HOSPITAL INFECTION CONTROL ANALYSIS

NAME – AANCHAL RAJENDRA PANWAR

EMAIL ADDRESS – panwaraanchal047@gmail

DATE OF SUBMISSION – 22/12/2025

PROJECT AIMS:

1 Identify Hospital-Acquired Infections (HAIs)

To distinguish infections acquired **after hospital admission** from pre-existing infections by analyzing admission and infection detection dates.

---

2 Detect High-Risk Wards

To analyze infection rates across different hospital wards (ICU, General, Oncology, etc.) and identify areas requiring stricter infection control measures.

---

3 Analyze Length of Stay vs Infection Risk

To evaluate whether longer hospital stays increase the probability of patients acquiring infections.

---

4 Track Repeat Infections

To identify patients with multiple infection occurrences using SQL window functions, helping assess treatment effectiveness and hygiene standards.

---

5 Support Data-Driven Infection Prevention

To provide actionable insights that help hospital administrators improve sanitation protocols, resource allocation, and patient safety.

PROJECT OBJECTIVES

The objective of the **Hospital Infection Control Analytics** project is to analyze patient admission, ward allocation, and infection data using SQL in order to identify hospital-acquired infections and understand infection patterns within the hospital. This project aims to determine high-risk wards by calculating infection rates, evaluate the relationship between length of hospital stay and infection occurrence, and detect patients with repeat infections using window functions. By classifying infections as hospital-acquired or pre-existing and generating meaningful insights, the project supports data-driven decision-making to improve infection control measures, enhance patient safety, and optimize hospital hygiene and resource management practices.

ERR DIAGRAM

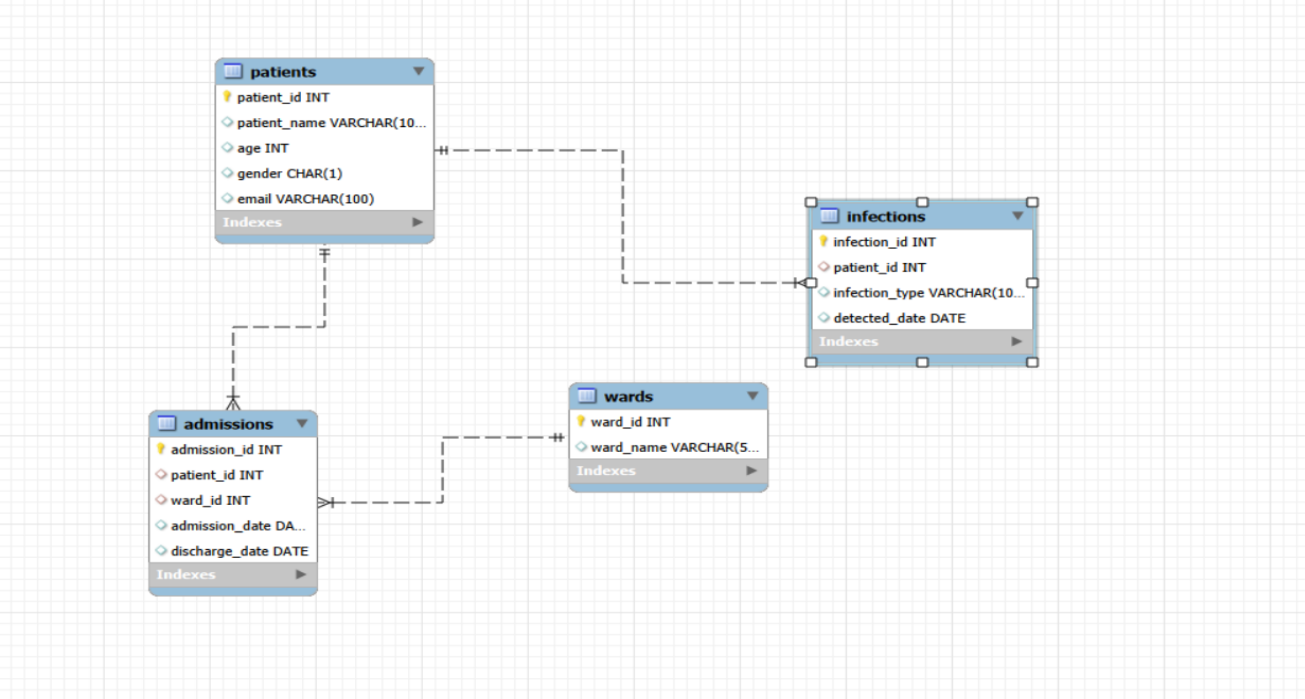


TABLE DESCRIPTION

1. PATIENTS TABLE

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
patient_id	int	NO	PRI	NULL	auto_increment
patient_name	varchar(100)	YES		NULL	
age	int	YES		NULL	
gender	char(1)	YES		NULL	
email	varchar(100)	YES		NULL	

2. WARDS TABLE

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

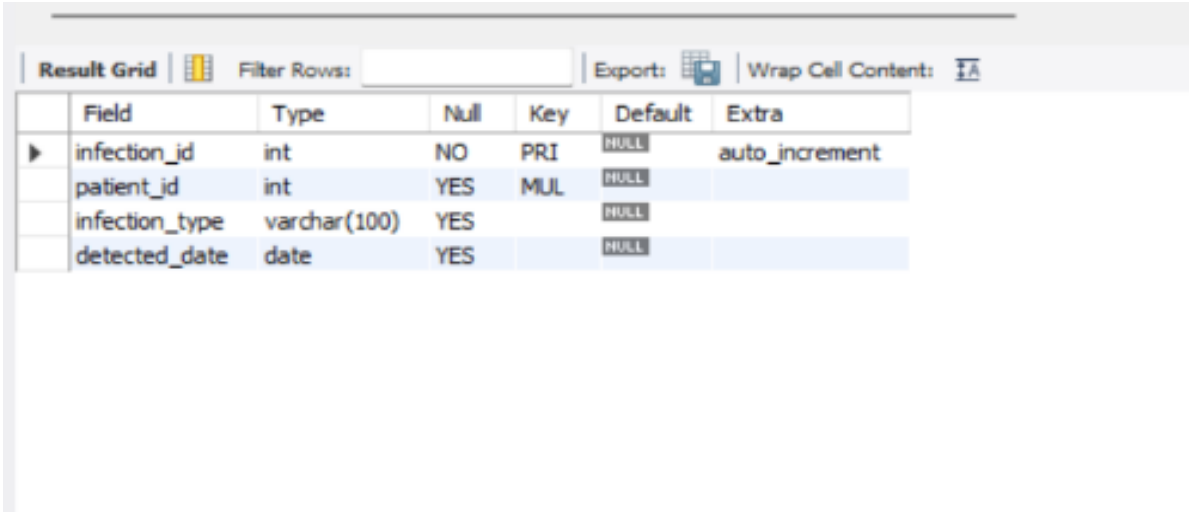
Field	Type	Null	Key	Default	Extra
ward_id	int	NO	PRI	NULL	auto_increment
ward_name	varchar(50)	YES		NULL	

3. ADMISSIONS TABLE

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Field	Type	Null	Key	Default	Extra
admission_id	int	NO	PRI	NULL	auto_increment
patient_id	int	YES	MUL	NULL	
ward_id	int	YES	MUL	NULL	
admission_date	date	YES		NULL	
discharge_date	date	YES		NULL	

4. INFECTIONS TABLE



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

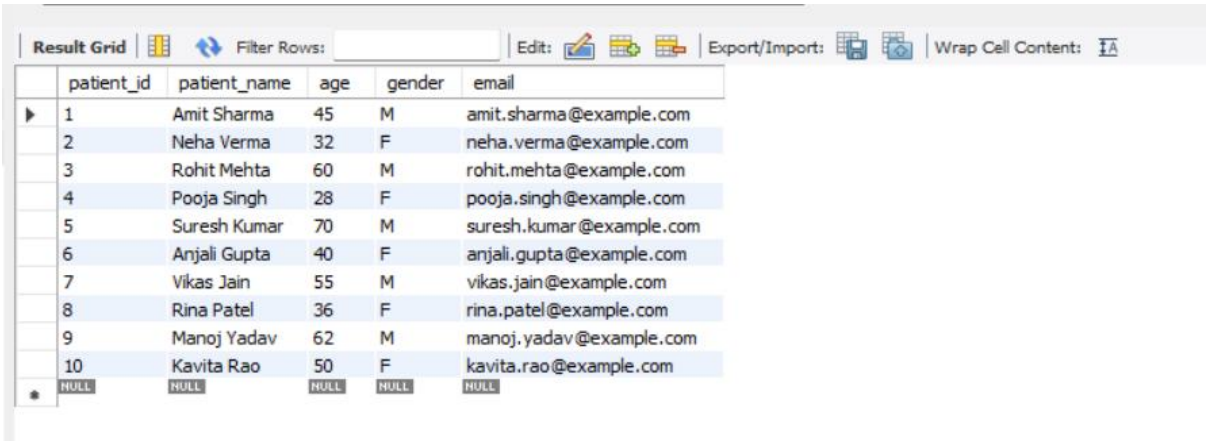
Field	Type	Null	Key	Default	Extra
infection_id	int	NO	PRI	NULL	auto_increment
patient_id	int	YES	MUL	NULL	
infection_type	varchar(100)	YES		NULL	
detected_date	date	YES		NULL	

COMMANDS

Patients Table:

```
CREATE TABLE patients (
    patient_id INT AUTO_INCREMENT PRIMARY KEY,
    patient_name VARCHAR(100),
    age INT,
    gender CHAR(1),
    email VARCHAR(100)
);
INSERT INTO patients (patient_name, age, gender, email) VALUES
('Amit Sharma', 45, 'M', 'amit.sharma@example.com'),
('Neha Verma', 32, 'F', 'neha.verma@example.com'),
('Rohit Mehta', 60, 'M', 'rohit.mehta@example.com'),
('Pooja Singh', 28, 'F', 'pooja.singh@example.com'),
('Suresh Kumar', 70, 'M', 'suresh.kumar@example.com'),
('Anjali Gupta', 40, 'F', 'anjali.gupta@example.com'),
('Vikas Jain', 55, 'M', 'vikas.jain@example.com'),
('Rina Patel', 36, 'F', 'rina.patel@example.com'),
('Manoj Yadav', 62, 'M', 'manoj.yadav@example.com'),
('Kavita Rao', 50, 'F', 'kavita.rao@example.com');
```

OUTPUT:

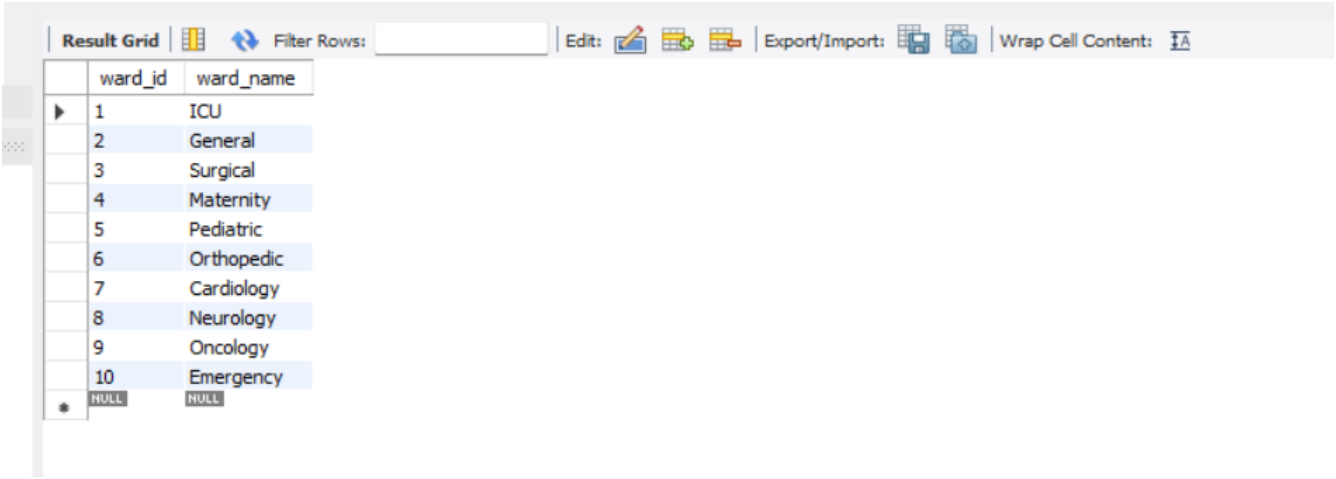


Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

patient_id	patient_name	age	gender	email
1	Amit Sharma	45	M	amit.sharma@example.com
2	Neha Verma	32	F	neha.verma@example.com
3	Rohit Mehta	60	M	rohit.mehta@example.com
4	Pooja Singh	28	F	pooja.singh@example.com
5	Suresh Kumar	70	M	suresh.kumar@example.com
6	Anjali Gupta	40	F	anjali.gupta@example.com
7	Vikas Jain	55	M	vikas.jain@example.com
8	Rina Patel	36	F	rina.patel@example.com
9	Manoj Yadav	62	M	manoj.yadav@example.com
10	Kavita Rao	50	F	kavita.rao@example.com
NULL	NULL	NULL	NULL	NULL

```
CREATE TABLE wards (
   ward_id INT AUTO_INCREMENT PRIMARY KEY,
   ward_name VARCHAR(50)
);
INSERT INTO wards (ward_id, ward_name) VALUES
(1, 'ICU'),
(2, 'General'),
(3, 'Surgical'),
(4, 'Maternity'),
(5, 'Pediatric'),
(6, 'Orthopedic'),
(7, 'Cardiology'),
(8, 'Neurology'),
(9, 'Oncology'),
(10, 'Emergency');
```

OUTPUT:



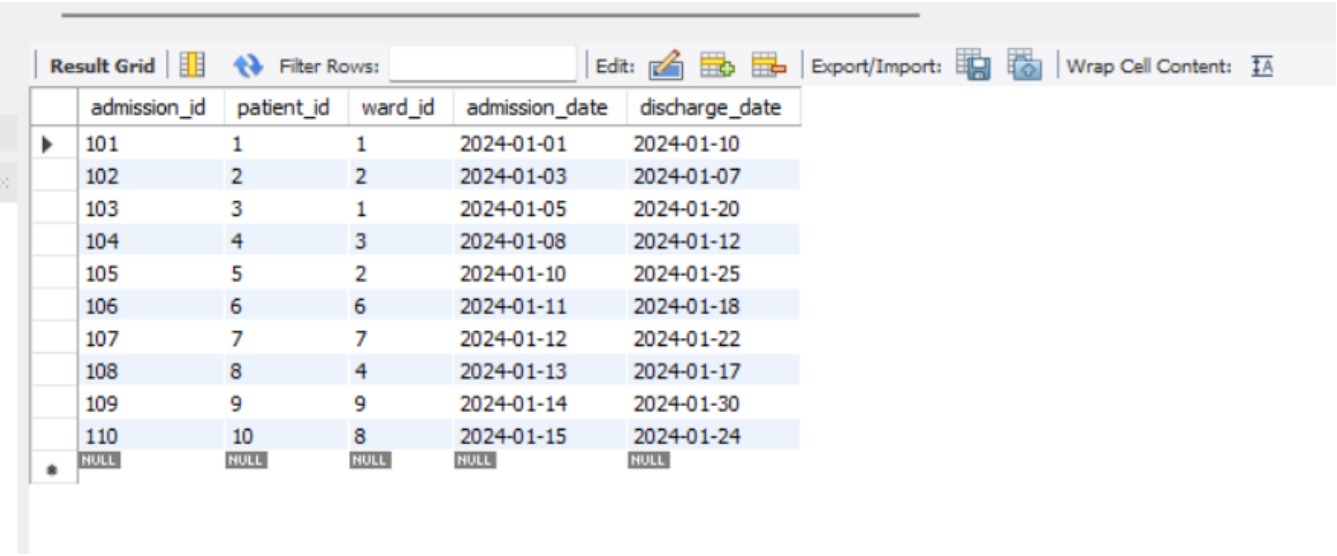
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

ward_id	ward_name
1	ICU
2	General
3	Surgical
4	Maternity
5	Pediatric
6	Orthopedic
7	Cardiology
8	Neurology
9	Oncology
10	Emergency
NULL	NULL

```
CREATE TABLE admissions (
    admission_id INT AUTO_INCREMENT PRIMARY KEY,
    patient_id INT,
    ward_id INT,
    admission_date DATE,
    discharge_date DATE,
    CONSTRAINT fk_patient
        FOREIGN KEY (patient_id) REFERENCES patients(patient_id),
    CONSTRAINT fk_ward
        FOREIGN KEY (ward_id) REFERENCES wards(ward_id)
);
INSERT INTO admissions (admission_id, patient_id, ward_id, admission_date,
discharge_date) VALUES
(101, 1, 1, '2024-01-01', '2024-01-10'),
(102, 2, 2, '2024-01-03', '2024-01-07'),
(103, 3, 1, '2024-01-05', '2024-01-20'),
(104, 4, 3, '2024-01-08', '2024-01-12'),
(105, 5, 2, '2024-01-10', '2024-01-25'),
(106, 6, 6, '2024-01-11', '2024-01-18'),
(107, 7, 7, '2024-01-12', '2024-01-22'),
(108, 8, 4, '2024-01-13', '2024-01-17'),
(109, 9, 9, '2024-01-14', '2024-01-30'),
```

(110, 10, 8, '2024-01-15', '2024-01-24');

OUTPUT:



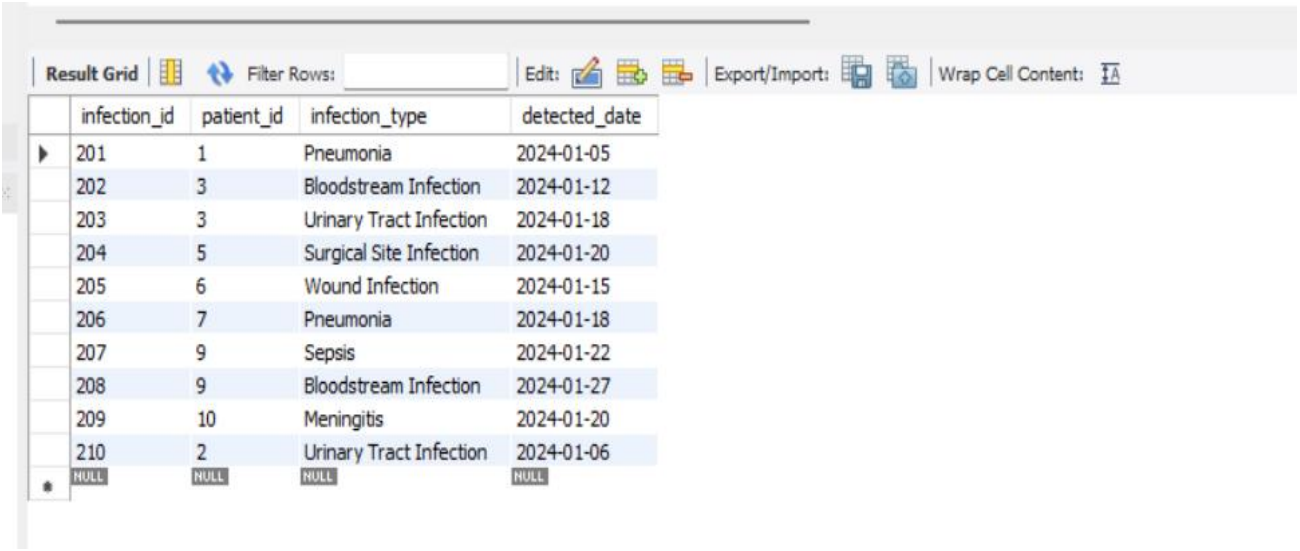
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

admission_id	patient_id	ward_id	admission_date	discharge_date
101	1	1	2024-01-01	2024-01-10
102	2	2	2024-01-03	2024-01-07
103	3	1	2024-01-05	2024-01-20
104	4	3	2024-01-08	2024-01-12
105	5	2	2024-01-10	2024-01-25
106	6	6	2024-01-11	2024-01-18
107	7	7	2024-01-12	2024-01-22
108	8	4	2024-01-13	2024-01-17
109	9	9	2024-01-14	2024-01-30
110	10	8	2024-01-15	2024-01-24
NULL	NULL	NULL	NULL	NULL

```
CREATE TABLE infections (
    infection_id INT AUTO_INCREMENT PRIMARY KEY,
    patient_id INT,
    infection_type VARCHAR(100),
    detected_date DATE,
    CONSTRAINT fk_patient_infection
        FOREIGN KEY (patient_id) REFERENCES patients(patient_id)
);
INSERT INTO infections (infection_id, patient_id, infection_type, detected_date) VALUES
(201, 1, 'Pneumonia', '2024-01-05'),
(202, 3, 'Bloodstream Infection', '2024-01-12'),
(203, 3, 'Urinary Tract Infection', '2024-01-18'),
(204, 5, 'Surgical Site Infection', '2024-01-20'),
(205, 6, 'Wound Infection', '2024-01-15'),
(206, 7, 'Pneumonia', '2024-01-18'),
(207, 9, 'Sepsis', '2024-01-22'),
(208, 9, 'Bloodstream Infection', '2024-01-27'),
(209, 10, 'Meningitis', '2024-01-20'),
```

(210, 2, 'Urinary Tract Infection', '2024-01-06');

OUTPUT:



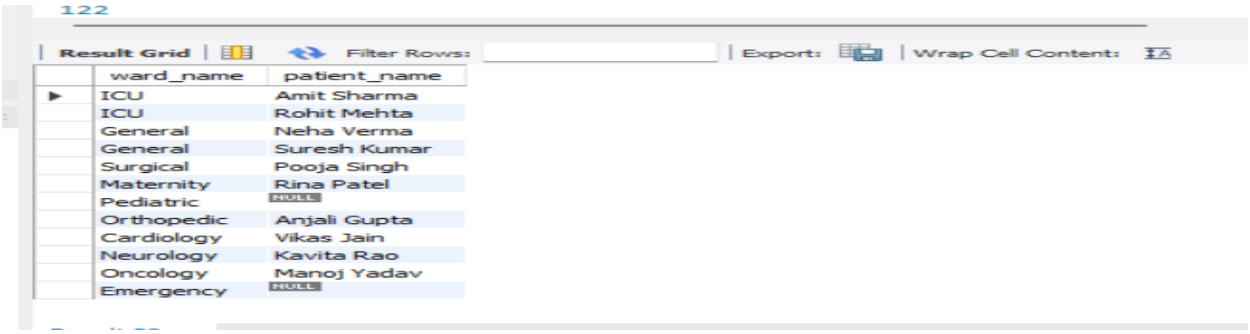
Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

infection_id	patient_id	infection_type	detected_date
201	1	Pneumonia	2024-01-05
202	3	Bloodstream Infection	2024-01-12
203	3	Urinary Tract Infection	2024-01-18
204	5	Surgical Site Infection	2024-01-20
205	6	Wound Infection	2024-01-15
206	7	Pneumonia	2024-01-18
207	9	Sepsis	2024-01-22
208	9	Bloodstream Infection	2024-01-27
209	10	Meningitis	2024-01-20
210	2	Urinary Tract Infection	2024-01-06
NULL	NULL	NULL	NULL

JOINS

Q1. Display patient name, ward name, and admission date for patients who were admitted

```
SELECT
p.patient_name,
w.ward_name,
a.admission_date
FROM patients p
INNER JOIN admissions a ON p.patient_id = a.patient_id
INNER JOIN wards w ON a.ward_id = w.ward_id;
```



122

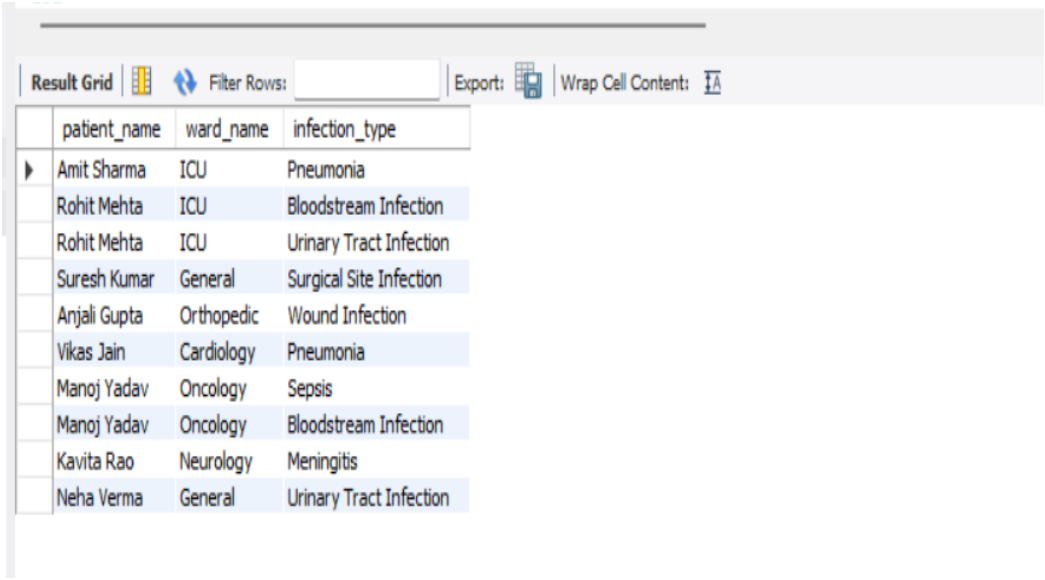
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

ward_name	patient_name
ICU	Amit Sharma
ICU	Rohit Mehta
General	Neha Verma
General	Suresh Kumar
Surgical	Pooja Singh
Maternity	Rina Patel
Pediatric	NULL
Orthopedic	Anjali Gupta
Cardiology	Vikas Jain
Neurology	Kavita Rao
Oncology	Manoj Yadav
Emergency	NULL

Q2.

Display patient name, ward name, and infection type for patients who developed an infection after being admitted.

```
SELECT
   p.patient_name,
   w.ward_name,
   i.infection_type
FROM patients p
INNER JOIN admissions a ON p.patient_id = a.patient_id
INNER JOIN wards w ON a.ward_id = w.ward_id
INNER JOIN infections i ON p.patient_id = i.patient_id;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

patient_name	ward_name	infection_type
Amit Sharma	ICU	Pneumonia
Rohit Mehta	ICU	Bloodstream Infection
Rohit Mehta	ICU	Urinary Tract Infection
Suresh Kumar	General	Surgical Site Infection
Anjali Gupta	Orthopedic	Wound Infection
Vikas Jain	Cardiology	Pneumonia
Manoj Yadav	Oncology	Sepsis
Manoj Yadav	Oncology	Bloodstream Infection
Kavita Rao	Neurology	Meningitis
Neha Verma	General	Urinary Tract Infection

- **LEFT JOIN (2 Questions)**

Q3.

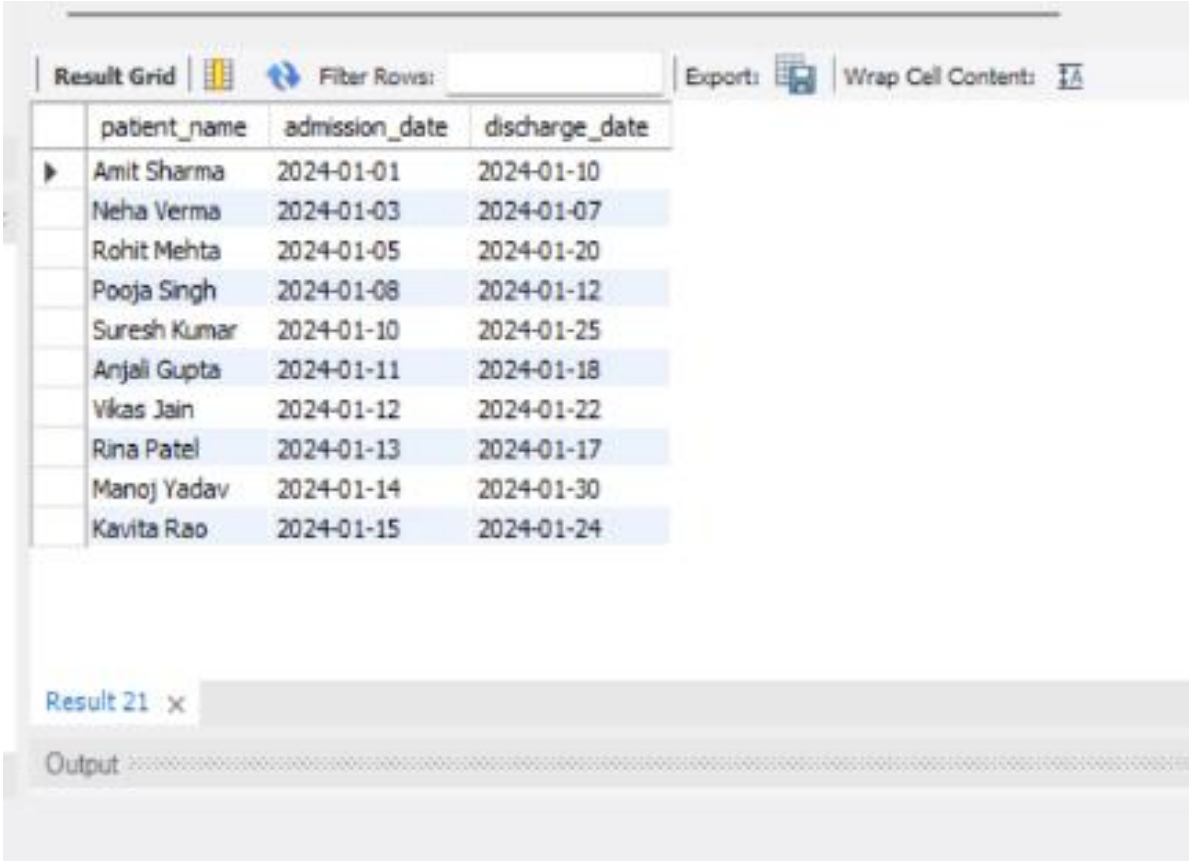
List all patients with their admission details, including patients not admitted.

✅ **Answer:**

```
SELECT
   p.patient_name,
   a.admission_date,
   a.discharge_date
```

FROM patients p

LEFT JOIN admissions a ON p.patient_id = a.patient_id;



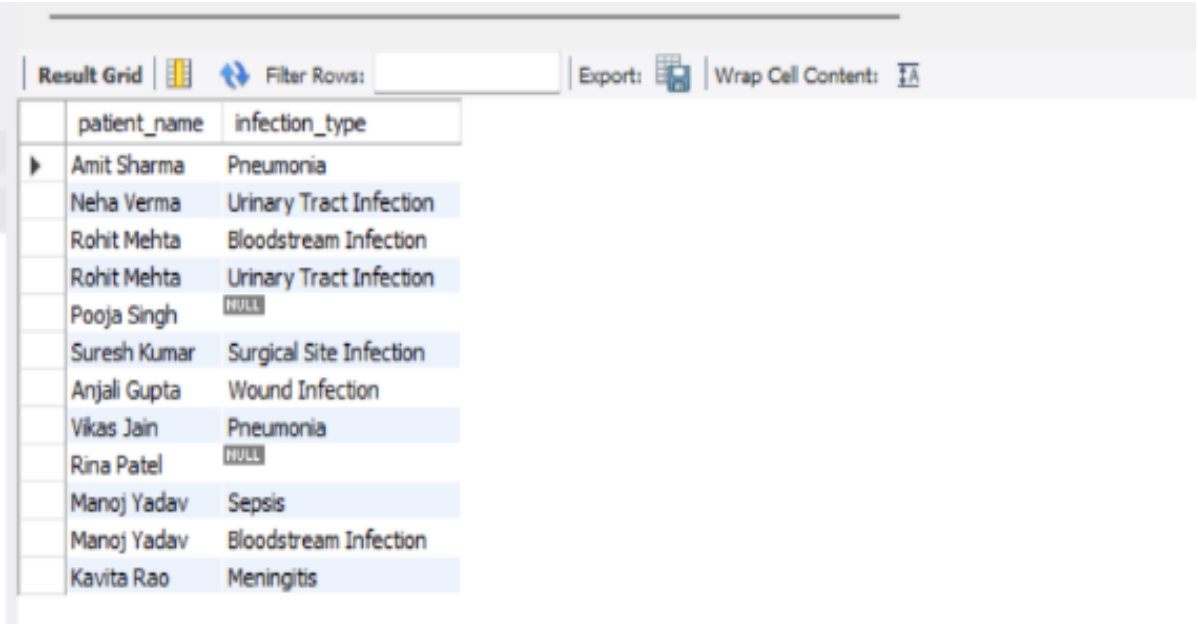
patient_name	admission_date	discharge_date
Amit Sharma	2024-01-01	2024-01-10
Neha Verma	2024-01-03	2024-01-07
Rohit Mehta	2024-01-05	2024-01-20
Pooja Singh	2024-01-08	2024-01-12
Suresh Kumar	2024-01-10	2024-01-25
Anjali Gupta	2024-01-11	2024-01-18
Vikas Jain	2024-01-12	2024-01-22
Rina Patel	2024-01-13	2024-01-17
Manoj Yadav	2024-01-14	2024-01-30
Kavita Rao	2024-01-15	2024-01-24

Q4.

Display all patients and their infection details, even if they have no infection.

✅ **Answer:**

```
SELECT
   p.patient_name,
   i.infection_type
FROM patients p
LEFT JOIN infections i ON p.patient_id = i.patient_id;
```



patient_name	infection_type
Amit Sharma	Pneumonia
Neha Verma	Urinary Tract Infection
Rohit Mehta	Bloodstream Infection
Rohit Mehta	Urinary Tract Infection
Pooja Singh	NULL
Suresh Kumar	Surgical Site Infection
Anjali Gupta	Wound Infection
Vikas Jain	Pneumonia
Rina Patel	NULL
Manoj Yadav	Sepsis
Manoj Yadav	Bloodstream Infection
Kavita Rao	Meningitis

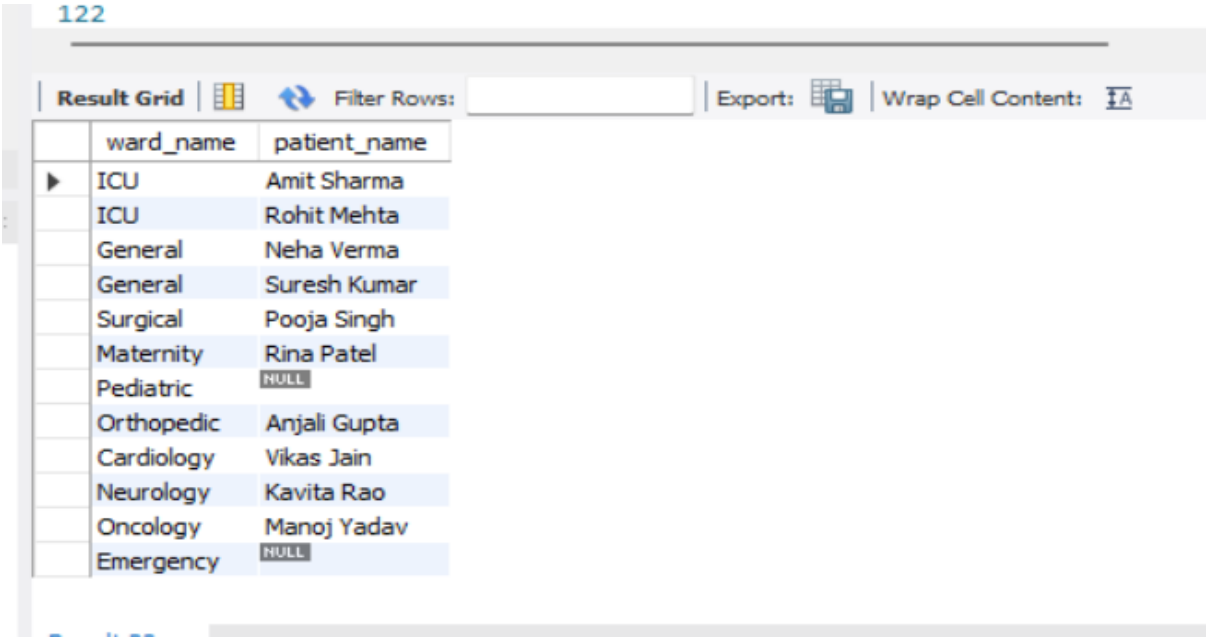
🔹 **RIGHT JOIN (2 Questions)**

Q5.

Show all wards and their admitted patients, including wards with no admissions.

✅ **Answer:**

```
SELECT
   w.ward_name,
   p.patient_name
FROM patients p
RIGHT JOIN admissions a ON p.patient_id = a.patient_id
RIGHT JOIN wards w ON a.ward_id = w.ward_id;
```

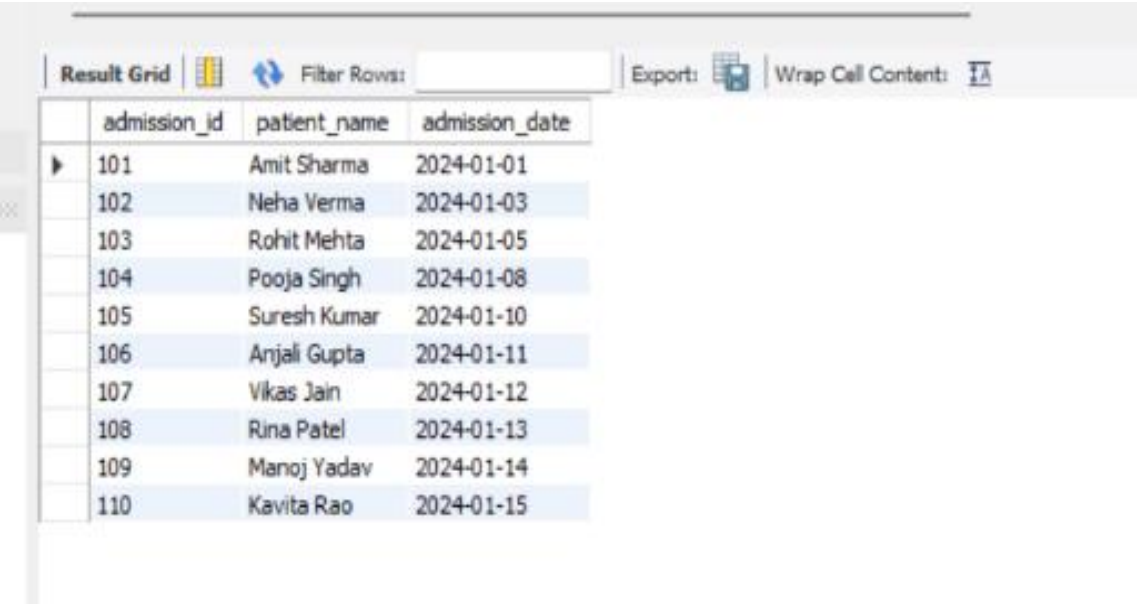


Q6.

Display all admissions with patient names, including admissions without patient details.

✅ **Answer:**

```
SELECT
    a.admission_id,
    p.patient_name,
    a.admission_date
FROM patients p
RIGHT JOIN admissions a ON p.patient_id = a.patient_id;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

admission_id	patient_name	admission_date
101	Amit Sharma	2024-01-01
102	Neha Verma	2024-01-03
103	Rohit Mehta	2024-01-05
104	Pooja Singh	2024-01-08
105	Suresh Kumar	2024-01-10
106	Anjali Gupta	2024-01-11
107	Vikas Jain	2024-01-12
108	Rina Patel	2024-01-13
109	Manoj Yadav	2024-01-14
110	Kavita Rao	2024-01-15

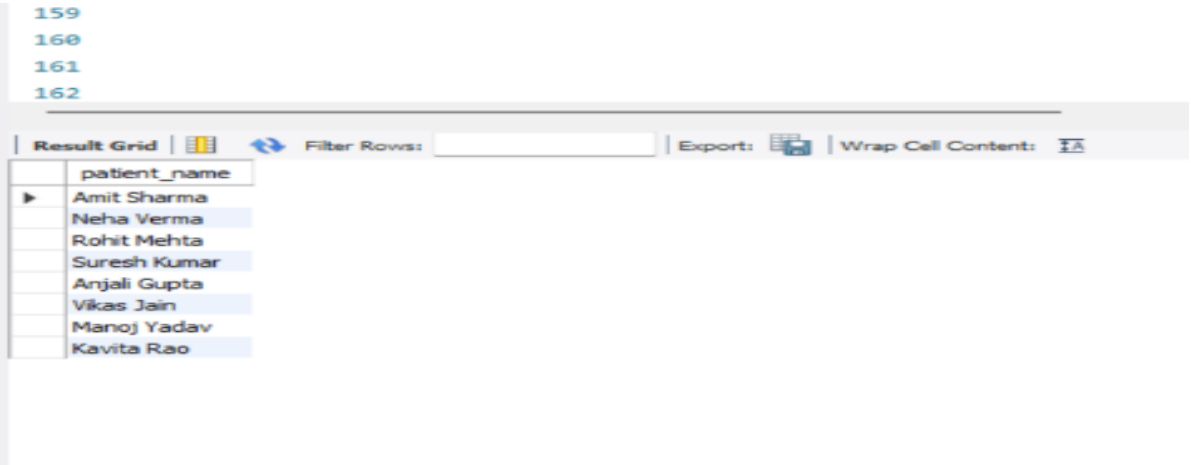
SUBQUERY

Q1.

Find the **names of patients who have at least one infection**.

✅ **Answer:**

```
SELECT patient_name
FROM patients
WHERE patient_id IN (
   SELECT patient_id
   FROM infections
);
```



159
160
161
162

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

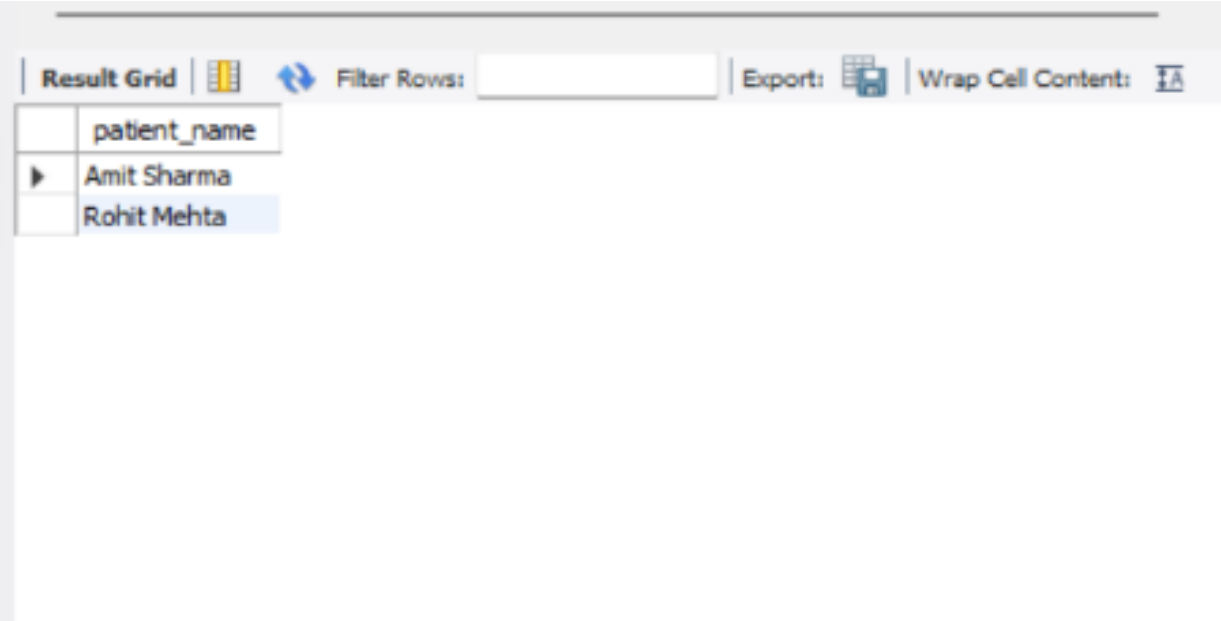
patient_name
Amit Sharma
Neha Verma
Rohit Mehta
Suresh Kumar
Anjali Gupta
Vikas Jain
Manoj Yadav
Kavita Rao

- **Q2.**

Display **patients admitted to the ICU**.

✅ **Answer:**

```
SELECT patient_name
FROM patients
WHERE patient_id IN (
   SELECT patient_id
   FROM admissions
   WHERE ward_id = (
      SELECT ward_id
      FROM wards
      WHERE ward_name = 'ICU'
   )
);
```



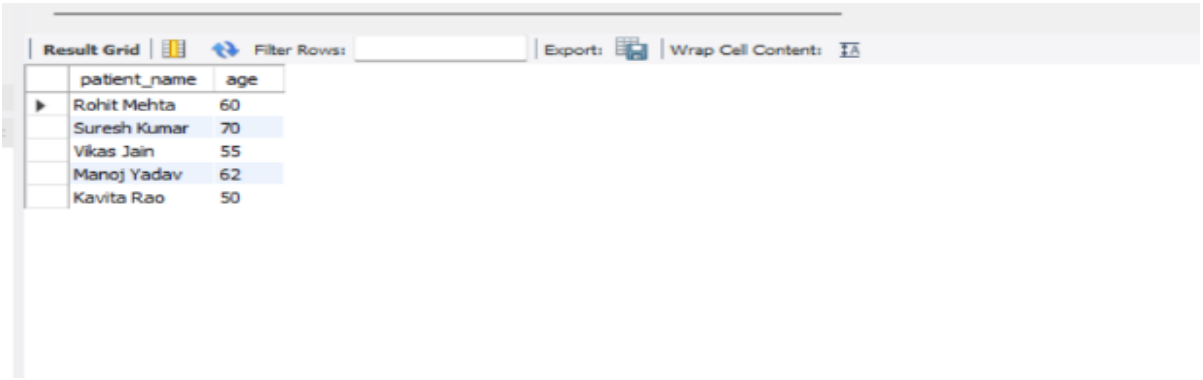
🔹 **Q3.**

List **patients older than the average age**.

✅ **Answer:**

```
SELECT patient_name, age
FROM patients
WHERE age > (
```

```
    SELECT AVG(age)
    FROM patients
);
```



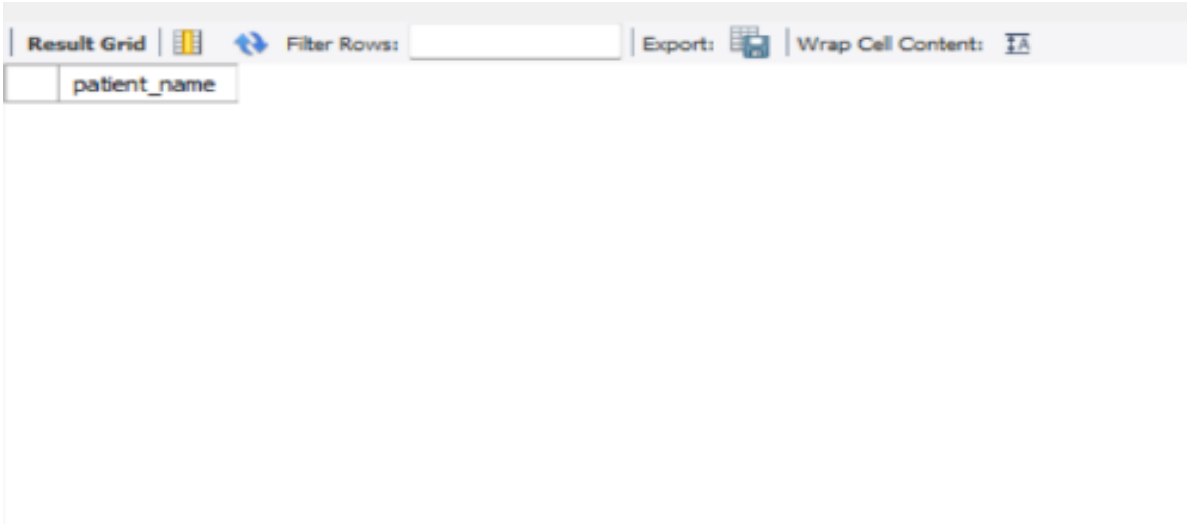
patient_name	age
Rohit Mehta	60
Suresh Kumar	70
Vikas Jain	55
Manoj Yadav	62
Kavita Rao	50

🔹 **Q4.**

Find **patients who were admitted more than once**.

✅ **Answer:**

```
SELECT patient_name
FROM patients
WHERE patient_id IN (
    SELECT patient_id
    FROM admissions
    GROUP BY patient_id
    HAVING COUNT(*) > 1
);
```



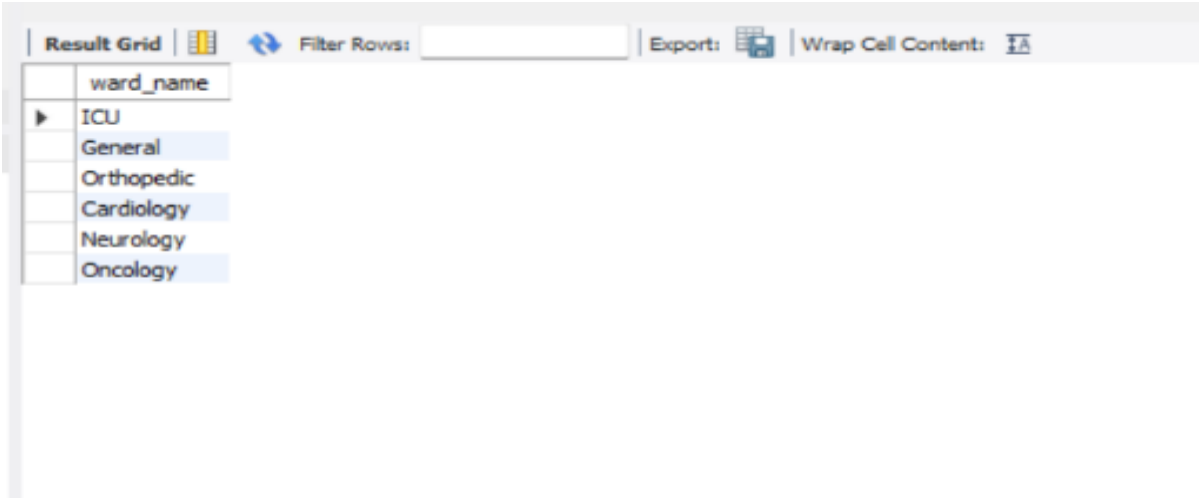
patient_name

🔹 **Q5.**

Display **wards where at least one patient had an infection**.

✅ **Answer:**

```
SELECT ward_name
FROM wards
WHERE ward_id IN (
   SELECT ward_id
   FROM admissions
   WHERE patient_id IN (
      SELECT patient_id
      FROM infections
   )
);
```



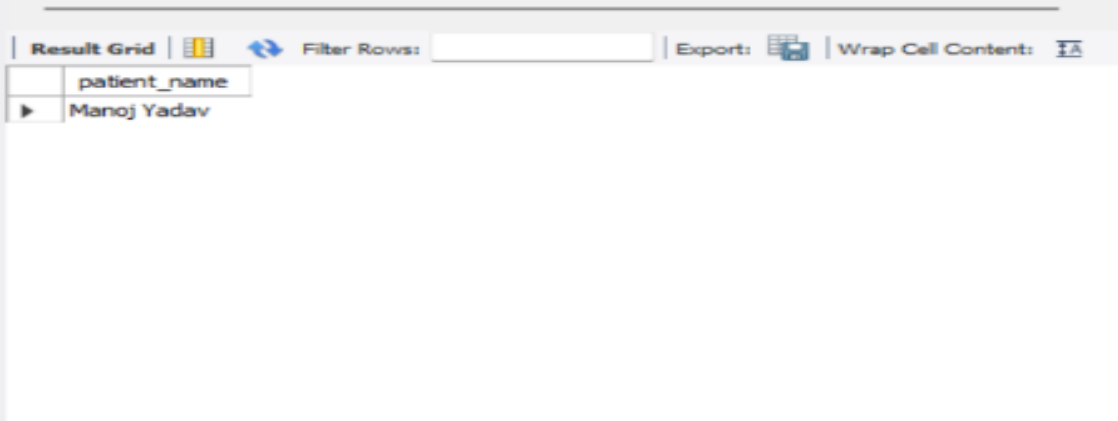
🔹 **Q6.**

Find the **patient(s) with the longest hospital stay**.

✅ **Answer:**

```
SELECT patient_name
FROM patients
WHERE patient_id IN (
   SELECT patient_id
   FROM admissions
```

```
    WHERE DATEDIFF(discharge_date, admission_date) = (
        SELECT MAX(DATEDIFF(discharge_date, admission_date))
        FROM admissions
    )
);
```



CONCLUSION

This project successfully demonstrates how SQL can be used to analyze and manage hospital infection control data in an efficient and structured manner. By designing a well-normalized database and applying SQL concepts such as joins, subqueries, and aggregations, the project provides meaningful insights into patient admissions, ward utilization, and the occurrence of hospital-acquired infections. The analysis helps in identifying high-risk wards, understanding patient infection patterns, and monitoring hospital stay durations, which are critical factors in improving patient safety.

Overall, the project highlights the practical importance of SQL in the healthcare domain by transforming raw hospital data into actionable information. It showcases the ability to handle real-world data scenarios, supports better decision-making for infection prevention, and strengthens database management and analytical skills. This project can be further extended with advanced analytics, dashboards, or real-time monitoring to support proactive healthcare management.