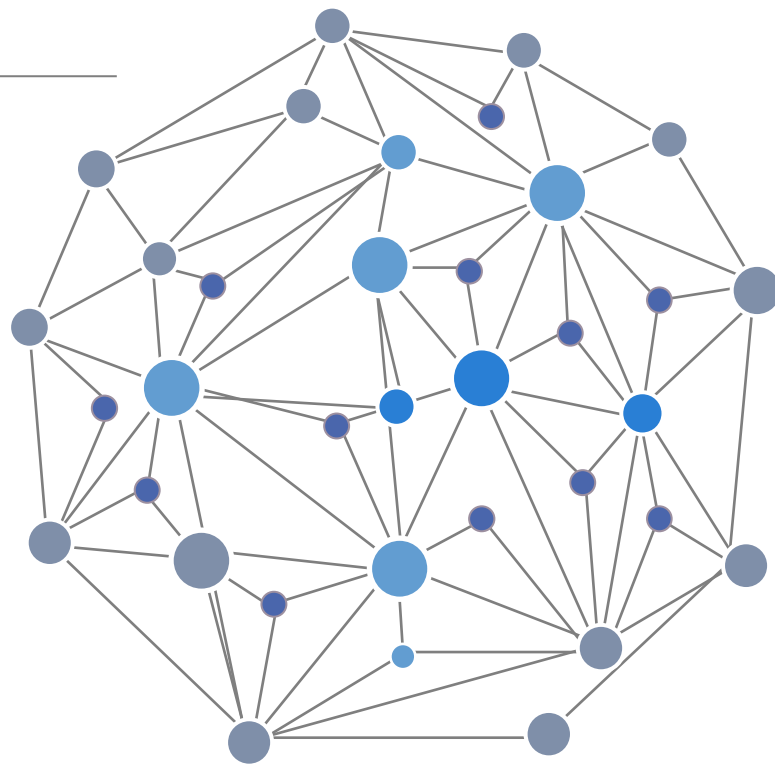

Web 程序设计

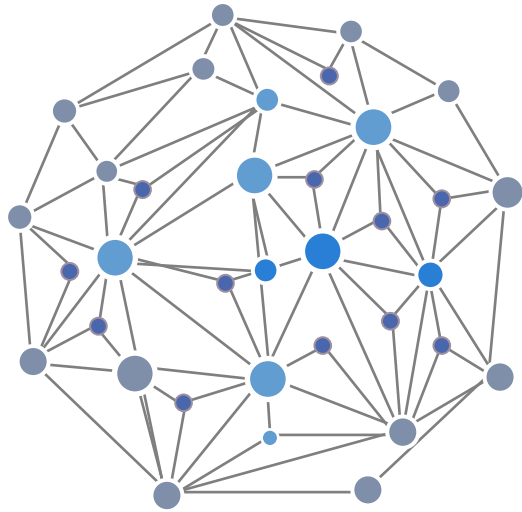
第八讲 PHP 编程 (3): 数据库编程

福州大学 计算机与大数据学院
软件工程系 陈昱



PHP & MySQL

- 数据库原理与操作回顾
- MySQL 的使用
- 在 PHP 中访问MySQL数据库
- MySQL 存储引擎
- MySQL 字符集问题
- 实例：简单博客系统



Database Terms

RDBMS 术语

- 数据库 database
- 表 table
- 列 (属性) column (attribute)
- 行 (记录) row (record)
- 主键 (primary key)
- 外键 (foreign key)

SQL DDL

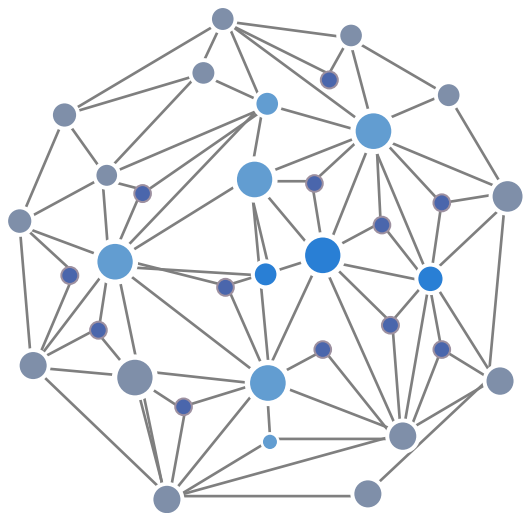
The Data Definition Language (DDL):

- CREATE TABLE
- ALTER TABLE
- DROP TABLE
- CREATE INDEX
- DROP INDEX

SQL DML

Data Manipulation Language (DML) :

- SELECT
- INSERT INTO
- UPDATE
- DELETE



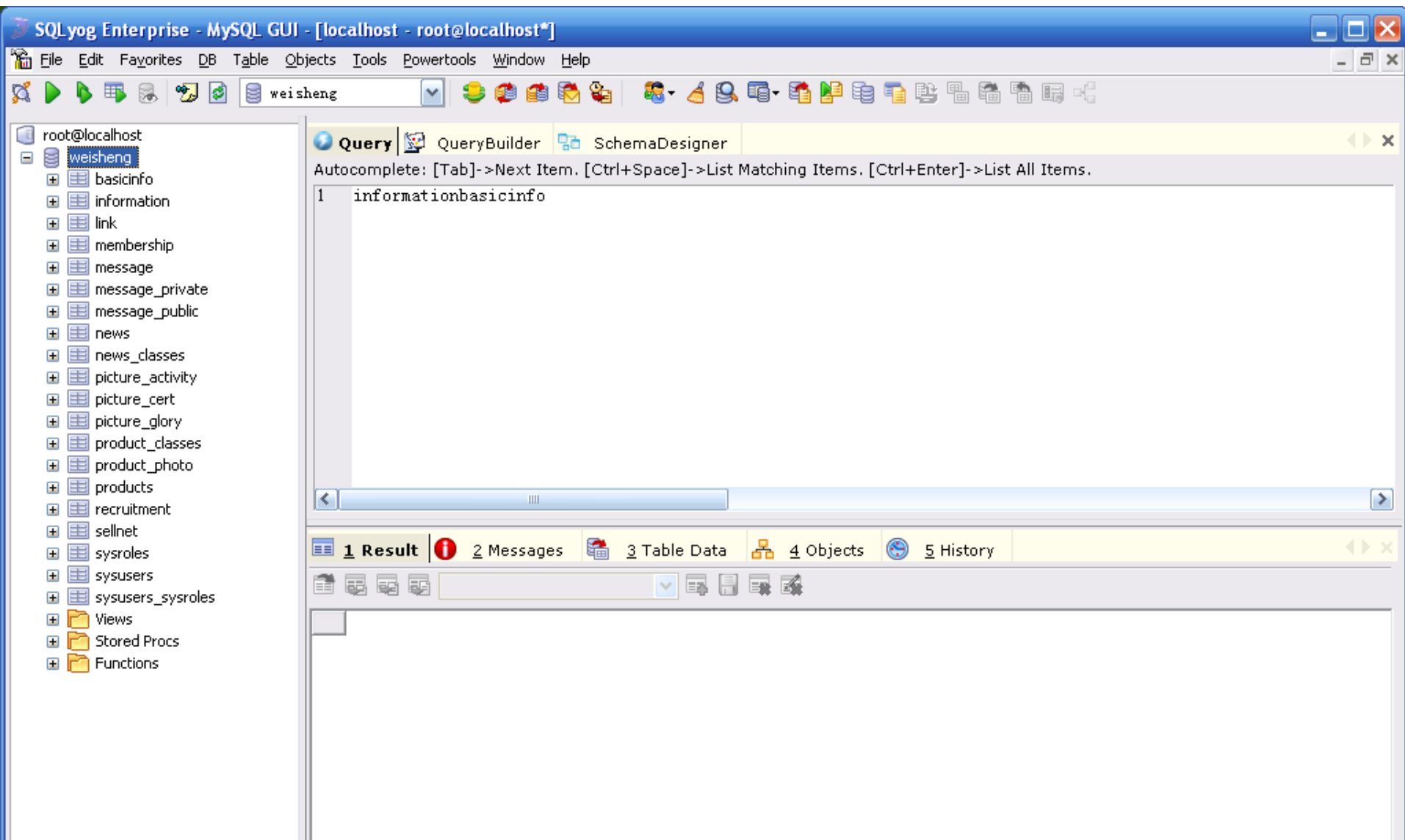
MySQL 数据库的使用

MySQL 的优点

- 支持绝大多数 ANSI 标准 SQL 语句
- 支持 Unicode 和 多种字符集
- 开源，低成本
- 与 Apache/PHP 一样被移植到许多操作系统中



使用 GUI 图形界面—SQLyog



使用 Web 界面—phpMyAdmin

phpMyAdmin

localhost » ssmis

结构 SQL 搜索 查询 导出 导入 操作 权限 程序 事件

表	操作	行数	类型
<input type="checkbox"/> admin_authassignment	浏览 结构 搜索 插入 清空 删除	14	InnoDB
<input type="checkbox"/> admin_authitem	浏览 结构 搜索 插入 清空 删除	22	InnoDB
<input type="checkbox"/> admin_authitemchild	浏览 结构 搜索 插入 清空 删除	25	InnoDB
<input type="checkbox"/> admin_organization	浏览 结构 搜索 插入 清空 删除	23	InnoDB
<input type="checkbox"/> admin_rights	浏览 结构 搜索 插入 清空 删除	22	InnoDB
<input type="checkbox"/> admin_user	浏览 结构 搜索 插入 清空 删除	12	InnoDB
<input type="checkbox"/> attachment	浏览 结构 搜索 插入 清空 删除	140	InnoDB
<input type="checkbox"/> category	浏览 结构 搜索 插入 清空 删除	0	MyISAM
<input type="checkbox"/> dic_category	浏览 结构 搜索 插入 清空 删除	41	InnoDB
<input type="checkbox"/> dic_item	浏览 结构 搜索 插入 清空 删除	4,877	InnoDB
<input type="checkbox"/> dic_test	浏览 结构 搜索 插入 清空 删除	19	MyISAM
<input type="checkbox"/> mental_consult_message	浏览 结构 搜索 插入 清空 删除	2	InnoDB
<input type="checkbox"/> mental_consult_reply	浏览 结构 搜索 插入 清空 删除	4	InnoDB
<input type="checkbox"/> mental_consult_type	浏览 结构 搜索 插入 清空 删除	2	InnoDB
<input type="checkbox"/> mental_factor_question_mapping	浏览 结构 搜索 插入 清空 删除	20	MyISAM
<input type="checkbox"/> mental_library_category	浏览 结构 搜索 插入 清空 删除	6	InnoDB
<input type="checkbox"/> mental_library_content	浏览 结构 搜索 插入 清空 删除	17	InnoDB
<input type="checkbox"/> mental_library_tag	浏览 结构 搜索 插入 清空 删除	56	InnoDB
<input type="checkbox"/> mental_rating_record	浏览 结构 搜索 插入 清空 删除	4	MyISAM
<input type="checkbox"/> mental_rating_scale	浏览 结构 搜索 插入 清空 删除	4	MyISAM

(最近使用的表) ...

ssmis

根据表名快速搜索

- admin_authassignment
- admin_authitem
- admin_authitemchild
- admin_organization
- admin_rights
- admin_user
- attachment
- category
- dic_category
- dic_item
- dic_test
- mental_consult_message
- mental_consult_reply
- mental_consult_type
- mental_factor_question_mapping
- mental_library_category
- mental_library_content
- mental_library_tag
- mental_rating_record
- mental_rating_scale

phpMyAdmin



- phpMyAdmin 是一个用 PHP 编写的应用程序，可以通过 Web 控制 and 操作 MySQL
- 可以对数据库进行各种操作，例如建立、复制/删除数据，执行SQL，导入导出SQL
- 支持 PHP 和 MySQL 最新版本
- <http://www.phpmyadmin.net/>

在PHP中访问MySQL数据库

PHP 中的 MySQL 接口

- 专用接口函数方式:
 - MySQL 扩展库
 - MySQLi 扩展库 (支持MySQL 4.1以后的新特性)
- 也就是有两套函数可以访问 MySQL
 - 一套函数名以 `mysql_` 开头
 - 一套函数名以 `mysqli_` 开头
- 访问各种数据库(PgSQL, Oracle, SQLServer..) 都有不同的接口函数
 - `pgsql_` , `oci_` , `mssql_` ,

PHP 中的 MySQL 接口

- 统一数据库接口(PDO, PHP Data Objects):
 - PDO_MYSQL
 - 从PHP 5.1 开始正式成为 PHP 的核心模块
- 选择
 - 越新的编程接口功能越强大, 使用越方便
 - MySQL 扩展在 PHP 5.5.0 之后废弃
 - MySQLi 兼容性很好
 - PDO 需要 PHP 5.1 之后的版本

php.ini 中的设置

打开以下设置

- `extension=php_mysql.dll`
 - `extension=php_mysqli.dll`
 - `extension=php_pdo.dll`
 - `extension=php_pdo_mysql.dll`
-
- 通过 `phpinfo()` 查看扩展是否被正常加载
 - 扩展被加载后才能使用其中的函数

从 Web 查询数据库的基本步骤

1. 检查并过滤来自用户的表单数据
 - trim、htmlspecialchars
 - 正则表达式
 - PHP Filter Functions (PHP 5.2 以上版本)
2. 建立到数据库的连接
3. 查询数据库
4. 获取查询结果
5. 将结果显示给用户

1、检查并过滤来自用户的表单数据

- 用户输入的一切数据都是靠不住的！这是 Web 开发中必须注意的一个重要原则
- trim : 删除字符串前后空格
- htmlspecialchars: 将 <, >, ", & 等转换成实体引用
- nl2br : 将换行符替换成

字符转义

- 单引号，双引号，反斜杠，NULL 字符
- 字符串中如有这些数据，插入数据库时可能会引起一些问题，因为数据库会将这些字符解释成控制符
- 为此需要对 SQL 语句的数据中出现的上述字符进行转义，两种方法：
 - PHP自动进行 (ini中): `magic_quotes_gpc = On`
 - 程序员手工完成: `addslashes` (转义), `stripslashes` (反转义)

2、连接数据库 (使用 **mysqli**)

```
<?php
// 建立数据库连接
@ $db = new mysqli('localhost', 'username',
    'password', 'books');
...
?>
```

host name

user name

MySQL password

数据库

- 这里使用的是**面向对象**的方式

检测错误

// 检测连接是否出错

```
if ( mysqli_connect_errno() )
```

```
{
```

```
    echo 'Error: Could not connect to database.  
        Please try again later.';
```

```
    exit;
```

```
}
```

3、查询数据库

// 执行数据库查询

```
$query = "SELECT * FROM books WHERE  
        $searchtype LIKE '%$searchterm%'";
```

```
$result = $db->query($query);
```

4、得到并显示查询结果

// 得到查询结果的行数（记录数）

```
$num_results = $result->num_rows;
```

```
for ($i=0; $i < $num_results; $i++) {
```

```
    // 处理并显示结果
```

```
    $row = $result->fetch_assoc();
```

```
    echo htmlspecialchars(stripslashes($row['title']));
```

```
    echo stripslashes($row['author']);
```

```
    echo stripslashes($row['isbn']);
```

```
    echo stripslashes($row['price']);
```

```
}
```

插入，删除，修改记录

```
$query = "INSERT INTO books VALUES  
        ('$isbn', '$author', '$title', '$price')";  
$result = $db->query($query);
```

```
// 获得受影响的行数（与SELECT的方法不同）  
echo $db->affected_rows.' book(s)  
        inserted into database.';
```

面向过程的方式 (mysqli 扩展版)

1. 连接到MySQL

`mysqli_connect(hostName,userName,dbasePassword)`

2. 选择数据库

`mysqli_select_db(connection, dbaseName)`

3. 运行查询并得到结果

`mysqli_query(sqlQuery, connection)`

4. 获取查询结果中的记录数

`mysqli_num_rows(mysqli_result result)`

5. 获得查询结果中的一行

`$row = mysqli_fetch_assoc($result);`

6. 获得一系列的值，格式化后输出

`$row['isbn'], $row['title']`

PDO 方式访问数据库

```
<?php
$dbh = new PDO("mysql:host=localhost;dbname=test", 'root', 'pass'); // 建立连接

foreach ($dbh->query('SELECT * from FOO') as $row) { // 检索查询结果
    print_r($row); // 显示结果
}

$dbh = null; // 关闭连接

?>
```

小技巧

- 为了增强安全性，并使得脚本易于移植修改，我们通常将数据库的连接信息放在单独的文件中，并用 `require()` 导入

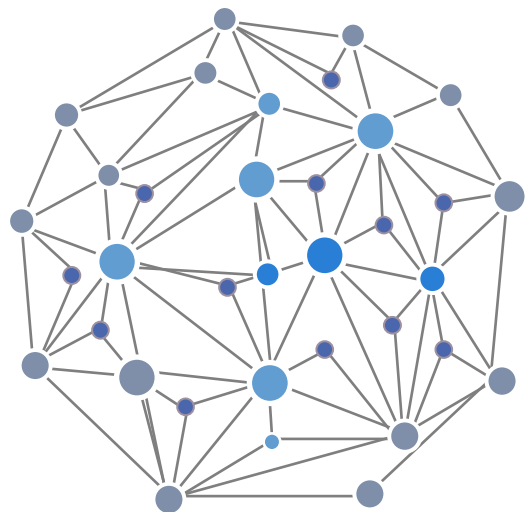
```
<?php
    $db_host = 'localhost';
    $db_database='database';
    $db_username='user';
    $db_password='password';
?>
```

通过 require 导入数据库连接信息

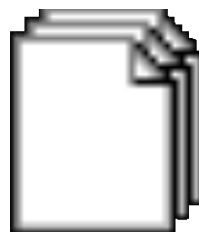
```
<?php
require ('config.php');
$db = new mysqli ($db_host,
                  $db_username, $db_password,
                  $db_database);

.....

?>
```



实例：简单博客系统



simpleblog.rar

Simple Blog System—发表文章 - Mozilla Firefox

文件 (E) 编辑 (E) 查看 (V) 历史 (S) 书签 (B) 工具 (I) 帮助 (H)

< > C X http://localhost/course/2009/simpleblog/list_post.php ☆ G cr ABP

Simple Blog System—发表文章

Simple Blog System—文章列表

共有文章：4 篇

1. 使至塞上

发表时间：2009-03-22 15:28:10

单车欲问边，属国过居延。
征蓬出汉塞，归雁入胡天。
大漠孤烟直，长河落日圆。
萧关逢候骑，都护在燕然。

2. 长相思

发表时间：2009-03-22 15:29:20

山一程
水一程
身向榆关那畔行
夜深千帐灯

风一更
雪一更

完成

Simple Blog System—发表文章 - Mozilla Firefox

文件 (E) 编辑 (E) 查看 (V) 历史 (S) 书签 (B) 工具 (I) 帮助 (H)

< > C X http://localhost/course/2009/simpleblog/n... ☆ G cr

Simple Blog System—发表文章

发表博客文章

标题：

正文：

提交

实例：简单博客系统

- 数据库设计
- 功能分析
- 客户端与服务器端设计

数据库设计

- 数据库： blog
- 表： post
- 字段
 - id: 文章编号, 主键, 从 1 开始自动编号
 - post_title: 文章标题
 - post_content: 文章内容
 - post_date: 文章发表时间

创建数据库

- 编写相应的 SQL 语句创建数据库

```
CREATE DATABASE `blog`  
    DEFAULT CHARACTER SET utf8  
    COLLATE utf8_general_ci;
```

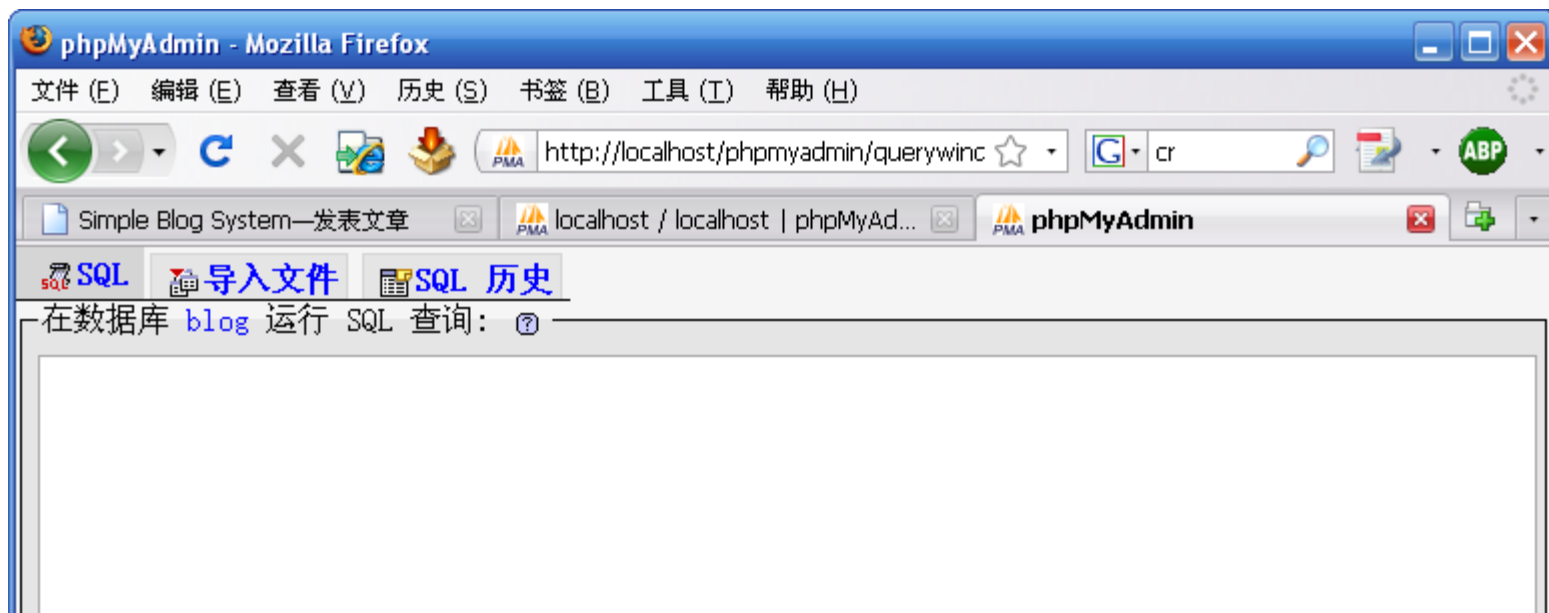
```
USE `blog`;
```

创建表

```
CREATE TABLE `post` (  
  `id` int NOT NULL auto_increment,  
  `post_title` text character set utf8,  
  `post_content` text character set utf8,  
  `post_date` datetime,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB  
  DEFAULT CHARSET=utf8  
  COLLATE=utf8_unicode_ci;
```

运行 SQL

- 通常将编写好的 SQL 保存到一个 .sql 文件
- 在命令行导入运行
 - `mysql -h localhost -u root -p < filename.sql`
- 或者通过 phpMyAdmin 等界面导入 SQL



功能分析

- 一个基本的博客系统，提供的客户端界面主要包括：
 - 添加新文章 Create
 - 显示文章列表 Read
 - 修改某篇文章 Update
 - 删除某篇文章 Delete
- 合称： CRUD

Create **R**ead **U**pdate **D**eleate

添加新文章

- 编写：new_post.html 与 new_post.php
- 一个是前端的界面，一个是后端的处理



new_post.html

```
<form action="new_post.php" method="post">
  <p>标题:</p>
  <p><input type="text" name="title"
    maxlength="60" size="30" /> </p>
  <p>正文:</p>
  <p><textarea name="content" rows="10"
    cols="80"> </textarea> </p>
  <p><input type="submit" value="提交" /> </p>
</form>
```

new_post.php 1/4

```
<?php
```

```
require('config.php'); // 数据库配置文件  
date_default_timezone_set('Asia/Shanghai');
```

```
// 获取表单提交的数据
```

```
$title=$_POST['title'];
```

```
$content=$_POST['content'];
```

new_post.php 2/4

```
if (!$title || !$content)
{
    echo '你未输入文章的标题或正文.<br />'
        . '请退回再次重试.';
    exit;
}
// 如果没有打开 PHP 的 magic quotes 功能, 则对提交的数据中的特殊符号进行转义处理
if (!get_magic_quotes_gpc()) {
    $title = addslashes($title);
    $body = addslashes($content);
}
```

new_post.php 3/4

```
@ $db = new mysqli($db_host, $db_username,  
    $db_password, $db_database);  
  
if (mysqli_connect_errno()) {  
    echo '错误: 无法连接到数据库. 请稍后再次重试.';  
    exit;  
}  
  
// 设置 utf8 字符集  
$db->query("SET NAMES utf8");
```

new_post.php 4/4

```
$query = "insert into post values('', '' . $title . '', '' .  
    $content . '', '' . date('Y-m-d H:i:s') . '')";  
echo '<pre>' . $query . '</pre>';  
  
$result = $db->query($query);  
if ($result) // 成功返回 TRUE, 失败返回 FALSE  
    echo $db->affected_rows.' 条记录被插入数据库中';  
else  
    echo '插入记录错误, 请检查程序代码';  
  
$db->close();
```

显示文章列表

- 编写一个 list_post.php



list_post.php 1/4

```
<?php
require('config.php');

@ $db = new mysqli($db_host, $db_username,
    $db_password, $db_database);

if (mysqli_connect_errno()) {
    echo '错误: 无法连接到数据库. 请稍后再次重试.';
    exit;
}
// 设置字符集
$db->query("SET NAMES utf8");
```

list_post.php 2/4

```
$query = "select * from post";  
$result = $db->query($query);
```

```
$num_results = $result->num_rows;
```

```
echo '<p>共有文章: '.$num_results.' 篇  
</p>';
```

list_post.php 3/4

```
for ($i=0; $i <$num_results; $i++) {  
    $row = $result->fetch_assoc();  
    echo '<h2>' . ($i+1) . '. ' .  
        htmlspecialchars(stripslashes(  
        $row['post_title'])) . "</h2>\n";  
    echo '<p>发表时间: ' .  
        htmlspecialchars(stripslashes(  
        $row['post_date'])) . "</p>\n";  
}
```

list_post.php 4/4

```
echo
```

```
'<p>'.nl2br(htmlspecialchars(stripslashes($row['post_content']))).'</p>';
```

```
}
```

```
// 释放查询结果集
```

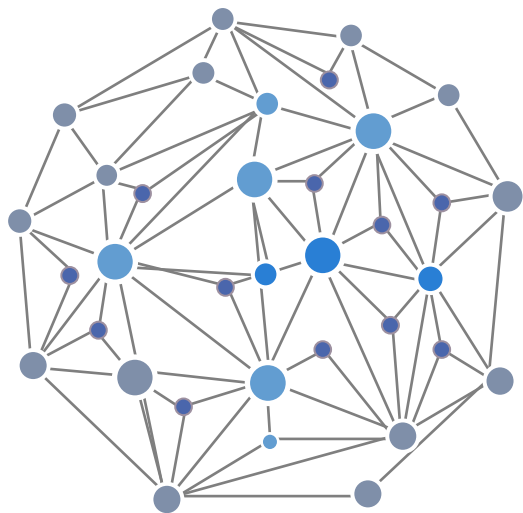
```
$result->free();
```

```
// 关闭数据库
```

```
$db->close();
```

用到的一些字符串处理函数

- **nl2br**
 - 用
替换字符串中的换行("0D0A")
- htmlspecialchars
 - 将<, >, ", &等转换成实体引用
- **addslashes**
 - 将字符串中的单引号, 双引号, 反斜杠等前面加上反斜杠进行转义
- **stripslashes**
 - 将转义的反斜杠删除掉



MySQL 存储引擎

存储引擎是什么？

- 存储引擎是 MySQL 将数据存储在文件(或是内存)中的格式
- 存储引擎也称为表格式
- 不同的存储引擎提供了不同的功能

MySQL 最常用的两种存储引擎

- MyISAM
 - 速度最快
 - 不支持外键，事务（为了追求速度）
- InnoDB
 - 支持外键
 - 支持事务

如何确定有哪些存储引擎可用

```
mysql> show engines;
```

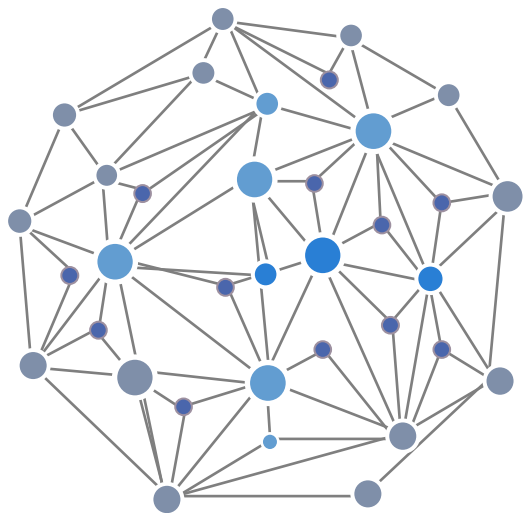
Engine	Support	Comment
MyISAM	DEFAULT	Default engine as of MySQL 3.23 with
HEAP	YES	Alias for MEMORY
MEMORY	YES	Hash based, stored in memory, useful
MERGE	YES	Collection of identical MyISAM tables
MRG_MYISAM	YES	Alias for MERGE
ISAM	NO	Obsolete storage engine, now replaced
MRG_ISAM	NO	Obsolete storage engine, now replaced
InnoDB	YES	Supports transactions, row-level locking
INNODB	YES	Alias for INNODB
BDB	NO	Supports transactions and page-level locking

启用存储引擎

- MyISAM 是默认启用的
- InnoDB 可能会被关闭 (可以节省内存和硬盘占用, 加速 MySQL 运行速度)
 - XAMPP 套装有可能就关闭了 InnoDB
- 启用 InnoDB 方法:
 - 修改 MySQL 配置文件: my.ini 或 my.cnf
 - 将配置文件中的 skip-innodb 前面加上#, 重启 mysqld 就可以启用 InnoDB

SQL 中指定和修改存储引擎

- `CREATE TABLE table ENGINE = InnoDB;`
- `ALTER TABLE table ENGINE = MyISAM;`



MySQL 字符集问题

字符集 和 校对规则

- 字符集是指一套符号及其编码
- 校对规则是在字符集内用于比较字符的一套规则
- 校对规则用途：字符串比较

字符集 和 校对规则

- 假设我们有一个字母表使用了四个字母：
'A'、'B'、'a'、'b'。我们为每个字母
赋予一个数值：'A'=0, 'B'=1, 'a'=2,
'b'=3。
- 字母 'A' 是一个符号 (symbol) , 数字 0
是 'A' 的编码 (encoding)
- 这四个字母和它们的编码组合在一起就构成了一个字符集

字符集 和 校对规则

- 假设我们希望比较两个字符串的值： 'A' 和 'B'
- 比较的最简单的方法是为字符编码： 'A' 为 0, 'B' 为 1
- 因为 0 小于 1, 我们可以说 'A' 小于 'B'。这就是我们为这个字符集定义的一套用于比较字符的规则。

校对规则

- 现实情况要复杂得多
- 例如：
- 在德语中存在如下校对规则：
Ü=Ue Ä=Ae Ö=Oe ß=ss
- 因此字符串在比较时不能简单的逐字符比较，需考虑到校对规则（字符的比较结果对排序等算法的结果将产生影响）

MySQL 中的字符集和校对规则

- MySQL 5 能够做这些事情
 - 使用多种字符集来存储字符串
 - 使用多种校对规则来比较字符串
 - 在同一台服务器、同一个数据库或甚至在同一个表中使用不同字符集或校对规则来混合字符串
 - 允许定义任何级别的字符集和校对规则

MySQL 中的字符集和校对规则

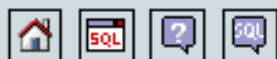
- 当一个 PHP 程序与 MySQL 建立连接后，这个程序发送给 MySQL 的数据采用的是什
么字符集？
- MySQL 无从得知 (它最多只能猜测)，所以
MySQL 要求客户端必须先告诉服务器字符
集是什么
- 只有正确告诉 MySQL 我们的数据是采用什
么字符集和校对规则，MySQL 才能保证数
据存储和读取的正确

中文之道

- 许多的 MySQL 服务器默认采用 latin1 编码, 这是读取中文时乱码的原因
- 最简单解决方法:
- 建立连接后立即执行
 - `$db->query("SET NAMES utf8");`
- 并在客户端(网页)和服务端全部采用UTF-8编码, 也就是 Unicode 字符集



phpMyAdmin



数据库

(数据库) ...

请选择数据库

localhost

服务器版本: 5.0.27-community-nt

Protocol version: 10

服务器: localhost via TCP/IP

用户: root@localhost

MySQL 字符集: UTF-8 Unicode (utf8)

MySQL 连接校对:

utf8_unicode_ci

创建一个新的数据库

utf8_unicode_ci

创建

显示 MySQL 的运行信息

显示 MySQL 的系统变量

进程

字符集和整理

Storage Engines

Reload privileges

权限

数据库

导出

phpMyAdmin - 2.9.2

MySQL client version: 5.0.22

Used PHP extensions: mysql

Language

中文 - Chinese simplified

主题 / 风格: Original

Font size: 80%

phpMyAdmin 文档

phpMyAdmin 官方网站

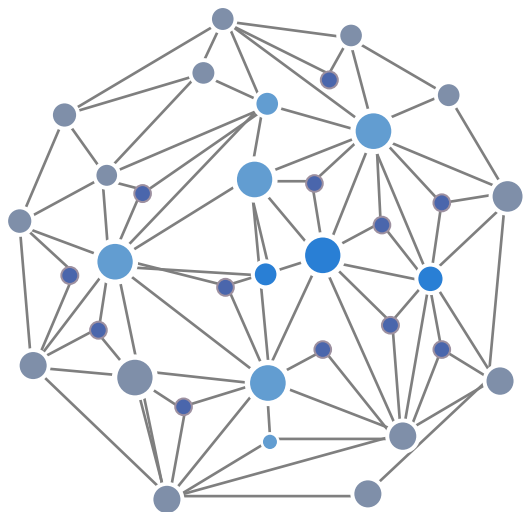
[ChangeLog] [CVS] [Lists]

Unicode 常用的两种校对规则

- utf8_unicode_ci 和 utf8_general_ci 是 Unicode 字符集常用的两种校对规则
- utf8_unicode_ci 支持字母组合：
例如：德语中 ß = ss
- utf8_general_ci 只能在字符之间逐个比较
因此：ß = s
- utf8_unicode_ci 较精确，utf8_general_ci 速度较快，根据需要选择

第八讲课后练习

- 编程作业
 - <http://chenyv.gitee.io/webprogramming/homeworks/simpleblog>
- 最好的学习方法，就是自己动手做一遍！



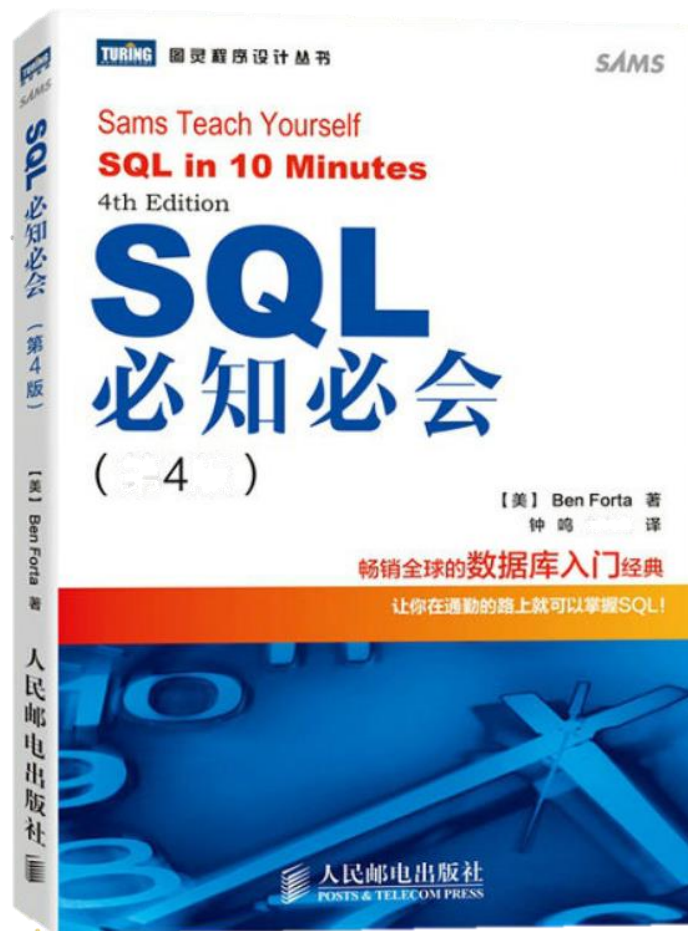
推荐书籍

/推荐书籍/WebProgramming/SQL

/推荐书籍/WebProgramming/PHP

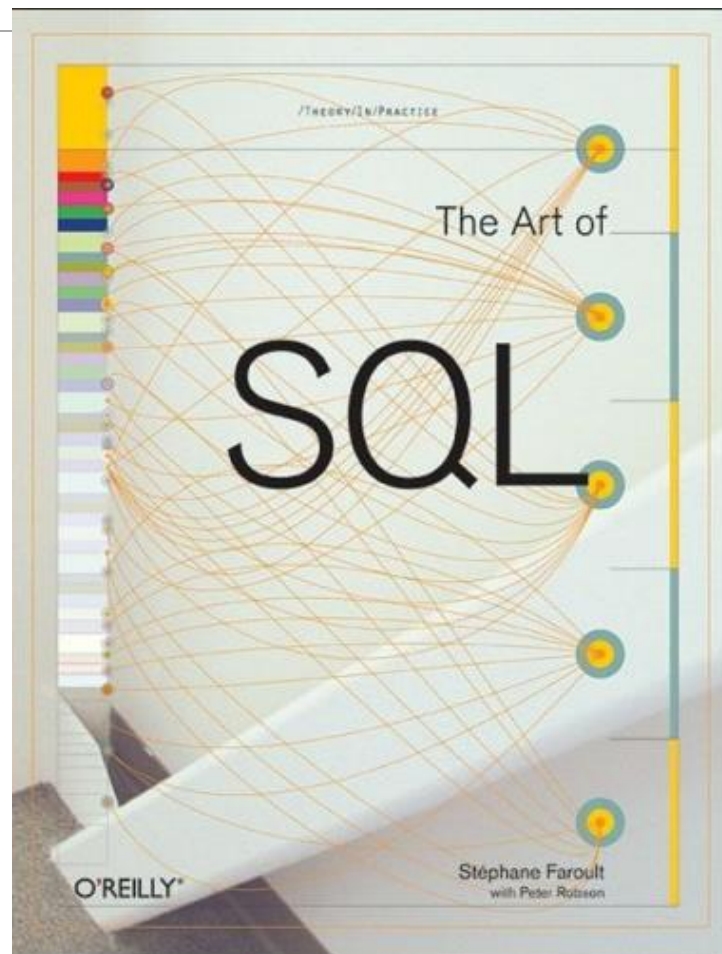
推荐书籍

- Sams Teach Yourself SQL in 10 Minutes
- 中文版：SQL 必知必会(第4版)
- SQL 入门书，手册



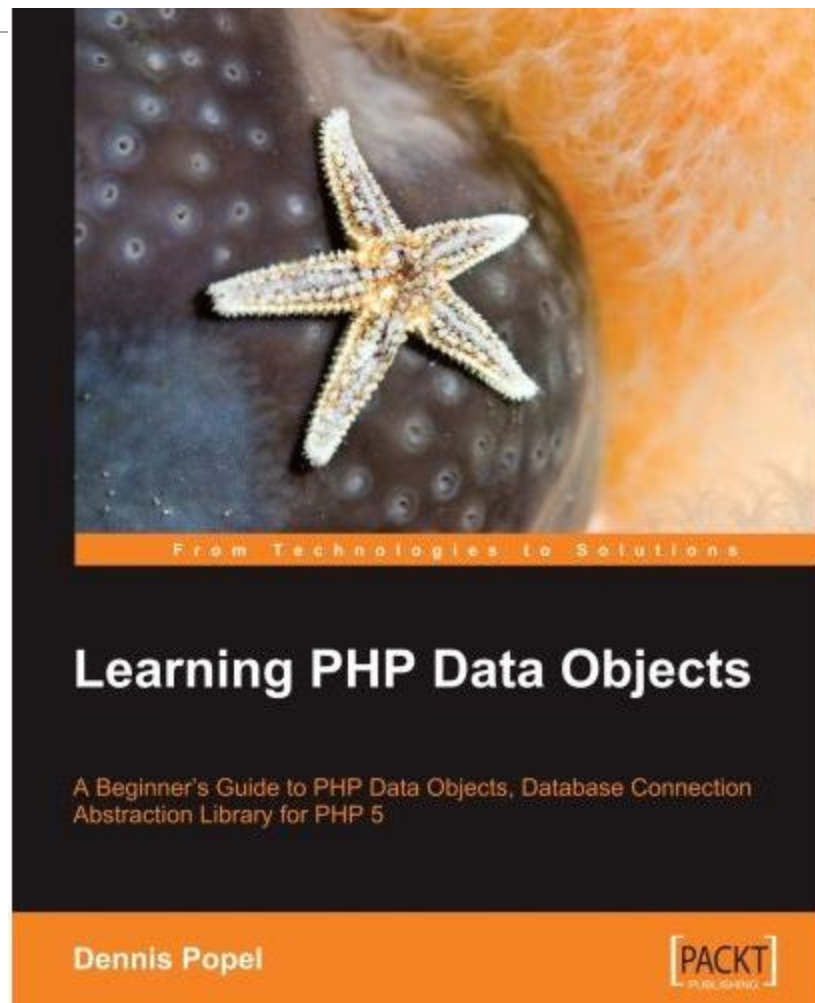
推荐书籍

- O'Reilly
- The Art of SQL
- 中文版：SQL 语言艺术
- 高级技巧



推荐书籍

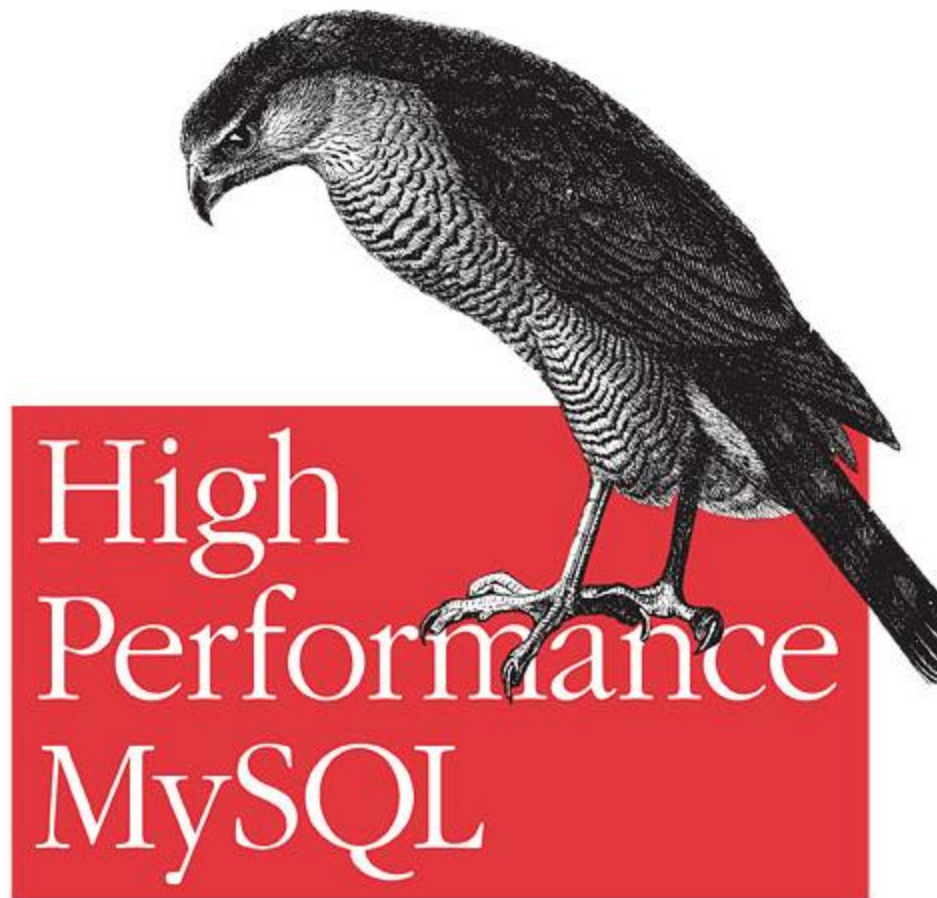
- Learning PHP Data Objects
- PDO 入门



推荐书籍

- O'Reilly
- High Performance MySQL: Optimization, Backups, Replication, and More
- MySQL 优化, 备份, 负载平衡等高性能技术

Advanced Tools & Techniques for MySQL Administrators



*Optimization, Backups,
Replication & Load Balancing*

O'REILLY®

Jeremy D. Zawodny & Derek J. Balling

THANKS

本章结束

陈昱

福州大学 计算机与大数据学院 软件工程系

