05 (E)ER TO RELATIONAL (AND DDL)

CS 340: INTRODUCTION TO DATABASE SYSTEMS

Adapted from: Dr. Omar Alomeir
2023

Course instructor:
Dr. Maram Alajlan.

LEARNING GOALS

CLO1: Define the "database", its architecture, the main concepts and characteristics of databases, as well as concepts related to the design, and implementation of a relational databases. (Knowledge & Understanding)

CLO3: Create a logical database design based on the conceptual ER model. (Skills)

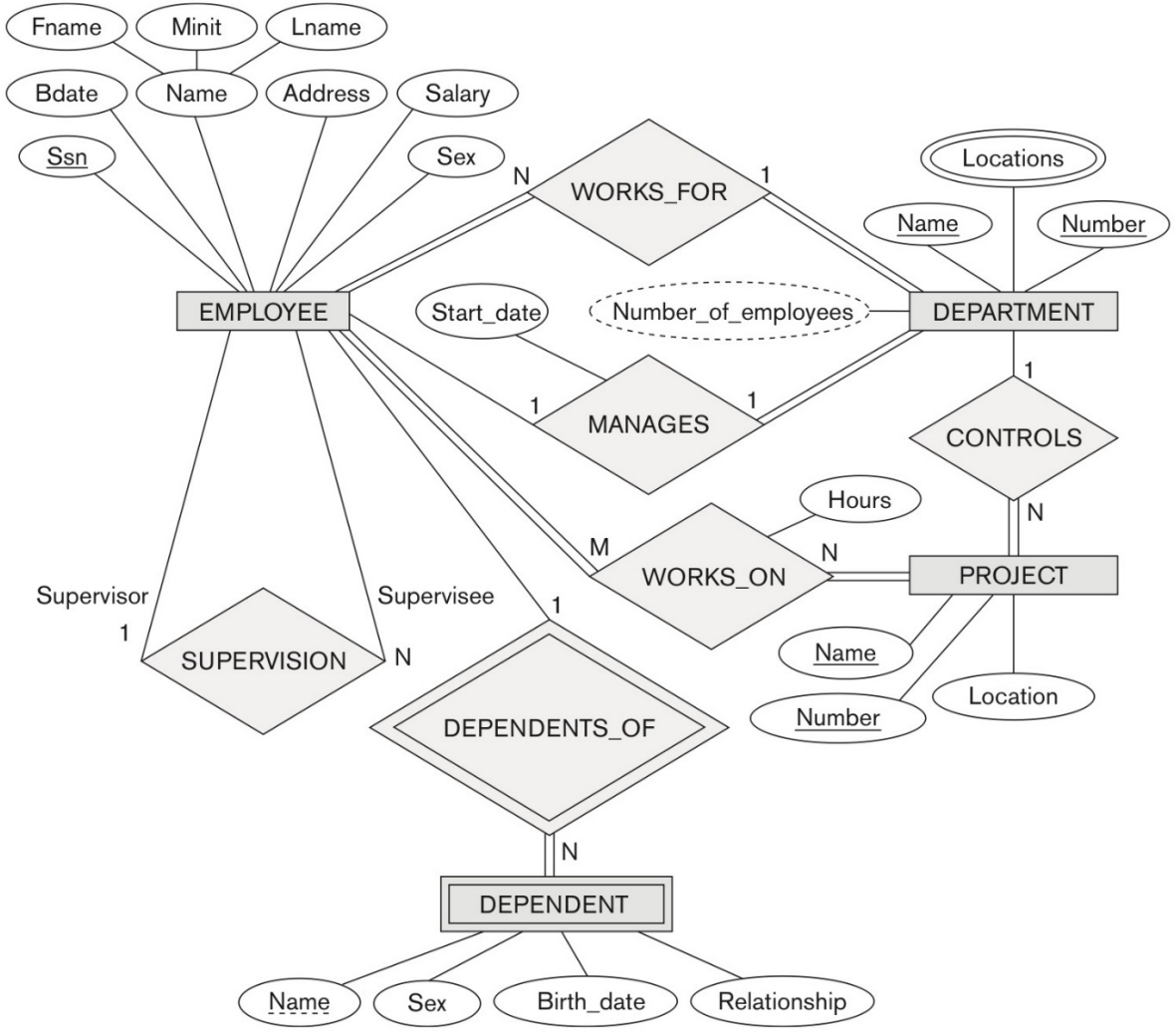
Chapter objectives:

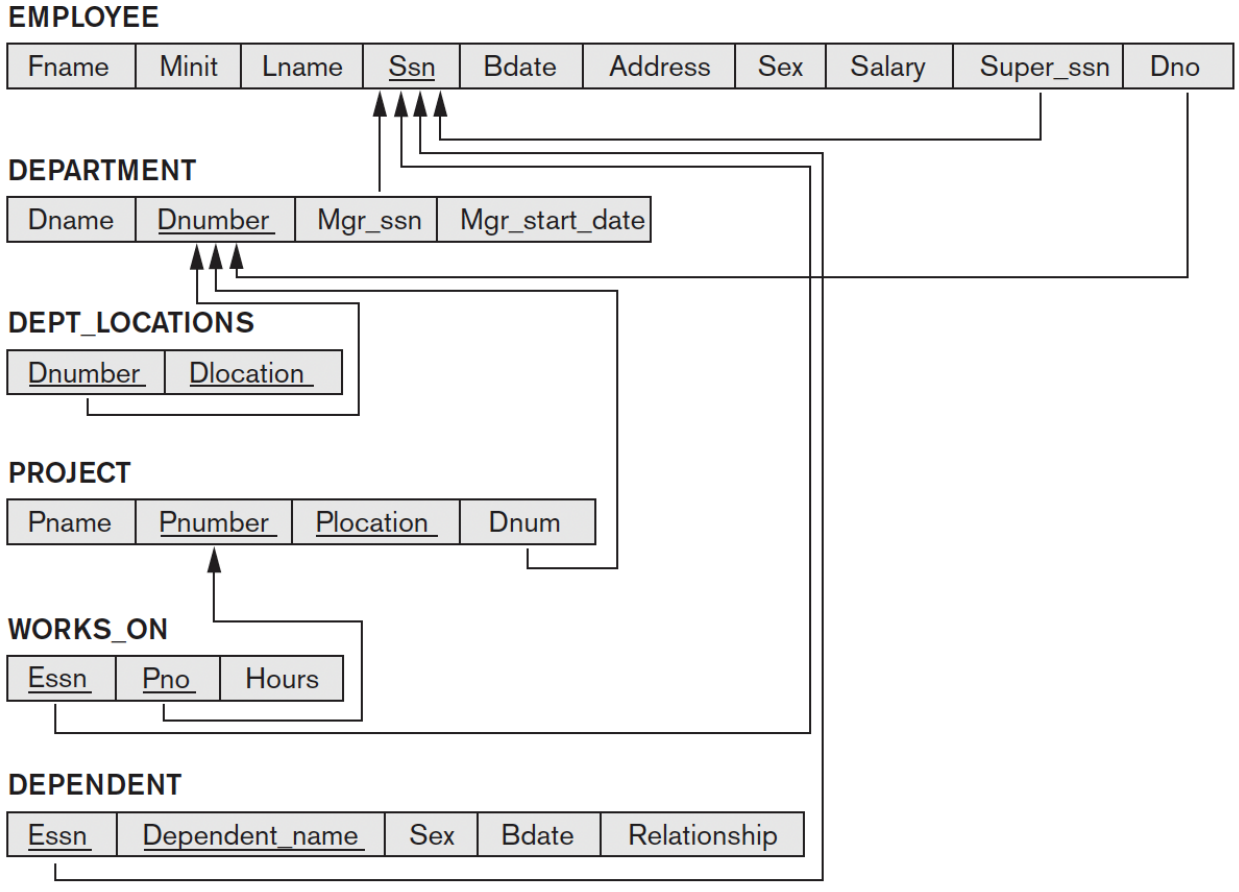
Learn ER/EER mapping to relational model. This includes mapping all the following:

- Regular/weak Entity Types
- Binary 1:1, 1:N, M:N relationship types
- Multivalued attributes.
- N-ary relationship types.
- Specialization/Generalization and union types.
- Learn Data Definition Language to create, drop, and alter database

GOALS DURING MAPPING

- Preserve all information (that includes entities, relationships and all attributes)
- Maintain the constraints as much as possible (Relational Model cannot preserve all constraints- e.g., max cardinality ratio such as 1:10 in ER; exhaustive classification into subtypes, e.g., STUDENTS are specialized into Domestic and Foreign).
- Minimize null values.


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


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

ER-to-Relational Mapping Algorithm

- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relation Types
- Step 4: Mapping of Binary 1:N Relationship Types.
- Step 5: Mapping of Binary M:N Relationship Types.
- Step 6: Mapping of Multivalued attributes.
- Step 7: Mapping of N-ary Relationship Types.

ER-TO-RELATIONAL MAPPING ALGORITHM

Step 1: Mapping of Regular Entity Types.

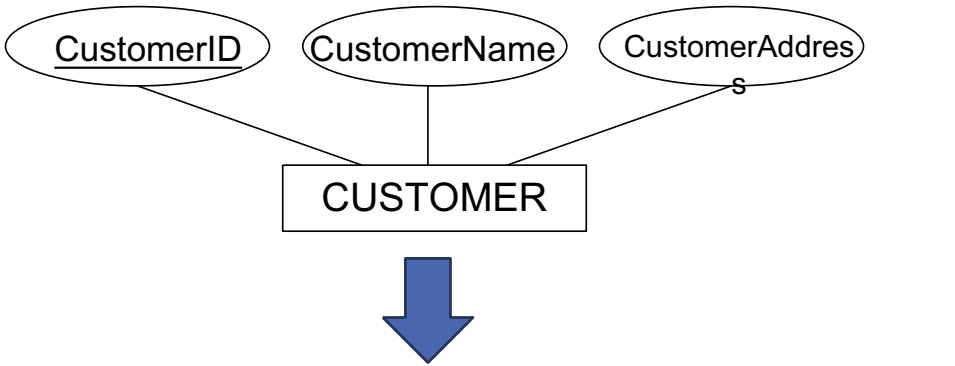
- For each regular (strong) entity type E in the ER schema, create a relation R that includes all the simple attributes of E.
- Choose one of the key attributes of E as the primary key for R.
- If the chosen key of E is composite, the set of simple attributes that form it will together form the primary 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.

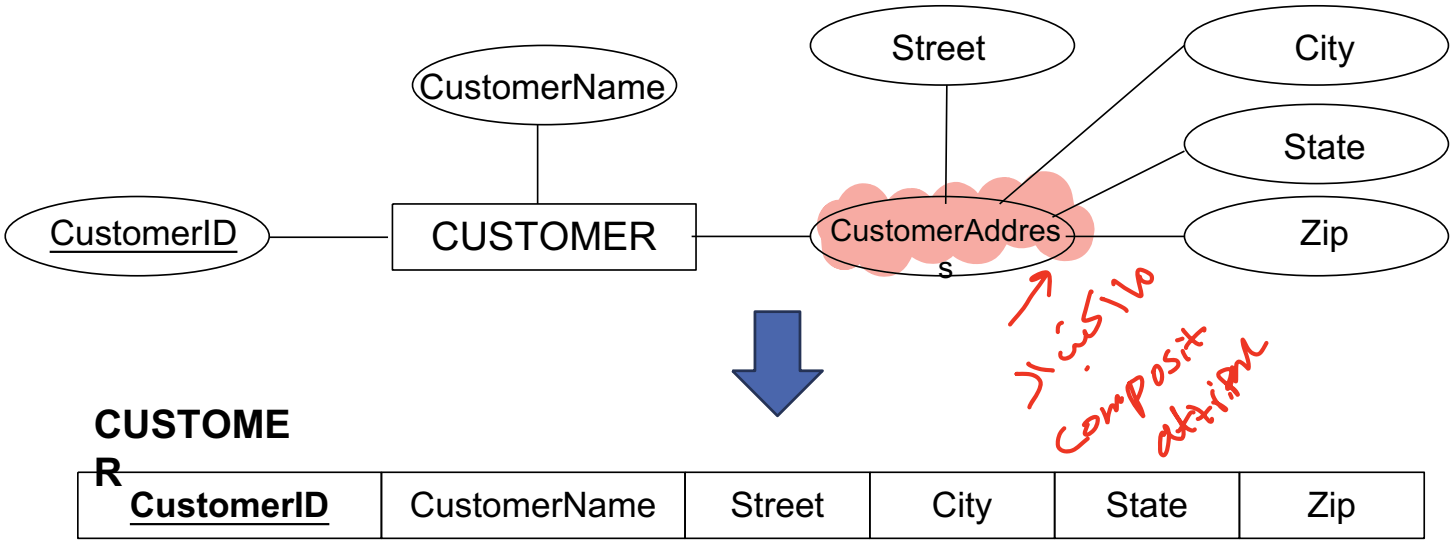
- SSN, DNUMBER, and PNUMBER are the primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT as shown.

Strong

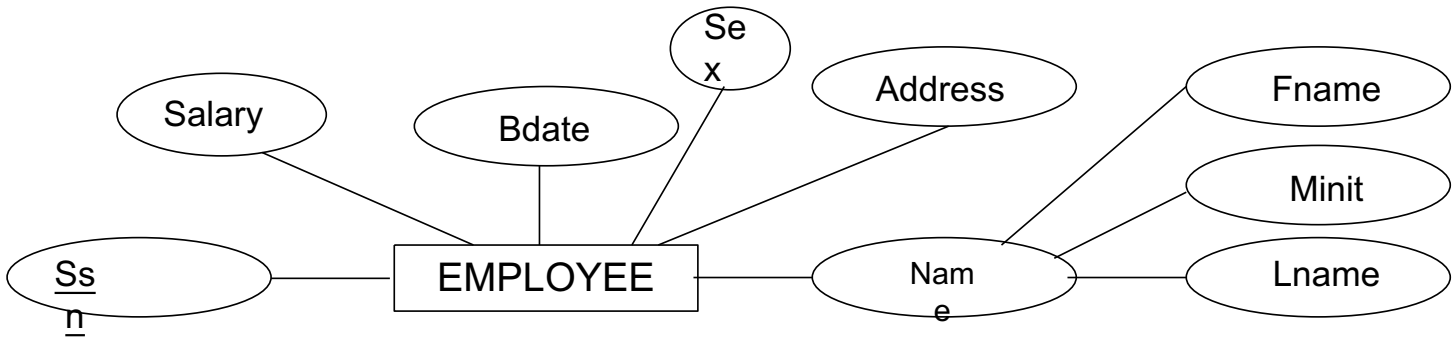
REGULAR ENTITY TYPES WITH SIMPLE ATTRIBUTES



REGULAR ENTITY TYPES WITH COMPOSITE ATTRIBUTES



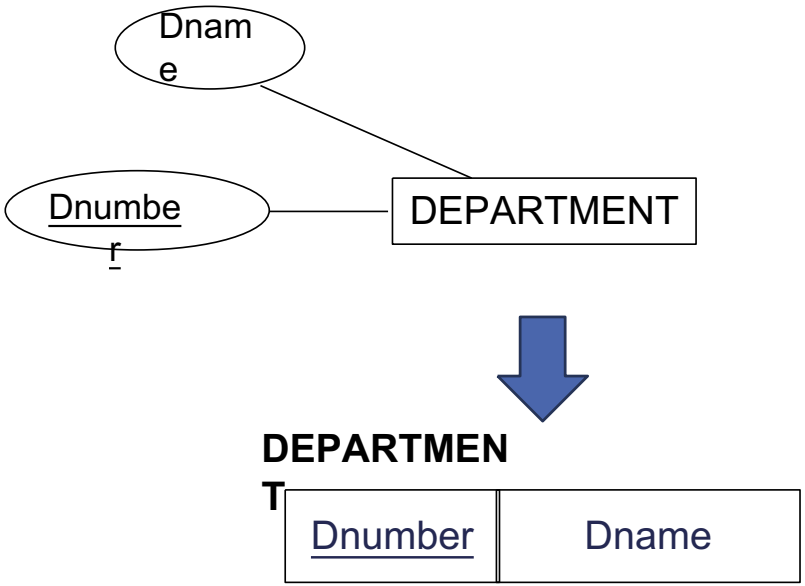
REGULAR ENTITY TYPES (EMPLOYEE)



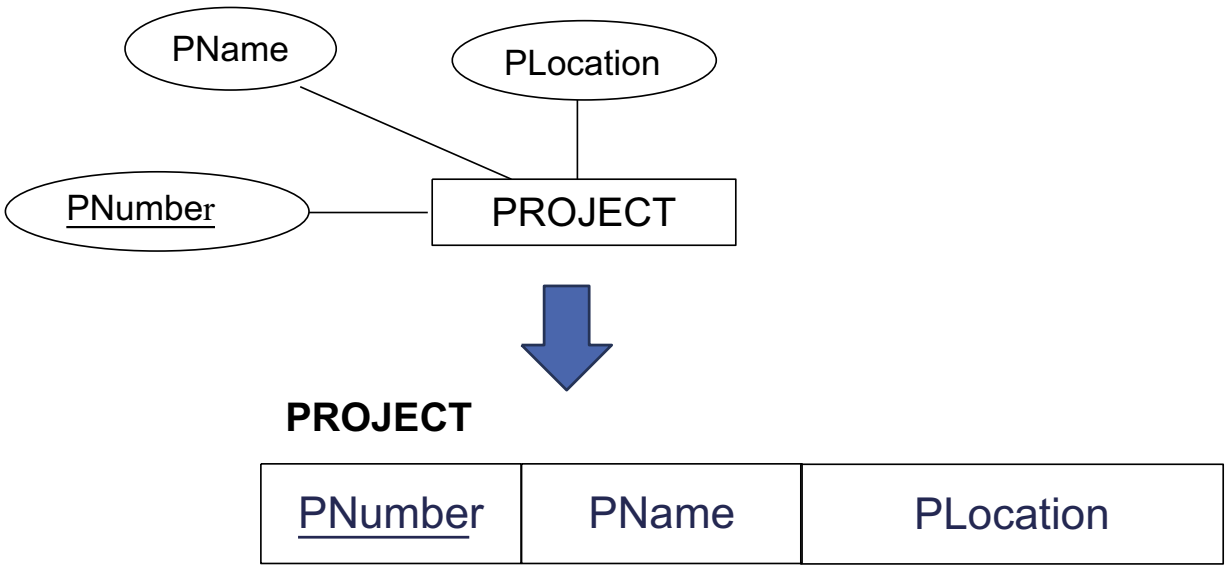
EMPLOYEE

Ssn	Fname	Minit	Lname	Address	Salary	Bdate	Sex

REGULAR ENTITY TYPES (DEPARTMENT)



REGULAR ENTITY TYPES (PROJECT)



ER-TO-RELATIONAL MAPPING ALGORITHM

Step 2: Mapping of Weak Entity Types

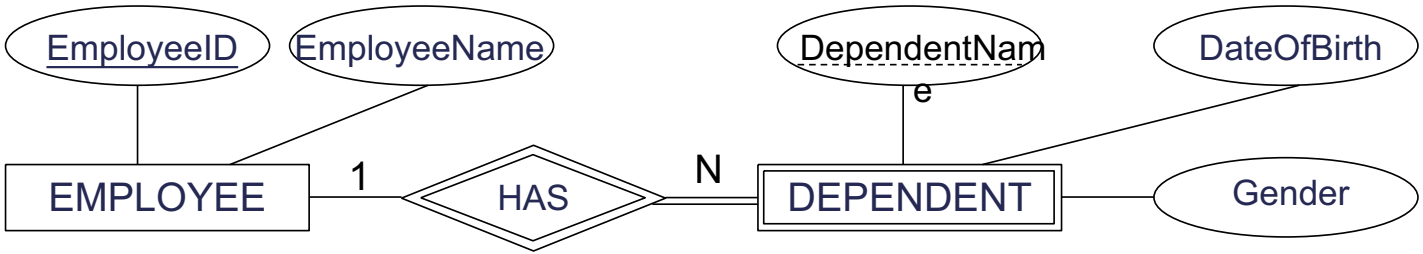
- For each weak entity type W in the ER schema with owner entity type E, create a relation R & include all simple attributes (or simple components of composite attributes) of W as attributes of R.
- Also, include as foreign key attributes of R the primary key attribute(s) of the relation(s) that correspond 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.

ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

Example: Create the relation DEPENDENT in this step to correspond to the weak entity type DEPENDENT.

- Include the primary key SSN of the EMPLOYEE relation as a foreign key attribute of DEPENDENT (renamed to ESSN).
- The primary key of the DEPENDENT relation is the combination {ESSN, DEPENDENT_NAME} because DEPENDENT_NAME is the partial key of DEPENDENT.

WEAK ENTITY TYPES



EMPLOYEE

EmployeeID	EmployeeName

Foreign key

DEPENDENT

DependentName	EmployeeID	DateOfBirth	Gender

Composite primary key

DDL BASICS

```
CREATE TABLE table_name(
  column_1 datatype,
  column_2 datatype,
  column_3 datatype,
  ....
);
```

Example:

```
CREATE TABLE CUSTOMER(
      Customer_id
      number(4),
      CustomerName
      varchar(30),
      CustomerAddress
      varchar(50),
);
```

DDL BASICS (PRIMARY KEY)

Example:

```
CREATE TABLE CUSTOMER(
     Customer_id number(4) NOT NULL,
     CustomerName varchar(30) NOT NULL,
     CustomerAddress varchar(50),
     PRIMARY KEY (Customer_id)
);
```

DDL BASICS (COMPOSITE KEYS AND NAMING CONSTRAINTS)

```
CREATE TABLE Persons (
    ID int NOT NULL,
    LastName varchar(255) NOT NULL,
    FirstName varchar(255),
    Age int,
    CONSTRAINT PK_Person PRIMARY KEY (ID,LastName)
);
```

DDL BASICS (FOREIGN KEYS)

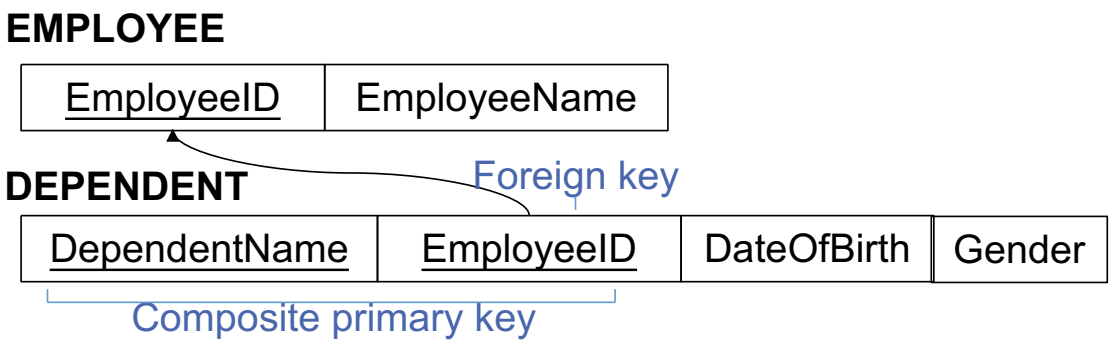
```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    FOREIGN KEY (PersonID) REFERENCES Persons(PersonID)
);
```

OR you can name the constraint:

```
CREATE TABLE Orders (
    OrderID int NOT NULL,
    OrderNumber int NOT NULL,
    PersonID int,
    PRIMARY KEY (OrderID),
    CONSTRAINT FK_PersonOrder FOREIGN KEY (PersonID)
    REFERENCES Persons(PersonID)
);
```

ACTIVITY

Let us try to write DDL statements to define the relations in this diagram. Two relations, Employee and Dependent with the appropriate key constraints.



ACTIVITY (SOLUTION)

```
CREATE TABLE Employee (
    ID int NOT NULL,
    EmployeeName varchar(255) NOT NULL,
    PRIMARY KEY (ID)
);

CREATE TABLE DEPENDENT (
    Name NOT NULL,
    EmployeeID int NOT NULL,
    DateOfBirth date,
    Gender varchar(1) DEFAULT 'f',
    FOREIGN KEY (EmployeeID) REFERENCES Employee(ID)

    CONSTRAINT PK_Dependent PRIMARY KEY (EmployeeID,Name)
);
```

A BIT MORE DDL

ALTER command:

```
ALTER TABLE name_of_table ADD column_name column_definition;
```

Add column, change column, etc.

DROP command:

```
DROP TABLE EMPLOYEE;
```

Drop table or drop database

Other commands:

```
TRUNCATE TABLE Table_Name; //clears table of all tuples

RENAME TABLE Old_Table_Name TO New_Table_Name;  //renames
table
```

SOME SQL DATA TYPES (MYSQL)

Data type	Description
CHAR(size)	A FIXED length string
VARCHAR(size)	A VARIABLE length string
INT(*size*)/FLOAT/DOUBLE	An integer, float, or double
BLOB(size)	For BLOBs (Binary Large Objects). Holds up to 65,535 bytes of data
ENUM(val1, val2, val3, ...)	A string object that can have only one value, chosen from a list of possible values. You can list up to 65535 values in an ENUM list. If a value is inserted that is not in the list, a blank value will be inserted.
DATE	A date. Format: YYYY-MM-DD. The supported range is from '1000-01-01' to '9999-12-31'

DDL CONSTRAINTS

Attribute constraints:

- `NOT NULL`: the column must have a value
- `UNIQUE`: the column value must be unique in the table
- `CHECK`: the column value conforms to an arbitrary condition

Foreign key conditions:

- `FOREIGN KEY(fk) REFERENCES refrenced_table(pk)`
  `ON DELETE CASCADE / ON UPDATE CASCADE`
- `FOREIGN KEY(fk) REFERENCES refrenced_table(pk)`
  `ON DELETE SET NULL`

ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

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.

There are three possible approaches:

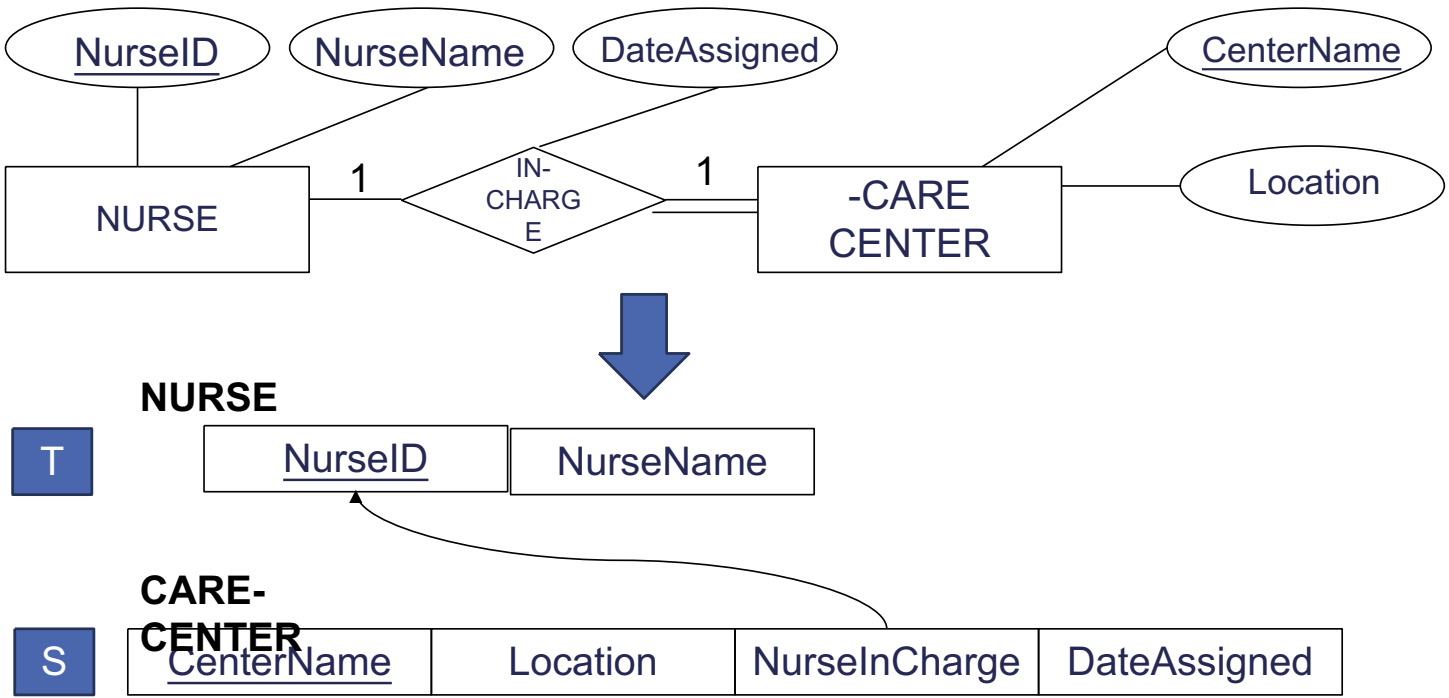
1. Foreign Key (2 relations) approach
2. Merged relation (1 relation) option
3. Cross-reference or relationship relation (3 relations) option

MAPPING OF BINARY 1:1 RELATIONSHIP TYPES

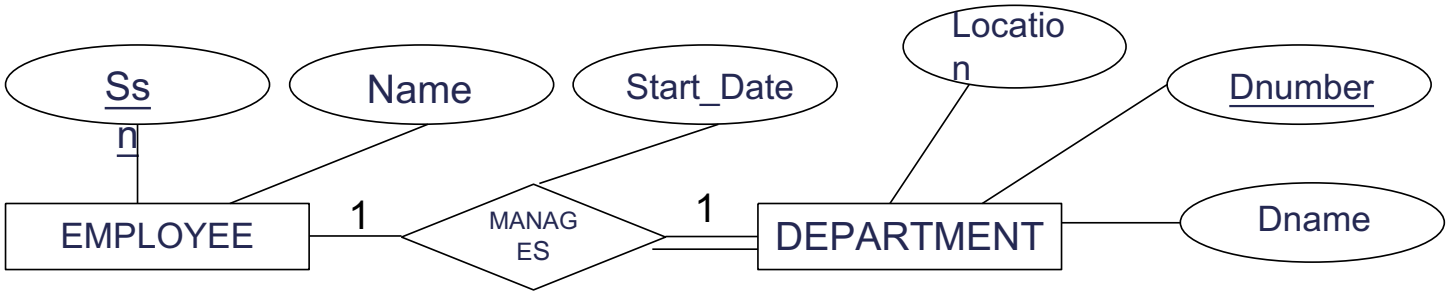
Foreign key approach : (Used when one participation is total and the other is partial)

- Most useful and should be followed unless special conditions exist.
- Choose one of the relations - **S**, say - and include as 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.
- Include all the simple attributes (or simple components of composite attributes) of the 1:1 relationship type R as attributes of S.

1:1 FOREIGN KEY APPROACH



1:1 FOREIGN KEY APPROACH



EMPLOYEE

T

Ssn	Name

DEPARTMEN

T

S

Dnumber	Dname	Location	Mgr_ssn	Mgr_Start_date

1:1 FOREIGN KEY APPROACH

What will happen if the primary key of S is included as a foreign key in T?

In the previous example, the S relation is DEPARTMENT and T relation is EMPLOYEE. Thus, the primary key of S is Dept_managed will be the foreign key for the EMPLOYEE table. But it will have NULL values for employee tuples who do not manage a department. If only 10% of employee manage a department, then 90% of the foreign keys would be NULL in this case.

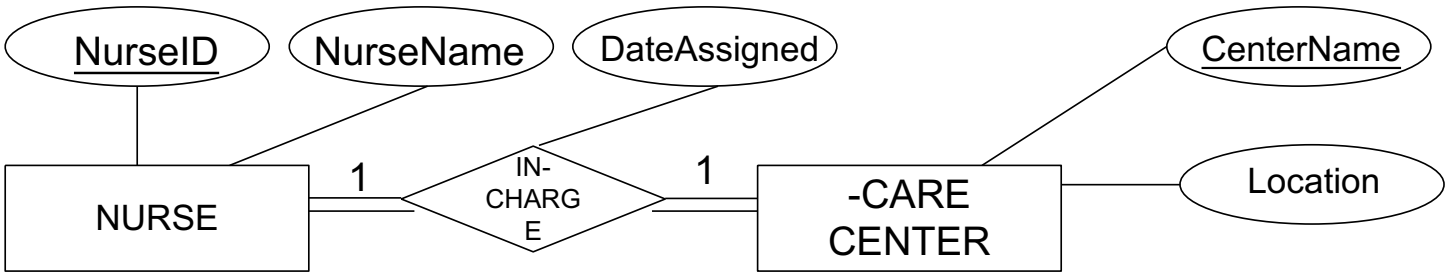
Another possibility is to have foreign keys in both relations S and T redundantly, but this incurs a penalty for consistency maintenance.

MAPPING OF BINARY 1:1 RELATIONSHIP TYPES

Merged relation option:

Merging the two entity types and the relationship into a single relation. (Used when both participations are total)

1:1 MERGED RELATION APPROACH



NURSE

NurseID	NurseName	CenterName	Location	DateAssigned

The primary Key will be NurseId, because NURSE is the name of the relation. We can also choose CenterName as the Primary Key. In that case Care_Center will be the name of the relation.

MAPPING OF BINARY 1:1 RELATIONSHIP TYPES

Cross-reference or relationship relation option: (Both participations are partial)

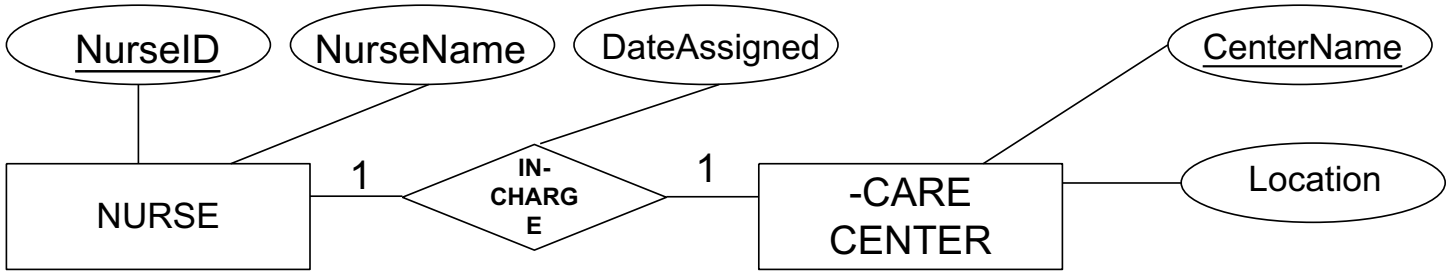
Setting up a third relation R for the purpose of cross-referencing the primary keys of the two relations S and T representing the entity types.

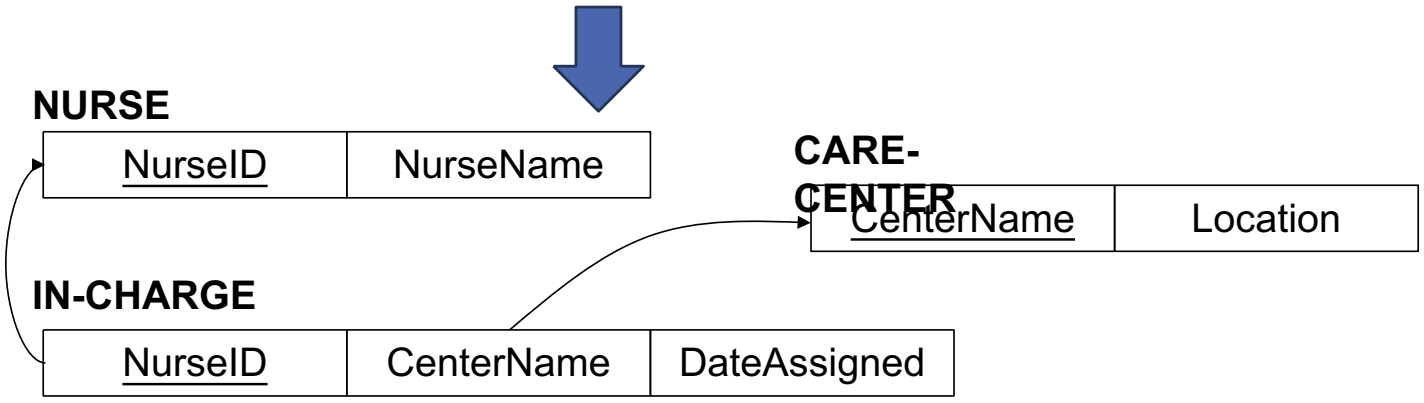
The relationship R is called a relationship relation (or sometimes a lookup table), because each tuple in R represents a relationship instance that relates one tuple from S with one tuple of T.

The primary key of the relationship relation will be either one of the foreign keys that reference the participating entity relations.

This alternative is useful when few relationship instances (i.e. partial participation) exist to avoid NULL values in foreign keys.

1:1 RELATIONSHIP RELATION APPROACH





ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

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

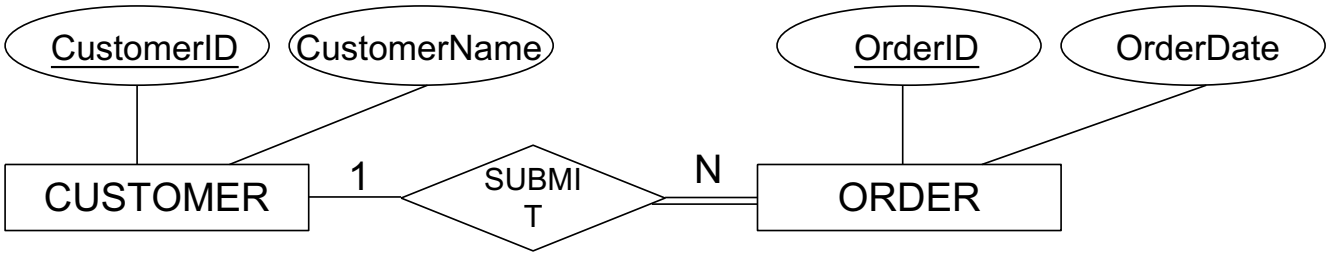
- For each regular binary 1:N relationship type R, identify the relation S that represent the participating entity type at the N-side of the relationship type.
- Include as foreign key in S the primary key of the relation T that represents the other entity type participating in R.
- Include any simple attributes of the 1:N relationship type as attributes of S.

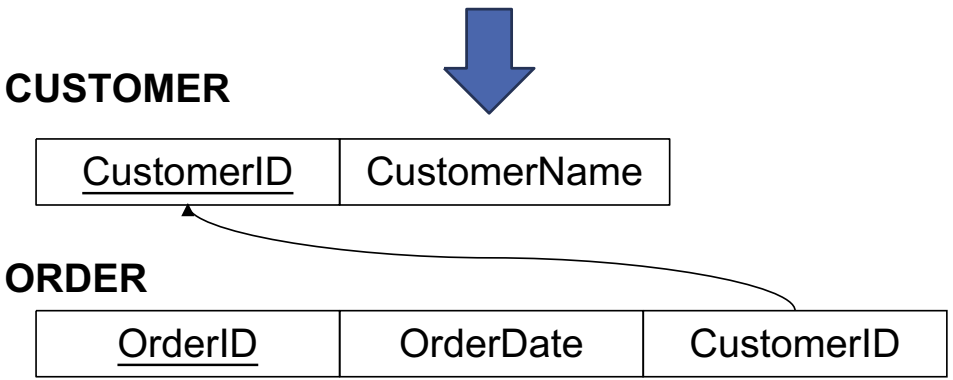
Example: 1:N relationship type: WORKS_FOR

For WORKS_FOR we include the primary key DNUMBER of the DEPARTMENT relation as foreign key in the EMPLOYEE relation and call it DNO.

An alternative approach is to use a Relationship relation (cross referencing relation) – this is rarely done and has little benefit.

1:N FOREIGN KEY APPROACH





Foreign key goes in the many side of the relationship

1:N FOREIGN KEY APPROACH

If CustomerId is added as a Foreign Key in the Order relation (the many side)

OrderID	CustomerID
O001	C001
O002	C001
O008	C001
O003	C005
O004	C002
O005	C002
O006	C002

If OrderId is added as a Foreign Key in the Customer Relation (the 1 side)

CustomerID	OrderID
C001	O001, O002, O008
C005	O003
C002	O004, O005, O006

This results in multi-valued attributes, which we do not want

ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

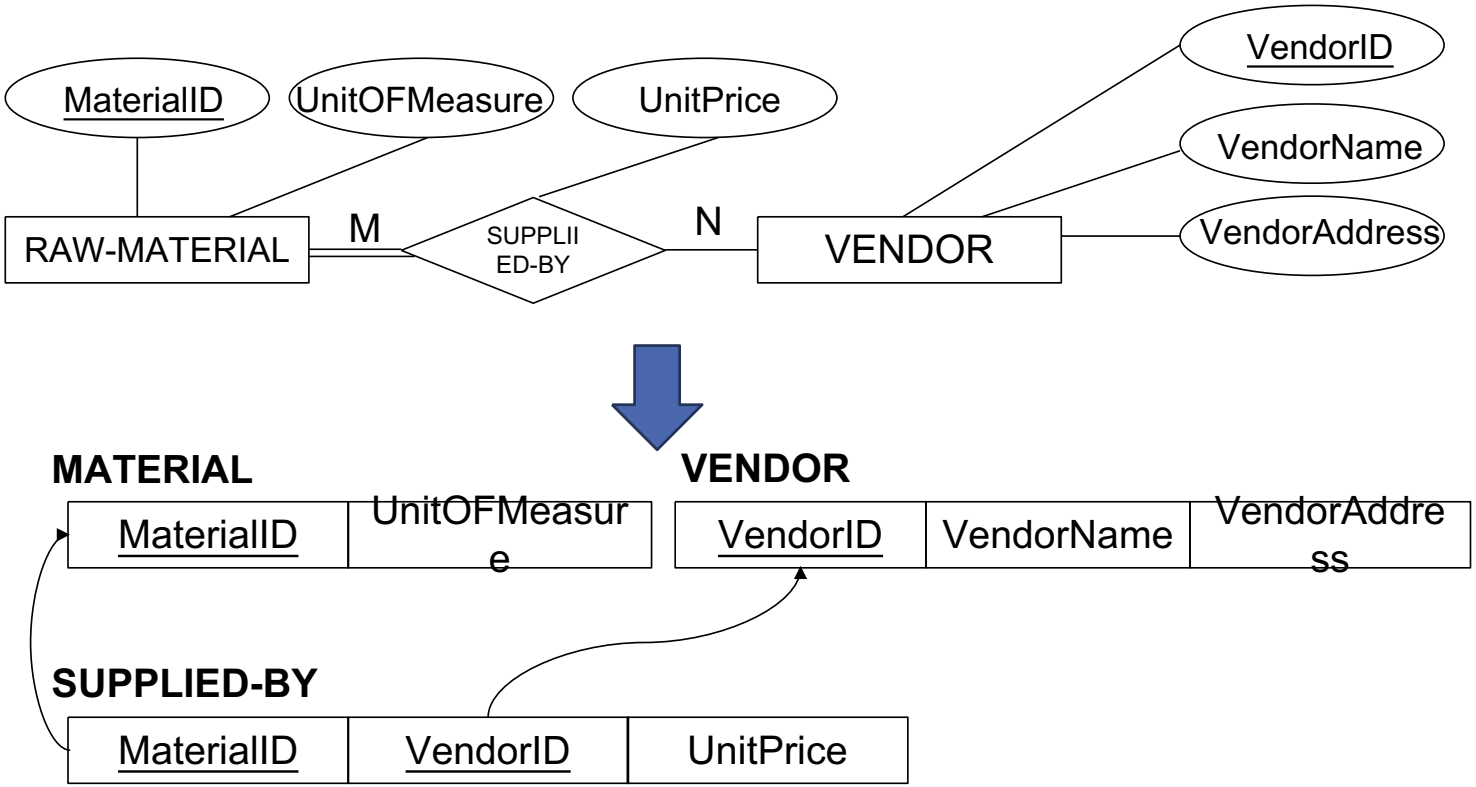
Step 5: Mapping of Binary M:N Relationship Types.

- For each regular binary M:N relationship type R, *create a new relation* S to represent R. This is a *relationship relation.*
- Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types; *their combination will form the primary key* of S.
- Also include any simple attributes of the M:N relationship type (or simple components of composite attributes) as attributes of S.

Example: The M:N relationship type WORKS_ON from the ER diagram is mapped by creating a relation WORKS_ON in the relational database schema.

- The primary keys of the PROJECT and EMPLOYEE relations are included as foreign keys in WORKS_ON and renamed PNO and ESSN, respectively.
- Attribute HOURS in WORKS_ON represents the HOURS attribute of the relationship type. The primary key of the WORKS_ON relation is the combination of the foreign key attributes {ESSN, PNO}.

N:M MAPPING TO RELATIONAL

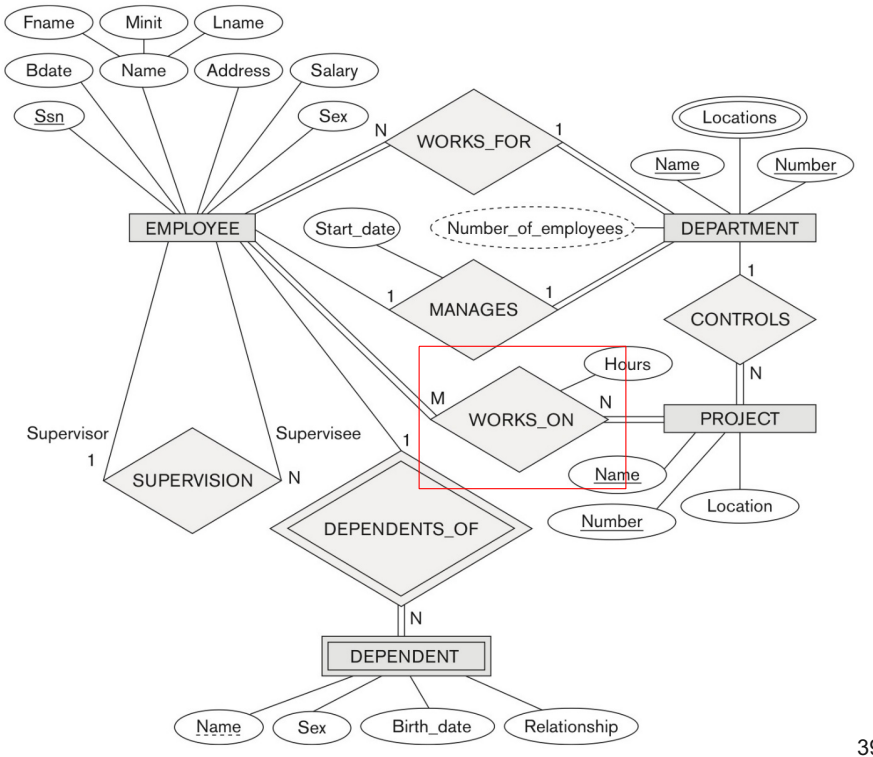


ACTIVITY

Map the WORKS_ON relationship into a relation.

Include the primary key and foreign keys.

Write the DDL for this specific relation.



ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

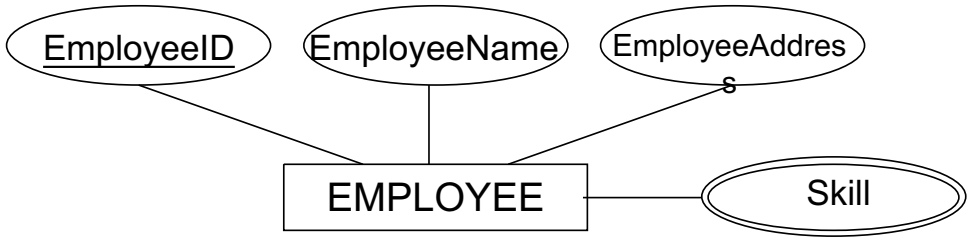
Step 6: Mapping of Multivalued attributes.

- For each multivalued attribute A, create a new relation R.
- This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type of relationship type that has A as an attribute.
- The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components.

Example: The relation DEPT_LOCATIONS is created.

- The attribute DLOCATION represents the multivalued attribute LOCATIONS of DEPARTMENT, while DNUMBER-as foreign key-represents the primary key of the DEPARTMENT relation.
- The primary key of R is the combination of {DNUMBER, DLOCATION}.

MAPPING MULTI-VALUED ATTRIBUTES



EMPLOYEE

EmployeeID	EmployeeName	EmployeeAddress

SKILLS

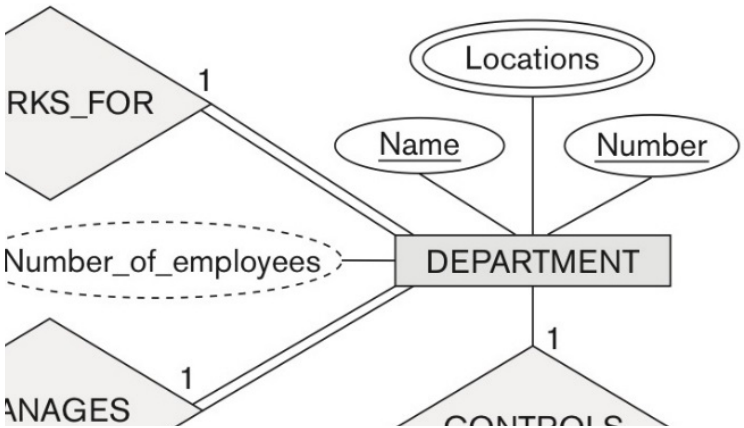
EmployeeID	Skill

ACTIVITY

Map the locations attribute into a relation DEPT_LOCATIONS.

Include the primary key and foreign keys.

Write the DDL for this specific relation.



ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

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.

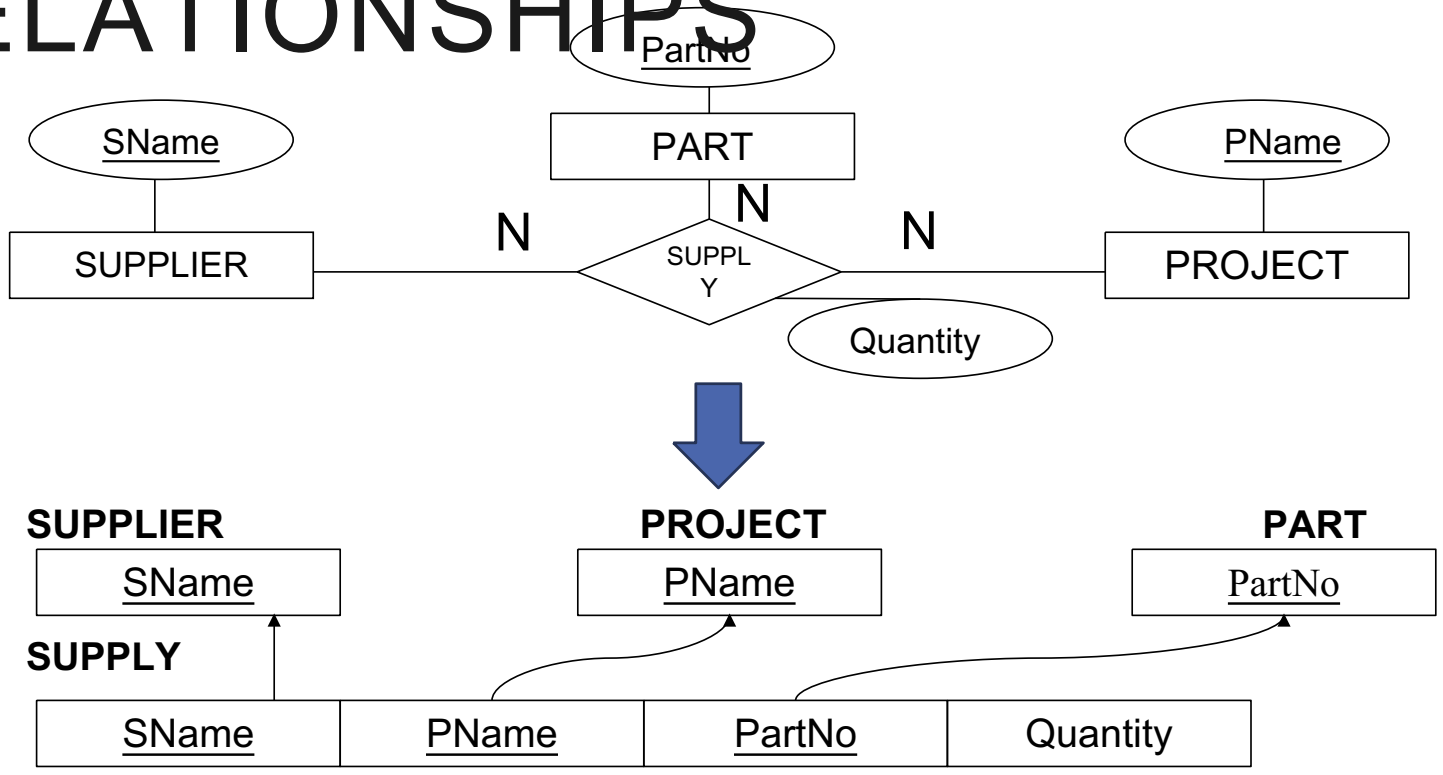
Example: The relationship type SUPPLY in the ER on the next slide.

- 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}

MAPPING OF N-ARY RELATIONSHIPS

- The primary key of S is usually a combination of all the foreign keys that reference the relations representing the participating entity types.
- If the cardinality constraints on any of the entity types E participating in R is 1, then the primary key of S should not include the foreign key attribute that references the relation E' corresponding to E.

MAPPING OF N-ARY RELATIONSHIPS



UNARY RELATIONSHIPS

1:N unary relationships:

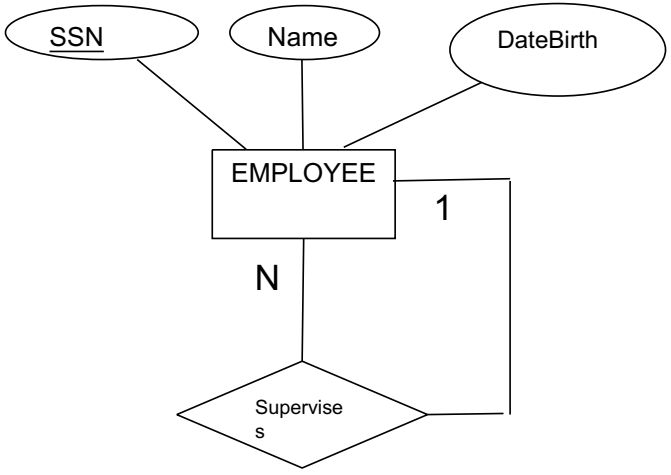
1. With this type of relationship, ONE relation is created: one entity represents the entity type in the relationship consisting of PK and FK in the same relation. The FK is referenced to the PK of the same relation
2. Any non key attribute of the relationship is included in the relation.

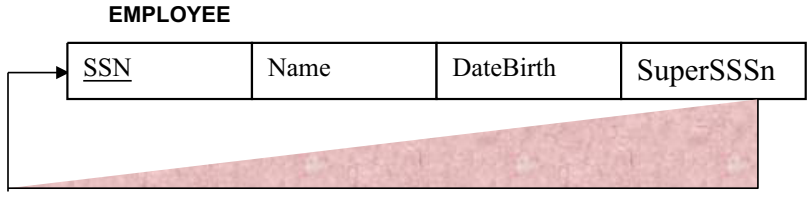
N:M unary relationships:

1. With this type of relationship, two relations are created: one to represent the entity type in the relationship and the other an associative relation to represent the M:N relationship.
2. The primary key of the associative relation consists of two attributes. These attributes (which need not to have the same name) both take their values from the primary key of the other relation. Any non key attribute of the relationship is included in the associative relation

1:N UNARY RELATIONSHIPS

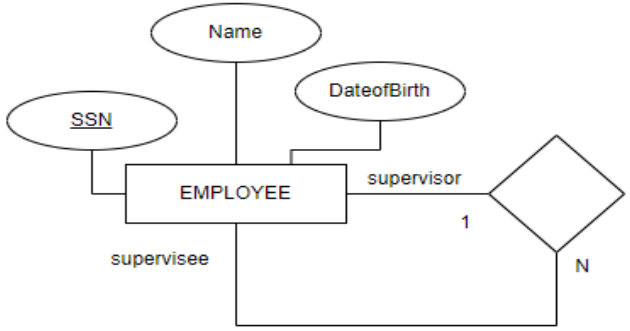
This approach will result in NULL values for employees without supervisors

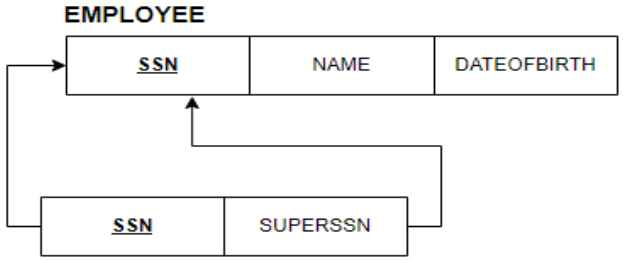




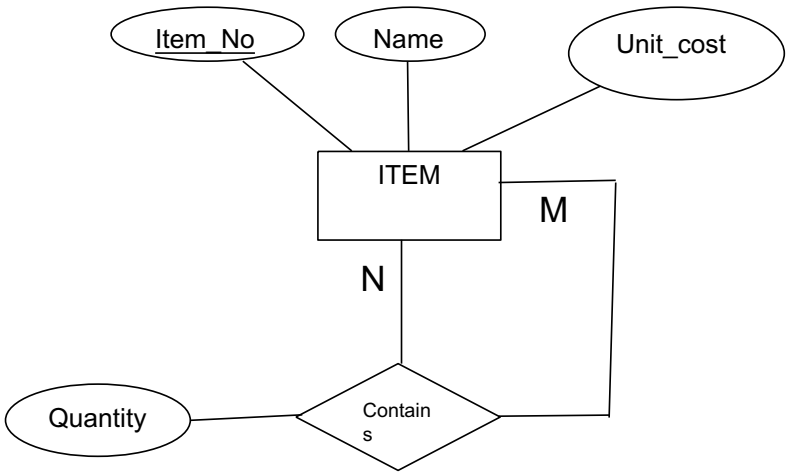
1:N UNARY RELATIONSHIPS (PARTIAL BOTH SIDES)

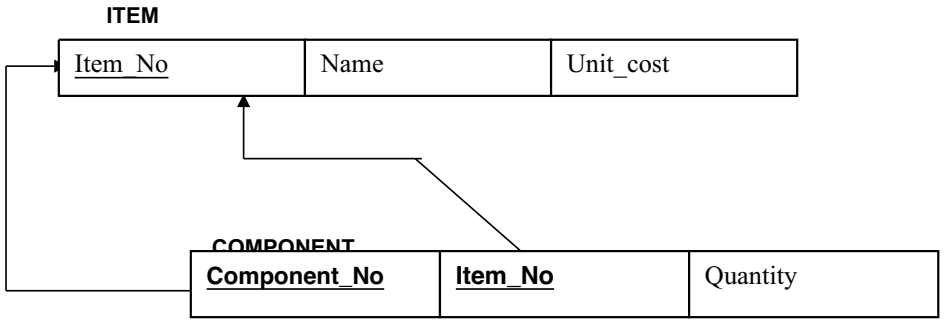
This approach will not result in any NULL values.



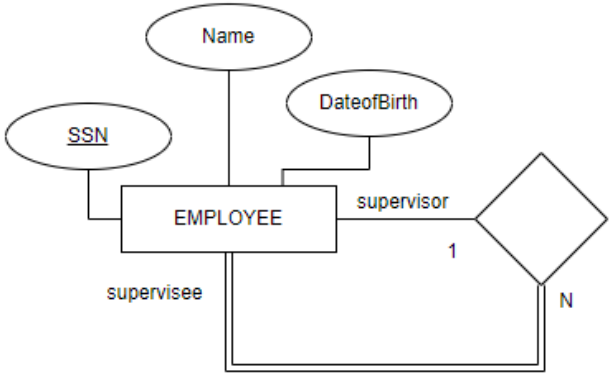


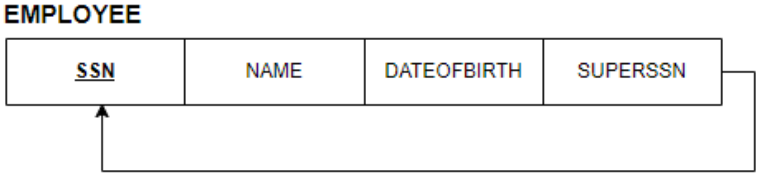
M:N UNARY RELATIONSHIPS





1:N UNARY RELATIONSHIPS (PARTIAL ONE SIDE)





SUMMARY OF MAPPING CONSTRUCTS AND CONSTRAINTS

Table 9.1 Correspondence between ER and Relational Models

ER MODEL	RELATIONAL MODEL
Entity type	*Entity* relation
1:1 or 1:N relationship type	Foreign key (or *relationship* relation)
M:N relationship type	*Relationship* relation and *two* foreign keys
n-ary relationship type	*Relationship* relation and *n* foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Value set	Domain
Key attribute	Primary (or secondary) key

ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

Step8: Options for Mapping Specialization or Generalization.

- Convert 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, into relational schemas using one of the four following options:
  - Option 8A: Multiple relations-Superclass and subclasses
  - Option 8B: Multiple relations-Subclass relations only
  - Option 8C: Single relation with one type attribute
  - Option 8D: Single relation with multiple type attributes

MAPPING EER MODEL CONSTRUCTS TO RELATIONS

Option 8A: Multiple relations-Superclass and subclasses

Create a relation L for C with attributes Attrs(L) = {k,a1,…an} and PK(L) = k. Create a relation Li for each subclass Si, 1 < i < m, with the attributesAttrs(Li) = {k} U {attributes of Si} and PK(Li)=k.

This option works for any specialization (total or partial, disjoint or over -lapping).

Option 8B: Multiple relations-Subclass relations only

Create a relation Li for each subclass Si, 1 < i < m, with the attributes Attr(Li) = {attributes of Si} U {k,a1…,an} and PK(Li) = k.

This option only works for a specialization whose subclasses are total (every entity in the superclass must belong to (at least) one of the subclasses).

MAPPING EER MODEL CONSTRUCTS TO RELATIONS

Option 8C: Single relation with one type attribute

Create a single relation L with attributes Attrs(L) = {k,a_1,…a_n} U {attributes of S_1} U…U {attributes of S_m} U {t} and PK(L) = k. The attribute t is called a type (or **discriminating**) attribute that indicates the subclass to which each tuple belongs

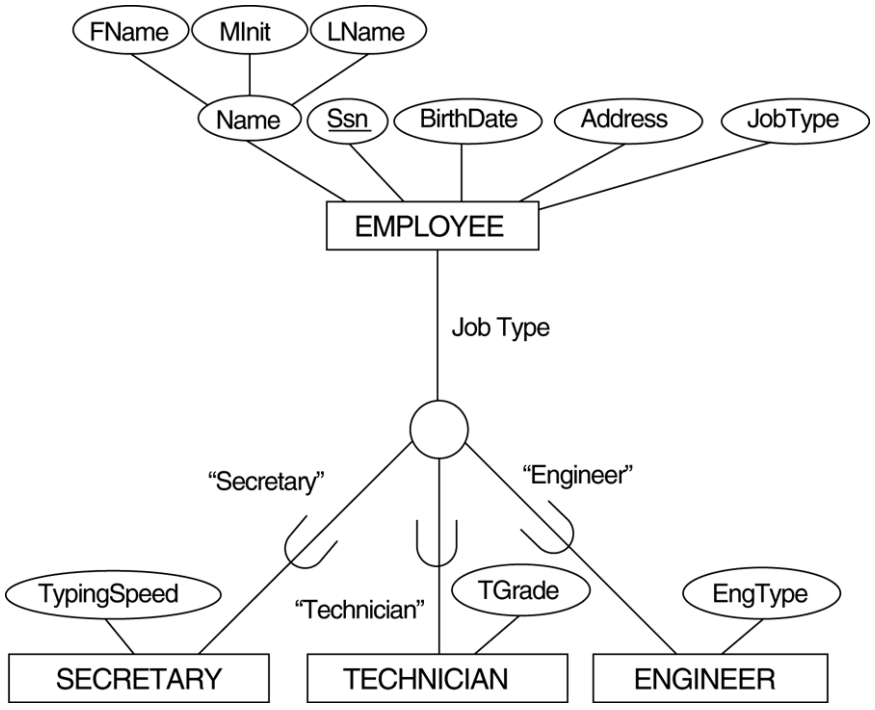
This approach works for disjoint sub-classes.

Option 8D: Single relation with multiple type attributes

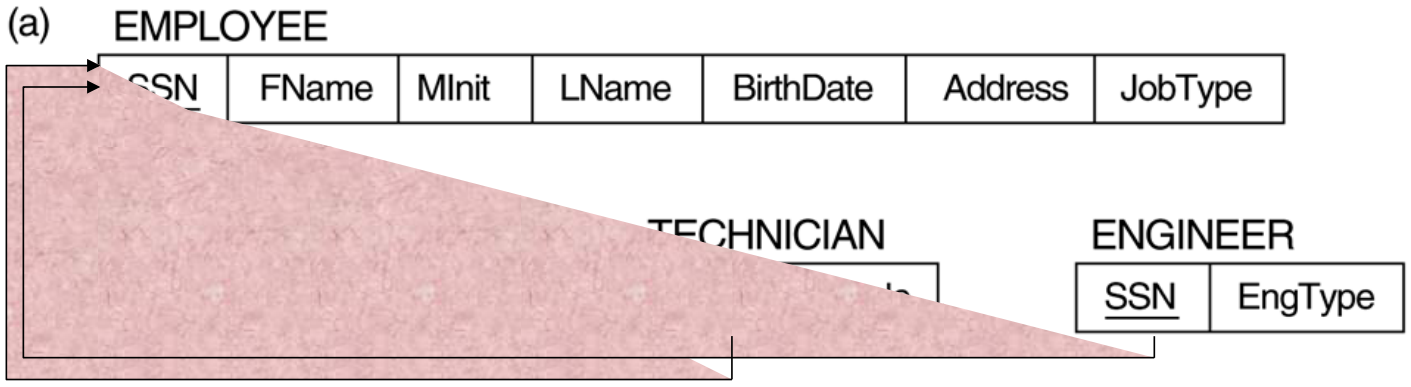
Create a single relation schema L with attributes Attrs(L) = {k,a_1,…a_n} U {attributes of S_1} U…U {attributes of S_m} U {t_1, t_2,…,t_m} and PK(L) = k. Each t_i, 1 < I < m, is a Boolean type attribute indicating whether a tuple belongs to the subclass S_i.

This approach works for overlapping sub-classes.

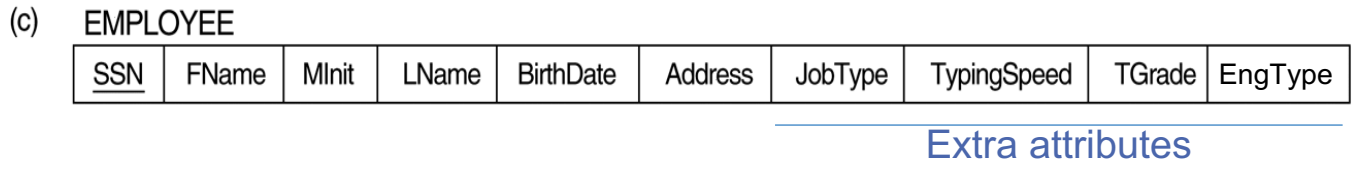
EXAMPLE: EMPLOYEE AND JOB TYPE



OPTION 8A: MULTIPLE RELATIONS-SUPERCLASS AND SUBCLASSES

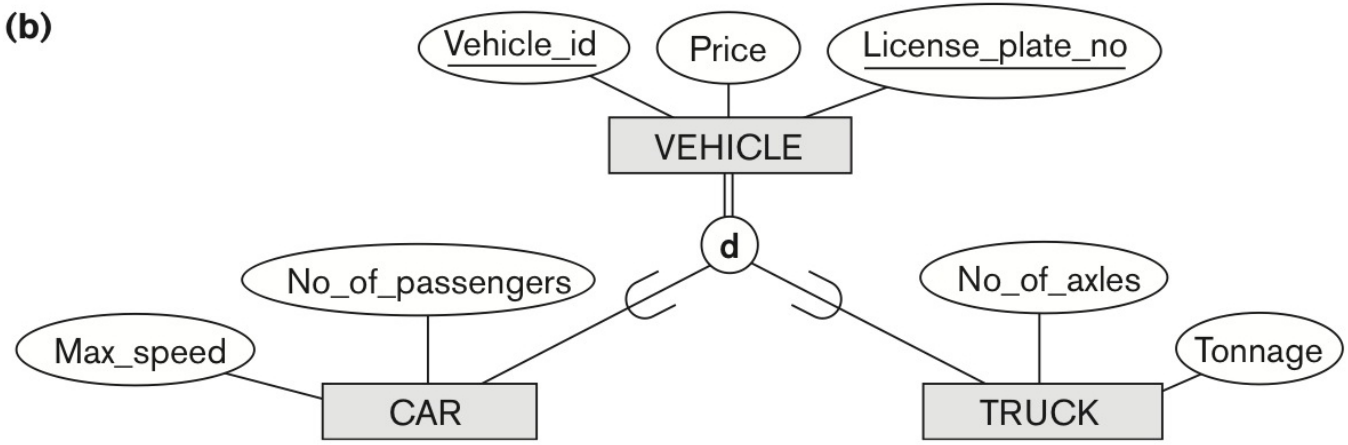


OPTION 8C: EXTRA ATTRIBUTES



EMPLOYEE is a single relation, with a type attribute JobType that determines the type of the employee. The other attributes (TypingSpeed, Tgrade, and EngType) belong to the different employee types (SECRETARY, TECHNICIAN, and ENGINEER respectively).

EXAMPLE: VEHICLE, CAR OR TRUCK



OPTION 8B: SUB-CLASSES ONLY

CAR and TRUCK contain both VEHICLE shared attributes and CAR, TRUCK specific attributes.

(b)

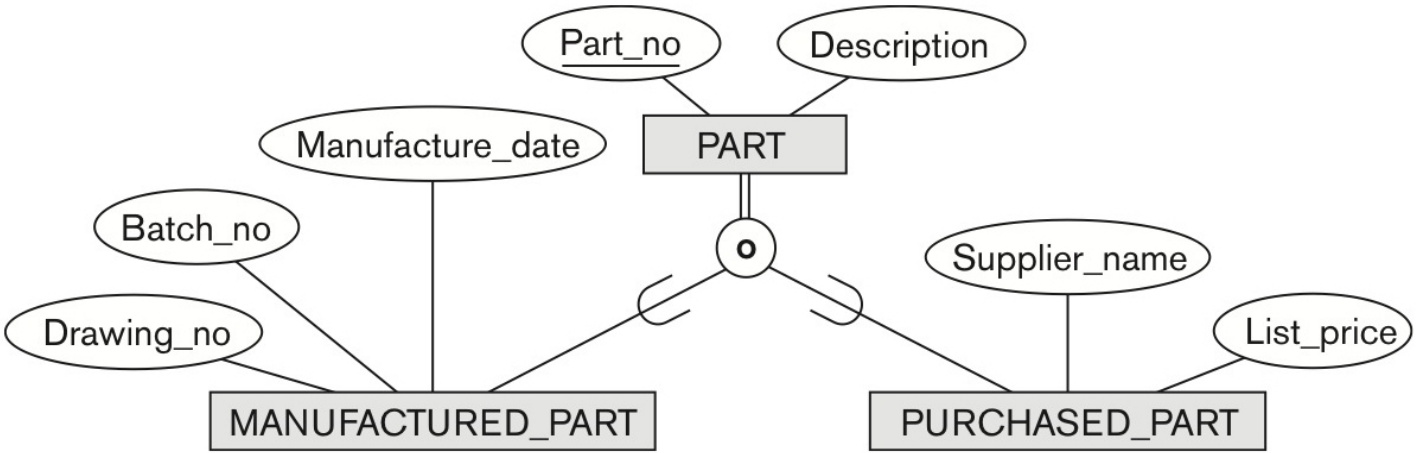
CAR

VehicleId	LicensePlateNo	Price	MaxSpeed	NoOfPassengers

TRUCK

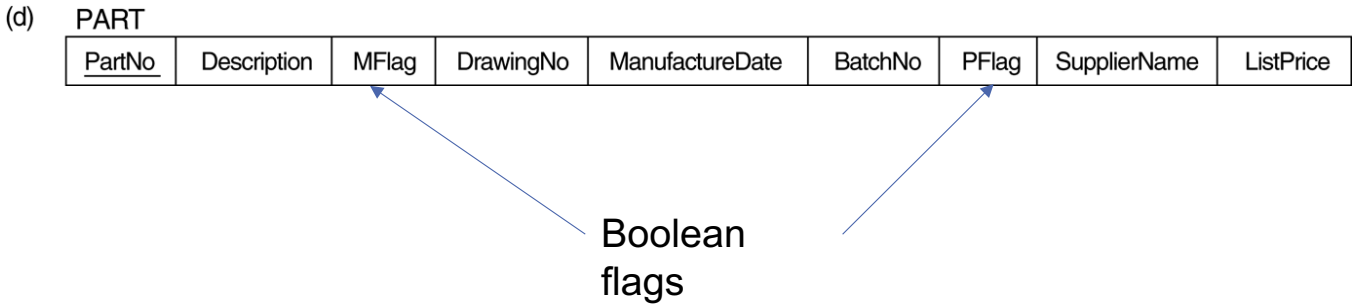
VehicleId	LicensePlateNo	Price	NoOfAxles	Tonnage

EXAMPLE: PART ENTITY



OPTION 8D: BOOLEAN FLAGS

The Boolean flags determine the type of PART. If Mflag is 1 and Pflag is 0, it is a MANUFACTURED_PART. If Mflag is 0 and Pflag is 1, it is a PURCHASED_PART. If both are 1, the part is both manufactured and purchased.



DIFFERENT OPTIONS FOR MAPPING

Next Slide : Options for mapping specialization or generalization.

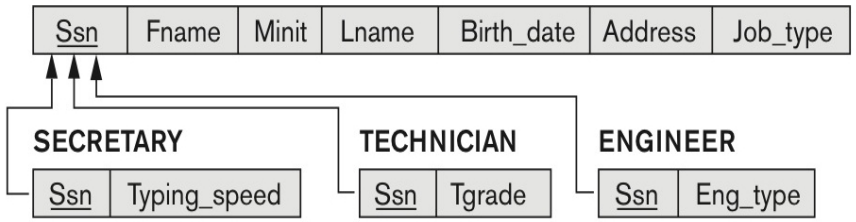
(a) Mapping the EER schema using option 8A.

(b) Mapping the EER schema using option 8B.

(c) Mapping the EER schema using option 8C.

(d) Mapping EER schema using option 8D with Boolean type fields Mflag and Pflag.

(a) EMPLOYEE



(b) CAR

Vehicle_id	License_plate_no	Price	Max_speed	No_of_passengers

TRUCK

Vehicle_id	License_plate_no	Price	No_of_axles	Tonnage

(c) EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type	Typing_speed	Tgrade	Eng_type

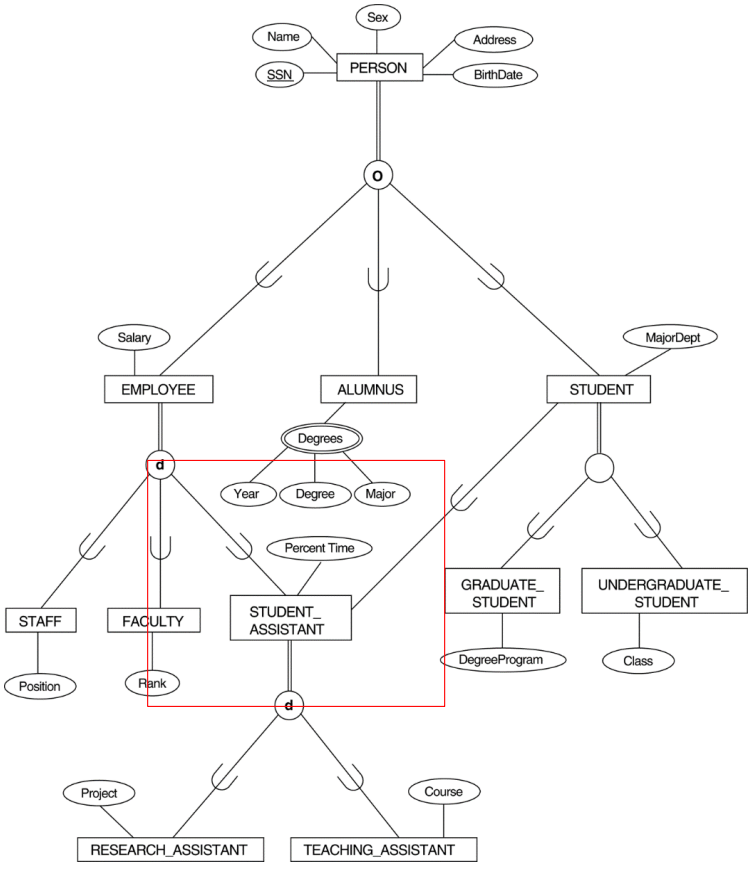
(d) PART

Part_no	Description	Mflag	Drawing_no	Manufacture_date	Batch_no	Pflag	Supplier_name	List_price

MAPPING OTHER EER MODEL CONSTRUCTS TO RELATIONS

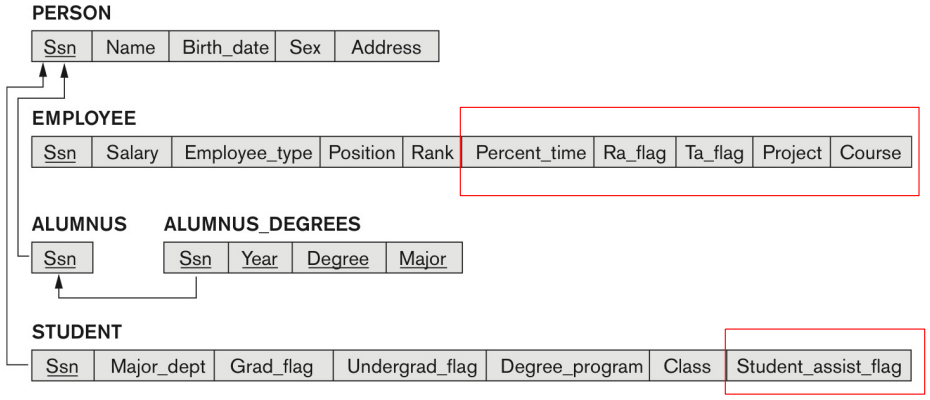
Mapping of Shared Subclasses (Multiple Inheritance)

- A shared subclass, such as STUDENT_ASSISTANT, is a subclass of several classes, indicating multiple inheritance. These classes must all have the same key attribute; otherwise, the shared subclass would be modeled as a category.
- We can apply any of the options discussed in Step 8 to a shared subclass, subject to the restriction discussed in Step 8 of the mapping algorithm. Below both 8C and 8D are used for the shared class STUDENT_ASSISTANT.



SHARED SUBCALSS

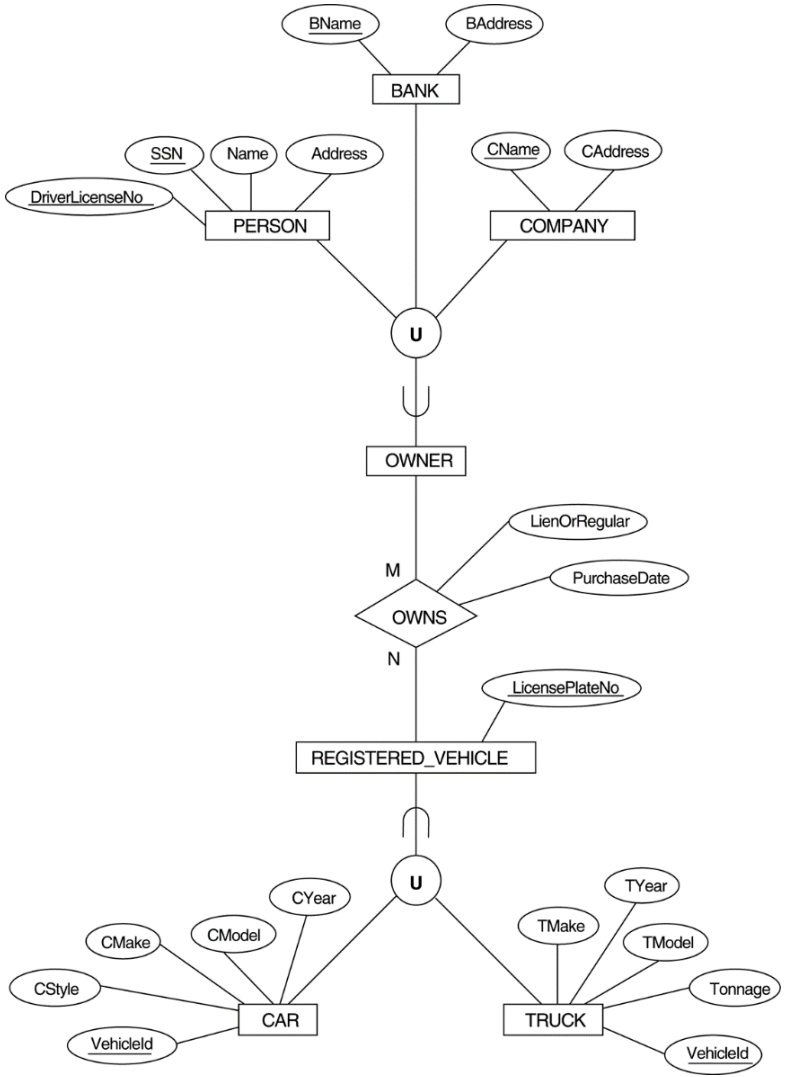
RELATIONAL MAPPING

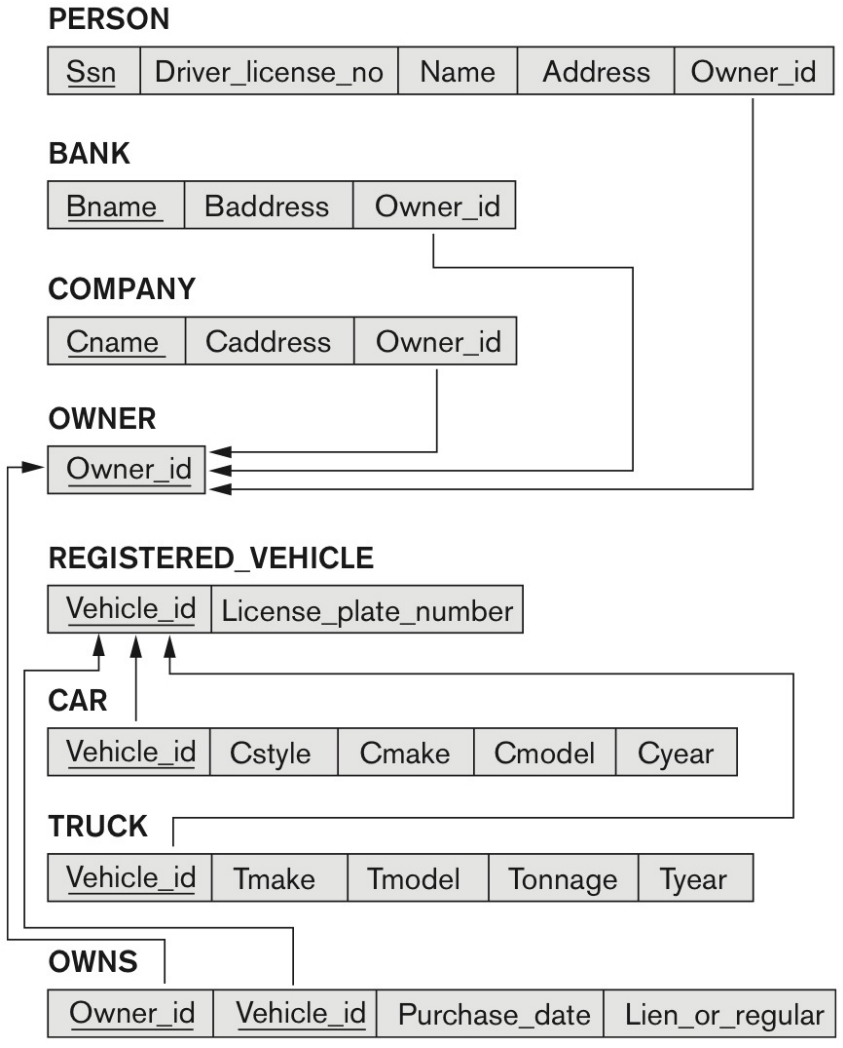


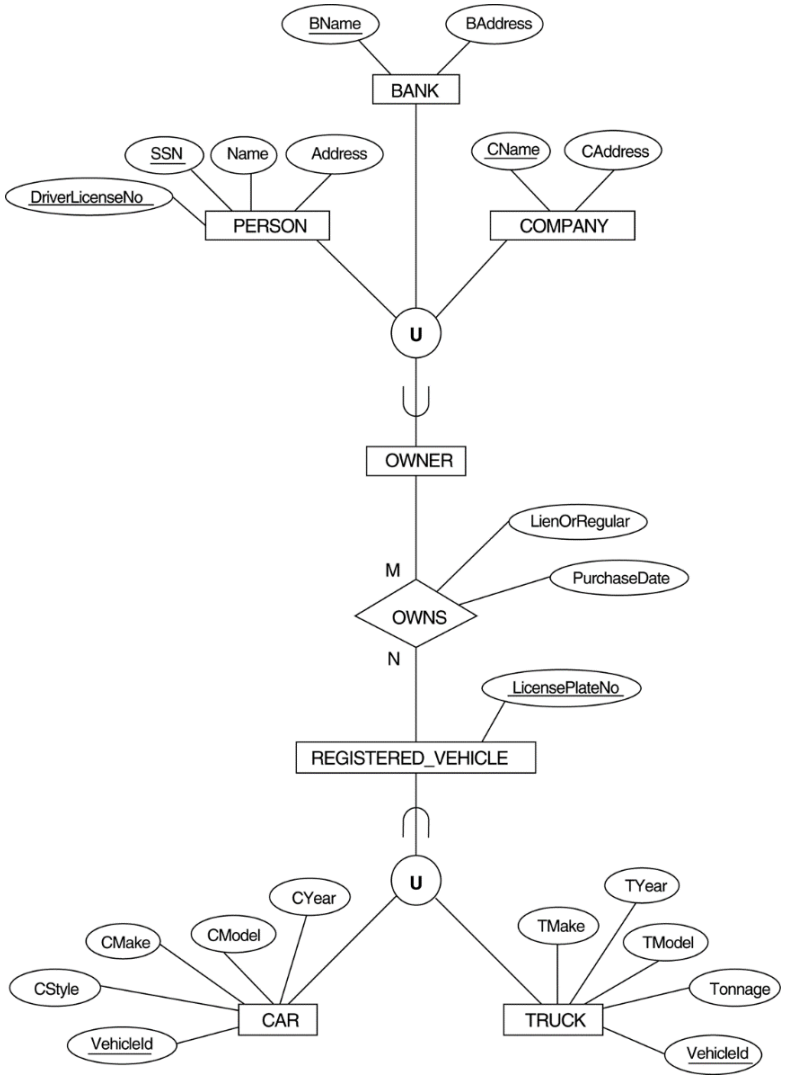
ER-TO-RELATIONAL MAPPING ALGORITHM (CONTD.)

Step 9: Mapping of Union Types (Categories).

- For mapping a category whose defining superclass have different keys, it is customary to specify a new key attribute, called a surrogate key, when creating a relation to correspond to the category.
- In the example below we can create a relation OWNER to correspond to the OWNER category and include any attributes of the category in this relation. The primary key of the OWNER relation is the surrogate key, which we called OwnerID.


BName
BAddress
BANK
SSN
Name
Address
DriverLicenseNo
PERSON
CName
CAddress
COMPANY
U
OWNER
LienOrRegular
M
PurchaseDate
OWNS
N
LicensePlateNo
REGISTERED_VEHICLE
U
CYear
CMake
CModel
CStyle
CAR
VehicleId
TYear
TMake
TModel
Tonnage
TRUCK
VehicleId


PERSON
Ssn
Driver_license_no
Name
Address
Owner_id
BANK
Bname
Baddress
Owner_id
COMPANY
Cname
Caddress
Owner_id
OWNER
Owner_id
REGISTERED_VEHICLE
Vehicle_id
License_plate_number
CAR
Vehicle_id
Cstyle
Cmake
Cmodel
Cyear
TRUCK
Vehicle_id
Tmake
Tmodel
Tonnage
Tyear
OWNS
Owner_id
Vehicle_id
Purchase_date
Lien_or_regular



PERSON

Ssn	Driver_license_no	Name	Address	Owner_id

BANK

Bname	Baddress	Owner_id

COMPANY

Cname	Caddress	Owner_id

OWNER

Owner_id

REGISTERED_VEHICLE

Vehicle_id	License_plate_number

CAR

Vehicle_id	Cstyle	Cmake	Cmodel	Cyear

TRUCK

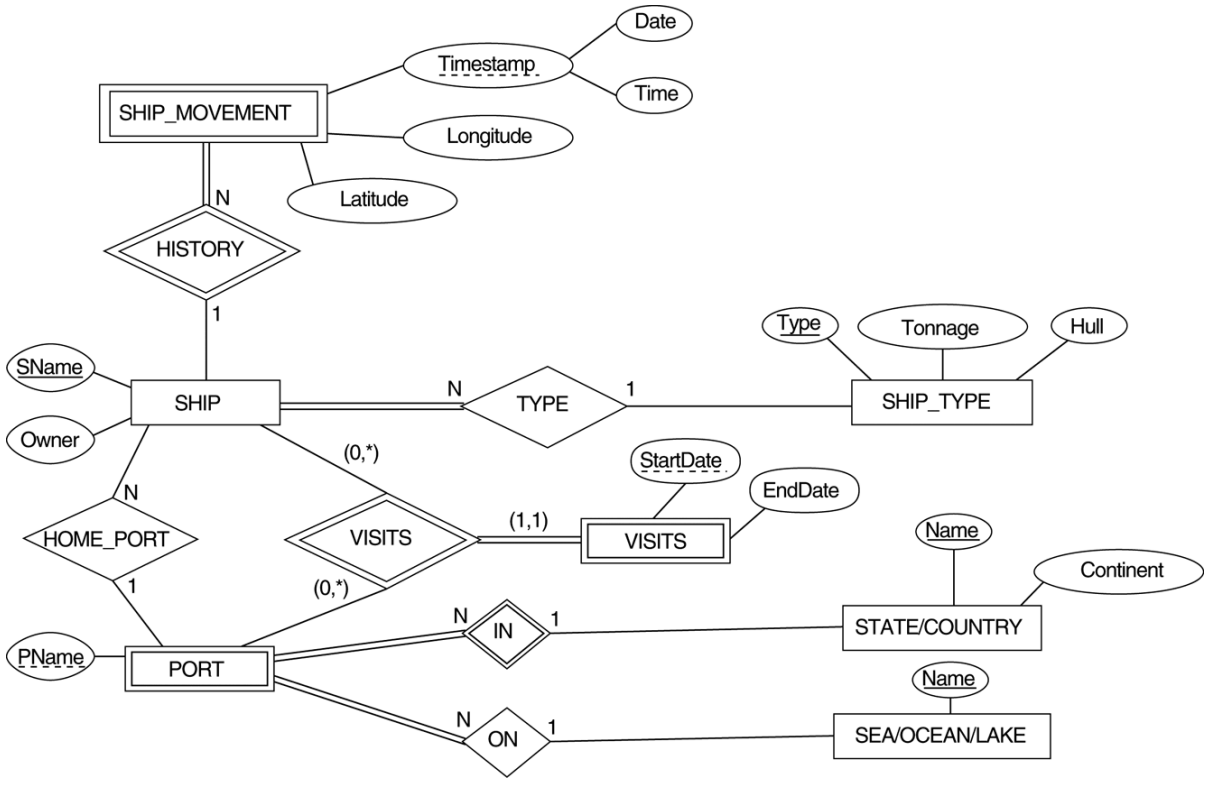
Vehicle_id	Tmake	Tmodel	Tonnage	Tyear

OWNS

Owner_id	Vehicle_id	Purchase_date	Lien_or_regular

ACTIVITY

Exercise 9.4 : Map this schema into a set of relations.



SOLUTION

Step 1: strong entities

SHIP (Sname, Owner)

SHIP_TYPE (Type, Tonnage, Hull)

STATE/COUNTRY (Name, Continent)

SEA/OCEAN/LAKE (Name)

Step 2: weak entities

SHIP_MOVEMENT (ShipName (FK), Date, time, latitude, longitude)

PORT (SCName (FK), SOLName (FK), Pname)

VISIT (Pname, SCName (FK), Sname (FK), StartDate, EndDate)

SOLUTION

Step 3: 1:1

None

Step 4: N:1

SHIP (Sname, Owner, PortName, SCName (FK), Stype(FK)) UPDATE

Step5: N:M and step6: Multi-valued attributes

None

Step 7: N-ary relationship

VISITS_RELATIONSHIP(ShipName(FK), PortName, SCName (FK), (FK), VisitStartDate(FK))
We combine with VISIT

No EER structures, no need for further steps

FINAL SOLUTION

SHIP (Sname, Owner, Pname (FK), Stype(FK))

SHIP_TYPE (Type, Tonnage, Hull)

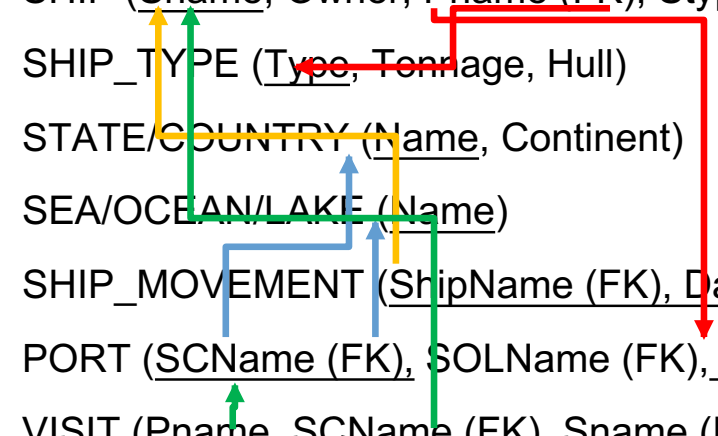
STATE/COUNTRY (Name, Continent)

SEA/OCEAN/LAKE (Name)

SHIP_MOVEMENT (ShipName (FK), Date, time, latitude, longitude)

PORT (SCName (FK), SOLName (FK), Pname)

VISIT (Pname, SCName (FK), Sname (FK), StartDate, EndDate)



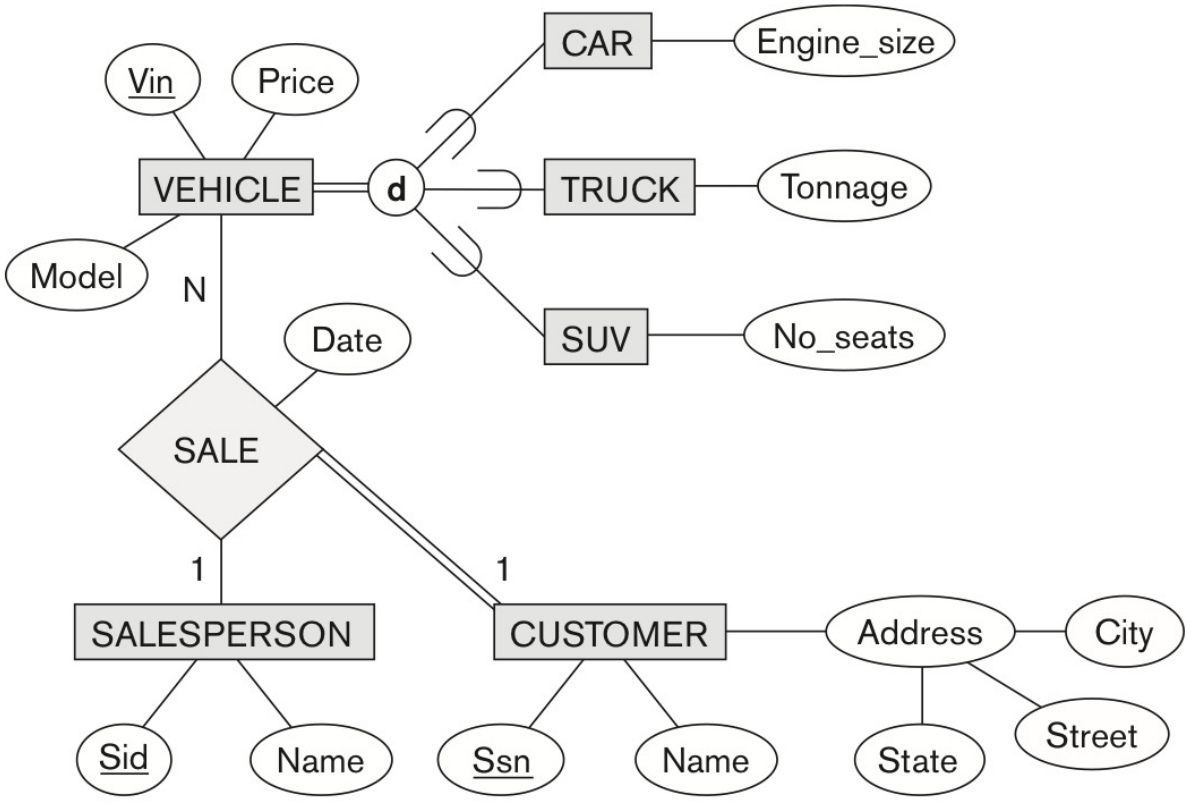
No need for two relations with subset of same attributes:

VISITS_RELATIONSHIP(ShipName(FK), PortName, SCName (FK), VisitStartDate(FK))

ACTIVITY

Exercise 9.9 : Map this schema into a set of relations

Solution in class



CHAPTER SUMMARY

ER-to-Relational Mapping Algorithm

- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relation Types
- Step 4: Mapping of Binary 1:N Relationship Types.
- Step 5: Mapping of Binary M:N Relationship Types.
- Step 6: Mapping of Multivalued attributes.
- Step 7: Mapping of N-ary Relationship Types.

Mapping EER Model Constructs to Relations

- Step 8: Options for Mapping Specialization or Generalization.
- Step 9: Mapping of Union Types (Categories).

Some DDL basics sprinkled in. Next: Full fledged SQL!