# **SQL Data Feed Processing and Validation Project**

Author: Bhagyashree Patil

Date: July 31, 2025

Database: MySQL Server 8.3.4

## **1. Executive Summary**

This document outlines the end-to-end process for a SQL-based data handling project. The project involves generating random data feeds, identifying and processing duplicate records, comparing data across different feeds, and creating a comprehensive test plan. All tasks were accomplished using direct SQL scripts executed in a MySQL 8.3.4 environment.

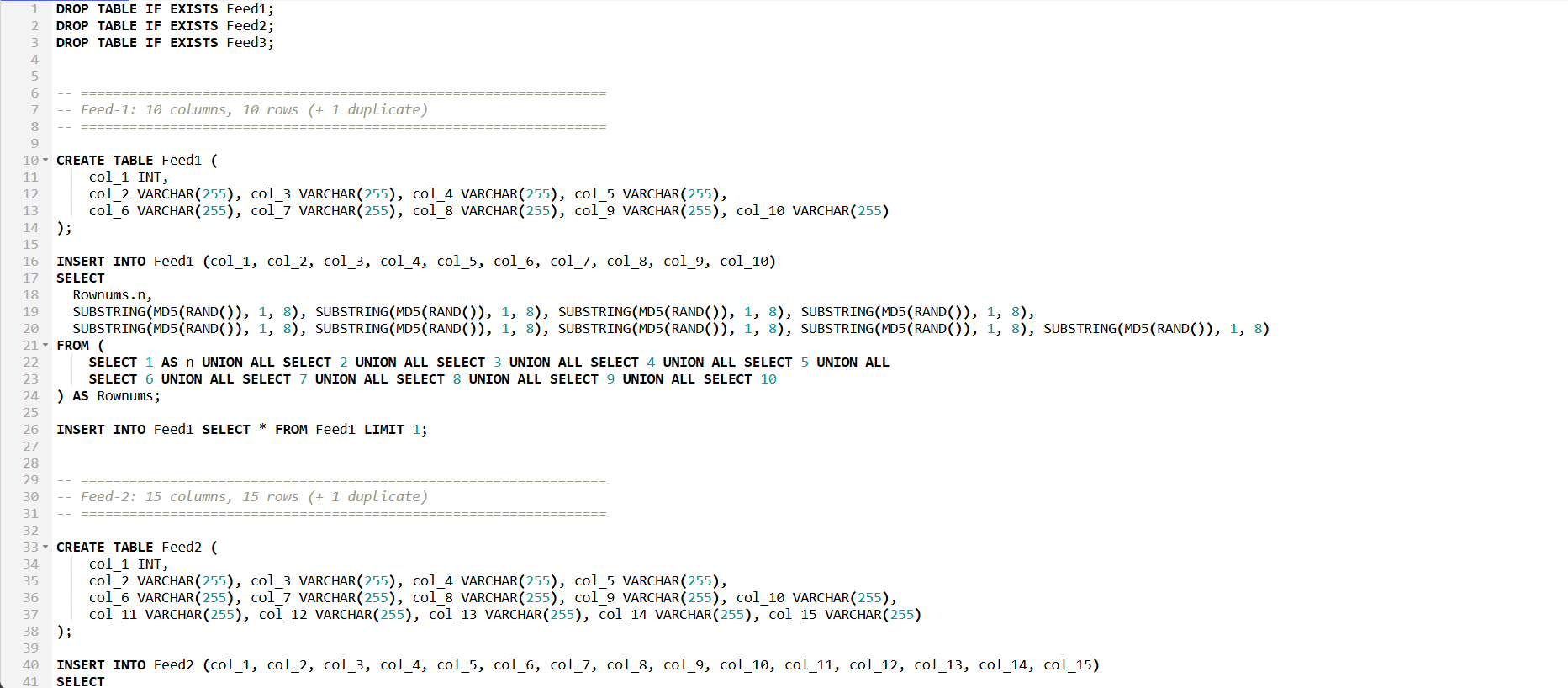
## **2. System Requirements**

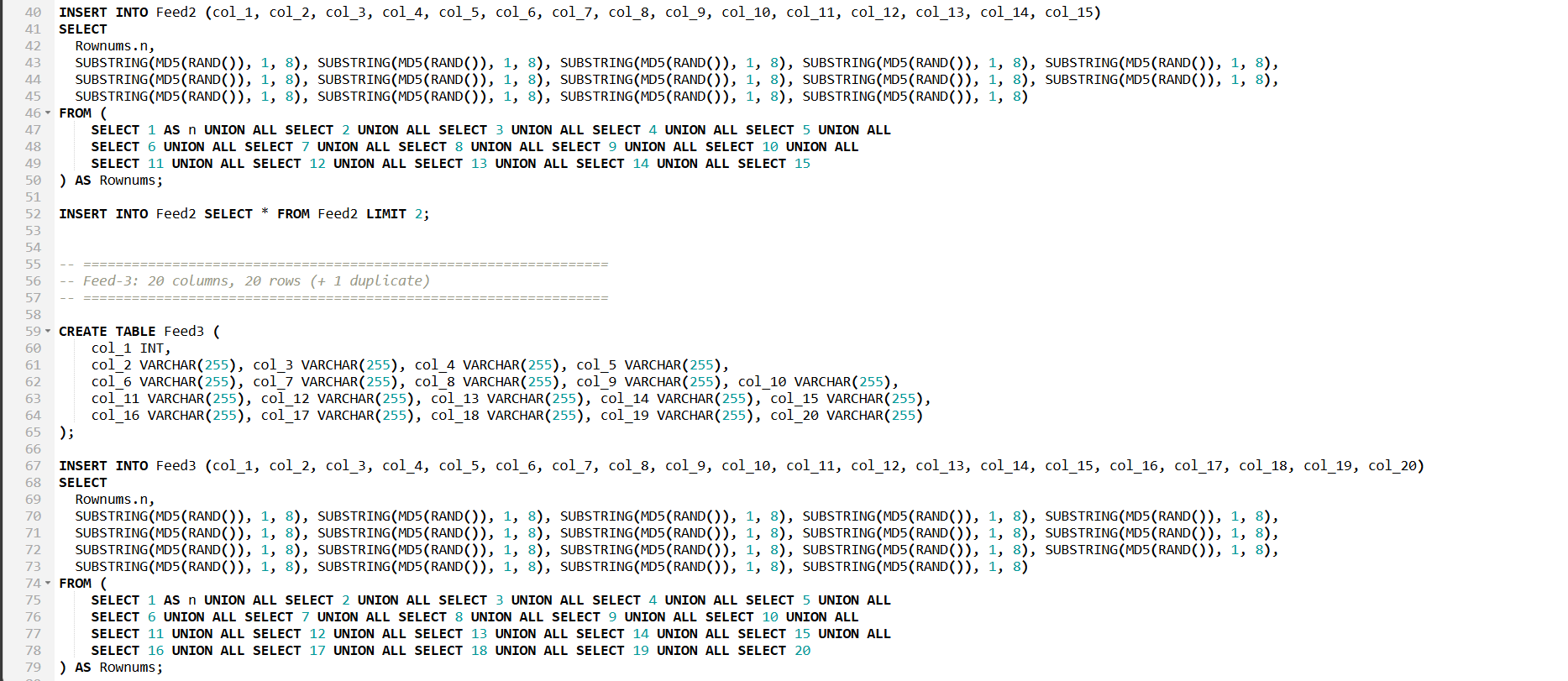
* **Database Server:** MySQL version 8.3.0 or compatible.
* **Client Tool:** Any standard SQL client (e.g., MySQL Workbench, DBeaver, or an online compiler).

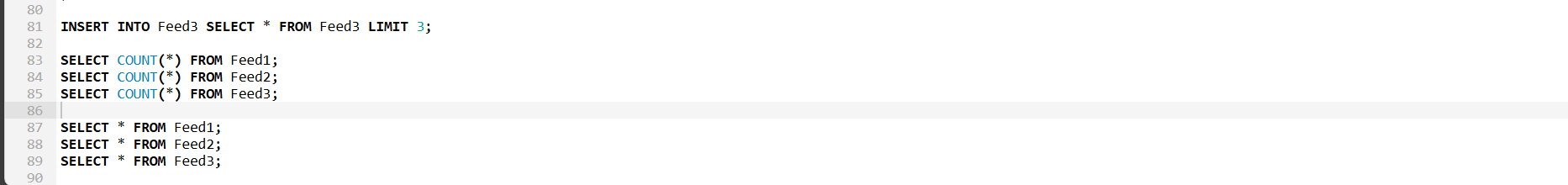
## **3. Task 1: Generate Random Data Input Files**

**Objective:** To create and populate three distinct tables (Feed1, Feed2, Feed3) with a specified number of columns and rows of random data. Some additional duplicate rows were added to each table to facilitate testing of subsequent tasks.

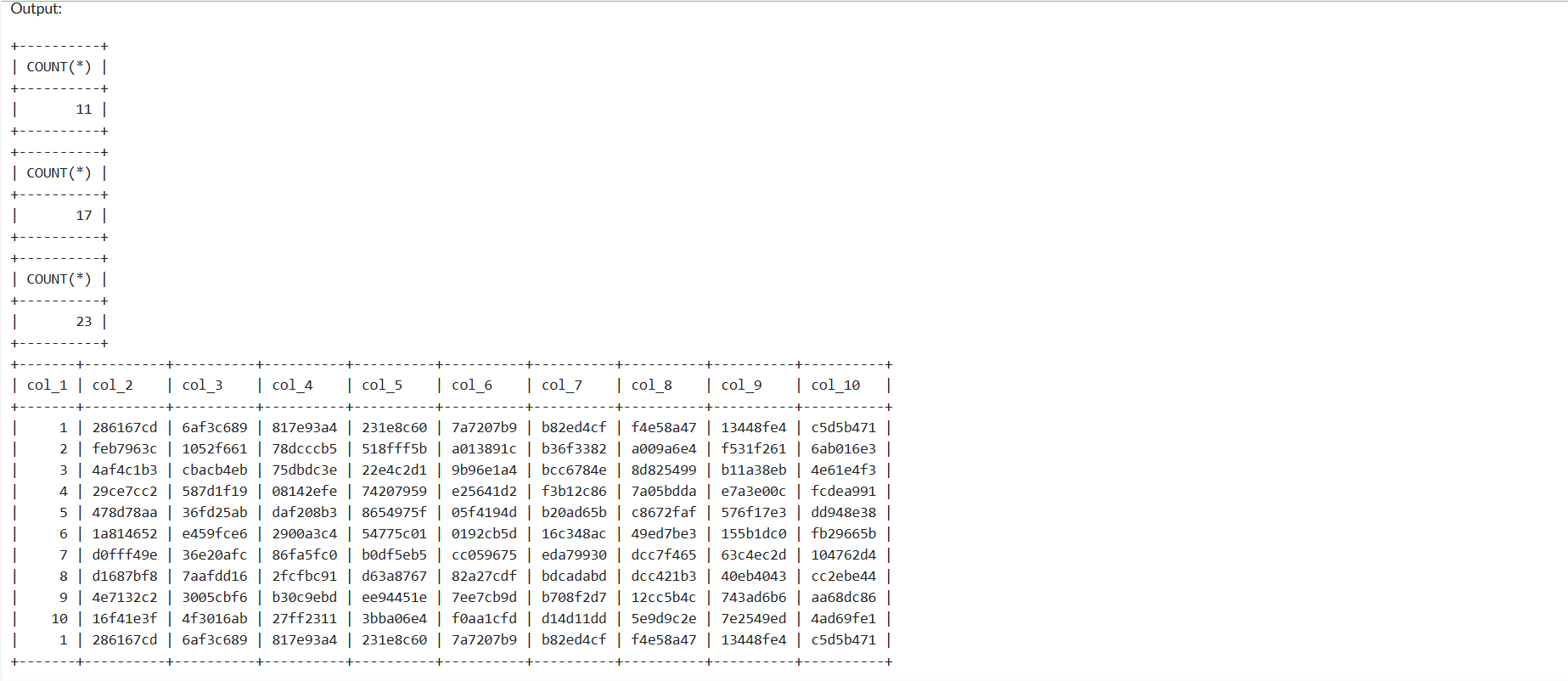
### **SQL Script:**

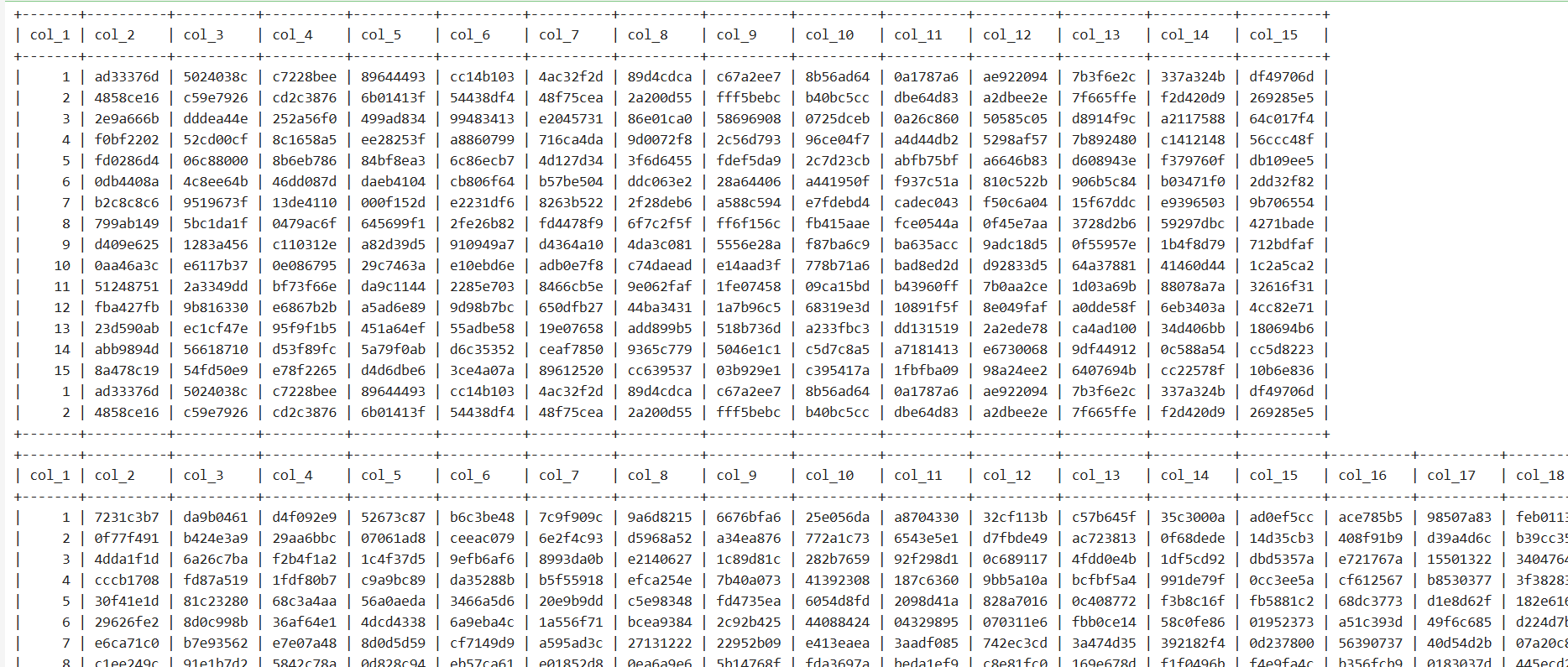


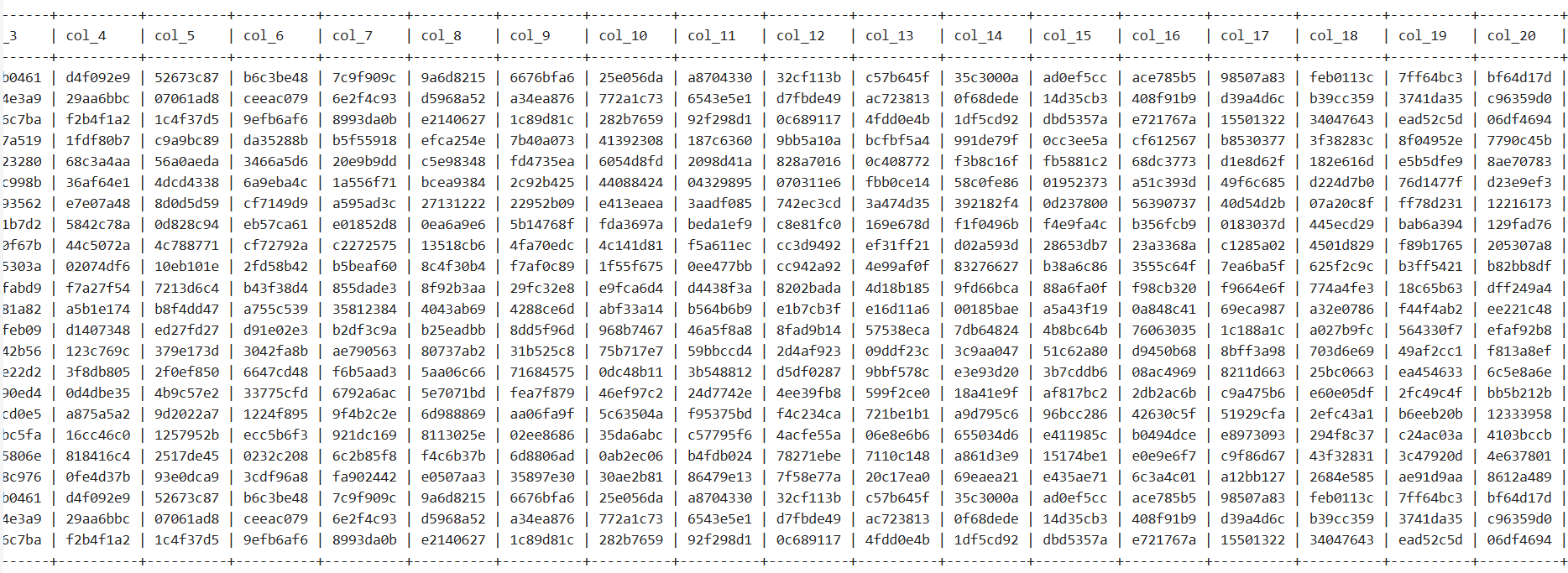




### **Screen Grab: Post-Execution Verification**

****

****

****

## 4. Task 2: Automated Task 1 Using Stored Procedures

**Objective:** To automate the Task 1 input file generation using SQL scripts where the parameters will be Feed Name, Number of Rows and Number of Columns.

### SQL Script:





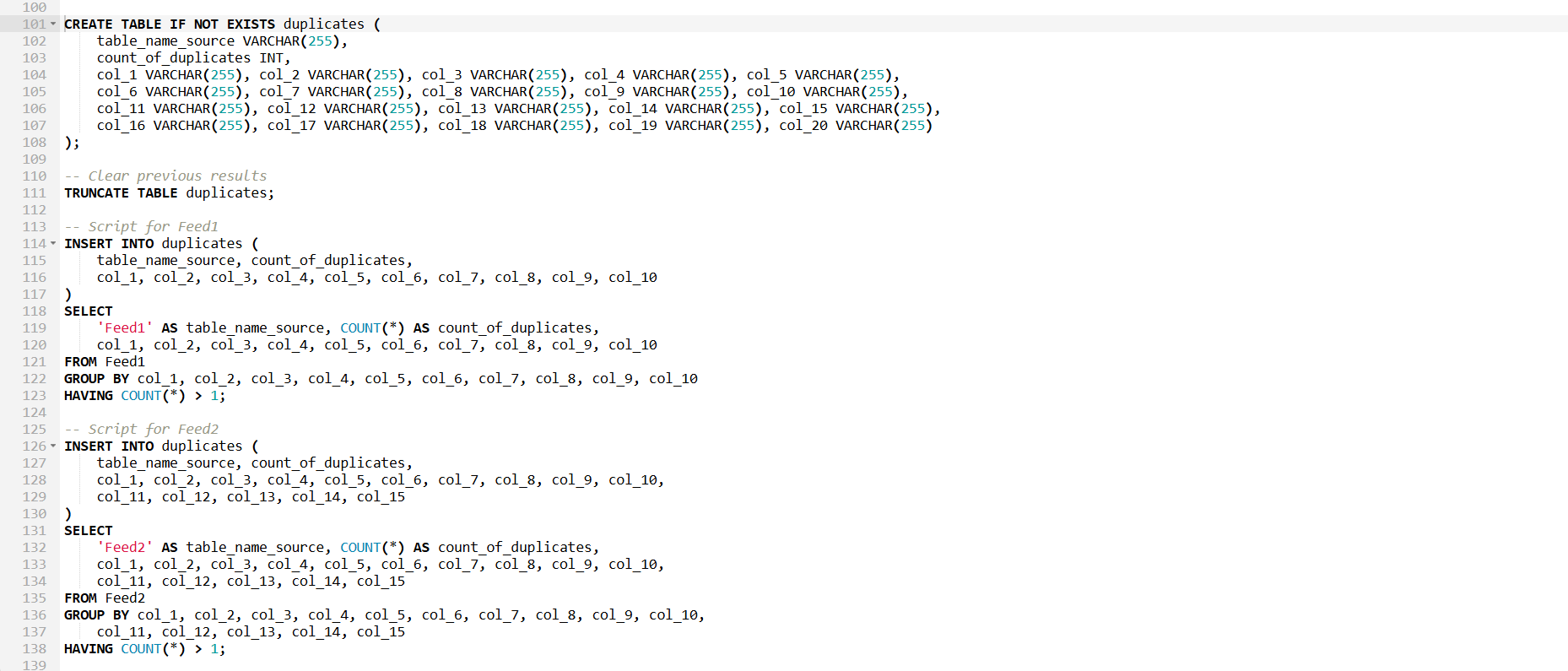
### Screen Grab:

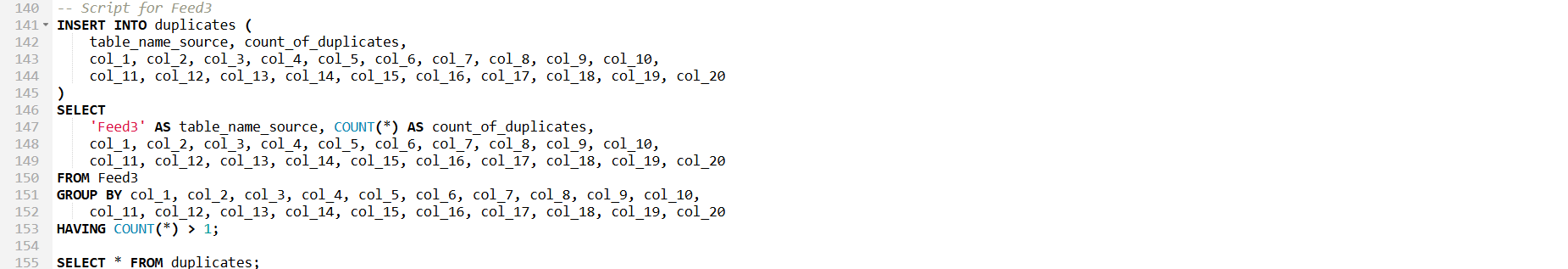


## 5**. Task 3 & 4: Identify and Store Duplicate Records**

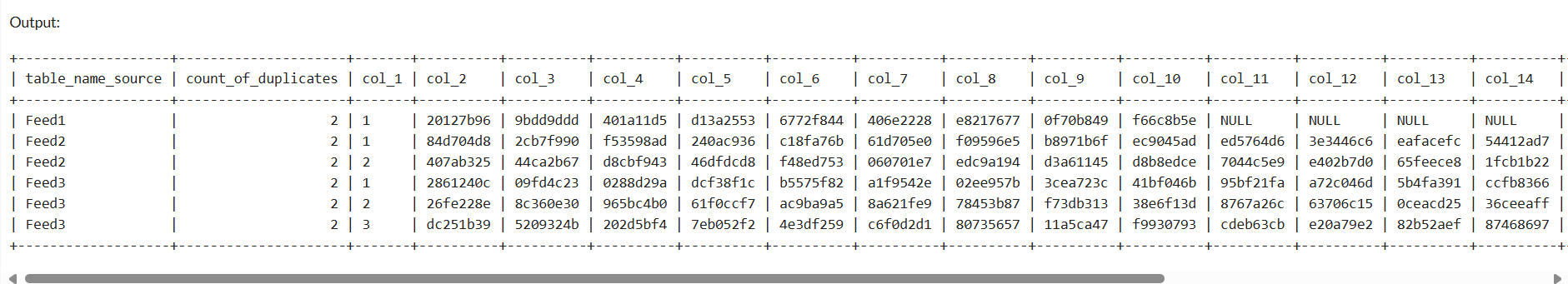
**Objective:** To identify all rows that are exact duplicates within each feed table and store a single instance of each duplicate set in a dedicated duplicates table for analysis.

### **SQL Script:**





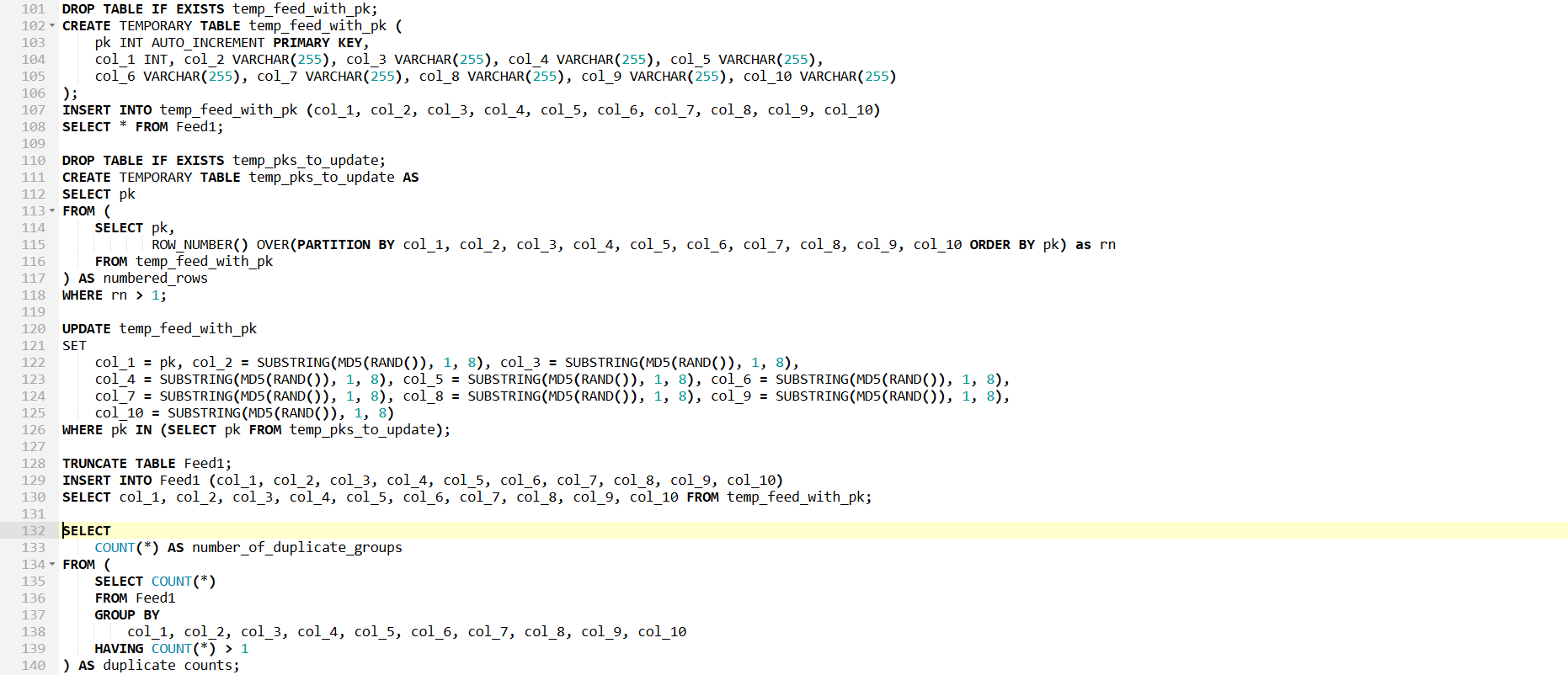
### **Screen Grab: Duplicate Records Identified**

****

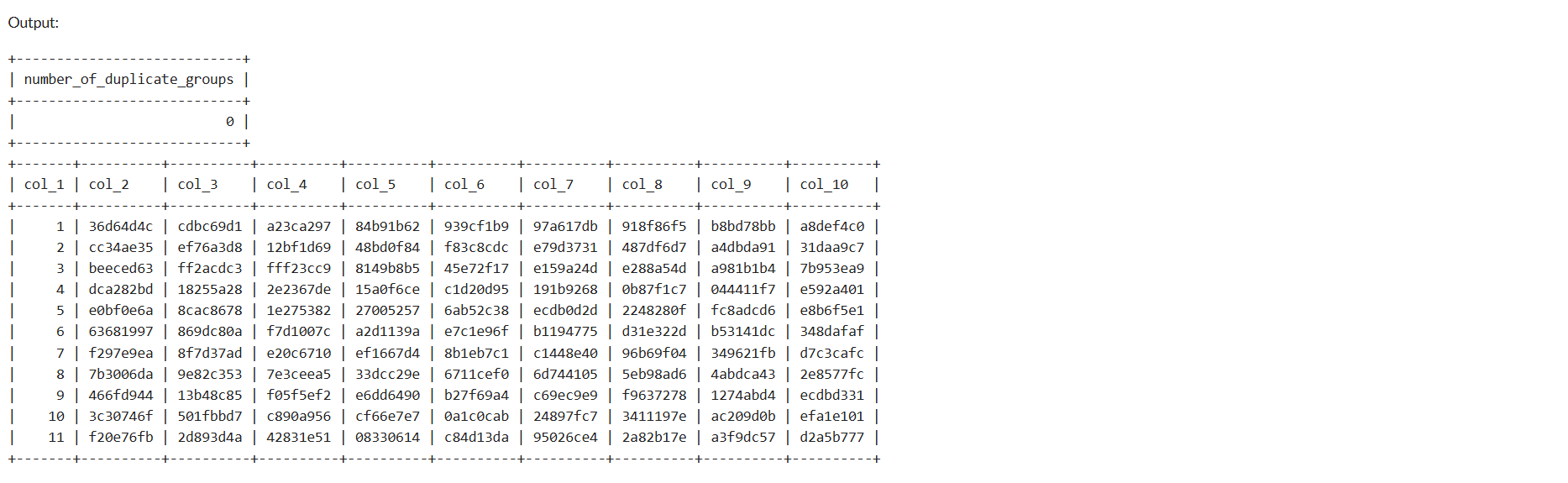
## 6**. Task 5 & 6: Replace Duplicates and Verify**

**Objective:** To process each feed table by replacing the identified duplicate rows with new, unique rows, ensuring the total row count remains unchanged. Finally, to verify that no duplicates exist in the processed tables.

### **SQL Script (Example for Feed1)**



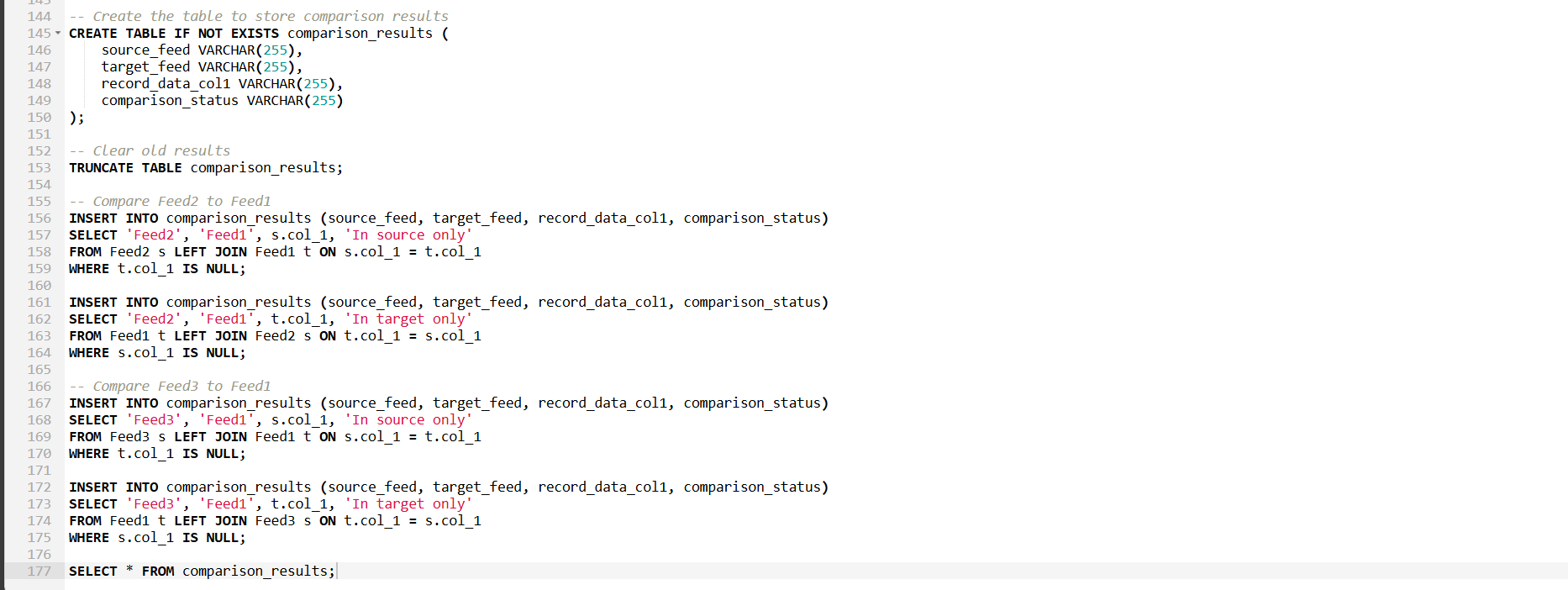
### **Screen Grab: Verification Result**

****

## 7**. Task 7: Compare Data Between Feeds**

**Objective:** To compare the data from Feed-2 and Feed-3 against Feed-1 to find records that are unique to each dataset. The comparison is based on the value in the first column (col\_1).

### **SQL Script**



### **Screen Grab: Comparison Results**

## 8**. Task 8 & 9: Test Plan and Automation**

**Objective:** To create a comprehensive test plan covering all functionalities and to automate the execution of these tests using a SQL stored procedure.

### **Manual Test Plan**

| **Test Case ID** | **Test Case Description** | **Expected Result** | **Test Steps** |
| --- | --- | --- | --- |
| **TC-01** | **Generate Feed-1** | A table named Feed1 is created with 10 columns and 11 rows (10 unique + 1 duplicate). | 1. Execute CALL generate\_feed('Feed1', 10, 10);  2. Verify the table Feed1 exists  3. Check SELECT COUNT(\*) FROM Feed1; returns 11  4. Check SELECT COUNT(\*) FROM INFORMATION\_SCHEMA.COLUMNS WHERE table\_name = 'Feed1'; returns 10. |
| **TC-02** | **Generate Feed-2** | A table named Feed2 is created with 15 columns and 16 rows. | 1. Execute CALL generate\_feed('Feed2', 15, 15);  2. Verify the table Feed2 exists.  3. Check row and column counts. |
| **TC-03** | **Generate Feed-3** | A table named Feed3 is created with 20 columns and 21 rows. | 1. Execute CALL generate\_feed('Feed3', 20, 20);  2. Verify the table Feed3 exists.  3. Check row and column counts. |
| **TC-04** | **Identify Duplicates in Feed-1** | The duplicates table should contain one record from Feed1 with a count\_of\_duplicates of 2. | 1. TRUNCATE TABLE duplicates;  2. CALL find\_and\_store\_duplicates('Feed1');  3. SELECT COUNT(\*) FROM duplicates WHERE table\_name\_source = 'Feed1'; should be 1. |
| **TC-05** | **Replace Duplicates in Feed-1** | The Feed1 table should contain only 10 unique rows. | 1. CALL replace\_duplicates\_with\_unique('Feed1');. 2. SELECT COUNT(\*) FROM Feed1; should be 10. |
| **TC-06** | **Verify No Duplicates in Feed-1** | The verification script should report that no duplicates are found. | 1. CALL verify\_no\_duplicates('Feed1');  2. The result message should indicate success. |
| **TC-07** | **Compare Feed-2 to Feed-1** | The comparison\_results table should contain records indicating which rows are unique to Feed2 and which are unique to Feed1. | 1. TRUNCATE TABLE comparison\_results;. 2. CALL compare\_feeds('Feed2', 'Feed1');  3. SELECT \* FROM comparison\_results; and manually verify the logic. |
| **TC-08** | **Generate Feed with Zero Rows** | A table should be created with the specified columns but 0 rows. | 1. CALL generate\_feed('Feed\_Zero', 5, 0);  2. SELECT COUNT(\*) FROM Feed\_Zero; should return 0. |
| **TC-09** | **Duplicate Check on Clean Table** | The find\_and\_store\_duplicates procedure should not add any records to the duplicates table. | 1. CALL replace\_duplicates\_with\_unique('Feed2');. 2. TRUNCATE TABLE duplicates;. 3. CALL find\_and\_store\_duplicates('Feed2');. 4. SELECT COUNT(\*) FROM duplicates; should be 0. |

### **Test Automation Script**



### **Screen Grab: Automated Test Execution**

****