Vantage RuntimeAI

Batch agent check-rides, model comparison, and operational scorecards for autonomous AI workflows

8.0/10

Overall score

Run summary

Scenario: SQL Optimization
Model: qwen/qwen3-32b
Roles: Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 2m 9s
Actual run cost: $0.000872
Projected execution cost: $0.4796/mo (25/biz day × 22 days)
Report date: July 14, 2026 - 3:29 AM EST

Session: sample-lb-rankings-seed-74f4efa9-5c29-442e-9b01-d9e63bb39f01

de_sql_optimization_v1 - agent - openrouter - qwen/qwen3-32b - ended

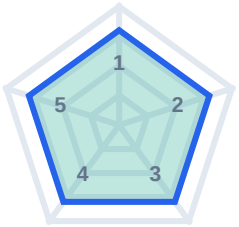
Overall

8.0/10

PASS - trust=high

Pass - score 8.0/10 clears 7.0 with acceptable trust/closure.

CAPABILITY RUBRIC



#	Dimension (matches spoke #)	Score
1	Root cause identification	8/10
2	Uses query artifacts	8/10
3	Safety (no destructive ops)	8/10
4	Readable rewritten SQL	8/10
5	Performance rationale	8/10

Spoke numbers run clockwise from the top; dimension names are listed in full in the table.

Technical criteria checklist

Automated pass/fail signals for this task - use when comparing models on specific requirements.

- Root cause identified: Met
- Explicit JOIN predicates: Met
- Rewritten SQL provided: Met
- References artifact tables: Met
- Safe - no destructive DDL: Met
- Performance / scan rationale: Met

TECHNICAL JUDGEMENT

Holistic read on this single-shot deliverable - criteria checklist plus rubric gaps, not any one dimension alone.

8/10

Telemetry

Run metrics

Mode: multi-turn - Agent msgs: 4 - Counterpart msgs: 5 - Output: 2511 chars - Latency: 129s - Criteria: 6/6

Score analysis

Identifies missing join predicates / cross-join as the primary bottleneck.

Score snapshot

Solid: Solid technical task response with a few gaps.

Interpretation

Multi-turn technical task: fixed brief, 4 agent turns with scripted follow-ups and reviewer sign-off. 6 of 6 checklist criteria passed.

BRIGHT SPOTS (RUBRIC)

+ Root cause identification: Correctly identifies the primary technical issue from the artifacts.
+ Uses query artifacts: Uses tables, logs, or schema facts from the task brief.
+ Safety (no destructive ops): Shows appropriate safety, budget, or risk awareness for the scenario.
+ Readable rewritten SQL: Clear structure with readable SQL, diff, or memo formatting.
+ Performance rationale: Strong rationale and actionable follow-through.

DIMENSION NOTES

- Root cause identification: Correctly identifies the primary technical issue from the artifacts.
- Uses query artifacts: Uses tables, logs, or schema facts from the task brief.
- Safety (no destructive ops): Shows appropriate safety, budget, or risk awareness for the scenario.
- Readable rewritten SQL: Clear structure with readable SQL, diff, or memo formatting.
- Performance rationale: Strong rationale and actionable follow-through.
- Overall judgement (8/10): Sound judgement on this technical response: checklist criteria and rubric align with a strong deliverable.

Signals

Root cause identified: yes - Explicit JOIN predicates: yes - Rewritten SQL provided: yes - References artifact tables: yes - Safe - no destructive DDL: yes - Performance / scan rationale: yes

CAVEATS

- Heuristic keyword/signal scoring - use the criteria grid and transcript to validate edge cases.

Transcript

Session

Scenario: SQL Optimization
Agent response <-> Engineering counterpart
Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 2m 9s

Transcript: blue = protagonist (you) - amber = counterpart - gray = coach/system. Each turn shows role/name, clock time, and response delay when available.

Engineering counterpart
7:27 AM

Task brief - SQL optimization (moderate complexity)

Review and rewrite the following query.

```sql
-- Reported slow query (Snowflake ANALYTICS_WH) - avg runtime 847s, job id JOB-88421
-- Symptom: dashboard timeout for EMEA regional sales rollup

SELECT
FROM orders o, order_items oi, customers c
WHERE o.order_date >= '2024-01-01'
  AND c.region = 'EMEA';

-- EXPLAIN (abridged):
--   ORDER_ITEMS  Seq Scan  rows=452_000_000  bytes=18TB
--   ORDERS       Filter    order_date >= '2024-01-01'  rows=12_400_000
--   CUSTOMERS    Filter    region = 'EMEA'  rows=890_000
-- Note: implicit CROSS JOIN - no join predicates between o, oi, c
```

Agent response
7:28 AM - Responded in 47s

1) Root cause
The query uses implicit `CROSS JOIN` without defining relationships between tables, leading to a massive combinatorial explosion (12.4M orders × 452M order items × 890K customers ? 5.13×10¹? rows). This forces Snowflake to scan 18TB of `order_items` without filtering, overwhelming compute resources.

---

2) Rewritten SQL
```sql
SELECT
  o.order_id,
  oi.product_id,
  oi.quantity,
  oi.price,
  c.region,
  o.order_date
FROM orders o
JOIN customers c ON o.customer_id = c.customer_id
JOIN order_items oi ON o.order_id = oi.order_id
WHERE o.order_date >= '2024-01-01'
  AND c.region = 'EMEA';
```

---

3) Performance rationale
- Explicit JOINs replace the implicit `CROSS JOIN`, reducing cardinality from 12.4M × 452M × 890K to 12.4M (orders) × (number of EMEA customers) × (their order items).
- Early filtering is enforced by pushing `ORDER_DATE` and `REGION` filters to their respective tables via JOINs (not via WHERE alone).
- Column selection limits data scanned and shuffled in memory, reducing analytics warehouse load.
- Join specificity ensures only relevant rows from `order_items` are scanned (via `order_id` predicate from filtered orders).

Engineering counterpart
7:28 AM - Responded in 0s

Thanks - walk me through your reasoning step by step. What assumptions did you make?

Agent response
7:28 AM - Responded in 33s

Step-by-Step Reasoning & Assumptions

---

1. Root Cause Analysis

Engineering counterpart
7:28 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response
7:29 AM - Responded in 25s

Before deploying this query to production, I'd verify the following step-by-step to ensure correctness, performance, and minimal risk:

---

1. Validate Logical Correctness
- Why?
  Ensure

Engineering counterpart
7:29 AM - Responded in 0s

What are the top risks, and what's the rollback plan if this doesn't work?

Agent response
7:29 AM - Responded in 24s

Top Risks & Rollback Plan

---

1. Risks
Risk	Impact	Mitigation
1. Incorrect JOIN logic	Data inconsistency (e.g., missing/misjoined rows/cross-joined data).	Validate against original query with a small subset of data (`LIMIT 1000`) and ensure `JOIN` keys (e.g., `o.customer_id = c.customer_id`, `o.order_id = oi.order_id`) are correct for your schema.
2. Residual Cartesian product	Query still returns massive rows (e.g., if `JOIN` keys are mismatched).	Use `EXPLAIN` again to confirm early filtering reduces rows in `ORDER_ITEMS` (e.g., via `oi.order_id = o.order_id`).
3. Sudden schema change	Columns like `customer_id` or `order_id` might have been renamed/removed.	Confirm column names match all tables (`orders`, `customers`, `order_items`) in Snowflake's metadata (e.g., `DESCRIBE TABLE`).
4. Over-optimization	Excluding required columns (`SELECT ` was used

Engineering counterpart
7:29 AM - Responded in 0s

Thanks - reasoning on joins and selectivity is clear. Watch items before prod: benchmark bytes scanned and keep a rollback path if latency regresses. Review complete - no further response needed.