



**DataDesk Tools**  
DATA TOOLKITS AND WORKFLOW ASSETS

## SQL Cheat Sheet

*Quick syntax patterns for analysts, reporting, and business data work*

**by Data Desk Tools**

*Published by Data Desk Tools. © 2026 Data Desk Tools. All rights reserved. Licensed for personal use only. Redistribution, resale, sharing, sublicensing, or uploading to marketplaces is not permitted.*

## Basic SELECT pattern

```
SELECT column_1, column_2
FROM table_name
WHERE condition
ORDER BY column_1;
```

## Common WHERE filters

```
WHERE status = 'Active'
WHERE amount > 1000
WHERE order_date >= DATE '2026-01-01'
WHERE country IN ('Italy', 'France', 'Germany')
WHERE customer_name LIKE '%Smith%'
```

## Aggregations

```
SELECT customer_id, COUNT(*) AS row_count, SUM(sales_amount) AS total_sales
FROM sales
GROUP BY customer_id;
```

## GROUP BY and HAVING

```
SELECT customer_id, SUM(sales_amount) AS total_sales
FROM sales
GROUP BY customer_id
HAVING SUM(sales_amount) > 10000;
```

## JOIN patterns

```
-- INNER JOIN: only matching rows
SELECT s.order_id, c.customer_name
FROM sales s
INNER JOIN customers c ON s.customer_id = c.customer_id;
```

```
-- LEFT JOIN: keep all rows from the left table
SELECT p.product_id, pr.unit_price
FROM products p
LEFT JOIN prices pr ON p.product_id = pr.product_id;
```

## Find records in A but not in B

```
SELECT a.product_id
FROM products a
LEFT JOIN prices b ON a.product_id = b.product_id
WHERE b.product_id IS NULL;
```

## CASE WHEN

```
SELECT order_id, sales_amount,
CASE
WHEN sales_amount >= 10000 THEN 'High'
WHEN sales_amount >= 1000 THEN 'Medium'
ELSE 'Low'
END AS order_value_segment
FROM sales;
```

## Text functions

TRIM(text\_value)  
UPPER(text\_value)  
LOWER(text\_value)  
LEFT(product\_code, 3)  
RIGHT(product\_code, 4)  
SUBSTRING(product\_code, 1, 5)  
REPLACE(product\_code, ' ', '')  
LEN(text\_value) -- SQL Server  
LENGTH(text\_value) -- Snowflake/PostgreSQL/MySQL/Oracle/DB2

## Date functions

CURRENT\_DATE  
DATE\_TRUNC('month', order\_date)  
EXTRACT(YEAR FROM order\_date)  
DATEADD(day, -30, CURRENT\_DATE)  
DATEDIFF(day, order\_date, CURRENT\_DATE)

## NULL handling

COALESCE(unit\_cost, 0)  
NULLIF(revenue, 0)  
ROUND((revenue - cost\_amount) / NULLIF(revenue, 0) \* 100, 2) AS margin\_percentage

## Window functions

ROW\_NUMBER() OVER (PARTITION BY customer\_id ORDER BY order\_date DESC)  
RANK() OVER (ORDER BY total\_sales DESC)  
LAG(total\_sales) OVER (ORDER BY sales\_month)  
SUM(sales\_amount) OVER (ORDER BY order\_date) AS running\_total

## Debug checklist before trusting results

1. Count rows before and after joins.
2. Check duplicate keys before joining.
3. Validate NULL values before calculations.
4. Test date filters with a small sample.
5. Compare totals with a trusted source report.
6. Add filters step by step instead of writing everything at once.