

The following are samples of database work I have done. I have attached a sql file as a sample. It is a dev snapshot of a CMS I wrote.

So you don't have to just stare at sql statements, I have highlighted some points in this post. I will paste sections of the sql statements and then comment on them.

```
1: --
2: -- Table structure for table `assets`
3:
4: --
5: CREATE TABLE `assets` (
6:
7: `id` int(11) NOT NULL auto_increment,
8:
9: `asset_title` varchar(255) default NULL,
10:
11: `original_file_name` varchar(255) default NULL,
12:
13: `new_file_name` varchar(255) NOT NULL default '',
14:
15: `user_id_who_added` int(11) default NULL,
16:
17: `time_stamp_added` int(11) default NULL,
18:
19: `file_size` int(11) default NULL,
20:
21: `key_words` varchar(255) default NULL,
22:
23: `file_name_ext` varchar(4) default NULL, PRIMARY KEY (`id`),
24:
25: KEY `asset_title` (`asset_title`),
26:
27: KEY `original_file_name` (`original_file_name`),
28:
29: KEY `new_file_name` (`new_file_name`),
30:
31: KEY `key_words` (`key_words`)
32:
33: ) TYPE=MyISAM AUTO_INCREMENT=10 ;
34:
35: --
36: -- Dumping data for table `assets`
37:
38: --
39: INSERT INTO `assets` VALUES (1, 'test', 'test.jpg', '457212', 20, 20, 20,
40:
41: 'you', 'jpg');
```

Prior to my building this CMS for the client, they had a number of users, managing several sections of their site, each uploading their own large number of images, flash and other assets.

This resulted in a few issues:

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

1. There was no sharing of assets across users
2. Often redundant images and assets were created by each user
3. Assets were often uploaded and then abandoned over time as they were no longer served up on web pages. The result was a large number of extra assets but with no easy way to clean them up.
4. End-users often uploaded large files sizes with no regard for optimization, therefore serving large files through the web pages to site visitors.

So, I implemented the following:

1. Managed assets via a table in the database
2. I created a script that upon upload through the CMS, meta data about the assets was recorded, such as title, id of the user who added it, file size, and user submitted keywords or tags for the asset
3. All site assets were then served via a file called asset.php with an id in the query string: For example, a call to asset.php?id=22 would serve an optimized and cached version of the original image, or flash file.
4. I created an asset search script which tied in with the FCK Editor WYSIWYG to allow end users to search for previously uploaded assets by title or key word so that assets were now shared between content writers.
5. Assets could be deleted by a master admin, and before deletion a check was made to the CMS nodes to make sure the asset was not used in an existing page.

A few things to note from a database engineering perspective:

For the primary key id I used the MySQL built-in auto increment. I probably would not do this now, if I planned to migrate to other dbases. Drupal also recently switched to a more database agnostic method for generating auto increments.

The field user_id_who_added serves as a foreign key for us to join the user name of the person who added the file.

Note that for file_name_ext I only use a varchar(4) rather than the default varchar(255). It's these little practices of using just what's needed in your storage engine, that add up to better

performing systems.

I placed keys on

```
1: KEY `asset_title`, `original_file_name`, `new_file_name`, and `key_words`
```

because we allowed users to search assets by these fields.

These next tables manage my CMS groups and permissions.

They are modeled after the example system outlined in the book titled, "The PHP Anthology" by Harry Fuecks.

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

A few things to note: The groups table is titled collection rather than groups in order to avoid internal naming errors on some systems.

These tables represent an elegant and optimized way to manage user groups and their permissions.

The classes used to manage these tables may be found in the sample code. They are in the folder: cms portal code\includes\PORTALLIB\AccessControl

By the way, these classes use constants to define the table and column names for the user system. This has been helpful as I have applied these classes to several legacy user systems.

Here is an example method within the user class which checks user permissions.

You can see an example of my using a sql statement, using the constants for table names and such, and fetching the data via my database abstraction layer.

```
210      /**
211       * Checks to see if the user has the named permission
212       *
213       * Checks to see if the user has the named permission somewhere from
214       * the current content level or from one of the parents
215       * @param string $ name of a permission
216       * @param array $ an array of content id's (parents of the current location, plus the loc
217       * @return boolean TRUE is user has permission
218       * @access public
219       */
220      function checkPermission( $permission, $stopLevel )
221      {
222          $levelClause = " AND top_location_id = $stopLevel";
223          // If I don't have any permissions, fetch them
224          if ( !isset( $this->permissions ) )
225          {
226              $this->permissions = array();
227              $sql = "SELECT
228 p." . PERM_TABLE_NAME . " as permission
229 FROM
230 " . USER2COLL_TABLE . " uc, " . COLL2PERM_TABLE . " cp, " . PERM_TABLE . " p
231 WHERE
232 uc." . USER2COLL_TABLE_USER_ID . " = " . $this->userId . "
233 AND
234 uc." . USER2COLL_TABLE_COLL_ID . " = cp." . COLL2PERM_TABLE_COLL_ID . "
235 AND
236 cp." . COLL2PERM_TABLE_PERM_ID . " = p." . PERM_TABLE_ID . $levelClause;
237
238              $result = $this->db->query( $sql );
239              while ( $row = $result->fetch() )
240              {
241                  $this->permissions[] = $row['permission'];
242              }
243          }
244          if ( in_array( $permission, $this->permissions ) )
245              return true;
246          else
247              return false;
248      }
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
1: --
2: -- Table structure for table `collection`
3:
4: --
5: CREATE TABLE `collection` (
6:
7: `collection_id` int(11) NOT NULL auto_increment,
8:
9: `name` varchar(50) NOT NULL default '',
10:
11: `description` text NOT NULL,
12:
13: `required_permissions` varchar(255) NOT NULL default '',
14:
15: PRIMARY KEY (`collection_id`)
16:
17: ) TYPE=MyISAM AUTO_INCREMENT=5 ;
18:
19: --
20: -- Dumping data for table `collection`
21:
22: --
23: INSERT INTO `collection` VALUES (1, 'Master Admin', '', 'system');
24:
25: INSERT INTO `collection` VALUES (2, 'Admin', '', 'create user- micro admin');
26:
27: INSERT INTO `collection` VALUES (3, 'Webmaster', 'is a webmaster', 'create user- micro webmaster');
28:
29: INSERT INTO `collection` VALUES (4, 'Content Contributor', 'is a CONTENT
30:
31: contributor', 'create user- content contributor');
32:
33: -----
34:
35: --
36: -- Table structure for table `collection2permission`
37:
38: --
39: CREATE TABLE `collection2permission` (
40:
41: `collection_id` int(11) NOT NULL default '0',
42:
43: `permission_id` int(11) NOT NULL default '0', PRIMARY KEY (`collection_id`,`permission_id`)
44:
45: ) TYPE=MyISAM;
46:
47: --
48: -- Dumping data for table `collection2permission`
49:
50: --
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
51: INSERT INTO `collection2permission` VALUES (1, 1); INSERT INTO `collection2permission` VALUES (1,
52: 2); INSERT INTO `collection2permission` VALUES (1, 4); INSERT INTO `collection2permission` VALUES (1, 5);
53: ....
54:
55: INSERT INTO `collection2permission` VALUES (1, 16);
56:
57: .....
58:
59: INSERT INTO `collection2permission` VALUES (2, 4); INSERT INTO `collection2permission` VALUES (2,
60: 5); INSERT INTO `collection2permission` VALUES (2, 12); INSERT INTO `collection2permission` VALUES (2,
61: 13);
62: --
63: -- Table structure for table `permission`
64: --
65: CREATE TABLE `permission` (
66:
67: `permission_id` int(11) NOT NULL auto_increment,
68:
69: `name` varchar(50) NOT NULL default '',
70:
71: `description` text NOT NULL,
72:
73: PRIMARY KEY (`permission_id`)
74:
75: ) TYPE=MyISAM AUTO_INCREMENT=38 ;
76:
77: --
78: -- Dumping data for table `permission`
79:
80: --
81: INSERT INTO `permission` VALUES (1, 'Content:Delete', '');
82:
83: INSERT INTO `permission` VALUES (2, 'Content:SaveAndPublish', ''); INSERT INTO `permission` VALUES
84: (3, 'Content:SaveAndEmail', ''); INSERT INTO `permission` VALUES (4, 'Content:SaveAsDraft', ''); INSERT
85: INTO `permission` VALUES (5, 'Content:Begin New', '');
86:
87: INSERT INTO `permission` VALUES (6, 'Microsite:Make Microsite', 'Has
88: permission to make a content type a microsite');
89:
90:
91: INSERT INTO `permission` VALUES (7, 'Microsite>Delete Microsite', 'Has permission to delete a
92: microsite content type');
93: ....
94:
95: INSERT INTO `permission` VALUES (14, 'ContentTypePermissions:Parish Site',
96:
97: 'Can select this content type');
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
98:
99: INSERT INTO `permission` VALUES (15, 'ContentTypePermissions:Announcement
100:
101: Item', 'Can select this content type');
102:
103: INSERT INTO `permission` VALUES (16, 'ContentTypePermissions:Link', 'Can
104:
105: select this content type');
106:
107: --
108: -- Table structure for table `user`
109:
110: --
111: CREATE TABLE `user` (
112:
113: `user_id` int(11) NOT NULL auto_increment,
114:
115: `login` varchar(50) NOT NULL default '',
116:
117: `password` varchar(50) NOT NULL default '',
118:
119: `email` varchar(50) default NULL,
120:
121: `firstName` varchar(50) default NULL,
122:
123: `lastName` varchar(50) default NULL,
124:
125: `signature` text NOT NULL,
126:
127: `last_login` int(11) NOT NULL default '0',
128:
129: PRIMARY KEY (`user_id`),
130:
131: UNIQUE KEY `user_login` (`login`)
132:
133: ) TYPE=MyISAM AUTO_INCREMENT=5 ;
134:
135: --
136: -- Dumping data for table `user`
137:
138: --
139: INSERT INTO `user` VALUES (2, 'Admin', '43e91f41b28875fada7e6c1979ba7ff8',
140:
141: 'helpdesk@websupport.com', Admin', 'Support Desk', '', 1147383307);
142:
143: -- -----
144:
145: --
146:
147: -- Table structure for table `user2collection`
148:
149: --
150:
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
151: CREATE TABLE `user2collection` (  
152:  
153: `user_id` int(11) NOT NULL default '0',  
154:  
155: `collection_id` int(11) NOT NULL default '0',  
156:  
157: `top_location_id` int(11) NOT NULL default '0',  
158:  
159: PRIMARY KEY (`user_id`,`collection_id`,`top_location_id`)  
160:  
161: ) TYPE=MyISAM;  
162:  
163: --  
164: -- Dumping data for table `user2collection`  
165:  
166: --  
167: INSERT INTO `user2collection` VALUES (2, 1, 1); INSERT INTO `user2collection` VALUES (2, 2, 8813);  
INSERT INTO `user2collection` VALUES (2, 2, 9096);  
168:  
169:  
170:  
171: INSERT INTO `user2collection` VALUES (2, 3, 8832);
```

Next is a basic table for comments. Note the use of tinyint for boolean like data.

```
1: --  
2: -- Table structure for table `comments`  
3:  
4: --  
5: CREATE TABLE `comments` (  
6:  
7: `com_id` int(11) NOT NULL auto_increment,  
8:  
9: `time_stamp` int(11) NOT NULL default '0',  
10:  
11: `name_of_commenter` varchar(255) NOT NULL default '',  
12:  
13: `text` text NOT NULL,  
14:  
15: `content_id` int(11) NOT NULL default '0',  
16:  
17: `approved` tinyint(4) NOT NULL default '0',  
18:  
19: `ipaddress` varchar(255) NOT NULL default '',  
20:  
21: PRIMARY KEY (`com_id`),  
22:  
23: KEY `com_content_id` (`content_id`),  
24:  
25: KEY `approved` (`approved`)  
26:  
27: ) TYPE=MyISAM AUTO_INCREMENT=1 ;
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

Next we have the content type tables. A few items of note:

1. The field titled, „name“ has been indexed with a unique key to enforce unique content type names.
2. Permissions_required was used in conjunction with the users permissions object to check whether they had the required permission to view the content type in a list or to create the content type.
3. Sort_order was just something I stuck directly into the table to allow me to serve the content list in the order I desired, without rearranging them in the array after the list was called.

```
1: --
2: -- Table structure for table `content_types`
3:
4: --
5: CREATE TABLE `content_types` (
6:
7: `id` int(11) NOT NULL default '0',
8:
9: `name` varchar(255) NOT NULL default '',
10:
11: `description` text NOT NULL,
12:
13: `permissions_required` varchar(255) NOT NULL default '',
14:
15: `icon` varchar(255) NOT NULL default '',
16:
17: `icon_expanded` varchar(255) NOT NULL default '',
18:
19: `sort_order` int(11) NOT NULL default '0',
20:
21: PRIMARY KEY (`id`),
22:
23: UNIQUE KEY `name` (`name`), KEY `order` (`sort_order`)
24:
25: ) TYPE=MyISAM;
26:
27: --
28: -- Dumping data for table `content_types`
29:
30: --
31: INSERT INTO `content_types` VALUES (102, 'Main Site Home', 'Main site description',
'ContentTypePermissions:Main Site Home', 'home.gif',
32:
33: 'home.gif', 1);
34:
35: INSERT INTO `content_types` VALUES (1, 'Meta Section', 'Meta sections are
36:
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```

37: sections which are linked to by every page within the web site.\\n\\nThese include links to the
following sections:\\n\\nHome, About, Contact Us, Pressroom, Departments and Organizations, Parishes,
Articles, Privacy Policy and Sitemap\\n\\nMeta sections are found at the top level of our web site.',
38:
39: 'ContentTypePermissions:Meta Section', 'metasection.gif',
40:
41: 'metasectionopen.gif', 2);
42:
43: ....
44:
45: INSERT INTO `content_types` VALUES (2, 'Micro site', 'Acts as a mini site
46:
47: with its own Administrator assigned to it\\n\\nSets template and logo look for all sections
down\\n\\nActs as the portal into other sections of the micro site\\n\\nLinks to this content type shows
up on the main side navigation bars (left)\\r\\n', 'ContentTypePermissions:Micro site',
48:
49: 'microsite.gif', 'microsite.gif', 3);
50:
51: INSERT INTO `content_types` VALUES (6, 'Calendar', '',
52:
53: 'ContentTypePermissions:Article', 'news_event.gif', 'news_event.gif', 10);
54:
55: INSERT INTO `content_types` VALUES (7, 'Photo Album', '',
56:
57: 'ContentTypePermissions:Article', 'camera.gif', 'camera.gif', 10);
58:
59: INSERT INTO `content_types` VALUES (8, 'Book', '',
60:
61: 'ContentTypePermissions:Article', 'panels.gif', 'panels.gif', 11);
62:
63: INSERT INTO `content_types` VALUES (9, 'Subject', '', 'Subjects:Create New',
64:
65: 'attach.gif', 'attach.gif', 12);

```

Let's look at the node to subject system. This provided a way for us to begin to assign nodes to various subjects. Multiple nodes could be assigned to multiple subjects, but we did not want duplicates of the same node assigned to the same subject. Thus the use of:

```
1: KEY `nodeid` (`nodeid`,`subjectid`).
```

This was the beginning of a tagging system which ultimately was improved upon in the later migration to Drupal.

```

1: --
2: -- Table structure for table `node2subject`
3:
4: --
5: CREATE TABLE `node2subject` (
6:
7: `nodeid` int(11) NOT NULL default '0',
8:
9: `subjectid` int(11) NOT NULL default '0',

```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
10:
11: KEY `nodeid` (`nodeid`,`subjectid`)
12:
13: ) TYPE=MyISAM;
14:
15: --
16: -- Dumping data for table `node2subject`
17:
18: --
19: INSERT INTO `node2subject` VALUES (9023, 9023); INSERT INTO `node2subject` VALUES (9119, 9120);
INSERT INTO `node2subject` VALUES (9122, 9120); INSERT INTO `node2subject` VALUES (9123, 9121);
20:
21: --
22: -- Table structure for table `signup`
23:
24: --
25: CREATE TABLE `signup` (
26:
27: `signup_id` int(11) NOT NULL auto_increment,
28:
29: `login` varchar(50) NOT NULL default '',
30:
31: `password` varchar(50) NOT NULL default '',
32:
33: `email` varchar(50) default NULL,
34:
35: `firstName` varchar(50) default NULL,
36:
37: `lastName` varchar(50) default NULL,
38:
39: `signature` text NOT NULL,
40:
41: `confirm_code` varchar(40) NOT NULL default '',
42:
43: `created` int(11) NOT NULL default '0', PRIMARY KEY (`signup_id`),
44:
45: UNIQUE KEY `confirm_code` (`confirm_code`),
46:
47: UNIQUE KEY `user_login` (`login`), UNIQUE KEY `email` (`email`)
48:
49: ) TYPE=MyISAM AUTO_INCREMENT=1 ;
50:
```

This is a signup table. Note several unique keys placed on the fields such as email address and user login. This is an example of where I use:

- business logic in PHP to check during the sign up process for duplicates and then provide useful information back to the user, such as “The user name you have chosen already exists, please choose another”.
- still have my primary backup business logic also built into the database, incase something gets by.

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

By the way, this kind of system can now be improved with a bit of AJAX.

Okay, now the big kahuna tables.

For managing the nodes tables I used a Pear Package called, "DB_NestedSet".

http://pear.php.net/package/DB_NestedSet

This is the package description:

With this package, one can easily create trees with infinite depth inside a relational database. The package provides a way to

- create/update/delete nodes
- query nodes, trees and subtrees
- copy (clone) nodes, trees and subtrees
- move nodes, trees and subtrees
- etc.

It uses and improves upon the Modified Preorder Tree Traversal algorithm.

@see <http://www.sitepoint.com/article/hierarchical-data-database/2>

To walk the node tree and place left right ids for each node. This is an expensive process and is usually done during writes to the database.

Once done, it becomes a whole lot less expensive to call up a node tree from the database. Calling a node and its children might look something like this:

```
1: SELECT STRNA FROM tb_nodes WHERE l < 4 AND r > 5 ORDER BY l ASC;
```

I used the DB_NestedSet classes to manage the nodes, which used Pears database abstraction layer to query the database. I then abstracted most of this process by writing my own set of classes which interfaced with the library. These may be found in the folder : CMS portal code\includes\PORTALLIB\Content

This allowed me to switch class libraries for managing my tree without breaking interface within my CMS. I had already switched once from a tree building method similar to Drupal's but it would get bogged down with using large PHP arrays.

As these classes contain a lot of code, please take a look at the file ContentMenu.php to see how I managed the CMS menu.

Please see: ContentPage.php to see how I pulled out data for just a page view.

Finally, please see: Content.php to see the class I created to abstract most of the methods from the Pear Class. As you likely know, this is using an aspect of object oriented programming called, polymorphism.

The following screen shot shows an example of using methods to add or takeaway elements of the sql statement as needed.

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
94
95  /**
96   * Method to add archived material back into the content to be called
97   *
98   * @access protected
99   */
100  function addArchivestoSQL()
101  {
102      $this->addSQL = array (
103          'where' =>
104              'pub_status = 1
105  AND content_type !=3
106  AND date_to_show < ' . time()
107          . ' AND (date_stop_show IS NULL
108  OR date_stop_show =0
109  OR date_stop_show > ' . time() . ' )'
110          );
111  }
112
113
114  /**
115   * Method to remove the archived content back out of the sql call
116   *
117   * @access protected
118   */
119  function takeArchivesfromSQL()
120  {
121      $this->addSQL = array (
122          'where' =>
123              'archive !=1
124  AND pub_status = 1
125  AND content_type !=3
126  AND date_to_show < ' . time()
127          . ' AND (date_stop_show IS NULL
128  OR date_stop_show =0
129  OR date_stop_show > ' . time() . ' )'
130          );
131  }
```

Now, if I were reviewing this code, I might ask the question, “Why such a big node table? Why not optimize by splitting the large table into smaller ones? For example, Drupal uses node and node_revisions tables.

The brief answer is that it was easier and in many cases more efficient. It turns out that the only large data in the tables is page_content and keywords. Also, in the beginning, most of the nodes, if not all, had the same data, so there were few empty columns wasting space.

I had initially split the table, but ended up having to write a lot of joins to bring the data back together. In almost all cases, except for the menu building, all node data was used from each single query.

So, I could get almost all needed at runtime with one query which cost little because of the left right ids (selecting between them), rather than needing to do multiple secondary queries to join the data.

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

After about 4500 nodes in the system the table writes were beginning to take a lot of overhead. My next step would have been to look again at breaking up the table, but we switched instead to Drupal. By the way, it was Drupal 4 at the time, and it could not handle our 4.5K nodes. I had to archive a lot of stuff. Now, Drupal 5 does fine.

Another quick note:

```
1: FULLTEXT KEY `art_search` (`STRNA`,`page_content`,`keywords`)
```

Allowed me to use the MySQL built-in full text search which improved our search results significantly.

```
1: --
2: -- Table structure for table `tb_locks`
3:
4: --
5: CREATE TABLE `tb_locks` (
6:
7: `lockID` char(32) NOT NULL default '',
8:
9: `lockTable` char(32) NOT NULL default '',
10:
11: `lockStamp` int(11) NOT NULL default '0',
12:
13: PRIMARY KEY (`lockID`,`lockTable`)
14:
15: ) TYPE=MyISAM COMMENT='Table locks for NestedSet';
16:
17: --
18: -- Dumping data for table `tb_locks`
19:
20: --
21: -- -----
22:
23: --
24: -- Table structure for table `tb_nodes`
25:
26: --
27: CREATE TABLE `tb_nodes` (
28:
29: `STRID` int(11) NOT NULL auto_increment,
30:
31: `ROOTID` int(11) NOT NULL default '1',
32:
33: `l` int(11) NOT NULL default '0',
34:
35: `r` int(11) NOT NULL default '0',
36:
37: `content_type` tinyint(1) NOT NULL default '0',
38:
39: `STRNA` varchar(255) NOT NULL default '',
40:
41: `link_text` varchar(255) default NULL,
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
42:
43: `location` varchar(255) NOT NULL default '',
44:
45: `template_header` text NOT NULL,
46:
47: `template_footer` text NOT NULL,
48:
49: `keywords` text,
50:
51: `page_content` longtext,
52:
53: `LEVEL` int(11) NOT NULL default '0',
54:
55: `PARENT` int(11) default NULL,
56:
57: `link_url` varchar(255) default NULL,
58:
59: `intro_text` text,
60:
61: `time_stamp_added` int(11) default NULL,
62:
63: `date_to_show` int(11) default NULL,
64:
65: `date_stop_show` int(11) default NULL,
66:
67: `event_dates` text,
68:
69: `allow_comments` tinyint(1) NOT NULL default '0',
70:
71: `pub_status` tinyint(1) default '0',
72:
73: `added_by` int(11) default NULL,
74:
75: `time_stamp_last_edit` int(11) NOT NULL default '0',
76:
77: `last_edit_by` int(11) default NULL,
78:
79: `STREH` int(11) NOT NULL default '0',
80:
81: `archive` tinyint(1) NOT NULL default '0',
82:
83: `site_url` varchar(255) default NULL,
84:
85: `site_name` varchar(255) default NULL,
86:
87: `site_theme` varchar(255) default NULL,
88:
89: PRIMARY KEY (`STRID`), KEY `ROOTID` (`ROOTID`), KEY `archive` (`archive`), KEY `STREH` (`STREH`),
90:
91: KEY `l` (`l`),
92:
93: KEY `r` (`r`),
94:
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
95: KEY `LEVEL` (`LEVEL`),
96:
97: KEY `parent` (`PARENT`),
98:
99: KEY `time_stamp_last_edit` (`time_stamp_last_edit`), KEY `date_stop_show` (`date_stop_show`),
100:
101: KEY `pub_status` (`pub_status`),
102:
103: KEY `content_type` (`content_type`),
104:
105: KEY `location` (`location`),
106:
107: KEY `date_to_show` (`date_to_show`), KEY `SRLR` (`ROOTID`,`l`,`r`),
108:
109: KEY `site_url` (`site_url`), KEY `site_name` (`site_name`),
110:
111: FULLTEXT KEY `art_search` (`STRNA`,`page_content`,`keywords`)
112:
113: ) TYPE=MyISAM AUTO_INCREMENT=9158 ;
114:
115: --
116: -- Dumping data for table `tb_nodes`
117:
118: --
119: INSERT INTO `tb_nodes` VALUES (1, 1, 1, 62, 102, 'Home Page', 'Home', '/',
120:
121: '<p>My Site Slogan</p>', '', 'home page', '<h1>Header 1 </h1>\r\n<p>Lorem ipsum dolor sit amet,
consectetur adipiscing elit. In placerat vestibulum ligula. Pellentesque habitant morbi tristique
senectus et netus et malesuada fames ac turpis egestas. Praesent ut pede. Integer mauris neque, sodales
eu, interdum vitae, dapibus eget, augue. Nullam ligula felis, aliquam ut, eleifend sed, tincidunt et,
lacus. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Sed arcu.
Maecenas vehicula bibendum felis. Nulla neque libero, molestie nec, pellentesque ac, varius sed, neque.
Aenean turpis tellus, pretium a, nonummy a, pretium ac, velit. Nunc rutrum. Sed tempus eros a lectus.
Fusce iaculis. Ut quam massa, lacinia et, tincidunt eget, cursus a, libero. Aenean sagittis leo eget diam.
122:
123: </p>\r\n<h2>Header 2 </h2>\r\n<p>Morbi pulvinar sodales ipsum. Maecenas
124:
125: molestie augue eget orci. Morbi massa dolor, malesuada eu, porta eget,
126:
127: fringilla quis, elit. Nulla dapibus purus vitae libero. Phasellus luctus, lorem sed auctor aliquam,
ipsum mi gravida nunc, eu ullamcorper quam magna
128:
129: vel felis. Pellentesque pharetra ligula et nibh. Sed et nunc. Quisque ut orci non pede semper
sagittis. Donec vitae purus eu lacus ullamcorper auctor.
130:
131: Aliquam ac arcu. Nunc eget enim. Fusce interdum. Nullam luctus nibh ut mauris. Pellentesque at mi
quis ipsum euismod consectetur. Sed purus.
132:
133: </p>\r\n<h3>Header 3 </h3>\r\n<p>Donec vel leo vel odio luctus consectetur. Nulla facilisi. Donec
imperdiet. Aenean suscipit pretium libero. Aenean nunc eros, volutpat quis, vulputate ac, sagittis non,
ante. Maecenas vel enim vel enim tristique sagittis. Maecenas vel risus at turpis ullamcorper venenatis.
Mauris cursus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Suspendisse in orci. Duis id nisl
in est aliquet condimentum. Proin ut eros.
```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```

134:
135: </p>\r\n<h4>Header 4</h4>\r\n<h5>Header 5</h5>\r\n<h6>Header
136:
137: 6</h6>\r\n<p>Lists</p>\r\n<ul>\r\n <li>Lorem ipsum dolor sit amet, consectetur adipiscing elit.
</li>\r\n <li>Nam porttitor libero nec mi.
138:
139: </li>\r\n <li>Pellentesque congue purus sed sapien. </li>\r\n
140:
141: <li>Aliquam rutrum odio at leo. </li>\r\n <li>Aliquam a neque ac ipsum consectetur porta. </li>\r\n
<li>Nulla dictum ullamcorper ante.
142:
143: </li>\r\n</ul>\r\n<p>&nbsp;</p>\r\n<p>&nbsp;</p>\r\n<ul>\r\n <li>Quisque nonummy nunc bibendum nisl.
</li>\r\n <li>Nullam imperdiet diam quis sem.
144:
145: </li>\r\n <li>Aliquam hendrerit nisl id eros. </li>\r\n</ul>', 1, 0, '',
146:
147: 'Welcome to the home page', 1078155087, 1077944400, 0, '', 0, 1, 2,
148:
149: 1147188962, 2, 1, 0, '', '', 'almostspring');
150:
151: ....
152:
153: INSERT INTO `tb_nodes` VALUES (9101, 1, 51, 52, 3, 'Sample Announcement Item
154:
155: Two', 'Sample Announcement Item Two', '/1142098294', '', '', '', '<p>This is
156:
157: a sample announcement item.&nbsp;    Note, it does not appear as a link in the normal navigation
section of your page, it does not appear in your child content summary, and it does not appear in your
site map.&nbsp;    Announcement items are used to highlight events or important announcements.&nbsp;   Note, the
announcement item is shared on all pages below the page it is attached to.',
158:
159: 3, 9096, '', 'This is a sample announcement item. Note, it does not appear as a link in the normal
navigation section of your page, it does not appear in your child content summary, and it does not appear
in your site map.
160:
161: Announcement items are used to highlight events or important announcements.',
162:
163: 1142098352, 1142053200, 0, '', 0, 1, 2, 1142098598, 2, 8, 0, NULL, NULL, '');
164:
165: INSERT INTO `tb_nodes` VALUES (9105, 1, 32, 33, 3, 'Example Multiple Day
166:
167: Event One', 'Example Multiple Day Event One', '/1142099415', '', '', NULL,
168:
169: '<p>This is an example multiple day event.&nbsp;    The event spans from Monday to
Thursday.&nbsp;   &nbsp;   These dates are chosen in the publishing options tab.&nbsp;   ', 4, 9103, NULL, 'This is
an example multiple day event. The event spans from Monday to Thursday.', 1142099545, 1142053200, NULL,
170:
171: '2006/03/20,2006/03/21,2006/03/22,2006/03/23', 0, 1, 2, 1142099728, 2, 2, 0,
172:
173: NULL, NULL, NULL);
174:
175: -- -----
176:

```

Dev snapshot of a CMS I wrote-

<http://seanbuscay.com/drupal-web-development/dev-snapshot-cms-i-wrote>

```
177: --
178: -- Table structure for table `tb_nodes_strid_seq_seq`
179:
180: --
181: CREATE TABLE `tb_nodes_strid_seq_seq` (
182:
183: `id` int(10) unsigned NOT NULL auto_increment,
184:
185: PRIMARY KEY (`id`)
186:
187: ) TYPE=MyISAM PACK_KEYS=0 AUTO_INCREMENT=9158 ;
```

Wow, you made it! Thanks for taking the time to read through this code sample. Have an excellent day.