

Dova and Administrator English manual

Step-by-step data design, operation and platform administration.

EDITION 7.0.0 · OCTOBER 2026 · ENGLISH

41 chapters · Procedures · Illustrated examples · Laboratories · Reference

Mapping

Defines the transformation.

Session

Binds environment and parameters.

Workflow

Coordinates the execution.

Online manual and controlled learning procedures:

docs.cloudata.solutions/dova/en



Contents

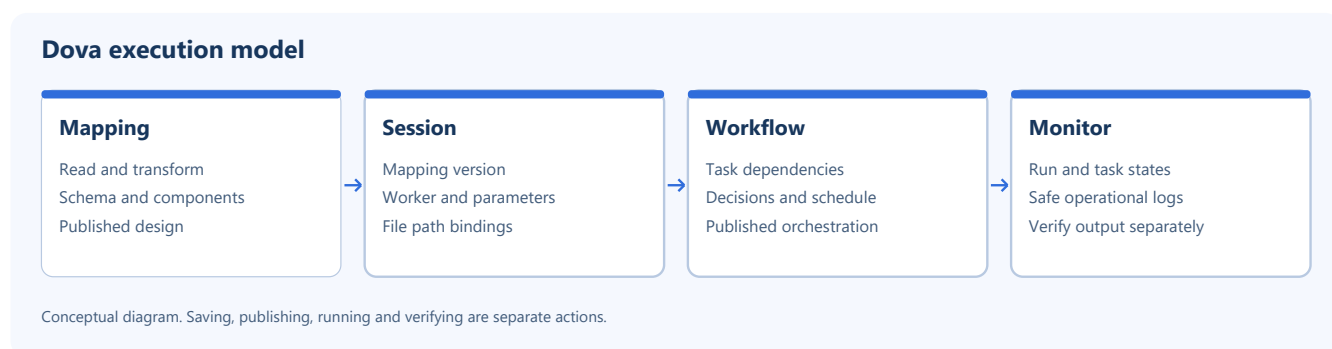
Your first verified Workflow: a complete guided exercise	4
Laboratory 1: deterministic customer normalization	6
Laboratory 2: join TXT and Excel into a regional summary	8
Secure operation: run, diagnose, recover and retire capacity	9
CLI and API: controlled, verifiable automation	11
Overview: understand the active workspace	14
Repository: organize Domains, Folders and objects	15
Mappings: design, validate and publish transformations	16
Sessions: bind a Mapping to its execution context	18
Workflows: orchestrate, publish, run and schedule	20
Monitor: follow attempts, diagnose failures and retain evidence	22
Connections: configure reusable access without exposing credentials	23
Notification channels: prepare and test reusable destinations	25
Workers: inspect eligible execution capacity	26
Configuration: runtime policy, profiles, scheduler and health	28
Plans and subscriptions: understand offers and workspace continuity	29
Developer Tools: download the CLI and manage personal API keys	31
Account menu: choose the workspace, role and appearance	32
Profile: personal preferences, password and data requests	33
Open items: work safely across Mapping, Session and Workflow editors	35
Administrator Overview	37
Cloud workers: inventory, profiles and provisioning	38



Users and account access	40
Tenants and workspace capacity	41
Roles and permissions	42
Subscription, billing, licenses and checkout	44
Audit trail and administrative evidence	45
Customer registration review	46
Developer Tools, Dova CLI and API keys	47
Preferences, appearance and limit policies	49
External authentication, SSO and MFA	50
Global catalog and tenant offers	52
Execution email templates, SMTP and notifications	53
Install workers on Linux, Windows Server and cloud	54
Worker reassignment, dependency backups and removal	56
ETL data cleanup and tenant lifecycle	57
Connectors and authentication: prepare the correct contract	59
Mapping sources and targets: formats, fields and runtime bindings	63
Transformations and quality rules: verify values, rows and ordering	66
Parameters, templates and Workflow conditions	69
Permissions and actions: distinguish visibility, design and operation	71

Your first verified Workflow: a complete guided exercise

This exercise builds a small, understandable process from a CSV to a verified CSV result. It covers Repository, Mapping, Session, Workflow and Monitor. Use a laboratory tenant with the permissions needed to create, edit, publish and run; those permissions can be separate. Prepare a compatible worker and a source/destination location it can access. A browser-local path is not automatically a worker path.



Conceptual execution model. Complete each responsibility before proceeding to the next.

1. Prepare a predictable source

Create a synthetic file named `customer_demo.csv` with this content:

```
id,name,amount
1," Ana ",10
2," Luis ",20
3,"Marta",30
```

The expected result contains three records and the fields `id`, `name` and `amount`. The name values should become `ANA`, `LUIS` and `MARTA` after trimming and uppercasing. Decide the expected row count and values before running so that acceptance is independent of the application's success message.

Place the file in an authorized laboratory location accessible by your worker. Reserve an output such as `customer_demo_clean.csv` that no operational process uses. Confirm read access to the input and write access to the destination.

2. Create the repository location

1. Open Repository and confirm the active tenant.
2. Create or reuse a laboratory Domain named `Training`.
3. Create a Folder named `CustomerDemo` in that Domain.
4. Select the Folder and verify the displayed location before creating objects.

Use distinct names rather than replacing a customer's existing objects. A Folder name alone is insufficient to identify its Domain and tenant.

3. Build and publish the Mapping

1. Open Mappings and create `m_CustomerDemo` in the laboratory Folder.
2. Add Flat File input, Clean text and Flat File output.
3. Configure comma delimiter, header and the intended fields on the input.
4. In Clean text, select `name` and the `trim` and `upper` operations.
5. Configure the output format, header, encoding and delimiter explicitly.
6. Connect input → cleaning → output. Select each component to inspect its settings.
7. Choose Validate and resolve the reported problems.
8. Use Save draft, then Publish with the required permission. Record the published version.

The Mapping describes format and transformations. Its Session supplies runtime file paths. Arrange and Fit help readability; they do not validate data or publish a version.

4. Bind and publish the Session

1. Open Sessions and choose New session.
2. Name it `s_CustomerDemo_Test`, select the published Mapping and choose Environment Test.
3. In Runtime, keep Use tenant default or select an authorized compatible override.
4. In File paths, fill the source and target bindings with worker-accessible laboratory paths.
5. Inspect Parameters; leave overrides empty unless this design declares them.
6. Save draft, address validation messages and publish the accepted Session.

Check its Mapping version. Choosing Test is a context label; it does not substitute for a separate output path or isolate an accidentally shared destination.

5. Orchestrate and execute

1. Open Workflows and create `w_CustomerDemo` in the same laboratory location.
2. Add a Session task that references your published Session.
3. Connect the available Start, Session and End tasks in the intended order.
4. Validate, save the draft and publish the accepted Workflow.
5. Run manually before adding any schedule and record the attempt identifier.
6. In Monitor, inspect the global state, task states and safe operational log.

6. Accept the actual output

Open the authorized result and confirm exactly three records, the expected field names and the normalized name values. Also inspect output encoding/header and whether an unwanted extra field appeared. A green task does not replace these checks.

For a controlled negative case, change only the laboratory Session input to a nonexistent file, save and publish the accepted Session, then review and accept the Workflow's dependency update before publishing and running the laboratory Workflow. Inspect the failure. Restore and publish the correct Session binding, review the Workflow dependency again, and inspect a separate new attempt. Keep the old error evidence according to retention policy.

Disable any temporary schedule and remove only laboratory objects whose dependencies and retention permit it. Continue with [Mappings](#), [Sessions](#), [Workflows](#) and [Monitor](#).

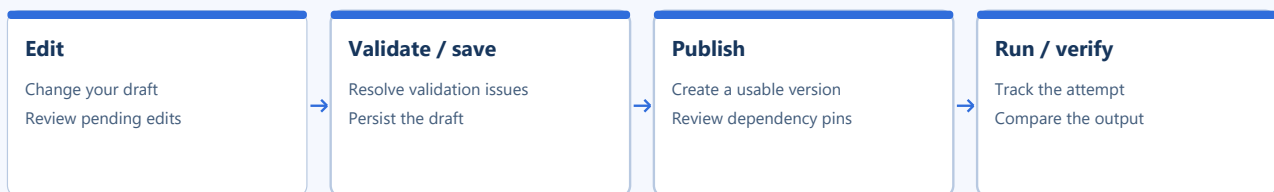
STEP-BY-STEP GUIDES · DOVA 7.0.0

Laboratory 1: deterministic customer normalization

This laboratory processes 1,000 synthetic CSV rows through eight Transform components and expects 667 normalized output rows. It demonstrates text cleanup, replacement, field splitting, concatenation, calculated fields, conversion, filtering and numeric sorting. Names and values are repeatable fixtures, not records of real people.

Use the existing [synthetic laboratory package, edition 6.0.2](#). The fixture format remains applicable to the documented transformation model; the package version is separate from the Dova 7 interface. It contains the sample inputs, expected outputs and coherent Mapping/Session/Workflow definitions. The ZIP is a fixture runner, not the Dova engine, and requires an authorized source checkout with the documented Python dependencies.

From a draft to verified execution



A saved draft does not publish a version, and a successful run does not establish data correctness.

The local transformation verification and a server Workflow run are separate exercises.

1. Inspect the data and expected result

After extracting the package, inspect `transform_workflows/inputs/customers.csv`. Its fields include `customer_id`, `raw_name`, `location`, `amount` and `status`. It intentionally contains repeated spaces, accents, a vertical-bar location separator and alternating `ACTIVE`, `ENABLED` and `INACTIVE` states.

Inspect `transform_workflows/expected/customer_normalized.csv` independently. Do not generate your expected output by simply copying a new engine result. The supplied independent Python oracle establishes the expected values, ordering and byte representation.

2. Review the eight transformations

Step	Operation	Rule to verify
1	Clean	Trim and collapse spaces and remove accents in name/location.
2	Replace	Replace exactly ENABLED with ACTIVE.
3	Split	Split location into city/country and remove the original field.
4	Concat	Build customer key from the cleaned name, # and identifier.
5	Derive	Create the segment from country and status.
6	Convert	Convert amount to a numeric output field and fail on invalid conversion.
7	Filter	Keep ACTIVE records only.
8	Sort	Order numeric amount descending.

Replace precedes Filter, so rows originally marked ENABLED are accepted. Conversion precedes Sort so the order is numeric rather than lexical. Inspect those dependencies before moving components on the canvas.

3. Run the isolated verifier

From the extracted package directory, point the runner to your authorized source checkout:

```
python .\run_pocs.py --repo C:\path\DovaETL --report .\verification.json
```

Replace the illustrative path. The runner does not start a server, create tenant objects or access network endpoints. It copies synthetic inputs into a temporary directory, validates definitions and verifies both documented laboratories. The optional report contains safe counts, hashes and comparison results.

If a dependency is missing, follow the project's authorized dependency setup. Do not modify expected CSVs to make a mismatch disappear. Inspect the first error and the fixture integrity result.

4. Check acceptance

Confirm 1,000 source rows, 667 output rows and byte-for-byte equality with both the independent oracle and supplied expected CSV. Inspect the columns and descending amount order as additional human checks. A correct count alone would not reveal an incorrect key, segment or sort order.

5. Reproduce through the UI

Review the definition in [Mappings](#), configure all bindings in [Sessions](#), and publish a laboratory [Workflow](#). Use worker-accessible synthetic files and an exclusive destination. Run manually and inspect [Monitor](#). Record the actual published dependency versions and run identifier, then compare the server output with the same expected result.



Local verification demonstrates transformation behavior for these fixtures. It does not establish remote worker authentication, storage permissions or production capacity. For a mismatch, inspect Replace/Filter ordering, numeric conversion, the split delimiter and output serialization before changing the oracle.

STEP-BY-STEP GUIDES · DOVA 7.0.0

Laboratory 2: join TXT and Excel into a regional summary

This example combines 1,000 synthetic pipe-delimited transactions with a 1,000-row Excel customer catalog. It filters approved transactions, joins customers to regions and produces a four-row CSV summary. The expected result is calculated by an independent oracle instead of reusing the engine's Join or Aggregate implementations.

Download the existing [synthetic fixture package, edition 6.0.2](#) and extract it. The fixture release is independent of the Dova 7 UI release. Use an authorized DovaETL source checkout and its required Python dependencies. No customer transactions, payment provider or external connection is involved in the isolated exercise.

1. Review both inputs

`transform_workflows/inputs/transactions.txt` contains `customer_id`, `amount` and `status`. Every fifth transaction is `REJECTED`; the others are `APPROVED`. Identifiers and amounts initially arrive as text.

`transform_workflows/inputs/customers.xlsx` contains a `Customers` sheet with a header and 1,000 rows in columns A-C: `customer_id`, `customer_name` and `region`. Its identifier is numeric, and text contains deliberate accent and spacing variations. The expected CSV contains exactly `sales_region` and `approved_total`.

2. Inspect the component order

Component	Required behavior
TXT Convert	Convert amount to numeric <code>amount_number</code> .
Excel Clean	Trim/collapse spaces, remove accents and uppercase customer/region text.
Join	Inner join by integer <code>customer_id</code> ; TXT is left, Excel is right; hash strategy and nulls unequal.
Filter	Keep <code>APPROVED</code> transactions.
Aggregate	Group by region and sum <code>amount_number</code> into <code>total_amount</code> .
Sort	Order total descending.
Rename	Produce <code>sales_region</code> and <code>approved_total</code> .
Select	Retain exactly those two fields.

The two input branches let you diagnose parsing before blaming the join. Numeric key normalization matters because one identifier comes from text and the other from an Excel cell. Duplicate customer keys can multiply joined records and inflate an aggregate.

3. Execute the isolated verification

Run from the extracted fixture directory:

```
python .\run_pocs.py --repo C:\path\DovaETL --report .\verification.json
```

The wrapper verifies both laboratories. It checks fixture integrity, validates definitions, resolves the `source_txt` and `source_xlsx` bindings, and verifies the reported 1,000 rows for the `Customers` sheet. Network and persistent application-storage access are forbidden by the runner. The optional JSON report records synthetic evidence only.

4. Accept the result

Confirm four rows and byte-for-byte equality with the independent oracle and `expected/regional_approved_totals.csv`. The oracle builds a region lookup by integer identifier, skips rejected transactions, sums by normalized region and sorts. Inspect exactly two output fields, UTF-8 encoding, header and LF line endings.

The sample's number representation does not establish a universal accounting or rounding policy. For a real financial process, define precision, scale, currency and rounding separately before implementing the rule.

5. Move into a laboratory Workflow

Publish the Mapping, then configure both source bindings in the Session. A single uploaded file does not replace the two sources. In the CLI, use the multi-source command shape from [CLI and API](#) with actual task and node identifiers. Publish the Workflow, run it manually and compare its output in the same way.

For too few regions, inspect unmatched keys and join type. For inflated totals, inspect lookup uniqueness. For regions split by accents or spaces, inspect Excel cleanup. For additional output fields, inspect Rename, Select and the target schema. Remote file access, worker capability and authorization need their own acceptance checks.

STEP-BY-STEP GUIDES · DOVA 7.0.0

Secure operation: run, diagnose, recover and retire capacity

Operational decisions require the current identity, tenant, permitted action and actual state. This guide connects observable execution to controlled recovery and worker retirement. It does not authorize access to another tenant or provider account, nor does an operational success message establish output correctness.

Worker retirement: evidence before removal



Unavailable is not absent. The workflow stops when provider state or dependency inventory is uncertain.

Conceptual retirement path. Provider availability, active jobs and dependency checks can block progression.

1. Establish the authorized context

Open the account menu and confirm user, tenant and active role. Review the target object identifier and location. Separate permission to view from permission to edit, publish, run or administer cloud resources. Keep MFA, normal expiry and logout behavior intact while troubleshooting.

Use a laboratory tenant for demonstrations. Store no credentials in names, parameter text, screenshots or exported incident reports. A safe error message and timestamp are usually enough to begin investigation without attaching raw data.

2. Prepare an observable run

1. Record the published Workflow, Session and Mapping versions.
2. Check worker eligibility, file bindings and non-sensitive parameters.
3. Use a controlled destination and understand whether retries can duplicate writes.
4. Launch once and keep the attempt identifier.
5. Open Monitor and review task states and the first relevant safe error.
6. Inspect and compare the actual result independently after terminal success.

Queued, waiting, running and retry-wait are active states. A timeout in a browser or CLI does not prove that server work stopped. Look up the existing identifier before repeating a launch.

3. Diagnose before choosing recovery

For file errors, inspect Session paths and worker access. For authentication errors, inspect the Connection method, custody and identity permissions. For ineligible capacity, inspect worker state, capability, slots and policy. For an upstream version notice, review pinned dependencies.

Choose cancellation, termination, repeat or checkpoint recovery according to the current state and supported contract. Checkpoint recovery only applies when usable checkpoint state exists. It does not guarantee undoing side effects at an external destination. Preserve the failed attempt while validating a corrected one.

4. Reassign references before worker removal

1. Inspect the source worker's current provider state and impact.
2. Review default-worker references, active or queued jobs, Sessions, Workflows and credential dependencies.
3. Select an authorized compatible replacement or distribution.

4. Generate Preview and inspect every intended change and blocker.
5. Regenerate Preview after changing a destination or selection.
6. Apply an accepted current plan and inspect the dependency backup/result.
7. Refresh the source impact before proceeding toward retirement.

The interface manages plan tokens and request identity. Do not alter those values to bypass a conflict. Dependency backups preserve supported reference projections; they are not complete VM images or database/file backups.

5. Retire with current provider evidence

If the VM is running, Stop and deprovision asks for explicit confirmation, sends stop, refreshes impact and proceeds only when allowed. Stop and delete use separate operation identities. A stop failure must not lead to delete; a newly discovered dependency must be resolved first.

The Dova record is retired only after provider absence and final reference checks. Unavailable is not absent. Clean up Dova record is appropriate for confirmed absence, not for hiding a provider-access failure. Closing a dialog cannot undo a provider effect already accepted.

Acceptance and incident evidence

Keep identifiers, versions, timestamps, safe messages, output comparison and current provider observations. Confirm the supported result after each stage and retain audit history. Do not claim cloud cost savings until the resource is confirmed retired at the provider. Follow [worker migration and backups](#) for the detailed administrative procedure.

STEP-BY-STEP GUIDES · DOVA 7.0.0

CLI and API: controlled, verifiable automation

The CLI can inspect authorized objects and runs and launch permitted actions. This chapter uses command shapes and routes reviewed in the current client; it is not a complete OpenAPI specification or a promise that every internal route is a stable public integration. Check your installed client's help before automating against a different release.

1. Download, verify and identify the client

Open [Developer Tools](#), select the distribution for your platform and inspect version, signature status and SHA-256. Calculate the downloaded artifact hash and compare it exactly. Install using the distribution instructions, then run `dova --version` to identify the executable on your PATH.

2. Authenticate to an authorized laboratory

Create a personal API key for the exercise, then log in interactively:

```
dova login --url https://dova-lab.example.invalid
```

The `.invalid` host is a nonfunctional placeholder. Replace it with your permitted laboratory environment. Enter the key through the prompt, not a shared command argument. Verify the reported identity and tenant before continuing. Protect the local CLI profile because it stores authentication configuration.

3. Start with read-only operations

```
dova status --output json
dova repository list --domain Training --output json
dova mapping list --name sample --limit 50 --output json
dova workflow list --status published --name sample --output json
dova run list --status failed,cancelled --sort createdAt --descending --output csv
```

Compare identifiers with Repository, Workflows and Monitor. An empty filtered collection can be correct; a failed request must not be converted to a valid empty list. JSON is useful for explicit field processing and CSV for an authorized tabular review. Names are not guaranteed to be unique.

4. Understand the reviewed route shapes

Purpose	Method and route used by the client
Current identity	GET <code>/api/session</code>
Domains/Folders	GET <code>/api/etl/domains</code>
Mappings	GET <code>/api/etl/mappings</code>
Sessions	GET <code>/api/etl/session-definitions</code>
Workflows	GET <code>/api/etl/orchestrations</code>
Workflow attempts	GET <code>/api/etl/workflow-runs</code>
Launch a Workflow	POST <code>/api/etl/orchestrations/{id}/run</code>

The client uses Bearer authentication and JSON responses. Collection keys differ: Workflows use `orchestrations`, while attempts use `runs`. For direct API integrations, confirm the actual response schema and retain normal authorization, failure and tenant handling.

5. Run a published synthetic example

The local laboratories require no network. A server run is a second phase and can write outputs, so first prepare published definitions, authorized inputs and a laboratory-only destination.

```
dova workflow run WORKFLOW_ID --input transform_workflows/inputs/customers.csv --wait --output json
```

For multiple sources, repeat `--source` using `TASK_ID:SOURCE_NODE_ID=FILE`:

```
dova workflow run WORKFLOW_ID --source TASK_ID:source_txt=transform_workflows/inputs/transactions.txt --source TASK_ID:source_xlsx=transform_workflows/inputs/customers.xlsx --wait --output json
```

Replace placeholders with the actual identifiers from your published definition. Do not combine `--input` and `--source`. The client also supports `--var KEY=VALUE` and `--param KEY=VALUE` for permitted values that match the Workflow contract. Keep secrets out of ordinary overrides.



6. Wait, verify and close

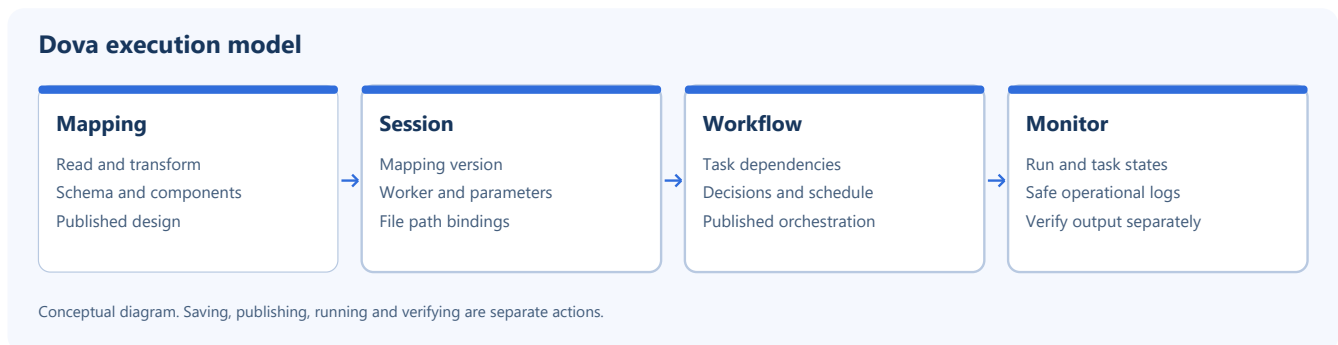
`--wait` polls toward a terminal state; `--poll` and `--wait-timeout` bound its behavior. A client timeout does not establish server cancellation. Query the attempt again before repeating, cancelling or recovering it. Compare the generated output with the independent expected data, not just the terminal command status.

Record client version, environment, object/run identifiers, safe parameters, terminal state and result comparison. Use `dova logout` to remove local authentication state when done, and revoke a temporary key in the UI when it should cease working everywhere. Revocation and local logout are distinct operations.

Overview: understand the active workspace

Overview is the starting point for understanding the operational state of the workspace you are currently using. It summarizes objects, execution activity and runtime health; it does not report business revenue or prove the quality of a transformation result. Use it to decide which module to inspect next.

Before interpreting a counter, open the account menu and confirm the tenant and active role. A change of workspace changes the scope of the information you are reading. A missing action can be correct when your role or subscription does not include it.



Conceptual diagram of the responsibilities behind Overview links.

Step-by-step procedure

1. **Confirm context.** Read the workspace and active role in the account menu. Close the menu and open Overview. For a laboratory, use an authorized synthetic workspace rather than switching to a customer tenant just to populate the cards.
2. **Read the object summaries.** Review the counts and links for available repository objects. Treat these as operational summaries. Follow a card to its underlying module to inspect names, status and the current filters before drawing a conclusion from the total.
3. **Inspect recent activity.** Review the execution activity presented by the page. Identify the relevant attempt and continue in Monitor for task-level state and safe logs. An earlier successful run does not establish the health of a later attempt.
4. **Inspect runtime health.** Review the displayed runtime and capacity indicators. If a worker is offline or ineligible, open Workers or Configuration to inspect heartbeat, capabilities and the selected policy. Do not change the default worker based only on a summary card.
5. **Open the next module.** Use Repository for location and ownership, Mappings for transformation design, Sessions for bindings, Workflows for orchestration, and Monitor for execution evidence. Keep the object identifier when names alone are ambiguous.
6. **Return and reconcile.** After an authorized change, revisit Overview and compare the summary with the underlying module. If a summary appears stale, refresh and inspect the affected record before repeating a save or run request.

Verify the result

Success means you can explain which workspace each indicator belongs to and trace it to an underlying object or run. For a controlled exercise, read the cards, inspect a synthetic Workflow in its module, then inspect its last run in Monitor. No mutation is needed to learn this path.

Errors and recovery

A zero count may reflect an empty workspace, filters or permissions. An error banner is different from an empty collection. Preserve the safe message and inspect the affected module; avoid interpreting an unavailable backend as a valid zero. If the tenant label is wrong, stop and choose the intended context before acting.

Continue

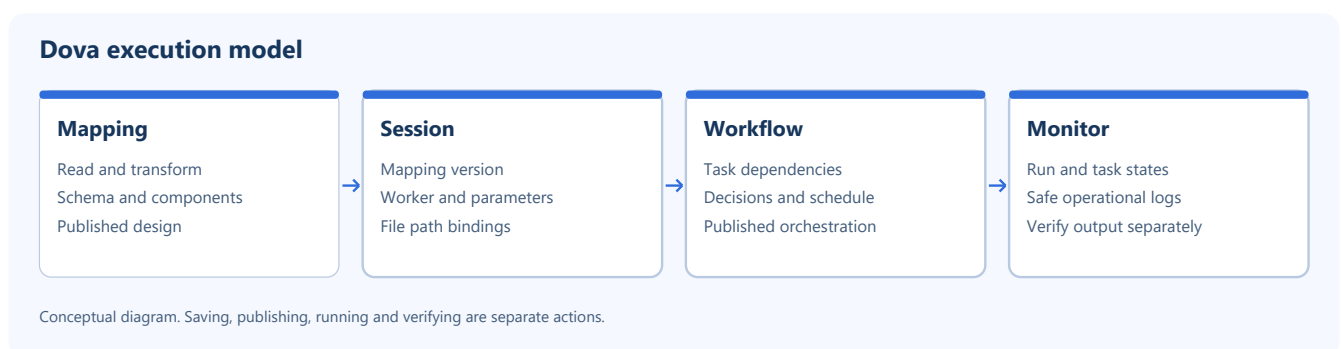
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Repository: organize Domains, Folders and objects

Repository gives Mappings, Sessions, Workflows and Connections a navigable location. Domains and Folders organize the workspace, while the explorer helps you distinguish object types, shared items and their dependencies. A location is part of the object context, not an independent authorization boundary that grants access by itself.

Prepare a simple laboratory structure such as Domain Training with Folder CustomerDemo. Use the role allowed to create or edit that structure. Keep names recognizable and avoid placing customer information or secrets in descriptions.



Repository organizes the designs and execution definitions shown in this conceptual diagram.

Step-by-step procedure

- 1. Inspect the existing tree.** Open Repository and review the Domains and Folders already present. Select the intended Domain before creating anything. Search for a similarly named laboratory Folder to avoid duplicating an existing exercise.
- 2. Create a laboratory Domain.** Use the creation action available to your role. Enter a clear name and description, review the scope and save. Confirm the new Domain appears in the tree before creating its Folder.

3. **Create and select a Folder.** Create CustomerDemo under Training. Select it and check the breadcrumb or location shown by the explorer. A Folder with the same name under another Domain is a different destination.
4. **Browse by object type.** Switch between All, Mappings, Sessions, Workflows, Connections and Shared where available. Use the search and grid controls to locate an object; check its type and identifier before opening it.
5. **Place an object deliberately.** When an editor asks for Domain and Folder, select the laboratory location. Save and return to Repository to confirm the object is listed there. Opening another Folder does not move an object automatically.
6. **Review dependencies before removal.** Inspect the objects and references involved before choosing a delete or move action. Read any blocking dependency message. Resolve the intended references through supported editors instead of changing stored records manually.

Verify the result

For acceptance, create one laboratory Folder, save one synthetic Mapping to it, and reopen that Mapping from Repository. Confirm the same identifier, name and location. Use the Shared view only to verify sharing already authorized within the tenant; it is not evidence of cross-tenant visibility.

Errors and recovery

If an object seems missing, check the selected Folder, object-type filter, search term, tenant and permissions. If deletion is blocked, preserve the object and inspect its references. A Folder containing no visible rows may still have dependencies outside your current filter. Refresh after an accepted save before repeating it.

Continue

[Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Mappings: design, validate and publish transformations

A Mapping defines how records are read, transformed and written. The canvas contains components, ports and connections; the component configuration defines formats, fields and transformation rules. The Session later supplies the execution context and file paths. The Workflow coordinates Sessions.

Keep four actions distinct: editing a draft, Save draft, Publish and Run. Saving persists a design. Publishing establishes a version that authorized consumers can reference. Running processes data. A successful save is not evidence that an output exists.

DATA INTEGRATION

Mappings

Visual regression fixture

Search by name or status

Refresh Clear filters Columns Export CSV Delete selected + New Mapping

0 selected 18 mappings

MAPPING	VERSION	STATUS	COMPONENTS	UPDATED	ACTIONS
Layout mapping 18 Viewport overflow contract	v1	Draft	2	Jul 18, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 17 Viewport overflow contract	v1	Published	2	Jul 17, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 16 Viewport overflow contract	v1	Draft	2	Jul 16, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 15 Viewport overflow contract	v1	Published	2	Jul 15, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 14 Viewport overflow contract	v1	Draft	2	Jul 14, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 13 Viewport overflow contract	v1	Published	2	Jul 13, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 12 Viewport overflow contract	v1	Draft	2	Jul 12, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 11 Viewport overflow contract	v1	Published	2	Jul 11, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 10 Viewport overflow contract	v1	Draft	2	Jul 10, 2026, 7:00 AM 2 mo ago	[Icons]
Layout mapping 09 Viewport overflow contract	v1	Published	2	Jul 9, 2026, 7:00 AM 2 mo ago	[Icons]

Rows 10 Page 1 of 2 | 18 results Previous Next

Connected Active 0 Workers 0 / 0 Queue 0 Plan Unknown Autosave No draft Updated 09:58:54 AM

Dova 7 Mappings manager, desktop light theme, using a synthetic local fixture.

Step-by-step procedure

- Create a named draft.** Open Mappings and create `m_CustomerDemo` in the laboratory Domain and Folder. Follow the editor validation: the name begins with a letter, uses letters, digits or underscores, and is within the displayed limit. The current editor accepts 2-79 characters.
- Add the source and target.** Use Components to add Flat File input and Flat File output for a small CSV example. Select each component and inspect Configuration. Define delimiter, header and columns explicitly so that the sample has a stable schema.
- Add the transformation.** Insert a Clean text component for a synthetic name field, or use the documented customer laboratory chain. Connect compatible ports in the intended direction. A connected port does not prove that a required column exists.
- Check component settings.** Review every source, transformation and target. Keep file format and schema here; put the actual runtime file paths in the Session. For a remote component, select an authorized Connection and ensure a compatible worker is available.
- Arrange and inspect the graph.** Use Auto arrange mapping, Zoom and Fit as needed. Arrange changes draft positions and requires at least two components and an acyclic graph. A wide graph can remain horizontally scrollable rather than being compressed into unreadable nodes.
- Validate and save.** Choose Validate and resolve each listed problem at the indicated component. Then use Save draft. Inspect the result message and any pending-change marker before leaving the editor.
- Publish the accepted version.** When the design is ready, choose Publish with the required permission. Record the published version. Existing Sessions may retain their earlier reference; review their version notice before updating consumers.

8. **Verify through a small execution.** Create a Session and Workflow using synthetic input and an exclusive laboratory destination. After the run, compare row count, columns, values and ordering with an independent expected result. Monitor success alone is insufficient.

Verify the result

The smallest acceptance case uses a two-row CSV and a predictable cleaning rule. Preserve two rows and the intended columns, and verify the transformed values. For the supplied customer laboratory, the documented eight-step chain is clean → replace → split → concat → derive → convert → filter → sort, with 1,000 inputs and 667 expected outputs.

Errors and recovery

For a cycle, fix dependencies before arranging. For a missing column, inspect the source schema and upstream transformations. For a missing file, inspect Session bindings and worker access. Authentication belongs in Connections, not canvas text. After publishing an upstream change, examine the Session and Workflow references explicitly.

Continue

[Repository](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Sessions: bind a Mapping to its execution context

A Session associates a Mapping with Environment, worker selection, execution profile, recovery, typed parameters and file path bindings. It does not replace the Mapping algorithm. File paths refer to the runtime worker; a file on the browser computer is not automatically accessible there.

Prepare a published Mapping, an authorized location and compatible execution capacity. Read access, editing, publication and execution are separate permissions. The manager supports search, environment/status/runtime filters, Columns, Export CSV and pagination; summary cards filter rather than modify Sessions.

DATA INTEGRATION
Sessions
Visual regression fixture

Total sessions **18** Published **0** Disabled **6** Production **4** Refresh Clear filters Filters Columns Export CSV New session

Search: Name, mapping, worker or region Environment: All environments Status: All statuses Runtime: All runtimes

18 of 18 sessions 0 selected Publish Disable Delete

SESSION	MAPPING	ENVIRONMENT	STATUS	UPDATED	ACTIONS
Layout session 18 ses_layout_18	Layout mapping 01 Mapping v1	Stage	Disabled	Jul 18, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 17 ses_layout_17	Layout mapping 01 Mapping v1	Test	Active	Jul 17, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 16 ses_layout_16	Layout mapping 01 Mapping v1	Dev	Active	Jul 16, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 15 ses_layout_15	Layout mapping 01 Mapping v1	Prod	Disabled	Jul 15, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 14 ses_layout_14	Layout mapping 01 Mapping v1	Stage	Active	Jul 14, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 13 ses_layout_13	Layout mapping 01 Mapping v1	Test	Active	Jul 13, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 12 ses_layout_12	Layout mapping 01 Mapping v1	Dev	Disabled	Jul 12, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 11 ses_layout_11	Layout mapping 01 Mapping v1	Prod	Active	Jul 11, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 10 ses_layout_10	Layout mapping 01 Mapping v1	Stage	Active	Jul 10, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]
Layout session 09 ses_layout_09	Layout mapping 01 Mapping v1	Test	Disabled	Jul 9, 2026, 8:00 AM 2 mo ago	[Edit] [Info] [Refresh] [Delete]

Rows: 10 Page 1 of 2 | 18 results Previous Next

Connected Active 0 Workers 0/0 Queue 0 Plan Unknown Unknown Autosave No draft Updated 09:59:04 AM

Dova 7 Sessions manager, desktop dark theme, with synthetic local data.

Step-by-step procedure

- Create the Session.** Choose New session and name the exercise s_CustomerDemo_Test. In General, select the published Mapping, Domain and Folder. Review the referenced Mapping version and any update notice before continuing.
- Set the environment.** Choose Development, Test, Stage or Production according to the context. Test is appropriate for this exercise. Selecting Production is a classification: it does not publish the Session or grant permission to run it.
- Configure Runtime.** Keep Use tenant default unless a deliberate override is needed. If allowed, enable Override worker and select compatible capacity. Review Execution profile and whether it inherits from the Workflow. Provider and region information comes from the worker and is read-only.
- Choose recovery deliberately.** Review the available fail-fast, retry and checkpoint policies. A checkpoint option only helps when the engine and task have persisted a usable checkpoint. It does not guarantee rollback of writes to an external destination.
- Complete File paths.** Open File paths after choosing the published Mapping. Fill every required source and target binding with a path accessible by the selected worker. Use a destination exclusive to the laboratory to avoid overwriting another attempt.
- Inspect Parameters.** Enter only overrides defined by the Mapping and with matching types. Use Parameter preview to inspect name, type and resolved value. Do not paste credentials into parameter JSON or descriptions.
- Save and publish.** Save draft, resolve validation issues and publish the accepted Session. Reopen it from the manager to confirm the persisted settings and Mapping reference. Then associate it with the laboratory Workflow.

8. **Compare contexts.** Create a second Session for Stage with the same Mapping version and equivalent input, but another output path. Run both through a laboratory Workflow and compare the transformed data while preserving independent files.

Verify the result

A typed override is valid only when the Mapping declares that parameter:

```
{"batchSize":{"type":"integer","value":5000}}
```

For acceptance, confirm the saved Mapping version, environment, worker policy, all file bindings and the parameter preview. Run the sample and compare outputs. Target timestamps may be inherited, enabled or disabled; an optional %RUN_TIMESTAMP% template affects the filename while preserving its directory.

Errors and recovery

A missing path should be corrected in this Session, not in a separate environment. An ineligible worker may reflect state, slots, capabilities, tenant policy or profile exclusions. Invalid JSON should be fixed without discarding other draft fields. Review consumers after accepting a Mapping version update; publication does not silently update every Workflow.

Continue

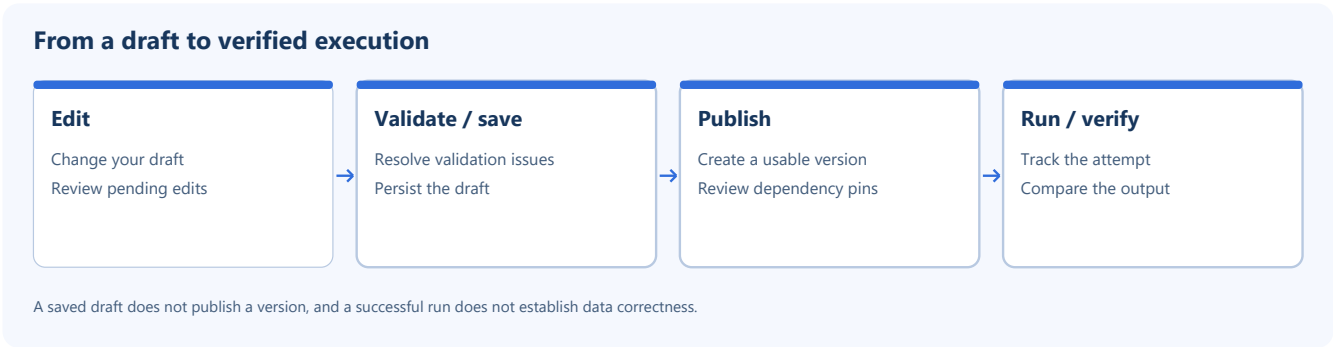
[Repository](#) · [Mappings](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Workflows: orchestrate, publish, run and schedule

A Workflow coordinates tasks and the dependencies that determine their execution order. Session tasks bind transformations to a runtime context. Decision tasks choose complementary branches; timers and terminal tasks express the intended orchestration. Published versions should be reviewed together with the Session and Mapping versions they consume.

Prepare a published Session and compatible worker before testing. Separate permissions may be required for editing, publishing, running and schedule management. Use a laboratory Domain and Folder and a distinct output destination.



Conceptual lifecycle: edit, validate/save, publish, then run and verify.

Step-by-step procedure

1. **Create the Workflow.** Open Workflows and create `w_CustomerDemo`. Review its name, Domain, Folder and sharing scope. Open the editor and save the initial draft before building a more complex orchestration.
2. **Add a Session task.** Add a task that references your prepared Session. Inspect its settings and the version of the dependency. Avoid selecting another similarly named Session without checking the identifier and execution environment.
3. **Connect the task graph.** Connect Start, Session and End for the smallest exercise. For additional tasks, review each dependency and link behavior. Use only the task and condition types available in the editor rather than assuming a general scripting language.
4. **Configure decisions when needed.** Enter the True branch expression (JSON); the interface generates the False branch automatically as its direct negation. Verify the referenced task or parameter identifiers. Do not create two independent conditions that could leave both branches active or neither active.
5. **Validate, save and publish.** Inspect the graph with Arrange and Fit, then validate it. Save draft after resolving errors and publish an accepted version. Save and Publish do not start an execution.
6. **Run manually first.** Launch the laboratory Workflow with the authorized inputs. Treat overrides as values for that attempt; they are not persistent edits to the Session or Workflow. Record the attempt identifier and non-sensitive parameter values.
7. **Inspect tasks and output.** Open Monitor and review global and task-level state, selected branches and the first relevant error. Compare the output with the expected synthetic result before calling the exercise successful.
8. **Add a reusable schedule.** In Configuration → Workflow Scheduler, prepare a schedule and assign it from the Workflow. Review its calendar, timezone and enabled state. Check the next planned execution and a resulting run separately; an enabled schedule is intent, not completion evidence.

Verify the result

Acceptance requires a published Workflow with the intended dependency versions, a terminal attempt consistent with the task graph, and an output that matches its oracle. The profile timezone controls presentation; a schedule has its own explicit timezone. Before repeated loads, confirm whether the destination supports retries without duplicates.

Errors and recovery

For a task that has not started, inspect parents, worker state, capacity and policy. For a failed task, inspect its first safe error rather than only the global state. A client timeout does not prove cancellation. For a dependency-version notice, review and republish deliberately. Checkpoint recovery depends on persisted state and cannot guarantee reversal of external writes.

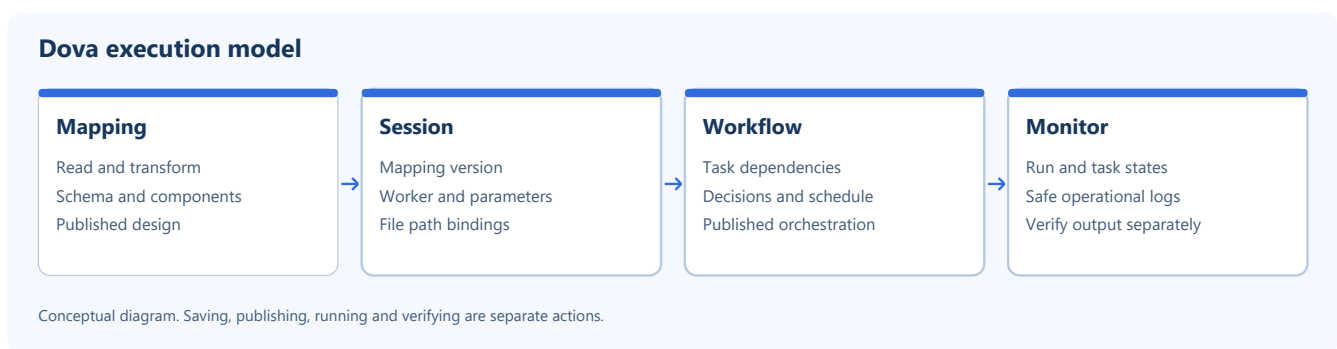
Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Monitor](#) · [Configuration](#).

Monitor: follow attempts, diagnose failures and retain evidence

Monitor is the execution evidence surface. It helps you distinguish a definition from an attempt and a Workflow status from its task-level outcomes. Use filters and operational logs to trace an attempt without exposing input rows, credentials or unrestricted payloads.

Before troubleshooting, keep the run identifier, tenant, Workflow version and approximate time. Names may be reused and a new attempt can have a different outcome. Read access and actions such as cancellation, recovery or history cleanup are separately authorized.



Monitor tracks the attempt; output validation remains a separate acceptance step.

Step-by-step procedure

- 1. Locate the attempt.** Open Monitor and narrow the grid by the relevant name, state or time controls. Clear an accidental search term if a known run is missing. Match the identifier and creation time rather than relying only on the Workflow name.
- 2. Read the global state.** Identify whether the attempt is still active or terminal. Queued or waiting is different from failed; cancelled is different from succeeded. Refresh the record before taking an action that depends on its current state.
- 3. Inspect individual tasks.** Open the run detail or operational log view available for the attempt. Review task states and dependencies in order. Find the first relevant failure and distinguish a skipped downstream task from the cause of the problem.
- 4. Check runtime context.** Review worker, parameter and file-binding context through the associated definitions. If the job is waiting for capacity, inspect Workers and Configuration. Do not start duplicate attempts just because the grid has not yet changed.
- 5. Choose a justified action.** For cancellation, termination, repeat or checkpoint recovery, read the action and confirmation carefully. Check the current state and destination effects. A retry can create another output or repeat writes when the destination is not idempotent.
- 6. Verify the actual result.** After a terminal success, locate the authorized output and compare row count, schema and expected values. Task success records execution behavior; it does not establish the business correctness of the transformation.

7. **Review cleanup separately.** History cleanup applies to completed history and has its own confirmation and retention scope. Inspect selected attempts and exclusions first. Do not remove evidence required for diagnosing an unresolved failure.

Verify the result

Create one successful laboratory run and one controlled failure using a missing synthetic source path. Locate both attempts, identify the failing task, restore the path, and observe a separate new attempt. Acceptance is a safe explanation of the cause and a verified result, with the earlier evidence retained according to policy.

Errors and recovery

An empty filtered grid is not the same as a failed request. If status changes while a dialog is open, reload before confirming. Keep exported reports in an authorized location. For provider or authentication errors, copy only the safe message and timestamp; never attach cookies, keys or decrypted values to an incident.

Continue

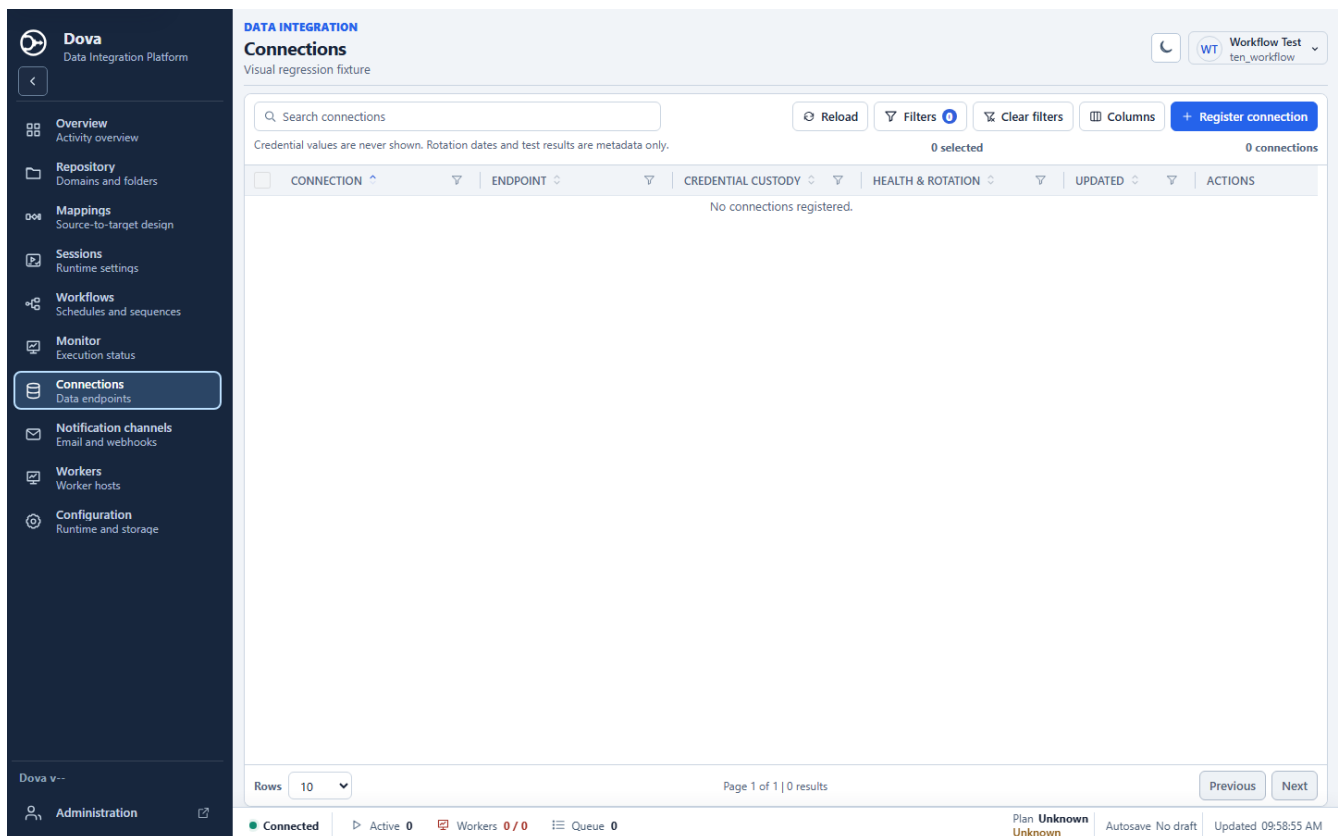
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Connections: configure reusable access without exposing credentials

A Connection separates reusable endpoint metadata, authentication method and credential custody from a Mapping operation. The selected connector determines the fields and methods shown; the worker must still have the required driver, network path and identity permissions. Catalog presence is not proof of configured access.

Prepare a read-only laboratory endpoint, an authorized authentication method and a compatible worker. Keep secrets in the supported custody flow. Use synthetic names and endpoint placeholders in documentation; avoid placing values or secret references in a screenshot.



Dova 7 Connections, desktop light theme. All displayed metadata belongs to a synthetic local fixture.

Step-by-step procedure

- Select the connector.** Open Connections and start the creation action allowed to your role. Select the intended connector and inspect its required fields. A database name, project identifier and HTTP path are different metadata even when they are all text.
- Enter endpoint metadata.** Provide an authorized host or HTTPS endpoint, port and connector-specific database/schema or resource information. Use the provider-specific defaults only as a starting point and confirm them with your environment owner.
- Choose custody and authentication.** Select the supported custody mode and one of its valid methods. Passwordless uses a runtime identity; a customer vault uses a protected reference; other protected flows collect values without redisplaying them. Do not force a hidden method into the form.
- Complete the protected input.** Follow the credential or enrollment instructions for the chosen method. Confirm the principal has only the resource permissions needed for the sample. A syntactically valid reference does not prove that a worker can resolve it.
- Save and inspect.** Review the label, tenant, connector, endpoint metadata and custody method, then save. Return to the inventory and inspect the persisted record. Metadata editing, credential rotation, testing and revocation have different effects.
- Test the intended path.** Use Test for the saved Connection and read the safe result. Then perform a small read-only extraction through a Mapping/Session on the compatible worker. A connection test does not certify every read, write or recovery mode.
- Verify and maintain.** Compare the small output with an independent expected result. Record the Connection identifier and relevant method without values. Review dependencies before revocation; rotate credentials through the supported action when needed.

Verify the result

For acceptance, a saved Connection exposes only permitted metadata, the test gives the expected safe status, and a small authorized extraction produces the expected rows. Consult the connector matrix for declared methods and metadata. File server protocol matters: SFTP uses SSH, while FTP is not an encrypted transport.

Errors and recovery

For authentication failure, inspect method, custody, principal permissions and expiry. For network failure, inspect DNS/endpoint, TLS and worker egress. For a missing driver, check installed worker capabilities. Keep errors safe; avoid pasting vault references or credentials into descriptions to make a failed form easier to debug.

Continue

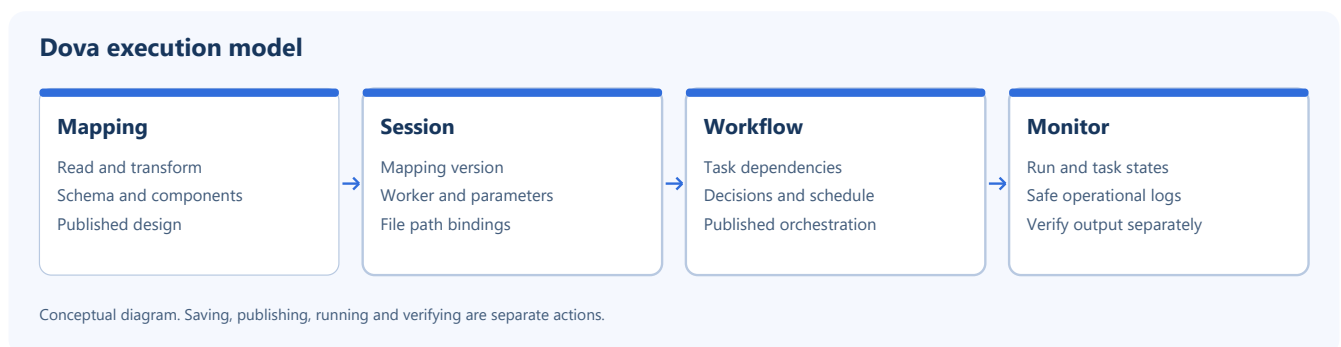
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Notification channels: prepare and test reusable destinations

Notification channels define reusable delivery destinations for authorized events. A channel is different from the notification policy, the execution email template and the platform SMTP transport. Configuring a destination alone does not establish when an event is generated or whether a recipient received it.

Choose a controlled laboratory destination and a role allowed to manage the channel. An endpoint or recipient can be sensitive even when it is not a password. Keep real customer addresses and webhook URLs out of documentation and screenshots.



Notifications follow execution events; event creation and external delivery are distinct checks.

Step-by-step procedure

1. **Choose the destination purpose.** Open Notification channels and inspect existing entries. Decide which event or team needs the channel and whether an existing authorized destination should be reused rather than copied.

2. **Create the channel.** Use the available creation action and enter a recognizable laboratory label. Choose the offered channel type and complete its required endpoint or recipient settings. Follow the type-specific validation shown by the form.
3. **Review scope and sensitive input.** Confirm the active tenant and which authorized audience the channel serves. Use the protected credential flow when the type requires it. Never replace a protected field with a secret embedded in a URL or description.
4. **Save before testing.** Save the channel and inspect the accepted result. Reopen the persisted record when needed. A test should validate the stored configuration rather than an unsaved draft that was never accepted.
5. **Send a controlled test.** Use the Test action where available and send only synthetic content to the laboratory destination. Read the reported result and inspect the receiving sandbox system. A local success message may describe handoff rather than final delivery.
6. **Associate the policy.** Review the relevant execution or administrative notification settings. Configure which authorized events use the destination and how suppression, severity or recipient rules apply. Do not confuse this with editing the email body.
7. **Check end-to-end delivery.** Trigger a synthetic event and review the delivery history. Compare generated, sent, failed or suppressed state with the receiving system. Restore temporary policies when the exercise finishes.

Verify the result

Acceptance requires a persisted channel, an expected test response and an independently observed laboratory delivery or a clearly explained suppression. Preserve a safe identifier and timestamp for troubleshooting. For email, verify transport and template separately; for webhooks, respect the endpoint contract and retry behavior.

Errors and recovery

If a test fails, inspect safe validation and transport messages without exposing the endpoint credential. If an event is suppressed, inspect notification policy before changing the channel. If delivery is delayed, check its current state before sending duplicates. Removing a channel may affect consumers; inspect references and use the confirmation shown.

Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

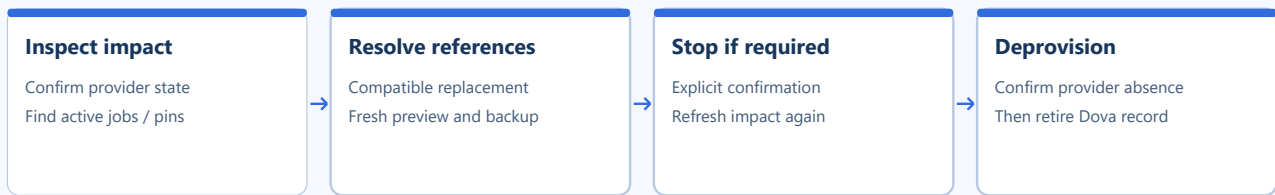
DOVA WORKSPACE · DOVA 7.0.0

Workers: inspect eligible execution capacity

Workers represent the capacity that executes a Session or Workflow under tenant policy. Online status, provider power state and execution eligibility are related but different observations. A worker can be online yet ineligible because of capabilities, slots, exclusions, sharing rules or lifecycle policy.

Prepare an authorized laboratory host or cloud capacity and know who owns it. Registration uses protected enrollment; tokens are not documentation content. Selecting a worker does not grant access to a database or automatically copy source files onto that host.

Worker retirement: evidence before removal



Unavailable is not absent. The workflow stops when provider state or dependency inventory is uncertain.

Conceptual worker retirement path; inspection and references remain required before removal.

Step-by-step procedure

- 1. Inspect the inventory.** Open Workers and locate the intended record. Review its identifier, tenant, state, version and recent heartbeat. Use grid search and columns rather than assuming a familiar name identifies the correct machine.
- 2. Review capabilities and capacity.** Open detail and inspect reported resources, supported connectors and slot capacity. Compare these with the Session requirements and execution profile. A compatible option can still be temporarily full.
- 3. Register only authorized capacity.** If using an existing host, start Register worker and select the supported operating system. Follow the generated enrollment instructions and use the hidden token prompt. Confirm the worker registers and reports a recent heartbeat.
- 4. Inspect health before assigning work.** Review service health and safe diagnostics. If the host was stopped, distinguish restarting the provider VM from restarting the worker service. Cloud provisioning and provider controls belong to Administrator.
- 5. Assign a small laboratory Session.** Use tenant default or an authorized explicit override in the Session/Configuration. Confirm file paths are available on that worker and the Connection method is resolvable from its identity.
- 6. Run and inspect.** Execute a small synthetic CSV process and observe the attempt in Monitor. Compare its result with the expected data. Record worker identity and package version, not enrollment secrets or private host details.
- 7. Maintain references deliberately.** Before removing or replacing a worker, review default policy, active jobs, Sessions, Workflows and dependencies. Use the supported migration/retirement workflow rather than deleting a local record to hide a provider problem.

Verify the result

Acceptance means the record is current, eligible for the selected task and capable of completing a verified laboratory run. Slot counts are weighted execution units: four slots can accept four weight-one jobs or two weight-two jobs, subject to other resource limits. Do not infer unlimited capacity from Online.

Errors and recovery

For no heartbeat, inspect the host service and permitted egress. For ineligibility, inspect tenant/pool rules, lifecycle, capabilities and policy. Provider unavailable is not proof of an absent VM. A failed removal preserves evidence and should be retried through the supported workflow after the actual blocker is resolved.

Continue

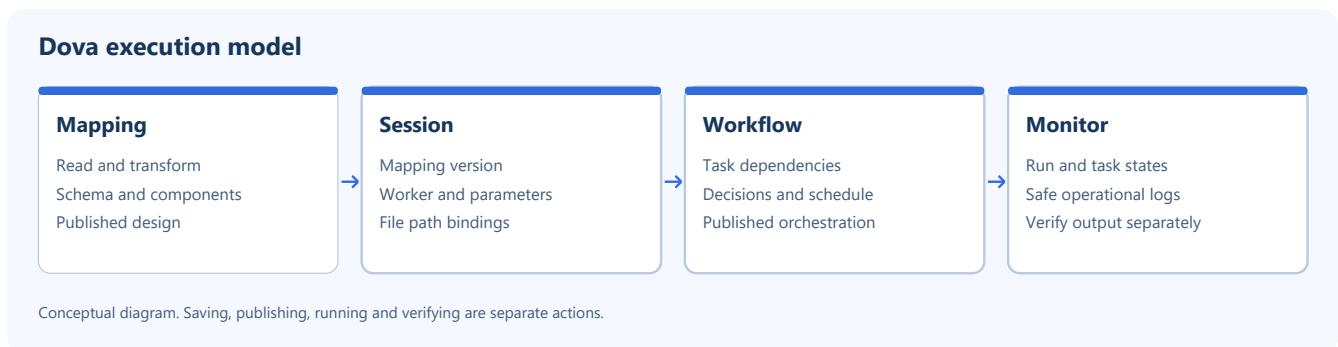
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Configuration: runtime policy, profiles, scheduler and health

Configuration separates four concerns: Dova Integration Runtime, Execution profiles, Workflow Scheduler and Runtime health. Runtime policy selects authorized capacity and affinity; profiles express execution limits; schedules express future orchestration; health reports the current condition.

Check the active scope and permission before saving. A tenant default differs from an explicit Session override. A schedule timezone differs from your profile display timezone. Document the current baseline so that a laboratory change can be restored deliberately.



Session and Workflow inherit or override runtime choices; Monitor verifies actual attempts.

Step-by-step procedure

- Inspect the four tabs.** Open Configuration and review each tab without making changes. Identify which problem you are solving before editing: default capacity, per-execution limits, a schedule, or diagnosis.
- Review runtime defaults.** In Dova Integration Runtime, inspect the tenant default worker and the affinity or selection options offered. Confirm state and compatibility in Workers before selecting a replacement default.
- Review overrides and inheritance.** Compare the default with a laboratory Session that uses Use tenant default. If an override is authorized, test it explicitly and record that distinction. Do not assume an override changes the default for other Sessions.
- Prepare an execution profile.** In Execution profiles, inspect existing policies and their units. Select or edit the appropriate authorized profile, using a concrete reason for each limit. Verify the Session/Workflow inheritance rather than copying arbitrary capacity values.
- Prepare a schedule.** In Workflow Scheduler, create or review a reusable schedule. Confirm its calendar, timezone and enabled state. Assign it from the intended published Workflow only after a successful manual laboratory run.
- Save and reload.** Review the intended changed fields, save and wait for the accepted result. Reopen the relevant record or tab to confirm persistence. If the baseline did not change, the save control may correctly remain disabled.

7. **Inspect health separately.** Open Runtime health and read current status. A saved policy does not establish that a worker is online or a scheduled attempt succeeded. Continue in Workers and Monitor for those checks.

Verify the result

Test inheritance with two synthetic Sessions: one using tenant default and one using a permitted compatible override. Run both and verify the selected capacity in execution context and Monitor. For the schedule exercise, record the explicit timezone, next planned time and the distinct resulting run identifier.

Errors and recovery

For an unavailable worker, diagnose its state rather than increasing limits. For an invalid field, correct its units and range. After a concurrent change, reload the baseline and compare before resubmitting. Disable a temporary laboratory schedule when the exercise ends; closing Configuration does not disable it.

Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#).

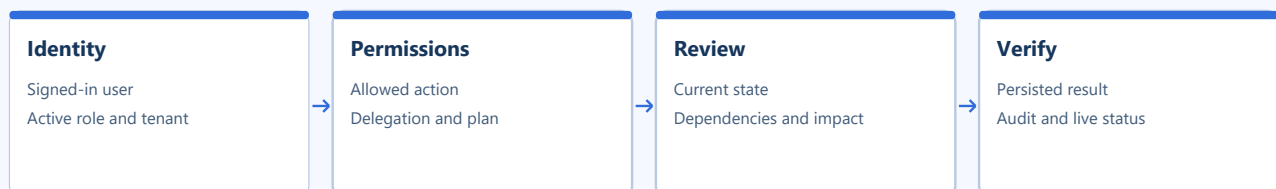
DOVA WORKSPACE · DOVA 7.0.0

Plans and subscriptions: understand offers and workspace continuity

Plans and subscriptions presents the effective commercial choices for the active tenant. Availability depends on tenant type, lifecycle, plan and authorized offers. A displayed price, a checkout order and an active subscription are different stages; one does not automatically prove the next.

Review the tenant identity before entering checkout. A public trial is designed to continue as the same operational tenant after an accepted conversion, preserving object identity. Do not recreate the tenant or move objects to simulate that continuation.

Administrator: scope before action



Conceptual review path. Administrative visibility depends on role and scope.

Commercial access depends on the active identity, tenant, permissions and plan.

Step-by-step procedure

1. **Confirm the workspace.** Open the account menu and record the active tenant. Open Plans and subscriptions and inspect the plan and lifecycle information available to your role. Do not use another tenant offer as the baseline for this workspace.
2. **Review the effective offer.** Read the offered plan version, price, currency, period, quantity and terms shown. Tenant-specific offers reference the global versioned catalog; they do not imply an independently duplicated product catalog.
3. **Inspect checkout fields.** If authorized to evaluate a purchase, review customer type, document type/number, legal name, email, country, address and payment method. Check totals and currency before advancing.
4. **Use a controlled sandbox exercise.** For learning, use supported fixtures or a separately authorized payment sandbox. Opening checkout is not permission to send a real payment confirmation. Synthetic tokens must never be submitted to a live provider.
5. **Track the order outcome.** After an actual authorized purchase, inspect the definitive order/subscription result through the supported flow. A delayed UI update is not evidence that no provider operation occurred. Resolve the existing order before repeating checkout.
6. **Verify continuity.** Confirm the active tenant identifier remains the intended one and existing Repository objects retain their identifiers. Inspect the accepted subscription and permitted capabilities independently of the payment UI message.
7. **Review later changes.** Before renewal or a new offer, inspect its validity and applicable contractual version. Historical orders retain their terms; a newly published price does not rewrite an earlier order.

Verify the result

A successful conversion is supported by an accepted commercial result, the operational tenant remaining the same and capabilities matching the applicable subscription. This manual provides a sandbox procedure; it does not certify a payment or provide a revenue forecast. Billing administrators investigate invoices, subscriptions and orders in their proper context.

Errors and recovery

If the offer is unavailable, check lifecycle, eligibility, plan and role. If amounts disagree, confirm currency, quantity and terms before submitting. For a pending order, investigate its current state rather than creating another. Do not mark an invoice paid or alter historical terms merely to make access appear correct.

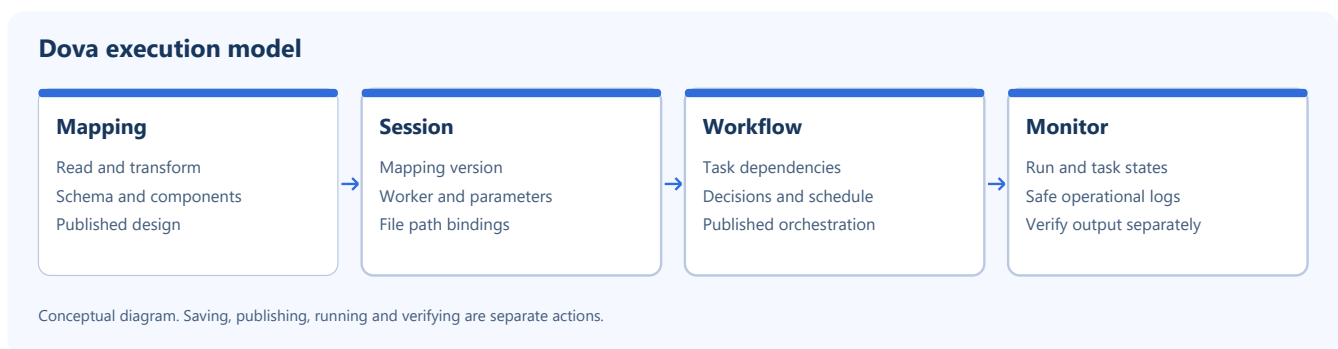
Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

Developer Tools: download the CLI and manage personal API keys

Developer Tools provides platform-specific CLI distributions, checksum and signature information, quick-start instructions, release notes and personal API key management. A package name or version label alone does not establish integrity; compare the downloaded SHA-256 with the published value.

Use an authorized laboratory environment and a personal key scoped to the required work. Keep keys out of shared commands, screenshots, repository files and incident reports. API authorization remains tied to identity, tenant and permissions.



The CLI accesses the same authorized objects and run lifecycle as the workspace.

Step-by-step procedure

- Choose the correct distribution.** Open Developer Tools or the Downloads entry in the account menu. Select your operating system and inspect the published artifact version, signature status and installation instructions. The CLI version can differ from the application version.
- Verify the download.** Download the selected artifact to an authorized local location. Calculate SHA-256 with the platform tool and compare it exactly with the published checksum before installing. Investigate a mismatch rather than bypassing verification.
- Install and inspect help.** Follow the package instructions. Run `dova --version` and the relevant command help. Confirm the executable you are using is the intended installation, especially if another version is already on PATH.
- Create a purposeful API key.** Use API Keys to create a personal laboratory key where permitted. Record its purpose and expiry according to your policy. Handle the displayed value through the protected workflow; it is not documentation evidence.
- Log into the laboratory.** Run `dova login --url` with the authorized service URL. Enter the key through the interactive prompt rather than adding it to a shared argument. Check the identity and tenant returned by login.
- Start with read-only commands.** Use status, repository list and filtered mapping/workflow/run lists. Compare their result with the corresponding UI module. A valid empty list is different from a failed authentication response.
- Close and revoke separately.** Use `dova logout` to remove local authentication state when appropriate. Revoke a temporary key in the server UI when it must stop working on every device. Logout and revocation have different scope.

Verify the result

Read-only laboratory examples:

```
dova --version
dova login --url https://dova-lab.example.invalid
dova status --output json
dova repository list --domain Training --output json
dova mapping list --name sample --limit 50 --output json
```

The `.invalid` URL is an intentional placeholder. Replace it with your authorized laboratory host. Confirm the returned identity and compare identifiers with the UI before automating writes.

Errors and recovery

For checksum mismatch, stop using the artifact and verify the distribution. For 401/403, review expiry, active key and permission; do not reinterpret the error as an empty list. A command timeout may leave server work running. Inspect the existing run before retrying. Protect the CLI profile because it can retain authentication state.

Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

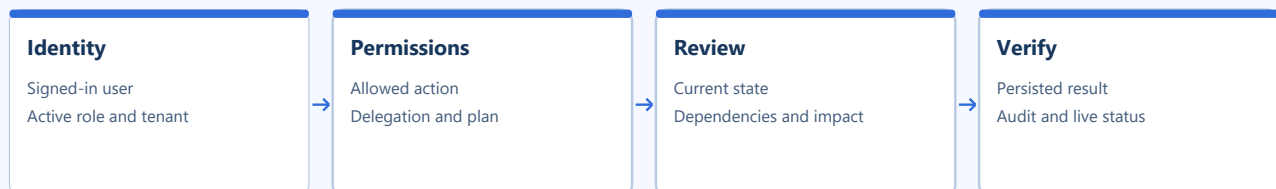
DOVA WORKSPACE · DOVA 7.0.0

Account menu: choose the workspace, role and appearance

The account menu exposes your current identity and working context and provides personal actions such as Profile, Downloads, API Keys, Documentation and Sign out. Appearance controls include light/dark mode and available palettes. A visual preference is different from an administrative permission or tenant change.

Before saving, running or removing an object, confirm the workspace and active role. Only offered, authorized contexts may be selected. A role label is not a general permission to change every module.

Administrator: scope before action



Conceptual review path. Administrative visibility depends on role and scope.

Context must be verified before an authorized action; appearance changes do not change permissions.

Step-by-step procedure

1. **Open and inspect.** Open the workspace account menu and read the signed-in identity, tenant/workspace and active role. Use keyboard navigation when desired and keep the menu open only long enough to confirm the context.
2. **Review pending editors.** Before switching context, inspect Open items for unsaved drafts. Save permitted changes or handle the discard prompt deliberately. A context switch should not be used to hide uncertainty about which tenant owns a draft.
3. **Choose an offered workspace.** Select only the authorized workspace offered by the menu. Wait for the accepted transition and read the new label. Reopen the relevant module and verify its tenant-scoped records before acting.
4. **Choose a permitted role.** If the interface offers a role/context choice, select the intended role and confirm the resulting actions. A hidden or disabled administrative action can be the expected result of the selected role.
5. **Adjust appearance.** Choose Light or Dark and the available palette for readability. Inspect text, focus outlines, status indicators and selected controls. Color alone should not be the only way you identify a status.
6. **Open personal tools.** Use Profile for timezone and personal security preferences, Downloads/Developer Tools for the CLI, API Keys for authorized personal automation and Documentation for this manual. These entries have different responsibilities.
7. **Sign out when finished.** Use Sign out at the end of a shared or temporary session. Follow the normal login and MFA flow when returning. Closing a browser tab is not the same as an explicit logout.

Verify the result

For a reversible exercise, inspect two authorized roles without changing data, compare the actions offered, then restore the original context. Try light and dark appearance and return to your preference. Confirm keyboard focus remains visible and the identity label remains correct in both themes.

Errors and recovery

If a switch fails, read the safe message and verify the current persisted context before retrying. If an object vanishes after switching, inspect tenant, location and filters rather than assuming deletion. Appearance does not grant privileges. Do not bypass MFA or tenancy checks to make a context switch succeed.

Continue

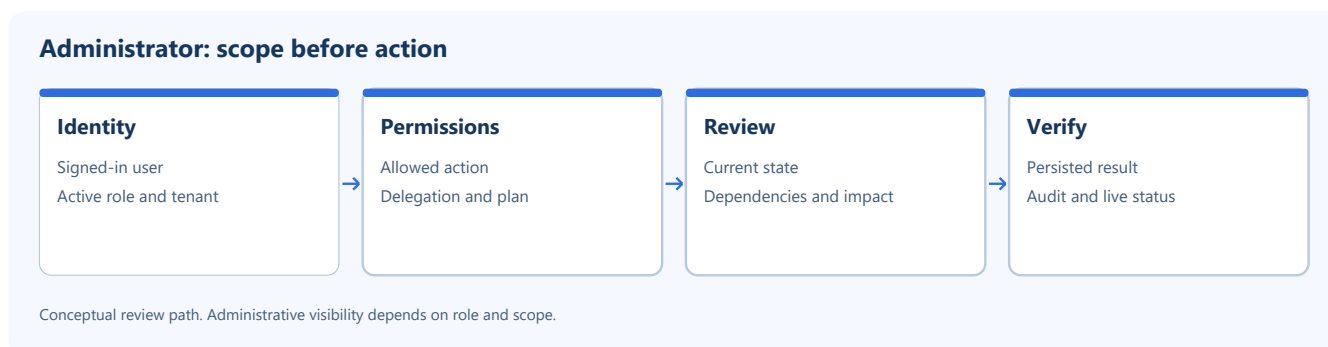
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

DOVA WORKSPACE · DOVA 7.0.0

Profile: personal preferences, password and data requests

Profile contains personal account information, preferences, security controls and privacy requests. It is separate from Administrator → Users and does not grant roles or arbitrary tenant access. Open it from the account menu and confirm the displayed identity before editing.

Use a laboratory account for reversible preference checks. Password changes and deletion requests have meaningful effects; prepare your authentication factors and choose those actions only for their intended purpose.



Profile actions apply to the signed-in identity and have their own validation and confirmation.

Step-by-step procedure

1. **Inspect account context.** Open Profile and review name, email, tenant and role. If the identity is unexpected, close the dialog and resolve the context before editing personal information.
2. **Set presentation timezone.** Choose a supported Timezone such as America/Lima, then Save preferences. Reopen Profile and confirm the saved value. This affects supported account date/time presentation; it is not a change to the Workflow schedule timezone.
3. **Update a laboratory photo.** Use Change photo with PNG, JPEG or WebP. The current flow accepts an input up to 10 MB, center-crops it to a square 256-pixel image and optimizes it before sending. Wait for accepted processing and saving.
4. **Remove a photo deliberately.** Use Remove photo when desired and confirm the accepted result. A processing error does not prove that a new image was saved or that the old one was removed.
5. **Change a password only when intended.** Provide Current password, New password and Confirm new password. The UI validates a new password length of 12-128 characters; the server can apply additional rules. Prepare normal login and MFA before submitting.
6. **Understand session effects.** The dialog states that a password change signs out browser sessions and trusted devices. Authenticate again through the authorized flow. Do not use password changes as a routine check on a shared operational account.
7. **Export or request deletion.** Download profile export only to an authorized location. Distinguish Request profile data deletion from Request account deletion. Read the email confirmation and Submit request action. Acceptance acknowledges a review request, not immediate deletion of shared tenant objects or retained audit records.

Verify the result

For acceptance, record the original timezone, change it in a laboratory account, save and reopen to confirm persistence, then restore it. A privacy-dialog inspection can be cancelled without submitting. A submitted request must be tracked as a request until the supported process confirms its outcome.

Errors and recovery

If saving fails, inspect the safe message and persisted value before retrying. If the session ends after a password change, use normal authentication rather than assuming object loss. Photo format/size failures should be corrected with another permitted image. Treat profile exports as personal information and keep them out of screenshots and shared logs.

Continue

[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

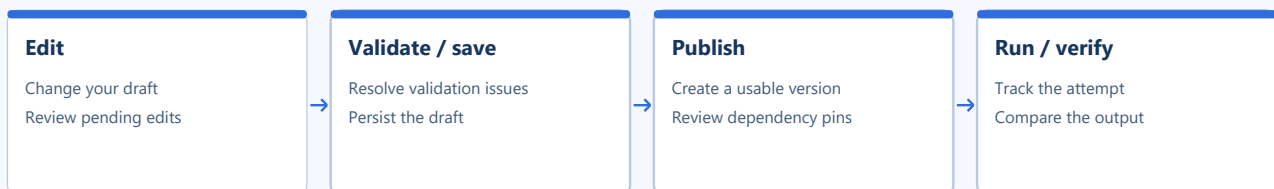
DOVA WORKSPACE · DOVA 7.0.0

Open items: work safely across Mapping, Session and Workflow editors

Open items gives you access to the Mapping, Session and Workflow editors already open in the workspace. A tab represents an editor context, not a published version or a running job. Its name, type and pending-change indication help you navigate without losing track of drafts.

Before opening several similar objects, choose distinct laboratory names and confirm their tenant/location. Use Repository or the corresponding manager to verify identity when names are ambiguous. Keep pending edits visible before switching workspace or closing tabs.

From a draft to verified execution



A saved draft does not publish a version, and a successful run does not establish data correctness.

Editor drafts, saved definitions and published versions are distinct lifecycle stages.

Step-by-step procedure

1. **Open a small related set.** Open the Mapping, Session and Workflow for one laboratory process. Inspect each tab name and type so that you can distinguish transformation design from execution binding and orchestration.
2. **Switch without assuming persistence.** Select another tab or use the supported keyboard navigation. Returning to a tab lets you continue the draft, but merely switching tabs does not save or publish it.
3. **Inspect pending-change state.** After a reversible edit, look for the pending-change indication. Save with the editor action when intended, or revert the edit. Check the result message rather than assuming the marker will disappear on its own.

4. **Use the context menu.** Open the tab context menu for the available navigation and closing actions. Read whether an action applies to one item, several items or other tabs. Avoid bulk closing when an editor still contains needed changes.
5. **Close a changed editor deliberately.** When a prompt asks about pending edits, choose the intended save/discard/cancel behavior. Cancelling should preserve your editing context. Discarding a draft change is different from deleting the saved object.
6. **Reopen and confirm.** Reopen the object from its manager or Repository after closing. Confirm the saved identifier and content. A new published version is only established by the corresponding publication action, not by reopening a tab.
7. **Review before leaving the workspace.** Inspect remaining pending items before switching tenant, signing out or ending a session. Keep the saved and published state clear so that a scheduled Workflow is not mistaken for an unsaved editor draft.

Verify the result

For a reversible exercise, make a harmless description change in a laboratory Mapping, switch to its Session, return and cancel the close prompt. Confirm the draft is retained. Then revert or save the intended change, close and reopen the object, and inspect the expected persisted result.

Errors and recovery

If the visible editor differs from your expectation, check tab type, object identifier, tenant and version. A closed tab is not proof of deletion, and a pending draft is not proof of publication. Use keyboard focus to locate tabs and menus on compact screens; investigate layout or loading errors before repeating actions.

Continue

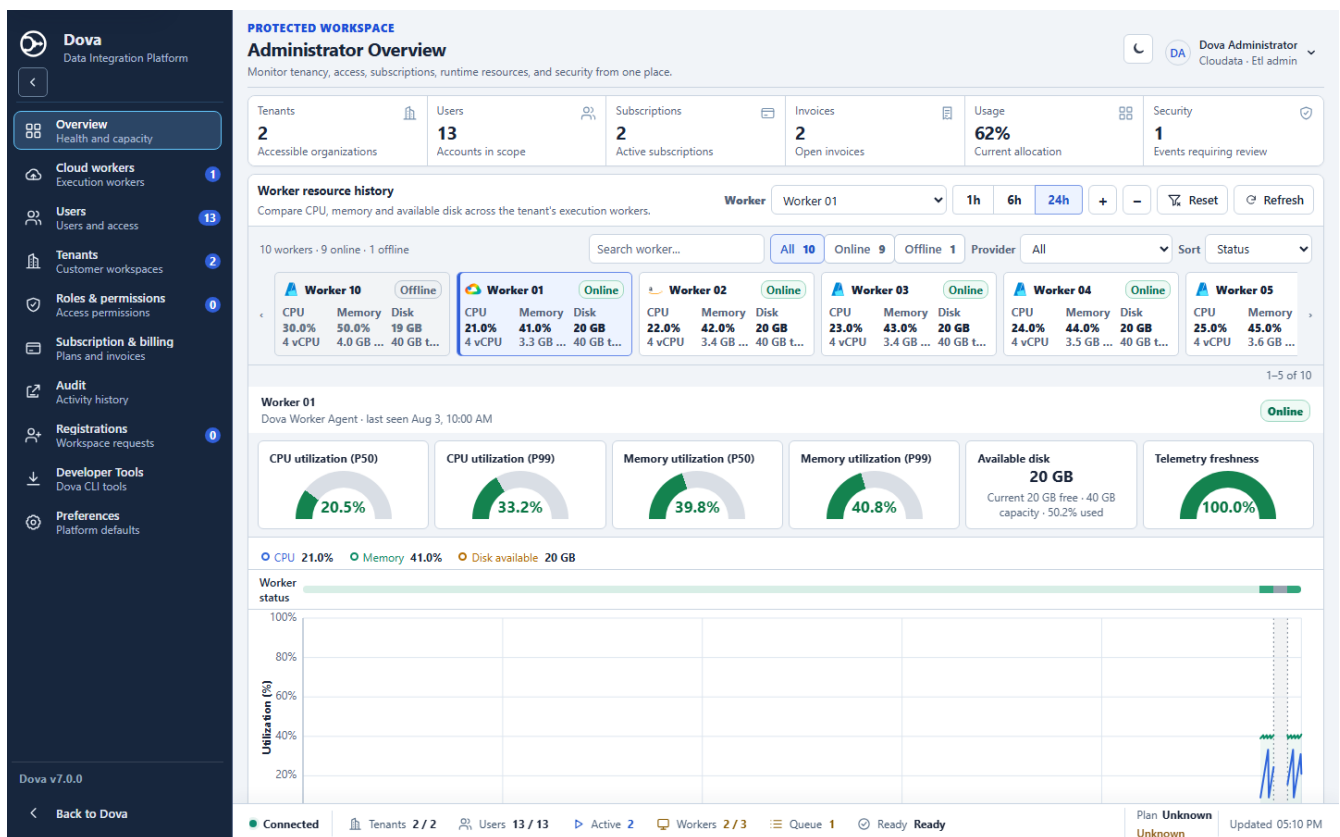
[Repository](#) · [Mappings](#) · [Sessions](#) · [Workflows](#) · [Monitor](#) · [Configuration](#).

Administrator Overview

Purpose and access

Administrator brings together workspace access, tenant capacity, cloud workers and commercial records. Open **Overview** to identify where to investigate before changing a setting. Your active workspace and role determine which information and actions are available. The account menu identifies the signed-in user and the current membership. Confirm both before interpreting counts or opening another section.

The examples in this guide use synthetic data. A worker named `worker-demo-a` and a tenant named `documentation-lab` are examples, not records to locate or create in a production environment.



Read the dashboard

The summary cards show **Tenants**, **Users**, **Subscriptions**, **Invoices**, **Usage** and **Security**. Read their labels and supporting text together. A count describes the data available to the current view; it does not establish that a payment cleared, a user can sign in, or an execution succeeded.

Worker resource history compares worker CPU, memory and available disk. Select a worker from **Worker**, or use the worker comparison cards. **Search worker...**, provider, status and sort controls help locate a worker in a larger inventory. The time controls offer **1h**, **6h** and **24h**. Zoom controls change the visible chart range; **Reset** restores it and **Refresh** requests current observations.



Other cards summarize **Users by status**, **Invoices by status** and **Usage allocation**. **Quick access** opens the corresponding administrative area when your role permits it. The operational footer provides connection and activity information while the main view is open.

Daily review procedure

1. Open Administrator and confirm the account, workspace and active role.
2. Select **Overview** and allow the view to finish loading. Distinguish a displayed zero from a loading error or missing measurements.
3. Read the summary cards and note which category requires investigation.
4. For a resource concern, select the affected worker and choose a comparable time range. Compare several samples, including the recorded coverage, rather than relying on one point.
5. Open **Users, Subscription & billing** or **Cloud workers** according to the symptom. Use the destination screen to verify the underlying record.
6. After any authorized corrective action, refresh the relevant view and confirm its resulting state.

Acceptance and recovery

A successful review identifies the workspace, affected record, observation period and next administrative action without exposing personal or sensitive information. For example, a laboratory memory increase should be checked against a synthetic execution in Dova before changing a worker profile.

If a card fails to load, retain the visible error message and retry after checking session and connectivity. If a shortcut is missing, review the active role; switching tenants does not resolve a permission denial. Old browser tabs may retain an earlier display until refreshed.

For execution outcomes, use Dova Monitor and the execution detail. For cloud infrastructure, continue with [Cloud workers](#). For navigation, appearance and account options, read [Preferences](#).

ADMINISTRATOR · DOVA 7.0.0

Cloud workers: inventory, profiles and provisioning

Purpose and prerequisites

Cloud workers manages execution infrastructure through four tabs: **Workers**, **Provisioning history**, **Connections** and **Profiles**. A worker record, a cloud VM and an enrolled execution agent are related but distinct states. Confirm the intended tenant, authorized cloud account, management permission and available capacity before provisioning infrastructure.

An approved cloud connection must be available, and its provider permissions and region policy must be satisfied. A worker profile defines requested capacity. Where the interface reports unsupported or unavailable capabilities, resolve that condition before proceeding. The optional price preview is an estimate, not a guaranteed bill.

Prepare a cloud connection

1. Open **Connections** and select **New connection**.
2. Enter **Name**, **Provider**, **Account / project ID** and **Default region**.
3. Complete provider-specific identity and networking fields using the approved cloud configuration. Use federation or temporary impersonation; this form does not accept cloud passwords, access keys or private keys.
4. Select the allowed **Authorized regions** and confirm that the default region is included.
5. Choose **Save connection**. Follow the displayed verification instructions, using **Copy steps** if needed, then select **Verify** or **Re-verify**.
6. Use **Capacity** to inspect the connection's current capacity and worker headroom. Read unavailable or unknown values as conditions requiring investigation.

Prepare a worker profile

1. Open **Profiles** and select **New profile**.
2. Enter **Name**, **Provider**, **vCPU**, **Memory (GB)** and **Disk (GB)**.
3. Choose **SKU compatibility**. **Exact resources** requires a matching provider size; **Allow next larger SKU** permits the supported larger-size selection.
4. Review **Status** and optional pricing information, then select **Save profile**.
5. Reopen the profile to verify its saved requirements before using it in provisioning.

Provision and verify a worker

1. Open **Workers** and choose **Provision worker**.
2. Confirm **Tenant** and **Worker access**. Review the distinction between **Private to this tenant** and **Public shared** before continuing.
3. Select **Cloud connection**, **Worker profile** and **Maximum concurrent slots**.
4. Set **Worker name**, **Region** and **Zone** when the provider requires them. Resolve placement errors with the supported refresh control.
5. Use **Continue** and **Back** to inspect each step and the final review. Request **Get optional quote** if a price preview is useful.
6. Select **Create worker** once, then inspect **Provisioning history** and the inventory until infrastructure and agent state are verified.

Operations and recovery

The worker row menu can expose **Edit configuration**, **Enable worker scheduling**, **Drain worker**, **Disable worker scheduling**, **Start VM** or **Stop VM** according to state and permission. Scheduling and VM power are separate controls. Read any disabled-action explanation before making a related change.

If creation times out, inspect the existing operation before submitting another request. If a worker is online but unavailable for a job, check tenant eligibility, required capabilities and slot capacity. Validate a synthetic execution and its Monitor result before claiming the worker is operational for the intended workload.

Continue with [Worker installation](#), [Migration and removal](#) and [Overview](#).

Users and account access

Purpose and prerequisites

Use **Users** to inspect identities, tenant memberships, roles and access status. Viewing accounts and managing them are separate permissions. An administrator who can open the table may still be unable to create users, change roles or generate an activation link. Confirm the current workspace and active role before editing an account.

The **Status** and **Access** columns answer different questions. Status describes the identity, while Access describes its sign-in readiness. **Last login** records an observed login or displays **Never**. A presence indicator is supporting information; it does not grant access or prove an execution completed.

Find and inspect a user

1. Open **Users** from the sidebar.
2. Use **Search users**, column filters or the mobile **Filters** control to narrow the table.
3. Select **Columns** to adjust the visible fields, or **Clear filters** to restore the unfiltered view.
4. Open **View details** for the required identity and confirm its tenant, email, roles, status and access state.
5. Use **Export CSV** only when an export is needed and you can handle its account information appropriately.

Create or edit an account

1. Select **New user**, or **Edit user** on an existing row.
2. Enter the user's **Name** and **Email**, then select the intended **Tenant**.
3. Choose **Active**, **Inactive**, **Suspended** or **Disabled** according to the account's intended availability.
4. Use **Select roles** to assign only the roles required for that tenant. Search the available roles when the list is long.
5. If membership management is available, use **Add tenant** to add a permitted membership. Select its membership before using **Set default** or **Remove**.
6. For a new account, choose the displayed first-access delivery option. **Copy activation link** reveals a one-time link after creation; **Send reset email** sends first-access instructions to the registered address.
7. Submit the form once and wait for completion. Reopen the record to verify the saved tenant, roles and status.

Manage first access

For an eligible existing account, open **Manage access** and review the identity and current access state. Select the intended option under **Available actions**. Local activation labels vary by state: **Reset access**, **Resend activation**, **Send new activation** or **Create activation**. Eligible accounts can also offer **Convert to local access**, **Restore SSO access** or **Revoke activation**. Read the displayed impact before proceeding.

Delivery method appears when creating a local activation or converting to local access. Choose **Copy activation link** or **Send activation email** for those actions. Restoration or revocation does not require a delivery selection. The submit button uses the selected action's name; confirm that action once and verify the resulting Access state. A displayed link



should be shared through an approved private channel. Do not place it in a screenshot, support note or shared document. Close the result with **Done** after completing the permitted delivery step.

Where available, **View as user (read only)** provides an inspection context. Confirm the read-only indication and exit that context before returning to normal administration.

Acceptance and error recovery

In a synthetic exercise, create `Documentation` operator using a controlled laboratory email address, assign a limited role and verify the final row. Successful account creation does not establish that the user completed first access; verify Access separately.

If **Manage access** is unavailable, check account status and whether you are inspecting your own identity. If saving fails, keep the form's message, correct the identified field and retry after checking the latest record. Review selected identities carefully before **Delete selected** and its confirmation.

Continue with [Roles and permissions](#), [Tenants](#) and [Authentication](#).

ADMINISTRATOR · DOVA 7.0.0

Tenants and workspace capacity

Purpose and access

A tenant is the operational workspace associated with users, data assets and commercial settings. **Tenants** provides a directory of workspaces and their capacity. Confirm that your role permits tenant viewing or management, as appropriate. Creating a tenant and creating a user are separate operations; one should not be assumed to complete the other.

The inventory includes **Tenant**, **Type**, **Parent**, **Plan**, **Usage / limits**, **Capacity** and **Status**. A displayed plan name is a commercial or administrative attribute. Check the applicable subscription and entitlement details before concluding what a tenant can use.

Inspect a tenant

1. Open **Tenants** and search for the workspace by its displayed name or identifier.
2. Use column filters to separate organization, individual or reseller tenants, or to inspect a status or capacity state.
3. Read **Usage / limits** with its units. Seats, monthly jobs and storage measure different resources and should not be added together.
4. Open **View details** to confirm the tenant's identity, parent relationship, plan, status and configured limits.
5. Use **Columns**, **Rows**, **Previous** and **Next** as needed. Wide data remains inside the table's scrolling area.

Create or maintain a tenant

1. Select **New tenant**, or **Edit tenant** for an existing row.
2. Set **Name** and the appropriate **Type**: **Organization**, **Individual** or **Reseller**.

3. Select **Parent reseller** when a supported reseller relationship applies. **Direct customer** indicates that no parent reseller is selected.
4. Choose **Status**, **Plan**, **Seats**, **Monthly jobs** and **Storage MB** using the values authorized for this workspace.
5. Review the complete form, submit once and wait for the result.
6. Refresh or reopen the tenant and confirm the resulting status, relationship and capacity values.

Use **Suspend tenant** and **Reactivate tenant** only after reviewing the affected workspace and expected availability. These are administrative status changes, not a migration of the tenant's objects. Inspect linked users and operational obligations before using bulk deletion.

Trials and lifecycle

A **Trial** status should be interpreted with the subscription's dates and lifecycle information. To inspect or extend an eligible trial, open **Subscription & billing**, select the same tenant and use the supported **Extend trial** flow. Do not recreate a workspace to change its commercial status.

When reviewing a trial-to-paid transition, verify that the expected workspace remains selected and that subscription and entitlement information reflect the intended outcome. Commercial activity and infrastructure readiness are separate checks.

Acceptance and recovery

For a laboratory tenant named `documentation-lab`, verify the visible usage, limit units, plan and status after saving. A successful exercise preserves the intended workspace identity and confirms that unrelated memberships remain unchanged.

If a save or capacity read fails, inspect the visible message and reload the latest record before retrying. A warning or critical capacity indicator calls for investigation of the individual resource and applicable policy. It does not automatically justify increasing every limit.

Read [Subscription and billing](#) for commercial lifecycle, [Preferences](#) for limit policies and [Cleanup and tenant lifecycle](#) before destructive maintenance.

ADMINISTRATOR · DOVA 7.0.0

Roles and permissions

Understand the access model

Roles & permissions separates assignable roles from the permission catalog. A permission describes an authorized operation; a role groups permissions for an intended responsibility. The role's scope and tenant determine where that responsibility applies. Access to the screen does not itself authorize editing every role or assigning a role to every user.

Before changing a role, identify its intended users, the tenant concerned and the operations those users actually need. Review a current account in **Users** to understand its existing assignments. Use a laboratory role and identity to validate a proposed access arrangement before applying it more widely.

Inspect existing roles and permissions

1. Open **Roles & permissions** and select the **Roles** tab.
2. Search for a role, use table filters and open **View details**.
3. Review the role's key, label, scope, description and assignment information.
4. Select **Permissions** to inspect the permission catalog. Search for the relevant capability and open its details.
5. Use **Columns** and **Export CSV** where available to support a permitted review. Keep exported account and access information in an approved location.

The permission catalog is an inspection reference. Do not infer that a displayed permission is currently granted to the signed-in user or to the user being reviewed. Check the role assignment and its actual scope.

Create or edit a tenant role

1. Select **New role**, or **Edit role** on an editable role.
2. Enter a stable **Key** and a readable **Label**. Choose names that describe the responsibility without implying broader access.
3. Confirm **Scope** and **Tenant** before selecting permissions.
4. Enter a **Description** explaining when the role should be used.
5. Review the available permission choices and select the smallest set needed for the intended workflow.
6. Submit the form and verify the saved role in the table.
7. Open **Users**, assign the role to a synthetic laboratory account and verify both a permitted action and a relevant denied action.

System or immutable roles may be viewable without exposing edit or delete actions. Do not attempt to duplicate an immutable role merely to bypass that restriction. If a new responsibility is needed, create an appropriate editable tenant role through the supported form.

Remove a role safely

Delete role is available only for an eligible editable role. A role with assignments cannot be deleted through that action. First inspect the affected identities and decide which supported reassignment is appropriate. Reopen the role after those changes and verify its current assignment count before confirming deletion.

Acceptance and recovery

Successful access validation demonstrates the expected capability within the selected tenant and a denial outside its authorized scope. A button appearing in a menu is not sufficient evidence of authorization.

If a save is denied, verify the active role and workspace rather than changing tenants to evade the denial. If deletion is disabled, inspect assignments and immutability. After a change, refresh the relevant accounts and review [Audit](#) where appropriate.

Related guides: [Users](#), [Tenants](#) and [Authentication](#).



Subscription, billing, licenses and checkout

Purpose and prerequisites

Subscription & billing contains **Subscription**, **Licenses**, **Commerce**, **Global catalog** and **Tenant offers**. Select the intended **Tenant** before inspecting or changing its commercial information. Viewing, operating billing and managing the global catalog require their respective permissions; global catalog changes require the authorized platform role.

A subscription, invoice, license and commerce order are separate records. A manual invoice status update does not independently establish payment settlement. Issuing a license does not prove an installation activated it. Use each record's supported detail and lifecycle to verify the intended result.

Maintain subscription and billing profile

1. Select the tenant and open **Subscription**.
2. Read the subscription summary and choose **Manage** when an authorized update is required.
3. Select **Plan** and the appropriate **Status**, then choose **Save subscription**.
4. Use **Edit profile** to review legal name, billing email, tax identification, country and address.
5. Save the profile and reload the tenant's subscription information to confirm persistence.

For an eligible trial, **Extend trial** opens a separate review. Inspect the current end, choose the displayed extension option and enter the required reason. Verify the new end and preserved tenant identity after acceptance. Detailed eligibility and recovery are explained in [Tenant lifecycle](#).

Create or maintain an invoice

1. Select **New invoice** in the invoice inventory.
2. Enter **Number**, **Amount**, **Currency**, **Due date**, **Status** and **Description** as required.
3. Review amount and currency together, then choose **Create invoice**.
4. Reopen the invoice's details and verify the saved record.
5. When a lifecycle update is authorized, use its status dialog and **Update status**. Select a status reflecting the verified administrative process.

The invoice editor offers USD, PEN and EUR. The set of invoice currencies should not be assumed to extend the separate Commerce catalog's supported currencies. Use search, filters and **Export CSV** only for the required billing review.

Issue, activate and revoke licenses

In **Licenses**, choose **Issue license** and review **Reference**, **Paid invoice**, **Maximum activations** and **Expires on**. Verify the invoice association and validity before issuing. Protect a displayed key using approved private storage.

Activate opens the supported license activation flow with **License key** and **Runtime / installation ID**. Verify the resulting activation separately. **Revoke license** requires its confirmation and does not imply a financial refund.



Commerce and acceptance

In **Commerce**, inspect **Plans available** and **Order summary**. **Continue to checkout** or **New checkout** opens the order form. Review price, quantity, payment method, customer type, fiscal document, legal details and billing address. Use the supported browser payment flow; never enter card data into a documentation example.

A synthetic exercise should use fixtures or an authorized sandbox and stop before any real charge. Confirm the tenant, order terms, subscription and invoice independently. If a checkout response is delayed, inspect recent activity before submitting another order because the provider may already have accepted an operation.

Read [Global catalog and offers](#) for versioned terms and [Audit](#) for administrative evidence.

ADMINISTRATOR · DOVA 7.0.0

Audit trail and administrative evidence

Purpose and access

Audit provides a read-oriented record of administrative activity available to the current workspace and role. Use it to inspect an event's action, actor, timing and associated record, then correlate those details with the relevant administrative screen. The visible audit trail is one source of evidence; it does not independently prove payment settlement, provider completion or execution success.

Before investigating, define the question you want to answer. For example: which permitted administrator changed a laboratory user's role, when was a trial extension recorded, or what happened around a worker configuration attempt? Keep the investigation within the workspace and information your role authorizes.

Locate and inspect events

1. Confirm the active account, workspace and role in the account menu.
2. Open **Audit** and wait for the inventory to load.
3. Use **Search audit events** to narrow the records by the displayed information relevant to your question.
4. Apply available column filters, checking the selected period and status where applicable.
5. Open **View details** for the event and inspect its visible action, actor, timestamp and subject.
6. Open the associated screen, such as **Users**, **Tenants** or **Cloud workers**, and confirm the current record state.
7. Refresh the audit view if you need to inspect a recent authorized action.

Columns changes presentation. **Clear filters** restores the broader visible inventory. **Rows**, **Previous** and **Next** control pagination. On mobile, use **Filters** and the existing table viewport rather than assuming hidden columns are absent from the record.

Build a useful evidence note

Record the administrative question, workspace scope, displayed event time, action, subject and observed result. Distinguish the recorded event from a conclusion that you infer after checking another screen. Preserve the relevant time context when comparing timestamps from different systems.



Use **Export CSV** only when the investigation requires an authorized export. Store the result in an approved location and share the minimum needed information. Exports may contain identifiers or account details even when the application avoids showing secrets.

Synthetic acceptance exercise

In an authorized laboratory, edit the description of an editable tenant role. Confirm the saved role in **Roles & permissions**, refresh **Audit** and locate the corresponding activity if recorded and visible to your role. The exercise succeeds when the role state and audit observation describe the same intended action within the same tenant.

If the event is absent, inspect filters, paging, workspace selection and the delay between the action and refresh. Do not invent an audit event or conclude that an action failed solely because the currently displayed page does not contain it.

Errors and recovery

A loading failure should be reported using the visible message and a bounded description of the scope and time. Retry after checking connectivity and session validity. A permission denial requires an authorized access review; switching tenants is not an evidence-preserving workaround.

For the action itself, return to [Users](#), [Roles and permissions](#), [Subscription and billing](#) or [Cloud workers](#). The current record and its supported detail view remain necessary parts of the investigation.

ADMINISTRATOR · DOVA 7.0.0

Customer registration review

Purpose and access

Registrations presents self-service customer requests for administrative review. The screen supports inspection and a decision on an eligible pending request. Viewing a registration and deciding it are separate capabilities. Confirm the current workspace and your review permission before acting on an applicant.

An approval records the review decision. Tenant and user provisioning remain separate. Do not tell an applicant that a workspace or account is ready solely because the registration changed to approved. Review the actual tenant and user records through their supported administrative screens.

Inspect an application

1. Open **Registrations** from the sidebar.
2. Use **Search registrations** or the column filters to locate the relevant request. On a narrow screen, open **Filters** as needed.
3. Open **View details** and confirm the applicant's name, email, organization, type and current status.
4. Check the received date and any existing reviewed date, reviewer and review reason.
5. If the row is stale or another administrator may have acted, select **Refresh** before deciding.

Export only the permitted information required for the review. Use **Columns**, **Rows**, **Previous** and **Next** to manage a large list without changing its underlying records. Search and filtering do not themselves approve or reject an application.

Approve a pending request

1. Open **Approve registration** for the eligible pending row.
2. Check the applicant shown in the confirmation dialog.
3. Enter **Reason (optional)** when your review process benefits from a short explanation.
4. Select the confirmation acknowledging the applicant and decision displayed above it.
5. Choose **Approve registration** once and wait for the response.
6. Confirm the saved status and review information in the refreshed list or details.

Reject a pending request

1. Open **Reject registration** for the intended pending row.
2. Check the applicant and enter **Reason (required)**.
3. Keep the reason between 1 and 1,000 characters and avoid personal information beyond what the review needs.
4. Select the confirmation and choose **Reject registration** once.
5. Verify the resulting status, reviewer and explanation after completion.

Acceptance and recovery

Use a synthetic request from `documentation-lab` to practice inspection without contacting a real applicant. A successful review produces one recorded decision for the intended request and preserves a traceable reason where required. It does not create a tenant, assign a role or generate first-access instructions.

If a decision fails because the record changed, close the stale review and refresh the list. Inspect the current status before reopening an action. If the response cannot be verified, use the same decision's supported retry rather than changing the decision or repeatedly submitting new requests.

Review controls disappear when the request is no longer pending or your role lacks management access. Treat that as a state or permission condition. Do not attempt to bypass it through another tenant or a handcrafted request.

Continue with [Tenants](#), [Users](#) and [Audit](#) to verify the separate provisioning and access steps.

ADMINISTRATOR · DOVA 7.0.0

Developer Tools, Dova CLI and API keys

Purpose and prerequisites

Developer Tools provides the published Dova CLI catalog, platform guidance and API key tools. It shares its presentation with the Dova workspace. Opening this view or downloading a distribution does not grant access to a tenant or authorize an administrative command.

Choose the operating system of the destination machine, which may differ from the computer running your browser. Before using a downloaded tool, identify the intended laboratory workspace, the permitted operation and the credential scope required for it. Keep operational credentials outside repositories, screenshots and documentation.

Download a published distribution

1. Open **Developer Tools** and allow its release catalog to load.
2. Inspect the available platform entries and select the destination platform.
3. Confirm that the main download and the **Quick start** guidance correspond to that selected distribution.
4. Read the release information, availability, checksum and signing information exposed for the artifact.
5. Download through the published link and verify the file against the checksum for that exact artifact.
6. Follow the distribution's current **Quick start** steps in an authorized environment.

Availability, integrity and signing are different properties. A successful download proves neither that the file's integrity matches nor that it is signed. Describe each property according to the published metadata, and do not present an unsigned artifact as signed.

Configure permitted access

Use the supported API key flow only when your role permits it. Check the intended tenant and the available scope before creating or selecting a key. If a one-time credential is displayed, transfer it to approved secure storage without including it in shared notes or screen captures.

Copy commands from the current distribution's **Quick start** or its linked CLI reference. This guide intentionally does not invent subcommands, package names or flags. Document the version and required configuration for an automation, but substitute descriptive placeholders for credential values.

Acceptance exercise

In a synthetic catalog, select a Linux entry and confirm that its download, metadata and instructions agree. Switch to a Windows entry and repeat. This checks navigation and catalog consistency without installing a binary or creating a real key.

For a functional laboratory exercise, download the verified distribution and perform one explicitly permitted read operation against a synthetic resource. Record three separate outcomes: the distribution was available, its integrity matched, and the authenticated operation was authorized. Passing one outcome does not establish the others.

Errors and recovery

If no distribution is available, inspect the release catalog's message rather than guessing a URL or filename. If the checksum differs, stop using that download and obtain the correct artifact through the supported process.

For an authentication or authorization error, inspect the selected workspace, credential validity and allowed scope. Creating a stronger credential is not the first diagnostic step. Revoke or retire keys through their supported lifecycle when they are no longer needed.

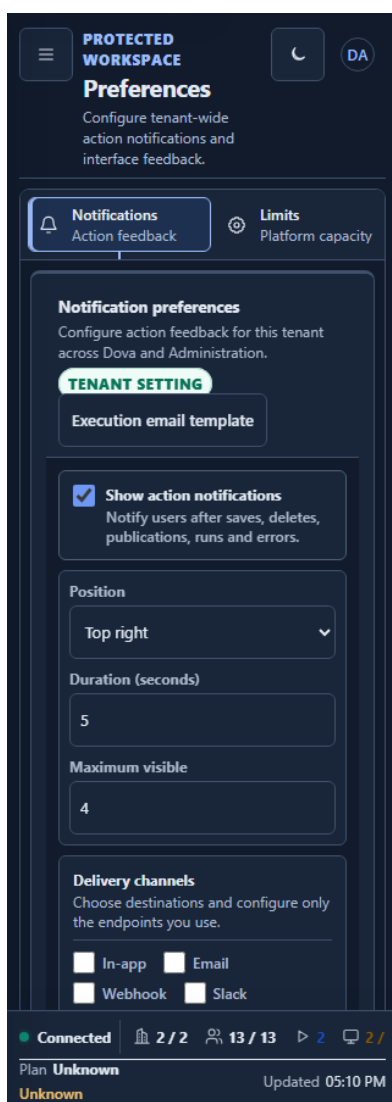
Keep commands complete and copyable in downloaded documentation. Long platform tables can scroll within their own viewport. Continue with [Worker installation](#) for execution capacity and [Preferences](#) for the account and appearance controls.

Preferences, appearance and limit policies

Find the correct scope

Preferences groups administrative settings while preserving the distinction between tenant settings and platform settings. The visible sections depend on your active role. Opening Preferences does not imply permission to edit authentication, email transport or every tenant's limits. Read the scope badge, selected tenant and enabled actions before changing a value.

The V7 application uses compact spacing, readable labels and consistent controls in day and night themes. On mobile, section navigation and toolbars remain available through wrapping or their local scrolling areas. Use the account menu for appearance and account options; use Preferences for the administrative policy concerned.



Account and appearance procedure

1. Open the account trigger in the top bar and confirm the identity and workspace membership shown.
2. Select another permitted membership or role only when it is the intended context for your work.

3. Adjust theme and palette using the appearance controls. Check the resulting selected values and status feedback.
4. Use **Back to workspace**, **Downloads**, **API Keys** or **Documentation** for the related destination when available.
5. Close the menu with Escape to return focus to its trigger. Use **Sign out** to end the account session when your work is complete.

Review tenant limit defaults

1. Open **Preferences** and select **Limits**.
2. Choose **Configuration scope**. **Platform defaults** establishes inherited defaults; a tenant selection addresses that tenant's permitted overrides.
3. Read the scope summary before editing. Distinguish inherited values from explicit overrides.
4. Select **Workers and capacity**, **Executions and objects**, **Retention**, **API and budget** or **Security and access**.
5. Read each limit's label, description, current value and unit. Workers, slots, memory, executions and retention periods are different resources.
6. Edit only the authorized values, remaining within the displayed validation range. Where available, use the inheritance control to restore a tenant's inherited value.
7. Select **Save changes** after the form records a real change. Reload the same scope and verify the saved policy.

The save control remains disabled when nothing changed. Reverting an edit to the loaded value should leave no pending update. If you change scope with unsaved edits, review **Keep editing** and **Discard changes** instead of assuming the previous values were saved.

Acceptance and recovery

For a synthetic tenant, change one permitted capacity value, save it and reload that tenant's policy. Verify the value, unit and inheritance state while confirming that platform defaults and another tenant's policy remain unchanged.

If a field is unavailable, inspect scope and permission before changing roles. If a save fails, retain the visible message and pending edit, refresh the relevant baseline when instructed, then retry the reviewed value. A limit change does not install infrastructure or guarantee that a workload is eligible to run.

Read [Notifications and email](#), [Authentication](#) and [Cleanup](#) for those specific preference sections.

ADMINISTRATOR · DOVA 7.0.0

External authentication, SSO and MFA

Purpose and authority

Preferences → **Authentication** configures supported external sign-in providers. Management of this global configuration requires the authorized Platform Administrator role. A role that can inspect ordinary Preferences does not automatically have authority to save provider settings, replace secrets or test global authentication.

SSO establishes identity, role assignments control authorization, and MFA completes the account's required verification. Enabling an external provider does not grant administrative access or remove MFA. Prepare an authorized recovery route and a laboratory identity before changing an integration.



Configure an external provider

1. Open **Preferences** → **Authentication** and select the intended provider's card.
2. Prepare an approved application in the provider's administration system.
3. Copy Dova's read-only **Authorized callback URL** and register that exact address with the provider.
4. Enter **Client ID** and the provider-specific configuration. Microsoft uses **Tenant ID**; Okta and Auth0 use **Issuer URL**; GitLab can use its supported issuer for the intended service.
5. Use **Replace client secret** only when replacing the stored secret is necessary. Leaving it blank preserves the current value according to the form's guidance.
6. Set the provider's enabled state and choose its corresponding save action, such as **Save Google** or **Save Microsoft**.
7. Confirm persistence, then use **Test discovery** or **Test provider** as displayed. The test uses the saved configuration, not unsaved fields.
8. Perform a fresh sign-in with the authorized laboratory identity and verify the final workspace and role.

The supported cards are Google, Microsoft, Okta, Auth0, GitHub and GitLab. The GitHub card states that a verified primary email must match an active Dova user. Follow each provider's own identity rules rather than applying one provider's onboarding behavior to all six.

Registration policy

Allow self-registration has a separate **Save registration policy** action. Review the current onboarding policy and the resulting workspace identity when testing it. Enabling this switch does not convert existing accounts, renew trials or assign platform roles. An access refusal caused by account lifecycle should be investigated through the tenant and subscription information.

Complete MFA

At sign-in, follow the **Verification or recovery code** flow. For first-time factor setup, register the displayed QR or setup key privately in the intended authenticator and submit its current code. Store one-use recovery codes securely. Do not capture the setup key, QR, recovery codes or verification codes for documentation.

Trust this device is an explicit account choice. It does not disable MFA globally. If you selected the wrong identity, use **Use another account**. If the factor is unavailable, follow the authorized recovery process.

Acceptance and error recovery

Test an authorized identity, an unauthorized identity and an incorrect verification code in a controlled environment. Successful configuration admits only the authorized account into the expected workspace. A provider discovery result alone does not prove complete login success or role authorization.

For failures, inspect callback accuracy, issuer or tenant, enabled state and account correspondence. Separate provider connectivity, MFA failure and permission denial. Avoid repeated secret replacement without identifying the failed stage.

Related guides: [Users, Roles and permissions](#) and [Tenant lifecycle](#).

Global catalog and tenant offers

Understand the commercial model

Subscription & billing contains **Global catalog** and **Tenant offers**. The catalog is shared across tenants, while an offer applies to its selected tenant. Products, plans, published plan versions and prices have different responsibilities. Confirm catalog-management authority and the selected tenant before changing commercial records.

Published plan terms remain associated with their version. A new set of terms begins as a draft, and an existing used price may require **New price** rather than editing its historical value. Review the record's status and available row actions before deciding how to make a change.

Create or revise the catalog

1. Open **Subscription & billing** → **Global catalog**.
2. Use search, filters, **Columns** and **View** to inspect the existing products, plans, versions and prices.
3. Create a product or plan using **New product** or **New plan** where permitted. Enter the form's identity and description fields.
4. On an eligible plan or version, choose **New draft** to prepare revised terms.
5. Review the draft's limits and entitlements. The editor's JSON fields require the indicated object format and supported value types.
6. Save and inspect the draft. Use **Publish** and **Publish version** only after reviewing the resulting terms.
7. Use **New price** to create a price for the intended **Published plan version**.
8. Set **Amount (minor units)**, **Currency**, **Billing interval**, **Tax behavior** and **Tax rate (basis points)**, then verify the saved price.

One hundred minor units equals one currency unit. One hundred basis points equals one percent. These units should be checked before saving an amount, discount or tax value. Review the final order summary rather than interpreting an amount alone as a tax-inclusive customer total.

Configure a tenant offer

1. Select the intended **Tenant** and open **Tenant offers**.
2. Read the scope description and choose **Reload offers** if another administrator may have made a change.
3. Choose **New offer**, or edit an eligible existing offer.
4. Select **Published price** and confirm its read-only currency.
5. Choose **Offer type**: a discount in basis points or a negotiated amount in minor units.
6. Set **Valid from (UTC)**, optional **Valid until (UTC)** and **Status**.
7. Save and reload the same tenant's offers. Verify that its eligible price is reflected in the applicable commerce view.

When no eligible offer exists, public prices apply. Creating an offer does not duplicate the global catalog or rewrite the terms of an earlier order.



Acceptance and recovery

In a laboratory catalog, create a draft change and a synthetic offer for one tenant. Verify that another tenant continues to receive the applicable public price and that published historical terms remain inspectable.

If an editor reports that a record changed, preserve the entries, close it, reload the catalog or offers and reopen the latest record for review. If access or tenant changed, reopen in the correct authorized scope. Use **Retire** only after inspecting the record and its lifecycle implications.

Continue with [Subscription and billing](#), [Tenants](#) and [Audit](#).

ADMINISTRATOR · DOVA 7.0.0

Execution email templates, SMTP and notifications

Separate content, transport and policy

Execution email content, SMTP delivery and notification policy are three configurations. **Execution email template** defines the message, **Email delivery** defines the platform transport, and **Notifications** chooses events, channels and routing. A valid preview does not prove message delivery, and a successful SMTP connection does not prove an event qualifies for a notification.

Check the selected tenant and the relevant view or management authority. Platform SMTP management requires the authorized Platform Administrator role. Use a controlled test recipient and synthetic execution data for validation.

Edit an execution email template

1. Open **Preferences** → **Notifications** → **Execution email template**.
2. Confirm the template's displayed scope and inheritance before editing.
3. Select the supported execution outcomes: succeeded, failed or cancelled. At least one outcome must be selected.
4. Enter a meaningful subject and optional preheader. Keep each on one line.
5. Compose both HTML and plain-text content. The text version is required and should communicate the same operational result as the HTML version.
6. Insert variables from the editor's supported catalog, using its documented percent-delimited syntax.
7. Preview the message at narrow width and inspect headings, contrast and links.
8. Save, reload and verify the content before testing actual delivery.

For a synthetic example, %WORKFLOW_NAME%, %STATUS%, %RUN_ID%, %OUTPUT_ROWS% and %EXECUTION_URL% can describe an execution and link to its authorized detail. Unknown variables are rejected. Use the available catalog instead of inventing placeholder syntax. Keep logs, credentials and sensitive data out of message bodies.

Configure SMTP delivery

1. Open **Preferences** → **Email delivery**.

2. Review **SMTP host**, **Port**, **Encryption**, **Username**, **From name**, **From email** and **Reply-to email**.
3. Enter **Replace password** only when replacing the stored credential. Blank preserves the current password according to the form.
4. Set **Test recipient email** to a mailbox you are authorized to use.
5. Save the reviewed configuration through **Save email**, then use **Test SMTP** as the supported form permits.
6. Choose **Send test email** and verify actual receipt, subject, text and link readability.

Configure notification policy

In **Notifications**, review **Show action notifications**, **Position**, **Duration (seconds)** and **Maximum visible**. Select the intended **Delivery channels**, endpoints, **Events and severity**, recipient roles, thresholds, deduplication, escalation and digest settings. For quiet hours, verify the selected timezone and both time boundaries.

Use **Preview**, **Test channels** and **Save preferences** for their distinct purposes. Inspect **Delivery history** after a controlled test. Sent, failed, suppressed and acknowledged outcomes should be interpreted separately; suppression by policy is not automatically an SMTP failure.

Acceptance and error recovery

A complete laboratory check verifies template persistence, correct substitution, the configured policy and actual receipt at the controlled mailbox. Record each outcome separately. Never include secret endpoint values or SMTP credentials in screenshots.

If testing fails, retain the safe displayed message and inspect one layer at a time: content, connection and credentials, then policy. Review event selection, quiet hours and deduplication before repeating a send.

Related guides: [Preferences](#), [Users](#) and [Audit](#).

ADMINISTRATOR · DOVA 7.0.0

Install workers on Linux, Windows Server and cloud

Choose the installation path

There are two supported paths: install Dova Worker Agent on an existing authorized server, or provision a cloud VM through **Administrator** → **Cloud workers**. Neither path automatically assigns every existing Session to the worker. Before starting, confirm tenant, operating system, approved capacity, outbound connectivity and responsibility for the infrastructure.

Download the currently published package through the supported catalog and verify its integrity. Package versions can differ from the Dova application version. Use the instructions bundled with the selected package and perform the initial exercise on an authorized laboratory server.

Linux prerequisites and installation

The documented prerequisites include Python 3.10 or later, a virtual environment, systemd, util-linux with `flock`, administrator access and outbound HTTPS to Dova and the approved package registry or mirror. The installer checks requirements; do not assume it installs a different Python or changes the operating system.

1. Extract the verified package on the laboratory server.
2. Run the prerequisite check from the package directory.
3. Complete installation only after resolving the reported requirements.

```
sudo sh execution-worker/install.sh --check
sudo sh execution-worker/install.sh
```

In Dova, use **Register worker**, select the intended system, enter the worker name and permitted concurrency, and generate the enrollment instructions. Run the generated command on the correct server. Supply the enrollment token through the supported hidden prompt, never through documentation, shell history or a shared screenshot.

After enrollment succeeds, enable and inspect the service:

```
sudo systemctl enable --now dovaetl-execution-worker
sudo systemctl status dovaetl-execution-worker
```

Windows Server prerequisites and installation

Use supported 64-bit Windows Server, PowerShell 5.1 or later and a machine-wide 64-bit Python 3.10+ installation accessible to the service account. A Windows Store alias or a Python installed only for the interactive user is unsuitable.

1. Extract the verified Windows package.
2. Open an elevated PowerShell in its directory.
3. Follow the bundled installer instructions and identify the actual machine-wide Python executable.
4. Generate enrollment for **Windows Server** and complete its hidden-prompt flow.
5. Start and inspect the installed service after successful enrollment.

```
Set-Service -Name DovaWorkerAgent -StartupType Automatic
Start-Service -Name DovaWorkerAgent
Get-Service -Name DovaWorkerAgent
```

The service uses a dedicated virtual account. Windows process limits do not make arbitrary untrusted code safe, and Windows workers remain private-only under the supported sharing contract.

Cloud installation and acceptance

Prepare **Connections** and **Profiles**, then use **Provision worker** as described in [Cloud workers](#). Observe **Provisioning history** until you distinguish VM creation from completed agent enrollment.

Verify service state, recent heartbeat, displayed resources and tenant eligibility. Assign a synthetic Session to the worker, execute a small controlled file transformation and inspect its Monitor result. Concurrency represents slot units: a job can reserve more than one unit, so capacity is not simply a count of jobs.

If the worker is offline, inspect service, outbound connectivity and safe logs. If it is online but ineligible, inspect tenant, capabilities, policy and capacity. Enrollment does not configure database credentials or transfer input files.

Continue with [Worker migration and removal](#) before replacing or retiring infrastructure.

Worker reassignment, dependency backups and removal

Scope and prerequisites

Worker migration changes execution references to compatible authorized capacity. It does not automatically transfer files, databases, credentials or an operating system. Confirm the source tenant, eligible destination, worker-management authority and operational window before beginning.

A dependency backup preserves assignment metadata and integrity information. It is not a VM image or a complete data backup. When restoring or reassigning from saved metadata, verify that the destination currently satisfies the required capabilities and that the referenced objects still exist.

Inventory and prepare

1. Open **Cloud workers** → **Workers** and inspect the source worker.
2. Use **Reassign references**, **Dependency backups** or **Restore/reassign to another worker** as appropriate.
3. Read the current references and confirm the source tenant and worker.
4. Review active or queued executions, default-worker usage, Sessions, Workflows and any credential or connection dependencies.
5. Use **Drain worker** when the intent is to stop assigning new work while allowing existing work to finish.
6. Resolve incomplete inventories and provider-state errors before attempting removal.

Scheduling changes and VM power changes are different actions. Stopping a VM is not a replacement for a controlled wait for active executions to finish.

Preview and apply reassignment

1. Select the supported **Destination worker** and the intended references.
2. Confirm **Destination credentials have been configured and verified** only after checking the actual destination configuration.
3. Choose **Preview changes** and inspect the proposed changes, warnings and blockers.
4. If you change destination, selection or relevant configuration, generate a new preview.
5. Apply the allowed plan through **Apply reassignment** and verify the reported result.
6. Refresh source and destination information and inspect the saved dependency backup where applicable.

The interface manages the preview and retry identifiers. Do not alter them to override a conflict. **Download metadata copy** provides the displayed backup scope; it should not be treated as a portable copy of all object contents or secrets.



Resolve Sessions and deprovision

The removal flow can offer **Replace worker**, **Distribute Sessions** or **Disable affected Sessions**. A replacement must be compatible with every affected Session. Distribution requires a complete assignment plan. Disabling affected Sessions changes their ability to run, and linked Workflows can show a dependency error.

Open the removal dialog, refresh impact and review provider resources, references and execution activity. If the supported flow offers **Stop and deprovision the VM**, read its effect before selecting it. Enter the displayed **Type to confirm** value and acknowledgment only after all permitted prerequisites are satisfied, then submit once.

Acceptance and recovery

In a laboratory, reassign one synthetic Session, verify the published references and backup, then refresh the source impact. Successful removal requires confirmed provider absence and final reference checks. A provider error preserves the Dova record for diagnosis and retry; unknown or unavailable is not equivalent to absent.

After a network error, reopen the operation and inspect current state before retrying. Closing a dialog does not cancel a request already accepted by the provider. Do not claim resource removal or cost savings until the provider result and the corresponding Dova state are verified.

Related guides: [Cloud workers](#), [Worker installation](#) and [Audit](#).

ADMINISTRATOR · DOVA 7.0.0

ETL data cleanup and tenant lifecycle

Choose the correct operation

Deleting ETL data, extending a trial, suspending a tenant and canceling a subscription have different scopes.

Preferences → **Data Cleanup** removes the selected tenant's ETL records while preserving the listed identity, access and commercial records. It does not delete a cloud VM or substitute for a lifecycle process with retention requirements.

Confirm the intended tenant, authorized purpose, management permissions and applicable retention process before acting. Use a synthetic laboratory workspace for practice. A managed individual tenant can reject generic ETL cleanup and require its separate lifecycle process; do not change tenant type or subscription management to evade that restriction.

Review cleanup scope

The screen lists Mappings, Sessions, Workflows, Connections, credentials, runs, Repository data and tenant ETL preferences as deletion categories. It identifies users, the tenant, roles, billing and global administrator audit as preserved records. Read the current inventory because the category list alone does not establish what data is present.

1. Open **Preferences** → **Data Cleanup** and select **Tenant**.
2. Choose **Refresh inventory** and inspect counts, scope and any blockers.
3. Obtain and verify the authorized backup required for the operation. The form is not an automatic backup of the entire tenant.
4. Choose **Clean tenant ETL data** and compare the dialog's tenant and summary with the inventory you reviewed.
5. Complete **Type to confirm** and the irreversible-data acknowledgment.

6. Select **Delete ETL data** once and wait for the verified response.
7. Refresh inventory, verify the affected categories and separately confirm the listed preserved records.

If the inventory changed, refresh and review the latest state. A conflict or failed response does not authorize direct deletion of storage files or modification of the inventory revision.

Extend an eligible trial

1. Open **Subscription & billing**, select the same tenant and use **Subscription → Extend trial**.
2. Review the current lifecycle, dates and eligibility.
3. Choose **7 days**, **Custom days** or **Exact date and time**.
4. For a custom number of days, use the displayed allowed range. For an exact date, verify the dialog's timezone and resulting end.
5. Enter the required **Reason**, review the new end and choose **Extend trial** once.
6. Confirm the accepted expiry while checking that tenant identity, existing objects and original trial history remain intact.

The supported day extension counts from the current end for a running trial, or acceptance for an expired eligible trial. The dialog shows local time and stores the instant consistently. An extension does not recreate the workspace, reset quotas or grant unrelated entitlements.

Lifecycle acceptance and recovery

An expired or past-due tenant may retain permitted safe reading, export and billing access while new business mutations or execution remain restricted. Available reading should not be interpreted as authorization to recreate objects, enroll infrastructure or generate new work.

Use separate laboratory exercises for cleanup and trial extension. For cleanup, verify deletion scope and preserved accounts. For extension, verify expiry and identity continuity. Following a network error, inspect current state before retrying; an identical retry should not produce an unintended second extension.

Read [Tenants](#), [Subscription and billing](#) and [Worker removal](#) for their distinct operations.



Connectors and authentication: prepare the correct contract

A Connection defines endpoint metadata, authentication and credential custody. A Mapping component selects that Connection and declares its operation. The worker supplies the actual driver, network access and runtime identity. These are three separate contracts: saving a Connection does not prove that every read, write, pagination or recovery mode is supported.

Use the matrix to prepare the visible form. Asterisks mark fields required by the catalog; provider or method validation may add requirements. Ports are initial values, not directions to open networks. Use only authorized endpoints, resource identifiers and principals.

Declared connection catalog

Connector	Initial endpoint / port	Specific metadata	Declared methods
PostgreSQL	Host / 5432	database, schema	username_password, client_certificate, aws_iam, azure_workload_identity, gcp_iam, kerberos
MySQL	Host / 3306	database *	username_password, client_certificate
MariaDB	Host / 3306	database *	username_password, client_certificate
Microsoft SQL Server	Host / 1433	database *, schema	username_password
Oracle Database	Host / 1521	database *, schema	username_password
Snowflake	Account host / 443	account, database *, schema, warehouse *, role	snowflake_key_pair, username_password, bearer_token
Google BigQuery	HTTPS API endpoint / 443	database, schema	service_account_json, oauth_client_credentials, workload_identity, attached_service_account
Databricks	Workspace host / 443	httpPath *, catalog *, schema	personal_access_token, oauth_client_credentials
Amazon S3	HTTPS S3 endpoint / 443	bucket *, region	access_key, aws_iam
Azure Blob Storage	Storage account URL / 443	container *, directoryId	oauth_client_credentials, sas_token, account_key, azure_managed_identity, azure_workload_identity
Google Cloud Storage	HTTPS API endpoint / 443	bucket *, project *	attached_service_account, workload_identity, service_account_json
Salesforce	Salesforce HTTPS URL / 443	apiVersion *	oauth_client_credentials, jwt_bearer
Dynamics 365	Dataverse HTTPS URL / 443	directoryId *	oauth_client_credentials, azure_managed_identity, azure_workload_identity
Odoo	Odoo HTTPS URL / 443	database *	api_key
REST API	Base HTTPS URL / 443	-	api_key, bearer_token, oauth_client_credentials, client_certificate, workload_identity, no_auth

Connector	Initial endpoint / port	Specific metadata	Declared methods
SAP HANA	HANA host / 39015	database *, schema	username_password
SAP S/4HANA / ECC	SAP HTTPS URL / 443	-	bearer_token, basic_auth
Oracle NetSuite	NetSuite account URL / 443	account *	bearer_token
MongoDB	MongoDB host / 27017	database *, authSource, mongoScheme *	username_password
FTP / SFTP	File server host / 22	protocol *, rootPath *	username_password, key_pair
Teradata	Teradata host / 1025	database *, schema	username_password, ldap
Elasticsearch / OpenSearch	Cluster HTTPS URL / 443	-	api_key, basic_auth, bearer_token
Microsoft SharePoint	SharePoint site URL / 443	directoryId *, siteId *	oauth_client_credentials, azure_managed_identity, azure_workload_identity
ServiceNow	ServiceNow instance URL / 443	-	bearer_token, basic_auth
HubSpot	HubSpot API URL / 443	-	bearer_token

The installed catalog and worker capability may offer a smaller effective set. Attached cloud identities do not automatically work outside their runtime. PostgreSQL has its own integration path; a generic connector registry alone should not be treated as its complete support boundary.

Step-by-step preparation

1. Identify the target resource and the read/write operation you actually need.
2. Select the matching connector in Connections and inspect required metadata.
3. Choose the supported custody model and one of its offered authentication methods.
4. Prepare the identity, protected credentials and resource permissions with the environment owner.
5. Confirm a compatible worker, required driver and authorized network route.
6. Save the Connection and test its stored configuration.
7. Run a small authorized extraction or controlled write through the intended Mapping/Session.
8. Verify records and destination effects independently; expand scope only after the small case succeeds.

Authentication preparation

Declared method	Preparation and limits
username_password, basic_auth, ldap	A username and protected password, with the required identity/provider policy.
client_certificate	Certificate/private-key material and passphrase when required, through protected custody.
kerberos	A service principal and Kerberos environment available at runtime.
aws_iam	A runtime IAM identity; database authentication may also need a user/role. S3 uses the attached identity.
azure_managed_identity, azure_workload_identity	The supported managed/application identity and permissions on the actual resource.
gcp_iam	An IAM principal authorized for database authentication.
snowflake_key_pair	Snowflake user and a protected private key; the form recommends encrypted PKCS#8.
oauth_client_credentials	Client ID, protected secret, and provider-required audience or scope.
workload_identity, attached_service_account	Supported federation/role or an account attached to the execution environment.
service_account_json	A protected service-account document, not a pasted screenshot or shared file.
access_key, sas_token, account_key	Resource-scoped access metadata and protected secret material.
api_key	Required header/parameter name and protected value; Odoo uses its own user-key contract.
bearer_token, personal_access_token, jwt_bearer	The provider-specific token or signing identity, with adequate permissions and validity.
key_pair	SSH user/private key for SFTP only.
no_auth	A resource permitting unauthenticated access; it grants no resource permission by itself.

Metadata and protocol differences

BigQuery Database means Project ID and Schema means Dataset. Oracle Database is its Service name. Databricks needs HTTP path and Catalog. Snowflake Database/Warehouse are different fields from Role/Schema. Do not exchange these concepts simply because all values are text.

Salesforce uses the authorized My Domain URL and its configured integration. Dynamics 365 needs Directory/tenant context and the environment scope. Odoo identifies its JSON-2 API for version 19 or later. SharePoint adds a site identifier or relative site path. Confirm the current provider contract for your environment.

FTP/SFTP protocol selection changes defaults and offered methods: SFTP starts at port 22; FTP/FTPS at 21. SSH key authentication is not offered for non-SFTP protocols. FTP is not encrypted transport. Use the required protocol and TLS policy.

Custody, verification and recovery

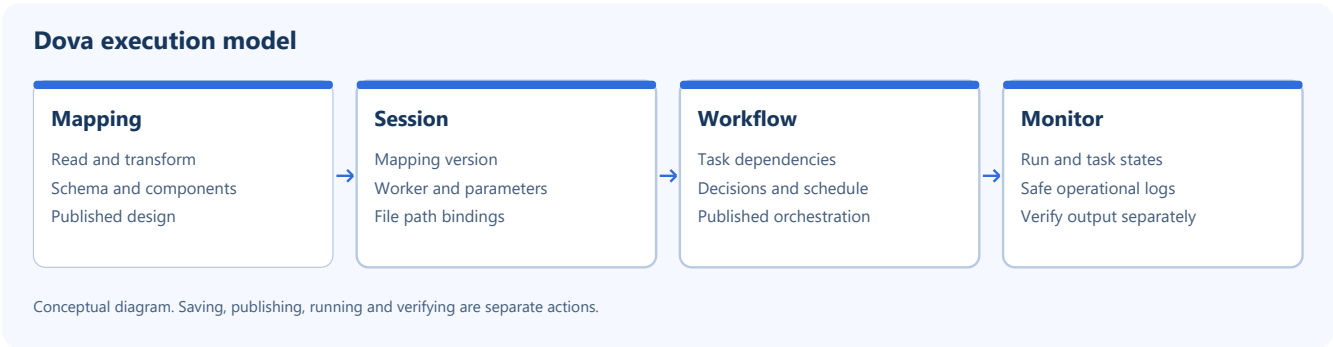
Dova Worker Agent keeps the secret in the customer runtime. Customer vault uses a protected external reference. Passwordless uses runtime identity without a stored reusable secret. Dova Vault is an exception path with protected encrypted input; Managed reference uses existing managed references. A valid-looking reference still requires runtime authorization and network access.

For a failed test, inspect safe messages, endpoint/TLS, identity permission, expiry and driver compatibility. Keep credentials and vault references out of evidence. For partial or duplicate writes, inspect destination semantics before retrying. Continue with [Connections](#) and [Mapping components](#).

REFERENCE · DOVA 7.0.0

Mapping sources and targets: formats, fields and runtime bindings

A source produces records, transformations alter them, and targets write results. The Mapping declares component format/schema; the Session supplies runtime file bindings and execution context. Database, API and Object storage components require a compatible worker. Selecting them does not install drivers, create permissions or establish network access.



Conceptual separation of transformation design and runtime configuration.

Source catalog

Component / identifier	Declared fields and options
Flat File input / source_csv	format (csv, tsv, txt); delimiter; hasHeader; headerRowNumber; columns; headerMappings; encoding (utf-8, utf-8-bom)
JSON input / source_json	format (json, jsonl); recordPath; fieldMappings; encoding (utf-8, utf-8-bom)
Excel input / source_excel	workbookFormat (auto, xlsx, xlsxm, xls); sheetMode (single, list, pattern, regex, all); sheetName; sheetNames; sheetPattern; sheetNameRegex; includeHiddenSheets; hasHeader; headerRowNumber; dataStartRowNumber; startColumn; endColumn; columns; headerMappings; skipEmptyRows; includeSheetName; sheetNameField; passwordProtected
Database input / source_database	connectionId; sourceMode (table, query); schema; objectType (table, view); objectName; sqlQuery; selectedColumns; fetchSize; queryTimeoutSeconds
API input / source_api	connectionId; method (GET); resourcePath; responsePath; fieldMappings; paginationMode (none, page, offset, cursor); pageSize; retries; timeoutSeconds
Object storage input / source_object	connectionId; resource; format (csv, jsonl); selectedColumns; delimiter

Configure file inputs step by step

1. Prepare a tiny synthetic file with a known number of records and expected fields.
2. For CSV/TSV/TXT, choose the real delimiter, header presence and one-based physical header row. Without a header, declare field names. Check logical field mapping and types.
3. Choose UTF-8 or UTF-8 with BOM as required by the file contract.
4. For JSON, choose array JSON or JSON Lines, identify the record path and map nested fields deliberately. Valid JSON can still have the wrong record structure.
5. For Excel, select workbook format, sheet mode and the intended sheets. Set header/data row and column bounds explicitly. Review hidden sheets and empty-row policy.
6. Include the sheet-name field when needed for provenance. Its initial name is `_sheet_name`.
7. Supply a workbook password only through the supported runtime mechanism. Reading XLSM is not permission to run macros.
8. Save the Mapping, then complete the actual source paths in the Session. Verify the worker can read them.

For multiple Excel sheets, confirm compatible layouts before selecting All. Notes and subtotal sheets can introduce records that do not belong to the intended dataset.

Configure remote inputs

Database input offers a table/view or a read query. Select an authorized Connection, schema/object or supported query, projected columns, fetch size and timeout. These operational fields do not replace business filters or guarantee every SQL dialect.

API input declares GET and a resource/record path, field mappings and supported pagination. Test with more records than one page to establish completeness. Object storage input uses an object key and supported CSV/JSON Lines format; its key is not a browser-local path.

Target catalog

Component / identifier	Declared fields and options
Flat File output / target_csv	format (csv, txt); fileName; delimiter; includeHeader; textQualifier (double, single, none); quoteMode (minimal, all); escapeCharacter; lineEnding (lf, crlf); encoding (utf-8, utf-8-bom); nullValue
JSON output / target_json	format (json, jsonl); fileName; pretty; includeNulls; lineEnding (lf, crlf); encoding (utf-8, utf-8-bom)
Database output / target_database	connectionId; schema; objectName; writeMode (append, replace, upsert); batchSize; commitIntervalRows; keyColumns; createIfMissing; queryTimeoutSeconds
API output / target_api	connectionId; method (POST, PUT, PATCH); resourcePath; writeMode (append, upsert); keyColumns; uniqueKeysEnforced; batchSize; successCodes; retries; timeoutSeconds
Object storage output / target_object	connectionId; resource; format (csv, jsonl); writeMode (replace); delimiter; idempotencyKey; ifAbsent; expectedEtag; expectedGeneration

Select write behavior deliberately

For Flat File output, choose header, delimiter, quote mode, escape, line endings, encoding and null representation before comparing bytes. For JSON, distinguish an array from JSON Lines and decide whether null fields are included. These are exchange contracts, not cosmetic settings.

Database append, replace and upsert have different effects; inspect keys, creation policy, batch size and commit interval. A failed load may have committed a previous batch. API write methods, accepted status codes, uniqueness and retries must match the endpoint contract. A retry is safe only when the actual operation is idempotent.

Object storage replacement supports declared preconditions such as absence, expected ETag/generation and idempotency. Preserve a failed precondition as a conflict to investigate; do not remove it merely to force the next write to succeed.

Acceptance and recovery

Create a two-row id,name CSV and a Flat File input → Flat File output Mapping. Publish, bind laboratory paths in the Session, run through a Workflow and verify two records and two fields. Quoting or line endings may differ if you selected another representation; compare against the declared contract.

For missing fields, inspect parsing and projection. For incomplete API data, inspect pagination. For wrong Excel rows, inspect sheet/range/header selection. For a partial destination write, inspect committed effects before retrying. Consult [Sessions](#) and [transforms](#).



Transformations and quality rules: verify values, rows and ordering

Transformation components can change fields, values, row count and order. Before connecting a target, declare which properties must be preserved and which should change. A valid graph can still produce incorrect business results if keys, join cardinality or rule order are wrong.

The following table preserves the declared Transform and Quality configuration fields. These identifiers are component contracts, not an open programming language. Source and target configuration is described in [Mapping components](#).

Transform and Quality catalog

Component / identifier	Category	Declared configuration
Row filter / filter	Transform	field; operator (equals, not_equals, contains, starts_with, ends_with, gt, gte, lt, lte, not_empty); value
Field selector / select	Transform	fields
Field mapper / rename	Transform	mapping
Calculated field / derive	Transform	field; template
Convert type / convert	Transform	field; targetType (string, integer, number, boolean, date, datetime); outputField; onError (fail, null, keep, default); defaultValue
Clean text / clean	Transform	fields; operations
Replace values / replace	Transform	field; find; replacement; matchMode (contains, exact, starts_with, ends_with); caseSensitive
Split field / split	Transform	field; delimiter; outputFields; keepSource; trimParts
Concatenate fields / concat	Transform	fields; outputField; separator; skipEmpty
Join / join	Transform	joinType (inner, left, right, full); leftInputId; rightInputId; leftKey; rightKey; conditions; keyType (auto, string, integer, number, boolean, date, datetime); strategy (auto, hash, sorted); caseSensitive; trimStrings; nullsEqual; rightPrefix
Default value / defaults	Quality	field; value; when (null_or_empty, null, empty)
Limit rows / limit	Transform	count; offset
Deduplicate / dedupe	Quality	fields
Row sort / sort	Transform	field; direction (asc, desc)
Group aggregate / aggregate	Transform	groupBy; metric (count, sum, avg); field; outputField
Quality check / validate	Quality	field; maxNullPercent
Required fields / quality_required	Quality	fields; maxInvalidPercent
Type check / quality_type	Quality	field; expectedType (string, integer, number, boolean, date, datetime); allowEmpty; maxInvalidPercent
Pattern check / quality_pattern	Quality	field; pattern; allowEmpty; maxInvalidPercent

Component / identifier	Category	Declared configuration
Range check / quality_range	Quality	field; minimum; maximum; inclusive; allowEmpty; maxInvalidPercent
Allowed values / quality_domain	Quality	field; allowedValues; caseSensitive; allowEmpty; maxInvalidPercent

Projection, calculation and conversion

Field selector keeps the selected fields. Field mapper establishes an output schema by renaming fields; check collisions rather than letting distinct concepts share one name. Calculated field uses a text template: `${customer}` reads a record field and `${param.region}` reads a parameter. It is not an arithmetic expression and does not execute SQL or JavaScript.

Convert type accepts string, integer, number, boolean, date and datetime. Leaving output field blank replaces the source field; another name can retain the original. Error policy chooses fail, null, keep or default. Keeping an invalid original can make a column heterogeneous, so verify it before writing to a typed database.

Text normalization

Clean text applies ordered `trim`, `collapse_whitespace`, `lower`, `upper` and `remove_accents` operations. Normalize both sides of a join key consistently. Keep an original field when legal or business requirements require its exact spelling; a comparison field can be normalized separately.

Replace values distinguishes `contains`, `exact`, `starts-with` and `ends-with` matching, with case sensitivity. Split field has `delimiter/output` names and `source/trim` options. Concatenate fields controls `separator`, `output` name and skipping empty values. Review sample edge cases; these controls are not an arbitrary regular-expression program.

Rows, joins and aggregates

Filter operators differ from Workflow decision operators. Convert textual numbers before numeric comparisons. Test values exactly at a boundary as well as above/below it. Limit rows with `offset` is a controlled window, not acceptance of the full population.

Sort controls one field/direction. Aggregate offers `count`, `sum` or `average` and the declared `group/value/output` fields; do not assume undeclared `median`, `percentile` or multiple implicit metrics. Deduplicate uses configured `match` fields; verify which record remains rather than assuming a latest-update rule.

Join offers `inner`, `left`, `right` and `full` modes with explicit `left/right` parents, `keys`, `key` types and supported strategies. Case sensitivity, trimming and null equality affect matching. The initial collision prefix is `right_`. Two matching left rows and three matching right rows can multiply records. Test unmatched keys, unique matches, duplicates and nulls before aggregating.

Test quality controls step by step

1. Prepare `id,name,amount` with two valid records and one missing amount.
2. Connect Flat File input → Clean text → Convert type → Required fields → Flat File output.
3. Apply `trim/upper` to `name`, convert `amount` with `Set null on error`, and require `id/amount`.
4. Set Maximum invalid percent to zero when no invalid record is acceptable.
5. Validate and publish the laboratory definitions, then run once.
6. Expect a quality failure caused by the invalid synthetic record; do not accept it as a completed load.

7. Correct the input, run a new attempt and verify three accepted records with expected types and names.
8. Inspect partial destination effects before retrying when a remote target is involved.

Quality percentage thresholds are acceptance limits. They do not create an automatic quarantine output unless the design explicitly implements one. Continue with [customer normalization](#), [regional summary](#) and [expressions](#).

REFERENCE · DOVA 7.0.0

Parameters, templates and Workflow conditions

Typed parameters, Calculated field templates and Workflow decision conditions are separate mechanisms. A text template is not executable code. A Workflow condition reads only the supported execution context. Preserve these boundaries when moving a value between an editor, Session override and task condition.

Typed parameter definitions

Definitions may declare name, type, description, required, sensitive, overridable, scopes, default, enum, minimum, maximum and maxLength. Up to 100 definitions are supported with unique names. Names begin with a letter, have up to 64 characters and may use letters, numbers, underscores, dots and hyphens. Prefixes `sys.` and `dova.` are reserved.

Type	Accepted value
string	Text; default maximum length 4,000 unless another valid limit is defined.
integer	A JSON integer; boolean is not an integer for this contract.
number	A finite JSON number, excluding boolean, infinity and NaN.
boolean	JSON true or false, not equivalent-looking text.
datetime	Zoned ISO text normalized to UTC at second precision.
json	Bounded JSON, up to 32,768 bytes in the normalizer.
secret	A validated protected reference rather than a cleartext value.

Description is limited to 300 characters. Max length can be 1-32,768, minimum cannot exceed maximum, and enum has 1-50 normalized candidates and is unavailable for JSON/Secret. A sensitive parameter must use Secret; marking an ordinary string sensitive does not turn it into protected custody.

Resolve value layers

The declared precedence is definition default → global → environment → worker → task. A later allowed layer replaces an earlier value. Scopes and Overridable constrain substitutions, and Required checks the resolved value. Inspect provenance in the preview when the value differs from your expectation.

For a laboratory REGION string, use default NORTH, an allowed environment override SOUTH, and an allowed task override CENTRAL. Confirm the task layer wins only when its scope and override rules permit it. Do not remove those restrictions just to make an unauthorized substitution succeed.

Calculated field text templates

`${customer}` substitutes a field and `${param.region}` substitutes a parameter. The result is text, bounded to 4,000 characters. A recognized missing identifier becomes an empty string. A dotted field name is not automatically navigation through a nested object. Use a separate typed conversion when a numeric result is required.

Workflow JSON condition operators

Operator	Meaning
eq, ne	Equality/inequality against the expected JSON value.
gt, gte, lt, lte	Ordered comparison; incompatible types evaluate false.
in, not_in	Membership/absence in an expected list.
contains	Containment in a supported string/list/dictionary value.
exists, truthy	Reference existence or truth value, without a required comparison value.

References are limited to `parameters.name`, `variables.name`, `tasks.id.status` and `tasks.id.outputs.name`. Missing references make ordinary comparisons false. Numbers stay JSON numbers; the condition does not promise automatic conversion from text.

```
{ "all": [
  { "ref": "parameters.RUN_MODE", "op": "eq", "value": "full" },
  { "ref": "tasks.load.status", "op": "eq", "value": "succeeded" }
]}
```

load must be the actual task identifier. Do not assume a universal output field such as row count. Groups all/any contain 1-20 expressions; not negates one expression. Depth over five, trees over 40 nodes and comparison values over 4,096 bytes are rejected.

Test decisions step by step

1. Define RUN_MODE as String with allowed full and sample values.
2. Create a small published laboratory Session and Workflow.
3. Add a Decision with a true expression over parameters.RUN_MODE.
4. Inspect the false branch generated automatically by the UI as the direct negation of the true expression.
5. Publish and run with full, then sample, as two distinct attempts.
6. Verify the selected branch and safe resolved parameters in each attempt.

Workflow task types include Start, Session, Decision, Timer and End; link behavior includes Success, Failure, Always and Expression. Always does not remove all parent/eligibility rules. Inspect references, types and parameter provenance when a decision differs from expectation.

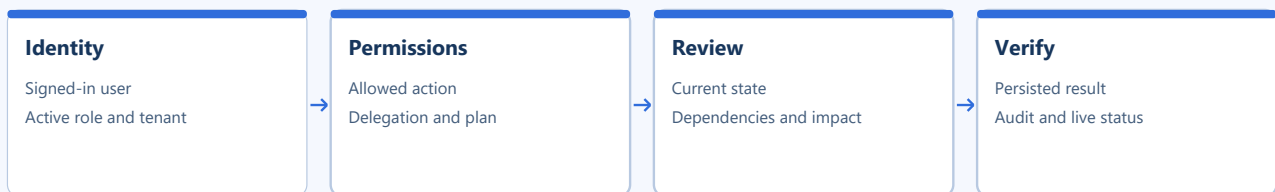
REFERENCE · DOVA 7.0.0

Permissions and actions: distinguish visibility, design and operation

Authorization combines the authenticated user, active tenant, active role, effective permissions and resource restrictions. Showing a module is a navigation choice; each action still needs its server-side authorization. Multiple assigned roles do not mean every privilege is active simultaneously.

Subscription, lifecycle and resource eligibility also matter. An execution permission does not guarantee capacity or Connection access. An inactive subscription is not resolved by granting indiscriminate privileges. Verify tenant and role first, then the action permission, plan and current resource state.

Administrator: scope before action



Conceptual review path. Administrative visibility depends on role and scope.

Conceptual authorization review. Visible controls do not replace server checks.

Dova action matrix

Surface/action	Read	Edit/operate
Overview	etl.overview.view	No Overview management permission in the catalog.
Domains/Folders	etl.domains.view	etl.domains.manage
Mappings	etl.mappings.view	etl.mappings.manage
Sessions	etl.sessions.view	etl.sessions.manage
Workflows	etl.workflows.view	etl.workflows.manage
Run Workflow	Appropriate design access	etl.workflows.run
Cancel/terminate a run	etl.monitor.view	etl.workflows.terminate
Rerun/recover	etl.monitor.view	etl.workflows.run
Monitor maintenance	etl.monitor.view	etl.monitor.manage, separate from run-specific operations.
Connections/custody catalog	etl.connections.view	etl.connections.manage
Workers/runtime/profiles	etl.runtime.view	etl.runtime.manage
Workflow Scheduler	etl.scheduler.view	etl.scheduler.manage
General policies/preferences	etl.configuration.view	etl.configuration.manage
Notification channels/settings	etl.notifications.view	etl.notifications.manage
Plans/subscriptions	billing.view	Administrative operations require billing.manage and applicable scope.

Configuration can be visible through Configuration, Runtime or Scheduler read access. Its tabs retain their own rules: seeing scheduler does not grant permission to change runtime defaults. Artifact selectors may use permitted read access without granting permission to create Domains or Folders.

Administrator matrix

Surface	Read	Manage
Administrator entry	<code>admin.access</code>	Does not replace section permissions.
Overview/resource history	<code>admin.overview.view</code>	Resources have their own management permissions.
Users/presence	<code>admin.users.view</code>	<code>admin.users.manage</code>
Tenants/tenant ETL cleanup	<code>admin.tenants.view</code>	<code>admin.tenants.manage</code>
Roles/permission catalog	<code>admin.roles.view</code>	<code>admin.roles.manage</code>
Audit	<code>admin.audit.view</code>	No audit modification permission in the catalog.
Cloud workers/dependencies	<code>admin.cloud_workers.view</code>	<code>admin.cloud_workers.manage</code>
Registrations	<code>admin.registrations.view</code>	<code>admin.registrations.manage</code>
Billing/Commerce	<code>billing.view</code>	<code>billing.manage</code>
Execution email/notifications	<code>etl.notifications.view</code>	<code>etl.notifications.manage</code>

Registrations permissions are platform-only. Global catalog and tenant-offer mutations also require the active platform role together with `billing.manage`. A tenant billing role cannot change global terms simply because it can see billing.

Interpret built-in role boundaries

ETL Viewer is read-oriented. ETL Developer includes ETL design and execution but excludes Monitor management and Workflow termination. ETL Operator includes read, execution/termination and Monitor management rather than design management. Tenant Administrator does not include billing management in its base set. Public Trial is an explicit limited set, not every ETL action; plan and expiry remain independent checks.

Custom roles and actual assignments may differ. Inspect Roles & permissions in your organization and treat the server session as authority, not browser compatibility labels. Historical route naming can differ from the current module label, so check the actual action contract instead of inferring permission from a URL word.

Run a least-privilege acceptance exercise

1. Prepare a published two-row laboratory Mapping, Session and Workflow.
2. Sign in as a permitted viewer and confirm read access while edit/run actions are denied.
3. Use an authorized operator context and verify a run and Monitor inspection without acquiring design management.
4. Use a developer context to edit/validate the design and check its separate termination restrictions.
5. Record the action, tenant, active role and safe result; do not reuse an administrator session to test a viewer.
6. Inspect relevant Audit events and restore temporary laboratory assignments.

For a rejection, investigate the correct context and permission before changing a role. Do not solve every 403 by assigning a platform wildcard. Continue with [account menu](#), [roles](#) and [Audit](#).