Flex Shipping Documentation

Complete PDF Manual

Guide for Magento administrators, technical project owners, and implementation teams.

LAST UPDATED
2026-10-13

SOURCE
Open complete documentation online

EXTENSION
Magento 2 Extension Flex Shipping

Table of Contents

Flex Shipping Documentation · flex-shipping

Overview
Flex Shipping
Getting Started
Installation
Configuration
Features
Methods and rate matrix
Rules
Package splitting and calculation scope
Simulator and audit log
Reference
Configuration Fields
Database Reference
Troubleshooting
Common Issues
FAQ

Flex Shipping

SoftwareSilo Flex Shipping lets you build shipping methods around the way an order will actually be packed. Rates can depend on destination, postcode, weight, volume and package count. You can also limit a method with cart and address conditions.

The extension is useful when a single flat rate cannot describe your shipping costs. Typical examples include bulky goods, products that must ship separately, regional freight rates and mixed B2B orders.

How the setup fits together

1. A **method** defines what the customer sees and how packages are built.
2. **Rates** define where the method is available and how much it costs.
3. Optional **rules** decide whether the method applies to a particular cart or address.
4. The **simulator and audit log** help you check the result against a real quote.

Methods, rates, rules and audit entries are managed under **Sales > Flex Shipping**. The carrier itself is enabled under **Stores > Configuration > Sales > Delivery Methods > Flex Shipping**.

Access is controlled by separate ACL resources per area — **Methods**, **Rates**, **Rules**, **Audit** and **Simulator** (with its own **Simulator Search** and **Simulator Run** permissions) under *Sales > Flex Shipping* in the admin role tree.

The audit log is enabled by default and keeps entries for 30 days (`audit_enabled`, `audit_retention_days`).

Start here

- Install Flex Shipping
- Configure your first method
- Build the rate matrix
- Limit a method with rules
- Configure package splitting
- Test a quote and read the audit log
- Troubleshoot a missing or unexpected rate
- FAQ

Installation

Install the package through the authenticated SoftwareSilo Composer repository — the repository URL and token are listed under **Account > Licenses** in your SoftwareSilo account (see Installation & Updates).

Install the package

```
composer config bearer.repo.softwaresilo.io <token>
composer config repositories.softwaresilo composer
https://repo.softwaresilo.io/
composer require mageb2b/flex-shipping:*
```

Enable the module and apply setup changes:

```
php bin/magento module:enable MageB2B_ExtensionManager
MageB2B_FlexShipping
php bin/magento setup:upgrade
```

Production-mode stores normally require dependency injection compilation and static-content deployment. Follow your normal release process, then clear the Magento cache.

Confirm the installation

After deployment:

1. Open **Stores > Configuration > Sales > Delivery Methods**.
2. Confirm that **Flex Shipping** is listed.
3. Open **Sales > Flex Shipping > Methods**.
4. Confirm that the Methods, Rates, Rules and Audit pages are available to your admin role.

If a menu entry is missing, sign out and back in after refreshing the Magento cache. Also check that the admin role has the required Flex Shipping permissions.

Before creating the first method

Decide which product attributes contain:

- weight
- volume, or length, width and height
- an optional packing mode
- an optional maximum quantity for separately packed products

Standard Magento weight can be used. Volume and packing attributes are only needed

when your rules depend on them.

Uninstall

The module ships an uninstall routine: `bin/magento module:uninstall MageB2B_FlexShipping` drops the four Flex Shipping tables (`flex_shipping_method`, `flex_shipping_rate`, `flex_shipping_rule`, `flex_shipping_audit`) and removes its columns from quote and order tables. That permanently deletes all methods, rates, rules and audit history — export anything you still need first.

Continue with Configuration to enable the carrier and create a working method.

Configuration

Start with one method and one broad rate. Once that method appears in checkout, add narrower regions, package limits and rules one at a time.

1. Enable the carrier

Go to **Stores > Configuration > Sales > Delivery Methods > Flex Shipping** and set:

- **Enabled** to Yes
- **Carrier Title** to the group label customers should see
- **Sort Order** to its position among other carriers
- **Enable Audit Logging** according to your support needs
- **Audit Retention (Days)** to the time audit records should remain available; `0` keeps them indefinitely

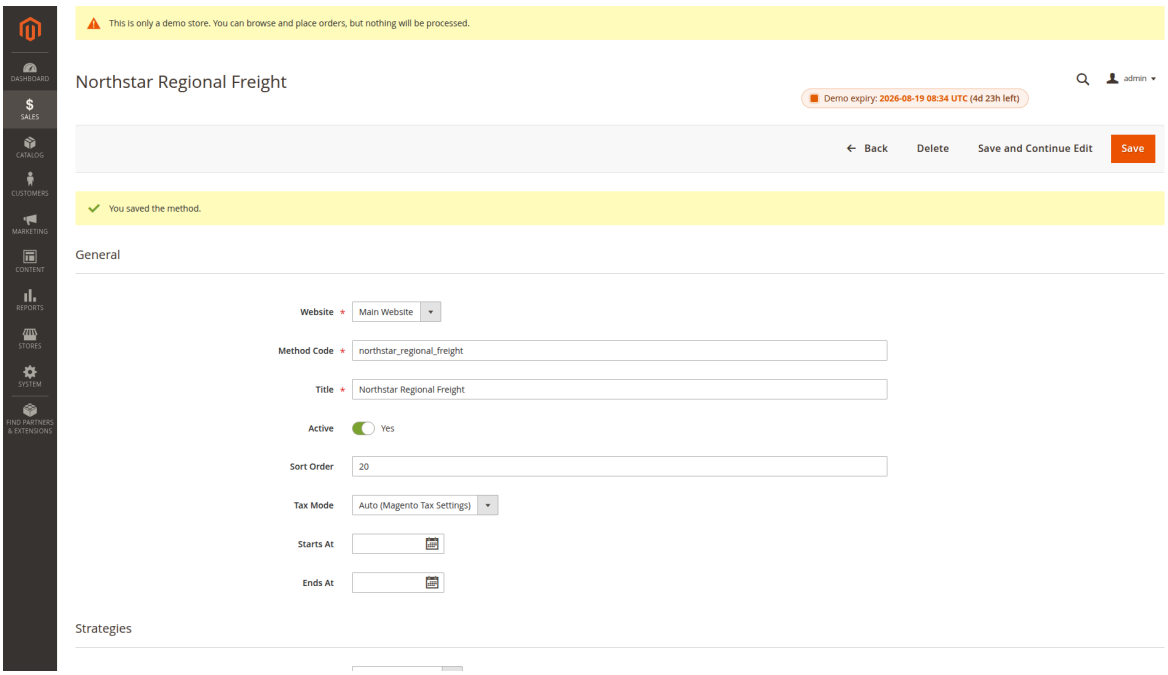
Configuration can be scoped by website where Magento offers that scope.

2. Create a method

Go to **Sales > Flex Shipping > Methods** and select **Add Method**.

Use a stable **Method Code**, such as `regional_freight`. Set the storefront title, website, active state and optional start or end date. Then choose:

- the package split strategy
- package limits and their source
- the rate conflict strategy
- the **Payload Depth** — how much package detail is stored with the order
- product attribute mappings and missing-value behavior



3. Add a first rate

Open **Sales > Flex Shipping > Rates** and add a shipment-level rate for all countries. Give it generous weight and volume ranges, a fixed price and an active state.

Use **Validate Methods** on the Methods page, then test a cart. If the method appears, replace the broad rate with the real destination and package bands you need.

4. Add rules only when needed

A method without rules is eligible whenever its active settings and rate matrix match. Adding rules changes this: at least one active rule must match the quote address for the method to remain available.

Rules use Magento cart and address conditions. Keep the first rule simple, verify it with a real quote, and only then add nested conditions.

5. Test before release

Use **Sales > Flex Shipping > Audit** to simulate a saved quote. Check the returned amount, package summary and audit record before enabling the method on a live website.

Methods and rate matrix

A method is the shipping option offered to the customer. Its rate rows decide whether that option can serve the destination and what it costs.

Method settings

Each method belongs to one website or all websites and has its own code, title, active dates, tax mode and sort order. It also owns the package, conflict and attribute settings used during calculation.

The tax modes are:

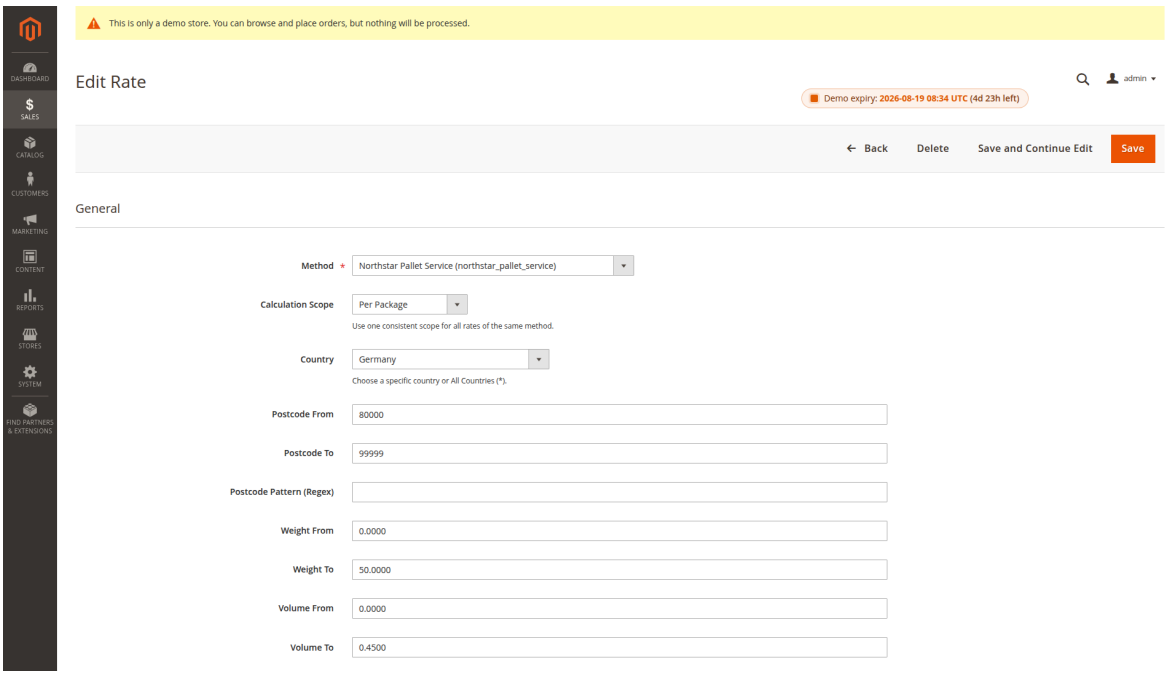
- **Auto** follows Magento's shipping tax configuration.
- **Net** treats the configured rate as excluding tax.
- **Gross** converts the configured amount to a net shipping amount using the applicable shipping tax rate.

Rate matching

A rate can be limited by:

- country, including the `*` option for all countries
- postcode range or postcode pattern
- minimum and maximum weight
- minimum and maximum volume
- shipment or package calculation scope
- active date range

All configured bounds are inclusive. A specific-country row is considered before an all-country row when both match.



Postcodes

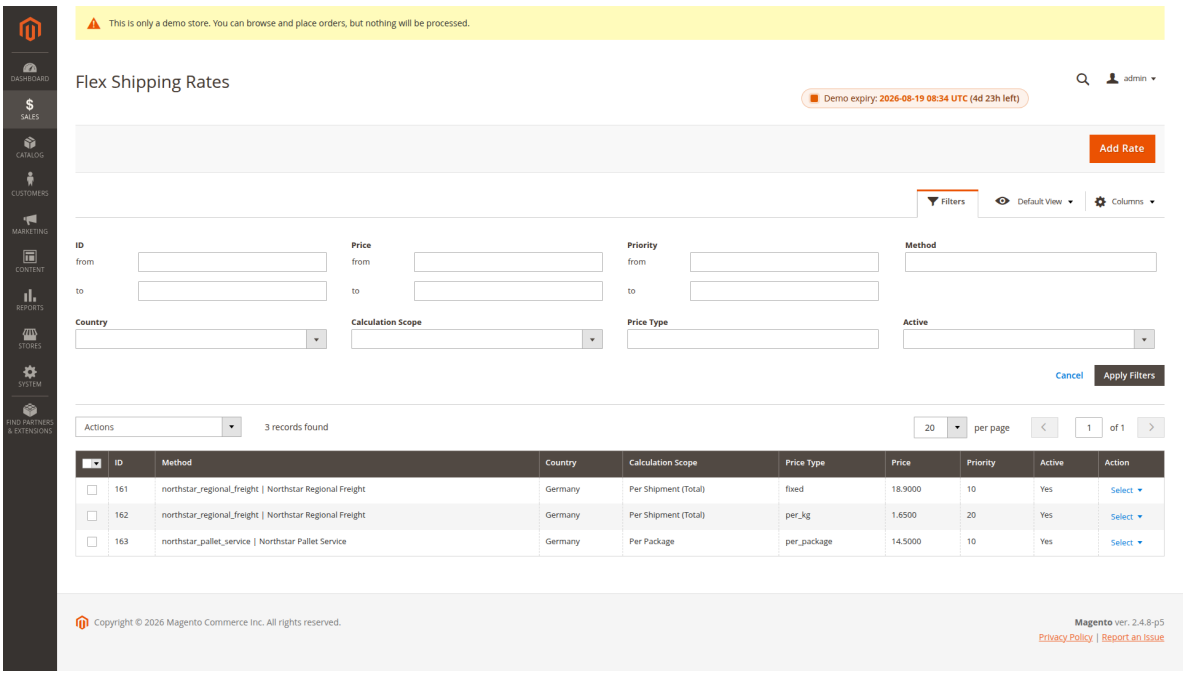
Use **Postcode From** and **Postcode To** for straightforward ranges. If both are empty, every postcode matches. A `*` endpoint also matches every postcode.

Use **Postcode Pattern (Regex)** only when a range cannot express the area. The pattern must match the full normalized postcode. For numeric ZIP+4 destinations, a value such as `12345-6789` is evaluated as `12345`.

Keep patterns short and test them in the simulator. Unsafe constructs such as lookarounds and backreferences are rejected.

Price types

Price type	Calculation
Fixed	Uses the entered amount once for the selected rate.
Per Weight Unit	Multiplies the amount by the relevant shipment or package weight.
Per Package	Multiplies the amount by the package count. In package scope, each package is evaluated separately.



Overlapping rates

The method's conflict strategy decides which matching row wins:

- **First Match** uses the first row after country specificity, priority and ID ordering.
- **Cheapest** compares the calculated prices of all matching rows.
- **Priority then Cheapest** considers only the lowest priority number and chooses the cheapest row in that group.
- **Custom** uses the strategy selected in the **Custom Conflict Fallback** field directly below.

Use distinct bands where possible. Conflict strategies are useful for intentional overlaps, but they should not be the only way to understand a large matrix.

Method rules

Rules sit above the rate matrix. If a method has no active rules, it can proceed to rate matching. If it has active rules, at least one must match the shipping address.

Rules are checked by ascending priority. **Stop Processing** ends the rule loop after a match; it does not replace rate conflict handling.

Rules

Rules decide when a method's rates apply to a cart. Open them under **Sales > Flex Shipping > Rules**.

Rule grid

The grid lists all rules with their method, name, priority, status and validity dates. Use **Add Rule** to create one, or select rows and apply the **Delete** mass action to remove several at once.

Rule fields

Field	Purpose
Method	The Flex Shipping method this rule belongs to.
Name	Internal label shown in the grid and the audit log.
Priority	Lower numbers are evaluated first when several rules could apply.
Stop Processing	When enabled, no lower-priority rule is evaluated after this one matches.
Active	Disabled rules are kept but ignored.
Starts At / Ends At	Optional validity window — leave empty for a rule that always applies.

Conditions

The **Conditions** fieldset uses the same condition builder as Magento cart price rules:

- Combine conditions with **ALL** or **ANY** and nest groups for more complex logic.
- Each condition compares a cart or address attribute (for example subtotal, weight, shipping country or postcode) against a value.
- A rule with no conditions applies to every cart.

Rules are evaluated together with the method's package split strategy: a matching rule makes its rates available, and the rate conflict strategy decides which rate is offered.

Related articles

- Methods and rates
- Simulator and audit

Package splitting and calculation scope

Package splitting turns the physical items in a cart into one or more packages before Flex Shipping chooses a price. Virtual products are ignored, and parent and child quote items are handled so their weight and volume are not counted twice.

Calculation scope

Per Shipment (Total) matches one rate against the totals for the complete shipment. The selected price is calculated once.

Per Package matches every package separately and adds the results. If even one package has no matching package-level rate, the method is not returned.

Use one calculation scope consistently across the rate rows for a method.

Product packing modes

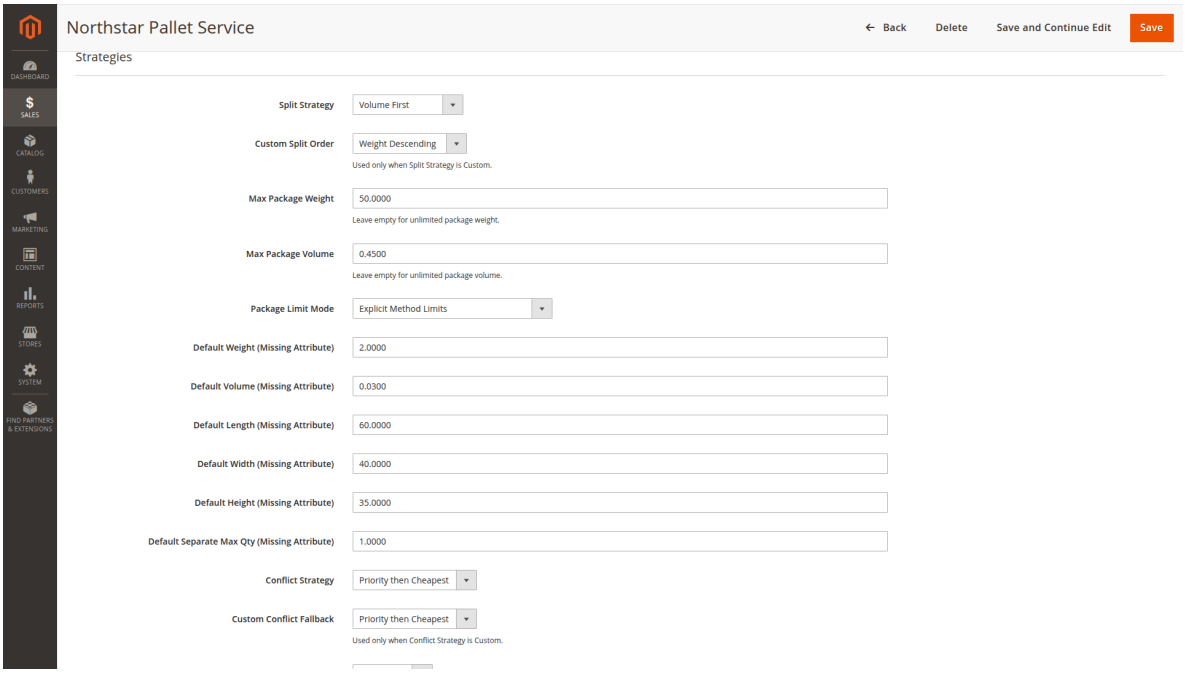
The mapped product attribute can contain:

- `normal`, which allows the item to share a package
- `always_single`, which creates one package for every unit
- `separate_max_n`, which separates the product into packages containing no more than the configured quantity

Special packing modes require whole units. For a fractional quantity, the extension falls back to normal packing.

Split strategies

- **First Fit Decreasing** fills packages in the normal packing order.
- **Weight First** starts with the heaviest items.
- **Volume First** starts with the largest items by volume.
- **Custom** lets you order items by descending weight, volume or density.



Where package limits come from

Explicit Method Limits uses **Max Package Weight** and **Max Package Volume** from the method.

Derived from Rate Table derives limits from the matching rate band for the current destination and shipment metrics.

Hybrid uses the lower available value from the explicit and derived limits.

If no limit is available, normal items may remain in one combined package. When limits are derived from rates, the extension uses a matching band rather than combining unrelated limits from different rows.

Missing product data

Choose one policy for missing mapped values:

- **Block Method** removes the method from the result.
- **Use Zero** substitutes zero.
- **Use Custom Defaults** uses the default values entered on the method.

Blocking is safer when dimensions are essential to the carrier price. Defaults are useful when your catalog has known, consistent gaps. Avoid zero unless a zero value is genuinely meaningful for that attribute.

What is stored with the order

Payload Depth controls the package summary copied from the quote to the order:

- **Count Only** stores the number of packages.
- **Summary** stores useful package totals without every item detail.
- **Full Details** stores the full package breakdown.

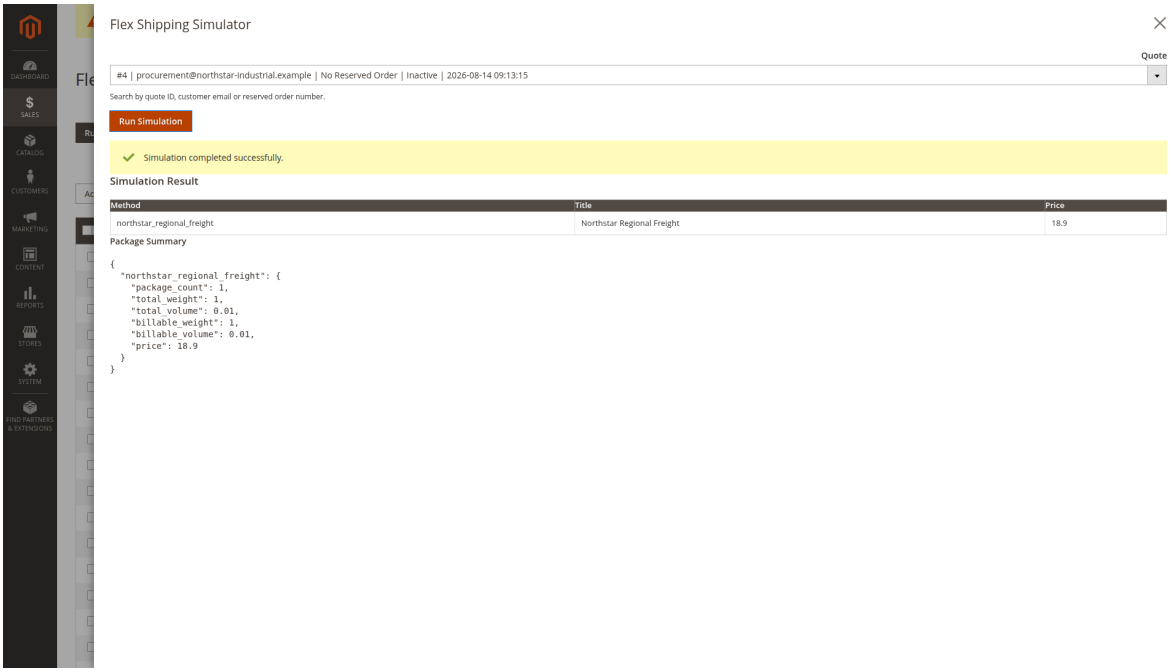
Choose the smallest level that still supports fulfilment and customer service.

Simulator and audit log

The simulator runs Flex Shipping against an existing Magento quote. It is the quickest way to check destination matching, package construction and the final amount without completing an order.

Run a simulation

1. Prepare a cart with realistic products, quantities and a shipping address.
2. Open **Sales > Flex Shipping > Audit**.
3. Open the simulator and search by quote ID, customer email or reserved order number.
4. Run the simulation.
5. Compare the returned methods and amounts with the package summary.



The simulator uses saved quote data. If you changed the cart or address, make sure the quote was saved before testing again.

Audit entries

When **Enable Audit Logging** is active, a successful method calculation records:

- quote and website IDs
- method code
- calculated amount
- split and conflict strategies
- configured payload depth
- package metrics at the selected detail level

Audit logging is diagnostic. A logging failure is written to the Magento log and does not deliberately block shipping rate collection.

Retention

Audit Retention (Days) controls automatic cleanup. Entries older than the configured number of days are removed by Magento cron each day. A value of `0` disables automatic deletion.

Keep enough history to investigate customer reports, but do not retain full package details longer than your operational policy requires.

Configuration Fields

This reference lists the settings exposed by `mageb2b/flex-shipping` in Magento Admin.

Configuration paths are included for controlled deployments and `config:set`. Use Magento Admin for normal changes so field validation and scope inheritance still apply. The extension status display is not listed because it reports installation state rather than a configurable value.

Flex Shipping

Field	Configuration path	Input	Scope
Enabled	`carriers/flexshipping/active`	Select	Default, Website
Carrier Title	`carriers/flexshipping/title`	Text	Default, Website
Sort Order	`carriers/flexshipping/sort_order`	Text	Default, Website
Enable Audit Logging	`carriers/flexshipping/audit_enabled`	Select	Default, Website
Audit Retention (Days)	`carriers/flexshipping/audit_retention_days`	Text	Default

Database Reference

Flex Shipping keeps the customer-facing method, its rates and its eligibility rules separate. This lets one shipping method support several rate tiers and rule conditions.

Relationship	Why it matters
`flex_shipping_method` belongs to one `store_website`	Limits the shipping method to the website where it is configured.
`flex_shipping_rate` belongs to one `flex_shipping_method`	Stores the prices or tiers evaluated for that method.
`flex_shipping_rule` belongs to one `flex_shipping_method`	Controls when the method is available for a cart or customer.
`flex_shipping_audit` belongs to one `store_website`	Keeps evaluation and change history within the correct website scope.

This overview does not document rule payloads or internal audit columns. Configure methods through Magento Admin.

Common Issues

The carrier is missing from checkout

Check the carrier and method first:

1. Flex Shipping is enabled for the current website.
2. The method is active and within its start and end dates.
3. The method belongs to the current website or all websites.
4. At least one active method rule matches, if rules exist.
5. A rate matches the destination, postcode, weight, volume and calculation scope.

Use **Validate Methods** and then simulate the same quote under **Sales > Flex Shipping > Audit**.

A postcode does not match

Check whether a regex pattern is set. A pattern takes precedence over the From and To fields and must match the full postcode. Invalid or disallowed patterns do not match.

For ranges, confirm that From is not greater than To. Numeric postcodes are compared numerically; other values are compared as normalized uppercase text.

Package scope returns no method

Every package needs a matching rate row with **Calculation Scope** set to **Per Package**. One unmatched package removes the entire method.

Check the package summary for an unexpected weight or volume. Then review product attribute mappings, missing-value policy, packing mode and package limits.

The selected rate is not the expected row

List all rows that match the same country, postcode, metrics and date. Then review the method's conflict strategy:

- First Match also considers country specificity and priority.
- Cheapest compares calculated prices, not only the raw amount field.
- Priority then Cheapest ignores rows outside the best priority group.

The amount is lower when free shipping is involved

For a shipment-level **Per Weight Unit** rate, Magento's free-method weight is used for the price calculation. Package building and rate-band matching still use the physical shipment metrics.

Audit entries are missing

Confirm that **Enable Audit Logging** is set to Yes. The module writes audit entries after a method price is resolved, so rejected methods do not produce the same successful calculation record.

Also check **Audit Retention (Days)** and ensure Magento cron is running.

A rule appears to be ignored

Rules only decide whether their own method is applicable. They do not edit a rate price. Check the rule's active dates, priority and conditions against the quote's shipping address. If the method has rules, at least one active rule must match.

FAQ

Can I create several storefront shipping options?

Yes. Create one method for each option, give each a unique method code and assign its own rates, package settings and optional rules.

Can rates differ by postcode?

Yes. Use a postcode range for ordinary bands or a full-match regular expression for more complex formats.

What happens if several rates match?

The method's conflict strategy chooses the first match, the cheapest calculated result, or the cheapest result within the best priority group.

Can the extension price every package separately?

Yes. Set the method and its rates to **Per Package** calculation scope. Each built package must have a matching rate.

Can one product always ship alone?

Yes. Map a product attribute as **Pack Mode** and give the product the `always_single` value. Use `separate_max_n` when a package may contain a limited number of units.

What if a product has no dimensions?

The method can be blocked, use zero, or use custom defaults. Choose the behavior under **Missing Attribute Policy**.

Are virtual products included in package weight and volume?

No. Virtual products are excluded from physical package building.

Does a free-shipping cart always make Flex Shipping free?

Not in every pricing mode. For shipment-level weight pricing, the price weight can become zero or use Magento's remaining free-method weight. Fixed and package-based rows follow their own calculations.

Can I test a rate without placing an order?

Yes. Save a realistic quote, then run it through the simulator on the Flex Shipping Audit page.

Does the audit log clean itself up?

Yes, when Magento cron is running and **Audit Retention (Days)** is greater than zero. Set it to `0` to keep entries indefinitely.