2516
TECHNOLOGIES

AI Readiness and
Data Transformations

Migrating from
Qlik Sense to Astrato

A practitioner's 1:1 reference for BI engineers, analytics leads, and data architects moving to a warehouse-native model.

6	Core sections	20+	Expression mappings	90	Day migration	1	Audit checklist

Mark Henning · Founder & CDO, 2516 Technologies

HOW TO USE THIS GUIDE

Each section covers one Qlik concept, explains the mental model shift required, and provides a concrete before/after example. All SQL and script examples use abstract placeholder table names (fact_table, dim_table, bridge_table) so patterns are portable to any domain and data warehouse.

Assumptions: your warehouse is Redshift, Snowflake, or BigQuery (SQL-standard); you have an Astrato workspace with a live warehouse connection configured; you are familiar with Qlik Sense but new to Astrato's semantic layer model.

SECTION 01

System Requirements & Infrastructure

A successful Qlik → Astrato migration is as much an infrastructure project as a BI project. Before any data moves or any dashboard gets rebuilt, both environments need to be assessed and prepared. This section documents what you need from your existing Qlik Sense environment, what Astrato and your warehouse require on the other side, and what needs to be true about the connection between them.

Qlik Sense Enterprise on Windows (QSEoW) and Qlik Sense SaaS have different access models, export capabilities, and QVD storage structures. Your approach to exporting QVDs (covered in Section 02) depends on which deployment you are running. Both are covered here. Astrato itself is a SaaS platform with no server-side installation — the infrastructure weight of this migration falls on the warehouse side.

Qlik Sense Infrastructure Inventory

Before migration begins, document the following from your Qlik environment. This inventory drives every downstream decision — QVD export strategy, warehouse schema design, and semantic layer architecture.

Component	QSEoW (Enterprise)	Qlik Sense SaaS
QVD storage	File system path on Qlik server — mapped network share or local disk	Data Files in Management Console — download or use Data Gateway
Script access	Full load script via QMC or Dev Hub	Load script in each app editor
Reload scheduling	Qlik Scheduler Service — document task names and frequencies	Cloud Automation or manual reload — document trigger type
Authentication	Windows Auth, SAML, or Qlik ticket-based auth	IdP via SAML/OIDC — note provider (Okta, Azure AD, etc.)
User management	QMC: Streams, Security Rules, custom properties	Management Console: Spaces, Roles, Groups
App metadata	Export full app list from QMC with stream assignments	Export app list from MC with space assignments

Qlik Infrastructure → Astrato Equivalent

Qlik Component	Astrato Equivalent
QIX in-memory engine	Live warehouse query (Snowflake / Redshift / BigQuery)
QVD files on file system	Warehouse tables and views
Load script + Master Items	Astrato semantic layer — dataset definitions
Qlik Scheduler Service	Warehouse pipeline — dbt, Fivetran, or native scheduler

Streams / Spaces	Astrato Spaces with role-based access control
Qlik Hub (browser)	Browser-based — Chrome 90+ or Edge 90+ recommended
Set analysis expressions	SQL window functions and pre-aggregated views
Master Items library	Semantic layer calculated fields and measures

Warehouse Requirements

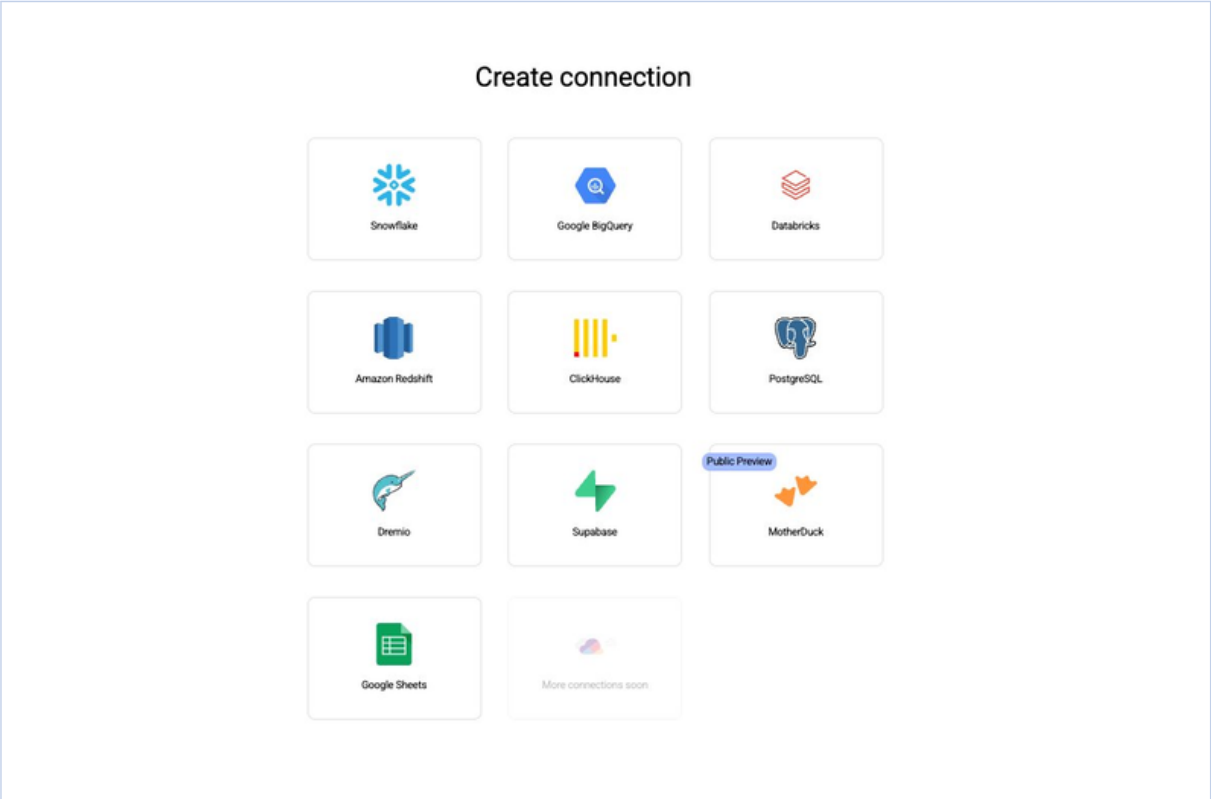
Astrato connects natively to your warehouse via a direct connection — no on-premise agent or middleware required. Snowflake is the primary recommended warehouse: its elastic compute model, automatic suspend/resume, and column-store architecture align well with Astrato's live query pattern.

Warehouse	Minimum Configuration	Notes
Snowflake	X-Small warehouse (1 credit/hr)	Auto-suspend recommended. Scale to Small/Medium for large datasets or concurrent users.
Amazon Redshift	RA3.xlplus or Serverless	Serverless for variable workloads. Port 5439 must be open to Astrato IPs.
Google BigQuery	On-demand (default)	No infrastructure to provision. Service account needs dataViewer + jobUser roles.
PostgreSQL / Supabase / ClickHouse / Others	Per Astrato connector docs	See Astrato connection documentation for version requirements.

Sizing rule of thumb: total CSV export volume from QVDs x 1.5 = estimated warehouse storage needed post-load. Compressed warehouse storage is typically 3-5x smaller than raw CSV.

Supported Warehouse Connections

Astrato supports direct connections to all major cloud warehouses and several emerging platforms. The following connectors are available in the Astrato workspace setup. Snowflake, Redshift, and BigQuery are fully production-ready; MotherDuck is currently in public preview.



Network & Connectivity

Astrato connects to your warehouse via a direct connection from its cloud infrastructure. No on-premise agent is required. The warehouse must be reachable from Astrato's published IP ranges — consult Astrato's network documentation for the current list.

→ **Snowflake:** Network policy must whitelist Astrato IPs. Private Link supported on Business Critical tier. Snowflake uses HTTPS (443) — no additional port changes required.

→ **Redshift:** VPC security group must allow inbound TCP on port 5439 from Astrato IP ranges. Publicly accessible endpoint required unless VPC peering is configured.

→ **BigQuery:** No IP allowlisting required. Access is controlled via service account IAM roles — bigquery.dataViewer and bigquery.jobUser at minimum.

→ **All warehouses:** SSL/TLS encryption required in transit. Astrato enforces encrypted connections by default.

Access & Permissions Checklist

Verify the following before migration work begins. Items marked [MANDATORY] will prevent the migration from starting if not resolved.

Qlik Sense Enterprise (QSEoW)

[ ] **[MANDATORY]** Admin access to Qlik Management Console (QMC)

[ ] **[MANDATORY]** QVD storage file path(s) documented and accessible

[ ] **[MANDATORY]** Ability to run reload tasks or execute load scripts

[ ] Full app list exported with stream assignments

[ ] Load scripts reviewed and transformation logic documented

[ ] Reload schedule documented — task names and frequencies

Qlik Sense SaaS

[ ] **[MANDATORY]** Admin access to Qlik Cloud Management Console

[ ] **[MANDATORY]** Data Files location documented or Data Gateway configured

[ ] App list exported with space assignments

[ ] IdP / SSO configuration documented — provider, entity ID

[ ] Reload automation type documented — Cloud Automation or manual

Snowflake (Primary Warehouse)

[ ] **[MANDATORY]** Account identifier and region documented

[ ] **[MANDATORY]** Service account created with USAGE on target database and schema

[ ] **[MANDATORY]** Service account granted SELECT on all source tables and views

[ ] **[MANDATORY]** Service account granted CREATE TABLE/VIEW in staging schema

[ ] **[MANDATORY]** Compute warehouse created with auto-suspend configured

[ ] **[MANDATORY]** Network policy updated to allow Astrato IPs

[ ] **[MANDATORY]** Connection string tested successfully from Astrato workspace

[ ] Row-level security policies reviewed — RLS may affect query results

Astrato Workspace

[ ] **[MANDATORY]** Workspace created and admin access confirmed

[ ] **[MANDATORY]** Warehouse connection configured, tested, and saved

[ ] **[MANDATORY]** User roles and Spaces defined before workbook build begins

[ ] SSO / SAML configured if required by organization policy

[ ] Astrato IP ranges documented and shared with network team

Browser / Client

[ ] **[MANDATORY]** Chrome 90+ or Edge 90+ available on all developer machines

[ ] Safari noted as limited support — not recommended for dev work

[ ] Popup blocker exceptions added for Astrato domain if required

SECTION 02

Exporting QVDs to CSV

Why This Step Matters

QVDs are Qlik's proprietary binary format. Before any data can be loaded into your warehouse and queried by Astrato, QVD contents must be extracted into a portable format. CSV is the most universally accepted intermediate format for warehouse ingestion — compatible with Redshift COPY, Snowflake COPY INTO, BigQuery bq load, and every major ETL tool.

You have two main options: the native Qlik STORE command (no additional tooling required) or Talend (recommended when you need transformation, scheduling, or enterprise-grade error handling during the export).

Option A: Qlik STORE Command

The STORE command is built into Qlik's load script. It writes any loaded table or QVD-sourced dataset directly to a CSV or QVD file. No additional software required — run it from any Qlik Sense app that has access to the source QVD.

Basic STORE Export

```
// Load the source QVD into memory
source_table:
LOAD *
FROM [lib://QVD_Library/fact_table.qvd] (qvd);

// Export to CSV
STORE source_table INTO [lib://CSV_Export/fact_table.csv] (txt, delimiter is ',', embedded labels);

// Drop the in-memory table when done (optional, saves RAM)
DROP TABLE source_table;
```

For large QVDs or when you need to export multiple tables in one pass, loop the STORE command across a list of table names:

Batch Export Loop

```
// Export multiple QVDs in a single script pass
SET vTableList = 'fact_table|dim_account|dim_product|dim_territory';
SET vIdx = 1;

DO WHILE vIdx <= NoOfFields('vTableList')
LET vTable = Pick(vIdx, $(vTableList));

$(vTable):
LOAD * FROM [lib://QVD_Library/$(vTable).qvd] (qvd);

STORE $(vTable) INTO [lib://CSV_Export/$(vTable).csv]
(txt, delimiter is ',', embedded labels);

DROP TABLE $(vTable);
LET vIdx = vIdx + 1;
```

```
LOOP
```

STORE Command — Key Options

Option	Usage
`delimiter is ','`	Standard CSV comma delimiter (default)
`delimiter is '\t'`	Tab-delimited — safer if fields contain commas
`embedded labels`	Include header row with field names (recommended)
`no labels`	Omit header row — useful if warehouse schema is pre-defined
`(qvd)`	Re-export as QVD instead of CSV — useful for archiving
`utf8`	Force UTF-8 encoding: STORE ... (txt, utf8, embedded labels)

Option B: Talend

Talend Open Studio (free) and Talend Data Integration provide a GUI-based pipeline for reading QVDs and writing CSVs or loading directly into the warehouse. Talend is the better choice when you need scheduled exports, row-level transformation during export, error logging, or enterprise audit trails.

Talend does not have a native QVD connector. The standard approach is a two-component pipeline: a tFileInputDelimited or tJava component to read the QVD binary, paired with a tFileOutputDelimited to write the CSV.

Talend tJava — QVD to CSV

```
// Talend tJava component — read QVD using QvxReader library
// Requires: qvx-reader.jar on the Talend classpath
import com.qliktech.qvx.QvxReader;
import java.io.FileWriter;
import com.opencsv.CSVWriter;
QvxReader reader = new QvxReader("path/to/fact_table.qvd");
CSVWriter writer = new CSVWriter(new FileWriter("output/fact_table.csv"));
// Write header
writer.writeNext(reader.getFieldNames().toArray(new String[0]));
// Stream rows
while (reader.next()) {
writer.writeNext(reader.getCurrentRowAsStringArray());
}
reader.close(); writer.close();
```

STORE vs Talend — Decision Guide

Scenario	Recommended Approach
Small number of QVDs, one-time export	STORE command — fastest path
Large QVD library (50+ tables)	Talend batch job with loop
Need transformation during export	Talend — apply tMap transforms
Need scheduled / recurring export	Talend with job scheduler
No Qlik Sense server access	Talend with QvxReader library
Enterprise audit trail required	Talend with tLogRow + file log
Loading directly to warehouse (skip CSV)	Talend tRedshiftOutput / tSnowflakeOutput

Loading CSVs into the Warehouse

Once CSVs are generated, load them into your warehouse before connecting Astrato. The load command varies by warehouse:

```
-- Redshift: COPY from S3
COPY analytics.fact_table
FROM 's3://your-bucket/exports/fact_table.csv'
IAM_ROLE 'arn:aws:iam::account:role/RedshiftCopyRole'
CSV IGNOREHEADER 1 DATEFORMAT 'auto' TIMEFORMAT 'auto';
-- Snowflake: COPY INTO from stage
COPY INTO analytics.fact_table
FROM @your_stage/fact_table.csv
FILE_FORMAT = (TYPE = CSV SKIP_HEADER = 1 FIELD_OPTIONALLY_ENCLOSED_BY = '"');
-- BigQuery: bq load CLI
-- bq load --autodetect --source_format=CSV analytics.fact_table gs://bucket/fact_table.csv
```

SECTION 03

Load Script / QVD Layer → Semantic Layer

The Concept Shift

In Qlik, the load script is everything. It extracts data, transforms it, joins it, derives fields, and loads it into the in-memory associative engine. QVDs act as a caching layer. The result is a self-contained model that lives inside the app — powerful, but isolated from the rest of your data infrastructure.

In Astrato, the warehouse is the engine. There is no load script. You build a semantic layer in Astrato that sits on top of your existing warehouse tables — defining joins, calculated fields, and business logic there. Dashboards query the warehouse live.

Migration decision rule:

→ Joins, filters, aggregations, field derivations · warehouse view or model

→ Business metric definitions and reusable calculations · Astrato semantic layer

→ Incremental loading / scheduling · warehouse pipeline (dbt, Fivetran, etc.)

→ QVDs · replaced by warehouse result caching / materialized views

Before: Qlik Load Script

```
// Qlik: load script join + derived field
fact_data:
LOAD transaction_id, account_id, amount,
IF(amount > 1000, 'High', 'Standard') AS value_tier
FROM [lib://QVD_Library/fact_table.qvd] (qvd);

dim_data:
LOAD account_id, account_name, region, segment

FROM [lib://QVD_Library/dim_table.qvd] (qvd);
```

After: Warehouse View + Astrato Semantic Layer

```
-- Warehouse view: handles join + derived field
CREATE OR REPLACE VIEW analytics.v_transactions AS
SELECT
f.transaction_id, f.account_id, f.amount,
CASE WHEN f.amount > 1000 THEN 'High' ELSE 'Standard' END AS value_tier,
d.account_name, d.region, d.segment
FROM fact_table f
LEFT JOIN dim_table d ON f.account_id = d.account_id;

-- Connect analytics.v_transactions in Astrato as a single dataset.

-- No join logic needed in Astrato -- the view handles it.
```

SECTION 04

Set Analysis → Astrato Equivalents

The Concept Shift

Set analysis is one of Qlik's most powerful — and most Qlik-specific — features. It allows expressions to evaluate over a different set of data than the current selection context. Common use cases: prior period comparisons, fixed-denominator share calculations, ignoring one filter while respecting others.

Astrato does not have set analysis syntax. The equivalent logic is handled in SQL using window functions, CTEs, or pre-aggregated views. The key shift: in Qlik you modify the selection context inside the expression. In Astrato, you define the comparison logic once in SQL and expose it as a clean measure.

Before: Qlik Set Analysis

```
// Fixed denominator (ignore all selections)
Sum({1} amount)
// Prior year only
Sum({} amount)
// Fixed segment, ignoring segment filter
Sum({} amount)
```

After: SQL Equivalents

```
-- Fixed denominator: window function
SUM(amount) OVER () AS total_amount_all
-- Prior period: self-join pattern
SELECT cy.transaction_year,
cy.amount AS current_amount,
py.amount AS prior_year_amount
FROM (SELECT transaction_year, SUM(amount) AS amount FROM fact_table GROUP BY 1) cy
LEFT JOIN (SELECT transaction_year, SUM(amount) AS amount FROM fact_table GROUP BY 1) py
ON cy.transaction_year = py.transaction_year + 1
-- Fixed segment: pre-filtered view
CREATE OR REPLACE VIEW analytics.v_enterprise AS SELECT * FROM fact_table WHERE segment =
'Enterprise';
```

SECTION 05

Variables & Parameters → Switch Action

The Concept Shift

Qlik variables are global, mutable values set by user interaction and referenced inside expressions. Commonly used for dynamic measure switching, parameter inputs, and conditional display logic.

Astrato has its own variable system, but variable changes can trigger each other, creating cascading loops if not managed carefully. The Switch Action orchestrates multi-variable changes in a controlled, sequential way — preventing race conditions where variables fire in the wrong order or loop back on themselves.

Before: Qlik Variable-Driven Measure Switch

```
// Variable: vMeasureSelector (set via button to 1, 2, or 3)
IF(vMeasureSelector = 1, Sum(amount),
IF(vMeasureSelector = 2, Sum(units),
Sum(margin)))
```

After: Astrato Switch Action

```
// Switch Action — guaranteed sequential execution
Trigger: Button click / dropdown change
Actions (in order):
1. Set variable: measure_selector = [selected value]
2. Set variable: period_filter = 'All' // reset dependent
3. Set variable: comparison_toggle = 'Off' // reset dependent
// Prevents cascading variable loop -- variables fire sequentially, not concurrently.
```

Variable Pattern Translation

Qlik Pattern	Astrato Equivalent
`Input box → variable`	Parameter control → variable
`Button → Set Variable action`	Button → Switch Action (single or multi-step)
`$(vMyVar) in expression`	Variable reference in calculated measure
`Conditional show/hide`	Sheet/object conditional visibility via variable
`Variable default on app open`	Variable default value in definition

SECTION 06

Calculated Dimensions & Measures

The Concept Shift

In Qlik, calculated fields can live inline in a chart or be promoted to Master Items. Logic is written in Qlik expression syntax evaluated against the current selection context. In Astrato, calculated fields are defined in the semantic layer — available everywhere that dataset is used — and written in SQL-compatible syntax evaluated against the live warehouse query.

Before: Qlik Calculated Fields

```
// Calculated dimension (bucketing)
If(amount < 100, 'Small', If(amount < 1000, 'Medium', 'Large'))
// Share of total measure
Sum(amount) / Sum({1} amount) * 100
```

After: Astrato / Warehouse Equivalents

```
-- Bucketing: derived column in warehouse view
CASE
WHEN amount < 100 THEN 'Small'
WHEN amount < 1000 THEN 'Medium'
ELSE 'Large'
END AS amount_tier
-- Share of total: window function in semantic layer
SUM(amount) / SUM(SUM(amount)) OVER () * 100
```

Expression Function Translation Reference

Qlik Function	SQL / Astrato Equivalent
`If(cond, a, b)`	CASE WHEN cond THEN a ELSE b END
`IsNull(field)`	field IS NULL
`Len(field)`	LENGTH(field)
`Left(field, n)`	LEFT(field, n)
`Upper(field)`	UPPER(field)
`Date(field, 'fmt')`	TO_CHAR(field, 'fmt') -- Redshift/PG
`Year(date_field)`	EXTRACT(YEAR FROM date_field)

`Month(date_field)`	EXTRACT(MONTH FROM date_field)
`Dual(label, value)`	CASE to map display labels -- no direct equiv.
`Pick(n, v1, v2, v3)`	CASE WHEN n=1 THEN v1 WHEN n=2 THEN v2 ...
`Concat(field, delim)`	LISTAGG(field, delim) -- Redshift
`RangeSum(...)`	SUM(...) OVER (...) -- window function
`Above(expr)`	LAG(expr) OVER (ORDER BY ...)

SECTION 07

Master Items → Astrato Reusable Objects

The Concept Shift

Qlik Master Items are a governed library of reusable dimensions, measures, and visualizations defined once and shared across all sheets in an app. In Astrato, the semantic layer itself is the Master Item library — defined once, available to every workbook using that dataset. The governance upgrade: Qlik Master Items are app-scoped. Astrato semantic layer definitions are workspace-scoped, shared across all workbooks connected to the same dataset.

Before: Qlik Master Items

```
// Master Measure
Name: Net Revenue
Expression: Sum(amount) - Sum(returns)
Label: Net Revenue
// Master Dimension with value map
Field: segment
Value map: ENT='Enterprise', MM='Mid-Market', SMB='SMB'
```

After: Astrato Semantic Layer

```
-- Semantic layer measure (available all workbooks on this dataset)
Measure name: net_revenue
Display label: Net Revenue
Expression: SUM(amount) - SUM(returns)
Format: Currency
-- Dimension label mapping: handle in warehouse view
CASE
WHEN segment = 'ENT' THEN 'Enterprise'
WHEN segment = 'MM' THEN 'Mid-Market'
WHEN segment = 'SMB' THEN 'SMB'
ELSE segment
END AS segment_label
-- Expose segment_label as the dimension in Astrato.
```

Master Items Migration Decision Table

For each Master Item, ask…	Action
Simple field alias or label?	Rename column in warehouse view
Derived / calculated field?	Add to warehouse view OR semantic layer

Measure (aggregation)?	Define as semantic layer measure
Color / formatting rule?	Recreate in Astrato visualization settings
Used across multiple apps?	Define at semantic layer -- workspace-wide reuse
Contains set analysis?	See Section 03 -- move logic to warehouse

MIGRATION AUDIT CHECKLIST

Pre-Migration Audit Worksheet

Use this table as your audit pass before migration begins. Every Qlik element type needs to be inventoried and assigned a migration path before work starts.

Qlik Element	Inventory Action	Migration Path
`QVD library`	List all QVDs + row counts	Export via STORE or Talend (Section 01)
`Load script data sources`	Document all source connections	Confirm warehouse equivalents exist
`Load script transformations`	Categorize each transformation	Warehouse view vs. semantic layer
`Set analysis expressions`	List all unique patterns	SQL window functions or pre-agg views
`Variables`	Name, type, default, triggers	Recreate in Astrato; Switch Action for multi-var
`Calculated dimensions`	List all inline + master	CASE expressions in warehouse view
`Calculated measures`	List all inline + master	Semantic layer measures
`Master Items`	Full inventory	Semantic layer definitions
`Conditional show/hide`	Document all conditions	Astrato conditional visibility
`Chart types used`	List all chart types	Confirm Astrato equivalents available
`QVD refresh schedule`	Document reload frequency	Replace with warehouse pipeline

ABOUT 2516 TECHNOLOGIES

2516 Technologies is an AI readiness and analytics modernization consultancy focused on regulated industries — pharma, defense, and financial services. We help organizations build the data foundation that makes AI work: governed, warehouse-native, and audit-ready.

Working through a Qlik → Astrato migration in a regulated environment? We offer implementation support, semantic layer architecture, and migration execution for teams that can't afford to get it wrong.

2516technologies.com

Mark Henning · Founder & CDO · mark@2516technologies.com