Week 3 Tutorial Notes

▼ Agenda

- Q2: ER → Relational for different totalities and cardinalities
- **Q3,4: different ways to model subclasses in relational model**
- **Q9: Discussion on serial (and the side effects)**
- Q10: ER → SQL baseline example
- **Q11: ER → SQL composite attributes**
- **Q14: ER → SQL, totality constraint lost in ER, char vs varchar vs text, order of create tables (foreign keys)**
- Q15: ER → Relational → SQL for 1:1 and 1:n combinations
- Q17: ER → Relational → SQL for mutually recursive pair of foreign keys

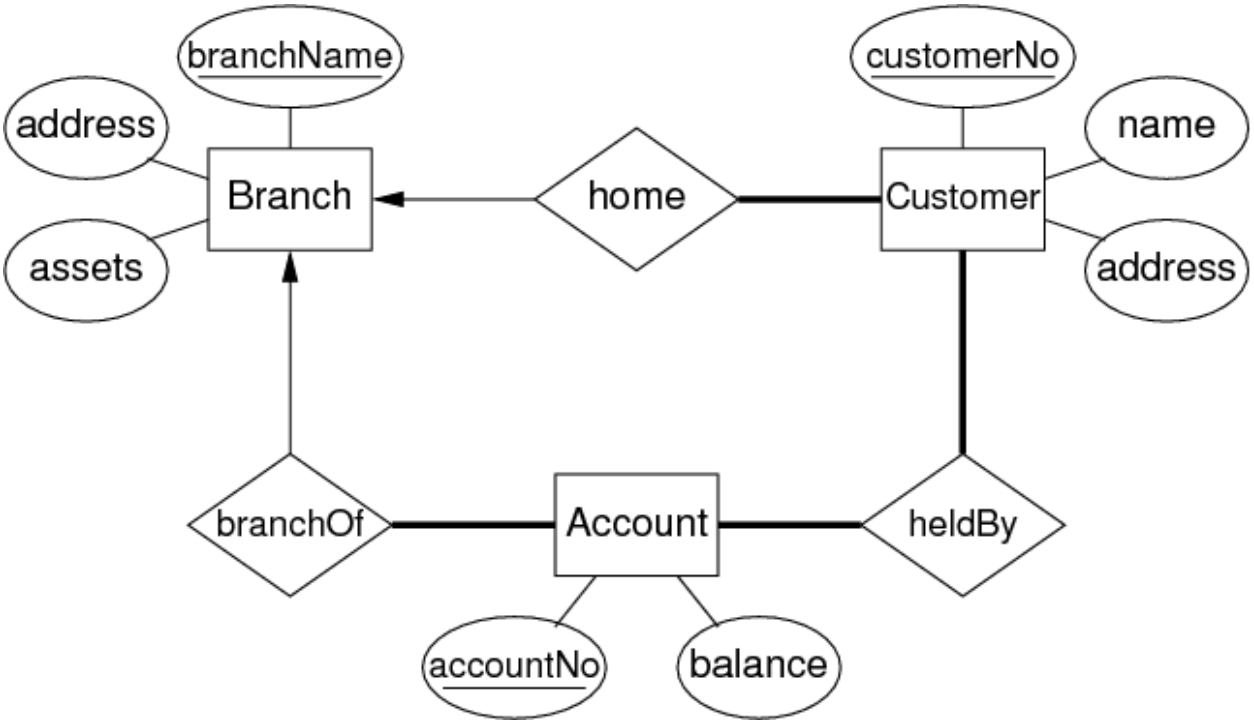
Pre-Tute

- Quiz due **Friday 11:59pm**
- Assignment 1 releasing this Wednesday (probably)
  - Deadline: **8:59pm Friday 4th July (Week 5)**

ER Diagrams (recap)

- Entities (rectangle)
- Attributes (oval)
  - Composite, multivalued, **key**
- Relationships
  - Cardinality (N:M vs N:1 vs 1:1)
  - Totality (participation)
- Other stuff:

- Weak entities
- Derived attributes

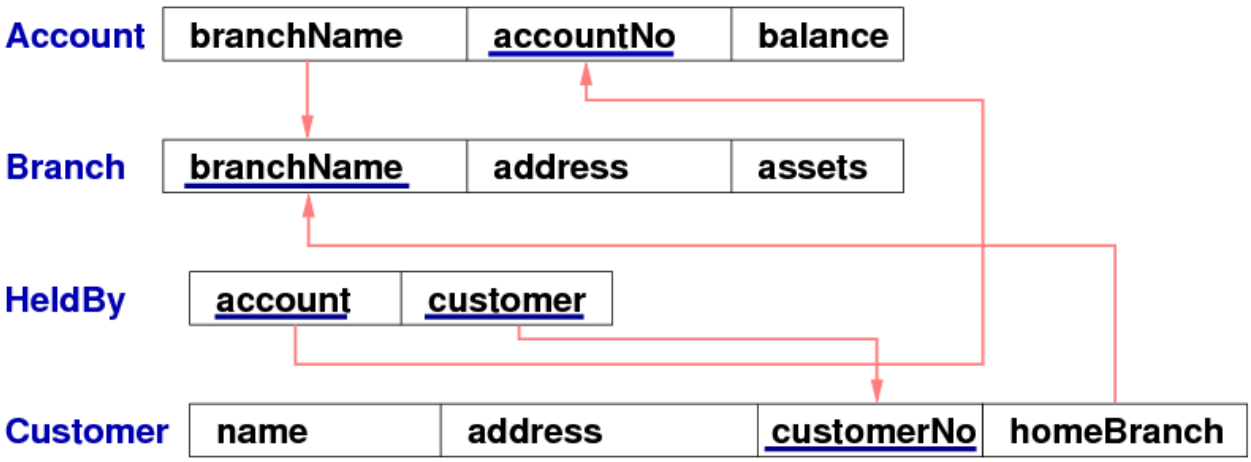


ER → Relational Models

The general process is:

1. Each entity becomes a relation
2. Attributes of entities become attributes of the corresponding relation (make sure to preserve primary keys)
3. Relationship between entities are represented by foreign ↔ primary key pairs

> 🛠️ Whether relationships become relations or not depends on the cardinality of the relationship (more on this later)

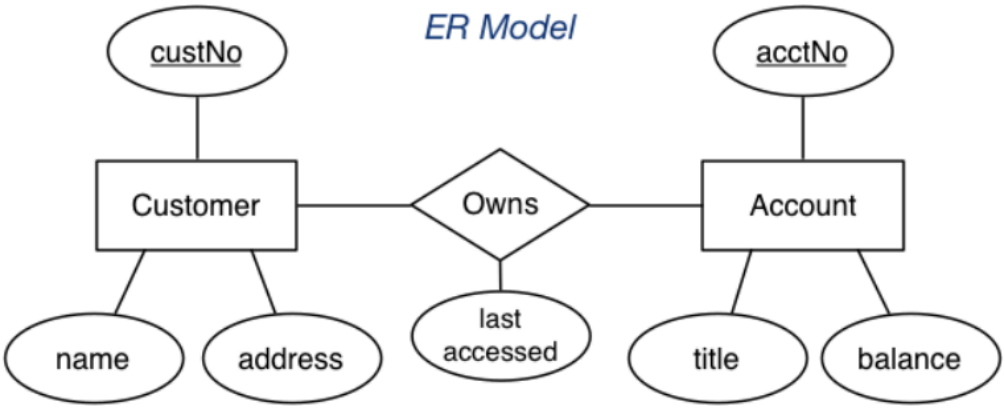


Scenarios of ER → Relational

▼ Baseline

Relationships

▼ N:M Relationships



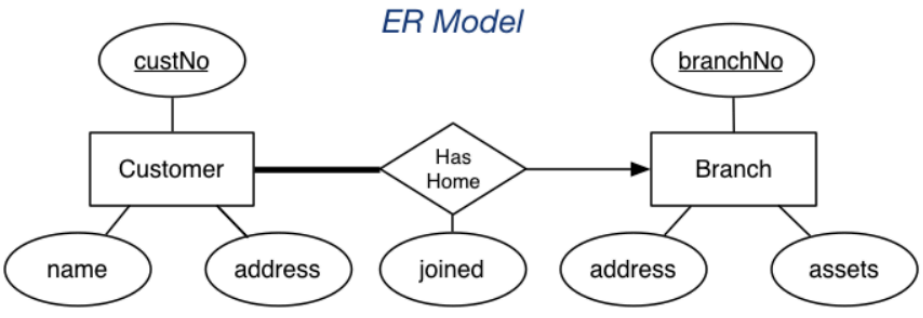
Relational Version

Customer	**custNo**	name	address

Account	**acctNo**	title	balance

Owns	**acctNo**	**custNo**	lastAccessed

▼ N:1 Relationships

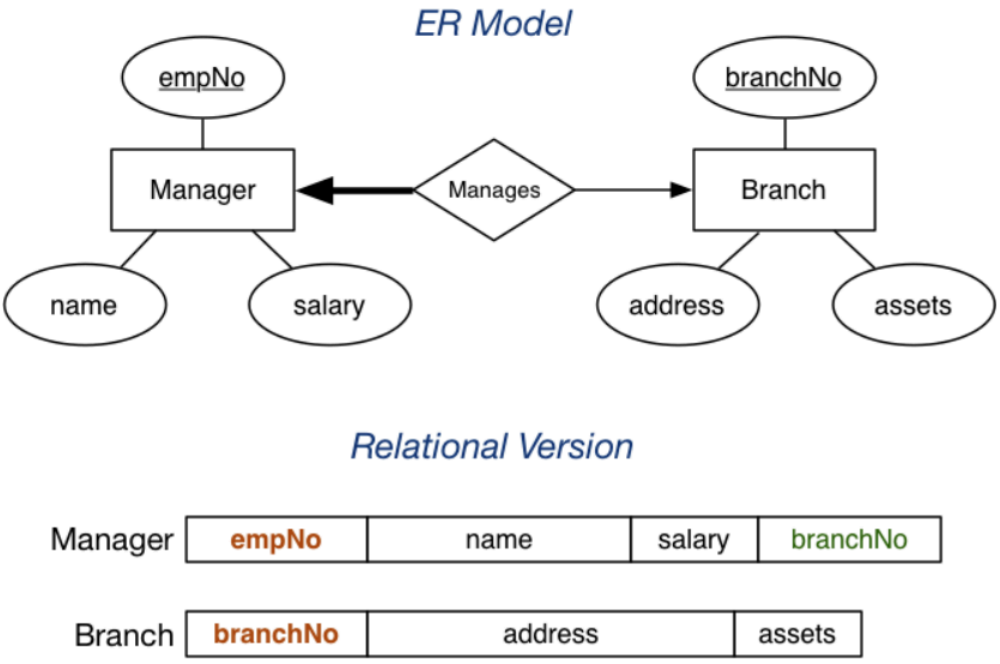


Relational Version

Customer	**custNo**	name	address	branchNo	joined

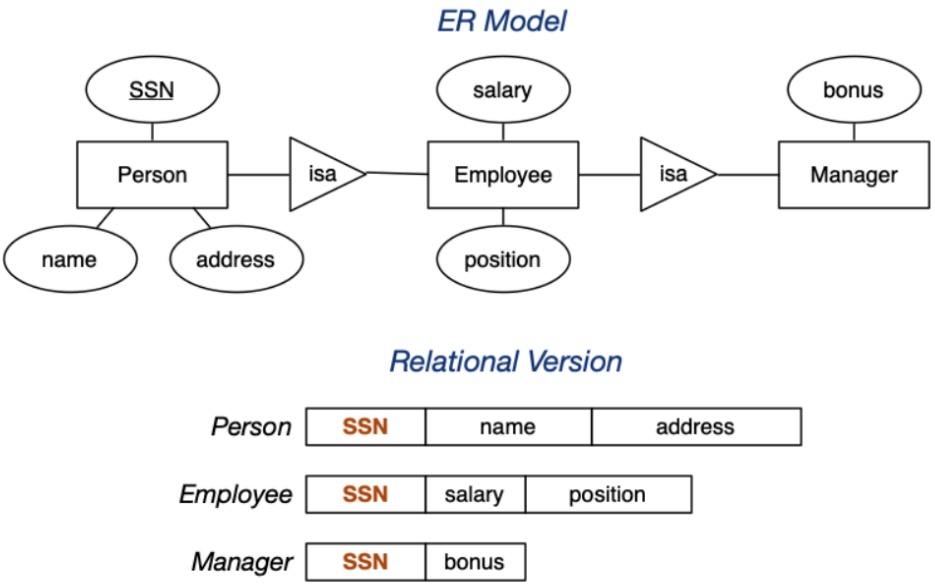
Branch	**branchNo**	address	assets

▼ 1:1 Relationships

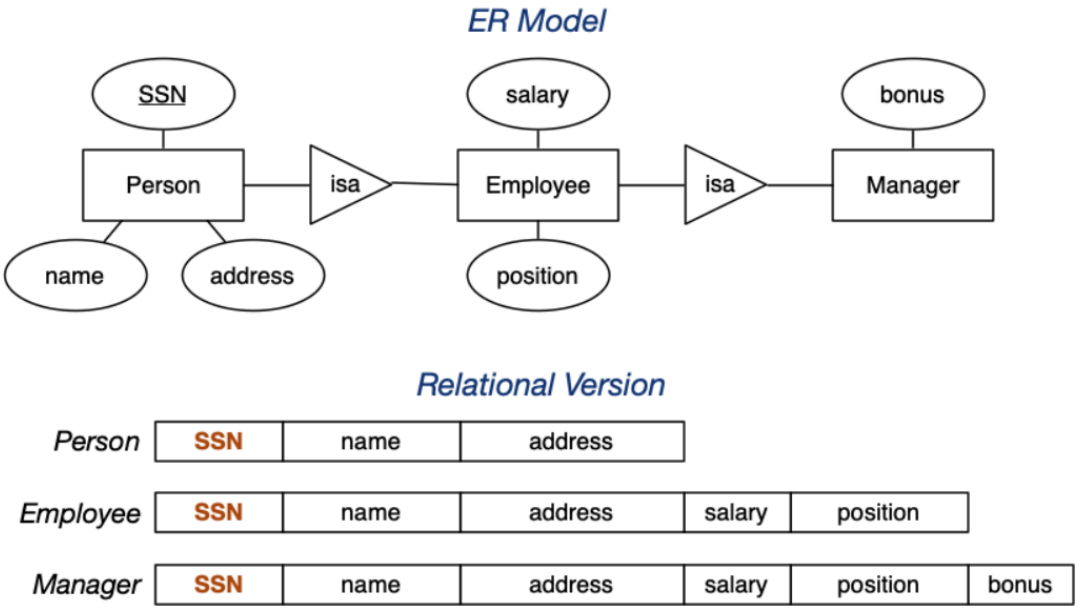


Inheritance / Class Hierarchies

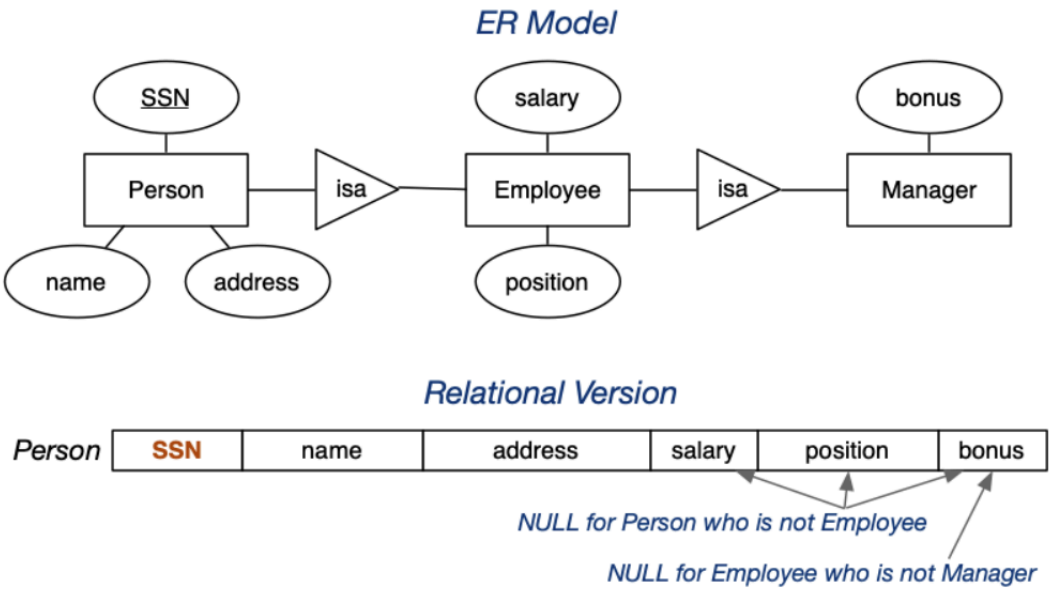
▼ ER-Style Mapping



▼ Object-Oriented Mapping

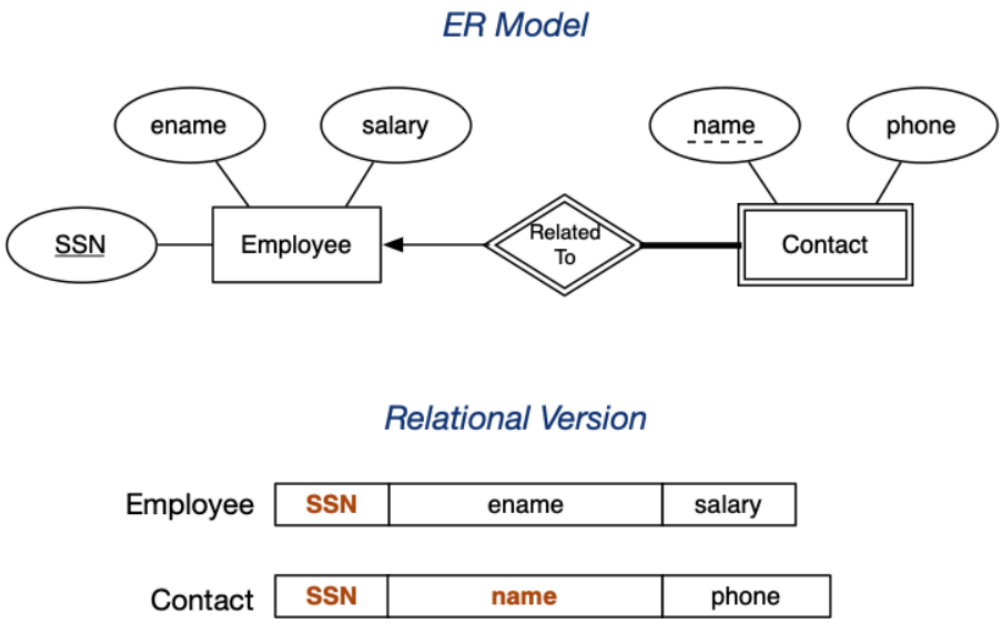


▼ Single-Table-With-Nulls Mapping

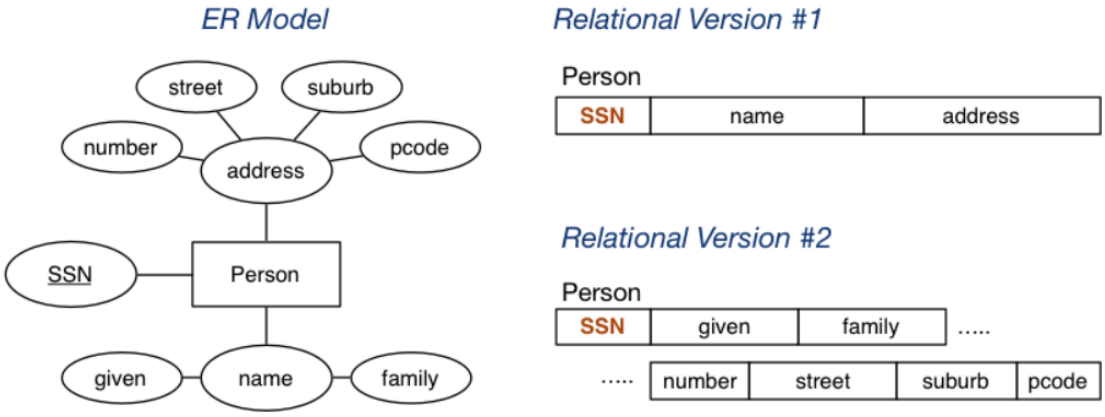


Special Entities and Attributes

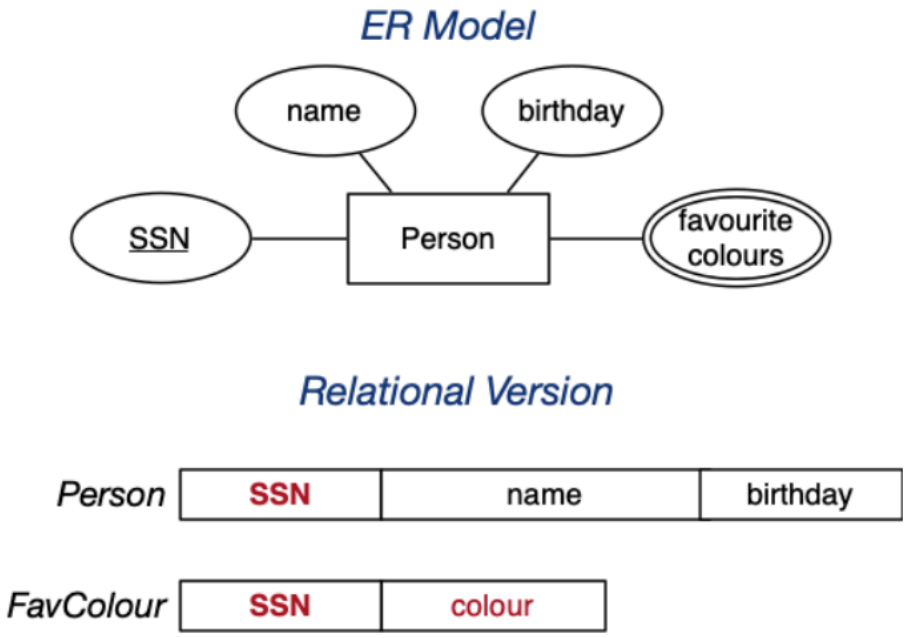
▼ Weak Entities



▼ Composite Attributes



▼ Multivalued Attributes



SQL Data Definition Language

SQL data definition language (DDL) is the formal way of describing the above relational schemas. Below is the structure of creating tables in SQL DDL:

```
create table TableName (
  attr1Name  type [constraints],
  attr2Name  type [constraints],
  attr3Name  type [constraints],
```

```
  ...
  primary key (attrxName),
  foreign key (attryName)
          references OtherTable (attrzName )
);
```

!! ER to SQL is not a perfect transformation. There are some constraints that can't be (automatically) enforced by SQL. The ones you need to be aware of at this point are:

Disjoint inheritance: a **Person** is a **Student** or **Teacher**, but can't be both (Example: Tutorial Q3 and 4)

Total Participation (N:M): every **Employee** must work at a **Department** (Example: Tutorial Q14)