Entity Relationship Modeling

pm jat @ daiict

Data Modeling

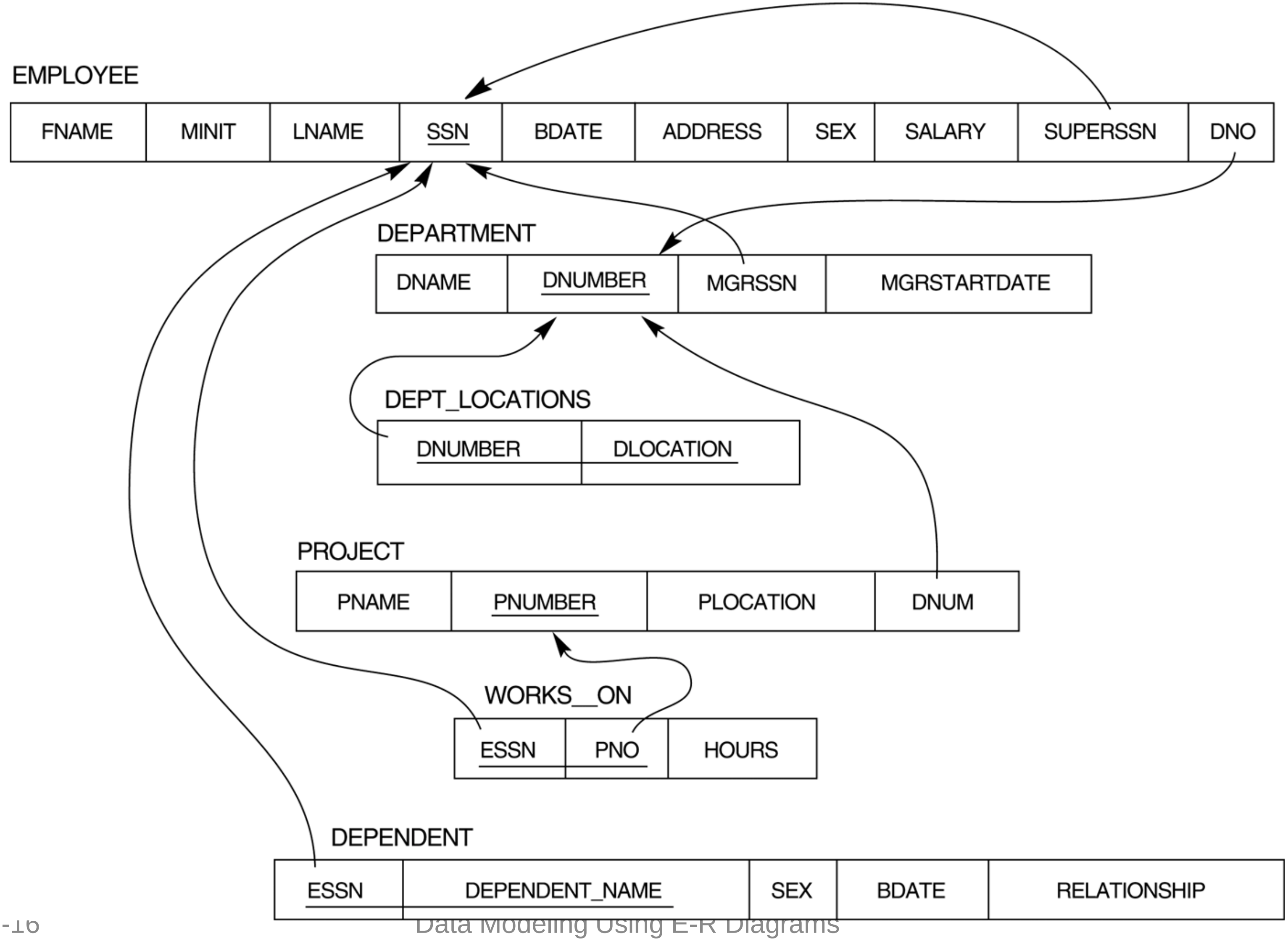
- Used to depict the database schema.
- Data Modeling done at three levels-
  - Conceptual Level (for capturing correct and complete database requirements)
  - Implementation Level
  - Physical Level

What to talk about?

- What schema diagram tells you?

- ER Modeling
  - What is ER Modeling and ER Diagram?
  - ERD concepts and their notations
  - How do you draw ERD for given situation

What does this diagram tells you?



EMPLOYEE

FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO

DEPARTMENT

DNAME	DNUMBER	MGRSSN	MGRSTARTDATE

DEPT_LOCATIONS

DNUMBER	DLOCATION

PROJECT

PNAME	PNUMBER	PLOCATION	DNUM

WORKS__ON

ESSN	PNO	HOURS

DEPENDENT

ESSN	DEPENDENT_NAME	SEX	BDATE	RELATIONSHIP

What schema diagram tells you?

- This is diagram that shows partial description of "relational" "schema" of company database.

- That is database schema in relational implementation model

- We also use term database model for schema diagram

- What does a relational schema describes about?

What schema diagram tells you?

- Relational schema typically describes about –
  - Relations
  - Constraints: Domain, Entity Integrity, Referential Integrity, Not Null, Unique, Check etc.
- There are more data modeling techniques than relational

Entity Relationship Modeling

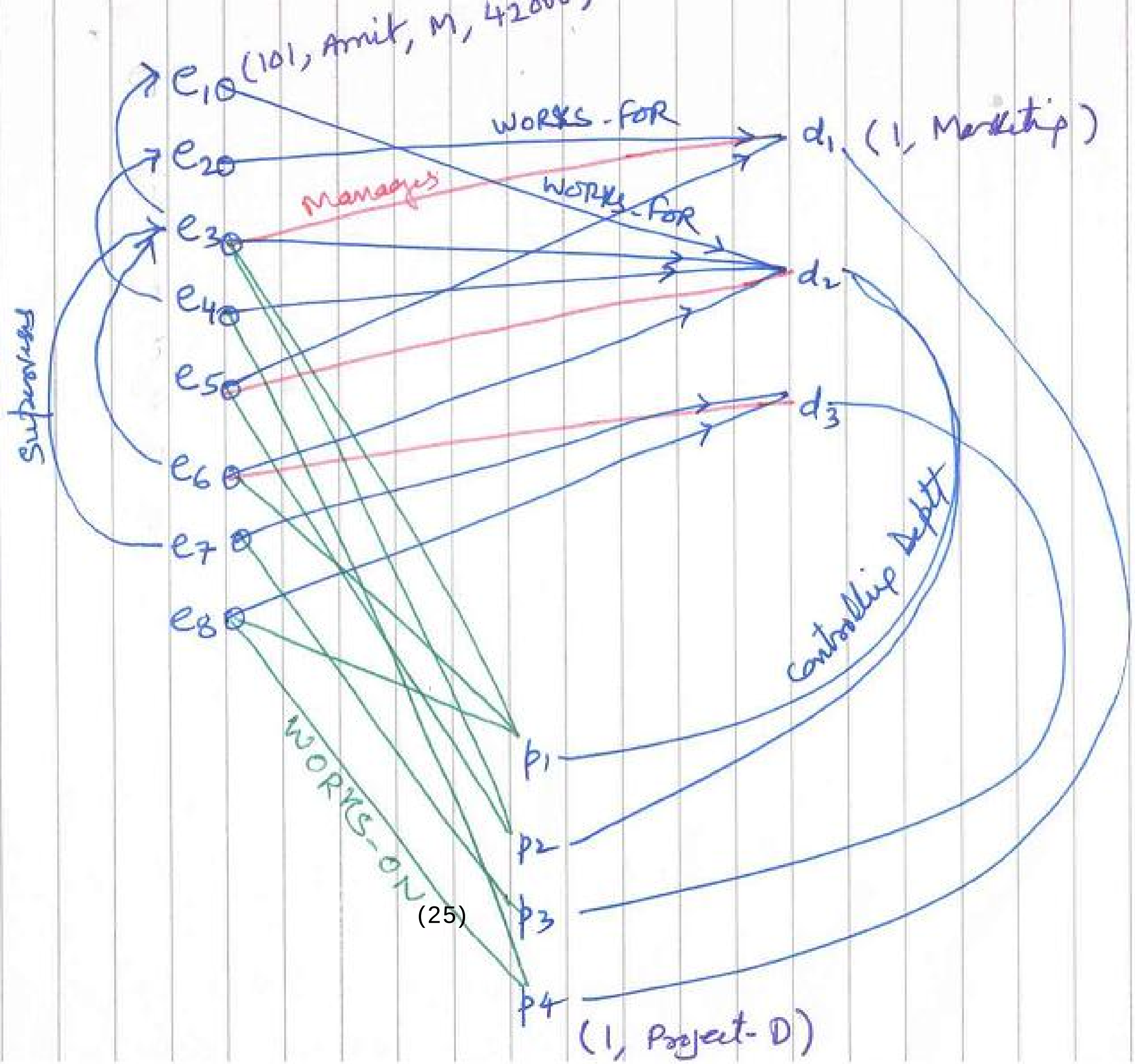
- We have learned Relational Modeling.
- Another popular modeling technique is Entity-Relationship, or ER-Modeling for short
- ER modeling is depicted using ER Diagrams (ERD)
- ER Modeling is a conceptual modeling, and does not allow defining manipulation of databases.

Entity Relationship Modeling

- In ER model, database is seen as a group of Entities and relationship between these entities

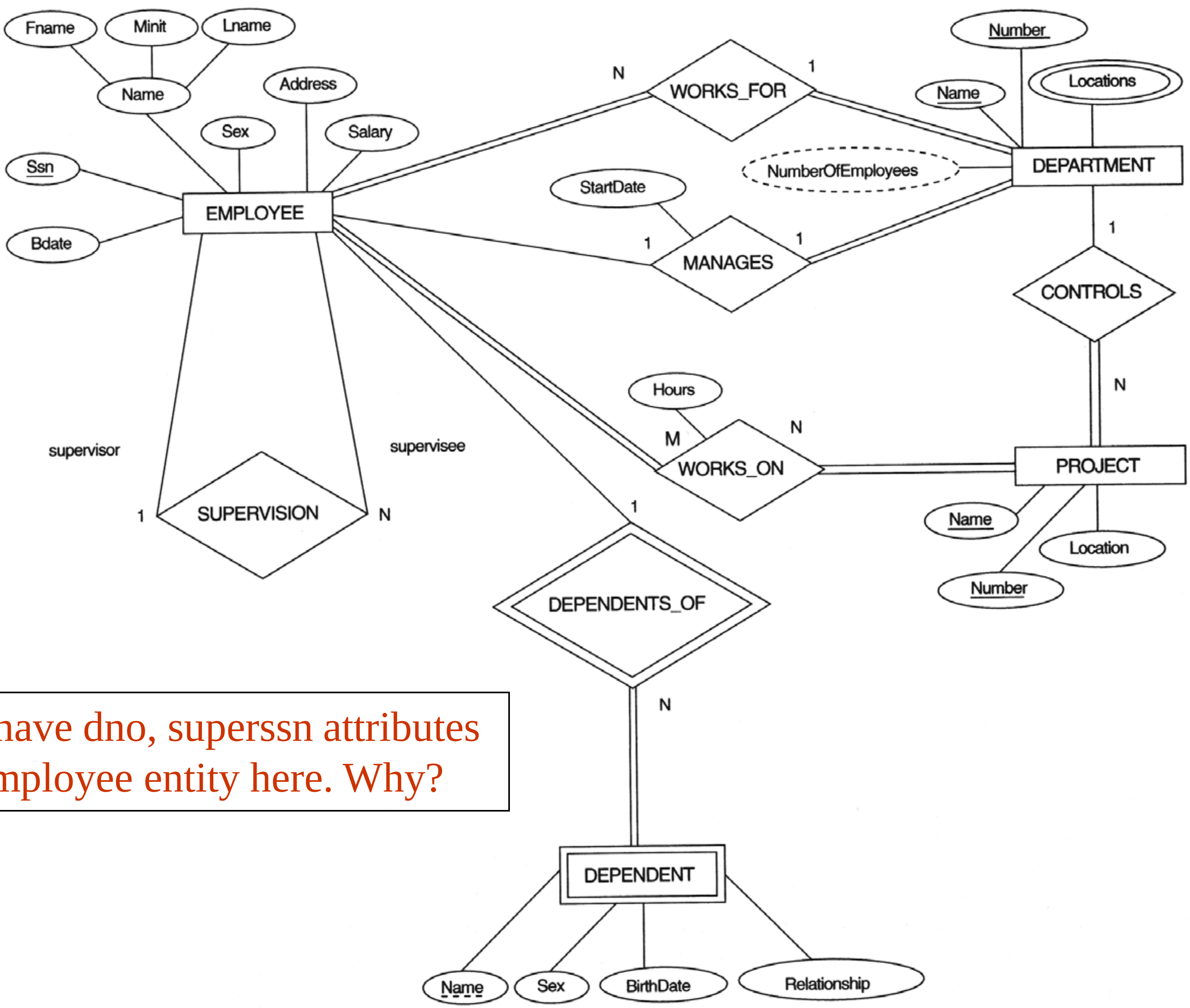
- It is primarily used as first hand sketch for database design – aims to capture required data-items with their semantics

- Recall that model typically captures database schema, and hence ERD also used to depict conceptual schema of the database


e1 (101, Amit, M, 42000)
e2
e3
e4
e5
e6
e7
e8
WORKS-FOR
d1 (1, Marketing)
Manages
WORKS-FOR
d2
d3
Supervises
Controlling Deptt
WORKS-ON
p1
p2
p3
p4 (1, Project-D)

(25)

ERD: Company Schema



We do not have dno, superssn attributes [FKs] in Employee entity here. Why?

E-R Diagram

- A detailed ER Diagram uses three concepts to describe database schema -
  - Entities – contains their name and attributes
  - Relationship between entities
  - Cardinality/Participation Constraints on entities for their participation into relationships

Entity Relationship Modeling

- How to go about creating ERDs for given database scenario?
  - Should know various notations used in the diagram – simpler part
  - Identify Entities and their relationship (and other participatory constraints) and put them in diagram – difficult part.

Entity in ER Model

- Entity is most important thing in ER model.
- Entity means something that is visible and has independent existence in given database scenario.
- A good check for Independent existence - if the thing has key attribute to identify itself in set of entities of this type.

Important Concepts in ERD

- Entity: Type and Instance
- Attribute: Atomic and Composite, Key Attribute
- Entity Set
- Relationship: Type and Instance
- Relationship Set
- Cardinality and Participation constraints
- Weak Entity, and
- Some more ….

Entity and Entity Types

- All entities are instances of some entity type.
- In ERD, that depicts conceptual schema, we define entity types.

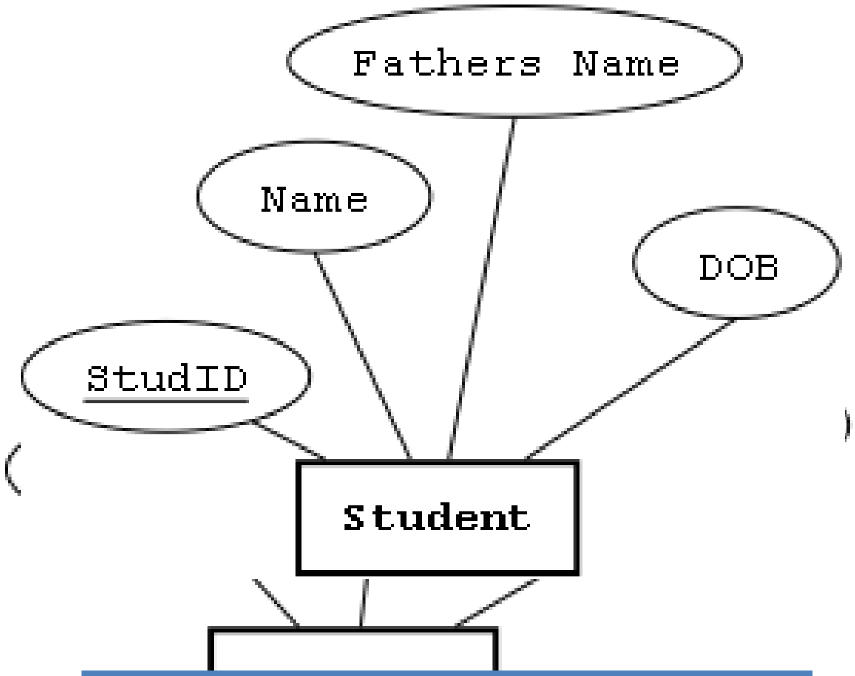
Describing Entity Type in ERD

- If we attempt to describe student entity, typically it can be like this-
  - Name of Entity: Student
  - Attributes: StudID, Name, DOB, FathersName, ..
  - Each Attribute derives value from a value set (domain)
  - StudID uniquely identifies a student entity in the set

- This is description of **Student** entity, and we call it “Entity Type”

- Similarly we describe **Program** *Entity Type*

Describing Entity Type in ERD

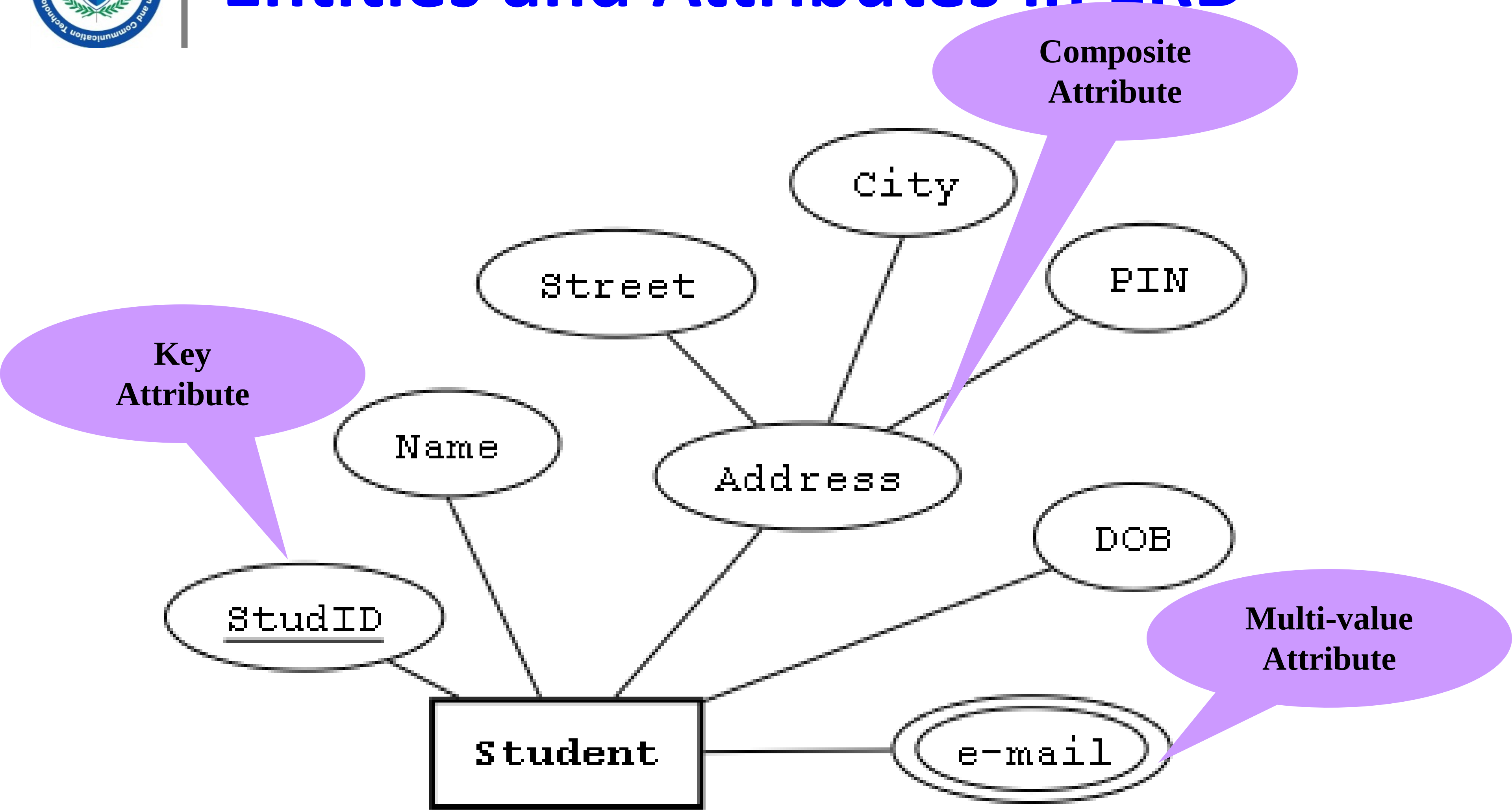


Notations used here are, originally suggested by Dr Peter Chen in ’76, known as Chen’s notation

Entities and Attributes

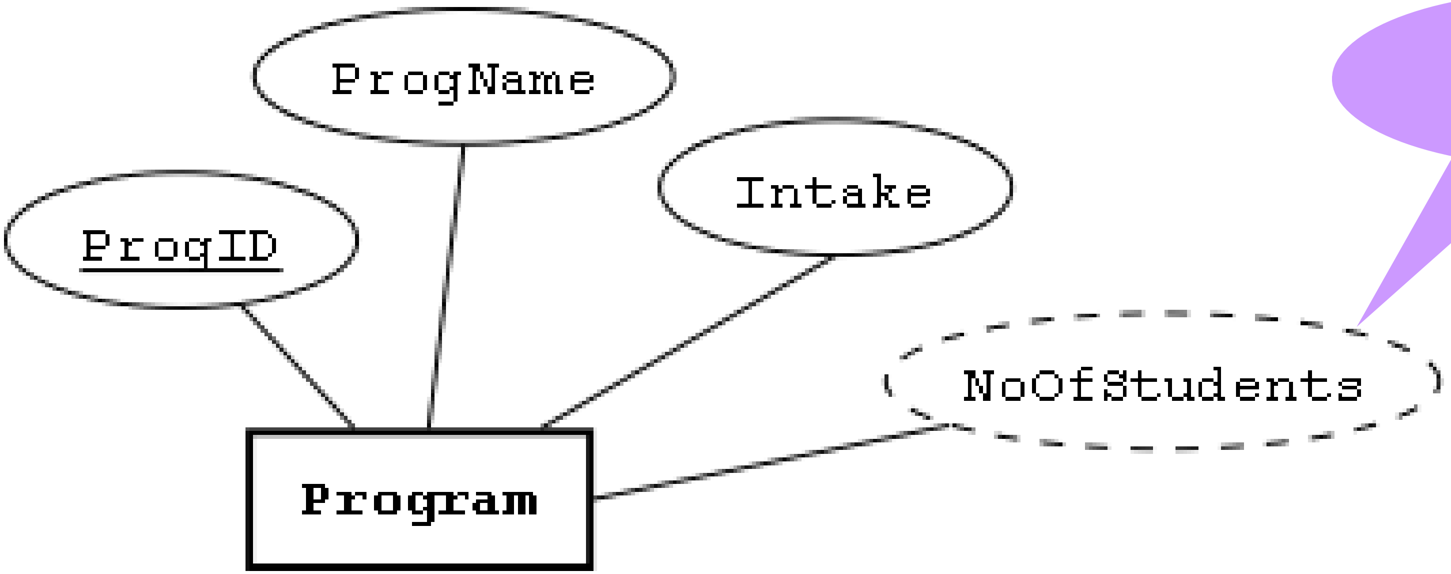
- Attributes could be
  - <u>Atomic</u> – can not be meaningfully decomposed: Example DOB can not be decomposed in Day/Month/Year – DOB is atomic data
  - Student-Name may or may not be atomic
    - If FirstName, LasteName have individual meaning
  - <u>Key</u> Attributes
  - <u>Composite</u>: Example Address (Street, City, PIN)
  - <u>Single valued and multi-valued</u>
  - <u>Stored and Derived attributes</u>

Entities and Attributes in ERD



Key attribute Uniquely Identifies an Entity in Entity Set

Entities and Attributes



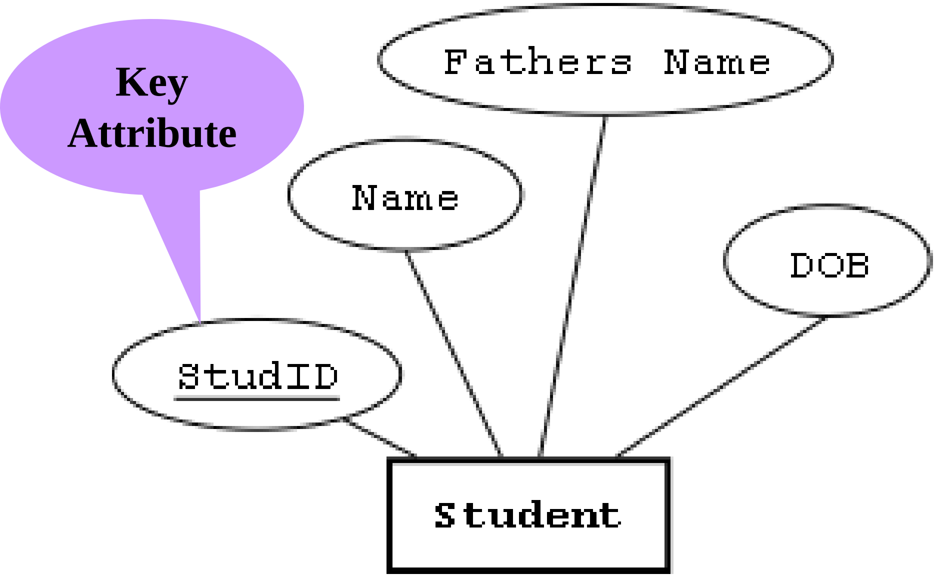
Derived Attribute

Terminology: Entity Type and Entity

- Casually, very often term “entity” is used for “Entity Type”.
- An *Entity Type* describes schema or intention for a set of entities sharing the same structure.
- In OO terminology, an entity type is similar to a class, and an entity is similar to an instance

Key Attributes

- It is Attribute or set of Attributes, called Key Attribute which are used to uniquely identify the entity in the set
- An important constraint on entities of an entity type is the key constraint or uniqueness constraint.



Summary – Entity Type, Entity, and Entity Set

- Entity Type: Description of an Entity. For example:
  Student(ID, Name, Batch, CPI)

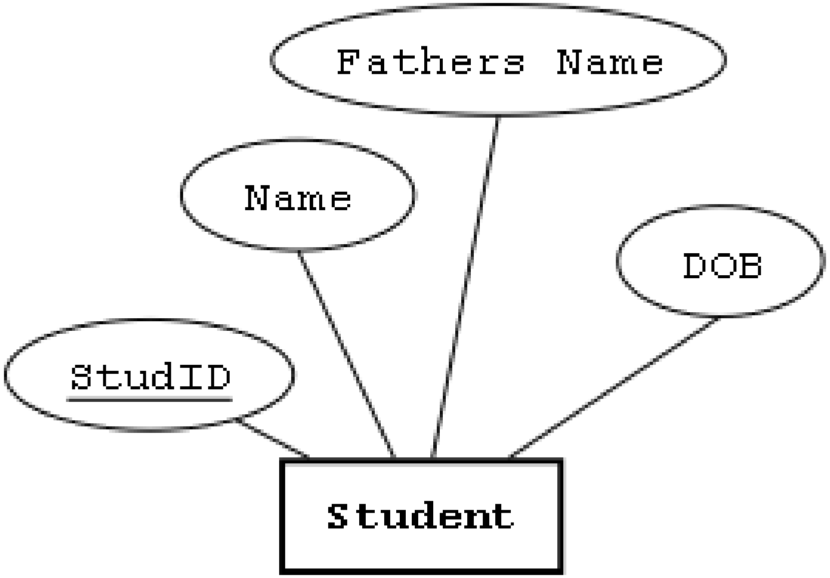
- Entity: for example a student
  {ID:200701001, Name:Charu Chawla, Batch:2007, CPI:6.12}

- Entity Set: set of all entitities belonging to a type, for example all students

Entity-Set

```
{
{ID:200701001,Name:Charu,Batch:2007, CPI:6.12},
{ID:200711002,Name:Amit Khanna,Batch:2007,CPI:7.12},
{ID:200711003,Name:Kamla Kiran,Batch:2007,CPI:7.50},
{ID:200711004,Name:Raj Kumar,Batch:2007,CPI:4.00},
{ID:200711005,Name:Raj Tiwari,Batch:2007,CPI:5.56},
{ID:200811001,Name:Rama Kant,Batch:2008,CPI:8.12},
{ID:200811002,Name:Akshya,Batch:2008,CPI:9.22},
{ID:200811003,Name:Unnati Gupta,Batch:2008,CPI:5.52}
}
```

Summary – Entity Type, Entity, and Entity Set



Entity Set with entities having Multi-value attributes

```
{
{ID:200701001,Name:Charu,Batch:2007,
   email:{charu@gmail.com,
  charu_x@abc.in},CPI:6.8},
{ID:200711002,Name:Amit Khanna,Batch:2007,
   email:{amil@yahoo.com}, CPI:7.12},
{ID:200711003,Name:Kamla Kiran,Batch:2007,
  email:{},CPI:7.50},
...
}
```

Entity Set with entities having composite attributes

```
{
{ID:200701001,Name:{Fname:Charu,Minit:K,
  Lname:Chawla}, Batch:2007, CPI:6.8},
{ID:200711002, Name:{Fname:Amit,Minit:C,
  Lname:Patel},Batch:2007,CPI:7.12},
{ID:200711003, Name:{Fname:Kamla,Minit:S,
  Lname:Kiran},Batch:2007,CPI:7.50},
...
}
```

Observe semantics captured in following two representation of same entity

```
{ID:200701001, Name:{Fname:Charu,Minit:K,
  Lname:Chawla}, Batch:2007, CPI:6.8},
```

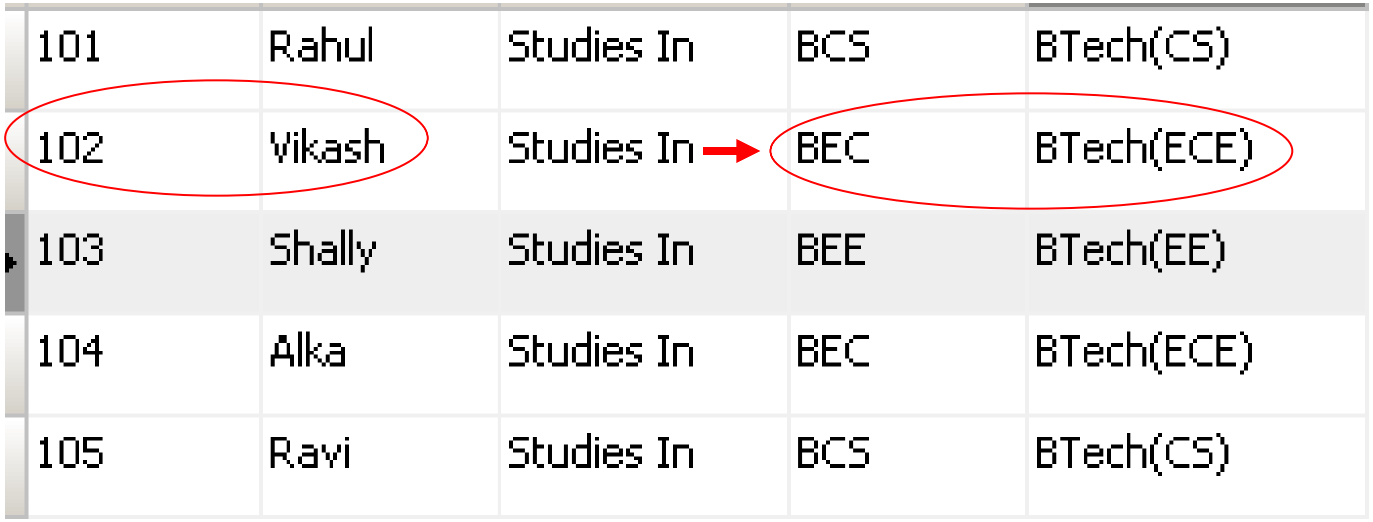
```
{ID:200701001, Fname:Charu, Minit:K,
  Lname:Chawla, Batch:2007, CPI:6.8},
```

- Next: Relationships.

Relationships

- Let us look at following facts –
  - Aditi *studies* in BIT
  - Amit *studies* in BCS
  - Arpit *studies* in BIT
  - Varun *studies* in BEE
  - Radhika *studies* in BIT
  - Shyam *studies* in BCS
- Note here Aditi here is name of Entity, read here Aditi as entity identified by Aditi, and similarly for other students and programs

Relationships



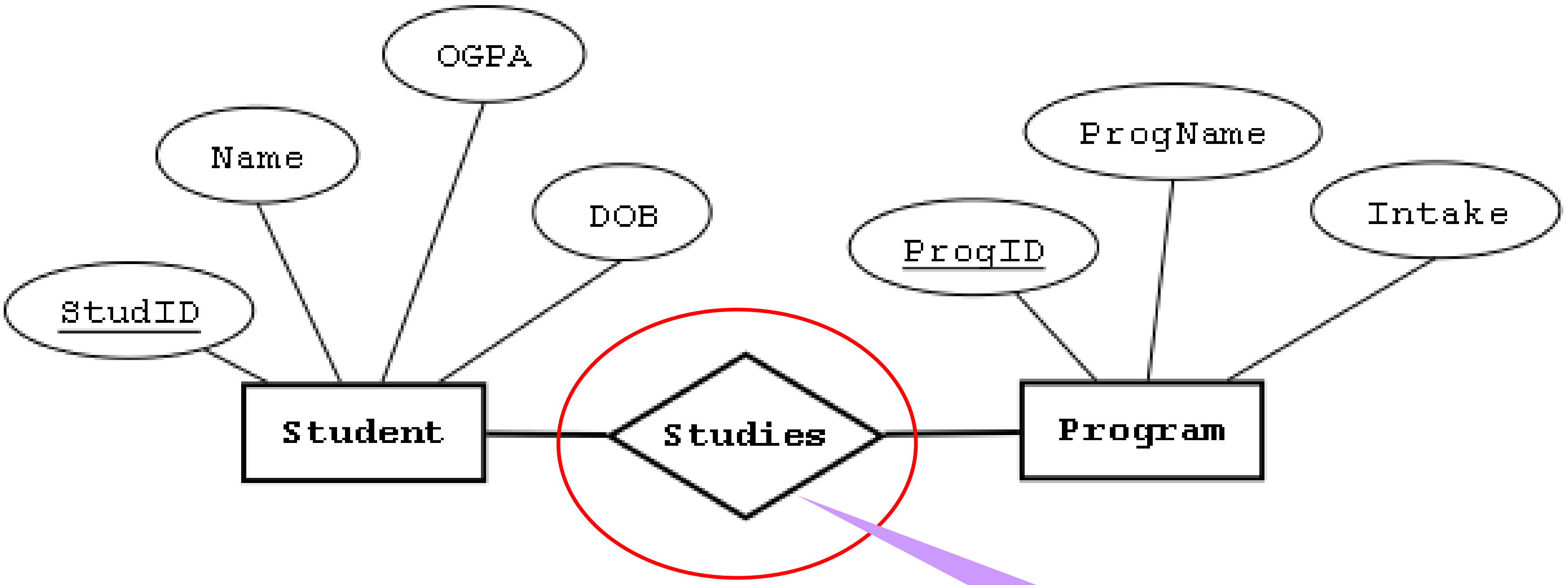
101	Rahul	Studies In	BCS	BTech(CS)
102	Vikash	Studies In	BEC	BTech(ECE)
103	Shally	Studies In	BEE	BTech(EE)
104	Alka	Studies In	BEC	BTech(ECE)
105	Ravi	Studies In	BCS	BTech(CS)

Relationships

- Let us look at following facts –
  - Aditi *studies* in BIT
  - Amit *studies* in BCS
  - Arpit *studies* in BIT
  - Varun *studies* in BEE
  - Radhika *studies* in BIT
  - Shyam *studies* in BCS
- This shows *Relationship* between Student and Program entities (entity types to be precise) … let us give a name to it "Studies-In" – obvious choice
- Relationship also has *type* and *set,* here studies is *Relationship Type*,
- Each of above statement is *instance* of the *type*, and All together form *relationship instance set*

Relationship [Type] in ER Diagram

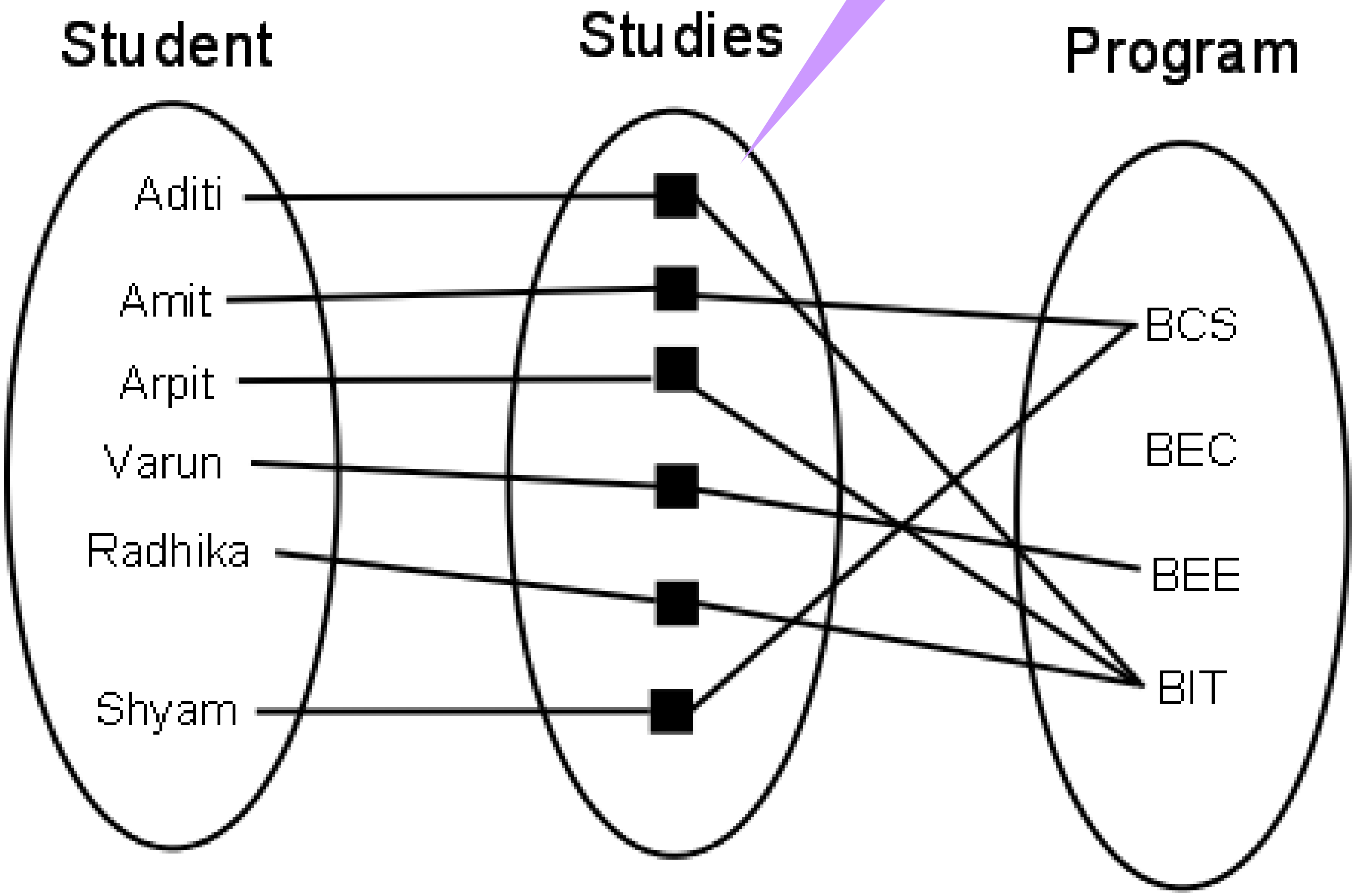
- Schema Diagram (ERD) tells that instance of one entity type is to be related with instances of other entity type



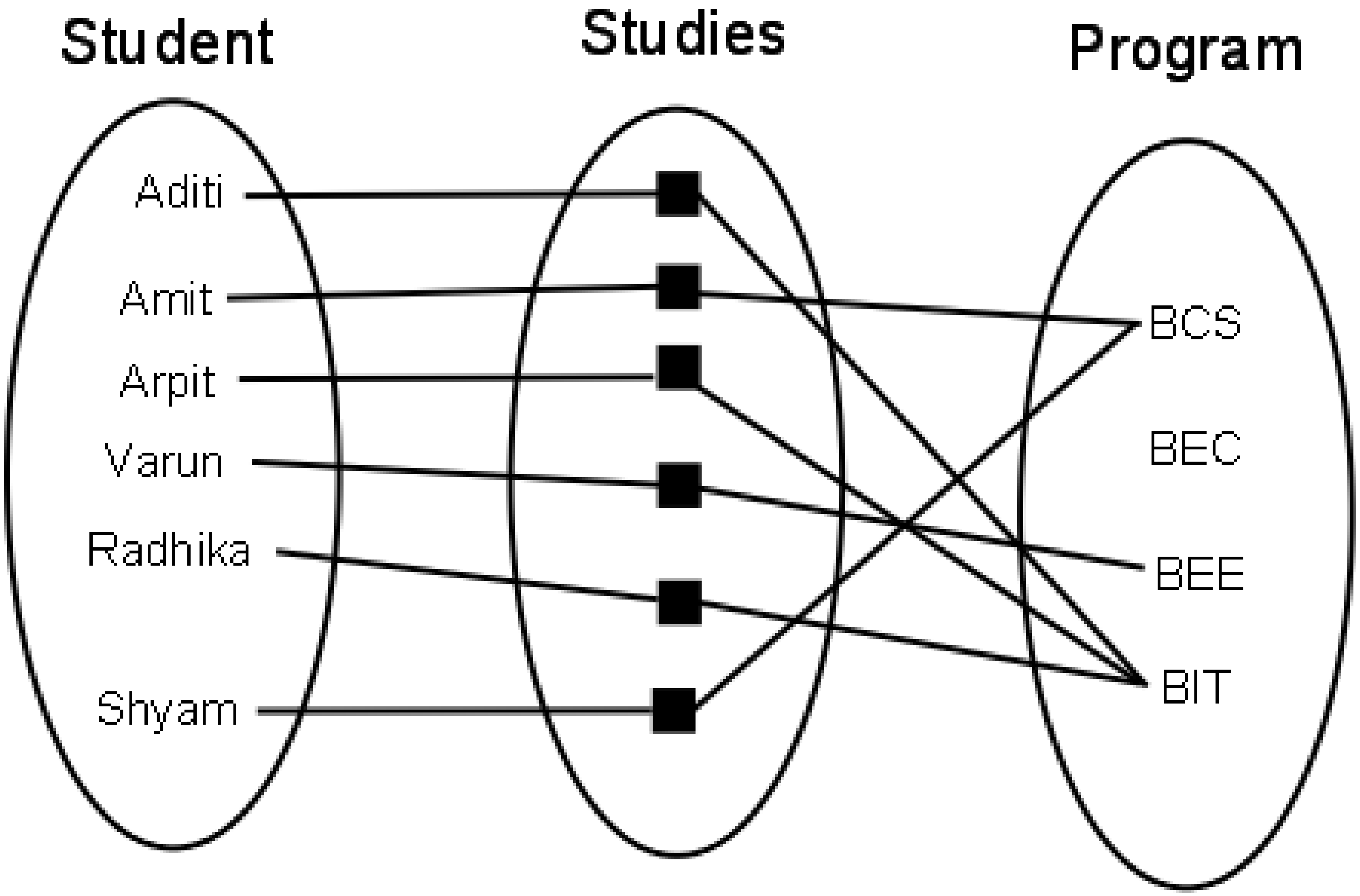
Relationship

Relationship Instance Diagram

Relationship Set

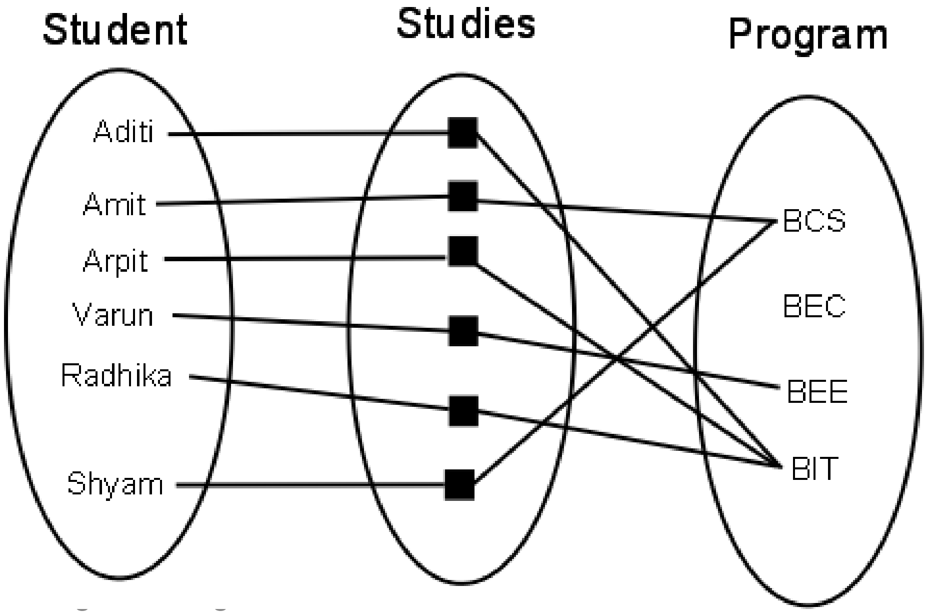


A relationship instance is identified in its set by key attributes of participating entities



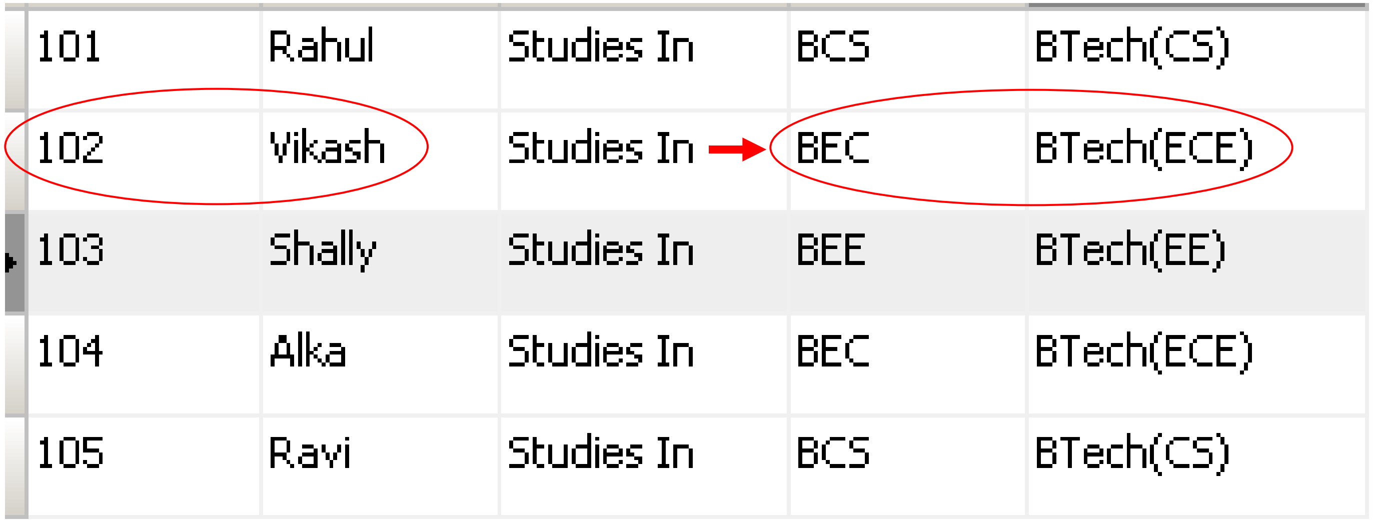
Relationship - instance

- When you say *Aditi studies in BIT*, it is one instance of studies relationship; and
- It is interpreted as “a student entity identified by Aditi is related with a program entity identified by BIT through studies relationship”.



Relationship - instance

- Example: entities related to another entities

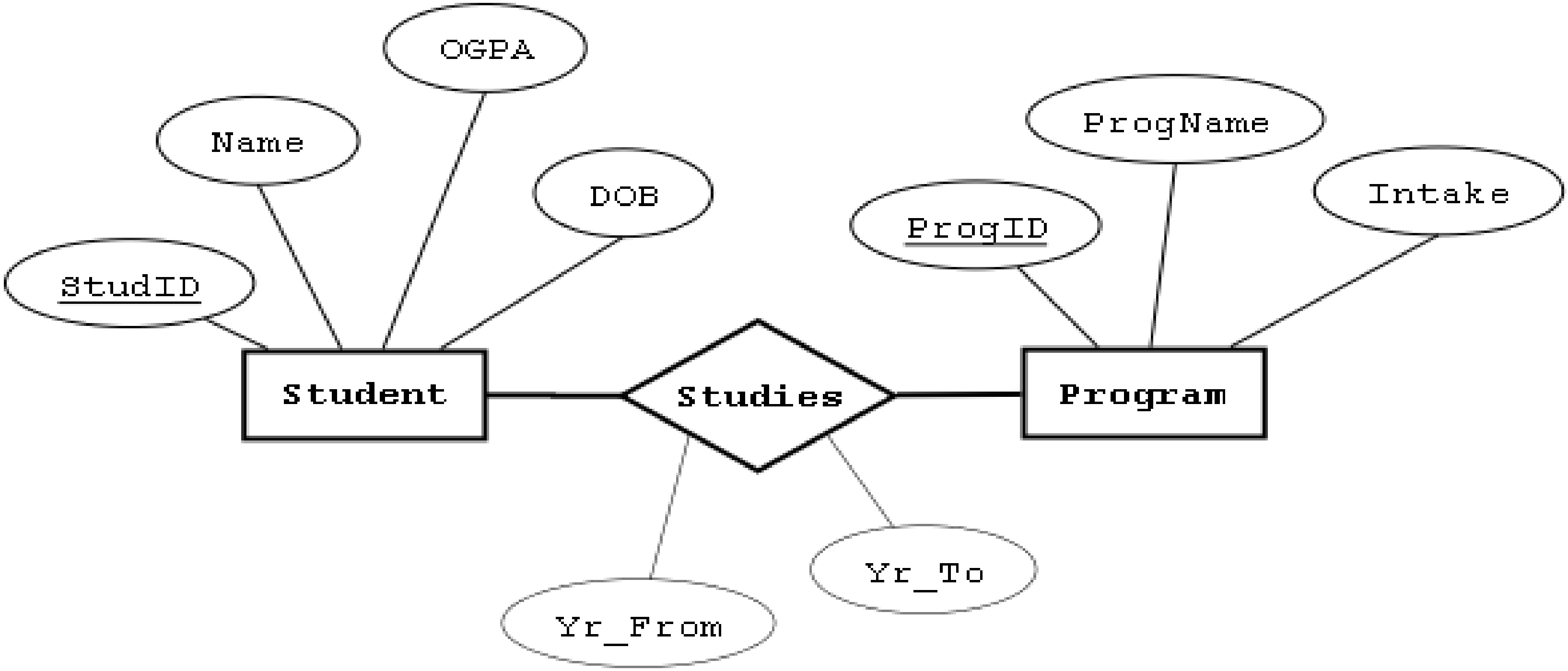


101	Rahul	Studies In	BCS	BTech(CS)
102	Vikash	Studies In	BEC	BTech(ECE)
103	Shally	Studies In	BEE	BTech(EE)
104	Alka	Studies In	BEC	BTech(ECE)
105	Ravi	Studies In	BCS	BTech(CS)

- When an entity is associated with other entity in the relationship type, then it is said to be participating in the relationship

Relationships can also have attributes

- In our previous example, suppose we also want to have study duration of a student in a program, then
- Let us say, we have attributes year_from, and year_to



Can you create relationship instance diagram for this?

- Cardinality Constraints in ERD

Examples: Cardinality Constraints

- A student can study in only one program, where as a program can have any number of students
- An Employee works for only one department, where as a department can have any number of employee.
- An Employee can work on any number of Project, and a Project can have any number of employees working on it.
- ==>These are examples of “business rules” that are specified as part of schema.

Cardinality Constraints in ERD

- Cardinality Constraints are specified to one of following -
  - One to One (1:1)
  - One to Many (1:N)
  - Many to Many (M:N)

- Cardinality Constraints are also called Cardinality Ratio (Elmasri/Navathe), though not ratio in precise terms. Probably, because it written in ratio format (1:N)! We will use the term *Cardinality Constraints*

Cardinality Constraints in ERD

- One to one (1:1): When entities of both types can be related to atmost one from entity of other type
- For example, **Manages** relationship in company schema: One department can have atmost one employee as manager, and one employee can be manager of at-most one department
- 1:1 is not very common type of relationship.

Cardinality Constraints in ERD

- One to Many (1:N): When entities of one types can be related to atmost one from entity of other type, and other can be related to any number (N) of entities from first type.

- For example, **`Works_For`** relationship in company schema: One employee works for only one department, but one department can have any number of employees.

- Other examples are Student Studies Program, and Department Offers Programs in XIT schema.

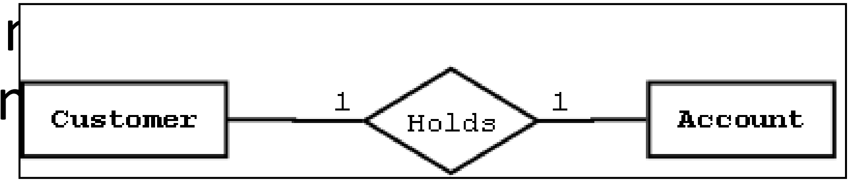
Cardinality Constraints in ERD

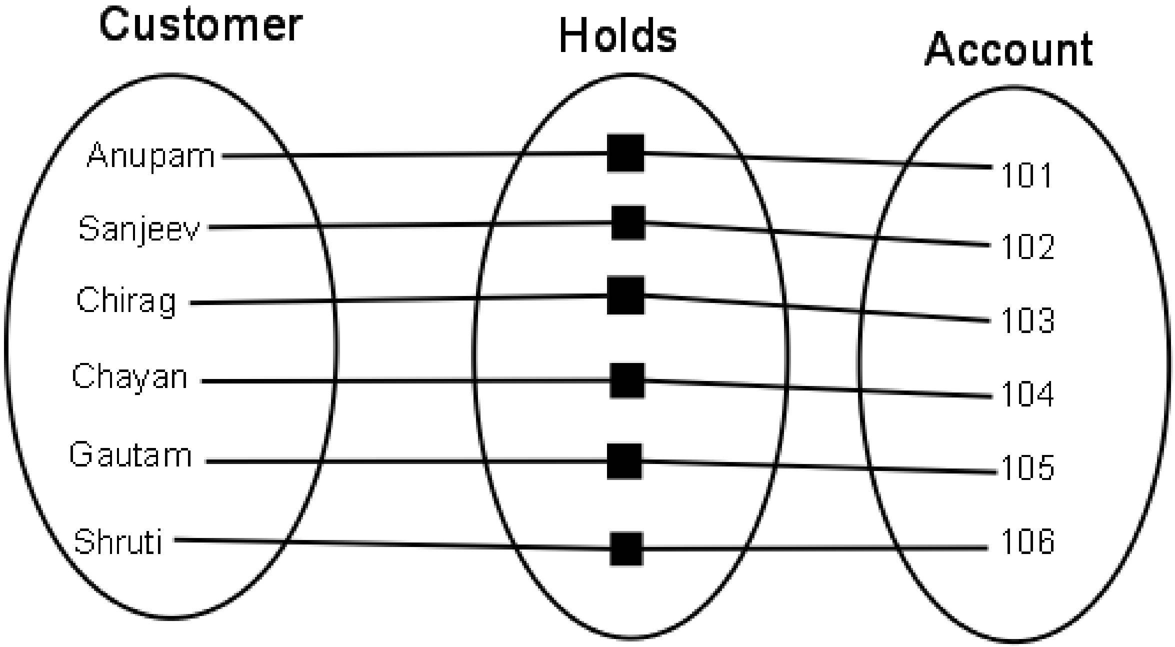
- Many to Many (M:N): When entities of both types can be related to any number (N) of entities from other type.

- For example, **`Works_On`** relationship in company schema: An employee can work on any number of projects and a project can have any number of employees working on.

One to One (1:1)

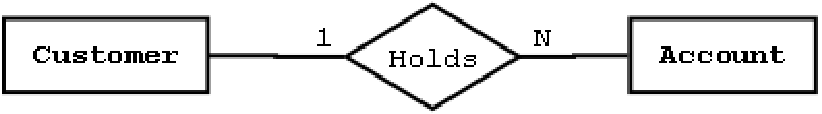
- One Customer has one account and one account is owned by one customer
- In other words: a customer er
account entity, and an accour
customer entity

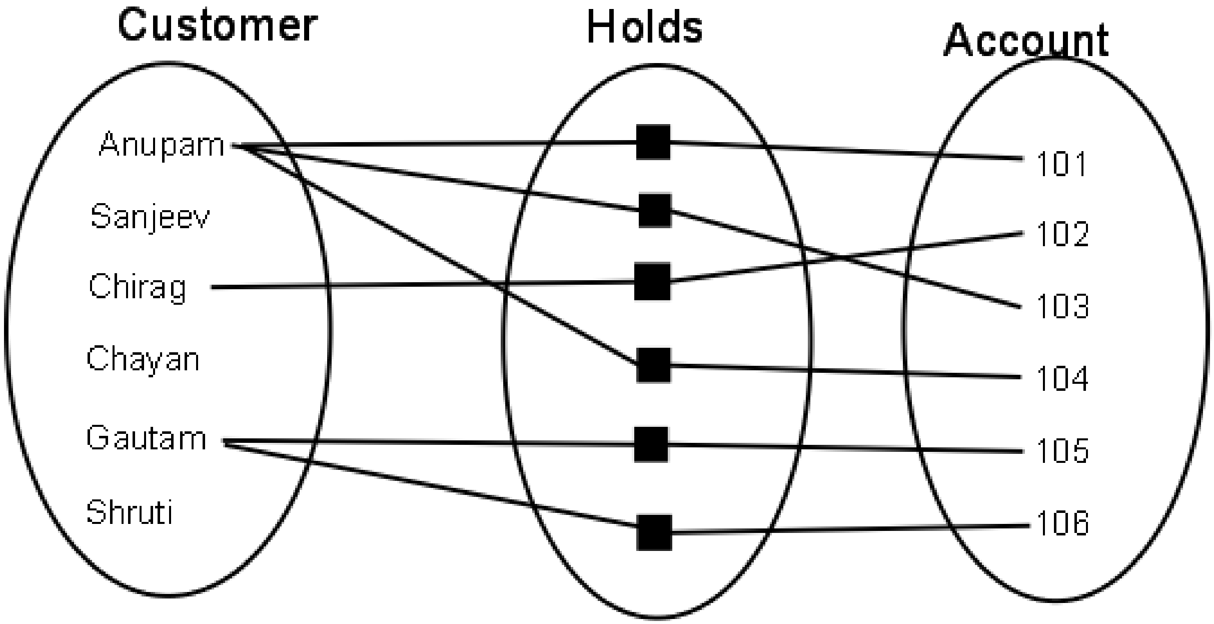




One to Many (1:N)

- One Customer has many accounts, and an account is owned by one customer
- In other words: A customer e entities, while an account ent customer entity





One to Many (1:N)

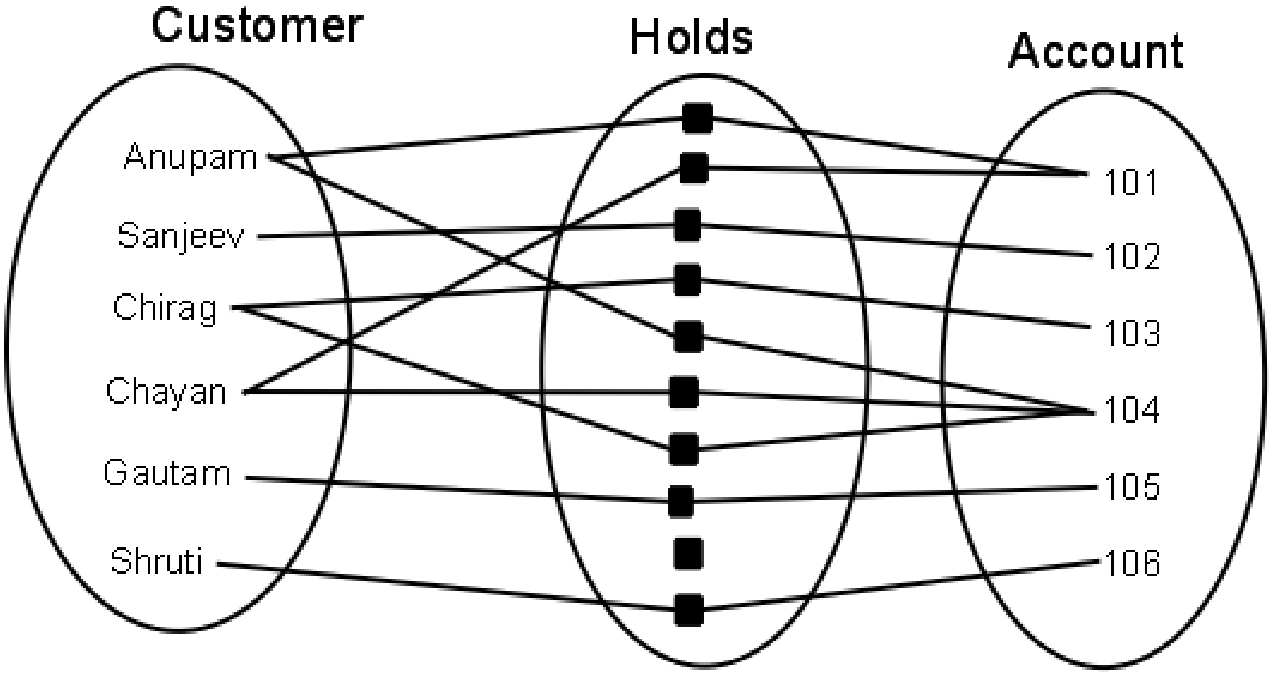
- Reverse of previous is also 1:N; i.e. One Customer can have only one accounts, but one account is owned by multiple customers

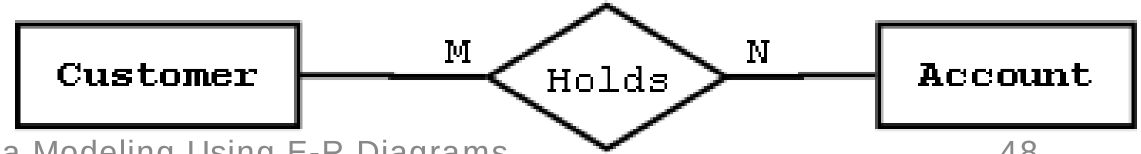
- In other terms: a customer entity can relate to atmost one account entity, while an account entity can be related to many customer entities

- Some people, though, call it many to one relationship (N:1), but this is just saying of same thing (1:N) other way round – just flip positions of entities !

Many to Many (M:N)

- One Customer has many accounts, and an account can be owned by multiple





Identify cardinality constraint

- Bill has Item(s)
- Student has one of Professor as Mentor
- Course uses Text Book(s), and one text book could be used in multiple courses
- A city is in a Country, and Each Country has one of its city as its capital

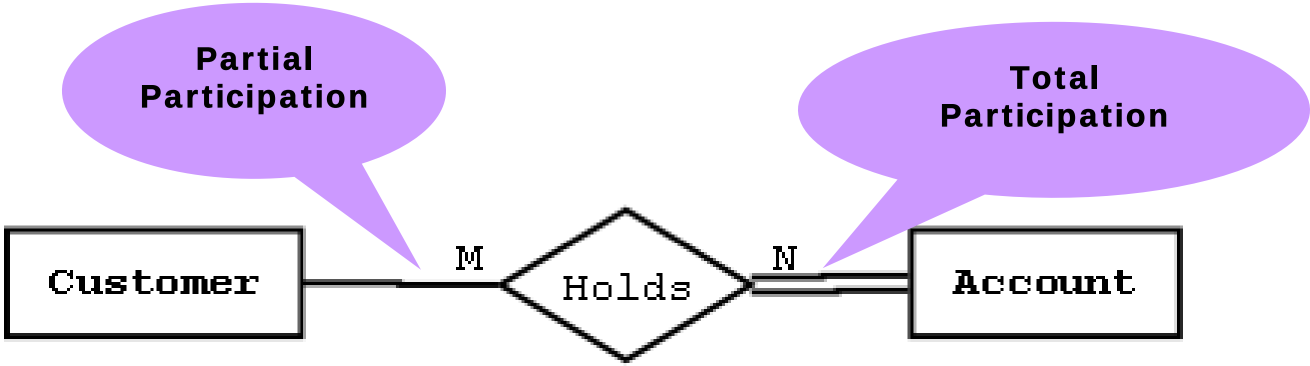
- Participation Constraints

Participation Constraint

- Participation Constraint can be Total (mandatory) or Partial (optional)
- If participating of an entity in a relationship is mandatory, then it is total participation otherwise, it is partial.
- For example, consider following
  - Can a Student Entity exist without related with Program or without participating in studies relation.
  - Can Employee exists without associated with a department as an employee
  - Can an employee exist without becoming manager of some department
- If answer of any question above is yes, then participation of the entity in the relationship is Partial, otherwise it is Total

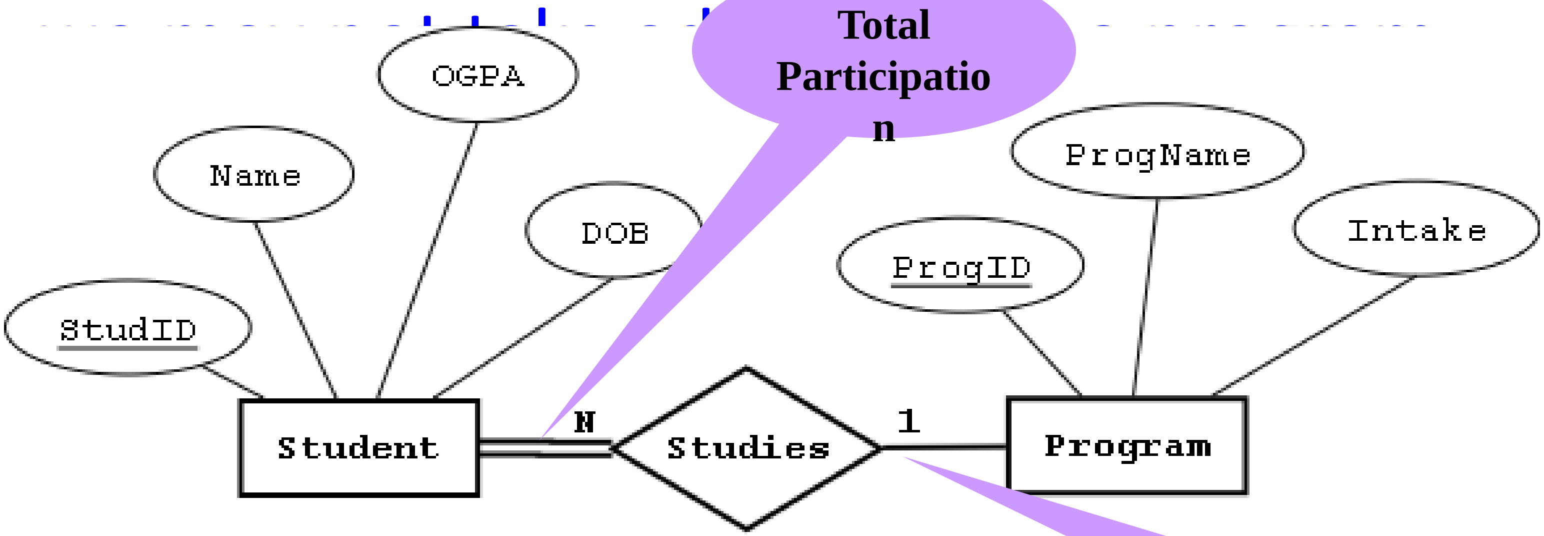
Participation Constraints

- In our Customer-Account example-
- If customer record can exist in database without having associated with an account – then participation of customer is partial otherwise total.
- Can a customer exist without having (associated with) an account?
- Can an account exist without having (associated with) a customer?



Student-Program "schema"

- Participation of Student in Studies relationship is mandatory – student has to study in a program.
- Participation of a Program can be optional –



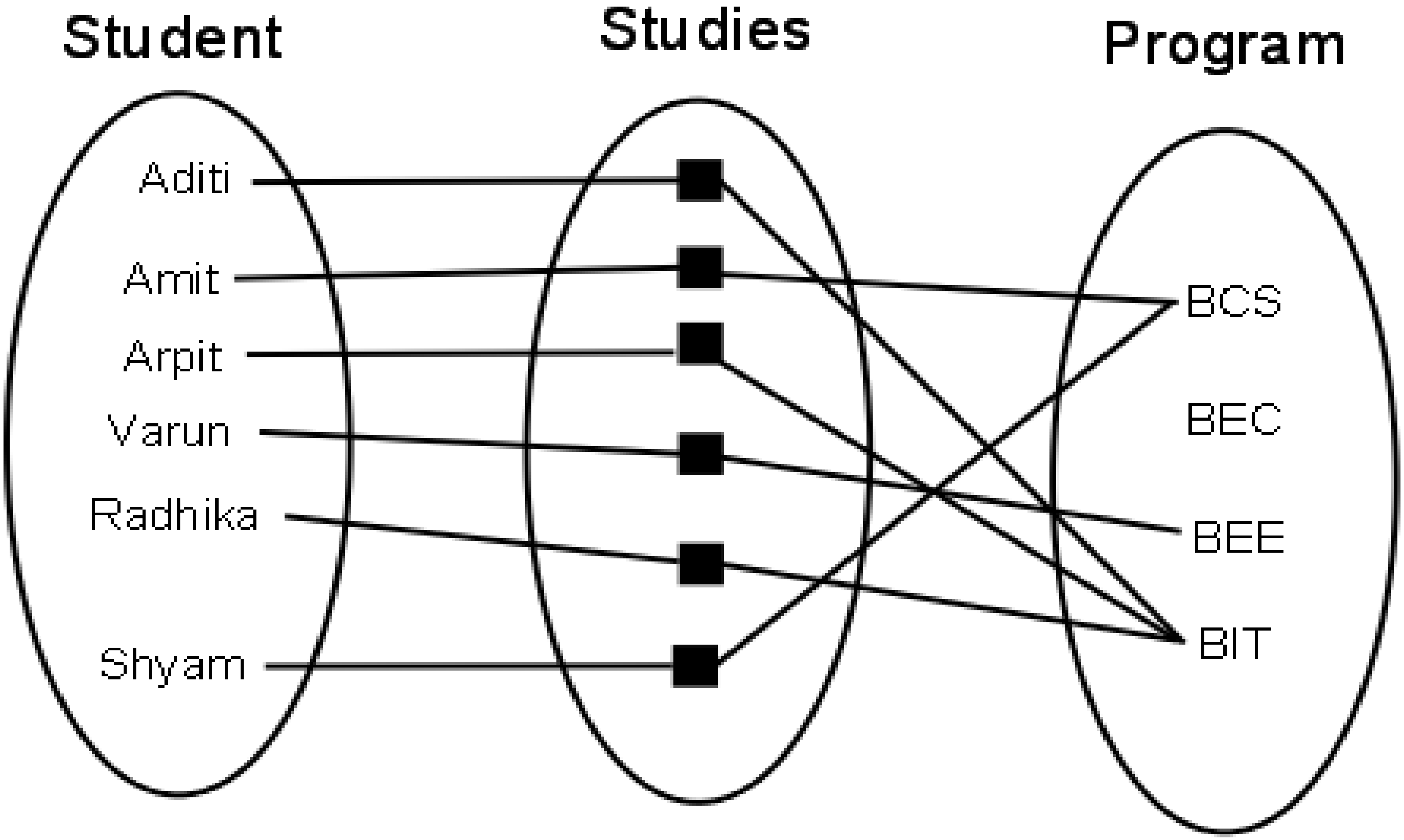
Total Participation

Partial Participation

- Most of the time figuring out cardinality and participation constraints are easy but sometimes may become hard.

- Relationship Instance Diagram should helps in visualizing and figuring out these constraints

Relationship diagram should help in better visualization of Cardinality and Participation constraints



Exercise

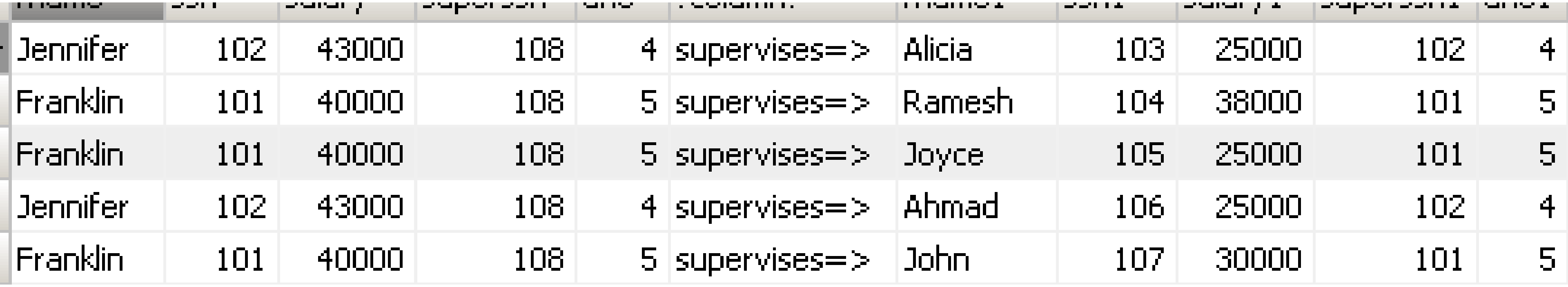
- ERD for Company Database (Elmasri/Navathe)

Recursive Relationship

- Entity having relationship with itself

- There can be many situations like this
  - Employee supervises another employee
  - One course has pre-requisite of another course
  - One category is parent of another category
  - One citizen is spouse of another citizen

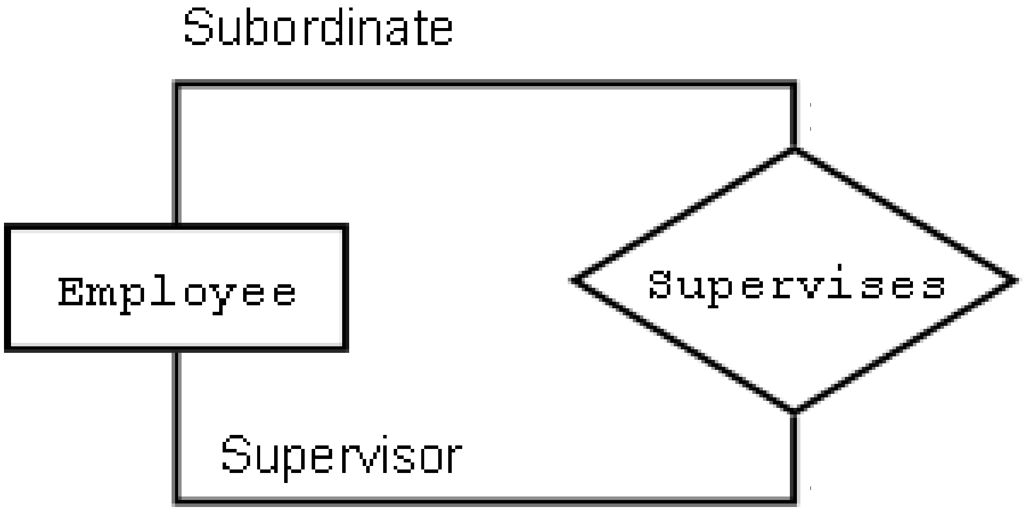
Recursive Relationship



Jennifer	102	43000	108	4	supervises=>	Alicia	103	25000	102	4
Franklin	101	40000	108	5	supervises=>	Ramesh	104	38000	101	5
Franklin	101	40000	108	5	supervises=>	Joyce	105	25000	101	5
Jennifer	102	43000	108	4	supervises=>	Ahmad	106	25000	102	4
Franklin	101	40000	108	5	supervises=>	John	107	30000	101	5

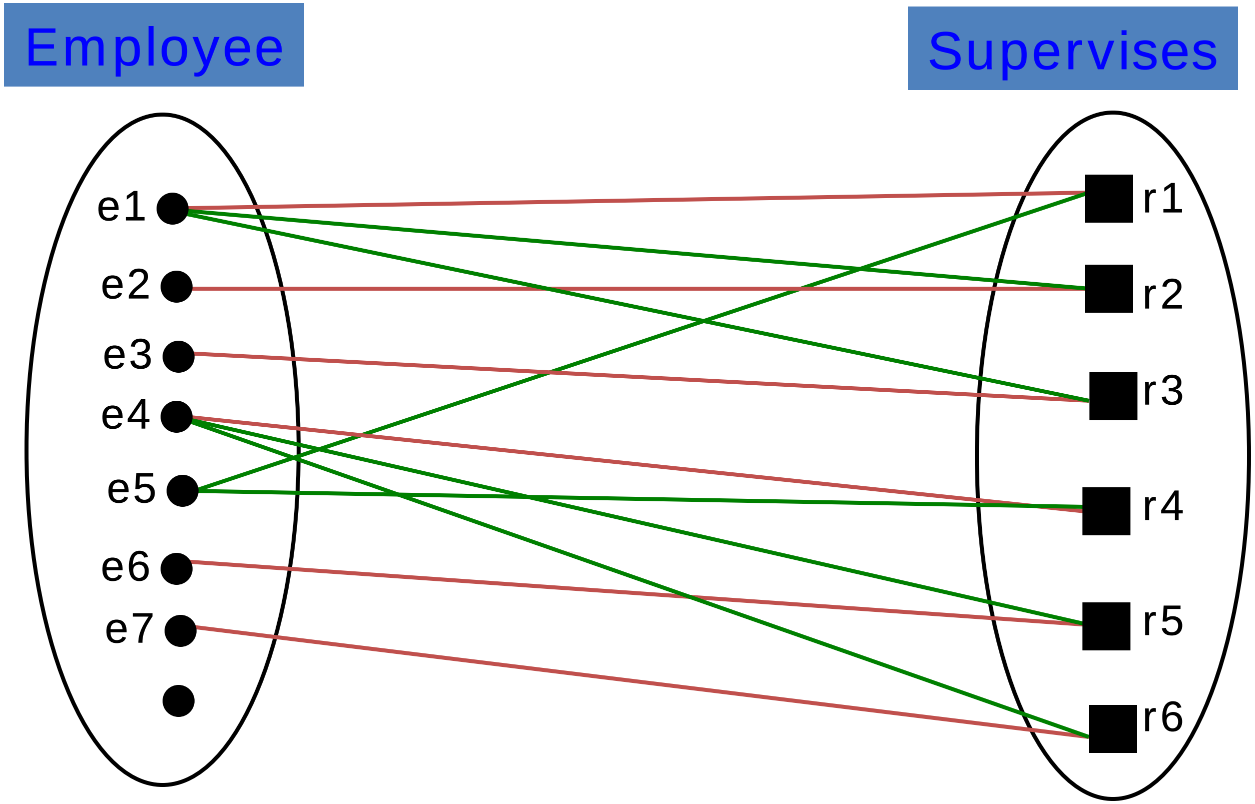
Recursive Relationship

- Employee supervises another employee



- Can you find out cardinality of this relationship?

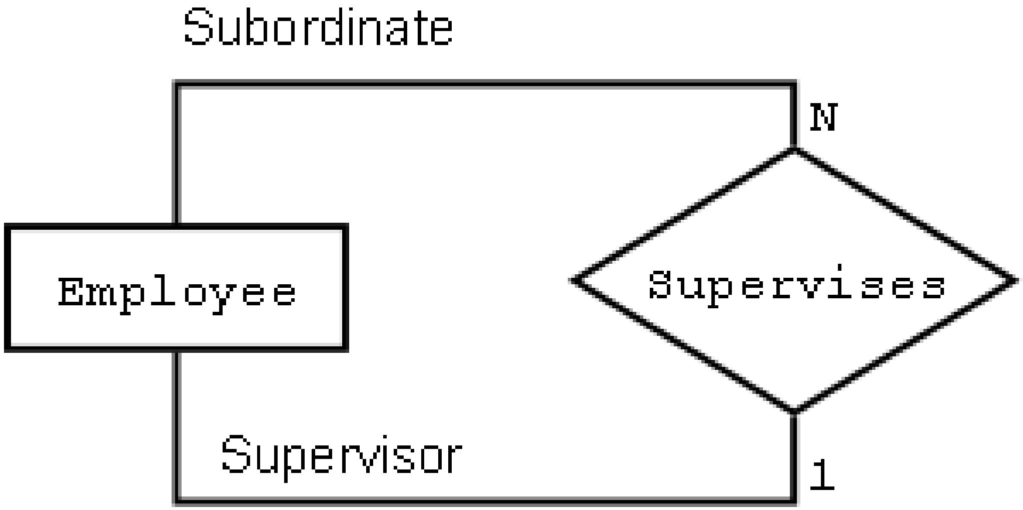
Instance diagram - Recursive Relationship



Roles: Supervisor (Green), Subordinate (Red)

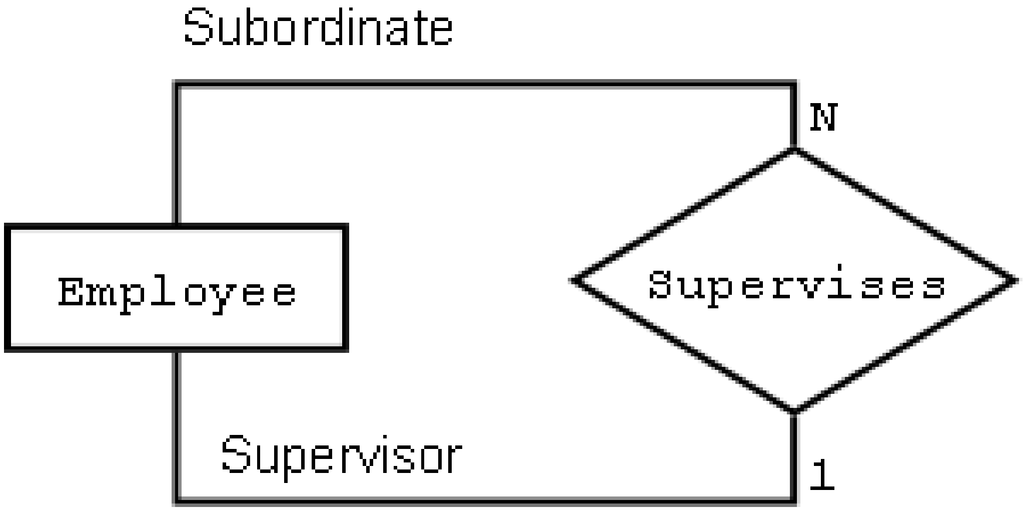
Cardinality constraints in Supervises relationship..

- Employee supervises another employee



Participation constraints in Supervises relationship..

- Employee supervises another employee



Weak Entity

- Entity that do not have key attributes of their own are called weak entity types
- Consider lecture rooms in your CEP? Is Room number is enough to identify a room ?

Weak Entity

- Another example: Consider a Football teams in a tournament.
- How do you identify a player? By Name, By Player Number? Is a weak entity?
- Team-Name followed by Player Name/Number would identify a player

Weak Entity Types

- Weak entities are attached to some regular/strong entity type, we call that entity as identifying or owner entity **type**, and we call the relationship between weak entity type and its owner entity type, identifying relationship**.**
- Week entities have partial key, that is, their key is unique within a given owner entity (room no within a builling, player name within a team, dependent within an employee)
- A weak entity type always has total participation in its identifying relationship.

Two parameters to look at while identifying weak entity

- Weak Entity does not have its own identifying attribute, require to be associated with other entity in order to make a weak entity to be identified
- Its existence is dependent on some other entity
- In most cases a entity is weak or regular is very much context dependent

Examples of Weak entity

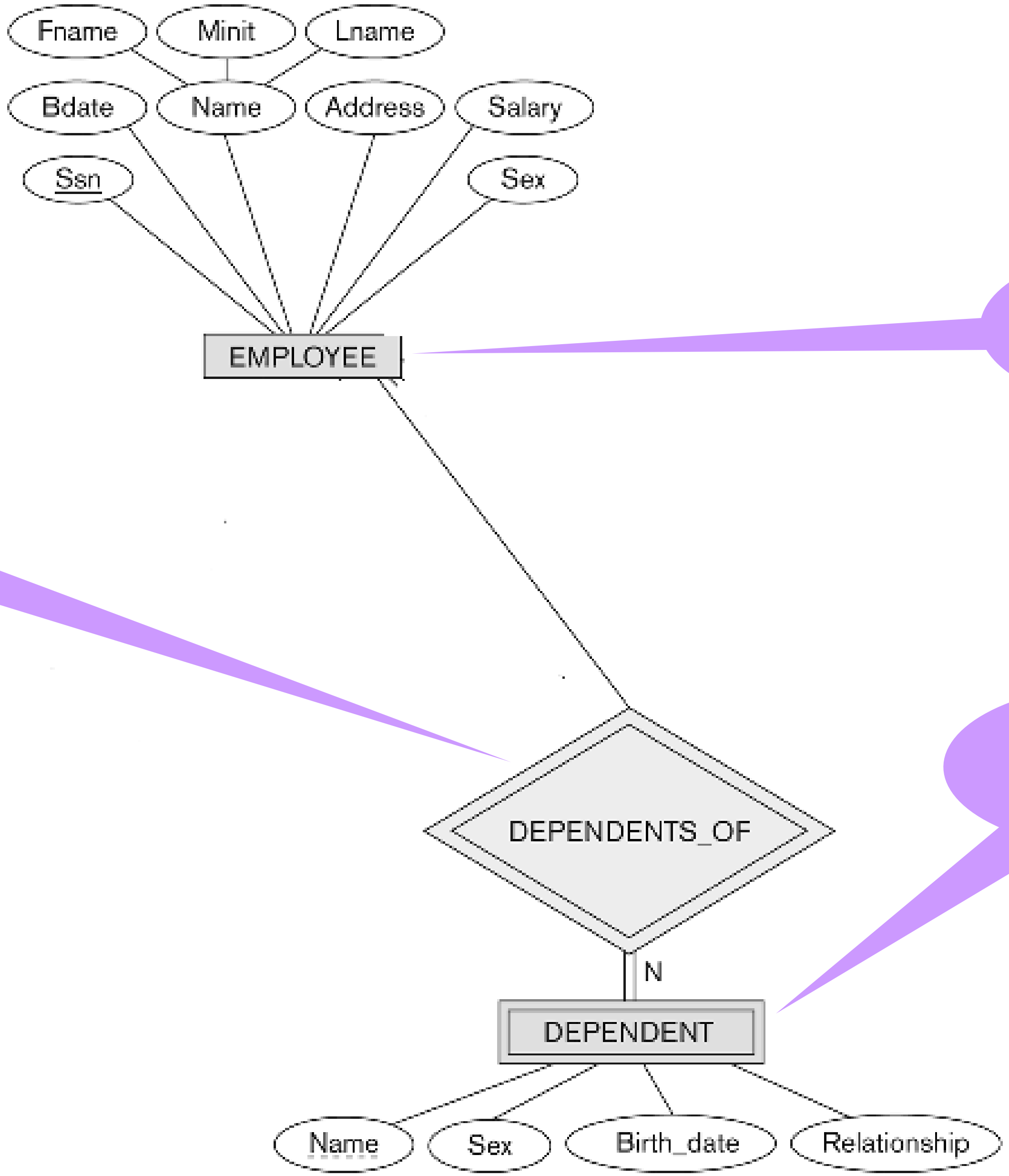
- An entity strong in one context might become weak in another context; rooms in a building could be strong entity, but if you mix rooms from different building then it may no more could be strong entity

- In DAIICT scenario course IT214 could be unique, but moment you mix course numbers from different institutions, course may become weak entity

More examples of Weak entity

- Pages of a book/magazine
- Transaction of an account
- Bill-Items in an invoice

Weak Entity Types in diagram



Owner Entity

Identifying Relationship

Weak Entity

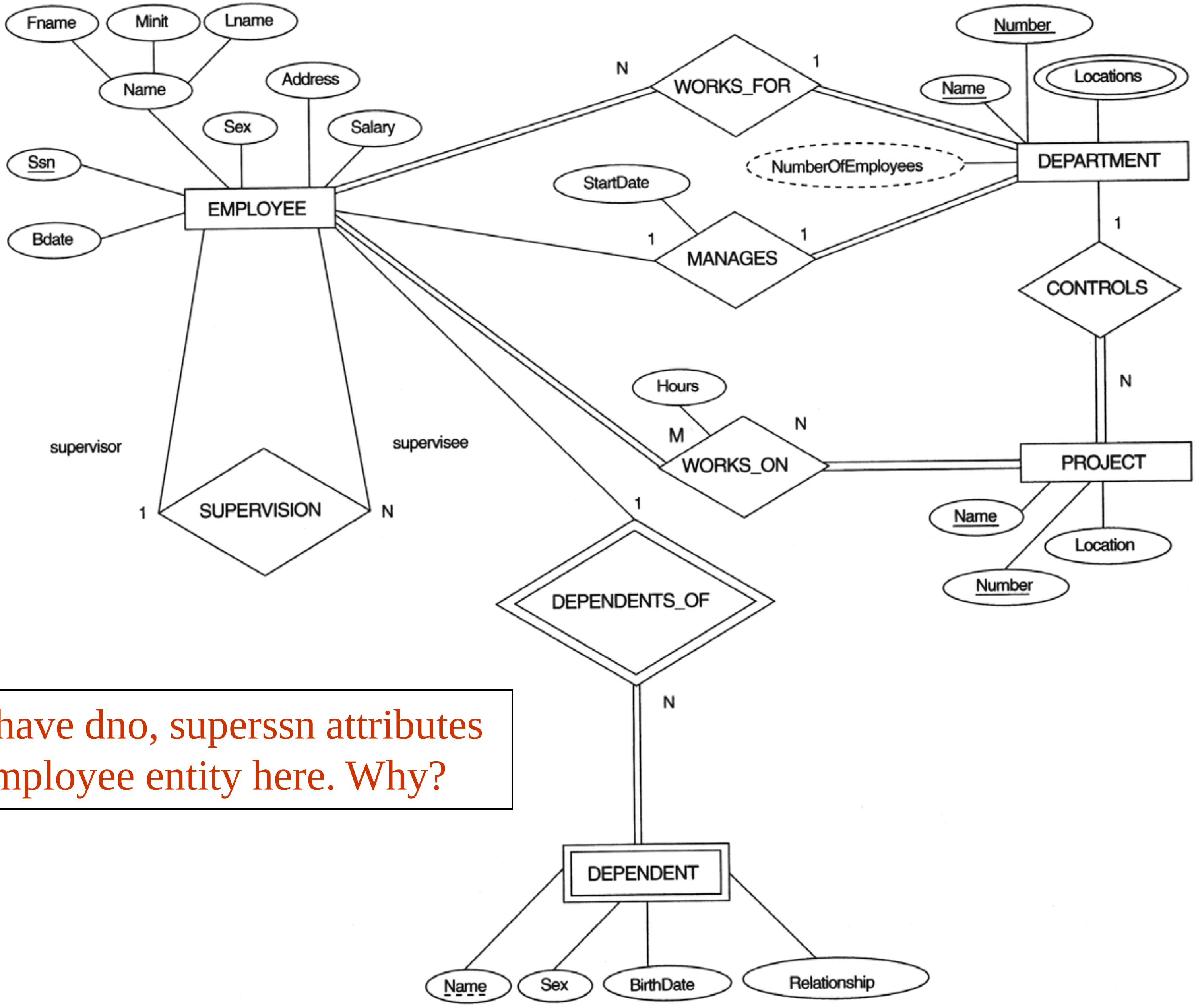
Weak-entity as “multi-value composite attribute”

- In some cases, database designer may choose to express weak entity as multi-value composite attribute of the owner entity type

- However, this is only possible if week entity does not participate in any other relationship than identifying relationships.

- Also, when loose meaning of partial key, when represented as multi-value composite attribute, therefore in most cases it is not advisable.

ERD: Company Schema



We do not have dno, superssn attributes [FKs] in Employee entity here. Why?

References in ERD

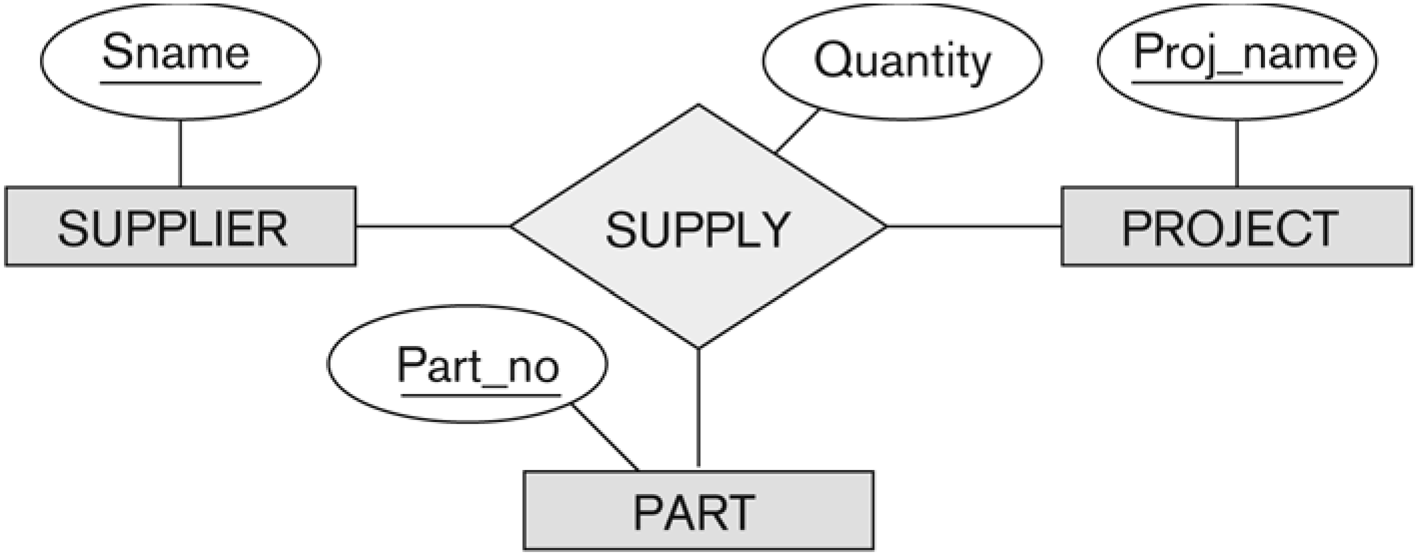
- We do not we have dno and superssn attributes in ERD where as these are there in company relational Schema?

- Dno and superssn in Employee relation are references to other associated entities (tuples)

- In ERD, these are shown by relationships. In example above Works_For and Supervises respectively are for the same purpose.

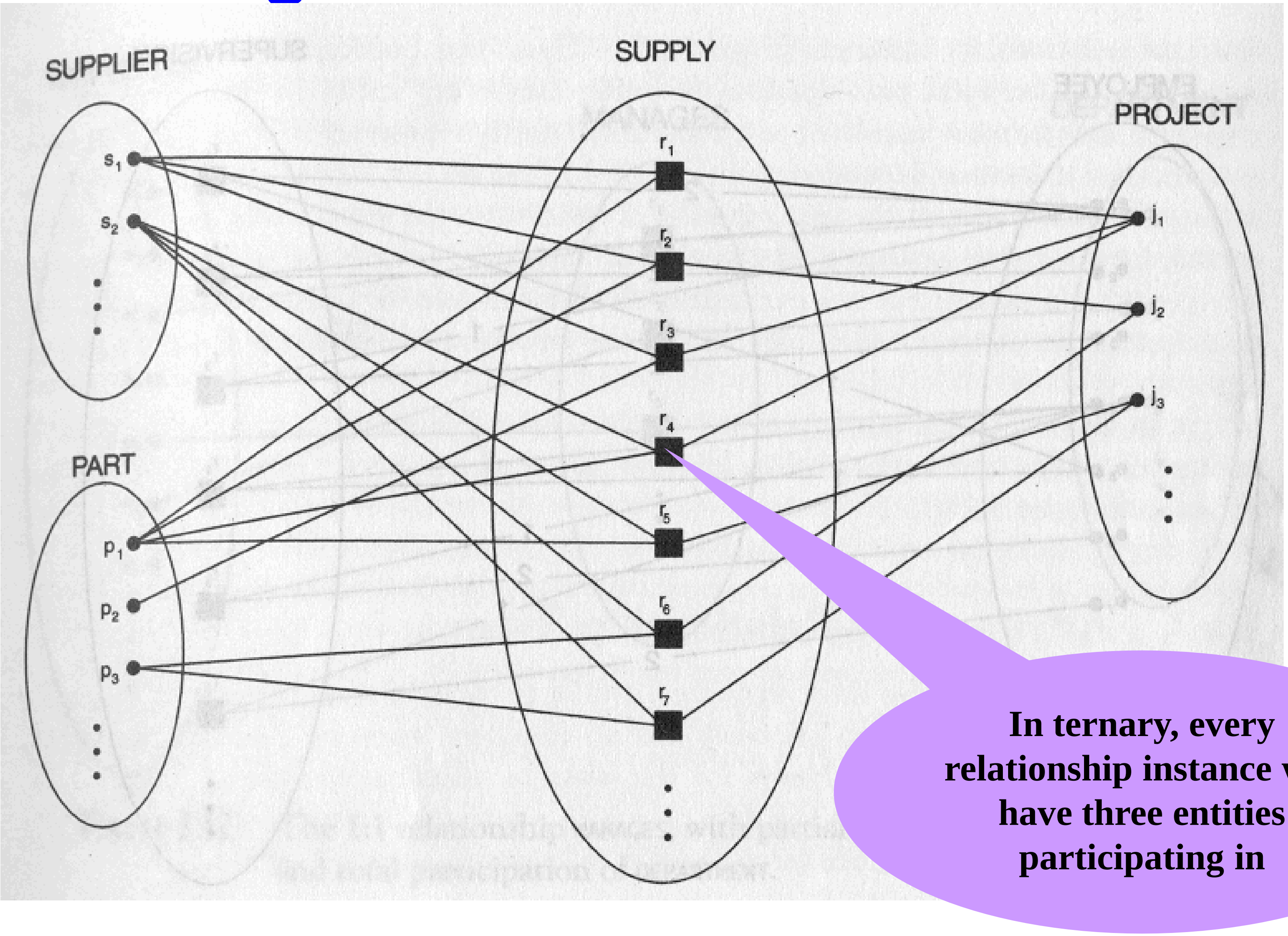
Ternary Relationships

- When you have three entities participating into a relationship.
- Here is a popular example (Project-Part-Supplier): A number of suppliers supply parts to projects. Any part may be supplied by any supplier to any project in any arbitrary manner.



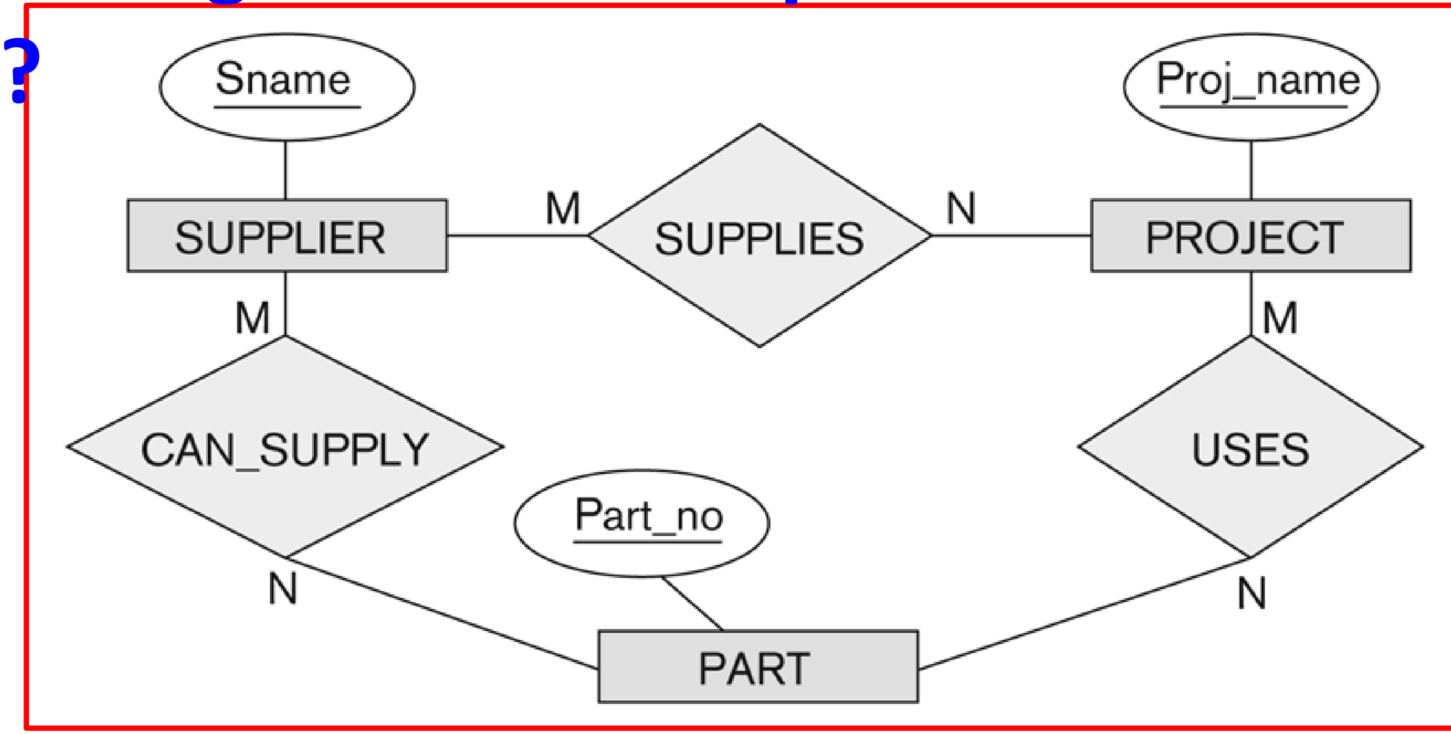
- Ternary relationship are not very common in practice, however!

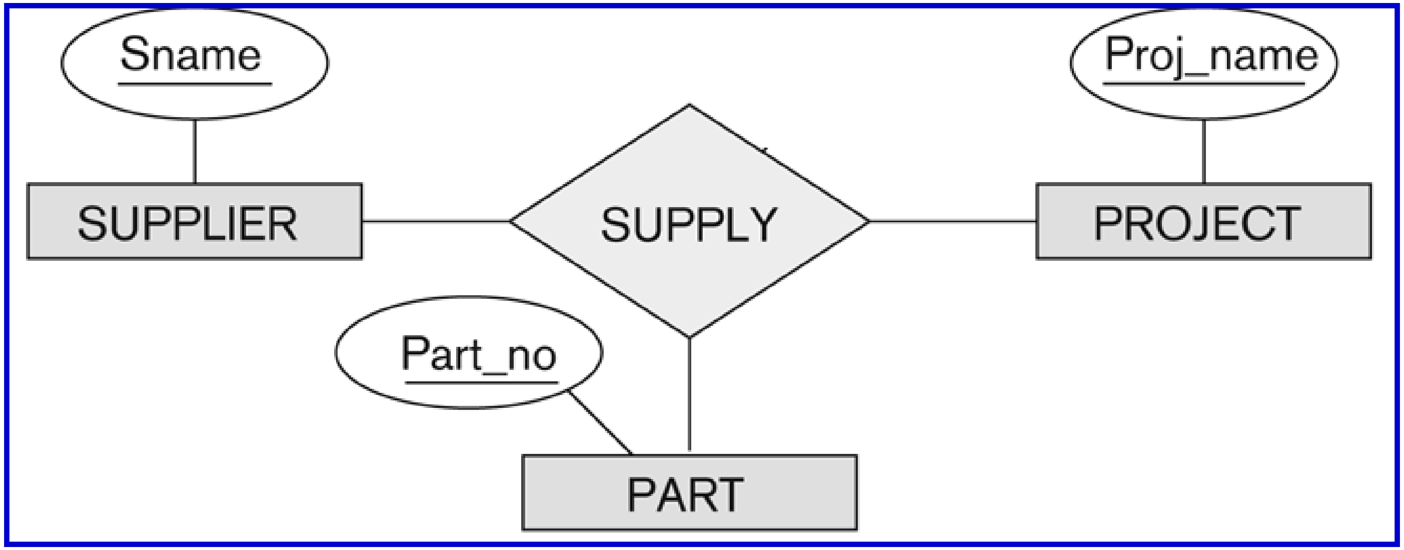
Ternary Relationship – instance diagram



In ternary, every relationship instance will have three entities participating in

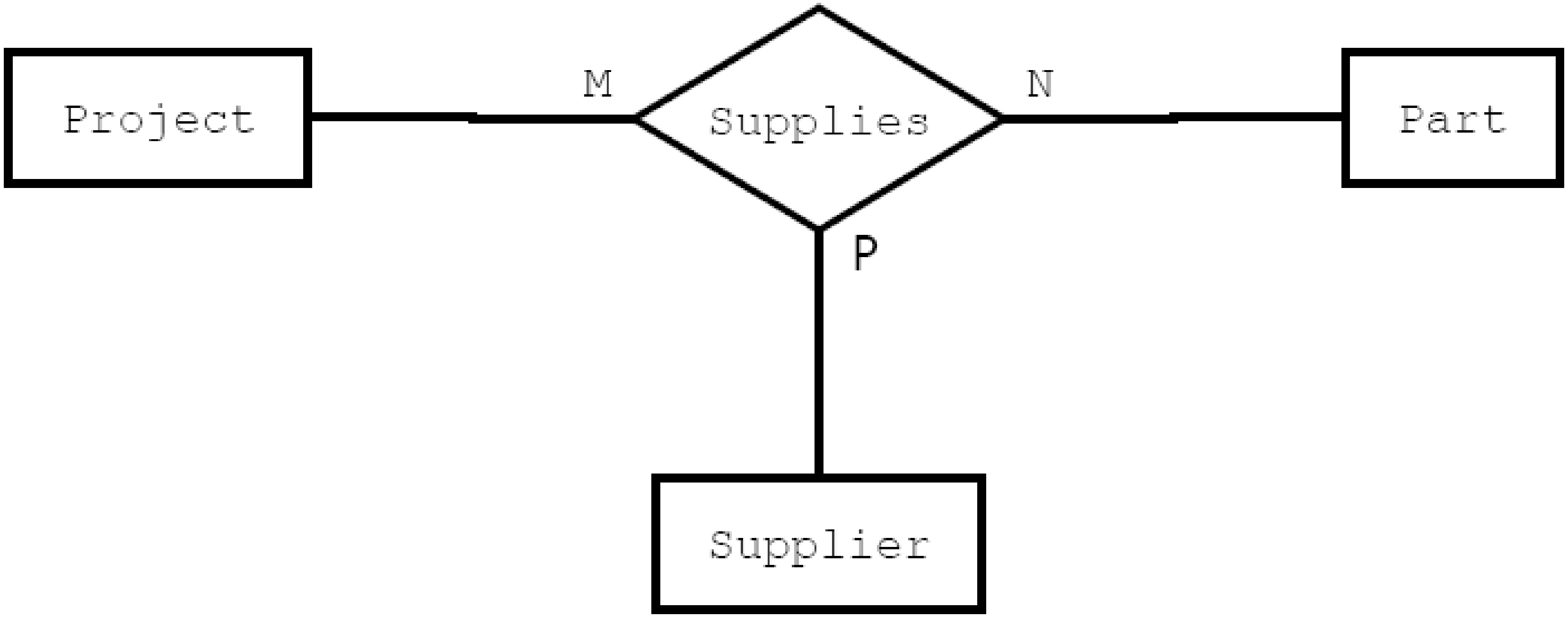
Do the diagram below represent same thing?





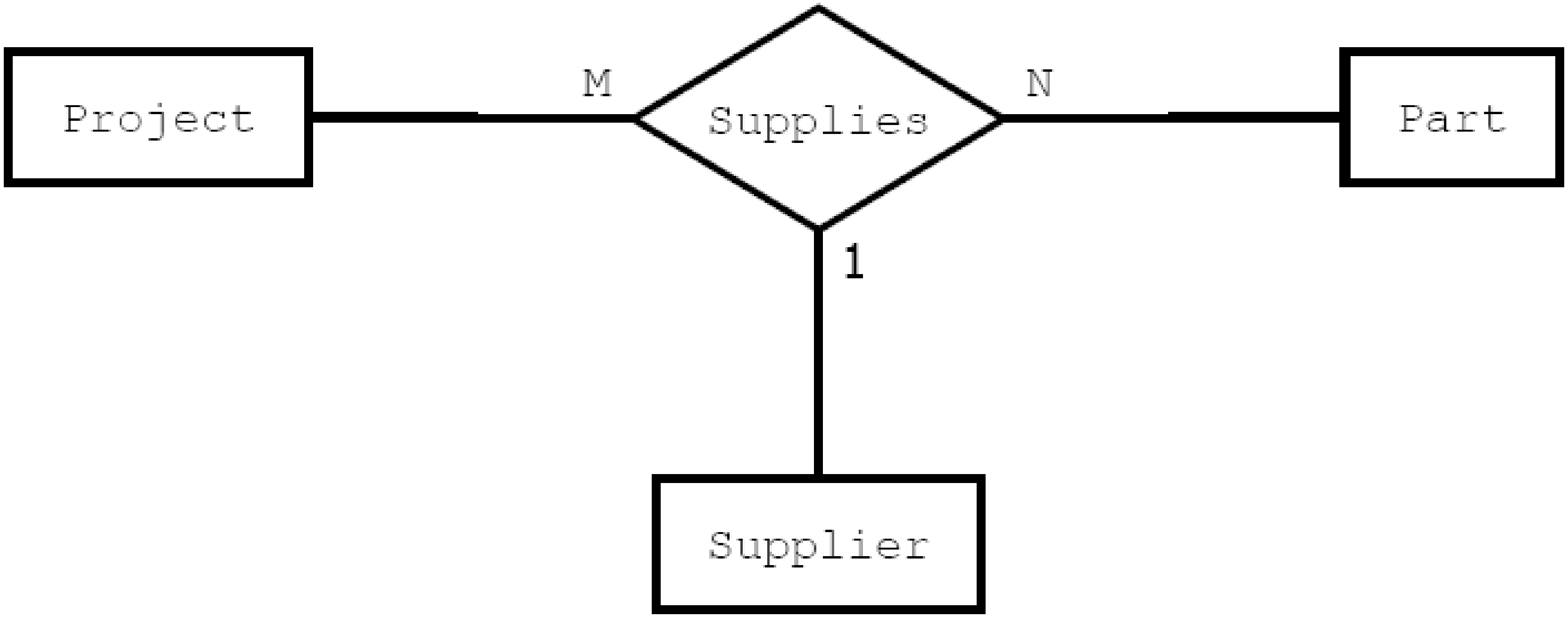
Cardinality Constraints in Ternary

- Here is one of the approach suggested by Elmasri and Navathe
- Here, cardinality is specified as number of corresponding entities from third entities for one distinct pair of other two



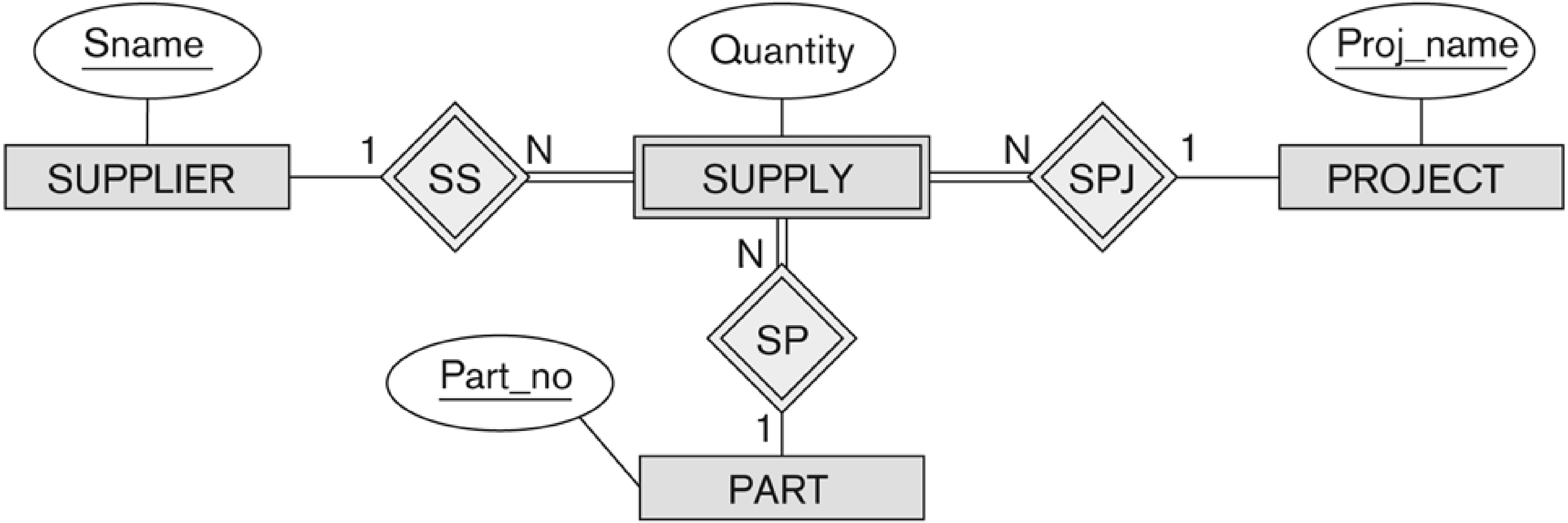
Cardinality Constraints in Ternary

- If we have constraint that there will be only one supplier of a part to a project. That is one distinct pair of project and part will have only one supplier entity



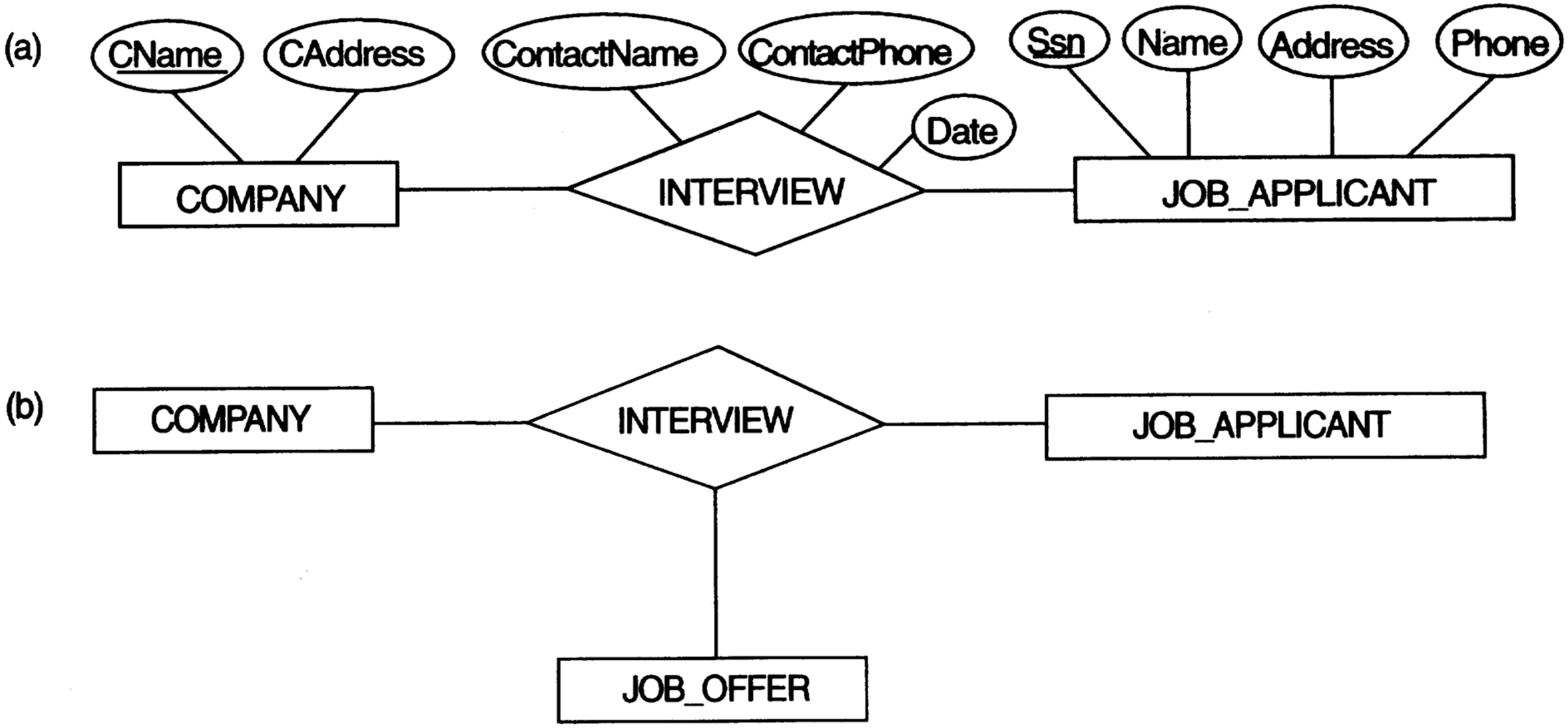
Ternary Relationships can always be representing as weak entity identified jointly by three owner entities (as shown in diagram below).

However this is less intuitive and adds complexities



Aggregation

- Let us take an Example Company-Interview-Applicant-Offer situation

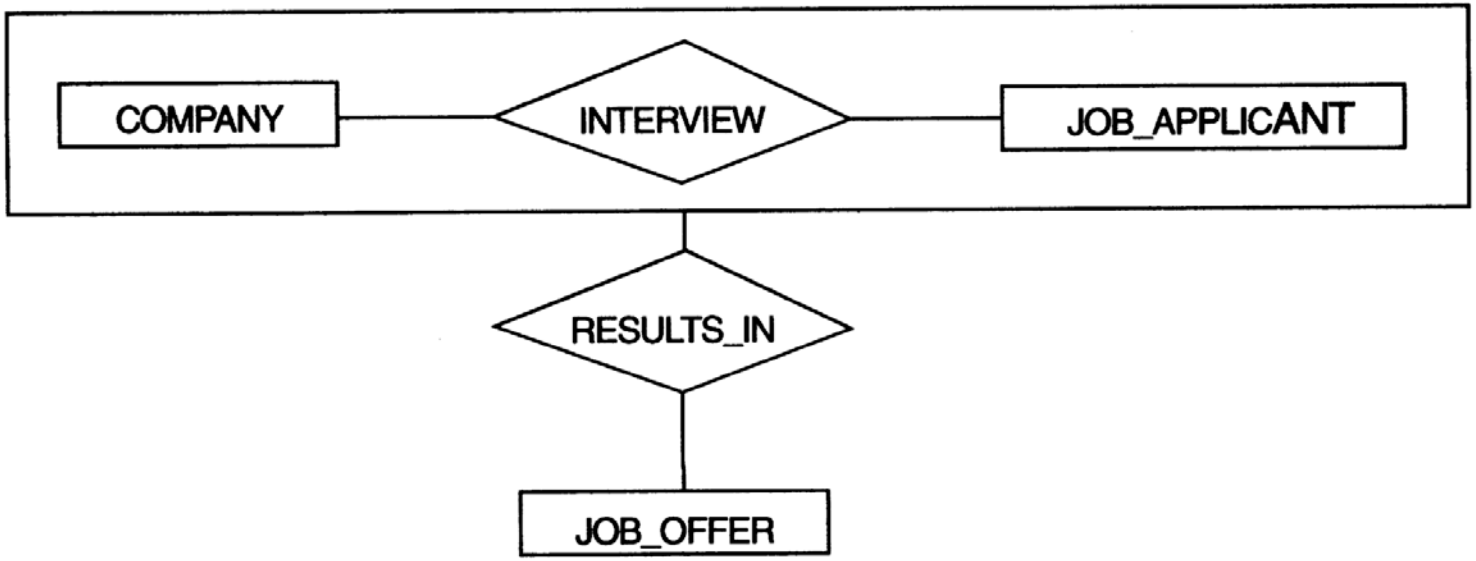


Modeling this as ternary is not correct, because not all Interviews result into job offers?

Aggregation: solves this problem

- Aggregation is used when relationship is to act as one of the participating entity.

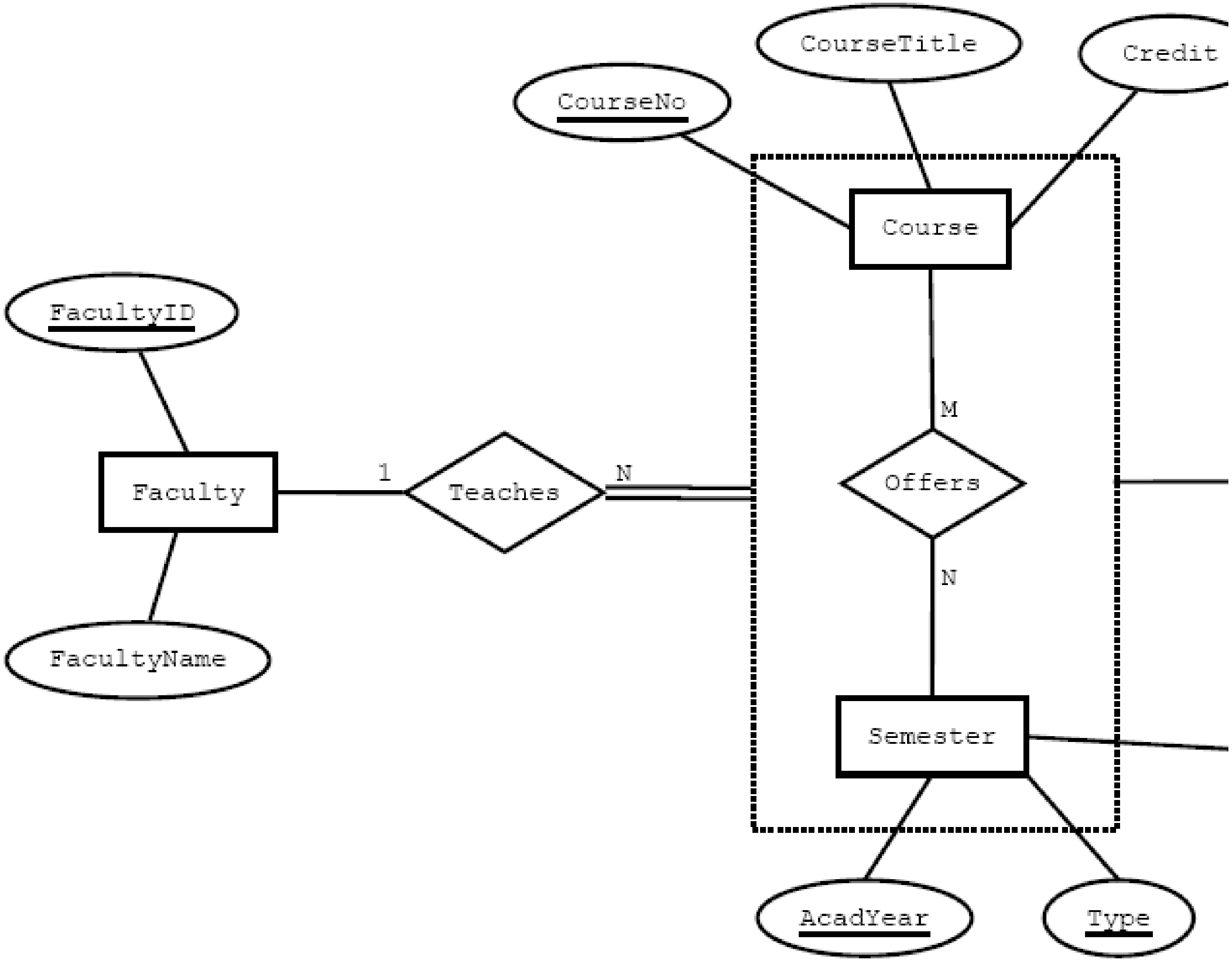
(d)



Consider following relationships

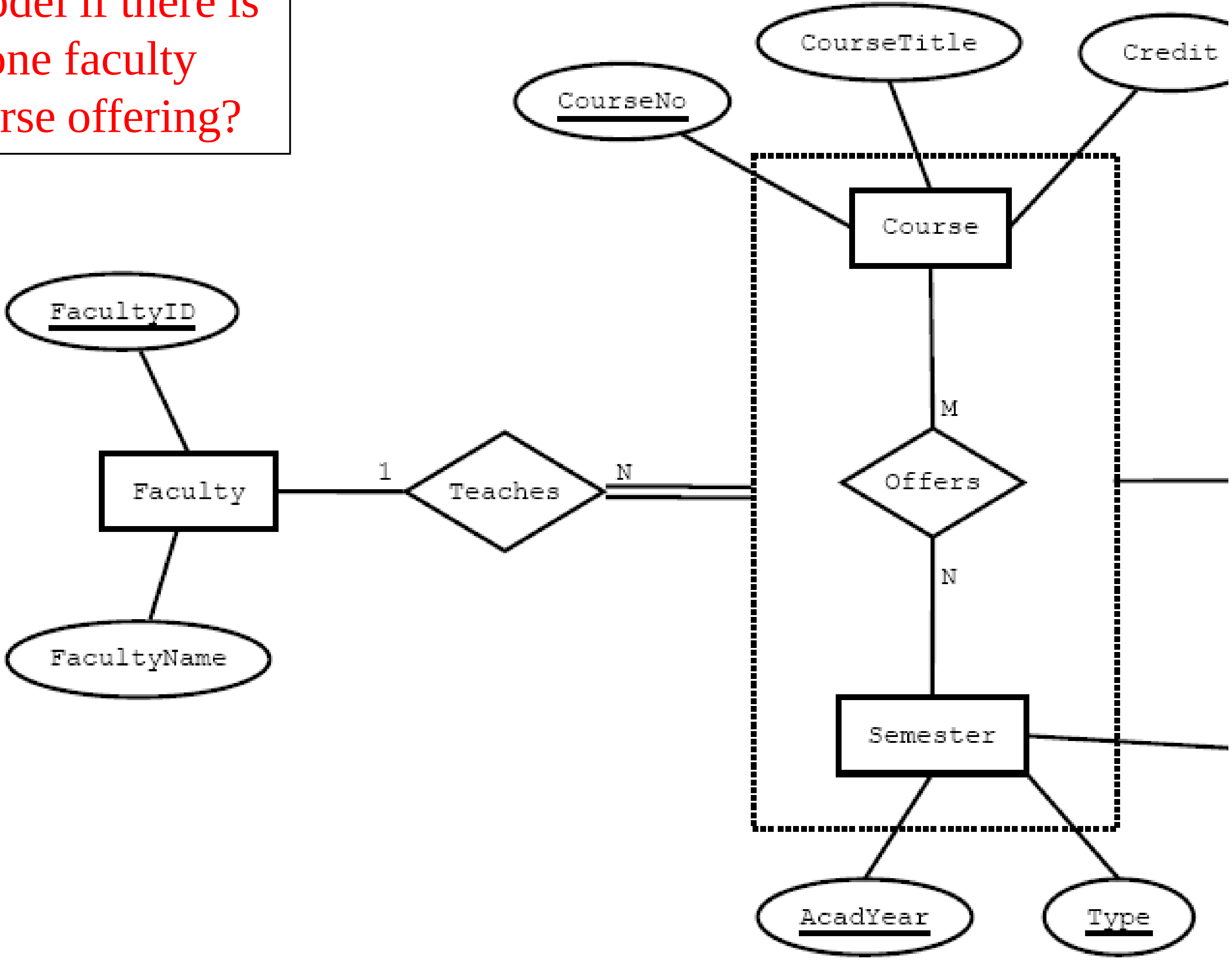
- In DA-Acad Scenario
  - Instructor takes a course in a semester
  - Student registers in a course in a semester
- Why not model above relationships as ternary ?

Instructor takes a course in a semester

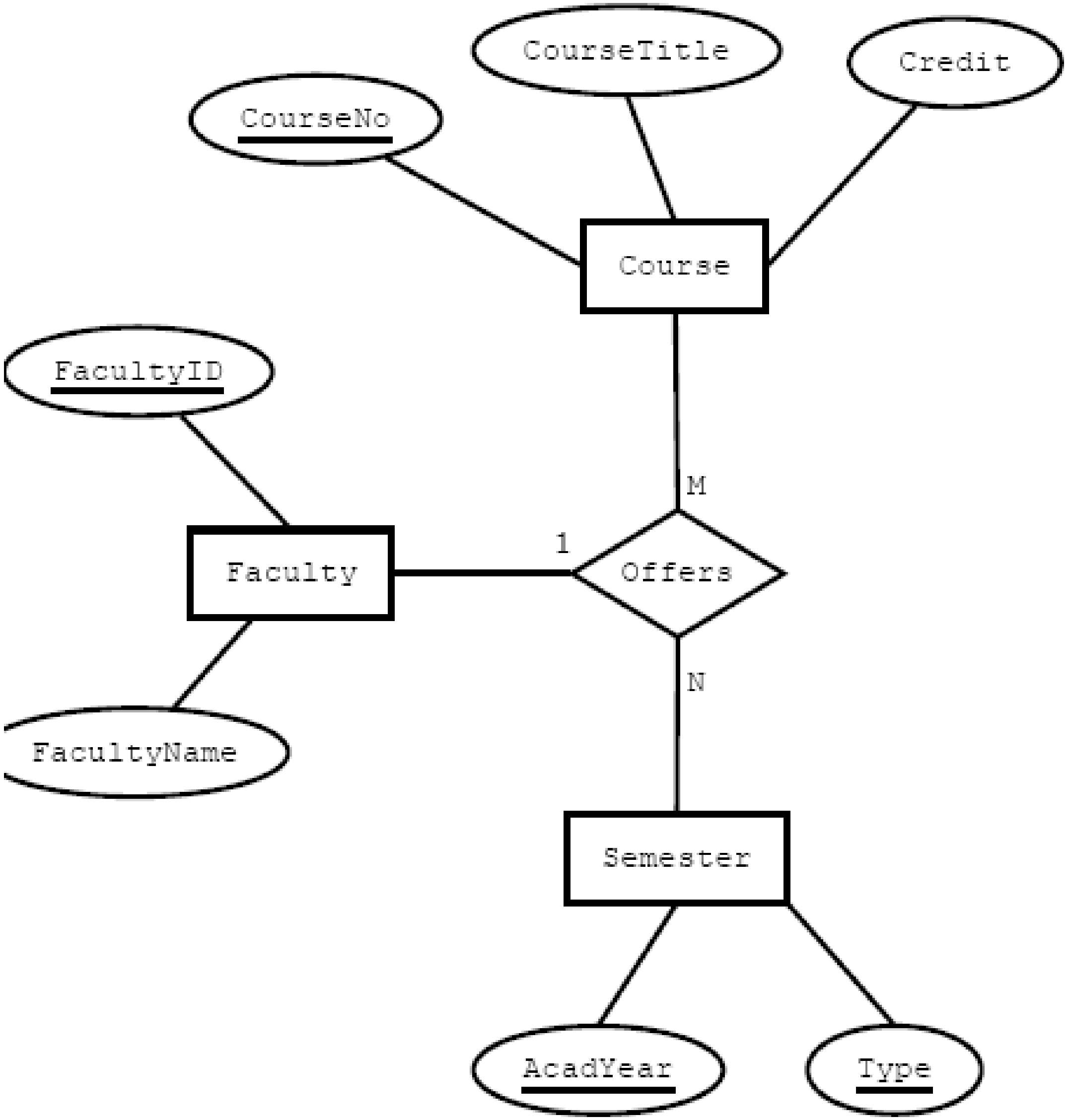


Instructor takes a course in a semester

How do we model if there is more than one faculty teaches a course offering?



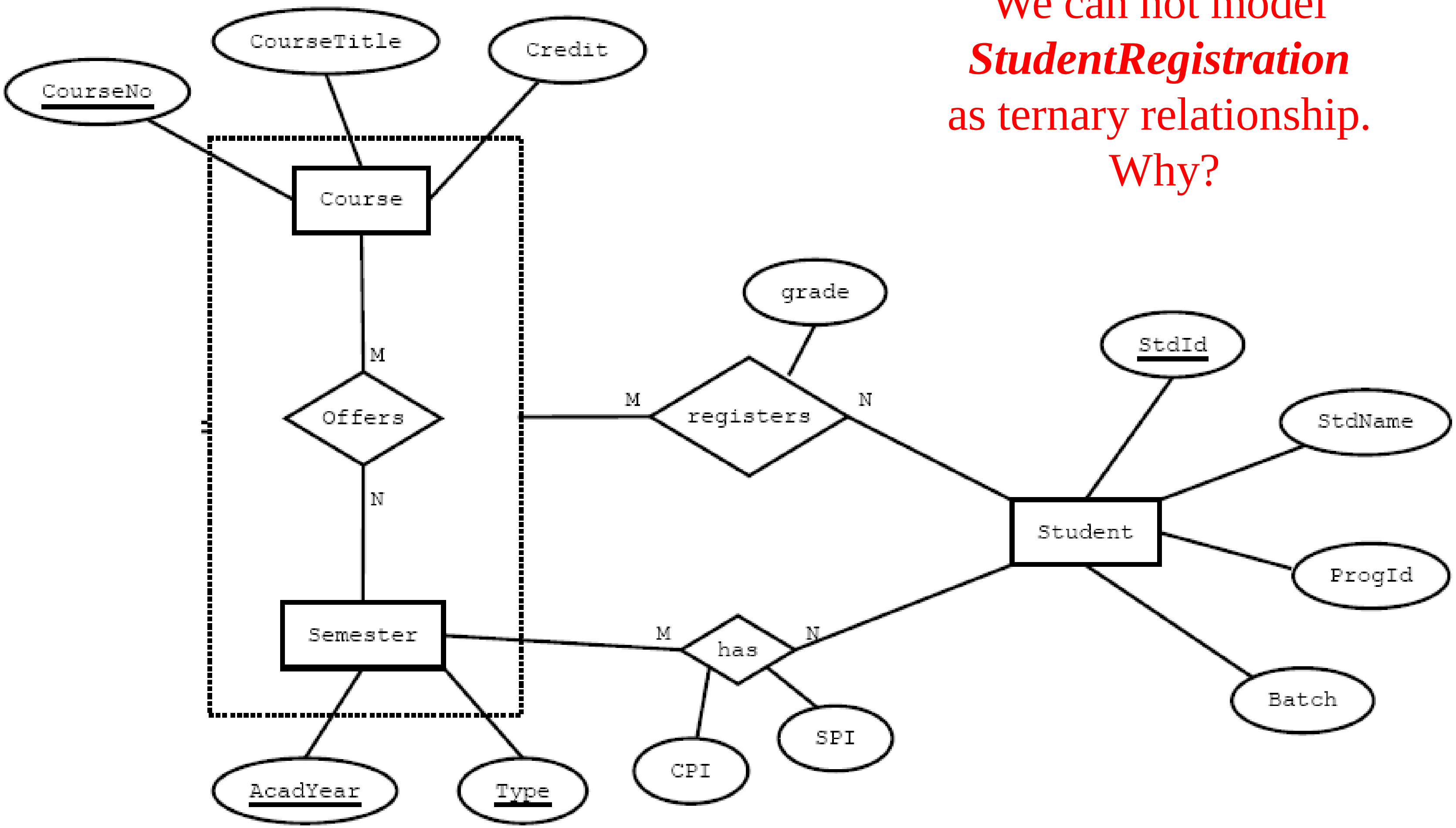
Instructor takes a course in a semester as ternary



Ternary and Aggregation

- Cases where you have an Aggregation in which participation of “the binary relationship” in a relationship with third entity is mandatory, ternary may be equally representative.
- For example in case of Course-Semester-Instructor case, since participation “Offers” relationship with instructor is mandatory, then this situation can also be represented through ternary relationship.
- However there could still be aggregation be a preferred way-
  – What if there is possibility of associating more than one instructor?

Student registering in a course



We can not model ***StudentRegistration*** as ternary relationship. Why?

Ternary and Aggregation

- Which is better ternary or Aggregation? and why?

- While aggregation is simple (with the simple reason that it is still a set of binary relationships) ; moreover it intuitive and natural too; therefore it should be preferred approach.

- In most cases, however, you should be able to live without ternary.

Ternary and n-ary Relationships

- If three are three entities participating in a relationship, then it is ternary
- A popular scenario: Project-Part-Supplier: A number of suppliers supply parts to projects. Any part may be supplied by any supplier to any project in any arbitrary manner.
- If participating entities are n, then it is n-ary relationship.

- Exercise Case Studies

Database for an online book-store

- Consider maintaining database of books (let us say by an on-line book-store) with details of ISBN, Title, Price (consists of Currency, and Amount), Author-Name, Publisher-Name, Publisher-Address.
- A book has one (at least) or more authors, and published by exactly one publisher. ISBN is a universally unique number each book has.
- Some of books are reprinted by different publisher in some other country or region. In that case reprint of the book is given different ISBN, and will have different price in some other currency.
- Let us also maintain relationship of reprint with original publication.

Library scenario

- A Library scenario. Suppose you need to create computerized database system for the library. It keeps tracks of books, library members, books issues and returns.
- Every copy of book is given a unique accession number in the library, suppose there are five copies of database systems by elmasri/navathe, each copy would have different accession numbers.
- Book issue and returns – keep record of date of issue, due date, date of actual return etc. We may also like to keep name, email, and phone number of all the library members

IBM TGMC

- A number of Teams from various institutes Participate in the competition.
- Each Team has a mentor, a faculty from same institute
  - Every team has to register online at their site www.tgmc.in, while registering they are required to furnish following details- Team Name, Login ID, Password, team members (email id and contact number), mentor and institute names
- IBM awards to Winner and Runner.
- Objective of Database: IBM want to keep record of all teams registered for the event and for there after math

Database for Hall of Residence at DAIICT

- Halls of Residence at DA-IICT - you have rooms and residents !

Academic Record Scenario at DAIICT

- A number of course are offered in a semester of an Academic Year (say 2010-11, and let us call it 2010 by a single value)
- Semesters are of three types- Autumn, Winter, Summer
- One Instructor is allocated to each offering
- Students registers in a set of offered courses
- Each course has CourseNo, Course Title, and Credit
- At the end of semester grades are awarded to every registered student in all offered courses.
- Based on grade student earns, and credit of the courses, SPI of student is computed for that semester.
- It also updates CPI of student
- For student, let us say we record, program in which he studies, year of registration, and CPI