Lab6

- ***Exercise 14.14***

(a)

Assumptions:

1. Patient Number uniquely identifies a patient.
2. Ward Number uniquely identifies a ward.
3. Each bed number is unique within a ward : the pair {Ward Number, Bed Number} uniquely identifies a bed.
4. Drug Number uniquely identifies a drug (with its name, description and dosage).
5. A patient can receive the same drug multiple times, so we must distinguish prescriptions by date and dosage.

Functional Dependencies (FDs):

1. Patient Number → Full Name, Ward Number, Bed Number
2. Ward Number → Ward Name
3. Drug Number → Drug Name, Description, Dosage
4. Patient Number, Drug Number, Start Date → Method of Administration, Units per Day, Finish Date

(We use Start Date to differentiate multiple prescriptions of the same drug.)

*(b)***Normalize to 3NF:**

Step 1: UNF → 1NF

Already presented in tabular format (no repeating groups), so it's in 1NF.

Step 2: 1NF → 2NF

Eliminate partial dependencies.

We create separate relations for:

Relation: PATIENT

PATIENT(PatientNumber, FullName, WardNumber, BedNumber)

FD: PatientNumber → FullName, WardNumber, BedNumber

Relation: WARD

WARD(WardNumber, WardName)

FD: WardNumber → WardName

Relation: DRUG

DRUG(DrugNumber, DrugName, Description, Dosage)

FD: DrugNumber → DrugName, Description, Dosage

Relation: PRESCRIPTION

PRESCRIPTION(PatientNumber, DrugNumber, StartDate, MethodOfAdministration, UnitsPerDay, FinishDate)

FD: PatientNumber, DrugNumber, StartDate → MethodOfAdministration, UnitsPerDay, FinishDate

Step 3: 2NF → 3NF

Check for transitive dependencies.

Already removed transitive dependencies in previous steps. All relations are in **3NF**.

(c) Keys

PATIENT

- Primary Key: PatientNumber
- Alternate Key: None
- Foreign Key: WardNumber → WARD

WARD

- Primary Key: WardNumber

- Alternate Key: None
- Foreign Key: None

DRUG

- Primary Key: DrugNumber
- Alternate Key: None
- Foreign Key: None

PRESCRIPTION

- Primary Key: (PatientNumber, DrugNumber, StartDate)
- Alternate Key: (PatientNumber, StartDate, DrugNumber) (same)
- Foreign Keys:
    - PatientNumber → PATIENT
- DrugNumber → DRUG

- **Exercise 14.15**

A. Update anomalies

- Insertion anomaly: The table doesn't allow creating a new dentist data unless they are assigned a patient.

- Deletion anomaly: Because appointment and dentist info are tightly coupled, deleting an appointment will also remove all details about that dentist.

- Update anomaly: Updating name of a dentist will require us to go through the list and update every occurrence of that dentist. Otherwise, the dentist info will be inconsistent.

B. Identify functional dependencies

- Each staff number (staffNo) corresponds to a unique dentist name

- Each patient number (pathNo) corresponds to one unique patient name

- staffNo and appointment date time combine to create a unique appointment ID

- A dentist will have one surgery a day

- A patient can have at most one surgery a day

- Surgery times are not shared

C. Describe and illustrate the process of normalizing the table

- Step 1- UNF
- Dentist name repeats for same staffNo
- Patient name repeats for same patNo
- Surgery number repeats for same dentist on same date
- Step 2- 1NF
- Table already satisfies this rule as there are no repeating groups
- Step 3 - 2NF
- Removing partial dependency
- We should create Dentist, Patient and SurgeryAssignment tables and use foreign key of each in the Appointment table.

Dentist {
staffNo - Primary key,
dentistName,
}
Patient {
patNo
patName
}
SurgeryAssignment {
staffNo,
appointmentDate,
surgeryNo,
PRIMARY KEY (staffNo, appointmentDate)
}Appointment (
staffNo,
appointmentDate,
time,
patNo,
PRIMARY KEY (staffNo, appointmentDate, time),
FOREIGN KEY (staffNo, appointmentDate) REFERENCES SurgeryAssignment,
FOREIGN KEY (staffNo) REFERENCES Dentist,
FOREIGN KEY (patNo) REFERENCES Patient

)

- Step 4 - 3NF

Above table satisfies 3NF

Primary keys

Dentist: staffNo

Patient: patNo

SurgeryAssignment: (staffNo, appointmentDate)

Appointment: (staffNo, appointmentDate, time)

Foreign keys

Appointment.staffNo - Dentist.staffNo

Appointment.patNo - Patient.patNo

Appointment.(staffNo, appointmentDate) - SurgeryAssignment.(staffNo, appointmentDate)

Alternate Keys

None are declared

- **Exercise 14.16**

(a)

Insertion anomalies: You cannot insert a new hotel until at least one staff member is assigned to it.

Delete anomalies: Deleting the last record of a staff working at a hotel may remove all traces of that hotel from the table.

Modification anomalies: If a hotel changes its name, you must update every occurrence. If one is missed, inconsistent data appears.

(b) Functional dependency are:

NIN → eName

hNo → hLoc

NIN, hNo, contractNo → hours

Assumptions:

- A staff member (NIN) works at only one hotel at a time.
- hLoc doesn't repeat across different hNo.

(c) To normalize the table to 3NF

Step 1: First Normal Form (1NF)

The original table seems to be in 1NF, no repeating groups or arrays.

Step 2: Second Normal Form (2NF)
Remove partial dependencies (attributes depending only on part of the composite key).

Create separate tables:

1. **Staff**
   NIN (PK), eName
2. **Hotel**
   hNo (PK), HotelName
3. **Contract**
   contractNo (PK), description
4. **Assignment**
   NIN (FK), HotelNo (FK), contractNo (FK), hours

Step 3: Third Normal Form (3NF)

Already in 3NF because:

- No transitive dependencies (non-key attributes depending on other non-key attributes)
- All attributes in each table are dependent on the key, the whole key, and nothing but the key.

Keys

- **Primary Keys:**
  - Staff → NIN
  - Hotel → hNo
  - Contract → contractNo
  - Assignment → NIN + hNo + contractNo

- **Foreign Keys:**
    - Assignment.NIN → Staff.NIN
    - Assignment.HotelNo → Hotel.hNo
    - Assignment.contractNo → Contract.contractNo