

PART – 1

DDL Commands to create tables

1. Patient

Schema: Patient (PatientID: integer, BloodTypeID: integer, Name: string, Age: integer, Gender: string)

```
CREATE TABLE Patient (  
    PatientID SERIAL PRIMARY KEY,  
    BloodTypeID INTEGER REFERENCES BloodGroup(BloodTypeID),  
    Name VARCHAR(50),  
    Age INTEGER,  
    Gender VARCHAR(10)
```

```
);
```

```
project=# CREATE TABLE Patient (  
project(#      PatientID SERIAL PRIMARY KEY,  
project(#      BloodTypeID INTEGER REFERENCES BloodGroup(BloodTypeID),  
project(#      Name VARCHAR(50),  
project(#      Age INTEGER,  
project(#      Gender VARCHAR(10)  
project(# );  
CREATE TABLE
```

2. BloodGroup

Schema: BloodGroup (BloodTypeID: integer, BloodType: string)

```
CREATE TABLE BloodGroup (  
    BloodTypeID SERIAL PRIMARY KEY,  
    BloodType VARCHAR(5) UNIQUE
```

```
);
```

```
project=# CREATE TABLE BloodGroup (  
project(#      BloodTypeID SERIAL PRIMARY KEY,  
project(#      BloodType VARCHAR(5) UNIQUE  
project(# );  
CREATE TABLE
```

3. InsuranceProvider

Schema: InsuranceProvider (InsuranceProviderID: integer, PatientID: integer, InsuranceProviderName: string)

```
CREATE TABLE InsuranceProvider (  
    InsuranceProviderID SERIAL PRIMARY KEY,  
    PatientID INTEGER REFERENCES Patient(PatientID),  
    InsuranceProviderName VARCHAR(50)
```

```
);
```

```

project=# CREATE TABLE InsuranceProvider (
project(#      InsuranceProviderID SERIAL PRIMARY KEY,
project(#      PatientID INTEGER REFERENCES Patient(PatientID),
project(#      InsuranceProviderName VARCHAR(50)
project(# );
CREATE TABLE

```

4. Admission

Schema: Admission (AdmissionID: integer, PatientID: integer, DoctorID: integer, HospitalID: integer, MedicationID: integer, AdmissionDate: date, DischargeDate: date)

```

CREATE TABLE Admission (
    AdmissionID SERIAL PRIMARY KEY,
    PatientID INTEGER REFERENCES Patient(PatientID),
    DoctorID INTEGER REFERENCES Doctor(DoctorID),
    HospitalID INTEGER REFERENCES Hospital(HospitalID),
    MedicationID INTEGER REFERENCES Medication(MedicationID),
    AdmissionDate DATE,
    DischargeDate DATE
);

```

```

project=# CREATE TABLE Admission (
project(#      AdmissionID SERIAL PRIMARY KEY,
project(#      PatientID INTEGER REFERENCES Patient(PatientID),
project(#      DoctorID INTEGER REFERENCES Doctor(DoctorID),
project(#      HospitalID INTEGER REFERENCES Hospital(HospitalID),
project(#      MedicationID INTEGER REFERENCES Medication(MedicationID),
project(#      AdmissionDate DATE,
project(#      DischargeDate DATE
project(# );
CREATE TABLE

```

5. AdmissionType

Schema: AdmissionType (AdmissionTypeID: integer, AdmissionID: integer, AdmissionType: string)

```

CREATE TABLE AdmissionType (
    AdmissionTypeID SERIAL PRIMARY KEY,
    AdmissionID INTEGER REFERENCES Admission(AdmissionID),
    AdmissionType VARCHAR(20)
);

```

```

project=# CREATE TABLE AdmissionType (
project(#      AdmissionTypeID SERIAL PRIMARY KEY,
project(#      AdmissionID INTEGER REFERENCES Admission(AdmissionID),
project(#      AdmissionType VARCHAR(20)
project(# );
CREATE TABLE

```

6. Billing

Schema: Billing (BillingID: integer, AdmissionID: integer, BillingAmount: real)

```
CREATE TABLE Billing (  
    BillingID SERIAL PRIMARY KEY,  
    AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
    BillingAmount DECIMAL(10,2)
```

```
);
```

```
project=# CREATE TABLE Billing (  
project(#      BillingID SERIAL PRIMARY KEY,  
project(#      AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
project(#      BillingAmount DECIMAL(10,2)  
project(# );  
CREATE TABLE
```

7. TestResult

Schema: TestResult (AdmissionID: integer, TestResults: string)

```
CREATE TABLE TestResult (  
    AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
    TestResults TEXT,  
    PRIMARY KEY (AdmissionID)
```

```
);
```

```
project=# CREATE TABLE TestResult (  
project(#      AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
project(#      TestResults TEXT,  
project(#      PRIMARY KEY (AdmissionID)  
project(# );  
CREATE TABLE
```

8. Hospital

Schema: Hospital (HospitalID: integer, HospitalName: string)

```
CREATE TABLE Hospital (  
    HospitalID SERIAL PRIMARY KEY,  
    HospitalName VARCHAR(50) NOT NULL
```

```
);
```

```
project=# CREATE TABLE Hospital (  
project(#      HospitalID SERIAL PRIMARY KEY,  
project(#      HospitalName VARCHAR(50) NOT NULL  
project(# );  
CREATE TABLE
```

9. Room

Schema: Room (RoomID: integer, HospitalID: integer, RoomNumber: integer)

```
CREATE TABLE Room (  
    RoomID SERIAL PRIMARY KEY,  
    HospitalID INTEGER REFERENCES Hospital(HospitalID),  
    RoomNumber INTEGER
```

```
);
project=# CREATE TABLE Room (
project(#      RoomID SERIAL PRIMARY KEY,
project(#      HospitalID INTEGER REFERENCES Hospital(HospitalID),
project(#      RoomNumber INTEGER
project(# );
CREATE TABLE
```

10. Doctor

Schema: Doctor (DoctorID: integer, HospitalID: integer, DoctorName: string)

```
CREATE TABLE Doctor (
    DoctorID SERIAL PRIMARY KEY,
    HospitalID INTEGER REFERENCES Hospital(HospitalID),
    DoctorName VARCHAR(50) NOT NULL
);
```

```
project=# CREATE TABLE Doctor (
project(#      DoctorID SERIAL PRIMARY KEY,
project(#      HospitalID INTEGER REFERENCES Hospital(HospitalID),
project(#      DoctorName VARCHAR(50) NOT NULL
project(# );
CREATE TABLE
```

11. Medication

Schema: Medication (MedicationID: integer, DiagnosisID: integer, MedicineName: string)

```
CREATE TABLE Medication (
    MedicationID SERIAL PRIMARY KEY,
    DiagnosisID INTEGER REFERENCES Diagnosis(DiagnosisID),
    MedicineName VARCHAR(50) NOT NULL
);
```

```
project=# CREATE TABLE Medication (
project(#      MedicationID SERIAL PRIMARY KEY,
project(#      DiagnosisID INTEGER REFERENCES Diagnosis(DiagnosisID),
project(#      MedicineName VARCHAR(50) NOT NULL
project(# );
CREATE TABLE
```

12. Diagnosis

Schema: Diagnosis (DiagnosisID: integer, MedicalCondition: string)

```
CREATE TABLE Diagnosis (
    DiagnosisID SERIAL PRIMARY KEY,
    MedicalCondition VARCHAR(100) NOT NULL
);
```

```
project=# CREATE TABLE Diagnosis (
project(#      DiagnosisID SERIAL PRIMARY KEY,
project(#      MedicalCondition VARCHAR(100) NOT NULL
project(# );
CREATE TABLE
```

13. DiagAdm (Associate Entity)

Schema: DiagAdm (AdmissionID: integer, DiagnosisID: integer)

```
CREATE TABLE DiagAdm (  
    AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
    DiagnosisID INTEGER REFERENCES Diagnosis(DiagnosisID),  
    PRIMARY KEY (AdmissionID, DiagnosisID)  
);
```

```
project=# CREATE TABLE DiagAdm (  
project(#      AdmissionID INTEGER REFERENCES Admission(AdmissionID),  
project(#      DiagnosisID INTEGER REFERENCES Diagnosis(DiagnosisID),  
project(#      PRIMARY KEY (AdmissionID, DiagnosisID)  
project(# );  
CREATE TABLE
```

```
project=# \dt  
List of relations  
Schema | Name | Type | Owner  
-----+-----+-----+-----  
public | admission | table | postgres  
public | admissiontype | table | postgres  
public | billing | table | postgres  
public | bloodgroup | table | postgres  
public | diagadm | table | postgres  
public | diagnosis | table | postgres  
public | doctor | table | postgres  
public | hospital | table | postgres  
public | insuranceprovider | table | postgres  
public | medication | table | postgres  
public | patient | table | postgres  
public | room | table | postgres  
public | testresult | table | postgres  
(13 rows)
```

DML Commands to display all the rows inserted into the tables

```
project=# SELECT COUNT(*) FROM Patient;
count
-----
10000
(1 row)

project=# SELECT COUNT(*) FROM Admission;
count
-----
10001
(1 row)
```

```
project=# SELECT * FROM Patient;
patientid |      name      | age | gender | bloodtypeid
-----+-----+-----+-----+-----
4 | Tiffany Ramirez | 81 | Female | 1
5 | Ruben Burns    | 35 | Male   | 2
6 | Chad Byrd      | 61 | Male   | 3
7 | Antonio Frederick | 49 | Male   | 3
8 | Mrs. Brandy Flowers | 51 | Male   | 1
9 | Patrick Parker  | 41 | Male   | 4
10 | Charles Horton  | 82 | Male   | 4
11 | Patty Norman    | 55 | Female | 1
12 | Ryan Hayes      | 33 | Male   | 5
13 | Sharon Perez    | 39 | Female | 1
14 | Amy Roberts     | 45 | Male   | 3
15 | Mrs. Caroline Farrell | 23 | Female | 1
16 | Christina Williams | 85 | Female | 5
17 | William Page    | 72 | Female | 5
18 | Michael Bradshaw | 65 | Female | 4
19 | Brian Dorsey    | 32 | Female | 2
20 | Olivia Gonzalez | 64 | Male   | 6
21 | Teresa Caldwell | 23 | Male   | 5
22 | Desiree Williams MD | 66 | Male   | 2
23 | Sally Shaw      | 80 | Male   | 1
24 | William Johnson | 55 | Female | 4
25 | Steven Bennett  | 79 | Male   | 1
26 | Haley Li        | 51 | Male   | 3
27 | Angela Brown    | 33 | Female | 3
28 | Beverly Miller  | 54 | Male   | 7
29 | Daniel Dickson  | 26 | Female | 3
30 | Kimberly Mason  | 70 | Female | 3
-- More --
```

```
project=# SELECT * FROM BloodGroup;
bloodtypeid | bloodtype
-----+-----
1 | O-
2 | O+
3 | B-
4 | AB+
5 | A+
6 | AB-
7 | A-
8 | B+
(8 rows)
```

```
project=# SELECT * FROM Admission;
admissionid | patientid | doctorid | hospitalid | medicationid | admissiondate | dischargedate
-----+-----+-----+-----+-----+-----+-----
3 | 4 | 4 | 4 | 4 | 2022-11-17 | 2022-12-01
4 | 4 | 5 | 5 | 5 | 2023-06-01 | 2023-06-15
5 | 5 | 6 | 6 | 6 | 2019-01-09 | 2019-02-08
6 | 6 | 7 | 7 | 7 | 2020-05-02 | 2020-05-03
7 | 7 | 8 | 8 | 8 | 2021-07-09 | 2021-08-02
8 | 8 | 9 | 9 | 9 | 2020-08-20 | 2020-08-23
9 | 9 | 10 | 10 | 10 | 2021-03-22 | 2021-04-15
10 | 10 | 11 | 11 | 11 | 2019-05-16 | 2019-06-02
11 | 11 | 12 | 12 | 12 | 2020-12-17 | 2020-12-22
12 | 12 | 13 | 13 | 13 | 2022-12-15 | 2022-12-16
13 | 13 | 14 | 14 | 14 | 2021-04-13 | 2021-05-11
14 | 14 | 15 | 15 | 15 | 2019-06-09 | 2019-06-26
15 | 15 | 16 | 16 | 16 | 2021-11-29 | 2021-12-14
16 | 16 | 17 | 17 | 17 | 2021-07-29 | 2021-08-14
17 | 17 | 18 | 18 | 18 | 2021-06-05 | 2021-06-25
18 | 18 | 19 | 19 | 19 | 2021-08-07 | 2021-08-14
19 | 19 | 20 | 20 | 20 | 2019-11-15 | 2019-12-04
20 | 20 | 21 | 21 | 21 | 2022-03-08 | 2022-03-16
21 | 21 | 22 | 22 | 22 | 2022-06-19 | 2022-06-29
22 | 22 | 23 | 23 | 23 | 2019-07-10 | 2019-08-07
23 | 23 | 24 | 24 | 24 | 2023-02-25 | 2023-03-27
24 | 24 | 25 | 25 | 25 | 2022-12-12 | 2022-12-26
25 | 25 | 26 | 26 | 26 | 2022-10-09 | 2022-11-01
26 | 26 | 27 | 27 | 27 | 2019-01-10 | 2019-01-31
27 | 27 | 28 | 28 | 28 | 2022-08-05 | 2022-09-03
28 | 28 | 29 | 29 | 29 | 2021-05-27 | 2021-06-23
29 | 29 | 30 | 30 | 30 | 2021-07-12 | 2021-07-29
-- More --
```

```
project=# SELECT * FROM Billing ORDER BY BillingID DESC;
billingid | admissionid | billingamount
```

10003	10003	1300.00
10002	10002	37223.97
10001	10001	25236.34
10000	10000	49559.20
9999	9999	5995.72
9998	9998	39606.84
9997	9997	16793.60
9996	9996	12379.13
9995	9995	8296.30
9994	9994	14426.40
9993	9993	48753.13
9992	9992	36044.47
9991	9991	6532.31
9990	9990	27920.31
9989	9989	15872.81
9988	9988	27476.72
9987	9987	37181.84
9986	9986	1675.09
9985	9985	35961.41
9984	9984	46629.77
9983	9983	47369.55
9982	9982	14416.63
9981	9981	4997.58
9980	9980	22434.60
9979	9979	37726.18
9978	9978	20793.03
9977	9977	10021.90
9976	9976	18614.47
9975	9975	27648.10
9974	9974	38831.71
9973	9973	9641.00
9972	9972	7716.36
9971	9971	31861.54

```
project=# SELECT * FROM DiagAdm;
admissionid | diagnosisid
```

3	4
4	5
5	6
6	7
7	8
8	9
9	10
10	11
11	12
12	13
13	14
14	15
15	16
16	17
17	18
18	19
19	20
20	21
21	22
22	23
23	24
24	25
25	26
26	27
27	28
28	29
29	30

-- More --

```
project=# SELECT * FROM Diagnosis;
diagnosisid | medicalcondition
```

4	Diabetes
5	Asthma
6	Obesity
7	Asthma
8	Arthritis
9	Arthritis
10	Hypertension
11	Arthritis
12	Diabetes
13	Asthma
14	Cancer
15	Hypertension
16	Diabetes
17	Diabetes
18	Cancer
19	Arthritis
20	Diabetes
21	Arthritis
22	Obesity
23	Arthritis
24	Arthritis
25	Asthma

```
project=# SELECT * FROM Doctor;
doctorid | hospitalid | doctorname
```

4	4	Patrick Parker
5	5	Diane Jackson
6	6	Paul Baker
7	7	Brian Chandler
8	8	Dustin Griffin
9	9	Robin Green
10	10	Patricia Bishop
11	11	Brian Kennedy
12	12	Kristin Dunn
13	13	Jessica Bailey
14	14	Anthony Roberts
15	15	William Miller
16	16	Laura Roberts
17	17	James Carney
18	18	Katherine Lowe
19	19	Curtis Smith
20	20	Clayton Mcknight
21	21	Debra Meyers
22	22	Megan Sanders
23	23	Zachary Horton DDS
24	24	Kelly Thompson
25	25	Michael Chang
26	26	Nicole Wood
27	27	Angela Kim
28	28	Jodi Holland
29	29	Christina Flores
30	30	Natalie Sullivan
31	31	Carolyn Baker MD
32	32	Nicole McClain
33	33	Chloe Thomas
34	34	Brian Watson
35	35	Dr. Kyle Dickson
36	36	Michael Smith
37	37	Pamela Brown
38	38	Tiffany Crawford
39	39	Rachel Sullivan
40	40	Samuel Taylor
41	41	John Harvey
42	42	Sylvia Johnson

PART – 2

The tables are already in the BCNF form.

Justification:

1. Patient:
 - a. The primary key (PatientID) uniquely identifies each patient.
 - b. All attributes (Name, Age, Gender) depend on the entire primary key, not just a part of it.
2. BloodGroup:
 - a. The primary key (BloodTypeID) uniquely identifies each blood group.
 - b. BloodType depends on the entire primary key.
3. InsuranceProvider:
 - a. The primary key (InsuranceProviderID) uniquely identifies each provider for each patient.
 - b. PatientID (foreign key) depends on the whole primary key (ensures a provider is associated with a specific patient).
 - c. InsuranceProviderName depends on the whole primary key (indirectly through PatientID).
4. Admission:
 - a. AdmissionID uniquely identifies each admission record.
 - b. All foreign keys (PatientID, DoctorID, HospitalID, MedicationID) depend on the entire AdmissionID, not just PatientID.
5. AdmissionType:
 - a. AdmissionType depends on the Admission (via AdmissionID as foreign key). The dependency chain goes through the entire AdmissionID for AdmissionType to be valid.
6. Billing:
 - a. The primary key (BillingID) uniquely identifies each billing record.
 - b. AdmissionID (foreign key) depends on the whole primary key (ensures a billing is associated with a specific admission).
7. TestResult:
 - a. The foreign key (AdmissionID) depends on the whole primary key of Admission (assuming Admission is in BCNF).
8. Hospital:
 - a. The primary key (HospitalID) uniquely identifies each hospital.
 - b. HospitalName depends on the whole primary key.
9. Room:
 - a. The primary key (RoomID) uniquely identifies each room.
 - b. HospitalID (foreign key) depends on the whole primary key (ensures a room belongs to a specific hospital).
10. Doctor:
 - a. DoctorID uniquely identifies each doctor.

- b. Since a doctor can only work in one hospital (assumption according to the dataset), HospitalID directly depends on the entire DoctorID (no partial dependency on PatientID from Admission).
- 11. Medication:
 - a. MedicationID uniquely identifies each medication.
 - b. DiagnosisID (foreign key) as a foreign key referencing the Diagnosis table. Since a medication is specific to a diagnosis (assumption according to the dataset), DiagnosisID directly depends on the entire MedicationID (no partial dependency on PatientID from Admission).
- 12. Diagnosis:
 - a. The primary key (DiagnosisID) uniquely identifies each diagnosis.
 - b. MedicalCondition depends on the whole primary key.
- 13. DiagAdm (Associative Entity):
 - a. By definition, an associative entity has a composite primary key that uniquely identifies each relationship between the participating entities (Admission and Diagnosis in this case).

Here are all the relations (in BCNF) form:

- 1. Patient:
 - a. PatientID (Primary Key)
 - b. BloodTypeID (Foreign Key references BloodGroup)
 - c. Name
 - d. Age
 - e. Gender
- 2. BloodGroup:
 - a. BloodTypeID (Primary Key)
 - b. BloodType
- 3. InsuranceProvider:
 - a. InsuranceProviderID (Primary Key)
 - b. PatientID (Foreign Key references Patient)
 - c. InsuranceProviderName
- 4. Admission:
 - a. AdmissionID (Primary Key)
 - b. PatientID (Foreign Key references Patient)
 - c. DoctorID (Foreign Key references Doctor)
 - d. HospitalID (Foreign Key references Hospital)
 - e. MedicationID (Foreign Key references Medication)
 - f. AdmissionDate
 - g. DischargeDate
- 5. AdmissionType:
 - a. AdmissionTypeID (Primary Key)
 - b. AdmissionID (Foreign Key references Admission)

- c. AdmissionType (Emergency, Elective, Urgent)
- 6. Billing:
 - a. BillingID (Primary Key)
 - b. AdmissionID (Foreign Key references Admission)
 - c. BillingAmount
- 7. TestResult:
 - a. AdmissionID (Foreign Key references Admission)
 - b. TestResults
- 8. Hospital:
 - a. HospitalID (Primary Key)
 - b. HospitalName
- 9. Room:
 - a. RoomID (Primary Key)
 - b. HospitalID (Foreign Key references Hospital)
 - c. RoomNumber
- 10. Doctor:
 - a. DoctorID (Primary Key)
 - b. HospitalID (Foreign Key references Hospital)
 - c. DoctorName
- 11. Medication:
 - a. MedicationID (Primary Key)
 - b. DiagnosisID (Foreign Key references Diagnosis)
 - c. MedicineName
- 12. Diagnosis:
 - a. DiagnosisID (Primary Key)
 - b. MedicalCondition
- 13. DiagAdm (Associative Entity):
 - a. AdmissionID (Foreign Key references Admission) – Primary Key part 1
 - b. DiagnosisID (Foreign Key references Diagnosis) – Primary Key part 2

PART – 3

Functionalities and their corresponding screenshots are as follows:

1. Insert a new patient:

```
INSERT INTO BloodGroup (BloodType) VALUES ({blood_type}) RETURNING BloodTypeID;
INSERT INTO Patient (Name, Age, Gender, BloodTypeID) VALUES ({name}, {age}, {gender},
{bloodtype_id}) RETURNING PatientID;
INSERT INTO InsuranceProvider (PatientID, InsuranceProviderName) VALUES ({patient_id},
{insurance_provider});
INSERT INTO Hospital (HospitalName) VALUES ({hospital}) RETURNING HospitalID;
INSERT INTO Room (HospitalID, RoomNumber) VALUES ({hospital_id}, {room_number});
INSERT INTO Doctor (HospitalID, DoctorName) VALUES ({hospital_id}, {doctor}) RETURNING
DoctorID;
INSERT INTO Diagnosis (MedicalCondition) VALUES ({medical_condition}) RETURNING
DiagnosisID;
INSERT INTO Medication (DiagnosisID, MedicineName) VALUES ({diagnosis_id},
{medication}) RETURNING MedicationID;
INSERT INTO Admission (PatientID, DoctorID, HospitalID, MedicationID, AdmissionDate,
DischargeDate) VALUES ({patient_id}, {doctor_id}, {hospital_id}, {medication_id},
{admission_date}, {discharge_date}) RETURNING AdmissionID;
INSERT INTO AdmissionType (AdmissionID, AdmissionType) VALUES ({admission_id},
{admission_type});
INSERT INTO Billing (AdmissionID, BillingAmount) VALUES ({admission_id},
{billing_amount});
INSERT INTO TestResult (AdmissionID, TestResults) VALUES ({admission_id}, {test_results});
INSERT INTO DiagAdm (AdmissionID, DiagnosisID) VALUES ({admission_id}, {diagnosis_id});
```

Screenshot below:

```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 1
Enter Name: Harshit Jain
Enter Age: 20
Enter Gender: Male
Enter Blood Type: A+
Enter Medical Condition (optional): Lower Back Pain
Enter Date of Admission (MM/DD/YYYY): 03/20/2024
Enter Admission Type: Elective
Enter Doctor Name (optional): Jake Cole
Enter Hospital Name: Hospital ABC Inc.
Enter Insurance Provider (optional): UnitedHealthCareSR
Enter Room Number (optional): 202
Enter Discharge Date (MM/DD/YYYY) (optional): 04/01/2024
Enter Medication (optional): Yoga
Enter Test Results (optional): Normal
Enter Billing Amount (optional): 1200
New patient added successfully!

```

```

project=# SELECT * FROM Patient WHERE name = 'Harshit Jain';
 patientid |      name      | age | gender | bloodtypeid
-----+-----+-----+-----+-----
      10004 | Harshit Jain   |  20 | Male   |           5
(1 row)

```

2. Delete the expired insurance on file for a patient

```
DELETE FROM InsuranceProvider WHERE PatientID = {patient_id};
```

```

project=# SELECT * FROM InsuranceProvider WHERE PatientID = 10004;
 insuranceproviderid | patientid | insuranceprovidername
-----+-----+-----
              10002 |      10004 | UnitedHealthCareSR
(1 row)

```

```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 2
Enter the Patient ID: 10004
Insurance provider for Patient ID 10004 deleted successfully!

```

```

project=# SELECT * FROM InsuranceProvider WHERE PatientID = 10004;
insuranceproviderid | patientid | insuranceprovidername
-----+-----+-----
(0 rows)

```

3. Update the billing amount for a patient

```

UPDATE Billing SET BillingAmount = {new_billing_amount}
WHERE AdmissionID = {admission_id}

```

```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 3
Enter the Admission ID: 10003
Enter the new Billing Amount: 1300
Billing amount for Admission ID 10003 updated successfully!

```

```

project=# SELECT * FROM Billing WHERE AdmissionID = 10003;
billingid | admissionid | billingamount
-----+-----+-----
    10003 |         10003 |         1300.00
(1 row)

```

4. Search for patients by the doctor assigned to them

```
SELECT Admission.PatientID, Name, AdmissionID, Admission.DoctorID, DoctorName,
AdmissionDate, DischargeDate
FROM Admission
JOIN Patient ON Admission.PatientID = Patient.PatientID
JOIN Doctor ON Admission.DoctorID = Doctor.DoctorID
WHERE Doctor.DoctorName LIKE '%{doctor_name}%'
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 4
Enter the doctor's full name: Jake Cole

Search results for patients:
patientid    name    admissionid  doctorid  doctorname  admissiondate  dischargedate
10004 Harshit Jain    10003      10004    Jake Cole    2024-03-20    2024-04-01
```

5. Aggregate Functions on Patient table

```
SELECT COUNT(*) FROM Patient;
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 5
1. Count Total Patients
2. Find Average Age of Patients
Enter your choice (1-2): 1
Total Patients: 10001
```

SELECT AVG(Age) FROM Patient;

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 5
1. Count Total Patients
2. Find Average Age of Patients
Enter your choice (1-2): 2
Average Patient Age: 51.4490550944905509
```

6. Sorts patients by age in descending order

SELECT * FROM Patient ORDER BY Age DESC;

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 6

Patients Sorted by Age (Descending):
patientid      name      age gender  bloodtypeid
3933           Scott Underwood  85 Female    6
997            Joshua Phillips  85 Male      3
4399           George Steele   85 Female    1
5265           Robert Bates    85 Male      2
107            Amber Solomon   85 Male      5
6812           Craig Bryant    85 Female    2
7383           Thomas Bishop   85 Female    3
9671           Yesenia Marks   85 Female    5
9665           George Smith    85 Female    6
1964           Craig Jensen    85 Male      4
1316           Gregory Russell  85 Female    4
6540           Joseph Stevens  85 Female    8
6835           Tabitha Gray    85 Female    7
5737           Todd Cross      85 Female    2
5188           Michael Harding 85 Male      1
2802           Lisa Ferguson   85 Female    5
6935           James Richard   85 Male      1
```

7. Joins Patients and Admission tables for emergency admissions with O- blood type

```
SELECT P.Name, BG.BloodType, A.AdmissionDate, AT.AdmissionType
FROM Patient P
INNER JOIN Admission A ON P.PatientID = A.PatientID
INNER JOIN AdmissionType AT ON A.AdmissionID = AT.AdmissionID
INNER JOIN BloodGroup BG ON P.BloodTypeID = BG.BloodTypeID
WHERE AT.AdmissionType = 'Emergency' AND BG.BloodType = 'O-';
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 7

Emergency Admissions with Patients having Blood Type O-:
      name bloodtype admissiondate admissiontype
Mrs. Caroline Farrell      O-      2019-06-09      Emergency
      Sally Shaw           O-      2019-07-10      Emergency
      Rachael Davidson     O-      2022-03-29      Emergency
      Angela Sanchez       O-      2021-06-13      Emergency
      Tyler Rosario        O-      2021-07-07      Emergency
      Joyce Vaughn         O-      2021-09-22      Emergency
      Zachary Wood         O-      2021-06-23      Emergency
      Paula Knight         O-      2020-03-04      Emergency
      Ryan Cross           O-      2019-05-18      Emergency
      William Mahoney      O-      2023-10-28      Emergency
      Laurie Turner        O-      2023-07-02      Emergency
      Whitney Garza        O-      2021-12-10      Emergency
      Victor Gardner        O-      2019-11-12      Emergency
      Amanda Smith MD      O-      2019-02-02      Emergency
      Christopher Duffy     O-      2019-02-09      Emergency
      Kayla Jackson        O-      2023-01-24      Emergency
```

8. Groups patients based on user-specified columns

```
SELECT AVG(P.Age), P.{column} FROM Patient P GROUP BY {column};
```



```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 8
Enter the column to group by: BloodTypeID

Grouping results:
      avg bloodtypeid
52.2290996784565916      8
50.7552504038772213      7
51.1382636655948553      1
52.1143317230273752      5
51.3529411764705882      4
51.3990384615384615      2
51.7623529411764706      6
50.8362619808306709      3

```

```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients who have been admitted more than once using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 8
Enter the column to group by: Gender

Grouping results:
      avg gender
51.6084729064039409 Female
51.2848152659358506  Male

```

9. Finds patients with a length of stay exceeding a threshold using a subquery
subquery = SELECT PatientID, AdmissionDate, DischargeDate, (DischargeDate - AdmissionDate) AS LengthOfStay FROM Admission;
query = SELECT P.PatientID, P.Name, Stay.AdmissionDate, Stay.DischargeDate, Stay.LengthOfStay
FROM Patient P
INNER JOIN ({subquery}) AS Stay ON P.PatientID = Stay.PatientID
WHERE Stay.LengthOfStay > {threshold_days};

```

Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 9
Enter the minimum threshold for length of stay (in days): 28

Patients with average length of stay exceeding 28 days:

```

patientid	name	admissiondate	dischargedate	lengthofstay
6	Chad Byrd	2019-01-09	2019-02-08	30
24	William Johnson	2023-02-25	2023-03-27	30
28	Beverly Miller	2022-08-05	2022-09-03	29
54	Dylan Mcknight	2019-06-14	2019-07-13	29
85	Douglas Crawford	2023-08-16	2023-09-15	30
87	Kimberly Vargas	2018-12-28	2019-01-27	30
88	Travis Walker	2019-01-12	2019-02-11	30
95	Ariel Davis	2021-10-23	2021-11-21	29
96	Joyce Vaughn	2021-09-22	2021-10-22	30
100	Steven Boyer	2020-12-10	2021-01-09	30
104	Paul Greer	2020-10-05	2020-11-03	29
106	Scott Adams	2022-10-31	2022-11-29	29
108	Jacob Howell	2019-08-25	2019-09-23	29
110	Kurt Sloan	2022-06-14	2022-07-13	29
118	Susan Mills	2020-03-08	2020-04-06	29
119	Kelly Manning	2019-06-21	2019-07-20	29

10. Discharges a patient using a transaction

```

BEGIN TRANSACTION;
SELECT * FROM Admission WHERE PatientID = {patient_id} AND DischargeDate IS NULL;
UPDATE Admission SET DischargeDate = {discharge_date} WHERE AdmissionID =
{active_admission[0]};
COMMIT/ROLLBACK;

```

a. DischargeDate is NULL. That means, it's an active admission.

```

project=# SELECT * FROM Admission WHERE PatientID = 10004;

```

admissionid	patientid	doctorid	hospitalid	medicationid	admissiondate	dischargedate
10003	10004	10004	8643	10004	2024-03-20	

(1 row)

b. Patient is discharged and DischargeDate has been updated to the current date.

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 10
Enter Patient ID: 10004
Patient ID 10004 discharged successfully!
```

c. DischargeDate updated (COMMIT) in the database.

```
project=# SELECT * FROM Admission WHERE PatientID = 10004;
 admissionid | patientid | doctorid | hospitalid | medicationid | admissiondate | dischargedate
-----+-----+-----+-----+-----+-----+-----
          10003 |      10004 |      10004 |          8643 |          10004 | 2024-03-20 | 2024-04-19
(1 row)
```

d. Since the patient is successfully discharged and that patient is no longer having an active admission, the transaction will ROLLBACK if ran again.

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 10
Enter Patient ID: 10004
An error occurred: Patient not found or has no active admission.
Transaction rolled back.
```

11. Bonus: Custom Query

Here, the user can enter any additional query they would like.

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 11
Enter your desired SQL query:
> SELECT * FROM BloodGroup;

Query Results:
bloodtypeid bloodtype
1           0-
2           0+
3           B-
4           AB+
5           A+
6           AB-
7           A-
8           B+
```

12. Error Handling

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 2
Enter the Patient ID: 1ST452FGSW
An error occurred: invalid literal for int() with base 10: '1ST452FGSW'
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 3
Enter the Admission ID: 1AW235F3
An error occurred: invalid literal for int() with base 10: '1AW235F3'
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 4
Enter the doctor's full name: Iron Man
No patients found assigned to Dr. Iron Man.
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 8
Enter the column to group by: Product
An error occurred while fetching data: column p.product does not exist
```

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 9
Enter the minimum threshold for length of stay (in days): 40
No patients found with average stay exceeding 40 days.
```

13. Exit

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): 13
Exiting the system...
PS F:\US Documents_2\Classes\7_Spring 2024\CMPSC 431W\Project> |
```

PART – 4

CLI Interface

```
Welcome to the Hospital Management System!
1. Insert a new patient
2. Delete the expired insurance on file for a patient
3. Update the billing amount for a patient
4. Search for patients by the doctor assigned to them
5. Aggregate Functions on Patient table
6. Sorts patients by age in descending order
7. Joins Patients and Admission tables for emergency admissions with O- blood type
8. Groups patients based on user-specified columns
9. Finds patients with a length of stay exceeding a threshold using a subquery
10. Discharges a patient using a transaction
11. Bonus: Custom Query
12. Error Handling
13. Exit
Enter your choice (1-13): █
```