



## **Sales Data Analysis and Data Quality Check Report**

**Author - Shantanu Zodey**

**Date – 30/05/2025**

## Source Data Verification.

### Customer.xlsx –

- Consist of total 250 records.
- 5 Columns – (Customer\_ID, First, Last, Age, Country)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	F
1	Customer_ID	First	Last	Age	Country											
2	1	Joseph	Rice	43	USA											
3	2	Gary	Moore	71	USA			Total Rows	250							
4	3	John	Walker	44	UK											
5	4	Eric	Carter	38	UK											
6	5	William	Jackson	58	UAE											
7	6	Nicole	Jones	33	USA											
8	7	David	Davis	59	USA											
9	8	Jason	Montgomery	58	UK											
10	9	Kent	Weaver	61	UK											
11	10	Darrell	Dillon	50	UAE											
12	11	Jacqueline	Wang	22	USA											
13	12	Jodi	Gonzalez	69	USA											
14	13	Omar	Martin	28	UK											
15	14	Nicole	Lara	77	UK											
16	15	Jason	Brown	63	UAE											
17	16	David	Benson	61	USA											
18	17	Kimberly	Mora	34	USA											
19	18	Erik	Macias	25	UK											
20	19	James	Johnson	67	UK											
21	20	Derek	Peterson	36	UAE											
22	21	Diane	Henson	74	USA											
23	22	Cody	Lyons	47	USA											
24	23	Margaret	Hardy	25	UK											
25	24	Arthur	Hayes	75	UK											
26	25	Raymond	Taylor	36	UAE											
27	26	Tina	Moore	34	USA											
28	27	Kylie	White	63	USA											

Customers

Ready Accessibility: Unavailable

Average: 125.5 Count: 250 Sum: 31375

### Orders.csv –

- Consist of total 250 records.
- 5 Columns – (Order\_ID, Item, Amount, Customer\_ID)

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
1	Order_ID	Item	Amount	Customer_ID															
2	1	Keyboard	400	139															
3	2	Mouse	300	250			Total Row	250											
4	3	Monitor	12000	239															
5	4	Keyboard	400	153															
6	5	Mousepad	250	153															
7	6	Harddisk	5000	164															
8	7	Webcam	350	124															
9	8	Mousepad	200	98															
10	9	DDR RAM	1500	119															
11	10	Headset	900	159															
12	11	Keyboard	400	167															
13	12	Mouse	300	157															
14	13	Monitor	12000	49															
15	14	Keyboard	400	234															
16	15	Mousepad	250	202															
17	16	Harddisk	5000	181															
18	17	Webcam	350	15															
19	18	Mousepad	200	177															
20	19	DDR RAM	1500	5															
21	20	Headset	900	225															
22	21	Keyboard	400	171															
23	22	Mouse	300	67															
24	23	Monitor	12000	68															
25	24	Keyboard	400	86															

Order

Ready Accessibility: Unavailable

Average: 125.5 Count: 250 Sum: 31375

Shipping.json -> Shipping.csv –

- Consist of total 250 records.
- 5 Columns – (Shipping\_ID, Status, Customer\_ID)

A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
1	Shipping_ID	Status	Customer_ID														
2	1	Pending	173														
3	2	Pending	155														
4	3	Delivered	242														
5	4	Pending	223														
6	5	Delivered	72														
7	6	Pending	29														
8	7	Pending	236														
9	8	Delivered	193														
10	9	Pending	199														
11	10	Delivered	17														
12	11	Pending	157														
13	12	Pending	37														
14	13	Delivered	141														
15	14	Pending	155														
16	15	Delivered	131														
17	16	Pending	24														
18	17	Pending	186														
19	18	Delivered	177														
20	19	Pending	129														
21	20	Delivered	6														
22	21	Pending	193														
23	22	Pending	109														
24	23	Delivered	232														
25	24	Pending	145														

## Customer Data Quality

Validating row count and the records that are loaded into staging table from Customer.xlsx.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for LAPTOP-B4HM0U40. The central pane shows a query window with the following SQL code:

```
-- Checking row counts in staging tables  
SELECT COUNT(*) AS Total_Customer_Rows FROM stg_customers;
```

The Results pane on the right shows the output of the query:

Total_Customer_Rows
250

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for LAPTOP-B4HM0U40. The central pane shows a query window with the following SQL code:

```
select * from stg_customers
```

The Results pane on the right shows the output of the query, displaying a list of customer records with columns: Customer\_ID, FirstName, LastName, Age, and Country.

Customer_ID	FirstName	LastName	Age	Country
1	Joseph	Rice	43	USA
2	Gary	Moore	71	USA
3	John	Walker	44	UK
4	Eric	Carter	38	UK
5	William	Jackson	58	USA
6	Nicole	Jones	33	USA
7	David	Cole	59	USA
8	Jason	Montgomery	58	UK
9	Kent	Weaver	61	UK
10	Danell	Dillon	50	USA
11	Jacqueline	Wang	22	USA
12	Jodi	Gonzalez	69	USA
13	Ormar	Martin	28	UK
14	Nicole	Lara	77	UK
15	Jason	Brown	63	USA
16	David	Benson	61	USA
17	Kristin	Mora	34	USA
18	Eric	Macias	25	UK
19	James	Johnson	67	UK
20	Derek	Peterson	36	USA
21	Diane	Henson	74	USA
22	Cody	Lyons	47	USA
23	Margaret	Hardy	29	UK
24	Arthur	Hayes	75	UK
25	Raymond	Taylor	36	USA
26	Tina	Moore	34	USA
27	Kylie	White	63	USA
28	Jonathan	Manning	30	UK
29	Angela	Bryant	23	UK
30	Michael	Merr	56	USA

## Completeness Checks

### Checking Nulls in stg\_customer table

The screenshot shows a SQL Server Enterprise Manager interface. The Object Explorer on the left shows the database structure. The main window displays a query titled "Checking Nulls in stg\_customers table". The query is as follows:

```
-- Checking Nulls in stg_customers table
SELECT
COUNT(*) AS total_records,
SUM(CASE WHEN Customer_ID IS NULL THEN 1 ELSE 0 END) AS null_customer_id,
SUM(CASE WHEN FirstName IS NULL THEN 1 ELSE 0 END) AS null_firstname,
SUM(CASE WHEN LastName IS NULL THEN 1 ELSE 0 END) AS null_lastname,
SUM(CASE WHEN Age IS NULL THEN 1 ELSE 0 END) AS null_age,
SUM(CASE WHEN Country IS NULL THEN 1 ELSE 0 END) AS null_country
FROM stg_customers;
```

The Results pane shows the following data:

	total_records	null_customer_id	null_firstname	null_lastname	null_age	null_country
1	250	0	0	0	0	0

## Accuracy Checks

### Checking Alpha-Numeric and Special characters in First Name

The screenshot shows a SQL Server Enterprise Manager interface. The Object Explorer on the left shows the database structure. The main window displays a query titled "/\*Checking Data Type Validity\*/". The query is as follows:

```
/*Checking Data Type Validity*/
-- Alpha-Numeric and Special Characters in First Name
SELECT * FROM stg_customers
WHERE FirstName LIKE '%[^a-zA-Z0-9 _-]%'
```

The Results pane shows the following data:

Customer_ID	FirstName	LastName	Age	Country
6	Nicole	Jones	33	USA
14	Nicole	Lara	77	UK
162	Nicole	Bennett	51	USA
171	L@rry	Cole	50	USA
214	Nicole	Mcintyre	18	UK

In the above screenshot, there are few records in First Name that contains special characters. This column needs transformation in order to get the clean and accurate data.

### Checking Alpha-Numeric and Special characters in Last Name and Country

The screenshot shows a SQL Server Enterprise Manager interface. The Object Explorer on the left shows the database structure. The main window displays two queries. The first query is titled "Alpha-Numeric and Special Characters in Last Name" and the second query is titled "Alpha-Numeric and Special Characters in Country".

The first query is as follows:

```
-- Alpha-Numeric and Special Characters in Last Name
SELECT * FROM stg_customers
WHERE LastName LIKE '%[^a-zA-Z0-9 _-]%'
```

The second query is as follows:

```
-- Alpha-Numeric and Special Characters in Country
SELECT * FROM stg_customers
WHERE Country LIKE '%[^a-zA-Z0-9 _-]%'
```

The Results pane shows the following data:

Customer_ID	FirstName	LastName	Age	Country
-------------	-----------	----------	-----	---------

## Checking Negative or Unrealistic Ages.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure. The main window shows a query window with the following SQL code:

```
-- Age Range
SELECT MIN(Age) as Min_Age, MAX(Age) as Max_Age, AVG(Age) as Avg_Age FROM stg_customers;

-- Negative or unrealistic ages
SELECT * FROM stg_customers WHERE Age <= 0 OR Age >= 120;
```

The query results are displayed in a table with the following columns: Min\_Age, Max\_Age, and Avg\_Age. The results are:

Min_Age	Max_Age	Avg_Age
18	80	47

## Reliability Checks

### Checking Duplicate Customer IDs in stg\_customer table

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure. The main window shows a query window with the following SQL code:

```
-- Duplicate CustomerIDs
SELECT Customer_ID, COUNT(*) AS Duplicate_CustomerIDs
FROM stg_customers
GROUP BY Customer_ID
HAVING COUNT(*) > 1;
```

The query results are displayed in a table with the following columns: Customer\_ID and Duplicate\_CustomerIDs.

## Orders Data Quality

### Validating row count and the records that are loaded into staging table from Customer.xlsx.

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure. The main window shows a query window with the following SQL code:

```
SELECT COUNT(*) AS Total_Orders_Rows FROM stg_orders;
```

The query results are displayed in a table with the following columns: Total\_Orders\_Rows. The results are:

Total_Orders_Rows
250

Object Explorer

Connect - LAPTOP-B4HMOU4O (SQL Server 15.0.2130.3 - LAPTOP-B4)

Databases

Security

Server Objects

Replication

PolyBase

Always On High Availability

Management

Integration Services Catalogs

SQL Server Agent (Agent XPs disabled)

XEvent Profiler

SQLQuery2.sql - LA...antanu Zodey (58)\*

select \* from stg\_orders

100 %

Results Messages

Order_ID	Item	Amount	Customer_ID
1	Keyboard	400	139
2	Mouse	300	250
3	Monitor	12000	239
4	Keyboard	400	153
5	Mousepad	250	153
6	Harddisk	5000	184
7	Webcam	350	124
8	Mousepad	200	98
9	DDR RAM	1500	119
10	Headset	900	159
11	Keyboard	400	167
12	Mouse	300	157
13	Monitor	12000	49
14	Keyboard	400	234
15	Mousepad	250	202
16	Harddisk	5000	181
17	Webcam	350	15
18	Mousepad	200	177
19	DDR RAM	1500	5
20	Headset	900	225
21	Keyboard	400	171
22	Mouse	300	67
23	Monitor	12000	68
24	Keyboard	400	86
25	Mousepad	250	8
26	Harddisk	5000	56
27	Webcam	350	60
28	Mousepad	200	188
29	DDR RAM	1500	33
30	Headset	900	157

Query executed successfully.

LAPTOP-B4HMOU4O (15.0 RTM) LAPTOP-B4HMOU4O/Shanta... PEISalesDB 00:00:00 250 rows

## Completeness Checks

### Checking Nulls in stg\_orders table

Object Explorer

Connect - LAPTOP-B4HMOU4O (SQL Server 15.0.2130.3 - LAPTOP-B4)

Databases

Security

Server Objects

Replication

PolyBase

Always On High Availability

Management

Integration Services Catalogs

SQL Server Agent (Agent XPs disabled)

XEvent Profiler

PEI Sales Data Ana...antanu Zodey (52)

```
-- Checking Nulls in stg_orders table
SELECT
COUNT(*) AS total_records,
SUM(CASE WHEN Order_ID IS NULL THEN 1 ELSE 0 END) AS null_order_id,
SUM(CASE WHEN Item IS NULL THEN 1 ELSE 0 END) AS null_item,
SUM(CASE WHEN Amount IS NULL THEN 1 ELSE 0 END) AS null_amount,
SUM(CASE WHEN Customer_ID IS NULL THEN 1 ELSE 0 END) AS null_customer_id
FROM stg_orders;
```

100 %

Results Messages

total_records	null_order_id	null_item	null_amount	null_customer_id
250	0	0	0	0

### Checking Product Distribution

Object Explorer

Connect - LAPTOP-B4HMOU4O (SQL Server 15.0.2130.3 - LAPTOP-B4)

Databases

Security

Server Objects

Replication

PolyBase

Always On High Availability

Management

Integration Services Catalogs

SQL Server Agent (Agent XPs disabled)

XEvent Profiler

SQLQuery4.sql - LA...antanu Zodey (63)\*

PEI Sales Data Ana...antanu Zodey (58)

```
-- Product distribution
SELECT Item, COUNT(*) FROM stg_orders GROUP BY Item;
```

121 %

Results Messages

Item	(No column name)
DDR RAM	25
Harddisk	25
Headset	25
Keyboard	50
Monitor	25
Mouse	25
Mousepad	50
Webcam	25

## Accuracy Checks

### Amount Validation

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the server 'LAPTOP-B4HM0U40' selected. The right pane shows a query window with the following SQL code:

```
-- Amount validation by product
SELECT Item, MIN(Amount) as Min_Amount, MAX(Amount) as Max_Amount, AVG(Amount) as Avg_Amount
FROM stg_orders
GROUP BY Item;
```

The query results are displayed in a table with the following data:

Item	Min_Amount	Max_Amount	Avg_Amount
DDR RAM	1500	1500	1500
Harddisk	5000	5000	5000
Headset	900	900	900
Keyboard	400	400	400
Monitor	12000	12000	12000
Mouse	300	300	300
Mousepad	200	250	225
Webcam	350	350	350

### Invalid Customer\_ID references

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the server 'LAPTOP-B4HM0U40' selected. The right pane shows a query window with the following SQL code:

```
-- Invalid Customer_ID references
SELECT o.* FROM stg_orders o
LEFT JOIN stg_customers c ON o.Customer_ID = c.Customer_ID
WHERE c.Customer_ID IS NULL;
```

The query results are displayed in a table with the following data:

Order_ID	Item	Amount	Customer_ID
----------	------	--------	-------------

## Reliability Checks

### Checking Duplicate Order\_IDs in stg\_order table

The screenshot shows the SQL Server Enterprise Manager interface. The left pane displays the Object Explorer with the server 'LAPTOP-B4HM0U40' selected. The right pane shows a query window with the following SQL code:

```
-- Duplicate Order_IDs
SELECT Order_ID, COUNT(*) AS Duplicate_Order_IDs
FROM stg_orders
GROUP BY Order_ID
HAVING COUNT(*) > 1;
```

The query results are displayed in a table with the following data:

Order_ID	Duplicate_Order_IDs
----------	---------------------

# Shipping Data Quality

Validating row count and the records that are loaded into staging table from Customer.xlsx.

The first screenshot shows a SQL query in SQLQuery4.sql: `SELECT COUNT(*) AS Total_Shipping_Rows FROM stg_shippings;`. The results pane shows a single row with the value 250.

The second screenshot shows a SQL query in SQLQuery2.sql: `select * from stg_shippings`. The results pane displays a table with 30 rows of data.

Shipping_ID	Status	Customer_ID
1	Pending	173
2	Pending	155
3	Delivered	242
4	Pending	223
5	Delivered	72
6	Pending	29
7	Pending	236
8	Delivered	193
9	Pending	199
10	Delivered	17
11	Pending	157
12	Pending	37
13	Delivered	141
14	Pending	155
15	Delivered	131
16	Pending	24
17	Pending	186
18	Delivered	177
19	Pending	129
20	Delivered	6
21	Pending	193
22	Pending	109
23	Delivered	232
24	Pending	145
25	Delivered	2
26	Pending	113
27	Pending	58
28	Delivered	93
29	Pending	182
30	Delivered	205

## Completeness Checks

Checking Nulls in stg\_shipping table

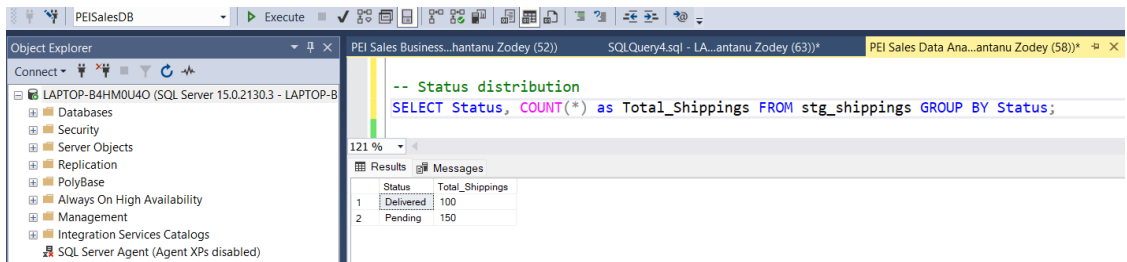
The screenshot shows a SQL query in SQLQuery2.sql: `-- Checking Nulls in stg_shippings table  
SELECT  
COUNT(*) AS total_records,  
SUM(CASE WHEN Shipping_ID IS NULL THEN 1 ELSE 0 END) AS null_shipping_id,  
SUM(CASE WHEN Status IS NULL THEN 1 ELSE 0 END) AS null_status,  
SUM(CASE WHEN Customer_ID IS NULL THEN 1 ELSE 0 END) AS null_customer_id  
FROM stg_shippings;`

The results pane shows a single row with the following values:

total_records	null_shipping_id	null_status	null_customer_id
250	0	0	0

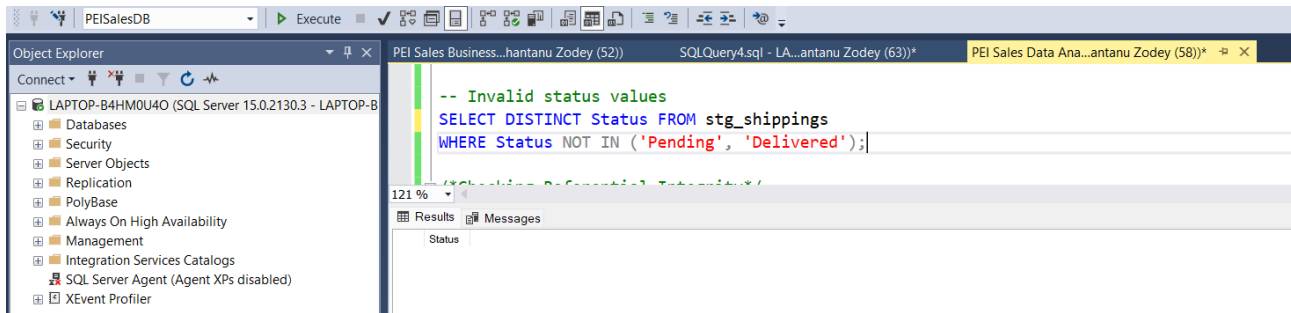
Shipping by Status Distribution





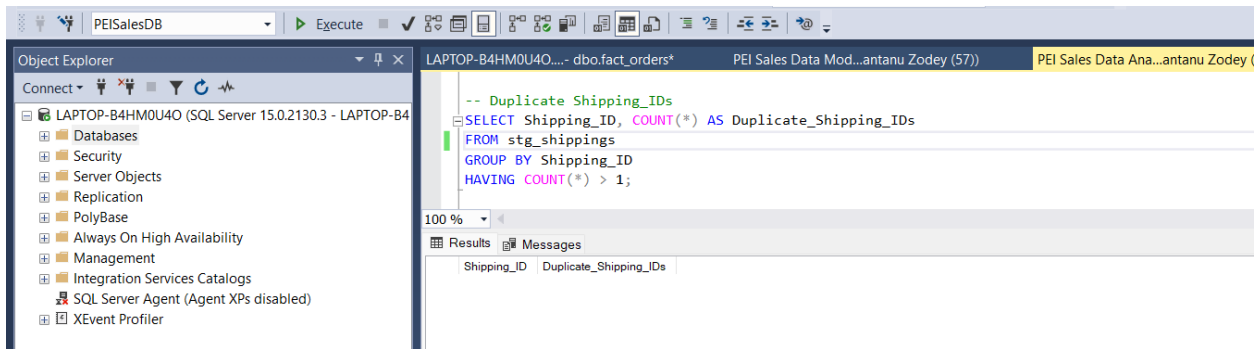
# Accuracy Checks

## Checking other status apart from Pending and Delivered

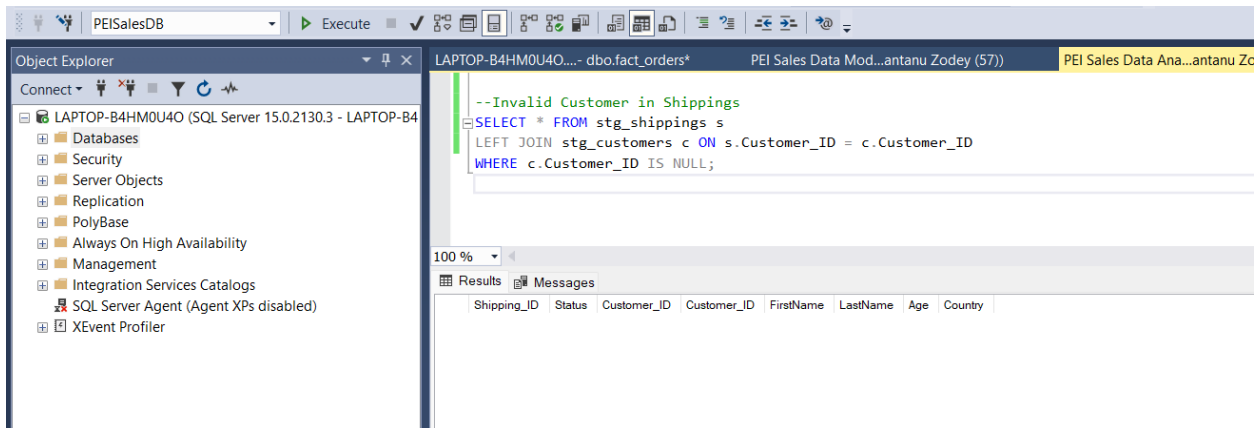


# Reliability Checks

## Checking Duplicate Order\_IDs in stg\_shipping table



## Invalid customers present in stg\_shippings



# Cross-Dataset Integrity Analysis

## Orders vs Shipping

### Unique CustomerIDs in Orders and Shippings table

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the server structure. The main window shows a query titled "SQLQuery4.sql - LA...antanu Zodey (63))\*" with the following SQL code:

```
-- Count unique customers in Orders
SELECT COUNT(DISTINCT Customer_ID) AS unique_order_customers FROM stg_orders;

-- Count unique customers in Shipping
SELECT COUNT(DISTINCT Customer_ID) AS unique_shipping_customers FROM stg_shippings;
```

The Results pane shows the output of the query:

unique_order_customers
160

unique_shipping_customers
154

### Customer\_IDs that are present in Orders but not in Shipping tables

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the server structure. The main window shows a query titled "SQLQuery4.sql - LA...antanu Zodey (63))\*" with the following SQL code:

```
-- Customers in Orders but not Shipping (61)
SELECT COUNT(DISTINCT o.Customer_ID) as Customers_order_vs_shippings
FROM stg_orders o
LEFT JOIN stg_shippings s ON o.Customer_ID = s.Customer_ID
WHERE s.Shipping_ID IS NULL;
```

The Results pane shows the output of the query:

Customers_order_vs_shippings
61

### Customer\_IDs that are present in Shipping but not in Order tables

The screenshot shows the SQL Server Enterprise Manager interface. The Object Explorer on the left displays the server structure. The main window shows a query titled "SQLQuery4.sql - LA...antanu Zodey (63))\*" with the following SQL code:

```
-- Customers in Shipping but not Orders
SELECT COUNT(DISTINCT s.Customer_ID) as Customers_shipping_vs_orders
FROM stg_shippings s
LEFT JOIN stg_orders o ON s.Customer_ID = o.Customer_ID
WHERE o.Order_ID IS NULL;
```

The Results pane shows the output of the query:

Customers_shipping_vs_orders
55

## Key Findings

### 1. Orders Analysis:

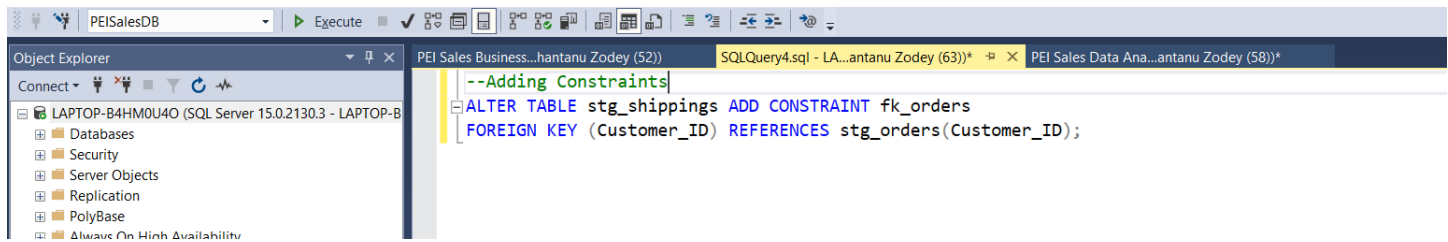
- Found 160 unique Customer\_ID values in stg\_orders
- When compared with stg\_shippings, 61 of these didn't exist in shipping records
- These represent orders that were placed but never shipped

### 2. Shipping Analysis:

- Found 154 unique Customer\_ID values in stg\_shippings
- When compared with stg\_orders, 55 had shipping records but no orders
- These are problematic as shipping should always reference an order

## Recommendations

Preventive Measure to tackle this issue:



Implementing FOREIGN KEY constraint between Orders and Shipping tables will help prevent invalid records.