November 8, 2022

-- TABLES

DROP TABLE IF EXISTS Bank;

DROP TABLE IF EXISTS Loans;

DROP TABLE IF EXISTS Accounts;

DROP TABLE IF EXISTS Customers;

DROP TABLE IF EXISTS Cards;

DROP TABLE IF EXISTS Services;

DROP TABLE IF EXISTS Transactions;

CREATE TABLE Bank (

id INT IDENTITY(1,1) PRIMARY KEY,

bank\_name VARCHAR(255),

address VARCHAR(255),

);

CREATE TABLE Loans (

id INT IDENTITY(1,1) PRIMARY KEY,

amount DECIMAL(8,2),

bank\_id INT FOREIGN KEY (bank\_id) REFERENCES Bank (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

);

CREATE TABLE Accounts (

id INT IDENTITY(1,1) PRIMARY KEY,

acc\_no VARCHAR(255),

acc\_type VARCHAR(255),

balance SMALLMONEY,

bank\_id INT FOREIGN KEY (bank\_id) REFERENCES Bank (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

);

CREATE TABLE Customers (

id INT IDENTITY(1,1) PRIMARY KEY,

first\_name VARCHAR(255),

last\_name VARCHAR(255),

phone\_no VARCHAR(255),

email\_addr VARCHAR(255),

address VARCHAR(255),

acc\_id INT FOREIGN KEY (acc\_id) REFERENCES Accounts (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

loan\_id INT FOREIGN KEY (loan\_id) REFERENCES Loans (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

join\_dt DATE

);

CREATE TABLE Cards (

id INT IDENTITY(1,1) PRIMARY KEY,

card\_type VARCHAR(255) DEFAULT 'DEBIT',

acc\_id INT FOREIGN KEY (acc\_id) REFERENCES Accounts (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

);

CREATE TABLE Services (

id INT IDENTITY(1,1) PRIMARY KEY,

serv\_type VARCHAR(255),

status VARCHAR(255),

);

CREATE TABLE Transactions (

id INT IDENTITY(1,1) PRIMARY KEY,

tran\_dt DATE,

tran\_amt SMALLMONEY,

cust\_id INT FOREIGN KEY (cust\_id) REFERENCES Customers (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

acc\_id INT FOREIGN KEY (acc\_id) REFERENCES Accounts (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION,

loan\_id INT FOREIGN KEY (loan\_id) REFERENCES Loans (id)

ON DELETE NO ACTION

ON UPDATE NO ACTION

);

-- INSERT INTO TABLES

INSERT INTO

Bank (bank\_name, address)

VALUES

('NCB', 'Kingston, Jamaica');

INSERT INTO

Loans ( amount, bank\_id)

VALUES

(12000, 1),

(1000, 1);

INSERT INTO

Accounts (acc\_no, acc\_type, balance, bank\_id)

VALUES

('acc-001', 'SAVINGS', 1000, 1),

('acc-002', 'SAVINGS', 300, 1),

('acc-003', 'CURRENT', 1000, 1);

INSERT INTO

Customers (first\_name, last\_name, phone\_no, email\_addr, address, acc\_id, loan\_id, join\_dt)

VALUES

('ezra', 'muir', '(876) 356-7600', 'ezra@gmail.com', 'abc avenue', 1, 1, '2022-01-28'),

('trizzel', 'white', '(876) 000-7600', 'name@email.com', 'xyz harbour', 2, 2, '2009-03-20'),

('jada', 'kingdom', '(876) 344-7000', 'jada@gmail.com', 'abc avenue', 3, 2, '2010-05-24');

INSERT INTO

Cards (card\_type, acc\_id)

VALUES

('DEBIT', 3),

('DEBIT', 1),

('DEBIT', 2);

INSERT INTO

Services (serv\_type, status)

VALUES

('account', 'ACTIVE'),

('loan', 'CLOSED'),

('card', 'INACTIVE');

INSERT INTO

Transactions (tran\_dt, tran\_amt, cust\_id, acc\_id, loan\_id)

VALUES

('2021-12-31', 100001, 1, 1, NULL),

('2022-02-23', 6000, 1, 1, NULL),

('2015-06-19', 700, 1, 1, NULL),

('2022-08-31', 90, 2, 2, 2),

('2020-10-30', 300, 3, 3, 2),

('2015-06-25', 200, 3, 3, 2);

-- QUERIES -- --------------------------------

SELECT \* FROM Bank;

SELECT \* FROM Loans;

SELECT \* FROM Accounts;

SELECT \* FROM Customers;

SELECT \* FROM Cards;

SELECT \* FROM Services;

SELECT \* FROM Transactions;

-- Question 5 - Generate a SQL query for eligible customers for the LOAN

DECLARE @month\_gap DATE = DATEADD(month, -24, GETDATE())

SELECT

\*

FROM

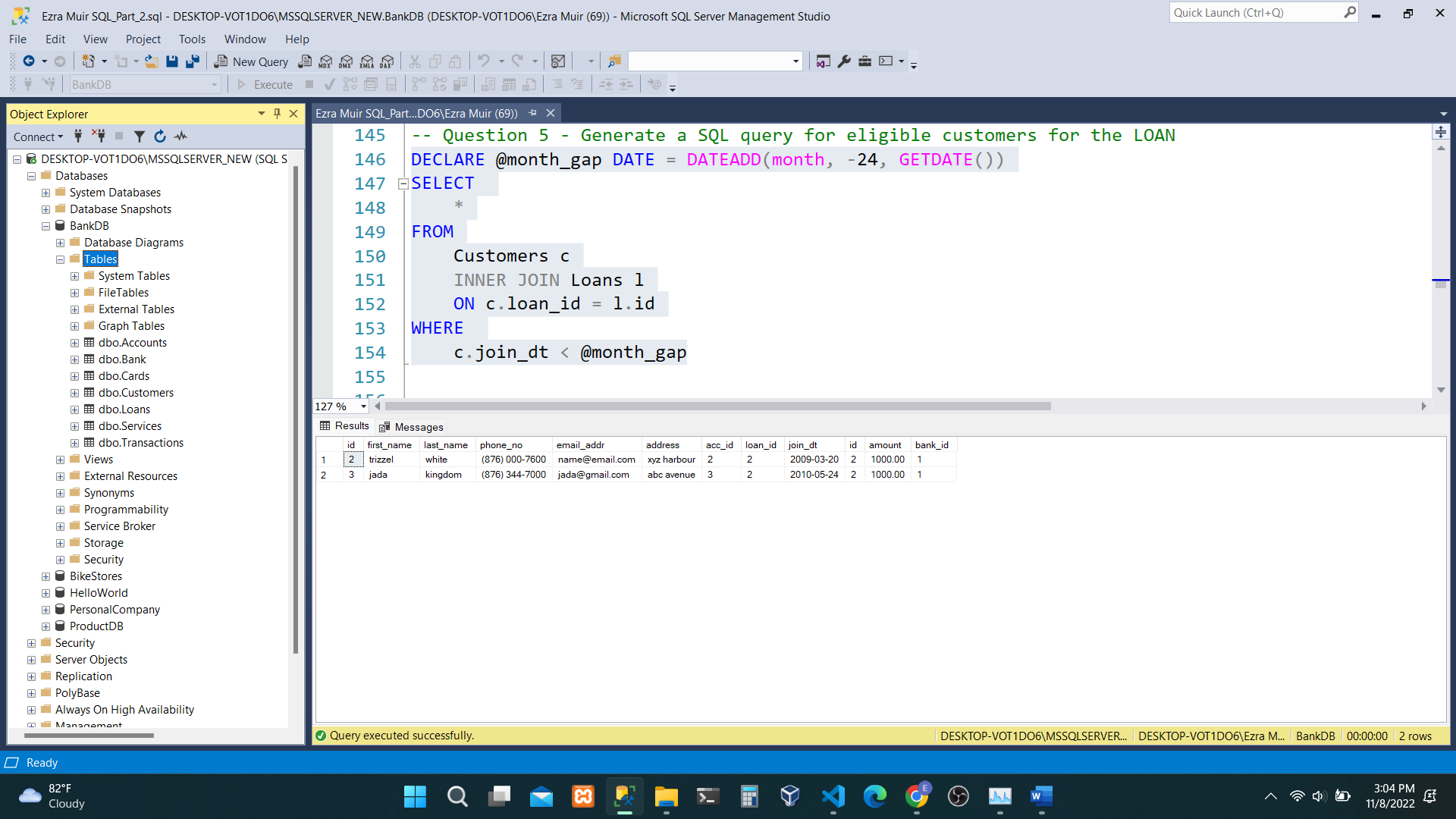
Customers c

INNER JOIN Loans l

ON c.loan\_id = l.id

WHERE

c.join\_dt < @month\_gap



-- Question 6 - Generate a SQL query for eligible customers for the Credit card service

DECLARE @year\_gap DATE = DATEADD(MONTH, -12, GETDATE())

DECLARE @current\_dt DATE = CAST(GETDATE() AS DATE)

DECLARE @card\_grant INT = 100000

SELECT

\*

FROM

Customers c

INNER JOIN Cards cs

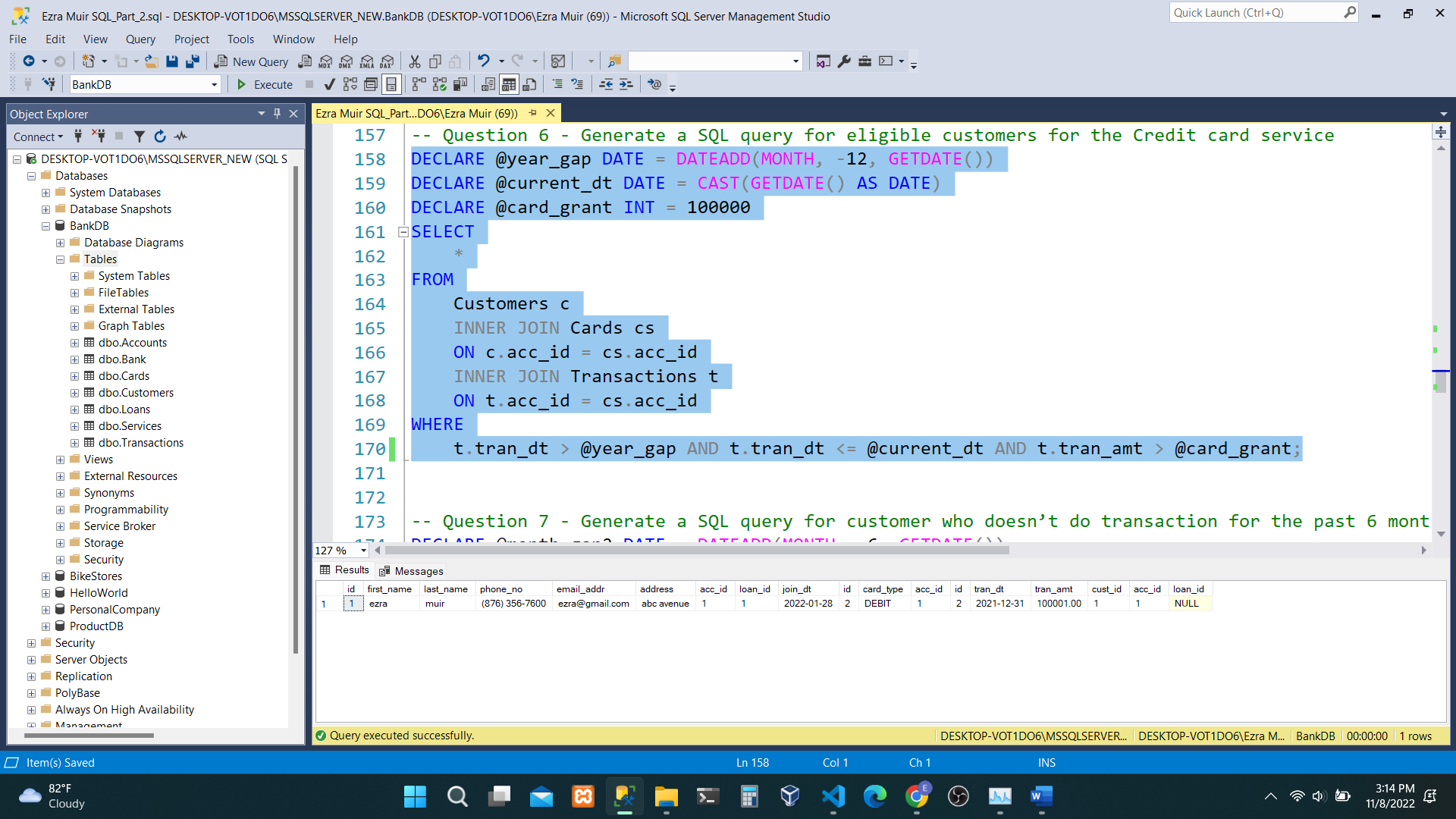
ON c.acc\_id = cs.acc\_id

INNER JOIN Transactions t

ON t.acc\_id = cs.acc\_id

WHERE

t.tran\_dt > @year\_gap AND t.tran\_dt <= @current\_dt AND t.tran\_amt > @card\_grant;



-- Question 7 - Generate a SQL query for customer who doesn’t do transaction for the past 6 months.

DECLARE @month\_gap2 DATE = DATEADD(MONTH, -6, GETDATE())

SELECT

c.id,

t.cust\_id 'Customer ID',

c.loan\_id,

c.first\_name,

c.last\_name,

c.join\_dt,

t.id,

t.tran\_dt

FROM

Customers c

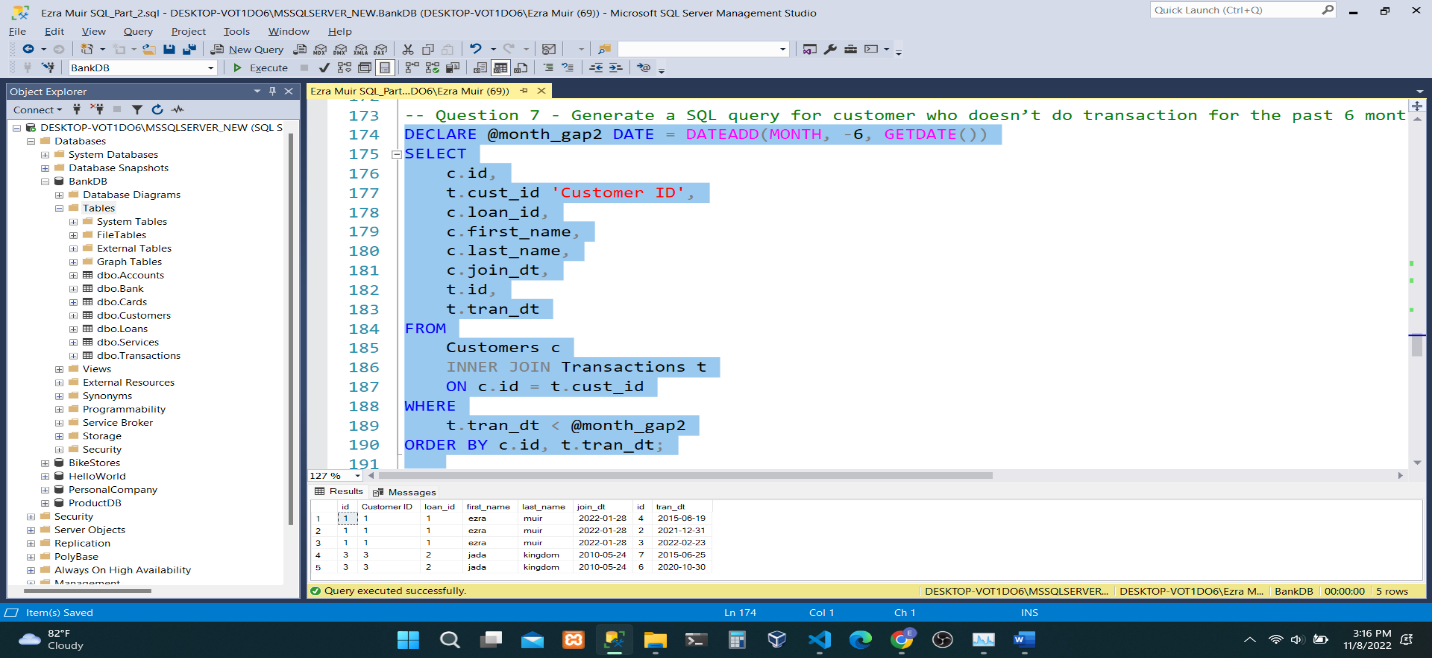
INNER JOIN Transactions t

ON c.id = t.cust\_id

WHERE

t.tran\_dt < @month\_gap2

ORDER BY c.id, t.tran\_dt;



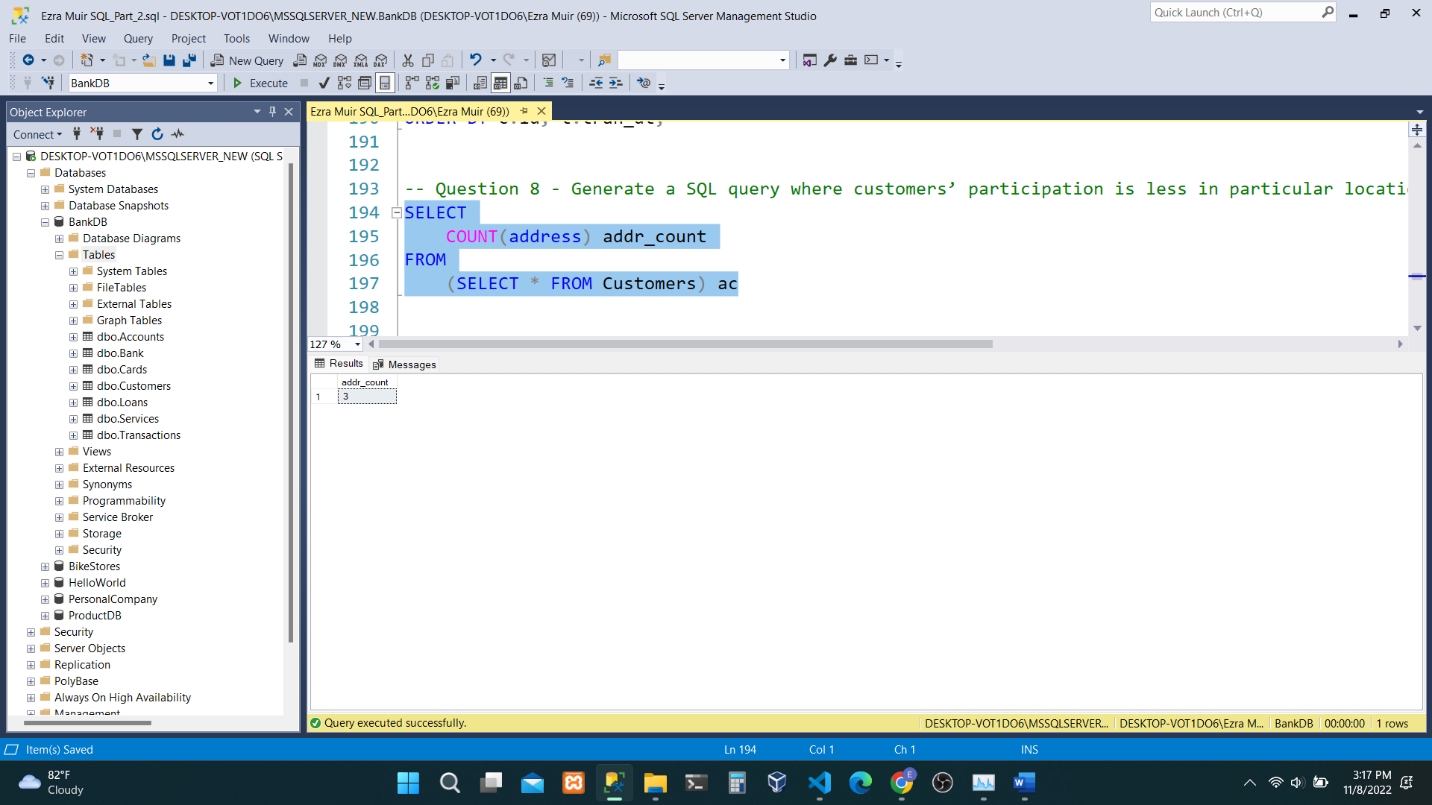
-- Question 8 - Generate a SQL query where customers’ participation is less in particular location.

SELECT

COUNT(address) addr\_count

FROM

(SELECT \* FROM Customers) ac



-- Question 9 - Generate a SQL query where customer and loan and outstanding amount.

SELECT

c.first\_name + ' ' + c.last\_name AS 'Customer Name',

t.tran\_dt,

'Outstanding Balance' = l.amount - sum(t.tran\_amt)

OVER (

PARTITION BY a.acc\_no

ORDER BY a.acc\_no rows

BETWEEN UNBOUNDED PRECEDING AND CURRENT row

)

FROM

Transactions t

INNER JOIN Accounts a

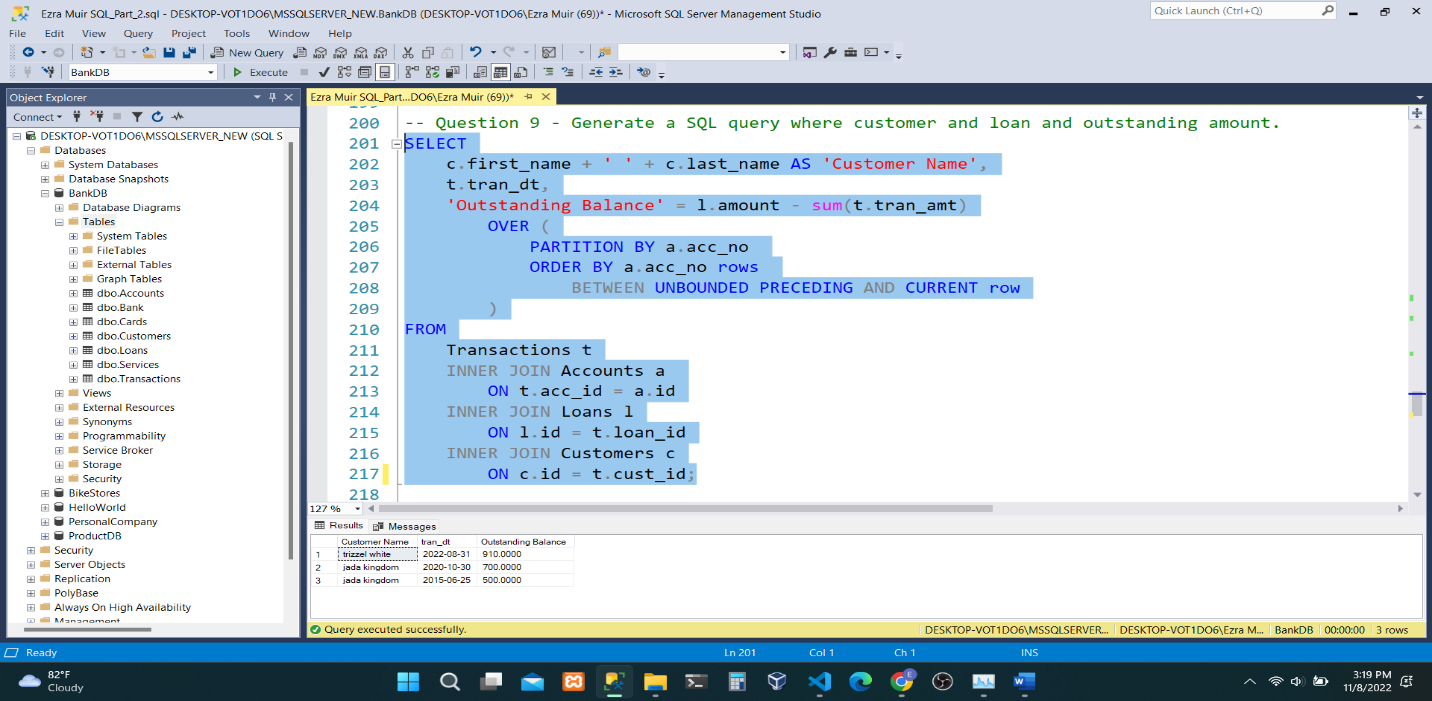
ON t.acc\_id = a.id

INNER JOIN Loans l

ON l.id = t.loan\_id

INNER JOIN Customers c

ON c.id = t.cust\_id



-- Question 10 - Generate a SQL query with all details like Customer, account, type, Balance amount based on the transaction.

SELECT

t.id 'Transaction ID',

c.first\_name 'First Name',

c.last\_name 'Last Name',

a.acc\_type 'Account Type',

a.balance 'Balance'

FROM

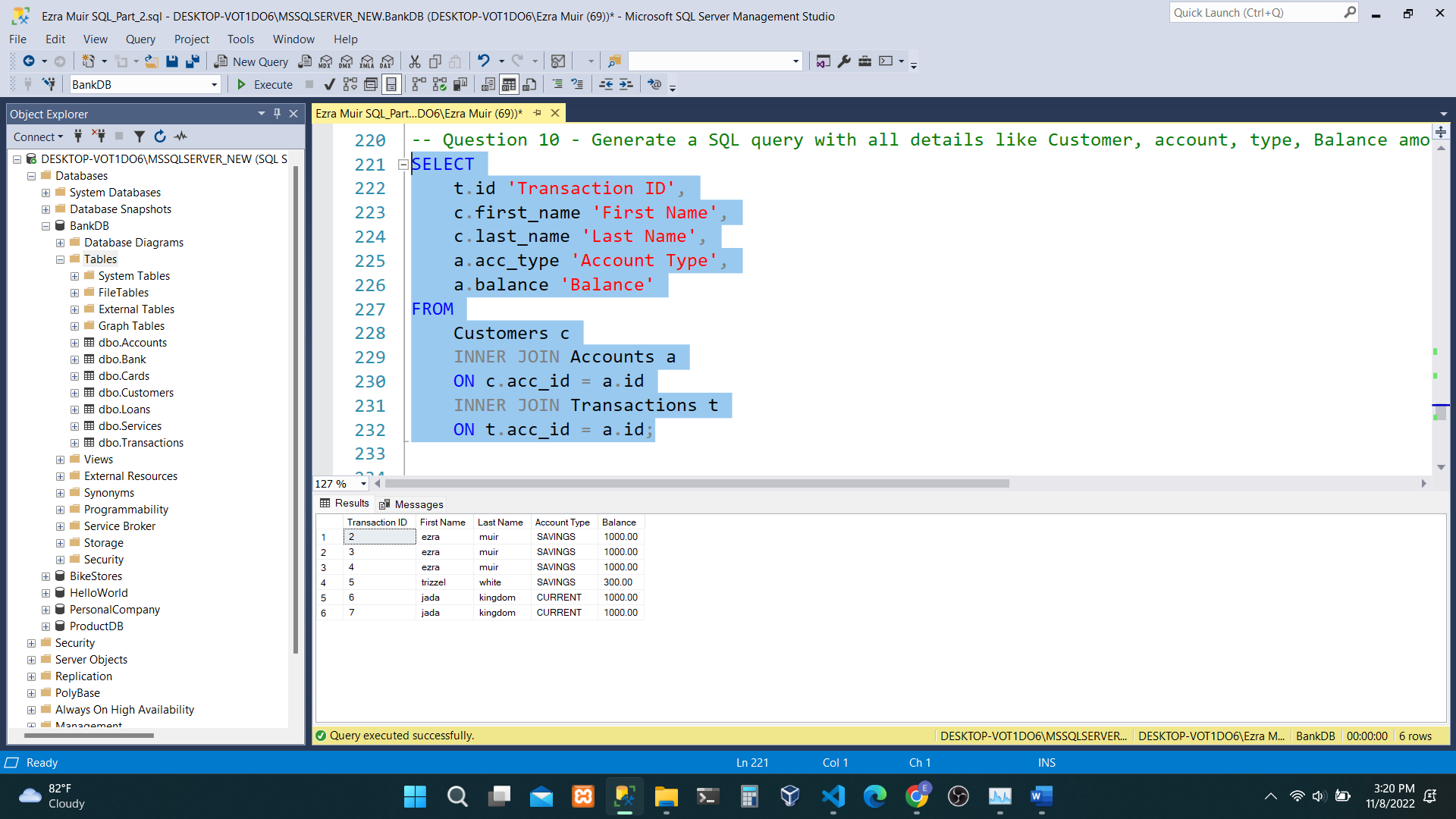
Customers c

INNER JOIN Accounts a

ON c.acc\_id = a.id

INNER JOIN Transactions t

ON t.acc\_id = a.id;



-- Question 11 - Generate a SQL query with loan details and history of payments done by the customer and outstanding amount to be paid.

SELECT

c.first\_name 'First Name',

c.last\_name 'Last Name',

l.id 'Loan ID',

l.amount 'Loan Amount',

t.id 'Transaction ID',

t.acc\_id 'Account ID',

t.cust\_id 'Customer ID',

t.loan\_id 'Loan ID',

t.tran\_amt 'Transaction Amount',

t.tran\_dt 'Transaction Date'

FROM

Customers c

INNER JOIN Loans l

ON c.loan\_id = l.id

INNER JOIN Transactions t

ON t.id = l.id

