Please answer the following normalization review questions:

* Discuss insertion, deletion, and modification anomalies.
* Why are they considered bad? Illustrate with examples?  How we may prevent these anomalies in a database application?
* Why should NULLs in a relation be avoided as much as possible?  Discuss the problem of spurious tuples and how we may prevent it.
* When you perform Normalization process on a relation(s), please elaborate a major challenge(s) you may encounter.
* Why you need to perform normalization? Why you need to perform 1NF, 2NF, 3NF, BCNF in sequence?
* Define 1NF, 2NF, 3NF and BCNF.  How does BCNF different from 3NF? Why is it considered a stronger from of 3NF?
* Which normal form does not consider functional dependency between key and non-key attributes?

A successful database design will be one such that there is no insertion, deletion or modification anomalies present in the relations. An insertion anomaly can occur in the case where it is impossible to insert a row into the database, for example if two relations are required to have a reference to one another, it would be impossible to insert just one or the other. For example, if a team in a sport league database is required to have players, and players are required to be assigned to a team, then neither will be able to be inserted to the database. Deletion anomalies occur when deleting a single entity has unintended consequences to the rest of the database. Using a similar example from above, if a team is deleted and it went and deleted all the players as well, that would be a delete anomaly (since the players could find their way to other teams after the deleted team is disbanded). Update anomalies occur when changing one attribute could change attributes for other entities. For example, if a player changed teams, an update anomaly could be if the database updated the team value for each player on the team. These anomalies are considered bad because they corrupt the data, and we never want to corrupt the data in our database.

Null is a database value that should be avoided if possible. Some disadvantages of storing null include requiring extra data storage to track nulls, null may not work with aggregate operations like sum, avg, count, etc. and that null may have different interpretations. Spurious tuples can occur in join operations where database design does not enforce that relational schemas can be joined with equality conditions on attributes without creating the spurious tuples. Spurious tuples can cause issues in data aggregation and downstream applications.

One challenge I faced when performing normalization is in determining which entities are dependent on which primary keys. For example, in the homework it can be tricky to determine if the OwnerNo belongs to the PropertyNo key or both the PropertyNo and the ClientNo keys. This is important in performing 2NF, and can cause challenges down the line for the normalization. Normalization is important because it is a process to review and validate relations as well as to eliminate duplicate data. Each normalization must be performed in sequence, since the operations are not transitive. A mistake in 1NF can cause the solution from 3NF to be invalid, for example. The only normalization form which does not consider functional dependency between key and non-key attributes is 1NF, where you split data with multiple values into its own relations.