第三章 PHP 操作数据库

案例 3-1 连表查询的实现

一、案例描述

1、考核知识点

SQL 的连表查询

2、练习目标

- 能够熟练使用连表查询从多个表中获取组合数据

3、需求分析

在实际项目中，员工的详细数据保存在员工表中，部门的详细数据保存在部门表中。如果需要在查询员工信息时，一并显示其所属的部门信息，就需要同时获取员工表和部门表两张表的数据，这样的查询在 MySQL 中就被称为连表查询。下面在教材【案例 11】的基础上，演示连表查询的实现。

4、设计思路

1） 创建部门信息表，该表用来保存部门的相关信息。

2） 修改用户表，删除部门名称字段，添加部门 ID 字段。

3） 修改 showList.php 文件，重新组合 SQL 语句。

4） 修改 list_html.php 文件，主要修改表格中输出的字段信息，以符合查询到的数据。

二、案例实现

1、创建部门信息表，SQL 语句如下：

```
create table `dept_info` (
  `d_id` int unsigned primary key auto_increment,
  `d_name` varchar(20) not null,
  `d_leader_id` int unsigned not null
)default charset=utf8;
```

在上述 SQL 语句中，d_id 表示部门编号，将其声明为无符号的 int 类型，并作为该表的主键且为自增形式。d_name 表示部门名称，将其声明为 varchar 类型。d_leader_id 表示部门负责人的 id，该字段为之后查询部门详细信息时提供查询条件。

接下来向该表中插入数据，用来做查询测试，插入的 SQL 语句如下：

```
INSERT INTO `dept_info` VALUES
 (1, '开发部', 13), (2, '媒体部', 7), (3, '人事部', 10),(4, '后勤部', 3),
 (5, '市场部', 6),  (6, '运维部', 9), (7, '销售部', 8);
```

2、修改用户表

删除用户表中的部门名称字段，再添加部门 ID 字段，SQL 语句如下：

```
drop table if exists emp_info;
create table `emp_info` (
```

```
  `e_id` int unsigned primary key auto_increment,
  `e_name` varchar(20) not null,
  `d_id` int unsigned not null,
  `date_of_birth` timestamp not null,
  `date_of_entry` timestamp not null
)default charset=utf8;
```

完成用户表修改后，向其中添加用户数据，此时部门名称字段被部门 ID 字段代替，因此员工添加的 SQL 语句如下：

```
insert into `emp_info` values
 (1, '小红', 1, '2015-4-9 17:51:00', '2015-4-9 17:52:00'),
 (2, '李四', 5, '2008-4-3 13:33:00', '2013-10-24 17:53:00'),
 (3, '王五', 4, '2008-4-3 13:33:00', '2015-4-21 13:33:00'),
 (4, '赵六', 4, '2008-4-3 13:33:00', '2015-3-20 17:54:00'),
 (5, '小兰', 2, '1989-5-4 17:33:00', '2012-6-18 17:54:00'),
 (6, '小新', 5, '1993-9-18 17:36:00', '2015-2-28 17:36:00'),
 (7, '小白', 2, '1991-10-17 17:37:00', '2014-8-16 17:37:00'),
 (8, '小智', 7, '1987-6-20 17:37:00', '2015-1-10 17:38:00'),
 (9, '大头', 6, '1991-2-14 08:49:00', '2014-7-12 08:49:00'),
 (10, '小明', 3, '1991-2-14 08:49:00', '2015-3-4 09:10:00'),
 (11, '小刘', 1, '1992-3-18 14:52:00', '2014-7-21 09:00:00');
```

3、重新组合 SQL 语句

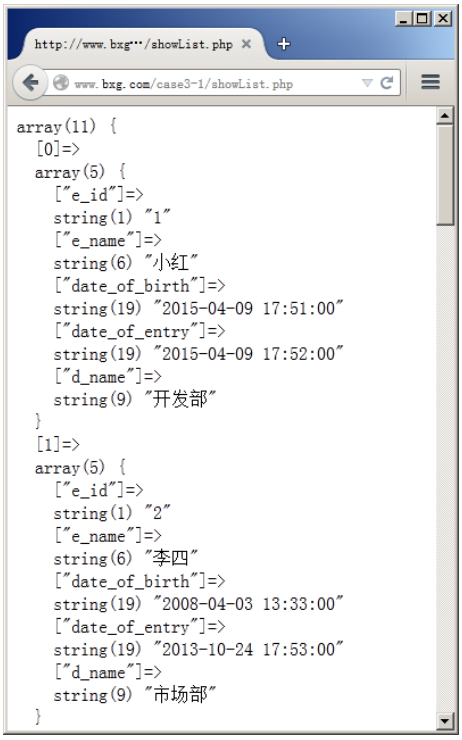
由于部门信息从员工表中抽取了出来，存放到了部门表中。因此想要获取员工数据的同时获取部门名称，就需要使用连接查询。下面修改 showList.php 文件，具体代码如下：

```
<?php
//声明文件解析的编码格式
header('content-type:text/html;charset=utf-8');
//连接数据库
$link = mysql_connect('localhost','root','123456');
//判断数据库连接是否成功，如果不成功则显示错误信息并终止脚本继续执行
if(!$link){
    die('连接数据库失败！'.mysql_error());
}
//设置字符集，选择数据库
mysql_query('set names utf8');
mysql_query('use itcast');
//准备 SQL 语句
$sql = 'select emp.e_id,emp.e_name,emp.date_of_birth,emp.date_of_entry,dept.d_name
    from emp_info as emp
    left join dept_info as dept
    on emp.d_id = dept.d_id';
//执行 SQL 语句，获取结果集
$res = mysql_query($sql,$link);
//定义员工数组，用以保存员工信息
$emp_info = array();
```

```
//遍历结果集，获取每位员工的详细数据
while($row = mysql_fetch_assoc($res)){
    //把从结果集中取出的每一行数据赋值给$emp_info 数组
    $emp_info[] = $row;
}
//设置常量，用以判断视图页面是否由此页面加载
define('APP', 'itcast');
//加载视图页面，显示数据
require './list_html.php';
```

上述代码中，变化的只有 SQL 语句。在该 SQL 语句中，为 emp_info 表定义别名 emp，为 dept_info 表定义别名 dept，通过别名来标识要获取的字段。由于获取的是员工信息以及员工所属的部门名称，因此该查询以 emp_info 表为主，而 emp_info 表又在 dept_info 之前，因此使用 left join 进行左连接查询。最后需要使用 on 关键字标识出两个表的关联条件。

为了让大家直观的感受到连表查询，下面将该 SQL 语句查询获取到的数据进行打印，通过观察获取的数据来体会连表查询的作用，查询结果下图所示。



从上图显示的两组数据可以看出，本不属于用户表数据的 d_name（部门名称）也被获取到了，这就是连表查询下的结果。

4、修改视图文件

由于表结构的变化，获取到的数据字段名也随之发生了变化。因此需要修改视图文件，按照查询到的字段来显示表单信息，具体代码如下：

```
<?php if (!defined('APP')) die('error!'); ?>
<!doctype html>
<html>
    <head>
        <meta charset="utf-8">
        <title>员工信息列表</title>
```

```
        </head>
        <body>
            <div>
                <div>员工信息列表</div>
                <table border="1">
                    <tr>
                        <th width="5%">ID</th><th>姓名</th><th>所属部门</th><th>出生日期</th><th>
入职时间</th><th width="25%">相关操作</th>
                    </tr>
                    <?php if (!empty($emp_info)) { ?>
                        <?php foreach ($emp_info as $row) { ?>
                            <tr>
                                <td><?php echo $row['e_id']; ?></td>
                                <td><?php echo $row['e_name']; ?></td>
                                    <!–下面一行就是被修改的部分-->
                                <td><?php echo $row['d_name']; ?></td>
                                <td><?php echo $row['date_of_birth']; ?></td>
                                <td><?php echo $row['date_of_entry']; ?></td>
                                <td><div><span><img src="images/edt.gif" width="16" height="16"
/>编辑   <img src="images/del.gif" width="16" height="16" />删除</span></div></td>
                            </tr>
                        <?php } ?>
                    <?php } else { ?>
                        <tr><td colspan="6">暂无员工数据! </td></tr>
                    <?php } ?>
                </table>
            </div>
        </body>
    </html>
```

上述代码中，主要修改的是代表部门名称的部分，将原本的$row['e_dept']修改为$row['d_name']即可。

三、案例总结

1、最好为参与连表查询的各表设置不同的别名，以便区分各表字段。
2、连表操作需要明确以那张表为主，以便选择 left join 或 right join，这关系到最终获取的数据形式。
3、连表操作需要明确两张表有关联的字段。

案例 3-2　动态获取部门信息

一、案例描述

1、考核知识点

SQL 的连表查询

2、练习目标

- 熟练掌握连表查询，能够根据需求修改现有代码以实现功能

3、需求分析

在添加员工信息时，一般员工所属部门是不允许用户手动输入的。而是通过动态查询数据库，获取到部门表数据，以下拉菜单的形式展示到表单中，以供选择。下面就以教材【案例 15】为例，完成员工添加及修改时动态获取部门信息的功能。

4、设计思路

1） 修改 showList.php 文件的 SQL 语句，使用连接查询完成数据获取。
2） 在显示添加表单前获取部门表数据。
3） 在添加员工页面中以下拉菜单来展现部门数据。
4） 修改 empAdd.php 文件中保存合法字段的$fields 变量。
5） 在显示修改表单前获取到部门数据。
6） 在修改员工页面中以下拉菜单来展现部门数据。
7） 修改 empUpdate.php 文件中保存合法字段的$fields 变量。

二、案例实现

1、修改 showList.php 文件的 SQL 语句，使用连接查询完成数据获取，具体代码如下：

```
<?php
//声明文件解析的编码格式
header('content-type:text/html;charset=utf-8');
require './public_function.php';
//初始化数据库
dbInit();
//准备 SQL 语句
$sql = 'select emp.e_id,emp.e_name,emp.date_of_birth,emp.date_of_entry,dept.d_name
    from emp_info as emp
    left join dept_info as dept
    on emp.d_id = dept.d_id';
//定义员工数组，用以保存员工信息,执行 SQL 语句，获取结果集
$emp_info = fetchAll($sql);
//设置常量，用以判断视图页面是否由此页面加载
define('APP', 'itcast');
//加载视图页面，显示数据
require './list_html.php';
```

2、修改 empAdd.php 文件，在显示添加表单前需要获取部门表数据，具体代码如下：

```
<?php
//声明文件解析的编码格式
header('content-type:text/html;charset=utf-8');
//引入功能函数文件
require './public_function.php';
//初始化数据库
```

```
dbInit();
//判断是否有表单提交
……
//没有表单提交时，显示员工添加页面,从部门表获取部门信息
//编写 SQL 语句，用于查询部门表数据
$sql = 'select * from dept_info';
//调用 fetchAll()函数，执行 SQL 并进行数据处理，把处理后的部门数据赋值给$dept_info
$dept_info = fetchAll($sql);
define('APP', 'itcast');
require './add_html.php';
```

上述代码中，在展示添加员工表单前，先通过执行“select * from dept_info”获取到了部门信息，然后再载入员工添加页面 add_html.php。

3、修改 add_html.php 文件，在添加员工页面中以下拉菜单来展现部门数据，关键部分代码如下：

```
<form method="post" action="./empAdd.php">
   <table>
      <tr><th>姓名：</th><td><input type="text" name="e_name" /></td></tr>
      <tr>
        <th>所属部门：</th>
         <td>
<!--以下为 select 下拉框的关键代码-->
<select name="d_id">
<?php foreach ($dept_info as $row) { ?>
<option value="<?php echo $row['d_id']; ?>"><?php echo $row['d_name']; ?></option>
<?php } ?>
</select>
<!—select 下拉框结束-->
         </td>
      </tr>
      <tr>
        <th>出生年月：</th>
        <td><input id="date_of_birth" name="date_of_birth" type="text" ></td>
      </tr>
     <tr>
        <th>入职日期：</th>
        <td><input id="date_of_entry" name="date_of_entry" type="text" ></td>
     </tr>
    <tr>
        <td colspan="2">
            <input type="submit" value="保存资料"/>
            <input type="reset" value="重新填写"/>
         </td>
    </tr>
  </table>
</form>
```

在上述代码中，主要将原本部门的输入文本框变为了下拉菜单，在其中使用 foreach 将保存部门信息的数组进行遍历，保存到<option>标签中。需要注意的是<option>标签的 value 值需要保存的是部门 id，而非部门名称，部门名称仅仅是为了向使用者友好的显示部门而显示在下拉菜单中的。

此时访问 showList.php 文件，并点击“添加员工”链接，跳转到员工添加页面，运行结果如下图所示。

从上图可以看到，员工所属部门已经全部获取并显示到了下拉菜单中。

4、由于表示员工所属部门的表单元素的 name 属性不再是 e_depte，而变成了 d_id。因此需要修改 empAdd.php 文件中保存合法字段的$fields 变量，修改代码如下：

```
//声明变量$value，用来保存字段信息
$fields = array('e_name', 'd_id', 'date_of_birth', 'date_of_entry');
```

完成上述修改后，动态获取部门信息并添加员工数据的功能就可以实现了。

5、修改 empUpdate.php 文件，与员工添加类似，在显示修改表单前先获取到部门数据，具体代码如下：

```
<?php
//声明文件解析的编码格式
header('content-type:text/html;charset=utf-8');
//引入功能函数文件
require './public_function.php';
//获取数据库连接
dbInit();
//获取要编辑的员工的 id
$e_id = isset($_GET['e_id']) ? intval($_GET['e_id']) : 0;
//判断是否有 POST 数据提交
if(!empty($_POST)){
    ……
    }
}else{
    //当没有表单提交时，查询当前要编辑的员工信息，展示到页面中
    //编写 SQL 语句，查询相应 ID 的员工数据
    $sql = "select * from `emp_info` where `e_id` = $e_id";
    //使用 fetchRow()函数处理数据
```

```
        $emp_info = fetchRow($sql);
        //编写 SQL 语句，查询所有部门信息数据
        $sql = 'select * from dept_info';
        $dept_info = fetchAll($sql);
        //显示员工修改页面
        define('APP', 'itcast');
        require './update_html.php';
    }
```

上述代码中，在展示修改员工表单前，先通过执行“select * from dept_info”获取到了部门信息，然后再载入员工添加页面 update_html.php。

6、修改 update_html.php 文件，与员工添加页面类似，需要修改表单以显示部门信息，关键部分代码如下：

```
    <form method="post">
        <input type="hidden" name="e_id" value="<?php echo $emp_info['e_id']; ?>"/>
        <table>
            <tr><th> 姓 名 ： </th><td><input type="text" name="e_name" value="<?php echo $emp_info['e_name']; ?>"/></td></tr>
            <tr>
                <th>所属部门：</th>
             <!–下拉菜单开始-->
                <td><select name="d_id">
                        <?php foreach ($dept_info as $row) { ?>
         <!–在输出每个部门信息时，判断是否为该员工当前所属部门，如果是设置为默认-->
       <option value="<?php echo $row['d_id']; ?>" <?php if ($emp_info['d_id'] == $row['d_id']) echo "selected='selected'"; ?>><?php echo $row['d_name']; ?></option>
                        <?php } ?>
                    </select></td>
             <!–下拉菜单结束-->
            </tr>
            <tr><th> 出 生 年 月 ： </th><td><input id="date_of_birth" name="date_of_birth" type="text" value="<?php echo $emp_info['date_of_birth']; ?>"></td></tr>
            <tr><th> 入 职 日 期 ： </th><td><input id="date_of_entry" name="date_of_entry" type="text" value="<?php echo $emp_info['date_of_entry']; ?>"></td></tr>
            <tr><td colspan="2">
                    <input type="submit" value="保存资料"/>
                    <input type="reset" value="重新填写"/>
                </td></tr>
        </table>
    </form>
```

上述代码，显示部门信息的过程与员工添加基本一致。唯一不同的是，需要在遍历部门信息时，与员工所属部门 id 进行比较，以便将员工当前所属的部门显示为下拉菜单的默认选项。

7、由于表示员工所属部门的表单元素的 name 属性不再是 e_depte，而变成了 d_id。因此需要修改 empUpdate.php 文件中保存合法字段的$fields 变量，修改代码如下：

```
    //声明变量$value，用来保存字段信息
```

```
$fields = array('e_name', 'd_id', 'date_of_birth', 'date_of_entry');
```

至此就完成了在添加和修改员工时，动态获取部门信息的功能。

三、案例总结

1、在 select 下拉菜单中，option 的值表示员工所属的部门，而员工表中保存的部门信息是其部门 id，因此 option 的值也是部门 id。

2、在添加和修改员工信息时，都需要经过字段合法性验证。而在修改表结构后，员工表字段发生了变化，因此需要修改保存合法字段的变量数组。

案例 3-3　实现复杂分页效果

一、案例描述

1、考核知识点

分页显示的业务逻辑

2、练习目标

- 根据需求完成指定形式的分页样式

3、需求分析

分页链接的显示样式也是一个网站中相对重要的环节，好的分页样式能增加用户的体验。下面就在教材【案例 13】的基础上，修改生成分页链接的方法，完成一个相对灵活的分页样式。

该分页样式由“首页”、“尾页”以及指定数量的中间页码组成。具体效果如下图所示。

首页显示效果

尾页显示效果

4、设计思路

1） 修改 page_function.php，能够灵活控制显示的页码数量。
2） 循环输出指定数量的页码链接。
3） 判断显示的页码前后是否还有分页，并进行相关处理。
4） 为分页链接添加相关样式。

二、案例实现

1、修改 page_function.php，能够灵活控制显示的页码数量。具体代码如下：

```
<?php
/**
 * 分页链接生成函数
 * $page  URL 传递的 page 值
 * $max_page  最大页码值
 * $show_page_num 页面中显示多少页码
 */
function makePageHtml($page, $max_page, $show_page_num = 5) {
……
}
```

在上述代码中，为 makePageHtml()函数添加一个形式参数$show_page_num，该参数用来指定页面中显示的页码数量，并为其设置默认值为 5。

2、循环输出指定数量的页码链接。

这一步是实现整个分页样式至关重要的一步，首先需要认真分析分页链接该如何循环输出。我们需要实现的是：一次显示指定数量的页码，例如“1 2 3 4 5”、“6 7 8 9 10”。

这里面就需要我们确定 2 个问题：

① 当前显示的页码第一位的数字是几
② 当前显示的页码最后一位的数字是几

那么根据现有条件，我们可以知道页面中显示的页码数量、当前选择的页码值以及最大页码值，这三个数值都是在调用 makePageHtml()函数时需要传递的参数。

通过分析我们可以把“1 2 3 4 5”、“6 7 8 9 10”这样的页码看做一个小组，每组的最小值就是当前页码的第一位数字，最大值就是最后一位数字。只要我们知道当前显示的页码是第几组，就可以算出当前显示的第一位和最后一位。

下面定义$page_grep 表示当前页码所属组，通过分析其与页码数量之间的数学关系可以得到下面的表达式：

```
$page_grep = ceil($page / $show_page_num);
```

当前页码属于第几组，可以通过$page 和$show_page_num 相除并向上取整来确定。

接下来就可以确定当前分组的第一位页码的数字，下面定义$i 表示该数字，通过分析可以得到下面的表达式：

```
$i = $show_page_num * ($page_grep - 1) + 1;
```

然后确定当前分组的最后一位页码的数字，下面定义$max 表示该数字，通过分析可以得到下面的表达式：

```
$max = $page_grep * $show_page_num;
```

而实际上当前分组最后一位页码的数字可能小于$max，因此需要加以判断，判断表达式如下：

```
$max_i = ($max <= $max_page) ? $max : $max_page;
```

最后就可以根据上述条件进行循环输出页码，具体代码如下：

```
<?php
/**
 * 分页链接生成函数
 * $page  URL 传递的 page 值
 * $max_page  最大页码值
 * $show_page_num 页面中显示多少页码
 */
function makePageHtml($page, $max_page, $show_page_num = 5) {
    //设置变量，保留 GET 参数
    $prams = '';
    //删除 GET 参数中的 page
    unset($_GET['page']);
    //循环遍历$_GET 数组，重新拼接 URL 中的 GET 参数
    foreach ($_GET as $k => $v) {
        $prams .= "$k=$v&";
    }
    //重新拼接分页链接的 html 代码
    $page_html = '<a href="?' . $prams . 'page=1">首页</a> ';
    //使用当前页页码和页面显示页码量计算出
    $page_grep = ceil($page / $show_page_num);
    $i = $show_page_num * ($page_grep - 1) + 1;
    $max = $page_grep * $show_page_num;
    $max_i = ($max <= $max_page) ? $max : $max_page;
    for ($i; $i <= $max_i; $i++) {
        $page_html .= '<a href="?' . $prams . 'page=' . $i . '">' . $i . '</a> ';
    }
    $page_html .= '<a href="?' . $prams . 'page=' . $max_page . '">尾页</a> ';
    //返回分页链接
    return $page_html;
}
```

完成上述步骤后，就可以随意控制显示的页码数量进行分页了，例如我们选择显示 4 个页码，则结果如下图所示。

3、判断显示的页码前后是否还有分页，并进行相关处理。

此时还有一个问题没有解决，第 4 页之后还有分页该怎么办。这就需要我们在循环页码的时候进行判断，如果当前显示的页码之后还有分页，则在页码最后显示“>”表示后面还有内容，例如“1 2 3 4 5 >”。如果当前显示的页码之前还有分页，则在页码最前显示“<”表示前面还有内容，例如“< 6 7 8 9 10”。关键部分代码如下：

```
    for ($i; $i <= $max_i; $i++) {
        //判断是否有上一页
        if (($i + $show_page_num - 1) % $show_page_num == 0 && $i > 1) {
            $page_html .= '<a href="?' . $prams . 'page=' . ($i - 1) . '">&lt;</a> ';
        }
        //输出当前页页码
        $page_html .= '<a href="?' . $prams . 'page=' . $i . '">' . $i . '</a> ';
        //判断是否有下一页
        if ($i % $show_page_num == 0 && $i < $max_page) {
            $page_html .= '<a href="?' . $prams . 'page=' . ($i + 1) . '">&gt;</a> ';
        }
    }
```

最后为了突出显示当前是哪一页，还需要判断并添加不同的样式，最终分页代码如下：

```
    for ($i; $i <= $max_i; $i++) {
        //判断是否有上一页
        if (($i + $show_page_num - 1) % $show_page_num == 0 && $i > 1) {
            $page_html .= '<a href="?' . $prams . 'page=' . ($i - 1) . '">&lt;</a> ';
        }
         //判断是否为当前选中页
        if($i == $page){
            $page_html .= '<a class = "curl" href="?' . $prams . 'page=' . $i . '">' . $i .
'</a> ';
        }else{
            $page_html .= '<a href="?' . $prams . 'page=' . $i . '">' . $i . '</a> ';
        }
        //判断是否有下一页
        if ($i % $show_page_num == 0 && $i < $max_page) {
            $page_html .= '<a href="?' . $prams . 'page=' . ($i + 1) . '">&gt;</a> ';
        }
    }
```

4、为分页链接添加相关样式，并修改 showList.php 文件，调用 makePageHtml()函数来生成分页链接的 html 代码。

html 模板文件中有关分页链接的样式代码如下：

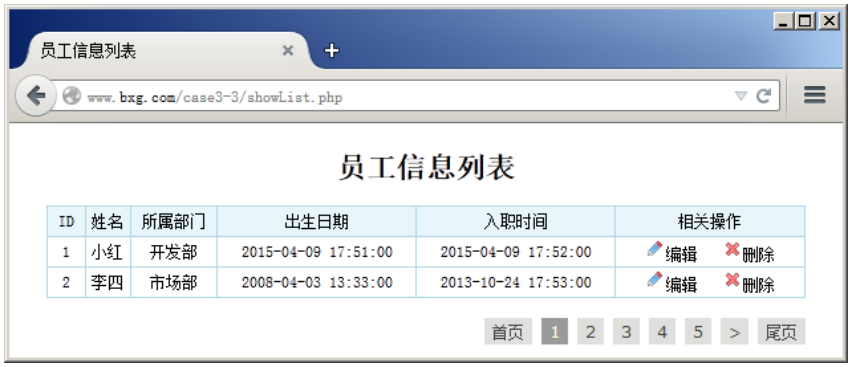
```
    .page{font-size:12px;float:right;}
    .page a{padding: 2px 6px;background-color: #DFDFDD;color:#454545;display: block;float:
left;margin-left:5px;text-decoration: none;font-family: verdana;}
    .page .curl{background-color: #999999;color: #FFFFE0;}
    .page a:hover{background-color: #999999;color: #FFFFE0;}
```

showList.php 文件调用 makePageHtml()函数，代码如下：

```
    <?php
```

```
    header('content-type:text/html;charset=utf-8');
    require './page_function.php';
    //获取数据库连接并选择数据库
    mysql_connect('localhost', 'root', '123456') or die('连接数据库失败! ' . mysql_error());
    mysql_query('set names utf8');
    mysql_query('use itcast');
    //获取最大分页数
    $page_size = 2;
    $res = mysql_query('select count(*) from emp_info');
    while ($row = mysql_fetch_row($res)) {
        $count = $row[0];
    }
    $max_page = ceil($count / $page_size);
    //获取当前选择的页码，并作容错处理
    $page = isset($_GET['page']) ? intval($_GET['page']) : 1;
    $page = $page > $max_page ? $max_page : $page;
    $page = $page < 1 ? 1 : $page;
    //组合分页链接
    $page_html = makePageHtml($page, $max_page);
    //拼接查询语句并执行，获取查询数据
    $lim = ($page - 1) * 2;
    $sql = "select emp.e_id,emp.e_name,emp.date_of_birth,emp.date_of_entry,dept.d_name from
emp_info as emp
    left join dept_info as dept
    on emp.d_id = dept.d_id limit {$lim},{$page_size}";
    $res = mysql_query($sql);
    //读取数据并作相关处理
    $emp_info = array();
    while ($row = mysql_fetch_assoc($res)) {
        $emp_info[] = $row;
    }
    //设置常量，用以判断视图页面是否由此页面加载
    define('APP', 'itcast');
    //载入视图页面
    require './list_html.php';
```

完成上述修改后，复杂分页样式效果就制作完成，如下图所示。



三、案例总结

1、分页样式需要考虑的就是分页相关数据间的数学关系，只要清楚了其中的数学关系就能够轻松完成复杂分页显示。