An Introduction to the Database Management Systems

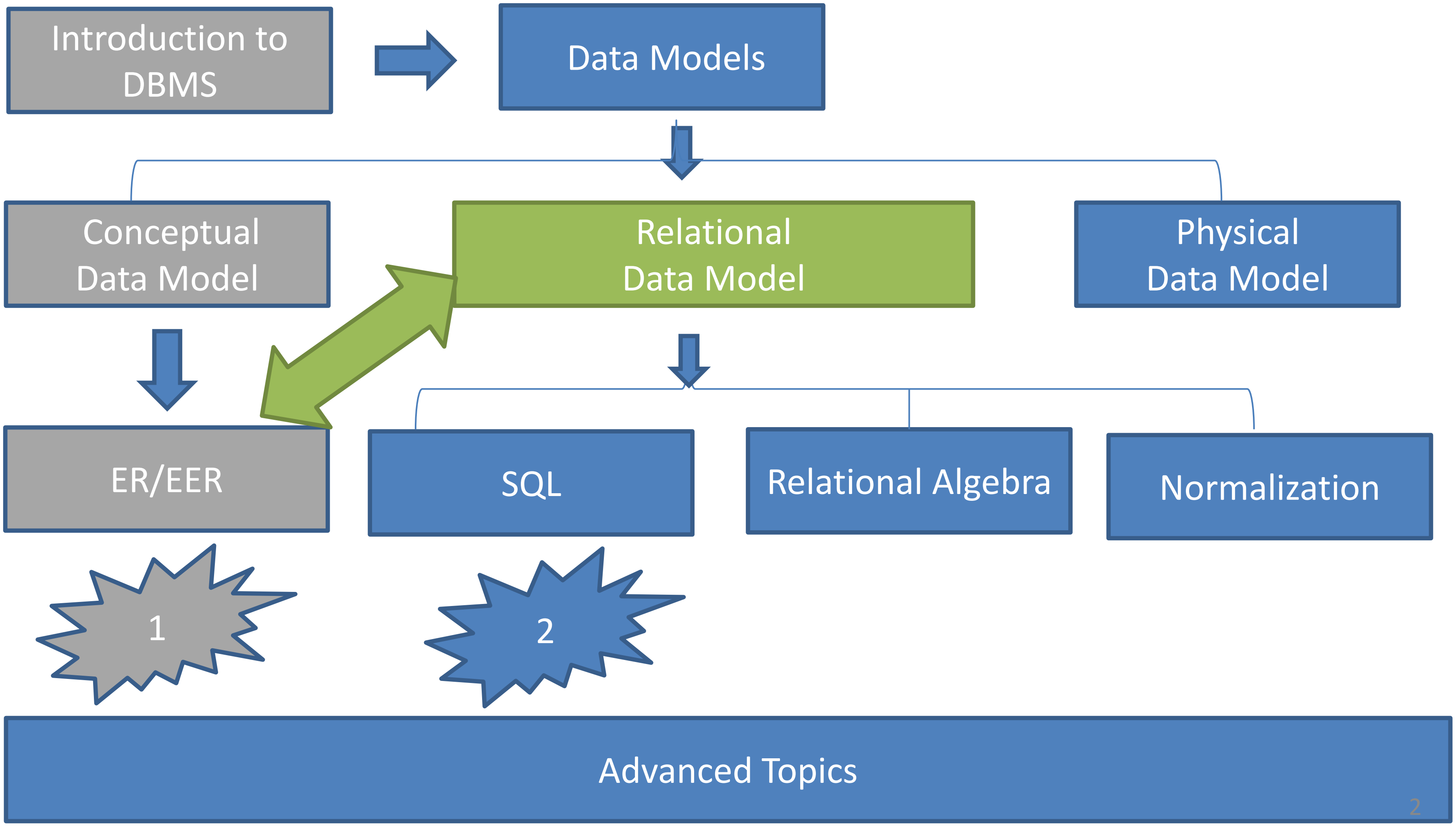
By

Hossein Rahmani

Slides originally by Book(s) Resources

Road Map

(Might change!)

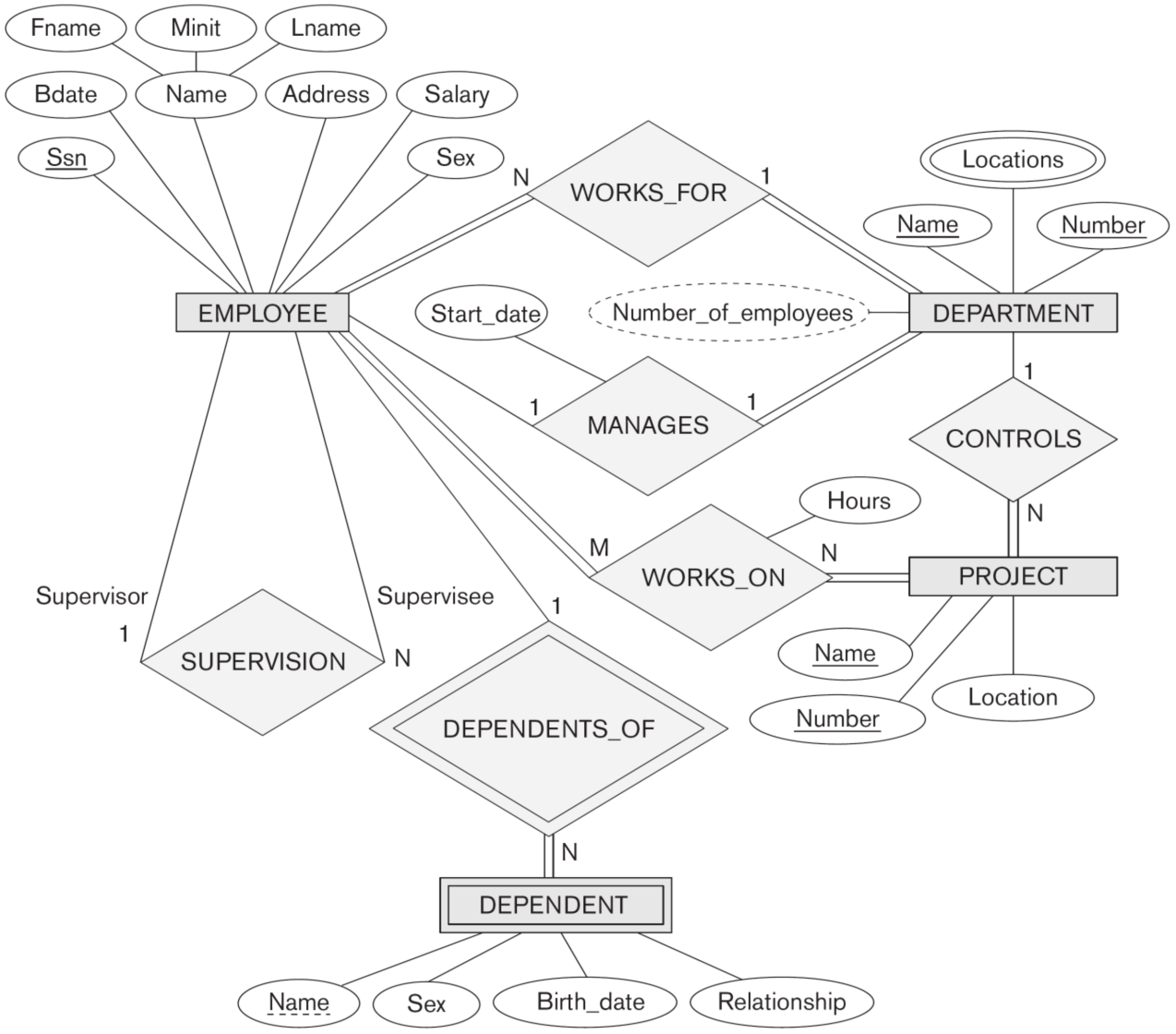


Logical design / “Data mapping”

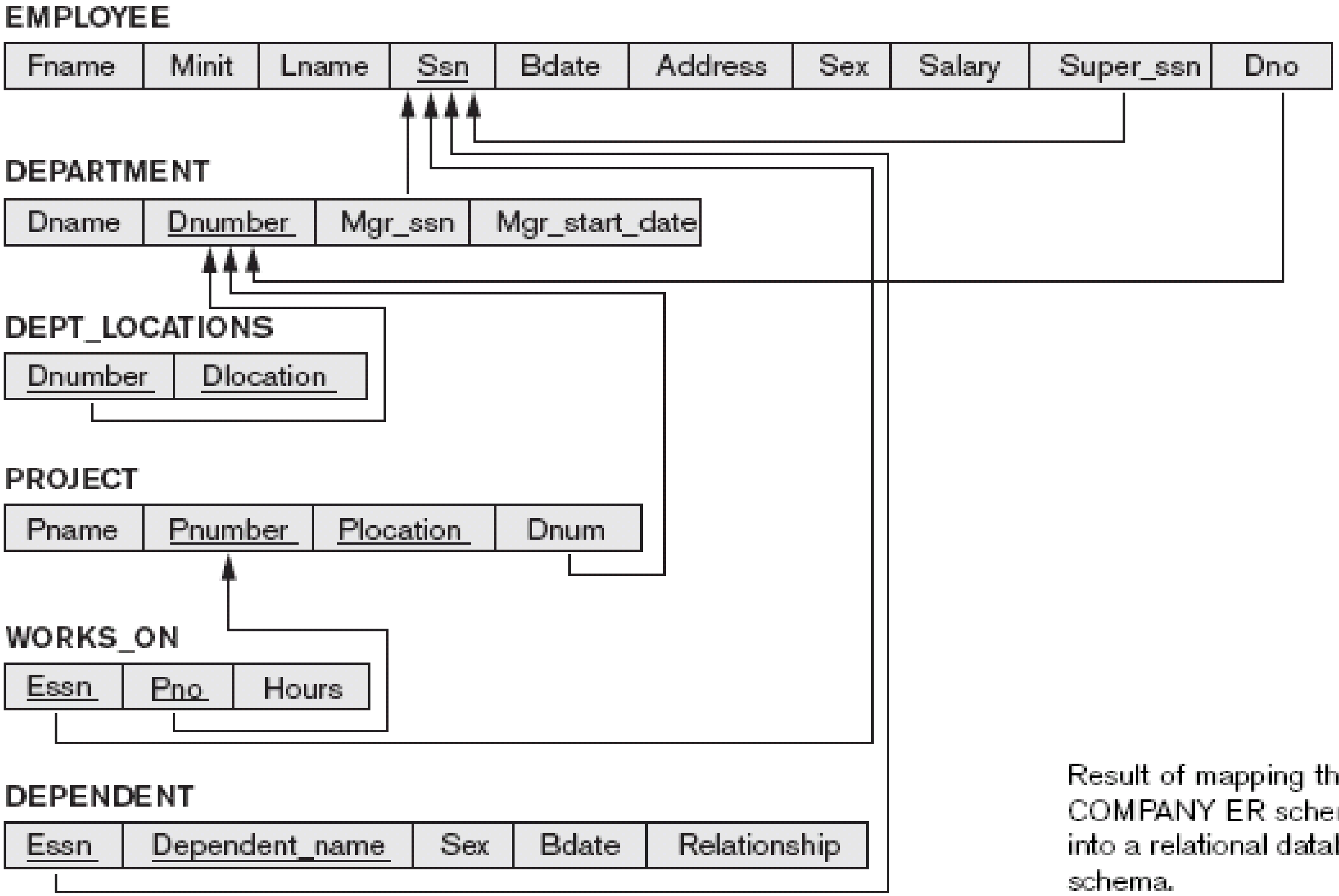
- We have to translate all constructs from the ER model into constructs of the relational model
- It starts with a cookbook recipe
- Then creativity and insight is required

Start from conceptual ER diagram

Figure 9.1
The ER conceptual schema diagram for the COMPANY database.



Finish at relational database 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

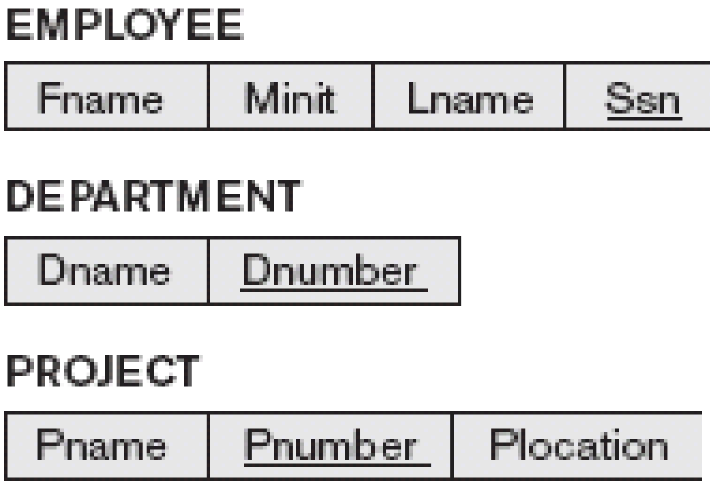
Essn	Dependent_name	Sex	Bdate	Relationship

Result of mapping the COMPANY ER schema into a relational database schema.

Step 1: Strong entity types

- Create a relation R for each strong entity type E
- Include all single-valued attributes and all single-valued parts of composite attributes
- Appoint one of the keys in E as primary key in R

Step 1: the example



EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary

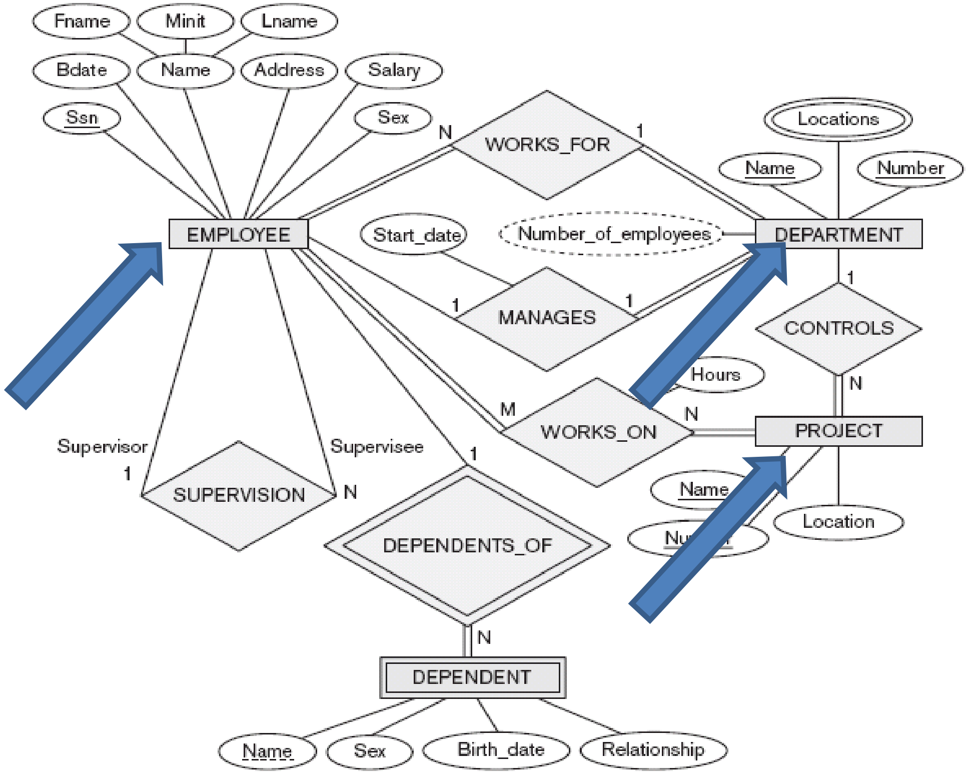
DEPARTMENT

Dname	Dnumber

PROJECT

Pname	Pnumber	Plocation

The ER conceptual schema diagram for the COMPANY database.



Step 2: Weak Entity types

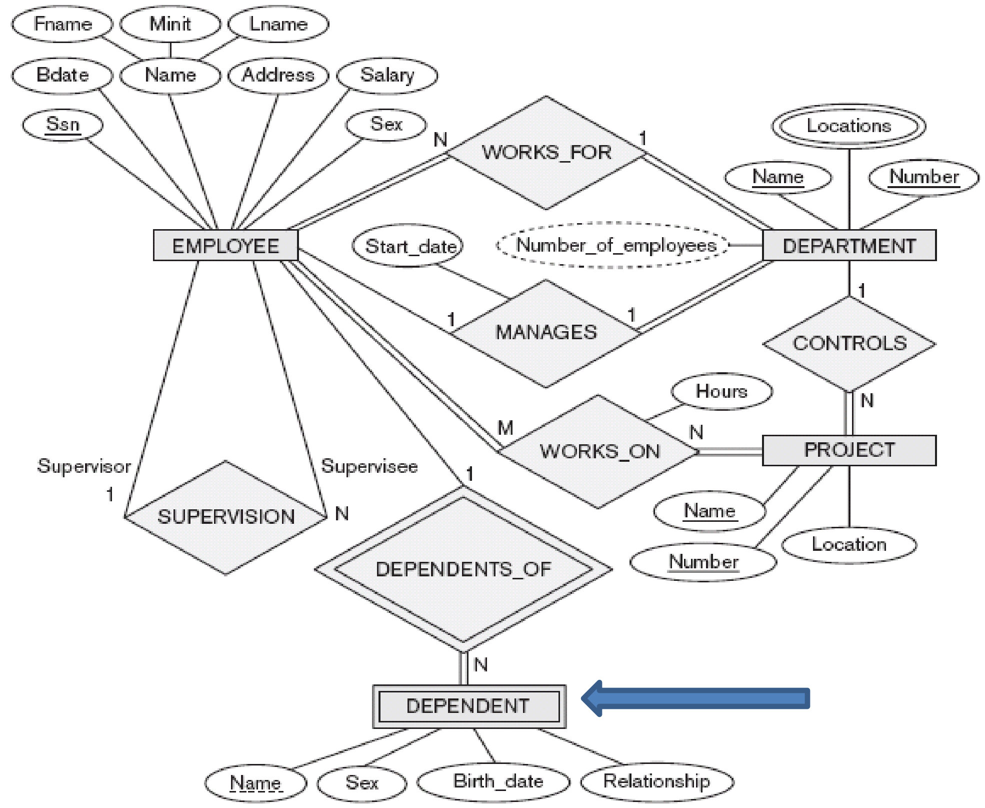
- Make a relation R for each weak entity type W, including the identifying relation
- Add all single-valued attributes and single-valued parts of composite attributes
- Add a foreign key FK to R pointing to the primary key of the identifying entity
- The primary key of R is the foreign key FK together with a partial key of W

Step 2: the example

DEPENDENT

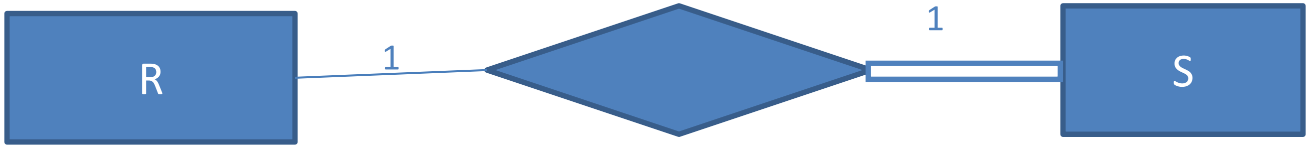
Essn	Dependent_name	Sex	Bdate	Relationship

The ER conceptual schema diagram for the COMPANY database.

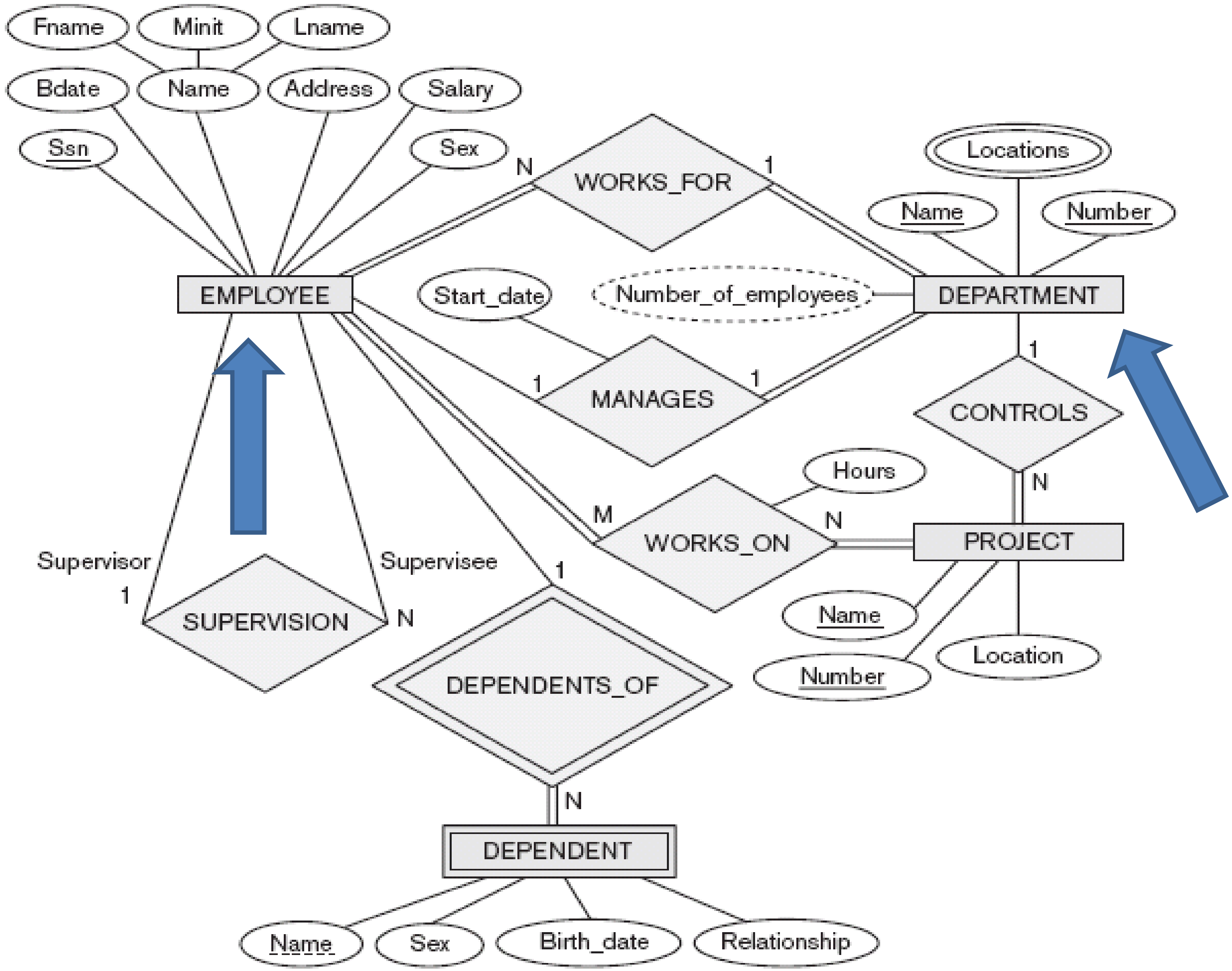


Step 3: 1:1 relations

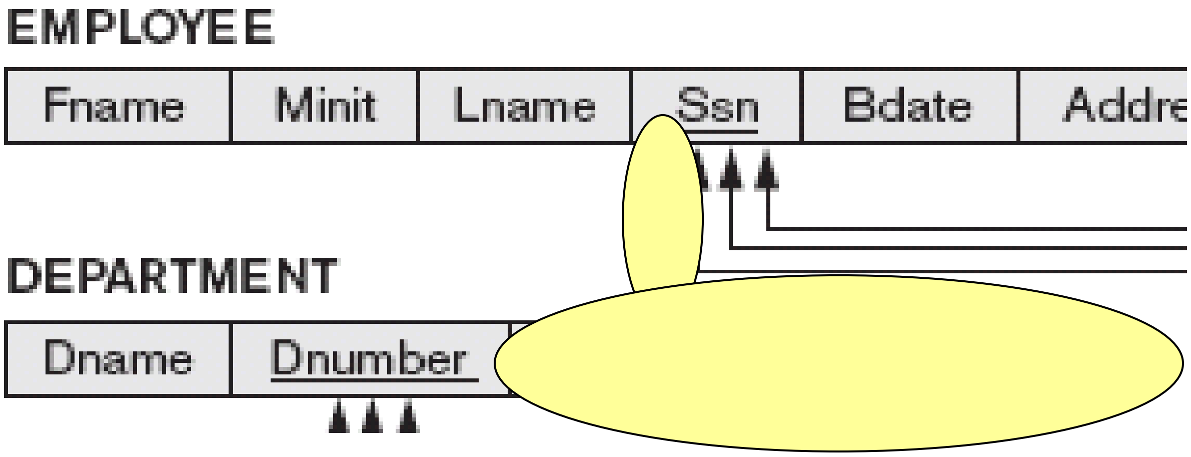
- Identify relations S and R that represent the connected entity types
- Add to one of them (say S, preferably totally participating):
  - A foreign key to R
  - all (single-valued) attributes of the 1-1 relation



The ER conceptual schema diagram for the COMPANY database.



Step 3: the example



EMPLOYEE

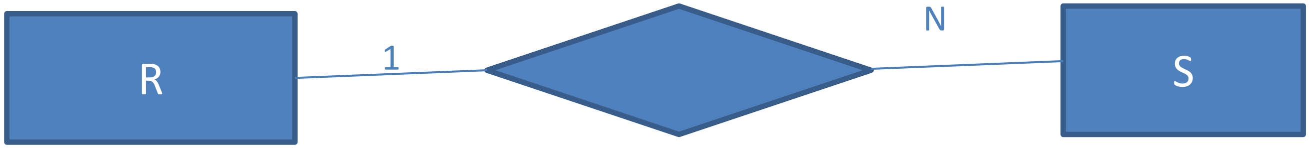
Fname	Minit	Lname	Ssn	Bdate	Addre

DEPARTMENT

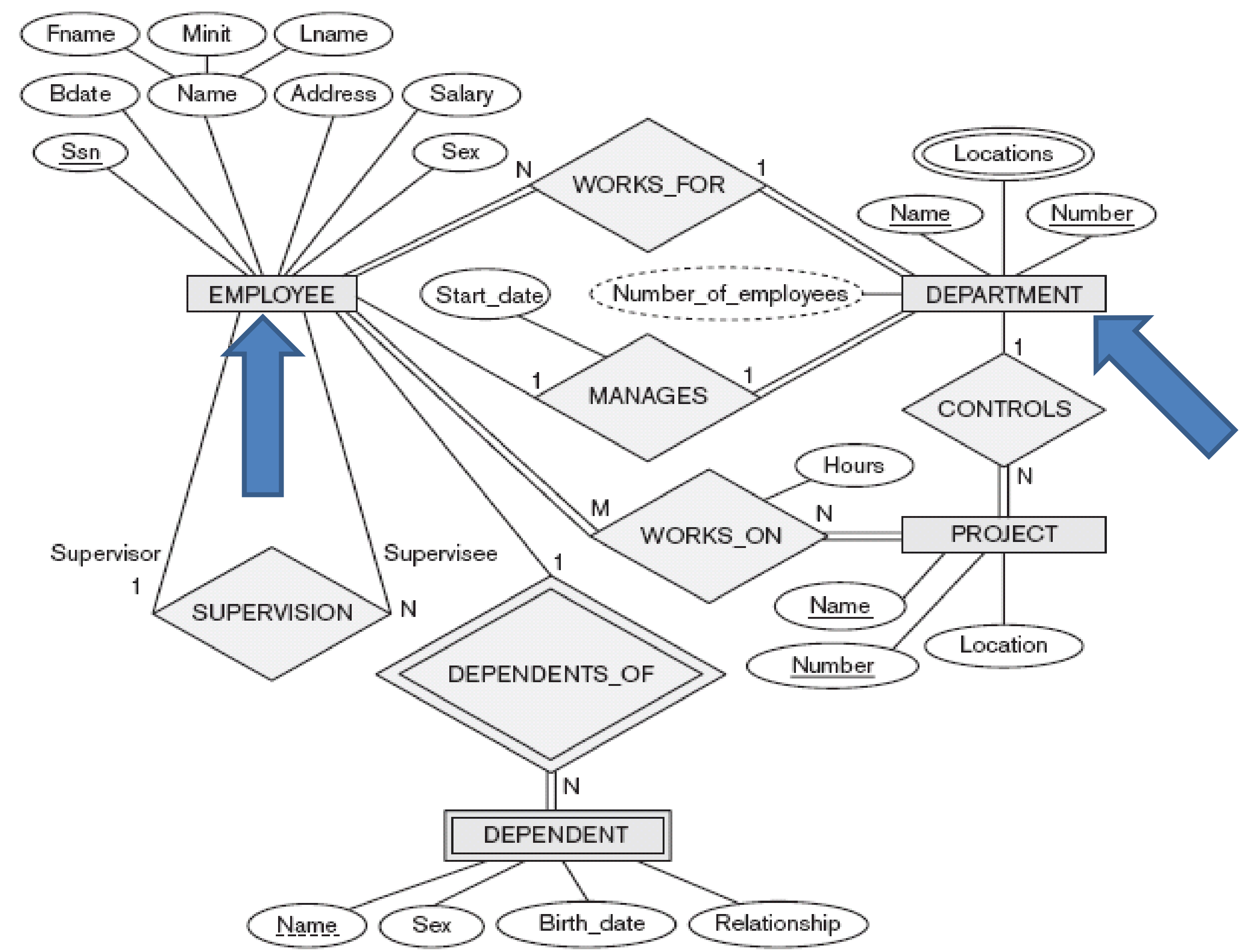
Dname	Dnumber

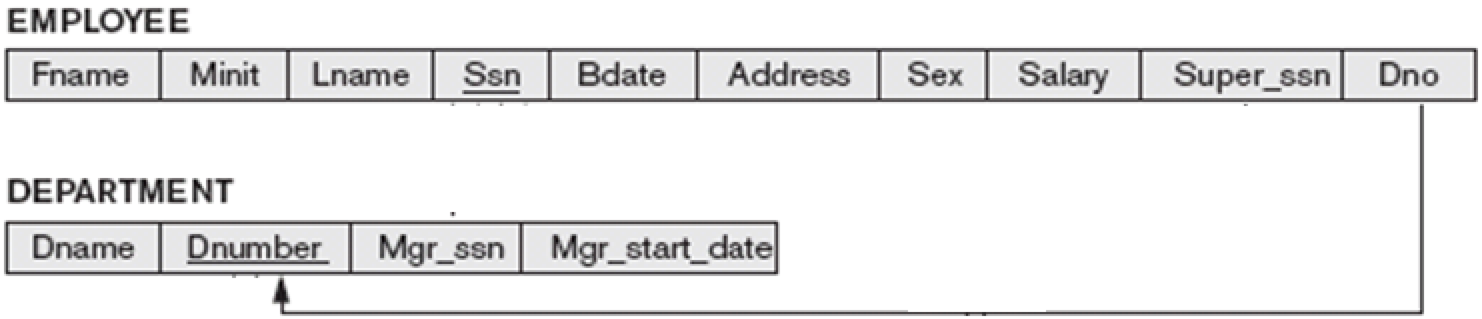
Step 4: 1:N Relations

- Identify S and R that represent the connected entity types, having S at the N-side
- Add a foreign key in S, pointing to R
- Add all (single-valued) attributes of the relation to S



The ER conceptual schema diagram for the COMPANY database.





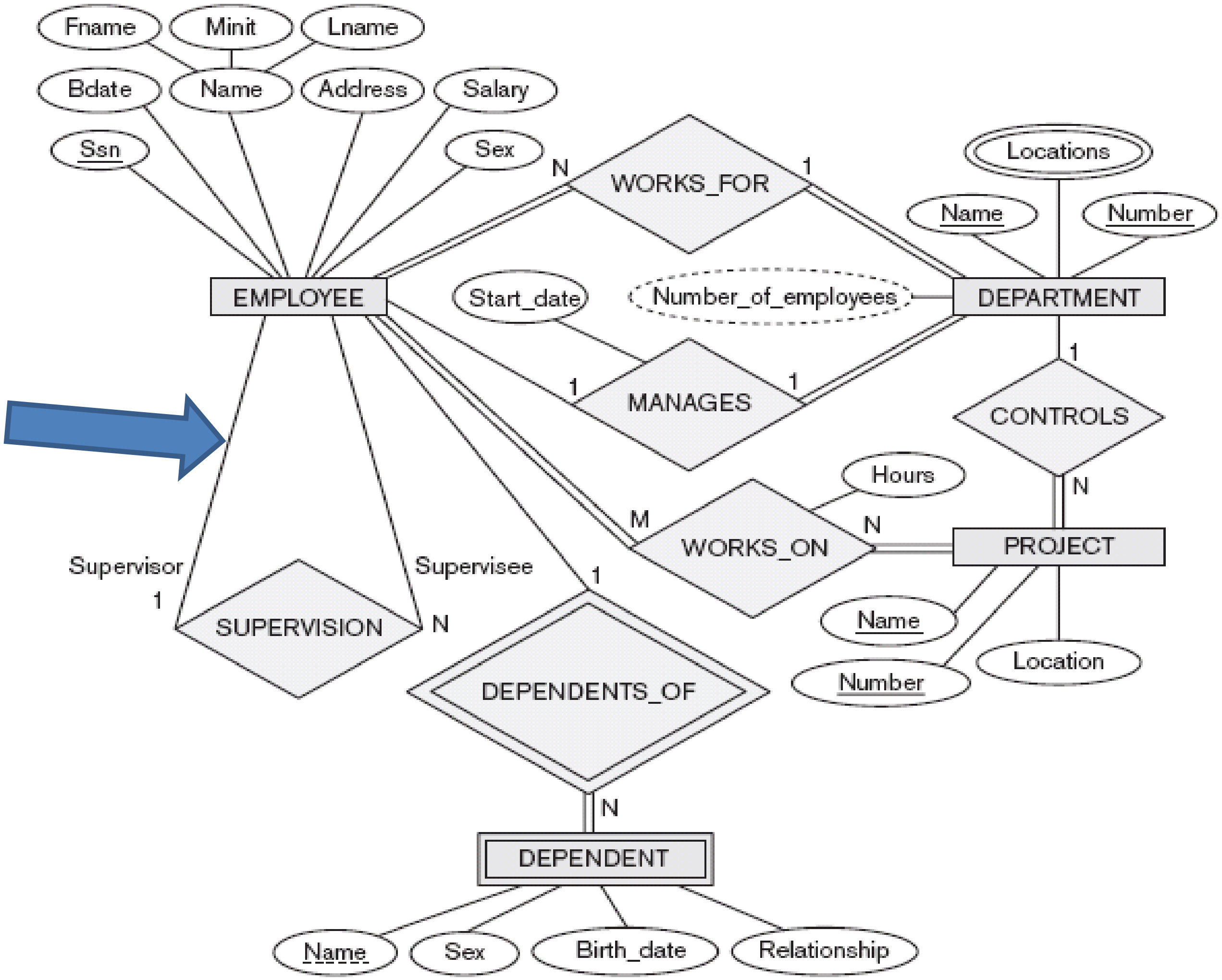
EMPLOYEE

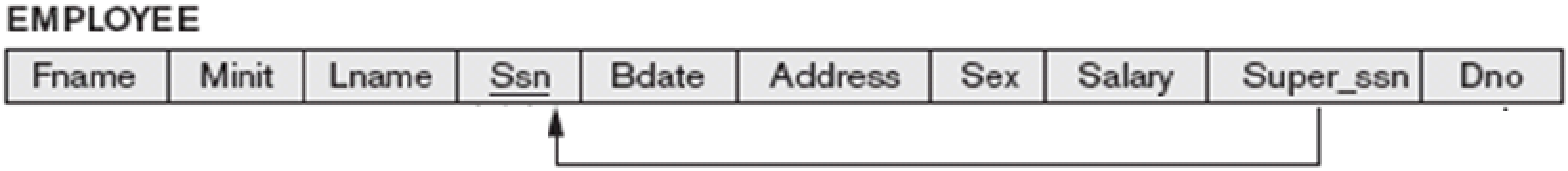
Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

The ER conceptual schema diagram for the COMPANY database.

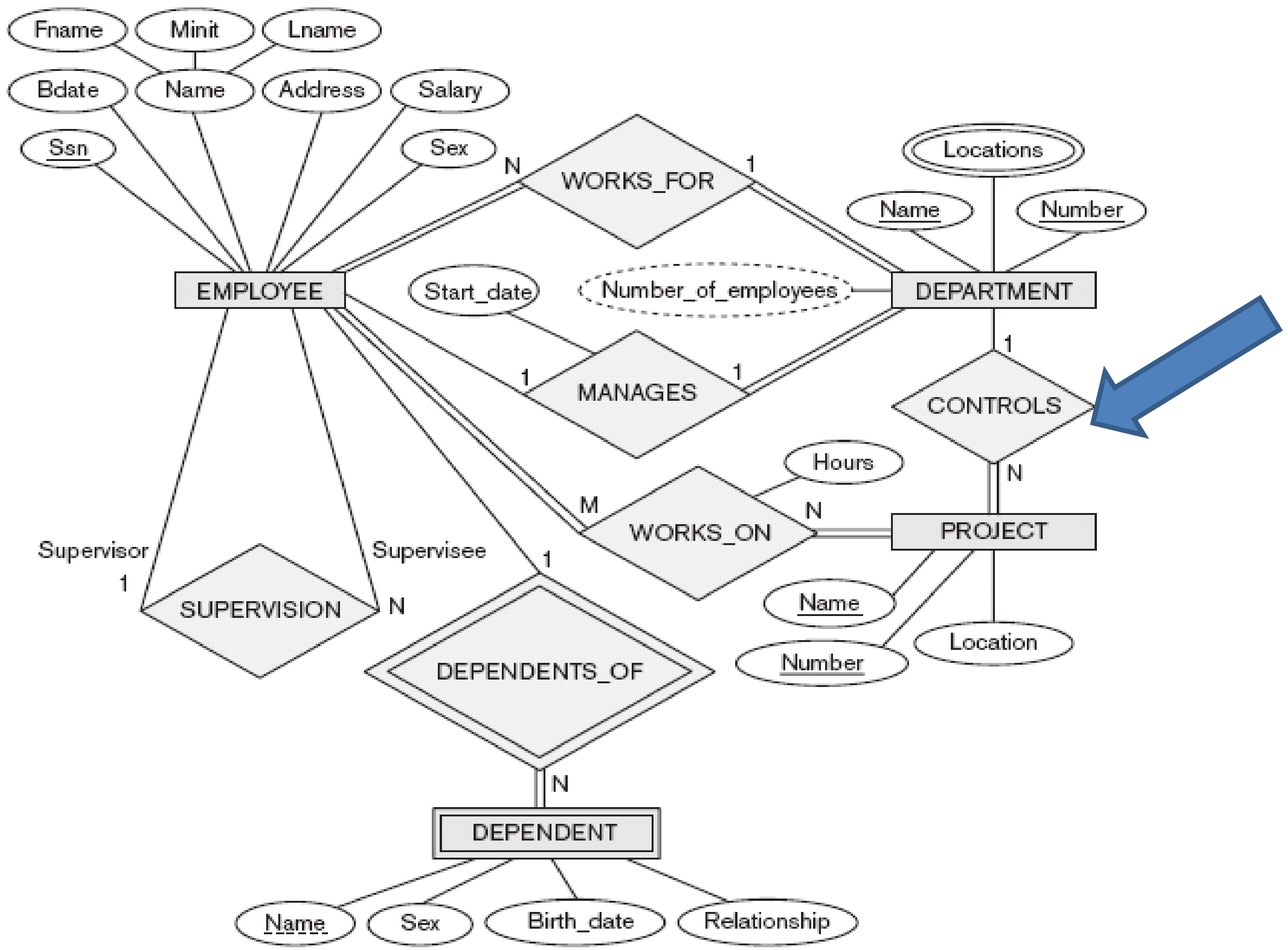


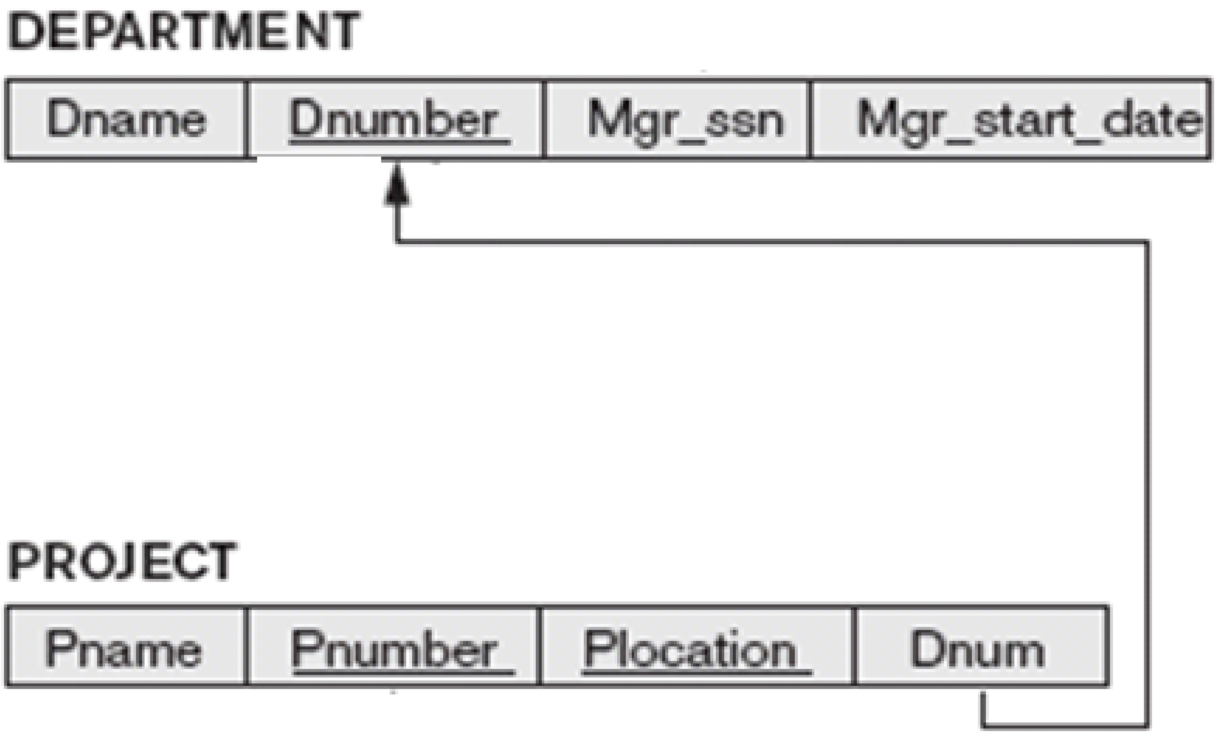


EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

The ER conceptual schema diagram for the COMPANY database.





DEPARTMENT

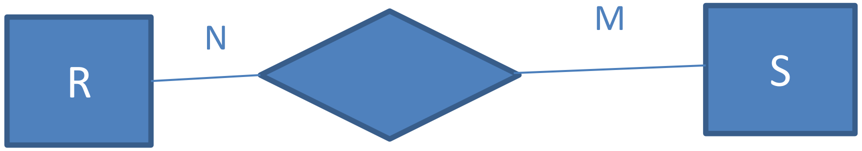
Dname	Dnumber	Mgr_ssn	Mgr_start_date

PROJECT

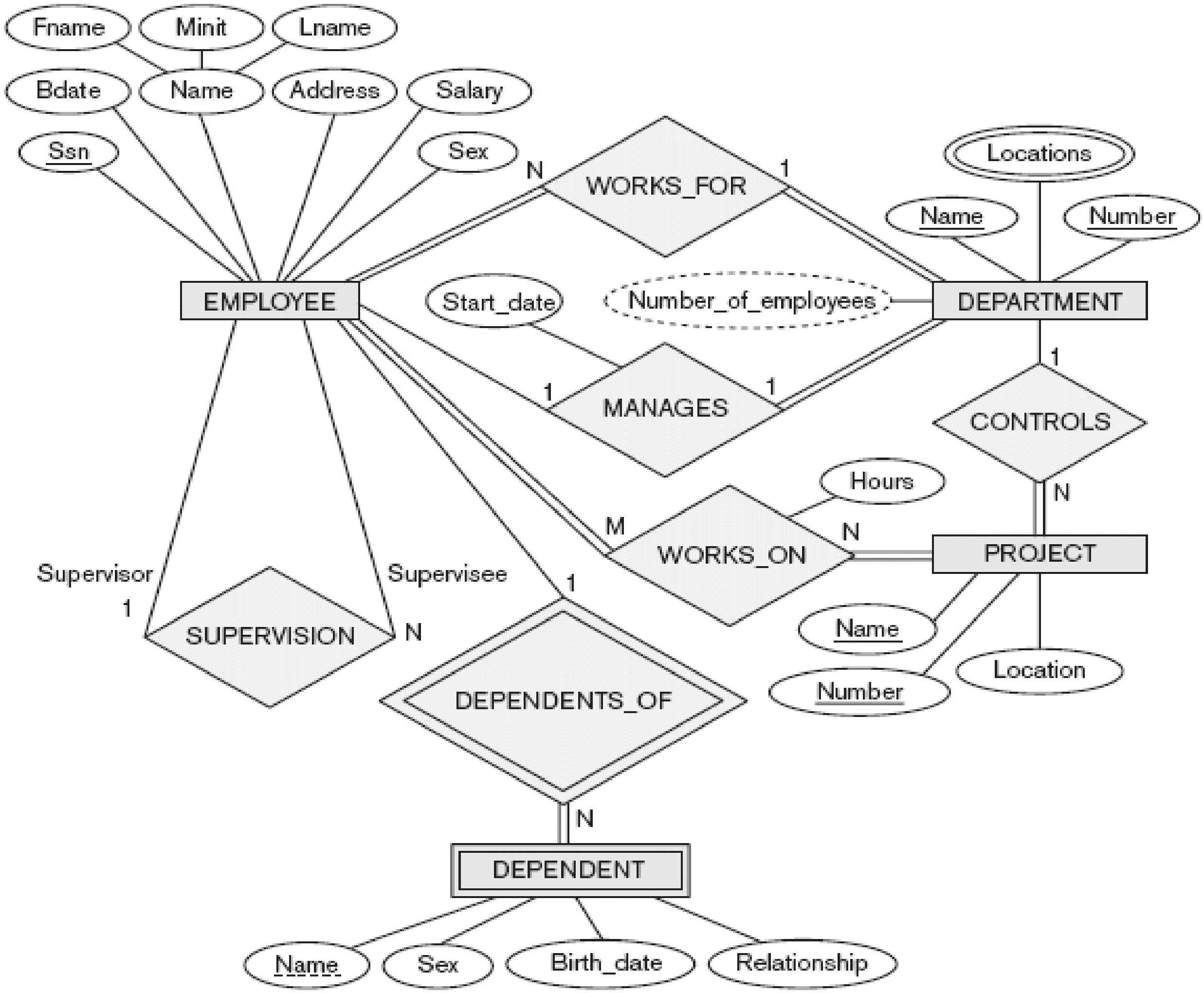
Pname	Pnumber	Plocation	Dnum

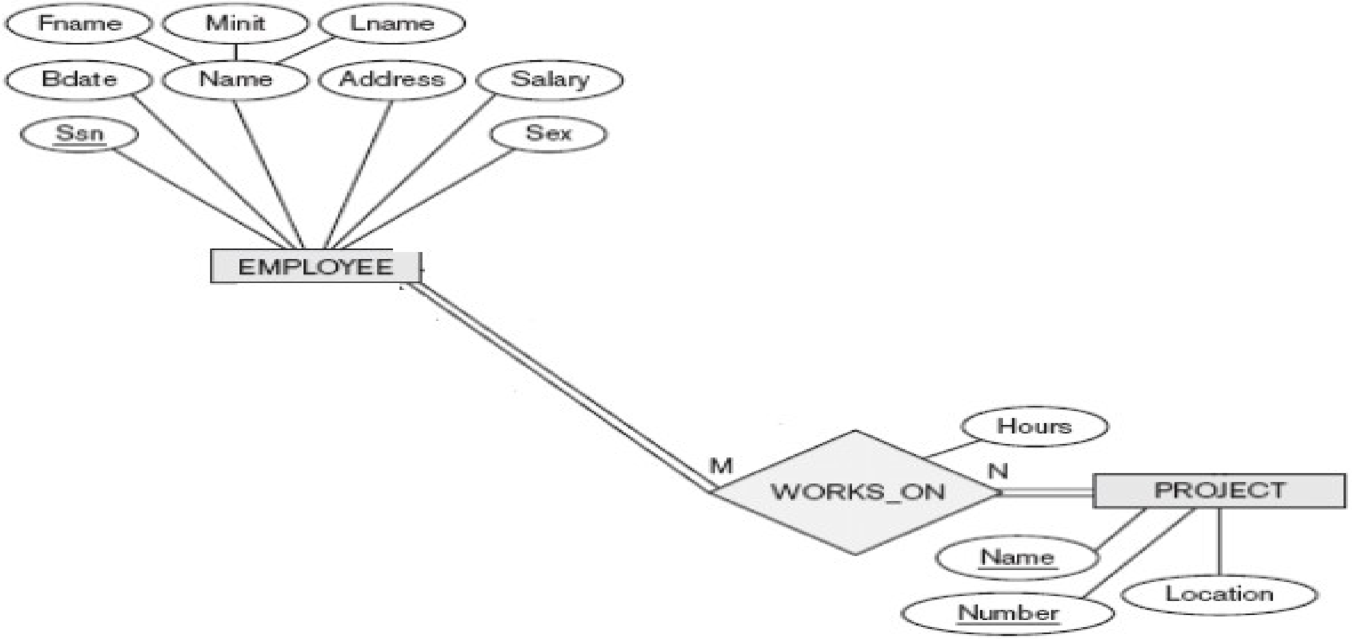
Step 5: M:N Relations

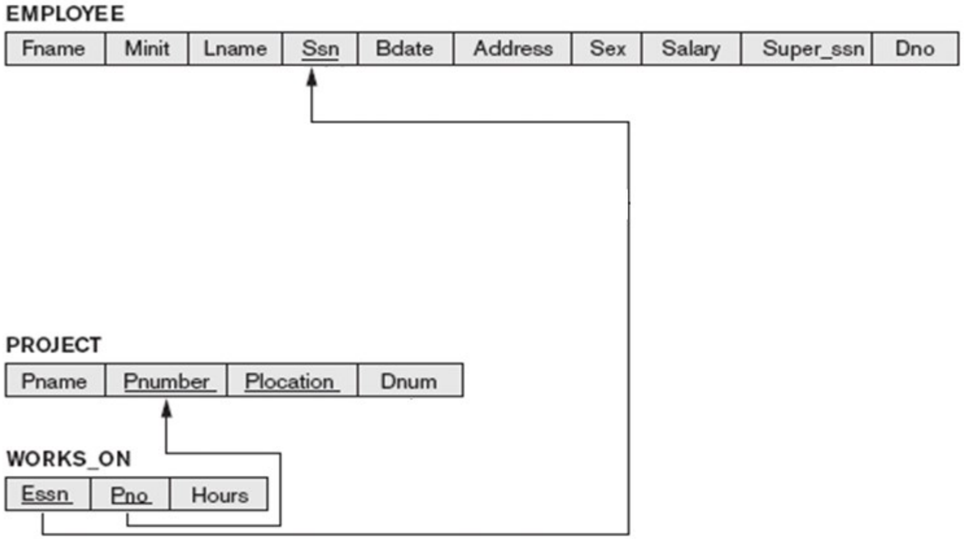
- Identify S and T that represent the connected entity types
- Make a new relation R
- Include:
  - Foreign keys to both entities S and T
  - all (single-valued) attributes
  - as primary key: The combination of the two foreign keys



The ER conceptual schema diagram for the COMPANY database.







EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno

PROJECT

Pname	Pnumber	Plocation	Dnum

WORKS_ON

Essn	Pno	Hours

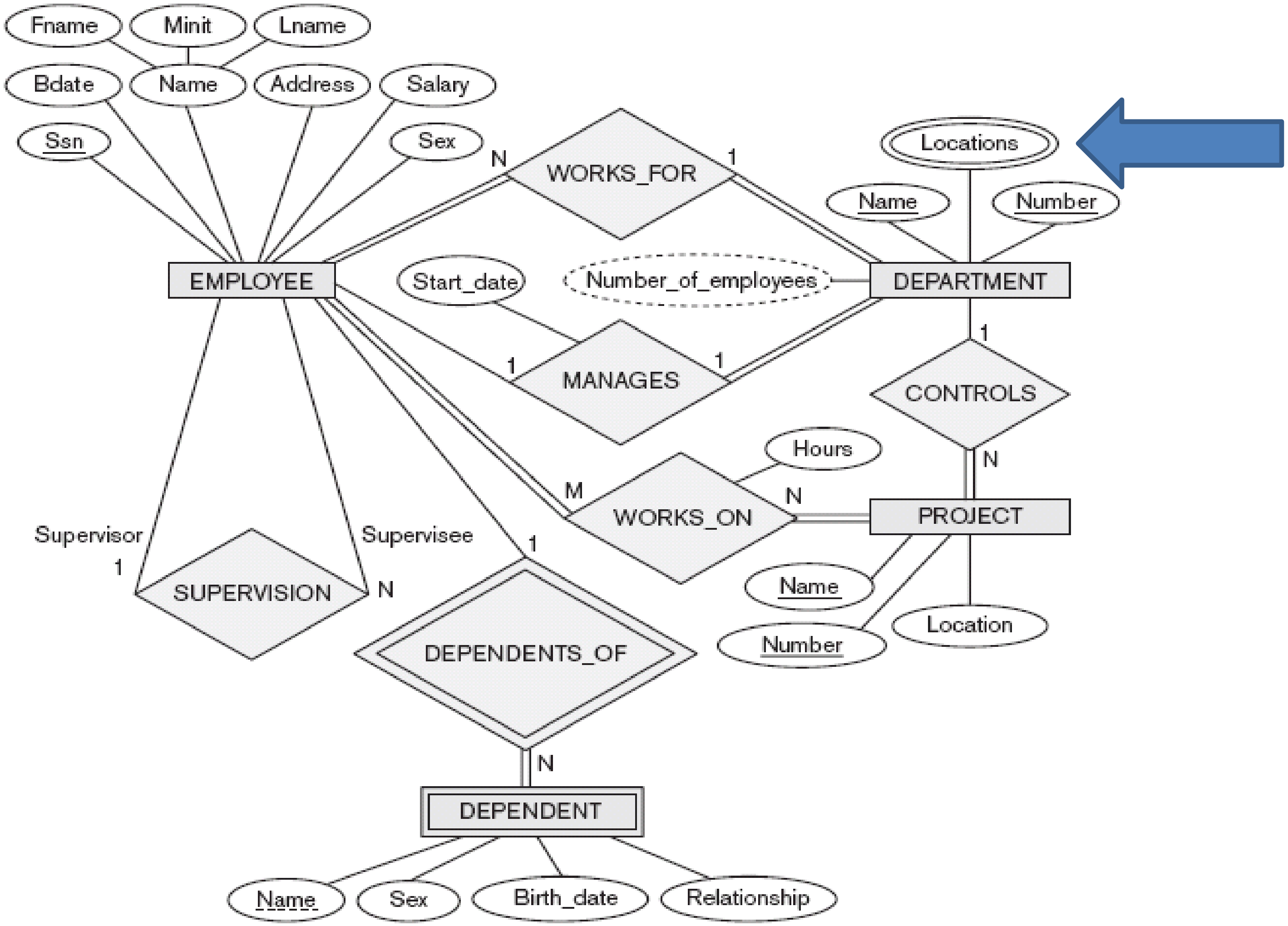
Note

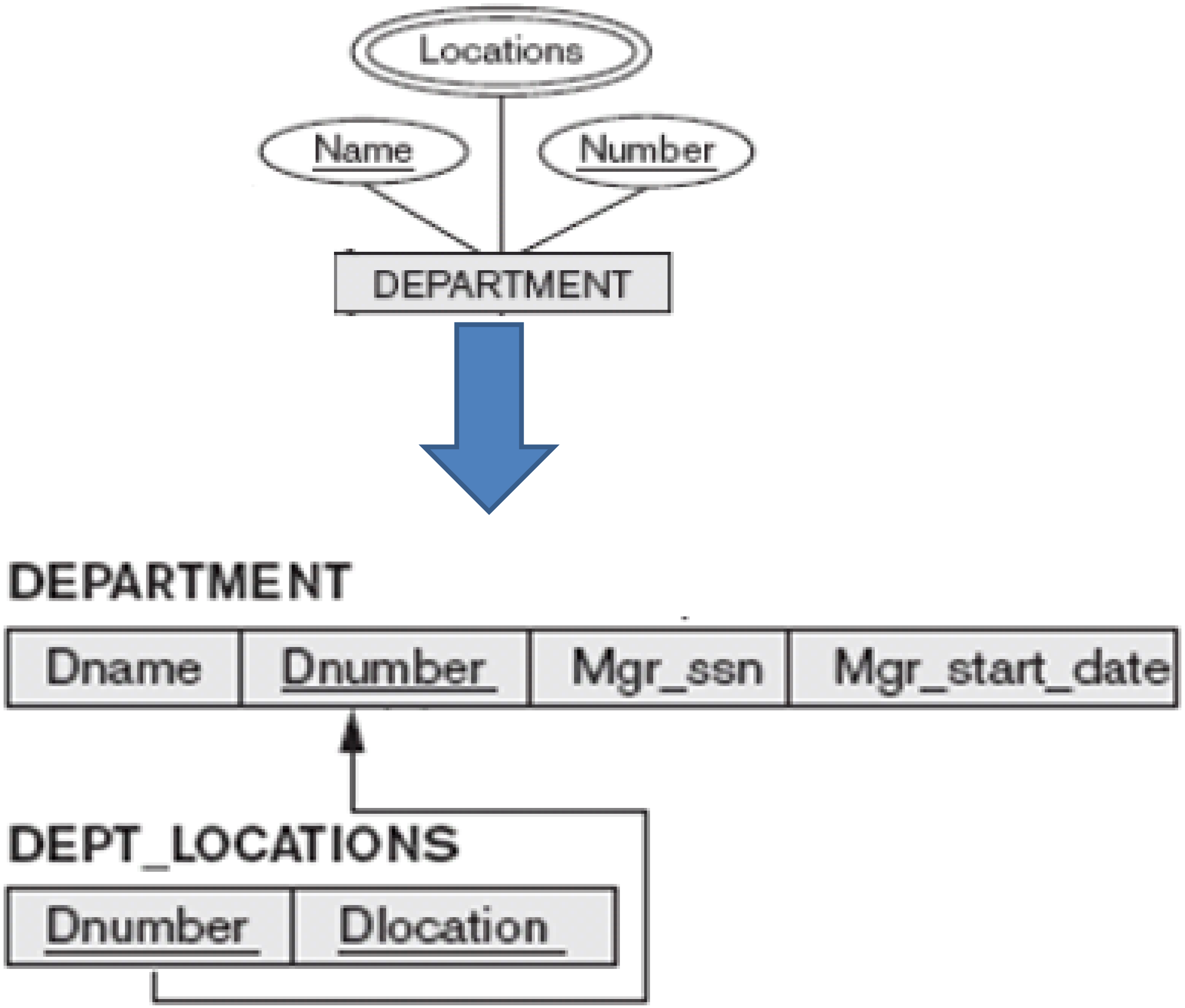
- Note that step 5 **can** be used to model 1:1 and 1:N relations also.
- In that case also define a new relation R for the 1:1 or 1:N relation from S to T
- This is, however, never needed and only useful in rare cases (e.g., when only very few instances of T are related to S, to avoid many NULL values)

Step 6: multivalued attributes

- Make a relation R for each multivalued attribute A of entity E or relation S.
- Imagine K as primary key of S
- Add to R a foreign key to K
- The primary key of R is the combination of all attributes in R

The ER conceptual schema diagram for the COMPANY database.





DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date

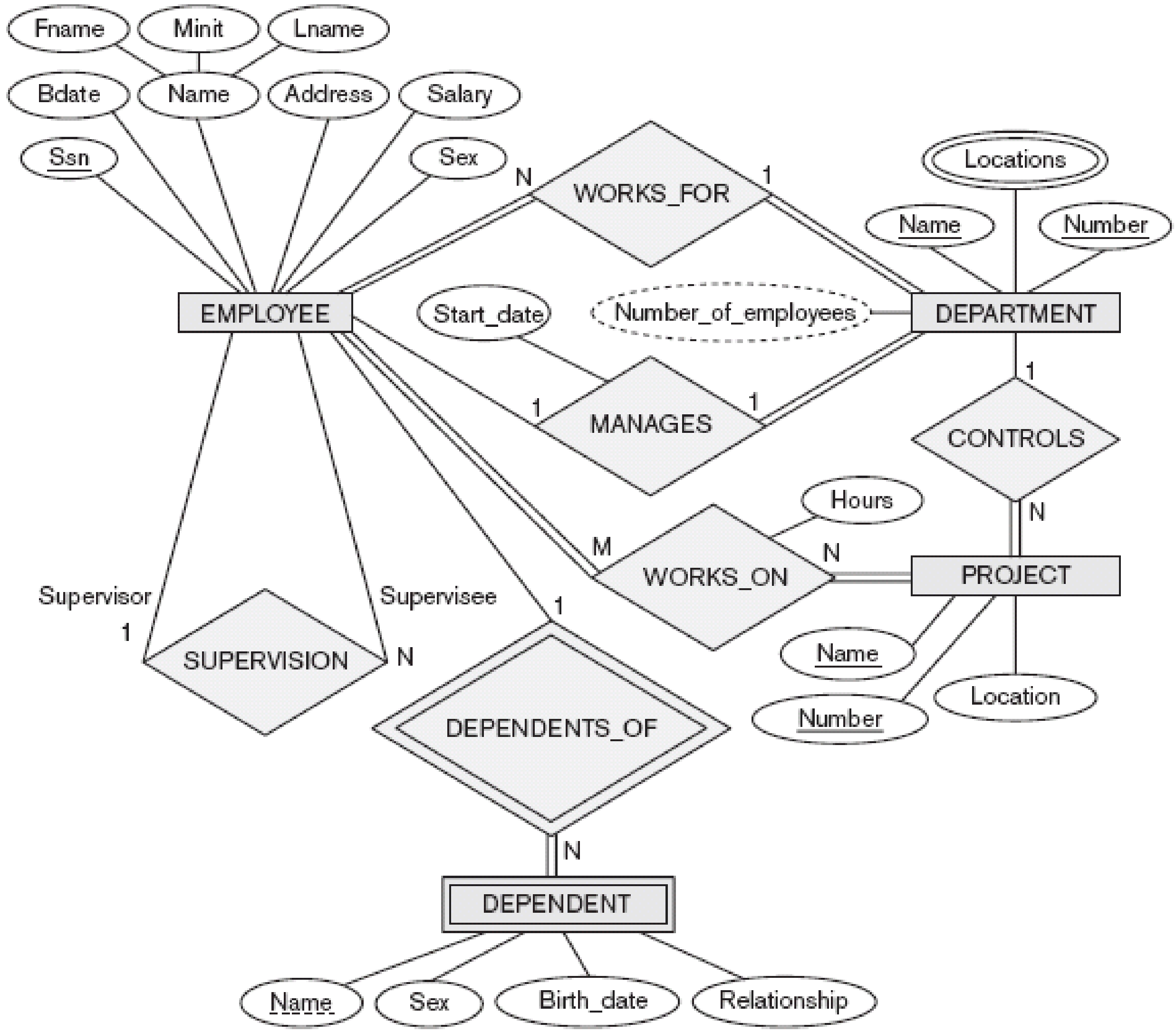
DEPT_LOCATIONS

Dnumber	Dlocation

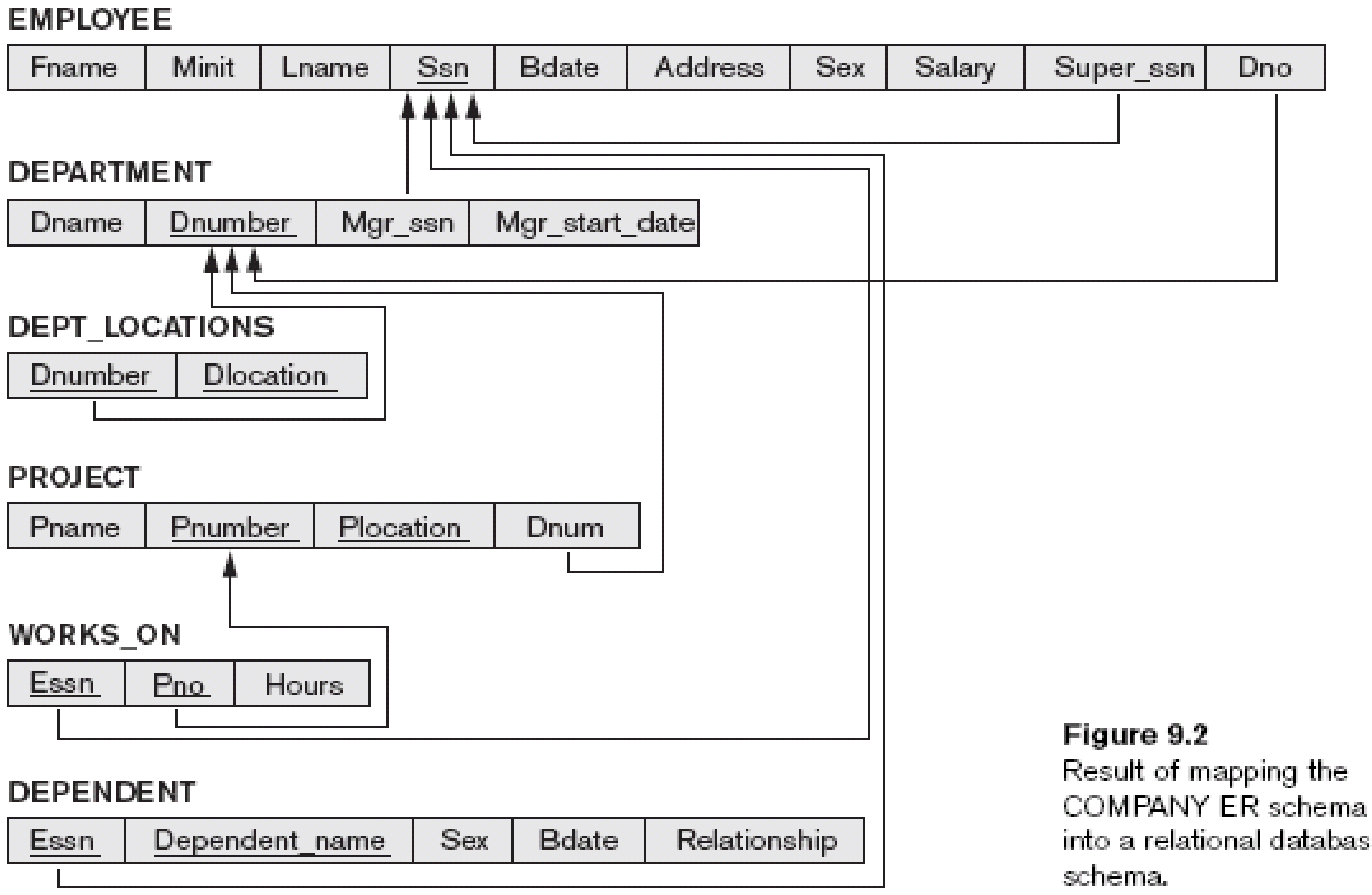
Step 7: Relation types of degree > 2

- Create a new relation R
- Add foreign keys for all participations
- Add single-valued attributes
- Primary key is the combination of foreign keys

The ER conceptual schema diagram for the COMPANY database.



Final relational database 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

Figure 9.2
Result of mapping the COMPANY ER schema into a relational database schema.

Entity types in ER diagrams

- Until now, each entity was assigned to one entity type
- In many cases an entity type has numerous subgroupings or subtypes of its entities that are meaningful and need to be represented explicitly
- Members of the EMPLOYEE entity type may be distinguished further into SECRETARY, ENGINEER, MANAGER, etc

Subclasses and Superclasses

- Each of these subgroupings is called a **subclass** or **subtype** of the EMPLOYEE entity type, and the EMPLOYEE entity type is called the **superclass** or **supertype**
- Each subclass member is the same as the entity in the superclass, but in a distinct *specific role*

EER to Relational Model

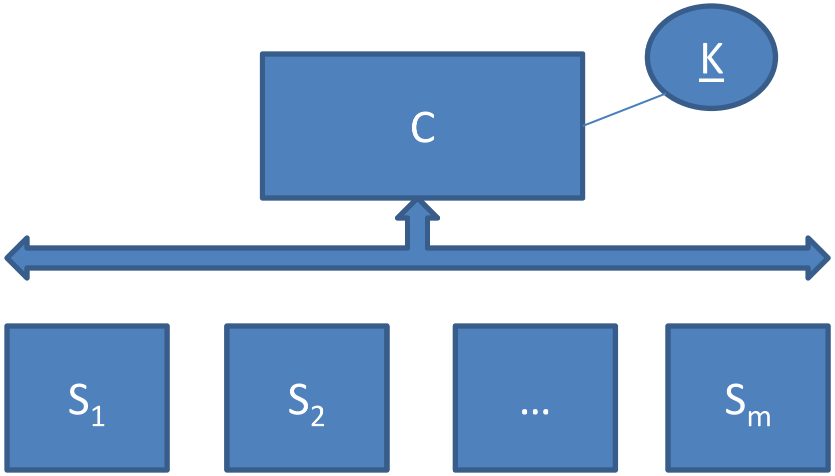
- There are several ways in which sub/superclasses can be translated into relations (see next slides)
- Inheritance of attributes and relations is not automatic and has to be programmed explicitly in a relational DBMS with SQL
- We need a new type of DBMS!

Mapping EER to relation schema

- We extend our 7-step algorithm for transforming an ER diagram into a relation schema with 2 additional steps
- Step 8: mapping of generalizations / specializations (4 options: 8A-8D)
- Step 9: mapping of categories

Step 8: generalization / specialization

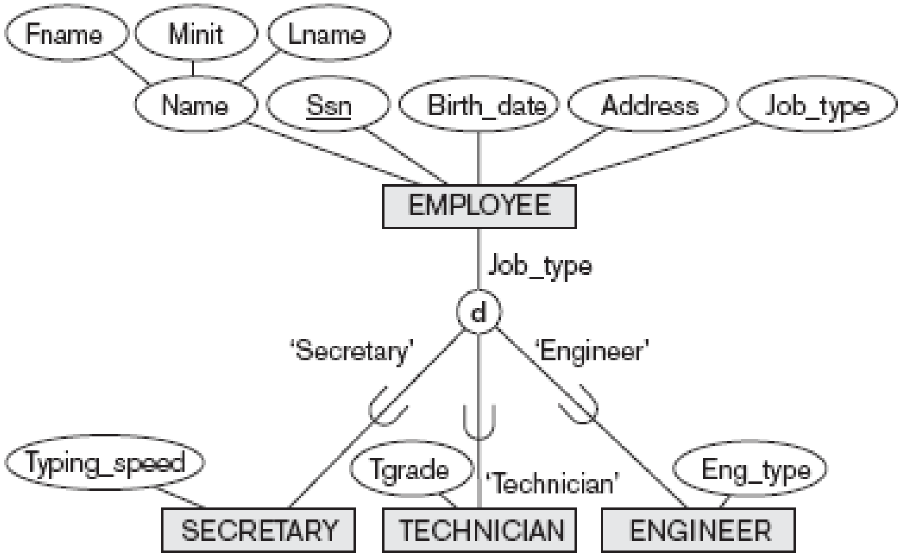
- Convert each specialization with *m* subclasses $\{S_1, \ldots, S_m\}$ and superclass *C*, with attributes $\{k, a_1, \ldots, a_n\}$, *k* being the primary key, into a relation scheme, using 1 of 4 options

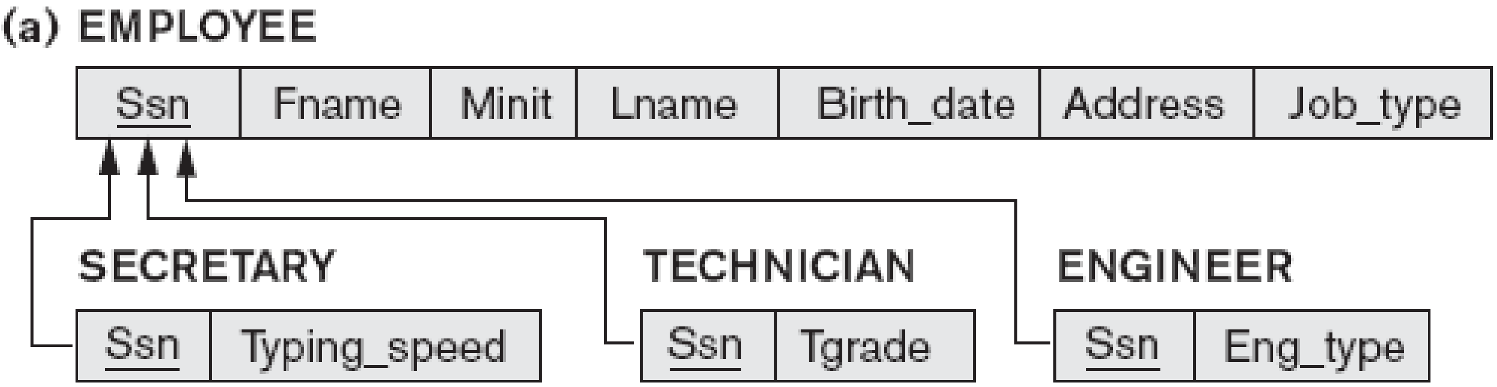


Step 8A: multiple relations – superclass and subclasses

- Create a relation L for C with attributes {k, a_1, …, a_n}, and PK(L) = k; create a relation L_i for every subclass S_i, with attributes {k} U {attributes of S_i}, and PK(L_i) = k
- Works for **any** specialization (total, partial, overlapping, disjoint)
- Example: next slide

Step 8A: example





(a) EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type

SECRETARY

Ssn	Typing_speed

TECHNICIAN

Ssn	Tgrade

ENGINEER

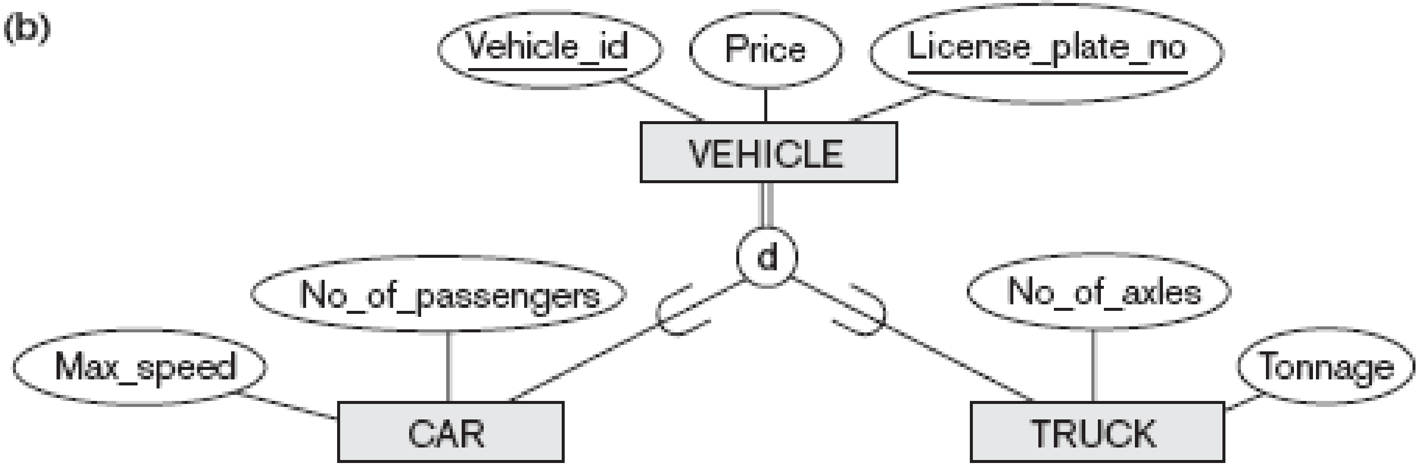
Ssn	Eng_type

Step 8B: multiple relations – subclasses only

- Like 8A, but don't create a relation for *C*, only for the subclasses S_i, with attributes $\{k, a_1, \ldots, a_n\} \cup$ {attributes of S_i} and PK(L_i) = k
- Only for **total** specialization (otherwise, entries may "get lost"); recommended to be ***disjoint*** (otherwise duplications)
- Example: next slide

Step 8B: example

(b)



(b) CAR

Vehicle_id	License_plate_no	Price	Max_speed	No_of_passengers

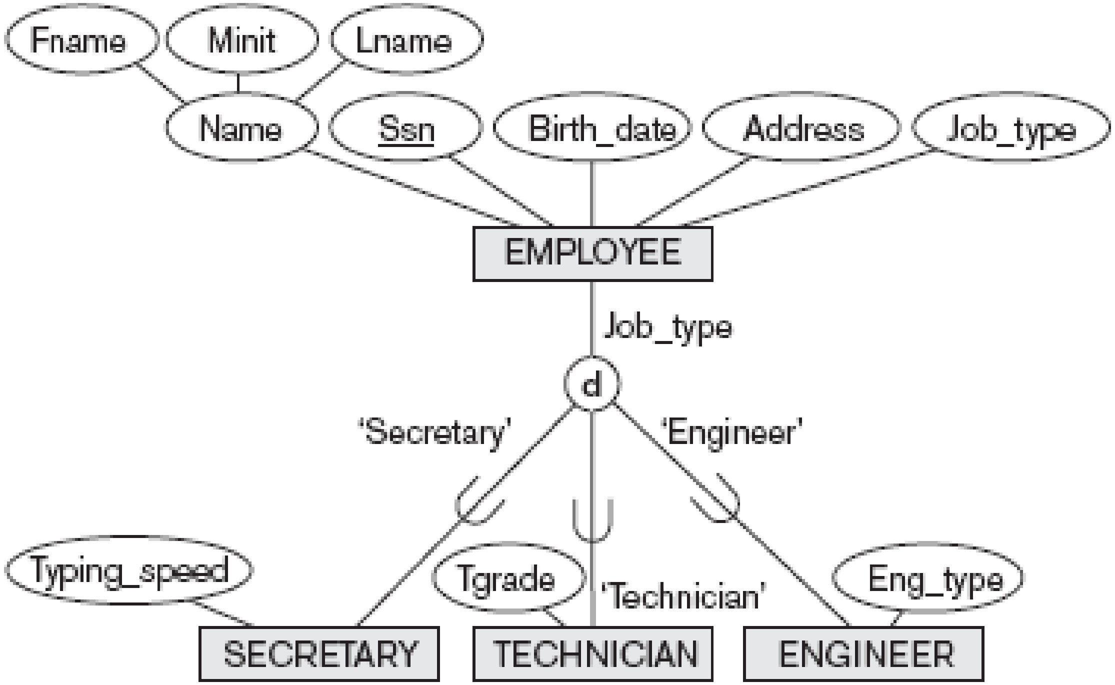
TRUCK

Vehicle_id	License_plate_no	Price	No_of_axles	Tonnage

Step 8C: single relation with 1 type attribute

- Single relation L with attributes $\{k, a_1, \ldots, a_n\}$ U {attributes of S_1} U … U {attributes of S_m} U $\{t\}$, t being a *type* (or *discriminating*) attribute, and PK(L) = k
- The type attribute is only needed if it is not already contained in the superclass
- Only for ***disjoint*** specialization; possibly many NULL values
- Example: next slide

Step 8C: example



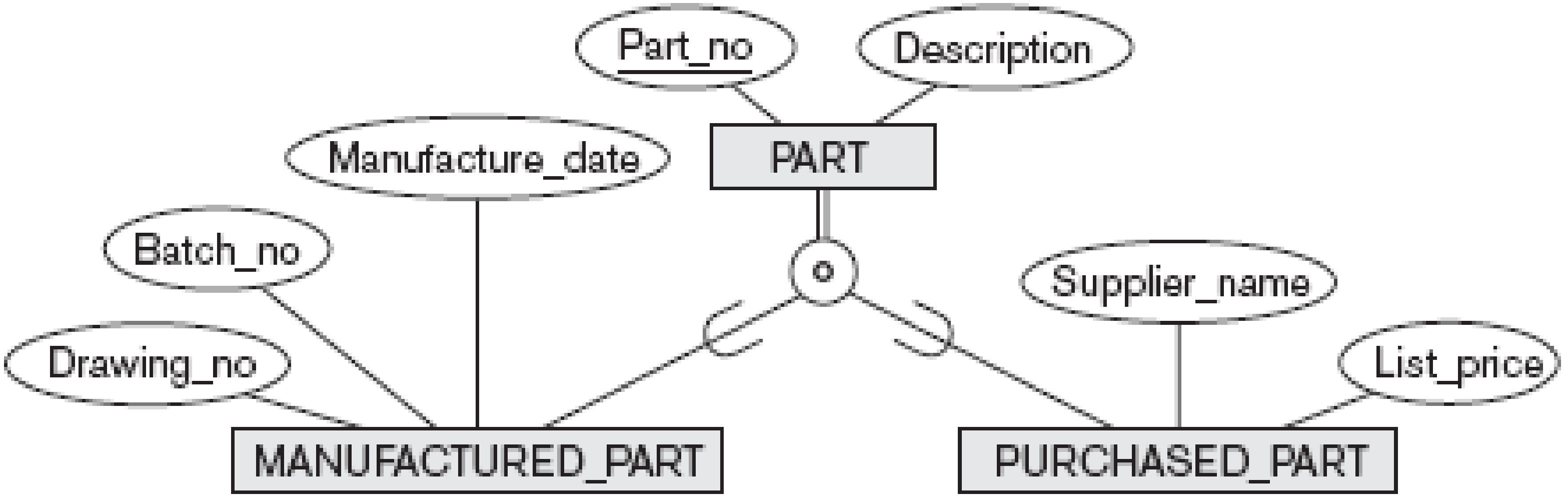
(c) EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type	Typing_speed	Tgrade	Eng_type

Step 8D: single relation with multiple type attributes

- Single relation L with attributes $\{k, a_1, \ldots, a_n\}$ U {attributes of S_1} U … U {attributes of S_m} U $\{t_1, \ldots, t_m\}$, t_i being Boolean *type* (or *discriminating*) attributes, and PK(L) = k
- Useful for **overlapping** specializations (but works also for ***disjoint*)**;
- Example: next slide

Step 8D: example



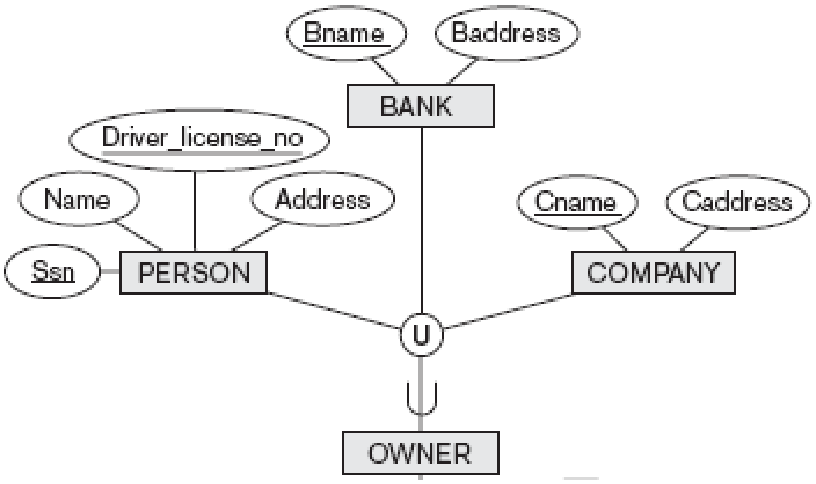
(d) PART

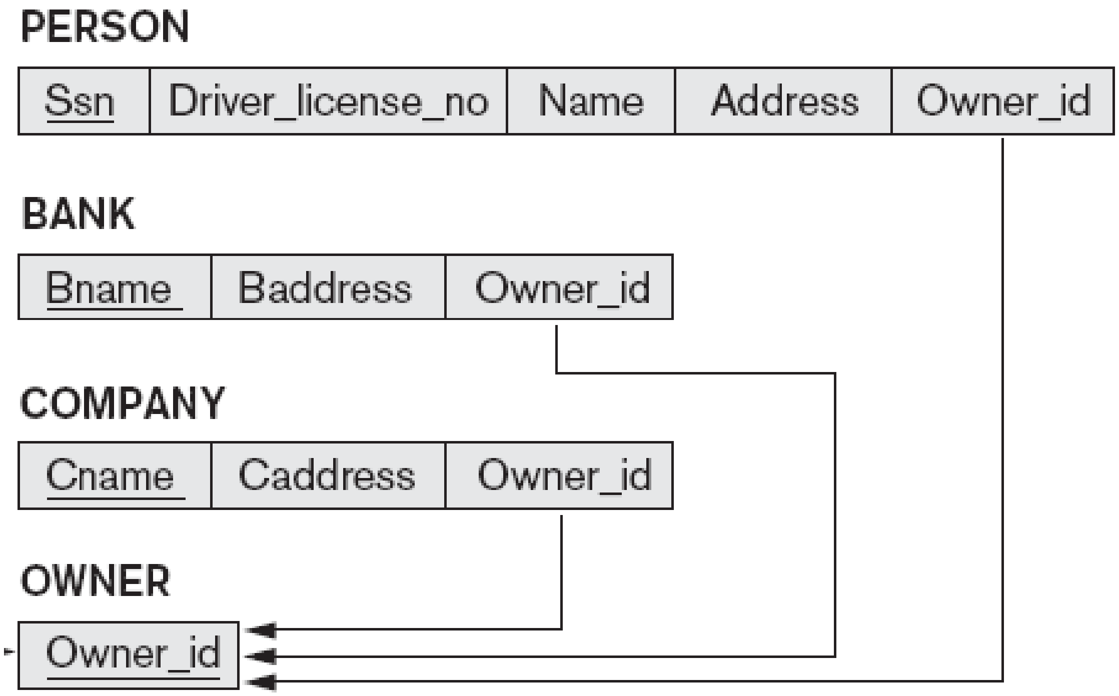
Part_no	Description	Mflag	Drawing_no	Manufacture_date	Batch_no	Pflag	Supplier_name	List_price

Step 9: mapping of union types (categories)

- Generate a special relation with only 1 attribute, the so-called *surrogate* key, for the category, and separate relations for all superclasses.
- Example: next slide

Step 9: example





PERSON

Ssn	Driver_license_no	Name	Address	Owner_id

BANK

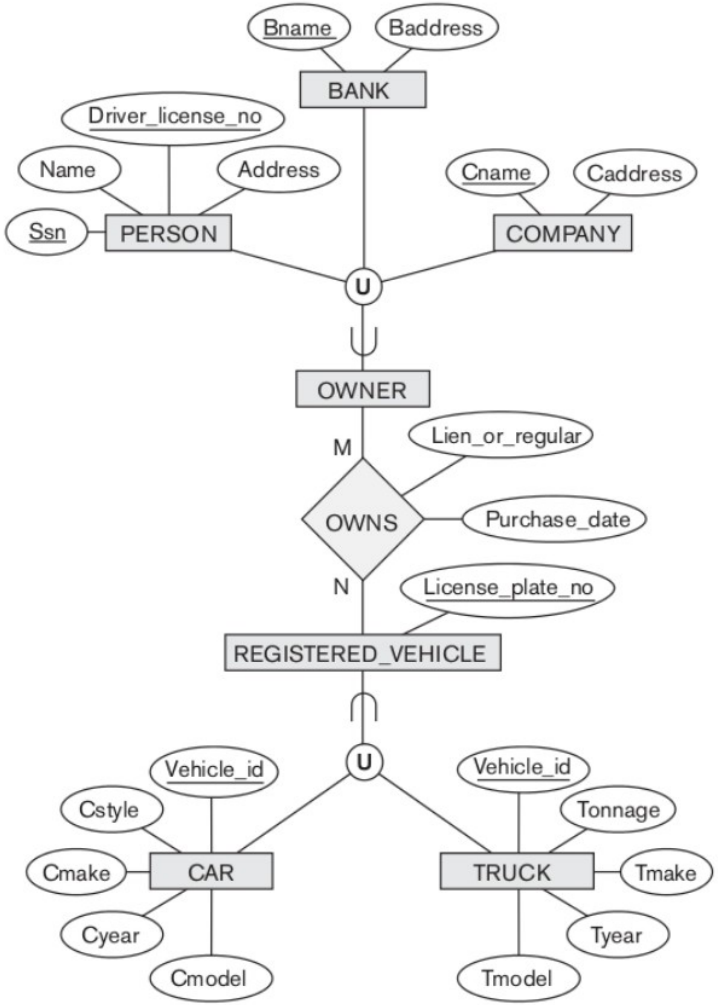
Bname	Baddress	Owner_id

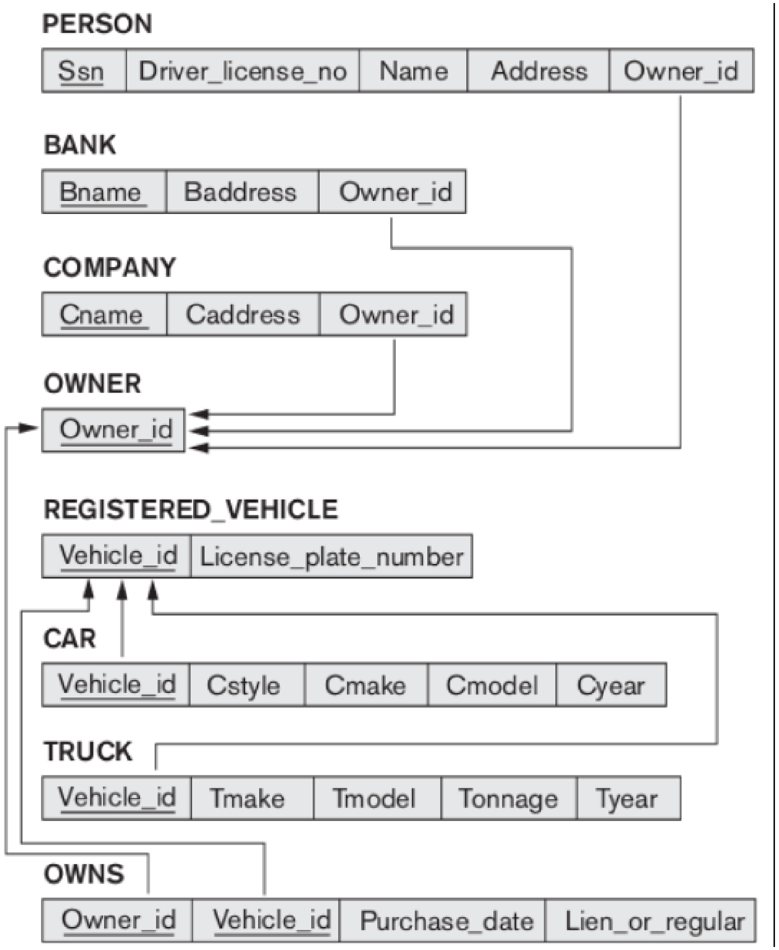
COMPANY

Cname	Caddress	Owner_id

OWNER

Owner_id





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

Notes

- Steps 8A-8D/9 may be intermixed where appropriate
- For a more elaborated way of handling generalization / specialization, and especially (multiple) inheritance, we might need a more suited DBMS system, like an object-oriented DBMS (OODBMS).

Ready? No!

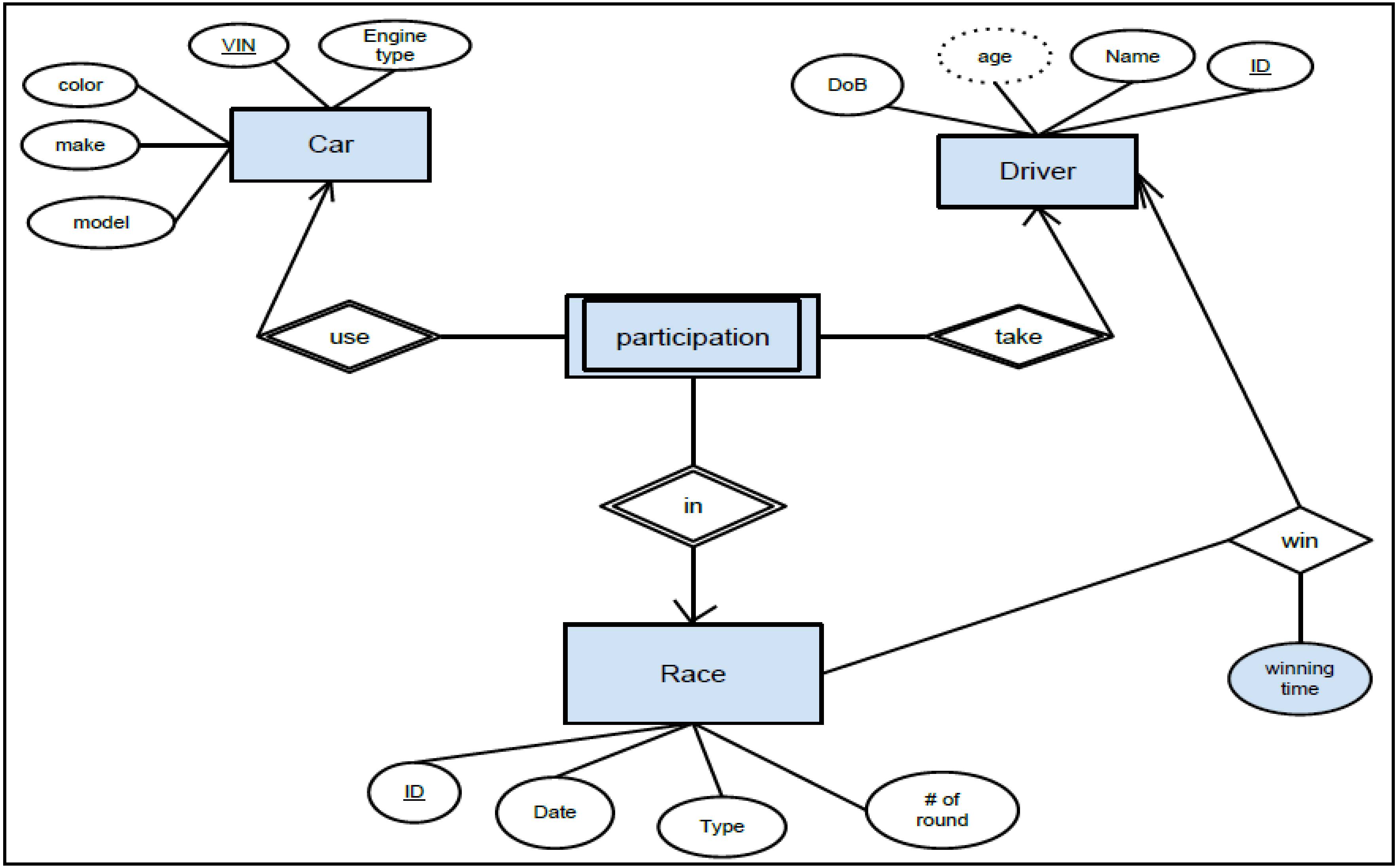
- The schema has to be refined:
  - Entities with 1-to-1 relations might be joined into one table
  - High-degree relation-types might be included in a table
  - If there are no keys for a strong entity type, a identification code has to be invented
  - Domain restrictions have to be recorded
  - Candidate keys have to be appointed (using UNIQUE).

And...

- Set cascade on delete and update for foreign keys of:
  - weak entity types
  - M:N and higher-degree relations
  - Multivalued attributes
- Take over extra restrictions

Quiz
(Car Race Database)

- Convert the following ER model to Relational Model.


VIN
Engine type
color
make
model
Car
age
Name
ID
DoB
Driver
use
participation
take
in
win
Race
winning time
ID
Date
Type
# of round