CSCI 326

Group Project Report

Amir Beshir – 2023006017

Adam El Mouddene – 2023006464

Abdullah Farah– 2023006424

Mohamed AbouSaada – 2023006306

Khalid Hafiz – 2023006169

Anas Qaiser – 2023005920

Fall 2025

Instructor: Dr Zubaidah ALHazza

1- Introduction :

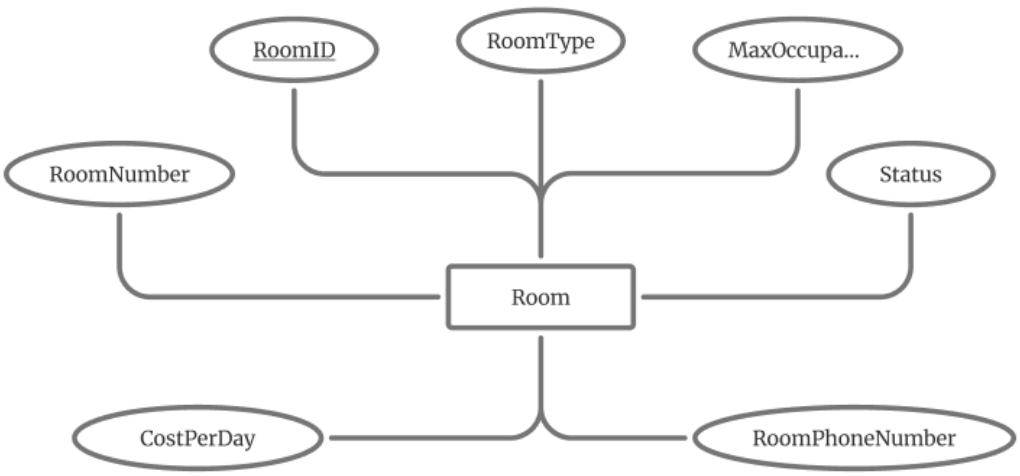
The capacity to effectively store, arrange, and retrieve information is a crucial factor in determining high-quality patient care in nowadays healthcare settings. Every day, hospitals deal with massive amounts of data, including patient records, appointments, staff schedules, medication administration, billing, and resource management. Errors, delays, and inconsistencies frequently arise when such data is handled manually or with basic spreadsheet software. An organized, dependable, and expandable solution to these problems is offered by a well-designed database management system.

The design and implementation of an HMS database to support a healthcare facility's essential functions is the main goal of this project. All of the essential elements, including patient data, medical personnel, rooms, treatments, appointments, and administrative procedures, will be managed and arranged by the system. This project demonstrates how an efficient database can accomplish data integrity, improve workflow coordination, and encourage better informed decision-making at a healthcare facility by applying database design principles, ER modeling, relational mapping, and normalization.

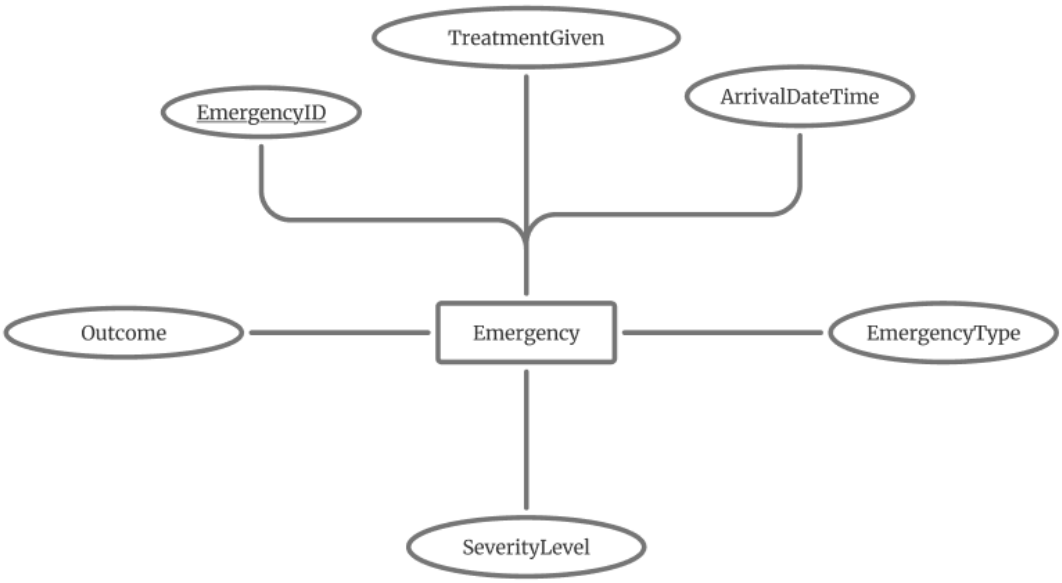
2.- ER Model:

- ***Entities :***

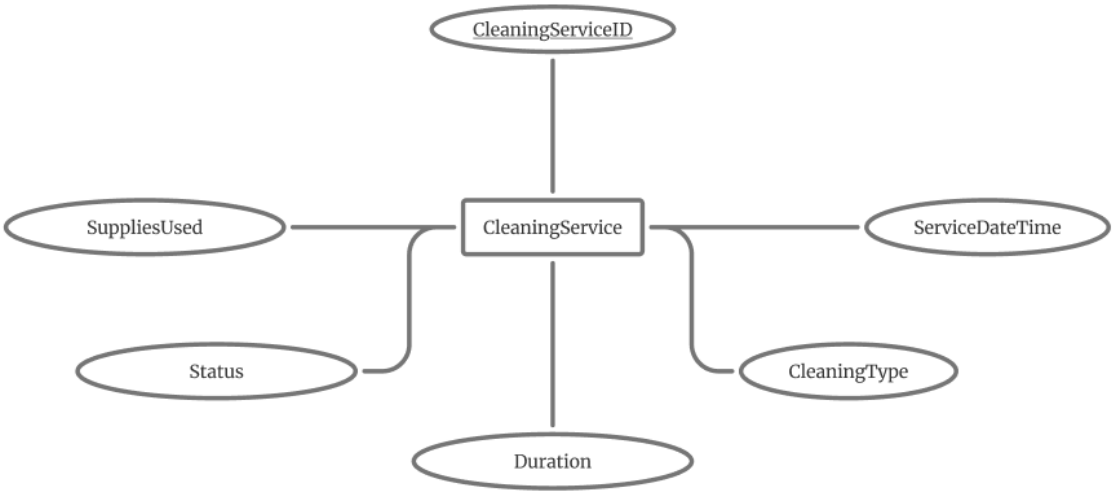
Adam :



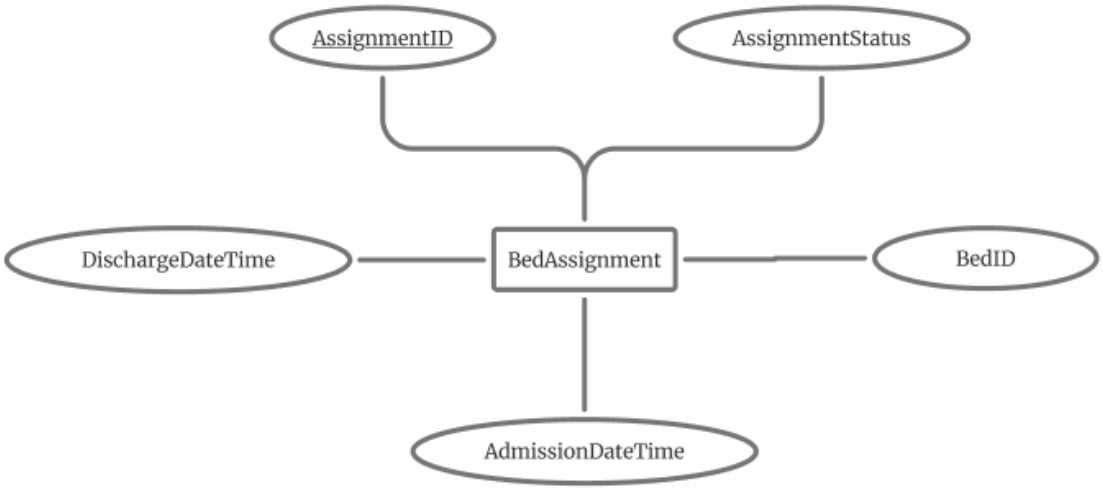
The *Room* entity stores information about each hospital room.



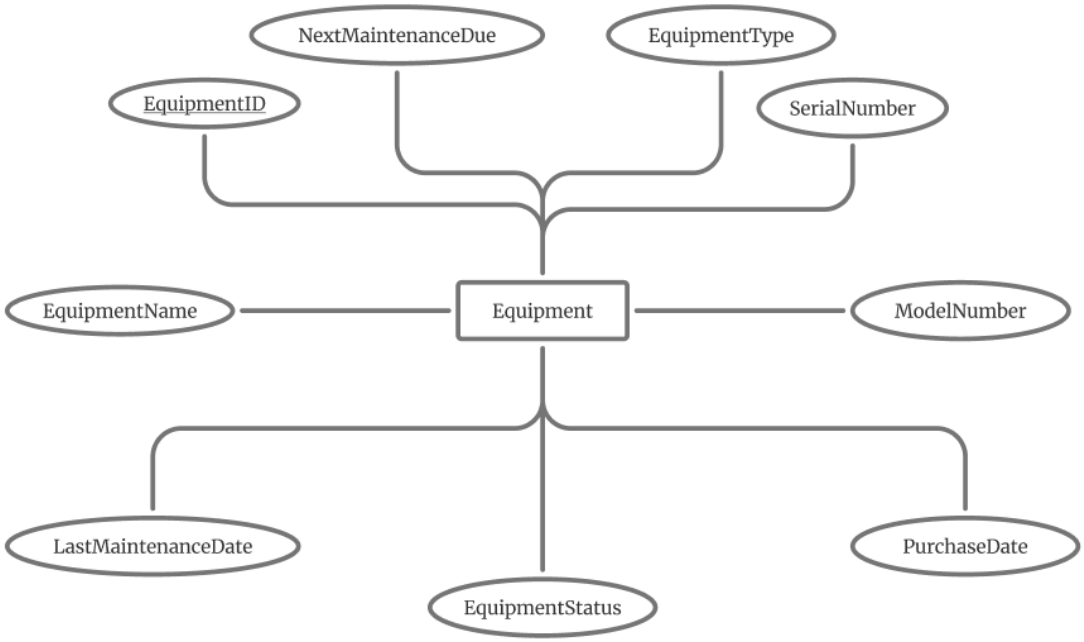
The Emergency entity represents emergency cases that arrive at the hospital.



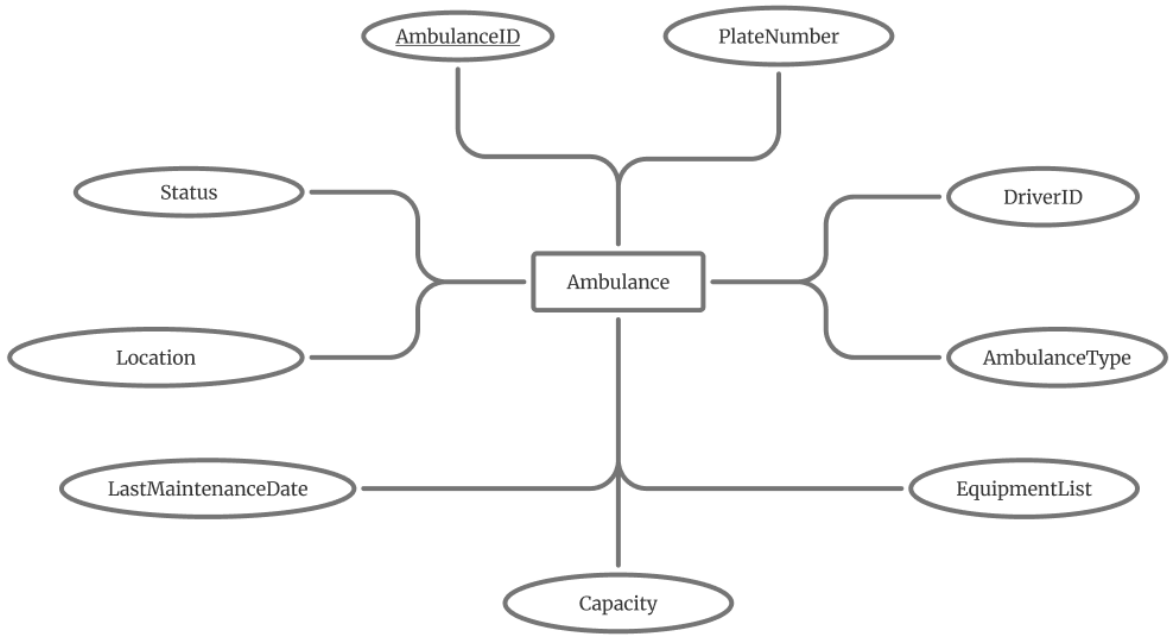
The CleaningService entity tracks cleaning operations performed in the hospital.



The BedAssignment entity manages the allocation of beds to patients over time.

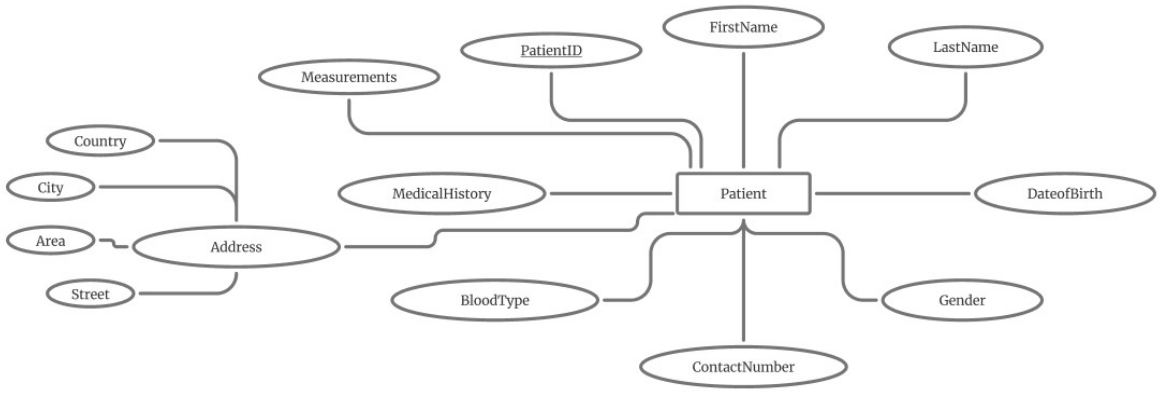


The Equipment entity contains data about medical equipment in the hospital.

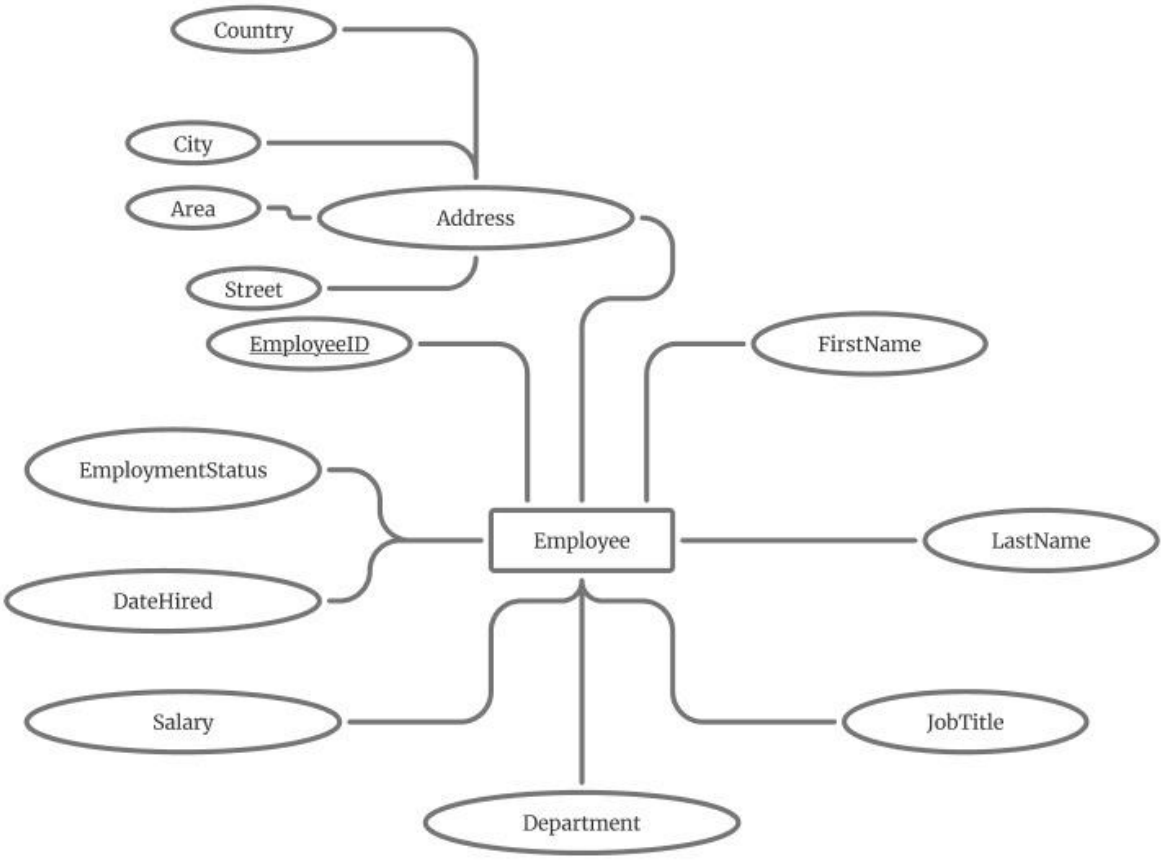


The Ambulance entity manages all information related to hospital ambulances.

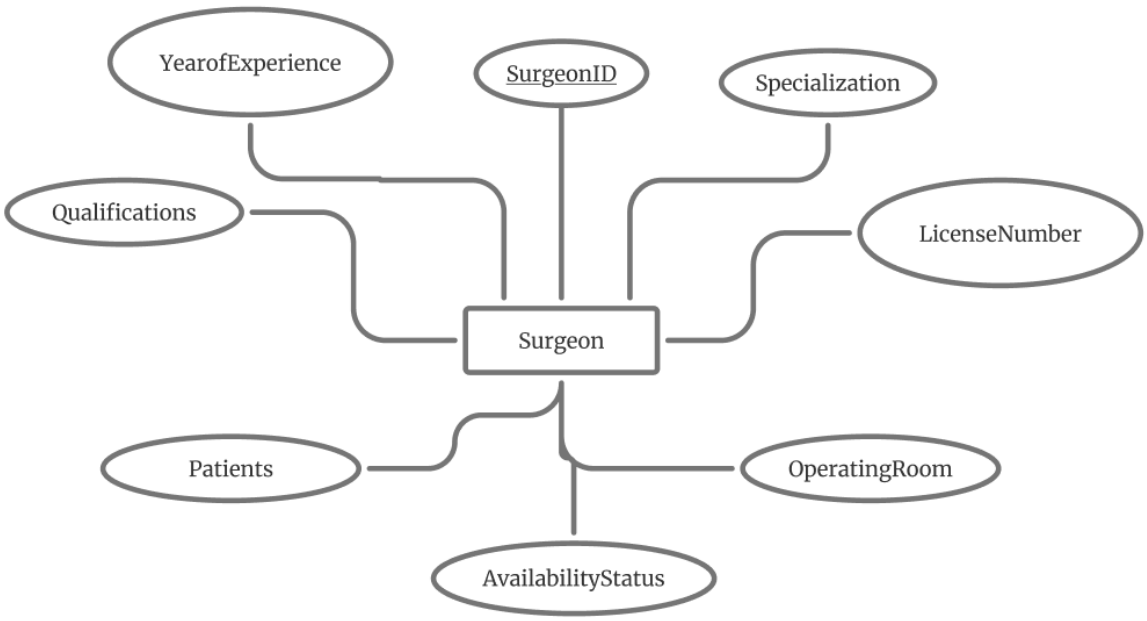
Abdullah:



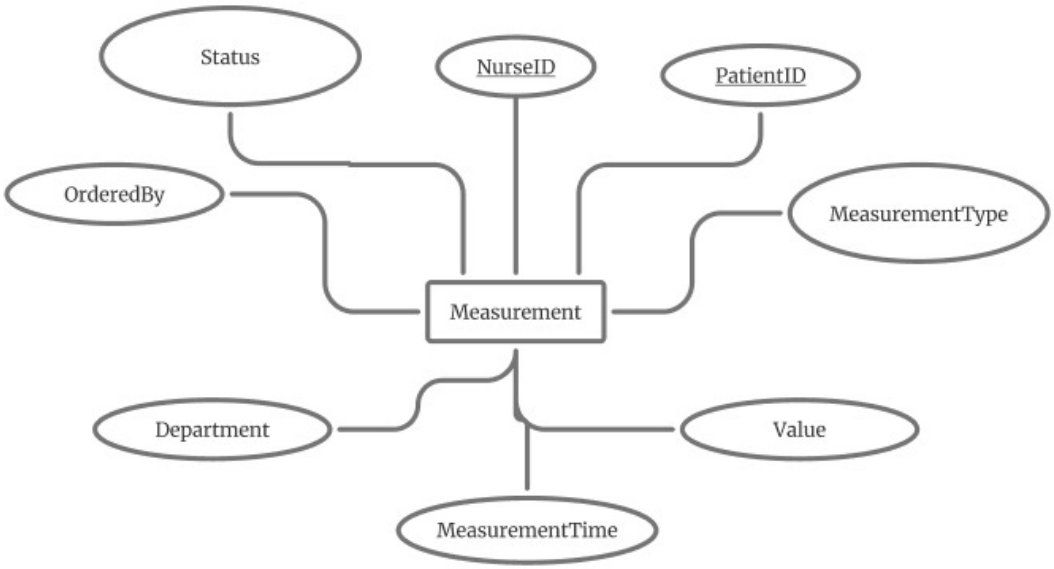
The *Patient* entity manages all information related to all patients in the hospital.



The *Employee* entity manages all information related to all employees in the hospital.

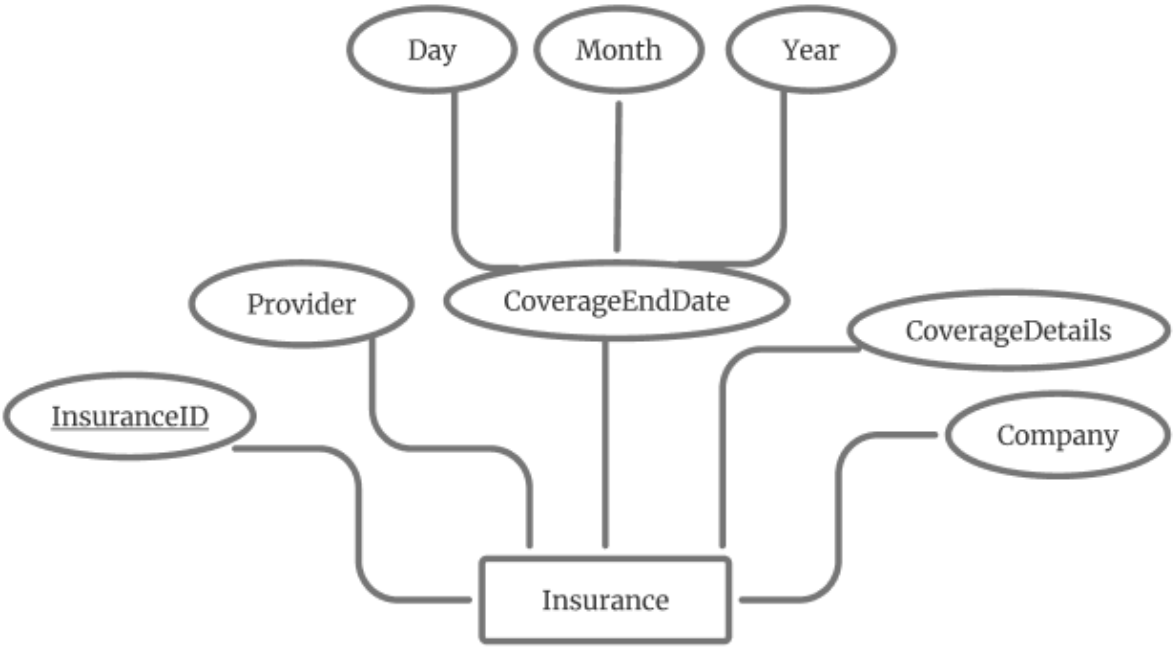


The *Surgeon* entity manages all information related to all surgeons in the hospital.

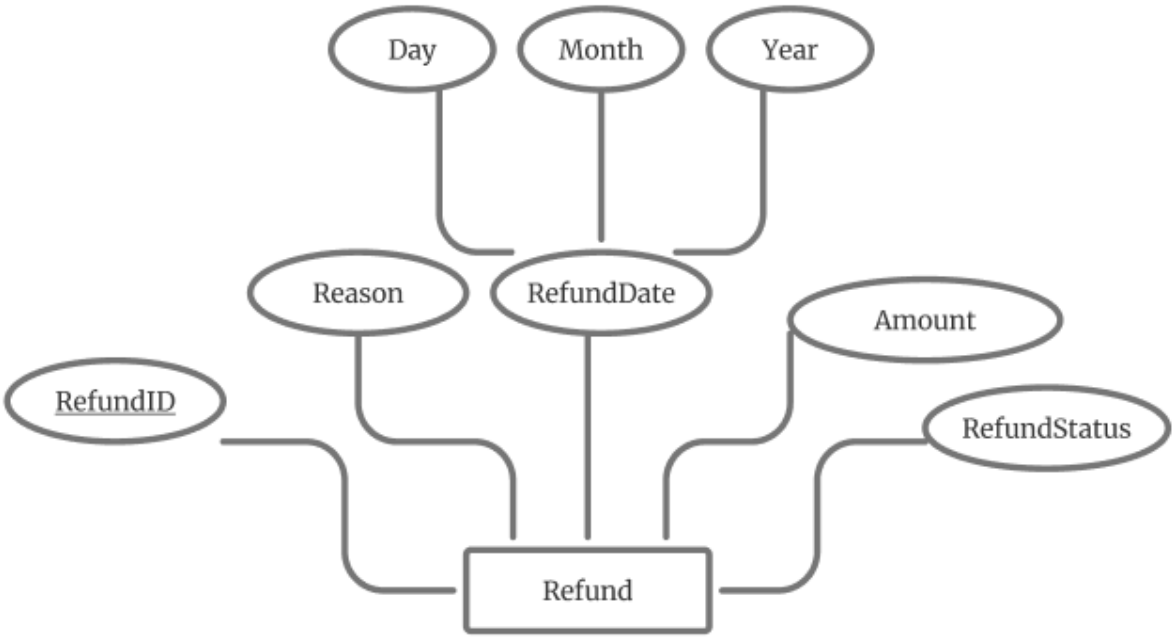


The *Measurement* entity manages all information related to patient measurements in the hospital.

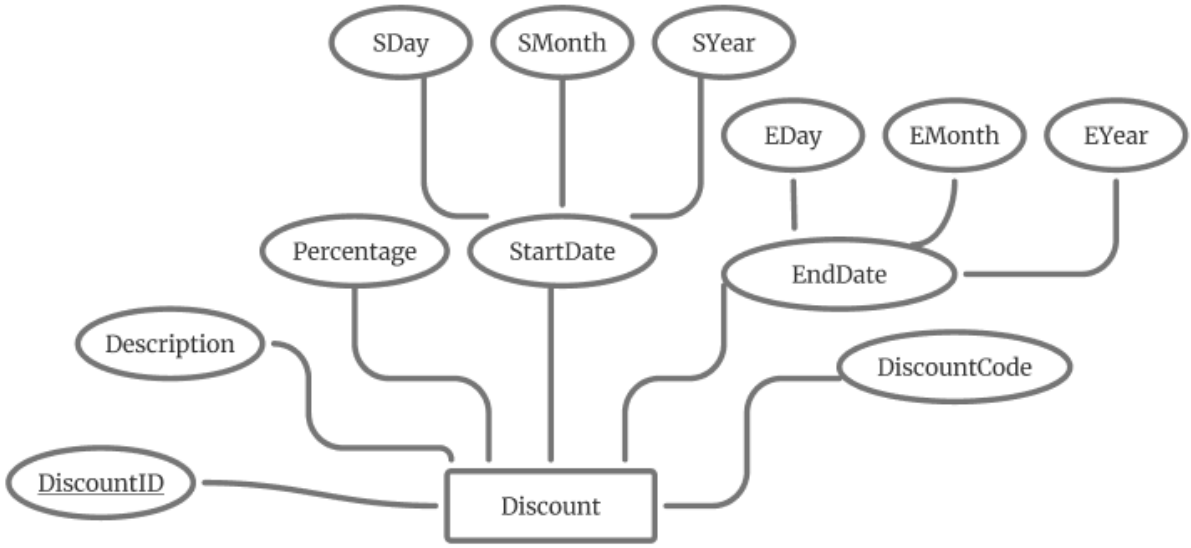
Amir:



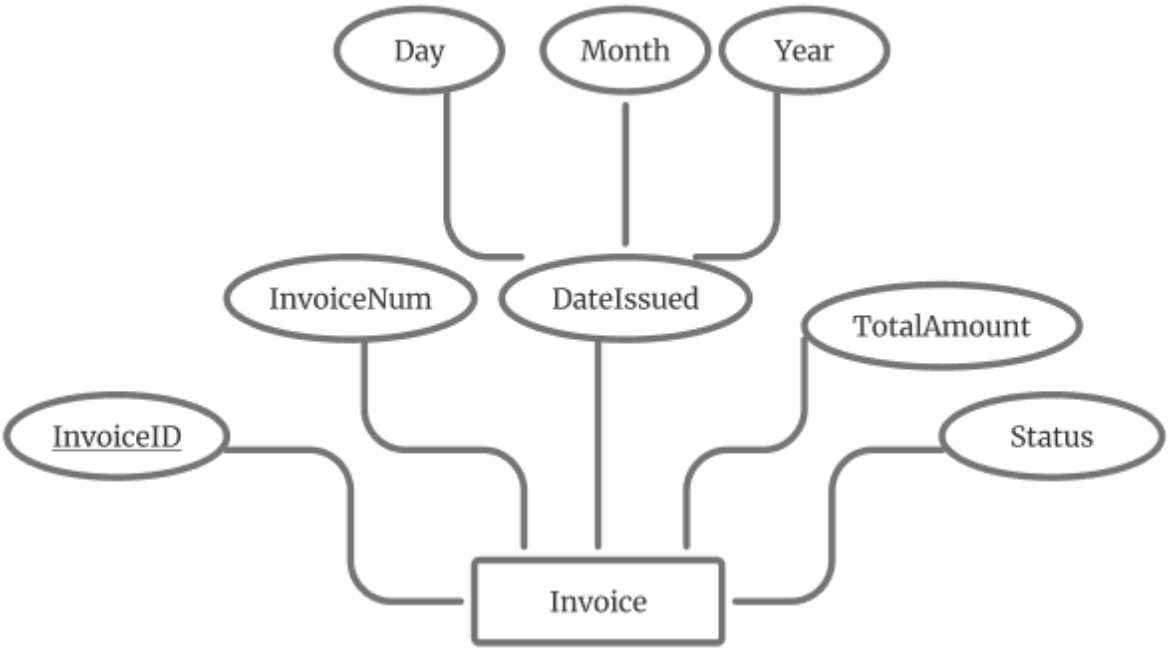
The Insurance entity covers all attributes related to the insurance of a patient such as the unique ID, the insurance provider, the expiry date, the details and the company.



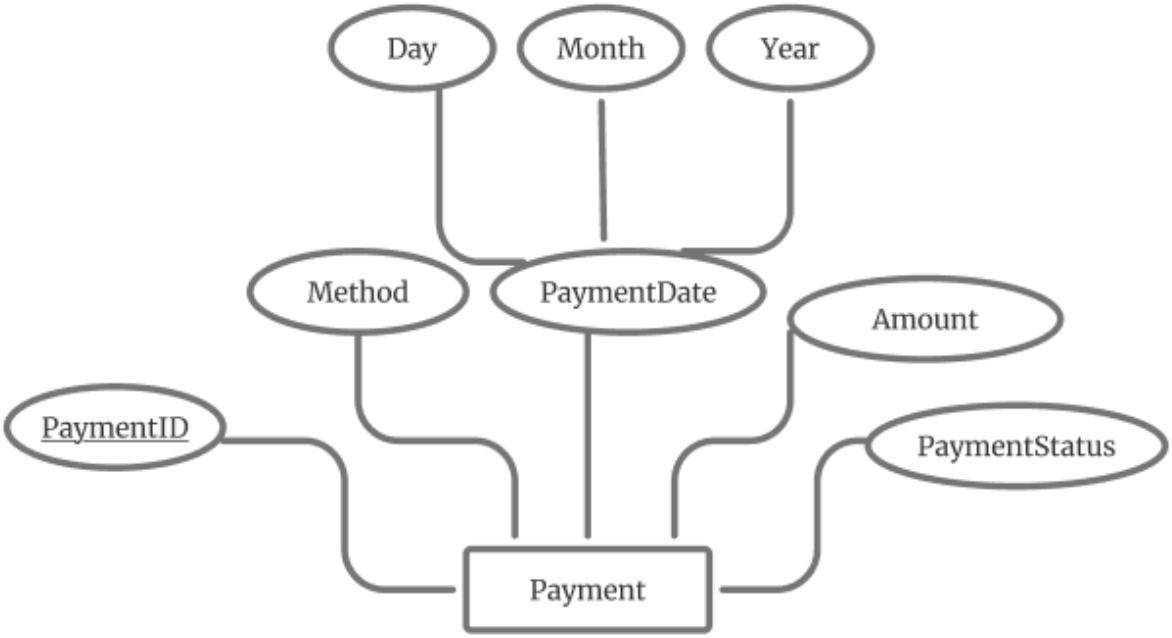
The Refund entity holds all the details related to a refund done in the hospital such as the refund ID, the reason, the date, the amount and its status (pending, rejected, accepted).



The Discount entity has all the information related to the discounts given such as their IDs, the code, the description, the percentage, and the start/end dates.

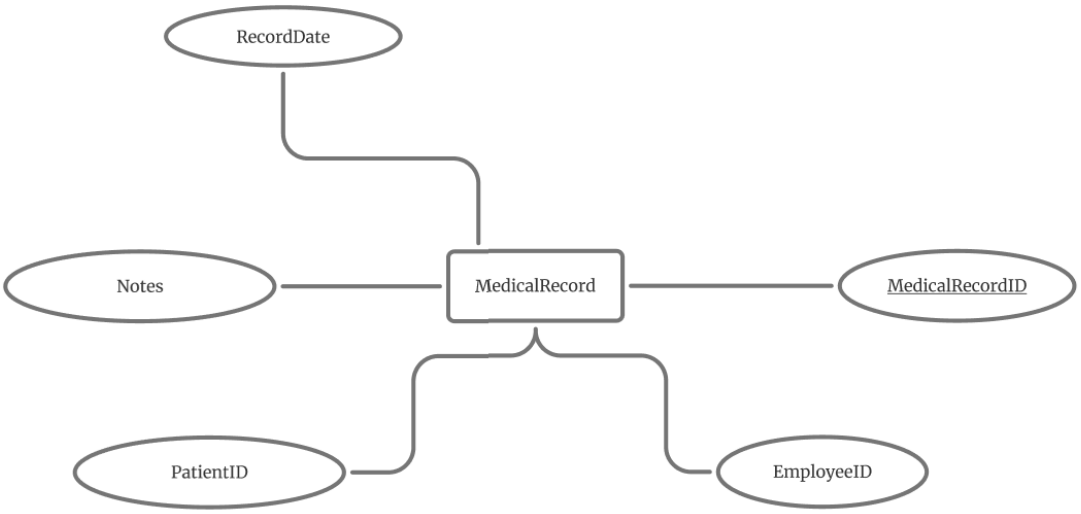


The Invoice entity has all the attributes related to the invoices printed and given out in the hospital such as the unique IDs, the number, the date issued, the total amount and its status.

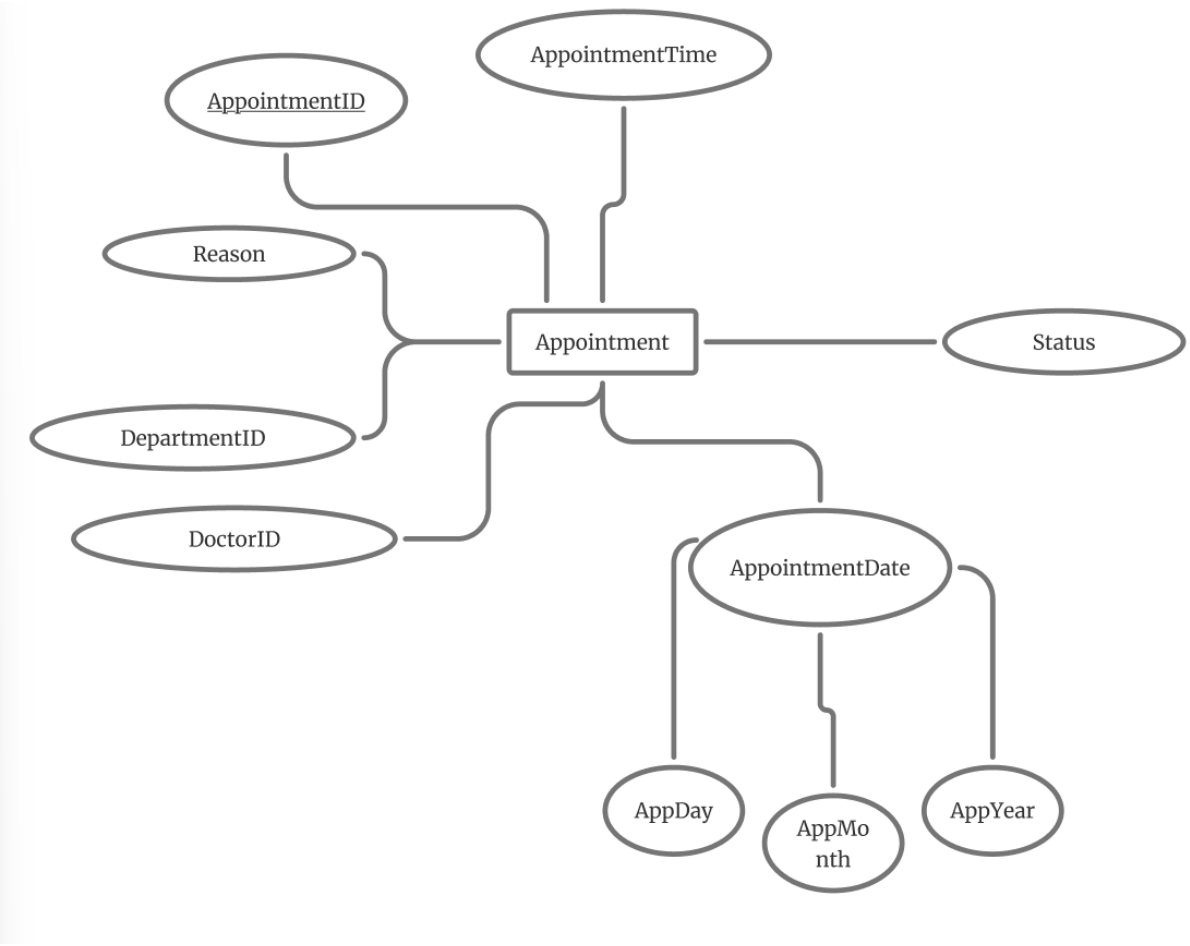


The Payment entity tracks the payments across the hospital by checking the unique IDs, the method of payment (card, cash, bank transfer, etc.), the payment date, the amount, and the status (pending, rejected, accepted)

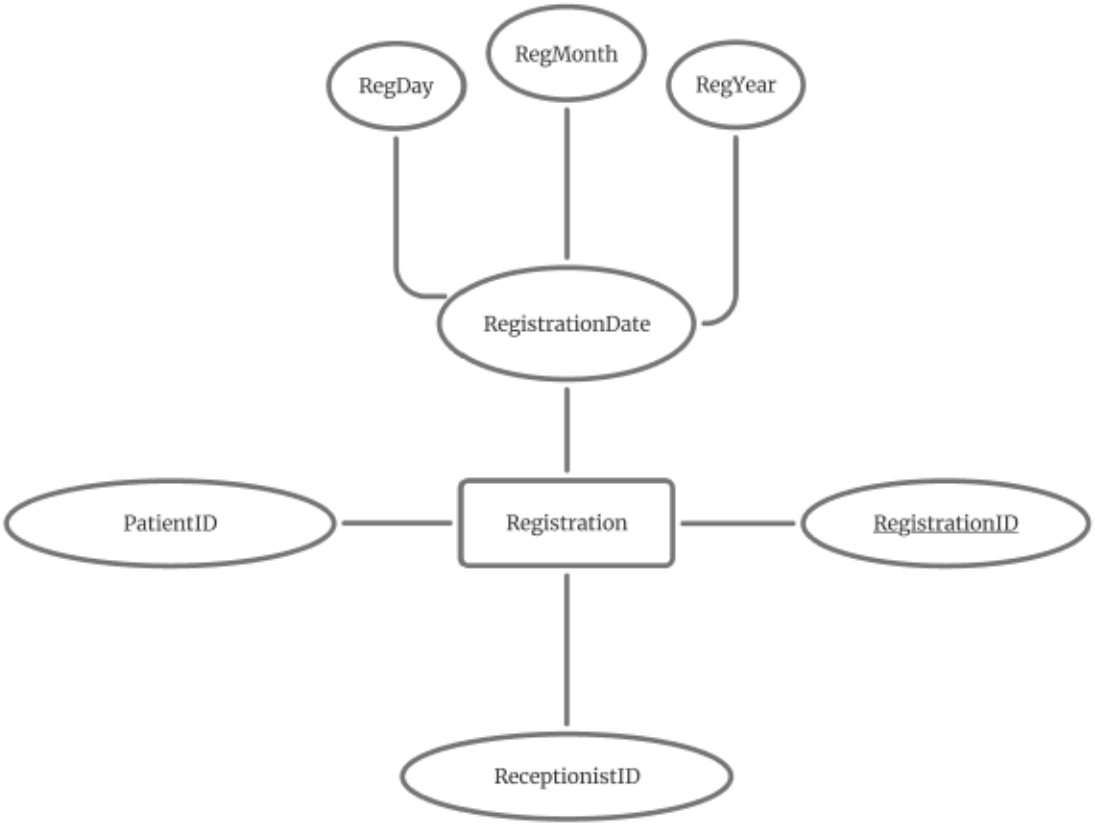
Anas:



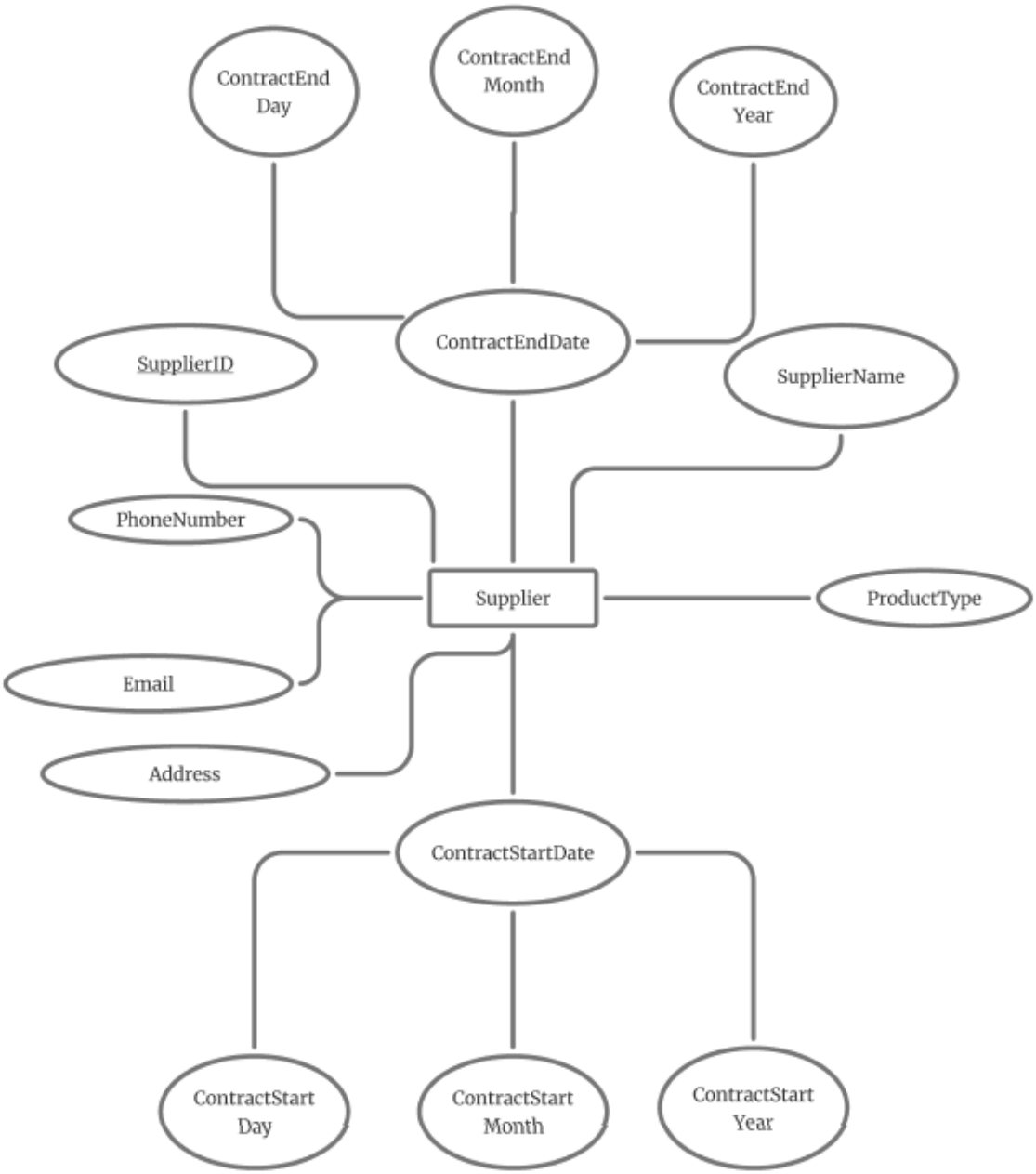
MedicalRecord entity manages all information regarding the records of patients



The Appointment Entity handles all information related to Patient Appointments such as their timings , which doctor they are having an appointment with and the reason for the appointment.

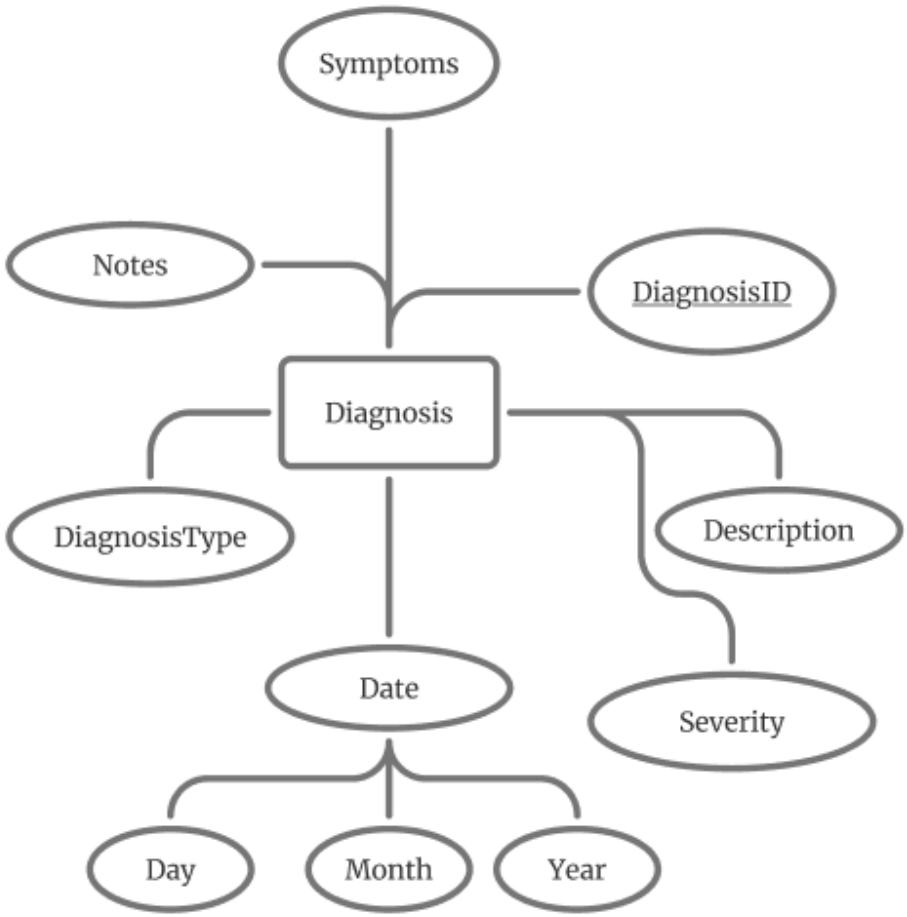


Registration Entity Manages All information regarding the Registration of a new Patient. It stores the PatientID , the date and the ID of the receptionist who registered the patient.

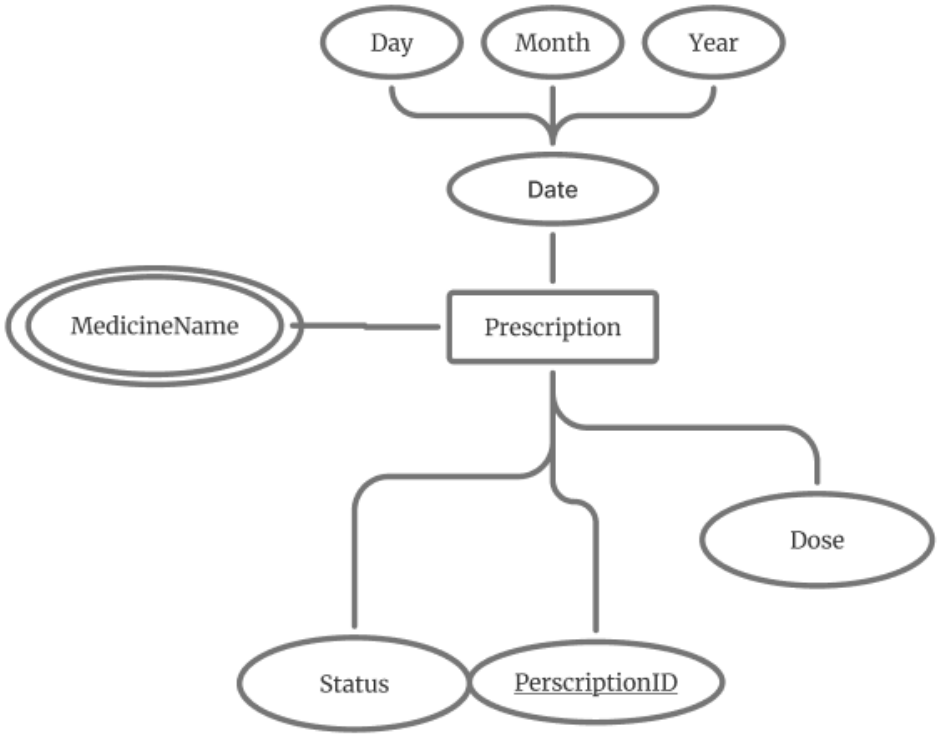


The Supplier Entity Stores All Information about the supplier such as their name and contact Information, The Start and End dates of the contracts and what type of products they supply to the Hospital.

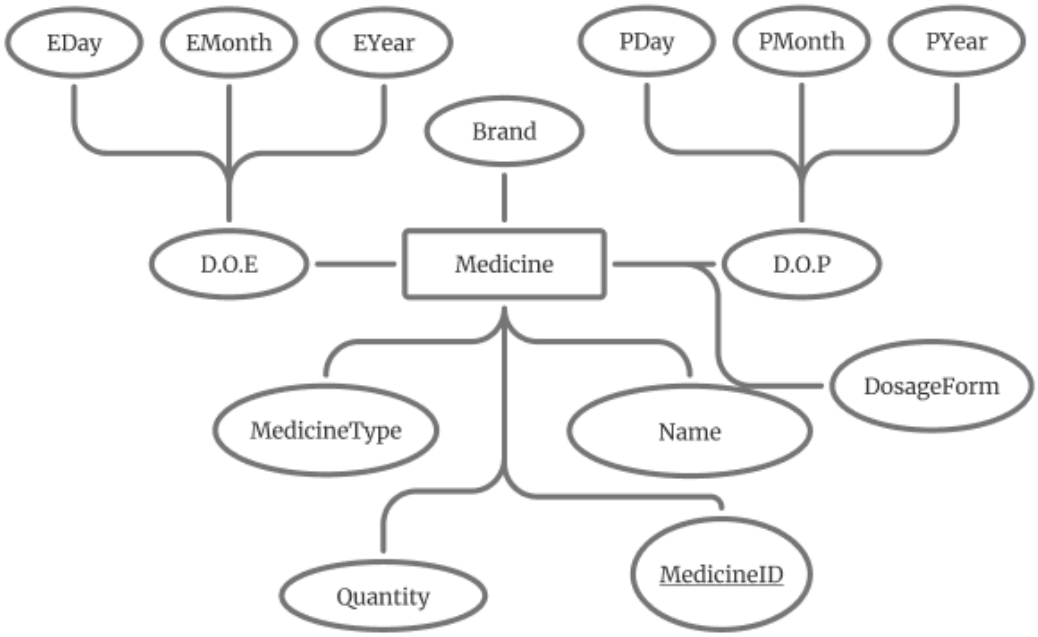
Mohamed:



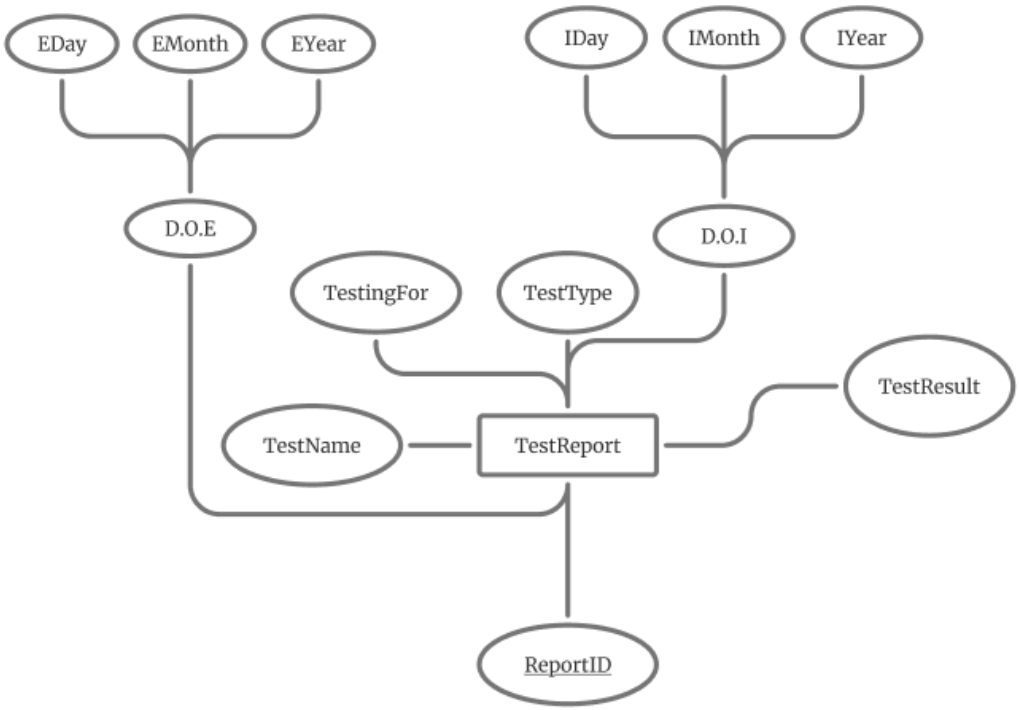
The diagnosis entity handles all the information regarding the diagnosis a doctor makes when a patient is admitted/walks in the hospital such as their symptoms and its description, severity, the date, any additional notes the doctor wants to add, and the type of diagnosis (differential, etc.)



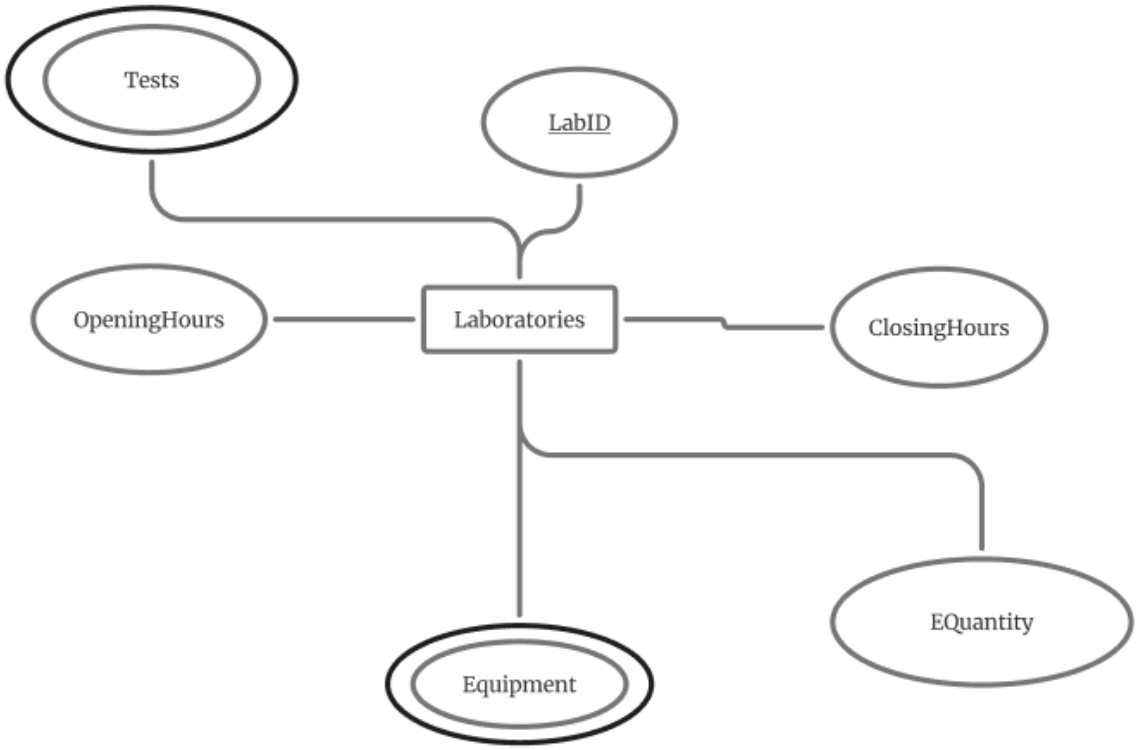
The prescription entity holds the information written on each prescription should a patient require one such as the medicine name (multi value since a prescription can have more than one), the unique ID, dosage, date produced and the prescription status (active, complete, or cancelled)



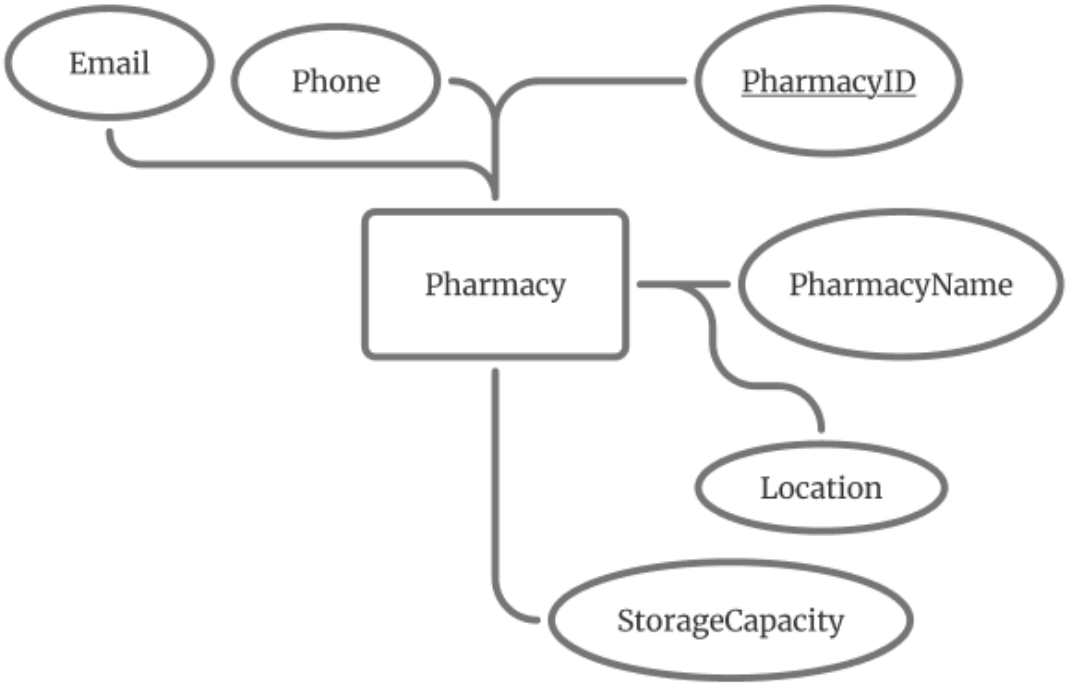
The medicine entity holds all the information regarding the medicine stored in the hospital with a unique local identifier (MedicineID) that can be used to search up each specific one. It has attributes such as the medicine name, type (syrup, pills, etc.), quantity held in storage, dosage form (oral, injected, etc.), the company/brand, and the production and expiry dates.



The test report entity has all the information regarding tests printed from the laboratories at the order of any allowed faculty. Its attributes are the unique report ID, the name of the test conducted, what the technician is testing for, the type of test, the results, and the issuing and expiry dates.

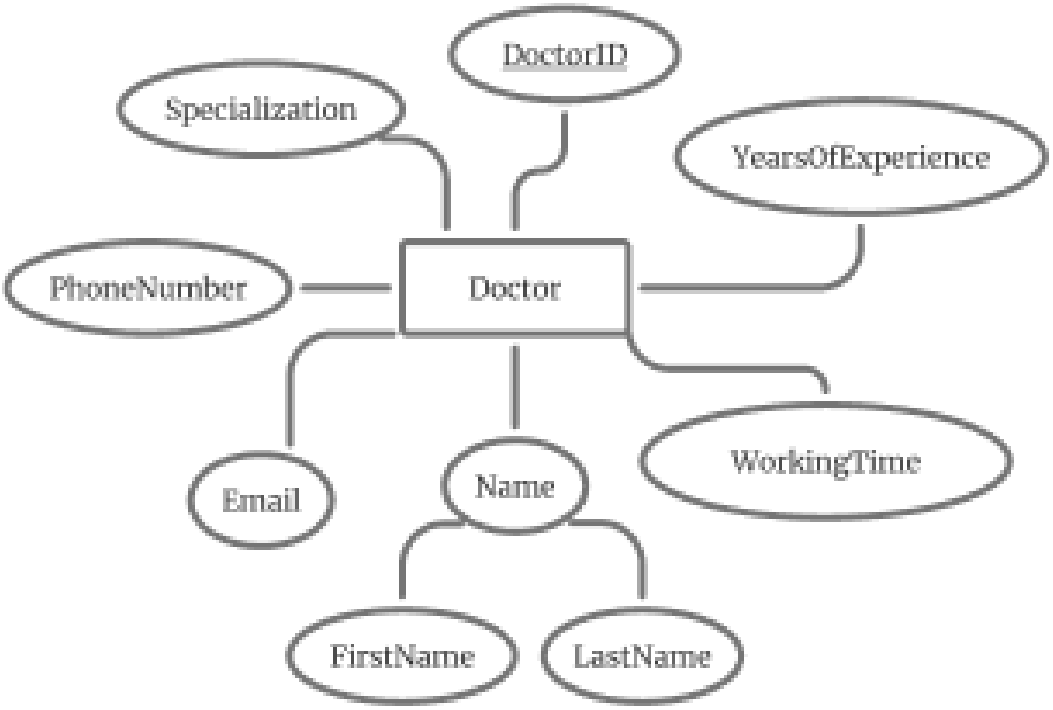


The laboratories entity describes all the laboratories in the hospital with the assumption that there are multiple that can do different tests. The attributes are the unique lab ID, the opening and closing hours, the tests, equipment held and their quantity.



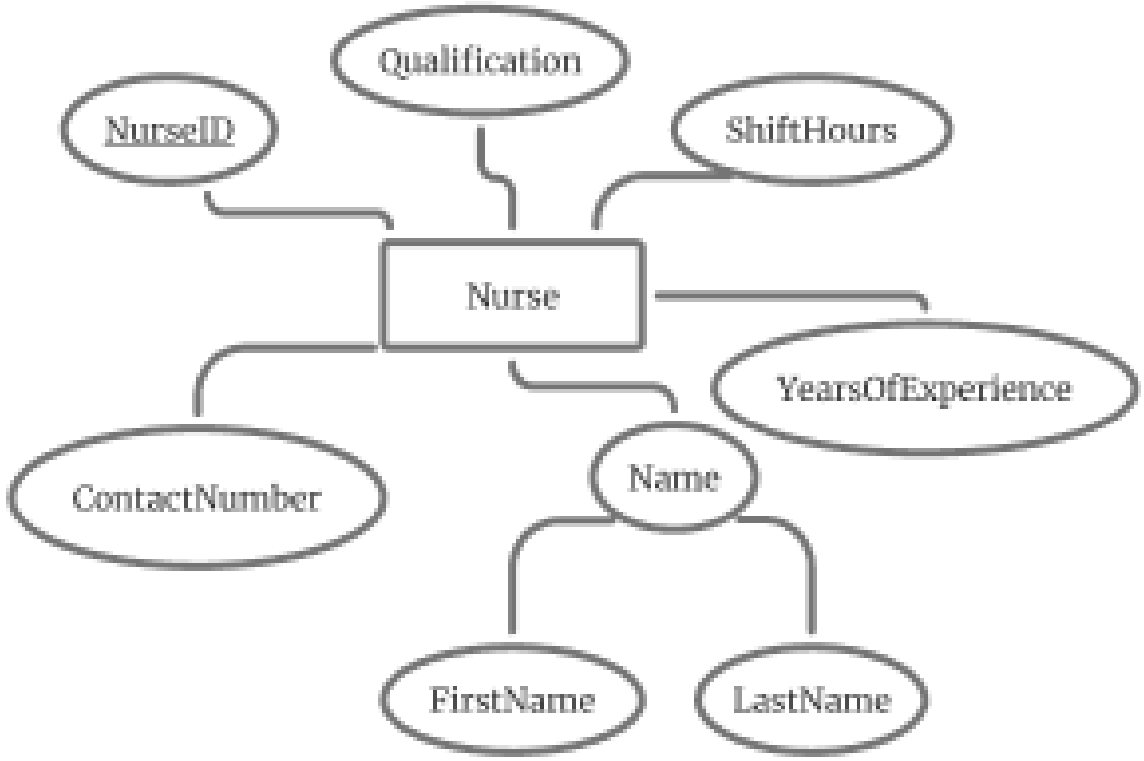
The pharmacy entity records all the information regarding the pharmacies across the hospital with the assumption that there can be more than 1. The attributes are the unique ID given to each one, their email, phone numbers, name, location, and storage capacity.

Khalid:



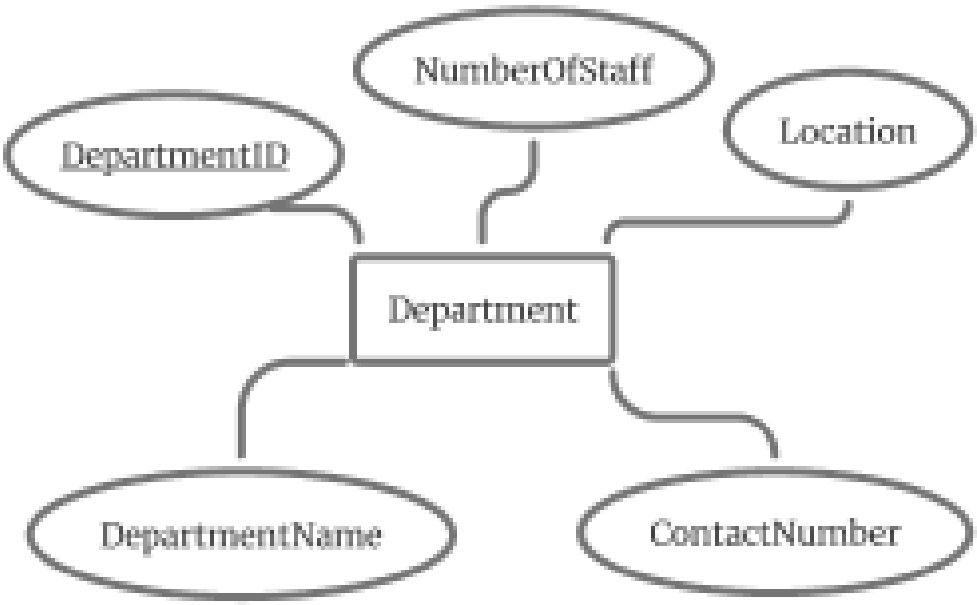
Doctor

The Doctor entity manages all information related to doctors working in the hospital, including their specialization, experience, and contact details.



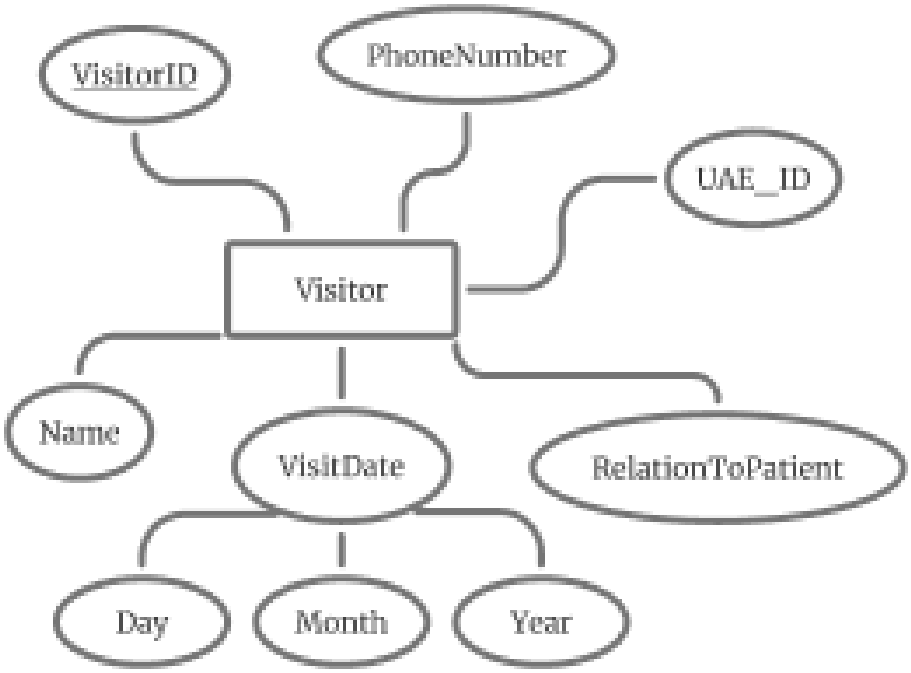
Nurse

The Nurse entity stores details about all nurses, such as their qualifications, shift hours, experience, and personal contact information.



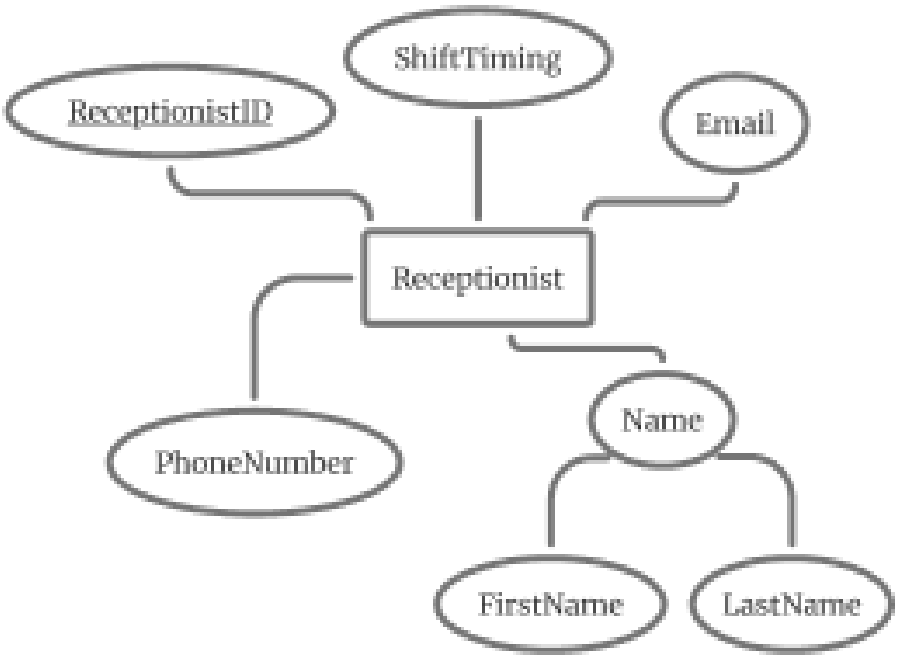
Department

The Department entity keeps track of each hospital department, including its name, location, number of staff, and contact information.



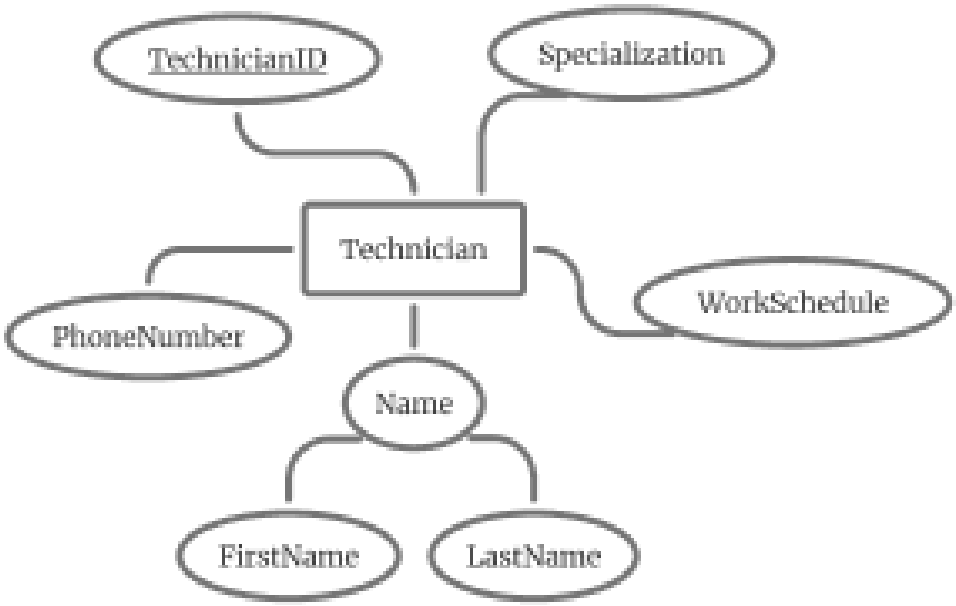
Visitor

The Visitor entity records information about people visiting patients, including their visit date, relationship to the patient, and contact details.



Receptionist

The Receptionist entity maintains all information about receptionists, such as their shift timings, email, and basic personal details.



Technician

The Technician entity manages information about hospital technicians, including their specialization, work schedule, and contact details.

- ***Relationships:***

Color code :

T= Total Participation

- Means
  1-N(T) & 1(T)-N(T) Foreign Key approach, PK of 1 on N table
- Means
  (T) 1-N Foreign Key approach, pk of N in table of 1
- Means
  1-N Relationship Relation
- Means
  N-N Relationship relation
- Means
  1(T)-1(T) Merge Tables
- Means
  1-1(T) Foreign Key approach, PK of side without (T) into table of 1

Adam :

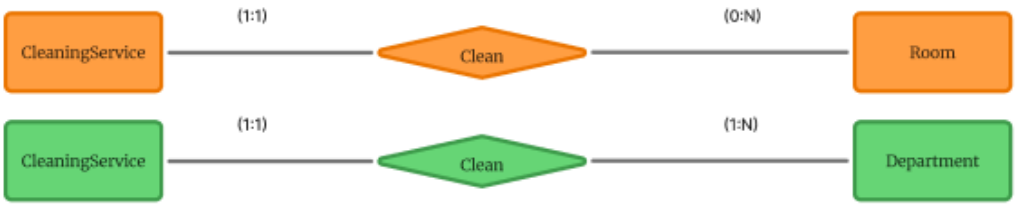
Room :

<u>Assumptions:</u>

A particular surgeon may be responsible for a specific room during a procedure, but only one surgeon at a time can be assigned to that room for an operation.

A patient can be assigned to only one room at a time, but a room may have had many patients over time.

CleaningService :

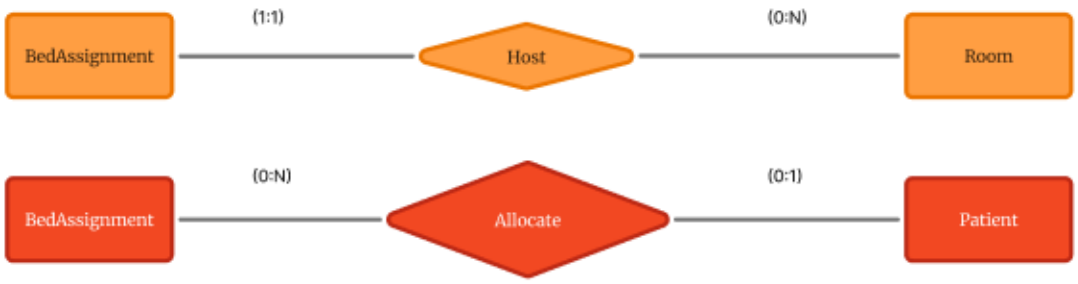


Assumptions:

A team of cleaning services may clean several rooms, or sometimes no rooms if the service was scheduled but not completed.

Every cleaning service is linked to exactly one department, and each department receives many separate cleaning services over time.

BedAssignment :

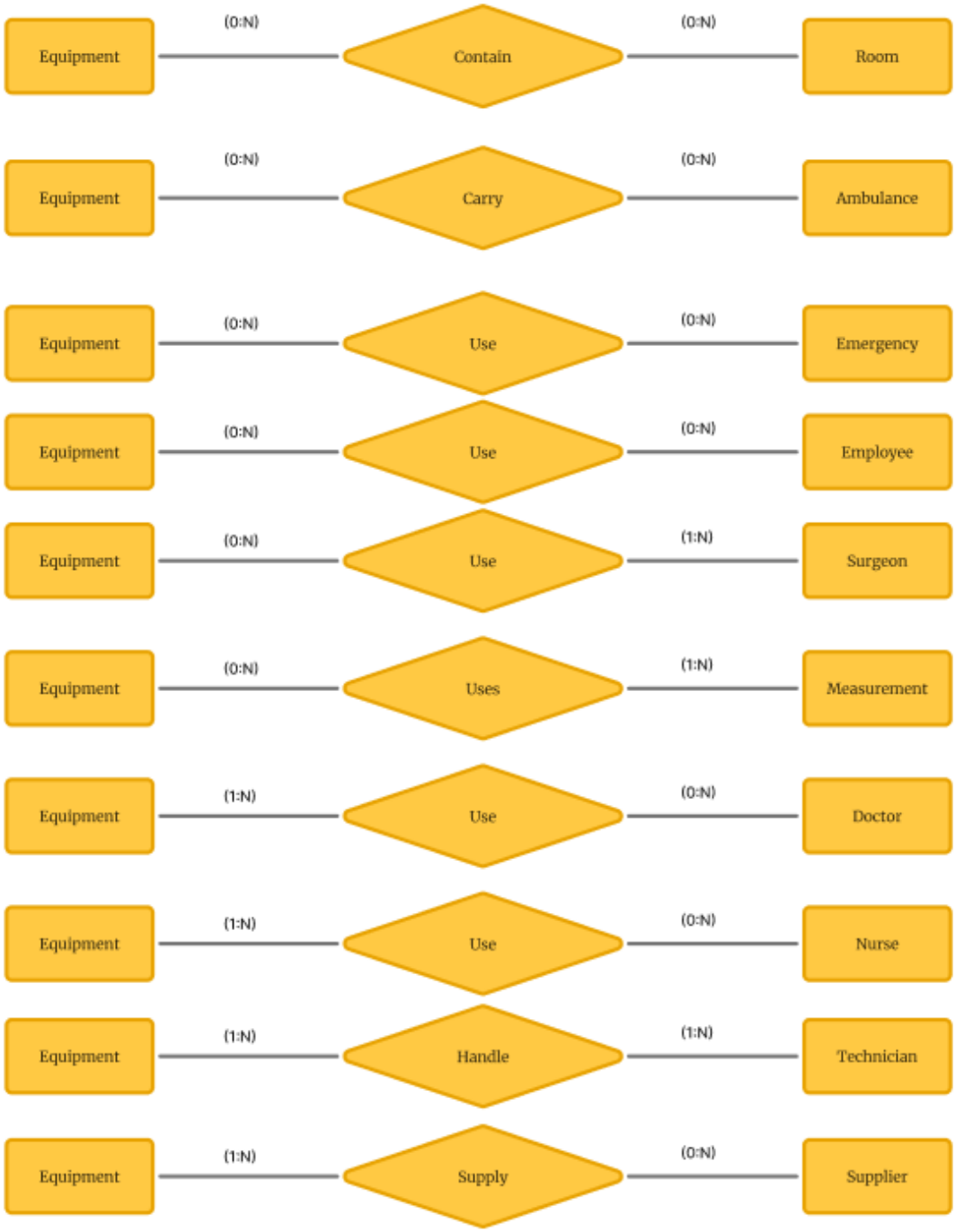


Assumptions:

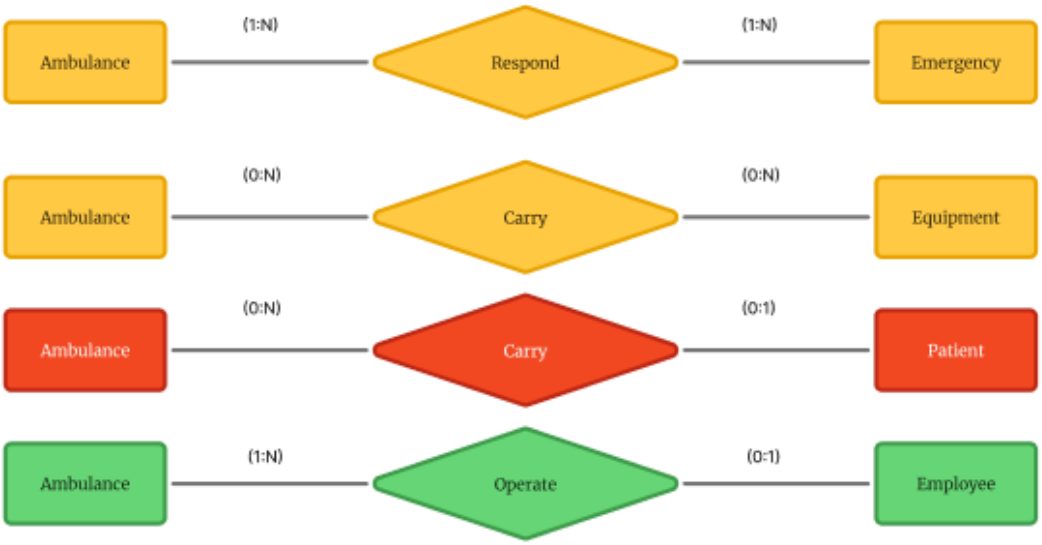
Each bed assignment is linked to one specific room, but a room can host many bed assignments over time.

A patient can receive multiple bed assignments during their stay, but each bed assignment refers to only one patient.

Equipment :


Equipment
(0:N)
Contain
(0:N)
Room
Equipment
(0:N)
Carry
(0:N)
Ambulance
Equipment
(0:N)
Use
(0:N)
Emergency
Equipment
(0:N)
Use
(0:N)
Employee
Equipment
(0:N)
Use
(1:N)
Surgeon
Equipment
(0:N)
Uses
(1:N)
Measurement
Equipment
(1:N)
Use
(0:N)
Doctor
Equipment
(1:N)
Use
(0:N)
Nurse
Equipment
(1:N)
Handle
(1:N)
Technician
Equipment
(1:N)
Supply
(0:N)
Supplier

Ambulance :

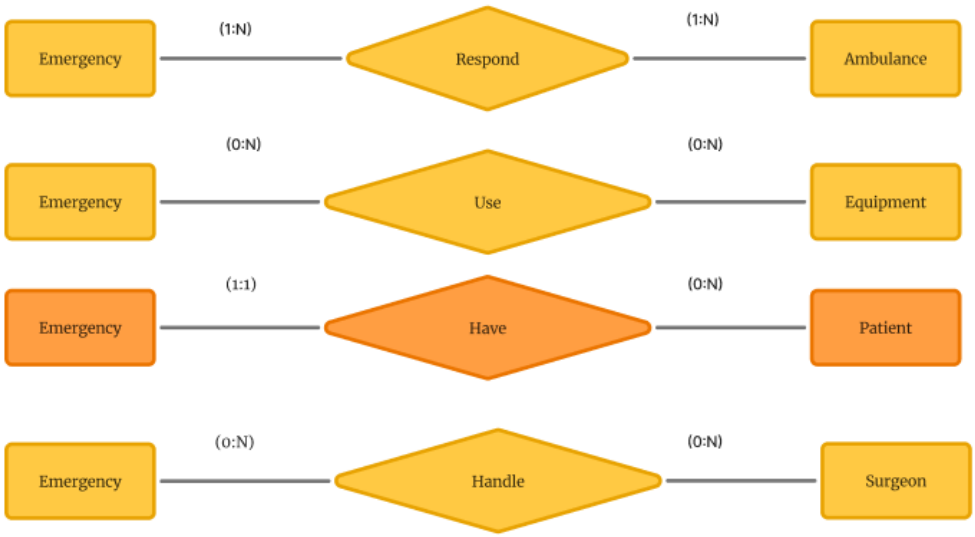


Assumptions:

An ambulance can respond to many emergency calls, and each emergency can require many ambulances.

Each ambulance is operated by one employee, but an employee may operate multiple ambulances over time.

Emergency :

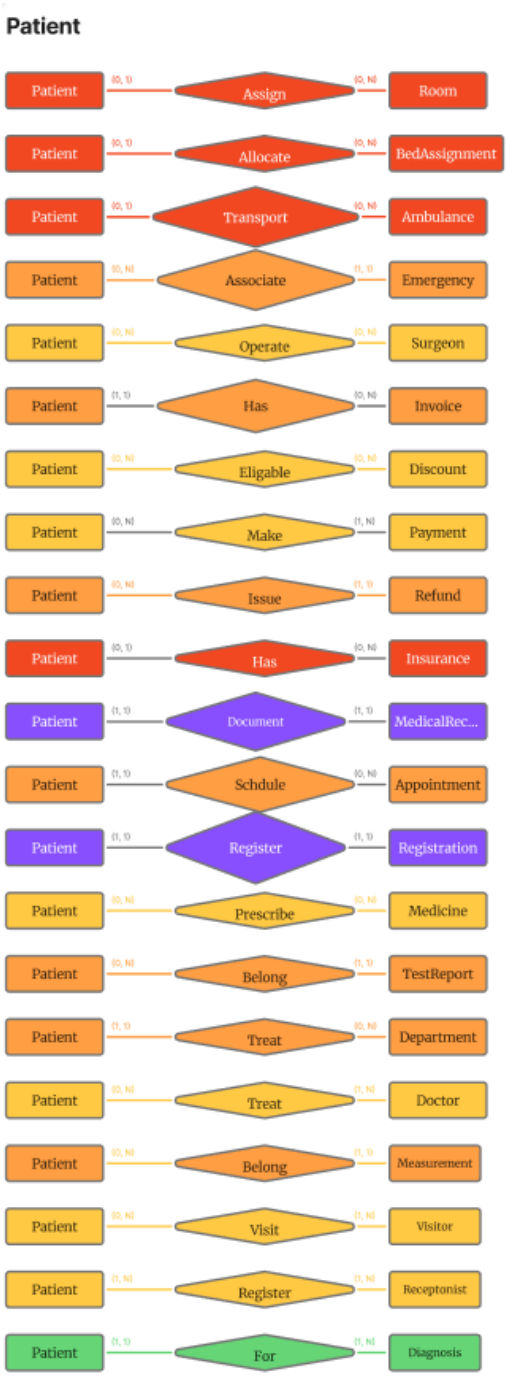


Assumptions:

An emergency may require many ambulances, and each ambulance can respond to many emergencies.

Each emergency involves exactly one patient, but the same patient may have many emergencies over time.

Abdullah:



Assumptions:

A patient can be associated with multiple emergencies, but an emergency can be associated with exactly 1 patient.

A patient can be documented in a single medical record, and a medical record can be for only one patient.

A patient can have 0 or 1 insurance, but an insurance can be associated with many patients.

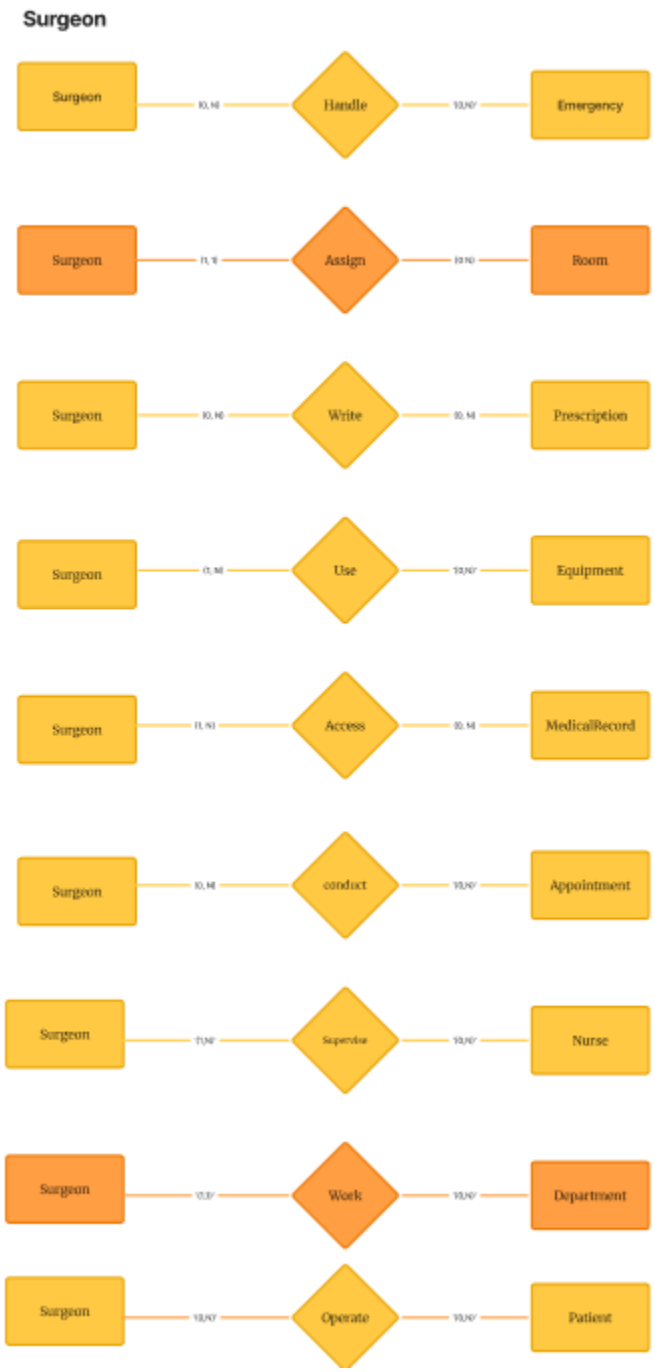
A patient can have a single diagnosis, but a diagnosis can be associated with multiple patients.

Assumptions:

An employee can grant multiple discounts, and a discount be granted by multiple employees.

An employee can operate at most one ambulance, and an ambulance can be operated by at least one employee.

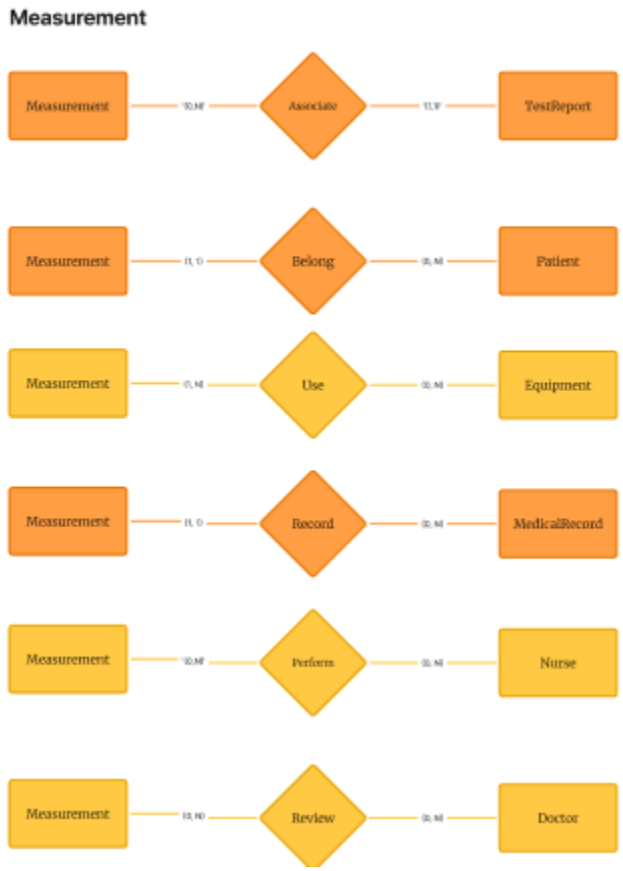
An employee can record zero or more medical records, and a medical record can only be recorded by one employee.



Assumptions:

A surgeon can handle many emergencies, and an emergency can be handled by multiple surgeons.

A surgeon can be assigned to only one room, and a room can be assigned to multiple surgeons.



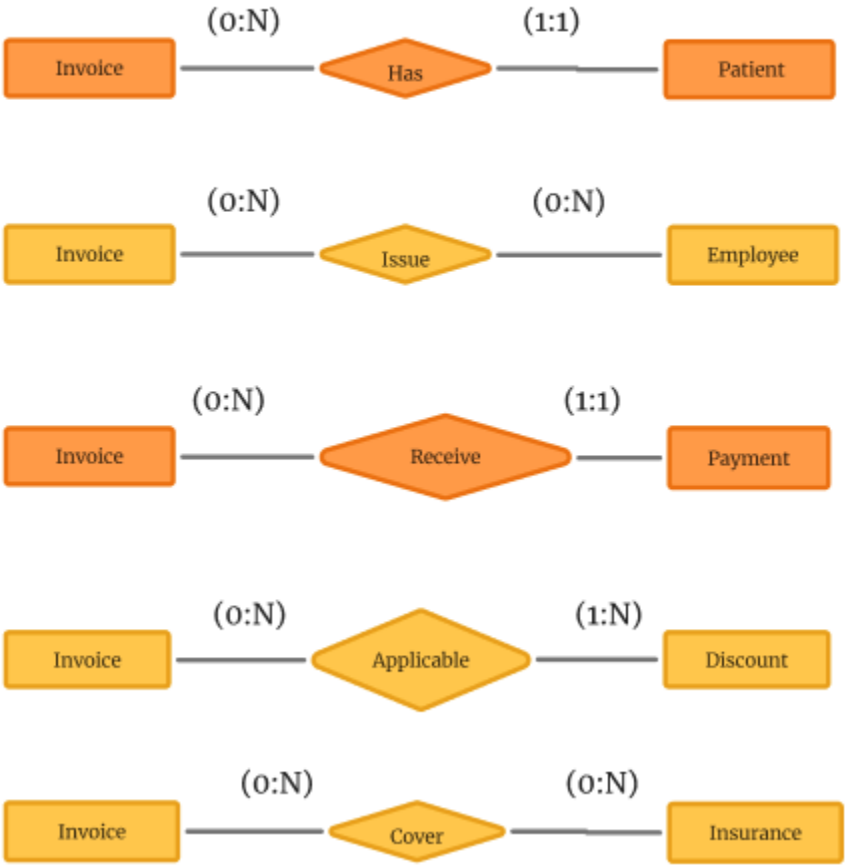
Assumptions:

A measurement can associated with many test reports, and a test report can be associated with only one measurement.

A measurement can use at least one equipment, and an equipment can be used in zero or many measurements.

Amir:

Invoice



Assumptions:

1) One patient can have many invoices, each invoice must be linked to exactly one patient

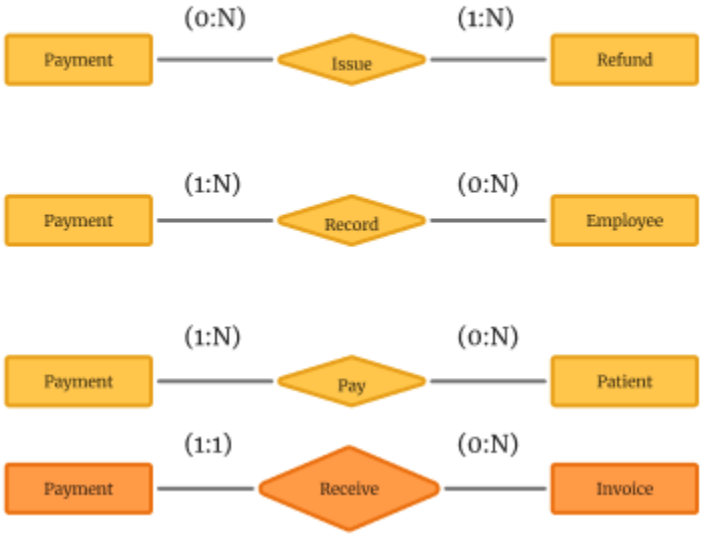
2) Many invoices can be issued by employees, and an employee can issue many invoices (M:N)

3) One payment pays only one invoice, but an invoice may have multiple payments

4) A discount can apply to many invoices, but an invoice may have zero or multiple discounts

5) Many invoices can be covered by many insurance contracts (M:N)

Payment

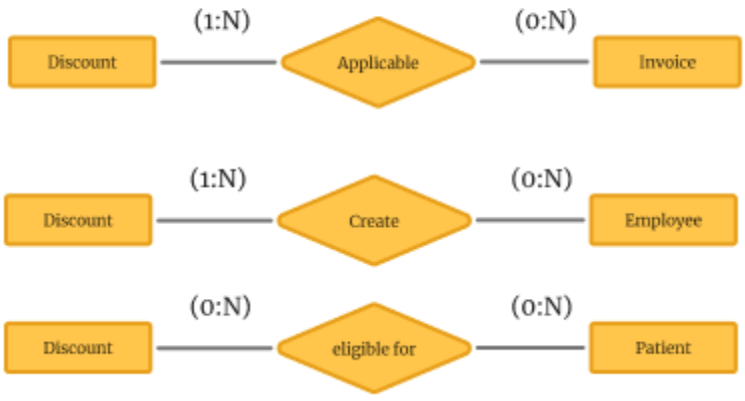


Assumptions:
1) A refund is always linked to one payment, many payment can have multiple refunds

2) An employee can record many payments; a payment must be recorded by an employee

3) A patient may make many payments; a payment must be linked to one patient

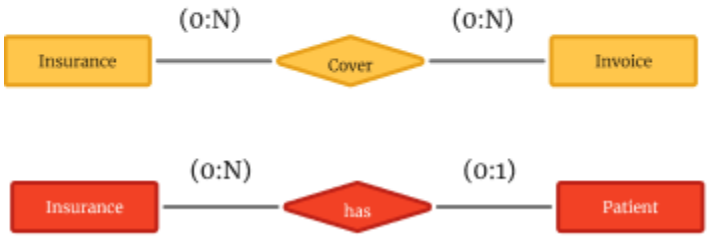
Discount



Assumptions:
1) Same as already shown in Invoice section

2) An employee can create many discounts, discounts always created by employees

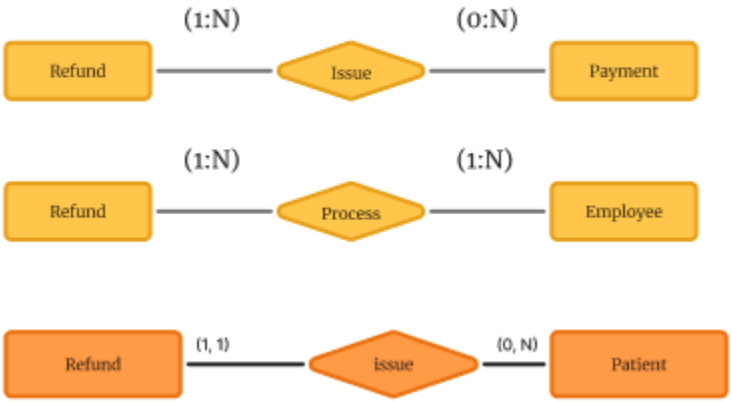
Insurance



Answers:
1) Insurances cover many Invoices; vice versa

2) An insurance company can cover 0 or many patients; an insurance must belong to exactly one patient

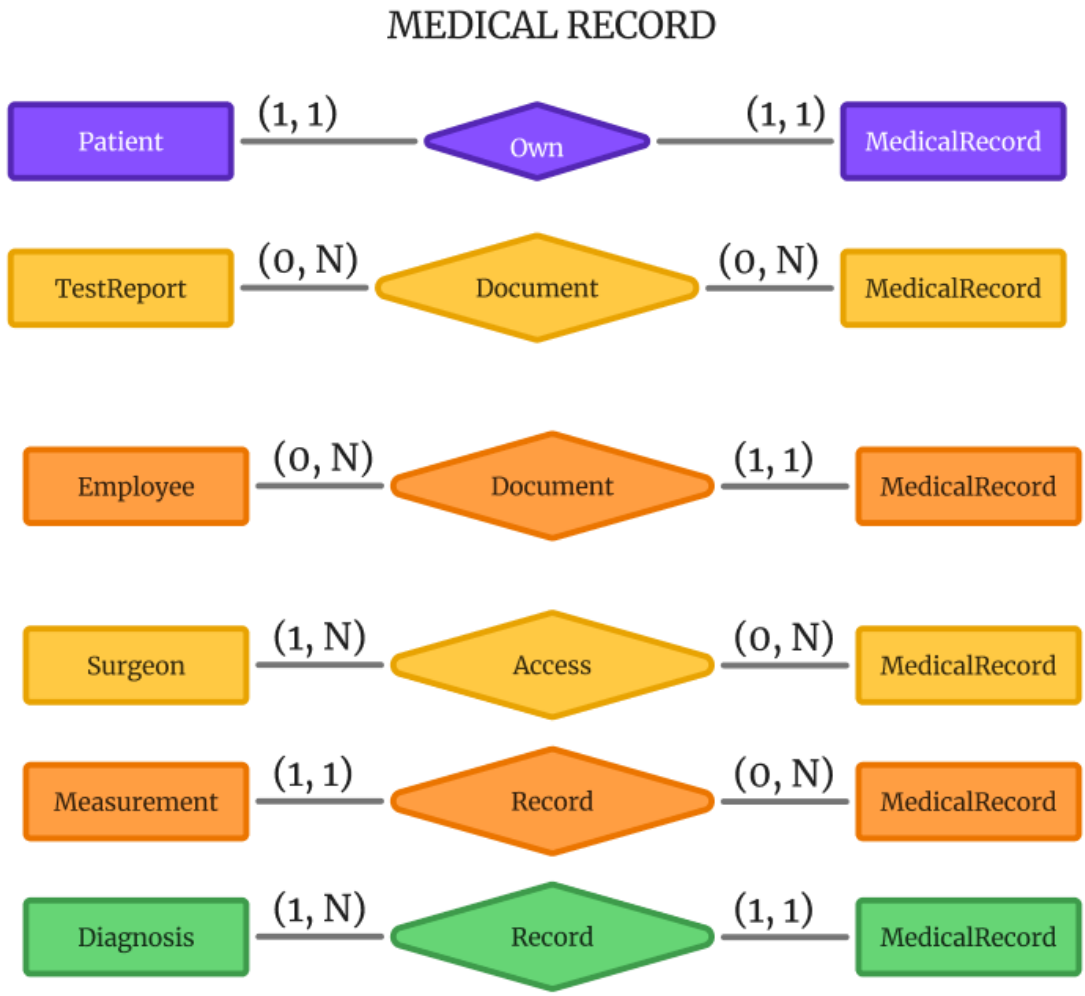
Refund



Answers:
1) Same as shown in Payment section

2) An employee processes many refunds; a refund always processed by employees

Anas:



The Patient owns at least 1 and at most 1 Medical Record and The Medical Record is Owned by at least 1 and at most 1 Patient .

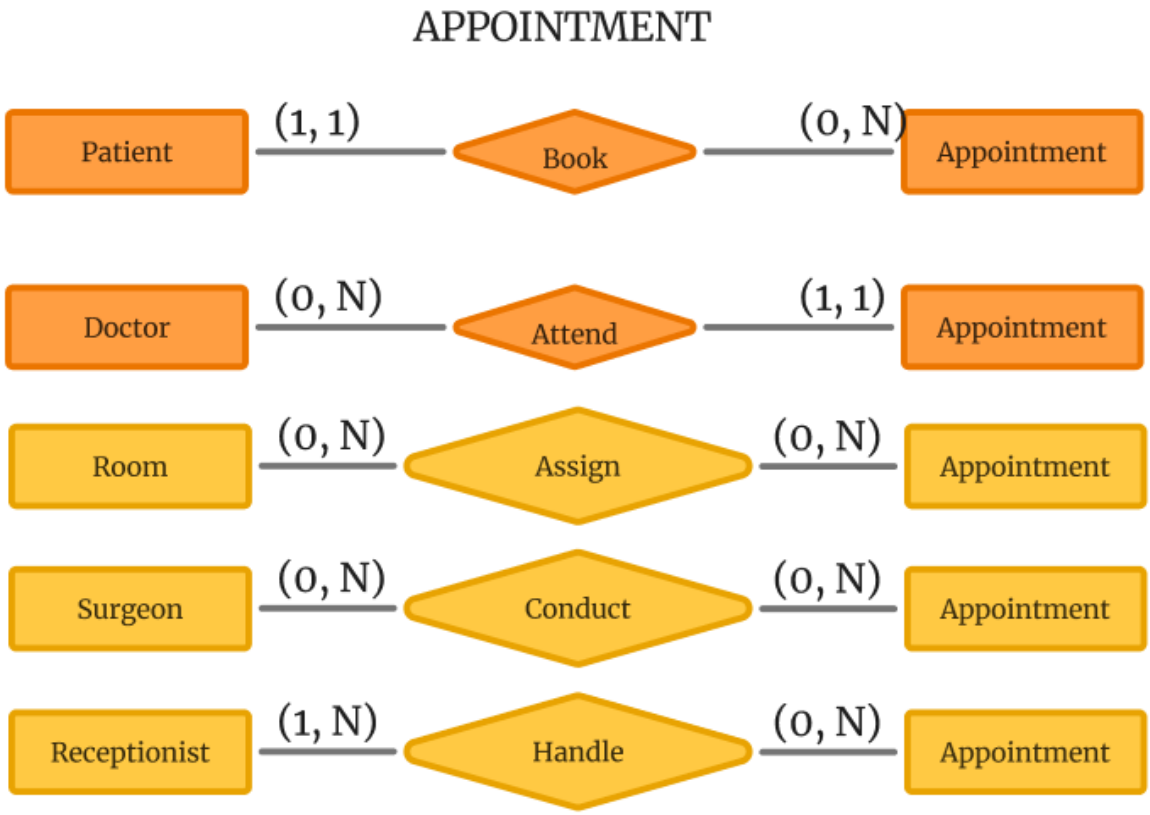
The TestReport documents many MedicalRecord and The MedicalRecord is documented by many TestReports

The Employee documents many MedicalRecord but The MedicalRecord is documented by ONLY 1 Employee.

The Surgeon Accesses at least 1 and at most many MedicalRecords and a MedicalRecord is accessed by many Surgeons.

A MedicalRecord records many measurements and a measurement can be recorded by 1 and only 1 MedicalRecord.

The MedicalRecord records at least 1 and at most 1 Diagnosis and The Diagnosis is recorded by at least 1 and at most many MedicalRecords.



A patient must book exactly one appointment.

An appointment may be booked by zero patients or by many patients..

A doctor may attend zero appointments or may attend many appointments.

An appointment must be attended by exactly one doctor.

A room may be assigned to zero appointments or may be assigned to many appointments.

An appointment may have zero rooms assigned or may have many rooms assigned.

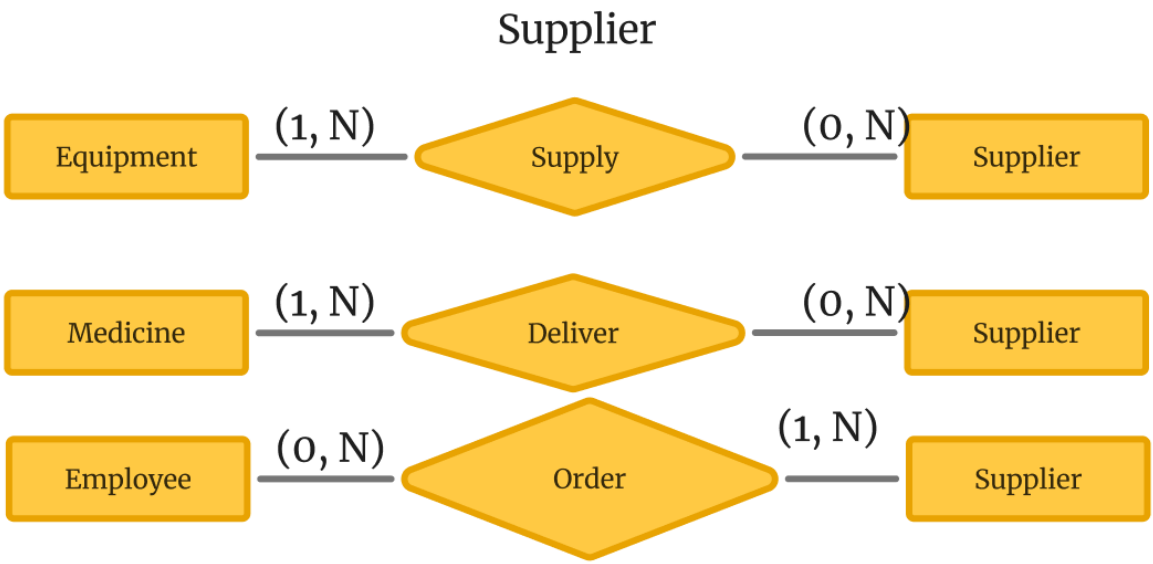
A surgeon may conduct zero appointments or may conduct many appointments.

An appointment may be conducted by zero surgeons or may be conducted by many surgeons.

A receptionist must handle at least one appointment and may handle many appointments.

An appointment may be handled by zero receptionists or may be handled by many receptionists

The Patient books at least 1 and at most 1 Registration and The Registration is booked by at least 1 and at most 1 Patient .



Each piece of Equipment must be supplied by at least one (1) supplier.
Each piece of Equipment can be supplied by many (N) suppliers.

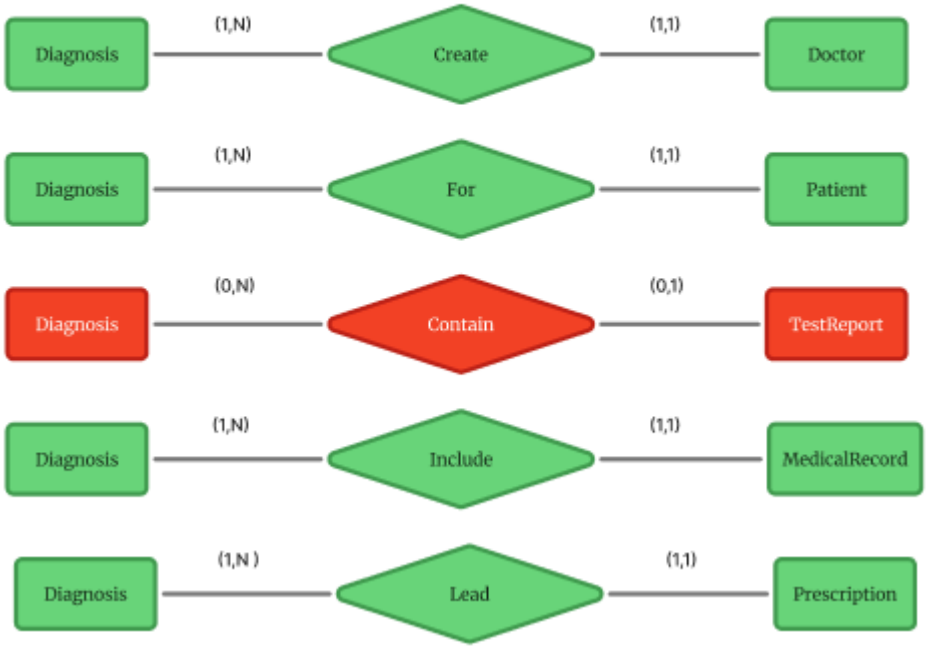
A Supplier may not supply any equipment (0).
A Supplier can supply many (N) pieces of equipment.

Each type of Medicine must be delivered by at least one (1) supplier.
Each type of Medicine can be delivered by many (N) suppliers.
A Supplier may not deliver any medicine (0).
A Supplier can deliver many (N) types of medicine.

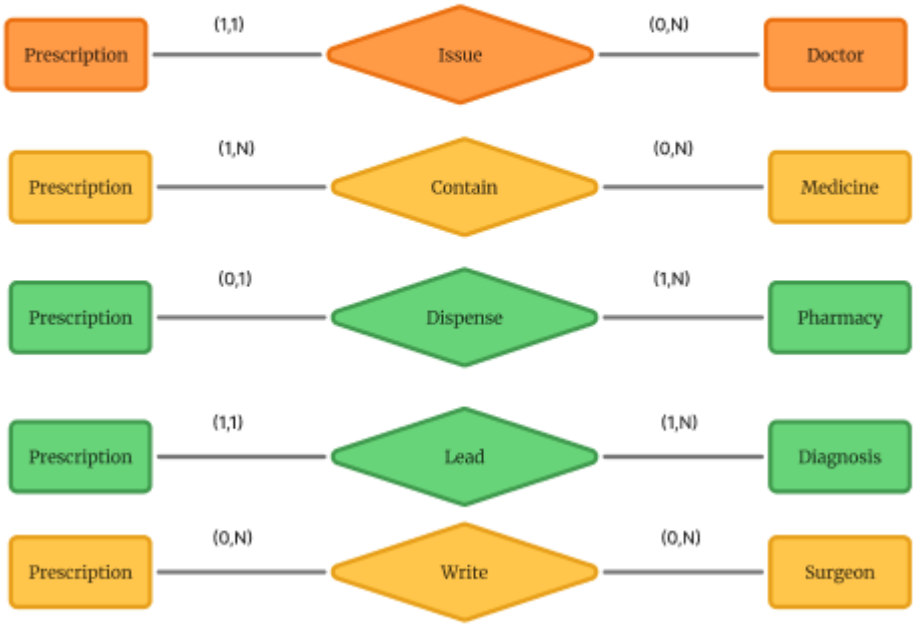
An Employee may not place any orders (0).
An Employee can place many (N) orders.
An order must be placed with at least one (1) supplier. (This implies an order is always associated with a supplier).
A Supplier can receive many (N) order

Mohamed:

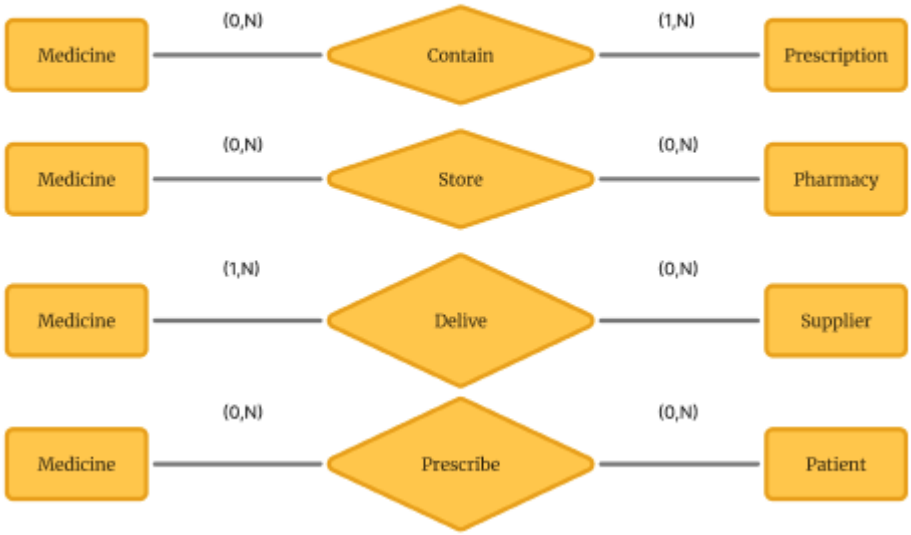
Diagnosis



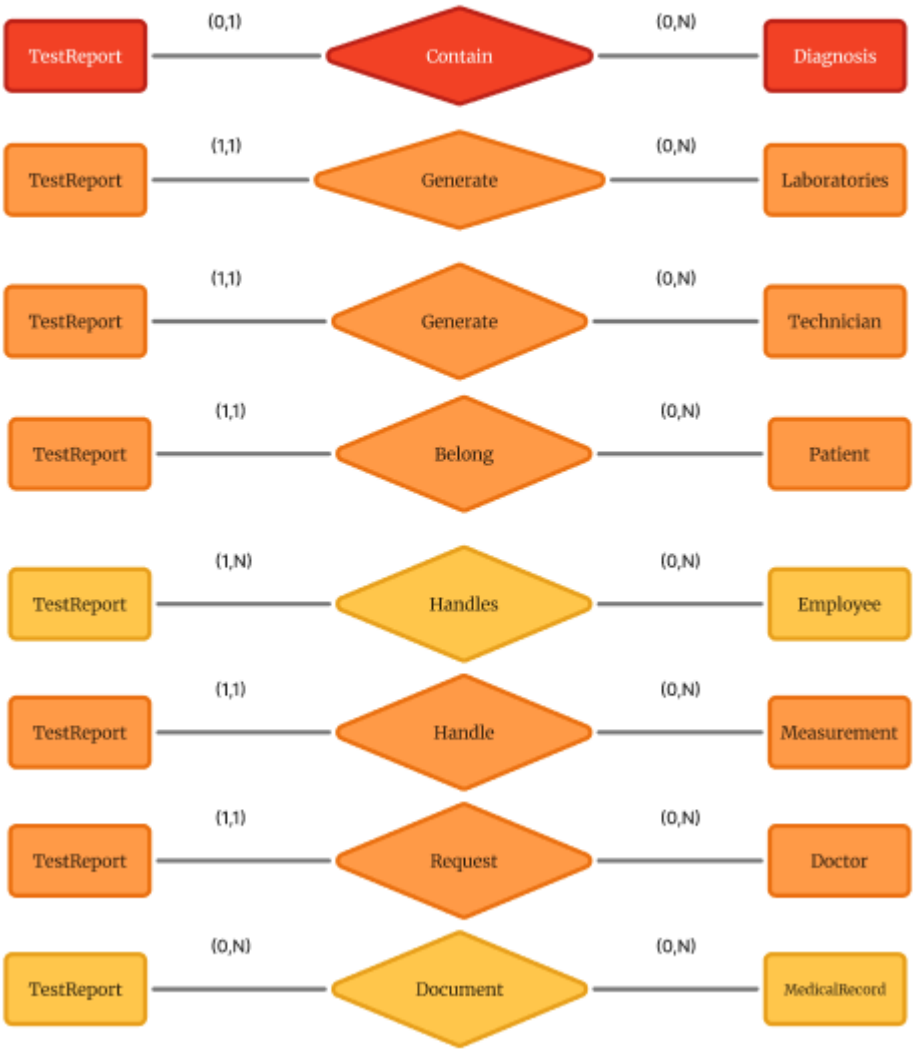
Prescription



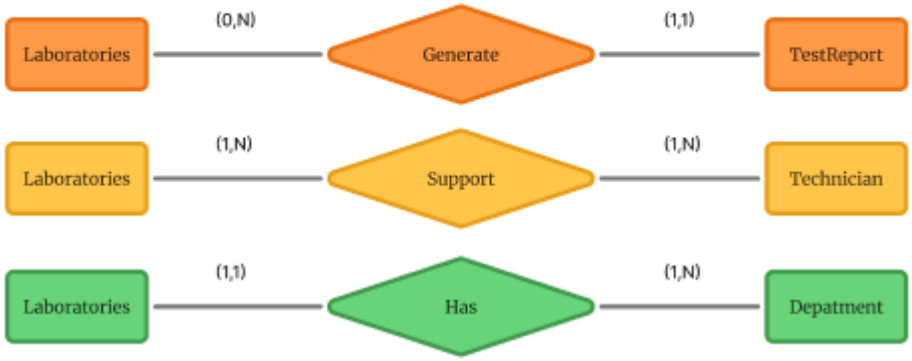
Medicine


Medicine
(0,N)
Contain
(1,N)
Prescription
Medicine
(0,N)
Store
(0,N)
Pharmacy
Medicine
(1,N)
Delive
(0,N)
Supplier
Medicine
(0,N)
Prescribe
(0,N)
Patient

TestReport



Laboratories

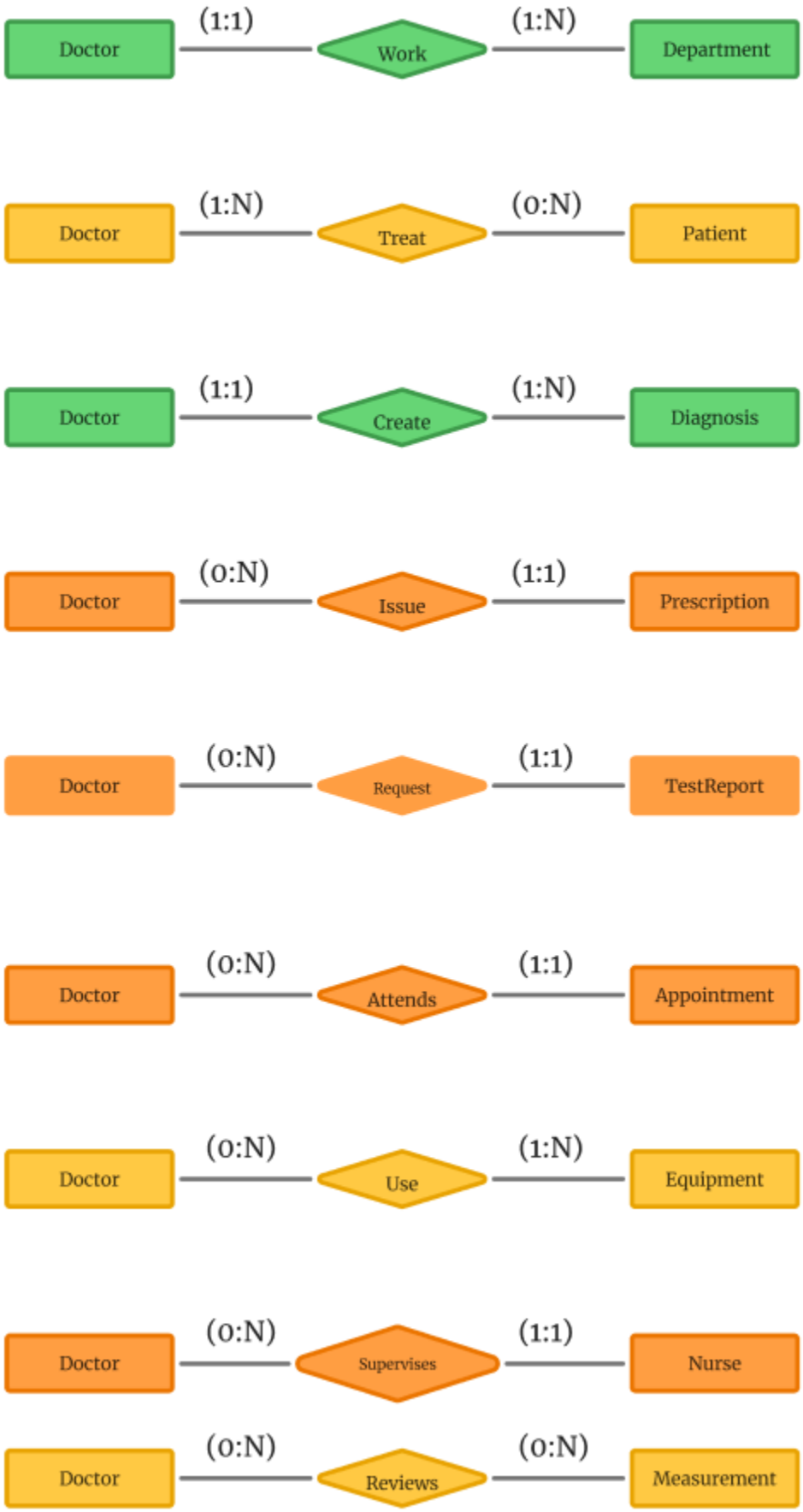


Assumptions:

- Each diagnosis that comes out by 1 doctor is unique for that 1 patient
- A medical record can be clean without a diagnosis, but a diagnosis must be included in a medical record. A medical record is a record of multiple different visits all of which have a diagnosis therefore it is M:N
- A doctor in the hospital does not necessarily need to write a prescription such as a consultant
- Patient may not need prescriptions for certain diagnoses like a sore throat
- A prescription can get canceled by pharmacies
- Medicine can be stored in either the supplier or the pharmacy
- A laboratory can exist without currently generating reports
- Technicians may work in a lab but don't necessarily produce a report daily
- A technician belongs to one primary lab
- There could be more than 1 Pharmacy in the hospital

Khalid:

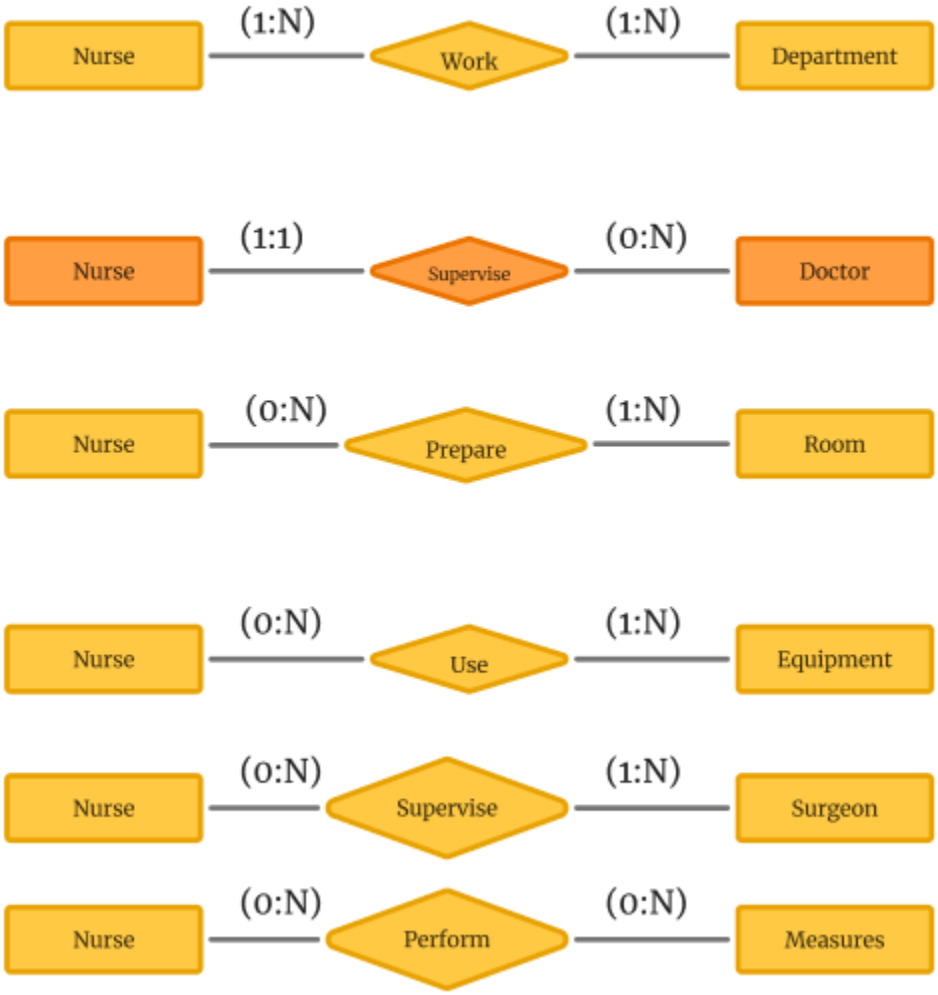
Doctor


Doctor
(1:1)
Work
(1:N)
Department
Doctor
(1:N)
Treat
(0:N)
Patient
Doctor
(1:1)
Create
(1:N)
Diagnosis
Doctor
(0:N)
Issue
(1:1)
Prescription
Doctor
(0:N)
Request
(1:1)
TestReport
Doctor
(0:N)
Attends
(1:1)
Appointment
Doctor
(0:N)
Use
(1:N)
Equipment
Doctor
(0:N)
Supervises
(1:1)
Nurse
Doctor
(0:N)
Reviews
(0:N)
Measurement

Assumptions

A Doctor Works in a single Department and a Department can have multiple Doctors. A Doctor can Treat multiple Patients and a Patient may not have been Treated by a Doctor yet. A Doctor Creates multiple Diagnoses over time and each Diagnosis is Created by a single Doctor. If a Prescription is Issued, it is by a single Doctor. If a TestReport is Requested, it is by a single Doctor. If an Appointment is Attended, it is by a single Doctor. A Doctor Uses Equipment and a piece of Equipment can be Used by multiple Doctors. If a Nurse is Supervised, it is by a single Doctor. A Doctor Reviews Measurements and a Measurement may or may not be Reviewed by a Doctor.

Nurse



Assumptions

A Nurse Works in a single Department and a Department can have multiple Nurses. A Nurse Supervises a single Doctor. A Nurse Prepares Rooms and a Room is Prepared by multiple Nurses. A Nurse Uses Equipment and a piece of Equipment can be Used by multiple Nurses. If a Surgeon is Supervised, it is by a single Nurse. A Nurse Performs Measures and a Measure may or may not be Performed by a Nurse.

Department


Department
(1:N)
Work
(1:1)
Doctor


Department
(1:N)
Work
(1:N)
Nurse


Department
(1:N)
Work
(1:N)
Technician


Department
(1:N)
Has
(1:N)
Receptionist


Department
(1:N)
Has
(1:1)
Laboratories


Department
(1:N)
Clean
(1:1)
CleaningService

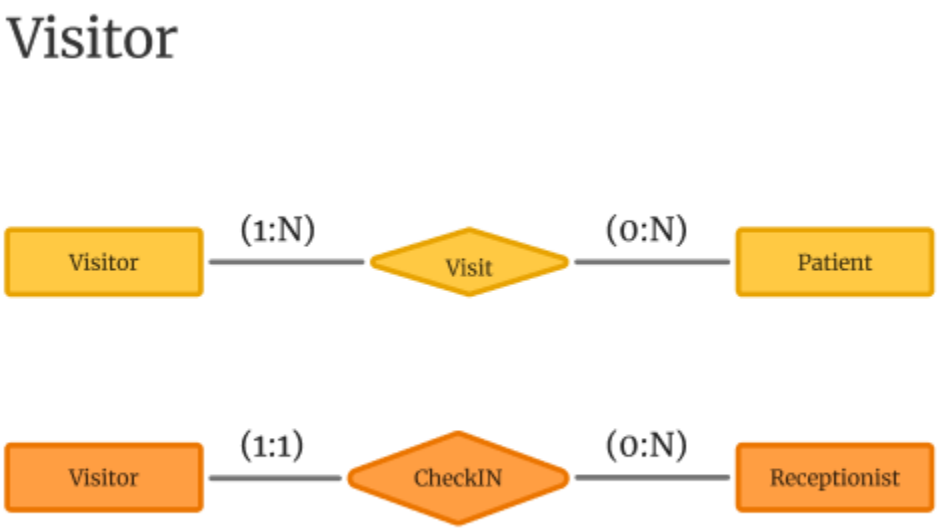

Department
(0:N)
Request
(1:1)
Patient


Department
(1:N)
Work
(1:1)
Employee


Department
(0:N)
Works
(1:1)
Surgeon

Assumptions

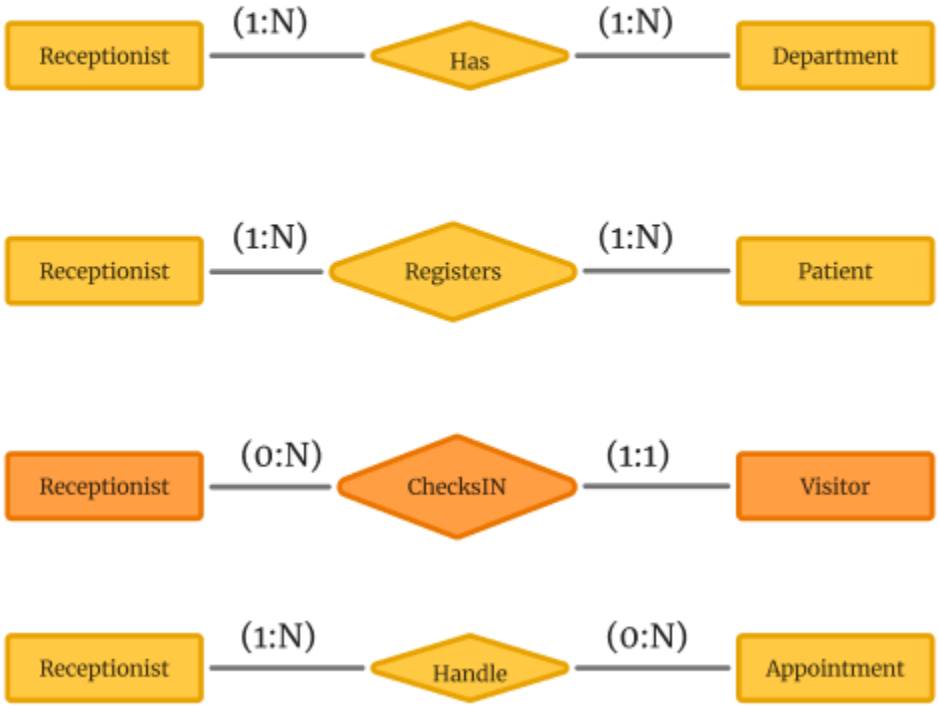
A Doctor Works in a single Department and a Department can have multiple Doctors. A Nurse Works in a single Department and a Department can have multiple Nurses. A Technician Works in a single Department and a Department can have multiple Technicians. A Receptionist Has a single Department and a Department can have multiple Receptionists. A Department Has a single Laboratory and a Laboratory Belongs to a single Department. A Department is Cleaned by a single CleaningService and a CleaningService Cleans a single Department. If a Department Requests a service, it is for a single specific Patient. An Employee Works in a single Department and a Department can have multiple Employees. A Surgeon is assigned to a single Department.



Assumptions

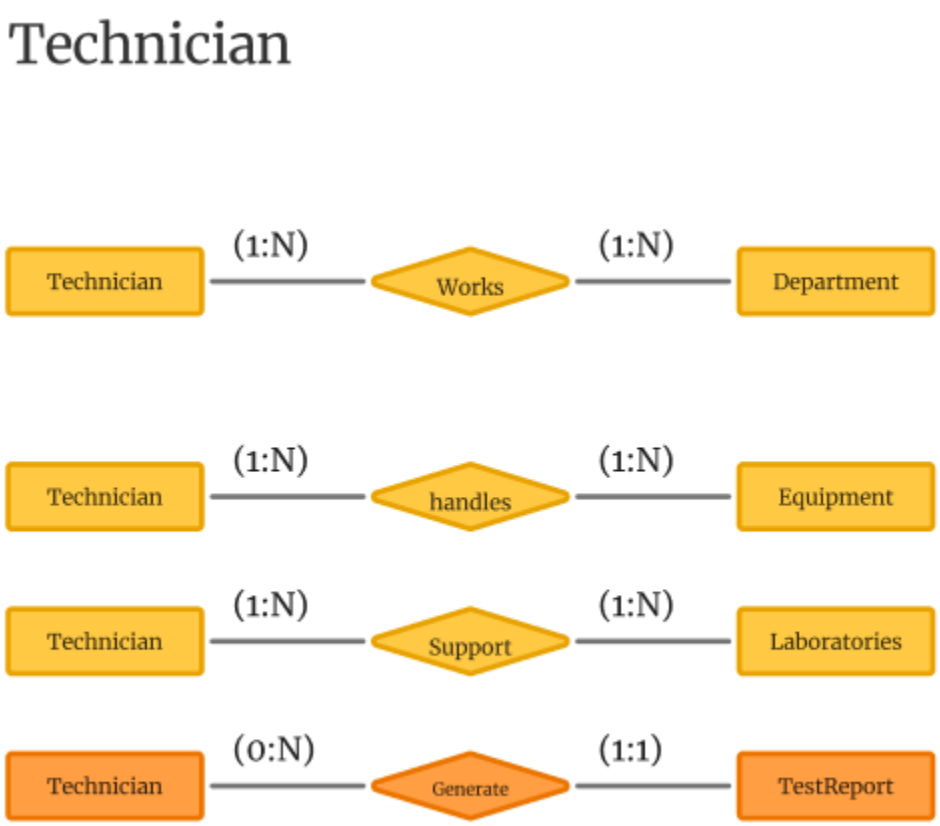
A Visitor can Visit multiple Patients and a Patient may have no Visitors. A Visitor is CheckedIN by a single Receptionist.

Receptionist



Assumptions

A Receptionist Has a single Department and a Department can have multiple Receptionists. A Receptionist Registers multiple Patients over time and a Patient can be Registered by multiple Receptionists. If a Visitor ChecksIN, it is with a single Receptionist. A Receptionist Handles multiple Appointments over time, and an Appointment may not be Handled by a Receptionist.



Assumptions

A Technician works in a Department and a Department has multiple Technicians. A Technician handles a single piece of Equipment and a piece of Equipment can be handled by multiple Technicians. A Technician provides Support to a single Laboratory and a Laboratory can have multiple Technicians for support. If a TestReport is Generated, it is done by a single Technician.

3.- Relational Model :

Color code :

T= Total Participation

- Means
  1-N(T) & 1(T)-N(T) Foreign Key approach, PK of 1 on N table
- Means
  (T) 1-N Foreign Key approach, pk of N in table of 1
- Means
  1-N Relationship Relation
- Means
  N-N Relationship relation
- Means
  1(T)-1(T) Merge Tables
- Means
  1-1(T) Foreign Key approach, PK of side without (T) into table of 1

Adam :

CleaningService

CleaningServiceID	ServiceDateTime	CleaningType	Status	Duration	SuppliesUsed	RoomID

Emergency

EmergencyID	ArrivalDateTime	EmergencyType	SeverityLevel	TreatmentGiven	Outcome	PatientID

Room

RoomID	RoomType	MaxOccupancy	RoomNumber	Status	CostPerDay	RoomPhoneNumber

BedAssignment

AssignmentID	AssignmentStatus	AdmissionDateTime	DischargeDateTime	RoomID

Equipment

EquipmentID	EquipmentName	EquipmentType	SerialNumber	ModelNumber	PurchaseDate	LastMaintenanceDate	NextMaintenanceDue	EquipmentStatus

Ambulance

AmbulanceID	PlateNumber	DriverID	AmbulanceType	Status	Location	LastMaintenanceDate	Capacity	EmployeeID

RoomEquipment

RoomID	EquipmentID

RoomPatient

PatientID	RoomID

RoomNurse

RoomID	NurseID

RoomAppointment

RoomID	AppointmentID

EquipmentSurgeon

EquipmentID	SurgeonID

EquipmentMeasurement

EquipmentID	MeasurementID

EquipmentDoctor

EquipmentID	DoctorID

EquipmentNurse

EquipmentID	NurseID

EquipmentAmbulance

EquipmentID	AmbulanceID

EquipmentEmployee

EquipmentID	EmployeeID

PatientAmbulance

PatientID	AmbulanceID

EmergencyEquipment

EmergencyID	EquipmentID

EmergencySurgeon

EmergencyID	SurgeonID

EquipmentTechnician

EquipmentID	TechnicianID

EquipmentSupplier

EquipmentID	SupplierID

AmbulanceEmergency

AmbulanceID	EmergencyID

PatientBedAssignment

PatientID	BedAssignmentID

Abdullah:

Measurement

MeasurementID	Status	MeasurementTime	Value	MeasurementType	PatientID	MedicalRecordID

EmployeeSupplier

EmployeeID	SupplierID

PatientSurgeon

PatientID	SurgeonID

SurgeonMedicalRecord

SurgeonID	MedicalRecordID

SurgeonAppointment

SurgeonID	AppointmentID

SurgeonAppointment

SurgeonID	PatientID

Employee

EmployeeID	FirstName	LastName	JobTitle	Department	Salary	DateHired	EmployementStatus	Address

Surgeon

SurgeonID	FirstName	LastName	Specialization	LicenseNumber	OperatingRoom	AvailabilityStatus	Qualifications	YearsofExperience

The following is a single table (but it's a bit too long, so it split into smaller images)

Patient

PatientID	FirstName	LastName	DOB	Gender	ContactNumber	BloodType	Address

RegistrationID	RegDay	RegMonth	RegYear	Type	FirstName	LastName	DOB	Gender	ContactNumber

BloodType	Address	RecordID	Notes	RDay	RMonth	RYear	FirstName	LastName	DOB

Gender	ContactNumber	BloodType	Address	RecordID	Notes	RDay	RMonth	RYear	FirstName

Amir:

Refund

Insurance

Discount

Payment

Invoice

InvoiceDiscount

PaymentEmployee

InvoiceInsurance

PaymentPatient

InvoiceEmployee

EmployeeDiscount

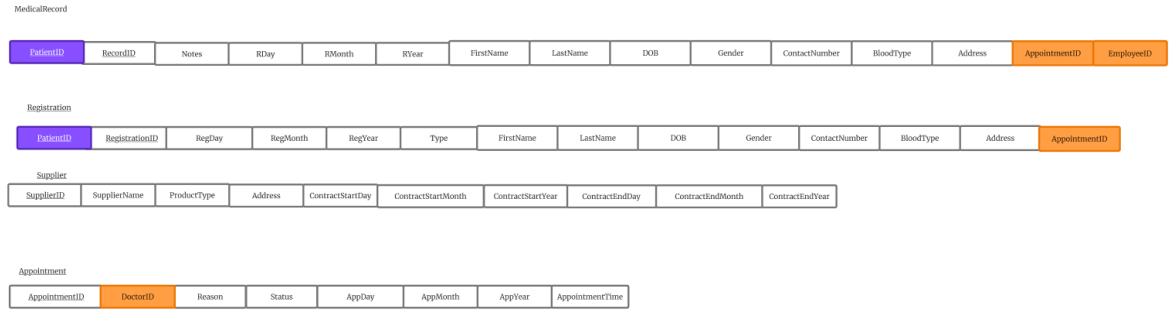
PaymentRefund

PatientDiscount

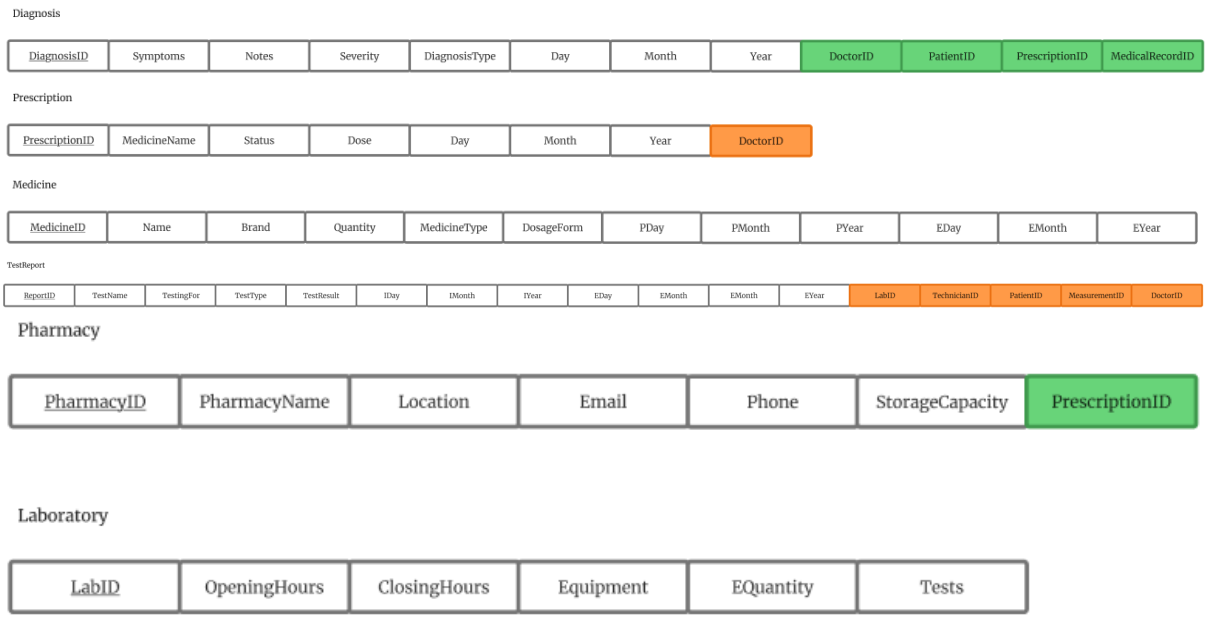
EmployeeRefund

PatientInsurance

Anas:



Mohamed:



DiagnosisTestReport

DiagnosisID	ReportID

MedicinePrescription

PrescriptionID	MedicineID

SurgeonPrescription

SurgeonID	PrescriptionID

PharmacyMedicine

PharmacyID	MedicineID

SupplierMedicine

SupplierID	MedicineID

PatientMedicine

PatientID	MedicineID

EmployeeTestReport

EmployeeID	ReportID

MedicalRecordTestReport

RecordID	ReportID

LaboratoryTechnician

TechnicianID	LaboratoryID

Khalid:

Doctor

DoctorID	Specialization	PhoneNumber	YearsOfExperien...	Email	FirstName	LastName	WorkingTime

Nurse

NurseID	Qualification	ShiftHours	YearsOfExperien...	FirstName	LastName	ContactNumber	DoctorID

Department

DepartmentID	NumberOfStaff	Location	ContactNumber	DepartmentName	DoctorID	LabID	CleaningServiceID	EmployeeID

Visitor

VisitorID	PhoneNumber	UAE_ID	RelationToPatient	Name	Day	Month	Year	ReceptionistID

Receptionist

ReceptionistID	ShiftTiming	PhoneNumber	Email	FirstName	LastName

Technician

TechnicianID	Specialization	PhoneNumber	WorkSchedule	FirstName	LastName

Doctor treats Patient

DoctorID	PatientID

Doctor reviews Measurement

DoctorID	MeasurementID

Nurse Supervise Surgeon

NurseID	SurgeonID

Nurse Perform Measures

NurseID	MeasurementID

Nurse work Department

NurseID	DepartmentID

Department work Technician

DepartmentID	TechnicianID

Department has Receptionist

DepartmentID	ReceptionistID

VisitorVisitsPatient

VisitorID	PatientID

Receptionist has Department

ReceptionistID	DepartmentID

Receptionist handle Appointment

ReceptionistID	AppointmentID

Technician works department

TechnicianID	DepartmentID

Technician handles Equipment

TechnicianID	EquipmentID

Technician Support Laboratories

TechnicianID	LabID

Receptionist registers Patient

ReceptionistID	PatientID

4.- Normalization :

Adam :

Normalization (1NF): *All relations satisfy 1NF since every attribute holds only one value, and no table includes multi-valued fields. All data is atomic.*

Normalization (2NF): *2NF is automatically met because every table uses a primary key made up of a single attribute.*

Normalization (3NF): *In 3NF, each entity is examined for transitive dependencies to ensure. All attributes should depend directly on the primary key.*

Tables after normalization :

CleaningService

CleaningServiceID	ServiceDateTime	CleaningType	Status	Duration	SuppliesUsed	RoomID

Emergency

EmergencyID	ArrivalDateTime	EmergencyType	SeverityLevel	TreatmentGiven	Outcome	PatientID

Room

RoomID	RoomType	RoomNumber	Status	RoomPhoneNumber

RoomTypeMaxOccupancyCostPerDay

RoomType	MaxOccupancy	CostPerDay

BedAssignment

AssignmentID	AssignmentStatus	AdmissionDateTime	DischargeDateTime	RoomID

Equipment

EquipmentID	EquipmentName	EquipmentType	SerialNumber	ModelNumber	PurchaseDate	LastMaintenanceDate	NextMaintenanceDue	EquipmentStatus

Ambulance

AmbulanceID	PlateNumber	DriverID	AmbulanceType	Status	Location	LastMaintenanceDate	EmployeeID

AmbulanceTypeCapacity

AmbulanceType	Capacity

RoomEquipment

RoomPatient

RoomNurse

RoomAppointment

EquipmentAmbulance

EquipmentEmployee

PatientAmbulance

EmergencyEquipment

EmergencySurgeon

EquipmentSurgeon

EquipmentMeasurement

EquipmentDoctor

EquipmentNurse

EquipmentTechnician

EquipmentSupplier

AmbulanceEmergency

PatientBedAssignment

Abdullah:

Normalization (1NF):

In employee table: changed `Address` into its atomic equivalent.

In surgeon table: Removed `Qualifications` from the table since `Qualifications` is multivalued.

Normalization (2NF): *The Patient table qualifies for this normalization but it is normalized in a different place, done by Anas*

Normalization (3NF): *Not needed since in 3NF, each entity is examined for transitive dependencies to ensure*

all attributes depend directly on the primary key. Meaning there were no transitive dependency

The Patient table qualifies for this normalization but it is normalized in a different place, done by Anas

Tables after normalization :

Measurement

MeasurementID	Status	MeasurementTime	Value	MeasurementType	PatientID	MedicalRecordID

EmployeeSupplier

EmployeeID	SupplierID

PatientSurgeon

PatientID	SurgeonID

SurgeonMedicalRecord

SurgeonID	MedicalRecordID

SurgeonAppointment

SurgeonID	AppointmentID

SurgeonAppointment

SurgeonID	PatientID

Employee

EmployeeID	FirstName	LastName	JobTitle	Department	Salary	DateHired	EmployementStatus	Country	City	Area	Street

Surgeon

SurgeonID	FirstName	LastName	Specialization	LicenseNumber	OperatingRoom	AvailabilityStatus	YearsofExperience

SurgeonID	Qualifications

Patient

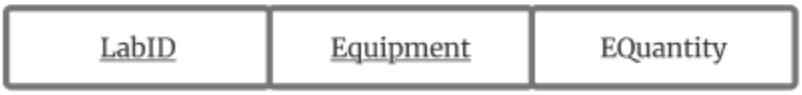
PatientID	FirstName	LastName	DOB	Gender	ContactNumber	BloodType	Address	AppointmentID

Amir:

Normalization (1NF):

Prescription

Laboratory



Normalization (2NF): This condition is inherently satisfied because all entities in this design possess a single-attribute primary key. Partial dependencies can only occur when a primary key is composite (made of multiple attributes); since no such keys exist, no partial dependencies are possible

Normalization (3NF)

TestNameTestingForTestType

TestName	TestingFor	TestType

Table After Normalization

Anas:

1NF:

All Tables are Already In 1NF as all values are atomic.

{PatientID, RegistrationID} → RegDay
{PatientID, RegistrationID} → RegMonth
{PatientID, RegistrationID} → RegYear
{PatientID, RegistrationID} → Type
{PatientID, RegistrationID} → FirstName
{PatientID, RegistrationID} → LastName
{PatientID, RegistrationID} → DOB
{PatientID, RegistrationID} → Gender
{PatientID, RegistrationID} → ContactNumber
{PatientID, RegistrationID} → BloodType
{PatientID, RegistrationID} → Address
{PatientID, RegistrationID} → AppointmentID

RegistrationID → RegDay
RegistrationID → RegMonth
RegistrationID → RegYear
RegistrationID → Type

PatientID → FirstName
PatientID → LastName
PatientID → DOB
PatientID → Gender
PatientID → ContactNumber
PatientID → BloodType
PatientID → Address
PatientID → AppointmentID

2NF:

MedicalRecord

PatientID	RecordID	Notes	RDay	RMonth	RYear	EmployeeID

Patient

PatientID	FirstName	LastName	DOB	Gender	ContactNumber	BloodType	Address	AppointmentID

The Medical Record is Now in 2NF as all attributes that depend on PatientID are separate and they belong to the WHOLE primary key

Registration

PatientID	RegistrationID	RegDay	RegMonth	RegYear	Type

We Remove the attributes of Patient and since we have a patient table already we dont need to remake another one. Registration table is now is 2NF as the attributes depend on the full primary key

Supplier

SupplierID	SupplierName	ProductType	Address	ContractStartDay	ContractStartMonth	ContractStartYear	ContractEndDay	ContractEndMonth	ContractEndYear

The table is already in 2NF as all attributes are dependent on the Full primary key

Appointment

AppointmentID	DoctorID	Reason	Status	AppDay	AppMonth	AppYear	AppointmentTime

The table is already in 2NF as all attributes are dependent on the Full primary key

3NF:

RegistrationID → RegDay

RegistrationID → RegMonth

RegistrationID → RegYear

RegistrationID → Type

PatientID → FirstName

PatientID → LastName

PatientID → DOB

PatientID → Gender

PatientID → ContactNumber

PatientID → BloodType

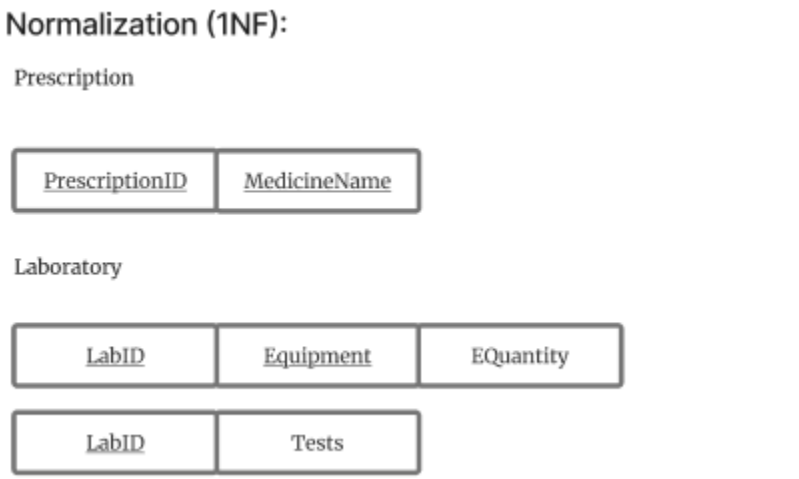
PatientID → Address

PatientID → AppointmentID

ALL tables are already in 3NF as they don't have transitive dependencies

Table After Normalization:

Mohamed:



Normalization (2NF): This condition is inherently satisfied because all entities in this design possess a single-attribute primary key. Partial dependencies can only occur when a primary key is composite (made of multiple attributes); since no such keys exist, no partial dependencies are possible

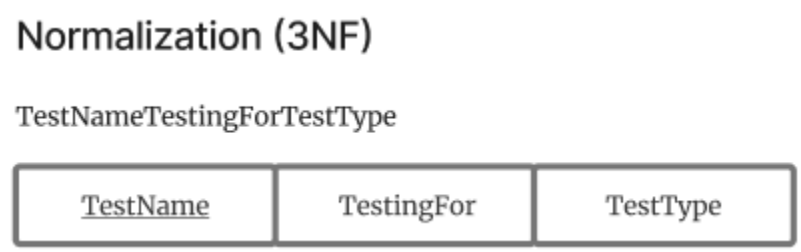


Table After Normalization

Khalid:

Normalization (1NF):

Nurse Qualifications (multi value)

NurseID	Qualification

Doctor Specialization (multi value)

DoctorID	Specialization

Technician Specialization (multi value)

TechnicianID	Specialization

Normalization (2NF): This normalization is already done because all entities in this design possess a single primary key. Partial dependencies can only happens when a primary key is composite (made of multi attributes), no such keys exist, so partial dependencies are not possible.

Normalization (3NF): No transitive dependencies from all entities, they all depend on the pk only, so 3NF is already done for those 6 entities in the design.

Tables after normalization :

Doctor

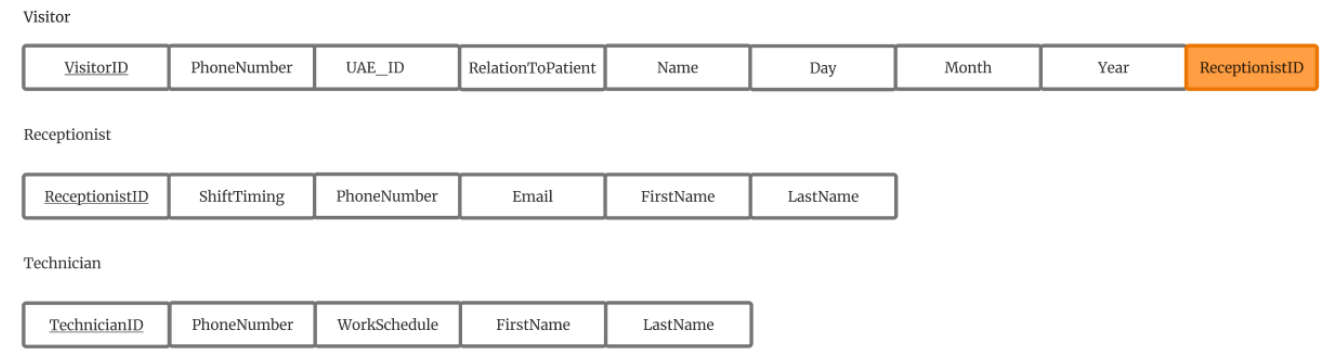
DoctorID	PhoneNumber	YearsOfExperien...	Email	FirstName	LastName	WorkingTime

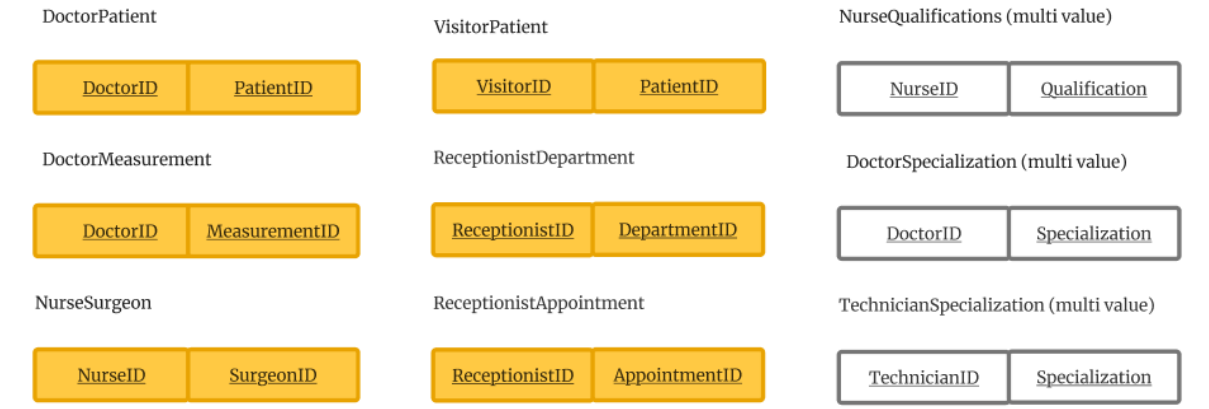
Nurse

NurseID	ShiftHours	YearsOfExperien...	FirstName	LastName	ContactNumber	DoctorID

Department

DepartmentID	NumberOfStaff	Location	ContactNumber	DepartmentName	DoctorID	LabID	CleaningServiceID	EmployeeID





5.- SQL Implementation:

*For the implementation phase, we used **MySQL** as our Relational Database Management System (RDBMS). The implementation process involved translating our Relational Schema into actual SQL code, ensuring that the database structure was efficient, and capable of handling complex hospital workflows.*

Key Technical Achievements:

- ***Schema Creation (DDL):*** *We executed CREATE TABLE statements for over* ***30 entities****, covering four major operational clusters: Infrastructure, Staff, Clinical, and Billing. Each table was defined with appropriate data types (e.g., DECIMAL for financials, DATE for schedules, VARCHAR for text).*
- ***Data Integrity & Constraints:*** *To ensure the reliability of the data, we enforced strict constraints at the schema level:*
    - ***Primary Keys (PK):*** *Every entity has a unique identifier (e.g., PatientID, InvoiceID) to ensure row uniqueness.*
    - ***Foreign Keys (FK):*** *Relationships were enforced using FK constraints to maintain referential integrity (e.g., an Appointment cannot exist without a valid DoctorID and PatientID).*
    - ***Business Rules (CHECK Constraints):*** *We implemented logic directly into the database to prevent invalid data entry. Examples include CHECK (Salary >= 0) to prevent negative payroll errors and CHECK (Month BETWEEN 1 AND 12) to ensure valid date entries.*
- ***Handling Circular Dependencies:*** *A significant challenge encountered was the circular dependency between the Patient and Appointment tables (a Patient needs an Appointment to register, but an Appointment needs a Patient ID). We resolved this by temporarily disabling foreign key checks (SET FOREIGN_KEY_CHECKS = 0) during the schema creation phase and re-enabling them immediately after to ensure structural integrity.*
- ***Data Population (DML):*** *We populated the database with a robust set of dummy data, including 20+ patient records, staff profiles, and simulated clinical workflows (Diagnosis → Prescription → Invoice), proving that the system can successfully handle end-to-end hospital operations.*

```
-- CSCI 326 Project: Hospital Management System Database

-- Implemented by: [Amir Beshir,Adam El Moudenne,Anas Qaisar, Abdullah Farah, Khalid Hafiz, Mohammed Abousaada]


-- Disable foreign key checks to prevent errors during table creation due to circular dependencies

SET FOREIGN_KEY_CHECKS = 0;


-- Create the Database

CREATE DATABASE IF NOT EXISTS HospitalDB;
```

```
USE HospitalDB;

-- ============================================
-- 1. ADMINISTRATIVE & BILLING
-- ============================================

CREATE TABLE Refund (
    RefundID INT PRIMARY KEY,
    PatientID INT,
    Reason VARCHAR(255),
    RefundStatus VARCHAR(50),
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT CHECK (Year > 1900),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE Insurance (
    InsuranceID INT PRIMARY KEY,
    Provider VARCHAR(255),
    CoverageDetails TEXT,
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT CHECK (Year > 1900)
);
```

```
CREATE TABLE Discount (
    DiscountID INT PRIMARY KEY,
    Description VARCHAR(255),
    DiscountCode VARCHAR(50) UNIQUE, -- Business Rule: Codes must be unique
    SDay INT CHECK (SDay BETWEEN 1 AND 31),
    SMonth INT CHECK (SMonth BETWEEN 1 AND 12),
    SYear INT,
    EDay INT CHECK (EDay BETWEEN 1 AND 31),
    EMonth INT CHECK (EMonth BETWEEN 1 AND 12),
    EYear INT
);

CREATE TABLE Payment (
    PaymentID INT PRIMARY KEY,
    InvoiceID INT,
    Method VARCHAR(50),
    Amount DECIMAL(10, 2) CHECK (Amount >= 0), -- Business Rule: No negative payments
    PaymentStatus VARCHAR(50),
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT,
    FOREIGN KEY (InvoiceID) REFERENCES Invoice(InvoiceID)
);
```

```
CREATE TABLE Invoice (
    InvoiceID INT PRIMARY KEY,
    InvoiceNum VARCHAR(50) UNIQUE, -- Business Rule: Invoice numbers must be unique
    TotalAmount DECIMAL(10, 2) CHECK (TotalAmount >= 0),
    CoverageDetails TEXT,
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT
);

CREATE TABLE RefundReasonAmount (
    Reason VARCHAR(255) PRIMARY KEY,
    Amount DECIMAL(10, 2) CHECK (Amount >= 0)
);

CREATE TABLE InsuranceCompanyProvider (
    Provider VARCHAR(255) PRIMARY KEY,
    Company VARCHAR(255)
);

CREATE TABLE DiscountCodePercentage (
    DiscountCode VARCHAR(50) PRIMARY KEY,
    Percentage DECIMAL(5, 2) CHECK (Percentage BETWEEN 0 AND 100) -- Business
Rule: Percentage 0-100
```

```
);

-- Relationship Tables (Admin & Billing)

CREATE TABLE InvoiceDiscount (
    InvoiceID INT,
    DiscountID INT,
    PRIMARY KEY (InvoiceID, DiscountID),
    FOREIGN KEY (InvoiceID) REFERENCES Invoice(InvoiceID),
    FOREIGN KEY (DiscountID) REFERENCES Discount(DiscountID)
);

CREATE TABLE PaymentEmployee (
    PaymentID INT,
    EmployeeID INT,
    PRIMARY KEY (PaymentID, EmployeeID),
    FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);

CREATE TABLE InvoiceInsurance (
    InvoiceID INT,
    InsuranceID INT,
    PRIMARY KEY (InvoiceID, InsuranceID),
    FOREIGN KEY (InvoiceID) REFERENCES Invoice(InvoiceID),
```

```
    FOREIGN KEY (InsuranceID) REFERENCES Insurance(InsuranceID)
);

CREATE TABLE PaymentPatient (
    PaymentID INT,
    PatientID INT,
    PRIMARY KEY (PaymentID, PatientID),
    FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE InvoiceEmployee (
    InvoiceID INT,
    EmployeeID INT,
    PRIMARY KEY (InvoiceID, EmployeeID),
    FOREIGN KEY (InvoiceID) REFERENCES Invoice(InvoiceID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);

CREATE TABLE EmployeeDiscount (
    DiscountID INT,
    EmployeeID INT,
    PRIMARY KEY (DiscountID, EmployeeID),
    FOREIGN KEY (DiscountID) REFERENCES Discount(DiscountID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
```

```
);

CREATE TABLE PaymentRefund (
    PaymentID INT,
    RefundID INT,
    PRIMARY KEY (PaymentID, RefundID),
    FOREIGN KEY (PaymentID) REFERENCES Payment(PaymentID),
    FOREIGN KEY (RefundID) REFERENCES Refund(RefundID)
);

CREATE TABLE PatientDiscount (
    DiscountID INT,
    PatientID INT,
    PRIMARY KEY (DiscountID, PatientID),
    FOREIGN KEY (DiscountID) REFERENCES Discount(DiscountID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE EmployeeRefund (
    RefundID INT,
    EmployeeID INT,
    PRIMARY KEY (RefundID, EmployeeID),
    FOREIGN KEY (RefundID) REFERENCES Refund(RefundID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);
```

```
CREATE TABLE PatientInsurance (
    InsuranceID INT,
    PatientID INT,
    PRIMARY KEY (InsuranceID, PatientID),
    FOREIGN KEY (InsuranceID) REFERENCES Insurance(InsuranceID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- ==========================================
-- 2. STAFF & VISITORS
-- ==========================================

CREATE TABLE Doctor (
    DoctorID INT PRIMARY KEY,
    PhoneNumber VARCHAR(20),
    YearsOfExperience INT CHECK (YearsOfExperience >= 0),
    Email VARCHAR(100) UNIQUE, -- Business Rule: Unique Email
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    WorkingTime VARCHAR(50)
);

CREATE TABLE Nurse (
    NurseID INT PRIMARY KEY,
```

```
    ShiftHours VARCHAR(50),
    YearsOfExperience INT CHECK (YearsOfExperience >= 0),
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    ContactNumber VARCHAR(20),
    DoctorID INT,
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

CREATE TABLE Department (
    DepartmentID INT PRIMARY KEY,
    NumberOfStaff INT CHECK (NumberOfStaff >= 0),
    Location VARCHAR(100),
    ContactNumber VARCHAR(20),
    DepartmentName VARCHAR(100) UNIQUE,
    DoctorID INT,
    LabID INT,
    CleaningServiceID INT,
    EmployeeID INT,
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID),
    FOREIGN KEY (LabID) REFERENCES Laboratory(LabID),
    FOREIGN KEY (CleaningServiceID) REFERENCES
CleaningService(CleaningServiceID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);
```

```
CREATE TABLE Visitor (
    VisitorID INT PRIMARY KEY,
    PhoneNumber VARCHAR(20),
    UAE_ID VARCHAR(50),
    RelationToPatient VARCHAR(50),
    Name VARCHAR(100),
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT,
    ReceptionistID INT,
    FOREIGN KEY (ReceptionistID) REFERENCES Receptionist(ReceptionistID)
);

CREATE TABLE Receptionist (
    ReceptionistID INT PRIMARY KEY,
    ShiftTiming VARCHAR(50),
    PhoneNumber VARCHAR(20),
    Email VARCHAR(100),
    FirstName VARCHAR(50),
    LastName VARCHAR(50)
);

CREATE TABLE Technician (
    TechnicianID INT PRIMARY KEY,
```

```
    PhoneNumber VARCHAR(20),
    WorkSchedule VARCHAR(100),
    FirstName VARCHAR(50),
    LastName VARCHAR(50)
);

-- Multi-Value Attributes

CREATE TABLE NurseQualifications (
    NurseID INT,
    Qualification VARCHAR(100),
    PRIMARY KEY (NurseID, Qualification),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID)
);

CREATE TABLE DoctorSpecialization (
    DoctorID INT,
    Specialization VARCHAR(100),
    PRIMARY KEY (DoctorID, Specialization),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

CREATE TABLE TechnicianSpecialization (
    TechnicianID INT,
    Specialization VARCHAR(100),
```

```
    PRIMARY KEY (TechnicianID, Specialization),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID)
);

-- Relationship Tables (Staff)

CREATE TABLE DoctorPatient (
    DoctorID INT,
    PatientID INT,
    PRIMARY KEY (DoctorID, PatientID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE DoctorMeasurement (
    DoctorID INT,
    MeasurementID INT,
    PRIMARY KEY (DoctorID, MeasurementID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID),
    FOREIGN KEY (MeasurementID) REFERENCES Measurement(MeasurementID)
);

CREATE TABLE NurseSurgeon (
    NurseID INT,
    SurgeonID INT,
```

```
    PRIMARY KEY (NurseID, SurgeonID),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID)
);

CREATE TABLE NurseMeasures (
    NurseID INT,
    MeasurementID INT,
    PRIMARY KEY (NurseID, MeasurementID),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID),
    FOREIGN KEY (MeasurementID) REFERENCES Measurement(MeasurementID)
);

CREATE TABLE NurseDepartment (
    NurseID INT,
    DepartmentID INT,
    PRIMARY KEY (NurseID, DepartmentID),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID),
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);

CREATE TABLE DepartmentTechnician (
    DepartmentID INT,
    TechnicianID INT,
    PRIMARY KEY (DepartmentID, TechnicianID),
```

```
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID)
);

CREATE TABLE DepartmentReceptionist (
    DepartmentID INT,
    ReceptionistID INT,
    PRIMARY KEY (DepartmentID, ReceptionistID),
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID),
    FOREIGN KEY (ReceptionistID) REFERENCES Receptionist(ReceptionistID)
);

CREATE TABLE VisitorPatient (
    VisitorID INT,
    PatientID INT,
    PRIMARY KEY (VisitorID, PatientID),
    FOREIGN KEY (VisitorID) REFERENCES Visitor(VisitorID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE ReceptionistDepartment (
    ReceptionistID INT,
    DepartmentID INT,
    PRIMARY KEY (ReceptionistID, DepartmentID),
    FOREIGN KEY (ReceptionistID) REFERENCES Receptionist(ReceptionistID),
```

```
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);

CREATE TABLE ReceptionistAppointment (
    ReceptionistID INT,
    AppointmentID INT,
    PRIMARY KEY (ReceptionistID, AppointmentID),
    FOREIGN KEY (ReceptionistID) REFERENCES Receptionist(ReceptionistID),
    FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
);

CREATE TABLE Techniciandepartment (
    TechnicianID INT,
    DepartmentID INT,
    PRIMARY KEY (TechnicianID, DepartmentID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID),
    FOREIGN KEY (DepartmentID) REFERENCES Department(DepartmentID)
);

CREATE TABLE TechnicianEquipment (
    TechnicianID INT,
    EquipmentID INT,
    PRIMARY KEY (TechnicianID, EquipmentID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID)
```

```
);

CREATE TABLE TechnicianLaboratories (
    TechnicianID INT,
    LabID INT,
    PRIMARY KEY (TechnicianID, LabID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID),
    FOREIGN KEY (LabID) REFERENCES Laboratory(LabID)
);

CREATE TABLE ReceptionistPatient (
    ReceptionistID INT,
    PatientID INT,
    PRIMARY KEY (ReceptionistID, PatientID),
    FOREIGN KEY (ReceptionistID) REFERENCES Receptionist(ReceptionistID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- ============================================
-- 3. CLINICAL, PHARMACY & LAB
-- ============================================

CREATE TABLE Diagnosis (
    DiagnosisID INT PRIMARY KEY,
    Symptoms TEXT,
```

```
    Notes TEXT,
    Severity VARCHAR(50),
    DiagnosisType VARCHAR(50),
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT,
    DoctorID INT,
    PatientID INT,
    PrescriptionID INT,
    MedicalRecordID INT,
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (PrescriptionID) REFERENCES Prescription(PrescriptionID),
    FOREIGN KEY (MedicalRecordID) REFERENCES MedicalRecord(RecordID)
);

CREATE TABLE Prescription (
    PrescriptionID INT PRIMARY KEY,
    Status VARCHAR(50),
    Dose VARCHAR(50),
    Day INT CHECK (Day BETWEEN 1 AND 31),
    Month INT CHECK (Month BETWEEN 1 AND 12),
    Year INT,
    DoctorID INT,
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
```

```
);

CREATE TABLE Medicine (
    MedicineID INT PRIMARY KEY,
    Name VARCHAR(100),
    Brand VARCHAR(100),
    Quantity INT CHECK (Quantity >= 0), -- Business Rule: Non-negative stock
    MedicineType VARCHAR(50),
    DosageForm VARCHAR(50),
    PDay INT CHECK (PDay BETWEEN 1 AND 31),
    PMonth INT CHECK (PMonth BETWEEN 1 AND 12),
    PYear INT,
    EDay INT CHECK (EDay BETWEEN 1 AND 31),
    EMonth INT CHECK (EMonth BETWEEN 1 AND 12),
    EYear INT
);

CREATE TABLE TestReport (
    ReportID INT PRIMARY KEY,
    LabID INT,
    TechnicianID INT,
    PatientID INT,
    MeasurementID INT,
    DoctorID INT,
    TestResult TEXT,
```

```
    IDay INT CHECK (IDay BETWEEN 1 AND 31),
    IMonth INT CHECK (IMonth BETWEEN 1 AND 12),
    IYear INT,
    EDay INT CHECK (EDay BETWEEN 1 AND 31),
    EMonth INT CHECK (EMonth BETWEEN 1 AND 12),
    EYear INT,
    TestingFor VARCHAR(255),
    FOREIGN KEY (LabID) REFERENCES Laboratory(LabID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (MeasurementID) REFERENCES Measurement(MeasurementID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

CREATE TABLE Pharmacy (
    PharmacyID INT PRIMARY KEY,
    PharmacyName VARCHAR(100),
    Location VARCHAR(100),
    Email VARCHAR(100),
    Phone VARCHAR(20),
    StorageCapacity VARCHAR(50),
    PrescriptionID INT,
    FOREIGN KEY (PrescriptionID) REFERENCES Prescription(PrescriptionID)
);
```

```
CREATE TABLE Laboratory (
    LabID INT PRIMARY KEY,
    OpeningHours VARCHAR(50),
    ClosingHours VARCHAR(50),
    Equipment TEXT,
    Tests TEXT
);

-- Relationship Tables (Clinical)

CREATE TABLE DiagnosisTestReport (
    DiagnosisID INT,
    ReportID INT,
    PRIMARY KEY (DiagnosisID, ReportID),
    FOREIGN KEY (DiagnosisID) REFERENCES Diagnosis(DiagnosisID),
    FOREIGN KEY (ReportID) REFERENCES TestReport(ReportID)
);

CREATE TABLE MedicinePrescription (
    PrescriptionID INT,
    MedicineID INT,
    PRIMARY KEY (PrescriptionID, MedicineID),
    FOREIGN KEY (PrescriptionID) REFERENCES Prescription(PrescriptionID),
    FOREIGN KEY (MedicineID) REFERENCES Medicine(MedicineID)
);
```

```
CREATE TABLE LabEquipment (
    LabID INT,
    Equipment VARCHAR(100),
    EQuantity INT CHECK (EQuantity >= 0),
    PRIMARY KEY (LabID, Equipment),
    FOREIGN KEY (LabID) REFERENCES Laboratory(LabID)
);

CREATE TABLE LabResults (
    LabID INT,
    Tests VARCHAR(255),
    PRIMARY KEY (LabID, Tests),
    FOREIGN KEY (LabID) REFERENCES Laboratory(LabID)
);

CREATE TABLE SupplierMedicine (
    SupplierID INT,
    MedicineID INT,
    PRIMARY KEY (SupplierID, MedicineID),
    FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID),
    FOREIGN KEY (MedicineID) REFERENCES Medicine(MedicineID)
);

CREATE TABLE PatientMedicine (
```

```
    PatientID INT,
    MedicineID INT,
    PRIMARY KEY (PatientID, MedicineID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (MedicineID) REFERENCES Medicine(MedicineID)
);

CREATE TABLE EmployeeTestReport (
    EmployeeID INT,
    ReportID INT,
    PRIMARY KEY (EmployeeID, ReportID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
    FOREIGN KEY (ReportID) REFERENCES TestReport(ReportID)
);

CREATE TABLE MedicalRecordTestReport (
    RecordID INT,
    ReportID INT,
    PRIMARY KEY (RecordID, ReportID),
    FOREIGN KEY (RecordID) REFERENCES MedicalRecord(RecordID),
    FOREIGN KEY (ReportID) REFERENCES TestReport(ReportID)
);

CREATE TABLE MedicineNamePrescription (
    PrescriptionID INT,
```

```
    MedicineName VARCHAR(100),
    PRIMARY KEY (PrescriptionID, MedicineName),
    FOREIGN KEY (PrescriptionID) REFERENCES Prescription(PrescriptionID)
);

CREATE TABLE EquipmentQuantity (
    Equipment VARCHAR(100) PRIMARY KEY,
    EQuantity INT CHECK (EQuantity >= 0)
);

CREATE TABLE TestTypeReason (
    TestName VARCHAR(100) PRIMARY KEY,
    TestingFor VARCHAR(255),
    TestType VARCHAR(100)
);

-- ==========================================
-- 4. FACILITIES, EMERGENCY & EQUIPMENT
-- ==========================================

CREATE TABLE CleaningService (
    CleaningServiceID INT PRIMARY KEY,
    ServiceDateTime DATETIME,
    CleaningType VARCHAR(50),
    Status VARCHAR(50),
```

```
    Duration VARCHAR(50),
    SuppliesUsed TEXT,
    RoomID INT,
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID)
);

CREATE TABLE Emergency (
    EmergencyID INT PRIMARY KEY,
    ArrivalDateTime DATETIME,
    EmergencyType VARCHAR(50),
    SeverityLevel VARCHAR(50),
    TreatmentGiven TEXT,
    Outcome VARCHAR(255),
    PatientID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE Room (
    RoomID INT PRIMARY KEY,
    RoomType VARCHAR(50),
    RoomNumber VARCHAR(20) UNIQUE, -- Business Rule: Unique Room Number
    Status VARCHAR(50),
    RoomPhoneNumber VARCHAR(20)
);
```

```
CREATE TABLE RoomTypeMaxOccupancyCostPerDay (
    RoomType VARCHAR(50) PRIMARY KEY,
    MaxOccupancy INT CHECK (MaxOccupancy > 0),
    CostPerDay DECIMAL(10, 2) CHECK (CostPerDay >= 0)
);

CREATE TABLE BedAssignment (
    AssignmentID INT PRIMARY KEY,
    AssignmentStatus VARCHAR(50),
    AdmissionDateTime DATETIME,
    DischargeDateTime DATETIME,
    RoomID INT,
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID),
    CHECK (DischargeDateTime >= AdmissionDateTime) -- Business Rule: Discharge after Admission
);

CREATE TABLE Equipment (
    EquipmentID INT PRIMARY KEY,
    EquipmentName VARCHAR(100),
    EquipmentType VARCHAR(50),
    SerialNumber VARCHAR(50) UNIQUE, -- Business Rule: Unique Serial
    ModelNumber VARCHAR(50),
    PurchaseDate DATE,
    LastMaintenanceDate DATE,
```

```
    NextMaintenanceDue DATE,
    EquipmentStatus VARCHAR(50)
);

CREATE TABLE Ambulance (
    AmbulanceID INT PRIMARY KEY,
    PlateNumber VARCHAR(20) UNIQUE,
    DriverID INT,
    AmbulanceType VARCHAR(50),
    Status VARCHAR(50),
    Location VARCHAR(100),
    LastMaintenanceDate DATE,
    EmployeeID INT,
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);

CREATE TABLE AmbulanceTypeCapacity (
    AmbulanceType VARCHAR(50) PRIMARY KEY,
    Capacity INT CHECK (Capacity > 0)
);

-- Relationship Tables (Facilities)

CREATE TABLE RoomEquipment (
    RoomID INT,
```

```
    EquipmentID INT,
    PRIMARY KEY (RoomID, EquipmentID),
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID)
);

CREATE TABLE RoomPatient (
    PatientID INT,
    RoomID INT,
    PRIMARY KEY (PatientID, RoomID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID)
);

CREATE TABLE RoomNurse (
    RoomID INT,
    NurseID INT,
    PRIMARY KEY (RoomID, NurseID),
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID)
);

CREATE TABLE RoomAppointment (
    RoomID INT,
    AppointmentID INT,
```

```
    PRIMARY KEY (RoomID, AppointmentID),
    FOREIGN KEY (RoomID) REFERENCES Room(RoomID),
    FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
);

CREATE TABLE EquipmentSurgeon (
    EquipmentID INT,
    SurgeonID INT,
    PRIMARY KEY (EquipmentID, SurgeonID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID)
);

CREATE TABLE EquipmentMeasurement (
    EquipmentID INT,
    MeasurementID INT,
    PRIMARY KEY (EquipmentID, MeasurementID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (MeasurementID) REFERENCES Measurement(MeasurementID)
);

CREATE TABLE EquipmentDoctor (
    EquipmentID INT,
    DoctorID INT,
    PRIMARY KEY (EquipmentID, DoctorID),
```

```
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

CREATE TABLE EquipmentNurse (
    EquipmentID INT,
    NurseID INT,
    PRIMARY KEY (EquipmentID, NurseID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (NurseID) REFERENCES Nurse(NurseID)
);

CREATE TABLE EquipmentAmbulance (
    EquipmentID INT,
    AmbulanceID INT,
    PRIMARY KEY (EquipmentID, AmbulanceID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (AmbulanceID) REFERENCES Ambulance(AmbulanceID)
);

CREATE TABLE EquipmentEmployee (
    EquipmentID INT,
    EmployeeID INT,
    PRIMARY KEY (EquipmentID, EmployeeID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
```

```
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);

CREATE TABLE PatientAmbulance (
    PatientID INT,
    AmbulanceID INT,
    PRIMARY KEY (PatientID, AmbulanceID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (AmbulanceID) REFERENCES Ambulance(AmbulanceID)
);

CREATE TABLE EmergencyEquipment (
    EmergencyID INT,
    EquipmentID INT,
    PRIMARY KEY (EmergencyID, EquipmentID),
    FOREIGN KEY (EmergencyID) REFERENCES Emergency(EmergencyID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID)
);

CREATE TABLE EmergencySurgeon (
    EmergencyID INT,
    SurgeonID INT,
    PRIMARY KEY (EmergencyID, SurgeonID),
    FOREIGN KEY (EmergencyID) REFERENCES Emergency(EmergencyID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID)
```

```
);

CREATE TABLE EquipmentTechnician (
    EquipmentID INT,
    TechnicianID INT,
    PRIMARY KEY (EquipmentID, TechnicianID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (TechnicianID) REFERENCES Technician(TechnicianID)
);

CREATE TABLE EquipmentSupplier (
    EquipmentID INT,
    SupplierID INT,
    PRIMARY KEY (EquipmentID, SupplierID),
    FOREIGN KEY (EquipmentID) REFERENCES Equipment(EquipmentID),
    FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)
);

CREATE TABLE AmbulanceEmergency (
    AmbulanceID INT,
    EmergencyID INT,
    PRIMARY KEY (AmbulanceID, EmergencyID),
    FOREIGN KEY (AmbulanceID) REFERENCES Ambulance(AmbulanceID),
    FOREIGN KEY (EmergencyID) REFERENCES Emergency(EmergencyID)
);
```

```
CREATE TABLE PatientBedAssignment (
    PatientID INT,
    BedAssignmentID INT,
    PRIMARY KEY (PatientID, BedAssignmentID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (BedAssignmentID) REFERENCES BedAssignment(AssignmentID)
);

-- ==========================================
-- 5. RECORDS, EMPLOYEES & SURGERY
-- ==========================================

CREATE TABLE MedicalRecord (
    PatientID INT,
    RecordID INT UNIQUE, -- Added UNIQUE Constraint to allow FK Reference
    Notes TEXT,
    RDay INT CHECK (RDay BETWEEN 1 AND 31),
    RMonth INT CHECK (RMonth BETWEEN 1 AND 12),
    RYear INT,
    EmployeeID INT,
    PRIMARY KEY (PatientID, RecordID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID)
);
```

```
CREATE TABLE Appointment (
    AppointmentID INT PRIMARY KEY,
    DoctorID INT,
    Reason TEXT,
    Status VARCHAR(50),
    AppDay INT CHECK (AppDay BETWEEN 1 AND 31),
    AppMonth INT CHECK (AppMonth BETWEEN 1 AND 12),
    AppYear INT,
    AppointmentTime VARCHAR(20),
    FOREIGN KEY (DoctorID) REFERENCES Doctor(DoctorID)
);

CREATE TABLE Patient (
    PatientID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    DOB DATE,
    Gender VARCHAR(20), -- CHECK (Gender IN ('Male', 'Female')) could be added here
    ContactNumber VARCHAR(20),
    BloodType VARCHAR(10), -- CHECK (BloodType IN ('A+', 'A-', 'B+', 'B-', 'O+', 'O-', 'AB+', 'AB-')) could be added here
    Address TEXT,
    AppointmentID INT,
    FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
);
```

```
CREATE TABLE Registration (
    PatientID INT,
    RegistrationID INT,
    RegDay INT CHECK (RegDay BETWEEN 1 AND 31),
    RegMonth INT CHECK (RegMonth BETWEEN 1 AND 12),
    RegYear INT,
    Type VARCHAR(50),
    PRIMARY KEY (PatientID, RegistrationID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

CREATE TABLE Supplier (
    SupplierID INT PRIMARY KEY,
    SupplierName VARCHAR(100),
    ProductType VARCHAR(100),
    Address TEXT,
    ContractStartDay INT CHECK (ContractStartDay BETWEEN 1 AND 31),
    ContractStartMonth INT CHECK (ContractStartMonth BETWEEN 1 AND 12),
    ContractStartYear INT,
    ContractEndDay INT CHECK (ContractEndDay BETWEEN 1 AND 31),
    ContractEndMonth INT CHECK (ContractEndMonth BETWEEN 1 AND 12),
    ContractEndYear INT
);
```

```
CREATE TABLE Measurement (
    MeasurementID INT PRIMARY KEY,
    Status VARCHAR(50),
    MeasurementTime VARCHAR(50),
    Value VARCHAR(50),
    MeasurementType VARCHAR(50),
    PatientID INT,
    MedicalRecordID INT,
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (MedicalRecordID) REFERENCES MedicalRecord(RecordID)
);

CREATE TABLE Employee (
    EmployeeID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    JobTitle VARCHAR(50),
    Department VARCHAR(50),
    Salary DECIMAL(10, 2) CHECK (Salary >= 0),
    DateHired DATE,
    EmployementStatus VARCHAR(50),
    Country VARCHAR(50),
    City VARCHAR(50),
    Area VARCHAR(50),
    Street VARCHAR(100)
```

```
);

CREATE TABLE Surgeon (
    SurgeonID INT PRIMARY KEY,
    FirstName VARCHAR(50),
    LastName VARCHAR(50),
    Specialization VARCHAR(100),
    LicenseNumber VARCHAR(50) UNIQUE,
    OperatingRoom VARCHAR(50),
    AvailabilityStatus VARCHAR(50),
    YearsofExperience INT CHECK (YearsofExperience >= 0)
);

CREATE TABLE SurgeonQualifications (
    SurgeonID INT,
    Qualifications VARCHAR(255),
    PRIMARY KEY (SurgeonID, Qualifications),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID)
);

CREATE TABLE EmployeeSupplier (
    EmployeeID INT,
    SupplierID INT,
    PRIMARY KEY (EmployeeID, SupplierID),
    FOREIGN KEY (EmployeeID) REFERENCES Employee(EmployeeID),
```

```
    FOREIGN KEY (SupplierID) REFERENCES Supplier(SupplierID)
);

CREATE TABLE PatientSurgeon (
    PatientID INT,
    SurgeonID INT,
    PRIMARY KEY (PatientID, SurgeonID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID)
);

CREATE TABLE SurgeonMedicalRecord (
    SurgeonID INT,
    MedicalRecordID INT,
    PRIMARY KEY (SurgeonID, MedicalRecordID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID),
    FOREIGN KEY (MedicalRecordID) REFERENCES MedicalRecord(RecordID)
);

CREATE TABLE SurgeonAppointment (
    SurgeonID INT,
    AppointmentID INT,
    PRIMARY KEY (SurgeonID, AppointmentID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID),
    FOREIGN KEY (AppointmentID) REFERENCES Appointment(AppointmentID)
```

```
);

CREATE TABLE SurgeonPatient (
    SurgeonID INT,
    PatientID INT,
    PRIMARY KEY (SurgeonID, PatientID),
    FOREIGN KEY (SurgeonID) REFERENCES Surgeon(SurgeonID),
    FOREIGN KEY (PatientID) REFERENCES Patient(PatientID)
);

-- Re-enable foreign key checks
SET FOREIGN_KEY_CHECKS = 1;
```

Sample Data:

```
-- CSCI 326 Project: Hospital Management System - Dummy Data




USE HospitalDB;

-- Disable checks for bulk insertion
SET FOREIGN_KEY_CHECKS = 0;

-- 1. INSERT STAFF (Doctors, Nurses, Admin)
```

-- Doctors

INSERT INTO Doctor (DoctorID, FirstName, LastName, Email, PhoneNumber, YearsOfExperience, WorkingTime) VALUES

(1, 'Gregory', 'House', 'house@hospital.com', '050-111-2222', 20, '08:00-16:00'),

(2, 'Lisa', 'Cuddy', 'cuddy@hospital.com', '050-111-2223', 18, '09:00-17:00'),

(3, 'James', 'Wilson', 'wilson@hospital.com', '050-111-2224', 15, '08:00-16:00'),

(4, 'Allison', 'Cameron', 'cameron@hospital.com', '050-111-2225', 5, '14:00-22:00'),

(5, 'Robert', 'Chase', 'chase@hospital.com', '050-111-2226', 8, '14:00-22:00');

-- Doctor Specializations

INSERT INTO DoctorSpecialization (DoctorID, Specialization) VALUES

(1, 'Nephrology'), (1, 'Infectious Disease'),

(2, 'Endocrinology'),

(3, 'Oncology'),

(4, 'Immunology'),

(5, 'Intensive Care');

-- Nurses

INSERT INTO Nurse (NurseID, FirstName, LastName, ContactNumber, ShiftHours, YearsOfExperience, DoctorID) VALUES

(101, 'Carla', 'Espinosa', '055-222-3331', 'Morning', 10, 1),

(102, 'Morgan', 'Tookers', '055-222-3332', 'Morning', 5, 2),

(103, 'Ann', 'Perkins', '055-222-3333', 'Night', 4, 3),

(104, 'Rory', 'Gilmore', '055-222-3334', 'Night', 2, 4),

(105, 'April', 'Ludgate', '055-222-3335', 'Evening', 3, 5);

-- Receptionists

INSERT INTO Receptionist (ReceptionistID, FirstName, LastName, Email, PhoneNumber, ShiftTiming) VALUES

(201, 'Pam', 'Beesly', 'pam@hospital.com', '056-333-4441', '07:00-15:00'),

(202, 'Erin', 'Hannon', 'erin@hospital.com', '056-333-4442', '15:00-23:00');

-- General Employees (Admin/Staff)

INSERT INTO Employee (EmployeeID, FirstName, LastName, JobTitle, Department, Salary, DateHired, EmployementStatus, Country, City, Area, Street) VALUES

(301, 'Michael', 'Scott', 'Manager', 'Administration', 15000.00, '2010-01-01', 'Active', 'UAE', 'Dubai', 'Downtown', 'Main St'),

(302, 'Dwight', 'Schrute', 'Assistant Manager', 'Administration', 12000.00, '2012-03-15', 'Active', 'UAE', 'Dubai', 'JLT', 'Cluster F'),

(303, 'Toby', 'Flenderson', 'HR Rep', 'HR', 10000.00, '2011-05-20', 'Active', 'UAE', 'Abu Dhabi', 'Corniche', 'Zayed St');

-- 2. INSERT INFRASTRUCTURE

-- Departments

INSERT INTO Department (DepartmentID, DepartmentName, Location, ContactNumber, NumberOfStaff, DoctorID, EmployeeID) VALUES

(10, 'Emergency', 'Ground Floor', '04-100-1000', 20, 5, 301),

(20, 'Cardiology', 'First Floor', '04-100-2000', 15, 1, 301),

(30, 'Oncology', 'Second Floor', '04-100-3000', 10, 3, 302),

(40, 'General Surgery', 'Third Floor', '04-100-4000', 25, 2, 302);

-- Rooms

INSERT INTO Room (RoomID, RoomNumber, RoomType, Status, RoomPhoneNumber) VALUES

(1, '101', 'Private', 'Occupied', '1001'),

(2, '102', 'Private', 'Vacant', '1002'),

(3, '201', 'ICU', 'Occupied', '2001'),

(4, '202', 'ICU', 'Vacant', '2002'),

(5, '301', 'Ward', 'Occupied', '3001'),

(6, '302', 'Ward', 'Occupied', '3002');

-- 3. INSERT APPOINTMENTS (Needed before Patients due to FK)

INSERT INTO Appointment (AppointmentID, DoctorID, AppDay, AppMonth, AppYear, AppointmentTime, Status, Reason) VALUES

(501, 1, 15, 11, 2025, '09:00', 'Completed', 'Persistent Headache'),

(502, 2, 15, 11, 2025, '10:00', 'Completed', 'Routine Checkup'),

(503, 3, 16, 11, 2025, '11:00', 'Scheduled', 'Back Pain'),

(504, 1, 16, 11, 2025, '14:00', 'Scheduled', 'Flu Symptoms'),

(505, 4, 17, 11, 2025, '09:30', 'Scheduled', 'Allergy Test'),

(506, 2, 18, 11, 2025, '12:00', 'Cancelled', 'Follow-up'),

(507, 5, 19, 11, 2025, '15:00', 'Scheduled', 'Chest Pain'),

(508, 1, 20, 11, 2025, '10:00', 'Scheduled', 'Leg Injury'),

(509, 3, 20, 11, 2025, '13:00', 'Scheduled', 'Screening'),

(510, 4, 21, 11, 2025, '08:00', 'Scheduled', 'Vaccination'),

(511, 2, 22, 11, 2025, '11:00', 'Scheduled', 'Blood Test Analysis'),

(512, 5, 23, 11, 2025, '16:00', 'Scheduled', 'Emergency Follow-up'),

(513, 1, 24, 11, 2025, '09:00', 'Scheduled', 'Consultation'),

(514, 3, 25, 11, 2025, '14:00', 'Scheduled', 'Chemo Session'),

(515, 4, 26, 11, 2025, '10:30', 'Scheduled', 'Skin Rash'),

(516, 2, 27, 11, 2025, '11:30', 'Scheduled', 'Diabetes Check'),

(517, 5, 28, 11, 2025, '13:30', 'Scheduled', 'Surgery Consult'),

(518, 1, 29, 11, 2025, '09:15', 'Scheduled', 'Migraine'),

(519, 3, 30, 11, 2025, '15:45', 'Scheduled', 'MRI Review'),

(520, 2, 01, 12, 2025, '10:00', 'Scheduled', 'Physical Therapy');

-- 4. INSERT PATIENTS (20 Minimum)

INSERT INTO Patient (PatientID, FirstName, LastName, DOB, Gender, ContactNumber, BloodType, Address, AppointmentID) VALUES

(1001, 'John', 'Doe', '1985-05-15', 'Male', '050-000-0001', 'O+', 'Dubai, Marina', 501),

(1002, 'Jane', 'Smith', '1990-08-20', 'Female', '050-000-0002', 'A+', 'Dubai, Jumeirah', 502),

(1003, 'Michael', 'Brown', '1978-02-10', 'Male', '050-000-0003', 'B-', 'Abu Dhabi, Yas', 503),

(1004, 'Emily', 'Davis', '1995-11-05', 'Female', '050-000-0004', 'AB+', 'Sharjah, Al Nahda', 504),

(1005, 'Chris', 'Wilson', '2000-01-25', 'Male', '050-000-0005', 'O-', 'Dubai, Deira', 505),

(1006, 'Sarah', 'Johnson', '1982-06-14', 'Female', '050-000-0006', 'A-', 'Ajman, Corniche', 506),

(1007, 'David', 'Miller', '1965-09-30', 'Male', '050-000-0007', 'B+', 'Dubai, Downtown', 507),

(1008, 'Laura', 'Garcia', '1988-03-22', 'Female', '050-000-0008', 'O+', 'RAK, Al Hamra', 508),

(1009, 'James', 'Martinez', '1992-12-12', 'Male', '050-000-0009', 'AB-', 'Dubai, Silicon Oasis', 509),

(1010, 'Linda', 'Anderson', '1975-07-07', 'Female', '050-000-0010', 'A+', 'Fujairah, City Center', 510),

(1011, 'Robert', 'Taylor', '1950-04-18', 'Male', '050-000-0011', 'O+', 'Dubai, The Palm', 511),

(1012, 'Patricia', 'Thomas', '1999-09-09', 'Female', '050-000-0012', 'B+', 'Abu Dhabi, Reem Island', 512),

(1013, 'Charles', 'Hernandez', '1980-01-30', 'Male', '050-000-0013', 'A-', 'Dubai, Mirdif', 513),

(1014, 'Barbara', 'Moore', '1960-10-20', 'Female', '050-000-0014', 'O-', 'Sharjah, Muweilah', 514),

(1015, 'Paul', 'Martin', '1993-05-05', 'Male', '050-000-0015', 'AB+', 'Dubai, Barsha', 515),

(1016, 'Jennifer', 'Jackson', '1987-08-17', 'Female', '050-000-0016', 'B-', 'Al Ain, Civic Center', 516),

(1017, 'Mark', 'Thompson', '1970-12-01', 'Male', '050-000-0017', 'O+', 'Dubai, JVC', 517),

(1018, 'Elizabeth', 'White', '2005-02-28', 'Female', '050-000-0018', 'A+', 'Abu Dhabi, Saadiyat', 518),

(1019, 'Steven', 'Lopez', '1991-06-15', 'Male', '050-000-0019', 'B+', 'Dubai, Discovery Gardens', 519),

(1020, 'Jessica', 'Lee', '1996-11-11', 'Female', '050-000-0020', 'AB-', 'Dubai, Motor City', 520);

-- 5. INSERT MEDICAL RECORDS & DIAGNOSIS

INSERT INTO MedicalRecord (RecordID, PatientID, Notes, RDay, RMonth, RYear, EmployeeID) VALUES

(6001, 1001, 'Patient complains of severe migraines.', 15, 11, 2025, 301),

(6002, 1002, 'Annual physical checkup details.', 15, 11, 2025, 301),

(6003, 1007, 'Admitted via Emergency. Chest pains.', 19, 11, 2025, 302),

(6004, 1003, 'Chronic back pain evaluation.', 16, 11, 2025, 302),

(6005, 1004, 'High fever and chills.', 16, 11, 2025, 301);

INSERT INTO Diagnosis (DiagnosisID, PatientID, DoctorID, MedicalRecordID, DiagnosisType, Severity, Symptoms, Day, Month, Year) VALUES

(701, 1001, 1, 6001, 'Neurological', 'Moderate', 'Headache, Sensitivity to light', 15, 11, 2025),

(702, 1002, 2, 6002, 'General', 'Low', 'None', 15, 11, 2025),

(703, 1007, 5, 6003, 'Cardiac', 'High', 'Chest tightness, Shortness of breath', 19, 11, 2025),

(704, 1003, 3, 6004, 'Orthopedic', 'Moderate', 'Lower lumbar pain', 16, 11, 2025),

(705, 1004, 1, 6005, 'Viral', 'Moderate', 'Fever, Body ache', 16, 11, 2025);

-- 6. INSERT PRESCRIPTIONS & MEDICINE

INSERT INTO Medicine (MedicineID, Name, Brand, Quantity, MedicineType, DosageForm, PDay, PMonth, PYear, EDay, EMonth, EYear) VALUES

(801, 'Paracetamol', 'Panadol', 1000, 'Analgesic', 'Tablet', 01, 01, 2024, 01, 01, 2026),

(802, 'Ibuprofen', 'Advil', 500, 'NSAID', 'Tablet', 01, 06, 2024, 01, 06, 2026),

(803, 'Amoxicillin', 'Amoxil', 200, 'Antibiotic', 'Capsule', 15, 03, 2024, 15, 03, 2025),

(804, 'Atorvastatin', 'Lipitor', 300, 'Statin', 'Tablet', 10, 02, 2024, 10, 02, 2026),

(805, 'Insulin', 'Lantus', 50, 'Hormone', 'Injection', 01, 11, 2025, 01, 11, 2026);

INSERT INTO Prescription (PrescriptionID, DoctorID, Status, Dose, Day, Month, Year) VALUES

(901, 1, 'Active', '500mg twice daily', 15, 11, 2025),

(902, 5, 'Active', '10mg once daily', 19, 11, 2025),

(903, 1, 'Completed', '250mg thrice daily', 16, 11, 2025);

-- Linking Medicine to Prescription

INSERT INTO MedicinePrescription (PrescriptionID, MedicineID) VALUES

(901, 801), -- Panadol for headache

(902, 804), -- Lipitor for cardiac patient

(903, 803); -- Amoxicillin for flu

-- 7. INSERT BILLING (Invoices & Payments)

INSERT INTO Invoice (InvoiceID, InvoiceNum, TotalAmount, CoverageDetails, Day, Month, Year) VALUES

(2001, 'INV-2025-001', 350.00, 'Self-Pay', 15, 11, 2025),

(2002, 'INV-2025-002', 1500.00, 'Insurance Cover 80%', 19, 11, 2025),

(2003, 'INV-2025-003', 200.00, 'Full Coverage', 16, 11, 2025);

INSERT INTO Payment (PaymentID, InvoiceID, Method, Amount, PaymentStatus, Day, Month, Year) VALUES

(3001, 2001, 'Credit Card', 350.00, 'Paid', 15, 11, 2025),

(3002, 2002, 'Insurance', 1200.00, 'Paid', 19, 11, 2025),

(3003, 2002, 'Cash', 300.00, 'Paid', 19, 11, 2025);

-- Re-enable checks

SET FOREIGN_KEY_CHECKS = 1;

Sample output:
Patient table output after Inserting Data:

PatientID	FirstName	LastName	DOB	Gender	ContactNumber	BloodType	Address	AppointmentID
1001	John	Doe	1985-05-15	Male	050-000-0001	O+	Dubai, Marina	501
1002	Jane	Smith	1990-08-20	Female	050-000-0002	A+	Dubai, Jumeirah	502
1003	Michael	Brown	1978-02-10	Male	050-000-0003	B-	Abu Dhabi, Yas	503
1004	Emily	Davis	1995-11-05	Female	050-000-0004	AB+	Sharjah, Al Nahda	504
1005	Chris	Wilson	2000-01-25	Male	050-000-0005	O-	Dubai, Deira	505
1006	Sarah	Johnson	1982-06-14	Female	050-000-0006	A-	Ajman, Corniche	506
1007	David	Miller	1965-09-30	Male	050-000-0007	B+	Dubai, Downtown	507
1008	Laura	Garcia	1988-03-22	Female	050-000-0008	O+	RAK, Al Hamra	508
1009	James	Martinez	1992-12-12	Male	050-000-0009	AB-	Dubai, Silicon Oasis	509
1010	Linda	Anderson	1975-07-07	Female	050-000-0010	A+	Fujairah, City Center	510
1011	Robert	Taylor	1950-04-18	Male	050-000-0011	O+	Dubai, The Palm	511
1012	Patricia	Thomas	1999-09-09	Female	050-000-0012	B+	Abu Dhabi, Reem Isl...	512
1013	Charles	Hernandez	1980-01-30	Male	050-000-0013	A-	Dubai, Mirdif	513
1014	Barbara	Moore	1960-10-20	Female	050-000-0014	O-	Sharjah, Muweilah	514
1015	Paul	Martin	1993-05-05	Male	050-000-0015	AB+	Dubai, Barsha	515
1016	Jennifer	Jackson	1987-08-17	Female	050-000-0016	B-	Al Ain, Civic Center	516
1017	Mark	Thompson	1970-12-01	Male	050-000-0017	O+	Dubai, JVC	517
1018	Elizabeth	White	2005-02-28	Female	050-000-0018	A+	Abu Dhabi, Saadiyat	518
1019	Steven	Lopez	1991-06-15	Male	050-000-0019	B+	Dubai, Discovery Ga...	519
1020	Jessica	Lee	1996-11-11	Female	050-000-0020	AB-	Dubai, Motor City	520
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Doctor table output after Inserting Data:

DoctorID	PhoneNumber	YearsOfExperience	Email	FirstName	LastName	WorkingTime
1	050-111-2222	20	house@hospital.com	Gregory	House	08:00-16:00
2	050-111-2223	18	cuddy@hospital.com	Lisa	Cuddy	09:00-17:00
3	050-111-2224	15	wilson@hospital.com	James	Wilson	08:00-16:00
4	050-111-2225	5	cameron@hospital.com	Allison	Cameron	14:00-22:00
5	050-111-2226	8	chase@hospital.com	Robert	Chase	14:00-22:00
NULL	NULL	NULL	NULL	NULL	NULL	NULL

Nurse table output after inserting data:

NurseID	ShiftHours	YearsOfExperience	FirstName	LastName	ContactNumber	DoctorID
101	Morning	10	Carla	Espinosa	055-222-3331	1
102	Morning	5	Morgan	Tookers	055-222-3332	2
103	Night	4	Ann	Perkins	055-222-3333	3
104	Night	2	Rory	Gilmore	055-222-3334	4
105	Evening	3	April	Ludgate	055-222-3335	5
NULL	NULL	NULL	NULL	NULL	NULL	NULL

6.- Ethical Considerations:

Designing a database for healthcare requires strict adherence to ethical standards and legal regulations due to the sensitive nature of the data involved. Our Hospital Management System was designed with ***Privacy by Design*** *principles to ensure patient confidentiality and data security.*

- ***Data Privacy & Confidentiality:***
  - ***Separation of Concerns:*** *We structured the database to logically separate administrative data from clinical data. For example, the Patient table contains demographic details (Name, Address), while sensitive health data is stored in*

separate MedicalRecord and Diagnosis tables. This separation facilitates better access control, ensuring that administrative staff (like Receptionists) can access contact information without being exposed to sensitive medical history.

- ***Minimization:*** *We avoided storing unnecessary personal data. For instance, NurseQualifications are stored in a separate table, linked only by ID, reducing the risk of exposing staff profiles unnecessarily.*

- ***Responsible Data Access (Access Control):***
  - *The schema supports* ***Role-Based Access Control (RBAC)****. By having distinct tables for Billing (Invoices) and Clinical (Prescriptions), the database administrator can easily grant the Finance Department access only to billing tables while restricting them from viewing patient diagnoses. This prevents unauthorized personnel from browsing sensitive health records.*
- ***Compliance (GDPR & HIPAA Standards):***
  - ***Right to be Forgotten:*** *The use of clearly defined Primary and Foreign Keys allows for the systematic deletion or anonymization of a patient's data if requested, complying with GDPR regulations.*
  - ***Auditability:*** *The clear structure of relationships (e.g., linking a Prescription directly to a specific DoctorID) ensures accountability. Every medical decision is traceable back to the specific staff member responsible, which is crucial for medical ethics and legal liability.*
- ***Avoiding Bias:***
  - *By using standardized lookup tables (e.g., DoctorSpecialization, RoomType) rather than free-text fields for everything, we reduce the risk of subjective or biased data entry, ensuring that all patients are categorized using standard medical and administrative terminology.*

8.- Timeline of our progress:

During our initial meeting, we selected our project topic: designing a comprehensive database system for a hospital. After finalizing the idea, we chose Figma as our primary platform because it enables real-time collaborative work and automatically maintains backups at each stage, so that we never lose progress.

We then divided the tasks among the team members by assigning each person a specific set of entities. Most of the organizational and managerial coordination throughout the project was handled by Amir. Our workflow followed a structured progression: we began by identifying the entities and defining their attributes, then proceeded to establish the relationships between them. Afterward, we created both the entity tables and the relationship tables.

Once we completed the ER diagrams for all entities and relationships, we faced a significant challenge. Because the hospital system required many entities and lot of interconnections, the

ER diagrams became dense and difficult to interpret. It was easy to get lost in the details, and the diagrams themselves became visually overwhelming. To solve this, we developed a solution: a simple, interactive website. This website allowed users to explore the ER model more intuitively by clicking on a pressable card for any entity (for example, Patients), the user could immediately view all relevant information, relationships, and attributes associated with that entity. This greatly improved clarity and helped us navigate the system more efficiently.

We also set internal deadlines to maintain steady progress and prevent delays. After completing the diagrams and ensuring all components were aligned correctly, we reviewed and cross-checked each other's work. We then proceeded to the normalization phase, which involved transforming all tables into First Normal Form (1NF), Second Normal Form (2NF), and Third Normal Form (3NF).

Following normalization, we began preparing the final report. This phase included the SQL implementation, which was primarily completed by Amir, while the remaining sections of the report were distributed among the other team members. Finally, we assembled all components, integrated them into a single document, and completed the project.

7.- Conclusion:

In conclusion, the design and implementation of this Hospital Management System (HMS) database successfully addresses the critical need for structured, efficient, and secure information management in healthcare environments. Through a rigorous process of Requirement Analysis, Entity-Relationship (ER) modeling, and Relational Mapping, our team has developed a robust schema capable of handling the multifaceted operations of a modern hospital.

By strictly adhering to Normalization principles (up to Third Normal Form), we ensured that data redundancy was minimized and potential anomalies were eliminated. The successful implementation of over 30 interconnected entities—ranging from clinical workflows like diagnoses and prescriptions to administrative tasks like billing and facility management—demonstrates the system's comprehensive nature. Furthermore, the integration of strict business rules via SQL constraints guarantees data integrity, preventing errors such as negative financial values or illogical dates.

A significant technical achievement of this project was resolving complex relationship challenges, particularly the circular dependency between Patients and Appointments, which required a strategic approach to database construction. Additionally, our focus on ethical considerations, specifically the logical separation of clinical and administrative data, ensures that the system is not only functional but also compliant with data privacy standards.

Ultimately, this project solidified our understanding of how theoretical database concepts translate into real-world solutions. This database now serves as a scalable, reliable backend

foundation that can be expanded into a full-stack software application, significantly improving operational efficiency and patient care delivery.