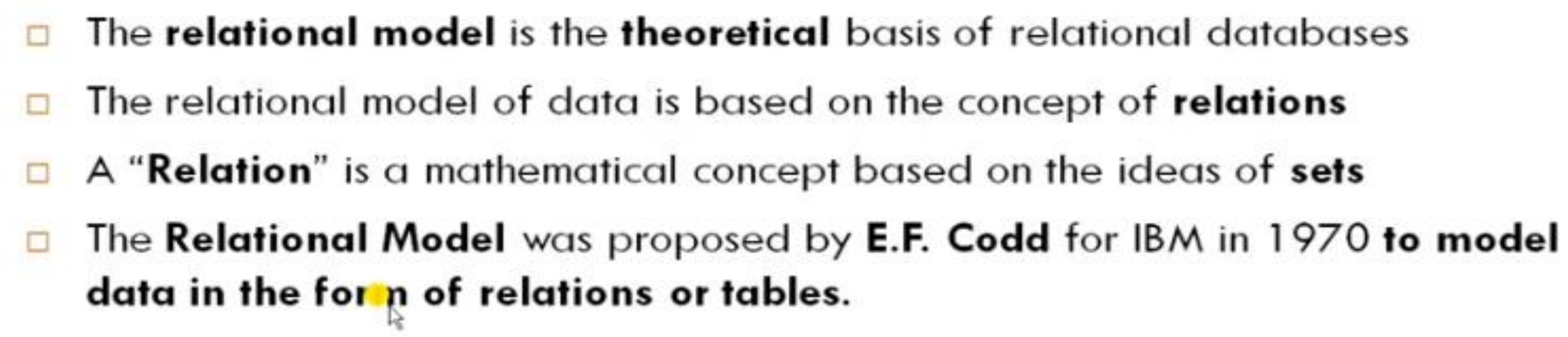
Relational Model and Relational Algebra

Dr. Jyoti Wadmare

Contents

- Introduction to the Relational Model,
- Relational schema and concept of keys.
- Mapping the ER and EER Model to the Relational Model
- Relational Algebra-operators,
- Relational Algebra Queries

Introduction to Relational Model



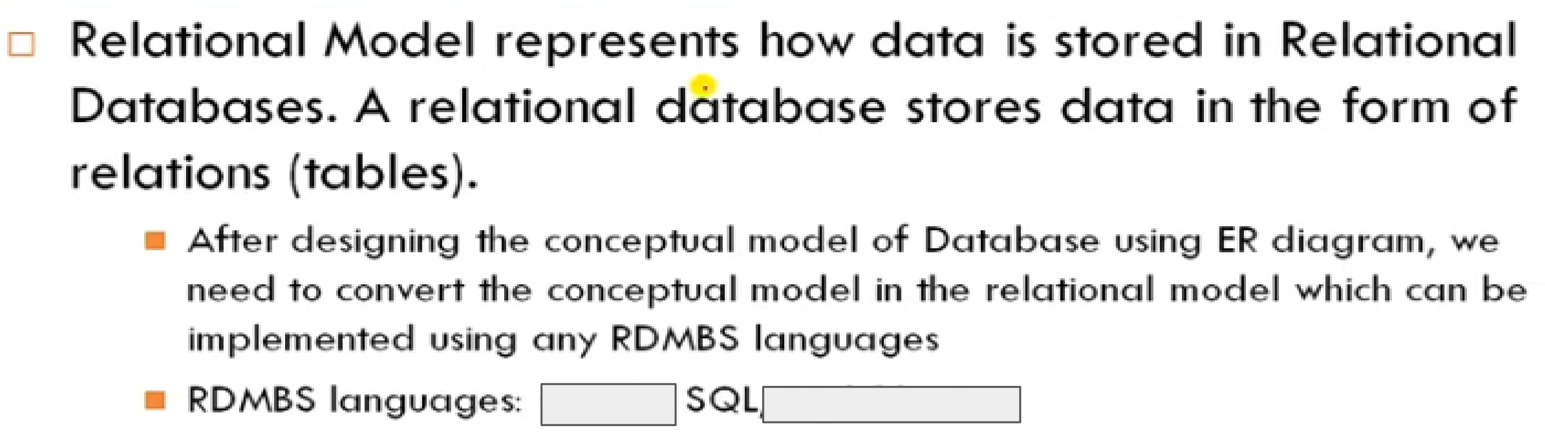
□ The **relational model** is the **theoretical** basis of relational databases

□ The relational model of data is based on the concept of **relations**

□ A “**Relation**” is a mathematical concept based on the ideas of **sets**

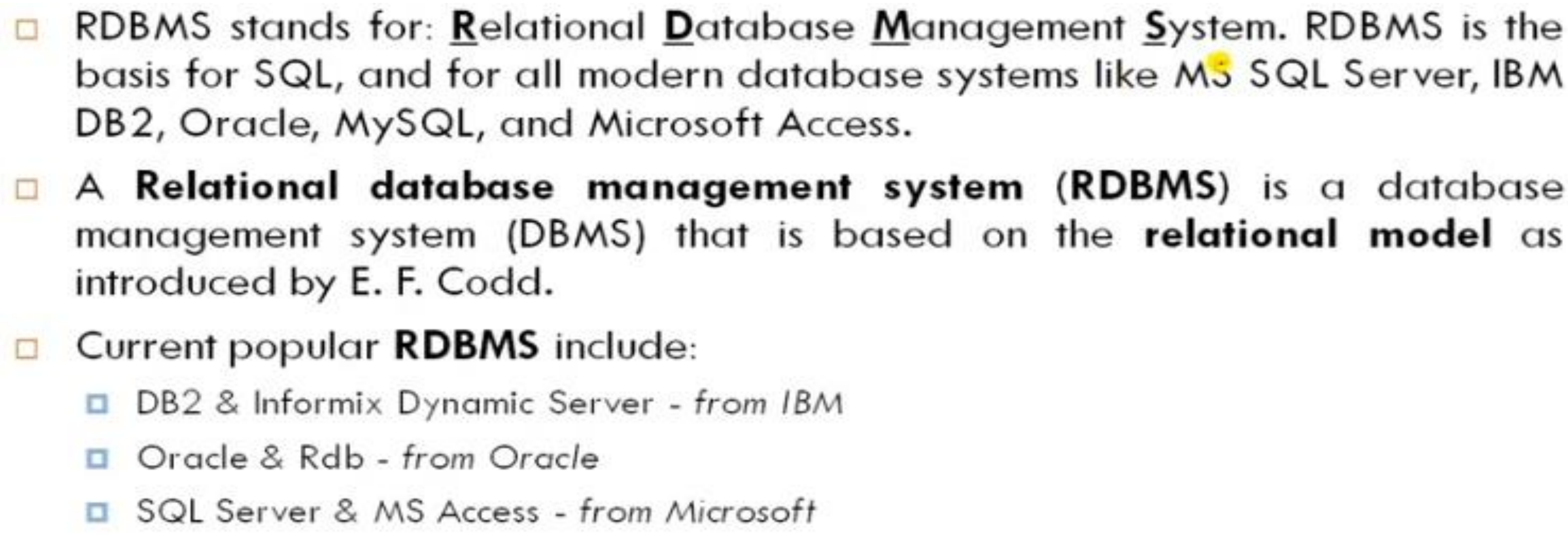
□ The **Relational Model** was proposed by **E.F. Codd** for IBM in 1970 **to model data in the form of relations or tables.**

Introduction to Relational Model



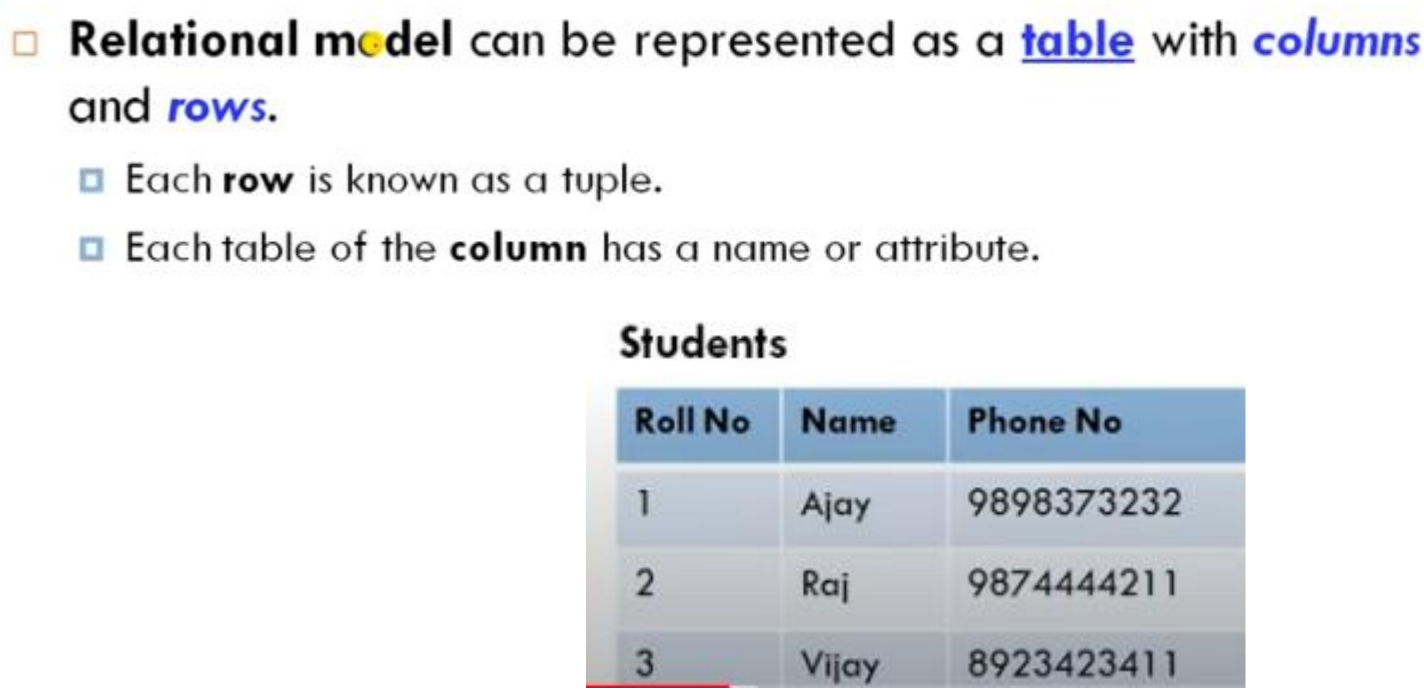
- Relational Model represents how data is stored in Relational Databases. A relational database stores data in the form of relations (tables).
  - After designing the conceptual model of Database using ER diagram, we need to convert the conceptual model in the relational model which can be implemented using any RDMBS languages
  - RDMBS languages: [redacted] SQL, [redacted]

Relational Model



- RDBMS stands for: **R**elational **D**atabase **M**anagement **S**ystem. RDBMS is the basis for SQL, and for all modern database systems like MS SQL Server, IBM DB2, Oracle, MySQL, and Microsoft Access.
- A **Relational database management system** (**RDBMS**) is a database management system (DBMS) that is based on the **relational model** as introduced by E. F. Codd.
- Current popular **RDBMS** include:
  - DB2 & Informix Dynamic Server - *from IBM*
  - Oracle & Rdb - *from Oracle*
  - SQL Server & MS Access - *from Microsoft*

Relational Model Concept



- **Relational model** can be represented as a <u>***table***</u> with ***columns*** and ***rows***.
  - Each **row** is known as a tuple.
  - Each table of the **column** has a name or attribute.

Students

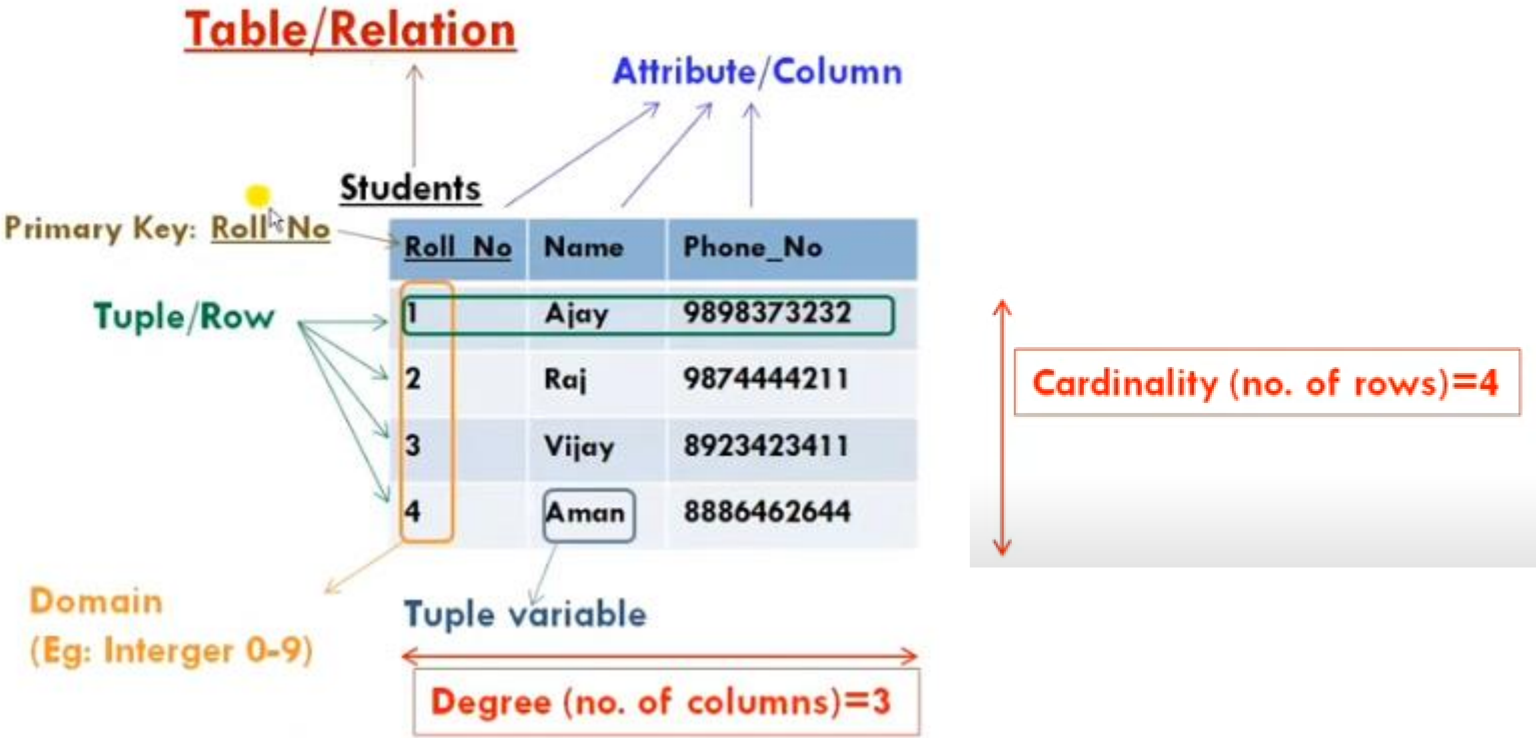
Roll No	Name	Phone No
1	Ajay	9898373232
2	Raj	9874444211
3	Vijay	8923423411

Students

Roll_No	Name	Phone_No
1	Ajay	9898373232
2	Raj	9874444211
3	Vijay	8923423411
4	Aman	8886462644

Relational Model concept

- □ **Relation:** A relation is a table with columns and rows.
- □ **Attribute:** An attribute is a named column of a relation.
- □ **Domain:** A domain is the set of allowable values for one or more attributes.
- □ **Tuple:** A tuple is a row of a relation.
- □ **Relation Schema:** A relation schema represents the name of the relation with its attributes.
- □ **Relation instance** (State): Relation instance is a finite set of tuples. Relation instances never have duplicate tuples.
- □ **Degree:** The total number of columns or attributes in the relation
- □ **Cardinality:** Total number of rows present in the Table.
- □ **Relation key:** Every row has one or multiple attributes, that can uniquely identify the row in the relation, which is called relation key (Primary key).



Table/Relation

Attribute/Column

Students

Primary Key: Roll_No

Roll_No	Name	Phone_No
1	Ajay	9898373232
2	Raj	9874444211
3	Vijay	8923423411
4	Aman	8886462644

Tuple/Row

Cardinality (no. of rows)=4

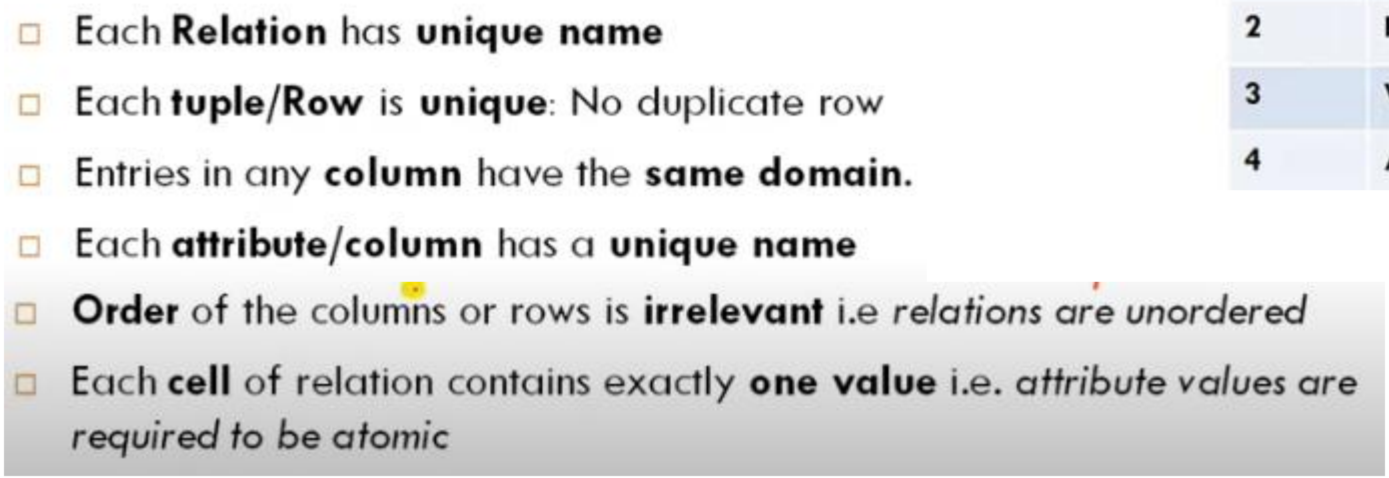
Domain
(Eg: Interger 0-9)

Tuple variable

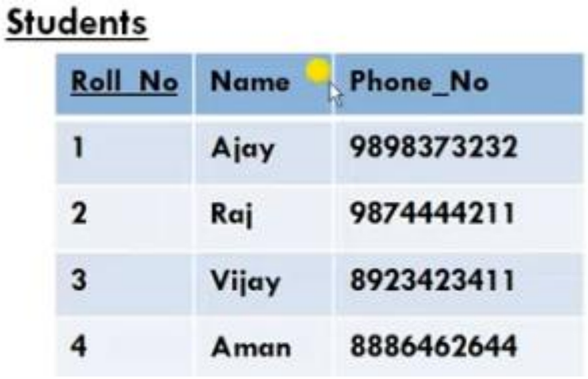
Degree (no. of columns)=3

Relation Schema: Students (Roll_No, Name, Phone_No)

Properties of Relational Model



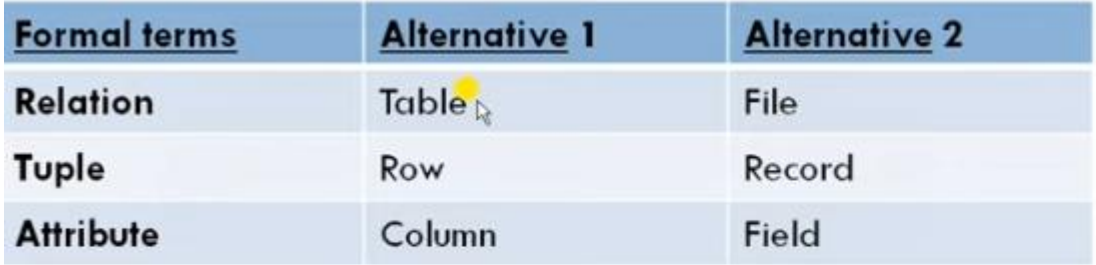
- □ Each **Relation** has **unique name**
- □ Each **tuple/Row** is **unique**: No duplicate row
- □ Entries in any **column** have the **same domain.**
- □ Each **attribute/column** has a **unique name**
- □ **Order** of the columns or rows is **irrelevant** i.e *relations are unordered*
- □ Each **cell** of relation contains exactly **one value** i.e. *attribute values are required to be atomic*



Students

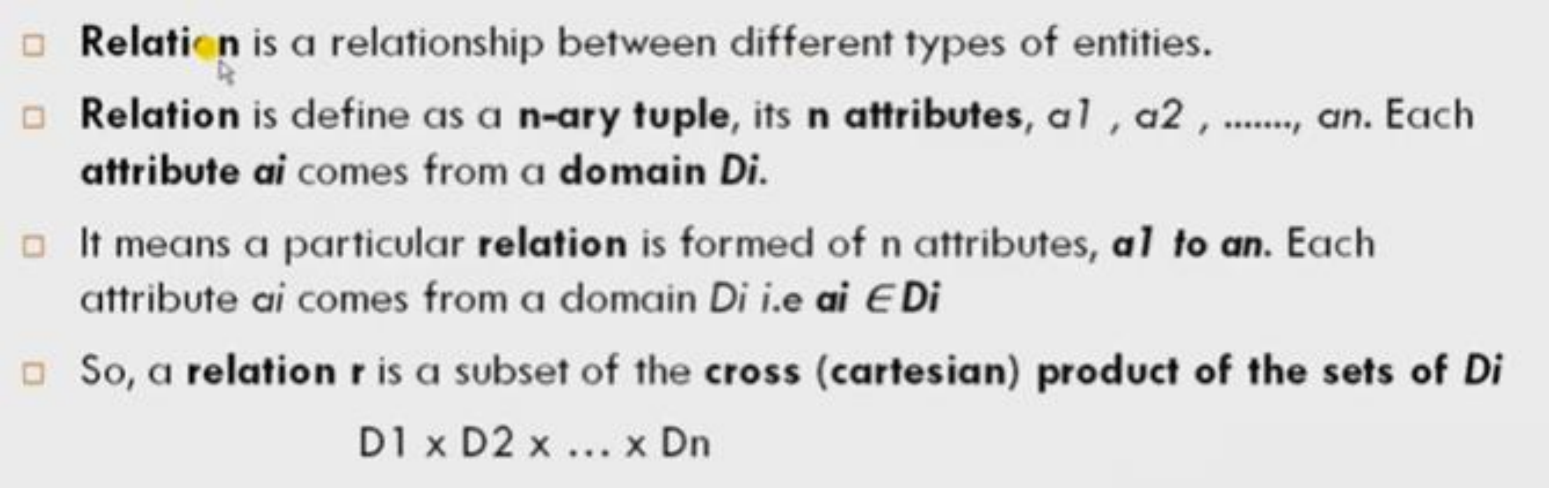
Roll_No	Name	Phone_No
1	Ajay	9898373232
2	Raj	9874444211
3	Vijay	8923423411
4	Aman	8886462644

Alternative Terminology for Relational Model



Formal terms	Alternative 1	Alternative 2
Relation	Table	File
Tuple	Row	Record
Attribute	Column	Field

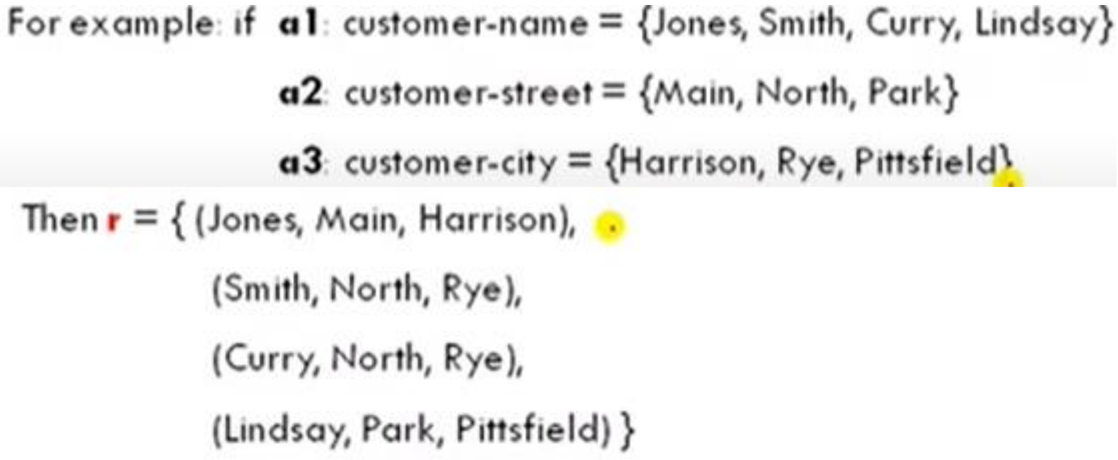
Relational Model- Mathematical structure



- **Relation** is a relationship between different types of entities.
- **Relation** is define as a **n-ary tuple**, its **n attributes**, *a1* , *a2* , ..., *an*. Each **attribute *ai*** comes from a **domain *Di***.
- It means a particular **relation** is formed of n attributes, ***a1* *to an*.** Each attribute *ai* comes from a domain *Di* *i*.**e *ai*** $\in$ ***Di***
- So, a **relation r** is a subset of the **cross (cartesian) product of the sets of *Di***

  D1 x D2 x … x Dn

Example



For example: if **a1**: customer-name = {Jones, Smith, Curry, Lindsay}

a2: customer-street = {Main, North, Park}

a3: customer-city = {Harrison, Rye, Pittsfield}

Then r = { (Jones, Main, Harrison),

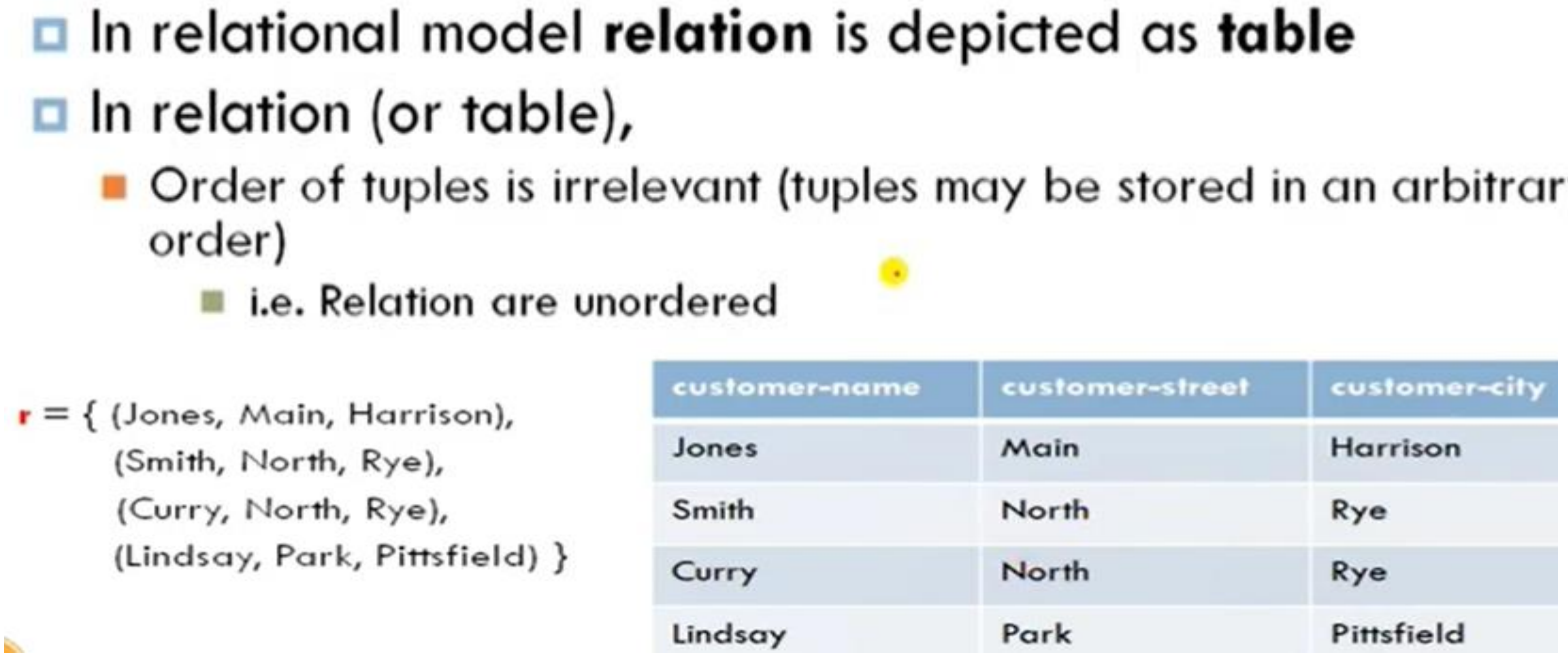
(Smith, North, Rye),

(Curry, North, Rye),

(Lindsay, Park, Pittsfield) }

So Relation is Cartesian product of Customer-name, customer-street, customer-city

Relations are unordered

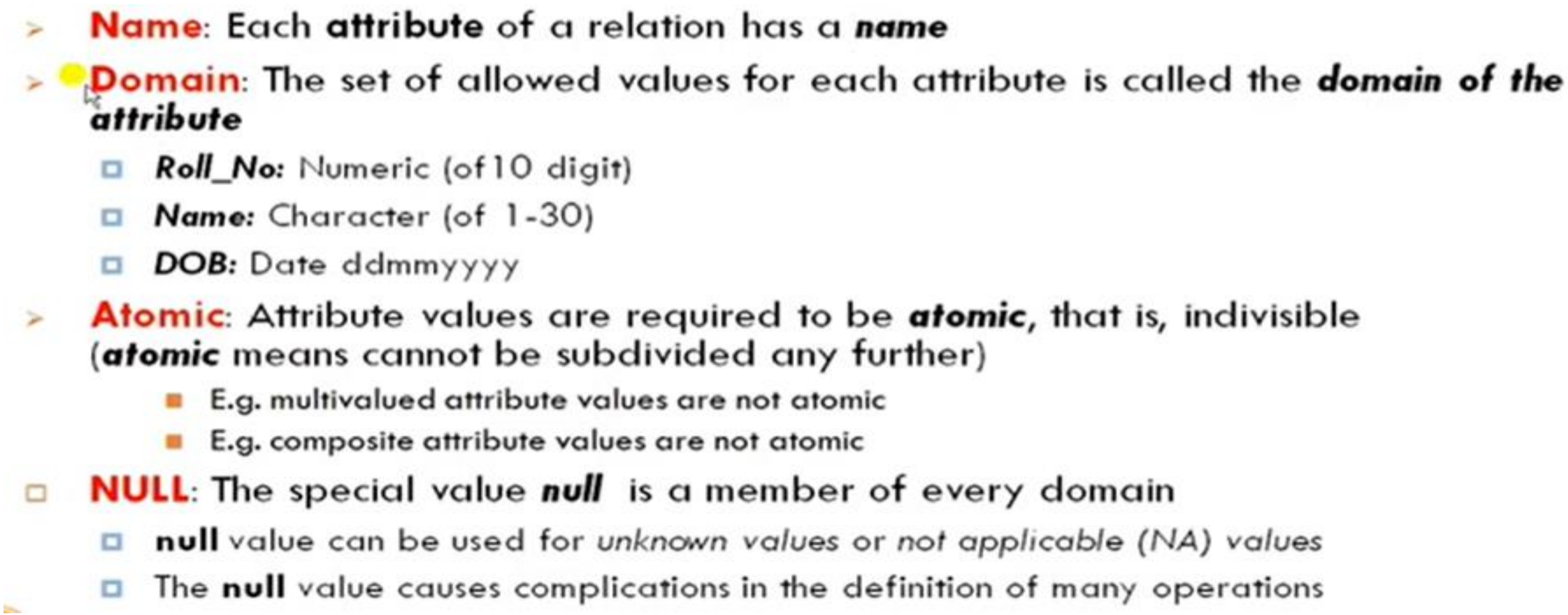


- In relational model **relation** is depicted as **table**
- In relation (or table),
  - Order of tuples is irrelevant (tuples may be stored in an arbitrary order)
    - i.e. Relation are unordered

r = { (Jones, Main, Harrison),
(Smith, North, Rye),
(Curry, North, Rye),
(Lindsay, Park, Pittsfield) }

customer-name	customer-street	customer-city
Jones	Main	Harrison
Smith	North	Rye
Curry	North	Rye
Lindsay	Park	Pittsfield

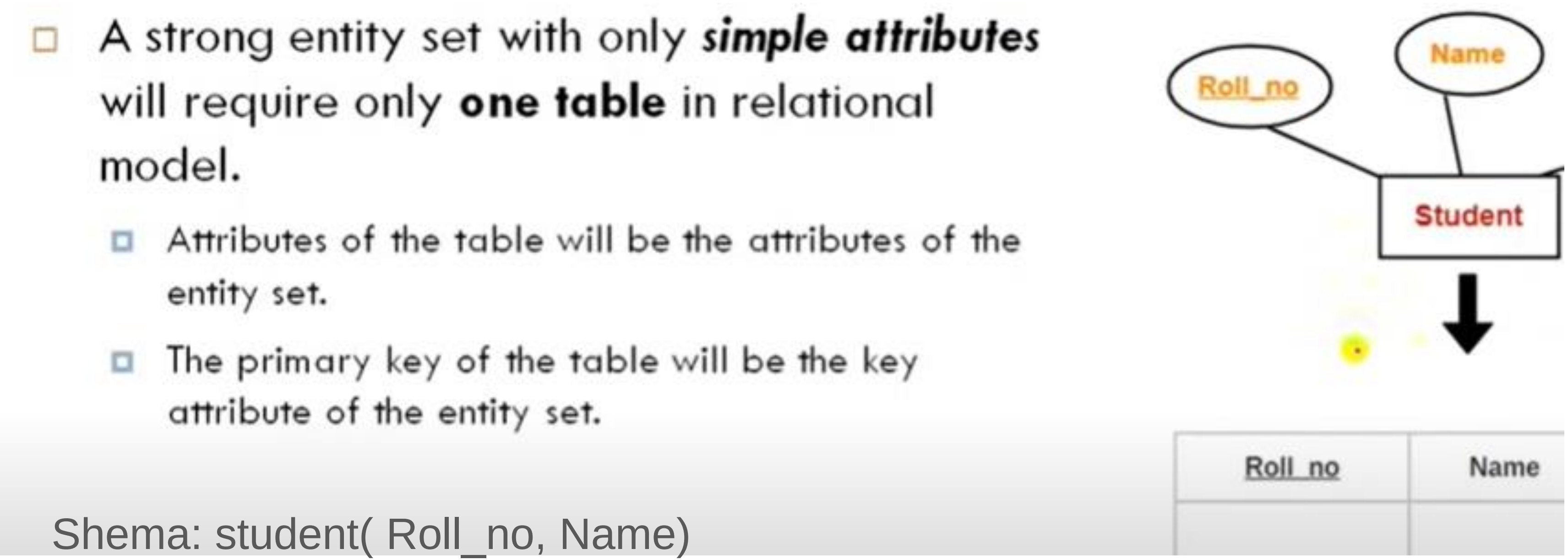
Attributes type



- **Name**: Each **attribute** of a relation has a ***name***
- **Domain**: The set of allowed values for each attribute is called the ***domain of the attribute***
  - ***Roll_No:*** Numeric (of10 digit)
  - ***Name:*** Character (of 1-30)
  - ***DOB:*** Date ddmmyyyy
- **Atomic**: Attribute values are required to be ***atomic***, that is, indivisible (***atomic*** means cannot be subdivided any further)
  - E.g. multivalued attribute values are not atomic
  - E.g. composite attribute values are not atomic
- **NULL**: The special value ***null*** is a member of every domain
  - **null** value can be used for *unknown values* or *not applicable (NA) values*
  - The **null** value causes complications in the definition of many operations

Conversion of ER diagram to Relational model

Rule1: Strong Entity set with simple attributes



- A strong entity set with only ***simple attributes*** will require only **one table** in relational model.
  - Attributes of the table will be the attributes of the entity set.
  - The primary key of the table will be the key attribute of the entity set.

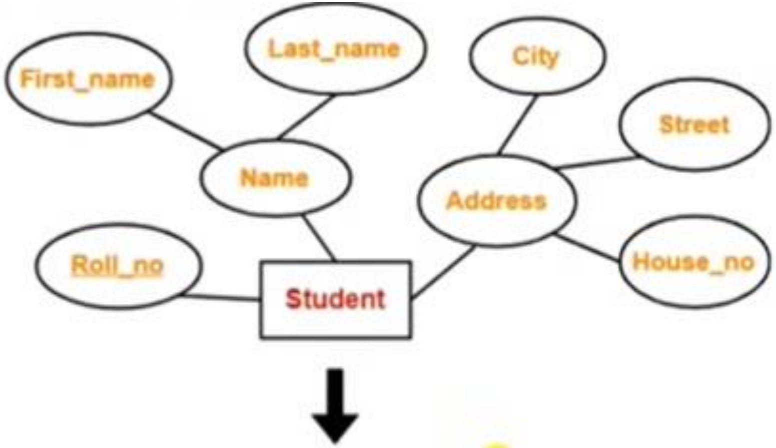
Roll_no	Name

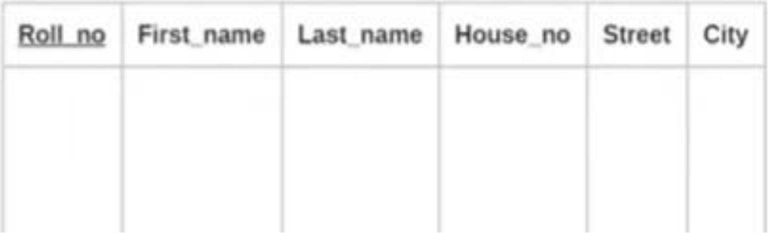
Shema: student(Roll_no, Name)

Rule2: Strong Entity set with composite attributes

□ A strong entity set with any number of ***composite attributes*** will require **only one table** in relational model.

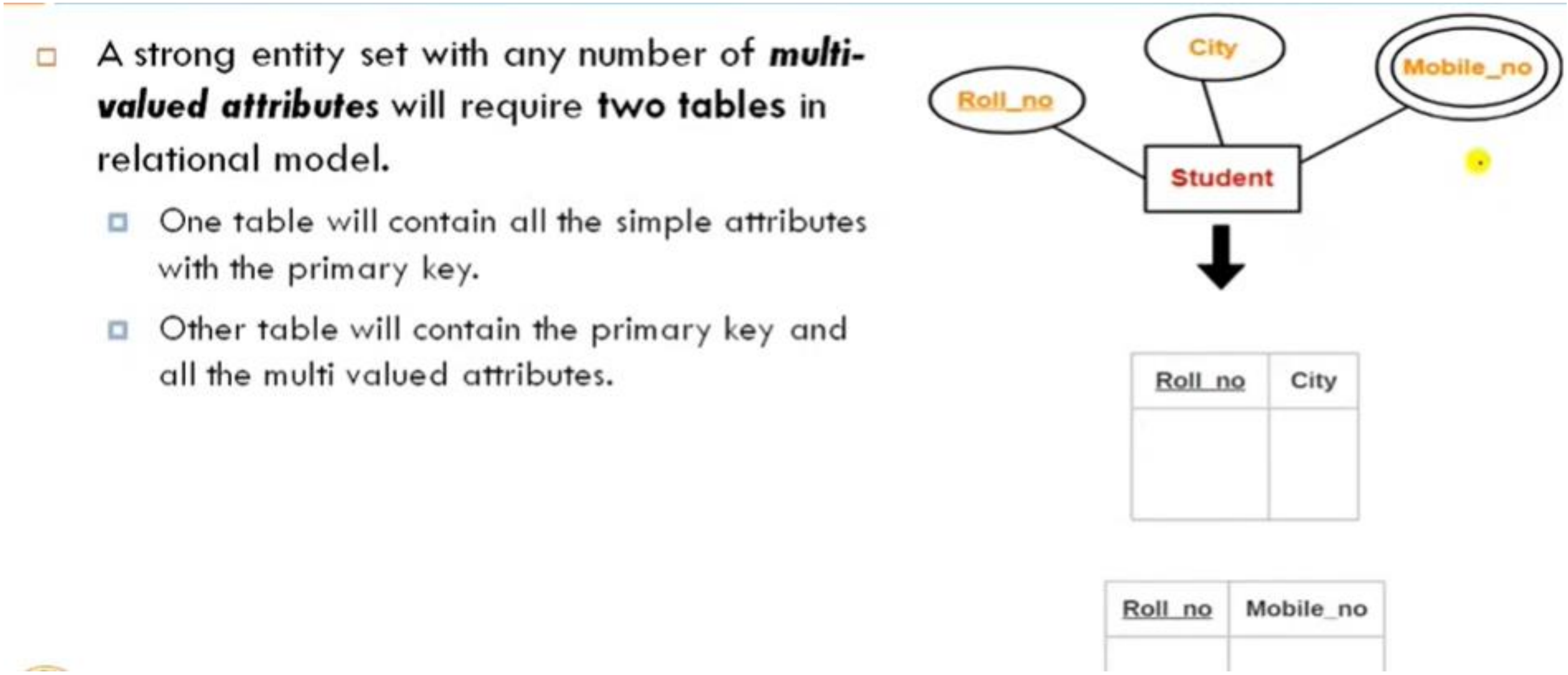
□ While conversion, simple attributes of the composite attributes are taken into account and not the composite attribute itself.





Roll_no	First_name	Last_name	House_no	Street	City

Rule3: Strong Entity set with multivalued attributes

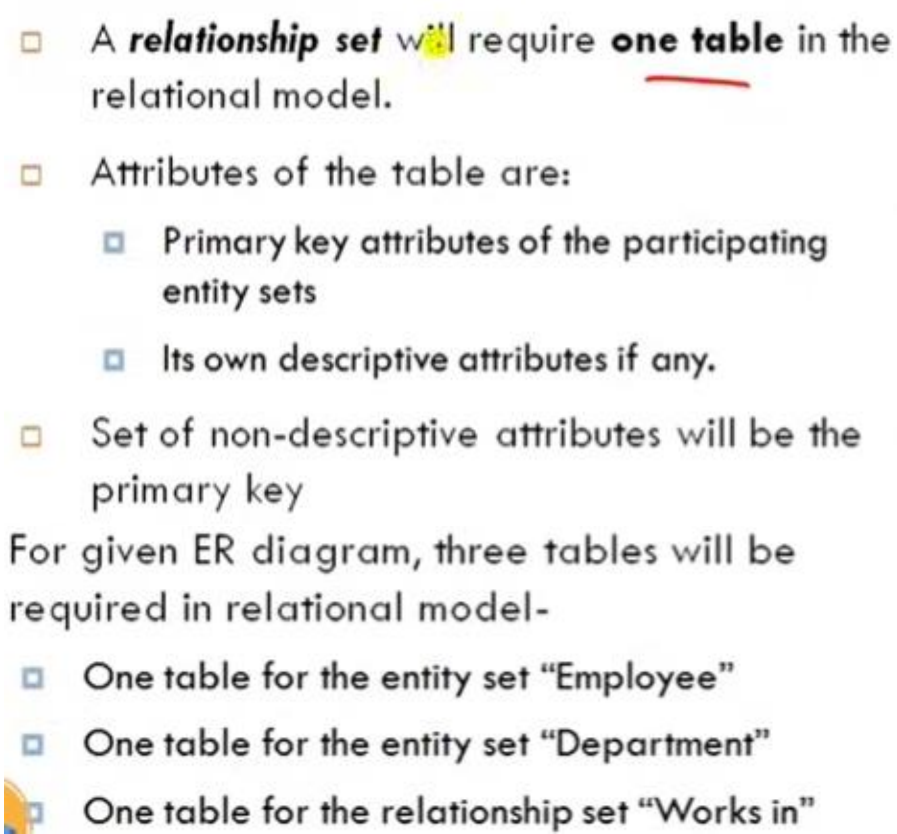


- A strong entity set with any number of ***multi-valued attributes*** will require **two tables** in relational model.
  - One table will contain all the simple attributes with the primary key.
  - Other table will contain the primary key and all the multi valued attributes.

Roll_no	City

Roll_no	Mobile_no

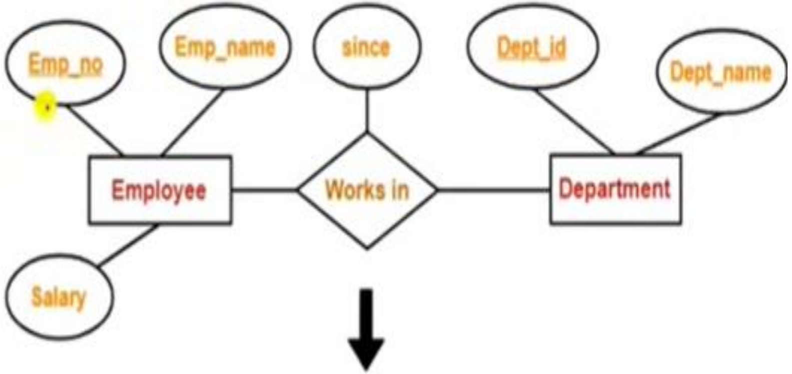
Rule 4: Translating Relationship set into tables

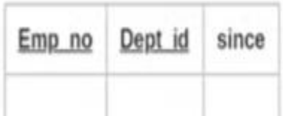


- A ***relationship set*** will require **one table** in the relational model.
- Attributes of the table are:
  - Primary key attributes of the participating entity sets
  - Its own descriptive attributes if any.
- Set of non-descriptive attributes will be the primary key

For given ER diagram, three tables will be required in relational model-

- One table for the entity set "Employee"
- One table for the entity set "Department"
- One table for the relationship set "Works in"

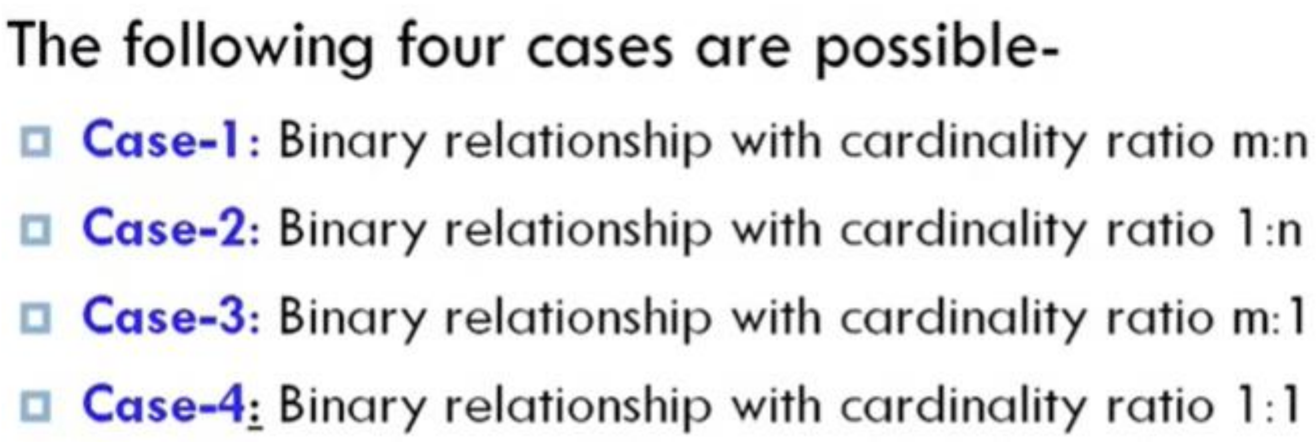




Emp_no	Dept_id	since

Schema: Works in(Emp_no, Dept_id, since)

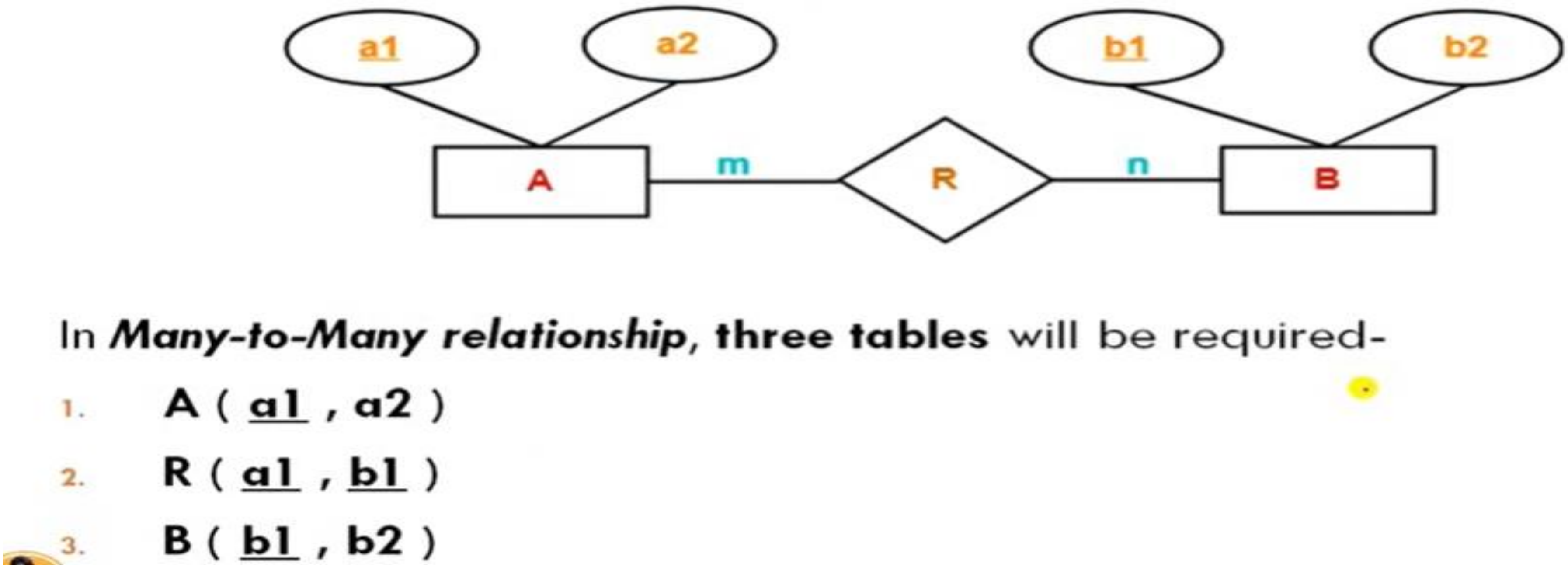
Rule 5: Binary relationships with cardinality ratios



The following four cases are possible-

- □ **Case-1:** Binary relationship with cardinality ratio m:n
- □ **Case-2:** Binary relationship with cardinality ratio 1:n
- □ **Case-3:** Binary relationship with cardinality ratio m:1
- □ **Case-4:** Binary relationship with cardinality ratio 1:1

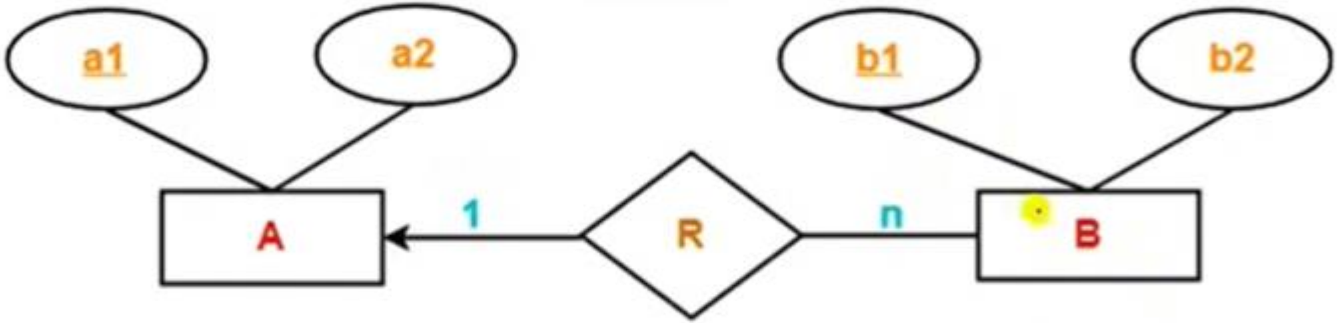
Case1: Binary relationship with cardinality Ratio m:n

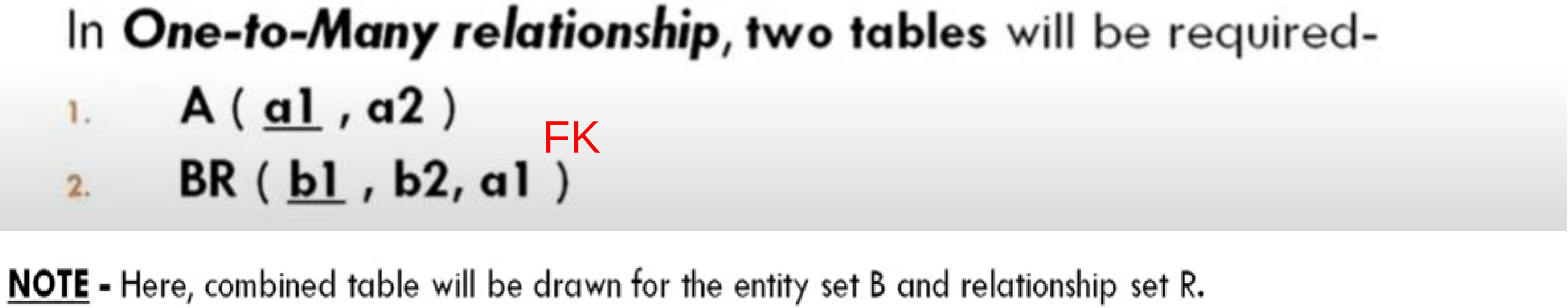


In ***Many-to-Many relationship*****, three tables** will be required-

1. A (a1 , a2)
2. R (a1 , b1)
3. B (b1 , b2)

Case2: Binary relationship with cardinality Ratio 1:n




In **One-to-Many relationship**, **two tables** will be required-

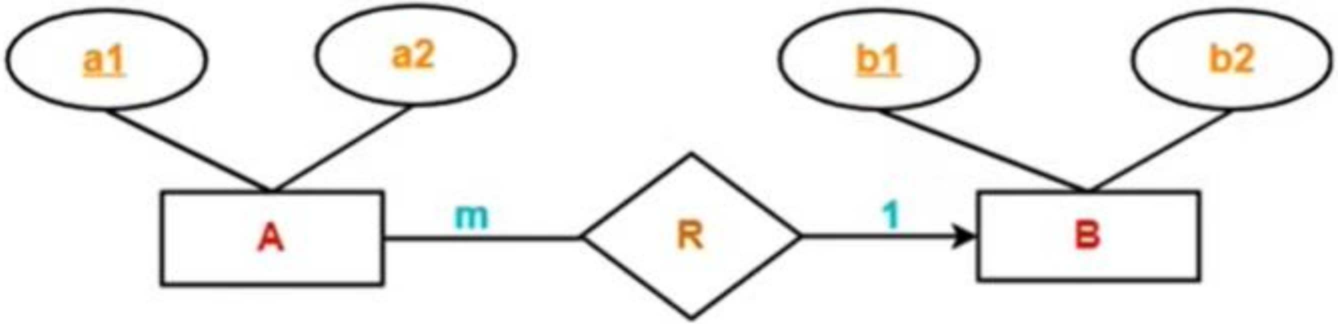
1. A (a1 , a2)

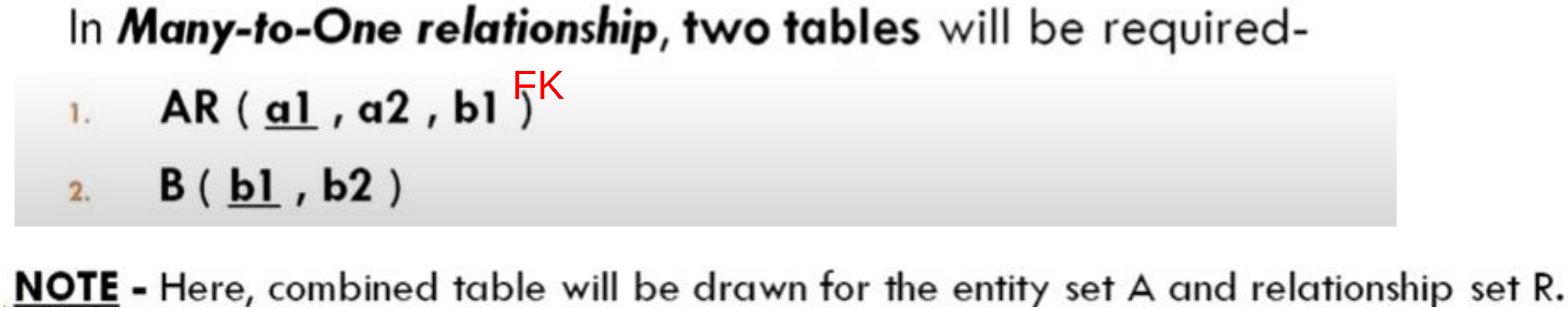
FK

2. BR (b1 , b2, a1)

NOTE - Here, combined table will be drawn for the entity set B and relationship set R.

Case3: Binary relationship with cardinality Ratio m:1

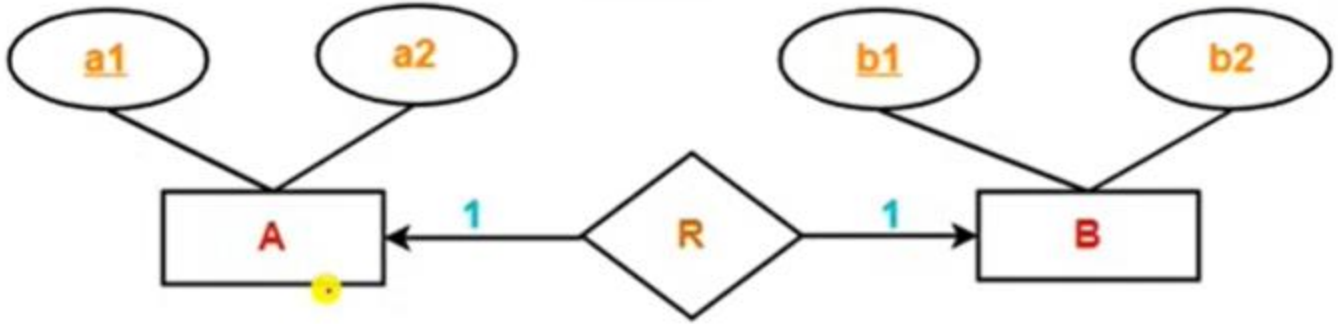


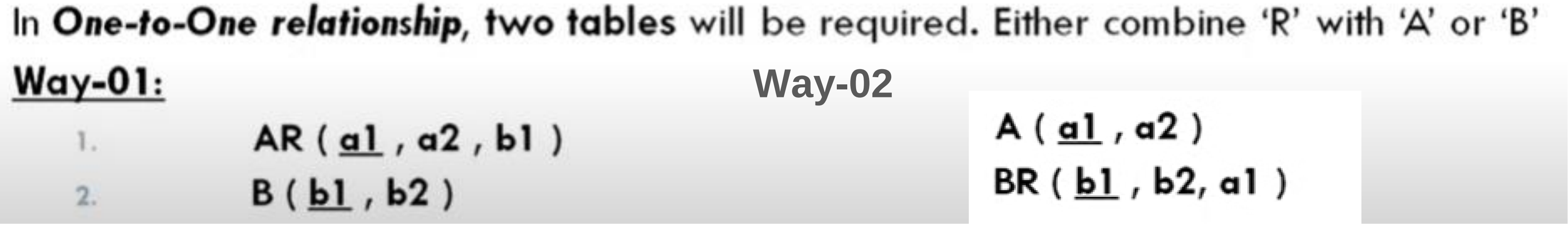

In ***Many-to-One relationship***, **two tables** will be required-

1. AR (a1 , a2 , b1) FK
2. B (b1 , b2)

NOTE - Here, combined table will be drawn for the entity set A and relationship set R.

Case4: Binary relationship with cardinality Ratio 1:1





In **One-to-One relationship**, **two tables** will be required. Either combine 'R' with 'A' or 'B'

Way-01:

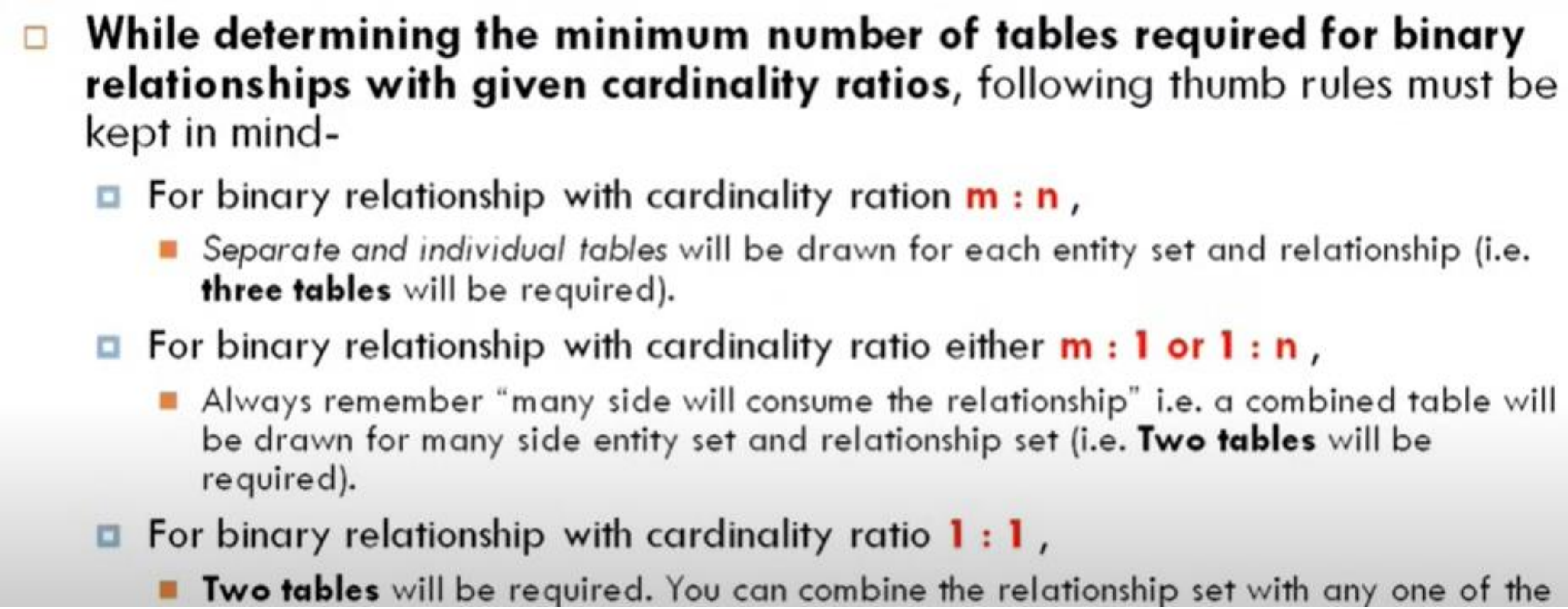
1. AR (a1 , a2 , b1)
2. B (b1 , b2)

Way-02

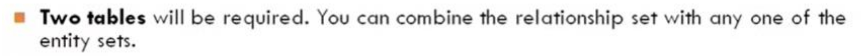
A (a1 , a2)

BR (b1 , b2, a1)

Thumb rules to Remember for determining minimum number of tables

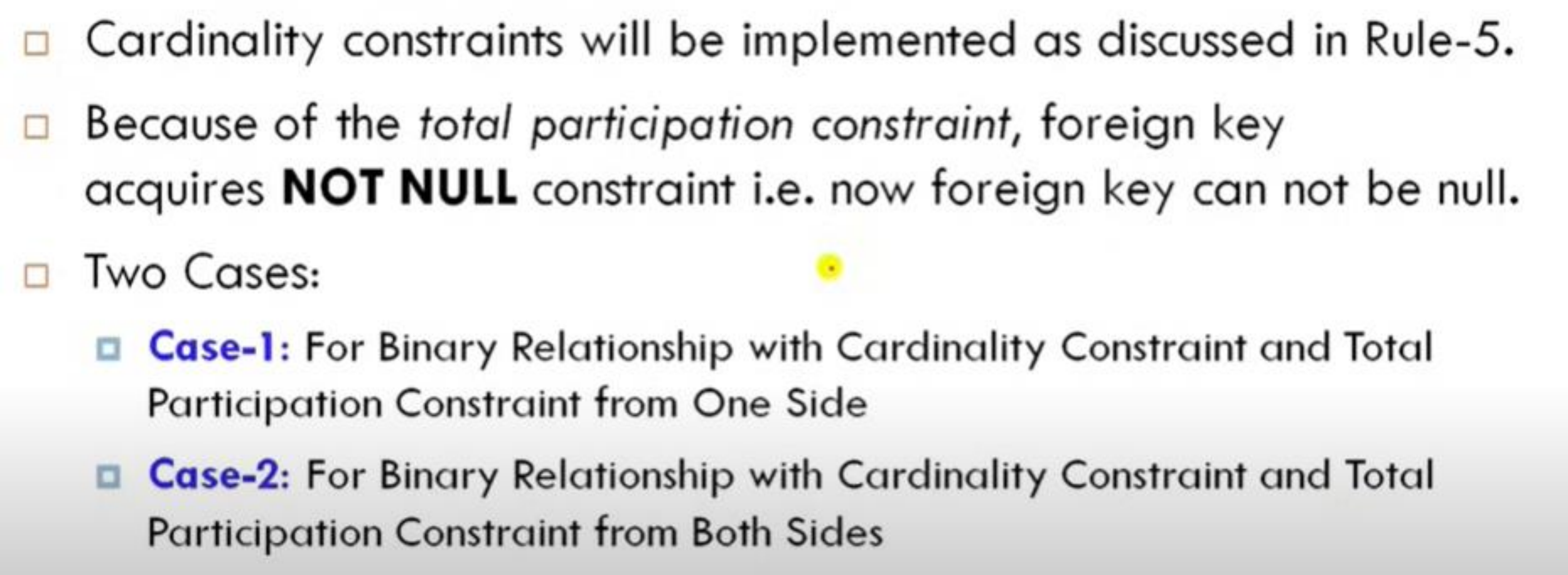


- **While determining the minimum number of tables required for binary relationships with given cardinality ratios**, following thumb rules must be kept in mind-
  - For binary relationship with cardinality ration **m : n** ,
    - *Separate and individual tables* will be drawn for each entity set and relationship (i.e. **three tables** will be required).
  - For binary relationship with cardinality ratio either **m : 1 or 1 : n** ,
    - Always remember "many side will consume the relationship" i.e. a combined table will be drawn for many side entity set and relationship set (i.e. **Two tables** will be required).
  - For binary relationship with cardinality ratio **1 : 1** ,
    - **Two tables** will be required. You can combine the relationship set with any one of the entity sets



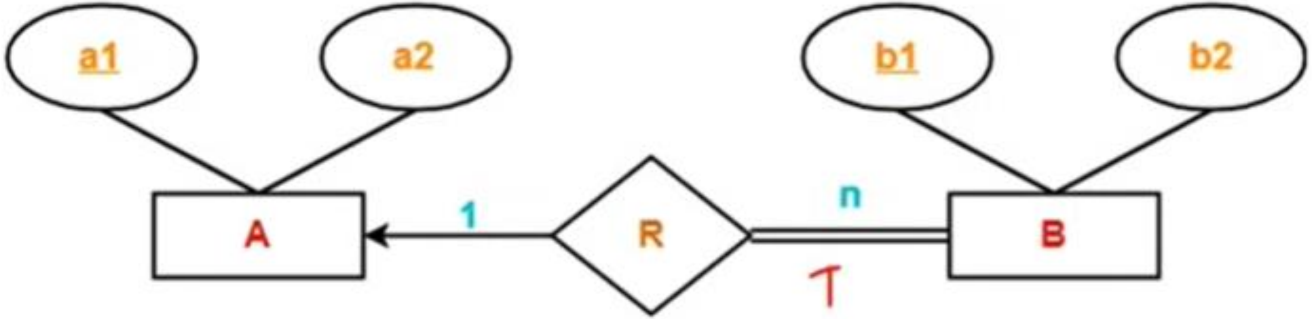
- **Two tables** will be required. You can combine the relationship set with any one of the entity sets.

Rule 6: Binary relationships with both cardinality constraints and participation constraints



- Cardinality constraints will be implemented as discussed in Rule-5.
- Because of the *total participation constraint*, foreign key acquires **NOT NULL** constraint i.e. now foreign key can not be null.
- Two Cases:
  - **Case-1:** For Binary Relationship with Cardinality Constraint and Total Participation Constraint from One Side
  - **Case-2:** For Binary Relationship with Cardinality Constraint and Total Participation Constraint from Both Sides

Case-1: For Binary Relationship with Cardinality Constraint and Total Participation Constraint from One Side

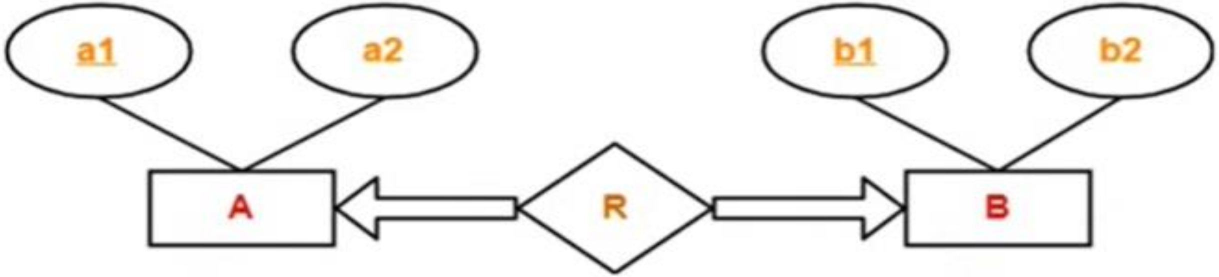


- □ Because cardinality ratio = 1 : n , we will combine the entity set B and relationship set R.
- □ Then, **two tables** will be required-
  1. A (<u>a1</u> , a2)
  2. BR (<u>b1</u> , b2 , a1)

Because of total participation, ***foreign key a1*** has acquired ***NOT NULL constraint***,

Case-2: For Binary Relationship with Cardinality Constraint and Total Participation Constraint from Both Sides

- If there is a key constraint from both the sides of an entity set with total participation, then that binary relationship is represented using **only single table.**

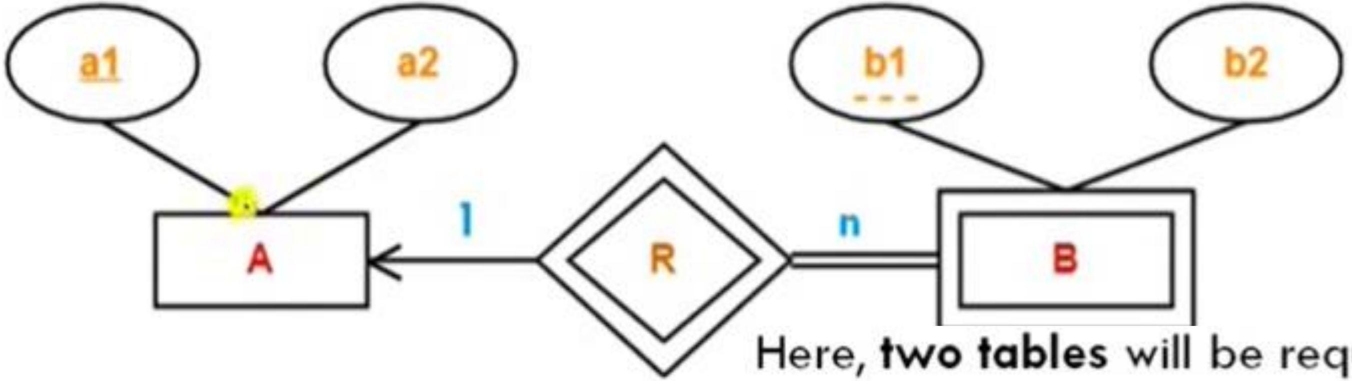


- Here, Only **one table** is required.

1. **ARB (<u>a1</u> , a2 , <u>b1</u> , b2)**

Rule-7: For Binary Relationship with Weak Entity Set

- Weak entity set always appears in association with identifying relationship with **total** participation constraint and there is always **1: n** relationship from identifying entity set to weak entity set.



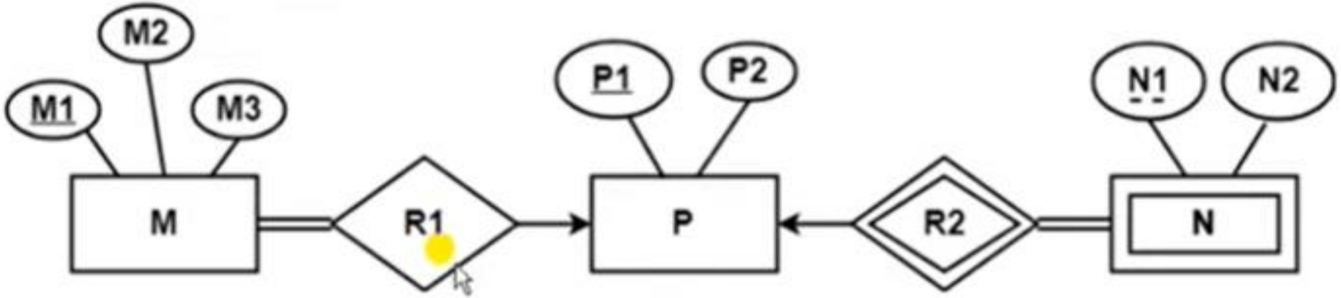
Here, **two tables** will be required-

1. **A** (a1 , a2)
2. **BR** (a1 , b1 , b2)

- Here, **two tables** will be required-

Question 1

Find the minimum number of tables required for the following ER diagram in relational model



3 Tables:

FK

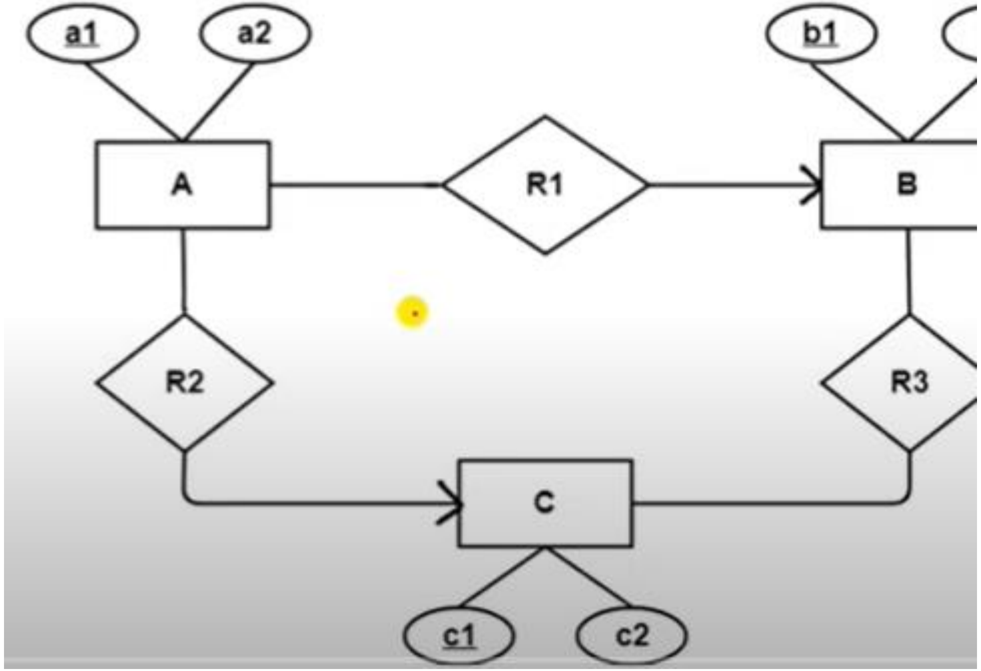
MR1 (M1 , M2 , M3 , P1)

P (P1 , P2)

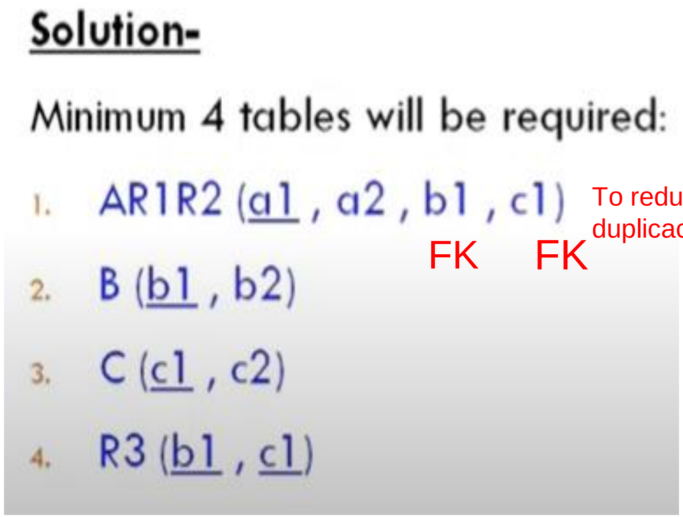
NR2 (P1 , N1 , N2)

Question 2

Find the Minimum No. of Tables



Question 2



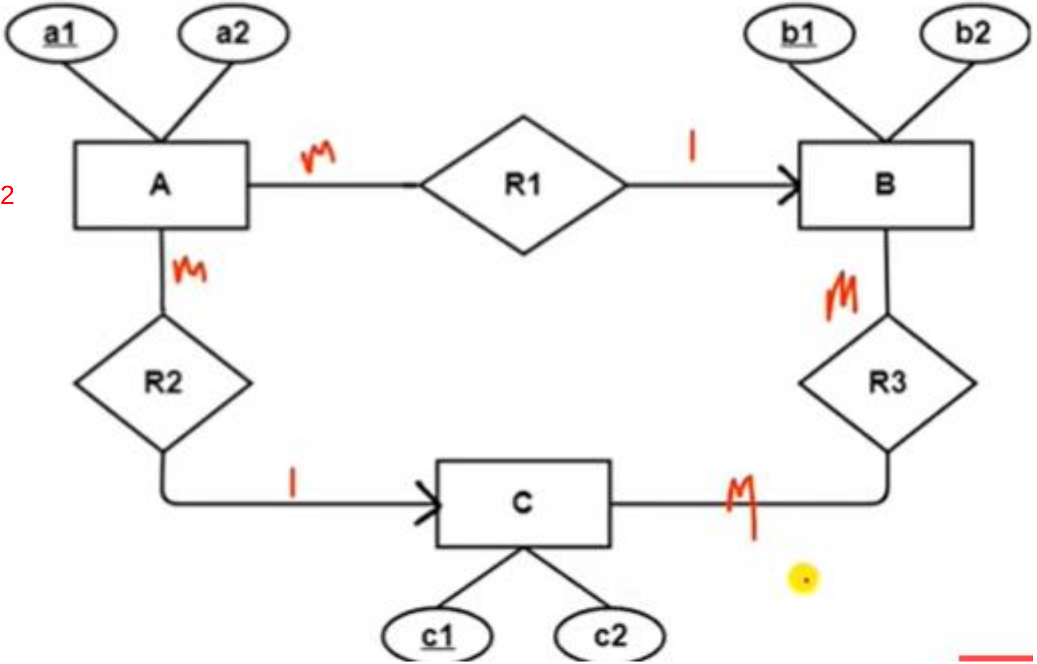
Solution-

Minimum 4 tables will be required:

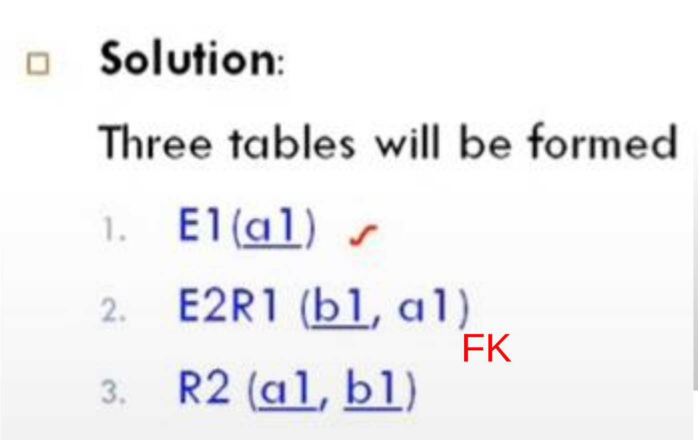
1. AR1R2 (a1 , a2 , b1 , c1)
2. B (b1 , b2)
3. C (c1 , c2)
4. R3 (b1 , c1)

FK FK

To reduce duplicacy a1,a2



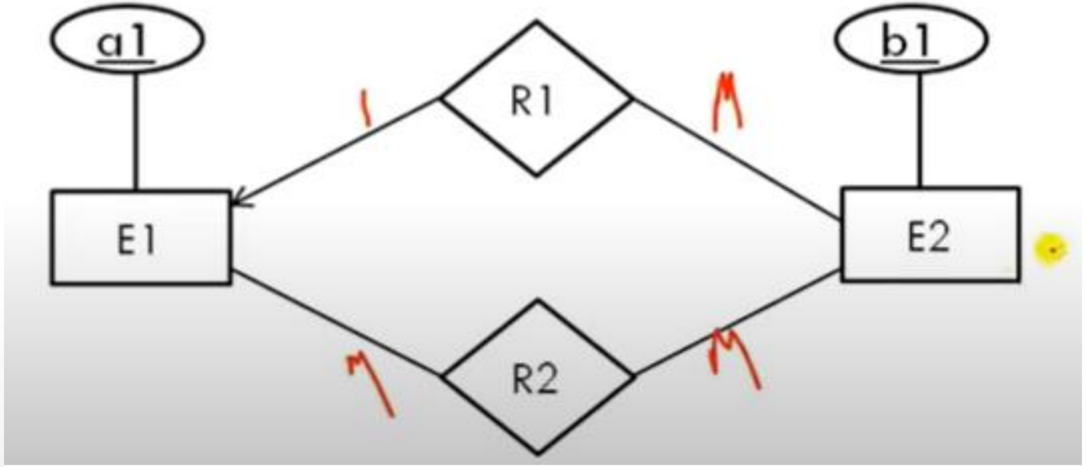
Question 3



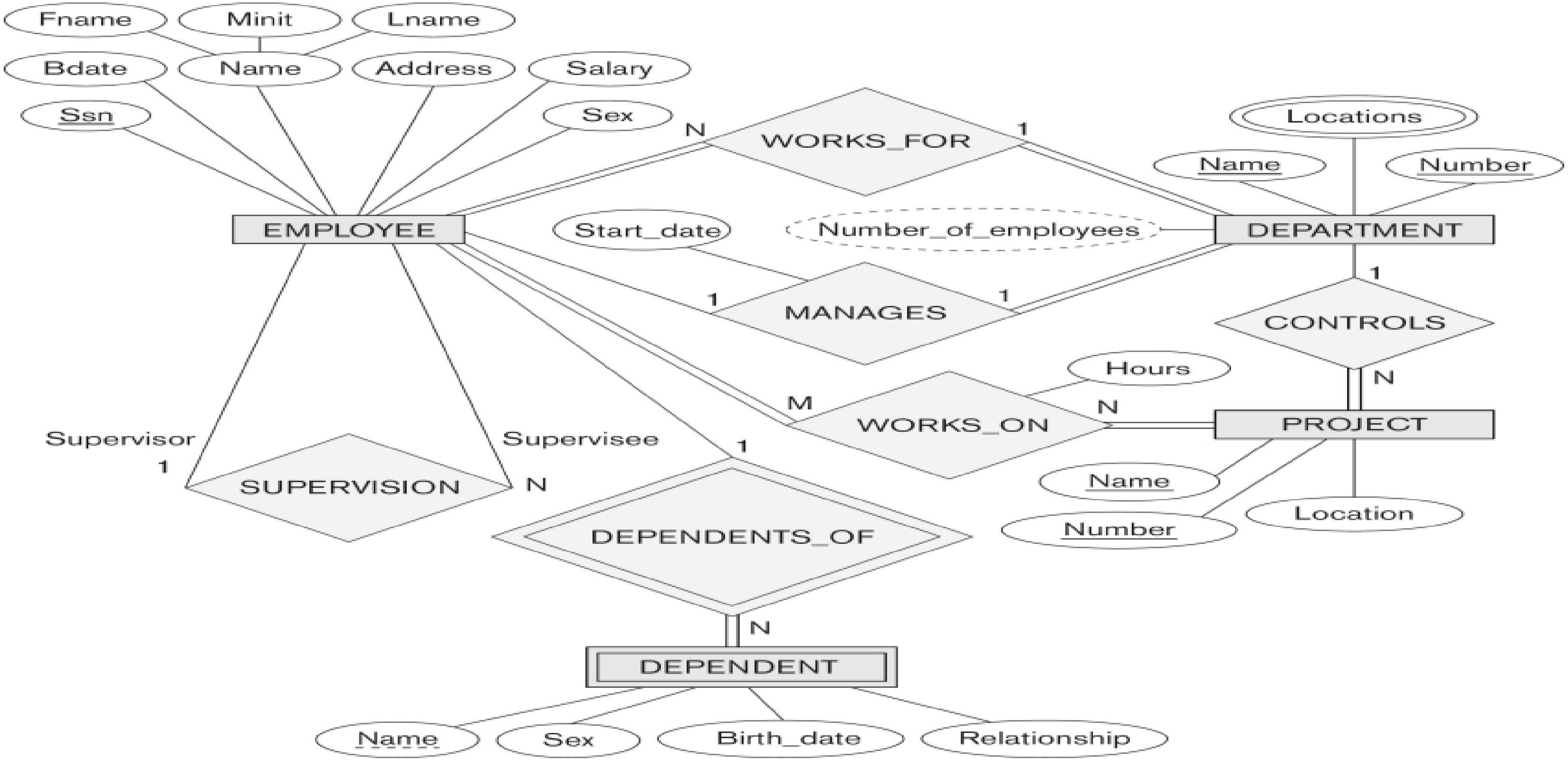
- **Solution:**

  Three tables will be formed

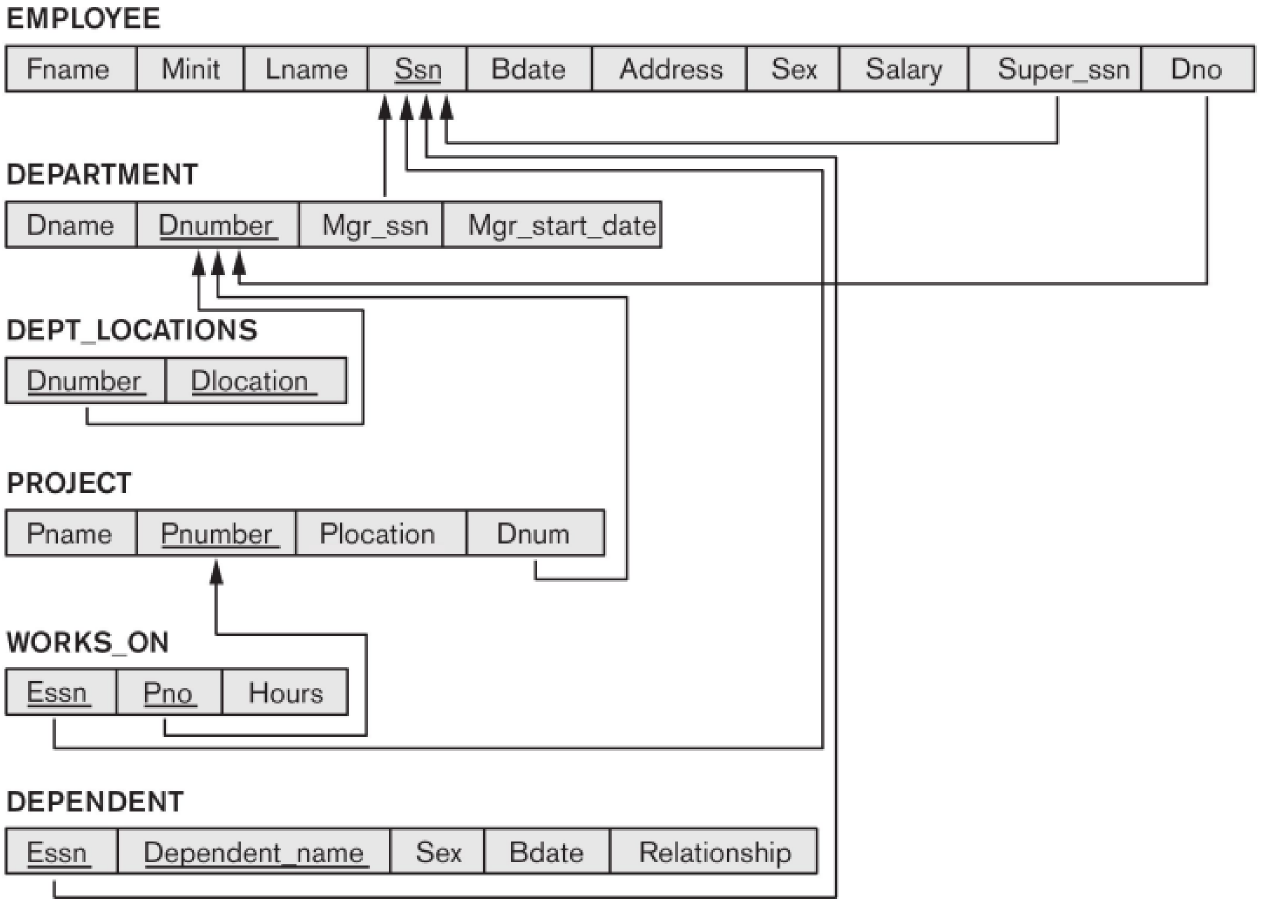
  1. E1(a1) ✓
  2. E2R1 (b1, a1) FK
  3. R2 (a1, b1)



Example


Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
N
WORKS_FOR
1
Locations
Name
Number
EMPLOYEE
Start_date
Number_of_employees
DEPARTMENT
1
1
MANAGES
1
CONTROLS
Hours
N
M
WORKS_ON
N
PROJECT
Supervisor
Supervisee
1
1
SUPERVISION
N
Name
Location
Number
DEPENDENTS_OF
N
DEPENDENT
Name
Sex
Birth_date
Relationship

Resulting Relational Schema



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

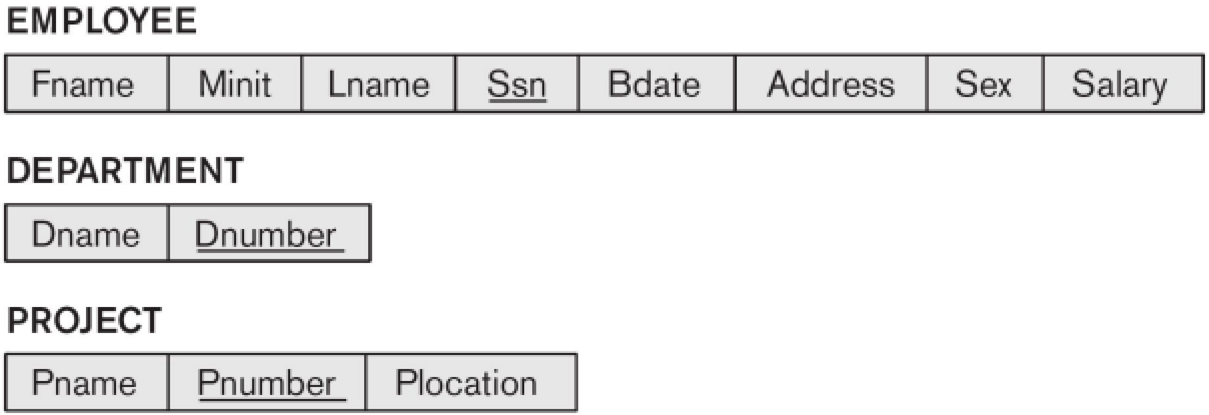
DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

- **Step 1: Regular Entity Types.**
  - For each regular (strong) entity type E in the ER diagram, create a relation R that includes all the simple attributes of E.
  - Choose one of the key attributes of E as the primary key for R.
  - If the chosen key of E is composite, the set of simple attributes that form it will together form the primary key of R.
- **Example:** We create the relations EMPLOYEE, DEPARTMENT, and PROJECT in the relational schema corresponding to the regular entities in the ER diagram.

SSN, DNUMBER, and PNUMBER are the primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT as shown.

Step 1 Result



EMPLOYEE

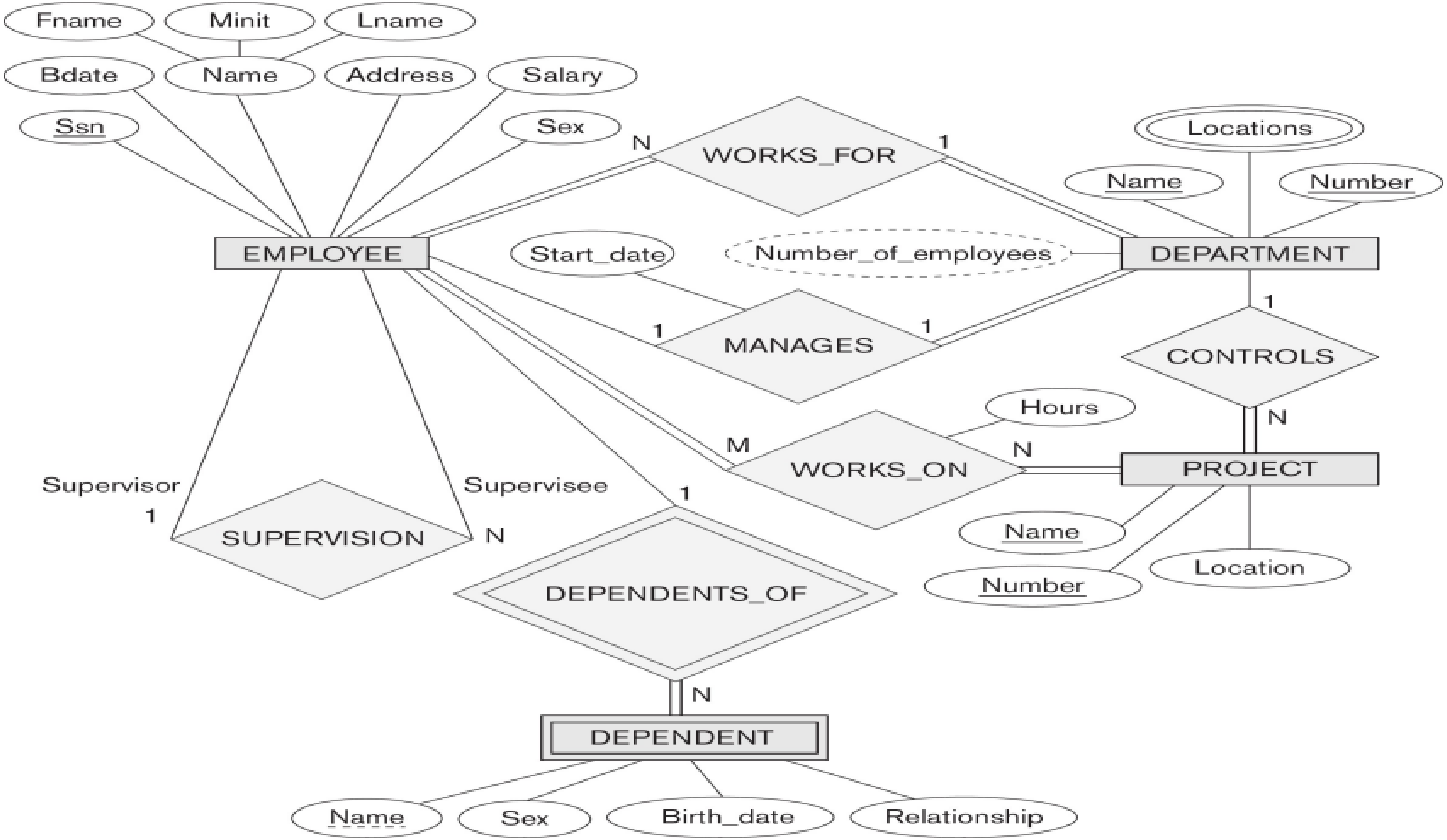
Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary

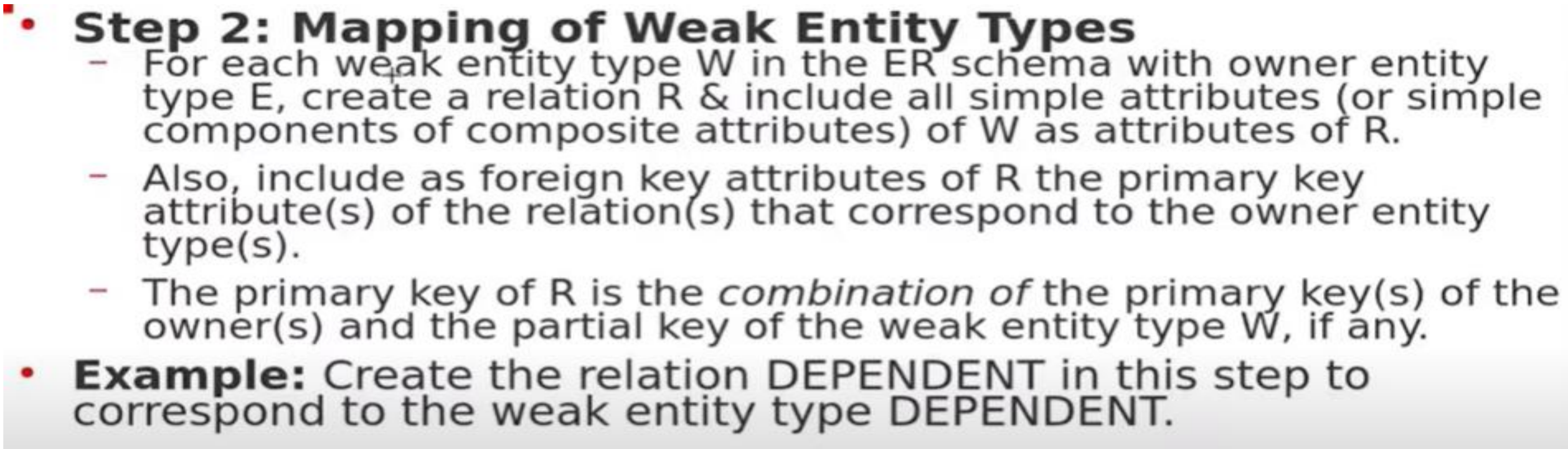
DEPARTMENT

Dname	Dnumber

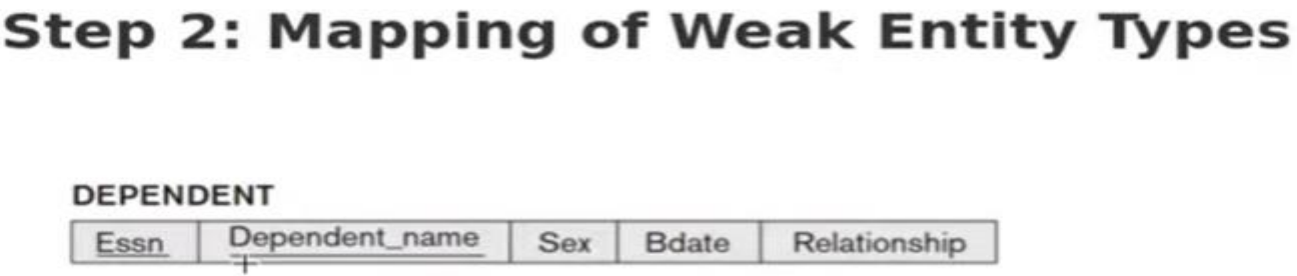
PROJECT

Pname	Pnumber	Plocation


Fname
Minit
Lname
Bdate
Name
Address
Salary
Ssn
Sex
EMPLOYEE
N
WORKS_FOR
1
Locations
Name
Number
DEPARTMENT
Start_date
Number_of_employees
1
MANAGES
1
1
CONTROLS
N
Hours
M
WORKS_ON
N
PROJECT
Supervisor
1
SUPERVISION
Supervisee
N
1
DEPENDENTS_OF
Name
Number
Location
N
DEPENDENT
Name
Sex
Birth_date
Relationship



- **Step 2: Mapping of Weak Entity Types**
  - For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes (or simple components of composite attributes) of W as attributes of R.
  - Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s).
  - The primary key of R is the *combination of* the primary key(s) of the owner(s) and the partial key of the weak entity type W, if any.
- **Example:** Create the relation DEPENDENT in this step to correspond to the weak entity type DEPENDENT.



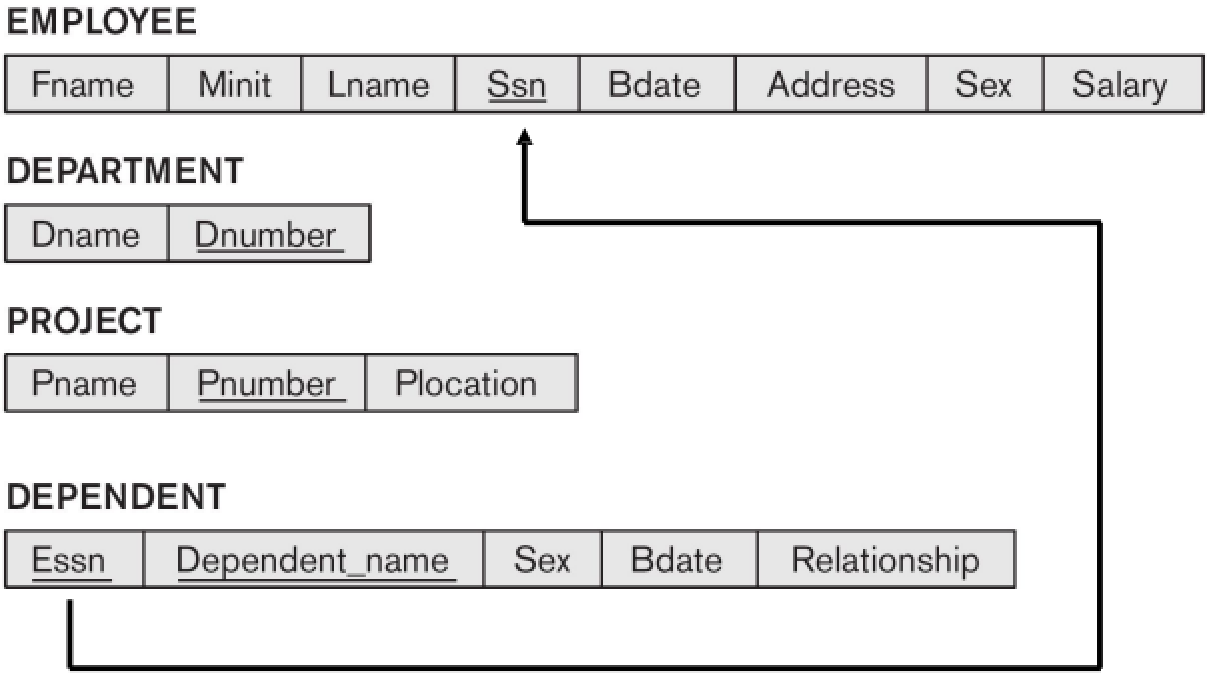
Step 2: Mapping of Weak Entity Types

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

FK + Partial K = Primary key

Step 2 Result



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary

DEPARTMENT

Dname	Dnumber

PROJECT

Pname	Pnumber	Plocation

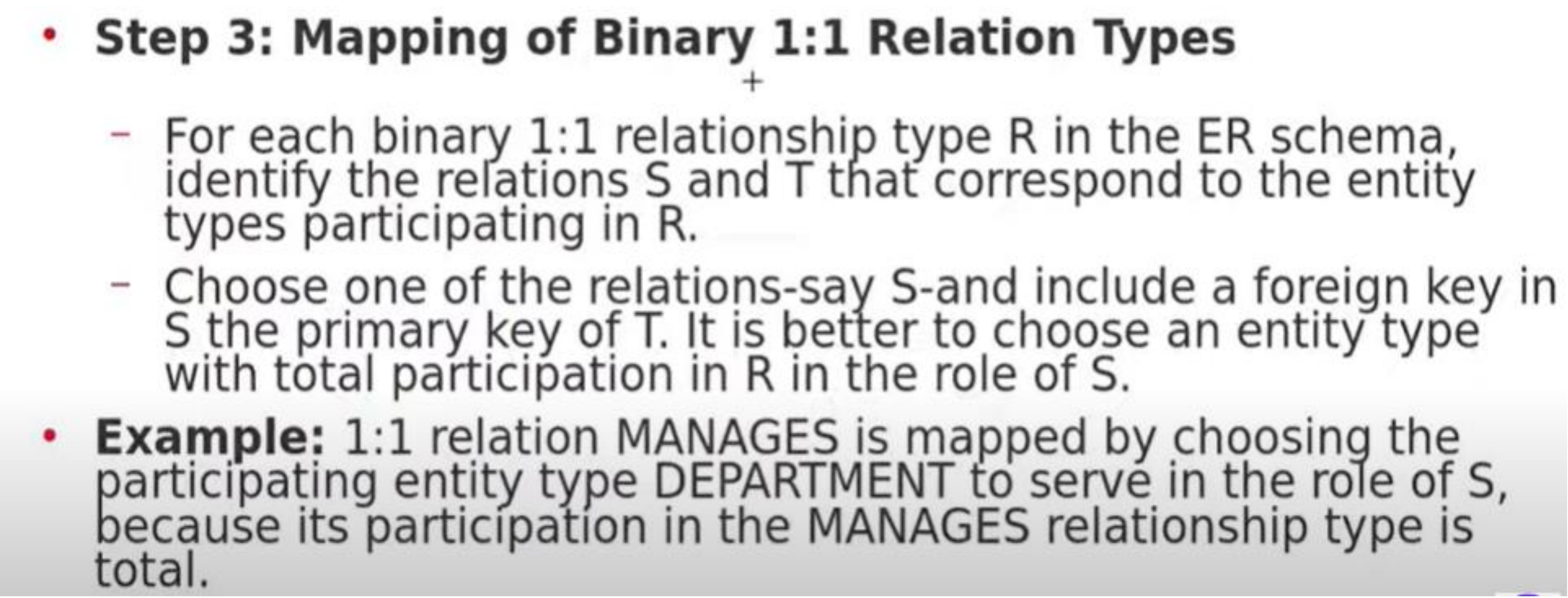
DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Step 3: Mapping Binary 1-to-1

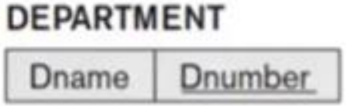
Three approaches

- **Foreign Key**
  - Usually appropriate
- Merged Relation
  - Possible when both participations are total
- Relationship Relation
  - Not discussed



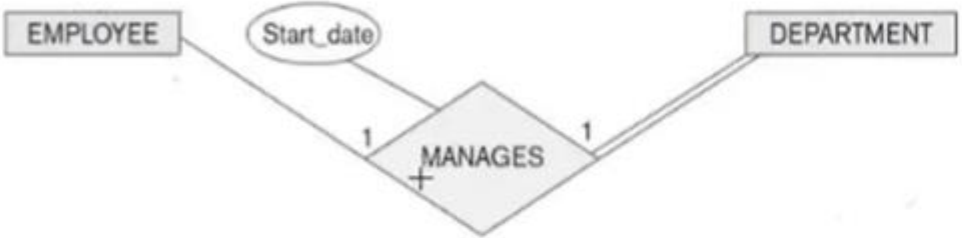
- **Step 3: Mapping of Binary 1:1 Relation Types**
  - For each binary 1:1 relationship type R in the ER schema, identify the relations S and T that correspond to the entity types participating in R.
  - Choose one of the relations-say S-and include a foreign key in S the primary key of T. It is better to choose an entity type with total participation in R in the role of S.
- **Example:** 1:1 relation MANAGES is mapped by choosing the participating entity type DEPARTMENT to serve in the role of S, because its participation in the MANAGES relationship type is total.

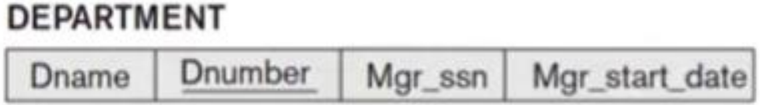
Step 3: Mapping of Binary 1:1 Relation Types



DEPARTMENT

Dname	Dnumber

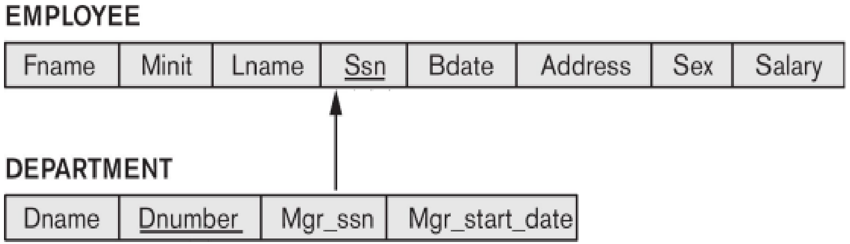




DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

Step 3 Result



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary

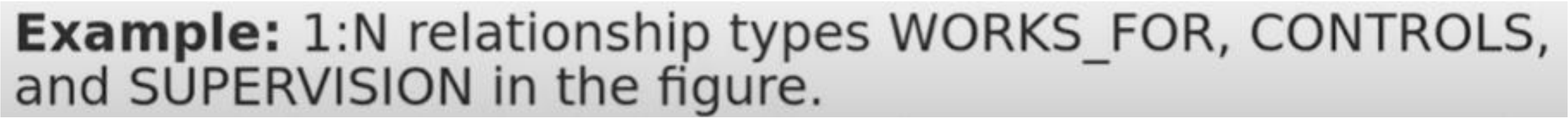
DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

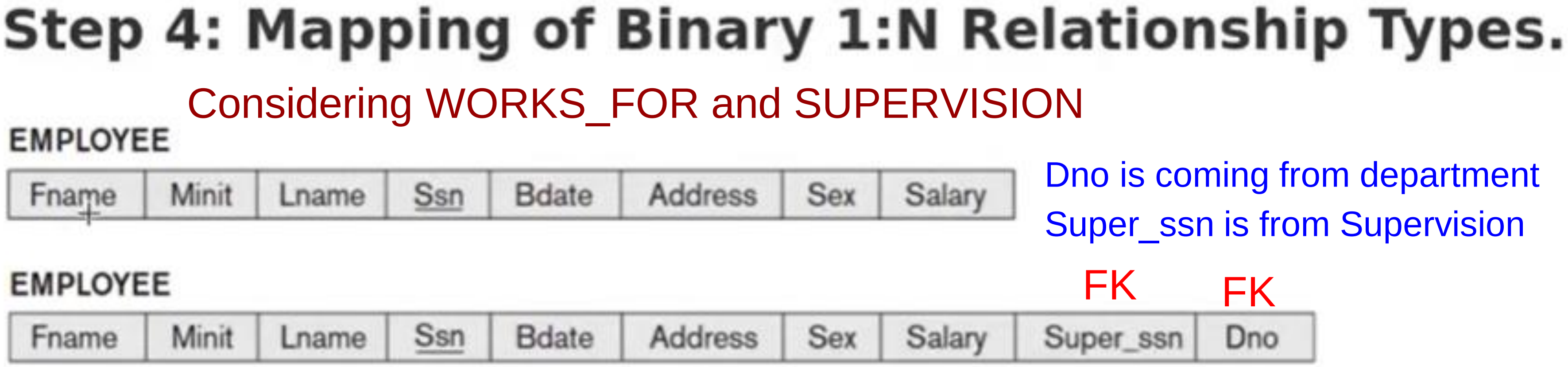
Step 4: Binary 1-to-N

i. Choose the *S* relation as the type at the N-side of the relationship, other is *T*

ii. Add as a foreign key to *S* all of the primary key attribute(s) of *T*



Example: 1:N relationship types WORKS_FOR, CONTROLS, and SUPERVISION in the figure.



Step 4: Mapping of Binary 1:N Relationship Types.

Considering WORKS_FOR and SUPERVISION

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary

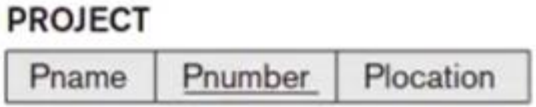
Dno is coming from department

Super_ssn is from Supervision

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn FK	Dno FK

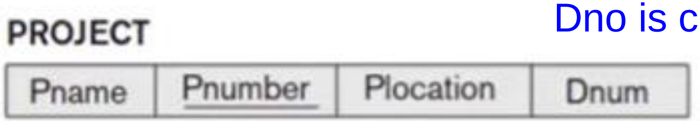
Considering CONTROLS relationship



PROJECT

Pname	Pnumber	Plocation

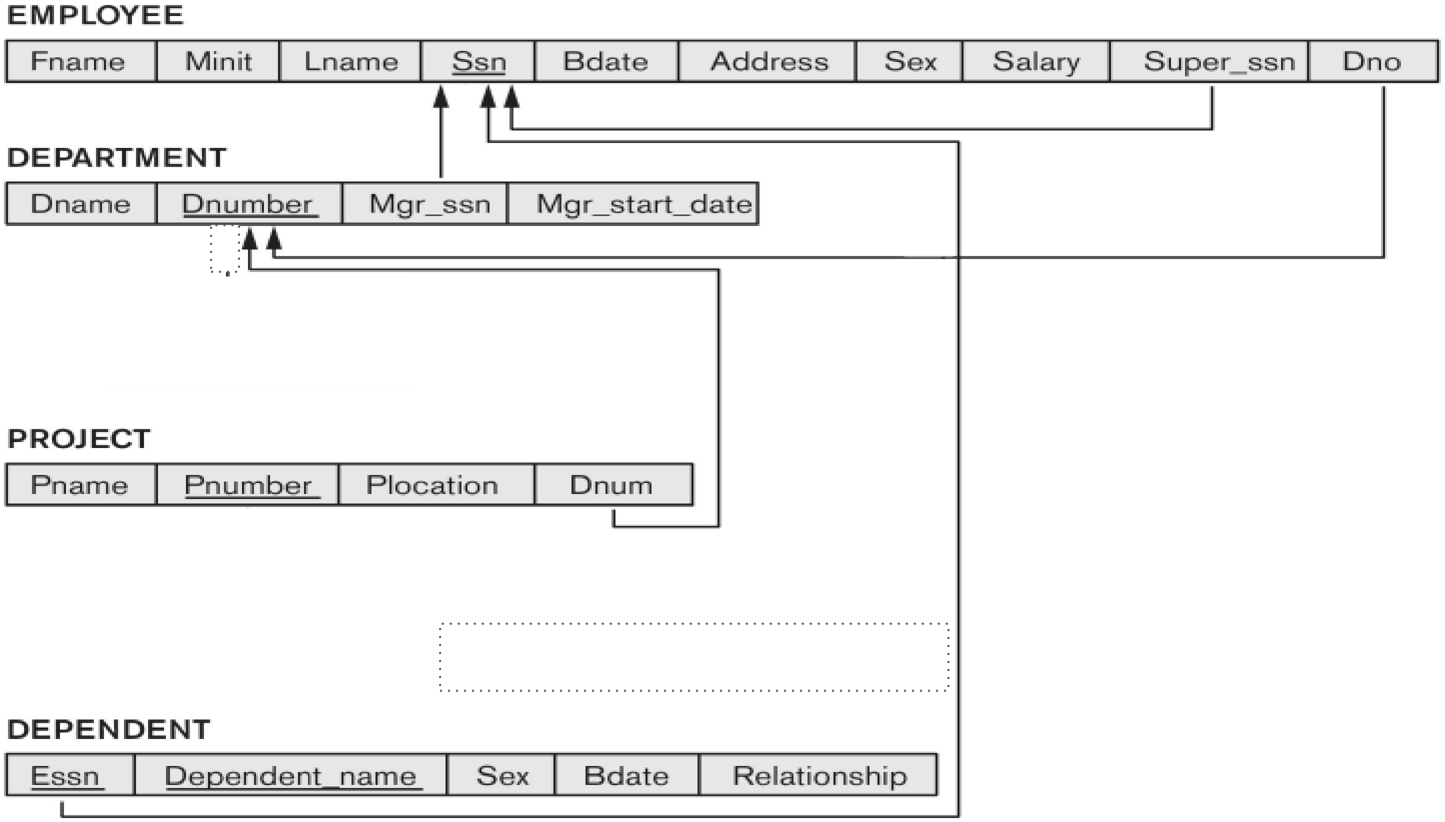
Dno is coming from department



PROJECT

Pname	Pnumber	Plocation	Dnum

Step 4 Result



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

PROJECT

Pname	Pnumber	Plocation	Dnum

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship

Step 5: Binary M-to-N

i. Create a new relation *S* (termed: *relationship relation*)

- In some ERD dialects, actually drawn in

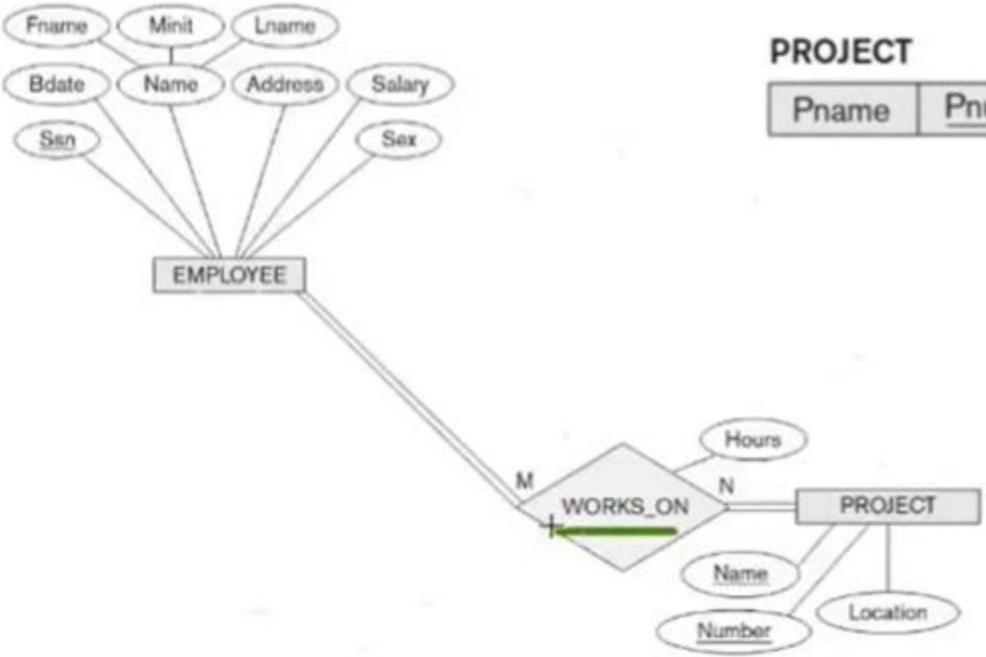
ii. Add as foreign keys the primary keys of both relations; their combination forms the primary key of *S*

iii. Add any simple attributes of the M:N relationship to *S*

Step 5: Mapping of Binary M:N Relationship Types

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno



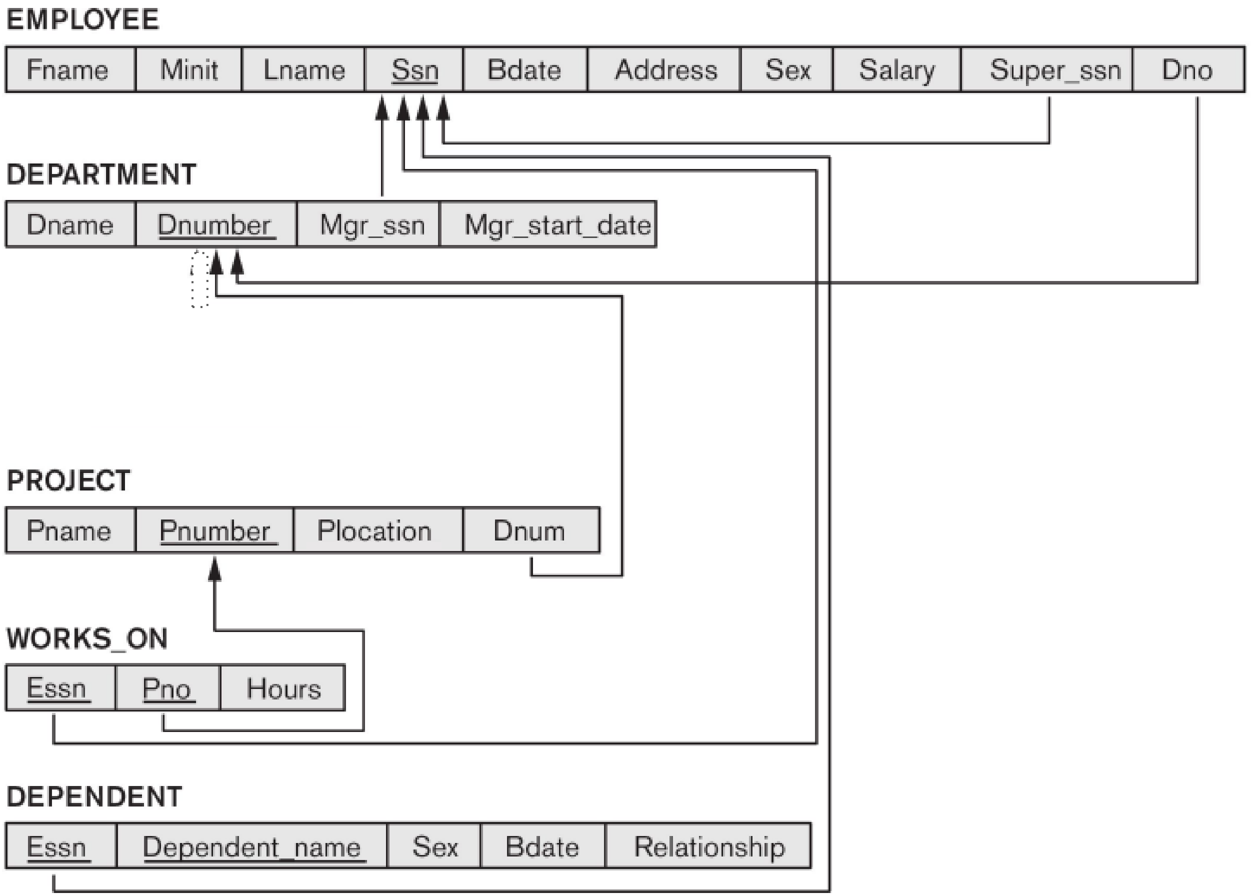
PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours
FK	FK	

Step 5 Result



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

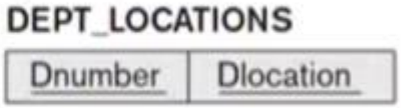
Essn	Dependent_name	Sex	Bdate	Relationship

Step 6: Multivalued Attributes

i. Create a new relation *S*

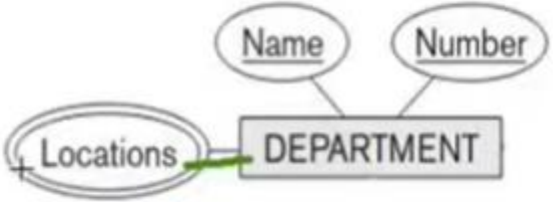
ii. Add as foreign keys the primary keys of the corresponding relation

iii. Add the attribute to *S* (if composite, the simple attributes); the combination of all attributes in *S* forms the primary key

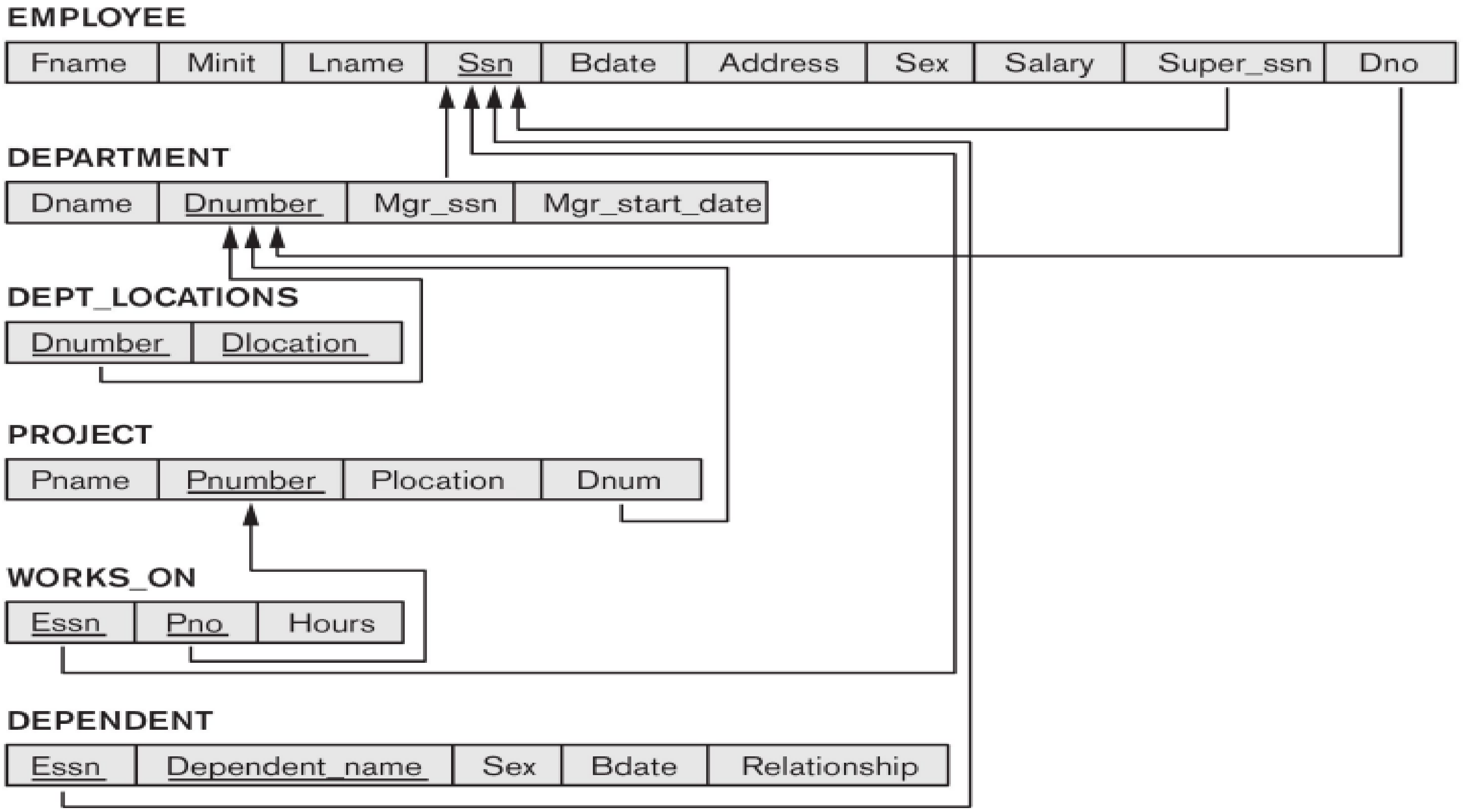


DEPT_LOCATIONS

Dnumber	Dlocation



Step 6 Result



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

DEPT_LOCATIONS

Dnumber	Dlocation

PROJECT

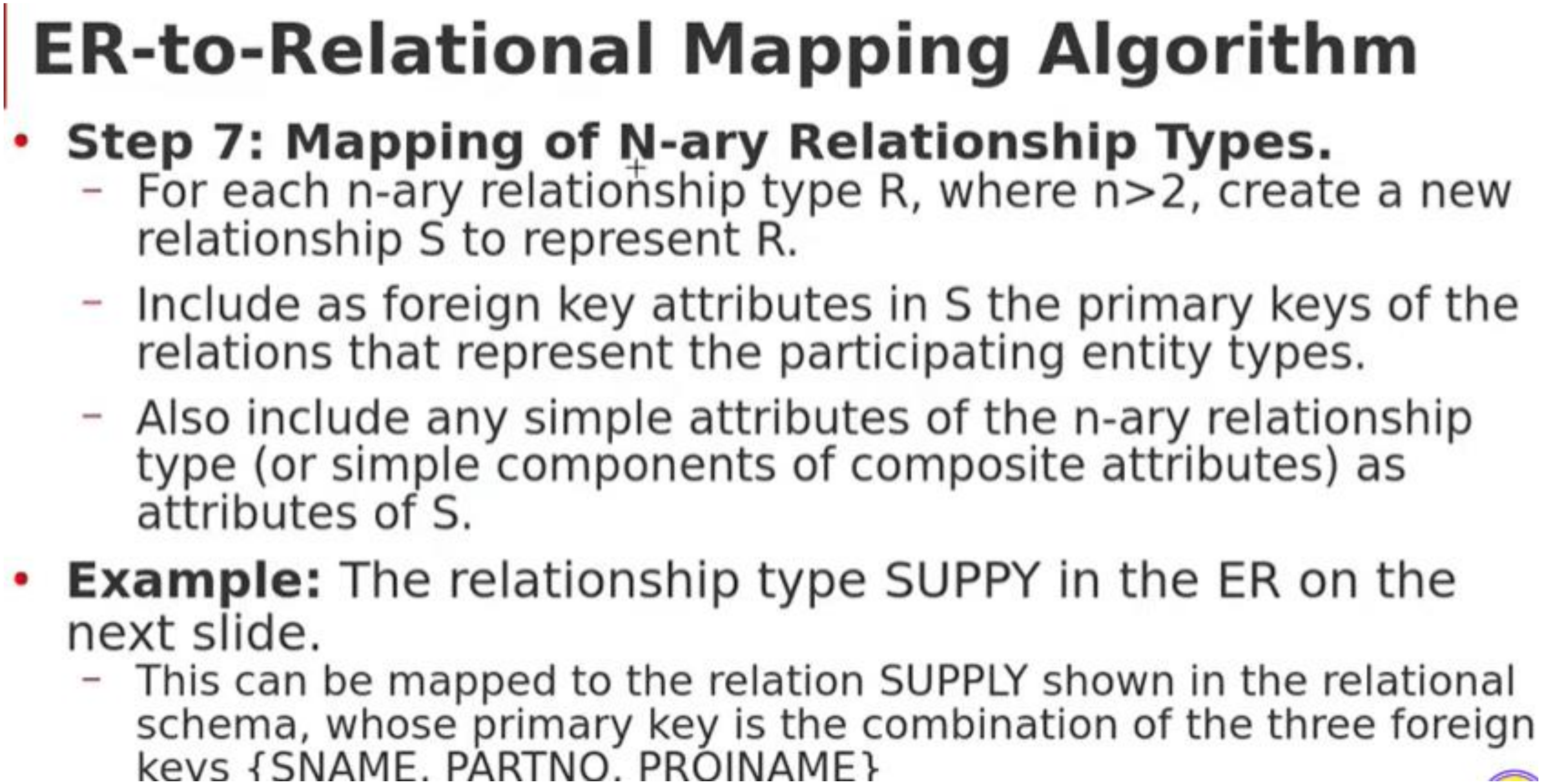
Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

DEPENDENT

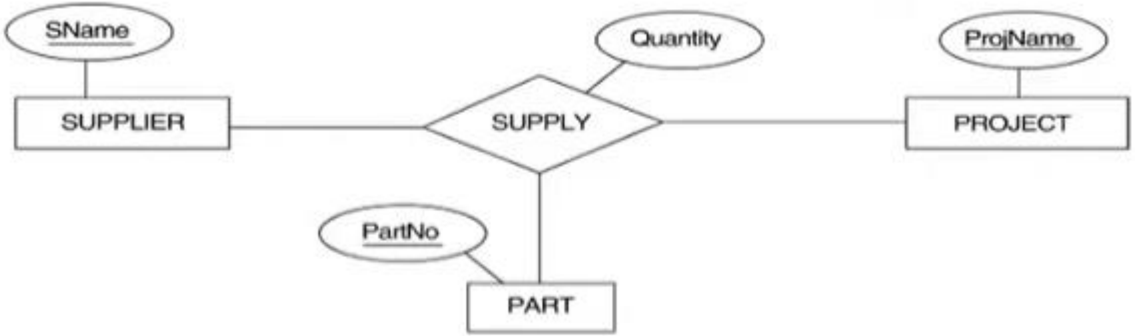
Essn	Dependent_name	Sex	Bdate	Relationship



ER-to-Relational Mapping Algorithm

- **Step 7: Mapping of N-ary Relationship Types.**
  - For each n-ary relationship type R, where n>2, create a new relationship S to represent R.
  - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types.
  - Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S.
- **Example:** The relationship type SUPPY in the ER on the next slide.
  - This can be mapped to the relation SUPPLY shown in the relational schema, whose primary key is the combination of the three foreign keys {SNAME, PARTNO, PROJNAME}

(a)



SUPPLIER

SNAME	. . .

PROJECT

PROJNAME	. . .

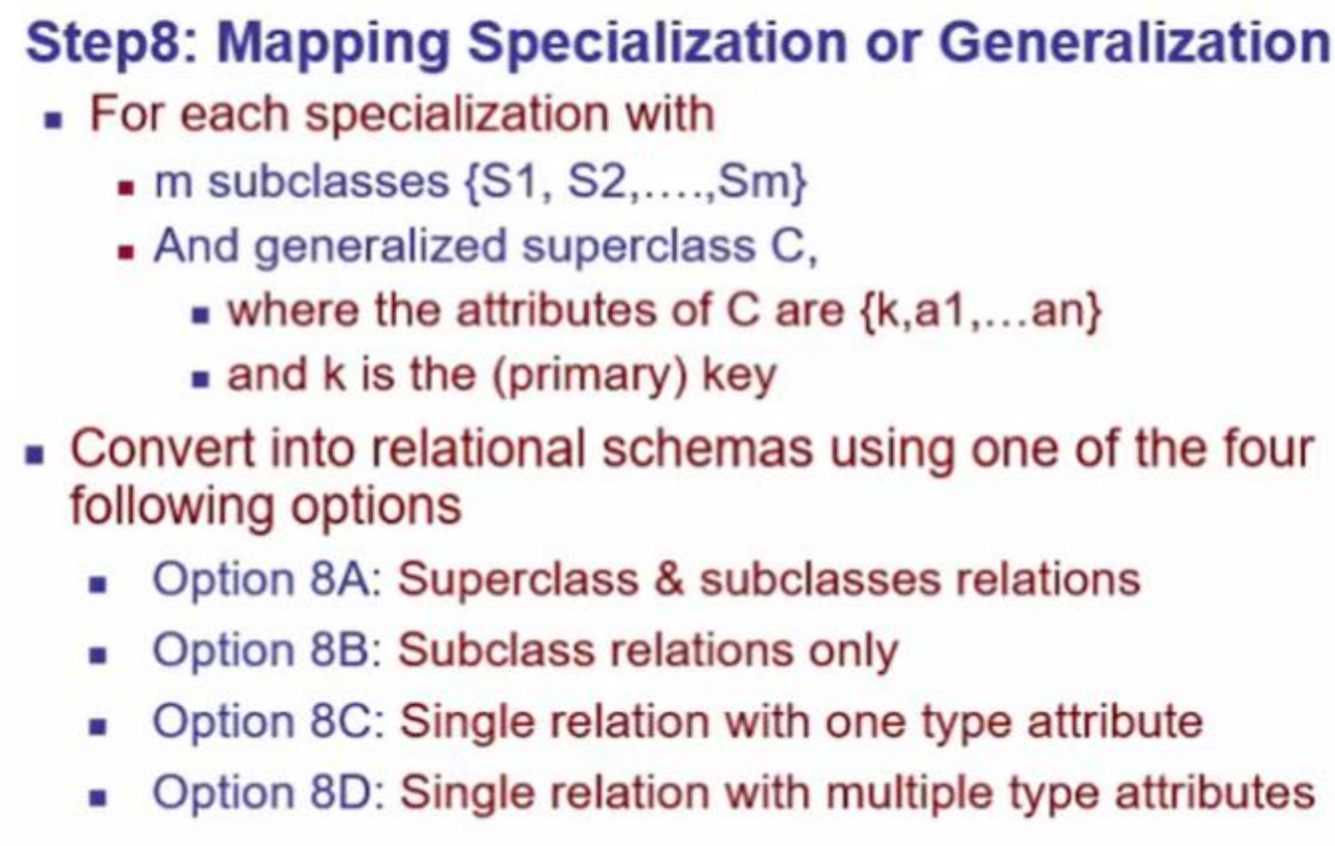
PART

PARTNO	. . .

SUPPLY

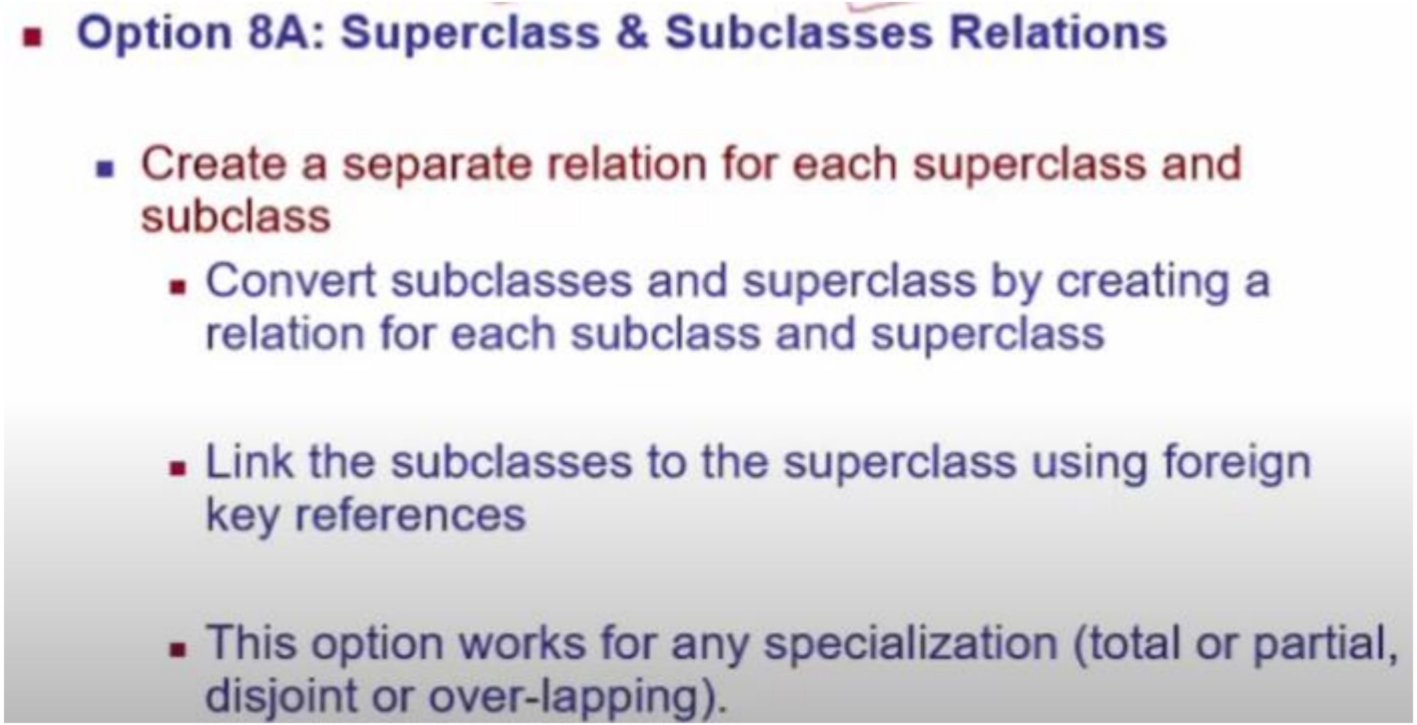
SNAME	PROJNAME	PARTNO	QUANTITY

Mapping of EER to Relation schema



Step8: Mapping Specialization or Generalization

- For each specialization with
  - m subclasses {S1, S2,….,Sm}
  - And generalized superclass C,
    - where the attributes of C are {k,a1,…an}
    - and k is the (primary) key
- Convert into relational schemas using one of the four following options
  - Option 8A: Superclass & subclasses relations
  - Option 8B: Subclass relations only
  - Option 8C: Single relation with one type attribute
  - Option 8D: Single relation with multiple type attributes



- **Option 8A: Superclass & Subclasses Relations**
  - Create a separate relation for each superclass and subclass
    - Convert subclasses and superclass by creating a relation for each subclass and superclass
    - Link the subclasses to the superclass using foreign key references
    - This option works for any specialization (total or partial, disjoint or over-lapping).

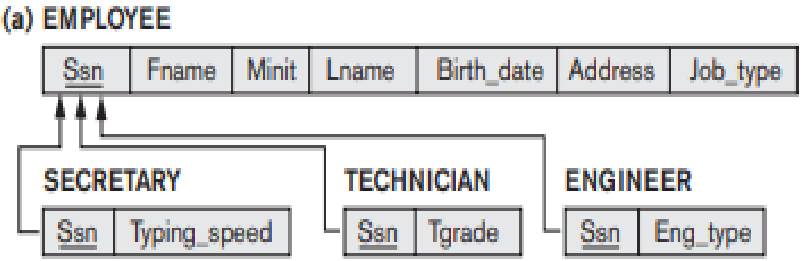
- **Option 8B: Subclass relations only**
  - This option only works for a specialization whose subclasses are total (Mandatory participation)
  - Every entity in the superclass must belong to (at least) one of the subclasses.
- **Option 8C: Single relation with one type attribute**
  - Single attribute is used to indicate the type of subclass
  - The attribute is called a type (or **discriminating**) attribute that indicates the subclass to which each tuple belongs
  - Works only if the subclasses are disjoint
- **Option 8D: Single relation with multiple type attributes**
  - Have a Boolean valued attribute for each subclass.
    - True if in a class otherwise False
  - Works if subclasses may be overlapping

Mapping of EER to Relation schema

Step 8: Options for Mapping Specialization or Generalization

Option 8A: Multiple relations-Superclass

and subclasses.



(a) EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type

SECRETARY

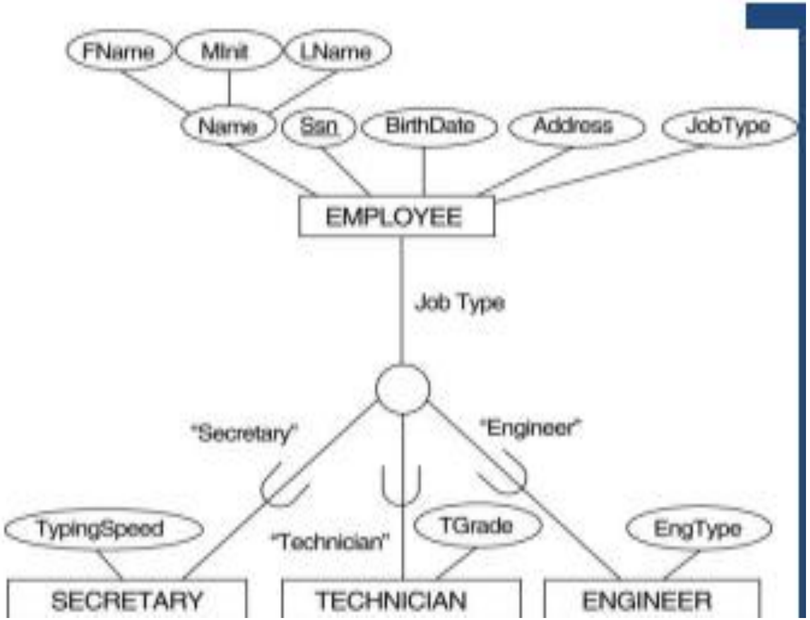
Ssn	Typing_speed

TECHNICIAN

Ssn	Tgrade

ENGINEER

Ssn	Eng_type

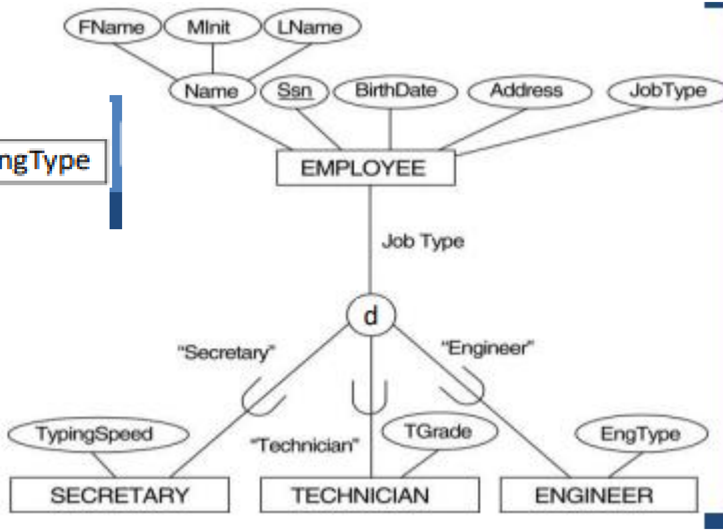


Option 8C: Single relation with one type attribute.

(c) EMPLOYEE

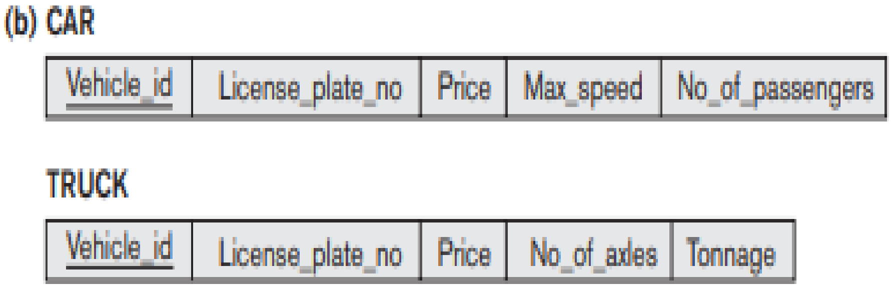
SSN	FName	MInit	LName	BirthDate	Address	JobType	TypingSpeed	TGrade	EngType

Based on Job type Attribute



Generalizing CAR and TRUCK into the superclass VEHICLE.

Option 8B: Multiple relations- Subclass relations only

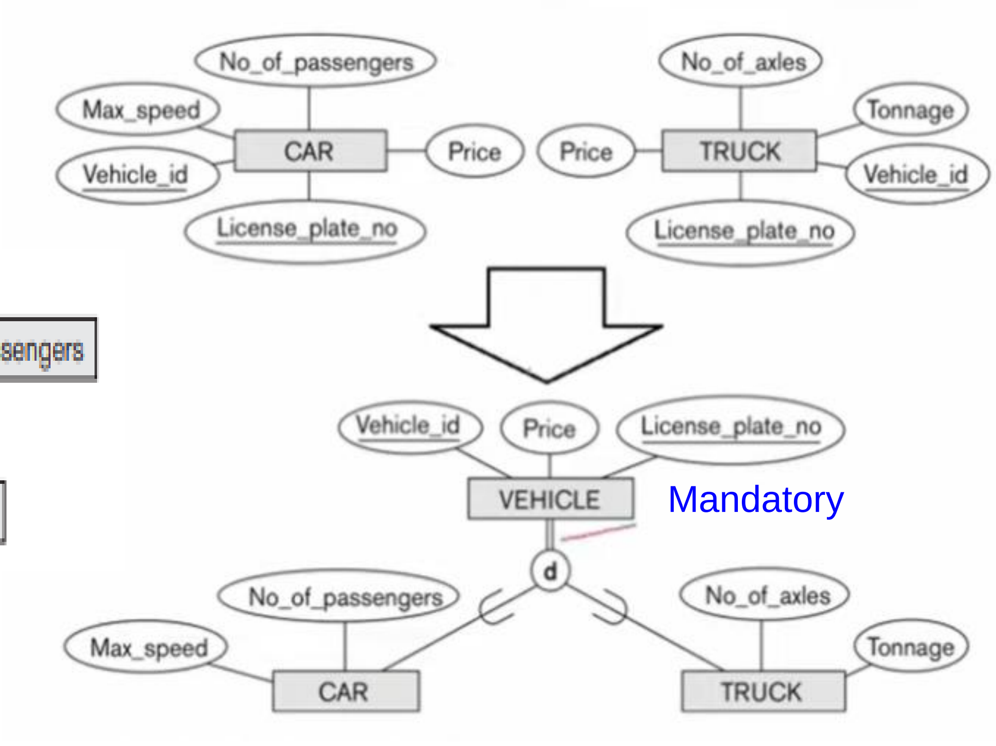


(b) CAR

Vehicle_id	License_plate_no	Price	Max_speed	No_of_passengers

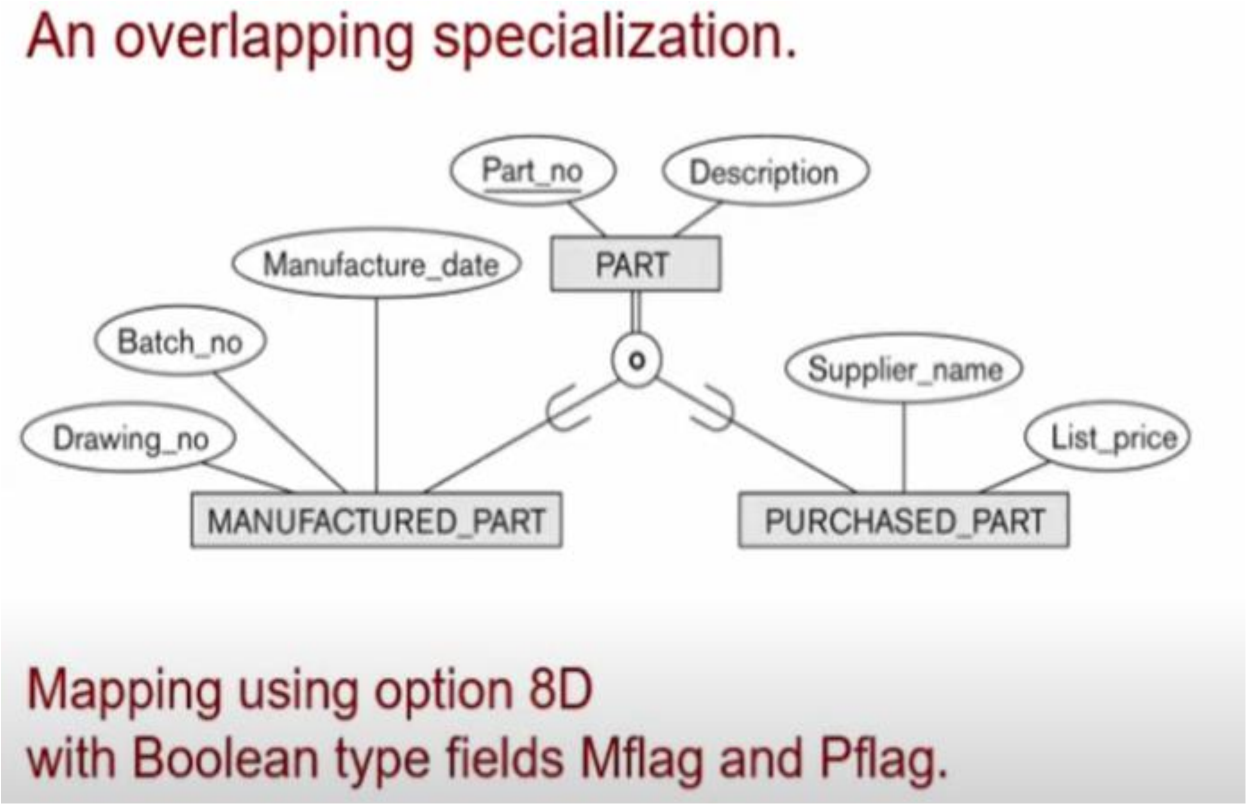
TRUCK

Vehicle_id	License_plate_no	Price	No_of_axles	Tonnage



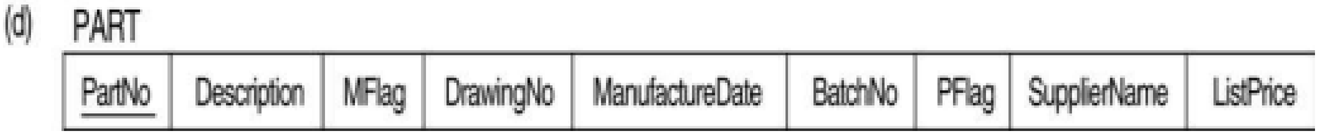
Mandatory

Option 8D: Single relation with multiple type attributes.



An overlapping specialization.

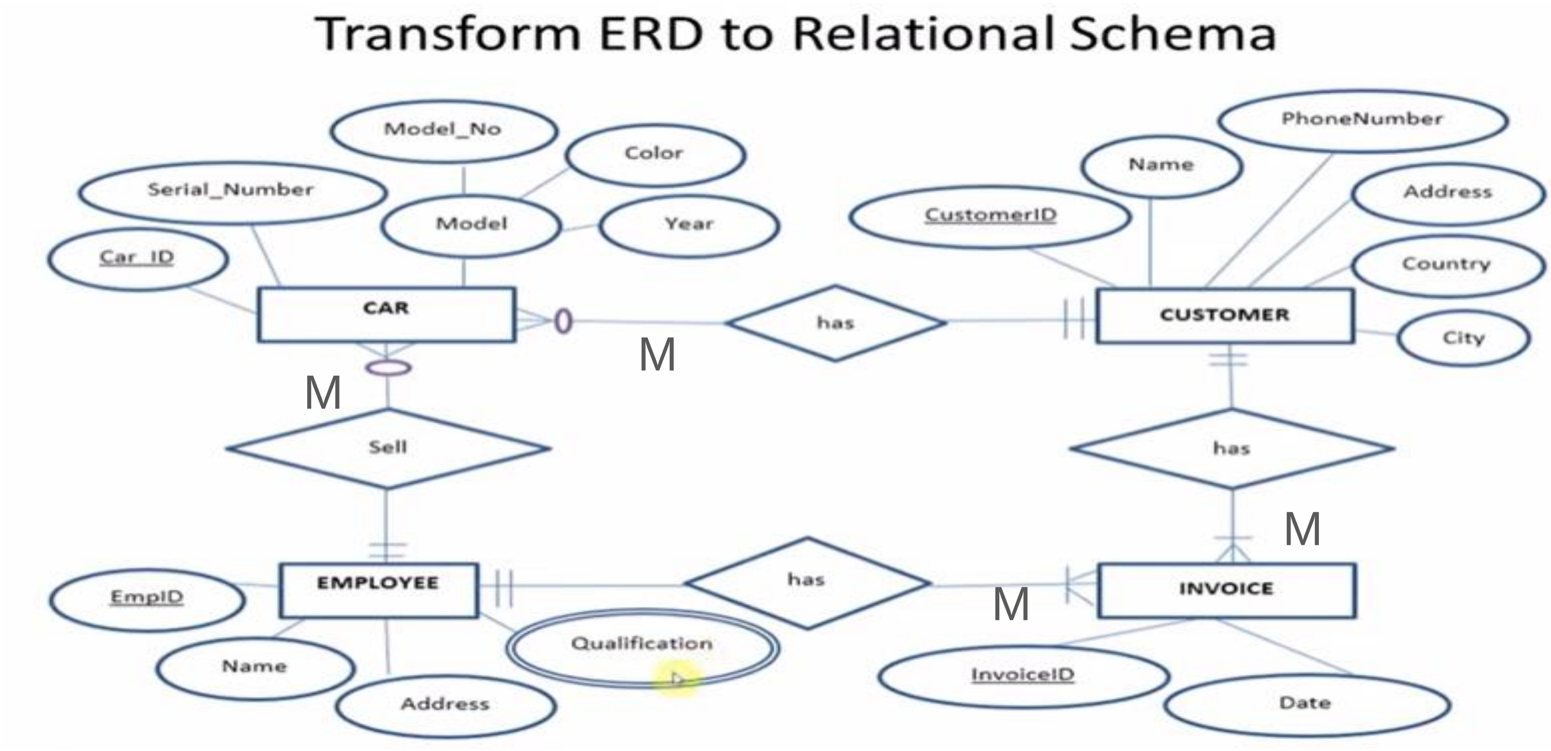
Mapping using option 8D
with Boolean type fields Mflag and Pflag.



(d) PART

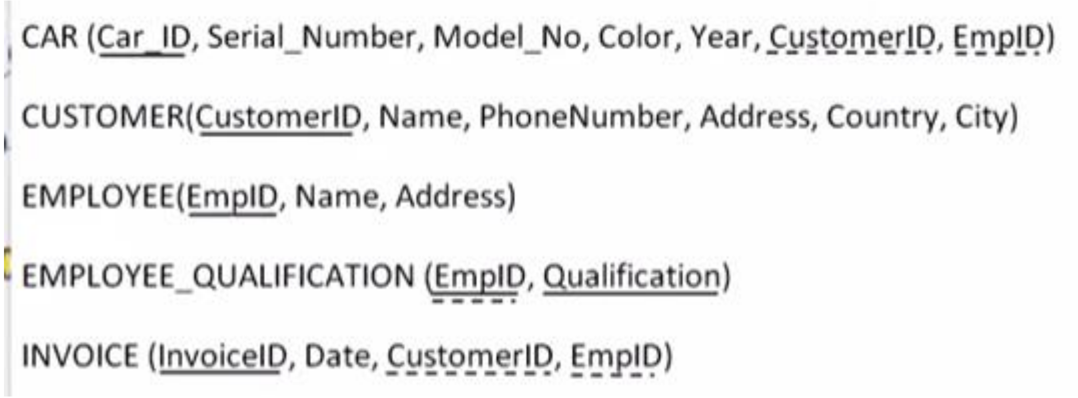
PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

Exercise:



Transform ERD to Relational Schema

ER diagram to Relational schema



CAR (Car_ID, Serial_Number, Model_No, Color, Year, CustomerID, EmpID)

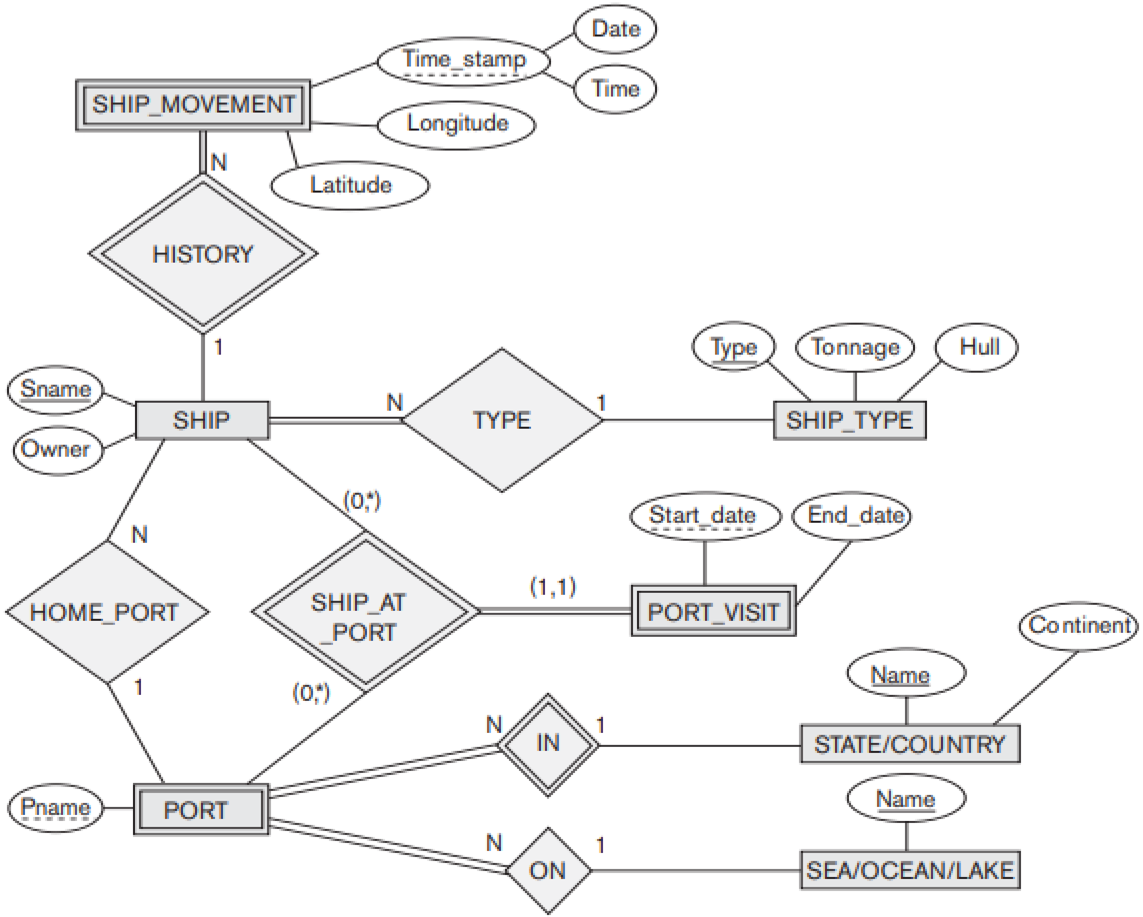
CUSTOMER(CustomerID, Name, PhoneNumber, Address, Country, City)

EMPLOYEE(EmpID, Name, Address)

EMPLOYEE_QUALIFICATION (EmpID, Qualification)

INVOICE (InvoiceID, Date, CustomerID, EmpID)

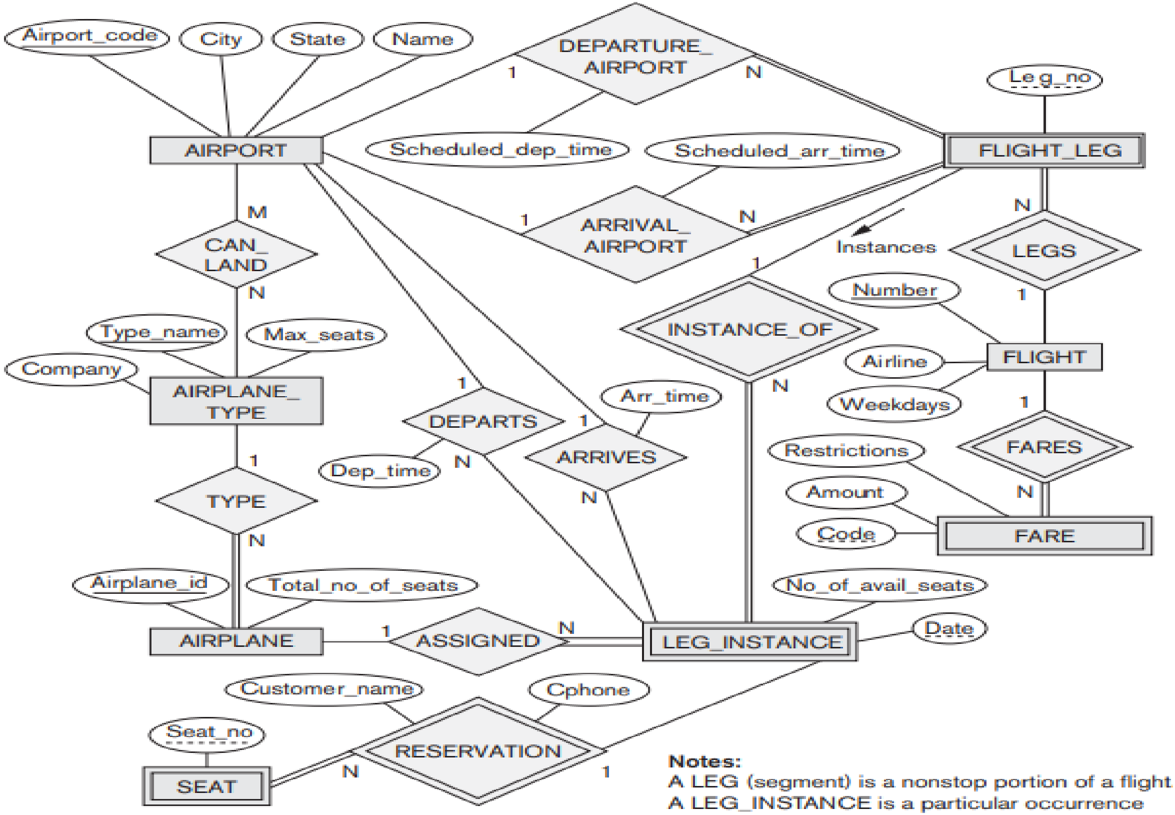
Exercise:



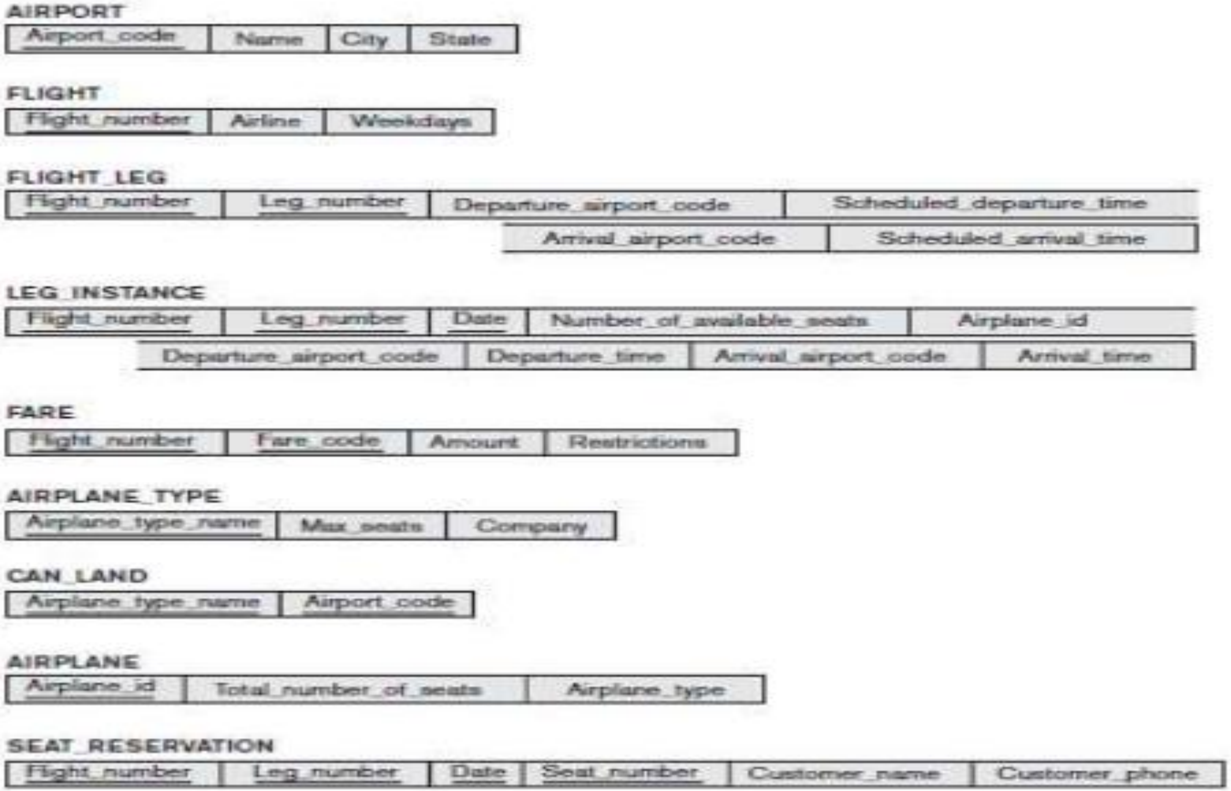
SHIP_MOVEMENT
Time_stamp
Date
Time
Longitude
Latitude
N
HISTORY
1
Sname
Owner
SHIP
N
TYPE
1
Type
Tonnage
Hull
SHIP_TYPE
(0,*)
Start_date
End_date
N
HOME_PORT
SHIP_AT _PORT
(1,1)
PORT_VISIT
Continent
1
(0,*)
Name
N
IN
1
STATE/COUNTRY
Name
Pname
PORT
N
ON
1
SEA/OCEAN/LAKE

Solution:

Exercise:



Notes:
A LEG (segment) is a nonstop portion of a flight
A LEG_INSTANCE is a particular occurrence
of a LEG on a particular date.



AIRPORT

Airport_code	Name	City	State

FLIGHT

Flight_number	Airline	Weekdays

FLIGHT_LEG

Flight_number	Leg_number	Departure_airport_code	Scheduled_departure_time	Arrival_airport_code	Scheduled_arrival_time

LEG_INSTANCE

Flight_number	Leg_number	Date	Number_of_available_seats	Airplane_id	Departure_airport_code	Departure_time	Arrival_airport_code	Arrival_time

FARE

Flight_number	Fare_code	Amount	Restrictions

AIRPLANE_TYPE

Airplane_type_name	Max_seats	Company

CAN_LAND

Airplane_type_name	Airport_code

AIRPLANE

Airplane_id	Total_number_of_seats	Airplane_type

SEAT_RESERVATION

Flight_number	Leg_number	Date	Seat_number	Customer_name	Customer_phone