Data Platform Architecture Decision Guide

Decision trees & scoring rubrics for modern data stack choices

HOW TO USE THIS GUIDE

Use this guide when evaluating, redesigning, or scaling your data platform. Each section presents a decision tree or scoring rubric. Work through the questions in order -- your answers direct you to a recommended architecture pattern or tool category.

DECISION 1: LAKEHOUSE VS DATA WAREHOUSE

Criteria	Score Lakehouse (+1)	Score Warehouse (+1)
Primary consumers	Data scientists, ML engineers	Analysts, BI tools, SQL-first
Data types	Unstructured + structured	Structured / semi-structured only
Update frequency	High (streaming or hourly)	Low (nightly batch sufficient)
ML use cases	Central to roadmap	Minimal / exploratory only
Vendor preference	Open formats (Parquet/Iceberg)	Managed SaaS preferred
Team skills	Spark, Python, cloud-native	SQL, dbt, traditional BI
Cost sensitivity	Compute/storage separation needed	All-inclusive pricing acceptable

Decision rule: If Lakehouse score >= 4, proceed with Lakehouse (Databricks, Apache Iceberg on S3/GCS, or Azure Fabric). If Warehouse score >= 5, evaluate Snowflake, BigQuery, or Redshift. Mixed score -> hybrid architecture with a lakehouse foundation and a semantic layer.

DECISION 2: TRANSFORMATION LAYER

dbt Core / Cloud ***
Best for SQL-fluent teams; strong ecosystem; native docs & lineage.

Spark / PySpark ****o
Best for large-scale Python transformations; needed for ML feature prep.

Dataform (GCP) ***oo
Google-native; good for BigQuery-first orgs.

Matillion **ooo
ELT GUI tool; good for non-engineers; lower flexibility.

Data Platform Architecture Decision Guide

Orchestration, Compute & Ingestion Rubrics

DECISION 3: ORCHESTRATION PLATFORM

Apache Airflow (MWAA / Cloud Composer / Astronomer)

Industry standard. Best when: large team, complex DAGs, existing Airflow investment. Cons: operational overhead.

Prefect 2 / Prefect Cloud

Modern Python-native; great DX; easy local dev. Best when: Python-first team, want fast iteration.

Dagster

Asset-centric; best for data asset cataloguing + lineage-native pipelines. Steep learning curve.

dbt as Orchestrator (dbt Cloud Jobs)

Sufficient for pure transformation workflows. Avoid for cross-system orchestration.

AWS Step Functions / GCP Workflows

Good for cloud-native serverless pipelines; limited debugging UI.

DECISION 4: COMPUTE ENGINE

- **Databricks (Spark):** Large-scale ML + transformation; unified platform.
- **dbt + Serverless SQL:** Pure ELT; cost-effective for SQL workloads.
- **AWS Glue:** Serverless Spark; low ops overhead; AWS-locked.
- **Ray:** Distributed Python for ML inference; growing ecosystem.
- **Snowpark:** Python inside Snowflake; avoids data movement.

DECISION 5: INGESTION & CDC

Tool / Pattern	Best For	Complexity
Fivetran / Airbyte	Pre-built connector libraries; fast time to value; best for SaaS sou	High
Kafka + Debezium	Real-time CDC; high throughput; operationally complex.	High
Striim / Qlik Replicate	Enterprise CDC with low latency; good for mainframe/ERP.	Medium
Custom Python / Singer	Full control; required for non-standard sources.	Low

Data Platform Architecture Decision Guide

Serving Layer, Metrics & Evaluation Scorecard

DECISION 6: SEMANTIC / METRICS LAYER

- **dbt Semantic Layer (MetricFlow):** Recommended for dbt-centric stacks. Define metrics once, query anywhere.
- **Cube.dev:** Headless BI API; good for embedding analytics; language-agnostic.
- **Looker (LookML):** Enterprise-grade; strong governance; Google Cloud native.
- **Power BI Datasets / Tabular:** Best when Microsoft ecosystem is dominant.

PLATFORM EVALUATION SCORECARD (0-5 PER CRITERION)

Criterion	What to Evaluate	Score /5
Scalability	Handles 10× current data volume without re-architecture.	
Developer Experience	Time-to-first-pipeline for a new engineer < 1 day.	
Operational Overhead	Managed service vs. self-hosted; pager burden.	
Cost Efficiency	Cost per TB processed and cost per query at scale.	
Ecosystem & Integrations	Connector coverage, API richness, partner ecosystem.	
Governance & Lineage	Native data lineage, access controls, audit logging.	
Vendor Lock-in Risk	Open formats, portability, licence terms.	
Support & Community	Documentation quality, community size, SLA options.	

Decision rule: Choose the platform option with the highest aggregate score. For tied scores (within 3 points), weight Scalability and Developer Experience 2×. A score < 20/40 total across all candidates is a signal to evaluate emerging options or pursue a custom build for the weakest dimension.