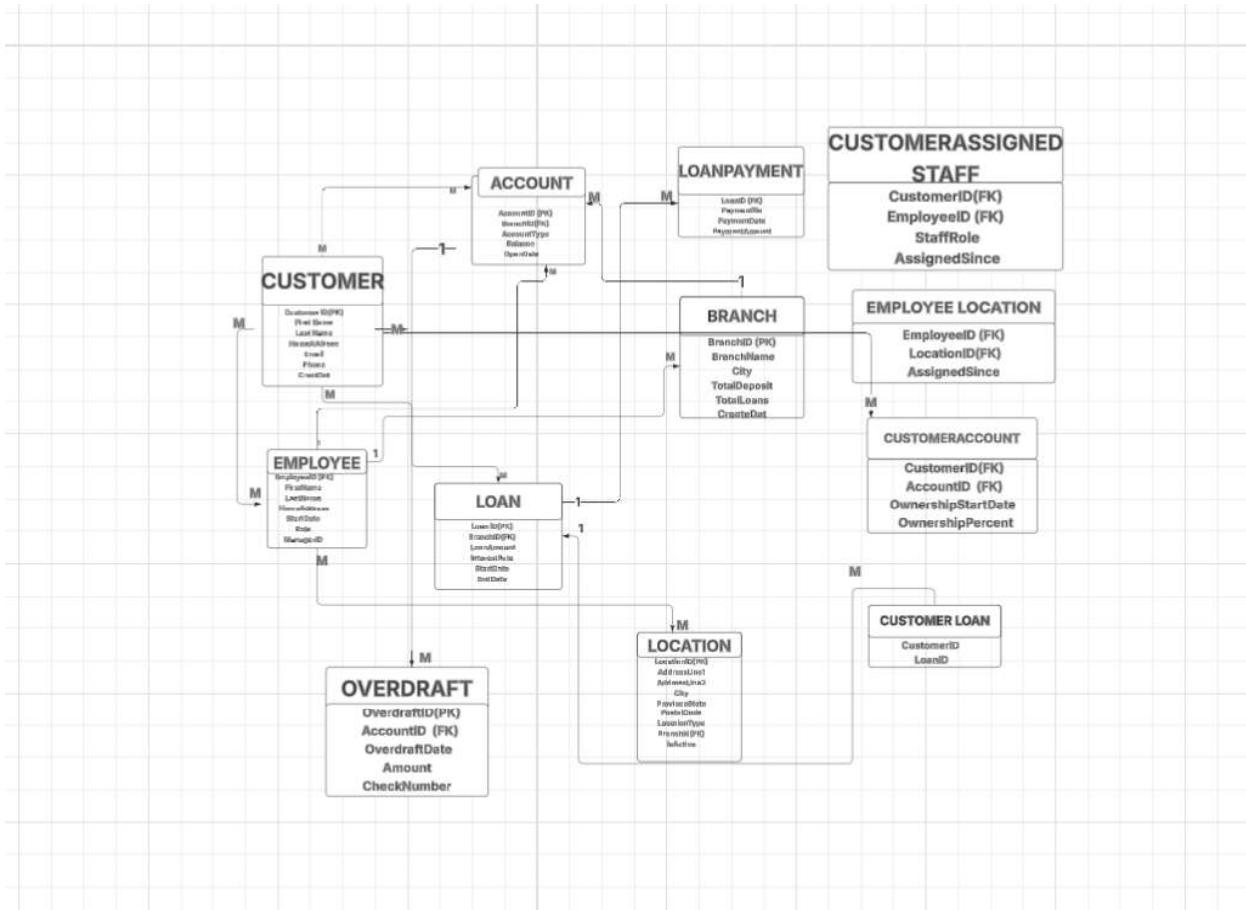
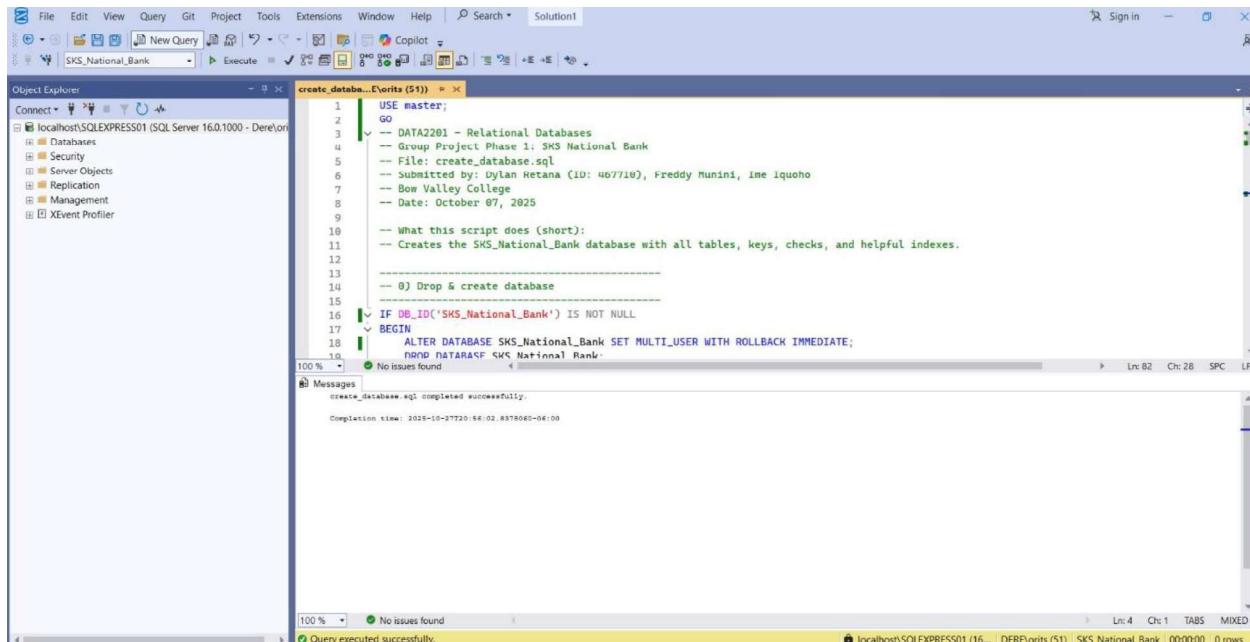


## ERD Diagram.



## create\_database\_database



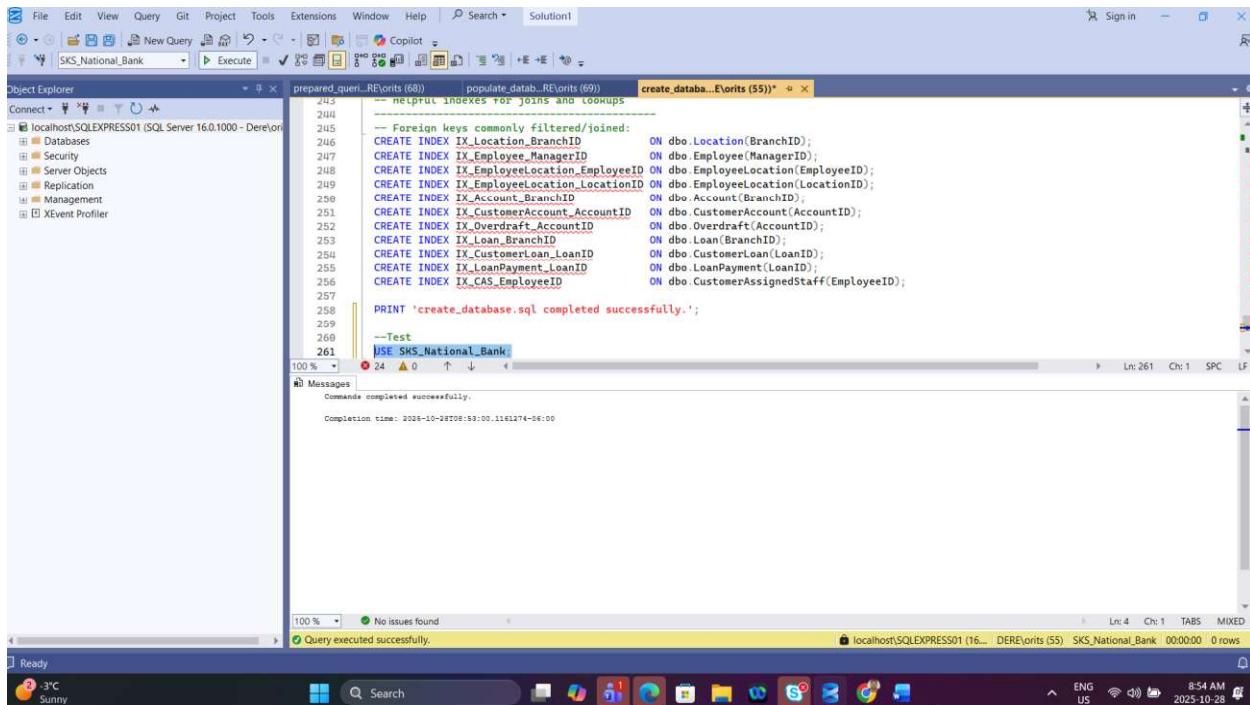
```
USE master;
GO
-- DATA2201 - Relational Databases
-- Group Project Phase 1: SKS National Bank
-- File: create_database.sql
-- Submitted by: Dylan Retana (ID: 407710), Freddy Munini, Ime Iquoho
-- Bow Valley College
-- Date: October 07, 2025

-- What this script does (short):
-- Creates the SKS_National_Bank database with all tables, keys, checks, and helpful indexes.

-- 0) Drop & create database
IF DB_ID('SKS_National_Bank') IS NOT NULL
BEGIN
    ALTER DATABASE SKS_National_Bank SET MULTI_USER WITH ROLLBACK IMMEDIATE;
    DROP DATABASE SKS_National_Bank;
END
```

create\_database.sql completed successfully.  
Completion time: 2025-10-27T20:54:02.837000-04:00

### 1) Confirm database exist



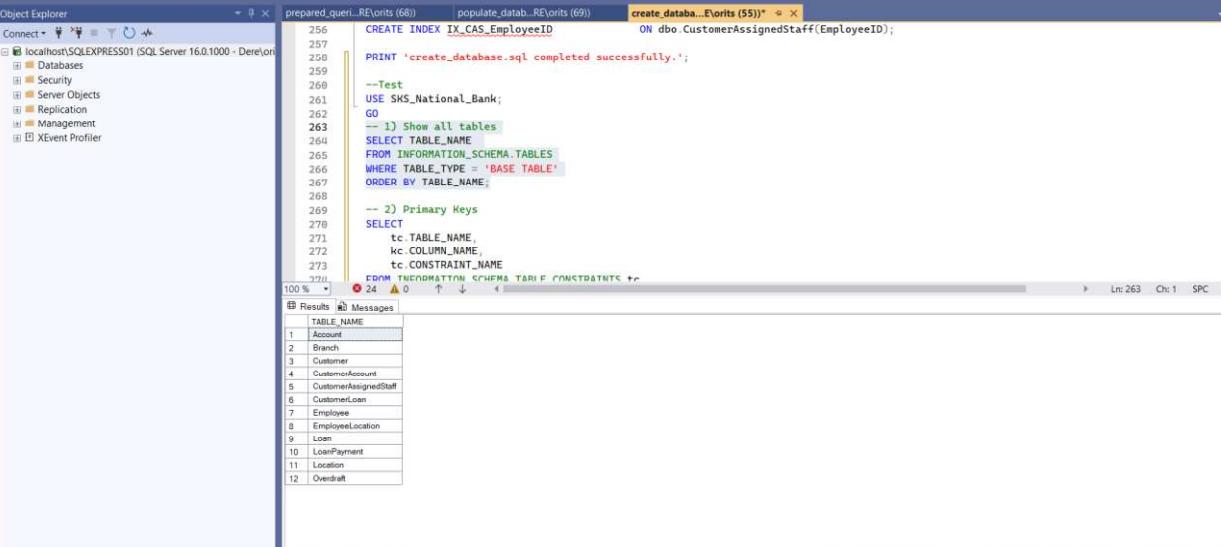
```
-- helpful indexes for joins and lookups
-- Foreign keys commonly filtered/joined:
CREATE INDEX IX_Location_BranchID ON dbo.Location(BranchID);
CREATE INDEX IX_Employee_ManagerID ON dbo.EmployeeManagerID;
CREATE INDEX IX_EmployeeLocation_EmployeeID ON dbo.EmployeeLocation(EmployeeID);
CREATE INDEX IX_EmployeeLocation_LocationID ON dbo.EmployeeLocation(LocationID);
CREATE INDEX IX_Account_BranchID ON dbo.Account(BranchID);
CREATE INDEX IX_CustomerAccount_AccountID ON dbo.CustomerAccount(AccountID);
CREATE INDEX IX_Overdraft_AccountID ON dbo.Overdraft(AccountID);
CREATE INDEX IX_Loan_BranchID ON dbo.Loan(BranchID);
CREATE INDEX IX_CustomerLoan_LoanID ON dbo.CustomerLoan(LoanID);
CREATE INDEX IX_LoanPayment_LoanID ON dbo.LoanPayment(LoanID);
CREATE INDEX IX_CAS_EmployeeID ON dbo.CustomerAssignedStaff(EmployeeID);

PRINT 'create_database.sql completed successfully.';

--Test
USE SKS_National_Bank;
```

Commands completed successfully.  
Completion time: 2024-10-28T08:59:00.1161274-05:00

## 2) Show\_Table



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left shows a connection to 'localhost\SQLEXPRESS01'. The 'Databases' node is expanded, showing 'SKS\_National\_Bank' as the selected database. The 'Scripting' toolbar is visible at the top. The main window contains a query editor with the following script:

```
CREATE INDEX IX_CAS_EmployeeID
ON dbo.CustomerAssignedStaff(EmployeeID);

PRINT 'create_database.sql completed successfully.';

--Test
USE SKS_National_Bank;
GO
-- 1) Show all tables
SELECT TABLE_NAME
FROM INFORMATION_SCHEMA.TABLES
WHERE TABLE_TYPE = 'BASE TABLE'
ORDER BY TABLE_NAME;

-- 2) Primary Keys
SELECT
    tc.TABLE_NAME,
    kc.COLUMN_NAME,
    tc.CONSTRAINT_NAME
FROM INFORMATION_SCHEMA.CONSTRAINTS tc
    JOIN INFORMATION_SCHEMA.KEY_COLUMN_USAGE kc
        ON kc.CONSTRAINT_NAME = tc.CONSTRAINT_NAME
        AND tc.TABLE_NAME = kc.TABLE_NAME;
```

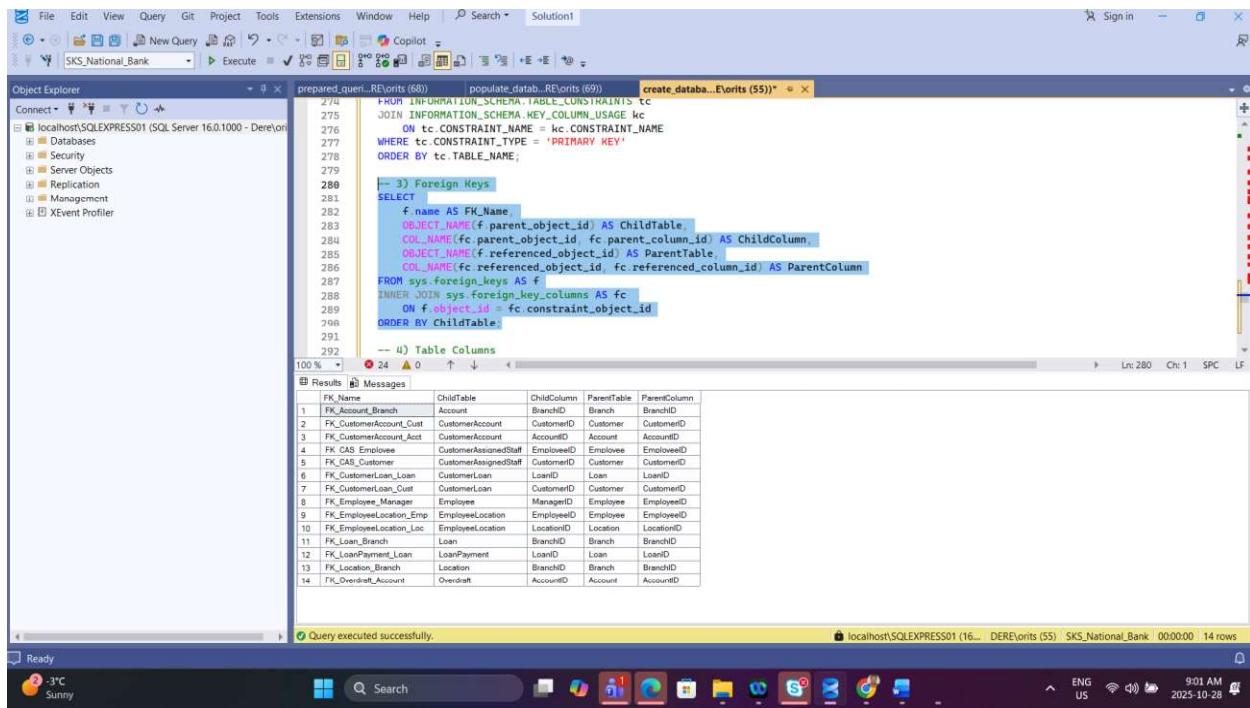
The results grid shows the following table data:

TABLE_NAME
1 Account
2 Branch
3 Customer
4 CustomerAccount
5 CustomerAssignedStaff
6 CustomerLoan
7 Employee
8 EmployeeLocation
9 Loan
10 LoanPayment
11 Location
12 Overdraft

At the bottom, a status bar indicates 'Query executed successfully.' and shows the connection details: 'localhost\SQLEXPRESS01 (16... DFRPrint (55) SKS\_National\_Bank 00:00:00 17 rows'.

### 3) Primary\_key

#### 4) Foreign\_key



```

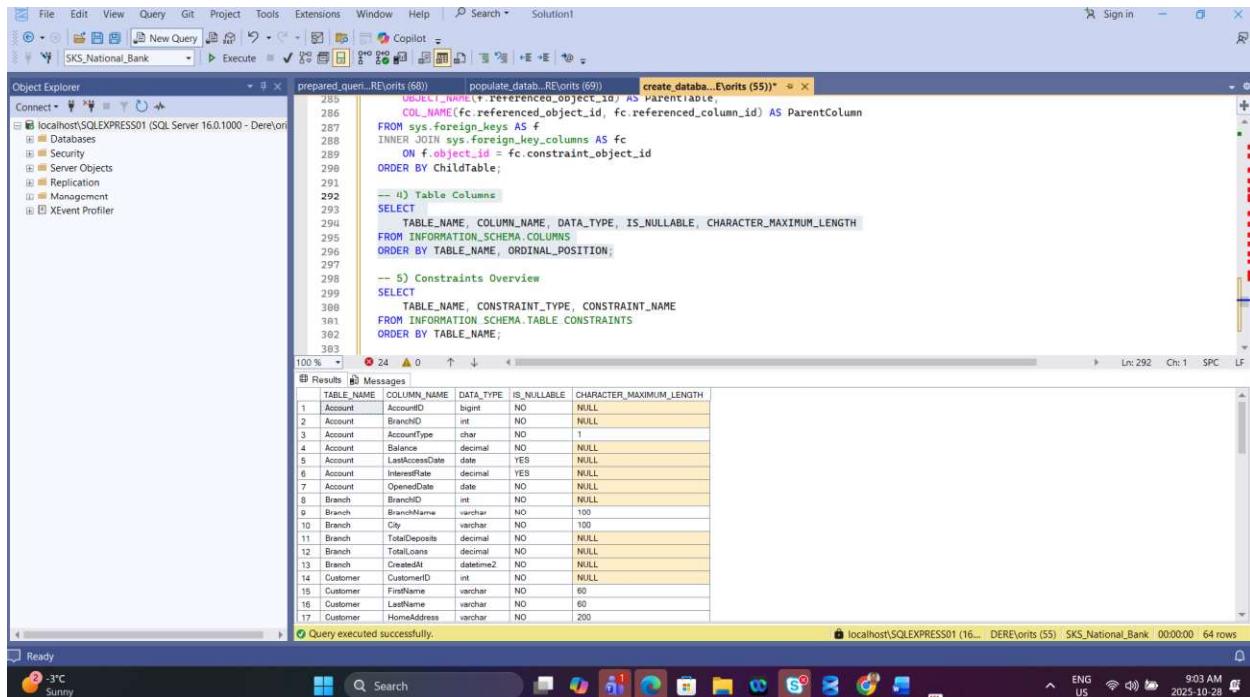
-- 3) Foreign Keys
SELECT
    + name AS FK_Name,
    OBJECT_NAME(f.parent_object_id) AS ChildTable,
    COL_NAME(fc.parent_object_id, fc.parent_column_id) AS ChildColumn,
    OBJECT_NAME(f.referenced_object_id) AS ParentTable,
    COL_NAME(fc.referenced_object_id, fc.referenced_column_id) AS ParentColumn
FROM sys.foreign_keys AS f
INNER JOIN sys.foreign_key_columns AS fc
    ON f.object_id = fc.constraint_object_id
ORDER BY ChildTable
    
```

-- 4) Table Columns

FK_Name	ChildTable	ChildColumn	ParentTable	ParentColumn
FK_Account_Branch	Account	BranchID	Branch	BranchID
FK_CustomerAccount_Cust	CustomerAccount	CustomerID	Customer	CustomerID
FK_CustomerAccount_Account	CustomerAccount	AccountID	Account	AccountID
FK_Employee_Employee	CustomerAssignedStaff	EmployeeID	Employee	EmployeeID
FK_CAB_Customer	CustomerAssignedStaff	CustomerID	Customer	CustomerID
FK_CustomerLoan_Loan	CustomerLoan	LoanID	Loan	LoanID
FK_CustomerLoan_Cust	CustomerLoan	CustomerID	Customer	CustomerID
FK_Employee_Manager	Employee	ManagerID	Employee	EmployeeID
FK_EmployeeLocation_Emp	EmployeeLocation	EmployeeID	Employee	EmployeeID
FK_EmployeeLocation_Loc	EmployeeLocation	LocationID	Location	LocationID
FK_Loan_Branch	Loan	BranchID	Branch	BranchID
FK_LoanPayment_Loan	LoanPayment	LoanID	Loan	LoanID
FK_Location_Branch	Location	BranchID	Branch	BranchID
FK_Overdraft_Account	Overdraft	AccountID	Account	AccountID

Query executed successfully.

#### 5) Table\_columns



```

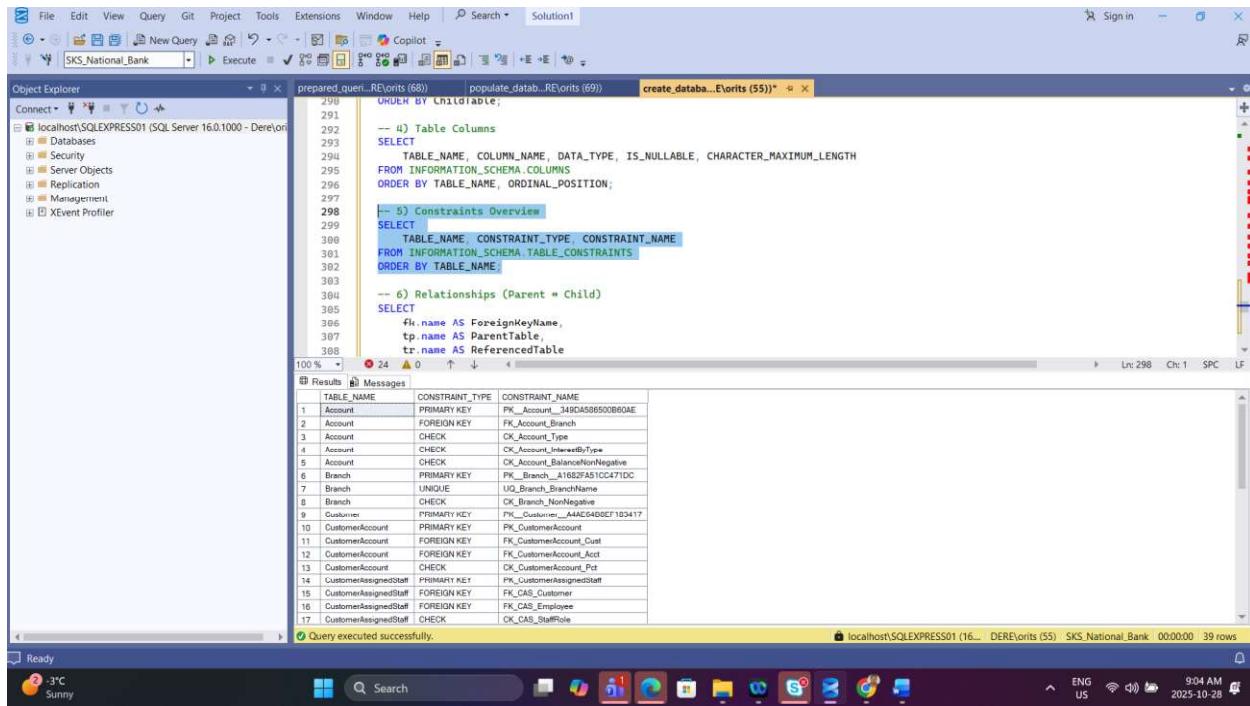
-- 4) Table Columns
SELECT
    TABLE_NAME, COLUMN_NAME, DATA_TYPE, IS_NULLABLE, CHARACTER_MAXIMUM_LENGTH
FROM INFORMATION_SCHEMA.COLUMNS
ORDER BY TABLE_NAME, ORDINAL_POSITION;
    
```

-- 5) Constraints Overview

TABLE_NAME	COLUMN_NAME	DATA_TYPE	IS_NULLABLE	CHARACTER_MAXIMUM_LENGTH
Account	AccountID	bigint	NO	NULL
Account	BranchID	int	NO	NULL
Account	AccountType	char	NO	1
Account	Balance	decimal	NO	NULL
Account	LastAccessed	date	YES	NULL
Account	InterestRate	decimal	YES	NULL
Account	OpenedDate	date	NO	NULL
Branch	BranchID	int	NO	NULL
Branch	BranchName	varchar	NO	100
Branch	City	varchar	NO	100
Branch	TotalDeposits	decimal	NO	NULL
Branch	TotalLoans	decimal	NO	NULL
Branch	CreatedID	int	NO	NULL
Customer	CustomerID	int	NO	NULL
Customer	FirstName	varchar	NO	60
Customer	LastName	varchar	NO	60
Customer	HomeAddress	varchar	NO	200

Query executed successfully.

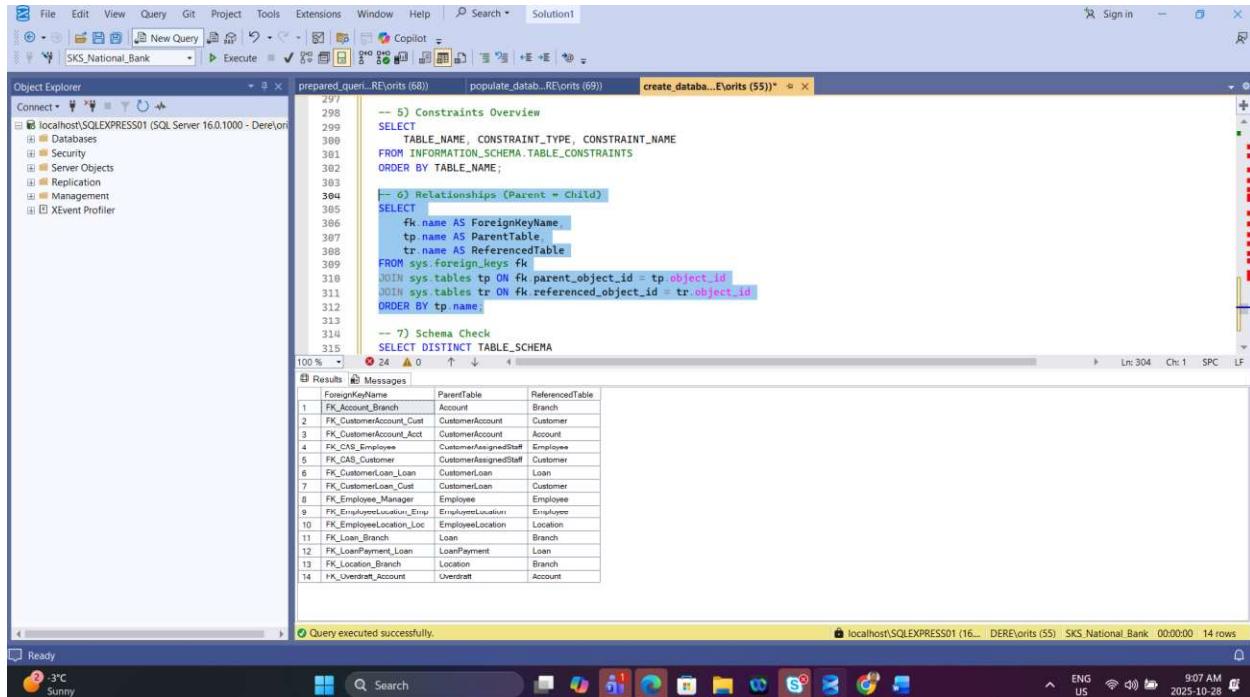
## 6) Constraints\_overview



Screenshot of SQL Server Management Studio (SSMS) showing a query results grid for constraints. The results grid displays 39 rows of data, including columns for TABLE\_NAME, CONSTRAINT\_TYPE, and CONSTRAINT\_NAME. The data includes various constraint types such as PRIMARY KEY, FOREIGN KEY, CHECK, and UNIQUE, across different tables like Account, Branch, Customer, and Employee.

TABLE_NAME	CONSTRAINT_TYPE	CONSTRAINT_NAME
1 Account	PRIMARY KEY	PK_Account_349D55B500B0AE
2 Account	FOREIGN KEY	FK_Account_Branch
3 Account	CHECK	OK_Account_Type
4 Account	CHECK	OK_Account_InheritedByType
5 Account	CHECK	OK_Account_IsNegative
6 Branch	PRIMARY KEY	PK_Branch_116837A1C0C4710C
7 Branch	UNIQUE	UQ_Branch_BranchName
8 Branch	CHECK	OK_Branch_NonNegative
9 Customers	PRIMARY KEY	PK_Customers_44AC5408EF103417
10 CustomerAccount	PRIMARY KEY	PK_CustomerAccount
11 CustomerAccount	FOREIGN KEY	FK_CustomerAccount_Cust
12 CustomerAccount	FOREIGN KEY	FK_CustomerAccount_Accd
13 CustomerAccount	CHECK	OK_CustomerAccount_Pct
14 CustomerAssignedStaff	PRIMARY KEY	PK_CustomerAssignedStaff
15 CustomerAssignedStaff	FOREIGN KEY	FK_CAS_Customer
16 CustomerAssignedStaff	FOREIGN KEY	FK_CAS_Employee
17 CustomerAssignedStaff	CHECK	OK_CAS_StaffRole

## 7) Relationship\_parents\_child



Screenshot of SQL Server Management Studio (SSMS) showing a query results grid for relationships. The results grid displays 14 rows of data, including columns for ForeignKeyName, ParentTable, and ReferencedTable. The data lists various foreign key relationships across different tables such as Account, Branch, Customer, and Employee.

ForeignKeyName	ParentTable	ReferencedTable
1 FK_Account_Branch	Account	Branch
2 FK_CustomerAccount_Cust	CustomerAccount	Customer
3 FK_CustomerAccount_Accd	CustomerAccount	Account
4 FK_CAS_Employee	CustomerAssignedStaff	Employee
5 FK_CAS_Customer	CustomerAssignedStaff	Customer
6 FK_CustomerLoan_Loan	CustomerLoan	Loan
7 FK_CustomerAcc_Cust	CustomerAcc	Customer
8 FK_Employee_Manager	Employee	Employee
9 FK_EmployeeLocation_Emp	EmployeeLocation	Employee
10 FK_EmployeeLocation_Loc	EmployeeLocation	Location
11 FK_Loan_Branch	Loan	Branch
12 FK_LoanPayment_Loan	LoanPayment	Loan
13 FK_Location_Branch	Location	Branch
14 FK_Overdraft_Account	Overdraft	Account

## 8) Schema\_check

The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top menu bar includes File, Edit, View, Query, Git, Project, Tool, Extensions, Window, Help, and Search. The toolbar has icons for New Query, Execute, and Save. The Object Explorer on the left shows a connection to 'localhost\SQLExpress01' (SQL Server 16.0.1000 - Databases). The main pane displays a query window with the following T-SQL code:

```
SELECT
    fk.name AS ForeignKeyName,
    tp.name AS ParentTable,
    tr.name AS ReferencedTable
FROM sys.foreign_keys fk
JOIN sys.tables tp ON fk.parent_object_id = tp.object_id
JOIN sys.tables tr ON fk.referenced_object_id = tr.object_id
ORDER BY tp.name;
```

Below the code, a comment block is shown:

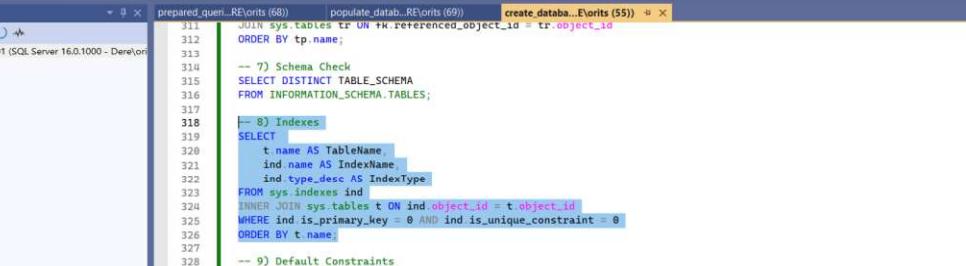
```
-- 7) Schemas Check
SELECT DISTINCT TABLE_SCHEMA
FROM INFORMATION_SCHEMA.TABLES;
```

Another comment block follows:

```
-- 8) Indexes
SELECT
    t.name AS TableName,
    ind.name AS IndexName,
    ind.type_desc AS IndexType
```

The results grid shows a single row with the value 'dbo' under the 'TABLE\_SCHEMA' column. The status bar at the bottom indicates 'Query executed successfully.' and 'localhost\SQLExpress01 (16... DFRUnits (55) - SKS\_National\_Bank 00:00:00 1 rows'.

## 9) Indexes



prepared\_queri...RE(68) populate\_data...RE(69) create\_database...E(55) ↴

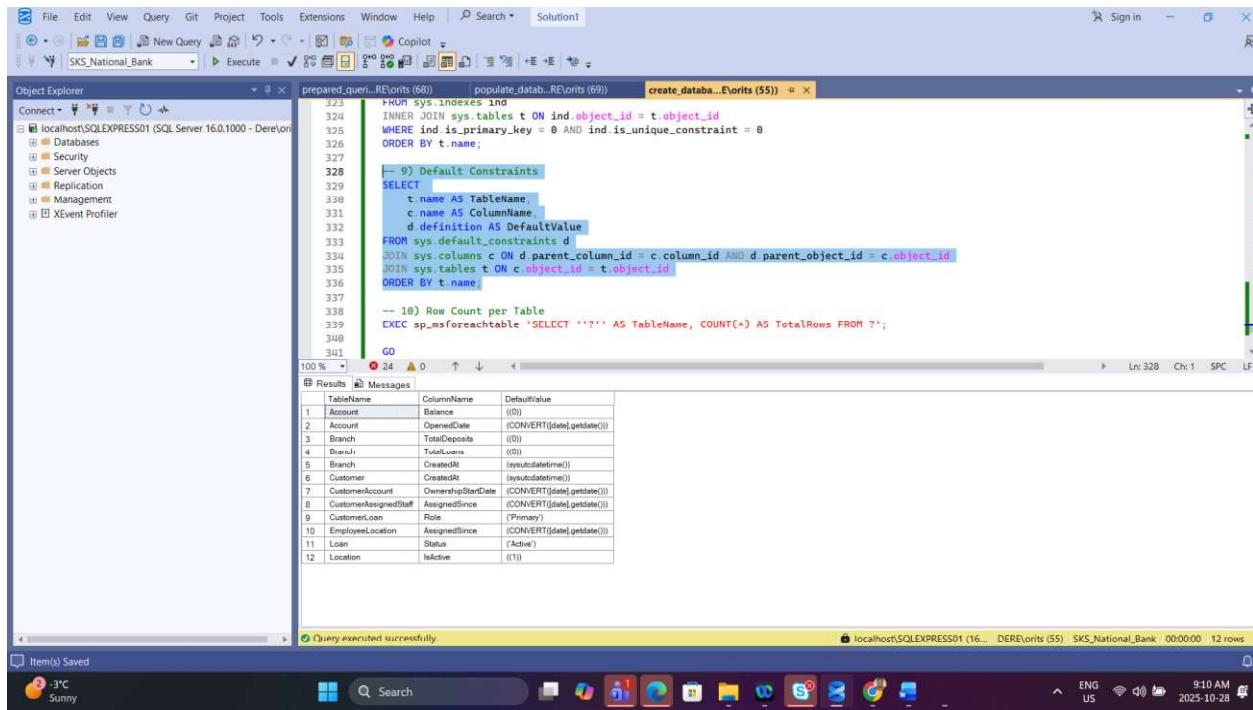
```
311 JOIN sys.tables tr UN t.referenced_object_id = tr.object_id
312 ORDER BY tp.name;
313
314 -- 7) Schema Check
315 SELECT DISTINCT TABLE_SCHEMA
316 FROM INFORMATION_SCHEMA.TABLES;
317
318 [-- 8) Indexes
319 SELECT
320     t.name AS TableName,
321     ind.name AS IndexName,
322     ind.type_desc AS IndexType
323 FROM sys.indexes ind
324 INNER JOIN sys.tables t ON ind.object_id = t.object_id
325 WHERE ind.is_primary_key = 0 AND ind.is_unique_constraint = 0
326 ORDER BY t.name;
327
328 -- 9) Default Constraints
329 SELECT
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
```

100 % 24 0 ↑ ↓ < > Ln: 318 Ch: 1 SPC LF

TableName	IndexName	IndexType
1 Account	IX_Account_BranchID	NONCLUSTERED
2 Customer	IX_Customer_Email_IsNull	NONCLUSTERED
3 CustomerAccount	IX_CustomerAccount_AccountID	NONCLUSTERED
4 CustomerHasInvested	IX_CMS_EmployeeID	NONCLUSTERED
5 CustomerLoan	IX_CustomerLoan_LoanID	NONCLUSTERED
6 Employee	IX_Employee_ManagerID	NONCLUSTERED
7 EmployeeLocation	IX_EmployeeLocation_EmployeeID	NONCLUSTERED
8 EmployeeLocation	IX_EmployeeLocation_LocationID	NONCLUSTERED
9 Loan	IX_Loan_BranchID	NONCLUSTERED
10 LoanPayment	IX_LoanPayment_LoanID	NONCLUSTERED
11 Location	IX_Location_BranchID	NONCLUSTERED
12 Overdraft	IX_Overdraft_AccountID	NONCLUSTERED

Query executed successfully.

## 10) Default\_constraint



The screenshot shows the SSMS interface with the 'Object Explorer' on the left and a 'Results' tab on the right. The query window contains the following T-SQL code:

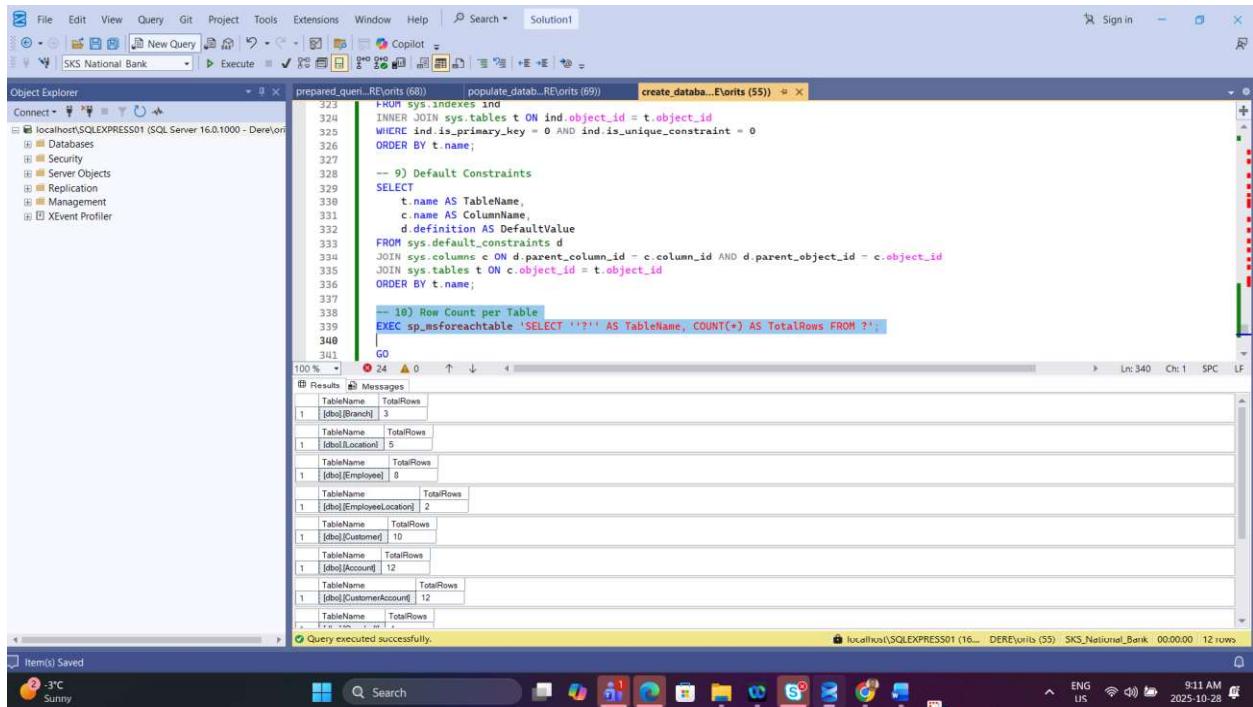
```
323 PRINT sys.indexes ind
324  INNER JOIN sys.tables t ON ind.object_id = t.object_id
325 WHERE ind.is_primary_key = 0 AND ind.is_unique_constraint = 0
326 ORDER BY t.name;
327
328 -- 9) Default Constraints
329 SELECT
330     t.name AS TableName,
331     c.name AS ColumnName,
332     d.definition AS DefaultValue
333 FROM sys.default_constraints d
334 JOIN sys.columns c ON d.parent_column_id = c.column_id AND d.parent_object_id = c.object_id
335 JOIN sys.tables t ON c.object_id = t.object_id
336 ORDER BY t.name;
337
338 -- 10) Row Count per Table
339 EXEC sp_msforeachtable 'SELECT ''?'' AS TableName, COUNT(*) AS TotalRows FROM ?';
340
341 GO
```

The results table shows 12 rows of data:

TableName	ColumnName	DefaultValue
Account	Balance	(0)
Account	OpenedDate	(CONVERT(date, getdate()))
Branch	TotalDeposits	(0)
Branch	TotalLearns	(0)
Branch	CreatedAt	(sysutcdatetime())
Customer	CreatedAt	(sysutcdatetime())
CustomerAccount	OwnershipStartDate	(CONVERT(date, getdate()))
CustomerAssignedStatus	AssignedSince	(CONVERT(date, getdate()))
CustomerLoan	Rate	(Percent)
EmployeeLocation	AssignedSince	(CONVERT(date, getdate()))
Loan	Status	(Active)
Location	IsActive	(1)

Message bar: Query executed successfully.

## 11) Row-count\_per\_table



The screenshot shows the SSMS interface with the 'Object Explorer' on the left and a 'Results' tab on the right. The query window contains the following T-SQL code:

```
323 PRINT sys.indexes ind
324  INNER JOIN sys.tables t ON ind.object_id = t.object_id
325 WHERE ind.is_primary_key = 0 AND ind.is_unique_constraint = 0
326 ORDER BY t.name;
327
328 -- 9) Default Constraints
329 SELECT
330     t.name AS TableName,
331     c.name AS ColumnName,
332     d.definition AS DefaultValue
333 FROM sys.default_constraints d
334 JOIN sys.columns c ON d.parent_column_id = c.column_id AND d.parent_object_id = c.object_id
335 JOIN sys.tables t ON c.object_id = t.object_id
336 ORDER BY t.name;
337
338 -- 10) Row Count per Table
339 EXEC sp_msforeachtable 'SELECT ''?'' AS TableName, COUNT(*) AS TotalRows FROM ?';
340
341 GO
```

The results table shows 12 rows of data:

TableName	TotalRows
[dbo].[Branch]	3
[dbo].[Location]	5
[dbo].[Employee]	8
[dbo].[EmployeeLocation]	2
[dbo].[Customer]	10
[dbo].[Account]	12
[dbo].[CustomerAccount]	12

Message bar: Query executed successfully.

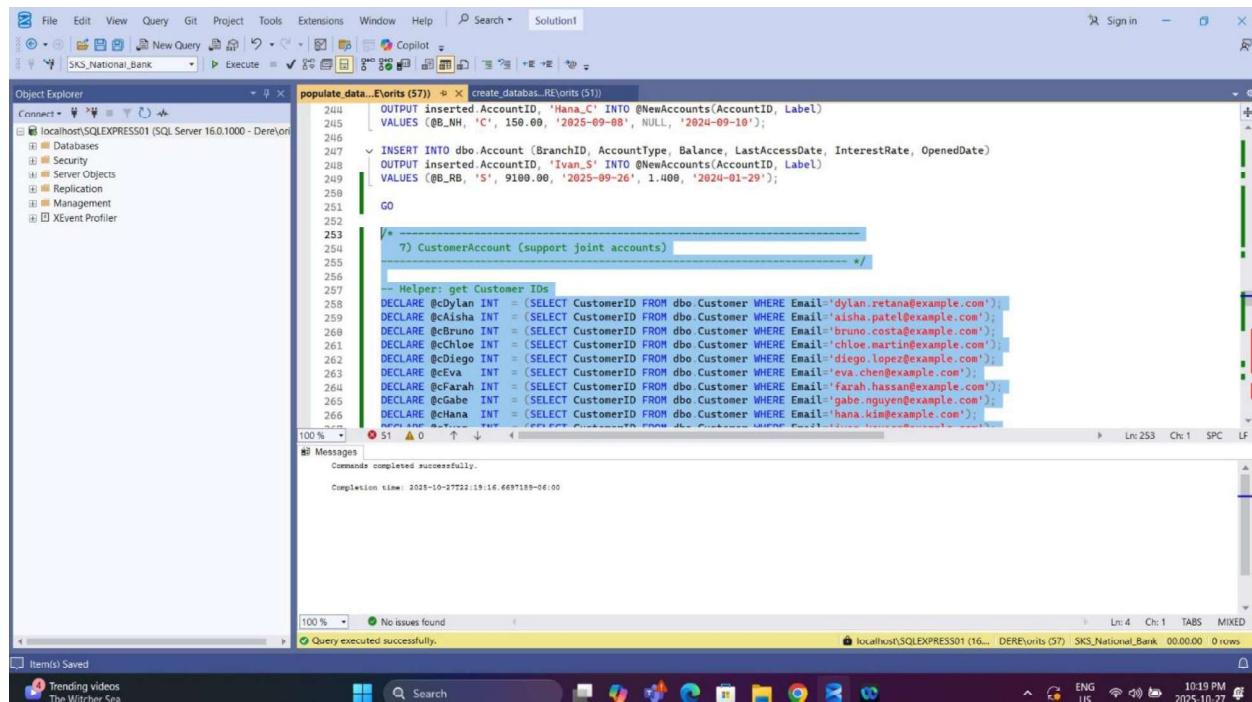
## populate\_database\_database



```
File Edit View Query Git Project Tools Extensions Window Help | Search | Solution1
Sign in
Object Explorer
Connect x
localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - DERE) | SKS_National_Bank
Databases
Security
Server Objects
Replication
Management
XEvent Profiler
populate_data...Eroris (57) x create_databases...RE(roris (51))
181     ('Diego', 'Lopez', '15 Elm St, Edmonton, AB', 'diego.lopez@example.com', '780-100-1004'),  
182     ('Eva', 'Chen', '16 Elm St, Calgary, AB', 'eva.chen@example.com', '403-100-1005'),  
183     ('Farah', 'Hassan', '17 Elm St, Edmonton, AB', 'farah.hassan@example.com', '780-100-1006'),  
184     ('Gabe', 'Nguyen', '18 Elm St, Calgary, AB', 'gabe.nguyen@example.com', '403-100-1007'),  
185     ('Hana', 'Kim', '19 Elm St, Calgary, AB', 'hana.kim@example.com', '403-100-1008'),  
186     ('Ivan', 'Kovacs', '20 Elm St, Edmonton, AB', 'ivan.kovacs@example.com', '780-100-1009');  
187  
188 GO  
189  
190 /*  
191 6) Account (x12; mix C & S). Capture generated IDs with OUTPUT.  
192 */  
193  
194 DECLARE @NewAccounts TABLE (AccountID BIGINT, Label VARCHAR(50));  
195  
196 -- Downtown HQ branch accounts  
197 DECLARE @B_DTHQ INT = (SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ');  
198 DECLARE @B_NH INT = (SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill');  
199 DECLARE @B_RB INT = (SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend');  
200  
201 INSERT INTO dbo.Account (BranchID, AccountType, Balance, LastAccessDate, InterestRate, OpenedDate)  
202     OUTPUT inserted.AccountID, 'Dylan,C' INTO @NewAccounts(AccountID, Label)  
203     VALUES (@B_DTHQ, 'C', 1200.00, '2025-09-10', NULL, '2024-03-01');  
204  
205 INSERT INTO dbo.Account (BranchID, AccountType, Balance, LastAccessDate, InterestRate, OpenedDate)  
206     OUTPUT inserted.AccountID, 'Dylan,S' INTO @NewAccounts(AccountID, Label)  
207     VALUES (@B_DTHQ, 'S', 3500.00, '2025-09-20', 1.25, '2024-03-01');
```

## 1) populate\_database\_account

## 2) populate\_database.sql Customer Account



```
File Edit View Query Git Project Tools Extensions Window Help Search Solution1
localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Derevon)
Object Explorer
populate_data...Exorts (57)  create_database...RE(exorts (51))
204     OUTPUT inserted.AccountID, 'Hana_C' INTO @NewAccounts(AccountID, Label)
205     VALUES (@B_NH, 'C', 156.00, '2025-09-08', NULL, '2024-09-10');
206
207     INSERT INTO dbo.Account (BranchID, AccountType, Balance, LastAccessDate, InterestRate, OpenedDate)
208     OUTPUT inserted.AccountID, 'Ivan_S' INTO @NewAccounts(AccountID, Label)
209     VALUES (@B_RB, 'S', 9100.00, '2025-09-26', 1.400, '2024-01-29');
210
211     GO
212
213     /* ?) CustomerAccount (support joint accounts)
214
215
216     -- Helper: get Customer IDs
217     DECLARE @cDylan INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='dylan.retana@example.com');
218     DECLARE @cAisha INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='aisha.patel@example.com');
219     DECLARE @cBrun INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='bruno.costa@example.com');
220     DECLARE @cChloe INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='chloe.martin@example.com');
221     DECLARE @cDiego INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='diego.lopez@example.com');
222     DECLARE @cEva INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='eva.chen@example.com');
223     DECLARE @cFarah INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='farah.hassan@example.com');
224     DECLARE @cGabe INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='gabe.nguyen@example.com');
225     DECLARE @cHana INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='hana.kim@example.com');
226
227     -- Create on joint account example (Chloe & Aisha share Chloe_C 50/50)
228     INSERT INTO dbo.CustomerAccount (CustomerID, AccountID, OwnershipPercent)
229     VALUES (@cAisha, @aChloeC, 50);
230
231     GO
232
233     /* 8) Overdraft (chequing-only) - tie to chequing accounts
234
235
236     -- Re-declare valid chequing AccountID directly from dbo.Account
237     DECLARE @aDylanC BIGINT = (SELECT TOP 1 AccountID
238         FROM dbo.Account
239         WHERE AccountType = 'C'
240         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'Downtown HQ'));
241
242     DECLARE @aAishaC BIGINT = (SELECT TOP 1 AccountID
243         FROM dbo.Account
244         WHERE AccountType = 'C'
245         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'Downtown HQ')
246         AND Balance = 800.00);
247
248     DECLARE @aChloeC BIGINT = (SELECT TOP 1 AccountID
249         FROM dbo.Account
250         WHERE AccountType = 'C'
251         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'North Hill'));
252
253     DECLARE @aHanaC BIGINT = (SELECT TOP 1 AccountID
254         FROM dbo.Account
255         WHERE AccountType = 'C'
256         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'North Hill'));
```

Messages

Commands completed successfully.

Completion time: 2025-10-27T22:19:16.6497189-06:00

100 % 0 issues found

Query executed successfully.

localhost\SQLEXPRESS01 (16... DERE(exorts (57) SKS\_National\_Bank 00:00:00 0 rows

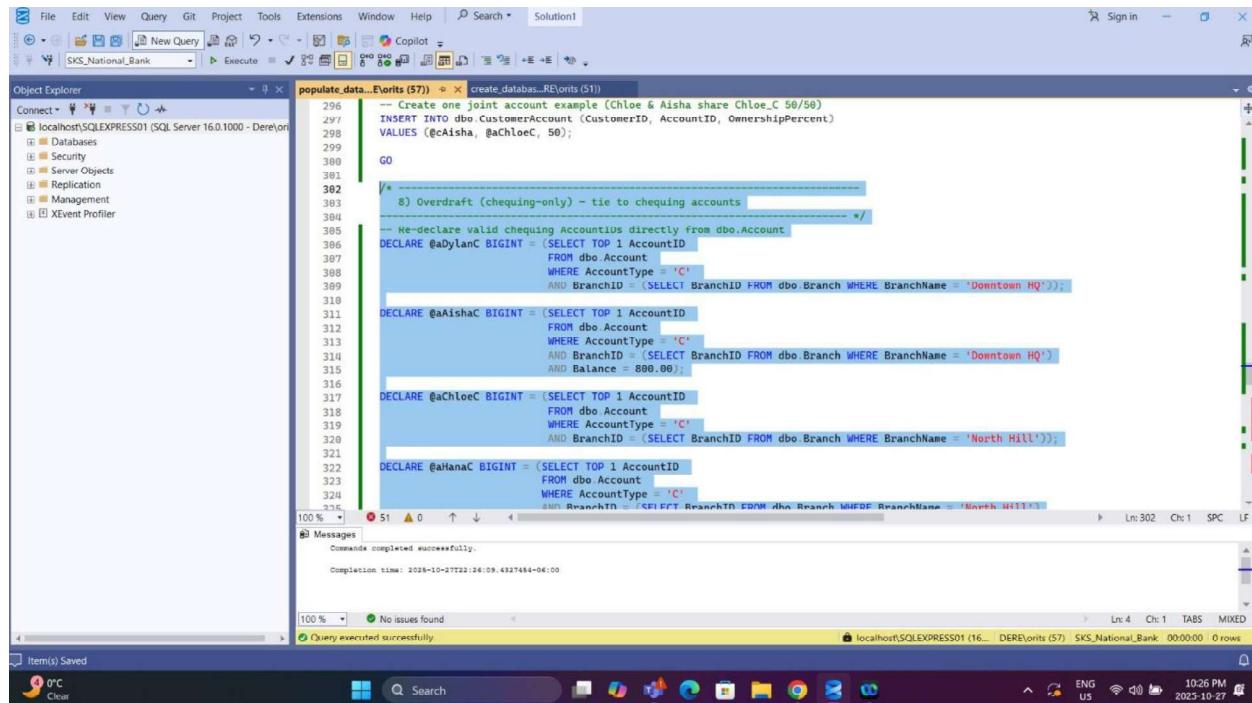
Item(s) Saved

0% 0C Clear

Search

10:19 PM 2025-10-27

## 3) populate\_database.sql Overdraft



```
File Edit View Query Git Project Tools Extensions Window Help Search Solution1
localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Derevon)
Object Explorer
populate_data...Exorts (57)  create_database...RE(exorts (51))
296     -- Create on joint account example (Chloe & Aisha share Chloe_C 50/50)
297     INSERT INTO dbo.CustomerAccount (CustomerID, AccountID, OwnershipPercent)
298     VALUES (@cAisha, @aChloeC, 50);
299
300     GO
301
302     /* 8) Overdraft (chequing-only) - tie to chequing accounts
303
304
305     -- Re-declare valid chequing AccountID directly from dbo.Account
306     DECLARE @aDylanC BIGINT = (SELECT TOP 1 AccountID
307         FROM dbo.Account
308         WHERE AccountType = 'C'
309         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'Downtown HQ'));
310
311     DECLARE @aAishaC BIGINT = (SELECT TOP 1 AccountID
312         FROM dbo.Account
313         WHERE AccountType = 'C'
314         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'Downtown HQ')
315         AND Balance = 800.00);
316
317     DECLARE @aChloeC BIGINT = (SELECT TOP 1 AccountID
318         FROM dbo.Account
319         WHERE AccountType = 'C'
320         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'North Hill'));
321
322     DECLARE @aHanaC BIGINT = (SELECT TOP 1 AccountID
323         FROM dbo.Account
324         WHERE AccountType = 'C'
325         AND BranchID = (SELECT BranchID FROM dbo.Branch WHERE BranchName = 'North Hill'));
```

Messages

Commands completed successfully.

Completion time: 2025-10-27T22:24:09.4327484-06:00

100 % 0 issues found

Query executed successfully.

localhost\SQLEXPRESS01 (16... DERE(exorts (57) SKS\_National\_Bank 00:00:00 0 rows

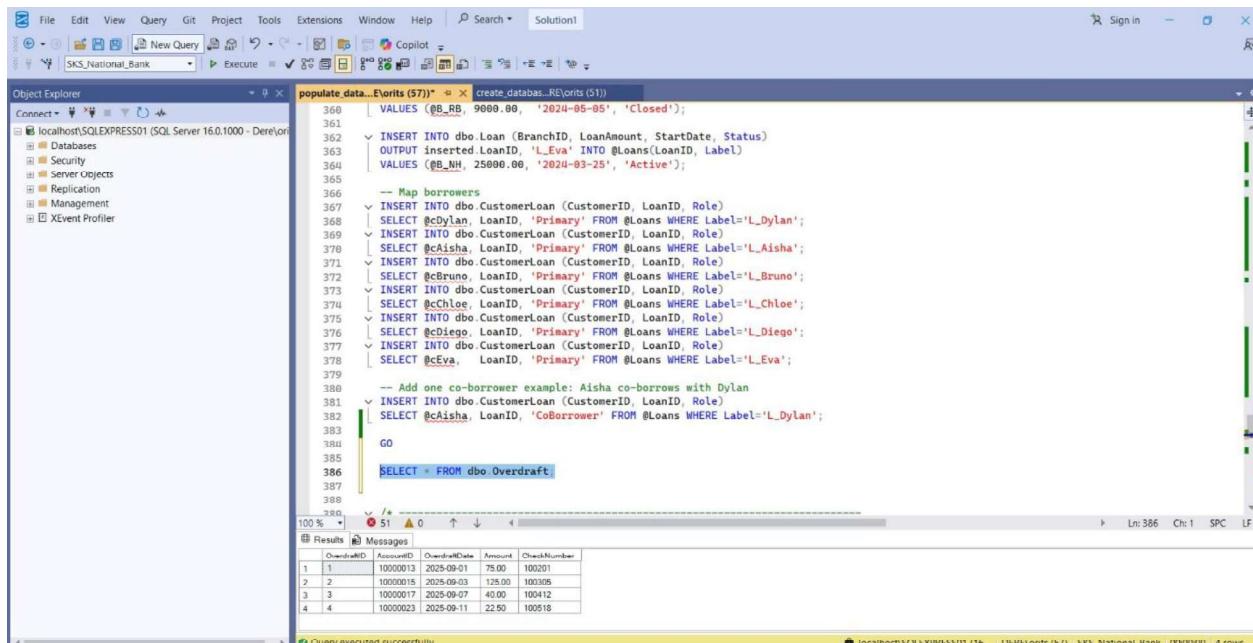
Item(s) Saved

0% 0C Clear

Search

10:26 PM 2025-10-27

#### 4) populate\_database.sql Overdraft\_Test



```

File Edit View Query Git Project Tools Extensions Window Help | Search | Solution1
localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Deref) | Execute | New Query | Copilot | Object Explorer | Databases | Security | Server Objects | Replication | Management | XEvent Profiler |(populate_database.sql(57)) | create_databases.REOrbits (51)
VALUES (@B_RB, 9000.00, '2024-05-05', 'Closed');
INSERT INTO dbo.Overdraft (OverdraftID, AccountID, OverdraftDate, Amount, CheckNumber)
VALUES (@B_RB, 9000.00, '2024-05-05', 'Closed');
SELECT @LoanID = inserted.LoanID FROM @Loans(LoanID, Label);
VALUES (@B_NH, 25000.00, '2024-03-25', 'Active');
-- Map borrowers
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cDylan.LoanID, 'Primary' FROM @Loans WHERE Label='L_Dylan';
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cAisha.LoanID, 'Primary' FROM @Loans WHERE Label='L_Aisha';
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cBruno.LoanID, 'Primary' FROM @Loans WHERE Label='L_Bruno';
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cChloe.LoanID, 'Primary' FROM @Loans WHERE Label='L_Chloe';
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cDiego.LoanID, 'Primary' FROM @Loans WHERE Label='L_Diego';
SELECT @cEva.LoanID, 'Primary' FROM @Loans WHERE Label='L_Eva';
-- Add one co-borrower example: Aisha co-borrows with Dylan
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
SELECT @cAisha.LoanID, 'CoBorrower' FROM @Loans WHERE Label='L_Dylan';
GO
SELECT * FROM dbo.Overdraft;

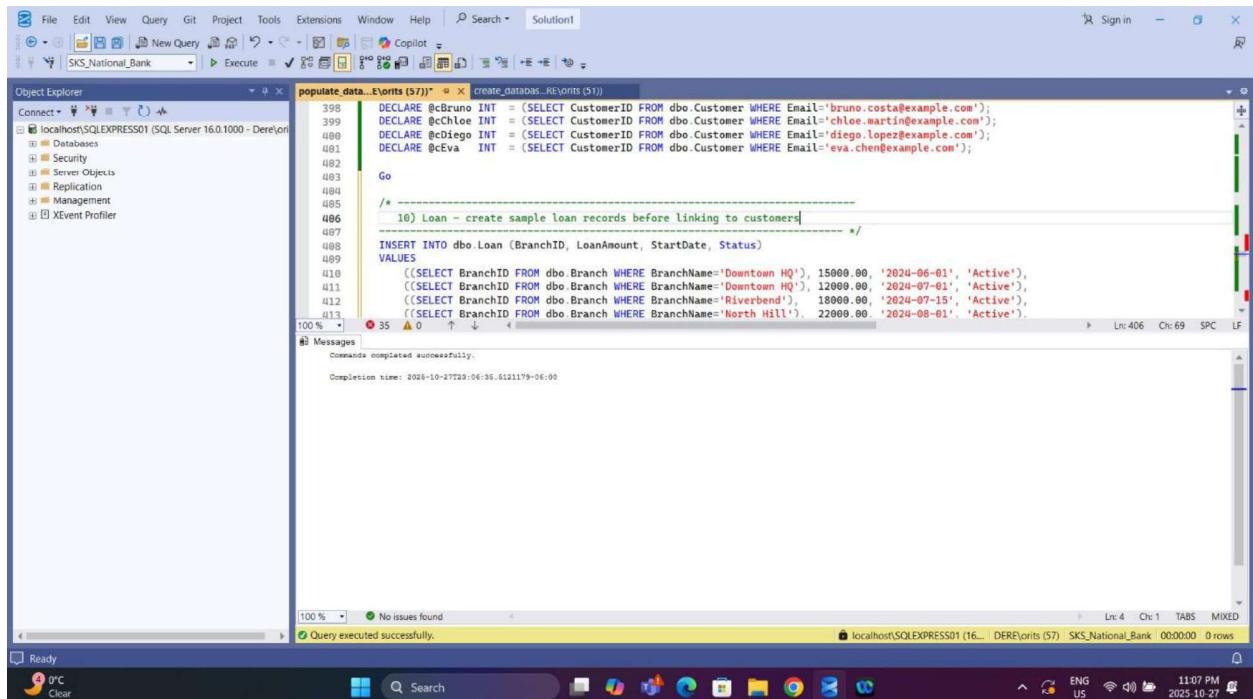
```

Result

OverdraftID	AccountID	OverdraftDate	Amount	CheckNumber
1	10000013	2025-09-01	75.00	100201
2	10000015	2025-09-03	125.00	100305
3	10000017	2025-09-07	40.00	100412
4	10000023	2025-09-11	22.50	100518

Query executed successfully.

#### 5) populate\_database.sql Loan Table.



```

File Edit View Query Git Project Tools Extensions Window Help | Search | Solution1
localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Deref) | Execute | New Query | Copilot | Object Explorer | Databases | Security | Server Objects | Replication | Management | XEvent Profiler |(populate_database.sql(57)) | create_databases.REOrbits (51)
DECLARE @cBruno INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='bruno.costa@example.com');
DECLARE @cChloe INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='chloe.martin@example.com');
DECLARE @cDiego INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='diego.lopez@example.com');
DECLARE @cEva INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='eva.chen@example.com');
Go
/* -----
10) Loan - create sample loan records before linking to customers
----- */
INSERT INTO dbo.Overdraft (OverdraftID, AccountID, OverdraftDate, Amount, CheckNumber)
VALUES
((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 15000.00, '2024-06-01', 'Active'),
((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 12000.00, '2024-07-01', 'Active'),
((SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend'), 18000.00, '2024-07-15', 'Active'),
((SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill'), 22000.00, '2024-08-01', 'Active');

```

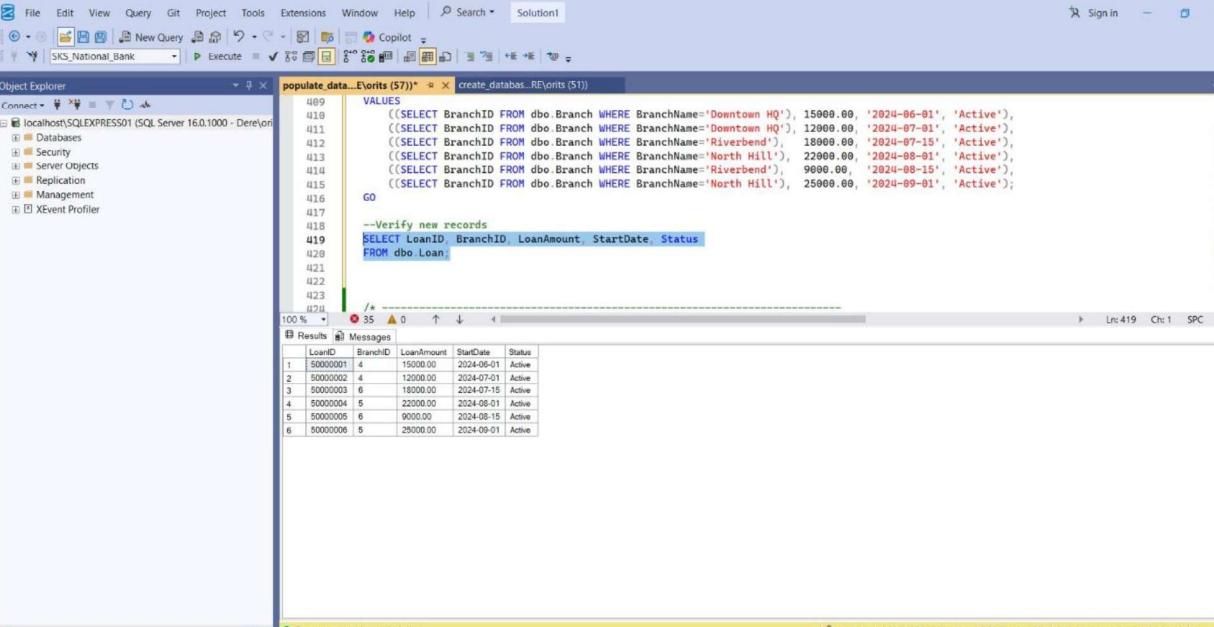
Messages

Commands completed successfully.

Completion time: 2025-10-27T23:06:38.5121179-06:00

Query executed successfully.

## 6) populate\_database.sql Loan Table-New records



File Edit View Query Git Project Tools Extensions Window Help Search Solution

Object Explorer

localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Derevents) -> [SKS\_National\_Bank]

populate\_data..Events (57) > create\_databases..RE\events (51)

```
VALUES
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 15000.00, '2024-06-01', 'Active'),
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 12000.00, '2024-07-01', 'Active'),
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend'), 18000.00, '2024-07-15', 'Active'),
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill'), 22000.00, '2024-08-01', 'Active'),
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend'), 9000.00, '2024-08-15', 'Active'),
    ((SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill'), 25000.00, '2024-09-01', 'Active');

GO

--Verify new records
SELECT LoanID, BranchID, LoanAmount, StartDate, Status
FROM dbo.Loan;
```

Result Messages

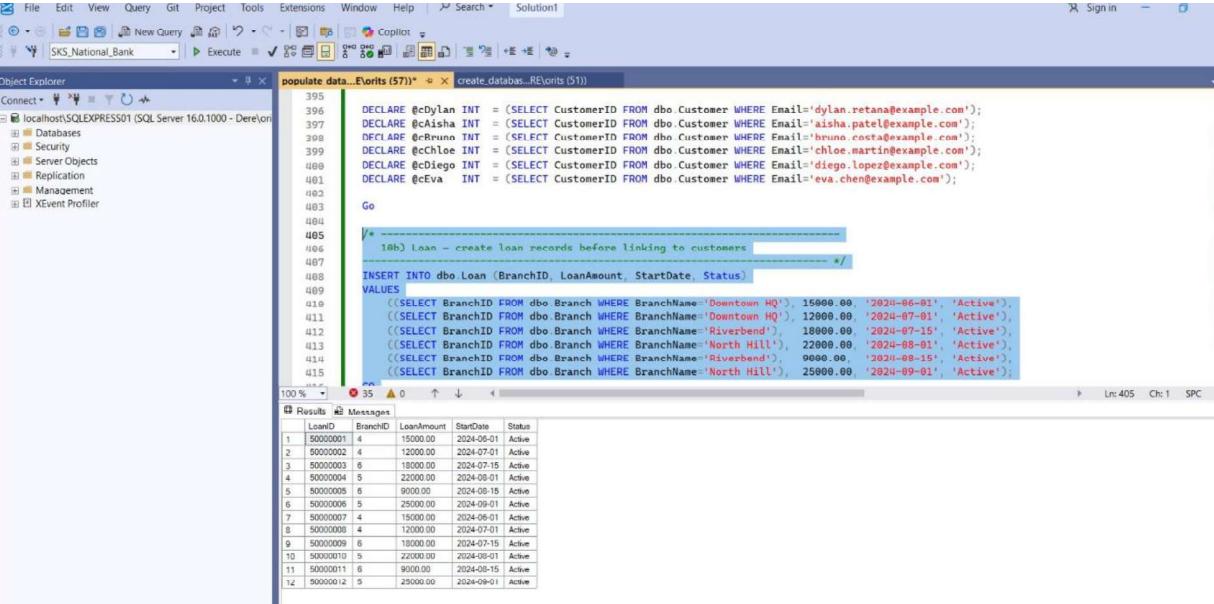
LoanID	BranchID	LoanAmount	StartDate	Status
1	50000001	15000.00	2024-06-01	Active
2	50000002	12000.00	2024-07-01	Active
3	50000003	18000.00	2024-07-15	Active
4	50000004	22000.00	2024-08-01	Active
5	50000005	9000.00	2024-08-15	Active
6	50000006	25000.00	2024-09-01	Active

Query executed successfully.

localhost\SQLEXPRESS01 (16... DERE\events (57) SKS\_National\_Bank 00:00:00 6 rows

Ready

7) populate\_database.sql Loan Table-New records accounts linkage



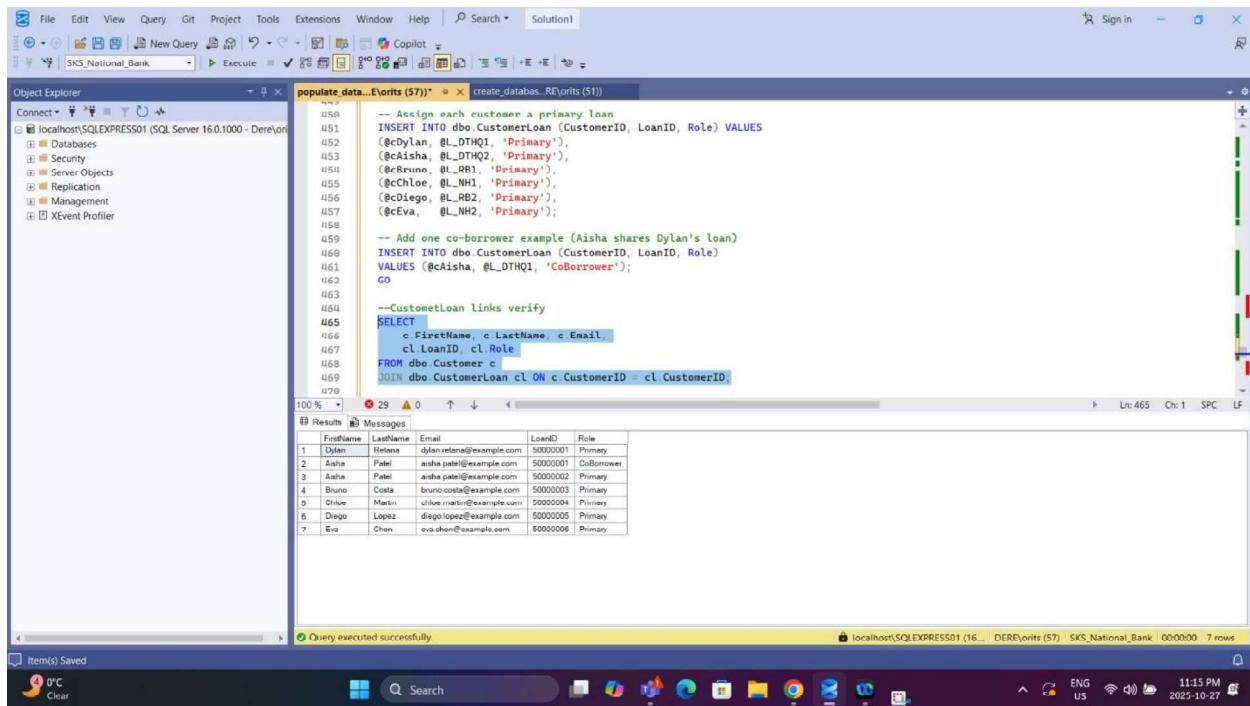
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left shows a connection to 'localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - Dereon)'. The query editor window contains a script for populating a table named 'Everts' with customer data and then inserting loan records into the 'Loan' table. The results grid at the bottom shows 12 rows of loan data with columns: LoanID, BranchID, LoanAmount, StartDate, and Status. A status bar at the bottom right indicates the connection is to 'localhost\SQLEXPRESS01 (16...' and the session is 'DERE\orbits (51)'.

```
395
396  DECLARE @dylan INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='dylan.retana@example.com');
397  DECLARE @aisha INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='aisha.patel@example.com');
398  DECLARE @Bruno INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='bruno.castro@example.com');
399  DECLARE @chloe INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='chloe.martin@example.com');
400  DECLARE @Diego INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='diego.lopez@example.com');
401  DECLARE @Eva INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='eva.chen@example.com');
402
403  Go
404
405  /* 10b) Loan - create loan records before linking to customers */
406
407  INSERT INTO dbo.Loan (BranchID, LoanAmount, StartDate, Status)
408  VALUES
409      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 15000.00, '2024-06-01', 'Active'),
410      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Downtown HQ'), 12000.00, '2024-07-01', 'Active'),
411      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend'), 18000.00, '2024-07-15', 'Active'),
412      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill'), 22000.00, '2024-08-01', 'Active'),
413      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='Riverbend'), 9000.00, '2024-08-15', 'Active'),
414      ((SELECT BranchID FROM dbo.Branch WHERE BranchName='North Hill'), 25000.00, '2024-09-01', 'Active')
415
```

LoanID	BranchID	LoanAmount	StartDate	Status
1	1	15000.00	2024-06-01	Active
2	2	12000.00	2024-07-01	Active
3	3	18000.00	2024-07-15	Active
4	4	22000.00	2024-08-01	Active
5	5	9000.00	2024-08-15	Active
6	6	25000.00	2024-09-01	Active
7	7	15000.00	2024-06-01	Active
8	8	12000.00	2024-07-01	Active
9	9	18000.00	2024-07-15	Active
10	10	22000.00	2024-08-01	Active
11	11	9000.00	2024-08-15	Active
12	12	25000.00	2024-09-01	Active

Query executed successfully.

## 8) populate\_database.sql Loan Table-CustomerLoan link verify



```

-- Assign each customer a primary loan
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role) VALUES
    (@cDylan, @L_DTHQ1, 'Primary'),
    (@cAisha, @L_DTHQ2, 'Primary'),
    (@cBruno, @L_RB1, 'Primary'),
    (@cChloe, @L_NH1, 'Primary'),
    (@cDiego, @L_RB2, 'Primary'),
    (@cEva, @L_NH2, 'Primary');

-- Add one co-borrower example (Aisha shares Dylan's loan)
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
VALUES (@cAisha, @L_DTHQ1, 'CoBorrower');
GO

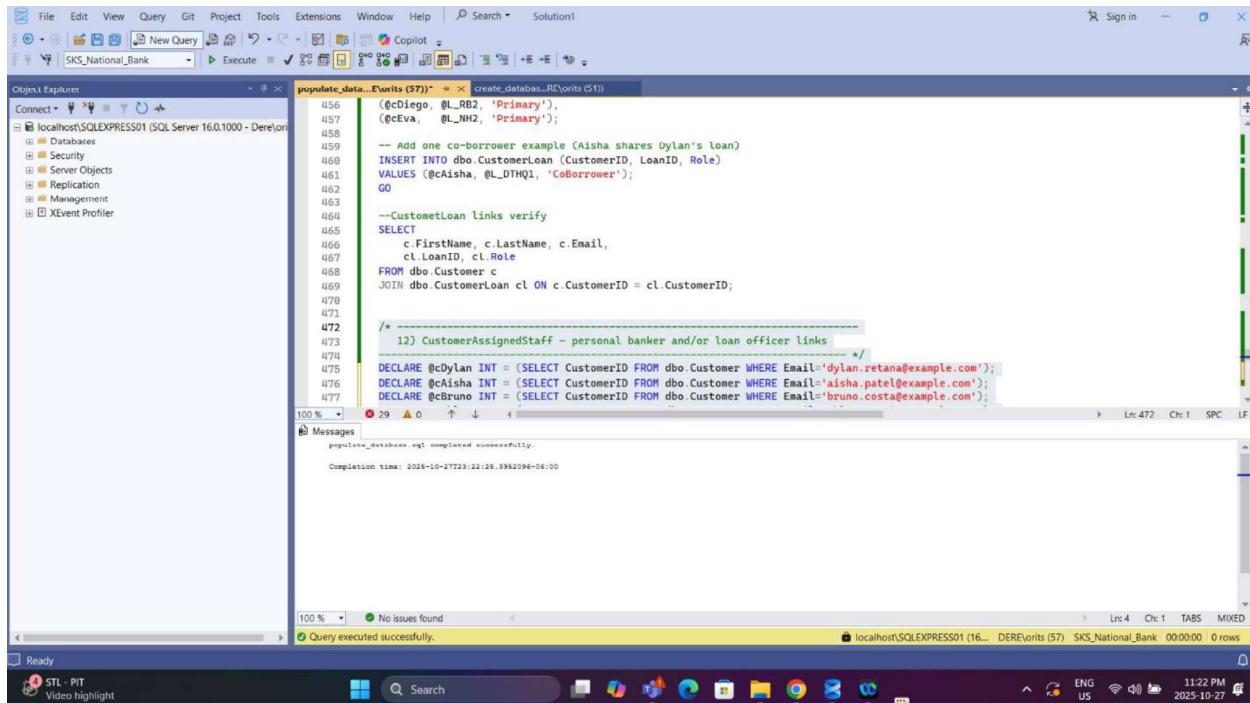
--CustomerLoan links verify
SELECT
    c.FirstName, c.LastName, c.Email,
    cl.LoanID, cl.Role
FROM dbo.Customer c
JOIN dbo.CustomerLoan cl ON c.CustomerID = cl.CustomerID;

```

FirstName	LastName	Email	LoanID	Role
Dylan	Retana	dylan.retana@example.com	50000001	Primary
Aisha	Patel	aisha.patel@example.com	50000001	CoBorrower
Aisha	Patel	aisha.patel@example.com	50000002	Primary
Bruno	Costa	bruno.costa@example.com	50000003	Primary
Chloe	Martin	chloe.martin@example.com	50000004	Primary
Diego	Lopez	diego.lopez@example.com	50000005	Primary
Eva	Chen	eva.chen@example.com	50000006	Primary

Query executed successfully.

## 9) populate\_database.sql Loan Table-CustomerAssigned Staff



```

-- Assign each customer a primary loan
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role) VALUES
    (@cDiego, @L_RB2, 'Primary'),
    (@cEva, @L_NH2, 'Primary');

-- Add one co-borrower example (Aisha shares Dylan's loan)
INSERT INTO dbo.CustomerLoan (CustomerID, LoanID, Role)
VALUES (@cAisha, @L_DTHQ1, 'CoBorrower');
GO

--CustomerLoan links verify
SELECT
    c.FirstName, c.LastName, c.Email,
    cl.LoanID, cl.Role
FROM dbo.Customer c
JOIN dbo.CustomerLoan cl ON c.CustomerID = cl.CustomerID;

/*
  12) CustomerAssignedStaff - personal banker and/or loan officer links
*/
DECLARE @cDylan INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='dylan.retana@example.com');
DECLARE @cAisha INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='aisha.patel@example.com');
DECLARE @cBruno INT = (SELECT CustomerID FROM dbo.Customer WHERE Email='bruno.costa@example.com');

```

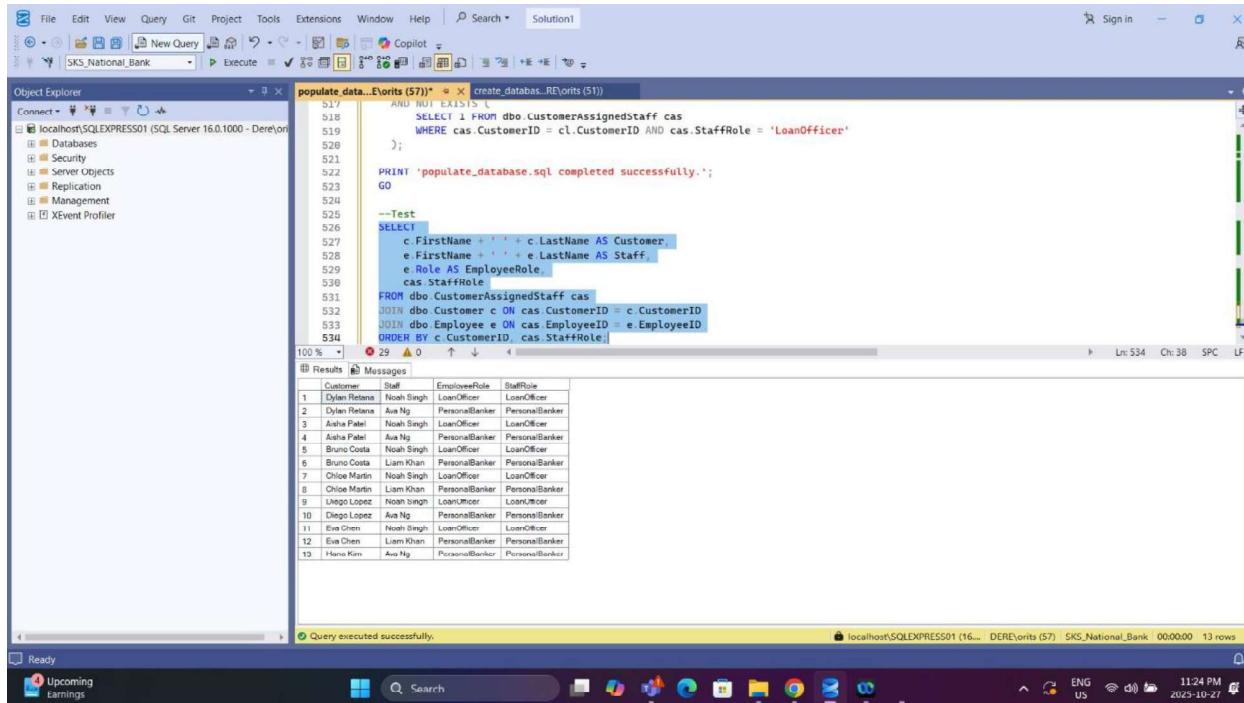
populate\_database\_Everts.sql completed successfully.

Compilation time: 2025-10-27T23:22:28.3982098+08:00

No issues found.

Query executed successfully.

## 10) populate\_database.sql Loan Table-CustomerAssigned Staff.Test



```

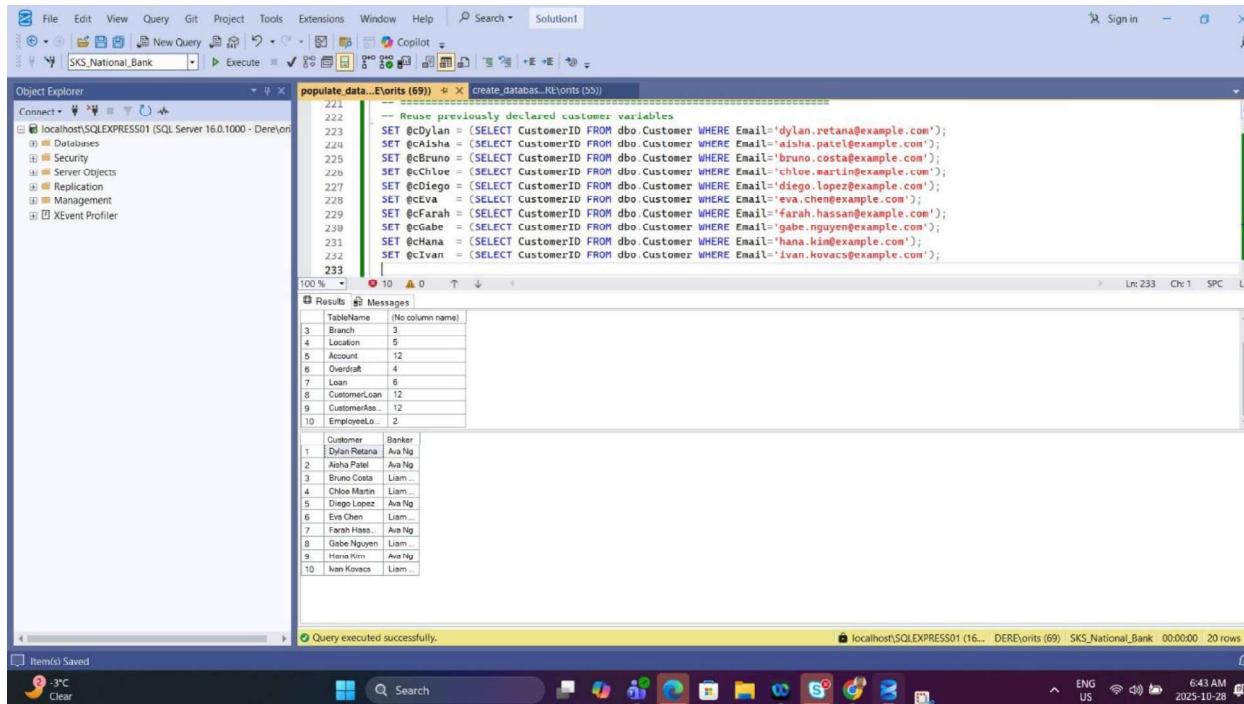
517      AND NOT EXISTS (
518          SELECT 1 FROM dbo.CustomerAssignedStaff cas
519          WHERE cas.CustomerID = cl.CustomerID AND cas.StaffRole = 'LoanOfficer'
520      );
521
522      PRINT 'populate_database.sql completed successfully.';
523      GO
524
525      --Test
526      SELECT
527          c.FirstName + ' ' + c.LastName AS Customer,
528          e.FirstName + ' ' + e.LastName AS Staff,
529          e.Role AS EmployeeRole,
530          cas.StaffRole
531      FROM dbo.CustomerAssignedStaff cas
532      JOIN dbo.Customer c ON cas.CustomerID = c.CustomerID
533      JOIN dbo.Employee e ON cas.EmployeeID = e.EmployeeID
534      ORDER BY c.CustomerID, cas.StaffRole;
  
```

Results

Customer	Staff	EmployeeRole	StaffRole
1 Dylan Retana	Noah Singh	LoanOfficer	LoanOfficer
2 Dylan Retana	Ava Ng	PersonnelBanker	PersonnelBanker
3 Aisha Patel	Noah Singh	LoanOfficer	LoanOfficer
4 Aisha Patel	Ava Ng	PersonnelBanker	PersonnelBanker
5 Bruno Costa	Noah Singh	LoanOfficer	LoanOfficer
6 Bruno Costa	Liam Khan	PersonnelBanker	PersonnelBanker
7 Chloe Martin	Noah Singh	LoanOfficer	LoanOfficer
8 Chloe Martin	Liam Khan	PersonnelBanker	PersonnelBanker
9 Diego Lopez	Noah Singh	LoanOfficer	LoanOfficer
10 Diego Lopez	Liam Khan	PersonnelBanker	PersonnelBanker
11 Eva Chen	Noah Singh	LoanOfficer	LoanOfficer
12 Eva Chen	Liam Khan	PersonnelBanker	PersonnelBanker
13 Hana Kim	Ava Ng	PersonnelBanker	PersonnelBanker

Query executed successfully.

## 11) populate\_database.sql Loan Table-Final Result



```

221      -- Reuse previously declared customer variables
222      SET @cDylan = (SELECT CustomerID FROM dbo.Customer WHERE Email='dylan.retana@example.com');
223      SET @cAisha = (SELECT CustomerID FROM dbo.Customer WHERE Email='aisha.patel@example.com');
224      SET @cBruno = (SELECT CustomerID FROM dbo.Customer WHERE Email='bruno.costa@example.com');
225      SET @cChloe = (SELECT CustomerID FROM dbo.Customer WHERE Email='chloe.martin@example.com');
226      SET @cDiego = (SELECT CustomerID FROM dbo.Customer WHERE Email='diego.lopez@example.com');
227      SET @cEva = (SELECT CustomerID FROM dbo.Customer WHERE Email='eva.chen@example.com');
228      SET @cFarah = (SELECT CustomerID FROM dbo.Customer WHERE Email='farah.hassan@example.com');
229      SET @cHana = (SELECT CustomerID FROM dbo.Customer WHERE Email='hana.kim@example.com');
230      SET @cIvan = (SELECT CustomerID FROM dbo.Customer WHERE Email='ivan.kovacs@example.com');
231
232      |
233
  
```

Results

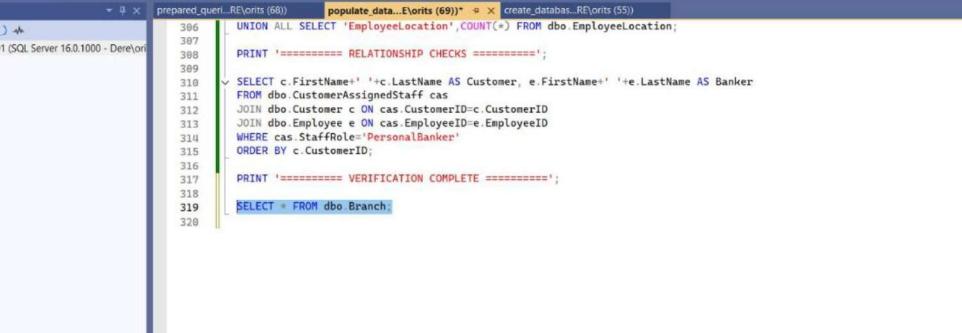
TableName	(No column name)
3 Branch	
4 Location	5
5 Account	12
6 Overdraft	4
7 Loan	6
8 CustomerLoan	12
9 Customers...	12
10 EmployeeLo...	2

Customer	Banker
1 Dylan Retana	Ava Ng
2 Aisha Patel	Ava Ng
3 Bruno Costa	Liam ...
4 Chloe Martin	Liam ...
5 Diego Lopez	Ava Ng
6 Eva Chen	Liam ...
7 Farah Hass...	Ava Ng
8 Hana Kim	Liam ...
9 Hana Kim	Ava Ng
10 Ivan Kovacs	Liam ...

Query executed successfully.

## 12) populate\_database.sql Loan Table-Confirmation

13) populate\_database.sql TestBranch



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top menu bar includes File, Edit, View, Query, Git, Project, Tools, Extensions, Window, Help, and a Search bar. The title bar says 'Solution1'. The left sidebar is 'Object Explorer' showing a connection to 'localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - DERE0r5)'. The main area has a 'prepared\_query...REverts (68)' tab, a 'populate\_data...Evorts (69)' tab (which is active), and a 'create\_databases...RE\orits (55)' tab. The 'populate\_data...Evorts (69)' tab contains the following T-SQL script:

```
386 UNION ALL SELECT 'EmployeeLocation', COUNT(*) FROM dbo.EmployeeLocation;
387
388 PRINT '===== RELATIONSHIP CHECKS =====';
389
390 SELECT c.FirstName + ' ' + c.LastName AS Customer, e.FirstName + ' ' + e.LastName AS Banker
391 FROM dbo.CustomerAssignedStaff cas
392 JOIN dbo.Customer c ON cas.CustomerID = c.CustomerID
393 JOIN dbo.Employee e ON cas.EmployeeID = e.EmployeeID
394 WHERE cas.StaffRole = 'PersonalBanker'
395 ORDER BY c.CustomerID;
396
397 PRINT '===== VERIFICATION COMPLETE =====';
398
399 SELECT * FROM dbo.Branch;
400
```

The bottom pane shows a 'Results' grid with the following data:

BranchID	BranchName	City	TotalDeposits	TotalLoans	CreatedAt
1	Downtown HQ	Calgary	2600000.00	1800000.00	2025-10-29 12:42:37.1621088
2	North Hill	Calgary	1400000.00	900000.00	2025-10-28 12:42:37.1521088
3	Riveland	Edmonton	1200000.00	1100000.00	2025-10-28 12:42:37.1521088

The status bar at the bottom indicates 'Query executed successfully.' and shows connection details: 'localhost\SQLEXPRESS01 (16... DERE0r5 (69) SKS\_National\_Bank 00:00:00 3 rows'.

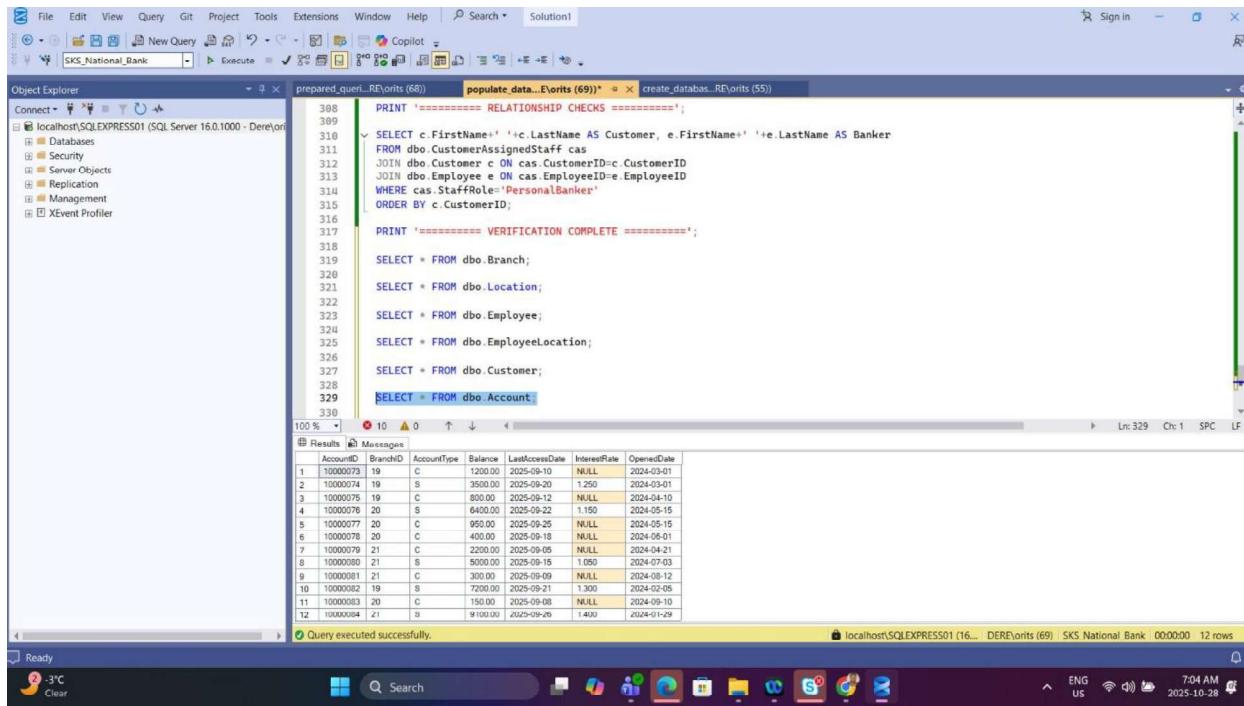
14) populate\_database.sql TestLocation

## 15) populate\_database.sql TestEmployee

16) populate\_database.sql EmployeeLocation

17) populate\_database.sql Customer

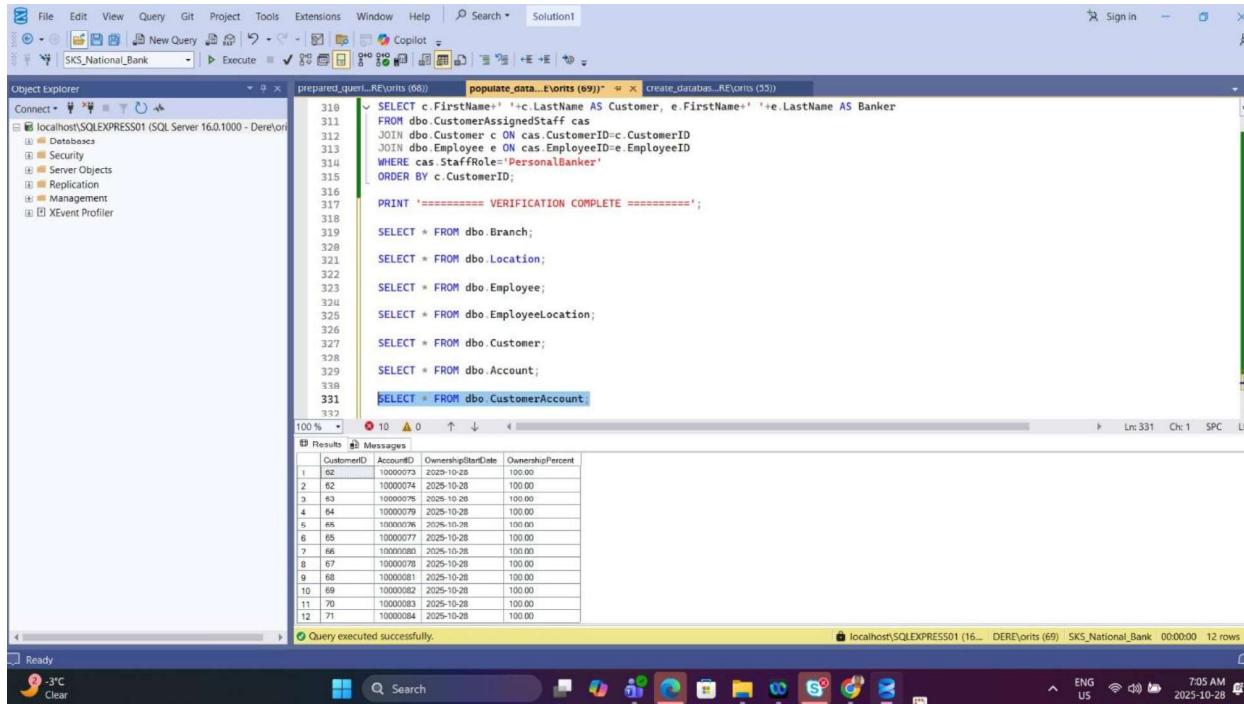
## 18) populate\_database.sql Account



The screenshot shows the SSMS interface with the 'Object Explorer' on the left and a 'Results' tab on the right. The results grid displays 12 rows of account data. The status bar at the bottom indicates the query was executed successfully.

AccountID	BranchID	AccountType	Balance	LastAccessDate	InterestRate	OpenedDate
1	10000073	19	C	1200.00	2025-09-10	NULL
2	10000074	19	S	3500.00	2025-09-20	1.250
3	10000075	19	C	800.00	2025-09-12	NULL
4	10000076	20	S	6400.00	2025-09-22	1.150
5	10000077	20	C	950.00	2025-09-25	NULL
6	10000078	20	C	400.00	2025-09-18	NULL
7	10000079	21	C	2200.00	2025-09-05	NULL
8	10000080	21	S	5000.00	2025-09-15	1.050
9	10000081	21	C	300.00	2025-09-09	NULL
10	10000082	18	S	7200.00	2025-09-21	1.300
11	10000083	20	C	150.00	2025-09-08	NULL
12	10000084	21	S	9100.00	2025-09-26	1.400

## 19) populate\_database.sql CustomerAccount



The screenshot shows the SSMS interface with the 'Object Explorer' on the left and a 'Results' tab on the right. The results grid displays 12 rows of customer account data. The status bar at the bottom indicates the query was executed successfully.

CustomerID	AccountID	OwnershipStartDate	OwnershipPercent	
1	62	2029-10-28	100.00	
2	62	2025-10-28	100.00	
3	63	2025-10-28	100.00	
4	64	2025-10-28	100.00	
5	65	2025-10-28	100.00	
6	65	10000077	2025-10-28	100.00
7	66	10000078	2025-10-28	100.00
8	67	10000078	2025-10-28	100.00
9	68	10000081	2025-10-28	100.00
10	69	10000082	2025-10-28	100.00
11	70	10000083	2025-10-28	100.00
12	71	10000084	2025-10-28	100.00

## 20) populate\_database.sql CustomerLoan



File Edit View Query Git Project Tools Extensions Window Help Search Solution

Object Explorer

SKS\_National\_Bank

prepared\_query...RE[orits (68)] **populate\_data...RE[orits (69)]** create\_database...RE[orits (55)]

```
315 ORDER BY c.CustomerID;
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
```

PRINT '===== VERIFICATION COMPLETE =====';

```
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
```

```
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
```

```
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
```

SELECT \* FROM dbo.Branch;

SELECT \* FROM dbo.Location;

SELECT \* FROM dbo.Employee;

SELECT \* FROM dbo.EmployeeLocation;

SELECT \* FROM dbo.Customer;

SELECT \* FROM dbo.Account;

SELECT \* FROM dbo.CustomerAccount;

SELECT \* FROM dbo.Overdraft;

SELECT \* FROM dbo.Loan;

**SELECT \* FROM dbo.CustomerLoan;**

Result Messages

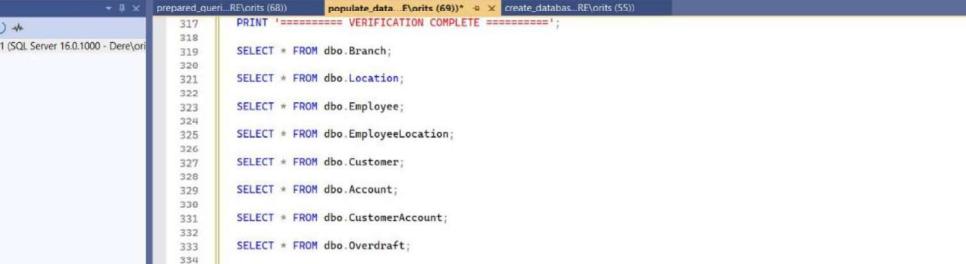
CustomerID	LoanID	Role
1	62	50000037
2	62	50000038
3	62	50000039
4	62	50000040
5	62	50000041
6	62	50000042
7	63	50000037
8	63	50000038
9	63	50000039
10	63	50000040
11	63	50000041

Query executed successfully.

localhost(SQL EXPRESSO01 (16... DERE\orits (69) SKS\_National\_Bank 00:00:00 12 rows

Ready 3°C Clear ENG US 7:09 AM 2025-10-28

## 21) populate\_database.sql LoanRepayment



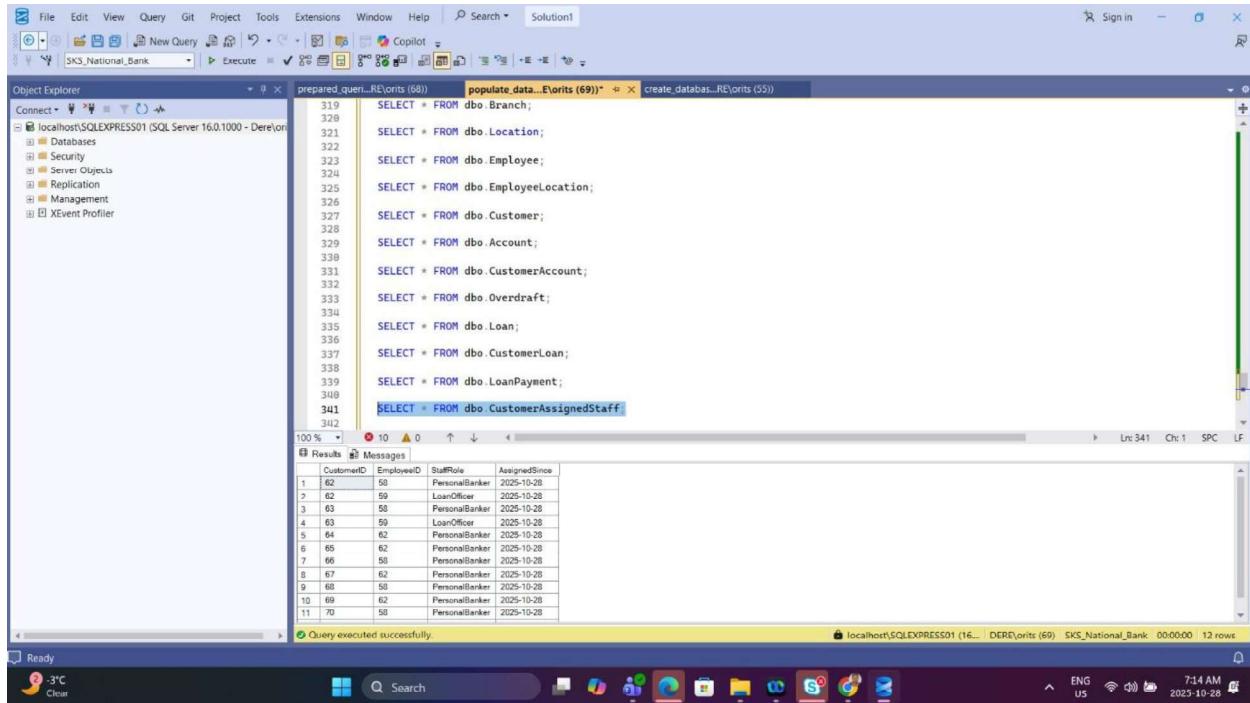
The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The Object Explorer on the left shows a connection to 'localhost\SQLEXPRESS01'. The 'populated\_data...' query window is active, displaying a series of SELECT statements from various tables in the 'dbo' schema. The results pane at the bottom shows a table with columns: LoanID, PaymentNo, PaymentDate, and PaymentAmount. A status bar at the bottom indicates 'Query executed successfully.' and the connection details 'localhost\SQLEXPRESS01 (16... DERE\orits (69) SKS\_National\_Bank 00:00:00 0 rows'.

```
317 PRINT ====== VERIFICATION COMPLETE ======;
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
```

LoanID	PaymentNo	PaymentDate	PaymentAmount

Query executed successfully.

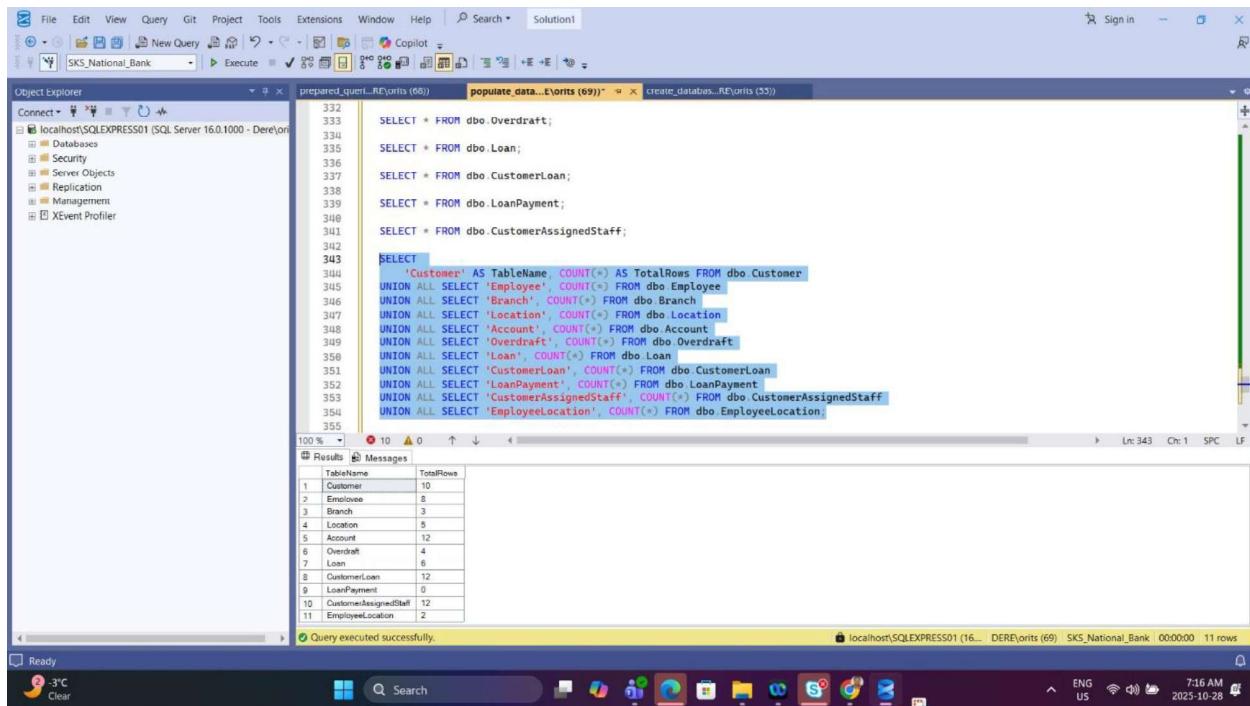
## 22) populate\_database.sql CustomerAssignedStaff



```
SELECT * FROM dbo.Branch;
SELECT * FROM dbo.Location;
SELECT * FROM dbo.Employee;
SELECT * FROM dbo.EmployeeLocation;
SELECT * FROM dbo.Customer;
SELECT * FROM dbo.Account;
SELECT * FROM dbo.CustomerAccount;
SELECT * FROM dbo.Overdraft;
SELECT * FROM dbo.Loan;
SELECT * FROM dbo.CustomerLoan;
SELECT * FROM dbo.LoanPayment;
SELECT * FROM dbo.CustomerAssignedStaff;
```

CustomerID	EmployeeID	StaffRole	AssignedSince
1	62	58	2025-10-28
2	62	59	2025-10-28
3	63	58	2025-10-28
4	63	59	2025-10-28
5	64	62	2025-10-28
6	65	62	2025-10-28
7	66	58	2025-10-28
8	67	62	2025-10-28
9	68	58	2025-10-28
10	69	62	2025-10-28
11	70	58	2025-10-28

## 23) populate\_database.sql Verification



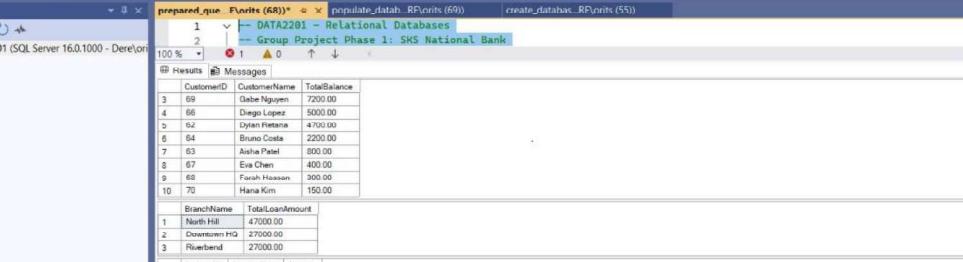
```
SELECT * FROM dbo.Overdraft;
SELECT * FROM dbo.Loan;
SELECT * FROM dbo.CustomerLoan;
SELECT * FROM dbo.LoanPayment;
SELECT * FROM dbo.CustomerAssignedStaff;

SELECT
    'Customer' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Customer
UNION ALL
SELECT
    'Employee' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Employee
UNION ALL
SELECT
    'Branch' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Branch
UNION ALL
SELECT
    'Location' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Location
UNION ALL
SELECT
    'Account' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Account
UNION ALL
SELECT
    'Overdraft' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Overdraft
UNION ALL
SELECT
    'Loan' AS TableName, COUNT(*) AS TotalRows
FROM dbo.Loan
UNION ALL
SELECT
    'CustomerLoan' AS TableName, COUNT(*) AS TotalRows
FROM dbo.CustomerLoan
UNION ALL
SELECT
    'LoanPayment' AS TableName, COUNT(*) AS TotalRows
FROM dbo.LoanPayment
UNION ALL
SELECT
    'CustomerAssignedStaff' AS TableName, COUNT(*) AS TotalRows
FROM dbo.CustomerAssignedStaff
UNION ALL
SELECT
    'EmployeeLocation' AS TableName, COUNT(*) AS TotalRows
FROM dbo.EmployeeLocation;
```

TableName	TotalRows
Customer	10
Employee	8
Branch	3
Location	5
Account	12
Overdraft	4
Loan	6
CustomerLoan	12
LoanPayment	0
CustomerAssignedStaff	12
EmployeeLocation	2

## prepare\_queries\_database

## 1) prepeare\_queries.sql



File Edit View Query Git Project Tools Extensions Window Help Search Solution

SKS\_National\_Bank Execute Copilot

Object Explorer

Connect ×

localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - DERE) ✓

Databases Security Server Objects Replication Management XEvent Profiler

prepared\_qui Events (68)\* × populate\_datah. RF(crits (69)) create\_databas. RF(crits (55))

1 DATA2201 - Relational Databases -- Group Project Phase 1: SKS National Bank

100% 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100

Results Messages

CustomerID	CustomerName	TotalBalance
3	Gabe Nguyen	7200.00
4	Diego Lopez	5000.00
5	Dylan Retana	4700.00
6	Bruno Costa	2200.00
7	Aisha Patel	800.00
8	Eva Chen	400.00
9	Farah Hussain	300.00
10	Hana Kim	150.00

BranchName	TotalLoanAmount	
1	North Hill	47000.00
2	Downtown HQ	27000.00
3	Riverbend	27000.00

AccountID	AccountType	Owners
1	2023-09-01	75.00 100201

AccountID	Balance	InterestRate	ProjectedOneYearInterest	
1	10000074	3500.00	12.50	43.75
2	10000076	6400.00	1.150	73.60
3	10000089	5000.00	1.050	52.50
4	10000095	7200.00	1.300	93.60
5	10000099	9100.00	1.400	127.40

BranchName	City	TotalDeposits	TotalLoans	CalculatedLoanSum
1	Downtown HQ	2500000.00	1800000.00	27000.00
2	North Hill	1400000.00	900000.00	47000.00
3	Riverbend	1200000.00	1100000.00	27000.00

EmployeeID	EmployeeName	Role	LocationCount
1	63	Aisha Patel	PersonalBanker

LoanID	PaymentId	PaymentDate	PaymentAmount	CumulativePaid
1	CustomerID	CustomerName	StatRole	

Query executed successfully.

localhost\SQLEXPRESS01 (16... DERE) (68) SKS\_National\_Bank 00:00:00 29 rows

Ready 3°C ENG 7:31 AM 2025-10-28

2) sp\_CustomerTotalBalance



The screenshot shows the Microsoft SQL Server Management Studio (SSMS) interface. The top menu bar includes File, Edit, View, Query, Git, Project, Tools, Extensions, Window, Help, and a Search bar. The title bar shows the connection to 'localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - DERE\orits)' and the database 'SKS\_National\_Bank'. The status bar at the bottom right shows the time as '7:54 AM' and the date as '2025-10-20'.

The Object Explorer on the left shows the database structure, including 'localhost\SQLEXPRESS01 (SQL Server 16.0.1000 - DERE\orits)' with its databases, security, server objects, replication, management, and XEvent Profiler.

The main window displays a query editor with the following script content:

```
232
233
234
235
236
237
238
239
240
241
242
243
244
245

ORDER BY b.TotalDeposits DESC, b.BranchName;
END;
GO
EXEC dbo.sp_TopBranchesByDeposits @TopN = 2;
GO

PRINT 'prepared_queries.sql executed successfully.';
GO

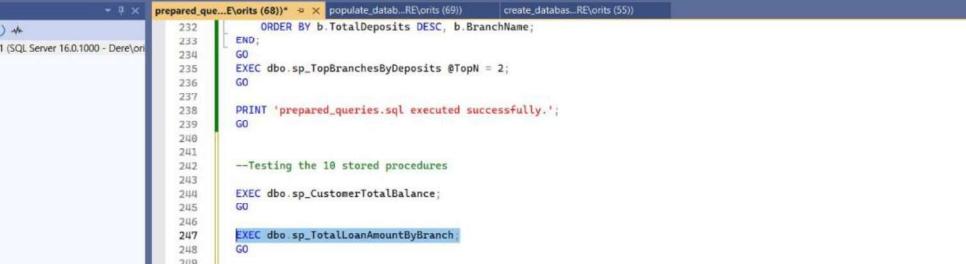
--Testing the 10 stored procedures
EXEC dbo.sp_CustomerTotalBalance
```

The results grid below the script shows the output of the 'EXEC dbo.sp\_CustomerTotalBalance' command:

CustomerID	CustomerName	TotalBalance	
1	71	Ivan Kovacs	9100.00
2	65	Chloe Martin	7350.00
3	69	Galen Nguyen	7200.00
4	66	Diego Lopez	5000.00
5	62	Dylan Retana	4700.00
6	64	Bruno Costa	2200.00
7	63	Lucas	1800.00
8	67	Eva Chen	1400.00
9	68	Faith Hassan	1200.00
10	70	Hania Kim	1500.00

The status bar at the bottom indicates 'Query executed successfully.' and shows the connection details: 'localhost\SQLEXPRESS01 (16... DERE\orits (68) SKS\_National\_Bank 00:00:00 10 rows'.

3) sp\_TotalLoanAmountByBranch



File Edit View Query Git Project Tools Extensions Window Help Search Solution!

SKS\_National\_Bank

Object Explorer

prepared\_query\_Evorits (68)\* | populate\_database\_Evorits (69) | create\_database\_Evorits (55)

```
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247 EXEC dbo.sp_TotalLoanAmountByBranch;
248
249

--Testing the 10 stored procedures

EXEC dbo.sp_CustomerTotalBalance;
GO

EXEC dbo.sp_TotalLoanAmountByBranch;
GO
```

100% 0 1 0 ↑ ↓

Results Messages

BranchName	TotalLoanAmount
North Hill	47000.00
Downtown HQ	27000.00
Riverbank	27000.00

Query executed successfully.

localhost\SQLEXPRESS01 (16... DERE\evorits (68) SKS\_National\_Bank 00:00:00 3 rows

Ready

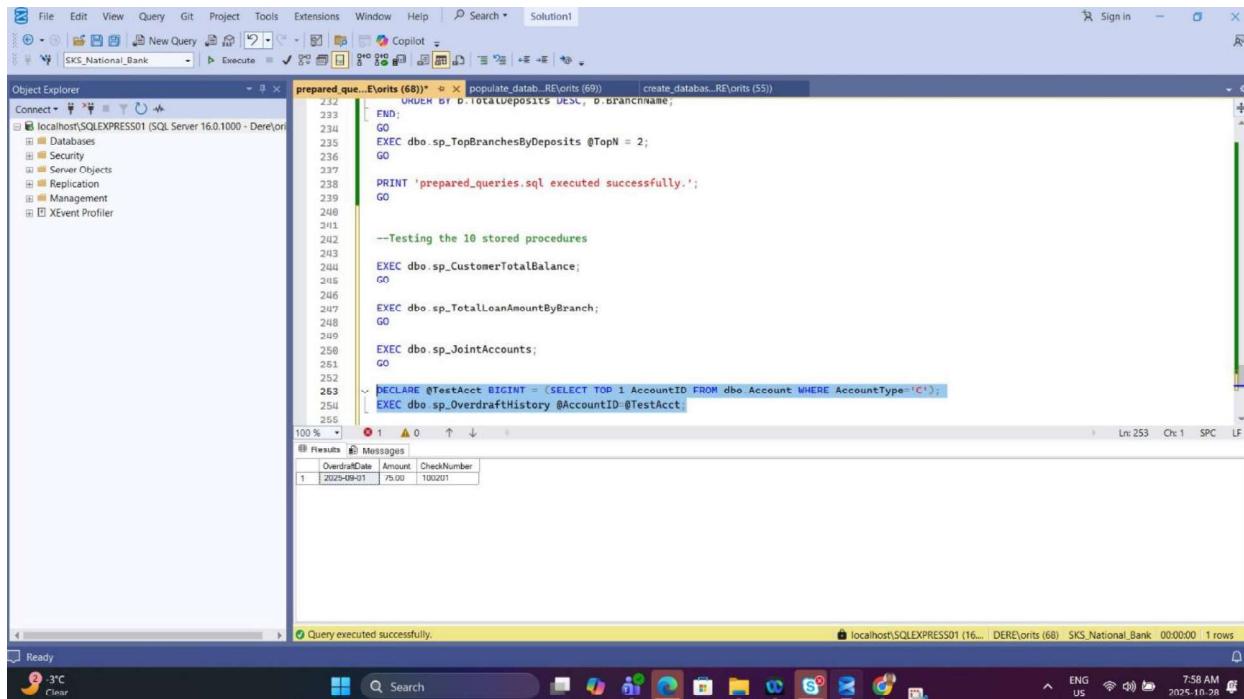
3°C Clear

ENG US

7:55 AM 2023-10-28

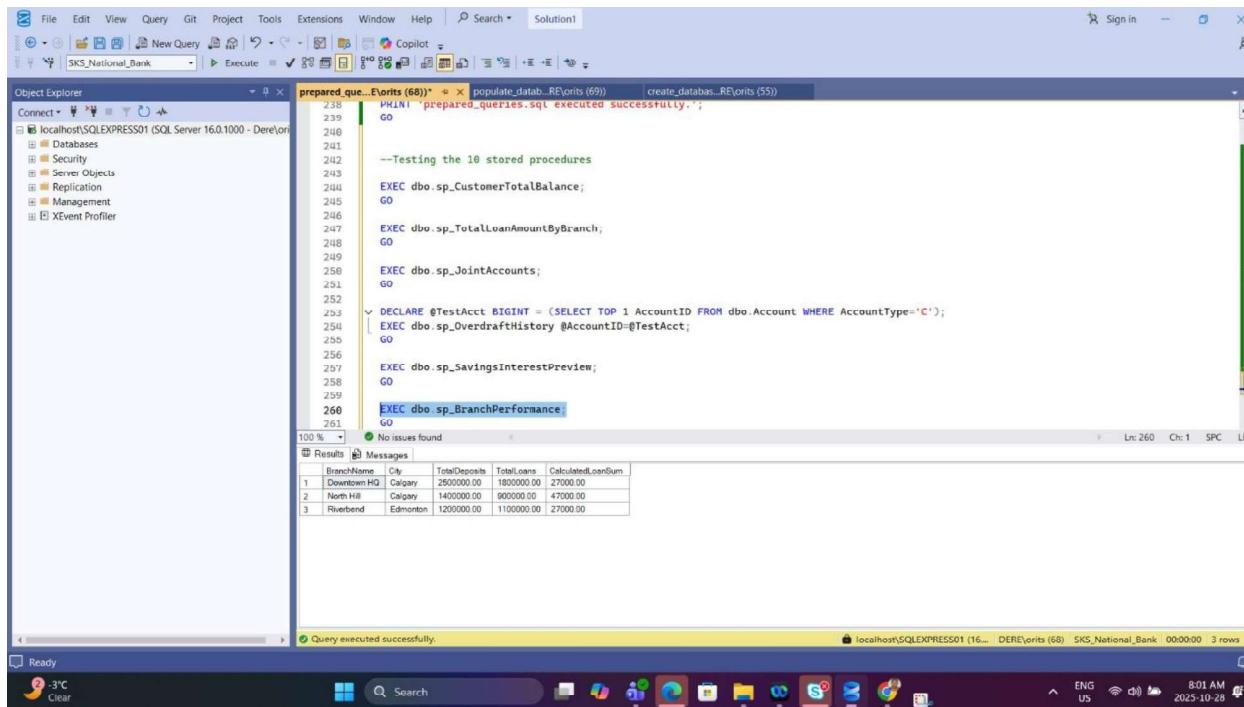
#### 4) sp\_JointAccounts

## 5) sp\_OverdraftHistory



```
prepared_qu...Events (68)* |(populate_database.REEvents (69)) |create_database.REEvents (55))  
232  
233  
234  
235  
236  
237  
238  
239  
240  
241  
242  
243  
244  
245  
246  
247  
248  
249  
250  
251  
252  
253  
254  
255  
256  
257  
258  
259  
260  
261  
262  
263  
264  
265  
266  
267  
268  
269  
270  
271  
272  
273  
274  
275  
276  
277  
278  
279  
280  
281  
282  
283  
284  
285  
286  
287  
288  
289  
290  
291  
292  
293  
294  
295  
296  
297  
298  
299  
300  
301  
302  
303  
304  
305  
306  
307  
308  
309  
310  
311  
312  
313  
314  
315  
316  
317  
318  
319  
320  
321  
322  
323  
324  
325  
326  
327  
328  
329  
330  
331  
332  
333  
334  
335  
336  
337  
338  
339  
340  
341  
342  
343  
344  
345  
346  
347  
348  
349  
350  
351  
352  
353  
354  
355  
356  
357  
358  
359  
360  
361  
362  
363  
364  
365  
366  
367  
368  
369  
370  
371  
372  
373  
374  
375  
376  
377  
378  
379  
380  
381  
382  
383  
384  
385  
386  
387  
388  
389  
390  
391  
392  
393  
394  
395  
396  
397  
398  
399  
400  
401  
402  
403  
404  
405  
406  
407  
408  
409  
410  
411  
412  
413  
414  
415  
416  
417  
418  
419  
420  
421  
422  
423  
424  
425  
426  
427  
428  
429  
430  
431  
432  
433  
434  
435  
436  
437  
438  
439  
440  
441  
442  
443  
444  
445  
446  
447  
448  
449  
450  
451  
452  
453  
454  
455  
456  
457  
458  
459  
460  
461  
462  
463  
464  
465  
466  
467  
468  
469  
470  
471  
472  
473  
474  
475  
476  
477  
478  
479  
480  
481  
482  
483  
484  
485  
486  
487  
488  
489  
490  
491  
492  
493  
494  
495  
496  
497  
498  
499  
500  
501  
502  
503  
504  
505  
506  
507  
508  
509  
510  
511  
512  
513  
514  
515  
516  
517  
518  
519  
520  
521  
522  
523  
524  
525  
526  
527  
528  
529  
530  
531  
532  
533  
534  
535  
536  
537  
538  
539  
540  
541  
542  
543  
544  
545  
546  
547  
548  
549  
550  
551  
552  
553  
554  
555  
556  
557  
558  
559  
559  
560  
561  
562  
563  
564  
565  
566  
567  
568  
569  
570  
571  
572  
573  
574  
575  
576  
577  
578  
579  
579  
580  
581  
582  
583  
584  
585  
586  
587  
588  
589  
589  
590  
591  
592  
593  
594  
595  
596  
597  
598  
599  
599  
600  
601  
602  
603  
604  
605  
606  
607  
608  
609  
609  
610  
611  
612  
613  
614  
615  
616  
617  
618  
619  
619  
620  
621  
622  
623  
624  
625  
626  
627  
628  
629  
629  
630  
631  
632  
633  
634  
635  
636  
637  
638  
639  
639  
640  
641  
642  
643  
644  
645  
646  
647  
648  
649  
649  
650  
651  
652  
653  
654  
655  
656  
657  
658  
659  
659  
660  
661  
662  
663  
664  
665  
666  
667  
668  
669  
669  
670  
671  
672  
673  
674  
675  
676  
677  
678  
679  
679  
680  
681  
682  
683  
684  
685  
686  
687  
688  
689  
689  
690  
691  
692  
693  
694  
695  
696  
697  
698  
699  
699  
700  
701  
702  
703  
704  
705  
706  
707  
708  
709  
709  
710  
711  
712  
713  
714  
715  
716  
717  
718  
719  
719  
720  
721  
722  
723  
724  
725  
726  
727  
728  
729  
729  
730  
731  
732  
733  
734  
735  
736  
737  
738  
739  
739  
740  
741  
742  
743  
744  
745  
746  
747  
748  
749  
749  
750  
751  
752  
753  
754  
755  
756  
757  
758  
759  
759  
760  
761  
762  
763  
764  
765  
766  
767  
768  
769  
769  
770  
771  
772  
773  
774  
775  
776  
777  
778  
779  
779  
780  
781  
782  
783  
784  
785  
786  
787  
788  
789  
789  
790  
791  
792  
793  
794  
795  
796  
797  
798  
799  
799  
800  
801  
802  
803  
804  
805  
806  
807  
808  
809  
809  
810  
811  
812  
813  
814  
815  
816  
817  
818  
819  
819  
820  
821  
822  
823  
824  
825  
826  
827  
828  
829  
829  
830  
831  
832  
833  
834  
835  
836  
837  
838  
839  
839  
840  
841  
842  
843  
844  
845  
846  
847  
848  
849  
849  
850  
851  
852  
853  
854  
855  
856  
857  
858  
859  
859  
860  
861  
862  
863  
864  
865  
866  
867  
868  
869  
869  
870  
871  
872  
873  
874  
875  
876  
877  
878  
879  
879  
880  
881  
882  
883  
884  
885  
886  
887  
888  
889  
889  
890  
891  
892  
893  
894  
895  
896  
897  
898  
899  
899  
900  
901  
902  
903  
904  
905  
906  
907  
908  
909  
909  
910  
911  
912  
913  
914  
915  
916  
917  
918  
919  
919  
920  
921  
922  
923  
924  
925  
926  
927  
928  
929  
929  
930  
931  
932  
933  
934  
935  
936  
937  
938  
939  
939  
940  
941  
942  
943  
944  
945  
946  
947  
948  
949  
949  
950  
951  
952  
953  
954  
955  
956  
957  
958  
959  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
969  
970  
971  
972  
973  
974  
975  
976  
977  
978  
979  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
989  
990  
991  
992  
993  
994  
995  
996  
997  
998  
999  
999  
1000  
1001  
1002  
1003  
1004  
1005  
1006  
1007  
1008  
1009  
1009  
1010  
1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1019  
1020  
1021  
1022  
1023  
1024  
1025  
1026  
1027  
1028  
1029  
1029  
1030  
1031  
1032  
1033  
1034  
1035  
1036  
1037  
1038  
1039  
1039  
1040  
1041  
1042  
1043  
1044  
1045  
1046  
1047  
1048  
1049  
1049  
1050  
1051  
1052  
1053  
1054  
1055  
1056  
1057  
1058  
1059  
1059  
1060  
1061  
1062  
1063  
1064  
1065  
1066  
1067  
1068  
1069  
1069  
1070  
1071  
1072  
1073  
1074  
1075  
1076  
1077  
1078  
1079  
1079  
1080  
1081  
1082  
1083  
1084  
1085  
1086  
1087  
1088  
1089  
1089  
1090  
1091  
1092  
1093  
1094  
1095  
1096  
1097  
1098  
1098  
1099  
1099  
1100  
1101  
1102  
1103  
1104  
1105  
1106  
1107  
1108  
1109  
1109  
1110  
1111  
1112  
1113  
1114  
1115  
1116  
1117  
1118  
1119  
1119  
1120  
1121  
1122  
1123  
1124  
1125  
1126  
1127  
1128  
1129  
1129  
1130  
1131  
1132  
1133  
1134  
1135  
1136  
1137  
1138  
1139  
1139  
1140  
1141  
1142  
1143  
1144  
1145  
1146  
1147  
1148  
1149  
1149  
1150  
1151  
1152  
1153  
1154  
1155  
1156  
1157  
1158  
1159  
1159  
1160  
1161  
1162  
1163  
1164  
1165  
1166  
1167  
1168  
1169  
1169  
1170  
1171  
1172  
1173  
1174  
1175  
1176  
1177  
1178  
1179  
1179  
1180  
1181  
1182  
1183  
1184  
1185  
1186  
1187  
1188  
1189  
1189  
1190  
1191  
1192  
1193  
1194  
1195  
1196  
1197  
1198  
1198  
1199  
1199  
1200  
1201  
1202  
1203  
1204  
1205  
1206  
1207  
1208  
1209  
1209  
1210  
1211  
1212  
1213  
1214  
1215  
1216  
1217  
1218  
1219  
1219  
1220  
1221  
1222  
1223  
1224  
1225  
1226  
1227  
1228  
1229  
1229  
1230  
1231  
1232  
1233  
1234  
1235  
1236  
1237  
1238  
1239  
1239  
1240  
1241  
1242  
1243  
1244  
1245  
1246  
1247  
1248  
1249  
1249  
1250  
1251  
1252  
1253  
1254  
1255  
1256  
1257  
1258  
1259  
1259  
1260  
1261  
1262  
1263  
1264  
1265  
1266  
1267  
1268  
1269  
1269  
1270  
1271  
1272  
1273  
1274  
1275  
1276  
1277  
1278  
1279  
1279  
1280  
1281  
1282  
1283  
1284  
1285  
1286  
1287  
1288  
1289  
1289  
1290  
1291  
1292  
1293  
1294  
1295  
1296  
1297  
1298  
1298  
1299  
1299  
1300  
1301  
1302  
1303  
1304  
1305  
1306  
1307  
1308  
1309  
1309  
1310  
1311  
1312  
1313  
1314  
1315  
1316  
1317  
1318  
1319  
1319  
1320  
1321  
1322  
1323  
1324  
1325  
1326  
1327  
1328  
1329  
1329  
1330  
1331  
1332  
1333  
1334  
1335  
1336  
1337  
1338  
1339  
1339  
1340  
1341  
1342  
1343  
1344  
1345  
1346  
1347  
1348  
1349  
1349  
1350  
1351  
1352  
1353  
1354  
1355  
1356  
1357  
1358  
1359  
1359  
1360  
1361  
1362  
1363  
1364  
1365  
1366  
1367  
1368  
1369  
1369  
1370  
1371  
1372  
1373  
1374  
1375  
1376  
1377  
1378  
1379  
1379  
1380  
1381  
1382  
1383  
1384  
1385  
1386  
1387  
1388  
1389  
1389  
1390  
1391  
1392  
1393  
1394  
1395  
1396  
1397  
1398  
1398  
1399  
1399  
1400  
1401  
1402  
1403  
1404  
1405  
1406  
1407  
1408  
1409  
1409  
1410  
1411  
1412  
1413  
1414  
1415  
1416  
1417  
1418  
1419  
1419  
1420  
1421  
1422  
1423  
1424  
1425  
1426  
1427  
1428  
1429  
1429  
1430  
1431  
1432  
1433  
1434  
1435  
1436  
1437  
1438  
1439  
1439  
1440  
1441  
1442  
1443  
1444  
1445  
1446  
1447  
1448  
1449  
1449  
1450  
1451  
1452  
1453  
1454  
1455  
1456  
1457  
1458  
1459  
1459  
1460  
1461  
1462  
1463  
1464  
1465  
1466  
1467  
1468  
1469  
1469  
1470  
1471  
1472  
1473  
1474  
1475  
1476  
1477  
1478  
1479  
1479  
1480  
1481  
1482  
1483  
1484  
1485  
1486  
1487  
1488  
1489  
1489  
1490  
1491  
1492  
1493  
1494  
1495  
1496  
1497  
1498  
1498  
1499  
1499  
1500  
1501  
1502  
1503  
1504  
1505  
1506  
1507  
1508  
1509  
1509  
1510  
1511  
1512  
1513  
1514  
1515  
1516  
1517  
1518  
1519  
1519  
1520  
1521  
1522  
1523  
1524  
1525  
1526  
1527  
1528  
1529  
1529  
1530  
1531  
1532  
1533  
1534  
1535  
1536  
1537  
1538  
1539  
1539  
1540  
1541  
1542  
1543  
1544  
1545  
1546  
1547  
1548  
1549  
1549  
1550  
1551  
1552  
1553  
1554  
1555  
1556  
1557  
1558  
1559  
1559  
1560  
1561  
1562  
1563  
1564  
1565  
1566  
1567  
1568  
1569  
1569  
1570  
1571  
1572  
1573  
1574  
1575  
1576  
1577  
1578  
1579  
1579  
1580  
1581  
1582  
1583  
1584  
1585  
1586  
1587  
1588  
1589  
1589  
1590  
1591  
1592  
1593  
1594  
1595  
1596  
1597  
1598  
1598  
1599  
1599  
1600  
1601  
1602  
1603  
1604  
1605  
1606  
1607  
1608  
1609  
1609  
1610  
1611  
1612  
1613  
1614  
1615  
1616  
1617  
1618  
1619  
1619  
1620  
1621  
1622  
1623  
1624  
1625  
1626  
1627  
1628  
1629  
1629  
1630  
1631  
1632  
1633  
1634  
1635  
1636  
1637  
1638  
1639  
1639  
1640  
1641  
1642  
1643  
1644  
1645  
1646  
1647  
1648  
1649  
1649  
1650  
1651  
1652  
1653  
1654  
1655  
1656  
1657  
1658  
1659  
1659  
1660  
1661  
1662  
1663  
1664  
1665  
1666  
1667  
1668  
1669  
1669  
1670  
1671  
1672  
1673  
1674  
1675  
1676  
1677  
1678  
1679  
1679  
1680  
1681  
1682  
1683  
1684  
1685  
1686  
1687  
1688  
1689  
1689  
1690  
1691  
1692  
1693  
1694  
1695  
1696  
1697  
1698  
1698  
1699  
1699  
1700  
1701  
1702  
1703  
1704  
1705  
1706  
1707  
1708  
1709  
1709  
1710  
1711  
1712  
1713  
1714  
1715  
1716  
1717  
1718  
1719  
1719  
1720  
1721  
1722  
1723  
1724  
1725  
1726  
1727  
1728  
1729  
1729  
1730  
1731  
1732  
1733  
1734  
1735  
1736  
1737  
1738  
1739  
1739  
1740  
1741  
1742  
1743  
1744  
1745  
1746  
1747  
1748  
1749  
1749  
1750  
1751  
1752  
1753  
1754  
1755  
1756  
1757  
1758  
1759  
1759  
1760  
1761  
1762  
1763  
1764  
1765  
1766  
1767  
1768  
1769  
1769  
1770  
1771  
1772  
1773  
1774  
1775  
1776  
1777  
1778  
1779  
1779  
1780  
1781  
1782  
1783  
1784  
1785  
1786  
1787  
1788  
1789  
1789  
1790  
1791  
1792  
1793  
1794  
1795  
1796  
1797  
1798  
1798  
1799  
1799  
1800  
1801  
1802  
1803  
1804  
1805  
1806  
1807  
1808  
1809  
1809  
1810  
1811  
1812  
1813  
1814  
1815  
1816  
1817  
1818  
1819  
1819  
1820  
1821  
1822  
1823  
1824  
1825  
1826  
1827  
1828  
1829  
1829  
1830  
1831  
1832  
1833  
1834  
1835  
1836  
1837  
1838  
1839  
1839  
1840  
1841  
1842  
1843  
1844  
1845  
1846  
1847  
1848  
1849  
1849  
1850  
1851  
1852  
1853  
1854  
1855  
1856  
1857  
1858  
1859  
1859  
1860  
1861  
1862  
1863  
1864  
1865  
1866  
1867  
1868  
1869  
1869  
1870  
1871  
1872  
1873  
1874  
1875  
1876  
1877  
1878  
1879  
1879  
1880  
1881  
1882  
1883  
1884  
1885  
1886  
1887  
1888  
1889  
1889  
1890  
1891  
1892  
1893  
1894  
1895  
1896  
1897  
1898  
1898  
1899  
1899  
1900  
1901  
1902  
1903  
1904  
1905  
1906  
1907  
1908  
1909  
1909  
1910  
1911  
1912  
1913  
1914  
1915  
1916  
1917  
1918  
1919  
1919  
1920  
1921  
1922  
1923  
1924  
1925  
1926  
1927  
1928  
1929  
1929  
1930  
1931  
1932  
1933  
1934  
1935  
1936  
1937  
1938  
1939  
1939  
1940  
1941  
1942  
1943  
1944  
1945  
1946  
1947  
1948  
1949  
1949  
1950  
1951  
1952  
1953  
1954  
1955  
1956  
1957  
1958  
1959  
1959  
1960  
1961  
1962  
1963  
1964  
1965  
1966  
1967  
1968  
1969  
1969  
1970  
1971  
1972  
1973  
1974  
1975  
1976  
1977  
1978  
1979  
1979  
1980  
1981  
1982  
1983  
1984  
1985  
1986  
1987  
1988  
1989  
1989  
1990  
1991  
1992  
1993  
1994  
1995  
1996  
1997  
1998  
1998  
1999  
1999  
2000  
2001  
2002  
2003  
2004  
2005  
2006  
2007  
2008  
2009  
2010  
2011  
2012  
2013  
2014  
2015  
2016  
2017  
2018  
2019  
2020  
2021  
2022  
2023  
2024  
2025  
2026  
2027  
2028  
2029  
2030  
2031  
2032  
2033  
2034  
2035  
2036  
2037  
2038  
2039  
2039  
2040  
2041  
2042  
2043  
2044  
2045  
2046  
2047  
2048  
2049  
2049  
2050  
2051  
2052  
2053  
2054  
2055  
2056  
2057  
2058  
2059  
2059  
2060  
2061  
2062  
2063  
2064  
2065  
2066  
2067  
2068  
2069  
2069  
2070  
2071  
2072  
2073  
2074  
2075  
2076  
2077  
2078  
2079  
2
```

## 7) sp\_BranchPerformance



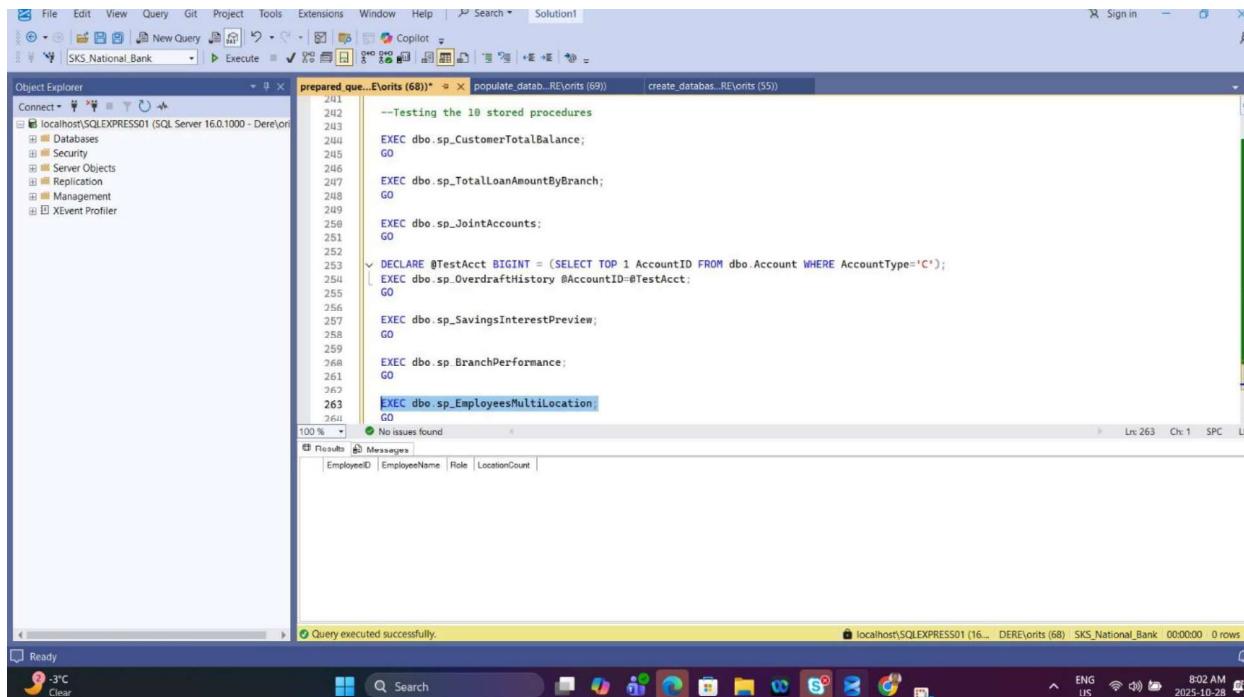
```
prepared que...Eorits (68)* + x populate_db...RE\orits (69) | create_databas...RE\orits (55)
238 PRINT 'populated_queries.sql executed successfully.';
239 GO
240
241 --Testing the 10 stored procedures
242
243 EXEC dbo.sp_CustomerTotalBalance;
244 GO
245
246 EXEC dbo.sp_TotalLoanAmountByBranch;
247 GO
248
249 EXEC dbo.sp_JointAccounts;
250 GO
251
252
253 DECLARE @TestAcct BIGINT = (SELECT TOP 1 AccountID FROM dbo.Account WHERE AccountType='C');
254 EXEC dbo.sp_OverdraftHistory @AccountID=@TestAcct;
255 GO
256
257 EXEC dbo.sp_SavingsInterestPreview;
258 GO
259
260 EXEC dbo.sp_BranchPerformance;
261 GO
```

Results

BranchName	City	TotalDeposits	TotalLoans	CalculatedComSum
Downtown HQ	Calgary	2500000.00	1800000.00	27000.00
North Hill	Calgary	1400000.00	900000.00	47000.00
Riverbend	Edmonton	1200000.00	1100000.00	27000.00

Query executed successfully.

## 8) sp\_EmployeesMultiLocation



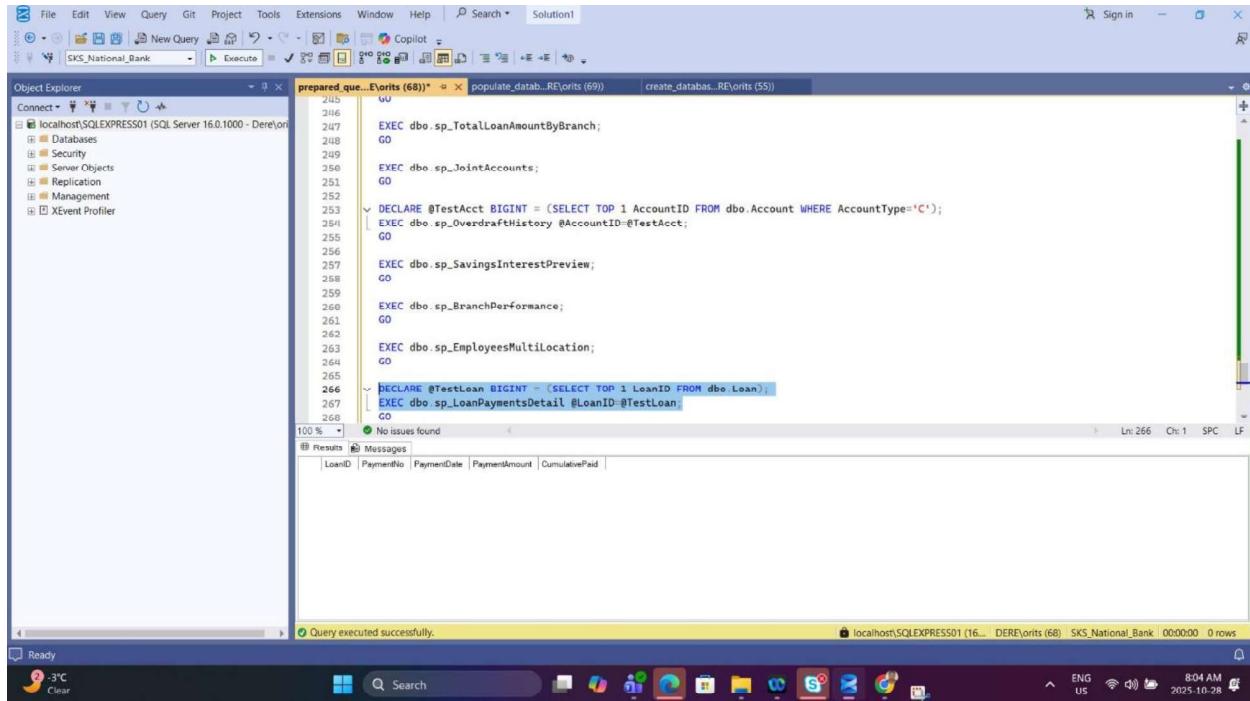
```
prepared que...Eorits (68)* + x populate_db...RE\orits (69) | create_databas...RE\orits (55)
201
202 --Testing the 10 stored procedures
203
204 EXEC dbo.sp_CustomerTotalBalance;
205 GO
206
207 EXEC dbo.sp_TotalLoanAmountByBranch;
208 GO
209
210 EXEC dbo.sp_JointAccounts;
211 GO
212
213
214 DECLARE @TestAcct BIGINT = (SELECT TOP 1 AccountID FROM dbo.Account WHERE AccountType='C');
215 EXEC dbo.sp_OverdraftHistory @AccountID=@TestAcct;
216 GO
217
218 EXEC dbo.sp_SavingsInterestPreview;
219 GO
220
221 EXEC dbo.sp_BranchPerformance;
222 GO
223
224 EXEC dbo.sp_EmployeesMultiLocation;
225 GO
```

Results

EmployeeID	EmployeeName	Role	LocationCount
1	John Doe	Manager	3
2	Jane Doe	Manager	3
3	Michael Jackson	Manager	3

Query executed successfully.

## 9) sp\_LoanPaymentsDetail



```
245
246 EXEC dbo.sp_TotalLoanAmountByBranch;
247 GO
248
249 EXEC dbo.sp_JointAccounts;
250 GO
251
252 DECLARE @TestAcct BIGINT = (SELECT TOP 1 AccountID FROM dbo.Account WHERE AccountType='C');
253 EXEC dbo.sp_OverdraftHistory @AccountID=@TestAcct;
254 GO
255
256 EXEC dbo.sp_SavingsInterestPreview;
257 GO
258
259 EXEC dbo.sp_BranchPerformance;
260 GO
261
262 EXEC dbo.sp_EmployeesMultiLocation;
263 GO
264
265 DECLARE @TestLoan BIGINT = (SELECT TOP 1 LoanID FROM dbo.Loan);
266 EXEC dbo.sp_LoanPaymentsDetail @LoanID=@TestLoan;
267 GO
268
```

100 % No issues found

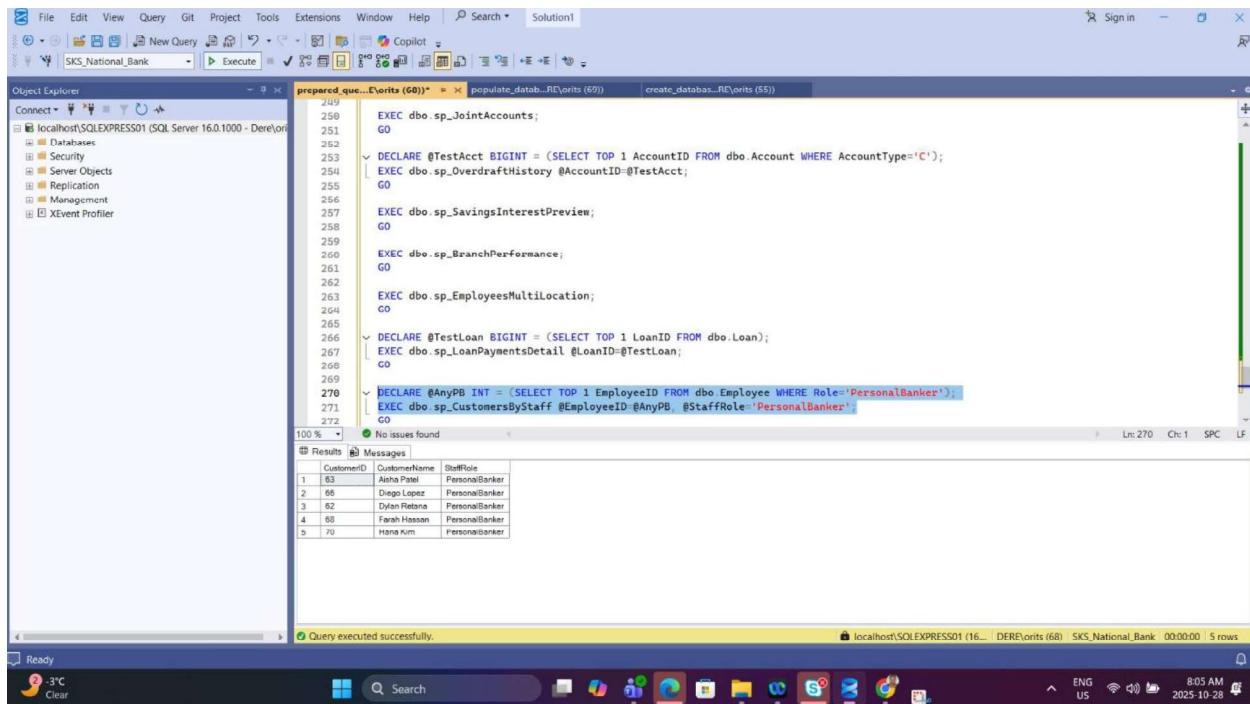
Results Messages

LoanID	PaymentNo	PaymentDate	PaymentAmount	CumulativePaid
--------	-----------	-------------	---------------	----------------

Query executed successfully.

localhost\SQLEXPRESS01 (16... | DERE\orits (68) | SKS\_National\_Bank | 00:00:00 | 0 rows

## 10) sp\_CustomersByStaff



```
249
250 EXEC dbo.sp_JointAccounts;
251 GO
252
253 DECLARE @TestAcct BIGINT = (SELECT TOP 1 AccountID FROM dbo.Account WHERE AccountType='C');
254 EXEC dbo.sp_OverdraftHistory @AccountID=@TestAcct;
255 GO
256
257 EXEC dbo.sp_SavingsInterestPreview;
258 GO
259
260 EXEC dbo.sp_BranchPerformance;
261 GO
262
263 EXEC dbo.sp_EmployeesMultiLocation;
264 GO
265
266 DECLARE @TestLoan BIGINT = (SELECT TOP 1 LoanID FROM dbo.Loan);
267 EXEC dbo.sp_LoanPaymentsDetail @LoanID=@TestLoan;
268 GO
269
270 DECLARE @AnyPB INT = (SELECT TOP 1 EmployeeID FROM dbo.Employee WHERE Role='PersonalBanker');
271 EXEC dbo.sp_CustomersByStaff @EmployeeID=@AnyPB, @StaffRole='PersonalBanker';
272 GO
273
```

100 % No issues found

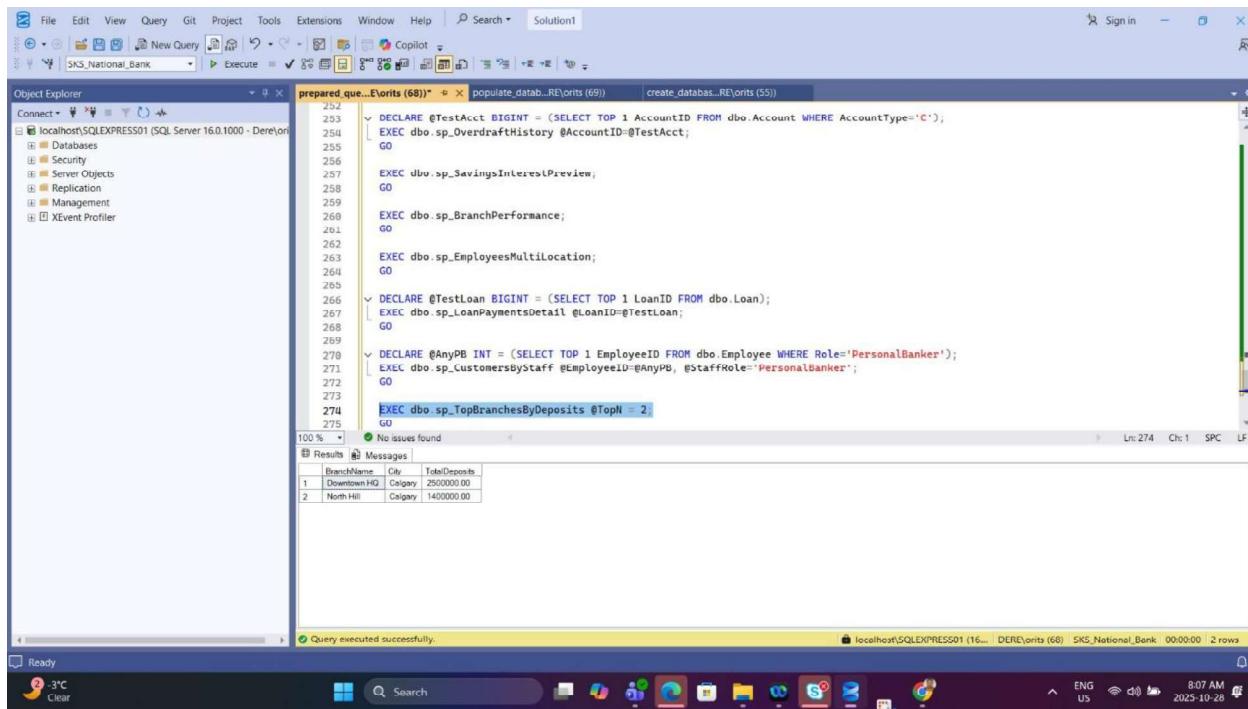
Results Messages

CustomerID	CustomerName	StaffRole
1	Asha Patel	PersonalBanker
2	Diego Lopez	PersonalBanker
3	Dylan Retana	PersonalBanker
4	Farah Hassan	PersonalBanker
5	Henna Kurn	PersonalBanker

Query executed successfully.

localhost\SQLEXPRESS01 (16... | DERE\orits (68) | SKS\_National\_Bank | 00:00:00 | 5 rows

## 11) sp\_TopBranchesByDeposits



```
253     v DECLARE @TestAcct BIGINT = (SELECT TOP 1 AccountID FROM dbo.Account WHERE AccountType='C');
254     l EXEC dbo.sp_OverdraftHistory @AccountID=@TestAcct;
255     GO
256
257     EXEC dbo.sp_SavingsInterestPreview;
258     GO
259
260     EXEC dbo.sp_BranchPerformance;
261     GO
262
263     EXEC dbo.sp_EmployeesMultilocation;
264     GO
265
266     v DECLARE @TestLoan BIGINT = (SELECT TOP 1 LoanID FROM dbo.Loan);
267     l EXEC dbo.sp_LoanPaymentsDetail @LoanID=@TestLoan;
268     GO
269
270     v DECLARE @AnyPB INT = (SELECT TOP 1 EmployeeID FROM dbo.Employee WHERE Role='PersonalBanker');
271     l EXEC dbo.sp_CustomersByStaff @EmployeeID=@AnyPB, @StaffRole='PersonalBanker';
272     GO
273
274     EXEC dbo.sp_TopBranchesByDeposits @TopN = 2;
275     GO
```

100 % 0 No issues found

Results Messages

	BranchName	City	TotalDeposits
1	Downtown HQ	Calgary	2500000.00
2	North Hill	Calgary	1400000.00

Query executed successfully.

localhost[SQLEXPRESS01 (16...)] DERE\orits (60) SKS\_National\_Bank 00:00:00 2 rows

Ready 3°C Clear

ENG US 8:07 AM 2025-10-26