Vantage RuntimeAI

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

8.0/10

Overall score

Run summary

Scenario: Partition Filter Fix
Model: qwen/qwen-2.5-7b-instruct
Roles: Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 9s
Actual run cost: $0.000440
Projected execution cost: $0.1936/mo (20/biz day × 22 days)
Report date: July 14, 2026 - 3:30 AM EST

Session: sample-lb-rankings-seed-cbebc06e-e5ee-4bbf-b7ed-c193544831a4

de_sql_partition_fix_v1 - agent - openrouter - qwen/qwen-2.5-7b-instruct - ended

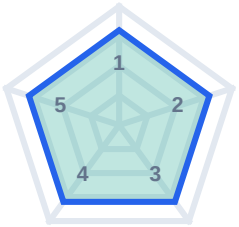
Overall

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	Partition scan diagnosis	8/10
2	Uses table/partition facts	8/10
3	Safety (no destructive ops)	8/10
4	Readable fixed SQL	8/10
5	Scan reduction 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.

- Partition scan root cause: Met
- event_date filter in fix: Met
- Corrected SQL provided: Met
- References app_events table: Met
- Safe - no destructive DDL: 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: 2 - Counterpart msgs: 3 - Output: 2693 chars - Latency: 9s - Criteria: 5/5

Score analysis

5/5 automated criteria met - see checklist below.

Score snapshot

Solid: Solid technical task response with a few gaps.

Interpretation

Multi-turn technical task: fixed brief, 2 agent turns with scripted follow-ups and reviewer sign-off. 5 of 5 checklist criteria passed.

BRIGHT SPOTS (RUBRIC)

+ Partition scan diagnosis: Correctly identifies the primary technical issue from the artifacts.
+ Uses table/partition facts: 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 fixed SQL: Clear structure with readable SQL, diff, or memo formatting.
+ Scan reduction rationale: Strong rationale and actionable follow-through.

DIMENSION NOTES

- Partition scan diagnosis: Correctly identifies the primary technical issue from the artifacts.
- Uses table/partition facts: 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 fixed SQL: Clear structure with readable SQL, diff, or memo formatting.
- Scan reduction 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

Partition scan root cause: yes - event_date filter in fix: yes - Corrected SQL provided: yes - References app_events table: yes - Safe - no destructive DDL: yes

CAVEATS

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

Transcript

Session

Scenario: Partition Filter Fix
Agent response <-> Engineering counterpart
Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 9s

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

Engineering counterpart
7:30 AM

Task brief - Partition filter fix (simple complexity)

Fix the query so it scans only the needed partition(s).

```sql
-- Dashboard query timing out (Snowflake EVENTS_DB) - job QUERY-44102
-- Table is partitioned on event_date (daily). User only needs yesterday.

SELECT user_id, event_type, COUNT() AS n
FROM events_db.public.app_events
WHERE user_id = 'usr_88421'
GROUP BY 1, 2;

-- EXPLAIN: scans 90 days of partitions (~4.1B rows) because event_date is not filtered.
```

Agent response
7:30 AM - Responded in 3s

1. Root cause: The query does not include a filter on the `event_date` partition key.

2. Corrected SQL:

```sql
SELECT user_id, event_type, COUNT() AS n
FROM events_db.public.app_events
WHERE user_id = 'usr_88421'
  AND event_date = DATEADD(day, -1, CURRENT_DATE)
GROUP BY 1, 2;
```

3. Rationale: Filter on the `event_date` partition key to limit the scanned partitions to only the most recent day.

Engineering counterpart
7:30 AM - Responded in 0s

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

Agent response
7:30 AM - Responded in 7s

Certainly! Let's walk through the reasoning step by step and address any assumptions:

Step-by-Step Reasoning

1. Identify the Problem:
   - The original query is timing out because it scans all partitions (90 days in this case).
   - This is happening because the `event_date` filter is missing.

2. Assumptions:
   - The table is partitioned on `event_date` with daily partitions.
   - The user only needs data from yesterday.
   - The current date can be used to determine yesterday's date.

3. Understand the Partition Structure:
   - The table has partitions such as `event_date=2023-10-01`, `event_date=2023-10-02`, etc.
   - To optimize, we need to filter the query to only include yesterday's partition.

4. Correct the SQL Query:
   - Add a filter on `event_date` to include only yesterday's date.
   - Use `DATEADD(day, -1, CURRENT_DATE)` to get yesterday's date dynamically.

Corrected SQL Query

```sql
SELECT user_id, event_type, COUNT() AS n
FROM events_db.public.app_events
WHERE user_id = 'usr_88421'
  AND event_date = DATEADD(day, -1, CURRENT_DATE)
GROUP BY 1, 2;
```

Detailed Explanation

1. Filter on `event_date`:
   - The filter `event_date = DATEADD(day, -1, CURRENT_DATE)` ensures that only yesterday's partition is scanned.
   - This reduces the number of partitions to a single partition, significantly decreasing the number of rows that need to be scanned.

2. Use of `DATEADD(day, -1, CURRENT_DATE)`:
   - `CURRENT_DATE` returns today's date.
   - `DATEADD(day, -1, CURRENT_DATE)` calculates yesterday's date.
   - This makes the query dynamic and ensures it always fetches the correct date without hardcoding.

Final SQL Query

```sql
SELECT user_id, event_type, COUNT() AS n
FROM events_db.public.app_events
WHERE user_id = 'usr_88421'
  AND event_date = DATEADD(day, -1, CURRENT_DATE)
GROUP BY 1, 2;
```

Assumptions Recap

- The table is partitioned by `event_date` with daily partitions.
- The query needs to filter on the `event_date` to limit scans to yesterday's partition.
- Using `DATEADD(day, -1, CURRENT_DATE)` dynamically computes yesterday's date, ensuring the query always fetches the correct data.

This approach minimizes the number of partitions scanned and improves query performance.

Engineering counterpart
7:30 AM - Responded in 0s

Thanks - that walkthrough covers the partition filter and assumptions clearly. Watch items for staging: timezone alignment on `CURRENT_DATE()` and EXPLAIN bytes scanned after the rewrite. Review complete - no further response needed.