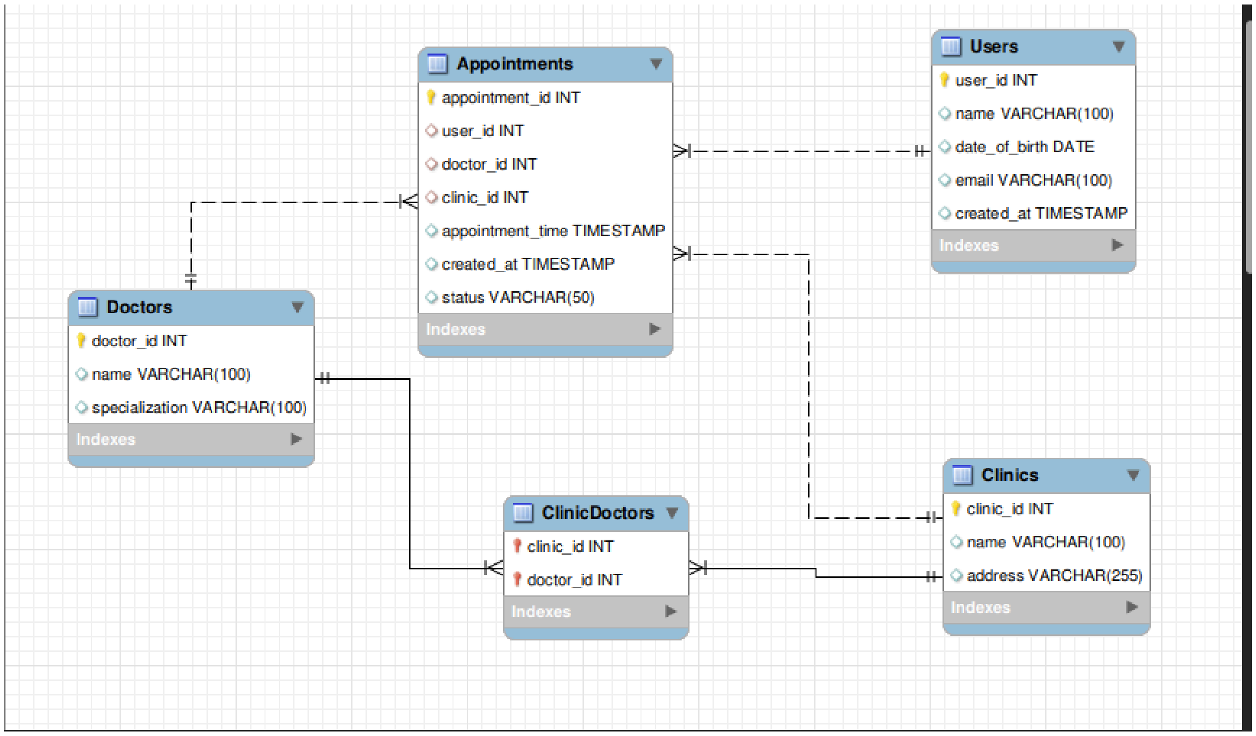
Aditya Gupta
I0338
aditya.gupta@practo.com

SQL assignment

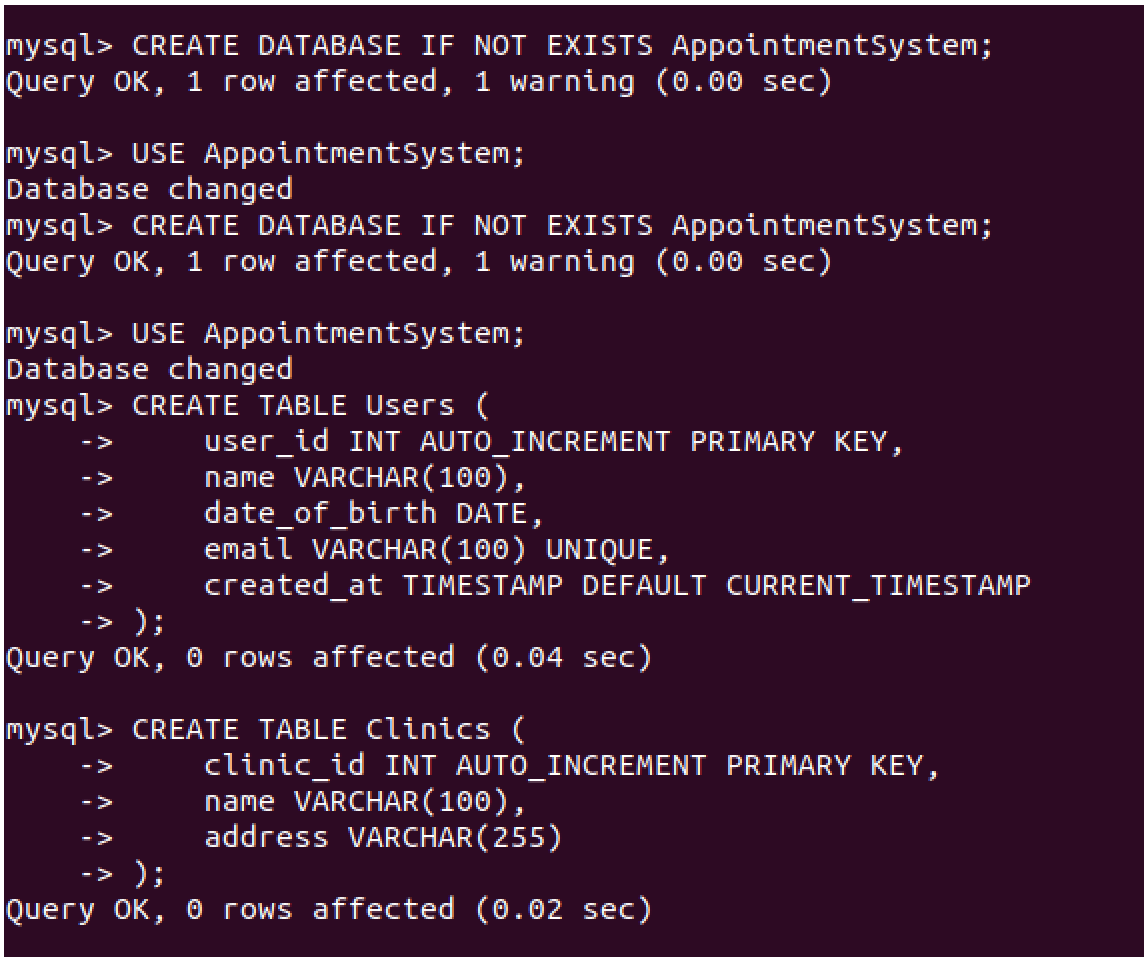
ER diagram



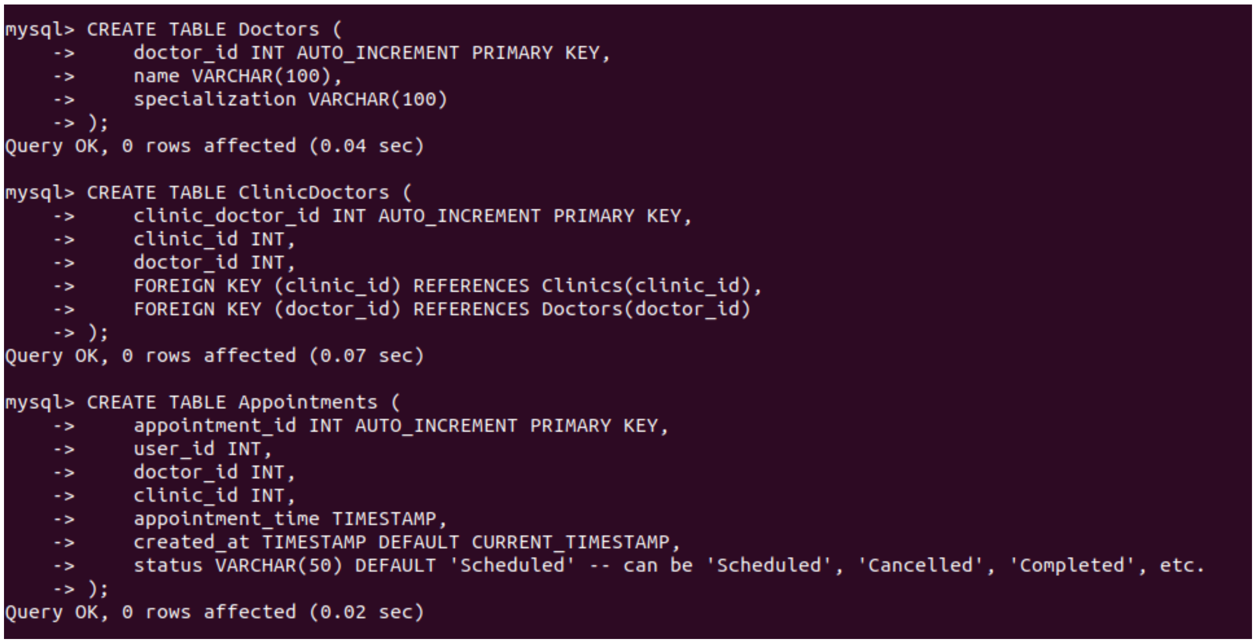
Below are screenshots attached
Starting from

a. creating tables
b. Inserting data in tables
c. Querying on db
d. Creating indexes for optimisation
e. Using EXPLAIN on each query

Creating database and tables

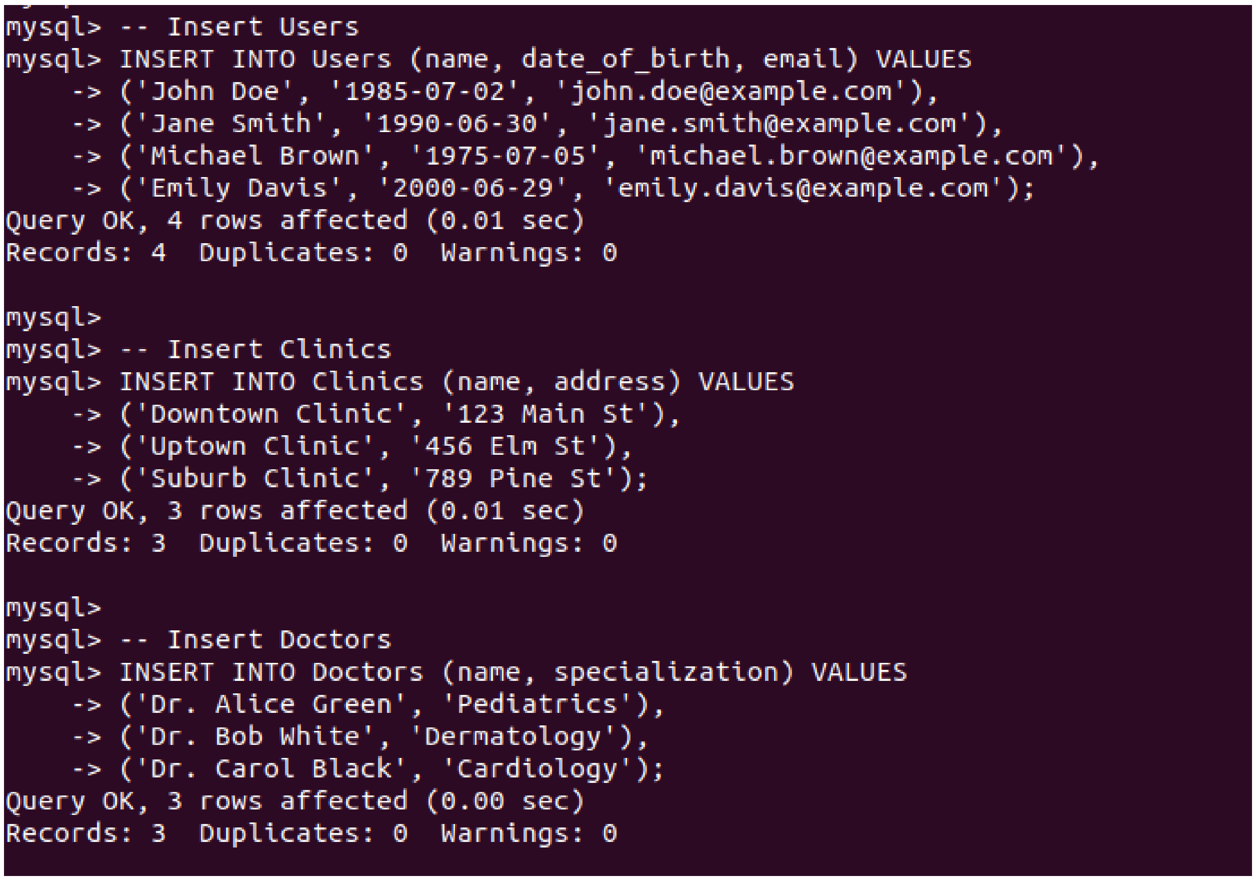


```
mysql> CREATE DATABASE IF NOT EXISTS AppointmentSystem;
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> USE AppointmentSystem;
Database changed
mysql> CREATE DATABASE IF NOT EXISTS AppointmentSystem;
Query OK, 1 row affected, 1 warning (0.00 sec)

mysql> USE AppointmentSystem;
Database changed
mysql> CREATE TABLE Users (
    ->     user_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     date_of_birth DATE,
    ->     email VARCHAR(100) UNIQUE,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE Clinics (
    ->     clinic_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     address VARCHAR(255)
    -> );
Query OK, 0 rows affected (0.02 sec)
```

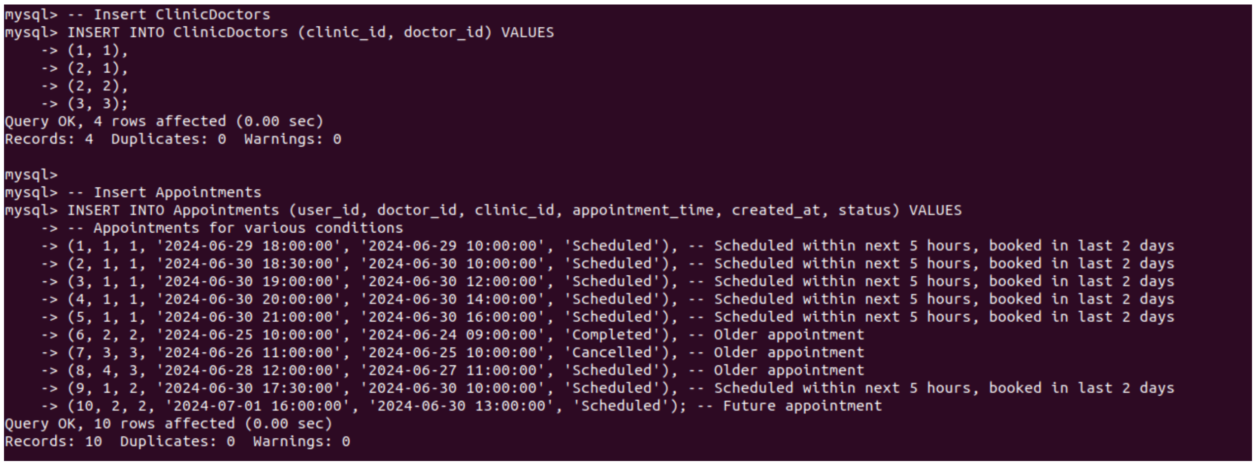


```
mysql> CREATE TABLE Doctors (
    ->     doctor_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     name VARCHAR(100),
    ->     specialization VARCHAR(100)
    -> );
Query OK, 0 rows affected (0.04 sec)

mysql> CREATE TABLE ClinicDoctors (
    ->     clinic_doctor_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     clinic_id INT,
    ->     doctor_id INT,
    ->     FOREIGN KEY (clinic_id) REFERENCES Clinics(clinic_id),
    ->     FOREIGN KEY (doctor_id) REFERENCES Doctors(doctor_id)
    -> );
Query OK, 0 rows affected (0.07 sec)

mysql> CREATE TABLE Appointments (
    ->     appointment_id INT AUTO_INCREMENT PRIMARY KEY,
    ->     user_id INT,
    ->     doctor_id INT,
    ->     clinic_id INT,
    ->     appointment_time TIMESTAMP,
    ->     created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    ->     status VARCHAR(50) DEFAULT 'Scheduled' -- can be 'Scheduled', 'Cancelled', 'Completed', etc.
    -> );
Query OK, 0 rows affected (0.02 sec)
```

Now inserting data in tables



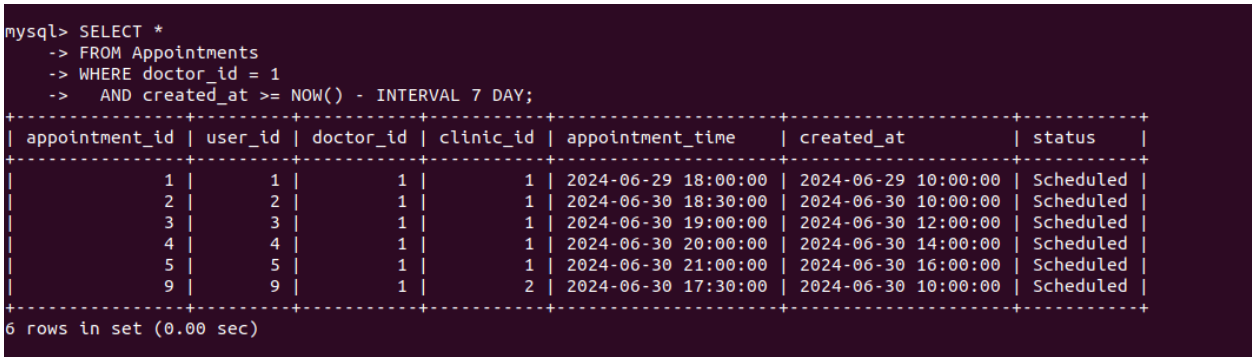
```
mysql> -- Insert Users
mysql> INSERT INTO Users (name, date_of_birth, email) VALUES
    -> ('John Doe', '1985-07-02', 'john.doe@example.com'),
    -> ('Jane Smith', '1990-06-30', 'jane.smith@example.com'),
    -> ('Michael Brown', '1975-07-05', 'michael.brown@example.com'),
    -> ('Emily Davis', '2000-06-29', 'emily.davis@example.com');
Query OK, 4 rows affected (0.01 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert Clinics
mysql> INSERT INTO Clinics (name, address) VALUES
    -> ('Downtown Clinic', '123 Main St'),
    -> ('Uptown Clinic', '456 Elm St'),
    -> ('Suburb Clinic', '789 Pine St');
Query OK, 3 rows affected (0.01 sec)
Records: 3  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert Doctors
mysql> INSERT INTO Doctors (name, specialization) VALUES
    -> ('Dr. Alice Green', 'Pediatrics'),
    -> ('Dr. Bob White', 'Dermatology'),
    -> ('Dr. Carol Black', 'Cardiology');
Query OK, 3 rows affected (0.00 sec)
Records: 3  Duplicates: 0  Warnings: 0
```



```
mysql> -- Insert ClinicDoctors
mysql> INSERT INTO ClinicDoctors (clinic_id, doctor_id) VALUES
    -> (1, 1),
    -> (2, 1),
    -> (2, 2),
    -> (3, 3);
Query OK, 4 rows affected (0.00 sec)
Records: 4  Duplicates: 0  Warnings: 0

mysql>
mysql> -- Insert Appointments
mysql> INSERT INTO Appointments (user_id, doctor_id, clinic_id, appointment_time, created_at, status) VALUES
    -> -- Appointments for various conditions
    -> (1, 1, 1, '2024-06-29 18:00:00', '2024-06-29 10:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (2, 1, 1, '2024-06-30 18:30:00', '2024-06-30 10:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (3, 1, 1, '2024-06-30 19:00:00', '2024-06-30 12:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (4, 1, 1, '2024-06-30 20:00:00', '2024-06-30 14:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (5, 1, 1, '2024-06-30 21:00:00', '2024-06-30 16:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (6, 2, 2, '2024-06-25 10:00:00', '2024-06-24 09:00:00', 'Completed'), -- Older appointment
    -> (7, 3, 3, '2024-06-26 11:00:00', '2024-06-25 10:00:00', 'Cancelled'), -- Older appointment
    -> (8, 4, 3, '2024-06-28 12:00:00', '2024-06-27 11:00:00', 'Scheduled'), -- Older appointment
    -> (9, 1, 2, '2024-06-30 17:30:00', '2024-06-30 10:00:00', 'Scheduled'), -- Scheduled within next 5 hours, booked in last 2 days
    -> (10, 2, 2, '2024-07-01 16:00:00', '2024-06-30 13:00:00', 'Scheduled'); -- Future appointment
Query OK, 10 rows affected (0.00 sec)
Records: 10  Duplicates: 0  Warnings: 0
```

Screenshots of queries with outputs

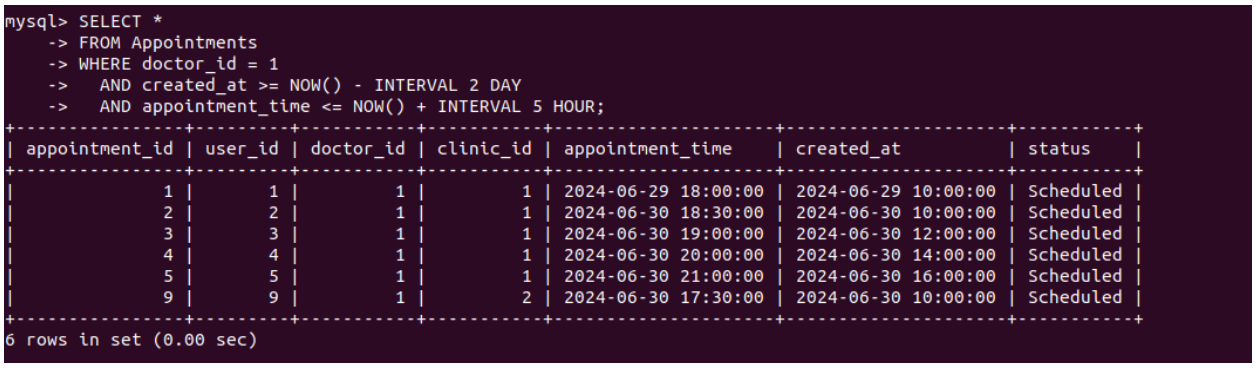
1. All appointments booked in last 7 days for a doctor



```
mysql> SELECT *
    -> FROM Appointments
    -> WHERE doctor_id = 1
    ->   AND created_at >= NOW() - INTERVAL 7 DAY;
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
| appointment_id | user_id | doctor_id | clinic_id | appointment_time    | created_at          | status    |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
|              1 |       1 |         1 |         1 | 2024-06-29 18:00:00 | 2024-06-29 10:00:00 | Scheduled |
|              2 |       2 |         1 |         1 | 2024-06-30 18:30:00 | 2024-06-30 10:00:00 | Scheduled |
|              3 |       3 |         1 |         1 | 2024-06-30 19:00:00 | 2024-06-30 12:00:00 | Scheduled |
|              4 |       4 |         1 |         1 | 2024-06-30 20:00:00 | 2024-06-30 14:00:00 | Scheduled |
|              5 |       5 |         1 |         1 | 2024-06-30 21:00:00 | 2024-06-30 16:00:00 | Scheduled |
|              9 |       9 |         1 |         2 | 2024-06-30 17:30:00 | 2024-06-30 10:00:00 | Scheduled |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
6 rows in set (0.00 sec)
```

Explanation: This query retrieves all appointments for doctor with `doctor_id = 1` that were created in the last 7 days from the current date and time.

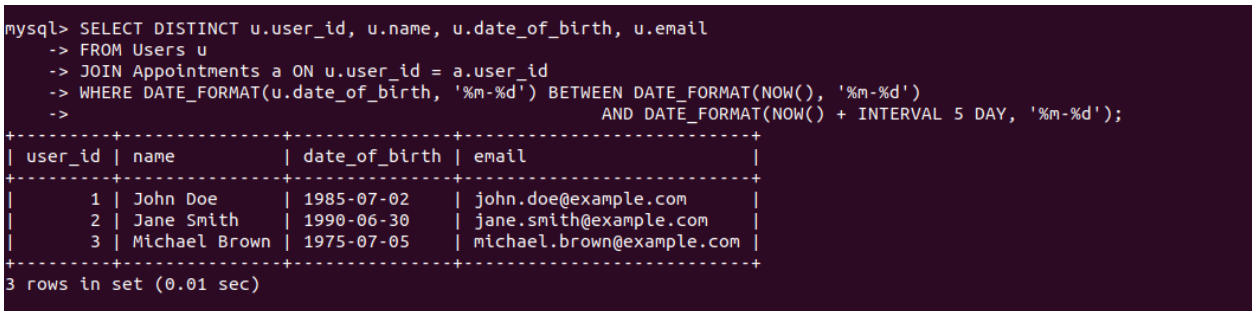
2. All appointments booked in last 2 days n scheduled within next 5 hours for a doctor



```
mysql> SELECT *
    -> FROM Appointments
    -> WHERE doctor_id = 1
    ->   AND created_at >= NOW() - INTERVAL 2 DAY
    ->   AND appointment_time <= NOW() + INTERVAL 5 HOUR;
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
| appointment_id | user_id | doctor_id | clinic_id | appointment_time    | created_at          | status    |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
|              1 |       1 |         1 |         1 | 2024-06-29 18:00:00 | 2024-06-29 10:00:00 | Scheduled |
|              2 |       2 |         1 |         1 | 2024-06-30 18:30:00 | 2024-06-30 10:00:00 | Scheduled |
|              3 |       3 |         1 |         1 | 2024-06-30 19:00:00 | 2024-06-30 12:00:00 | Scheduled |
|              4 |       4 |         1 |         1 | 2024-06-30 20:00:00 | 2024-06-30 14:00:00 | Scheduled |
|              5 |       5 |         1 |         1 | 2024-06-30 21:00:00 | 2024-06-30 16:00:00 | Scheduled |
|              9 |       9 |         1 |         2 | 2024-06-30 17:30:00 | 2024-06-30 10:00:00 | Scheduled |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
6 rows in set (0.00 sec)
```

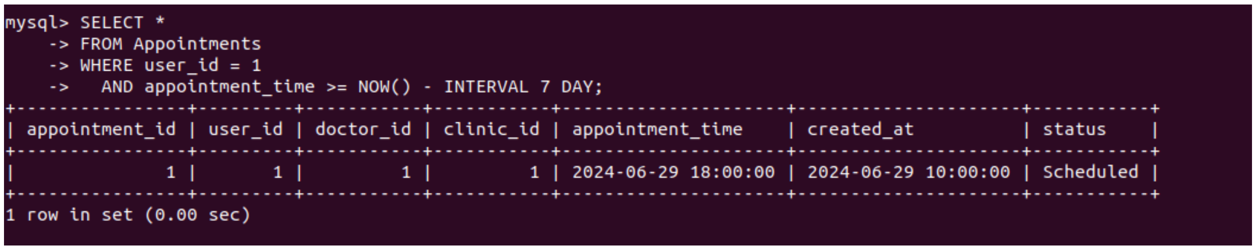
Explanation: This query retrieves all appointments for doctor with `doctor_id = 1` that were created in the last 2 days and are scheduled to occur within the next 5 hours from the current date and time.

3. User who have at least 1 appointment and have their birthday coming in next 5 days


```
mysql> SELECT DISTINCT u.user_id, u.name, u.date_of_birth, u.email
    -> FROM Users u
    -> JOIN Appointments a ON u.user_id = a.user_id
    -> WHERE DATE_FORMAT(u.date_of_birth, '%m-%d') BETWEEN DATE_FORMAT(NOW(), '%m-%d')
    ->                                                  AND DATE_FORMAT(NOW() + INTERVAL 5 DAY, '%m-%d');
+---------+---------------+---------------+---------------------------+
| user_id | name          | date_of_birth | email                     |
+---------+---------------+---------------+---------------------------+
|       1 | John Doe      | 1985-07-02    | john.doe@example.com      |
|       2 | Jane Smith    | 1990-06-30    | jane.smith@example.com    |
|       3 | Michael Brown | 1975-07-05    | michael.brown@example.com |
+---------+---------------+---------------+---------------------------+
3 rows in set (0.01 sec)
```

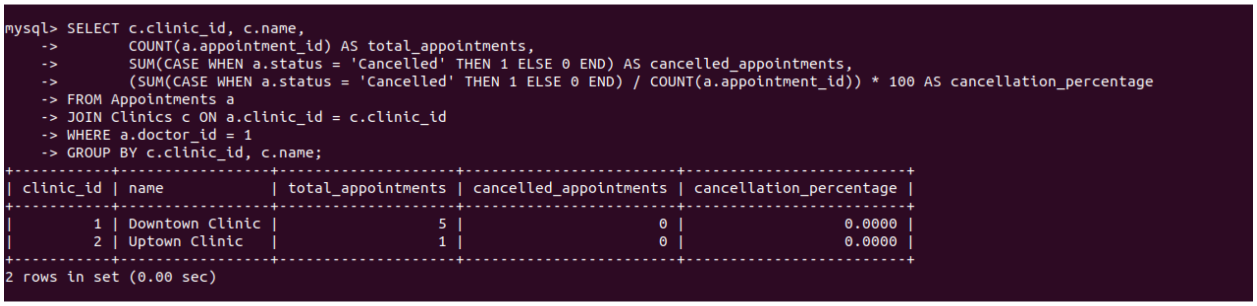
Explanation: This query retrieves users who have at least one appointment and whose birthday is coming in the next 5 days. The `DATE_FORMAT` function is used to ignore the year part and only consider the month and day.

4. Appointments for a particular patient in the last 7 days


```
mysql> SELECT *
    -> FROM Appointments
    -> WHERE user_id = 1
    ->   AND appointment_time >= NOW() - INTERVAL 7 DAY;
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
| appointment_id | user_id | doctor_id | clinic_id | appointment_time    | created_at          | status    |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
|              1 |       1 |         1 |         1 | 2024-06-29 18:00:00 | 2024-06-29 10:00:00 | Scheduled |
+----------------+---------+-----------+-----------+---------------------+---------------------+-----------+
1 row in set (0.00 sec)
```

Explanation: This query retrieves all appointments for the user with `user_id = 1` that are scheduled in the last 7 days from the current date and time. The `appointment_time` is used to filter the records.

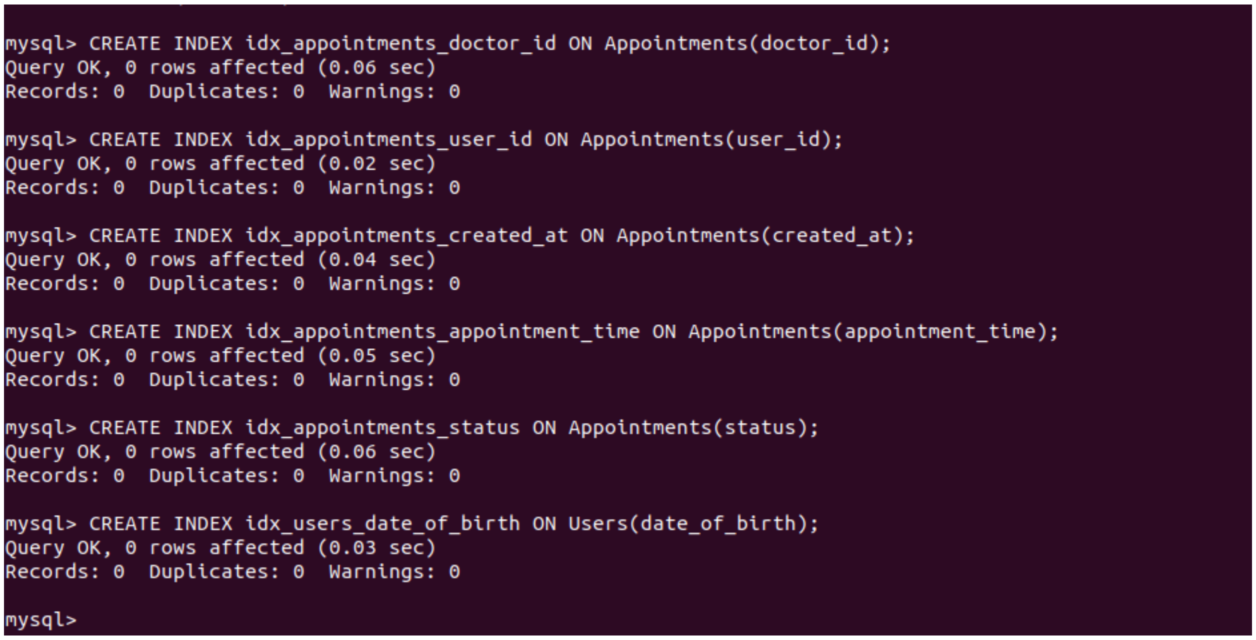
5. Appointment cancellation percentage for a doctor by clinic



```
mysql> SELECT c.clinic_id, c.name,
    ->        COUNT(a.appointment_id) AS total_appointments,
    ->        SUM(CASE WHEN a.status = 'Cancelled' THEN 1 ELSE 0 END) AS cancelled_appointments,
    ->        (SUM(CASE WHEN a.status = 'Cancelled' THEN 1 ELSE 0 END) / COUNT(a.appointment_id)) * 100 AS cancellation_percentage
    -> FROM Appointments a
    -> JOIN Clinics c ON a.clinic_id = c.clinic_id
    -> WHERE a.doctor_id = 1
    -> GROUP BY c.clinic_id, c.name;
+-----------+-----------------+--------------------+------------------------+-------------------------+
| clinic_id | name            | total_appointments | cancelled_appointments | cancellation_percentage |
+-----------+-----------------+--------------------+------------------------+-------------------------+
|         1 | Downtown Clinic |                  5 |                      0 |                  0.0000 |
|         2 | Uptown Clinic   |                  1 |                      0 |                  0.0000 |
+-----------+-----------------+--------------------+------------------------+-------------------------+
2 rows in set (0.00 sec)
```

Explanation: This query calculates the appointment cancellation percentage for doctor with `doctor_id = 1` grouped by clinic. It counts the total number of appointments and the number of canceled appointments, then calculates the cancellation percentage.

Now we can Create Indexes for Optimization

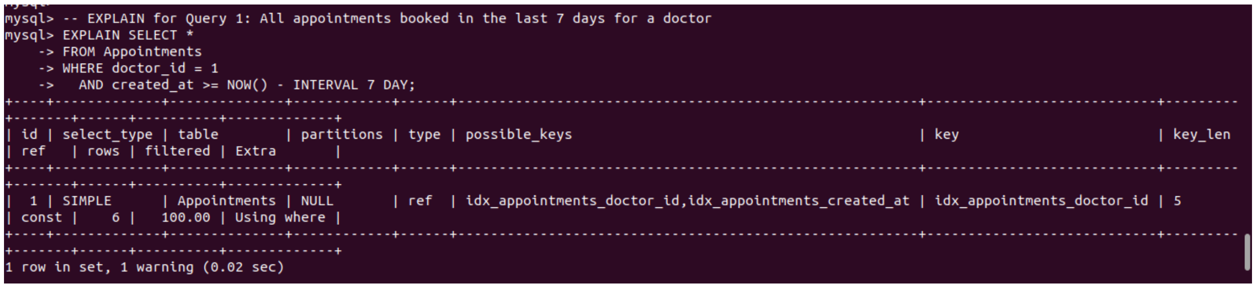


```
mysql> CREATE INDEX idx_appointments_doctor_id ON Appointments(doctor_id);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_appointments_user_id ON Appointments(user_id);
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_appointments_created_at ON Appointments(created_at);
Query OK, 0 rows affected (0.04 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_appointments_appointment_time ON Appointments(appointment_time);
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_appointments_status ON Appointments(status);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> CREATE INDEX idx_users_date_of_birth ON Users(date_of_birth);
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql>
```

Explanation: Indexes are crucial for optimizing query performance by reducing the amount of data that the database engine needs to scan. By creating indexes on columns that are frequently used in WHERE, JOIN, and ORDER BY clauses, the database can quickly locate the rows that satisfy the conditions, resulting in faster query execution times.

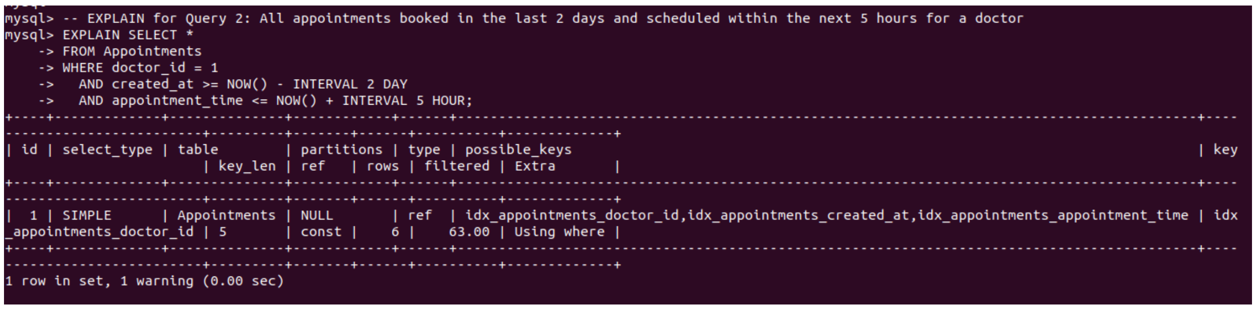
Next using EXPLAIN on each query

1.



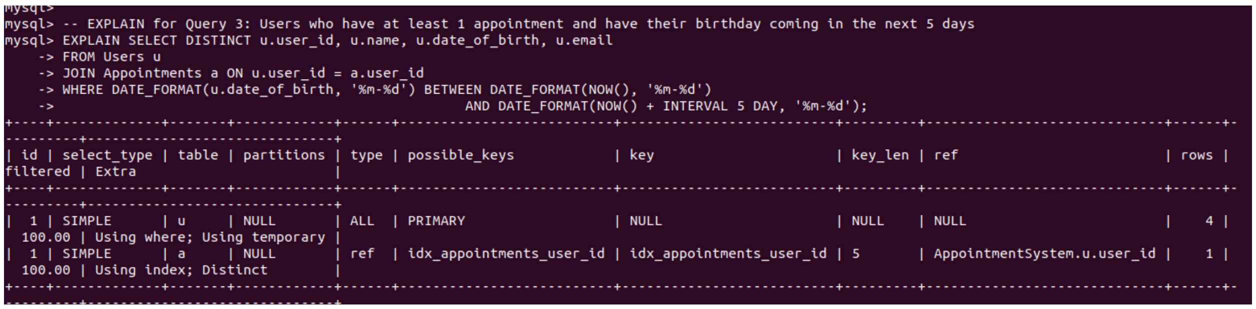
```
mysql> -- EXPLAIN for Query 1: All appointments booked in the last 7 days for a doctor
mysql> EXPLAIN SELECT *
    -> FROM Appointments
    -> WHERE doctor_id = 1
    ->   AND created_at >= NOW() - INTERVAL 7 DAY;
+----+-------------+--------------+------------+------+--------------------------------------------------------+----------------------------+---------
+-------+------+----------+-------------+
| id | select_type | table        | partitions | type | possible_keys                                          | key                        | key_len
| ref   | rows | filtered | Extra       |
+----+-------------+--------------+------------+------+--------------------------------------------------------+----------------------------+---------
+-------+------+----------+-------------+
|  1 | SIMPLE      | Appointments | NULL       | ref  | idx_appointments_doctor_id,idx_appointments_created_at | idx_appointments_doctor_id | 5
| const |    6 |   100.00 | Using where |
+----+-------------+--------------+------------+------+--------------------------------------------------------+----------------------------+---------
+-------+------+----------+-------------+
1 row in set, 1 warning (0.02 sec)
```

2.



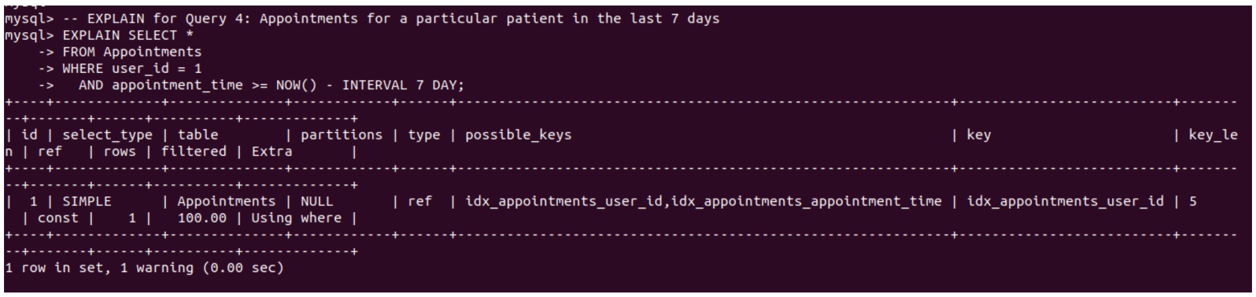
```
mysql> -- EXPLAIN for Query 2: All appointments booked in the last 2 days and scheduled within the next 5 hours for a doctor
mysql> EXPLAIN SELECT *
    -> FROM Appointments
    -> WHERE doctor_id = 1
    ->   AND created_at >= NOW() - INTERVAL 2 DAY
    ->   AND appointment_time <= NOW() + INTERVAL 5 HOUR;
+----+-------------+--------------+------------+------+------------------------------------------------------------------------------------------+----
------------------------+---------+-------+------+----------+-------------+
| id | select_type | table        | partitions | type | possible_keys                                                                            | key
                        | key_len | ref   | rows | filtered | Extra       |
+----+-------------+--------------+------------+------+------------------------------------------------------------------------------------------+----
------------------------+---------+-------+------+----------+-------------+
|  1 | SIMPLE      | Appointments | NULL       | ref  | idx_appointments_doctor_id,idx_appointments_created_at,idx_appointments_appointment_time | idx
_appointments_doctor_id | 5       | const |    6 |    63.00 | Using where |
+----+-------------+--------------+------------+------+------------------------------------------------------------------------------------------+----
------------------------+---------+-------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

3.



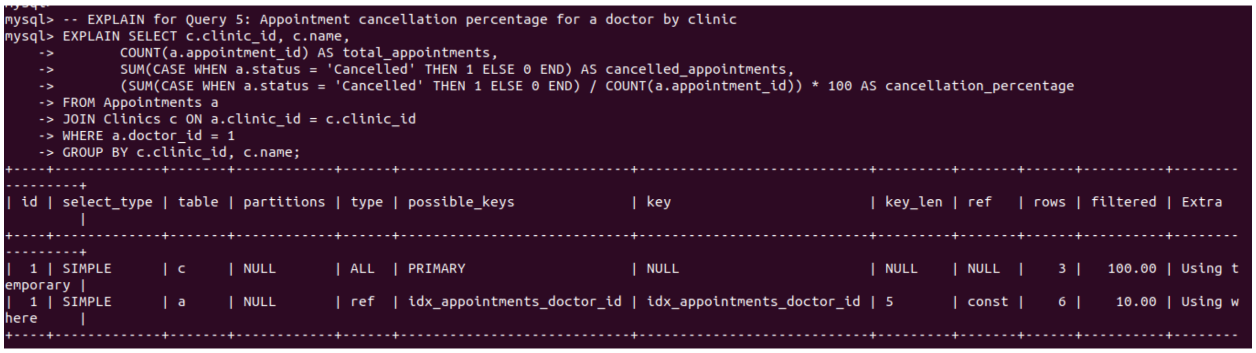
```
mysql>
mysql> -- EXPLAIN for Query 3: Users who have at least 1 appointment and have their birthday coming in the next 5 days
mysql> EXPLAIN SELECT DISTINCT u.user_id, u.name, u.date_of_birth, u.email
    -> FROM Users u
    -> JOIN Appointments a ON u.user_id = a.user_id
    -> WHERE DATE_FORMAT(u.date_of_birth, '%m-%d') BETWEEN DATE_FORMAT(NOW(), '%m-%d')
    ->                                                  AND DATE_FORMAT(NOW() + INTERVAL 5 DAY, '%m-%d');
+----+-------------+-------+------------+------+-------------------------+-------------------------+---------+-----------------------------+------+-
---------+------------------------------+
| id | select_type | table | partitions | type | possible_keys           | key                     | key_len | ref                         | rows |
filtered | Extra                        |
+----+-------------+-------+------------+------+-------------------------+-------------------------+---------+-----------------------------+------+-
---------+------------------------------+
|  1 | SIMPLE      | u     | NULL       | ALL  | PRIMARY                 | NULL                    | NULL    | NULL                        |    4 |
  100.00 | Using where; Using temporary |
|  1 | SIMPLE      | a     | NULL       | ref  | idx_appointments_user_id | idx_appointments_user_id | 5       | AppointmentSystem.u.user_id |    1 |
  100.00 | Using index; Distinct        |
+----+-------------+-------+------------+------+-------------------------+-------------------------+---------+-----------------------------+------+-
---------+------------------------------+
```

4.



```
mysql> -- EXPLAIN for Query 4: Appointments for a particular patient in the last 7 days
mysql> EXPLAIN SELECT *
    -> FROM Appointments
    -> WHERE user_id = 1
    ->   AND appointment_time >= NOW() - INTERVAL 7 DAY;
+----+-------------+--------------+------------+------+-------------------------------------------------------+--------------------------+-------
--+-------+------+----------+-------------+
| id | select_type | table        | partitions | type | possible_keys                                         | key                      | key_le
n | ref   | rows | filtered | Extra       |
+----+-------------+--------------+------------+------+-------------------------------------------------------+--------------------------+-------
--+-------+------+----------+-------------+
|  1 | SIMPLE      | Appointments | NULL       | ref  | idx_appointments_user_id,idx_appointments_appointment_time | idx_appointments_user_id | 5
  | const |    1 |   100.00 | Using where |
+----+-------------+--------------+------------+------+-------------------------------------------------------+--------------------------+-------
--+-------+------+----------+-------------+
1 row in set, 1 warning (0.00 sec)
```

5.



```
mysql> -- EXPLAIN for Query 5: Appointment cancellation percentage for a doctor by clinic
mysql> EXPLAIN SELECT c.clinic_id, c.name,
    ->        COUNT(a.appointment_id) AS total_appointments,
    ->        SUM(CASE WHEN a.status = 'Cancelled' THEN 1 ELSE 0 END) AS cancelled_appointments,
    ->        (SUM(CASE WHEN a.status = 'Cancelled' THEN 1 ELSE 0 END) / COUNT(a.appointment_id)) * 100 AS cancellation_percentage
    -> FROM Appointments a
    -> JOIN Clinics c ON a.clinic_id = c.clinic_id
    -> WHERE a.doctor_id = 1
    -> GROUP BY c.clinic_id, c.name;
+----+-------------+-------+------------+------+----------------------------+----------------------------+---------+-------+------+----------+--------
---------+
| id | select_type | table | partitions | type | possible_keys              | key                        | key_len | ref   | rows | filtered | Extra
         |
+----+-------------+-------+------------+------+----------------------------+----------------------------+---------+-------+------+----------+--------
---------+
|  1 | SIMPLE      | c     | NULL       | ALL  | PRIMARY                    | NULL                       | NULL    | NULL  |    3 |   100.00 | Using t
emporary |
|  1 | SIMPLE      | a     | NULL       | ref  | idx_appointments_doctor_id | idx_appointments_doctor_id | 5       | const |    6 |    10.00 | Using w
here     |
+----+-------------+-------+------------+------+----------------------------+----------------------------+---------+-------+------+----------+--------
```