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: openai/gpt-4.1-mini
Roles: Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 11s
Actual run cost: $0.002320
Projected execution cost: $1.02/mo (20/biz day × 22 days)
Report date: July 14, 2026 - 3:29 AM EST

Session: sample-lb-rankings-seed-74ae2ab2-2a81-406e-9b90-b151716311f9

de_sql_partition_fix_v1 - agent - openrouter - openai/gpt-4.1-mini - ended

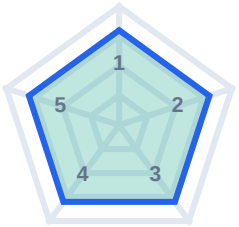
Overall

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: 2004 chars - Latency: 11s - 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: 11s

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

Engineering counterpart
7:29 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
7:29 AM - Responded in 3s

(1) Root cause: The query does not filter on the partitioned column event_date, causing a full scan of all partitions.

(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 = CURRENT_DATE - 1
GROUP BY user_id, event_type;
```

(3) Rationale: Adding a filter on event_date restricts the scan to only yesterday's partition, greatly reducing data scanned and improving query performance.

Engineering counterpart
7:29 AM - Responded in 0s

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

Agent response
7:29 AM - Responded in 8s

1. Understanding the problem:
The query is running on a large table partitioned by `event_date` (daily), but the query lacks any filter on this partitioning column. This causes Snowflake to scan all partitions (~90 days, 4.1B rows), leading to timeouts.

2. Assumptions:
- The `event_date` column exists and is correctly set as the partitioning column.
- The user only needs data for "yesterday" relative to the query execution date.
- The table is large and scanning unnecessary partitions significantly impacts performance.
- Adding an `event_date` filter does not change the semantic correctness of the query for the user's needs.

3. Reasoning:
Since the table is partitioned by `event_date` and the user wants only yesterday's data, adding a filter `event_date = CURRENT_DATE - 1` will restrict the scan to that single day's partition. This reduces the data scanned from billions of rows to the much smaller set for one day.

4. Solution approach:
Add the `event_date` filter in the WHERE clause alongside the existing `user_id` filter so that Snowflake can prune partitions and only scan the relevant partition.

5. Validation:
Verify the query still returns the same grouping and aggregation but now limited to yesterday's data, satisfying user requirements and improving performance.

In summary, the root cause was the missing partition filter; adding that filter optimizes the scan cost without altering query semantics for the user's intended timeframe.

Engineering counterpart
7:29 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.