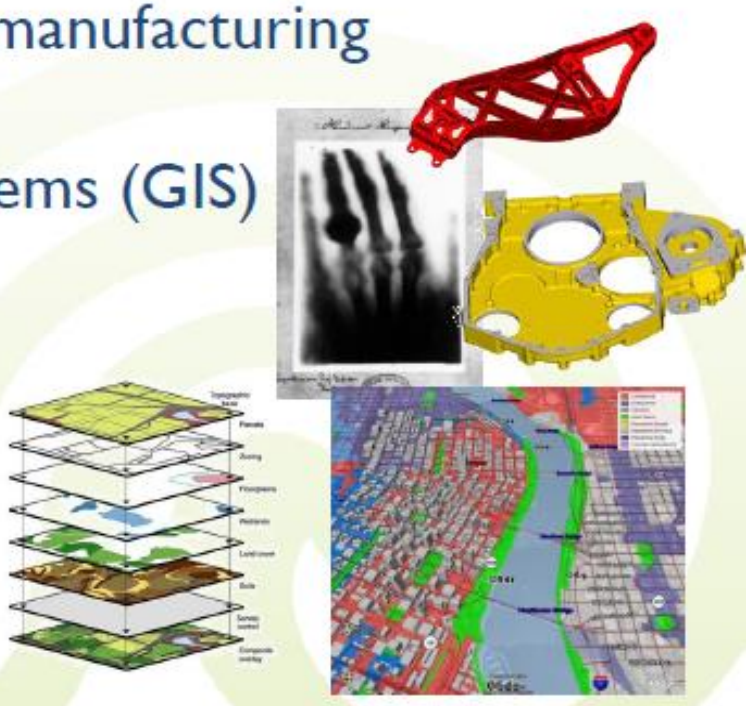
CS354: Database

Extended Data Modeling

- Traditional **ER modeling** proved to be very **successful** in classic "DB" domains:
  - Accounting
  - Banking
  - Airlines
  - Business and industry applications in general
  - …

Extended Data Modeling

- However, in the late 70s, popularity of DBs extended into fields with more **complicated data formats**
  - Computer-aided design and manufacturing (CAD/CAM)
  - Geographic information systems (GIS)
  - Medical information systems
  - …
- Expressiveness of ERD is **not sufficient** here

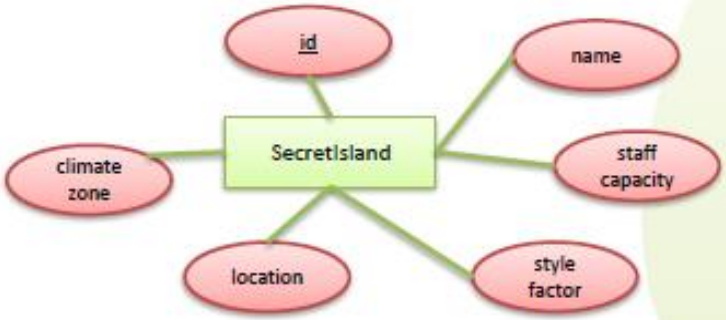


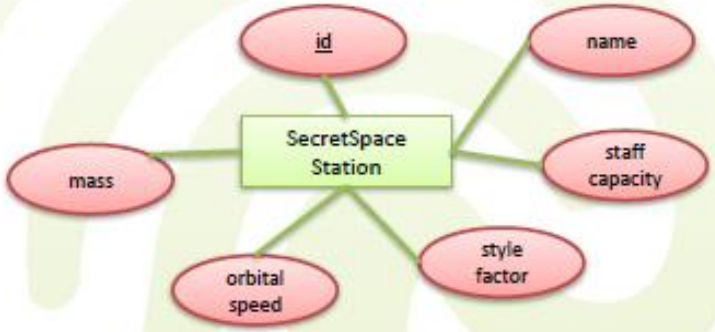
Extended Data Modeling

- Extended entity relationship (**EER**) models provide many additional **features** for more accurate **conceptual modeling**
  - Refinement of relationship types
    - Specialization and generalization
    - Class, subclass, and inheritance
  - Entity sets with existence dependencies
  - Extended modeling of domains and constraints
- Extended ER contains all features of "classic" ER

Subclasses / Superclasses

- Problem:
  - Model **secret lairs** to base highly secret research activities
  - **Secret island** and **secret space station** are special kinds of secret lairs, **share many attributes,** but still need some **unique attributes**



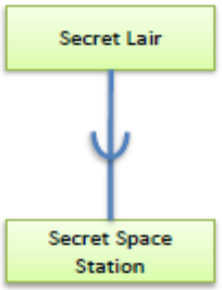


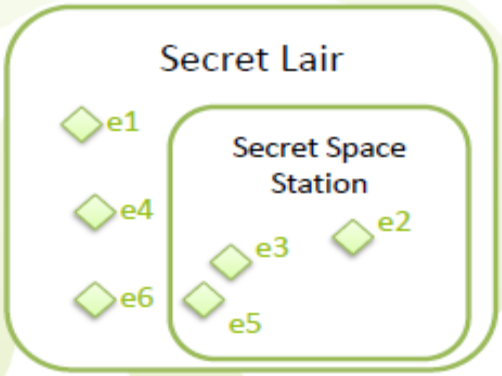
Subclasses / Superclasses

- Solution: **Subclasses** and **superclasses**
- A **subclass** entity type **inherits** all attributes and constraints from its **superclass** entity type
  - Subclasses may add additional attributes, constraints or relationship types
  - In EER, subclass relationship types are annotated with an open arc, which is opened to the super class (think of set inclusion)
  - Describes an "is-a" relationship

Subclasses / Superclasses

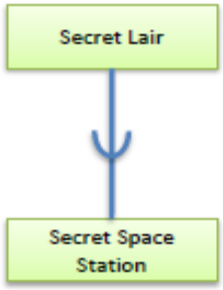
- **Subclass entity types** represent subsets of the entity set of the superclass' entity type
  - That is, an entity which is contained in the subclass is also contained in the superclass
  - In particular, no entity can **only** exist in a subclass set





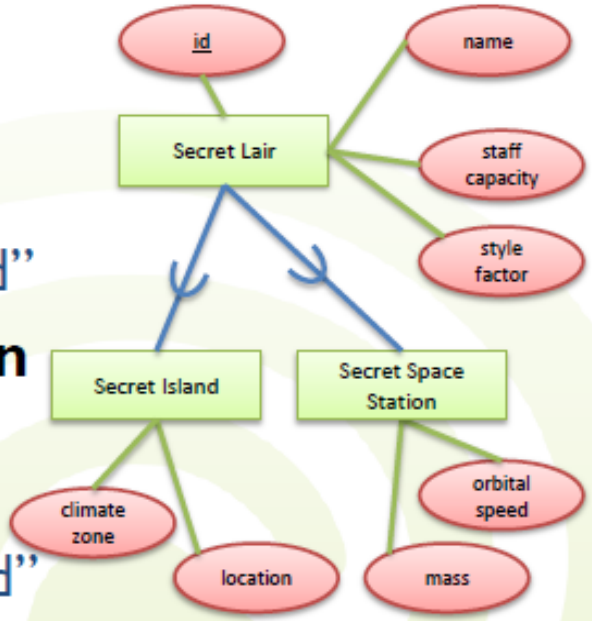
Subclasses / Superclasses

- Possible **implementation: Two distinct** database entries that represent the same instance
  - The same instance appears as a database entry in the superclass and subclass sets, and they are related to each other
  - 1:1 relationship on **entity level**
    - Linking two database entries of the **same entity** in a specialized **role**
  - Often, this solution is easier and more flexible to implement



Specialization / Generalization

- The process of defining a set of **subclasses** for a superclass is called **specialization**
  - Specialized entity types supplement additional attributes and relationships
  - "Secret lair" can be specialized into "secret space station" and "secret island"
- The inverse process is **generalization**
  - Generalization suppresses differences among specialized subclasses
  - "Secret space station" and "secret island" are generalized to "secret lair"



Specialization / Generalization

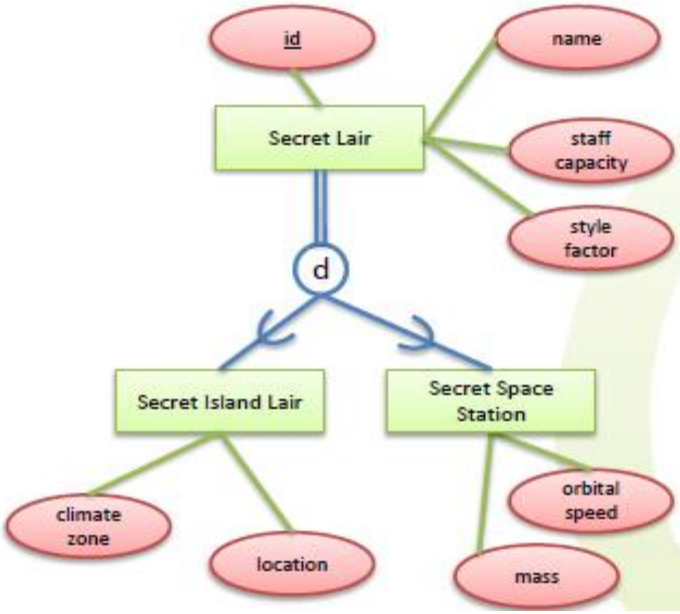
- Specialization and generalization usually result in the **same model**
  - However, the process of how to reach the model is different
  - **Specialization**: **top-down** conceptual refinement
    - Start with super classes, find suitable subclasses
  - **Generalization**: **bottom-up** conceptual synthesis
    - Model sub classes, find proper generalized super class

Constraints on Specialization

- Specializations can be constrained and modeled in further detail regarding two properties
  - **Exclusiveness** (indicated by a labeled circle)
    - **Disjoint:** Subclasses are mutually exclusive (default, label **d**)
    - **Overlapping:** Each entity may be contained in more than one subclass (label **o**)
  - **Completeness**
    - **Total:** No entity is member of the superclass without being member of a subclass (denoted by double line)
    - **Partial:** There are entities that are not contained in any subclass (default)

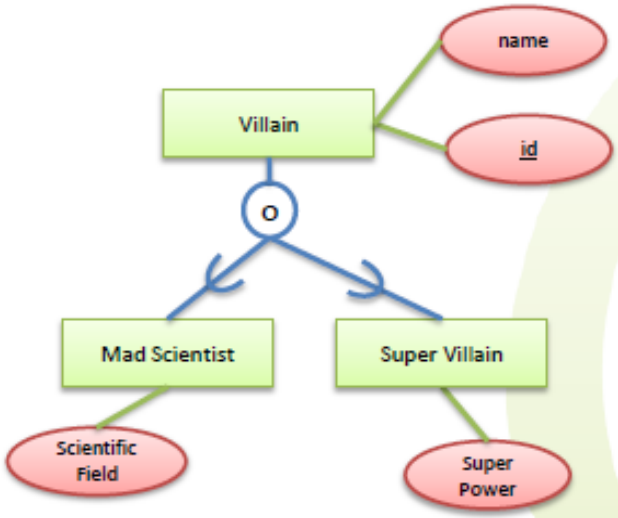
Constraints on Specialization

- Examples
  - **Disjoint** and **total:**
    "A secret lair may either be a secret island or a secret space station (but nothing else)."



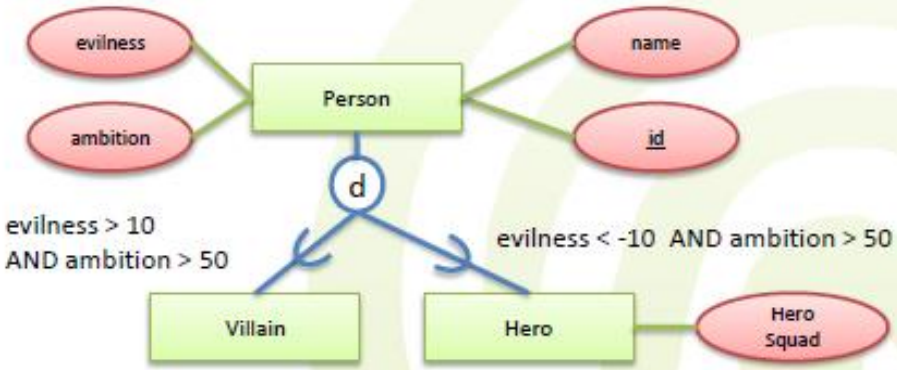
Constraints on Specialization

- Examples
  - **Overlapping** and **partial:**
    "A villain is a mad scientist, or a super villain, any combination of both, or something else (just a villain)."



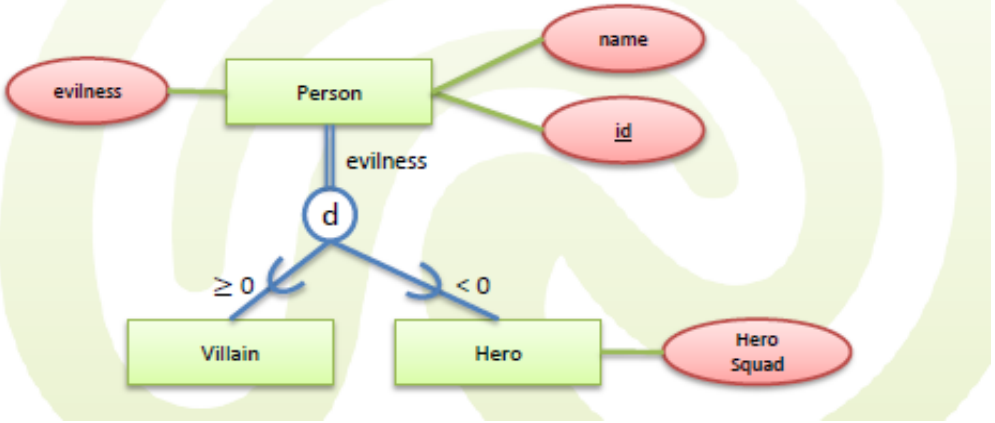
Constraints on Specialization

- Specializations may be **predicate-defined**
  - A subclass is predicate-defined if there is a predicate (condition) that implies an entity's membership
  - Condition is added to the specialization line
  - Predicate-defined specialization are not necessarily total



Constraints on Specialization

- Specializations may be **attribute-defined**
  - Attribute-defined is a special case of predicate-defined, where the membership in subclasses depends on a **single attribute value**
  - Attribute is added to line connecting circle and superclass, condition added to lines connecting circle and subclasses



Constraints on Specialization

- Consequences of specialization
  - **Deleting** an entity from the superclass also deletes it from all subclasses
  - **Inserting** an entity in a superclass automatically inserts it into all matching **predicate-defined** subclasses
  - In a **total** specialization, inserting one entity into a superclass implies that it has to be inserted into **at least one** subclass, too

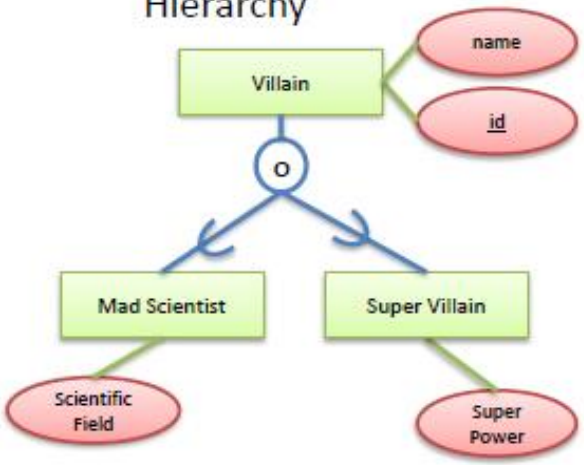
Hierarchies and Lattices

- A subclass may be further specialized
- If every subclass has just **one superclass,** the inheritance structure is a **specialization hierarchy**
- If there are subclasses having **more than one superclass** at the same time, the structure is a **specialization lattice**
  - Shared subclasses possible with multiple inheritance
- Subclasses recursively inherit all attributes and relationships of their superclasses up to the root

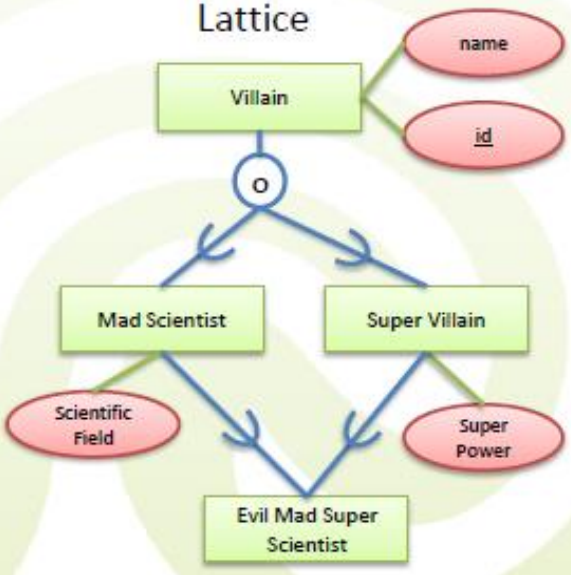
Hierarchies and Lattices

- An Evil Mad Super Scientist is a Mad Scientist as well as a Super Villain
  - Inherits attributes and relationships of both superclasses

Hierarchy



Lattice



Union Types

- In a superclass–subclass relationship, the **subclass inherits all** attributes and relationships of the superclass(es)
- However, sometimes it is beneficial that a subclass inherits from only **one superclass** (chosen from a **set** of potential distinct superclasses)
  - Every space station has an owner
  - A space station owner is either a space agency or a super villain

Union Types

- Solution: **Union types**
  - Denoted by a "u" in a circle
  - Space agency and Super villain are neither related, nor of the same type
  - An owner is **either** a space agency **or** a super villain