GROUP 3

Project Deliverable_5

CS425-02: Database Organization

HEALTHCARE MANAGEMENT SYSTEM

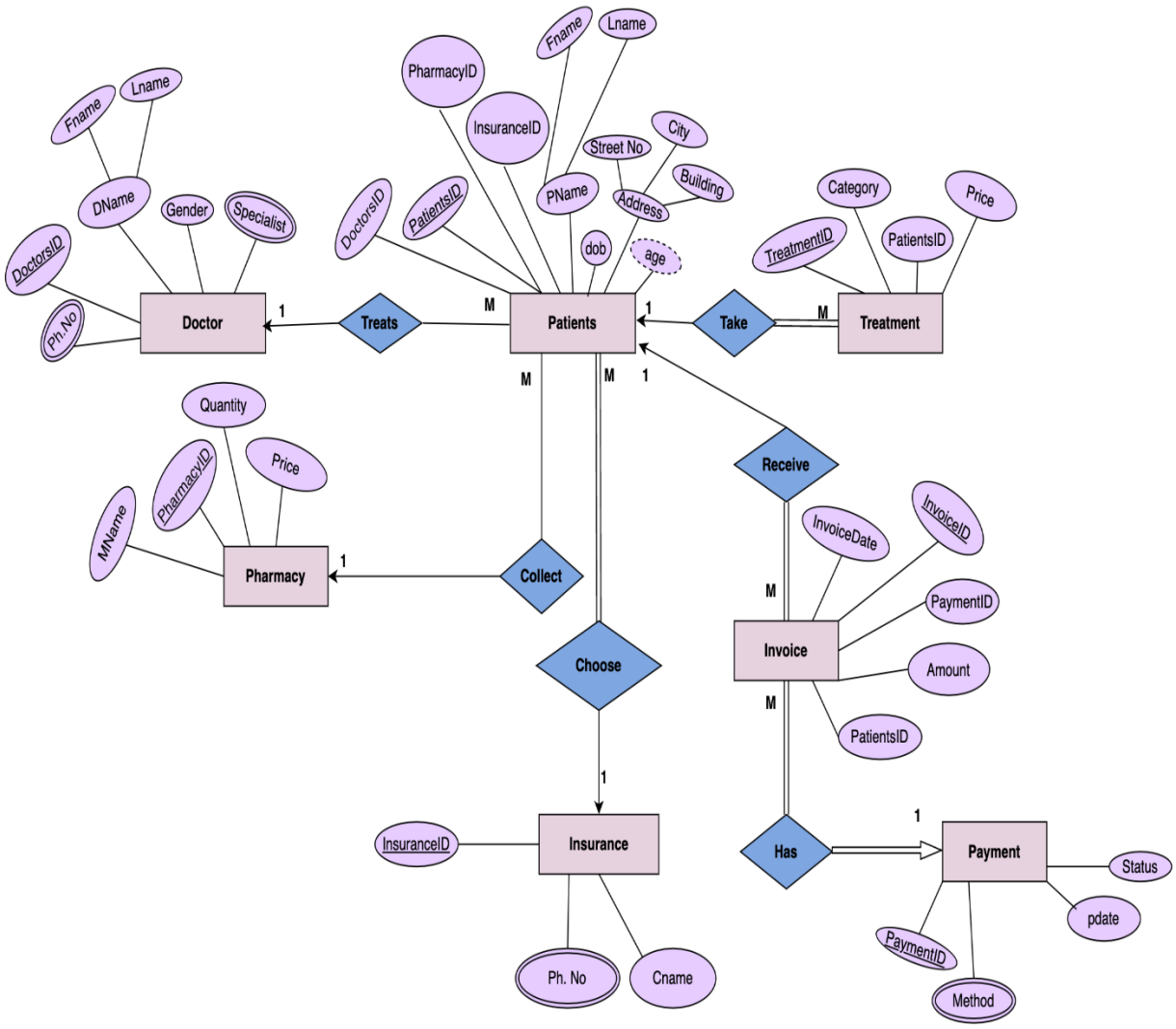
Group Members Details

Member name	CWID	Hawk Id
Akshitha Bedre Shivakumar	A20544641	abedreshivakumar@hawk.iit.edu
Jen Kai Wang	A20576222	jwang300@hawk.iit.edu
Ramya Venkatesh	A20546964	rvenkatesh1@hawk.iit.edu
Waseem Pasha Mohammad	A20551352	wmohammad@hawk.iit.edu

INTRODUCTION:

The effective administration of data is essential to the healthcare sector. A strong relational database that can store, organize, and retrieve vital patient data, medical records, and administrative data is the foundation of a strong Healthcare Management System (HMS). Consider the patient, physician, and billing information as discrete parts inside the library, each containing specific details such as names, dates, diagnoses, and expenses. These characteristics store certain bits of information, and this is where relationships come into play. Consider foreign keys as links between physicians and patients, diagnoses and patients, bills, and doctors. You can quickly browse the data on this linked web to learn more about a particular patient's medical. But creating this information center needs careful preparation. The initial step is to identify the key players, which are the patients, doctors, and treatment. The next step is to specify the information you wish to record for each, such as a patient's allergies or a doctor's specializations. After that, it's time to create those vital connections that guarantee consistent data and easy retrieval. Robust security measures are vital to safeguard it, adhering to strict regulations. Through comprehension of these fundamental ideas and customization to your requirements, you may construct a relational database that enables your HMS to efficiently handle medical data.

ENTITY RELATIONSHIP DIAGRAM:



ABBREVATIONS OF THE ABOVE ENTITY RELATIONSHIP DIAGRAM:

DOCTOR:
DoctorsID: Doctor's ID
Dname: Doctor Name
Fname: First name
Lname: Last name
Gender: Doctor Gender
Specialist: Doctor Specialization
Ph.no: Doctor Phone Number

PATIENT:
PatientsID: Patient's ID
DoctorsID: Doctor's ID
PharmacyID: Pharmacy ID
InsuranceID: Insurance ID
Pname: Patient's Name
Fname: Full name
Lname: Last name
DOB: Patient's Date of Birth
Age: Patient's age
Address: Patient's Address
Street No: Patient's Street Number
City: Patient's City
Building: Patient's Building Number

TREATMENT:
TreatmentID: Treatment ID
PatientsID: Patient's ID
Category: Treatment Category
Price: Price of the Treatment

PHARMACY:
PharmacyID: Pharmacy ID
Mname: Medicine Name
Quantity: Medicine Quantity
Price: Medicine Price

INSURANCE:
InsuranceID: INSURANCE ID
CName: Insurance Company Name
Ph. No: Insurance Phone Number

INVOICE:
InvoiceID: Invoice ID
PaymentID: Payment ID
PatientsID: Patients ID
InvoiceDate: Date of the invoice generated.
Amount: Patient's amount for the Treatment

PAYMENT:
PaymentID: Payment ID
Status: Payment Status
Method: Payment Method
Pdate: Payment Date

SCHEMA IN ER DIAGRAM:

- Doctor (**DoctorsID**, Dname [fname, lname], Ph.No, Gender, Specialist)
- Patient (**PatientsID**, DoctorsID, PharmacyID, InsuranceID, Pname [fname, lname], dob, Age, Address [Street NO, City, Building]
- Treatment (**TreatmentID**, Price, PatientsID, Category)
- Pharmacy (**PharmacyID**, Price, Quantity, MName)
- Insurance **(InsuranceID**, Ph. No, Cname)
- Invoice (**InvoiceID**, PaymentID, PatientsID, InvoiceDate, Amount)
- Payment (**PaymentID**, Status, Method, pdate)

BUSINESS RULES:

- Doctor-Patient Relationship:

  One Doctor can treat many Patients.

  One Patient must have one primary care Doctor.
- Patient-pharmacy Relationship:

  Many Patients can collect medicines from a Pharmacy.

  One Pharmacy can serve many Patients.
- Patient-Invoice Relationship:

  One Patient can receive many Invoices for services rendered.

  One Invoice is associated with one Patient.

- Patient-Treatment Relationship:

  One Patient can take many Treatments.

  Multiple Treatments can be given to a Patient.

- Invoice-Payment Relationship:

  One Payment has multiple Invoices.

  Invoices can be paid by one Payment.

- Patient-Insurance Relationship:

  Many Patients can choose one primary Insurance Provider.

  One Insurance Provider can cover many Patients.

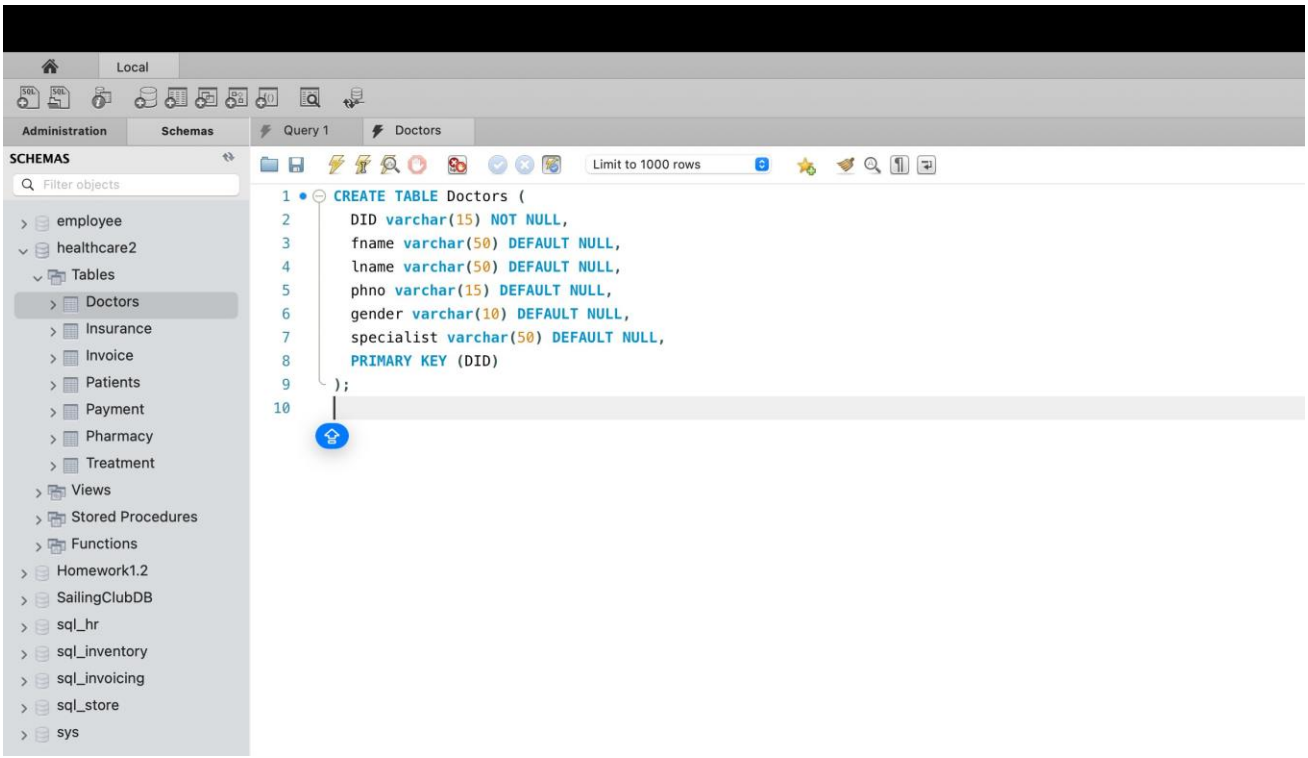
Initial Requirements:

Please make sure to fulfill the below requirements mentioned in the requirements.txt file before running the healthcare.py file.

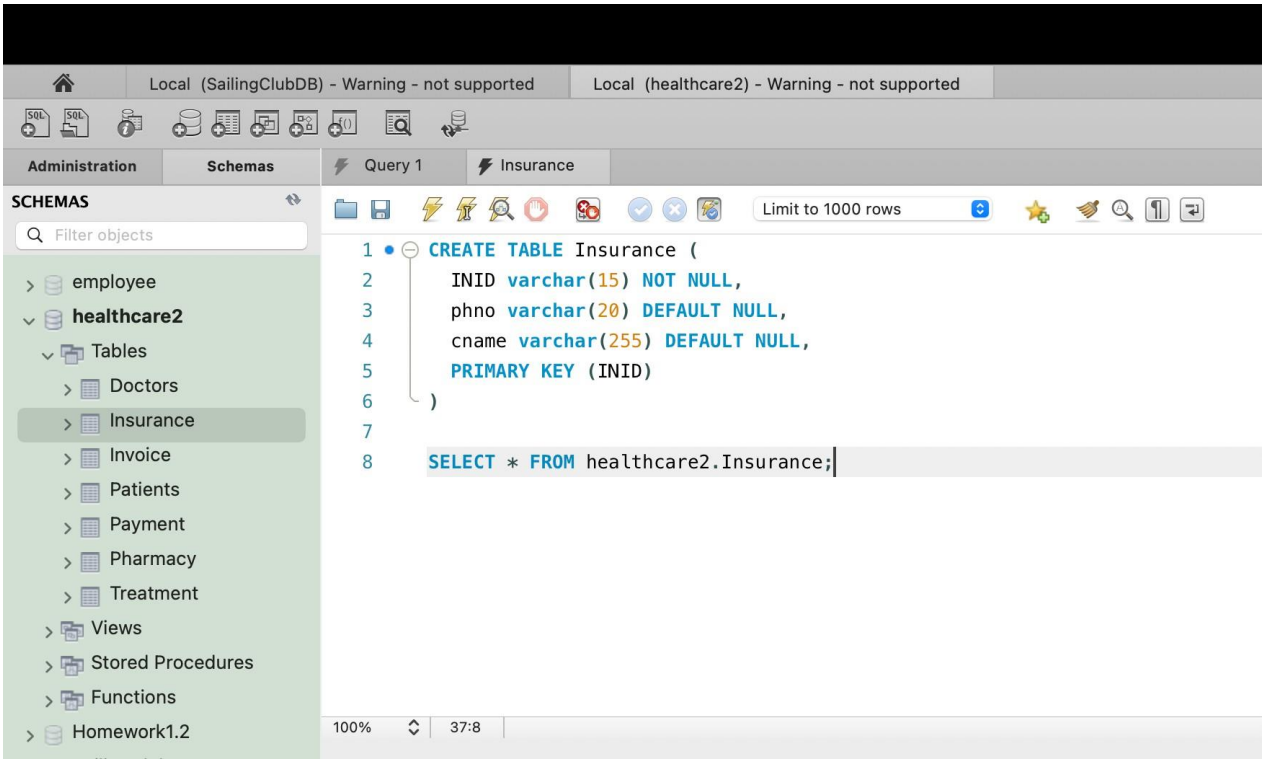
(user = 'root',
host =
'localhost',port
= 3306,
password = 'pass*******')

TABLES CREATION:

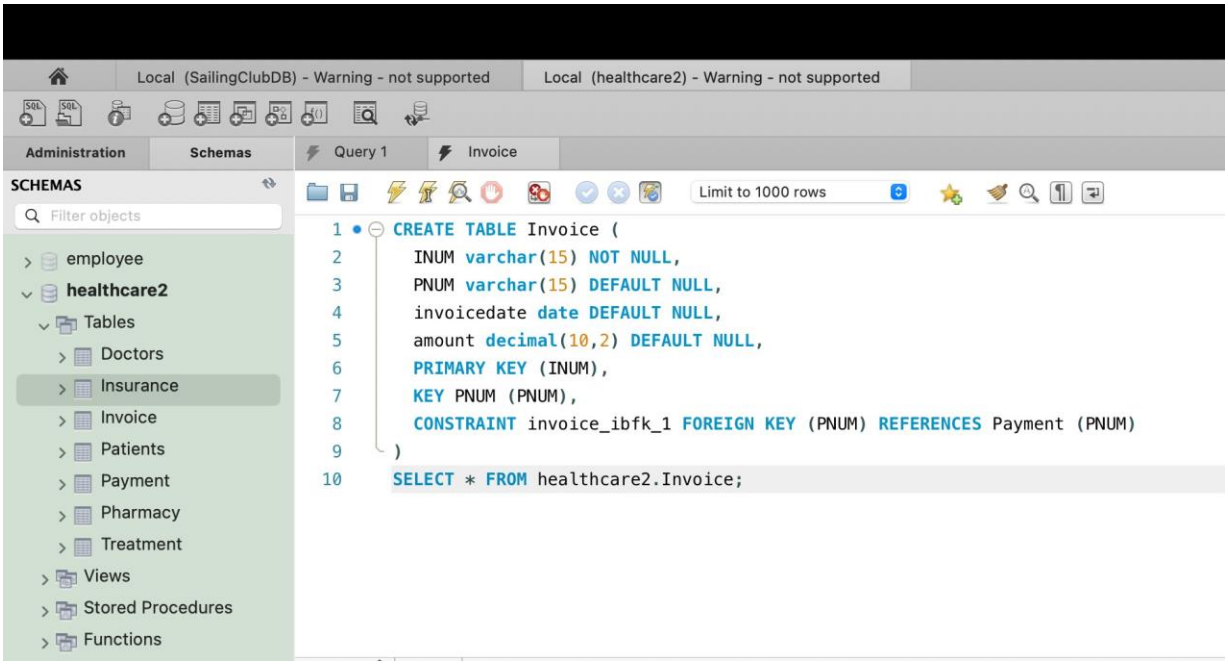
- **Create Doctor Table**



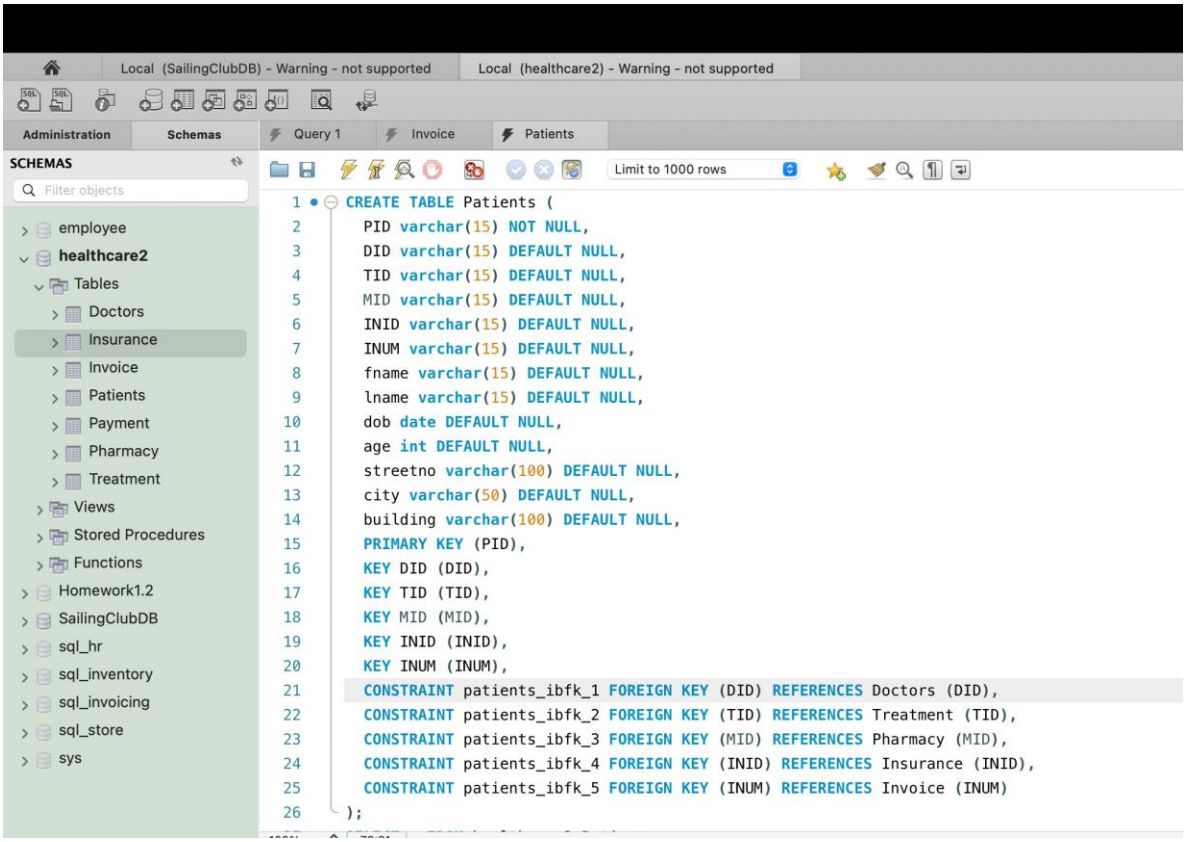
- **Create Insurance table**



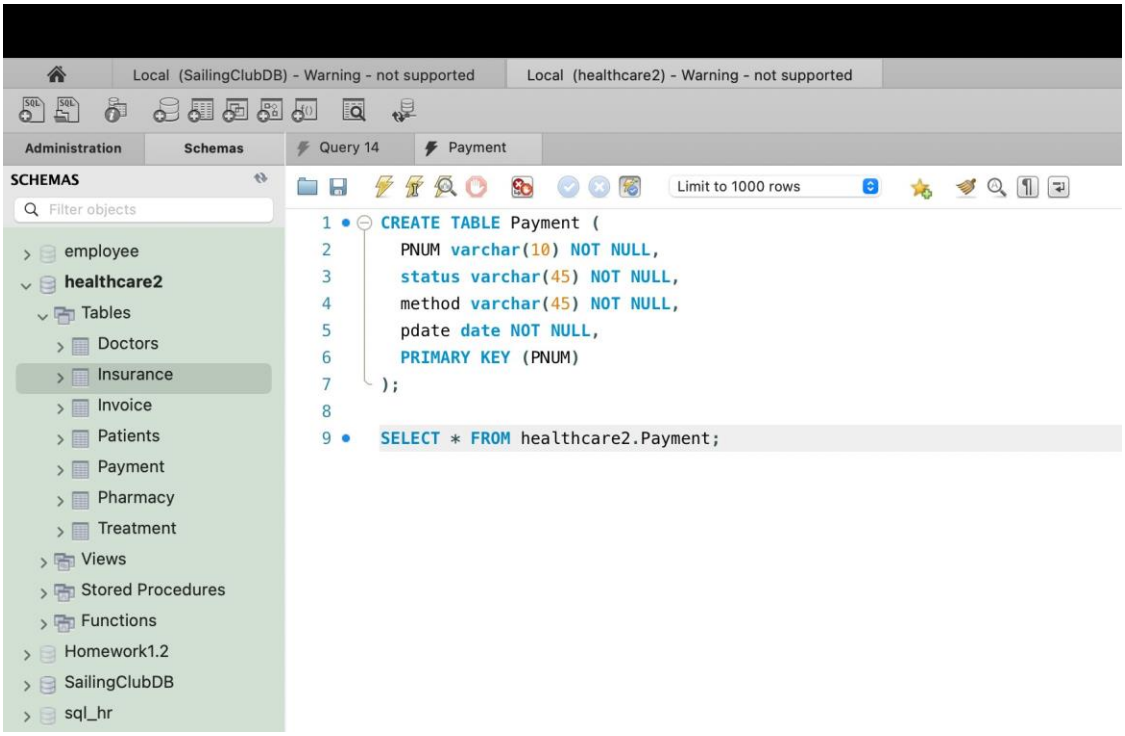
- **Create Invoice table**



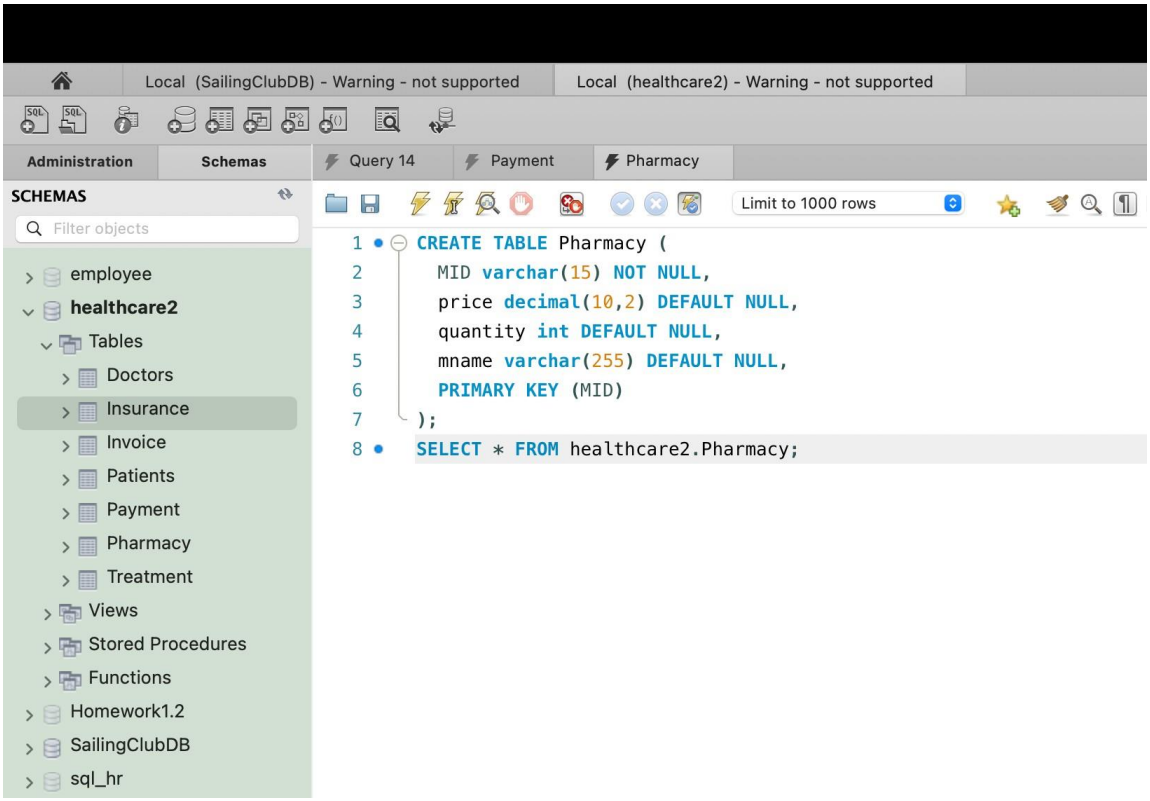
●Create Patients table



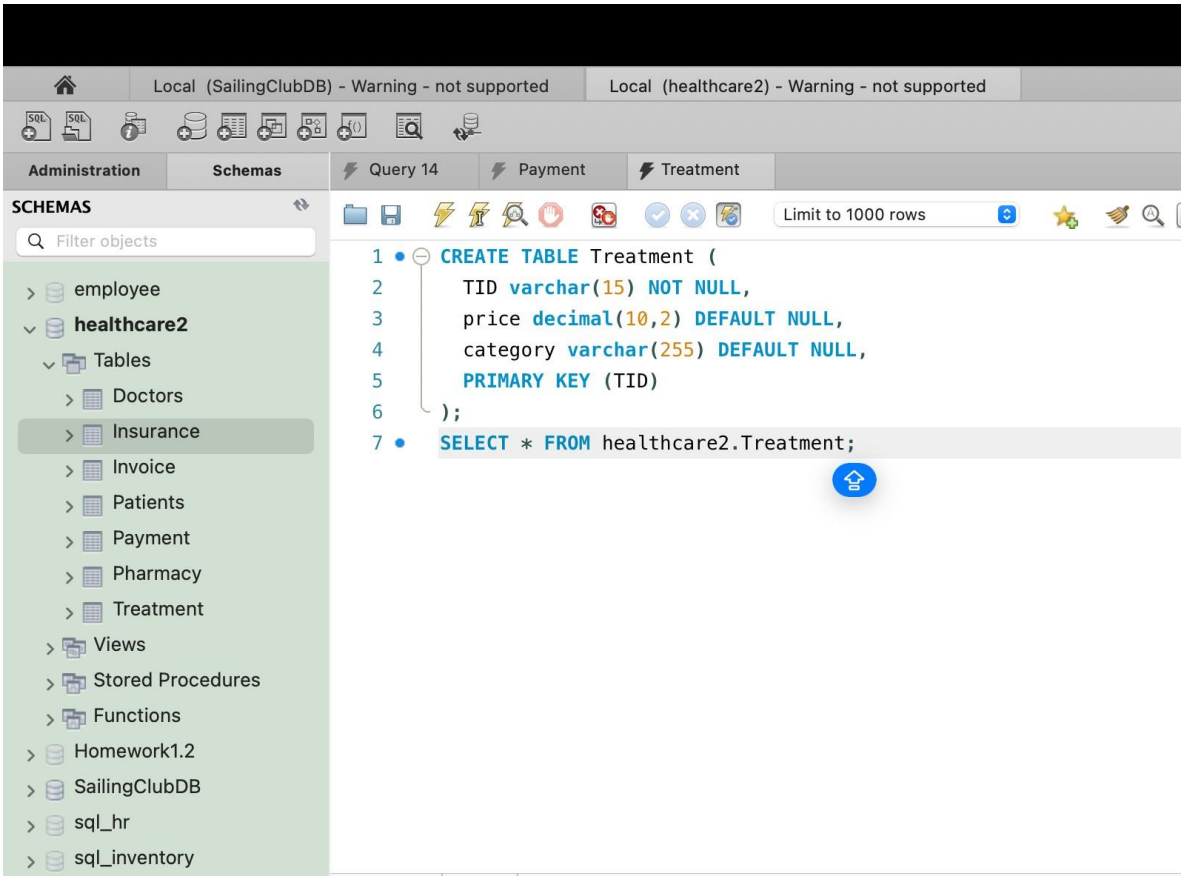
- **Create Payment table**



- **Create Pharmacy table**

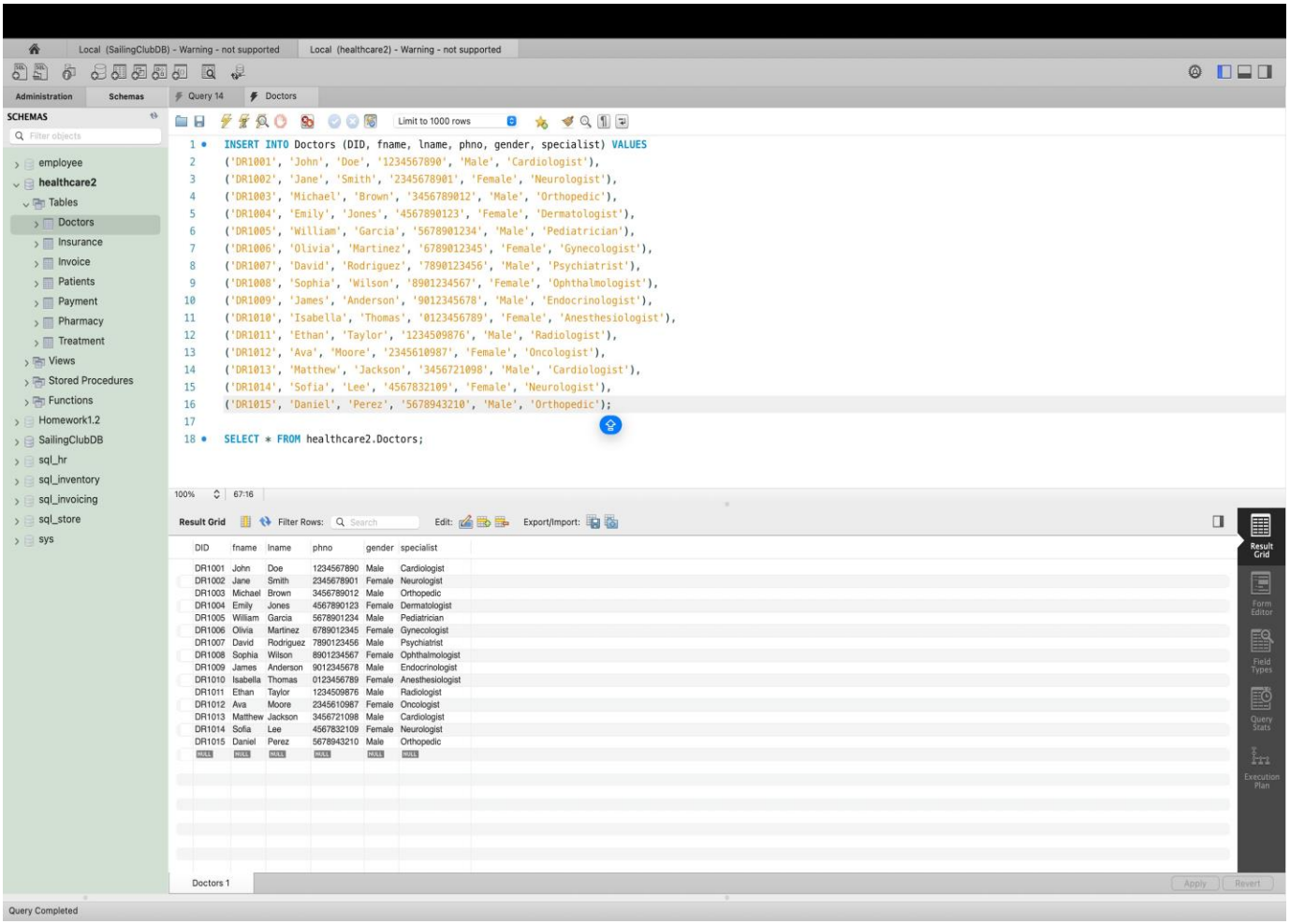


- **Create Treatment table**

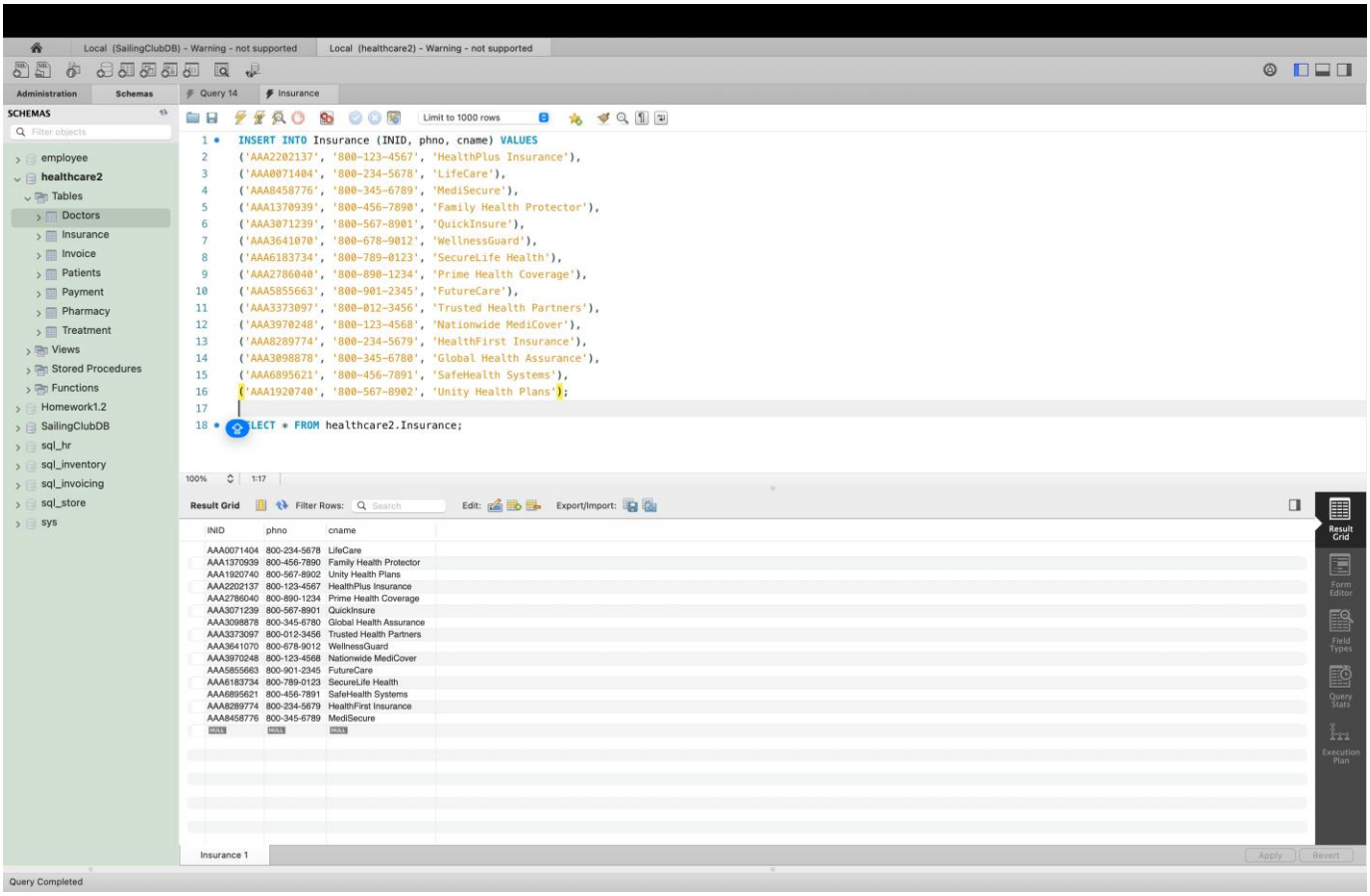


DATA INSERTION FOR THE ABOVE TABLES:

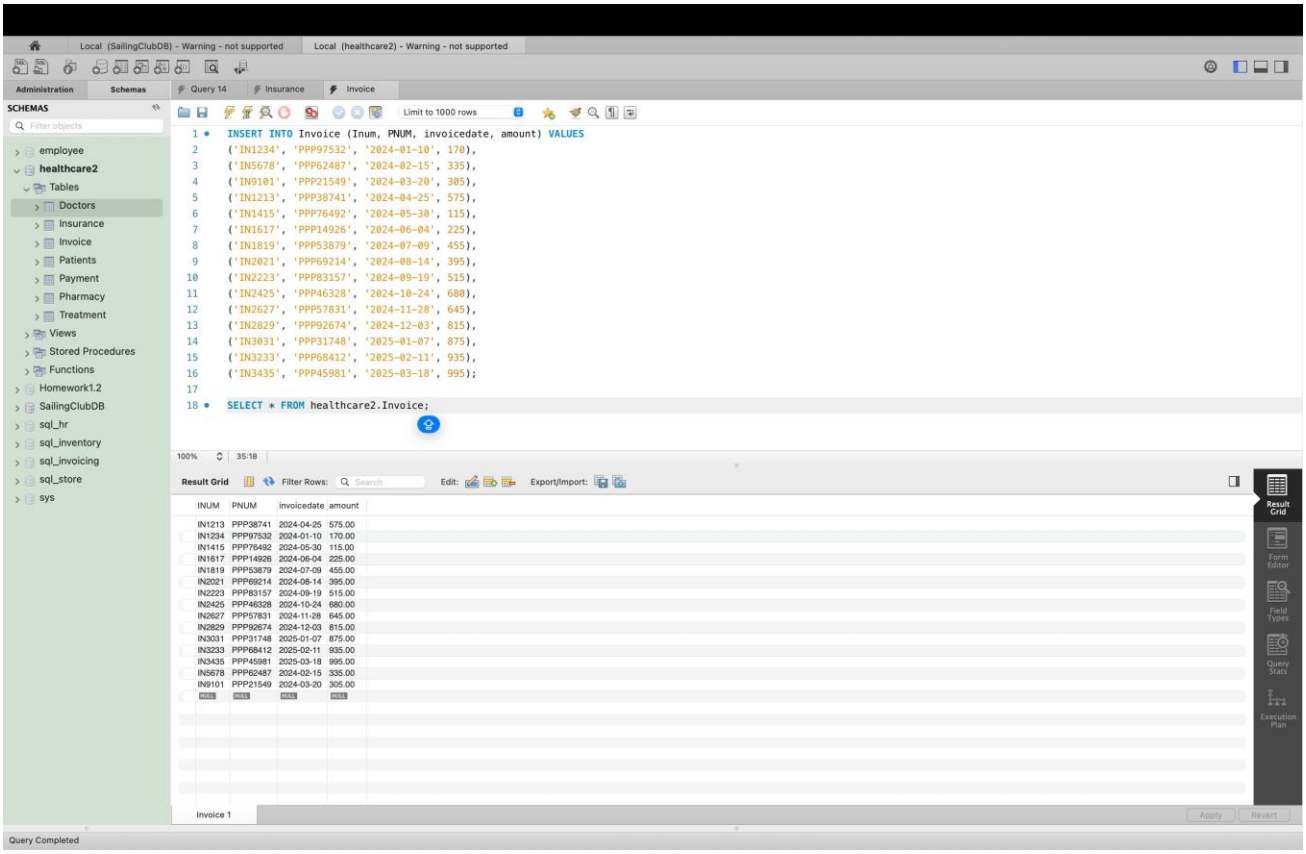
- **DATA INSERTION INTO DOCTORS TABLE**



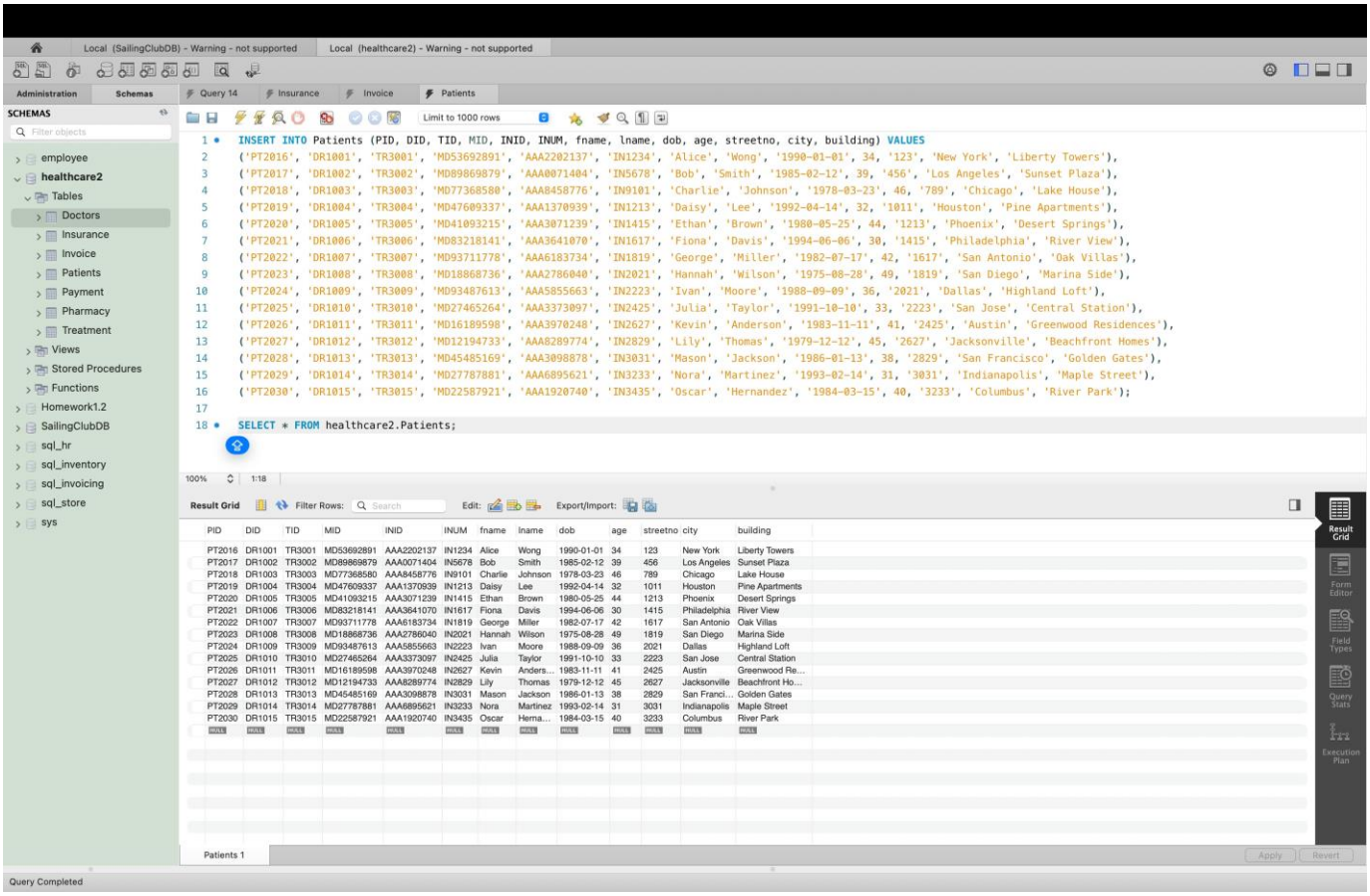
• DATA INSERTION INTO INSURANCE TABLE



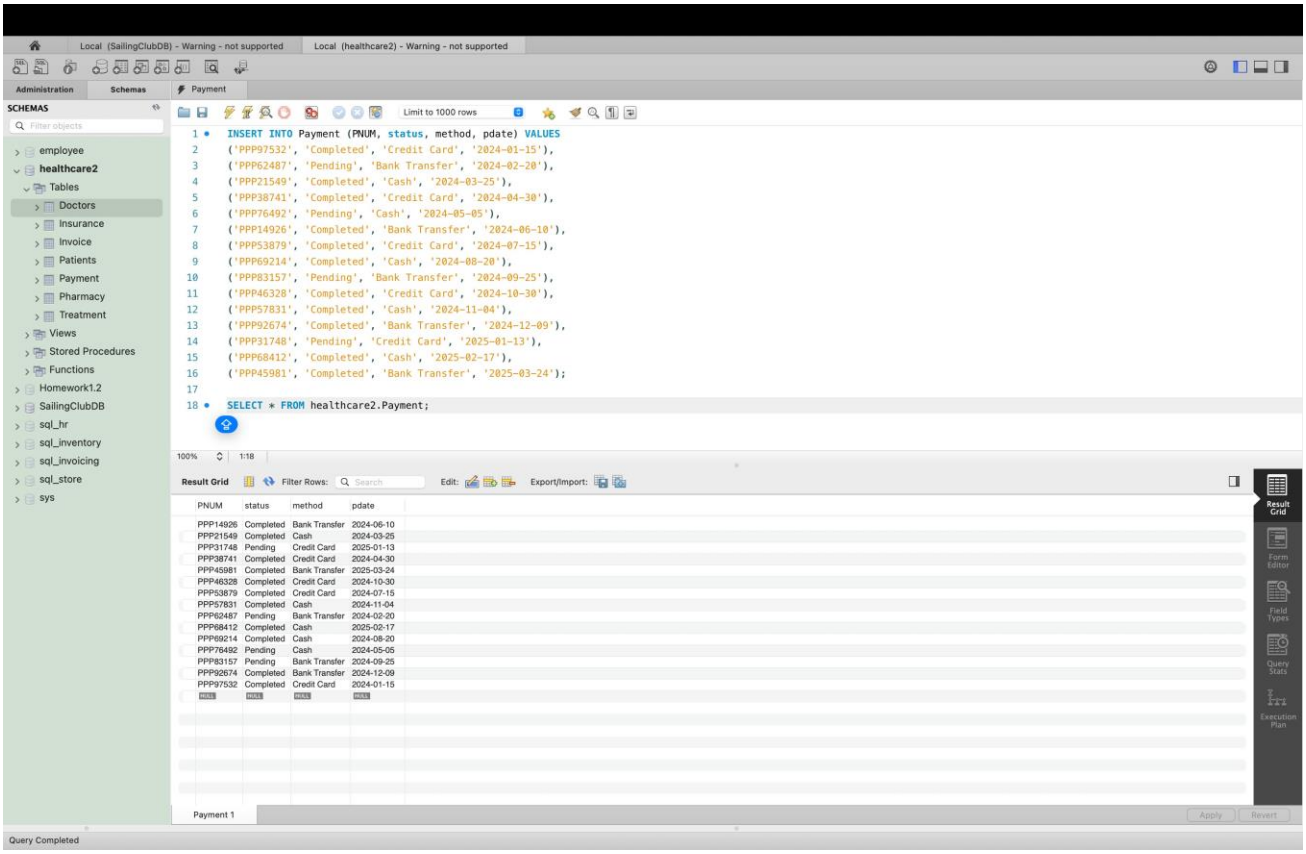
●DATA INSERTION INTO INVOICE TABLE



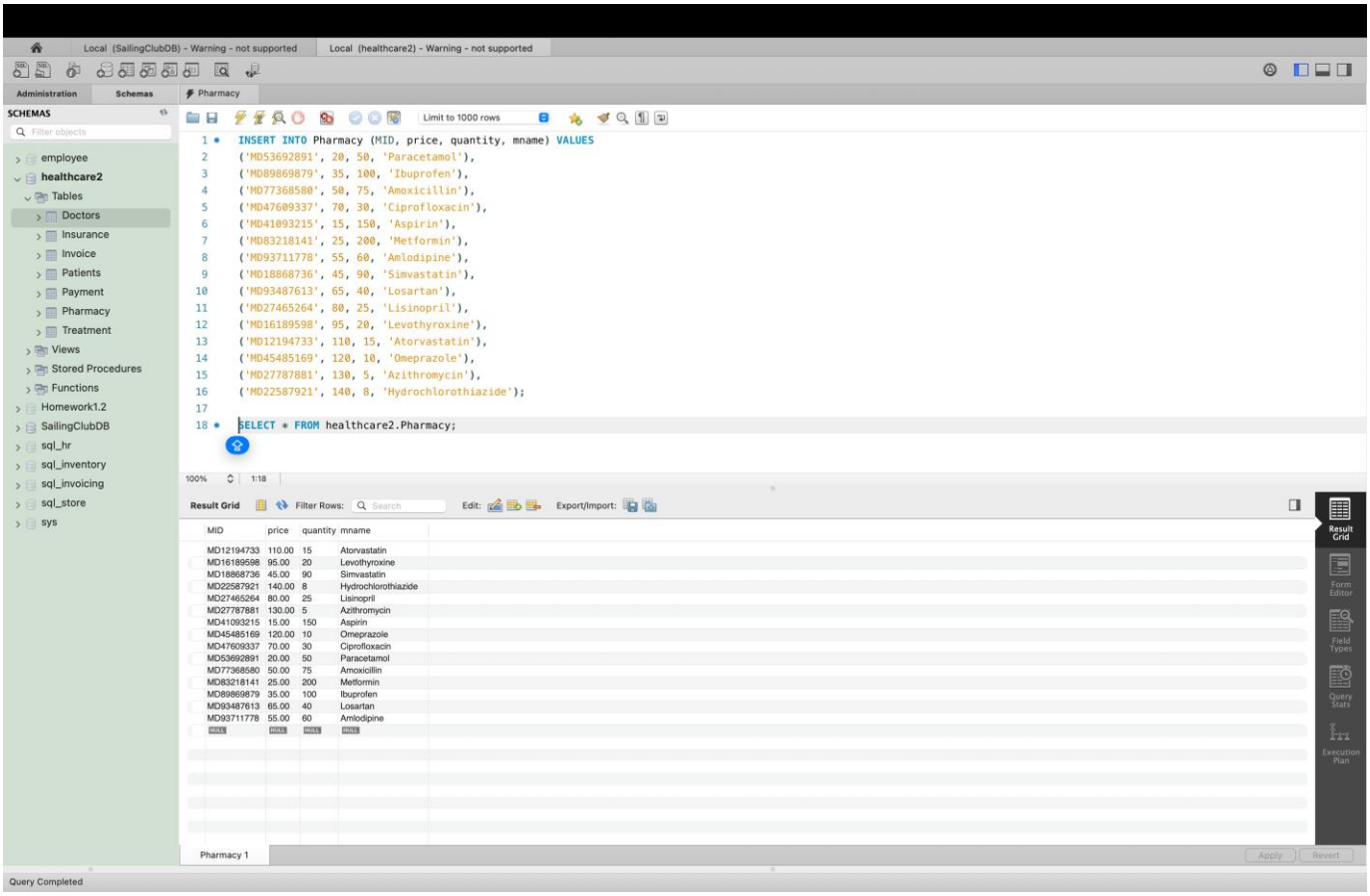
• DATA INSERTION INTO PATIENTS TABLE



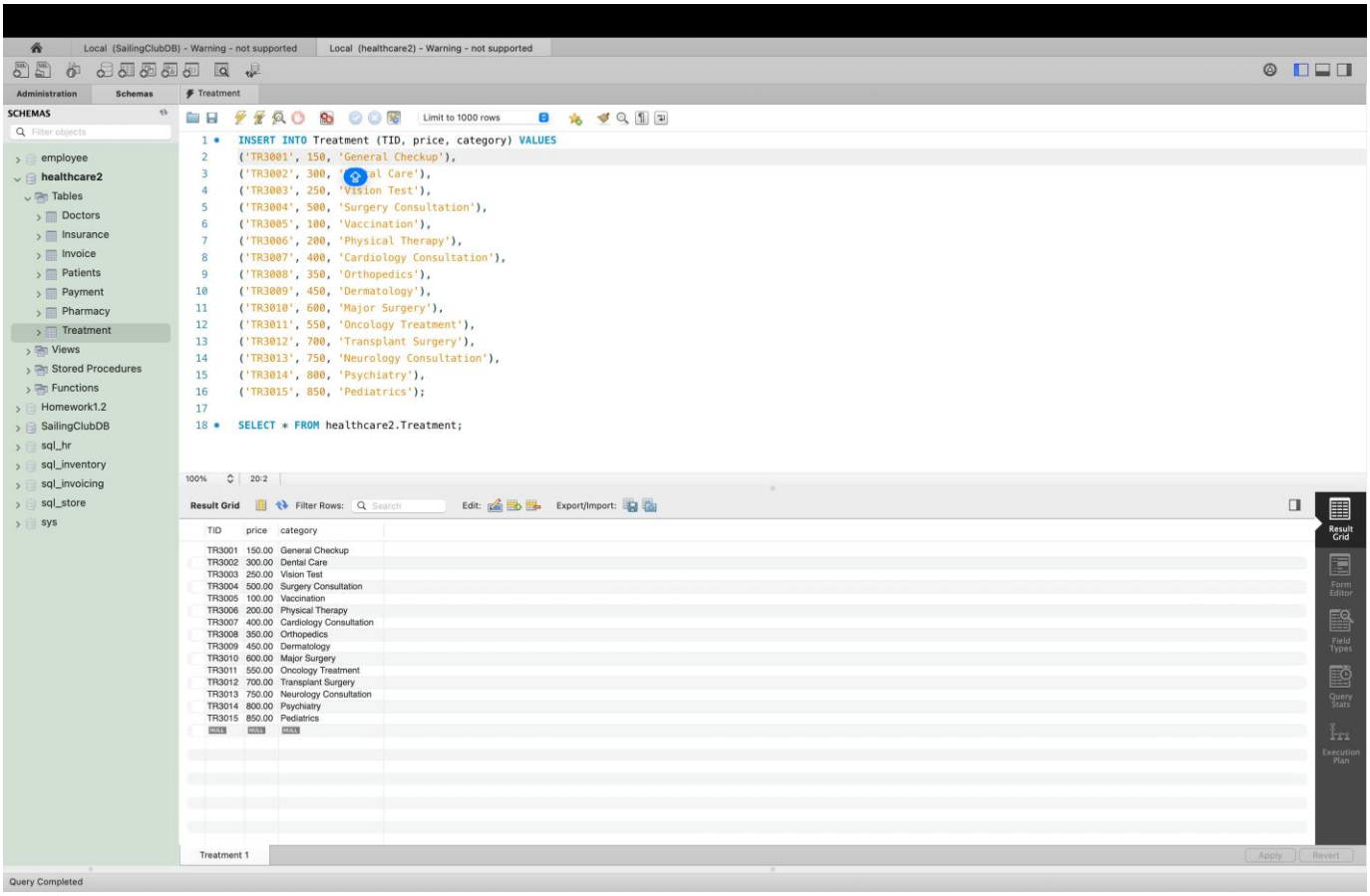
●DATA INSERTION INTO PAYMENTS TABLE



• DATA INSERTION INTO PHARMACY TABLE



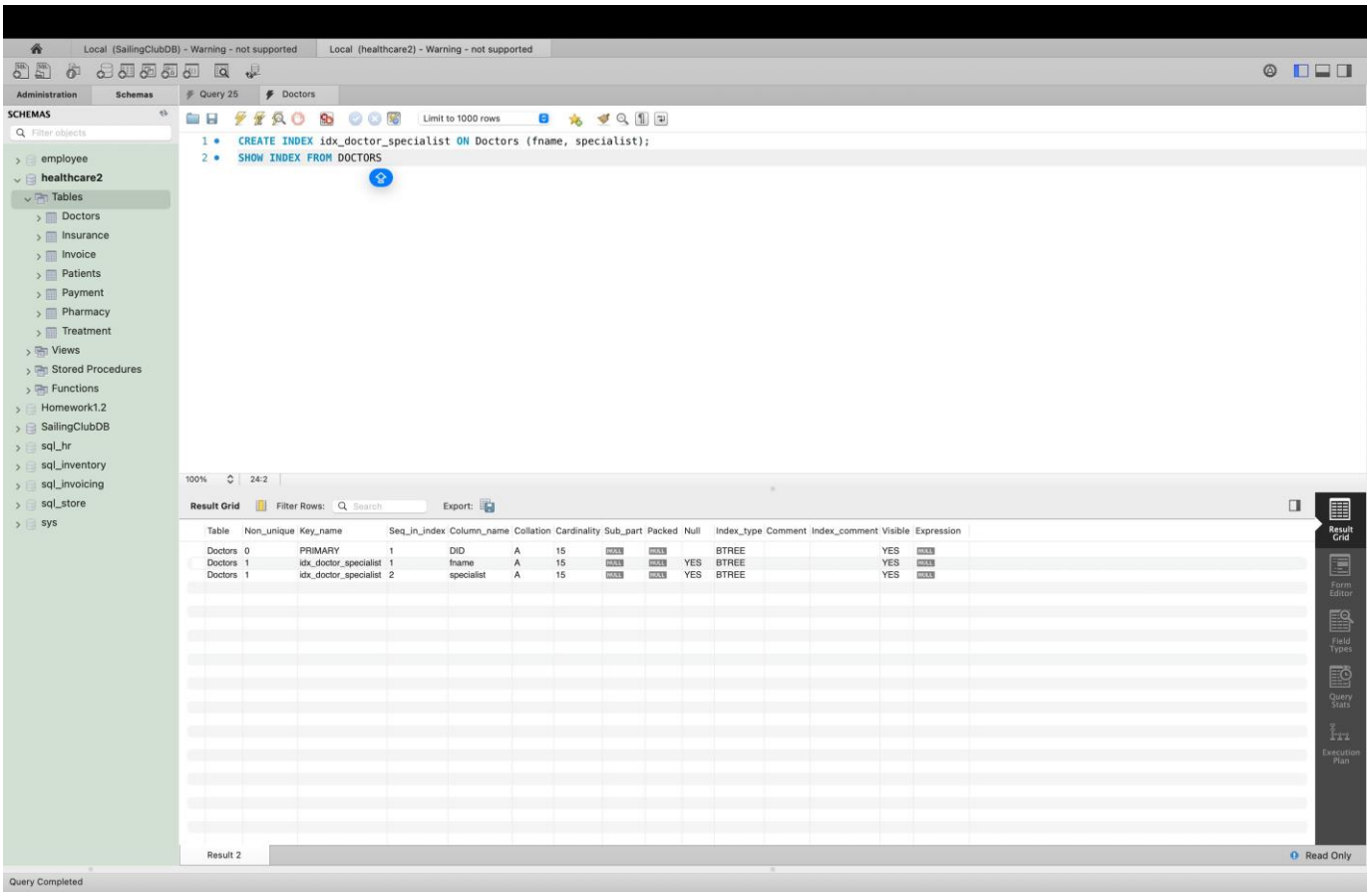
●DATA INSERTION INTO TREATMENT TABLE



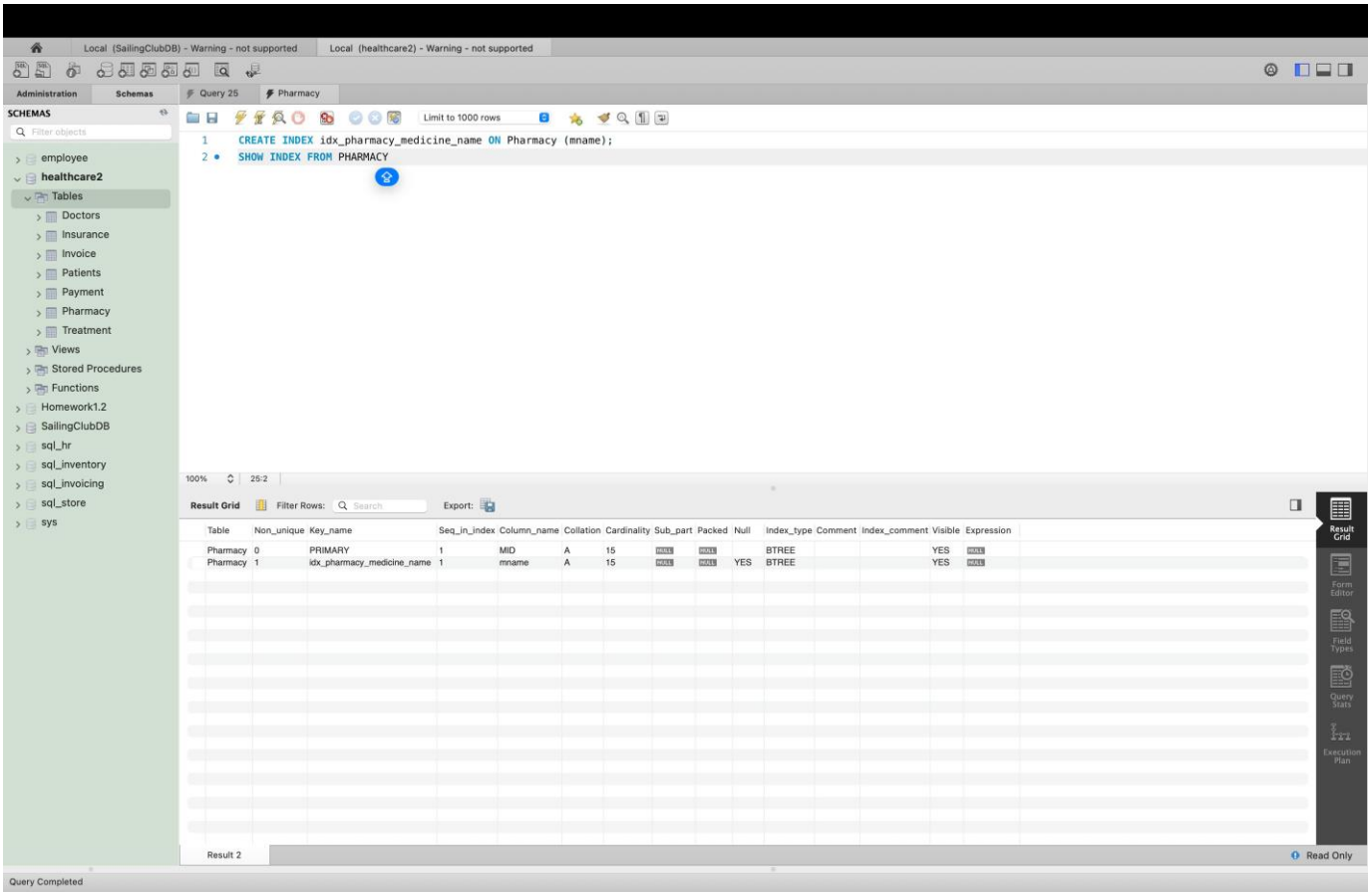
);

INDEX CREATION

CREATE INDEX idx_doctor_specialist ON Doctors (fname, specialist);

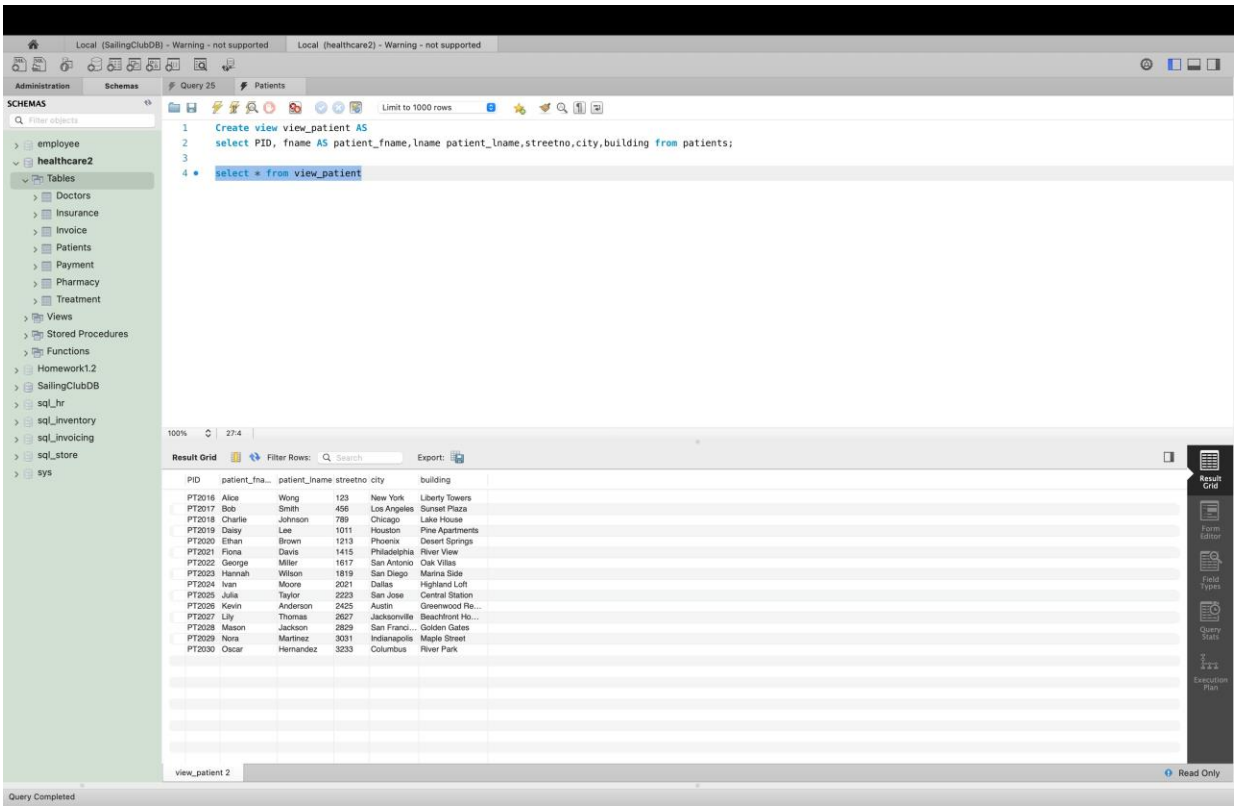


CREATE INDEX idx_pharmacy_medicine_name ON Pharmacy (mname);



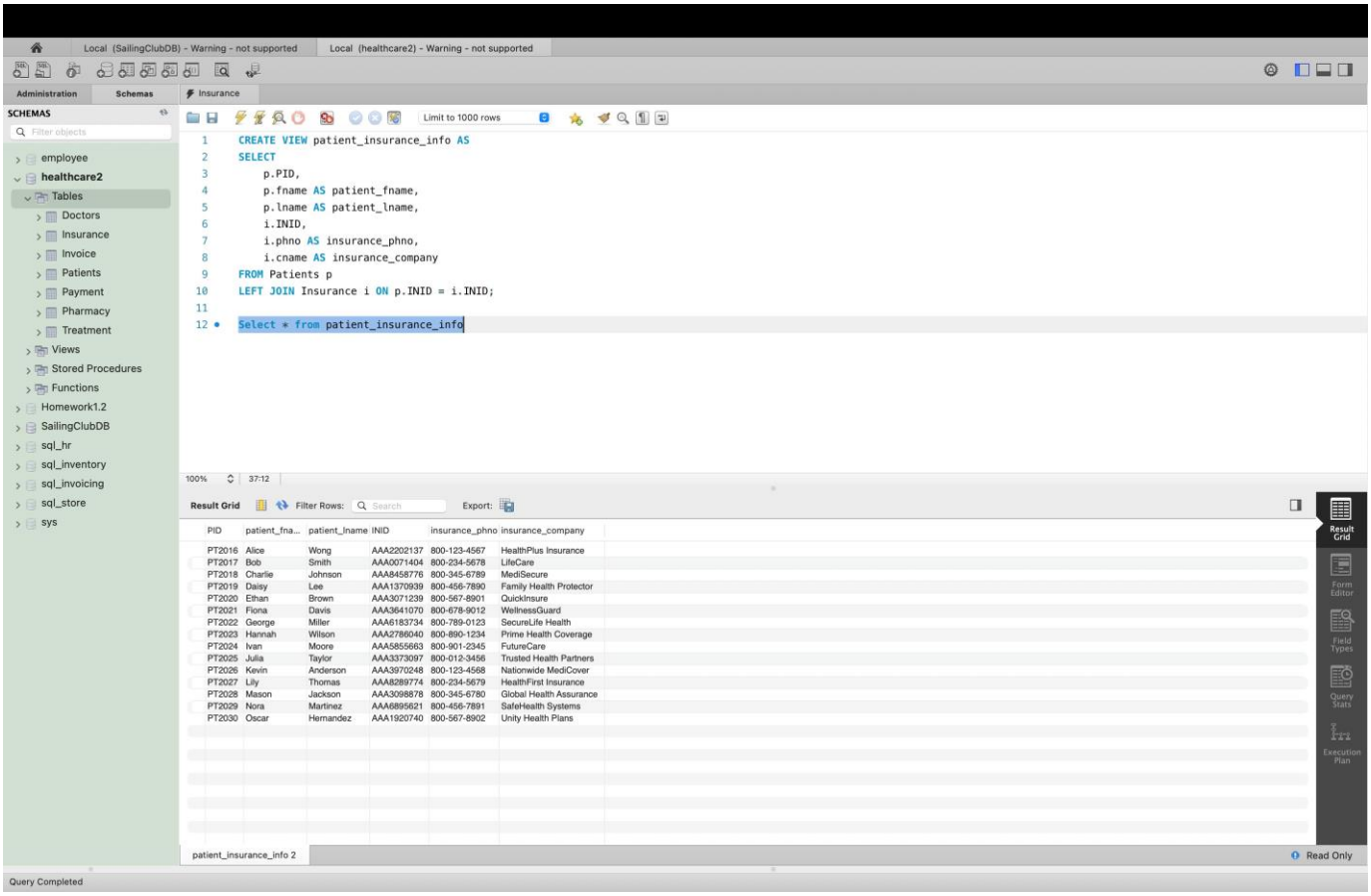
VIEWS CREATION:

Create view view_patient AS
select PID, fname AS patient_fname,lname patient_lname,streetno,city,building from patients;



● CREATE VIEW

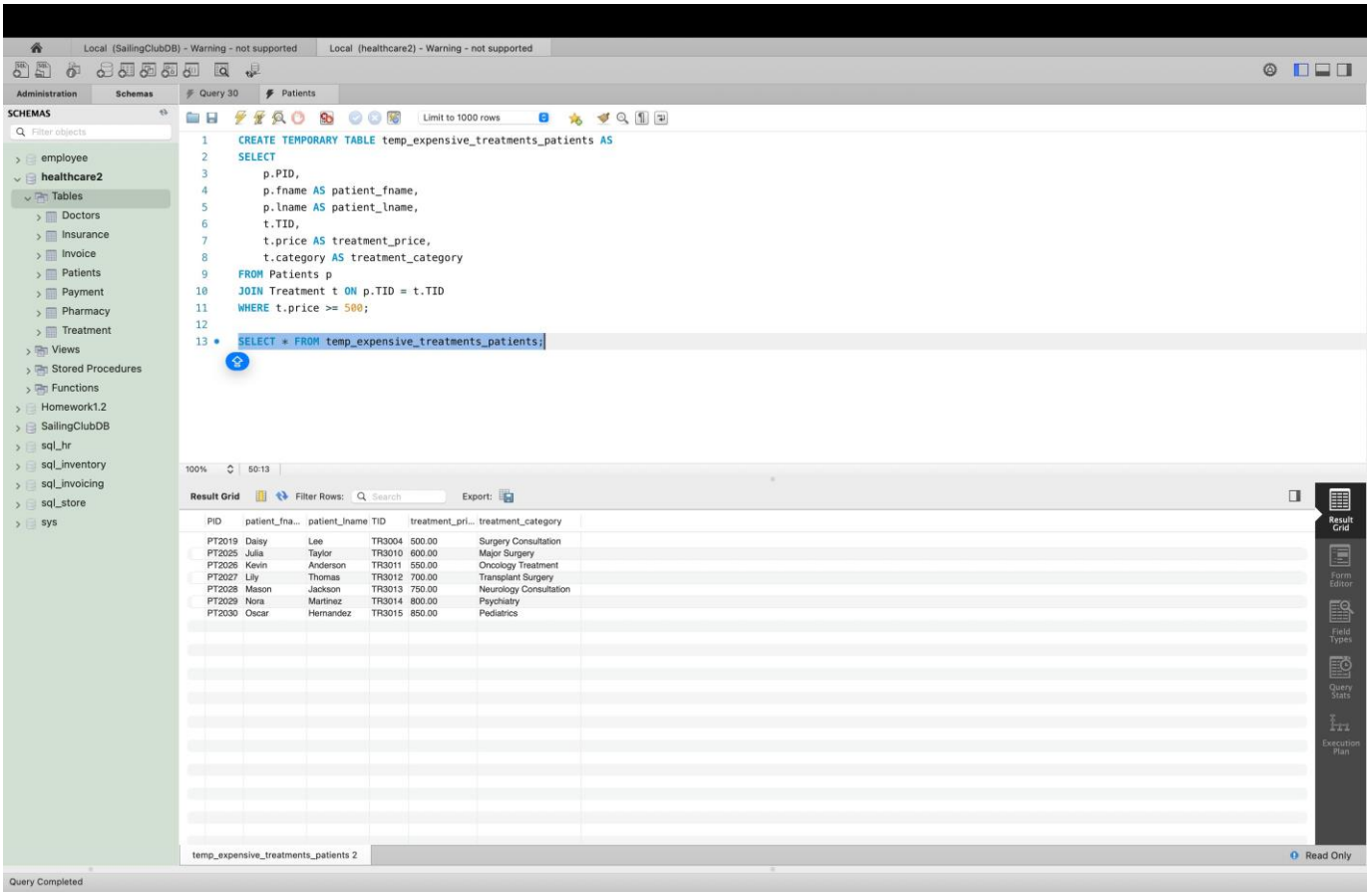
patient_insurance_info ASSELECT
p.PID,
p.fname AS patient_fname,
p.lname AS patient_lname,
i.INID,
i.phno AS insurance_phno,
i.cname AS insurance_company
FROM Patients p
LEFT JOIN Insurance i ON p.INID = i.INID;



TEMPORARY TABLES CREATION:

- **CREATE TEMPORARY TABLE**

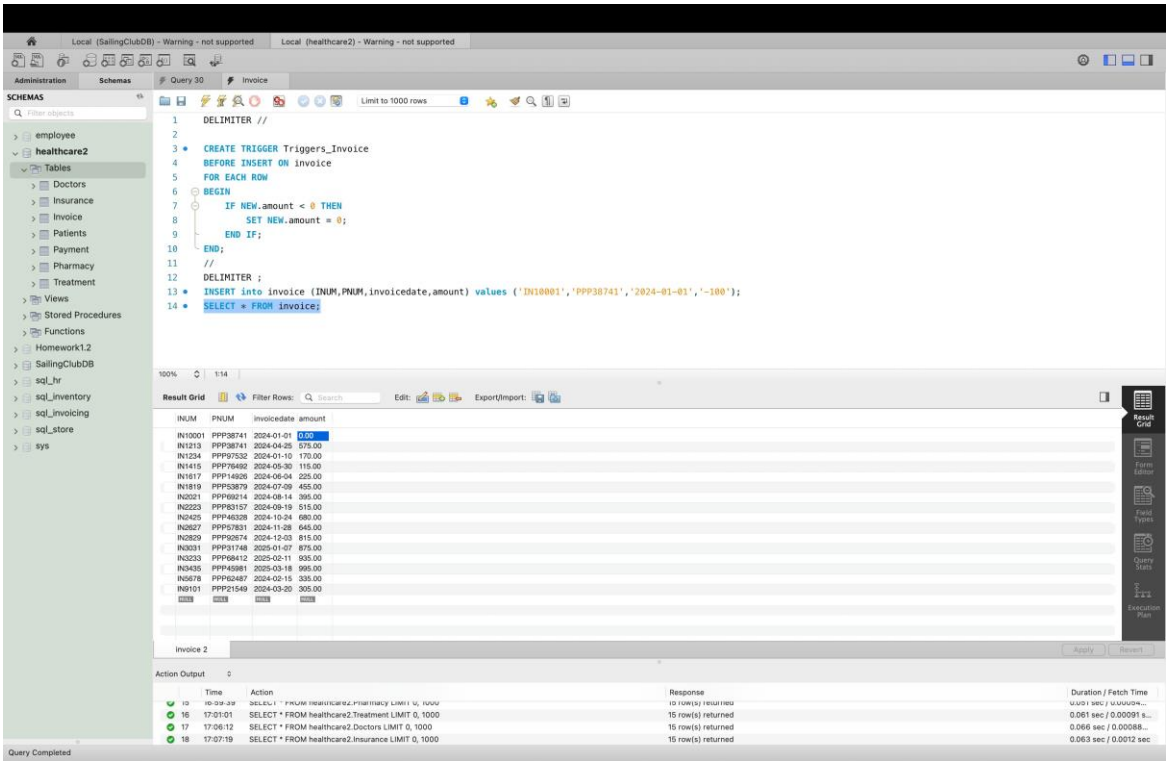
temp_expensive_treatments_patients ASSELECT
p.PID,
p.fname AS patient_fname,
p.lname AS patient_lname,
t.TID,
t.price AS treatment_price,
t.category AS treatment_category
FROM Patients p
JOIN Treatment t ON p.TID =
t.TIDWHERE t.price >= 500;



TRIGGERS CREATION:

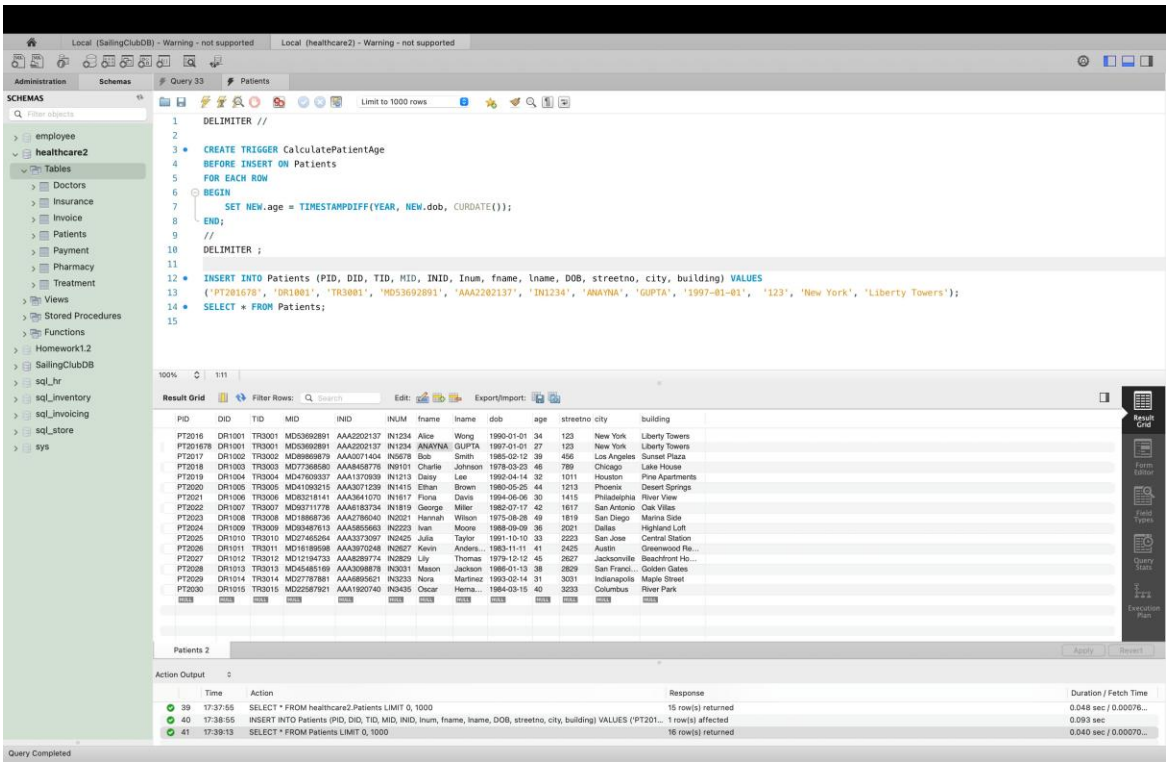
●Trigger to set amount as 0 if the value is less than 0.

```
DELIMITER //
CREATE TRIGGER
Triggers_InvoiceBEFORE
INSERT ON invoice
FOR EACH
ROWBEGIN
   IF NEW.amount < 0
      THENSET
      NEW.amount = 0;
   END
IF;
END;
//
DELIMITE
R ;

INSERT into invoice (INUM,PNUM,invoicedate,amount)
values('IN10001','PPP38741','2024-01-01','-100');
 SELECT * FROM invoice;
```



●Trigger to calculate the age from date of birth.

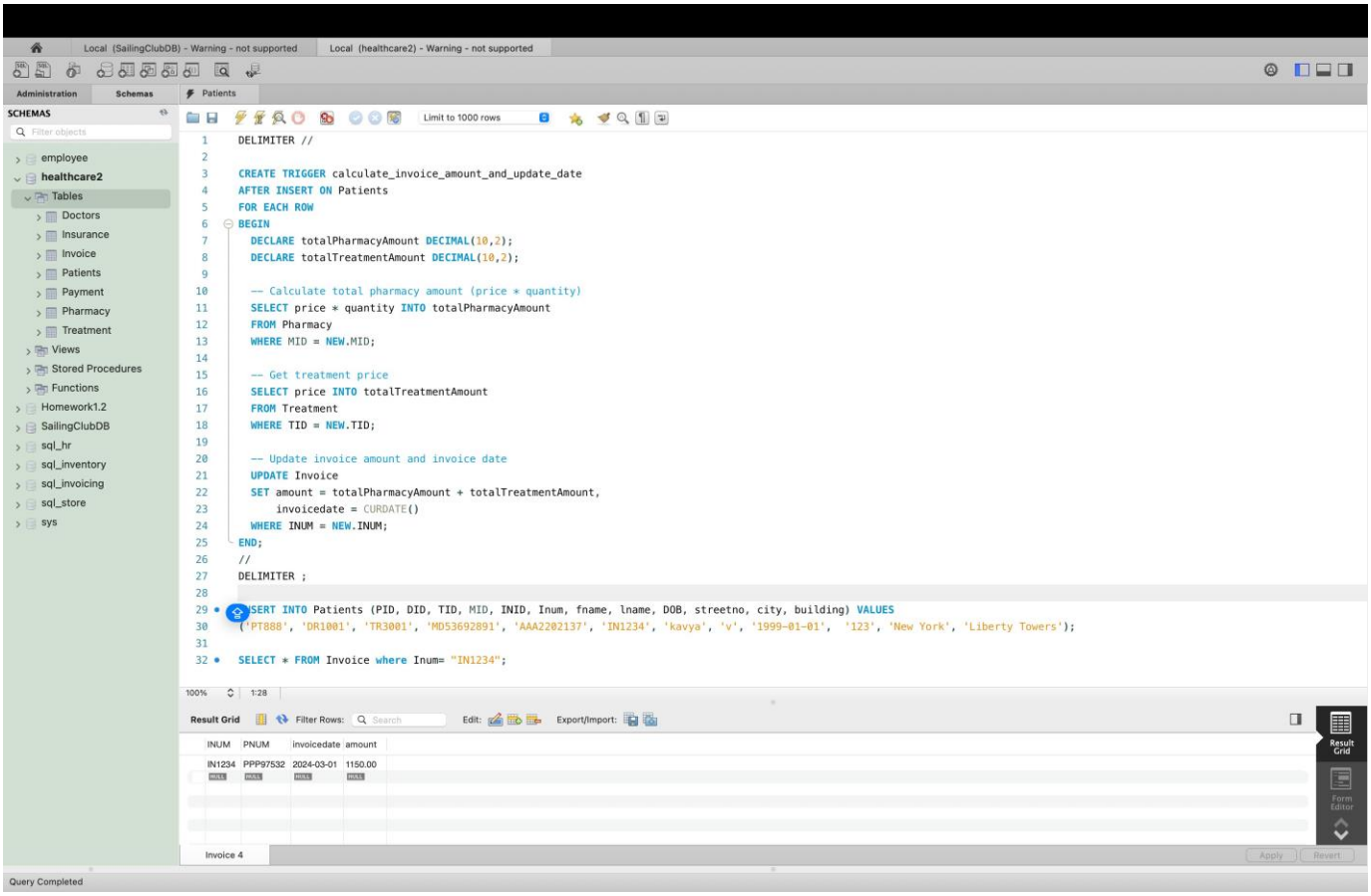
```
DELIMITER //

CREATE TRIGGER
CalculatePatientAgeBEFORE
INSERT ON Patients
FOR EACH
ROWBEGIN
   SET NEW.age = TIMESTAMPDIFF(YEAR, NEW.dob,
CURDATE());END;
//
DELIMITE
R ;

INSERT INTO Patients (PID, DID, TID, MID, INID, Inum, fname, lname, DOB,
streetno, city, building) VALUES
('PT201678', 'DR1001', 'TR3001', 'MD53692891', 'AAA2202137',
'IN1234','ANAYNA', 'GUPTA', '1997-01-01',  '123', 'New York', 'Liberty
Towers');

SELECT * FROM Patients;
```



- **Trigger to calculate total amount and update invoice amount to total amount of medicines and treatment.**

```
DELIMITER //
CREATE TRIGGER
calculate_invoice_amount_and_update_dateAFTER INSERT
ON Patients
FOR EACH
ROWBEGIN
  DECLARE totalPharmacyAmount DECIMAL(10,2);
  DECLARE totalTreatmentAmount DECIMAL(10,2);

  -- Calculate total pharmacy amount (price * quantity)
  SELECT price * quantity INTO totalPharmacyAmount
  FROM Pharmacy
  WHERE MID = NEW.MID;

  -- Get treatment price
  SELECT price INTO
  totalTreatmentAmountFROM Treatment
  WHERE TID = NEW.TID;

  -- Update invoice amount and invoice date
  UPDATE Invoice
  SET amount = totalPharmacyAmount + totalTreatmentAmount,
     invoicedate = CURDATE()
  WHERE INUM =
NEW.INUM;END;
//
DELIMITE
R ;
INSERT INTO Patients (PID, DID, TID, MID, INID, Inum, fname, lname, DOB,
streetno, city, building) VALUE
```

('PT888', 'DR1001', 'TR3001', 'MD53692891', 'AAA2202137', 'IN1234', 'kavya', 'v', '1999-01-01', '123', 'New York', 'Liberty Towers');SELECT * FROM Invoice where Inum= "IN1234";

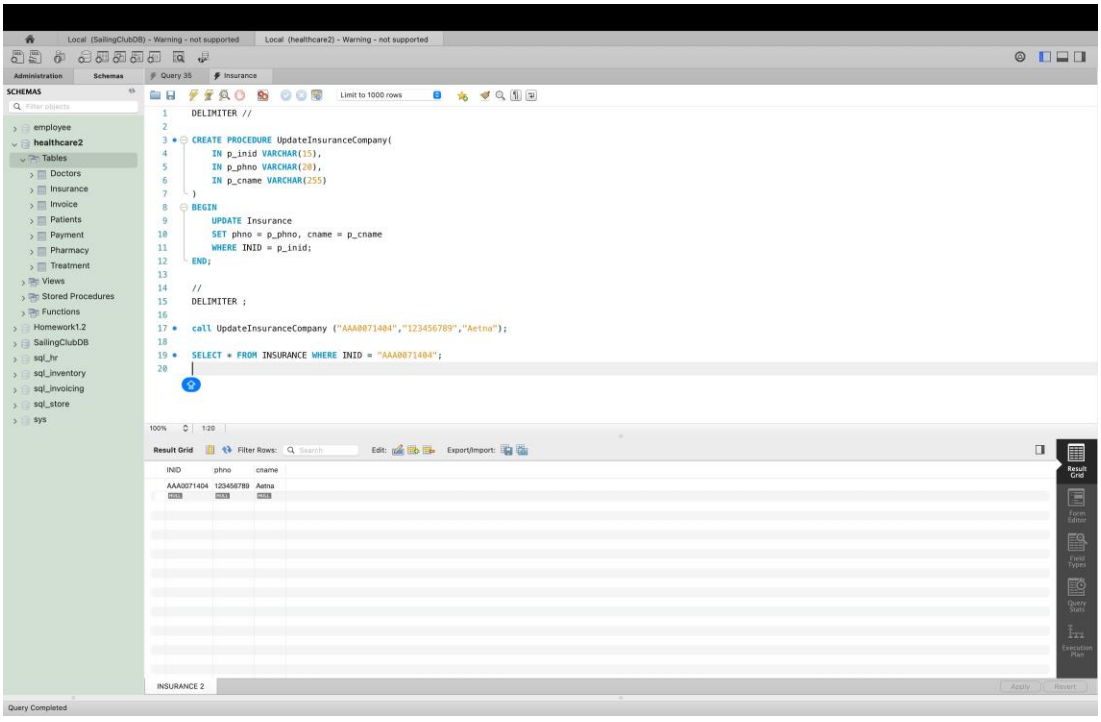


STORED PROCEDURES CREATION:

- **Stored Procedure to update insurance details**

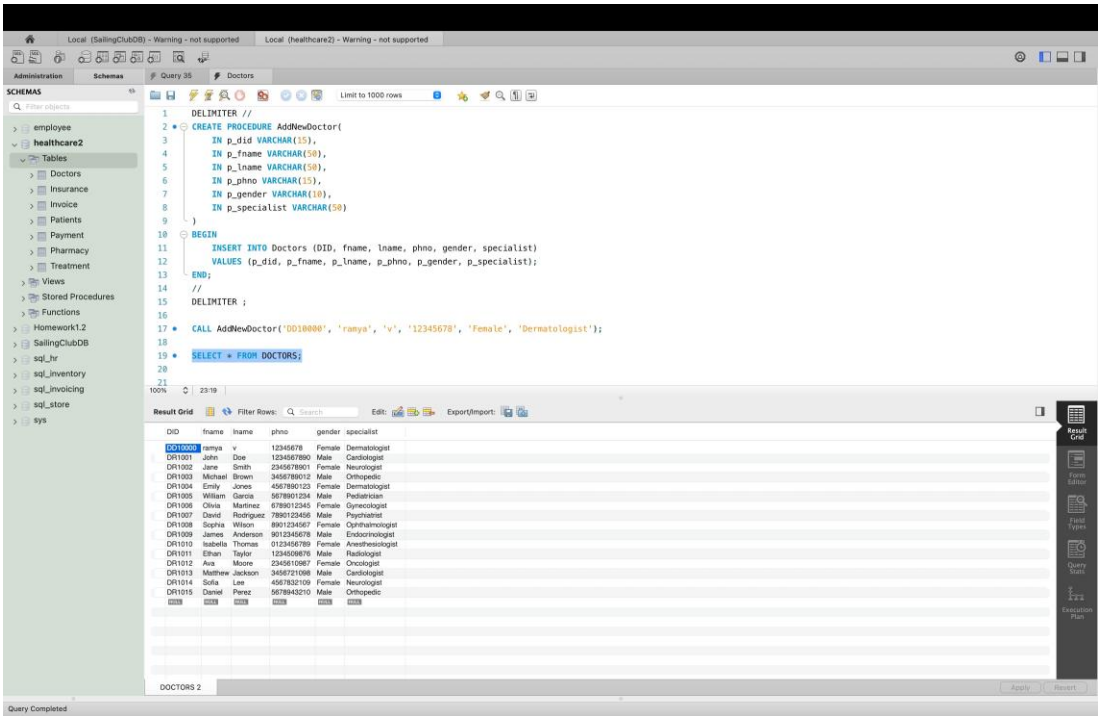
```
DELIMITER //
CREATE PROCEDURE
   UpdateInsuranceCompany(IN p_inid
   VARCHAR(15),
   IN p_phno
   VARCHAR(20), IN
   p_cname VARCHAR(255)
)
BEGI
N
   UPDATE Insurance
   SET phno = p_phno, cname =
   p_cnameWHERE INID = p_inid;
END;
//
DELIMITE
R ;
```

call UpdateInsuranceCompany ("AAA0071404","123456789","Aetna");
SELECT * FROM INSURANCE WHERE INID =”AAA0071404”



- **Stored Procedure to add new doctor into Doctors table.**

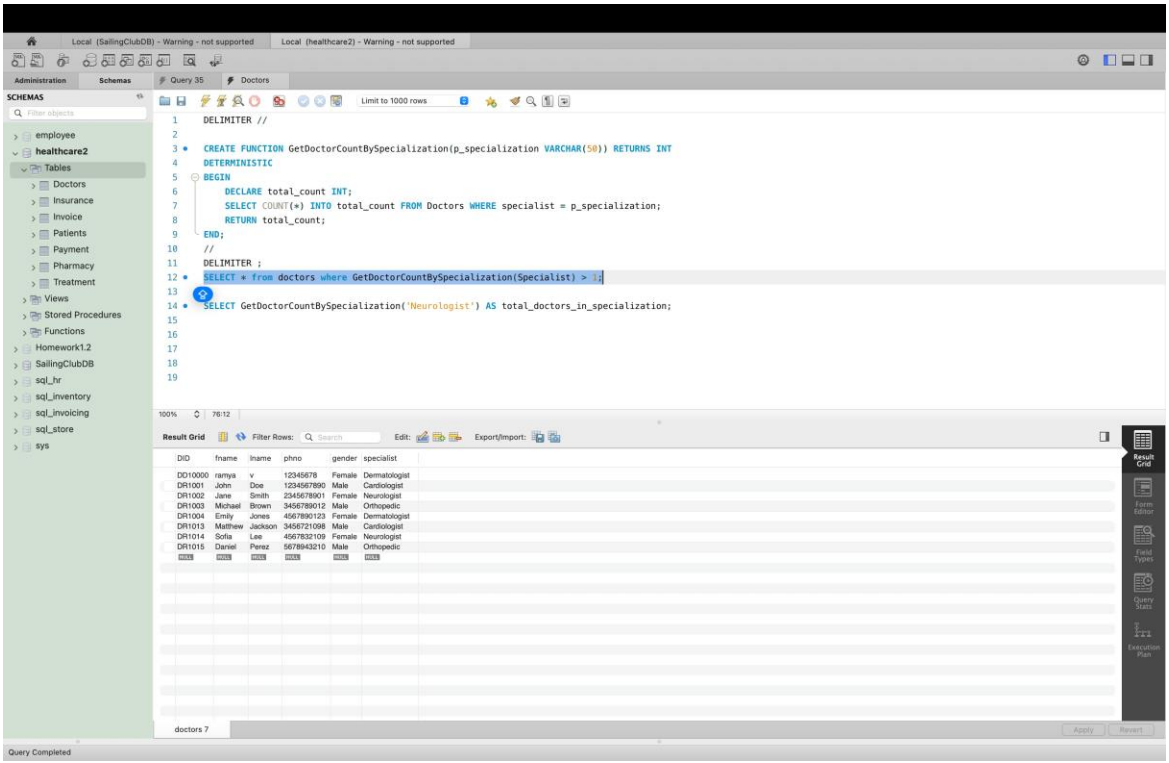
```
DELIMITER //
CREATE PROCEDURE
   AddNewDoctor(IN p_did
   VARCHAR(15),
   IN p_fname
   VARCHAR(50),IN
   p_lname VARCHAR(50),
   IN p_phno
   VARCHAR(15), IN
   p_gender VARCHAR(10),
   IN p_specialist VARCHAR(50)
)
BEGI
N
   INSERT INTO Doctors (DID, fname, lname, phno, gender, specialist)
   VALUES (p_did, p_fname, p_lname, p_phno, p_gender, p_specialist);
END;
// DELIMITER ;

CALL AddNewDoctor('DD10000', 'ramya', 'v', '12345678', 'Female',
'Dermatologist');
SELECT * FROM DOCTORS;
```



FUNCTIONS CREATION:

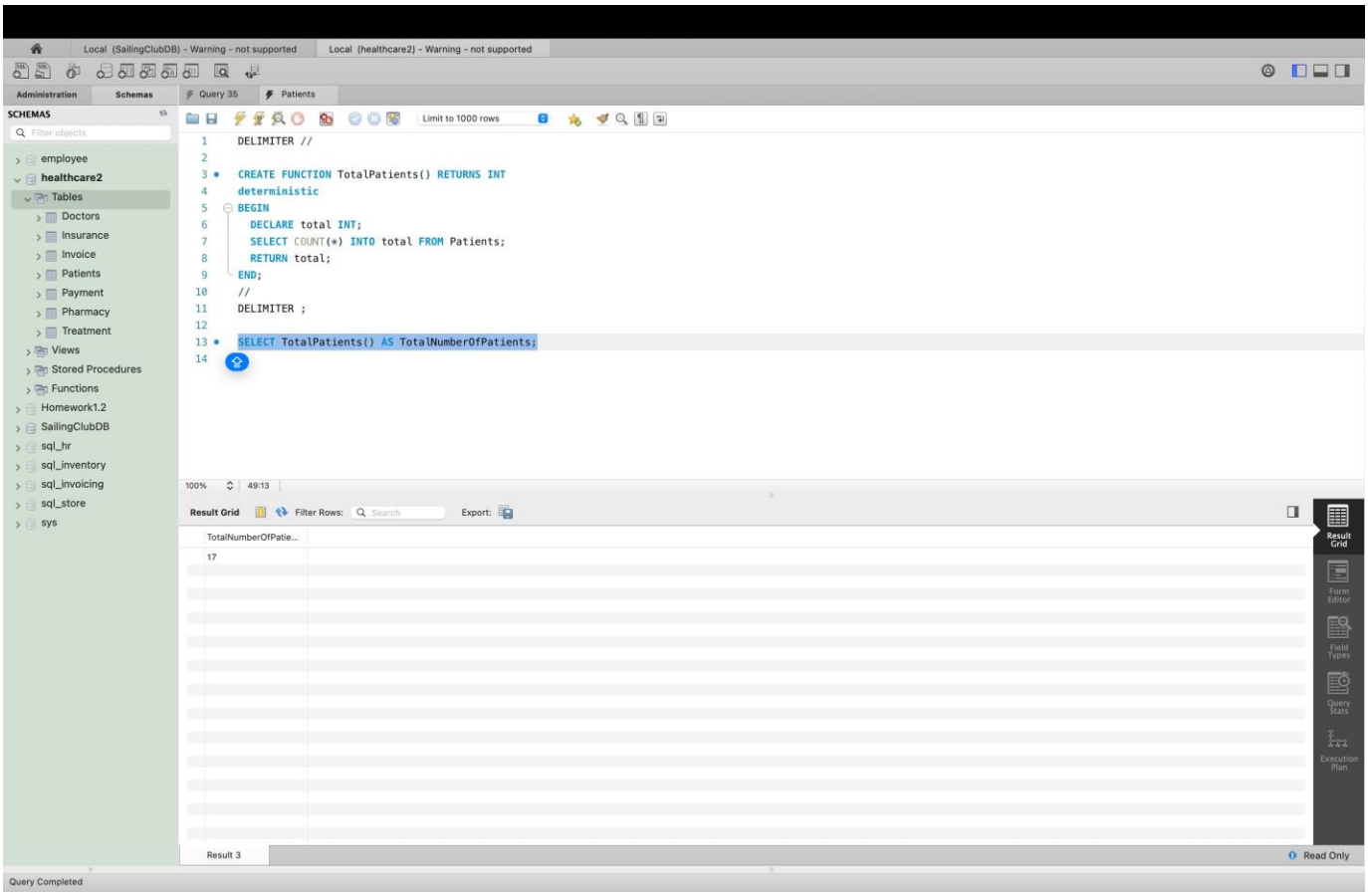
●Function to get count of doctors from same specialization

```
DELIMITER //
CREATE FUNCTION
GetDoctorCountBySpecialization(p_specializationVARCHAR(50))
RETURNS INT
DETERMINIST
IC BEGIN
   DECLARE total_count INT;
   SELECT COUNT(*) INTO total_count FROM Doctors WHERE specialist =
p_specialization;
   RETURN
total_count;END;
//DELIMITER ;

SELECT * from doctors where GetDoctorCountBySpecialization(Specialist) > 1;
SELECT GetDoctorCountBySpecialization('Neurologist') AS
total_doctors_in_specialization;
```



- **Function to get total count of patients.**

```
DELIMITER //
CREATE FUNCTION TotalPatients() RETURNS INT
Deterministic
BEGIN
  DECLARE total INT;
  SELECT COUNT(*) INTO
  total FROM Patients;RETURN
  total;
END:
//DELIMITER ;

SELECT TotalPatients() AS TotalNumberOfPatients;
```



We employed the Python Programming language to execute fundamental Create, Read, Update, and Delete (CRUD) operations. These operations entail adding, retrieving, modifying, and removing database records pertinent for the Database Organization project for “HEALTHCARE MANAGEMENT SYSTEM”. We crafted a program named "Healthcare.py" to establish the Database and for the CRUD operations we are using the CRUD.py file. Within this database, we constructed various tables including Patients, Doctors, Insurance, Invoice, payment, pharmacy, treatment, we populated all tables with relevant data for the Healthcare Management System. Additionally, it provides functionality for computing aggregate functions, performing set operations, and implementing OLAP (Online Analytical Processing) operations.

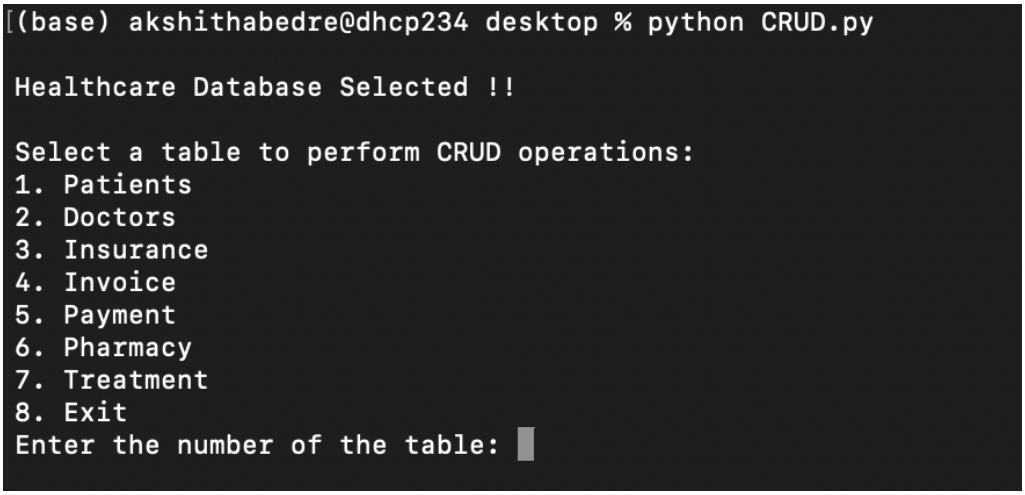
1. CRUD OPERATIONS:

We have developed a robust program named "CRUD.py," functioning as a command line tool tailored for our "Healthcare" database. This tool seamlessly performs essential Create, Read, Update, and Delete (CRUD) operations. With meticulous attention to detail, the program is crafted to uphold code readability and maintainability. Throughout its structure, insightful comments are strategically integrated, enhancing comprehension of both its functionality and implementation details.

2. PROGRAM MENU:

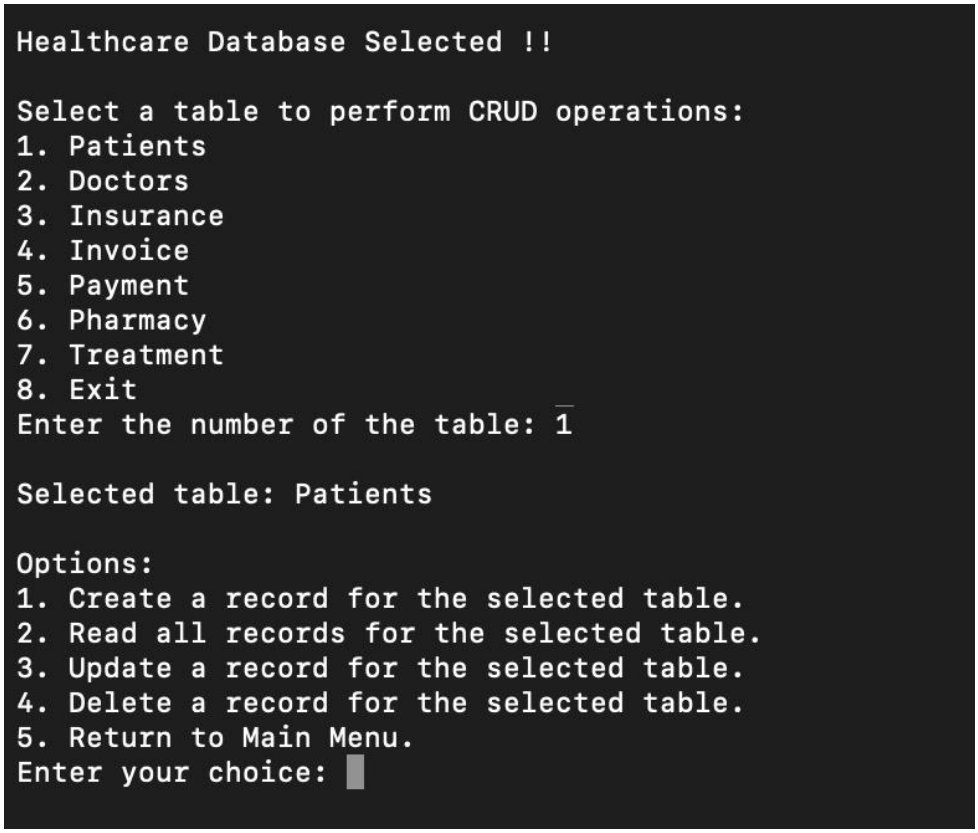
a) Table Menu

The script presents a menu to the user, displaying options to perform CRUD operations on different tables in the database such as Patients, Doctors, Insurance, etc. Users can select a table and then choose an operation (Create, Read, Update, Delete) to perform on that table. Based on the user's choice, the script prompts for necessary input such as record data, record ID, column name, and new values for updating records.



b) Operations Menu

The operations menu within the Python script serves as a pivotal component, facilitating user interaction with the healthcare database through a command-line interface. Upon execution, users are prompted to select a specific table from a comprehensive list including Patients, Doctors, Insurance, and more, indicating the domain of their desired database operation. Following table selection, users are presented with a set of operation choices, enabling them to Create, Read, Update, or Delete records within the chosen table. For instance, users can seamlessly add new records to the database, retrieve existing records for review, modify specific record attributes, or remove outdated entries as needed. The menu meticulously guides users through each operation, prompting for essential inputs such as record data, IDs, or column names, and ensuring data integrity through validation checks.



Test Case 1: Create Operation

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2

Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 1
Doctors
Enter data in this order:
 DoctorsID, fname, lname, phno, gender, specialist
Enter values (comma-separated):
 DR1016,Maheera,Khan,8265262891,Female,Cardiologist
Record created successfully in the Doctors table.
```

```
Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 2

 ------------------------------------------------
DoctorsID | fname | lname | phno | gender | specialist
------------------------------------------------
DR1001 | John | Doe | 1234567890 | Male | Cardiologist
DR1002 | Jane | Smith | 2345678901 | Female | Neurologist
DR1003 | Michael | Brown | 3456789012 | Male | Orthopedic
DR1004 | Emily | Jones | 4567890123 | Female | Dermatologist
DR1005 | William | Garcia | 5678901234 | Male | Pediatrician
DR1006 | Olivia | Martinez | 6789012345 | Female | Gynecologist
DR1007 | David | Rodriguez | 7890123456 | Male | Psychiatrist
DR1008 | Sophia | Wilson | 8901234567 | Female | Ophthalmologist
DR1009 | James | Anderson | 9012345678 | Male | Endocrinologist
DR1010 | Isabella | Thomas | 0123456789 | Female | Anesthesiologist
DR1011 | Ethan | Taylor | 1234509876 | Male | Radiologist
DR1012 | Ava | Moore | 2345610987 | Female | Oncologist
DR1013 | Matthew | Jackson | 3456721098 | Male | Cardiologist
DR1014 | Sofia | Lee | 4567832109 | Female | Neurologist
DR1015 | Daniel | Perez | 5678943210 | Male | Orthopedic
DR1016 | Maheera | Khan | 8265262891 | Female | Cardiologist
------------------------------------------------
```

● Test Case 2: Read Operation

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 1

Selected table: Patients

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 2

 ------------------------------------------------
PatientsID | DoctorsID | PharmacyID | InsuranceID | fname | lname | dob | age | streetno | city | building
------------------------------------------------
PT2016 | DR1001 | MD53692891 | AAA2202137 | Alice | Wong | 1990-01-01 | 34 | 123 | New York | Liberty Towers
PT2017 | DR1002 | MD89869879 | AAA0071404 | Bob | Smith | 1985-02-12 | 39 | 456 | Los Angeles | Sunset Plaza
PT2018 | DR1003 | MD77368580 | AAA8458776 | Charlie | Johnson | 1978-03-23 | 46 | 789 | Chicago | Lake House
PT2019 | DR1004 | MD47609337 | AAA1370939 | Daisy | Lee | 1992-04-14 | 32 | 1011 | Houston | Pine Apartments
PT2020 | DR1005 | MD41093215 | AAA3071239 | Ethan | Brown | 1980-05-25 | 44 | 1213 | Phoenix | Desert Springs
PT2021 | DR1006 | MD83218141 | AAA3641070 | Fiona | Davis | 1994-06-06 | 30 | 1415 | Philadelphia | River View
PT2022 | DR1007 | MD93711778 | AAA6183734 | George | Miller | 1982-07-17 | 42 | 1617 | San Antonio | Oak Villas
PT2023 | DR1008 | MD18868736 | AAA2786040 | Hannah | Wilson | 1975-08-28 | 49 | 1819 | San Diego | Marina Side
PT2024 | DR1009 | MD93487613 | AAA5855663 | Ivan | Moore | 1988-09-09 | 36 | 2021 | Dallas | Highland Loft
PT2025 | DR1010 | MD27465264 | AAA3373097 | Julia | Taylor | 1991-10-10 | 33 | 2223 | San Jose | Central Station
PT2026 | DR1011 | MD16189598 | AAA3970248 | Kevin | Anderson | 1983-11-11 | 41 | 2425 | Austin | Greenwood Residences
PT2027 | DR1012 | MD12194733 | AAA8289774 | Lily | Thomas | 1979-12-12 | 45 | 2627 | Jacksonville | Beachfront Homes
PT2028 | DR1013 | MD45485169 | AAA3098878 | Mason | Jackson | 1986-01-13 | 38 | 2829 | San Francisco | Golden Gates
PT2029 | DR1014 | MD27787881 | AAA6895621 | Nora | Martinez | 1993-02-14 | 31 | 3031 | Indianapolis | Maple Street
PT2030 | DR1015 | MD22587921 | AAA1920740 | Oscar | Hernandez | 1984-03-15 | 40 | 3233 | Columbus | River Park
------------------------------------------------
```

● Test Case 3: Update Operation

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 3

Selected table: Insurance

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 3
Enter the ID of the record you want to update in Insurance: AAA0071404
Insurance Table = ['InsuranceID', 'phno', 'cname']
Enter the name of the column you want to update from the above list: cname
Enter the new value for the cname column: Aetna
Aetna cname AAA0071404 Insurance
Record with ID AAA0071404 is successfully updated in the Insurance table.

Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 3

Selected table: Insurance

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 2

 ------------------------------------------------
InsuranceID | phno | cname
------------------------------------------------
AAA0071404 | 800-234-5678 | Aetna
AAA1370939 | 800-456-7890 | Family Health Protector
AAA1920740 | 800-567-8902 | Unity Health Plans
AAA2202137 | 800-123-4567 | HealthPlus Insurance
AAA2786040 | 800-890-1234 | Prime Health Coverage
AAA3071239 | 800-567-8901 | QuickInsure
AAA3098878 | 800-345-6780 | Global Health Assurance
AAA3373097 | 800-012-3456 | Trusted Health Partners
AAA3641070 | 800-678-9012 | WellnessGuard
AAA3970248 | 800-123-4568 | Nationwide MediCover
AAA5855663 | 800-901-2345 | FutureCare
AAA6183734 | 800-789-0123 | SecureLife Health
AAA6895621 | 800-456-7891 | SafeHealth Systems
AAA8289774 | 800-234-5679 | HealthFirst Insurance
AAA8458776 | 800-345-6789 | MediSecure
------------------------------------------------
```

- **Test Case 4: Delete Operation While executing this Testcase:**

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2

Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 4
Enter the ID of the record you want to delete from Doctors: DR1016
Record with ID DR1016 deleted from the Doctors table.

Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2

Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 2

 ------------------------------------------------
DoctorsID | fname | lname | phno | gender | specialist
------------------------------------------------
DR1001 | John | Doe | 1234567890 | Male | Cardiologist
DR1002 | Jane | Smith | 2345678901 | Female | Neurologist
DR1003 | Michael | Brown | 3456789012 | Male | Orthopedic
DR1004 | Emily | Jones | 4567890123 | Female | Dermatologist
DR1005 | William | Garcia | 5678901234 | Male | Pediatrician
DR1006 | Olivia | Martinez | 6789012345 | Female | Gynecologist
DR1007 | David | Rodriguez | 7890123456 | Male | Psychiatrist
DR1008 | Sophia | Wilson | 8901234567 | Female | Ophthalmologist
DR1009 | James | Anderson | 9012345678 | Male | Endocrinologist
DR1010 | Isabella | Thomas | 0123456789 | Female | Anesthesiologist
DR1011 | Ethan | Taylor | 1234509876 | Male | Radiologist
DR1012 | Ava | Moore | 2345610987 | Female | Oncologist
DR1013 | Matthew | Jackson | 3456721098 | Male | Cardiologist
DR1014 | Sofia | Lee | 4567832109 | Female | Neurologist
DR1015 | Daniel | Perez | 5678943210 | Male | Orthopedic
------------------------------------------------
```

- **Test Case 5: Quit Option**

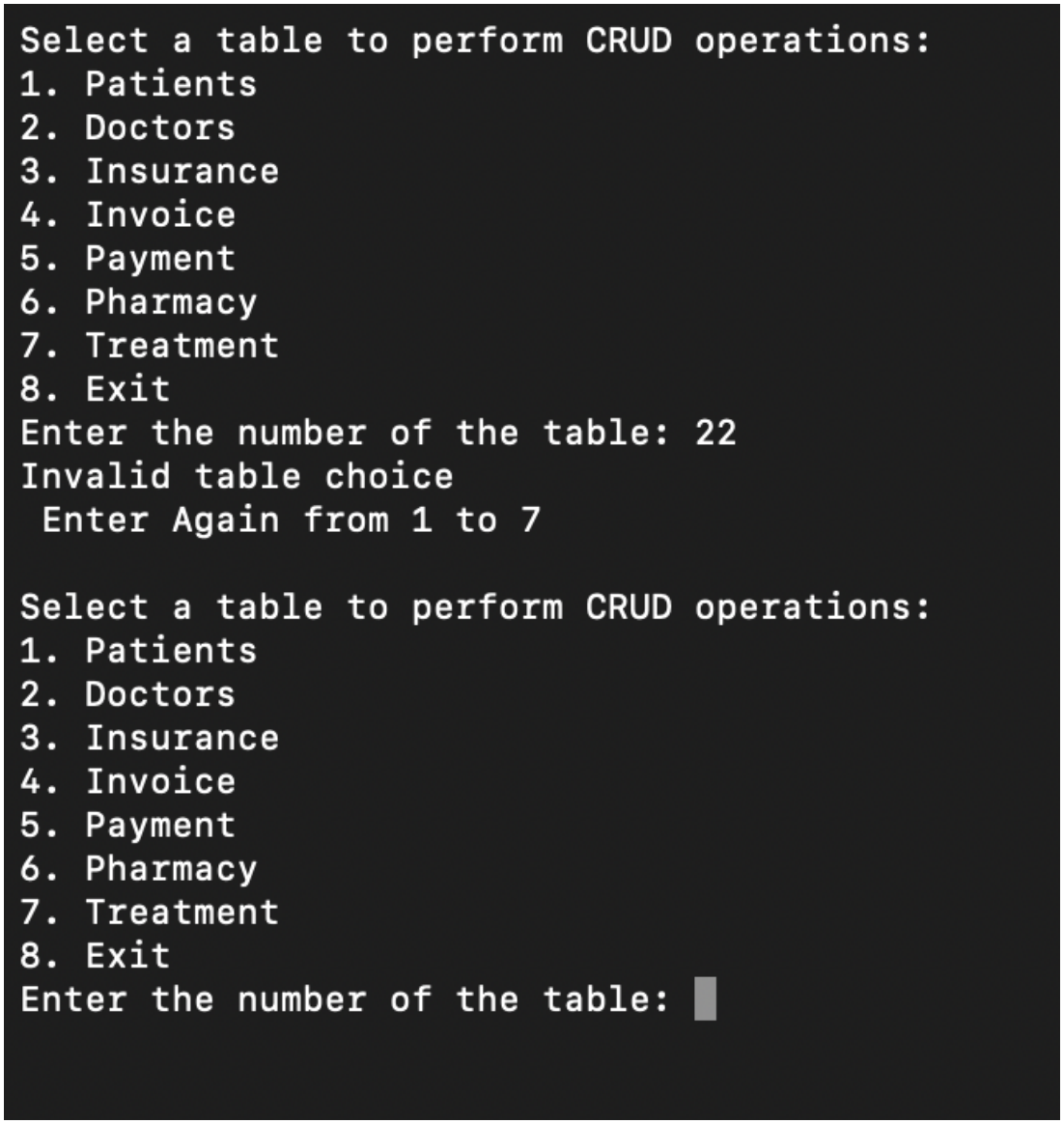
```
(base) akshithabedre@dhcp234 desktop % python CRUD.py

Healthcare Database Selected !!

Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 8
Exited from Healthcare Database
(base) akshithabedre@dhcp234 desktop %
```

3. # Error Handling

We've developed code to handle errors and display concise error messages during the execution of test cases in the program it signifies that the development team has implemented specific programming instructions to effectively manage errors encountered during testing. This code not only detects errors but also generates clear and succinct error messages to provide valuable feedback to users or developers. By prioritizing error handling during the testing phase, the software ensures robustness and reliability, facilitating efficient debugging and enhancing the overall quality of the program. Here we are showing different error handling test cases and how our program lets the user to handle edge cases with clear instructions.

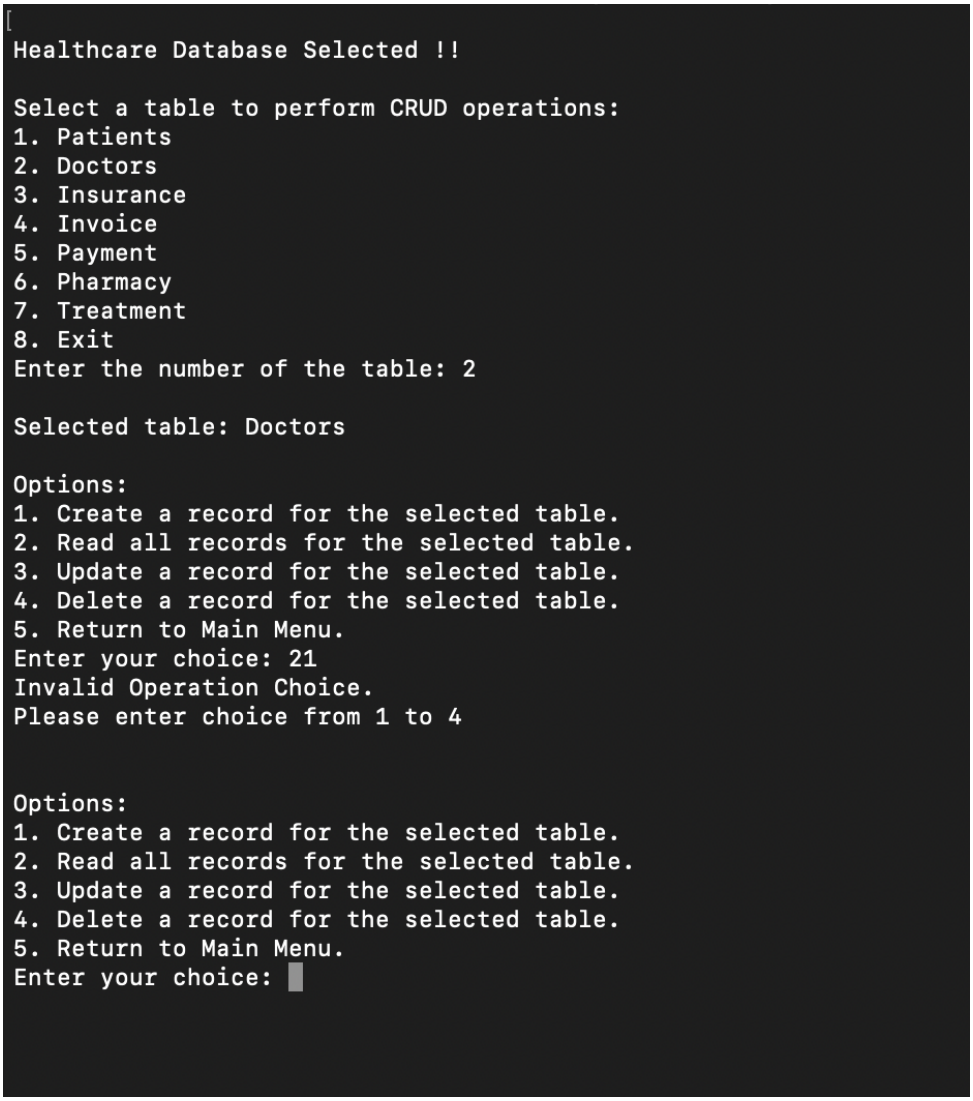
A) Invalid Table number:

When a user inputs a table number that is invalid, consisting of a combination of integers and alphabets, the system will generate a specific error message. This message will provide clear guidance on how to address the issue and proceed correctly. If the user's input does not adhere to our specified data entry format, an error is triggered.



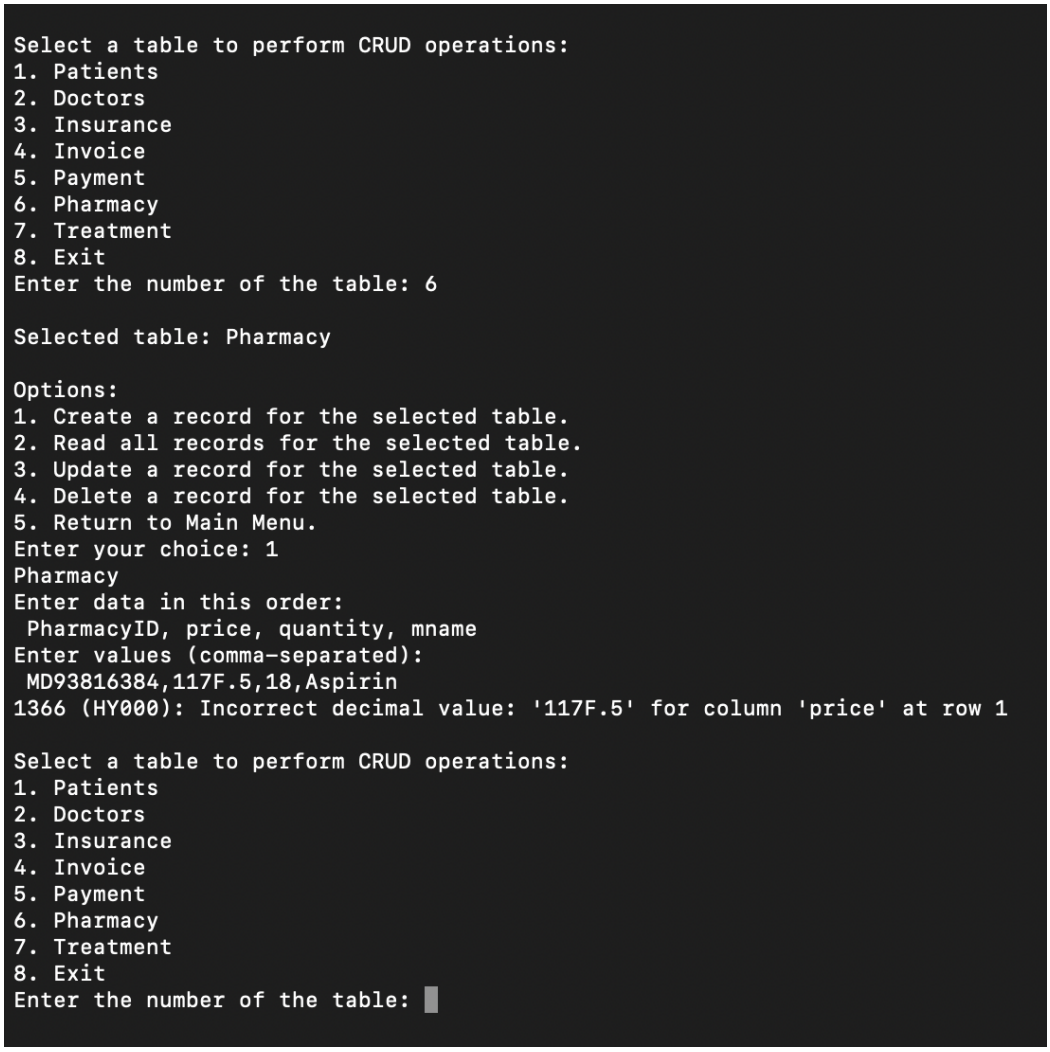
B) Invalid Operation Number:

If a user selects an operation number that is invalid, meaning it doesn't correspond to any recognized operation within the program, the system ensures clarity by offering specific instructions. These instructions are designed to guide the user on how to handle this edge case scenario effectively. Essentially, the program acknowledges the possibility of encountering invalid operation numbers and proactively provides guidance to users, helping them understand what went wrong and how to proceed correctly. This proactive approach enhances user experience and reduces confusion in navigating the system.



C)Error in data for Create Operation:

In the event of an error during the creation operation, particularly when incorrect or incomplete data is provided, the program responds by recognizing the discrepancy and prompts the user to re-enter the necessary data. For instance, when attempting to create a record in the Hospital Table, if the user inputs partial or inaccurate information, the program detects this inconsistency and intervenes by requesting the user to input the data again accurately. This proactive behavior ensures data integrity and completeness within the system, ultimately enhancing the reliability and effectiveness of the program's operations.



D)Error in Update Operation:

In the event of an error during the update operation, specifically when an invalid ID is provided, the program responds by detecting the discrepancy and presenting it as an error message to the user. For instance, when attempting to update the Patients Table and the user inputs an invalid ID, the program recognizes this inconsistency and prompts the user to correct the input by displaying an error message. This proactive approach ensures data accuracy and integrity within the system, guiding users to input valid information and enhancing the overall reliability of the program's update functionality.

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 1

Selected table: Patients

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 3
Enter the ID of the record you want to update in Patients: 12
Record with ID 12 does not exist in the Patients table. Please enter a valid ID from Patients.
Enter the ID of the record you want to update in Patients:
```

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 3

Selected table: Insurance

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 3
Enter the ID of the record you want to update in Insurance: AAA007404
Record with ID AAA007404 does not exist in the Insurance table. Please enter a valid ID from Insurance.
Enter the ID of the record you want to update in Insurance: AAA0071404
Insurance Table = ['InsuranceID', 'phno', 'cname']
Enter the name of the column you want to update from the above list: phno
Enter the new value for the phno column: 8127193214
8127193214 phno AAA0071404 Insurance
Record with ID AAA0071404 is successfully updated in the Insurance table.

Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table:
```

E) Error in Delete Operation

When attempting to delete a record from the "Hospital" table with an invalid ID, such as a random ID that does not correspond to any existing record, the system correctly responded with an error message indicating that the record was not found in the table. This error message is a safeguard against accidental deletion of data, ensuring that only valid and existing records can be removed, thus maintaining data integrity within the database, and providing clear feedback to users or developers about the issue at hand.

```
Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2

Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 4
Enter the ID of the record you want to delete from Doctors: DR1011
Record with ID DR1011 deleted from the Doctors table.

Select a table to perform CRUD operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2

Selected table: Doctors

Options:
1. Create a record for the selected table.
2. Read all records for the selected table.
3. Update a record for the selected table.
4. Delete a record for the selected table.
5. Return to Main Menu.
Enter your choice: 2

 ------------------------------------------------
DoctorsID | fname | lname | phno | gender | specialist
------------------------------------------------
DR1001 | John | Doe | 1234567890 | Male | Cardiologist
DR1002 | Jane | Smith | 2345678901 | Female | Neurologist
DR1003 | Michael | Brown | 3456789012 | Male | Orthopedic
DR1004 | Emily | Jones | 4567890123 | Female | Dermatologist
DR1005 | William | Garcia | 5678901234 | Male | Pediatrician
DR1006 | Olivia | Martinez | 6789012345 | Female | Gynecologist
DR1007 | David | Rodriguez | 7890123456 | Male | Psychiatrist
DR1008 | Sophia | Wilson | 8901234567 | Female | Ophthalmologist
DR1009 | James | Anderson | 9012345678 | Male | Endocrinologist
DR1010 | Isabella | Thomas | 0123456789 | Female | Anesthesiologist
DR1012 | Ava | Moore | 2345610987 | Female | Oncologist
DR1013 | Matthew | Jackson | 3456721098 | Male | Cardiologist
DR1014 | Sofia | Lee | 4567832109 | Female | Neurologist
DR1015 | Daniel | Perez | 5678943210 | Male | Orthopedic
------------------------------------------------
```

Demonstration (Loom Video)

We are providing high-quality Loom video recording that clearly explains the CRUD operations, highlights essential features, and showcases the system's functionality through various scenarios, all in a well-structured and engaging manner. This approach allows viewers to grasp the system's usage, observe its testing, and understand how it operates in different situations. Not only does this educate users on effective system utilization, but it also serves as a compelling display of the system's capabilities.Additionally, We are using functionality for computing aggregate functions, performing set operations, and implementing OLAP (Online Analytical Processing) operations.

Function Definitions:

CRUD Operations: Functions like create_record, read_records, update_record, and delete_record are defined to perform CRUD operations on the tables.

Aggregate Functions: Functions like get_average_value, get_min_value, get_max_value, and get_unique_values are defined to compute aggregate statistics on columns.

Set Operations: Functions like Union_record and Intersect_record are defined to perform set union and intersection operations between two tables.

OLAP Operations: Functions like ntile_table, cumulative_distribution, rollup_data, and cube_simulation are defined to perform OLAP operations such as NTILE, cumulative distribution, rollup, and cube.

Average(invoice):

```
waseempashamohammad@Waseems-MacBook-Pro Group assignment % python3 TD6.py
Healthcare database selected

Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 5
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the name of the column to compute the average value: amount
The average value in amount of Invoice is: 535.666667
```

Count(doctors):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 2
Selected table: Doctors

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 6
['DoctorsID', 'fname', 'lname', 'phno', 'gender', 'specialist']
Enter the name of the column to count records: gender
Enter the value to count in the gender column: male
Count of records with the value 'male' in gender: 8
```

Minimum(invoice):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 7
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the name of the column to compute the minimum value: amount
The minimum value in amount of Invoice is: 115.00
```

Maximum(invoice):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 8
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the name of the column to compute the maximum value: amount
The maximum value in amount of Invoice is: 995.00
```

UNION OPERATORS (pharmacy and payment):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 5
Selected table: Payment

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 9
```

```
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
Enter the second table to execute Union operation: 6
('PPP14926', 'Completed', 'Bank Transfer', '2024-06-10')
('PPP21549', 'Completed', 'Cash', '2024-03-25')
('PPP31748', 'Pending', 'Credit Card', '2025-01-13')
('PPP38741', 'Completed', 'Credit Card', '2024-04-30')
('PPP45981', 'Completed', 'Bank Transfer', '2025-03-24')
('PPP46328', 'Completed', 'Credit Card', '2024-10-30')
('PPP53879', 'Completed', 'Credit Card', '2024-07-15')
('PPP57831', 'Completed', 'Cash', '2024-11-04')
('PPP62487', 'Pending', 'Bank Transfer', '2024-02-20')
('PPP68412', 'Completed', 'Cash', '2025-02-17')
('PPP69214', 'Completed', 'Cash', '2024-08-20')
('PPP76492', 'Pending', 'Cash', '2024-05-05')
('PPP83157', 'Pending', 'Bank Transfer', '2024-09-25')
('PPP92674', 'Completed', 'Bank Transfer', '2024-12-09')
('PPP97532', 'Completed', 'Credit Card', '2024-01-15')
('MD12194733', '110.00', '15', 'Atorvastatin')
('MD16189598', '95.00', '20', 'Levothyroxine')
('MD18868736', '45.00', '90', 'Simvastatin')
('MD22587921', '140.00', '8', 'Hydrochlorothiazide')
('MD27465264', '80.00', '25', 'Lisinopril')
('MD27787881', '130.00', '5', 'Azithromycin')
('MD41093215', '15.00', '150', 'Aspirin')
('MD45485169', '120.00', '10', 'Omeprazole')
('MD47609337', '70.00', '30', 'Ciprofloxacin')
('MD53692891', '20.00', '50', 'Paracetamol')
('MD77368580', '50.00', '75', 'Amoxicillin')
('MD83218141', '25.00', '200', 'Metformin')
('MD89869879', '35.00', '100', 'Ibuprofen')
('MD93487613', '65.00', '40', 'Losartan')
('MD93711778', '55.00', '60', 'Amlodipine')
```

Intersect (patients and doctor):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 1
Selected table: Patients

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 10
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
Enter the second table to execute Intersect operation: 2
Common column is: DoctorsID
Intersection of two tables:
('DR1001', 'John', 'Doe', '1234567890', 'Male', 'Cardiologist')
('DR1002', 'Jane', 'Smith', '2345678901', 'Female', 'Neurologist')
('DR1003', 'Michael', 'Brown', '3456789012', 'Male', 'Orthopedic')
('DR1004', 'Emily', 'Jones', '4567890123', 'Female', 'Dermatologist')
('DR1005', 'William', 'Garcia', '5678901234', 'Male', 'Pediatrician')
('DR1006', 'Olivia', 'Martinez', '6789012345', 'Female', 'Gynecologist')
('DR1007', 'David', 'Rodriguez', '7890123456', 'Male', 'Psychiatrist')
('DR1008', 'Sophia', 'Wilson', '8901234567', 'Female', 'Ophthalmologist')
('DR1009', 'James', 'Anderson', '9012345678', 'Male', 'Endocrinologist')
('DR1010', 'Isabella', 'Thomas', '0123456789', 'Female', 'Anesthesiologist')
('DR1011', 'Ethan', 'Taylor', '1234509876', 'Male', 'Radiologist')
('DR1012', 'Ava', 'Moore', '2345610987', 'Female', 'Oncologist')
('DR1013', 'Matthew', 'Jackson', '3456721098', 'Male', 'Cardiologist')
('DR1014', 'Sofia', 'Lee', '4567832109', 'Female', 'Neurologist')
('DR1015', 'Daniel', 'Perez', '5678943210', 'Male', 'Orthopedic')
```

NTILE OPERATION:

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 11

NTILE Operation Calculation
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the column name to partition: amount
Enter the number of buckets to divide into: 5
Enter the column name to partition: amount
Enter the number of buckets to divide into: 5
Data in Invoice divided into 5 buckets based on the 'amount' column:
Value: 115.00, Bucket: 1
Value: 170.00, Bucket: 1
Value: 225.00, Bucket: 1
Value: 305.00, Bucket: 2
Value: 335.00, Bucket: 2
Value: 395.00, Bucket: 2
Value: 455.00, Bucket: 3
Value: 515.00, Bucket: 3
Value: 575.00, Bucket: 3
Value: 645.00, Bucket: 4
Value: 680.00, Bucket: 4
Value: 815.00, Bucket: 4
Value: 875.00, Bucket: 5
Value: 935.00, Bucket: 5
Value: 995.00, Bucket: 5
```

Cumulative Distribution (Invoice):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 12

Cumulative Distribution Calculation
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the column name for which to calculate the cumulative distribution: amount
Cumulative distribution for amount in Invoice:
Value: 115.00, Cumulative Distribution: 0.0143
Value: 170.00, Cumulative Distribution: 0.0355
Value: 225.00, Cumulative Distribution: 0.0635
Value: 305.00, Cumulative Distribution: 0.1014
Value: 335.00, Cumulative Distribution: 0.1431
Value: 395.00, Cumulative Distribution: 0.1923
Value: 455.00, Cumulative Distribution: 0.2489
Value: 515.00, Cumulative Distribution: 0.3130
Value: 575.00, Cumulative Distribution: 0.3846
Value: 645.00, Cumulative Distribution: 0.4648
Value: 680.00, Cumulative Distribution: 0.5495
Value: 815.00, Cumulative Distribution: 0.6509
Value: 875.00, Cumulative Distribution: 0.7598
Value: 935.00, Cumulative Distribution: 0.8762
Value: 995.00, Cumulative Distribution: 1.0000
```

CUBE (invoice):

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 13
```

```
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the first dimension for CUBE: amount
Enter the second dimension for CUBE: invoicedate
CUBE results for Invoice based on amount and invoicedate:
(Decimal('575.00'), datetime.date(2024, 4, 25), 1, Decimal('575.00'))
(Decimal('170.00'), datetime.date(2024, 1, 10), 1, Decimal('170.00'))
(Decimal('115.00'), datetime.date(2024, 5, 30), 1, Decimal('115.00'))
(Decimal('225.00'), datetime.date(2024, 6, 4), 1, Decimal('225.00'))
(Decimal('455.00'), datetime.date(2024, 7, 9), 1, Decimal('455.00'))
(Decimal('395.00'), datetime.date(2024, 8, 14), 1, Decimal('395.00'))
(Decimal('515.00'), datetime.date(2024, 9, 19), 1, Decimal('515.00'))
(Decimal('680.00'), datetime.date(2024, 10, 24), 1, Decimal('680.00'))
(Decimal('645.00'), datetime.date(2024, 11, 28), 1, Decimal('645.00'))
(Decimal('815.00'), datetime.date(2024, 12, 3), 1, Decimal('815.00'))
(Decimal('875.00'), datetime.date(2025, 1, 7), 1, Decimal('875.00'))
(Decimal('935.00'), datetime.date(2025, 2, 11), 1, Decimal('935.00'))
(Decimal('995.00'), datetime.date(2025, 3, 18), 1, Decimal('995.00'))
(Decimal('335.00'), datetime.date(2024, 2, 15), 1, Decimal('335.00'))
(Decimal('305.00'), datetime.date(2024, 3, 20), 1, Decimal('305.00'))
(Decimal('575.00'), None, 1, Decimal('575.00'))
(Decimal('170.00'), None, 1, Decimal('170.00'))
(Decimal('115.00'), None, 1, Decimal('115.00'))
(Decimal('225.00'), None, 1, Decimal('225.00'))
(Decimal('455.00'), None, 1, Decimal('455.00'))
(Decimal('395.00'), None, 1, Decimal('395.00'))
(Decimal('515.00'), None, 1, Decimal('515.00'))
(Decimal('680.00'), None, 1, Decimal('680.00'))
(Decimal('645.00'), None, 1, Decimal('645.00'))
(Decimal('815.00'), None, 1, Decimal('815.00'))
(Decimal('875.00'), None, 1, Decimal('875.00'))
(Decimal('935.00'), None, 1, Decimal('935.00'))
(Decimal('995.00'), None, 1, Decimal('995.00'))
(Decimal('335.00'), None, 1, Decimal('335.00'))
(Decimal('305.00'), None, 1, Decimal('305.00'))
(None, datetime.date(2024, 4, 25), 1, Decimal('575.00'))
(None, datetime.date(2024, 1, 10), 1, Decimal('170.00'))
(None, datetime.date(2024, 5, 30), 1, Decimal('115.00'))
(None, datetime.date(2024, 6, 4), 1, Decimal('225.00'))
(None, datetime.date(2024, 7, 9), 1, Decimal('455.00'))
(None, datetime.date(2024, 8, 14), 1, Decimal('395.00'))
(None, datetime.date(2024, 9, 19), 1, Decimal('515.00'))
(None, datetime.date(2024, 10, 24), 1, Decimal('680.00'))
(None, datetime.date(2024, 11, 28), 1, Decimal('645.00'))
(None, datetime.date(2024, 12, 3), 1, Decimal('815.00'))
(None, datetime.date(2025, 1, 7), 1, Decimal('875.00'))
(None, datetime.date(2025, 2, 11), 1, Decimal('935.00'))
(None, datetime.date(2025, 3, 18), 1, Decimal('995.00'))
(None, datetime.date(2024, 2, 15), 1, Decimal('335.00'))
(None, datetime.date(2024, 3, 20), 1, Decimal('305.00'))
(None, None, 15, Decimal('8035.00'))
```

14)ROLLUP (invoice)

```
Select a table to perform the operations:
1. Patients
2. Doctors
3. Insurance
4. Invoice
5. Payment
6. Pharmacy
7. Treatment
8. Exit
Enter the number of the table: 4
Selected table: Invoice

Options:
1. Create a record for the selected table
2. Read all records for selected table
3. Update a record for selected table
4. Delete a record for selected table
5. Average value of a column
6. Count appearances in a column
7. Minimun value of a column
8. Maximum value of a column
9. Union Operation
10. Intersect Operation
11. NTile
12. Cumulative Distribution
13. Cube
14. Rollup
15. Return to Main Menu
Enter your choice: 14

Performing ROLLUP Operation
['InvoiceID', 'PaymentID', 'PatientsID', 'invoicedate', 'amount']
Enter the column names to group by (comma-separated): amount
ROLLUP results for Invoice on amount:
(Decimal('115.00'), 1, Decimal('115.00'))
(Decimal('170.00'), 1, Decimal('170.00'))
(Decimal('225.00'), 1, Decimal('225.00'))
(Decimal('305.00'), 1, Decimal('305.00'))
(Decimal('335.00'), 1, Decimal('335.00'))
(Decimal('395.00'), 1, Decimal('395.00'))
(Decimal('455.00'), 1, Decimal('455.00'))
(Decimal('515.00'), 1, Decimal('515.00'))
(Decimal('575.00'), 1, Decimal('575.00'))
(Decimal('645.00'), 1, Decimal('645.00'))
(Decimal('680.00'), 1, Decimal('680.00'))
(Decimal('815.00'), 1, Decimal('815.00'))
(Decimal('875.00'), 1, Decimal('875.00'))
(Decimal('935.00'), 1, Decimal('935.00'))
(Decimal('995.00'), 1, Decimal('995.00'))
(None, 15, Decimal('8035.00'))
```