

Set Operators: UNION, UNION ALL, INTERSECT & EXCEPT

Stack and compare query results. Append data with UNION ALL, remove duplicates with UNION, find common rows with INTERSECT and differences with EXCEPT, follow the rules for compatible columns, and know when to use a set operator instead of a join.

● SQL class 4

✦ SQL Server 2022, 2026

Intermediate

🕒 80 min

By the end of this lesson you can

- Explain the difference between joins (add columns) and set operators (add or compare rows)
- Append results with UNION ALL and de-duplicate with UNION
- Find common rows with INTERSECT and missing rows with EXCEPT
- Apply the rules: same number of columns, compatible types, names from the first query, one ORDER BY at the end
- Use set operators for reconciliation and data-quality checks

Datasets used

superstore_sales.csv

customers.csv

Joins add columns; set operators add rows

Join vs set operator

JOIN

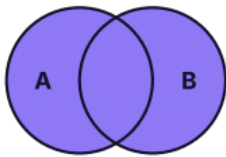
- Combines tables horizontally
- Matches rows on a key
- Result has columns from both tables
- Example: orders + customer details

Set operator

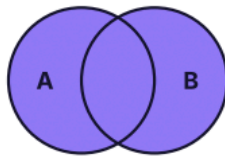
- Combines results vertically
- Compares whole rows
- Same columns in both queries
- Example: 2024 customers ☒ 2025 customers

A join puts tables side by side (more columns). A set operator stacks or compares the rows of two query results with the same columns.

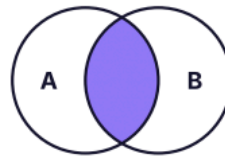
UNION ALL (keeps duplicates)



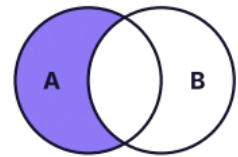
UNION (distinct)



INTERSECT



EXCEPT (A minus B)



The four set operators. A and B are the result sets of two queries.

Rules for set operators

1. Both queries must return the **same number of columns**.
2. Corresponding columns must have **compatible data types** (text with text, numbers with numbers).
3. Column **names come from the first query**.
4. Only **one ORDER BY**, at the very end, applies to the combined result.
5. **UNION**, **INTERSECT** and **EXCEPT** remove duplicates; **UNION ALL** does not.

UNION ALL: append

`UNION ALL` stacks results exactly as they are, like Power Query's Append or pasting one table under another. It is the fastest set operator because it does not check for duplicates.

SQL

```
SELECT 'Furniture' AS source, COUNT(*) AS lines, ROUND(SUM(sales), 0) AS sales FROM
superstore WHERE category = 'Furniture'
UNION ALL
SELECT 'Technology', COUNT(*), ROUND(SUM(sales), 0) FROM superstore WHERE category =
'Technology'
UNION ALL
SELECT 'Office Supplies', COUNT(*), ROUND(SUM(sales), 0) FROM superstore WHERE category =
'Office Supplies'
ORDER BY sales DESC;
```

RESULT

source	lines	sales
Technology	1892	156,438,413
Furniture	1917	34,841,853
Office Supplies	2514	10,133,304

3 rows × 3 columns

A very common real use: combining monthly or regional tables with the same structure.

SQL

```
SELECT * FROM sales_jan_2026
UNION ALL
SELECT * FROM sales_feb_2026
UNION ALL
SELECT * FROM sales_mar_2026;
```

UNION: append and remove duplicates

SQL

```
SELECT COUNT(*) AS cities_union_all
FROM (SELECT city FROM customers WHERE segment = 'Corporate'
      UNION ALL
      SELECT city FROM customers WHERE segment = 'Home Office') AS t;
```

RESULT

cities_union_all

407

1 rows × 1 columns

SQL

```
SELECT COUNT(*) AS cities_union
FROM (SELECT city FROM customers WHERE segment = 'Corporate'
      UNION
      SELECT city FROM customers WHERE segment = 'Home Office') AS t;
```

RESULT

cities_union

16

1 rows × 1 columns

UNION ALL returned one row per customer; UNION returned each city once.

TIP UNION or UNION ALL?

Use UNION ALL unless you **need** duplicates removed. UNION has to sort or hash the whole result to find duplicates, which is slow on large data, and it can silently remove rows that are genuinely repeated (two identical order lines).

INTERSECT: rows in both

"Which customers ordered in **both** 2024 and 2025?" (retained customers)

SQL

```
SELECT COUNT(*) AS customers_in_both_years
FROM (
  SELECT customer_id FROM orders WHERE YEAR(order_date) = 2024
  INTERSECT
  SELECT customer_id FROM orders WHERE YEAR(order_date) = 2025
) AS t;
```

RESULT

customers_in_both_years

467

1 rows × 1 columns

EXCEPT: rows in the first but not the second

"Which 2024 customers did **not** come back in 2025?" (churned customers)

SQL

```
SELECT COUNT(*) AS customers_lost_after_2024
FROM (
    SELECT customer_id FROM orders WHERE YEAR(order_date) = 2024
    EXCEPT
    SELECT customer_id FROM orders WHERE YEAR(order_date) = 2025
) AS t;
```

RESULT

customers_lost_after_2024

123

1 rows × 1 columns

"Which 2025 customers are **new** (did not order in 2024 or 2023)?"

SQL

```
SELECT COUNT(*) AS new_customers_2025
FROM (
    SELECT customer_id FROM orders WHERE YEAR(order_date) = 2025
    EXCEPT
    SELECT customer_id FROM orders WHERE YEAR(order_date) < 2025
) AS t;
```

RESULT

new_customers_2025

52

1 rows × 1 columns

Order matters for EXCEPT: A EXCEPT B ≠ B EXCEPT A. Oracle calls it MINUS.

Let's put the customer movement together in one table:

SQL

```
SELECT 'Ordered in 2024' AS measure, COUNT(DISTINCT customer_id) AS customers FROM orders
WHERE YEAR(order_date) = 2024
UNION ALL
SELECT 'Ordered in 2025', COUNT(DISTINCT customer_id) FROM orders WHERE YEAR(order_date) =
2025
UNION ALL
SELECT 'Retained (2024 and 2025)', COUNT(*) FROM (SELECT customer_id FROM orders WHERE
YEAR(order_date) = 2024 INTERSECT SELECT customer_id FROM orders WHERE YEAR(order_date) =
2025) AS r
UNION ALL
SELECT 'Lost (2024 only)', COUNT(*) FROM (SELECT customer_id FROM orders WHERE
YEAR(order_date) = 2024 EXCEPT SELECT customer_id FROM orders WHERE YEAR(order_date) =
2025) AS l;
```

RESULT

measure	customers
Ordered in 2024	590
Ordered in 2025	623
Retained (2024 and 2025)	467
Lost (2024 only)	123

4 rows × 2 columns

Retention rate 2024 → 2025 = retained ÷ customers in 2024. This is the core of customer-retention analysis.

Reconciliation and data-quality checks

Set operators are perfect for comparing two versions of the same data, for example a source system extract and the warehouse table:

SQL

```
-- Rows in the source extract that are missing or different in the warehouse
SELECT order_id, order_date, customer_id, total FROM staging.orders_extract
EXCEPT
SELECT order_id, order_date, customer_id, total FROM dw.fact_orders;

-- And the reverse: rows in the warehouse not in the source
SELECT order_id, order_date, customer_id, total FROM dw.fact_orders
EXCEPT
SELECT order_id, order_date, customer_id, total FROM staging.orders_extract;
```

If both queries return no rows, the two tables match exactly. Because set operators compare **whole rows**, any difference in any column shows up. (Set operators treat two NULLs as equal, unlike = in a join.)

Set operator or join?

Question	Use
Add columns from another table	JOIN
Stack tables with the same columns	UNION ALL
Distinct list from several sources	UNION
Which keys are in both lists?	INTERSECT, or INNER JOIN / EXISTS
Which keys are in A but not B?	EXCEPT, or LEFT JOIN ... IS NULL / NOT EXISTS
Compare entire rows column by column	EXCEPT (handles NULLs as equal)

Summary

KEY TAKEAWAYS

- Joins add **columns**; set operators add or compare **rows**.
- Rules: same number of columns, compatible types, names from the first query, one ORDER BY at the end.
- UNION ALL appends everything (fast); UNION removes duplicates.
- INTERSECT returns rows in both results; EXCEPT returns rows in the first but not the second (order matters).
- Use them for combining periods, retention/churn analysis and reconciling systems.

Quick quiz

6 questions · tap an answer to check it

1. Which operator keeps duplicate rows?

A UNION ALL

B UNION

C INTERSECT

D EXCEPT

Answer: A. UNION ALL does not remove duplicates; the others return distinct rows.

2. Where can ORDER BY appear in a UNION query?

A After each SELECT

B Once, at the end of the whole statement

C Only in the first SELECT

D ORDER BY is not allowed

Answer: B. One ORDER BY applies to the combined result.

3. SELECT a, b FROM t1 UNION SELECT c FROM t2 fails because:

A UNION needs WHERE

B The queries return different numbers of columns

C Column names differ

D t2 must be first

Answer: B. Both queries must return the same number of columns.

4. To find customers who ordered in 2024 but not in 2025, use:

A 2024 UNION 2025

B 2024 INTERSECT 2025

C 2024 EXCEPT 2025

D 2025 EXCEPT 2024

Answer: C. EXCEPT returns rows in the first set that are not in the second.

5. Which column names appear in the result of a UNION?

A Those of the first query

B Those of the last query

C Both, combined

D Generic names like col1

Answer: A. Names (and aliases) come from the first SELECT.

6. Why is UNION ALL usually faster than UNION?

A It uses an index

B It does not need to sort or hash rows to remove duplicates

C It returns fewer rows

D It runs queries in parallel

Answer: B. Duplicate removal is extra work that UNION ALL skips.

EXERCISE Customer movement by region

For each region, find the number of customers who ordered in 2024, the number retained in 2025, and the retention rate. Hint: build the retained list with INTERSECT, then join to customers for the region.

Show solution (worked answer)

SQL

```
WITH c24 AS (SELECT DISTINCT customer_id FROM orders WHERE YEAR(order_date) = 2024),
     kept AS (SELECT customer_id FROM c24
              INTERSECT
              SELECT customer_id FROM orders WHERE YEAR(order_date) = 2025)
SELECT c.region,
       COUNT(DISTINCT c24.customer_id) AS customers_2024,
       COUNT(DISTINCT kept.customer_id) AS retained_2025,
       ROUND(100.0 * COUNT(DISTINCT kept.customer_id) / COUNT(DISTINCT
c24.customer_id), 1) AS retention_pct
FROM c24
JOIN customers AS c ON c.customer_id = c24.customer_id
LEFT JOIN kept ON kept.customer_id = c24.customer_id
GROUP BY c.region
ORDER BY retention_pct DESC;
```

(CTEs with WITH are explained in Lesson 13.)

INTERVIEW CORNER Questions you may be asked

Q: UNION vs UNION ALL? UNION removes duplicate rows (slower); UNION ALL keeps all rows (faster). Use UNION ALL unless de-duplication is required.

Q: JOIN vs UNION? JOIN combines columns from related tables on a key; UNION stacks rows from queries with the same column structure.

Q: How would you check that two tables contain exactly the same data? Run A EXCEPT B and B EXCEPT A; both must return zero rows (also compare row counts to catch duplicates).

Practise more online

Free, trusted places to test yourself on this topic. Links open the original site in a new tab.