



Vantage RuntimeAI

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

8.0/10

Overall score

Run summary

Scenario: Freshness SLA Breach

Model: meta-llama/llama-4-maverick

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

Duration: 38s

Actual run cost: \$0.002320

Projected execution cost: \$2.55/mo (50/biz day × 22 days)

Report date: July 14, 2026 - 2:55 AM EST

Overall

8.0/10

PASS - trust=high

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

CAPABILITY RUBRIC



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.

- Upstream blocker identified: Met
- SLA breach acknowledged: Met
- Downstream impact noted: Met
- Restore steps listed: 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: 9909 chars - Latency: 38s - Criteria: 4/4

Score analysis

Identifies int_order_enriched failure as upstream blocker.

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. 4 of 4 checklist criteria passed.

BRIGHT SPOTS (RUBRIC)

- + Stale upstream identified: Correctly identifies the primary technical issue from the artifacts.
- + Uses SLA / freshness facts: Uses tables, logs, or schema facts from the task brief.
- + Downstream blast radius: Shows appropriate safety, budget, or risk awareness for the scenario.
- + Structured status summary: Clear structure with readable SQL, diff, or memo formatting.
- + Restore & verify steps: Strong rationale and actionable follow-through.

DIMENSION NOTES

- Stale upstream identified: Correctly identifies the primary technical issue from the artifacts.
- Uses SLA / freshness facts: Uses tables, logs, or schema facts from the task brief.
- Downstream blast radius: Shows appropriate safety, budget, or risk awareness for the scenario.
- Structured status summary: Clear structure with readable SQL, diff, or memo formatting.
- Restore & verify steps: 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

Upstream blocker identified: yes - SLA breach acknowledged: yes - Downstream impact noted: yes - Restore steps listed: yes

CAVEATS

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

Transcript

Session

Scenario: Freshness SLA Breach

Agent response <-> Engineering counterpart

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

Duration: 38s

Engineering counterpart

6:54 AM

Task brief - Freshness SLA breach (moderate complexity)

A P1 mart breached its freshness SLA. Triage using the metadata JSON:

```
```json
{
 "mart": "analytics.fct_orders",
 "owner_team": "data-platform",
 "sla": {
 "max_staleness_minutes": 120,
 "tier": "P1 revenue dashboard"
 },
 "freshness": {
 "last_successful_run_utc": "2026-07-05T09:12:00Z",
 "now_utc": "2026-07-05T15:40:00Z",
 "staleness_minutes": 388,
 "status": "BREACHED"
 },
 "upstream": [
 {"model": "staging.stg_orders", "last_run_utc": "2026-07-05T15:05:00Z", "status": "success"},
 {"model": "staging.stg_customers", "last_run_utc": "2026-07-05T14:58:00Z", "status": "success"},
 {"model": "intermediate.int_order_enriched", "last_run_utc": "2026-07-05T09:12:00Z", "status": "failed", "error":
"DBT_RUN_TIMEOUT after 45m"}
],
 "downstream_consumers": ["Looker Revenue Overview", "exec_daily_orders Slack bot"],
 "note": "int_order_enriched has been failing since morning; fct_orders has not refreshed."
}
```
```

Agent response

6:55 AM - Responded in 9s

Triage Report: Freshness SLA Breach for `analytics.fct\_orders`

Summary

The `analytics.fct\_orders` mart has breached its P1 freshness SLA due to a failure in its upstream model `intermediate.int\_order\_enriched`, which has not run successfully since `2026-07-05T09:12:00Z`. This has resulted in staleness of 388 minutes as of `2026-07-05T15:40:00Z`, exceeding the allowed 120 minutes.

Blocking Upstream Model

The upstream model blocking the refresh of `analytics.fct\_orders` is `intermediate.int\_order\_enriched`. It failed with a `DBT\_RUN\_TIMEOUT` error after 45 minutes at `2026-07-05T09:12:00Z`.

Blast Radius on Downstream Consumers

The breach affects the following downstream consumers:

1. Looker Revenue Overview: A critical dashboard likely used for revenue analysis and decision-making.
2. exec_daily_orders Slack bot: An automated bot that probably sends daily order summaries to executives, potentially impacting their daily operations or decisions.

Ordered Steps to Restore the Mart within SLA

To restore `analytics.fct\_orders` within its SLA, follow these steps:

1. Investigate the Failure of `intermediate.int\_order\_enriched`:
 - Review the logs and error messages associated with the `DBT\_RUN\_TIMEOUT` to understand the cause of the failure.
 - Check if the failure is due to a query performance issue, data volume spike, or dependency failures.
2. Optimize or Fix `intermediate.int\_order\_enriched`:
 - Based on the investigation, optimize the model to prevent timeouts. This might involve rewriting the query, adding indexes, or partitioning the data.
 - If the model depends on other models or sources, ensure they are up-to-date and not causing the delay.
3. Manually Trigger a Run for `intermediate.int\_order\_enriched`:
 - Once the issue is addressed, manually trigger a run for `intermediate.int\_order\_enriched` to refresh it.
4. Trigger a Refresh for `analytics.fct\_orders`:
 - After `intermediate.int\_order\_enriched` is successfully refreshed, trigger a run for `analytics.fct\_orders` to update it.
5. Verify Downstream Consumers:
 - Confirm that `Looker Revenue Overview` and `exec\_daily\_orders Slack bot` are functioning correctly with the updated data.
6. Post-Mortem Analysis:
 - Conduct a post-mortem analysis to identify root causes and implement measures to prevent similar breaches in the future.

By following these steps, we can restore the freshness of `analytics.fct\_orders` within its SLA and minimize the impact on downstream consumers.

Engineering counterpart

6:55 AM - Responded in 0s

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

Agent response

6:55 AM - Responded in 6s

Let's walk through the reasoning step by step using the provided metadata JSON.

Step 1: Understand the Problem

The `analytics.fct\_orders` mart has breached its freshness SLA. The metadata JSON provides details about the mart, its SLA, freshness status, upstream models, and downstream consumers.

Step 2: Analyze the Freshness Status

The freshness status indicates that `analytics.fct\_orders` has a staleness of 388 minutes, which exceeds the allowed 120 minutes. The `last\_successful\_run\_utc` is `2026-07-05T09:12:00Z`, and `now\_utc` is `2026-07-05T15:40:00Z`.

Step 3: Identify the Blocking Upstream Model

To identify the blocking upstream model, we analyze the `upstream` section of the JSON:

```
```json
"upstream": [
 {"model": "staging.stg_orders", "last_run_utc": "2026-07-05T15:05:00Z", "status": "success"},
 {"model": "staging.stg_customers", "last_run_utc": "2026-07-05T14:58:00Z", "status": "success"},
 {"model": "intermediate.int_order_enriched", "last_run_utc": "2026-07-05T09:12:00Z", "status": "failed", "error": "DBT_RUN_TIMEOUT after 45m"}
]
```
```

`staging.stg\_orders` and `staging.stg\_customers` have run successfully after `analytics.fct\_orders`' last successful run. `intermediate.int\_order\_enriched` failed with a `DBT\_RUN\_TIMEOUT` error at `2026-07-05T09:12:00Z`, which is the same time as `analytics.fct\_orders`' last successful run.

Assumption 1: The failure of `intermediate.int\_order\_enriched` is blocking the refresh of `analytics.fct\_orders` because it has not been successfully updated since `2026-07-05T09:12:00Z`.

Step 4: Assess the Blast Radius on Downstream Consumers

The `downstream\_consumers` section lists the affected consumers:

```
```json
"downstream_consumers": ["Looker Revenue Overview", "exec_daily_orders Slack bot"]
```
```

These consumers are likely relying on fresh data from `analytics.fct\_orders`. The breach may impact their functionality or the decisions made based on their output.

Assumption 2: The downstream consumers are directly dependent on `analytics.fct\_orders` and will be affected by its staleness.

Step 5: Determine the Steps to Restore the Mart within SLA

To restore `analytics.fct\_orders`, we need to address the failure of `intermediate.int\_order\_enriched`.

1. Investigate the Failure: Review logs and error messages to understand the cause of the `DBT\_RUN\_TIMEOUT`.
2. Optimize or Fix `intermediate.int\_order\_enriched`: Based on the investigation, optimize the model to prevent timeouts.
3. Manually Trigger a Run: Trigger a run for `intermediate.int\_order\_enriched` and subsequently for `analytics.fct\_orders`.

Assumptions:

The failure is related to the `DBT\_RUN\_TIMEOUT` error, and optimizing the model will resolve the issue.
The dependency chain is correctly represented in the `upstream` section, and there are no other blocking dependencies.

By following these steps and making these assumptions, we can restore `analytics.fct\_orders` within its SLA and minimize the impact on downstream consumers.

Engineering counterpart
6:55 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

6:55 AM - Responded in 9s

Before shipping the triage report and recommended steps to production, I would verify the following:

1. Accuracy of Metadata

Verify that the metadata JSON accurately represents the current state of `analytics.fct\_orders` and its dependencies. This includes:

Checking the actual last successful run times and statuses of `analytics.fct\_orders` and its upstream models.

Confirming that the listed downstream consumers are indeed dependent on `analytics.fct\_orders`.

2. Dependency Chain

Validate the dependency chain represented in the `upstream` section to ensure:

All relevant upstream models are listed.

There are no other dependencies that might be blocking the refresh of `analytics.fct\_orders`.

3. Root Cause Analysis

Perform a preliminary root cause analysis for the failure of `intermediate.int\_order\_enriched` to:

Understand the nature of the `DBT\_RUN\_TIMEOUT` error.

Identify potential fixes or optimizations.

4. Impact on Downstream Consumers

Assess the potential impact of the staleness on the listed downstream consumers:

Evaluate the criticality of `Looker Revenue Overview` and `exec\_daily\_orders Slack bot`.

Consider any potential business or operational implications of delayed data.

5. Feasibility of Recommended Steps

Verify that the recommended steps are feasible and will likely resolve the issue:

Check if the necessary permissions and access are available to investigate and fix `intermediate.int\_order\_enriched`.

Ensure that triggering a manual run for `intermediate.int\_order\_enriched` and `analytics.fct\_orders` is possible and will not cause any unintended consequences.

6. SLA and Tier Implications

Understand the implications of the P1 SLA tier for `analytics.fct\_orders`:

Confirm the expected response time and resolution procedures for P1 SLA breaches.

Ensure that the recommended steps align with the SLA requirements.

By verifying these aspects, we can ensure that the triage report and recommended steps are accurate, effective, and suitable for production.

Engineering counterpart
6:55 AM - Responded in 0s

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

Agent response

6:55 AM - Responded in 14s

Top Risks

1. Root Cause Not Addressed: If the optimization or fix for ``intermediate.int_order_enriched`` does not address the root cause of the ``DBT_RUN_TIMEOUT`` error, the issue may recur, and ``analytics.fct_orders`` may remain stale.
2. Cascading Failures: If ``intermediate.int_order_enriched`` has dependencies or downstream models not listed in the metadata, fixing it might trigger cascading failures or issues elsewhere in the data pipeline.
3. Manual Intervention Fails: Manual triggering of ``intermediate.int_order_enriched`` and ``analytics.fct_orders`` might fail due to underlying issues, such as data inconsistencies or infrastructure problems.
4. SLA Breach Persists: If the recommended steps do not resolve the issue within the required timeframe, the SLA breach may persist, impacting downstream consumers and potentially leading to business or operational consequences.

Rollback Plan

To mitigate these risks, the following rollback plan can be implemented:

1. Revert Changes: If the optimization or fix for ``intermediate.int_order_enriched`` causes unintended consequences, revert the changes to restore the previous state.
2. Re-Trigger Previous Successful Run: If manual intervention fails, re-trigger the previous successful run for ``analytics.fct_orders`` (if available) to maintain data consistency and minimize staleness.
3. Implement Temporary Workaround: If the issue persists, implement a temporary workaround, such as:
 - Using a previous snapshot or backup of ``analytics.fct_orders`` to satisfy downstream consumers.
 - Manually updating critical fields or tables to meet immediate business needs.
4. Escalate to Next Tier: If the issue cannot be resolved within the required timeframe, escalate it to the next tier of support or management, ensuring that the necessary resources and expertise are allocated to resolve the issue.
5. Post-Mortem Analysis: Conduct a thorough post-mortem analysis to identify the root cause of the issue, document lessons learned, and implement measures to prevent similar breaches in the future.

By having a rollback plan in place, we can minimize the impact of potential failures and ensure a swift recovery in case the recommended steps do not resolve the issue.

Engineering counterpart

6:55 AM - Responded in 0s

Thanks - upstream blocker and blast radius are well ordered. Prioritize revenue-facing dashboards if full freshness slips past the hour mark. Review complete - no further response needed.