



DATABLAU · DAM

Security Gateway

Product module user manual

Version 9.1.2

Contents

01	Security gateway
02	Security gateway quick start
03	Audit access and troubleshoot gateways
04	Check gateway compatibility
05	Configure Datablau JDBC clients
06	Configure gateway groups and servers
07	Verify dynamic masking and query audit

Security gateway

Security Gateway is the DDS dynamic masking module. The server manages groups, gateways, ports, states, and audit. Clients use Datablau JDBC Driver to access backend data through it.

Dynamic masking changes sensitive values presented during policy-controlled access, not source records. Verify actual queries and logs; a saved rule does not prove gateway routing.

Client basics

- URL: `jdbc:datablau://[host]:[port]`
- Driver: `com.datablau.jdbc.Driver`
- Available guidance covers virtual accounts, DAM users, DBeaver, and FineBI.

Find a task

- [Quick start](#)
- [Server and login modes](#)
- [JDBC and BI clients](#)
- [Result verification](#)
- [Audit and troubleshooting](#)
- [Compatibility](#)

Compatibility

Confirmed types include Oracle, Hive, MySQL, PostgreSQL, DB2, and Vertica. Historical tests directly covered MySQL, Oracle, and Hive. Use only documented version combinations; do not infer missing versions. See [Compatibility notes](#).

Security gateway quick start

Screenshot note: Screenshots are illustrative. Controls or labels may remain untranslated in the demo release, and pages without a faithful English capture retain the source illustration. Control names and layouts can vary by release.

Use one account to query a few approved samples through JDBC and verify dynamic masking.

Before you begin

Requirement	Check
Source/samples	Available objects and approved scope
Identity	DAM or virtual mode and intended user
Expectations	Allowed rows, masked and unchanged columns
Gateway/driver	Address, service port, package
Client	Custom-driver and connection rights

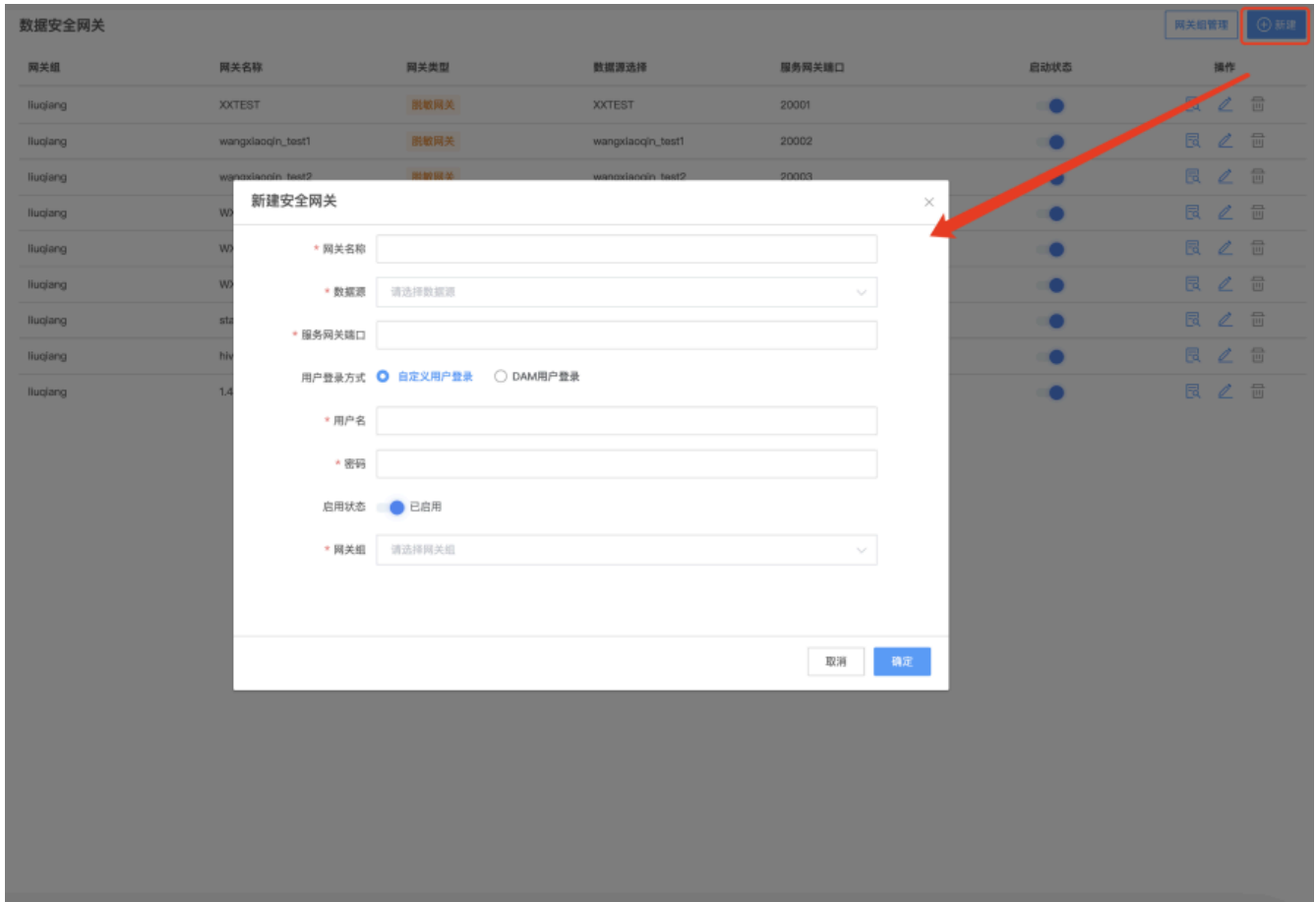
Use actual business identities and authorized samples, not administrator substitutes or altered sensitive production data.

1. Configure scope and masking

Prepare [Rules](#) and [Policies](#) for the same identities/objects. Record field expectations. Saved rules alone do not prove masking.

2. Configure the server

Follow [Server setup](#), enable, and inspect state.



Verify source, port, login mode, advertised addresses, and credential origin. Click to open the full image.

3. Connect the client

Follow [JDBC setup](#). In DBeaver add `com.datablau.jdbc.Driver`, then use `jdbc:datablau://[host]:[port]` with real values and mode-matched credentials. Test and browse. Resolve network/service/driver/authentication before changing masking.

4. Query and inspect audit

1. Query explicit policy-covered fields within scope.
2. Compare rows, masked fields, and other columns with expectations.
3. Record gateway, identity, time, and result.
4. Find the same SQL in Log Audit.
5. Extend approved cases through [Verification](#).

Results must match expectations and audit. Source data remains unchanged.

Completion and next steps

Verify connection, policy persistence, and masking separately before expanding. For BI identities, use [FineBI integration](#) and test mappings/defaults independently.

Audit access and troubleshoot gateways

Gateway audit covers connections, authentication, SQL, and results. DDS platform operation logs are separate.

Review a request

1. Record account, client, gateway, target, and query time.
2. Filter logs by time, gateway, user, and result.
3. Check JDBC routing, matched policy, and status.
4. Compare raw/masked client values and target columns.
5. Mask accounts, addresses, and data in exported evidence.

Diagnose failures

Symptom	First checks	Follow-up
Connection failure	State, port, firewall, URL	Minimal network/authentication query
Authentication failure	Mode, DAM/virtual account	Disabled/expired account
Raw values	Objects, mapping, precedence, timing	Single-field test
Query fails, gateway healthy	Backend credentials, parameters, dialect	Validate backend connection
No audit	Actual route, time, proxy identity	Correlate client logs

Use dedicated test data and least-privilege accounts instead of repeatedly querying sensitive data.

Retain evidence

Record goal, identity, gateway/backend, policy, time, client result, server log ID, and conclusion.
Performance tests do not replace policy validation.

Check gateway compatibility

Compatibility depends on backend, Datablau driver, client, and SQL dialect. Confirmed types include Oracle, Hive, MySQL, PostgreSQL, DB2, and Vertica; undocumented versions are not implied.

Build a validation matrix

Dimension	Record
Backend	Type, deployment, charset, connection
Driver	Class, package, parameters
Client	Tool/BI product, runtime
SQL	Queries, paging, aggregates, functions, transactions, large fields
Policies	Column/table masking, rows, exceptions
Outcome	Connection, authentication, query, masking, audit

Validate in order

1. Connect as a least-privilege user.
2. Query nonsensitive data for dialect/charset.
3. Test one masking field against expected values.
4. Add paging, aggregation, and large results.
5. Correlate gateway and backend logs.

Versions, thresholds, and production readiness require actual test reports or product confirmation.

Configure Datablau JDBC clients

Clients connect through the gateway using Datablau JDBC Driver. Direct database connections do not automatically apply gateway policies.

Connection parameters

Parameter	Value
URL	<code>jdbc:datablau://[host]:[port]</code>
Driver class	<code>com.datablau.jdbc.Driver</code>
host	Deployed gateway address
port	Proxy source's gateway service port
Credentials	Match DAM or virtual-account mode

Replace bracketed placeholders. Use administrator-provided packages and the Datablau class, not a native database class.

Connect database tools

1. In DBeaver, open **Database > Driver Manager** and create a driver.
2. Add the JAR and class.
3. Create a connection with the gateway URL and credentials.
4. Test and browse expected objects.
5. Query known samples using [Verification](#).

Check generated URLs even with driver templates. Match gateway detail links/login modes. Expand database/schema/tables/columns and test explicitly covered fields. Separate connection failures from masking problems.

Prepare FineBI integration

Deploy `datablau-driver-jdbc-bridge.jar`, the companion thread-context agent, and JDBC plugin from the approved package. Add companion agent parameters to actual startup configuration, start, then create the source.

The bridge passes the BI identity through thread context. Matching BI/DAM usernames use that DAM user's policies; passwords may differ. Nonmatching names use default permissions. Test real user results, not only administrator connections. Use deployment-specific paths/scripts and revalidate identity propagation after upgrades.

Troubleshoot

Symptom	Checks
Class missing	Package loading and name
Refused	Enabled service, host, port, network
Authentication failure	Mode and credentials
Unexpected data	Actual route, identity, scope
Identical BI results for all users	Propagation, name mapping, defaults

Configure gateway groups and servers

Prepare a data source, deployment address, service port, access policies, and masking rules.

Create a group

Open **Data Security > Security Gateway > Gateway Group Management** and add name/code. Check associated gateways before editing/deleting. The code must match deployed JDBC gateway configuration. Creating a group does not deploy/start a service; confirm operations has started the matching service.

Create a gateway

1. Select **New Security Gateway**.
2. Enter name, source, and group.
3. Set the gateway service port, not the database's original port.
4. Choose login mode and credentials.
5. Enable and save.
6. Verify configuration/state, then test a client.

Use Service Gateway Connection Information in details. Verify each advertised address and enforcement. Details need not repeat edit-form port/type fields separately. Authorized administrators can download `datablau-jdbc.jar`; retest after replacement.

Choose an identity mode

Mode	Credentials	Policy identity
DAM user	DAM username/password	That user's policies
Custom user / JDBC virtual account	Gateway-configured username/password	Matching DAM username's policies; password is the gateway password

Virtual accounts commonly serve applications; DAM users commonly use database tools. Do not interchange passwords.

Change or disable

Edit source, port, or login mode and retest. Enabled gateways cannot be deleted; assess impacts and disable first.

Next: [JDBC clients](#).

Verify dynamic masking and query audit

Check identity, results, and audit together using approved samples and accounts.

Prepare cases

Record database, gateway, tables/columns, identity, and expectations.

Case	Expectation
Allowed access	Approved scope readable
Restricted access	Excluded objects unavailable through normal allow path
Row filtering	Returned records satisfy conditions
Masking	Selected fields follow rules
Other fields	No incorrect rule applied
Different users	Their own identities/policies respected

Do not substitute administrators for all roles.

Execute and compare

1. Connect using [JDBC configuration](#).
2. Confirm mode/username and run a scoped query.
3. Compare rows, fields, and presentation.
4. Repeat applicable cases for each identity.
5. Find the same SQL in gateway audit.
6. Retain passed/failed cases, errors, and results.

Presentation changes do not rewrite source data.

Distinguish logs

Gateway audit records SQL; security operation logs record administration. For missing audit, check direct connections, selected gateway, and time filters.

Check compatibility

Confirmed types include Oracle, Hive, MySQL, PostgreSQL, DB2, and Vertica. Validate the database/client/driver/gateway combination. Historical tests do not establish arbitrary SQL, concurrency, or fixed performance guarantees.