Lesley Garcia

Professor Yu

CS460 Senior Project Development

24 November 2025

Introduction

The Healthcare Information System for Small Clinics is a secure, web-based platform that enables small healthcare practices to manage patient records, appointments, and billing efficiently through a centralized digital system.

Vision Statement

To empower small clinics with an affordable, user-friendly, and HIPAA-compliant information system that improves patient care, reduces administrative errors, and supports scalable growth.

Project Goals

- Develop a centralized database for patients, appointments, and billing.
- Provide an intuitive web interface for clinic staff with role-based access.
- Implement robust security controls (encryption, authentication, audit logging).
- Automate routine administrative tasks to improve efficiency.
- Provide documentation and training materials for deployment and maintenance.

System Analysis

3.1 Analysis Activities — Functional Requirements

- **Patient Management:** add, update, delete, and retrieve patient records.
- **Appointment Scheduling:** create, update, cancel appointments; prevent double-booking.
- **Billing:** generate invoices, accept/record payments, track outstanding balances.
- **User Management:** secure login, role-based permissions (Admin, Doctor, Receptionist, Billing).
- **Reporting:** produce daily/weekly/monthly reports (visits, revenue, outstanding balances).
- **Data Export:** export reports/data as CSV or PDF.

Nonfunctional Requirements

- **Usability:** intuitive UI with minimal training.
- **Reliability:** target 99% uptime; daily automatic backups.
- **Performance:** support 20+ concurrent users, typical query response < 2s.
- **Security:** encryption (at rest & in transit), audit logging, HIPAA alignment.
- **Portability:** browser compatible (Chrome/Edge/Firefox).
- **Maintainability:** modular code, documentation and tests.

Workflows

Patient Registration: Receptionist logs in → enters new patient info → system validates → data stored → confirmation displayed.

- **Appointment Scheduling:** Staff selects patient → checks available slots → confirm appointment → system updates schedule + sends confirmation.
- **Billing:** Billing staff retrieves visit record → generates invoice → records payment → updates billing history.

3.4 Use Cases

- **Register Patient** — *Actor:* Receptionist — *Precondition:* authenticated — *Flow:* enter data → validate → save → confirm.

- **Schedule Appointment** — *Actor:* Receptionist/Doctor — checks availability → reserve slot → confirm.
- **Process Billing** — *Actor:* Billing Staff — create invoice → record payment → update account.
- **View Reports** — *Actor:* Admin — select report parameters → generate → export.

3.5 ERD

Entities & keys:

- Patient (PatientID PK, Name, DOB, Address, Contact, MedicalHistory)
- Appointment (AppointmentID PK, PatientID FK, Doctor, Date, Time, Status)
- Billing (BillID PK, PatientID FK, Amount, PaymentStatus, DateIssued)
- User (UserID PK, Name, Role, Username, PasswordHash)

Relationships: Patient 1→* Appointment, Appointment 1→1 Billing. Users interact based on role.

3.6 Inputs & Outputs (SSD textual)

- **Inputs:** patient demographics, appointment request, payment amount, user credentials.
- **Outputs:** confirmation messages, invoices, appointment reminders, reports (CSV/PDF).
- **Example SSD text:** Receptionist → System: AddPatient(data). System → DB: INSERT Patient. DB → System: success. System → Receptionist: confirmation.

System Design

4.1 Environment (hardware, OS, network)

- **Client:** Windows 10/11 or macOS, 8GB RAM, modern browser.
- **Server:** Cloud VM (AWS/Azure/DigitalOcean) or on-prem Linux (Ubuntu 22.04), 2 vCPU, 4GB RAM min for prototype.
- **Network:** HTTPS/TLS for all traffic; VPN for admin; firewall rules and secure subnets.

4.2 Architecture and Software

Three-tier architecture:

- **Presentation layer:** HTML/CSS/JS (Bootstrap for quick UI).
- **Application layer:** Python (Flask for prototype; Django if scaling up).
- **Data layer:** MySQL or PostgreSQL (SQLAlchemy ORM).
  Supporting tools: Nginx (reverse proxy), Gunicorn (WSGI), Certbot (TLS), Git for version control.

4.3 User Interfaces

Screens to build:

- Login (role-based redirect).
- Dashboard (today's appointments, quick stats).
- Patient management (add/search/edit).
- Appointment scheduler (calendar+slot picker).
- Billing (generate invoices, payments).
- Reports (filter, export).

4.4 System Interfaces

- **DB Interface:** SQLAlchemy or direct SQL.
- **Web API:** REST endpoints for CRUD operations.

- **External (future):** Insurance/EHR APIs, OAuth2 for SSO.

4.5 Database Design

```
CREATE TABLE Patient (
  PatientID INT AUTO_INCREMENT PRIMARY KEY,
  Name VARCHAR(100),
  DOB DATE,
  Address VARCHAR(255),
  Contact VARCHAR(20),
  MedicalHistory TEXT
);
```

4.6 Design Systems, Control, and Security

- HTTPS/TLS, database encryption for sensitive fields.
- Password hashing (bcrypt) and optional MFA.
- RBAC (least privilege).
- Input validation & prepared statements (prevent SQLi).
- Audit logging & session timeouts.
- Daily backups and offsite retention; regular patching and vulnerability scans.

Project Management

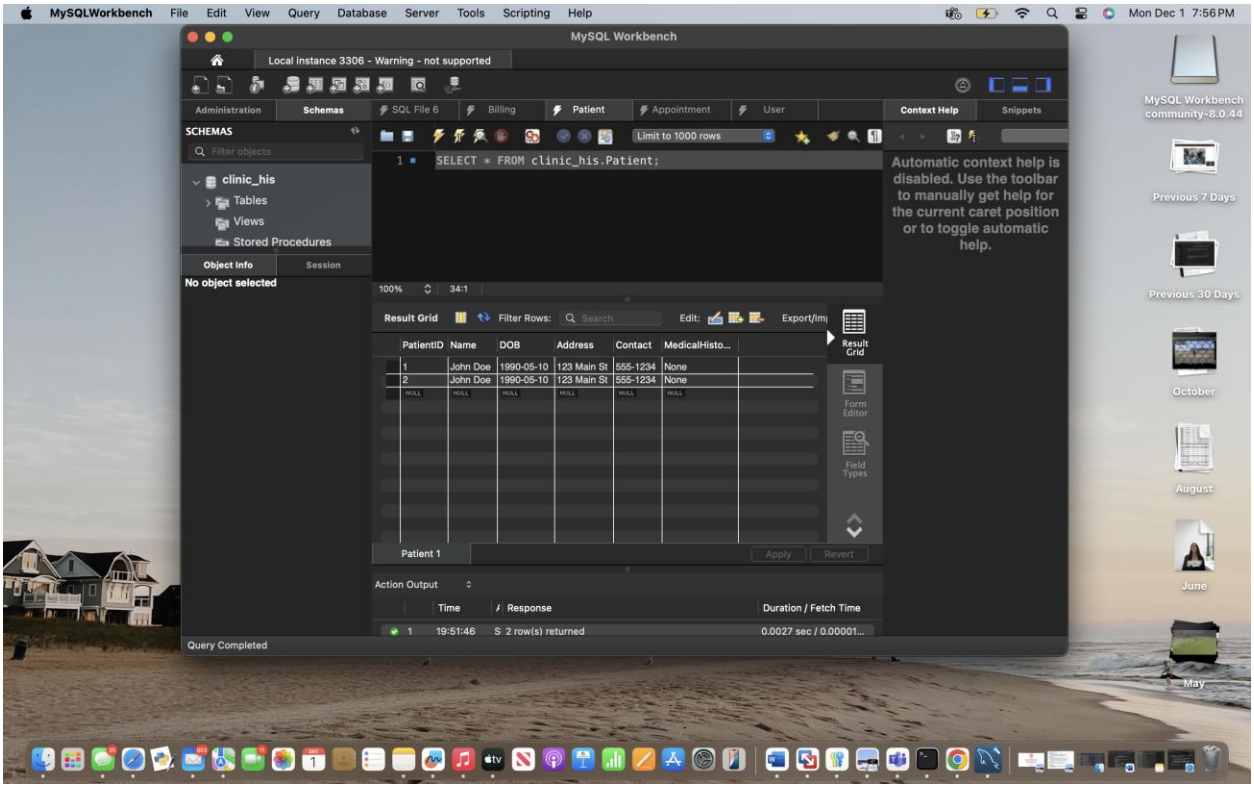
Risk	Likelihood	Impact	Mitigation
Data breech	Medium	High	Encryption, RBAC, MFA, audits
Ransomware	Low	High	Offline/offsite backups, anti-malware
Server downtime	Low	Medium	Redundant hosting, monitoring & alerts
Human error	Medium	Medium	Validation, training, soft-delete recovery
HIPAA noncompliance	Low	High	Policy reviews, audits, documentation

5.2 Schedule / Plan for Implementation

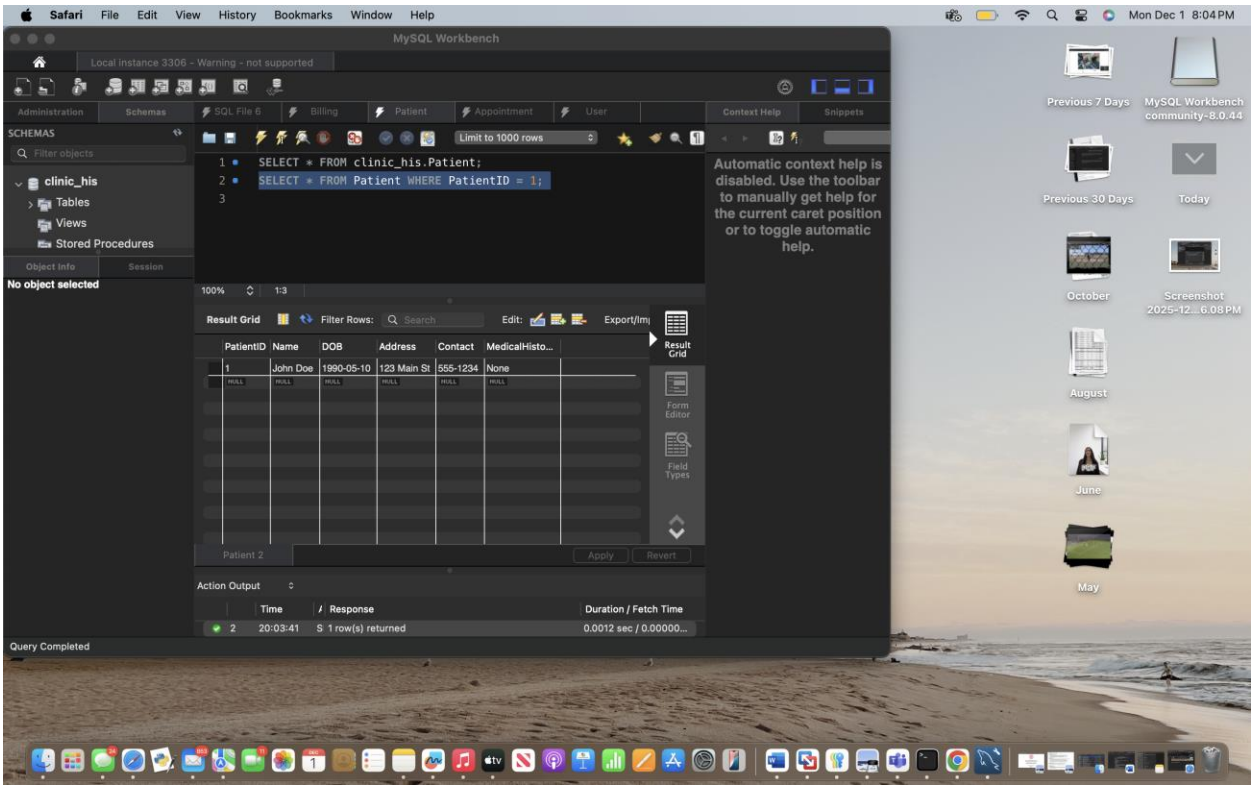
- **Week 1:** Finalize requirements, SDLC plan, development environment setup.
- **Week 2:** Design ERD and create database; seed sample data.
- **Week 3:** Implement backend (Flask) and REST endpoints for Patient and Appointment.
- **Week 4:** Implement UI pages for Login, Dashboard, Patient Management.
- **Week 5:** Implement Billing module & reporting; add export features.

6. Testing

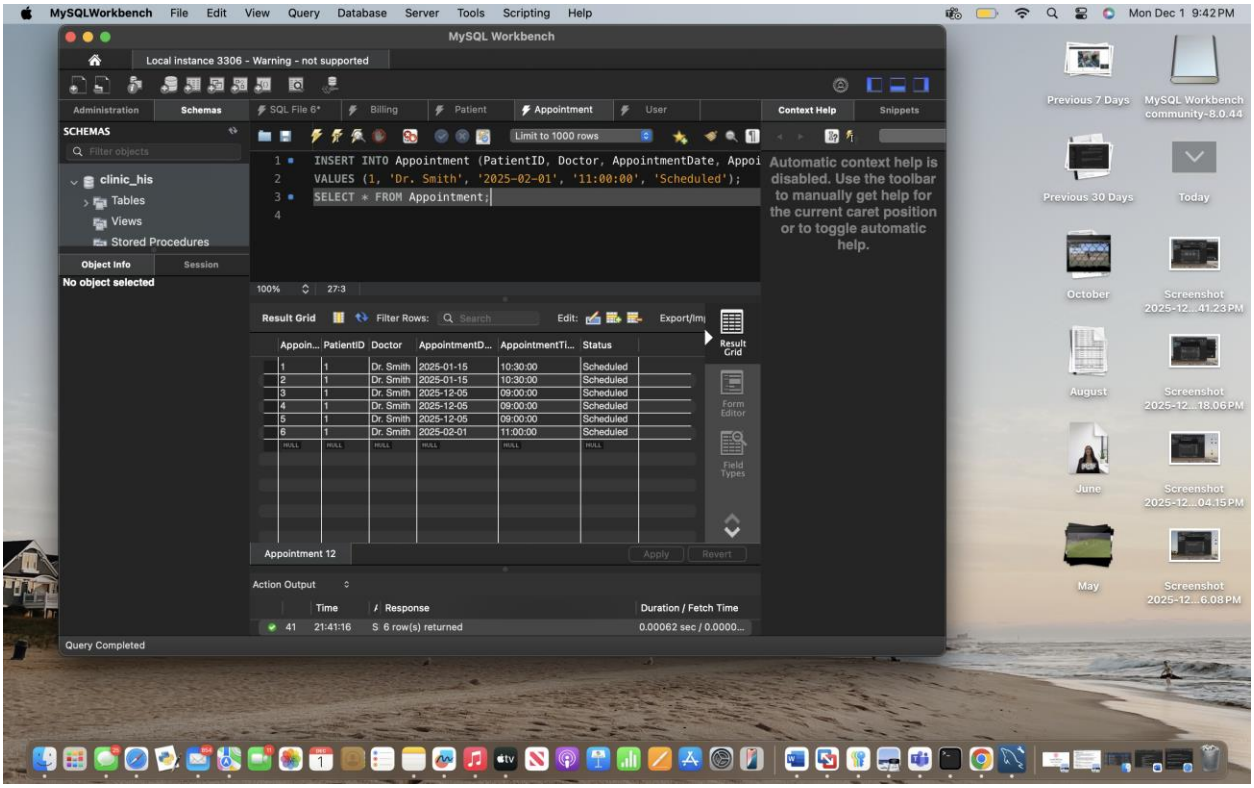
Test Case 1: Create Patient Record & Retrieve Patient Records



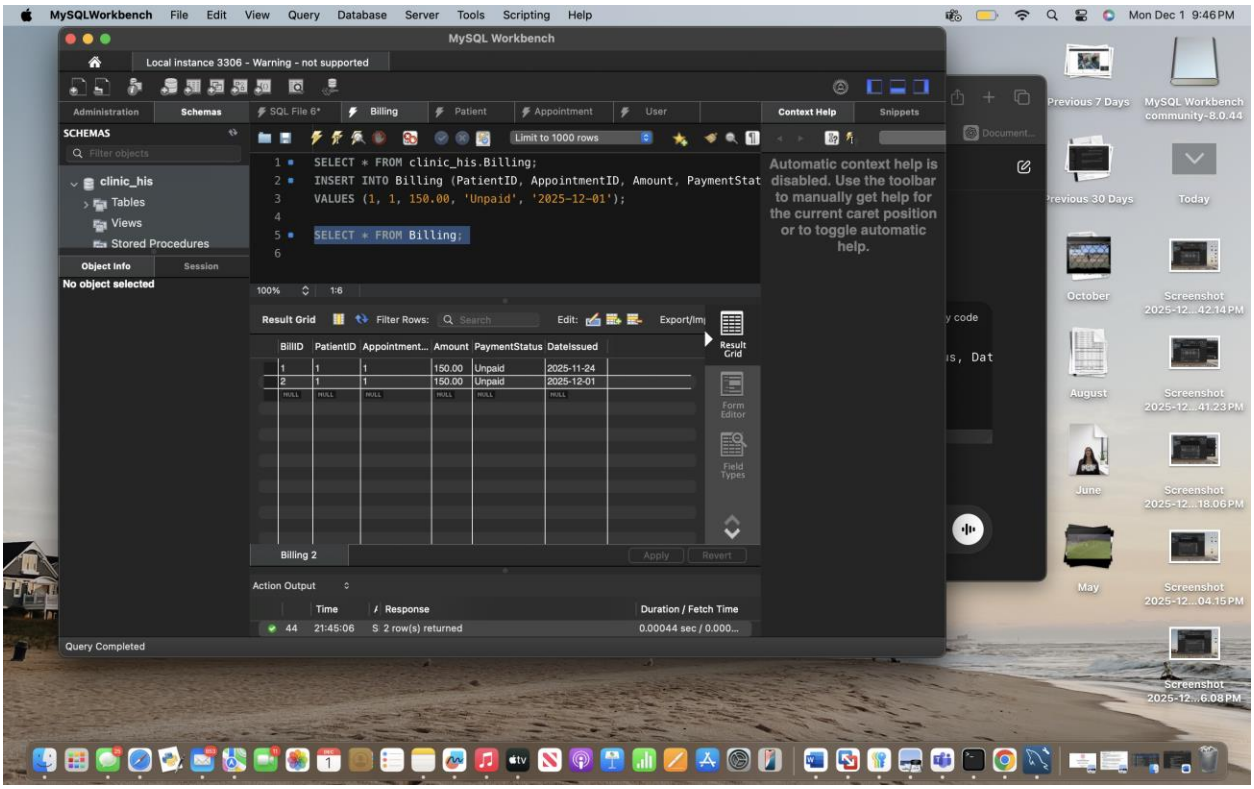
Test Case 3: Update Patient Record



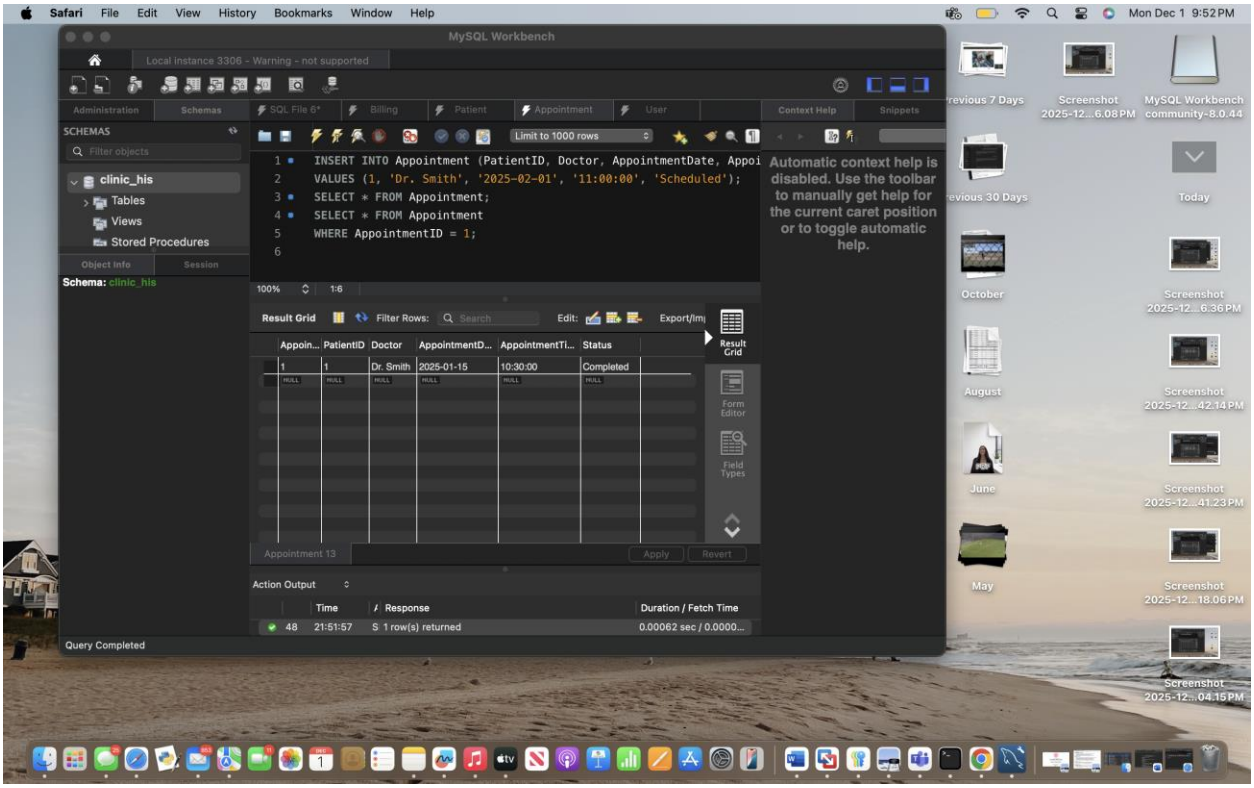
Test Case 4: Create Appointment Record



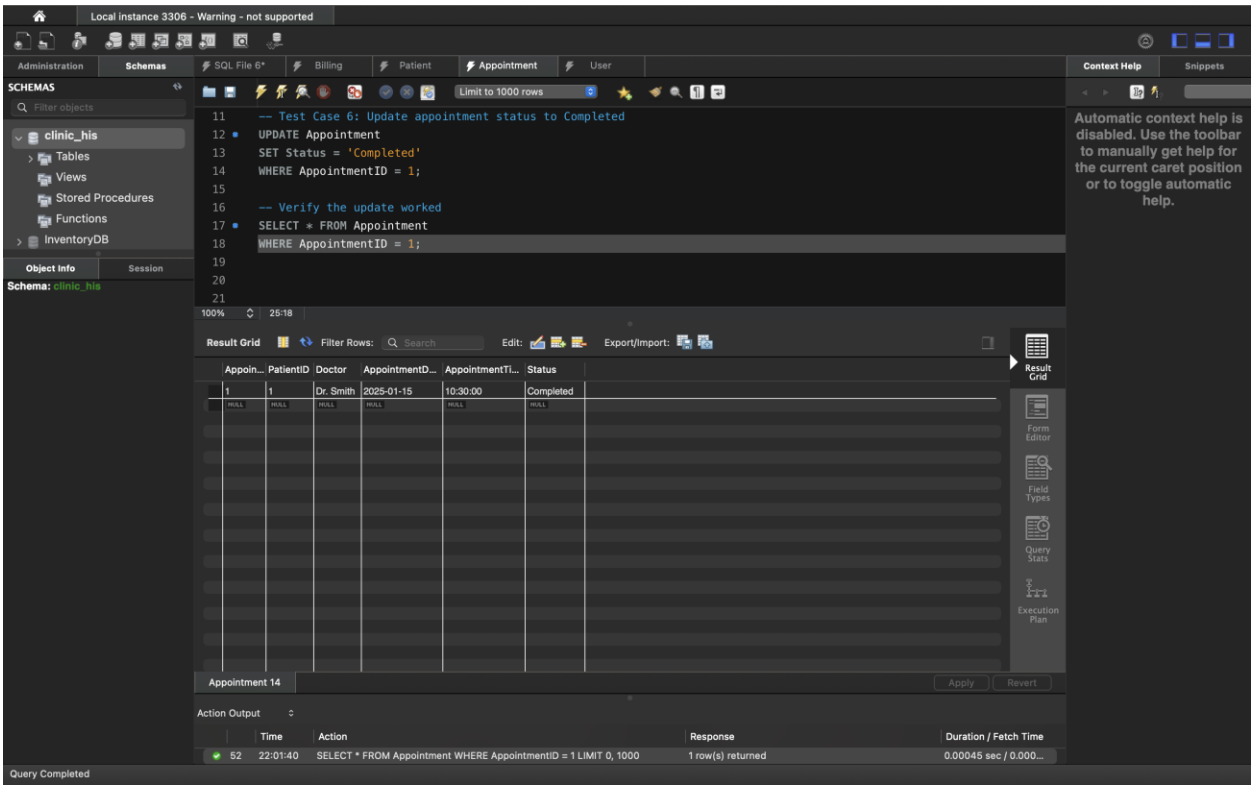
Test Case 4:



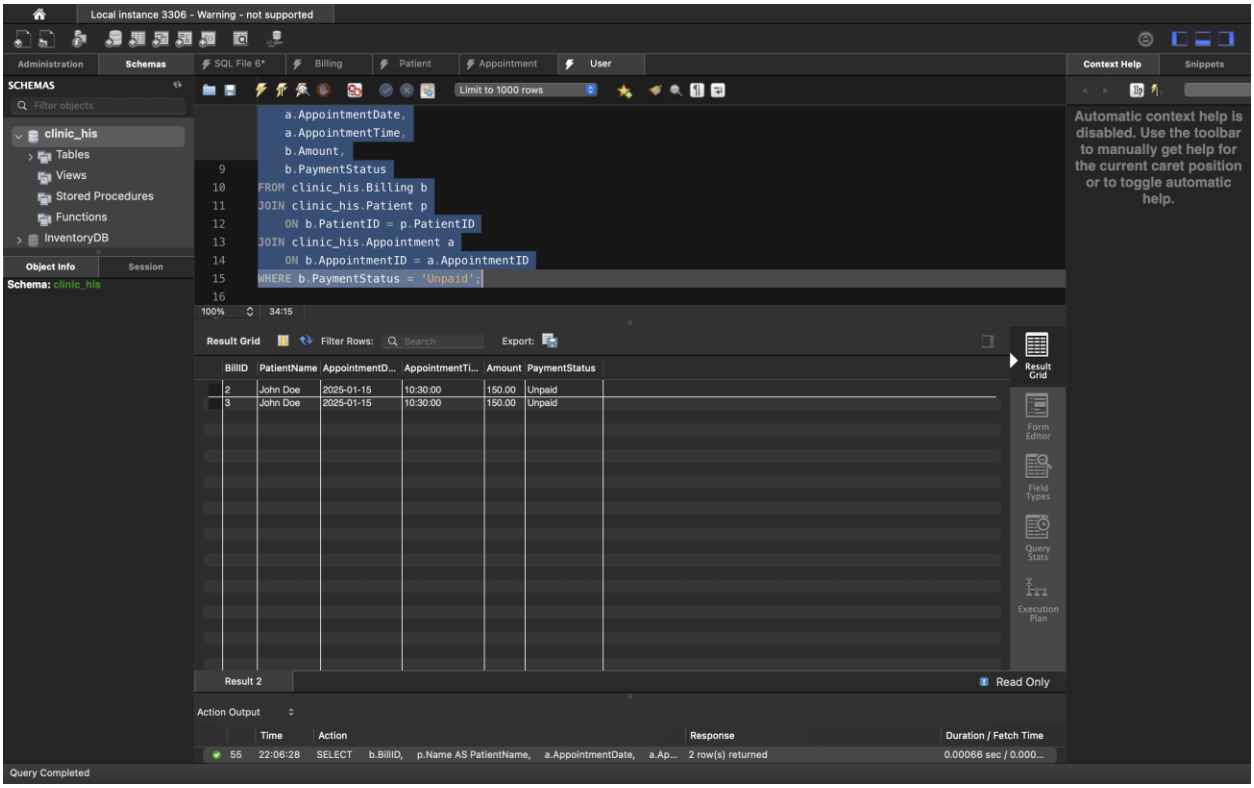
Test Case 5:



Test Case 6:


Local instance 3306 - Warning - not supported
SCHEMAS
clinic_his
Tables
Views
Stored Procedures
Functions
InventoryDB
-- Test Case 6: Update appointment status to Completed
UPDATE Appointment
SET Status = 'Completed'
WHERE AppointmentID = 1;
-- Verify the update worked
SELECT * FROM Appointment
WHERE AppointmentID = 1;
Result Grid
Appoin... PatientID Doctor AppointmentD... AppointmentTi... Status
1 1 Dr. Smith 2025-01-15 10:30:00 Completed
Appointment 14
Action Output
Time Action Response Duration / Fetch Time
52 22:01:40 SELECT * FROM Appointment WHERE AppointmentID = 1 LIMIT 0, 1000 1 row(s) returned 0.00045 sec / 0.000...
Query Completed

Test Case 7:


Local instance 3306 - Warning - not supported
SCHEMAS
clinic_his
Tables
Views
Stored Procedures
Functions
InventoryDB
a.AppointmentDate,
a.AppointmentTime,
b.Amount,
b.PaymentStatus
FROM clinic_his.Billing b
JOIN clinic_his.Patient p
ON b.PatientID = p.PatientID
JOIN clinic_his.Appointment a
ON b.AppointmentID = a.AppointmentID
WHERE b.PaymentStatus = 'Unpaid';
Result Grid
BillID PatientName AppointmentD... AppointmentTi... Amount PaymentStatus
2 John Doe 2025-01-15 10:30:00 150.00 Unpaid
3 John Doe 2025-01-15 10:30:00 150.00 Unpaid
Result 2
Read Only
Action Output
Time Action Response Duration / Fetch Time
55 22:06:28 SELECT b.BillID, p.Name AS PatientName, a.AppointmentDate, a.Ap... 2 row(s) returned 0.00066 sec / 0.000...
Query Completed

7. Existing Data Conversion

When a small clinic transitions from a paper-based or Excel-based system to the new Clinic-HIS database, existing patient, appointment, and billing information must be transferred into the new system. Data conversion ensures that the new system starts with accurate records and does not require staff to re-enter information manually.

7.1 Identify the Data Sources

Before conversion, the clinic must identify where the old data currently exists. Typical sources include:

- Paper files containing patient charts
- Excel spreadsheets containing appointments or billing
- Old scheduling notebooks
- Separate Word/Google Sheets patient lists
- Receipts or billing logs
- **7.2 Data to be Converted**

For this project, the main data categories to convert include:

1. **Patient Data**
    a. Patient name
    b. Date of birth
    c. Address
    d. Contact number
    e. Medical notes/history
2. **Appointment Data**
    a. Patient name (must match PatientID)
    b. Appointment dates and times
    c. Doctor assigned
    d. Appointment status (Scheduled, Completed, Cancelled, No-Show)
3. **Billing Data**
    a. Patient name (must match PatientID)
    b. AppointmentID
    c. Amount charged
    d. Payment status (Paid, Unpaid, Partial)
    e. Date issued

7.3 Verification After Conversion

Once data is added:

- Run SELECT queries to verify accuracy
- Check for missing or duplicated records
- Ensure relationships are correct (e.g., every appointment references a valid PatientID)
- Validate sample billing records

Example:
SELECT PatientID, Name, DOB FROM Patient;

SELECT a.AppointmentID, a.Doctor, a.AppointmentDate, p.Name

FROM Appointment a

JOIN Patient p ON a.PatientID = p.PatientID;

8. Planning User Training

The primary goals of the user training program are:

- To ensure users understand how to navigate the system and perform their assigned tasks confidently
- To minimize errors caused by unfamiliarity with digital workflows
- To ensure proper data entry practices for maintaining accurate patient records
- To train users on privacy and HIPAA-compliant behavior
- To prepare staff to troubleshoot minor issues independently
- To build user confidence in transitioning from paper or spreadsheet systems to the new HIS

8.1 Training Objectives

The primary goals of the user training program are:

- To ensure users understand how to navigate the system and perform their assigned tasks confidently
- To minimize errors caused by unfamiliarity with digital workflows
- To ensure proper data entry practices for maintaining accurate patient records
- To train users on privacy and HIPAA-compliant behavior

- To prepare staff to troubleshoot minor issues independently
- To build user confidence in transitioning from paper or spreadsheet systems to the new HIS

8.2 Training Schedule

Day	User Role	Training Topics
Day 1	Admin & Receptionists	Login, patient registration, appointment scheduling, searching records
Day 2	Billing Staff	Invoice creation, payments, financial reports
Day 3	Clinical staff	Viewing patient charts, appointment details, HIPAA guidelines
Ongoing	All users	Refresher sessions, troubleshooting guidance, Q&A

8.3 Evaluation of Training

Training effectiveness will be measured using:

- **Hands-on tests** (users perform tasks without guidance)
- **Short surveys** assessing user confidence
- **Monitoring early system usage** to identify needed retraining
- **Feedback forms** to identify unclear instructions or workflow issues

8.4 Post-Training Support

To ensure continued success:

- **Help Desk Email:** A dedicated support contact for reporting issues
- **Weekly check-ins** during first month of deployment
- **Updated documentation** whenever system features change
- **Retraining sessions** offered quarterly if needed