

Tips and guidelines

Various useful tips and guidelines when working with database from your component.

Table names for your component

If you intend to distribute your component to other people you should be careful how to name tables that your component uses. Tables that are created when installing Joomla have no prefix (except table name prefix, e.g. 'jos_'). Let's say that your component needs table 'inbox'. Although Joomla doesn't have table named #__inbox it is not a good practice to name a table for your component that way. That is because another component might be using the same name and when another user tries to install your component it won't work.

If you look at the tables after you install some other component you will see that its tables are named with one or more prefixes beside the default table name prefix.

The table name prefix for your component should be logical to your component name (abbreviation), plus you can add your company name or your developer name.

For example, if you are developing e-mail client component for Joomla you can give a prefix 'joommail' so that you 'inbox' table will be named '#__joommail_inbox'. If you want to be even more certain that your table names will not be in conflict with other components you can add your company name, so your 'inbox' table will be named '#__joomworks_joommail_inbox'.

Note that the prefix names I used here are not quite long and in real world you will probably use shorter prefixes.

SQL injection protection

SQL injection is a technique that exploits a security vulnerability occurring in the database layer of an application. If your component accepts user input and process trough query, you should secure your component against SQL injection.

```
$value = $_GET['username'];
```

```
$database->setQuery( "SELECT * FROM #__users WHERE username = '$username' " );
```

The attacker could hand over the string john' or 'e'='e resulting in query

```
SELECT * FROM #__users WHERE username='john' or 'e'='e'
```

This could in some cases display all data from table users.

The first line of protection would be using a mosGetParam function instead \$_GET, because mosGetParam will return escaped values.

This will not protect you against some SQL injections that use numbers. If you expect a parameter that must be number you must try to convert it to integer using intval function.