MIDTERM PROJECT

HEALTHCARE INDUSTRY DATABASE

TEAM:

Bharath Anand (02044023) – 2

Pradyoth Singenahalli Prabhu (02071847) - 42

Varun W R (02070206) - 49

Part-1: Planned Database-driven Application requirements

Developing a comprehensive database for the US healthcare sector requires careful consideration of several factors to ensure the accuracy, privacy, and security of the data.

The database will need to store and maintain various critical information related to patient health, such as treatment records, prescribed medications, physician visits, and other important medical data. It will also need to store confidential patient demographics, including personal identifying information such as name, date of birth, address, and phone number.

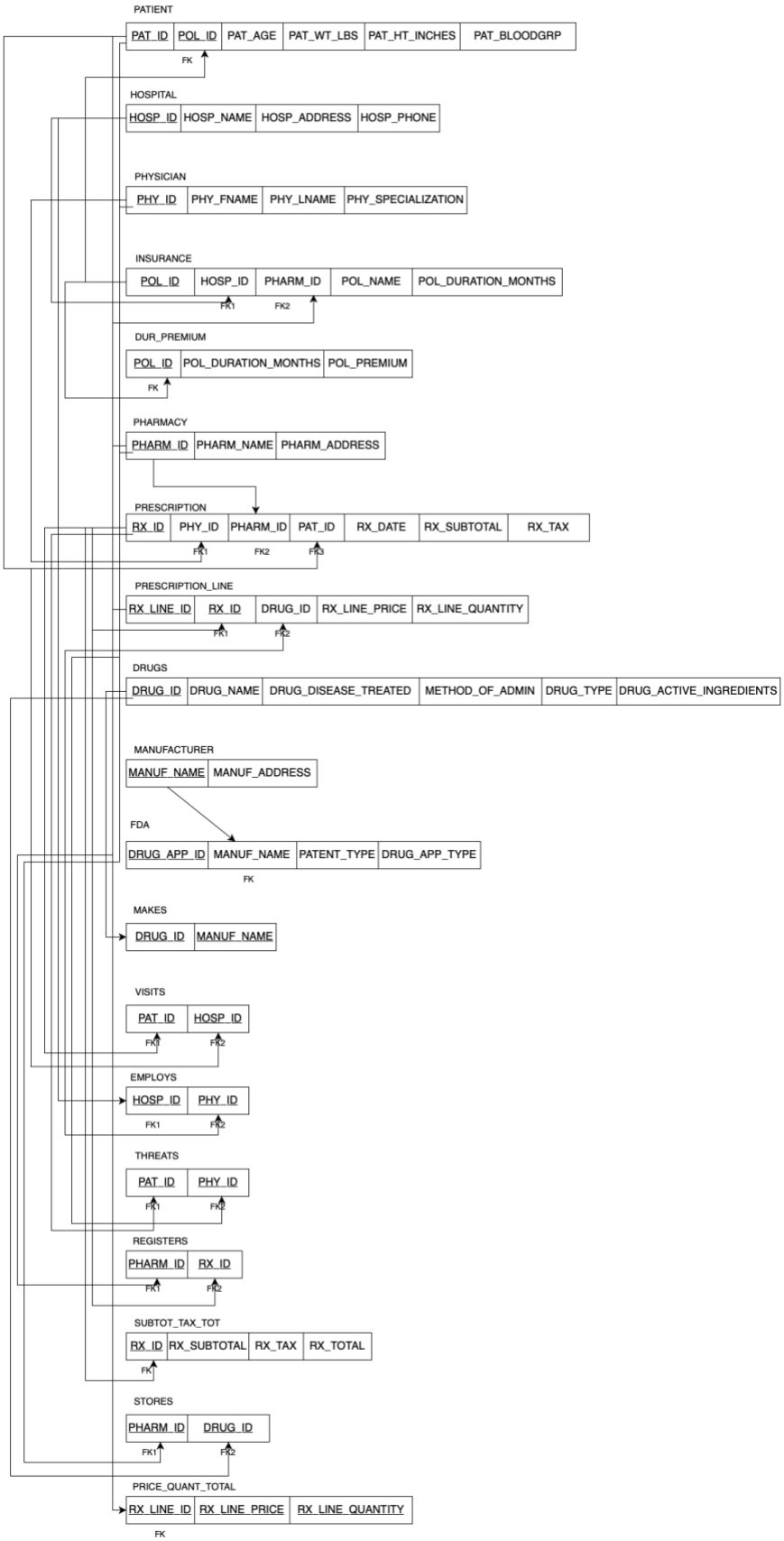
To ensure data consistency and integrity, the database will need to incorporate various mechanisms such as validation checks, constraints, and triggers to prevent incorrect data entry or manipulation. Additionally, to ensure secure storage and maintenance of the data, the database will need to incorporate appropriate security measures, such as access control, encryption, and backup and recovery procedures.

Overall, developing a comprehensive database for the US healthcare sector is a complex task that requires a thorough understanding of the healthcare industry, its data requirements, and the regulatory environment surrounding it.

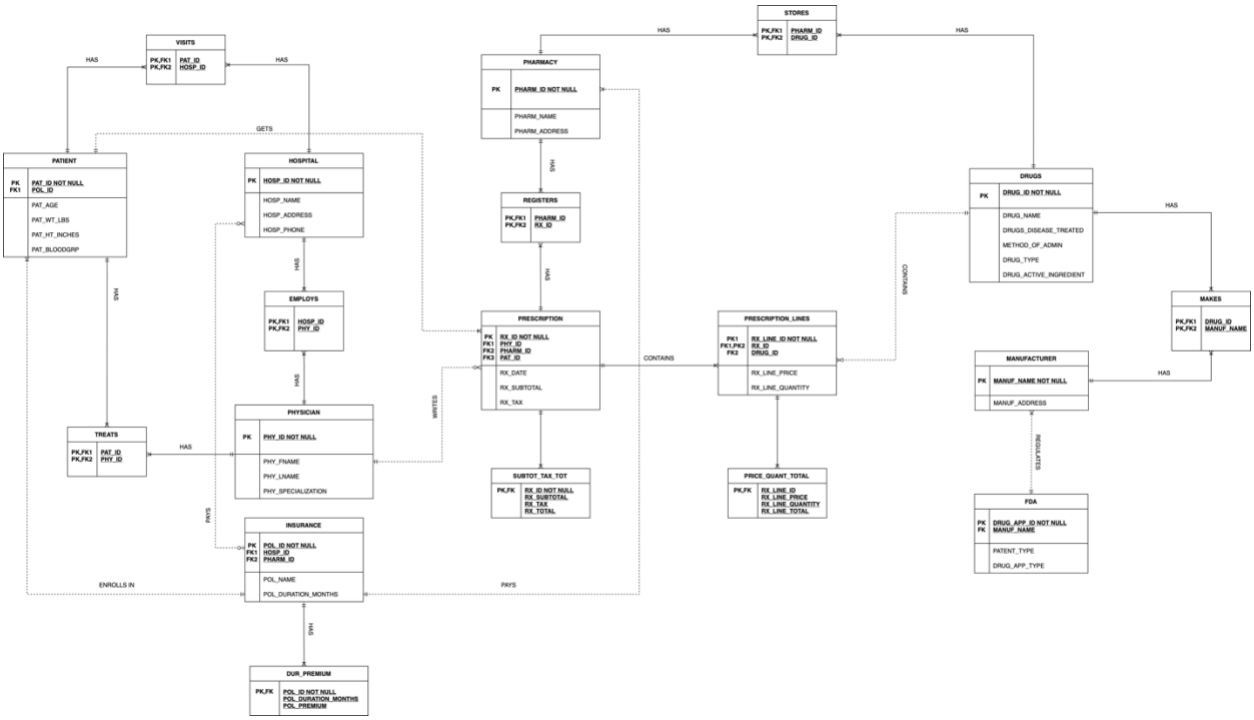
Part-2: Entity – Relationship Diagram

The Entity-Relationship diagram for a healthcare database system visually represents the different entities and relationships involved in the system. It shows the core entity, which is the PATIENT, and the other entities connected to it. Each entity is connected to other entities through relationships that show the nature of the connection between them. The diagram also includes attributes for each entity that provide additional details about the data stored within them. Overall, the Entity-Relationship diagram provides a clear and comprehensive view of the healthcare database system, making it easier to understand the relationships between the different entities and the data stored within them. It serves as a useful reference for database administrators, developers, and users to design, maintain, and use the database effectively.

RELATIONAL SCHEMA



Part-3: Corresponding Relational Schema



Part-4: Corresponding DB Implementation

In this project we have created total of 19 tables. Below we have given brief about each of the table. We have written each create statements on our own and not used forward engineering.

Hospital table: The hospital table will contain information about each hospital or medical center. The hospital_id column will be the primary key, and each hospital will be identified by a unique ID.

Drugs table: The drugs table will contain information about each drug or medication. The drug_id column will be the primary key, and each drug will be identified by a unique ID.

Pharmacy table: The pharmacy table will contain information about each pharmacy or drugstore. The pharmacy_id column will be the primary key, and each pharmacy will be identified by a unique ID.

Insurance table: This table will store information about the different insurance providers in the healthcare industry. Insurance ID tells us about a unique identifier for each insurance provider.Insurance Name gives information about the name of the insurance provider.

Patient table: This table will store information about patients in the healthcare industry. Patient ID is a unique identifier for each patient and insurance ID is a foreign key linking to the insurance provider for the patient.

Physician table: This table will store information about physicians in the healthcare industry. Physician ID is a unique identifier for each physician.

Manufacturers table: This table will store information about the different drug manufacturers in the healthcare industry. Manufacturer ID is a unique identifier for each manufacturer.

FDA table: This table will store information about the different drugs approved by the Food and Drug Administration (FDA). Drug ID is a unique identifier for each drug and Drug Name is the name of the drug.

Prescription table: This table will store information about prescriptions given to patients by physicians. Prescription ID is a unique identifier for each prescription and Patient ID is a foreign key linking to the patient the prescription was given to.

Prescription Lines Table: This table will store information about the different drug in each lines. RX_Line is the unique and RX_ID and drug_ID.

Stores Table: This table will store information about pharmacies or other healthcare providers that dispense medications.

Visits Table: This table will store information about patient visits to healthcare and hospital ID.

Treats Table: This table will store information about the treatments prescribed during patient visits.

Registers Table: This table will store information about the registration of patients with healthcare providers.

Employees Table: This table will store information about the employees working in the healthcare industry, including physicians.

Drug_Premium Table: This table will store information about premium drugs, including policy ID, policy premium.

Price_Quant_Total Table: This table will store information about the total price and quantity of a drug.

Makes Table: This table will store information about the drugs manufactured by different manufacturers.

Below screenshots shows creation of tables in MySQL Workbench.

```
CREATE SCHEMA HEALTHCARE;
USE HEALTHCARE;

CREATE TABLE HOSPITAL(
   HOSP_ID INT PRIMARY KEY auto_increment,
    HOSP_NAME VARCHAR(20),
    HOSP_ADDRESS VARCHAR(50),
    HOSP_PHONE BIGINT(13)
);

CREATE TABLE DRUGS(
   DRUG_ID INT PRIMARY KEY auto_increment,
    DRUG_NAME VARCHAR(100),
    DRUGS_DISEASE_TREATED VARCHAR(100),
    METHOD_OF_ADMIN VARCHAR(100),
```

```
    DRUG_TYPE VARCHAR(100),
    DRUG_ACTIVE_INGREDIENT VARCHAR(100)
);

CREATE TABLE PHARMACY(
   PHARM_ID INT PRIMARY KEY auto_increment,
    PHARM_NAME VARCHAR(100),
    PHARM_ADDRESS VARCHAR(100)
);

CREATE TABLE INSURANCE(
   POL_ID INT PRIMARY KEY auto_increment,
    HOSP_ID INT,
    PHARM_ID INT,
    POL_NAME VARCHAR(100),
    POL_DURATION_MONTHS INT,
    FOREIGN KEY (HOSP_ID) REFERENCES HOSPITAL(HOSP_ID),
    FOREIGN KEY (PHARM_ID) REFERENCES PHARMACY(PHARM_ID)
);

CREATE TABLE PATIENT(
   PAT_ID INT PRIMARY KEY auto_increment,
    POL_ID INT,
    PAT_AGE INT,
    PAT_WT_LBS FLOAT,
    PAT_HT_INCHES FLOAT,
    PAT_BLOODGRP VARCHAR(4),
    FOREIGN KEY (POL_ID) REFERENCES INSURANCE(POL_ID)
);

CREATE TABLE PHYSICIAN(
   PHY_ID INT PRIMARY KEY auto_increment,
    PHY_FNAME VARCHAR(50),
    PHY_LNAME VARCHAR(50),
    PHY_SPECIALIZATION VARCHAR(100)
);

CREATE TABLE MANUFACTURER(
   MANUF_NAME VARCHAR(100) PRIMARY KEY,
    MANUF_ADDRESS VARCHAR(100)
);

CREATE TABLE FDA(
   DRUG_APP_ID INT PRIMARY KEY auto_increment,
    MANUF_NAME VARCHAR(100),
    PATENT_TYPE VARCHAR(100),
    DRUG_APP_TYPE VARCHAR(100),
    FOREIGN KEY (MANUF_NAME) REFERENCES MANUFACTURER(MANUF_NAME)
);

CREATE TABLE PRESCRIPTION(
   RX_ID INT PRIMARY KEY auto_increment,
    PHY_ID INT,
    PHARM_ID INT,
    PAT_ID INT,
    RX_DATE DATE,
    RX_SUBTOTAL FLOAT,
```

```
    RX_TAX FLOAT,
    FOREIGN KEY (PHY_ID) REFERENCES PHYSICIAN(PHY_ID),
    FOREIGN KEY (PHARM_ID) REFERENCES PHARMACY(PHARM_ID),
    FOREIGN KEY (PAT_ID) REFERENCES PATIENT(PAT_ID)
);

CREATE TABLE PRESCRIPTION_LINES(
   RX_LINE_ID INT PRIMARY KEY auto_increment,
    RX_ID INT,
    DRUG_ID INT,
    RX_LINE_PRICE FLOAT,
    RX_LINE_QUANTITY INT,
    FOREIGN KEY (DRUG_ID) REFERENCES DRUGS(DRUG_ID),
    FOREIGN KEY (RX_ID) REFERENCES PRESCRIPTION(RX_ID)
);

CREATE TABLE STORES(
   PHARM_ID INT,
    DRUG_ID INT,
    PRIMARY KEY(PHARM_ID, DRUG_ID),
    FOREIGN KEY (DRUG_ID) REFERENCES DRUGS(DRUG_ID),
    FOREIGN KEY (PHARM_ID) REFERENCES PHARMACY(PHARM_ID)
);

CREATE TABLE VISITS(
   PAT_ID INT,
    HOSP_ID INT,
    PRIMARY KEY(PAT_ID, HOSP_ID),
    FOREIGN KEY (PAT_ID) REFERENCES PATIENT(PAT_ID),
    FOREIGN KEY (HOSP_ID) REFERENCES HOSPITAL(HOSP_ID)
);

CREATE TABLE TREATS(
   PAT_ID INT,
    PHY_ID INT,
    PRIMARY KEY(PAT_ID, PHY_ID),
    FOREIGN KEY (PAT_ID) REFERENCES PATIENT(PAT_ID),
    FOREIGN KEY (PHY_ID) REFERENCES PHYSICIAN(PHY_ID)
);

CREATE TABLE EMPLOYS(
   HOSP_ID INT,
    PHY_ID INT,
    PRIMARY KEY(HOSP_ID, PHY_ID),
    FOREIGN KEY (HOSP_ID) REFERENCES HOSPITAL(HOSP_ID),
    FOREIGN KEY (PHY_ID) REFERENCES PHYSICIAN(PHY_ID)
);

CREATE TABLE DUR_PREMIUM(
   POL_ID INT PRIMARY KEY,
    POL_DURATION_MONTHS INT,
    POL_PREMIUM FLOAT,
    FOREIGN KEY (POL_ID) REFERENCES INSURANCE(POL_ID)
);

CREATE TABLE REGISTERS(
   PHARM_ID INT,
```

```
    RX_ID INT,
    PRIMARY KEY(RX_ID, PHARM_ID),
    FOREIGN KEY (RX_ID) REFERENCES PRESCRIPTION(RX_ID),
    FOREIGN KEY (PHARM_ID) REFERENCES PHARMACY(PHARM_ID)
);

CREATE TABLE SUBTOT_TAX_TOT(
   RX_ID INT PRIMARY KEY,
   RX_SUBTOTAL FLOAT,
    RX_TAX FLOAT,
    RX_TOTAL FLOAT,
    FOREIGN KEY (RX_ID) REFERENCES PRESCRIPTION(RX_ID)
);

CREATE TABLE PRICE_QUANT_TOTAL(
   RX_LINE_ID INT PRIMARY KEY,
    RX_LINE_PRICE FLOAT,
    RX_LINE_QUANTITY INT,
    RX_LINE_TOTAL FLOAT,
    FOREIGN KEY (RX_LINE_ID) REFERENCES PRESCRIPTION_LINES(RX_LINE_ID)
);

CREATE TABLE MAKES(
   DRUG_ID INT,
    MANUF_NAME VARCHAR(100),
    PRIMARY KEY(DRUG_ID, MANUF_NAME),
    FOREIGN KEY (DRUG_ID) REFERENCES DRUGS(DRUG_ID),
    FOREIGN KEY (MANUF_NAME) REFERENCES MANUFACTURER(MANUF_NAME)
);
```

Part-5: Initial Population of DB

In order to populate a database with data, insert statements are used to add new information to existing tables.

The process of inserting data is essential to building a functional database, as it allows for the storage and retrieval of information in a structured format.

```
INSERT INTO HOSPITAL(HOSP_NAME, HOSP_ADDRESS, HOSP_PHONE) VALUES("Hospital
A","New Bedford", FLOOR( RAND() * 10000000000));
INSERT INTO HOSPITAL(HOSP_NAME, HOSP_ADDRESS, HOSP_PHONE) VALUES("Hospital
B","Boston", FLOOR( RAND() * 10000000000));
INSERT INTO HOSPITAL(HOSP_NAME, HOSP_ADDRESS, HOSP_PHONE) VALUES("Hospital
C","Fair Haven", FLOOR( RAND() * 10000000000));
INSERT INTO HOSPITAL(HOSP_NAME, HOSP_ADDRESS, HOSP_PHONE) VALUES("Hospital
D","Fall River", FLOOR( RAND() * 10000000000));
INSERT INTO HOSPITAL(HOSP_NAME, HOSP_ADDRESS, HOSP_PHONE) VALUES("Hospital
E","Providence", FLOOR( RAND() * 10000000000));

SELECT * FROM HOSPITAL;

INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Ventolin", "Asthma", "Orally
inhaled", "Branded", "Salbutamol");
```

```
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Lisinopril", "High Blood
Pressure", "Orally ingested", "Generic", "Prinivil");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Proair", "Asthma", "Orally
inhaled", "Branded", "Albuterol Sulfate");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Azithromycin", "Bacterial
Infections", "Orally ingested", "Generic", "Azithromycin Dihydrate");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Omeprazole", "Acid Reflux",
"Orally ingested", "Generic", "Omeprazole Magnesium");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Losartan Potassium", "High Blood
Pressure", "Orally ingested", "Generic", "Hydrochlorothiazide");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Sertraline", "Depression", "Orally
ingested", "Generic", "Sertraline Hydrochloride");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Amoxicillin", "Bacterial
Infections", "Injection", "Generic", "Amoxicillin Trihydrate");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Gabapentin", "Seizures", "Orally
ingested", "Generic", "Gabapentin");
INSERT INTO DRUGS(DRUG_NAME, DRUGS_DISEASE_TREATED, METHOD_OF_ADMIN,
DRUG_TYPE, DRUG_ACTIVE_INGREDIENT) VALUES("Basaglar", "Diabetes",
"Injection", "Branded", "Insulin Glargine");

SELECT * FROM DRUGS;

INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES("CVS Health", "New
Bedford");
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES("Walgreens Boots
Alliance", "Dartmouth") ;
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES ("Cigna", "Boston") ;
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES ("UnitedHealth
Group", "Fair Haven") ;
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES ("Walmart",
"Providence") ;
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES ("Kroger", "New
Bedford") ;
INSERT INTO PHARMACY (PHARM_NAME, PHARM_ADDRESS) VALUES("Rite Aid", "Fall
River");

SELECT * FROM PHARMACY;

INSERT INTO INSURANCE(HOSP_ID, PHARM_ID, POL_NAME, POL_DURATION_MONTHS)
VALUES(1, 7, "POS", 3);
INSERT INTO INSURANCE(HOSP_ID, PHARM_ID, POL_NAME, POL_DURATION_MONTHS)
VALUES(2, 6, "PPO", 5);
INSERT INTO INSURANCE(HOSP_ID, PHARM_ID, POL_NAME, POL_DURATION_MONTHS)
VALUES(3, 5, "HMO", 7);
INSERT INTO INSURANCE(HOSP_ID, PHARM_ID, POL_NAME, POL_DURATION_MONTHS)
VALUES(4, 4, "EPO", 12);

SELECT * FROM INSURANCE;
```

```
INSERT INTO DUR_PREMIUM VALUES (1, 3, 1500.00);
INSERT INTO DUR_PREMIUM VALUES (2, 5, 1200.00);
INSERT INTO DUR_PREMIUM VALUES (3, 7, 800.00);
INSERT INTO DUR_PREMIUM VALUES (4, 12, 500.00);

SELECT * FROM DUR_PREMIUM;

INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(2, 35, 189, 68, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(2, 28, 168, 70, "O+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 60, 171, 72, "AB+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 73, 140, 75, "O+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(4, 18, 162, 69, "AB-");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 94, 185, 70, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 46, 201, 66, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 58, 193, 63, "O-");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 83, 213, 70, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(3, 26, 169, 71, "B-");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(2, 39, 188, 75, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 50, 200, 68, "A+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 59, 150, 67, "B+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(4, 20, 196, 71, "O+");
INSERT INTO PATIENT(POL_ID, PAT_AGE, PAT_WT_LBS, PAT_HT_INCHES, PAT_BLOODGRP)
VALUES(1, 57, 250, 73, "B+");

SELECT * FROM PATIENT;

INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Jim", "Halpert", "Pediatrician");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Michael", "Scott", "Orthopedic");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Dwight", "Schrute", "Neurologist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Mose", "Schrute", "Dermatologist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Creed", "Bratton", "Psychiatrist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Pam", "Beesly", "OB/GYN");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Stanley", "Hudson", "Cardiologist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Jon", "Snow", "Radiologist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
```

```
("Thomas", "Shelby", "Oncologist");
INSERT INTO PHYSICIAN(PHY_FNAME, PHY_LNAME, PHY_SPECIALIZATION) VALUES
("Jesse", "Pinkman", "Anesthesiologist");

SELECT * FROM PHYSICIAN;

INSERT INTO MANUFACTURER VALUES ("GSK", "New Bedford");
INSERT INTO MANUFACTURER VALUES ("TEVA", "Boston");
INSERT INTO MANUFACTURER VALUES ("Eli Lilly and Company", "Fairhaven");
INSERT INTO MANUFACTURER VALUES ("Lupin", "Dartmouth");
INSERT INTO MANUFACTURER VALUES ("Zydus", "Fall River");

SELECT * FROM MANUFACTURER;

INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("GSK", "AIP",
"NDA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("GSK", "FP",
"BLA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("TEVA", "AIP",
"NDA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Lupin", "FP",
"BLA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("TEVA", "FP",
"NDA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Eli Lilly and
Company", "AIP", "BLA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Zydus", "FP",
"BLA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Lupin", "FP",
"NDA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Zydus",
"AIP", "BLA");
INSERT INTO FDA(MANUF_NAME, PATENT_TYPE, DRUG_APP_TYPE) VALUES("Eli Lilly and
Company", "FP", "NDA");

SELECT * FROM FDA;

INSERT INTO PRESCRIPTION(PHY_ID, PHARM_ID, PAT_ID, RX_DATE, RX_TAX)
VALUES(FLOOR(RAND()*9+1), FLOOR(RAND()*6+1), FLOOR(RAND()*14+1), curdate(),
6.25);
INSERT INTO PRESCRIPTION(PHY_ID, PHARM_ID, PAT_ID, RX_DATE, RX_TAX)
VALUES(FLOOR(RAND()*9+1), FLOOR(RAND()*6+1), FLOOR(RAND()*14+1), curdate(),
6.25);
INSERT INTO PRESCRIPTION(PHY_ID, PHARM_ID, PAT_ID, RX_DATE, RX_TAX)
VALUES(FLOOR(RAND()*9+1), FLOOR(RAND()*6+1), FLOOR(RAND()*14+1), curdate(),
6.25);
INSERT INTO PRESCRIPTION(PHY_ID, PHARM_ID, PAT_ID, RX_DATE, RX_TAX)
VALUES(FLOOR(RAND()*9+1), FLOOR(RAND()*6+1), FLOOR(RAND()*14+1), curdate(),
6.25);
INSERT INTO PRESCRIPTION(PHY_ID, PHARM_ID, PAT_ID, RX_DATE, RX_TAX)
VALUES(FLOOR(RAND()*9+1), FLOOR(RAND()*6+1), FLOOR(RAND()*14+1), curdate(),
6.25);

SELECT * FROM PRESCRIPTION;

INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(1, 10, 70.0, 30);
```

```
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(1, 9, 63.0, 60);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(2, 8, 56.0, 30);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(2, 7, 48.99, 90);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(3, 6, 42.0, 30);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(3, 5, 34.99, 60);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(4, 4, 27.5, 90);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(4, 3, 20.99, 30);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(5, 2, 14.0, 30);
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
RX_LINE_QUANTITY) VALUES(5, 1, 6.99, 60);

SELECT * FROM PRESCRIPTION_LINES;

INSERT INTO MAKES VALUES (1, "Eli Lilly and Company");
INSERT INTO MAKES VALUES (2, "Lupin");
INSERT INTO MAKES VALUES (3, "GSK");
INSERT INTO MAKES VALUES (4,"TEVA");
INSERT INTO MAKES VALUES (5, "GSK");
INSERT INTO MAKES VALUES (6, "Zydus");
INSERT INTO MAKES VALUES (7, "Eli Lilly and Company");
INSERT INTO MAKES VALUES (8, "Zydus");
INSERT INTO MAKES VALUES (9, "Lupin");
INSERT INTO MAKES VALUES (10, "TEVA");

SELECT * FROM MAKES;

INSERT INTO STORES VALUES (7, 1);
INSERT INTO STORES VALUES (6, 2);
INSERT INTO STORES VALUES (5, 3);
INSERT INTO STORES VALUES (4, 4);
INSERT INTO STORES VALUES (3, 5);
INSERT INTO STORES VALUES (2, 6);
INSERT INTO STORES VALUES (1, 7);
INSERT INTO STORES VALUES (5, 8);
INSERT INTO STORES VALUES (7, 9);
INSERT INTO STORES VALUES (3, 10);

SELECT * FROM STORES;

INSERT INTO VISITS VALUES (1, 1);
INSERT INTO VISITS VALUES (2, 2);
INSERT INTO VISITS VALUES (3, 3);
INSERT INTO VISITS VALUES (4, 4);
INSERT INTO VISITS VALUES (5, 5);
INSERT INTO VISITS VALUES (6, 1);
INSERT INTO VISITS VALUES (7, 2);
INSERT INTO VISITS VALUES (8, 3);
INSERT INTO VISITS VALUES (9, 4);
INSERT INTO VISITS VALUES (10, 5);
```

```
INSERT INTO VISITS VALUES (11, 1);
INSERT INTO VISITS VALUES (12, 2);
INSERT INTO VISITS VALUES (13, 3);
INSERT INTO VISITS VALUES (14, 4);
INSERT INTO VISITS VALUES (15, 5);

SELECT * FROM VISITS;

INSERT INTO TREATS VALUES (1, 1);
INSERT INTO TREATS VALUES (2, 2);
INSERT INTO TREATS VALUES (3, 3);
INSERT INTO TREATS VALUES (4, 4);
INSERT INTO TREATS VALUES (5, 5);
INSERT INTO TREATS VALUES (6, 10);
INSERT INTO TREATS VALUES (7, 9);
INSERT INTO TREATS VALUES (8, 8);
INSERT INTO TREATS VALUES (9, 7);
INSERT INTO TREATS VALUES (10, 6);
INSERT INTO TREATS VALUES (11, 1);
INSERT INTO TREATS VALUES (12, 2);
INSERT INTO TREATS VALUES (13, 3);
INSERT INTO TREATS VALUES (14, 4);
INSERT INTO TREATS VALUES (15, 5);

SELECT * FROM TREATS;

INSERT INTO REGISTERS VALUES (5, 1);
INSERT INTO REGISTERS VALUES (7, 2);
INSERT INTO REGISTERS VALUES (2, 3);
INSERT INTO REGISTERS VALUES (3, 4);
INSERT INTO REGISTERS VALUES (1, 5);

SELECT * FROM REGISTERS;

INSERT INTO EMPLOYS VALUES (1, 1);
INSERT INTO EMPLOYS VALUES (1, 2);
INSERT INTO EMPLOYS VALUES (2, 3);
INSERT INTO EMPLOYS VALUES (2, 4);
INSERT INTO EMPLOYS VALUES (3, 5);
INSERT INTO EMPLOYS VALUES (3, 6);
INSERT INTO EMPLOYS VALUES (4, 7);
INSERT INTO EMPLOYS VALUES (4, 8);
INSERT INTO EMPLOYS VALUES (5, 9);
INSERT INTO EMPLOYS VALUES (5, 10);

SELECT * FROM EMPLOYS;
```

Part-6: Initial Demonstration of DB Querying

1. This SQL query retrieves data from three tables - PATIENT, INSURANCE, and DUR_PREMIUM - using inner join to connect them together. It selects the patient's age and their corresponding insurance policy's premium amount, and categorizes the policy premium as high, medium, or low based on certain conditions. The resulting output will display the patient's age, policy premiums, and their corresponding premium profile category. This query could be useful

in analyzing patient demographics and insurance policy premiums for further research or business intelligence purposes.

```
SELECT P.PAT_AGE AS 'Patient\'s Age', DP.POL_PREMIUM AS 'Policy Premiums',
CASE
   WHEN DP.POL_PREMIUM > 1200 THEN 'High'
    WHEN DP.POL_PREMIUM > 800 AND DP.POL_PREMIUM <= 1200 THEN 'Medium'
    WHEN DP.POL_PREMIUM <= 800 THEN 'Low'
END AS 'Premium Profile'
FROM PATIENT AS P
INNER JOIN INSURANCE AS I ON P.POL_ID = I.POL_ID
INNER JOIN DUR_PREMIUM AS DP ON I.POL_ID = DP.POL_ID;
```

2. This SQL query retrieves data from four tables - PATIENT, PRESCRIPTION, PRESCRIPTION_LINES, and DRUGS - using inner join to connect them together. It selects patient IDs and their corresponding diseases based on the drugs that were prescribed to them, which are linked through the prescription lines. The resulting output will display the patient IDs and the diseases they were treated for based on the prescribed drugs. This query could be useful in tracking the types of diseases that patients are being treated for, and identifying any trends or patterns in prescribed medications.

```
SELECT P.PAT_ID AS 'Patient IDs', DR.DRUGS_DISEASE_TREATED AS 'Diseases'
FROM PATIENT AS P
INNER JOIN PRESCRIPTION AS PRES ON P.PAT_ID = PRES.PAT_ID
INNER JOIN PRESCRIPTION_LINES AS PRES_LINE ON PRES.RX_ID = PRES_LINE.RX_ID
INNER JOIN DRUGS AS DR ON PRES_LINE.DRUG_ID = DR.DRUG_ID;
```

3. This SQL query retrieves data from two tables - DRUGS and PRESCRIPTION_LINES - using a natural join to connect them together. It selects the distinct method of administration for drugs, and counts the number of times each method appears in the prescription lines. It then groups the results by method of administration, orders the results by the count of prescription lines in descending order, and limits the output to the highest count. The resulting output will display the method of administration that is most commonly used in prescription lines. This query could be useful in identifying the most frequently used method of drug administration, which could inform healthcare providers and policymakers in their decision-making processes.

```
SELECT DISTINCT METHOD_OF_ADMIN, COUNT(METHOD_OF_ADMIN)
FROM DRUGS
NATURAL JOIN PRESCRIPTION_LINES
GROUP BY METHOD_OF_ADMIN
ORDER BY COUNT(RX_LINE_PRICE) DESC
LIMIT 1;
```

Part-7: Stored Procedure and Embedded Query

This SQL code creates a stored procedure named insert_drugs, which takes six input parameters for inserting new records into the DRUGS table. The procedure executes an insert statement that adds a new record to the DRUGS table with the values provided by the input parameters. The subsequent call statement calls the insert_drugs procedure with specific parameter values to insert a new drug record into the table. Finally, the SELECT statement is used to retrieve all records from the DRUGS table, which will include the newly inserted record. This code could be useful for

inserting new drugs into the database system as needed, such as when a new drug becomes available for treating a certain disease.

```
CREATE PROCEDURE insert_drugs(
    IN DRUG_ID INT,
    IN DRUG_NAME VARCHAR(100),
    IN DRUGS_DISEASE_TREATED VARCHAR(100),
    IN METHOD_OF_ADMIN VARCHAR(100),
    IN DRUG_TYPE VARCHAR(100),
    IN DRUG_ACTIVE_INGREDIENT VARCHAR(100)
)

BEGIN
    INSERT INTO DRUGS VALUES (DRUG_ID, DRUG_NAME, DRUGS_DISEASE_TREATED,
METHOD_OF_ADMIN, DRUG_TYPE, DRUG_ACTIVE_INGREDIENT);
END;

call insert_drugs(
    12,
    "Proair_4",
    "Asthma",
    "Orally inhaled",
    "Branded",
    "Insulin Glargine"
);

SELECT * FROM DRUGS;
```

This SQL code creates a trigger named Prescription_trigger that is executed automatically after every new record is inserted into the PRESCRIPTION_LINES table. The trigger inserts a new record into the PRICE_QUANT_TOTAL table, which calculates the total price of the prescription line by multiplying the RX_LINE_PRICE by the RX_LINE_QUANTITY.

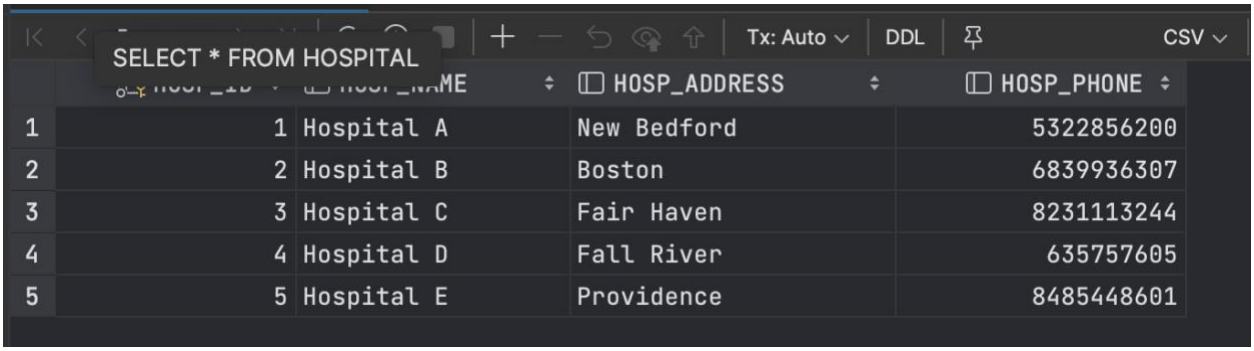
The subsequent INSERT statement adds a new record to the PRESCRIPTION_LINES table with specific values for RX_ID, DRUG_ID, RX_LINE_PRICE, and RX_LINE_QUANTITY. This will trigger the Prescription_trigger trigger, which in turn inserts a new record into the PRICE_QUANT_TOTAL table.

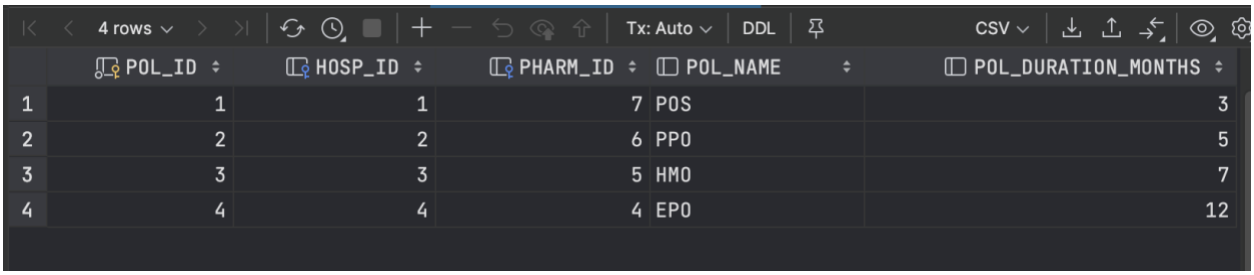
Finally, the SELECT statement is used to retrieve all records from the PRICE_QUANT_TOTAL table, which will include the newly inserted record. This code could be useful in tracking the total prices of prescription lines and analyzing the costs associated with different medications or treatments.

```
DELIMITER ||
CREATE TRIGGER Prescription_trigger AFTER INSERT ON PRESCRIPTION_LINES
    FOR EACH ROW
    BEGIN
        INSERT INTO PRICE_QUANT_TOTAL VALUES (NEW.RX_LINE_ID,
NEW.RX_LINE_PRICE, NEW.RX_LINE_QUANTITY,
NEW.RX_LINE_PRICE*NEW.RX_LINE_QUANTITY);
    END
||

INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE,
```

```
RX_LINE_QUANTITY) VALUES(1,7,200,30);

SELECT * FROM PRICE_QUANT_TOTAL;
```

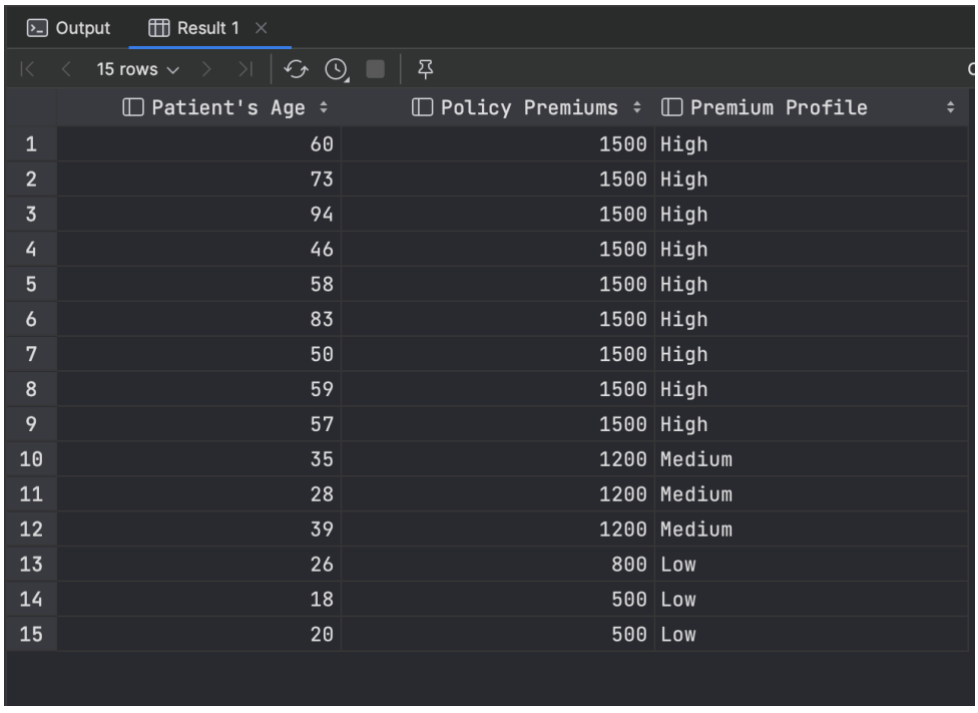
Results:





Query results:

1.



2.

Output | Result 2

10 rows

	`Patient IDs`	Diseases
1	5	Bacterial Infections
2	5	Depression
3	5	Bacterial Infections
4	5	Asthma
5	6	Diabetes
6	6	Seizures
7	12	High Blood Pressure
8	12	Acid Reflux
9	12	High Blood Pressure
10	12	Asthma

3.

1 row

	METHOD_OF_ADMIN	`COUNT(METHOD_OF_ADMIN)`
1	Orally ingested	6

Trigger output:

```
INSERT INTO PRESCRIPTION_LINES(RX_ID, DRUG_ID, RX_LINE_PRICE, RX_LINE_QUANTITY) VALUES(1,7,200,30);

SELECT * FROM PRICE_QUANT_TOTAL;
```

Output | HEALTHCARE.PRICE_QUANT_TOTAL

11 rows | Tx: Auto | DDL | CSV

	RX_LINE_ID	RX_LINE_PRICE	RX_LINE_QUANTITY	RX_LINE_TOTAL
1	2	70	30	2100
2	3	63	60	3780
3	4	56	30	1680
4	5	48.99	90	4409.1
5	6	42	30	1260
6	7	34.99	60	2099.4
7	8	27.5	90	2475
8	9	20.99	30	629.7
9	10	14	30	420
10	11	6.99	60	419.4
11	12	200	30	6000

Procedure output:

```
call insert_drugs(
      DRUG_ID: 12,
      DRUG_NAME: "Proair_4",
      DRUGS_DISEASE_TREATED: "Asthma",
      METHOD_OF_ADMIN: "Orally inhaled",
      DRUG_TYPE: "Branded",
      DRUG_ACTIVE_INGREDIENT: "Insulin Glargine"
);

SELECT * FROM DRUGS;
```

Output | HEALTHCARE.DRUGS

11 rows | Tx: Auto | DDL | CSV

	DRUG_ID	DRUG_NAME	DRUGS_DISEASE_TREATED	METHOD_OF_ADMIN	DRUG_TYPE
1	1	Ventolin	Asthma	Orally inhaled	Branded
2	2	Lisinopril	High Blood Pressure	Orally ingested	Generic
3	3	Proair	Asthma	Orally inhaled	Branded
4	4	Azithromycin	Bacterial Infections	Orally ingested	Generic
5	5	Omeprazole	Acid Reflux	Orally ingested	Generic
6	6	Losartan Potassium	High Blood Pressure	Orally ingested	Generic
7	7	Sertraline	Depression	Orally ingested	Generic
8	8	Amoxicillin	Bacterial Infections	Injection	Generic
9	9	Gabapentin	Seizures	Orally ingested	Generic
10	10	Basaglar	Diabetes	Injection	Branded
11	12	Proair_4	Asthma	Orally inhaled	Branded