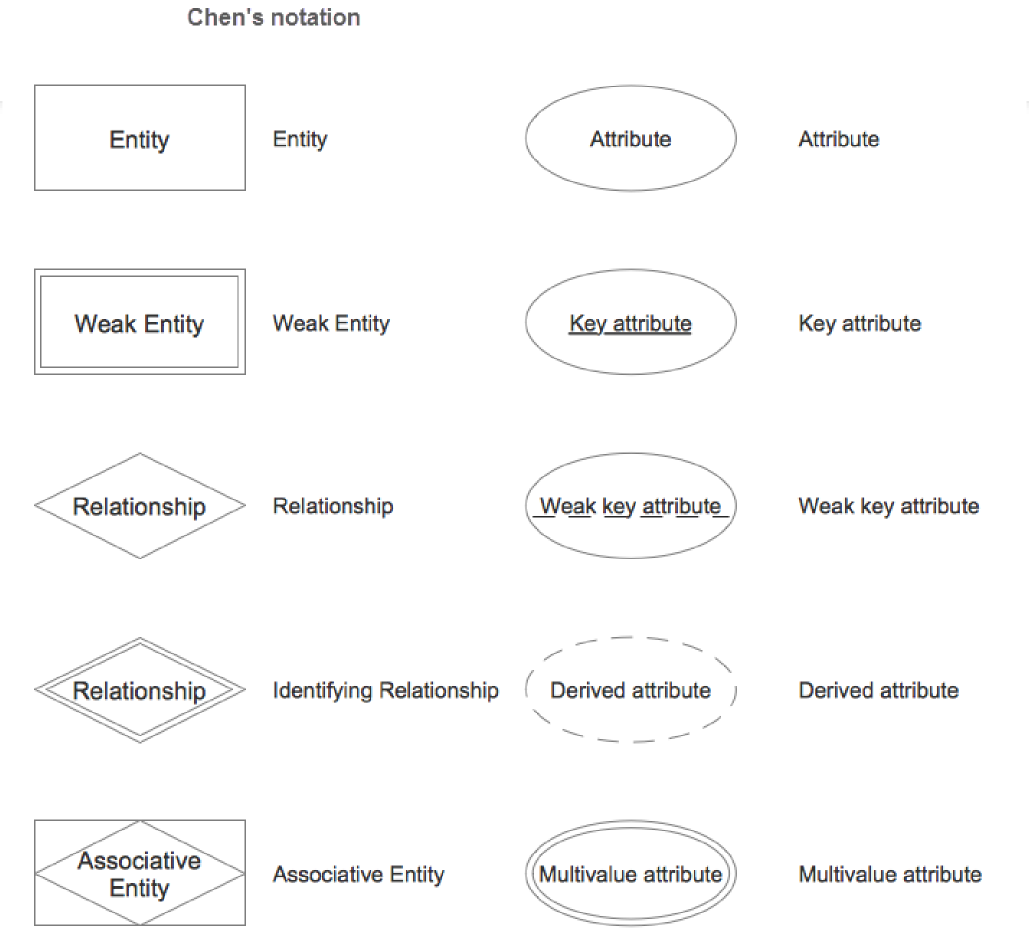
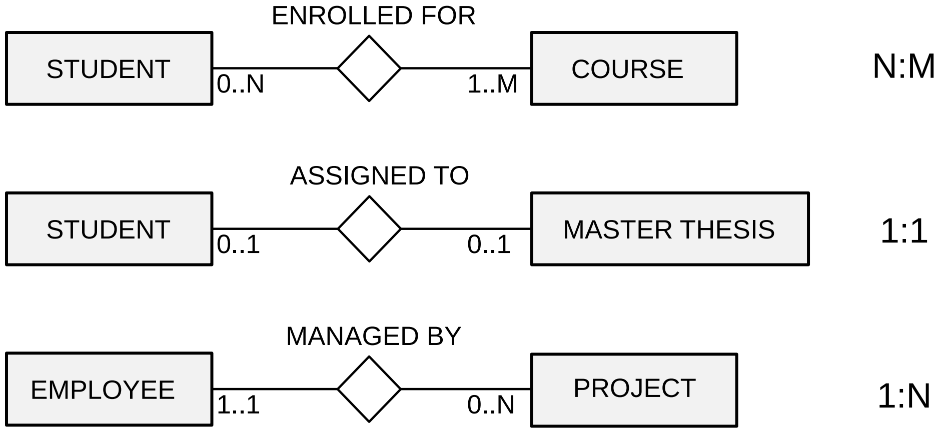
Entity Relationship (ER) Model

CS 341 Database Systems

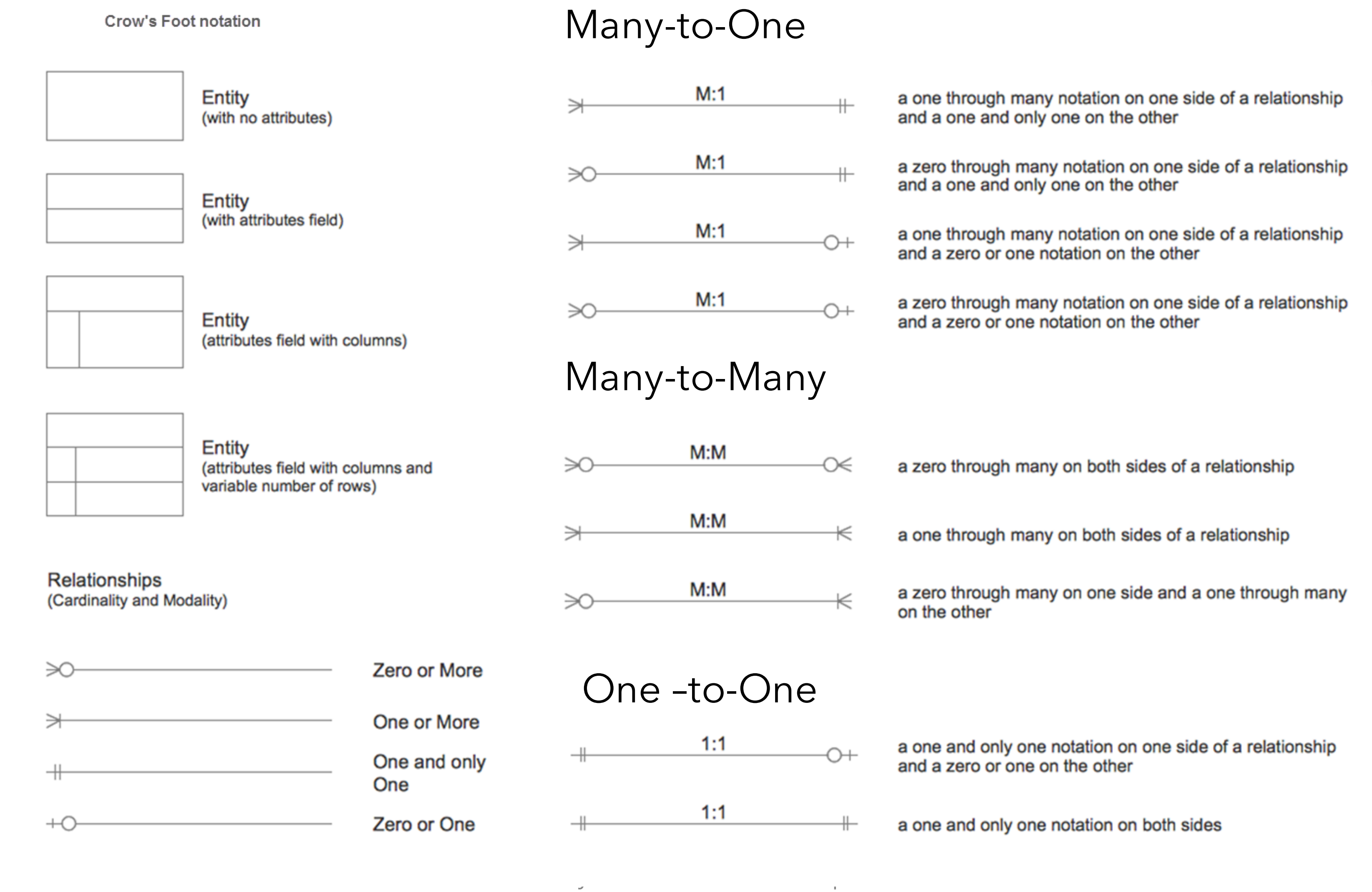
Chen’s Notation





In some books, the diagrams in chen’s notation which have a single number for multiplicity instead of 2 for example (1..N), the participation is explained by single line (partial participation) and double lines (total participation) for relations between 2 entities.

Crow’s Foot Notation



Crow's Foot notation

Entity
(with no attributes)

Entity
(with attributes field)

Entity
(attributes field with columns)

Entity
(attributes field with columns and variable number of rows)

Relationships
(Cardinality and Modality)

Zero or More

One or More

One and only One

Zero or One

Many-to-One

M:1 a one through many notation on one side of a relationship and a one and only one on the other

M:1 a zero through many notation on one side of a relationship and a one and only one on the other

M:1 a one through many notation on one side of a relationship and a zero or one notation on the other

M:1 a zero through many notation on one side of a relationship and a zero or one notation on the other

Many-to-Many

M:M a zero through many on both sides of a relationship

M:M a one through many on both sides of a relationship

M:M a zero through many on one side and a one through many on the other

One -to-One

1:1 a one and only one notation on one side of a relationship and a zero or one on the other

1:1 a one and only one notation on both sides

Keys

Candidate Key

- Minimal set of attributes that uniquely identifies each occurrence of an entity type.

Primary Key

- Candidate key selected to uniquely identify each occurrence of an entity type.

Composite Key

- A candidate key that consists of two or more attributes.

Summary of multiplicity constraints

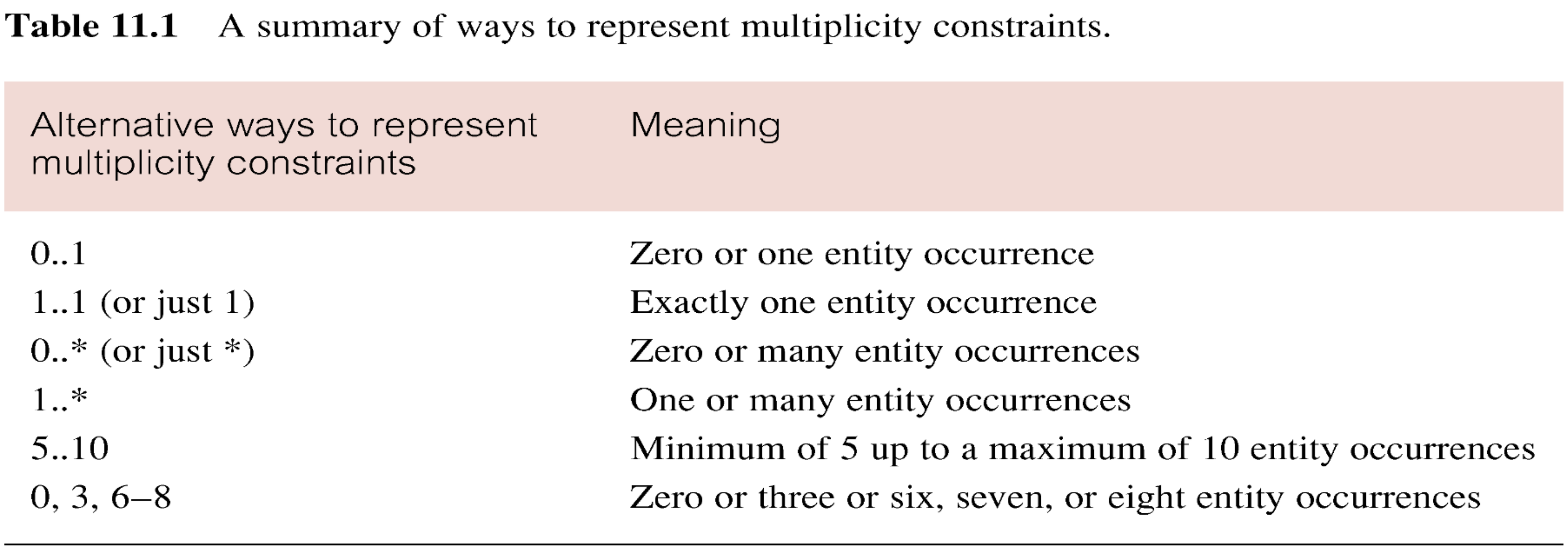
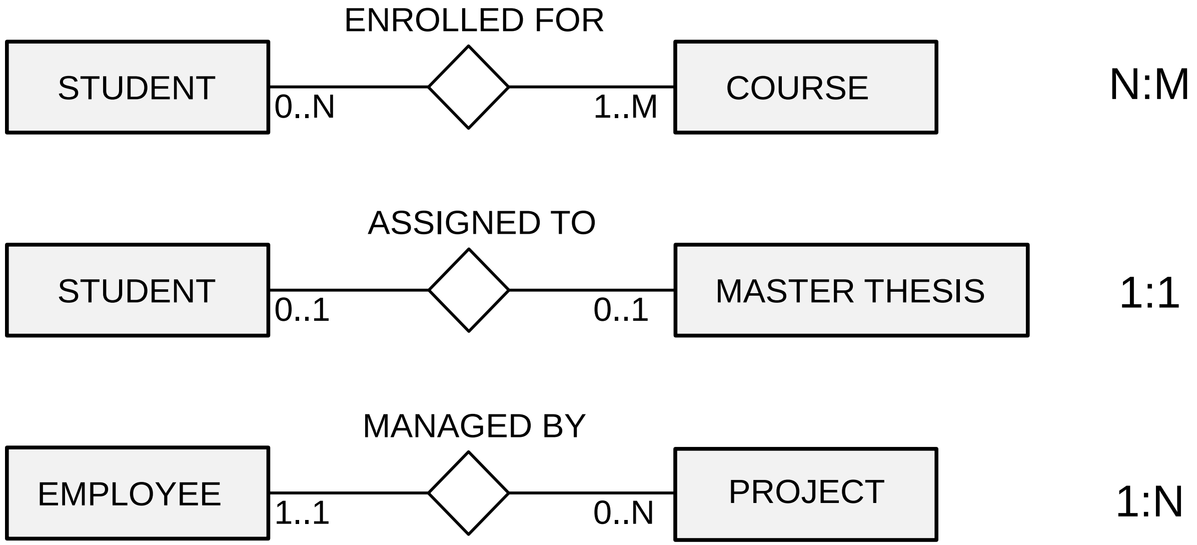


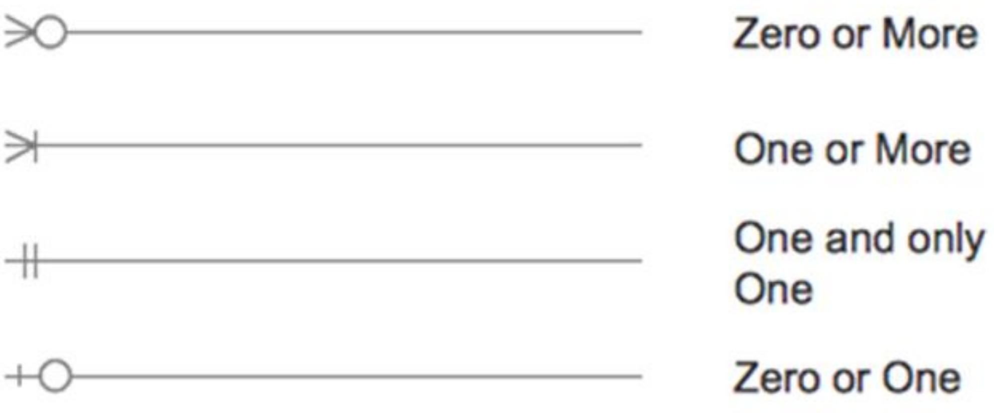
Table 11.1 A summary of ways to represent multiplicity constraints.

Alternative ways to represent multiplicity constraints	Meaning
0..1	Zero or one entity occurrence
1..1 (or just 1)	Exactly one entity occurrence
0..* (or just *)	Zero or many entity occurrences
1..*	One or many entity occurrences
5..10	Minimum of 5 up to a maximum of 10 entity occurrences
0, 3, 6–8	Zero or three or six, seven, or eight entity occurrences

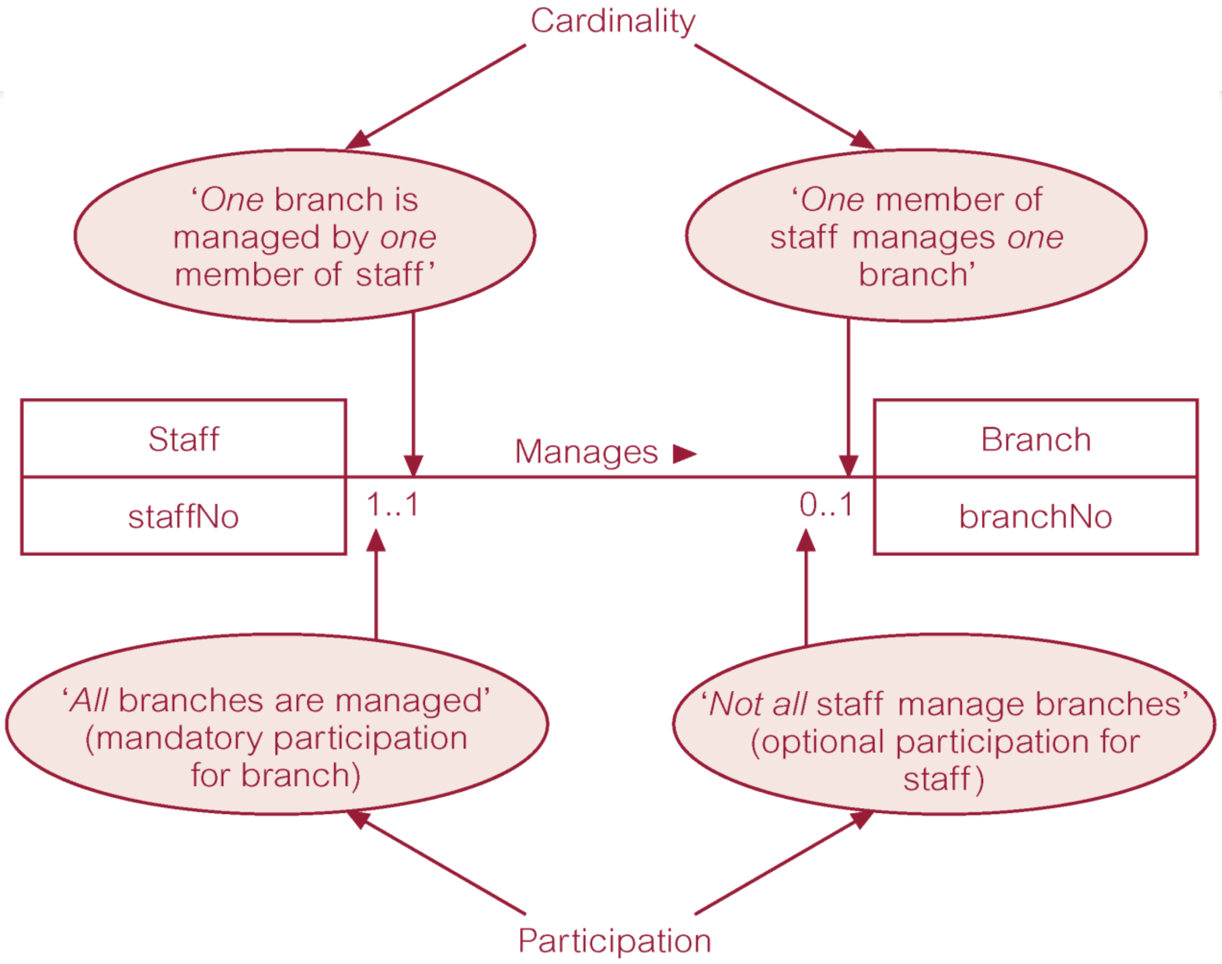
Multiplicity

Chen’s VS Crows Foot





Multiplicity as *cardinality* and *participation* constraints


Cardinality
'*One* branch is managed by *one* member of staff'
'*One* member of staff manages *one* branch'
Staff
staffNo
Manages ►
1..1
0..1
Branch
branchNo
'*All* branches are managed' (mandatory participation for branch)
'*Not all* staff manage branches' (optional participation for staff)
Participation

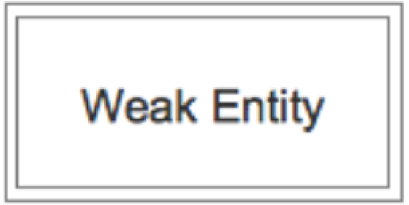
Entity Type

Strong Entity Type

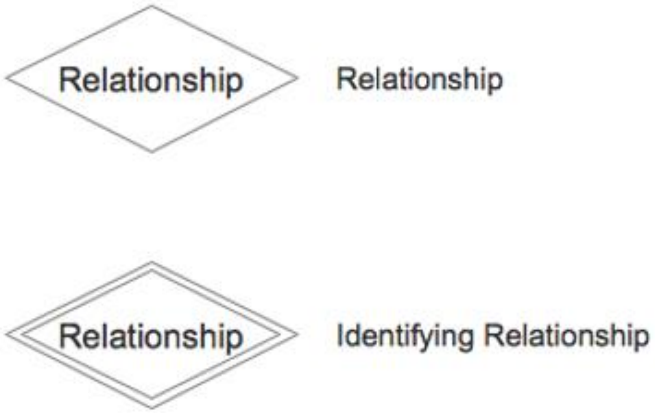
- Entity type that is not existence-dependent on some other entity type.

Weak Entity Type

- Entity type that is existence-dependent on some other entity type.



Relationship Type



Strong Relationship

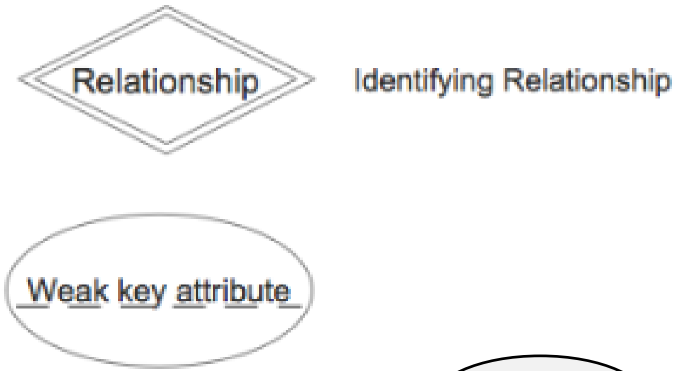
- A strong relationship, also known as an **identifying** relationship, exists when the PK of the related entity contains a PK component of the parent entity.

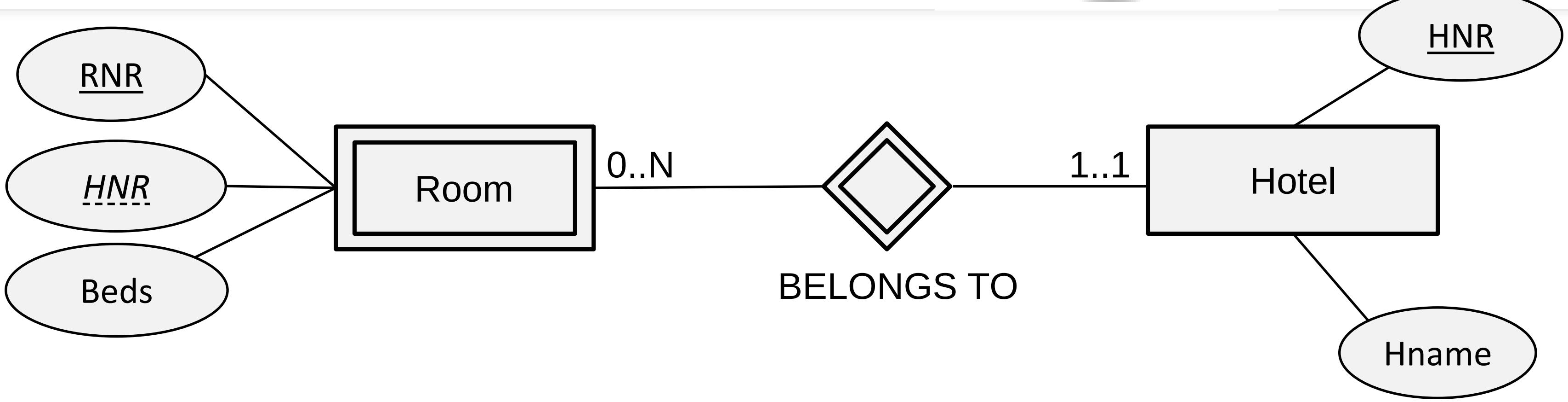
Weak Relationship

- A weak relationship ,also known as a **non-identifying** relationship, exists if the PK of the related entity does not contain a PK component of the parent entity.

- Crow's foot notation: Solid line for strong and dotted for weak relationships.

Weak Entity
Strong Relationship

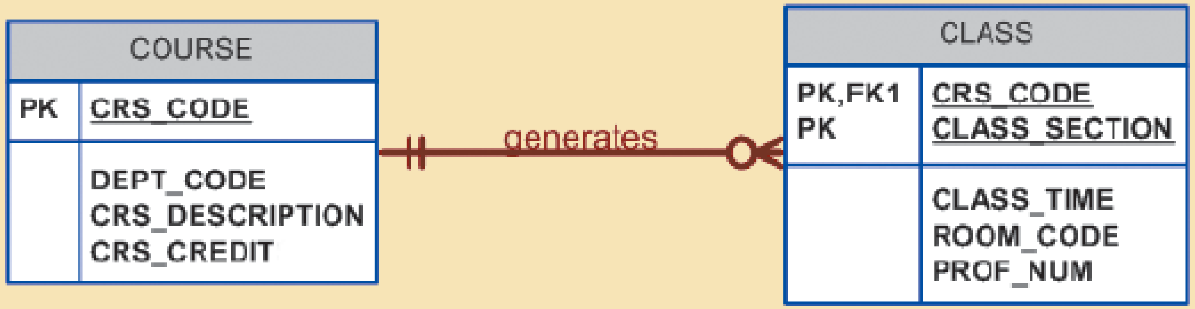




- A strong entity type is an entity type that has a key attribute type
- A weak entity type is an entity type that does not have a key attribute type of its own
  - related to owner entity type from which it borrows an attribute type to make up a key attribute type

Strong Vs Weak Relationship

Fig. 4.9 A strong (identifying) relationship between COURSE and CLASS



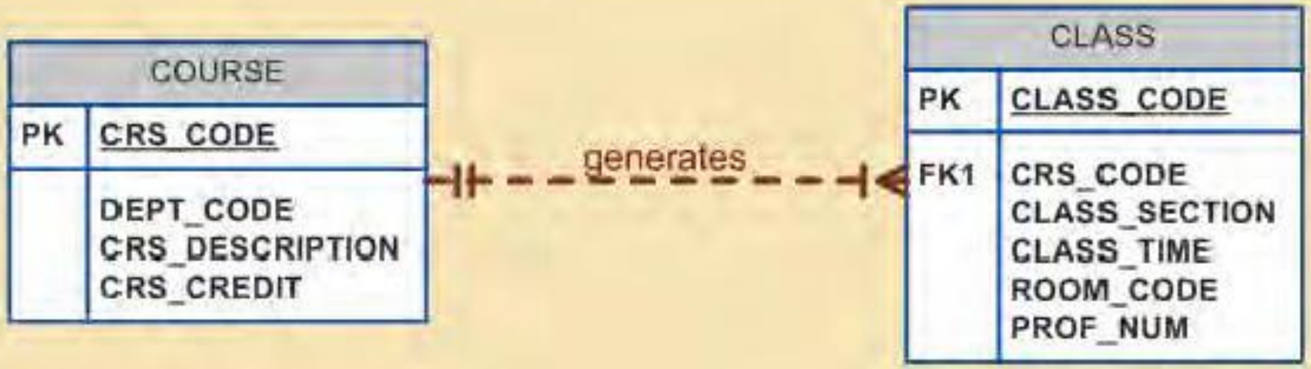
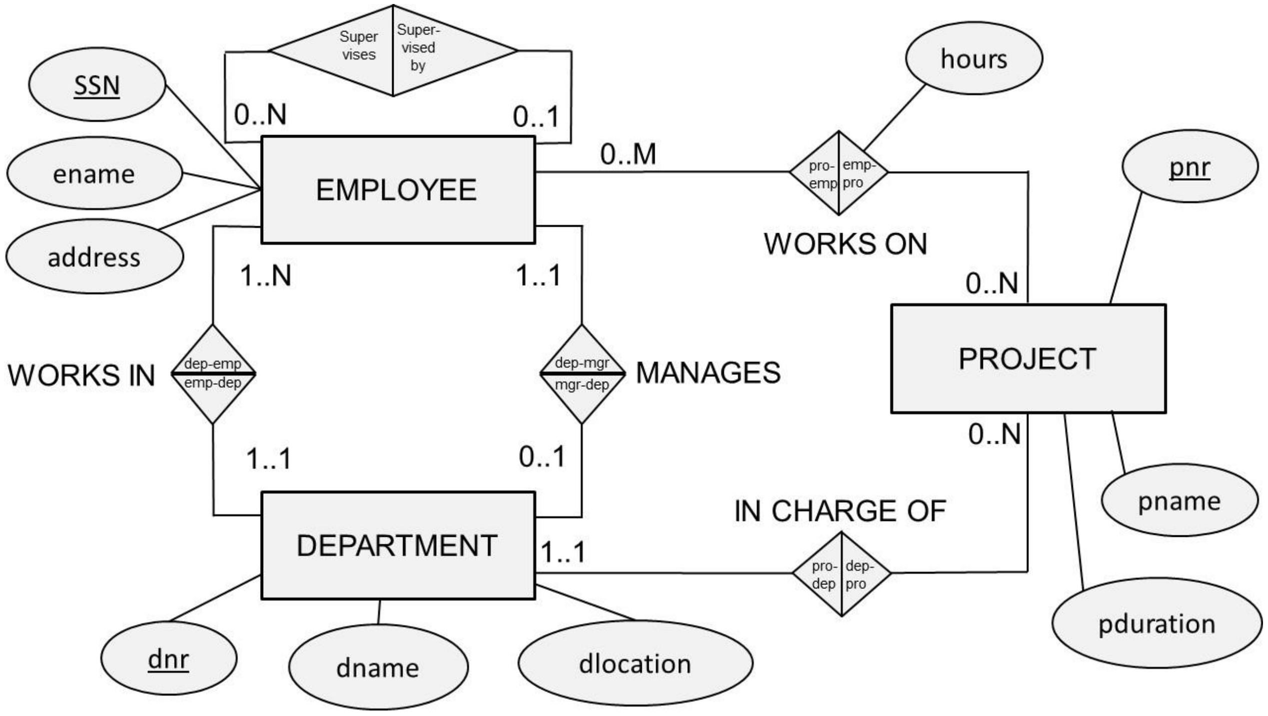
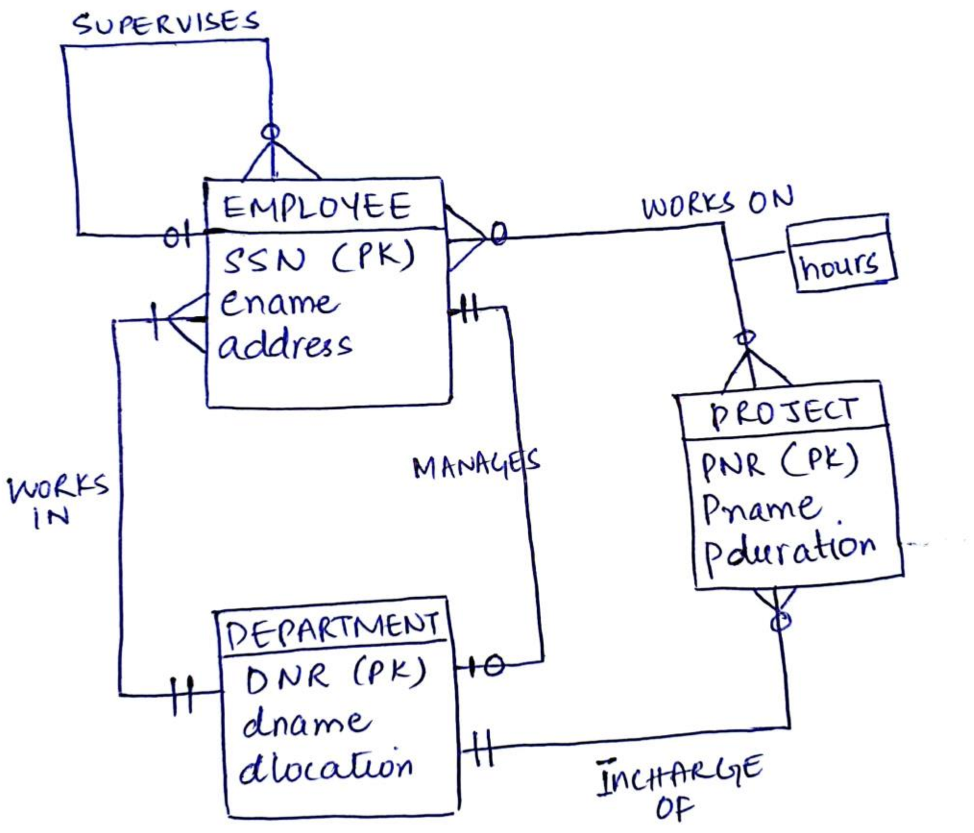


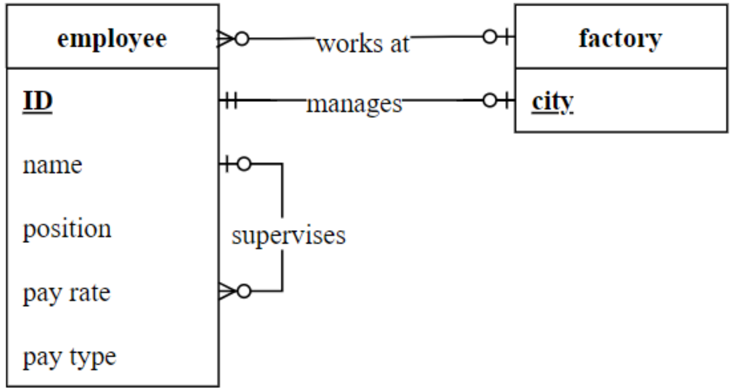
Fig. 4.8 A Weak Relationship between COURSE and CLASS

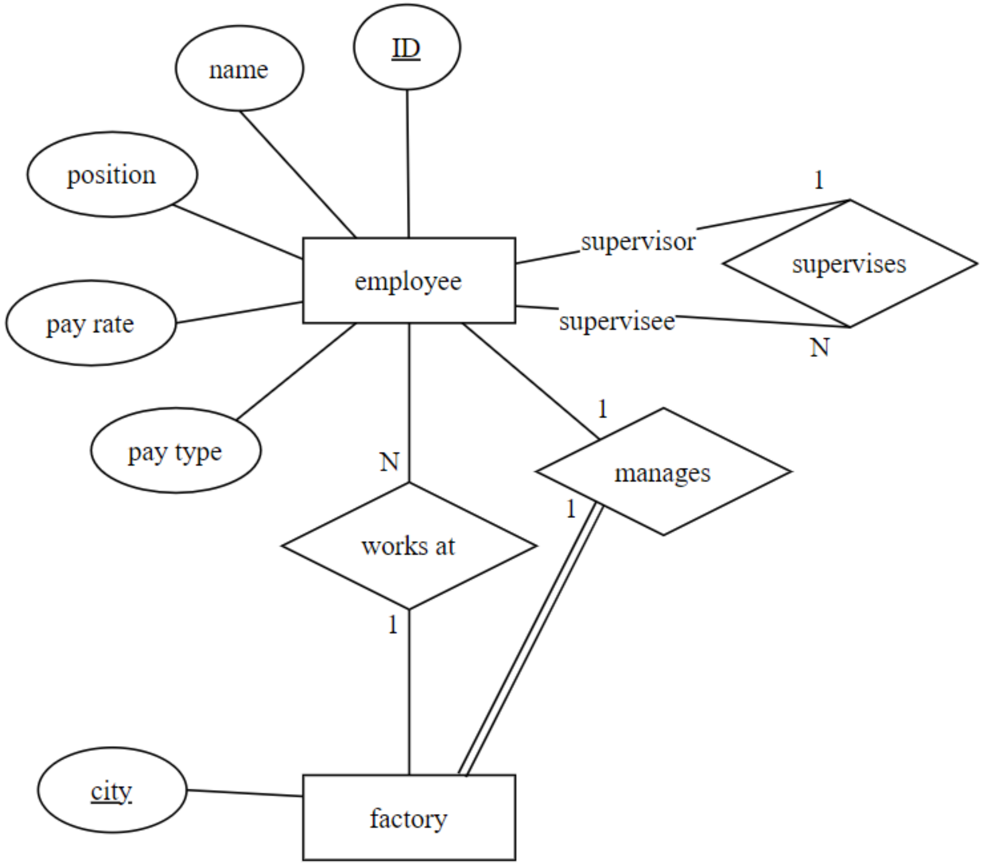
Comparison


SSN
ename
address
Super vises
Super-vised by
0..N
0..1
EMPLOYEE
0..M
hours
pro-emp
emp-pro
WORKS ON
pnr
0..N
PROJECT
1..N
1..1
WORKS IN
dep-emp
emp-dep
dep-mgr
mgr-dep
MANAGES
1..1
0..1
0..N
IN CHARGE OF
pname
DEPARTMENT
1..1
pro-dep
dep-pro
pduration
dnr
dname
dlocation


SUPERVISES
EMPLOYEE
SSN (PK)
ename
address
WORKS ON
hours
PROJECT
PNR (PK)
Pname
Pduration
WORKS IN
MANAGES
DEPARTMENT
DNR (PK)
dname
dlocation
INCHARGE OF

Comparison


employee
ID
name
position
pay rate
pay type
works at
manages
supervises
factory
city


name
ID
position
pay rate
pay type
employee
supervisor
supervisee
supervises
1
N
N
works at
1
1
manages
1
city
factory

Limitations of the ER Model

Problems with ER Models

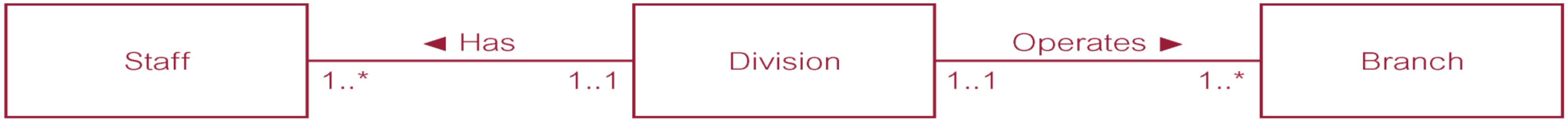
- Problems may arise when designing a conceptual data model called **connection traps**.
- Often due to a misinterpretation of the meaning of certain relationships.
- Two main types of connection traps are called ***fan traps*** and ***chasm traps*****.**

Connection Traps

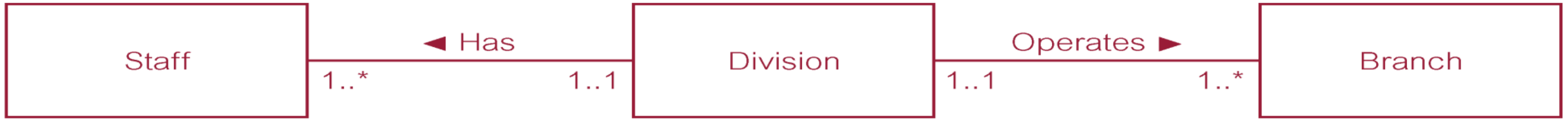
- **Fan Trap**
  - Where a model represents a relationship between entity types, but pathway between certain entity occurrences is ambiguous.
- **Chasm Trap**
  - Where a model suggests the existence of a relationship between entity types, but pathway does not exist between certain entity occurrences.

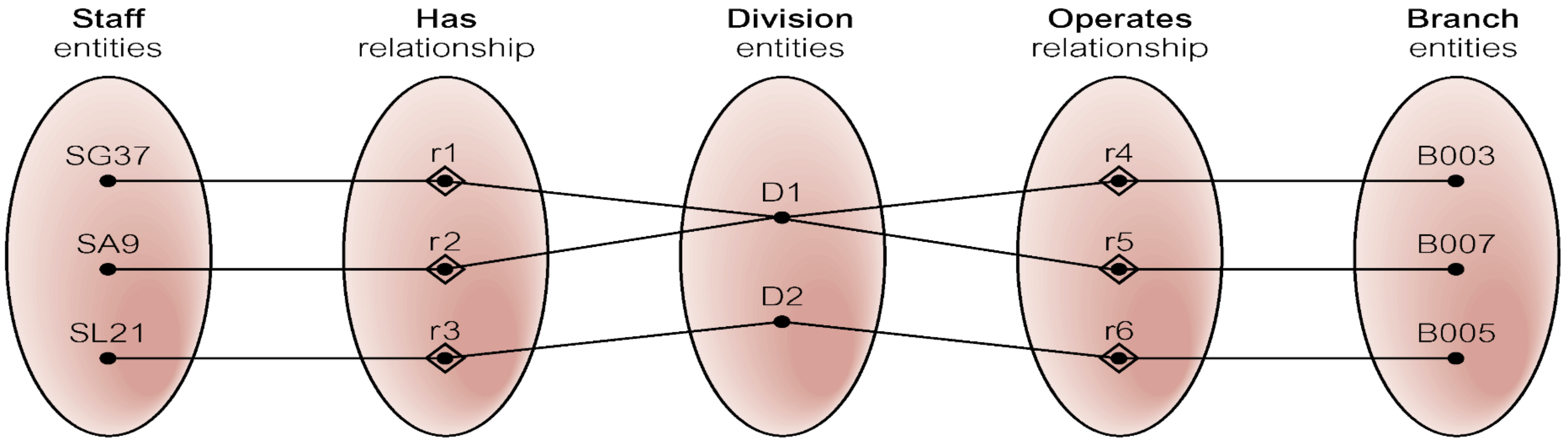
Fan Trap

- A fan trap may exist where more than one (1:M) relationships fan out from a single entity.



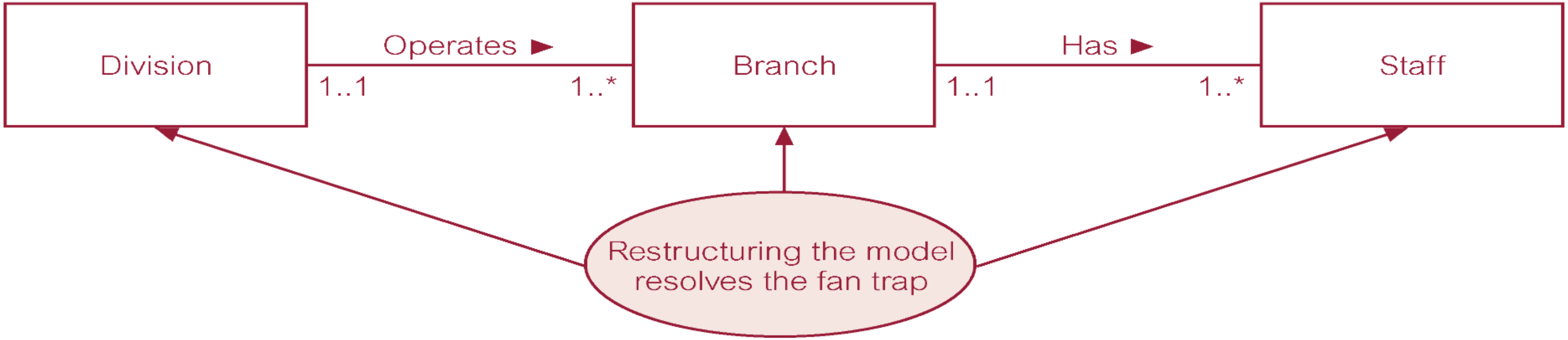
An Example of a Fan Trap



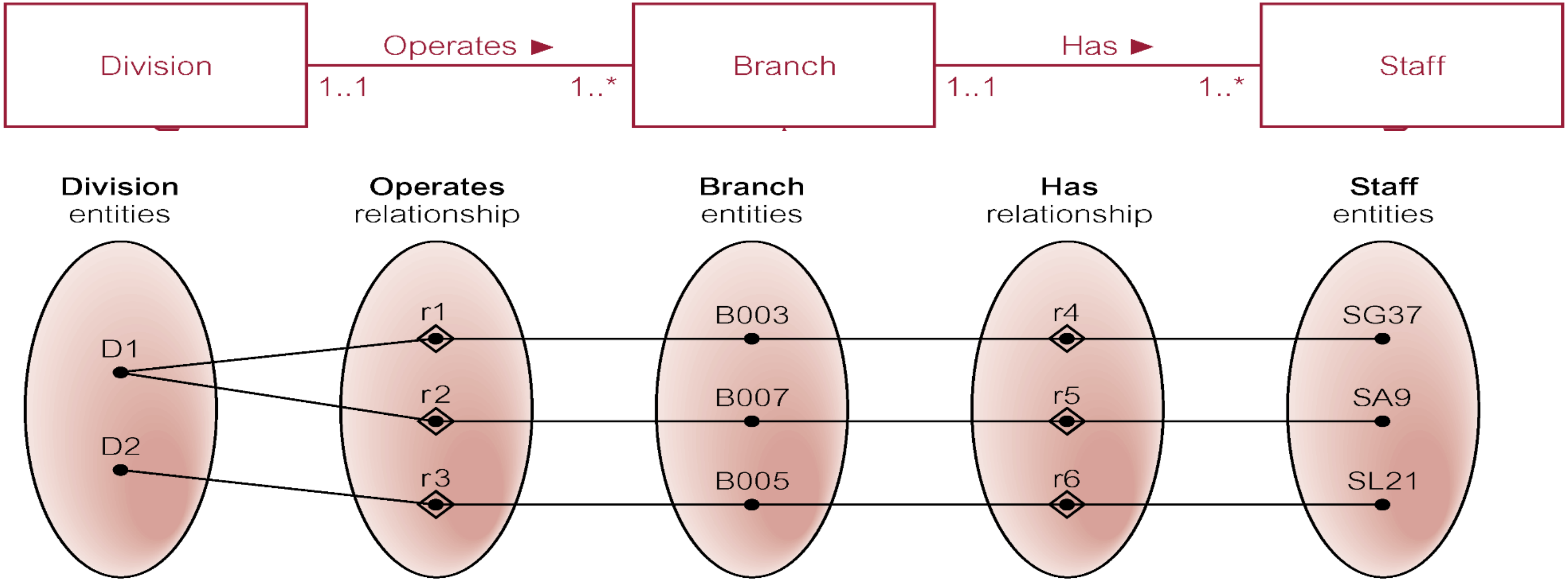


At which branch office does staff number SG37 work?

Restructuring ER model to remove Fan Trap



Semantic Net of Restructured ER Model with Fan Trap Removed

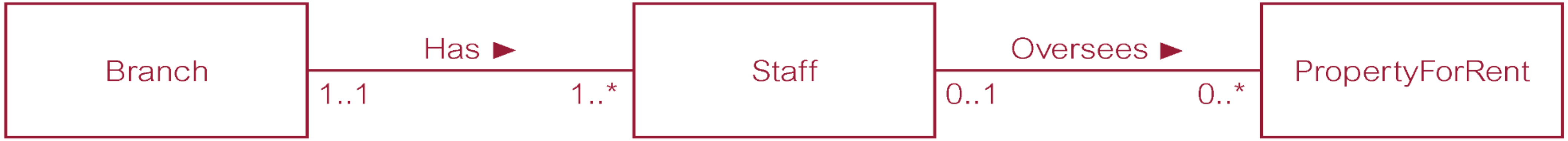


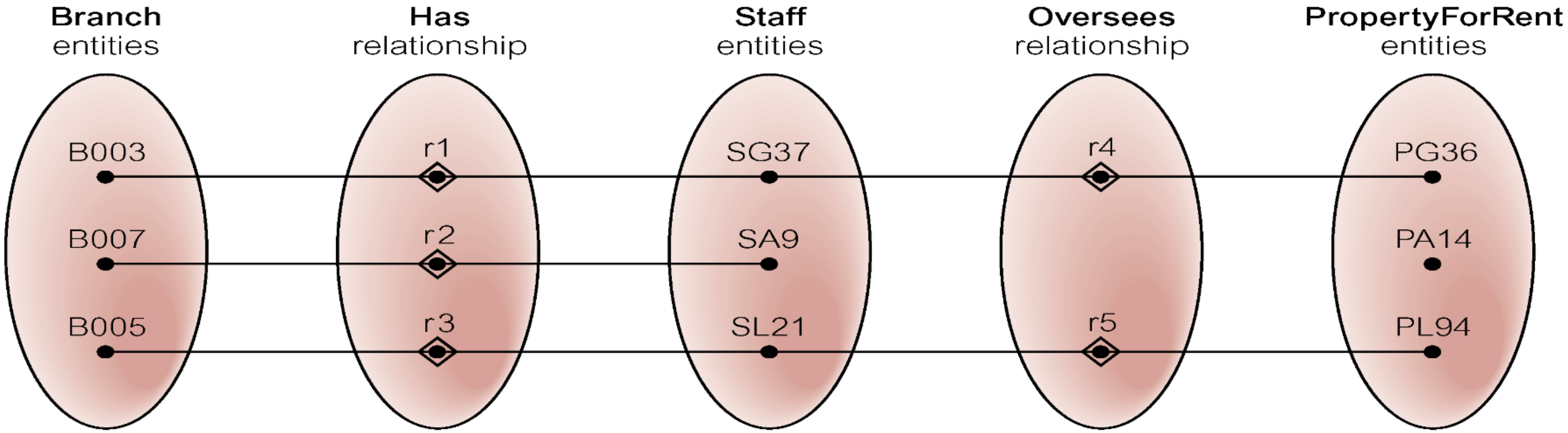
SG37 works at branch B003.

Chasm Trap

- Where a model suggests the existence of a relationship between entity types, but pathway does not exist between certain entity occurrences.
- A chasm trap may occur when there are one or more relationships with partial participation

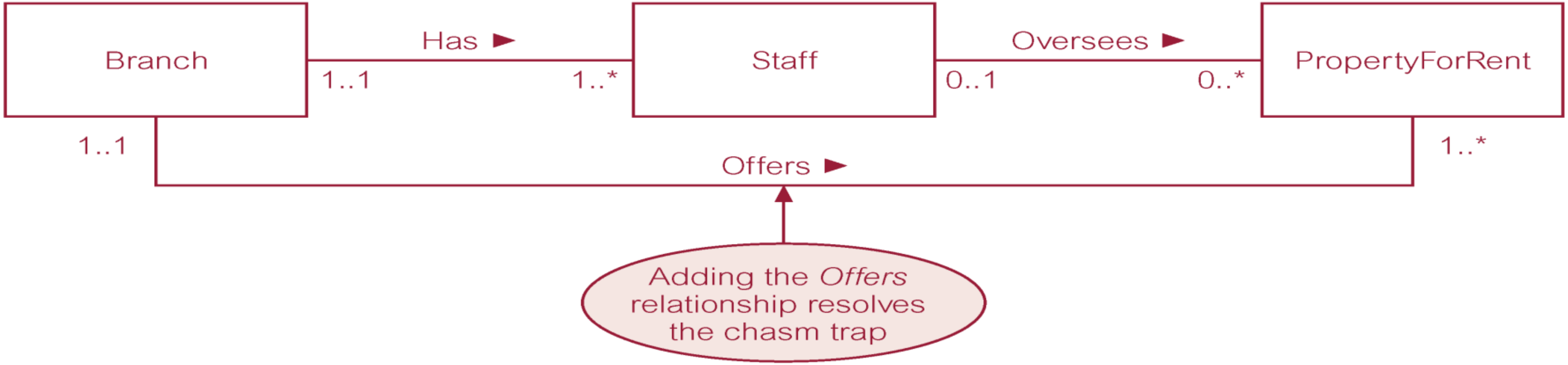
An Example of a Chasm Trap



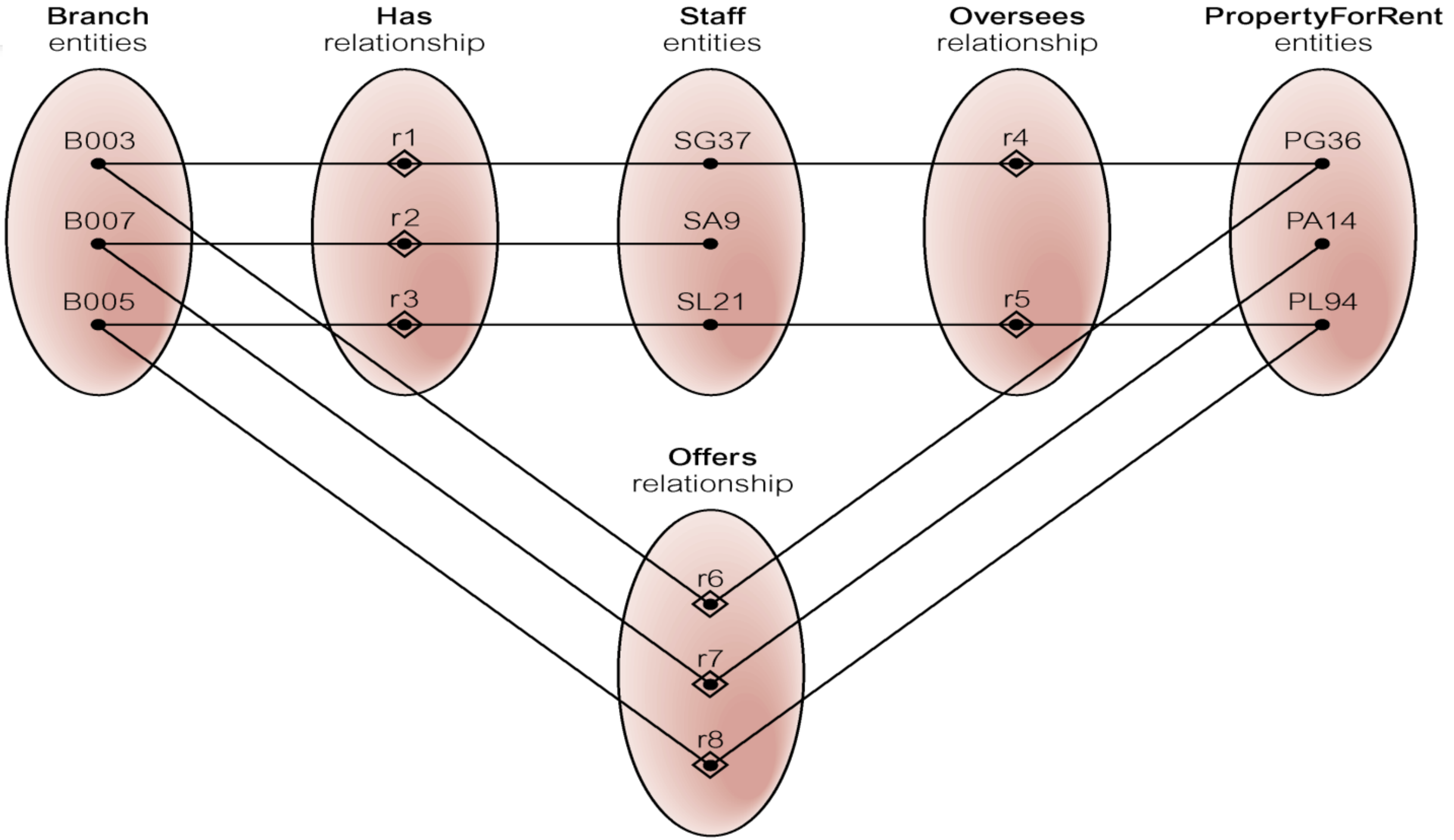


At which branch office is property PA14 available?

Restructured to remove Chasm Trap



Semantic Net of Restructured ER Model with Chasm Trap Removed



Limitations of the ER model

- **ER model presents a temporary snapshot and cannot model temporal constraints**
  - Examples:
    - a project needs to be assigned to a department after one month
    - an employee cannot return to a department of which he previously was a manager
    - a purchase order must be assigned to a supplier after two weeks
    - etc

Limitations of the ER model

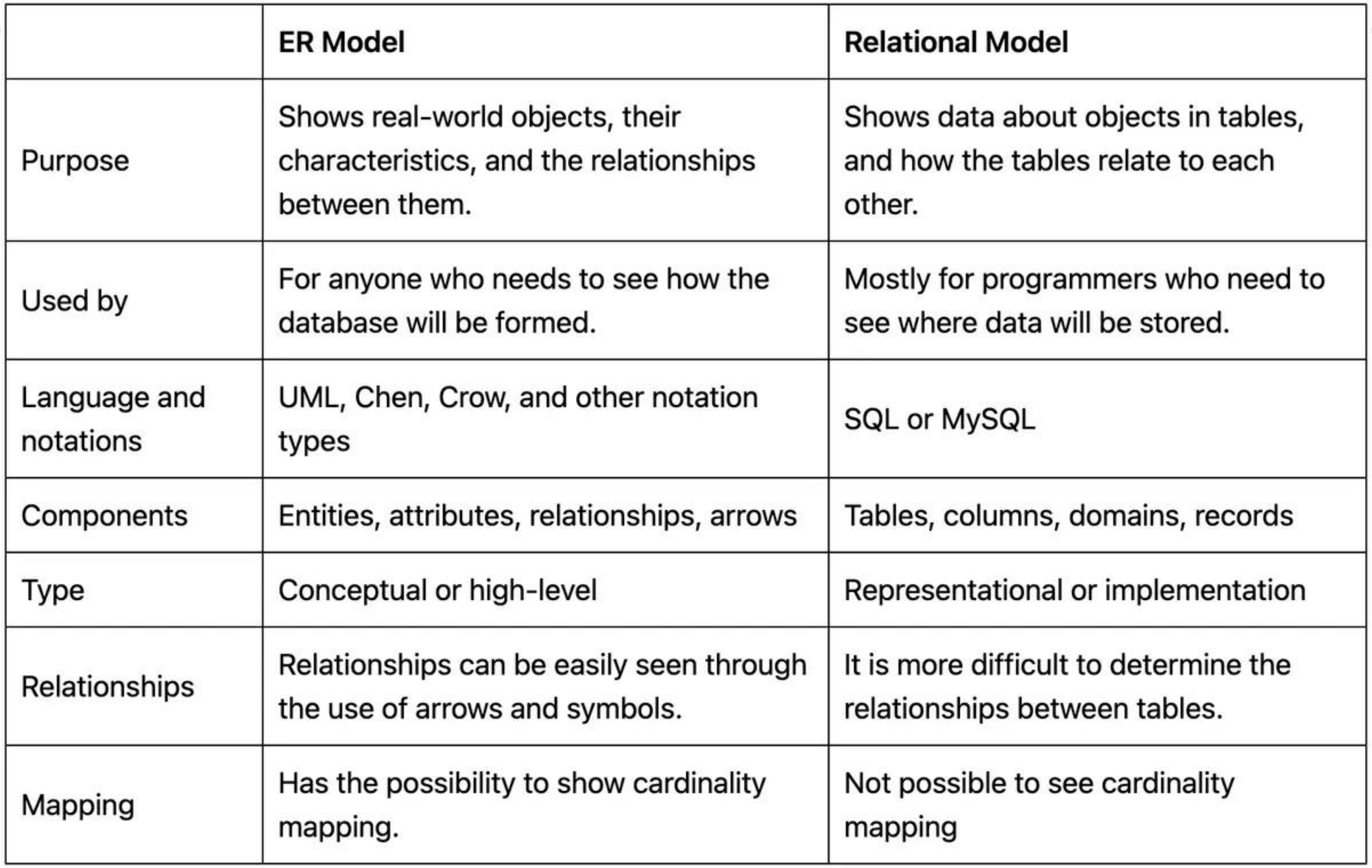
- **ER model cannot guarantee the consistency across multiple relationship types**
  - Examples:
    - An employee should work in the department that he/she manages
    - Employees should work on projects assigned to departments to which the employees belong
    - Suppliers can only be assigned to purchase orders for products they can supply

Limitations of the ER model

- **Attribute domains are not included in the ER model**
  - Examples: hours should be positive; prodtype must be red, white or sparkling, supstatus is an integer between 0 and 100
- **Functions are not included in the ER model**
  - Examples: calculate average number of projects an employee works on; determine which supplier charges the maximum price for a product

Implementing ER Model to Relational Model

ER Vs Relational



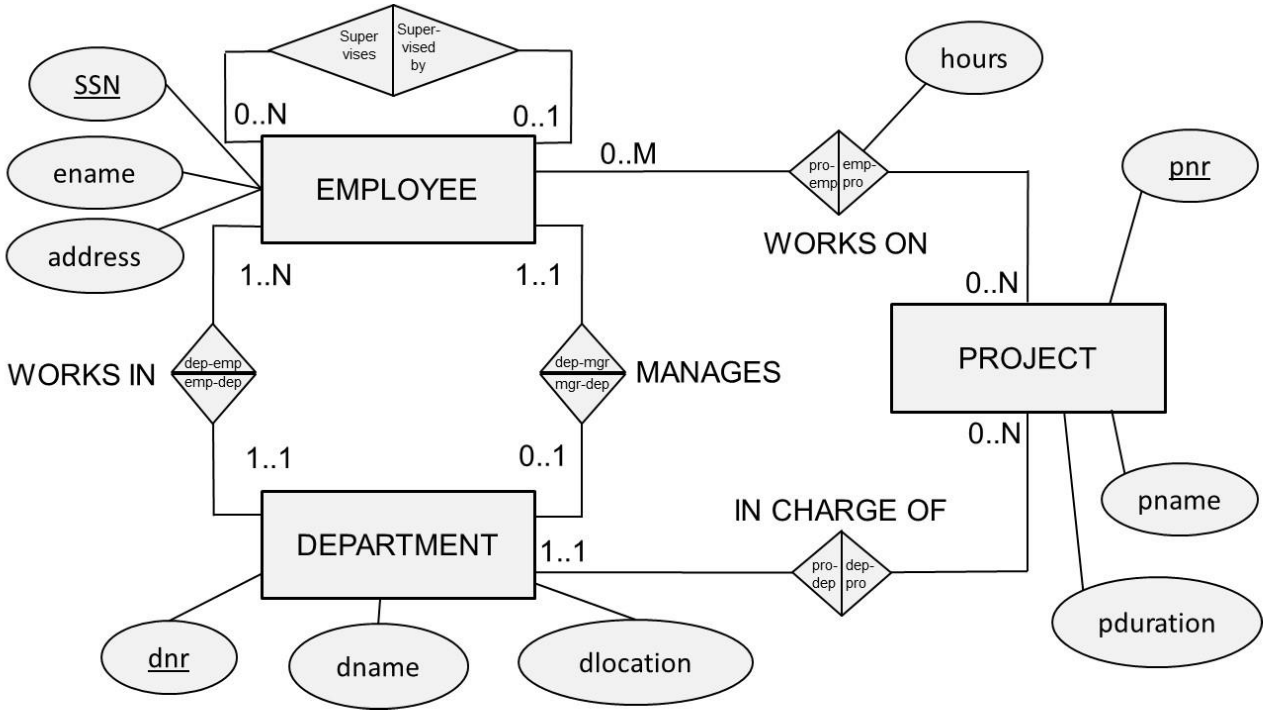
	ER Model	Relational Model
Purpose	Shows real-world objects, their characteristics, and the relationships between them.	Shows data about objects in tables, and how the tables relate to each other.
Used by	For anyone who needs to see how the database will be formed.	Mostly for programmers who need to see where data will be stored.
Language and notations	UML, Chen, Crow, and other notation types	SQL or MySQL
Components	Entities, attributes, relationships, arrows	Tables, columns, domains, records
Type	Conceptual or high-level	Representational or implementation
Relationships	Relationships can be easily seen through the use of arrows and symbols.	It is more difficult to determine the relationships between tables.
Mapping	Has the possibility to show cardinality mapping.	Not possible to see cardinality mapping

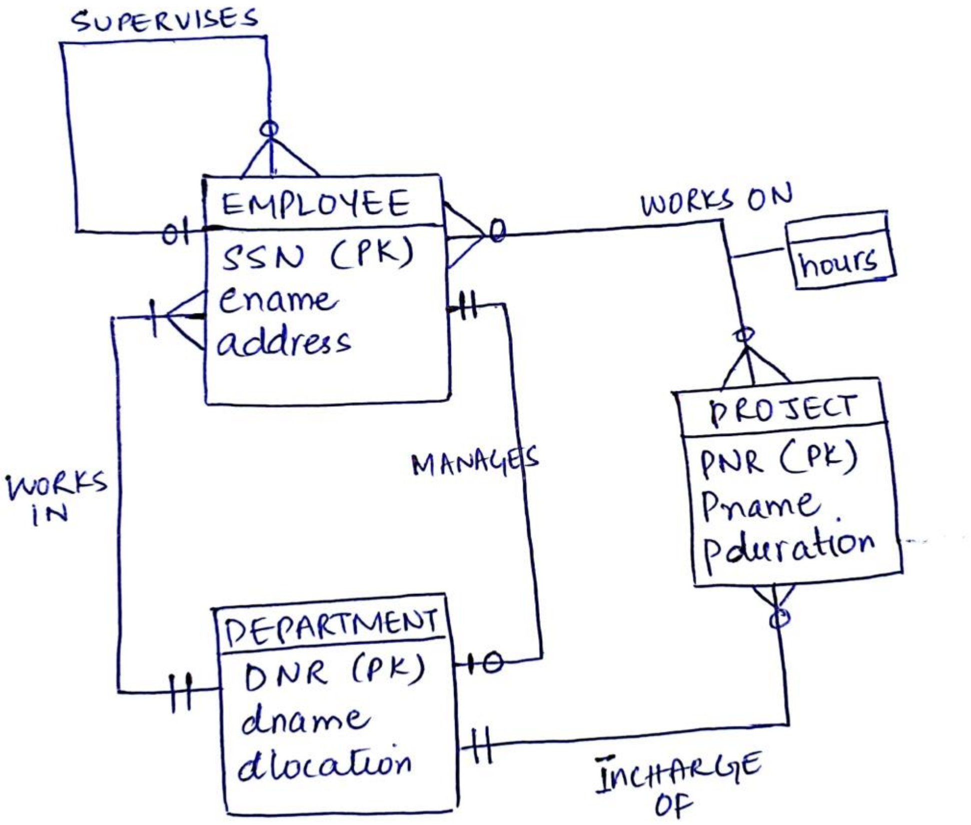
Developing an ERD

Building an ERD usually involves the following activities:

1. Create a detailed narrative of the organization’s description of operations.
2. Identify the **business rules** based on the description of operations.
3. Identify the main *entities* and *relationships* from the business rules.
4. Develop the *initial ERD.*
5. Identify the *attributes* and *primary keys* that adequately describe the entities.
6. Revise and review the ERD. (**iterative** process) 🔄

ER Model


Super vises
Super-vised by
0..N
0..1
SSN
ename
address
EMPLOYEE
0..M
pro-emp
emp-pro
hours
WORKS ON
pnr
0..N
PROJECT
1..N
1..1
WORKS IN
dep-emp
emp-dep
dep-mgr
mgr-dep
MANAGES
1..1
0..1
0..N
IN CHARGE OF
DEPARTMENT
1..1
pro-dep
dep-pro
pname
pduration
dnr
dname
dlocation


SUPERVISES
EMPLOYEE
SSN (PK)
ename
address
WORKS ON
hours
PROJECT
PNR (PK)
Pname
Pduration
WORKS IN
MANAGES
DEPARTMENT
DNR (PK)
dname
dlocation
INCHARGE OF

One to One Relation

- **Rule**: In a *one-to-one* relationship, each record in one table is associated with exactly one record in another table, and vice versa. To move a PK to FK in this scenario:
  - Create a new table (if one doesn't exist already) for one of the related entities.
  - Add a FK column in either of the related tables that references the PK of the other table.
  - Better to add to the side of total participation (NOT NULL)
  - *Ensure that the FK column has a UNIQUE constraint to maintain the one-to-one relationship.*

One to Many Relation

- **Rule**: In a *one-to-many* relationship, one record in one table can be associated with multiple records in another table.
  To move a PK to FK in this scenario:
  - The table with the "one" side of the relationship (the parent table) should have the PK.
  - The table with the "many" side of the relationship (the child table) should have a FK column that references the PK of the parent table.
  - Ensure that the FK column in the child table enforces referential integrity, meaning it should only allow values that exist in the parent table's PK column. (FOREIGN KEY constraint)

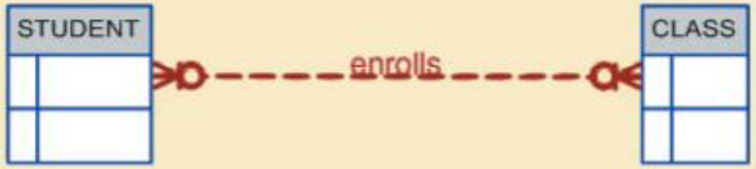
Many to Many Relation

- **Rule**: In a *many-to-many* relationship, multiple records in one table can be associated with multiple records in another table.
  To move a PK to FK in this scenario:
  - Create an **intermediary table** (also known as a junction, linking or bridging table) that contains FK columns referencing the PKs of both related tables.
  - The PK of the intermediary table often consists of a combination of the two FK columns.
  - Each FK column in the intermediary table should enforce referential integrity to the respective tables it references.
  - The intermediary table allows you to model the M:M relationship by storing pairs of related records.

Implementing Many to Many relationship

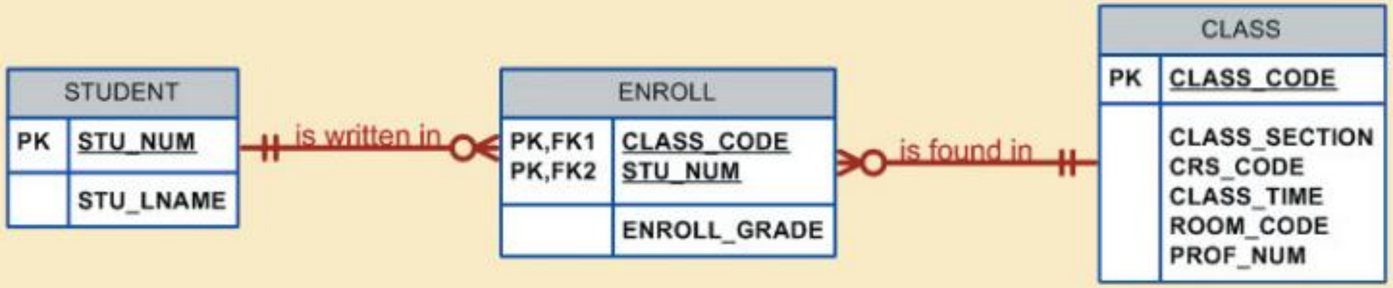
- Create a bridge table for M:M relations
- Split into 2 One-to-Many relations
- Both FKs form the composite key for this table.
- If the relation was ternary, the bridge will contain 3 FKs that make up the PK.

FIGURE 4.24 The M:N relationship between STUDENT and CLASS

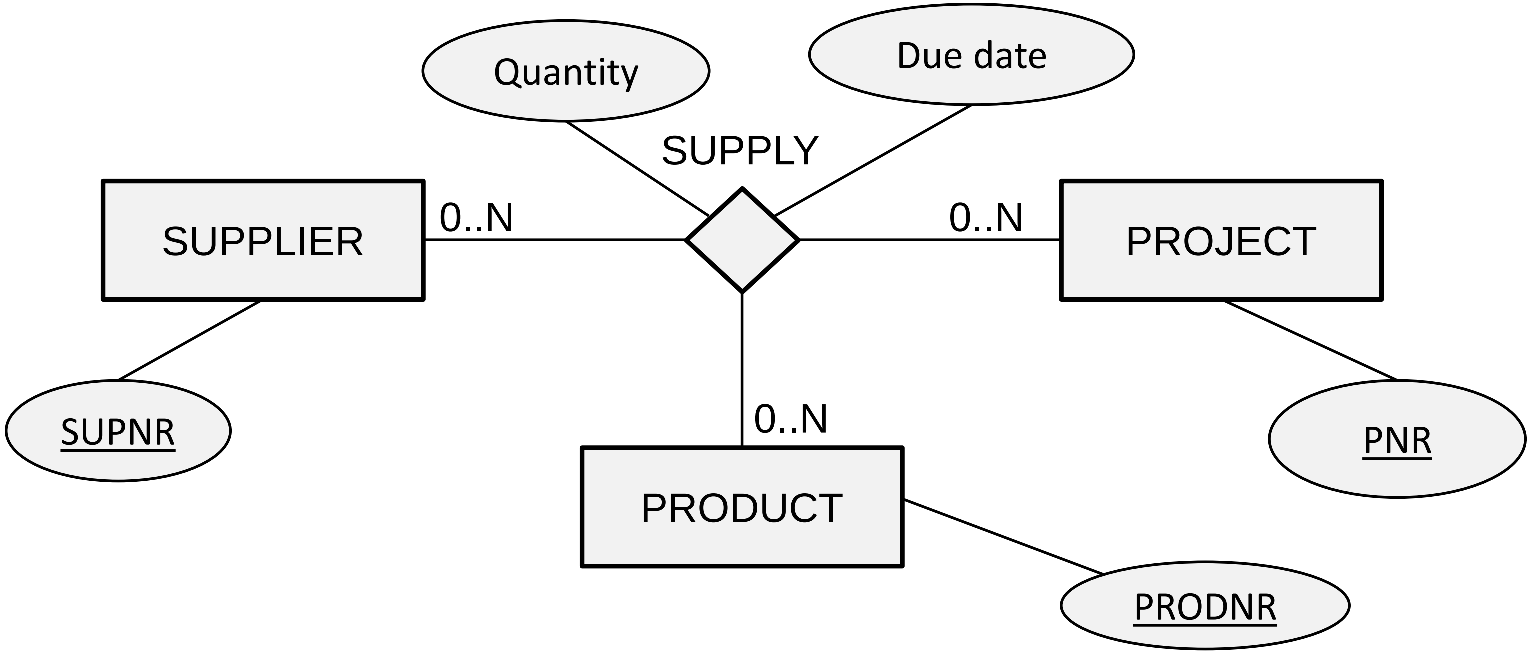


Visio does not permit the definition of a M:N relationship. To make this illustration, two 1:M relationships have been superimposed.

FIGURE 4.25 A composite entity in an ERD



Ternary Relation - Supply



Relational Schema:

Supplier (<u>SUPNR</u>)

Project (<u>PNR</u>)

Product (<u>PRODNR</u>)

Supply (<u>SUPNR</u>, <u>PNR</u>, <u>PRODNR</u>, Due_date, Quantity)

Quiz 01

30 mins

Multi-valued Attributes

CAR has **multiple** colors.

1. *Within the original entity, create several new attributes, one for each of the original multivalued attribute's components.*
   - For example, the CAR entity's attribute CAR_COLOR can be split to create the new attributes: CAR_**TOP**COLOR, CAR_**BODY**COLOR, and CAR_**TRIM**COLOR, which are then assigned to the CAR entity

- What if values are no longer 3? 4? 5?

Multi-valued Attributes

2. *Create a new entity composed of the original multivalued attribute's components.*
   - This new entity allows the designer to define color for different sections of the car.
   - Then, this new CAR_COLOR entity is related to the original CAR entity in a **1:M** relationship.

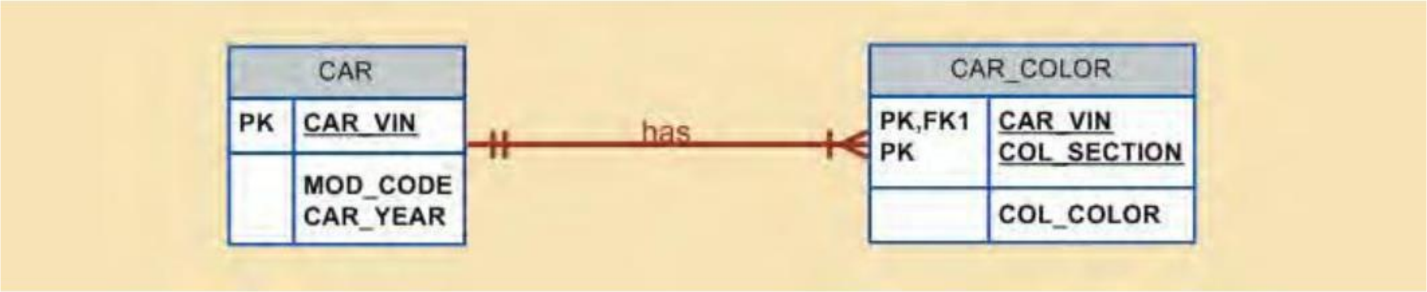
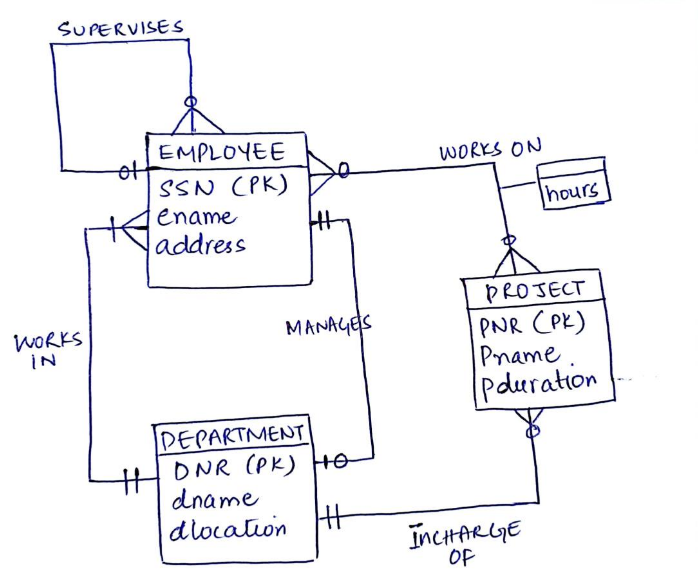
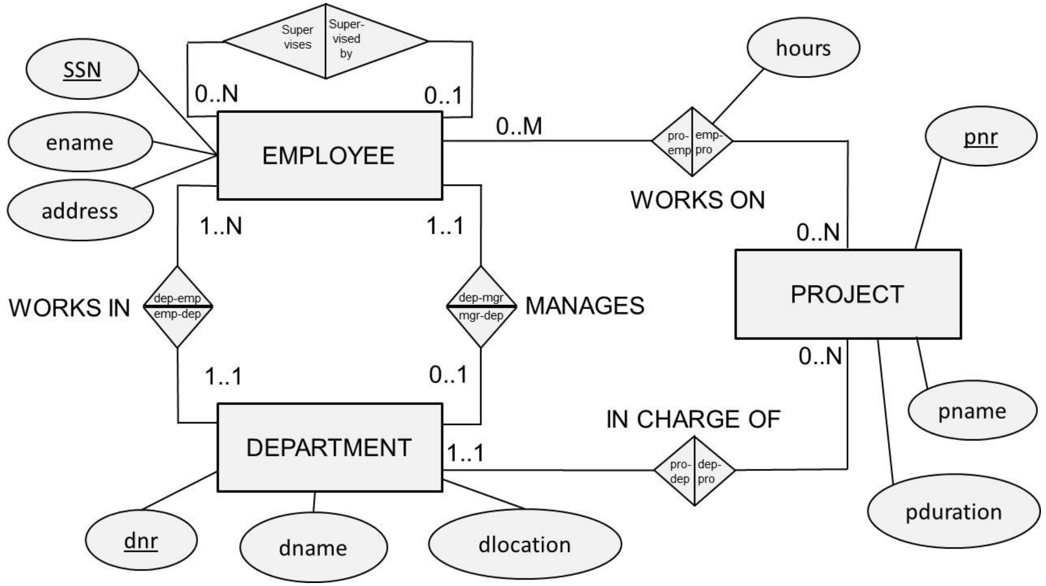


Fig. 4.6 A new entity set composed of a multivalued attribute's components

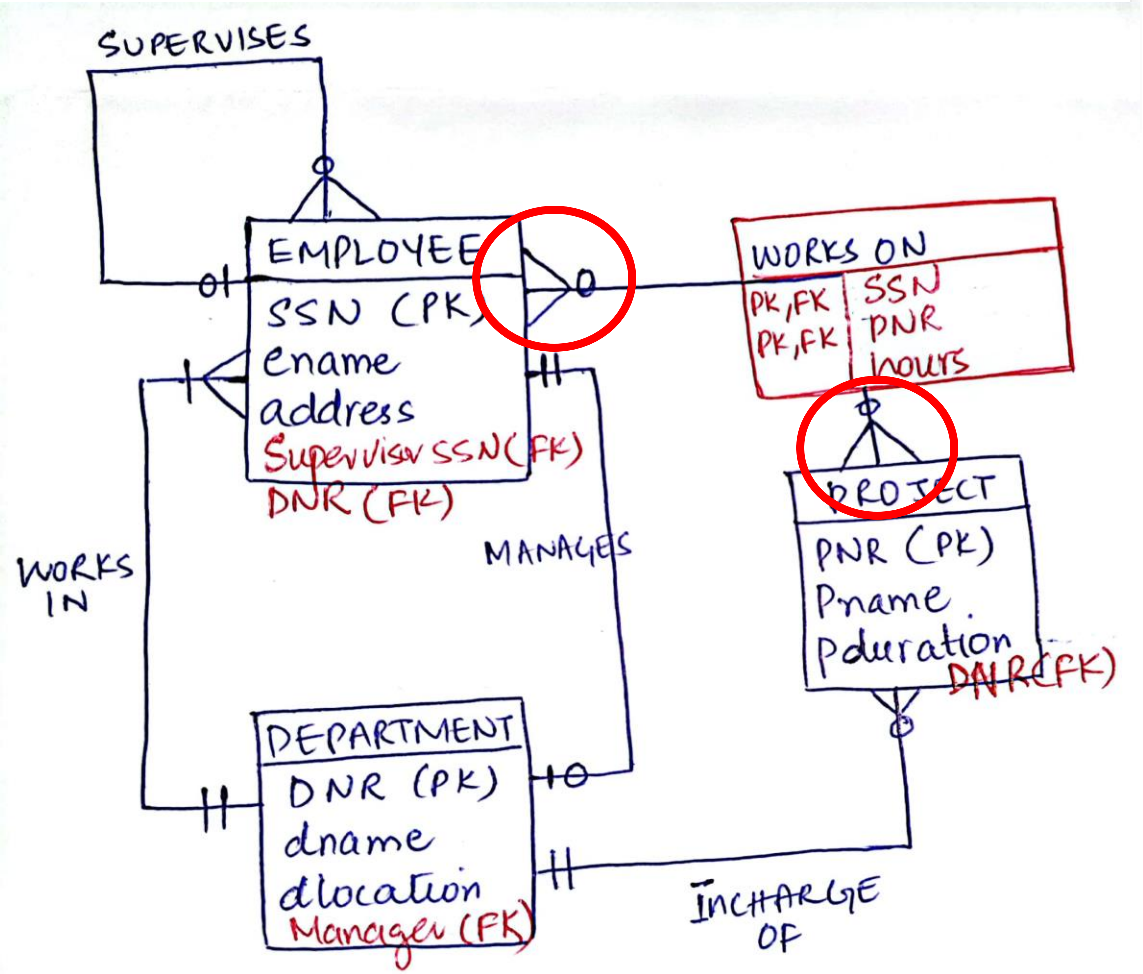
Derived Attributes

- The derived attribute need not be physically stored within the database; instead, it can be derived by using an algorithm

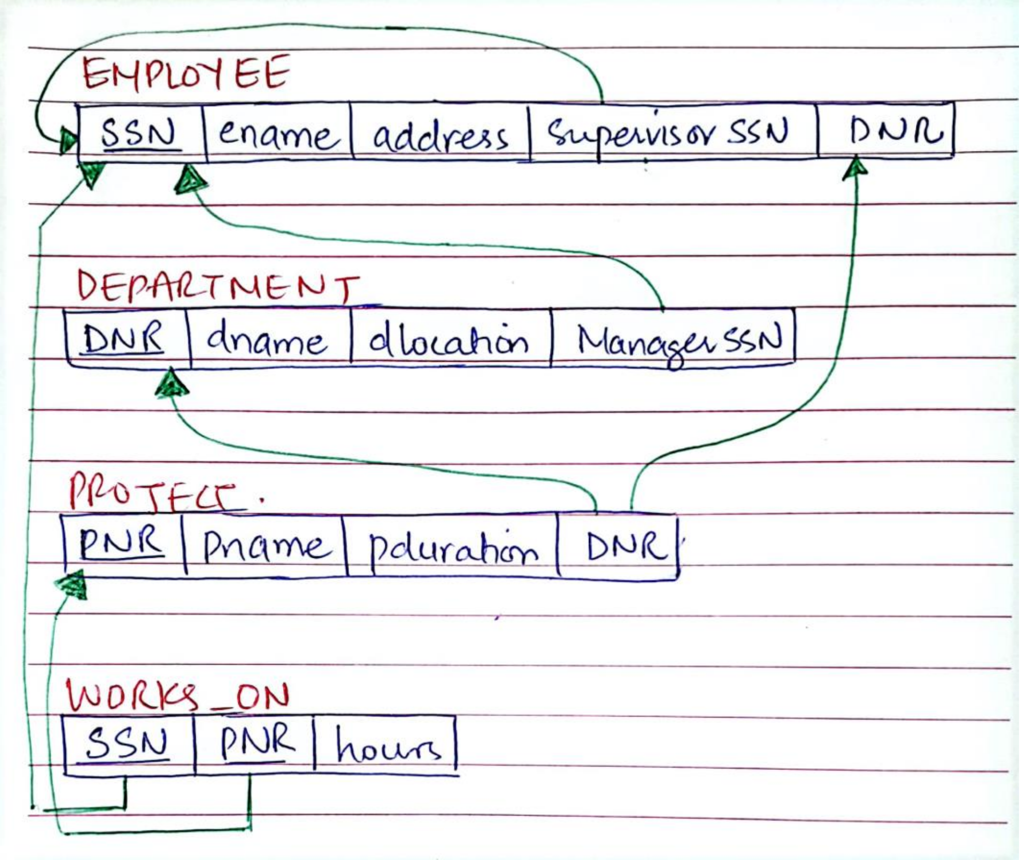




SSN
ename
address
Super vises
Super-vised by
0..N
0..1
EMPLOYEE
0..M
pro-emp
emp-pro
hours
WORKS ON
0..N
pnr
PROJECT
1..N
1..1
WORKS IN
dep-emp
emp-dep
dep-mgr
mgr-dep
MANAGES
1..1
0..1
0..N
IN CHARGE OF
pro-dep
dep-pro
1..1
DEPARTMENT
pname
pduration
dnr
dname
dlocation



Relational Schema



EMPLOYEE

SSN	ename	address	Supervisor SSN	DNR

DEPARTMENT

DNR	dname	dlocation	Manager SSN

PROJECT.

PNR	Pname	pduration	DNR

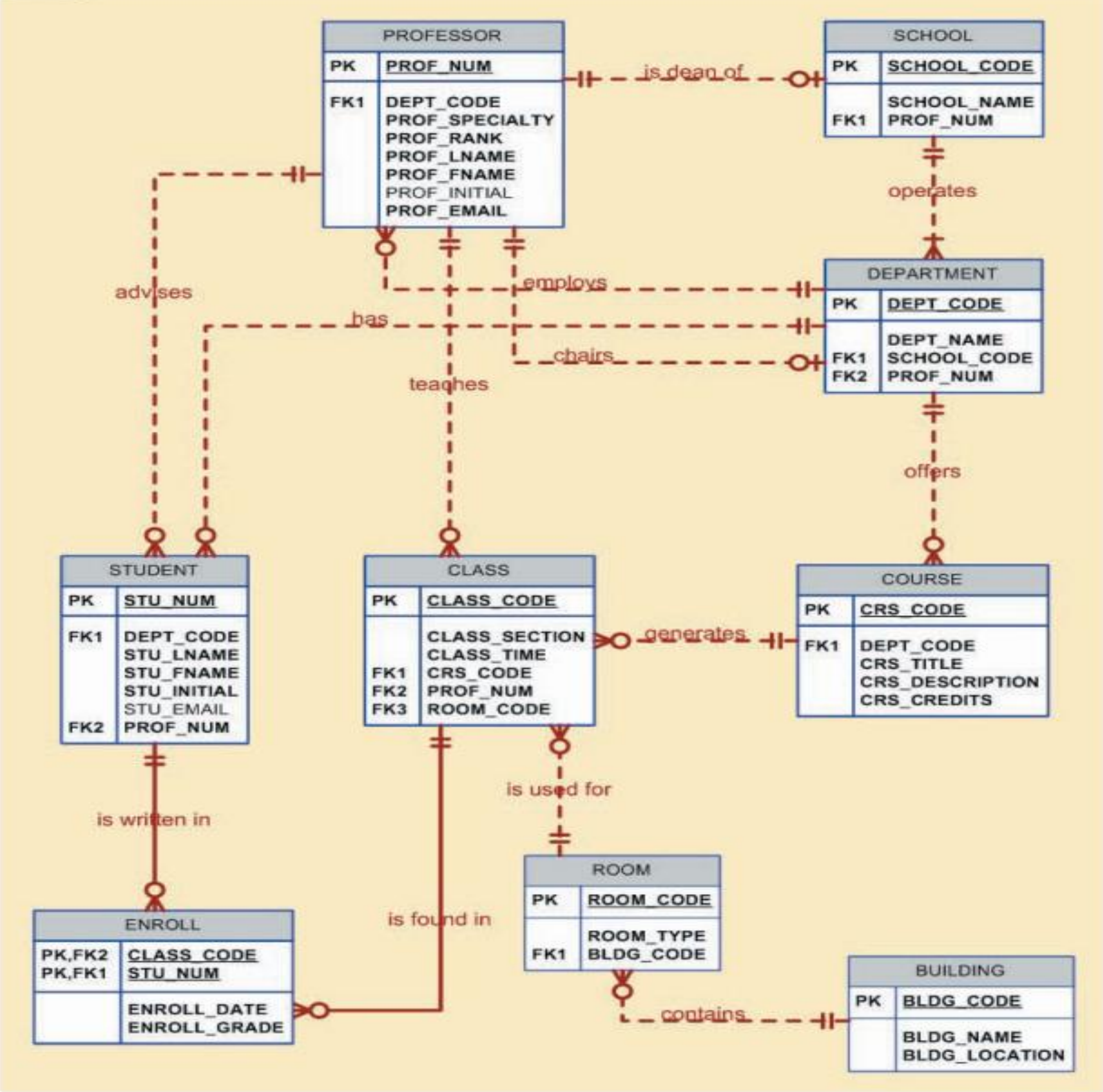
WORKS_ON

SSN	PNR	hours

- Can be represented as table headers
- Or comma separated values within parenthesis
- Or entities without the mention of cardinalities (e.g., HR schema)

(The reader of this schema will have to infer the cardinalities by reading the foreign keys)

FIGURE 4.35 The completed Tiny College ERD



Scenario – Flight Database

Business Rules

Complete Relational Schema in Crow's foot notation

- The airline has one or more airplanes.
- An airplane has a model number, a unique registration number, and the capacity to take one or more passengers.
- An airplane flight has a unique flight number, a departure airport, a destination airport, a departure date and time, and an arrival date and time.
- Each flight is carried out by a single airplane.
- A passenger has given names, a surname, and a unique email address.
- A passenger can book a seat on a flight.

Scenario – Flight Database

Business Rules

Complete Relational Schema in Crow's foot notation

- The **airline** **has** one or more airplanes.
- An **airplane** has a model number, a unique registration number, and the capacity to take one or more passengers.
- An airplane **flight** has a unique flight number, a departure airport, a destination airport, a departure date and time, and an arrival date and time.
- Each flight is **carried out** by a single airplane.
- A **passenger** has given names, a surname, and a unique email address.
- A passenger can **book** a seat on a flight.