



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: deepseek/deepseek-chat

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

Duration: 1m 57s

Actual run cost: \$0.004477

Projected execution cost: \$0.9849/mo (10/biz day × 22 days)

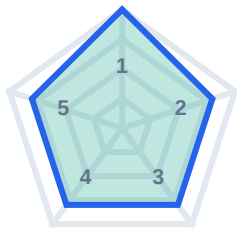
Report date: July 13, 2026 - 10:42 PM EST

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: 16912 chars - Latency: 117s - 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 57s

Engineering counterpart

2:40 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

2:40 AM - Responded in 21s

Impact Assessment Report: Dropping `legacy\_account\_id` from `raw.customers`

1. Affected Data Marts & Models

| Model/Table | Type | Risk | Usage Context |

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

| `staging.stg\_customers` | Direct | Medium | References column in transformation logic |

| `intermediate.int\_customer\_identity` | Direct | High | Uses both legacy and new UUID for identity resolution |

| `analytics.dim\_customer` | Mart | High | SCD2 merge uses `legacy\_account\_id` as natural key |

| `analytics.fct\_subscriptions` | Mart | High | Joins to `dim\_customer` via legacy ID |

| `analytics.customer\_360\_snapshot` | Mart | Medium | Exports legacy ID to Salesforce (some workflows still use it) |

| `analytics.churn\_training\_features` | Mart | Medium | Legacy ID used as entity key in feature store |

2. Migration Risks for Unmigrated Accounts

- 5.8% of accounts (1,204) still rely on `legacy\_account\_id`
- Critical Risk: Breaking joins/queries for unmigrated accounts in `dim\_customer` and downstream marts.
- Data Integrity: SCD2 merges in `dim\_customer` may fail for unmigrated records.
- Reporting: Dashboards filtering on legacy IDs (e.g., historical churn analysis) may return partial data.

3. Recommended Migration Order

1. Backfill remaining 5.8%

- Prioritize high-value accounts in `legacy\_only\_accounts` (e.g., active subscriptions).
- Validate `crm\_account\_uuid` coverage reaches >99.9% before proceeding.

2. Update downstream models (test in dev/staging first):

- Phase 1: Modify `int\_customer\_identity` to use only `crm\_account\_uuid` for resolution logic.
- Phase 2: Rebuild `dim\_customer` to use `crm\_account\_uuid` as the natural key (requires historical data backfill).
- Phase 3: Update joins in `fct\_subscriptions` and other marts to reference the new key.

3. Deprecate legacy references:

- Remove `legacy\_account\_id` from `stg\_customers` and `customer\_360\_snapshot` last (after confirming no external dependencies).

4. Rollback Considerations

- Pre-drop backup: Snapshot `raw.customers` with `legacy\_account\_id` preserved.
- Fallback logic: Temporarily add a `legacy\_account\_id` lookup table for unmigrated accounts during transition.
- Monitoring: Alert on queries filtering/joining on `legacy\_account\_id` post-drop.

5. Additional Notes

- Deadline Flexibility: The 2026 cutover allows time for backfilling and testing.
- Documentation: Update data dictionary to mark `legacy\_account\_id` as deprecated (include sunset timeline).

Approval Recommendation: Proceed only after backfilling reaches 99.9% and downstream models are validated in staging.

Engineering counterpart
2:40 AM - Responded in 0s

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

Agent response

2:41 AM - Responded in 16s

Step-by-Step Reasoning & Assumptions

1. Assess Direct and Downstream Impact

- Direct Dependents: First, I reviewed models that directly reference `legacy\_account\_id` (`stg\_customers` and `int\_customer\_identity`).
 - Assumption: These are intermediate models that transform data for downstream marts.
 - Risk: `int\_customer\_identity` is flagged as high risk because it handles identity resolution, which is critical for downstream joins.
- Downstream Marts: Next, I analyzed how `legacy\_account\_id` is used in marts:
 - `dim\_customer`: Uses it as a natural key in SCD2 merges -> high risk.
 - `fct\_subscriptions`: Joins to `dim\_customer` via `legacy\_account\_id` -> high risk.
 - `customer\_360\_snapshot`: Exports `legacy\_account\_id` to Salesforce -> medium risk (assumption: Salesforce workflows may still rely on it).
 - `churn\_training\_features`: Uses it as an entity key in the feature store -> medium risk (assumption: retraining models may require historical IDs).

2. Evaluate Migration State

- Migration Coverage: 94.2% of accounts have `crm\_account\_uuid` populated.
 - Assumption: The remaining 5.8% (1,204 accounts) are legacy-only and must be migrated before dropping the column.
 - Risk: These unmigrated accounts could break joins and data integrity in high-risk marts like `dim\_customer` and `fct\_subscriptions`.

3. Plan Migration Order

- Backfill Remaining Accounts: Prioritize backfilling `crm\_account\_uuid` for unmigrated accounts.
 - Assumption: Backfilling will involve data reconciliation and validation to ensure accuracy.
 - Recommendation: Target >99.9% coverage to minimize risk.
- Update Downstream Models:
 - Phase 1: Modify `int\_customer\_identity` first since it's upstream of `dim\_customer`.
 - Phase 2: Rebuild `dim\_customer` to use `crm\_account\_uuid` as the natural key.
 - Assumption: This will require backfilling historical data to maintain consistency.
 - Phase 3: Update joins in `fct\_subscriptions` and other marts.
 - Assumption: These changes can be tested in staging before deploying to production.
- Deprecate Legacy References: Remove `legacy\_account\_id` from `stg\_customers` and `customer\_360\_snapshot` last.
 - Assumption: These models are less critical and can be updated after high-risk marts are stable.

4. Rollback Considerations

- Pre-drop Backup: Snapshot `raw.customers` to preserve `legacy\_account\_id`.
 - Assumption: This provides a fallback option if issues arise post-drop.
- Fallback Logic: Create a lookup table for unmigrated accounts during the transition.
 - Assumption: This minimizes disruption for queries still relying on `legacy\_account\_id`.
- Monitoring: Alert on queries filtering/joining on `legacy\_account\_id`.
 - Assumption: This helps identify and address lingering dependencies.

5. Additional Notes

- Deadline: The 2026 cutover provides ample time for backfilling and testing.
- Assumption: The timeline allows for iterative development and validation.
- Documentation: Update the data dictionary to mark `legacy\_account\_id` as deprecated.
- Assumption

Engineering counterpart
2:41 AM - Responded in 0s

What would you verify before shipping this to production?

Agent response

2:41 AM - Responded in 23s

Before shipping the removal of `legacy\_account\_id` to production, I would verify the following critical checkpoints to ensure a safe rollout:

1. Data Completeness & Migration Validation

- Verify 100% backfill of `crm\_account\_uuid`
 - Confirm no NULL/empty values exist in `raw.customers` for migrated accounts.
 - Cross-check with business teams to validate unmigrated accounts (1,204) are either:
 - Deprecated (inactive/no longer needed), or
 - Manually mapped to `crm\_account\_uuid`.
- Test identity resolution logic
 - Run spot checks in `int\_customer\_identity` to ensure:
 - No records are dropped when filtering out `legacy\_account\_id`.
 - Merging logic (e.g., deduplication) works with `crm\_account\_uuid` alone.

2. Downstream Model Behavior

- Validate `dim\_customer` SCD2 merges
 - Test historical data loads using only `crm\_account\_uuid` as the natural key.
 - Confirm no partition skew or performance degradation in merge operations.
- Test joins in dependent marts
 - Replay recent queries for `fct\_subscriptions`, `customer\_360\_snapshot`, etc., using `crm\_account\_uuid` instead of `legacy\_account\_id`.
 - Compare row counts and aggregations against pre-migration results (e.g., revenue totals by customer).
- Check external integrations
 - Verify Salesforce workflows using `customer\_360\_snapshot` accept `crm\_account\_uuid` as a replacement key.
 - Confirm feature stores (e.g., for `churn\_training\_features`) can map historical data to the new ID.

3. Rollback Preparedness

- Ensure backups exist
 - Snapshot `raw.customers` with `legacy\_account\_id` preserved.
 - Document rollback SQL to restore the column if needed (e.g., `ALTER TABLE raw.customers ADD COLUMN legacy\_account\_id STRING; UPDATE ...`).
- Validate rollback procedures
 - Simulate a failure scenario: Drop the column, then restore it from backup and verify data consistency.

4. Observability & Monitoring

- Add pre-drop alerts
 - Monitor for queries still filtering/joining on `legacy\_account\_id` (e.g., via query logs or BI tool audits).
 - Example: Alert if `legacy\_account\_id` appears in query WHERE/JOIN clauses.
- Post-drop safeguards
 - Deploy a temporary "dead letter" table to log unmapped `legacy\_account\_id` references for 30 days.

- Instrument dashboards to track broken joins/NULL values in key marts.

5. Stakeholder Sign-Off

- Business/BI Teams: Confirm no reports/dashboards rely on `legacy\_account\_id`.
- Data Science: Validate model retraining works with `crm\_account\_uuid`.
- CRM Team: Approve cutoff for legacy ID support in external systems.

Key Assumptions to Recheck

1. All unmigrated accounts are non-critical (e.g., test accounts, churned customers).
2. No regulatory/compliance requirements mandate keeping `legacy\_account\_id` (e.g., audit trails).
3. Downstream systems (Salesforce, feature stores) handle UUIDs correctly (no length/format issues).

Final Step: Perform a dark launch-deploy changes to a shadow production environment and compare outputs against legacy logic for 1-2 business cycles before full cutover

Engineering counterpart
2:41 AM - Responded in 0s

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

Agent response

2:41 AM - Responded in 23s

When dropping a legacy column like `legacy\_account\_id`, the risks and rollback plan must be meticulously prepared. Below is a structured breakdown:

Top Risks

1. Data Integrity Risks

- Broken Joins:

- Downstream models (`dim\_customer`, `fct\_subscriptions`) may fail if they still reference `legacy\_account\_id` in JOIN conditions.

- Example: Revenue reports could show NULL values for unmigrated accounts.

- SCD2 Merge Failures:

- `dim\_customer`'s slowly changing dimension logic might break if it relies on `legacy\_account\_id` as a natural key.

- Export Discrepancies:

- Salesforce or other external systems consuming `customer\_360\_snapshot` may reject records missing `legacy\_account\_id`.

2. Migration Coverage Risks

- Unmigrated Accounts (5.8%):

- If these accounts are active (not deprecated), their data becomes inaccessible post-drop.

- Worst-case: Customer service tools fail to look up accounts.

3. Performance Risks

- Query Degradation:

- Joins on `crm\_account\_uuid` (a longer string) may slow down compared to integer-based `legacy\_account\_id`.

4. Compliance Risks

- Audit Trail Gaps:

- If `legacy\_account\_id` was used in legal/financial records, its removal might violate retention policies.

Rollback Plan

1. Immediate Rollback (0-1 hour post-failure)

- Restore the Column:

```
```sql
```

```
ALTER TABLE raw.customers ADD COLUMN legacy_account_id STRING;
```

```
UPDATE raw.customers
```

```
SET legacy_account_id = backup.legacy_account_id
```

```
FROM temp_backup_table backup
```

```
WHERE raw.customers.crm_account_uuid = backup.crm_account_uuid;
```

```
```
```

- Rollback DAGs:

- Revert all downstream models to their pre-drop versions (e.g., redeploy prior git tags).

2. Short-Term Mitigation (1-24 hours)

- Fallback Lookup Table:

```
```sql
```

```
CREATE TABLE legacy_id_fallback AS
```

```
SELECT crm_account_uuid, legacy_account_id
```

```
FROM pre_drop_snapshot;
```

---

- Redirect queries to this table while diagnosing issues.

- Hotfix Critical Models:

- Patch `dim\_customer` to use `legacy\_account\_id` again, even if temporarily.

#### #### 3. Long-Term Remediation (1+ days)

- Root-Cause Analysis:

- Audit logs to find which queries failed due to missing `legacy\_account\_id`.

- Example: Look for `ERROR: column "legacy\_account\_id" does not exist` in query logs.

- Re-migration:

- Re-run the migration for missed accounts (e.g., the 1,204 unmigrated records).

---

#### ### Key Verification Before Rollback

1. Check Data Consistency:

- Compare row counts in `dim\_customer` pre- and post-drop.

2. Validate External Systems:

- Test Salesforce integrations with `customer\_360\_snapshot`.

3. Monitor Alerts:

- Look for spikes in NULL values or failed jobs in orchestration tools (e.g., Airflow).

---

#### ### Pro Tip: Gradual Rollout

- Canary Deployment:

- Drop the column for 1% of accounts first, monitor for issues, then proceed to 100%.

- Feature Flags:

- Use runtime flags

**Engineering counterpart**  
**2:41 AM - Responded in 0s**

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

## Agent response

2:42 AM - Responded in 24s

Here's a concise, actionable plan with owners, deadlines, and escalation paths:

### ### Pre-Migration Checklist (1-2 Weeks Before Drop)

| Task | Owner | Deadline | Success Metric | Escalation Path |

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

- | 1. Backfill remaining 5.8% unmigrated accounts | Data Ops (Alice) | D-14 | 99.99% coverage in `raw.customers` | CRM Team Lead |
- | 2. Validate no BI reports use `legacy\_account\_id` | Analytics (Bob) | D-10 | Zero references in Looker/Tableau logs | Head of Analytics |
- | 3. Test all downstream models in staging | DE Team (Carlos) | D-7 | All test pipelines pass | Data Eng Manager |
- | 4. Secure legal sign-off on column drop | Legal (Dana) | D-5 | Email confirmation | Chief Data Officer |

### ### Execution Day (T-0)

| Task | Owner | Time | Verification | Rollback Trigger |

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

- | 1. Final backup: `CREATE TABLE legacy\_id\_backup AS SELECT \* FROM raw.customers` | Data Ops (Alice) | 00:00 UTC | Backup row count matches prod | Backup failure |
- | 2. Deploy updated models to prod | DE Team (Carlos) | 01:00 UTC | Airflow dags succeed | Any pipeline failure |
- | 3. Drop column: `ALTER TABLE raw.customers DROP COLUMN legacy\_account\_id` | DB Admin (Eve) | 02:00 UTC (low-traffic) | `DESCRIBE raw.customers` confirms drop | Immediate rollback SQL ready |

### ### Post-Migration Monitoring (First 72H)

| Metric | Threshold | Owner | Alert Channel | Action if Breached |

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

- | Null values in `dim\_customer` | < 0.1% | DE On-Call | PagerDuty | Rollback models first |
- | Failed joins in `fct\_subscriptions` | 0 failures | DE On-Call | Slack #alerts | Enable fallback table |
- | Salesforce sync errors | < 5 incidents | CRM Team | Email | Revert snapshot model |

### ### Rollback Runbook (Owner: DE On-Call)

#### 1. Severity 1 (Full Outage)

- Immediate action:

```
```sql
```

```
-- Restore column (5 min)
```

```
ALTER TABLE raw.customers ADD COLUMN legacy_account_id STRING;
```

```
INSERT INTO raw.customers(crm_account_uuid, legacy_account_id)
```

```
SELECT crm_account_uuid, legacy_account_id FROM legacy_id_backup;
```

```
```
```

- Notify: Data Eng Manager + Incident Commander

#### 2. Severity 2 (Partial Issues)

- Hotfix path:

```
```sql
```

```
-- Deploy fallback view (2 min)
```

```
CREATE VIEW analytics.dim_customer_fallback AS
```

```
SELECT d., b.legacy_account_id
```

```
FROM analytics.dim_customer d
```

```
JOIN legacy_id_backup b ON d.crm_account_uuid = b.crm_account_uuid;
```

```
```
```

- Notify: Product Owners

### ### Timeline Summary

- D-14 to D-7: Prep work (backfill, testing)
- D-1: Final verification sync

- T-0: 2-hour maintenance window at 00:00-02:00 UTC
- T+72H: Declare success if monitoring thresholds hold

Decision Authority: Data Engineering Manager must approve each phase transition. Escalate to CDO if >1hr

**Engineering counterpart**  
**2:42 AM - Responded in 0s**

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

**Agent response**

**2:42 AM - Responded in 11s**

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

The unmigrated 5.8% of accounts (1,204 records) still using `legacy\_account\_id`.

**### Why This Matters Most**

**1. Hidden Dependencies:**

- A senior engineer should verify that these aren't secretly powering:
  - Critical internal tools (e.g., customer support dashboards)
  - Legal/financial reports (e.g., tax filings tied to legacy IDs)

**2. Data Validation:**

- Confirm the "unmigrated" label is accurate:
- Are these truly inactive/test accounts, or are some high-value customers stuck in limbo?
- Spot-check 10-20 random unmigrated accounts with business teams.

**3. Fallback Logic:**

- Review the proposed `legacy\_id\_fallback` table design:
  - Does it cover all edge cases (e.g., accounts created during migration)?
  - Is the lookup performance acceptable for emergency use?

**### Who Should Review**

- Data Architect or Principal Engineer: For technical validation.
- Business Analyst from CRM Team: To confirm no operational workflows rely on these IDs.

**### Red Flags to Watch For**

? If any unmigrated accounts are linked to:

- Active subscriptions in `fct\_subscriptions`
- Recent customer support tickets
- Regulatory documentation

Action if Red Flags Found:

- Halt the migration and initiate a targeted backfill for those accounts.

This is the make-or-break item-everything else can be rolled back, but missed legacy dependencies could cause silent data corruption.

**Engineering counterpart**  
**2:42 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.