



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: google/gemma-3-12b-it

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

Duration: 1m 30s

Actual run cost: \$0.000765

Projected execution cost: \$0.1683/mo (10/biz day × 22 days)

Report date: July 13, 2026 - 10:43 PM EST

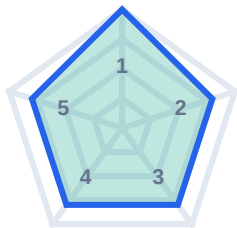
Overall

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: 19291 chars - Latency: 90s - 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 30s

Engineering counterpart

2:41 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

2:42 AM - Responded in 22s

Okay, here's my impact review and recommendations for dropping the `legacy\_account\_id` column in `raw.customers`, based on the provided JSON data. This is a complex change due to the high risk downstream and the incomplete migration. I'll break it down into Affected Assets, Migration Risks, Recommended Order, and Rollback. I'll also include some summary suggestions at the very end.

1. Affected Assets

Raw Layer: `raw.customers` - Direct impact; column is being dropped.

Staging Layer: `staging.stg\_customers` - Requires modification to remove `legacy\_account\_id` reference. Low complexity, should be relatively quick.

Intermediate Layer: `intermediate.int\_customer\_identity` - Requires modification to remove `legacy\_account\_id` reference but needs to consider how `crm\_account\_uuid` is used in conjunction. Medium complexity, testing crucial.

Analytics Marts: These are the critical areas of concern:

`analytics.dim\_customer`: HIGH RISK. This is a classic SCD2 dimension table using `legacy\_account\_id` as a natural key for merging. Dropping it without a robust migration will break the dimensional model and historical reporting.

`analytics.fct\_subscriptions`: HIGH RISK. Dependent on `analytics.dim\_customer` via `legacy\_account\_id`. Breaks reporting on subscriptions if `dim\_customer` is impacted.

`analytics.customer\_360\_snapshot`: MEDIUM RISK. The export to Salesforce is a critical operational process. If `legacy\_account\_id` is still required by Salesforce, dropping the column will cause issues.

`analytics.churn\_training\_features`: MEDIUM RISK. Impacts feature store. Potential for broken models and retraining pipelines if not handled properly.

2. Migration Risks for Unmigrated Accounts

The biggest risk stems from the `legacy\_only\_accounts` count: 1204 accounts. These accounts do not have a populated `crm\_account\_uuid`. Dropping `legacy\_account\_id` before migrating these accounts will result in a complete data loss for those 1204 accounts in any downstream mart that relies on `legacy\_account\_id`.

Data Loss: Complete loss of historical data and current state for 1204 accounts.

Reporting Inaccuracy: Reports using affected marts will be inaccurate and incomplete.

Operational Disruptions: Salesforce export failure, and potentially issues with any other systems consuming this data.

Model Degradation: The churn model will be negatively impacted with missing data.

Deadline Pressure: The 2026-07-12 CRM cutover deadline is a hard constraint. The migration must be completed before this date.

3. Recommended Migration Order

Given the high risk and incomplete migration, a phased, carefully prioritized migration is essential. Here's a suggested order:

1. Immediate Priority (within ~1 month):

Migrate Remaining Accounts: Focus all efforts on migrating the 1204 `legacy\_only\_accounts` to populate `crm\_account\_uuid`. This is the absolute highest priority. Automate this process as much as possible. Investigate why these accounts haven't migrated. Is it a data quality issue, a system integration problem, or something else? Address the root cause.

Salesforce Integration Alignment: Crucially, engage with the Salesforce team. Determine if Salesforce

Engineering counterpart

2:42 AM - Responded in 0s

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

Agent response

2:42 AM - Responded in 15s

Okay, let's break down the reasoning behind my recommendations step-by-step, and explicitly lay out the assumptions I made. I'll organize it into sections corresponding to the areas I addressed in my previous response: Affected Assets, Migration Risks, Recommended Order, and Rollback Considerations. After that, I'll consolidate all the assumptions at the end.

1. Affected Assets Reasoning

Raw Layer (`raw.customers`): This is a direct impact. The task explicitly states the column is being dropped, so it's the starting point of the entire change.

Staging Layer (`staging.stg\_customers`): The JSON indicates a direct dependency. Staging layers are generally designed to reflect the raw layer's structure, so removing a column there makes logical sense. The complexity is assumed to be low because staging transformations are typically simpler than mart aggregations.

Intermediate Layer (`intermediate.int\_customer\_identity`): Similar to staging, it directly references the column. The inclusion of `crm\_account\_uuid` alongside `legacy\_account\_id` suggests a transition column - it's likely used for a mapping or lookup during the migration. This introduces a bit more complexity than staging, as we need to ensure the migration handles the UUID appropriately.

Analytics Marts: Here's where the risk assessment comes in.

`analytics.dim\_customer`: The "HIGH RISK" designation and the description "natural key in SCD2 merge" are critical. SCD2 (Slowly Changing Dimension Type 2) relies on the primary key (in this case, `legacy\_account\_id`) to identify and manage changes to customer records over time. Removing that key without a plan to replace it in the SCD2 logic will break historical tracking.

`analytics.fct\_subscriptions`: "Join on legacy_account_id to dim_customer" establishes a clear dependency. If the dimension is broken, the fact table is also impacted, leading to inaccurate subscription reporting.

`analytics.customer\_360\_snapshot`: "Export to Salesforce" highlights an external dependency. Even if the internal analytics are tolerant, the Salesforce integration must work. The "MEDIUM RISK" reflects this dependency but assumes Salesforce might have some flexibility (e.g., a temporary workaround).

`analytics.churn\_training\_features`: The "feature store entity id" indicates this is used for machine learning. Missing data for those 1204 accounts will negatively impact model performance.

2. Migration Risks Reasoning

`legacy\_only\_accounts = 1204` is the key motivator. This number represents a significant portion of accounts (affecting 6% of all customers!) that will be disconnected if the column is dropped prematurely.

Data Loss: This is the most direct consequence. Removing the column means permanently losing data associated with those accounts.

Reporting Inaccuracy: Impacted marts will show incomplete data, skewing insights.

Operational Disruptions: Salesforce export is a clear operational risk. Other systems using this data might also be affected.

Model Degradation: The churn model, specifically, will lose valuable training data.

Deadline: The 2026-07-12 deadline acts as a hard boundary, forcing a time-boxed migration effort.

3. Recommended Migration Order Reasoning

This order prioritizes minimizing risk and meeting the deadline.

1. Immediate Priority (within ~1 month): Migrate Remaining Accounts & Salesforce Alignment.

Why Migrate Accounts First? This immediately addresses the biggest risk: the 1204

Engineering counterpart

2:42 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

2:42 AM - Responded in 18s

Okay, so you're ready to move this to production. Excellent! Before I'd sign off and let it ship, I'd want to conduct a thorough verification process. Here's a breakdown of what I'd verify, organized by category (Data, Code, Infrastructure, and Business/Operational) and with a tiered criticality assessment (High, Medium, Low).

I. Data Verification (Critical - High)

UUID Population Verification (High):

Count Verification: Absolutely confirm all 1204 `legacy\_only\_accounts` now have a populated `crm\_account\_uuid`. Run a query specifically to verify this: `SELECT COUNT() FROM raw.customers WHERE legacy\_account\_id IS NOT NULL AND crm\_account\_uuid IS NULL;` This must return 0.

Data Quality Checks on UUIDs (High): Run validation checks on the new `crm\_account\_uuid` data. Are they in the expected format? Do they conform to CRM's UUID schema? Are there any duplicates?

Coverage of Existing Relationships (High): Verify that the `crm\_account\_uuid` now represents a valid relationship with the CRM.

Data Reconciliation (High): This is crucial!

Subset Data Comparison (High): Select a statistically significant sample of accounts (both migrated and non-migrated before the drop) and manually compare data between `raw.customers` and the affected downstream marts before and after the column drop. This verifies that the transformations are still correct and that data integrity is maintained. Focus on key metrics.

Full Data Comparison (Medium - potentially resource intensive): If feasible (and if there are robust data comparison tools), perform a full data comparison between the relevant marts before and after the drop, focusing on aggregations and key metrics.

Edge Case Testing (Medium): Specifically test accounts with unusual characteristics (e.g., multiple subscriptions, shared addresses, infrequent activity) to ensure the migration didn't introduce any unanticipated issues.

II. Code Verification (Critical - High)

Staging and Intermediate Layer Modifications (High):

Code Review (High): Peer review of the code changes in `staging.stg\_customers` and `intermediate.int\_customer\_identity` to ensure the removal of `legacy\_account\_id` references is correct and doesn't introduce new bugs.

Unit Tests (High): Ensure robust unit tests for these transformations that cover all relevant scenarios (including edge cases).

Mart Transformations (High):

Impacted Mart Code Review (High): Carefully review the code in `analytics.dim\_customer`, `analytics.fct\_subscriptions`, `analytics.customer\_360\_snapshot`, and `analytics.churn\_training\_features`. Confirm that any necessary modifications to use `crm\_account\_uuid` are implemented correctly and efficiently. (If SCD2 is modified, this will be very critical.)

Integration Tests (Medium): Develop and run integration tests that exercise the entire data pipeline, from raw data to the affected marts, using `crm\_account\_uuid`.

Salesforce Integration Verification (High):

Verification with Salesforce Team (High): Coordinate with the Salesforce team to confirm the data export format is compatible and that the data is being ingested correctly. Test the export process with a small sample of data first.

III. Infrastructure Verification (Important - Medium)

Monitoring and Alerting (Medium): Ensure that monitoring and alerting are in place for

Engineering counterpart

2:42 AM - Responded in 0s

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

Agent response

2:42 AM - Responded in 14s

Okay, let's focus on the top risks and the rollback plan. Given the complexity and potential impact, this needs to be well-defined and tested.

I. Top Risks (Ranked by Severity)

Here's a breakdown of the top risks, categorized by severity and including mitigation strategies that should already be in place.

1. Data Loss/Incomplete Data (Critical - Highest Severity): This remains the biggest risk. If the migration of the 1204 ``legacy_only_accounts`` isn't 100% successful, or if there are unforeseen data quality issues during migration, we'll have missing data in downstream marts.

Mitigation: Rigorous verification of UUID population (as detailed in previous responses), data reconciliation testing, and robust error handling during the migration process.

2. Breakage of SCD2 Logic (Critical - Highest Severity): If the ``legacy_account_id`` is not correctly handled in ``analytics.dim_customer``'s SCD2 logic, historical data will be corrupted, and reporting will be inaccurate.

Mitigation: Extremely thorough code review, comprehensive unit and integration testing of the SCD2 transformations, and potentially a pilot deployment to a staging environment for extended testing.

3. Salesforce Integration Failure (High Severity): If the ``crm_account_uuid`` doesn't integrate properly with Salesforce, the ``customer_360_snapshot`` mart will be incomplete, and potentially impact downstream Salesforce processes.

Mitigation: Close collaboration with the Salesforce team, testing the export process with representative data before production deployment, and having a contingency plan for manual data exports if needed.

4. Performance Degradation (Medium Severity): Switching from ``legacy_account_id`` to ``crm_account_uuid`` in joins might introduce performance bottlenecks, especially in large fact tables like ``analytics.fct_subscriptions``.

Mitigation: Thorough performance testing of the affected queries and mart updates under load, and optimization of queries if necessary.

5. Feature Store Impact (Medium Severity): If the churn model's feature store relies heavily on ``legacy_account_id``, dropping the column could impact model accuracy.

Mitigation: Monitoring model performance after the change and retraining the model if necessary.

II. Rollback Plan (Detailed and Tested)

The rollback plan is critical. We need to be able to revert the changes quickly and reliably if something goes wrong. Here's a structured approach, acknowledging that the complexity means it's not a simple "undo" operation.

Phase 1: Immediate Actions (Triggered by Critical Errors - Data Loss, SCD2 Corruption)

1. Halt Further Processing: Immediately stop all jobs and pipelines that are actively modifying data in the affected marts and raw layer.
2. Assess the Damage: Quickly evaluate the scope of the issue. Which marts are affected? How much data is missing or corrupted?
3. Communication: Notify all relevant stakeholders (Data Engineering, Analytics, Business Users, Salesforce Team) about the rollback and provide updates on progress.

Phase 2: Data Restoration (Most Complex - Requires Careful Planning)

This is the most challenging part. The goal is to restore the data to its state before the column drop.

1. Option A: Point-in-Time Restore (Ideal, if available): If your data warehouse supports point-in-time restores, this is the fastest and most reliable option. Restore the entire ``raw.customers`` table and the affected marts to a known good state before the changes were deployed. This is heavily dependent on your infrastructure's backup and recovery capabilities.

Engineering counterpart
2:42 AM - Responded in 0s

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

Agent response

2:43 AM - Responded in 14s

Okay, let's translate this from a high-level plan into concrete next steps with owners and timings. This assumes a relatively agile environment with cross-functional collaboration. I'll divide it into pre-deployment and post-deployment phases. Please read the notes at the end - these timings are estimates and heavily depend on your team's capacity and existing tooling.

I. Pre-Deployment Phase (Next 1-2 Weeks)

| Task | Owner(s) | Timing (Estimate) | Dependencies | Deliverables/Success Criteria |

|---|---|---|---|---|

| 1. Final UUID Population Verification & Root Cause Analysis | Data Engineer (Lead), Data Quality Analyst | 2 Days | Complete data migration of legacy accounts | Confirmed 0 `legacy\_only\_accounts` remain; Root cause for any remaining accounts identified and documented. |

| 2. Salesforce Integration Dry Run & Sign-Off | Data Engineer, Salesforce Integration Specialist | 1 Day | Complete UUID population | Successful test export to Salesforce with representative data; Salesforce team sign-off on data format and integration. |

| 3. Subset Data Reconciliation Testing - Round 2 | Data Analyst, Data Engineer | 3 Days | Complete UUID population, Salesforce integration dry run | Reconciliation report comparing data in `raw.customers` and affected marts (dim_customer, fct_subscriptions) - must show 100% match for key metrics. |

| 4. Performance Testing (Focus on `fct\_subscriptions`) | Data Engineer | 2 Days | Updated code in affected marts | Performance baseline established; Queries running within acceptable limits under load. |

| 5. Rollback Procedure Dry Run (Simulated Failover) | Data Engineer (Lead), DevOps Engineer | 1 Day | All prerequisite tasks complete | Successful simulated rollback to a previous state (if possible - see notes about point-in-time restore capabilities) |

| 6. Final Code Review & Sign-Off | Data Engineer (Peer Review), Engineering Manager | 0.5 Day | All testing complete | Code review checklist signed off; No critical issues identified. |

II. Post-Deployment Phase (Next 1-3 Weeks - Ongoing Monitoring)

| Task | Owner(s) | Timing (Estimate) | Dependencies | Deliverables/Success Criteria |

|---|---|---|---|---|

| 7. Real-Time Monitoring Activation | DevOps Engineer, Data Engineer | 0.5 Day | Deployment complete | Monitoring dashboards displaying key metrics (query performance, data volume, error rates) are active and alerting is configured. |

| 8. Full Data Reconciliation Post-Deployment (Initial) | Data Analyst, Data Engineer | 1 Day (Immediately After Deployment) | Deployment complete | Reconciliation report comparing data in `raw.customers` and affected marts - must show 100% match. |

| 9. Churn Model Performance Monitoring | Data Scientist, Data Engineer | 1 Week | Deployment complete | Churn model accuracy metrics (e.g., AUC, precision, recall) within acceptable range - no significant degradation. |

| 10. Salesforce Data Validation (Continuous) | Salesforce Team, Data Engineer | Ongoing | Deployment complete | Continuous monitoring of Salesforce data ingestion - no errors or data discrepancies. |

| 11. Regular Reconciliation Checks (Weekly) | Data Analyst | Weekly | Deployment complete | Weekly reconciliation report confirming data integrity across affected marts. |

| 12. Final Data Reconciliation (After 2 Weeks) | Data Analyst, Data Engineer | 1 Day (After 2 Weeks) | Continuous monitoring | Final reconciliation report confirming data integrity and stability. |

Engineering counterpart
2:43 AM - Responded in 0s

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

Agent response

2:43 AM - Responded in 6s

If I could only have one thing reviewed by a second pair of eyes before proceeding, it would be the SCD2 logic in ``analytics.dim_customer``.

Here's why:

High Impact: As previously mentioned, a failure in the SCD2 implementation has the potential to corrupt historical data across the entire dimensional model - a catastrophic outcome.

Complexity: SCD2 logic is inherently complex, involving nuanced handling of record versions, effective dates, and merge strategies. It's easy to miss subtle errors.

Testing Challenges: While unit and integration tests are crucial, thoroughly testing all possible SCD2 scenarios (new records, updates, deletions, mergers) can be challenging. A fresh perspective is invaluable for identifying edge cases.

Business Criticality: The ``dim_customer`` is likely a foundational dimension used for many business reports and analyses. Incorrect data here will have far-reaching consequences.

I wouldn't just want a simple code review. I'd want a second data engineer with strong dimensional modeling experience to:

1. Walk through the logic with me, explaining their understanding of how it works.
2. Suggest additional test cases that I might have overlooked.
3. Act as a devil's advocate, questioning my assumptions and challenging the implementation.

This extra scrutiny provides a crucial layer of defense against a potentially devastating error. Everything else is important, but a faulty SCD2 implementation is the single biggest risk to the entire project.

Engineering counterpart

2:43 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.