CS457 Assignment

Wellness Clinic Medical Group

Yixuan Liang

Shanshan Du

Jiajia Ding

Scenario

The Wellness Clinic is a facility providing medical care in a rural area of the country. Its professional staff consists of five medical doctors (physicians), two nurse-practitioners who provide non-acute care and can prescribe medication, two registered nurses, two midwives who provide pre-natal care and supervise delivery except in cases with complications, a pharmacist, and a medical technician. The non-professional staff members include an office administrator, a receptionist, and a bookkeeper who works part time. The clinic serves several thousand patients, each of whom may visit the clinic any number of times per year, both for preventative care such as checkups or immunizations, and for treatment of illness. Its facilities consist of a waiting room with a reception desk, an administrative office, a nurses' station, ten examining rooms with adjoining consultation rooms, a small operating room, a birthing room, a recovery room, a pharmacy, and a small laboratory.

- Basic Operations

The clinic has regular hours of operation weekdays, Saturday mornings, and two evenings per week. Normally two physicians, one nurse practitioner, one registered nurse, and one midwife are in the clinic during regular hours. In addition, the physicians and nurse-practitioners rotate responsibility for covering emergency calls 24 hours per day, 7 days a week. At the end of each day, the receptionist sets up call forwarding so that emergency calls are automatically directed to the telephone number of the person providing emergency coverage. When the clinic opens in the morning, the call forwarding is halted. Two of the physicians are surgeons who perform routine surgery not requiring general anesthesia at the clinic one morning a week, assisted by a registered nurse. Others have specialties in pediatrics and internal medicine. However, all of the physicians can provide general and acute care for any of the patients. Patients who require major surgery or other hospital care must go to a hospital located outside the immediate area served by the clinic. The clinic physicians do not normally visit their patients who are in the hospital, instead leaving their care to the hospital staff with whom they communicate during the hospitalization. However, the clinic provides both pre- and post-hospital care for the patients.

Hours of operation are divided into scheduled appointments and unscheduled hours, which are open for walk-ins. Patients usually schedule checkups and immunizations well in advance. Patients suffering from chronic or acute illness can usually schedule appointments promptly, or they may come in during the unscheduled hours. The administrator is responsible for setting up all schedules, and for keeping records updated. Prior to the beginning of each month, the administrator makes up complete coverage schedules for all professional and support staff. The bookkeeper is responsible for doing all billing and recording payments. The receptionist is responsible for making patient appointments, for handling traffic, and for making the patient's medical records available during the visit. The nurse prepares the patient, takes medical history, performs some medical routines or tests, takes samples for lab tests, updates the medical record, and assists the practitioner (the physician, nurse practitioner, or midwife) during the visit. The practitioner examines the patient, administers medical treatment, can perform some tests, can also take samples for lab tests, and writes prescriptions for medications or orders for additional lab tests during a visit. Each visit results in one or more diagnoses, which the practitioner adds to the patient's medical record, along with any comments or observations. Prescriptions can be filled at the clinic's pharmacy, or they can be sent to another pharmacy at the patient's request. Some laboratory tests are performed at the clinic by the medical technician, using samples taken by one of the professionals. More specialized tests are performed at an outside medical laboratory. Whenever possible, specimens, such as blood samples, are taken at the clinic by one of the professionals there and sent to the laboratory. If the lab test requires the presence of the patient and equipment that is not available at the clinic, the patient is sent to the outside laboratory for the test, and results are sent back to the clinic.

Medical care is provided for all patients, regardless of their ability to pay. Bills are generated based on the services provided, not on the payment method. Private patients who can afford to pay out of pocket can do so at the time of service or be billed at the end of each month. Those who have medical insurance provide information about their insurance policies, and the insurance companies are billed. Usually in that case patients pay a small amount of co-insurance (co-pay), which is determined by the type of policy they hold, at the time of the visit. Those who cannot afford to pay normally have government-provided health care, for which they have a government-issued medical card. They pay nothing and the clinic is reimbursed by the government for the entire cost of the visit, including any lab tests performed and medications dispensed there. A small number of indigent patients who do not have health coverage are treated and the cost is absorbed by the clinic until they qualify for government-provided coverage.

- Information Needs

Currently all information about patients and their care is kept manually, and billing information is kept on a spreadsheet. Physicians use computer or telephone communications to provide information to the hospital and receive information about patients who need hospital care. The clinic has computer access to hospital records for its patients, as well as on-line systems provided by insurance companies and the government for third-party billing. It needs a database that keeps track of all the patient-related activities of the clinic and to provide information about billing and payments. The database will not keep track of medical supplies, plant maintenance, or payroll information.

Abstract

Problem Statement

The **Wellness Clinic Medical Group** is currently keeping their records such as patient information, staff (e.g., physicians, nurse-practitioners, registered nurses, and midwives) schedules, hospital facilities usage logs, and billing information manually on an Excel spreadsheet. Now the **Wellness Clinic Medial Group** is considering converting all the manual works into automated process and saved in a database. We are now helping the medical group to build a database from scratch to keep track of all the patient-related activities of the clinic and to provide information about billing payments based on their needs.

Their needs, in terms of Excel forms and reports, include:

- Patient Intake Form that is filled out by patients prior to or during their initial visit containing their personal information.
- Weekly Coverage Schedule that lists the weekly schedule of the clinic staffs.
- Daily Master Schedule that lists daily appointment schedule for the practitioners
- Individual Practitioner's Daily Schedule that is a printed copy of his/her own daily schedule in the clinic.
- Provider's Statement for Insurance Form that is a pre-printed form used as a receipt primarily for insurance purposes.
- Patient Monthly Statement that has the unpaid balance for any patient at the end of each month.
- Prescription Label and Receipt that consist of two parts. The first part used as a label for the container of the medication and the second part is used for submitting claims for insurance coverage.

- Daily Laboratory Log that is used to record all lab test performed each day.
- Operation Room Schedule that provides information about schedule surgeries for that day.
- Operation Room Log that records information about the surgeries performed on a given day.
- Daily Delivery Room Log that records information about the deliveries performed each day.
- Recovery Room Log that records information about the use of the recovery room.
- Monthly Activity Report that is an internal report summarizing the clinic's activity each month.

Approach

We are going to resolve the problem using the following steps:

1. Identify the needs of **Wellness Clinic Medical Group** (by collecting forms and reports)
2. Build the data dictionary (based on the data attributes from the forms and reports)
3. Build Entity Relationship Diagram (after identifying the entities)
4. Build Extended Entity Relationship Diagram
5. Create the database and the tables based on the Extended Entity Relationship Diagram

Based on their current manual forms and reports, we are expected to fully understand their operation process, summarize the information and collect the data attributes and points to build the data dictionary. And based on the manual input from the forms and reports, we can determine the data definition, datatype, data length, and provide examples for specific data attributes. For example, from the form of Prescription Label and Receipt, 'Drug Name' is defined as a patient's perscribed medication name that is going to paste as a label on the container, and the data attribute name can be 'drugName' and the datatype and data length can be a VARCHAR(30). Some examples for the 'drugName' could be Alprazolam, Tylenol, Mucinex, etc.

After the requirements are collected and data dictionary is built, we are going to build an Entity Relation Diagram from which major entities (strong entities and weak entities) such as Patient, Staff, Appointment, and Facilities can be defined, and their relationship can be drawn using a straight line and simple attributes can be added to the entities. Afterwards, assumptions about the entities' relationship – cardinality, would be made and more comprehensive Extended Entity Relation Diagram can be created. For example, the cardinality of a patient and a physician is many-to-many because a patient can visit multiple physicians and a physician can treat multiple patients.

In the end, the database will be initiated by creating tables based on the entities with the attributes and relationship using SQL statements. The attributes can be served as columns of the tables, and the relationship can be represented using primary key and foreign key. In details,

- Tables can be initiated using 'CREATE TABLE' statement, from which the column data type, data length, constraints (NOT NULL, CHECK, and UNIQUE, etc.) are added to guarantee the security of data.
- PRIMARY KEY and FOREIGN KEY can be used to define the relationship of tables. PRIMARY KEY and FOREIGN KEY can be added using 'ALTER TABLE' after the tables are initiated.
- The records can be dumped into the table using 'INSERT INTO' statement.

Last but not least, read over the requirements of the **Wellness Clinic Medical Group**, compare the information from the manual forms and reports against the data attributes from the data dictionary, compare the columns of the database and tables against the entities, attributes, relationship, and cardinality from the Extended Entity Relationship Diagram.

Solution Approach

Patient Intake Form													
Patient ID	Last Name	First Name	Initial Visit Date	Email Address	Phone Number	Primary Residential Address	City	State	Zip Code	Current Medications	Insurance Company	Insurance Plan	Member ID
111	XX	XX	1/1/2023	abc@hotmail.com	999-999-9999	123 Linden St	XX	XX	11111	XX	XX	XX	1111111
222	XX	XX	1/1/2023	abc@hotmail.com	999-999-9999	123 Linden St	XX	XX	11111	XX	XX	XX	1111111
333	XX	XX	1/1/2023	abc@hotmail.com	999-999-9999	123 Linden St	XX	XX	11111	XX	XX	XX	1111111

Weekly Coverage Schedule							
Date	Daily Hours From	Daily Hours To	Staff ID	Staff Last Name	Staff First Name	Professional	Phone Number
1/1/2023	10:00 AM	12:00 PM	111	XX	XX	Yes	111-111-1111
1/1/2023	2:00 PM	5:00 PM	222	XX	XX	No	111-111-1111

Daily Master Schedule									
Appointment ID	Appointment Hours From	Appointment Hours To	Appointment Type	Assigned Practitioner	Practice	Staff ID	Patient ID	Patient Last Name	Patient First Name
1	10:10 AM	10:20 AM	scheduled	physician	Surgery	999	111	XX	XX
2	10:20 AM	10:30 AM	scheduled	physician	Surgery	999	222	XX	XX
3	4:20 PM	4:30 PM	scheduled	nurse-practitioner	Prescription	999	333	XX	XX

4	4:30 PM	4:40 PM	scheduled	midwife	Delivery	999	444	XX	XX
5	10:20 AM	4:20 PM	walk-in	physician	Care	999	555	XX	XX

Individual Practitioner's Daily Schedule							
Date	Appointment Hours From	Appointment Hours To	Appointment Type	Patient ID	Patient Last Name	Patient First Name	Reason for Visit
1/1/2023	11:00 AM	11:30 AM	scheduled	111	XX	XX	XX
1/1/2023	11:30 AM	11:40 AM	scheduled	222	XX	XX	XX
1/1/2023	11:40 AM	11:50 AM	scheduled	333	XX	XX	XX
1/1/2023	12:00 PM	1:00 PM	walk-in	444	XX	XX	XX

Provider's Statement for Insurance Forms

Information of Clinic

Clinic Name: _______________________
Street _________________________ City ___________ State __________ Zip __________
Telephone: Area Code _____ Number ______________________

Practitioners on Staff

□ Medical Doctor1	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Medical Doctor2	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Medical Doctor3	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Medical Doctor4	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Medical Doctor5	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Non-acute Nurse1	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Non-acute Nurse1	XXX	Tax ID Number : XXXXXXXXXXXXXXX

□ Registered Nurse1	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Registered Nurse2	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Midwive1	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Midwive1	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Pharmacist	XXX	Tax ID Number : XXXXXXXXXXXXXXX
□ Medical Technician	XXX	Tax ID Number : XXXXXXXXXXXXXXX

Visit Type

□ Checkups
□ Pre-care
□ Immunizations
□ Post-care
□ Chronic illness
□ Emergency call
□ Acute illness
□ Other:_______________

Procedures Performed

Procedure	Code	Fee
□ Perform medical routines or tests	XXX	_________________
□ Take samples	XXX	_________________
□ Prescriptions	XXX	_________________
□ Laboratory tests(inside)	XXX	_________________
□ Laboratory tests(outside)	XXX	_________________
□ Pre-hospital care	XXX	_________________
□ Post-hospital care	XXX	_________________
□ Other: _________________	_________________	_________________
□ Other: _________________	_________________	_________________
□ Other: _________________	_________________	_________________

Diagnosis

Diagnosis	Code
□ Upper respiratory infections	XXX
□ Hypertension	XXX
□ Diabetes	XXX
□ Minor injuries	XXX
□ Other: _________________	_________________

□ Other: ________________ ________________
□ Other: ________________ ________________

Total Charge: ___________________
Amount Paid: ___________________
Balance Due: ____________________

Patient Monthly Statement

Dear [Patient Name],

We are pleased to provide you with your Patient Monthly Statement for [Month, Year].

Services Provided:
[Date of Service] [Service Description] $[Amount]
[Date of Service] [Service Description] $[Amount]
[Date of Service] [Service Description] $[Amount]
[Date of Service] [Service Description] $[Amount]
[Date of Service] [Service Description] $[Amount]

Payments Received:
[Date of Payment] [Payment Description] $[Amount]
[Date of Payment] [Payment Description] $[Amount]

Balance Due:
Total Services Provided: $[Total Amount]
Total Payments Received: $[Total Payment]
Balance Due: $[Balance Due]

Please remit payment in the amount of [Balance Due] by [Due Date] to avoid any further collection efforts. If you have any questions about your statement, please do not hesitate to contact us.

Sincerely,

[Your Name]
[Your Title]
[Wellness Clinic]

Prescription Label and Receipt

Prescription Label

Rx Number: _________
Doctor Name: _________
Patient Name: _________ Patient Address: _________
Directions: _________
Drug Name: _________ Form: _________ Strength: _________ Quantity: _________
Pharmacist's Name: _________ Date Filled: _________ Original Date: _________
Refills Remaining: _________

Prescription Receipt

Rx Number: _________
Doctor Name: _________
Patient Name: _________ Patient Address: _________
Directions: _________
Drug Name: _________ Form: _________ Strength: _________ Quantity: _________
Pharmacist's Name: _________ Date Filled: _________ Original Date: _________
Refills Remaining: _________
Total Price: _________
Amount Covered by Insurance/Government: _________
Balance Due from Patient: __________________________
Drug Information:

Directions for Use:

Warnings:

Daily Laboratory Log
Date: xxxx-xx-xx

TestID	Patient Name	Test Name	Specimen Type	Result	Description	Performed By	Reviewed By
xxx	xxx	xxx	xxx	xxx	xxx	xx	xxx
xxx	xxx	xxx	xxx	xxx	xxx	xx	xxx
xxx	xxx	xxx	xxx	xxx	xxx	xx	xxx
xxx	xxx	xxx	xxx	xxx	xxx	xx	xxx

Operating Room Schedule

surgeon_id	patient_id	nurse_id	surgery_date	surgery_start_time	surgery_end_time	surgery_type	Anesthesia	assistant	room_id	remarks
1	1001	2001	03-05-2023	15: 30:33, 03-05-2023	15: 30:33, 03-05-2023	xxxxx	yes	Susan	3001	xxxxxx
2	1002	2002	03-05-2024	15: 30:33, 03-05-2024	15: 30:33, 03-05-2024	xxxxx	no	July	3002	xxxxxx
3	1003	2003	03-05-2025	15: 30:33, 03-05-2025	15: 30:33, 03-05-2025	xxxxx	yes	Alic	3003	xxxxxx
4	1004	2004	03-05-2026	15: 30:33, 03-05-2026	15: 30:33, 03-05-2026	xxxxx	no	Tom	3004	xxxxxx
5	1005	2005	03-05-2027	15: 30:33, 03-05-2027	15: 30:33, 03-05-2027	xxxxx	yes	Jason	3005	xxxxxx

Operating Room Log

patient_id	surgeon_id	nurse_id	surgery_date	surgery_start_time	surgery_end_time	surgery_type	observations
1	1001	2001	03-05-2023	15: 30:33, 03-05-2023	15: 30:33, 03-05-2023	xxxxx	xxxxx
2	1002	2002	03-05-2024	15: 30:33, 03-05-2024	15: 30:33, 03-05-2024	xxxxx	xxxxx
3	1003	2003	03-05-2025	15: 30:33, 03-05-2025	15: 30:33, 03-05-2025	xxxxx	xxxxx
4	1004	2004	03-05-2026	15: 30:33, 03-05-2026	15: 30:33, 03-05-2026	xxxxx	xxxxx
5	1005	2005	03-05-2027	15: 30:33, 03-05-2027	15: 30:33, 03-05-2027	xxxxx	xxxxx

Daily Delivery Room Log

deliveryId	patientId	pharmacist	medical technician	midwives	supervise delivery
1	1001	Susan	Susan	Susan	Susan
2	1002	July	July	July	July
3	1003	Alic	Alic	Alic	Alic
4	1004	Tom	Tom	Tom	Tom
5	1005	Jason	Jason	Jason	Jason

Recovery Room Log

PatientName	AttendingPractitioner	Bed	DateIn	TimeIn	DateOut	TimeOut	SignatureOfThePractitioner	MedicalChecks
Susan	Susan	1001	03-05-2023	15: 30:33, 03-05-2023	03-20-2023	16: 30:33, 03-05-2023	Susan	xxxxx
July	July	1002	03-05-2024	15: 30:33, 03-05-2024	03-20-2024	16: 30:33, 03-05-2024	July	xxxxx
Alic	Alic	1003	03-05-2025	15: 30:33, 03-05-2025	03-20-2025	16: 30:33, 03-05-2025	Alic	xxxxx

Tom	Tom	1004	03-05-2026	15: 30:33, 03-05-2026	03-20-2026	16: 30:33, 03-05-2026	Tom	xxxxx
Jason	Jason	1005	03-05-2027	15: 30:33, 03-05-2027	03-20-2027	16: 30:33, 03-05-2027	Jason	xxxxx

*Note: A nurse should record any medical checks performed and their results in a separate log or system, and this information should be cross-referenced with the recovery room log. Additionally, the practitioner who signs the patient out should review and confirm the information recorded in the log.

Monthly Activity Report						
Activity Item	Monthly Total	Provider 1	Provider 2	Provider 3	…	Average Time per Visit
Visits Conducted	345	Susan	Susan	Susan		1 hour
Surgeries Performed	345	July	July	July		2 hours
Deliveries	345	Alic	Alic	Alic		3 hours
Lab Tests Conducted	345	Tom	Tom	Tom		4 hours
Lab Tests (Type 1)	345	Jason	Jason	Jason		5 hours

Data Dictionary

Data Attribute Name	Data Definition	Datatype and Data Length	Examples	Patient Intake Form	Weekly Coverage Schedule	Daily Master Schedule	Individual Practitioner Daily Schedule	Provider's Statement for Insurance Forms	Patient Monthly Statement	Prescription Label and Receipt	Daily Laboratory Log	Monthly Activity Report	Recovery Room Log	Daily Delivery Room Log	Operating Room Log	Operating Room Schedule
activity_item	The name of the hospital business,for example, visist number, Deliveries, test .etc.	VARCHAR(38)	the number of visitor, the number of surgeries performed, the number of deliveries, etc.									x				
AmountPaid	The amount of money patient has already paid	FLOAT						x	x							
anesthesia	What type of anesthesia is the patient undergoing for	VARCHAR(38)	general anesthesia or local anesthesia													x

	this operation, general anesthesia or local anesthesia															
AppointmentHoursFrom	Appointment Starting Hour on Daily Schedule	DATETIME	2018/01/01 10:00:00			x	x									
AppointmentHoursTo	Appointment Ending Hour on Daily Schedule	DATETIME	2018/01/01 10:00:00			x	x									
AppointmentID	Appointment ID	INT(3)				x										
AppointmentType	Appointment Type	VARCHAR(38)	scheduled, walk-in			x	x									
AssignedPractitioner	Assigned Practitioner	VARCHAR(38)	physician, nurse practitioner, midwife			x										
assistant	Operating theater nurses, provide support to surgeons, ensure the safety and hygiene of operating theaters, and care for surgical patients.	VARCHAR(38)	itinerant nurse, scrub nurses													x
attending_practitioner	The name of the attending practitioner.	VARCHAR(38)											x			
BalanceDue	The balance patient should pay	FLOAT						x	x	x						
bed_id	The number of the bed where the patient is recovering.	INT(5)											x			
City	City	VARCHAR(38)		x												
ClinicCity	The city of the clinic	VARCHAR(50)						x								
ClinicName	The name of the clinic	VARCHAR(38)	Wellness Clinic					x	x							

ClinicState	The state of the clinic	VARCHAR(2)	NY, CA					x								
ClinicStreet	The street of the clinic	VARCHAR(100)						x								
ClinicTelephone	The telephone number for the clinic	VARCHAR(14)	(123)456-7890					x								
ClinicZip	The zip code of the clinic	VARCHAR(20)	10001-1234, 90210					x								
CurrentMedications	Patient's Current Medications	VARCHAR(100)	Tylenol, Iburprophen, Laxative	x												
DailyHoursFrom	Shift Starting Hours on Weekly Coverage Schedule	DATETIME	2018/01/01 10:00:00		x											
DailyHoursTo	Shifting Finishing Hours on Weekly Coverage Schedule	DATETIME	2018/01/01 10:00:00		x											
Date	Date of the Individual Practitioner's Daily Schedule	DATE					x									
date_in	The date when the patient entered the recovery room	DATE	2023-01-01										x			
date_out	The date when the patient left the recovery room.	DATE											x			
delivery_Id	The unique identifier for each delivery performed each day.	INT(5)												x		
Diagnosis	The result doctors diagnose for patient, including upper respiratory infection, hypertension, diabetes, minor injuries and so on	VARCHAR(100)	upper respiratory infection					x								

DiagnosisCode	The unique code for a specific diagnosis	INT(20)						x								
Direction	The instruction for drug use	VARCHAR(200)								x						
DrugFillDate	Date that the drug was actually filled in the pharmacy	DATE								x						
DrugForm	The form of the drug	VARCHAR(50)	tablet, Capsule, Syrup, Injection, Topical cream							x						
DrugName	The name of the drug	VARCHAR(50)								x						
DrugOriginalDate	The original date for the drug	DATE								x						
DrugPrice	Price for the drug	FLOAT								x						
DrugQuantity	The quantity or weight of the specific drug	FlOAT	10mg, 100ml, 30 tablets							x						
DrugRefillsRemain	How many refills you have left	INT								x						
DrugStrength	The amount of drug in a given dosage form	FLOAT	25mg, 5,5g, 2.5ml							x						
DrugWarning	The warning for a drug	VARCHAR(200)								x						
EmailAddress	Patient Email Address	VARCHAR(38)		x												
Fee	The money patient paid for a procedure	FLOAT						x	x							
InitialVisitDate	Patient Initial Visit Date	DATE	2023-01-01	x												
InsuranceCompany	Patient Insurance Company	VARCHAR(38)		x												
InsuranceCoverAmount	The amount of money that could be covered by insurance	FLOAT								x						

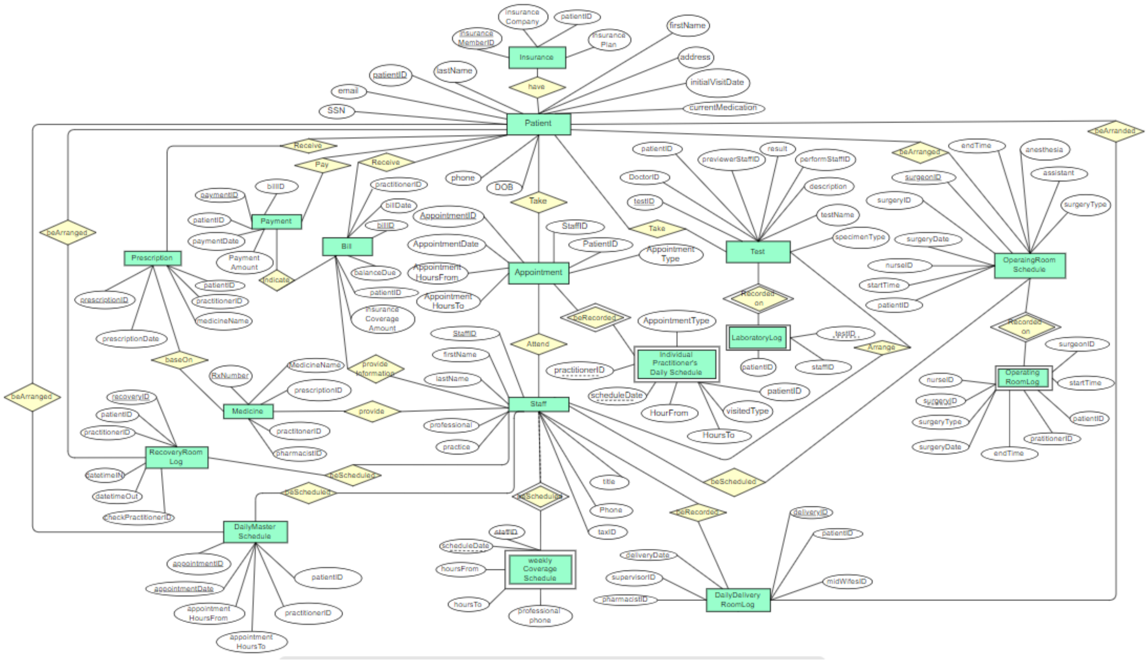
InsurancePlan	Patient Insurance Plan	VARCHAR(38)		x												
LogDate	The date of the log	DATE									x					
medical_checks	Times and results of any medical checks performed on the patient while in recovery.	VARCHAR(38)											x			
medical_technician	Professional staff who use a variety of medical equipment and technology to assist doctors and other medical professionals in diagnosing, treating and monitoring the health of patients	VARCHAR(38)	Medical Laboratory Technician,Radiologic Technologist											x		
MemberID	Patient Insurance Member ID	INT(7)		x												
midwives	Provide pre-natal care and supervise delivery except in cases with complications,	VARCHAR(38)												x		
nurse_id	The unique identifier for the nurse who is assisting the surgeon during the surgery.	INT(5)													x	x
observations	Any additional observations or notes that are relevant to the surgery or the patient's condition.	VARCHAR(38)	device breakage, bleeding more than expected												x	
patient_Id	The unique identifier for each maternity	INT(5)												x		

PatientAddress	The address for the patient	VARCHAR(100)								x						
PatientFirstName	Patient First Name	VARCHAR(38)		x		x	x									
PatientID	Patient ID	INT(3)		x		x	x								x	
PatientLastName	Patient Last Name	VARCHAR(38)		x		x	x									
PatientName	The name of the patient	VARCHAR(30)							x	x	x		x			
pharmacist	The name of the pharmacist responsible for each pregnant woman's medication	VARCHAR(38)												x		
PhoneNumber	Patient Phone Number	INT(10)		x												
PhoneNumber	Staff Phone Number	INT(10)			x											
Practice	Practice	VARCHAR(38)	Surgery, Prescription, Delivery, Care			x										
PractitionersName	The name of the practitioner in the clinic	VARCHAR(50)	John Smith					x	x	x	x					
PractitionersTaxID	The tax ID of the practitioner in the clinic	VARCHAR(9)	123-45-6789					x								
PractitionersTitle	The title for the practitioner in the clinic	VARCHAR(50)	pharmacist					x	x							
PrimaryResidentialAddress	Patient Primary Residential Address	VARCHAR(100)		x												
Procedure	Procedures for patient performed in the clinic, including medical routines or test, take samples, prescriptions, laboratory test, pre-hospital care,	VARCHAR(100)	Medical routine, take samples, prescriptions, laboratory tests, pre-hospital care, post-hospital care, other					x	x							

	post-hospital care and other															
ProcedureCode	The unique for a specific procedure	INT(20)						x								
Professional	Professional	Bool	Yes, No		x											
ReasonforVisit	Reason for Visit	VARCHAR(38)	flu, pregnancy				x									
room_id	The unique identifier for operating room	INT(5)														x
RxNumber	The unique number for prescription modecine	VARCHAR(10)								x						
ServiceDate	The date patient come to the clinic and perform procedure	DATE							x							
signature_of_practitioner	The signature of the practitioner who signs the patient out.	VARCHAR(38)											x			
SpecimenType	The type of species used in the test	VARCHAR(50)	blood, tissue, urine								x					
StaffFirstName	Staff First Name	VARCHAR(38)			x											
StaffID	Staff ID	INT(3)			x	x										
StaffLastName	Staff Last Name	VARCHAR(38)			x											
State	State	VARCHAR(2)		x												
surgeon_id	The unique identifier for the surgeon who is performing the surgery.	INT(5)													x	x
surgery_date	The date on which the surgery is scheduled to take place.	DATE													x	x

surgery_end_time	The time at which the surgery is scheduled to end.	DATETIME													x	x
surgery_start_time	The time at which the surgery is scheduled to begin.	DATETIME													x	x
surgery_type	The type of surgery that is being performed.	VARCHAR(38)	urgent, therapeutic												x	x
TestDescription	The description of the test	VARCHAR(200)									x					
TestID	The unique id for the test of laboratory	INT(50)									x					
TestName	The name for the test	VARCHAR(40)									x					
TestResult	The result of the test	VARCHAR(30)	positive, negative, high, low								x					
time_in	The time when the patient entered the recovery room.	DATETIME	2023-01-01 10:10:30										x			
time_out	The time when the patient left the recovery room.	DATETIME											x			
TotalCharge	The total amount of money patient should pay	FLOAT						x	x							
VisitType	The visite type for patient, including Checkups, immunizations, chronic illness, acute illness, pre-care, post-care, emergency call and other	VARCHAR(40)	checkups, immunizations , chronic illness, acute illness, pre-care, post-care, emergency call, other					x								
WeeklyScheduleDate	Date of Weekly Coverage Schedule	DATE			x											
ZipCode	Zip Code	INT(5)		x												

Step 3: based on the requirements outlined in Forms and Reports, we identified the entities (strong entities and weak entities) with their attributes, and the relationships among the entities. For their cardinality, in details, Patient and Appointment is 1:M, Staff and Appointment is 1:M, Patient and Test is 1:M, Patient and Prescription is 1:M, Patient and Operating Room Schedule is 1:M, Prescription and Medicine is 1:M, Staff and Test is 1:M, Staff and Medicine is 1:M, Staff and Operating Room Schedule is M:M, Patient and Recovery Room Log is 1: M, Patient and Daily Delivery Room Log is 1:M, Staff and Recovery Room Log is M:M, Staff and Daily Delivery Room Log is M:M, Patient and Insurance is 1:1, Patient and Payment is 1:M, Patient and Bill is 1:M, Staff and Bill is M:M, and Bill and Payment is M:M.



In our EER diagram shown in the above Figure, we have multiple entities, their relationships (i.e., one-to-many, many-to-one, one-to-one, and many-to-many), and unions of specialties. For example, Staff is a union of professional staff and non-professional staff. Professional staff consists of practitioners, registered nurses, midwives, a laboratory technician, and a pharmacist, and non-professional staff consists of a receptionist, an administrator and a bookkeeper. We examine each of the entities and relationships to decide whether to represent them in a relational schema.

For staff hierarchy, we decided not to create one table for each specialty, but to use a single table for all staff hierarchy in general. We decide to include the other entities in the relational tables, Patient, Appointment, Staff, Test, Operating Room Schedule, Insurance, Bill, Prescription, Medicine, Payment, Daily Master Schedule, Weekly Coverage Schedule, Individual Practitioner Schedule, Laboratory Log, Operating Room Log, Daily Delivery Room Log, and Recovery Room Log. We added a unique identifier ID to Patient, Staff, Appointment, Test, Operating Room Schedule, Insurance, Bill, Payment, Prescription, Medicine, Weekly Coverage Schedule, Individual Practitioner Schedule as the primary key, and make corresponding changes for the foreign keys of Appointment, Test, Laboratory Log, Operating Room Schedule, Operating Room Log, Daily Delivery Room Log, Recovery Room Log, Insurance, Bill, Payment, Prescription, Medicine, Daily Master Schedule, Weekly Coverage Schedule, Individual Practitioner Schedule.

The EER to relational mapping therefore gives us the following schema:

Patient (patientID, lastName, firstName, InitialVisitDate, email, phone, address, city, state, zipcode, DOB, SSN, currentMedication)

Staff (staffID, firstName, lastName, Phone, title, taxID, professional, practice)

Appointment (AppointmentID, AppointmentDate, AppointmentHoursFrom, AppointmentHoursTo, AppointmentType, StaffID, PatientID)

Insurance (InsuranceMemberID, patientID, InsuranceCompany, InsurancePlan)

Bill (billID, billDate, practitionerID, patientID, InsuranceCoverageAmount, BalanceDue)

Payment (PaymentID, BillID, PaymentDate, PaymentAmount, PatientID)

Prescription (PrescriptionID, PrescriptionDate, practitionerID, patientID, medicineName)

Medicine (RxNumber, MedicineName, prescriptionID, practitionerID, PharmacistID)

Test (testID, testName, DoctorID, patientID, specimenType, result, description, performStaffID, previewerStaffID)

OperatingRoomSchedule (surgeryId, surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType, anesthesia, assistant)

DailyMasterSchedule (appointmentId, appointmentDate, appointmentHoursFrom, appointmentHoursTo, practitionerId, patientId)

RecoveryRoomLog (recoveryID, patientID, practitionerId, dateIn, timeIn, dateOut, timeOut, checkPractitionerId)

DailyDeliveryRoomLog (deliveryId, deliveryDate, patientId, pharmacistId, midWifeId, supervisorId)

OperatingRoomLog (surgeryId, surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType)

LaboratoryLog (testID, StaffID, PatientID)

IndividualPractionerSchedule (practitionerID,ScheduleDate, HoursFrom, HoursTo, AppointmentType, PatientID, VisitType)

WeeklyCoverageSchedule (staffID, scheduleDate, hoursFrom, hoursTo, Professional, Phone)

Following the schema and tables creation, 6 steps are taken to manipulate the relational database.

Step 5.1 Update the data dictionary and list of assumptions as needed. For each table, write the table name and write out the names, data types, and sizes of all the data items, and identify any constraints, using the conventions of the DBMS you will use for implementation.

Table Patient

patientID	INT	3	(patientID) PRIMARY KEY NOT NULL
firstName	VARCHAR	38	
lastName	VARCHAR	38	
initialVisitDate	DATE		
email	VARCHAR	38	
phone	INT	10	
address	VARCHAR	1000	
city	VARCHAR	38	
state	VARCHAR	2	
zipcode	INT	5	
DOB	DATE		
SSN	INT	9	UNIQUE
currentMedicineTaken	VARCHAR	1000	

Table Staff

staffID	INT	3	(staffID) PRIMARY KEY NOT NULL
firstName	VARCHAR	38	
LastName	VARCHAR	38	
phone	INT	10	
title	VARCHAR	38	
taxID	INT	10	UNIQUE
professional	BOOL		
practice	VARCHAR	38	

Table Appointment

appointmentID	INT	3	(appointmentID) PRIMARY KEY NOT NULL

appointmentDate	DATE		
appointmentHoursFrom	DATETIME		
appointmentHoursTo	DATETIME		
appointmentType	VARCHAR	38	
staffID	INT	3	(staffID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table Test

testID	INT	38	(testID) PRIMARY KEY NOT NULL
testName	VARCHAR	38	
patientID	INT	3	(patientID) FOREIGN KEY
specimentType	VARCHAR	38	
result	VARCHAR	38	
description	VARCHAR	1000	
performStaffID	INT	3	(performStaffID) FOREIGN KEY
previewerStaffID	INT	3	(previewerStaffID) FOREIGN KEY

Table LaboratoryLog

testID	INT	38	(testID) FOREIGN KEY
staffID	INT	3	(staffID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table OperatingRoomSchedule

surgeryID	INT	3	(surgeryID) PRIMARY KEY NOT NULL
surgeonID	INT	3	(surgeonID) FOREIGN KEY
sugeryDate	DATE		
patientID	INT	3	(patientID) FOREIGN KEY
nurseID	INT	3	(nurseID) FOREIGN KEY
startTime	DATETIME		
endTime	DATETIME		
surgeryType	VARCHAR	38	
anesthesia	VARCHAR	38	
assistant	VARCHAR	38	

Table OperatingRoomLog

surgeryID	INT	3	(surgeryID) FOREIGN KEY
surgeonID	INT	3	(surgeonID) FOREIGN KEY
surgeryDate	DATE		
patientID	INT	3	(patientID) FOREIGN KEY
nurseID	INT	3	(nurseID) FOREIGN KEY
startTime	DATETIME		
endTime	DATETIME		
surgeryType	VARCHAR	38	

Table DailyDeliveryRoomLog

deliveryID	INT	5	(deliveryID, deliveryDate) PRIMARY KEY NOT NULL
deliveryDate	DATE		(deliveryID, deliveryDate) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY
pharmacistID	INT	3	(pharmacistID) FOREIGN KEY
midwifeID	INT	3	(midwifeID) FOREIGN KEY
supervisorID	INT	3	(supervisorID) FOREIGN KEY

Table RecoveryRoomLog

recoveryID	INT	3	(recoveryID) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY
practitionerID	INT	3	(practitionerID) FOREIGN KEY
dateIn	DATE		
timeIn	DATETIME		
dateOut	DATE		
timeOut	DATETIME		
checkPractitionerID	INT	3	(checkPractitionerID) FOREIGN KEY

Table Insurance

insuranceMemberID	INT	7	(insuranceMemberID) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY
insuranceCompany	VARCHAR	38	
insurancePlan	VARCHAR	38	

Table Bill

billID	INT	5	(billID) PRIMARY KEY NOT NULL
billDate	DATE		
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY
insuranceCoverageAmount	DECIMAL	38,2	
balanceDue	DECIMAL	38,2	

Table Payment

paymentID	INT	3	(paymentID) PRIMARY KEY NOT NULL
billID	INT	5	(billID) FOREIGN KEY
paymentDate	DATE		
paymentAmount	DECIMAL	38,2	
patientID	INT	3	(patientID) FOREIGN KEY

Table Prescription

prescriptionID	INT	3	(prescriptionID) PRIMARY KEY NOT NULL
prescriptionDate	DATE		
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY
medicineName	VARCHAR	38	

Table Medicine

RxNumber	VARCHAR	10	(RxNumber) PRIMARY KEY NOT NULL
medicineName	VARCHAR	38	
prescriptionID	INT	3	(prescriptionID) FOREIGN KEY
practitonerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY
pharmacistID	INT	3	(pharmacistID) FOREIGN KEY

Table DailyMasterSchedule

appointmentID	INT	3	(appointmentID) FOREIGN KEY
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table WeeklyCoverageSchedule

staffID	INT	3	(staffID, scheduleDate) PRIMARY KEY NOT NULL
scheduleDate	DATE		(staffID, scheduleDate) PRIMARY KEY NOT NULL
hoursFrom	DATETIME		
hoursTo	DATETIME		

Table IndividualPractitionerSchedule

practitionerID	INT	3	(practitionerID, scheduleDate) PRIMARY KEY NOT NULL
scheduleDate	DATE		(practitionerID, scheduleDate) PRIMARY KEY NOT NULL
hoursFrom	DATETIME		
hoursTo	DATETIME		
appointmentType	VARCHAR	38	
patientID	INT	3	(patientID) FOREIGN KEY
visitType	VARCHAR	38	

Step 5.2 Write and execute SQL statements to create all tables needed to implement the design.

```
---- Create the tables for the initial relational model
CREATE TABLE PATIENT (
patientID INT(3) NOT NULL,
firstName VARCHAR(38),
lastName VARCHAR(38),
initialVisitDate DATE,
email VARCHAR(38),
phone INT(10),
address VARCHAR(1000),
city VARCHAR(38),
state VARCHAR(2),
zipcode INT(5),
DOB DATE,
SSN INT(9),
currentMedicineTaken VARCHAR(1000),
CONSTRAINT patientID_pk PRIMARY KEY (patientID),
CONSTRAINT SSN_uk UNIQUE (SSN));

CREATE TABLE STAFF (
staffID INT(3) NOT NULL,
firstName VARCHAR(38),
LastName VARCHAR(38),
phone INT(10),
title VARCHAR(38),
taxID INT(10),
professional BOOL,
practice VARCHAR(38),
CONSTRAINT staffID_pk PRIMARY KEY (staffID),
CONSTRAINT taxID_uk UNIQUE (taxID));

CREATE TABLE APPOINTMENT (
appointmentID INT(3) NOT NULL,
appointmentDate DATE,
appointmentHoursFrom DATETIME,
appointmentHoursTo DATETIME,
appointmentType VARCHAR(38),
staffID INT(3),
patientID INT(3),
CONSTRAINT appointmentID_pk PRIMARY KEY (appointmentID),
CONSTRAINT staffID_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL);

CREATE TABLE Insurance (
insuranceMemberID INT(7) NOT NULL,
patientID INT(3),
insuranceCompany VARCHAR(38),
insurancePlan VARCHAR(38),
CONSTRAINT insuranceMemberID_pk PRIMARY KEY (insuranceMemberID),
CONSTRAINT patientID7_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL);

CREATE TABLE Bill (
billID INT(5) NOT NULL,
billDate DATE,
practitionerID INT(3),
patientID INT(3),
insuranceCoverageAmount DECIMAL(38, 2),
balanceDue DECIMAL(38, 2),
CONSTRAINT billID_pk PRIMARY KEY (billID),
CONSTRAINT practitionerID1_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID8_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE Payment (
paymentID INT(3) NOT NULL,
billID INT(5),
paymentDate DATE,
paymentAmount DECIMAL(38,2),
patientID INT(3),
CONSTRAINT paymentID_pk PRIMARY KEY (paymentID),
CONSTRAINT billID1_fk FOREIGN KEY (billID) REFERENCES BILL (billID) ON DELETE SET NULL,
CONSTRAINT patientID9_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE Prescription (
prescriptionID INT(3) NOT NULL,
prescriptionDate DATE,
practitionerID INT(3),
patientID INT(3),
medicineName VARCHAR(38),
CONSTRAINT prescriptionID_pk PRIMARY KEY (prescriptionID),
CONSTRAINT practitionerID2_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID10_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE Medicine (
RxNumber VARCHAR(10),
medicineName VARCHAR(38),
prescriptionID INT(3),
practitonerID INT(3),
patientID INT(3),
pharmacistID INT(3),
CONSTRAINT RxNumber_pk PRIMARY KEY (RxNumber),
CONSTRAINT prescriptionID1_fk FOREIGN KEY (prescriptionID) REFERENCES PRESCRIPTION (prescriptionID) ON DELETE SET NULL,
CONSTRAINT practitionerID11_fk FOREIGN KEY (practitonerID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID11_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT pharmacistID1_fk FOREIGN KEY (pharmacistID) REFERENCES STAFF (staffID) ON DELETE SET NULL);
```

```
CREATE TABLE TEST (
testID INT(38) NOT NULL,
testName VARCHAR(38),
patientID INT(3),
specimentType VARCHAR(38),
result VARCHAR(38),
description VARCHAR(1000),
performStaffID INT(3),
previewerStaffID INT(3),
CONSTRAINT testID_pk PRIMARY KEY (testID),
CONSTRAINT patientID1_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT performStaffID_fk FOREIGN KEY (performStaffID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT previewerStaffID_fk FOREIGN KEY (previewerStaffID) REFERENCES STAFF (staffID) ON DELETE SET NULL);


CREATE TABLE OperatingRoomSchedule (
surgeryID INT(3) NOT NULL,
surgeonID INT(3),
sugeryDate DATE,
patientID INT(3),
nurseID INT(3),
startTime DATETIME,
endTime DATETIME,
surgeryType VARCHAR(38),
anesthesia VARCHAR(38),
assistant VARCHAR(38),
CONSTRAINT surgeryID_pk PRIMARY KEY (surgeryID),
CONSTRAINT surgeonID_fk FOREIGN KEY (surgeonID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID3_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT nurseID_fk FOREIGN KEY (nurseID) REFERENCES STAFF (staffID) ON DELETE SET NULL);


CREATE TABLE DailyMasterSchedule (
appointmentID INT(3),
appointmentDate DATE,
practitionerID INT(3),
patientID INT(3),
appointmentHoursFrom DATETIME,
appointmentHoursTo DATETIME,
CONSTRAINT dailymasterschedule_pk PRIMARY KEY (appointmentID, appointmentDate),
CONSTRAINT appointmentID_fk FOREIGN KEY (appointmentID) REFERENCES APPOINTMENT (appointmentID) ON DELETE CASCADE,
CONSTRAINT practitionerID12_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID12_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE RecoveryRoomLog (
recoveryID INT(3) NOT NULL,
patientID INT(3),
practitionerID INT(3),
dateIn DATE,
timeIn DATETIME,
dateOut DATE,
timeOut DATETIME,
checkPractitionerID INT(3),
CONSTRAINT recoveryID_pk PRIMARY KEY (recoveryID),
CONSTRAINT patientID6_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT practitionerID_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT checkPractitionerID_fk FOREIGN KEY (checkPractitionerID) REFERENCES STAFF (staffID) ON DELETE SET NULL);


CREATE TABLE DailyDeliveryRoomLog (
deliveryID INT(5) NOT NULL,
deliveryDate DATE NOT NULL,
patientID INT(3),
pharmacistID INT(3),
midwifeID INT(3),
supervisorID INT(3),
CONSTRAINT deliveryID_Date_pk PRIMARY KEY (deliveryID, deliveryDate),
CONSTRAINT patientID5_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT pharmacistID_fk FOREIGN KEY (pharmacistID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT midwifeID_fk FOREIGN KEY (midwifeID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT supervisorID_fk FOREIGN KEY (supervisorID) REFERENCES STAFF (staffID) ON DELETE SET NULL);


CREATE TABLE OperatingRoomLog (
surgeryID INT(3),
surgeonID INT(3),
surgeryDate DATE,
patientID INT(3),
nurseID INT(3),
startTime DATETIME,
endTime DATETIME,
surgeryType VARCHAR(38),
CONSTRAINT surgeryID1_pk PRIMARY KEY (surgeryID),
CONSTRAINT surgeonID1_fk FOREIGN KEY (surgeonID) REFERENCES OperatingRoomSchedule (surgeonID) ON DELETE SET NULL,
CONSTRAINT patientID4_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL,
CONSTRAINT nurseID1_fk FOREIGN KEY (nurseID) REFERENCES STAFF (staffID) ON DELETE SET NULL);


CREATE TABLE LaboratoryLog (
testID INT(38),
staffID INT(3),
patientID INT(3),
CONSTRAINT testID_pk PRIMARY KEY (testID),
CONSTRAINT staffID1_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID2_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE IndividualPractitionerSchedule (
practitionerID INT(3),
scheduleDate DATE,
hoursFrom DATETIME,
hoursTo DATETIME,
appointmentType VARCHAR(38),
patientID INT(3),
visitType VARCHAR(38),
CONSTRAINT practitionerID_Date_pk PRIMARY KEY (practitionerID, scheduleDate),
CONSTRAINT practitionerID13_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE CASCADE,
CONSTRAINT patientID13_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL);


CREATE TABLE WeeklyCoverageSchedule (
staffID INT(3),
scheduleDate DATE,
hoursFrom DATETIME,
hoursTo DATETIME,
CONSTRAINT staffID_Date_pk PRIMARY KEY (staffID, scheduleDate),
CONSTRAINT staffID2_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE CASCADE);
```

Step 5.3 Create indexes for foreign keys and for any other columns as needed.

```
CREATE UNIQUE INDEX PATIENT_SSN ON PATIENT (SSN);
CREATE UNIQUE INDEX PATINET_ID ON PATIENT (PATIENTID);


CREATE UNIQUE INDEX STAFF_ID ON STAFF (STAFF_ID);
CREATE UNIQUE INDEX TAXID ON STAFF (TAXID);
CREATE UNIQUE INDEX APPOINTMENTID ON APPOINTMENT (APPOINTMENTID);
CREATE UNIQUE INDEX INSURANCEMEMBERID ON INSURANCE (INSURANCEMEMBERID);


CREATE UNIQUE INDEX BILLID ON BILL (BILLID);
CREATE UNIQUE INDEX PAYMENTID ON PAYMENT (PAYMENTID);
CREATE UNIQUE INDEX PRESCRIPTIONID ON PRESCRIPTION (PRESCRIPTIONID);
CREATE UNIQUE INDEX RxNumber ON MEDICINE (RXNUMBER);
CREATE UNIQUE INDEX TESTID ON TEST (TESTID);
CREATE UNIQUE INDEX SurgeryID ON OperatingRoomSchedule (SurgeryID);


CREATE UNIQUE INDEX APPOINTMENT_ID_DATE ON DailyMasterSchedule (AppointmentID, AppointmentDATE);
CREATE UNIQUE INDEX RECOVERY_ID ON RecoveryRoomLog (RecoveryID);


CREATE UNIQUE INDEX Delivery_ID_DATE ON DailyDeliveryRoomLog (DeliveryID, DeliveryDate);
CREATE UNIQUE INDEX Surgery_ID ON OperationRoomLog (SurgeryID);
CREATE UNIQUE INDEX testID ON LaboratoryLog (testID);


CREATE UNIQUE INDEX Practitioner_ID_DATE ON individualpractitionerschedule (practitionerID,
ScheduleDate);
CREATE UNIQUE INDEX Staff_ID_DATE ON weeklycoverageschedule (staffID, ScheduleDate);
```

Step 5.4 Insert at least five records in each table, preserving all constraints. Put in enough data to demonstrate how the database will function.

```
INSERT INTO PATIENT (patientID, firstName, lastName, initialVisitDate, email, phone, address, city,
state, zipcode, DOB, SSN, currentMedicineTaken)
VALUES
(1, 'John', 'Doe', '2022-01-01', 'johndoe@gmail.com', 123456789, '123 Main St', 'Anytown', 'CA',
12345, '2000-01-01', 123456789, 'Aspirin'),
(2, 'Jane', 'Doe', '2022-01-02', 'janedoe@gmail.com', 234567890, '456 Elm St', 'Othertown', 'NY',
23456, '1999-01-01', 234567890, 'Ibuprofen'),
(3, 'Bob', 'Smith', '2022-01-03', 'bobsmith@gmail.com', 345678901, '789 Oak St', 'Somewhere', 'TX',
34567, '1980-01-01', 345678901, 'Paracetamol'),
(4, 'Mary', 'Johnson', '2022-01-04', 'maryjohnson@gmail.com', 456789012, '321 Pine St', 'Nowhere',
'FL', 45678, '1975-01-01', 456789012, 'Acetaminophen'),
(5, 'Tom', 'Brown', '2022-01-05', 'tombrown@gmail.com', 567890124, '654 Maple St', 'Everywhere',
'WA', 56789, '1985-01-01', 567890123, 'None');


INSERT INTO STAFF (staffID, firstName, lastName, phone, title, taxID, professional, practice)
VALUES
(1, 'Dr.', 'Johnson', 123456780, 'Doctor', 123456789, true, 'General Practice'),
(2, 'Nurse', 'Smith', 234567801, 'Nurse Practitioner', 234567890, true, 'Pediatrics'),
(3, 'Dr.', 'Lee', 345678901, 'Surgeon', 345678901, true, 'Cardiology'),
(4, 'Receptionist', 'Davis', 467890123, 'Receptionist', 456789012, false, 'General Practice'),
(5, 'Pharmacist', 'Brown', 568901234, 'Pharmacist', 567890123, true, 'Pharmacy');
```

```
INSERT INTO APPOINTMENT (appointmentID, appointmentDate, appointmentHoursFrom, appointmentHoursTo,
appointmentType, staffID, patientID)
VALUES
(1, '2022-01-01', '2022-01-01 09:00:00', '2022-01-01 09:30:00', 'Check-up', 1, 1),
(2, '2022-01-02', '2022-01-02 10:00:00', '2022-01-02 10:30:00', 'Follow-up', 2, 2),
(3, '2022-01-03', '2022-01-03 11:00:00', '2022-01-03 11:30:00', 'Check_up', 3, 3),
(4, '2022-01-04', '2022-01-04 12:00:00', '2022-01-04 12:30:00', 'Check_up', 4, 4),
(5, '2022-01-05', '2022-01-05 13:00:00', '2022-01-05 13:30:00', 'Follow-up', 5, 5);


INSERT INTO TEST (testID, testName, patientID, specimentType, result, description, performStaffID,
previewerStaffID)
VALUES
(1, 'Blood Test', 1, 'Blood', 'Normal', 'Complete Blood Count', 1, 2),
(2, 'Urine Test', 2, 'Urine', 'Normal', 'Urine analysis', 2, 3),
(3, 'MRI', 3, 'Body Tissue', 'Abnormal', 'Magnetic Resonance Imaging', 3, 4),
(4, 'CT Scan', 4, 'X-ray', 'Normal', 'Computed Tomography Scan', 4, 5),
(5, 'Ultrasound', 5, 'Body Tissue', 'Abnormal', 'Abdominal Ultrasound', 1, 2);


INSERT INTO LaboratoryLog (testID, staffID, patientID)
VALUES (1, 2, 3), (2, 1, 4), (3, 3, 5), (4, 2, 1), (5, 1, 2);


INSERT INTO OperatingRoomSchedule (surgeryID, surgeonID, sugeryDate, patientID, nurseID, startTime,
endTime, surgeryType, anesthesia, assistant)
VALUES
(1, 1, '2023-04-10', 1, 2, '2023-04-10 10:00:00', '2023-04-10 12:00:00', 'Heart Surgery', 'General
Anesthesia', 'John'),
(2, 2, '2023-04-11', 2, 3, '2023-04-11 11:00:00', '2023-04-11 13:00:00', 'Brain Surgery', 'Local
Anesthesia', 'Mary'),
(3, 3, '2023-04-12', 3, 4, '2023-04-12 09:00:00', '2023-04-12 11:00:00', 'Knee Surgery', 'Regional
Anesthesia', 'Bob'),
(4, 4, '2023-04-13', 4, 5, '2023-04-13 12:00:00', '2023-04-13 14:00:00', 'Eye Surgery', 'Local
Anesthesia', 'Kate'),
(5, 5, '2023-04-14', 5, 1, '2023-04-14 13:00:00', '2023-04-14 15:00:00', 'Lung Surgery', 'General
Anesthesia', 'Chris');


INSERT INTO OperatingRoomLog (surgeryID, surgeonID, surgeryDate, patientID, nurseID, startTime,
endTime, surgeryType)
VALUES
(1, 1, '2023-04-10', 1, 2, '2023-04-10 08:00:00', '2023-04-10 09:00:00', 'Appendectomy'),
(2, 2, '2023-04-10', 2, 3, '2023-04-10 10:00:00', '2023-04-10 11:30:00', 'Knee Replacement'),
(3, 3, '2023-04-11', 3, 4, '2023-04-11 08:30:00', '2023-04-11 09:45:00', 'Hernia Repair'),
(4, 4, '2023-04-12', 4, 5, '2023-04-12 11:00:00', '2023-04-12 12:30:00', 'Cataract Surgery'),
(5, 5, '2023-04-12', 5, 1, '2023-04-12 13:00:00', '2023-04-12 14:00:00', 'Laser Eye Surgery');


INSERT INTO DailyDeliveryRoomLog (deliveryID, deliveryDate, patientID, pharmacistID, midwifeID,
supervisorID)
VALUES
```

```
(1, '2023-04-09', 1, 1, 2, 3),
(2, '2023-04-08', 2, 2, 3, 4),
(3, '2023-04-07', 3, 3, 4, 5),
(4, '2023-04-06', 4, 4, 5, 1),
(5, '2023-04-05', 5, 5, 1, 2);

INSERT INTO RecoveryRoomLog (recoveryID, patientID, practitionerID, dateIn, timeIn, dateOut,
timeOut, checkPractitionerID)
VALUES
(101, 1, 1, '2023-04-09', '2023-04-09 12:00:00', '2023-04-09', '2023-04-09 15:30:00', 2),
(102, 2, 2, '2023-04-10', '2023-04-10 09:00:00', '2023-04-10', '2023-04-10 13:00:00', 3),
(103, 3, 3, '2023-04-11', '2023-04-11 14:00:00', '2023-04-11', '2023-04-11 17:00:00', 4),
(104, 4, 4, '2023-04-12', '2023-04-12 11:30:00', '2023-04-12', '2023-04-12 14:45:00', 5),
(105, 5, 5, '2023-04-13', '2023-04-13 08:45:00', '2023-04-13', '2023-04-13 12:00:00', 1);

INSERT INTO Insurance (insuranceMemberID, patientID, insuranceCompany, insurancePlan)
VALUES
(1234567, 1, 'Blue Cross Blue Shield', 'Standard Plan'),
(2345678, 2, 'Aetna', 'Gold Plan'),
(3456789, 3, 'Cigna', 'Platinum Plan'),
(4567890, 4, 'UnitedHealthcare', 'Basic Plan'),
(5678901, 5, 'Humana', 'Premium Plan');

INSERT INTO Bill (billID, billDate, practitionerID, patientID, insuranceCoverageAmount, balanceDue)
VALUES
(1, '2022-03-15', 1, 1, 500.00, 1000.00),
(2, '2022-03-20', 2, 2, 750.00, 1500.00),
(3, '2022-04-02', 3, 3, 250.00, 750.00),
(4, '2022-04-10', 4, 4, 1000.00, 2000.00),
(5, '2022-04-12', 5, 5, 500.00, 1000.00);

INSERT INTO Payment(paymentID, billID, paymentDate, paymentAmount, patientID)
VALUES
(1, 1, '2022-03-15', 100.00, 1),
(2, 2, '2022-04-10', 50.00, 2),
(3, 3, '2022-05-05', 75.00, 3),
(4, 4, '2022-06-20', 25.00, 4),
(5, 5, '2022-07-30', 200.00, 5);

INSERT INTO Prescription (prescriptionID, prescriptionDate, practitionerID, patientID, medicineName)
VALUES
(1, '2023-04-01', 1, 1, 'Aspirin'),
(2, '2023-04-02', 2, 2, 'Ibuprofen'),
(3, '2023-04-03', 3, 3, 'Acetaminophen'),
(4, '2023-04-04', 4, 4, 'Penicillin'),
(5, '2023-04-05', 5, 5, 'Amoxicillin');

INSERT INTO Medicine (RxNumber, medicineName, prescriptionID, practitonerID, patientID,
pharmacistID) VALUES
('RX12345', 'Ibuprofen', 1, 1, 1, 2),
('RX23456', 'Aspirin', 2, 2, 2, 3),
('RX34567', 'Acetaminophen', 3, 3, 3, 4),
('RX45678', 'Amoxicillin', 4, 4, 4, 5),
('RX56789', 'Lisinopril', 5, 5, 5, 1);

INSERT INTO WeeklyCoverageSchedule (staffID, scheduleDate, hoursFrom, hoursTo) VALUES
(1, '2023-04-10', '08:00:00', '16:00:00'),
(2, '2023-04-10', '09:00:00', '17:00:00'),
(3, '2023-04-11', '12:00:00', '20:00:00'),
(4, '2023-04-11', '10:00:00', '18:00:00'),
(5, '2023-04-12', '11:00:00', '19:00:00');

INSERT INTO DailyMasterSchedule (appointmentID, practitionerID, patientID) VALUES
(1, 1, 1),
(2, 2, 2),
(3, 3, 3),
(4, 4, 4),
(5, 5, 5);

INSERT INTO IndividualPractitionerSchedule (practitionerID, scheduleDate, hoursFrom, hoursTo,
appointmentType, patientID, visitType)
VALUES
(1, '2023-04-10', '2023-04-10 09:00:00', '2023-04-10 10:00:00', 'Follow-up', 1, 'In-person'),
(2, '2023-04-11', '2023-04-11 14:00:00', '2023-04-11 16:00:00', 'New patient', 2, 'Virtual'),
(3, '2023-04-12', '2023-04-12 11:00:00', '2023-04-12 12:30:00', 'Follow-up', 3, 'In-person'),
(4, '2023-04-13', '2023-04-13 13:30:00', '2023-04-13 14:30:00', 'New patient', 4, 'Virtual'),
(5, '2023-04-14', '2023-04-14 15:00:00', '2023-04-14 16:00:00', 'Follow-up', 5, 'In-person');
```

Step 5.5 Write SQL statements that will process five non-routine requests for information from the database just created. For each, write the request in English, followed by the corresponding SQL command.

- Retrieve the appointment details of a particular patient with patient ID 1
  ```
  SELECT * FROM APPOINTMENT WHERE patientID = 1;
  ```

- Retrieve the list of patients who have undergone surgery:
  ```
  SELECT DISTINCT P.firstName, P.lastName
  FROM PATIENT P
  ```

```
JOIN OperatingRoomSchedule OS ON P.patientID = OS.patientID;
```

- Retrieve the list of staff members who have performed a test on a particular patient with patient ID 2:
  ```
  SELECT S.firstName, S.LastName
  FROM STAFF S
  JOIN TEST T ON S.staffID = T.performStaffID
  WHERE T.patientID = 2;
  ```

- Retrieve the list of patients who have not undergone any surgery:
  ```
  SELECT P.firstName, P.lastName
  FROM PATIENT P
  LEFT JOIN OperatingRoomSchedule OS ON P.patientID = OS.patientID
  WHERE OS.surgeryID IS NULL;
  ```

- Retrieve the total number of appointments for each staff member:
  ```
  SELECT S.staffID, S.firstName, S.LastName, COUNT(*) AS appointment_count
  FROM STAFF S
  JOIN APPOINTMENT A ON S.staffID = A.staffID
  GROUP BY S.staffID, S.firstName, S.LastName;
  ```

Step 5.6 Create at least one trigger and write the code for it.

```
CREATE TRIGGER payment_greater_than_billing
AFTER UPDATE ON PAYMENT
FOR EACH ROW
BEGIN
  IF NEW.amount > (SELECT amount FROM BILLING WHERE billingID =
NEW.billingID) THEN
    UPDATE BILLING SET paid_in_full = true WHERE billingID = NEW.billingID;
  END IF;
END;
```

Step 6.1 Begin with the list of the tables that the entities and relationships from the ER diagram mapped to naturally, from the sample project section at the end of chapter 4. For each table on the list, identify functional dependencies and normalize the relation to BCNF. Then decide whether the resulting tables should be implemented in that form. If not, explain why.

The following tables resulted from the mapping:

(1)Patient(patientID, lastName, firstName, InitialVisitDate, email, phone,address, city, state, zipcode, DOB, SSN, currentMedication)

For Patient, we have the following FDs

patientID -> lastName, firstName, InitialVisitDate, email, phone, address, city, state, zipcode, DOB, SSN, currentMedication

Based on the given assumptions, each patientID is unique, and patientID is the primary key. And there are no multivalued dependencies or non-prime attributes dependent on non-candidate key attributes. Since we are running a medical clinic, we have to keep all the patient's PII information on file such as last name, first name, DOB, SSN, and address. And since SSN is long, it cannot be substituted for patientID. Thus, it meets the criteria of 2NF, 3NF, and BCNF even though there are some transititive dependency among the attributes. Thus, Patient table is already normalized and there is no need to modify the table.

(2)Appointment(AppointmentID, AppointmentDate, AppointmentHoursFrom, AppointmentHoursTo, AppointmentType, StaffID, PatientID)

For Appointment, we have the following FDs

AppointmentID -> AppointmentDate, AppointmentHoursFrom, AppointmentHoursTo, AppointmentType, StaffID, PatientID

StaffID -> AppointmentDate, AppointmentHoursFrom, AppointmentHoursTo, AppointmentType, PatientID

Based on the given assumptions, each AppointmentID is unique, and AppointmentID is the primary key. Each StaffID and PatientID may correspond to multiple Appointments, and therefore are not unique. There are no multivalued dependencies or non-prime attributes dependent on non-candidate key attributes. Thus, we must have appointmentID as primary key for Appointment table. Also, the staffID and patientID are included by Appointment table to map to Staff table and Patient table to get their personal information. Thus, Appointment table is already normalized and there is no need to modify the table.

(3)Staff(staffID, firstName, lastName, Phone, title, taxID, professional, practice)

For Staff, we have the following FDs

staffID -> firstName, lastName, Phone, title, taxID, professional, practice

To normalize the relation to Boyce-Codd Normal Form (BCNF), we need to identify all determinants (i.e., left-hand side of the FDs) that are not candidate keys, and decompose the relation accordingly. In this case, the only determinant is staffID, which is also the primary key, and there are no non-prime attributes that depend on non-candidate key attributes. Therefore, the "Staff" relation is already in BCNF, as there are no FDs with determinants that are not candidate keys.

(4) Test(testID, testName, DoctorID, patientID, specimenType, result, description, performStaffID, previewerStaffID)

For Test, we have the following FDs

{testID} → {testName}, {DoctorID}, {patientID}, {specimenType}, {result}, {description}, {performStaffID}, {previewerStaffID}

We assume that a test can only have one result, and a test can be performed by multiple staff members, but only one staff member can perform it at a time.

1NF: Each attribute in each tuple of the relation contains only atomic, this relation satisfies 1NF. 2NF: Since all attributes depend only on the primary key (testID), this relation is in 2NF. 3NF: There are no transitive dependencies, so the relation is in 3NF. BCNF: The relation is in 3NF, and for every non-trivial functional dependency (X → Y), X is a superkey. Here, all functional dependencies have the primary key (testID) on the left-hand side, so the relation is in BCNF.

(5)LaboratoryLog(testID, StaffID, PatientID)

For LaboratoryLog, we have the following FDs

{testID} → {StaffID, PatientID}.

It satisfy 1NF, since there would be no repeating groups or multiple values for a single attribute.

For 2NF, since testID is the primary key, we don't have to worry about partial dependencies, and the relation automatically satisfies 2NF.

For 3NF, since there are no other non-key attributes besides StaffID and PatientID, there are no transitive dependencies, and the relation automatically satisfies 3NF.

For BCNF, since testID is the primary key, any functional dependency in the relation would have a candidate key as the determinant, and the relation automatically satisfies BCNF.

(6)OperatingRoomSchedule(surgeryId, surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType, anesthesia, assistant)

For OperatingRoomSchedule, we have the following FDs

{surgeryId} → {surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType, anesthesia, assistant}

We assume that Each surgery has a unique surgeryId, and each surgeon, patient, and nurse can be involved in multiple surgeries. Each surgery will have one patient, and a patient can have multiple surgeries. The table satisfies 1NF, 2NF, 3NF, and BCNF as there are no transitive dependencies and all functional dependencies have candidate keys as determinants.

(7)OperatingRoomLog(surgeryId, surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType)

For OperatingRoomLog, we have the following FDs

{surgeryId} → {surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType}

The relation is normalized, since each column contains atomic values, and all non-key attributes are fully functionally dependent on surgeryId, there is no transitive dependencies, and have non-trivial functional dependencies, so it satisfies 1NF, 2NF,3NF, and BCNF.

(8)DailyDeliveryRoomLog(deliveryId, deliveryDate, patientId, pharmacistId, midWifeId, supervisorId)

For DailyDeliveryRoomLog, we have the following FDs:

{deliveryId} → {deliveryDate, patientId, pharmacistId, midWifeId, supervisorId}

This means that the deliveryId uniquely determines the deliveryDate, patientId, pharmacistId, midWifeId, and supervisorId. There are no other functional dependencies present in the relation.

In terms of normalization, the relation satisfies 1NF as there are no repeating groups or nested values present in any attribute. It also satisfies 2NF as there is only one candidate key (deliveryId) and all non-key attributes depend fully on it. The relation also satisfies 3NF and BCNF as there are no transitive dependencies or non-trivial functional dependencies present in the relation where the determinant is not a candidate key.

(9)RecoveryRoomLog(recoveryID, patientID, practitionerId, dateIn, timeIn, dateOut, timeOut, checkPractitionerId)

For RecoveryRoomLog, we have the following FDs

{recoveryID} → {patientID}, {practitionerID}, {dateIn}, {timeIn}, {dateOut}, {timeOut}, {checkPractitionerId}

Since there is only one candidate key (recoveryID}, all the attributes on the right-hand side (patientID, practitionerID, dateIn, timeIn, dateOut, timeOut, and checkPractitionerId) are functionally dependent on the left-hand side attribute (recoveryID). Therefore, the relation satisfies 2NF, 3NF, and BCNF.

(10)Insurance(InsuranceMemberID, patientID, InsuranceCompany, InsurancePlan)

For Insurance, we have the following FDs

{InsuranceMemberID} → {patientID, InsuranceCompany, InsurancePlan}

This means that given an InsuranceMemberID, we can determine the corresponding patientID, InsuranceCompany, and InsurancePlan. Since InsuranceMemberID is the primary key, this relation satisfies the requirements of 1NF, 2NF, 3NF, and BCNF.

(11)Bill(billID, billDate, practitionerID, patientID, InsuranceCoverageAmount, BalanceDue)

For Bill, we have the following FDs

{billID} -> {billDate}, {practitionerID}, {patientID},{InsuranceCoverageAmount}, {BalanceDue}

We assume that each patient has multiple bills and each bill is associated with one patient. Since the relation has atomic value, there is no transitive dependencies, and non-trivial functional dependencies, so it satisfies 1NF, 2NF, 3NF, BCNF

(12)Prescription(PrescriptionID, PrescriptionDate, practitionerID, patientID, medicineName)

For medicine, we have the FD:

{PrescriptionID} → {PrescriptionDate, practitionerID, patientID, medicineName}

The relation is satisfied 1NF, 2NF, 3NF, BCNF, since each attribute has atomic values, there is only one non-prime attribute, and no transitive dependency in the relation, non-trivial functional dependencies. But based on (13) we change schema to: Prescription(prescriptionID, prescriptionDate, practitionerID, patientID, RxNumber)

(13)Medicine(RxNumber, MedicineName, prescriptionID, practitionerID, PharmacistID)

For medicine, we have the FD:

{RxNumber} → {MedicineName}, {prescriptionID}, {practitionerID}, {PharmacistID}

{prescriptionID} → {MedicineName}, {practitionerID}

Based on these functional dependencies, the medical table is in 3NF. To make the relation BCNF, we need to ensure that every determinant is a candidate key, which means the only candidate key should be {RaxNumber}, we can split the table into two tables:

1. Medicine(RxNumber, MedicineName) with RxNumber as the primary key.

2. Prescription(prescriptionID, practitionerID, PharmacistID, RxNumber) with prescriptionID as the primary key and RxNumber as a foreign key referencing the Medicine table.

Since we already have Prescription(PrescriptionID, PrescriptionDate, practitionerID, patientID, medicineName), we just change it to Prescription(prescriptionID, practitionerID, PharmacistID, patientID, RxNumber), and for Medicine, now the schema is Medicine(RxNumber, MedicineName), which satisfied BCNF.

(14)Payment(PaymentID, BillID, PaymentDate, PaymentAmount, PatientID)

For Payment, we have the FD:

{PaymentID} → {BillID}, {PaymentDate}, {PaymentAmount}, {PatientID}

{BillID} → {PatientID}

Based on these functional dependencies, we can say that the Payment relation is in 2NF, since all non-key attributes (PaymentDate, PaymentAmount, and PatientID) are fully functionally dependent on the primary key (PaymentID) and there are no partial dependencies. However, the Payment relation is not in 3NF, since there is a transitive dependency between BillID and PatientID. To remove this transitive dependency, we decompose the relation, since we have Bill(billID, billDate, practitionerID, patientID, InsuranceCoverageAmount, BalanceDue), we just delete attribute patientID.

After deleting patientID, the resulting schema would be Payment(PaymentID, BillID, PaymentDate, PaymentAmount) here. The schema now satisfies the requirements of 1NF, 2NF, 3NF, and BCNF.

(15)DailyMasterSchedule(appointmentId, appointmentDate, appointmentHoursFrom, appointmentHoursTo, practitionerId, patientId)

For DailyMasterSchedule, we have the FD:

{appointmentId, appointmentDate} → {appointmentHoursFrom}, {appointmentHoursTo}, {practitionerId},{patientId}

We assume that the appointment IDs are reset to 0 at the beginning of each day, each appointment is scheduled for a specific date (appointmentDate), and there can be multiple appointments scheduled for the same date. Therefore, the combination of appointmentId and appointmentDate forms a composite primary key that uniquely identifies each appointment in the DailyMasterSchedule relation.

The relation satisfies the requirements of 1NF since all attributes are atomic. Since all non-key attributes are fully dependent on the only candidate key (appointmentID, appointmentDate), It satisfies the requirements of 2NF. There are no non-key attributes in the relation, so there cannot be an transitive dependencies, and satisfies the requirement of 3NF and also for BCNF.

(16) WeeklyCoverageSchedule(staffID, scheduleDate, hoursFrom, hoursTo, Professional, Phone)

For WeeklyCoverageSchedule, we have the FD:

{staffID, scheduleDate} →{ hoursFrom}, { hoursTo}, { Professional}, { Phone}

In the table, we assume each schedule date is unique for each staff member, and the hoursFrom and hoursTo attributes represent the start and end times of a staff member's work shift on the specified schedule date.

Based on the assumptions, we can conclude that the table is normalized. All attributes in the relation is atomic, the relation is in 1NF. All non-key attributes hours from, hours to, professional, and phone are fully dependent on the candidate key staffID and scheduleDate. Since the only functional dependency is staffID, scheduleDate → hoursFrom, hoursTo, Professional, Phone. There is no transitive dependencies, to the relation also satisfies 3NF and BCNF.

(17) IndividualPractionerSchedule (practitionerID,ScheduleDate, HoursFrom, HoursTo, AppointmentType, PatientID, VisitType)

For the IndividualPractionerSchedule, we have the FD:

{practitionerID, ScheduleDate} -> {HoursFrom}, {HoursTo}, {AppointmentType}, {PatientID}, {VisitType}

In the table, we assume that each row in the table represents a unique combination of a practitioner's schedule for a specific date, HoursFrom, HoursTo, AppointmentType, PatientID, and VisitType, are fully dependent on this candidate key.

Based on the attributes in the IndividualPractionerSchedule relation, it is possible to have non-trivial functional dependencies where the determinant is not the candidate key. For example, there could be a functional dependency where AppointmentType determines VisitType. However, this is not a violation of BCNF since we assume that AppointmentType and VisitType always have a one-to-one correspondence, then the relation is still in BCNF.

1NF: it has a primary key and each attribute contains only atomic values. 2NF: there is only one candidate key, and all non-key attributes are fully dependent on it. There are no partial dependencies. 3NF: there are no transitive dependencies. Each non-key attribute is dependent only on the candidate key. BCNF: the functional dependency {practitionerID, ScheduleDate} -> HoursFrom, HoursTo, AppointmentType, PatientID, VisitType is a trivial dependency on the candidate key {practitionerID, ScheduleDate}. To summarize, the IndividualPractionerSchedule relation is already normalized.

Step 6.2 Update the data dictionary and list of assumptions as needed.

The data dictionary is based on the forms and reports being filled and the list of assumptions are based on the entity relationships - their cardinality, thus the data dictionary and list of assumptions remain unchanged.

Step 6.3 For each table, write the table name and write out the names, data types, and sizes of all the data items, identify any constraints, using the conventions of the DBMS you will use for implementation.

Table Patient

patientID	INT	3	(patientID) PRIMARY KEY NOT NULL
firstName	VARCHAR	38	
lastName	VARCHAR	38	
initialVisitDate	DATE		
email	VARCHAR	38	
phone	INT	10	
address	VARCHAR	1000	
city	VARCHAR	38	
state	VARCHAR	2	
zipcode	INT	5	
DOB	DATE		
SSN	INT	9	UNIQUE
currentMedicineTaken	VARCHAR	1000	

Table Staff

staffID	INT	3	(staffID) PRIMARY KEY NOT NULL
firstName	VARCHAR	38	
LastName	VARCHAR	38	
phone	INT	10	
title	VARCHAR	38	
taxID	INT	10	UNIQUE
professional	BOOL		
practice	VARCHAR	38	

Table Appointment

appointmentID	INT	3	(appointmentID) PRIMARY KEY NOT NULL
appointmentDate	DATE		
appointmentHoursFrom	DATETIME		
appointmentHoursTo	DATETIME		

appointmentType	VARCHAR	38	
staffID	INT	3	(staffID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table Test

testID	INT	38	(testID) PRIMARY KEY NOT NULL
testName	VARCHAR	38	
patientID	INT	3	(patientID) FOREIGN KEY
specimentType	VARCHAR	38	
result	VARCHAR	38	
description	VARCHAR	1000	
performStaffID	INT	3	(performStaffID) FOREIGN KEY
previewerStaffID	INT	3	(previewerStaffID) FOREIGN KEY

Table LaboratoryLog

testID	INT	38	(testID) FOREIGN KEY
staffID	INT	3	(staffID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table OperatingRoomSchedule

surgeryID	INT	3	(surgeryID) PRIMARY KEY NOT NULL
surgeonID	INT	3	(surgeonID) FOREIGN KEY
sugeryDate	DATE		
patientID	INT	3	(patientID) FOREIGN KEY
nurseID	INT	3	(nurseID) FOREIGN KEY
startTime	DATETIME		
endTime	DATETIME		
surgeryType	VARCHAR	38	
anesthesia	VARCHAR	38	
assistant	VARCHAR	38	

Table OperatingRoomLog

surgeryID	INT	3	(surgeryID) FOREIGN KEY
surgeonID	INT	3	(surgeonID) FOREIGN KEY
surgeryDate	DATE		
patientID	INT	3	(patientID) FOREIGN KEY
nurseID	INT	3	(nurseID) FOREIGN KEY
startTime	DATETIME		
endTime	DATETIME		
surgeryType	VARCHAR	38	

Table DailyDeliveryRoomLog

deliveryID	INT	5	(deliveryID, deliveryDate) PRIMARY KEY NOT NULL
deliveryDate	DATE		(deliveryID, deliveryDate) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY

pharmacistID	INT	3	(pharmacistID) FOREIGN KEY
midwifeID	INT	3	(midwifeID) FOREIGN KEY
supervisorID	INT	3	(supervisorID) FOREIGN KEY

Table RecoveryRoomLog

recoveryID	INT	3	(recoveryID) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY
practitionerID	INT	3	(practitionerID) FOREIGN KEY
dateIn	DATE		
timeIn	DATETIME		
dateOut	DATE		
timeOut	DATETIME		
checkPractitionerID	INT	3	(checkPractitionerID) FOREIGN KEY

Table Insurance

insuranceMemberID	INT	7	(insuranceMemberID) PRIMARY KEY NOT NULL
patientID	INT	3	(patientID) FOREIGN KEY
insuranceCompany	VARCHAR	38	
insurancePlan	VARCHAR	38	

Table Bill

billID	INT	5	(billID) PRIMARY KEY NOT NULL
billDate	DATE		
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY
insuranceCoverageAmount	DECIMAL	38,2	
balanceDue	DECIMAL	38,2	

Table Payment

paymentID	INT	3	(paymentID) PRIMARY KEY NOT NULL
billID	INT	5	(billID) FOREIGN KEY
paymentDate	DATE		
paymentAmount	DECIMAL	38,2	

Table Prescription

prescriptionID	INT	3	(prescriptionID) PRIMARY KEY NOT NULL
prescriptionDate	DATE		
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY
RxNumber	VARCHAR	38	

Table Medicine

RxNumber	VARCHAR	10	(RxNumber) PRIMARY KEY NOT NULL
medicineName	VARCHAR	38	

Table DailyMasterSchedule

appointmentID	INT	3	(appointmentID) FOREIGN KEY
practitionerID	INT	3	(practitionerID) FOREIGN KEY
patientID	INT	3	(patientID) FOREIGN KEY

Table WeeklyCoverageSchedule

staffID	INT	3	(staffID, scheduleDate) PRIMARY KEY NOT NULL
scheduleDate	DATE		(staffID, scheduleDate) PRIMARY KEY NOT NULL
hoursFrom	DATETIME		
hoursTo	DATETIME		

Table IndividualPractitionerSchedule

practitionerID	INT	3	(practitionerID, scheduleDate) PRIMARY KEY NOT NULL
scheduleDate	DATE		(practitionerID, scheduleDate) PRIMARY KEY NOT NULL
hoursFrom	DATETIME		
hoursTo	DATETIME		
appointmentType	VARCHAR	38	
patientID	INT	3	(patientID) FOREIGN KEY
visitType	VARCHAR	38	

Step 6.4 Write and execute SQL statements to create all the tables needed to implement the design.

```
CREATE TABLE PATIENT
(
patientID INT(3) NOT NULL,
firstName VARCHAR(38),
lastName VARCHAR(38),
initialVisitDate DATE,
email VARCHAR(38),
phone INT(10),
address VARCHAR(1000),
city VARCHAR(38),
state VARCHAR(2),
zipcode INT(5),
DOB DATE,
SSN INT(9),
currentMedicineTaken VARCHAR(1000),
CONSTRAINT patientID_pk PRIMARY KEY (patientID),
CONSTRAINT SSN_uk UNIQUE (SSN)
);

 CREATE TABLE STAFF
(
staffID INT(3) NOT NULL,
firstName VARCHAR(38),
LastName VARCHAR(38),
phone INT(10),
title VARCHAR(38),
taxID INT(10),
professional BOOL,
practice VARCHAR(38),
CONSTRAINT staffID_pk PRIMARY KEY (staffID),
CONSTRAINT taxID_uk UNIQUE (taxID)
);
```

```
CREATE TABLE APPOINTMENT
(
appointmentID INT(3) NOT NULL,
appointmentDate DATE,
appointmentHoursFrom DATETIME,
appointmentHoursTo DATETIME,
appointmentType VARCHAR(38),
staffID INT(3),
patientID INT(3),
CONSTRAINT appointmentID_pk PRIMARY KEY (appointmentID),
CONSTRAINT staffID_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET NULL
);

CREATE TABLE TEST
(
testID INT(38) NOT NULL,
testName VARCHAR(38),
patientID INT(3),
specimentType VARCHAR(38),
result VARCHAR(38),
description VARCHAR(1000),
performStaffID INT(3),
previewerStaffID INT(3),
CONSTRAINT testID_pk PRIMARY KEY (testID),
CONSTRAINT patientID1_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT performStaffID_fk FOREIGN KEY (performStaffID) REFERENCES STAFF (staffID) ON DELETE
SET NULL,
CONSTRAINT previewerStaffID_fk FOREIGN KEY (previewerStaffID) REFERENCES STAFF (staffID) ON
DELETE SET NULL
);

CREATE TABLE LaboratoryLog
(
testID INT(38),
staffID INT(3),
patientID INT(3),
CONSTRAINT testID_fk FOREIGN KEY (testID) REFERENCES TEST (testID) ON DELETE CASCADE,
CONSTRAINT staffID1_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID2_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL
);



CREATE TABLE OperatingRoomSchedule
(
surgeryID INT(3) NOT NULL,
surgeonID INT(3),
sugeryDate DATE,
patientID INT(3),
nurseID INT(3),
startTime DATETIME,
endTime DATETIME,
surgeryType VARCHAR(38),
anesthesia VARCHAR(38),
assistant VARCHAR(38),
CONSTRAINT surgeryID_pk PRIMARY KEY (surgeryID),
CONSTRAINT surgeonID_fk FOREIGN KEY (surgeonID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT patientID3_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT nurseID_fk FOREIGN KEY (nurseID) REFERENCES STAFF (staffID) ON DELETE SET NULL
);

CREATE TABLE OperatingRoomLog
(
surgeryID INT(3),
surgeonID INT(3),
surgeryDate DATE,
patientID INT(3),
nurseID INT(3),
startTime DATETIME,
endTime DATETIME,
surgeryType VARCHAR(38),
CONSTRAINT surgeryID1_pk FOREIGN KEY (surgeryID) REFERENCES OperatingRoomSchedule (surgeryID) ON
DELETE CASCADE,
CONSTRAINT surgeonID1_pk FOREIGN KEY (surgeonID) REFERENCES OperatingRoomSchedule (surgeonID) ON
DELETE SET NULL,
CONSTRAINT patientID4_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT nurseID1_fk FOREIGN KEY (nurseID) REFERENCES STAFF (staffID) ON DELETE SET NULL
);

CREATE TABLE DailyDeliveryRoomLog
(
deliveryID INT(5) NOT NULL,
deliveryDate DATE NOT NULL,
patientID INT(3),
pharmacistID INT(3),
midwifeID INT(3),
supervisorID INT(3),
CONSTRAINT deliveryID_Date_pk PRIMARY KEY (deliveryID, deliveryDate),
CONSTRAINT patientID5_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT pharmacistID_fk FOREIGN KEY (pharmacistID) REFERENCES STAFF (staffID) ON DELETE SET
NULL,
CONSTRAINT midwifeID_fk FOREIGN KEY (midwifeID) REFERENCES STAFF (staffID) ON DELETE SET NULL,
CONSTRAINT supervisorID_fk FOREIGN KEY (supervisorID) REFERENCES STAFF (staffID) ON DELETE SET
NULL
);

CREATE TABLE RecoveryRoomLog
(
recoveryID INT(3) NOT NULL,
patientID INT(3),
practitionerID INT(3),
dateIn DATE,
timeIn DATETIME,
dateOut DATE,
timeOut DATETIME,
checkPractitionerID INT(3),
CONSTRAINT recoveryID_pk PRIMARY KEY (recoveryID),
CONSTRAINT patientID6_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT practitionerID_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE
SET NULL,
CONSTRAINT checkPractitionerID_fk FOREIGN KEY (checkPractitionerID) REFERENCES STAFF (staffID) ON
DELETE SET NULL
);

CREATE TABLE Insurance
(
insuranceMemberID INT(7) NOT NULL,
patientID INT(3),
insuranceCompany VARCHAR(38),
insurancePlan VARCHAR(38),
CONSTRAINT insuranceMemberID_pk PRIMARY KEY (insuranceMemberID),
CONSTRAINT patientID7_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL
);

CREATE TABLE Bill
(
billID INT(5) NOT NULL,
billDate DATE,
practitionerID INT(3),
patientID INT(3),
insuranceCoverageAmount DECIMAL(38, 2),
balanceDue DECIMAL(38, 2),
CONSTRAINT billID_pk PRIMARY KEY (billID),
```

```sql
CONSTRAINT practitionerID1_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE
SET NULL,
CONSTRAINT patientID8_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL
);

CREATE TABLE Payment
(
paymentID INT(3) NOT NULL,
billID INT(5),
paymentDate DATE,
paymentAmount DECIMAL(38,2),
CONSTRAINT paymentID_pk PRIMARY KEY (paymentID),
CONSTRAINT billID1_fk FOREIGN KEY (billID) REFERENCES BILL (billID) ON DELETE SET NULL
);

CREATE TABLE Medicine
(
RxNumber VARCHAR(10) primary key,
medicineName VARCHAR(38)
);

CREATE TABLE Prescription
(
prescriptionID INT(3) NOT NULL primary key,
practitionerID INT(3),
PrescriptionDate date,
patientID INT(3),
RxNumber VARCHAR(10),
CONSTRAINT practitionerID2_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE
SET NULL,
CONSTRAINT patientID10_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL,
CONSTRAINT RxNumber_fk FOREIGN KEY (RxNumber) REFERENCES Medicine (RxNumber) ON DELETE SET NULL
);


CREATE TABLE DailyMasterSchedule
(
appointmentID INT(3),
practitionerID INT(3),
patientID INT(3),
CONSTRAINT appointmentID_fk FOREIGN KEY (appointmentID) REFERENCES APPOINTMENT (appointmentID) ON
DELETE CASCADE,
CONSTRAINT practitionerID12_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE
SET NULL,
CONSTRAINT patientID12_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL
);

CREATE TABLE WeeklyCoverageSchedule
(
staffID INT(3),
scheduleDate DATE,
hoursFrom DATETIME,
hoursTo DATETIME,
CONSTRAINT staffID_Date_pk PRIMARY KEY (staffID, scheduleDate),
CONSTRAINT staffID2_fk FOREIGN KEY (staffID) REFERENCES STAFF (staffID) ON DELETE CASCADE
);

CREATE TABLE IndividualPractitionerSchedule
(
practitionerID INT(3),
scheduleDate DATE,
hoursFrom DATETIME,
hoursTo DATETIME,
appointmentType VARCHAR(38),
patientID INT(3),
visitType VARCHAR(38),
CONSTRAINT practitionerID_Date_pk PRIMARY KEY (practitionerID, scheduleDate),
CONSTRAINT practitionerID13_fk FOREIGN KEY (practitionerID) REFERENCES STAFF (staffID) ON DELETE
CASCADE,
CONSTRAINT patientID13_fk FOREIGN KEY (patientID) REFERENCES PATIENT (patientID) ON DELETE SET
NULL
);
```

Step 6.5 Create indexes for foreign keys and any other columns that will be used most often for queries.

```
CREATE UNIQUE INDEX PATIENT_SSN ON PATIENT (SSN);
CREATE UNIQUE INDEX PATINET_ID ON PATIENT (PATIENTID);

CREATE UNIQUE INDEX STAFF_ID ON STAFF (STAFF_ID);
CREATE UNIQUE INDEX TAXID ON STAFF (TAXID);

CREATE UNIQUE INDEX APPOINTMENTID ON APPOINTMENT (APPOINTMENTID);
CREATE UNIQUE INDEX INSURANCEMEMBERID ON INSURANCE (INSURANCEMEMBERID);

CREATE UNIQUE INDEX BILLID ON BILL (BILLID);
CREATE UNIQUE INDEX PAYMENTID ON PAYMENT (PAYMENTID);
CREATE UNIQUE INDEX PRESCRIPTIONID ON PRESCRIPTION (PRESCRIPTIONID);
CREATE UNIQUE INDEX RxNumber ON MEDICINE (RXNUMBER);
CREATE UNIQUE INDEX TESTID ON TEST (TESTID);
CREATE UNIQUE INDEX SurgeryID ON OperatingRoomSchedule (SurgeryID);

CREATE UNIQUE INDEX APPOINTMENT_ID_DATE ON DailyMasterSchedule (AppointmentID, AppointmentDATE);
CREATE UNIQUE INDEX RECOVERY_ID ON RecoveryRoomLog (RecoveryID);

CREATE UNIQUE INDEX Delivery_ID_DATE ON DailyDeliveryRoomLog (DeliveryID, DeliveryDate);
CREATE UNIQUE INDEX Surgery_ID ON OperationRoomLog (SurgeryID);
CREATE UNIQUE INDEX testID ON LaboratoryLog (testID);

CREATE UNIQUE INDEX Practitioner_ID_DATE ON individualpractitionerschedule (practitionerID,
ScheduleDate);
CREATE UNIQUE INDEX Staff_ID_DATE ON weeklycoverageschedule (staffID, ScheduleDate);
```

Step 6.6 Insert about five records in each table, preserving all constraints. Put in enough data to demonstrate how the database will function.

```
INSERT INTO PATIENT (patientID, firstName, lastName, initialVisitDate, email, phone, address,
city, state, zipcode, DOB, SSN, currentMedicineTaken)
VALUES
```

```
(1, 'John', 'Doe', '2022-01-01', 'johndoe@gmail.com', 123456789, '123 Main St', 'Anytown', 'CA',
12345, '2000-01-01', 123456789, 'Aspirin'),
(2, 'Jane', 'Doe', '2022-01-02', 'janedoe@gmail.com', 234567890, '456 Elm St', 'Othertown', 'NY',
23456, '1999-01-01', 234567890, 'Ibuprofen'),
(3, 'Bob', 'Smith', '2022-01-03', 'bobsmith@gmail.com', 345678901, '789 Oak St', 'Somewhere',
'TX', 34567, '1980-01-01', 345678901, 'Paracetamol'),
(4, 'Mary', 'Johnson', '2022-01-04', 'maryjohnson@gmail.com', 456789012, '321 Pine St',
'Nowhere', 'FL', 45678, '1975-01-01', 456789012, 'Acetaminophen'),
(5, 'Tom', 'Brown', '2022-01-05', 'tombrown@gmail.com', 567890124, '654 Maple St', 'Everywhere',
'WA', 56789, '1985-01-01', 567890123, 'None');

INSERT INTO STAFF (staffID, firstName, lastName, phone, title, taxID, professional, practice)
VALUES
(1, 'Dr.', 'Johnson', 123456780, 'Doctor', 123456789, true, 'General Practice'),
(2, 'Nurse', 'Smith', 234567801, 'Nurse Practitioner', 234567890, true, 'Pediatrics'),
(3, 'Dr.', 'Lee', 345678901, 'Surgeon', 345678901, true, 'Cardiology'),
(4, 'Receptionist', 'Davis', 467890123, 'Receptionist', 456789012, false, 'General Practice'),
(5, 'Pharmacist', 'Brown', 568901234, 'Pharmacist', 567890123, true, 'Pharmacy');

INSERT INTO APPOINTMENT (appointmentID, appointmentDate, appointmentHoursFrom,
appointmentHoursTo, appointmentType, staffID, patientID)
VALUES
(1, '2022-01-01', '2022-01-01 09:00:00', '2022-01-01 09:30:00', 'Check-up', 1, 1),
(2, '2022-01-02', '2022-01-02 10:00:00', '2022-01-02 10:30:00', 'Follow-up', 2, 2),
(3, '2022-01-03', '2022-01-03 11:00:00', '2022-01-03 11:30:00', 'Check_up', 3, 3),
(4, '2022-01-04', '2022-01-04 12:00:00', '2022-01-04 12:30:00', 'Check_up', 4, 4),
(5, '2022-01-05', '2022-01-05 13:00:00', '2022-01-05 13:30:00', 'Follow-up', 5, 5);

INSERT INTO TEST (testID, testName, patientID, specimentType, result, description,
performStaffID, previewerStaffID)
VALUES
(1, 'Blood Test', 1, 'Blood', 'Normal', 'Complete Blood Count', 1, 2),
(2, 'Urine Test', 2, 'Urine', 'Normal', 'Urine analysis', 2, 3),
(3, 'MRI', 3, 'Body Tissue', 'Abnormal', 'Magnetic Resonance Imaging', 3, 4),
(4, 'CT Scan', 4, 'X-ray', 'Normal', 'Computed Tomography Scan', 4, 5),
(5, 'Ultrasound', 5, 'Body Tissue', 'Abnormal', 'Abdominal Ultrasound', 1, 2);
```

```
INSERT INTO LaboratoryLog (testID, staffID, patientID)
VALUES (1, 2, 3), (2, 1, 4), (3, 3, 5), (4, 2, 1), (5, 1, 2);

INSERT INTO OperatingRoomSchedule (surgeryID, surgeonID, sugeryDate, patientID, nurseID,
startTime, endTime, surgeryType, anesthesia, assistant)
VALUES
    (1, 1, '2023-04-10', 1, 2, '2023-04-10 10:00:00', '2023-04-10 12:00:00', 'Heart Surgery',
'General Anesthesia', 'John'),
    (2, 2, '2023-04-11', 2, 3, '2023-04-11 11:00:00', '2023-04-11 13:00:00', 'Brain Surgery',
'Local Anesthesia', 'Mary'),
    (3, 3, '2023-04-12', 3, 4, '2023-04-12 09:00:00', '2023-04-12 11:00:00', 'Knee Surgery',
'Regional Anesthesia', 'Bob'),
    (4, 4, '2023-04-13', 4, 5, '2023-04-13 12:00:00', '2023-04-13 14:00:00', 'Eye Surgery',
'Local Anesthesia', 'Kate'),
    (5, 5, '2023-04-14', 5, 1, '2023-04-14 13:00:00', '2023-04-14 15:00:00', 'Lung Surgery',
'General Anesthesia', 'Chris');

INSERT INTO OperatingRoomLog (surgeryID, surgeonID, surgeryDate, patientID, nurseID, startTime,
endTime, surgeryType)
VALUES
    (1, 1, '2023-04-10', 1, 2, '2023-04-10 08:00:00', '2023-04-10 09:00:00', 'Appendectomy'),
    (2, 2, '2023-04-10', 2, 3, '2023-04-10 10:00:00', '2023-04-10 11:30:00', 'Knee Replacement'),
    (3, 3, '2023-04-11', 3, 4, '2023-04-11 08:30:00', '2023-04-11 09:45:00', 'Hernia Repair'),
    (4, 4, '2023-04-12', 4, 5, '2023-04-12 11:00:00', '2023-04-12 12:30:00', 'Cataract Surgery'),
    (5, 5, '2023-04-12', 5, 1, '2023-04-12 13:00:00', '2023-04-12 14:00:00', 'Laser Eye
Surgery');

INSERT INTO DailyDeliveryRoomLog (deliveryID, deliveryDate, patientID, pharmacistID, midwifeID,
supervisorID)
VALUES
(1, '2023-04-09', 1, 1, 2, 3),
(2, '2023-04-08', 2, 2, 3, 4),
(3, '2023-04-07', 3, 3, 4, 5),
(4, '2023-04-06', 4, 4, 5, 1),
(5, '2023-04-05', 5, 5, 1, 2);
```

```
INSERT INTO RecoveryRoomLog (recoveryID, patientID, practitionerID, dateIn, timeIn, dateOut,
timeOut, checkPractitionerID)
VALUES
(101, 1, 1, '2023-04-09', '2023-04-09 12:00:00', '2023-04-09', '2023-04-09 15:30:00', 2),
(102, 2, 2, '2023-04-10', '2023-04-10 09:00:00', '2023-04-10', '2023-04-10 13:00:00', 3),
(103, 3, 3, '2023-04-11', '2023-04-11 14:00:00', '2023-04-11', '2023-04-11 17:00:00', 4),
(104, 4, 4, '2023-04-12', '2023-04-12 11:30:00', '2023-04-12', '2023-04-12 14:45:00', 5),
(105, 5, 5, '2023-04-13', '2023-04-13 08:45:00', '2023-04-13', '2023-04-13 12:00:00', 1);

INSERT INTO Insurance (insuranceMemberID, patientID, insuranceCompany, insurancePlan)
VALUES
(1234567, 1, 'Blue Cross Blue Shield', 'Standard Plan'),
(2345678, 2, 'Aetna', 'Gold Plan'),
(3456789, 3, 'Cigna', 'Platinum Plan'),
(4567890, 4, 'UnitedHealthcare', 'Basic Plan'),
(5678901, 5, 'Humana', 'Premium Plan');

INSERT INTO Bill (billID, billDate, practitionerID, patientID, insuranceCoverageAmount,
balanceDue)
VALUES
(1, '2022-03-15', 1, 1, 500.00, 1000.00),
(2, '2022-03-20', 2, 2, 750.00, 1500.00),
(3, '2022-04-02', 3, 3, 250.00, 750.00),
(4, '2022-04-10', 4, 4, 1000.00, 2000.00),
(5, '2022-04-12', 5, 5, 500.00, 1000.00);

INSERT INTO Payment(paymentID, billID, paymentDate, paymentAmount)
VALUES
(1, 1, '2022-03-15', 100.00),
(2, 2, '2022-04-10', 50.00),
(3, 3, '2022-05-05', 75.00),
(4, 4, '2022-06-20', 25.00),
(5, 5, '2022-07-30', 200.00);

INSERT INTO Medicine (RxNumber, medicineName) VALUES
```

```
('RX12345', 'Ibuprofen'),
('RX23456', 'Aspirin'),
('RX34567', 'Acetaminophen'),
('RX45678', 'Amoxicillin'),
('RX56789', 'Lisinopril');

INSERT INTO Prescription (prescriptionID,practitionerID, prescriptionDate,  patientID, RxNumber)
VALUES
(1, 1, '2023-04-01', 1, 'RX12345'),
(2, 2, '2023-04-02', 2, 'RX23456'),
(3, 3, '2023-04-03', 3, 'RX34567'),
(4, 4, '2023-04-04', 4, 'RX45678'),
(5, 5, '2023-04-05', 5, 'RX56789');


INSERT INTO WeeklyCoverageSchedule (staffID, scheduleDate, hoursFrom, hoursTo) VALUES
(1, '2023-04-10', '08:00:00', '16:00:00'),
(2, '2023-04-10', '09:00:00', '17:00:00'),
(3, '2023-04-11', '12:00:00', '20:00:00'),
(4, '2023-04-11', '10:00:00', '18:00:00'),
(5, '2023-04-12', '11:00:00', '19:00:00');

INSERT INTO IndividualPractitionerSchedule (practitionerID, scheduleDate, hoursFrom, hoursTo,
appointmentType, patientID, visitType)
VALUES
(1, '2023-04-10', '2023-04-10 09:00:00', '2023-04-10 10:00:00', 'Follow-up', 1, 'In-person'),
(2, '2023-04-11', '2023-04-11 14:00:00', '2023-04-11 16:00:00', 'New patient', 2, 'Virtual'),
(3, '2023-04-12', '2023-04-12 11:00:00', '2023-04-12 12:30:00', 'Follow-up', 3, 'In-person'),
(4, '2023-04-13', '2023-04-13 13:30:00', '2023-04-13 14:30:00', 'New patient', 4, 'Virtual'),
(5, '2023-04-14', '2023-04-14 15:00:00', '2023-04-14 16:00:00', 'Follow-up', 5, 'In-person');

INSERT INTO DailyMasterSchedule (appointmentID, practitionerID, patientID) VALUES
(1, 1, 1),
(2, 2, 2),
(3, 3, 3),
(4, 4, 4),
(5, 5, 5);
```

Step 6.7 Write SQL statements that will process five non-routine requests for information from the database just created. For each, write the request in English, followed by the corresponding SQL command.

- Retrieve the appointment details of a particular patient with patient ID 1

  ```
  SELECT * FROM APPOINTMENT WHERE patientID = 1;
  ```

- Retrieve the list of patients who have undergone surgery:

  ```
  SELECT DISTINCT P.firstName, P.lastName
  FROM PATIENT P
  JOIN OperatingRoomSchedule OS ON P.patientID = OS.patientID;
  ```

- Retrieve the list of staff members who have performed a test on a particular patient with patient ID 2:

  ```
  SELECT S.firstName, S.LastName
  FROM STAFF S
  JOIN TEST T ON S.staffID = T.performStaffID
  WHERE T.patientID = 2;
  ```

- Retrieve the list of patients who have not undergone any surgery:

  ```
  SELECT P.firstName, P.lastName
  FROM PATIENT P
  LEFT JOIN OperatingRoomSchedule OS ON P.patientID = OS.patientID
  WHERE OS.surgeryID IS NULL;
  ```

- Retrieve the total number of appointments for each staff member:

  ```
  SELECT S.staffID, S.firstName, S.LastName, COUNT(*) AS appointment_count
  FROM STAFF S
  JOIN APPOINTMENT A ON S.staffID = A.staffID
  GROUP BY S.staffID, S.firstName, S.LastName;
  ```

Next, we would like to implement security features for Wellness Clinic Medical Group

Step 8.1 Create a value-independent view that hides some private information.

The view will be of the Patient table that has the schema

Patient (patientID, lastName, firstName, InitialVisitDate, email, phone, address, city, state, zipcode, DOB, SSN, currentMedication)

The view should not contain the social security number, date of birth, or address information.
The SQL code is

```
CREATE VIEW PatientView1 AS

      SELECT patientID, lastName, firstName, InitialVisitDate, email, phone,
city, state, zipcode, currentMedication

      FROM Patient;
```

Step 8.2 Create a value-independent view that shows the patient's surgery information including patient first name, patient last name, surgery date, surgeon first name, surgeon last name, start time, end time, surgery type, and anesthesia of the patients who have undergone surgery.

The view will use a join of the Patient table with the Operating Room Schedule table, which has the schema.

OperatingRoomSchedule (surgeryId, surgeonId, surgeryDate, patientId, nurseId, startTime, endTime, surgeryType, anesthesia, assistant)

Staff (staffID, firstName, lastName, Phone, title, taxID, professional, practice)

```
CREATE VIEW PatientSurgeryView1 AS

      SELECT DISTINCT P.firstName AS PatientFirstName, P.lastName AS
PatientLastName, OS.surgeryDate, S.firstName AS SurgeonFirstName, S.lastname
AS SurgeonLastName, OS.startTime, OS.endTime, OS.surgeryType, OS.anesthesia

FROM PATIENT P

JOIN OperatingRoomSchedule OS ON P.patientID = OS.patientID

JOIN STAFF S ON OS.surgeonID = S.staffID;
```

Step 8.3 Write a trigger for an audit trail for updates to a sensitive item that users can update and test it by updating the item.

The trigger will monitor changes when the patient's payment is greater than the billing so that it indicates the patient has paid the bill in pull.

```
CREATE TABLE PaymentBillingAudit(

datePaid DATE,

billingID INT,

balanceDue float,

payment float,

paid_in_full Bool);


CREATE OR REPLACE TRIGGER PaymentBillingAuditTrail

AFTER UPDATE ON PAYMENT

FOR EACH ROW

BEGIN

  WHEN NEW.payment > (SELECT balanceDue FROM BILLING WHERE billingID = NEW.billingID) THEN

    UPDATE PaymentBillingAudit SET paid_in_full = true WHERE billingID = NEW.billingID;
```

```
END;

/*
-- test the trigger
COMMIT;
UPDATE BILL SET balanceDue = 999.99 HWERE BILLID = 1;
SELECT balanceDue from BILL WHERE BILLID = 1;
UPDATE PAYMENT SET PaymentAmount = 1000;
SELECT PaymentAmount from PAYMENT WHERE BILLID = 1;
SELECT * FROM PaymentBillingAudit;
ROLLBACK;
SELECT balanceDue from BILL;
SELECT paymentAmount from PAYMENT;
*/
```