

SQL Window Functions, Indexing & Query Optimization Cheat Sheet

ROW_NUMBER vs RANK, frame clauses (ROWS vs RANGE), B-Tree composite index selectivity, and EXPLAIN ANALYZE.

EXECUTIVE SUMMARY

Practical reference for analytical SQL and relational query tuning: window frame semantics, LAG/LEAD sessionization, composite B-Tree index column ordering (ESR rule), and join execution plans.

KEY EQUATIONS & INVARIANTS

* Index Selectivity Ratio:

$$\text{LaTeX: } S = \frac{\text{COUNT(DISTINCT col)}}{\text{COUNT(*)}} \in (0, 1]$$

Note: Higher selectivity (closer to 1.0) makes B-Tree index scans strongly preferred over sequential table scans.

1. Ranking, Offset & Frame Clause Semantics

When `ORDER BY` is specified inside `OVER(...)`, the default window frame is `RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW`. Always specify `ROWS BETWEEN ...` when duplicate sort values exist and you want deterministic physical row counts.

Function | Tie Behavior (Values: 100, 100, 90) | Primary Use Case

ROW_NUMBER() | 1, 2, 3 (no ties, arbitrary break) | Deduplication (`QUALIFY` / `WHERE rn = 1`)

RANK() | 1, 1, 3 (skips next rank after tie) | Competition leaderboards

DENSE_RANK() | 1, 1, 2 (no gaps in rank sequence) | Top-N distinct salary/revenue tiers

NTILE(k) | Splits partition into k equal buckets | Quartile / decile cohort segmentation

[SQL] Rolling 7-Day Revenue Average & Top-N Per Group Without Self-Joins

```
WITH ranked_Metrics AS (
    SELECT
        department_id,
        employee_id,
        metric_date,
        revenue,
        DENSE_RANK() OVER (
            PARTITION BY department_id
            ORDER BY revenue DESC
        ) AS dept_rank,
        AVG(revenue) OVER (
            PARTITION BY department_id
            ORDER BY metric_date
            ROWS BETWEEN 6 PRECEDING AND CURRENT ROW
        ) AS rolling_7d_avg
    FROM daily_sales
)
SELECT * FROM ranked_Metrics WHERE dept_rank <= 3;
```

2. The ESR Rule for Composite B-Tree Indexes

Order composite B-Tree index columns by **E**quality predicates first (`WHERE status = 'active'`), **S**ort columns second (`ORDER BY created\_at DESC`), and **R**ange columns last (`WHERE amount > 100`). Use `INCLUDE (col1, col2)` for covering Index-Only Scans.

COMMON PRODUCTION & INTERVIEW PITFALLS

- Using `LAST_VALUE(col)` with the default frame returns the current row's value instead of the partition's last row; use `ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING`.
- Wrapping an indexed column in a function (`WHERE DATE(created_at) = '2026-10-23'`) disables standard B-Tree index usage (non-SARGable); rewrite as a half-open range.

FREQUENTLY ASKED QUESTIONS

Q: What is the difference between ROWS BETWEEN and RANGE BETWEEN in SQL window functions?

A: ROWS counts physical rows before/after the current row, whereas RANGE groups all peer rows that share the same ORDER BY value.