



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: qwen/qwen3-32b

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

Duration: 1m 17s

Actual run cost: \$0.000872

Projected execution cost: \$0.9592/mo (50/biz day × 22 days)

Report date: July 14, 2026 - 2:58 AM EST

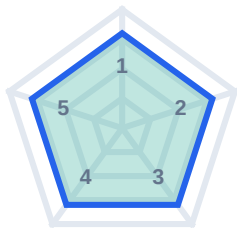
Overall

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	Stale upstream identified	8/10
2	Uses SLA / freshness facts	8/10
3	Downstream blast radius	8/10
4	Structured status summary	8/10
5	Restore & verify steps	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.

- 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: 5721 chars - Latency: 77s - 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: 1m 17s

Engineering counterpart

6:56 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

6:57 AM - Responded in 14s

Analysis & Resolution for Analytics.fct_orders Freshness SLA Breach

Root Cause

The intermediate.int_order_enriched model is blocking the refresh of the analytics.fct_orders mart.

- Status: Failed (`DBT\_RUN\_TIMEOUT` after 45m)
- Last Run: 09:12 UTC (coincides with the last successful run of the mart)
- Staleness Gap: 388 minutes (6h28m) exceeds the 120-minute SLA.

Blast Radius

Downstream consumers relying on fct_orders are impacted:

1. Looker Revenue Overview (P1 revenue dashboard): Critical for real-time/recent revenue tracking, likely affecting business decisions.
2. exec_daily_orders Slack bot: Delivers daily summaries to exec teams (though less time-sensitive than the dashboard, its delay could still raise concerns).

Steps to Restore Freshness

1. Investigate the `int\_order\_enriched` Timeout
 - Check DBT logs for resource constraints (e.g., query complexity, data volume, or infrastructure limits).
 - Confirm if the model is processing recent data (e.g., incremental load issues) or if it's a full-refresh bottleneck.
2. Optimize or Reconfigure the Model
 - Optimize SQL: Simplify joins, reduce subqueries, or partition data to speed execution.
 - Incremental Load: If not already implemented, configure the model to process only new/changed data since 09:12 UTC.
 - Scale Resources: Request temporary compute scaling (e.g., bigger cluster or parallelization) to handle the job.
3. Manual Rerun of `int\_order\_enriched`
 - Trigger a manual run with the latest data. Monitor progress to ensure it completes within the SLA window (≤ 120 minutes).
4. Resume `fct\_orders` Dependency Chain
 - Once `int\_order\_enriched` succeeds, the mart will rebuild automatically (if scheduled).
 - If time-sensitive, manually trigger `fct\_orders` to expedite restoration.
5. Verify Freshness
 - Confirm the mart's next successful run completes within 120 minutes of real-time (target: next run by 17:00 UTC).
6. Preventative Measures
 - Set up automatic retries for timeouts.
 - Implement monitoring alerts for long-running models.
 - Add freshness checks for `int\_order\_enriched` to surface issues earlier.

Time Window to Meet SLA

- Current Staleness: 388 minutes (last successful run at 09:12 UTC).

- Target Resolution Time: Next successful `fct\_orders` run must complete by 17:00 UTC (i.e., staleness $\leq$ 120 minutes).
- Priority: Resolve `int\_order\_enriched` timeout first.

Escalation Note

- Coordinate with the DBT/ETL team to address the `int\_order\_enriched` failure.
- Inform revenue dashboard stakeholders of the expected resolution timeline.

By prioritizing optimization and manual reruns, we can restore the mart to SLA compliance and prevent recurrence.

Engineering counterpart

6:57 AM - Responded in 0s

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

Agent response

6:57 AM - Responded in 19s

Here's a step-by-step reasoning breakdown and the assumptions I made:

Step 1: Identify the Root Cause

Reasoning:

- The mart (`fct\_orders`) hasn't refreshed since 09:12 UTC, even though the current time is 15:40 UTC (388 minutes ago).
 - Upstream models:
 - `stg\_orders` and `stg\_customers` ran recently (at 15:05 and 14:58 UTC) and succeeded.
 - The intermediate model (`int\_order\_enriched`) failed at 09:12 UTC with a timeout.
- Conclusion: The mart depends on `int\_order\_enriched`, which is failing.

Assumptions:

1. The mart pipeline is linear:

Engineering counterpart

6:57 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

6:57 AM - Responded in 17s

Here's a checklist of critical verifications to perform before shipping the fix to production to ensure the solution is robust and safe:

1. Validate the

Engineering counterpart

6:57 AM - Responded in 0s

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

Agent response

6:58 AM - Responded in 28s

Here are the top risks of implementing the proposed solution, along with a rollback plan in case the fix fails:

Top Risks

1. Data Quality Impact

- Risk: Optimizing the ``int_order_enriched`` model (e.g., incremental load) might introduce gaps or inconsistencies if the logic for partial updates is flawed (e.g., missing partitions, incomplete joins).
- Why It Matters: The mart (``fct_orders``) relies on accurate enriched data for revenue reporting. Incorrect data could lead to incorrect insights.

2. Unintended Side Effects

- Risk: Temporarily scaling infrastructure (e.g., larger compute resources) could strain other workloads or introduce instability (e.g., memory leaks, query contention).
- Why It Matters: Overprovisioning might resolve the timeout locally but cause broader system issues.

3. Downstream Dependency Breakage

- Risk: Modifying ``int_order_enriched`` (e.g., incremental logic, schema changes) could break other consumers of this model (not just ``fct_orders``).
- Why It Matters: Unknown dependencies might surface only after deployment.

4. Manual Rerun Failure

- Risk: A manual rerun of ``int_order_enriched`` might fail again (e.g., same timeout, new errors from unhandled edge cases).
- Why It Matters: This would delay SLA recovery and require urgent troubleshooting.

5. Time Pressure

- Risk: High-pressure manual interventions (e.g., urgent DBT changes) increase the chance of human error (e.g., misconfigured incremental filters, accidental data deletion).

Rollback Plan

If the fix fails and the issue persists, execute the following steps in order:

1. Revert to Last Known Good Version

- Action: Revert the ``int_order_enriched`` model to its previous working version (before optimization/incremental changes).
- Tools: Use version control (e.g., Git) to roll back changes to the model's ``sql`` file

Engineering counterpart

6:58 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.