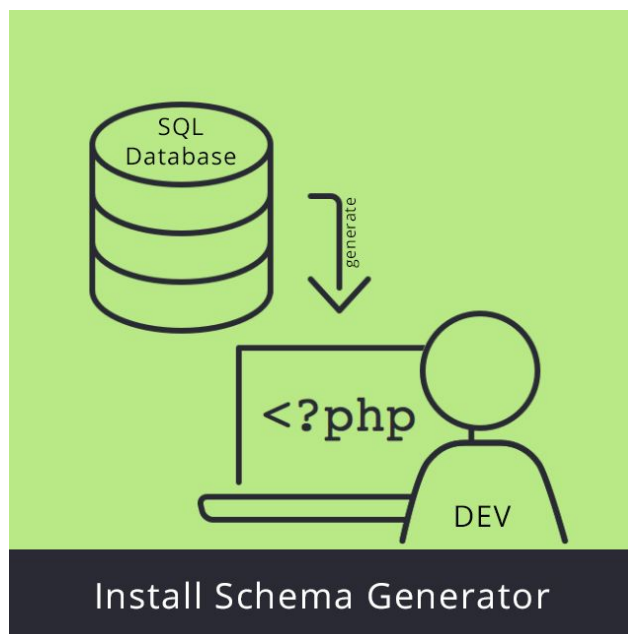


blackbird

Install Schema Generator User Guide

- [1. Information](#)
- [2. Build your database tables](#)
- [3. Generate the InstallSchema.php file](#)
 - [How to find Install Schema Generator](#)
 - [How to use Install Schema Generator](#)
- [4. Put the file in your project](#)



Install Schema Generator (ISG) is a development tool destined to speed up the table setup script creation of a Magento 2 module. It helps you to generate and download your custom install schema from your Magento database.

1. Information

Install Schema Generator has been [developed for developers](#). It used to develop faster your setup script of your Magento 2.x Extension. You can actually create your tables on Phpmyadmin in your Magento database, then easily convert them into an install schema! It's more convenient and faster than to code yourself the entire setup script. Moreover, you avoid spelling and programming errors and you can be more focused on the database design.

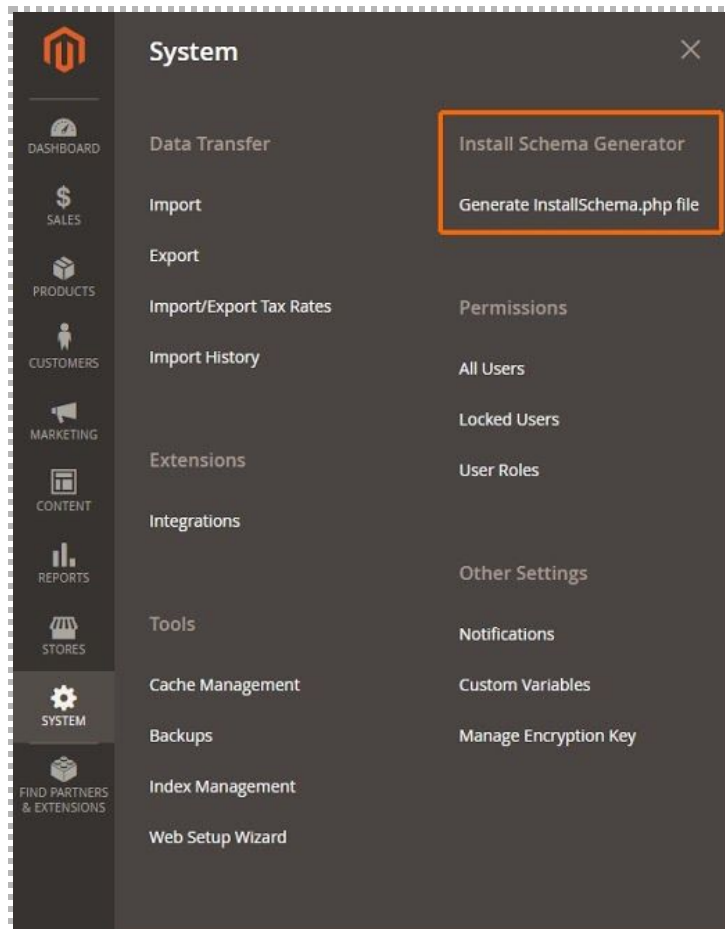
2. Build your database tables

Create your table(s) in SQL or easilly with phpMyAdmin, for your Magento 2.x database. Make sure that your table(s) are now available from your database.

3. Generate the InstallSchema.php file

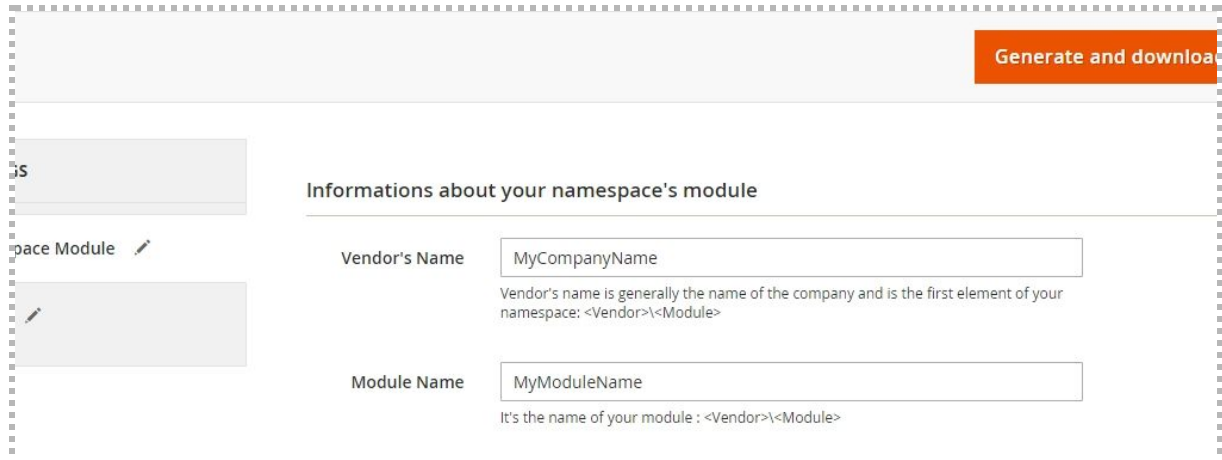
A. How to find Install Schema Generator

Connect to your Magento 2.x admin panel, and find "Install Schema Generator".



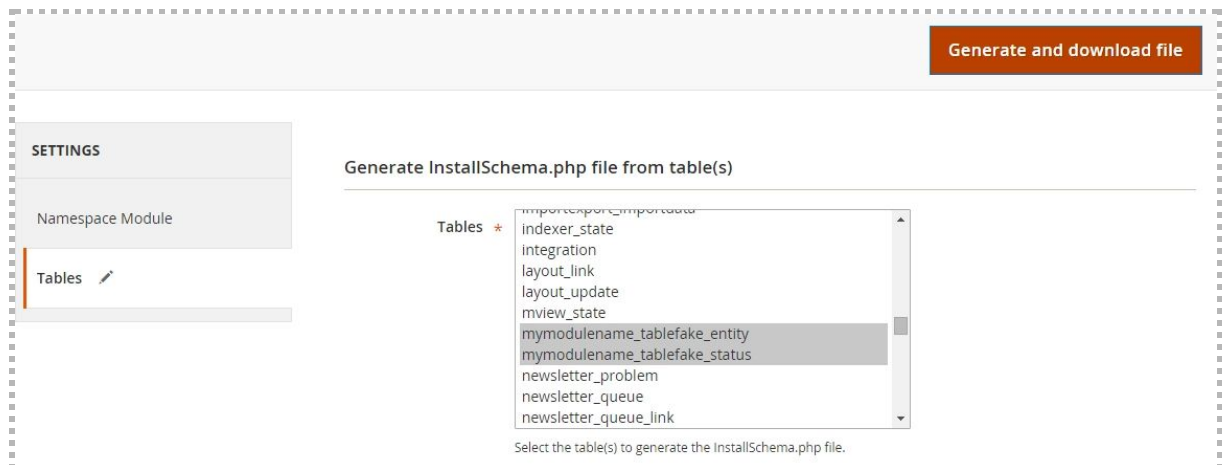
B. How to use Install Schema Generator

In the "Namespace Module" tab, enter your vendor name and your module name.



The screenshot shows the 'Namespace Module' tab of the Install Schema Generator. On the left, there is a sidebar with a 'Namespace Module' tab selected. The main content area is titled 'Informations about your namespace's module'. It contains two input fields: 'Vendor's Name' with the value 'MyCompanyName' and 'Module Name' with the value 'MyModuleName'. Below each field is a small explanatory text: 'Vendor's name is generally the name of the company and is the first element of your namespace: <Vendor>\<Module>' and 'It's the name of your module : <Vendor>\<Module>'. A red 'Generate and download' button is located in the top right corner.

In the "Tables" tab, select the tables to transform to an InstallSchema.php file.



The screenshot shows the 'Tables' tab of the Install Schema Generator. On the left, there is a sidebar with a 'Tables' tab selected. The main content area is titled 'Generate InstallSchema.php file from table(s)'. It features a list of tables in a scrollable container. The tables listed are: 'indexer_state', 'integration', 'layout_link', 'layout_update', 'mview_state', 'mymodulename_tablefake_entity', 'mymodulename_tablefake_status', 'newsletter_problem', 'newsletter_queue', and 'newsletter_queue_link'. The 'mymodulename_tablefake_entity' and 'mymodulename_tablefake_status' tables are highlighted. Below the list, there is a small instruction: 'Select the table(s) to generate the InstallSchema.php file.' A red 'Generate and download file' button is located in the top right corner.

Click on the "Generate and download file" button. Your browser will suggest you to download the InstallSchema PHP file. ISG reads the structure of the selected tables and extracts the column setup including constraints, indexes or foreign keys. These data build the table creation scripts which match Magento coding standards!

However, if you recover the setup script of Magento 1.x database tables, you should check if the constraints of foreign keys are correct. Some tables may have been renamed, changed or deleted in Magento 2.

4. Put the file in your project

Guess you have now a downloaded file, you can open it and check if the data are correct. Place the file under the "Setup" folder of your Magento 2.x module. Enjoy it!

```

1 <?php
2 namespace MyAgency\MyModuleName\Setup;
3
4 use Magento\Framework\Setup\InstallSchemaInterface;
5 use Magento\Framework\Setup\ModuleContextInterface;
6 use Magento\Framework\Setup\SchemaSetupInterface;
7
8 class InstallSchema implements InstallSchemaInterface
9 {
10     public function install\SchemaSetupInterface $setup, ModuleContextInterface
11     {
12
13         $installer = $setup;
14
15         $installer->startSetup();
16
17
18         /**
19          * Create table 'mymodulename_tablefake_entity'
20          */
21         $table = $installer->getConnection()
22             ->newTable($installer->getTable('mymodulename_tablefake_entity'))
23             ->addColumn(
24                 'id',
25                 \Magento\Framework\DB\Ddl\Table::TYPE_INTEGER,
26                 11,
27                 [
28                     'nullable' => false,
29                     'precision' => '10',
30                     'auto_increment' => true,
31                 ],
32                 null
33             )
34             ->addColumn(
35                 'status',
36                 \Magento\Framework\DB\Ddl\Table::TYPE_SMALLINT,
37                 1,
38                 [
39                     'nullable' => false,
40                     'precision' => '3',
41                 ],
42                 null
43             )
44             ->addColumn(
45                 'comment',
46                 \Magento\Framework\DB\Ddl\Table::TYPE_TEXT,
47                 255,
48                 [
49                     'nullable' => false,
50                 ],
51                 null
52             )
53             ->addColumn(
54                 'mark',
55                 \Magento\Framework\DB\Ddl\Table::TYPE_INTEGER,
56                 10,
57                 [

```

For further information, check our documentation:

<https://bitbucket.org/blackbirdagency/magento2-extensions-isgenerator>