* Discuss pros and cons for various single-relation or multiple-relation mappings for enhanced ER data model.
* Share any experience or conduct a research on "supertype and subtype" or "superclass and subclass" implementations.
* Discuss different ways to map the enhanced ER specialization lattice about ENGINEERING\_MANAGER shown on Figure 4.6 (7th Ed.) (or Figure 8.6, 6th Ed.) in the textbook and what is an effective way for relational mapping.
* Discuss any challenges for EER-to-relational mapping. Elaborate your reasons to choose single-relation or multiple-relation mappings.
* If you use UML Class diagram for conceptual database design, you will have the challenges for Class-to-relational mapping. Share your views about the challenges and how to resolve them accordingly.

There are many options for mapping single-relation and multiple-relation mappings from the enhanced ER model.

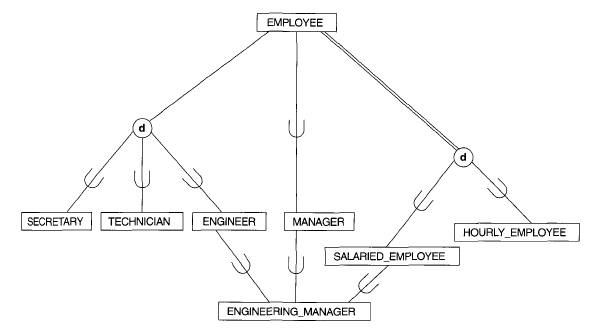
One option is to create the superclass relation and all subclass relations first with their attributes and then migrate the PK from the superclass relation into each subclass relation. This is a simple solution, but the PK from the superclass can be passed down multiple levels, which can cause confusion and take the database design out of 2NF.

Another option is to create a subclass relation and add all superclass attributes to the subclass relation. This will result in fewer relations, but will open the database up to the potential for duplicate data and will also take the database out of 2NF.

Another option is to create a relation which is a union of all the attributes from all subclass relations with a type attribute. This is for specializations with disjoint subclasses. This will result in far fewer relations, however will create many null values and you have to track different flags for each entity.

A final option is to create a single relation which combines all attributes from the superclass and subclass relations with a set of Boolean flags to indicate whether the tuple has the specified types. This is an extension of the previous option for overlapping subclasses, and has the added flexibility of maintaining overlapping attributes, but will again create many null values and force you to track flags for each entity.

These are all valid options to handle the mapping of the enhanced ER specialization lattice for engineering manager from figure 4.6 of the textbook (shown below):



There are two options of the bunch that stand out to me as a way to map the subclass/superclass relationships for the given figure. The first option discussed is typically a good option for relationships without too many levels, and mapping the primary keys from employee all the way down to engineering manager is only two levels of mapping. The fourth option discussed is also a good option due to the overlapping subclasses, since engineer, manager, and salaried employee all overlap into engineering manager. However, due to the increased complexity of the fourth option and all the null values introduced, the first discussed option is likely best for the given superclass/subclass relation. Option one is a single-relation mapping, thus each relation in the eventual class diagram will match each entity in the specialization lattice in figure 4.6.

Some of the challenges associated will be mapping each of the foreign keys from each of the relations when using the first discussed option. Engineering manager will have two foreign keys that are also primary keys, while engineer, manager, and salaried employee will each have one. Another potential challenge is in collecting all associated keys for the engineering manager. This will require a join between each of engineering manager, engineer, manager, salaried employee and employee. While collecting the attributes for engineering manager in this way may not be the most efficient, it feels much cleaner than option 4 where the data may have many more null values, flags to point which keys are valid, etc.

Using UML class diagrams for the conceptual database design phase may result in some challenges for mapping class to relational mappings. One way to get around this is to ensure that the primary key mappings are shown as attributes for the subclass. This is one way to show the mappings. Another way to get around this potential pitfall is to simply use an ER diagram in IE notation with labels on the relations to show the relationships. Again, some of the information is lost from the EERD, but this can provide a solution which will save as much data from the EERD as possible.