### Using OUTPUT Clause

Here's how you can do it with the OUTPUT clause, which directly returns the inserted ID:

DECLARE @InsertedIDs TABLE (ID INT);

-- Insert into DATA1 and get the inserted ID

INSERT INTO DATA1 (column1, column2, ...)

OUTPUT INSERTED.ID INTO @InsertedIDs

VALUES (value1, value2, ...);

-- Use the inserted ID to insert into DATA2

INSERT INTO DATA2 (data1\_id, columnA, columnB, ...)

SELECT ID, 'valueA', 'valueB', ...

FROM @InsertedIDs

### Using SCOPE\_IDENTITY()

Another way to retrieve the ID is using the SCOPE\_IDENTITY() function, which returns the last identity value generated in the current scope:

-- Insert into DATA1

INSERT INTO DATA1 (column1, column2, ...)

VALUES (value1, value2, ...);

-- Get the last inserted ID

DECLARE @LastInsertedID INT;

SET @LastInsertedID = SCOPE\_IDENTITY();

-- Use the inserted ID to insert into DATA2

INSERT INTO DATA2 (data1\_id, columnA, columnB, ...)

VALUES (@LastInsertedID, 'valueA', 'valueB', ...);

### Integrating with JavaScript

If you're doing this from a JavaScript environment (e.g., Node.js), you can execute these SQL statements and handle the results accordingly. Here’s an example using node-mssql:

const sql = require('mssql');

async function insertData() {

try {

// Connect to the database

let pool = await sql.connect('your\_connection\_string');

// Start a transaction

let transaction = new sql.Transaction(pool);

await transaction.begin();

// Insert into DATA1 and get the inserted ID

let request = new sql.Request(transaction);

let result = await request.query(`

DECLARE @InsertedIDs TABLE (ID INT);

INSERT INTO DATA1 (column1, column2, ...)

OUTPUT INSERTED.ID INTO @InsertedIDs

VALUES (value1, value2, ...);

SELECT ID FROM @InsertedIDs;

`);

// Get the inserted ID

let insertedId = result.recordset[0].ID;

// Insert into DATA2 using the inserted ID

await request.query(`

INSERT INTO DATA2 (data1\_id, columnA, columnB, ...)

VALUES (${insertedId}, 'valueA', 'valueB', ...);

`);

// Commit the transaction

await transaction.commit();

} catch (err) {

console.error(err);

if (transaction) await transaction.rollback();

}

}

insertData();