1.

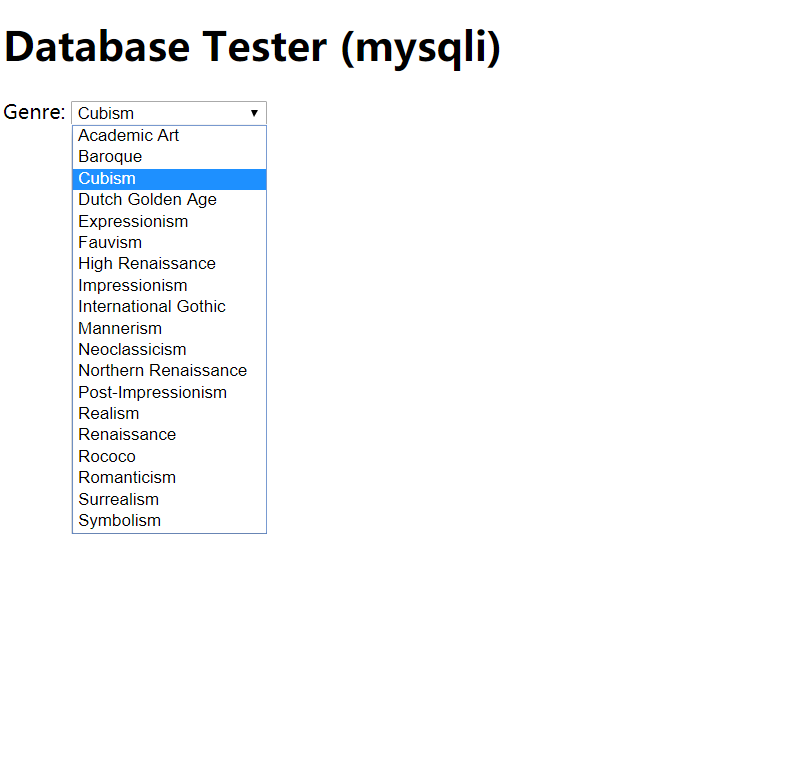
先声明一个pdo对象，并设置它的属性值为要读取的内容。

再声明一个字符串sql，它的值为搜索last name指令

然后声明一个result字符串，里面储存搜索的结果

当搜索到时，就把这个字符串push到result里面

Result打印出来即可

2.

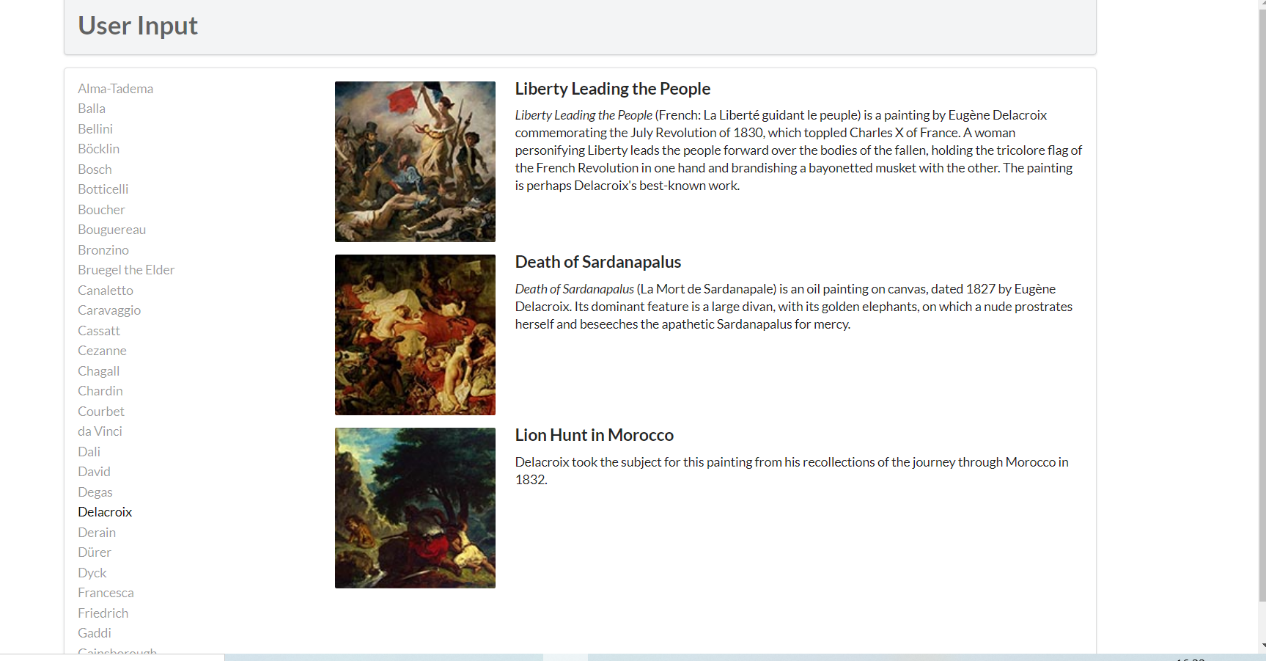
先声明一个connection变量，并设置他的属性为目标值

再声明一个sql的值为搜索指令

再利用fetch方法进行匹配

当匹配到时就把这个选项放置到option里面

即可完成检索

3.

先声明一些div用来放置照片和作者等信息

在相应的div执行outputePainting和outputArtists();函数即可将作者和图片打印到相应位置

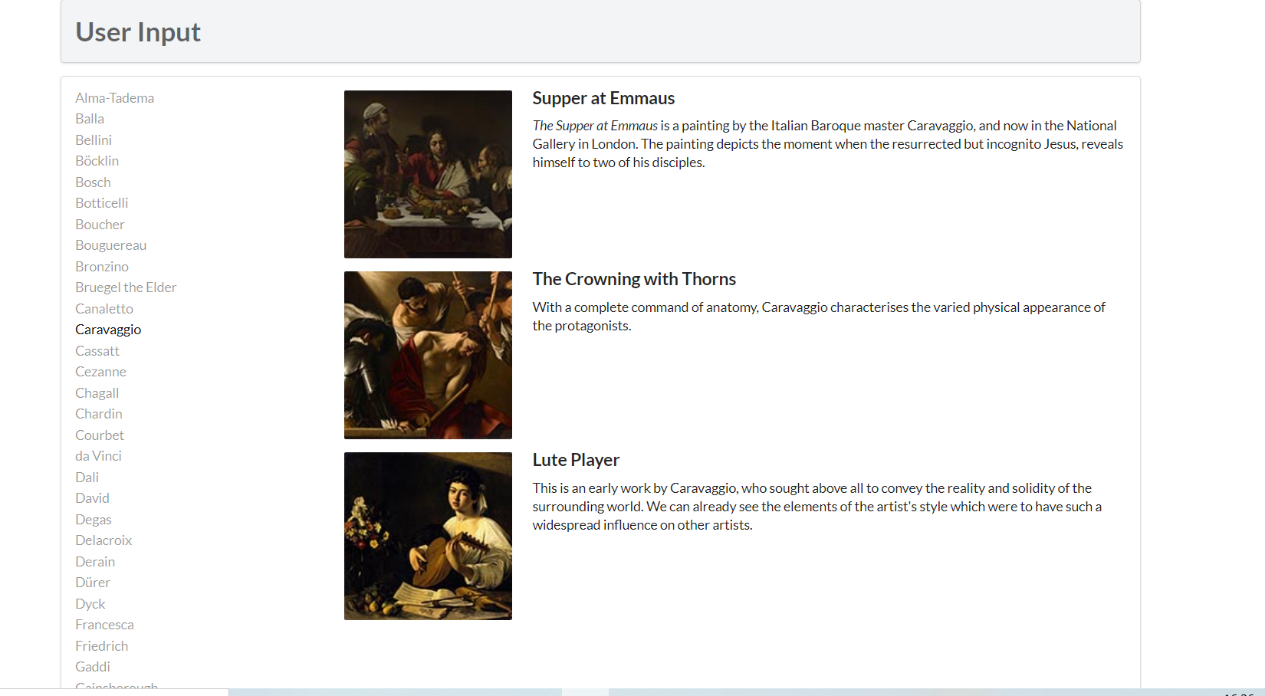
outputPaintings()

1. 检索方式与第一个方式相同，
2. Sql的值为要匹配的对象
3. while ($row = $result->fetch()) {
4. outputSinglePainting($row);
5. }
6. $pdo = null;
7. }当匹配到时就触发单独打印照片函数
8. 当没有匹配到时值为null

outputSinglePainting（）

1. 对传进来的row数组进行检索
2. 根据Ro6w数组写出图片的路径
3. 根据路径找出照片
4. 将照片打印到页面

即可完成单个图片的打印



**执行sql语句的方式**

1. **使用executeUpdate或者使用executeLargeUpdate方法来执行DDL和DML语句**

先创建一个数据库表，直接利用sql字符串对数据库操作

1. **使用execute方法执行SQL语句**

**execute()方法执行了三种SQL语句，分别是查询语句，DML语句与DDL语句，其中查询语句返回的是True值，根据true值从Statement对象中获取ResultSet对象，而根据false值，从Statement对象中获取getUpdateCount()方法的值，其运行效果为**

1. **PreparedStatement执行SQL语句**结构基本相似，值时执行插入时的值不同而已。对于这种情况，可以使用带占位符(?)参数的SQL语句来代替它们，PreparedStatement接口可以预编译SQL语句，预编译后的SQL语句被存储到PreparedStatement对象中，然后可以使用该对象对此高效地执行该语句

它的优势：

1.PreparedStatement预编译SQL语句，性能更好

2.PreparedStatement无须“拼接”SQL语句，编程更简单

　　3.PreparedStatement可以放置SQL注入，安全性更好

**4.** **CallableStatement调用存储过程**

若想编写存储过程，是使用CallableStatement对象，该对象是由Connection的perpareCall(String sql)方法来创建，调用存储过程的SQL语句总是这种格式：{call 存储过程名称(?,?,?.....)}，其中的问号作为存储过程参数的占位符，CallableStatement需要调用registerOutParameter(int index,int SQLTypes)方法来注册该参数，第二个参数是直接调用Types的类属性便可