Lab06

**Problem 14.15**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| staffNo | dentistName | patNo | patName | appointmentDate | time | surgeryNo |
| S1011 | Tony | P100 | White | 12-Sep-13 | 10:00 | S15 |
| S1011 | Tony | P105 | Bell | 12-Sep-13 | 12:00 | S15 |
| S1024 | Helen | P108 | Mackay | 14-Sep-13 | 10:00 | S10 |

a) With table like this, we have some problem when we insert, delete and update.

\* Insertion anomaly:

- We cannot insert a new dentist if we don’t have appointment with this dentist. For example, we have new dentist S1025 – John, we cannot add new record like this into the table

S1025 | John | ? | ? | ? | ? | ?

We don’t have information of patNo, patName, appointmentDate, time, and surgeryNo

\* Deletion anomaly

- If we want to delete an appointment, it may result in loss of dentist or patient information.

- For example, we delete both first two lines, we will lose the information of Tony. Although Tony is still working, he just has no appointments scheduled for now.

\* Update anomaly

- If we want to update name of Tony to Tony Nguyen, you must update all record have S1011 in staffNo to Tony Nguyen. If you don’t follow like that, we will have 2 different dentist names but have same staffNo. It will result in wrong data

b)

There is several reasonable assumptions for defining functional dependencies:

\* Each staff has a name: **staffNo -> dentistName**

\* Each patient has a name: **patNo-> patName**

\* An appointment is identified by date, time and clinic: **apppointmentDate, time, surgeryNo -> staffNo, patNo**

\* A patient can have many appointments: patNo does not identify appointmentDate or time

\* A dentist can work in different clinic: dentistName does not identify surgeryNo

\* Each appointment related to a dentist: **appointmentDate, time, surgeryNo -> dentistName**

\* Each appointment related to a patient: **appointmentDate, time, surgeryNo -> patName**

c)

- 1NF done because each cell contains atomic value, that is, it cannot contain multiple values in a cell. Beside that, no repeating groups, each attribute must represent a single, unique data type.

- We have several Functional dependencies identified from part a.

\* staffNo -> dentistName

\* patNo -> patName

\* appointmentDate, time, surgeryNo-> staffNo, patNo

\* appointmentDate, time, surgeryNo-> dentistName

\* appointmentDate, time, surgeryNo-> patName

- Here, the primary key is (appointmentDate, time, surgeryNo)

We see: appointmentDate, time, surgeryNo -> staffNo. But staffNo -> dentistName, so dentistName is transitive dependent. Violates 3NF

Similarly: appointmentDate, time surgeryNo -> patNo -> patName

- We will split the table into smaller tables to remove transitive dependencies

**Appointment(appointmentDate, time, surgeryNo, staffNo, patNo)**

Primary key: (appointmentDate, time, surgeryNo)

Foreign key: staffNo, patNo

**Staff (staffNo, dentistName)**

Primary key: staffNo

Alternate key: dentistName ( assume dentist name are unique)

**Patient(patNo, patName)**

Primary key: patNo

Alternate key: patName ( assume patient name are unique)

**Problem 14.14**

a) There is several reasonable assumptions for defining functional dependencies:

\* Each patient code uniquely identifies a name, a room, and a bed: PatientNumber -> FullName, WardNumber, BedNumber

\* Each room code uniquely identifies a room name: WardNumber -> WardName

\* Each drug code uniquely identifies the name and description of the drug: DrugNumber -> Name, Description

\* Each patient may be prescribed the medication multiple times, but each time with a different start date, dosage, route of administration, and end date: PatientNumber, DrugNumber, StartDate -> Dosage, MethodOfAdmin, UnitPerDate, FinishDate

b) The table have 1NF already because we have atomic value and no repeating group

Based on functional dependencies that we identified above, we have transitive dependencies.

PaitientNumber -> FullName, WardNumber, BedNumber

But WardNumber -> WardName

So we need split table into smaller tables to remove transitive dependencies

Patient (PatientNumber -> FullName, WardNumber, BedNumber)

Ward (WardNumber -> WardName)

Drug (DrugNumber -> Name, Description)

Prescription (PatientNumber, DrugNumber, StartDate -> Dosage, MethodOfAdmin, UnitsPerDay, FinishDate)

c)

**Patient (PatientNumber -> FullName, WardNumber, BedNumber)**

Primary key: PatientNumber

Alternate key: FullName, WardNumber, BedNumber (assume patient names are unique within a room and bed)

Foreign key: BedNumber

**Ward (WardNumber -> WardName)**

Primary key: WardNumber

Alternate key: WardName (assume room name are unique)

**Drug (DrugNumber -> Name, Description)**

Primary key: DrugNumber

Alternate key: Name (assume drug name are unique)

**Prescription (PatientNumber, DrugNumber, StartDate -> Dosage, MethodOfAdmin, UnitsPerDay, FinishDate)**

Primary key: PatientNumber, DrugNumber, StartDate