IFT 530: Advanced Database Management System

Team – 75

Sreeja Vaddi

Sai Krishna Samudrala

Final Project: Rental Car Management System

Due Date: 04th Dec 2024

**Creating the Tables**

**Tables we have used:**

1. **Customer\_Details:**

SQL Script:

CREATE TABLE Customer\_Details

(

FNAME VARCHAR(35) NOT NULL,

MNAME VARCHAR(20),

LNAME VARCHAR(35) NOT NULL,

EMAIL\_ID VARCHAR(50) NOT NULL,

PHONE\_NUMBER CHAR(12) NOT NULL,

STREET VARCHAR(50) NOT NULL,

CITY VARCHAR(30) NOT NULL,

STATE\_NAME VARCHAR(30) NOT NULL,

ZIPCODE CHAR(6) NOT NULL,

MEMBERSHIP\_ID CHAR(8),

MEMBERSHIP\_TYPE CHAR(1) DEFAULT 'N' NOT NULL,

DL\_NUMBER CHAR(10) NOT NULL,

CONSTRAINT CUSTOMERPK PRIMARY KEY (DL\_NUMBER)

);

1. **Car\_Category:**

SQL Script:

CREATE TABLE Car\_Category

(

CATEGORY\_NAME VARCHAR(30) NOT NULL,

COST\_PER\_DAY DECIMAL(7, 2) NOT NULL,

LATE\_FEE\_PER\_HOUR DECIMAL(6, 2) NOT NULL,

NO\_OF\_PERSON INTEGER NOT NULL,

NO\_OF\_LUGGAGE INTEGER NOT NULL,

CONSTRAINT CARCATEGORYPK PRIMARY KEY (CATEGORY\_NAME)

);

1. **Location\_Details:**

SQL Script:

CREATE TABLE Location\_Details

(

LOCATION\_ID CHAR(6) NOT NULL,

LOCATION\_NAME VARCHAR(60) NOT NULL,

CITY VARCHAR(30) NOT NULL,

STATE\_NAME VARCHAR(30) NOT NULL,

STREET VARCHAR(40) NOT NULL,

ZIPCODE CHAR(6) NOT NULL,

CONSTRAINT LOCATIONPK PRIMARY KEY (LOCATION\_ID)

);

1. **Car:**

SQL Script:

CREATE TABLE Car

(

REGISTRATION\_NUMBER CHAR(10) NOT NULL,

MODEL\_NAME VARCHAR(30) NOT NULL,

MAKE VARCHAR(30) NOT NULL,

CAR\_CATEGORY\_NAME VARCHAR(30) NOT NULL,

LOC\_ID CHAR(6) NOT NULL,

MODEL\_YEAR DECIMAL(4) NOT NULL,

MILEAGE INTEGER NOT NULL,

AVAILABILITY\_FLAG CHAR(1) NOT NULL,

CONSTRAINT CARPK PRIMARY KEY (REGISTRATION\_NUMBER),

CONSTRAINT CARFK1 FOREIGN KEY (CAR\_CATEGORY\_NAME) REFERENCES Car\_Category(CATEGORY\_NAME),

CONSTRAINT CARFK2 FOREIGN KEY (LOC\_ID) REFERENCES Location\_Details(LOCATION\_ID)

);

1. **Discount\_Details:**

SQL Script:

CREATE TABLE Discount\_Details

(

DISCOUNT\_CODE CHAR(6) NOT NULL,

DISCOUNT\_NAME VARCHAR(30) NOT NULL,

DISCOUNT\_PERCENTAGE DECIMAL(5, 2) NOT NULL,

EXPIRY\_DATE DATE NOT NULL,

CONSTRAINT DISCOUNTPK PRIMARY KEY (DISCOUNT\_CODE),

CONSTRAINT DISCOUNTSK UNIQUE (DISCOUNT\_NAME)

);

1. **Rental\_car\_Insurance:**

SQL Script:

CREATE TABLE Rental\_Car\_Insurance

(

INSURANCE\_CODE CHAR(6) NOT NULL,

INSURANCE\_NAME VARCHAR(60) NOT NULL,

COST\_PER\_DAY DECIMAL(6, 2) NOT NULL,

COVERAGE\_TYPE VARCHAR(250) NOT NULL,

CONSTRAINT INSURANCEPK PRIMARY KEY (INSURANCE\_CODE),

CONSTRAINT INSURANCESK UNIQUE (INSURANCE\_NAME)

);

1. **Booking\_Details:**

SQL Script:

CREATE TABLE Booking\_Details

(

BOOKING\_ID CHAR(5) NOT NULL,

REG\_NUM CHAR(10) NOT NULL,

DL\_NUM CHAR(10) NOT NULL,

AMOUNT DECIMAL(12, 2) NOT NULL,

BOOKING\_STATUS CHAR(1) NOT NULL,

PICKUP\_LOC CHAR(6) NOT NULL,

DROP\_LOC CHAR(6) NOT NULL,

INS\_CODE CHAR(6),

DISCOUNT\_CODE CHAR(6),

CONSTRAINT BOOKINGPK PRIMARY KEY (BOOKING\_ID),

CONSTRAINT BOOKINGFK1 FOREIGN KEY (PICKUP\_LOC) REFERENCES Location\_Details(LOCATION\_ID),

CONSTRAINT BOOKINGFK2 FOREIGN KEY (DROP\_LOC) REFERENCES Location\_Details(LOCATION\_ID),

CONSTRAINT BOOKINGFK3 FOREIGN KEY (REG\_NUM) REFERENCES Car(REGISTRATION\_NUMBER),

CONSTRAINT BOOKINGFK4 FOREIGN KEY (DL\_NUM) REFERENCES Customer\_Details(DL\_NUMBER),

CONSTRAINT BOOKINGFK5 FOREIGN KEY (INS\_CODE) REFERENCES Rental\_Car\_Insurance(INSURANCE\_CODE),

CONSTRAINT BOOKINGFK6 FOREIGN KEY (DISCOUNT\_CODE) REFERENCES Discount\_Details(DISCOUNT\_CODE)

);

1. **Billing\_Details:**

SQL Script:

CREATE TABLE Billing\_Details (

BILL\_ID CHAR(6) NOT NULL,

BILL\_DATE DATE NOT NULL,

BILL\_STATUS CHAR(1) NOT NULL,

DISCOUNT\_AMOUNT DECIMAL(10, 2) NOT NULL,

TOTAL\_AMOUNT DECIMAL(10, 2) NOT NULL,

TAX\_AMOUNT DECIMAL(10, 2) NOT NULL,

BOOKING\_ID CHAR(5) NOT NULL,

TOTAL\_LATE\_FEE DECIMAL(10, 2) NOT NULL,

CONSTRAINT BILLINGPK PRIMARY KEY (BILL\_ID),

CONSTRAINT BILLINGFK1 FOREIGN KEY (BOOKING\_ID) REFERENCES BOOKING\_DETAILS(BOOKING\_ID)

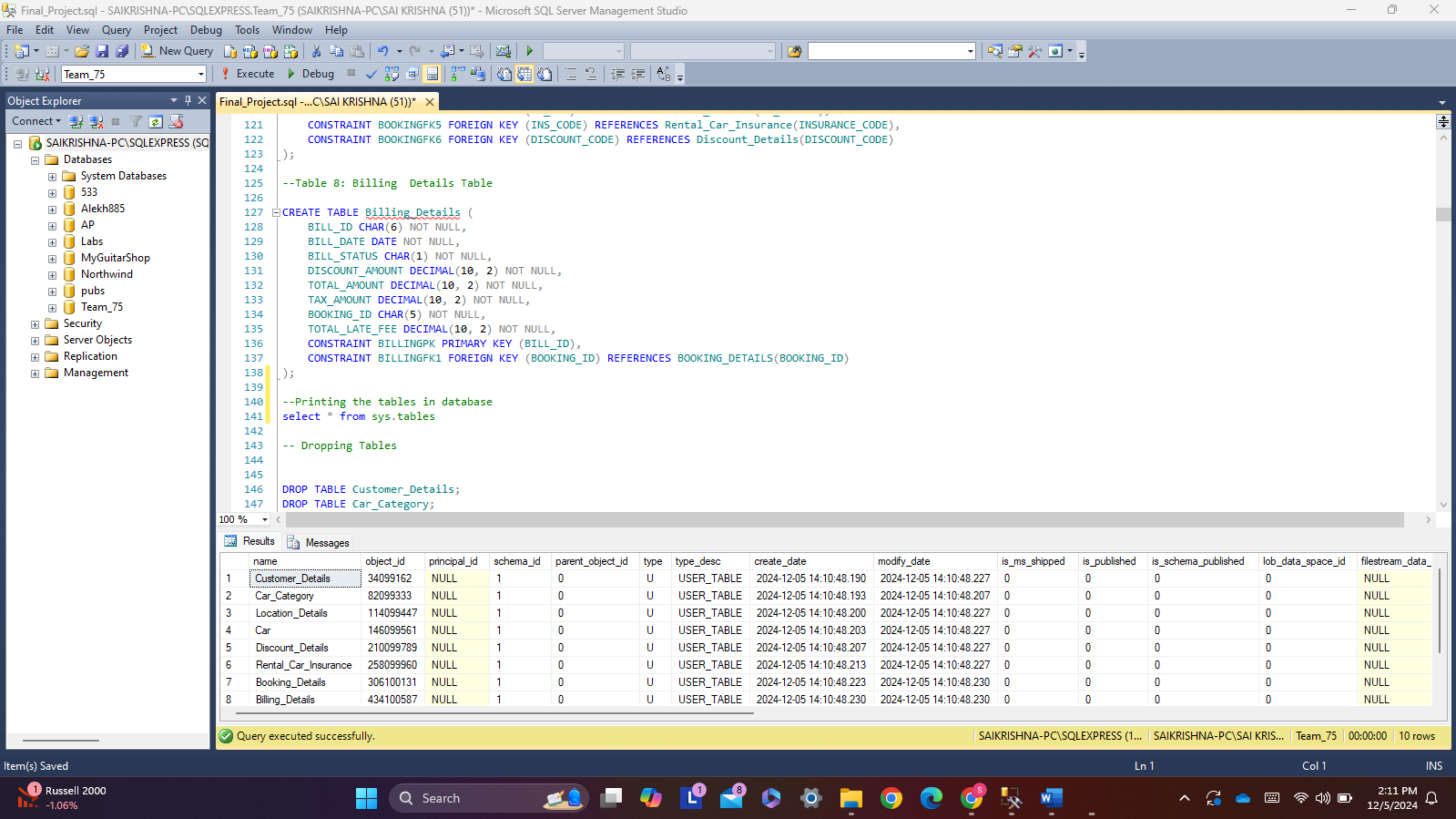
);

**Command to Show the tables created:**

**Script:**

select \* from sys.tables

**Output:**

****

**Command to Drop the tables:**

DROP TABLE Customer\_Details;

DROP TABLE Car\_Category;

DROP TABLE Location\_Details;

DROP TABLE Car;

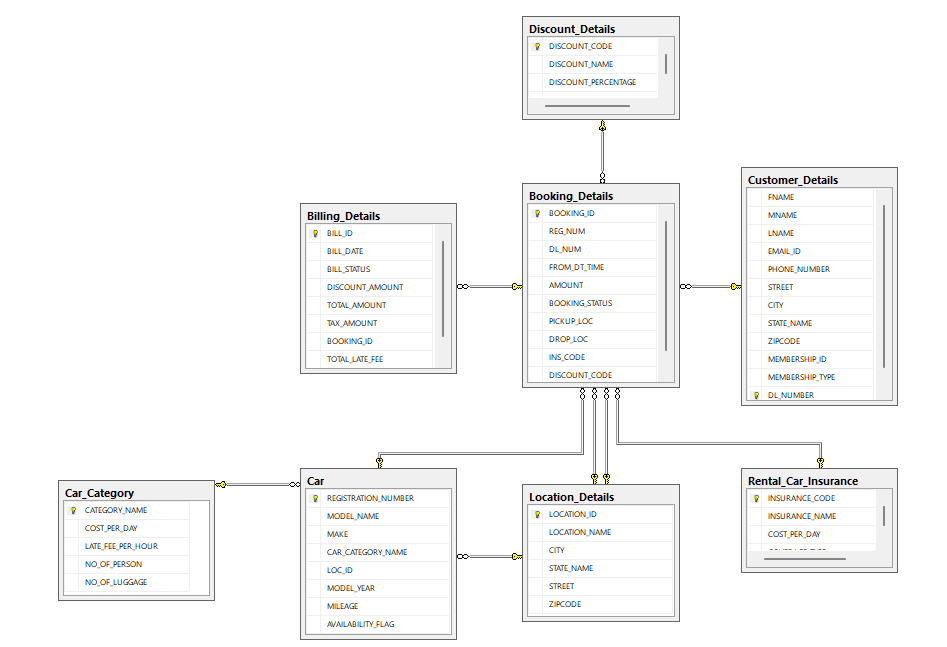
DROP TABLE Discount\_Details;

DROP TABLE Rental\_Car\_Insurance;

DROP TABLE Booking\_Details;

DROP TABLE Billing\_Details;

**Entity Relationship Diagram**



**Populating the Tables**

Script for Customer Table:

INSERT INTO Customer\_Details

(

FNAME, MNAME, LNAME, EMAIL\_ID, PHONE\_NUMBER, STREET, CITY, STATE\_NAME, ZIPCODE, MEMBERSHIP\_ID, MEMBERSHIP\_TYPE, DL\_NUMBER

)

VALUES

('John', 'A.', 'Doe', 'john.doe@example.com', '123-456-7890', '123 Elm St', 'Springfield', 'Illinois', '62701', 'MEM12345', 'Y', 'D123456789'),

('Jane', 'B.', 'Smith', 'jane.smith@example.com', '987-654-3210', '456 Oak St', 'Riverside', 'California', '92501', 'MEM54321', 'N', 'S987654321'),

('Michael', 'C.', 'Johnson', 'michael.johnson@example.com', '555-123-4567', '789 Pine St', 'Hometown', 'Texas', '75001', 'MEM67890', 'Y', 'M123456789'),

('Emily', 'D.', 'Williams', 'emily.williams@example.com', '444-321-9876', '101 Maple Ave', 'Weston', 'Florida', '33301', 'MEM09876', 'N', 'W987654321'),

('Daniel', 'E.', 'Brown', 'daniel.brown@example.com', '666-543-2109', '202 Birch Rd', 'Chattanooga', 'Tennessee', '37401', 'MEM24680', 'Y', 'B123456789'),

('Sophia', 'F.', 'Jones', 'sophia.jones@example.com', '333-765-4321', '303 Cedar Ln', 'Madison', 'Wisconsin', '53703', 'MEM13579', 'N', 'J987654321'),

('Lucas', 'G.', 'Miller', 'lucas.miller@example.com', '777-234-5678', '404 Pine Rd', 'Boulder', 'Colorado', '80301', 'MEM36912', 'Y', 'L123456789'),

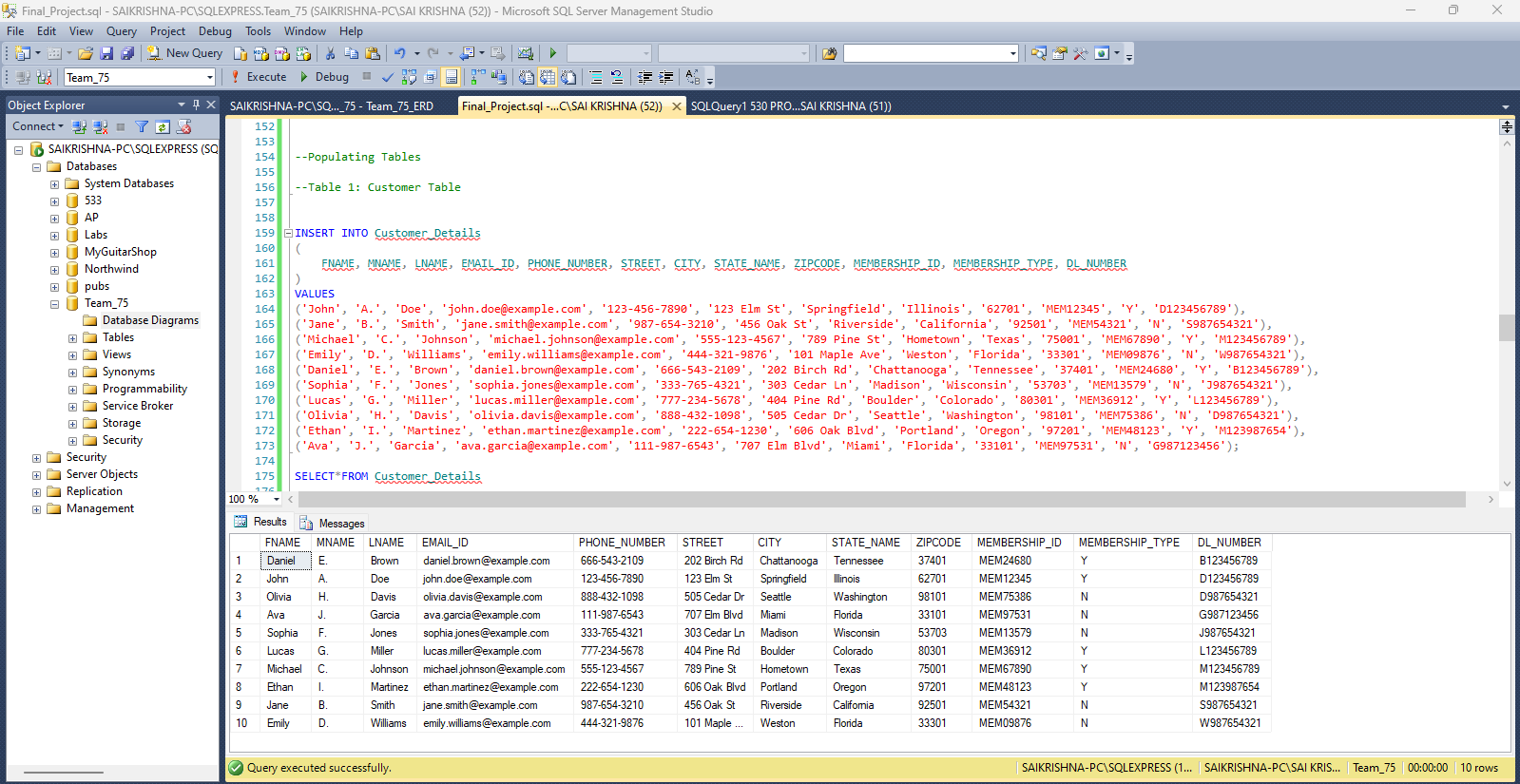
('Olivia', 'H.', 'Davis', 'olivia.davis@example.com', '888-432-1098', '505 Cedar Dr', 'Seattle', 'Washington', '98101', 'MEM75386', 'N', 'D987654321'),

('Ethan', 'I.', 'Martinez', 'ethan.martinez@example.com', '222-654-1230', '606 Oak Blvd', 'Portland', 'Oregon', '97201', 'MEM48123', 'Y', 'M123987654'),

('Ava', 'J.', 'Garcia', 'ava.garcia@example.com', '111-987-6543', '707 Elm Blvd', 'Miami', 'Florida', '33101', 'MEM97531', 'N', 'G987123456');

SELECT\*FROM Customer\_Details;

Output:



Script for Car Category Table:

INSERT INTO Car\_Category

(CATEGORY\_NAME, COST\_PER\_DAY, LATE\_FEE\_PER\_HOUR, NO\_OF\_PERSON, NO\_OF\_LUGGAGE)

VALUES

('Electric', 100.00, 20.00, 4, 2),

('Sedan', 60.00, 15.00, 4, 3),

('SUV', 80.00, 18.00, 5, 4),

('Compact', 50.00, 12.00, 4, 2),

('Hybrid', 75.00, 16.00, 4, 3),

('Truck', 120.00, 25.00, 2, 5),

('Minivan', 90.00, 20.00, 7, 3),

('Sports', 150.00, 30.00, 2, 3),

('Luxury', 200.00, 35.00, 4, 2),

('Convertible', 180.00, 28.00, 2, 2),

('Coupe', 70.00, 14.00, 4, 3),

('Crossover', 85.00, 17.00, 5, 4),

('Wagon', 95.00, 19.00, 5, 4),

('Hatchback', 65.00, 13.00, 4, 2),

('Midsize', 72.00, 16.00, 4, 3),

('Fullsize', 85.00, 18.00, 5, 4),

('Roadster', 130.00, 27.00, 2, 2),

('Luxury SUV', 150.00, 30.00, 5, 5),

('Executive', 160.00, 32.00, 4, 2),

('Off-road', 140.00, 29.00, 4, 5),

('Pickup', 110.00, 22.00, 2, 6),

('Minivan XL', 115.00, 23.00, 8, 4),

('Economy', 55.00, 11.00, 4, 2),

('Cargo Van', 120.00, 25.00, 2, 8),

('Subcompact', 48.00, 10.00, 4, 2),

('MPV', 95.00, 19.00, 7, 4),

('Pickup Extended', 125.00, 24.00, 3, 5),

('Sport Coupe', 135.00, 26.00, 2, 3),

('Grand Tourer', 170.00, 34.00, 4, 3),

('Utility Vehicle', 90.00, 20.00, 5, 5),

('City Car', 45.00, 9.00, 4, 1),

('Compact SUV', 78.00, 17.00, 5, 3),

('Luxury Sedan', 210.00, 40.00, 4, 3),

('Four-door Sedan', 65.00, 14.00, 5, 3),

('Coupe SUV', 95.00, 19.00, 4, 3),

('Convertible SUV', 170.00, 35.00, 4, 2),

('Touring Sedan', 120.00, 25.00, 4, 3),

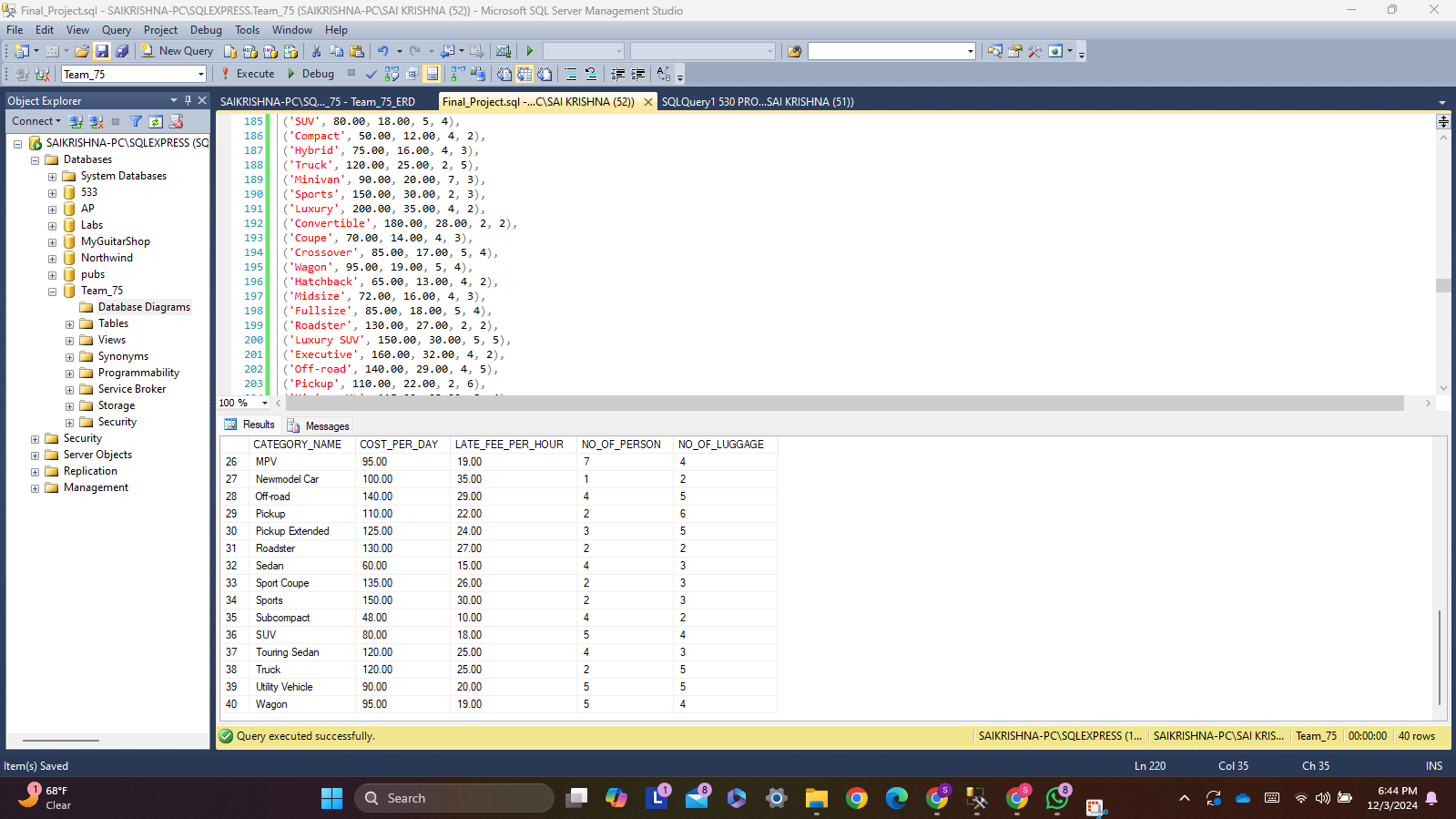
('Family Van', 85.00, 18.00, 6, 4),

('Executive Sedan', 190.00, 38.00, 4, 3),

('Newmodel Car', 100.00, 35.00, 1, 2);

SELECT\*FROM Car\_Category;

Output:



Script for Location Table:

INSERT INTO Location\_Details

(LOCATION\_ID, LOCATION\_NAME, CITY, STATE\_NAME, STREET, ZIPCODE)

VALUES

('LOC001', 'Downtown Center', 'New York', 'NY', '123 Main St', '10001'),

('LOC002', 'Uptown Plaza', 'New York', 'NY', '456 Oak Ave', '10002'),

('LOC003', 'City Square', 'Los Angeles', 'CA', '789 Pine Rd', '90001'),

('LOC004', 'Beachside Park', 'Los Angeles', 'CA', '101 Beach Dr', '90002'),

('LOC005', 'Park View', 'Chicago', 'IL', '234 Elm St', '60001'),

('LOC006', 'Greenwood Station', 'Chicago', 'IL', '567 Maple Ave', '60002'),

('LOC007', 'Mountain Ridge', 'Denver', 'CO', '890 Mountain Rd', '80201'),

('LOC008', 'City Gateway', 'Denver', 'CO', '321 Canyon St', '80202'),

('LOC009', 'Lakefront Park', 'San Francisco', 'CA', '678 Lakeshore Blvd', '94101'),

('LOC010', 'Sunset View', 'San Francisco', 'CA', '135 Sunset Rd', '94102'),

('LOC011', 'Riverbend Plaza', 'Austin', 'TX', '246 River Rd', '73301'),

('LOC012', 'Highland Square', 'Austin', 'TX', '789 Highland Blvd', '73302'),

('LOC013', 'Northern Lights', 'Seattle', 'WA', '102 Pine St', '98101'),

('LOC014', 'Southgate Mall', 'Seattle', 'WA', '654 Birch Ave', '98102'),

('LOC015', 'Ocean Breeze', 'Miami', 'FL', '321 Ocean Dr', '33101'),

('LOC016', 'Bayfront Park', 'Miami', 'FL', '654 Bay Rd', '33102'),

('LOC017', 'Sunny Hills', 'Phoenix', 'AZ', '987 Desert Ave', '85001'),

('LOC018', 'Cactus Valley', 'Phoenix', 'AZ', '654 Cactus St', '85002'),

('LOC019', 'Golden Gate', 'San Diego', 'CA', '741 Gold St', '92101'),

('LOC020', 'Pacific Horizon', 'San Diego', 'CA', '852 Pacific Blvd', '92102'),

('LOC021', 'Mountain Pass', 'Salt Lake City', 'UT', '123 Snowy Rd', '84101'),

('LOC022', 'Lakeview Heights', 'Salt Lake City', 'UT', '234 Lake St', '84102'),

('LOC023', 'Vista Ridge', 'Dallas', 'TX', '345 Hillcrest Rd', '75201'),

('LOC024', 'Parkway View', 'Dallas', 'TX', '456 Park St', '75202'),

('LOC025', 'Riverstone Plaza', 'Houston', 'TX', '567 Riverstone Dr', '77001'),

('LOC026', 'Clearwater Shores', 'Houston', 'TX', '678 Clearwater Blvd', '77002'),

('LOC027', 'Lakeside Gardens', 'Boston', 'MA', '789 Lakeside Ave', '02101'),

('LOC028', 'Beacon Hill', 'Boston', 'MA', '890 Beacon St', '02102'),

('LOC029', 'Maple Grove', 'Minneapolis', 'MN', '123 Maple Ave', '55101'),

('LOC030', 'Stonebrook Park', 'Minneapolis', 'MN', '456 Stone Rd', '55102'),

('LOC031', 'Brighton Square', 'Detroit', 'MI', '567 Brighton Blvd', '48201'),

('LOC032', 'Windy Acres', 'Detroit', 'MI', '678 Windy St', '48202'),

('LOC033', 'Crystal Bay', 'Portland', 'OR', '789 Crystal Rd', '97201'),

('LOC034', 'Silver Ridge', 'Portland', 'OR', '890 Silver St', '97202'),

('LOC035', 'Silverlake Mall', 'Las Vegas', 'NV', '345 Silverlake Ave', '89101'),

('LOC036', 'Desert Oasis', 'Las Vegas', 'NV', '456 Desert Rd', '89102'),

('LOC037', 'Westside Plaza', 'Philadelphia', 'PA', '567 West Ave', '19101'),

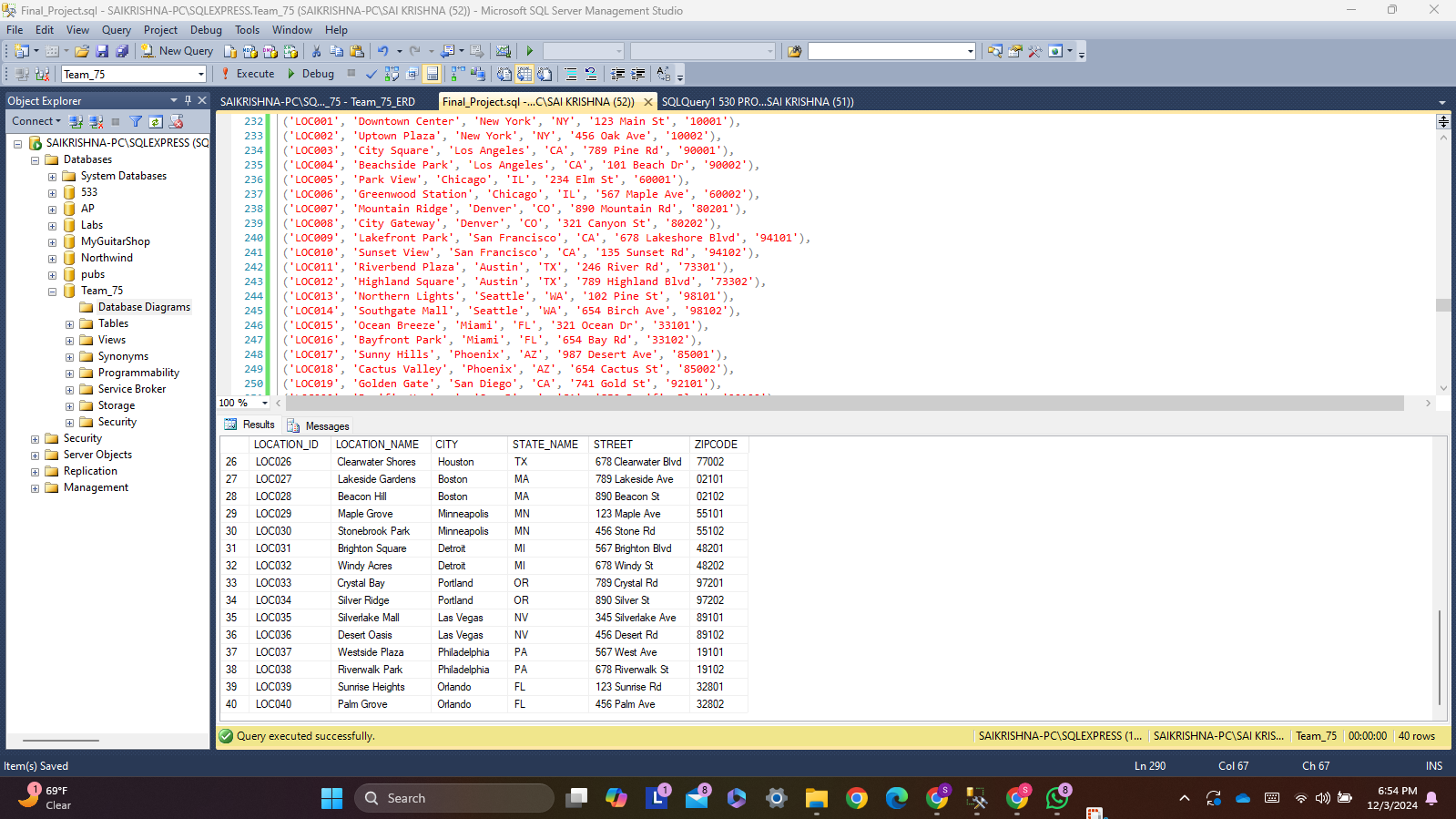
('LOC038', 'Riverwalk Park', 'Philadelphia', 'PA', '678 Riverwalk St', '19102'),

('LOC039', 'Sunrise Heights', 'Orlando', 'FL', '123 Sunrise Rd', '32801'),

('LOC040', 'Palm Grove', 'Orlando', 'FL', '456 Palm Ave', '32802');

SELECT\*FROM Location\_Details;

Output:



Script for Car Table:

INSERT INTO Car

(REGISTRATION\_NUMBER, MODEL\_NAME, MAKE, CAR\_CATEGORY\_NAME, LOC\_ID, MODEL\_YEAR, MILEAGE, AVAILABILITY\_FLAG)

VALUES

('REG001', 'Model A', 'Toyota', 'Electric', 'LOC001', 2020, 15000, 'Y'),

('REG002', 'Model B', 'Honda', 'Sedan', 'LOC002', 2019, 20000, 'Y'),

('REG003', 'Model C', 'Ford', 'SUV', 'LOC003', 2021, 12000, 'Y'),

('REG004', 'Model D', 'Chevrolet', 'Compact', 'LOC004', 2020, 18000, 'Y'),

('REG005', 'Model E', 'Nissan', 'Hybrid', 'LOC005', 2022, 8000, 'Y'),

('REG006', 'Model F', 'Ram', 'Truck', 'LOC006', 2021, 25000, 'Y'),

('REG007', 'Model G', 'Chrysler', 'Minivan', 'LOC007', 2019, 22000, 'Y'),

('REG008', 'Model H', 'Porsche', 'Sports', 'LOC008', 2020, 9000, 'Y'),

('REG009', 'Model I', 'BMW', 'Luxury', 'LOC009', 2021, 5000, 'Y'),

('REG010', 'Model J', 'Mercedes', 'Convertible', 'LOC010', 2021, 7000, 'Y'),

('REG011', 'Model K', 'Mazda', 'Coupe', 'LOC011', 2020, 15000, 'Y'),

('REG012', 'Model L', 'Jeep', 'Crossover', 'LOC012', 2022, 6000, 'Y'),

('REG013', 'Model M', 'Subaru', 'Wagon', 'LOC013', 2021, 13000, 'Y'),

('REG014', 'Model N', 'Hyundai', 'Hatchback', 'LOC014', 2020, 14000, 'Y'),

('REG015', 'Model O', 'Kia', 'Midsize', 'LOC015', 2022, 7000, 'Y'),

('REG016', 'Model P', 'Toyota', 'Fullsize', 'LOC016', 2019, 19000, 'Y'),

('REG017', 'Model Q', 'Ford', 'Roadster', 'LOC017', 2021, 8000, 'Y'),

('REG018', 'Model R', 'Audi', 'Luxury SUV', 'LOC018', 2021, 5000, 'Y'),

('REG019', 'Model S', 'Tesla', 'Executive', 'LOC019', 2022, 2000, 'Y'),

('REG020', 'Model T', 'Land Rover', 'Off-road', 'LOC020', 2020, 15000, 'Y'),

('REG021', 'Model U', 'Toyota', 'Pickup', 'LOC021', 2021, 10000, 'Y'),

('REG022', 'Model V', 'Honda', 'Minivan XL', 'LOC022', 2020, 18000, 'Y'),

('REG023', 'Model W', 'Ford', 'Economy', 'LOC023', 2019, 22000, 'Y'),

('REG024', 'Model X', 'Chevrolet', 'Cargo Van', 'LOC024', 2021, 30000, 'Y'),

('REG025', 'Model Y', 'Nissan', 'Subcompact', 'LOC025', 2022, 5000, 'Y'),

('REG026', 'Model Z', 'Volkswagen', 'MPV', 'LOC026', 2021, 12000, 'Y'),

('REG027', 'Model AA', 'Ram', 'Pickup Extended', 'LOC027', 2020, 25000, 'Y'),

('REG028', 'Model AB', 'BMW', 'Sport Coupe', 'LOC028', 2019, 22000, 'Y'),

('REG029', 'Model AC', 'Mercedes', 'Grand Tourer', 'LOC029', 2020, 7000, 'Y'),

('REG030', 'Model AD', 'BMW', 'Utility Vehicle', 'LOC030', 2021, 8000, 'Y'),

('REG031', 'Model AE', 'Ford', 'City Car', 'LOC031', 2019, 30000, 'Y'),

('REG032', 'Model AF', 'Honda', 'Compact SUV', 'LOC032', 2020, 20000, 'Y'),

('REG033', 'Model AG', 'Porsche', 'Luxury Sedan', 'LOC033', 2021, 4000, 'Y'),

('REG034', 'Model AH', 'Chevrolet', 'Four-door Sedan', 'LOC034', 2020, 15000, 'Y'),

('REG035', 'Model AI', 'Audi', 'Coupe SUV', 'LOC035', 2021, 12000, 'Y'),

('REG036', 'Model AJ', 'BMW', 'Convertible SUV', 'LOC036', 2020, 10000, 'Y'),

('REG037', 'Model AK', 'Lexus', 'Touring Sedan', 'LOC037', 2022, 5000, 'Y'),

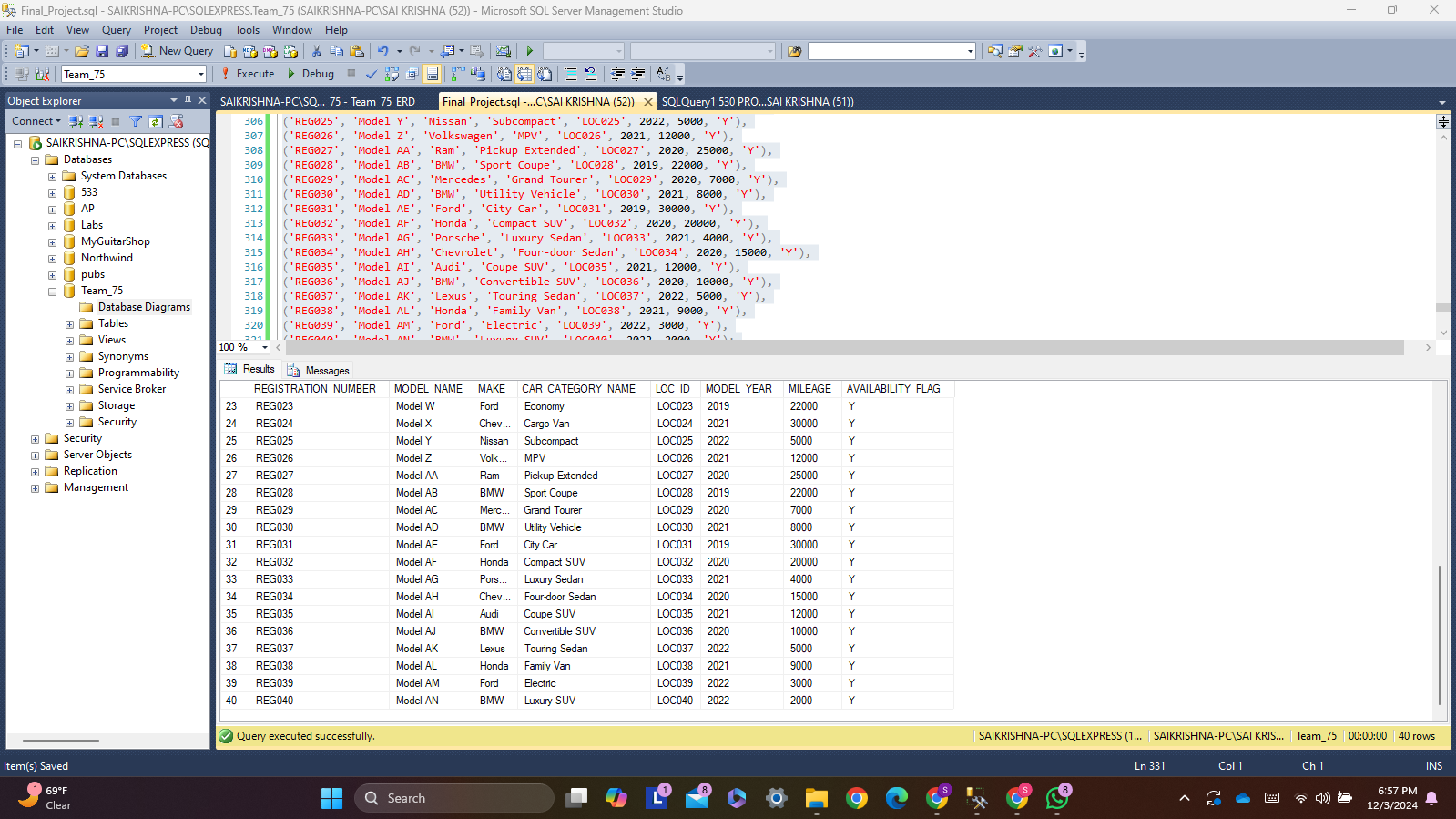
('REG038', 'Model AL', 'Honda', 'Family Van', 'LOC038', 2021, 9000, 'Y'),

('REG039', 'Model AM', 'Ford', 'Electric', 'LOC039', 2022, 3000, 'Y'),

('REG040', 'Model AN', 'BMW', 'Luxury SUV', 'LOC040', 2022, 2000, 'Y');

SELECT\*FROM Car;

Output:



Script for Discount Table:

INSERT INTO Discount\_Details

(DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE)

VALUES

('DISC01', 'New Year Discount', 10.00, '2025-01-01'),

('DISC02', 'Summer Sale', 15.00, '2025-06-30'),

('DISC03', 'Black Friday', 25.00, '2024-11-30'),

('DISC04', 'Holiday Special', 20.00, '2024-12-31'),

('DISC05', 'Spring Discount', 10.00, '2025-03-31'),

('DISC06', 'Weekend Offer', 5.00, '2024-12-07'),

('DISC07', 'VIP Member Discount', 30.00, '2025-01-01'),

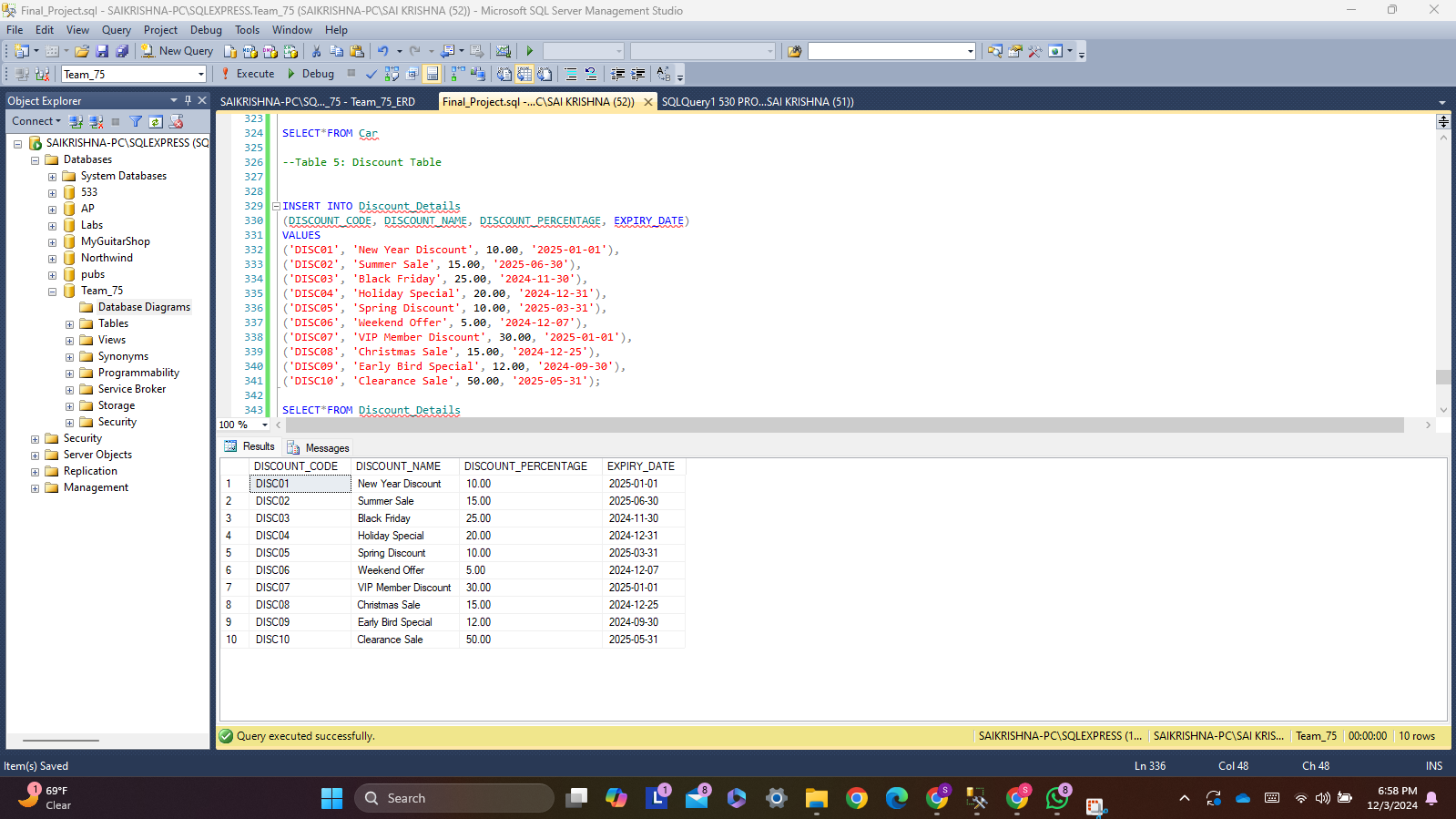
('DISC08', 'Christmas Sale', 15.00, '2024-12-25'),

('DISC09', 'Early Bird Special', 12.00, '2024-09-30'),

('DISC10', 'Clearance Sale', 50.00, '2025-05-31');

SELECT\*FROM Discount\_Details;

Output:



Script for Rental Car Insurance Table:

INSERT INTO Rental\_Car\_Insurance

(INSURANCE\_CODE, INSURANCE\_NAME, COST\_PER\_DAY, COVERAGE\_TYPE)

VALUES

('INS001', 'Basic Coverage', 15.00, 'Covers damage to the car and basic liability'),

('INS002', 'Standard Coverage', 25.00, 'Covers damage to the car, basic liability, and theft protection'),

('INS003', 'Premium Coverage', 35.00, 'Covers full damage to the car, theft protection, and roadside assistance'),

('INS004', 'Roadside Assistance', 10.00, 'Provides emergency roadside assistance like flat tire change, battery jump, etc.'),

('INS005', 'Collision Damage Waiver', 18.00, 'Waives responsibility for damage to the rental car if you are at fault in an accident'),

('INS006', 'Liability Protection', 12.00, 'Provides protection against third-party claims for bodily injury or property damage'),

('INS007', 'Personal Accident Insurance', 8.00, 'Covers medical expenses for the driver and passengers in case of an accident'),

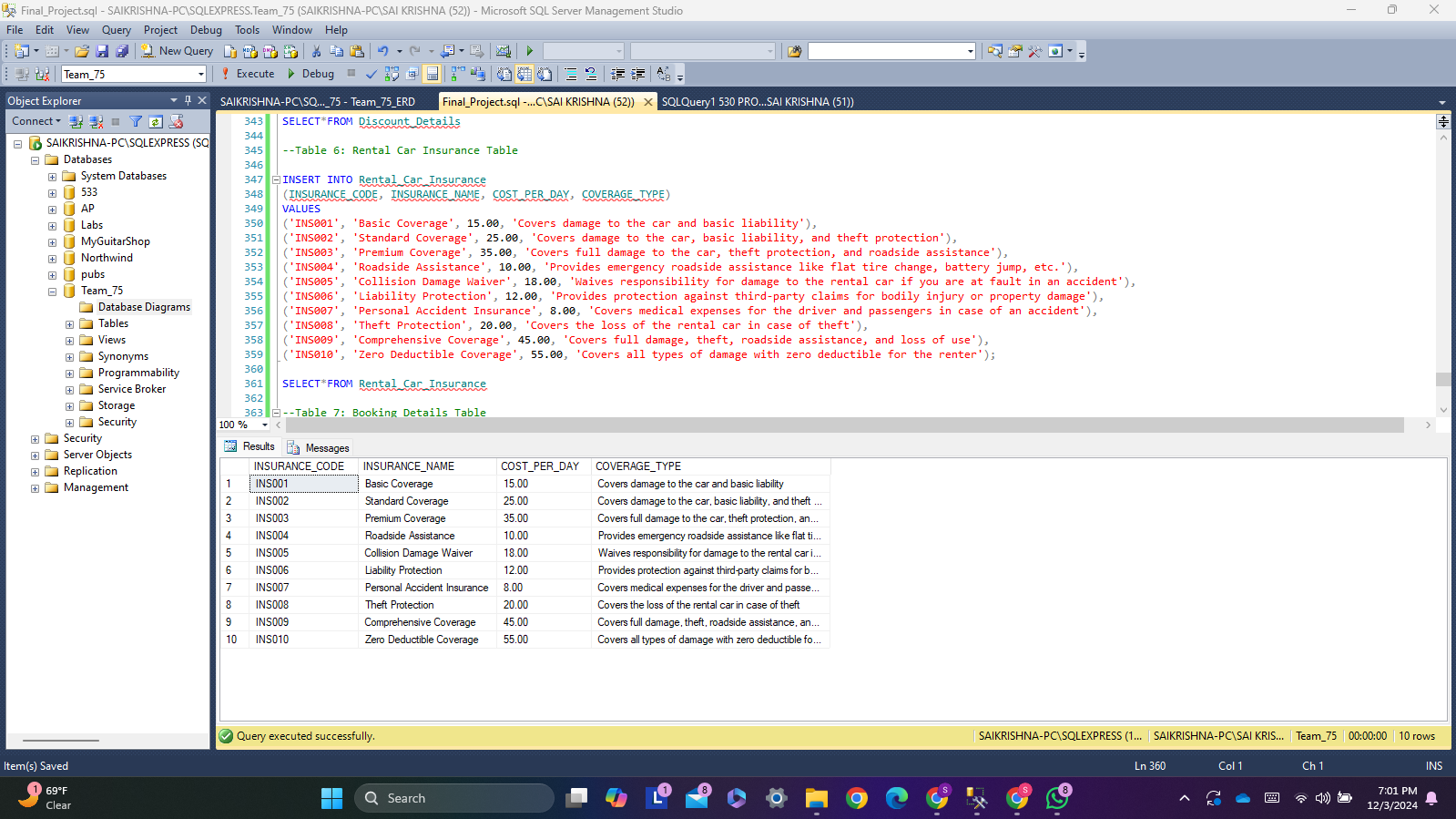
('INS008', 'Theft Protection', 20.00, 'Covers the loss of the rental car in case of theft'),

('INS009', 'Comprehensive Coverage', 45.00, 'Covers full damage, theft, roadside assistance, and loss of use'),

('INS010', 'Zero Deductible Coverage', 55.00, 'Covers all types of damage with zero deductible for the renter');

SELECT\*FROM Rental\_Car\_Insurance;

Output:



Script for Booking Details Table:

INSERT INTO Booking\_Details

(BOOKING\_ID, REG\_NUM, DL\_NUM, AMOUNT, BOOKING\_STATUS, PICKUP\_LOC, DROP\_LOC, INS\_CODE, DISCOUNT\_CODE)

VALUES

('B001', 'REG001', 'D123456789', 100.00, 'A', 'LOC001', 'LOC002', 'INS001', 'DISC01'),

('B002', 'REG002', 'S987654321', 150.50, 'A', 'LOC003', 'LOC004', 'INS002', 'DISC02'),

('B003', 'REG003', 'M123456789', 200.00, 'A', 'LOC005', 'LOC006', 'INS003', 'DISC03'),

('B004', 'REG004', 'W987654321', 175.75, 'A', 'LOC002', 'LOC003', 'INS004', 'DISC04'),

('B005', 'REG005', 'B123456789', 125.50, 'A', 'LOC006', 'LOC005', 'INS005', 'DISC05'),

('B006', 'REG006', 'J987654321', 140.00, 'A', 'LOC004', 'LOC002', 'INS006', 'DISC06'),

('B007', 'REG007', 'L123456789', 220.00, 'A', 'LOC005', 'LOC004', 'INS007', 'DISC07'),

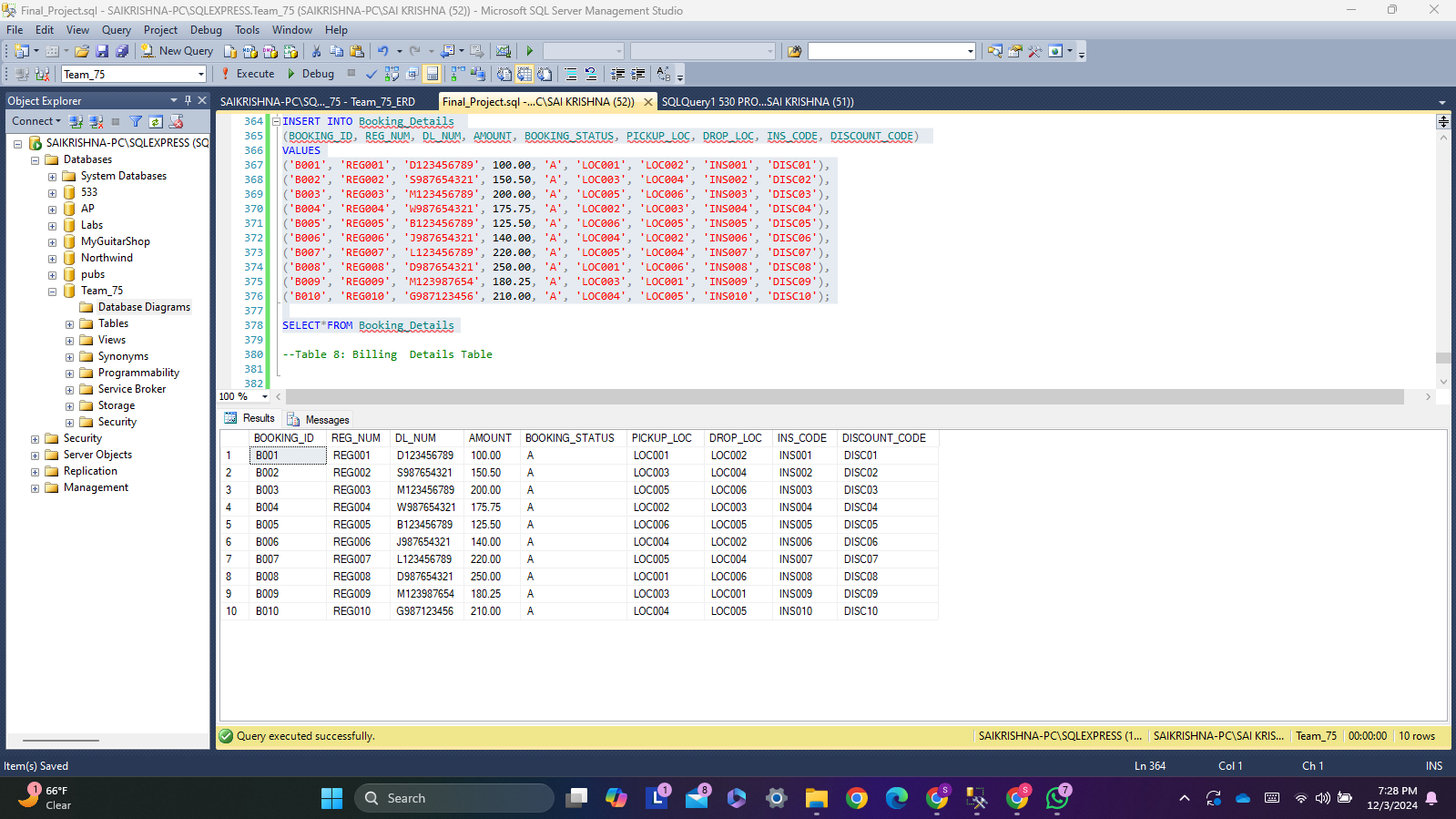
('B008', 'REG008', 'D987654321', 250.00, 'A', 'LOC001', 'LOC006', 'INS008', 'DISC08'),

('B009', 'REG009', 'M123987654', 180.25, 'A', 'LOC003', 'LOC001', 'INS009', 'DISC09'),

('B010', 'REG010', 'G987123456', 210.00, 'A', 'LOC004', 'LOC005', 'INS010', 'DISC10');

SELECT\*FROM Booking\_Details;

Output:



Script for Billing Details Table:

INSERT INTO Billing\_Details

(BILL\_ID, BILL\_DATE, BILL\_STATUS, DISCOUNT\_AMOUNT, TOTAL\_AMOUNT, TAX\_AMOUNT, BOOKING\_ID, TOTAL\_LATE\_FEE)

VALUES

('BILL01', '2024-12-01', 'P', 10.00, 100.00, 5.00, 'B001', 2.50),

('BILL02', '2024-12-02', 'P', 15.00, 150.00, 7.50, 'B002', 5.00),

('BILL03', '2024-12-03', 'C', 0.00, 200.00, 10.00, 'B003', 0.00),

('BILL04', '2024-12-04', 'P', 20.00, 175.75, 8.79, 'B004', 3.00),

('BILL05', '2024-12-05', 'C', 5.00, 125.50, 6.25, 'B005', 0.00),

('BILL06', '2024-12-06', 'P', 12.00, 140.00, 7.00, 'B006', 1.50),

('BILL07', '2024-12-07', 'P', 8.00, 220.00, 11.00, 'B007', 4.00),

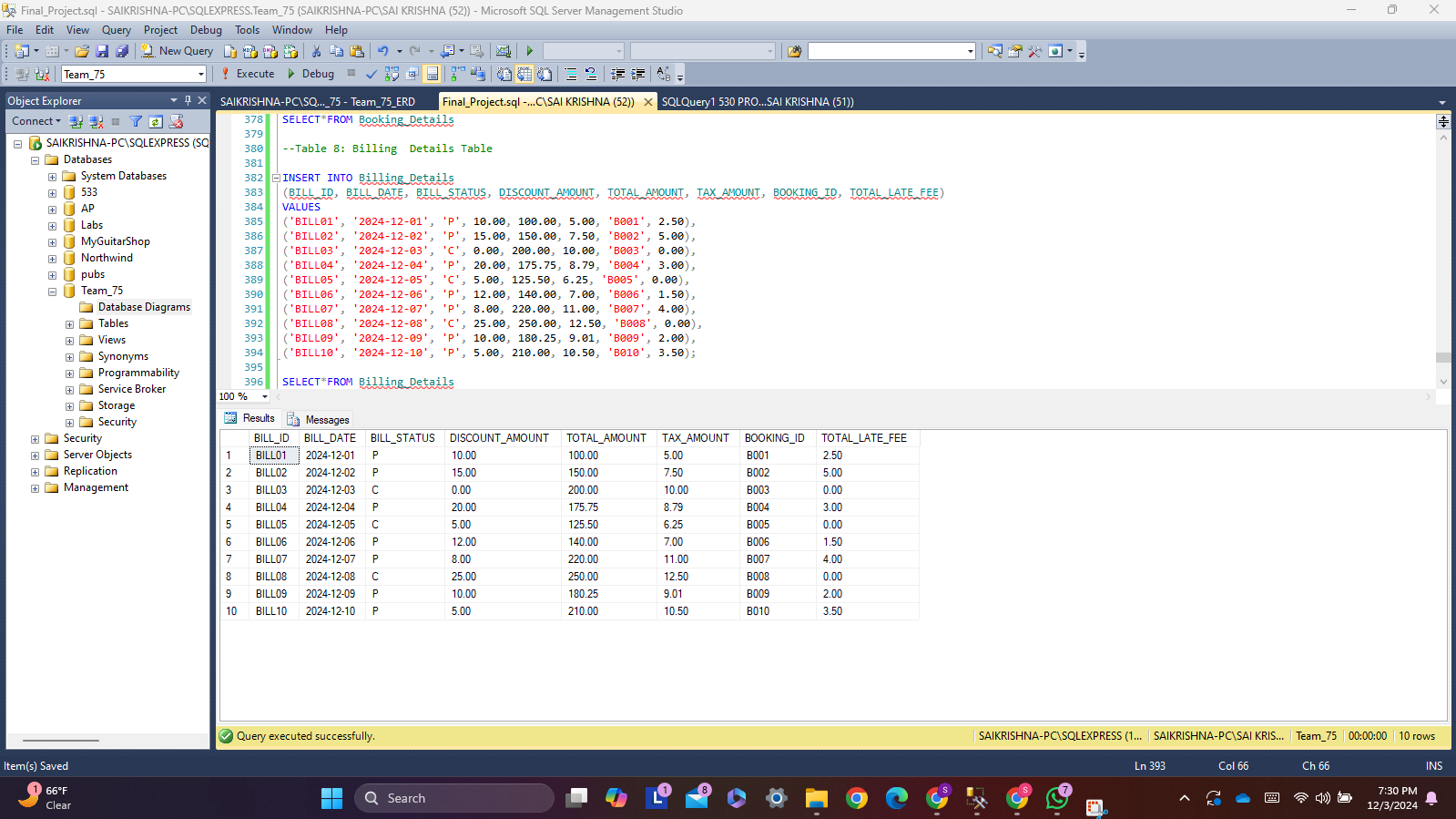
('BILL08', '2024-12-08', 'C', 25.00, 250.00, 12.50, 'B008', 0.00),

('BILL09', '2024-12-09', 'P', 10.00, 180.25, 9.01, 'B009', 2.00),

('BILL10', '2024-12-10', 'P', 5.00, 210.00, 10.50, 'B010', 3.50);

SELECT\*FROM Billing\_Details;

Output:



**Creating queries and converting them into views**

**Script for View 1:**

CREATE VIEW Customer\_Membership\_Info

AS

SELECT

CD.DL\_NUMBER AS Driver\_License\_Number,

CD.FNAME AS First\_Name,

CD.MNAME AS Middle\_Name,

CD.LNAME AS Last\_Name,

CD.PHONE\_NUMBER AS Phone\_Number,

CD.EMAIL\_ID AS Email\_Address,

CD.STREET AS Street\_Address,

CD.CITY AS City\_Name,

CD.STATE\_NAME AS State\_Name,

CD.ZIPCODE AS Zip\_Code,

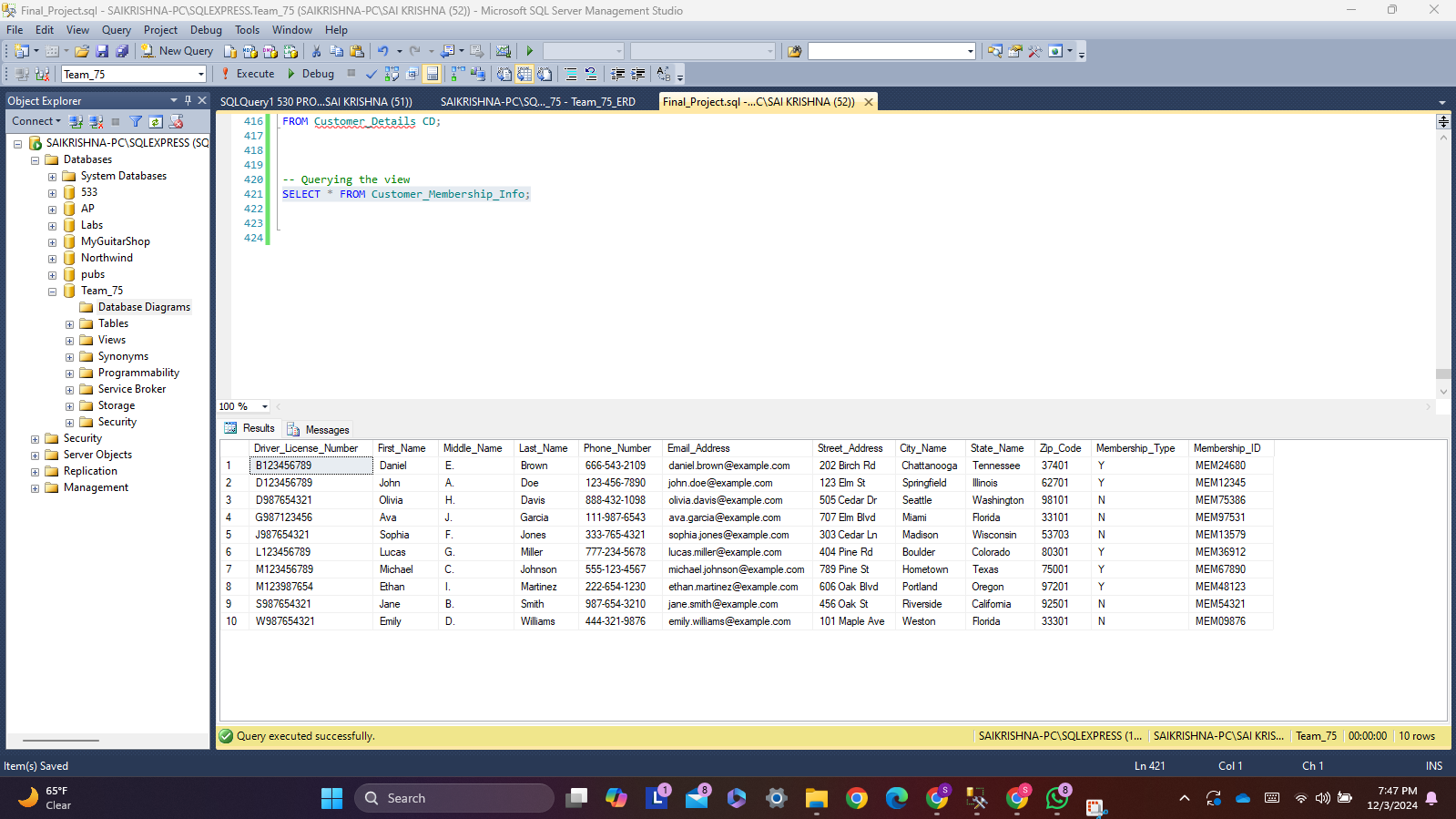
CD.MEMBERSHIP\_TYPE AS Membership\_Type,

CD.MEMBERSHIP\_ID AS Membership\_ID

FROM Customer\_Details CD;

SELECT \* FROM Customer\_Membership\_Info;

**Output:**



**Usage Explanation:** The "Customer\_Membership\_Info" view gives a complete summary of customers and their membership details. It helps the rental car agency manage customers better by showing important data like contact details, membership type, and ID. This makes customer service easier, improves marketing, and saves time by avoiding the need to check multiple tables.

**Script for View 2:**

CREATE VIEW Available\_Vehicles AS

SELECT

Vehicle.REGISTRATION\_NUMBER AS Vehicle\_Registration\_ID,

Vehicle.MODEL\_NAME AS Vehicle\_Model,

Vehicle.MAKE AS Vehicle\_Manufacturer,

VehicleCategory.CATEGORY\_NAME AS Category\_Type,

VehicleCategory.NO\_OF\_LUGGAGE AS Luggage\_Capacity,

VehicleCategory.NO\_OF\_PERSON AS Passenger\_Capacity,

VehicleCategory.COST\_PER\_DAY AS Daily\_Rental\_Cost,

VehicleCategory.LATE\_FEE\_PER\_HOUR AS Hourly\_Late\_Fee,

Vehicle.AVAILABILITY\_FLAG AS Availability\_Status

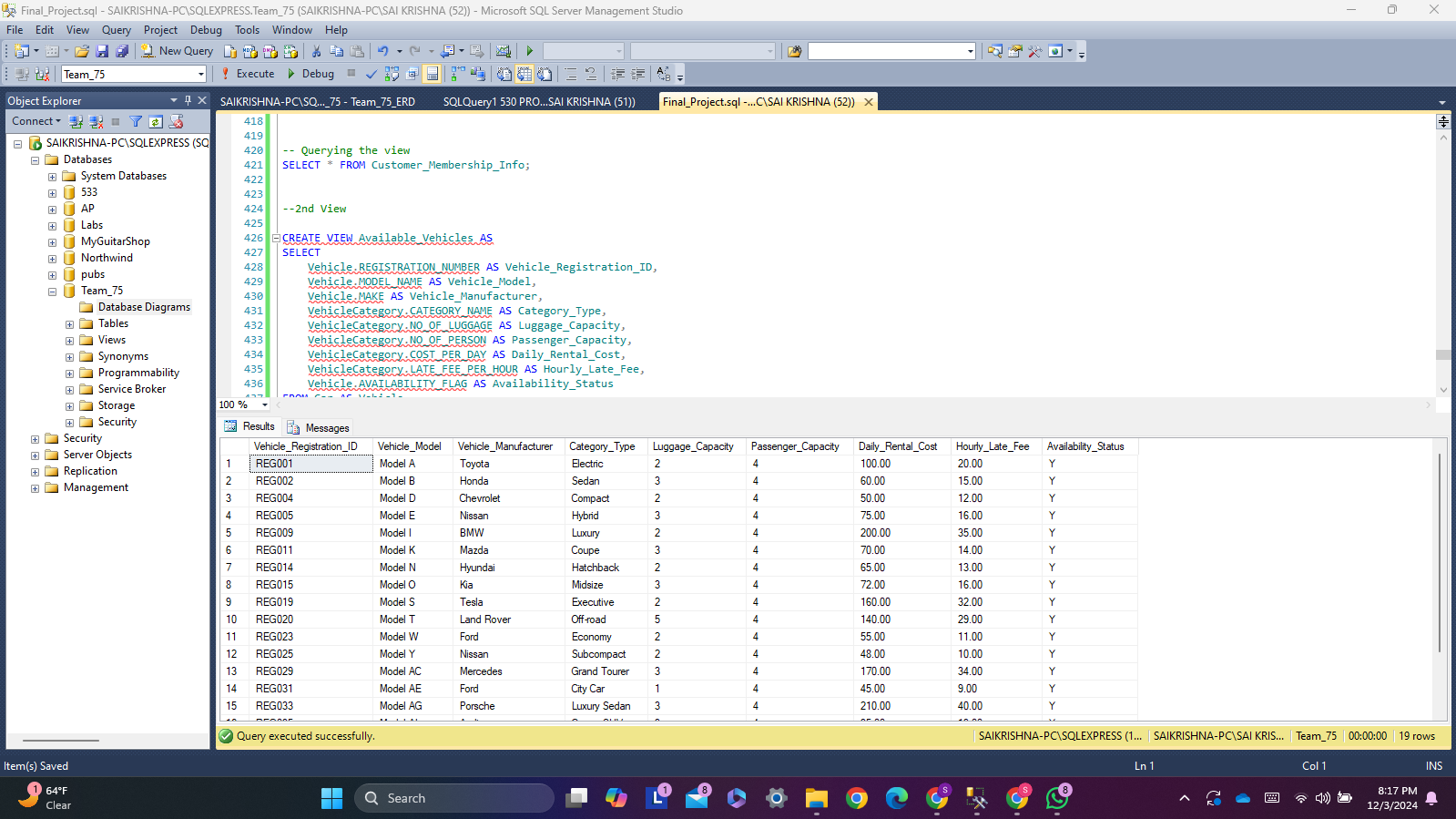
FROM Car AS Vehicle

JOIN Car\_Category AS VehicleCategory ON Vehicle.CAR\_CATEGORY\_NAME = VehicleCategory.CATEGORY\_NAME

WHERE VehicleCategory.NO\_OF\_PERSON = 4;

SELECT \* FROM Available\_Vehicles

**Output:**



**Usage Explanation:** The "Available\_Cars" view is useful for managing cars. It shows all available cars with details like model, make, luggage space, and daily rental cost. This helps the agency’s team to quickly find cars for customers. It also keeps the fleet balanced, ensuring the right cars are ready for use.

**Script for View 3:**

CREATE VIEW Active\_Bookings AS

SELECT

SR.BOOKING\_ID,

SR.AMOUNT,

SR.BOOKING\_STATUS,

SK.BILL\_ID,

SK.BILL\_DATE,

SK.BILL\_STATUS,

SK.DISCOUNT\_AMOUNT,

SK.TOTAL\_AMOUNT,

SK.TAX\_AMOUNT,

SK.TOTAL\_LATE\_FEE

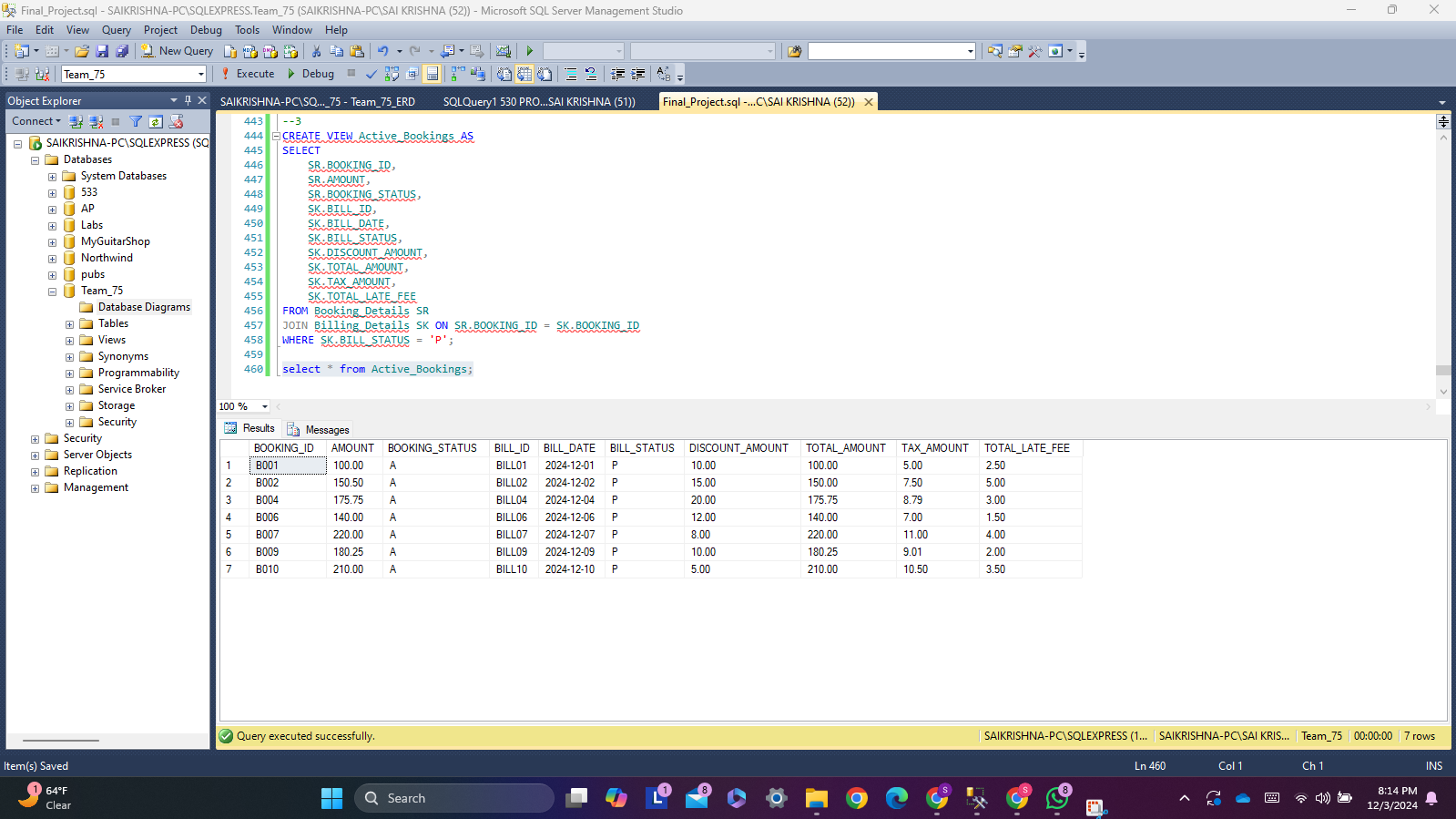
FROM Booking\_Details SR

JOIN Billing\_Details SK ON SR.BOOKING\_ID = SK.BOOKING\_ID

WHERE SK.BILL\_STATUS = 'P';

select \* from Active\_Bookings;

**Output:**



**Usage Explanation:** The "Active\_Bookings" view shows important booking and billing details together. It connects booking info with billing records, making it easy to see ongoing bookings and their payment status. This helps the accounts team manage payments and keep records accurate. Customer service can also use it to answer billing or booking questions quickly, improving customer trust.

**Creating Audit Table:**

**Script for Creating Audit Table:**

CREATE TABLE Discount\_Details\_Audit (

AUDIT\_ID INT IDENTITY(1,1) PRIMARY KEY,

ACTION\_TYPE VARCHAR(15) NOT NULL,

ACTION\_DATETIME DATETIME DEFAULT GETDATE(),

DISCOUNT\_CODE CHAR(6) NOT NULL,

DISCOUNT\_NAME VARCHAR(50) NOT NULL,

DISCOUNT\_PERCENTAGE DECIMAL(5, 2) NOT NULL,

EXPIRY\_DATE DATE NOT NULL

);

**Script for Insert Trigger:**

CREATE TRIGGER trg\_InsertDiscount

ON Discount\_Details

AFTER INSERT

AS

BEGIN

INSERT INTO Discount\_Details\_Audit (ACTION\_TYPE, DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE)

SELECT 'INSERT', DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE

FROM INSERTED;

END;

**Script for Update Trigger:**

CREATE TRIGGER trg\_UpdateDiscount

ON Discount\_Details

AFTER UPDATE

AS

BEGIN

INSERT INTO Discount\_Details\_Audit (ACTION\_TYPE, DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE)

SELECT 'UPDATE', DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE

FROM INSERTED;

END;

**Script for delete Trigger:**

CREATE TRIGGER trg\_DeleteDiscount

ON Discount\_Details

AFTER DELETE

AS

BEGIN

INSERT INTO Discount\_Details\_Audit (ACTION\_TYPE, DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE)

SELECT 'DELETE', DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE

FROM DELETED;

END;

**Testing the Audit Results:**

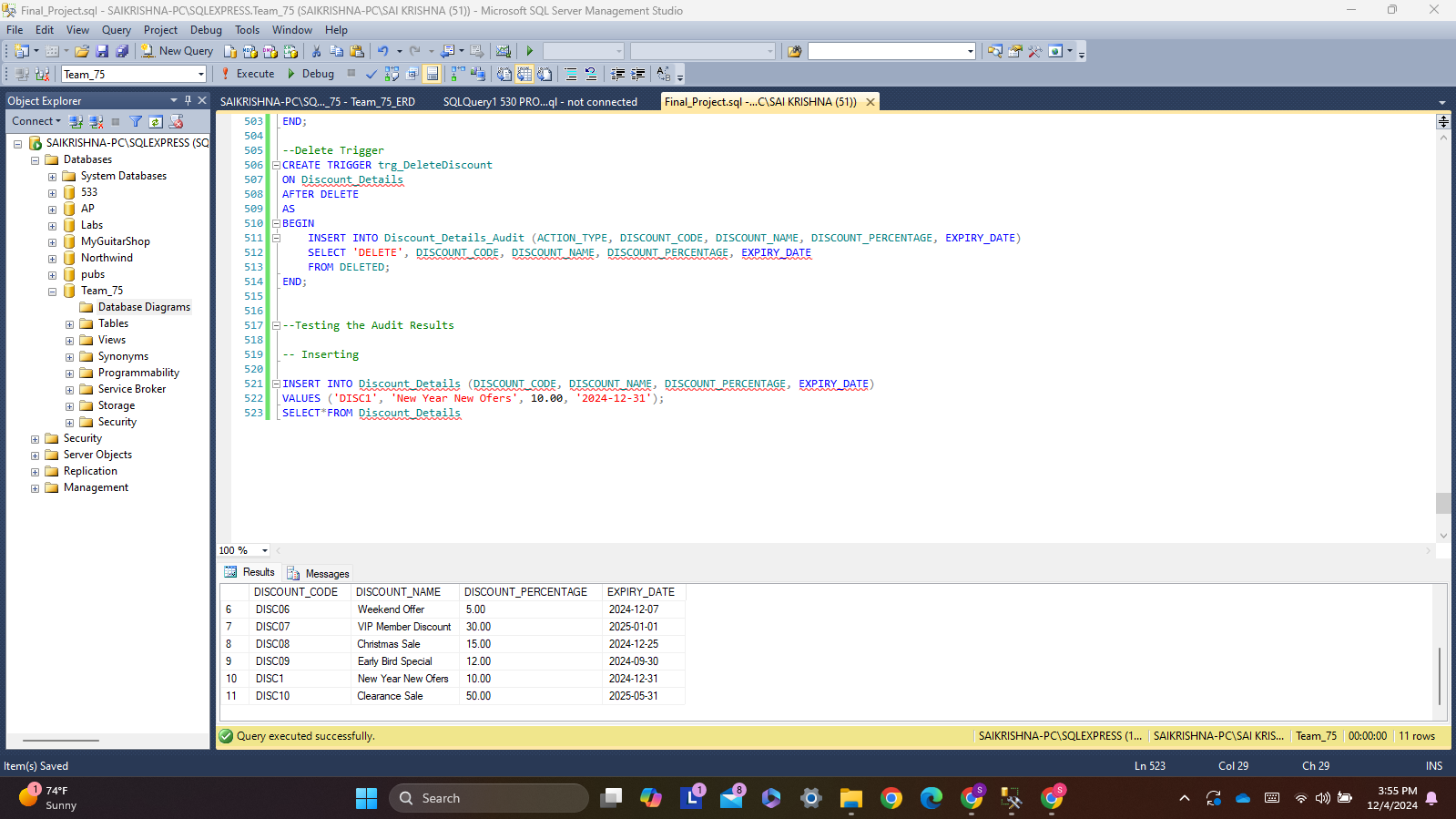
**Script for Insert:**

INSERT INTO Discount\_Details (DISCOUNT\_CODE, DISCOUNT\_NAME, DISCOUNT\_PERCENTAGE, EXPIRY\_DATE)

VALUES ('DISC1', 'New Year New Ofers', 10.00, '2024-12-31');

SELECT\*FROM Discount\_Details

**Output:**



**Script for Update:**

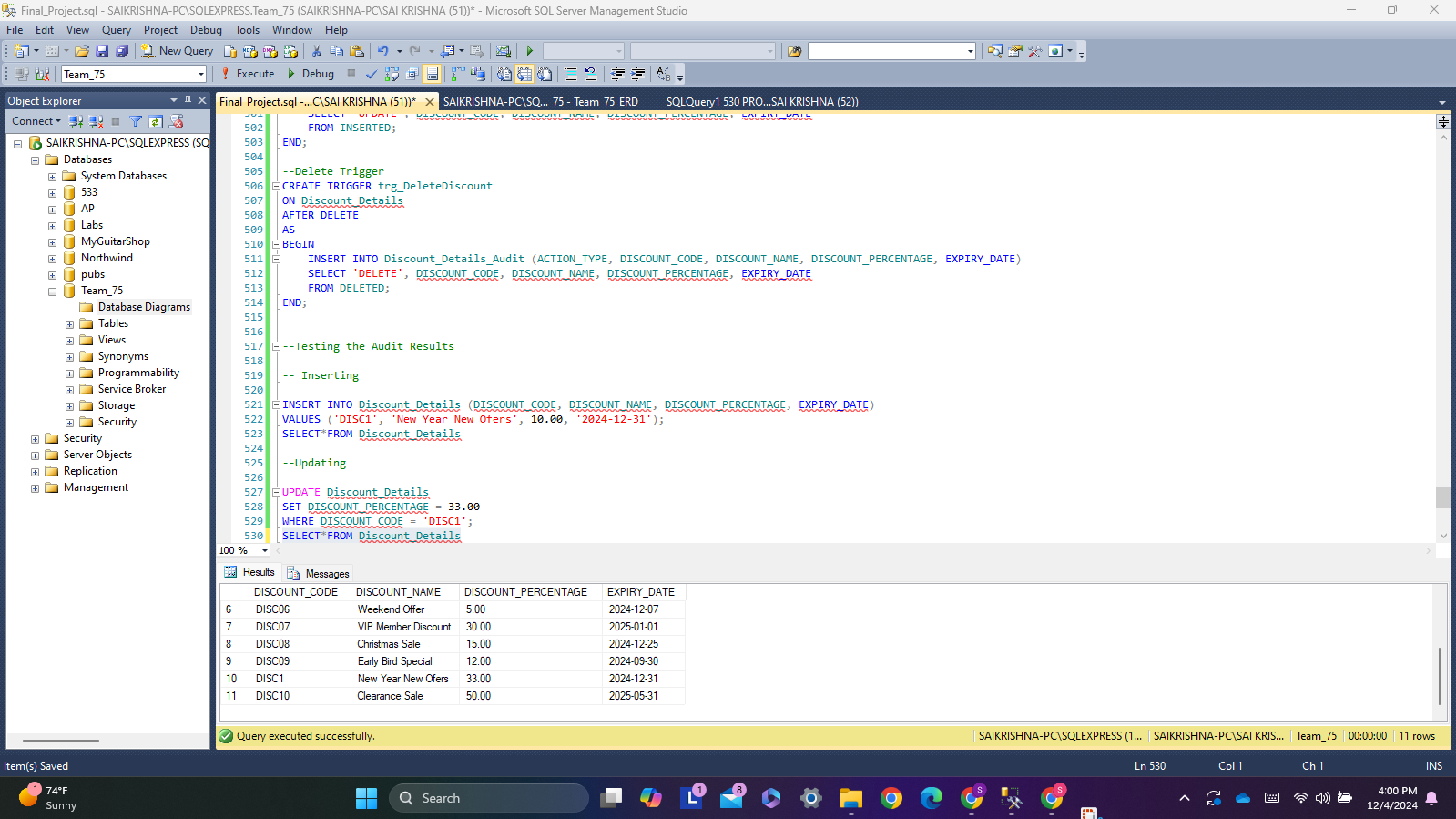
UPDATE Discount\_Details

SET DISCOUNT\_PERCENTAGE = 33.00

WHERE DISCOUNT\_CODE = 'DISC1';

SELECT\*FROM Discount\_Details

**Output:**



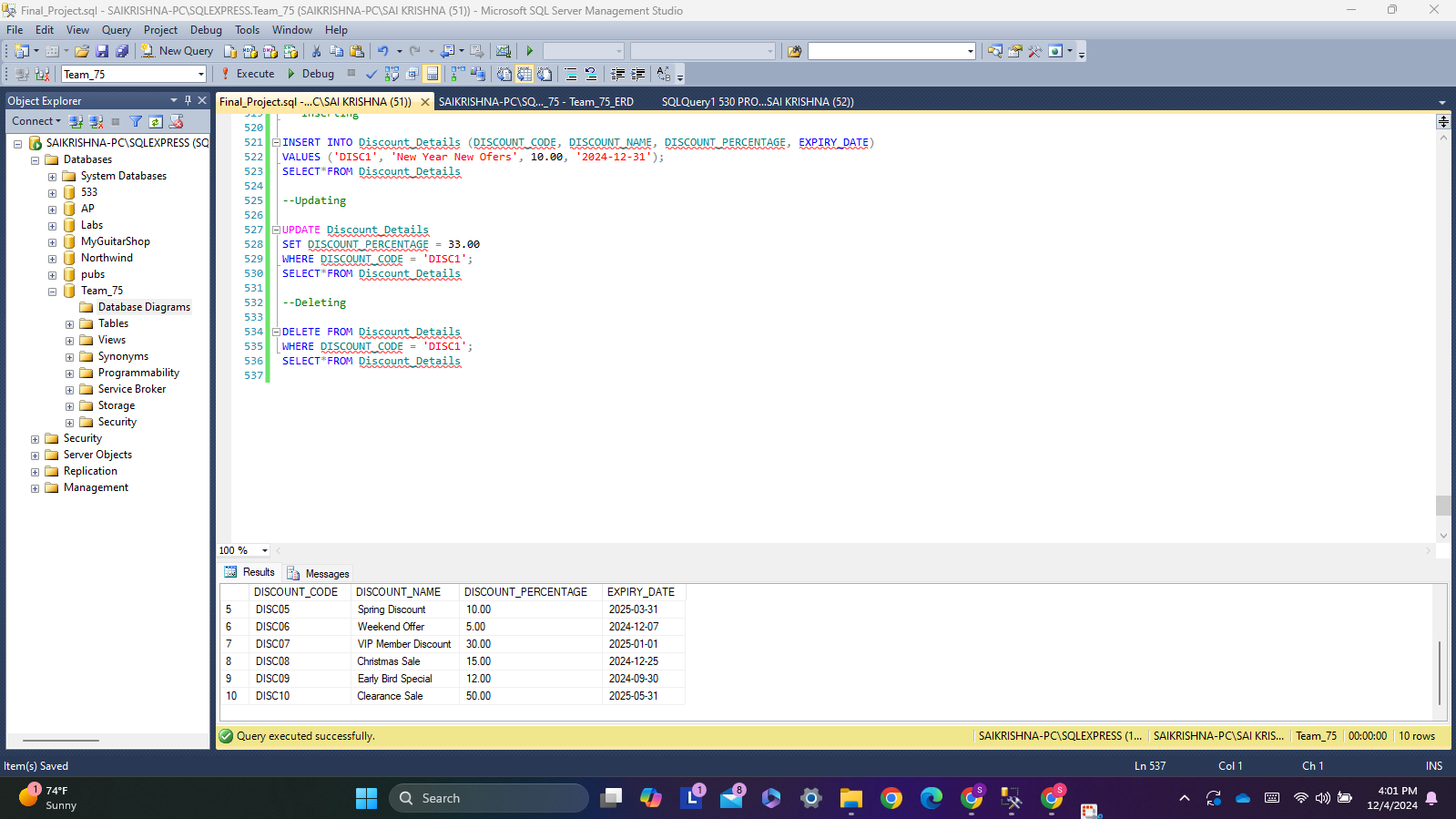
**Script for Delete:**

DELETE FROM Discount\_Details

WHERE DISCOUNT\_CODE = 'DISC1';

SELECT\*FROM Discount\_Details

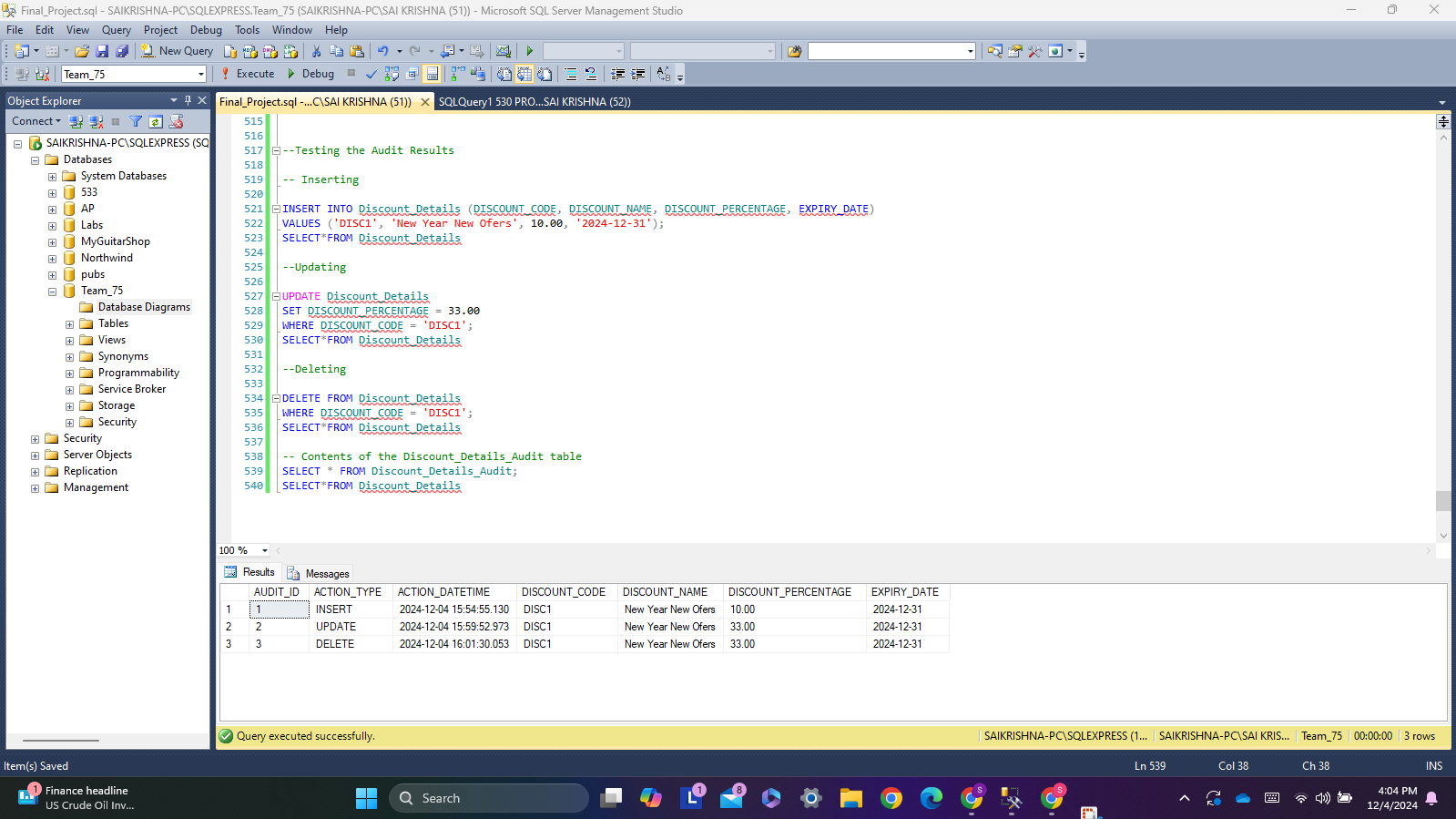
**Output:**



**Script To show Discount\_Details\_Audit table:**

SELECT \* FROM Discount\_Details\_Audit;

**Output:**



**Summary and Usage:**

The Discount\_Details\_Audit table keeps track of changes made to the original Discount\_Details table, such as when data is inserted, updated, or deleted. This can be done using triggers in SQL Server. This setup makes sure that every change in the Discount\_Details table is recorded in the Discount\_Details\_Audit table, creating a complete history of all actions.

**Stored Procedure and User Defined Function**

**Script for Stored Procedure:**

CREATE PROCEDURE GetAvailableCars

@CategoryName VARCHAR(30),

@LocationID CHAR(6)

AS

BEGIN

SELECT

REGISTRATION\_NUMBER,

MODEL\_NAME,

MAKE,

CAR\_CATEGORY\_NAME,

LOC\_ID,

MODEL\_YEAR,

MILEAGE

FROM

Car

WHERE

CAR\_CATEGORY\_NAME = @CategoryName

AND LOC\_ID = @LocationID

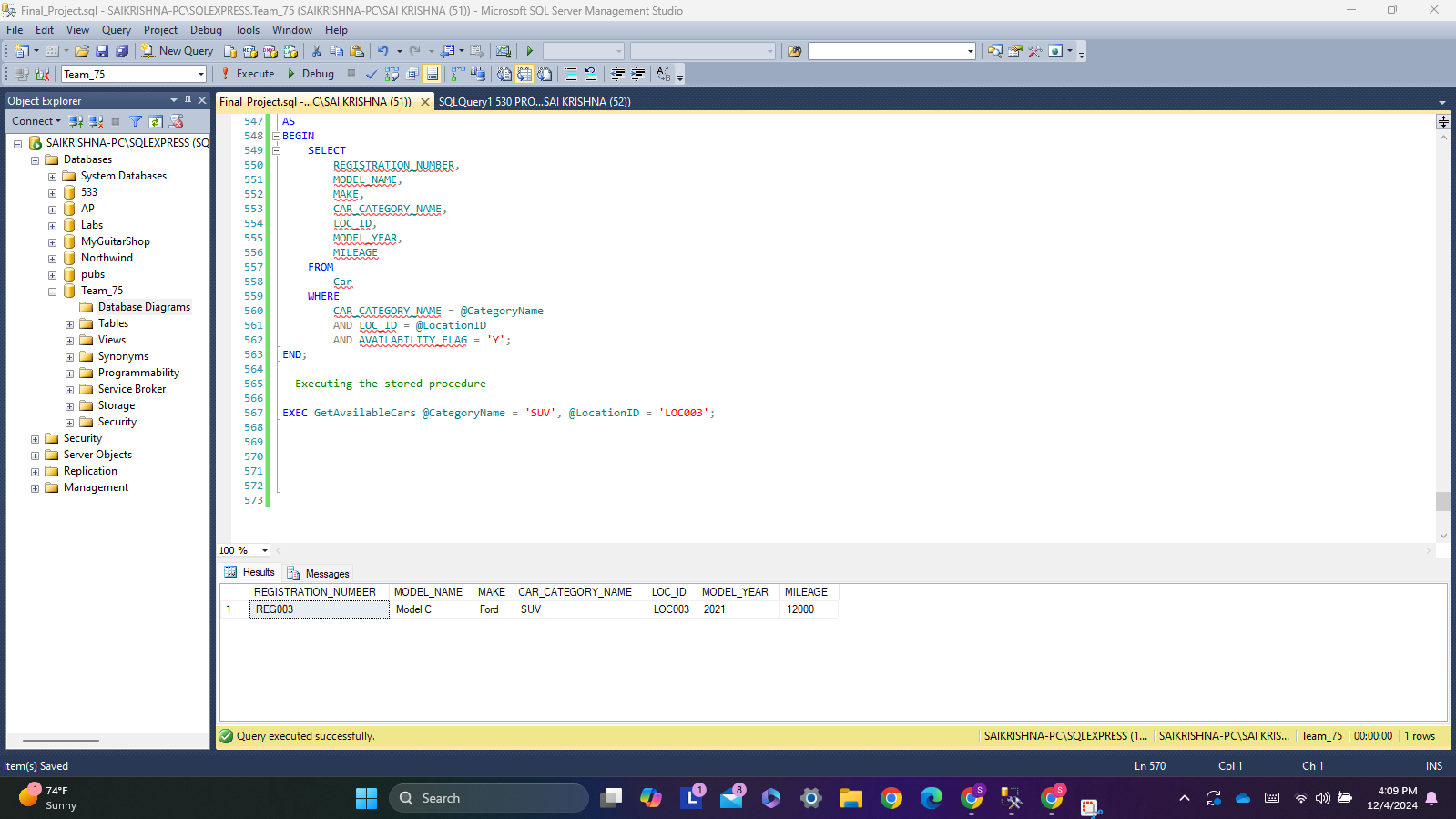
AND AVAILABILITY\_FLAG = 'Y';

END;

**Execution:**

EXEC GetAvailableCars @CategoryName = 'SUV', @LocationID = 'LOC003';

**Output:**



**Dropping:**

DROP PROCEDURE GetAvailableCars;

The location and a car category are used by this stored function to retrieve available cars.

**Script for User Defined Function:**

CREATE FUNCTION CalculateRentalCost

(

@CostPerDay DECIMAL(7, 2),

@LateFeePerHour DECIMAL(6, 2),

@RentalDays INT,

@LateHours INT

)

RETURNS DECIMAL(10, 2)

AS

BEGIN

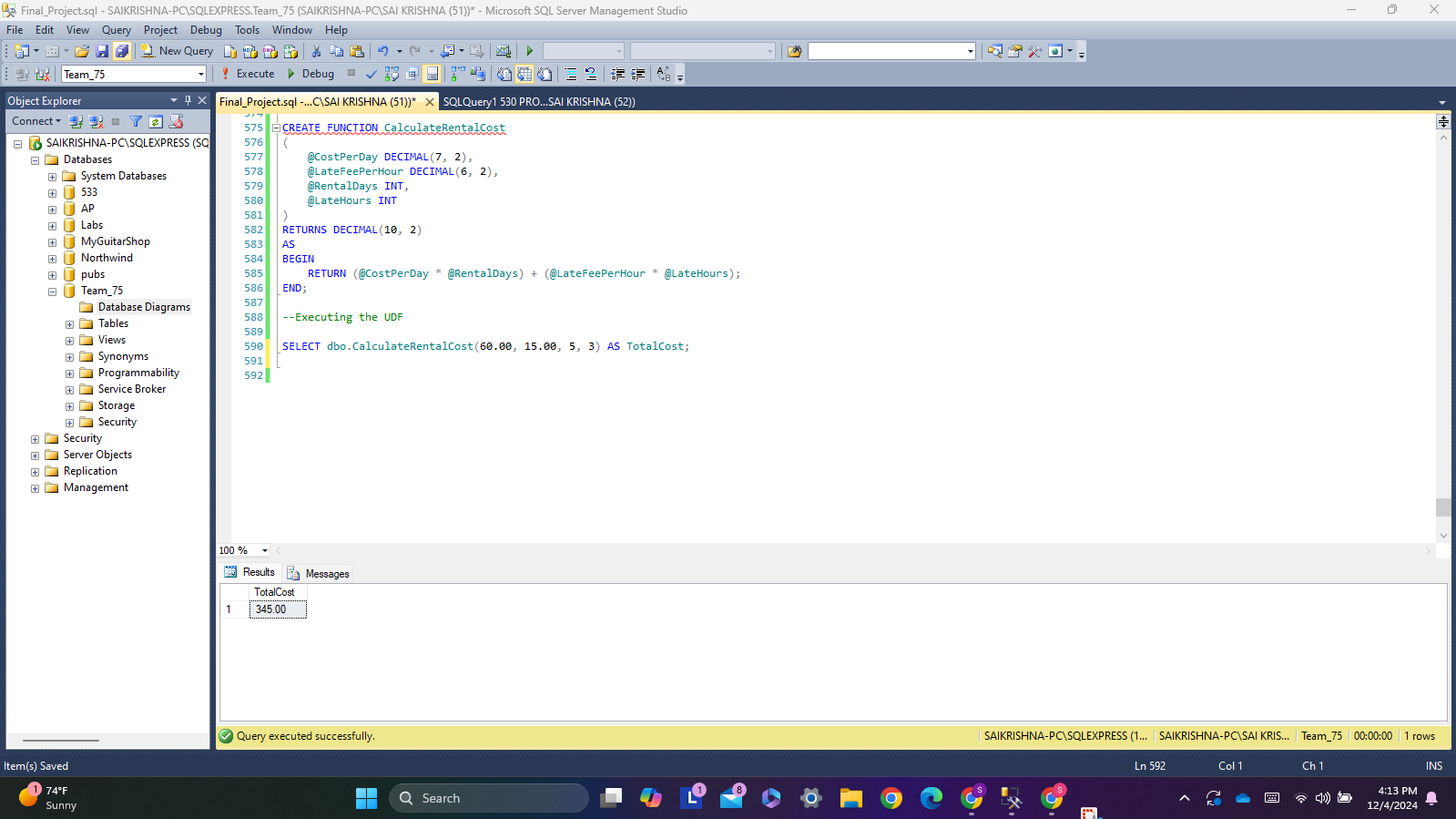
RETURN (@CostPerDay \* @RentalDays) + (@LateFeePerHour \* @LateHours);

END;

**Execution:**

SELECT dbo.CalculateRentalCost(60.00, 15.00, 5, 3) AS TotalCost;

**Output:**



**Dropping:**

DROP FUNCTION CalculateRentalCost;

Based on the amount of rental days and late hours, this function determines the entire cost of a car rental, including late fees.

**Cursor**

**Script For Cursor Creation:**

-- Cursor

CREATE PROCEDURE MarkUnavailableCars

AS

BEGIN

-- Declare the cursor

DECLARE CarCursor CURSOR FOR

SELECT REGISTRATION\_NUMBER, MILEAGE

FROM Car

WHERE MILEAGE > 100;

-- Declare variables to hold values from the cursor

DECLARE @RegistrationNumber CHAR(10);

DECLARE @Mileage INT;

-- Open the cursor

OPEN CarCursor;

-- Fetch the first row

FETCH NEXT FROM CarCursor INTO @RegistrationNumber, @Mileage;

-- Loop through all rows

WHILE @@FETCH\_STATUS = 0

BEGIN

-- Print a message about the car

PRINT 'Car with Registration Number ' + @RegistrationNumber +

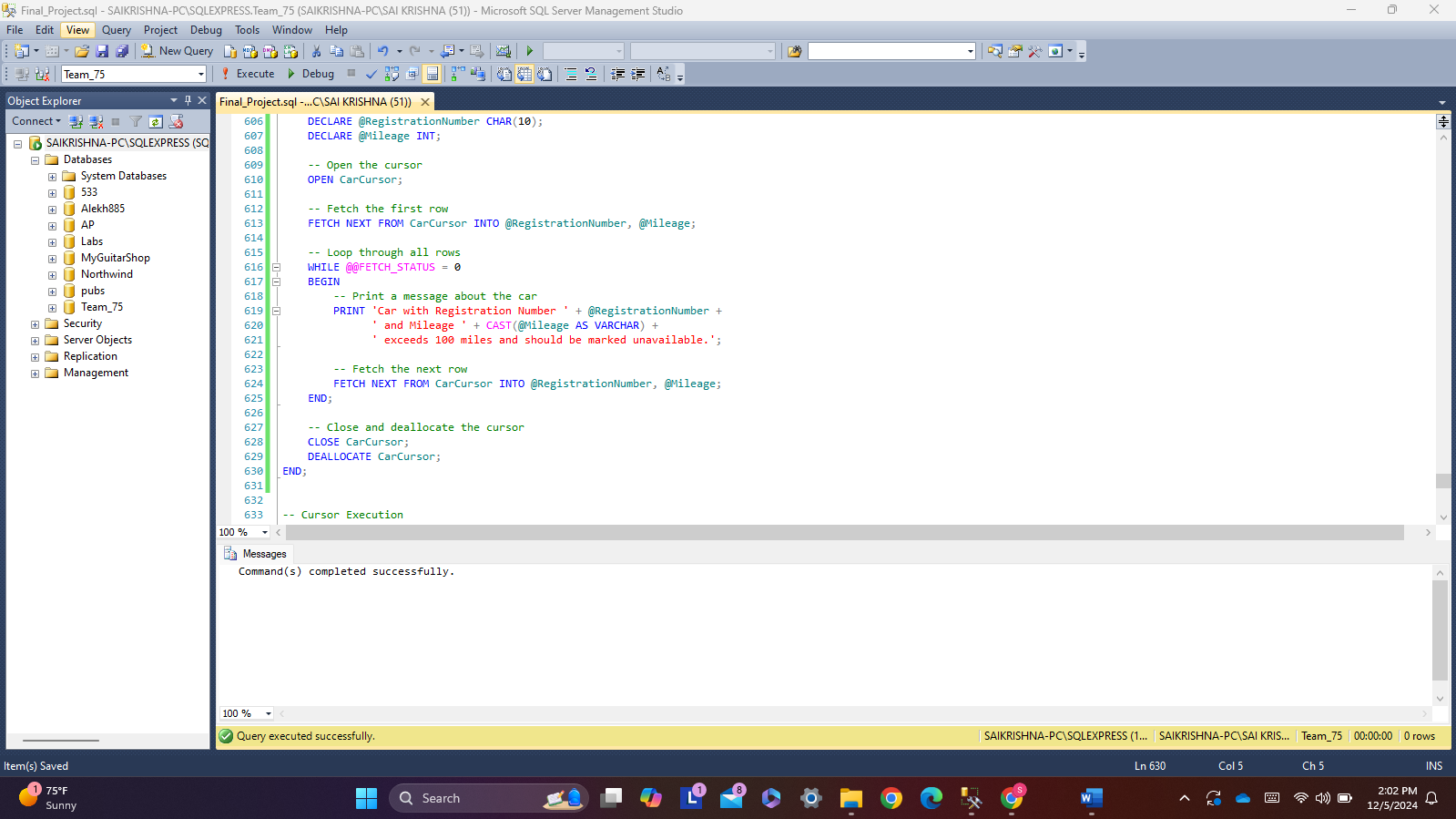
' and Mileage ' + CAST(@Mileage AS VARCHAR) +

' exceeds 100 miles and should be marked unavailable.';

-- Fetch the next row

FETCH NEXT FROM CarCursor INTO @RegistrationNumber, @Mileage;

END;

**Output: **

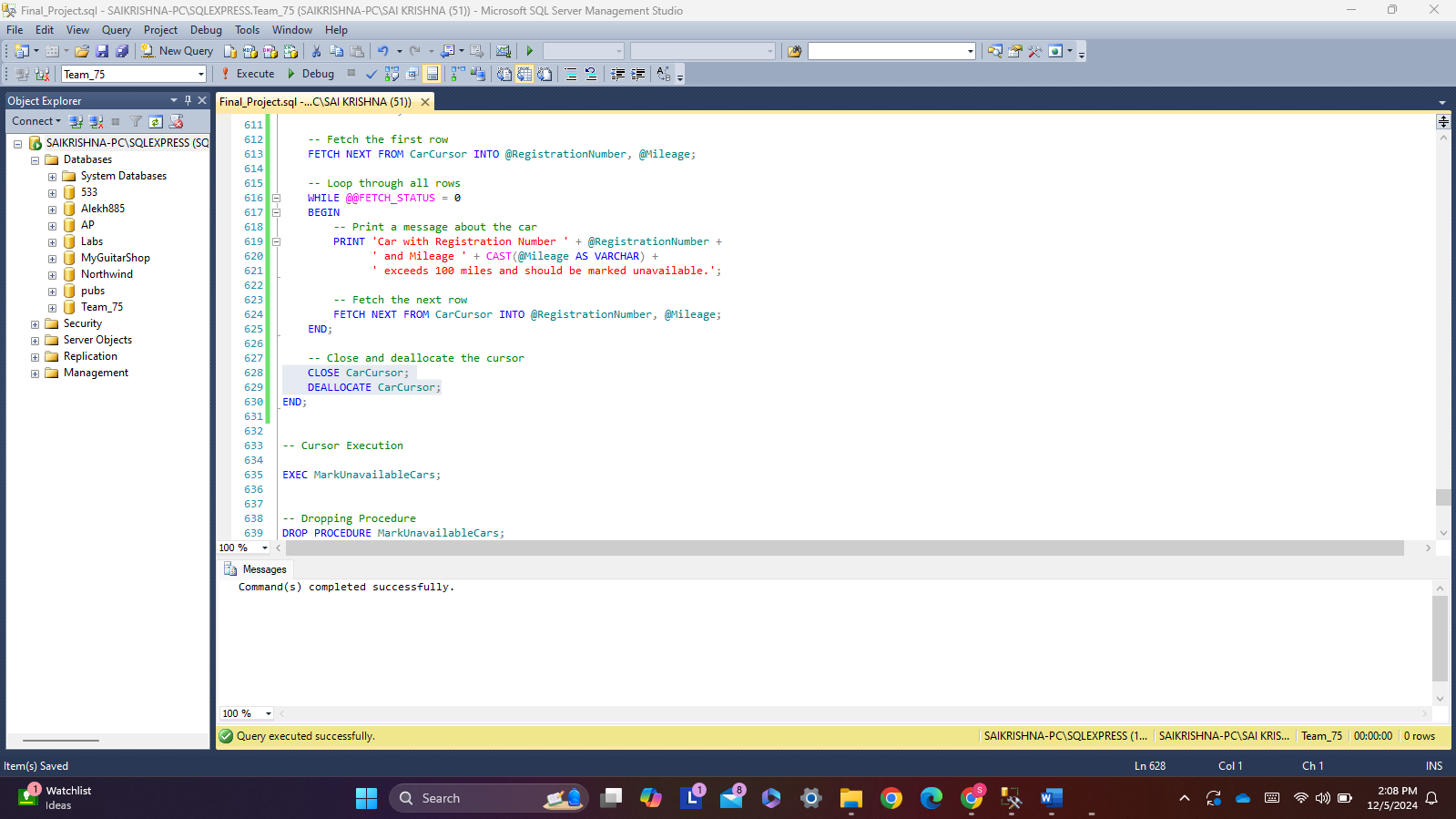
**Script for Closing and Deallocating Cursor:**

-- Close and deallocate the cursor

CLOSE CarCursor;

DEALLOCATE CarCursor;

**Output:**

****

**Script for Cursor Execution:**

-- Cursor Execution

EXEC MarkUnavailableCars;

**Output:**

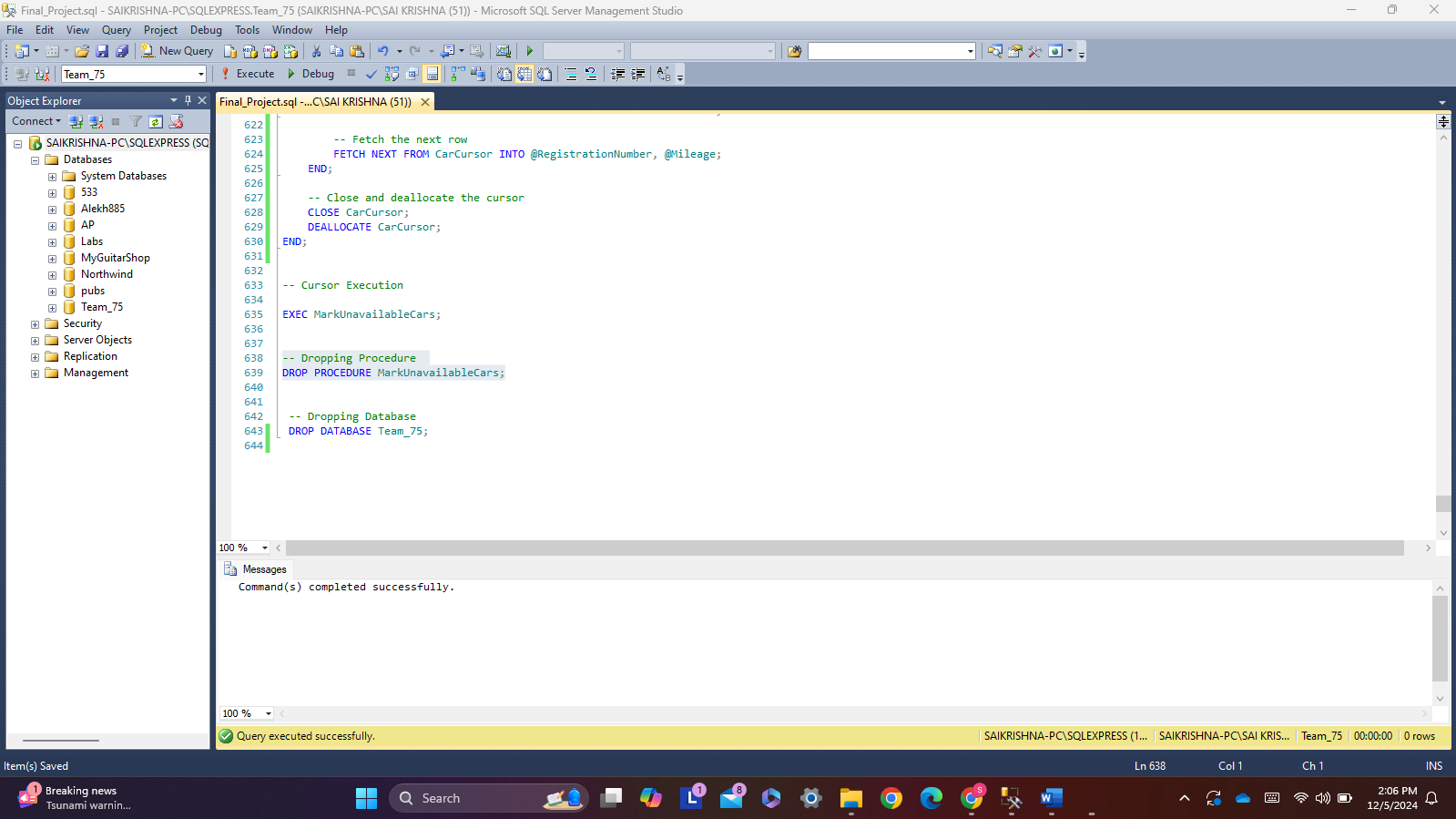
****

**Dropping Procedure:**

-- Dropping Procedure

DROP PROCEDURE MarkUnavailableCars;

**Output:**

****

In this example, a list of cars is iterated through, and if their mileage surpasses a predetermined level, they are marked as unavailable.

**Explanation of Cursor Usage:**

**Declaration of Cursor:**

A SELECT query that gets vehicles with more mileage than the threshold (100000 in this case) is used to declare the cursor.

**Cursor Functions:**

OPEN CarCursor: To begin reading rows, open the cursor.

FETCH NEXT FROM CarCursor INTO: This function assigns the following row to the variables after retrieving it from the result set.

The loop continues until every row has been handled. WHILE @@FETCH\_STATUS = 0.

When the cursor is not anymore required, resources are freed by CLOSE and DEALLOCATE CarCursor.

**Operation Update:**

The UPDATE statement indicates that the automobile is unavailable for every row that is fetched.