Strong entity

* Create a relation that includes all the simple attributes of that entity with the primary mey speicifed

1 to many relationship

* The child side (the many) receive the foreign key

1:1 binary relationship

* Depends on
* Mandatory participation on both sides
  + Ex:
  + College dean manages College
  + Primary key doesn’t matter so much since both can be linked
  + College dean primary key can be collegeID or deanID
* Mandatory participation on one side
  + The entity with mandatory participation receives the foreign key
  + Ex:
  + Professor manages a discipline
  + Professor does not need to manage a discipline, but a discipline needs a professor to manage it, so the discipline receives the foreign key
* Optional participation on both sides side
  + Foreign key on either side is fine

Many to many relationship

* Create a new relation as an associative entity
* Primary key
  + Both primary keys of its associate entities but may add other attributes
  + Ex:
  + Student enrols on course
  + Enrols is the associative entity
    - Both student and course relate to it

Recursive Relationship

* Same rules as binary relationships
* 1:1 or 1:\* are implemented using foreign keys
* \*:\* are implemented using an associative entity
  + Ex:
  + Person, PersonSponsoredBy,SponsoredBy,PersonSponsored
  + So there is a person being sponsored by another person
  + This means that there are 2 people (recursive)
  + It also means that there is now a:
    - Person that is sponsored
    - Person this is doing the sponsoring
    - An associative entity between the two: sponsored by

Diagram

Description automatically generated

Create a new relation with its own attributes

Use the primary keys of the participating entities that then act as foreign keys in the new relation

Each foreign key acts as a primary key of sorts to the course

Weak entity, find out what that shit is

Composite attributes are split into individual attributes

Derived attributes are not usually stored and are denoted by a /

Multi-valued attributes represented as a relation