How to install the extension

1. Backup your store's database and web directory.
2. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
3. Copy the installation instructions from page My Downloadable Products to the SSH console and press ENTER.
4. Run command `php -f bin/magento module:enable Mirasvit_Core Mirasvit_Feed Mirasvit_Report` to enable the extension.
5. Run command `php -f bin/magento setup:upgrade` to install the extension.
6. Run command `php -f bin/magento cache:clean` to clean the cache.
7. Deploy static view files

   ```
   rm -rf pub/static/*; rm -rf var/view_preprocessed/*; php -f bin/magento setup:static-content:deploy
   ```

Manage templates

To manage templates go to **Products ? Advanced Product Feeds ? Templates**.

The extension includes more then 45 ready to use templates for all the popular price comparison engines (Google Shopping, Amazon, eBay, Shopzilla, etc.).

Note
Before you create a new template, you need to check the product feed specification for the specified marketplace (template format type, fields delimiter, required fields, etc).

You can create templates for any comparison shopping engine.

To create a new template, follow these steps:

1. Go to **Products ? Advanced Product Feeds ? Templates**, then click the **Add Template** button in the upper right-hand corner.
2. Fill in the following fields:
   - **Name** - name of the new template.
   - **File Type** - feed output format. There are 3 types that are available for the data feed:
     - **CSV** - comma-separated values, each item placed on a new line. File extension is *.csv*.
     - **TXT** - same as CSV file, but with *.txt* extension.
     - **XML** - uses tags to define blocks of content. Information about your items is enclosed within these tags, which are indicated by angle brackets. File extension is *.xml*.
   - At **Content Settings** tab, you you need to configure the template depending on requirements and file type. Currently supported formats are:
     - **CSV, TXT**
     - **XML**
3. Click the **Save** button.

List of the pre-installed templates

Template name	Format
Amazon(inventory)	XML
Amazon(Marketplace)	XML
Amazon(pictures)	XML
Amazon(price)	XML
AmazonAds	TXT
Become Europe	CSV
Beslist	XML
Billiger.de	CSV
Bing Shopping	TXT
CJ	XML
Domodi	XML
eBay(Commerce Network)	CSV
eBay.com (Store)	CSV
Facebook Dynamic Ads	XML
Facebook (storefront)	CSV
Fishpond	CSV
GetPrice Categories	XML
GetPrice Products	XML
Google Shopping	XML
Google Shopping (configurable products)	XML
Google Shopping Review	XML
Google Shopping Update	XML
idealo.it	CSV
it.bestshopping.com	CSV
Kelkoo	XML
Kieskeurig	XML
LeGuide.com	TXT
Marktplaats	XML
Newegg	XML
Newegg(inventory)	XML
Nextag	TXT
pagineprezzi	TXT
Partner-Ads	XML
PriceGrabber	TXT
PriceMe	XML
PriceRunner	XML
PriceSpy	TXT
Rakuten (Apparel)	TXT

Template name	Format
Sears.com Inventory	XML
Sears.com Item	XML
Sears.com Price	XML
ShareASale	CSV
ShopMania	XML
Shopping.com	XML
ShopPrice	XML
Shopzilla	TXT
SingleFeed	CSV
The Find	CSV
TradeDoubler	CSV
TradeTracker	CSV
Twenga	CSV
Webgains	CSV
Yandex Market	XML

How to create a new data feed

Note
Pre installed templates may require additional attributes and settings by reason of the specific selling items or country location. Check a product's attribute requirements using marketplace specifications.

To create a new data feed, follow these steps:

1. Go to **Product ? Advanced Product Feeds ? Feeds**. Press button **Add Feed**.
2. Select one of the existing templates to create a feed. To create an empty feed, select **Empty Template**.
3. Press button **Continue**.
4. Fill-in a few requirement fields:
    - **Name** - name of the data feed.
    - **Filename** - name of the data feed file. File will be located at `[magento_path]/pub/media/feed/filename`.
    - **Store View** - store view for which a data feed will be generated.
    - **Is Active**

    Additionally, if you selected **Empty Template**, you need to fill-in these fields:

    - **File Type** - there are three file types available for the data feed.
        - **CSV** - a comma-separated values each item placed on a new line. File extension is *.csv*.
        - **TXT** - same as CSV file, but with *.txt* extension.
        - **XML** - uses tags to define blocks of content. Information about your items is enclosed within these tags, which are indicated by angle brackets. File extension is *.xml*.
5. Press button **Save and Continue Edit**.
6. To generate data feed, press button **Generate** at the top right corner.

How to configure XML Feed

If you select the **XML** file type at the tab **Content Settings**, you can create/edit xml schema for your feed.

By default we provide templates for XML feeds, so you can easily copy it and change it for your requirements.

Usually, Comparison Shopping Engines provide a template of the xml file. Based on this template, you can create your own xml schema.

Typical xml schema:

```
<?xml version="1.0" encoding="utf-8" ?>
<rss version="2.0" xmlns:g="http://base.google.com/ns/1.0">
    {% for product in context.products %}
      <item>
        <attribute_1><![CDATA[{{ product.attribute_1 }}]]></attribute_1>
        <attribute_2><![CDATA[{{ product.attribute_2 }}]]></attribute_2>
        .....
      </item>
    {% endfor %}</rss>
```

Product cycle block:

```
{% for product in context.products %}
    ...{% endfor %}
```

Inside this block, you can use any product attribute.

Category cycle block:

```
{% for category in context.categories %}
    ...{% endfor %}
```

Inside this block, you can use any category attribute.

Review cycle block:

```
{% for review in context.reviews %}
    ...{% endfor %}
```

Inside this block, you can use any review attribute.

Attribute (pattern) block:

```
<attribute_1><![CDATA[{{ product.attribute_1 }} ]]></attribute_1>
<attribute_1><![CDATA[{{ category.name }} ]]></attribute_1>
<attribute_1><![CDATA[{{ review.nickname }} ]]></attribute_1>
```

The attribute code must be enclosed in double curly brackets: *{{ product.attribute_code }}*. You can use all attribute codes available at **Store** > **Attributes** > **Product** and all static attributes (ex. entity_id, created_at etc).

Additionally in curly brackets you can place any available pattern. [Full pattern list of patterns]

Note

Characters like < and & are illegal in XML elements.

- < will generate an error because the parser interprets it as the start of a new element.
- & will generate an error because the parser interprets it as the start of an character entity.

We suggest to enclose all patterns in **CDATA** block `<attribute><![CDATA[{{ pattern }}]]></attribute>`. In this case, xml data feed will be valid.

How to configure CSV, TXT Feed

If you select a **CSV** or **TXT** file type at tab **Content Settings**, you can create/edit attribute schema for your feed.

Note

When you use pre-installed templates, you need to check if the attributes from the template response are for the same values as your store attributes.
If this attribute doesn't exist in your store, set the appropriate product attributes or patterns for the same line.

Content Settings

Before creating the attribute scheme, you need to fill in the required file settings:

- **Fields Delimiter** - delimiter, which allows you to split text into columns in your feed file. Supported delimiters are:
  - Comma ","
  - Tab "\t"
  - Colon ":"
  - Space " "
  - Vertical pipe "|"
  - Semi-colon ";"
- **Fields enclosure** - allows you to enclose data in your feed file.
- **Include Columns Header** - set "Yes" to include a header row (attribute names) in the first line of your feed file.
- **Extra header** - set "Yes" to include an additional header row in the first line of your feed file. It will always be above the first attributes row or the columns header.

Field Mapping

In field mapping table you can add/remove rows, change rows ordering, set output type, symbols limit. Each row in a mapping table is a column in the data feed file.

To add a new column to your CSV feed, you need to create a new row and fill it with a few params:

- **Column Name** - the header column name.
- **Value Type** - the following types are available:

    - **Pattern** - allows you to enter a static value or use patterns like from the XML template.
    - **Attribute** - allows you to select any store attribute from the drop down list.
    - **Parent Product** - allows to export configurable products. In this case, simple associated products will have an attribute. If you have configurable or bundle products, we suggest you use this Type **Parent Product** with fields: "Product URL", "Grouped id"
- **Value** - allows you to select the attribute or put the pattern for the column output.

Note

To modify the **Value** output, click on the cogwheel in the required column to use the **Add Modifier** button. There will appear a list of available modifiers for usage.
To check information about modifiers, go to the page: Output Filters

FTP settings

Extension can automatically deliver data feed file via FTP to Shopping Engine Service.

Note
Check the marketplace merchant account for FTP details, or ask about FTP credentials at the marketplace Support Center

To configure FTP delivery, follow these steps:

1. Open tab **FTP Settings** at feed edit page.
2. At the tab you need to enable FTP delivery and fill in these fields:
    - **Protocol** - you can select FTP/SFTP or SFTP connection.
    - **Host Name** - the server where you would like to send your feed.
    - **User Name** - the username to FTP server.
    - **Password** - the password to FTP server.
    - **Path** - optional field, enter path to your merchant folder provided by the Shopping Engine Service.
    - **Passive mode** - most FTP servers work in Passive mode, even when you use a firewall.
3. Press the button **Save And Continue Edit**.

After enabling FTP delivery, you can run delivery of feed manually by pressing the button **Delivery Feed** at the right top corner.

Additionally, the extension can deliver the feed by schedule after each feed generation.

Schedule Task Settings

You can configure your feed to automatically to generate the data feed file by schedule.
If the FTP settings are enabled, the feed will be automatically delivered to the marketplace after generation by the schedule.

Note

To generate the feed by schedule, magento cron must be configured. See How to Setup Cron for Magento.

To configure the schedule follow these steps:

1. Open tab **Scheduled Task** at feed edit page.
2. Set **Status - Enabled** to enable the feed generated by schedule
3. Set up the following fields:
    - **Days of the week** - days of the feed generation.
    - **Time of the day** - time of the feed generation.
4. Press button **Save And Continue Edit**.

For example, if selected **days** are *Monday*, *Wednesday* and **time** *03:00AM*, *05:00AM*, then the feed will be generated 4 times during a week.

Note

If you have a few feeds with scheduled tasks, to prevent cron job errors you need to set a different scheduler **time** for each feed.

Email Notifications

Open your the feed **Additional** tab on the feed's edit page.

The extension can automatically send an email to the notifications for the next events:

- Successful Export
- Unsuccessful Export
- Successful Delivery
- Unsuccessful Delivery

Fill in the following lines:

- **Email** - email addreses for email notifications (use comma separator to set more then one email address).
- **Notification Events** - sets events for further email notification.

Press the button **Save And Continue Edit**.

Google Analytics Campaign

Note

Google Analytics should be configured and activated in order to use this feature.

Extension can automatically append google analytics campaign parameters to your product's URLs.

To configure **Google Analytics Campaign**, follow these steps:

1. Open tab **Google Analytics** at feed edit page.

2. Fill in the 3 required fields:
    - **Campaign Source** - Identifies a search engine, newsletter name, or other source.(i.e. google, citysearch, newsletter)
    - **Campaign Medium** - Identifies a medium such as email or cost-per-click. (i.e. cpc, banner, email)
    - **Campaign Name** - Identifies a specific product promotion or strategic campaign. (i.e product, promo code, or slogan)
      Also, as an option, you can fill in other fields:
    - **Campaign Term** - Identifies paid keywords.
    - **Campaign Content** - Differentiates ads or links that point to the same URL.
3. Press button **Save And Continue Edit**.

After adding google analytics parameters, you need to generate your feed. In the feed file all product URLs will be
`http://example.com/product.html?fep=…&fee=…&utm_source=…&utm_medium=…&utm_name`
After feed generating, you don't need to do any additional configuration adjustments.
Additionaly, in the campaign fields, you can use any pattern.
To track **Google Analytics Campaign** log-in into your account, go to **Traffic Sources** > **Campaigns**. Select campaign source from the list.

Additional Settings

Extension allows you to set up additional settings for feed export.
Go to the feed tab **Additional**.

Email Notifications

Check the **Email Notifications** tab.

Report the Configuration

For each feed, you can enable/disable tracking clicks and orders by changing the setting **Enable Reports**.

If the feature is enabled, the extension appends two special arguments to the product's url (fee=, fep=) to track clicks and orders,
where **fee** - ID of the feed, and **fep** - ID of the product. For example:
http://example.com/product.html?fee=1&fep=3

Product Filters

Filtering is one of the most important parts of the feed creation process.
By using filters, you will not export products which have zero price or are out of stock, products without images, disabled products etc.

To create a new product filter follow these steps:

1. Go to **Product ? Advanced Product Feeds ? Manage Filters**. Press button **Add Filter**.

2. Set filter **Name**.
3. Select feeds for which the filter will be applied. Also you can select applied filters on the feed edit page.
4. At the **Rules** tab you can specify all required conditions.
5. Select any attribute from the list and use the filter conditions. The extension allows you to set the **conditions combination**, using **if ALL** and **if ANY**, **TRUE** or **FALSE** rules.

Note

Do not set in the filter **Visibility** conditions: **Catalog** or **Catalog, Search** if you have configurable products. Otherwise, simple **Associated Products** which have the status **Not Visible Individually**, will not be included into the feed.

Product Filter Examples

- **Filter example for Simple products**

  For example, you want to export only **Simple** products from the **Furniture** category with a product **Price** greater than 50 and products that are available in the **Stock**:

  - Set **Status is Enabled** (not necessarily)
  - Set **Product Type is Simple Product**
  - Add a new **Conditions Combination**. Set **if ANY** rule. Add 2 conditions for attribute **Visibility** :
    - **Visibility is Catalog**
    - **Visibility is Catalog, Search**

  This condition allows you to include products which have the **Visibility** status: **Catalog** or **Catalog, Search**.
  If you don't set this condition combination, the feed will be generated also with products which have status: **Not Visible Individually**.
  Thus, Configurable **Associated Products** will not be included into the feed.

  Continue setting up main conditions:

  - Set **Price greater than 50**
  - Set **Stock availability is In Stock**
  - Select **Category** attribute and set condition **is one of**.
    Click the **Chooser** icon and you will see the store category tree.
    To include products from certain categories, tick the **Furniture** category and their subcategories from the list.
    Category ID's will be added automatically to the text line.

- **Filter example for Simple and Configurable products**

  For example, you want to export **Simple** and **Configurable** products which are **not** from **Cell Phones** and **Cameras** categories.
  Product **Name** contains word **ecco** and product **quantity** equals or greater than 10:

  - Set **Status is Enabled** (not necessarily)

    - Set **Name contains ecco**
    - Set **Stock availability is In Stock**
    - Set **Quantity equals or greater than 10**
    - Select **Category** attribute and set condition **is not one of**.
      Click the **Chooser** icon and you will see the store category tree.
      To exclude products from certain categories, tick the **Cell Phones** and **Cameras** categories and subcategories from the list.
      Category ID's will be added to the text line automatically.
- **How to exclude products without Base images**

  Add next condition to your filter:

  `Base Image is not ...`

List of Patterns

All patterns must be enclosed in curly brackets. In patterns you can use codes of attributes, filters, links to parent products, base php functions and calculations.

The base pattern schema `{{ entity.attribute | filter | filter }}`

Attribute Patterns

- `{{ product.entity_id }}` - ID of the product
- `{{ product.sku }}` - an identifier of the product
- `{{ product.name }}` - the name of the product
- `{{ product.description }}` - the description of the product
- `{{ product.short_description }}` - the short description of the product
- `{{ product.status }}` - the status of the product

  Possible values:

    - Enabled
    - Disabled
- `{{ product.visibility }}` - the visibility of the product

Possible values:

- Not Visible Individually
- Catalog
- Search
- Catalog, Search

- `{{ product.url_key }}` - the url key of the product
- `{{ product.url }}` -the direct url to the product
- `{{ product.price }}` - price of product (without discounts, catalog rules etc)
- `{{ product.regular_price }}` - regular price of the product
- `{{ product.final_price }}` - final price (saleable) of the product

  The price of product after applying special price and catalog price rules.
- `{{ product.special_price }}` - special price of the product

  The special price of the product.

  Special price ignore values of *Special Price From Date* and *Special Price To Date*
- `{{ product.regular_price }}` - regular/base price of the product
- `{{ product.tax_rate }}` - tax rate for the product
- `{{ product.category }}` - the name of the assigned category to the product

Note

If the product is assigned to a few categories, the extension selects **Category** using this logic:
The most nested category is always selected. For example, if a product is assigned to a few categories at different levels, the attribute `{category}` returns the **name** of the category, that is the most nested in the category tree.
If the product is assigned to a few categories at the same level, the extension selects a category with the lowest position of the product. Change the position of the product you can at **Catalog** > **Manage Categories**, tab **Category Products**

- `{{ product.category.id }}` - a ID of the assigned category to the product

Note

If the product is assigned to a few categories, the extension selects **Category Id** using this logic:
The most nested category id is always selected. For example, if a product is assigned to a few categories at different levels, the attribute `{category_id}` returns the **id** of the category, that is the most nested in the category tree.
If a product is assigned to a few categories at the same level, the extension selects a category id with the lowest position of the product. To change position of the product go to **Catalog > Manage Categories**, tab **Category Products**

- `{{ product.category.path }}` - a path of the category names

  E.g. Computers > Notebooks > Apple

Note

If product is assigned to a few categories, the extension selects **Category Path** using this logic:
The most nested category path is always selected. For example, if a product is assigned to a few categories at different levels, the attribute returns the **path** that is the most nested in the category tree.
If a product is assigned to a few categories at the same level, the extension selects a category with the lowest position of the product. To change the position of the product go to **Catalog > Manage Categories**, tab **Category Products**

- `{{ product.category.url }}` - a direct url of the assigned category to the product

  The direct url to the parent category.

Note

If the product is assigned to a few categories, the extension selects a **Category Url** using this logic:
The most nested category url is always selected. For example, if a product is assigned to a few categories at different levels, the attribute `{{ product.category.url }}` returns the **url** of the category that is the most nested in the category tree.
If the product is assigned to a few categories at the same level, the extension selects a category id with the lowest position of the product. To change the position of the product go to **Catalog > Manage Categories**, tab **Category Products**

- `{{ product.attribute_set }}` - name of the assigned attribute set to the product
- `{{ product.qty }}` - quantity of the product
- `{{ product.is_in_stock }}` - stock status of the product

  Possible values:

  - **0** - Out of Stock
  - **1** - In Stock

- `{{ product.image }}` - direct url to base image of the product
- `{{ product.thumbnail }}` - direct url to the thumbnail image of the product
- `{{ product.small_image }}` - direct url to the small image of the product
- `{{ product.gallery[0] }}`, `{{ product.gallery[1] }}` ... - a direct url to gallery images of the product
- `{{ product.rating_summary }}` - average product rating (from 0 to 5)
- `{{ product.reviews_count }}` - number of approved reviews

Parent product values

You can use the suffix **.parent** (`{{ product.parent.name }}`, `{{ product.parent.price }}`, `{{ product.parent.url }}` etc), if you need to return value to the parent product.

Example

If the current product associated with configurable/grouped/bundled product, pattern `{{ product.parent.url }}`, will return the URL to the parent product. If extension can't find the parent product, it uses the current product.

Note: Parent suffix is very useful when you export **simple** products with visibility **Not Visible Individually**. In this case, the product can't have a direct link, so you must use a link to the parent product.

Examples

```
{% for product in context.products %}
    {% for image in product.gallery %}
    <picture>{{ image }}</picture>
    {% endfor %}
    <created>{{ product.created_at | dateFormat: 'd.m.Y H:i:s' }}</created>
{% endfor %}
```

Filters

Filters are simple methods that modify the output of numbers, strings, variables and arrays. They are placed within an output tag `{{ }}` and are separated with a pipe character `|`.

String/HTML Filters

- lowercase - converts a string into lowercase.

  ```
  {{ product.name | lowercase }}
  ```

  ```
  Original: Dash Digital WatchOutput: dash digital watch
  ```

- uppercase - converts a string into uppercase.

  ```
  {{ product.name | lowercase }}
  ```

  ```
  Original: Dash Digital WatchOutput: DASH DIGITAL WATCH
  ```

- replace - replaces all occurrences of a string with a substring.

  ```
  {{ product.name | replace: 'Digital', 'Analog' }}
  ```

  ```
  Original: Dash Digital WatchOutput: Dash Analog watch
  ```

- append - appends characters to a string.

  ```
  {{ product.name | append: ' - best choice' }}
  ```

  ```
  Original: Dash Digital WatchOutput: Dash Digital Watch - best choice
  ```

- prepend - prepends characters to a string.

  ```
  {{ product.name | prepend: 'Best choice - ' }}
  ```

  ```
  Original: Dash Digital WatchOutput: Best choice - Dash Digital Watch
  ```

- capitalize - capitalizes the first word in a string.

  ```
  {{ product.color | capitalize }}
  ```

  ```
  Original: dark redOutput: Dark red
  ```

- escape - escapes html tags in a string.

  ```
  {{ product.description | escape }
  ```

- nl2br - inserts a `<br>` linebreak HTML tag in front of each line break in a string.

  ```
  {{ product.short_description | nl2br }
  ```

- remove - removes all occurrences of a substring from a string.

  ```
  {{ product.name | remove: 'Digital' }}
  ```

```
Original: Dash Digital WatchOutput: Dash Watch
```

- stripHtml - strips all HTML tags from a string.

```
{{ product.description | stripHtml }}
```

- truncate - truncates a string down to 'x' characters.

```
{{ product.name | truncate: '15' }}
Original: Dash Digital WatchOutput: ash Digital Wa
```

- plain - converts any text to plain format.

```
{{ product.description | plain }}
```

- ifEmpty - return argument, if value is empty string

```
{{ product.color | ifEmpty: 'Multi color' }}
Original: Output: Multi color
```

- dateFormat - converts string to specified date-time format.

```
{{ product.created_at | dateFormat: 'd.m.Y H:i' }}
Original: 2016-02-18 10:11:12Output: 18.02.2016 10:11
```

- json - converts object or array to JSON format.

```
{{ product.gallery | json }}
```

Number Filters

- ceil - rounds an output up to the nearest integer.

```
{{ product.weight | ceil }}
Original: 1.423 Output: 2
```

- floor - rounds an output down to the nearest integer.

```
{{ product.weight | floor }}
```

```
Original: 1.423 Output: 1
```

- round - rounds the output to the nearest integer or specified number of decimals.

```
{{ product.weight | round: '2' }}
Original: 1.423 Output: 1.42
{{ product.weight | round }}
Original: 1.423 Output: 1
```

- numberFormat - formats number to specified format (php function).

```
{{ product.price | numberFormat: '2', '.', ',' }}
```

Price/Currency Filters

- price - formats price to default format.

```
{{ product.price | price }}
Original: 100.42 Output: 100.42
```

- convert - converts price from base currency to specified currency.

```
{{ product.price | convert: 'EUR' }}
Original: 100Output: 92.28
```

- inclTax - include tax to product price

```
{{ product.price | inclTax | price }}
```

- exclTax - exclude tax from product price

```
{{ product.price | exclTax | price }}
```

Array Filters

- first - return first element in array.
-

last - return last element in array.

- join - join array to string using glue.
- count - return number elements in array.
- select - select values for key from array.

URL Filters

- secure - return secure url (with https://)
- unsecure - return unsecure url (with http://)
- mediaSecure - return media url using "Base URL for User Media Files"

```
{{ product.image | mediaSecure }}

Original: http://example.com/pub/media/catalog/product/m/h/mh03-black_main
Output: https://cdn-secure.com/catalog/product/m/h/mh03-black_main.jpg
```

- mediaUnsecure - return media url using "Secure Base URL for User Media Files "

```
{{ product.image | mediaUnsecure }}

Original: http://example.com/pub/media/catalog/product/m/h/mh03-black_main
Output: http://cdn.com/catalog/product/m/h/mh03-black_main.jpg
```

Image Filters

- resize - resize image

```
{{ product.image | resize: 100, 100 }}

Original: http://example.com/pub/media/catalog/product/m/h/mh03-black_main
Output: http://example.com/pub/media/cache/200x200/catalog/product/m/h/mh0
```

Note

- `{{ product.name | truncate: '150' }}`
- `{{ product.description | plain | truncate: '1000' }}`

- `{{ product.weight | round: '2' }} kg`
- `{{ product.manufacturer | ifEmpty: 'not set' }}`

Dynamic attributes

Dynamic attribute - an attribute that can return values depending on conditions or the value of other attributes.

To create a new dynamic attribute, follow these steps:

1. Go to **Product ? Advanced Product Feeds ? Dynamic Attributes**. Press button **Add Attribute**.
2. Fill in the following fields:
    - Name - name of the dynamic attribute
    - Code - code of the dynamic attribute. You will see this code when you select this dynamic attribute in the templates.
    - Set conditions:
        - Press "Add "OR" condition"
        - Press "Add "AND" condition"
        - Select attribute at left corner
        - Select condition
        - Select or input condition value
        - Select Output Type
        - Select attribute or input pattern - when condition is true, dynamic attribute will return this value.
    - Press the button Save.

    To set default value, just leave Conditions column without conditions.

Dynamic variables

Dynamic variable - its a user defined php code that must return a string value.

To create a new dynamic variable, follow these steps:

1. Go to **Product ? Advanced Product Feeds ? Dynamic Variables**. Press the button **Add Variable**.
2. Fill in the following fields:
    - Name - name of the dynamic variable
    - Code - code of the dynamic variable. You will see this code when you select this dynamic variable in the templates.
    - PHP Code

Example

- Name: GTIN
- Code: gtin

PHP Code:

```
$gtin = $product->getGtin();
if (!$gtin) {
    $gtin = $product->getId();
    $gtin .= str_repeat('0', 12 - strlen($gtin));
}return $gtin;
```

Usage:

```
{{ product.variable:gtin }}
```

Output:

```
124200000000
```

Category Mapping

Note

What is category mapping?

The category names you are using in your Magento store aren't always the same as the ones used by Comparison Shopping Engines to reference your products.
This means you have to find out which Shopping Engine categories have the best match with yours.

To create new category mapping, follow these steps:

1. Go to **Product ? Advanced Product Feeds ? Category Mapping**. Press the button **Add Category Mapping**.
2. Fill in the mapping **Name**.
3. Fill in the new names related to store categories.
   If the child category doesn't have its own related name, it will use the related name of the parent category.

   Note

   When you start typing the category names, the extension will automatically show all the available categories from the Google Shopping taxonomy txt file.
   You can add other txt files with taxonomies to use it in a category mapping in the folder **magento_root/vendor/mirasvit/module-feed/src/Feed/Setup/data/mapping**
4. Press the button **Save**.

Now you can use new category mapping in your feeds. You can select created **Category Mapping attributes** from the drop down list of the template attributes.

Note

- For CSV feeds, at attribute selector you need to select the attribute **Category Mapping: Name**
- For XML feeds, you should use the pattern, **{{product.mapping:mapping_id}}**

Example

For example:

{{ product.mapping:1 }}

Where **1** is a category mapping id.

Reports

The extension allows you to track the clicks and orders after the customer redirects from the marketplaces.

Note

The extension can track the clicks and orders only via special arguments **ff=** and **fp=**, which are appended to the product Urls.
To enable this option, go to the feed tab **Additional** > **Enable Reports** and set **Yes**.

The extension allows you to show the Report graphics by **Feed**, **SKU**, **Day**, **Week**, **Month**, **Year**.

Main columns of the **Report**:

- **Number of Clicks** - numbers of times the customer redirects from the marketplace.
- **Number of Orders** - numbers of the orders which were made after customer is redirected from the marketplace.
- **Revenue** - revenue of the order.
- **Revenue per Click** - average revenue for each individual click.

Import/Export Data

Go to **Products > Import/Export Data**.

Extension allows you to import and export next feed entities:

- **Templates**
- **Filters**
- **Dynamic Attributes**
- **Dynamic Categories**
- **Dynamic Attributes**

Select the required tab and import/export the required files.

Command Line Interface

Usage: `php -f bin/magento [options]`

- `mirasvit:feed:export` - generate all active feeds
- `mirasvit:feed:export --id=<id>` - generate feed with specified id
- `mirasvit:feed:delivery` - delivery of all active feeds
- `mirasvit:feed:delivery --id=<id>` - delivery feed with specified id

  Note

  `mirasvit:feed:export --step=100` - if your server has enough resources you can use additional parameter **--step**. Changing of this parameter can decrease feed generation time or necessary server resources. By default step equals 50.

How to upgrade extension

To upgrade extension follow next steps:

1. Backup your store's database and web directory.
2. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
3. Run this command to update current extension with all dependencies: `composer require mirasvit/module-feed:* --update-with-dependencies`.

   Note

   In some cases the command above is not applicable, it's not possible to update just current module, or you just need to upgrade all Mirasvit modules in a bundle. In this case command above will have no effect.

   Run instead `composer update mirasvit/*` command. It will update all Mirasvit modules, installed on your store.

4. Run command `php -f bin/magento module:enable Mirasvit_Core Mirasvit_Feed Mirasvit_Report` to re-enable the extension.
5. Run command `php -f bin/magento setup:upgrade` to install updates.
6. Run command `php -f bin/magento cache:clean` to clean the cache.
7. Deploy static view files

   ```
   rm -rf pub/static/*; rm -rf var/view_preprocessed/*; php -f bin/magento setup:static-content:deploy
   ```

Disabling the Extension

Temporarily Disable

To temporarily disable the extension please follow these steps:

1. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
2. Run command `php -f bin/magento module:disable Mirasvit_Feed` to disabled the extension.
3. Login in to the Magento back-end and refresh the store cache (if enabled).

Extension Removal

To uninstall the extension please follow these steps:

1. Login to the SSH console of your server and navigate to the root directory of the Magento 2 store.
2. Run command `composer remove mirasvit/module-feed` to remove the extension.
3. Login in to the Magento back-end and refresh the store cache (if enabled).

Change Log

1.0.91

(2018-11-30)

Improvements

- M2.3 support

1.0.90

(2018-11-29)

Fixed

- ## fix of the problem with the data serialization for magento versions < 2.3

1.0.89

(2018-11-28)

Fixed

- support of magento 2.3

1.0.88

(2018-11-23)

Fixed

- export full path for small image pattern

1.0.87

(2018-11-20)

Improvements

- added "Product ID" filter condition
- added "Qty of children in stock products" attribute for export

---

1.0.86

(2018-11-19)

Fixed

- getLevel() issue

1.0.85

(2018-11-15)

Improvements

- Added new templates: "Facebook Dynamic Ads", "Domodi", "Marktplaats"
- Improved "Google Shopping Review" template according to the new XML Schema requirements
- Updated existing templates

1.0.84

(2018-11-09)

Improvements

- added 'Stock Status' product attribute

Fixed

- fixed an issue with the incorrect categories export according to their position

---

1.0.83

(2018-10-25)

Fixed

- fixed an issue with the incorrect timezone export date at the 'time' pattern
- fixed an issue with the exporting empty feed if any filter conditions are applied

---

1.0.82

(2018-10-02)

Fixed

- Possible issue with sorting

1.0.81

(2018-09-11)

Improvements

- Performance

1.0.80

(2018-09-06)

Improvements

- Added memory/interation time to CLI command

1.0.79

(2018-09-05)

Fixed

- Feed emails are not working
- When using filters out of stock products excluded from feed
- Issue with truncate filter (multibyte strings)
- Fixed error "A technical problem with the server created an error. Try again to continue what you were doing. If the problem persists, try again later."

Improvements

- ## Created separate option group for Category Mappings

1.0.77

(2018-08-09)

Improvements

- improved the "Is Salable" filter condition to exclude children products according to their parent salable status

Fixed

- fixed an issue with the incorrect export of the category path by levels

---

1.0.76

(2018-07-24)

Fixed

- fixed an issue with the incorrect Review Summary Rating export
- fixed an issue with the SQL errors during Magento installation

---

1.0.74

(2018-07-23)

Fixed

- fixed the Eval function Error via using the Dynamic Variables

---

1.0.73

(2018-07-17)

Fixed

- added compatibility of Notification Email classes with the Magento 2.1.x versions

---

1.0.72

(2018-07-16)

Fixed

- fixed an issue with the incorrect feeds delivery and export via CLI commands
- fixed an issue with the sending notification emails

1.0.71

(2018-07-10)

Fixed

- Pie chart is not displayed: set default column for chart

1.0.70

(2018-06-28)

Improvements

- added 'Final Price' filter condition
- improved 'Is Salable' filter condition

Fixed

- fixed issues with processing image and image sizes filter conditions
- ## fixed an issue with the incorrect and missing enclosures in the txt and csv feeds

1.0.69

(2018-06-04)

Fixed

- fixed an issue with the incorrect feed schedule time execution by cron

---

1.0.68

(2018-03-19)

Fixed

- fixed an issue with incorrect filtering of "Stock Availability" condition (#37)

1.0.67

(2018-03-12)

Fixed

- fixed an issue with incorrect filtering products by category ids
- Reports are not visible in 'developer' mode

1.0.66

(2018-02-26)

Fixed

- Report is not displayed (affects since 1.0.65)

1.0.65

(2018-02-23)

Improvements

- compatibility with latest version of Mirasvit module Reports

Fixed

- fixed an issue with the showing store categories on the Category Mapping page

---

1.0.64

(2018-02-15)

Improvements

- Added filter condition to export products by amount of in stock children

---

1.0.63

(2018-02-02)

Bugfixes

- fixed an issue with the files locking on Windows #23

1.0.62

(2017-12-15)

Fixed

- Issue with yaml parsing library

1.0.61

(2017-12-07)

Fixed

- Fixed issue with deleting dynamic variables from the mass action grid

---

1.0.60

(2017-11-22)

Feature

- Feed generation report

Improvements

- Display number of generated items in the feed

1.0.59

(2017-11-17)

Fixed

- Issue with price export by cron

1.0.58

(2017-10-12)

Fixed

- Fixed an issue with the exploding array instead of string

1.0.57

(2017-10-12)

Fixed

- Fixed an issue related to incorrect export of parent product values in Magento Enterprise edition
- Fixed an issue with the exploding array instead of string

1.0.56

(2017-09-28)

Improvements

- Compatibility with Magento 2.2

1.0.55

(2017-06-27)

Fixed

- Dynamic attribute conditions do not work properly when admin user has custom permissions
- Solve 'Duplicate entry' error occurred on a filtration step

Improvements

- Modifier to remove all non-utf8 characters
- Ability to limit gallery collection

1.0.54

(2017-06-13)

Fixed

- Issue with image resolver

1.0.53

(2017-05-05)

Fixed

- Product.price to product.finalPrice for google templates
- Fixed a feed generation error via CLI - "Area code is already set"
- Fixed feed generation error via command Command Line Interface - "Area code is already set"

Features

- Ability to use Product attributes in the Google Analytics tabs

---

1.0.52

(2017-03-30)

Improvements

- Changed clicks logging mechanism to more stable

Fixed

- CI for import/export

1.0.51

(2017-03-24)

Features

- Import/export Feed entities

Documentation

- updated documentation

1.0.50

(2017-03-21)

Fixed

- Possible issue with filtration

1.0.49

(2017-02-27)

Fixed

- Fixed an issue with patterns preview

1.0.48

(2017-02-27)

Improvements

- Changed report version to 1.2.*

Fixed

- Fixed an issue with dynamic attributes

1.0.47

(2017-02-20)

Improvements

- Ability to export swatches values

1.0.46

(2017-02-14)

Improvements

- Added dedicated cron group for feed generation process

Fixed

- Fixed an issue with Plain filter

1.0.45

(2017-02-01)

Fixed

- Fixed an issue with special char "|" in filters

1.0.44

(2017-01-20)

Improvements

- Changed logic of exporting configurable product attributes. If configurable product return empty value, module select values for child products

---

1.0.43

(2017-01-19)

Fixed

- Fixed an issue with thumbnail images

1.0.42

(2017-01-11)

Fixed

- Fixed an issue with json

1.0.41

(2017-01-10)

Improvements

- Ability to export all attributes (CSV header XALL)
- Added json filter {{ product.gallery | json }}

1.0.40

(2017-01-09)

Fixed

- Fixed an issue with tax rate preview

1.0.39

(2017-01-06)

Improvements

- Implemented lock mechanism for prevent parallel feed generation (CLI)

1.0.38

(2017-01-06)

Fixed

- Fixed an issue with category mapping (edit page)

1.0.37

(2017-01-06)

Fixed

- Fixed an issue with mapping

1.0.36

(2017-01-03)

Fixed

- Fixed an issue with nested category mapping

1.0.35

(2017-01-03)

Fixed

- Added symfony/yaml to depends

Improvements

- Ability to place category taxonomy files to pub/media/feed/mapping

---

1.0.33

(2016-12-21)

Fixed

- Fixed an issue with filtration by category (simple products that not visible in catalog)

1.0.32

(2016-12-15)

Improvements

- Improved performance for csv feed edit page

1.0.31

(2016-12-14)

Fixed

- Fixed an issue with current date

1.0.30

(2016-12-09)

Improvements

- Compatibility with M2.2

1.0.29

(2016-12-06)

Fixed

- Fixed an issue with pattern output for dynamic attributes

1.0.28

(2016-12-05)

Fixed

- Fixed an issue with parent selector in dynamic attributes

1.0.27

(2016-11-11)

Fixed

- Fixed an issue with filtration by Yes/No attributes

1.0.26

(2016-11-04)

Fixed

- Fixed an issue with Status filter

1.0.25

(2016-11-02)

Fixed

- Changed crontab
- Fixed an issue with feed delivery (SFTP)

1.0.24

(2016-10-21)

Fixed

- Fixed an issue with compilation

1.0.23

(2016-10-20)

Fixed

- Fixed an issue with deleting dynamic attribute
- Fixed an issue with filter by stock quantity

1.0.21

(2016-10-12)

Improvements

- Use the same font-awesome.min.css for all extensions

1.0.20

(2016-10-12)

Fixed

- Fixed an issue with dynamic attributes

1.0.19

(2016-10-07)

Fixed

- Fixed an issue with images url
- Fixed an issue with wrong product url (multi-store configuration)
- Select product category depend on current store

1.0.18

(2016-09-06)

Improvements

- Export category with maximum level that related with product

Fixed

- Fixed an issue with category mapping

1.0.17

(2016-09-05)

Fixed

- Fixed possible issue with generation process

1.0.16

(2016-08-23)

Fixed

- Fixed an issue with delivery button

1.0.15

(2016-08-23)

Fixed

- Fixed a possible issue with filtration

1.0.14

(2016-07-18)

Fixed

- Fixed an issue with reports
- CI

1.0.13

(2016-06-24)

Fixed

- Compatibility with Magento 2.1
- Fixed an issue with validation dynamic attribute rules

1.0.12

(2016-06-17)

Fixed

- Fixed an issue with dynamic attribute values

1.0.11

(2016-05-25)

Fixed

- Fixed an issue with empty attribute value, if attribute contains numbers

---

1.0.10

(2016-05-25)

Features

- Implemented autocomplete for category mapping

Improvements

- Ability use dynamic variables in {% for %} cycle

Fixed

- Fixed an issue with dynamic attribute conditions (multi-select)

1.0.9

(2016-04-11)

Fixed

- Fixed an issue with menu
- ACL for dynamic variables

1.0.8

(2016-04-07)

Improvements

- Integration tests for Dynamic Variables
- Dynamic Variables
- Offset attribute for "for" cycle

Fixed

- Remove Root Category from categoryCollection method
- Fixed an issue with capture tag

Documentation

- Dynamic Variables

1.0.7

(2016-04-01)

Fixed

- Fixed an issue with cross-browsing ajax requests

1.0.6

(2016-04-01)

Fixed

- Fixed an issue with feed generation (not all products for feed with >100K products)
- Styles

Improvements

- Capture tag
- Ability to defined product ids for feed preview (stored in cookies)
- Dynamic Attributes
- Reports
- Added ability to export Related/CrossSell/UpSell products

1.0.5

(2016-03-14)

Improvements

- Added random param to export/progress url for prevent request caching
- Added ability to export Related/CrossSell/UpSell products
- Added 2 new filters: mediaSecure, mediaUnsecure
- Export all images with direct link (without cdn if defined)
- Updated "Save" button for Templates, Filters and Category Mapping
- Added tax rate resolver "{{ product.tax_rate }}"
- Clean feed history by cron (leave history for last 3 days)

Fixed

- Updated "Google Shopping" template

- Fixed an issue with cron job scheduling

1.0.4

(2016-02-17)

Improvements

- Improved feed generation process by cron, plus added integration tests for cron job
- Improved feed history (CLI and manual export process)
- Added new filters inclTax, exclTax

Fixed

- Fixed an issue with broken link to Category Mapping in top menu
- Fixed an issue with gallery images (for cycle)
- CouplingBetweenObjects
- Fixed an issue with rounding prices when apply filter inclTax, exclTax
- Fixed an issue with trimming chars in {for} cycle
- Fixed an issue with removing liquid filters durign change expression (xml)
- Fixed an issue with mysql error at feed preview on empty catalog
- Fixed an issue with exclude tax calculations
- Fixed an issue with wrong js mapping (reports.js)

1.0.3

(2016-02-08)

Improvements

- Added new filters for urls: "secure" and "unsercure"
- Added prices including tax

1.0.2

(2016-02-07)

Improvements

- Improved xml highlighting
- Split product resolve to few files depends on product type
- Added date filter
- Added ability to select associated products {{ product.associatedProducts }}

Fixed

- Integration tests
- Fixed an issue with select/multiselect attribute values
- Fixed an issue related with wrong loop length in liquid cycles

- Fixed an issue with "No elements to pop"
- Minor issue

Welcome to Advanced Product Feeds User Manual

Whether you are new to Advanced Product Feeds or an advanced user, you can find some useful information here.

In this guide, you'll learn how to: