EthanCorp

CHECKLIST AND TEMPLATE

Data Integration Health Check

Six failure modes and 28 checks that decide whether an integration estate can be trusted, changed and handed over.

How to use this

Use it on an integration estate that is already running, or on a design before the first pipeline is built. Any item you cannot answer with a document, a test or a named owner is where the risk is. The last page is a source data contract: fill in one per source system, and agree it with the team that owns that system. Integration work starts with these agreements rather than with connectors.

AUTHOR

Dat Tran, Enterprise Data & AI Analytics Architect

EDITION

October 2026

CONTACT

dattran@ethancorp.com

www.ethancorp.com

01 Record identity

Why it matters: when several systems hold the same customer, student or supplier under different keys, matching on name and email works for most records, and the rest becomes the project: false duplicates, merges that destroy history, two real entities turned into one.

- [ ] For each shared entity, the system of record that issues the identifier is named in writing.
- [ ] Where no system of record exists, an explicit mapping table holds the identity links, reviewed and under version control.
- [ ] Fuzzy matching only proposes matches; a person or a written rule confirms them.
- [ ] Merges are logged and reversible: the original records and their keys are kept.
- [ ] Duplicate and orphan records are counted on every load, not discovered by users.

02 Late corrections

Why it matters: most source systems allow backdated changes. An extract that only picks up new records misses them, and the target quietly disagrees with the source for that period.

- [ ] For every source, it is written down what a correction looks like and how far back it can reach.
- [ ] Extraction uses a modification watermark, change data capture, or a rolling re-extract window that covers the realistic correction period.
- [ ] The chosen method and its window are stated in the source data contract.
- [ ] Corrections to closed periods are flagged to the people who report on them.

03 Safe re-runs

Why it matters: a pipeline fails halfway, someone re-runs it, and part of the data now exists twice. Append-only loads are the easiest to build and fail only under conditions nobody tested.

- [ ] Every load is idempotent: an upsert on a defined key, or an atomic replace of a defined partition.
- [ ] Re-running a load is tested, including after a failure partway through.
- [ ] A failed run leaves the last good version in place.
- [ ] Each runbook says how to re-run its pipeline, as the first thing to try.

04 Reconciliation

Why it matters: without it, the first wrong number is found by a senior reader weeks later, and from then on every number in the platform is provisional.

- [ ] Row counts are compared between source and target after every load.
- [ ] Control totals for the measures that matter are compared after every load.
- [ ] Variance thresholds are agreed with the business in advance.
- [ ] On a mismatch, the publish is held and the previous good version stays live.
- [ ] The alert names the failing step and reaches a named person.
- [ ] Reconciliation results are kept, so any past load can be checked later.

05 Schema change

Why it matters: a source upgrade renames a field or adds a code value. The pipeline either fails in a confusing way or succeeds while dropping or mis-casting the new data.

- [ ] Each source is validated against an explicit contract: expected fields, types, nullability and handled code values.
- [ ] An unexpected structure or code value fails the load visibly instead of being dropped or cast silently.
- [ ] The source owner's duty to announce planned upgrades is written into the contract.
- [ ] Contract changes are versioned and reviewed like code.

06 Ownership and handover

Why it matters: the pipelines run and reconcile, then the person who built them changes role, and the estate becomes something the organisation is afraid to touch.

- [ ] Transformations are under version control, and every change is reviewed.
- [ ] Each pipeline has a runbook: what it does, what it depends on, and what to do when it fails.
- [ ] Each pipeline and each source has a named owner and a named backup.
- [ ] Failure alerts reach the owner before users notice.
- [ ] Handover includes a walkthrough with whoever owns it next, not only documents.

Integration & Migration Review

For when systems disagree, a migration is coming, or nobody is sure which job feeds which report. A fixed-scope review, three to four weeks, from US$1,500. It ends with:

- A profile of the source systems and the flows between them, as they actually run today
- Failure points ranked by business impact, each with the evidence
- A first source-to-target mapping for the domain that matters most
- A recommended first integration domain, with the constraints I expect to hit

Discuss this review →

Source data contract

What it is for: the agreement between the integration and the team that owns a source system. Fill in one per source. Review it whenever the source changes.

SOURCE SYSTEM		VERSION		STATUS	
BUSINESS OWNER		TECHNICAL OWNER		AGREED ON	

Agreements

Targets and decisions that depend on this source	
Unique key, and the system of record for each shared entity	
How corrections arrive, and how far back they reach	
Extraction method (watermark, change data capture or rolling window)	
Delivery: method, schedule and expected latency	
Reconciliation: row count, control totals, variance threshold, who is alerted	
How upgrades are announced, and how far ahead	

Authoritative fields

Field	Type	Nullable	Authoritative for	Allowed values / code list	On an unknown value	Notes

Sign-off

Role	Name	Signature	Date
Business owner			
Technical owner			
Integration owner			

EMAIL
dattran@ethancorp.com

WEBSITE
www.ethancorp.com

BOOK A 30-MINUTE CALL
calendly.com/ethancorp/30min

LATEST VERSION
www.ethancorp.com/resources.html · Edition October 2026