



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: openai/gpt-4.1-nano

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

Duration: 1m 23s

Actual run cost: \$0.001740

Projected execution cost: \$0.3828/mo (10/biz day × 22 days)

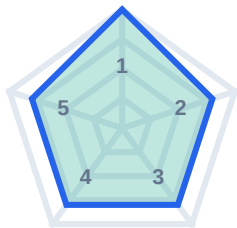
Report date: July 14, 2026 - 3:02 AM EST

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: 18235 chars - Latency: 83s - 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: 1m 23s

Engineering counterpart

7:01 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

7:01 AM - Responded in 18s

Based on the lineage and dependencies provided, here's a comprehensive impact assessment regarding dropping the `legacy\_account\_id` column from `raw.customers`:

1. Affected Marts/Models & Impact Severity

| Target Model | Usage Description | Impact Level | Comments |
|-------------------------------------|---|--------------|--|
| ----- ----- ----- ----- | | | |
| `analytics.dim_customer` | Uses `legacy_account_id` as a natural key in SCD Type 2 merge | High | Removal invalidates the natural key, risking data integrity and SCD merge failures. Requires re-engineering. |
| `analytics.fct_subscriptions` | Joins on `legacy_account_id` to `dim_customer` | High | Join will break; downstream reports and analytics relying on this join will be affected. |
| `analytics.customer_360_snapshot` | Uses in exports to Salesforce | Medium | Export data relying on `legacy_account_id` may be incomplete or invalid. |
| `analytics.churn_training_features` | Uses `legacy_account_id` as an entity ID in feature store | Medium | Model training and churn predictions may be compromised if IDs are missing or inconsistent. |

2. Migration Risks for Unmigrated Accounts

- Current State: Approximately 94.2% of `crm\_account\_uuid` are populated; 1,204 accounts are legacy-only.
- Risks:
 - Data Loss / Mismatch: The 1,204 legacy-only accounts will lose their identifiers if the column is dropped before migration.
 - Broken Dependencies: Marts relying on legacy IDs will be invalidated or incomplete, especially for high-impact models (`dim\_customer`, `fct\_subscriptions`).
 - Operational Disruption: Downstream reporting and ML workflows may break or produce inconsistent results until migration is completed.
 - Progressive Degradation: Since most accounts are migrated, risks are concentrated on legacy-only accounts, but impacts extend to all dependent systems if not handled properly.

3. Recommended Migration Order

Phase 1: Preparation

- Ensure all `legacy\_account\_id` are mapped and migrated to `crm\_account\_uuid`.
- Update `staging.stg\_customers` and `intermediate.int\_customer\_identity` to support `crm\_account\_uuid`.
- Confirm that all downstream marts have been evaluated for ID dependencies.

Phase 2: Migration

- Step 1: Backfill and verify `crm\_account\_uuid` for legacy-only accounts.
- Step 2: Test downstream systems (`dim\_customer`, `fct\_subscriptions`) with `crm\_account\_uuid` as the natural key, updating SCD and join logic.
- Step 3: Gradually switch dependencies from legacy IDs to `crm\_account\_uuid` in marts.
- Step 4: Once all systems are validated, formalize the deprecation of `legacy\_account\_id`.

Phase 3: Drop Column

- Post successful migration (target date before 2026-07-12), drop `legacy\_account\_id`.

4. Rollback Considerations

- Pre-drop backups: Ensure full backups/snapshots of `raw.customers` and dependent marts before dropping.
- Fallback: Maintain the `legacy\_account\_id` column until all critical systems are thoroughly tested and migrated.
- Rollback plan: If issues arise post-drop, restore from backups or reintroduce the `legacy\_account\_id` column, and re-run migration scripts.

Summary & Recommendations

- Urgency: Prioritize migration for legacy accounts (~1,204) to prevent data gaps.
- Communication: Inform downstream teams of the upcoming change and coordinate updates to dependent models.
- Incremental approach: Gradually phase out the legacy ID by first migrating

Engineering counterpart
7:01 AM - Responded in 0s

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

Agent response

7:02 AM - Responded in 19s

Certainly! Here's a step-by-step walkthrough of my reasoning process, including the assumptions I made:

Step 1: Understanding the Proposed Change

- The engineering team proposes dropping the `legacy\_account\_id` column from `raw.customers`.
- The reason is that it's deprecated after CRM migration, replaced by `crm\_account\_uuid`.
- Key assumption: The CRM migration has been largely completed (~94.2% accounts migrated), but some accounts (1,204) remain with only legacy IDs.

Step 2: Identifying Downstream Dependents

- The `legacy\_account\_id` is directly used in:
 - `staging.stg\_customers`
 - `intermediate.int\_customer\_identity`
- It's also used downstream in:
 - `analytics.dim\_customer` (as a natural key in SCD2)
 - `analytics.fct\_subscriptions` (join on legacy ID)
 - `analytics.customer\_360\_snapshot` (export to Salesforce)
 - `analytics.churn\_training\_features` (entity ID in feature store)

Assumption: The `legacy\_account\_id` is integral to these models' logic. Removing it without migration would break their data pipelines or produce invalid outputs.

Step 3: Impact Severity per Downstream Model

- High Impact: Models that rely directly on the legacy ID as a key or for joins, especially `dim\_customer` and `fct\_subscriptions`.
- Medium Impact: Models that use the ID in exports or features, but might have some fallback or alternative identifiers.

Assumption: The models do not currently have alternative identifiers or fallback mechanisms; thus, dropping the column would cause failures or data inconsistencies.

Step 4: Assessing Migration Risks

- The 94.2% population with `crm\_account\_uuid` suggests most accounts are migrated.
- The remaining 1,204 legacy-only accounts pose the highest risk, as they lack `crm\_account\_uuid`.
- Assumption: These accounts are still active or relevant; dropping the ID before migrating them would result in data gaps or orphaned records.

Step 5: Defining Migration Strategy

- To mitigate risks, a migration plan must:
 - Map remaining legacy accounts to `crm\_account\_uuid`.
 - Update all dependent models to use `crm\_account\_uuid` instead.
 - Validate migration success before dropping the column.

Assumption: The migration process will be sufficiently tested and thoroughly validated before dropping the column.

Step 6: Timing and Rollback Planning

- The deadline is July 12, 2026, aligned with CRM cutover.
- Assumption: The migration timeline will be planned such that dropping the column occurs after the successful migration of legacy-only accounts.
- Rollback involves restoring the column if issues arise; hence, a backup strategy must be in place.

Step 7: Final Recommendations & Assumptions

- Proceed incrementally, starting with migrating remaining legacy accounts.
- Update dependent models to prefer the `crm\_account\_uuid`.
- Only drop the column after confirming all systems are migrated and validated.

Summary of Key Assumptions

- The CRM migration is mostly complete (~94.2%) and can be finalized.
- The 1,204 legacy-only accounts are active or important, requiring migration before dropping.
- Downstream systems rely heavily on `legacy\_account\_id`, with no current fallback.
- Adequate testing, validation, and backups will be in place before dropping the column.
- The timeline allows for a phased migration, with a cutoff before the 2026 deadline.

Would you like me to elaborate

Engineering counterpart
7:02 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

7:02 AM - Responded in 13s

Before deploying the column drop to production, I would verify a comprehensive set of checks to ensure data integrity, system stability, and minimal disruption. Here's an outline of the key verification steps:

1. Data Migration Completeness & Accuracy

- Verify migration status: Confirm that all legacy account IDs, especially the remaining 1,204 legacy-only accounts, have been successfully migrated to `crm\_account\_uuid`.
- Data consistency checks: Ensure that for these accounts:
 - The new `crm\_account\_uuid` is correctly populated.
 - No discrepancies exist between legacy IDs and new UUIDs.
 - Cross-reference source systems and migration logs.

2. Downstream Model Readiness

- Update and test dependent models (`dim\_customer`, `fct\_subscriptions`, etc.):
 - Confirm they are refactored to use `crm\_account\_uuid` as the primary key or join key.
 - Run unit and integration tests to verify that joins, SCD merges, and aggregations behave correctly with the new key.
 - Check that no errors occur during data loads or refreshes with the new identifiers.
- Validate critical reports and dashboards that rely on these models-ensure they produce consistent, correct results post-migration.

3. System and Data Lineage Validation

- Use lineage tools or scripts to verify that:
 - No active dependencies or workflows still rely solely on the `legacy\_account\_id`.
 - All pipelines referencing the column have been updated.

4. Backup and Rollback Preparedness

- Take full backups of:
 - The `raw.customers` table.
 - Downstream marts and models.
- Prepare a rollback plan:
 - Ensure schemas with the `legacy\_account\_id` column are versioned or can be restored quickly.
 - Test restoration procedures.

5. Testing in a Staging Environment

- Replicate the production environment as closely as possible.
- Perform a dry run:
 - Drop the `legacy\_account\_id` column.
 - Run all associated migration scripts, tests, and validations.
- Confirm that:
 - No errors or failures occur.
 - All dependent systems operate correctly.
 - Data remains consistent and accurate.

6. Validation of Edge Cases and Legacy-only Accounts

- Specifically test with legacy-only accounts:
 - Confirm that their data is migrated correctly.
 - Check that they are correctly handled in downstream models.

7. Stakeholder Sign-off

- Obtain sign-off from data owners, analysts, and downstream consumers confirming:
 - The migration is complete.

- The models and reports are correct.
- The column drop will not cause surprise issues.

Summary

In essence, I would verify:

- Migration completeness and correctness.
- Downstream systems are updated and validated.
- Backup and rollback plans are in place and tested.
- Testing replicates production scenarios.
- All stakeholders are aligned and informed.

Only after thorough verification of these points should the drop be executed in production to ensure a safe and smooth transition.

Engineering counterpart
7:02 AM - Responded in 0s

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

Agent response

7:02 AM - Responded in 11s

Top Risks:

1. Data Inconsistency or Loss for Legacy-Only Accounts

- The 1,204 legacy-only accounts may lose their identifiers if the column is dropped prematurely.
- Downstream models (like `dim\_customer`, `fct\_subscriptions`) that rely on `legacy\_account\_id` might fail or produce incomplete data.

2. Broken Downstream Dependencies

- Critical processes, reports, or ML models that depend on `legacy\_account\_id` could break if they haven't been migrated to use `crm\_account\_uuid`.
- This could cause data pipeline failures, incorrect analytics, or disrupted operations.

3. Incomplete or Incorrect Migration

- If migration scripts for legacy accounts are incomplete, some accounts might still refer to `legacy\_account\_id`, leading to inconsistent states.
- Differences between environments (staging vs. production) might cause unexpected failures.

4. Timelines and Coordination Risks

- Delay in migration completion or testing could push the production drop before readiness, risking operational impact.
- Lack of stakeholder communication could lead to unexpected failures.

5. Operational Disruption During Drop

- Unexpected errors during the drop could cause outages or partial data unavailability.

Rollback Plan:

- Pre-Drop Backups:

- Take full backups of the `raw.customers` table, including schema and data.
- Backup all downstream marts and models impacted by the change.
- Ensure version control or schema snapshots are available.

- Restoring the Column:

- If issues are detected immediately after drop, restore the `legacy\_account\_id` column:
- Use backups to re-add the column and restore its data.
- Revert downstream models to previous versions if necessary.

- Pause All Dependent Processes:

- Halt affected ETL/ELT workflows until data integrity is confirmed.
- Communicate with stakeholders about the rollback status.

- Investigation and Issue Resolution:

- Identify the root cause of the failure.
- Correct migration gaps or system bugs.
- Re-validate data and system health.

- Post-Restoration Actions:

- Once restored, verify data integrity and system stability.
- Reassess and finalize any outstanding migration tasks.
- Plan a controlled retry of the drop after addressing issues.

Summary:

- The highest-priority rollback measure is data and schema backups before dropping.
- If the drop causes failures or data gaps, revert to backups and restore the previous schema.
- Implement monitoring and communication protocols for rapid detection and response.
- Ensure that migration completeness and testing are verified before the next attempt.

Would you like a detailed step-by-step rollback checklist or contingency communication plan?

Engineering counterpart
7:02 AM - Responded in 0s

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

Agent response

7:02 AM - Responded in 17s

Certainly! Here's a structured, actionable plan with clear owners and timing to ensure a safe and coordinated approach for dropping the `legacy\_account\_id` column.

Concrete Next Steps for Column Drop Readiness

1. Final Data Migration & Validation

- Owner: Data Engineering Team Lead / Migration Lead
- Timing: Within the next 2 weeks (e.g., by YYYY-MM-DD)
- Actions:
 - Confirm that migration scripts for the remaining 1,204 legacy-only accounts are completed.
 - Validate that all legacy account IDs are successfully mapped to `crm\_account\_uuid`.
 - Run data validation queries to ensure no residual legacy IDs exist in `raw.customers` and downstream models.
 - Document the migration status.

2. Dependency and Impact Assessment

- Owner: Data Platform Architect / BI Lead
- Timing: Within 1 week (e.g., YYYY-MM-DD)
- Actions:
 - Review and update all downstream processes that depend on `legacy\_account\_id`.
 - Ensure all models, reports, and workflows are refactored to use `crm\_account\_uuid`.
 - Perform regression testing in staging.

3. Backups & Testing in Staging

- Owner: Data Operations / Engineering Ops
- Timing: Immediately after validation (e.g., YYYY-MM-DD)
- Actions:
 - Take full backups of `raw.customers`, relevant downstream marts, and models.
 - Execute a dry run of dropping the `legacy\_account\_id` in a staging environment.
 - Validate data integrity and process stability.

4. Stakeholder Validation & Sign-off

- Owner: Data Product Owner / Business Stakeholders
- Timing: Within 2 days after staging testing (e.g., YYYY-MM-DD)
- Actions:
 - Review test results.
 - Confirm readiness to proceed.
 - Obtain formal sign-off to proceed to production.

5. Final Pre-Deployment Check & Communication

- Owner: Project Manager / Release Manager
- Timing: 1 day before scheduled drop (e.g., YYYY-MM-DD)
- Actions:
 - Notify all stakeholders of scheduled change window.
 - Confirm backup verification.
 - Prepare rollback instructions and resources.

Execution & Rollback Plan

6. Production Drop

- Owner: Data Engineering Lead / Deployment Lead
- Timing: Scheduled date, e.g., YYYY-MM-DD, during low-traffic window
- Actions:
 - Perform the drop of `legacy\_account\_id`.
 - Monitor logs and system health closely.
 - Validate key downstream jobs and reports.

7. Post-Drop Monitoring

- Owner: Data Ops / Support Team
- Timing: Immediately after drop for 24-48 hours
- Actions:
 - Monitor for failures or anomalies.
 - Verify data completeness and correctness in downstream systems.
 - Confirm key stakeholders are satisfied.

8. Rollback if Necessary

- Owner: Data Engineering Lead / Data Ops
- Trigger: Detection of critical failure or data inconsistency
- Actions:
 - Initiate restoration from backups.
 - Revert any schema changes.
 - Communicate issues and delay further migration steps until resolved.

Summary Table

| Step | Owner | Timing | Key Deliverables |
|--|-------------------------|-----------------------------|--|
| Final migration validation migration | Data Migration Lead | Within 2 weeks | Confirmed complete and validated migration |
| Dependency assessment & refactor tests | Data Platform Architect | Within 1 week | Updated models, regression tests |
| Backup & staging tests | Data Ops | Immediately post-validation | Successful dry-run in staging |
| Stakeholder communication | Stakeholders | Ongoing | Clear communication and buy-in |

Engineering counterpart

7:02 AM - Responded in 0s

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

Agent response

7:02 AM - Responded in 5s

The one thing I would want a second pair of eyes on is the completeness and correctness of the migration of the remaining legacy-only accounts-specifically, verifying that all 1,204 accounts have been successfully migrated to `crm\_account\_uuid` and that no residual legacy IDs remain in `raw.customers` or downstream systems.

Why? Because dropping the `legacy\_account\_id` without ensuring that all accounts are fully migrated and that no dependencies rely solely on the legacy ID poses a significant risk of data gaps or process failures. An independent review of the migration validation results, data quality checks, and impact assessments would provide additional assurance that the migration is truly complete and safe to proceed with the drop.

In summary:

> I'd want someone to double-check the migration completion reports, data validation scripts, and impact assessments, ensuring no overlooked legacy IDs or dependencies remain.

Engineering counterpart

7:02 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.