Database Using SQL Server & Raw Data available over my GitHub Profile

MySQL

PostgreSQL

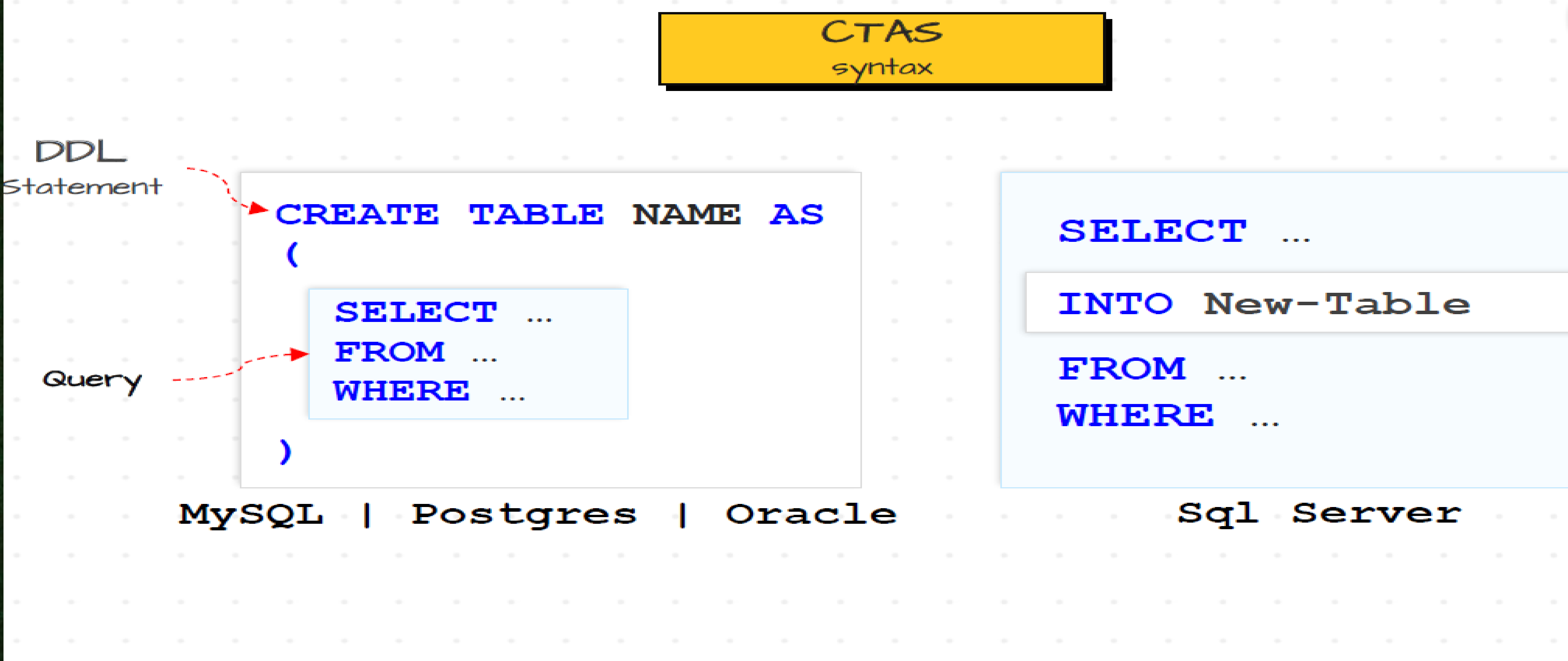
SQL CTAS With 12 Most Used Queries

AnshLibrary

What is a CTAS

The **CREATE TABLE AS SELECT (CTAS)** statement is one of the most important T-SQL features available. **CTAS** is a parallel operation that creates a new table based on the output of a **SELECT** statement.

Syntax :

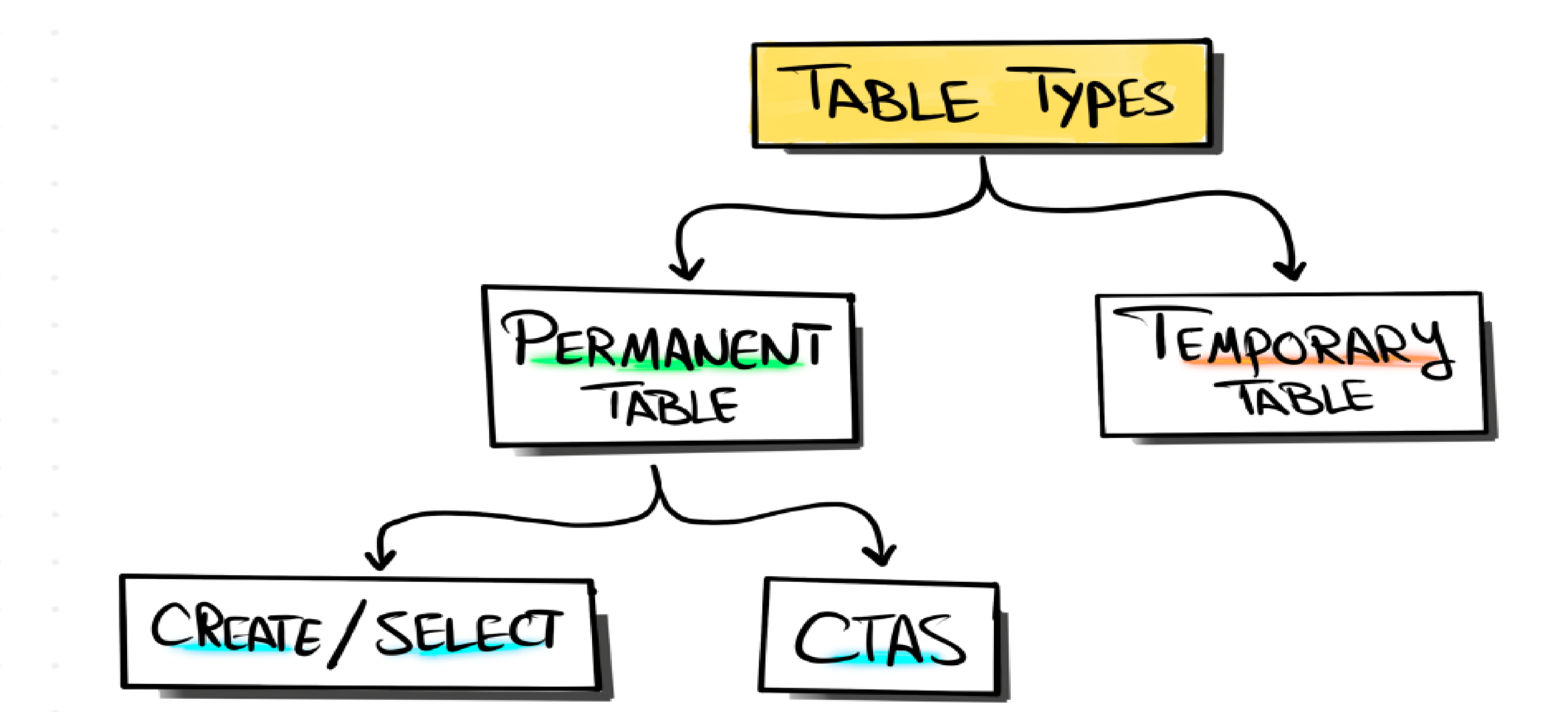


```
CREATE TABLE NAME AS
(
   SELECT ...
   FROM ...
   WHERE ...
)
```

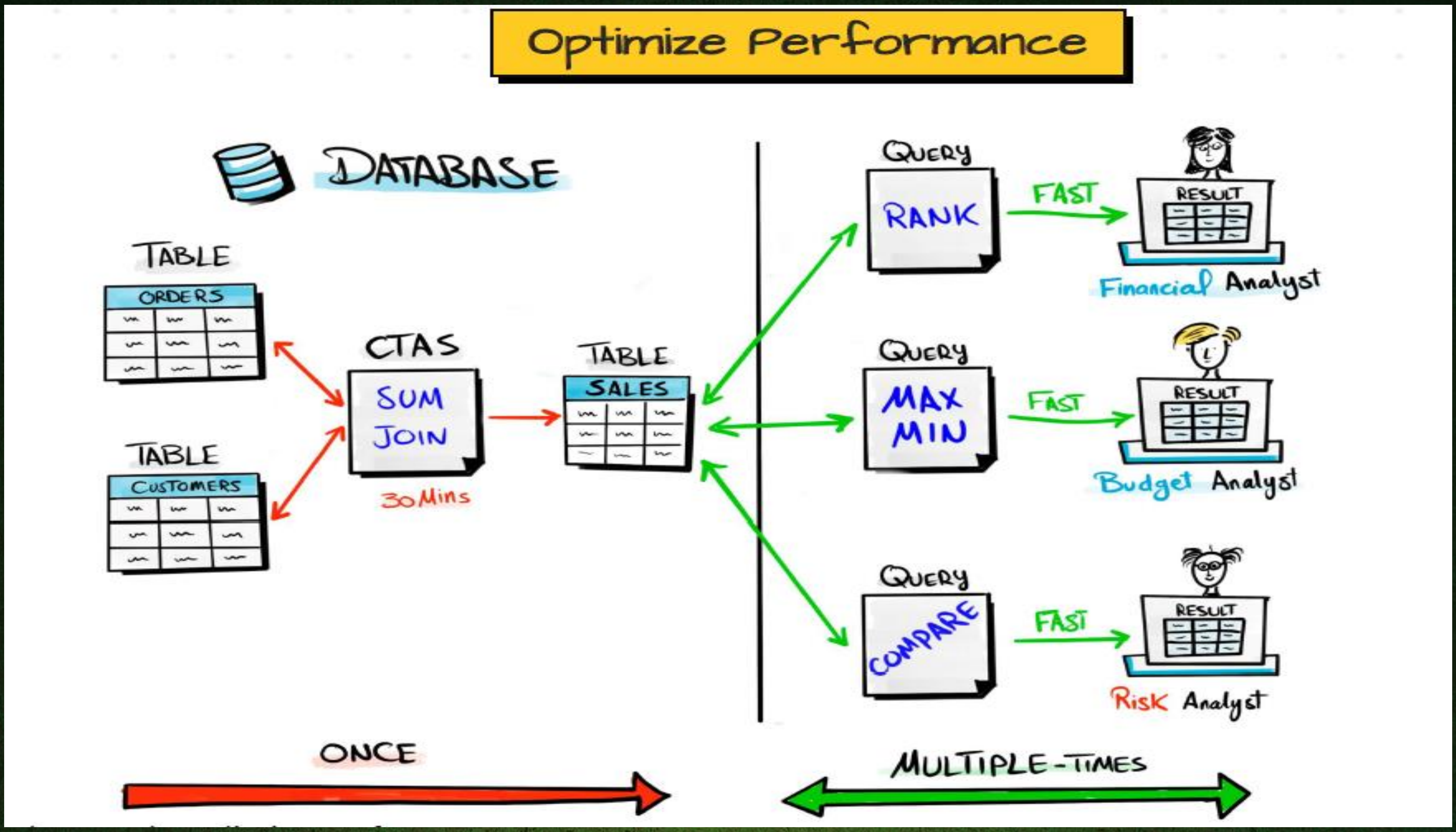
MySQL | Postgres | Oracle

```
SELECT ...
INTO New-Table
FROM ...
WHERE ...
```

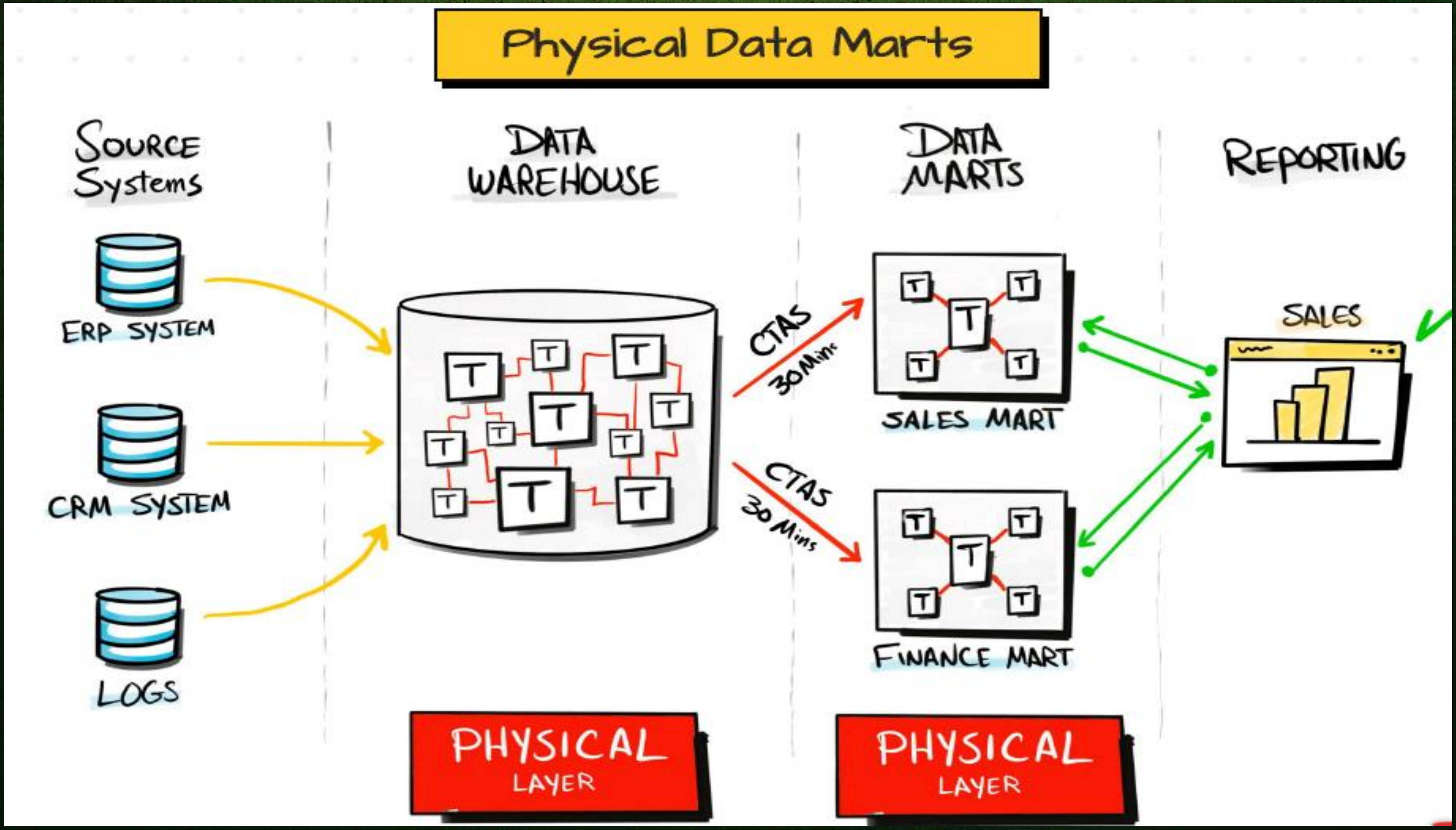
Sql Server



Why We Using :



Persisting the Data Marts of a DWH improves the speed of data retrieval compared to using views



ETL / Staging Use Cases

01

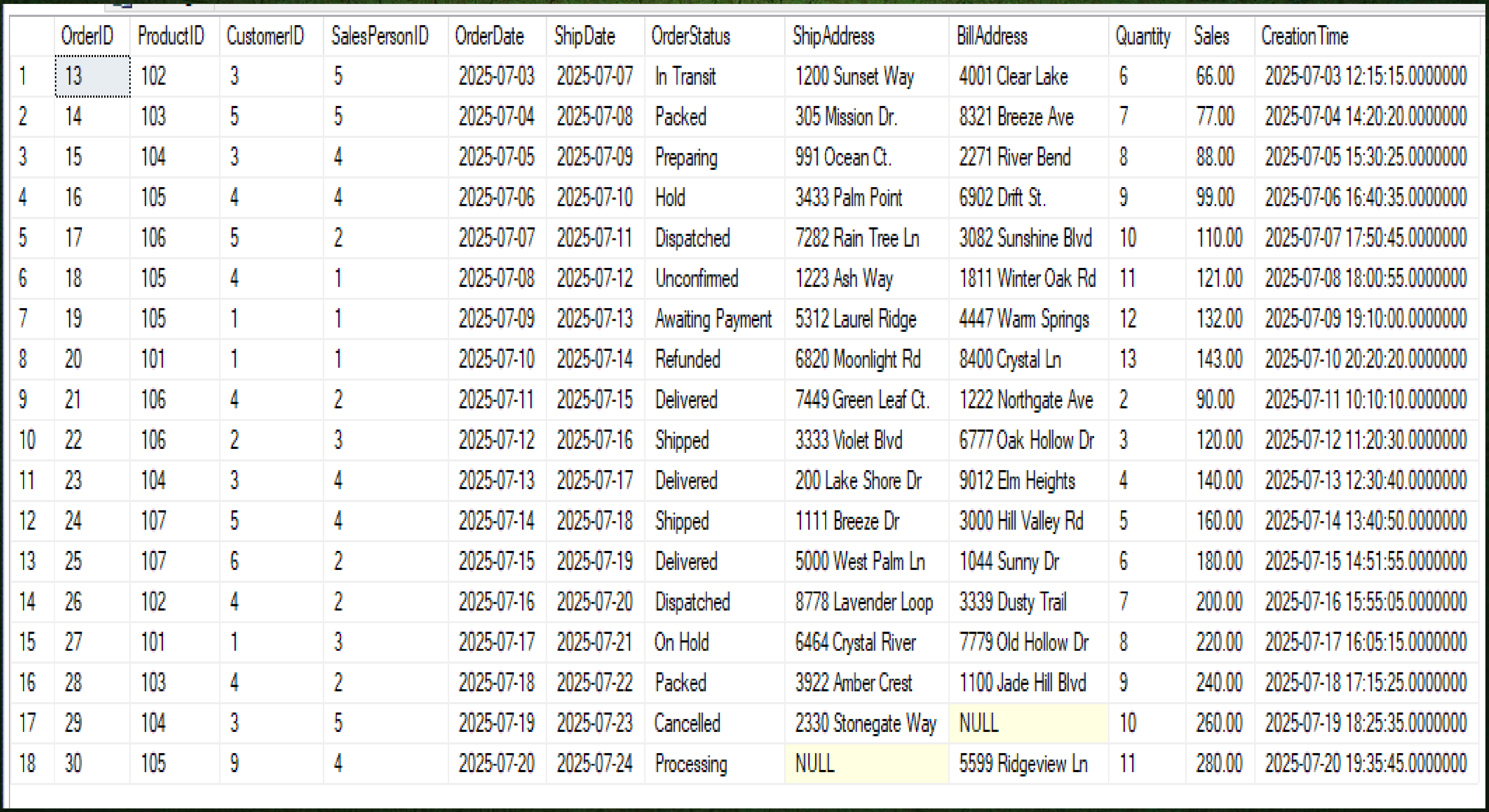
Stage Orders for Last 7 Days (ETL staging)

--Created Like This

```
SELECT *
INTO stage_Orders_Last7Days
FROM orders
WHERE OrderDate >= DATEADD(DAY, -7, GETDATE());
```

--USED BY This Select Query

```
SELECT * FROM stage_Orders_Last7Days
```



	OrderID	ProductID	CustomerID	SalesPersonID	OrderDate	ShipDate	OrderStatus	ShipAddress	BillAddress	Quantity	Sales	CreationTime
1	13	102	3	5	2025-07-03	2025-07-07	In Transit	1200 Sunset Way	4001 Clear Lake	6	66.00	2025-07-03 12:15:15.0000000
2	14	103	5	5	2025-07-04	2025-07-08	Packed	305 Mission Dr.	8321 Breeze Ave	7	77.00	2025-07-04 14:20:20.0000000
3	15	104	3	4	2025-07-05	2025-07-09	Preparing	991 Ocean Ct.	2271 River Bend	8	88.00	2025-07-05 15:30:25.0000000
4	16	105	4	4	2025-07-06	2025-07-10	Hold	3433 Palm Point	6902 Drift St.	9	99.00	2025-07-06 16:40:35.0000000
5	17	106	5	2	2025-07-07	2025-07-11	Dispatched	7282 Rain Tree Ln	3082 Sunshine Blvd	10	110.00	2025-07-07 17:50:45.0000000
6	18	105	4	1	2025-07-08	2025-07-12	Unconfirmed	1223 Ash Way	1811 Winter Oak Rd	11	121.00	2025-07-08 18:00:55.0000000
7	19	105	1	1	2025-07-09	2025-07-13	Awaiting Payment	5312 Laurel Ridge	4447 Warm Springs	12	132.00	2025-07-09 19:10:00.0000000
8	20	101	1	1	2025-07-10	2025-07-14	Refunded	6820 Moonlight Rd	8400 Crystal Ln	13	143.00	2025-07-10 20:20:20.0000000
9	21	106	4	2	2025-07-11	2025-07-15	Delivered	7449 Green Leaf Ct.	1222 Northgate Ave	2	90.00	2025-07-11 10:10:10.0000000
10	22	106	2	3	2025-07-12	2025-07-16	Shipped	3333 Violet Blvd	6777 Oak Hollow Dr	3	120.00	2025-07-12 11:20:30.0000000
11	23	104	3	4	2025-07-13	2025-07-17	Delivered	200 Lake Shore Dr	9012 Elm Heights	4	140.00	2025-07-13 12:30:40.0000000
12	24	107	5	4	2025-07-14	2025-07-18	Shipped	1111 Breeze Dr	3000 Hill Valley Rd	5	160.00	2025-07-14 13:40:50.0000000
13	25	107	6	2	2025-07-15	2025-07-19	Delivered	5000 West Palm Ln	1044 Sunny Dr	6	180.00	2025-07-15 14:51:55.0000000
14	26	102	4	2	2025-07-16	2025-07-20	Dispatched	8778 Lavender Loop	3339 Dusty Trail	7	200.00	2025-07-16 15:55:05.0000000
15	27	101	1	3	2025-07-17	2025-07-21	On Hold	6464 Crystal River	7779 Old Hollow Dr	8	220.00	2025-07-17 16:05:15.0000000
16	28	103	4	2	2025-07-18	2025-07-22	Packed	3922 Amber Crest	1100 Jade Hill Blvd	9	240.00	2025-07-18 17:15:25.0000000
17	29	104	3	5	2025-07-19	2025-07-23	Cancelled	2330 Stonegate Way	NULL	10	260.00	2025-07-19 18:25:35.0000000
18	30	105	9	4	2025-07-20	2025-07-24	Processing	NULL	5599 Ridgeview Ln	11	280.00	2025-07-20 19:35:45.0000000

Use by: Use for transforming only recent orders in ETL pipelines.

AnshLibrary

02

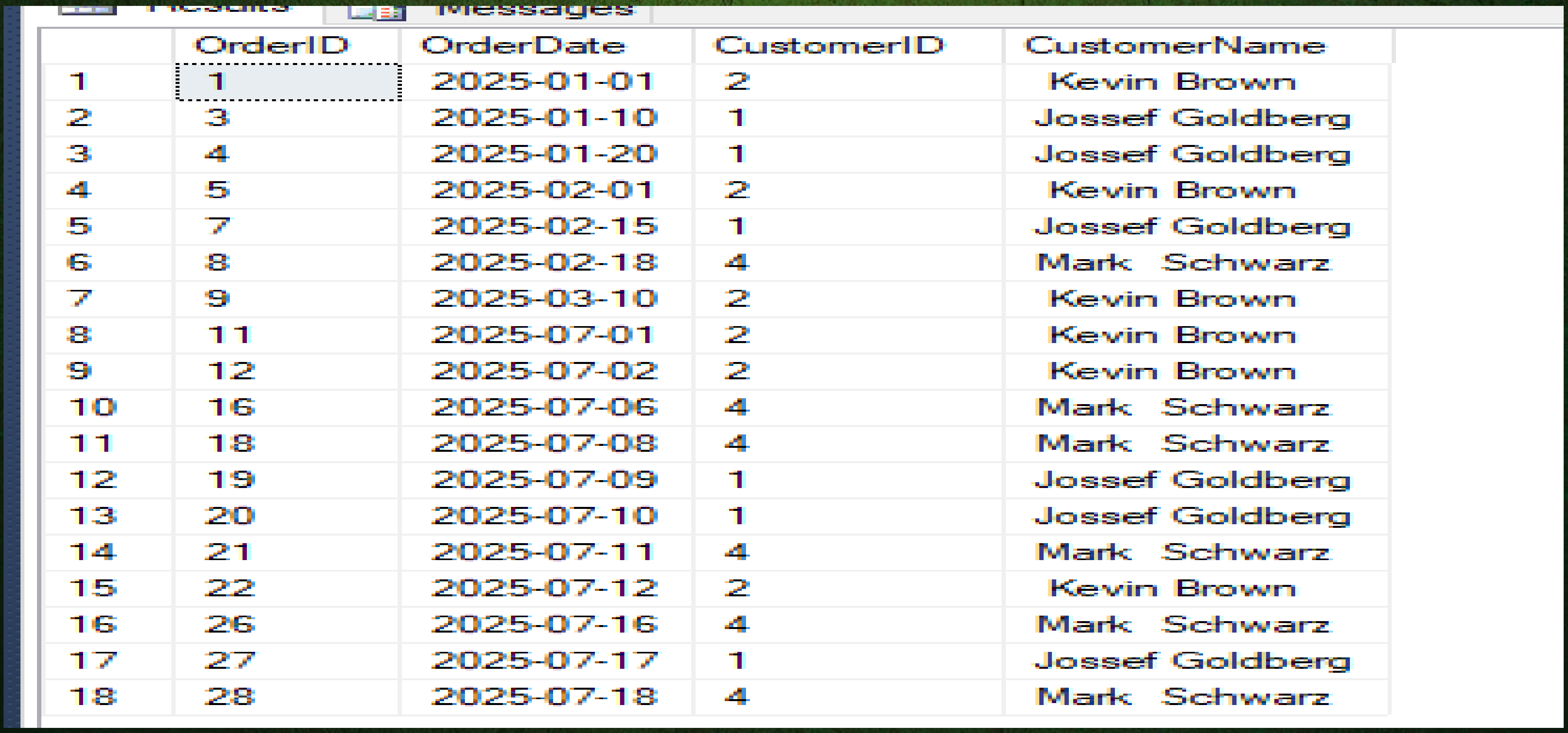
Join Orders with Customers (Data Enrichment)

```
SELECT
    o.OrderID,
    o.OrderDate,
    o.CustomerID,
    CONCAT(c.First_Name, ' ', c.Last_Name) AS CustomerName

INTO enriched_OrdersWithCustomers
FROM Orders o
JOIN Customers c ON o.CustomerID = c.Customer_id;

---------------------------------------------------------------------------------------------------------

SELECT * FROM enriched_OrdersWithCustomers
WHERE CustomerID IN (1,2,4)
```



	OrderID	OrderDate	CustomerID	CustomerName
1	1	2025-01-01	2	Kevin Brown
2	3	2025-01-10	1	Jossef Goldberg
3	4	2025-01-20	1	Jossef Goldberg
4	5	2025-02-01	2	Kevin Brown
5	7	2025-02-15	1	Jossef Goldberg
6	8	2025-02-18	4	Mark Schwarz
7	9	2025-03-10	2	Kevin Brown
8	11	2025-07-01	2	Kevin Brown
9	12	2025-07-02	2	Kevin Brown
10	16	2025-07-06	4	Mark Schwarz
11	18	2025-07-08	4	Mark Schwarz
12	19	2025-07-09	1	Jossef Goldberg
13	20	2025-07-10	1	Jossef Goldberg
14	21	2025-07-11	4	Mark Schwarz
15	22	2025-07-12	2	Kevin Brown
16	26	2025-07-16	4	Mark Schwarz
17	27	2025-07-17	1	Jossef Goldberg
18	28	2025-07-18	4	Mark Schwarz

Use by: Prepares enriched records for reporting/BI

AnshLibrary

Reporting & Analytics Use Cases

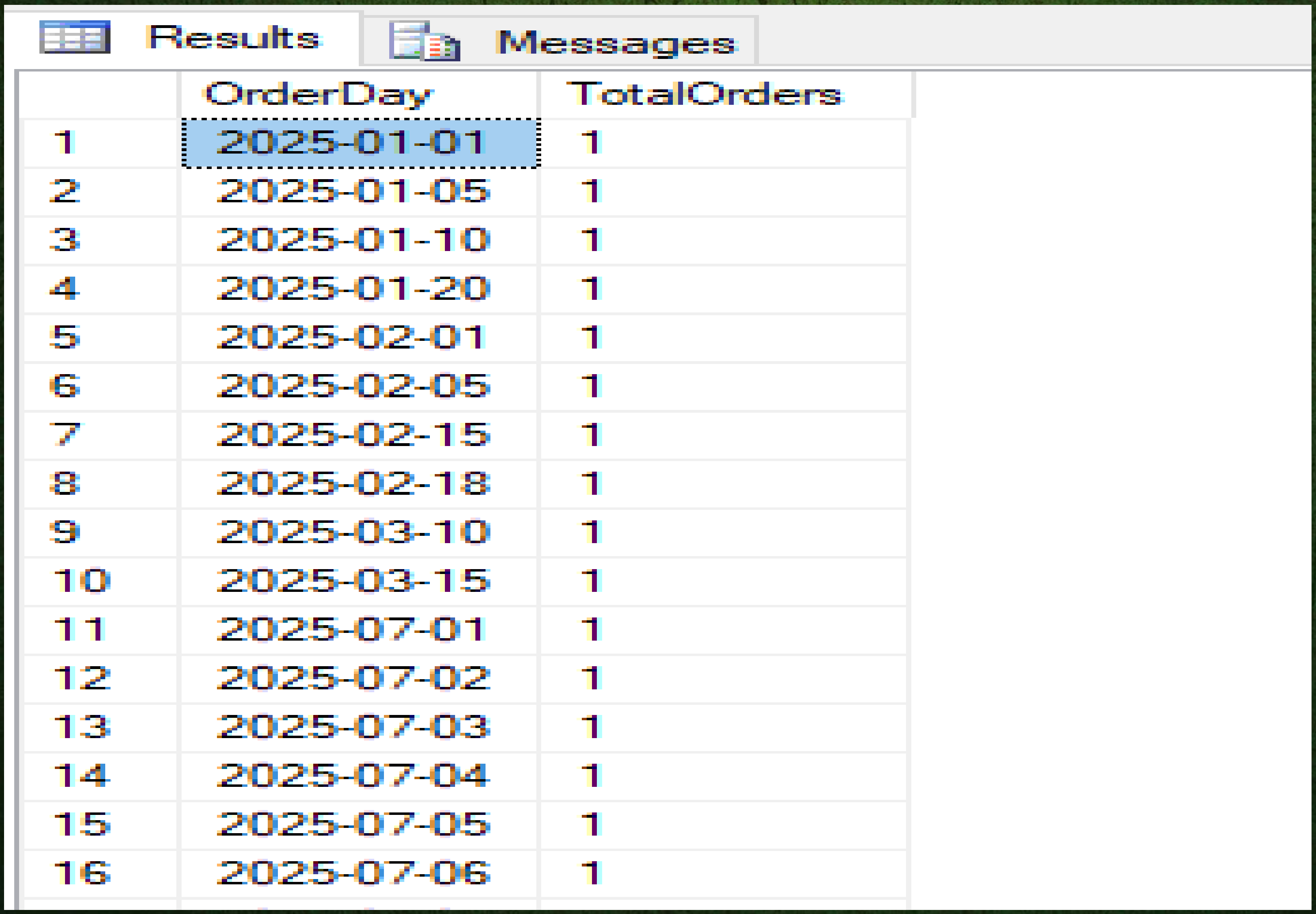
03

Daily Order Summary

```sql
SELECT
  CAST(OrderDate AS DATE) AS OrderDay,
  COUNT(*) AS TotalOrders

INTO rpt_DailyOrderSummary
FROM Orders
GROUP BY CAST(OrderDate AS DATE);

-------------------------------------------------------------
SELECT * FROM rpt_DailyOrderSummary
```



	OrderDay	TotalOrders
1	2025-01-01	1
2	2025-01-05	1
3	2025-01-10	1
4	2025-01-20	1
5	2025-02-01	1
6	2025-02-05	1
7	2025-02-15	1
8	2025-02-18	1
9	2025-03-10	1
10	2025-03-15	1
11	2025-07-01	1
12	2025-07-02	1
13	2025-07-03	1
14	2025-07-04	1
15	2025-07-05	1
16	2025-07-06	1

Use by: Daily operational dashboard or Power BI source.

AnshLibrary

04

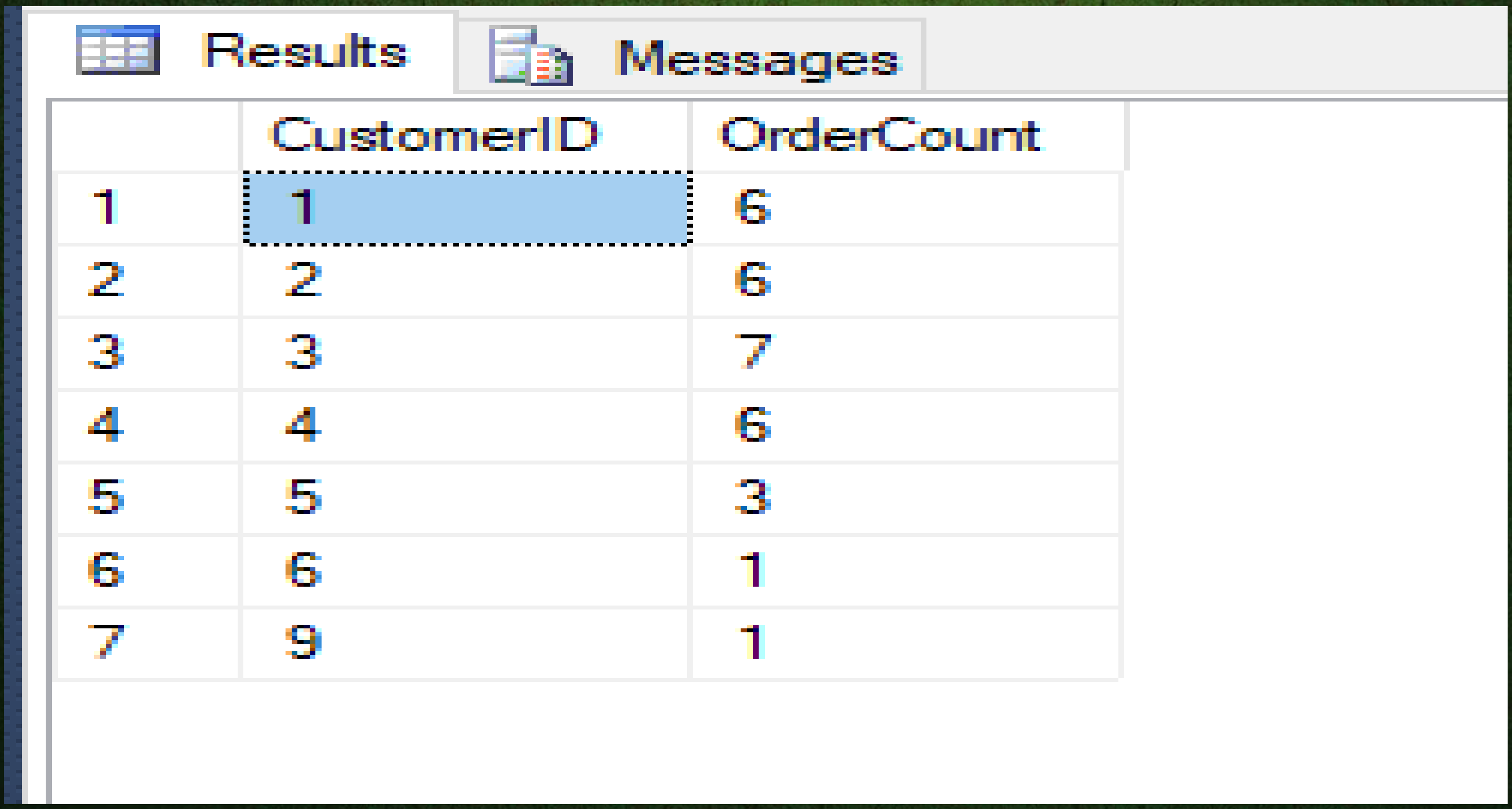
Customer Order Frequency

```
SELECT

    CustomerID,
    COUNT(*) AS OrderCount

INTO rpt_CustomerOrderFrequency
FROM Orders
GROUP BY CustomerID;

------------------------------------------------------------
SELECT * FROM rpt_CustomerOrderFrequency
```



	CustomerID	OrderCount
1	1	6
2	2	6
3	3	7
4	4	6
5	5	3
6	6	1
7	9	1

Use by: For loyalty scoring, churn analysis, or CRM integration

AnshLibrary

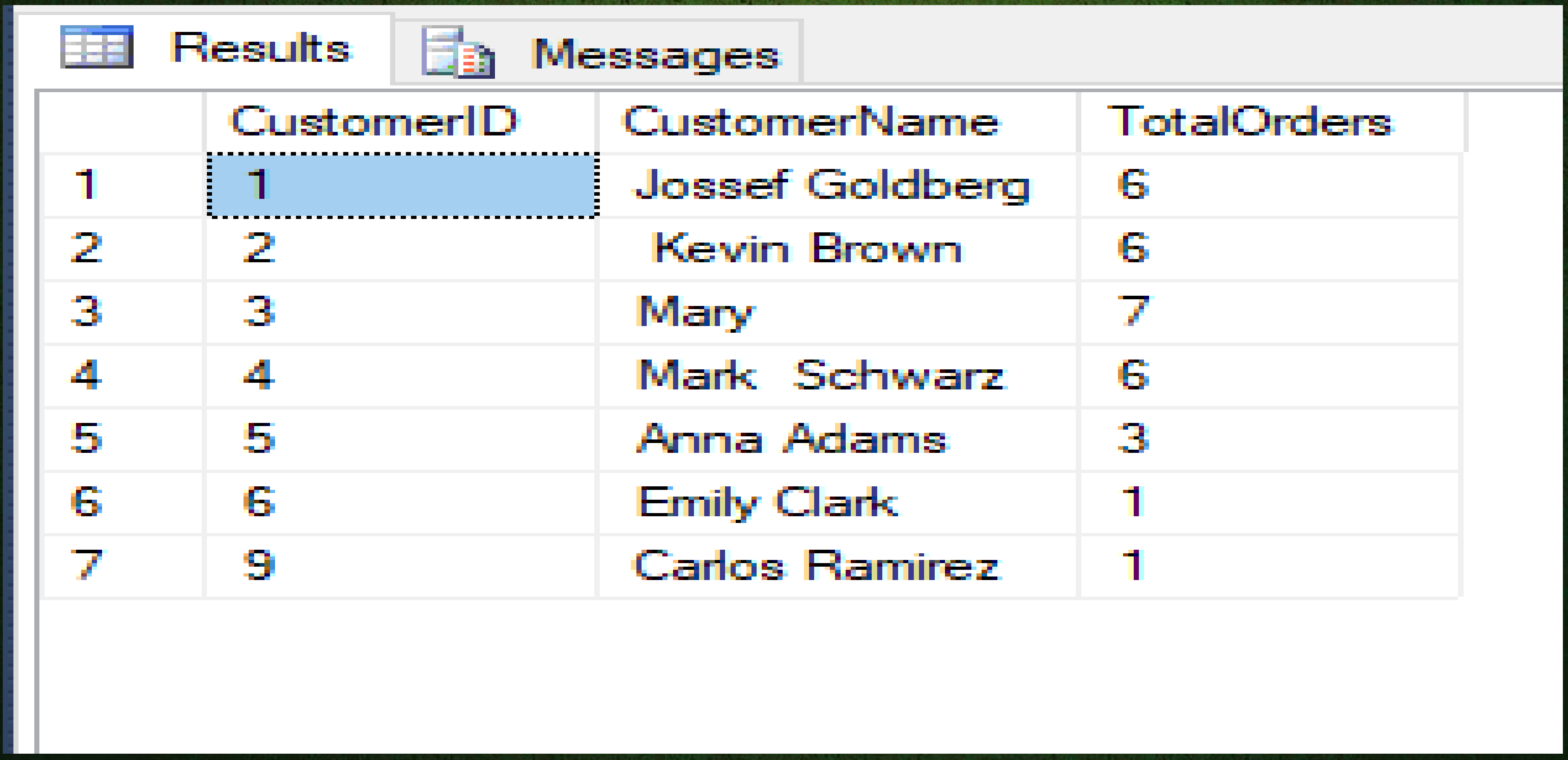
05

Customer-wise Total Orders

```
SELECT
    o.CustomerID,
    CONCAT(c.First_Name, ' ', c.Last_Name) AS CustomerName,
    COUNT(o.OrderID) AS TotalOrders

INTO rpt_EmployeeOrderStats
FROM Orders o
JOIN Customers c ON o.CustomerID = c.Customer_id
GROUP BY o.CustomerID, c.First_Name, c.Last_Name;

--------------------------------------------------------------------------------
SELECT * FROM rpt_EmployeeOrderStats
```



	CustomerID	CustomerName	TotalOrders
1	1	Jossef Goldberg	6
2	2	Kevin Brown	6
3	3	Mary	7
4	4	Mark Schwarz	6
5	5	Anna Adams	3
6	6	Emily Clark	1
7	9	Carlos Ramirez	1

Used For :HR dashboard or performance tracking.

AnshLibrary

06

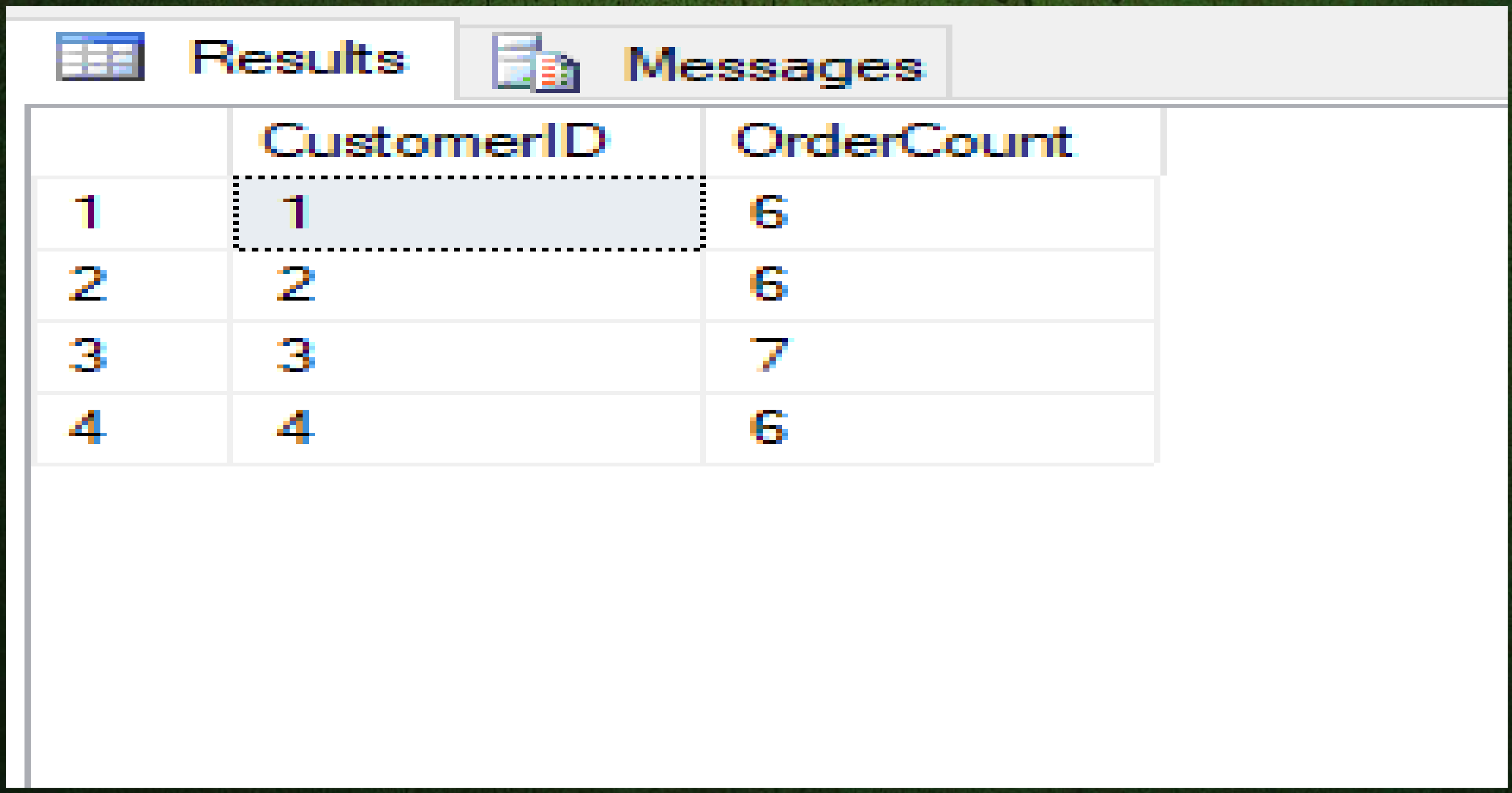
High-Value Customers (based on order count)

```
SELECT
    CustomerID,
    COUNT(OrderID) AS OrderCount

INTO rpt_HighValueCustomers
FROM Orders
GROUP BY CustomerID

----------------------------------------------------------------

SELECT * FROM rpt_HighValueCustomers
Where OrderCount >4
```



	CustomerID	OrderCount
1	1	6
2	2	6
3	3	7
4	4	6

Use by: For marketing or upsell targeting

AnshLibrary

Archival & Backup Use Cases

07

Copy of Orders Before Deleting From Live Table

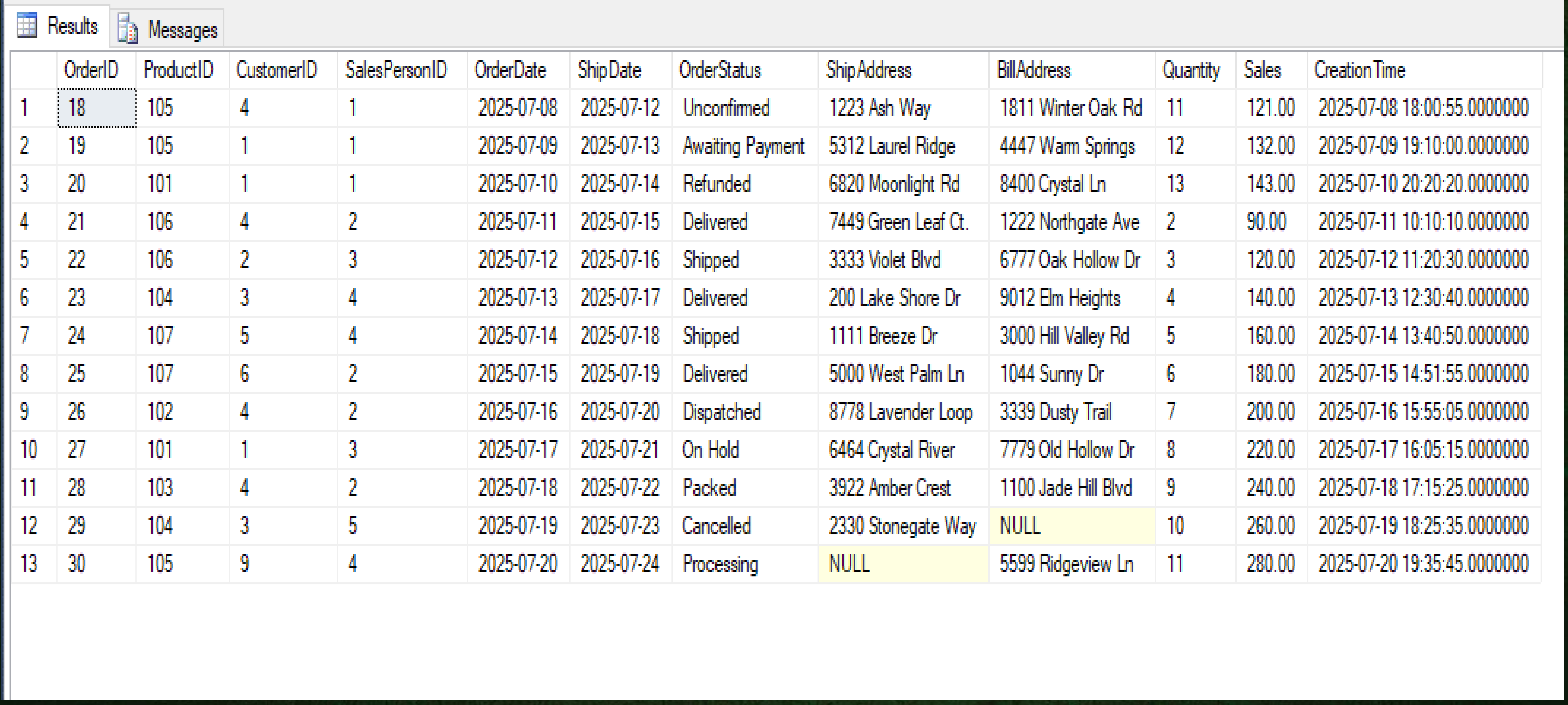
```
SELECT *

INTO backup_Orders_ToDelete

FROM Orders
WHERE OrderID NOT IN (SELECT OrderID FROM
OrdersArchive);

------------------------------------------------------------------------

SELECT * FROM backup_Orders_ToDelete
```



	OrderID	ProductID	CustomerID	SalesPersonID	OrderDate	ShipDate	OrderStatus	ShipAddress	BillAddress	Quantity	Sales	CreationTime
1	18	105	4	1	2025-07-08	2025-07-12	Unconfirmed	1223 Ash Way	1811 Winter Oak Rd	11	121.00	2025-07-08 18:00:55.0000000
2	19	105	1	1	2025-07-09	2025-07-13	Awaiting Payment	5312 Laurel Ridge	4447 Warm Springs	12	132.00	2025-07-09 19:10:00.0000000
3	20	101	1	1	2025-07-10	2025-07-14	Refunded	6820 Moonlight Rd	8400 Crystal Ln	13	143.00	2025-07-10 20:20:20.0000000
4	21	106	4	2	2025-07-11	2025-07-15	Delivered	7449 Green Leaf Ct.	1222 Northgate Ave	2	90.00	2025-07-11 10:10:10.0000000
5	22	106	2	3	2025-07-12	2025-07-16	Shipped	3333 Violet Blvd	6777 Oak Hollow Dr	3	120.00	2025-07-12 11:20:30.0000000
6	23	104	3	4	2025-07-13	2025-07-17	Delivered	200 Lake Shore Dr	9012 Elm Heights	4	140.00	2025-07-13 12:30:40.0000000
7	24	107	5	4	2025-07-14	2025-07-18	Shipped	1111 Breeze Dr	3000 Hill Valley Rd	5	160.00	2025-07-14 13:40:50.0000000
8	25	107	6	2	2025-07-15	2025-07-19	Delivered	5000 West Palm Ln	1044 Sunny Dr	6	180.00	2025-07-15 14:51:55.0000000
9	26	102	4	2	2025-07-16	2025-07-20	Dispatched	8778 Lavender Loop	3339 Dusty Trail	7	200.00	2025-07-16 15:55:05.0000000
10	27	101	1	3	2025-07-17	2025-07-21	On Hold	6464 Crystal River	7779 Old Hollow Dr	8	220.00	2025-07-17 16:05:15.0000000
11	28	103	4	2	2025-07-18	2025-07-22	Packed	3922 Amber Crest	1100 Jade Hill Blvd	9	240.00	2025-07-18 17:15:25.0000000
12	29	104	3	5	2025-07-19	2025-07-23	Cancelled	2330 Stonegate Way	NULL	10	260.00	2025-07-19 18:25:35.0000000
13	30	105	9	4	2025-07-20	2025-07-24	Processing	NULL	5599 Ridgeview Ln	11	280.00	2025-07-20 19:35:45.0000000

Use by: Used before purging data.

AnshLibrary

08

Copy Current Month Orders to Archive Table Format

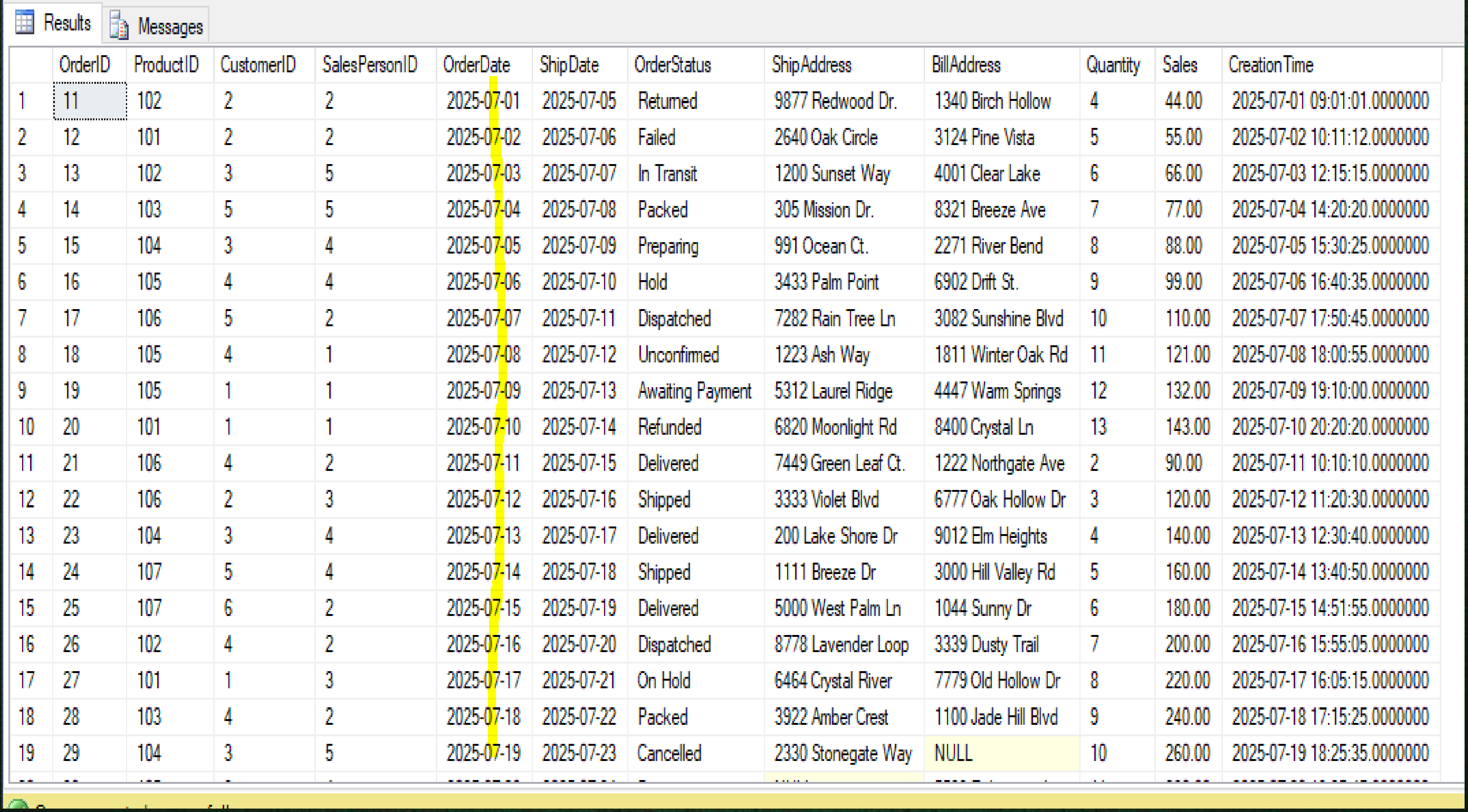
```
SELECT *

INTO OrdersArchive_July

FROM Orders
WHERE MONTH(OrderDate) = 7 AND YEAR(OrderDate)
= YEAR(GETDATE());

-----------------------------------------------------------------------------

SELECT * FROM OrdersArchive_July
```



	OrderID	ProductID	CustomerID	SalesPersonID	OrderDate	ShipDate	OrderStatus	ShipAddress	BillAddress	Quantity	Sales	CreationTime
1	11	102	2	2	2025-07-01	2025-07-05	Returned	9877 Redwood Dr.	1340 Birch Hollow	4	44.00	2025-07-01 09:01:01.0000000
2	12	101	2	2	2025-07-02	2025-07-06	Failed	2640 Oak Circle	3124 Pine Vista	5	55.00	2025-07-02 10:11:12.0000000
3	13	102	3	5	2025-07-03	2025-07-07	In Transit	1200 Sunset Way	4001 Clear Lake	6	66.00	2025-07-03 12:15:15.0000000
4	14	103	5	5	2025-07-04	2025-07-08	Packed	305 Mission Dr.	8321 Breeze Ave	7	77.00	2025-07-04 14:20:20.0000000
5	15	104	3	4	2025-07-05	2025-07-09	Preparing	991 Ocean Ct.	2271 River Bend	8	88.00	2025-07-05 15:30:25.0000000
6	16	105	4	4	2025-07-06	2025-07-10	Hold	3433 Palm Point	6902 Drift St.	9	99.00	2025-07-06 16:40:35.0000000
7	17	106	5	2	2025-07-07	2025-07-11	Dispatched	7282 Rain Tree Ln	3082 Sunshine Blvd	10	110.00	2025-07-07 17:50:45.0000000
8	18	105	4	1	2025-07-08	2025-07-12	Unconfirmed	1223 Ash Way	1811 Winter Oak Rd	11	121.00	2025-07-08 18:00:55.0000000
9	19	105	1	1	2025-07-09	2025-07-13	Awaiting Payment	5312 Laurel Ridge	4447 Warm Springs	12	132.00	2025-07-09 19:10:00.0000000
10	20	101	1	1	2025-07-10	2025-07-14	Refunded	6820 Moonlight Rd	8400 Crystal Ln	13	143.00	2025-07-10 20:20:20.0000000
11	21	106	4	2	2025-07-11	2025-07-15	Delivered	7449 Green Leaf Ct.	1222 Northgate Ave	2	90.00	2025-07-11 10:10:10.0000000
12	22	106	2	3	2025-07-12	2025-07-16	Shipped	3333 Violet Blvd	6777 Oak Hollow Dr	3	120.00	2025-07-12 11:20:30.0000000
13	23	104	3	4	2025-07-13	2025-07-17	Delivered	200 Lake Shore Dr	9012 Elm Heights	4	140.00	2025-07-13 12:30:40.0000000
14	24	107	5	4	2025-07-14	2025-07-18	Shipped	1111 Breeze Dr	3000 Hill Valley Rd	5	160.00	2025-07-14 13:40:50.0000000
15	25	107	6	2	2025-07-15	2025-07-19	Delivered	5000 West Palm Ln	1044 Sunny Dr	6	180.00	2025-07-15 14:51:55.0000000
16	26	102	4	2	2025-07-16	2025-07-20	Dispatched	8778 Lavender Loop	3339 Dusty Trail	7	200.00	2025-07-16 15:55:05.0000000
17	27	101	1	3	2025-07-17	2025-07-21	On Hold	6464 Crystal River	7779 Old Hollow Dr	8	220.00	2025-07-17 16:05:15.0000000
18	28	103	4	2	2025-07-18	2025-07-22	Packed	3922 Amber Crest	1100 Jade Hill Blvd	9	240.00	2025-07-18 17:15:25.0000000
19	29	104	3	5	2025-07-19	2025-07-23	Cancelled	2330 Stonegate Way	NULL	10	260.00	2025-07-19 18:25:35.0000000

Using: Use in a monthly ETL job or scheduled archival process.

AnshLibrary

Auditing & Quality Checks

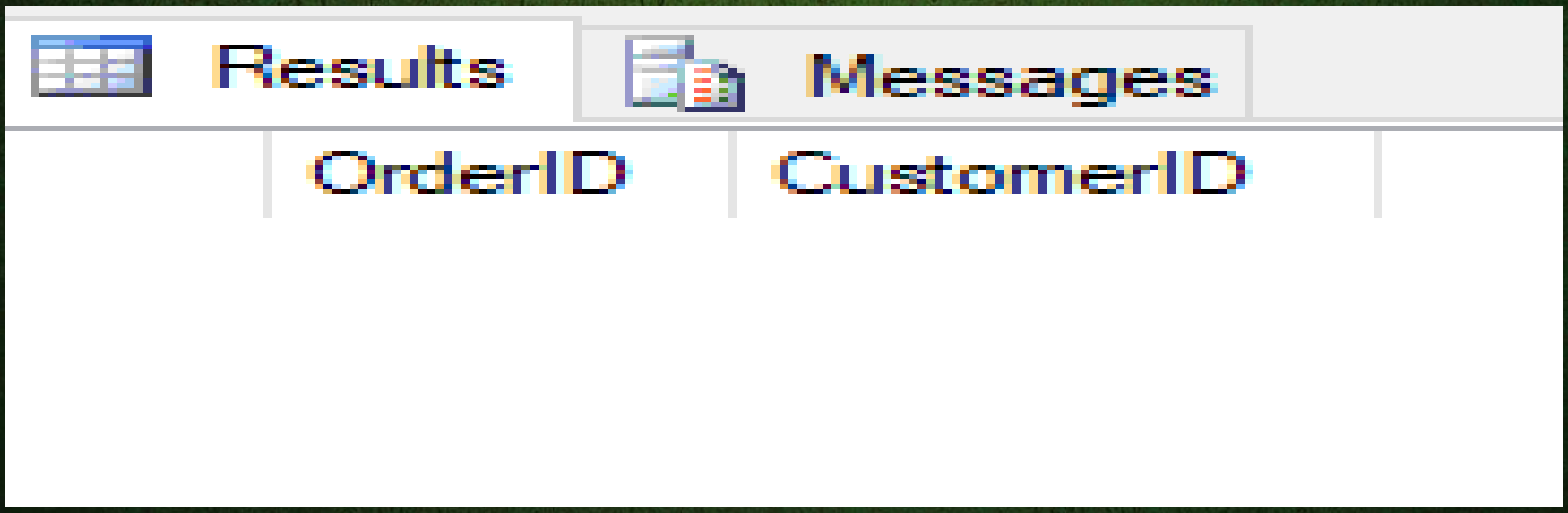
09

Identify Orphan Orders (No Matching Customer)

```
SELECT
   o.OrderID,
   o.CustomerID
INTO audit_OrdersOrphanCustomers
FROM Orders o
LEFT JOIN Customers c ON o.CustomerID =
c.Customer_id
WHERE c.Customer_id IS NULL;

--------------------------------------------------------------

Select * from audit_OrdersOrphanCustomers
```



	OrderID	CustomerID

Use by: Data integrity validation.

AnshLibrary

Utility & Development

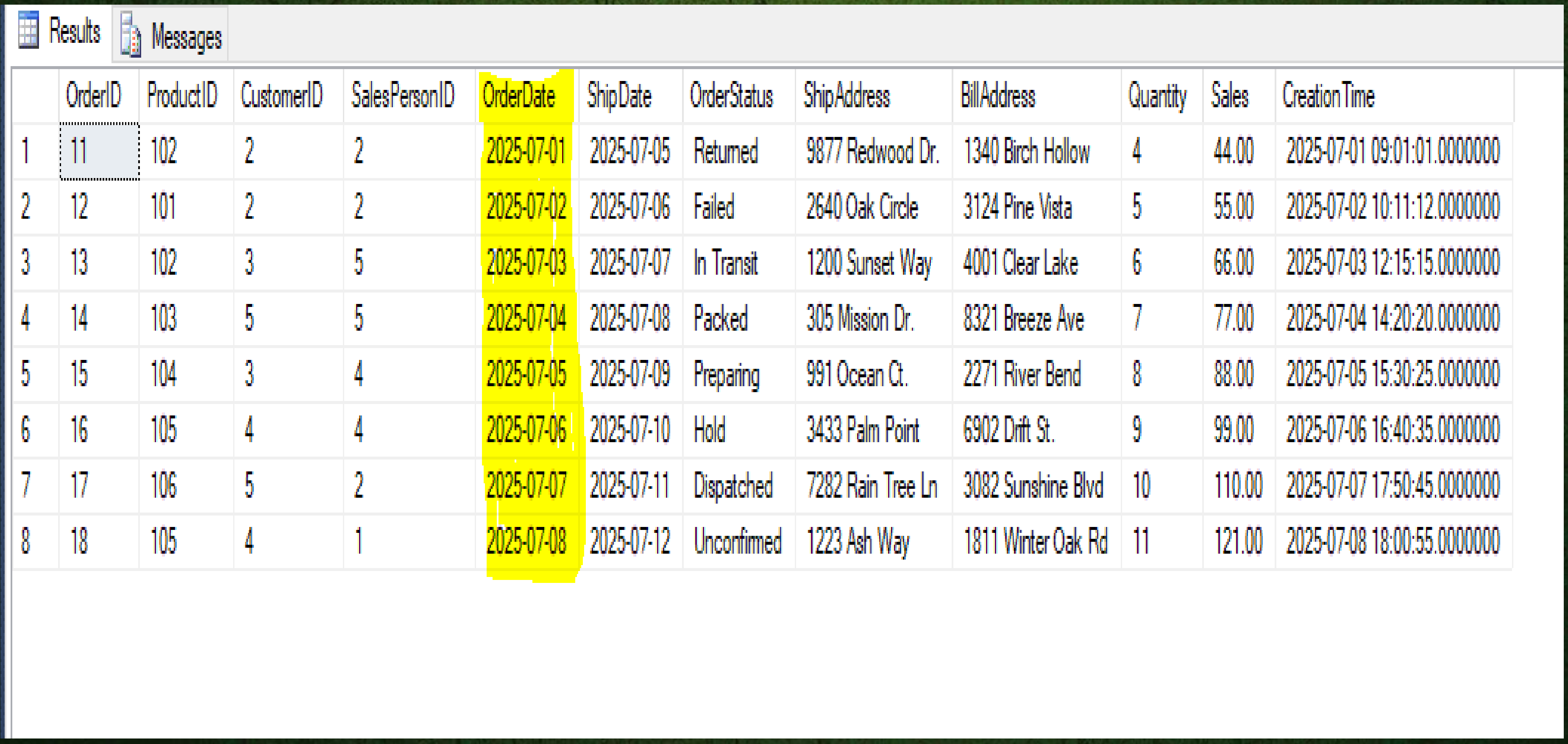
10

Snapshot Orders for Dev/Testing

```
SELECT *
INTO dev_SnapshotOrders
FROM Orders
WHERE OrderDate >= '2025-07-01' AND
OrderDate <= '2025-07-08’;

------------------------------------------------------------------

SELECT * FROM dev_SnapshotOrders
```



	OrderID	ProductID	CustomerID	SalesPersonID	OrderDate	ShipDate	OrderStatus	ShipAddress	BillAddress	Quantity	Sales	CreationTime
1	11	102	2	2	2025-07-01	2025-07-05	Returned	9877 Redwood Dr.	1340 Birch Hollow	4	44.00	2025-07-01 09:01:01.0000000
2	12	101	2	2	2025-07-02	2025-07-06	Failed	2640 Oak Circle	3124 Pine Vista	5	55.00	2025-07-02 10:11:12.0000000
3	13	102	3	5	2025-07-03	2025-07-07	In Transit	1200 Sunset Way	4001 Clear Lake	6	66.00	2025-07-03 12:15:15.0000000
4	14	103	5	5	2025-07-04	2025-07-08	Packed	305 Mission Dr.	8321 Breeze Ave	7	77.00	2025-07-04 14:20:20.0000000
5	15	104	3	4	2025-07-05	2025-07-09	Preparing	991 Ocean Ct.	2271 River Bend	8	88.00	2025-07-05 15:30:25.0000000
6	16	105	4	4	2025-07-06	2025-07-10	Hold	3433 Palm Point	6902 Drift St.	9	99.00	2025-07-06 16:40:35.0000000
7	17	106	5	2	2025-07-07	2025-07-11	Dispatched	7282 Rain Tree Ln	3082 Sunshine Blvd	10	110.00	2025-07-07 17:50:45.0000000
8	18	105	4	1	2025-07-08	2025-07-12	Unconfirmed	1223 Ash Way	1811 Winter Oak Rd	11	121.00	2025-07-08 18:00:55.0000000

Use by: Sample data for non-prod testing.

AnshLibrary

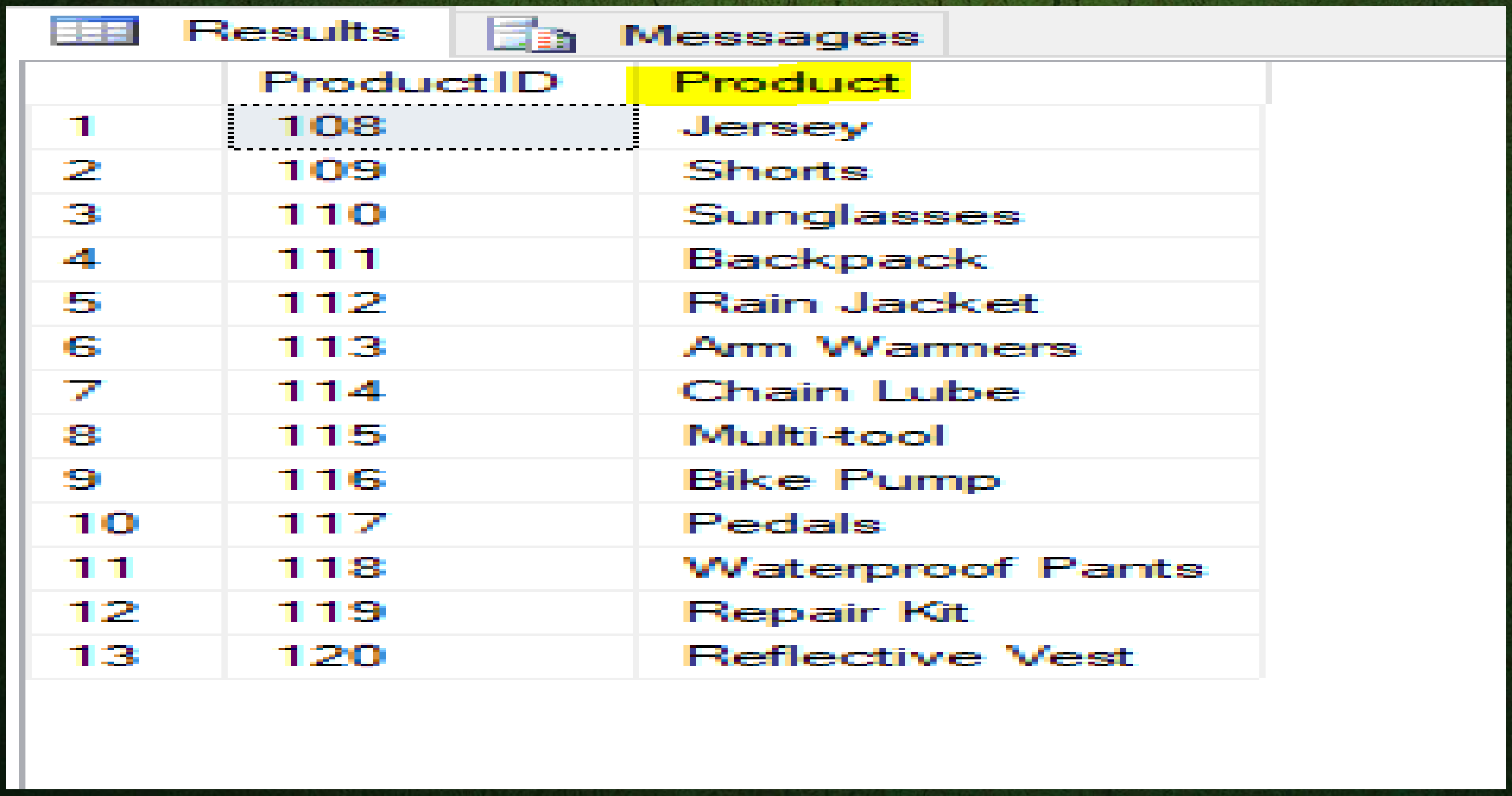
11

Products Without Orders (Zero Sales)

```
SELECT
    p.ProductID,
    p.Product
INTO rpt_UnsoldProducts
FROM Products p
LEFT JOIN OrdersArchive oi
ON p.ProductID = oi.ProductID
WHERE oi.ProductID IS NULL;

------------------------------------------------------------

select * from rpt_UnsoldProducts
```



	ProductID	Product
1	108	Jersey
2	109	Shorts
3	110	Sunglasses
4	111	Backpack
5	112	Rain Jacket
6	113	Arm Warmers
7	114	Chain Lube
8	115	Multi-tool
9	116	Bike Pump
10	117	Pedals
11	118	Waterproof Pants
12	119	Repair Kit
13	120	Reflective Vest

Use Case: Detect inventory that hasn't moved..

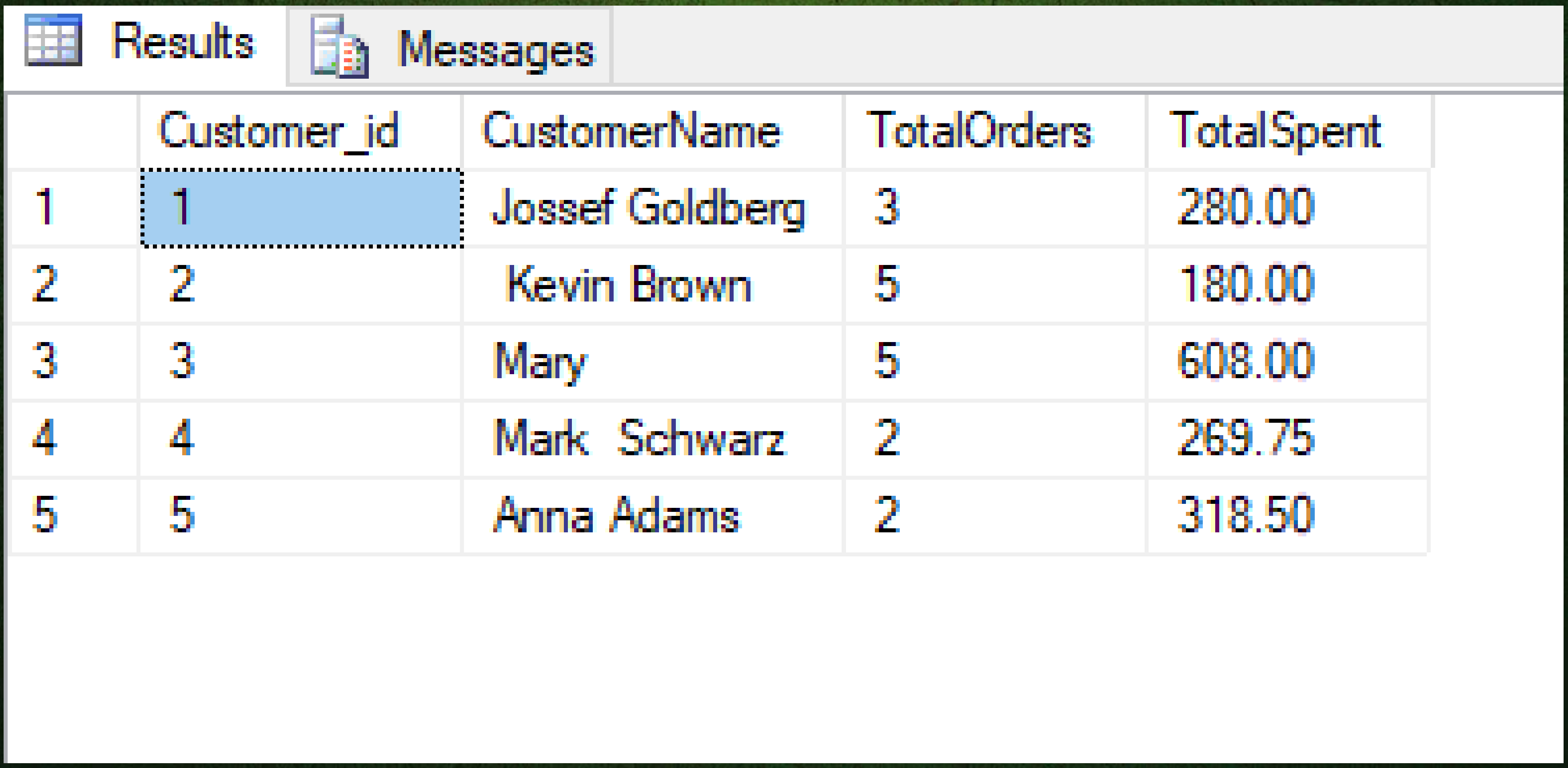
AnshLibrary

12

Generate Customer Order Summary

```
SELECT
    c.Customer_id,
    CONCAT(c.First_Name, ' ', c.Last_Name) AS CustomerName,
    COUNT(DISTINCT o.OrderID) AS TotalOrders,
    SUM(oi.Quantity * oi.Sales) AS TotalSpent
INTO rpt_CustomerOrderSummary
FROM customers c
JOIN orders o ON c.Customer_id = o.CustomerID
JOIN OrdersArchive oi ON o.OrderID = oi.OrderID
GROUP BY c.Customer_id, c.First_Name, c.Last_Name;

---------------------------------------------------------------------------------
select * from rpt_CustomerOrderSummary
```



	Customer_id	CustomerName	TotalOrders	TotalSpent
1	1	Jossef Goldberg	3	280.00
2	2	Kevin Brown	5	180.00
3	3	Mary	5	608.00
4	4	Mark Schwarz	2	269.75
5	5	Anna Adams	2	318.50

Use Case: Combines total orders and value per customer.

AnshLibrary