Database Design & Development for

ROMS (Rapid Operations Management System)

By

Ramkumar Krishnamoorthy

https://github.com/ramkrish1988/DataScience.git

Table of Contents:

1.Introduction

The ROMS provides advanced healthcare services to more than two million people from Ontario, Alberta, British Columbia, and Saskatchewan.

It is one of Canada's most recognized medical facilities and one of the leading research Institute and developing centre. As the healthcare landscape evolves, ROMS remains at the forefront of innovation, investing in research, advanced treatments, and patient care technologies to enhance the quality of life for its patients. Whether facing a routine procedure or a complex medical condition, patients can trust ROMS to provide the expertise, compassion, and care they need.

Services Offered

- General Cardiology Clinic
- Respiratory
- Gynaecology
- Dental
- Paediatrics
- 24/7 emergency care

Clinics Contain

- Behavioural Development
- Intraoperative MR
- Cancer Screening
- X-ray
- Laboratory Services

This document outlines the design of a relational database for ROMS (Rapid Operations Management System), a hospital management system aimed at running multiple hospital locations efficiently and reducing patient wait times. The database supports operations related to patient management, doctor scheduling, department resource allocation, and appointment handling across multiple clinics within a city.

The goal is to provide a highly efficient structure that ensures quick access to essential information, minimizes bottlenecks, and reduces patient wait time through proper resource management.

2.a Mission:

Ensure that patients receive prompt, high-quality care, the Rapid Operations Management System (ROMS) optimizes hospital operations across various locations. ROMS seeks to reduce bottlenecks and considerably decrease patient wait times by streamlining appointment handling, doctor scheduling, department resource allocation, and patient management using an effective and scalable relational database. Our mission is to equip medical professionals with the resources they require to serve populations inside cities with seamless, well-coordinated, and responsive medical care.

2.b Objectives:

Efficient Data Management: Develop a robust relational database structure that efficiently handles large volumes of data related to patients, doctors, appointments, and hospital resources, ensuring easy access and retrieval of critical information.

Reduced Patient Wait Time: Implement optimized scheduling and resource allocation processes to minimize bottlenecks and reduce patient wait times across all hospital locations.

Centralized Operations Control: Provide a centralized platform for managing multiple clinics and hospital locations within a city, enabling real-time updates, monitoring, and decision-making across departments.

Scalability and Flexibility: Design the database to be scalable, allowing ROMS to accommodate future growth and adapt to the changing needs of different hospital branches and healthcare departments.

Enhanced Resource Allocation: Ensure effective management of hospital resources, such as staff, equipment, and facilities, by integrating a dynamic resource allocation system that supports optimal utilization.

Doctor and Staff Scheduling: Provide a streamlined scheduling system for doctors and healthcare professionals, ensuring proper distribution of workload and availability to meet patient demand efficiently.

Improved Patient Experience: Create a user-friendly interface that supports smooth patient registration, appointment booking, and management, enhancing the overall patient experience and satisfaction.

3.Database Tables

S no	Table Name	Description
1	Clinic Table	This table stores details about each clinic (or branch) of the hospital.
2	Department Table	The Department table manages the various medical departments available in each clinic.
3	Doctor Table	Stores information about doctors in the hospital.
4	Doctor Schedule Table	Tracks the availability of doctors across different clinics, helping to reduce patient wait times by optimizing doctor schedules.
5	Patient Table	Tracks patients, their medical conditions, and associated appointments.
6	Appointment Table	Handles patient appointments and their schedules, connecting patients, doctors, and clinics.
7	Staff Table	Manages non-doctor staff, such as nurses, technicians, or administrative personnel.
8	Resource Table	Tracks hospital resources (e.g., operating rooms, ICU beds) and their availability.

4.Why this tables are Important?

Structured Data Management: By organizing hospital operations into distinct entities (clinics, departments, doctors, patients, resources), these tables allow the database to maintain clear, structured, and easily retrievable information.

Efficient Resource Allocation: Tracking doctors, staff, and resources helps ensure that the hospital uses its resources efficiently, reducing waste and patient wait times.

Scalability: The use of separate tables for different entities (clinics, departments, resources) makes the database scalable, enabling it to handle the addition of new locations or services seamlessly.

Each table plays a vital role in creating a cohesive and functional hospital management system that aligns with ROMS's goals of efficiency and improving patient experiences.

5.Fields in the Database Tables

5.1. Clinic Table (Hospital Locations)

Field Name	Data Type	Description
Clinicid	INTEGER	**Primary key**, unique identifier for clinic
clinic_name	VARCHAR(100)	Name of the clinic
Address	VARCHAR(200)	Address of the clinic
Region	VARCHAR(100)	Region of the city where clinic is located
avg_patient_perday	INTEGER	Average number of patients per day
avg_appointment_request_perday	INTEGER	Average number of appointment requests per day

5.2. Department Table

Field Name	Data Type	Description
Departmentid	INTEGER	Primary key, unique identifier for department
Clinicid	INTEGER	Foreign key, links to the clinic
Department_name	VARCHAR(100)	Name of the department (e.g., Surgery, Pediatrics)

5.3. Doctor Table

Field Name	Data Type	Description
Doctorid	INTEGER	Primary key, unique identifier for doctor
Name	VARCHAR(100)	Name of the doctor
Specialization	VARCHAR(100)	Doctor's specialization (e.g., Surgeon)
Clinicid	INTEGER	Foreign key, links to the clinic
Departmentid	INTEGER	Foreign key, links to the department
Noof_patients_perday	INTEGER	Number of patients the doctor can see per day

5.4. DoctorSchedule Table

Field Name	Data Type	Description
Doctorid	INTEGER	Foreign key, links to the doctor
Clinicid	INTEGER	Foreign key, links to the clinic
Monday	TIME	Doctor's available time on Monday
Tuesday	TIME	Doctor's available time on Tuesday
Wednesday	TIME	Doctor's available time on Wednesday
Thursday	TIME	Doctor's available time on Thursday
Friday	TIME	Doctor's available time on Friday
Saturday	TIME	Doctor's available time on Saturday
Sunday	TIME	Doctor's available time on Sunday

5.5. Patient Table

Field Name	Data Type	Description
Patientid	INTEGER	Primary key, unique identifier for patient
Name	VARCHAR(100)	Patient's name
Gender	VARCHAR(10)	Patient's gender
Age	INTEGER	Patient's age
Symptoms	VARCHAR(200)	Symptoms reported by the patient
Diagnosis	VARCHAR(2000)	Doctor's diagnosis for the patient

5.6. Appointment Table

Field Name	Data Type	Description
Appointmentid	INTEGER	Primary key, unique identifier for appointment
Clinicid	INTEGER	Foreign key, links to the clinic
Patientid	INTEGER	Foreign key, links to the patient
Doctorid	INTEGER	Foreign key, links to the doctor
Appointment_type	VARCHAR(50)	Type of appointment (e.g., Consultation, Surgery)
Scheduled_time	TIME	Scheduled time of the appointment
Diagnosis	VARCHAR(2000)	Diagnosis or purpose of the appointment

5.7. Staff Table

Field Name	Data Type	Description
Staffid	INTEGER	Primary key, unique identifier for staff
Name	VARCHAR(100)	Staff member's name
Role	VARCHAR(100)	Staff's role in the clinic (e.g., Nurse)
Clinicid	INTEGER	Foreign key, links to the clinic
Departmentid	INTEGER	Foreign key, links to the department

5.8. Resource Table

Field Name	Data Type	Description
Resourceid	INTEGER	Primary key, unique identifier for resource
Resource_name	VARCHAR(100)	Name of the resource (e.g., Operating Room)
Availability_status	VARCHAR(50)	Status of resource (e.g., Available, In-use)
Clinicid	INTEGER	Foreign key, links to the clinic
Departmentid	INTEGER	Foreign key, links to the department

6.Relationship types between Tables

6.1. One-to-Many Relationships:

- **Clinic ↔ Department**: One clinic can have multiple departments, but each department belongs to only one clinic.

  Clinic.Clinicid → Department.Clinicid

- **Clinic ↔ Doctor**: One clinic can have many doctors, but each doctor works in only one clinic.

  Clinic.Clinicid → Doctor.Clinicid

- **Clinic ↔ Appointment**: One clinic can have many appointments, but each appointment takes place in only one clinic.

  Clinic.Clinicid → Appointment.Clinicid

- **Doctor ↔ Appointment**: One doctor can have many appointments, but each appointment is assigned to only one doctor.

  Doctor.Doctorid → Appointment.Doctorid

- **Patient ↔ Appointment**: One patient can have multiple appointments, but each appointment is linked to only one patient.

  Patient.Patientid → Appointment.Patientid

- **Department ↔ Staff**: One department can have many staff members, but each staff member works in one department.

  Department.Departmentid → Staff.Departmentid

6.2.**One-to-One Relationships**:

- **Appointment ↔ AppointmentType**: Each appointment has one type, and each appointment type is linked to a specific appointment.

  Appointment.Appointmentid → AppointmentType.Appointmentid

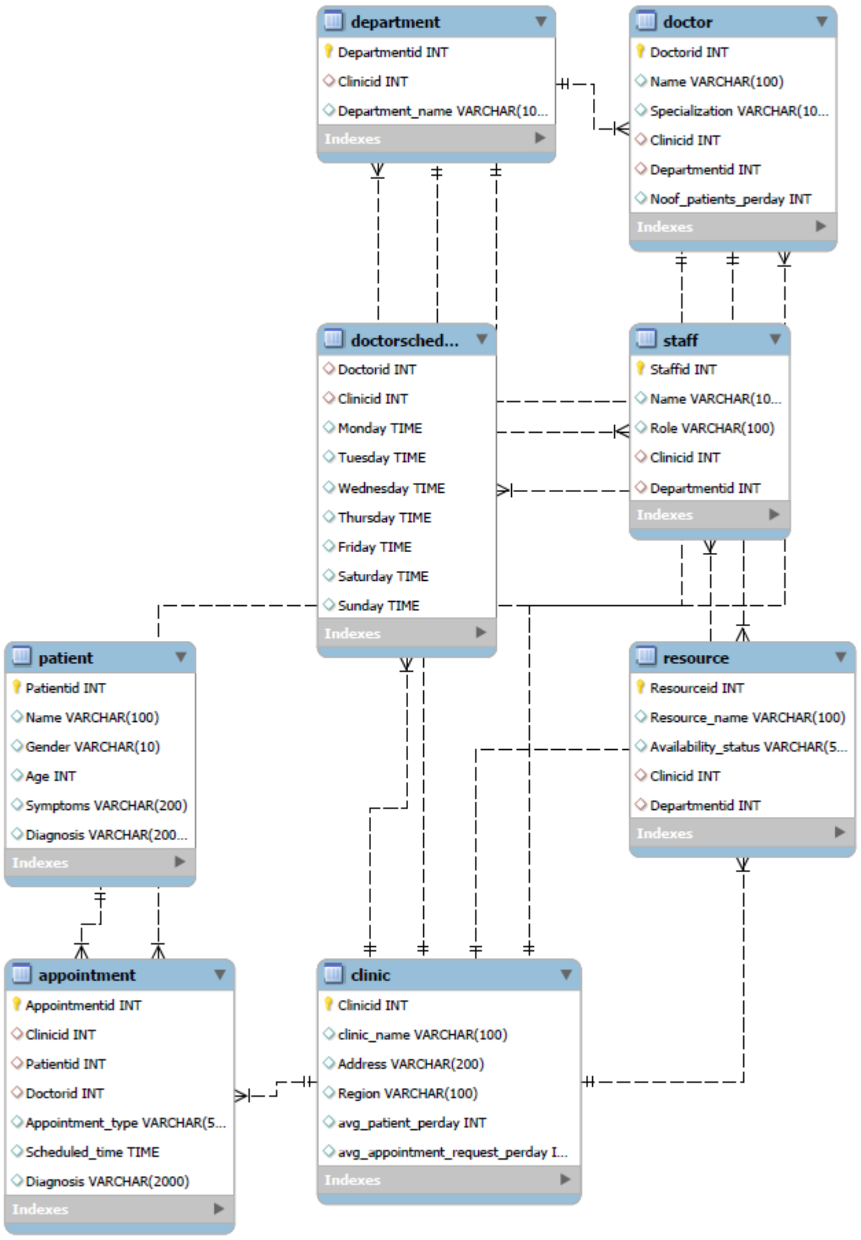
6.3. **Many-to-Many Relationships**:

- **Doctor ↔ DoctorSchedule**: A doctor may have different schedules across different clinics, and each clinic can have multiple doctors with different schedules. This forms a many-to-many relationship that is handled by DoctorSchedule, where each doctor has a schedule at different clinics.

  Doctor.Doctorid ↔ DoctorSchedule.Doctorid

  Clinic.Clinicid ↔ DoctorSchedule.Clinicid

7.ER Diagram


department
Departmentid INT
Clinicid INT
Department_name VARCHAR(10...
Indexes
doctor
Doctorid INT
Name VARCHAR(100)
Specialization VARCHAR(10...
Clinicid INT
Departmentid INT
Noof_patients_perday INT
Indexes
doctorsched...
Doctorid INT
Clinicid INT
Monday TIME
Tuesday TIME
Wednesday TIME
Thursday TIME
Friday TIME
Saturday TIME
Sunday TIME
Indexes
staff
Staffid INT
Name VARCHAR(10...
Role VARCHAR(100)
Clinicid INT
Departmentid INT
Indexes
patient
Patientid INT
Name VARCHAR(100)
Gender VARCHAR(10)
Age INT
Symptoms VARCHAR(200)
Diagnosis VARCHAR(200...
Indexes
resource
Resourceid INT
Resource_name VARCHAR(100)
Availability_status VARCHAR(5...
Clinicid INT
Departmentid INT
Indexes
appointment
Appointmentid INT
Clinicid INT
Patientid INT
Doctorid INT
Appointment_type VARCHAR(5...
Scheduled_time TIME
Diagnosis VARCHAR(2000)
Indexes
clinic
Clinicid INT
clinic_name VARCHAR(100)
Address VARCHAR(200)
Region VARCHAR(100)
avg_patient_perday INT
avg_appointment_request_perday I...
Indexes

8. Developing Database for ROMS

8.1.Clinic Table

```
CREATE TABLE Clinic (
    Clinicid INTEGER PRIMARY KEY,
    clinic_name VARCHAR(100),
    Address VARCHAR(200),
    Region VARCHAR(100),
    avg_patient_perday INTEGER,
    avg_appointment_request_perday INTEGER
);
```

8.2. Department Table

```
CREATE TABLE Department (
    Departmentid INTEGER PRIMARY KEY,
    Clinicid INTEGER,
    Department_name VARCHAR(100),
    FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid)
);
```

8.3. Doctor Table

```
CREATE TABLE Doctor (
    Doctorid INTEGER PRIMARY KEY,
    Name VARCHAR(100),
    Specialization VARCHAR(100),
    Clinicid INTEGER,
    Departmentid INTEGER,
    Noof_patients_perday INTEGER,
    FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid),
    FOREIGN KEY (Departmentid) REFERENCES Department(Departmentid) );
```

8.4. DoctorSchedule Table

```
CREATE TABLE DoctorSchedule (
   Doctorid INTEGER,
   Clinicid INTEGER,
   Monday TIME,
   Tuesday TIME,
   Wednesday TIME,
   Thursday TIME,
  Friday TIME,
   Saturday TIME,
   Sunday TIME,
   FOREIGN KEY (Doctorid) REFERENCES Doctor(Doctorid),
   FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid)
);
```

8.5. Patient Table

```
CREATE TABLE Patient (
   Patientid INTEGER PRIMARY KEY,
   Name VARCHAR(100),
   Gender VARCHAR(10),
   Age INTEGER,
   Symptoms VARCHAR(200),
   Diagnosis VARCHAR(2000)
);
```

8.6. Appointment Table

```
CREATE TABLE Appointment (
   Appointmentid INTEGER PRIMARY KEY,
   Clinicid INTEGER,
   Patientid INTEGER,
   Doctorid INTEGER,
   Appointment_type VARCHAR(50),
   Scheduled_time TIME,
   Diagnosis VARCHAR(2000),
   FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid),
   FOREIGN KEY (Patientid) REFERENCES Patient(Patientid),
   FOREIGN KEY (Doctorid) REFERENCES Doctor(Doctorid)
);
```

8.7. Staff Table

```
CREATE TABLE Staff (
    Staffid INTEGER PRIMARY KEY,
    Name VARCHAR(100),
    Role VARCHAR(100),
    Clinicid INTEGER,
    Departmentid INTEGER,
    FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid),
    FOREIGN KEY (Departmentid) REFERENCES Department(Departmentid)
);
```

8.8. Resource Table

```
CREATE TABLE Resource (
    Resourceid INTEGER PRIMARY KEY,
    Resource_name VARCHAR(100),
    Availability_status VARCHAR(50),
    Clinicid INTEGER,
    Departmentid INTEGER,
    FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid),
    FOREIGN KEY (Departmentid) REFERENCES Department(Departmentid)
);
```

9.Conclusion

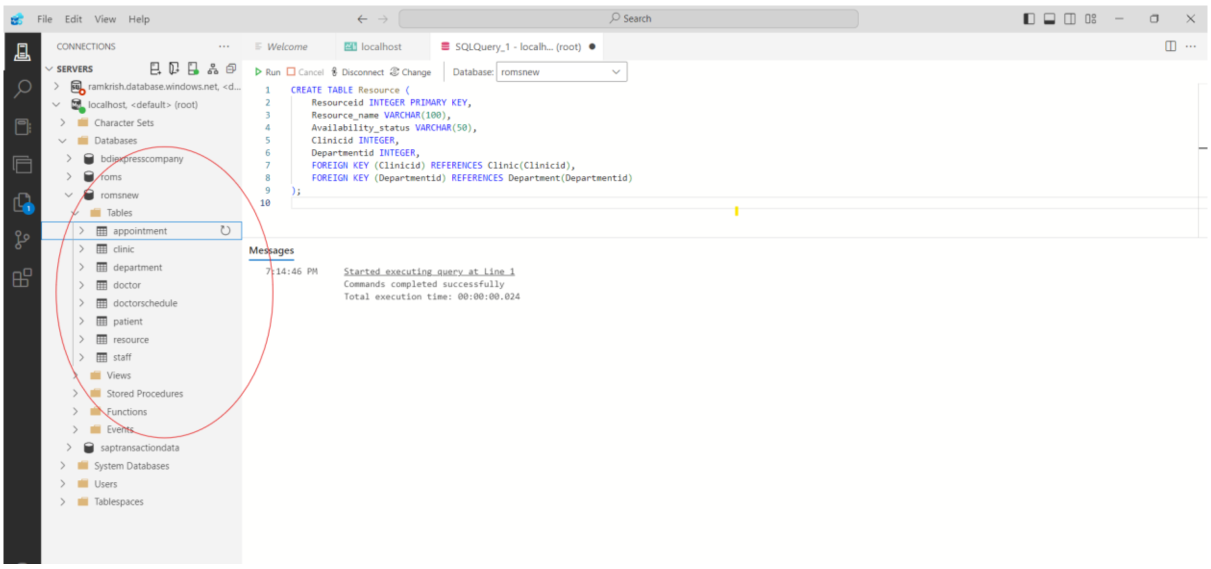
The ROMS (Rapid Operations Management System) hospital database is designed to streamline hospital operations and improve patient care across multiple clinic locations within a city. By focusing on efficient data management, the system reduces patient wait times, enhances resource allocation, and ensures smooth scheduling of appointments and doctor availability.

The structured database incorporates essential entities such as Clinic, Doctor, Patient, Appointment, and Department, all linked through well-defined relationships. This design allows hospitals to efficiently manage multiple locations, optimize staff and doctor schedules, and monitor patient care in real time. Through a robust relational model, the database not only supports the day-to-day operations but also enables predictive analysis for resource utilization, helping hospitals anticipate patient needs and improve diagnostic efficiency.

By integrating real-time scheduling and resource management, the ROMS database enables better patient flow, minimizes administrative overhead, and enhances overall hospital performance. This comprehensive system offers the foundation for hospitals to continue evolving with advancements in healthcare technology, ultimately improving patient satisfaction and care outcomes.

10 Appendix

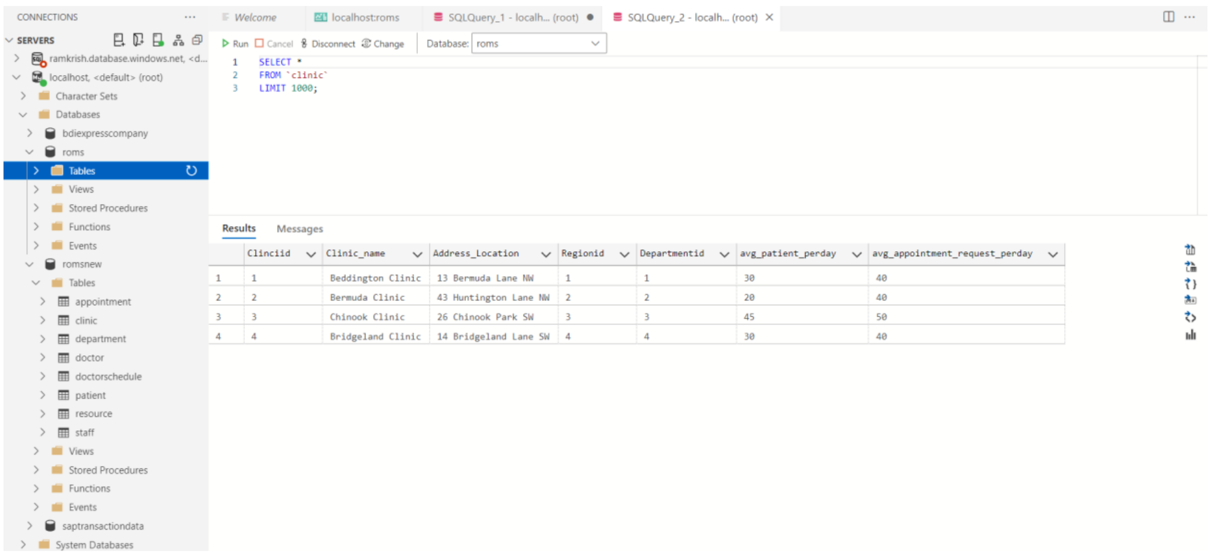
10.1 New Database and tables Created.



```
CREATE TABLE Resource (
    Resourceid INTEGER PRIMARY KEY,
    Resource_name VARCHAR(100),
    Availability_status VARCHAR(50),
    Clinicid INTEGER,
    Departmentid INTEGER,
    FOREIGN KEY (Clinicid) REFERENCES Clinic(Clinicid),
    FOREIGN KEY (Departmentid) REFERENCES Department(Departmentid)
);
```

```
7:14:46 PM    Started executing query at Line 1
              Commands completed successfully
              Total execution time: 00:00:00.024
```

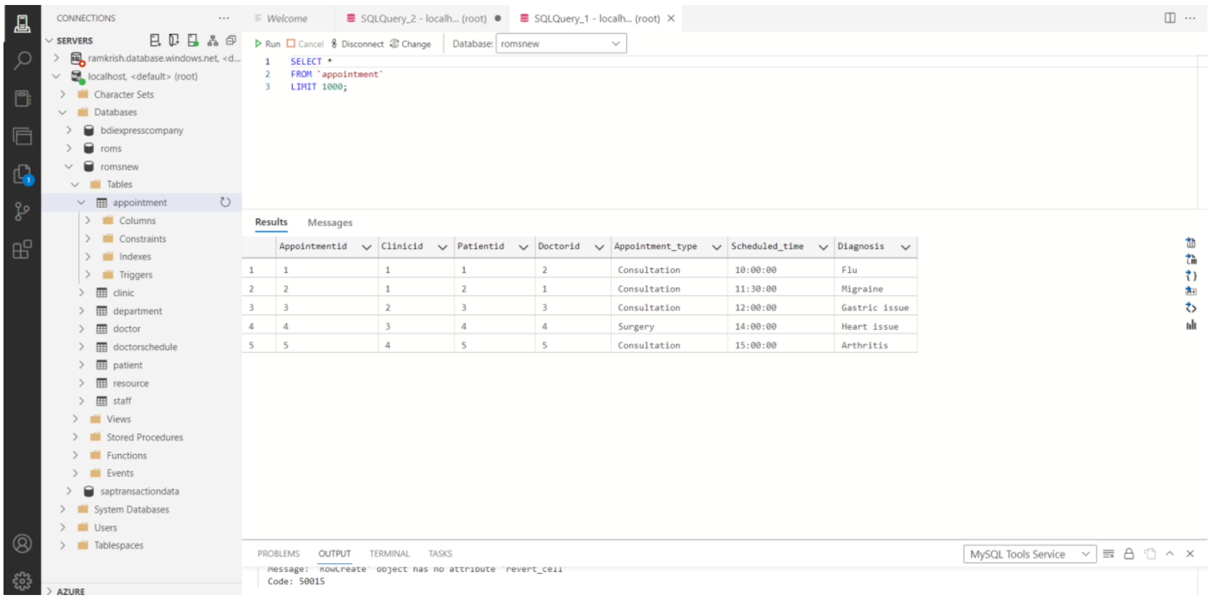
10.2 Values in the table.



```
SELECT *
FROM `clinic`
LIMIT 1000;
```

	Clinciid	Clinic_name	Address_Location	Regionid	Departmentid	avg_patient_perday	avg_appointment_request_perday
1	1	Beddington Clinic	13 Bermuda Lane NW	1	1	30	40
2	2	Bermuda Clinic	43 Huntington Lane NW	2	2	20	40
3	3	Chinook Clinic	26 Chinook Park SW	3	3	45	50
4	4	Bridgeland Clinic	14 Bridgeland Lane SW	4	4	30	40

10.3 Values in the Appointment Table

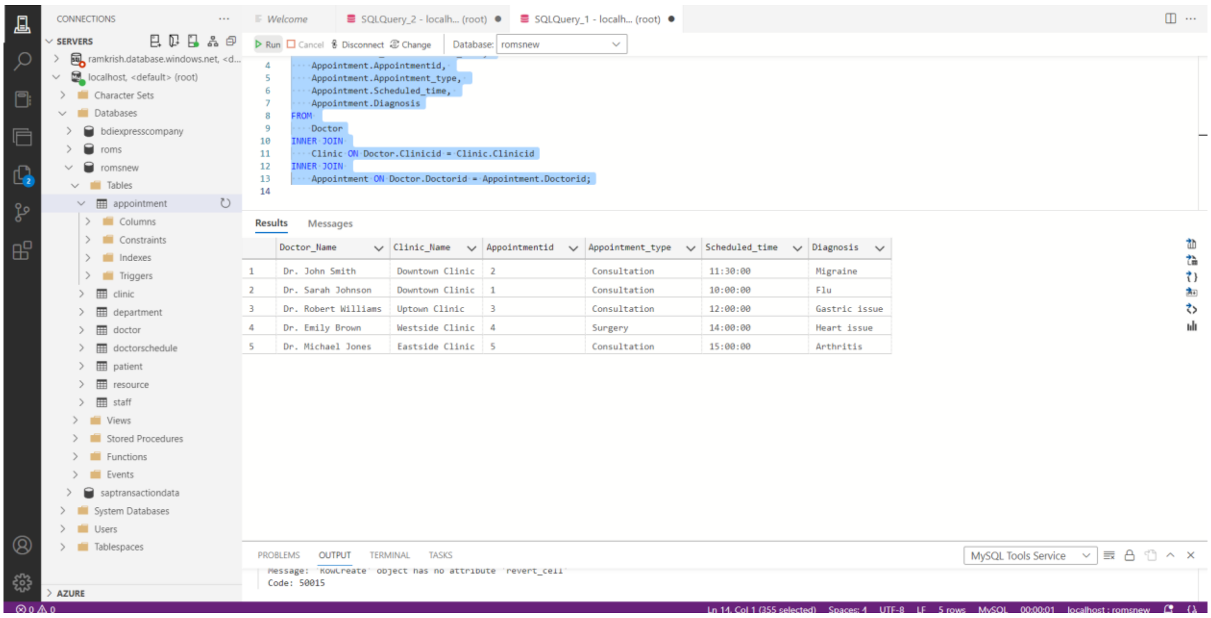


```
SELECT *
FROM `appointment`
LIMIT 1000;
```

	Appointmentid	Clinicid	Patientid	Doctorid	Appointment_type	Scheduled_time	Diagnosis
1	1	1	1	2	Consultation	10:00:00	Flu
2	2	1	2	1	Consultation	11:30:00	Migraine
3	3	2	3	3	Consultation	12:00:00	Gastric issue
4	4	3	4	4	Surgery	14:00:00	Heart issue
5	5	4	5	5	Consultation	15:00:00	Arthritis

```
Message: `RowCreate` object has no attribute `revert_cell`
Code: 50015
```

10.4 Inner Join Query to see Appointment



```
    Appointment.Appointmentid,
    Appointment.Appointment_type,
    Appointment.Scheduled_time,
    Appointment.Diagnosis
FROM
    Doctor
INNER JOIN
    Clinic ON Doctor.Clinicid = Clinic.Clinicid
INNER JOIN
    Appointment ON Doctor.Doctorid = Appointment.Doctorid;
```

	Doctor_Name	Clinic_Name	Appointmentid	Appointment_type	Scheduled_time	Diagnosis
1	Dr. John Smith	Downtown Clinic	2	Consultation	11:30:00	Migraine
2	Dr. Sarah Johnson	Downtown Clinic	1	Consultation	10:00:00	Flu
3	Dr. Robert Williams	Uptown Clinic	3	Consultation	12:00:00	Gastric issue
4	Dr. Emily Brown	Westside Clinic	4	Surgery	14:00:00	Heart issue
5	Dr. Michael Jones	Eastside Clinic	5	Consultation	15:00:00	Arthritis

```
Message: `RowCreate` object has no attribute `revert_cell`
Code: 50015
```