



## 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: meta-llama/llama-3.3-70b-instruct

Roles: Data Engineer (AI agent) <> Engineering counterpart (AI agent)

Duration: 24s

Actual run cost: \$0.000670

Projected execution cost: \$0.2948/mo (20/biz day × 22 days)

Report date: July 14, 2026 - 3:29 AM EST

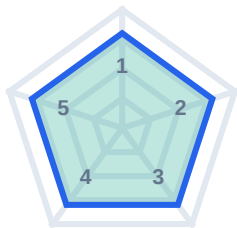
Overall

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: 2226 chars - Latency: 24s - 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: 24s

**Engineering counterpart**

**7:28 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 8s**

1. The root cause of the issue is that the query is scanning 90 days of partitions because the `event\_date` column, which the table is partitioned on, is not being filtered.

2. The corrected SQL query is:

```
```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. By adding a filter on `event\_date` to only include yesterday's date, we reduce the number of partitions scanned, thus 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 17s

Here's a step-by-step walkthrough of my reasoning:

1. Understanding the problem: The query is timing out because it's scanning 90 days of partitions, which translates to approximately 4.1 billion rows. This suggests that the query is not efficient and needs optimization.
2. Identifying the root cause: The EXPLAIN plan indicates that the query scans 90 days of partitions because the `event\_date` column is not filtered. This implies that the table is partitioned on the `event\_date` column, and the query is not taking advantage of this partitioning.
3. Assumptions:
  - I assumed that the `event\_date` column is of a date type (e.g., DATE or TIMESTAMP) and is used as the partition key for the table.
  - I assumed that the user only needs data from yesterday, as mentioned in the task brief.
  - I assumed that the `CURRENT\_DATE` function returns the current date in Snowflake.
4. Correcting the query: To fix the issue, I added a filter on the `event\_date` column to only include rows where `event\_date` is equal to yesterday's date. I used the `DATEADD` function to subtract one day from the current date, which gives me yesterday's date.
5. Rationale: By adding this filter, I'm reducing the number of partitions that need to be scanned, which should significantly improve query performance. Since the table is partitioned on `event\_date`, Snowflake can now prune the partitions and only scan the partition(s) that contain data for yesterday's date.

Overall, my goal was to add a filter that would allow Snowflake to take advantage of the partitioning on the `event\_date` column, reducing the amount of data that needs to be scanned and improving query performance.

### 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.