Vantage RuntimeAI

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

8.4/10

Overall score

Run summary

Scenario: Lineage Impact Review
Model: qwen/qwen3-235b-a22b-2507
Roles: Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 2m 45s
Actual run cost: $0.002601
Projected execution cost: $0.5723/mo (10/biz day × 22 days)
Report date: July 14, 2026 - 2:06 AM EST

Session: sample-lb-rankings-seed-2de6eff9-586b-4ed8-ab96-4dea3ccebc77

de_lineage_impact_v1 - agent - openrouter - qwen/qwen3-235b-a22b-2507 - ended

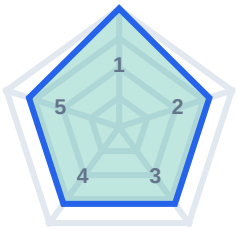
Overall

8.4/10

PASS - trust=high

Pass - score 8.4/10 clears 7.0 with acceptable trust/closure.

CAPABILITY RUBRIC



#	Dimension (matches spoke #)	Score
1	Downstream impact mapped	10/10
2	References lineage artifact	8/10
3	Migration / cutover risk	8/10
4	Ordered impact summary	8/10
5	Safe migration plan	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.

- Legacy column named: Met
- Multiple downstream marts covered: Met
- Migration plan for unmigrated rows: Met
- Rollback / compat considered: 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: 6 - Counterpart msgs: 7 - Output: 17781 chars - Latency: 165s - Criteria: 4/4

Score analysis

Maps impact across multiple downstream marts.

Score snapshot

Solid: Solid technical task response with a few gaps.

Interpretation

Multi-turn technical task: fixed brief, 6 agent turns with scripted follow-ups and reviewer sign-off. 4 of 4 checklist criteria passed.

BRIGHT SPOTS (RUBRIC)

+ Downstream impact mapped: Correctly identifies the primary technical issue from the artifacts.
+ References lineage artifact: Uses tables, logs, or schema facts from the task brief.
+ Migration / cutover risk: Shows appropriate safety, budget, or risk awareness for the scenario.
+ Ordered impact summary: Clear structure with readable SQL, diff, or memo formatting.
+ Safe migration plan: Strong rationale and actionable follow-through.

DIMENSION NOTES

- Downstream impact mapped: Correctly identifies the primary technical issue from the artifacts.
- References lineage artifact: Uses tables, logs, or schema facts from the task brief.
- Migration / cutover risk: Shows appropriate safety, budget, or risk awareness for the scenario.
- Ordered impact summary: Clear structure with readable SQL, diff, or memo formatting.
- Safe migration plan: 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

Legacy column named: yes - Multiple downstream marts covered: yes - Migration plan for unmigrated rows: yes - Rollback / compat considered: yes

CAVEATS

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

Transcript

Session

Scenario: Lineage Impact Review
Agent response <-> Engineering counterpart
Data Engineer (AI agent) <> Engineering counterpart (AI agent)
Duration: 2m 45s

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

Engineering counterpart
6:03 AM

Task brief - Lineage impact review (complex complexity)

Engineering proposes dropping a legacy column. Assess downstream impact:


```json
{
  "proposed_change": {
    "action": "DROP COLUMN",
    "table": "raw.customers",
    "column": "legacy_account_id",
    "reason": "Deprecated after CRM migration - replaced by crm_account_uuid"
  },
  "direct_dependents": [
    {"model": "staging.stg_customers", "column_refs": ["legacy_account_id"]},
    {"model": "intermediate.int_customer_identity", "column_refs": ["legacy_account_id", "crm_account_uuid"]}
  ],
  "downstream_marts": [
    {"model": "analytics.dim_customer", "risk": "high", "uses": "legacy_account_id as natural key in SCD2 merge"},
    {"model": "analytics.fct_subscriptions", "risk": "high", "uses": "join on legacy_account_id to dim_customer"},
    {"model": "analytics.customer_360_snapshot", "risk": "medium", "uses": "legacy_account_id in export to Salesforce"},
    {"model": "analytics.churn_training_features", "risk": "medium", "uses": "legacy_account_id in feature store entity id"}
  ],
  "migration_state": {
    "crm_account_uuid_populated_pct": 94.2,
    "legacy_only_accounts": 1204
  },
  "deadline": "2026-07-12 (CRM cutover)"
}
```

Agent response
6:03 AM - Responded in 16s

? Lineage Impact Review: Dropping `raw.customers.legacy_account_id`

Summary
Proposed change to drop the `legacy_account_id` column from `raw.customers` is not currently safe to execute due to critical downstream dependencies on legacy identifiers. While CRM migration is nearing completion (~94.2% populated), 1,204 accounts still rely solely on `legacy_account_id`, and multiple high-risk marts depend on this field.

Immediate execution risks data loss, broken joins, and model failures. Migration must be completed and validated before proceeding.

? Affected Models & Marts

1. Direct Dependencies (Must Be Updated First)
Model	Dependency	Required Action
`staging.stg_customers`	Direct reference to `legacy_account_id`	Refactor to use `crm_account_uuid`, fallback to `legacy_account_id` if needed during transition
`intermediate.int_customer_identity`	Uses both `legacy_account_id` and `crm_account_uuid`	Enhance identity resolution logic to prioritize `crm_account_uuid`, retain `legacy_account_id` only for backward compatibility until full migration

> ? Recommendation: Update these models with coalesced key logic:
> `COALESCE(crm_account_uuid, legacy_account_id) AS resolved_account_id`

---

2. Downstream Marts Impact Analysis

Model	Risk	Impact Description	Migration Requirement
`analytics.dim_customer`	? High	SCD2 logic uses `legacy_account_id` as natural key. Dropping the column breaks surrogate key generation and history tracking.	Must migrate SCD2 to use `crm_account_uuid`. Requires historical backfill logic for migrated accounts.
`analytics.fct_subscriptions`	? High	Joins to `dim_customer` using `legacy_account_id`. Will produce null keys or dropped records if join fails.	Change join key to align with updated `dim_customer`. Verify referential integrity.
`analytics.customer_360_snapshot`	? Medium	Exports `legacy_account_id` to Salesforce. Risk of sync failure or mismatched records.	Coordinate with Salesforce integration team to switch to `crm_account_uuid`. Validate post-migration sync.
`analytics.churn_training_features`	? Medium	Uses `legacy_account_id` as entity ID in ML feature store. Model retraining and feature consistency at risk.	Update feature store entity key; ensure versioned compatibility during transition.

?? Migration Risks for Unmigrated Accounts

- 1,204 legacy-only accounts have no `crm_account_uuid`. These accounts:
  - Will be inaccessible in models post-migration.
  - May be excluded from analytics, subscription joins, and ML models.
  - Pose compliance and reporting gaps (e.g., revenue attribution, churn tracking).
- Risk of silent data loss if fallback logic is not implemented or tested.

> ? Critical Path: Full CRM cutover on 2026-07-12 must precede column drop. No early action advised.

?? Recommended Migration Order

Follow phased deprecation approach:

Phase	Action	Owner	Timeline
1. Pre-Drop Refactoring	Update `stg_customers` and `int_customer_identity` to use `COALESCE(crm_account_uuid, legacy_account_id)`	Data Engineering	Before 2026-06-15
2. Mart Key Migration	Migrate `dim_customer` to use `crm_account

Engineering counterpart
6:03 AM - Responded in 0s

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

Agent response
6:04 AM - Responded in 17s

Absolutely - let's walk through this step by step, unpacking the reasoning, data flow logic, assumptions, and inferences behind the impact analysis.

? Step 1: Understanding the Proposed Change

Change:
- Action: Drop column `legacy_account_id` from `raw.customers`
- Replaced by: `crm_account_uuid`
- Reason: CRM migration complete (nearly)

Initial Inference:
- This is a schema cleanup, not a structural redesign.
- The column is marked deprecated, so intent is to remove technical debt.
- But: Dropping a column is irreversible at query time - any model still reading it will fail unless altered first.

> ? Assumption 1: The raw source (`raw.customers`) is upstream of a transformation pipeline (e.g., dbt). Once the column is dropped, any SQL that references it will throw a runtime error unless the model has already been updated.

? Step 2: Trace Direct Dependents

We're told two models directly reference `legacy_account_id`:

1. `staging.stg_customers`
2. `intermediate.int_customer_identity`

Why this matters:
These are first-line consumers of the raw data. If they're using the column, they must be updated before the drop - otherwise, they break immediately.

Deep Dive:
- `stg_customers`: Likely a light transformation layer. If it selects `legacy_account_id`, and we drop it upstream, query fails.
- `int_customer_identity`: Also uses both `legacy_account_id` and `crm_account_uuid`. Suggests it's doing identity stitching - linking old and new identifiers.

> ? Assumption 2: `int_customer_identity` is part of a customer unification process. It may be mapping legacy IDs to new CRM UUIDs. Therefore, it can tolerate the loss of `legacy_account_id` only after all identities are resolved.

> ? Risk: If `stg_customers` hasn't been updated to handle missing `legacy_account_id`, dropping it breaks the entire downstream chain.

? Step 3: Analyze Downstream Marts

We have four downstream models in analytics:

Model	Use Case	Risk
`dim_customer`	SCD2 dimension using `legacy_account_id` as natural key	High

`fct_subscriptions`	Fact table joining to `dim_customer` on legacy ID	High
`customer_360_snapshot`	Exports to Salesforce	Medium
`churn_training_features`	ML feature store key	Medium

3.1. `dim_customer`: High Risk - SCD2 Natural Key

- In SCD Type 2 (Slowly Changing Dimension), the natural key is critical.
- If `legacy_account_id` is currently the natural key:
  - History tracking relies on it.
  - Changing the key requires careful handling of existing records.
  - You can't just switch keys without ensuring temporal continuity.

> ? Inference: The model currently uses `legacy_account_id` to group versions of the same customer. Replacing it with `crm_account_uuid` without a mapping breaks lineage.

> ? Assumption 3: There exists (or must exist) a crosswalk table or logic that maps `legacy_account_id -> crm_account_uuid` for historical records.

> ? Required: Backfill or remap history using the crosswalk, or coalesce keys during transition.

3.2. `fct_subscriptions`: High Risk - Join Breakage

- This fact table likely has a foreign key to `dim_customer`.
- If that FK is

Engineering counterpart
6:04 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response
6:04 AM - Responded in 43s

Excellent question.

Before shipping the column drop and associated migration to production, you must verify a comprehensive set of data, functional, operational, and business controls to prevent data loss, downtime, or downstream breakage.

Here's a structured checklist of what to verify, grouped by category:

? 1. Data Completeness & Coverage

Verify:
- 100% population of `crm_account_uuid` for all active accounts
  - Current state: 94.2% populated, 1,204 legacy-only accounts
  - ? Confirm these 1,204 have been migrated or are inactive/archived
  - ? Run:
    ```sql
    SELECT COUNT()
    FROM raw.customers
    WHERE crm_account_uuid IS NULL
      AND is_active = TRUE;
    ```
    -> Must return 0

- Bidirectional mapping exists and is accurate
  - A persistent `legacy_account_id <-> crm_account_uuid` crosswalk table exists
  - No duplicates or one-to-many mismatches
  - Verified via:
    ```sql
    SELECT legacy_account_id, COUNT(DISTINCT crm_account_uuid)
    FROM xwalk_customer_ids
    GROUP BY 1 HAVING COUNT() > 1;
    ```
    -> Should return zero rows

- Historical backfill completeness
  - All historical customer records in `dim_customer` have been updated with `crm_account_uuid` where possible
  - Gaps documented and accepted (e.g., sunsetted accounts)

? 2. Model Logic & Key Migration

Confirm:
- All models have been refactored to use `crm_account_uuid` as primary key
  - `stg_customers`: No direct references to `legacy_account_id` (except for fallback during transition)
  - `dim_customer`: Natural key is now `crm_account_uuid`
  - SCD2 logic functions correctly on new key
- No hard dependencies remain
  - Run lineage scan (via dbt or data catalog) to confirm `legacy_account_id` is not selected anywhere in production marts
  - Use:
    ```bash
    dbt ls --select "+configured.columns.legacy_account_id"
    ```

```
-> Should return nothing

- Join integrity verified
  - `fct_subscriptions` joins to `dim_customer` on `crm_account_uuid` successfully
  - Row counts and key coverage match expectations:
    ```sql
    SELECT
      COUNT() AS total_subscriptions,
      COUNT(dim.customer_key) AS matched_to_customer
    FROM fct_subscriptions fct
    LEFT JOIN dim_customer dim
      ON fct.crm_account_uuid = dim.crm_account_uuid
    ```
    -> Match rate should be ?99.9%

? 3. Downstream System Impact

External Consumers:
- Salesforce sync (`customer_360_snapshot`):
  - Field mapping updated to use `crm_account_uuid`
  - API payloads validated - no rejected records due to missing/changed ID
  - Test sync completed in staging Salesforce org

- ML / Feature Store (`churn_training_features`):
  - Feature store re-registered with `entity_id = crm_account_uuid`
  - Model training pipeline uses new key
  - Historical features accessible via remapped entity

- BI Reports & Dashboards:
  - All dashboards (Looker, Tableau, etc.) using `legacy_account_id` have been updated
  - Verified via usage logs or tagging in data catalog

? 4. Operational & Observability Checks

Before Drop:
- Run in "dry mode" with column flagged as deprecated
  - Replace `SELECT  FROM raw.customers` with explicit column
    ```

Engineering counterpart
6:04 AM - Responded in 0s

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

Agent response
6:05 AM - Responded in 25s

Excellent follow-up. Even with careful planning, schema changes at the root of the data pipeline carry high systemic risk - especially when legacy identifiers are involved and downstream systems are tightly coupled.

Let's break this down clearly:

?? Top Risks of Dropping `legacy_account_id`

Risk	Severity	Description	Trigger Condition
1. Silent Data Loss in Analytics	? Critical	`dim_customer` SCD2 logic fails to track history under new key -> customer history breaks or duplicates form	Incomplete crosswalk or flawed key migration logic
2. Join Failures in Fact Tables	? High	`fct_subscriptions` fails to join to `dim_customer` -> subscription data disappears from reports	`crm_account_uuid` missing or mismatched on active records
3. ML Model & Feature Store Incompatibility	? High	`churn_training_features` uses old entity ID -> model serving fails or returns stale predictions	Feature store not re-synced with new key
4. Salesforce Sync Failures	? Medium	`customer_360_snapshot` exports missing/invalid IDs -> CRM sync errors or lost tracking	Salesforce integration not updated or key format mismatch
5. Uncaught Model Dependencies	? High	Hidden or undocumented model (e.g., ad hoc, marketing DB) still selects `legacy_account_id` -> query fails post-drop	Incomplete lineage scan or shadow analytics usage
6. Rollback Cannot Restore Data	? Medium	Once column is dropped and pipeline runs, downstream models may overwrite or lose legacy context	No backup of raw data or staging layers

> ? Biggest Fear: Not a loud crash - but silent corruption: data still flows, but joins are incomplete, history is broken, and ML models degrade - and no one notices for days.

? Rollback Plan: "If This Goes Wrong"

We assume the worst: after deployment, critical reports are missing data, joins fail, and teams alert.

? Goal of Rollback:
1. Minimize downtime
2. Restore full data integrity
3. Preserve ability to re-attempt migration safely

?? Step-by-Step Rollback Procedure

? Phase 1: Immediate Containment (0-15 mins)
- Pause dbt scheduler or DAGs to prevent corrupted data from propagating
- Alert stakeholders: Analytics, ML, Salesforce teams - "Migration paused due to data integrity issue"

? Phase 2: Restore Raw Layer Access
- Re-add `legacy_account_id` to `raw.customers` by:
  - Restoring the source table from backup (if dropped), OR
  - Re-enabling ingestion of the column in the ETL pipeline (e.g., Fivetran, Airbyte, custom loader)
  - If using a data lake (e.g., Iceberg, Delta), time-travel to pre-drop version:
    ```sql
    -- Example: Delta Lake
    ```

RESTORE TABLE raw.customers TO VERSION AS OF 123;
```

> ? Requires: Time-travel enabled or raw data backups retained for ?7 days

#### ? Phase 3: Revert Model Deployments
- Deploy previous version of dbt project (pre-migration):
  ```bash
  git checkout release/v1.4  # before drop
  dbt deploy -t production
  ```
- Ensures `stg_customers`, `dim_customer`, etc., fall back to using `legacy_account_id`

? Phase 4: Backfill Critical M
  ```

Engineering counterpart
6:05 AM - Responded in 0s

Can you tighten this into concrete next steps with owners and timing?

Agent response
6:05 AM - Responded in 17s

Absolutely - here's a tight, actionable execution plan with owners, deliverables, and timing, designed for real-world deployment and risk control.

? Concrete Next Steps: Drop `legacy_account_id` Migration

? Goal: Safely retire `legacy_account_id` by 2026-07-12 (CRM cutover)
? Do not proceed with drop until all prior steps are verified

---

? Timeline & Execution Plan

Week	Step	Owner	Deliverable	Status
Wk 1 (Now - 2026-06-15)	? Refactor Staging & Identity Models Update `stg_customers` and `int_customer_identity` to use `COALESCE(crm_account_uuid, legacy_account_id)` as `account_key`	Data Engineer	PR merged, deployed to staging	?
	?? Validate Identity Crosswalk Verify 1:1 mapping of `legacy_account_id <-> crm_account_uuid` and backfill gaps	Data Engineer + CRM Eng	Audited mapping table, with reconciliation report	?
	? Lineage Sweep Scan all models, dashboards, and APIs for use of `legacy_account_id`	Data Lead	Lineage report + list of consumers to update	?
Wk 2 (2026-06-16 - 2026-06-22)	? Migrate & Test Marts in Staging Deploy updated `dim_customer`, `fct_subscriptions`, and downstream models using `crm_account_uuid` as key	Data Engineer	Green CI/CD run, test queries validated	?
	? Backfill Historical Records Run backfill of `dim_customer` SCD2 using crosswalk	Data Engineer	Confirmation: 100% of active customers mapped	?
	? Validate Join Integrity Compare row counts and key coverage pre/post migration	Analytics Engineer	Match rate ?99.9% confirmed	?
Wk 3 (2026-06-23 - 2026-06-29)	? Update Downstream Systems Coordinate with Salesforce and ML teams to switch to `crm_account_uuid`	Data Eng + CRM Ops + ML Engineer	Salesforce sync test passed; Feature store re-registered	?
	? Announce Deprecation Send comms to analytics teams: "`legacy_account_id` to be dropped on 2026-07-05 (final week)"	Data Lead	Email + Slack alert sent	?
Wk 4 (2026-06-30 - 2026-07-05)	? Final Go/No-Go Check Confirm: 100% `crm_account_uuid` coverage, all models migrated, no active dependencies	Tech Lead + Data Lead	Go/No-Go sign-off document	?
	? Enable Monitoring & Alerting Deploy dbt tests, volume checks, null key alerts for `crm_account_uuid`	Data Engineer	Alerts active in production	?
Wk 5 (2026-0				

Engineering counterpart
6:05 AM - Responded in 0s

What's the one thing you'd want a second pair of eyes on?

Agent response
6:06 AM - Responded in 47s

The one thing I'd want a second pair of eyes on is:

> ? The completeness and correctness of the `legacy_account_id` -> `crm_account_uuid` crosswalk mapping - especially for the remaining 1,204 legacy-only accounts.

? Why This?
- This mapping is the linchpin of the entire migration.
- If even 1% of these accounts are mis-mapped or missing, it will cause:
  - Silent data loss in `dim_customer` and `fct_subscriptions`
  - Broken joins that look like "data dropped" but are actually key resolution failures
  - Churn model inaccuracies (e.g., returning zero features for real customers)
- These errors may not surface immediately in summary metrics - they erode data trust over time.

? What a Peer Should Review:
1. Is the crosswalk source of truth reliable?
   -> Was it generated during CRM migration? From a trusted reconciliation job?

2. Are the 1,204 unmapped accounts truly inactive or safe to ignore?
   -> Verified via:
   ```sql
   SELECT
     COUNT(),
     SUM(revenue_last_12m),
     MAX(last_activity_at)
   FROM raw.customers
   WHERE crm_account_uuid IS NULL
   ```
   -> A single active high-revenue account here invalidates the go-ahead.

3. Is the mapping bidirectional and conflict-free?
   -> No one-to-many or many-to-one mismatches:
   ```sql
   -- Check for duplicates
   SELECT crm_account_uuid, COUNT()
   FROM xwalk_customer_ids
   GROUP BY 1 HAVING COUNT() > 1
   ```

4. Is the crosswalk durable?
   -> Stored as a versioned, queryable table (not a transient notebook output)

? Final Sign-Off Ask:
> "Please confirm: We can correctly and confidently resolve every active customer through the new key, and any unmapped accounts are intentionally out of scope."

This isn't just a data check - it's a business continuity validation.
One extra pair of eyes here prevents a month of firefighting later.

Engineering counterpart
6:06 AM - Responded in 0s

Thanks - impact read and migration risks are clear. Migrate unmigrated accounts before dropping the legacy column in prod. Review complete - no further response needed.