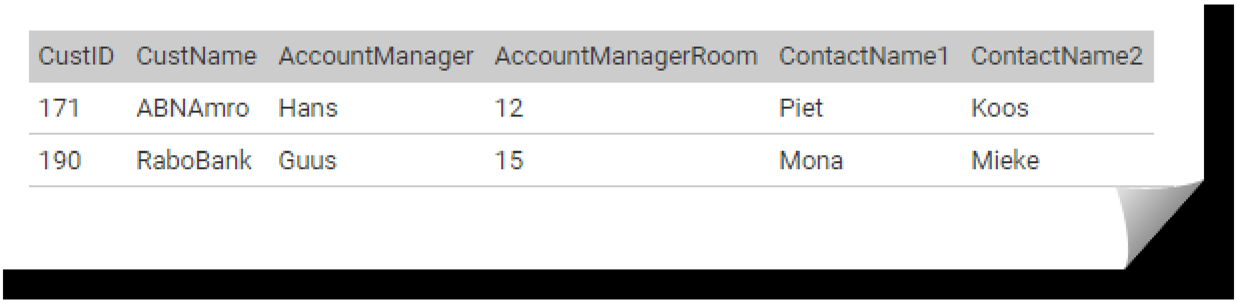
RDBMS Concepts – Morning Session ASSIGNMENT:

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

Using contextual knowledge and the values for the attributes:

- Since each customer must have unique ID, unique AccountManager and unique Contacts
  CustID → CustName, AccountManager, ContactName1, ContactName2 (1)

- Since each AccountManager must have a unique AccountManagerRoom
  AccountManager → AccountManagerRoom (2)

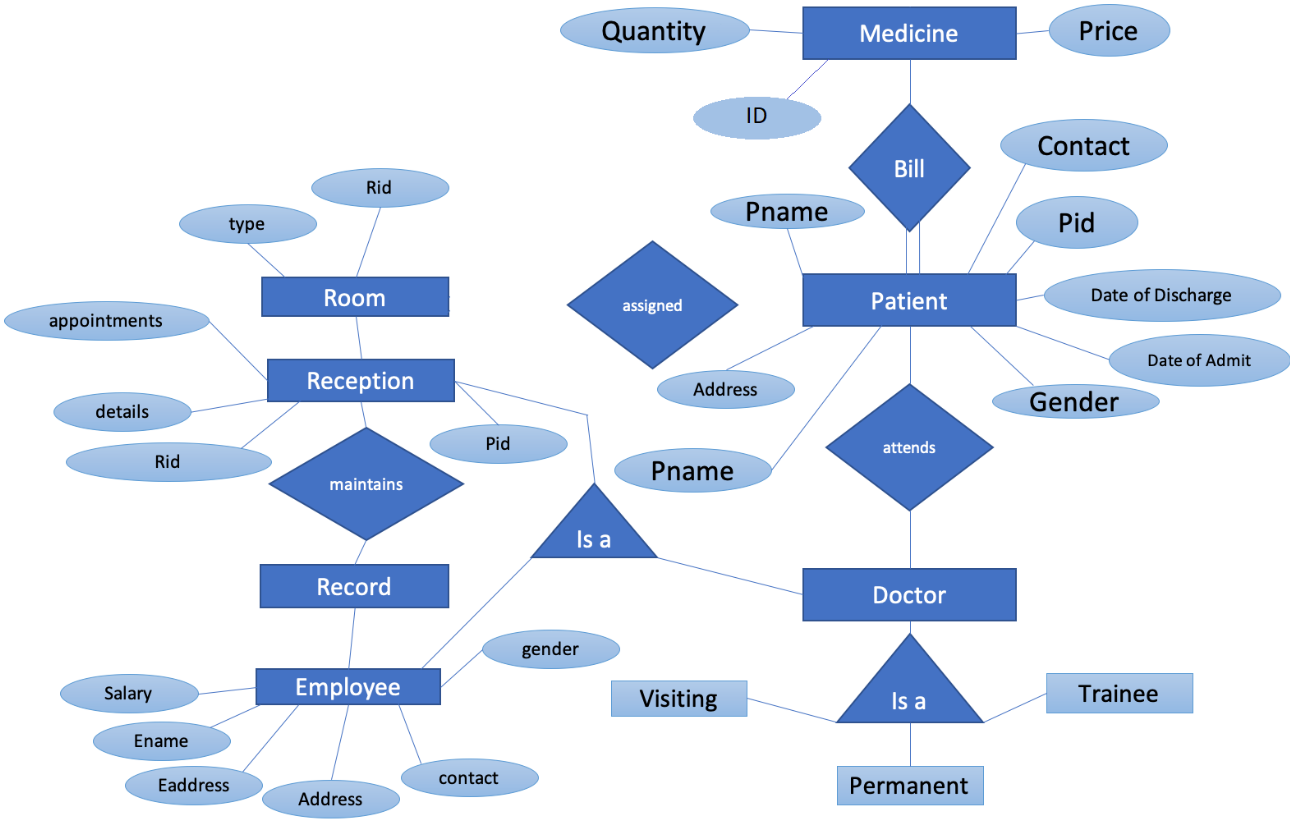
  Since the closure of (CustID) gives the complete *Customer* relation, it can uniquely identify all the tuples in the relation and thus, it is the candidate key.

Customer is already in 1NF as it only contains atomic values of its attributes and in 2NF as there are no partial dependencies on CustID. However, it is not in 3NF as transitive dependency due to (2) exists.

Thus, it needs to be decomposed as follows:
Customer (CustID, CustName, ContactName1, ContactName2, AccountManager)
AccManager (AccountManager, AccountManagerRoom)

Since, the LHS in all the FDs (1) and (2), are only superkeys, the relations are BCNF.

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



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.

Functional Dependencies:

- Given that each module has a unique ID
  ModuleID ⟶ ModuleName (1)

- Given that each student has a unique ID and a unique tutor
  StudentID ⟶ StudentName, StudentAddress, TutorId, TutorName (2)

- Given that each tutor has a unique ID
  TutorID ⟶ TutorName (3)

We can see that (StudentID, ModuleID) can uniquely identify all the tuples in the relation since its closure gives the complete Student relation. It is the candidate key.

Student is already in 1NF since it has only atomic values as its attributes. However, it is not in 2NF as there are many partial dependencies, (1) and (2), on StudentID and ModuleID.

Thus, it needs to be decomposed as follows:

Student (StudentID, StudentName, StudentAddress, TutorId, TutorName).

Module (ModuleID, ModuleName)

Since there exists a many-to-many relationship between Student and Module, any foreign key in either relation cannot establish the relationship being in 2NF. Thus, a new relation is defined as follows:

Stud_Mod (StudentID, ModuleID)

Now, the new Student relation is not in 3NF as it has transitive dependency (StudentID → TutorID and TutorID → TutorName). Decomposing using this dependency, the final relations in 3NF are as follows:

Student (StudentID, StudentName, StudentAddress, TutorID)

Tutor (TutorID, TutorName)

Module (ModuleID, ModuleName)

Stud_Mod (StudentID, ModuleID)