SQL functions, the ones you actually use.

D8A.ACADEMY/RESOURCES/SQL-FUNCTIONS

AGGREGATES

`COUNT(*)`	rows, nulls included
`COUNT(DISTINCT x)`	uniques
`SUM(x) · AVG(x)`	skip nulls
`MIN(x) · MAX(x)`	also on dates

STRINGS

`LOWER(x) · TRIM(x)`	clean first
`CONCAT(a,b) / a||b`	glue
`SUBSTRING(x,1,3)`	slice
`REPLACE(x,'a','b')`	swap
`SPLIT_PART(x,',',1)`	split

DATES

`DATE_TRUNC('month',d)`	first day
`EXTRACT(year FROM d)`	pull a part
`d + INTERVAL '7 day'`	date math
`CURRENT_DATE`	today

NULLS

`COALESCE(x,0)`	first non-null
`NULLIF(x,0)`	÷0 armor
`x IS NULL`	← never "= NULL". never.

WINDOW

`ROW_NUMBER() OVER(…)`	dedupe
`RANK() · DENSE_RANK()`	ties differ
`LAG(x) · LEAD(x)`	prev / next row
`SUM() OVER(ORDER BY d)`	running total

CONDITIONAL

`CASE WHEN … END`	if / else
`FILTER (WHERE …)`	if-count
`GREATEST(a,b)`	row-wise max

WORTH MEMORIZING

01 The dedupe pattern

Strings + window functions together: the five lines that clean most real tables.

```
SELECT * FROM (
  SELECT *, ROW_NUMBER() OVER (
    PARTITION BY LOWER(TRIM(email))
    ORDER BY created_at DESC) AS rn
  FROM customers
) t WHERE rn = 1;
```

→ Normalize inside the PARTITION BY, keep the newest row per person, drop the rest.

02 Trap: NULL breaks your math

Three lines that look right and return wrong numbers.

YOU WROTE	WHAT HAPPENS	WRITE INSTEAD
`WHERE x = NULL`	always empty	`WHERE x IS NULL`
`AVG(score)`	ignores nulls silently	`AVG(COALESCE(score,0))` if 0 is real
`a / b`	errors when b = 0	`a / NULLIF(b,0)`

→ NULL is not a value, it is "unknown". Every comparison with it returns unknown, not false.