Modernizing the transformation layer

From stored procedures or nothing to production-grade dbt — without a big bang

A practical guide for data and analytics leaders: why legacy transformation fails, the migration path to dbt, and the dimensional modeling and CI/CD discipline that separate production from prototype. Featuring DBDeux, AICG's dbt delivery platform.

What you'll find inside

Transformation is where data projects are won or lost

Extraction, loading, storage, and compute are commodities. The middle — turning raw data into trusted, documented, business-ready models — is not. It is also where two decades of technical debt lives: stored procedures nobody can change, ETL jobs nobody trusts, and metrics that mean three different things.

dbt brought software engineering discipline to that layer — version control, testing, documentation, CI/CD — and became the de facto standard inside Snowflake, Databricks, BigQuery, and Redshift. But adopting dbt and succeeding with dbt are different things, and migrating legacy logic into it is where most programs stall.

This paper lays out AICG's approach: a migration path that rebuilds rather than translates, dimensional models grounded in Kimball methodology, and the CI/CD standards that make the result a production system. DBDeux — AICG's dbt delivery platform — compresses each step with prebuilt, proven components.

KEY TAKEAWAYS

Don't translate legacy code — rebuild it	Lift-and-shift recreates twenty years of debt in new syntax. The migration that works re-expresses business logic in tested, layered models.
Dimensional modeling matters more, not less	Conformed dimensions and explicit fact grains are what keep metrics consistent. dbt is the best vehicle Kimball methodology has ever had.
CI/CD is the real differentiator	Slim CI, deferred builds, model contracts, and freshness gating separate a demo from a system. The models are the easy part.
Accelerators compress the timeline	DBDeux's prebuilt source models, marts, and pipeline templates remove months of undifferentiated build time from every engagement.

Your most important logic lives where nobody can see it

Every number your leadership acts on — revenue, labor cost, inventory, margin — is produced by transformation logic. In most organizations that logic accumulated over decades in stored procedures, ETL tool jobs, and undocumented SQL scripts written by people who have since moved on.

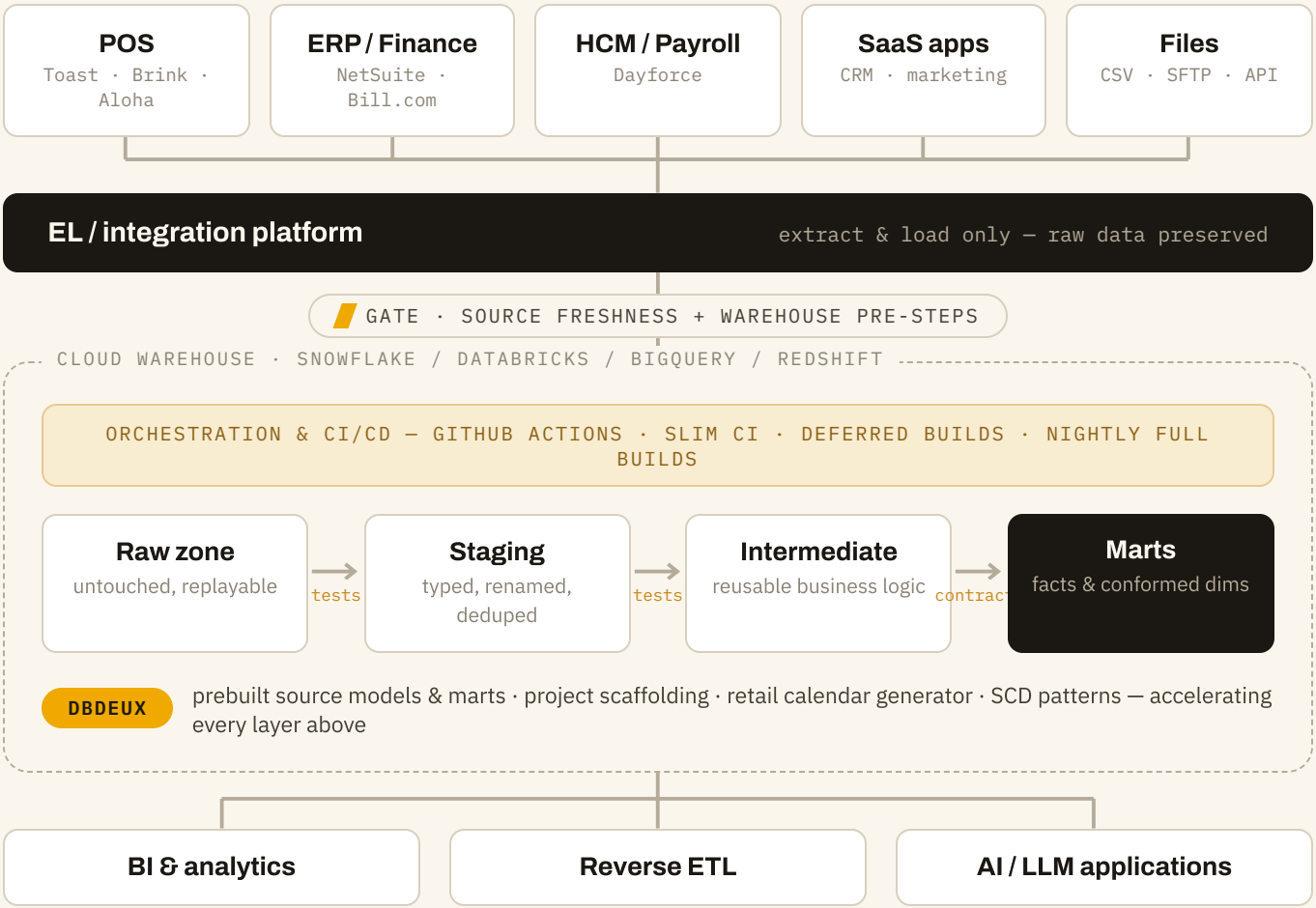
The result is a layer with no version history, no tests, no documentation, and no review process — maintained by the one engineer who remembers how it works. The symptoms are consistent across industries.

Conflicting versions of the truth	Multiple definitions of "revenue" living in different procedures and BI tools, each rebuilt slightly differently by every analyst who needed it. Nobody can say which dashboard is right, so nobody fully trusts any of them.
Silent breakage	A source system changes a column and the numbers quietly go wrong. With no tests and no contracts, the error is discovered by the CFO, not the pipeline — often weeks later.
Key-person risk	Change requests queue behind a single overloaded engineer, because touching anything risks breaking everything. The backlog grows; the business routes around the data team.
A ceiling on AI ambitions	AI and LLM applications need well-modeled, documented, tested data underneath them. An opaque transformation layer caps every initiative built on top of it.

> **“A stored procedure is a business rule with no version history, no tests, and one retirement party between it and unmaintainable.”**

dbt in the modern pipeline

ELT, not ETL: load raw data first, preserve it untouched, and transform inside the warehouse — so every model is reproducible and auditable from source.



AI-ready by design. The same semantic clarity that serves a dashboard serves an AI agent querying your warehouse — well-modeled, tested, documented marts are the substrate both need.

Rebuild, don't translate

The tempting migration is the literal one: convert each stored procedure to dbt syntax and call it done. That recreates the debt in a new dialect. AICG's approach re-expresses the business logic in layered, tested models — retiring the legacy estate mart by mart, with no big-bang cutover.

PHASE 1

Inventory & triage

Catalog every procedure, job, and script; map lineage to actual consumers

Score each asset: retire, rebuild, or defer — dead code is common and goes first

Sequence marts by business value, not technical convenience

PHASE 2

Foundation & staging

DBDeux scaffolding: layered structure, naming standards, CI pipelines preconfigured

One staging model per source table — typed, renamed, deduplicated, key-tested

AI-assisted tooling drafts staging models, tests, and docs from source DDL for engineer review

PHASE 3

Dimensional rebuild

Business logic re-expressed as intermediate models and dimensional marts — not transliterated

Conformed dimensions built once, shared everywhere; facts at explicit grains

Model contracts on everything consumed downstream

PHASE 4

Parallel run & cutover

Legacy and dbt outputs reconciled side by side until the numbers agree — auditably

BI repointed mart by mart; procedures decommissioned as each is replaced

Enablement: docs site, runbooks, and training so your team owns it

The output is not a translation — it is a tested, documented, version-controlled rebuild your team can maintain, with the legacy estate retired incrementally and every cutover reconciled against the old numbers.

Kimball was right all along

Dimensional modeling did not become obsolete with the cloud — it became more important. The "modern data stack" era of shipping one big table per dashboard produced exactly the metric drift and duplicated logic that facts and conformed dimensions were designed to prevent.

AICG's practice is grounded in Kimball Group methodology, delivered through dbt. The semantic layer can't fix what the model layer got wrong — these are the structures that make numbers agree everywhere they appear.

Conformed dimensions

Date, location, customer, employee, product — built once, shared across every mart. When "location" means the same thing in the sales mart and the labor mart, cross-domain analysis stops being a research project.

Explicit fact grains

Every fact table declares exactly what one row means — one order line, one shift, one payment — and is documented and tested at that grain. Ambiguous grain is the root cause of most double-counting incidents.

History that survives change

Surrogate keys and SCD Type 2 tracking where history matters — so a store that changed regions in March reports correctly in both the before and the after. DBDeux ships standardized SCD and deduplication patterns for change-tracked sources.

Calendars the business actually uses

Integer date keys joined to a proper calendar dimension — including 4-4-5, 4-5-4, and 5-4-4 retail calendars for industries that live by them. DBDeux's retail calendar generator produces warehouse-ready SQL for all of them.

> **“Dimensional modeling predates dbt by decades. dbt is simply the best delivery vehicle it has ever had.”**

The difference between a project and a system

Most dbt content stops at "write tests." These are the standards AICG configures as defaults on every engagement — preconfigured in DBDeux's pipeline templates.

	THE PROTOTYPE DEFAULT	THE AICG PRODUCTION STANDARD
CI scope	Every PR rebuilds the entire project. CI takes 90 minutes and burns warehouse credits.	**Slim CI with state comparison.** Build and test only changed models and their downstream dependents (`state:modified+`). Feedback in minutes; cost doesn't scale with repo size.
Upstream models	Rebuilt from scratch in every CI run, at full compute cost.	**Deferred builds.** CI borrows unchanged upstream models from production (`--defer`) instead of rebuilding them.
State baseline	No baseline exists, so "what changed" can't be answered reliably.	**Manifest preservation.** Production artifacts persisted so every CI run compares against an accurate baseline — with bootstrap fallbacks when none exists yet.
Scheduling	Cron fires on time whether or not new data arrived — stale numbers propagate silently.	**Source freshness gating.** Runs confirm sources actually received new data first; warehouse pre-steps (deduplication, ingestion finalization) complete synchronously as failure-gating stages.
Schema changes	A changed column type surfaces as a broken dashboard, days later.	**Model contracts on marts.** Enforced schemas make breaking changes fail CI — not surprise a BI tool or AI application.
Drift control	Incremental models drift quietly; nobody knows if the project still builds from scratch.	**Nightly full builds + tag-based selection.** A full-refresh build catches drift; models tagged by domain, criticality, and schedule run precisely the right subset at the right cadence.

Start from proven assets, not a blank repository

DBDeux is AICG's dbt delivery platform: the accumulated accelerators, prebuilt components, and codified standards from our implementation practice. dbt is the engine; DBDeux is the delivery system around it.

01
Project scaffolding
Layered structure, naming standards, and the full CI/CD pipeline from §06 — preconfigured on day one.

02
Prebuilt source models & marts
POS (Toast, Brink, Aloha), payroll/HCM (Dayforce), ERP and financials (NetSuite, Bill.com) — ready to configure, not build.

03
Retail calendar generator
4-4-5, 4-5-4, 5-4-4, and Gregorian calendars with warehouse-ready SQL output.

04
SCD & deduplication patterns
Standardized handling for change-tracked and soft-delete sources — solved once, applied everywhere.

05 **AI-assisted generation**

Tooling that converts source DDL into draft staging models, tests, and documentation — for engineer review, not blind acceptance. This is where the Phase 2 timeline compression comes from.

CLIENT SPOTLIGHT [[placeholder — replace logo, quote & result before publishing]]

Client logo
or browse files

“Client quote goes here — one or two sentences on what changed: trust in the numbers, speed of delivery, or what the team can now do on its own.”

Name, Title — Company

THE RESULT One-line engagement summary with a metric — e.g. "X sources to production dimensional marts in Y weeks, with Z legacy procedures decommissioned."

You could migrate this yourself

The question is whether you should spend two quarters learning lessons a partner has already paid for. AICG brings three things generic dbt shops don't:

Kimball Group grounding	Dimensional methodology is our practice's foundation, not a checkbox. Conformed dimensions, explicit grains, and retail calendars are defaults — the disciplines that keep metrics consistent for the next decade.
DBDeux accelerators	Prebuilt source models, marts, pipeline templates, and generation tooling mean engagements start from proven assets. Compressed time-to-first-mart, and consistency across everything delivered.
Enablement built in	Documentation sites, runbooks, and training are a project phase, not a handoff email. An implementation your team cannot maintain is a liability with good lineage — we don't ship those.

THE FIRST CONVERSATION

Start with a health check of your existing transformation layer

Two weeks of assessment is cheaper than two quarters of rework. AICG's dbt implementation services — powered by DBDeux — are available directly and through AWS Marketplace via private offer.

sales@aicg.com aicg.com