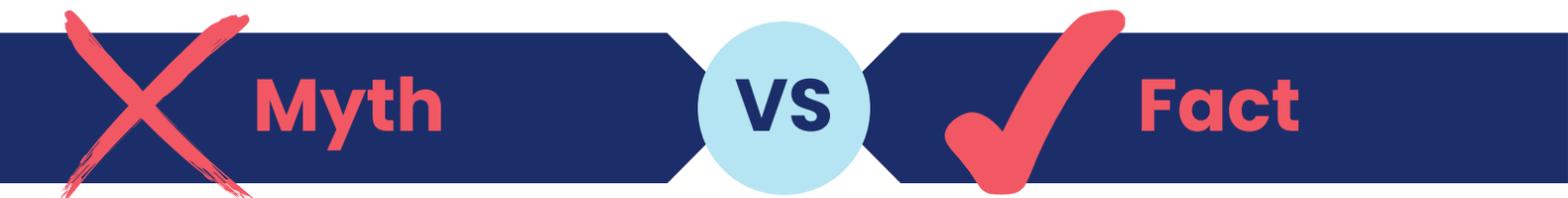


Myth vs. Reality of Technical Data Lineage

Lessons from mature data teams in large organisations



Myth

VS

Fact

Lineage means mapping every data field and system.

This creates a dense web that's impossible to maintain or use.



Lineage should focus on what matters – typically 100–200 business-critical attributes or entities that underpin regulatory reporting, risk, or customer outcomes.

Lineage exists only to keep regulators happy.

Seen as a compliance tick-box, it often gets deprioritised or ignored by business teams.



Lineage enables trusted reporting, root-cause analysis, and customer journey insights – making it a strategic business enabler as well as a compliance requirement.

Legacy systems must all be mapped in full.

This drains time and resources on platforms already being retired.



“Fix forward” by embedding lineage in new cloud and lakehouse platforms – and only backfill legacy lineage when regulations demand it.

Only detailed, field-level lineage satisfies regulators.

Chasing this level of detail everywhere is prohibitively expensive and unsustainable.



High-level lineage usually meets most needs; reserve deep, field-level lineage for high-risk or high-value cases where regulators or operations require it.

Lineage is the governance team's responsibility.

Treating lineage as a governance bolt-on creates bottlenecks and low adoption.



Lineage is an engineering artefact – captured automatically in delivery pipelines like test plans, with governance acting as a partner and consumer.

Lineage outputs are only useful for data specialists.

Dense technical maps often alienate business stakeholders, auditors, and regulators.



Lineage outputs must be usable across the organisation – simple, interpretable views that business, compliance, and technical audiences can all trust.

Top Advice

- **Prioritise scope** → Target critical attributes and entities agreed with the business
- **Anchor in data products** → Reusable entities make lineage scalable
- **Embed in engineering** → Capture lineage automatically during build, not afterwards
- **Leverage transformation** → Cloud and lakehouse platforms reduce complexity
- **Apply risk-based detail** → Use high-level lineage by default; deep dives only when stakes are high
- **Communicate value** → Frame lineage as enabling business journeys, not policing
- **Experiment cautiously with AI** → Test LLMs for metadata, but validate outputs carefully