

Relational Model

CS 377: Database Systems

ER Model: Recap

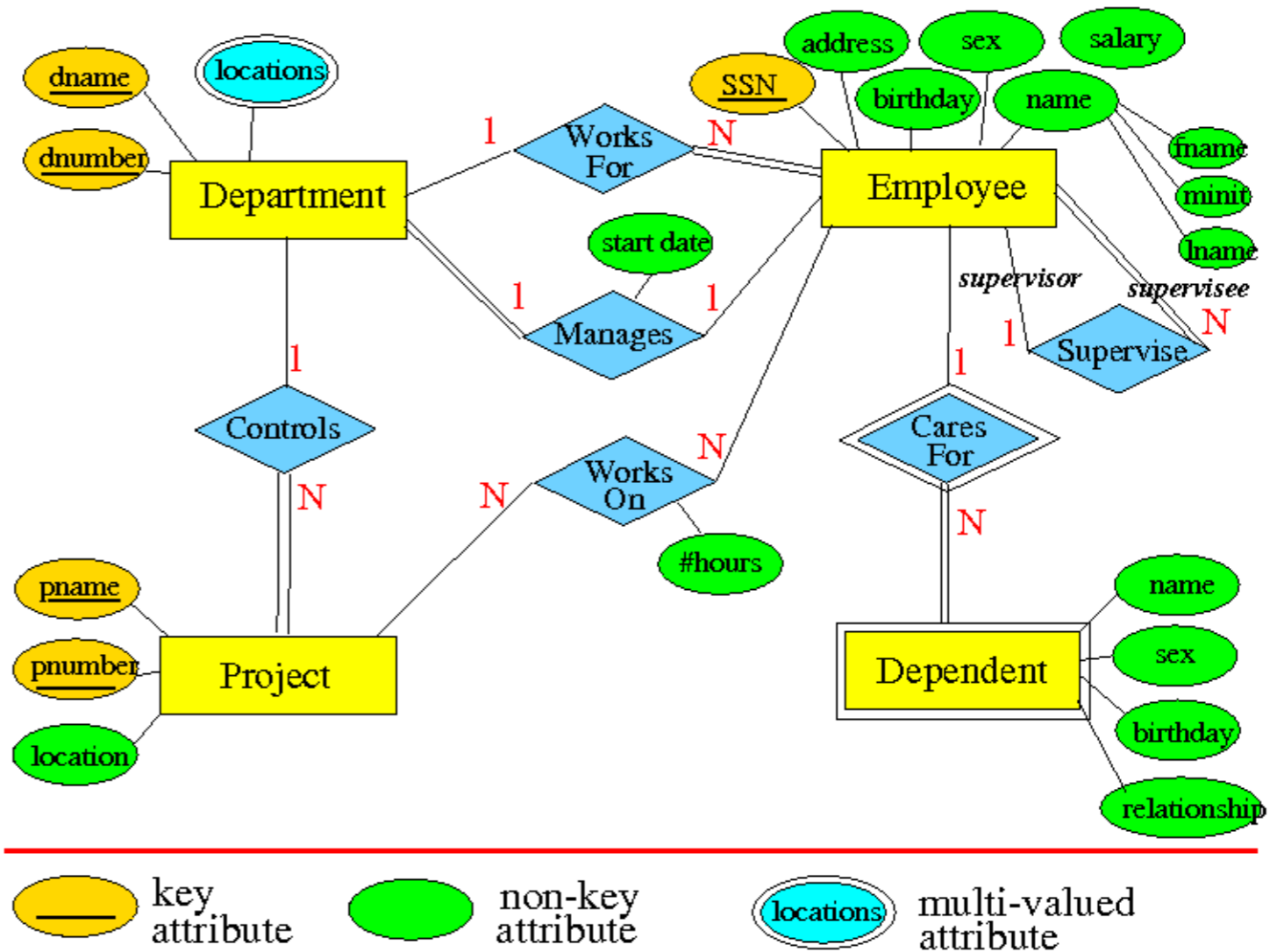
Recap: Conceptual Models

- A high-level description of the database
- Sufficiently precise that technical people can understand it
- But, not so precise that non-technical people can participate in the process

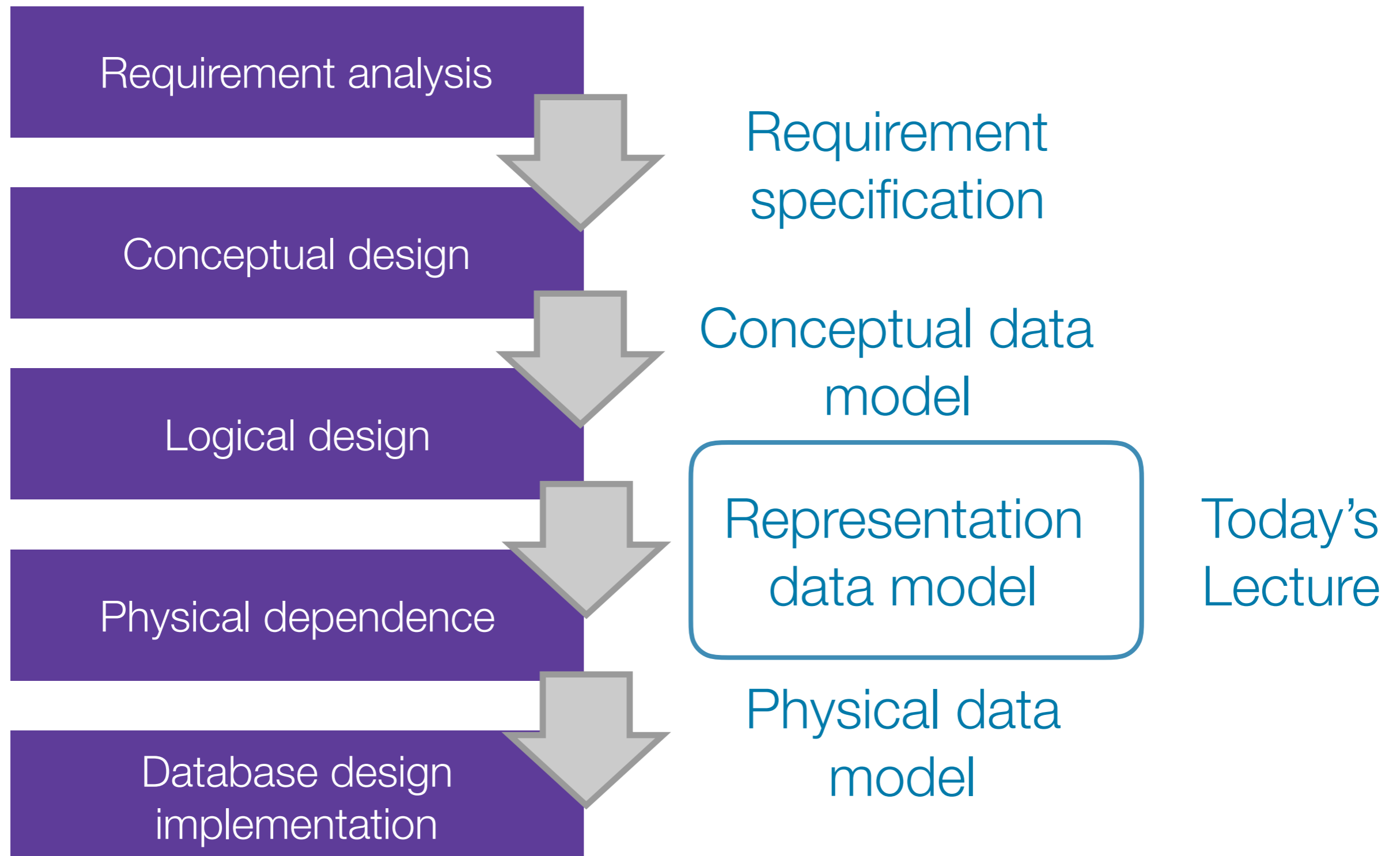
This is where ER models fit in

Recap: ER Model

- Entities
- Attributes
- Relationships
- Degree
- Cardinality
- Participation



Recap: Building a Database System



Today's Lecture

1. Relational Model
2. ER to Relational Mapping
 - Example: Company Database
 - Exercise: Football

Relational Model

- First formal database model
- Introduced by Ted Codd in 1970
- Conceptual basis of relational databases
 - Simple and based on the mathematical relations
 - Declarative method for specifying data and queries
- Previous models include hierarchical and network models

Relational Model: Relation

Data is stored in **tables (relations)**

PRODUCT

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Relational Model: Schema

Relational schema $R(A_1, A_2, \dots, A_n)$: made of of a relation name R and a set of attributes A_1, A_2, \dots, A_n

Product(name, category, price, manufacturer)

PRODUCT

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Relational Model: Attribute

Attribute is a column header in the table

PRODUCT

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Relational Model: Tuple

Tuple or **row** or **record** is a single entry in the table having the attributes specified by the schema

PRODUCT

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Relational Model: Instance

Instance of a relation is a set of tuples or records

PRODUCT

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Relation Definitions

- **Domain:** set of atomic values that are assigned to an attribute (e.g., name: string, category: string, price: real)
 - In practice, the domain is added for each attribute of the relational schema
- **Degree of a relation:** number of attributes in the relation schema
 - this is different than the degree in ER model!

Database: Schema & Instance

- **Database schema:** a collection of relation schemas
- **Instance of a database:** a collection of relation instances
- Schemas are stable over long periods of time while instance changes constantly with data inserts, updates, and deletions

Can view schemas as types while instances as values in a programming language

Relational Model Notation

Notation	Description
$R(A_1, A_2, \dots, A_n)$	Relation schema R of degree n
Q, R, S	Relation names
q, r, s	Relations
t, u, v	Tuples
$t(a_1, a_2, \dots, a_n)$	tuple t of a relation
$t[A_i]$	the value of the attribute A_i in the tuple t
$t[A_i, A_j, A_k]$	value of the attributes A_i, A_j, A_k in the tuple t

Relational Model Constraints

- Restrictions on actual values in a database
- **Inherent model-based constraints** or **implicit constraints**: inherent in the data model (e.g., no duplicate tuples)
- **Schema-based constraints** or **explicit constraints**: can be directly expressed in schemas of the data model
- **Application-based / semantic constraints**, or **business rules**: cannot be directly expressed in schemas and can only be enforced and expressed in the application program

Schema-based Constraints: Domain Constraints

- Value of attribute *A*: atomic value from the domain of *A*
- Typical data types associated with domains

PRODUCT

Should be numeric, not string

Name	Category	Price	Manufacturer
iPad	Tablet	\$399.00	Apple
Surface	Tablet	\$299.00	Microsoft
Kindle	eReader	\$79.00	Amazon
...

Schema-based Constraints: Key Constraints

- No two tuples can have the same combination of values for all their attributes
- **Superkey**: set of attributes in a relation R such that no 2 different tuples will have the same values for that set of attributes


$$\forall t_1, t_2 \in R : t_1[SK] \neq t_2[SK]$$

Schema-based Constraints: Key Constraints

- **Key:** minimal set of attributes in relation R such that no 2 tuples have the same values (i.e., key is a minimal superkey)
- **Candidate key:** any key

PERSON

key(s)



PID	SSN	Name	Address
52032	111-12-2345	John Doe	123 My Street
12345	444-23-1234	Jane Smith	555 South Street
79823	555-67-8910	Tom Thumb	224 First Street
...

Schema-based Constraints: Key Constraints

- **Primary key:** key chosen to be used to identify tuples in a relation
 - Once chosen, you must use that primary key throughout the database
 - Other candidate keys are unique keys
 - Every relation schema must have a primary key
- **Foreign key:** set of attributes inside some relation $R1$ that is a primary key of another relation $R2$

Example: Primary & Foreign Key

PERSON primary key

PID	SSN	Name	Address
52032	111-12-2345	John Doe	123 My Street
12345	444-23-1234	Jane Smith	555 South Street
79823	555-67-8910	Tom Thumb	224 First Street
...

PURCHASE primary key foreign key

TID	PID	Product	Price
123456778	52032	iPad Air 2	\$399.00
123470901	52032	Kindle	\$79.00
234096701	79823	Surface	\$499.00
...

Schema-based Constraints: Entity Integrity

- Primary key values cannot have NULL values
- Primary key is used to identify a tuple
- NULL value means not applicable or not available which hinders the ability to identify a tuple

PERSON

PID	SSN	Name	Address
52032	111-12-2345	John Doe	123 My Street
NULL	444-23-1234	Jane Smith	555 South Street
...

↑
Not allowed!

Schema-based Constraints: Referential Integrity

- A tuple in one relation (t_1 in R_1) that refers to another relation (t_2 in R_2) must refer to an existing tuple in that relation (t_2 must exist): $t_1[FK] = t_2[PK]$
- R_1 is the referencing relation and R_2 is the referenced relation

PURCHASE

tuple must exist in PERSON table

TID	PID	Product	Price
123456778	52032	iPad Air 2	\$399.00
123470901	52032	Kindle	\$79.00
234096701	79823	Surface	\$499.00
...

Relational Model Virtues

- Physical & logical independence
- Declarative
- Simple, elegant and clean: everything is a relation

Why did it take so long?
Doubted it could be done efficiently

ER Model vs Relational Model

ER model (conceptual model)

- Several concepts: entities, relationships, attributes
- Well-suited for capturing application requirements
- Not well-suited for computer implementation

Relational model (implementation model)

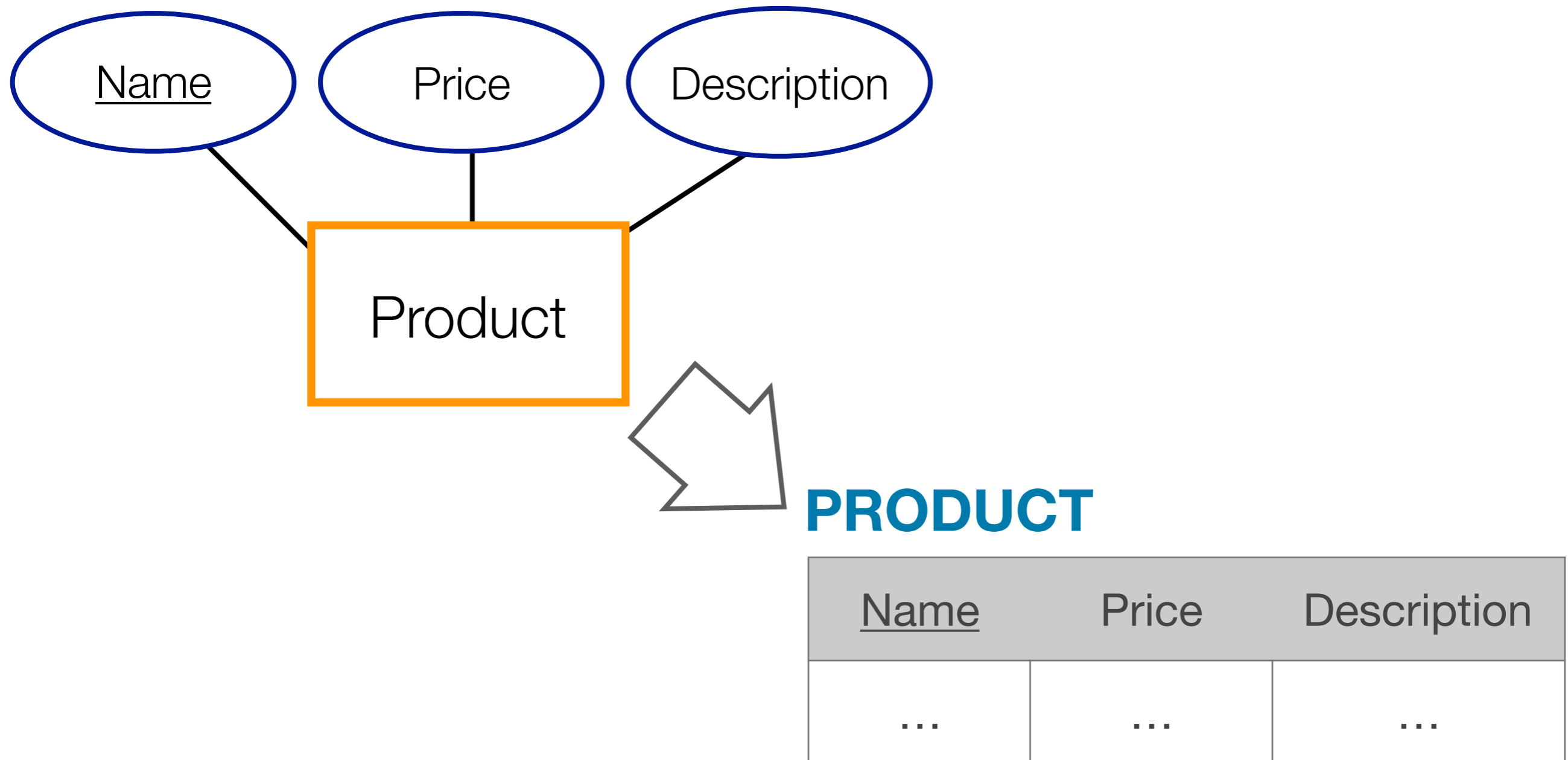
- Single concept: relation (not same as mathematical concept!)
- Everything is represented with a collection of tables
- Well-suited for efficient manipulations on computers

ER-to-Relational Mapping: Step 1

Convert Entities to Relations

- Basic case: entity set $E \rightarrow$ relation with attributes of E
- Special case: weak entity & multi-valued attributes

Basic Case: Entity to Relation



Special Case: Multi-valued Attribute

- Naive storing of multi-valued attributes:
 - Variable-length records causes inefficient in storage
 - Multiple tuples leads to lots of redundancy

STORE

<u>ID</u>	<u>Location</u>
1	Cumberland Mall
1	Lenox Square
2	Lenox Square
2	Cumberland Mall
...	...

STORE

<u>ID</u>	Locations
1	{Cumberland Mall, Lenox Square}
2	{Cumberland Mall, Lenox Square}
...	...

Special Case: Multi-valued Attribute

- Naive storing of multi-valued attributes:
 - Variable-length records causes inefficient in storage
 - Multiple tuples leads to lots of redundancy
- Use the key concept
 - Convert multi-valued attribute to new relation X
 - Add foreign key to that relation

Special Case: Multi-valued Attribute

STORE_LOC

<u>locID</u>	Location
1	Lenox Square
2	Cumberland Mall
...	...

STORE

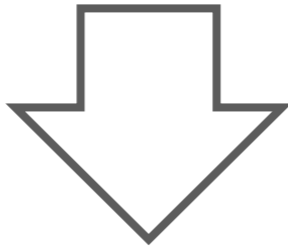
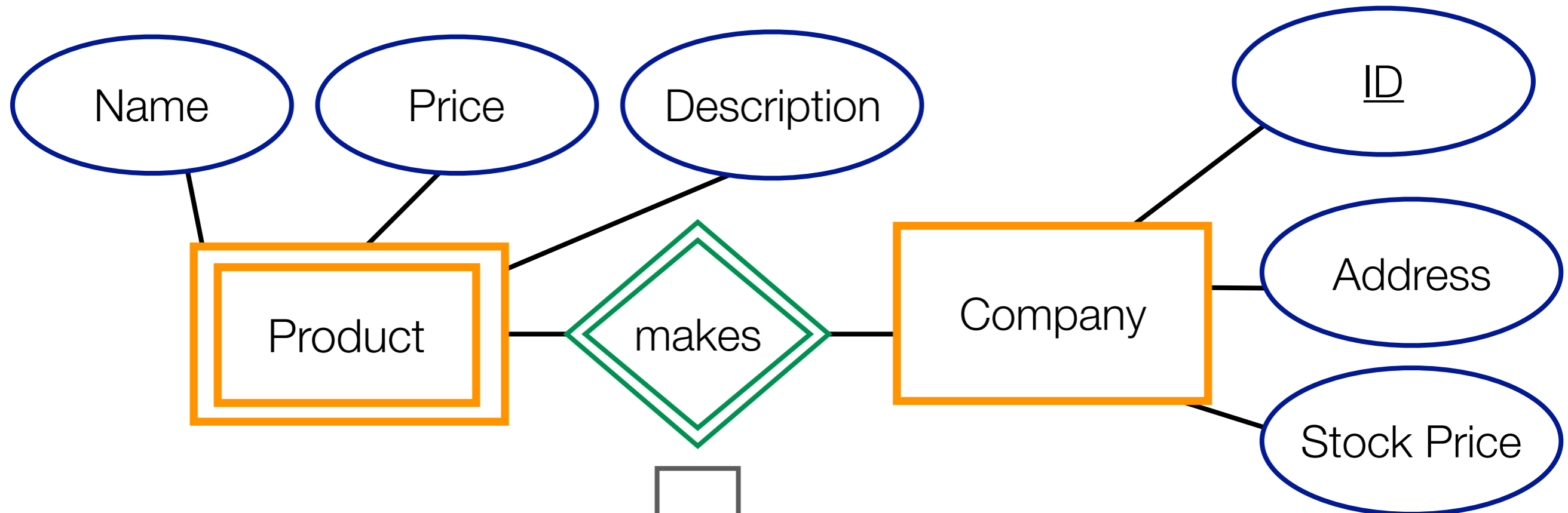
foreign key: referential integrity

<u>sNumber</u>	<u>locID</u>
1	1
1	2
2	1
2	2
...	...

Special Case: Weak Entity

- Weak entity does not have a key \rightarrow relation violation
- Borrow key from the other entity in the identifying relationship (E) and add it to the weak entity (W)
- Result: key of weak entity consists of the key of the related entity and some identifying attribute of the weak entity

Special Case: Weak Entity



PRODUCT

<u>Name</u>	<u>cID</u>	Price	Manufacturer
...

foreign key

weak entity key

ER-to-Relational Mapping: Step 2

Map Relationships to Relations

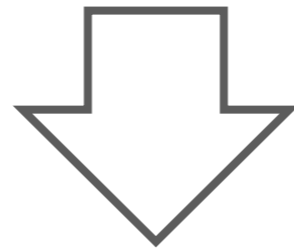
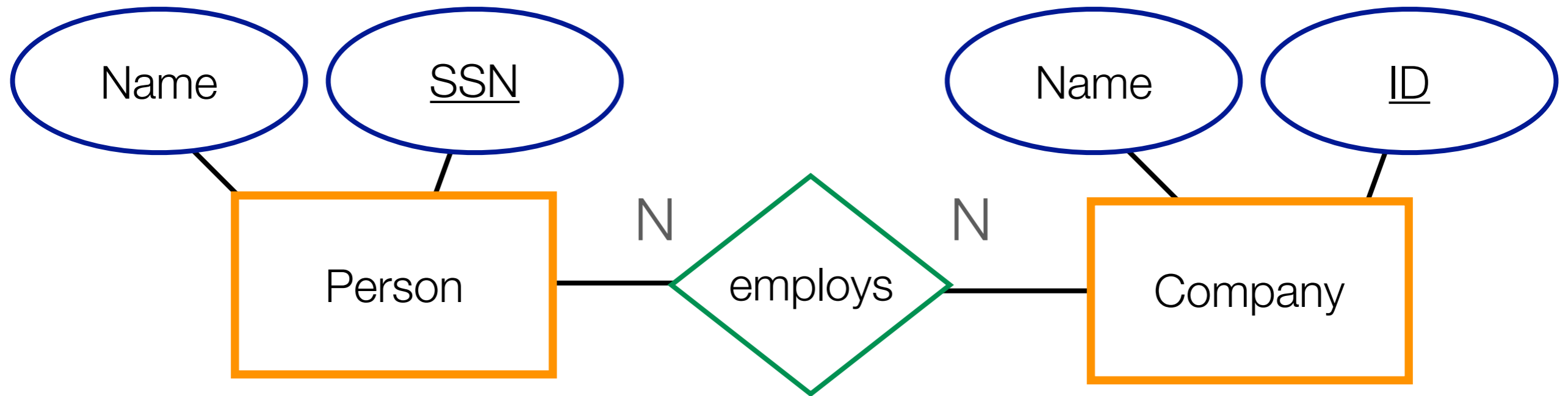
- Basic case: relationship $R \rightarrow$ relation with attributes being keys of related entity sets and attributes of R
- Special case: expansion, merging, & n-ary relationship types

Basic Case: Relationship to Entity

Create a new relation (S — R — T)

- New tuples of relationship R stored in this table with foreign keys from the entities S and T
- Pro: always possible
- Con: Increasing the number of relations

Basic Case: Relationship to Entity



EMPLOY

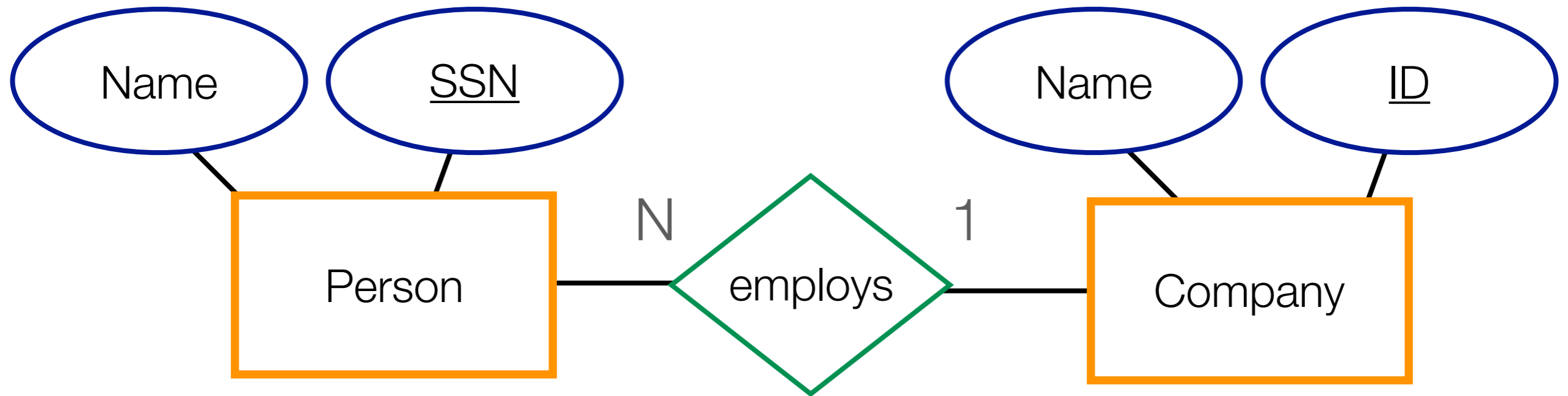
<u>SSN</u>	<u>cID</u>
...	...

Special Case: Expansion

Expand an existing relation (foreign key approach)

- Tuples of relationship are stored inside the table of an existing entity
- Use key of that entity to store tuples of the relationship
- Pro: only makes an existing relation a bit larger
- Con: not always possible

Special Case: Expansion



PERSON

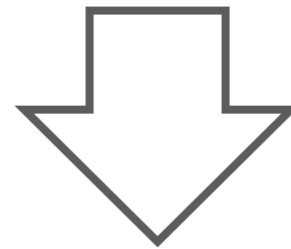
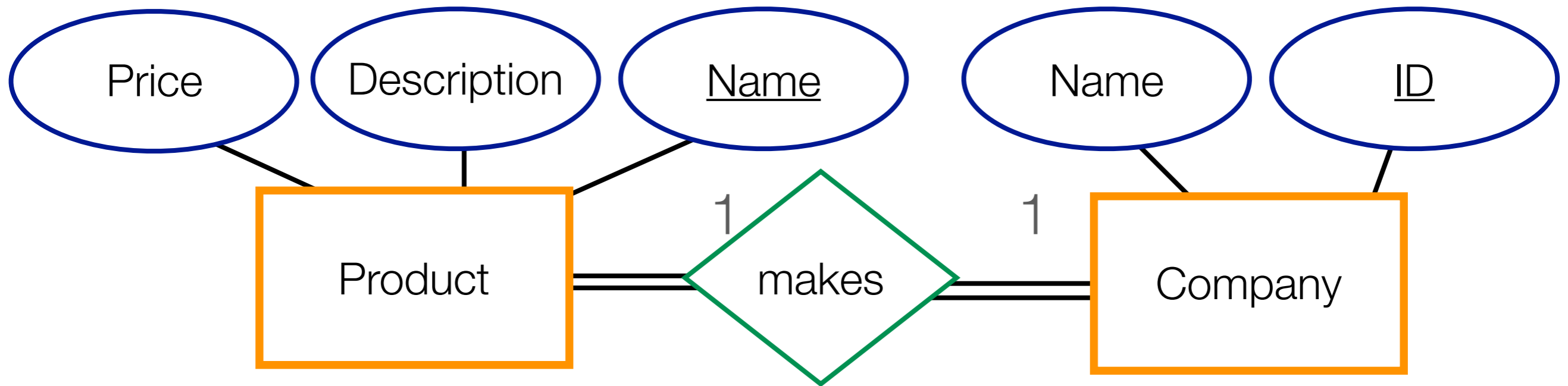
<u>SSN</u>	Name	cID
...

Special Case: Merging

Merge two existing relations

- Merge two entity types and relationship into one relation
- Only possible in 1:1 mapping and both have total participation
- Pro: reduction of relations
- Con: rarely used

Special Case: Merging



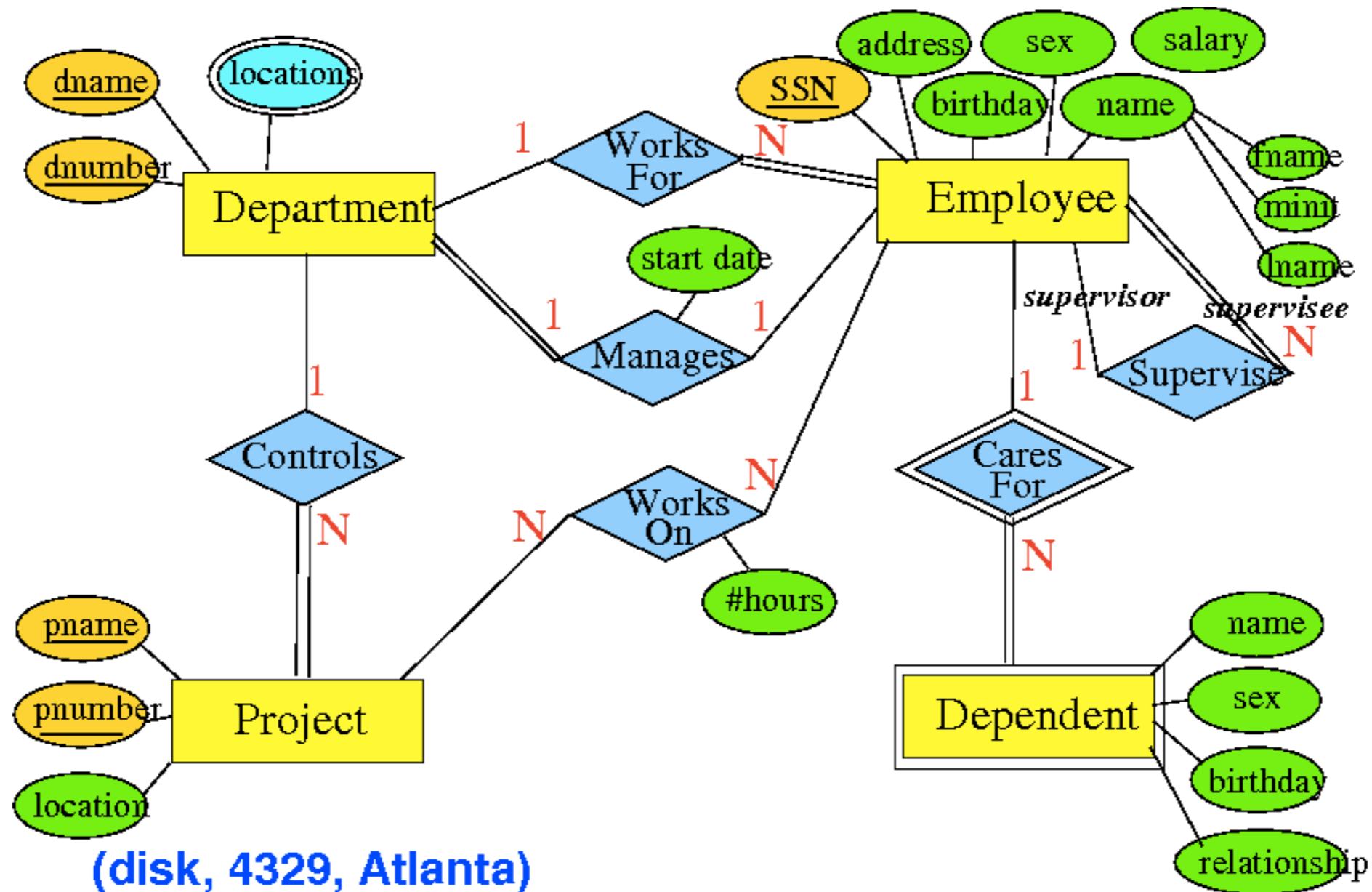
COMPANY_PROD

<u>pName</u>	cName	cID	Price	Description
...		

Relation Mapping Design Principles

- Relationship R where Entity1: Entity2 = 1:N \rightarrow expand the relation that represents Entity2
- Relationship R where Entity1: Entity2 = 1:1 \rightarrow expand either Entity1 or Entity2
- Avoid having attributes that can take on NULL values (e.g., expand a relationship where entity is total participation over entity with partial participation)

Example: Company database

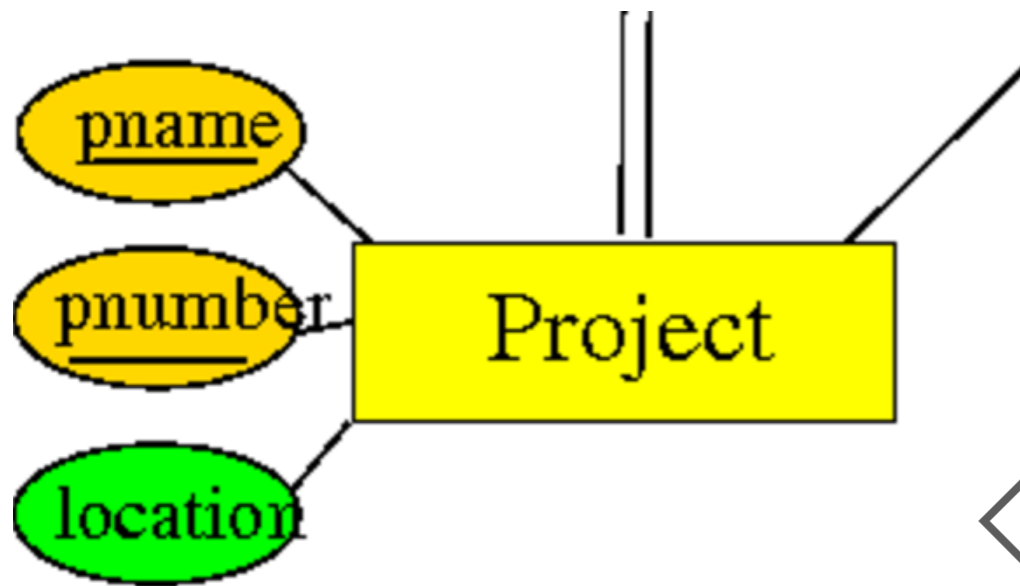


(disk, 4329, Atlanta)
 (CPU, 1562, Boston)
 (Printer, 7862, Denver)

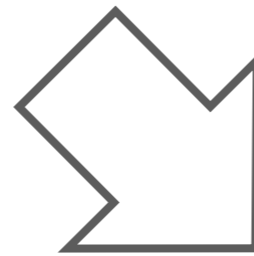
...

Example from Prof Cheung's lectures

Step 1: Project Entity



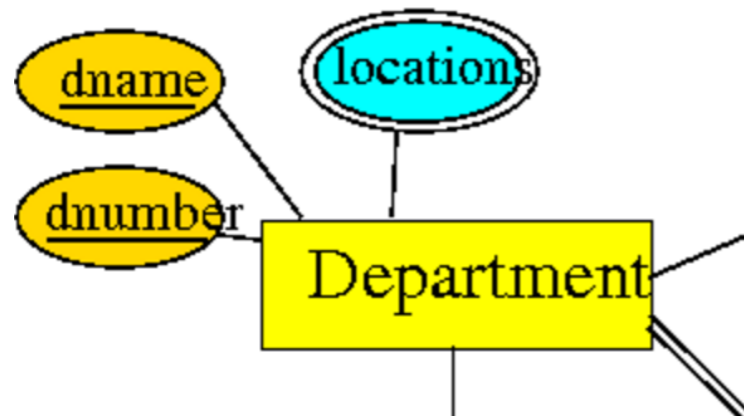
Pick one of the candidate keys to be primary key



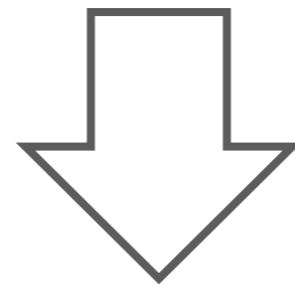
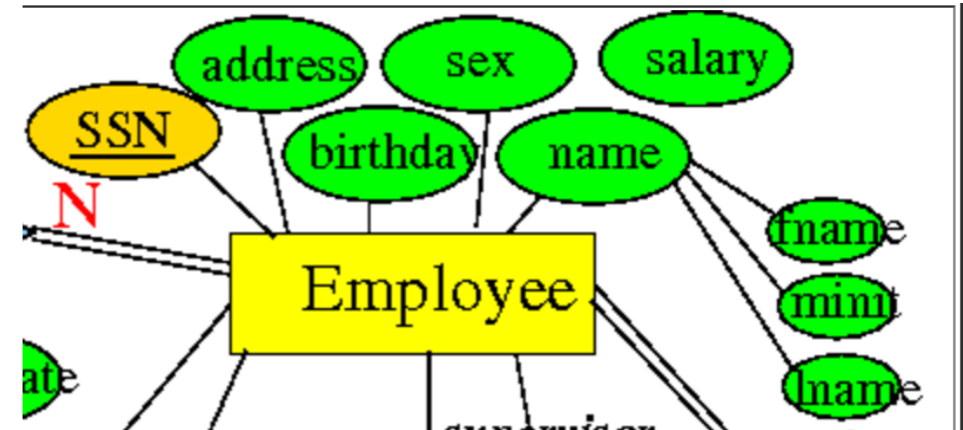
PROJECT

PName	<u>PNumber</u>	Location
...

Step 1: Employee & Department Entities



...



EMPLOYEE

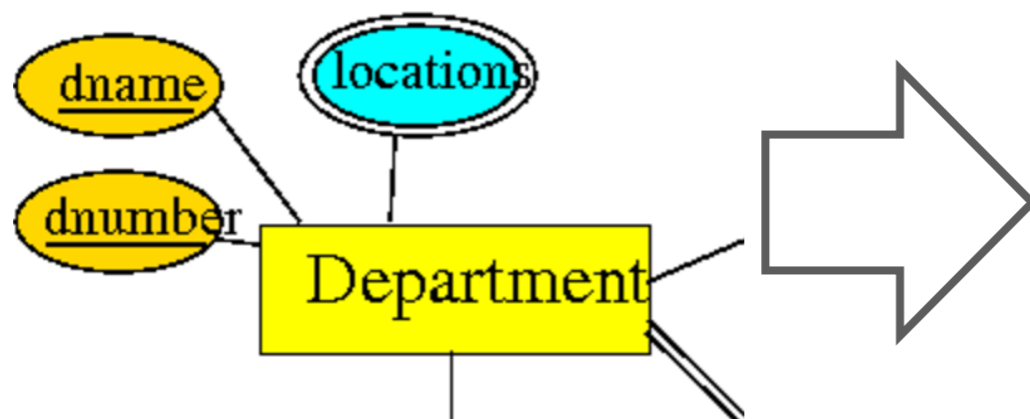
<u>SSN</u>	FName	MI	LName	Sex	Address	BDate	Salary
...					

DEPARTMENT

<u>DNumber</u>	DName	{Locations}
...

Attribute values are not ATOMIC!

Step 1: Department Location



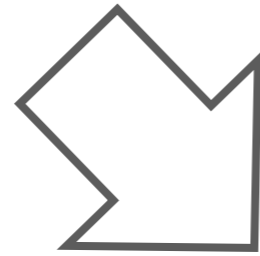
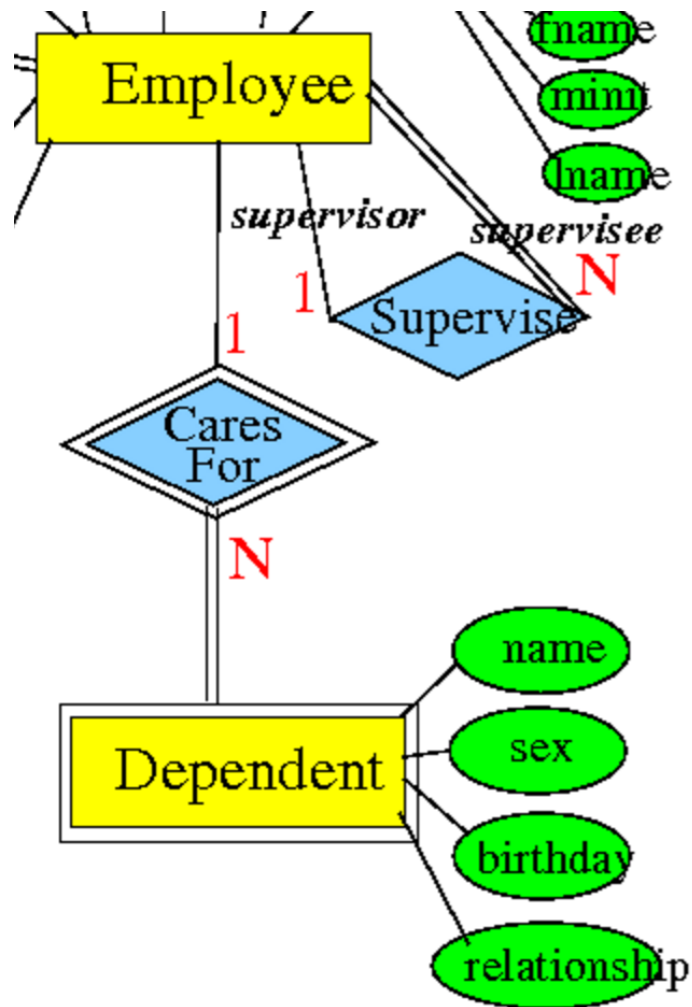
DEPARTMENT

<u>DName</u>	<u>DNumber</u>
Manufacturing	D1234
Research	D7652
...	...

DEPT_LOC

<u>DNumber</u>	<u>Location</u>
D1234	Atlanta
D1234	New York
D1234	Denver
D7652	San Jose
D7652	Austin
...	...

Step 1: Dependent Entity

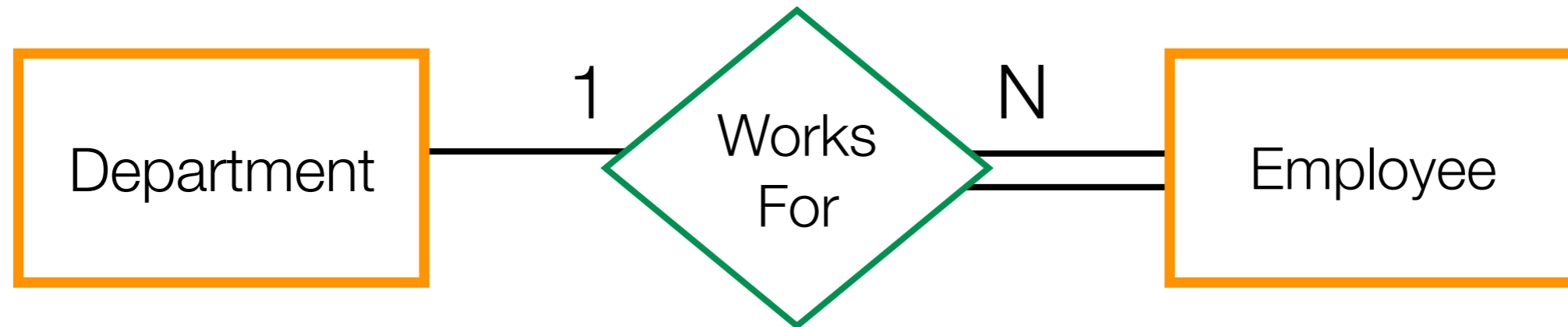


Necessary to identify weak entity

DEPENDENT

<u>ESSN</u>	<u>FName</u>	Sex	BDate	Relationship
	

Step 2: WorksFor Relationship



Use expansion by adding attribute to Employee because 1:N relationship

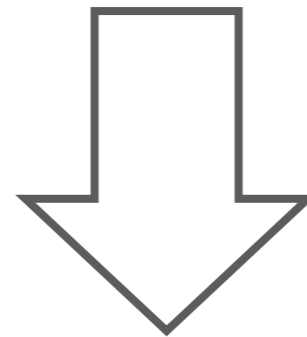
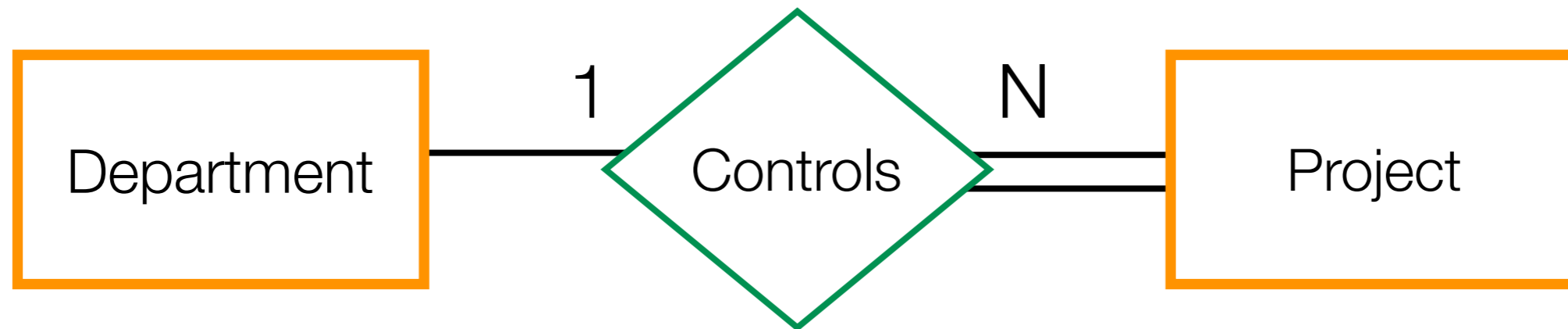
EMPLOYEE

<u>SSN</u>	FName	MI	LName	Sex	Address	BDate	Salary	DNo
...						

DEPARTMENT

DName	<u>DNumber</u>
...	...

Step 2: Controls-Project

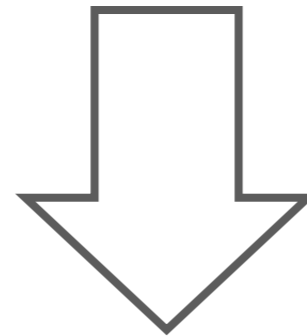
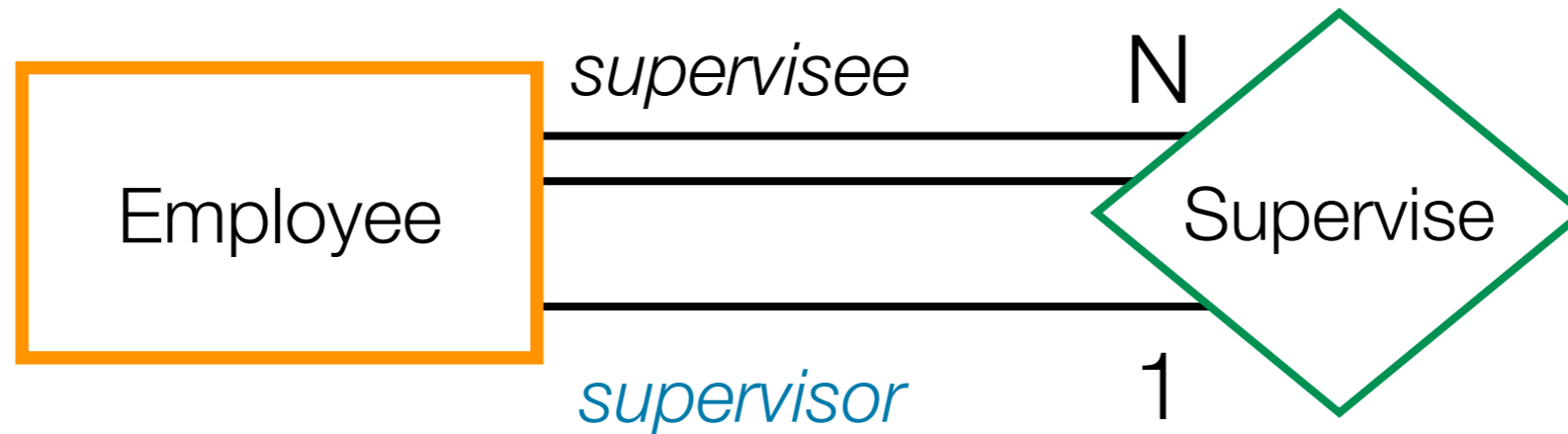


Expand Project
because of the 1:N
relationship

PROJECT

PName	<u>PNumber</u>	Location	DNum
...	

Step 2: Supervisor

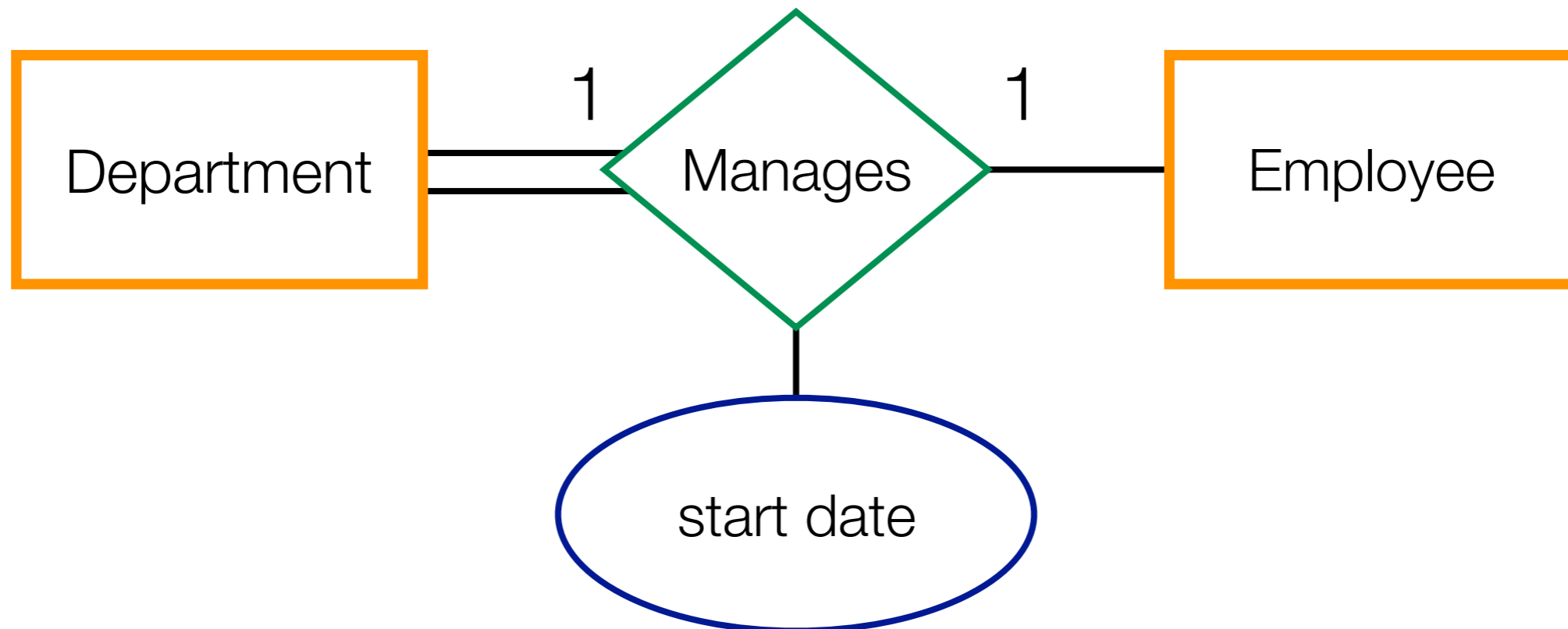


Expand supervisor because of the 1:N relationship

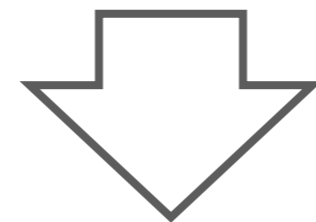
EMPLOYEE

<u>SSN</u>	FName	MI	LName	Sex	Address	BDate	Salary	superSSN	DNo
...							

Step 2: Manager



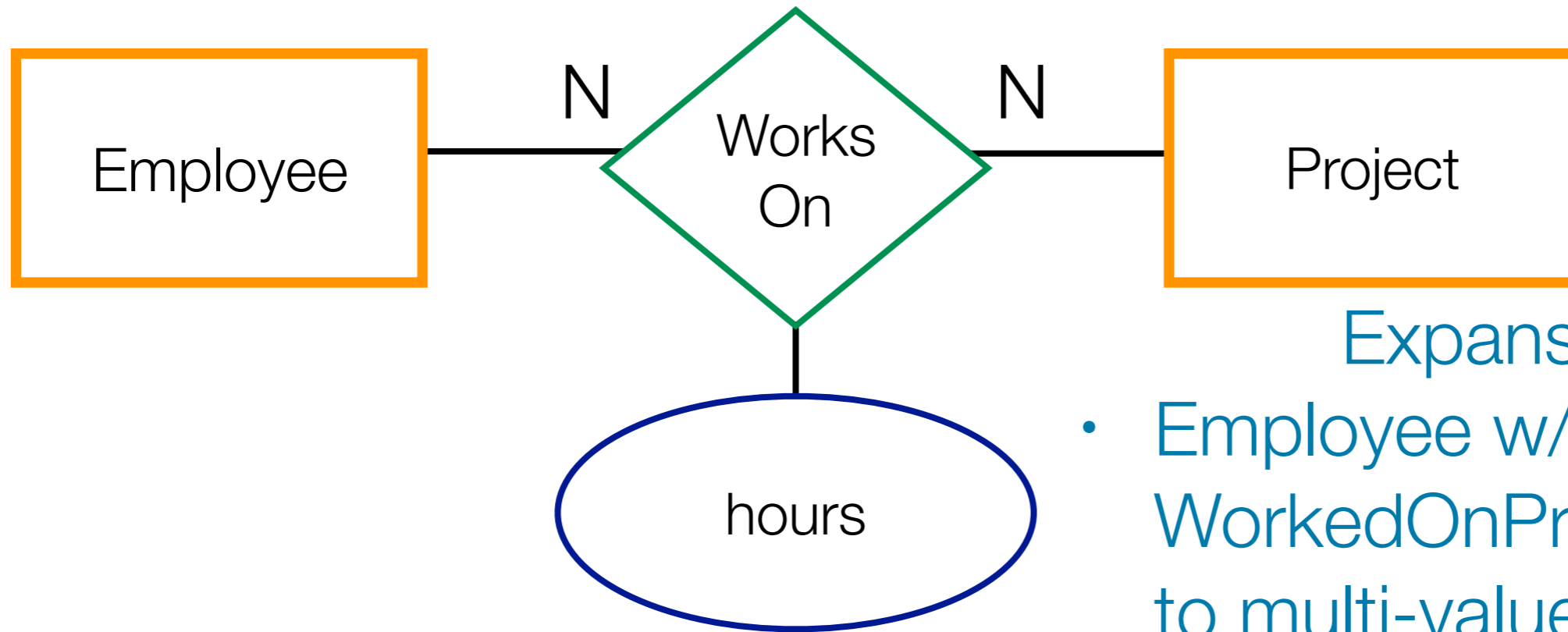
Expand Department
because of total
participation



DEPARTMENT

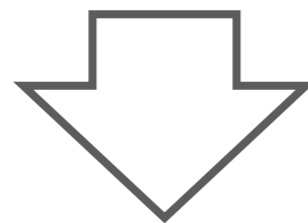
DName	<u>DNumber</u>	mgrSSN	mgrStart
...	...		

Step 2: Works On



Expansion?

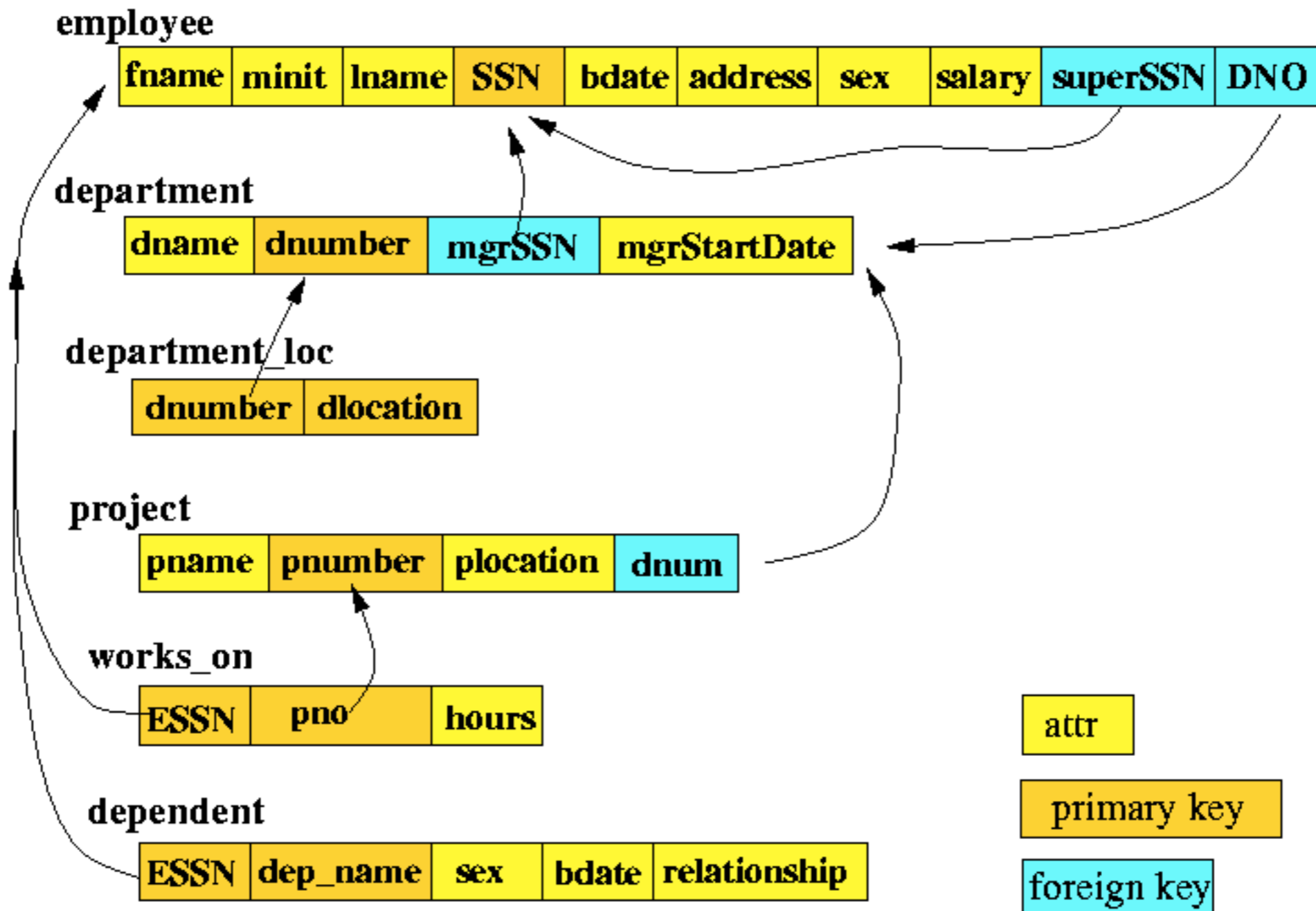
- Employee w/ attribute WorkedOnProject leads to multi-valued attribute
- Project w/ attribute WorkerSSN also results in multi-valued attribute



WORKS_ON

<u>ESSN</u>	<u>PNO</u>	Hours
...	...	

Example: Full Relational Model



Mapping Summary

ER Model	Relational model
Entity type	Entity relation
1:1 or 1:N relationship	Expand (or create R relation)
M:N relationship	Create R relation with two foreign keys
n-ary relationship type	Create R relation with n foreign keys
Simple attribute	Attribute
Composite attribute	Set of simple component attributes
Multivalued attribute	Relation and foreign key
Key attribute	Primary (or secondary) key

Exercise: Football ER Model

Relational Model: Recap

- Relational Model
 - Relation, attributes
 - Schema vs instance
 - Relational model constraints
- ER to Relational
 - Entity set, relationship \rightarrow relation

