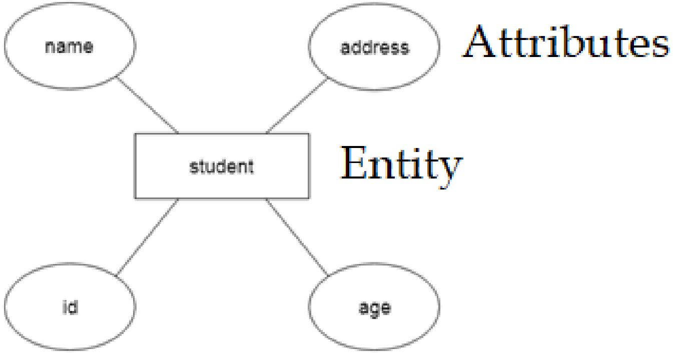
DBMS: Database Design

Akhilesh Arya

ER- Diagrams

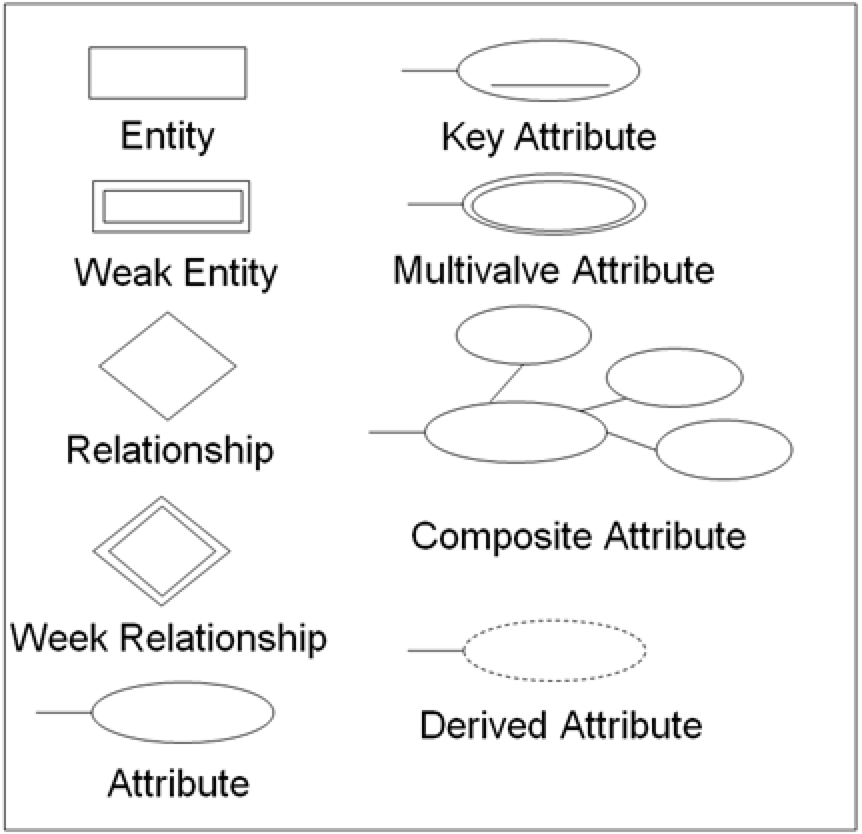
- An ER model is the logical representation of data as objects and relationships among them.
- These objects are known as entities, and relationship is an association among these entities.
- This model was designed by Peter Chen and published in 1976 papers



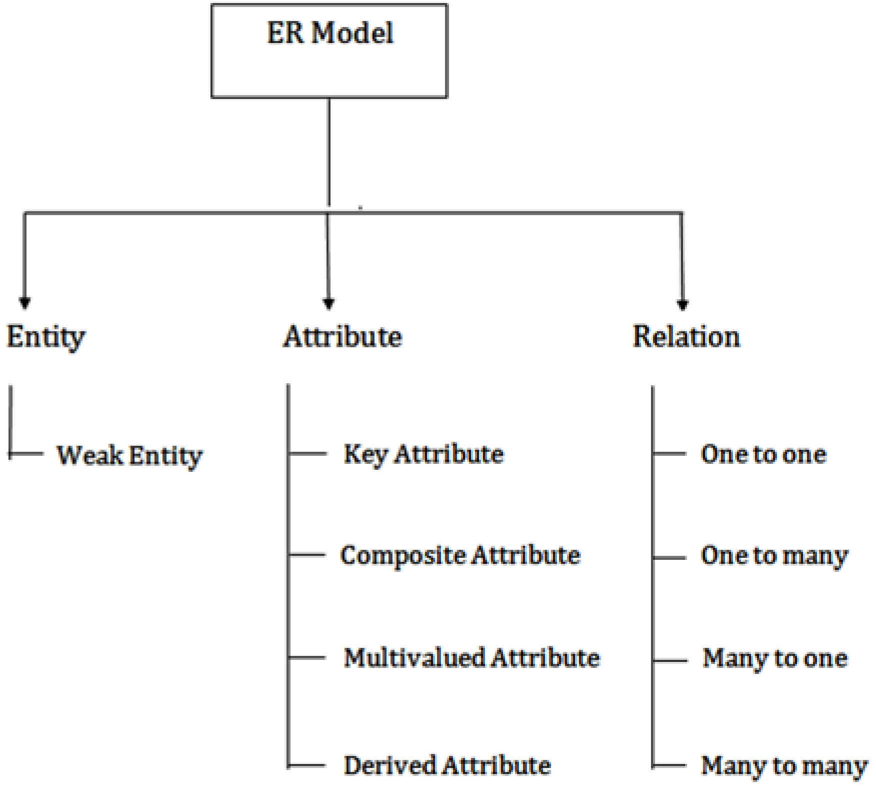
Cont..

- Entity Relationship diagrams have three different components-
  - Entities
  - Attributes
  - Relationships

Notation of ER- Diagram



Components of ER Diagram



ER Model

Entity

- Weak Entity

Attribute

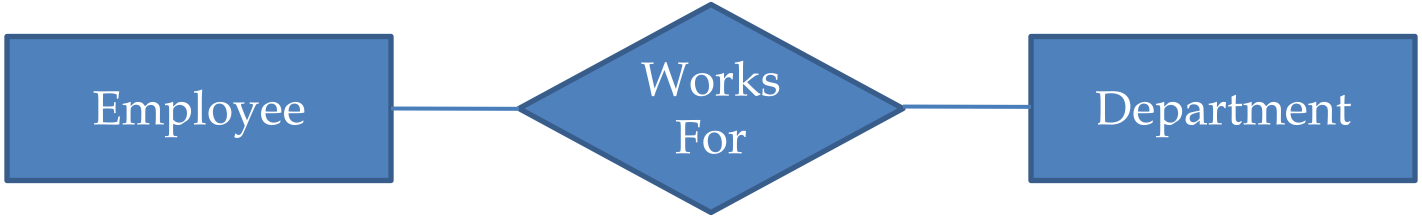
- Key Attribute
- Composite Attribute
- Multivalued Attribute
- Derived Attribute

Relation

- One to one
- One to many
- Many to one
- Many to many

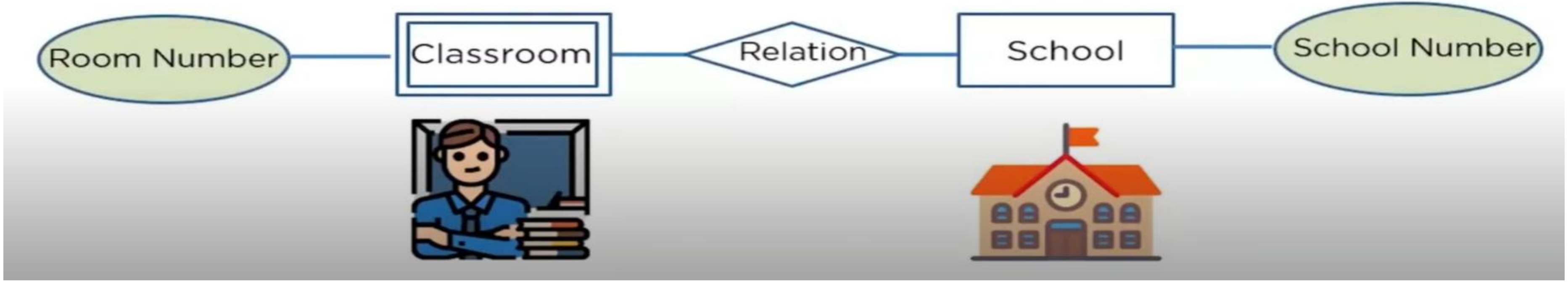
Entities

- Entities are the people, places, things, and events.
- In short, anything which an organization wants to store data about.



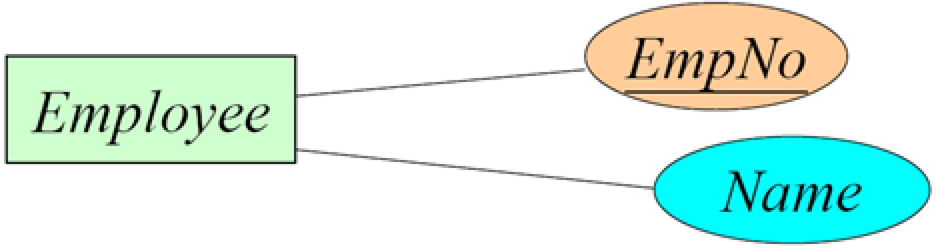
Cont..

- **Week Entity:** An entity that depends on another entity called a weak entity.
- The weak entity doesn't contain any key attribute of its own.
- The weak entity is represented by a double rectangle.



Attribute

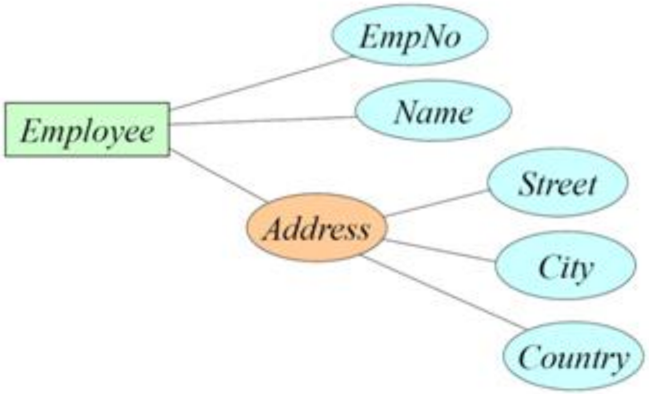
- Attributes are properties used to describe an entity
  - For example an EMPLOYEE entity may have the attributes Name, SSN, Address, Sex, BirthDate



Key Attribute: An attribute of an entity type for which each entity must have a unique value is called a key attribute of the entity type

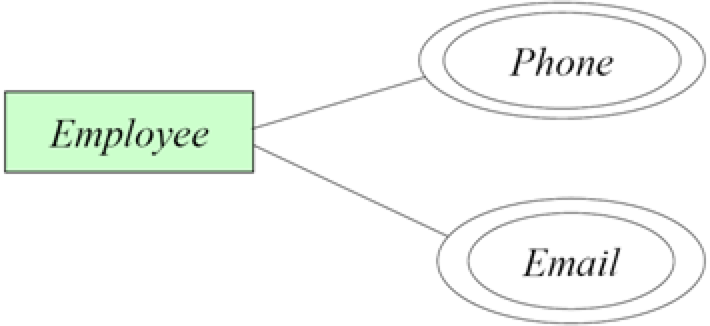
Cont..

- **Composite attribute:** consists of several components (e.g., address)



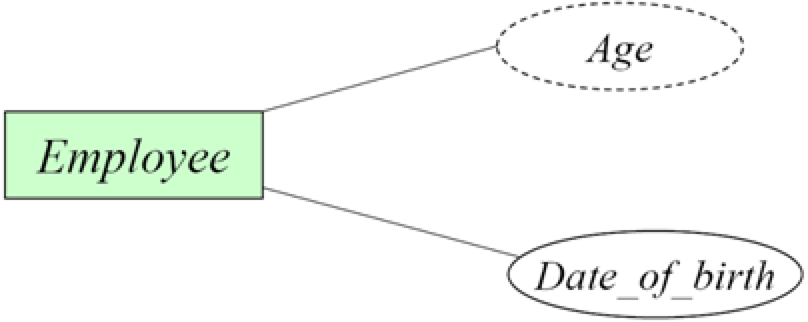
Cont..

- **Multivalued attribute:** contains more than one value

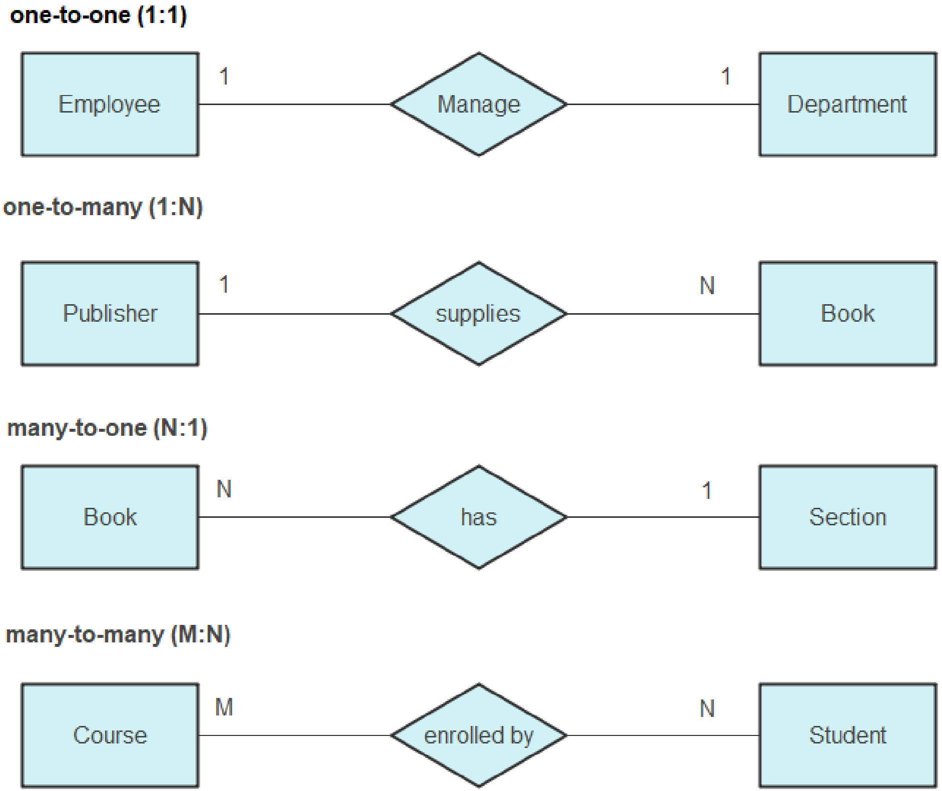


Cont..

- **Derived attribute:** computed from other attributes (e.g., age can be computed from the date of birth and the current date)

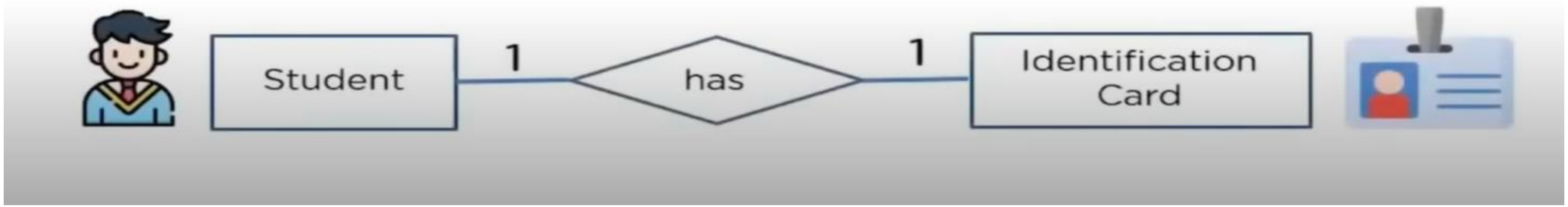


Relation


one-to-one (1:1)
Employee
1
Manage
1
Department
one-to-many (1:N)
Publisher
1
supplies
N
Book
many-to-one (N:1)
Book
N
has
1
Section
many-to-many (M:N)
Course
M
enrolled by
N
Student

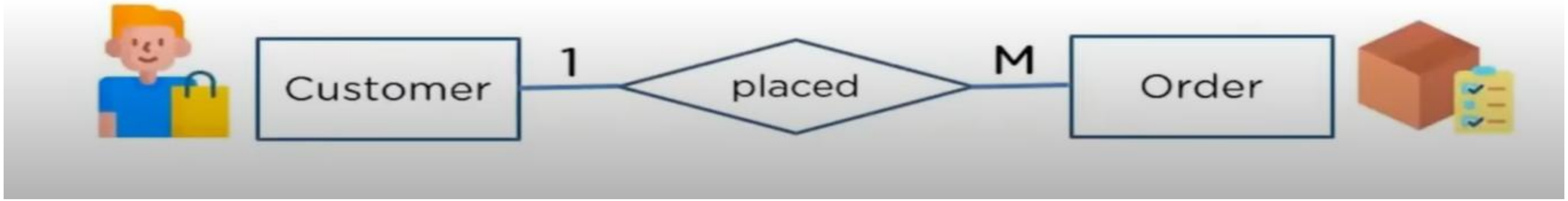
Example: One is to One

- A student has only one identification card ad an identification card is given to one person only



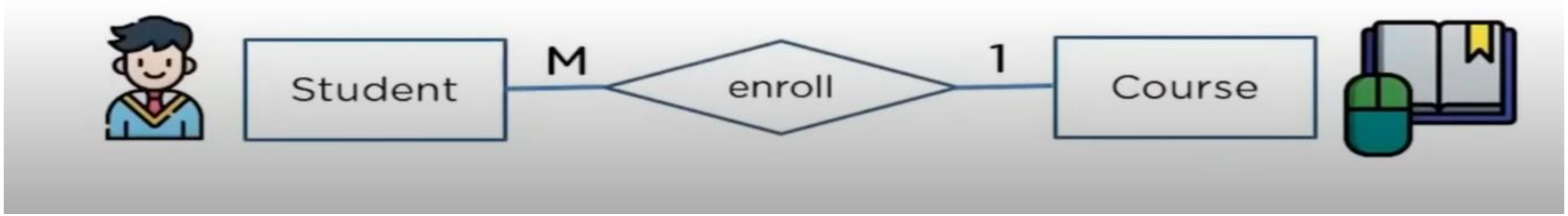
Example: One is to Many

- A customer can place multiple orders, but an order cannot be placed by many customers



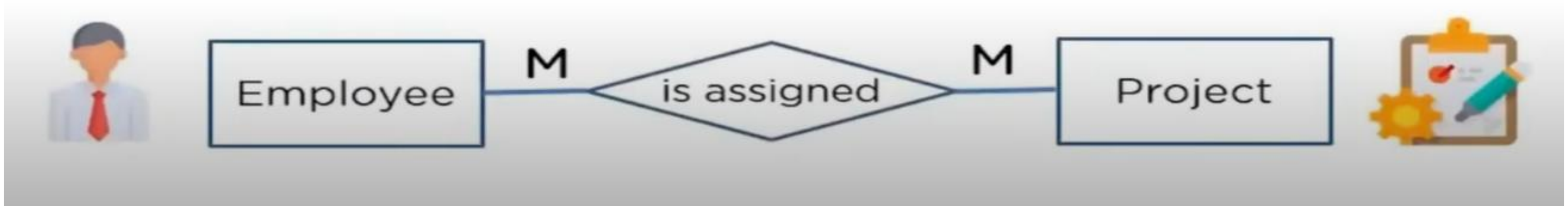
Example: Many to One

- Student enrolls for only one course, but a course can have many students.



Example: Many to Many

- Employee can have multiple projects and project can have many employees

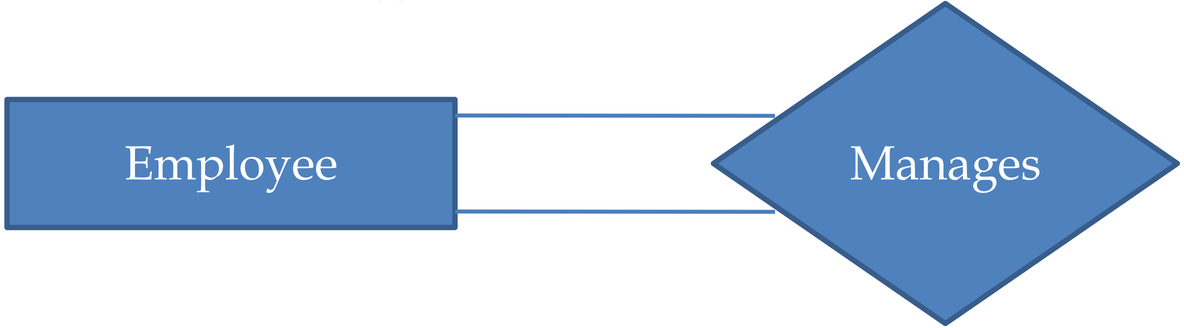


Relationship

- A relationship is association between entities
  - Unary
  - Binary
  - Ternary

Unary

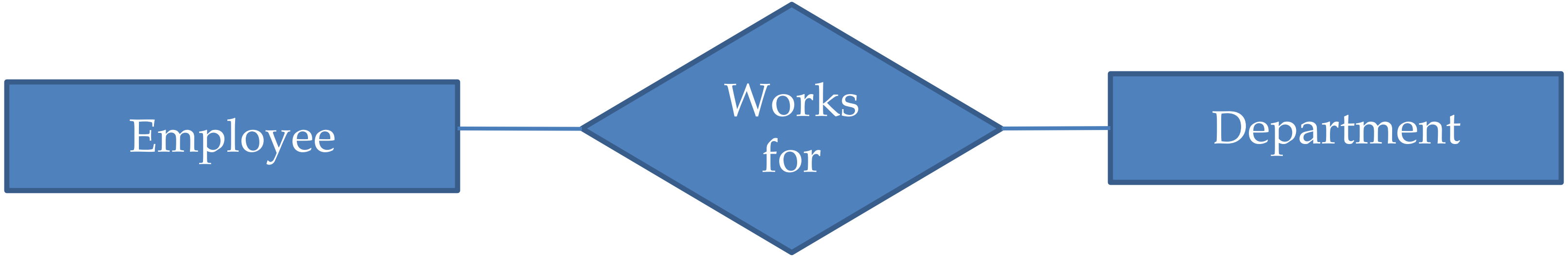
- A unary relationship is presented as a diamond which connects one entity to itself as a loop



The relationship means, some instance of the employee manages other instance of employee

Binary

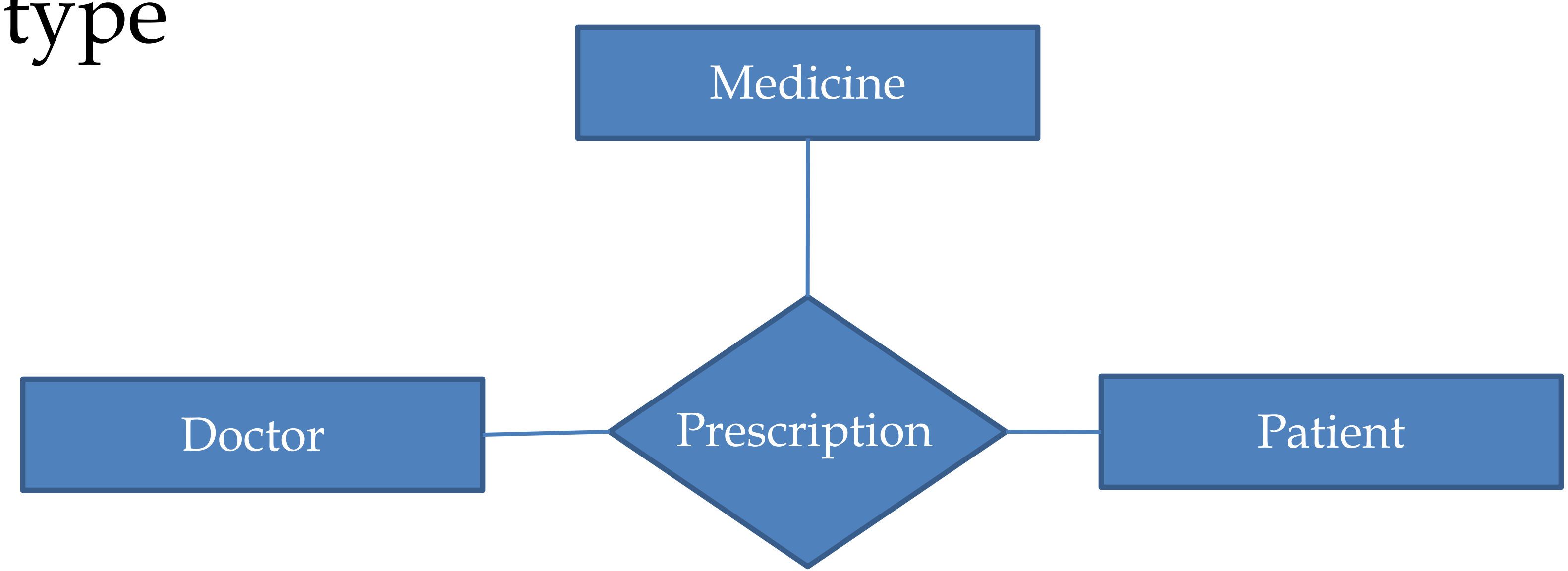
- Relationship between two entities types



Every Employee works for a department

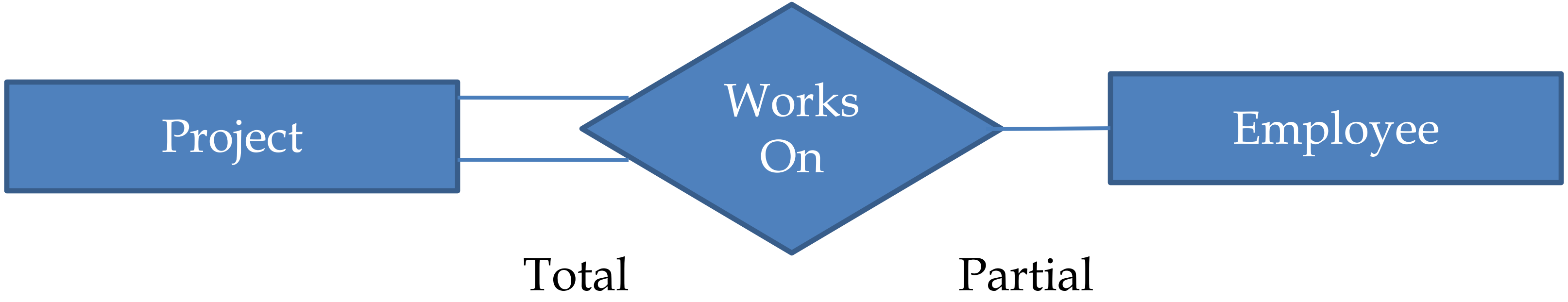
Ternary

- A relationship connecting three entity type



Participation

- **Partial:** all instance of the entity employee don't participate in the relationship, head of the department
- **Total:** all instance of the entity type department participate in the relation

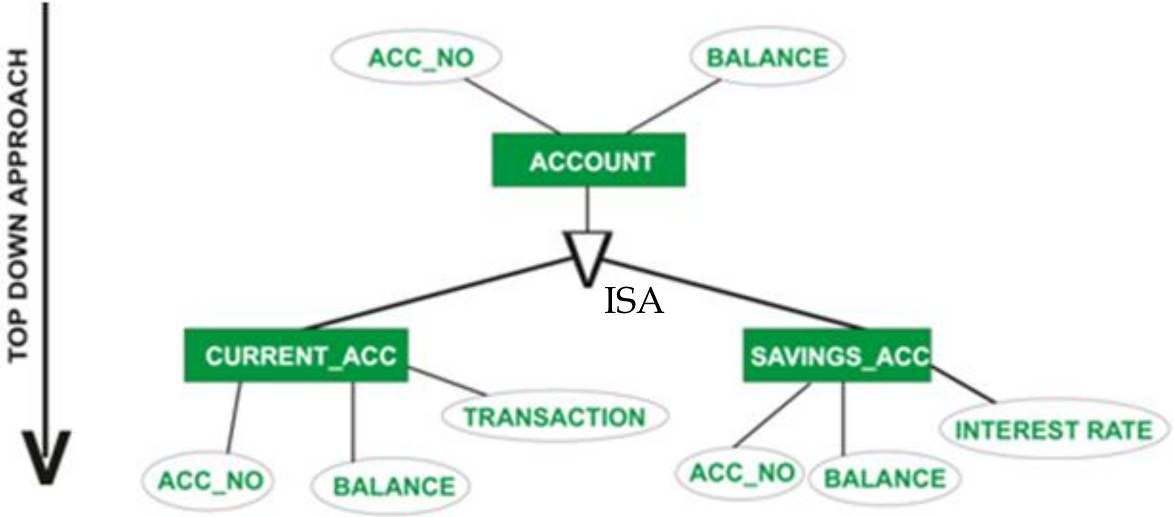


Extended ER Features

- Certain instances of the an entity class can include attributes that are not needed in other instances of the same entity class
- In this case it is useful to a **super-class/ sub-class** structure
- This structure is also called as **generalization/ specialization** hierarchy

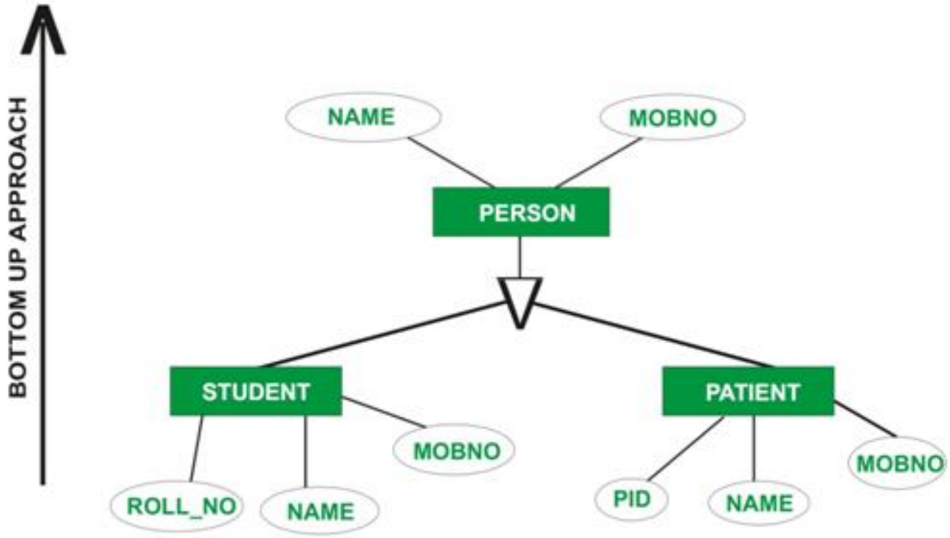
Specialization

- Is the process of defining a set of sub-class of an entity type; this entity is called the **super-class** of the specialization

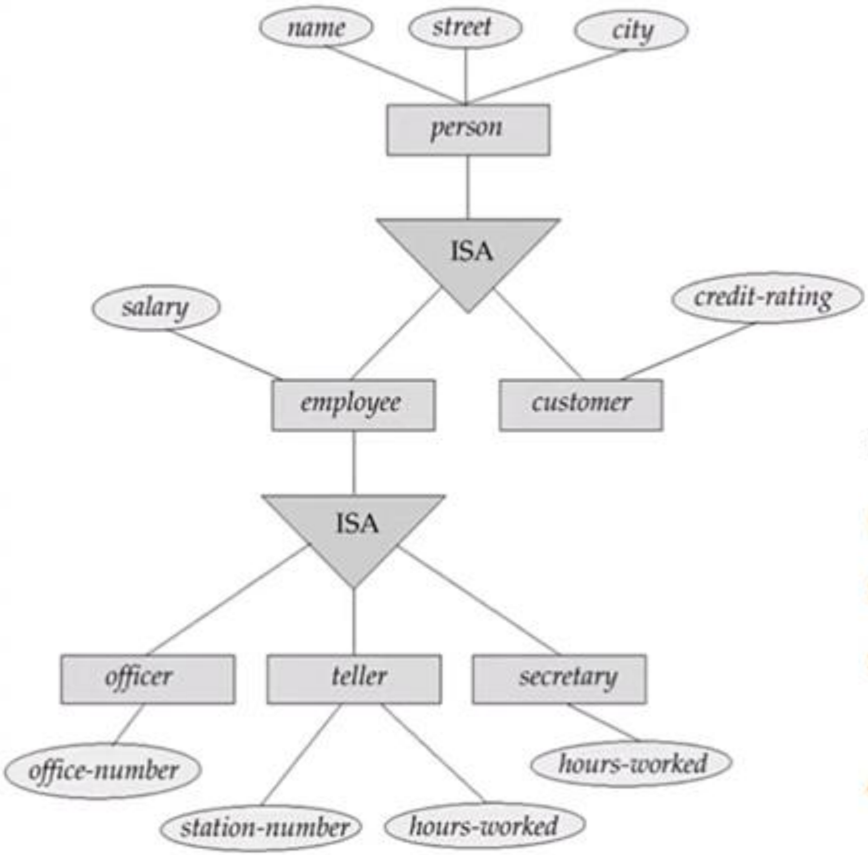


Generalization

- It is the form of abstraction that specifies that two or more entities that share common attributes can be generalized into higher level entity type



How Schema or Tables can be formed?



Four tables can be formed:

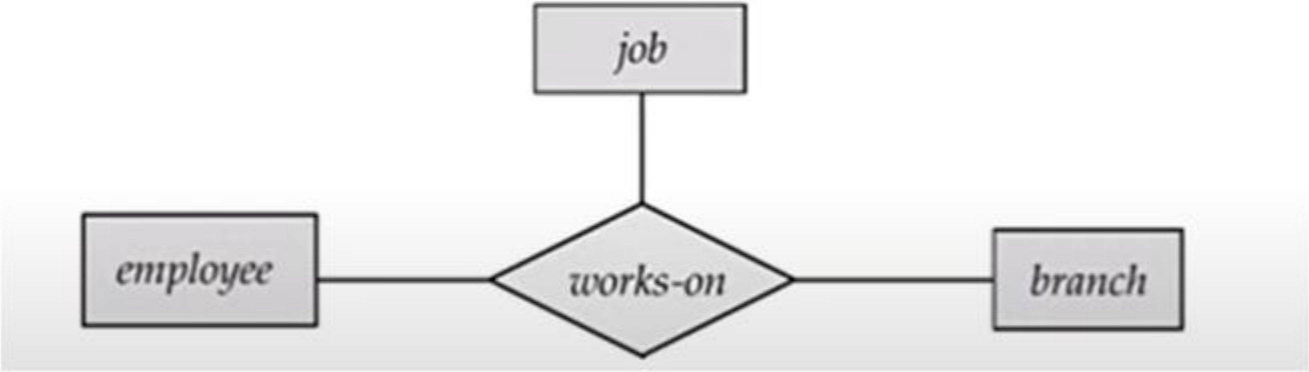
1. **customer** (name, street, city, credit_rating)
2. **officer** (name, street, city, salary, office_number)
3. **teller** (name, street, city, salary, station_number, hours_worked)
4. **secretary** (name, street, city, salary, hours_worked)

Aggregation

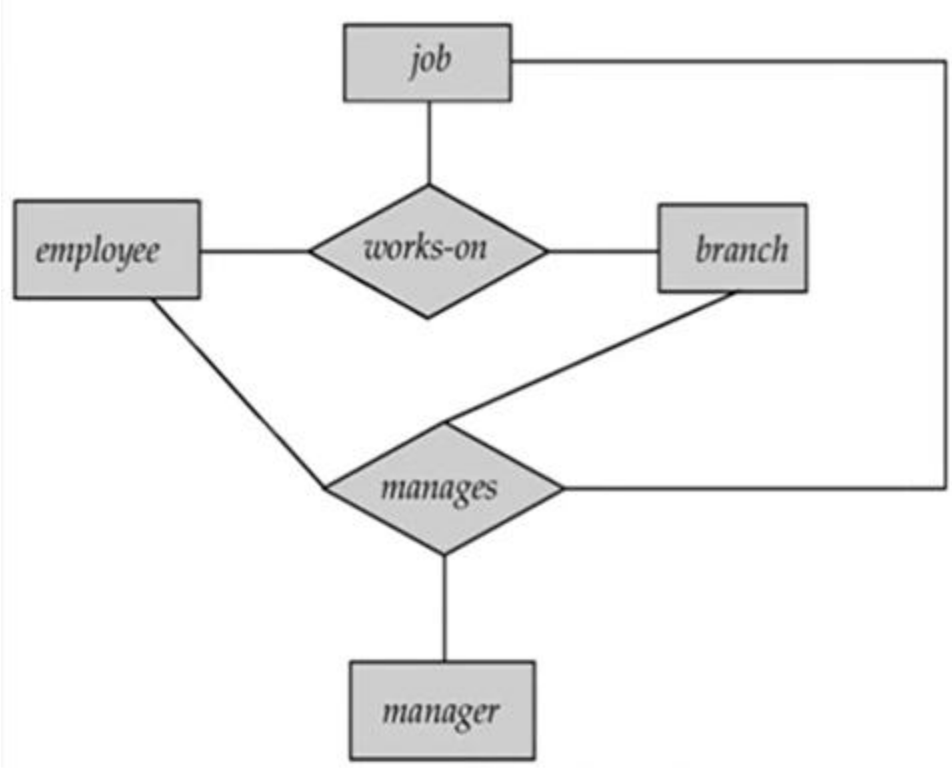
- Is used when we need to express a **relationship among relationship**
- Aggregation is an abstraction through which relationships are treated as a higher level entities
- A relationship between two entities is considered as a **single entity** and then we draw a relationship with this entity to another entity

Example

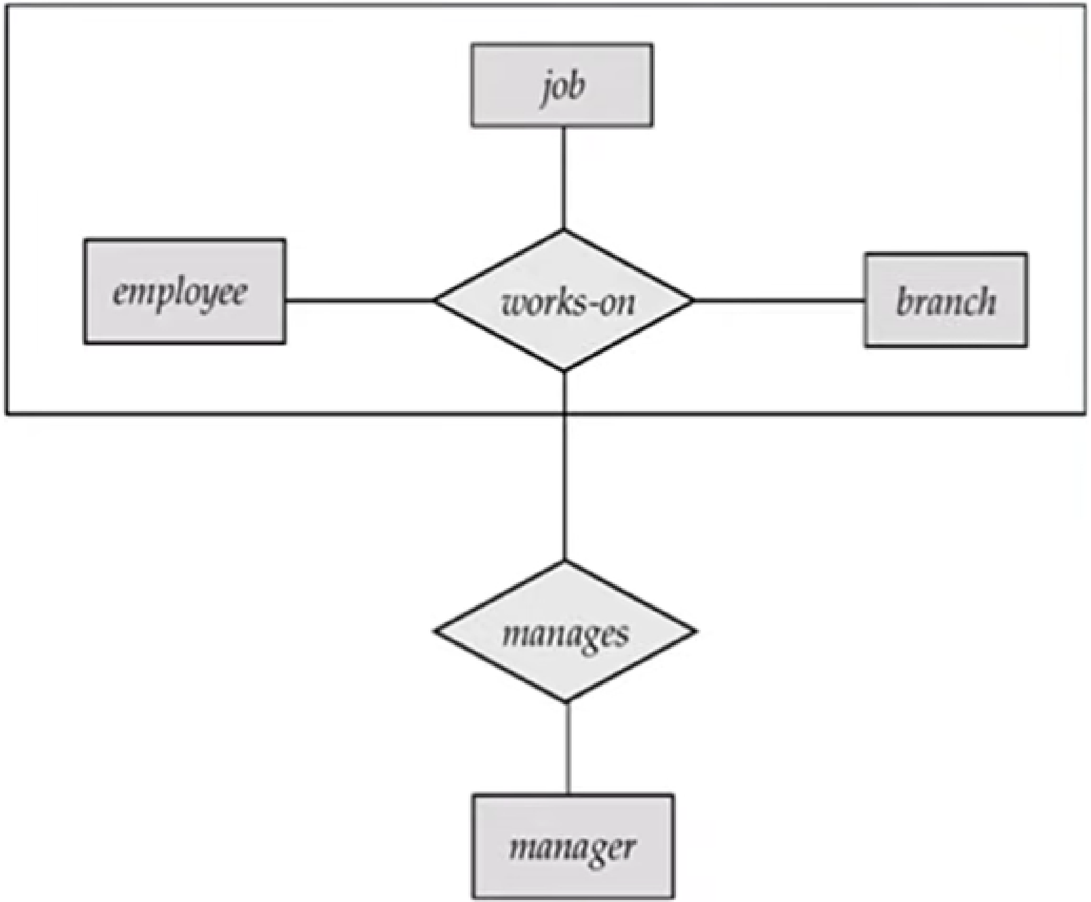
- Suppose we have a relationship among job, employee and branch
- Now if we want to add a manager entity into the existing relationship



- Manager entity will connect with all the three entity with the relation manages



- With aggregation (without introducing redundancy) the ER Diagram can be represented as:
  - An employee works on a job at particular branch
  - An employee, branch and job combination may have an associate manager



Steps to draw ER diagrams

1. Identify the entity
2. Identify the relationships
3. Identify the attributes (keys)
4. Identify participation constraints and other relevant attributes
5. Draw Complete ER diagram with all attributes including the primary keys

E-R model for college database

- Assumptions are given
  - A college contains many departments
  - Each department can offer many number of courses
  - Many instructor can work only in one department
  - An instructor can work only in one department
  - For each department there is a head
  - An instructor can be head of only one department
  - Each instructor can take any number of courses
  - A course can be taken by only one instructor
  - A student can enroll for any number of courses
  - Each course can have any number of student

1. Identify the Entities

1. Identify the Entities

- Department
- Student
- Course
- Instructor

2. Identify the relationships

- Between Student and Course
- Between Department and Course
- Between Department and Instructor
- Between Head of the Department and Instructor
- Between Course and instructor

2. Identify the relationships

- One course is enrolled by multiple students and one student enrolls for multiple courses hence the cardinality between course and student is many to many
- The department offers many courses and each course belongs to only one department, hence the cardinality between department and course is one to many
- One department has multiple instructors and one instructor belongs to one and only one department, hence the cardinality between department and instructor is one to many

Cont..

- Each department there is a “head of the department” and one instructor is “head of the department”, hence the cardinality is one to one.
- One course is taught by only one instructor, but the instrutor teaches many courses, hence the cardinality between course and instructor is many to one.

3. Identify the key attributes

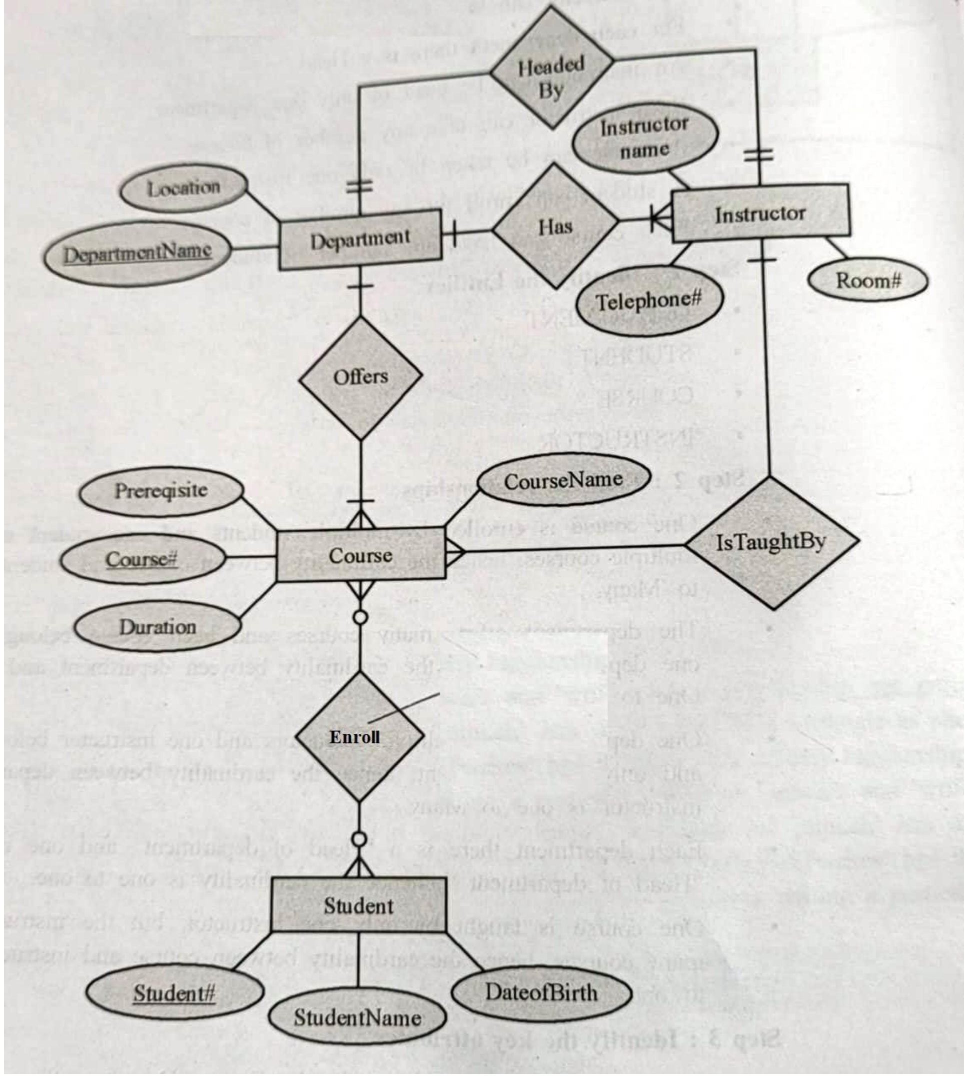
3. Identify the key attributes

- Deptname is the key attribute for the entity “Department”
- Courseid is the key attribute for “Course” entity
- Studid is the key attribute for “Student” entity
- Instructor name or Id is the key attribute for “instructor” entity

4. Identify other relevant attributes

4. Identify other relevant attributes

- For the department entity, the relevant attribute is location
- For course entity, course name, duration, prerequisite
- For instructor entity, room, telephone
- For student entity, student name, DOB



E-R diagram College Database

DBMS- 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.

Types of Keys

- Primary Key
- Candidate Key
- Super Key
- Foreign Key
- Alternate Key
- Composite Key

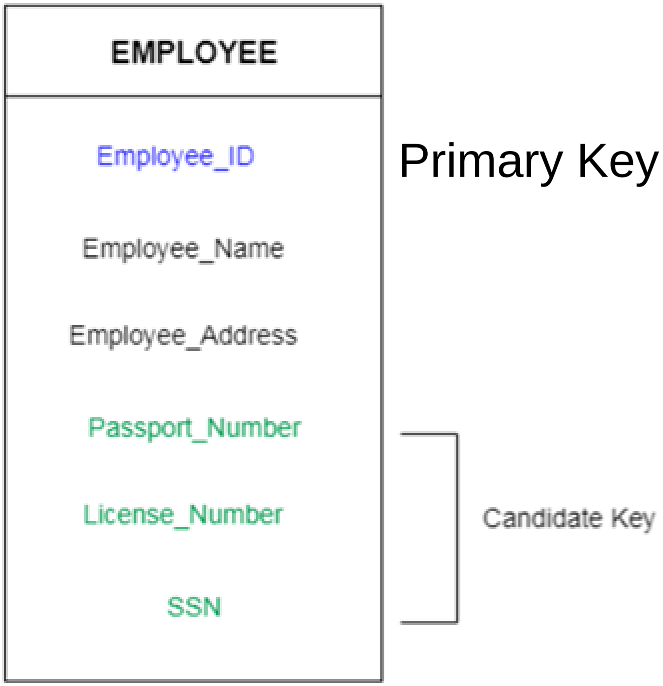
Primary Key

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

Candidate Key

- A candidate key is an attribute or set of attributes that can uniquely identify a tuple.
- Except for the primary key, the remaining attributes are considered a candidate key.
- The candidate keys are as strong as the primary key.
- In the EMPLOYEE: **SSN, Passport_Number, License_Number,** etc., are considered a candidate key.

Example



EMPLOYEE

Employee_ID

Employee_Name

Employee_Address

Passport_Number

License_Number

SSN

Primary Key

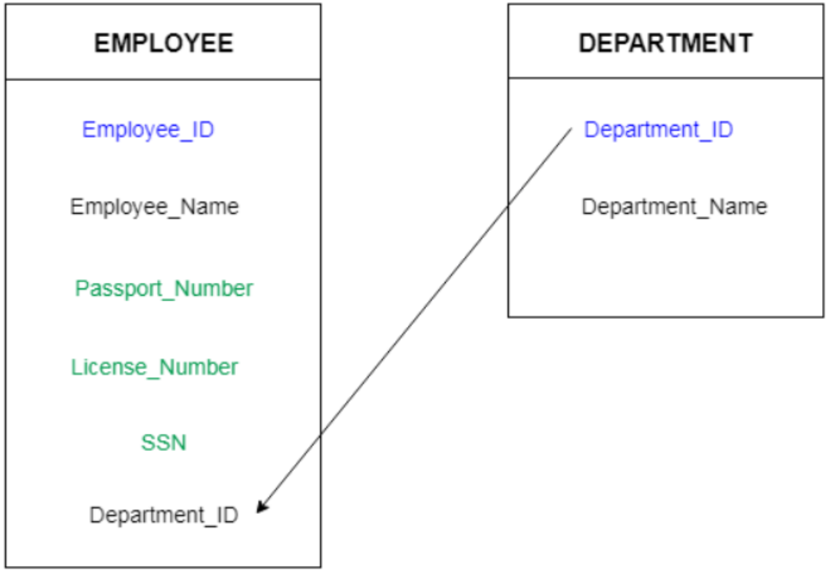
Candidate Key

Super Key

- Super key is an attribute set that can uniquely identify a tuple.
- A 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.

Foreign Key

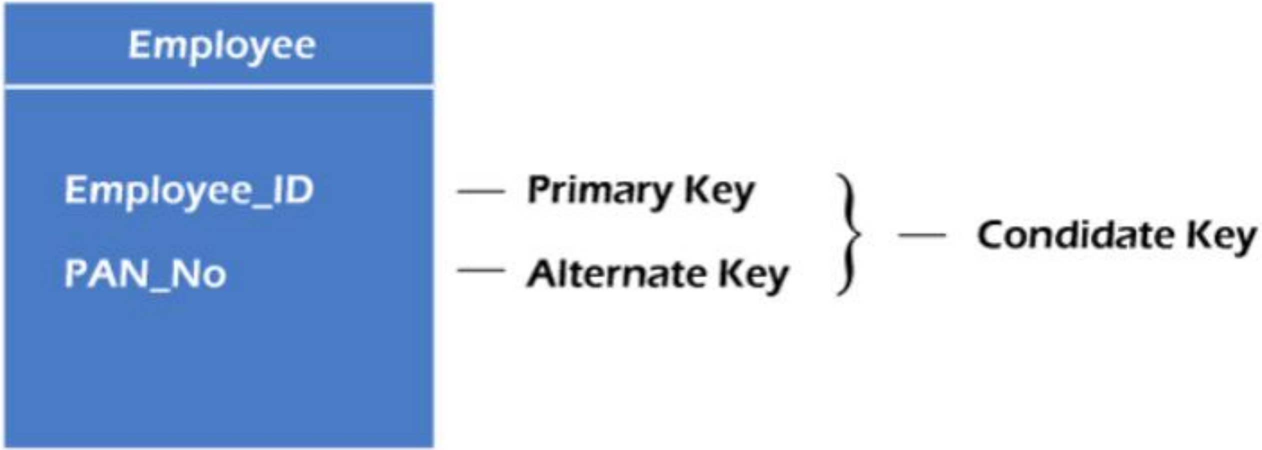
- Foreign keys are the column of the table used to point to the primary key of another table.



Alternate Key

- There may be one or more attributes or a combination of attributes that uniquely identify each tuple in a relation.
- These attributes or combinations of the attributes are called the candidate keys.
- One key is chosen as the primary key from these candidate keys, and the remaining candidate key, if it exists, is termed the **alternate key**.

Alternate Key- Example



Employee

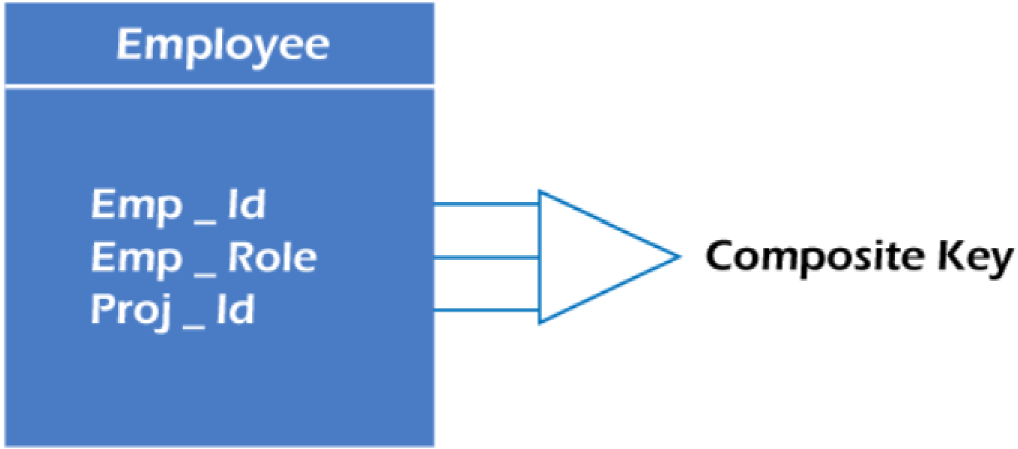
Employee_ID — Primary Key

PAN_No — Alternate Key

— Condidate Key

Composite Key

- Whenever a primary key consists of more than one attribute, it is known as a composite key. This key is also known as Concatenated Key.


Employee
Emp _ Id
Emp _ Role
Proj _ Id
Composite Key