Intro to RDBMS, Database design

Rajeev Gupta
Java trainer & consultant

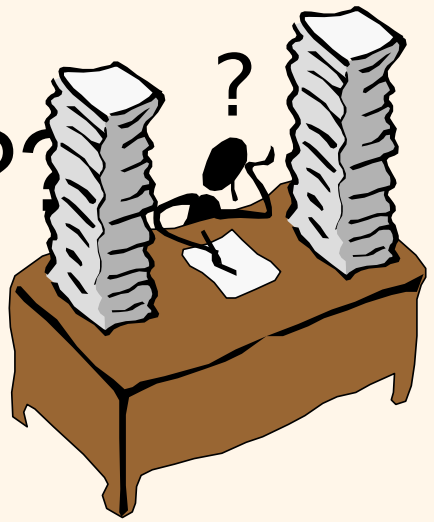
What is DBMS?

- Need for information management
- A very large, integrated collection of data.
- Models real-world *enterprise.*
  - Entities (e.g., students, courses)
  - Relationships (e.g., John is taking CS662)
- A *Database Management System (DBMS)* is a software package designed to store and manage databases.

Why Use a DBMS?

- Data independence and efficient access.
- Data integrity and security.
- Uniform data administration.
- Concurrent access, recovery from crashes.
- Replication control
- Reduced application development time.

Why Study Databases??

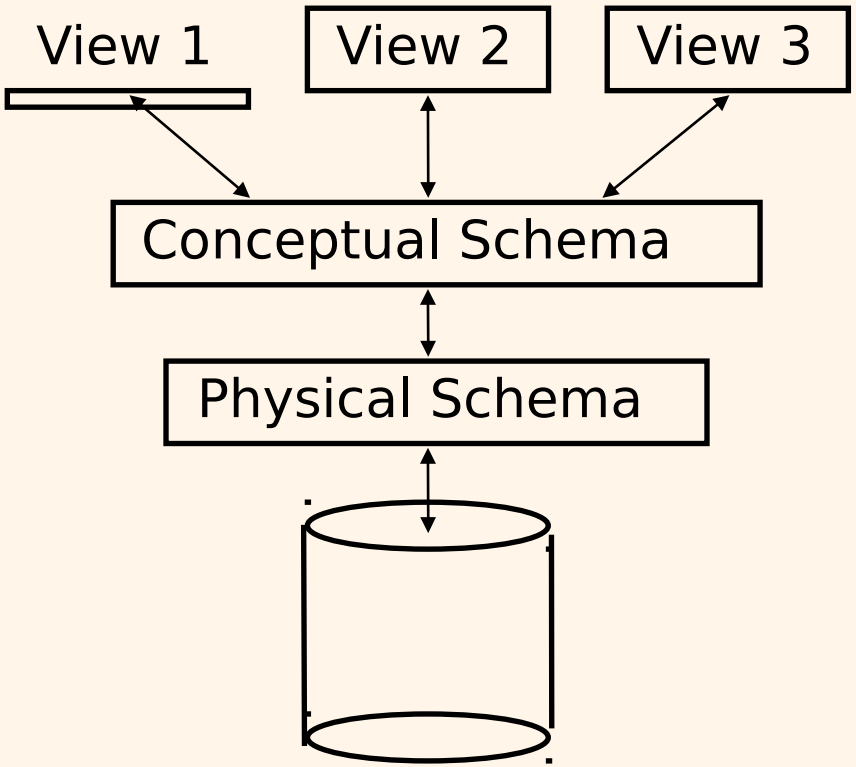


- Shift from *computation* to *information*
  - at the “low end”: access to physical world
  - at the “high end”: scientific applications
- Datasets increasing in diversity and volume.
  - Digital libraries, interactive video, Human Genome project, e-commerce, sensor networks
  - ... need for DBMS/data services exploding
- DBMS encompasses several areas of CS
  - OS, languages, theory, AI, multimedia, logic

Data Models

- A *data model* is a collection of concepts for describing data.
- A *schema* is a description of a particular collection of data, using the a given data model.
- The *relational model of data* is the most widely used model today.
  - Main concept: *relation*, basically a table with rows and columns.
  - Every relation has a *schema*, which describes the columns, or fields.

Levels of Abstraction



- Many *views*, single *conceptual (logical) schema* and *physical schema*.
  - Views describe how users see the data.
  - Conceptual schema defines logical structure
  - Physical schema describes the files and indexes used.

☛ *Schemas are defined using DDL; data is modified/queried using DML*.

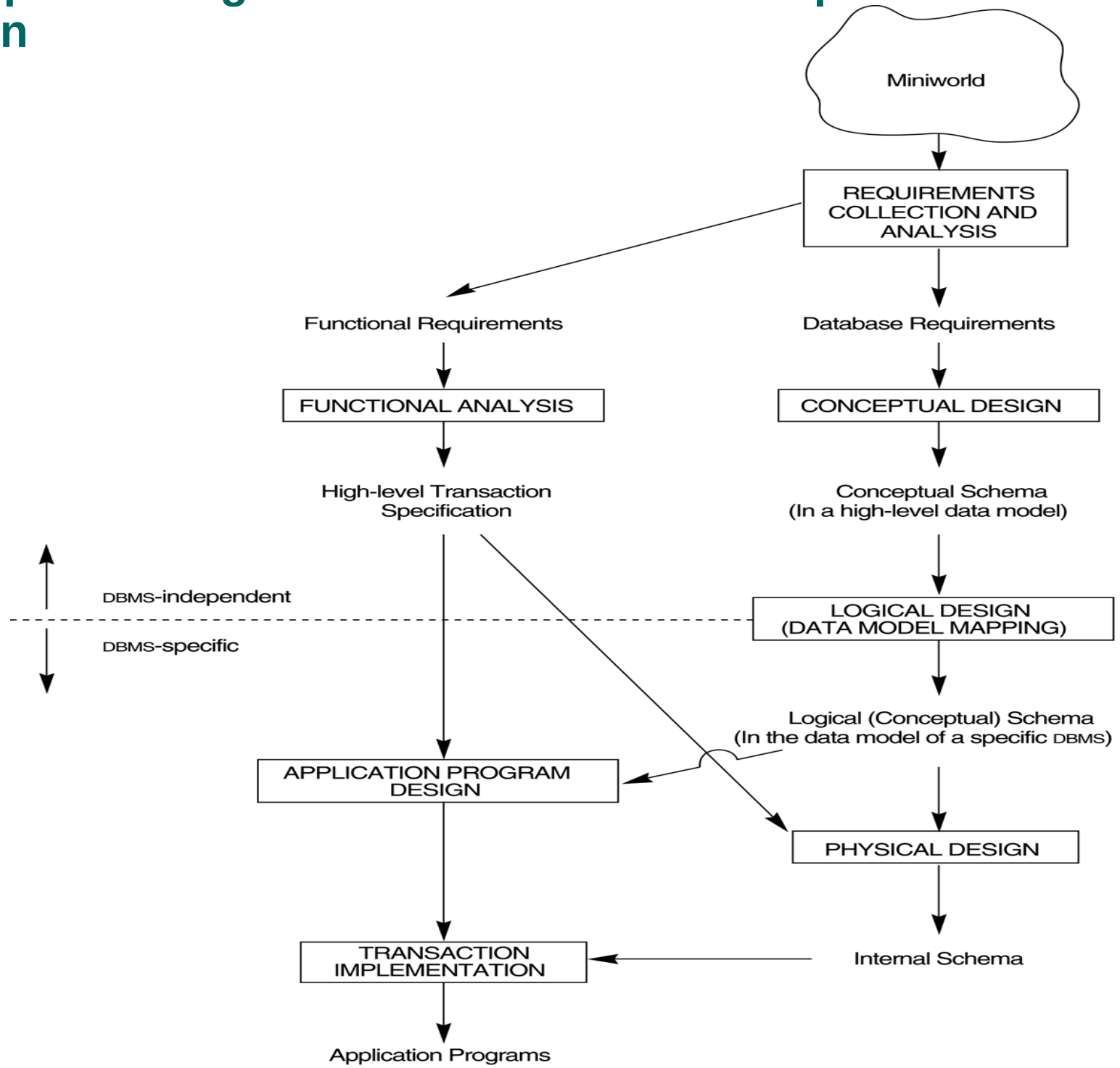
Outline

- Main Phases of Database Design
- Conceptual Database Design
- Logical Database Design
  - ER- & EER-to-Relational Mapping
- Exercises

Main Phases of Database Design

- Three main phases
  - Conceptual database design
  - Logical database design
  - Physical database design

A simplified diagram to illustrate the main phases of database design



Main Phases of Database Design

- Conceptual database design
  - The process of constructing a model of the data used in an enterprise, independent of *all* physical considerations
- Logical database design
  - The process of constructing a model of the data used in an enterprise based on a specific data model (e.g. relational), but independent of a particular DBMS and other physical considerations

Main Phases of Database Design

- Physical database design
  - The process of producing a description of the implementation of the database on secondary storage; it describes the base relations, file organizations, and indexes design used to achieve efficient access to the data, and any associated integrity constraints and security measures

Conceptual Database Design
Summarization

- To build a conceptual data model of the data requirements of the enterprise
  - Model comprises entity types, relationship types, attributes and attribute domains, primary and alternate keys, structural and integrity constraints

Conceptual Database Design
Summarization

- Step 1: Identify entity types
- Step 2: Identify relationship types
- Step 3: Identify and associate attributes with entity or relationship types
- Step 4: Determine attribute domains
- Step 5: Determine candidate, primary, and alternate key attributes
- Step 6: Consider use of enhanced modeling concepts (optional step)
- Step 7: Check model for redundancy
- Step 8: Validate conceptual model against user transactions
- Step 9: Review conceptual data model with user

Conceptual Database Design
Summarization

- Step 1: Identify entity types
  - To identify the required entity types
- Step 2: Identify relationship types
  - To identify the important relationships that exist between the entity types
- Step 3: Identify and associate attributes with entity or relationship types
  - To associate attributes with the appropriate entity or relationship types and document the details of each attribute
- Step 4: Determine attribute domains
  - To determine domains for the attributes in the data model and document the details of each domain

Conceptual Database Design
Summarization

- Step 5: Determine candidate, primary, and alternate key attributes
  - To identify the candidate key(s) for each entity and if there is more than one candidate key, to choose one to be the primary key and the others as alternate keys
- Step 6: Consider use of enhanced modeling concepts (optional step)
  - To consider the use of enhanced modeling concepts, such as specialization/generalization, categories (union types)
- Step 7: Check model for redundancy
  - To check for the presence of any redundancy in the model and to remove any that does exist

Conceptual Database Design
Summarization

- Step 8: Validate conceptual model against user transactions
  - To ensure that the conceptual model supports the required transactions
- Step 9: Review conceptual data model with user
  - To review the conceptual data model with the user to ensure that the model is a ‘true’ representation of the data requirements of the enterprise

Logical Database Design

- To translate the conceptual data model into a logical data model and then to validate this model to check that it is structurally correct using normalization and supports the required transactions

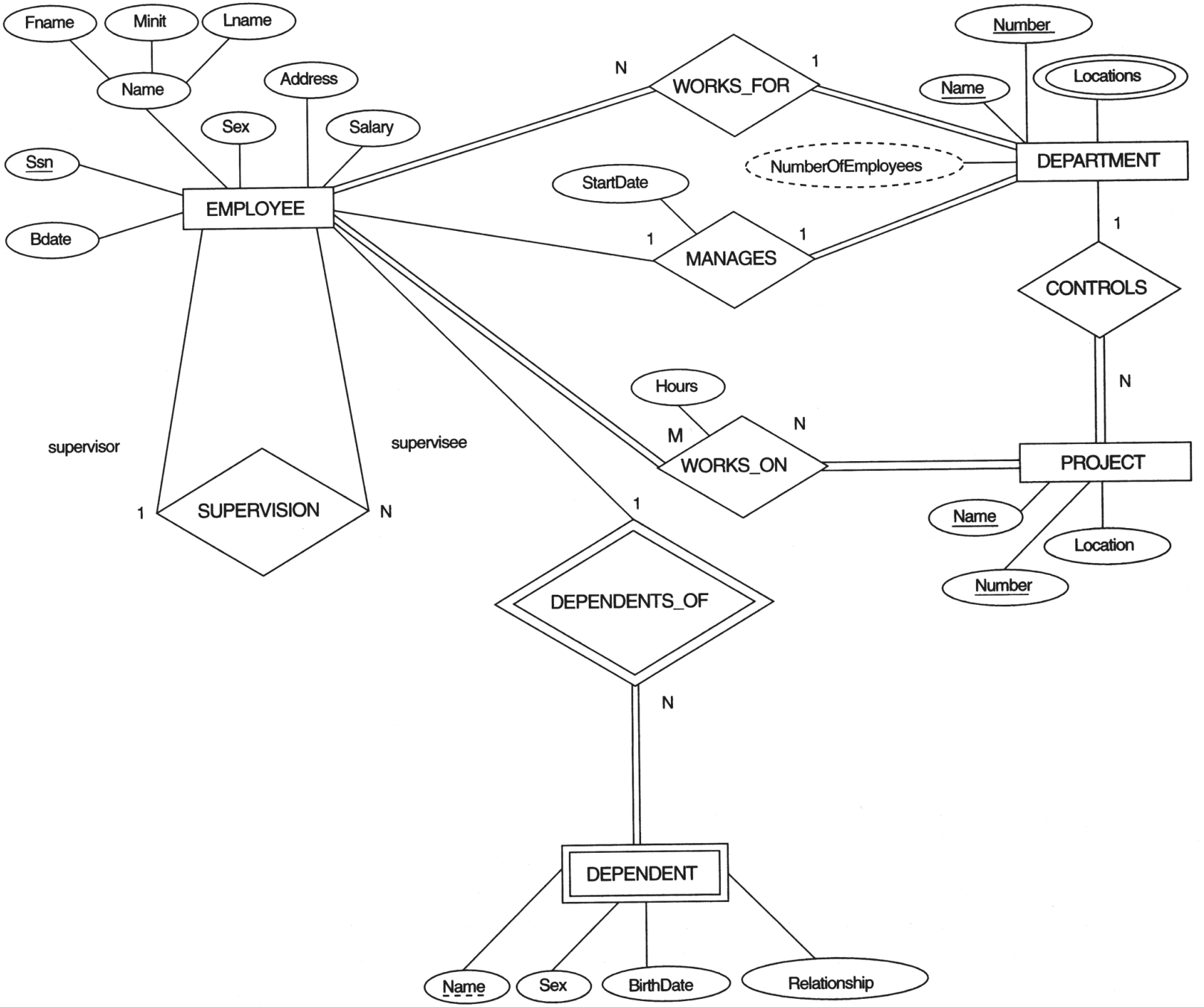
Logical Database Design

- Logical database design for the relational model
  - Step 1: Derive relations for logical data model
  - Step 2: Validate relations using normalization
  - Step 3: Validate relations against user transactions
  - Step 4: Define integrity constraints
  - Step 5: Review logical data model with user
  - Step 6: Merge logical data models into global model (optional step)
  - Step 7: Check for future growth
- ER-to-Relational Mapping

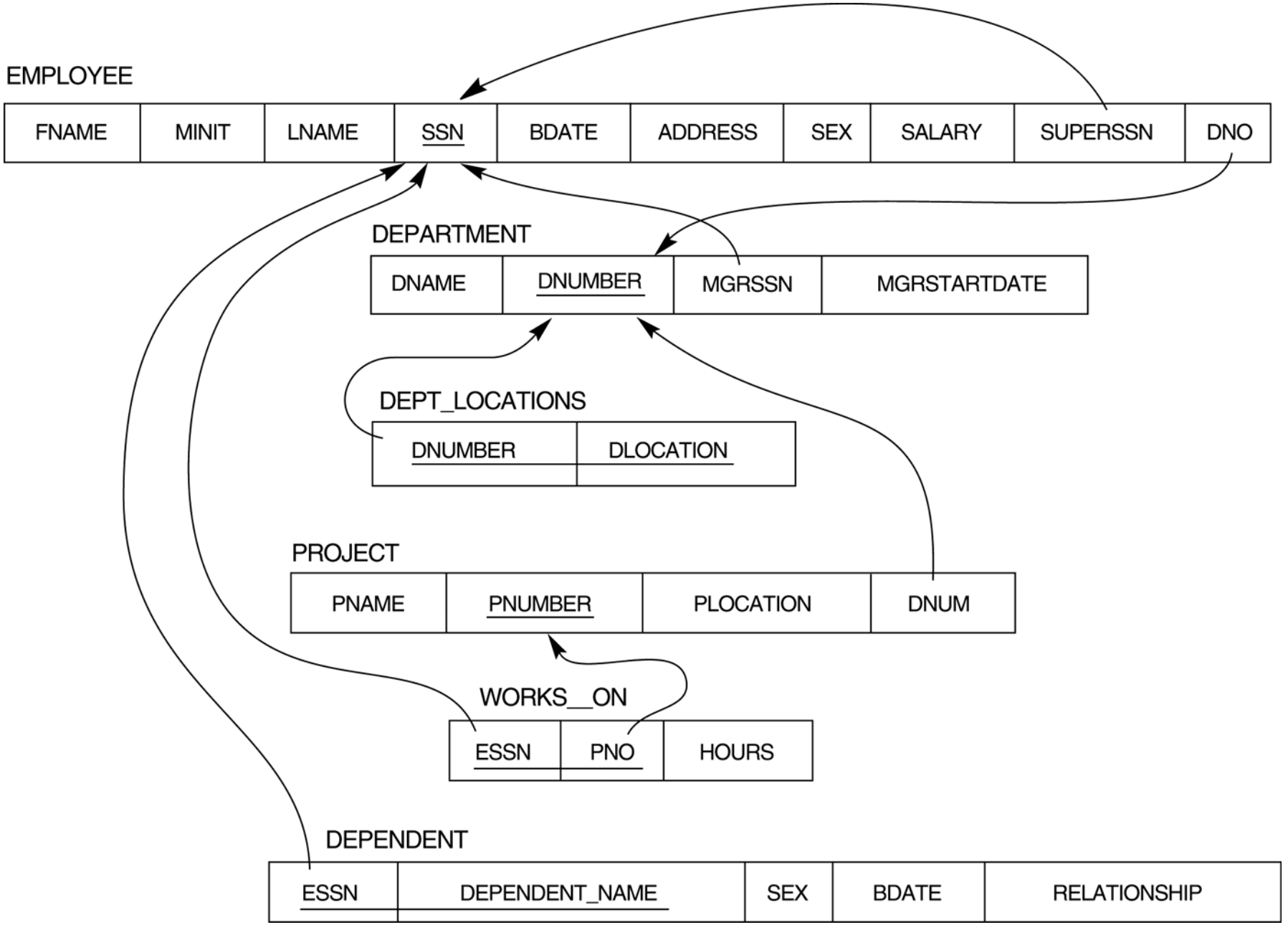
Case study

- A company is organized in departements each department has name, no and name of employee who managed the department. Each dept has manay locations. We keep track of start date of the manager.
- Eeach dept controls numbers of projects. Each project has a name, no and is located at single location.
- We store each employee SSN, address,salary, gender, dob. Each employee works for one dept but can work on many projects. We keep track of number of hr_per_week that an employee currently working on an project. We also keep track of direct superwiser.
- Each employee may have no of dependents. For each dependent we keep track of there name, gender,dob and relationship.
- Design ERD

The ERD for the COMPANY database



Result of mapping the COMPANY ER schema into a relational schema



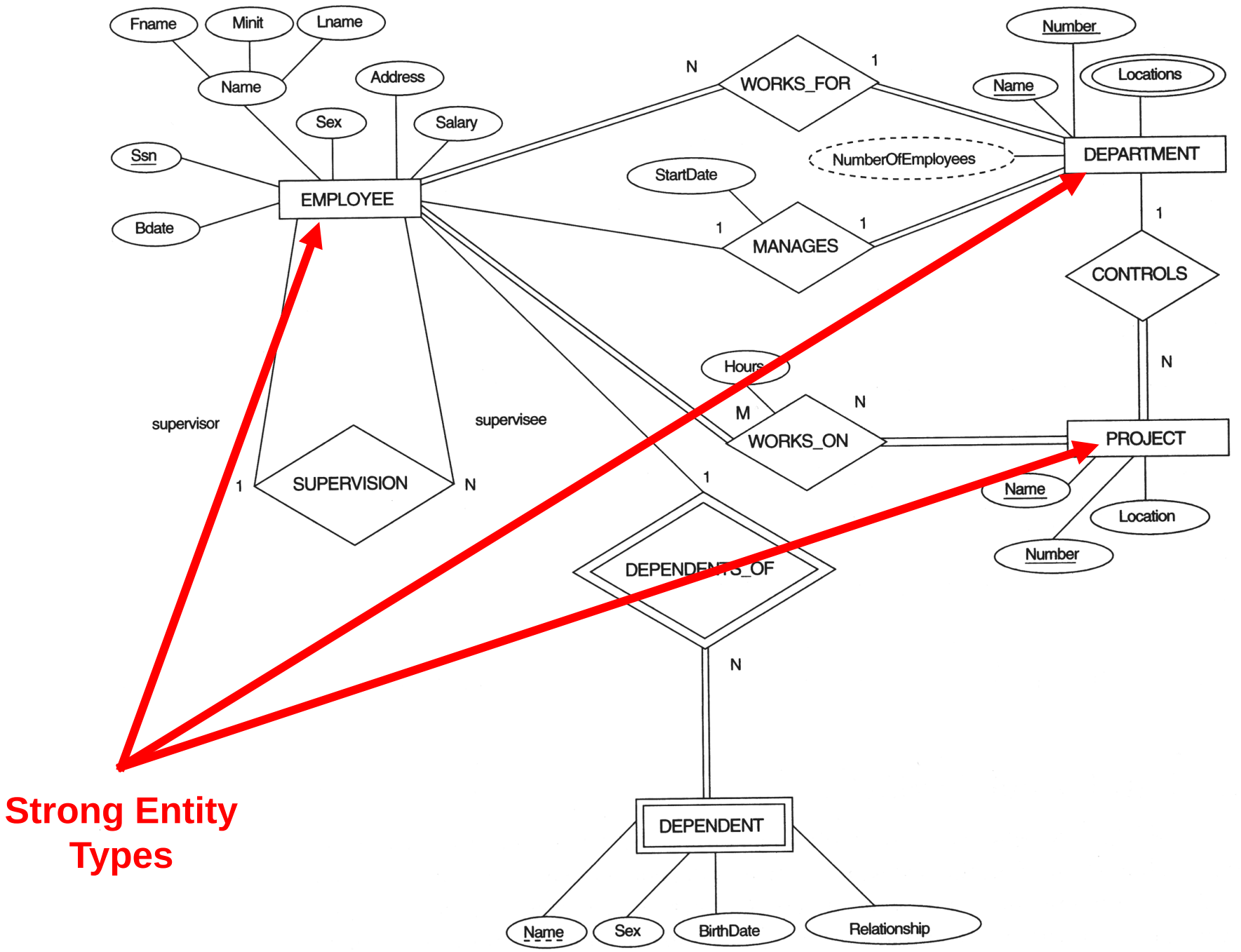
ER- & EER-to-Relational Mapping

- ER-
  - 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

ER-to-Relational Mapping

- Step 1: Mapping of Regular (strong) Entity Types
  - Entity --> Relation
  - Attribute of entity --> Attribute of relation
  - Primary key of entity --> Primary key of relation
  - **Example:** We create the relations EMPLOYEE, DEPARTMENT, and PROJECT in the relational schema corresponding to the regular entities in the ER diagram. SSN, DNUMBER, and PNUMBER are the primary keys for the relations EMPLOYEE, DEPARTMENT, and PROJECT as shown

The ERD for the COMPANY database



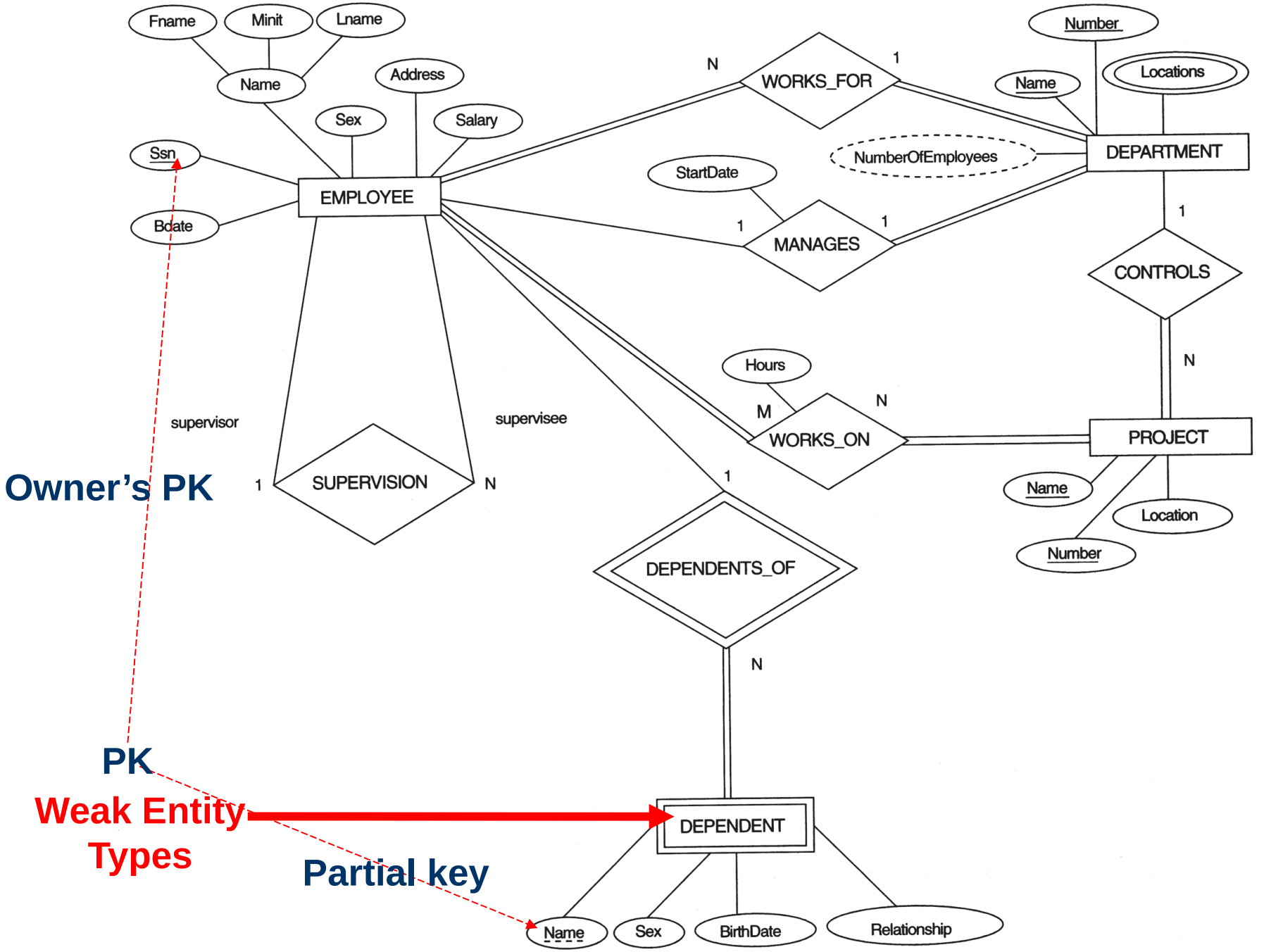
Strong Entity Types

ER-to-Relational Mapping

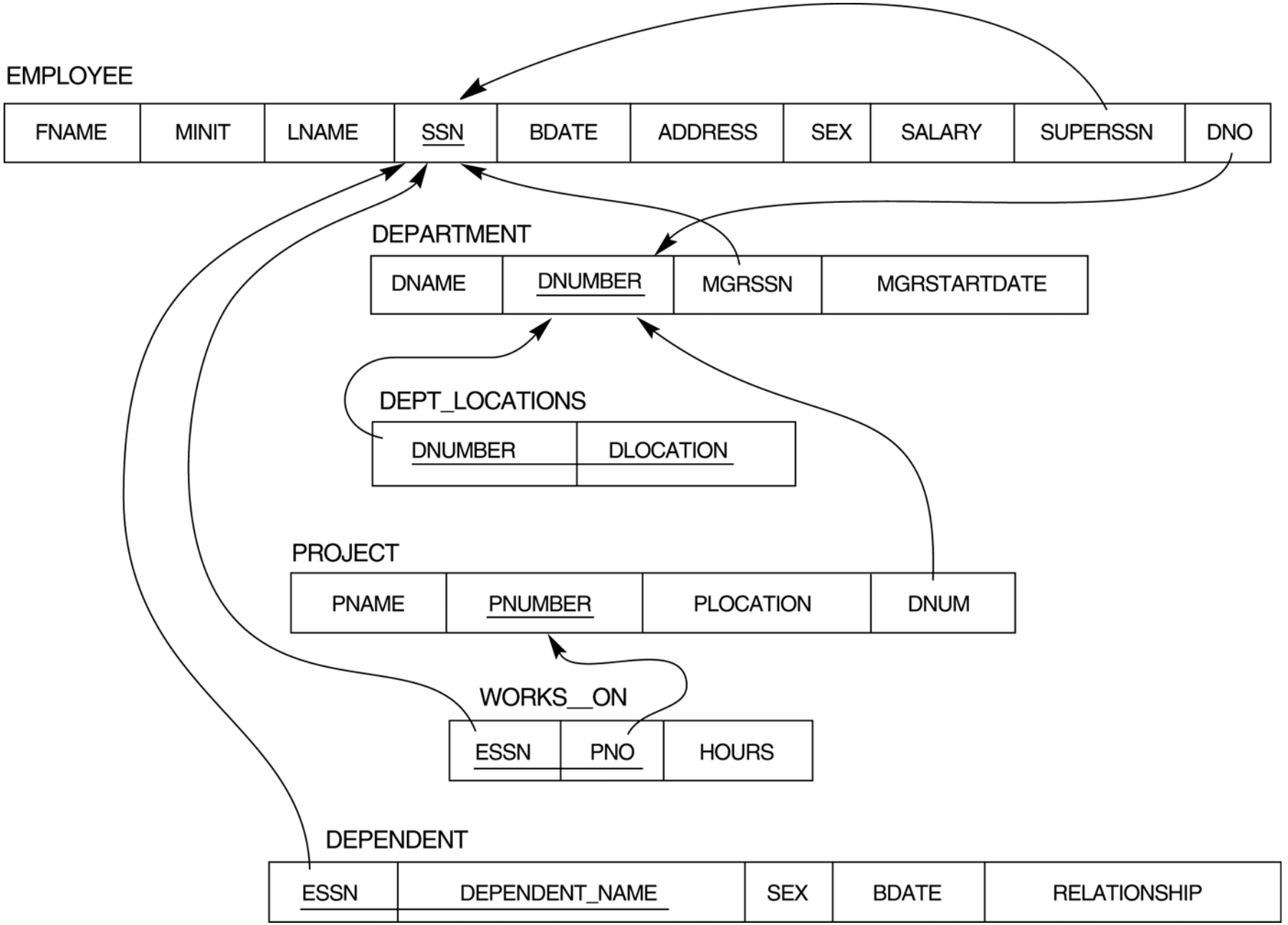
- Step 2: Mapping of Weak Entity Types
  - For each weak entity type W in the ER schema with owner entity type E, create a relation R and include all simple attributes (or simple components of composite attributes) of W as attributes of R
  - In addition, include as foreign key attributes of R the primary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

    The primary key of the DEPENDENT relation is the combination {ESSN, DEPENDENT_NAME} because DEPENDENT_NAME is the partial key of DEPENDENT
  - Note: CASCADE option as implemented

The ERD for the COMPANY database



Result of mapping the COMPANY ER schema into a relational schema



ER-to-Relational Mapping

- ER-
  - 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
- Transformation of binary relationships - depends on *functionality* of relationship and *membership class* of participating entity types

ER-to-Relational Mapping

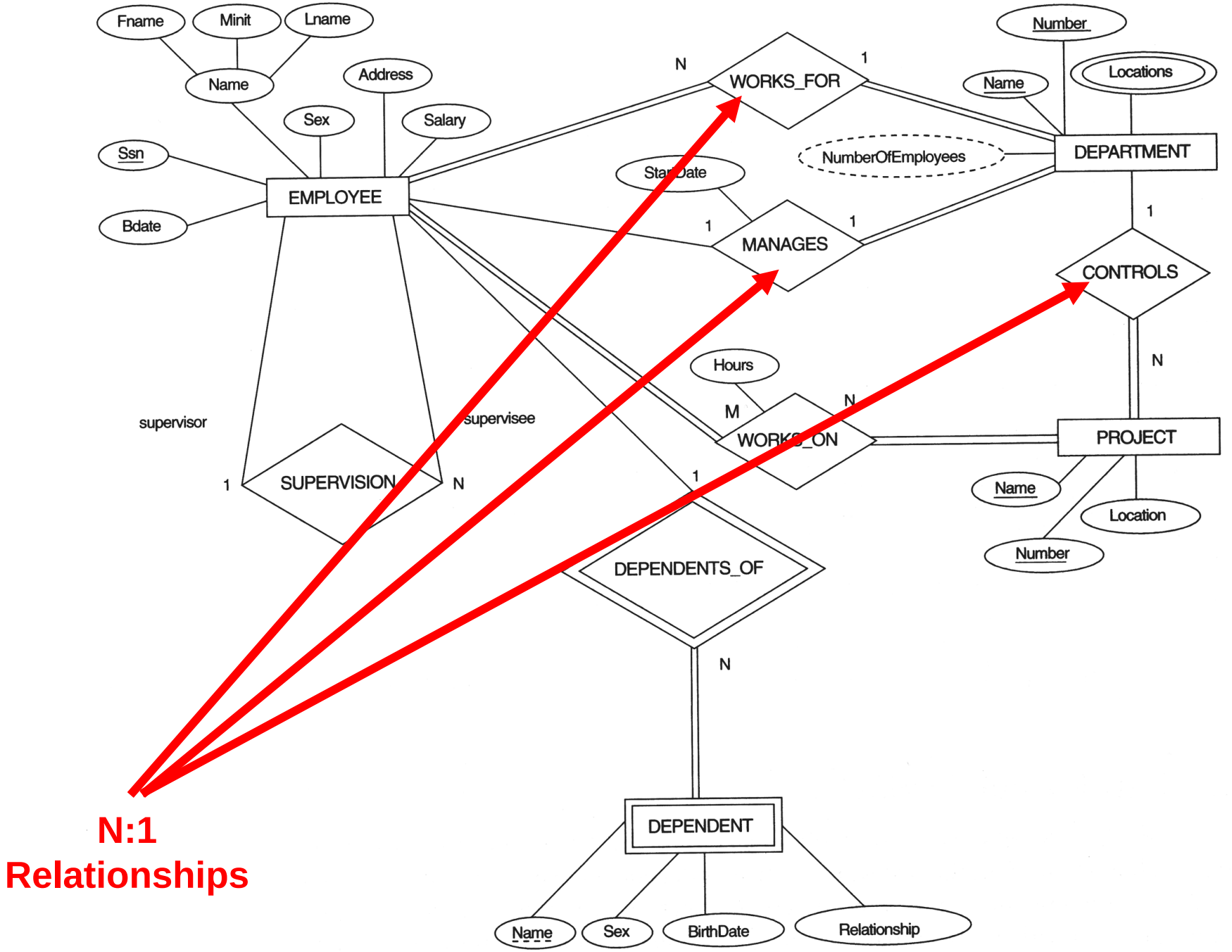
- **Mandatory** membership class
  - For two entity types E1 and E2: If E2 is a mandatory member of an N:1 (or 1:1) relationship with E1, then the relation for E2 will include the prime attributes of E1 as a foreign key to represent the relationship
  - For a 1:1 relationship: If the membership class for E1 and E2 are both mandatory, a foreign key can be used in either relation
  - For an N:1 relationship: If the membership class of E2, which is at the N-side of the relationship, is *optional* (i.e. partial), then the above guideline is not applicable

ER-to-Relational Mapping

- Assume every module must be offered by a department, then the entity type MODULE is a **mandatory** member of the relationship OFFER. The relation for MODULE is:

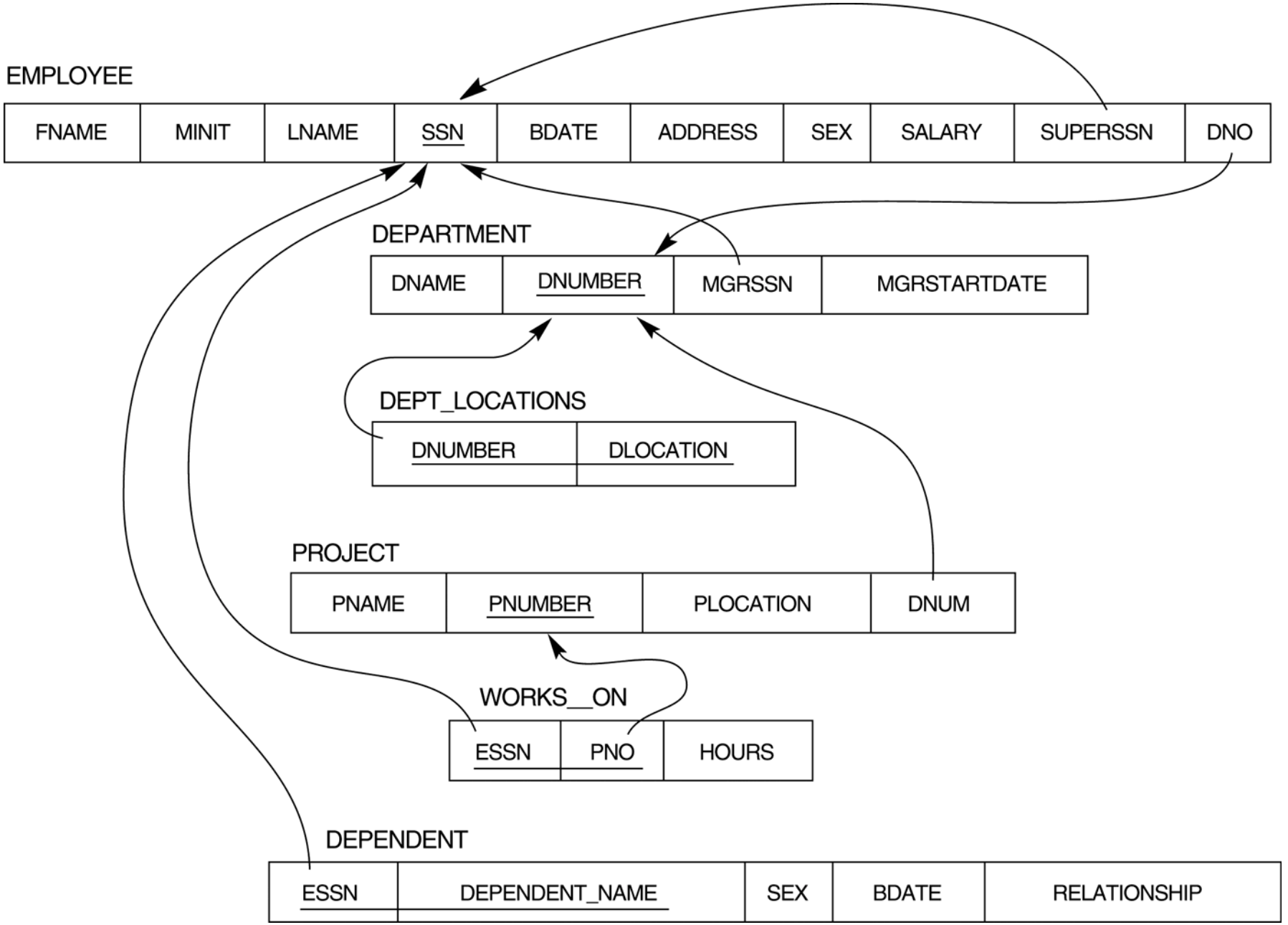
MODULE(MDL-NUMBER, TITLE, TERM, ..., **DNAME**)

The ERD for the COMPANY database



N:1 Relationships

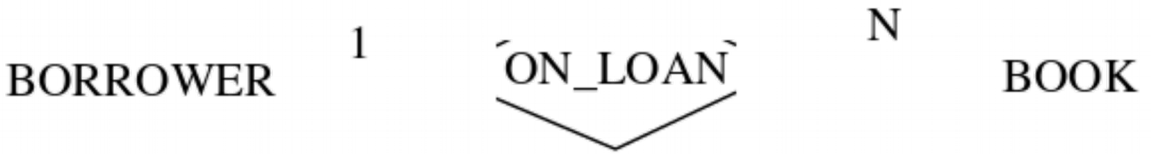
Result of mapping the COMPANY ER schema into a relational schema



ER-to-Relational Mapping

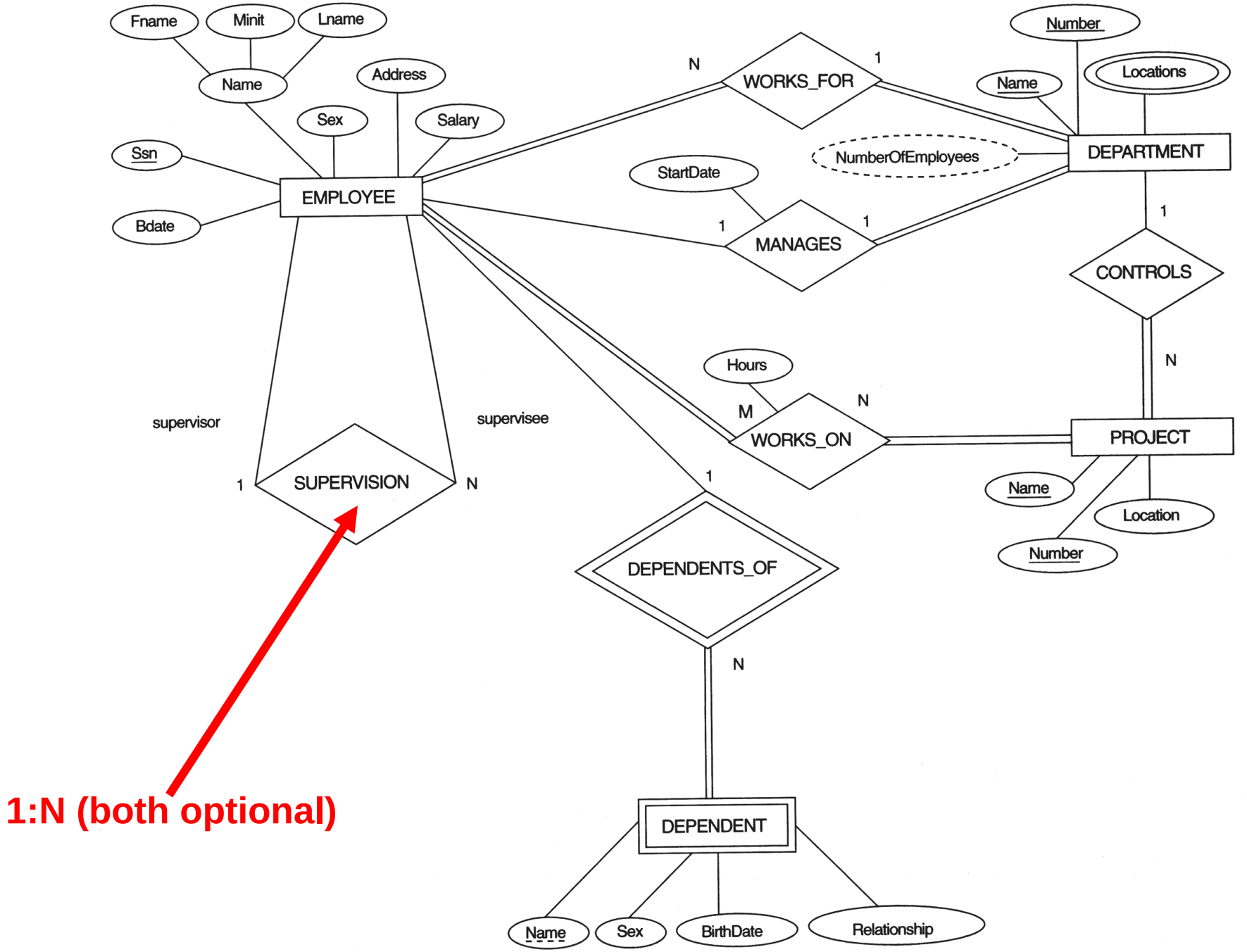
- **Optional** membership classes
  - If entity type E2 is an optional member of the N:1 relationship with entity type E1 (i.e. E2 is at the N-side of the relationship), then the relationship is **usually** represented by a new relation containing the prime attributes of E1 and E2, together with any attributes of the relationship. The key of the entity type at the N-side (i.e. E2) will become the key of the new relation
  - If both entity types in a 1:1 relationship have the optional membership, a new relation is created which contains the prime attributes of both entity types, together with any attributes of the relationship. The prime attribute(s) of either entity type will be the key of the new relation

ER-to-Relational Mapping



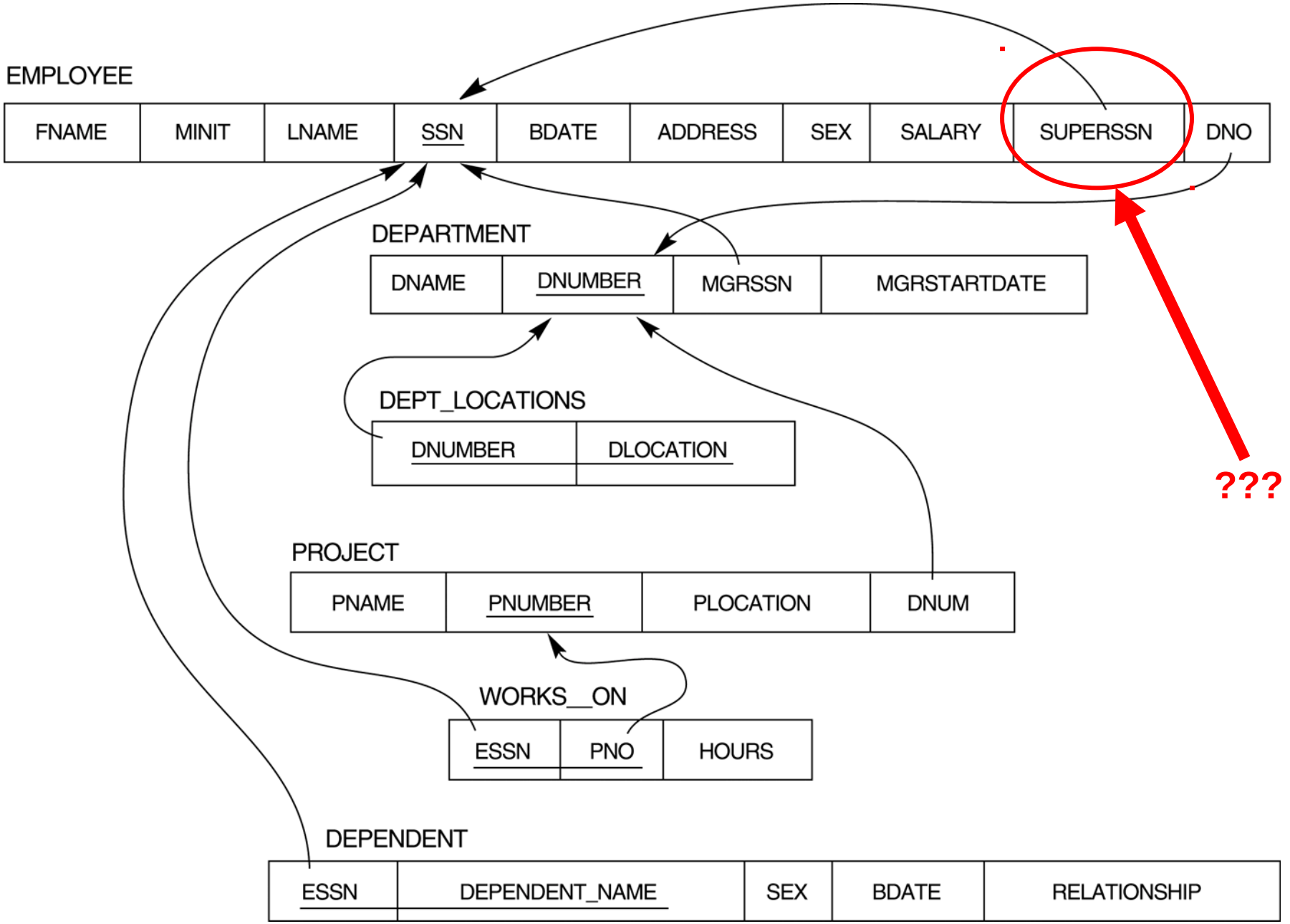
- One possible representation of the relationship:
  - BORROWER(<u>BNUMBER</u>, NAME, ADDRESS, ...)
  - BOOK(<u>ISBN</u>, TITLE, ..., **BNUMBER**)
- A better alternative:
  - BORROWER(<u>BNUMBER</u>, NAME, ADDRESS, ...)
  - BOOK(<u>ISBN</u>, TITLE, ...)
  - ON_LOAN(<u>ISBN</u>, BNUMBER)

The ERD for the COMPANY database



1:N (both optional)

Result of mapping the COMPANY ER schema into a relational schema



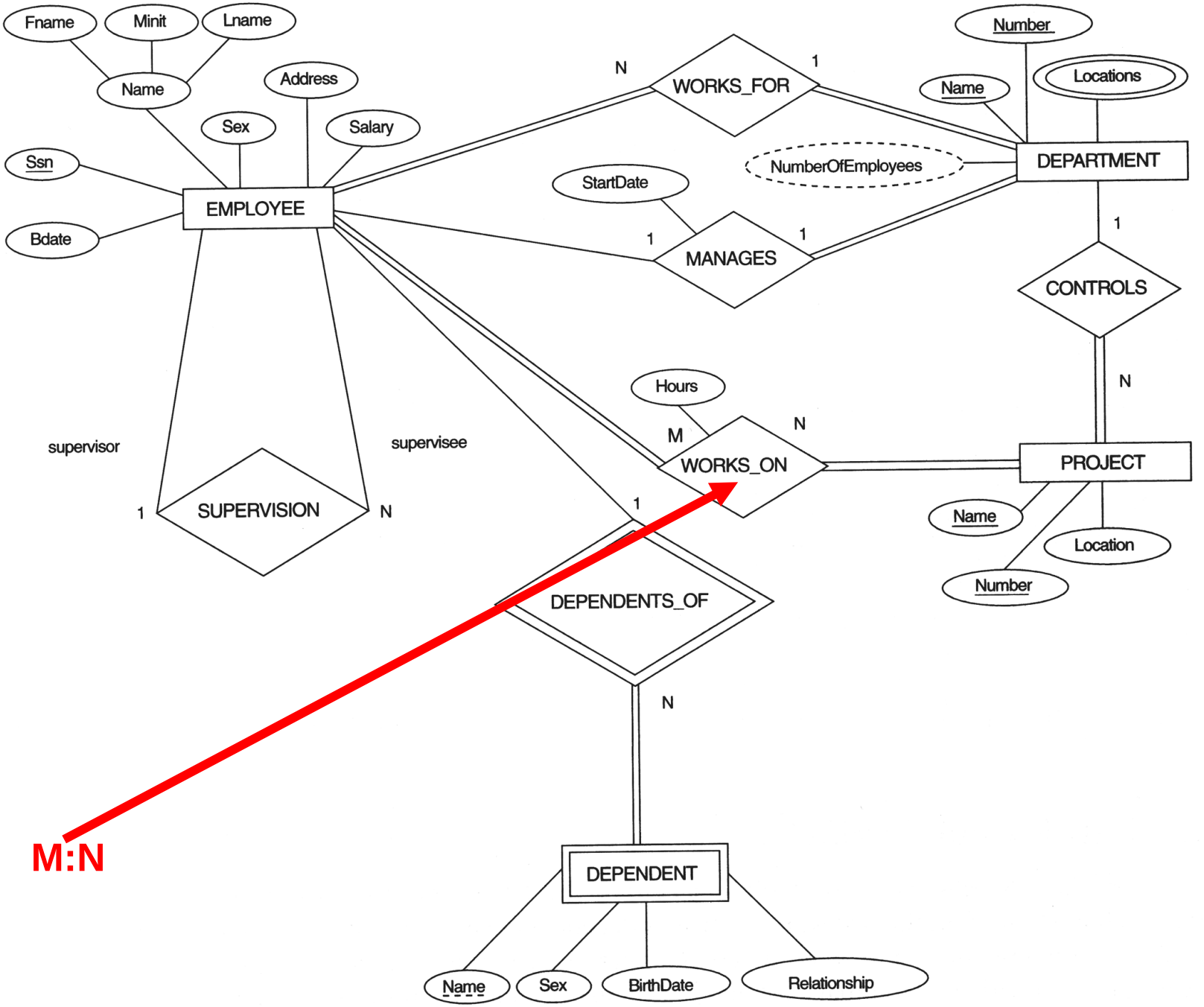
ER-to-Relational Mapping

- N:M binary relationships:
  - An N:M relationship is always represented by a new relation which consists of the prime attributes of both participating entity types together with any attributes of the relationship
  - The combination of the prime attributes will form the primary key of the new relation

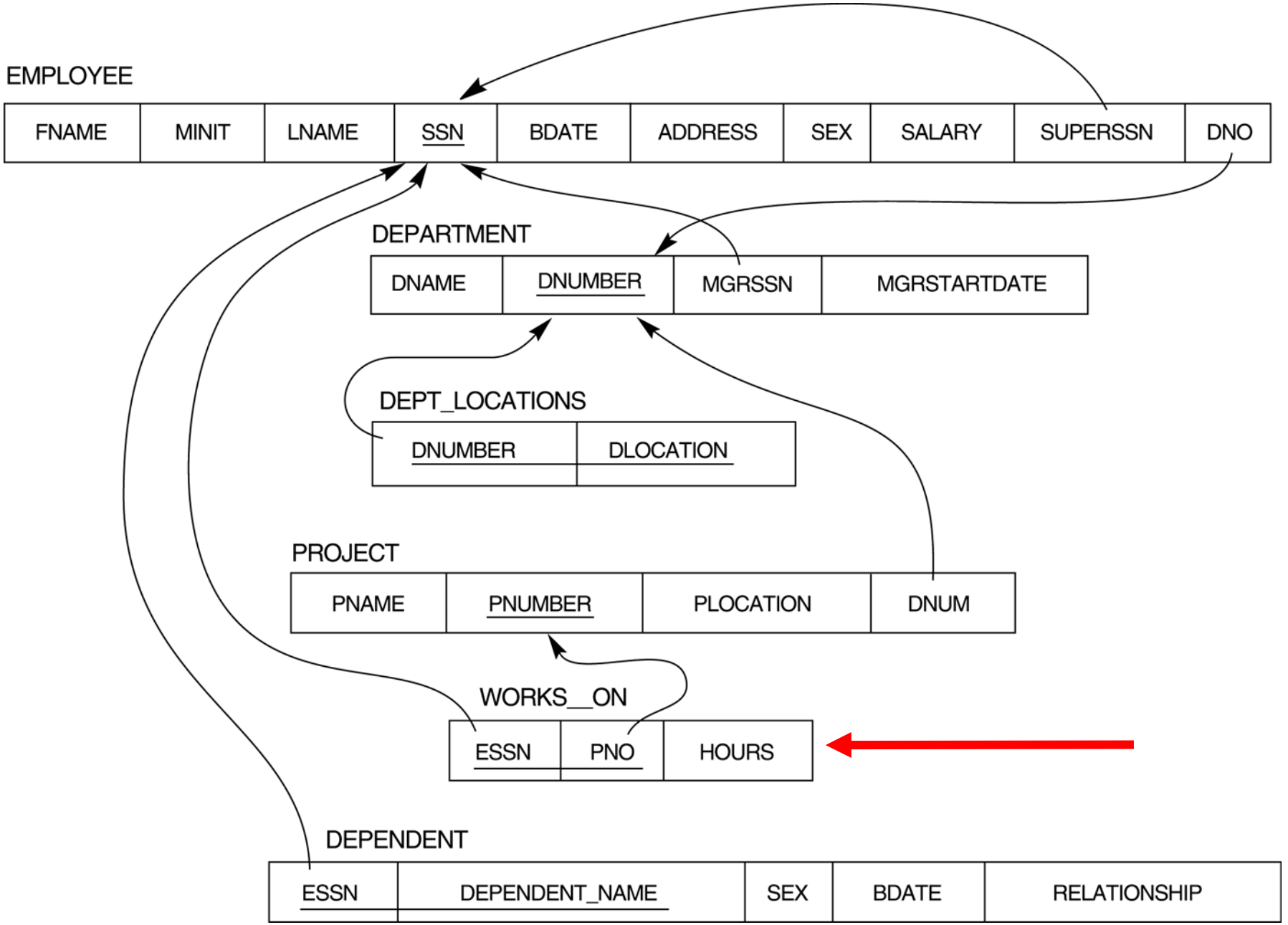
- **Example:** ENROL is an M:N relationship between STUDENT and MODULE. To represent the relationship, we have a new relation:

  ENROL(SNUMBER, MDL-NUMBER, DATE)

The ERD for the COMPANY database



Result of mapping the COMPANY ER schema into a relational schema



ER- & EER-to-Relational Mapping

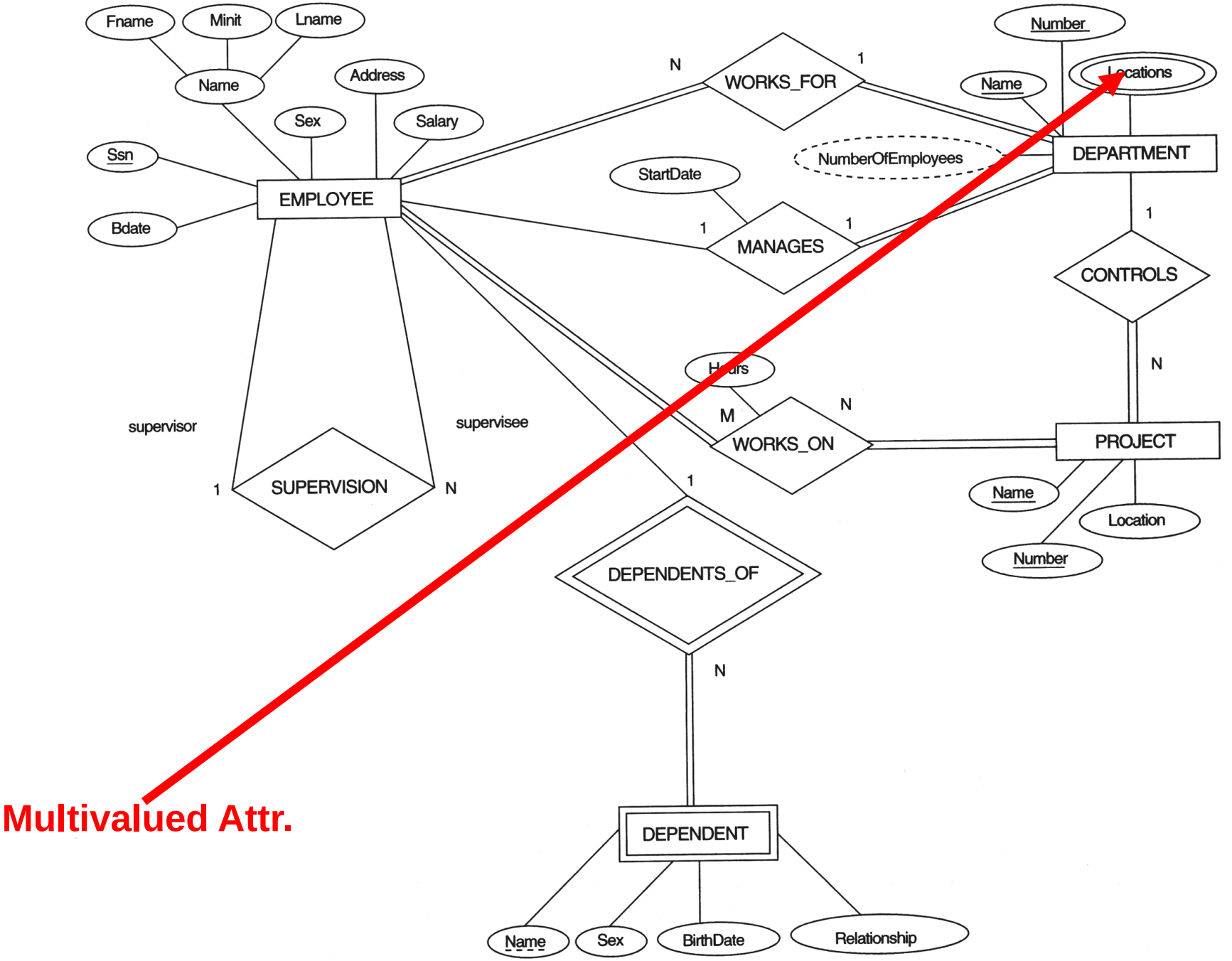
- ER-
  - Step 1: Mapping of Regular Entity Types
  - Step 2: Mapping of Weak Entity Types
  - Step 3: Mapping of Binary 1:1 Relationship Types
  - Step 4: Mapping of Binary 1:N Relationship Types
  - Step 5: Mapping of Binary M:N Relationship Types
  - Step 6: Mapping of Multivalued attributes
  - Step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

ER-to-Relational Mapping

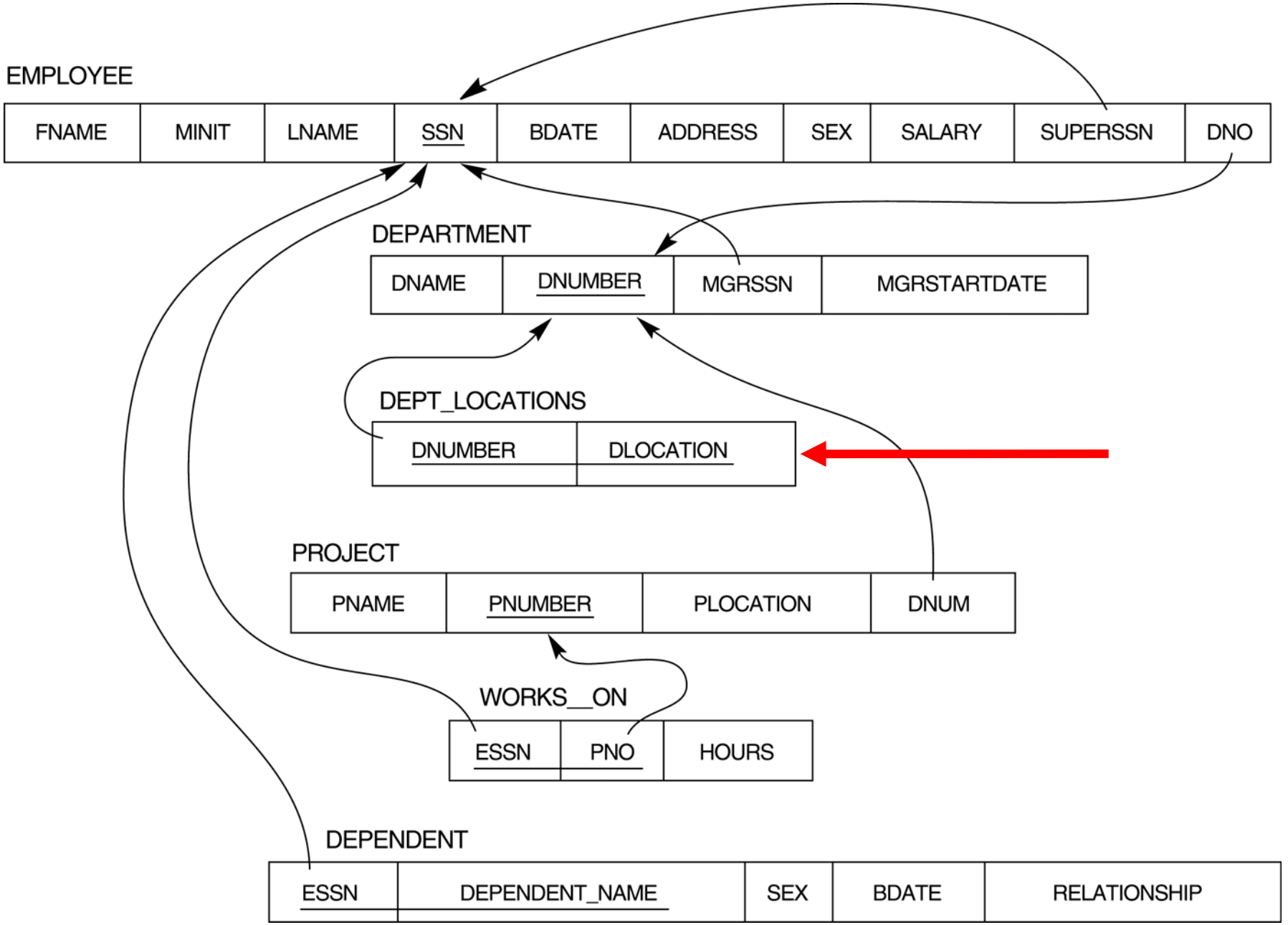
- Step 6: Mapping of Multivalued attributes
  - For each multivalued attribute A, create a new relation R. This relation R will include an attribute corresponding to A, plus the primary key attribute K-as a foreign key in R-of the relation that represents the entity type or relationship type that has A as an attribute
  - The primary key of R is the combination of A and K. If the multivalued attribute is composite, we include its simple components

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

The ERD for the COMPANY database



Result of mapping the COMPANY ER schema into a relational schema



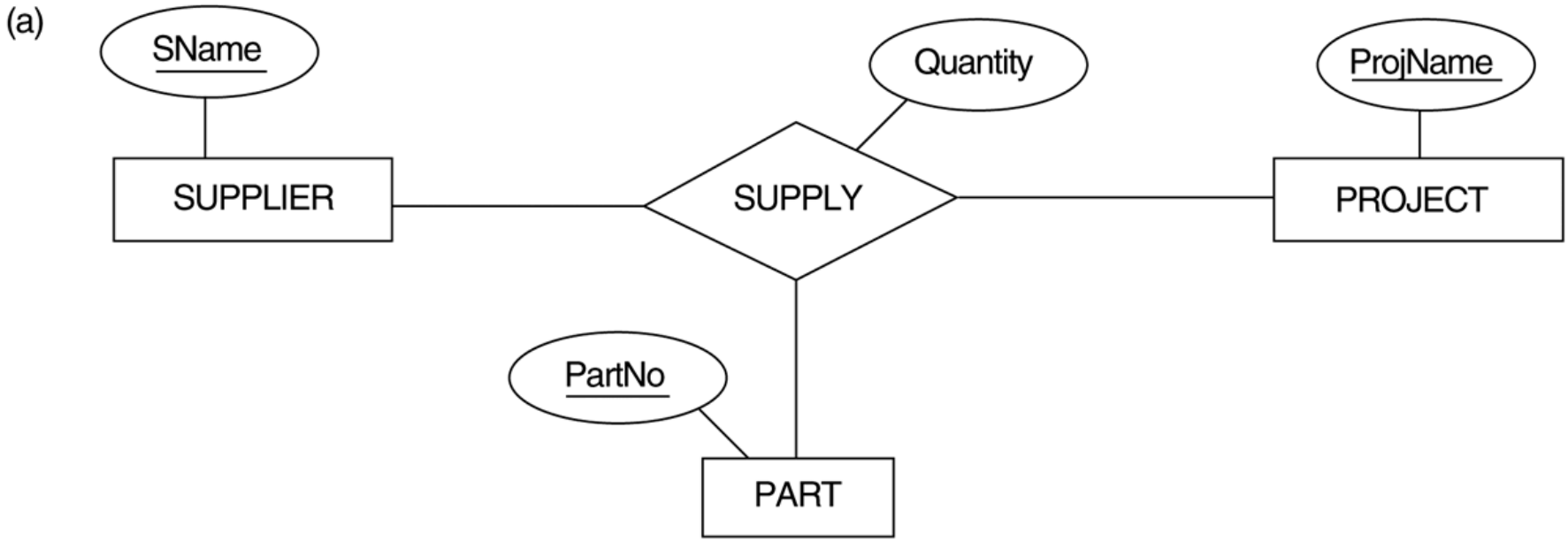
ER-to-Relational Mapping

- Step 7: Mapping of N-ary Relationship Types
  - For each n-ary relationship type R, where n>2, create a new relationship S to represent R
  - Include as foreign key attributes in S the primary keys of the relations that represent the participating entity types
  - Also include any simple attributes of the n-ary relationship type (or simple components of composite attributes) as attributes of S

    Example: The relationship type SUPPY in the ER below. This can be mapped to the relation SUPPLY shown in the relational schema, whose primary key is the combination of the three foreign keys {SNAME, PARTNO, PROJNAME}

ER-to-Relational Mapping

FIGURE 4.11 Ternary relationship types
(a) The SUPPLY relationship



SUPPLIER

SNAME	. . .

PROJECT

PROJNAME	. . .

PART

PARTNO	. . .

SUPPLY

SNAME	PROJNAME	PARTNO	QUANTITY

Note: if the cardinality constraint on any of the entity types E participating in the relationship is 1, the PK should not include the FK attributes that reference the relation E' corresponding to E (see section 4.7 [1])

ER-to-Relational Mapping
Summary of Mapping Constructs & Constraints

Correspondence between ER and Relational Models

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