Faculty of Computer Science and Engineering
Ho Chi Minh City University of Technology

Chapter 3:
The Relational Data Model

Database Systems
(CO2013)

Computer Science Program

Assoc. Prof. Dr. Võ Thị Ngọc Châu

(chauvtn@hcmut.edu.vn)

Semester 1 – 2022-2023

Content

- Chapter 1: An Overview of Database Systems
- Chapter 2: The Entity-Relationship Model
- **Chapter 3: The Relational Data Model**
- Chapter 4: The SQL Language
- Chapter 5: Relational Database Design
- Chapter 6: Physical Storage and Data Management
- Chapter 7: Database Security

Chapter 3:
The Relational Data Model

- 3.1. Concepts
- 3.2. Relation schemas. Relations
- 3.3. Mapping an entity-relationship schema into a relational database schema
- 3.4. The Relational algebra

Main References

Text:

- [1] R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 6th Edition, Pearson- Addison Wesley, 2011.
  - ***R. Elmasri, S. R. Navathe, Fundamentals of Database Systems- 7th Edition, Pearson, 2016.***

References:

[1] S. Chittayasothorn, *Relational Database Systems: Language, Conceptual Modeling and Design for Engineers*, Nutcha Printing Co. Ltd, 2017.

[3] A. Silberschatz, H. F. Korth, S. Sudarshan, Database System Concepts – 7th Edition, McGraw-Hill, 2020.

[4] H. G. Molina, J. D. Ullman, J. Widom, *Database Systems: The Complete Book - 2nd Edition,* Prentice-Hall, 2009.

[5] R. Ramakrishnan, J. Gehrke, *Database Management Systems – 4th Edition,* McGraw-Hill, 2018.

[6] M. P. Papazoglou, S. Spaccapietra, Z. Tari, *Advances in Object-Oriented Data Modeling*, MIT Press, 2000.

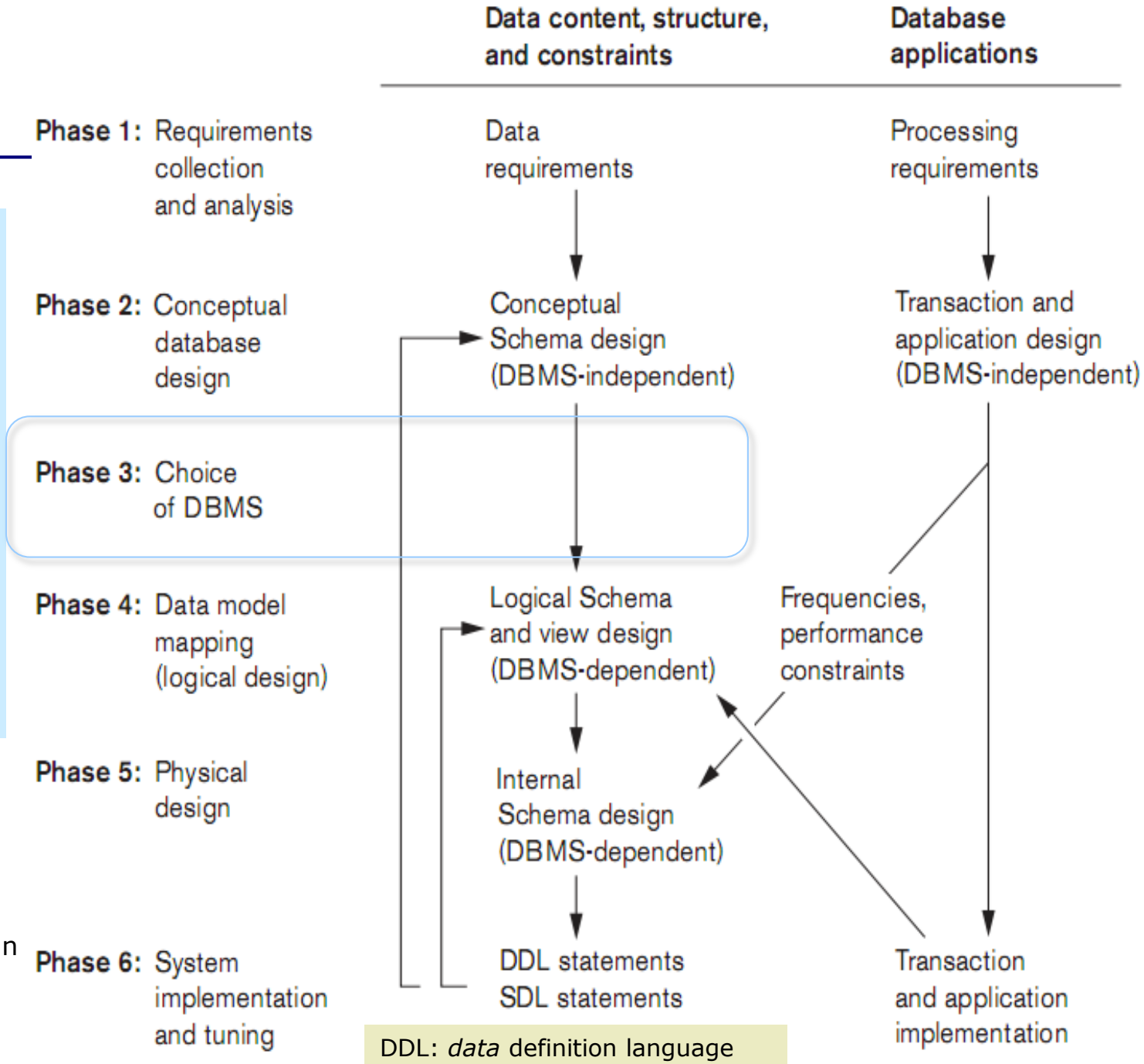
[7]. G. Simsion, *Data Modeling: Theory and Practice*, Technics Publications, LLC, 2007.

Database design

A representational data model supported by the chosen DBMS

Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.



Phases of database design and implementation for large databases

Source: [1]

Representational data model

- A data model (aka implementation/logical data model)
  - Provides concepts understood by end-users and able to be used to describe *the structure of a database*
    - the data types, relationships, and constraints that should hold for the data
    - a set of basic operations for specifying retrievals and updates on the database
  - Hides some details of data storage but able to be implemented on a computer system in a direct way (in some DBMS)
  - Example: the *relational* data model

Data model

E. F. Codd. Data models in database management, ACM, 1980.

- A combination of three following components
  - (1). A collection of *data structure types* (the building blocks of any database that conforms to the model);
  - (2). A collection of *operators or inferencing rules*, which can be applied to any valid instances of the data types listed in (1), to retrieve or derive data from any parts of those structures in any combinations desired;
  - (3). A collection of *general integrity rules*, which implicitly or explicitly define the set of consistent database states or changes of state or both –-- these rules may sometimes be expressed as insert-update-delete rules.

Concepts

- Relational data model
- Relational database
- Relation
  - Degree, cardinality
- Tuple
- Attribute
- Domain
- (*Atomic*) Value
  - NULL
- Operation (Operator)
- Constraint
  - Inherent model-based (*implicit*) constraint
  - Schema-based (*explicit*) constraint
  - Application-based (*semantic*) constraint

Data model

E. F. Codd. Data models in database management, ACM, 1980.

- A combination of three following components
  - (1). A collection of *data structure types* (the building blocks of any database that conforms to the model);
  - (2). A collection of *operators or inferencing rules*, which can be applied to any valid instances of the data types listed in (1), to retrieve or derive data from any parts of those structures in any combinations desired;
  - (3). A collection of *general integrity rules*, which implicitly or explicitly define the set of consistent database states or changes of state or both –-- these rules may sometimes be expressed as insert-update-delete rules.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- The term *relation* is used here in its accepted mathematical sense.
- A *relational database* based on the relational data model is a collection of relations.
- Given sets S_1, S_2, …, S_n (not necessarily distinct), R is a *relation* on these *n* sets if it is a set of *n*-tuples each of which has its first element from S_1, its second element from S_2, and so on.
  - S_j is the *j*th *domain* of R, including *atomic* values.
  - R is said to have *degree n*.
  - R is said to have *cardinality* |R|.
- More concisely, *relation* R is a subset of the *Cartesian product* $S_1 x S_2 x \ldots x S_n$.

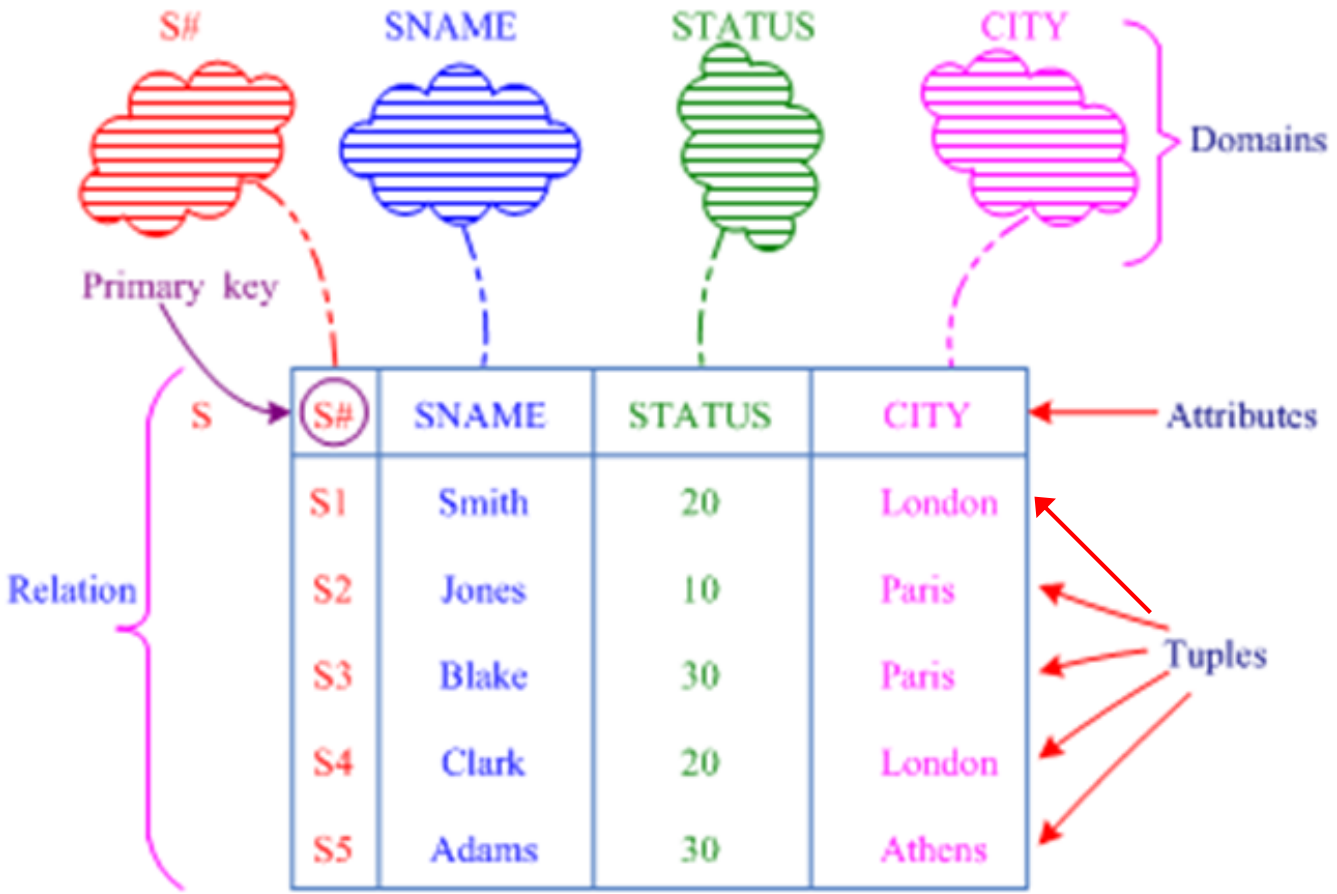
The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- An *n*-ary relation R has the following properties (aka inherent model-based (*implicit*) constraints):
  - Each row represents an *n*-tuple of R.
  - The ordering of rows is immaterial (not important).
  - All rows are distinct.
  - The ordering of columns is significant – it corresponds to the ordering S_1, S_2, …, S_n of the domains on which R is defined.
  - The significance of each column is partially conveyed by labeling it with the *name of the corresponding domain*, which is called *attribute*.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.



The *supplier* relation S.

Relation schema: S (S#, SNAME, STATUS, CITY)
Degree (= the number of attributes): 4
Cardinality (=|S|) : 5

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

supply	(*supplier*	*part*	*project*	*quantity*)
	1	2	5	17
	1	3	5	23
	2	3	7	9
	2	7	5	4
	4	1	1	12

A relation of degree 4, cardinality 5 (=|supply|)

A relation of degree 4, called *supply*, which reflects the shipments-in-progress of *parts* from specified *suppliers* to specified *projects* in specified *quantities*.

Its attributes are: supplier, part, project, quantity.

Its relation schema: *supply* (supplier, part, project, quantity)

The Relational Data Model

- Relation schema ∞ Relation scheme ∞

  Relation intension

  - $R\ (A_1, A_2, \ldots\ , A_n)$

- Relation ∞ Relation state ∞ Relation instance

  ∞ Relation extension of the relation schema

  - r = a set of n-tuples = $\{t_1, t_2, \ldots\ , t_m\}$

Data model

E. F. Codd. Data models in database management, ACM, 1980.

- A combination of three following components
  - (1). A collection of *data structure types* (the building blocks of any database that conforms to the model);
  - (2). A collection of *operators or inferencing rules*, which can be applied to any valid instances of the data types listed in (1), to retrieve or derive data from any parts of those structures in any combinations desired;
  - (3). A collection of *general integrity rules*, which implicitly or explicitly define the set of consistent database states or changes of state or both –-- these rules may sometimes be expressed as insert-update-delete rules.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- Basic operations on relations
  - Usual set operations: union, intersection, minus
    - Compatible relations
  - Projection (π)
    - Select certain columns of a relation (striking out the others)
  - Selection (σ)
    - Select a subset of the tuples from a relation that satisfy a selection condition
  - Cartesian product (cross product) (x)
    - Combine tuples from two relations in a combinatorial fashion
  - Join (inner/outer theta join, equijoin, natural join)
    - Combine related tuples from two relations into single tuples

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

R	(*supplier*	*part*)	S	(*part*	*project*)
	1	1		1	1
	2	1		1	2
	2	2		2	1

FIG. 5. Two joinable relations

R*S	(*supplier*	*part*	*project*)
	1	1	1
	1	1	2
	2	1	1
	2	1	2
	2	2	1

FIG. 6. The natural join of R with S (from Figure 5)

The Relational Data Model

- A *Complete* Set of Relational Algebra Operations :
  - $\{\sigma, \pi, \cup, \rho, -, \times\}$
    - σ = select
    - π = project
    - $\cup$ = union
    - ρ = rename
    - $-$ = minus (difference)
    - $\times$ = Cartesian product

Data model

E. F. Codd. Data models in database management, ACM, 1980.

- A combination of three following components
  - (1). A collection of *data structure types* (the building blocks of any database that conforms to the model);
  - (2). A collection of *operators or inferencing rules*, which can be applied to any valid instances of the data types listed in (1), to retrieve or derive data from any parts of those structures in any combinations desired;
  - (3). A collection of *general integrity rules*, which implicitly or explicitly define the set of consistent database states or changes of state or both –-- these rules may sometimes be expressed as insert-update-delete rules.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- Schema-based (*explicit*) constraints
  - Domain constraints
  - Key constraints
  - Constraints on nulls
  - Entity integrity constraints
  - Referential integrity constraints

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- Domain constraints
  - Within each tuple, the value of each attribute A must be an *atomic* value from the domain dom(A).
- Key constraints

  1. Two *distinct* tuples in any state of the relation cannot have identical values for (all) the attributes in the *key*.

  2. It is a *minimal superkey* – that is, a superkey from which we cannot remove any attributes and still have the uniqueness constraint in condition 1 hold.

     - *Superkey* of the relation schema R specifies a *uniqueness* constraint that no two distinct tuples in any state r of R can have the same value for the superkey.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- Key constraints
  - A relation schema may have more than one key.
    - Candidate keys: *primary key* and secondary keys
- Constraints on nulls
  - Specify whether *null* values are or are not permitted on attributes
- Entity integrity constraint
  - *No primary key value can be null*.
- Referential integrity constraint
  - Specify a referential integrity constraint between the two relation schemas R_1 and R_2

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- Referential integrity constraint
  - A set of attributes FK (*foreign key*) in relation schema R_1 is a ***foreign key*** of R_1 that references relation R_2 if it satisfies the following two rules:
    - The attributes in FK have the ***same domain***(s) as the ***primary key*** attributes PK of R_2; the attributes FK are said to reference or refer to the relation R_2.
    - A value of FK in a tuple t_1 of the current state $r_1(R_1)$ either occurs as a value of PK for some tuple t_2 in the current state $r_2(R_2)$ or ***is null***. In the former case, we have $t_1[FK] = t_2[PK]$, and we say that the tuple t_1 references or refers to the tuple t_2.

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

Relation Employee

Relation *schema*: EMPLOYEE (FNAME, MINIT, LNAME, SSN, BDATE, ADDRESS, SEX, SALARY, SUPERSSN, DNO)

Attributes = FNAME, MINIT, LNAME, SSN, BDATE, ADDRESS, SEX, SALARY, SUPERSSN, DNO

Domain of Attribute SEX = {F, M}

Domain of SALARY = a set of positive integer numbers

Primary key = SSN

Foreign key = SUPERSSN, DNO

Check all the *implicit and explicit constraints* on this relation!

The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

EMPLOYEE	FNAME	MINIT	LNAME	SSN	BDATE	ADDRESS	SEX	SALARY	SUPERSSN	DNO
	John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
	Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
	Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
	Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
	Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
	Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
	Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
	James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	null	1

Relation Employee

Relation *schema*: EMPLOYEE (FNAME, MINIT, LNAME, SSN, BDATE, ADDRESS, SEX, SALARY, SUPERSSN, DNO)

Examine more application-based (*semantic*) constraints:

- Each department has at most 20 employees.
- Each employee must has a supervisor in the same department. Otherwise, he/ she is the manager of his/ her department.
- The salary of an employee must not be greater than the salary of the manager of the department that the employee works for.
- ...

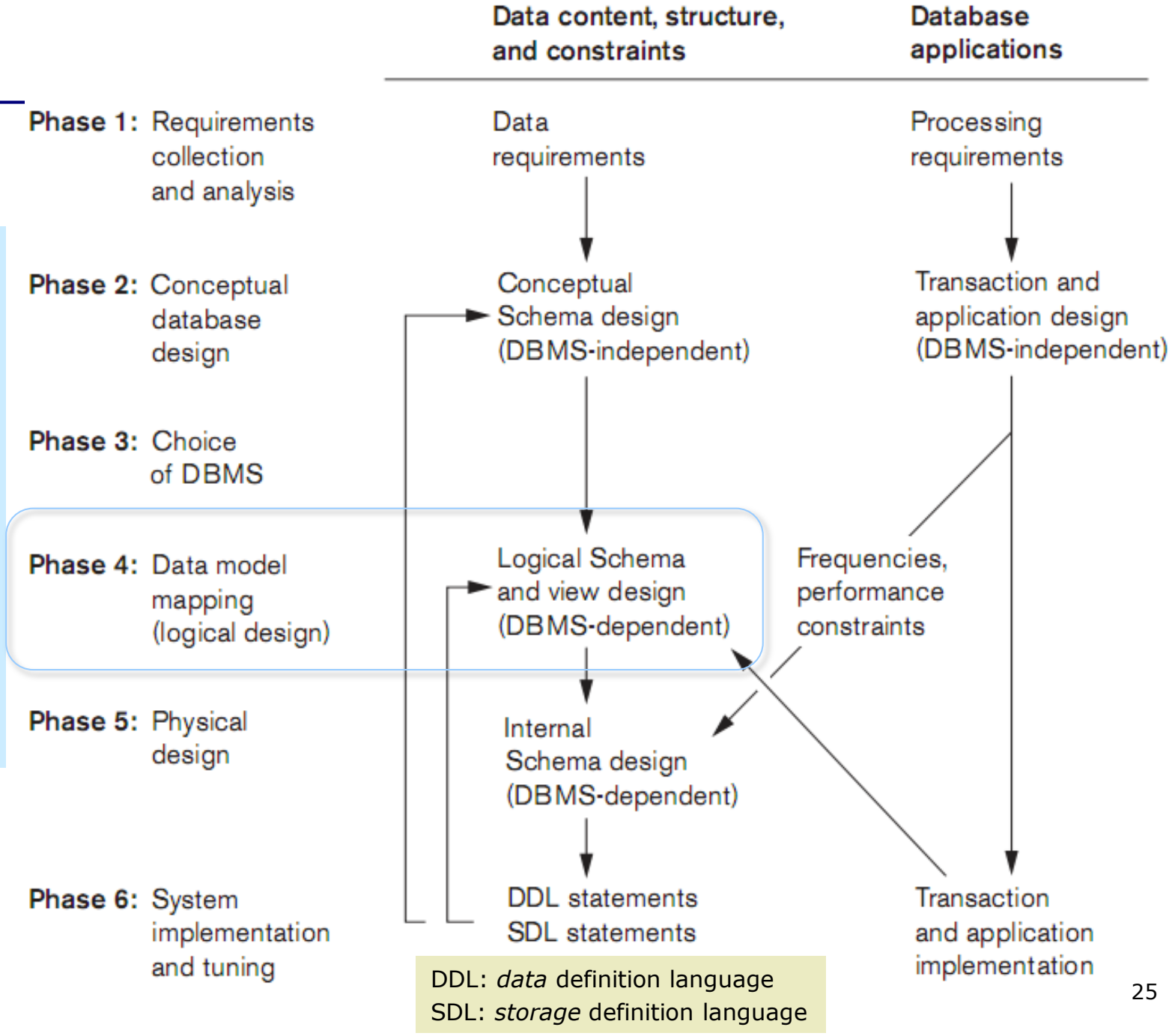
The Relational Data Model

E. F. Codd. *A Relational Model of Data for Large Shared Data Banks*. Communications of the ACM 13(6)(June, 1970) 377-387.

- High-level database language on relational databases
  - Structured Query Language (SQL)
    - Data definition language (DDL)
    - Data query language (SELECT)
    - Data manipulation language (INSERT, DELETE, UPDATE)
    - …
  - Versions: SQL-86, SQL-89, SQL-92, SQL-99, SQL-2003, SQL-2008, SQL-2011, SQL-2016, …

Database design

Data model mapping from a conceptual schema based on the ER model to a relational database schema based on the relational data model



Phases of database design and implementation for large databases

Source: [1]

Data model mapping

- Mapping algorithms
  - Automatically create a relational schema from a conceptual schema design
- Notes on data model mapping
  - Slightly different for other post-relational data models as compared to the relational data model
  - Constraints on databases
    - Inherent model-based implicit constraints
    - Schema-based constraints
    - Application-based constraints

Data model mapping

- Informal measures of quality for relation schema design
  - Semantics of the attributes
    - How to interpret the attribute values stored in a tuple of the relation – how the attribute values in a tuple relate to one another
  - Reducing the redundant values in tuples
    - Storage space & update anomalies
  - Reducing the null values in tuples
    - Multiple interpretations of *null*s
  - Disallowing the possibility of generating spurious tuples
    - Join relations with equality conditions on attributes that are either primary keys or foreign keys in a way that guarantees that no spurious tuples are generated

Data model mapping

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

Correspondence between the ER and Relational Models

Source: [1]

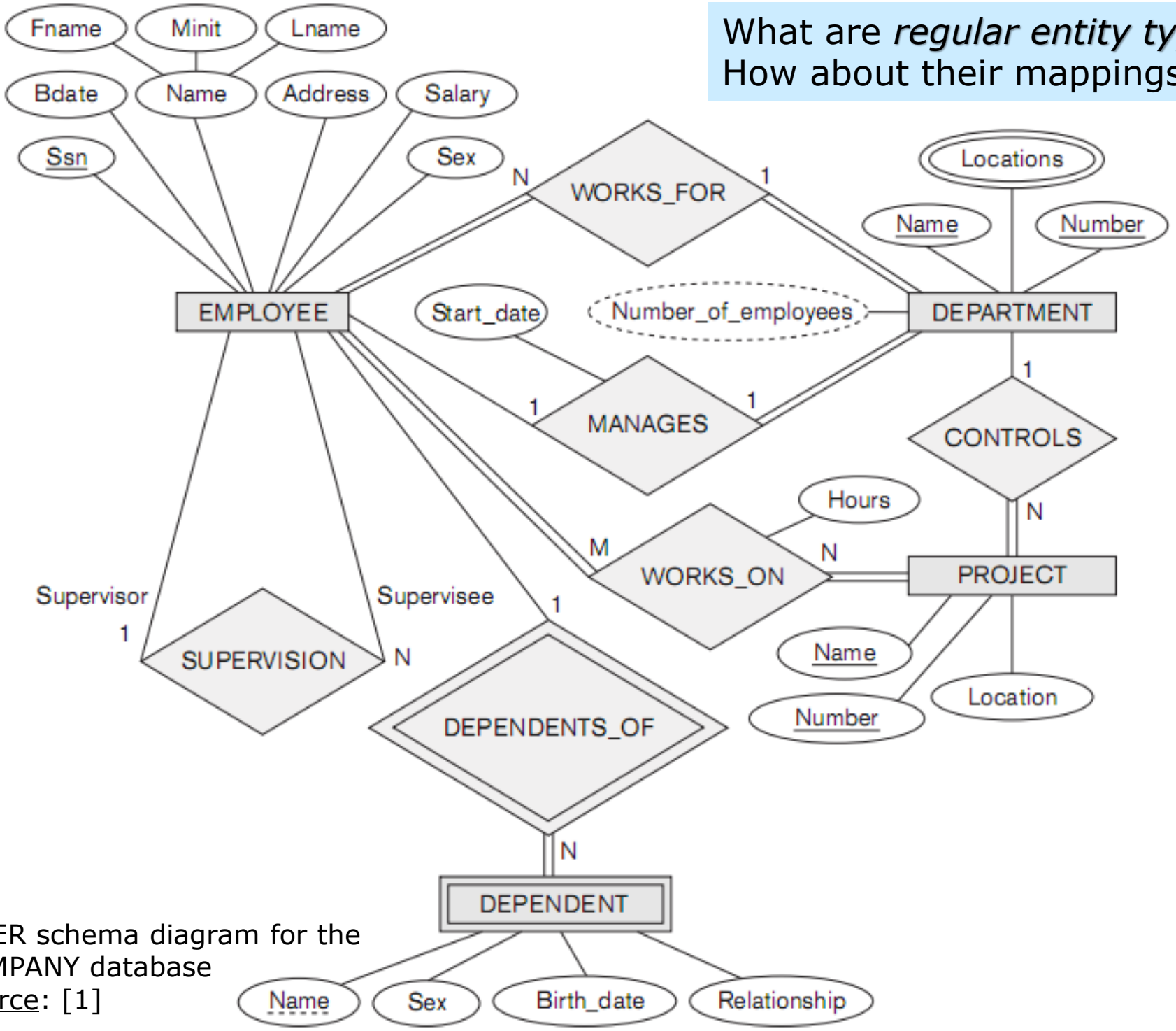
Data model mapping

- Step 1: Mapping of Regular Entity Types
- Step 2: Mapping of Weak Entity Types
- Step 3: Mapping of Binary 1:1 Relationship 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
- Step 8: Mapping of Specialization or Generalization
- Step 9: Mapping of Union Types (Categories)

Data model mapping

- Step 1: Mapping of Regular Entity Types
  - For each *regular entity type E*, create *a relation R* that includes all the simple attributes of E.
  - Include only the *simple component attributes of a composite attribute*.
  - Choose *one of the key attributes of E* as the *primary key for R*. If the chosen key of E is a composite, then the set of simple attributes that form it will together form the primary key of R.
  - If multiple keys were identified for E during the conceptual design, the information describing the attributes that form each additional key is kept in order to specify secondary (*unique*) keys of relation R. Knowledge about keys is also kept for indexing purposes and other types of analyses.
  - The relations that are created from the mapping of entity types are sometimes called *entity relations* because each tuple represents an entity instance.

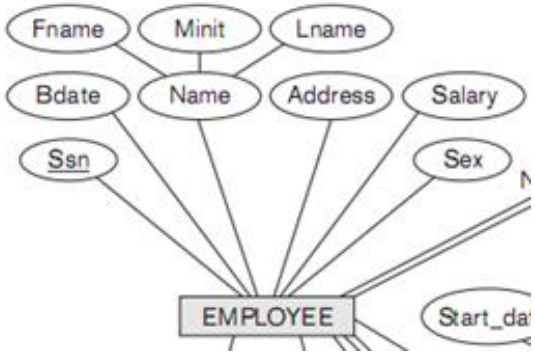
What are *regular entity types*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

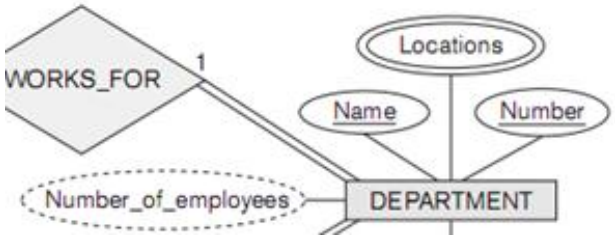
- Step 1: Mapping of Regular Entity Types



Entity Relation Schemas:

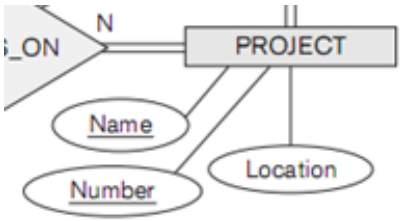
EMPLOYEE (Ssn, Fname, Minit, Lname, Bdate, Address, Salary, Sex)

Primary key: Ssn



DEPARTMENT (Number, Name)

Primary key: Number
Secondary (*unique, not null*) key: Name



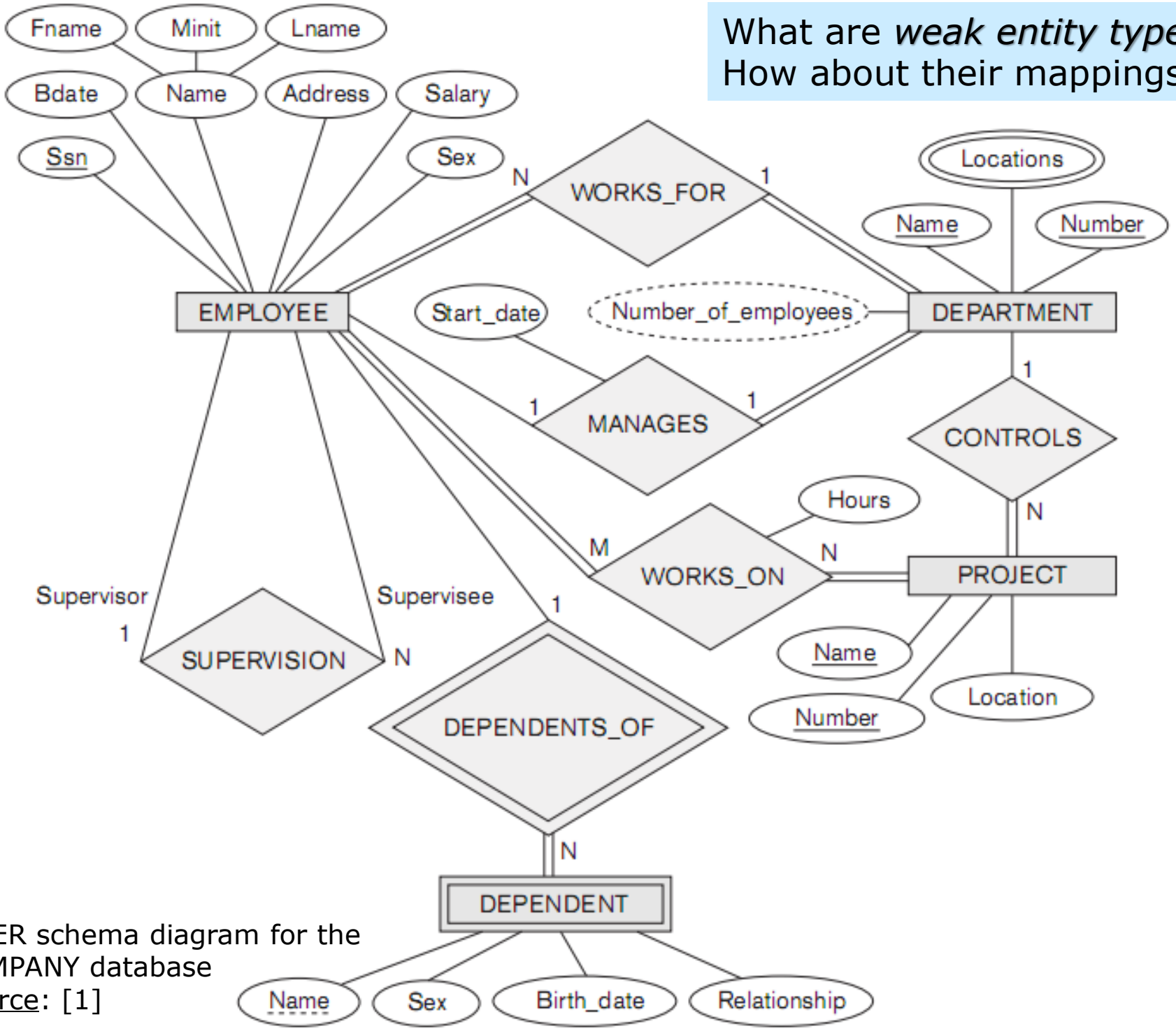
PROJECT (Number, Name, Location)

Primary key: Number
Secondary (*unique, not null*) key: Name

Data model mapping

- Step 2: Mapping of Weak Entity Types
  - For each *weak entity type W* with owner entity type E, create *a relation R* and include *all simple attributes* (or *simple components of composite attributes*) of W as *attributes of R*.
  - In addition, include as foreign key attributes of R, the primary key attribute(s) of the relation(s) that correspond to the owner entity type(s); this takes care of mapping the *identifying relationship type* of W.
  - 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.
  - If there is a weak entity type E2 whose owner is also a weak entity type E1, then E1 should be mapped before E2 to determine its primary key first.

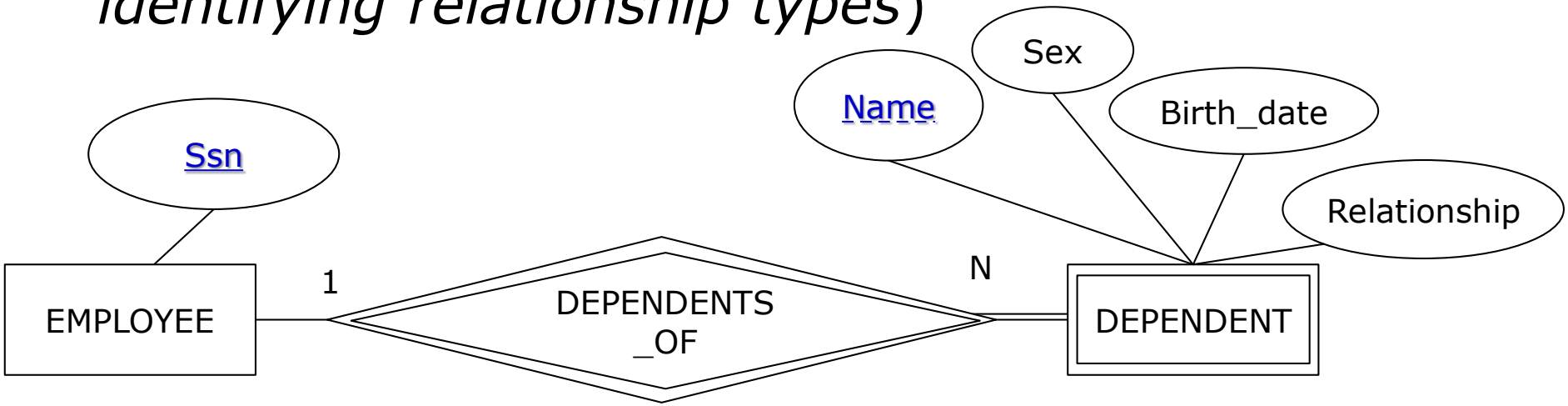
What are *weak entity types*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

- Step 2: Mapping of Weak Entity Types (*and their identifying relationship types*)



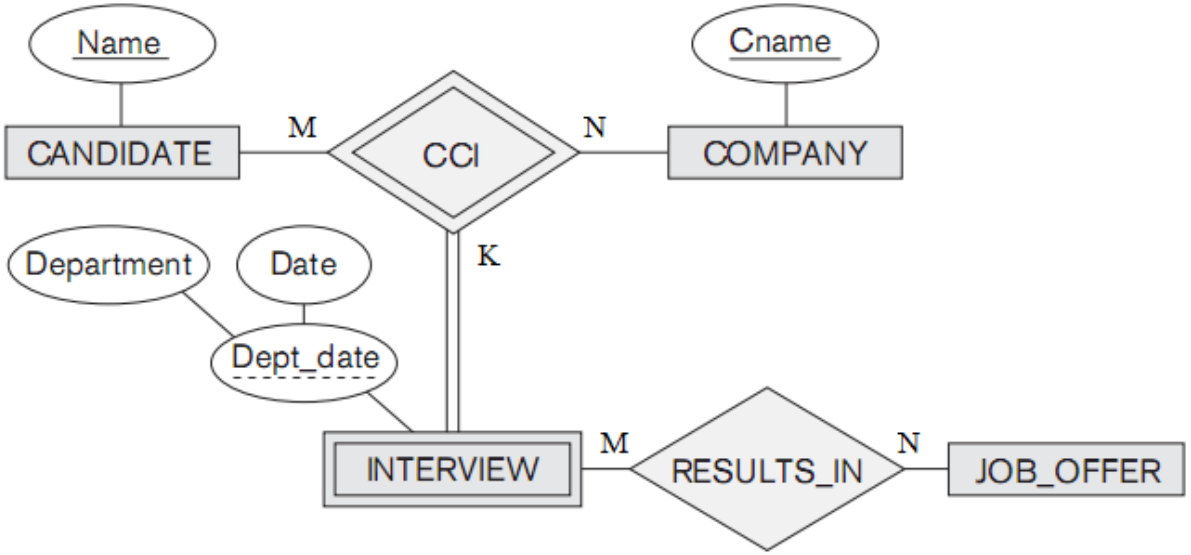
Relation Schemas:

EMPLOYEE (Ssn, …)
Primary key: Ssn

DEPENDENT (Ssn, Name, Birth_date, Sex, Relationship)
Primary key: Ssn, Name
Foreign key: Ssn *with reference to* EMPLOYEE.Ssn

Data model mapping

- Step 2: Mapping of Weak Entity Types



Relation Schemas:

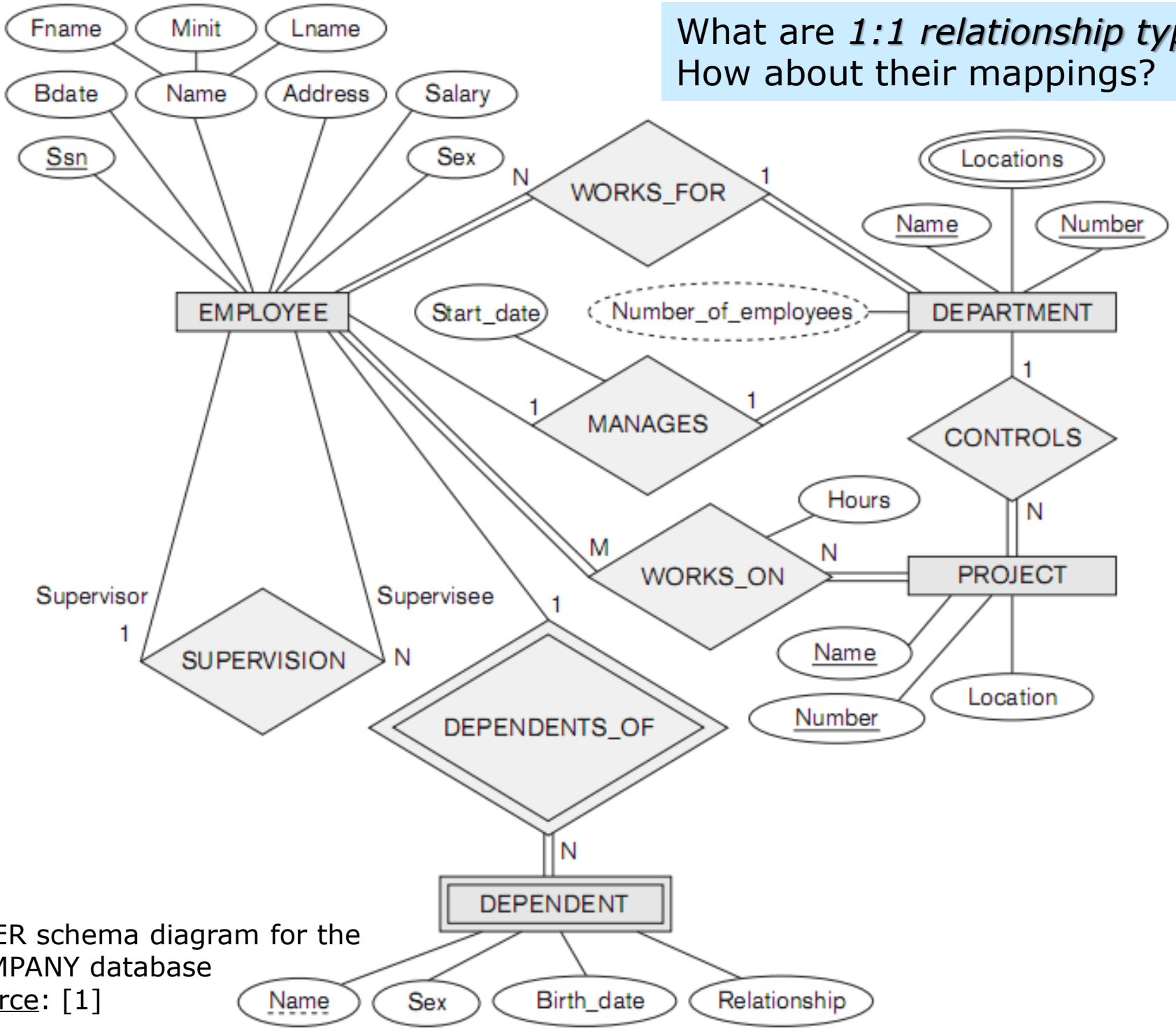
CANDIDATE (Name, …)
COMPANY (Cname, …)

INTERVIEW (Cname, Name, Department, Date)
Primary key: Cname, Name, Department, Date
Foreign key: Cname *to* COMPANY.Cname, Name *to* CANDIDATE.Name

Data model mapping

- Step 3: Mapping of Binary 1:1 Relationship Types
  - For each binary 1:1 relationship type R, identify the relations S and T that correspond to the entity types participating in R.
    - *Foreign key approach*: 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.
    - *Merged relation approach*: An alternative mapping of a 1:1 relationship type is to merge the two entity types and the relationship into a *single relation*. This is possible when *both participations are total*, as this would indicate that the two tables will have the exact same number of tuples at all times.
    - *Cross-reference or relationship relation approach*: The third option is to set 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. As we will see, this approach is required for binary M:N relationships. The relation 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 from T. *The relation R will include the primary key attributes of S and T as foreign keys to S and T*. The *primary key of R will be one of the two foreign keys, and the other foreign key will be a unique key of R*. The drawback is having an extra relation, and requiring an extra join operation when combining related tuples from the tables.

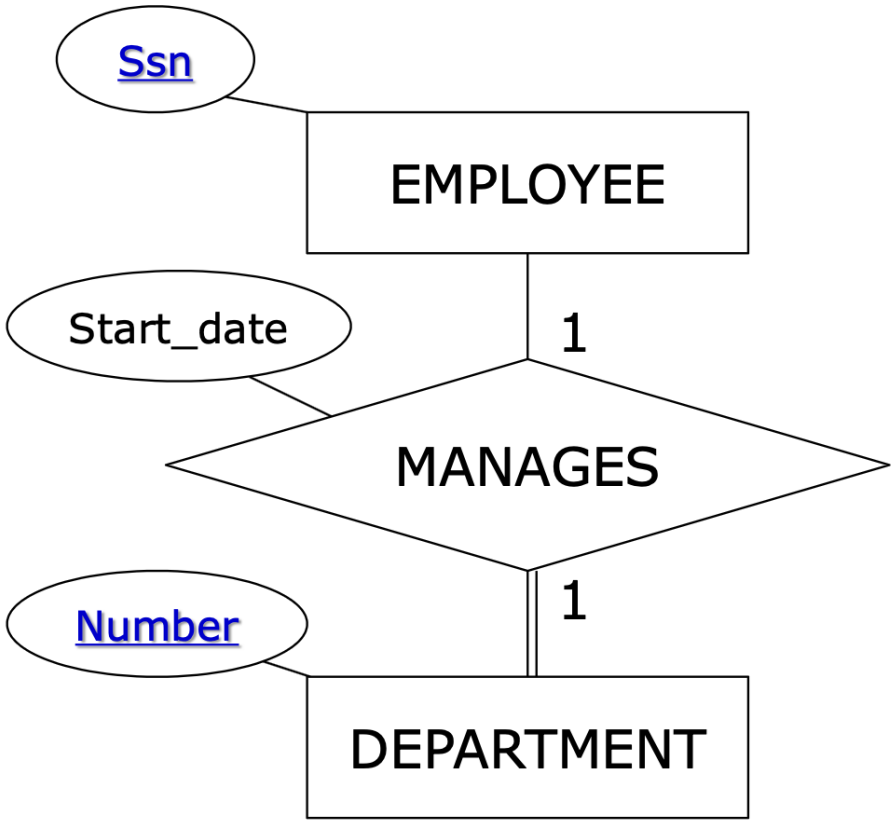
What are *1:1 relationship types*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

- Step 3: Mapping of Binary 1:1 Relationship Types



Foreign key approach:

EMPLOYEE (Ssn, …)
DEPARTMENT (Number, …, Ssn, Start_date)
Foreign key (on *total participation*): Ssn *to* EMPLOYEE.Ssn
NOT NULL: Ssn, *Start_date (?)*
Uniqueness: Ssn

Merged relation approach:

Not applicable because only DEPARTMENT has total participation.

Relationship relation approach:

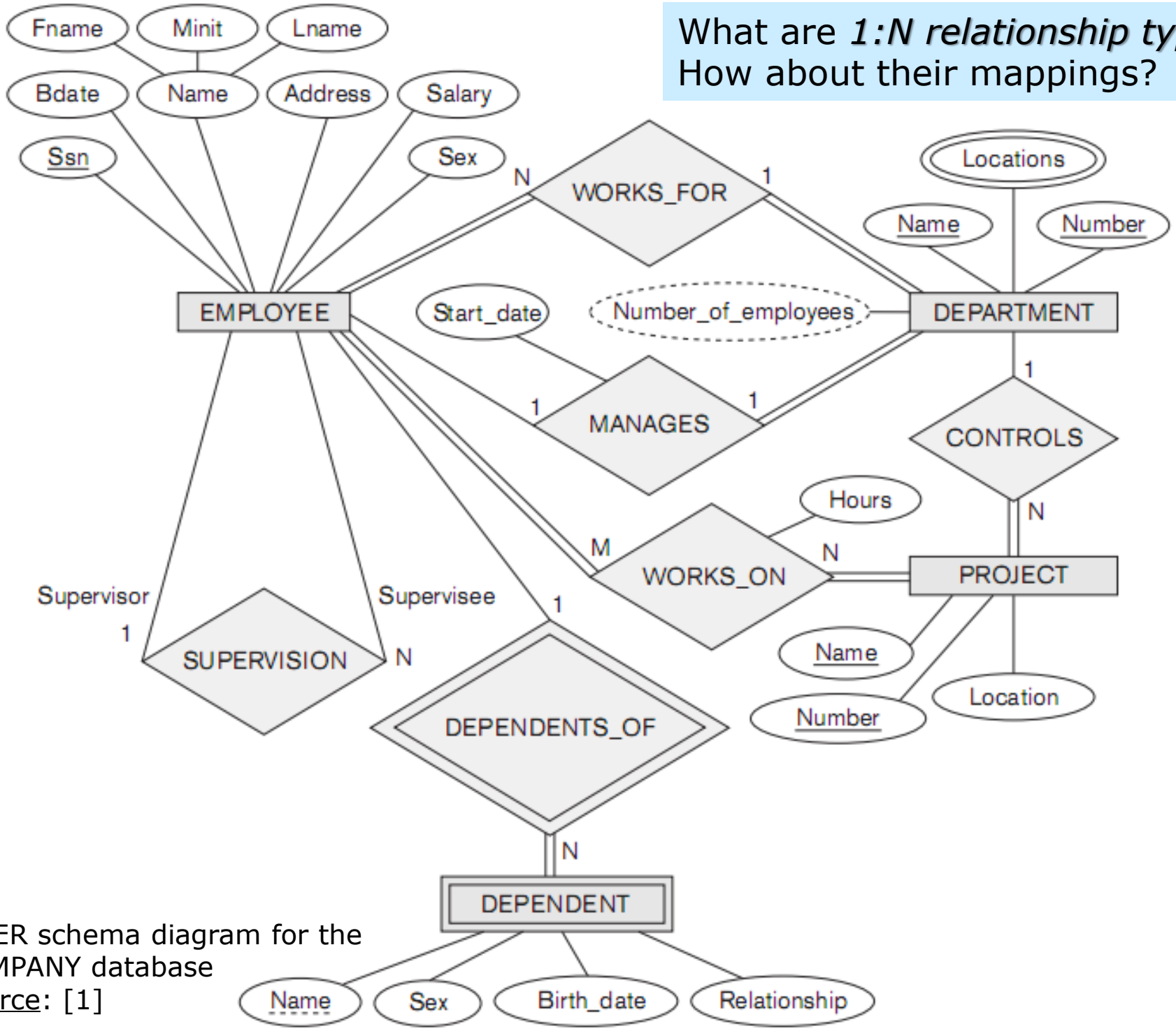
EMPLOYEE (Ssn, …)
DEPARTMENT (Number, …)

MANAGES (Ssn, Number, Start_date)
Primary key: Number
Unique key: Ssn
Foreign key: Ssn *to* EMPLOYEE.Ssn, Number *to* DEPARTMENT.Number
NOT NULL: Ssn, *Start_date (?)*

Data model mapping

- Step 4: Mapping of Binary 1:N Relationship Types
  - For each regular binary 1:N relationship type R, identify the *relation S* that represents 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 at the *1-side*; because each entity instance on the *N*-side is related to at most one entity instance on the 1-side of the relationship type. Include any simple attributes (or simple components of composite attributes) of the 1:N relationship type as attributes of S.
  - An alternative approach is to use the *relationship relation* (cross-reference) option. We create a *separate relation R* whose attributes are the primary keys of S and T, which will also be foreign keys to S and T. The *primary key of R is the same as the primary key of S*. This option can be used if *few tuples in S participate in the relationship to avoid excessive NULL values in the foreign key*.

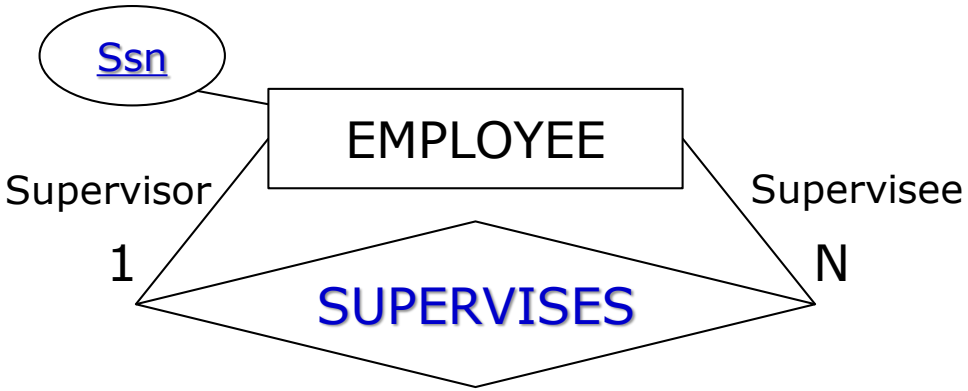
What are *1:N relationship types*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

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



Foreign key approach:

EMPLOYEE (Ssn, …, Supervisor_Ssn)
Foreign key (at the *N-side*): Supervisor_Ssn *to* EMPLOYEE.Ssn

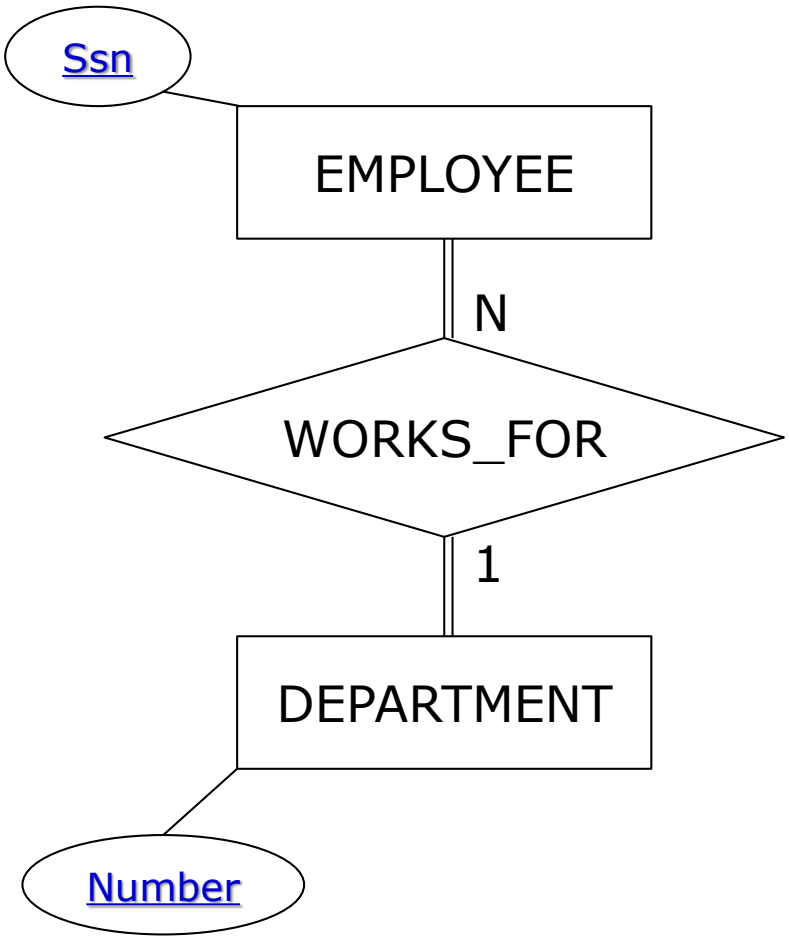
Relationship relation approach:

EMPLOYEE (Ssn, …)

SUPERVISES (Supervisee_Ssn, Supervisor_Ssn)
Primary key (at the *N-side*): Supervisee_Ssn
Foreign key: Supervisee_Ssn *to* EMPLOYEE.Ssn, Supervisor_Ssn *to* EMPLOYEE.Ssn
NOT NULL: Supervisor_Ssn

Data model mapping

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



Foreign key approach:

DEPARTMENT (Number, …)
EMPLOYEE (Ssn, …, Number)
Foreign key (at the *N-side*) of EMPLOYEE: Number *to* DEPARTMENT.Number
NOT NULL: Number
Check total participation of DEPARTMENT.Number in EMPLOYEE

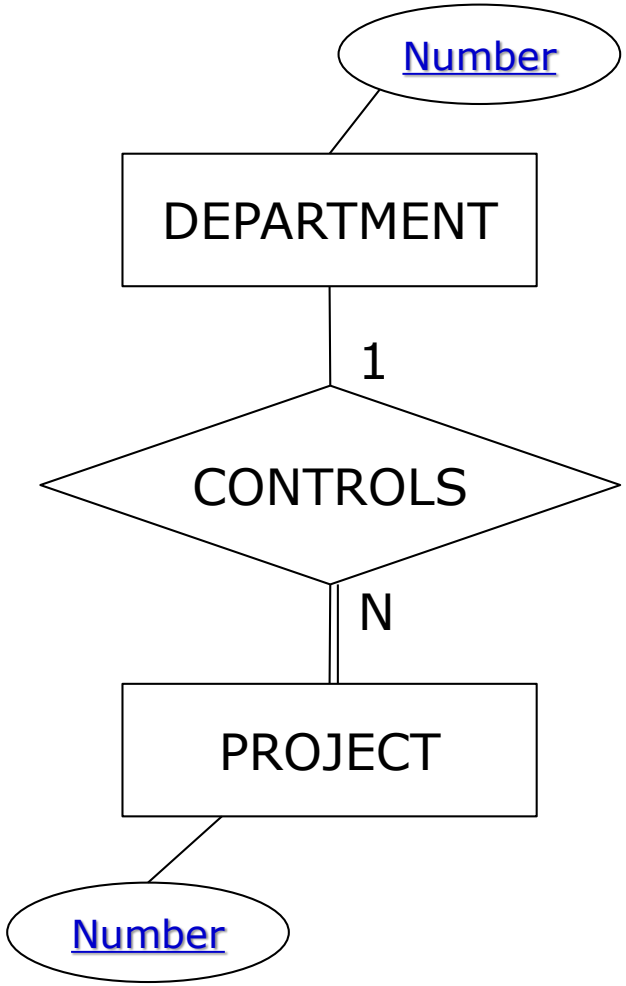
Relationship relation approach:

EMPLOYEE (Ssn, …)
DEPARTMENT (Number, …)

WORKS_FOR (Ssn, Number)
Primary key (at the *N-side*): Ssn
Foreign key: Ssn *to* EMPLOYEE.Ssn, Number *to* DEPARTMENT.Number
NOT NULL: Number
Check total participation of EMPLOYEE.Ssn and DEPARTMENT.Number in WORKS_FOR

Data model mapping

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



Foreign key approach:

DEPARTMENT (Number, …)
PROJECT (Number, …, Department_Number)
Foreign key (at the *N-side*) of PROJECT:
Department_Number *to* DEPARTMENT.Number
NOT NULL: Department_Number

Relationship relation approach:

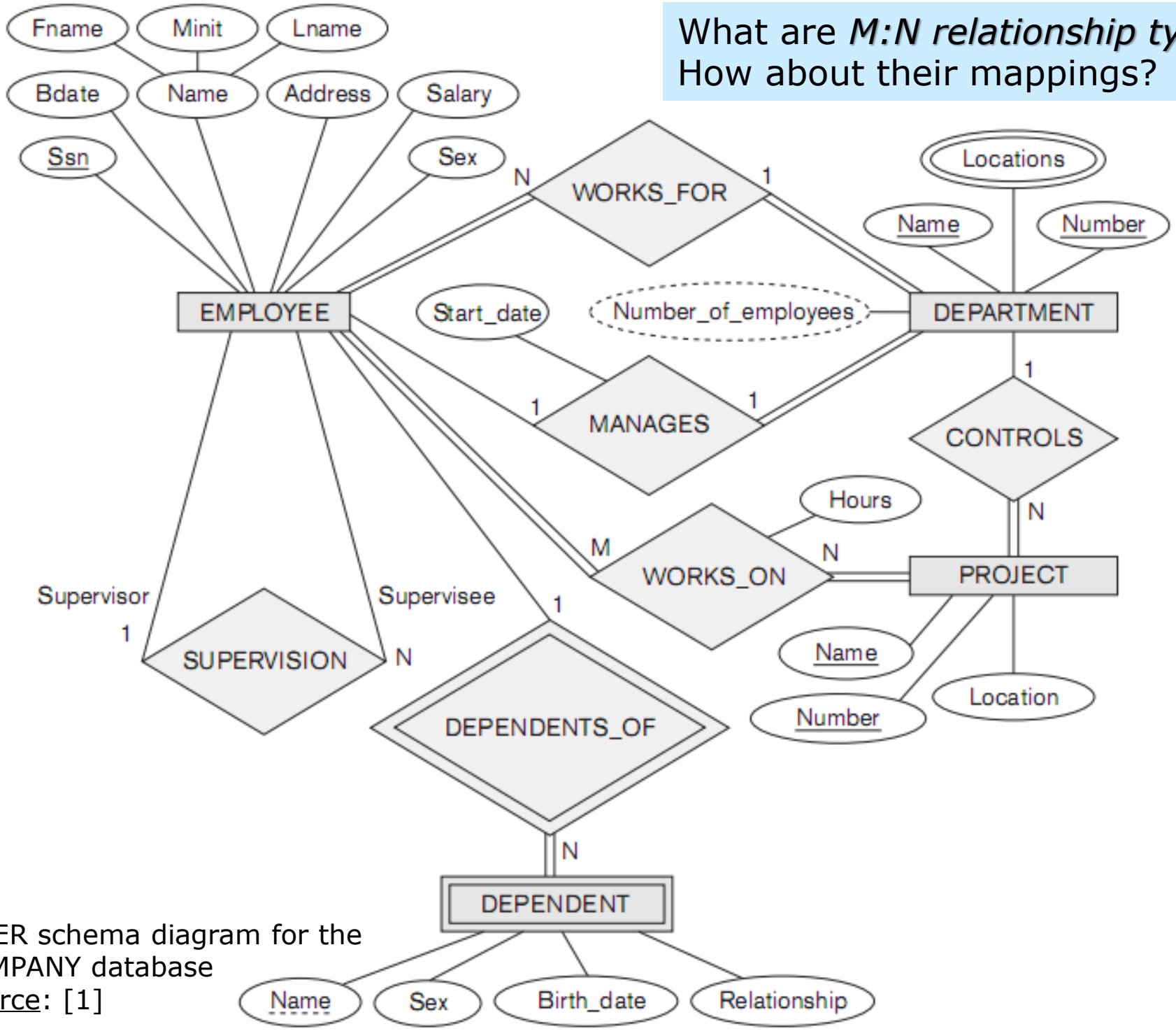
DEPARTMENT (Number, …)
PROJECT (Number, …)

CONTROLS (Project_Number, Department_Number)
Primary key (at the *N-side*): Project_Number
Foreign key: Project_Number *to* Project.Number,
Department_Number *to* DEPARTMENT.Number
NOT NULL: Department_Number
Check total participation of PROJECT.Number in CONTROLS

Data model mapping

- Step 5: Mapping of Binary M:N Relationship Types
  - For each binary M:N relationship type R, create a *new relation S* to represent R.
  - 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.

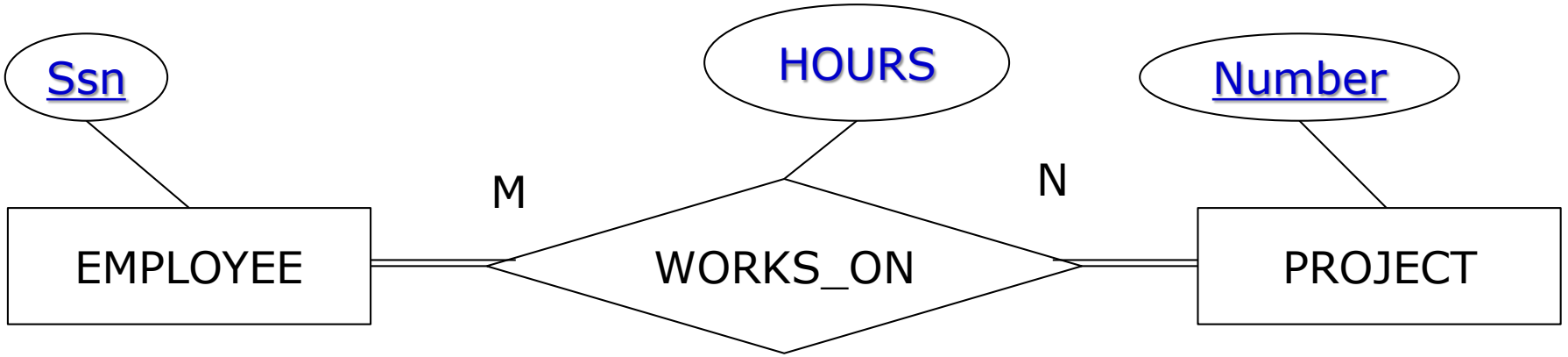
What are *M:N relationship types*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

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



Relationship relation approach:

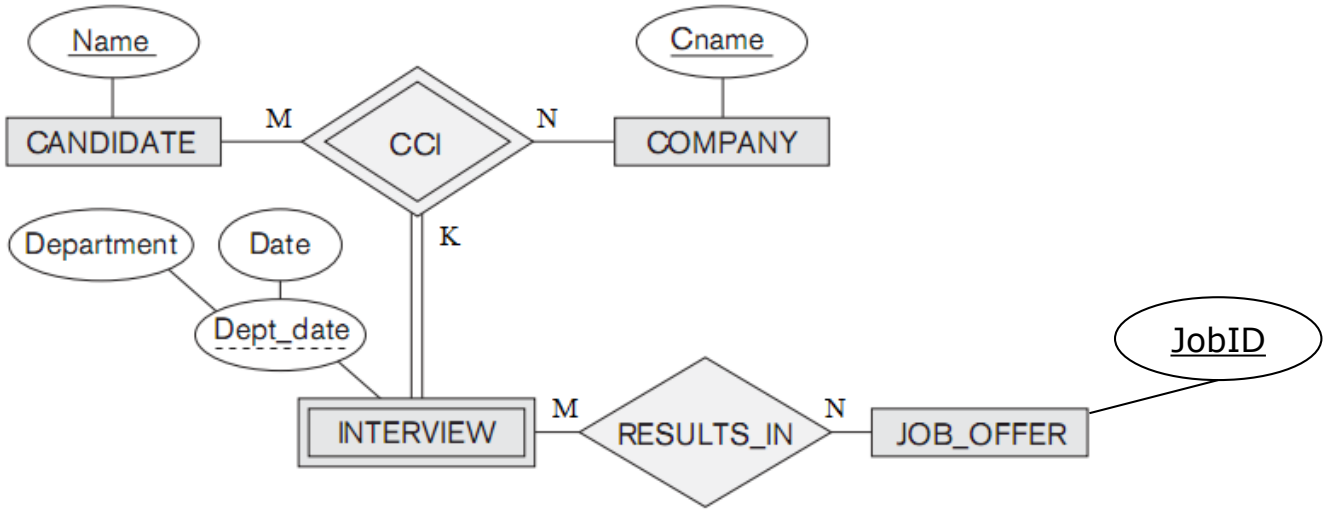
EMPLOYEE (Ssn, …)
PROJECT (Number, …)

WORKS_ON (Ssn, Number, Hours)

Primary key: Ssn, Number
Foreign key: Ssn *to* EMPLOYEE.Ssn, Number *to* PROJECT.Number
Check total participation of EMPLOYEE.Ssn and PROJECT.Number in WORKS_ON

Data model mapping

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



Relation Schemas:

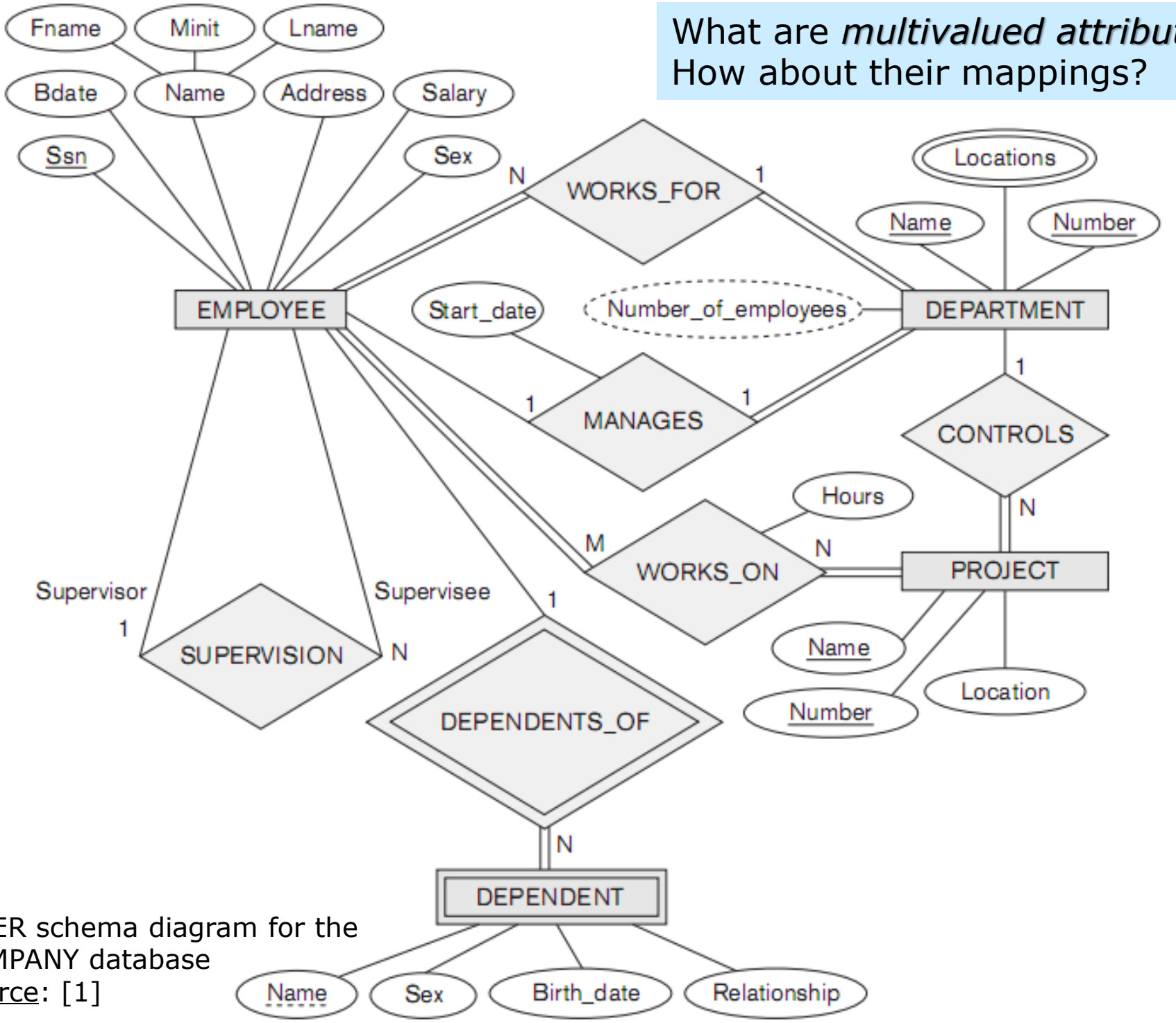
INTERVIEW (InterviewID, Cname, Name, Department, Date)
JOB_OFFER (JobID, …)

RESULTS_IN (InterviewID, JobID)
Primary key: InterviewID, JobID
Foreign key: InterviewID *to* INTERVIEW.InterviewID, JobID *to* JOB_OFFER.JobID

Data model mapping

- 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 or relationship type that has A as a multivalued attribute.
  - The *primary key of R* is the *combination of A and K*.
  - If the multivalued attribute is composite, we include its simple components.

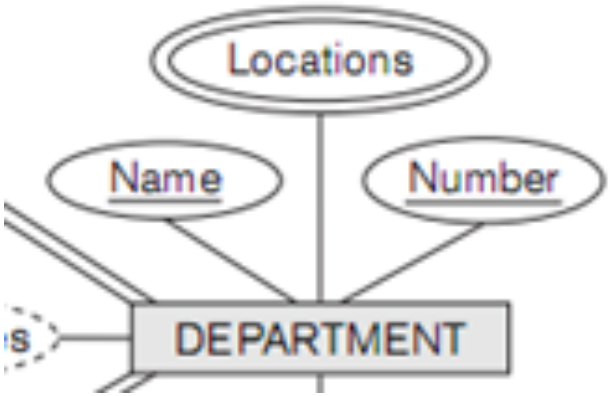
What are *multivalued attributes*?
How about their mappings?



An ER schema diagram for the COMPANY database
Source: [1]

Data model mapping

- Step 6: Mapping of Multivalued Attributes



Relation Schemas:

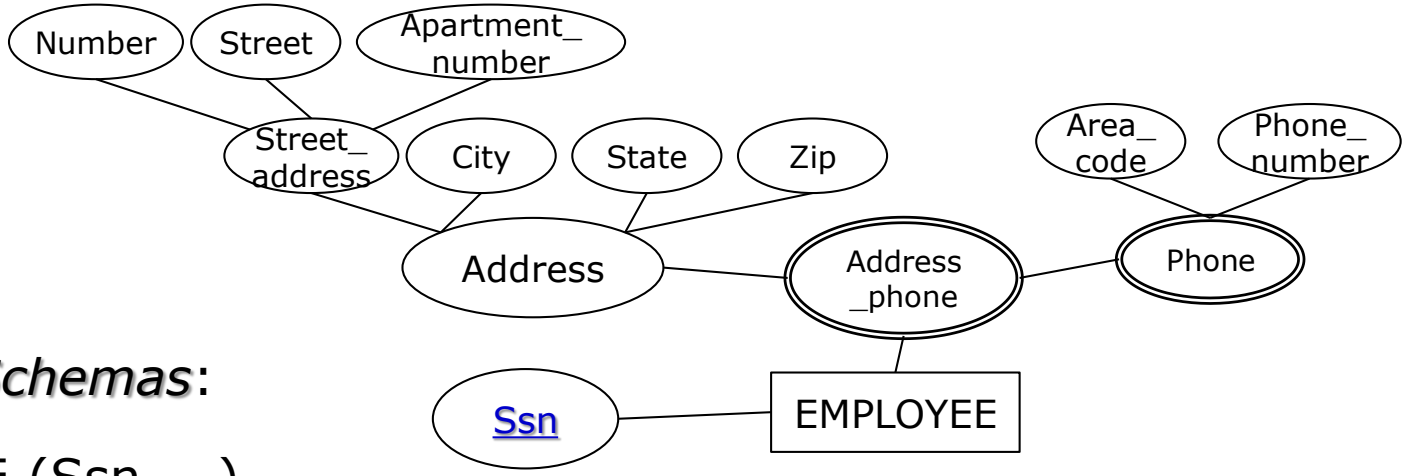
DEPARTMENT (Number, …)

Locations (Number, ALocation)

Primary key: Number, ALocation
Foreign key: Number *to* Department.Number

Data model mapping

- Step 6: Mapping of Multivalued Attributes



Relation Schemas:

EMPLOYEE (Ssn, …)

Address_phone (Address_ID, Ssn, Number, Street, Apartment_number, City, State, Zip)

Primary key: Address_ID
Foreign key: Ssn *to* EMPLOYEE.Ssn

Phone (Address_ID, Area_code, Phone_number)

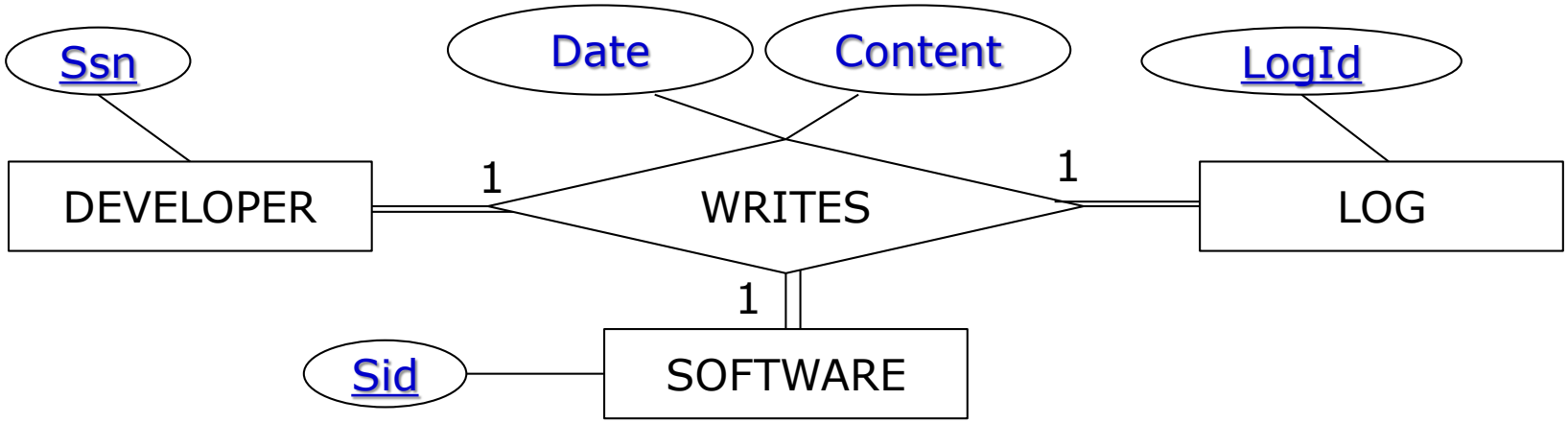
Primary key: Address_ID, Area_code, Phone_number
Foreign key: Address_ID *to* Address_phone.Address_ID

Data model mapping

- Step 7: Mapping of N-ary Relationship Types
  - For *each N-ary relationship type R*, where N > 2, create a *new relation 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.
  - The *primary key of S is usually a combination of all the foreign keys that reference the relations representing the participating entity types*. However, 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.

Data model mapping

- Step 7: Mapping of N-ary Relationship Types



Relationship Relation Schemas:

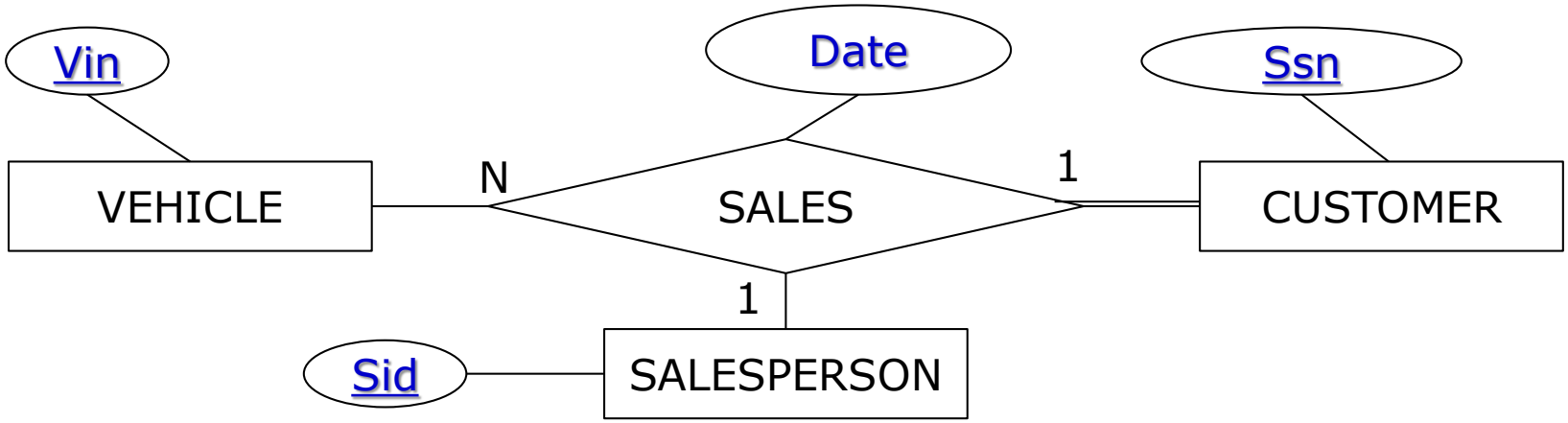
DEVELOPER (Ssn, …)
SOFTWARE (Sid, …)
LOG (LogId, …)

WRITES (Ssn, Sid, LogId, Content, Date)

Primary key: Ssn, Sid
Secondary keys: Ssn, LogId, Sid, LogId
Foreign key: Ssn *to* DEVELOPER.Ssn, Sid *to* SOFTWARE.Sid, LogId *to* LOG.LogId
Check total participation of DEVELOPER.Ssn, LOG.LogId, and SOFTWARE.Sid in WRITES

Data model mapping

- Step 7: Mapping of N-ary Relationship Types



Relationship Relation Schemas:

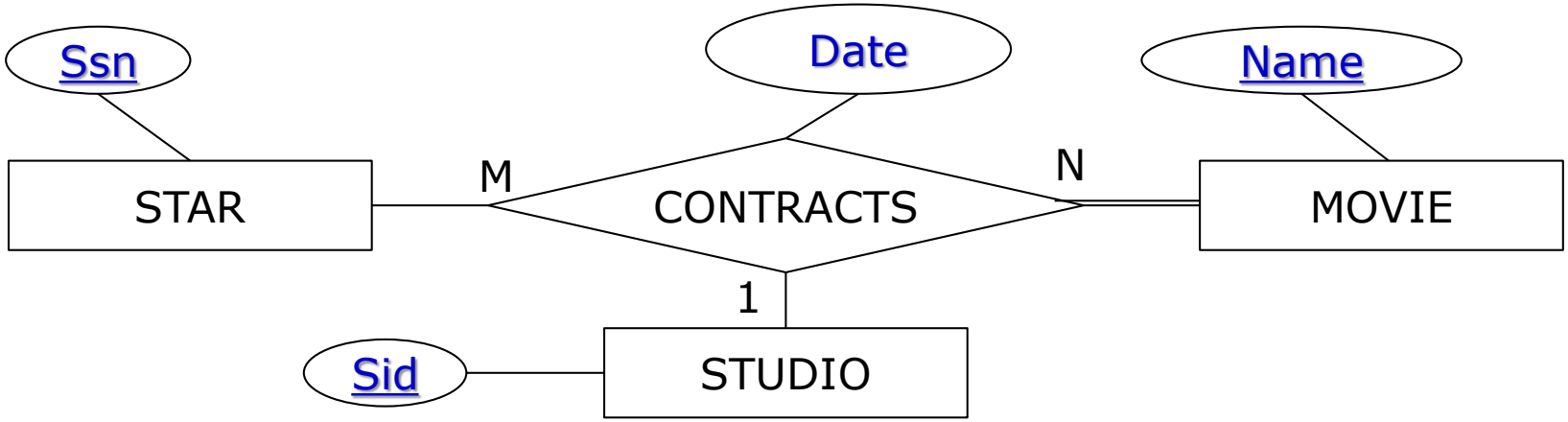
VEHICLE (Vin, …)
SALESPERSON (Sid, …)
CUSTOMER (Ssn, …)

SALES (Vin, Sid, Ssn, Date)

Primary key: Vin, Sid
Secondary key: Vin, Ssn
Foreign key: Vin *to* VEHICLE.Vin, Sid *to* SALESPERSON.Sid, Ssn *to* CUSTOMER.Ssn
Check total participation of CUSTOMER.Ssn in SALES

Data model mapping

- Step 7: Mapping of N-ary Relationship Types



Relationship Relation Schemas:

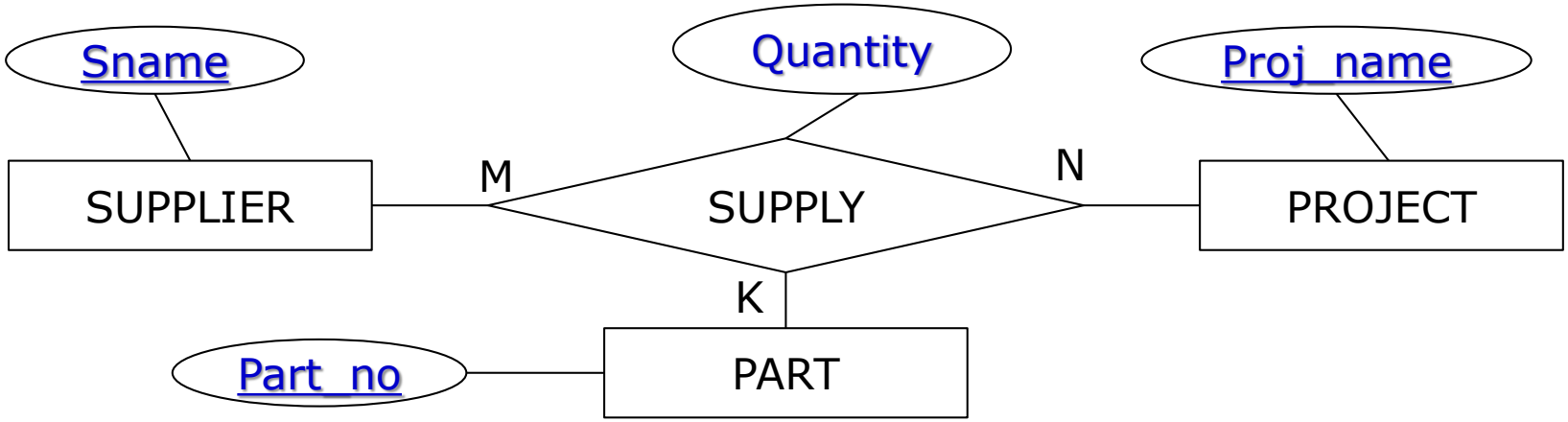
STAR (Ssn, …)
MOVIE (Name, …)
STUDIO (Sid, …)

CONTRACTS (Ssn, Name, Sid, Date)

Primary key: Ssn, Name
Foreign key: Ssn *to* Star.Ssn, Name *to* MOVIE.Name, Sid *to* STUDIO.Sid
Check total participation of MOVIE.Name in CONTRACTS

Data model mapping

- Step 7: Mapping of N-ary Relationship Types



Relationship Relation Schemas:

SUPPLIER (Sname, …)
PROJECT (Proj_name, …)
PART (Part_no, …)

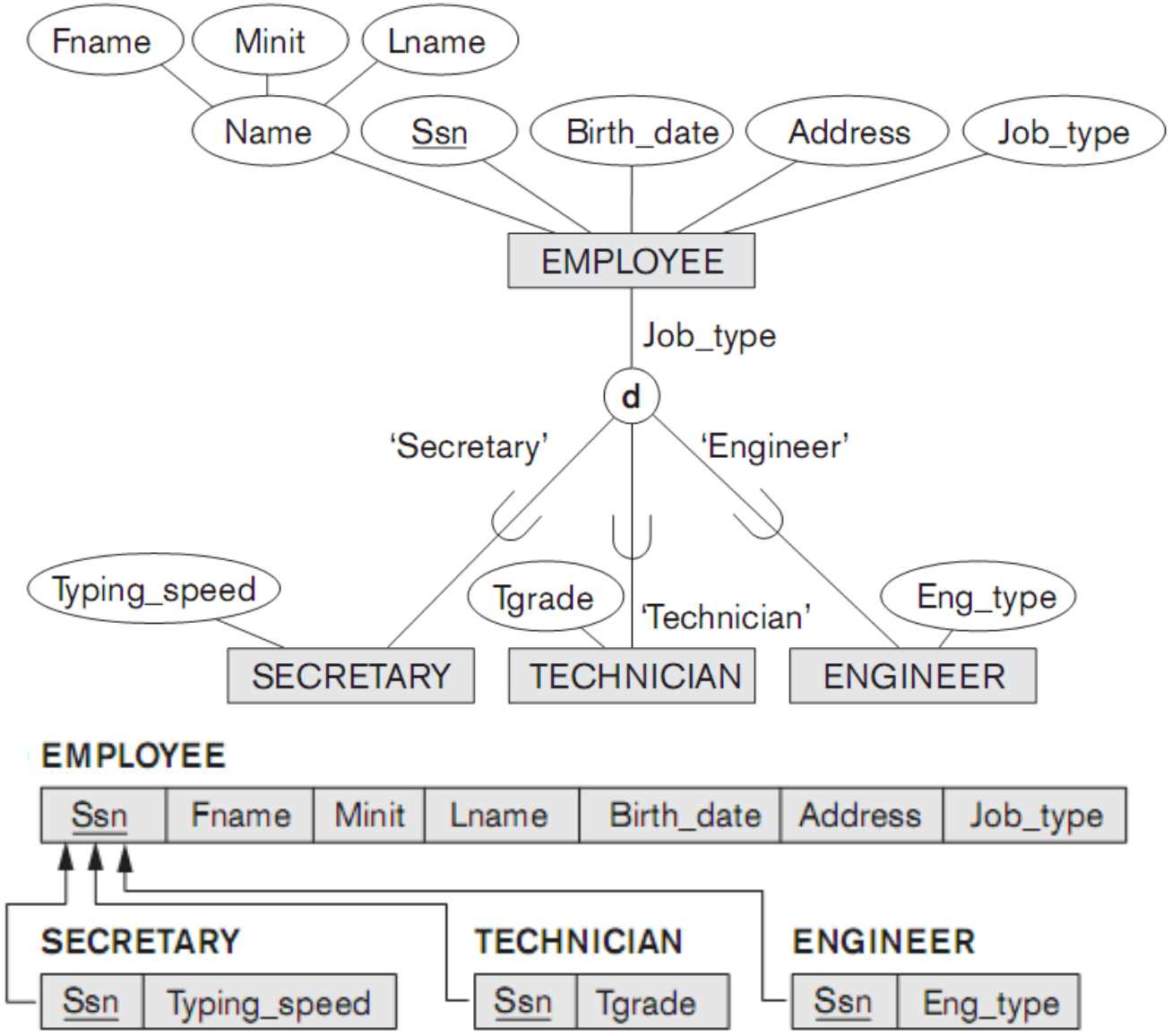
SUPPLY (Sname, Proj_name, Part_no, Quantity)

Primary key: Sname, Proj_name, Part_no
Foreign key: Sname *to* SUPPLIER.Sname, Proj_name *to* PROJECT.Proj_name, Part_no *to* PART.Part_no

Data model mapping

- Step 8: Mapping of Specialization or Generalization
  - Convert each specialization with *m* subclasses {S_1, S_2, …, S_m} and (generalized) superclass *C*, where the attributes of *C* are {*k*, a_1, …, a_n} and *k* is the (primary) key, into relation schemas:
    - Multiple relations – one relation for superclass and one relation for each subclass
      - *Any* specialization/ generalization
    - Multiple relations – one relation for each subclass (*total*)
      - Every entity in the superclass must belong to (at least) one of the subclasses.
    - Single relation with *one* type attribute
      - Subclasses are *disjoint*.
    - Single relation with *multiple* type attributes each of which corresponds to each subclass
      - Subclasses are *overlapping* (but able if *disjoint*)

Data model mapping



Using **Multiple relations – Superclass and subclasses**

CAR

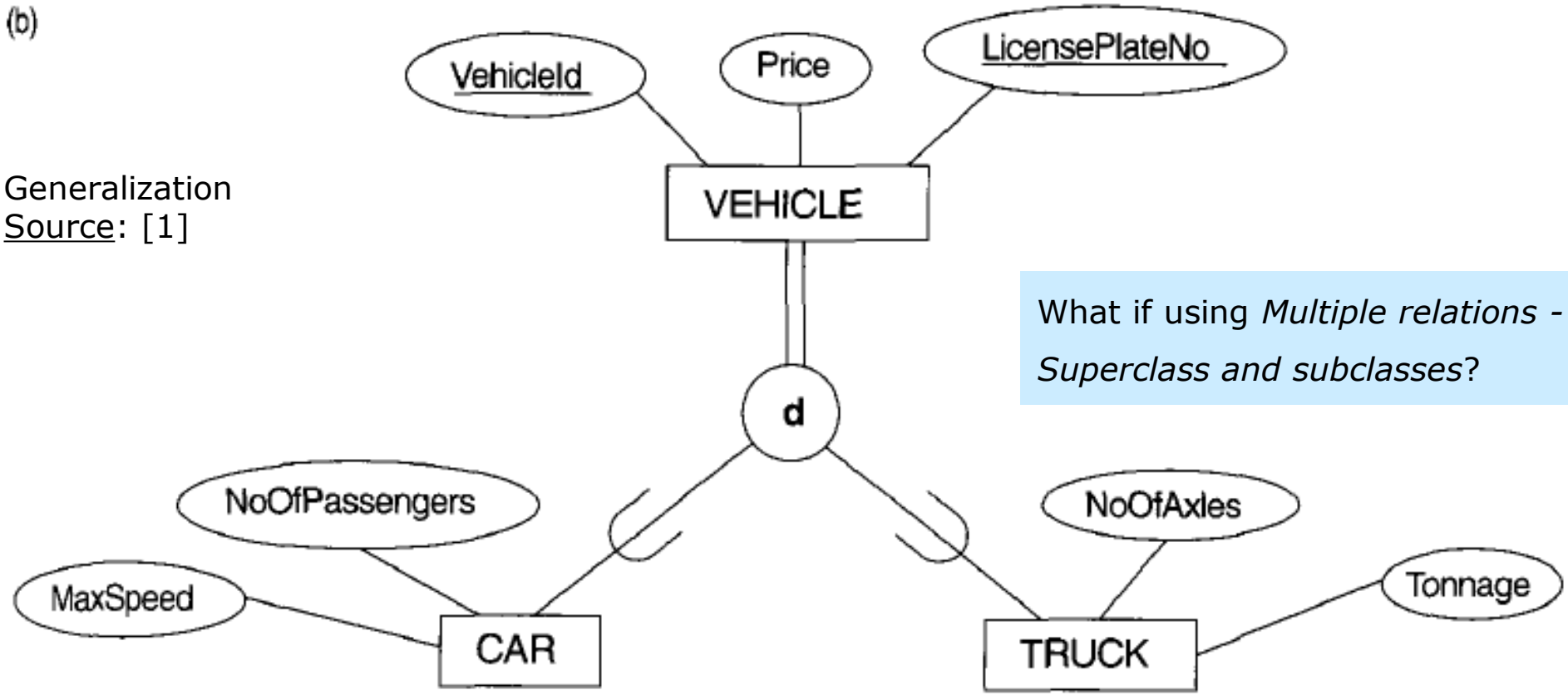
VehicleId	LicensePlateNo	Price	MaxSpeed	NoOfPassengers

TRUCK

VehicleId	LicensePlateNo	Price	NoOfAxles	Tonnage

Secondary key (UNIQUE, NOT NULL): LicensePlateNo
Using **Multiple relations – Subclass relations only (total)**

(b)



Generalization
Source: [1]

What if using *Multiple relations - Superclass and subclasses*?

Data model mapping

VEHICLE (Vehicle_id, License_plate_no, Price)

Secondary key (UNIQUE, NOT NULL): License_plate_no

CAR (Vehicle_id, Max_speed, No_of_passengers)

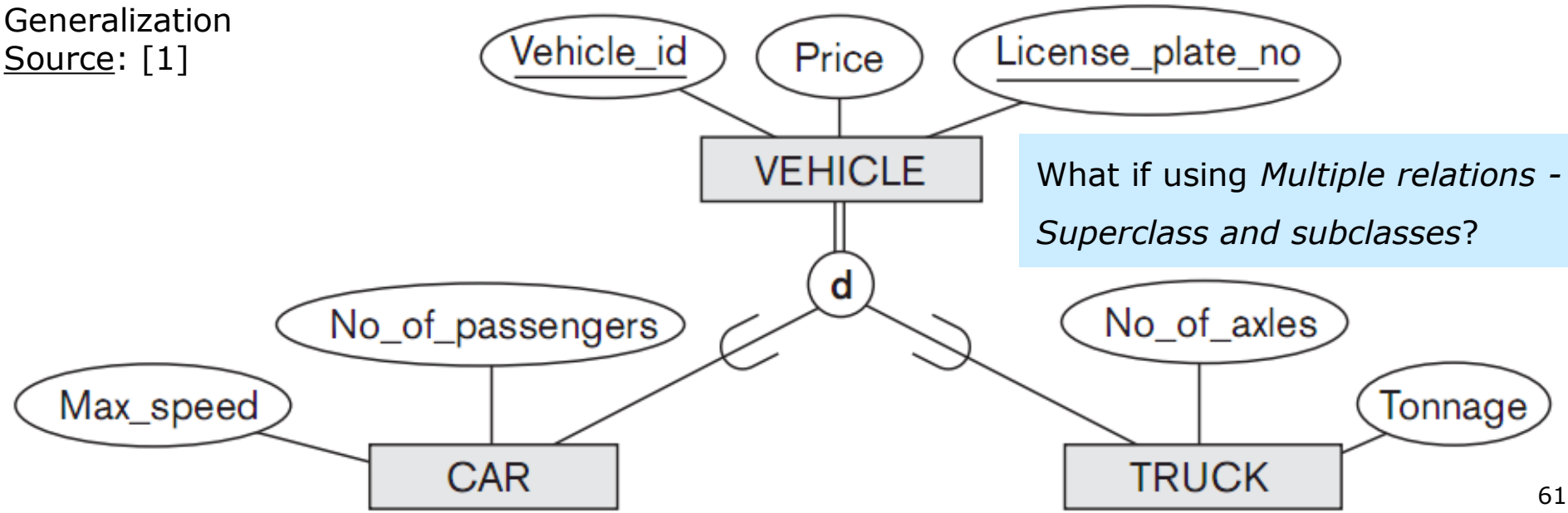
TRUCK (Vehicle_id, No_of_axles, Tonnage)

Foreign key (of CAR, TRUCK): Vehicle_id *to* VEHICLE.Vehicle_id

Check total specialization of VEHICLE.Vehicle_id in CAR and TRUCK

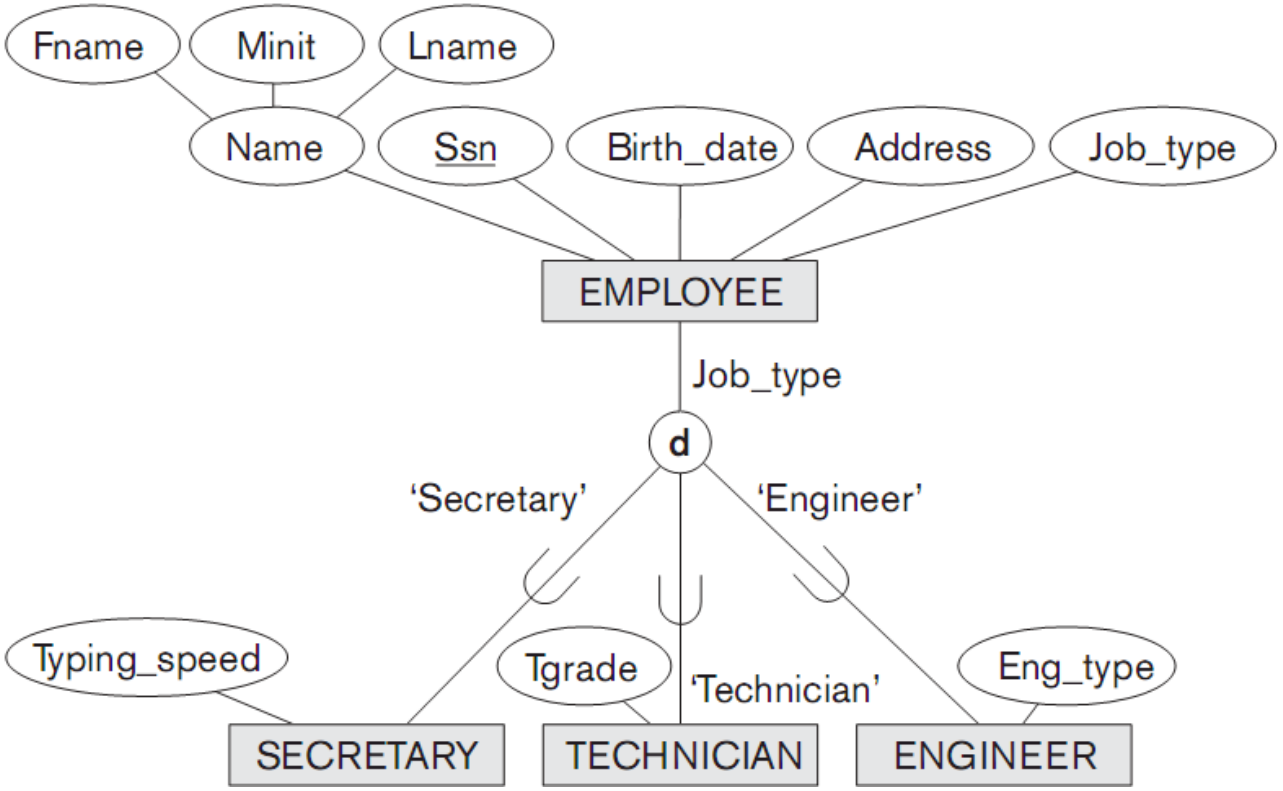
Using **Multiple relations – Superclass and Subclasses (*total*)**

Generalization
Source: [1]



What if using *Multiple relations - Superclass and subclasses*?

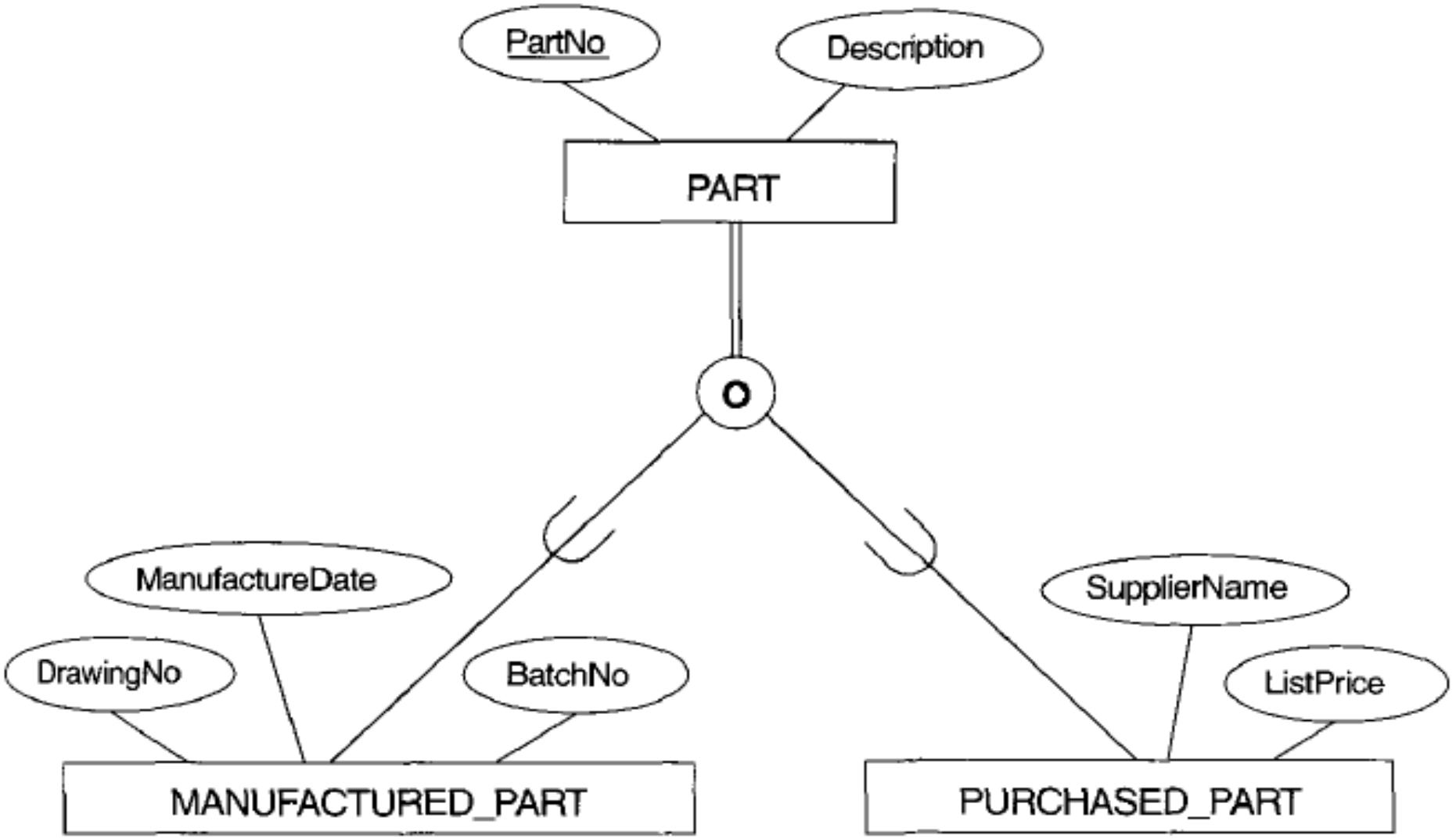
Data model mapping



EMPLOYEE

Ssn	Fname	Minit	Lname	Birth_date	Address	Job_type	Typing_speed	Tgrade	Eng_type

Using a **Single relation with one type attribute (*disjoint*)**

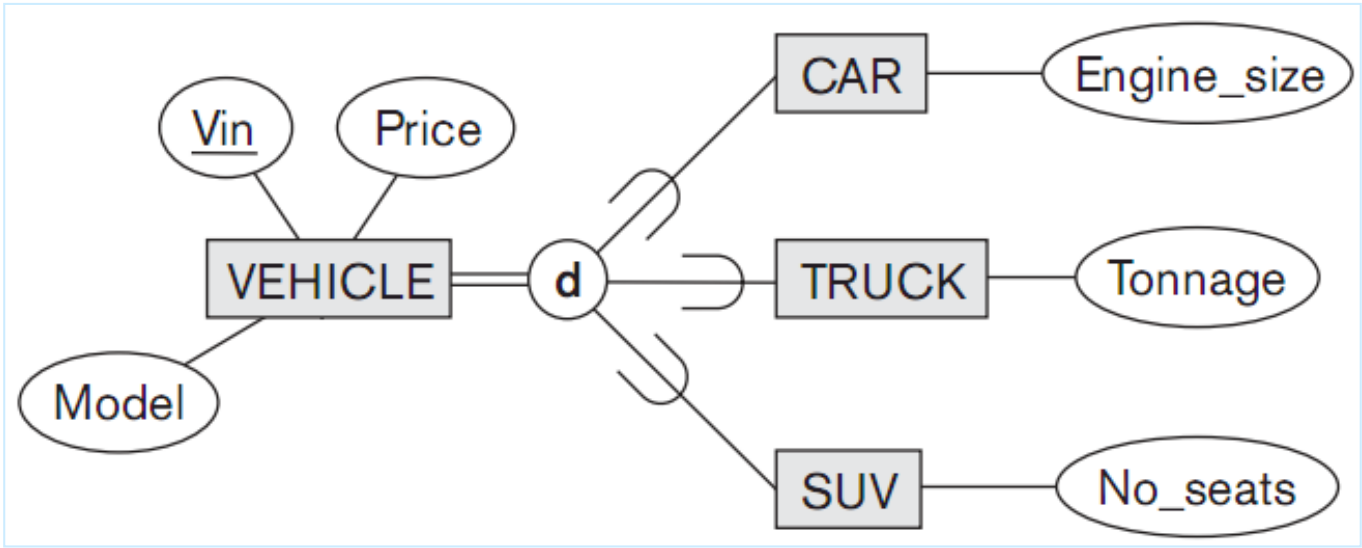


PART

PartNo	Description	MFlag	DrawingNo	ManufactureDate	BatchNo	PFlag	SupplierName	ListPrice

Using a **Single relation with multiple type attributes** with Boolean fields MFlag and PFlag.
Source: [1]

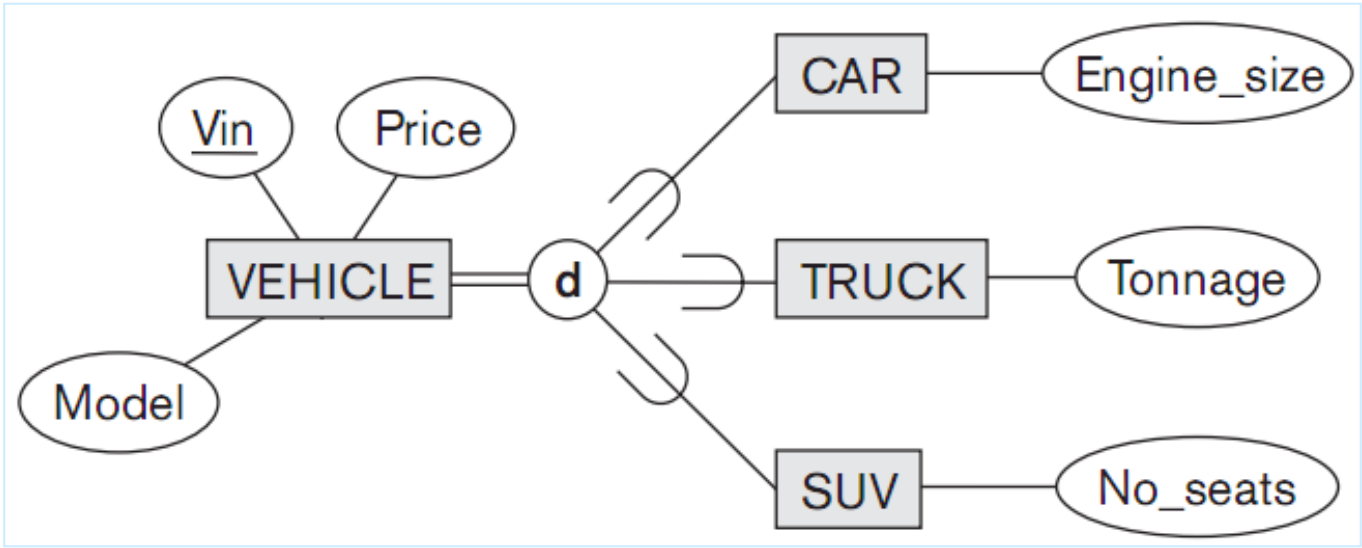
Data model mapping



Describe specialization/ generalization in this example.

Using different options to map this example into relation schemas.

Data model mapping



Using **multiple relations – superclass and subclasses**

VEHICLE (Vin, Model, Price)

CAR (Vin, Engine_size)

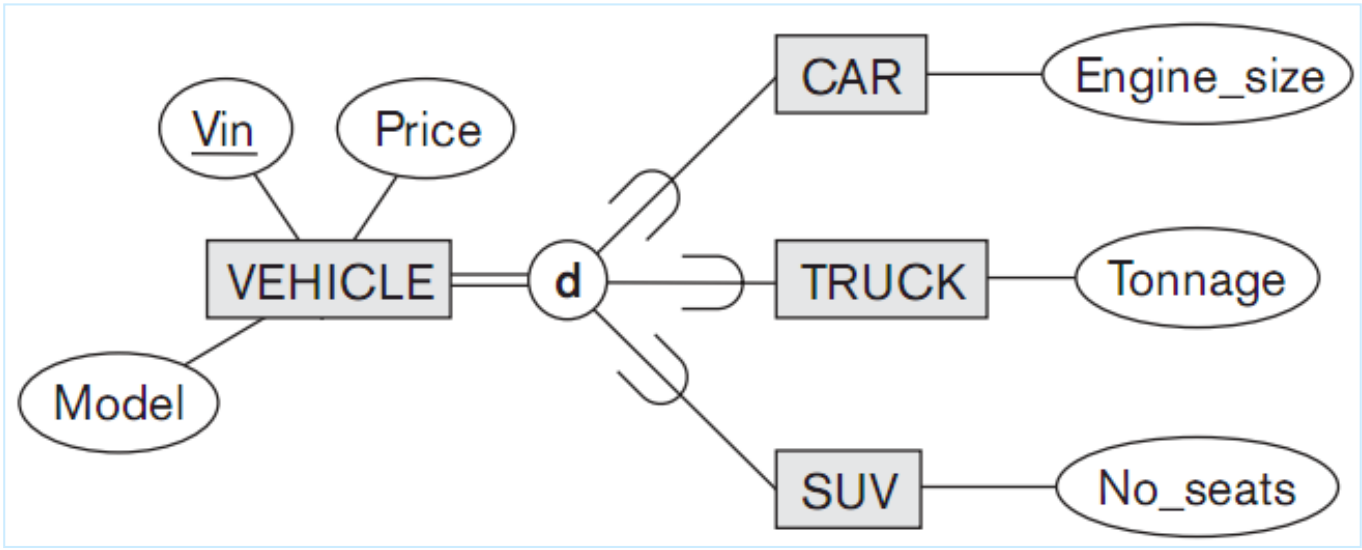
TRUCK (Vin, Tonnage)

SUV (Vin, No_seats)

Foreign key (of CAR, TRUCK, SUV): Vin *to* VEHICLE.Vin

Check total specialization of VEHICLE.Vin in CAR, TRUCK, and SUV

Data model mapping



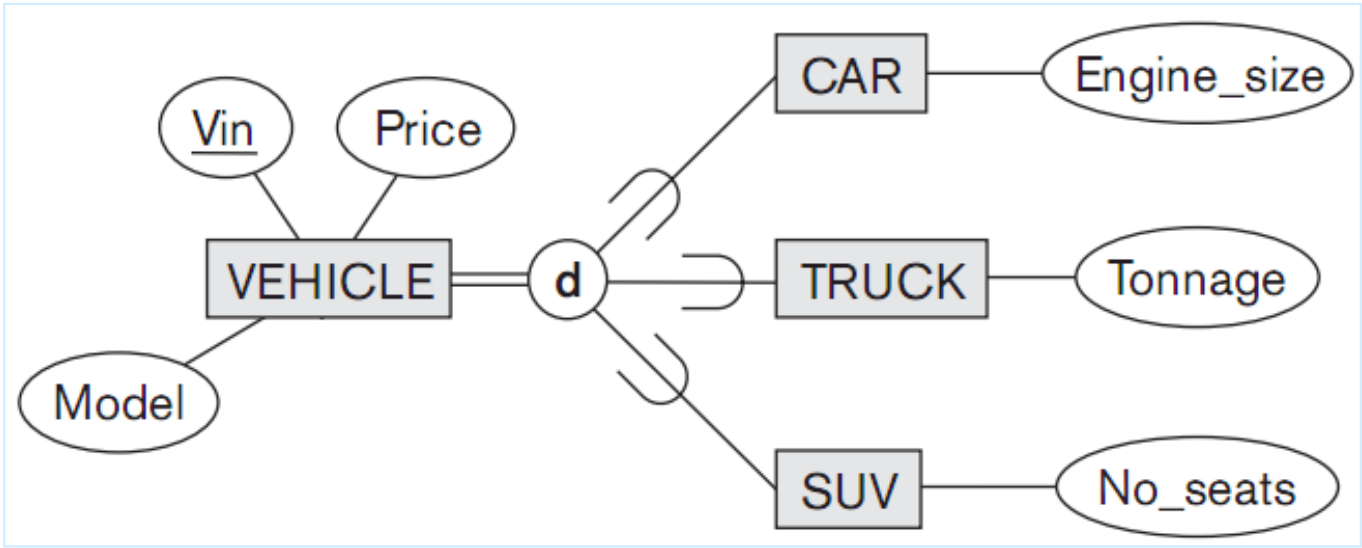
Using **multiple relations – subclasses only (*total*)**

CAR (Vin, Engine_size, Model, Price)

TRUCK (Vin, Tonnage, Model, Price)

SUV (Vin, No_seats, Model, Price)

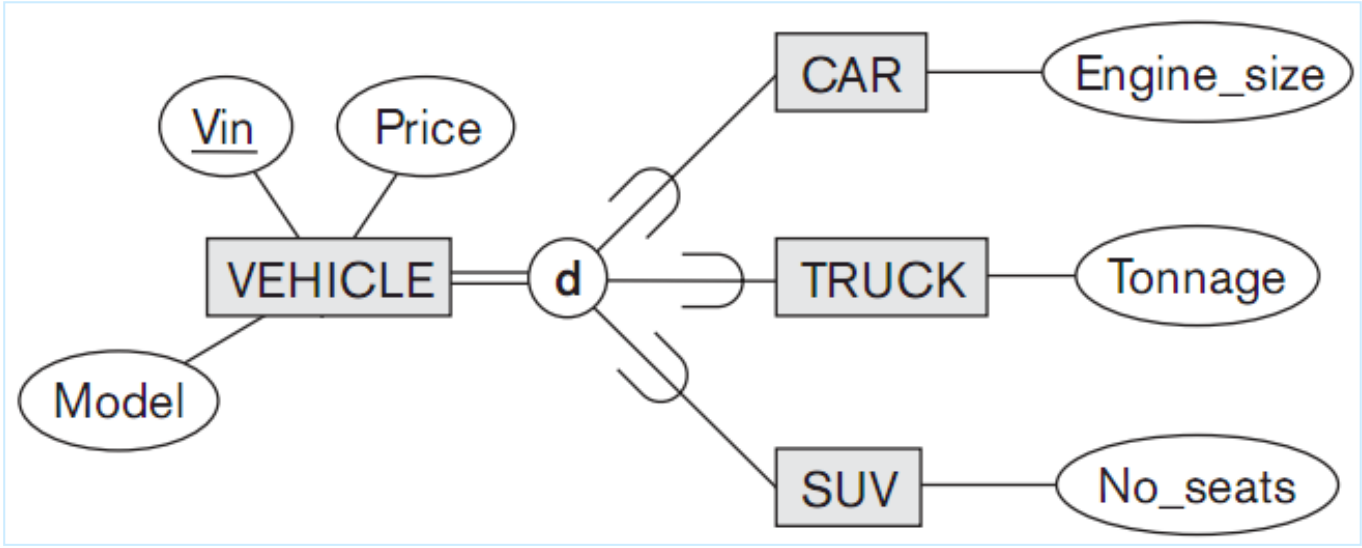
Data model mapping



Using a **single relation with one type attribute (*disjoint*)**

Vehicle (Vin, Model, Price, VehicleType, Engine_size, Tonnage, No_seats)

Data model mapping



Using a **single relation with multiple attributes. Why not?**

Vehicle (Vin, Model, Price, CarType, TruckType, SUVType, Engine_size, Tonnage, No_seats)

⇩

Vehicle (Vin, Model, Price, ~~CarType, TruckType, SUVType~~, Engine_size, Tonnage, No_seats)

VehicleType

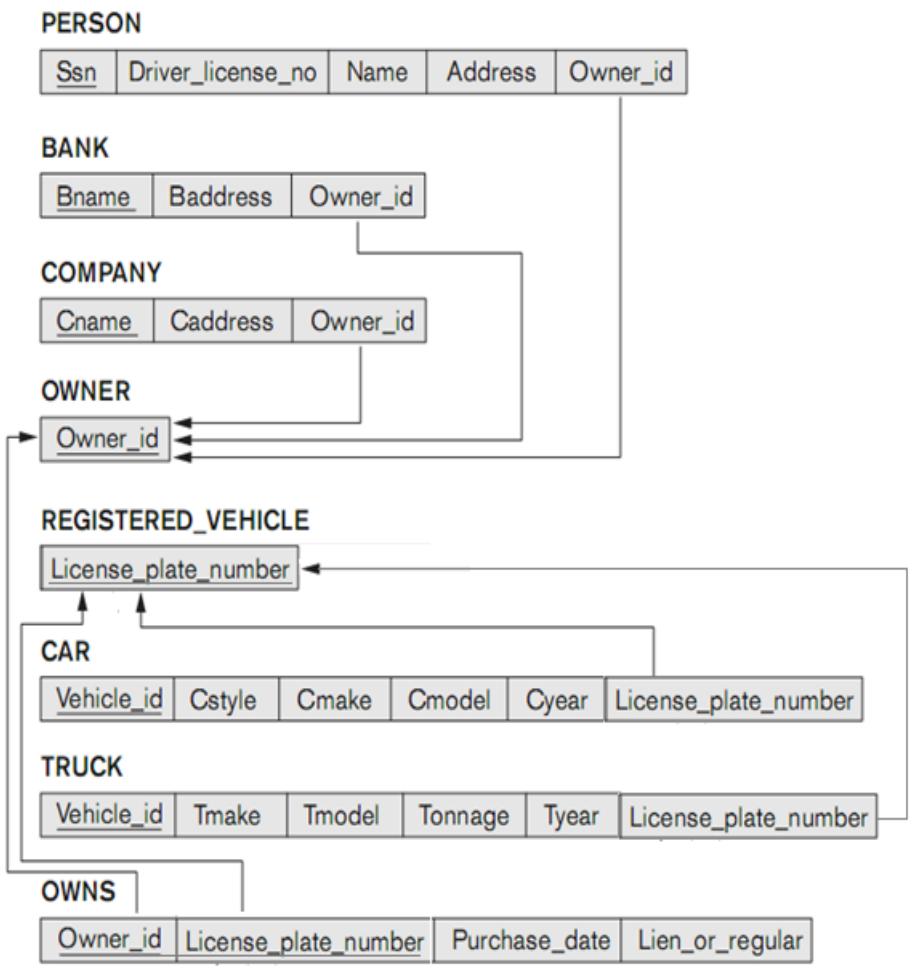
Data model mapping

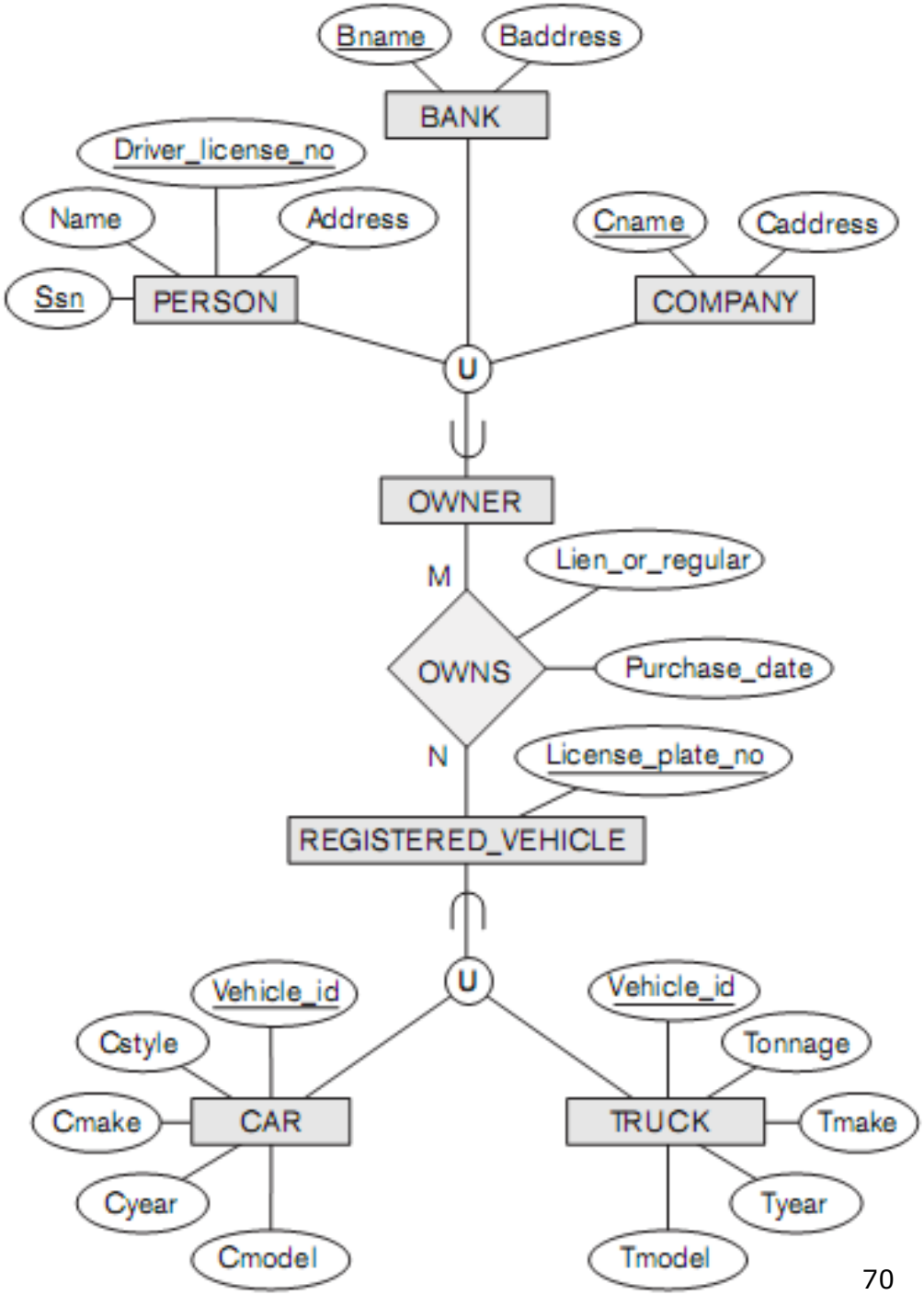
- Step 9: Mapping of Union Types (Categories)
  - For mapping a category whose defining superclasses have different keys, it is customary (traditional) to specify a new key attribute, called a *surrogate key*, when creating *a relation* to correspond to the *category*.
  - Include the surrogate key attribute as *foreign key* in each relation corresponding to a *superclass* of the category, to specify the correspondence in values between the surrogate key and the key of each superclass.
  - For a category whose *superclasses* have the *same key*, there is *no need for a surrogate key*.

Two categories (union types):

OWNER and REGISTERED_VEHICLE

Source: [1]





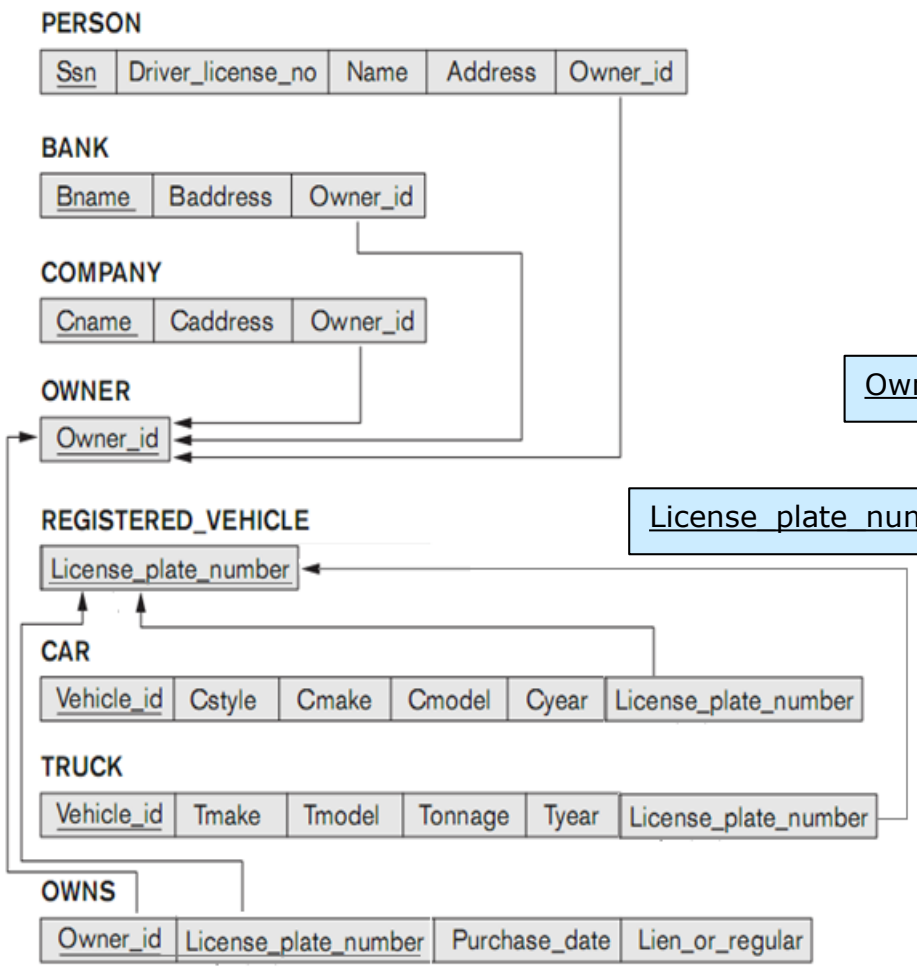
Mapping the EER categories

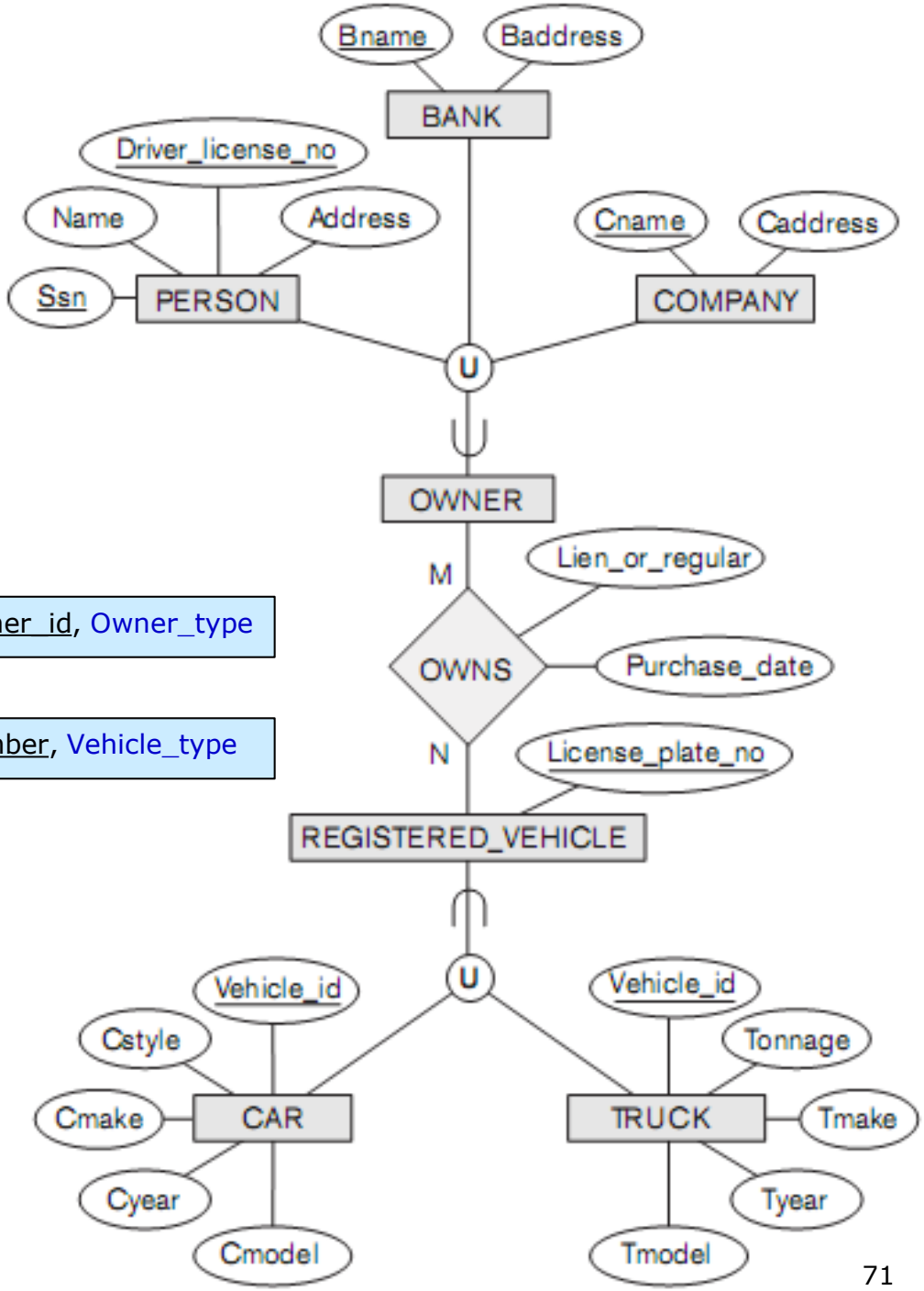
Source: [1]

Two categories (union types):

OWNER and REGISTERED_VEHICLE

Source: [1]





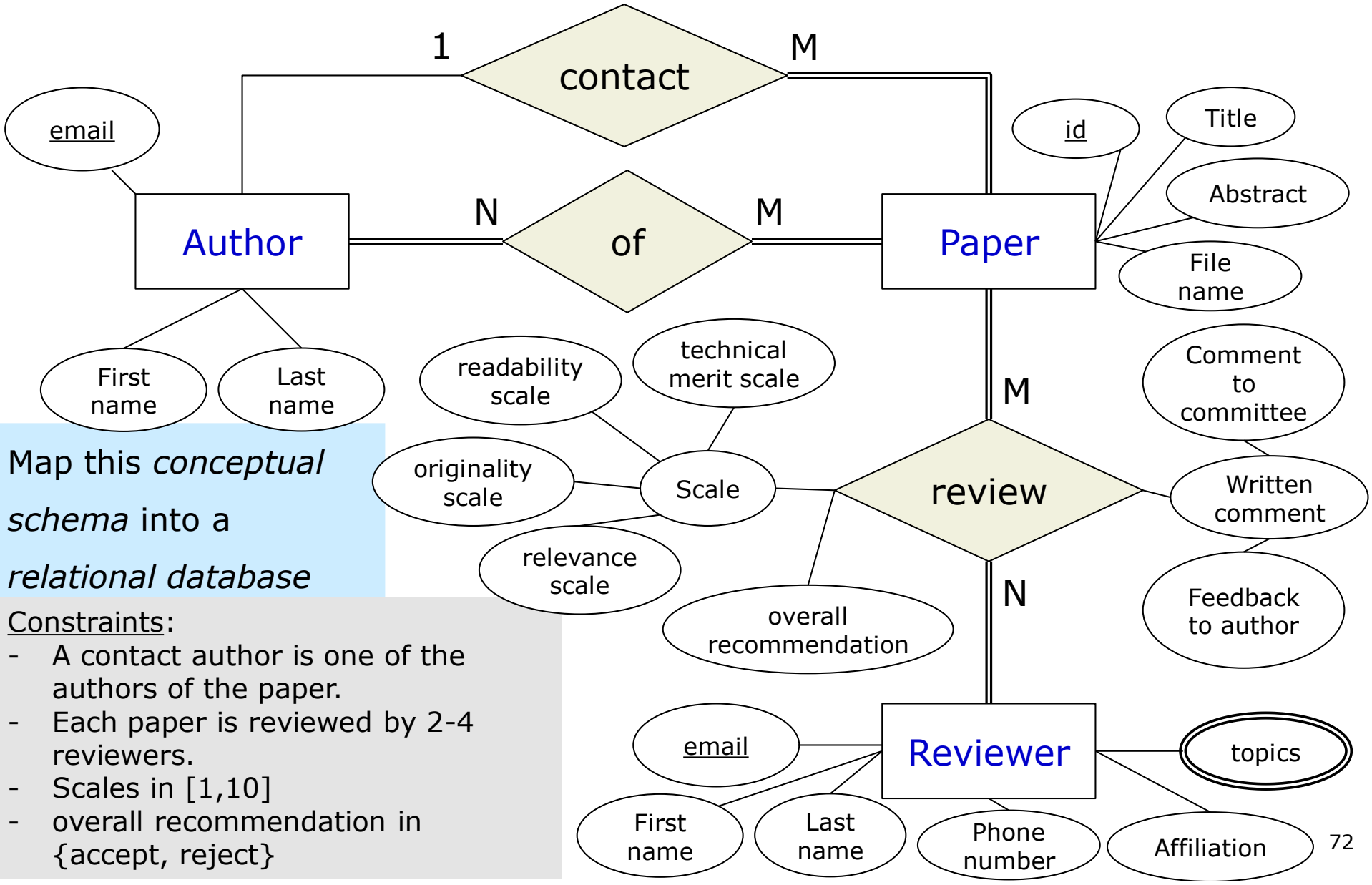
Owner_id, Owner_type

License_plate_number, Vehicle_type

Mapping the EER categories

Source: [1]

The Entity–Relationship diagram for the CONFERENCE_REVIEW database

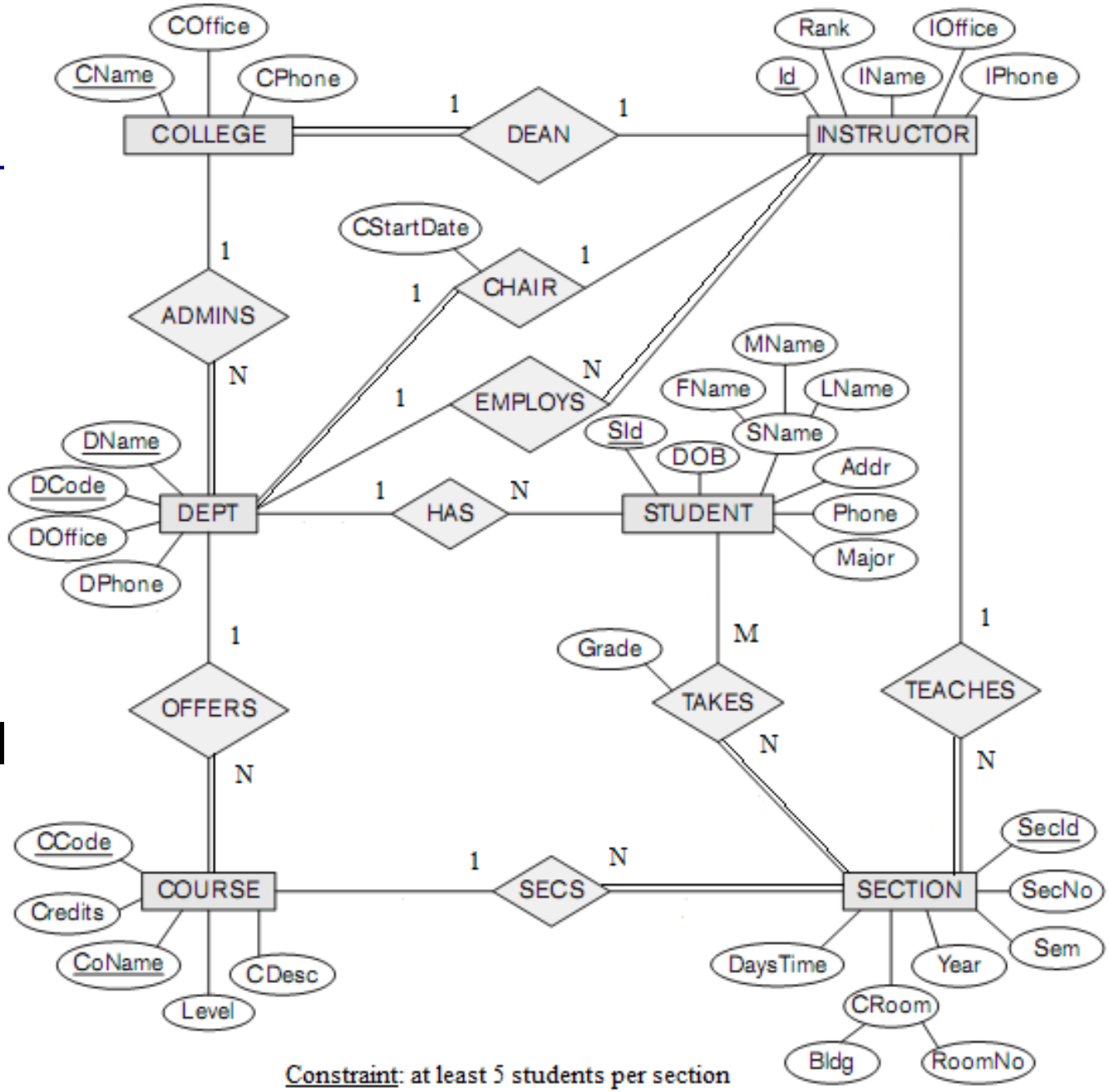


Map this *conceptual schema* into a *relational database*

Constraints:
- A contact author is one of the authors of the paper.
- Each paper is reviewed by 2-4 reviewers.
- Scales in [1,10]
- overall recommendation in {accept, reject}

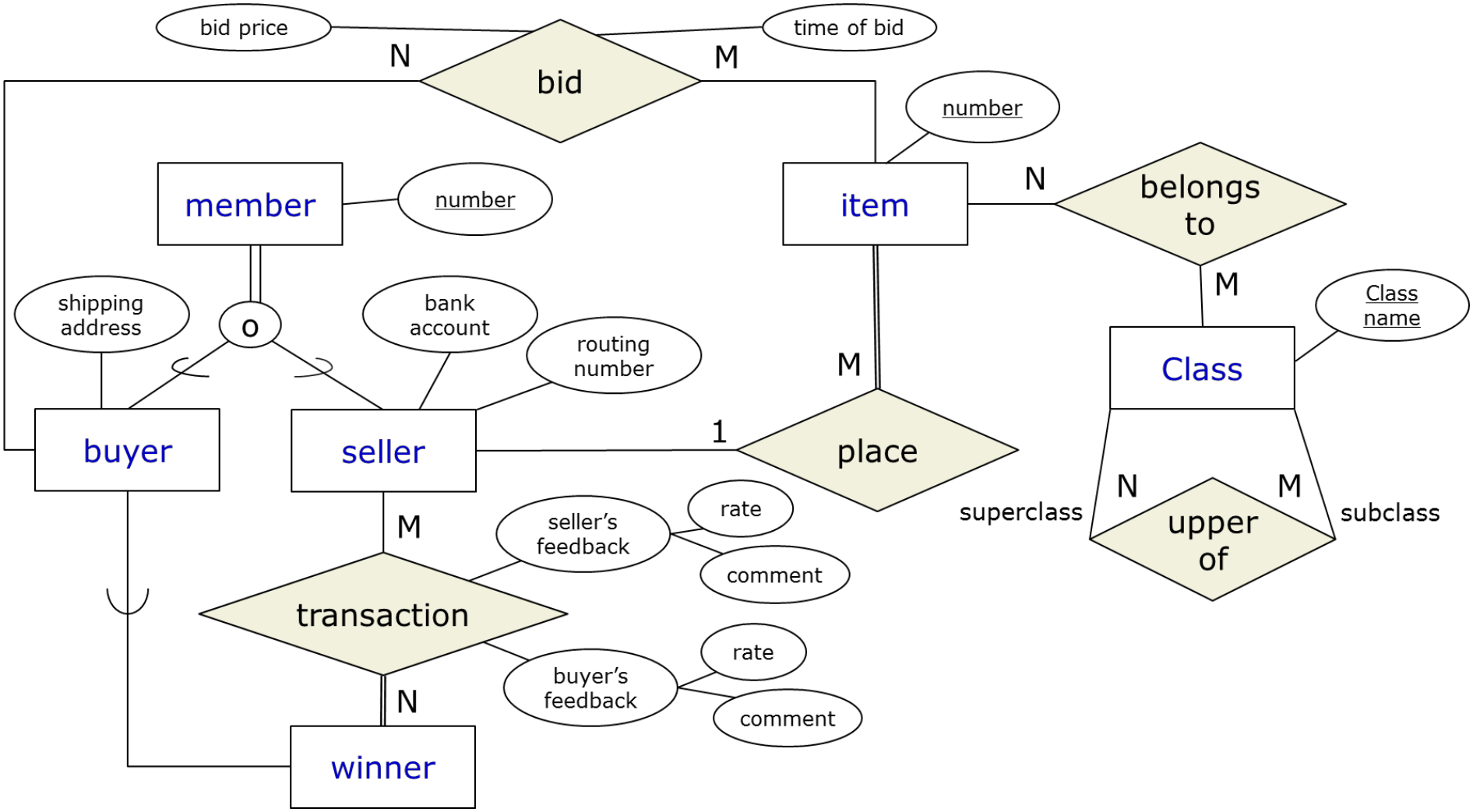
Data Modeling Mapping

- Given an ER schema of a University database, map this schema into a relational database schema.



A University database schema
Source: [1]

The EER diagram for the ONLINE_AUCTION database system in which members (buyers and sellers) participate in the sale of items

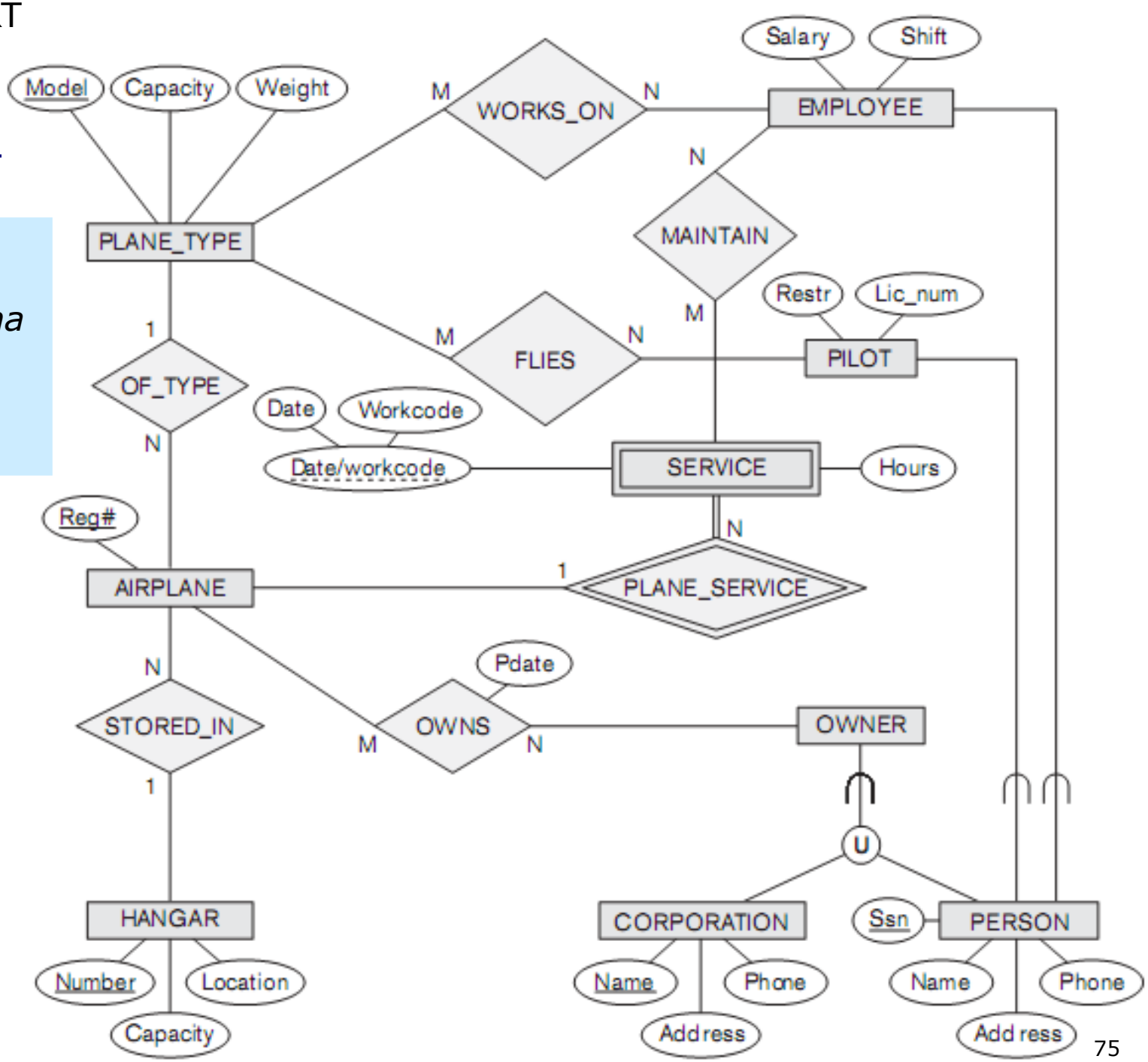


Map this *conceptual schema* into a *relational database schema*

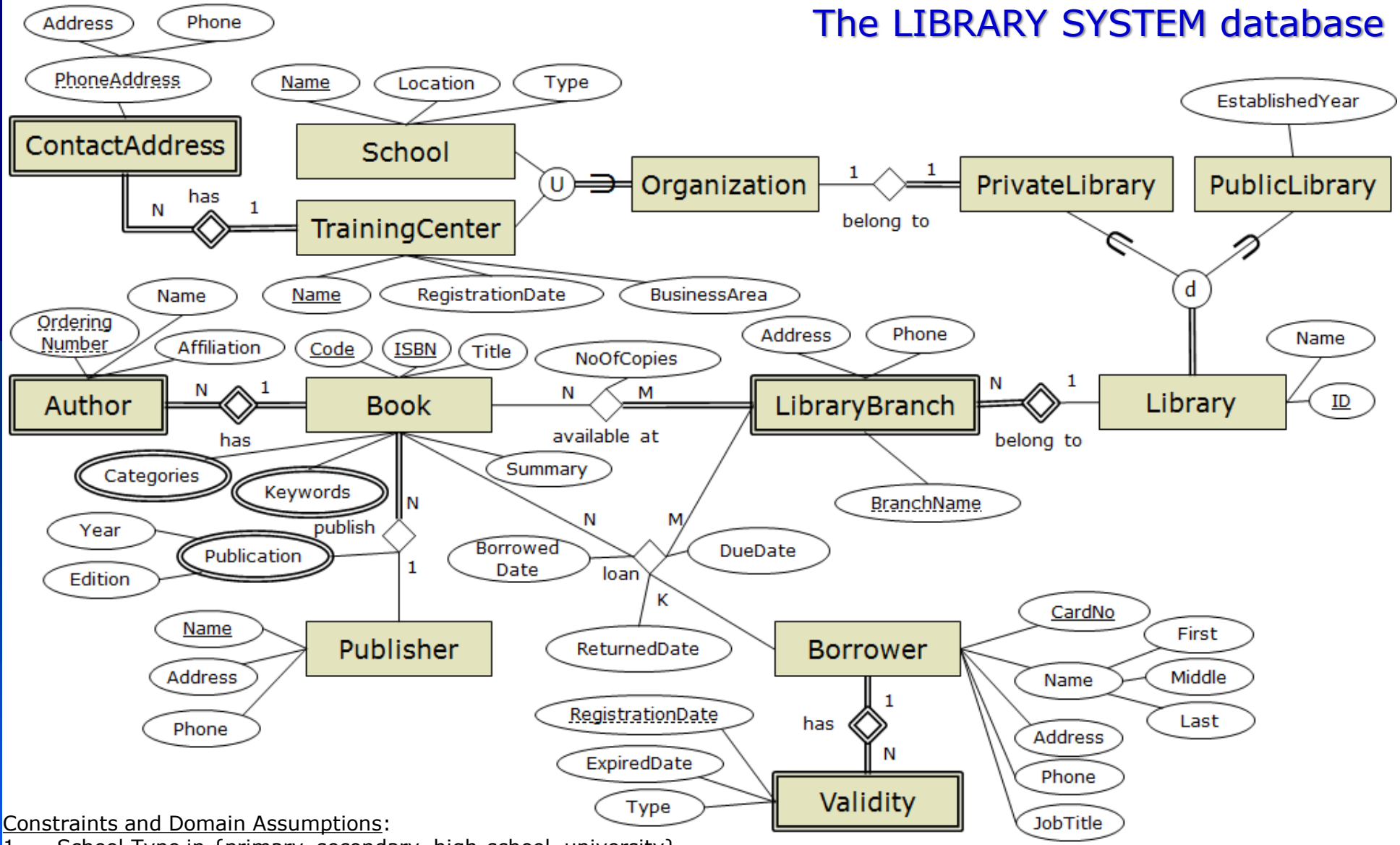
A SMALL_AIRPORT database schema

Source: [1]

Map this *conceptual schema* into a *relational database schema*



The LIBRARY SYSTEM database



Constraints and Domain Assumptions:

1. School.Type in {primary, secondary, high-school, university}
2. Each library branch has at least two copies per book.
3. Book.Code is composed of 12 digits.
4. DueDate – BorrowedDate = a month.
5. The maximum number of books that a borrower can borrow from a library branch is 5 for any date.
6. If a borrower returns more than 3 books late, his/ her card becomes invalid.
7. Year, Edition, EstablishedYear are positive integers.
8. TrainingCenter.RegistrationDate, BorrowedDate, DueDate, ReturnedDate, Validity.RegistrationDate, ExpiredDate are DATE values.
9. The rest including the attributes not mentioned above are STRING values.

Map this *conceptual schema* into a *relational database schema*

A sample database application - COMPANY

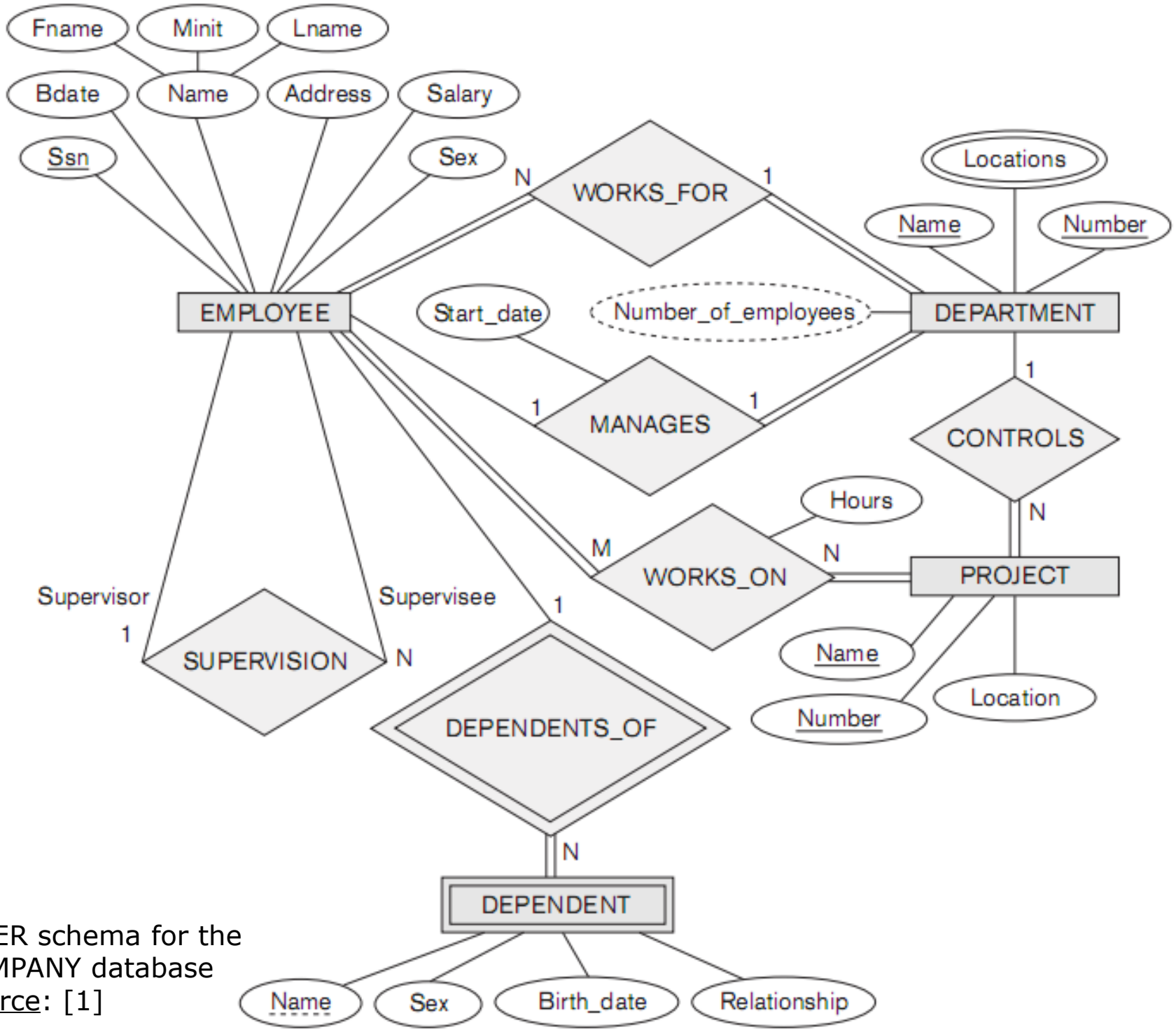
The COMPANY database keeps track of a company's employees, departments, and projects. Suppose that after the requirements collection and analysis phase, the database designers provide the following description of the miniworld that will be represented in the database:

- The company is organized into departments. Each department has a unique name, a unique number, and a particular employee who manages the department. We keep track of the start date when that employee began managing the department. A department may have several locations.

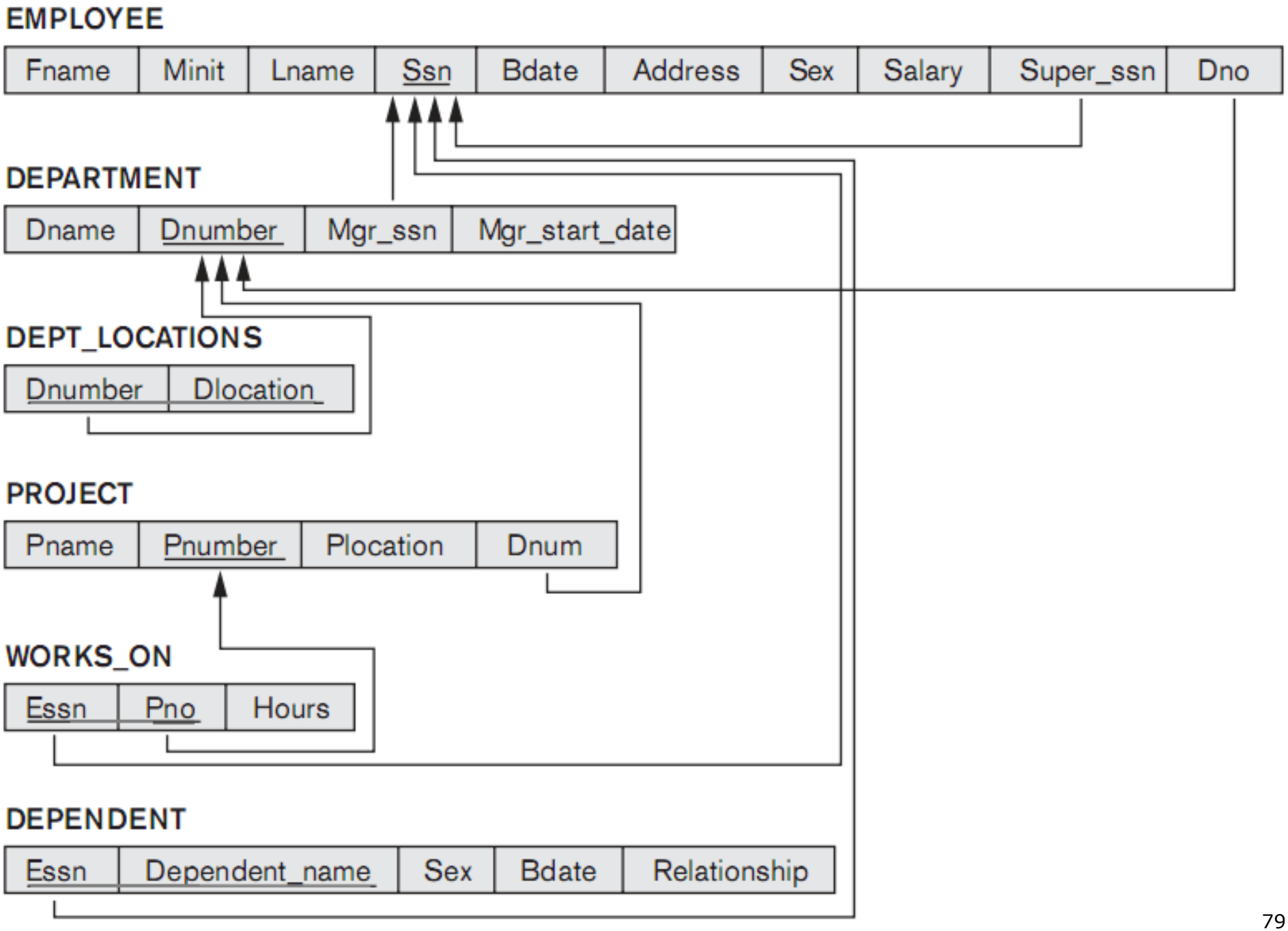
- A department controls a number of projects, each of which has a unique name, a unique number, and a single location.

- The database will store each employee's name, Social Security number (SSN), address, salary, sex (gender), and birth date. An employee is assigned to one department, but may work on several projects, which are not necessarily controlled by the same department. It is required to keep track of the current number of hours per week that an employee works on each project, as well as the direct supervisor of each employee (who is another employee).

- The database will keep track of the dependents of each employee for insurance purposes, including each dependent's first name, sex, birth date, and relationship to the employee.



An ER schema for the COMPANY database
Source: [1]


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

```
CREATE SCHEMA COMPANY;

CREATE TABLE EMPLOYEE (
        FNAME           VARCHAR(15)     NOT NULL,
        MINIT           CHAR,
        LNAME           VARCHAR(15)     NOT NULL,
        SSN             CHAR(9),
        BDATE           DATE,
        ADDRESS         VARCHAR(30),
        SEX             CHAR,
        SALARY          DECIMAL(10,2),
        SUPER_SSN       CHAR(9),
        DNO             INT             NOT NULL,
        PRIMARY KEY (SSN),
        FOREIGN KEY (SUPER_SSN) REFERENCES EMPLOYEE (SSN)
);

CREATE TABLE DEPENDENT (
        ESSN            CHAR(9),
        DEPENDENT_NAME  VARCHAR(15),
        SEX             CHAR,
        BDATE           DATE,
        RELATIONSHIP    VARCHAR(8),
        PRIMARY KEY (ESSN, DEPENDENT_NAME),
        FOREIGN KEY (ESSN) REFERENCES EMPLOYEE (SSN)
);
```

```
CREATE TABLE DEPARTMENT (
        DNAME           VARCHAR(15)     NOT NULL,
        DNUMBER         INT,
        MGR_SSN         CHAR(9)         NOT NULL,
        MGR_STAR_TDATE  DATE,
        PRIMARY KEY (DNUMBER),
        UNIQUE (DNAME),
        FOREIGN KEY (MGR_SSN) REFERENCES EMPLOYEE (SSN)
);

CREATE TABLE DEPT_LOCATIONS (
        DNUMBER         INT,
        DLOCATION       VARCHAR(15),
        PRIMARY KEY (DNUMBER, DLOCATION),
        FOREIGN KEY (DNUMBER) REFERENCES DEPARTMENT(DNUMBER)
);

ALTER TABLE EMPLOYEE
ADD FOREIGN KEY (DNO) REFERENCES DEPARTMENT (DNUMBER);
```

```
CREATE TABLE PROJECT (
        PNAME           VARCHAR(15)     NOT NULL,
        PNUMBER         INT,
        PLOCATION       VARCHAR(15),
        DNUM            INT             NOT NULL,
        PRIMARY KEY (PNUMBER),
        UNIQUE (PNAME),
        FOREIGN KEY (DNUM) REFERENCES DEPARTMENT (DNUMBER)
);

CREATE TABLE WORKS_ON (
        ESSN    CHAR(9),
        PNO     INT,
        HOURS   DECIMAL(3,1),
        PRIMARY KEY (ESSN, PNO),
        FOREIGN KEY (ESSN) REFERENCES EMPLOYEE (SSN),
        FOREIGN KEY (PNO) REFERENCES PROJECT (PNUMBER)
);
```

The COMPANY database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

Source: [1]

Part of the COMPANY database

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

How to *populate data* into a relational database?

How to *remove data* from a relational database?

How to *modify data* in a relational database?

How to *extract data* from a relational database?

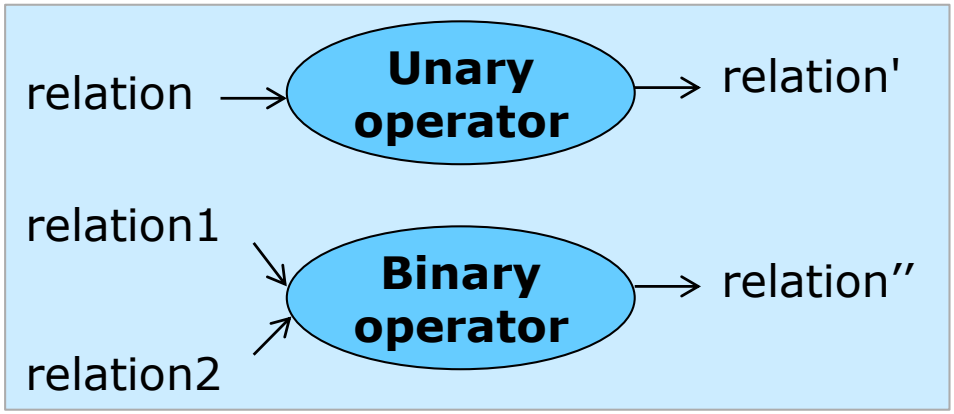
How to maintain the *consistency* of a relational database with respect to its *constraints*?

The Relational Algebra

- The relational algebra is the *formal* language for the relational data model.
  - Operations to specify basic retrieval requests as relational algebra expressions, reflecting *how* a request is resolved
  - A relational algebra expression produces a relation that represents the result of a query (or retrieval request).
- Why is the relational algebra studied?
  - It provides a formal foundation for relational model operations.
  - It is used as a basis for implementing and optimizing queries in the query processing and optimization modules of relational database management systems (RDBMSs).
  - Some of its concepts are incorporated into the SQL standard query language for RDBMSs.
  - The core operations and functions in the internal modules of most RDBMSs are based on relational algebra operations.
    - Internal representation of a query is a query tree based on its corresponding relational algebra expressions.

The Relational Algebra

- The Relational Algebra
  - The basic set of operations that specify basic retrieval requests
    - The result of each operation is a new relation that has been formed from one or many relations.
    - As a relation, it can be then used as an input of other operations.
    - Evaluation is from the inner to the outer of an operation sequence.
  - Operations:
    - $\sigma_{<conditions>}$, $\pi_{<an\ attribute\ list>}$
    - x, $\bowtie_{<conditions>}$, *
    - Outer joins: $\bowtie$ $\bowtie$ $\bowtie$
    - Semi-join: $\ltimes$
    - Anti-join: $\overline{\ltimes}$
    - ∪, ∩, -, ÷
    - *<a grouping attribute list>* $\Im$ *<function list>*,
    - Rename: ρ



OPERATION	PURPOSE	NOTATION
SELECT	Selects all tuples that satisfy the selection condition from a relation R.	$\sigma_{\text{<selection condition>}}(R)$
PROJECT	Produces a new relation with only some of the attributes of R, and removes duplicate tuples.	$\pi_{\text{<attribute list>}}(R)$
THETA JOIN	Produces all combinations of tuples from R_1 and R_2 that satisfy the join condition.	$R_1 \bowtie_{\text{<join condition>}} R_2$
EQUIJOIN	Produces all the combinations of tuples from R_1 and R_2 that satisfy a join condition with only equality comparisons.	$R_1 \bowtie_{\text{<join condition>}} R_2$, OR $R_1 \bowtie_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$
NATURAL JOIN	Same as EQUIJOIN except that the join attributes of R_2 are not included in the resulting relation; if the join attributes have the same names, they do not have to be specified at all.	$R_1 *_{\text{<join condition>}} R_2$, OR $R_1 *_{\text{(<join attributes 1>), (<join attributes 2>)}} R_2$ OR $R_1 * R_2$
UNION	Produces a relation that includes all the tuples in R_1 or R_2 or both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cup R_2$
INTERSECTION	Produces a relation that includes all the tuples in both R_1 and R_2; R_1 and R_2 must be union compatible.	$R_1 \cap R_2$
DIFFERENCE	Produces a relation that includes all the tuples in R_1 that are not in R_2; R_1 and R_2 must be union compatible.	$R_1 - R_2$
CARTESIAN PRODUCT	Produces a relation that has the attributes of R_1 and R_2 and includes as tuples all possible combinations of tuples from R_1 and R_2.	$R_1 \times R_2$
DIVISION	Produces a relation $R(X)$ that includes all tuples $t[X]$ in $R_1(Z)$ that appear in R_1 in combination with every tuple from $R_2(Y)$, where $Z = X \cup Y$.	$R_1(Z) \div R_2(Y)$

The Relational Algebra

- A *Complete* Set of Relational Algebra Operations :
  - $\{\sigma, \pi, \cup, \rho, -, \times\}$
    - σ = select
    - π = project
    - $\cup$ = union
    - ρ = rename
    - $-$ = minus (difference)
    - $\times$ = Cartesian product

The Relational Algebra

- The RENAME operator: ρ (*rho*), a unary operator applied to a relation *R (A1, A2, …, An)* of degree *n* to rename either the relation name or the attribute names, or both
  - Change relation name: $\rho_S(R)$

Example 1: Employee (SSN, Fname, Lname, Dno) *is changed to:*

Emp (SSN, Fname, Lname, Dno)

$$\rho_{Emp}(\text{Employee})$$

  - Change attribute names: $\rho_{(B1, B2, \ldots, Bn)}(R)$

Example 2: Employee (SSN, Fname, Lname, Dno) *is changed to:*

Employee (SSNumber, F_name, L_name, Dnumber)

$$\rho_{(\text{SSNumber}, F\_name, \text{L\_name}, \text{Dnumber})}(\text{Employee})$$

  - Change both attribute and relation name: $\rho_{S(B1, B2, \ldots, Bn)}(R)$

Example 3: Employee (SSN, Fname, Lname, Dno) *is changed to:*

Emp (SSNumber, F_name, L_name, Dnumber)

$$\rho_{\text{Emp}(\text{SSNumber}, F\_name, \text{L\_name}, \text{Dnumber})}(\text{Employee})$$

The Relational Algebra

- **SELECT**: $\sigma_{<conditions>}(R)$
  - Retrieve the tuples in relation R that satisfy <*conditions*>.

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

Retrieve all the information of each employee who is with department 4 and salary > 25000 or with department 5 and salary > 30000.

$\sigma_{\mathbf{(Dno=4\ AND\ Salary>25000)\ OR\ (Dno=5\ AND\ Salary>30000)}}\mathbf{(EMPLOYEE)}$

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5

The Relational Algebra

- SELECT: $\sigma_{<conditions>}(R)$
  - Retrieve the tuples in relation R that satisfy <*conditions*>.
  - The relation S = $\sigma_{<conditions>}(R)$ has the same schema (same attributes) as R.
  - A cascade (sequence) of SELECT operations in any order:
    - $\sigma_{<cond1>}(\sigma_{<cond2>}(R)) = \sigma_{<cond1> \text{ AND } < cond2>}(R)$
  - SELECT is commutative:
    - $\sigma_{<condition1>}(\sigma_{<condition2>}(R)) = \sigma_{<condition2>}(\sigma_{<condition1>}(R))$
  - The number of tuples in the result of SELECT is less than or equal to the number of tuples in the input relation R

Retrieve all the information of each employee who is with department 4 and salary > 25000.

$\sigma_{\mathbf{Dno=4 \text{ AND } Salary>25000}}(\mathbf{EMPLOYEE})$
$= \sigma_{\mathbf{Dno=4}}(\sigma_{\mathbf{Salary>25000}}(\mathbf{EMPLOYEE}))$
$= \sigma_{\mathbf{Salary>25000}}(\sigma_{\mathbf{Dno=4}}(\mathbf{EMPLOYEE}))$

The Relational Algebra

- **PROJECT**: $\pi_{<Attribute\ list>}(R)$
  - Return a relation of distinct tuples from relation R with the attributes listed in *<Attribute list>*

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07			25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03			25000	987654321	4
James	E	Borg	888665555	1937-11			55000	NULL	1

Retrieve Ssn, date of birth, and department number of each employee.

$\pi_{Ssn,\ Bdate,\ Dno}$ **(EMPLOYEE)**

Ssn	Bdate	Dno
123456789	1965-01-09	5
333445555	1955-12-08	5
999887777	1968-01-19	4
987654321	1941-06-20	4
666884444	1962-09-15	5
453453453	1972-07-31	5
987987987	1969-03-29	4
888665555	1937-11-10	1

π_{Dno} **(EMPLOYEE)**

Dno
5
4
1

The Relational Algebra

- CARTESIAN PRODUCT (CROSS JOIN): R x S
  - Produce a new relation *T* by combining every member (tuple) from one relation (set) with every member (tuple) from the other relation (set)
    - Degree of *T* = Degree of R + Degree of S
    - Cardinality of *T* = |T| = |R|*|S|

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Return all the combinations of departments and their locations.
DEPARTMENT x DEPT_LOCATIONS?

The Relational Algebra

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

DEPARTMENT x DEPT_LOCATIONS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	1	Houston
Research	5	333445555	1988-05-22	4	Stafford
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	1	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Administration	4	987654321	1995-01-01	5	Bellaire
Administration	4	987654321	1995-01-01	5	Sugarland
Administration	4	987654321	1995-01-01	5	Houston
Headquarters	1	888665555	1981-06-19	1	Houston
Headquarters	1	888665555	1981-06-19	4	Stafford
Headquarters	1	888665555	1981-06-19	5	Bellaire
Headquarters	1	888665555	1981-06-19	5	Sugarland
Headquarters	1	888665555	1981-06-19	5	Houston

The Relational Algebra

- **THETA JOIN**: $R \bowtie_{<conditions>} S$
  - Produce a new relation Q with $n + m$ attributes Q (A_1, A_2, … , A_n, B_1, B_2, … , B_m) in that order; Q has one tuple for each combination of tuples—one from R (A_1, A_2, … , A_n) and one from S (B_1, B_2, … , B_m)—whenever the combination satisfies the join condition.
  - $R \bowtie_{<conditions>} S = \sigma_{<conditions>} (R \times S)$

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Return all the combinations of Research department with Houston location.
DEPARTMENT $\bowtie_{\text{Dname = 'Research' AND Dlocation = 'Houston'}}$ **DEPT_LOCATIONS?**

The Relational Algebra

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

DEPARTMENT x DEPT_LOCATIONS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	1	Houston
Research	5	333445555	1988-05-22	4	Stafford
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	1	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Administration	4	987654321	1995-01-01	5	Bellaire
Administration	4	987654321	1995-01-01	5	Sugarland
Administration	4	987654321	1995-01-01	5	Houston
Headquarters	1	888665555	1981-06-19	1	Houston
Headquarters	1	888665555	1981-06-19	4	Stafford
Headquarters	1	888665555	1981-06-19	5	Bellaire
Headquarters	1	888665555	1981-06-19	5	Sugarland
Headquarters	1	888665555	1981-06-19	5	Houston

The Relational Algebra

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Return all the combinations of Research department with Houston location.

DEPARTMENT $\bowtie_{\text{Dname = 'Research' AND Dlocation = 'Houston'}}$ **DEPT_LOCATIONS**

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	1	Houston
Research	5	333445555	1988-05-22	5	Houston

The Relational Algebra

- EQUIJOIN: THETA JOIN with *equality* (=) comparisons only.
  - $R \bowtie_{A=B} S = \sigma_{A=B}(R \times S)$, where A are attributes in R and B are attributes in S.

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT $\bowtie_{\text{Dnumber = Dnumber}}$ **DEPT_LOCATIONS?**

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT $\bowtie_{\text{Dnumber = Dnumber}}$ **DEPT_LOCATIONS?**

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	1	Houston
Research	5	333445555	1988-05-22	4	Stafford
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	1	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Administration	4	987654321	1995-01-01	5	Bellaire
Administration	4	987654321	1995-01-01	5	Sugarland
Administration	4	987654321	1995-01-01	5	Houston
Headquarters	1	888665555	1981-06-19	1	Houston
Headquarters	1	888665555	1981-06-19	4	Stafford
Headquarters	1	888665555	1981-06-19	5	Bellaire
Headquarters	1	888665555	1981-06-19	5	Sugarland
Headquarters	1	888665555	1981-06-19	5	Houston

The Relational Algebra

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT $\bowtie_{\text{Dnumber = Dnumber}}$ **DEPT_LOCATIONS**

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Headquarters	1	888665555	1981-06-19	1	Houston

The Relational Algebra

- NATURAL JOIN: EQUIJOIN by *getting rid of the second attribute* in an equality condition when the two join attributes (or each pair of join attributes) have the ***same name*** in both relations
  - R * S

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT * DEPT_LOCATIONS?

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT $\bowtie_{\text{Dnumber = Dnumber}}$ **DEPT_LOCATIONS**

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dnumber	Dlocation
Research	5	333445555	1988-05-22	5	Bellaire
Research	5	333445555	1988-05-22	5	Sugarland
Research	5	333445555	1988-05-22	5	Houston
Administration	4	987654321	1995-01-01	4	Stafford
Headquarters	1	888665555	1981-06-19	1	Houston

Retrieve all the information of each department and its locations.
DEPARTMENT * DEPT_LOCATIONS

Dname	Dnumber	Mgr_ssn	Mgr_start_date	Dlocation
Research	5	333445555	1988-05-22	Bellaire
Research	5	333445555	1988-05-22	Sugarland
Research	5	333445555	1988-05-22	Houston
Administration	4	987654321	1995-01-01	Stafford
Headquarters	1	888665555	1981-06-19	Houston

From *equijoin* to *natural join*

The Relational Algebra

- OUTER JOIN:
  - A set of operations, called *outer joins*, can be used when we want to keep all the tuples in R, or all those in S, or all those in both relations in the result of the JOIN, regardless of whether or not they have matching tuples in the other relation.
    - Left outer join, right outer join, full outer join
  - The join operations where only matching tuples are kept in the result are called *inner joins*.
    - Theta join, equijoin, natural join

The Relational Algebra

- LEFT OUTER JOIN: R ⟕ S
  - In addition to the result of the corresponding inner join, the LEFT OUTER JOIN operation keeps every tuple in the *first*, or *left*, *relation R* in R ⟕ S; if *no* matching tuple is found in S, then the attributes of S in the join result are filled or "padded" with null values.

Return the information of all the employees and their departments that they manage.

EMPLOYEE $\bowtie_{\text{Ssn = Mgr\_ssn}}$ **DEPARTMENT**

Return the information of all the employees and their departments that they manage if any.

EMPLOYEE ⟕$_{\text{Ssn = Mgr\_ssn}}$ **DEPARTMENT**

The Relational Algebra

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

The Relational Algebra

Return the information of all the employees and their departments that they manage.

EMPLOYEE $\bowtie_{\text{Ssn = Mgr\_ssn}}$ **DEPARTMENT**

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno	Dname	Dnumber	Mgr_ssn	Mgr_start_date
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5	Research	5	333445555	1988-05-22
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4	Administration	4	987654321	1995-01-01
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1	Headquaters	1	888665555	1981-06-09

The Relational Algebra

Return the information of all the employees and their departments that they manage if any.

EMPLOYEE $\bowtie_{\text{Ssn = Mgr\_ssn}}$ **DEPARTMENT**

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno	Dname	Dnumber	Mgr_ssn	Mgr_start_date
Franklin	T	Wong	333345555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5	Research	5	333345555	1988-05-22
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4	Administration	4	987654321	1995-01-01
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1	Headquaters	1	888665555	1981-06-09
John	B	Smith	123456789	1965-01-09	732 Fondren, Houston, TX	M	30000	333445555	5	NULL	NULL	NULL	NULL
Alicia	J	Zelaya	999778888	…	…	…	…	…	4	NULL	NULL	NULL	NULL
Ramesh	K	Narayan	…	…	…	…	…	…	5	NULL	NULL	NULL	NULL
Joyce	A	English	…	…	…	…	…	…	5	NULL	NULL	NULL	NULL
Ahmad	V	Jabbar	…	…	…	…	…	…	4	NULL	NULL	NULL	NULL

The Relational Algebra

- RIGHT OUTER JOIN: R ⟖ S
  - In addition to the result of the corresponding inner join, the RIGHT OUTER JOIN operation keeps every tuple in the *second*, or *right*, *relation S* in R ⟖ S; if *no* matching tuple is found in R, then the attributes of R in the join result are filled or "padded" with null values.
- FULL OUTER JOIN: R ⟗ S
  - In addition to the result of the corresponding inner join, the FULL OUTER JOIN operation keeps every tuple in both the *left* and the *right relations* when *no* matching tuples are found, padding them with null values as needed.

The Relational Algebra

- SEMI-JOIN: $T1 \ltimes_{T1.X=T2.Y} T2$
  - A tuple of T1 is returned as soon as T1.X finds a match with any value of T2.Y *without* searching for *further* matches.
    - This is in contrast to finding all possible matches in inner join.
  - Semi-join is generally used for unnesting EXISTS, IN, and ANY subqueries.
  - SEMI-JOIN can be extended with any *conditions*.

Return the information of all the departments *with at least one* employee who has salary > 30000.

DEPARTMENT $\ltimes_{\text{Dnumber=Dno}} (\sigma_{\text{Salary>30000}}(\text{EMPLOYEE}))$

The Relational Algebra

- ANTI JOIN: $T1 \overline{\ltimes}_{X = Y} T2$
  - A tuple of T1 is returned, only if T1.X *does not* match with any value of T2.Y. A tuple of T1 is rejected as soon as T1.X finds a match with any value of T2.Y.
  - Anti-join is used for unnesting NOT EXISTS, NOT IN, and ALL subqueries.
  - ANTI JOIN can be extended with any *conditions*.

Return the information of all the employees who *do not* work in any department which was managed since 1990.

EMPLOYEE $\overline{\ltimes}_{\text{Dno=Dnumber}} (\sigma_{\text{Mgr\_start\_date>='1990-01-01'}}(\text{DEPARTMENT}))$

The Relational Algebra

- UNION: R U S
  - *Union compatibility*: R and S has the same degree and each corresponding pair of attributes has the same domain.
  - Produce a new relation that includes all tuples that are either in R or in S or in both R and S. Duplicate tuples are eliminated.

RESULT1 U RESULT2

RESULT1	SSN
	123456789
	333445555
	666884444
	453453453

RESULT2	SSN
	333445555
	888665555

RESULT	SSN
	123456789
	333445555
	666884444
	453453453
	888665555

The Relational Algebra

- INTERSECTION: R ∩ S
  - *Union compatibility*: R and S has the same degree and each corresponding pair of attributes has the same domain.
  - Produce a new relation that includes all tuples that are in both R and S.

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

INSTRUCTOR	FNAME	LNAME
	John	Smith
	Ricardo	Browne
	Susan	Yao
	Francis	Johnson
	Ramesh	Shah

STUDENT ∩ INSTRUCTOR

FN	LN
Susan	Yao
Ramesh	Shah

The Relational Algebra

- DIFFERENCE: R – S
  - *Union compatibility*: R and S has the same degree and each corresponding pair of attributes has the same domain.
  - Produce a new relation that includes all tuples that are in R but not in S.

STUDENT	FN	LN
	Susan	Yao
	Ramesh	Shah
	Johnny	Kohler
	Barbara	Jones
	Amy	Ford
	Jimmy	Wang
	Ernest	Gilbert

INSTRUCTOR	FNAME	LNAME
	John	Smith
	Ricardo	Browne
	Susan	Yao
	Francis	Johnson
	Ramesh	Shah

STUDENT - INSTRUCTOR

FN	LN
Johnny	Kohler
Barbara	Jones
Amy	Ford
Jimmy	Wang
Ernest	Gilbert

INSTRUCTOR - STUDENT

FNAME	LNAME
John	Smith
Ricardo	Browne
Francis	Johnson

The Relational Algebra

- DIVISION: T(Y) = R(Z) ÷ S(X), where $X \subseteq Z$, Y = Z - X
  - Produce a new relation T each tuple of which appears in R in combination with *every* tuple in S
  - A relation T(Y) includes a tuple *t* if tuples t_R appear in R with $t_R[Y] = t$, and with $t_R[X] = t_S$ for every tuple t_S in S.
  - DIVISION can be expressed for the *all* condition as a sequence of π, x, and – operations.

Return the employees who worked on *all* the projects that Smith worked:

SSN_PNOS ÷ SMITH_PNOS

The projects of all employees

SSN_PNOS	ESSN	PNO
	123456789	1
	123456789	2
	666884444	3
	453453453	1
	453453453	2
	333445555	2
	333445555	3
	333445555	10
	333445555	20
	999887777	30
	999887777	10
	987987987	10
	987987987	30
	987654321	30
	987654321	20
	888665555	20

Smith's projects

SMITH_PNOS	PNO
	1
	2

The Relational Algebra

- **DIVISION**: T(Y) = R(Z) ÷ S(X), where $X \subseteq Z$, $Y = Z - X$
  - Produce a new relation T each tuple of which appears in R in combination with *every* tuple in S

$$T1 = \pi_Y(R)$$
$$T2 = \pi_Y(T1 \times S - R)$$
$$T = T1 - T2$$

Return the employees who worked on *all* the projects that Smith worked:

SSN_PNOS ÷ SMITH_PNOS

SSNS	SSN
	123456789
	453453453

The projects of all employees

SSN_PNOS	ESSN	PNO
	123456789	1
	123456789	2
	666884444	3
	453453453	1
	453453453	2
	333445555	2
	333445555	3
	333445555	10
	333445555	20
	999887777	30
	999887777	10
	987987987	10
	987987987	30
	987654321	30
	987654321	20
	888665555	20

Smith's projects

SMITH_PNOS	PNO
	1
	2

The Relational Algebra

- AGGREGATION: *<a grouping attribute list>* $\Im$ *<function list>* (R)
  - <grouping attributes> is a list of attributes of R.
  - <function list> is a list of (<function> <attribute>) pairs. In each such pair, <function> is one of the allowed *aggregate* functions—such as SUM, AVERAGE, MAXIMUM, MINIMUM, COUNT—and <attribute> is an attribute of R.
  - The resulting relation has the grouping attributes plus one attribute for each element in the function list.

How many employees work in the company?

How many employees work in each department of the company?

The Relational Algebra

- **AGGREGATION**: *<a grouping attribute list>* ℑ *<function list>* (R)

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

How many employees work in the company?

ℑ COUNT Ssn (EMPLOYEE)

COUNT_Ssn
8

How many employees work in each department of the company?

Dno ℑ COUNT Ssn (EMPLOYEE)

Dno	COUNT_Ssn
5	4
4	3
1	1

The Relational Algebra

- PUT THEM ALTOGETHER:
  - 1. Retrieve Ssn, name, and address of the employees who work in department 5.
  - 2. Retrieve Ssn, name, and address of the employees who work in 'Research' department.
  - 3. Retrieve Ssn, name, and address of the employees who work in departments in Houston.
  - 4. Return the average salary of the employees who work in department 1.
  - 5. Return the maximum salary of the employees in each department.
  - 6. Return the number of the employees who have salaries greater than the average salary of the employees who work in department 1.

The Relational Algebra

- PUT THEM ALTOGETHER:
  - 7. For every project located in ‘Stafford’, list the project number, the controlling department number, and the department manager’s last name, address, and birth date.
  - 8. Find the names of employees who work on all the projects controlled by department number 5.
  - 9. Make a list of project numbers for projects that involve an employee whose last name is ‘Smith’, either as a worker or as a manager of the department that controls the project.
  - 10. List the names of all employees with two or more dependents in department 5.
  - 11. Retrieve the names of employees who have no dependents.
  - 12. List the names of managers who have at least one dependent.

Summary

- The relational data model (*representational*)
  - Relation
  - Tuples
  - Attributes
  - Domains
  - Primary key
  - Constraints
    - Domain constraints
    - Key constraints
    - Constraints on nulls
    - Entity integrity constraints
    - Referential integrity constraints

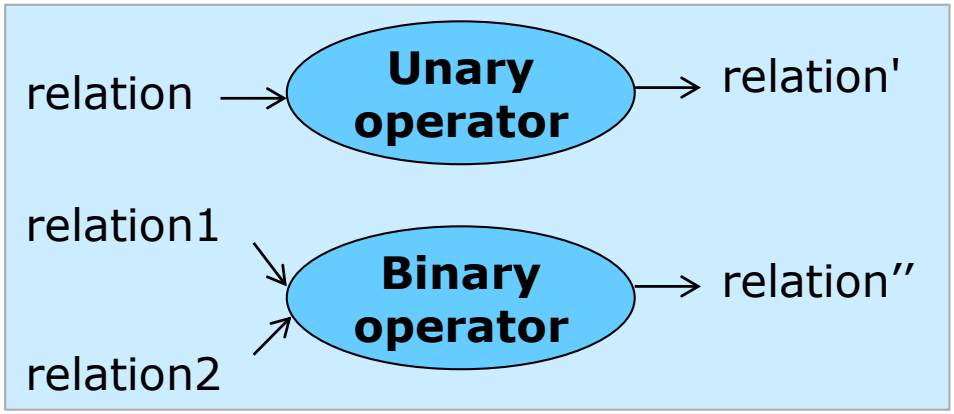
Summary

- Data model mapping
  - Step 1: Mapping of Regular Entity Types
  - Step 2: Mapping of Weak Entity Types
  - Step 3: Mapping of Binary 1:1 Relationship 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
  - Step 8: Mapping of Specialization or Generalization
  - Step 9: Mapping of Union Types (Categories)

Pay attention to Primary key, Foreign key, and other constraints

Summary

- The relational algebra
  - The basic set of operations that specify basic retrieval requests
    - The result is a new relation that has been formed from one or many relations.
  - A basis for implementing and optimizing queries in relational database management systems
  - Operations:
    - $\sigma_{<conditions>}$, $\pi_{<an\ attribute\ list>}$
    - x, $\bowtie_{<conditions>}$, *
    - Outer joins: $\bowtie$ $\bowtie$ $\bowtie$
    - Semi-join: $\ltimes$
    - Anti-join: $\overline{\ltimes}$
    - ∪, ∩, -, ÷
    - *<a grouping attribute list>* $\Im$ *<function list>*
    - ρ



Chapter 3:
The Relational Data Model

Review

- 1. Given the following examples, which one is a relation and which is not? Why not? If a relation, please specify its keys.

Book

Name	ISBN	Year	Publisher
ABC	12345	2000	ACM
EFGH	45932	2000	IEEE
ABC	28018	2000	Elsevier
EFHIO	98712	2000	Springer

iPhone

ID	Model	Supplier
I120	8	FPT
I2810	10	DiDong
I120	11pro	VTA
I2020_12	12	FPT
I2020_1	12	TGV

Customer

SSN	Name	Start	Contact
12345	NVA	09.20	123921
92012	TTT	01.19	{212921, 018271}
89213	CVV	02.20	890243
78406	DMT	03.18	{821571, 901823, 777890}

Trainee

Name	Age	Year
NVA	18	2000
NDV	20	2001
LAV	19	1999
NDV	20	2001

Review

- 2. Why are tuples in a relation not ordered? Why are duplicate tuples not allowed in a relation?
- 3. What is the difference between a key and a superkey? Why do we designate one of the candidate keys of a relation to be the primary key?
- 4. Discuss the various reasons that lead to the occurrence of NULL values in relations.
- 5. Discuss the entity integrity and referential integrity constraints. Give examples for illustration. Why is each considered important?

EMPLOYEE

Fname	Minit	Lname	Ssn	Bdate	Address	Sex	Salary	Super_ssn	Dno
John	B	Smith	123456789	1965-01-09	731 Fondren, Houston, TX	M	30000	333445555	5
Franklin	T	Wong	333445555	1955-12-08	638 Voss, Houston, TX	M	40000	888665555	5
Alicia	J	Zelaya	999887777	1968-01-19	3321 Castle, Spring, TX	F	25000	987654321	4
Jennifer	S	Wallace	987654321	1941-06-20	291 Berry, Bellaire, TX	F	43000	888665555	4
Ramesh	K	Narayan	666884444	1962-09-15	975 Fire Oak, Humble, TX	M	38000	333445555	5
Joyce	A	English	453453453	1972-07-31	5631 Rice, Houston, TX	F	25000	333445555	5
Ahmad	V	Jabbar	987987987	1969-03-29	980 Dallas, Houston, TX	M	25000	987654321	4
James	E	Borg	888665555	1937-11-10	450 Stone, Houston, TX	M	55000	NULL	1

DEPARTMENT

Dname	Dnumber	Mgr_ssn	Mgr_start_date
Research	5	333445555	1988-05-22
Administration	4	987654321	1995-01-01
Headquarters	1	888665555	1981-06-19

DEPT_LOCATIONS

Dnumber	Dlocation
1	Houston
4	Stafford
5	Bellaire
5	Sugarland
5	Houston

WORKS_ON

Essn	Pno	Hours
123456789	1	32.5
123456789	2	7.5
666884444	3	40.0
453453453	1	20.0
453453453	2	20.0
333445555	2	10.0
333445555	3	10.0
333445555	10	10.0
333445555	20	10.0
999887777	30	30.0
999887777	10	10.0
987987987	10	35.0
987987987	30	5.0
987654321	30	20.0
987654321	20	15.0
888665555	20	NULL

PROJECT

Pname	Pnumber	Plocation	Dnum
ProductX	1	Bellaire	5
ProductY	2	Sugarland	5
ProductZ	3	Houston	5
Computerization	10	Stafford	4
Reorganization	20	Houston	1
Newbenefits	30	Stafford	4

DEPENDENT

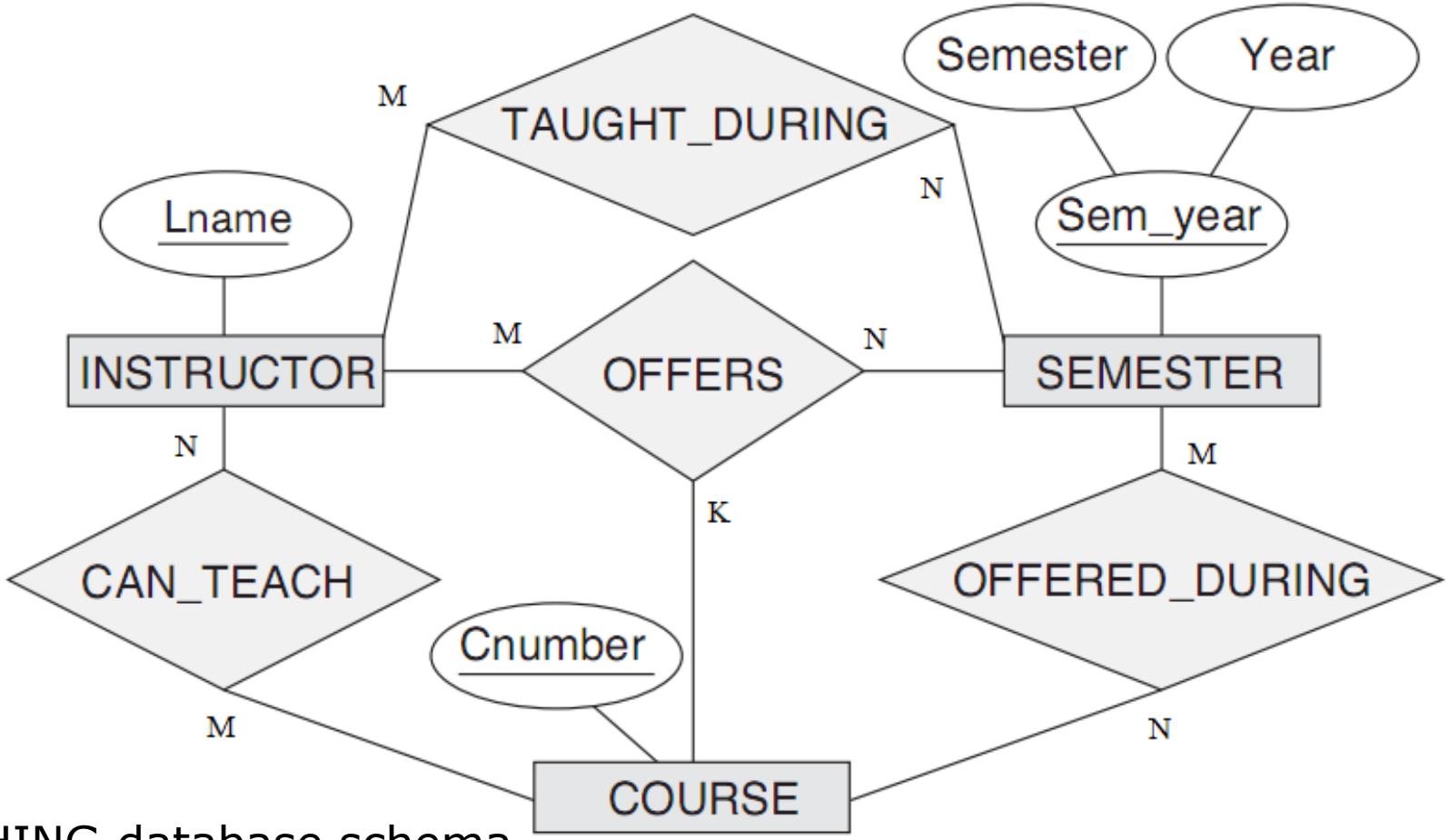
Essn	Dependent_name	Sex	Bdate	Relationship
333445555	Alice	F	1986-04-05	Daughter
333445555	Theodore	M	1983-10-25	Son
333445555	Joy	F	1958-05-03	Spouse
987654321	Abner	M	1942-02-28	Spouse
123456789	Michael	M	1988-01-04	Son
123456789	Alice	F	1988-12-30	Daughter
123456789	Elizabeth	F	1967-05-05	Spouse

- 6. Given a part of the COMPANY database.
  - Discuss all integrity constraints violated by each operation provided next, if any, and the different ways of enforcing these constraints.

- 6. Discuss all integrity constraints violated by each operation, if any, and the different ways of enforcing these constraints.
  - a. Insert <'Robert', 'F', 'Scott', '943775543', '1972-06-21', '2365 Newcastle Rd, Bellaire, TX', M, 58000, '888665555', 1> into EMPLOYEE.
  - b. Insert <'ProductA', 4, 'Bellaire', 2> into PROJECT.
  - c. Insert <'Production', 4, '943775543', '2007-10-01'> into DEPARTMENT.
  - d. Insert <'677678989', NULL, '40.0'> into WORKS_ON.
  - e. Insert <'453453453', 'John', 'M', '1990-12-12', 'spouse'> into DEPENDENT.
  - f. Delete the WORKS_ON tuples with Essn = '333445555'.
  - g. Delete the EMPLOYEE tuple with Ssn = '987654321'.
  - h. Delete the PROJECT tuple with Pname = 'ProductX'.
  - i. Modify the Mgr_ssn and Mgr_start_date of the DEPARTMENT tuple with Dnumber = 5 to '123456789' and '2007-10-01', respectively.
  - j. Modify the Super_ssn attribute of the EMPLOYEE tuple with Ssn = '999887777' to '943775543'.
  - k. Modify the Hours attribute of the WORKS_ON tuple with Essn = '999887777' and Pno = 10 to '5.0'.

Review

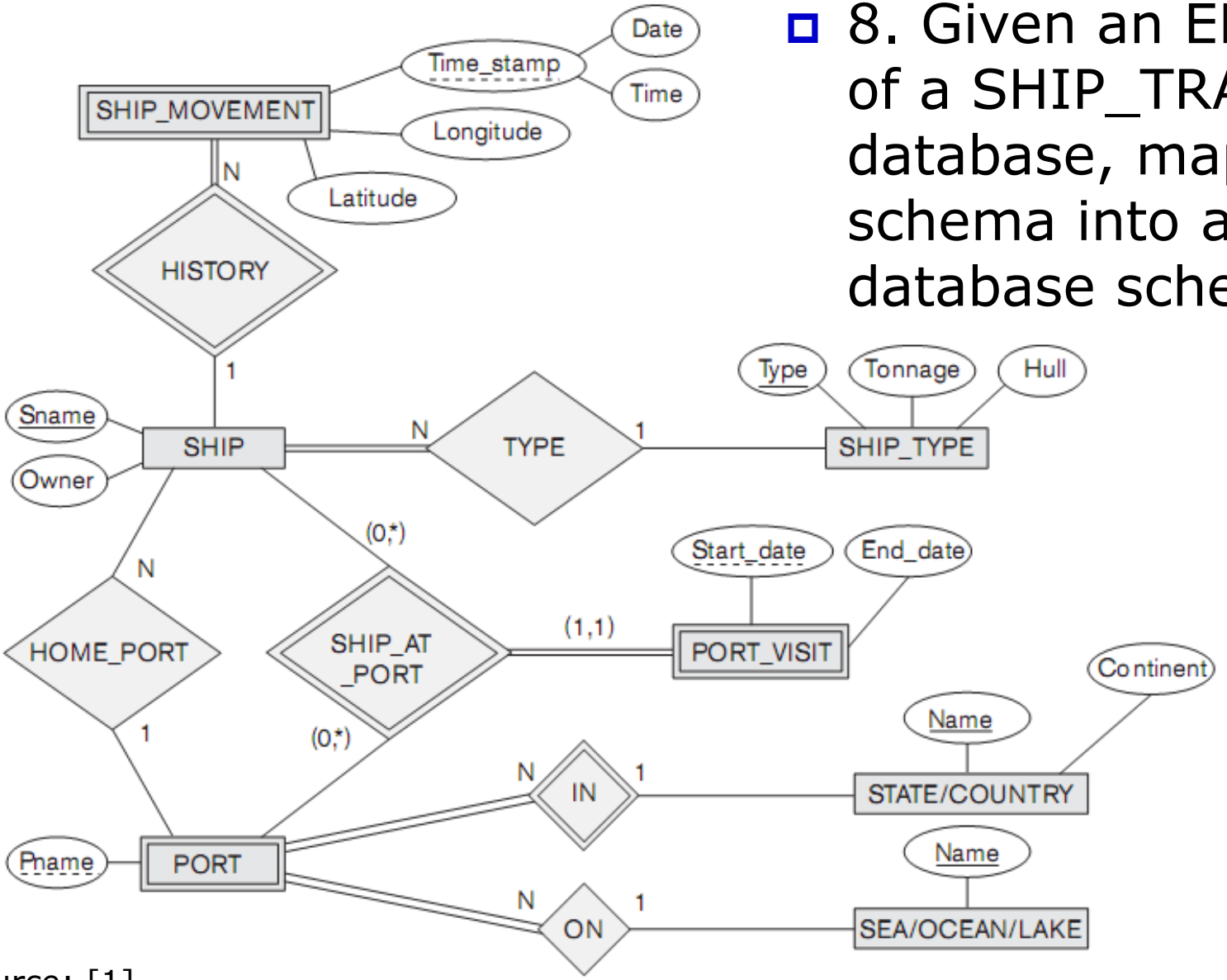
- 7. Given an ER schema of a TEACHING database, map this schema into a relational database schema.



A TEACHING database schema

Source: [1]

Review

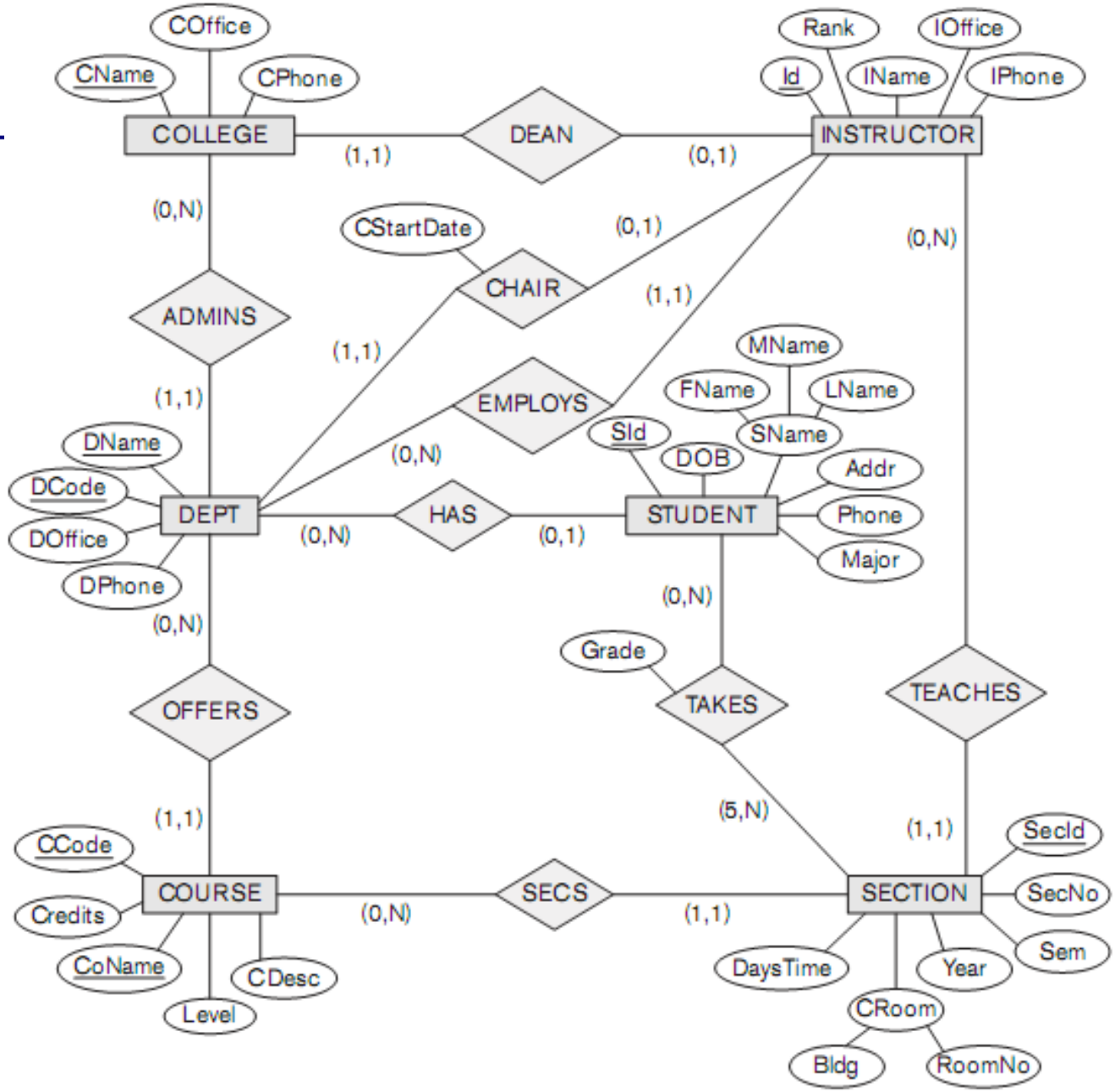


- 8. Given an ER schema of a SHIP_TRACKING database, map this schema into a relational database schema.

Source: [1]

Review

- 9. Given an ER schema of a University database, map this schema into a relational database schema.



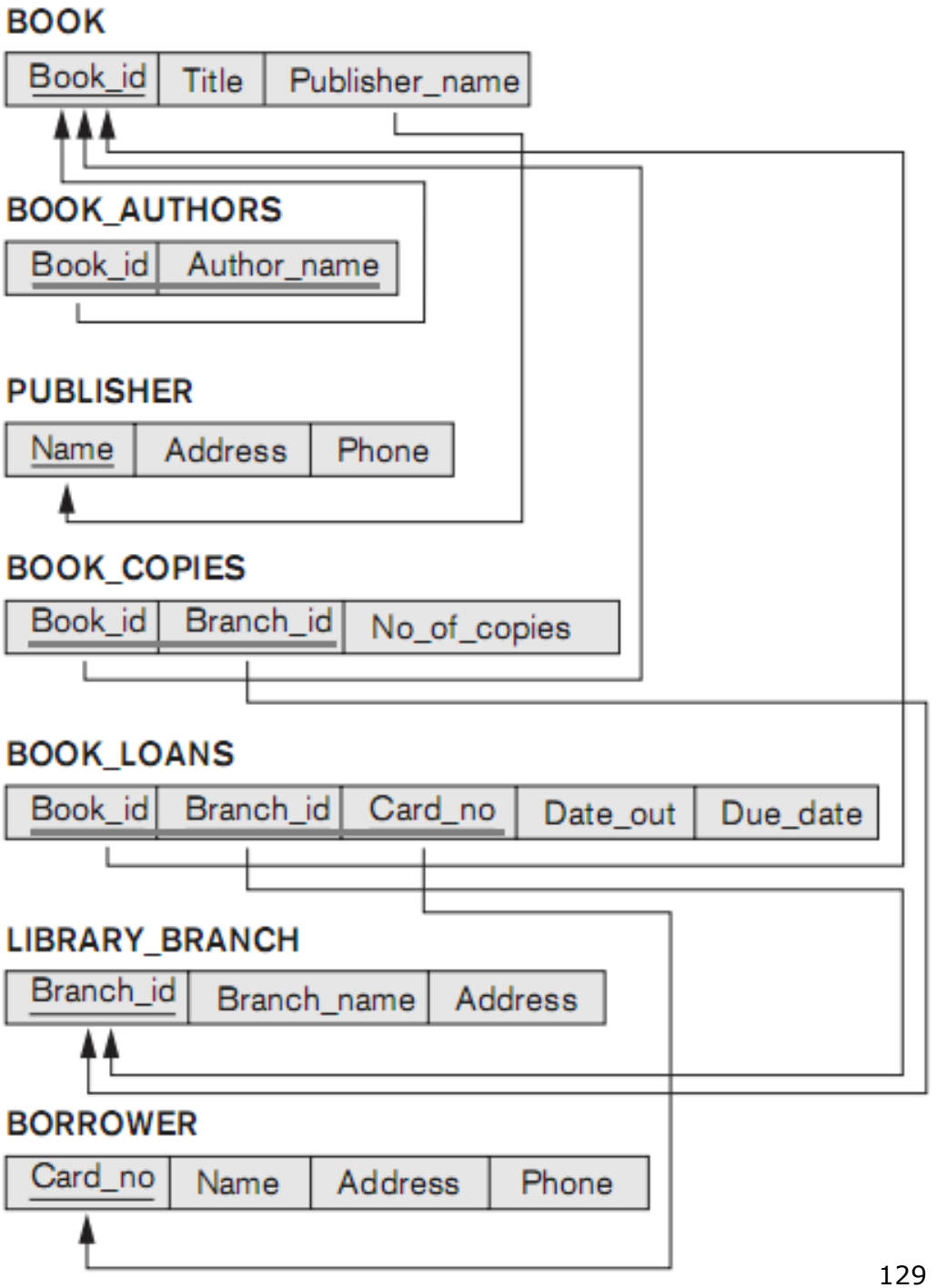
A University database schema
Source: [1]

Review

10. Try to map the relational schema of a LIBRARY database into an ER schema.

This is part of a process known as *reverse engineering*, where a conceptual schema is created for an existing implemented database.

State any assumptions you make.



Review

- 11. Specify the following queries on the COMPANY relational database schema using the relational algebra:
  - a. Retrieve the names of all employees in department 5 who work more than 10 hours per week on the ProductX project.
  - b. List the names of all employees who have a dependent with the same first name as themselves.
  - c. Find the names of all employees who are directly supervised by 'Franklin Wong'.
  - d. For each project, list the project name and the total hours per week (by all employees) spent on that project.
  - e. Retrieve the names of all employees who work on every project.
  - f. Retrieve the names of all employees who do not work on any project.
  - g. For each department, retrieve the department name and the average salary of all employees working in that department.
  - h. Retrieve the average salary of all female employees.
  - i. Find the names and addresses of all employees who work on at least one project located in Houston but whose department has no location in Houston.
  - j. List the last names of all department managers who have no dependents.

Review

- 12. Specify the following queries on the LIBRARY relational database schema in Question 10 using the relational algebra:
  - a. How many copies of the book titled The Lost Tribe are owned by the library branch whose name is 'Sharpstown'?
  - b. How many copies of the book titled The Lost Tribe are owned by each library branch?
  - c. Retrieve the names of all borrowers who do not have any books checked out.
  - d. For each book that is loaned out from the Sharpstown branch and whose Due_date is today, retrieve the book title, the borrower's name, and the borrower's address.
  - e. For each library branch, retrieve the branch name and the total number of books loaned out from that branch.
  - f. Retrieve the names, addresses, and number of books checked out for all borrowers who have more than five books checked out.
  - g. For each book authored (or coauthored) by Stephen King, retrieve the title and the number of copies owned by the library branch whose name is Central.

Next
Chapter 4: The SQL Language

- 4.1. Introduction to the SQL language
- 4.2. DDL
- 4.3. DML
- 4.4. DCL
- 4.5. Stored Functions
- 4.6. Stored Procedures
- 4.7. Triggers