Homework Assignment 8

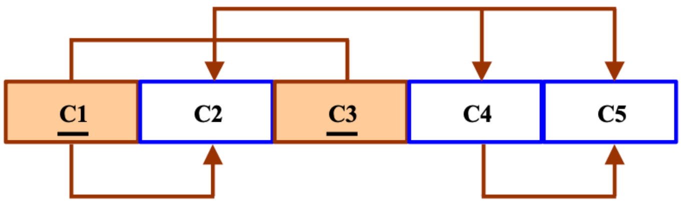
Question 1:

Normalization

Question 1

Given the dependency diagram shown in Figure below, Answer Items 6a–6c

Dependency Diagram for Question below



Identify and discuss each of the indicated dependencies

a. **Identify and discuss each of the indicated dependencies.**

b. Create a database whose tables are at least in 2NF, showing the dependency diagrams for each table

c. **Create a database whose tables are at least in 3NF, showing the dependency diagrams for each table.**

Solution:

Part a: Identify and Discuss Each of the Indicated Dependencies

From the dependency diagram:

1. **Primary Key**: The underlined attributes (C1 and C2) together seem to be the composite primary key of this table.
2. **Functional Dependencies**:

    - C1, C2 → C3: This indicates that the combination of C1 and C2 uniquely determines C3.
    - C3 → C4: This shows a transitive dependency where C3 determines C4.
    - C4 → C5: Another transitive dependency where C4 determines C5.

In summary:

- **Partial Dependency**: C1, C2 → C3 is fully dependent on the composite key, so there are no partial dependencies here.
- **Transitive Dependencies**: The dependencies C3 → C4 and C4 → C5 are transitive since they are not directly dependent on the primary key (C1, C2), but rather on C3 and C4.

Part b: Create a Database Whose Tables are at Least in 2NF

To achieve **Second Normal Form (2NF)**:

1. Eliminate partial dependencies (if any) by ensuring that non-key attributes depend on the whole primary key.

Since C3 is fully dependent on C1, C2 (the primary key), the table is already in 2NF. However, we still have transitive dependencies (C3 → C4 and C4 → C5), which we will address in the next step.

Table Structure in 2NF

In 2NF, the table structure remains as follows:

C1 (PK)	**C2 (PK)**	**C3**	**C4**	**C5**

Part c: Create a Database Whose Tables are at Least in 3NF

To achieve **Third Normal Form (3NF)**:

1. Eliminate transitive dependencies by creating new tables where attributes are only dependent on primary keys.

Step 1: Separate C3 → C4

- Create a new table for C3 and C4 so that C4 depends directly on C3.

Step 2: Separate C4 → C5

- Create another table for C4 and C5 so that C5 depends directly on C4.

Final Table Structure in 3NF

1. **Original Table** (revised for 3NF):
   - This table now only contains attributes directly dependent on the primary key.

C1 (PK)	C2 (PK)	C3 (FK)

2. **Table for C3 → C4**:
   - This table holds the dependency between C3 and C4.

C3 (PK)	C4

3. **Table for C4 → C5**:
   - This table holds the dependency between C4 and C5.

C4 (PK)	C5

Dependency Diagrams for Each Table in 3NF

1. **Diagram for Original Table**:

Copy code

C1, C2 → C3

   - No transitive dependencies are present in this table.

2. **Diagram for Table with C3 and C4**:

Copy code

C3 → C4

   - C4 is fully dependent on C3.

3. **Diagram for Table with C4 and C5**:

Copy code

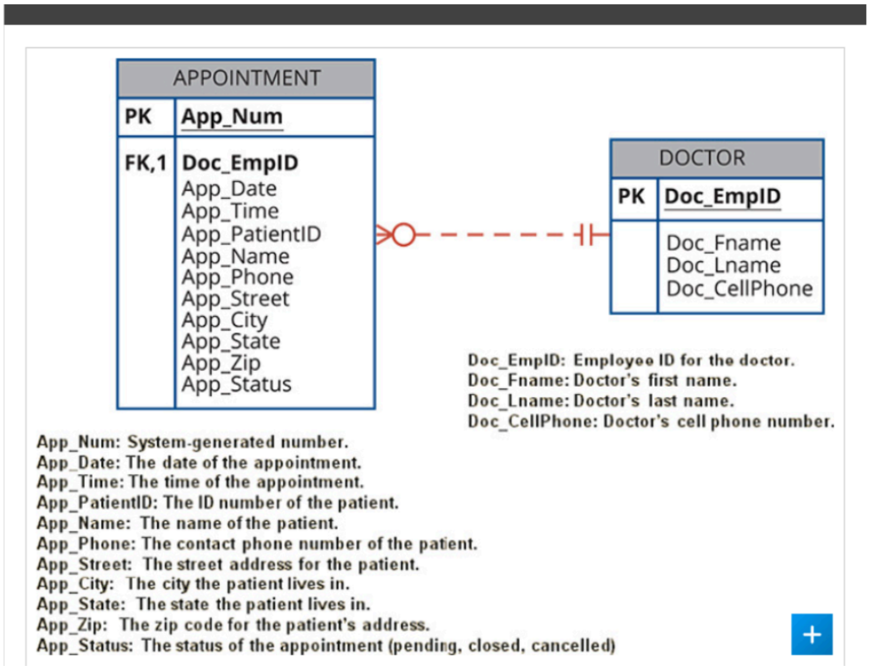
C4 → C5

- C5 is fully dependent on C4.

This final structure in 3NF ensures that each non-key attribute is only dependent on the primary key, with no transitive dependencies, thereby achieving a well-normalized database design.

Question 2:

Using the descriptions of the attributes given in the figure, convert the ERD shown in Figure below into a dependency diagram that is in 3NF.



Doc_EmpID: Employee ID for the doctor.
Doc_Fname: Doctor's first name.
Doc_Lname: Doctor's last name.
Doc_CellPhone: Doctor's cell phone number.

App_Num: System-generated number.
App_Date: The date of the appointment.
App_Time: The time of the appointment.
App_PatientID: The ID number of the patient.
App_Name: The name of the patient.
App_Phone: The contact phone number of the patient.
App_Street: The street address for the patient.
App_City: The city the patient lives in.
App_State: The state the patient lives in.
App_Zip: The zip code for the patient's address.
App_Status: The status of the appointment (pending, closed, cancelled)

Detail:

Appointment ERD for problem. Here, one and only one doctor gives 0 to many appointments. The primary key for appointment is App underscore number and the foreign key1 is doctor underscore emp ID. An explanation for each of the attributes for appointment are as follows. App underscore n u m is a system generated number. App underscore date is the date of appointment. App underscore time is the time of the appointment. App underscore Patient ID is the ID number of the patient. App underscore Street is the street address for the patient. App underscore city is the city the patient lives in. App underscore state is the state the patient lives in. App underscore Zip is the zip code for the patient's address. App Status is the status of the appointment, pending, closed or cancelled. The attributes for doctor are as follows. Doc underscore emp ID is the employee I D for the doctor. Doc underscore F name is the doctor's first name. Doc underscore L name is the Doctor's last name. Doc underscore cell phone is doctor's cell phone number.

Solution:

Step 1: Analyzing the APPOINTMENT and DOCTOR Tables

1. APPOINTMENT Table

- **Primary Key**: App_Num
- **Foreign Key**: Doc_EmpID (refers to DOCTOR table)
- **Attributes**:
  - App_Date, App_Time: Appointment details
  - App_PatientID, App_Name, App_Phone, App_Street, App_City, App_State, App_Zip: Patient details
  - App_Status: Appointment status (e.g., pending, closed, cancelled)

2. DOCTOR Table

- **Primary Key**: Doc_EmpID
- **Attributes**:
  - Doc_Fname: Doctor's first name
  - Doc_Lname: Doctor's last name
  - Doc_CellPhone: Doctor's cell phone number

Step 2: Identifying Dependencies

1. **Functional Dependencies in APPOINTMENT Table**:
   - App_Num → All other attributes in APPOINTMENT (since App_Num is the primary key)
   - Doc_EmpID → Doc_Fname, Doc_Lname, Doc_CellPhone (transitive dependency via foreign key Doc_EmpID from DOCTOR)
2. **Functional Dependencies in DOCTOR Table**:
   - Doc_EmpID → Doc_Fname, Doc_Lname, Doc_CellPhone (since Doc_EmpID is the primary key)

Step 3: Breaking Down into 3NF

To convert the schema into 3NF, we need to ensure that each table contains attributes that are only dependent on its primary key, with no transitive dependencies.

1. **Primary Table 1: APPOINTMENT**
   - This table will contain only the attributes specific to each appointment.
   - **Attributes**:
     - App_Num (Primary Key)
     - App_Date, App_Time
     - App_PatientID, App_Name, App_Phone, App_Street, App_City, App_State, App_Zip
     - App_Status
     - Doc_EmpID (Foreign Key referencing DOCTOR)
   - **Dependency Diagram**:
     - App_Num → App_Date, App_Time, App_PatientID, App_Name, App_Phone, App_Street, App_City, App_State, App_Zip, App_Status, Doc_EmpID
2. **Primary Table 2: DOCTOR**
   - This table will contain only the doctor-specific information.
   - **Attributes**:
     - Doc_EmpID (Primary Key)
     - Doc_Fname, Doc_Lname, Doc_CellPhone
   - **Dependency Diagram**:
     - Doc_EmpID → Doc_Fname, Doc_Lname, Doc_CellPhone

3NF Dependency Diagram

1. **APPOINTMENT Table**:

Copy code

App_Num → App_Date, App_Time, App_PatientID, App_Name, App_Phone, App_Street, App_City, App_State, App_Zip, App_Status, Doc_EmpID

- This table is now in 3NF because all non-key attributes are directly dependent on the primary key App_Num, and there are no transitive dependencies.

2. **DOCTOR Table**:

Copy code

Doc_EmpID → Doc_Fname, Doc_Lname, Doc_CellPhone

- This table is in 3NF because each non-key attribute is directly dependent on the primary key Doc_EmpID, with no transitive dependencies.

The `APPOINTMENT` table and `DOCTOR` table are now normalized to 3NF. The `APPOINTMENT` table contains only data related to individual appointments, including a foreign key `Doc_EmpID` that links to the `DOCTOR` table. The `DOCTOR` table contains information specific to each doctor. This design ensures that there are no partial or transitive dependencies, achieving 3NF for both tables.