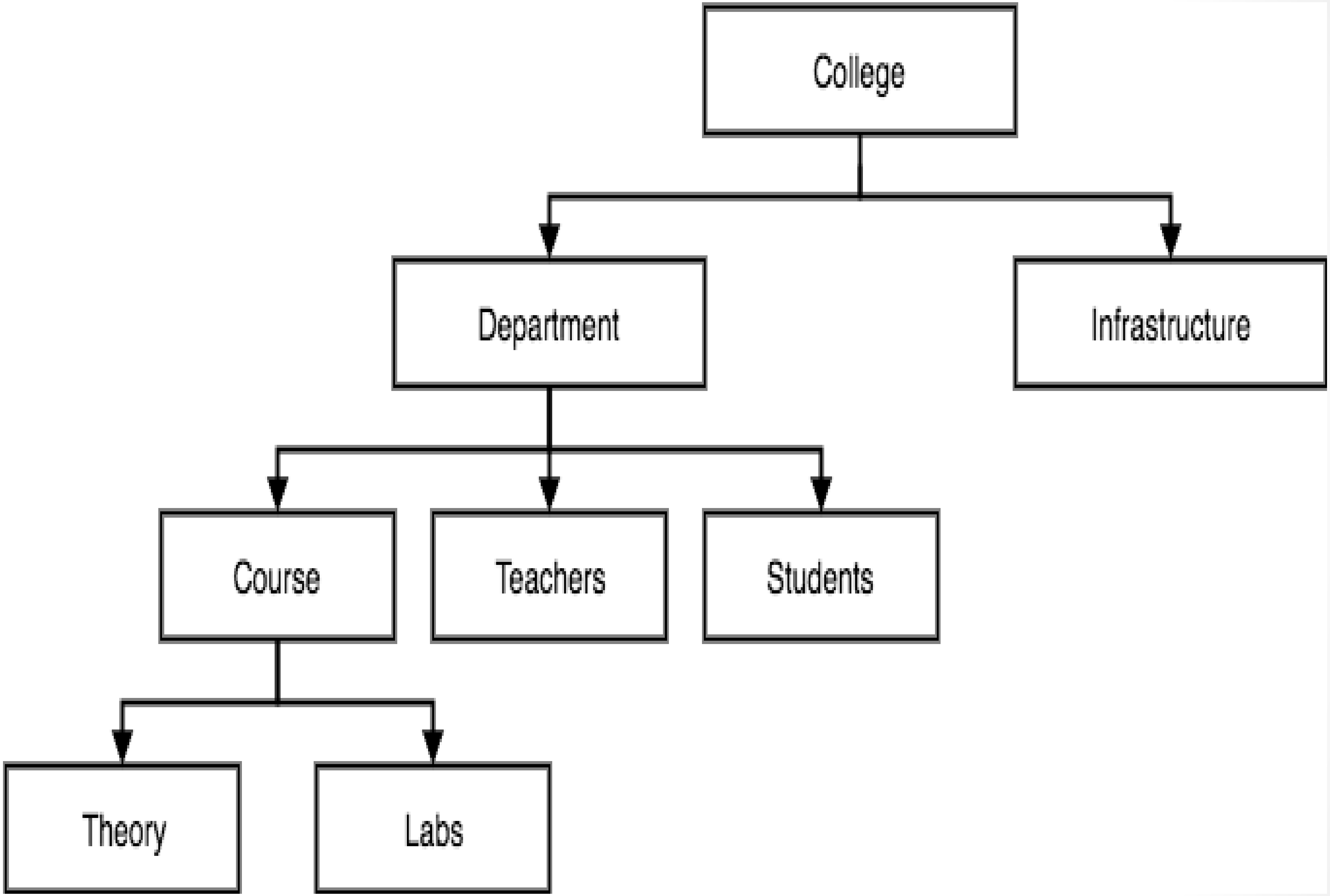
Summary

1. The Relational database model represents the database as a collection of relations (tables)
2. Attribute, Tables, Tuple, Relation Schema, Degree, Cardinality, Column, Relation instance, are some important components of Relational Model
3. Relational Integrity constraints are referred to conditions which must be present for a valid relation
4. Domain constraints can be violated if an attribute value is not appearing in the corresponding domain or it is not of the appropriate data type
5. Insert, Select, Modify and Delete are operations performed in Relational Model
6. The relational database is only concerned with data and not with a structure which can improve the performance of the model
7. Advantages of relational model is simplicity, structural independence, ease of use, query capability, data independence, scalability.
8. Few relational databases have limits on field lengths which can't be exceeded.

Hierarchical Model

- In **hierarchical model**, data is organized into a tree like structure with each record is having one parent record and many children. The main drawback of this model is that, it can have only one to many relationships between nodes.
- Lets say we have few students and few courses and a course can be assigned to a single student only, however a student take any number of courses so this relationship becomes one to many.


Root Node
Stu_Id
Stu_Name
Stu_Age
123 | Steve | 29
234 | Ajeet | 28
367 | Chaitanya | 27
C01 | Cobol
C33 | JQuery
Beginnersbook.com
Course_Id
Course_Name
C21 | Java
C22 | Perl


College
Department
Infrastructure
Course
Teachers
Students
Theory
Labs

Advantages and Disadvantages

- The key advantages of hierarchical databases are:

1. Traversing through a tree structure is very simple and fast due to its one-to-many relationships format. Major several programming languages provide functionality to read tree structure databases.
2. Easy to understand due to its one-to-many relationships.

- Key disadvantages of hierarchical databases are:

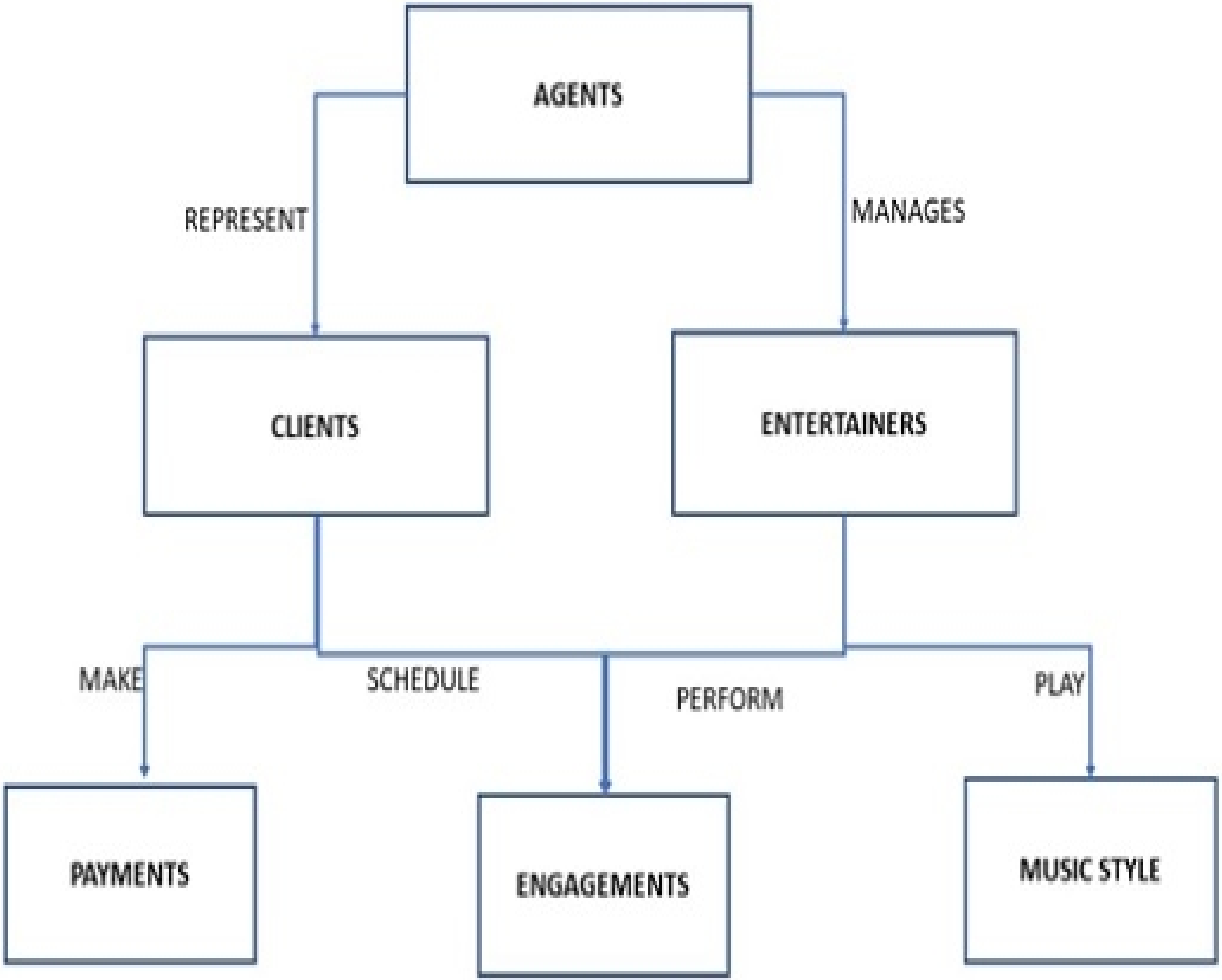
1. It’s rigid format of one-to-many relationships. That means, it doesn’t allow more than one parent of a child.
2. Multiple nodes with same parent will add redundant data.
3. Moving one record from one level to other level could be challenging.

Most Popular Hierarchical DBMS

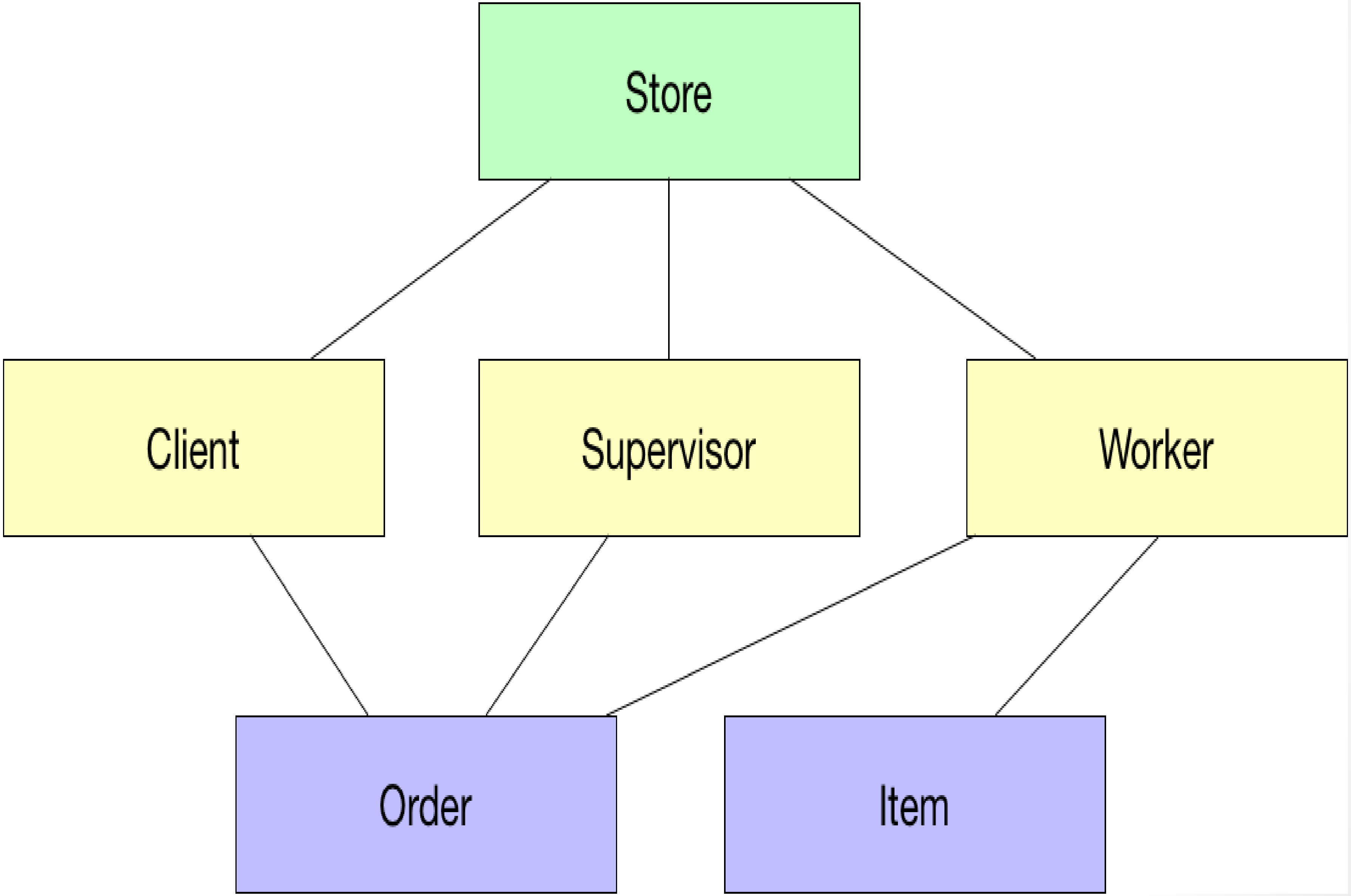
- The most popular hierarchical databases are IBM Information Management System (IMS) and RDM Mobile. Windows Registry is another example of a real-world use cases of a hierarchical database system.
- XML and XAML are two more popular and most widely use data storages that are based on hierarchical data model. In XML and XAML, each file starts with a root node that may be one or more child nodes. Each child node again can have one or more child nodes and so on.

Network Model

- Charles Bachman was the original inventor of the network model. In 1969, the Conference on Data Systems Languages (CODASYL) Consortium developed the network model into a standard specification
- This is an extension of the Hierarchical model. In this model data is organized more like a graph, and are allowed to have more than one parent node.
- In this database model data is more related as more relationships are established in this database model. Also, as the data is more related, hence accessing the data is also easier and fast. This database model was used to map many-to-many data relationships.
- Sets are used to define one-to-many relationships between records that contain one owner, many members.


AGENTS
REPRESENT
MANAGES
CLIENTS
ENTERTAINERS
MAKE
SCHEDULE
PERFORM
PLAY
PAYMENTS
ENGAGEMENTS
MUSIC STYLE

The Network Database Model



- Some of the popular network databases are:

1. Integrated Data Store (IDS)
2. IDMS (Integrated Database Management System)
3. TurboIMAGE
4. Univac DMS-1100

Advantages & Disadvantages

- The benefits of the network model include:

1. **Simple Concept:** Similar to the hierarchical model, this model is simple and the implementation is effortless.
2. **Ability to Manage More Relationship Types:** The network model has the ability to manage one-to-one (1: 1) as well as many-to-many (N: N) relationships.
3. **Easy Access to Data:** Accessing the data is simpler when compared to the hierarchical model.
4. **Data Integrity:** In a network model, there's always a connection between the parent and the child segments because it depends on the parent-child relationship.
5. **Data Independence:** Data independence is better in network models as opposed to the hierarchical models.

- The drawbacks of the network model include:

1. **System Complexity:** Each and every record has to be maintained with the help of pointers, which makes the database structure more complex.
2. **Functional Flaws:** Because a great number of pointers is essential, insertion, updates, and deletion become more complex.
3. **Lack of Structural Independence:** A change in structure demands a change in the application as well, which leads to lack of structural independence.

OODB

- **Object-oriented databases** (OODB) are databases that represent data in the form of objects and classes.
- In object-oriented terminology, an **object** is a real-world entity, and a **class** is a collection of objects.
- Object-oriented databases follow the fundamental principles of object-oriented programming (OOP).
- The combination of relational model features (concurrency, transaction, and recovery) with object-oriented principles results in an object-oriented database model.
- Some examples are GemStone/S, NeoDatis ODB, ObjectDatabase++, ObjectDB, ZODB

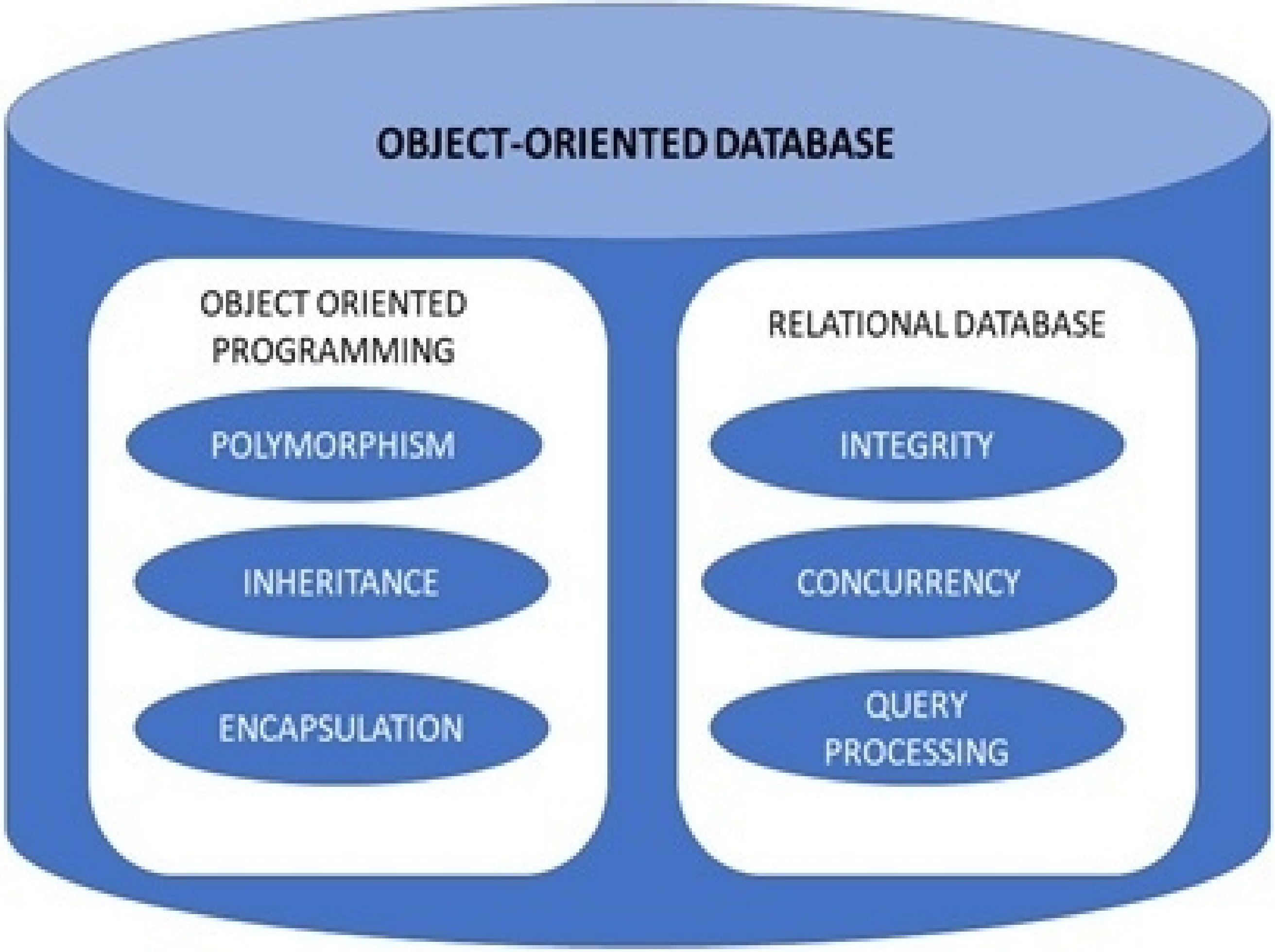
- The object-oriented database model contains the following properties.
- **Object-oriented programming properties**

1. Objects
2. Classes
3. Inheritance
4. Polymorphism
5. Encapsulation

- **Relational database properties**

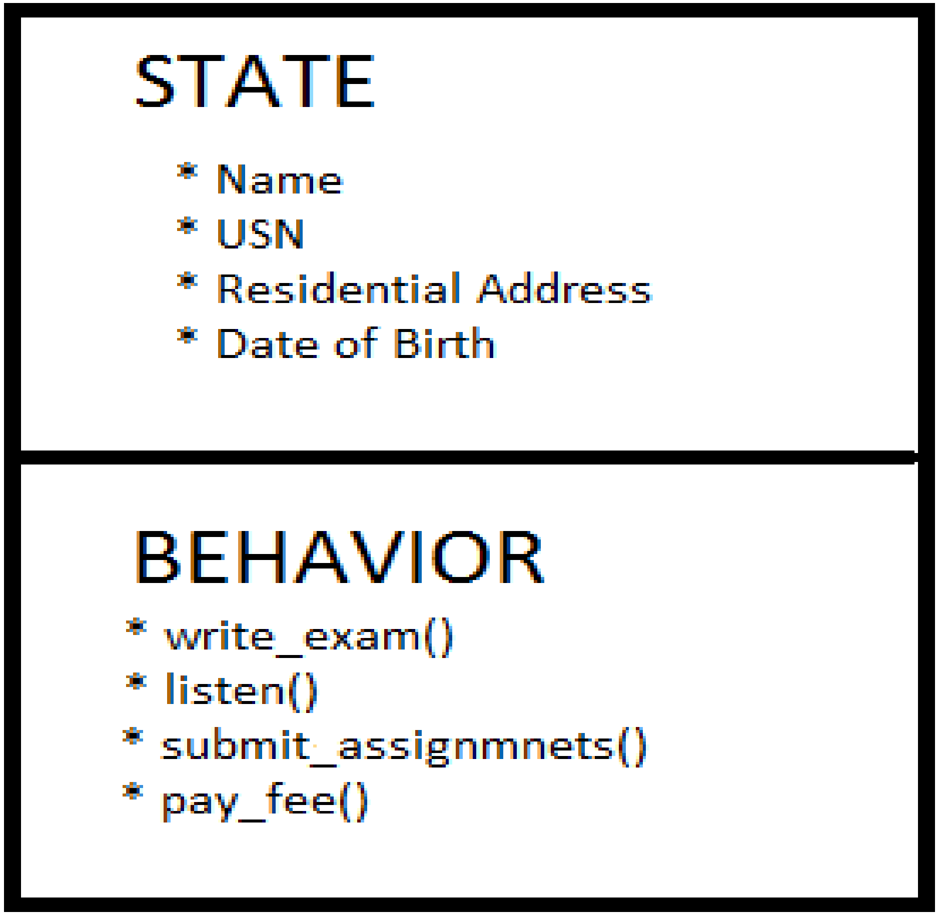
1. Atomicity
2. Consistency
3. Integrity
4. Durability
5. Concurrency
6. Query processing

- That said, we can use the following formula to outline the OODBM:
- **Object-Oriented Programming + Relational Database Features = Object-Oriented Database Model**
- The figure below outlines the object-oriented database model along with its principles and features.


OBJECT-ORIENTED DATABASE
OBJECT ORIENTED PROGRAMMING
POLYMORPHISM
INHERITANCE
ENCAPSULATION
RELATIONAL DATABASE
INTEGRITY
CONCURRENCY
QUERY PROCESSING

Fundamental Concept

- The object-oriented approach considers all entities as objects. An **object** has properties (state) and methods (behavior).
- Each object is identified using a unique object identifier. For example, let us consider a real-world entity called 'Student'.
- A student has states or properties such as a name, USN, date of birth, address, etc. Similarly, the student has behaviors or methods including attending classes, writing exams, paying fees, etc.
- This next figure below shows how the 'Student' object can be represented.



STATE

* Name
* USN
* Residential Address
* Date of Birth

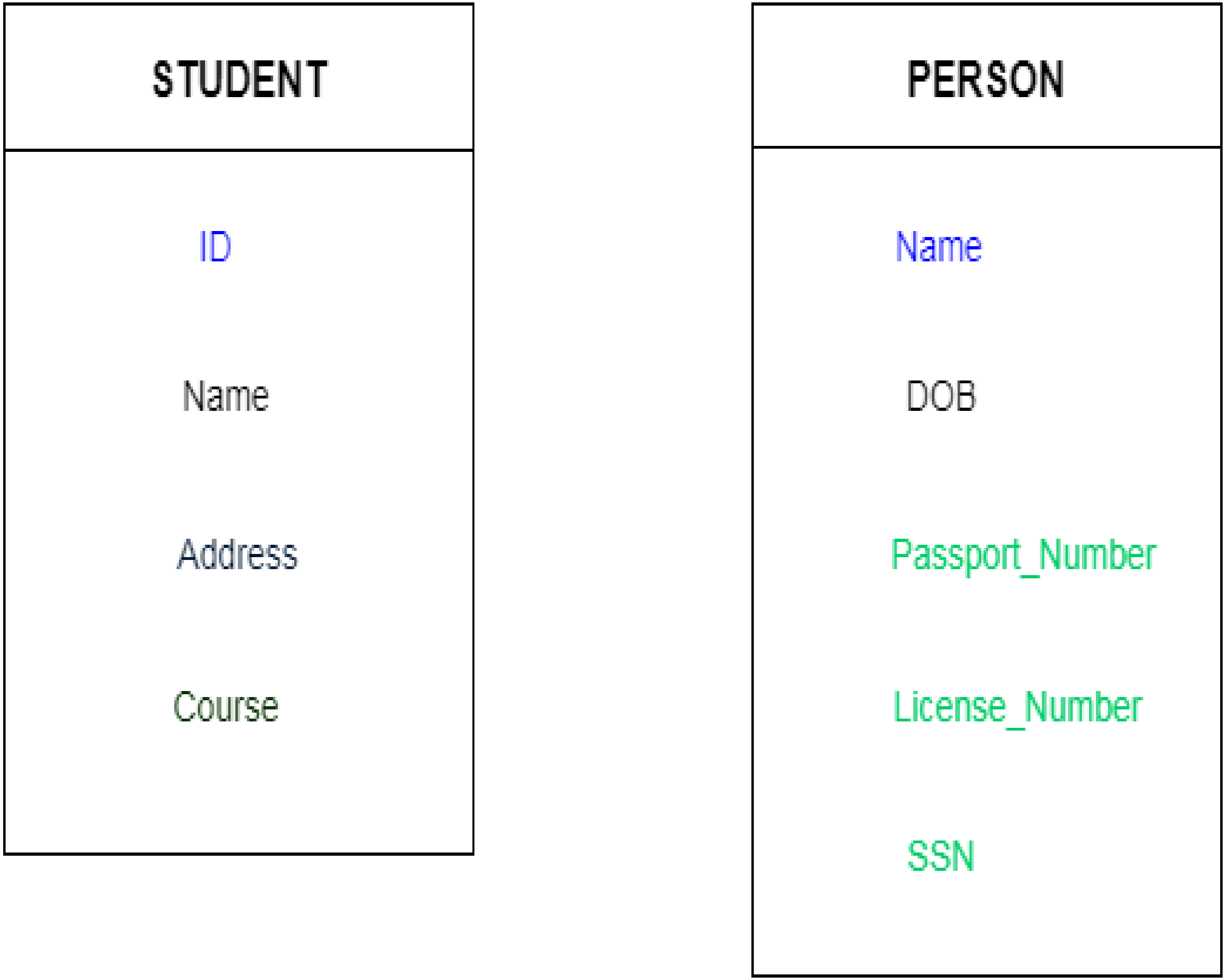
BEHAVIOR

* write_exam()
* listen()
* submit_assignmnets()
* pay_fee()

- The object-oriented database allows for the creation of persistent objects.
- A **persistent object** is one that lives in computer memory even after completing its execution. This is different from the lifespan of **normal objects**, which expire after execution, are destroyed immediately, and freed from memory.
- Object persistence solves the database challenges of concurrency and recovery.

Keys

- Keys play an important role in the relational database.
- It is used to uniquely identify any record or row of data from the table.
- It is also used to establish and identify relationships between tables.
- **For example:** In Student table, ID is used as a key because it is unique for each student. In PERSON table, passport_number, license_number, SSN are keys since they are unique for each person.



STUDENT

ID

Name

Address

Course

PERSON

Name

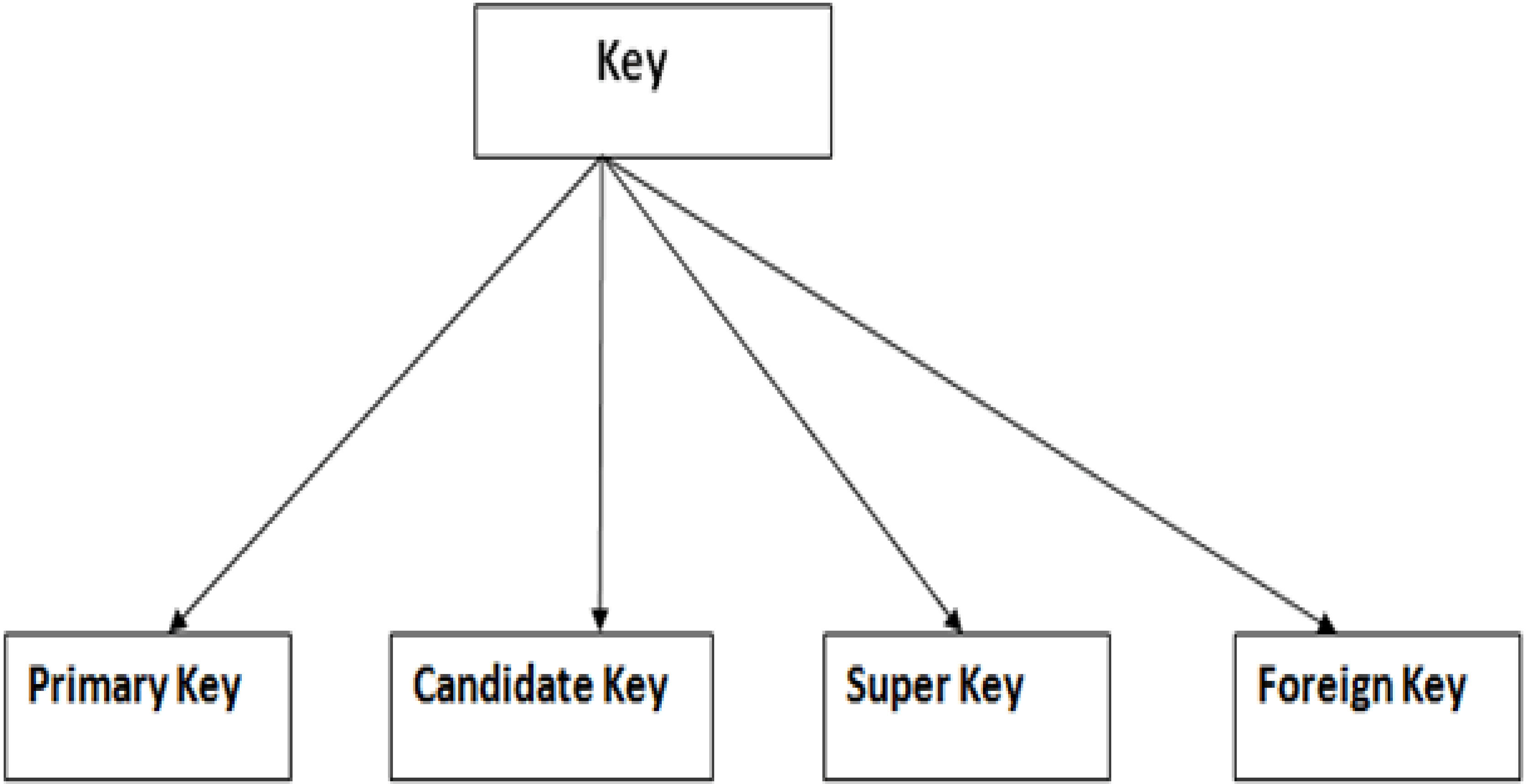
DOB

Passport_Number

License_Number

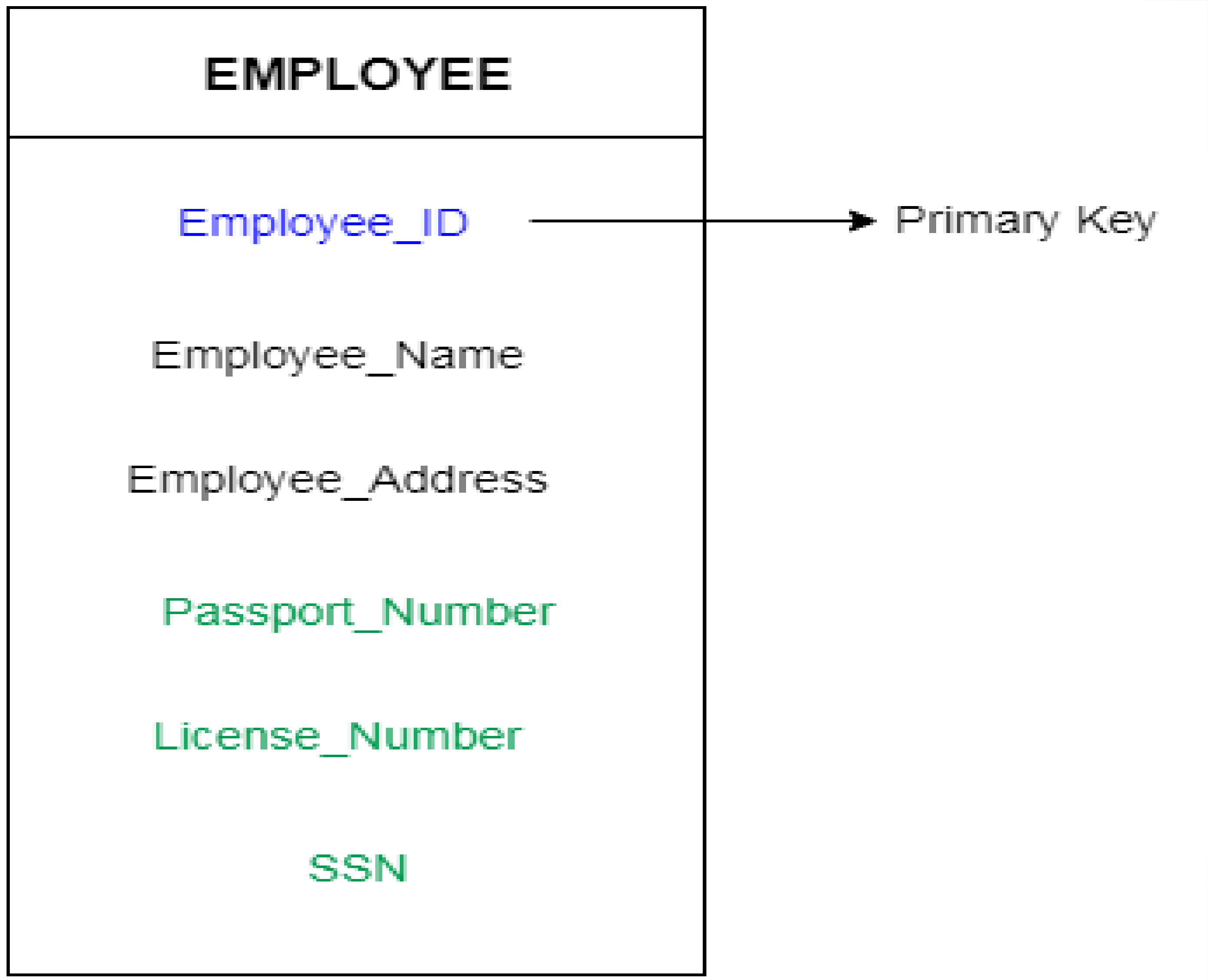
SSN

Types of Keys



Primary Key

- It is the first key which is used to identify one and only one instance of an entity uniquely. An entity can contain multiple keys as we saw in PERSON table. The key which is most suitable from those lists become a primary key.
- In the EMPLOYEE table, ID can be primary key since it is unique for each employee. In the EMPLOYEE table, we can even select License_Number and Passport_Number as primary key since they are also unique.
- For each entity, selection of the primary key is based on requirement and developers.



EMPLOYEE

Employee_ID → Primary Key

Employee_Name

Employee_Address

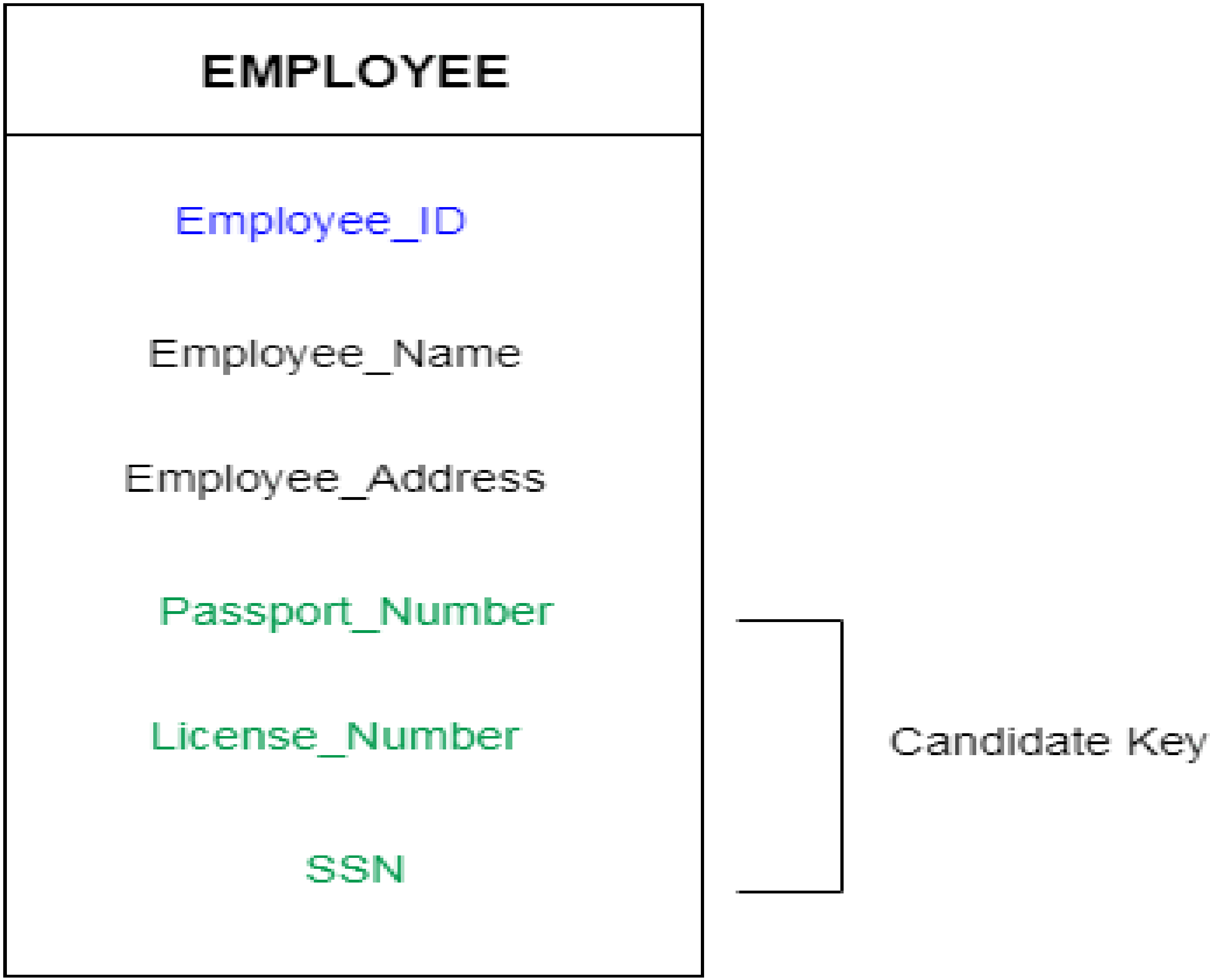
Passport_Number

License_Number

SSN

Candidate Key

- A candidate key is an attribute or set of an attribute which can uniquely identify a tuple.
- The remaining attributes except for primary key are considered as a candidate key. The candidate keys are as strong as the primary key.
- **For example:** In the EMPLOYEE table, id is best suited for the primary key. Rest of the attributes like SSN, Passport_Number, and License_Number, etc. are considered as a candidate key.



EMPLOYEE

Employee_ID

Employee_Name

Employee_Address

Passport_Number

License_Number

SSN

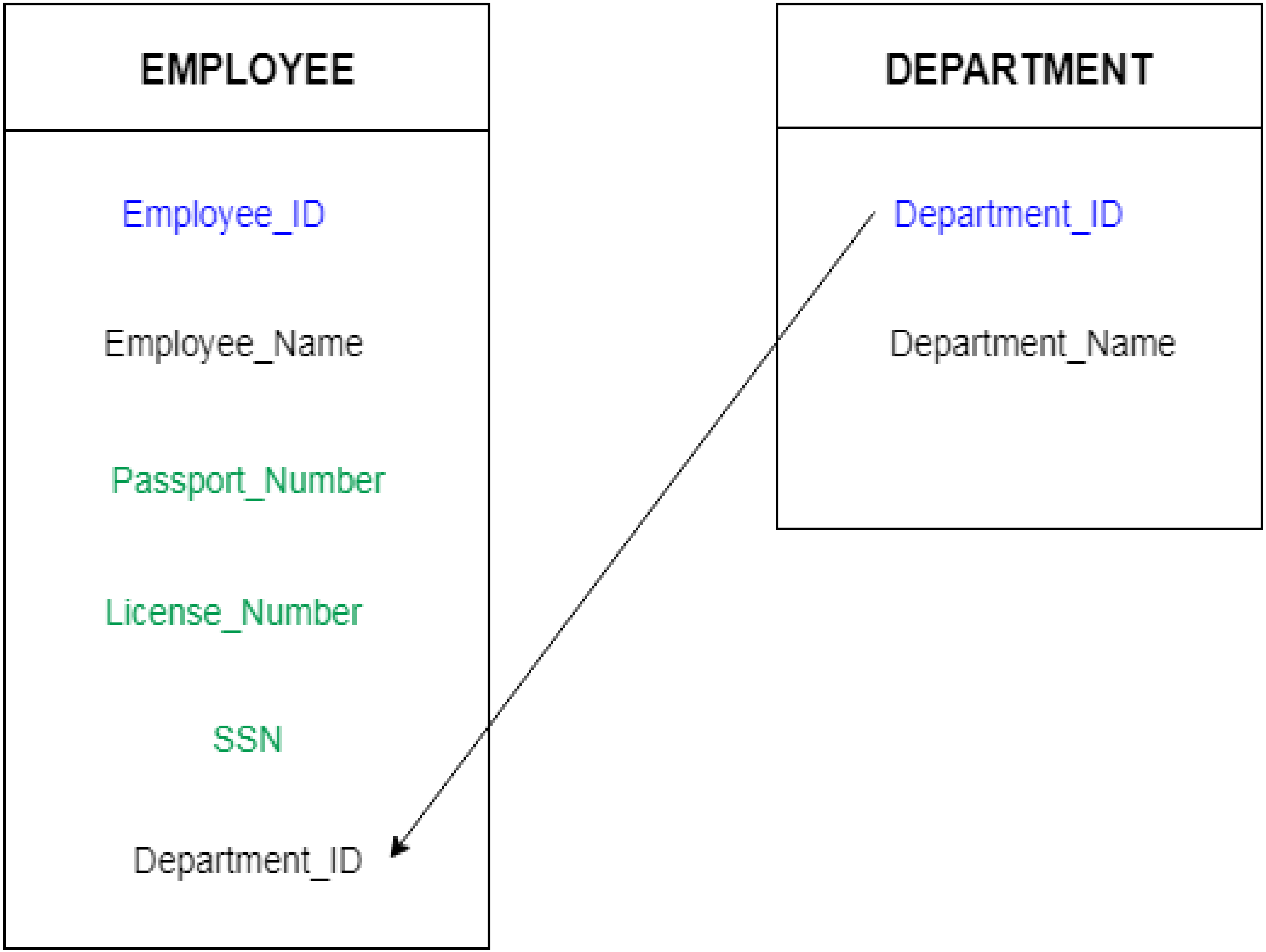
Candidate Key

Super Key

- Super key is a set of an attribute which can uniquely identify a tuple. Super key is a superset of a candidate key.
- **For example:** In the above EMPLOYEE table, for(EMPLOEE_ID, EMPLOYEE_NAME) the name of two employees can be the same, but their EMPLYEE_ID can't be the same. Hence, this combination can also be a key.
- The super key would be EMPLOYEE-ID, (EMPLOYEE_ID, EMPLOYEE-NAME), etc.

Foreign Key

- Foreign keys are the column of the table which is used to point to the primary key of another table.
- In a company, every employee works in a specific department, and employee and department are two different entities. So we can't store the information of the department in the employee table. That's why we link these two tables through the primary key of one table.
- We add the primary key of the DEPARTMENT table, Department_Id as a new attribute in the EMPLOYEE table.
- Now in the EMPLOYEE table, Department_Id is the foreign key, and both the tables are related.



EMPLOYEE

Employee_ID

Employee_Name

Passport_Number

License_Number

SSN

Department_ID

DEPARTMENT

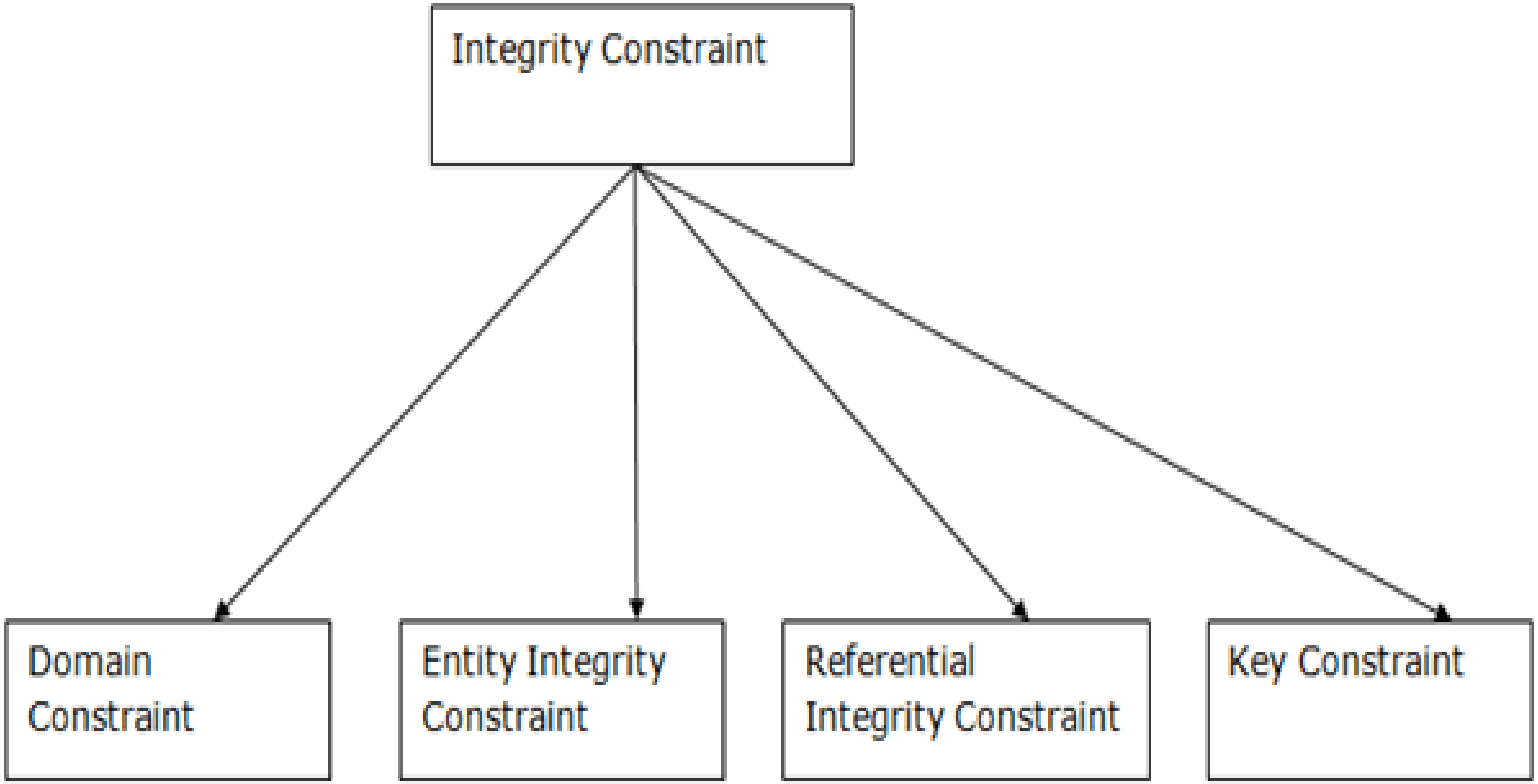
Department_ID

Department_Name

Integrity Constraints

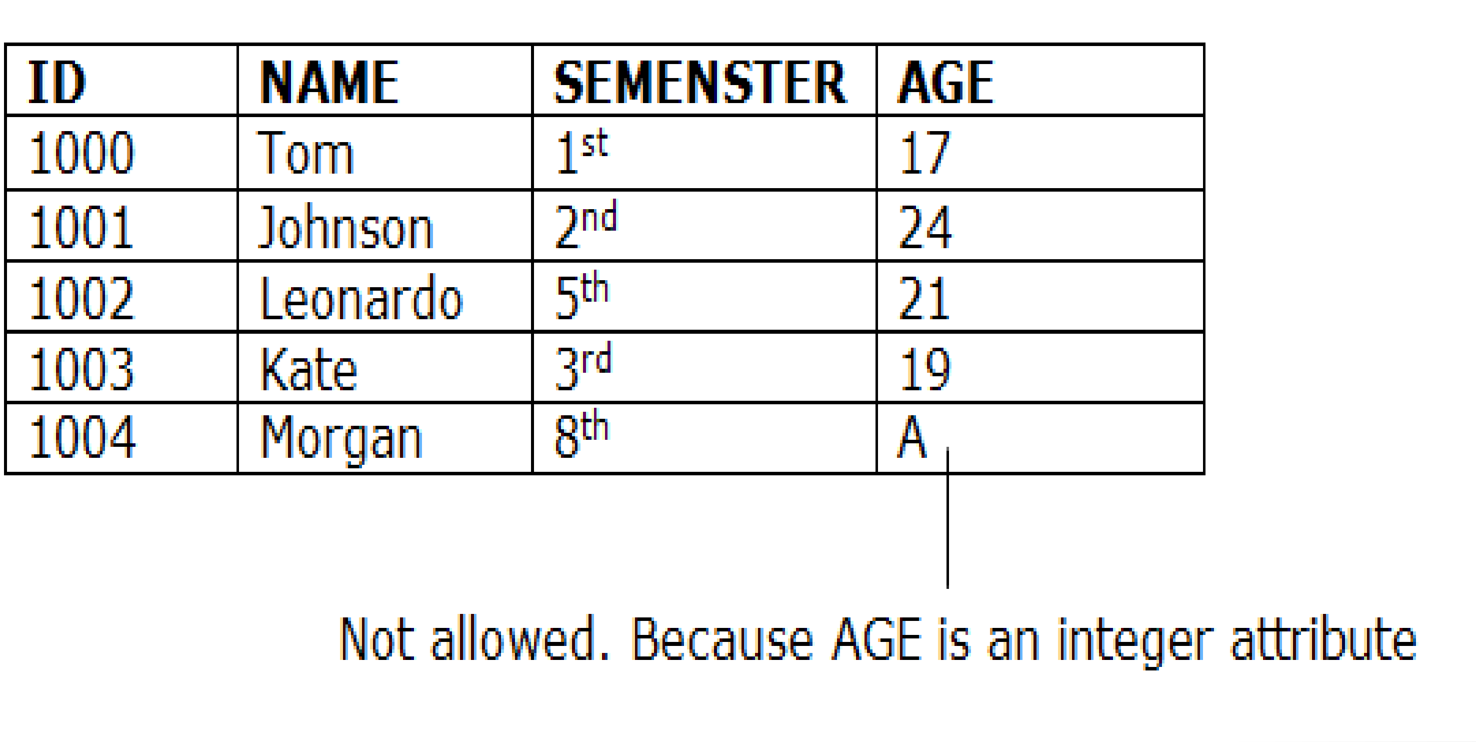
- Integrity constraints are a set of rules. It is used to maintain the quality of information.
- Integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.
- Thus, integrity constraint is used to guard against accidental damage to the database.

Types of Integrity Constraints



Domain Constraints

- Domain constraints 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.

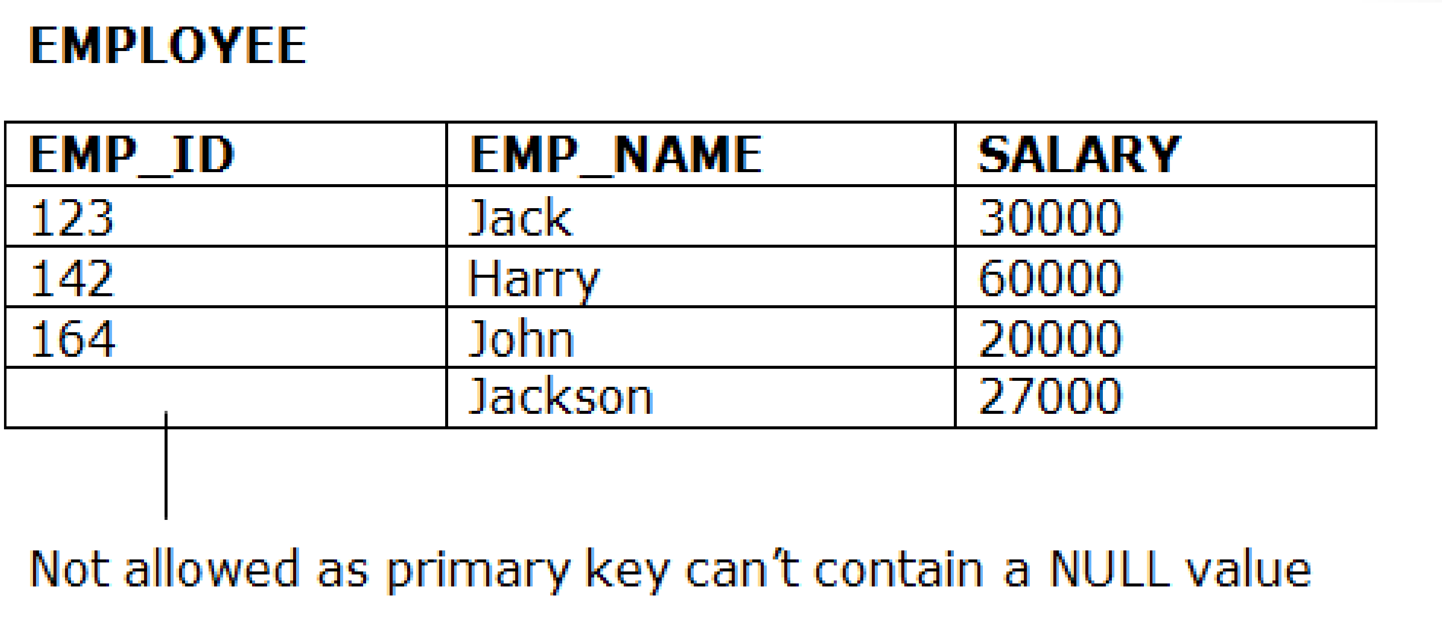


ID	NAME	SEMENSTER	AGE
1000	Tom	1st	17
1001	Johnson	2nd	24
1002	Leonardo	5th	21
1003	Kate	3rd	19
1004	Morgan	8th	A

Not allowed. Because AGE is an integer attribute

Entity Integrity Constraint

- The entity integrity 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.



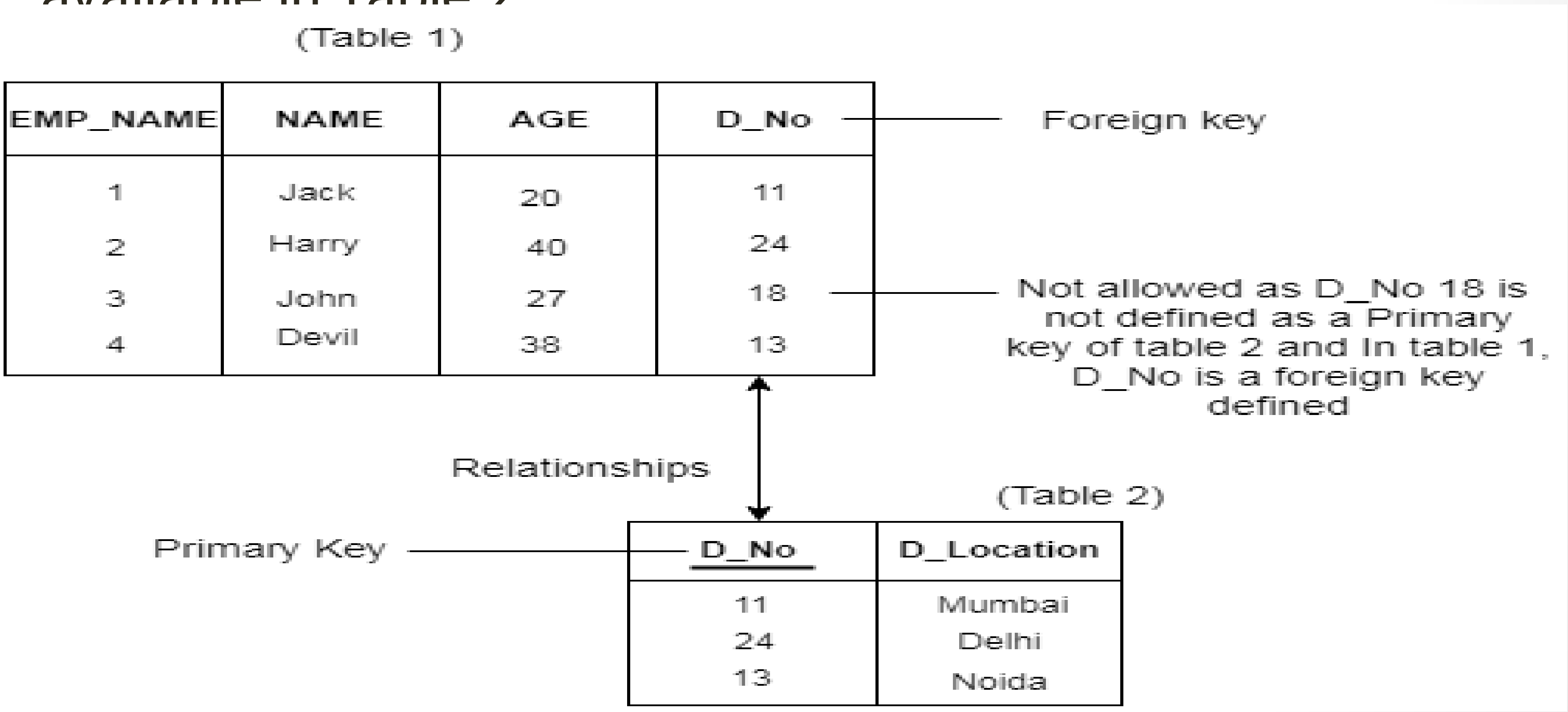
EMPLOYEE

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 Integrity Constraints

- A referential integrity 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



(Table 1)

EMP_NAME	NAME	AGE	D_No
1	Jack	20	11
2	Harry	40	24
3	John	27	18
4	Devil	38	13

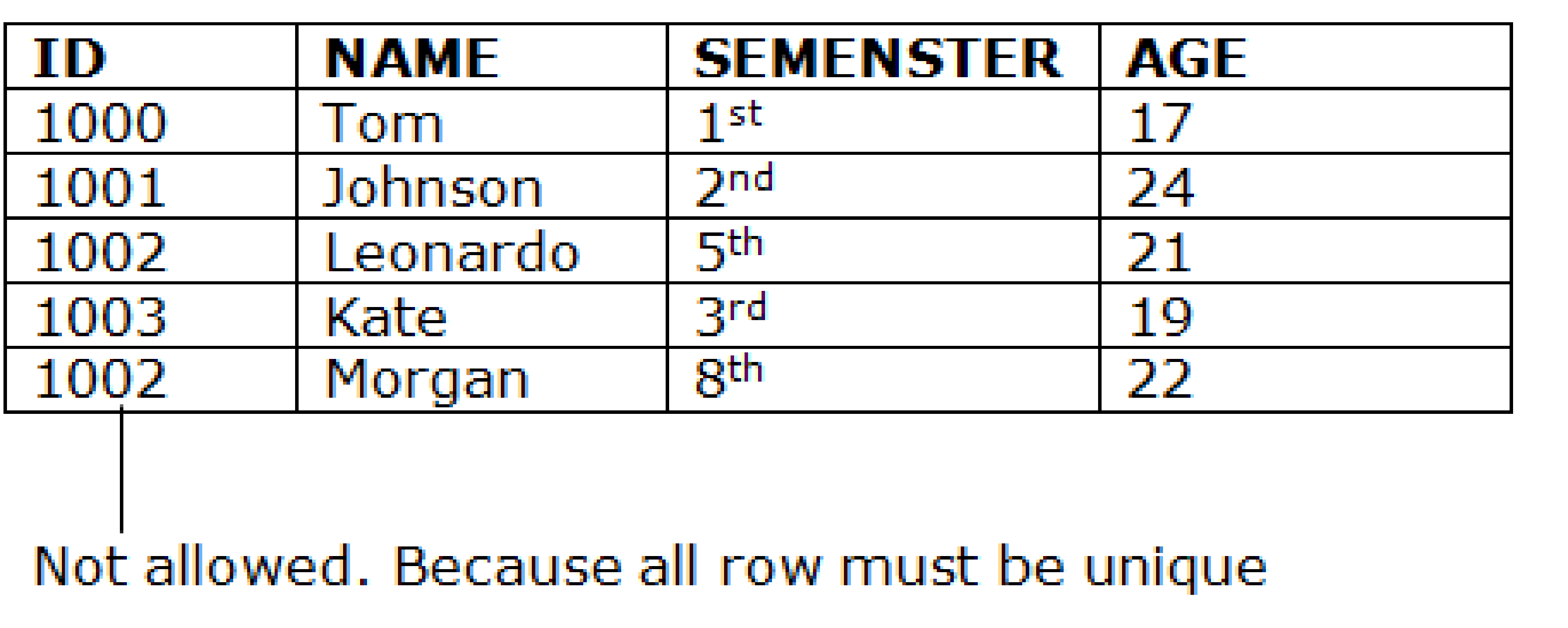
Foreign key
Not allowed as D_No 18 is not defined as a Primary key of table 2 and In table 1, D_No is a foreign key defined
Relationships
Primary Key

(Table 2)

D_No	D_Location
11	Mumbai
24	Delhi
13	Noida

Key Integrity Constraint

- Keys are the entity set that is used to identify an entity within its entity set uniquely.
- An entity set can have multiple keys, but out of which one key will be the primary key. A primary key can contain a unique value relational table.



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

Database Administrator (DBA)

- A Database Administrator is a person or a group of person who are responsible for managing all the activities related to database system. This job requires a high level of expertise by a person or group of person. There are very rare chances that only a single person can manage all the database system activities so companies always have a group of people who take care of database system.

1. Deciding the hardware device

- Depending upon the cost, performance and efficiency of the hardware, it is DBA who have the duty of deciding which hardware devise will suit the company requirement. It is hardware that is an interface between end users and database so it needed to be of best quality.

2. Managing Data Integrity

- Data integrity should be managed accurately because it protects the data from unauthorized use. DBA manages relationship between the data to maintain data consistency.

3. Decides Data Recovery and Back up method

- If any company is having a big database, then it is likely to happen that database may fail at any instance. It is require that a DBA takes backup of entire database in regular time span. DBA has to decide that how much data should be backed up and how frequently the back should be taken. Also the recovery of data base is done by DBA if they have lost the database.

4. Tuning Database Performance

- Database performance plays an important role for any business. If user is not able to fetch data speedily then it may loss company business. So by tuning an modifying sql commands a DBA can improves the performance of database.

5. Capacity Issues

- All the databases have their limits of storing data in it and the physical memory also has some limitations. DBA has to decide the limit and capacity of database and all the issues related to it.

6. Database design

- The logical design of the database is designed by the DBA. Also a DBA is responsible for physical design, external model design, and integrity control.

7. Database accessibility

- DBA writes subschema to decide the accessibility of database. He decides the users of the database and also which data is to be used by which user. No user has to power to access the entire database without the permission of DBA.

8. Decides validation checks on data

- DBA has to decide which data should be used and what kind of data is accurate for the company. So he always puts validation checks on data to make it more accurate and consistence.

9. Monitoring performance

- If database is working properly then it doesn't mean that there is no task for the DBA. Yes of course, he has to monitor the performance of the database. A DBA monitors the CPU and memory usage.

10. Decides content of the database

- A database system has many kind of content information in it. DBA decides fields, types of fields, and range of values of the content in the database system. One can say that DBA decides the structure of database files.

11. Provides help and support to user

- If any user needs help at any time then it is the duty of DBA to help him. Complete support is given to the users who are new to database by the DBA.

12. Database implementation

- Database has to be implemented before anyone can start using it. So DBA implements the database system. DBA has to supervise the database loading at the time of its implementation.

13. Improve query processing performance

- Queries made by the users should be performed speedily. As we have discussed that users need fast retrieval of answers so DBA improves query processing by improving their performance.