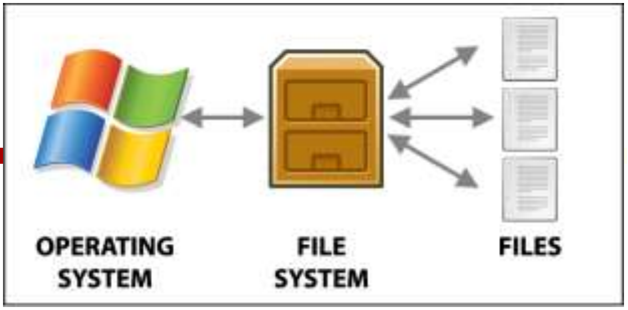
INTRODUCTION TO DBMS & THE E-R MODEL

Courtesy:
Ramakrishnan and Gehrke
Silberschatz, Korth and Sudarshan
Yong Choi, CSUB
Kabindra Koirala
Anjali Jivani

WHAT IS A DATABASE?

- A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS).
- Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.
- Data within the most common types of databases in operation today is typically modelled in rows and columns in a series of tables to make processing and data querying efficient.
- The data can then be easily accessed, managed, modified, updated, controlled, and organized.
- Most databases use structured query language (SQL) for writing and querying data.

THE FILE SYSTEMS



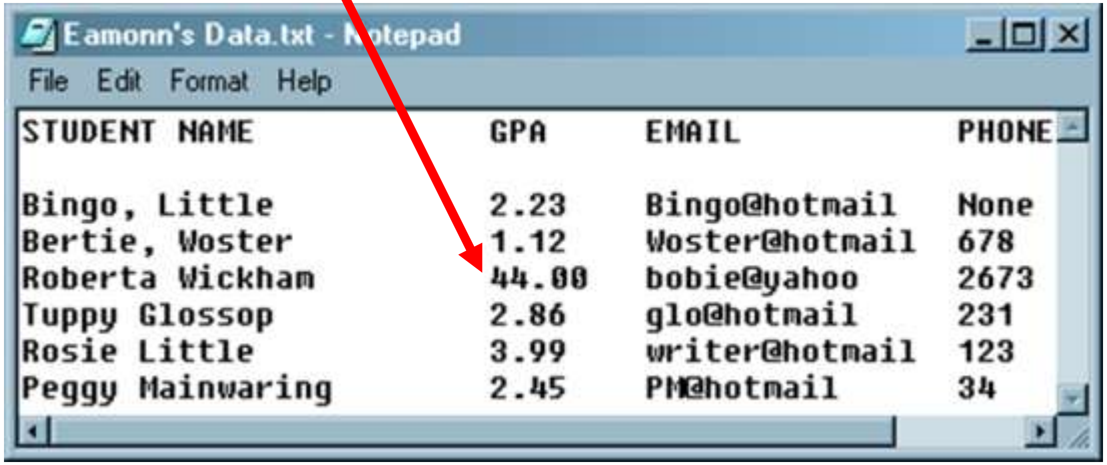
Drawbacks of the File System:

- Data redundancy and inconsistency
  - Multiple file formats, duplication of information in different files
- Difficulty in accessing data
  - Need to write a new program to carry out each new task
- Data isolation
  - Multiple files and formats
- Integrity problems
  - Integrity constraints (e.g., account balance > 0) become “buried” in program code rather than being stated explicitly
  - Hard to add new constraints or change existing ones

ENFORCING CONSTRAINTS

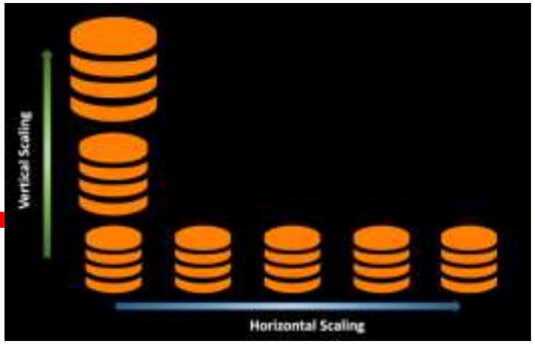
A DBMS allows us to enforce all kinds of constraints. This really helps (but does not guarantee) that our data is correct.

A typo gives Roberta Wickham a GPA of 44.00



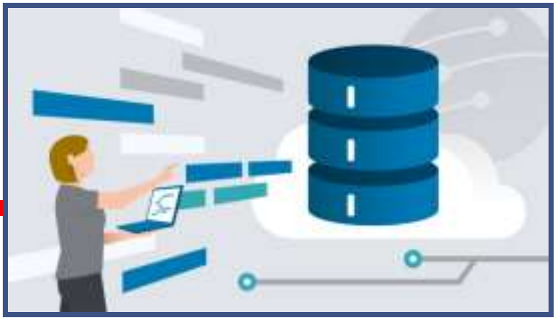
```
STUDENT NAME              GPA     EMAIL           PHONE

Bingo, Little             2.23    Bingo@hotmail   None
Bertie, Woster            1.12    Woster@hotmail  678
Roberta Wickham           44.00   bobie@yahoo     2673
Tuppy Glossop             2.86    glo@hotmail     231
Rosie Little              3.99    writer@hotmail  123
Peggy Mainwaring          2.45    PM@hotmail      34
```

SCALABILITY



- Most real world datasets are so large that we can only have a small fraction of them in main memory at any time, the rest has to stay on disk.
- Database scalability is the ability to scale out or scale up a database to allow it to hold increasing amounts of data without sacrificing performance.
- Scaling out often involves moving the database across multiple database servers in a distributed cluster while scaling up involves increasing the computing power and resources of the database server.

QUERY EXPRESSIVENESS

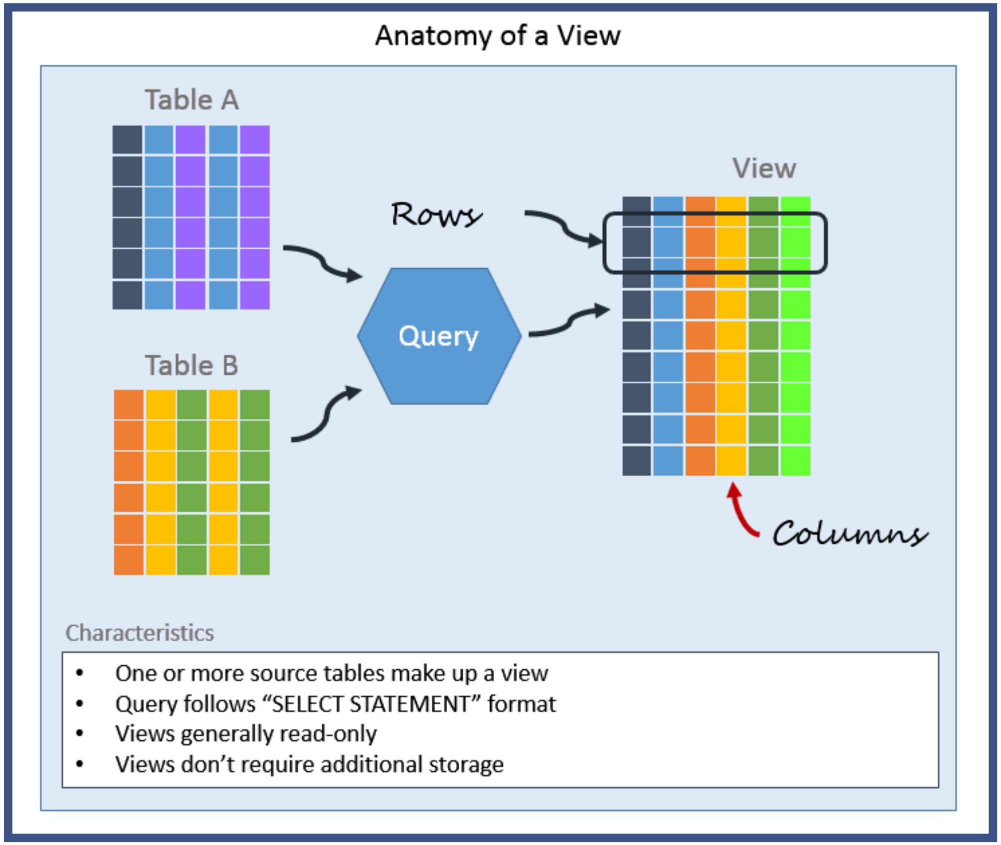


- Can execute complex queries which would not be possible with a flat file.

- Can write some program that might allow more expressive queries on the text file, but it would be tied into the structure of the data and the operating system etc..

- With a DBMS we are completely isolated from the physical structure of our data. If we change the structure of our data (by adding a field, for example) or moving from a PC to a Mac, nothing changes at the front end!

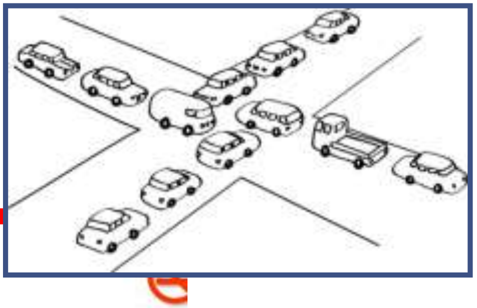
DIFFERENT VIEWS OF DATA

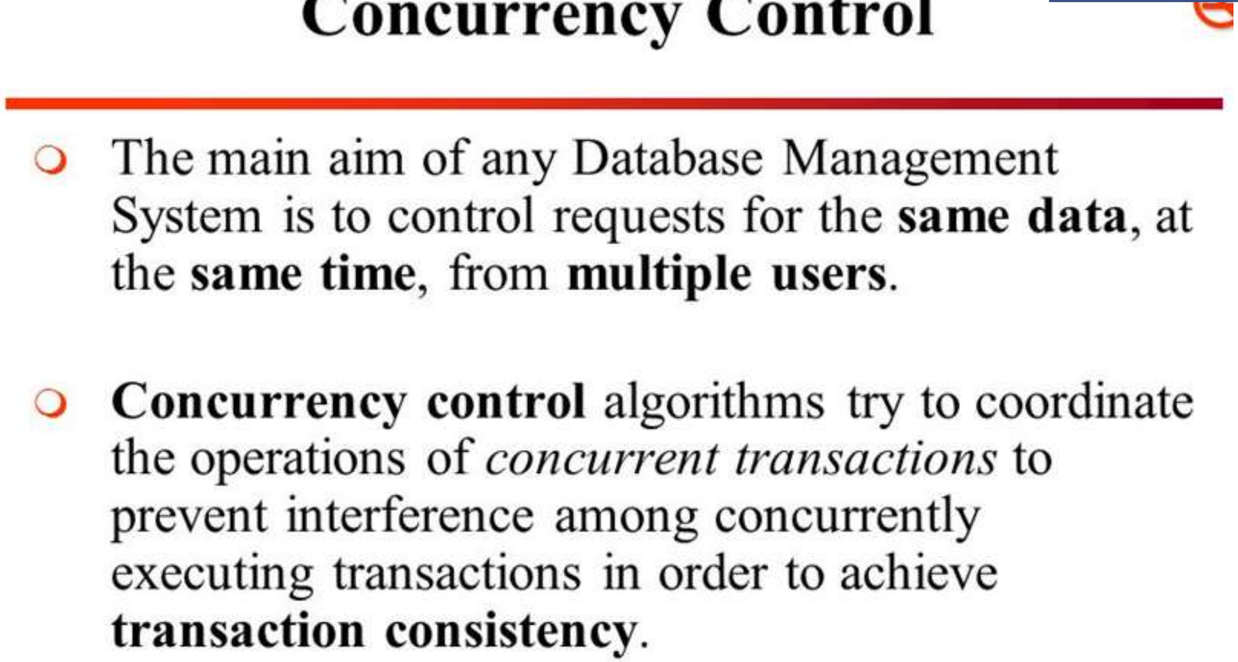


- One or more source tables make up a view
- Query follows "SELECT STATEMENT" format
- Views generally read-only
- Views don't require additional storage

With a DBMS we can arrange for different people to have different views of the data. For example, I can see everything, a student can see only his/her data, the clerk can see…

CONCURRENCY



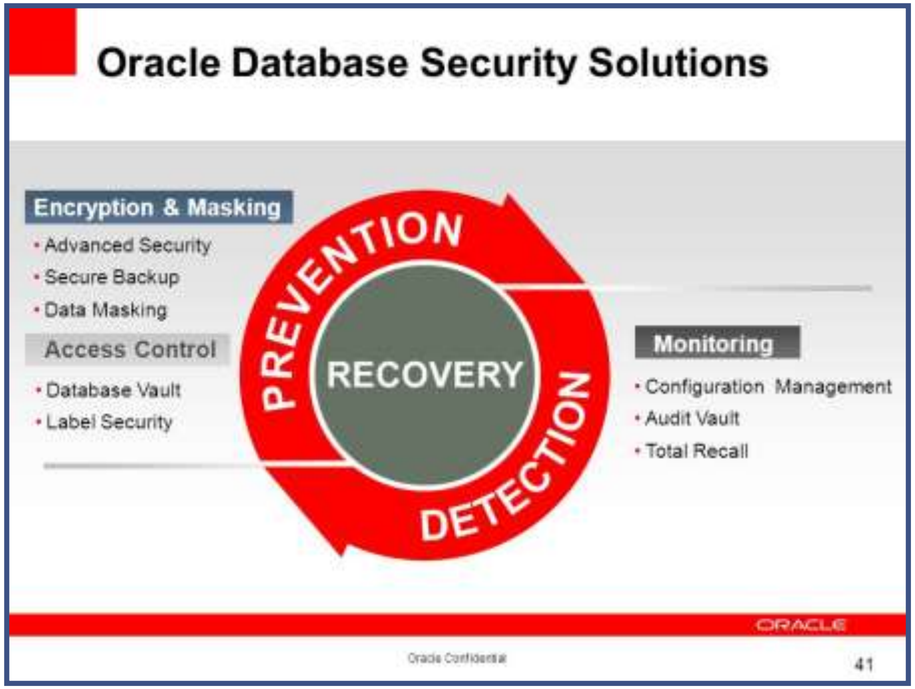


Concurrency Control

- The main aim of any Database Management System is to control requests for the **same data**, at the **same time**, from **multiple users**.
- **Concurrency control** algorithms try to coordinate the operations of *concurrent transactions* to prevent interference among concurrently executing transactions in order to achieve **transaction consistency**.

- Two people trying to modify the same data at the same time!
- A DBMS will automatically make sure that this kind of thing cannot happen.

SECURITY



Oracle Database Security Solutions

Encryption & Masking

- Advanced Security
- Secure Backup
- Data Masking

Access Control

- Database Vault
- Label Security

PREVENTION

RECOVERY

DETECTION

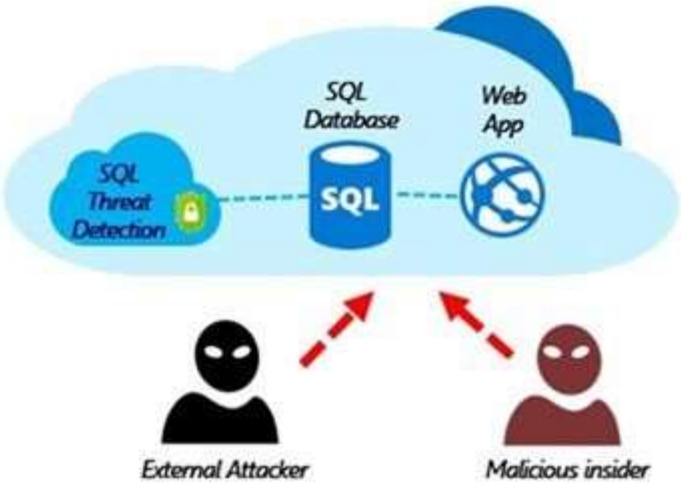
Monitoring

- Configuration Management
- Audit Vault
- Total Recall

ORACLE

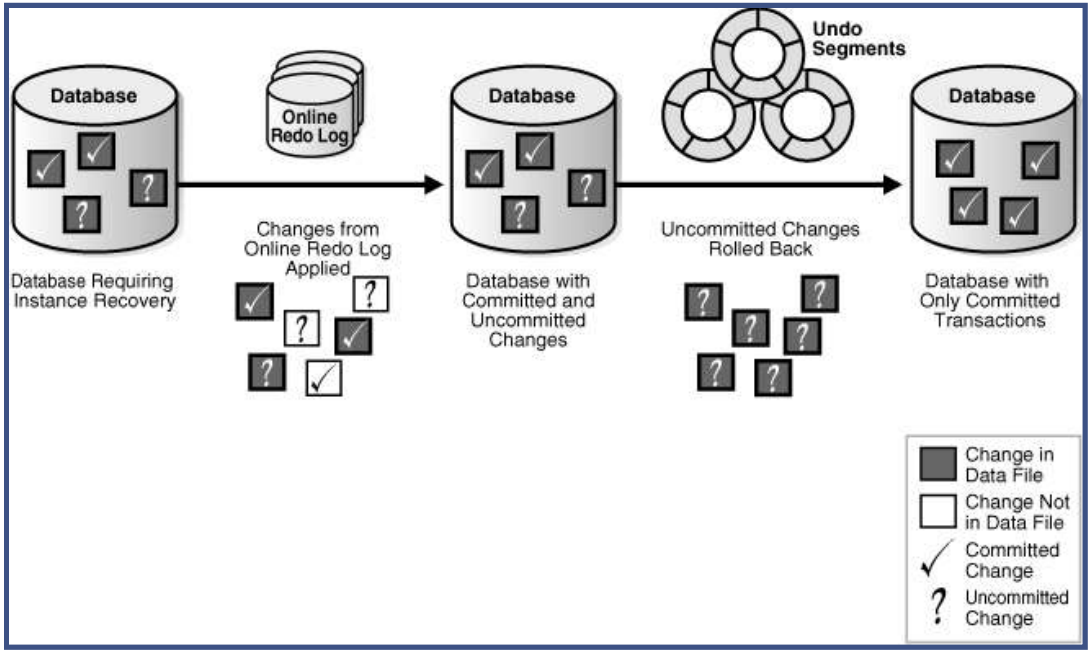
Oracle Confidential

41



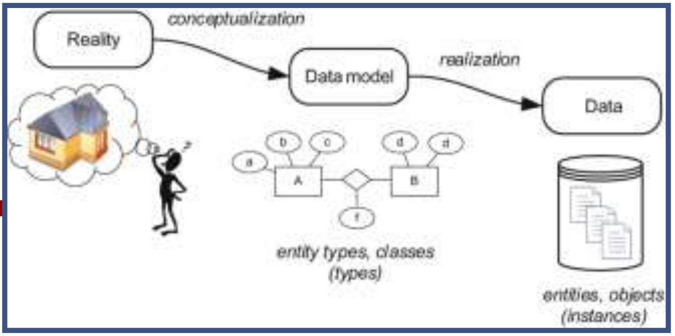
- ➢ Suppose I leave my text file in my user account, and a student hacks in and changes their grades…
- ➢ A DBMS will allow multiple levels of security.

CRASH RECOVERY


Database
Online Redo Log
Database
Undo Segments
Database
Database Requiring Instance Recovery
Changes from Online Redo Log Applied
Database with Committed and Uncommitted Changes
Uncommitted Changes Rolled Back
Database with Only Committed Transactions
Change in Data File
Change Not in Data File
Committed Change
Uncommitted Change

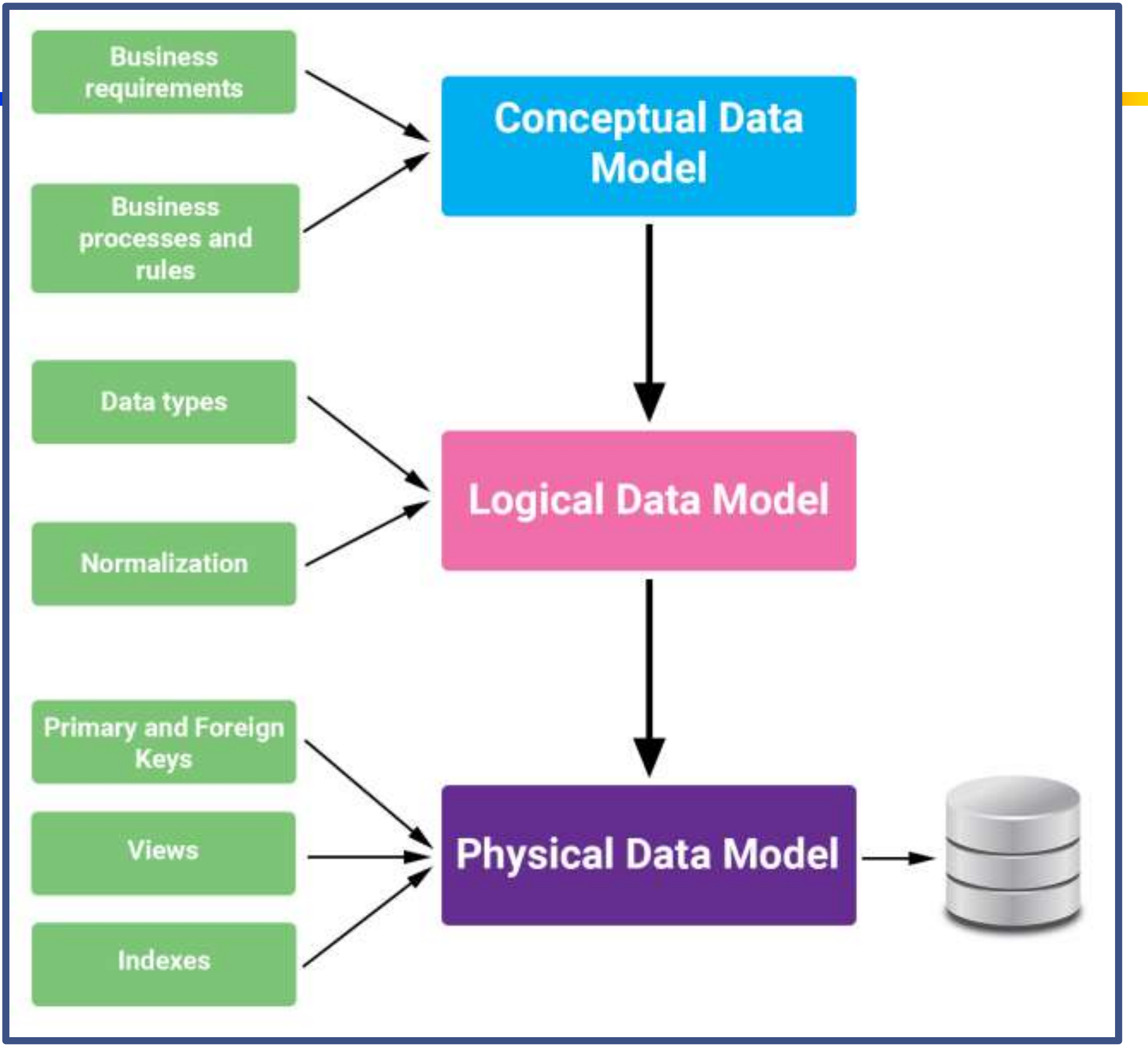
- Suppose you are editing a text file and the system crashes!
- A DBMS is able to guarantee 100% recovery from system crashes.
- **Crash recovery** is the process by which the database is moved back to a consistent and usable state.
- This is done by rolling back incomplete transactions and completing committed transactions that were still in memory when the **crash** occurred

DATA MODELS



What is Data Modelling?

- Data modelling is the process of creating a data model for the data to be stored in a database. This data model is a conceptual representation of Data objects, the associations between different data objects, and the rules.
- Data modelling helps in the visual representation of data and enforces business rules, regulatory compliances, and government policies on the data.
- Data Models ensure consistency in naming conventions, default values, semantics, security while ensuring quality of the data.
- Data Model is a diagram that displays a set of tables and the relationships between them.



Business requirements

Business processes and rules

Conceptual Data Model

Data types

Normalization

Logical Data Model

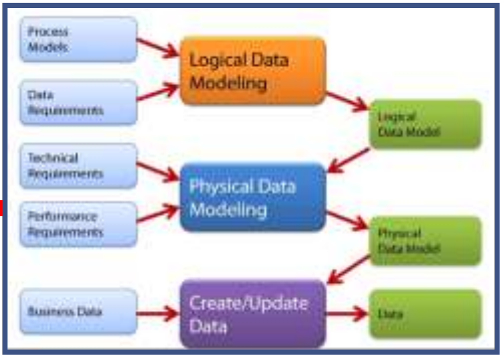
Primary and Foreign Keys

Views

Indexes

Physical Data Model

DATA MODELS

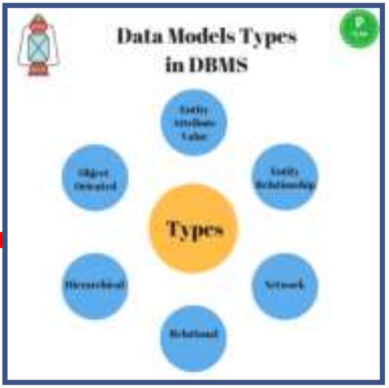


- **Conceptual Data Model:** This Data Model defines **WHAT** the system contains. This model is typically created by Business stakeholders and Data Architects. The purpose is to organize, scope and define business concepts and rules.

- **Logical Data Model:** Defines **HOW** the system should be implemented regardless of the DBMS. This model is typically created by Data Architects and Business Analysts. The purpose is to developed technical map of rules and data structures.

- **Physical Data Model**: This Data Model describes **HOW** the system will be implemented using a specific DBMS system. This model is typically created by DBA and developers. The purpose is actual implementation of the database.

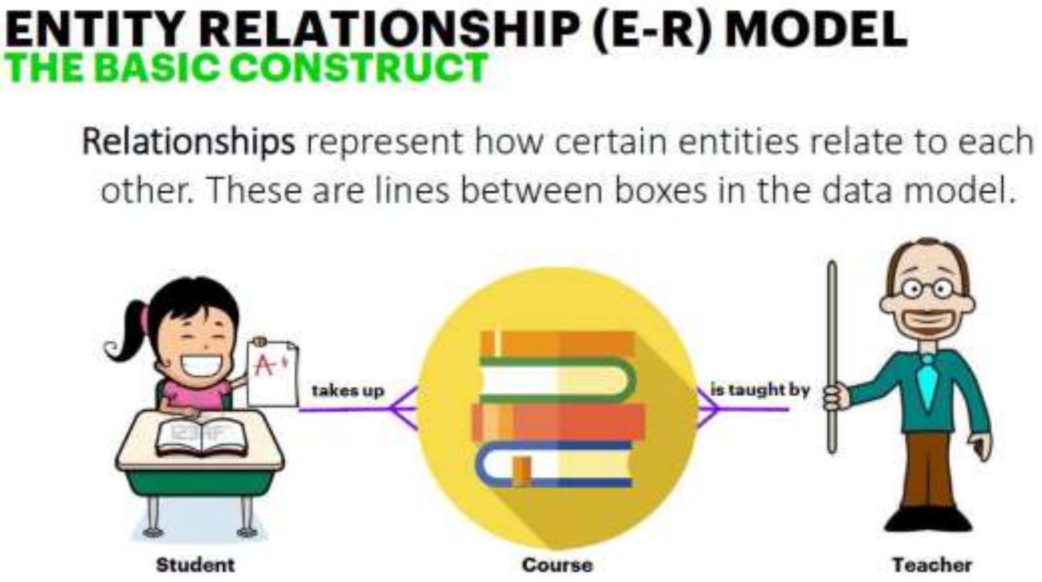
TYPES OF DATA MODELS



There are many kinds of data models:

- File based model
- Entity-relationship model
- Relational model
- Hierarchical model
- Network model
- Object-Oriented model
- Object-Relational model
- Document model
- Star schema
- …

THE E-R MODEL



ENTITY RELATIONSHIP (E-R) MODEL
THE BASIC CONSTRUCT

Relationships represent how certain entities relate to each other. These are lines between boxes in the data model.

Designed by Peter Chen. Entity-Relationship data model views the real world as a set of basic **objects** (entities) and **relationships** among these objects.

- An **E-R model** describes the structure of a database with the help of a **diagram**, which is known as **Entity Relationship Diagram** (**E-R Diagram**).

- An **E-R model** is a design or blueprint of a database that can later be implemented as a database.

BASIC CONCEPTS

- The ER model defines the three most relevant steps. It works around real-world entities and the associations among them. At view level, the ER model is considered a good option for designing databases.
  - Requirement Analysis
  - Conceptual Database Design
  - Logical Database Design

Requirement Analysis:

The very first step in designing a database application is to understand what data is to be stored in the database, what applications must be built on the top of it, and what operations the users want from the database.

BASIC CONCEPTS

Conceptual Database Design:

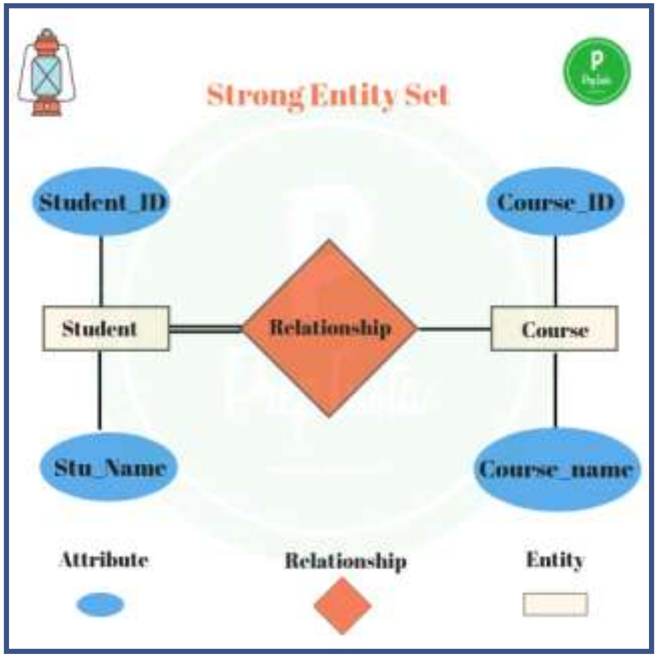
The information gathered in the requirements analysis step is used to develop a high-level description of the data to be stored in database, along with the constraints known to hold over the data. The ER model is one of the high-level or semantic, data models used in database.

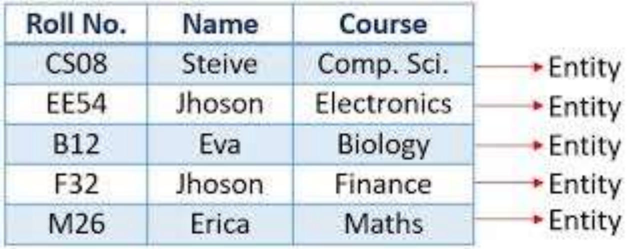
Logical Database Design:

We must choose a DBMS to implement our database design, and convert the conceptual database design into a database schema in the data model of chosen DBMS. Sometimes conceptual schema is called logical schema in Relational Data Model.

ENTITIES AND ENTITY SETS

- An **entity** is a tangible or non-tangible object that exists and is distinguishable from other objects and about which we can store information.
  Example: person, company, event, plant, account,…
- Entities have **attributes**
  Example: people have *names* and *addresses*
- An **entity set** is a set of entities of the same type that share the same properties.
  Example: set of all persons, companies, trees, holidays

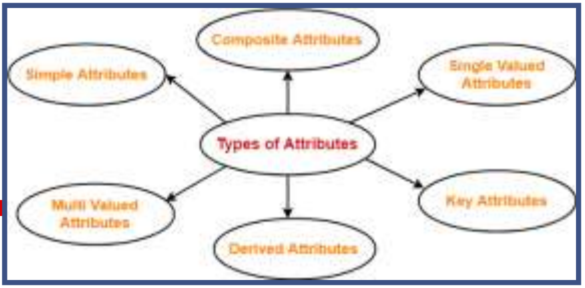




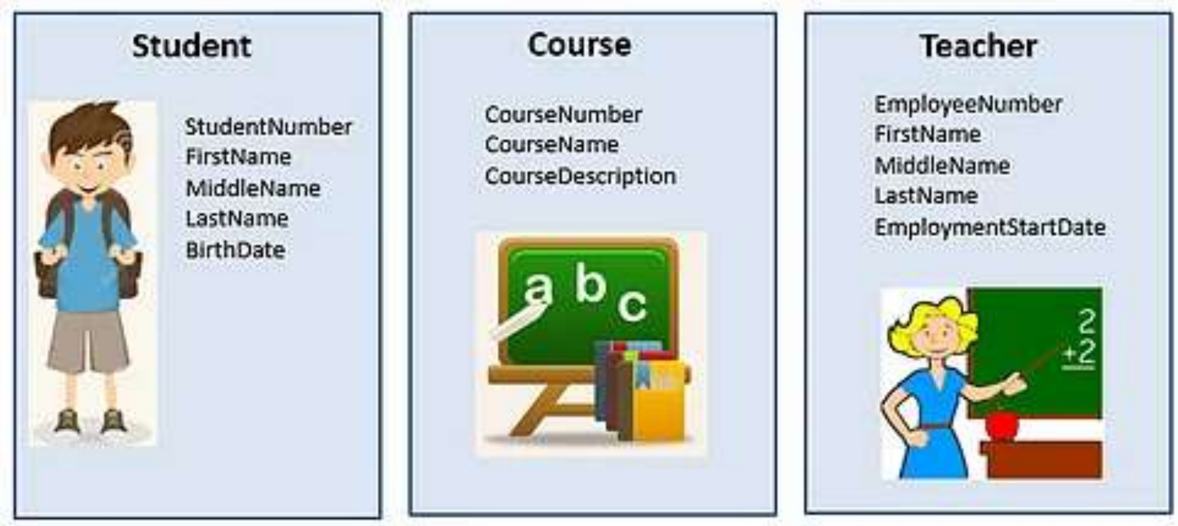
Roll No.	Name	Course	
CS08	Steive	Comp. Sci.	Entity
EE54	Jhoson	Electronics	Entity
B12	Eva	Biology	Entity
F32	Jhoson	Finance	Entity
M26	Erica	Maths	Entity

Student Table

ATTRIBUTES



- An entity is represented by a set of attributes, that is descriptive properties possessed by all members of an entity set.
- **Domain** – the set of permitted values for each attribute
- Attribute types:
  - **Simple** and **composite** attributes (Studentnumber / Name (FN,MN,LN))
  - **Single-valued** and **multivalued** attributes (Passportno / Phoneno)
  - **Derived** attributes (Age from birthdate)



KEYS

- Key is an attribute or collection of attributes that uniquely identifies an entity among entity set. For example, the roll_number of a student makes him/her identifiable among students.
- There are mainly three types of keys:
  - Super Key - A set of attributes (one or more) that collectively identifies an entity in an entity set.
  - Candidate Key - A minimal super key is called a candidate key. An entity set may have more than one candidate key.
  - Primary Key - A primary key is one of the candidate keys chosen by the database designer to uniquely identify the entity set.

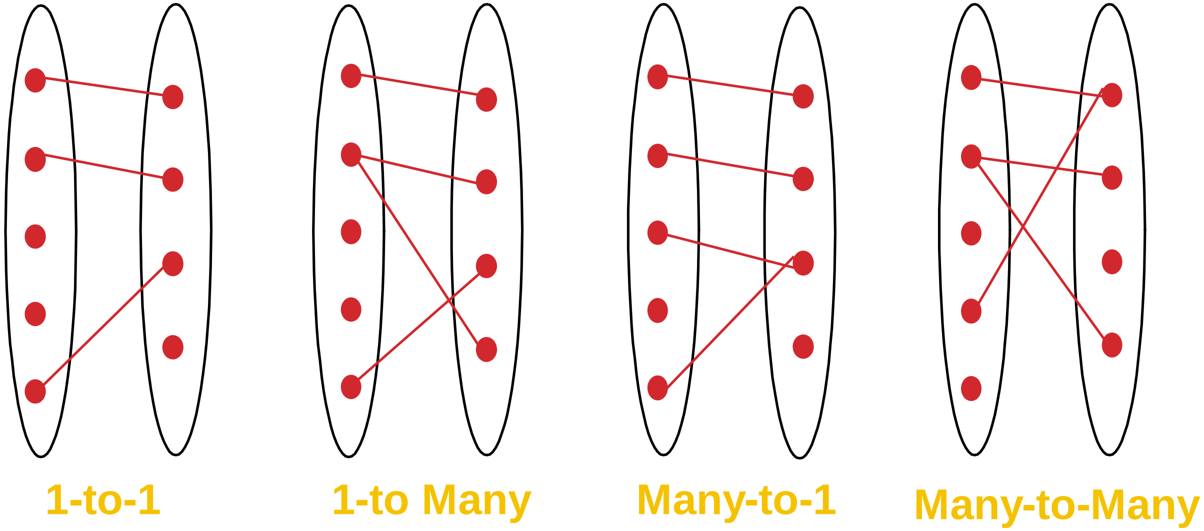
RELATIONSHIP

- The association among entities is called a relationship. For example, an employee works_at a department, a student enrolls in a course. Here, Works_at and Enrolls are called relationships.

- Relationship Set: A set of relationships of similar type is called a relationship set. Like entities, a relationship too can have attributes. These attributes are called descriptive attributes.

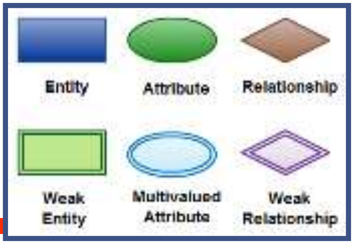
- Degree of Relationship: The number of participating entities in a relationship defines the degree of the relationship.
  - Binary = degree 2
  - Ternary = degree 3
  - n-ary = degree n

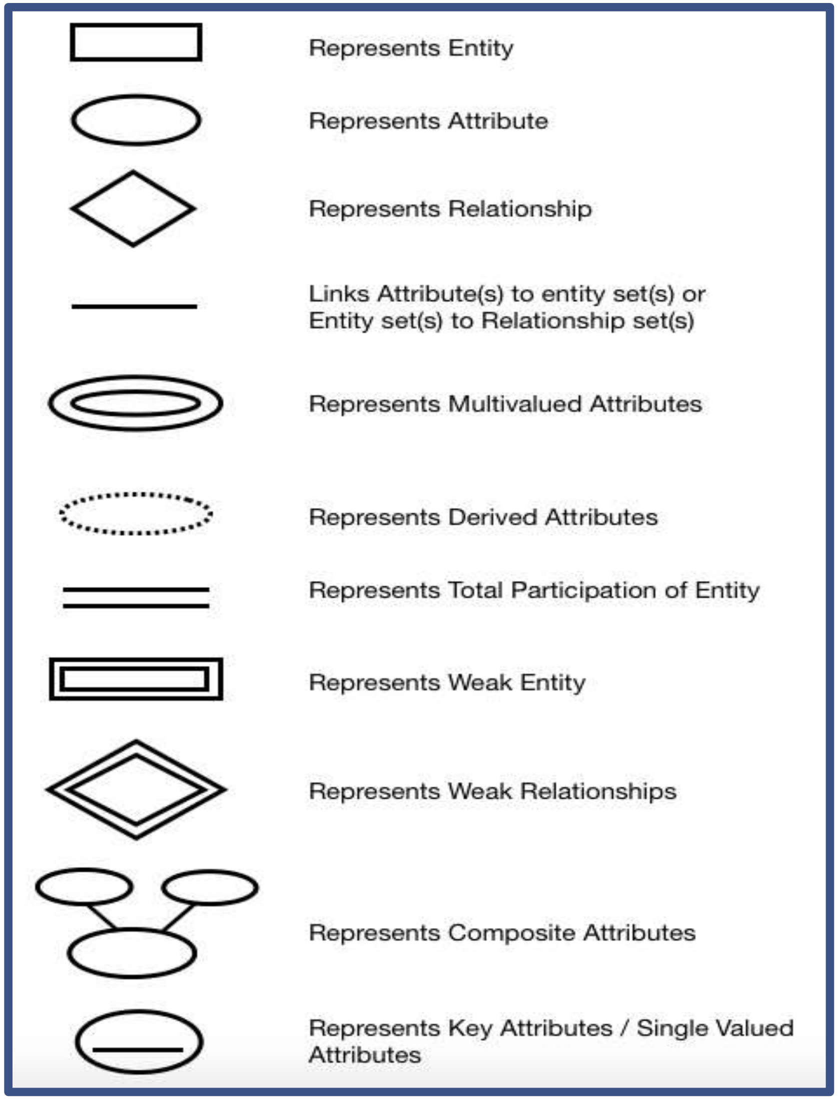
MAPPING CARDINALITY



- Types of Relationships (**Mapping Cardinality / Key Constraints**)
  - One-to-one relationship (1:1): One instance in an entity (parent) refers to one and only one instance in the related entity (child).
  - One-to-many relationship (1:M): One instance in an entity (parent) refers to one or more instances in the related entity (child)
  - Many-to-one relationship (M:1): One or more instances in an entity (parent) refers to one instance in the related entity (child)
  - Many-to-many relationship (M:M): One or more instances in an entity (parent) refers to one or more instances in the related entity (child)

E-R DIAGRAM





Represents Entity

Represents Attribute

Represents Relationship

Links Attribute(s) to entity set(s) or
Entity set(s) to Relationship set(s)

Represents Multivalued Attributes

Represents Derived Attributes

Represents Total Participation of Entity

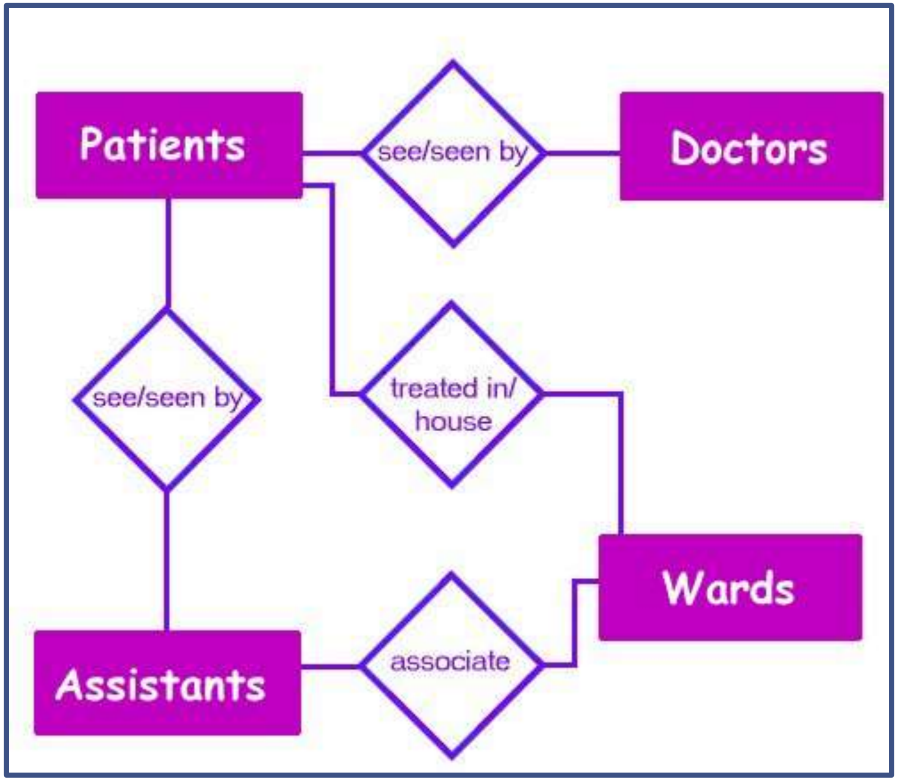
Represents Weak Entity

Represents Weak Relationships

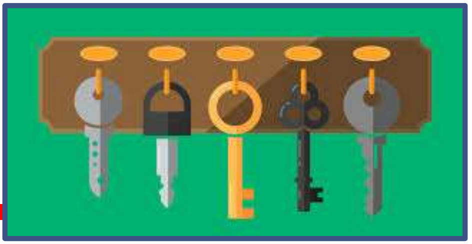
Represents Composite Attributes

Represents Key Attributes / Single Valued Attributes

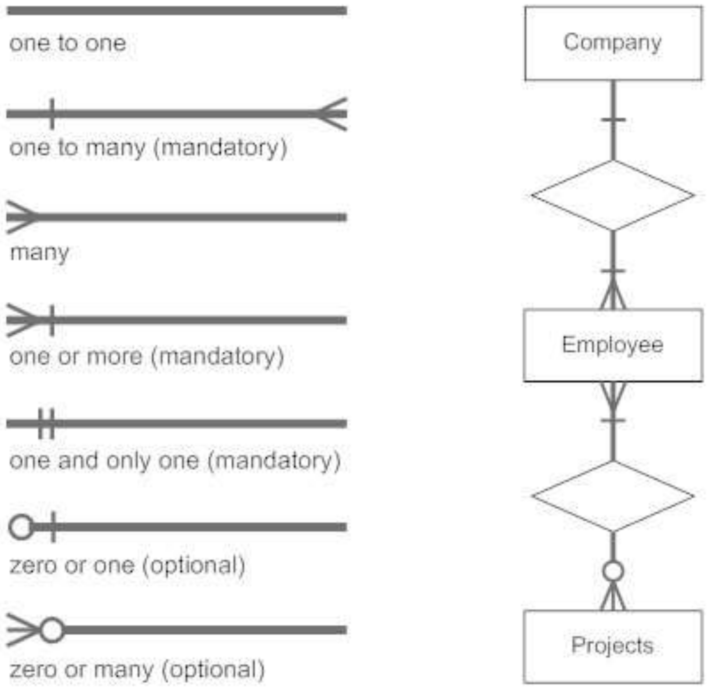
Entity names: Nouns
Relationship names: Verbs

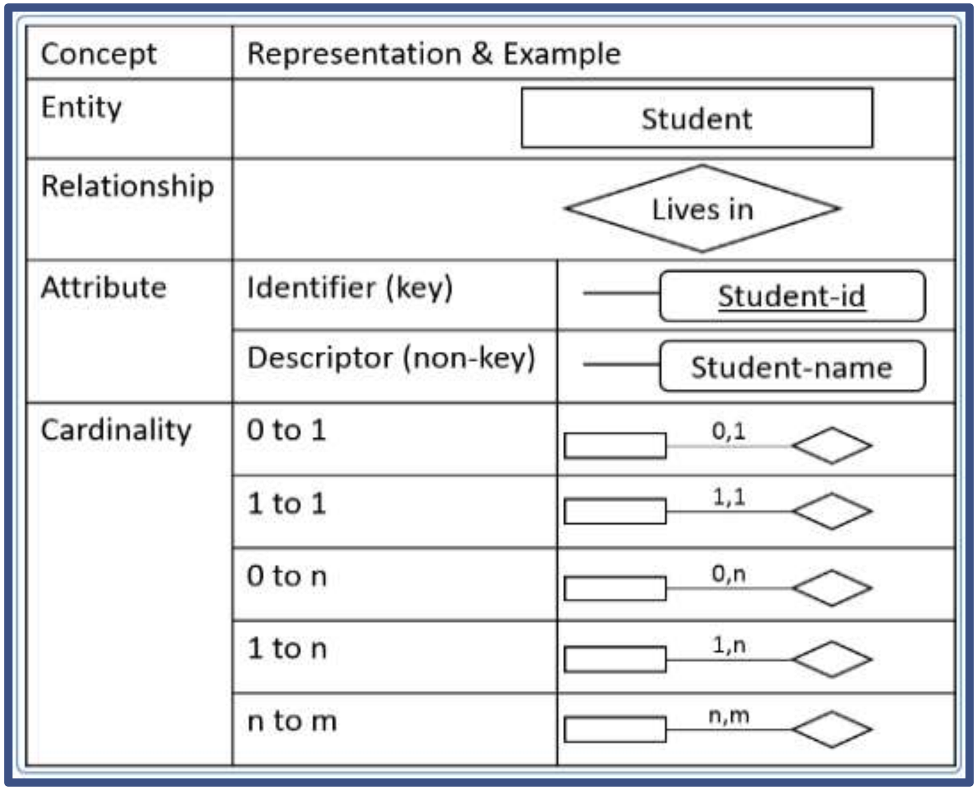


KEY CONSTRAINTS



Information Engineering Style

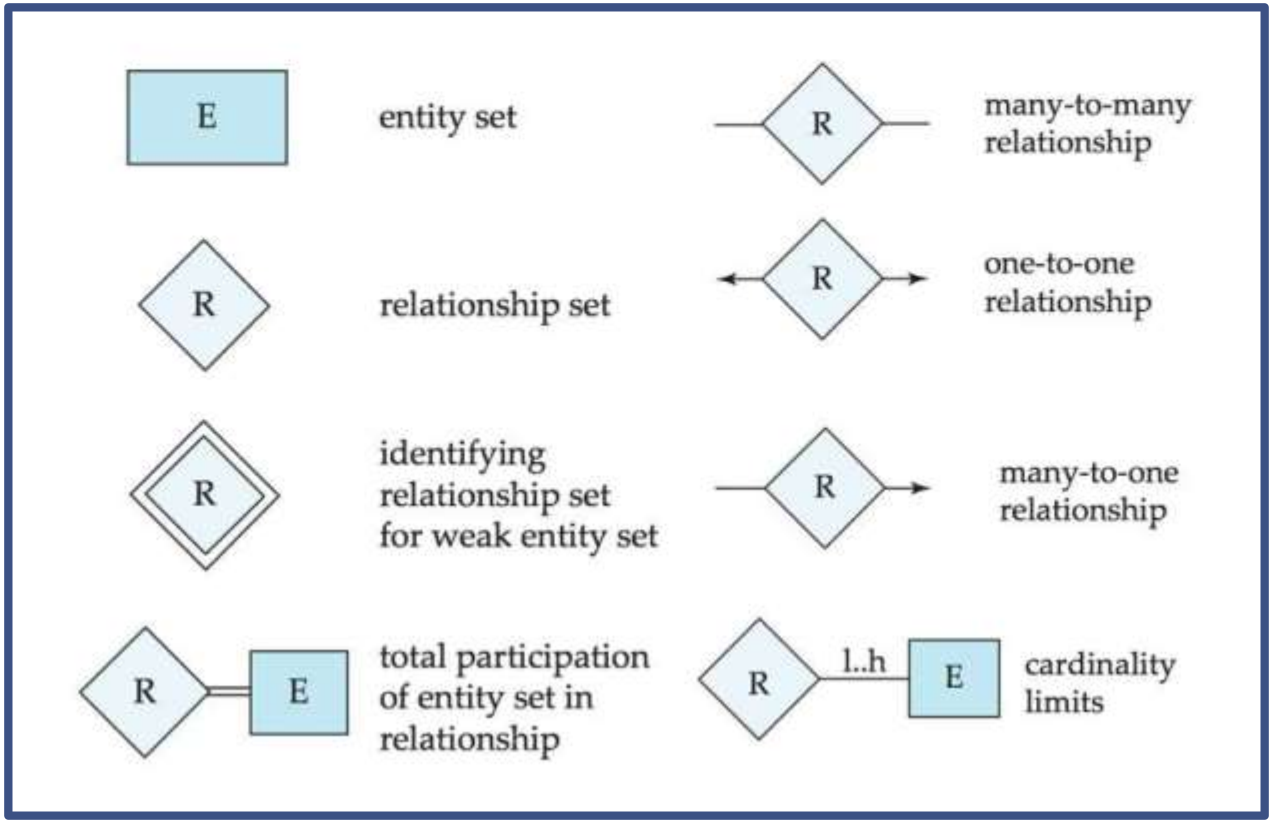




Concept	Representation & Example	
Entity	Student	
Relationship	Lives in	
Attribute	Identifier (key)	Student-id
	Descriptor (non-key)	Student-name
Cardinality	0 to 1	0,1
	1 to 1	1,1
	0 to n	0,n
	1 to n	1,n
	n to m	n,m

KEY CONSTRAINTS

Other popular symbols:



entity set

relationship set

identifying relationship set for weak entity set

total participation of entity set in relationship

many-to-many relationship

one-to-one relationship

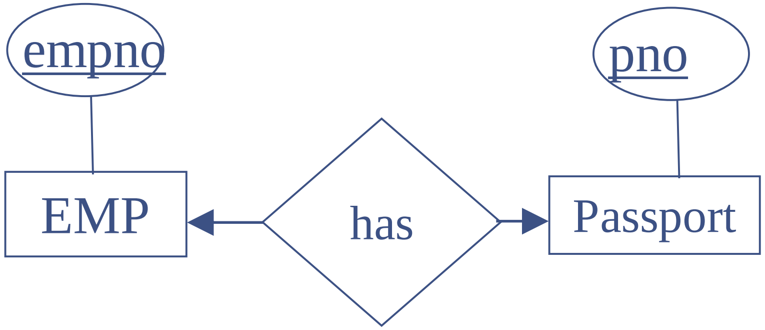
many-to-one relationship

cardinality limits

KEY CONSTRAINTS

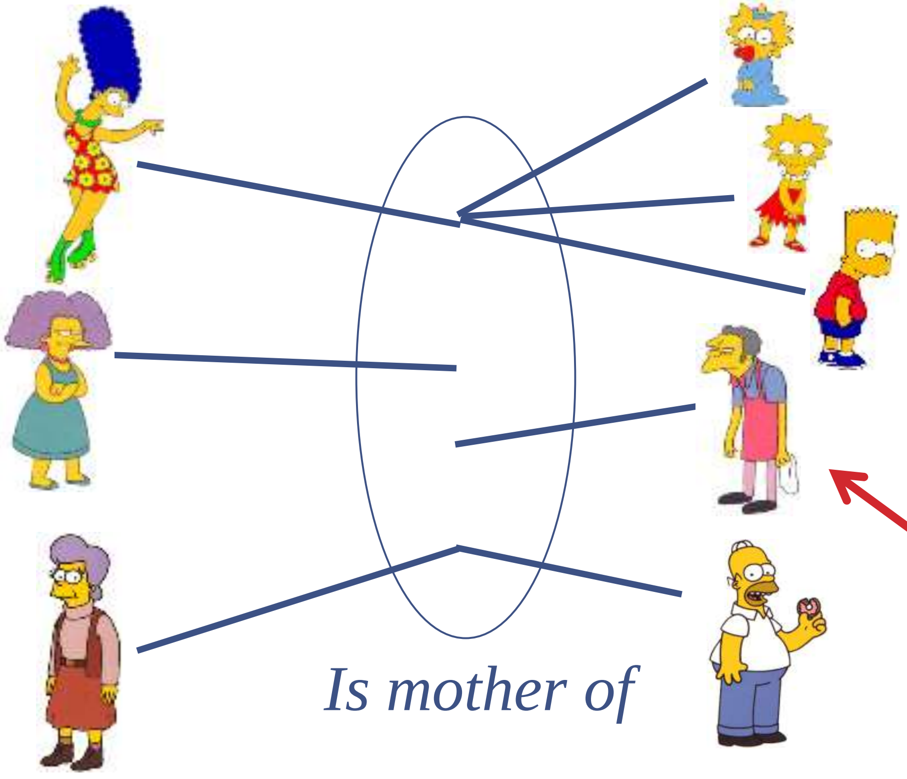
- **One-to-one**: An entity in A is associated with at most one entity in B, and an entity in B is associated with at most one entity in A.

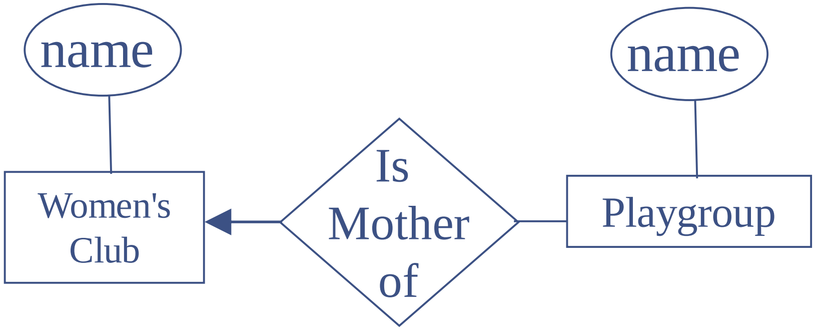
ONE-TO-ONE RELATIONSHIP
PASSPORT
one passport
one person



KEY CONSTRAINTS

- **One-to-many**: An entity in A is associated with any number in B. An entity in B is associated with at most one entity in A.
- A woman from a club may be the mother of many (or no) children. A person may have at most one mother e.g. from set of women registered in a club and set of children in a playgroup

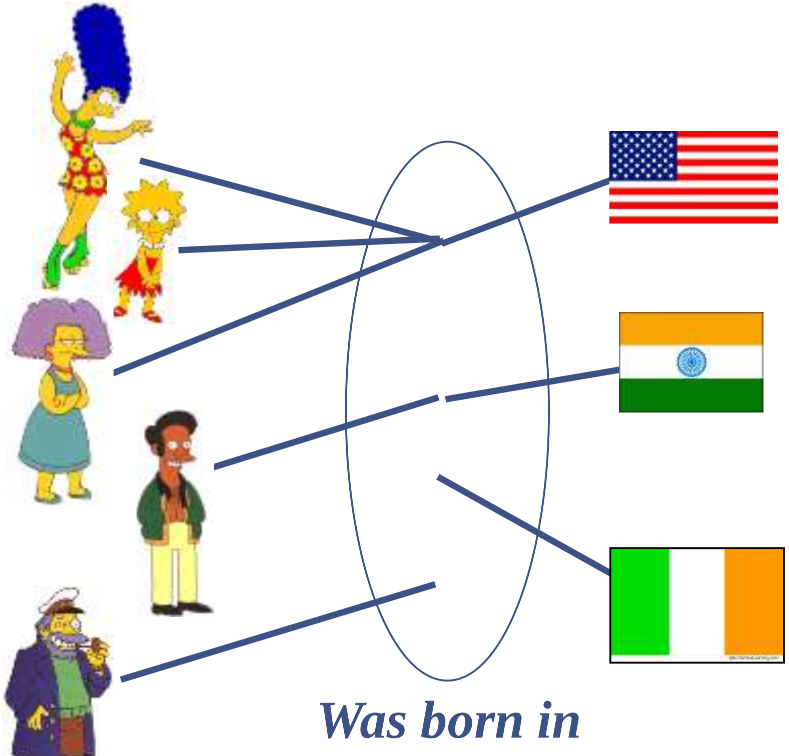


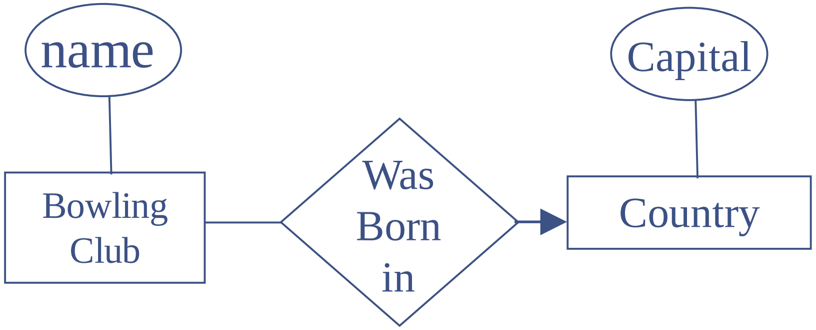


Note that this example is not saying that Moe does not have a mother, since we know as a biological fact that everyone has a mother.
It is simply the case that Moe's mom is not a member of the Women's club.

KEY CONSTRAINTS

- **Many-to-one**: An entity in A is associated with at most one entity in B. An entity in B is associated with any number in A.
- Many people are be born in one county, but any individual is born in at most one country e.g. an entity set of people registered in a Bowling Club and an entity set of countries.

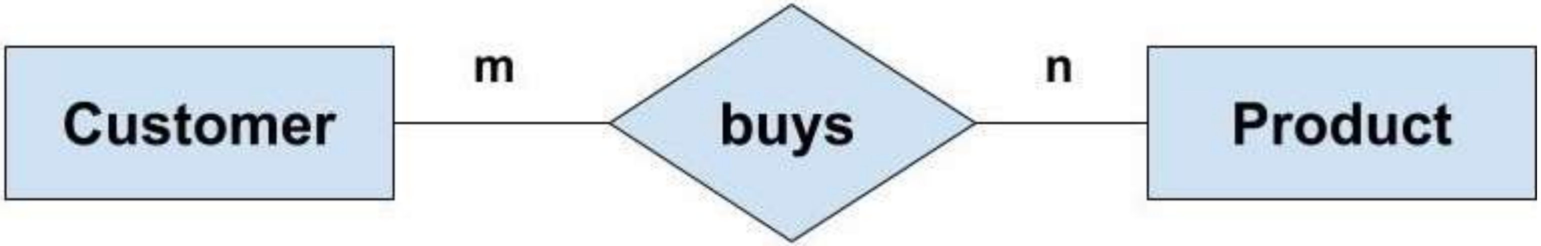


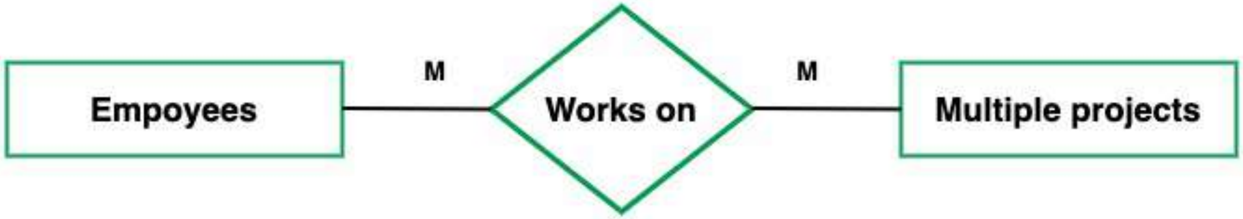


Note that we are not saying that the Sea Captain was not born in some country, he almost certainly was, we just don't know which country, or it is not in our Country entity set.
Also note that we are not saying that no one was born in Ireland, it is just that no one in the Bowling Club was.

KEY CONSTRAINTS

- **Many-to-many**: Entities in A and B are associated with any number from each other.

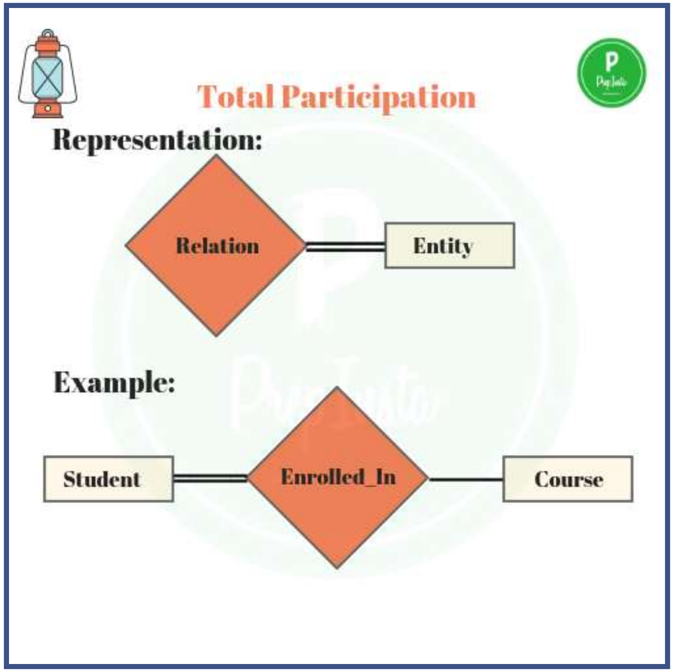


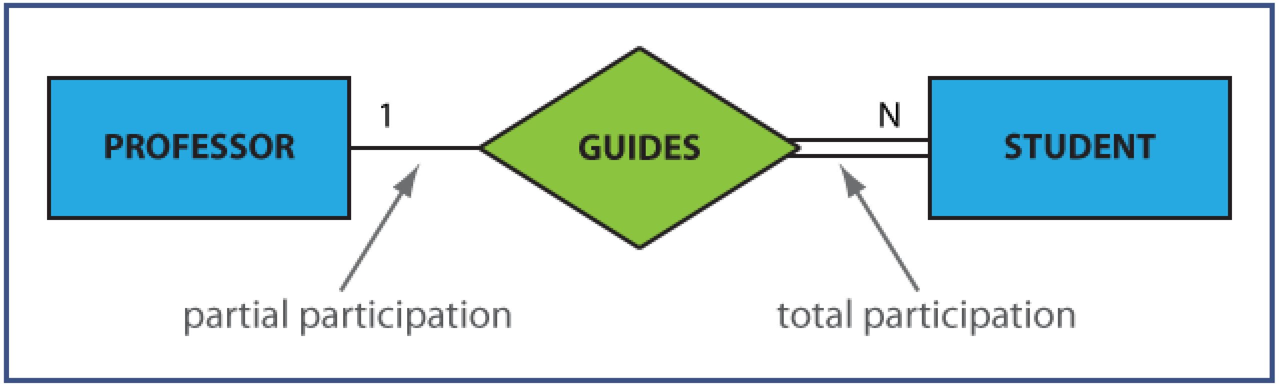


PARTICIPATION CONSTRAINTS

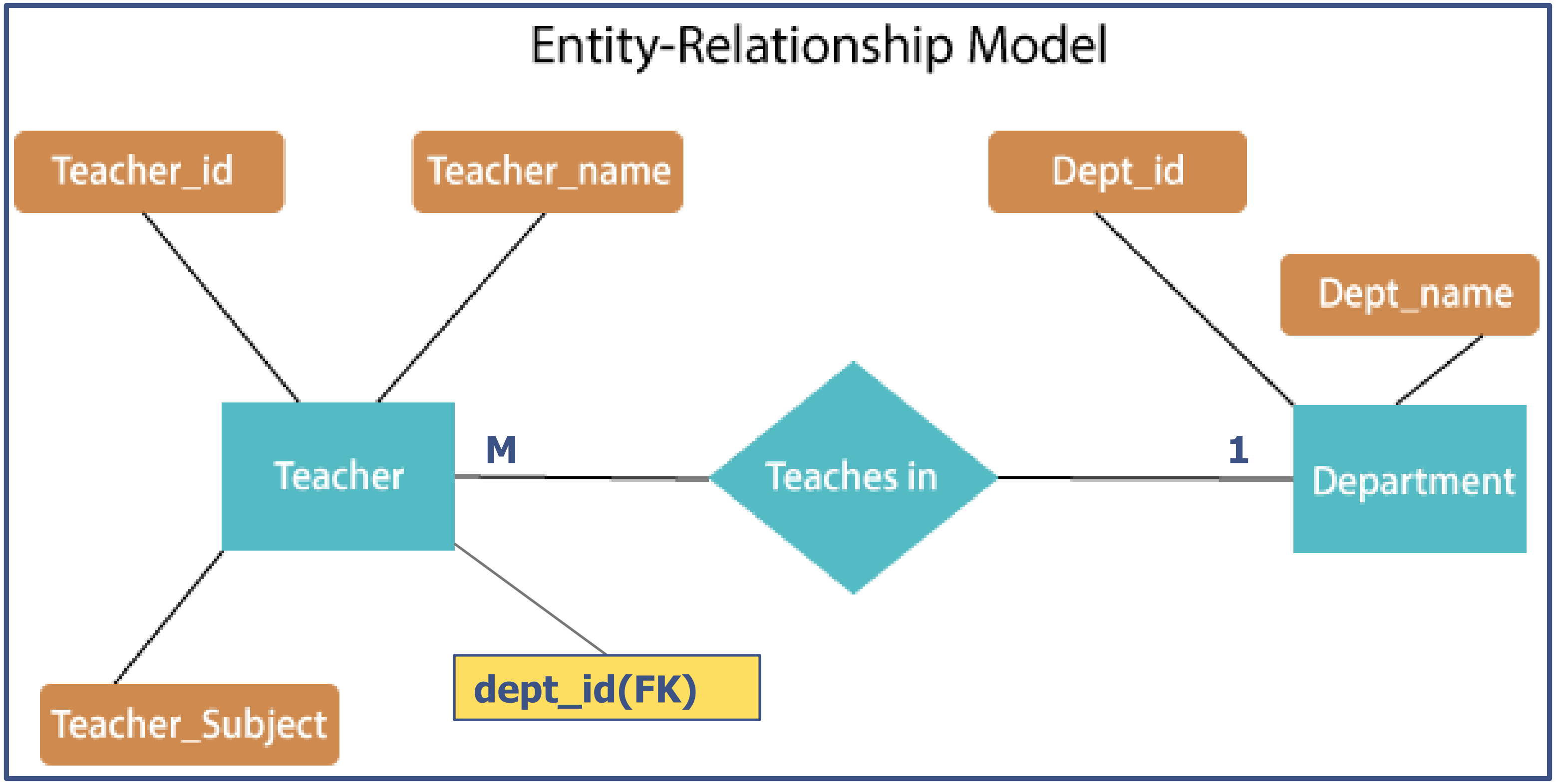
- **Total Participation** means each entity is involved in the relationship.
- **Partial participation** means not all entities are involved in the relationship.
- Total participation is shown in the form of bold line or double lines.

Partial participation is represented by single lines.





E-R DIAGRAMS



Entity-Relationship Model

Teacher_id

Teacher_name

Dept_id

Dept_name

Teacher

M

Teaches in

1

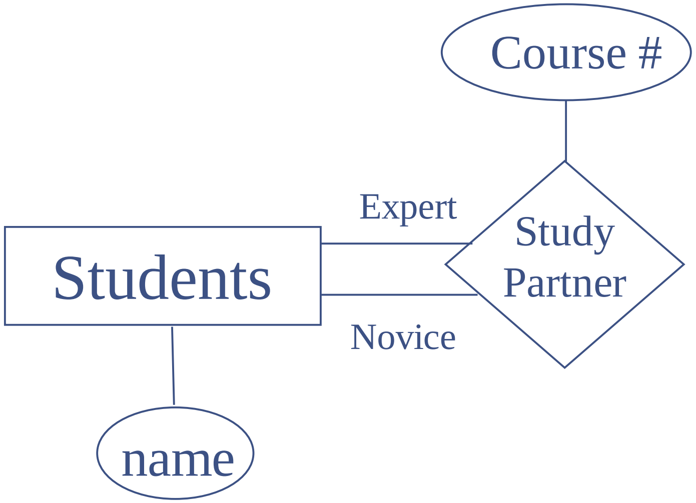
Department

Teacher_Subject

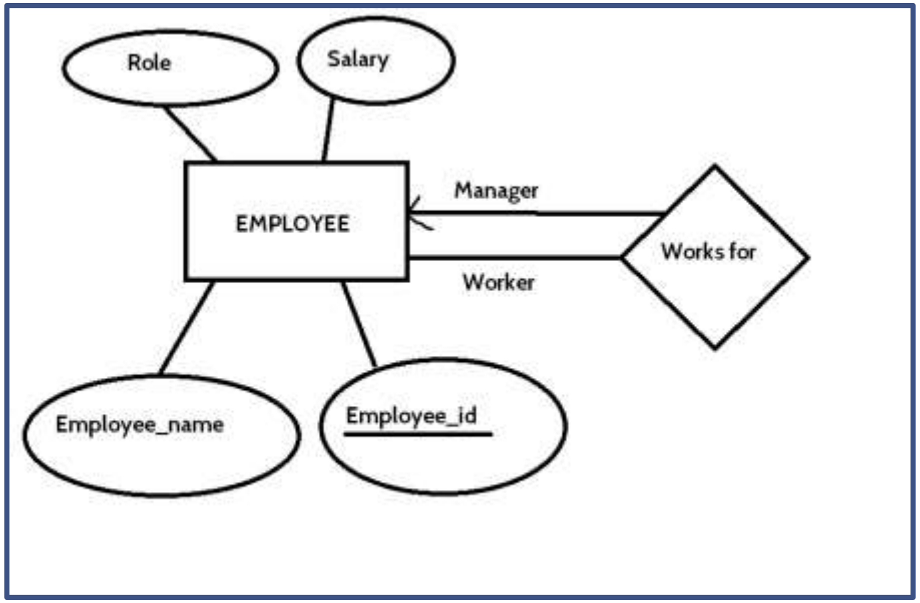
dept_id(FK)

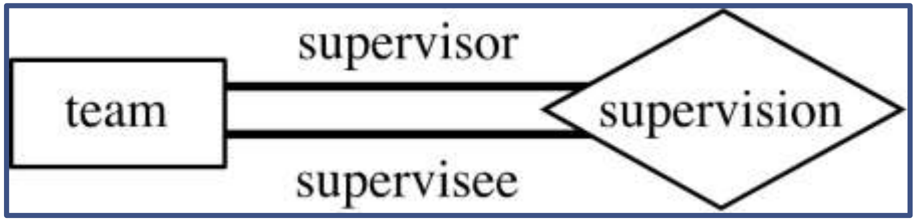
UNARY RELATION

A student taking a course could be a novice or expert.



A **unary relationship** is when both participants in the **relationship** are the same entity.
For Example: A team member is a supervisor for other team members.
Two connections with relation.



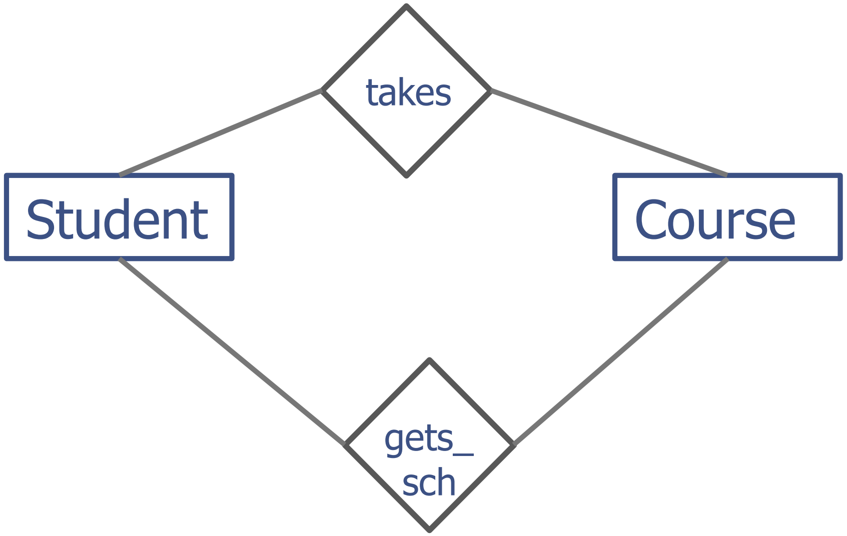


MULTIPLE RELATIONS

- Draw an E-R diagram to depict details about students studying/taking different courses and some students getting scholarship on some of the courses.

- In each case there are a set of their own attributes.

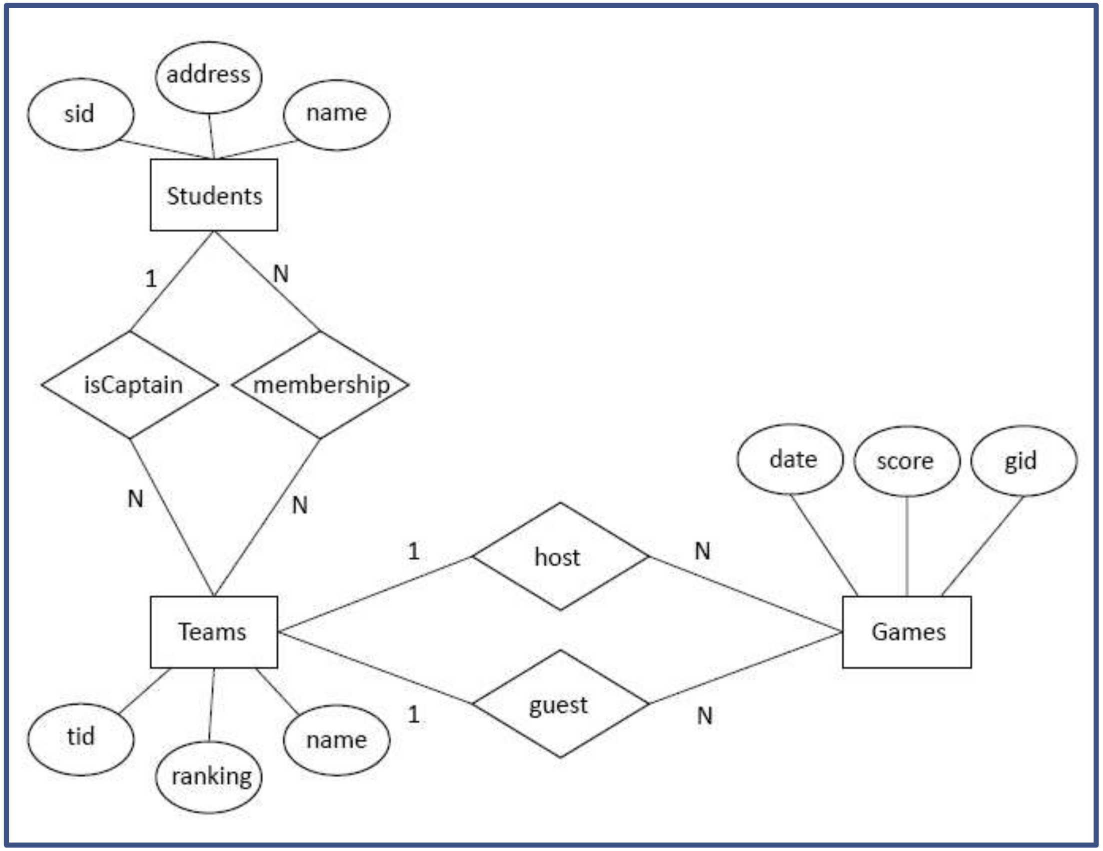
- There can be more than one relationship between entities.



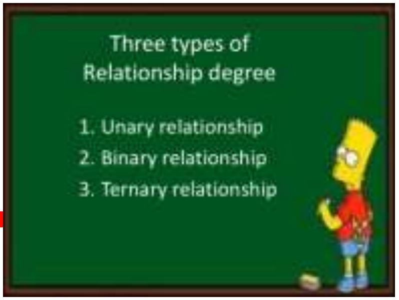
MULTIPLE RELATIONS

- Relationship:
  - A relationship is defined as an association among Entities.

- ER diagram to depict students who are members of Teams and some students are captains.
- Each have their own attributes and that is why separate relationships.
- The teams can be the hosting teams or the guest teams.



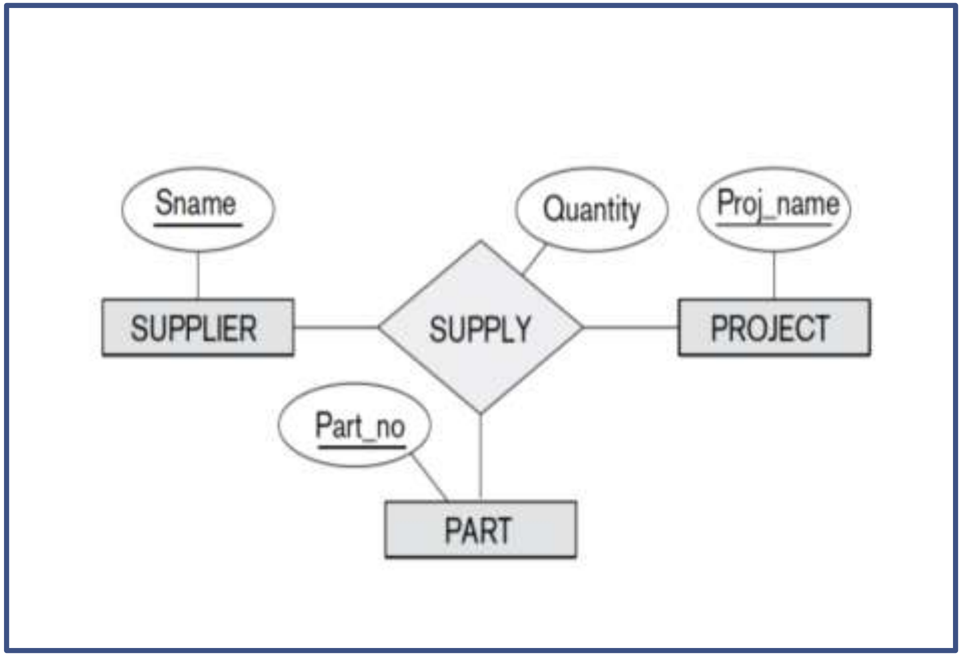
TERNARY RELATIONS

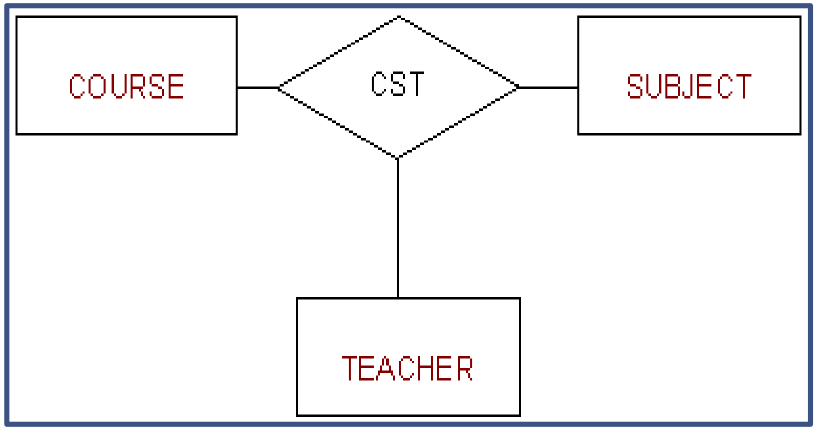


Three types of Relationship degree

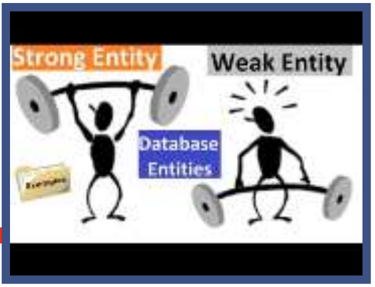
1. Unary relationship
2. Binary relationship
3. Ternary relationship

- Draw an E-R diagram for an application which stores details about parts being supplied to different projects by different suppliers.
- A **ternary relationship** is when three entities participate in the **relationship**.
- In general, unless you really need a ternary relationship, use binary relationships.

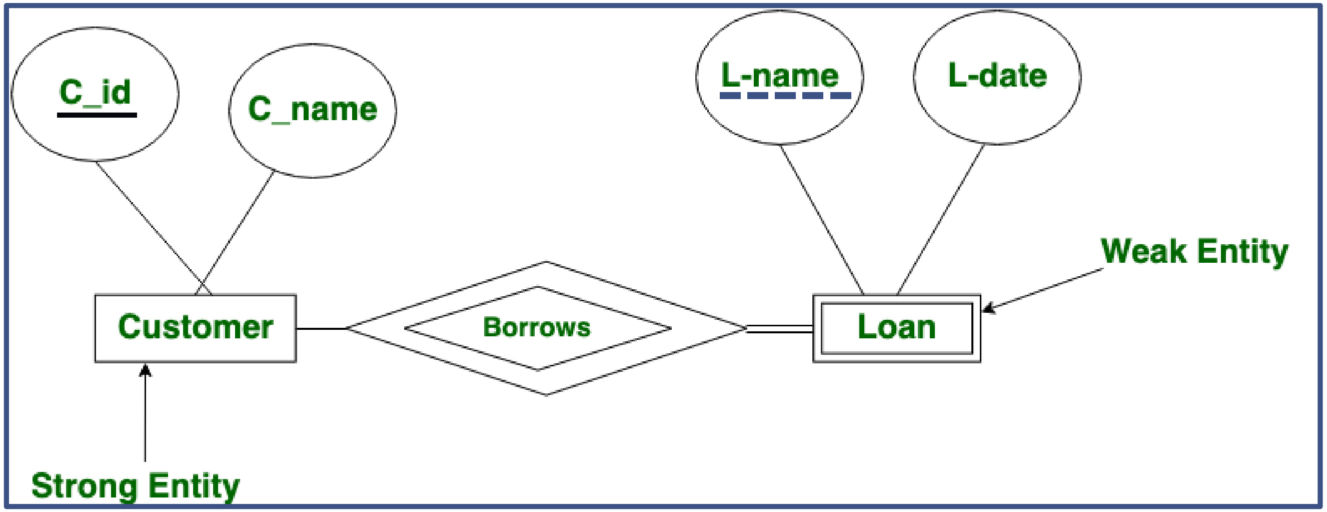




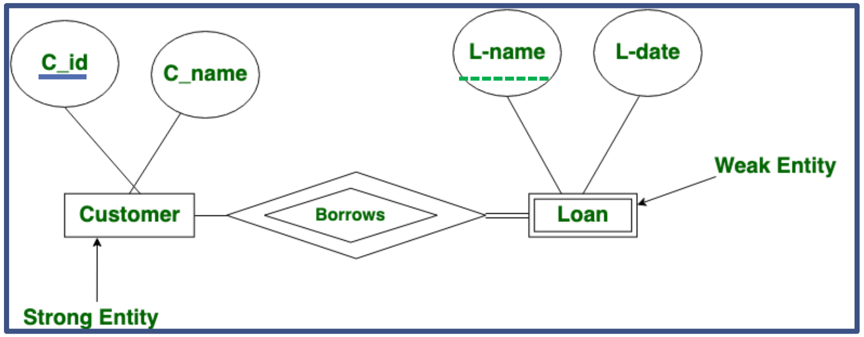
WEAK ENTITIES



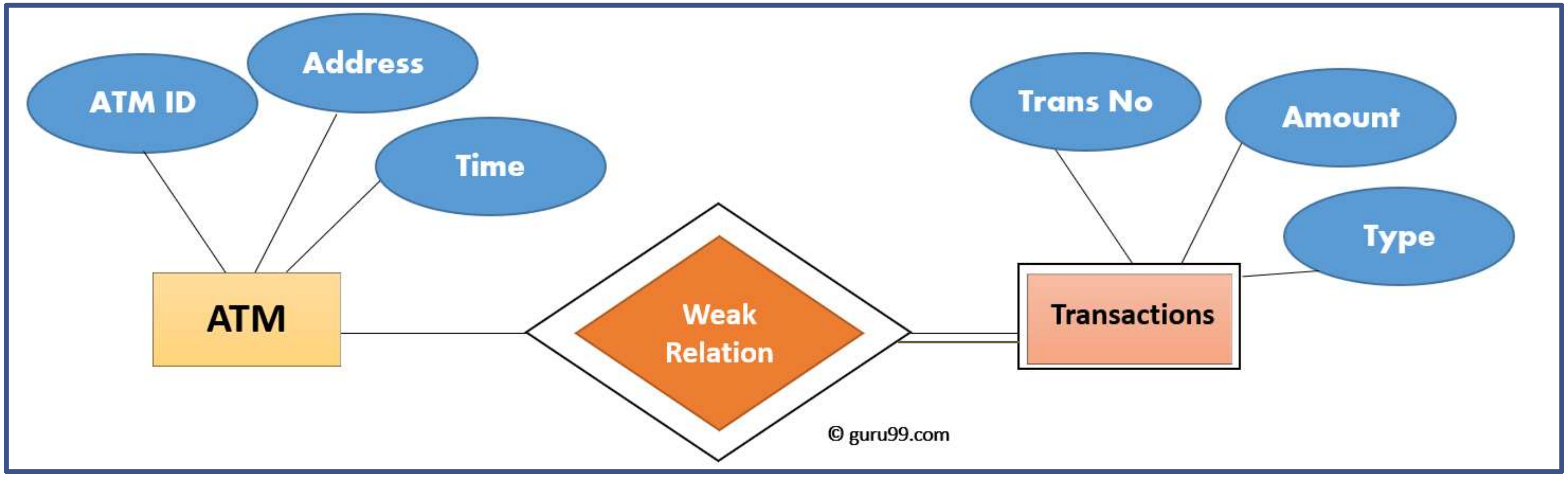
- An entity type which is dependent totally on another entity set is called **Weak Entity** type.
- The entity sets which do not have sufficient attributes to form a primary key are known as **weak entity sets** and the entity sets which have a primary key are known as **strong entity sets**.
- The weak entities have total participation constraint (**existence dependency**) in its identifying relationship with owner identity.
- Primary key of Loan is **C_id+L_name** (L_name is called **discriminator**)



WEAK ENTITIES

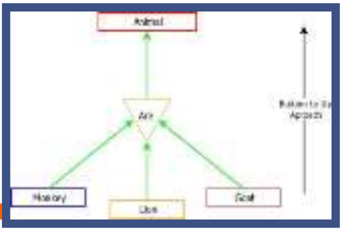


C_id	L_name	L-date
1112	L123	12/10/2007
1112	L889	10/03/2010
…	…	…
1323	L889	25/12/2012

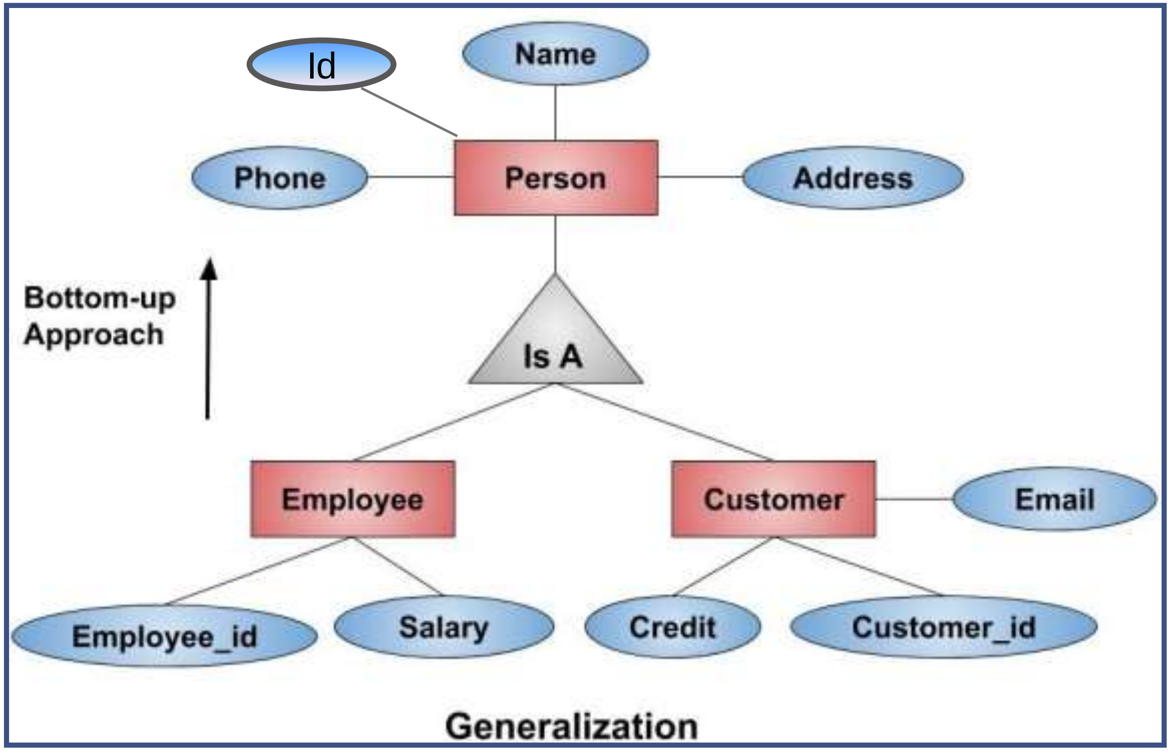


© guru99.com

EXTENDED E-R MODEL



- Generalization, Specialization and Aggregation in ER model are used for data abstraction in which abstraction mechanism is used to hide details of a set of objects.
- **Generalization** is like a bottom-up approach in which two or more entities of lower level combine to form a higher level entity if they have some attributes in common.
- In **generalization**, entities are combined to form a more generalized entity, i.e., subclasses are combined to make a superclass. It is a **bottom-up approach.**



Generalization

E-R TO TABLES (IS-A)

Two Options:

- **Option 1(Three tables)**

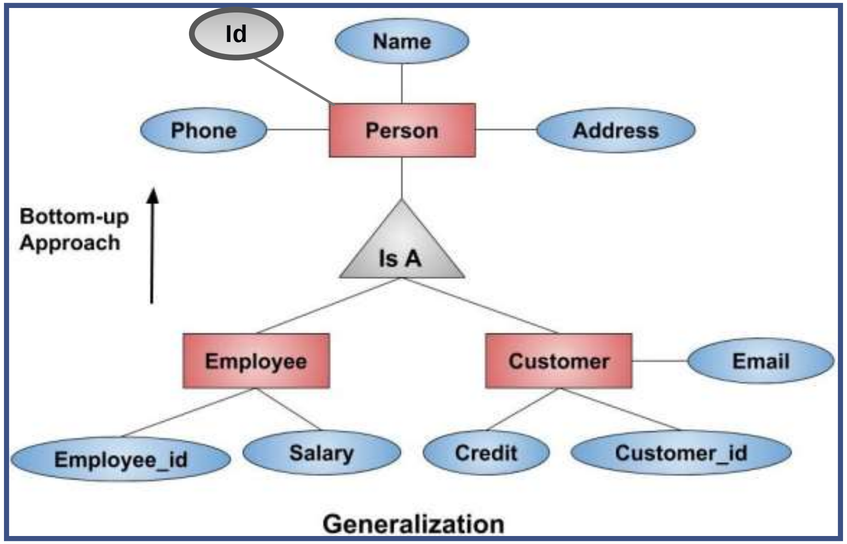
The Employee and Customer table inherit the related id from Person table.

1. Person (id, name, phone, address)
2. Employee(employee_id,salary)
3. Customer(cust_id, credit, email)

- **Option 2 (Two tables)**

The Employee and Customer table inherit all the related columns from Person table.

1. Employee(emp_id, name, phone, address, salary)
2. Customer(customer_id, name, phone, address, credit, email)



Generalization

Person table stores details about the Employees as well as Customers. (e.g. employee_ids start with ‘E’ and customer_ids start with ‘C’)

EXTENDED E-R MODEL

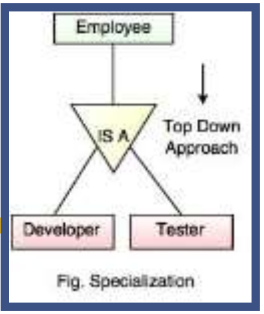
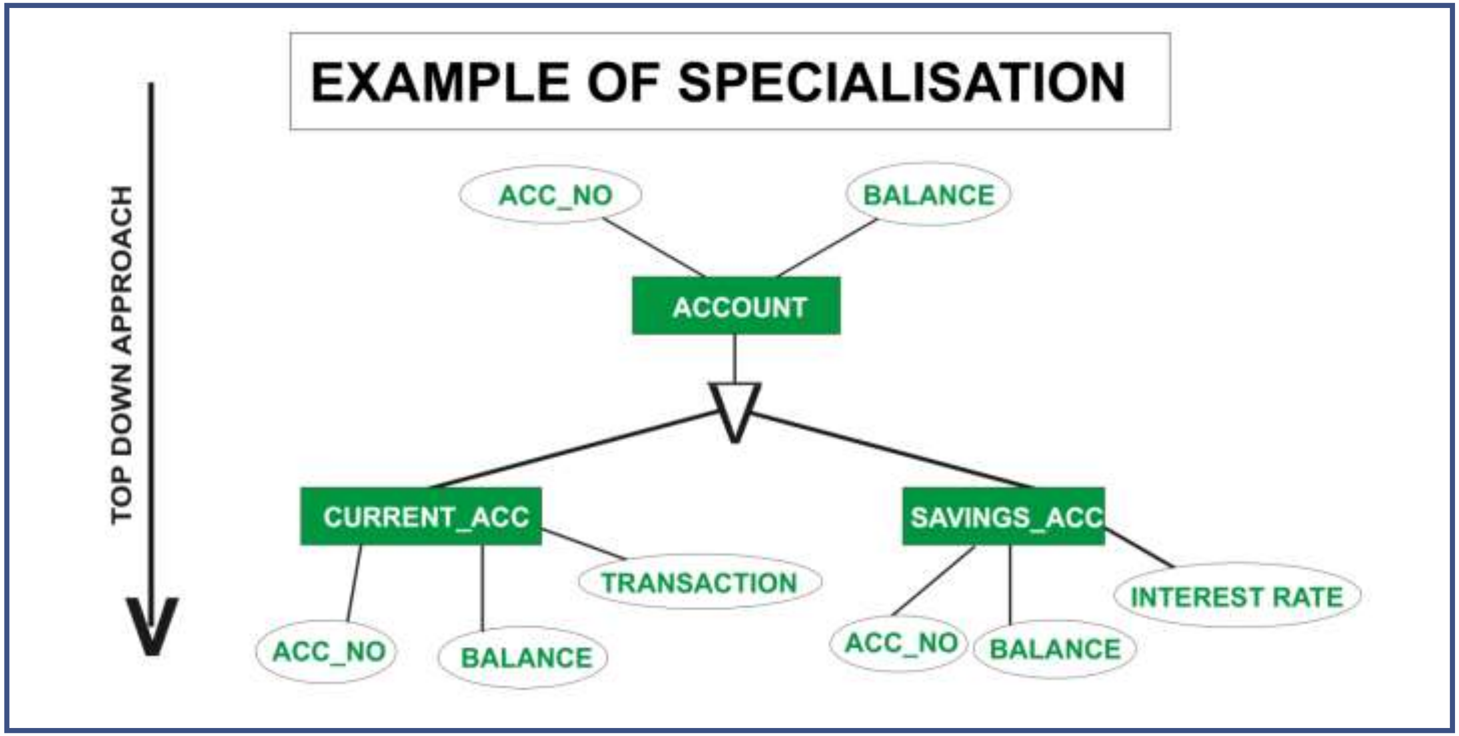
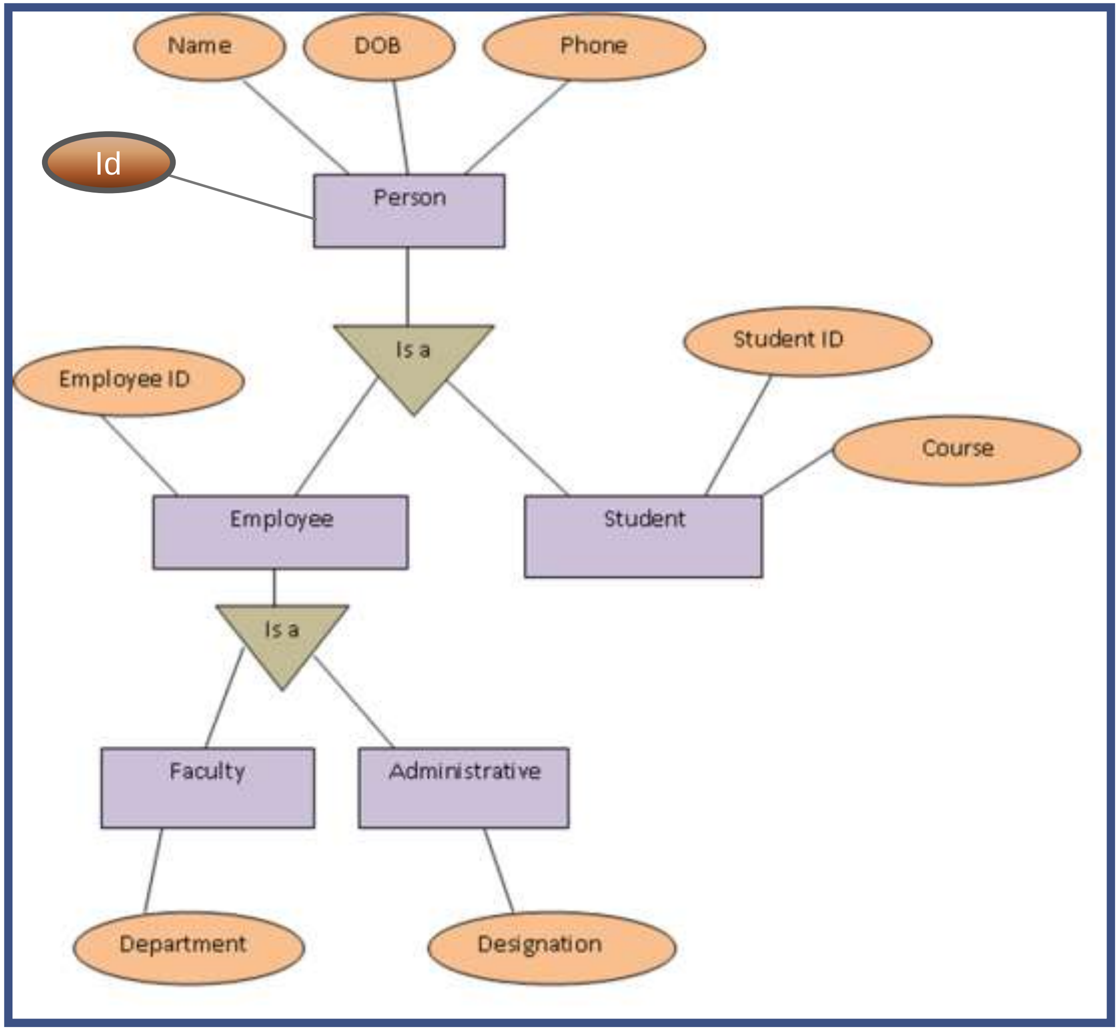


Fig. Specialization

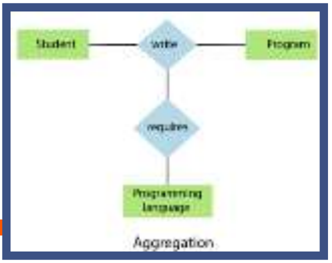
- **Specialization** is a process of identifying subsets of an entity that shares different characteristics.
- It breaks an entity into multiple entities from higher level (super class) to lower level (sub class). It is a **top-down approach** where higher level entity is specialized into two or more lower level entities.



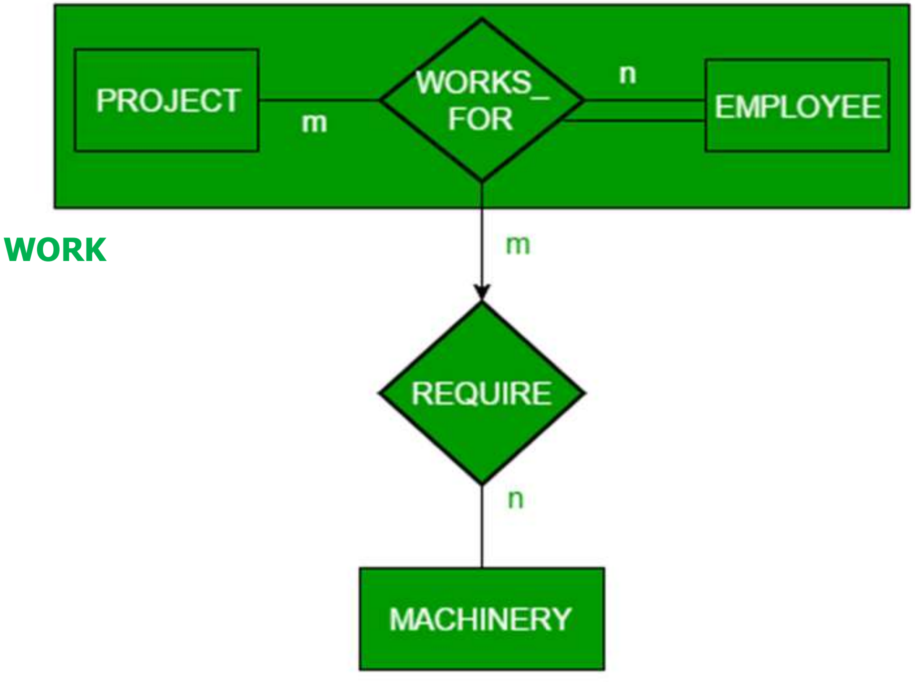
A LATTICE


Name
DOB
Phone
Id
Person
Is a
Student ID
Employee ID
Course
Employee
Student
Is a
Faculty
Administrative
Department
Designation

AGGREGATION



Draw an E-R diagram to depict details about employees working on number of projects and while they work the details of the machinery that they use.



Aggregation

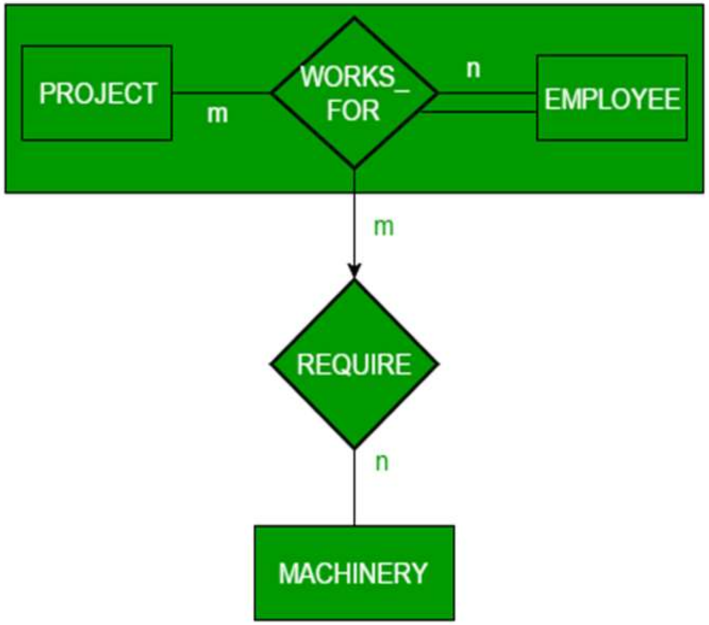
- Using **aggregation** we can express relationship among relationships.
- **Aggregation** shows 'has-a' or 'is-part-of' relationship between entities where one represents the 'whole' and other 'part'.
- The primary key of REQUIRE would be the combination of primary keys of all three entities.

AGGREGATION

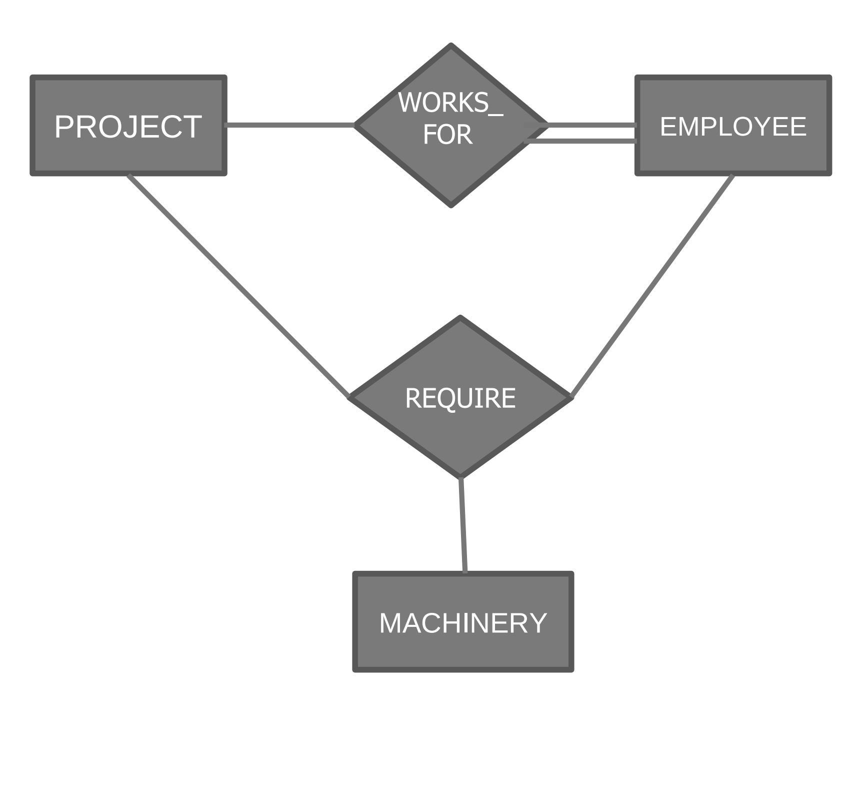
Student
write
Program
requires
Programming language

Aggregation

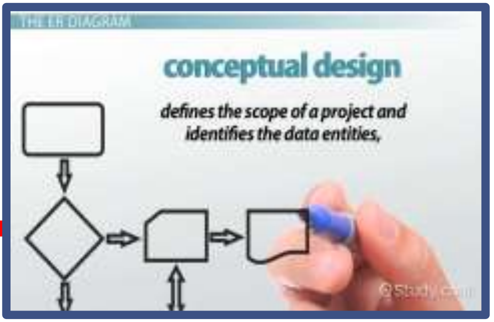
Draw an E-R diagram to depict details about employees working on number of projects and while they work the details of the machinery that they use.



Aggregation



E-R DESIGN DECISIONS

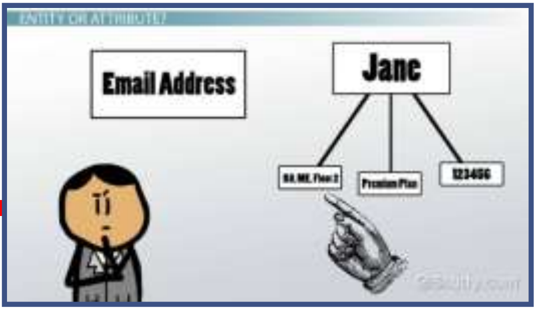


conceptual design

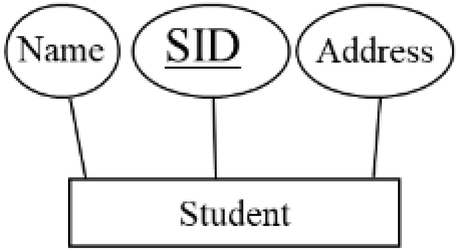
defines the scope of a project and identifies the data entities,

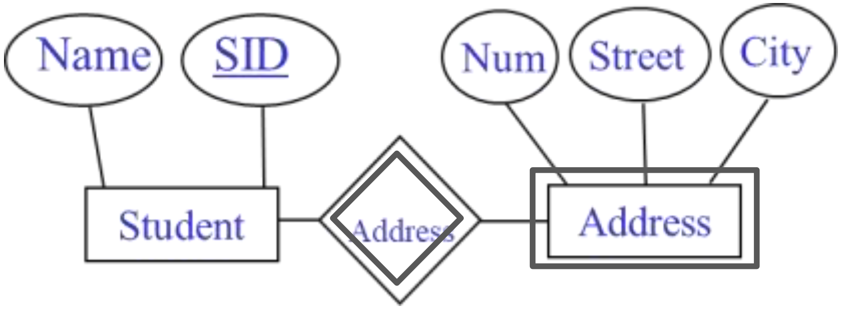
- The use of an attribute or entity set to represent an object.
- Whether a real-world concept is best expressed by an entity set or a relationship set.
- The use of a ternary relationship versus a pair of binary relationships.
- The use of a strong or weak entity set.
- The use of aggregation – can treat the aggregate entity set as a single unit without concern for the details of its internal structure.

ENTITY *vs.* ATTRIBUTE



- **Should *address* be an attribute of Employees or an entity connected to Employees by a relationship?**
- **Depends upon the use we want to make of address information, and the semantics of the data:**
  - If we have several addresses per employee, *address* must be an entity.
  - If the structure (city, street, etc.) is important, e.g., we want to retrieve employees in a given city, *address* must be modeled as an entity (since attribute values are atomic).

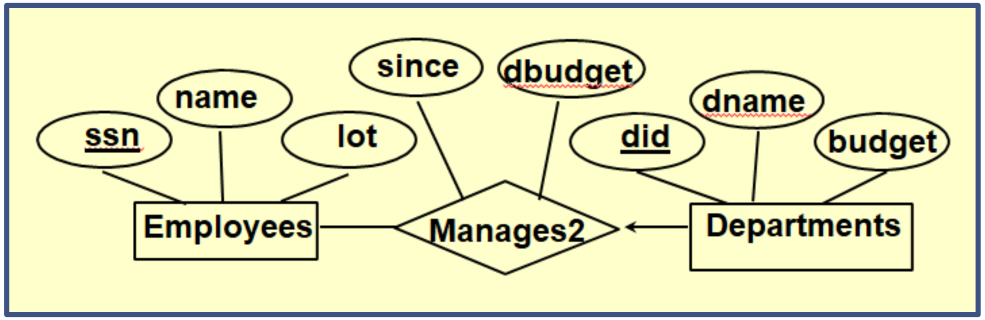




ENTITY *vs.* RELATIONSHIP

ER diagram is ok if a manager gets a separate discretionary budget for each dept.

What if a manager gets a common budget that covers all the managed depts? (Manages2 relation is for the manager)



Tables:

manages2(did, ssn, since, dbudget)

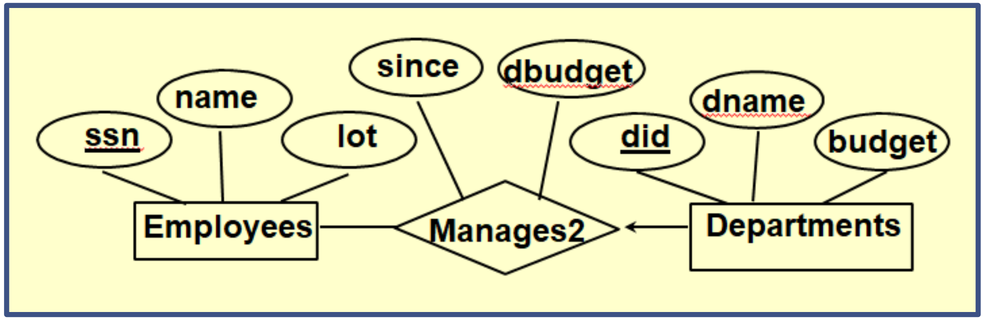
d1, 111, , 100000

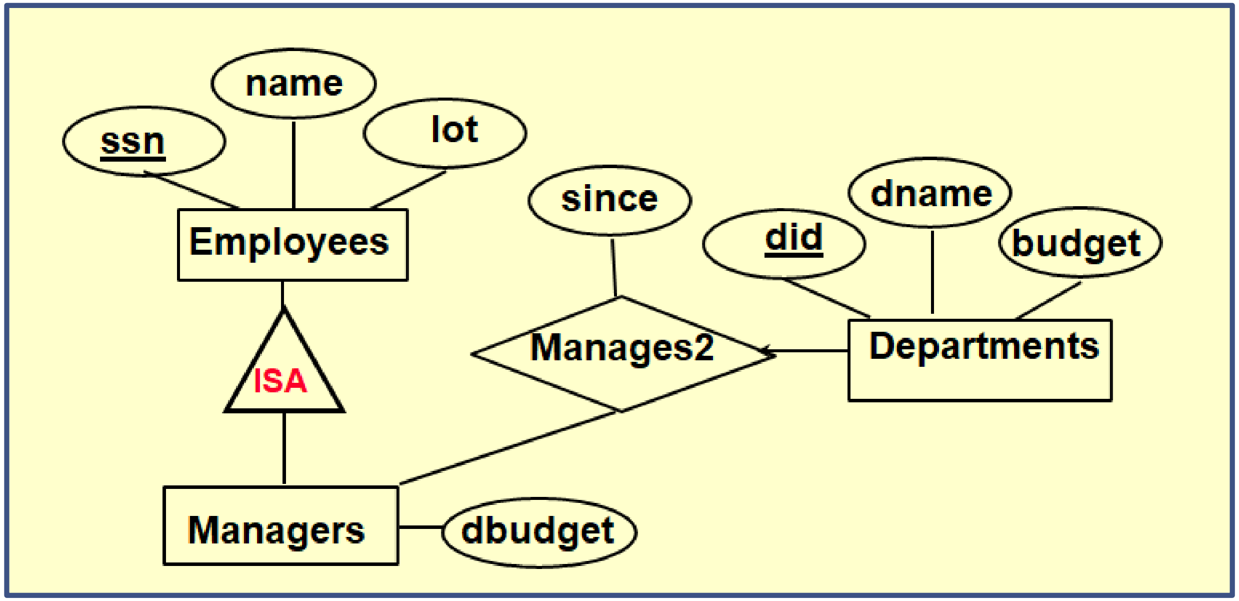
d2, 111, , 100000

d3, 123, , 250000

ENTITY *vs.* RELATIONSHIP

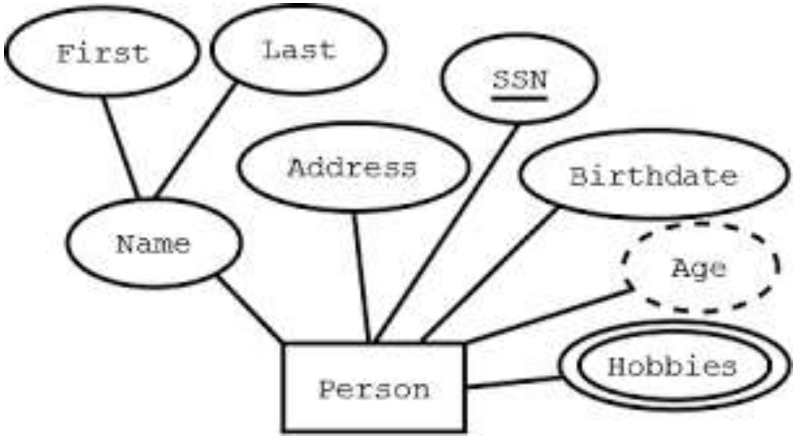
- What if a manager gets a common budget that covers all the managed depts? (Manages2 relation is for the manager)
  - Redundancy: *dbudget* stored for each dept managed by manager.
  - Misleading: Suggests *dbudget* associated with department-mgr combination.





E-R TO TABLES (STRONG)

- Converting Strong entity types:
  - Each entity type becomes a table (Person)
  - Each single-valued attribute becomes a column (SSN, Address, Birthdate)
  - Derived attributes are ignored (Age)
  - Composite attributes are represented by components (First, Last)
  - Multi-valued attributes are represented by a separate table (Hobbies)
  - The key attribute of the entity type becomes the primary key of the table (SSN)



Person

SSN	Fname	Lname	Address	Birthdate
111	Bill	Clinton	LA	12/12/64
112	Angelina	Jolie	NY	06/10/68
…				

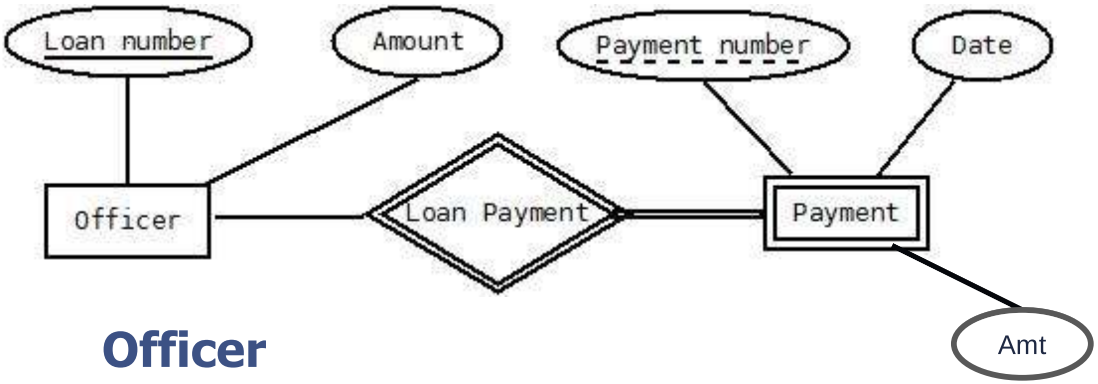
Hobby

SSN	Hobby
111	Reading
111	Hiking
112	cooking

SSN – Fk to Person
SSN+Hobby – Pk

E-R TO TABLES (WEAK)

- Converting weak entity types:
  - Weak entity types are converted into a table of their own, with the primary key of the strong entity acting as a foreign key in the table.
  - This foreign key along with the discriminator of the weak entity form the composite primary key of this table
  - Payment_number is the discriminator.



Officer

Loan_number	Name	Amount
111	Bill	100000
112	Angelina	350000
…		

Payment

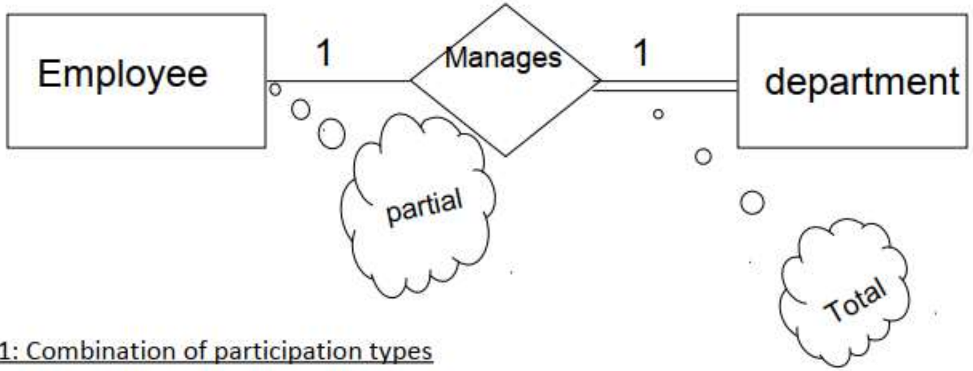
Loan_number	Payment_number	Date	Amt
111	1	1/1/20	1500
111	2	1/2/20	1500
112	1	5/6/21	500
112	2	7/7/21	500

E-R TO TABLES (RELATIONSHIPS)

Converting relationships:

- ➢ The way relationships are represented depends on the cardinality and the degree of the relationship
- ➢ The possible cardinalities are:
  - 1:1
  - 1:M / M:1
  - N:M
- ➢ The degrees are:
  - Unary
  - Binary
  - Ternary

Binary 1:1

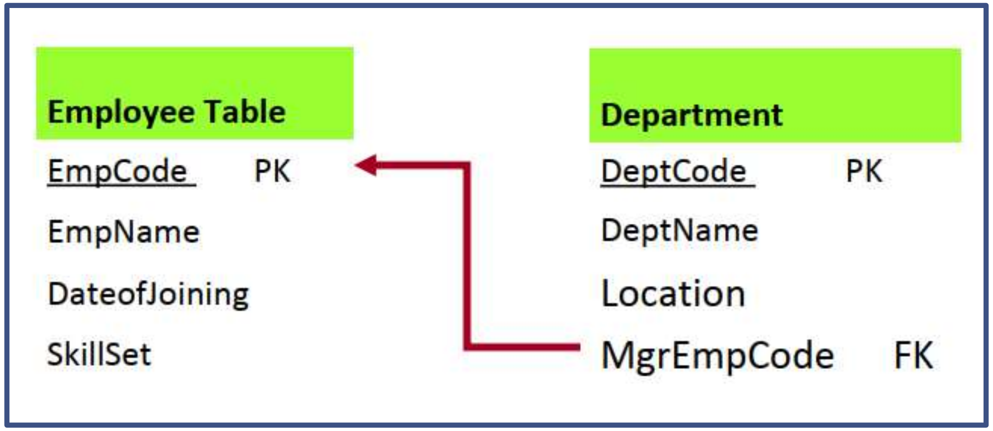


- Case 1: Combination of participation types

The primary key of the partial participant will become the foreign key of the total participant

Employee(E#, Name,...)

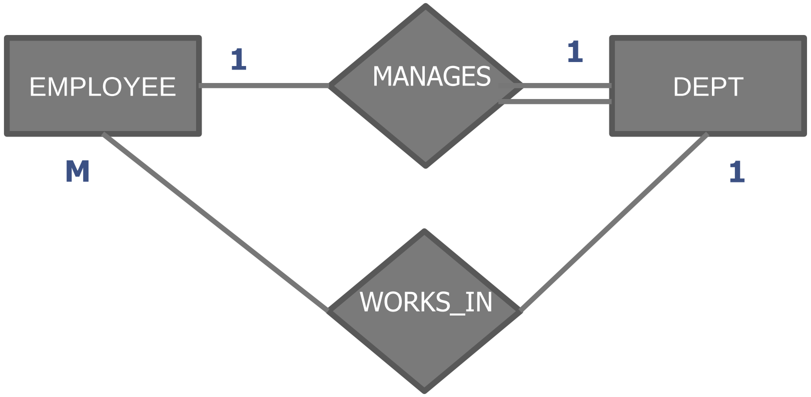
Department (Dept#, Name...,MgrE#)

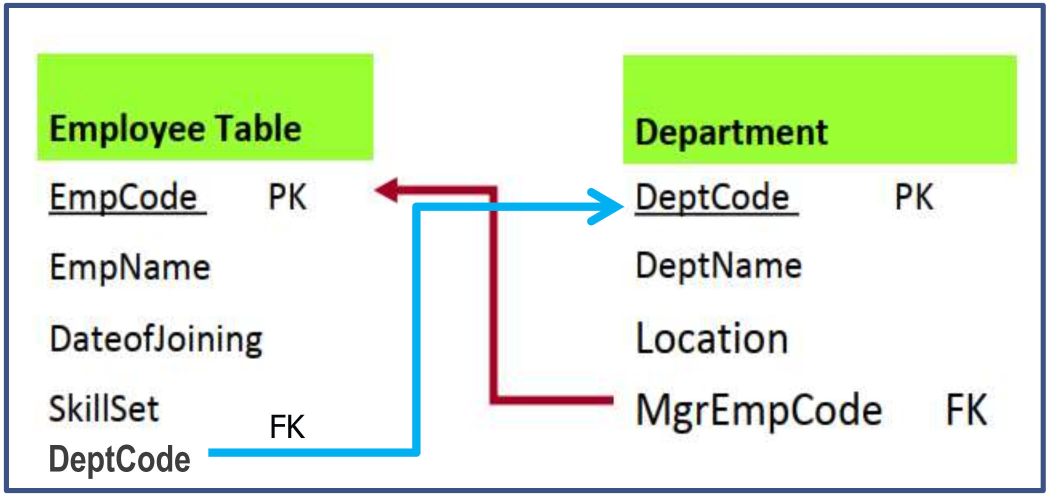


Employee Table	
EmpCode	PK
EmpName	
DateofJoining	
SkillSet	

Department	
DeptCode	PK
DeptName	
Location	
MgrEmpCode	FK

E-R TO TABLES (RELATIONSHIPS)





Employee Table

EmpCode PK

EmpName

DateofJoining

SkillSet

DeptCode FK

Department

DeptCode PK

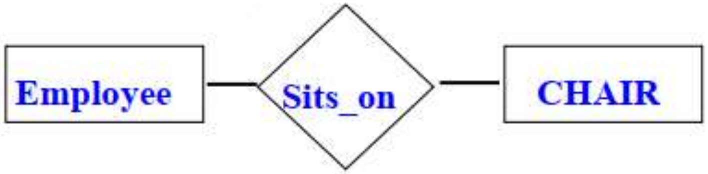
DeptName

Location

MgrEmpCode FK

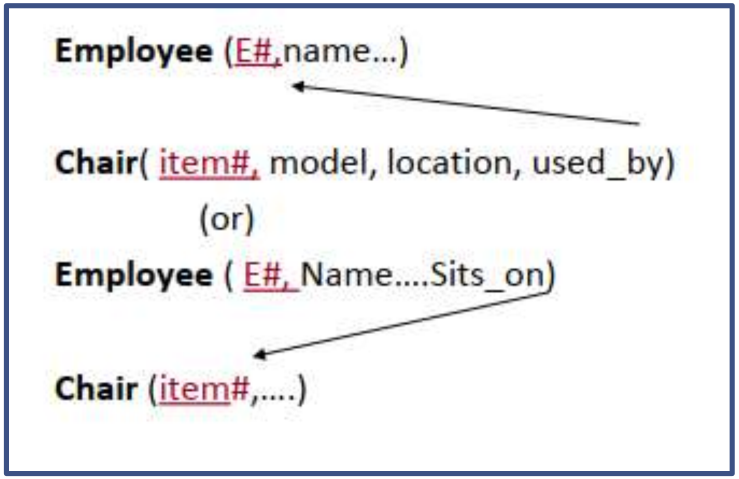
E-R TO TABLES (1:1)

Binary 1:1



- Case 2: Uniform participation types

The primary key of either of the participants can become a foreign key in the other



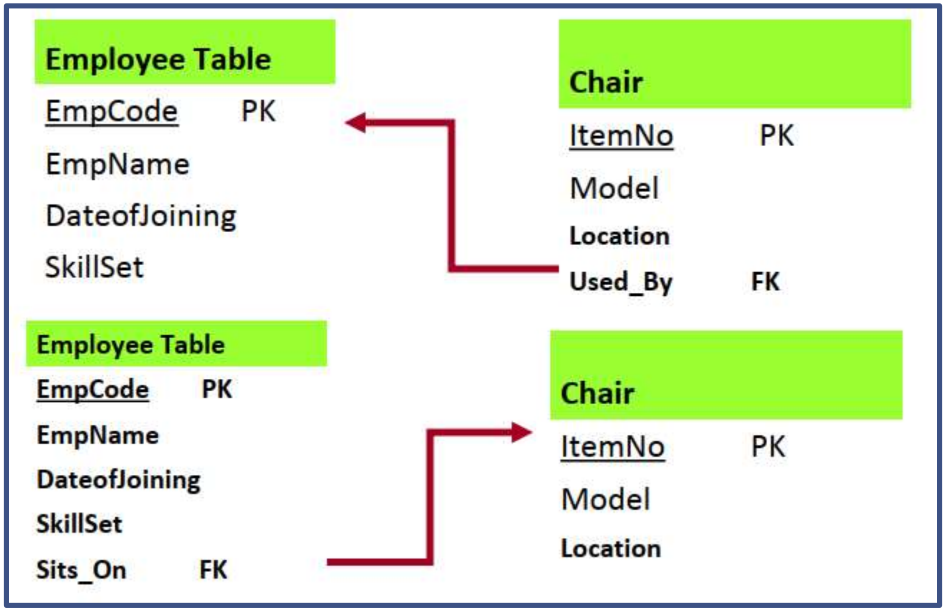
Employee (E#,name…)

Chair(item#, model, location, used_by)

(or)

Employee (E#, Name….Sits_on)

Chair (item#,….)



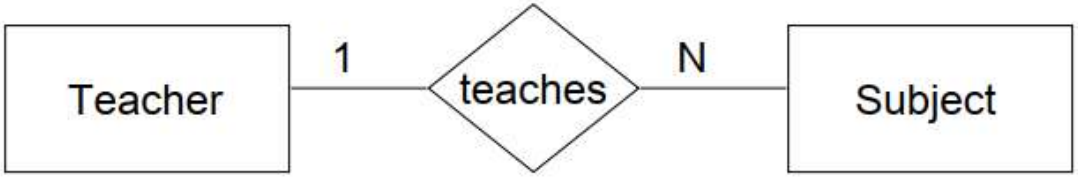
Employee Table	
EmpCode	PK
EmpName	
DateofJoining	
SkillSet	

Chair	
ItemNo	PK
Model	
Location	
Used_By	FK

Employee Table	
EmpCode	PK
EmpName	
DateofJoining	
SkillSet	
Sits_On	FK

Chair	
ItemNo	PK
Model	
Location	

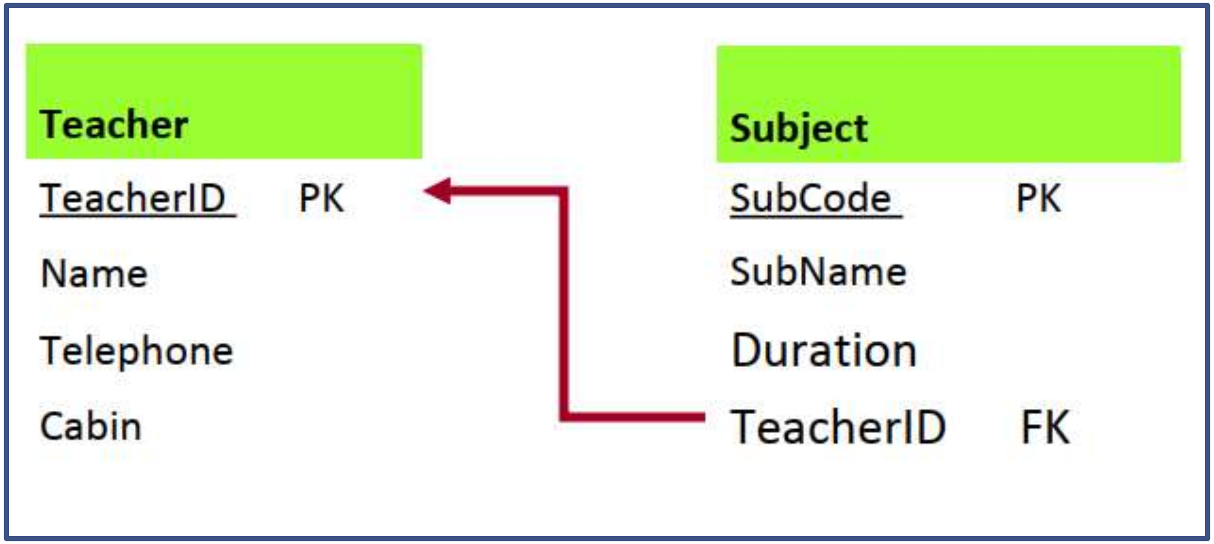
E-R TO TABLES (1:N)



The primary key of the relation on the “1” side of the relationship becomes a foreign key in the relation on the “N” side

Teacher (ID, Name, Telephone, …)

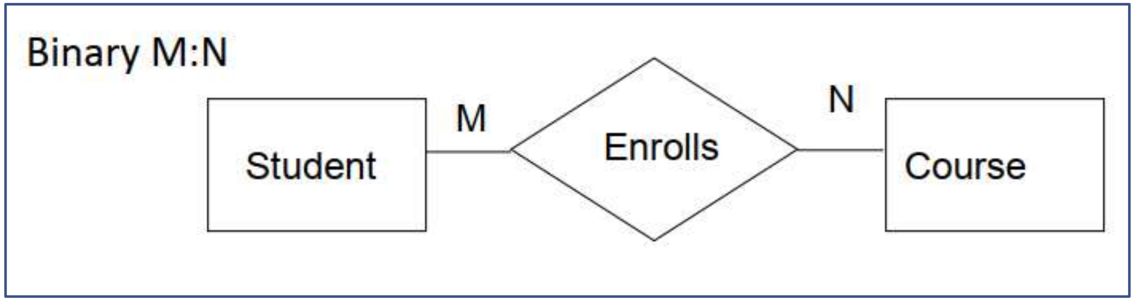
Subject (Code, Name, …, Teacher)

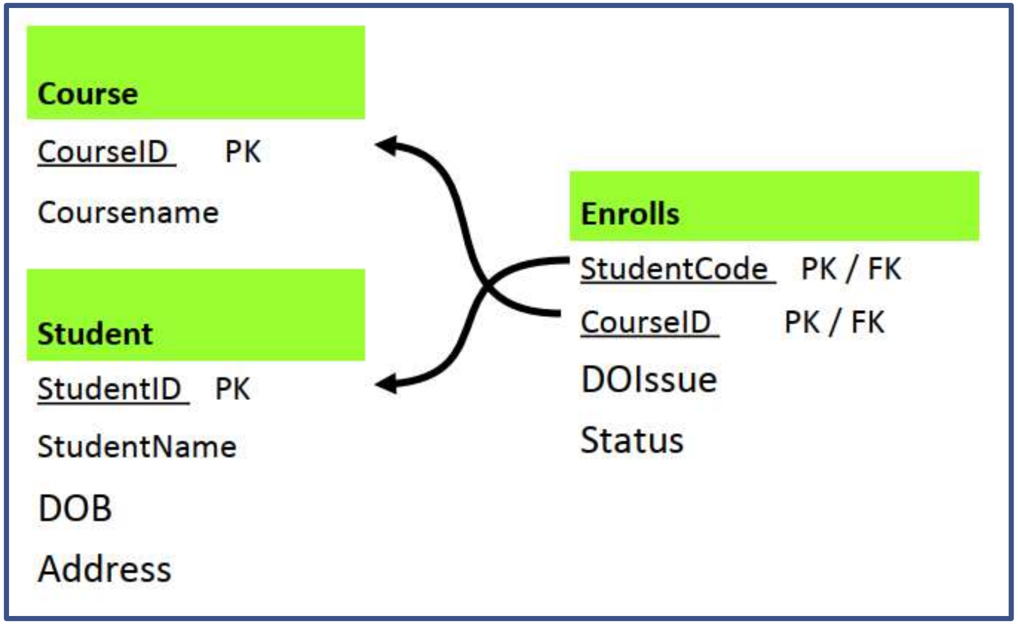


Teacher	
TeacherID	PK
Name	
Telephone	
Cabin	

Subject	
SubCode	PK
SubName	
Duration	
TeacherID	FK

E-R TO TABLES (M:N)





Course

CourseID PK

Coursename

Student

StudentID PK

StudentName

DOB

Address

Enrolls

StudentCode PK / FK

CourseID PK / FK

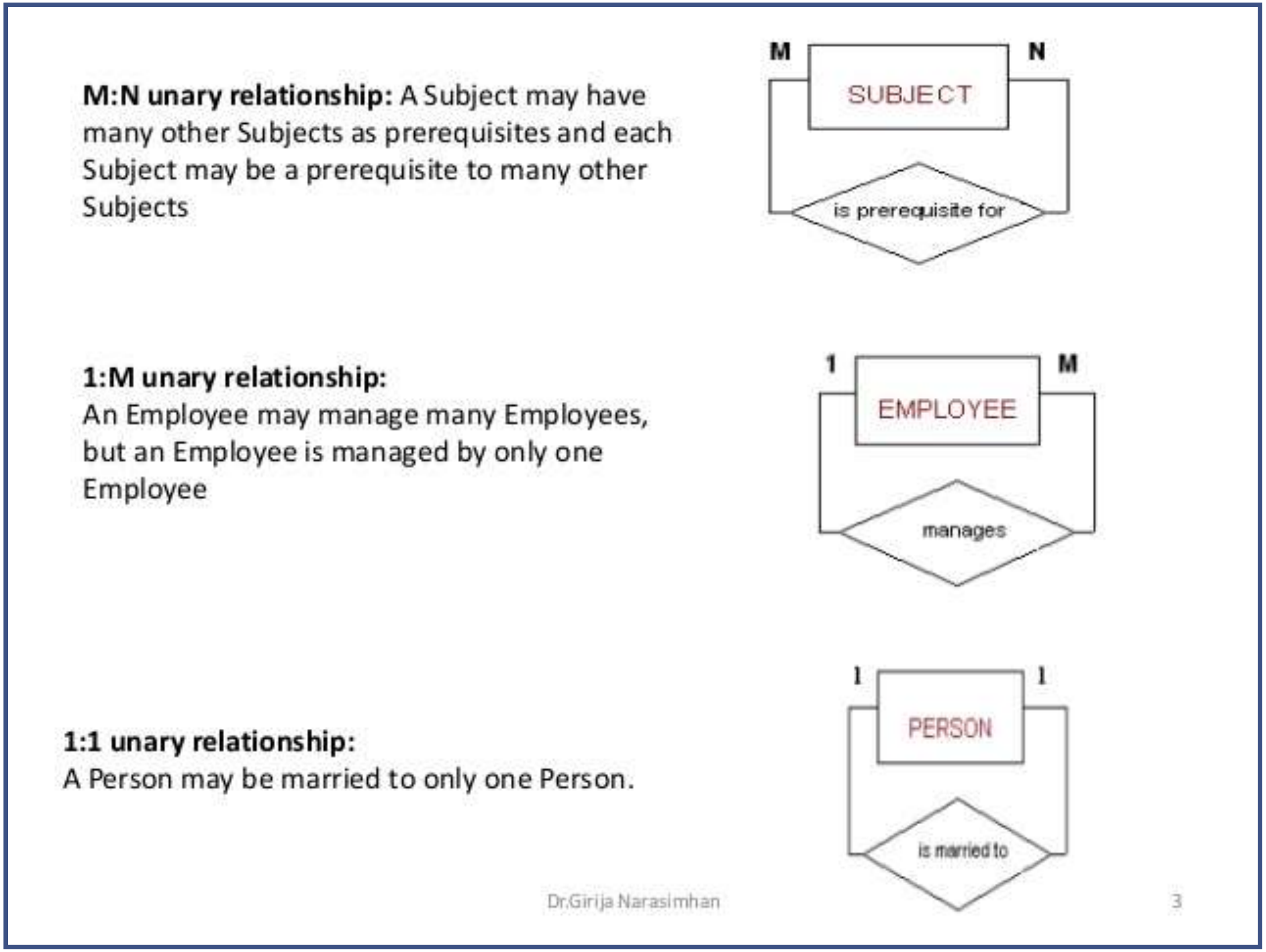
DOIssue

Status

- A new table is created to represent the relation
- Contains two FKs
- PK is the combination of both i.e.

StudentID CourseID

SELF REFERENCING



M:N unary relationship: A Subject may have many other Subjects as prerequisites and each Subject may be a prerequisite to many other Subjects

1:M unary relationship:
An Employee may manage many Employees, but an Employee is managed by only one Employee

1:1 unary relationship:
A Person may be married to only one Person.

M
N
SUBJECT
is prerequisite for
1
M
EMPLOYEE
manages
1
1
PERSON
is married to

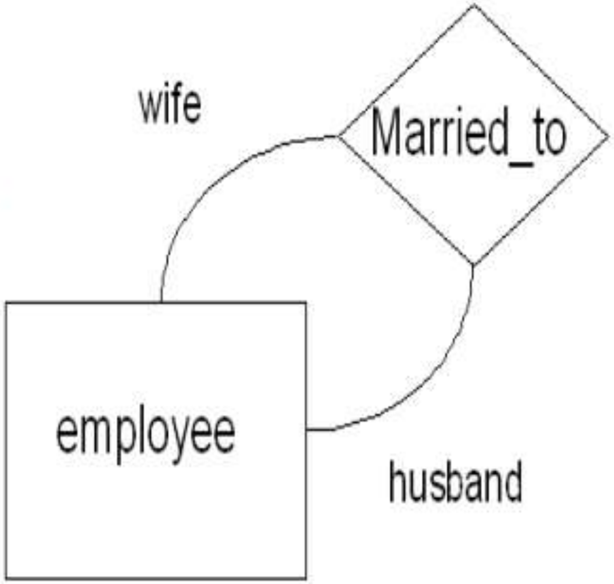
SELF REFERENCING (1:1)

shutterstock.com • 647795605

Self referencing 1:1

- Consider employees who are also a couple
- The primary key field itself will become foreign key in the same table

Employee(E#, Name,... **Spouse**)

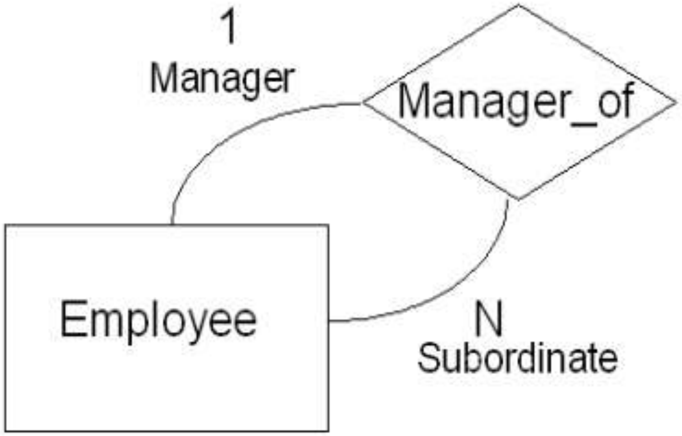


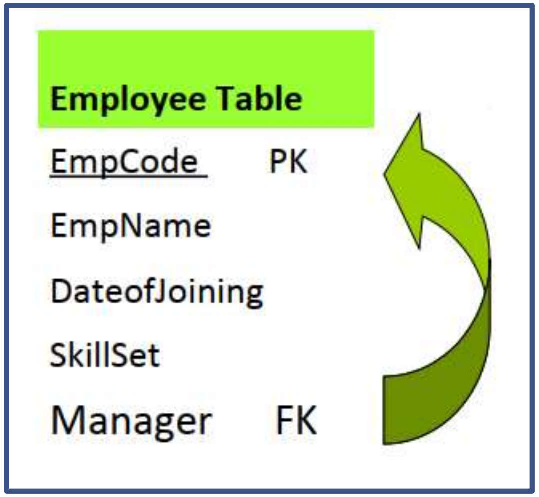
Employee Table	
EmpCode	PK
EmpName	
DateofJoining	
SkillSet	
Spouse	FK

SELF REFERENCING (1:N)

Self referencing 1:N

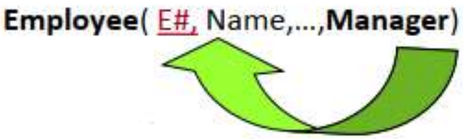
- The primary key field itself will become foreign key in the same table
- Same as unary 1:1





Employee Table

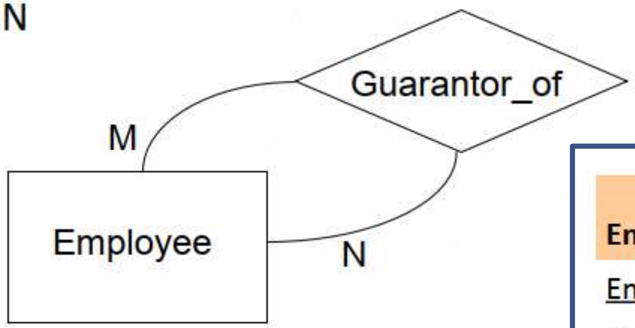
<u>EmpCode</u> PK
EmpName
DateofJoining
SkillSet
Manager FK

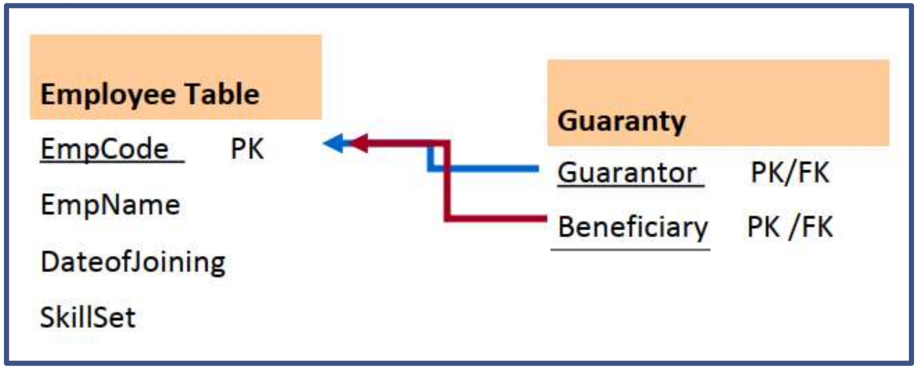


Employee(<u>E#,</u> Name,...,**Manager**)

SELF-REFERENCING M:N

Self referencing M:N





Employee Table

EmpCode PK

EmpName

DateofJoining

SkillSet

Guaranty

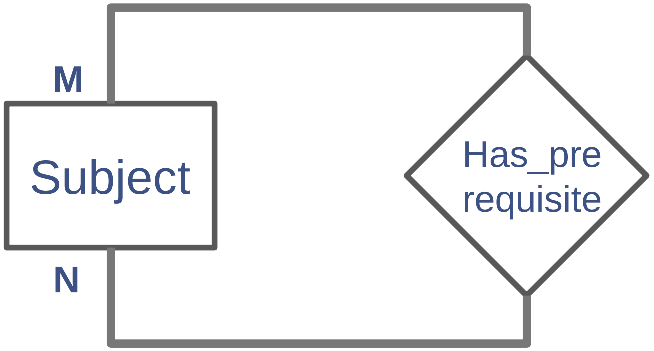
Guarantor PK/FK

Beneficiary PK /FK

- There will be two resulting tables. One to represent the entity and another to represent the M:N relationship as follows

Employee(E#, Name,...)

Guaranty(Guarantor, beneficiary)

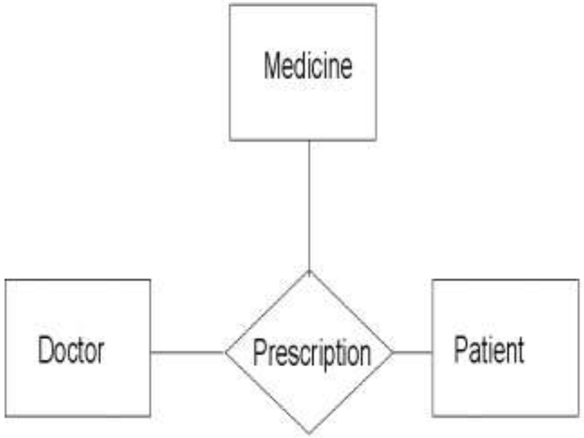


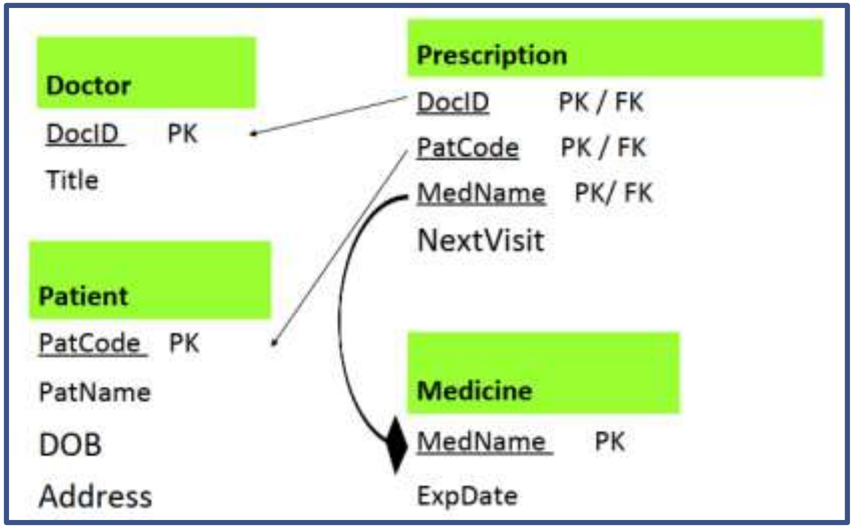
Subno	P_subno
DS	C
DBMS	DS
DBMS	C

TERNARY RELATIONS

Ternary relationship

- Represented by a new table
- The new table contains three foreign keys - one from each of the participating Entities
- The primary key of the new table is the combination of all three foreign keys
- Prescription (Doctor#, Patient #, Medicine_Name)





Doctor
DocID PK
Title

Patient
PatCode PK
PatName
DOB
Address

Prescription
DocID PK / FK
PatCode PK / FK
MedName PK/ FK
NextVisit

Medicine
MedName PK
ExpDate

EXAMPLES

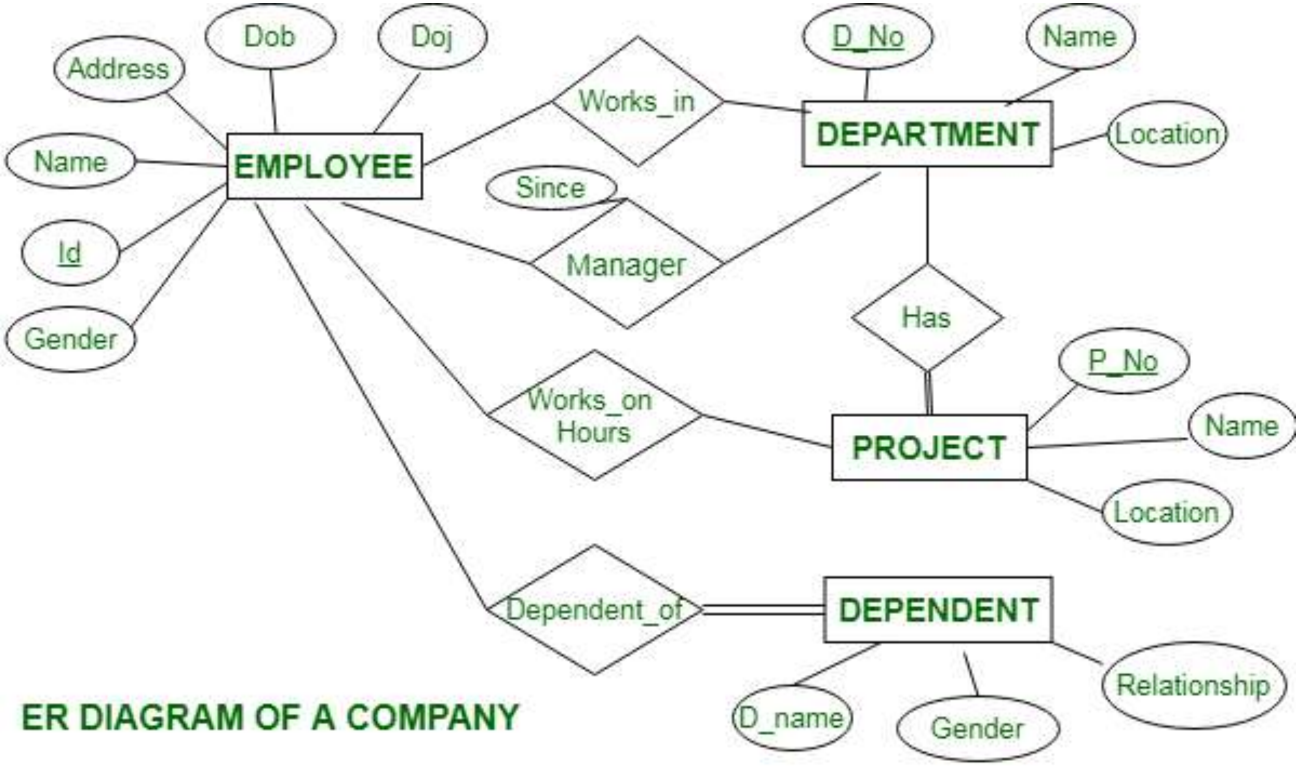
Draw an E-R Diagram to depict employees working in different departments. Some employees are managers of the departments. Every department handles a number of projects. One employee can work on more than one project. An employee can have zero or more dependents.

Entities:
DEPT
EMP
PROJECT
DEPENDENT

Relationships:
EMP – DEPT (M:1)
EMP – DEPT (1:1) – for manager of department
DEPT – PROJ (1:M)
EMP – PROJ (M:N)
EMP – DEPENDENT (1:M)

EXAMPLES

Draw an E-R Diagram to depict employees working in different departments. Some employees are managers of the departments. Every department handles a number of projects. One employee can work on more than one project. An employee can have zero or more dependents.



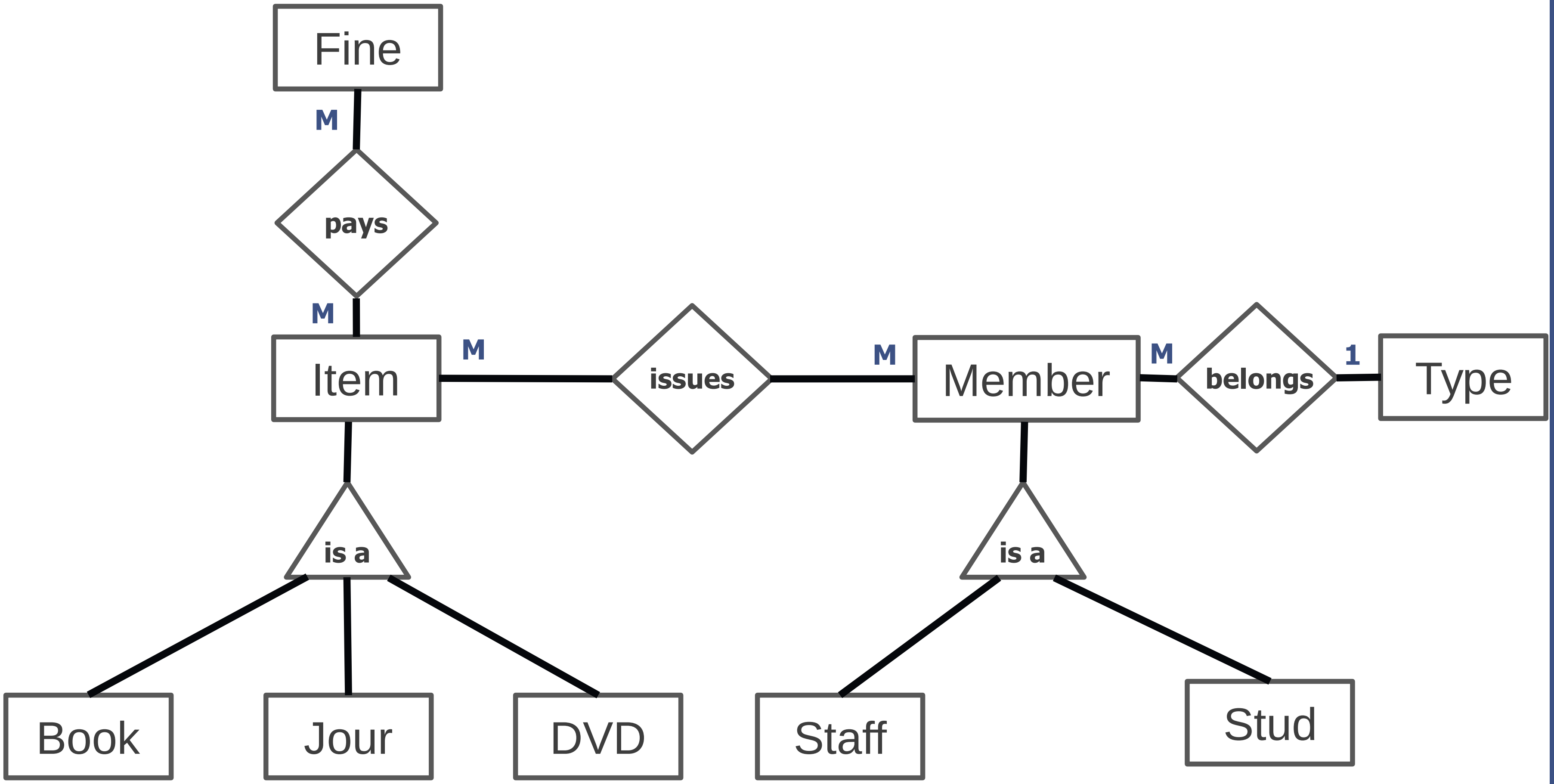
ER DIAGRAM OF A COMPANY

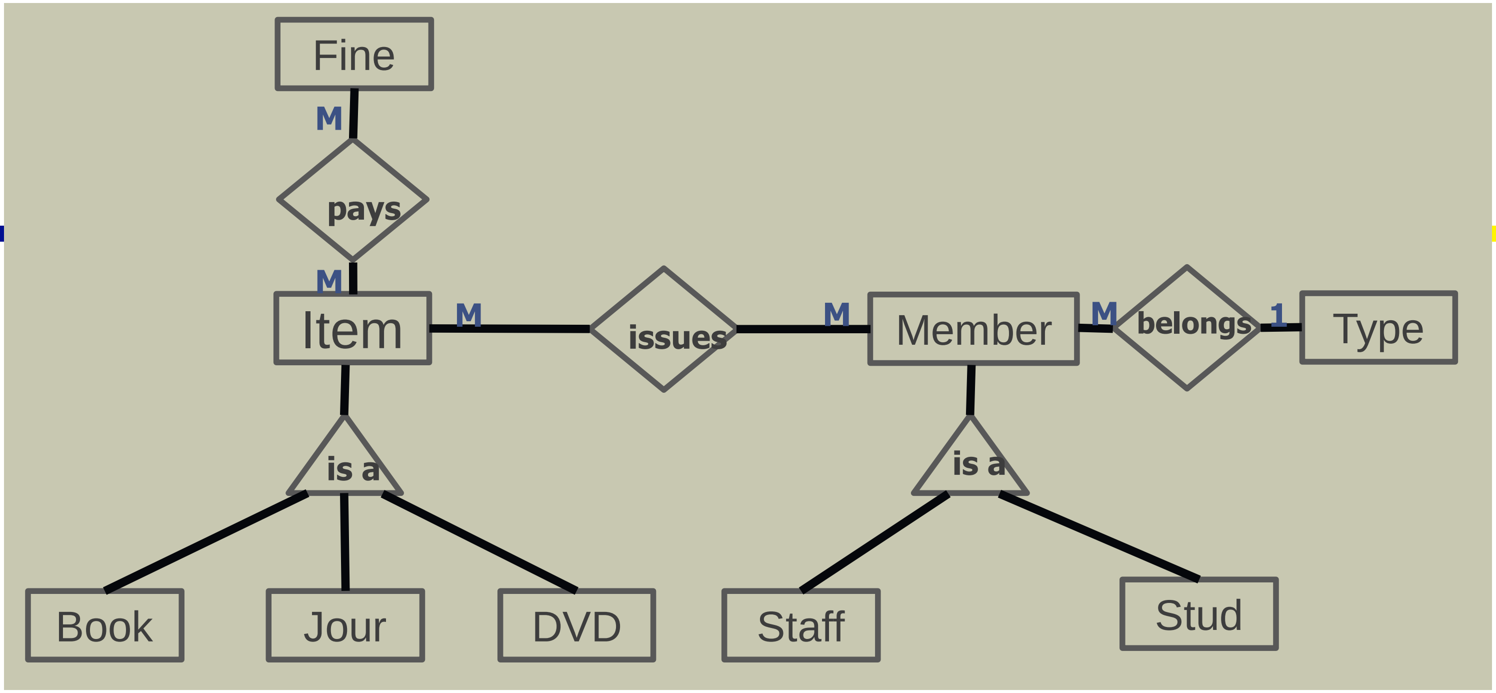
EXAMPLES

Draw an E-R Diagram to depict details about different departments of Faculty of Technology and Engineering. There are staff members and students in every department. Staff member can be Teaching or Administrative. Each have their own set of different information which needs to be stored. Passport details of staff members with all information is also to be maintained.

Draw an E-R Diagram for the CSE Departmental Library Management System. There are books, journals and DVDs in the library which can be issued. The members of the library are the staff and students. Currently every staff member can issue 6 books and keep them for 6 months. Every student can issue 2 books and keep them for 15 days. Include fine options for late return also.

Draw an E-R Diagram for the CSE Departmental Library Management System. There are books, journals and DVDs in the library which can be issued. The members of the library are the staff and students. Currently every staff member can issue 6 books and keep them for 6 months. Every student can issue 2 books and keep them for 15 days. Include fine options for late return also.





TABLES:

item, book, journal, DVD

type(typeid, type_desc, no_books, no_days)

t1	staff	6	180
t2	student	2	15

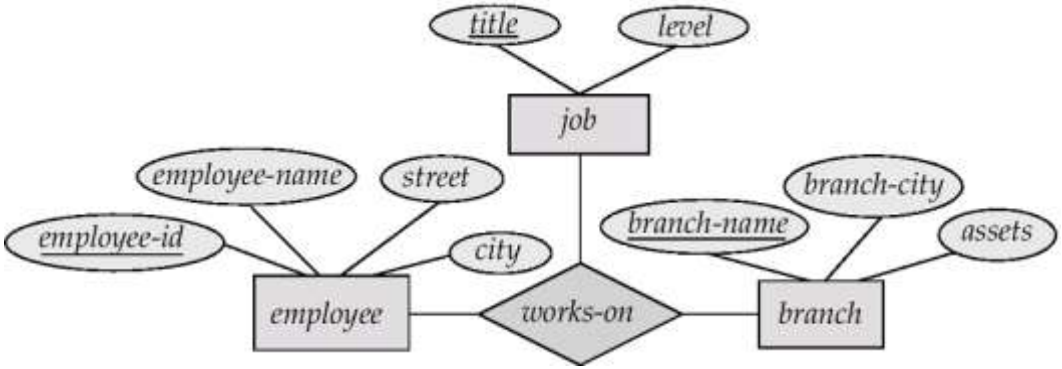
member(mid, mname, …, typeid), staff, student

issues(issno, itemno, mid, dt_issue, dt_return, fine_applied)

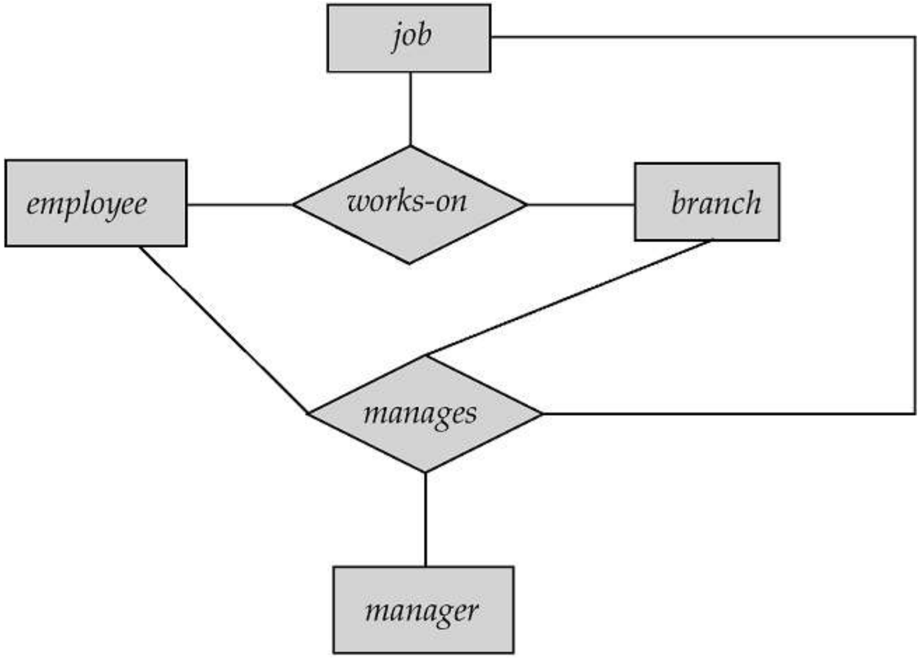
fine(days, amt)

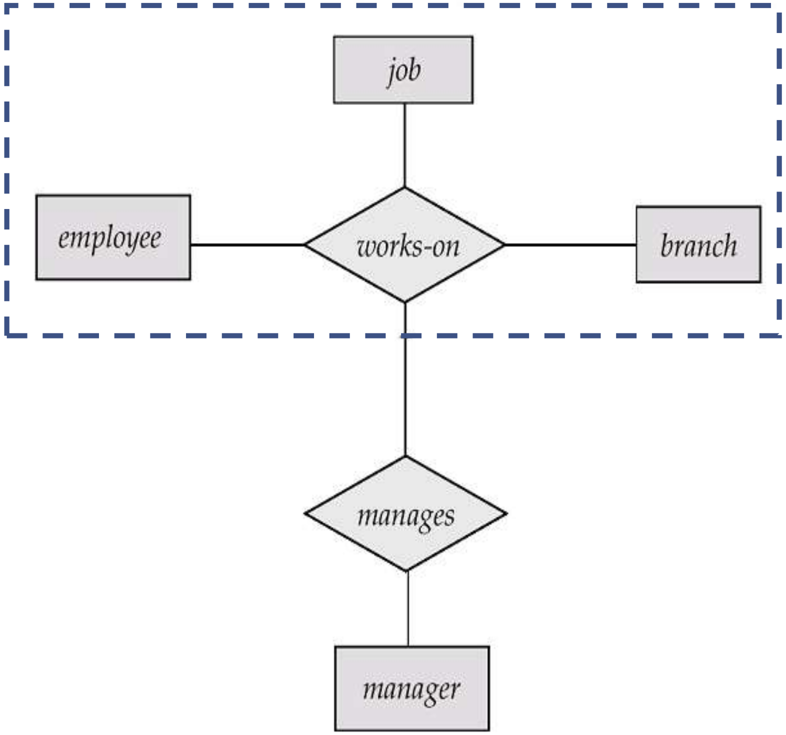
pays(issno, fine_amt)

EXAMPLES



An E-R diagram with a ternary relation and an alternate Aggregation replacing the ternary relation.





END OF E-R MODEL