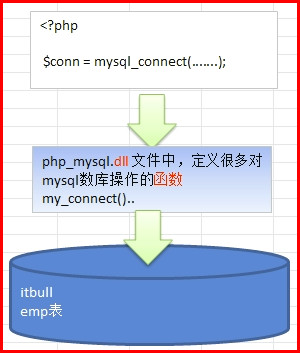
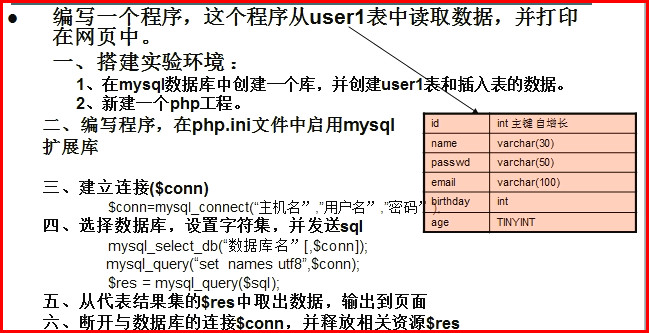
# PHP的数据库扩展

***第一章 MySQL的扩展(了解)***

1.1 基本介绍

MySQL扩展是指php程序可以通过mysql扩展来完成对mysql数据库的操作,对应dll文件是php\_mysql.dll文件.该文件在~php/ext

mysql扩展运行原理图

mysql扩展使用的流程

***第二章 MySQLi扩展***

2.1 MySQLi基本介绍

1.mysqli扩展和mysql扩展都可以完成对mysql数据库的操作

2.mysqli扩展可以看作是mysql扩展的增强版

3.mysqli扩展的性能比mysql扩展性能好(php7彻底放弃mysql扩展),新项目开发中不再使用mysql扩展,新项目主要采用PDO与MySQLi

4.MySQLi扩展***支持面向对象开发***,mysqli也支持面向过程(尽量不要使用)

5.使用MySQLi扩张需要引用一个dll文件,该文件是php\_mysqli.dll,在我们php安装目录下的php7/ext/php\_mysqli.dll

2.2 mysqli扩展的使用原理图

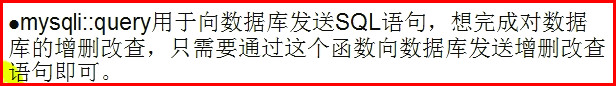


2.3 mysqli快速入门案例



$res表示SQL语句的执行结果

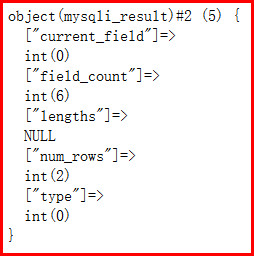
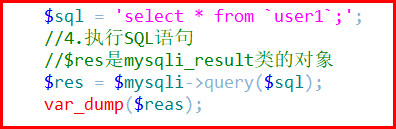
2.4 mysqli的细节说明



1.如果执行的是DML语句(增,删改),则返回bool值



2.如果执行的是DQL语句(查询).则返回查询结果***MySQLi\_Result结果集对象***



3.如何从mysqli result结果集对象取出查询的结果有如下方式:

***1.mysqli\_result::fetch\_assoc***

***2. mysqli\_result::fetch\_row***

***3. mysqli\_result::fetch\_array***

***4. mysqli\_result::fetch\_object***



**实际开发中推荐使用第一种(关联数组)或第四种(对象)!!**

4.释放资源

mysqli\_result::free(void);

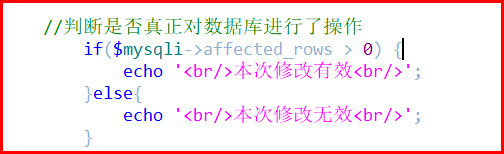
mysqli\_result::close(void);

mysqli\_result::free\_result(void)

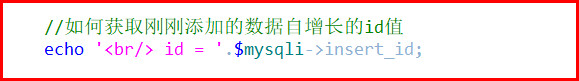
5.关闭连接

mysqli::close(void); 或者 过程化mysqli\_close($link);

6.如何判断DML语句是否真正影响到数据库



7.如何获取自增长字段的id值





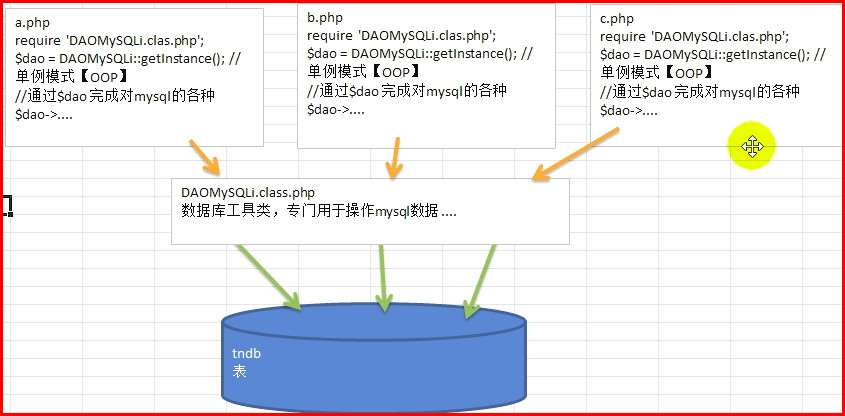
***第三章 DAOMySQLi.class.php工具类***

**3.1 基本介绍**

这是一个更加完善的数据库操作类,可以在以后的项目中直接使用.

使用场景: 我们在一个项目中会经常使用到对MySQL数据库的各种操作(CRUD),我们可以讲这些操作封装到一个类中,当我们需要操作数据库时,可以直接实例化一个对象,然后使用对象来操作数据库.

示意图:

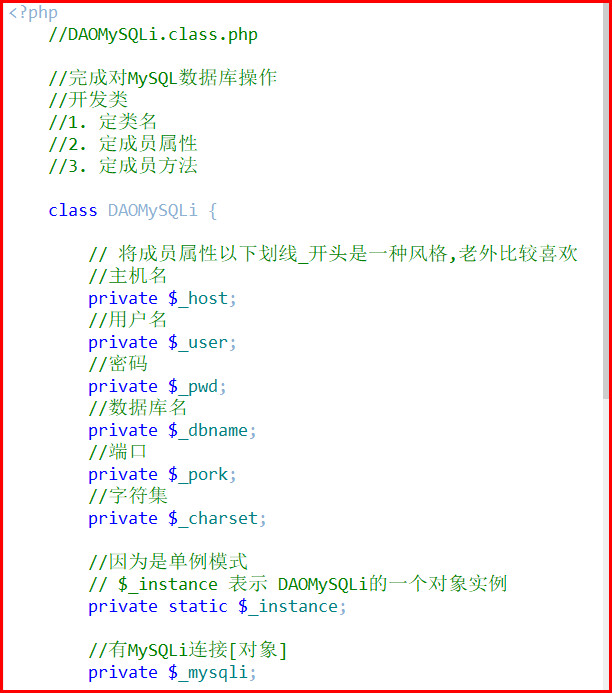


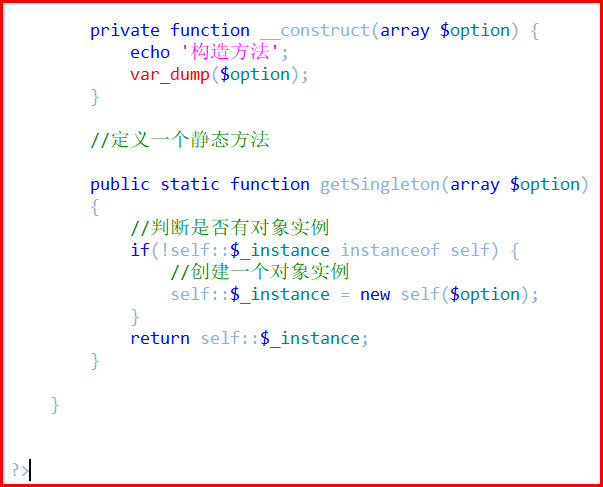
说明:

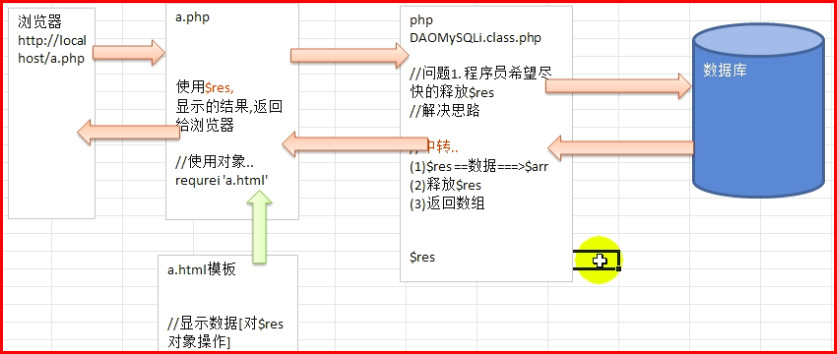
**1.DAO: data access object 数据访问对象,专门对MySQL操作**

**2.DAOMySQLi.class.php 单例模式**

**3.2 代码走起来(OOP/MySQL的SQL语句/封装)=>必须熟练手写**





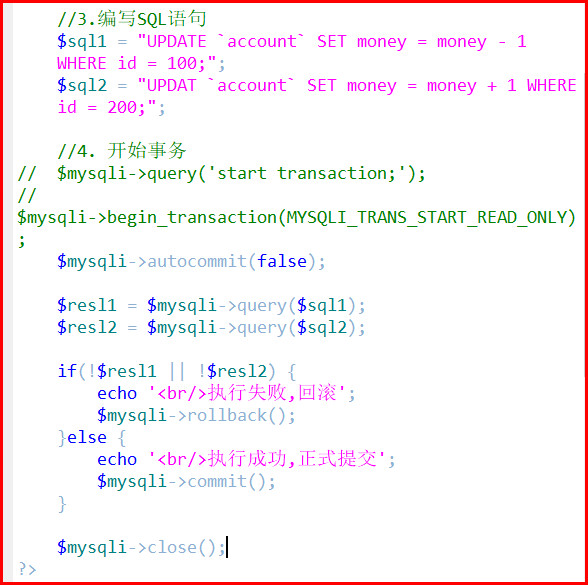


***第四章MySQLi扩展增强-----事务控制***

**4.1 基本说明**

事务用于保证数据的一致性,它由一组相关DML语句组成,该组的DML语句要么全部成功,要么全部失败.经典案例:网上的转账就是典型的事务控制来处理,保证数据的一致性

**4.2 案例演示**

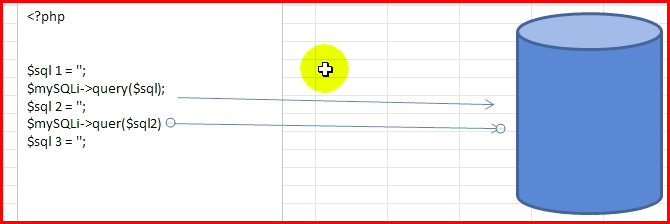


***第五章MySQLi扩展增强-----批量执行SQL语句***

**5.1 基本说明**

先看需求: 在一些项目中,需要一次性执行多条SQL语句,比如批量增加用户,采用单条发送SQL语句效率太低,可以采用批量SQL语句操作哦

传统执行SQL方式图:



批量执行SQL语句的基本语法

$sqls = “$sql1;$sql2;$sql3;...............”;

$mysqli->***multi\_query(***$sqls***);***

**说明:在拼接sqls语句时,最后的SQL语句不要给分号**;

**5.2 案例演示**

基本用法:



将3个用户一次性添加到mysql数据库中的account表中

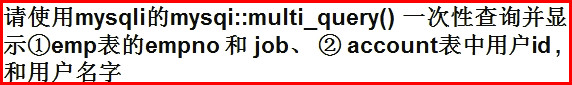


**5.3 批量执行DML语句细节说明**

1.批量执行SQL语句分成两大类(DML与DQL),这两大类之间不要混用;

2.批量执行DML语句,返回的结果是以第一条DML语句执行的结果为准,当某条DML语句错误了,后面代码就不在执行,但是错误之前的DML语句还是成功的.以此类推

**5.4 批量执行SELECT(DQL)语句**







用到函数有

***mysqli::multi\_query(); 发送多条SQL语句***

***mysqli::more\_results(); 只判断是否还有下一个结果集***

***mysqli::next\_result(); 取出下一个结果集***

练习:



2018-01-28_172947.jpg

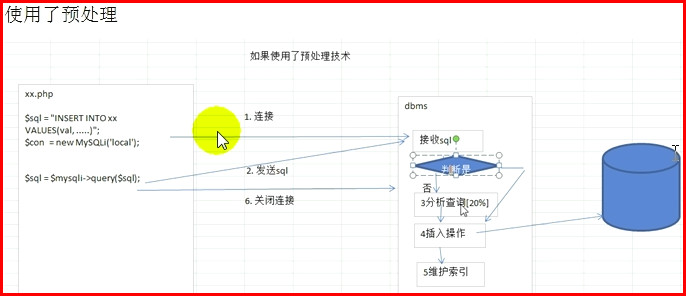
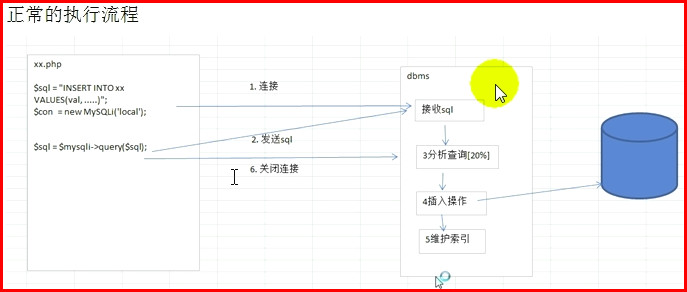


***第六章MySQLi扩展增强-----预处理技术***

**6.1 基本说明**

问题导出: 需要向mysql数据库添加100个用户,请问如何实现?(除了批量处理技术外)

**分析MySQL语句执行流程**



使用预处理技术的优点:

(与传统的执行SQL语句相比有以下优点)

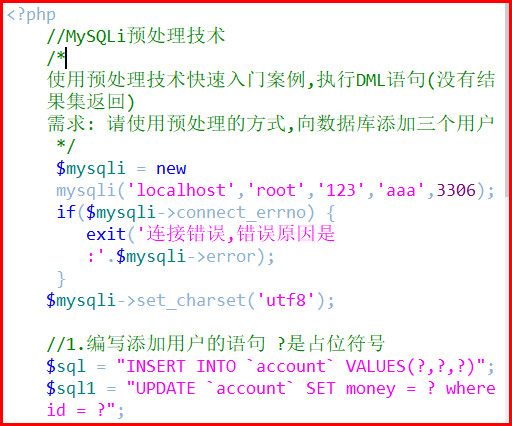
1.效率高,执行速度快(优化分析查询)

2.安全性高,可以防止SQL注入

**6.2 案例演示**

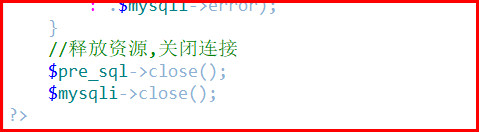
1.案例说明: 使用预处理技术快速入门案例,执行DML语句(没有结果集返回)

需求: 请使用预处理的方式,向数据库添加三个用户









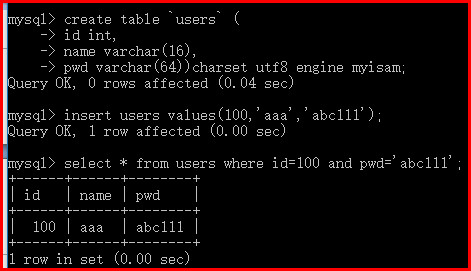
2.案例说明: 需求->使用预处理的方法,查询所有id>5的用户信息

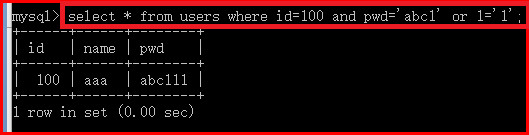
代码:



**6.3 预处理优越性---防止SQL注入**

看一个SQL注入的实际案例





解决注入的方式有

1.使用PDO;

2.使用一些好的工具类,该工具类有自身过滤SQL语句的功能(框架也带),

3.PHP的一些过滤函数比如addslashes(string $str),mysqli\_real\_escape\_string()等等;

4.使用预处理技术—案例演示

