NEPHROCARE:
AVANCED RENAL PATIENT MANAGEMENT SYSTEM

DBMS REPORT

CSE 212

DATABASE SYSTEM LAB

HELLFIRE CLUB

ID 23101154 AUNJOLEE NUSRATH ANU

ID 23101157 DALILA BINTE ALAM

SUBMITTED TO

ALIF RUSLAN

LECTURER, DEPARTMENT OF CSE, UAP

DATE OF SUBMITTION

6TH MAY 2025

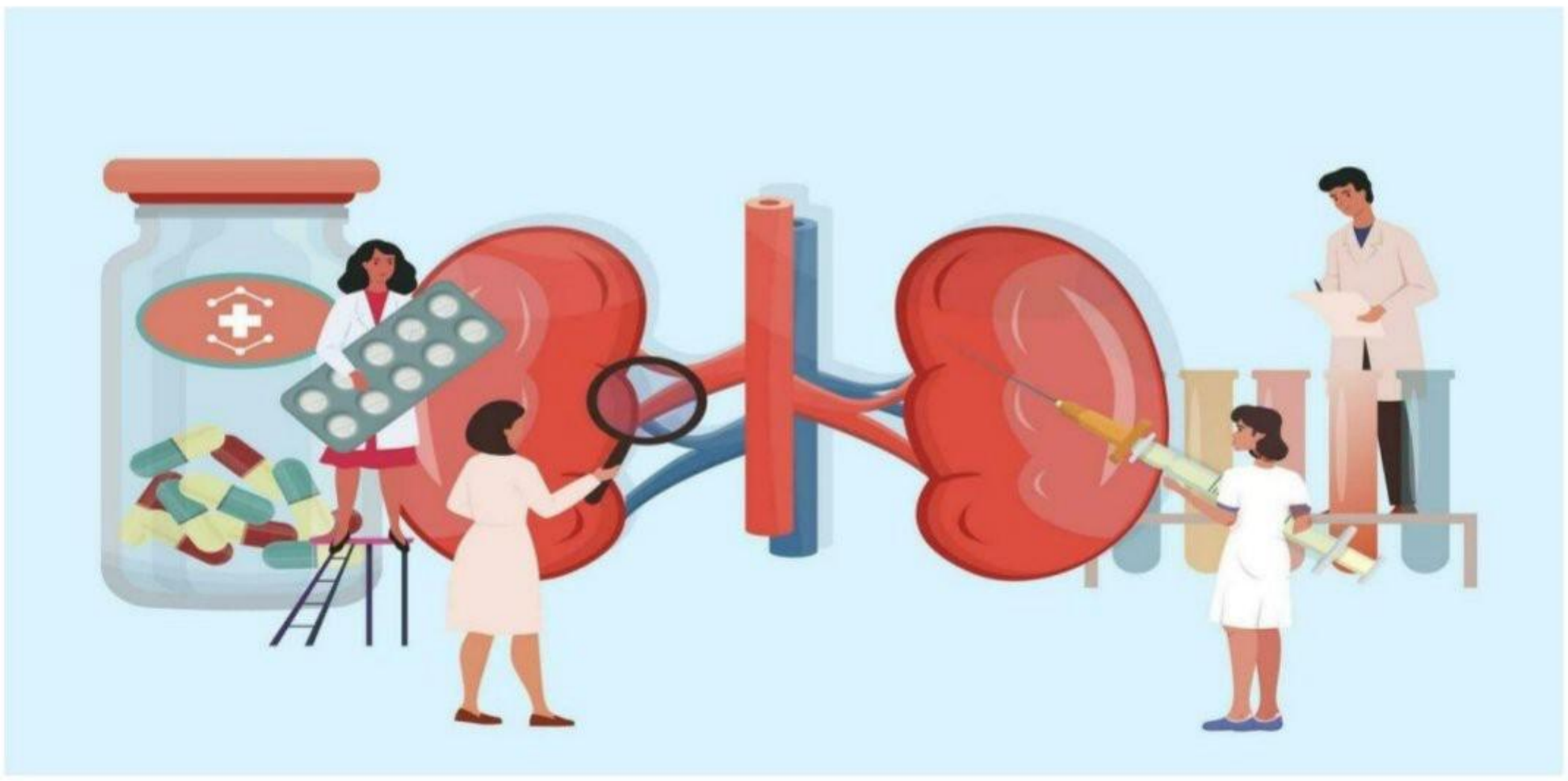
TABLE OF CONTENTS

PROJECT TITLE

NEPHROCARE: Advanced Renal Patient Management system

Introduction:

The **NephroCare Advanced Renal Patient Management System** is a meticulously engineered digital solution designed to revolutionize the continuum of care for individuals suffering from chronic kidney disease (CKD) and related renal disorders. Rooted in clinical precision and informed by nephrological best practices, this system encapsulates the complexities of renal patient care through an integrative, data-driven framework that streamlines diagnostics, treatment, and long-term management.



NephroCare unifies renal care teams through a secure system that streamlines patient data and supports informed, coordinated treatment. It reduces redundancy, enables early intervention, and elevates care standards through smart integration

Aims of the Project:

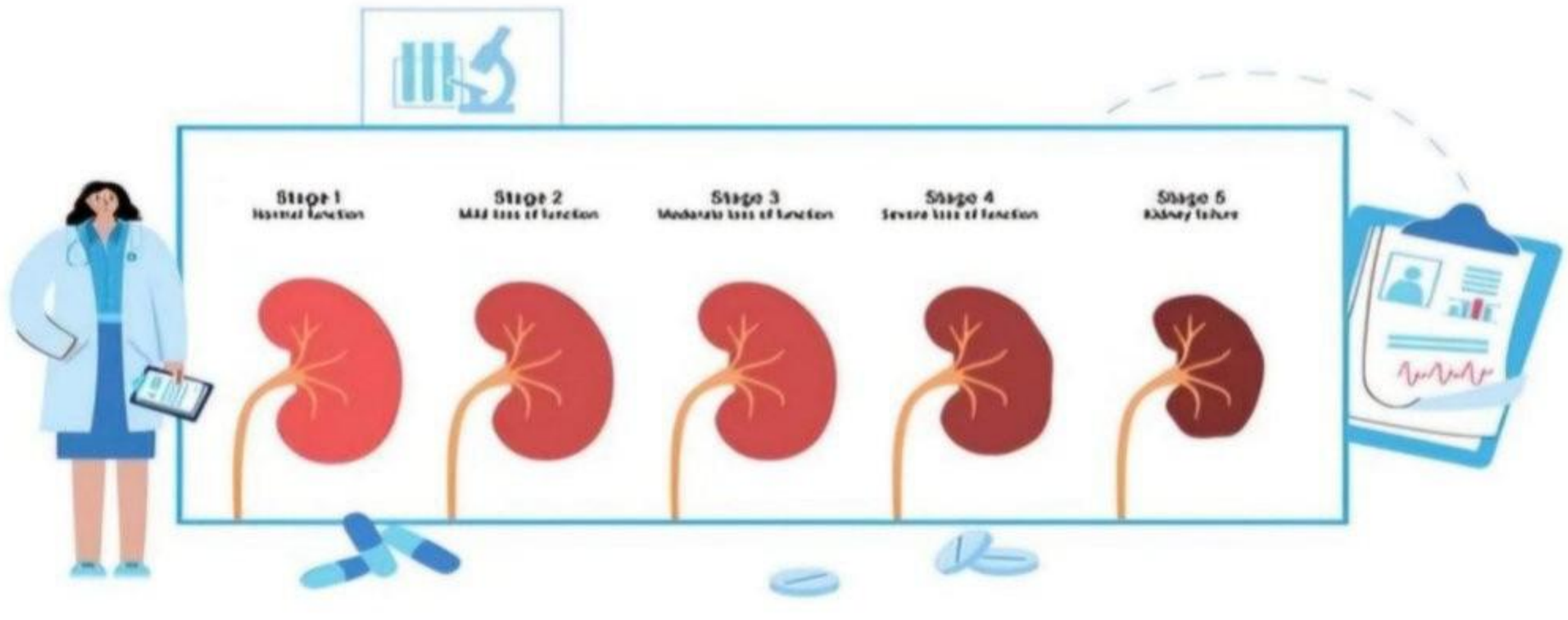
Here are the **project delivarables** for the *NephroCare Advanced Renal Patient Management System*:

- **Centralize Patient Data**: Integrate all relevant patient information (medical history, treatments, labs, nutrition) into a single, accessible platform.

- **Enhance Clinical Decision-Making**: Support nephrologists and care teams with up-to-date data for evidence-based diagnosis and treatment planning.

- **Improve Continuity of Care**: Facilitate seamless communication among healthcare providers involved in renal patient care.

- **Enable Early Intervention**: Track renal disease progression to allow timely actions that can delay or prevent end-stage complications.



- **Reduce Operational Redundancies**: Minimize duplication of tests, prescriptions, and procedures across departments.

- **Increase System Efficiency**: Streamline administrative workflows like appointments, follow-ups, and insurance documentation.

- **Ensure Data Security**: Maintain patient confidentiality through role-based access control and secure database design.

ENTITY OVERVIEW:PURPOSE AND FUCTIONALITY OF THE TABLES

1. Patients

Purpose: Central entity storing demographic, medical, and contact details of each renal patient.
Functionality:

- Tracks identity, health stats, allergies, renal stage, and insurance status.
- Supports linkage to treatments, consultations, appointments, and follow-ups.

2. Doctors

Purpose: Maintains credentials and specialization info of medical professionals.
Functionality:

- Links with treatment records.
- Helps assign doctors to patients based on expertise (e.g., nephrologist, cardiologist).

3. Appointments

Purpose: Schedules and logs patient visits with medical staff.
Functionality:

- Prevents overlaps, ensures timely treatment.
- Supports both in-person and telemedicine appointments.

4. Treatments

Purpose: Records medical procedures, diagnoses, and medications administered to patients.
Functionality:

- Logs treatment dates, outcomes, and follow-ups.
- Connects both Patients and Doctors through foreign keys.

5. Lab_Reports

- **Purpose:** Stores results of diagnostic tests such as blood tests, urine analysis, creatinine levels, etc.
- **Functionality:**
  - Helps track patient lab values over time.
  - Supports early detection of kidney dysfunction or complications.
  - Integrated into clinical decisions and treatment adjustments.

6. Billing

- **Purpose:** Manages billing information for all patient services (e.g., consultations, lab tests, dialysis, medications).
- **Functionality:**
  - Generates bills with itemized charges.
  - Links to payment and insurance tables.
  - Ensures financial transparency and hospital revenue tracking.

7. Dialysis_Sessions

- **Purpose:** Records each dialysis treatment session a patient undergoes.
- **Functionality:**
  - Tracks session details: date, time, duration, technician, machine used.
  - Ensures adherence to prescribed dialysis frequency.
  - Helps monitor patient response and complications.

8. Medication_History

- **Purpose:** Logs all medications prescribed/administered to a patient over time.
- **Functionality:**
  - Maintains a timeline of medication changes and dosages.
  - Supports medication reconciliation and allergy checks.
  - Assists in compliance monitoring and drug interaction management.

9. Transplant_History

- **Purpose:** Stores data on kidney transplant events for patients.
- **Functionality:**
  - Tracks transplant dates, donor type (living/deceased), outcomes.
  - Includes post-transplant monitoring details like graft survival.
  - Critical for long-term follow-up of transplant patients.

10. Nutrition_Consultations

- **Purpose:** Records nutritional guidance given to patients, especially those with CKD or post-transplant needs.
- **Functionality:**
    - Logs dietary plans, recommendations, and follow-ups.
    - Supports renal diet optimization (e.g., low sodium/potassium/phosphorus).
    - Linked to lab results and patient weight/fluid balance.

11. Kidney_Biopsy_Reports

- **Purpose:** Contains histopathology results of kidney biopsies for diagnostic evaluation.
- **Functionality:**
    - Stores microscopic findings, pathology interpretations, and clinical impressions.
    - Used to diagnose types of nephritis, rejection, or other renal diseases.
    - Guides targeted treatment decisions.

12. Renal_Risk_Assessments

- **Purpose:** Evaluates a patient's risk of developing or progressing kidney disease.
- **Functionality:**
    - Based on risk factors like hypertension, diabetes, genetic markers, lifestyle.
    - Helps prioritize preventive interventions and surveillance frequency.

DELINATION OF ER DIAGRAM RELATIONSHIP

Patients

- **Many-to-Many → Appointments**
  - Multiple patient can have multiple appointments.
  - Appointments (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **Many-to-Many → Billings**
  - Many patient can have multiple billing entries.
  - Billings (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Treatments**
  - One patient can undergo multiple treatments.
  - Treatments (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Lab_Reports**
  - One patient can receive multiple lab reports.
  - Lab_Reports (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Dialysis_Sessions**
  - One patient can attend multiple dialysis sessions.
  - Dialysis_Sessions (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Medication_History**
  - One patient can have multiple medication history records.
  - Medication_History (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Nutrition_Consultations**
    - One patient can receive multiple nutrition consultations.
    - Nutrition_Consultations (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Kidney_Biopsy**
    - One patient can have multiple kidney biopsy reports.
    - Kidney_Biopsy_Reports (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Renal_Risk_Assessments**
    - One patient can undergo multiple renal risk assessments.
    - Renal_Risk_Assessments (Patient_ID) → **FK** referencing Patients (Patient_ID)

- **One-to-Many → Transplant_History**
    - One patient can have multiple Transplant History.
    - Transplant_History (Patient_ID) → **FK** referencing Patients (Patient_ID)

Treatments

- **Many-to-Many → Billings**
    - Many treatments can have multiple billing entries.
    - Billings (Patient_ID) → **FK** referencing Patients (Patient_ID)

Doctors

- **One-to-Many → Appointments**
  - One doctor can attend to many appointments.
  - Appointments (Doctor_ID) → FK to Doctors (Doctor_ID)

- **One-to-Many → Treatments**
  - One doctor can administer many treatments.
  - Treatments (Doctor_ID) → FK to Doctors (Doctor_ID)

- **One-to-Many → Medication_History**
  - One doctor can prescribe many medications.
  - Medication_History (Doctor_ID) → FK to Doctors (Doctor_ID)

INVENTORY OF ATTRIBUTES

1.Patients

Stores information about patients

Name	Data Type	Description
Patient_id	INT,Primary Key, AUTO_INCREMENT	Unique identifier for each patient
name	VARCHAR(100)	Full name of patient
dateOfBirth	DATE	Patient's date of birth
gender	VARCHAR(10)	Gender of the patient(e.g,Male,Female,others)
contact	VARCHAR(15)	Patient's primary contact number
address	TEXT	Residential address of the patient
blood_group	VARCHAR(5)	Blood Group(e.g,A+,O-,AB+)
renel_stage	VARCHAR(50)	Current stage of renel disease (e.g, Stage 3 CKD)
insurance_id	VARCHAR(20)	Insurance policy number ,if applicable
emergency_contact	VARCHAR(15)	Emergency contact number
allergies	TEXT	Known allergies(e.g,Penicillin, Latex)
height_cm	INT	Height of the patients in centimeters
weight_kg	DECIMAL(5,2)	Weight of the patients in kilogram(e.g, 72.50)

2.Doctors

Stores information about doctors

Name	Data Type	Description
doctor_id	INT,Primary Key, AUTO_INCREMENT	Unique indentifier for each doctor
name	VARCHAR(100)	Full name of the doctor
specialization	VARCHAR(100)	Area of medical expertise (e.g,Nephrologist,Dialysis specialist)
phone	VARCHAR(15)	Contact number of the doctor
email	VARCHAR(100)	Email address of the docotor
experience_years	INT	Number of years of professional experience
qualification	VARCHAR(100)	Academic or professional qualification(e.g, MD,DM Nephrology

3.Appoinments

Stores appoinments recrods

Namr	Data Type	Description
appointment_id	INT,Primary Key, AUTO_INCREMENT	Unique identifier for each appoinments
patient_id	INT,Foreign Key (Patients(patient_id)	Reference the patient attending the appoinment
doctor_id	INT,Foreign Key (Doctors(doctor_id)	Reference the doctor handling the appoinment
appointment_date	DATE	Date on which the appointment is scheduled
purpose	TEXT	Reason for the appointment (e.g, consultation, follow-up)
status	VARCHAR(20)	Current status of the appointment(e.g, scheduled,completed,cancelled)
consultation_mode	VARCHAR(20)	Mode of consultation (e.g, In-person,Online,Telephonic)
notes	TEXT	Additional remarks or observations noted during the appointment

4.Treatments

Records of medical procedures

Name	Data Type	Description
treatment_id	INT, Primary Key, AUTO_INCREMENT	Unique identifier for each treatment entry
patient_id	INT,Foreign Key(Patients(patient_id))	Identifies the patient receiving the treatment
doctor_id	INT, Foreign Key (Doctors(doctor_id))	Identifies the doctor administering or supervising the treatment
treatment_date	DATE	Date when the treatment was provided
treatment_type	VARCHAR(100)	Type or category of treatment (e.g, Dialysis,Surgery,Medication)
description	TEXT	Detailed explanation of the treatment performed
outcome	VARCHAR(100)	Result or response to the treatment(e.g, Improved,No change, Complications)
medication_prescribed	TEXT	Medications recommended as part of of after the treatment
next_followup_date	DATE	Suggested or scheduled date for the next follow-up

5.Lab_Reports

Stores results of diagnostic texts

Name	Data Type	Description
report_id	INT,Primary Key, AUTO_INCREMENT	Unique identifier for each lab report
patient_id	INT,Foreign Key (Patients(patient_id)	References the patient for whom the report is generated
test_date	DATE	The date on which the lab tests were performed
creatinine	DECIMAL(4,2)	Measured level of creatinine in the blood (indicator of kidney function)
gfr	DECIMAL(5,2)	Glomerular Filtration Rate,a key measure of kidney filtration capacity
blood_pressure	VARCHAR(20)	Recorded blood pressure reading(e.g,120/80 mmHg)
hemoglobin	DECIMAL(4,2)	Hemoglobin level in the blood(indicator of anemia status)
potassium	DECIMAL(4,2)	Blood pressure level,relevant for electrolyte balance
phosphorus	DECIMAL(4,2)	Blood phosphorus level,monitored in kidney patients
sodium	DECIMAL(6,2)	Blood sodium level
notes	TEXT	----------

6.Billing

Stores billing information

Name	Data Type	Description
bill_id	INT, Primary Key, AUTO_INCREMENT	Unique identifier for each billing record
patient_id	INT,Foreign Key (Patients(patient_id))	References the patient associated with the bill
treatment_id	INT,Foreign Key (Treatments(treatment_id))	References the specific treatment for which bill is generated
amount	DECIMAL(10,2)	The total amount charged for the treatment or services
bill_date	DATE	The date when the bill was issued
paid-status	VARCHAR(20)	Indicates whether the bill is ‘paid’,’unpaid’,or ‘pending’
payment_mode	VARCHAR(20)	Specifies the method of payment (e.g, ‘cash’,’card’,’insurance’)

7.Dialysis_Sessions

Records each dialysis sessions

Name	Data Type	Description
session_id	INT,Primary Key, AUTO_INCREMENT	Unique identifiers for each dialysis session
patient_id	INT,Foreign Key (Patients(patient_id))	References the patient undergoing dialysis session
Session_date	DATE	Date on which dialysis session took place
Machine_id	VARCHAR(10)	Identifier of the dialysis machine used
duration_minutes	INT	Duration of the dialysis session in minutes
technician_name	VARCHAR(100)	Name of the technician who supervised the session
complications	TEXT	Notes of any complications or adverse events during the session
fluid_removed_ml	INT	Volume of fluid removed during the session
status	VARCHAR(20)	Status of the session (e.g, ‘completed’,’ongoing’,’cancelled’

8.Medication_History

Stores prescribed medicines

Name	Data type	Description
medication_id	INT, Primary Key AUTO_INCREMENT	Unique identifier for each medication entry in a patient's history
patient_id	INT,Foreign Key (Patients(patient_id))	References the patient who was prescribed for the medication
medication_name	VARCHAR(100)	The name of the prescribed medication
dosage	VARCHAR(50)	Dosage details(e.g," 10mg twice daily")
start_date	DATE	Date when the patient began taking the medication
end_date	DATE	Date when the medication was stopped
prescribed_by	INT,Foreign Key (Doctors(doctor_id))	ID of the doctor who prescribed the medication
side_effects	TEXT	Noted or potential side effects of the medication

9.Transplant_History

Stores information on kidney transplant

Name	Data Type	Description
transplant_id	INT,Primary Key, AUTO_INCREMENT	Unique identifier for each transplant record
patient_id	INT,Foreign Key (Patients(patient_id)	References the patient who underwent the transplant
donor_type	VARCHAR(50)	Type of the donor(e.g, living,deceased)
donor_relation	VARCHAR(100)	Relationship of the donor to the patient(e.g, sibling,spouse)
transplant_date	DATE	The date of the transplant
hospital_name	VARCHAR(100)	Name of the hospital where the transplant took place
rejection_incidents	INT	Number of incidents of transplant rejection
immunosupport_therapy	TEXT	Details of any immunosupport therapy the patient is receiving post-transplant

10.Nutrition_Consultations

Records nutritional guidance

Name	Data Type	Description
consultation_id	INT,Primary Key AUTO_INCREMENT	Unique identifier for each nutrition consultation record
patient_id	INT,Foreign Key (Patients(patient_id)	References the patient who attended the nutrition consultation
consult_date	DATE	The date of the consultation with the dietitian
dietitan_name	VARCHAR(100)	Name of the dietitian conducting the consultation
nutrition_plan	TEXT	Details of the nutrition plan provided to the patient
recommendations	TEXT	Any specific recommendations made by the dietitian regarding the patient's diet and nutrition
followup_date	DATE	The scheduled date for the next follow-up consultation

11.Kidney_Biopsy_Reports

Stores histopathology results

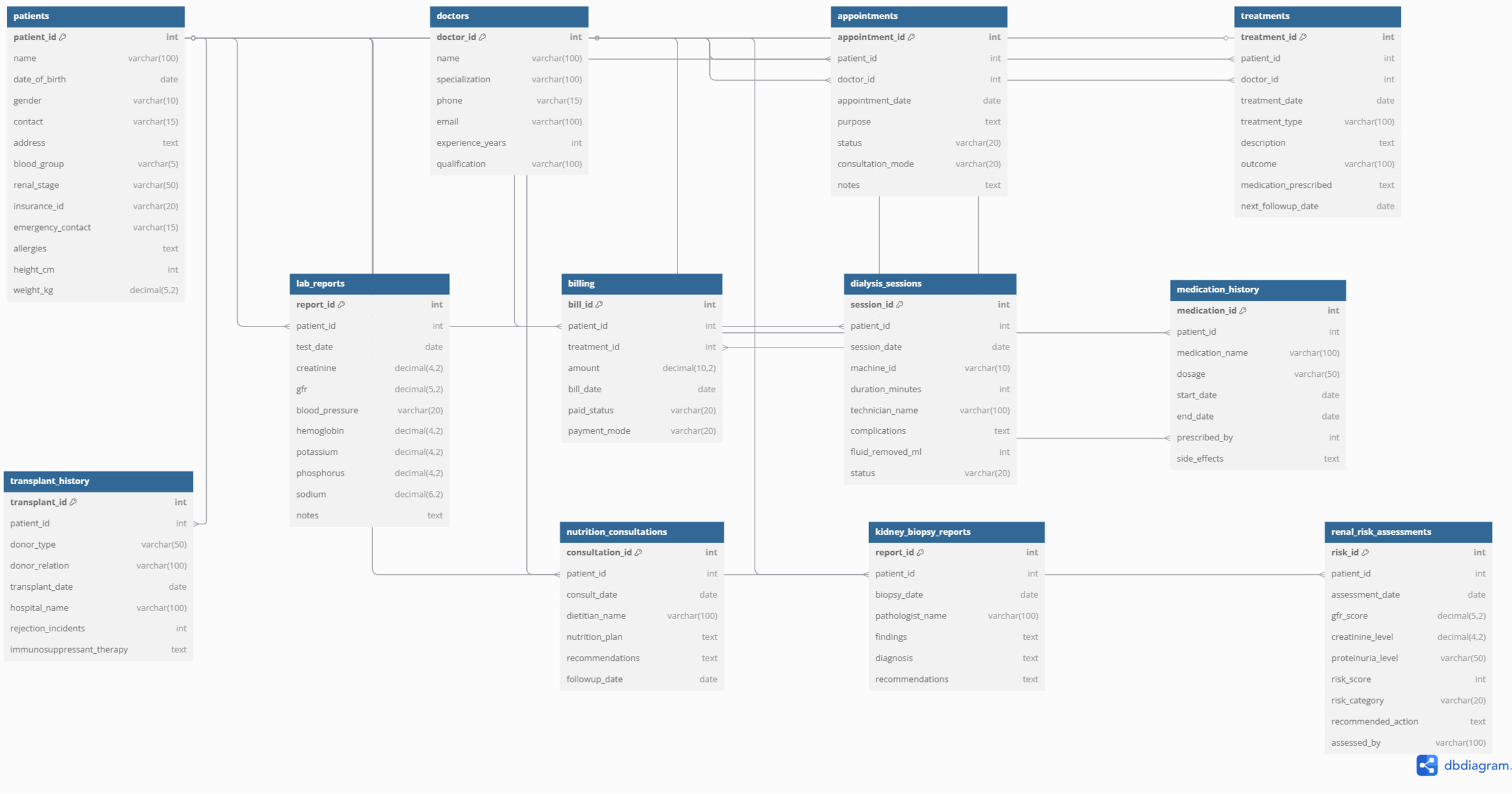
Name	Data Type	Description
report_id	INT, Primary Key AUTO_INCREMENT	Unique identifier for each kidney biopsy report
patient_id	INT,Foreign Key (Patients(patient_id))	References the patient for whom the kidney biopsy report is generated
biopsy_date	DATE	The date on which the kidney biopsy was performed
P_athologist_name	VARCHAR(100)	Name of the pathologist who examined the biopsy sample
findings	TEXT	Detailed findings based on the biopsy results
diagnosis	TEXT	Diagnosis made by the pathologist based on the biopsy report
recommendations	TEXT	Any follow-up recommendations provided by the pathologist

12.Renel_Risk_Assessments

Stores information of any risk

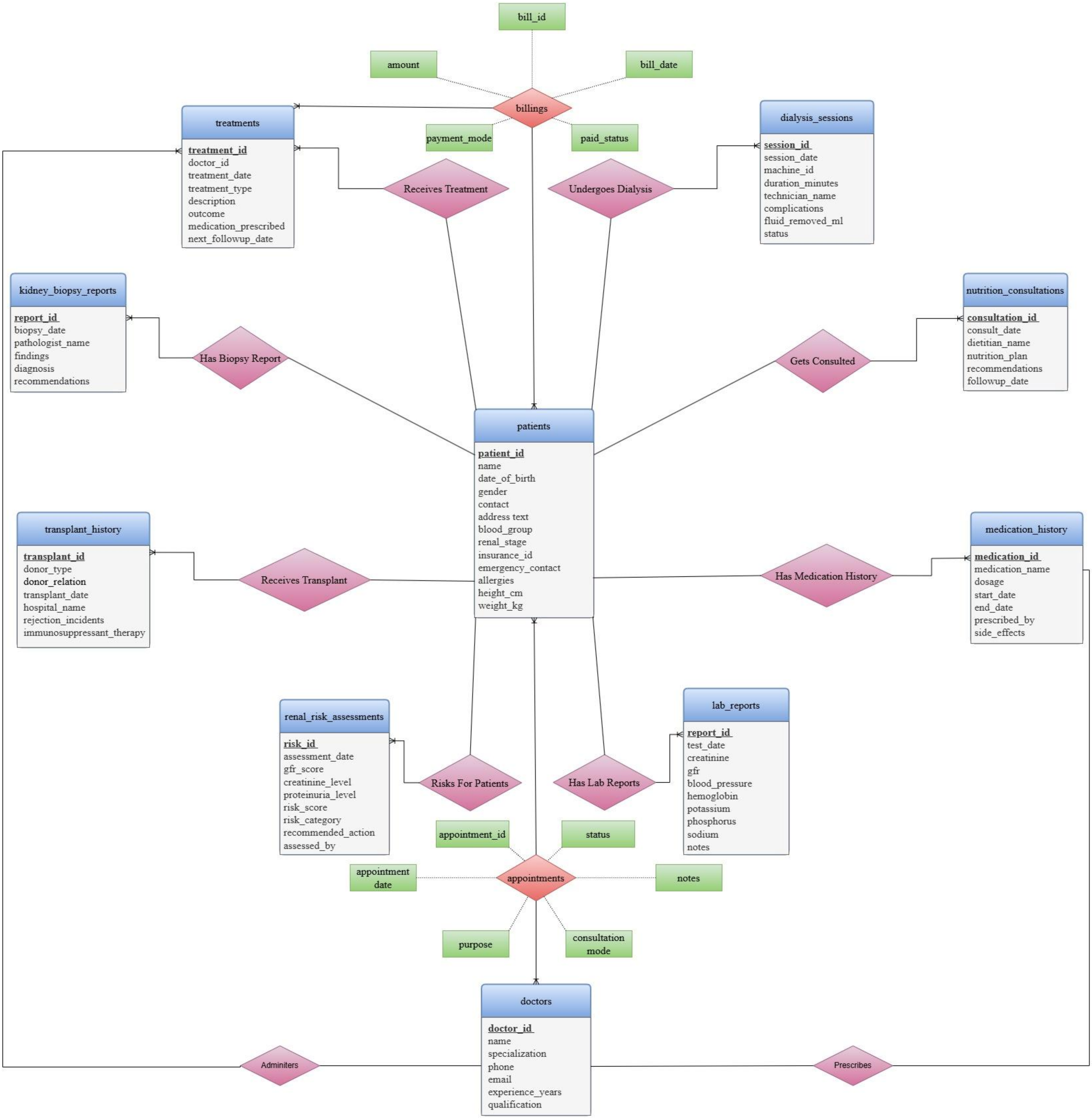
Name	Data Type	Description
risk_id	INT,Primary Key AUTO_INCREMENT	Unique identifier for each renel risk assessment
patient_id	INT,Foreign Key (Patients(patient_id))	References the patient for whom the renel risk assessment is conducted
assessment_date	DATE	The date when the renel risk assessment was performed
gfr_score	DECIMAL(5,2)	The estimated Glomerular Filtration Rate score, which helps assess kidney function
creatinint_level	DECIMAL(4,2)	The level of creatinine in the patient's blood,which is an important marker for kidney function
proteinuria_level	VARCHAR(50)	The level of protein in the urine,which can indicate kidney damage
risk_score	INT,CHECK:BETWEEN(0 and 100)	The calculated risk score based on the renal score assessment,indicating the likelihood of kidney failure
risk_category	VARCHAR(20)	Categorized the patient's risk level ,such as low, moderate or high
recommended_action	TEXT	Any recommendation for further action,treatments,or monitoring based on the assessment results
assessed_by	VARCHAR(100)	Name of the healthcare professionals who conducted the renel risk assessment

SCHEMA DIAGRAM



https://dbdiagram.io/d/6818b14a1ca52373f580752a

ER DIAGRAM


bill_id
amount
bill_date
billings
payment_mode
paid_status
treatments
treatment_id
doctor_id
treatment_date
treatment_type
description
outcome
medication_prescribed
next_followup_date
Receives Treatment
Undergoes Dialysis
dialysis_sessions
session_id
session_date
machine_id
duration_minutes
technician_name
complications
fluid_removed_ml
status
kidney_biopsy_reports
report_id
biopsy_date
pathologist_name
findings
diagnosis
recommendations
Has Biopsy Report
Gets Consulted
nutrition_consultations
consultation_id
consult_date
dietitian_name
nutrition_plan
recommendations
followup_date
patients
patient_id
name
date_of_birth
gender
contact
address text
blood_group
renal_stage
insurance_id
emergency_contact
allergies
height_cm
weight_kg
transplant_history
transplant_id
donor_type
donor_relation
transplant_date
hospital_name
rejection_incidents
immunosuppressant_therapy
Receives Transplant
Has Medication History
medication_history
medication_id
medication_name
dosage
start_date
end_date
prescribed_by
side_effects
renal_risk_assessments
risk_id
assessment_date
gfr_score
creatinine_level
proteinuria_level
risk_score
risk_category
recommended_action
assessed_by
Risks For Patients
Has Lab Reports
lab_reports
report_id
test_date
creatinine
gfr
blood_pressure
hemoglobin
potassium
phosphorus
sodium
notes
appointment_id
status
appointment date
appointments
notes
purpose
consultation mode
doctors
doctor_id
name
specialization
phone
email
experience_years
qualification
Adminiters
Prescribes

SQL QUERY

1. AND

- Fetches all columns for **male patients** who are in **Stage 4** of renal disease.
- If no patients meet both criteria, it will return an empty result.

SELECT * FROM Patients WHERE gender = 'Male' AND

renal_stage = 'Stage 4';

Screenshot of input and output

```
SELECT * FROM Patients WHERE gender = 'Male' AND renal_stage = 'Stage 4';
```

patient_id	name	dateOfBirth	gender	contact	address	blood_group	renal_stage	insurance_id	emergency_contact	alle
1	CGO MD AAMEER-UL-HAQUE	1980-08-16	MALE	01711934444	CANTONMENT	B+	Stage 2	INS001	01858147994	None

2. OR

- Fetches all patients who are **either female OR have O+ blood group**.
- Logical OR means a row qualifies if **either** condition is true.

```
SELECT * FROM Patients WHERE gender = 'Female' OR
blood_group = 'O+';
```

Screenshot of input and output

```
SELECT * FROM Patients WHERE gender = 'Female' OR blood_group = 'O+';
```

patient_id	name	dateOfBirth	gender	contact	address	blood_group	renal_stage	insurance_id	emergency_contact	all
2	lily collins	1992-07-05	Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655	Pen
4	julia roberts	1975-02-08	Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011	Sul
6	emma stone	2000-05-22	Female	9123456785	peru	A-	Stage 1	NULL	9876601234	Asp
8	nicole kidman	1995-04-14	Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789	Sea
9	lara jean	1992-07-05	Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655	Pen
11	julia mary	1975-02-08	Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011	Sul
13	jordan fisher	2000-05-22	Female	9123456785	peru	A-	Stage 1	NULL	9876601234	Asp
15	ryan gosling	1995-04-14	Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789	Sea

3.RETURNS SELECTED COLUMS

- Returns selected columns (patient_id, name, renal_stage) for patients in **Stage 3**, **Stage 4**, or **End Stage Renal Disease**.
- Uses IN for checking multiple values concisely.

```
SELECT patient_id, name, renal_stage
FROM Patients
WHERE renal_stage IN ('Stage 3', 'Stage 4', 'Stage 5 (ESRD)');
```

Screenshot of input and output

```
SELECT patient_id, name, renal_stage
FROM Patients
WHERE renal_stage IN ('Stage 3', 'Stage 4', 'Stage 5 (ESRD)');
```

patient_id	name	renal_stage
2	lily collins	Stage 5 (ESRD)
3	brad pitt	Stage 3
5	tom cruise	Stage 5 (ESRD)
7	hugh jackman	Stage 3
9	lara jean	Stage 5 (ESRD)
10	tom holland	Stage 3
12	noah centineo	Stage 5 (ESRD)

4.DELETE OPERATION

This deletes the patient with patient_id = 14, which is 'andrew garfeild'.

```
DELETE FROM Patients WHERE patient_id = 14;
```

Screenshot of input and output

```
DELETE FROM Patients WHERE patient_id = 14;
```

patient_id	name	dateOfBirth	gender	contact	address	blood_group	renal_stage	insurance_id	emergency_contact
1	CGO MD AAMEER-UL-HAQUE	1980-08-16	MALE	01711934444	CANTONMENT	B+	Stage 2	INS001	01858147994
2	lily collins	1992-07-05	Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655
3	brad pitt	1985-11-20	Male	9123456782	cambodia	A+	Stage 3	INS002	8877665544
4	julia roberts	1975-02-08	Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011
5	tom cruise	1969-09-18	Male	9123456784	italy	O-	Stage 5 (ESRD)	INS003	6655443322
6	emma stone	2000-05-22	Female	9123456785	peru	A-	Stage 1	NULL	9876601234
7	hugh jackman	1988-12-30	Male	9123456786	barcelona	B-	Stage 3	INS004	9123409876
8	nicole kidman	1995-04-14	Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789
9	lara jean	1992-07-05	Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655
10	tom holland	1985-11-20	Male	9123456782	cambodia	A+	Stage 3	INS002	8877665544
11	julia mary	1975-02-08	Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011
12	noah centineo	1969-09-18	Male	9123456784	italy	O-	Stage 5 (ESRD)	INS003	6655443322
13	jordan fisher	2000-05-22	Female	9123456785	peru	A-	Stage 1	NULL	9876601234
15	ryan gosling	1995-04-14	Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789

5. ALTER TABLE-ADD COLUMN

Adds a new column named insurance_provider to the existing Patients table to store the name of the insurance company/provider.

```
ALTER TABLE Patients ADD COLUMN
        insurance_provider   VARCHAR(100);
```

Screenshot of input and output

```
ALTER TABLE Patients ADD COLUMN insurance_provider VARCHAR(100);
```

gender	contact	address	blood_group	renal_stage	insurance_id	emergency_contact	allergies	height_cm	weight_kg	insurance_provider
MALE	01711934444	CANTONMENT	B+	Stage 2	INS001	01858147994	None	172	68.50	NULL
Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655	Penicillin	158	54.20	NULL
Male	9123456782	cambodia	A+	Stage 3	INS002	8877665544	Latex	180	80.00	NULL
Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011	Sulfa drugs	162	59.30	NULL
Male	9123456784	italy	O-	Stage 5 (ESRD)	INS003	6655443322	None	170	75.70	NULL
Female	9123456785	peru	A-	Stage 1	NULL	9876601234	Aspirin	155	48.60	NULL
Male	9123456786	barcelona	B-	Stage 3	INS004	9123409876	None	168	72.10	NULL
Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789	Seafood	161	56.80	NULL

Female	9123456781	jordan	O+	Stage 5 (ESRD)	NULL	9988776655	Penicillin	158	54.20	NULL
Male	9123456782	cambodia	A+	Stage 3	INS002	8877665544	Latex	180	80.00	NULL
Female	9123456783	egypt	AB-	Stage 2	NULL	7788990011	Sulfa drugs	162	59.30	NULL
Male	9123456784	italy	O-	Stage 5 (ESRD)	INS003	6655443322	None	170	75.70	NULL
Female	9123456785	peru	A-	Stage 1	NULL	9876601234	Aspirin	155	48.60	NULL
Female	9123456787	UAE	AB+	Stage 2	NULL	9345612789	Seafood	161	56.80	NULL

6. ALTER TABLE-DROP COLOUMN

- This removes the email column from the Doctors table.
- After this operation, that column no longer exists in the table, and its data lost.

ALTER TABLE Doctors DROP COLUMN email;

Screenshot of input and output

```
ALTER TABLE Doctors DROP COLUMN email;
```

doctor_id	name	specialization	phone	experience_years	qualification
1	Dr. rahtun zaman	Nephrologist	123-456-7890	15	MD
2	Dr. aunjolee nusrath anu	Cardiologist	123-555-7891	8	MD, FACC
3	Dr. fairuz anam	Endocrinologist	123-555-7892	2	MD, FACE
4	Dr. dalila alam	Internist	123-555-7893	19	MD
5	Dr. mehjabinusha	General Practitioner	123-555-7894	20	MBBS
6	Dr. reshawat hossain	Nephrologist	123-555-7895	7	MD
7	Dr. yeanur islam	Cardiologist	123-555-7896	14	MD, FACC
8	Dr. nuzhath kantomee	Endocrinologist	123-555-7897	18	MD, FACE

7. QUERY

- This returns **doctor names and years of experience** for those with **less than 10 years** of experience.
- Helps identify relatively **less experienced** doctors.

```
SELECT name, experience_years FROM Doctors

WHERE experience_years < 10;
```

Screenshot of input and output

```
SELECT name, experience_years
FROM Doctors
WHERE experience_years < 10;
```

name	experience_years
Dr. aunjolee nusrath anu	8
Dr. fairuz anam	2
Dr. reshawat hossain	7

8. LIKE

- Fetches the **names of doctors** whose names **start with 'Dr. a'**.
- The % is a wildcard that matches any sequence of characters.
- Useful for searching names by pattern (like autocomplete behavior).

SELECT name FROM Doctors

WHERE name LIKE 'Dr. a%';

Screenshot of input and output

```
SELECT name
FROM Doctors
WHERE name LIKE 'Dr. a%';
```

```
+---------------------------+
| name                      |
+---------------------------+
| Dr. aunjolee nusrath anu  |
+---------------------------+
```

9. ORDER BY(DESCENDING,ASCENDING)

This fetches **all columns and rows** from the Appointments table
This **sorts the results** using two criteria:

- **appointment_date DESC**:
  - Sorts appointments by date in **descending** order
- **status ASC**:
    - Within appointments that share the same date, sorts by **status in ascending (alphabetical) order**.
    - So statuses like 'Cancelled', 'Completed', 'Scheduled' will be ordered alphabetically if dates match.

```
SELECT * FROM Appointments

ORDER BY appointment_date DESC, status ASC;
```

Screenshot of input and output

```
SELECT * FROM Appointments
ORDER BY appointment_date DESC, status ASC;
```

appointment_id	patient_id	doctor_id	appointment_date	purpose	status	consultation_mode	notes
1	1	2	2025-04-10	Routine check-up	Completed	In-Person	Vitals stable.
2	2	1	2025-04-12	Post-transplant follow-up	Completed	In-Person	Continue immunosuppressants.
3	3	3	2025-04-15	New patient consultation	Completed	Virtual	Advised blood tests.
4	4	2	2025-04-17	Dialysis review	Completed	In-Person	Mild cramping observed.
5	5	4	2025-04-19	Medication adjustment	Cancelled	In-Person	Patient did not show.
6	6	1	2025-04-21	Nutrition advice	Completed	Virtual	Diet chart provided.
7	7	3	2025-04-23	Lab report review	Completed	In-Person	Creatinine levels rising.
8	8	4	2025-04-25	Renal risk assessment	Scheduled	Virtual	Awaiting test results.

appointment_id	patient_id	doctor_id	appointment_date	purpose	status	consultation_mode	notes
8	8	4	2025-04-25	Renal risk assessment	Scheduled	Virtual	Awaiting test results.
7	7	3	2025-04-23	Lab report review	Completed	In-Person	Creatinine levels rising.
6	6	1	2025-04-21	Nutrition advice	Completed	Virtual	Diet chart provided.
5	5	4	2025-04-19	Medication adjustment	Cancelled	In-Person	Patient did not show.
4	4	2	2025-04-17	Dialysis review	Completed	In-Person	Mild cramping observed.
3	3	3	2025-04-15	New patient consultation	Completed	Virtual	Advised blood tests.
2	2	1	2025-04-12	Post-transplant follow-up	Completed	In-Person	Continue immunosuppressants.
1	1	2	2025-04-10	Routine check-up	Completed	In-Person	Vitals stable.

10. COUNT (TREATMENT PER PATIENT)

- SELECT patient_id, COUNT(*) AS treatment_count:
  This selects each patient_id and counts how many treatment records exist for that patient.
  The result is labeled as treatment_count.
- FROM Treatments:
  Pulls data from the Treatments table.
- GROUP BY patient_id:
  Groups the treatments **per patient**, so each row in the result represents **one patient**.
- HAVING COUNT(*) > 1:
  Filters out any patients who have only **one treatment** — so only patients with **more than one treatment** will appear in the result.

  Helps identify patients under **ongoing care**, **multiple procedures**, or **follow-ups** — useful for treatment tracking.

```
SELECT patient_id, COUNT(*) AS treatment_count

FROM Treatments

GROUP BY patient_id

HAVING COUNT(*) > 1;
```

Screenshot of input and output

```
SELECT patient_id, COUNT(*) AS treatment_count
FROM Treatments
GROUP BY patient_id
HAVING COUNT(*) > 1;
```

patient_id	treatment_count
1	2
2	2

11. DROP TABLE

- This checks **if the table Lab_Reports exists**, and **deletes it** if it does.
- It prevents errors in case you try to drop a table that doesn't exist.

Once dropped:

- The **table structure** and **all data in it** are **permanently deleted**.
- You would have to recreate the table and reinsert the data if needed again.

DROP TABLE IF EXISTS Lab_Reports;

Screenshot of input and output

```
SELECT * FROM Lab_Reports;

DROP TABLE IF EXISTS Lab_Reports;
```

12. FINDING PATIENTS WHO HAVENT PAID

- This query retrieves **all patients** who have **no paid billing records**.
- It uses a **subquery** to select patient_ids from the Billing table where the paid_status = 'Paid', and excludes those.
- The result will be patients who either:
  - Only have unpaid bills, or
  - Have no billing records at all.
- Helps identify patients who still owe payment or haven't been billed yet.

SELECT * FROM Patients

WHERE patient_id NOT IN (

SELECT patient_id FROM Billing WHERE paid_status = 'Paid');

Screenshot of input and output

```
SELECT * FROM Patients
WHERE patient_id NOT IN (SELECT patient_id FROM Billing WHERE paid_status = 'Paid');
```

bill_id	patient_id	treatment_id	amount	bill_date	paid_status	payment_mode
1	1	1	5000.00	2025-04-20	Paid	Insurance
2	2	2	15000.00	2025-04-15	Unpaid	Cash
3	3	3	3000.00	2025-04-18	Paid	Credit Card
4	4	4	18000.00	2025-04-21	Unpaid	Insurance
5	5	5	2500.00	2025-04-22	Paid	Cash
6	6	6	4000.00	2025-04-23	Paid	Credit Card
7	7	7	12000.00	2025-04-25	Unpaid	Insurance
8	8	8	7500.00	2025-04-26	Paid	Cash

13. TOTAL AMOUNT(SUM)

This calculates the **total revenue (or billing)** from all patients by summing the amount column.

- The result is returned as a single value under the alias total_billing_amount.
- Useful for financial reporting or analytics dashboards.

```
SELECT SUM(amount) AS total_billing_amount FROM Billing;
```

Screenshot of input and output

```
SELECT SUM(amount) AS total_billing_amount FROM Billing;
```

```
+-----------------------+
| total_billing_amount |
+-----------------------+
|              67000.00 |
+-----------------------+
```

14. UPDATE

- This query **modifies** one or more rows in the Dialysis_Sessions table.
- It sets the machine_id to 'MCH100000' **only** for those rows where technician_name is 'Tech dalala'.
- In your dataset, this applies to the record with session_id = 4.
- Useful when correcting machine IDs, assigning a new machine, or fixing data entry errors.

```
update Dialysis_Sessions
set machine_id='MCH100000' WHERE technician_name='Tech dalala';
```

Screenshot of input and output

```
update Dialysis_Sessions
  set machine_id='MCH100000' WHERE technician_name='Tech dalala';
```

session_id	patient_id	session_date	machine_id	duration_minutes	technician_name	complications	fluid_removed_ml	status
1	1	2025-04-01	MCH1001	240	Tech takan	None	2500	Completed
2	2	2025-04-02	MCH1002	235	Tech anu	Mild cramps	2300	Completed
3	3	2025-04-03	MCH1003	245	Tech fairuz	Hypotension	2600	Completed
4	4	2025-04-04	MCH100000	220	Tech dalala	None	2200	Completed
5	5	2025-04-05	MCH1001	240	Tech knatom	Mild headache	2550	Completed
6	6	2025-04-06	MCH1002	230	Tech rusho	None	2400	Completed
7	7	2025-04-07	MCH1003	255	Tech bhugi	Fatigue	2650	Completed
8	8	2025-04-08	MCH1004	250	Tech Sara	None	2450	Completed

15. DUPLICATE

- This query creates a **new table** called Dialysis_Sessions_Copy.
- It **copies all the data** from the current Dialysis_Sessions table into this new one.
- This is useful for creating **backups**, **snapshots**, or **temporary working copies**.

Important Notes:

- The new table (Dialysis_Sessions_Copy) will **not** copy constraints, indexes, or primary/foreign keys — just the data and column structure.
- If you want to preserve constraints as well, you'd need to define them manually.

```
CREATE TABLE Dialysis_Sessions_Copy AS

SELECT * FROM Dialysis_Sessions;


SELECT * FROM Dialysis_Sessions_Copy;
```

Screenshot of input and output

```
CREATE TABLE Dialysis_Sessions_Copy AS
SELECT * FROM Dialysis_Sessions;

SELECT * FROM Dialysis_Sessions_Copy;
```

session_id	patient_id	session_date	machine_id	duration_minutes	technician_name	complications	fluid_removed_ml	status
1	1	2025-04-01	MCH1001	240	Tech takan	None	2500	Completed
2	2	2025-04-02	MCH1002	235	Tech anu	Mild cramps	2300	Completed
3	3	2025-04-03	MCH1003	245	Tech fairuz	Hypotension	2600	Completed
4	4	2025-04-04	MCH100000	220	Tech dalala	None	2200	Completed
5	5	2025-04-05	MCH1001	240	Tech knatom	Mild headache	2550	Completed
6	6	2025-04-06	MCH1002	230	Tech rusho	None	2400	Completed
7	7	2025-04-07	MCH1003	255	Tech bhugi	Fatigue	2650	Completed
8	8	2025-04-08	MCH1004	250	Tech Sara	None	2450	Completed

16. SUBQUERY

- This query retrieves **all rows** from Medication_History where the medication_name is the **same** as that of patient with patient_id = 3.
- In your data:
- patient_id = 3 is taking **Erythropoietin**.
- So the query returns **all rows where medication_name = 'Erythropoietin'**.
- That matches both:
  - patient_id = 3 (3000 IU)
  - patient_id = 8 (2000 IU)
  - To find other patients on the same medication as patient 3.

SELECT * FROM Medication_History

WHERE medication_name = (SELECT medication_name

FROM Medication_History WHERE patient_id = 3);

Screenshot of input and output

```
SELECT * FROM Medication_History
WHERE medication_name = (
    SELECT medication_name
    FROM Medication_History
    WHERE patient_id = 3
);
```

medication_id	patient_id	medication_name	dosage	start_date	end_date	prescribed_by	side_effects
3	3	Erythropoietin	3000 IU	2025-04-10	2025-05-10	3	Headache, nausea
8	8	Erythropoietin	2000 IU	2025-05-01	2025-06-01	8	No side effects

17. GROUP BY

- This query groups all transplant records based on donor_type (e.g., *Living Donor*, *Deceased Donor*).
- It then counts how many transplants fall under each type.

The result shows:

- How many *Living Donor* transplants were performed
- How many *Deceased Donor* transplants were performed.
- Summary report for transplant types — useful for audits, medical reports, or policy analysis.

```
SELECT donor_type, COUNT(*) AS total_transplants
FROM Transplant_History GROUP BY donor_type;
```

Screenshot of input and output

```
SELECT donor_type, COUNT(*) AS total_transplants
FROM Transplant_History
GROUP BY donor_type;
```

donor_type	total_transplants
Living Donor	5
Deceased Donor	3

18. DATE FILTERING

- This query retrieves all consultations where the **follow-up date is in April 2025**.
- The BETWEEN clause is **inclusive**, so it includes:

  Dates on **April 1st** and **April 30th**, as well as everything in between.

- Useful for generating an **April follow-up schedule**.

```
SELECT * FROM Nutrition_Consultations
WHERE followup_date BETWEEN '2025-04-01' AND '2025-04-30';
```

Screenshot of input and output

```
SELECT * FROM Nutrition_Consultations
WHERE followup_date BETWEEN '2025-04-01' AND '2025-04-30';
```

consultation_id	patient_id	consult_date	dietitian_name	nutrition_plan	recommendations
1	1	2025-03-01	munira mustarin	Low-protein diet for renal health	Increase water intake, reduce sodium
2	2	2025-03-05	shanjida neela	Renal disease management diet	Increase potassium-rich foods, reduce phosphorus
3	3	2025-03-10	tanjida muna	Diabetic kidney disease diet	Moderate protein intake, limit sugar
4	4	2025-03-15	shameem sadat	High-fiber renal diet	Increase fiber, avoid processed foods
5	5	2025-03-20	mikail hossain	Low-salt kidney diet	Reduce sodium, focus on fresh vegetables
6	6	2025-03-25	akash neerjhor	Heart-healthy renal diet	Increase omega-3 intake, limit saturated fats
7	7	2025-03-30	reshawat hossain	Renal transplant diet	Avoid high-calcium foods, take vitamins as prescribed

19. CASE LOGIC

- Selects all key biopsy report data and **adds a derived column** called severity_level.
- This new column **categorizes biopsy severity** based on keywords in the findings field using a CASE expression.

CASE Logic Breakdown:

- If findings contains:
  - 'severe' → outputs 'High Severity'
  - 'moderate' → outputs 'Moderate Severity'
  - 'mild' → outputs 'Low Severity'
  - 'no significant abnormalities' → outputs 'Normal'
  - If none match → 'Uncategorized'
- This helps quickly **classify the severity** of kidney conditions from text data in a readable format.

```
SELECT
    report_id,
    patient_id,
    biopsy_date,
    pathologist_name,
    findings,
    diagnosis,
    recommendations,
    CASE
      WHEN findings LIKE '%severe%' THEN 'High Severity'
      WHEN findings LIKE '%moderate%' THEN 'Moderate Severity'
```

```
        WHEN findings LIKE '%mild%' THEN 'Low Severity'

        WHEN findings LIKE '%no significant abnormalities%' THEN
'Normal'

      ELSE 'Uncategorized'

    END AS severity_level

FROM Kidney_Biopsy_Reports;
```

Screenshot of input and output

```
SELECT
    report_id,
    patient_id,
    biopsy_date,
    pathologist_name,
    findings,
    diagnosis,
    recommendations,
    CASE
        WHEN findings LIKE '%severe%' THEN 'High Severity'
        WHEN findings LIKE '%moderate%' THEN 'Moderate Severity'
        WHEN findings LIKE '%mild%' THEN 'Low Severity'
        WHEN findings LIKE '%no significant abnormalities%' THEN 'Normal'
        ELSE 'Uncategorized'
    END AS severity_level
FROM Kidney_Biopsy_Reports;
```

report_id	patient_id	biopsy_date	pathologist_name	findings	diagnosis	recommendations	severity_level
1	1	2025-03-01	Dr. Aunjolee	Minimal fibrosis, mild inflammation	Chronic glomerulonephritis	Start immunosuppressive therapy	Low Severity
2	2	2025-03-05	Dr. Nusrath	Moderate scarring, mild cellular infiltration	Focal segmental glomerulosclerosis	Monitor GFR, adjust medication	Moderate Severity
3	3	2025-03-10	Dr. Anu	No significant abnormalities	Normal kidney tissue	Continue current treatment	Normal
4	4	2025-03-15	Dr. Nuzhath	Severe glomerular damage, moderate fibrosis	IgA nephropathy	Consider a renal transplant evaluation	High Severity
5	5	2025-03-20	Dr. Tasnim	Minimal changes, mild proteinuria	Minimal change disease	Review medication dosage	Low Severity
6	6	2025-03-25	Dr. Kantomee	Patchy inflammation, some fibrosis	Diabetic nephropathy	Aggressive blood sugar management	Uncategorized
7	7	2025-03-30	Dr. Faiz	Severe tubular atrophy, advanced fibrosis	Chronic kidney disease stage 3	Evaluate dialysis options	High Severity
8	8	2025-04-01	Dr. Ahmed	Mild mesangial expansion, no fibrosis	Hypertensive nephropathy	Tighten blood pressure control	Low Severity

20. RISK MESSAGE ANALYSIS

This query selects:

- Patient ID
- Assessment Date
- GFR score
- Risk score
- Risk category
- A **custom risk_message** generated based on the risk_score using a CASE statement.

CASE Breakdown:

- **0–10** → "Excellent kidney function"
- **11–20** → "Monitor regularly"
- **21–30** → "Immediate medical attention advised"
- **>30 (hypothetically)** → "Critical condition – urgent dialysis review"

None of the current risk_score values are >30, but the ELSE clause provides a safety net for future high-risk entries.

```
SELECT
patient_id,
assessment_date,
gfr_score,
risk_score,
risk_category,
CASE
    WHEN risk_score BETWEEN 0 AND 10 THEN 'Excellent kidney
function'
```

WHEN risk_score BETWEEN 11 AND 20 THEN 'Monitor regularly'

WHEN risk_score BETWEEN 21 AND 30 THEN 'Immediate medical attention advised'

ELSE 'Critical condition – urgent dialysis review'

END AS risk_message

FROM Renal_Risk_Assessments;

Screenshot of input and output

```
SELECT
    patient_id,
    assessment_date,
    gfr_score,
    risk_score,
    risk_category,
    CASE
        WHEN risk_score BETWEEN 0 AND 10 THEN 'Excellent kidney function'
        WHEN risk_score BETWEEN 11 AND 20 THEN 'Monitor regularly'
        WHEN risk_score BETWEEN 21 AND 30 THEN 'Immediate medical attention advised'
        ELSE 'Critical condition - urgent dialysis review'
    END AS risk_message
FROM Renal_Risk_Assessments;
```

patient_id	assessment_date	gfr_score	risk_score	risk_category	risk_message
1	2025-03-01	45.50	10	Low Risk	Excellent kidney function
2	2025-03-05	35.20	20	Moderate Risk	Monitor regularly
3	2025-03-10	50.00	8	Low Risk	Excellent kidney function
4	2025-03-15	40.75	15	Low Risk	Monitor regularly
5	2025-03-20	38.30	30	High Risk	Immediate medical attention advised
6	2025-03-25	42.50	12	Low Risk	Monitor regularly
7	2025-03-30	41.60	18	Moderate Risk	Monitor regularly
8	2025-04-01	44.20	10	Low Risk	Excellent kidney function

PRAGMATIC IMPORTANCE OF NEPHROCARE

NephroCare plays a vital role in kidney healthcare by improving patient outcomes, operational efficiency, and fostering a comprehensive care approach. Here's how it impacts kidney care:

1. Patient-Centered Care

NephroCare centralizes patient data for **personalized care**, improving decision-making and health outcomes.

2. Chronic Kidney Disease Management

It tracks disease progression, allowing for **proactive adjustments** in treatment and **long-term management**.

3. Dialysis Treatment Optimization

Streamlines **dialysis session management**, ensuring **effective treatment** and **complication prevention**.

4. Interdisciplinary Collaboration

Facilitates seamless communication between healthcare providers, promoting **coordinated care**.

5. Accurate Documentation

Ensures **detailed, accurate medical records**, improving **clinical transparency** and **legal compliance**.

6. Early Detection

Monitors biomarkers for **early disease detection**, enabling timely interventions to **slow progression**.

7. Billing and Financial Management

Automates billing processes, reducing **errors** and speeding up **reimbursement**.

8. Support for Research

Aggregates data for **clinical studies** and **evidence-based practices**, advancing nephrology care.

9. Patient Empowerment

Gives patients access to their **health data**, encouraging **active participation** in their care.

10. Cost Reduction

Improves **resource management**, reducing **hospital readmissions** and overall healthcare costs.

In essence, **NephroCare** is far more than just a management system—it is a catalyst for **enhanced patient outcomes**, **interdisciplinary collaboration**, and **data-driven clinical decision-making**

COMPLEX ENGINEERING PROBLEM

Mapping The Knowledge Profiles (Ks) To The NEPHROCARE: Advanced Renal Patient Management System

Level	Cognitive Level	Description	Key Action Verbs	Example in NEPHROCARE
K2	Remembering	Recall engineering knowledge and facts.	Define, Identify, List	Identify fields in patients, like blood group, emergency contact.
K3	Applying	Apply principles and concepts in real contexts.	Apply, Use, Implement	Apply SQL joins to retrieve lab and treatment data.
K5	Creating	Design engineering solutions and models.	Design, Create, Develop	Design the complete NephroCare ER model.
K6	Analyzing	Analyze problems and system behavior.	Analyze, Debug, Examine	Analyze causes of data redundancy in billing and appointments.
K8	Understanding	Interpret technical research and specifications.	Describe, Explain, Interpret	Explain how GFR and creatinine values influence treatment plans.

Mapping The Complex Engineering Problems (Ps) To The NEPHROCARE: Advanced Renal Patient Management System

Level	Cognitive Level	Description	Key Action Verbs	Example in NEPHROCARE
P1	Evaluating	Assess depth and breadth of knowledge required for problems.	Evaluate, Judge, Justify	Evaluate schema coverage for managing dialysis and transplant history.
P3	Analyzing	Analyze unfamiliar problems across multiple tables and modules.	Analyze, Organize, Compare	Analyze relationship between lab_reports and treatments.
P4	Applying	Apply known solutions to new healthcare domains.	Apply, Solve, Demonstrate	Apply query logic to manage patient appointments and follow-ups.
P6	Creating	Devise integrated solutions for stakeholders with conflicting needs.	Design, Formulate, Integrate	Create a unified patient profile used by doctors, Nutritionists, and lab techs.
P7	Evaluating	Judge the reliability and effectiveness of interdependent components.	Evaluate, Reflect, Prioritize	Evaluate how changes in treatment affect billing and follow-ups.

Mapping The Complex Engineering Activities (As) To The NEPHROCARE: Advanced Renal Patient Management System

Level	Cognitive Level	Description	Key Action Verbs	Example in NEPHROCARE
A1	Understanding	Understand how various engineering resources function in real-world systems.	Describe, Summarize, Interpret	Describe the flow from appointments to billing and treatment.
A2	Applying	Demonstrate coordination across components and real-time implementation.	Execute, Demonstrate, Operate	Implement synchronization between patient visits and lab test updates.
A3	Creating	Design innovative solutions with realistic constraints and user needs.	Design, Invent, Develop	Develop a custom dialysis session tracker with machine and status info.
A5	Evaluating	Assess project viability, scope, and applicability in real-world scenarios.	Validate, Evaluate, Assess	Assess if NephroCare can be deployed at a hospital-level scale.

https://github.com/aunjoleenusrath/Nephrocare