1.Find the functional dependencies for the below and normalize it till BCNF :

CustID	CustName	AccountManager	AccountManagerRoom	ContactName1	ContactName2
171	ABNAmro	Hans	12	Piet	Koos
190	RaboBank	Guus	15	Mona	Mieke

Ans - The given relation has the following functional dependencies -
CustID → CustName, AccountManager, AccountManagerRoom, ContactName1 ,ContactName2
AccountManager → AccountManagerRoom

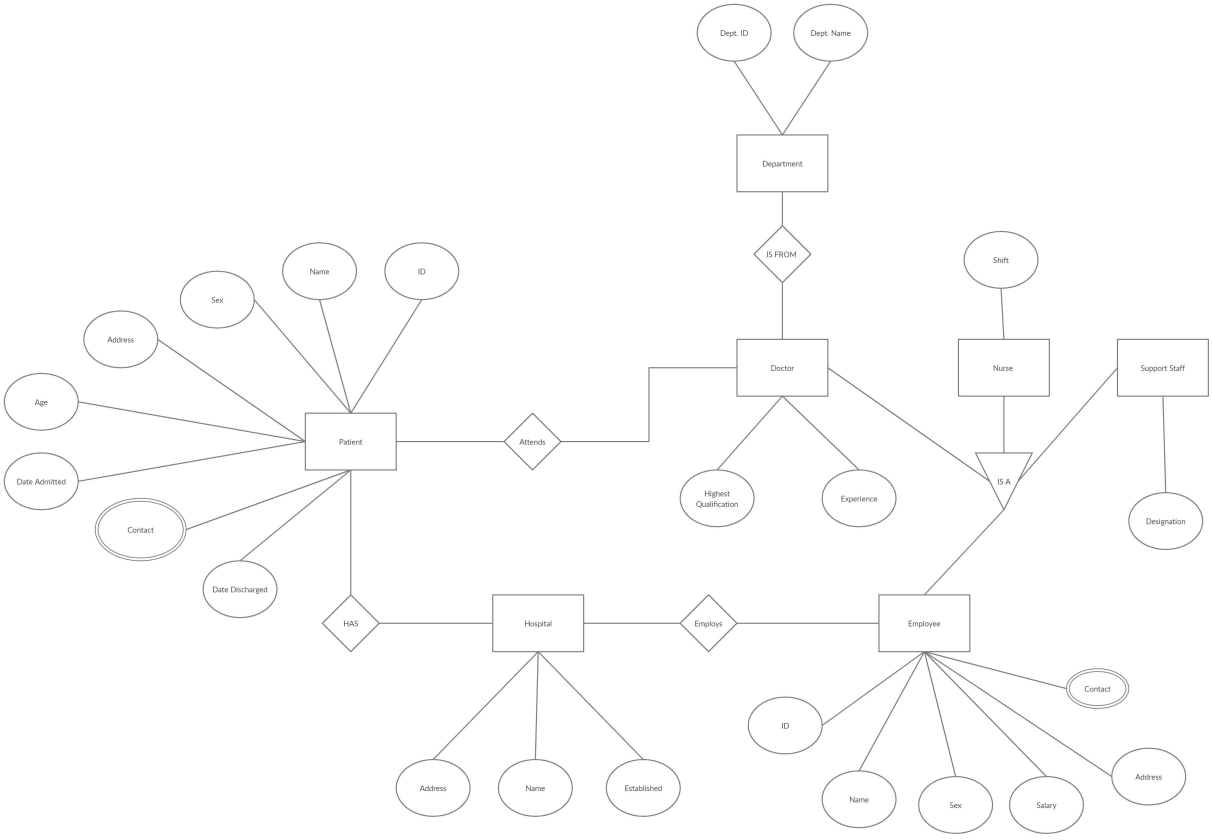
The given relation has CustID as the candidate key. There are no partial dependencies so, to convert it into BCNF we need to make sure that all the attributes are dependent only on candidate keys.

Here Account manger room has functional dependency on account manager which is not a candidate key so we need to split the given table in two tables.

Customer (CustID, CustName, AccountManager, ContactName 1 ,ContactName2)
Manager (Account Manager, Acoount manager room)

2. Draw an ER diagram for a hospital management system.

Ans -



3. Consider a relation Student (StudentID, ModuleID, ModuleName, StudentName, StudentAddress, TutorId, TutorName). Each student is given a StudentID and each module given a ModuleID. A student can register more modules and a module can be registered by more students. TutorID is the ID of the student's personal tutor, it is not related to the modules that the student is taking. Each student has only one tutor, but a tutor can have many tutees. Different students can have the same name. Different students can be living at the same address.
Find all the functional dependencies holding in this relation and normalize the table to 3NF.

Ans - The candidate keys here consists of StudentID and ModuleID as there closure gives us the complete student table.

The given relation is not 1NF as a student can have more than one modules (multi-valued) so **for 1NF** we need to have a different table for the module data of each student.
Student (StudentID, StudentName, StudentAddress, TutorID, TutorName)
Student_Module(StudentID, ModuleID, ModuleName)

Now here we can see that ModuleName has partial dependency on ModuleID in the Student_Module table. So **for 2NF** we need to split Student_Module table further.
Student (StudentID, StudentName, StudentAddress, TutorID, TutorName)
Student_Module (StudentID, ModuleID)
Modules (ModuleID, ModuleName)

Here these 3 are 2NF tables. For further normalization we can see that TutorName in the Student table depends on TutorId which is not a candidate key so **for 3NF** we can further divide the Student table...
Student (StudentID, StudentName, StudentAddress, TutorID)
Tutors (TutorID, TutorName)
Student_Module (StudentID, ModuleID)
Modules (ModuleID, ModuleName)