```
INSTALL COMPONENT 'file://component1', 'file://component2';
```

A component is named using a URN that begins with `file://` and indicates the base name of the library file that implements the component, located in the directory named by the `plugin_dir` system variable. Component names do not include any platform-dependent file name suffix such as `.so` or `.dll`. (These naming details are subject to change because component name interpretation is itself performed by a service and the component infrastructure makes it possible to replace the default service implementation with alternative implementations.)

If any error occurs, the statement fails and has no effect. For example, this happens if a component name is erroneous, a named component does not exist or is already installed, or component initialization fails.

A loader service handles component loading, which includes adding installed components to the `mysql.component` system table that serves as a registry. For subsequent server restarts, any components listed in `mysql.component` are loaded by the loader service during the startup sequence. This occurs even if the server is started with the `--skip-grant-tables` option.

If a component depends on services not present in the registry and you attempt to install the component without also installing the component or components that provide the services on which it depends, an error occurs:

```
ERROR 3527 (HY000): Cannot satisfy dependency for service 'component_a'
required by component 'component_b'.
```

To avoid this problem, either install all components in the same statement, or install the dependent component after installing any components on which it depends.

Note

For keyring components, do not use `INSTALL COMPONENT`. Instead, configure keyring component loading using a manifest file. See Section 6.4.4.2, “Keyring Component Installation”.

13.7.4.4 INSTALL PLUGIN Statement

```
INSTALL PLUGIN plugin_name SONAME 'shared_library_name'
```

This statement installs a server plugin. It requires the `INSERT` privilege for the `mysql.plugin` system table because it adds a row to that table to register the plugin.

`plugin_name` is the name of the plugin as defined in the plugin descriptor structure contained in the library file (see Plugin Data Structures). Plugin names are not case-sensitive. For maximal compatibility, plugin names should be limited to ASCII letters, digits, and underscore because they are used in C source files, shell command lines, M4 and Bourne shell scripts, and SQL environments.

`shared_library_name` is the name of the shared library that contains the plugin code. The name includes the file name extension (for example, `libmyplugin.so`, `libmyplugin.dll`, or `libmyplugin.dylib`).

The shared library must be located in the plugin directory (the directory named by the `plugin_dir` system variable). The library must be in the plugin directory itself, not in a subdirectory. By default, `plugin_dir` is the `plugin` directory under the directory named by the `pkglibdir` configuration variable, but it can be changed by setting the value of `plugin_dir` at server startup. For example, set its value in a `my.cnf` file:

```
[mysqld]
plugin_dir=/path/to/plugin/directory
```