<u>Bookwork:</u>

14.1) **Discuss attribute semantics as an informal measure of goodness for a relation schema.**
Semantics of a relation refers to the result from interpretation of values in a given tuple. In other words, given a schema that could be easily explained in relation, means that the semantics of the attributes are easy to interpret; therefore this acts as an informal measure of goodness for a relation schema.

14.8) **Define first, second, and third normal forms when only primary keys are considered.**
<u>1NF:</u> The domain of an attribute must include only atomic values and the values of any attribute in a given tuple must be a single value from the domain of that attribute.
<u>2NF:</u> Based on the concept of full functional dependency. A dependency is considered “full functional”, when if after removing an attribute, the dependency no longer stands. Else, it is considered a “partial”
<u>3NF:</u> Based on the concept of transitive dependency. A dependency is considered “transitive”, when if there is a set of attributes, that neither a candidate key nor a subset of any key, both hold.

How do the general definitions of 2NF and 3NF, which consider all keys of a relation, differ from those that consider only primary keys?
The general definitions of 2NF and 3NF are different because general definition allocates candidate keys as well. An attribute that is part of any candidate key will be considered prime. Partial, full, and transitive dependencies are considered with respect to all candidate keys of a relation.

14.12) **Define Boyce-Codd normal form. How does it differ from 3NF? Why is it considered a stronger form of 3NF?**
Boyce-Codd normal form (BCNF) is a normal form containing a relation R if when a functional dependency X -> A holds in R, X is a superkey of R. No attribute depending upon a key is in a cycle. Every relation in BCNF is also 3NF, but a 3NF is not necessarily also a BCNF. BCNF is considered a stronger form of 3NF, since 3NF allows for functional dependencies that conform to clause b of 3NF but BCNF disallows them.

14.15) **Define fourth normal form. When is it violated? When is it typically applicable?**
The fourth normal form says that a relationship R has functional and multivalued dependencies such that for every nontrivial multivalued dependency X -> Y, X is a superkey for R. It is violated whenever there is a multivariate dependency that is not handled by a superkey. It can be applied whenever there are multivariate attributes on a table.

14.16) **Join Dependency** - If a table can be recreated by joining multiple tables and each of this table have a subset of the attributes of the table, then the table is in Join Dependency.
It is denoted by JD (R1, R2, ..., Rn), specified on relation schema R, specifies a constraint on the states r of R. The constraint states that every legal state r of R should have a non-additive join decomposition into R1, R2, ..., Rn; that is, for every such r we have
 * (R1(r), R2(r), ..., Rn(r)) = r

15.7) Normal forms alone are not sufficient because, a good schema design will also have properties of decompositions. The properties are:
a. Loss less joint property- Property through which we can identify all the records of the smaller relations when we again join it.
b. Dependency preservation property- It ensures that each functional dependency is represented in some individual relation resulting from decomposition.

With the normal forms alone, it is difficult to test the relation schemas independently of one another. That is why we must also the include the properties of decompositions, making it a good schema design.

16.1) **What is the difference between primary and secondary storage?**
Primary storage is the data that can be controlled by the CPU directly. It provides fast access to data at the expense of storage capacity. Stores less data than secondary storage, but is less expensive.
Secondary storage is mainly used for the storage of online databases and moderate amounts of permanent data. It can store more data than primary storage, but it is more expensive.

16.3)
Disk: used to store some amount of data depending on type.
Disk pack: many disks packed together to increase storage capacity
Track: concentric circles that store information.
Block: division of track into equal sized pieces that is set up by the operating system during disk formatting.
Cylinder: tracks that have the same diameter located on various surfaces.
Sector: A division of a track that is hard-coded on the disk surface and can’t be changed.
Intereblock gap: separate disks, which include specially coded control information written during disk initialization.
Read/write head: hardware mechanism that reads or writes a block in the disk, which is part of the disk drive.