Northgrain Data

Data Platform Audit Checklist

dbt · Airflow · Snowflake

Use this checklist to review platform reliability, access, cost controls, dbt project safety and operational ownership. Mark each item during the audit and convert open findings into a prioritized remediation backlog.

ORGANIZATION _____ REVIEWER _____ DATE _____

1. Orchestration and reliability

- [ ] Review 30 days of DAG runs. Record repeated failures, missed schedules, long retries and unusually slow runs.
- [ ] Route failures to a named engineer. Alerts should include the DAG, task, run time and a direct link to logs.
- [ ] Assign an owner to every critical DAG. Define who responds during business hours and who handles urgent incidents.
- [ ] Document retry, replay and backfill behavior. Manual reruns should be intentional, traceable and safe to repeat.
- [ ] Run source freshness checks early enough to flag stale inputs before downstream reports are published.

2. Secrets and access

- [ ] Scan repositories, CI settings and deployment files for committed credentials, tokens and webhook URLs.
- [ ] Store runtime credentials in Airflow Connections, environment variables or an external secrets backend. Avoid plain-text Variables.
- [ ] Verify Fernet encryption when Airflow stores sensitive values in its metadata database.
- [ ] Use key-pair or OAuth authentication for Snowflake service accounts where supported. Avoid shared passwords.
- [ ] Review access after role changes and departures. Disable unused accounts and rotate shared credentials.

3. Warehouse cost and workload management

- [ ] Map dbt, ingestion, BI and ad hoc workloads to warehouses. Split them when queueing, cost attribution or access control requires it.
- [ ] Set resource monitors and credit alerts for warehouses with material spend. Document who responds to each threshold.
- [ ] Tune auto-suspend to workload cadence. Sixty seconds is a practical starting point for intermittent batch jobs.
- [ ] Review queued_overload_time, queued_provisioning_time and spill metrics for delayed or expensive runs.
- [ ] Schedule full refreshes and backfills explicitly. Estimate runtime, credit use and downstream impact before execution.

4. dbt project controls

- [ ] Apply documented naming and layering conventions across staging, intermediate and mart models.
- [ ] Declare raw tables as sources and reference them with source() from staging models.
- [ ] Configure freshness on business-critical sources using loaded_at_field or adapter-supported metadata.
- [ ] Use contracts on stable marts where breaking schema changes must fail during the build.
- [ ] Set an explicit on_schema_change policy for each incremental model and document the selected behavior.

- [ ] Configure unique_key when a model updates existing records. Document models designed as append-only.
- [ ] Test the declared grain with not_null, uniqueness or combination_of_columns checks.
- [ ] Run Slim CI with production artifacts, isolated CI schemas and merge blocking on failure.
- [ ] Keep development, CI and production schemas separate. Confirm that personal schemas cannot overwrite production objects.
- [ ] Remove stale models, unused configs, dead SQL and outdated comments during review.

5. Documentation and ownership

- [ ] Document business purpose, grain, owner, SLA and downstream consumers for every critical mart.
- [ ] Define exposures for high-value dashboards, exports and data products.
- [ ] Record major architectural decisions, including the selected option, alternatives and review date.
- [ ] Write deployment, rollback and incident-response steps that a new engineer can follow.
- [ ] Maintain a remediation backlog with impact, owner, estimate and target date.