Functional Dependencies and Normalization

**Exercise 1:**

Consider the following relation:

Stock(#prod, #dep, pname, quantity)

Determine the functional dependencies on Stock and express what they mean in English.

There are two functional dependencies in a Stock relation above:

1. #prod, #dep 🡪 quantity

Product ID and department ID determine the available quantity in stock

1. #prod 🡪 pname

Product ID determines the product name

**Exercise 2:**

Consider the following relation:

Plane(#plane, type, constructor, capacity, owner)

Determine the functional dependencies on Plane and express what they mean in English.

The functional dependency in a Plane relation above:

* #plane 🡪 type, constructor, capacity, owner

Plane IC fuctionaly determines type, constructor capacity, and owner

**Exercise 3:**

Compute the attribute closure of XZA with respect to the following set of dependencies:

F = {XZ => ZYB, YA -> GC, C -> W, B -> G, XZ -> G}

*XZ => ZYB*

*YA -> GC*

*C -> W*

*B -> G*

*XZ -> G*

Is the dependency XZA -> YB implied by F?

*Yes, the dependency XZA 🡪 YB is implied by F.*

*XZ --> ZYB => XZ --> YB; => XZ*A*--> YB*A; =>*XZ*A*--> YB*

Hint: Use the Armstrong axioms.

**Exercise 4:**

Is Stock (of Exercise 1) in 3NF? Why?

* *No, Stock is not in 3NF. It is not in 2nd Normal Form and it has Transitive Dependencies*

Is Plane (of Exercise 2) in 3NF? Why?

* *Yes, Plane is in 3NF, there are no Transitive Dependencies. It is in 2nd normal form.*

Is Stock (of Exercise 1) in BCNF? Why?

* *No, Stock is not in BCNF. It is not in 2nd Normal Form and it has Transitive Dependencies*

**Exercise 5:**

What are the properties of the BCNF decomposition algorithm?

* *Dependencies are not always preserved*
* *Lossless, information is not lost*

Apply the decomposition algorithm on Stock.

*After applying decomposition algorithms on Stock we get the following two tables which are both in 3NF and BCNF forms (every schema that is in BCNF is also in 3NF):*

1. *(#prod, #dep, quantity) where #prod and #dep are candidate keys*

*#prod, #dep 🡪quantity*

1. *(#prod, pname) where #prod is the candidate key*

*#prod🡪pname*