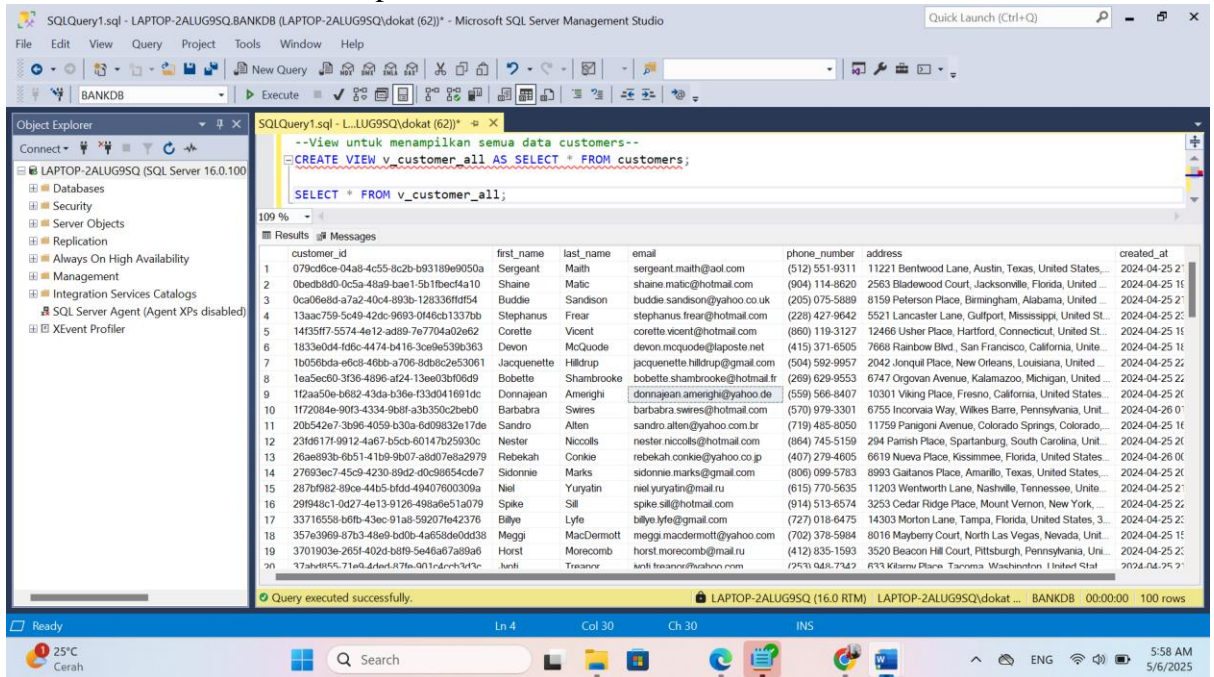


## Kelompok 2 MBD A2

- Atika Mutia Sabila (24060123130089)
- Ananda Rachmawati Purwanto (24060123130061)
- Aulia Raffa Satria (24060123140050)
- Bramantyo Kunni N (24060123130091)
- Muhammad Irfan Irsyad (24060123130085)

## Topik 1 Views

1. view v\_customer\_all menampilkan semua isi tabel customers



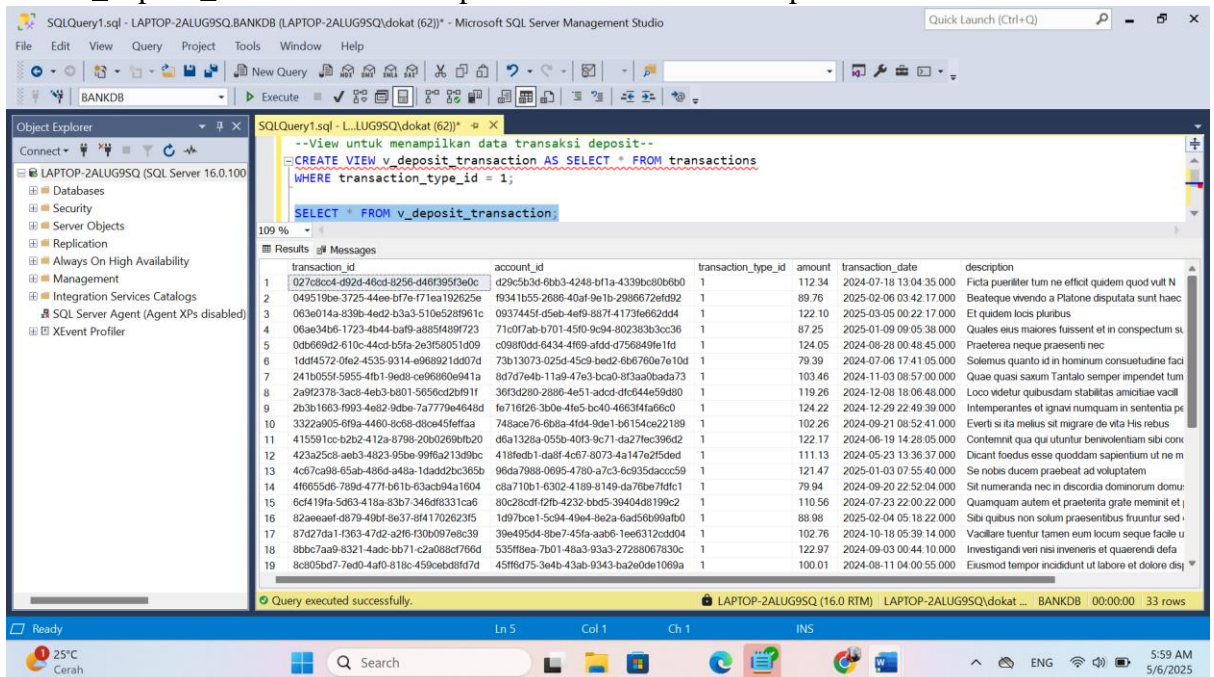
The screenshot shows the SQL Server Management Studio interface. The query editor contains the following SQL code:

```
--View untuk menampilkan semua data customers--
CREATE VIEW v_customer_all AS SELECT * FROM customers;

SELECT * FROM v_customer_all;
```

The query was executed successfully, returning 100 rows. The results are displayed in a table with the following columns: customer\_id, first\_name, last\_name, email, phone\_number, address, and created\_at. The data includes various customer records with their personal and contact information.

2. view v\_deposit\_transaction menampilkan semua transaksi deposit



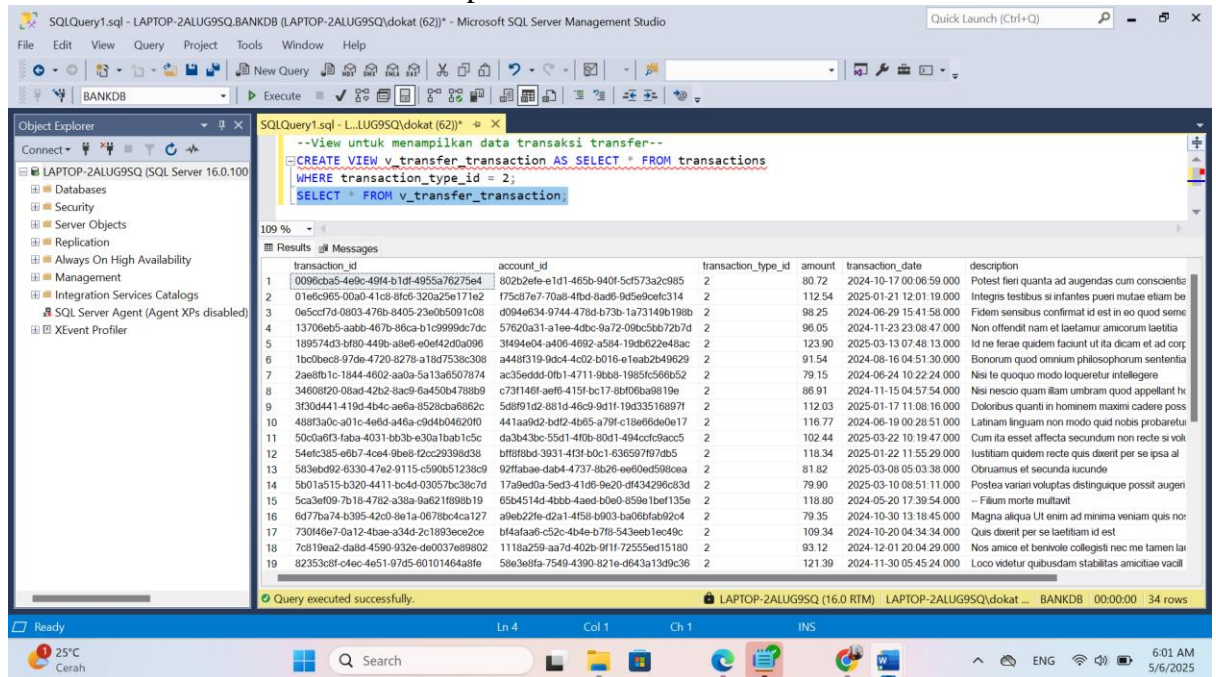
The screenshot shows the SQL Server Management Studio interface. The query editor contains the following SQL code:

```
--View untuk menampilkan data transaksi deposit--
CREATE VIEW v_deposit_transaction AS SELECT * FROM transactions
WHERE transaction_type_id = 1;

SELECT * FROM v_deposit_transaction;
```

The query was executed successfully, returning 33 rows. The results are displayed in a table with the following columns: transaction\_id, account\_id, transaction\_type\_id, amount, transaction\_date, and description. The data includes various deposit transactions with their amounts and dates.

### 3. view v\_transfer\_transaction menampilkan semua transaksi transfer.



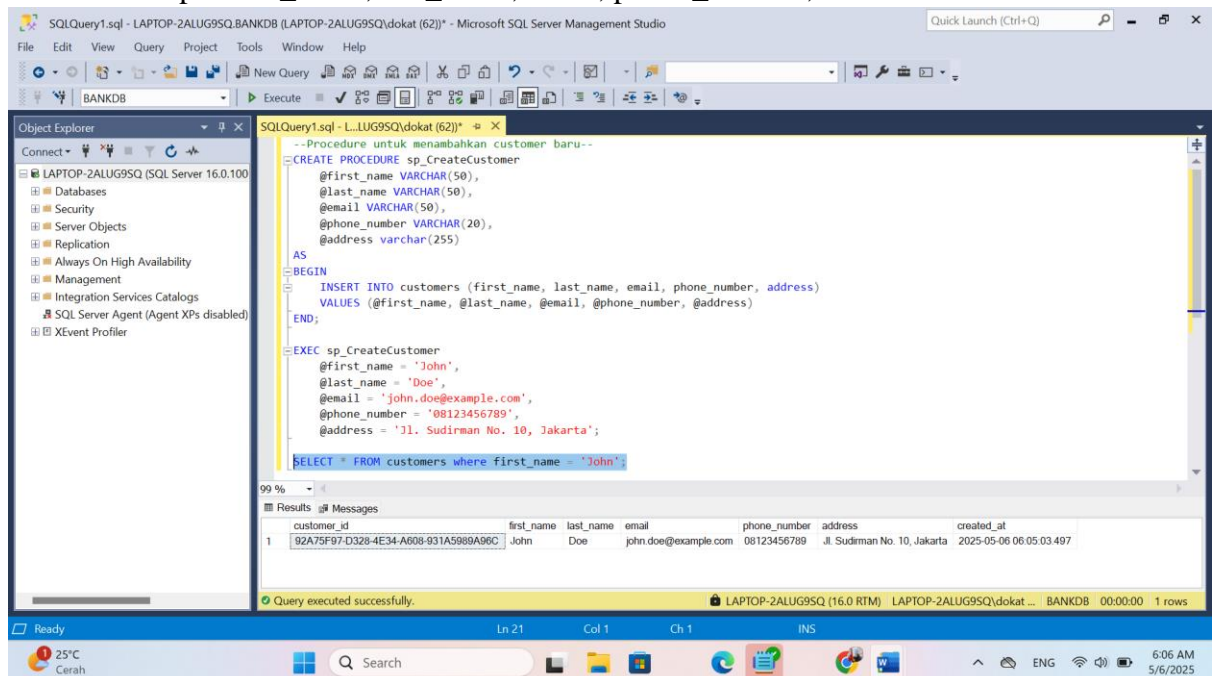
The screenshot shows the SQL Server Management Studio interface. The query editor contains the following SQL code:

```
--View untuk menampilkan data transaksi transfer--  
CREATE VIEW v_transfer_transaction AS SELECT * FROM transactions  
WHERE transaction_type_id = 2;  
SELECT * FROM v_transfer_transaction;
```

The Results pane shows the output of the query, displaying a list of transactions with columns: transaction\_id, account\_id, transaction\_type\_id, amount, transaction\_date, and description. The status bar indicates "Query executed successfully." and "34 rows".

## Topik 2 Procedure

### 1. Procedure sp\_CreateCustomer untuk menambahkan customer baru ke dalam tabel customers: input first\_name, last\_name, email, phone\_number, address.



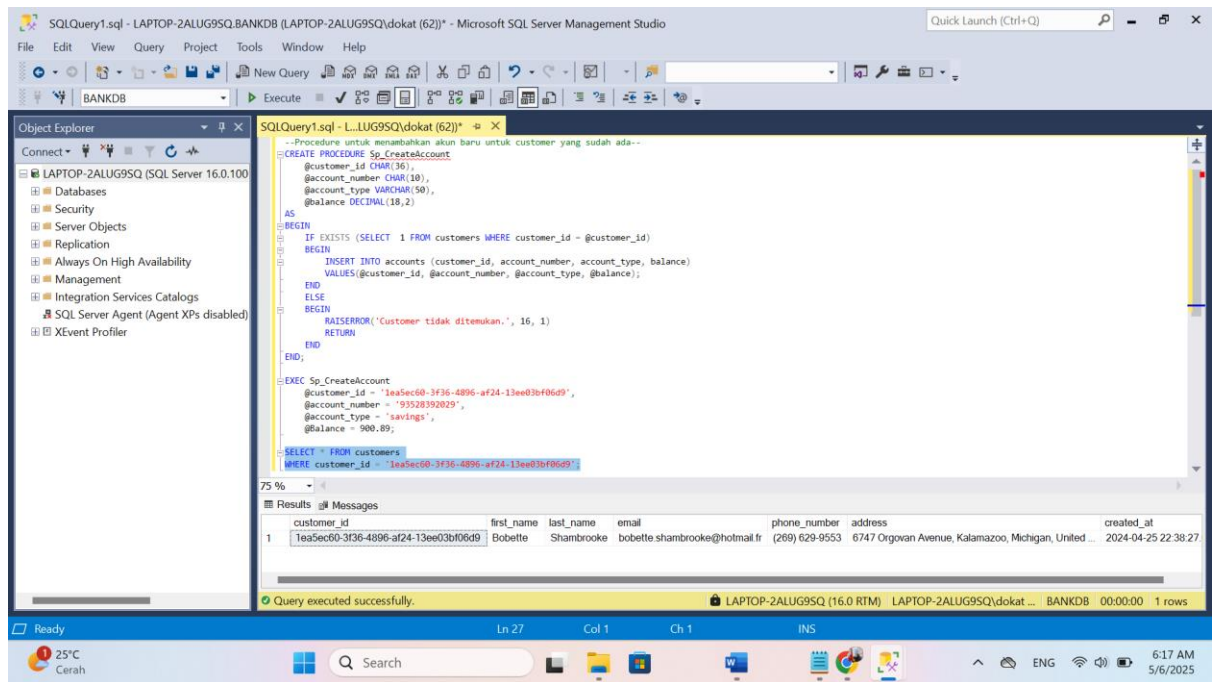
The screenshot shows the SQL Server Management Studio interface. The query editor contains the following SQL code:

```
--Procedure untuk menambahkan customer baru--  
CREATE PROCEDURE sp_CreateCustomer  
    @first_name VARCHAR(50),  
    @last_name VARCHAR(50),  
    @email VARCHAR(50),  
    @phone_number VARCHAR(20),  
    @address varchar(255)  
AS  
BEGIN  
    INSERT INTO customers (first_name, last_name, email, phone_number, address)  
    VALUES (@first_name, @last_name, @email, @phone_number, @address)  
END;  
  
EXEC sp_CreateCustomer  
    @first_name = 'John',  
    @last_name = 'Doe',  
    @email = 'john.doe@example.com',  
    @phone_number = '08123456789',  
    @address = 'Jl. Sudirman No. 10, Jakarta';  
  
SELECT * FROM customers where first_name = 'John';
```

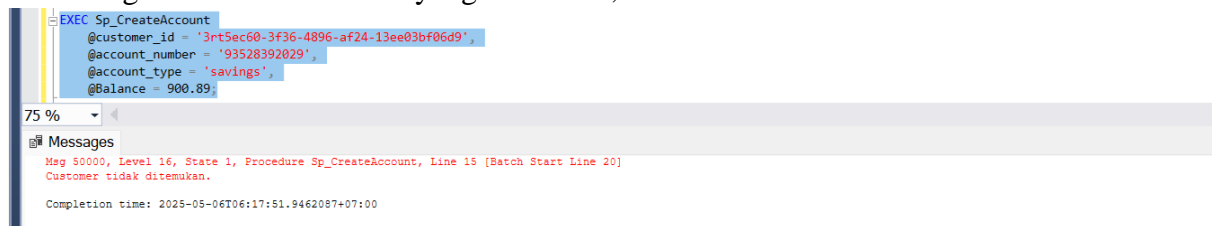
The Results pane shows the output of the query, displaying a single row with columns: customer\_id, first\_name, last\_name, email, phone\_number, address, and created\_at. The status bar indicates "Query executed successfully." and "1 rows".

### 2. Procedure sp\_CreateAccount untuk membuat akun baru untuk customer yang sudah ada: input customer\_id, account\_number, account\_type, balance. Notes: Tambahkan validasi untuk memastikan bahwa customer\_id benar benar ada pada table customers.

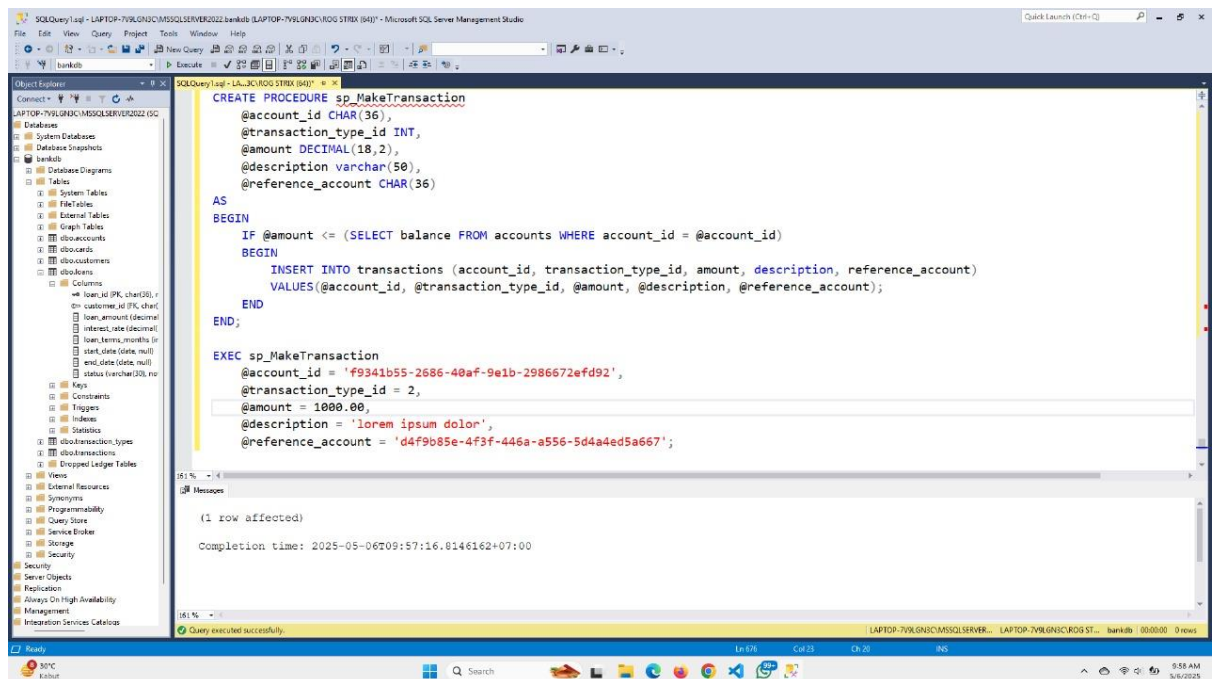




Jika mengeksekusi customer id yang tidak ada, maka:



3. Procedure `sp_MakeTransaction` untuk menambahkan transaksi baru: input `account_id`, `transaction_type_id`, `amount`, `description`, `reference_account` (opsional). Notes: Tambahkan validasi untuk transfer, jika jumlah transfer melebihi balance maka transaksi dihentikan



4. Procedure `sp_GetCustomerSummary` untuk menampilkan ringkasan data customer berdasarkan `customer_id`:
- 1.) Nama lengkap customer
  - 2.) Jumlah akun yang dimiliki
  - 3.) Jumlah total saldo semua akun
  - 4.) Jumlah pinjaman aktif
  - 5.) Total pinjaman amount aktif

The screenshot displays the Microsoft SQL Server Management Studio interface. The 'Object Explorer' on the left shows the database structure for 'LAPTOP-2ALUG9SQ (SQL Server 16.0.100)'. The main window shows a SQL query editor with the following code:

```
CREATE PROCEDURE sp_GetCustomerSummary
    @CustomerId CHAR(36)
AS
BEGIN
    SET NOCOUNT ON;

    SELECT
        CONCAT(c.first_name, ' ', c.last_name) AS full_name,
        COUNT(DISTINCT a.account_id) AS total_accounts,
        ISNULL(SUM(a.balance), 0) AS total_balance,
        COUNT(DISTINCT l.loan_id) AS active_loans,
        ISNULL(SUM(l.loan_amount), 0) AS total_active_loan_amount
    FROM customers c
    LEFT JOIN accounts a ON c.customer_id = a.customer_id
    LEFT JOIN loans l ON c.customer_id = l.customer_id AND l.status = 'active'
    WHERE c.customer_id = @CustomerId
    GROUP BY c.first_name, c.last_name;
END;
```

The query is executed, and the results are displayed in the 'Results' pane:

	full_name	total_accounts	total_balance	active_loans	total_active_loan_amount
1	Swen Crowley	1	10838.30	0	0.00

The status bar at the bottom indicates 'Query executed successfully.' and '1 rows'.