

ER→Relational Mapping

- ER to Relational Mapping
- Relational Model vs ER Model
- Mapping Strong Entities
- Mapping Weak Entities
- Mapping N:M Relationships
- Mapping 1:N Relationships
- Mapping 1:1 Relationships
- Mapping n-way Relationships
- Mapping Composite Attributes
- Mapping Multi-valued Attributes (MVAs)
- Mapping Subclasses

❖ ER to Relational Mapping

Reminder: a useful strategy for database design:

- perform initial data modelling using ER
(conceptual-level modelling)
- transform conceptual design into relational model
(implementation-level modelling)

A formal mapping exists for ER model \rightarrow Relational model.

This maps "structures"; but additional info is needed, e.g.

- concrete domains for attributes and other constraints

❖ Relational Model vs ER Model

Correspondences between relational and ER data models:

- $\text{attribute(ER)} \cong \text{attribute(Rel)}$, $\text{entity(ER)} \cong \text{tuple(Rel)}$
- $\text{entity set(ER)} \cong \text{relation(Rel)}$, $\text{relationship(ER)} \cong \text{relation(Rel)}$

Differences between relational and ER models:

- Rel uses relations to model entities *and* relationships
- Rel has no composite or multi-valued attributes (only atomic)
- Rel has no object-oriented notions (e.g. subclasses, inheritance)

Note that ...

- not all aspects of ER can be represented exactly in a relational schema
- some aspects of relational schemas (e.g. domains) do not appear in ER

❖ Mapping Strong Entities

An entity set E with atomic attributes a_1, a_2, \dots, a_n

maps to

A relation R with attributes (columns) a_1, a_2, \dots, a_n

Example:

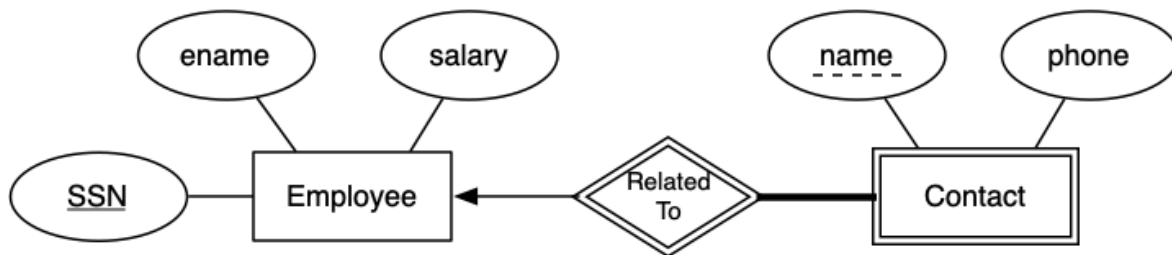


Note: the key is preserved in the mapping.

❖ Mapping Weak Entities

Example:

ER Model

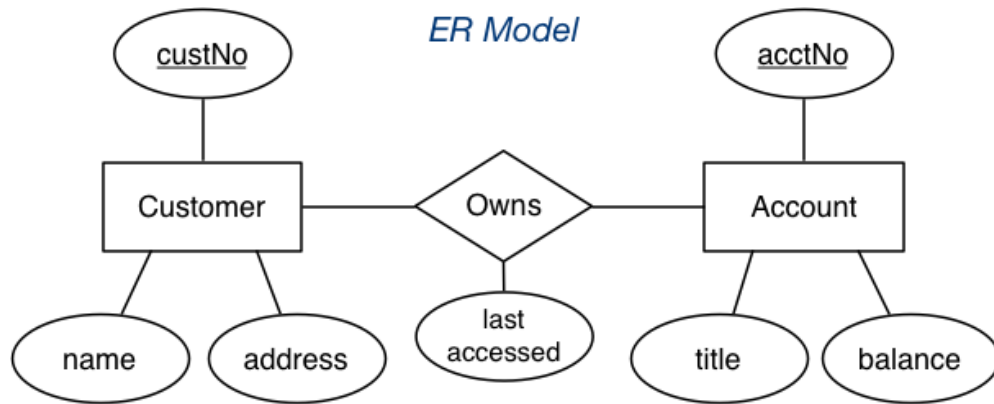


Relational Version

Employee	SSN	ename	salary
Contact	SSN	name	phone

❖ Mapping N:M Relationships

Example:

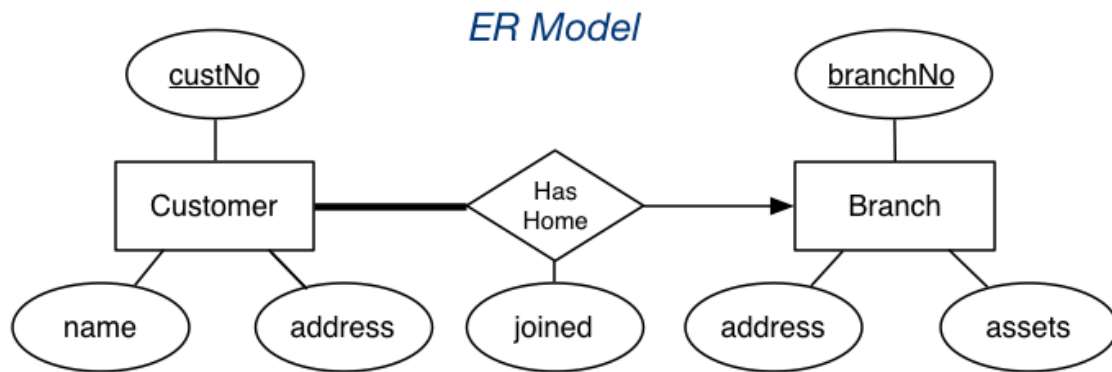


Relational Version

Customer	custNo	name	address
Account	acctNo	title	balance
Owns	acctNo	custNo	lastAccessed

❖ Mapping 1:N Relationships

Example:



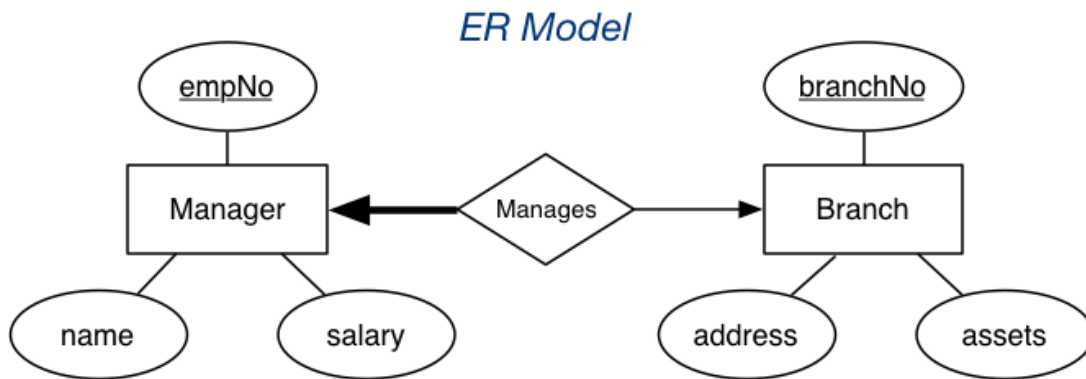
Relational Version

Customer	custNo	name	address	branchNo	joined
----------	---------------	------	---------	-----------------	--------

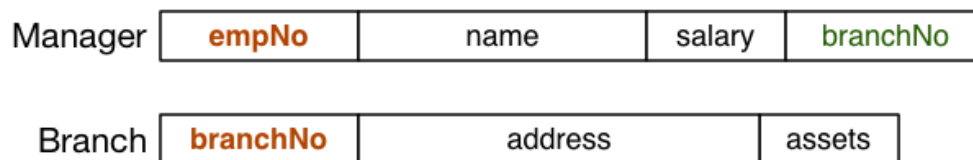
Branch	branchNo	address	assets
--------	-----------------	---------	--------

❖ Mapping 1:1 Relationships

Example:

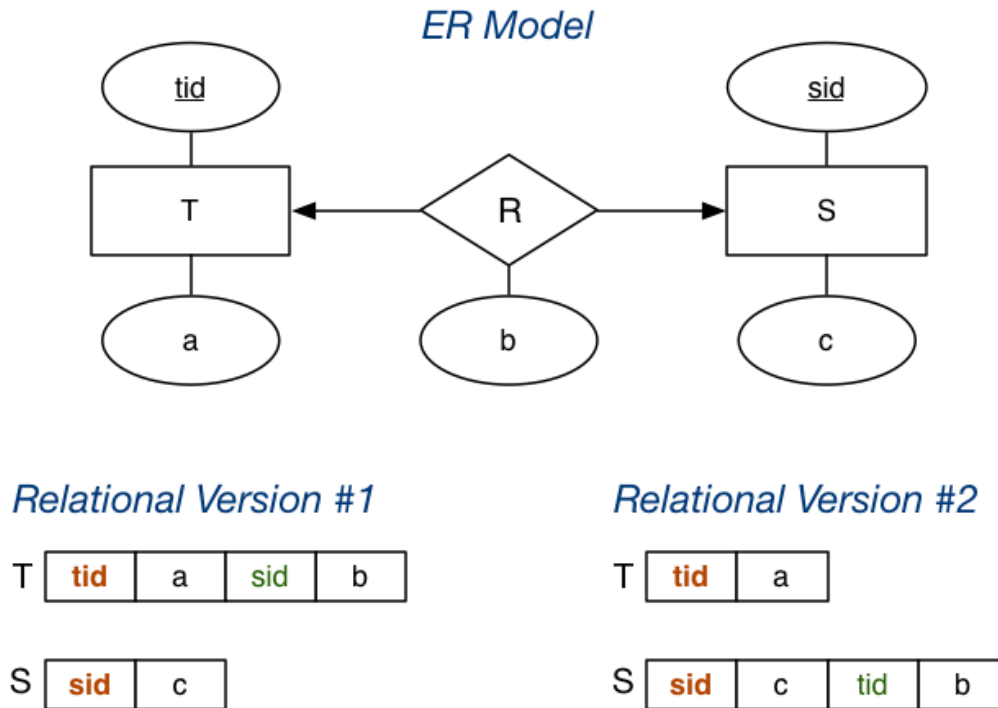


Relational Version



❖ Mapping 1:1 Relationships (cont)

If there is no reason to favour one side of the relationship ...



❖ Mapping n-way Relationships

Relationship mappings above assume binary relationship.

If multiple entities are involved:

- $n:m$ generalises naturally to $n:m:p:q$
 - include foreign key for each participating entity
 - include any other attributes of the relationship
- other multiplicities (e.g. $1:n:m$) ...
 - need to be mapped the same as $n:m:p:q$
 - so not quite an accurate mapping of the ER

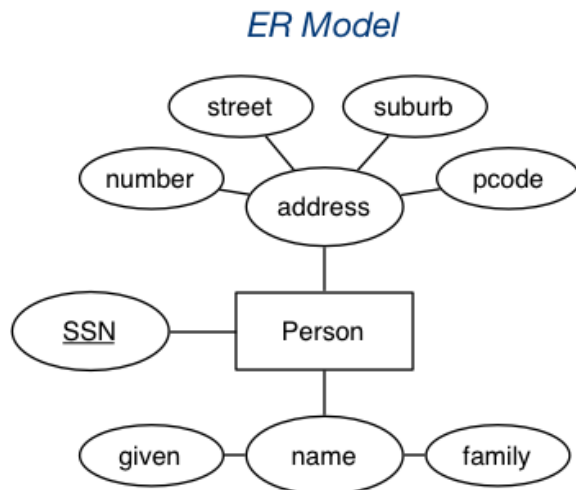
Some people advocate converting n-way relationships into:

- a new entity, and a set of n binary relationships

❖ Mapping Composite Attributes

Composite attributes are mapped by concatenation or flattening.

Example:



Relational Version #1

Person

SSN	name	address
------------	------	---------

Relational Version #2

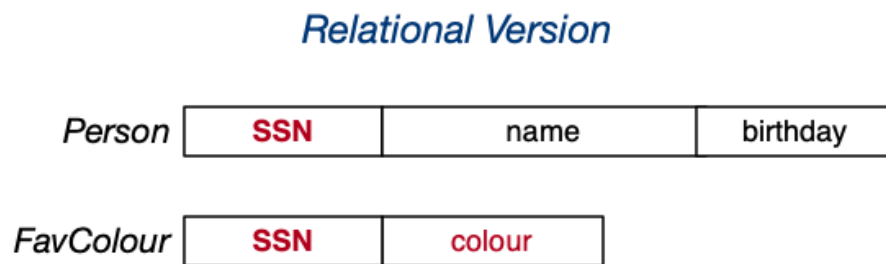
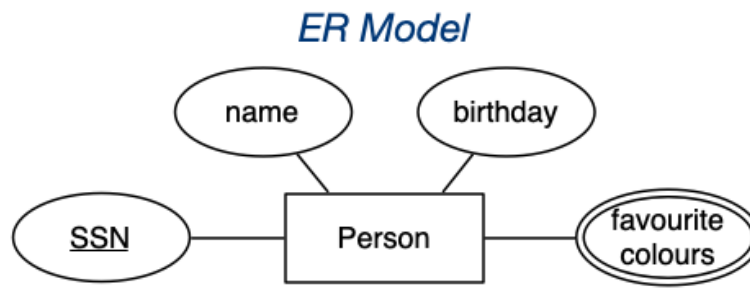
Person

SSN	given	family	
.....	number	street	suburb	pcode

❖ Mapping Multi-valued Attributes (MVAs)

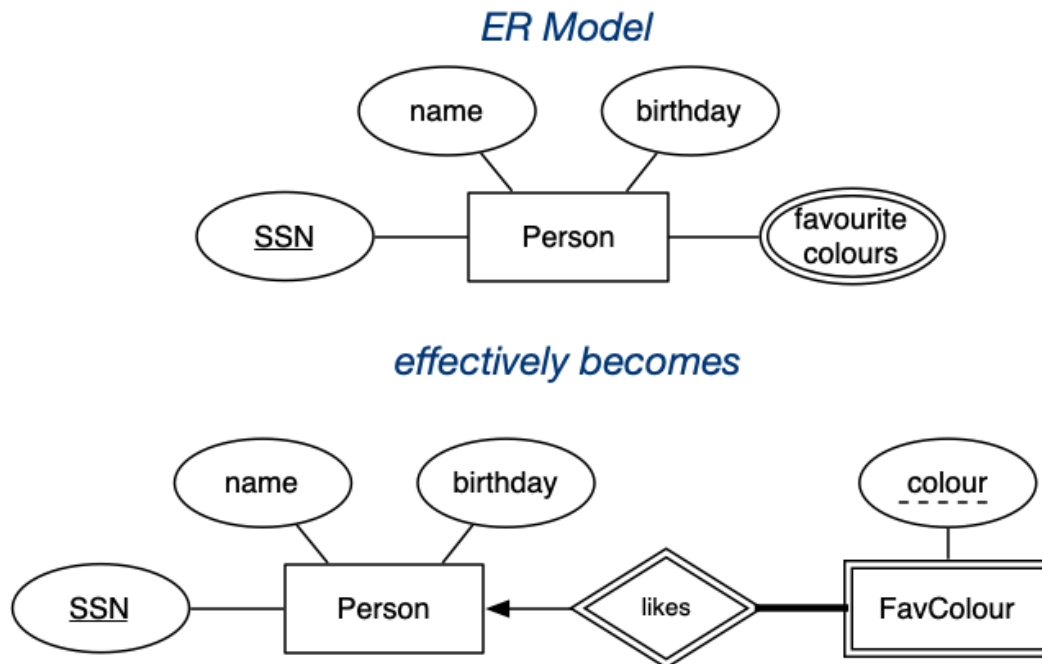
MVAs are mapped by a new table linking values to their entity.

Example:



❖ Mapping Multi-valued Attributes (MVAs) (cont)

Analogy:



COMP3311 21T1 ♦ ER→Rel Mapping ♦ [12/17]

❖ Mapping Multi-valued Attributes (MVAs) (cont)

Example: the two entities

Person(12345, John, 12-feb-1990, [red, green, blue])
Person(54321, Jane, 25-dec-1990, [green, purple])

would be represented as

Person(12345, John, 12-feb-1990)
Person(54321, Jane, 25-dec-1990)
FavColour(12345, red)
FavColour(12345, green)
FavColour(12345, blue)
FavColour(54321, green)
FavColour(54321, purple)

❖ Mapping Subclasses

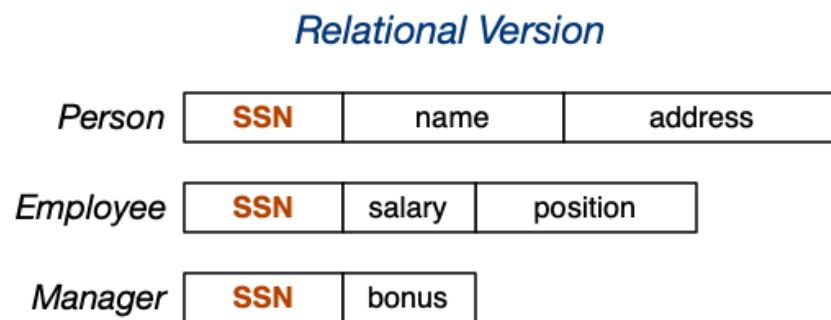
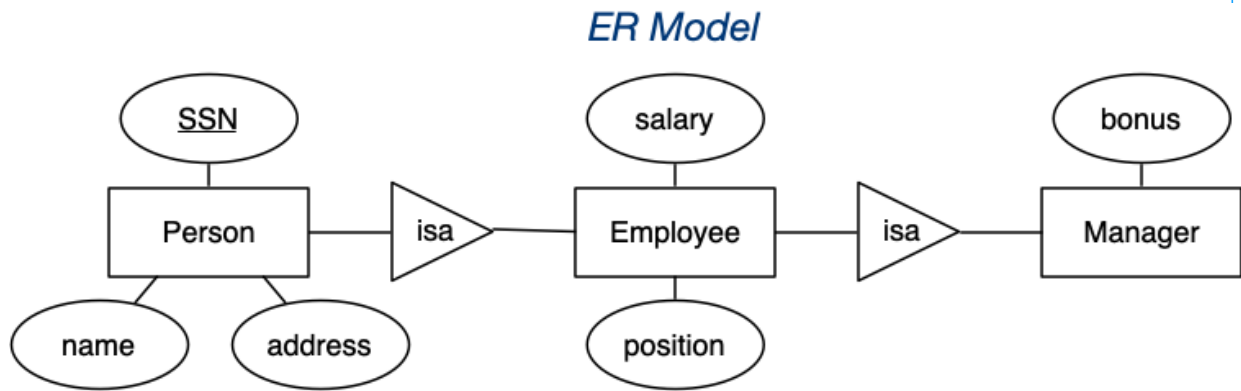
Three different approaches to mapping subclasses to tables:

- ER style
 - each entity becomes a separate table,
 - containing attributes of subclass + FK to superclass table
- object-oriented
 - each entity becomes a separate table,
 - inheriting all attributes from all superclasses
- single table with nulls
 - whole class hierarchy becomes one table,
 - containing all attributes of all subclasses (null, if unused)

Which mapping is best depends on how data is to be used.

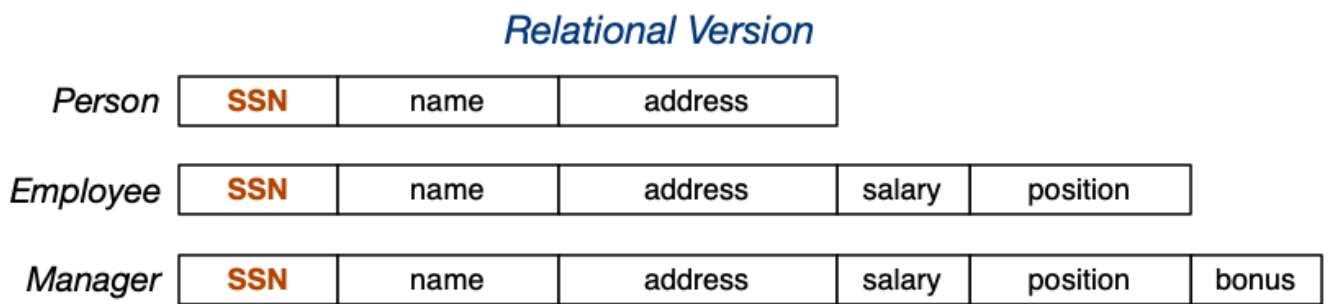
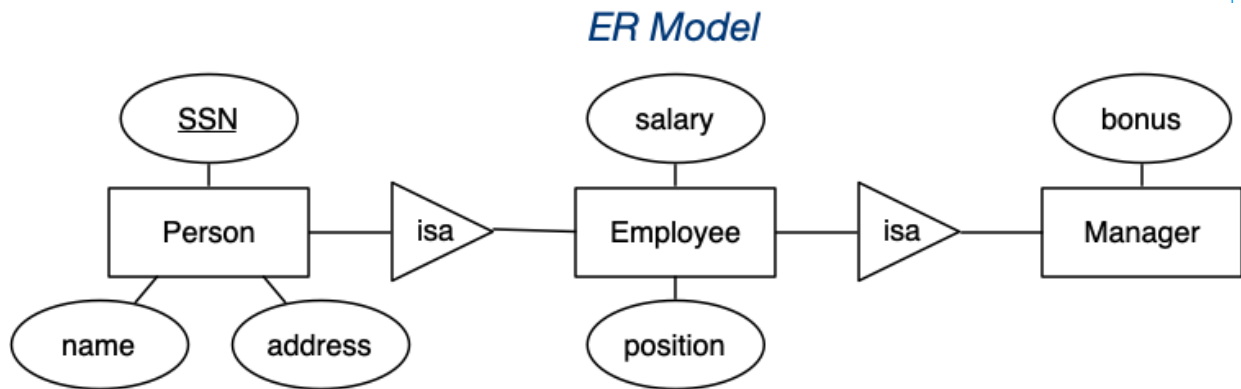
❖ Mapping Subclasses (cont)

Example of ER-style mapping:



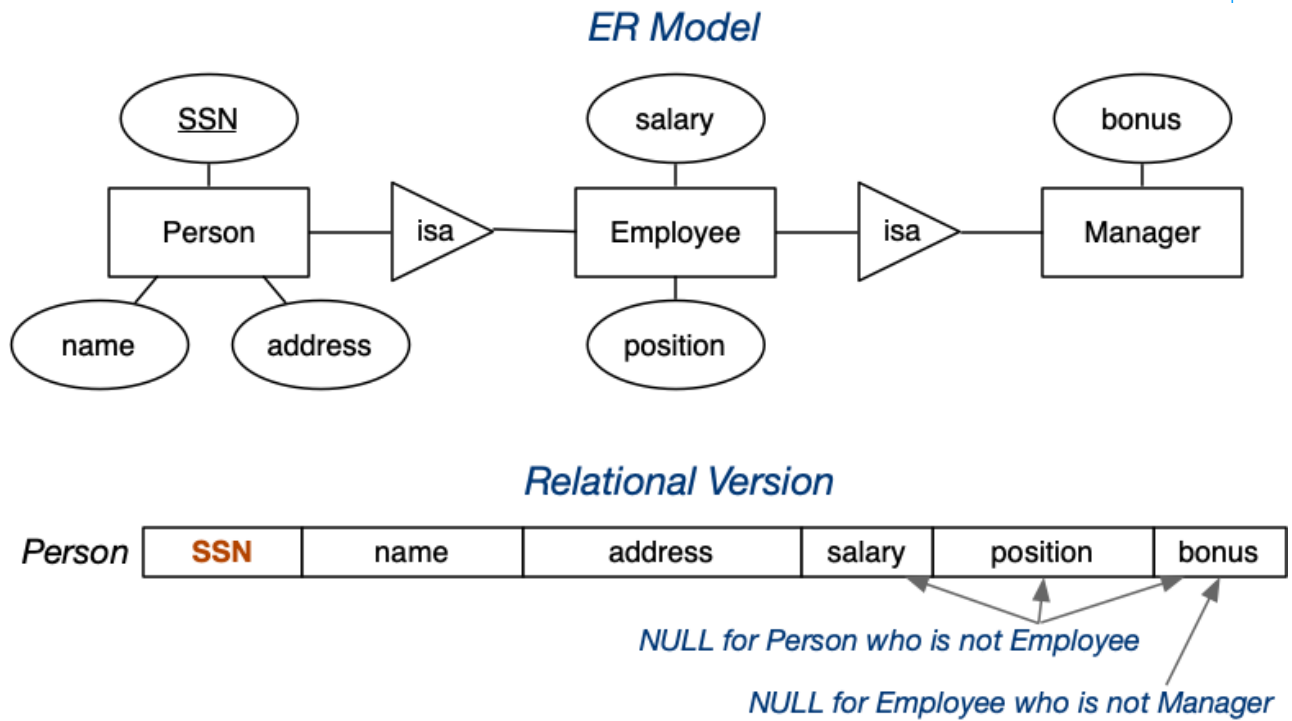
❖ Mapping Subclasses (cont)

Example of object-oriented mapping:



❖ Mapping Subclasses (cont)

Example of single-table-with-nulls mapping:



Produced: 10 Feb 2021