**Final Project**

/\* CREATE DATABASE "SmartBank"

WITH

OWNER = postgres

ENCODING = 'UTF8'

LOCALE\_PROVIDER = 'libc'

CONNECTION LIMIT = -1

IS\_TEMPLATE = False;\*/

-- 1) Creating Schema CustomerAccount

CREATE SCHEMA CustomerAccount

AUTHORIZATION postgres;

-- 2) Creating Schema Customer\_Offerings

CREATE SCHEMA Customer\_Offerings

AUTHORIZATION postgres;

-- 3) Creating Schema Branch\_Operations

CREATE SCHEMA Branch\_Operations

AUTHORIZATION postgres;

-- 1) Creating Customers Table

create table CustomerAccount.Customer

(Customer\_ID int primary key,

Customer\_Name varchar(100) not null,

Customer\_Address varchar(100) not null,

Phone\_Number char(10) unique not null,

Customer\_Email varchar(100) unique,

Customer\_DOB date not null,

Customer\_NationalID varchar(20) not null unique);

-- 2) Creating Accounts Table

create table customeraccount.Accounts(

Account\_ID int primary key,

Customer\_ID int not null,

Account\_Type varchar(20) not null,

Balance Decimal(15,2) not null,

Opening\_Date Date,

Branch\_Name Varchar(50)not null,

foreign key(Customer\_ID) references customeraccount.customer(Customer\_ID)

);

select \* from customeraccount.Accounts;

-- 3) Creating Transactions Table

create table customeraccount.Transactions(

Transaction\_ID int primary key,

Account\_ID int not null,

Transaction\_Date Timestamp not null,

Transaction\_Type varchar(30) not null,

Amount decimal(15,2) not null,

Transaction\_Mode varchar(30) not null,

Remarks Varchar(100),

foreign key(Account\_ID) References customeraccount.Accounts(Account\_ID)

);

-- 4) Create Laons Table

create table customer\_offerings.Loans(

Loan\_ID int primary key,

Customer\_ID int not null,

Loan\_Type varchar(30) not null not null,

Loan\_Amount decimal(15,2) not null,

Interest\_Rate decimal(5,2) not null,

Start\_Date Date not null,

End\_Date Date not null,

foreign key(Customer\_ID) references customeraccount.customer(customer\_id)

);

-- 5) creating Table Branches

create table branch\_operations.Branches(

Branch\_ID int primary key,

Branch\_Name varchar(100) Not Null,

City varchar(100),

States varchar(100),

Zip\_Code bigint,

Manager\_Name varchar(100),

Established\_Date Date

);

-- 6) Creating Employee Table

create table branch\_operations.Employees(

Employee\_ID int primary key,

Employee\_Name varchar(100),

Positions varchar(100) not null,

Branch\_ID int not null,

Hire\_Date date,

Salary decimal(15, 2) not null,

Contact\_Number char(10) not null,

Email varchar(100),

foreign key(Branch\_ID) references branch\_operations.Branches(Branch\_ID)

);

-- 7) Creating Product Table

create table customer\_offerings.Products(

Product\_ID int primary key,

Product\_Name varchar(100) not null,

Product\_Type varchar(100) not null,

Interest\_Rate decimal(5,2) not null,

Minimum\_Balance decimal(15,2) not null,

Bank\_ID int not null,

Branch\_ID int not null,

Launch\_Date date,

foreign key(Branch\_ID) references branch\_operations.Branches(Branch\_ID)

)

-- 8) Creating Documents Table

create table customeraccount.Documents(

Document\_ID int primary key,

Document\_Type varchar(100),

Associated\_With varchar(100) not null,

Associated\_ID int not null,

Upload\_Date date,

Expiry\_Date date,

Verified Varchar(10) not null,

foreign key(Associated\_ID) references customeraccount.customer(customer\_id)

);

select \* from customeraccount.documents;

-- 9) Creating Audit Logs table

create table Branch\_Operations.Audit\_Logs(

Log\_ID int primary key,

Entity\_Type varchar(50),

Employee\_ID int not null,

Action\_Type varchar(50),

User\_ID int not null,

Date\_Time timestamp,

foreign key(Employee\_ID) references branch\_operations.Employees(Employee\_ID)

);

-- 10) Creating Cards Table

create table customeraccount.Cards(

Card\_ID int Primary key,

Account\_ID int not null,

Card\_Type varchar(20)not null,

Card\_Number varchar(16) not null,

Card\_Expiry\_Date date not null,

CVV char(3) not null,

Issued\_Date date not null,

Card\_Status Varchar(10) not null,

foreign key(Account\_ID) References customeraccount.Accounts(Account\_ID)

);

-- 11) Creating Insurance Table

create table customer\_offerings.Insurance(

Policy\_ID int primary key,

Customer\_ID int not null,

Policy\_Type varchar(20),

Premium\_Amount decimal(15,2) not null,

Coverage\_Amount decimal(15,2) not null,

Start\_Date Date not null,

End\_Date Date not null,

Policy\_Status varchar(20),

foreign key(Customer\_ID) references customeraccount.customer(customer\_id)

);

-- Data Entery

-- 1) Customers Table

INSERT INTO CustomerAccount.Customer(Customer\_ID, Customer\_Name, Customer\_Address, Phone\_Number,

Customer\_Email, Customer\_DOB, Customer\_NationalID) VALUES

(1, 'John Doe', '123 Elm Street, New York', '9876543210', 'johndoe@example.com', '1985-02-15', 'S854321987'),

(2, 'Jane Smith', '456 Maple Avenue, Chicago', '8765432109', 'janesmith@example.com', '1990-08-22', 'B987654321'),

(3, 'Mike Johnson', '789 Pine Lane, Houston', '7654321098', 'mikejohnson@example.com', '1982-11-10', 'C654321987'),

(4, 'Emily Davis', '321 Oak Boulevard, San Francisco', '6543210987', 'emilydavis@example.com', '1995-03-05', 'D321098765'),

(5, 'David Brown', '654 Cedar Road, Miami', '5432109876', 'davidbrown@example.com', '1988-07-19', 'E543210987'),

(6, 'Sophia Wilson', '987 Birch Lane, Boston', '4321098765', 'sophiawilson@example.com', '1993-01-12', 'F654789321'),

(7, 'James Taylor', '222 Willow Way, Seattle', '3210987654', 'jamestaylor@example.com', '1987-12-25', 'G987321654'),

(8, 'Olivia Martinez', '333 Redwood Drive, Denver', '2109876543', 'oliviamartinez@example.com', '1999-09-30', 'H321654987'),

(9, 'Lucas Anderson', '888 Cypress Avenue, Atlanta', '1987654321', 'lucasanderson@example.com', '1992-04-18', 'I654123987'),

(10, 'Amelia Thompson', '777 Magnolia Street, Dallas', '5678901234', 'ameliathompson@example.com', '1983-08-08', 'J456123789'),

(11, 'Evelyn Murphy', '123 Maple Street, Austin', '6781234560', 'evelynmurphy@example.com', '1993-06-21', 'K123456780'),

(12, 'Samuel Hughes', '234 Oak Avenue, Dallas', '7892345671', 'samuelhughes@example.com', '1991-02-15', 'L234567891'),

(13, 'Victoria Rogers', '345 Pine Road, Denver', '8903456782', 'victoriarogers@example.com', '1985-08-09', 'M345678912'),

(14, 'Sebastian Cook', '456 Cedar Lane, Atlanta', '9014567893', 'sebastiancook@example.com', '1998-11-13', 'N456789123'),

(15, 'Zoe Campbell', '567 Willow Way, Chicago', '1235678904', 'zoecampbell@example.com', '1994-05-17', 'O567890234'),

(16, 'Jack Sanders', '678 Redwood Avenue, Miami', '2346789015', 'jacksanders@example.com', '1990-03-11', 'P678901345'),

(17, 'Lillian Cooper', '789 Spruce Drive, Seattle', '3457890126', 'lilliancooper@example.com', '1989-09-25', 'Q789012456'),

(18, 'Daniel Morgan', '890 Sycamore Boulevard, San Diego', '4568901237', 'danielmorgan@example.com', '1992-07-08', 'R890123567'),

(19, 'Hannah Jenkins', '901 Magnolia Street, Phoenix', '5679012348', 'hannahjenkins@example.com', '1996-12-30', 'S901234678'),

(20, 'Ryan Foster', '101 Birch Lane, Houston', '6780123459', 'ryanfoster@example.com', '1993-04-05', 'T012345789'),

(21, 'Leah Patterson', '202 Cypress Avenue, Austin', '7891234567', 'leahpatterson@example.com', '1990-10-14', 'U123456890'),

(22, 'Aaron Ross', '303 Maple Boulevard, Chicago', '8902345678', 'aaronross@example.com', '1988-02-28', 'V234567901'),

(23, 'Madison Howard', '404 Pine Drive, San Francisco', '9013456789', 'madisonhoward@example.com', '1995-01-09', 'W345678912'),

(24, 'Henry Ramirez', '505 Oak Street, Dallas', '1234567890', 'henryramirez@example.com', '1987-03-23', 'X456789023'),

(25, 'Elizabeth Ward', '606 Cedar Avenue, Houston', '2345678901', 'elizabethward@example.com', '1991-06-19', 'Y567890134'),

(26, 'Lucas Bailey', '707 Willow Lane, New York', '3456789012', 'lucasbailey@example.com', '1986-09-12', 'Z678901245'),

(27, 'Scarlett Brooks', '808 Redwood Road, Boston', '4567890123', 'scarlettbrooks@example.com', '1998-11-03', 'AA789012356'),

(28, 'Eli Gray', '909 Spruce Avenue, Denver', '5678904321', 'eligray@example.com', '1989-04-27', 'BB890123467'),

(29, 'Natalie Wood', '1010 Sycamore Way, Phoenix', '6789012345', 'nataliewood@example.com', '1994-07-06', 'CC901234578'),

(30, 'Carter Bell', '1011 Magnolia Lane, Atlanta', '7890123456', 'carterbell@example.com', '1990-01-19', 'DD012345689'),

(31, 'Chloe Martinez', '123 Oak Lane, New York', '8901230061', 'chloemartinez@example.com', '1995-07-19', 'EE123456781'),

(32, 'Isaac James', '234 Maple Boulevard, Boston', '7892345672', 'isaacjames@example.com', '1993-11-22', 'FF987654322'),

(33, 'Avery Lopez', '345 Cedar Street, Atlanta', '6783456783', 'averylopez@example.com', '1991-02-13', 'GG654321983'),

(34, 'Grace Carter', '456 Spruce Avenue, Dallas', '5674567894', 'gracecarter@example.com', '1990-04-15', 'HH321098764'),

(35, 'Owen Wilson', '567 Birch Drive, Chicago', '4565678568', 'owenwilson@example.com', '1992-08-12', 'II543210988'),

(36, 'Sophia Anderson', '678 Cypress Road, Miami', '3456789016', 'sophiaanderson@example.com', '1996-06-10', 'JJ654789324'),

(37, 'Ethan Perez', '789 Pine Lane, Houston', '2347890127', 'ethanperez@example.com', '1989-03-05', 'KK987321655'),

(38, 'Zoe Scott', '890 Magnolia Avenue, Phoenix', '1238901238', 'zoescott@example.com', '1988-12-11', 'LL321654988'),

(39, 'Luke Murphy', '901 Redwood Boulevard, Seattle', '8901234569', 'lukemurphy@example.com', '1990-09-14', 'MM654123978'),

(40, 'Scarlett Lee', '101 Spruce Street, Denver', '7892345678', 'scarlettlee@example.com', '1987-01-24', 'NN456123781'),

(41, 'Ella Young', '202 Cedar Avenue, San Francisco', '6783456787', 'ellayoung@example.com', '1994-05-18', 'OO789654322'),

(42, 'Mason Brown', '303 Oak Way, Dallas', '5674567896', 'masonbrown@example.com', '1991-03-30', 'PP321765498'),

(43, 'Lily Ramirez', '404 Birch Road, New York', '4565678905', 'lilyramirez@example.com', '1993-06-29', 'QQ123654789'),

(44, 'Jack Martinez', '505 Pine Lane, Atlanta', '3456789014', 'jackmartinez@example.com', '1986-08-04', 'RR543210965'),

(45, 'Grace Foster', '606 Maple Boulevard, Houston', '2347890123', 'gracefoster@example.com', '1998-07-21', 'SS765432109'),

(46, 'Henry Hill', '707 Willow Avenue, Seattle', '1238901232', 'henryhill@example.com', '1992-10-25', 'TT654321908'),

(47, 'Amelia Baker', '808 Redwood Drive, Phoenix', '8901234561', 'ameliabaker@example.com', '1995-04-14', 'UU543219876'),

(48, 'Oliver Reed', '909 Spruce Lane, Dallas', '7892345670', 'oliverreed@example.com', '1988-08-18', 'VV765432198'),

(49, 'Harper Morgan', '1010 Cedar Way, Boston', '6783456789', 'harpermorgan@example.com', '1987-03-22', 'WW432109876'),

(50, 'Logan Bennett', '1011 Birch Avenue, Denver', '5674567891', 'loganbennett@example.com', '1990-09-11', 'XX123456789'),

(51, 'Aria Wood', '123 Spruce Street, New York', '4565678902', 'ariawood@example.com', '1994-01-03', 'YY456789012'),

(52, 'Noah Bell', '234 Maple Boulevard, Miami', '3456789013', 'noahbell@example.com', '1989-11-10', 'ZZ567890123'),

(53, 'Luna Gray', '345 Cedar Lane, San Francisco', '2347890124', 'lunagray@example.com', '1985-06-15', 'AA678901234'),

(54, 'Henry Carter', '456 Pine Way, Atlanta', '1238901235', 'henrycarter@example.com', '1991-07-09', 'BB789012345'),

(55, 'Emily Ross', '567 Birch Avenue, Houston', '8901234566', 'emilyross@example.com', '1993-09-25', 'CC890123456'),

(56, 'James Rivera', '678 Magnolia Street, Boston', '7892345677', 'jamesrivera@example.com', '1988-03-12', 'DD901234567'),

(57, 'Chloe Cooper', '789 Willow Drive, Chicago', '6783456788', 'chloecooper@example.com', '1996-12-06', 'EE012345678'),

(58, 'Mila King', '890 Redwood Boulevard, Seattle', '5674567899', 'milaking@example.com', '1989-05-22', 'FF123456789'),

(59, 'Ethan Howard', '901 Cedar Way, Denver', '4565678900', 'ethanhoward@example.com', '1990-10-01', 'GG234567890'),

(60, 'Sophia Adams', '101 Birch Lane, Phoenix', '3456789011', 'sophiaadams@example.com', '1992-02-14', 'HH345678901'),

(61, 'Ella Carter', '101 Spruce Street, New York', '8901234567', 'ellacarter@example.com', '1994-01-12', 'AA123456781'),

(62, 'Henry Turner', '202 Sycamore Lane, Chicago', '7800123456', 'henryturner@example.com', '1989-05-22', 'BB987654322'),

(63, 'Liam Scott', '303 Maple Avenue, Houston', '6790123450', 'liamscott@example.com', '1991-03-11', 'CC654321983'),

(64, 'Grace Hill', '404 Pine Road, San Francisco', '5778901234', 'gracehill@example.com', '1986-07-05', 'DD321098764'),

(65, 'Mason Adams', '505 Birch Drive, Miami', '4567190123', 'masonadams@example.com', '1988-02-19', 'EE543210988'),

(66, 'Ava White', '606 Cedar Lane, Boston', '4456789012', 'avawhite@example.com', '1992-08-15', 'FF654789324'),

(67, 'Ethan Harris', '707 Willow Way, Seattle', '2340678901', 'ethanharris@example.com', '1995-04-20', 'GG987321655'),

(68, 'Sophia Clark', '808 Oak Avenue, Denver', '1234567899', 'sophiaclark@example.com', '1990-11-10', 'HH321654988'),

(69, 'Noah Lewis', '909 Redwood Street, Atlanta', '9876543211', 'noahlewis@example.com', '1984-06-25', 'II654123978'),

(70, 'Harper King', '1001 Cypress Road, Dallas', '8765432102', 'harperking@example.com', '1997-09-15', 'JJ456123781'),

(71, 'Alexander Bennett', '1002 Magnolia Lane, Phoenix', '8754321098', 'alexanderbennett@example.com', '1985-03-10', 'KK789654322'),

(72, 'Isabella Young', '1003 Spruce Drive, New York', '6043210987', 'isabellayoung@example.com', '1998-07-30', 'LL321765498'),

(73, 'Matthew Baker', '1004 Sycamore Boulevard, Chicago', '6432109876', 'matthewbaker@example.com', '1996-01-12', 'MM123654789'),

(74, 'Charlotte Rivera', '1005 Maple Avenue, Houston', '7821098765', 'charlotterivera@example.com', '1991-12-01', 'NN543210965'),

(75, 'Daniel Perez', '1006 Pine Street, San Francisco', '3256987654', 'danielperez@example.com', '1992-10-18', 'OO765432109'),

(76, 'Mia Brooks', '1007 Birch Avenue, Miami', '2109076543', 'miabrooks@example.com', '1993-08-05', 'PP654321908'),

(77, 'Logan Foster', '1008 Cedar Road, Boston', '1987784321', 'loganfoster@example.com', '1989-03-25', 'QQ543219876'),

(78, 'Abigail Reed', '1009 Willow Street, Seattle', '2398765432', 'abigailreed@example.com', '1994-06-10', 'RR765432198'),

(79, 'Benjamin Gray', '1010 Oak Avenue, Denver', '9076543201', 'benjamingray@example.com', '1987-11-05', 'SS432109876'),

(80, 'Emily Morris', '1011 Redwood Drive, Atlanta', '8745432198', 'emilymorris@example.com', '1995-02-14', 'TT123456789');

select \* from CustomerAccount.Customer;

-- Accounts Table

INSERT INTO customeraccount.Accounts (Account\_ID, Customer\_ID, Account\_Type, Balance, Opening\_Date, Branch\_Name) VALUES

(1, 1, 'Savings', 2500.75, '2020-01-15', 'New York'),

(2, 2, 'Current', 5000.00, '2019-07-20', 'Chicago'),

(3, 3, 'Savings', 12000.50, '2021-03-12', 'Houston'),

(4, 4, 'Savings', 750.00, '2022-06-10', 'San Francisco'),

(5, 5, 'Current', 9200.00, '2020-11-25', 'Miami'),

(6, 6, 'Savings', 6300.30, '2018-09-18', 'Boston'),

(7, 7, 'Current', 8300.10, '2023-02-22', 'Seattle'),

(8, 8, 'Savings', 2800.50, '2021-08-13', 'Denver'),

(9, 9, 'Current', 4500.90, '2020-04-15', 'Atlanta'),

(10, 10, 'Savings', 1500.00, '2019-12-01', 'Dallas'),

(11, 11, 'Savings', 3200.75, '2020-05-20', 'Austin'),

(12, 12, 'Current', 7200.00, '2022-03-05', 'Dallas'),

(13, 13, 'Savings', 8900.10, '2019-06-30', 'Denver'),

(14, 14, 'Current', 1500.50, '2021-01-12', 'Atlanta'),

(15, 15, 'Savings', 5600.00, '2020-10-10', 'Chicago'),

(16, 16, 'Savings', 2300.00, '2022-02-25', 'New York'),

(17, 17, 'Current', 9700.90, '2019-07-07', 'San Diego'),

(18, 18, 'Savings', 8100.50, '2021-11-22', 'Phoenix'),

(19, 19, 'Current', 4900.75, '2020-03-18', 'Houston'),

(20, 20, 'Savings', 2500.00, '2023-01-01', 'San Francisco'),

(21, 21, 'Savings', 3200.00, '2020-08-14', 'New York'),

(22, 22, 'Current', 4100.50, '2021-07-22', 'Chicago'),

(23, 23, 'Savings', 5900.75, '2020-10-05', 'Houston'),

(24, 24, 'Savings', 8600.00, '2022-04-16', 'San Francisco'),

(25, 25, 'Current', 6700.20, '2019-06-30', 'Miami'),

(26, 26, 'Savings', 2400.50, '2021-12-01', 'Boston'),

(27, 27, 'Current', 7500.00, '2023-03-10', 'Seattle'),

(28, 28, 'Savings', 1900.90, '2020-11-18', 'Denver'),

(29, 29, 'Current', 8900.00, '2019-09-24', 'Atlanta'),

(30, 30, 'Savings', 4500.50, '2021-02-28', 'Dallas'),

(31, 31, 'Savings', 7800.20, '2022-07-05', 'Austin'),

(32, 32, 'Current', 6300.00, '2020-06-12', 'Dallas'),

(33, 33, 'Savings', 5400.75, '2019-08-15', 'Denver'),

(34, 34, 'Current', 2900.50, '2021-05-20', 'Atlanta'),

(35, 35, 'Savings', 8700.00, '2020-09-11', 'Chicago'),

(36, 36, 'Savings', 6100.50, '2022-01-30', 'New York'),

(37, 37, 'Current', 9200.00, '2021-06-14', 'San Diego'),

(38, 38, 'Savings', 7600.50, '2019-10-09', 'Phoenix'),

(39, 39, 'Current', 3400.75, '2020-04-21', 'Houston'),

(40, 40, 'Savings', 5800.00, '2022-08-31', 'San Francisco'),

(41, 41, 'Savings', 6900.50, '2021-09-25', 'New York'),

(42, 42, 'Current', 4500.00, '2020-12-15', 'Chicago'),

(43, 43, 'Savings', 8100.75, '2019-03-28', 'Houston'),

(44, 44, 'Savings', 2500.50, '2021-10-19', 'San Francisco'),

(45, 45, 'Current', 7100.00, '2020-07-04', 'Miami'),

(46, 46, 'Savings', 3600.90, '2022-02-10', 'Boston'),

(47, 47, 'Current', 5200.50, '2019-11-22', 'Seattle'),

(48, 48, 'Savings', 4900.75, '2020-05-09', 'Denver'),

(49, 49, 'Current', 6600.00, '2021-04-12', 'Atlanta'),

(50, 50, 'Savings', 3100.50, '2023-01-20', 'Dallas'),

(51, 51, 'Savings', 4300.25, '2021-05-14', 'Los Angeles'),

(52, 52, 'Current', 7200.50, '2020-09-22', 'Chicago'),

(53, 53, 'Savings', 5600.75, '2019-08-05', 'Houston'),

(54, 54, 'Savings', 8300.00, '2022-03-16', 'San Francisco'),

(55, 55, 'Current', 6800.40, '2020-11-30', 'Miami'),

(56, 56, 'Savings', 2400.60, '2021-04-01', 'Boston'),

(57, 57, 'Current', 7900.00, '2018-12-10', 'Seattle'),

(58, 58, 'Savings', 2700.85, '2020-01-18', 'Denver'),

(59, 59, 'Current', 8400.00, '2021-08-24', 'Atlanta'),

(60, 60, 'Savings', 4700.50, '2022-05-28', 'Dallas'),

(61, 61, 'Savings', 6800.35, '2019-09-05', 'Austin'),

(62, 62, 'Current', 6500.00, '2020-02-12', 'Dallas'),

(63, 63, 'Savings', 5600.75, '2018-10-15', 'Denver'),

(64, 64, 'Current', 3100.40, '2022-06-20', 'Atlanta'),

(65, 65, 'Savings', 8700.20, '2020-07-11', 'Chicago'),

(66, 66, 'Savings', 6400.50, '2021-12-30', 'New York'),

(67, 67, 'Current', 9200.90, '2019-04-14', 'San Diego'),

(68, 68, 'Savings', 7100.50, '2020-03-09', 'Phoenix'),

(69, 69, 'Current', 3200.85, '2021-11-21', 'Houston'),

(70, 70, 'Savings', 5800.00, '2022-09-30', 'San Francisco'),

(71, 71, 'Savings', 7100.50, '2020-10-25', 'Los Angeles'),

(72, 72, 'Current', 4600.00, '2021-01-15', 'Chicago'),

(73, 73, 'Savings', 8200.75, '2020-02-28', 'Houston'),

(74, 74, 'Savings', 2900.50, '2022-11-19', 'San Francisco'),

(75, 75, 'Current', 7100.80, '2020-06-04', 'Miami'),

(76, 76, 'Savings', 3700.90, '2021-03-10', 'Boston'),

(77, 77, 'Current', 5300.50, '2019-08-22', 'Seattle'),

(78, 78, 'Savings', 5000.75, '2020-04-09', 'Denver'),

(79, 79, 'Current', 6800.10, '2021-07-12', 'Atlanta'),

(80, 80, 'Savings', 3200.50, '2023-02-20', 'Dallas'),

(81, 10, 'Savings', 4500.25, '2021-03-12', 'New York'),

(82, 10, 'Current', 7800.50, '2022-08-15', 'New York'),

(83, 20, 'Savings', 3100.40, '2019-05-22', 'Chicago'),

(84, 20, 'Current', 5400.75, '2021-11-05', 'Chicago'),

(85, 35, 'Savings', 8600.00, '2020-03-18', 'San Francisco'),

(86, 35, 'Current', 9200.85, '2022-09-30', 'San Francisco'),

(87, 42, 'Savings', 7100.90, '2021-02-14', 'Houston'),

(88, 42, 'Current', 4200.35, '2023-06-19', 'Houston'),

(89, 42, 'Savings', 3300.50, '2020-12-01', 'Austin'),

(90, 58, 'Savings', 5300.75, '2018-07-10', 'Denver'),

(91, 58, 'Current', 7800.20, '2022-05-25', 'Denver'),

(92, 58, 'Savings', 8900.00, '2023-01-12', 'Miami'),

(93, 63, 'Savings', 6100.40, '2020-06-14', 'Boston'),

(94, 63, 'Current', 2500.75, '2022-10-08', 'Boston'),

(95, 76, 'Savings', 7100.50, '2021-01-22', 'Los Angeles'),

(96, 76, 'Current', 4900.20, '2019-09-15', 'Los Angeles'),

(97, 76, 'Savings', 3300.85, '2022-04-30', 'Seattle'),

(98, 80, 'Savings', 4700.60, '2020-07-05', 'Atlanta'),

(99, 80, 'Current', 9200.10, '2021-12-12', 'Atlanta'),

(100, 80, 'Savings', 8200.90, '2023-02-25', 'Phoenix'),

(101, 15, 'Savings', 3600.40, '2020-03-18', 'Dallas'),

(102, 15, 'Current', 5800.30, '2022-01-20', 'Dallas'),

(103, 15, 'Savings', 4400.70, '2021-08-22', 'Austin'),

(104, 29, 'Savings', 5600.20, '2019-10-10', 'San Diego'),

(105, 29, 'Current', 8700.50, '2023-03-11', 'San Diego'),

(106, 52, 'Savings', 3800.25, '2020-05-09', 'Chicago'),

(107, 52, 'Current', 7500.90, '2021-11-06', 'Chicago'),

(108, 52, 'Savings', 6200.40, '2022-07-15', 'Houston'),

(109, 40, 'Savings', 3100.80, '2019-12-20', 'Phoenix'),

(110, 40, 'Current', 8200.50, '2023-04-01', 'Phoenix');

select \* from customeraccount.Accounts;

-- 3) Transaction Data

INSERT INTO customeraccount.Transactions

(Transaction\_ID, Account\_ID, Transaction\_Date, Transaction\_Type, Amount, Transaction\_Mode, Remarks) VALUES

(1, 2, '2023-01-15', 'Deposit', 1500.00, 'Branch', 'Initial Deposit'),

(2, 2, '2023-02-10', 'Withdrawal', 500.00, 'ATM', 'ATM Cash Withdrawal'),

(3, 4, '2023-03-01', 'Deposit', 750.00, 'Mobile', 'Mobile Transfer'),

(4, 4, '2023-03-20', 'Withdrawal', 300.00, 'Branch', 'Utility Bill Payment'),

(5, 5, '2022-11-25', 'Deposit', 2500.00, 'Online', 'Online Transfer'),

(6, 5, '2023-01-10', 'Withdrawal', 1000.00, 'ATM', 'ATM Cash Withdrawal'),

(7, 5, '2023-02-15', 'Deposit', 2000.00, 'Branch', 'Salary Credit'),

(8, 7, '2023-04-05', 'Deposit', 3000.00, 'Online', 'Savings Deposit'),

(9, 7, '2023-05-01', 'Withdrawal', 500.00, 'Mobile', 'Mobile Bill Payment'),

(10, 10, '2022-12-15', 'Deposit', 1500.00, 'ATM', 'Initial Deposit'),

(11, 12, '2023-06-20', 'Withdrawal', 750.00, 'Branch', 'Rent Payment'),

(12, 14, '2023-02-25', 'Deposit', 2000.00, 'Online', 'Online Transfer'),

(13, 14, '2023-03-15', 'Deposit', 1000.00, 'Branch', 'Salary Credit'),

(14, 16, '2022-12-30', 'Deposit', 2300.00, 'Mobile', 'Savings Deposit'),

(15, 18, '2023-01-12', 'Withdrawal', 900.00, 'ATM', 'ATM Cash Withdrawal'),

(16, 18, '2023-02-22', 'Deposit', 3000.00, 'Branch', 'Bonus Credit'),

(17, 20, '2023-01-25', 'Deposit', 2500.00, 'Online', 'Savings Deposit'),

(18, 22, '2023-03-10', 'Withdrawal', 1500.00, 'Branch', 'Utility Bill Payment'),

(19, 24, '2023-04-15', 'Deposit', 4000.00, 'ATM', 'Initial Deposit'),

(20, 25, '2023-02-08', 'Deposit', 2200.00, 'Online', 'Savings Credit'),

(21, 25, '2023-03-10', 'Withdrawal', 1500.00, 'ATM', 'ATM Cash Withdrawal'),

(22, 28, '2023-03-18', 'Deposit', 1900.00, 'Branch', 'Initial Deposit'),

(23, 30, '2023-05-05', 'Withdrawal', 1200.00, 'Online', 'Bill Payment'),

(24, 35, '2023-06-10', 'Deposit', 5600.00, 'Branch', 'Salary Credit'),

(25, 36, '2023-05-15', 'Deposit', 6100.00, 'Mobile', 'Savings Deposit'),

(26, 37, '2023-03-28', 'Withdrawal', 2000.00, 'ATM', 'Cash Withdrawal'),

(27, 40, '2023-04-22', 'Deposit', 4000.00, 'Branch', 'Initial Deposit'),

(28, 40, '2023-05-12', 'Deposit', 1800.00, 'Online', 'Bonus Credit'),

(29, 45, '2023-06-30', 'Withdrawal', 1000.00, 'Mobile', 'Mobile Bill Payment'),

(30, 50, '2023-02-20', 'Deposit', 3100.00, 'ATM', 'Initial Deposit'),

(31, 50, '2023-04-18', 'Withdrawal', 800.00, 'Branch', 'Utility Bill Payment'),

(32, 52, '2023-03-05', 'Deposit', 2000.00, 'Branch', 'Initial Deposit'),

(33, 52, '2023-04-01', 'Withdrawal', 1500.00, 'ATM', 'Cash Withdrawal'),

(34, 53, '2023-01-25', 'Deposit', 3500.00, 'Online', 'Savings Deposit'),

(35, 55, '2023-02-15', 'Withdrawal', 1200.00, 'Mobile', 'Mobile Bill Payment'),

(36, 55, '2023-03-05', 'Deposit', 4000.00, 'Branch', 'Bonus Credit'),

(37, 56, '2022-12-20', 'Deposit', 2800.00, 'ATM', 'Initial Deposit'),

(38, 56, '2023-01-15', 'Withdrawal', 700.00, 'Mobile', 'Mobile Transfer'),

(39, 57, '2023-02-28', 'Deposit', 1500.00, 'Online', 'Online Transfer'),

(40, 58, '2023-04-10', 'Deposit', 3300.00, 'Branch', 'Salary Credit'),

(41, 58, '2023-05-01', 'Withdrawal', 800.00, 'ATM', 'Utility Payment'),

(42, 59, '2023-03-22', 'Deposit', 2500.00, 'Mobile', 'Savings Credit'),

(43, 60, '2023-02-18', 'Deposit', 2700.00, 'Online', 'Initial Deposit'),

(44, 61, '2023-06-12', 'Withdrawal', 1500.00, 'Branch', 'Loan Payment'),

(45, 62, '2023-01-10', 'Deposit', 3100.00, 'Mobile', 'Savings Deposit'),

(46, 62, '2023-02-05', 'Deposit', 2000.00, 'ATM', 'Bonus Credit'),

(47, 63, '2023-04-20', 'Withdrawal', 1200.00, 'Mobile', 'Mobile Transfer'),

(48, 64, '2023-06-15', 'Deposit', 2900.00, 'Online', 'Salary Credit'),

(49, 66, '2023-03-18', 'Deposit', 3500.00, 'Branch', 'Savings Deposit'),

(50, 66, '2023-05-22', 'Withdrawal', 1000.00, 'ATM', 'Bill Payment'),

(51, 68, '2023-02-25', 'Deposit', 2300.00, 'Mobile', 'Savings Deposit'),

(52, 68, '2023-04-10', 'Deposit', 4500.00, 'Online', 'Bonus Credit'),

(53, 69, '2023-01-15', 'Withdrawal', 2000.00, 'Branch', 'Loan Payment'),

(54, 70, '2023-03-12', 'Deposit', 3200.00, 'ATM', 'Initial Deposit'),

(55, 71, '2023-05-01', 'Withdrawal', 700.00, 'Mobile', 'Mobile Transfer'),

(56, 72, '2023-06-08', 'Deposit', 1700.00, 'Branch', 'Utility Payment'),

(57, 74, '2023-01-22', 'Deposit', 2800.00, 'Online', 'Savings Credit'),

(58, 75, '2023-02-12', 'Deposit', 3000.00, 'ATM', 'Initial Deposit'),

(59, 75, '2023-04-15', 'Withdrawal', 900.00, 'Mobile', 'Mobile Transfer'),

(60, 77, '2023-03-05', 'Deposit', 4000.00, 'Online', 'Salary Credit'),

(61, 78, '2023-01-18', 'Withdrawal', 600.00, 'Branch', 'Utility Bill Payment'),

(62, 79, '2023-02-20', 'Deposit', 4500.00, 'ATM', 'Savings Deposit'),

(63, 79, '2023-05-05', 'Deposit', 5000.00, 'Mobile', 'Bonus Credit'),

(64, 80, '2023-03-01', 'Withdrawal', 1300.00, 'Online', 'Loan Payment'),

(65, 81, '2023-06-10', 'Deposit', 2100.00, 'Branch', 'Utility Payment'),

(66, 82, '2023-02-18', 'Withdrawal', 800.00, 'Mobile', 'Mobile Bill Payment'),

(67, 83, '2023-03-20', 'Deposit', 1800.00, 'ATM', 'Initial Deposit'),

(68, 85, '2023-05-22', 'Deposit', 3600.00, 'Online', 'Savings Credit'),

(69, 86, '2023-04-18', 'Deposit', 2300.00, 'Mobile', 'Bonus Credit'),

(70, 87, '2023-01-15', 'Withdrawal', 900.00, 'Branch', 'Loan Payment');

-- 4) Loans Data

INSERT INTO

customer\_offerings.Loans (Loan\_ID, Customer\_ID, Loan\_Type, Loan\_Amount, Interest\_Rate, Start\_Date, End\_Date) VALUES

(1, 12, 'Home', 150000.00, 5.5, '2020-02-15', '2030-02-15'),

(2, 8, 'Car', 27000.00, 7.0, '2021-06-10', '2026-06-10'),

(3, 15, 'Personal', 18000.00, 9.5, '2022-08-01', '2025-08-01'),

(4, 3, 'Education', 50000.00, 6.0, '2019-09-20', '2029-09-20'),

(5, 10, 'Business', 85000.00, 8.0, '2021-03-15', '2028-03-15'),

(6, 5, 'Home', 190000.00, 4.8, '2020-07-10', '2035-07-10'),

(7, 17, 'Car', 35000.00, 7.2, '2022-11-20', '2027-11-20'),

(8, 20, 'Personal', 20000.00, 10.0, '2023-04-15', '2025-04-15'),

(9, 7, 'Business', 70000.00, 7.5, '2020-01-10', '2027-01-10'),

(10, 2, 'Education', 65000.00, 6.2, '2019-12-01', '2029-12-01'),

(11, 9, 'Home', 140000.00, 5.0, '2020-05-15', '2030-05-15'),

(12, 18, 'Personal', 15000.00, 9.8, '2022-07-10', '2025-07-10'),

(13, 6, 'Business', 95000.00, 7.5, '2021-01-20', '2028-01-20'),

(14, 22, 'Education', 48000.00, 6.3, '2020-02-05', '2030-02-05'),

(15, 11, 'Home', 220000.00, 5.3, '2019-06-15', '2034-06-15'),

(16, 25, 'Car', 33000.00, 7.1, '2021-09-10', '2026-09-10'),

(17, 30, 'Business', 100000.00, 8.2, '2020-10-20', '2027-10-20'),

(18, 4, 'Personal', 25000.00, 9.0, '2023-01-05', '2026-01-05'),

(19, 16, 'Education', 52000.00, 6.0, '2020-03-15', '2030-03-15'),

(20, 19, 'Home', 130000.00, 4.9, '2022-06-10', '2032-06-10'),

(21, 27, 'Car', 31000.00, 7.4, '2021-12-01', '2026-12-01'),

(22, 13, 'Business', 87000.00, 8.0, '2020-08-15', '2027-08-15'),

(23, 21, 'Personal', 18000.00, 9.5, '2023-03-20', '2025-03-20'),

(24, 14, 'Education', 51000.00, 6.1, '2019-11-15', '2029-11-15'),

(25, 1, 'Home', 180000.00, 5.4, '2020-04-10', '2035-04-10'),

(26, 23, 'Car', 26000.00, 7.0, '2021-01-20', '2026-01-20'),

(27, 29, 'Personal', 14000.00, 9.2, '2023-02-10', '2026-02-10'),

(28, 26, 'Business', 110000.00, 7.8, '2020-07-25', '2028-07-25'),

(29, 28, 'Education', 47000.00, 6.4, '2021-03-15', '2031-03-15'),

(30, 31, 'Home', 210000.00, 4.7, '2020-12-05', '2035-12-05'),

(31, 33, 'Car', 34000.00, 7.3, '2022-10-20', '2027-10-20'),

(32, 35, 'Business', 93000.00, 8.5, '2021-07-10', '2028-07-10'),

(33, 32, 'Personal', 21000.00, 9.7, '2023-01-25', '2025-01-25'),

(34, 24, 'Education', 50000.00, 6.2, '2020-02-15', '2030-02-15'),

(35, 36, 'Home', 160000.00, 5.2, '2019-08-01', '2034-08-01'),

(36, 38, 'Car', 40000.00, 7.6, '2021-11-20', '2026-11-20'),

(37, 20, 'Business', 85000.00, 8.1, '2020-03-15', '2027-03-15'),

(38, 37, 'Personal', 22000.00, 10.0, '2023-06-01', '2026-06-01'),

(39, 34, 'Education', 61000.00, 6.0, '2021-04-10', '2031-04-10'),

(40, 40, 'Home', 170000.00, 5.0, '2020-09-15', '2035-09-15'),

(41, 45, 'Home', 185000.00, 4.9, '2021-02-10', '2036-02-10'),

(42, 48, 'Car', 29000.00, 7.1, '2022-05-15', '2027-05-15'),

(43, 50, 'Personal', 22000.00, 9.4, '2023-07-10', '2026-07-10'),

(44, 43, 'Education', 54000.00, 6.3, '2020-09-15', '2030-09-15'),

(45, 53, 'Business', 95000.00, 8.0, '2021-06-20', '2028-06-20'),

(46, 41, 'Home', 200000.00, 5.1, '2019-03-10', '2034-03-10'),

(47, 57, 'Car', 37000.00, 7.5, '2022-10-15', '2027-10-15'),

(48, 60, 'Personal', 17000.00, 9.7, '2023-01-20', '2026-01-20'),

(49, 42, 'Education', 61000.00, 6.0, '2020-07-15', '2030-07-15'),

(50, 55, 'Business', 105000.00, 7.9, '2021-11-10', '2028-11-10'),

(51, 63, 'Home', 210000.00, 5.0, '2020-05-25', '2035-05-25'),

(52, 67, 'Car', 32000.00, 7.3, '2021-04-15', '2026-04-15'),

(53, 65, 'Personal', 19000.00, 9.6, '2022-08-01', '2025-08-01'),

(54, 46, 'Education', 58000.00, 6.2, '2019-10-10', '2029-10-10'),

(55, 62, 'Business', 88000.00, 8.3, '2020-12-20', '2027-12-20'),

(56, 44, 'Home', 150000.00, 4.8, '2021-01-15', '2036-01-15'),

(57, 71, 'Car', 25000.00, 7.4, '2023-06-15', '2028-06-15'),

(58, 73, 'Personal', 13000.00, 10.0, '2022-09-10', '2025-09-10'),

(59, 75, 'Education', 46000.00, 6.4, '2021-03-20', '2031-03-20'),

(60, 80, 'Business', 97000.00, 7.8, '2020-11-05', '2027-11-05');

-- 5) Branches Data

INSERT INTO branch\_operations.Branches (Branch\_ID, Branch\_Name, City, States, Zip\_Code, Manager\_Name, Established\_Date) VALUES

(1, 'New York Main', 'New York', 'NY', '10001', 'John Doe', '1990-05-12'),

(2, 'Chicago Downtown', 'Chicago', 'IL', '60601', 'Emily Smith', '1985-07-20'),

(3, 'Houston Central', 'Houston', 'TX', '77002', 'Michael Brown', '1995-03-15'),

(4, 'San Francisco Bay', 'San Francisco', 'CA', '94103', 'Sarah Johnson', '2000-09-25'),

(5, 'Miami Beach', 'Miami', 'FL', '33139', 'Chris Lee', '1998-11-18'),

(6, 'Boston South', 'Boston', 'MA', '02110', 'Jessica Miller', '2002-01-14'),

(7, 'Seattle North', 'Seattle', 'WA', '98101', 'Daniel Wilson', '1997-06-30'),

(8, 'Denver Tech Center', 'Denver', 'CO', '80202', 'Sophia Martinez', '2005-08-22'),

(9, 'Atlanta Midtown', 'Atlanta', 'GA', '30309', 'David Garcia', '1993-10-10'),

(10, 'Dallas Uptown', 'Dallas', 'TX', '75201', 'Laura Rodriguez', '1992-04-08'),

(11, 'Austin Downtown', 'Austin', 'TX', '78701', 'Kevin Martinez', '2003-12-12'),

(12, 'Phoenix Central', 'Phoenix', 'AZ', '85001', 'Rachel Moore', '1999-07-05'),

(13, 'San Diego Coastal', 'San Diego', 'CA', '92101', 'Ethan White', '1991-03-22'),

(14, 'Los Angeles West', 'Los Angeles', 'CA', '90001', 'Mia Hernandez', '2004-05-16'),

(15, 'Chicago Southside', 'Chicago', 'IL', '60615', 'Oliver Green', '1988-10-18'),

(16, 'Houston Heights', 'Houston', 'TX', '77008', 'Charlotte King', '2001-02-20'),

(17, 'New York East', 'New York', 'NY', '10002', 'Liam Scott', '1994-09-30'),

(18, 'Denver Downtown', 'Denver', 'CO', '80203', 'Grace Young', '1996-01-25'),

(19, 'Miami Downtown', 'Miami', 'FL', '33131', 'Aiden Walker', '2000-11-11'),

(20, 'Boston North', 'Boston', 'MA', '02114', 'Isabella Hall', '1997-08-10'),

(21, 'Seattle Downtown', 'Seattle', 'WA', '98104', 'Elijah Adams', '2006-03-28'),

(22, 'Atlanta Buckhead', 'Atlanta', 'GA', '30305', 'Amelia Nelson', '1995-06-15'),

(23, 'San Francisco Downtown', 'San Francisco', 'CA', '94105', 'Benjamin Baker', '1998-02-14'),

(24, 'Dallas Northpark', 'Dallas', 'TX', '75225', 'Harper Carter', '1990-12-05'),

(25, 'Austin Westlake', 'Austin', 'TX', '78746', 'Jack Rivera', '2002-07-17'),

(26, 'Phoenix East', 'Phoenix', 'AZ', '85008', 'Evelyn Flores', '1993-05-19'),

(27, 'San Diego Central', 'San Diego', 'CA', '92103', 'James Powell', '1996-09-22'),

(28, 'Los Angeles Downtown', 'Los Angeles', 'CA', '90012', 'Sofia Torres', '1999-04-18'),

(29, 'Chicago Northshore', 'Chicago', 'IL', '60640', 'Henry Bennett', '1992-11-20'),

(30, 'Houston West', 'Houston', 'TX', '77077', 'Victoria Brooks', '2005-10-12');

-- 6) Employee Data

INSERT INTO branch\_operations.Employees(

Employee\_ID, Employee\_Name, Positions, Branch\_ID, Hire\_Date, Salary, Contact\_Number, Email) VALUES

(1, 'John Doe', 'Branch Manager', 1, '2015-05-12', 85000, '1234567890', 'j.doe@bank.com'),

(2, 'Emily Smith', 'Branch Manager', 2, '2012-07-20', 87000, '2345678901', 'e.smith@bank.com'),

(3, 'Michael Brown', 'Branch Manager', 3, '2010-03-15', 90000, '3456789012', 'm.brown@bank.com'),

(4, 'Sarah Johnson', 'Branch Manager', 4, '2018-09-25', 86000, '4567890123', 's.johnson@bank.com'),

(5, 'Chris Lee', 'Branch Manager', 5, '2016-11-18', 88000, '5678901234', 'c.lee@bank.com'),

(6, 'Jessica Miller', 'Branch Manager', 6, '2017-01-14', 85000, '6789012345', 'j.miller@bank.com'),

(7, 'Daniel Wilson', 'Branch Manager', 7, '2014-06-30', 87000, '7890123456', 'd.wilson@bank.com'),

(8, 'Sophia Martinez', 'Branch Manager', 8, '2013-08-22', 88000, '8901234567', 's.martinez@bank.com'),

(9, 'David Garcia', 'Branch Manager', 9, '2011-10-10', 89000, '9012345678', 'd.garcia@bank.com'),

(10, 'Laura Rodriguez', 'Branch Manager', 10, '2010-04-08', 91000, '0123456789', 'l.rodriguez@bank.com'),

(11, 'Kevin Martinez', 'Branch Manager', 11, '2015-12-12', 87000, '1235678901', 'k.martinez@bank.com'),

(12, 'Rachel Moore', 'Branch Manager', 12, '2018-07-05', 86000, '2346789012', 'r.moore@bank.com'),

(13, 'Ethan White', 'Branch Manager', 13, '2014-03-22', 89000, '3457890123', 'e.white@bank.com'),

(14, 'Mia Hernandez', 'Branch Manager', 14, '2019-05-16', 85000, '4568901234', 'm.hernandez@bank.com'),

(15, 'Oliver Green', 'Branch Manager', 15, '2013-10-18', 88000, '5679012345', 'o.green@bank.com'),

(16, 'Charlotte King', 'Branch Manager', 16, '2016-02-20', 87000, '6780123456', 'c.king@bank.com'),

(17, 'Liam Scott', 'Branch Manager', 17, '2012-09-30', 89000, '7891234567', 'l.scott@bank.com'),

(18, 'Grace Young', 'Branch Manager', 18, '2015-01-25', 86000, '8902345678', 'g.young@bank.com'),

(19, 'Aiden Walker', 'Branch Manager', 19, '2017-11-11', 85000, '9013456789', 'a.walker@bank.com'),

(20, 'Isabella Hall', 'Branch Manager', 20, '2013-08-10', 87000, '0124567890', 'i.hall@bank.com'),

(21, 'Elijah Adams', 'Branch Manager', 21, '2014-03-28', 88000, '1236789012', 'e.adams@bank.com'),

(22, 'Amelia Nelson', 'Branch Manager', 22, '2011-06-15', 89000, '2347890123', 'a.nelson@bank.com'),

(23, 'Benjamin Baker', 'Branch Manager', 23, '2015-02-14', 86000, '3458901234', 'b.baker@bank.com'),

(24, 'Harper Carter', 'Branch Manager', 24, '2012-12-05', 85000, '4569012345', 'h.carter@bank.com'),

(25, 'Jack Rivera', 'Branch Manager', 25, '2016-07-17', 87000, '5670123456', 'j.rivera@bank.com'),

(26, 'Evelyn Flores', 'Branch Manager', 26, '2010-05-19', 88000, '6781234567', 'e.flores@bank.com'),

(27, 'James Powell', 'Branch Manager', 27, '2013-09-22', 89000, '7892345678', 'j.powell@bank.com'),

(28, 'Sofia Torres', 'Branch Manager', 28, '2018-04-18', 86000, '8903456789', 's.torres@bank.com'),

(29, 'Henry Bennett', 'Branch Manager', 29, '2012-11-20', 85000, '9014567890', 'h.bennett@bank.com'),

(30, 'Victoria Brooks', 'Branch Manager', 30, '2015-10-12', 87000, '0125678901', 'v.brooks@bank.com'),

(31, 'Anna Scott', 'Loan Officer', 3, '2019-04-15', 55000, '2345678902', 'a.scott@bank.com'),

(32, 'Ethan James', 'Loan Officer', 5, '2020-08-01', 56000, '3456789013', 'e.james@bank.com'),

(33, 'Sophia Allen', 'Teller', 8, '2017-01-10', 45000, '4567890124', 's.allen@bank.com'),

(34, 'Michael Wright', 'Account Manager', 10, '2021-03-20', 60000, '5678901235', 'm.wright@bank.com'),

(35, 'Grace Lewis', 'Teller', 14, '2020-12-01', 44000, '6789012346', 'g.lewis@bank.com'),

(36, 'Daniel Harris', 'Customer Support', 20, '2022-05-15', 42000, '7890123457', 'd.harris@bank.com'),

(37, 'Emily Cook', 'Account Manager', 22, '2021-06-22', 58000, '8901234568', 'e.cook@bank.com'),

(38, 'David Price', 'Loan Officer', 26, '2022-08-09', 54000, '9012345679', 'd.price@bank.com'),

(39, 'Ava Brooks', 'Teller', 30, '2023-02-11', 45000, '0123456781', 'a.brooks@bank.com'),

(40, 'Liam Wood', 'Customer Support', 12, '2021-07-14', 43000, '1234567892', 'l.wood@bank.com'),

(41, 'Olivia Davis', 'Teller', 1, '2020-11-15', 44000, '2345678903', 'o.davis@bank.com'),

(42, 'William Carter', 'Loan Officer', 2, '2019-03-11', 55000, '3456789014', 'w.carter@bank.com'),

(43, 'Sophia Turner', 'Account Manager', 3, '2021-06-18', 58000, '4567890125', 's.turner@bank.com'),

(44, 'James Murphy', 'Customer Support', 4, '2020-08-05', 43000, '5678901236', 'j.murphy@bank.com'),

(45, 'Isabella Ross', 'Teller', 5, '2022-09-01', 45000, '6789012347', 'i.ross@bank.com'),

(46, 'Alexander Reed', 'Loan Officer', 6, '2021-10-12', 54000, '7890123458', 'a.reed@bank.com'),

(47, 'Charlotte Cooper', 'Teller', 7, '2018-12-20', 44000, '8901234569', 'c.cooper@bank.com'),

(48, 'Benjamin Evans', 'Account Manager', 8, '2020-04-25', 60000, '9012345671', 'b.evans@bank.com'),

(49, 'Amelia Scott', 'Loan Officer', 9, '2022-03-18', 56000, '0123456782', 'a.scott@bank.com'),

(50, 'Lucas Ramirez', 'Customer Support', 10, '2019-07-08', 42000, '1234567893', 'l.ramirez@bank.com'),

(51, 'Mason Walker', 'Loan Officer', 11, '2020-01-15', 55000, '2345678904', 'm.walker@bank.com'),

(52, 'Ella Perry', 'Teller', 12, '2021-09-10', 45000, '3456789015', 'e.perry@bank.com'),

(53, 'Ethan Howard', 'Customer Support', 13, '2019-02-05', 43000, '4567890126', 'e.howard@bank.com'),

(54, 'Harper Price', 'Account Manager', 14, '2020-11-12', 59000, '5678901237', 'h.price@bank.com'),

(55, 'Noah Rivera', 'Loan Officer', 15, '2021-05-07', 56000, '6789012348', 'n.rivera@bank.com'),

(56, 'Lily Morgan', 'Teller', 16, '2022-08-16', 45000, '7890123459', 'l.morgan@bank.com'),

(57, 'Aiden Bell', 'Account Manager', 17, '2020-10-22', 60000, '8901234561', 'a.bell@bank.com'),

(58, 'Zoe Sanders', 'Teller', 18, '2021-01-10', 44000, '9012345672', 'z.sanders@bank.com'),

(59, 'Emma Long', 'Customer Support', 19, '2022-07-04', 43000, '0123456783', 'e.long@bank.com'),

(60, 'Oliver Ward', 'Loan Officer', 20, '2021-03-15', 55000, '1234567894', 'o.ward@bank.com'),

(61, 'Mia Torres', 'Teller', 21, '2020-12-08', 45000, '2345678905', 'm.torres@bank.com'),

(62, 'Henry Baker', 'Account Manager', 22, '2021-05-19', 59000, '3456789016', 'h.baker@bank.com'),

(63, 'Luna Flores', 'Customer Support', 23, '2022-02-14', 43000, '4567890127', 'l.flores@bank.com'),

(64, 'Samuel King', 'Loan Officer', 24, '2020-06-25', 55000, '5678901238', 's.king@bank.com'),

(65, 'Chloe Nelson', 'Teller', 25, '2021-09-18', 45000, '6789012349', 'c.nelson@bank.com'),

(66, 'Evelyn Cook', 'Account Manager', 26, '2022-03-22', 59000, '7890123451', 'e.cook@bank.com'),

(67, 'Jack Brooks', 'Loan Officer', 27, '2021-04-10', 56000, '8901234562', 'j.brooks@bank.com'),

(68, 'Ava Hughes', 'Teller', 28, '2020-11-01', 44000, '9012345673', 'a.hughes@bank.com'),

(69, 'Logan Reed', 'Customer Support', 29, '2019-08-14', 42000, '0123456784', 'l.reed@bank.com'),

(70, 'Sophia Hayes', 'Account Manager', 30, '2021-06-20', 60000, '1234567895', 's.hayes@bank.com');

-- 7) Product Data

INSERT INTO customer\_offerings.Products

(Product\_ID, Product\_Name, Product\_Type, Interest\_Rate, Minimum\_Balance, Bank\_ID, Branch\_ID, Launch\_Date)

VALUES

(1, 'Basic Savings Account', 'Savings Account', 1.5, 500, 1, 1, '2015-01-15'),

(2, 'Premium Savings Account', 'Savings Account', 2.0, 2000, 1, 2, '2016-03-20'),

(3, 'Standard Checking Account', 'Checking Account', 0.5, 0, 2, 3, '2014-07-10'),

(4, 'Gold Checking Account', 'Checking Account', 1.0, 1000, 2, 4, '2017-05-30'),

(5, 'Fixed Deposit 1 Year', 'Fixed Deposit', 5.0, 10000, 3, 5, '2018-11-15'),

(6, 'Fixed Deposit 3 Years', 'Fixed Deposit', 5.5, 15000, 3, 6, '2019-09-01'),

(7, 'Personal Loan', 'Loan', 10.0, 0, 4, 7, '2020-02-12'),

(8, 'Home Loan', 'Loan', 8.5, 0, 4, 8, '2021-04-25'),

(9, 'Auto Loan', 'Loan', 9.0, 0, 5, 9, '2022-08-05'),

(10, 'Education Loan', 'Loan', 7.5, 0, 5, 10, '2023-03-19'),

(11, 'Corporate Savings Account', 'Savings Account', 2.5, 5000, 6, 11, '2016-06-15'),

(12, 'Student Checking Account', 'Checking Account', 0.2, 0, 6, 12, '2018-08-10'),

(13, 'Recurring Deposit 1 Year', 'Recurring Deposit', 6.0, 1000, 7, 13, '2019-05-30'),

(14, 'Recurring Deposit 5 Years', 'Recurring Deposit', 6.5, 5000, 7, 14, '2020-01-20'),

(15, 'Gold Loan', 'Loan', 12.0, 0, 8, 15, '2021-10-05'),

(16, 'Agricultural Loan', 'Loan', 6.0, 0, 8, 16, '2022-12-01'),

(17, 'Overdraft Account', 'Credit', 15.0, 2000, 9, 17, '2017-03-15'),

(18, 'Credit Card Classic', 'Credit', 18.0, 0, 9, 18, '2018-09-18'),

(19, 'Credit Card Platinum', 'Credit', 20.0, 0, 10, 19, '2019-11-12'),

(20, 'Credit Card Gold', 'Credit', 22.0, 0, 10, 20, '2020-07-05'),

(21, 'Senior Citizens Savings', 'Savings Account', 3.0, 1000, 11, 21, '2020-01-30'),

(22, 'Women Savings Account', 'Savings Account', 2.8, 1000, 11, 22, '2021-04-15'),

(23, 'Business Checking Account', 'Checking Account', 1.2, 5000, 12, 23, '2022-06-10'),

(24, 'Platinum Fixed Deposit', 'Fixed Deposit', 6.0, 20000, 12, 24, '2023-03-01'),

(25, 'Gold Recurring Deposit', 'Recurring Deposit', 7.0, 10000, 13, 25, '2021-09-10'),

(26, 'Startup Business Loan', 'Loan', 10.5, 0, 13, 26, '2022-11-20'),

(27, 'Travel Loan', 'Loan', 11.0, 0, 14, 27, '2023-05-30'),

(28, 'Holiday Savings Plan', 'Savings Account', 1.8, 500, 14, 28, '2019-12-15'),

(29, 'Young Saver Account', 'Savings Account', 2.0, 0, 15, 29, '2020-05-05'),

(30, 'Retirement Savings Plan', 'Savings Account', 3.2, 1000, 15, 30, '2021-02-25');

-- 8) Creating Documents Table

ALTER TABLE customeraccount.documents

ALTER COLUMN Verified TYPE VARCHAR(10);

INSERT INTO customeraccount.documents (Document\_ID, Document\_Type, Associated\_With, Associated\_ID, Upload\_Date, Expiry\_Date, Verified)

VALUES

(1, 'Proof of Identity', 'Customer', 1, '2022-01-15', NULL, 'Yes'),

(2, 'Proof of Address', 'Customer', 2, '2022-02-20', NULL, 'Yes'),

(3, 'Loan Agreement', 'Loan', 80, '2023-03-05', '2026-03-05', 'Yes'),

(4, 'Mortgage Deed', 'Loan', 78, '2022-10-12', '2032-10-12', 'No'),

(5, 'Tax Returns', 'Customer', 3, '2023-06-18', NULL, 'Yes'),

(6, 'Proof of Employment', 'Loan', 13, '2023-04-25', NULL, 'Yes'),

(7, 'Account Opening Form', 'Account', 45, '2022-11-01', NULL, 'Yes'),

(8, 'Signature Specimen', 'Customer', 4, '2022-09-10', NULL, 'Yes'),

(9, 'KYC Form', 'Customer', 5, '2023-01-01', NULL, 'No'),

(10, 'Guarantor Statement', 'Loan', 14, '2023-05-12', NULL, 'Yes'),

(11, 'Proof of Business', 'Loan', 15, '2023-07-20', NULL, 'No'),

(12, 'Insurance Policy', 'Loan', 16, '2022-12-15', '2027-12-15', 'Yes'),

(13, 'Credit Report', 'Customer', 6, '2023-03-10', NULL, 'Yes'),

(14, 'Branch Establishment Certificate', 'Branch', 1, '1990-05-12', NULL, 'Yes'),

(15, 'Branch License', 'Branch', 2, '1985-07-20', '2035-07-20', 'Yes'),

(16, 'Customer Declaration Form', 'Customer', 7, '2023-08-25', NULL, 'No'),

(17, 'Power of Attorney', 'Loan', 17, '2023-02-10', '2030-02-10', 'Yes'),

(18, 'Passport Copy', 'Customer', 8, '2023-09-15', NULL, 'Yes'),

(19, 'PAN Card Copy', 'Customer', 9, '2023-04-01', NULL, 'Yes'),

(20, 'Property Valuation Report', 'Loan', 18, '2023-06-05', NULL, 'No'),

(21, 'Proof of Tax Payment', 'Loan', 29, '2022-11-20', NULL, 'Yes'),

(22, 'Driver’s License', 'Customer', 50, '2023-02-22', NULL, 'Yes'),

(23, 'Loan Pre-Approval Letter', 'Loan', 60, '2023-03-01', NULL, 'Yes'),

(24, 'Bank Statement', 'Customer', 11, '2023-05-15', NULL, 'Yes'),

(25, 'Employment Contract', 'Loan', 41, '2023-01-12', NULL, 'Yes'),

(26, 'Birth Certificate', 'Customer', 12, '2023-07-10', NULL, 'No'),

(27, 'National ID Card', 'Customer', 33, '2023-08-15', NULL, 'Yes'),

(28, 'Business Registration Certificate', 'Loan', 12, '2022-09-30', NULL, 'Yes'),

(29, 'Credit Card Agreement', 'Account', 21, '2023-06-25', NULL, 'No'),

(30, 'Debit Card Agreement', 'Account', 20, '2023-07-10', NULL, 'Yes');

-- 9) Audit Logs Data

INSERT INTO Branch\_Operations.Audit\_Logs

(Log\_ID, Entity\_Type, Employee\_ID, Action\_Type, User\_ID, Date\_Time) VALUES

(1, 'Customer', 12, 'INSERT', 5, '2024-01-10 09:35:00'),

(2, 'Loan', 8, 'UPDATE', 23, '2024-02-12 14:15:00'),

(3, 'Branch', 4, 'DELETE', 34, '2024-03-01 11:45:00'),

(4, 'Transaction', 25, 'INSERT', 16, '2024-03-03 10:30:00'),

(5, 'Document', 14, 'INSERT', 10, '2024-02-25 12:00:00'),

(6, 'Product', 5, 'UPDATE', 7, '2024-02-18 08:45:00'),

(7, 'Customer', 3, 'UPDATE', 2, '2024-01-20 15:15:00'),

(8, 'Loan', 14, 'INSERT', 19, '2024-01-18 13:30:00'),

(9, 'Branch', 15, 'UPDATE', 11, '2024-01-28 09:00:00'),

(10, 'Employee', 20, 'INSERT', 6, '2024-02-05 14:00:00'),

(11, 'Transaction', 18, 'UPDATE', 21, '2024-03-08 10:20:00'),

(12, 'Document', 23, 'DELETE', 3, '2024-03-12 15:10:00'),

(13, 'Product', 9, 'INSERT', 8, '2024-01-22 11:15:00'),

(14, 'Customer', 42, 'DELETE', 14, '2024-02-07 14:30:00'),

(15, 'Loan', 27, 'UPDATE', 24, '2024-03-04 12:50:00'),

(16, 'Branch', 22, 'INSERT', 33, '2024-02-25 13:00:00'),

(17, 'Employee', 25, 'UPDATE', 13, '2024-01-25 09:10:00'),

(18, 'Transaction', 35, 'DELETE', 18, '2024-03-09 10:45:00'),

(19, 'Document', 11, 'INSERT', 9, '2024-01-30 15:30:00'),

(20, 'Customer', 60, 'UPDATE', 28, '2024-02-20 10:00:00'),

(21, 'Loan', 39, 'INSERT', 27, '2024-03-02 09:30:00'),

(22, 'Branch', 28, 'UPDATE', 17, '2024-01-21 11:00:00'),

(23, 'Employee', 45, 'DELETE', 20, '2024-02-08 14:15:00'),

(24, 'Transaction', 48, 'INSERT', 26, '2024-02-12 10:40:00'),

(25, 'Document', 36, 'UPDATE', 4, '2024-02-27 09:50:00'),

(26, 'Product', 13, 'DELETE', 12, '2024-03-06 14:45:00'),

(27, 'Customer', 60, 'INSERT', 15, '2024-02-22 15:25:00'),

(28, 'Loan', 42, 'UPDATE', 25, '2024-03-11 08:30:00'),

(29, 'Branch', 5, 'DELETE', 7, '2024-02-19 12:30:00'),

(30, 'Transaction', 55, 'UPDATE', 9, '2024-03-07 11:30:00'),

(31, 'Document', 41, 'INSERT', 2, '2024-01-26 14:05:00'),

(32, 'Employee', 10, 'UPDATE', 8, '2024-03-05 13:45:00'),

(33, 'Product', 17, 'INSERT', 23, '2024-02-28 09:15:00'),

(34, 'Customer', 35, 'DELETE', 11, '2024-01-23 12:00:00'),

(35, 'Loan', 18, 'UPDATE', 21, '2024-02-21 11:10:00'),

(36, 'Branch', 14, 'INSERT', 19, '2024-02-24 08:45:00'),

(37, 'Transaction', 66, 'UPDATE', 31, '2024-03-10 10:15:00'),

(38, 'Document', 51, 'DELETE', 10, '2024-03-13 12:20:00'),

(39, 'Product', 21, 'INSERT', 14, '2024-02-23 09:25:00'),

(40, 'Customer', 42, 'UPDATE', 24, '2024-03-03 14:50:00'),

(41, 'Loan', 17, 'INSERT', 32, '2024-03-15 10:05:00'),

(42, 'Transaction', 21, 'UPDATE', 29, '2024-03-14 11:20:00'),

(43, 'Branch', 7, 'INSERT', 19, '2024-03-16 09:35:00'),

(44, 'Document', 9, 'UPDATE', 40, '2024-03-17 13:50:00'),

(45, 'Product', 13, 'DELETE', 3, '2024-03-18 15:00:00'),

(46, 'Customer', 5, 'INSERT', 11, '2024-03-19 12:25:00'),

(47, 'Loan', 22, 'UPDATE', 15, '2024-03-20 08:40:00'),

(48, 'Transaction', 8, 'DELETE', 25, '2024-03-21 10:15:00'),

(49, 'Document', 18, 'INSERT', 12, '2024-03-22 16:00:00'),

(50, 'Employee', 10, 'DELETE', 8, '2024-03-23 14:30:00'),

(51, 'Product', 5, 'UPDATE', 23, '2024-03-24 11:10:00'),

(52, 'Loan', 13, 'INSERT', 30, '2024-03-25 09:55:00'),

(53, 'Customer', 22, 'UPDATE', 35, '2024-03-26 14:20:00'),

(54, 'Branch', 3, 'UPDATE', 16, '2024-03-27 10:45:00'),

(55, 'Transaction', 14, 'INSERT', 30, '2024-03-28 08:30:00'),

(56, 'Document', 24, 'UPDATE', 18, '2024-03-29 12:10:00'),

(57, 'Product', 12, 'INSERT', 11, '2024-03-30 09:05:00'),

(58, 'Customer', 30, 'DELETE', 20, '2024-03-31 11:25:00'),

(59, 'Loan', 19, 'UPDATE', 33, '2024-04-01 15:40:00'),

(60, 'Transaction', 32, 'DELETE', 14, '2024-04-02 13:50:00');

-- 10) Cards Data

INSERT INTO customeraccount.Cards

(Card\_ID, Account\_ID, Card\_Type, Card\_Number, Card\_Expiry\_Date, CVV, Issued\_Date, Card\_Status) VALUES

(1, 1, 'Debit', '1234567812345678', '2026-01-15', '123', '2020-01-15', 'Active'),

(2, 2, 'Credit', '2345678923456789', '2025-07-20', '456', '2019-07-20', 'Active'),

(3, 3, 'Debit', '3456789034567890', '2026-03-12', '789', '2021-03-12', 'Active'),

(4, 4, 'Debit', '4567890145678901', '2024-06-10', '321', '2022-06-10', 'Expired'),

(5, 5, 'Credit', '5678901256789012', '2025-11-25', '654', '2020-11-25', 'Active'),

(6, 6, 'Debit', '6789012367890123', '2024-09-18', '987', '2018-09-18', 'Expired'),

(7, 7, 'Credit', '7890123478901234', '2026-02-22', '111', '2023-02-22', 'Active'),

(8, 8, 'Debit', '8901234589012345', '2025-08-13', '222', '2021-08-13', 'Active'),

(9, 9, 'Credit', '9012345690123456', '2025-04-15', '333', '2020-04-15', 'Active'),

(10, 10, 'Debit', '0123456701234567', '2024-12-01', '444', '2019-12-01', 'Expired'),

(11, 11, 'Debit', '1234567811123456', '2026-05-20', '555', '2020-05-20', 'Active'),

(12, 12, 'Credit', '2345678922123457', '2025-03-05', '666', '2022-03-05', 'Active'),

(13, 13, 'Debit', '3456789033123458', '2026-06-30', '777', '2019-06-30', 'Active'),

(14, 14, 'Credit', '4567890144123459', '2025-01-12', '888', '2021-01-12', 'Active'),

(15, 15, 'Debit', '5678901255123460', '2024-10-10', '999', '2020-10-10', 'Expired'),

(16, 16, 'Debit', '6789012366123461', '2026-02-25', '123', '2022-02-25', 'Active'),

(17, 17, 'Credit', '7890123477123462', '2025-07-07', '234', '2019-07-07', 'Active'),

(18, 18, 'Debit', '8901234588123463', '2025-11-22', '345', '2021-11-22', 'Active'),

(19, 19, 'Credit', '9012345699123464', '2026-03-18', '456', '2020-03-18', 'Active'),

(20, 20, 'Debit', '0123456710123465', '2024-01-01', '567', '2023-01-01', 'Expired'),

(21, 25, 'Credit', '1234567822123466', '2026-12-30', '678', '2021-09-10', 'Active'),

(22, 30, 'Debit', '2345678933123467', '2025-10-20', '789', '2020-10-20', 'Active'),

(23, 35, 'Credit', '3456789044123468', '2026-07-10', '321', '2021-07-10', 'Active'),

(24, 40, 'Debit', '4567890155123469', '2024-08-31', '654', '2022-08-31', 'Expired'),

(25, 50, 'Debit', '5678901266123470', '2025-01-20', '987', '2023-01-20', 'Active'),

(26, 55, 'Credit', '6789012377123471', '2024-11-30', '111', '2020-11-30', 'Expired'),

(27, 60, 'Debit', '7890123488123472', '2025-05-28', '222', '2022-05-28', 'Active'),

(28, 66, 'Credit', '8901234599123473', '2026-12-30', '333', '2021-12-30', 'Active'),

(29, 70, 'Debit', '9012345610123474', '2024-09-30', '444', '2022-09-30', 'Expired'),

(30, 75, 'Debit', '0123456721123475', '2025-06-04', '555', '2021-06-04', 'Active'),

(31, 80, 'Credit', '1234567832123476', '2024-04-15', '666', '2023-04-15', 'Expired'),

(32, 52, 'Debit', '2345678943123477', '2026-09-22', '777', '2020-09-22', 'Active'),

(33, 22, 'Credit', '3456789054123478', '2025-07-22', '888', '2021-07-22', 'Active'),

(34, 18, 'Debit', '4567890165123479', '2026-05-15', '999', '2021-05-15', 'Active'),

(35, 9, 'Credit', '5678901276123480', '2025-08-24', '123', '2020-08-24', 'Active'),

(36, 12, 'Debit', '1234567845123481', '2026-02-20', '123', '2022-02-20', 'Active'),

(37, 15, 'Credit', '2345678956123482', '2025-05-10', '234', '2021-05-10', 'Active'),

(38, 25, 'Debit', '3456789067123483', '2024-12-15', '345', '2020-12-15', 'Expired'),

(39, 33, 'Credit', '4567890178123484', '2026-07-01', '456', '2022-07-01', 'Active'),

(40, 44, 'Debit', '5678901289123485', '2025-10-25', '567', '2021-10-25', 'Active'),

(41, 47, 'Credit', '6789012390123486', '2024-03-15', '678', '2020-03-15', 'Expired'),

(42, 53, 'Debit', '7890123401123487', '2026-01-30', '789', '2022-01-30', 'Active'),

(43, 62, 'Credit', '8901234512123488', '2025-08-20', '111', '2021-08-20', 'Active'),

(44, 67, 'Debit', '9012345623123489', '2024-07-05', '222', '2020-07-05', 'Expired'),

(45, 69, 'Credit', '0123456734123490', '2026-11-10', '333', '2022-11-10', 'Active'),

(46, 72, 'Debit', '1234567845123491', '2025-09-12', '444', '2021-09-12', 'Active'),

(47, 74, 'Credit', '2345678956123492', '2024-05-30', '555', '2020-05-30', 'Expired'),

(48, 78, 'Debit', '3456789067123493', '2026-03-18', '666', '2022-03-18', 'Active'),

(49, 79, 'Credit', '4567890178123494', '2025-07-22', '777', '2021-07-22', 'Active'),

(50, 80, 'Debit', '5678901289123495', '2026-06-15', '888', '2022-06-15', 'Active');

-- 11) Insurance Data

INSERT INTO customer\_offerings.Insurance

(Policy\_ID, Customer\_ID, Policy\_Type, Premium\_Amount, Coverage\_Amount, Start\_Date, End\_Date, Policy\_Status)

VALUES

(1, 2, 'Health', 1500.00, 200000.00, '2021-01-15', '2026-01-15', 'Active'),

(2, 5, 'Life', 2500.00, 500000.00, '2020-03-10', '2030-03-10', 'Active'),

(3, 10, 'Travel', 800.00, 50000.00, '2023-02-20', '2023-08-20', 'Lapsed'),

(4, 12, 'Vehicle', 1200.00, 300000.00, '2021-07-01', '2026-07-01', 'Active'),

(5, 15, 'Health', 1800.00, 250000.00, '2020-05-10', '2025-05-10', 'Active'),

(6, 20, 'Life', 2000.00, 400000.00, '2019-09-15', '2029-09-15', 'Active'),

(7, 25, 'Travel', 600.00, 75000.00, '2023-04-10', '2024-04-10', 'Active'),

(8, 30, 'Vehicle', 1400.00, 250000.00, '2020-11-05', '2025-11-05', 'Active'),

(9, 33, 'Life', 2200.00, 550000.00, '2022-01-15', '2032-01-15', 'Active'),

(10, 35, 'Health', 1600.00, 200000.00, '2021-12-10', '2026-12-10', 'Active'),

(11, 40, 'Travel', 700.00, 100000.00, '2023-06-01', '2024-06-01', 'Active'),

(12, 42, 'Vehicle', 1300.00, 200000.00, '2020-10-05', '2025-10-05', 'Cancelled'),

(13, 45, 'Life', 3000.00, 600000.00, '2018-07-15', '2028-07-15', 'Active'),

(14, 50, 'Health', 1700.00, 250000.00, '2022-03-20', '2027-03-20', 'Active'),

(15, 53, 'Travel', 900.00, 150000.00, '2023-05-12', '2024-05-12', 'Cancelled'),

(16, 55, 'Vehicle', 1100.00, 280000.00, '2021-04-01', '2026-04-01', 'Active'),

(17, 60, 'Life', 2800.00, 700000.00, '2020-12-20', '2030-12-20', 'Active'),

(18, 63, 'Health', 1500.00, 200000.00, '2023-03-15', '2028-03-15', 'Active'),

(19, 65, 'Travel', 750.00, 125000.00, '2022-11-10', '2023-11-10', 'Lapsed'),

(20, 67, 'Vehicle', 1450.00, 220000.00, '2021-08-20', '2026-08-20', 'Active'),

(21, 70, 'Life', 2400.00, 450000.00, '2019-05-15', '2029-05-15', 'Active'),

(22, 72, 'Health', 1900.00, 300000.00, '2022-06-30', '2027-06-30', 'Active'),

(23, 75, 'Travel', 850.00, 150000.00, '2023-01-05', '2024-01-05', 'Active'),

(24, 78, 'Vehicle', 1350.00, 250000.00, '2021-11-15', '2026-11-15', 'Active'),

(25, 80, 'Life', 2600.00, 500000.00, '2020-02-25', '2030-02-25', 'Active'),

(26, 3, 'Health', 1800.00, 230000.00, '2022-08-18', '2027-08-18', 'Active'),

(27, 6, 'Travel', 950.00, 120000.00, '2023-04-25', '2024-04-25', 'Active'),

(28, 9, 'Vehicle', 1500.00, 350000.00, '2021-09-10', '2026-09-10', 'Active'),

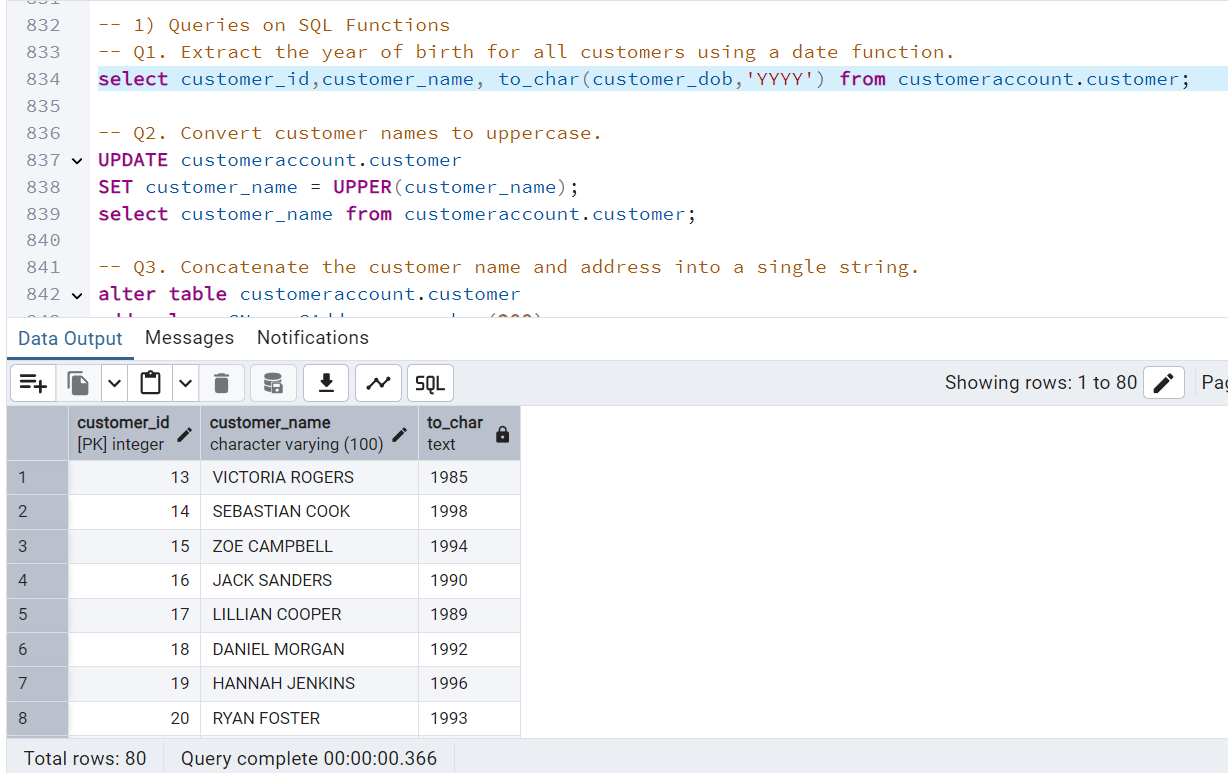
(29, 18, 'Life', 2100.00, 600000.00, '2020-07-15', '2030-07-15', 'Active'),

(30, 22, 'Health', 1750.00, 250000.00, '2023-02-20', '2028-02-20', 'Active');

-- 1) Queries on SQL Functions

-- Q1. Extract the year of birth for all customers using a date function.

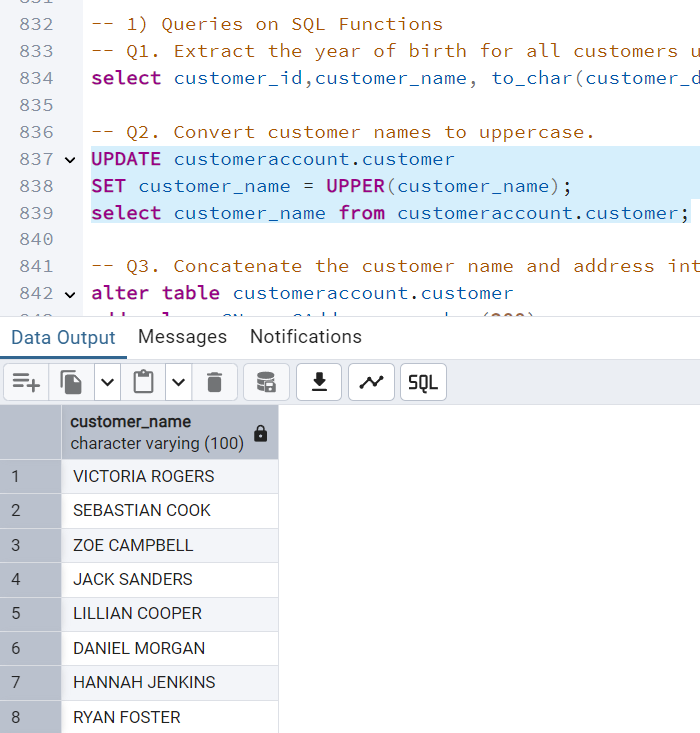
select customer\_id,customer\_name, to\_char(customer\_dob,'YYYY') from customeraccount.customer;



-- Q2. Convert customer names to uppercase.

UPDATE customeraccount.customer

SET customer\_name = UPPER(customer\_name);

select customer\_name from customeraccount.customer;

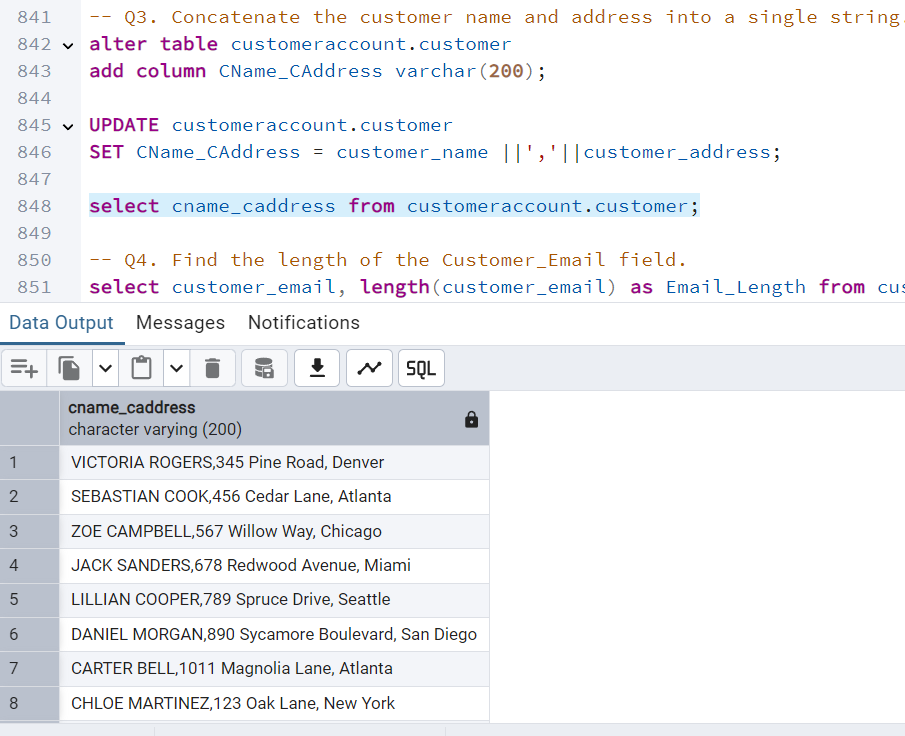
-- Q3. Concatenate the customer name and address into a single string.

alter table customeraccount.customer

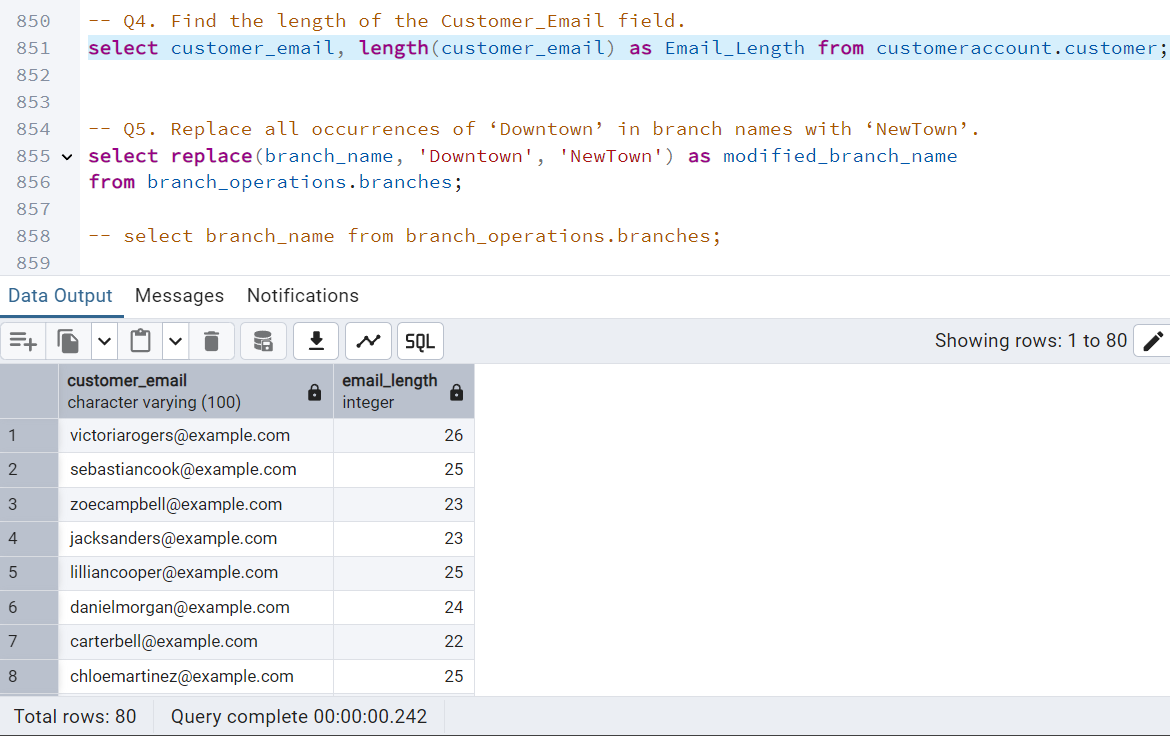
add column CName\_CAddress varchar(200);

UPDATE customeraccount.customer

SET CName\_CAddress = customer\_name ||','||customer\_address;

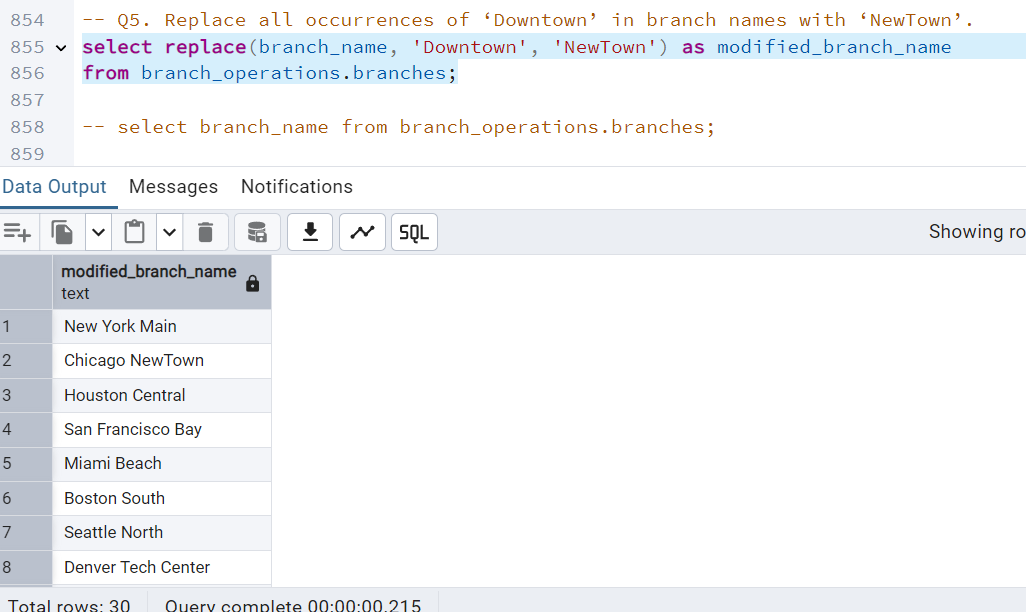
select cname\_caddress from customeraccount.customer;

-- Q4. Find the length of the Customer\_Email field.

select customer\_email, length(customer\_email) as Email\_Length from customeraccount.customer;

-- Q5. Replace all occurrences of ‘Downtown’ in branch names with ‘NewTown’.

select replace(branch\_name, 'Downtown', 'NewTown') as modified\_branch\_name

from branch\_operations.branches;

-- select branch\_name from branch\_operations.branches;

-- Q6. Trim extra spaces from Manager\_Name.

select trim(manager\_name) as Trim\_Manager\_Name from branch\_operations.branches;

-- Q7. Calculate the difference in days between the start and end dates of all loans.

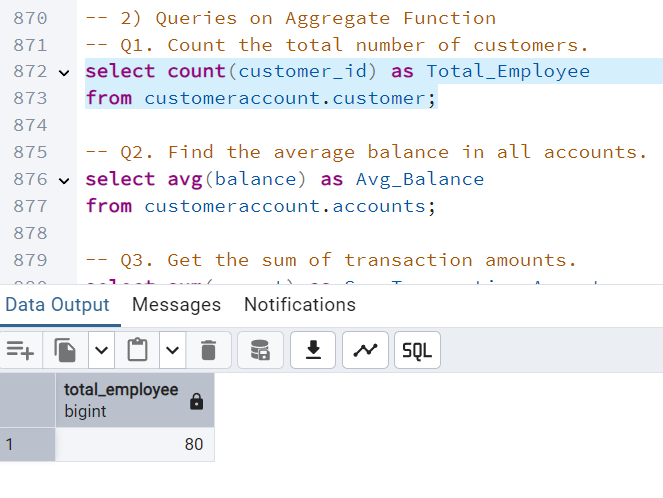
select loan\_id, start\_date, end\_date, (end\_date - start\_date) as Days\_Difference

from customer\_offerings.loans;

-- 2) Queries on Aggregate Function

-- Q1. Count the total number of customers.

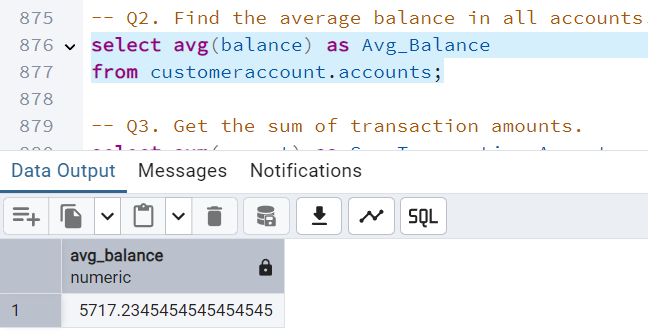
select count(customer\_id) as Total\_Employee

from customeraccount.customer;

-- Q2. Find the average balance in all accounts.

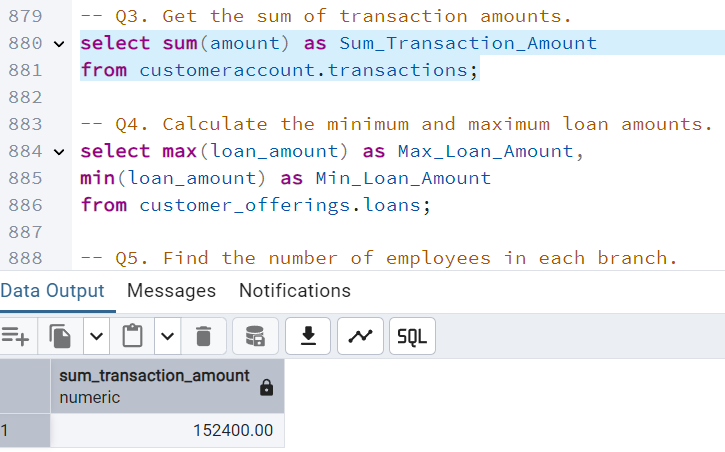
select avg(balance) as Avg\_Balance

from customeraccount.accounts;



-- Q3. Get the sum of transaction amounts.

select sum(amount) as Sum\_Transaction\_Amount

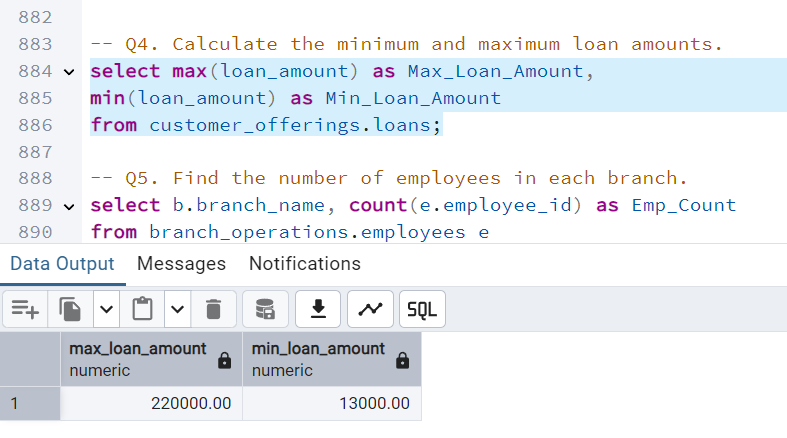
from customeraccount.transactions;

-- Q4. Calculate the minimum and maximum loan amounts.

select max(loan\_amount) as Max\_Loan\_Amount,

min(loan\_amount) as Min\_Loan\_Amount

from customer\_offerings.loans;

-- Q5. Find the number of employees in each branch.

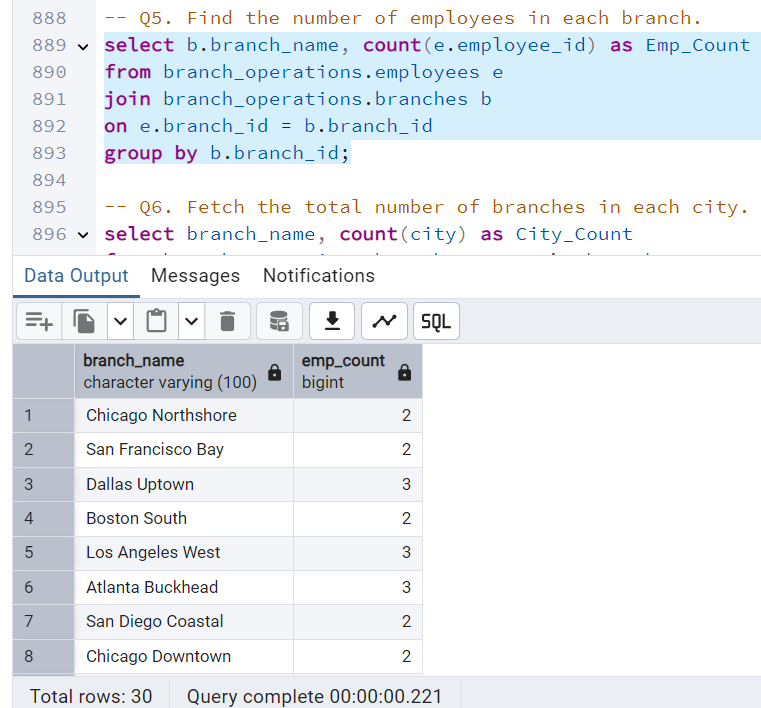
select b.branch\_name, count(e.employee\_id) as Emp\_Count

from branch\_operations.employees e

join branch\_operations.branches b

on e.branch\_id = b.branch\_id

group by b.branch\_id;



-- Q6. Fetch the total number of branches in each city.

select branch\_name, count(city) as City\_Count

from branch\_operations.branches group by branch\_name;

-- Q7. Calculate the average interest rate for loans.

select avg(interest\_rate) as Avg\_Interest\_Rate

from customer\_offerings.loans;

-- Q8. Count the number of insurance policies per customer.

select policy\_type, count(policy\_type) as Policy\_Count

from customer\_offerings.insurance group by policy\_type;

-- 3) Joins

-- Q1. List all customers along with their accounts using an Inner Join.

select c.customer\_id ,c.customer\_name, a.account\_id, a.account\_type, a.balance, a.branch\_name

from customeraccount.customer c

inner join customeraccount.accounts a

on c.customer\_id = a.customer\_id

order by c.customer\_id asc;

-- Q2. Fetch all employees and their respective branch details using an Inner Join.

select e.employee\_id, e.employee\_name, b.branch\_id, b.branch\_name

from branch\_operations.employees e

inner join branch\_operations.branches b

on e.branch\_id = b.branch\_id;

-- Q3. Get a list of all branches and the products they offer (Left Join).

select p.branch\_id,b.branch\_name, p.product\_id, p.product\_name

from branch\_operations.branches b

left join customer\_offerings.products p

on p.branch\_id = b.branch\_id;

-- Q4. List all accounts with their transactions using an Outer Join.

select a.account\_id, a.account\_type, t.transaction\_id, t.transaction\_type

from customeraccount.accounts a

full outer join customeraccount.transactions t

on a.account\_id = t.account\_id;

-- Q5. Identify accounts that do not have any associated transactions (Full Outer Join).

select a.account\_id, a.customer\_id, t.transaction\_id

from customeraccount.accounts a

full outer join customeraccount.transactions t

on a.account\_id = t.account\_id

where transaction\_id is null;

-- Q6. Fetch all customers who have taken loans and their loan details (Left Join).

select a.account\_id, a.customer\_id, l.loan\_id, l.loan\_type

from customeraccount.accounts a

left join customer\_offerings.loans l

on a.customer\_id = l.customer\_id where loan\_id is not null or l.loan\_type is not null;

-- Q7. List employees earning more than the average salary of their branch (Self Join).

select se.employee\_id, e.employee\_name

from branch\_operations.employees e

join branch\_operations.employees se

on e.employee\_id = se.employee\_id

where e.salary > (select avg(salary) from branch\_operations.employees);

-- Q8. Fetch the branches and employees who were hired after 2020 (Inner Join).

select e.employee\_id, e.employee\_name, b.branch\_name, e.hire\_date

from branch\_operations.employees e

inner join branch\_operations.branches b

on e.branch\_id = b.branch\_id

where to\_char(e.hire\_date,'YYYY') >'2020' order by e.employee\_id;

-- Subqueries

-- Q1. Find the customer with the highest account balance.

select account\_id, customer\_id, balance

from customeraccount.accounts

where balance = (select max(balance) from customeraccount.accounts);

-- Q2. Fetch branches with an average salary greater than 70,000.

select e.branch\_id, b.branch\_name, e.salary

from branch\_operations.employees e

join branch\_operations.branches b

on e.branch\_id = b.branch\_id

where (select avg(salary) from branch\_operations.employees)>50000;

-- Chat GPT

SELECT b.branch\_id, b.branch\_name

FROM branch\_operations.branches b

WHERE b.branch\_id IN (

SELECT e.branch\_id

FROM branch\_operations.employees e

GROUP BY e.branch\_id

HAVING AVG(e.salary) > 70000);

-- Q3. List accounts with a balance above the average for all accounts.

select a.account\_id,a.customer\_id, a.balance from customeraccount.accounts a

where balance > (select avg(balance) from customeraccount.accounts);

-- Q4. Identify employees earning more than the maximum salary in another branch.

select e.employee\_id, e.salary, b.branch\_name

from branch\_operations.employees e

join branch\_operations.branches b

on b.branch\_id = e.branch\_id

where e.salary > (select max(e2.salary)

from branch\_operations.employees e2

join branch\_operations.branches b2

on b2.branch\_id = e2.branch\_id

where b2.branch\_id != b.branch\_id

)

order by b.branch\_name, e.salary desc;

-- Q5. Fetch the names of branches offering the highest number of products.

-- Chat GPT

SELECT b.branch\_name, COUNT(p.product\_id) AS product\_count

FROM customer\_offerings.products p

JOIN customer\_offerings.branches b

ON b.branch\_id = p.branch\_id

GROUP BY b.branch\_id, b.branch\_name

HAVING COUNT(p.product\_id) = (

SELECT MAX(product\_count)

FROM (

SELECT COUNT(product\_id) AS product\_count

FROM customer\_offerings.products

GROUP BY branch\_id

) branch\_product\_counts

)

ORDER BY b.branch\_name;

-- Q6. Find the branch with the most employees.

select b.branch\_id, b.branch\_name

from branch\_operations.employees e

join branch\_operations.branches b

on e.branch\_id = b.branch\_id

where

-- Q7. Identify customers with the most transactions in their accounts.

select c.customer\_id, c.customer\_name, count(transaction\_id) as No\_Of\_Transaction

from customeraccount.customer c

join customeraccount.accounts a

on c.customer\_id = a.customer\_id

join customeraccount.transactions t

on a.account\_id = t.account\_id

-- where (select count(transaction\_id)>2 from customeraccount.transactions)

group by c.customer\_id;

-- Chat GPT

WITH CustomerTransactionCounts AS (

SELECT

c.customer\_id,

c.customer\_name,

COUNT(t.transaction\_id) AS transaction\_count

FROM customeraccount.customer c

JOIN customeraccount.accounts a

ON c.customer\_id = a.customer\_id

JOIN customeraccount.transactions t

ON a.account\_id = t.account\_id

GROUP BY c.customer\_id, c.customer\_name

)

SELECT customer\_id, customer\_name, transaction\_count

FROM CustomerTransactionCounts

WHERE transaction\_count = (

SELECT MAX(transaction\_count)

FROM CustomerTransactionCounts

);

-- 5. OVER Clause

-- Q1. Assign row numbers to all customers in the Customer table.

select row\_number() over (order by customer\_id) as RowNumber, customer\_id, customer\_name

from customeraccount.customer;

-- Q2. Rank employees based on their salary within each branch.

select employee\_id, employee\_name, salary,

rank() over(partition by branch\_id order by salary asc) as Rank\_Emp

from branch\_operations.employees;

-- Q3. Calculate the cumulative total balance for all accounts, ordered by Opening\_Date.

select account\_id, customer\_id, sum(balance) over(order by Opening\_Date) as Comulative\_Balance

from customeraccount.accounts;

-- Q4. Assign a row number to each transaction for a given account.

select row\_number() over (order by transaction\_id) as RowNumber, transaction\_id, account\_id, transaction\_date, transaction\_type

from customeraccount.transactions;

-- Q5. Find the running total of transaction amounts for each account.

select account\_id, transaction\_id, transaction\_date, amount,

sum(amount) over(partition by account\_id order by transaction\_id) as Running\_Amount

from customeraccount.transactions;

-- Q6. Calculate the average salary within each branch using OVER.

select branch\_id, avg(salary) over(partition by branch\_id order by branch\_id) as Avg\_Sal

from branch\_operations.employees;

-- 6) RANK(), DENSE\_RANK(), ROW\_NUMBER()

-- 1. Assign row numbers to all branches based on their Established\_Date.

select row\_number() over(order by established\_date) as Established\_Sequence,

branch\_id, branch\_name

from branch\_operations.branches;

-- 2. Rank customers based on their total balance using RANK().

select rank() over(order by balance) as Bal\_Rank, customer\_id, account\_id, balance

from customeraccount.accounts;

-- 3. Assign dense ranks to loans based on their Interest\_Rate for each branch.

select dense\_rank() over(order by interest\_rate) as Interest\_Rate,loan\_id, customer\_id

from customer\_offerings.loans;

-- 4. Rank employees by their salary within their branch using DENSE\_RANK().

select dense\_rank() over(order by salary asc) as Most\_Paid\_Employee, employee\_id, employee\_name, salary

from branch\_operations.employees;

-- 5. Assign row numbers to documents uploaded by each customer.

select row\_number() over(partition by document\_type order by document\_id) as document\_number, document\_id, document\_type

from customeraccount.documents;

-- 6. Rank transactions based on their Transaction\_Amount using RANK().

select rank() over(order by amount asc) as Transaction\_Rank, transaction\_id, account\_id, amount

from customeraccount.transactions;

-- 7. Assign dense ranks to accounts based on their balance within each branch.

select dense\_rank() over(order by balance) as Rank\_Balance, customer\_id, account\_id, balance

from customeraccount.accounts;

-- 8. Rank customers by the number of uploaded documents using DENSE\_RANK().

-- Chat GPT

SELECT

DENSE\_RANK() OVER (ORDER BY Document\_Count DESC) AS No\_Doc\_Uploaded,

Customer\_ID,

Document\_Count

FROM(

SELECT

Associated\_ID AS Customer\_ID,

COUNT(Document\_ID) AS Document\_Count

FROM customeraccount.Documents

WHERE Associated\_With = 'Customer'

GROUP BY Associated\_ID

)AS DocumentSummary;

7) Aggregate Functions with PARTITION BY

-- 1. Calculate the total transaction amount for each account using PARTITION BY.

select sum(amount) over(partition by account\_id ) as Total\_Transaction\_Amount, transaction\_id, account\_id, amount, transaction\_type

from customeraccount.transactions;

-- 2. Find the average loan amount partitioned by Loan\_Type.

select Avg(loan\_amount) over(partition by loan\_type) as Avg\_Loan\_Taken, customer\_id

from customer\_offerings.loans order by customer\_id;

-- 3. Fetch the total salary for each branch using PARTITION BY.

select sum(salary) over(partition by branch\_id) as Total\_Branch\_Salary, branch\_id, employee\_id, employee\_name

from branch\_operations.employees;

-- 4. Calculate the total balance of accounts grouped by Branch\_Name.

select sum(balance) over(partition by branch\_name) as Total\_Balance, account\_id, branch\_name, account\_type

from customeraccount.accounts order by account\_id;

-- 5. Determine the maximum transaction amount for each Transaction\_Type.

select max(amount) over(partition by transaction\_type) as MAx\_Transaction\_Amount, account\_id, transaction\_id

from customeraccount.transactions order by account\_id;

-- 6. Find the total premium collected, partitioned by Insurance\_Type.

select sum(premium\_amount) over(partition by policy\_type) as Total\_Premium, policy\_id, customer\_id

from customer\_offerings.insurance order by policy\_id;

-- 8) Data Dictionary and System Views

-- 1. List all tables in the database.

select table\_schema, table\_name

from information\_schema.tables

where table\_type = 'BASE TABLE' and table\_schema not in ('information\_schema', 'pg\_catalog')

order by table\_schema, table\_name;

-- 2. Fetch the schema details for the Accounts table.

SELECT

table\_schema,

table\_name,

column\_name,

data\_type,

is\_nullable,

character\_maximum\_length

FROM

information\_schema.columns

WHERE

table\_name = 'accounts'

AND table\_schema = 'customeraccount';

-- 3. Find the column names and their data types for the Loans table.

SELECT

column\_name,

data\_type

FROM

information\_schema.columns

WHERE

table\_name = 'loans'

AND table\_schema = 'customer\_offerings';

-- 4. Retrieve all foreign keys defined in the database.

SELECT

tc.constraint\_name AS foreign\_key\_name,

tc.table\_schema,

tc.table\_name,

kcu.column\_name,

ccu.table\_schema AS referenced\_table\_schema,

ccu.table\_name AS referenced\_table\_name,

ccu.column\_name AS referenced\_column\_name

FROM

information\_schema.table\_constraints AS tc

JOIN

information\_schema.key\_column\_usage AS kcu

ON tc.constraint\_name = kcu.constraint\_name

AND tc.table\_schema = kcu.table\_schema

JOIN

information\_schema.constraint\_column\_usage AS ccu

ON ccu.constraint\_name = tc.constraint\_name

AND ccu.table\_schema = tc.table\_schema

WHERE

tc.constraint\_type = 'FOREIGN KEY';

-- 5. Fetch the indexes applied to the Transactions table.

SELECT

indexname AS index\_name,

indexdef AS index\_definition

FROM

pg\_indexes

WHERE

tablename = 'transactions'

AND schemaname = 'customeraccount';

-- 6. List all schemas in the database.

SELECT

schema\_name

FROM

information\_schema.schemata

ORDER BY

schema\_name;

-- 9) Views

-- 1. Create a view to list all customers and their account balances.

CREATE VIEW View\_Cust\_List AS

SELECT

c.Customer\_ID,

c.Customer\_Name,

c.Customer\_Address,

c.Phone\_Number,

c.Customer\_Email,

a.Account\_ID,

a.Account\_Type,

a.Balance,

a.Branch\_Name

FROM

customeraccount.Customer c

JOIN

customeraccount.Accounts a

ON c.Customer\_ID = a.Customer\_ID;

-- 2. Create a view for transactions above a specified amount. CREATE VIEW EmployeeBranchDetails AS

SELECT

e.Employee\_ID,

e.Employee\_Name,

e.Positions,

e.Salary,

b.Branch\_Name,

b.City,

b.Manager\_Name

FROM

branch\_operations.Employees e

JOIN

branch\_operations.Branches b

ON e.Branch\_ID = b.Branch\_ID;

-- 3. Fetch data from a view showing employees and their branch details.

CREATE VIEW ActiveCustomerLoans AS

SELECT

l.Loan\_ID,

l.Customer\_ID,

l.Loan\_Type,

l.Loan\_Amount,

l.Interest\_Rate,

l.Start\_Date,

l.End\_Date

FROM

customer\_offerings.Loans l

JOIN

customeraccount.Customer c

ON l.Customer\_ID = c.Customer\_ID;

-- 4. Create a view to display loan details for active customers only.

CREATE OR REPLACE VIEW PolicyDetails AS

SELECT

Policy\_ID,

Customer\_ID,

Policy\_Type,

Premium\_Amount,

Coverage\_Amount,

Start\_Date,

End\_Date,

Policy\_Status

FROM

customer\_offerings.Insurance;

SELECT \* FROM customeraccount.customer;

-- 6. Create a view for accounts with a balance below 1,000.

CREATE VIEW LowBalanceAccounts AS

SELECT

Account\_ID,

Customer\_ID,

Balance,

Account\_Type,

Branch\_Name

FROM

customeraccount.Accounts

WHERE

Balance < 1000;

-- 10) Indexing

-- 1. Create an index on Customer\_Email in the Customers table.

CREATE INDEX idx\_Customer\_Email ON customeraccount.Customer(Customer\_Email);

-- 2. Create a unique index on Card\_Number in the Cards table.

CREATE UNIQUE INDEX idx\_Card\_Number ON customeraccount.Cards(Card\_Number);

-- 3. Create a clustered index on Transaction\_Date in the Transactions table.

CREATE INDEX idx\_Transaction\_Date ON customeraccount.Transactions(Transaction\_Date);

-- 4. Create a non-clustered index on Account\_Type in the Accounts table.

CREATE INDEX idx\_Account\_Type ON customeraccount.Accounts(Account\_Type);

-- 5. Create a composite index on Branch\_Name and City in the Branches table.

CREATE INDEX idx\_Branch\_Name\_City ON branch\_operations.Branches(Branch\_Name, City);

-- 6. Check the existing indexes on the Loans table.

SELECT

indexname AS Index\_Name,

indexdef AS Index\_Definition

FROM

pg\_indexes

WHERE

tablename = 'loans'

AND schemaname = 'customer\_offerings';