



# **Manual del plugin de Prestashop para Sales Layer**

# Índice

|  |           |
|--|-----------|
| <b>Notas previas del conector</b>                                | <b>4</b>  |
| <b>Propósito del conector - Versión</b>                          |           |
| <b>Instalación</b>   | <b>5</b>  |
| Primeros pasos   | <b>5</b>  |
| Sincronización   | 7         |
| Diagnóstico y solución de errores                                | 8         |
| Rendimiento  | 9         |
| <b>Configuración: otros parámetros</b>                           | <b>10</b> |
| Exportar títulos multidioma                                      | <b>11</b> |
| Creación de características del producto y atributos de variante | 11        |
| Creación de campos personalizados                                | 12        |
| Colores personalizados   | 14        |
| Tags   | 15        |
| Adjuntar archivos  | 15        |
| Precios  | 15        |
| Gestión de descuentos  | 16        |
| Valores permitidos por los campos de descuentos                  | 17        |
| Imágenes   | 17        |
| Transportes  | 18        |
| Productos Tipo Pack  | 19        |
| Proveedores  | 19        |
| Otros campos disponibles   | 20        |
| <b>Personalización de campos</b>                                 | <b>21</b> |
| <b>Ejecución de indexadores</b>                                  | <b>22</b> |
|  | <b>22</b> |

|  |           |
|--|-----------|
| <b>Exportación de imágenes de productos desde Prestashop a Sales Layer</b> | <b>22</b> |
| <b>Mejorar el rendimiento en multitienda</b>                               | <b>24</b> |
| Un conector para varias tiendas  | 24        |
| Un conector para cada tienda   | 25        |
| <b>Personalización del módulo</b>  | <b>26</b> |
| <b>Faqs: preguntas frecuentes</b>  | <b>26</b> |
| <b>Versionado</b>  | <b>28</b> |

# Notas previas del conector

## AVISO IMPORTANTE

Este es un manual de uso para realizar una instalación genérica de nuestro plugin en las versiones indicadas. En la instalación pueden influir distintos aspectos como:

- Configuración del servidor
- Personalizaciones del e-commerce
- Otros plugins instalados

Estas circunstancias hacen que para su instalación pueda ser necesaria la figura de un implementador.

De media, el tiempo para completar una implementación puede variar entre un día y 2 meses, dependiendo de:

- La experiencia del implementador
- La casuística concreta
- La personalización
- La preparación de los datos

Sales Layer como servicio SaaS no realiza implementaciones ni puede dar soporte a estas instalaciones, pero dispone de partners especializados con conocimiento de la herramienta.

En cualquiera de estas instalaciones recomendamos encarecidamente:

- La utilización de un entorno staging para pruebas, para verificar que la sincronización funciona con datos necesarios y, en caso de datos preexistentes, que no se pierdan. Es imprescindible que el entorno staging sea exactamente igual al de producción para que las pruebas sean eficaces.
- Que el implementador tenga acceso al servidor en producción para realizar la instalación final (dado que la configuración puede diferir de staging).
- Para agilizar el proceso de los primeros tests de integración, aconsejamos realizar importaciones iniciales con pocos datos para que el cliente se asegure de que la estructura de la información es la correcta. Para ello, es posible, desde el conector de Sales Layer, establecer múltiples filtros (etiquetas, filtros con búsquedas manuales, fórmulas...).

## ADICIONALMENTE

- La modificación del conector de Sales Layer implica el reenvío de toda la información de producto (con el consiguiente tiempo de carga).
- Si el vínculo entre Sales Layer y el E-commerce ya ha sido realizado, todos los cambios presentes en SL van a ser repercutidos en el E-commerce. Esto implica que si por ejemplo se cambia el producto de estado a borrador en SL, se va a cambiar el estado del producto a borrador en el E-commerce.
- El *plugin* conecta con la plataforma *e-commerce* de forma simple y genérica, teniendo en cuenta los diferentes casos personales de los clientes y sus puntos más vulnerables. Hay que tener muy en cuenta que, utilizando este conector, no podrán mapearse campos de otros plugins externos. Es importante ser conscientes de que el plugin puede tener dificultades a la hora de mapear los campos tal y cómo se requieren. Si surge alguna dificultad te invitamos a que contactes directamente con nuestro equipo de soporte para que podamos ver cómo solucionarlo.
- Para cambiar la versión del plugin, es recomendable actualizar y no desinstalar el módulo en PrestaShop. Al desinstalar el plugin, todos los vínculos de ítems entre Sales Layer y Prestashop se pierden, lo que hará con, al instalar de nuevo el plugin, algunas modificaciones (nombradamente el cambio del estado del producto para borrador/invisible) no se reflejarán en prestashop.

Una vez configurado el e-commerce la gestión de información de producto debe hacerse desde Sales Layer, para evitar conflictos si se mantiene a la vez en Sales Layer y en el propio e-commerce. Recomendamos que todas las personas que trabajen con información de producto tengan claro el proceso para un correcto funcionamiento.

## ATENCIÓN

Nuestra API está configurada para **enviar todos los cambios que se realicen en Sales Layer**, aunque no se incluyan dentro de los filtros del conector.

Por ejemplo, si tenemos un conector configurado para exportar **solo los ítems con estado Visible y realizamos cambios sobre productos que estén en Invisible o**

**Borrador, la API devolverá estos cambios sobre los ítems Invisibles y Borradores como "Deleted".**

**Como ejemplo:**

Si el conector está configurado para exportar solo productos visibles y cambias el estado de un producto a Invisible, la API lo enviará a eliminar.

Si en el conector hacemos un filtrado por etiquetas y quitas la etiqueta de un producto que previamente se estaba filtrando, la API lo enviará a eliminar.

# Propósito del conector - Versión

El plugin de Sales Layer para Prestashop se encarga de actualizar la información de producto de una o diversas tiendas de Prestashop desde uno o varios conectores configurados en Sales Layer.

El plugin está disponible para su descarga en:

- El **cloud Sales layer** (en la configuración de canal Prestashop).
- En [GitHUB](#) (para realizar modificaciones personalizadas).

**Versión plugin 1.5.0 (beta)**

**Prestashop: 1.6.1.6 => 1.7.7.2**

**Stable in 1.7.5.1**

## **Nota sobre Prestashop 1.7**

La versión de Prestashop 1.7 por defecto no tiene instalado el funcionamiento de cronjobs. Es posible su activación, mediante la descarga e instalación de un zip disponible en el [GitHUB](#) de Prestashop.

# Instalación

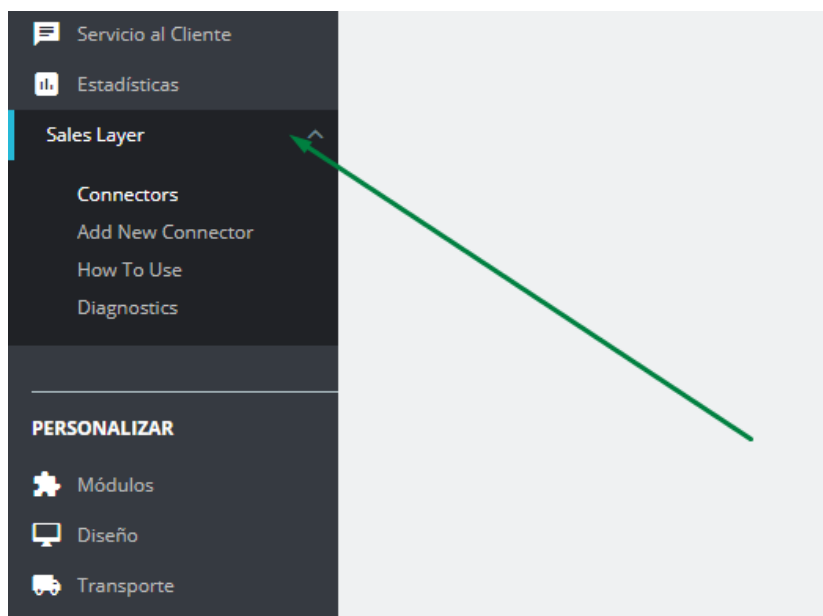
## Atención


Si se ha usado una versión anterior de nuestro plugin, es recomendable desinstalarla con el checkbox para eliminar la carpeta del plugin. Luego se podrá instalar la nueva versión asegurando que todo está correcto.

Es importante tener en cuenta que la actualización del plugin en la práctica puede equivaler a una nueva implementación, con todos los efectos señalados en el punto anterior sobre limitaciones de soporte y necesidad de un implementador especializado.

Desde la página de Admin en la tienda de Prestashop accede al menú izquierdo y elige la opción **Módulos > Module Manager > Subir un Módulo** y carga el zip del plugin descargado.

Si todo ha ido bien, presionando F5 (refrescar página) debería aparecer Sales Layer en el menú izquierdo.



En la opción **How To Use** Podemos encontrar un pequeño manual que nos ayudará a entender y verificar si tenemos todo lo necesario para el correcto funcionamiento del plugin. Si tenemos todos los requisitos necesarios aparecerá el icono (  ).



Todos los elementos marcados son necesarios para el correcto funcionamiento del plugin.

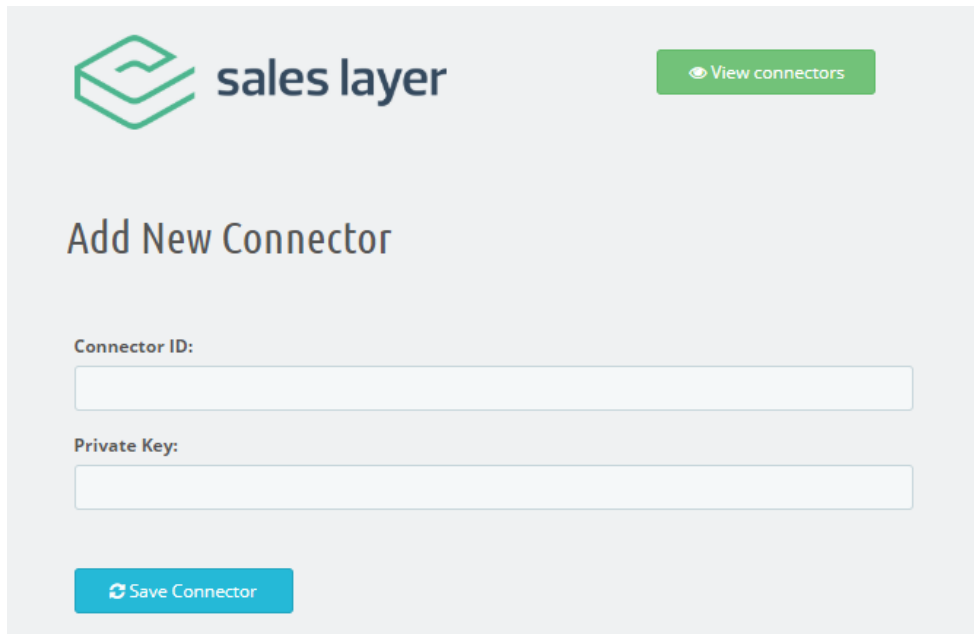
En caso de que “**Registered prestashop cronjob activity**” aparezca marcado como un error, se nos estará indicando que en la última hora no ha sido detectada la actividad de CRON de Prestashop. En tal caso, es posible que haya que acceder al servidor (*vía SSH o cpanel*) tal como se explica en la parte:

#### **HOW TO SYNCHRONIZE BY CRON.**

Si el plugin ya está activo, puede esperarse un tiempo hasta que se ejecute el CRON por lo menos una o dos veces para que se pueda calcular la frecuencia de ejecuciones del CRON y el mensaje de error desaparecerá.

Si se han producido errores, es posible que haya avisos en los archivos de diagnóstico en la pestaña **Diagnostics**. Pueden eliminarse los archivos para que empiece de nuevo con el contenido limpio antes de sincronizar.

## Primeros Pasos:

The screenshot shows the 'Add New Connector' interface in the Sales Layer dashboard. At the top left is the Sales Layer logo, and at the top right is a green button labeled 'View connectors'. The main heading is 'Add New Connector'. Below this, there are two input fields: 'Connector ID:' and 'Private Key:'. At the bottom left is a blue button labeled 'Save Connector' with a circular arrow icon.

Para agregar un nuevo conector debemos acceder a la pestaña **Connectors** y luego hacer click en **+ New connector**.

Para conectar el plugin con Sales Layer sólo hay que copiar y pegar las credenciales del conector configurado en Sales Layer para Prestashop y pulsar el botón **Save connector**.

Si el ID y clave privada son correctas, aparecerá el nuevo conector en la página de conectores.

El plugin necesita, para su funcionamiento automático, que **Prestashop cronjobs** esté **activo** y funcionando. La frecuencia de ejecución de los cronjobs recomendada es de 5 minutos. No obstante, si ya se están usando para otras tareas con un periodo de entre 1-30 minutos, puede mantenerse en esos tiempos. El plugin automáticamente ajusta el tiempo de un proceso para la sincronización, calcula la diferencia entre cada ejecución de cron, y si no tiene demasiada carga de CPU, se llama de nuevo para seguir sincronizando sin detenerse.

En el caso de que el servidor tenga demasiada carga no se autoejecutará, pero continuará con la sincronización cuando de nuevo se ejecute el cron (en segundo plano para que el servidor no sufra demasiada carga por procesos de sincronización; la web seguirá funcionando de forma fluida).

**Atención:** Para que el *Cron* funcione correctamente la URL tendrá que estar activa y disponible. Para ello, hay que tener en cuenta que la tienda deberá ser una tienda activa con el dominio disponible para la llamada. Para confirmar que se ha realizado

adecuadamente, bastará con verificar la URL de la tarea, creada para el plugin de Sales Layer, en el módulo de *Cronjobs* en PrestaShop.

Para tener un conector activo basta con elegir en “Select” la frecuencia con la que queremos que se ejecute. Si el valor supera 24 h, se desbloquea el “Select” de horario preferido para la sincronización (donde podemos configurar si queremos que se ejecute, por ejemplo, a las 3 de la mañana, cuando termine la jornada de los editores, etcétera; se sincronizará con todos los cambios realizados en el cloud desde la última sincronización).

## Sincronización

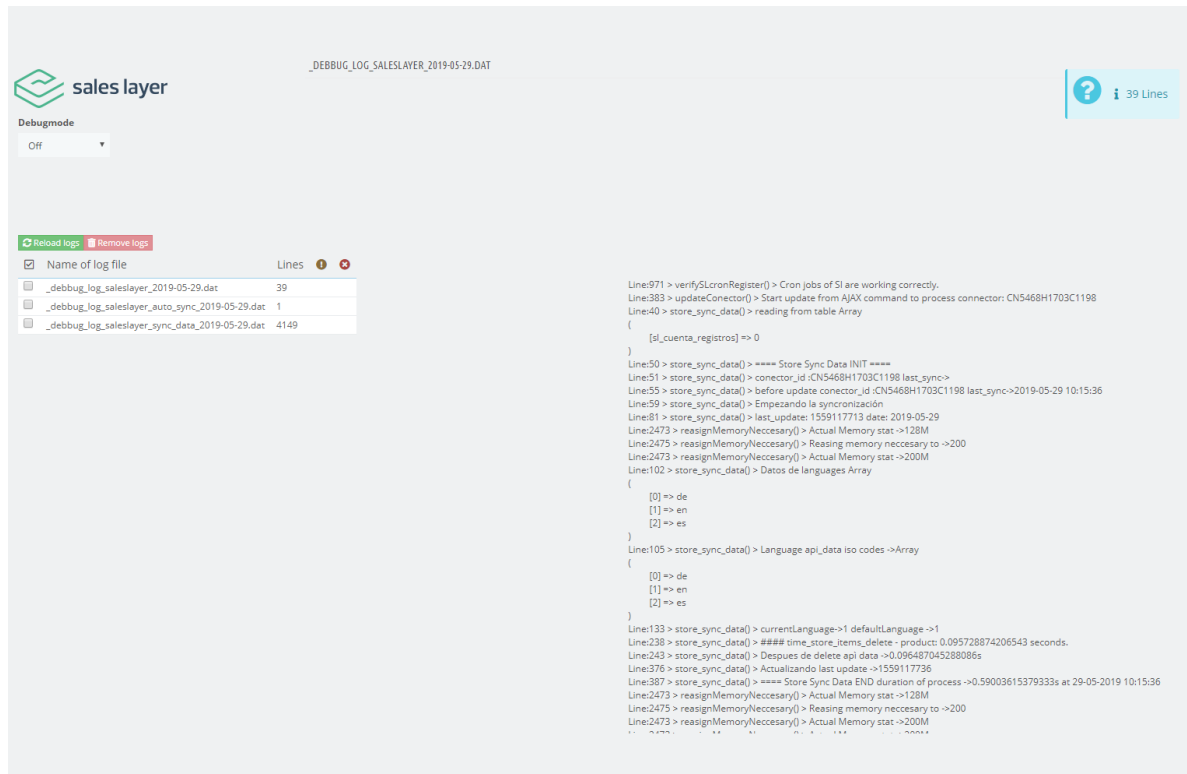
The screenshot displays the Sales Layer management interface. At the top, there's a status bar with the Sales Layer logo, a '+ New connector' button, and a 'Stop synchronization' button. Below this is a progress bar indicating 'Updating products 13% (34/253)'. The main section is titled 'Sales Layer - Update Categories & Products' and includes system resource usage (CPU 0%, Mem 24%, Swp 0%). A table lists the connectors, with the first one being 'Dev Prestashop 1731'. This connector is currently 'OFF'. It shows an 'Identification Code' of CN7116H2734C1622 and a 'Last update' of 2020-01-29 09:27:18. Under 'Shops to synchronize', 'Dev Prestashop 1731' is checked, while 'Secondary shop', 'tienda2', and 'tienda 3' are unchecked. The 'Autosync every' dropdown is set to 'Autosync Off'. The 'Preferred time' is set to '00:00'. There are 'Remove' and 'Download data' buttons for this connector.

Cada vez que el conector recibe nuevas actualizaciones de productos por sincronizar, los almacenará en la base de datos temporalmente y seguirá procesando todo su contenido de forma asíncrona, buscando ocupar la menor cantidad de memoria y procesamiento de la CPU.

Para reconocer si todavía está sincronizando podemos acceder al apartado de **Connectors** y, si todavía tiene registros por procesar, aparecerá la barra de progreso con los elementos totales pendientes y el porcentaje de progreso.

Si el proceso está detenido, puede aparecer un mensaje indicando en cuanto tiempo se ejecutará el CRON para reanudar la sincronización. Si ha ocurrido algún error con algún elemento, intentará sincronizar 2 veces más y creará un mensaje en un archivo de diagnóstico, indicando qué problema ha tenido lugar y, siempre que sea posible, cómo solucionarlo.

## Diagnóstico y solución de errores



The screenshot displays the Sales Layer debug interface. At the top, the title bar reads "\_DEBUG\_LOG\_SALESAYER\_2019-05-29.DAT". Below the title bar, there is a "Debugmode" section with a dropdown menu set to "Off". To the right of the dropdown is a help icon and the text "39 Lines". Below the "Debugmode" section, there are two buttons: "Reload logs" and "Remove logs". Below these buttons is a table with two columns: "Name of log file" and "Lines". The table contains three entries:

| Name of log file                                | Lines |
|---|-------|
| ._debug_log_saleslayer_2019-05-29.dat           | 39    |
| ._debug_log_saleslayer_auto_sync_2019-05-29.dat | 1     |
| ._debug_log_saleslayer_sync_data_2019-05-29.dat | 4149  |

To the right of the table, there is a large text area displaying log output. The log output includes various system messages and data, such as:

```
Line:971 > verifySLcronRegister() > Cron jobs of SI are working correctly.
Line:383 > updateConnector() > Start update from AJAX command to process connector: CN5468H1703C1198
Line:40 > store_sync_data() > reading from table Array
{
  [sl_cuenta_registros] => 0
}
Line:50 > store_sync_data() > Store Sync Data INIT *****
Line:51 > store_sync_data() > connector_id :CN5468H1703C1198 last_sync=>
Line:55 > store_sync_data() > before update connector_id :CN5468H1703C1198 last_sync=>2019-05-29 10:15:36
Line:59 > store_sync_data() > Empezando la sincronización
Line:81 > store_sync_data() > last_update: 1559117713 date: 2019-05-29
Line:2473 > reassignMemory(Necesario) > Actual Memory stat ->128M
Line:2475 > reassignMemory(Necesario) > Reasing memory necessary to ->200
Line:2473 > reassignMemory(Necesario) > Actual Memory stat ->200M
Line:102 > store_sync_data() > Datos de lenguajes Array
{
  [0] => de
  [1] => en
  [2] => es
}
Line:105 > store_sync_data() > Language api_data iso codes ->Array
{
  [0] => de
  [1] => en
  [2] => es
}
Line:133 > store_sync_data() > currentLanguage->1 defaultLanguage ->1
Line:238 > store_sync_data() > ### time_store_items_delete - product: 0.095728874206543 seconds.
Line:243 > store_sync_data() > Despues de delete api data ->0.096487045288086s
Line:376 > store_sync_data() > Actualizando last update ->1559117736
Line:387 > store_sync_data() > Store Sync Data END duration of process ->0.59003615379333s at 29-05-2019 10:15:36
Line:2473 > reassignMemory(Necesario) > Actual Memory stat ->128M
Line:2475 > reassignMemory(Necesario) > Reasing memory necessary to ->200
Line:2473 > reassignMemory(Necesario) > Actual Memory stat ->200M
```

En la opción **Diagnostics**, tenemos disponible una herramienta para listar los archivos de diagnóstico generados por nuestro conector. Inicialmente hay varias opciones a la hora de ejecutar en el modo "debug" en **Off**. En este modo, solo se crean archivos si ocurren errores e imprimirá sólo los textos de debug de un elemento en el que haya ocurrido un error.

En otras opciones de debug (*subiendo de "level"*) aparecerá más información acerca del uso de memoria y CPU. Esta información se puede usar para el desarrollo y solución de problemas. Por defecto, no obstante, la mejor opción es tenerlo en modo **Off** y así ver solo si ocurren errores para no crear archivos de gran tamaño.

Los archivos de diagnóstico se deberían eliminar de forma automática tras transcurrir 15 días. Los errores que aparecen en el archivo estarán marcados con un icono y en fondo rojo. Los ítems con error aparecerán con un mensaje informativo y la referencia identificativa de Sales Layer para poder ir a revisar la información si se requiere. Un ejemplo de mensaje sería:

**## Error. Creating category ID: 91**

Con este mensaje sabemos que se trata de una categoría y que su referencia en Sales Layer es "91".

En caso de que el error no tenga una referencia identificativa es posible que sea porque haya ocurrido en algún punto intermedio del código y, en este caso, lo podremos identificar en el archivo de log general que empieza por:

`_debug_log_saleslayer_sync_data_`

...seguido con la fecha del día de lo ocurrido. En el archivo podremos encontrar más información del elemento en el que ha ocurrido el problema (*pulsando en el icono de error*), podremos saltar de un error a otro y navegar entre los errores (*si se ha producido más de uno*).

Los errores que pudieran encontrarse antes de la sincronización de un ítem, muy probablemente tengan relación con este hecho. Con ello podremos identificar si se trata de una categoría u otro elemento, y qué referencia concreta tiene para revisarlo en Sales Layer.

## Rendimiento

La velocidad de sincronización depende de muchos factores como, por ejemplo, el número de ítems y la cantidad de datos que se necesitan para completar la sincronización, cuántas imágenes hay que subir y cómo de grandes son, la carga de proceso existente en el servidor, la potencia de las bases de datos, etcétera...

El plugin está configurado para que pueda calcular la memoria que puede necesitar durante este proceso. Al sincronizar, el mínimo que se le asigna por defecto son 300MBytes, pero en el caso de que, después de descargar muchos datos, le quede poco espacio de memoria, se auto-asignará más para poder terminar todo el proceso. En modo test, el uso promedio de memoria es de 15MBytes, pero con un límite máximo de seguridad establecido en 300MBytes.

En diferentes pruebas, la sincronización, ha arrojado estos resultados relativos:

categorías : 0.183 - 0.583 segundos  
productos : 0.185 - 1.085 segundos  
variantes : 0.357 - 0.550 segundos

```

pid:30899-mem:12.00-cpu:0.92-time:96-from:[saleslayerimport.php] Line:2068 > update_items() > #### time_sync_stored_category: 0.29174995422363 seconds.
pid:30899-mem:12.00-cpu:0.92-time:96-from:[saleslayerimport.php] Line:2068 > update_items() > #### time_sync_stored_category: 0.1877760887146 seconds.
pid:30899-mem:12.00-cpu:0.92-time:96-from:[saleslayerimport.php] Line:2068 > update_items() > #### time_sync_stored_category: 0.18359899520874 seconds.
pid:30899-mem:14.00-cpu:0.92-time:96-from:[saleslayerimport.php] Line:2106 > update_items() > #### time_sync_stored_product: 0.18540215492249 seconds.
pid:30899-mem:14.00-cpu:0.92-time:96-from:[saleslayerimport.php] Line:2154 > update_items() > #### time_sync_stored_product_accessories: 0.0042471885681152 seconds.
pid:30899-mem:14.00-cpu:0.93-time:97-from:[saleslayerimport.php] Line:2133 > update_items() > #### time_sync_stored_product_format: 0.5506329536438 seconds.
pid:30899-mem:14.00-cpu:0.93-time:98-from:[saleslayerimport.php] Line:2133 > update_items() > #### time_sync_stored_product_format: 0.55842781066895 seconds.
pid:30899-mem:14.00-cpu:0.93-time:98-from:[saleslayerimport.php] Line:2133 > update_items() > #### time_sync_stored_product_format: 0.35768008232117 seconds.

```

Para poder ver los tiempos hay que acceder al plugin en *modo debug* y localizar el archivo generado que empieza por `_debug_log_saleslayer_timers_{fecha del día}`. Ahí es posible ver los tiempos que ha durado cada ítem en procesarse.

## Configuración: otros parámetros

### Exportando en multi-tienda

Por medio de nuestro *plugin* de Prestashop es posible exportar, a través de un mismo conector, a varias tiendas de un mismo Prestashop. Para ello, al crear el conector en Prestashop, seleccionaremos en *Shops to synchronize* las tiendas que intervienen:

| Connection Details   | Shops to synchronize   | Autosync every                                 | Preferred time              | Overwrite stock status   | Remove connector       | Store data now                |
|--|--|--|-----------------------------|--------------------------|------------------------|-------------------------------|
| <b>OFF</b><br>Identification Code: CN12254H4936C4486<br>Last update: 2020-10-15 09:56:20 | <input checked="" type="checkbox"/> Sales Layer - Store 1<br><input checked="" type="checkbox"/> Sales Layer - Store 2 | Last sync: 2020-10-15 09:56:20<br>Autosync Off | Server time: 10:57<br>00:00 | <input type="checkbox"/> | <a href="#">Remove</a> | <a href="#">Download data</a> |

Otra característica posible es la configuración de diferentes conectores para diferentes tiendas con la posibilidad de enviar información distinta dependiendo de la tienda:

| Connection Details   | Shops to synchronize  | Autosync every                                 | Preferred time              | Overwrite stock status   | Remove connector       | Store data now                |
|--|---|--|-----------------------------|--------------------------|------------------------|-------------------------------|
| <b>OFF</b><br>Identification Code: CN12254H4936C4486<br>Last update: 2020-10-15 09:56:20 | <input checked="" type="checkbox"/> Sales Layer - Store 1<br><input type="checkbox"/> Sales Layer - Store 2 | Last sync: 2020-10-15 09:56:20<br>Autosync Off | Server time: 11:00<br>00:00 | <input type="checkbox"/> | <a href="#">Remove</a> | <a href="#">Download data</a> |
| <b>OFF</b><br>Identification Code: CN12259H2639C4486<br>Last update: 2020-10-15 09:57:28 | <input type="checkbox"/> Sales Layer - Store 1<br><input checked="" type="checkbox"/> Sales Layer - Store 2 | Last sync: 2020-10-15 09:57:28<br>Autosync Off | Server time: 11:00<br>02:00 | <input type="checkbox"/> | <a href="#">Remove</a> | <a href="#">Download data</a> |

Como se puede ver en la imagen el primer conector está conectado con la tienda 1 y el segundo conector con la 2.

Para este último caso tenemos que tener en cuenta los siguientes parámetros:

- si un producto se sincroniza para una tienda, aunque haya sido creado para todas, será generado como activo solo para la tienda generada.

- para desactivar en todas las tiendas una categoría afectada por varios conectores tenemos que enviarla con el estado “Desactivado” en el total de conectores.
- al enviar a través de un conector un producto como desactivado hay que desactivarlo en las tiendas afectadas por el mismo.

## Exportar títulos multiidioma

Entre las novedades que han sido agregadas en la actual versión se encuentra la funcionalidad de exportar títulos multi-idioma. Para utilizarla tenemos que tener activado **Campos multiidioma** en Sales Layer y configurarlos en el conector de sincronización.

En el conector de Prestashop en Sales Layer nos deberá aparecer la opción “**Exportar títulos multi-idioma**”:

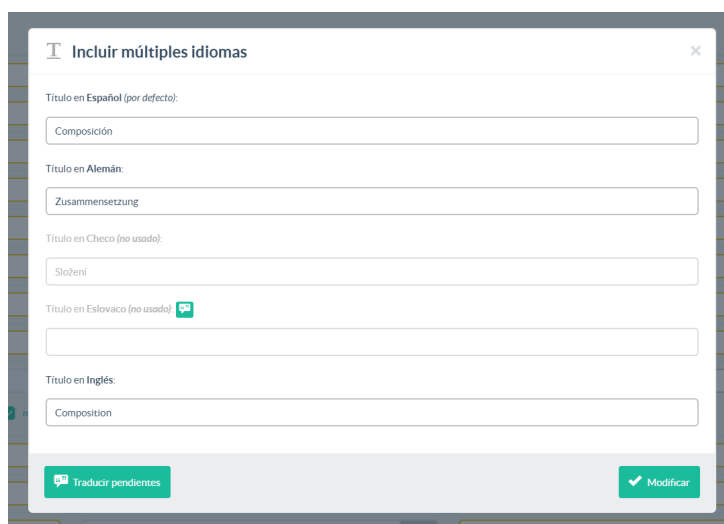
The screenshot shows the configuration interface for the Prestashop connector. At the top, there is a language selection dropdown set to 'Inglés'. Below it, there are two dropdown menus: 'Incluir las categorías vacías:' and 'Exportar títulos multiidioma:'. Both are currently set to 'No'. The 'Exportar títulos multiidioma:' dropdown is highlighted with a red rectangular box. There are also some partially visible labels like 'cios de los otros idiomas.' and 'és de él.'

*(Si no aparece esa funcionalidad en la cuenta de Sales Layer, hay que contactar con el equipo de soporte para pedir su activación)*

Después de activar “Exportar títulos multi-idioma” con la opción “**SÍ**”, hay que ir a la pestaña “**Datos De salida**” y comprobar que cada campo tiene el título en varios idiomas. Desde esa misma opción podemos editar los diferentes títulos si es necesario:

The screenshot shows the 'Datos De salida' (Output Data) tab. It displays a table of output fields. The first column shows the field name, and the second column shows the field value. The fields are: 'product\_supplier\_1', 'product\_supplier\_reference\_1', and 'Composición'. The 'Composición' field is highlighted with a red arrow. The table also includes a 'Fórmula' (Formula) column with icons for each field.

Al hacer click sobre el botón de los títulos traducidos, nos permite editar los títulos que deseamos para cada idioma:

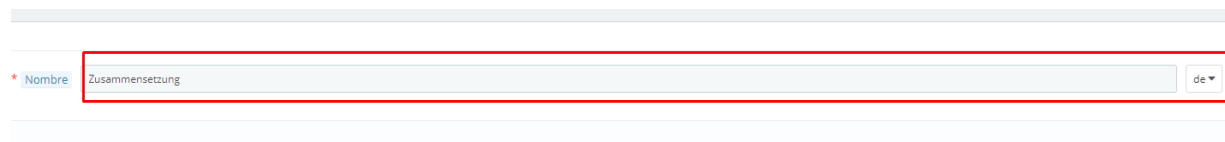


The modal window is titled "Incluir múltiples idiomas" and contains the following fields:

- Título en Español (por defecto): Composición
- Título en Alemán: Zusammensetzung
- Título en Checo (no usado): Složení
- Título en Eslovaco (no usado):
- Título en Inglés: Composition

At the bottom, there are two buttons: "Traducir pendientes" and "Modificar".

Durante la sincronización se asignan esos nombres al atributo o característica a la que pertenecen. Después de la sincronización podremos ver que los valores se crean en nuestra tienda de Prestashop:



The image shows a product attribute field in Prestashop. The field is labeled "Nombre" and contains the value "Zusammensetzung". The field is highlighted with a red border.

## Creación de características del producto y atributos de variante

El conector de Sales Layer permite configurar la exportación de forma que todos los campos adicionales que se agreguen en productos, se conviertan en características del producto, y que todos los campos adicionales agregados a la plantilla de variantes, se conviertan en atributos configurables de las variantes. **Si el valor está nulo o vacío, se eliminará del producto o variante en su tienda.**

Los nombres primarios de los campos configurados en el conector de Sales Layer se buscarán en Prestashop y, de no existir, se crearán como nuevos:



A partir de la versión 1.7 del Prestashop, hay posibilidad enviar features con más de un valor. Para separarlos puede usarse el símbolo "|". Para crear las características de tipo personalizadas, se puede añadir al final del valor del campo ":custom", con una fórmula o en el valor del formulario. Para crear todas las características de la misma manera, sin tener que enviar un comando custom, es posible editar el archivo saleslayerimport.php y editar la línea que dice:

```
public $create_new_features_as_custom = false;
```

y cambiar el valor a true.

- Las características, atributos y sus valores en multiidioma, se crean solo si no existen en la tienda.
- Si un valor en alguno de los idiomas ya existe, se vincula con el producto y rellenan los idiomas faltantes. Si ninguno de los valores ha sido encontrado en la tienda en alguno de los idiomas, se crea como nuevo.
- Una vez que se crean los atributos y se rellenan los valores multiidioma, no será posible actualizarlos en el futuro desde el plugin. El usuario debe ser consciente de esta casuística para tenerlo en cuenta. Este comportamiento es debido a la necesidad de proteger los valores existentes en PrestaShop. Desde el plugin nunca se actualizan valores multiidioma ya existentes.
- Es posible editar los idiomas desde Prestashop o eliminarlos, así en la siguiente se crearán los datos correctamente.

Sales Layer, por defecto, siempre genera nuevos atributos en su tienda. En el caso de no querer que se generen, ofrecemos un flag en el archivo saleslayerimport.php donde simplemente modificando el valor de la variable, a false, el sistema omitirá su creación.

```
public $create_new_attributes = true; a false;
```

A partir de ese momento, el sistema ofrece en su sistema de logs, información indicando que el sistema tiene desactivada esa opción.

## Creación de Campos personalizados

Para crear campos personalizados, se puede utilizar el campo **Customizable** en la configuración del conector en el cloud de Sales Layer. El plugin reconoce comandos

internos que es posible añadir detrás del nombre de un campo, como por ejemplo: `"Nombre_campo:file"` o `"Nombre_campo:archivo"`.

Si queremos que un campo sea obligatorio hay que utilizar los siguientes comandos: `"Nombre_campo:requerido"` o bien `"Nombre_campo:required"`.

Los campos se pueden crear como texto separado por comas, lista de atributos o bien rellenando con una fórmula un texto separado por coma. Si deseamos enviarlo en varias idiomas debemos elegir el campo de texto.

Veamos unos ejemplos:

- Al enviar desde Sales Layer el campo `"Nombre_campo:file"` en Prestashop será un campo para subir imágenes.
- Al enviar desde Sales Layer el campo `"Nombre_campo:required"` en Prestashop será un campo obligatorio para el cliente.
- Al enviar desde Sales Layer el campo `"Nombre_campo:file:required"` en Prestashop será un campo para subir imágenes obligatorio para el cliente.



## Colores personalizados

El plugin contiene desde la versión 1.4.3 una carpeta llamada **colors** que incluye una pequeña base de datos de colores que reconoce para cada idioma. Esta carpeta, se usa para detectar atributos de colores inexistentes en Prestashop para las variantes.

Esto implica que si desde Sales layer viene una variante con el color **verde claro** en español, el plugin intentará encontrar el archivo **es.txt**, eliminará acentos y espacios, y buscará el color **verde claro** para asignarle el color **#90ee90** que está predefinido en el archivo. Estos archivos se pueden personalizar según sus necesidades. La razón es que cada empresa usa distintos nombres, diferentes mapas de colores en varios idiomas

para reconocer los colores. Esta opción permite tener siempre el archivo y con cada actualización, simplemente copia y pega sus archivos de colores en el plugin. Es suficiente tener un archivo en el idioma que vamos a usar para reconocer colores y asignarlos la primera vez que se cree el atributo.

Otra forma de enviar colores, es enviar el color como texto del siguiente modo: **verdeclaro:#90ee90**. El plugin acepta todo antes de **:#** como nombre y todo después de **:"** como color **#90ee90**.

#### **Versión: 1.4.16**

Desde la versión 1.4.16 el sistema admite varios códigos de color para un mismo nombre.

Un ejemplo sería tener 20 colores que se llamarán blanco, cada uno con distinto código hex de color. En este caso, los códigos hex, formarán parte de la búsqueda en la sincronización para identificar el valor buscado. De manera, que si no existe el par (nombre, código) este será añadido en PrestaShop. Esta versión es totalmente compatible con la anterior.

## **Tags**

Desde la versión del plugin 1.4.7, ha sido agregada la posibilidad de gestionar los tags de productos. Si no desea que el conector edite o elimine los Tags existentes, tiene la posibilidad de deshabilitar el campo **Tags** en la configuración del conector.

## **Adjuntar archivos**

Adjuntar archivos **"File attachment"** está disponible desde la versión 1.4.7 con el nombre del campo **File attachment** en la configuración del conector en el cloud de sales layer.

### **Importante**

Si no se desea que el conector edite o elimine archivos existentes que se encuentren en Prestashop vinculados con el producto, hay que desactivar este campo en el conector. Si no se desea gestionarlos desde el plugin, en el momento en que el conector detecte que el archivo no se envía desde Sales Layer, se procederá a la eliminación del archivo del producto en la tienda de Prestashop. Además, el campo no soporta multi-idioma, solo archivos adjuntos que se asignan al producto.

## Precios

Existen diferentes opciones para gestionar la sincronización de precios en función de los campos enviados:

**Retail price:** Si proporcionamos información de este campo, automáticamente se guardará en prestashop sin ningún tipo de recálculo.

**Retail price with tax:** Al enviar valores en este campo, el proceso recuperará de PrestaShop, la actual regla en vigor, se descontará el impuesto (siempre que exista) y se guardará en la base de datos como **retail price**. Si no posee ninguna regla configurada para el cálculo del impuesto, se asignará el mismo valor para **retail price**.

**Tax rule:** este campo puede utilizarse para sincronizar el porcentaje de impuesto aplicado mediante dos formas:

- **Indicando el identificador "id\_tax\_rules\_group" de PrestaShop.** Si por defecto el plugin recibe un valor numérico, buscará en la tabla interna correspondiente el identificador de la regla correspondiente. Si no lo encuentra, asignará el valor establecido por defecto en la tienda.  
De cara a la integridad de la información, este sería el **procedimiento más adecuado y recomendado** para implementarlo.
- **Indicando directamente el % de impuesto a aplicar.** Para que el plugin funcione de esta manera es necesario añadir el sufijo "%" al campo de Sales Layer a exportar. Bien de forma manual, o bien mediante una fórmula sobre el campo a exportar del conector. (CONCAT({THIS}, "%")).  
Se permite este funcionamiento, pero es **muy importante** tener en cuenta que la asignación se realizará sobre el **primer registro que encuentre** para reglas duplicadas con el mismo %.

**wholesale\_price:** equivale al precio al por mayor en PrestaShop, si no existe el dato, se auto rellenará con el **retail price** de forma automática.

**unit\_price\_ratio:** es la relación de precio unitario por un producto.

**unity:** Es el texto que indica la unidad en que está marcado el precio **unit\_price\_ratio**. Por ejemplo (Kg,Ud,...).

**additional\_shipping\_cost** El precio adicional por transporte.

## Gestión de descuentos

En la versión 1.4.7 ha sido mejorada la gestión de descuentos para productos.

### Importante

Si no se desea que el plugin elimine descuentos actuales de Prestashop, se pueden desactivar los campos **product\_discount\_1** y **2** en el cloud de sales layer.

Hay varias posibilidades para crear descuentos, con un máximo de 2 descuentos por producto. Si se desea crear descuentos para todas las tiendas desde un conector, es posible adjuntar al campo **product\_discount\_1\_type** o **product\_discount\_2\_type** el comando **:all** o **:global** que el plugin reconoce para crear descuento para todas las tiendas. Por ejemplo : **percentage:all** o **%:global**.

Si vas a usar descuentos para todas las tiendas, el comando formateará todos los descuentos de todas la tiendas, dejará solo los descuentos que recibe del conector y los creará para todas las tiendas. Si va a usar descuentos desde un conector con el comando **:all**, hay que deshabilitar los campos de descuentos en otros conectores, ya que cada conector podría crear descuentos que el conector con el comando **:all** eliminaría.

No hay que olvidar que si se necesita tener descuentos diferentes por cada tienda, no se puede usar comando **all** o **global** en ninguno de los conectores. El comando **all** solo debe ser usado para gestionar descuentos en todas las tiendas desde un conector.

## Valores permitidos por los campos de descuentos

**product\_discount\_1** -> numérico 1-100 si el campo **product\_discount\_1\_type** está configurado como descuento por porcentaje. Para eliminar todos los descuentos hay que enviar esta línea vacía. Para no editar los descuentos, desactiva estos campos desde el conector en el cloud.

Si el campo **product\_discount\_1\_type** está configurado por cantidad, puede poner la cantidad que se debe descontar en el campo **product\_discount\_1**.

**product\_discount\_1\_type** -> Permitidos para cantidad (\$, euro, €, dollar, amount, importe)

Permitidos para porcentaje (% , porcentaje, percentage)  
Si no tiene valor, por defecto es cantidad.

Ejemplo para adjuntar el comando all -> **porcentaje:all**

**product\_discount\_1\_quantity** -> Número a partir de cuantas unidades es aplicable el descuento. Si no está relleno, el valor por defecto es 1.

**product\_discount\_1\_from** -> Fecha de inicio de descuento.

**product\_discount\_1\_to** -> Fecha de fin de descuento

## Imágenes

Para evitar imágenes duplicadas, recomendamos el uso de un mismo tamaño de imágenes para productos y variantes. De esta forma, el plugin puede reconocer que se trata de una misma imagen y no subirá dos imágenes idénticas con distinto tamaño. Por lo tanto, y como ejemplo, si elegimos en el campo imagen de productos el tamaño **ORG**, deberemos de elegir este mismo tamaño en las imágenes de variantes.

Si enviamos varias imágenes, hay que tener en cuenta que en PrestaShop, **la primera imagen que reciba** será la marcada como la **portada del producto** (cover).

## Atributos alt

**product\_alt** -> Atributos alt para imágenes de producto.

**format\_alt** -> Atributos alt para imágenes de variantes.

Para poder asignar un alt para los campos imagen tenemos dos posibilidades:

- Crear un campo de tipo texto con los atributos separados por comas, de forma que el orden de los atributos coincida con el orden de las imágenes.
- Crear un campo tipo tabla con una columna, donde cada línea, contenga el atributo a asignar a la imagen que le corresponde por orden.

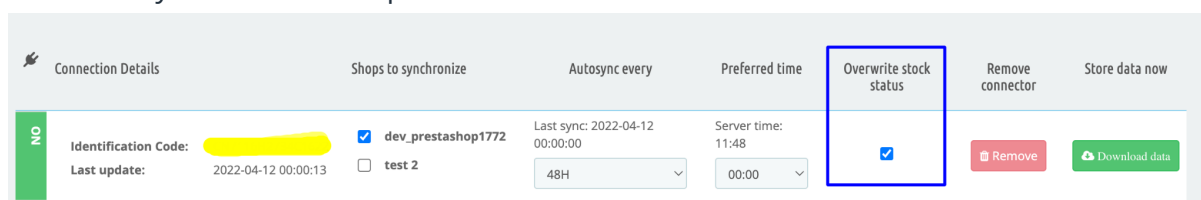
Si este campo lo enviamos vacío, se autorellena para los productos con el nombre en el idioma correspondiente y un número incremental para cada imagen. En el caso de las variantes, se rellena con la referencia y un número incremental.

Hay que tener en cuenta, que si hablamos de una tienda multiidioma, es necesario rellenar toda la información en los idiomas correspondientes; si no se hace, el sistema de forma automática, asignará el nombre de producto / variante al alt de las imágenes no rellenas.

## Stock

Como comentamos en varios puntos de nuestra documentación, Sales Layer no ha sido creado con el objetivo de trabajar con stock. Sin embargo, a través del conector de PrestaShop es posible enviar la cantidad de ítems almacenados. Para ello, en la pestaña de Datos de salida de Productos, tendríamos que mapear el campo *Quantity* con el correspondiente, de igual manera que lo haremos en la tabla de Variantes.

Para hacer el envío de forma óptima es indispensable que el *checkbox* Overwrite Stock Status que vemos a continuación, dentro de la configuración del conector en el plugin de Sales Layer en PrestaShop esté activo.



The screenshot shows the configuration interface for the Sales Layer connector in PrestaShop. The interface is divided into several sections: Connection Details, Shops to synchronize, Autosync every, Preferred time, Overwrite stock status, Remove connector, and Store data now. The 'Overwrite stock status' section contains a checkbox that is checked, indicating that the stock status will be overwritten. This checkbox is highlighted with a blue box. Below the checkbox, there are buttons for 'Remove' and 'Download data'.

**Atento:** si el checkbox Overwrite Stock Status no está seleccionado, Sales Layer solo actualizará el stock cuando cree el artículo. Es decir, en las futuras actualizaciones el plugin no cambiará el valor aunque sea diferente.

Si no deseas gestionar el stock desde Sales Layer deberás deshabilitar el campo *Quantity* dentro del conector en Sales Layer, seleccionándolo y yendo a *Acciones*→*Deshabilitar campo*, por lo que no habrá que tener seleccionado el checkbox Overwrite Stock Status.

## Transporte

Para elegir un transporte para el producto, debe crearse en Prestashop y desde Sales Layer enviar su **nombre** o **id\_reference** en el campo **Carrier** de la pestaña de productos. Para evitar la eliminación de transportes existentes es necesario deshabilitar el campo en Sales Layer; de lo contrario si un producto no tiene asignado un transportista, al sincronizar automáticamente, será eliminado del producto en cuestión.

## Productos tipo pack

En Sales Layer, para poder generar un pack de productos, es necesario seleccionar como **tipo** de producto el valor **pack**. Para agregar a dicho pack los productos y/o variantes asociados, simplemente, es necesario introducir el número de campos necesarios según la nomenclatura (**pack\_product\_x**, **pack\_format\_x** y/o **pack\_quantity\_x**) en la pestaña de productos de la configuración del conector, siendo x cualquier identificador único que relacione el producto, su variante y la cantidad.

Es importante tener presente que es posible definir el número de campos que sean necesarios, siempre con el prefijo adecuado, siendo imprescindible tener definido un `pack_product` o `pack_format`. El campo **pack\_quantity** solo es necesario si necesitamos más de una unidad por producto / variante en el pack.

El sistema en su sincronización, trabaja a nivel jerárquico, con la siguientes premisas contempladas: Los datos de una variante prevalecen sobre los del producto; es decir, si tenemos asociado un producto pero no su variante al pack en Prestashop, se vinculará la cantidad de unidades al producto. Si tiene variante, tenga o no producto, se vinculará la cantidad de unidades a la variante.

Ejemplo:

*Envío:*

```
pack_product_1=>"SKU789"  
pack_format_1 =>""  
pack_quantity_1=>"3"
```

*Resultado:*

*3 unidades del producto SKU789*

*Envío:*

```
pack_product_2=>""  
pack_format_2 =>"SKU445-FORMAT589"  
pack_quantity_2=>"2"
```

*Resultado:*

*2 unidades de la variante SKU445-FORMAT589*

*Envío:*

```
pack_product_3=>"SKU457"  
pack_format_3 =>"SKU457-FORMAT589"  
pack_quantity_3=>"1"
```

*Resultado:*

*1 unidad de la variante SKU457-FORMAT589*

Para rellenar el campo **pack\_product\_x** o **pack\_format\_x** en el producto, y que funcione como tipo pack, lo más óptimo para este caso, es usar campos de tipo **Ítems relacionados** configurados como tabla de productos. De esta manera, se facilita la búsqueda de los productos y variantes a la hora de enlazar los productos y/o variantes que pertenecen al pack. Así, al exportar en Prestashop, la sincronización asignará al producto "pack" los productos / variantes asociados mediante su referencia. Por ejemplo: al Producto Pack "Comedor" se le asignarán los productos "Mesa" y "Sillas" con sus cantidades.



## Proveedores

La gestión de proveedores está disponible a través de los campos que se crean o rellenan en la pestaña de productos en la configuración del conector. Siguiendo la nomenclatura habitual descrita en este mismo documento, es posible generar los proveedores que sean necesarios siempre con el prefijo **product\_supplier\_x** para el nombre y **product\_supplier\_reference\_x** para la referencia del proveedor.

En la sincronización se puede enviar su **id\_supplier** o el nombre de proveedor en el campo **product\_supplier\_x** para vincularlo con el producto. Si la referencia no existe, automáticamente se agrega al proveedor en Prestashop. De esta misma manera, es posible crear proveedores en la pestaña de variantes utilizando la nomenclatura **format\_supplier\_x**.

Desde la versión 1.4.16 de plugin:

Cuando el producto tiene varios proveedores, Sales Layer ha añadido la posibilidad de asignar al proveedor por defecto. Para ello, en el campo **product\_supplier** del proveedor, por defecto añadiremos el sufijo **:default** en el nombre o ID que enviamos en el campo **product\_supplier**. Si no elegimos un proveedor por defecto, se elegirá siempre el primer proveedor.

## Fabricantes

A través del conector, es posible gestionar los diferentes fabricantes de los productos. Para ello, basta rellenar el campo "Manufacturer", presente en el conector, con un campo que contenga el nombre del fabricante del ítem.

Si quieres añadir el valor a través del ID del fabricante ya creado en PrestaShop, basta con añadir al mismo texto que se va a enviar el sufijo ":ID".

## Tipo de Producto y Visibilidad

A través del conector es posible mapear el tipo del producto y la visibilidad de este en la tienda.

Para mapear el campo del tipo del producto, basta rellenar el campo "Type", presente en el conector, con un campo que contenga una de las siguientes palabras:

- *virtual*, en el caso de que sea un producto virtual
- *simple*, para productos simples
- *pack*, para productos de tipo pack.

Atención: En el caso de que se envíe el producto de tipo como *pack*, sin los campos **pack\_product\_x**, **pack\_format\_x** y/o **pack\_quantity\_x** rellenos, será creado como un producto simple.

|        |      |      |         |
|--------|------|------|---------|
| Normal | Type | Type | formula |
|--------|------|------|---------|

Para mapear el campo de Visibilidad de producto en la tienda, basta rellenar el campo "Visibility" con las siguientes palabras:

- *both*, para *Toda la tienda*
- *search*, para *Sólo resultados de búsqueda*
- *catalog*, para *Sólo catálogo*
- *none*, para *Oculto*

|        |            |            |         |
|--------|------------|------------|---------|
| Normal | Visibility | Visibility | formula |
|--------|------------|------------|---------|

## Visible en

¿Dónde deseas que aparezca el producto?

- ✓ toda la tienda
- Sólo catálogo
- Sólo resultados de búsqueda
- Oculto

ende en

## Redireccionamiento de Productos

Es posible redireccionar nuestros ítems, cuando estos se encuentran desactivados en la tienda, creando los siguientes campos en el conector:

|        |                  |                  |         |
|--------|------------------|------------------|---------|
| Normal | redirect_type    | Redirect Page    | formula |
| Normal | id_type_redirect | ID Redirect Page | formula |

El campo *redirect\_type* mapea el campo "Redirección cuando el producto está desactivado", y el campo *id\_type\_redirect* casa el campo del ID de destino.

## Página de redirección

Redirección cuando el producto está desactivado

Redirección temporal a una categoría (302) 

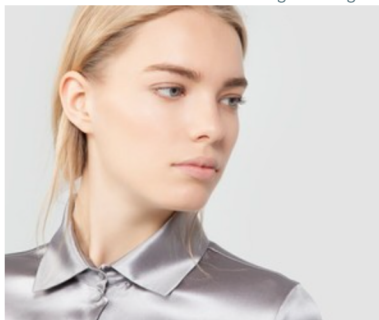
Categoría destino

¿A qué categoría debe redirigir la página?



Si no se selecciona ninguna categoría, se utiliza la categoría principal

Inicio > Mujer 








Para mapear el campo, `redirect_type`, basta rellenar con un campo que contenga las siguientes palabras:

- `301-product`, para “Redirección permanente a un producto”
- `302-product`, para “Redirección temporal a un producto”
- `301-category`, para “Redirección permanente a una categoría”
- `302-category`, para “Redirección temporal a una categoría”

Para mapear el campo `id_type_redirect` tenemos que enviar el ID interno del ítem en Prestashop.

## Productos Relacionados

Para enviar el campo de productos relacionados de PrestaShop basta rellenar el campo *Accessories*, en la tabla de productos, con un campo de tipo “Ítem Relacionado”.

|   |   |  |             |  |   |
|---|---|--|-------------|--|---|
|  |  | Normal  | Accessories | Productos Relacionados  |  formula |
|---|---|--|-------------|--|---|

### Producto relacionado

Buscar y añadir un producto relacionado



Central Park (ref:AAPA10H406401)



## Otros campos disponibles

### Productos

**product\_show\_price:** Mostrar precio. Valores permitidos: ('true', '1', 'yes', 'si', 'false', '0', 'no')

**product\_available\_for\_order :** Producto disponible para la venta. Valores permitidos: ('true', '1', 'yes', 'si', 'false', '0', 'no')

**product\_available\_online\_only :** Producto disponible solo online. Valores permitidos: ('true', '1', 'yes', 'si', 'false', '0', 'no')

**low\_stock\_threshold:** Bajo nivel de existencias. Valores permitidos: (INT) 0 - 10+

**product\_available\_date:** Fecha de disponibilidad de producto. Fecha en cualquier formato.

**product\_condition:** Condición del producto. Valores permitidos: 'new', 'refurbished' y 'used'.

### Variantes

**available\_date:** Fecha de disponibilidad de la variante. Fecha en cualquier formato.

**default\_on:** Definición de variante por defecto. Valores permitidos: ('true', '1', 'yes', 'si', 'false', '0', 'no')

**low\_stock\_threshold:** Bajo nivel de existencias. Valores permitidos: (INT) 0 - 10+

**ecotax:** (float) (default: 0.000000)

## Personalización de campos

Si tenemos PrestaShop personalizado y necesitamos gestionar campos adicionales en el conector propios de nuestro e-commerce, para que la información sea actualizada debemos agregarlos en el código, concretamente en el fichero ***saleslayerimport.php*** en la variable ***\$predefined\_product\_fields*** para productos. De esta forma, el sistema recorre todos los campos que se agregan en pestaña de productos del PIM, los busca en esta variable (siendo la búsqueda *case sensitive*) y si no los encuentra, los convierte en características del producto. De la misma manera, para las variantes hay que agregarlos en la variable ***\$product\_format\_base\_fields***.

De forma avanzada, si es requerida una mayor personalización del plugin, para realizar otras acciones propias del e-commerce, ofrecemos dos ficheros de personalización para productos, ***SIProducts.php*** y para variantes ***SIVariants.php***.

## Ejecución de indexadores

Al comiendo de la sincronización se detienen los indexadores. Cada elemento actualizado se reindexa automáticamente, y al final de toda la sincronización, restaura

la configuración de los indexadores. Si hubiese la necesidad de agregar llamadas extra o código para que sea ejecutado después de cada sincronización, el lugar indicado para ello es el archivo **saleslayerimport.php**; concretamente en la función **callIndexer** al final de la función, usando la función **urlSendCustomJson** que será llamada cuando se termine la sincronización.

Por ejemplo:

```
$mi_another_indexer = 'https://mistore.io/mainindexer/token/58161dcdf1a6c1a61cs1a56';  
$this->urlSendCustomJson('GET', $mi_another_indexer, null, false);
```

```
/**  
 * Call to indexers reindex all  
 * @throws PrestaShopException  
 */  
  
private function callIndexer()  
{  
    /**  
     * Deprecated! now indexes immediately with synchronization  
     */  
    /**  
     * This is code for reindex all.  
     * But it makes too much use for the cpu in version 1.7.6.0 version is deprecated.  
     */  
    $admin_folder = $this->getConfiguration('ADMIN_DIR');  
    $contextShopID = Shop::getContextShopID();  
    Shop::setContext(Shop::CONTEXT_ALL);  
    $default_shop = new Shop(Configuration::get('PS_SHOP_DEFAULT'));  
    $adminurl = 'http://' . $default_shop->domain . $default_shop->getBaseURI() .  
        $admin_folder . '/searchcron.php?full=1&token=' .  
        Tools::substr(  
            _COOKIE_KEY_,  
            34,  
            8  
        );  
    Shop::setContext(Shop::CONTEXT_SHOP, $contextShopID);  
    $this->debug('Calling indexer to start reindex all', 'syncdata');  
    $this->urlSendCustomJson('GET', $adminurl, null, false);  
    /**  
     * If you need to execute something after synchronization add the url here and uncomment the script  
     */  
    $url_for_run = 'Place here one url for run after sync';  
    $this->urlSendCustomJson( type: 'GET', $url_for_run, json: null, wait_for_response: false);  
}
```

## Exportación de imágenes de productos desde PrestaShop a Sales Layer

Es común encontrarse con la necesidad de realizar una carga de los datos desde PrestaShop a Sales Layer. En esos casos es frecuente encontrarse con un problema común a propósito de una peculiaridad de PrestaShop, como es el sistema de nomenclatura de las imágenes. Así, cuando se sube una imagen a PrestaShop, este la almacena mediante un sistema de identificación por carpetas y utilizando un nombre único por imagen.

*Por ejemplo:*

Para productos:

`http://dominio/img/p/1/2/3/4/1234.jpg`

`http://dominio/img/p/1/2/3/5/1235.jpg`

Para categorías:

`http://dominio/img/c/1/2/3/4/1234.jpg`

`http://dominio/img/c/1/2/3/5/1235.jpg`

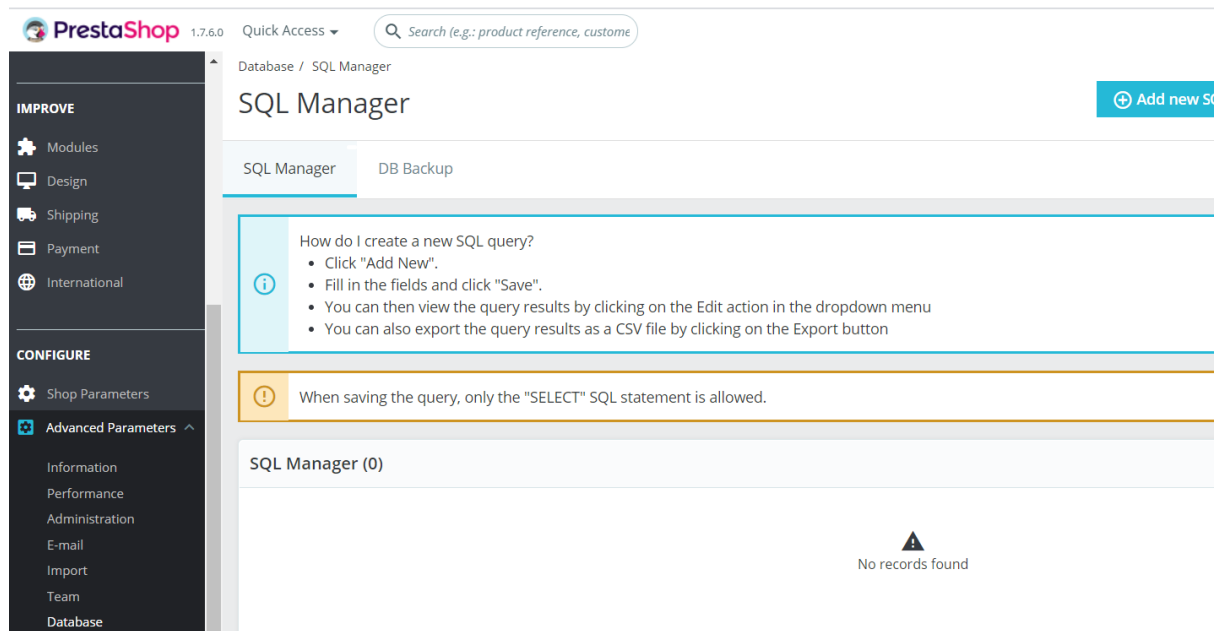
Entre tanto, cuando accedes a las mismas imágenes en la tienda, PrestaShop sustituye el identificador de la imagen por el nombre del producto / categoría de este modo:

`http://dominio/1234-prod_default/producto.jpg`

`http://dominio/1235-prod_default/producto.jpg`

Es por este motivo que cuando se exporta de PrestaShop la URL de la imagen se extrae siguiendo el segundo caso. El problema es que cuando se importa en Sales Layer, como el nombre de la imagen no puede estar duplicado, solo sube una de las repetidas.

Para solucionarlo, hay que acudir al administrador de PrestaShop, menú Configurar—>Parámetros Avanzados—>Base de datos->Añadir nueva consulta SQL:



Ahí tendremos que insertar una consulta para generar la exportación a formato CSV, que después podemos cargar en Sales Layer (siguiendo los pasos habituales de carga

de CSV en Sales Layer). Como ejemplo de la consulta, utilizando un dominio falso de ejemplo ("yourdomain") tendríamos:

```
SELECT aux.reference,
       Group_concat(aux.img SEPARATOR ',') AS imagenes
FROM   (SELECT p.id_product,
              p.reference,
              CASE
                WHEN Length(im.`id_image`) = 6 THEN
                  Concat('https://yourdomain.com', 'img/p/', INSERT(
                    INSERT(
                      INSERT(INSERT(im.`id_image`, 2, 0, '/'), 4, 0, '/'), 6, 0, '/'), 8, 0, '/'), 10, 0, '/'), '/',
                    im.`id_image`, '.jpg')
                WHEN Length(im.`id_image`) = 5 THEN
                  Concat('https://yourdomain.com',
                        'img/p/',
                        INSERT(INSERT(INSERT(INSERT(im.`id_image`, 2, 0, '/'), 4, 0, '/'), 6, 0, '/'), 8, 0, '/'), '/', im.`id_image`,
                        '.jpg')
                WHEN Length(im.`id_image`) = 4 THEN
                  Concat( 'https://yourdomain.com', 'img/p/',
                        INSERT(INSERT(INSERT(im.`id_image`, 2, 0, '/'), 4, 0, '/'), 6, 0, '/'), '/', im.`id_image`, '.jpg')
                WHEN Length(im.`id_image`) = 3 THEN Concat(
                  'https://yourdomain.com', 'img/p/', INSERT(
                    INSERT(im.`id_image`, 2, 0, '/'), 4, 0, '/'), '/', im.`id_image`, '.jpg')
                WHEN Length(im.`id_image`) = 2 THEN Concat(
                  'https://yourdomain.com', 'img/p/',
                  INSERT(im.`id_image`, 2, 0, '/'), '/', im.`id_image`, '.jpg')
                WHEN Length(im.`id_image`) = 1 THEN Concat(
                  'https://yourdomain.com', 'img/p/',
                  INSERT(im.`id_image`, 2, 0, '/'), im.`id_image`, '.jpg')
                ELSE ''
              end AS "img",
              im.cover
              FROM   ps_product p
                    LEFT JOIN ps_image im
                          ON ( im.id_product = p.id_product )
              ORDER BY cover DESC) AS aux
GROUP BY id_product;
```

Hay dos detalles a tener en cuenta sobre esta sentencia,

- **Se debe adaptar el dominio en todos los concat de la misma, con la URL base.**
- Hay que incluir la ordenación por el campo cover en Prestashop (tal y como se muestra en el ejemplo).

## Mejorar el rendimiento en multitienda

Existen varias posibilidades de montar el sistema de sincronización multi-tienda:

### Configurando un conector que sincroniza varias tiendas

Un conector se configura desde Sales Layer y toda la información que se configura en panel de productos, se sincronizará a cada una de las tiendas.

***Ventajas:***

Sincronización rápida.

***Desventajas:***

En caso de desactivar el producto en Sales Layer, se desactivará en todas las tiendas. No es posible hacerlo solo en una tienda en concreto, siempre se desactiva en todas las tiendas que tenemos activadas en el plugin. La información de descuentos, transportes y proveedores, se creará igual en todas las tiendas.

## **Configurando un conector por cada tienda**

***Ventajas:***

Se puede sincronizar toda la información de cada tienda por separado. De esta forma, la información de transportista, descuentos, tags... puede ser actualizada de forma individual para cada tienda por separado.

Si deseamos activar o desactivar un producto en una única tienda, enviamos en su conector correspondiente un campo **Enabled** con valor 0 para desactivar y el valor 1 para activarlo en PrestaShop.

***Desventajas:***

Cada conector recibirá la misma cantidad de información y, por lo tanto, tardará más en sincronizar todo.

## **Resumen de optimización**

Como recomendación para lograr una buena optimización de la sincronización a la hora de montar la tienda con varios conectores y obtener un buen rendimiento :

- En la sincronización, el mayor coste se produce al verificar todas las imágenes: si la imagen ya existe en categoría, producto y variante, si deshabilitamos el campo de imagenes en todos los conectores y dejamos activo solo en uno en una de las tiendas la imágenes, se cargarán solo desde un conector. Así, solo un conector verificará todas las imágenes que cargará para todas las tiendas. Es decir, estamos eliminando el coste de redundancia de carga de imágenes al sincronizar con todos los conectores.
- Los conectores reciben información de los últimos cambios realizados en el cloud de Sales Layer desde la última sincronización. Por lo tanto, si tenemos



configurado que nuestros conectores se sincronicen cada hora, cada conector recibirá únicamente los cambios realizados en la última hora. A excepción de: a) si es la primera sincronización; b) si se cambia la configuración del conector; c) si se refresca mediante el botón de refrescar (caso en que el conector recibe toda la información completa con el consiguiente coste de sincronización).

- Además, siempre que necesitemos que cierta información no se gestione en el conector, podemos desactivar el campo en la configuración del conector en el cloud de Sales Layer.

Hay que tener en cuenta todas estas circunstancias para que la sincronización sea más rápida y eficiente.

## Personalización del módulo

El módulo, tiene una verificación de integridad de los archivos instalados para comprobar que todo ha sido instalado correctamente y avisar de cualquier modificación, mediante mensajes en el log.

Si el propósito es precisamente personalizar el código del módulo, es necesario indicar en el fichero saleslayerimport.php que los cambios los está realizando un desarrollador, modificando la variable:

```
public $i_am_a_developer = false; a true
```

Una vez realizados todos los cambios pertinentes, simplemente es necesario activar el **Debugmode** del módulo y refrescar la página.

El proceso generará un nuevo mapeo de archivos para que el plugin pueda aceptar los cambios realizados y refrescar la integridad de módulo.

Por seguridad, aconsejamos desactivar el modo developer cuándo haya terminado con la edición, ya que el modo developer permite la eliminación masiva de contenido de Prestashop.

## FAQS / Preguntas frecuentes:

**P: Al sincronizar en un e-commerce, ¿sobrescribe datos existentes, los duplica, es necesario eliminarlos?**

**R:** Los datos existentes en una instalación no se eliminan, ni es necesario eliminarlos.

Para sincronizar los datos, se busca una relación de ID entre los elementos existentes con los de Sales Layer (normalmente por una tabla intermedia, o atributos generados en el e-commerce).

Si existe dicha relación, se actualizan los datos.

En caso de no existir, buscaremos entre los elementos existentes uno que tenga la misma referencia o nombre, y que no esté asignado con otro ID de Sales Layer (pertenece a otro conector u otra empresa).

Si lo encontramos, establecemos relación y se actualizan datos.

Si no se encuentra ningún registro, entonces se crea.

**P: Si elimino un producto directamente del PIM, ¿se borra en el e-commerce también?**

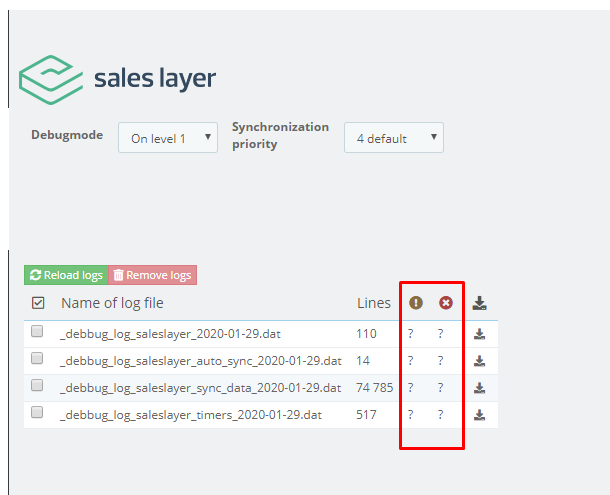
**R:** Las categorías y los productos cambian a estado a desactivado y las variantes se eliminan. Sin embargo, si se refresca, o modifica el conector antes de que pueda sincronizar, los productos no se eliminarán y se quedarán en un estado inconsistente (existen en el e-commerce pero no en Sales Layer). En este caso la única manera de eliminarlos es directamente desde el e-commerce.

**P: ¿Cuánto tiempo tarda en sincronizar un e-commerce?**

**R:** Es un poco relativo, pero aunque depende del e-commerce, recursos, cantidad de elementos existentes, etc. podríamos decir que 0,5-1,5 segundos por categoría y variantes, y 1-3 segundos por producto, al menos para la creación, la sincronización una vez generados los elementos tarda menos tiempo. Y si no se procesan muchas imágenes nuevas es más rápido.

**P: En la consola me sale un error de Ajax 404.**

**R:** Este error puede llegar por muchos motivos. La forma para reconocer que existe un error de Ajax es la que indicamos en la imagen.



Habitualmente, nos hemos encontrado con un error que proviene de cambios en los dominios de PrestaShop. De tal forma que, si en la instalación de prestashop figuraba antes un dominio que ya no existe, es posible que internamente queden restos de reglas de redireccionamiento no actualizados en el fichero htaccess. Para solucionarlo, simplemente hay que verificar en el fichero que el dominio en la configuración en el apartado #

Dispatcher es el actual.

Este error de Ajax no cancela la sincronización. El plugin usa Ajax para que el cliente de forma cómoda y gráfica, pueda gestionar y ver el estado de su sincronización. El módulo tiene implementada la capa de seguridad para que en el caso de que se produzca un error de Ajax, se realice el envío del formulario por post para completar la acción.

```
# Dispatcher
RewriteCond %{REQUEST_FILENAME} -s [OR]
RewriteCond %{REQUEST_FILENAME} -l [OR]
RewriteCond %{REQUEST_FILENAME} -d
```

## ¿Cuándo se actualiza un producto y un campo del mismo no está casado en el conector se seguirá añadiendo para actualizarlo?

Sí, cuando un producto se modifica siempre se envía.

# Versionado

## V 1.4.16

- Ampliación del sistema de asignación de reglas de impuestos (tax). A partir de esta versión, se permite, además del uso de identificadores, añadir el valor propio del impuesto agregando una cadena "número+%" Ej: 21%.
- Agregación de atributos para crear descuentos entre fechas.
- Inclusión de sincronización de atributos alt para imágenes.
- Réplica del funcionamiento del orden de imágenes de productos en variantes. La sincronización respeta el orden que recibe de las imágenes de Sales Layer en PrestaShop.
- Test de integridad de instalación del plugin. El sistema comprueba la integridad de los ficheros instalados para garantizar la integridad del plugin al realizar nuevas instalaciones.
- Añadimos la posibilidad de asignar varios proveedores y asignar uno de ellos por defecto.
- Aceleración del rendimiento en el proceso de indexación.
- Inclusión de mensajes del estado de los niveles de carga del servidor (procesador y memoria) y aumento del número de avisos de diagnóstico para facilitar información al usuario de posibles errores.
- Acceso directo a la documentación desde la pestaña de conectores en el plugin.

## V 1.4.17

- Corrección de la correcta asignación de las características multi-idioma
- Habilitación del campo type (mapear con las palabras 'virtual', 'simple', 'pack')
- Habilitación del campo visibility (mapear con las palabras 'both', 'search', 'catalog', 'none')
- Habilitación del campo redirect\_type (mapear con las palabras '301-product', '302-product', '301-category', '302-category')
- Corrección del campo accesories (para campos de Productos Relacionados, mapear con un campo de Ítem Relacionado)
- Corrección de los indexadores en multi-tiendas
- Corrección de la auto-sincronización para la hora 00:00
- Selección más estricta de atributos de variantes para omitir valores vacíos en un idioma determinado.

- Corrección de la asignación de imágenes como portada en multi-tienda. Ahora se asignan a todas las tiendas, cada vez que se edita el producto (al menos que este sea eliminado)
- Los productos serán creados en todas las tiendas como invisibles y serán solamente activados para las tiendas habilitadas en su configuración.

#### V 1.4.18

- Corrección del campo "atributo alt" de las imágenes.
- Corrección de la imagen de portada para un producto previamente eliminado de una multitienda.
- Se desactivarán las categorías y/o productos en todas las tiendas únicamente si todos los conectores afectados los envían para desactivar.

#### V 1.4.19

- Corrección de los indexadores en multitienda. **Atención:** esta corrección retrasa levemente la sincronización.
- Habilitación del campo "Available Date" tanto para productos como para variantes (mapear con un campo de tipo fecha).
- Corrección del campo "Category Default".

#### V 1.4.20

- Corrección de la URL del producto
- Renombre de las variantes en cada sincronización
- Corrección de la barra de progreso cuando esté trabajando con accesorios.
- Mejoras en el mapeo de los accesorios.
- Desarrollado un indexador más eficiente.
- Mejoras en los tiempos de sincronización

#### V 1.4.21

- Añadida función que repara las tablas del plugin en "modo diagnóstico".
- Solucionado el problema de variantes sin nombre.
- Añadido el campo "enabled" para variantes, que elimina una variante si es falso.

#### V 1.4.22

- Añadido el campo de condición.

#### V 1.4.23

- Solucionado el campo de condición para valores personalizados, leyendo de la clase de Product.
- Solucionado el campo Visibility para campos personalizados, leyendo de la clase de Producto.

- Solucionado el renombrado del valor de las features.
- Borrado estricto de las imágenes de producto si son desconocidas para el plugin.

#### **V 1.4.24**

- Solucionada la compresión en los valores de feature.
- Solucionado el campo "minimum quantity".
- Solucionado el campo "quality", ahora se puede poner en 0.

- 

#### **V 1.4.25**

- Mejora en la compatibilidad de PHP 7.3, mejorando y reduciendo las notificaciones de “undeclared variables”.
- Se puede cambiar el tipo de producto de la configuración en pack.
- Mensaje mejorado en los mensajes de “unlinked products” en pack.
- Solución aplicada al borrado de accesorios.

#### **V 1.4.26**

- Creación de producto únicamente para tiendas sólo configuradas en el conector.
- Creación de funcionalidades en todas las tiendas (Solución de errores en multitienda)
- Better control of product deallocation - categories.
- Mejor control de la designación de productos y categorías.
- Solucionado variantes con los mismos atributos no se crean.

#### **v 1.5.0**

- Mejoras de velocidad.
- Pequeñas correcciones.
- Actualización de stock asíncrona.
- Precarga de imagen asíncrona.
- Comparación de datos con hash de datos.
- Balanceador.
- Trabajo en multiprocesos.
- Las variantes con productos se sincronizan con productos.

#### **v 1.5.1**

- Corrección masiva de actualización de stock.
- Problema de categoría que se ha producido por no mantener el orden de procesamiento en las categorías.
- Pequeñas correcciones.
- Mejor compatibilidad con 1.7.8.x Prestashop multitienda.
- Selección de categoría principal para la primera categoría: ID de categoría de tienda si no existe -> PS\_HOME\_CATEGORY si no existe -> PS\_ROOT\_CATEGORY