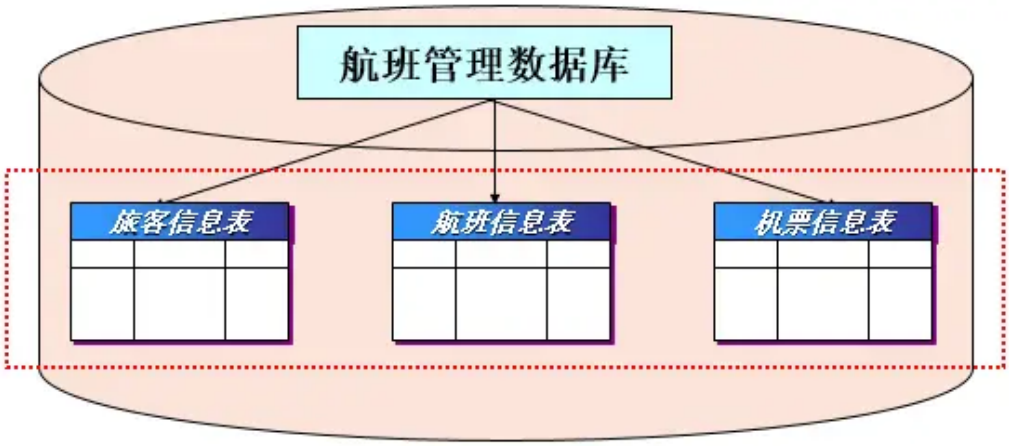
1.1 今日目标

1. 了解数据库的发展史
2. 了解什么是SQL语句
3. 了解启动和停止MySQL服务器
4. 能够连接MySQL数据库
5. 能够关闭数据库连接
6. 能够创建数据库
7. 能够查看所有数据库
8. 能够修改数据库的字符编码
9. 能够删除数据库
10. 能够选择数据库
11. 能够创建表
12. 能够查看当前数据库中的所有表
13. 能够查看表结构
14. 能够查看创建表的SQL语句
15. 能够能够复制表
16. 能够给表添加字段
17. 能够给表删除字段
18. 能够修改表字段
19. 能够修改表引擎
20. 能够修改表名
21. 能够向表中添加数据
22. 能够修改表中的数据
23. 能够删除表中的数据

1.2 数据库的作用

数据库是存放数据的仓库



数据库：数据库中存放的是表，一个数据库中可以存放多个表

表：表是用来存放数据的。

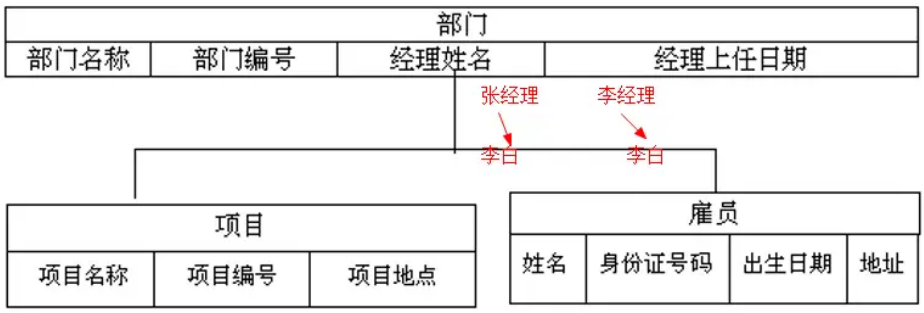
1.3 数据库的发展史

1.3.1 萌芽阶段：文件系统

最初始的数据库是用磁盘来存储数据的。文件就是最早的数据库。

1.3.2 第一代数据库：层次模型、网状模型

1.3.2.1 层次模型



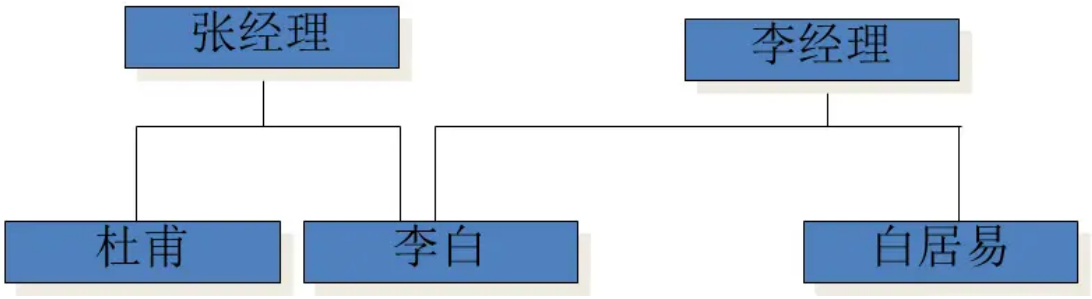
缺点：

1、 查找不同类的数据效率低了（导航的结构的缺点）

2、 数据不完整（不能区分到底是一个李白还是两个李白）

1.3.2.2 网状模型

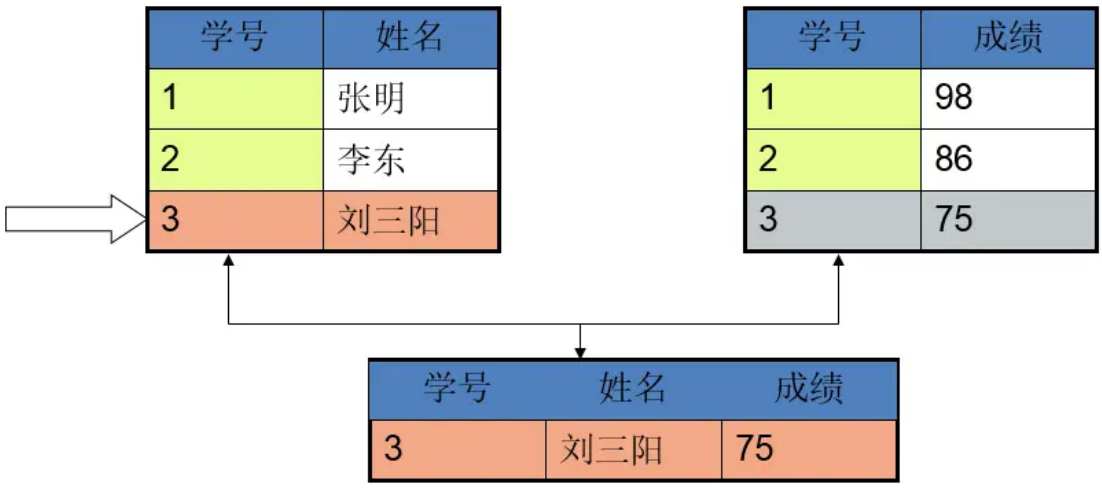
网状模型解决了层次数据的数据不完整的问题，但是没有解决层次模型的导航问题。



1.3.3 第二代数据库：关系型数据库

特点：

1. 每个表都是独立的
2. 表与表之间通过公共字段来建立关系



优点：解决了导航问题，并且数据完整性得到解决

缺点：多表查询效率低了

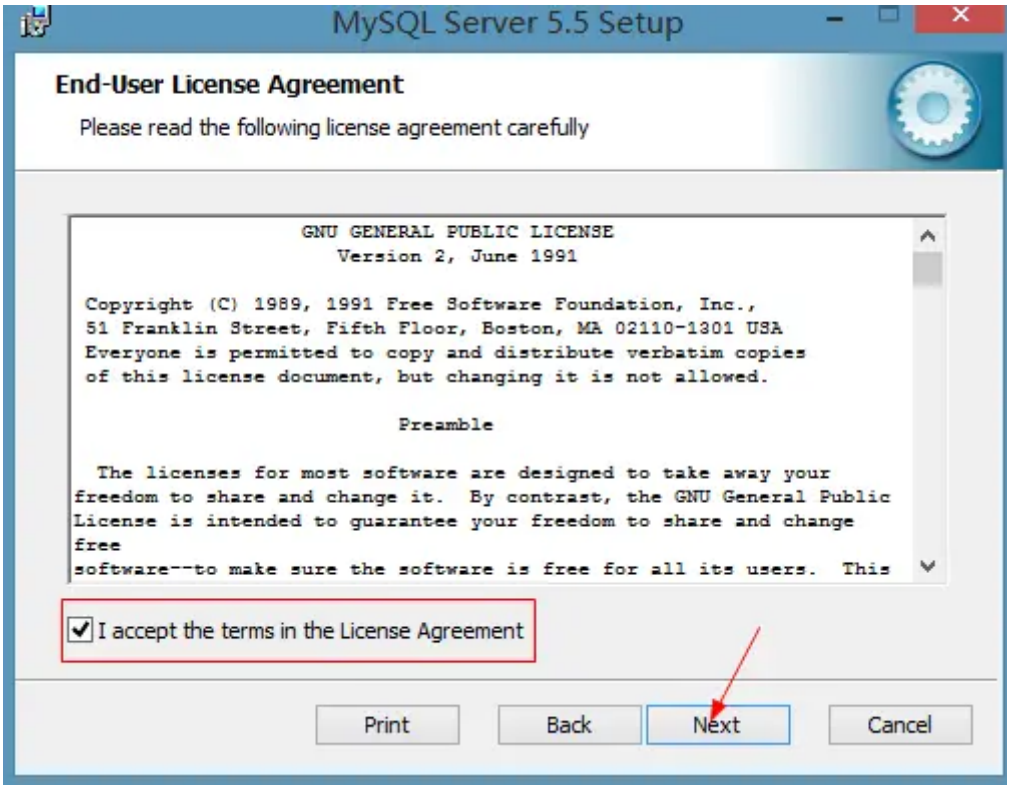
提示：我们现在用的主流的数据库都是关系模型的。

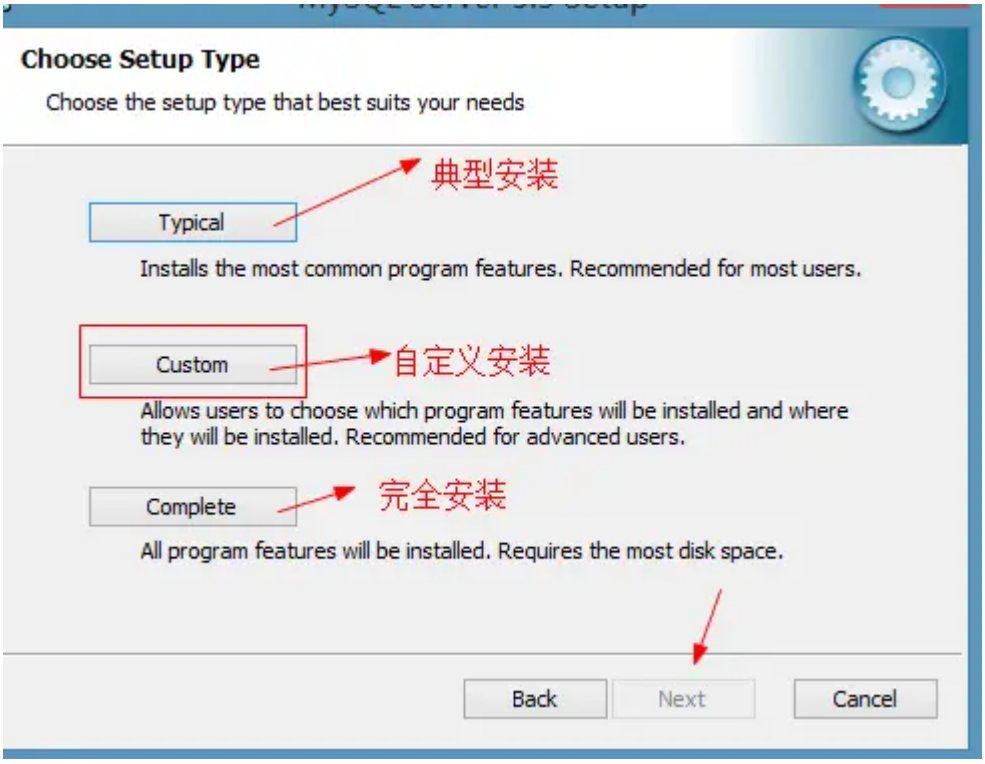
> 多学一招：NoSQL（非关系型数据库）解决关系型数据库多表查询效率的问题，常见的非关系型数据库有：Redis、mongodb。数据库中存储格式是键值对。

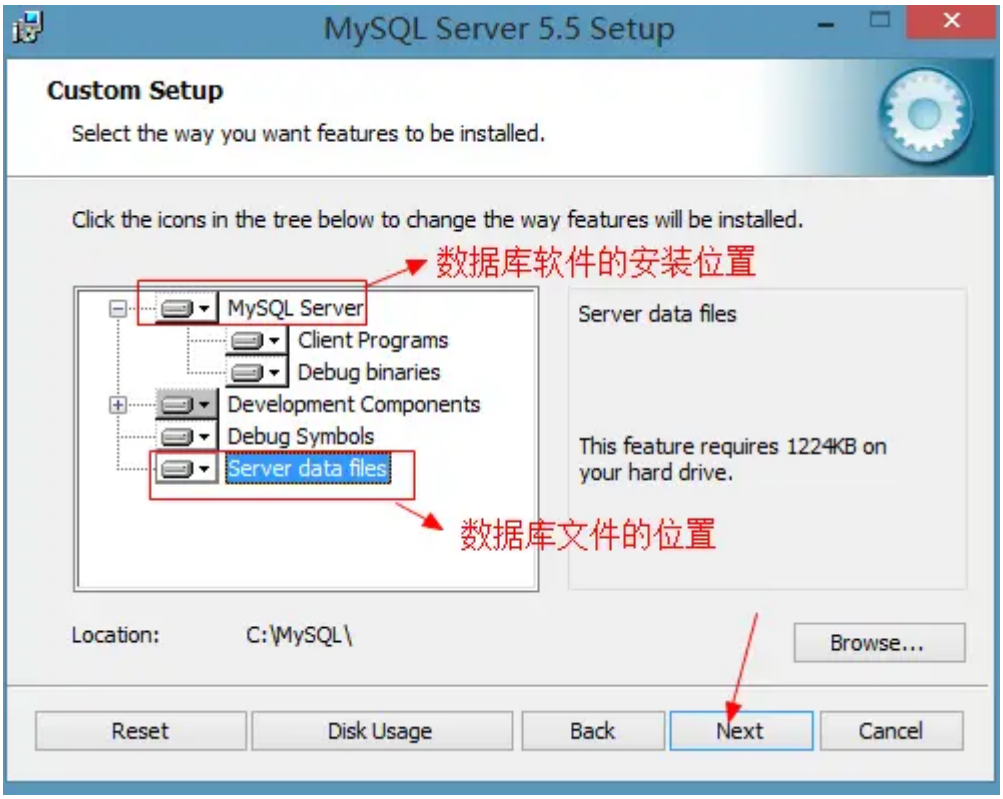
1.4 MySQL安装

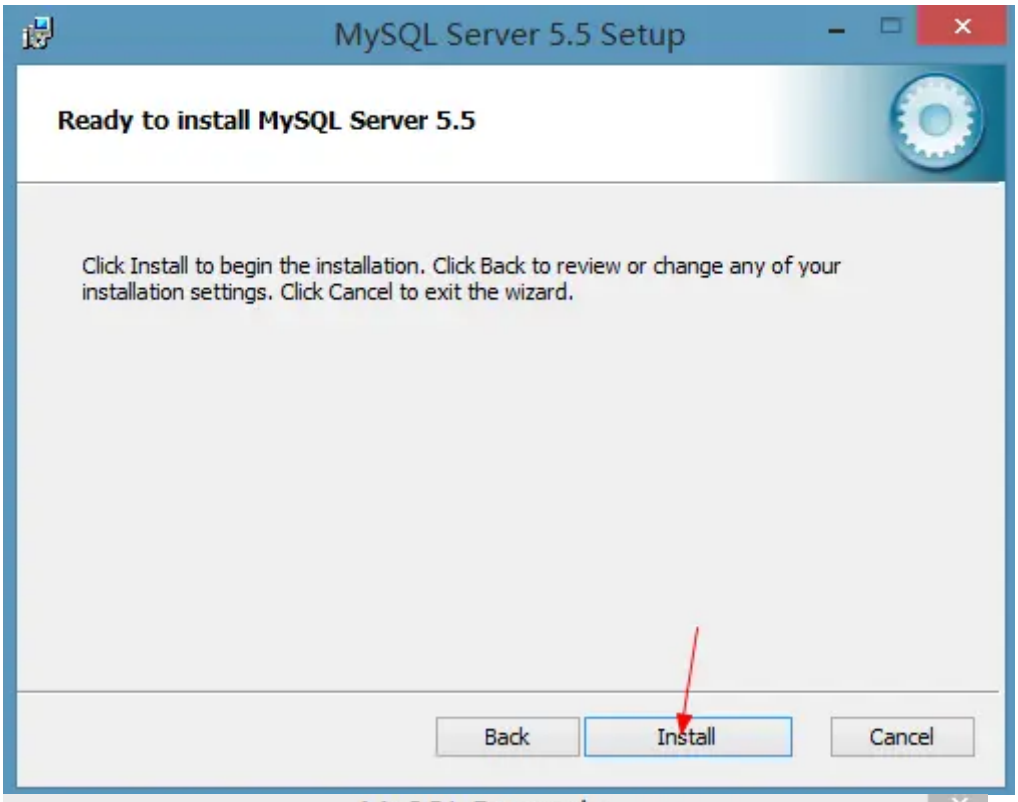
1.4.1 获取安装文件

1.4.2 双击安装

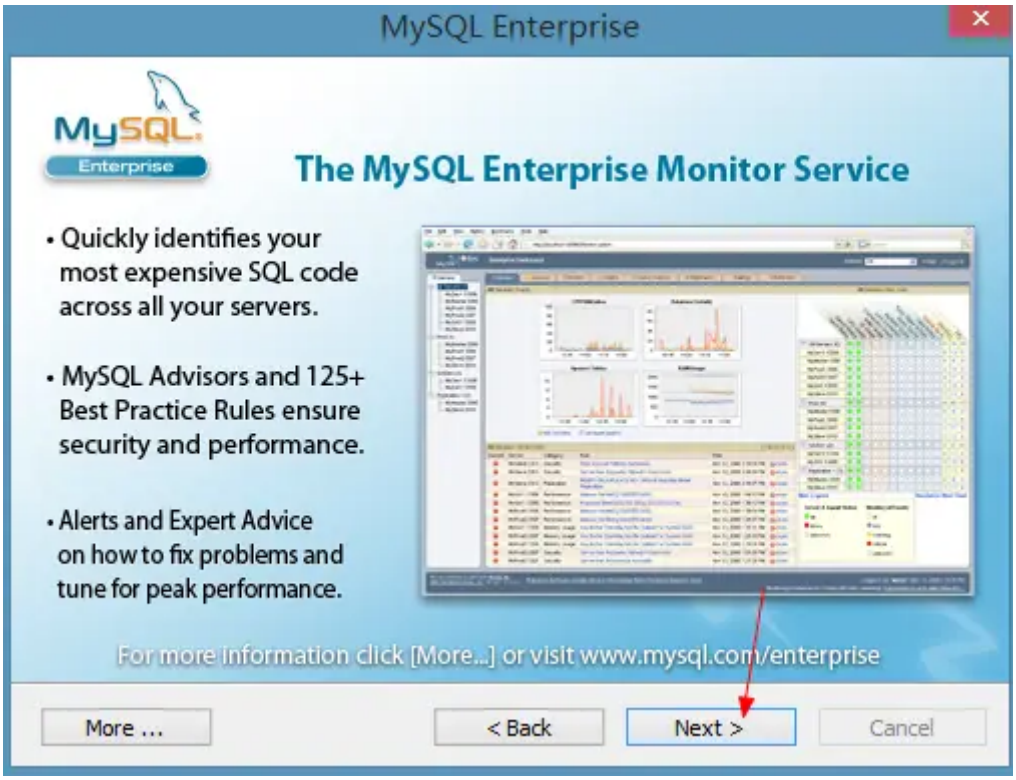

MySQL Server 5.5 Setup
End-User License Agreement
Please read the following license agreement carefully
GNU GENERAL PUBLIC LICENSE
Version 2, June 1991
Copyright (C) 1989, 1991 Free Software Foundation, Inc.,
51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA
Everyone is permitted to copy and distribute verbatim copies
of this license document, but changing it is not allowed.
Preamble
The licenses for most software are designed to take away your
freedom to share and change it. By contrast, the GNU General Public
License is intended to guarantee your freedom to share and change
free
software--to make sure the software is free for all its users. This
I accept the terms in the License Agreement
Print
Back
Next
Cancel

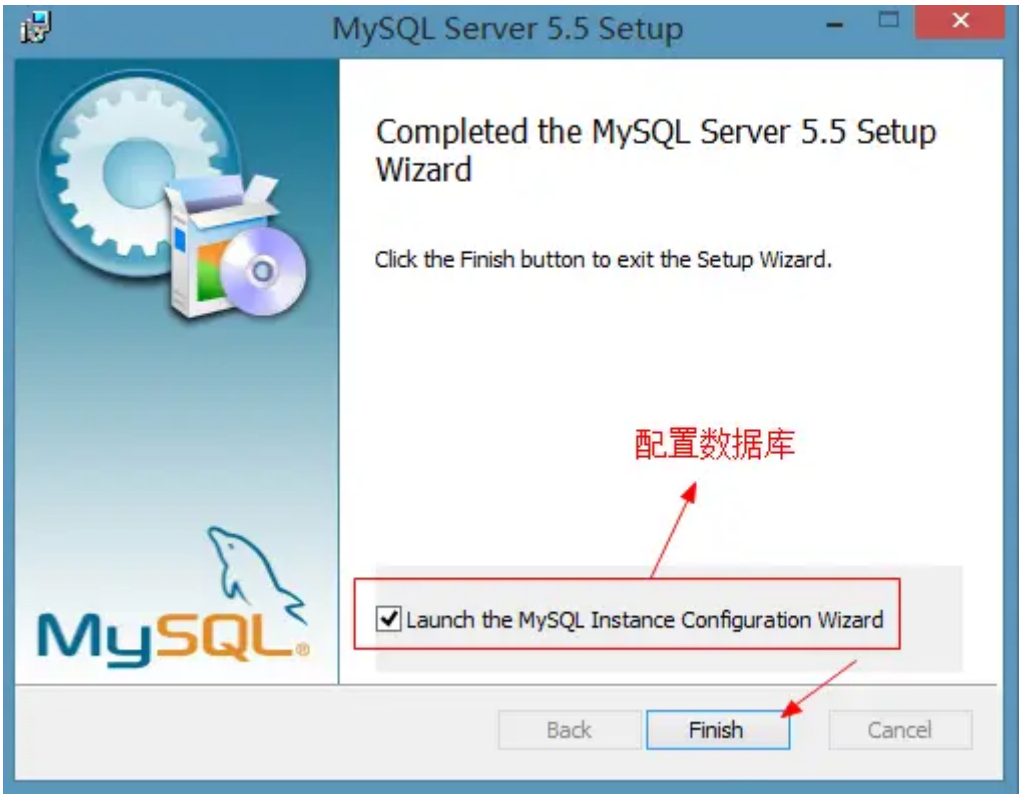

Choose Setup Type
Choose the setup type that best suits your needs
典型安装
Typical
Installs the most common program features. Recommended for most users.
Custom
自定义安装
Allows users to choose which program features will be installed and where
they will be installed. Recommended for advanced users.
Complete
完全安装
All program features will be installed. Requires the most disk space.
Back
Next
Cancel

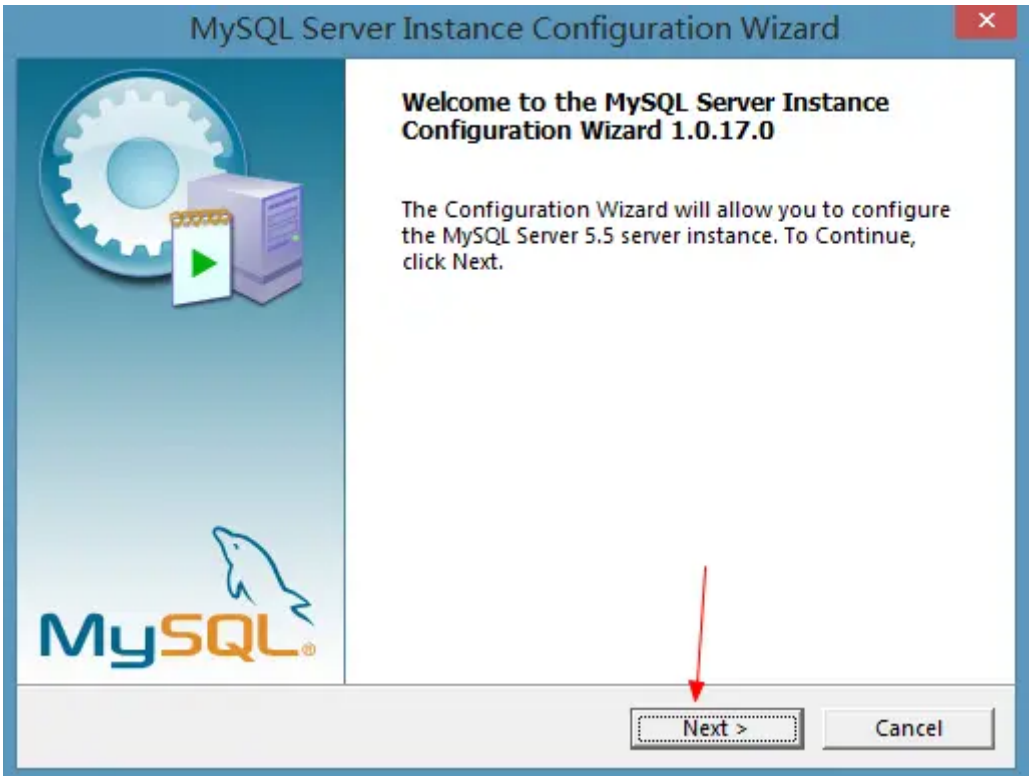

MySQL Server 5.5 Setup
Custom Setup
Select the way you want features to be installed.
Click the icons in the tree below to change the way features will be installed.
数据库软件的安装位置
MySQL Server
Client Programs
Debug binaries
Development Components
Debug Symbols
Server data files
Server data files
This feature requires 1224KB on your hard drive.
数据库文件的位置
Location:
C:\MySQL\
Browse...
Reset
Disk Usage
Back
Next
Cancel

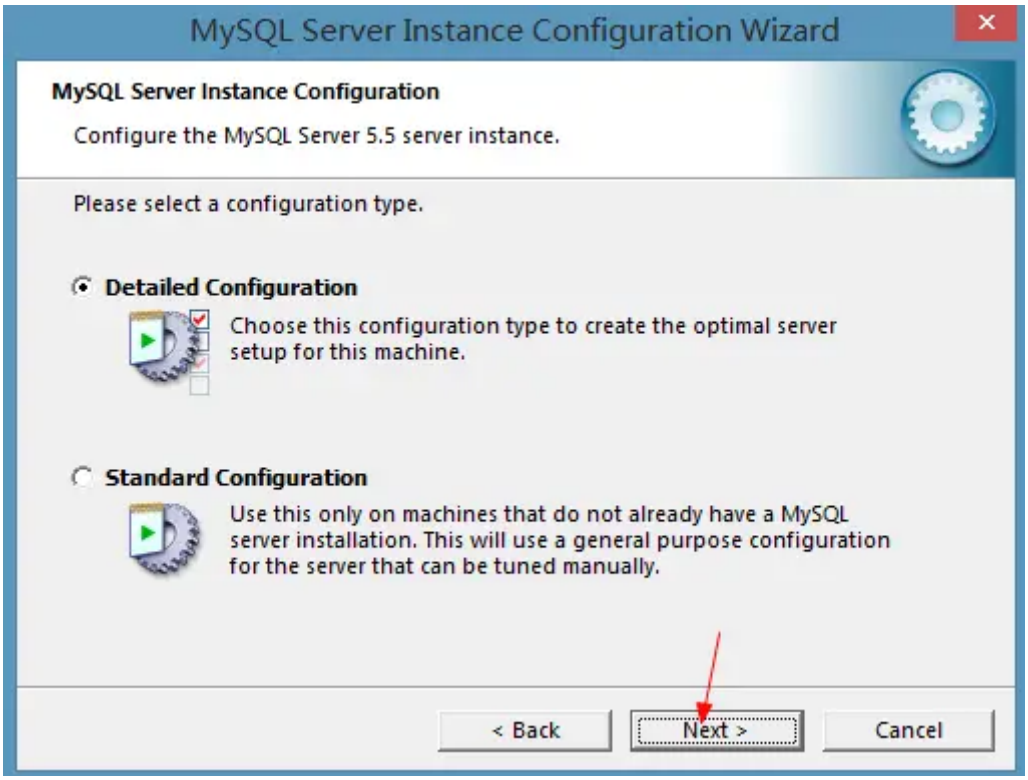

MySQL Server 5.5 Setup
Ready to install MySQL Server 5.5
Click Install to begin the installation. Click Back to review or change any of your installation settings. Click Cancel to exit the wizard.
Back
Install
Cancel

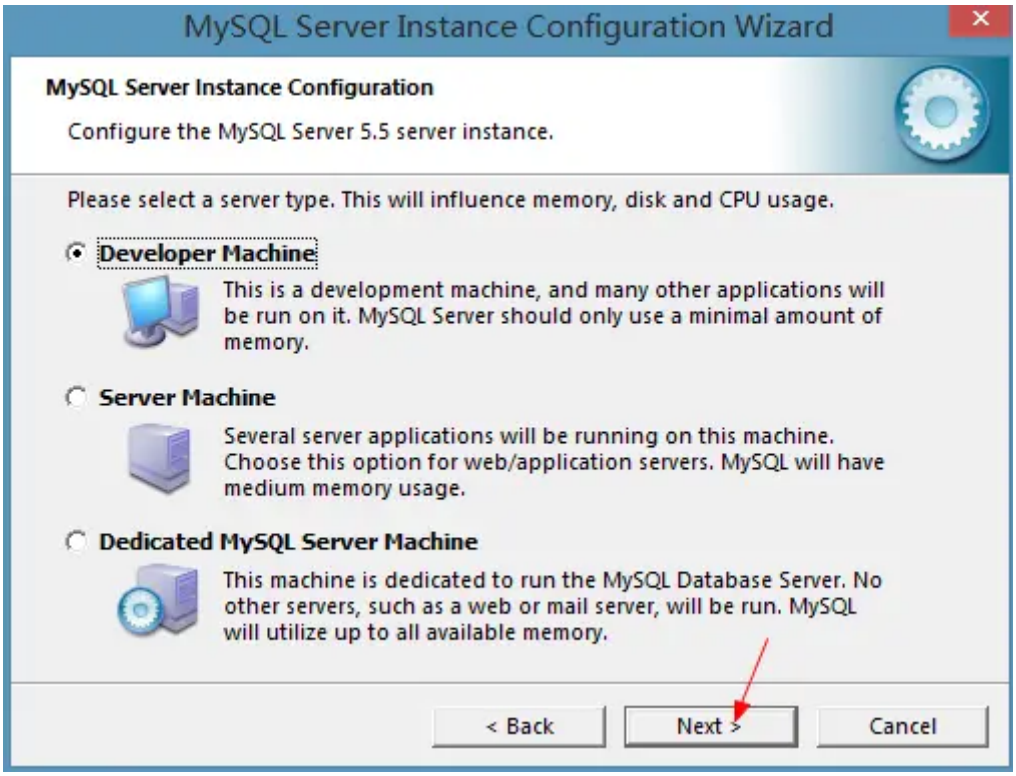

MySQL Enterprise
MySQL
Enterprise
A MySQL Enterprise subscription is the most comprehensive offering of MySQL database software, services, and support to ensure your business achieves the highest levels of reliability, security, and uptime.
An Enterprise Subscription includes:
1. The MySQL Enterprise Server - The most reliable, secure, and up-to-date version of the world's most popular open source database.
2. MySQL Enterprise Monitor Service - An automated virtual database assistant.
3. MySQL Production Support - Technical and consultative support when you need it, along with service packs, hot-fixes, and more.
For more information click [More...] or visit www.mysql.com/enterprise
More ...
< Back
Next >
Cancel

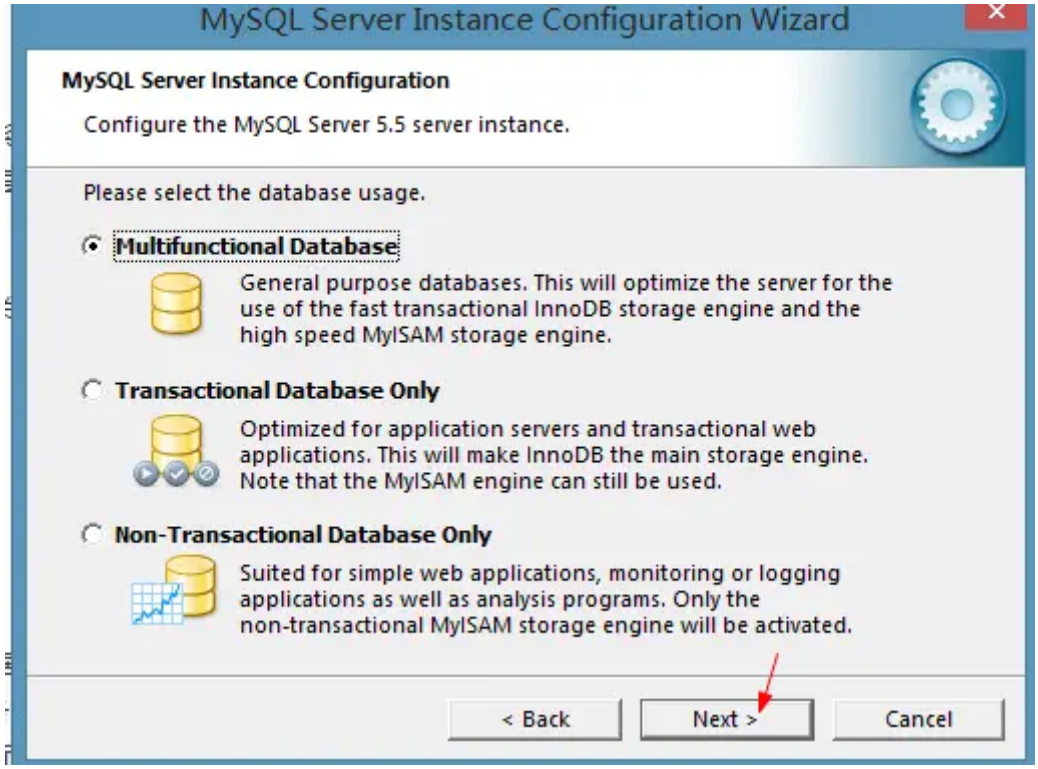

MySQL Enterprise
MySQL
Enterprise
The MySQL Enterprise Monitor Service
• Quickly identifies your most expensive SQL code across all your servers.
• MySQL Advisors and 125+ Best Practice Rules ensure security and performance.
• Alerts and Expert Advice on how to fix problems and tune for peak performance.
For more information click [More...] or visit www.mysql.com/enterprise
More ...
< Back
Next >
Cancel

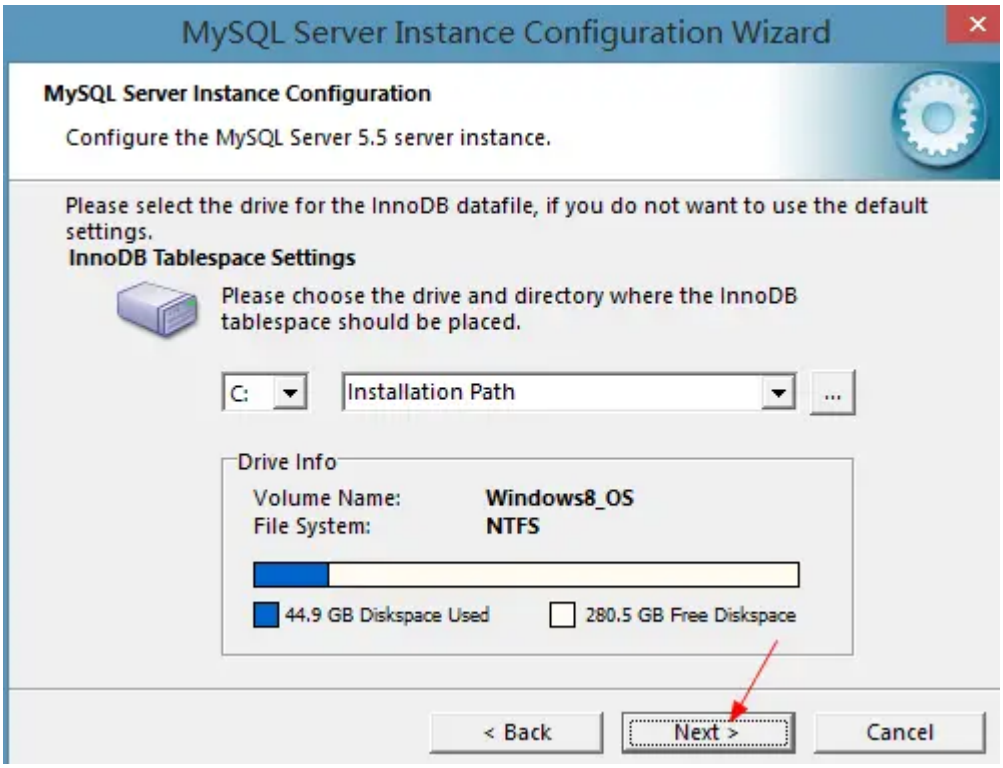

MySQL Server 5.5 Setup
Completed the MySQL Server 5.5 Setup Wizard
Click the Finish button to exit the Setup Wizard.
配置数据库
Launch the MySQL Instance Configuration Wizard
MySQL
Back
Finish
Cancel

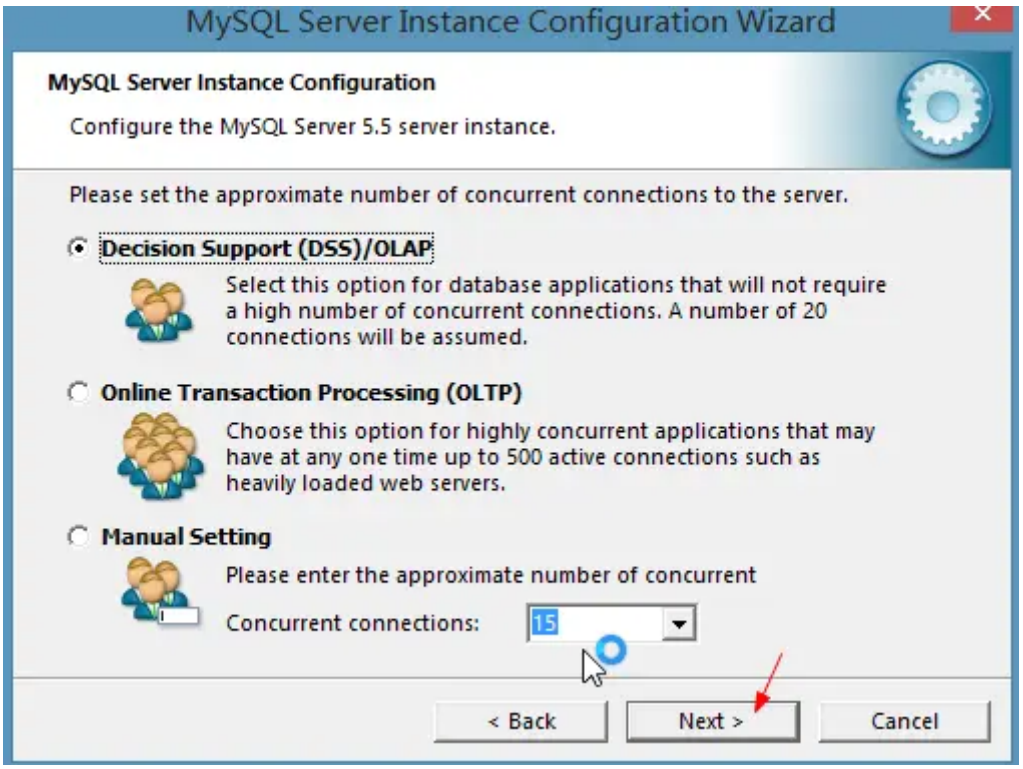

MySQL Server Instance Configuration Wizard
Welcome to the MySQL Server Instance Configuration Wizard 1.0.17.0
The Configuration Wizard will allow you to configure the MySQL Server 5.5 server instance. To Continue, click Next.
MySQL
Next >
Cancel

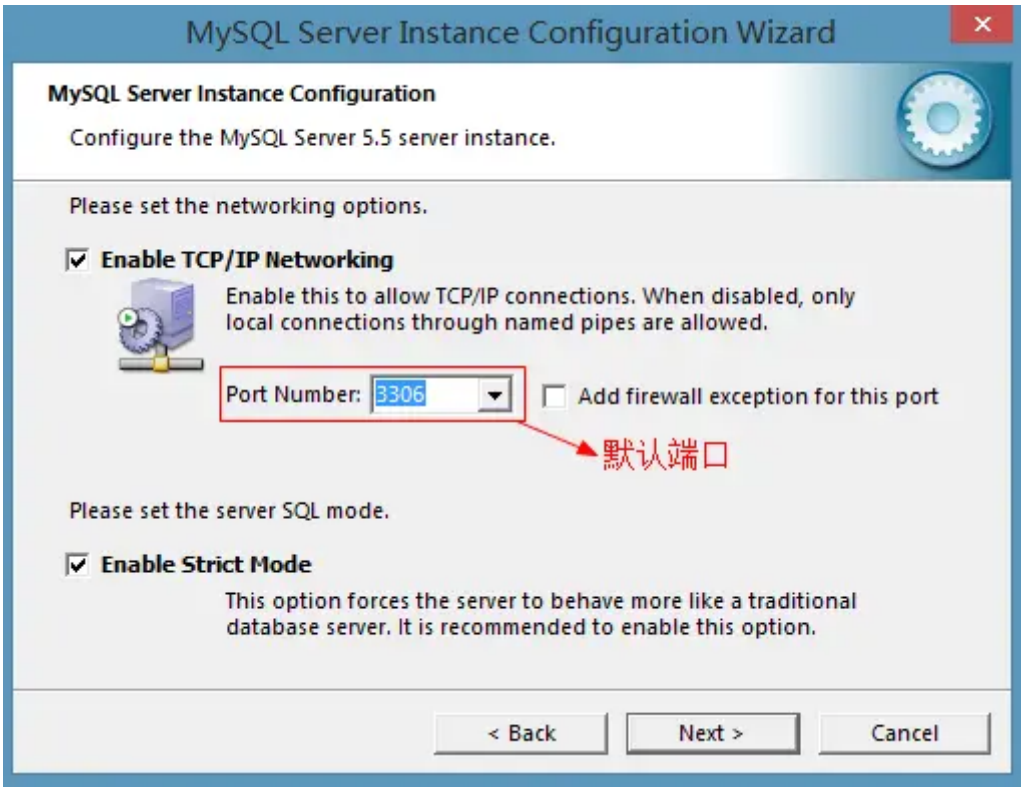

MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please select a configuration type.
Detailed Configuration
Choose this configuration type to create the optimal server setup for this machine.
Standard Configuration
Use this only on machines that do not already have a MySQL server installation. This will use a general purpose configuration for the server that can be tuned manually.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please select a server type. This will influence memory, disk and CPU usage.
Developer Machine
This is a development machine, and many other applications will be run on it. MySQL Server should only use a minimal amount of memory.
Server Machine
Several server applications will be running on this machine. Choose this option for web/application servers. MySQL will have medium memory usage.
Dedicated MySQL Server Machine
This machine is dedicated to run the MySQL Database Server. No other servers, such as a web or mail server, will be run. MySQL will utilize up to all available memory.
< Back
Next >
Cancel

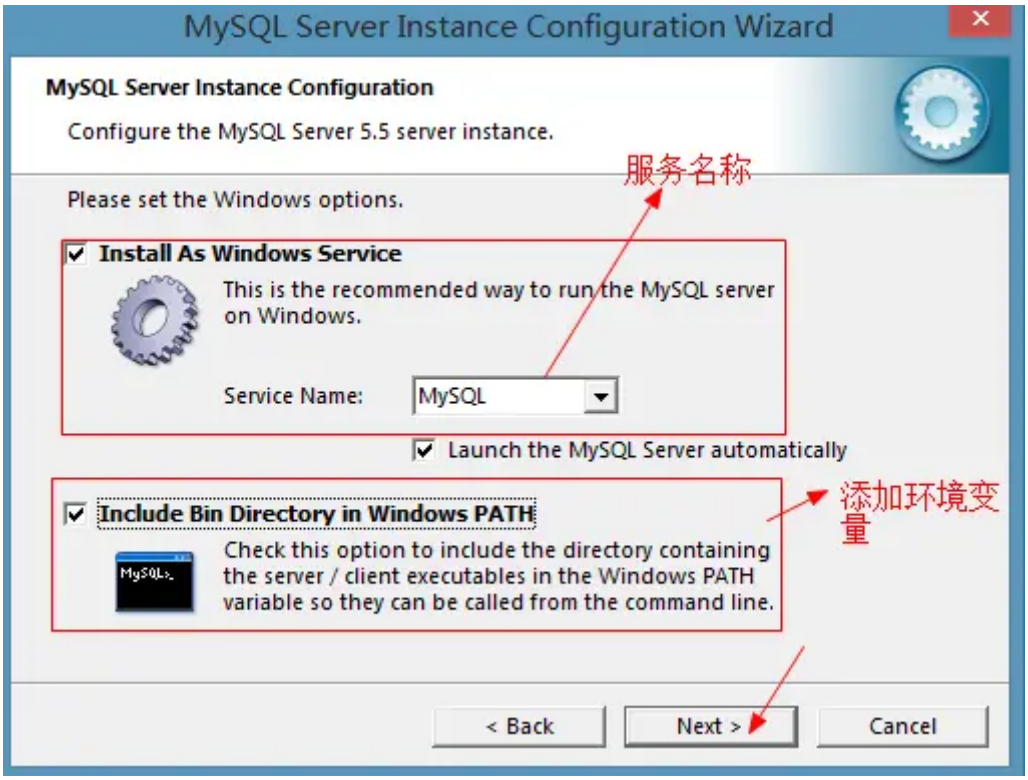

MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please select the database usage.
Multifunctional Database
General purpose databases. This will optimize the server for the use of the fast transactional InnoDB storage engine and the high speed MyISAM storage engine.
Transactional Database Only
Optimized for application servers and transactional web applications. This will make InnoDB the main storage engine. Note that the MyISAM engine can still be used.
Non-Transactional Database Only
Suited for simple web applications, monitoring or logging applications as well as analysis programs. Only the non-transactional MyISAM storage engine will be activated.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please select the drive for the InnoDB datafile, if you do not want to use the default settings.
InnoDB Tablespace Settings
Please choose the drive and directory where the InnoDB tablespace should be placed.
C:
Installation Path
...
Drive Info
Volume Name: Windows8_OS
File System: NTFS
44.9 GB Diskspace Used
280.5 GB Free Diskspace
< Back
Next >
Cancel

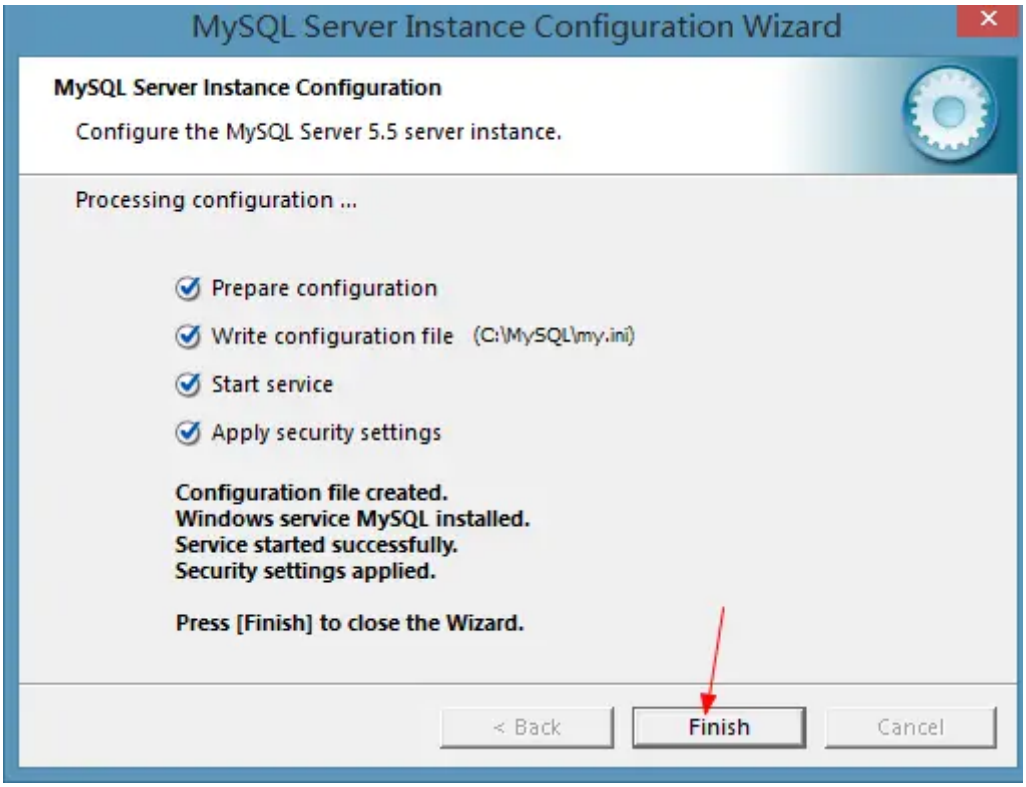

MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please set the approximate number of concurrent connections to the server.
Decision Support (DSS)/OLAP
Select this option for database applications that will not require a high number of concurrent connections. A number of 20 connections will be assumed.
Online Transaction Processing (OLTP)
Choose this option for highly concurrent applications that may have at any one time up to 500 active connections such as heavily loaded web servers.
Manual Setting
Please enter the approximate number of concurrent
Concurrent connections:
15
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please set the networking options.
Enable TCP/IP Networking
Enable this to allow TCP/IP connections. When disabled, only local connections through named pipes are allowed.
Port Number: 3306
Add firewall exception for this port
默认端口
Please set the server SQL mode.
Enable Strict Mode
This option forces the server to behave more like a traditional database server. It is recommended to enable this option.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
Please select the default character set.
Standard Character Set
Hello!
Makes Latin1 the default charset. This character set is suited for English and other West European languages.
Best Support For Multilingualism
日本語
Make UTF8 the default character set. This is the recommended character set for storing text in many different languages.
Manual Selected Default Character Set / Collation
编码方式
Please specify the character set to use.
Character Set: utf8
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
服务名称
Please set the Windows options.
Install As Windows Service
This is the recommended way to run the MySQL server on Windows.
Service Name:
MySQL
Launch the MySQL Server automatically
Include Bin Directory in Windows PATH
添加环境变量
Check this option to include the directory containing the server / client executables in the Windows PATH variable so they can be called from the command line.
< Back
Next >
Cancel


MySQL Server Instance Configuration Wizard
MySQL Server Instance Configuration
Configure the MySQL Server 5.5 server instance.
初始化密码
Please set the security options.
Modify Security Settings
New root password:
****
Enter the root password.
Confirm:
****
Retype the password.
Enable root access from remote machines
Create An Anonymous Account
This option will create an anonymous account on this server.
Please note that this can lead to an insecure system.
< Back
Next >
Cancel

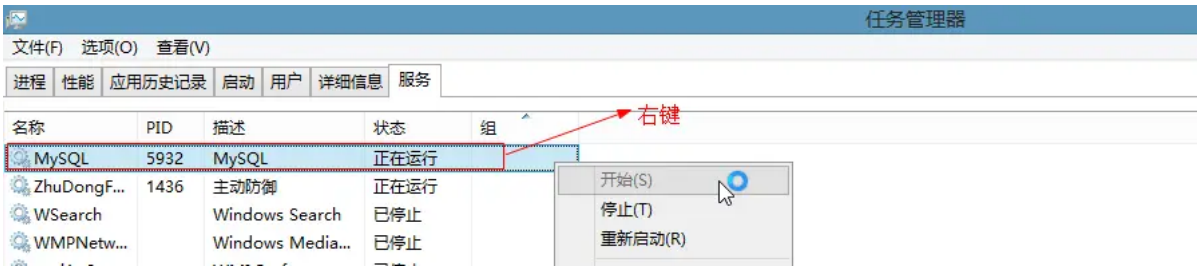


1.4.3 MySQL目录

1.5 启动/关闭MySQL服务

1.5.1 方法一：在服务面板中启动或关闭

控制面板项——管理工具——服务，选择相应服务，右键执行操作。



1.5.2 方法二：通过命令行启动\关闭

net start 服务名：　　启动MySQL服务

net stop 服务器：　　关闭MySQL服务

> 注意：必须通过管理员身份启动命令行

1.6 SQL介绍

1.6.1 SQL是什么

Structured Query Language（结构化查询语言），是用来操作关系型数据库的一门语言。这是一个关系型数据库的通用操作语言，也成为标准SQL，也叫SQL-92。

> 多学一招：数据库的生产厂商为了占有市场份额，都会在标准SQL的基础上扩展一些自己的东西以吸引用户。

1.6.2 常见的关系型数据库

关系型数据库	开发公司	使用语言
SQL Server	微软公司	T-SQL
Oracle	甲骨文公司	PL/SQL
MySQL	MySQL AB 公司开发——甲骨文公司收购	MySQL

思考：已知标准SQL可以在所有的关系型数据库上运行，在Oracle上编写的PL/SQL能否在MySQL上运行?

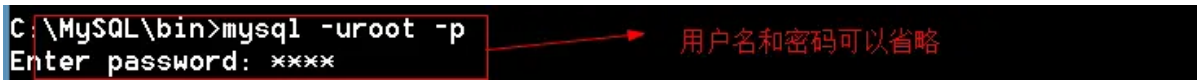
答：不能，只能运行标准SQL

1.7 连接服务器

通过命令行面板连接

```
host：主机          -h
username：用户名     -u
password：密码      -p
port：端口          -P
```

多学一招：如果MySQL服务器在本地，IP地址可以省略；如果MySQL服务器用的是3306端口，-P也是可以省略



1.8 关闭连接

方法一：exit

方法二：quit

方法三：\q

脚下留心：MySQL中的命令后面要加分号，windows命令行的命令后面不用加分号。

1.9 数据库的操作

1.9.1 显示数据库

语法：show databases

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| test               |
+--------------------+
4 rows in set (0.11 sec)
```

安装MySQL后，MySQL自带了4个数据库

1. information_schema：存储了MySQL服务器管理数据库的信息。
2. performance_schema：MySQL5.5新增的表，用来保存数据库服务器性能的参数
3. mysql：MySQL系统数据库，保存的登录用户名，密码，以及每个用户的权限等等
4. test：给用户学习和测试的数据库。

1.9.2 创建数据库

```
语法：create database [if not exists] `数据名` [字符编码]
```

创建数据库：

```
mysql> create database stu;
Query OK, 1 row affected (0.09 sec)
```

如果创建的数据库已存在，就会报错

```
mysql> create database stu;
ERROR 1007 (HY000): Can't create database 'stu'; database exists
```

解决：创建数据库的时候判断一下数据库是否存在，如果不存在再创建

```
mysql> create database if not exists stu;
Query OK, 1 row affected, 1 warning (0.00 sec)
```

如果数据库名是关键字和特殊字符要报错

解决：在特殊字符、关键字行加上反引号

```
mysql> create database `create`;
Query OK, 1 row affected (0.05 sec)
```

```
多学一招：为了创建数据库时万无一失，我们可以在所有的数据库名上加上反引号
```

创建数据库的时候可以指定字符编码

```
mysql> create database teacher charset=gbk;
Query OK, 1 row affected (0.01 sec)
gbk      简体中文
gb2312：  简体中文
utf8：    通用字符编码
```

```
脚下留心：创建数据库如果不指定字符编码，默认和MySQL服务器的字符编码是一致的。
```

1.9.3 删除数据库

```
语法：drop database [if exists] 数据库名
```

删除数据库

```
mysql> drop database teacher;
Query OK, 0 rows affected (0.00 sec)
```

如果删除的数据库不存在，会报错

```
mysql> drop database teacher;
ERROR 1008 (HY000): Can't drop database 'teacher'; database doesn't exist
mysql>
```

解决：删除之前判断一下，如果存在就删除

```
mysql> drop database if exists teacher;
Query OK, 0 rows affected, 1 warning (0.00 sec)
```

1.9.4 显示创建数据库的SQL语句

```
语法：show create database 数据库名
```

```
mysql> show create database stu;
+----------+-----------------------------------------------------------------+
| Database | Create Database                                                 |
+----------+-----------------------------------------------------------------+
| stu      | CREATE DATABASE `stu` /*!40100 DEFAULT CHARACTER SET utf8 */ |
+----------+-----------------------------------------------------------------+
1 row in set (0.01 sec)

mysql> show create database teacher;
+----------+-----------------------------------------------------------------+
| Database | Create Database                                                 |
+----------+-----------------------------------------------------------------+
| teacher  | CREATE DATABASE `teacher` /*!40100 DEFAULT CHARACTER SET gbk */ |
+----------+-----------------------------------------------------------------+
1 row in set (0.00 sec)
```

1.9.5 修改数据库

修改数据库的字符编码

语法：

```
alter database 数据库名 charset=字符编码
```

例题

```
mysql> alter database teacher charset=utf8;
Query OK, 1 row affected (0.00 sec)

mysql> show create database teacher;
+----------+-----------------------------------------------------------------+
| Database | Create Database                                                 |
+----------+-----------------------------------------------------------------+
| teacher  | CREATE DATABASE `teacher` /*!40100 DEFAULT CHARACTER SET utf8 */ |
+----------+-----------------------------------------------------------------+
1 row in set (0.00 sec)
```

1.9.6 选择数据库

语法:

```
use 数据库名
```

选择数据库

```
mysql> use stu;
Database changed
```

1.10 表的操作

1.10.1 显示所有表

语法:

```
show tables
```

1.10.2 创建表

语法:

```
create table [if not exists] 表名(
    字段名 数据类型 [null|not null] [auto_increment] [primary key] [comment],
    字段名 数据类型 [default]…
)engine=存储引擎
```

单词

```
null | not null      空|非空
default              默认值
auto_increment       自动增长
primary key          主键
comment              备注
engine               引擎   innodb  myisam  memory  引擎是决定数据存储的方式
```

创建简单的表

```
mysql> create database itcast;
Query OK, 1 row affected (0.00 sec)
```

```
mysql> use itcast;
Database changed
mysql> show tables;
Empty set (0.05 sec)

# 创建表
mysql> create table stu(
    -> id int,
    -> name varchar(30)
    -> );
Query OK, 0 rows affected (0.13 sec)
# 查看创建的表
mysql> show tables;
+------------------+
| Tables_in_itcast |
+------------------+
| stu              |
+------------------+
```

创建复杂的表

```
mysql> set names gbk;   # 设置字符编码
Query OK, 0 rows affected (0.05 sec)

mysql> create table if not exists teacher(
    -> id int auto_increment primary key comment '主键',
    -> name varchar(20) not null comment '姓名',
    -> phone varchar(20) comment '电话号码',
    -> `add` varchar(100) default '地址不详' comment '地址'
    -> )engine=innodb;
Query OK, 0 rows affected (0.09 sec)
```

多学一招：create table 数据库名.表名，用于给指定的数据库创建表

```
mysql> create table data.stu(  #给data数据库中创建stu表
    -> id int,
    -> name varchar(10));
Query OK, 0 rows affected (0.00 sec)
```

1.10.3 显示创建表的语句

语法：

```
show create table 表名
```

显示创建teacher表的语句

```
mysql> show create table teacher;
+---------+-------------------------------------------------------------------------
----
```

```
----------------------------------------------------------------------------
----
----------------------------------------------------------------------------
----
-------+
| Table   | Create Table


       |
+---------+-----------------------------------------------------------------
----
----------------------------------------------------------------------------
----
----------------------------------------------------------------------------
----
-------+
| teacher | CREATE TABLE `teacher` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '主键',
  `name` varchar(20) NOT NULL COMMENT '姓名',
  `phone` varchar(20) DEFAULT NULL COMMENT '电话号码',
  `add` varchar(100) DEFAULT '地址不详' COMMENT '地址',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8                          |
```

将两个字段竖着排列　　show create table 表名\G

```
mysql> show create table teacher\G;
*************************** 1. row ***************************
       Table: teacher
Create Table: CREATE TABLE `teacher` (
  `id` int(11) NOT NULL AUTO_INCREMENT COMMENT '主键',
  `name` varchar(20) NOT NULL COMMENT '姓名',
  `phone` varchar(20) DEFAULT NULL COMMENT '电话号码',
  `add` varchar(100) DEFAULT '地址不详' COMMENT '地址',
  PRIMARY KEY (`id`)
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)
```

1.10.4 查看表结构

语法:

```
desc[ribe] 表名
```

查看teacher表的结构

```
mysql> describe teacher;
+-------+-------------+------+-----+---------+----------------+
| Field | Type        | Null | Key | Default | Extra          |
+-------+-------------+------+-----+---------+----------------+
| id    | int(11)     | NO   | PRI | NULL    | auto_increment |
| name  | varchar(20) | NO   |     | NULL    |                |
| phone | varchar(20) | YES  |     | NULL    |                |
```

```
| add   | varchar(100) | YES  |     | 地址不详        |                |
+-------+--------------+------+-----+----------+----------------+
4 rows in set (0.08 sec)

mysql> desc teacher;
+-------+--------------+------+-----+----------+----------------+
| Field | Type         | Null | Key | Default  | Extra          |
+-------+--------------+------+-----+----------+----------------+
| id    | int(11)      | NO   | PRI | NULL     | auto_increment |
| name  | varchar(20)  | NO   |     | NULL     |                |
| phone | varchar(20)  | YES  |     | NULL     |                |
| add   | varchar(100) | YES  |     | 地址不详        |                |
+-------+--------------+------+-----+----------+----------------+
4 rows in set (0.01 sec)
```

1.10.5 删除表

语法：

```
drop table [if exists] 表1，表2,…
```

删除表

```
mysql> drop table stu;
Query OK, 0 rows affected (0.08 sec)
```

如果删除一个不存在的表就会报错，删除的时候可以判断一下，存在就删除。

```
mysql> drop table stu;
ERROR 1051 (42S02): Unknown table 'stu'

mysql> drop table if exists stu;
Query OK, 0 rows affected, 1 warning (0.00 sec)
```

可以一次删除多个表

```
mysql> drop table a1,a2;
Query OK, 0 rows affected (0.00 sec)
```

1.10.6 修改表

```
语法：alter table 表名
```

1、添加字段：alter table 表名add [column] 字段名 数据类型 [位置]

例题一：添加字段

```
mysql> alter table teacher add age int;
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc teacher;
```

```
+-------+-------------+------+-----+----------+----------------+
| Field | Type        | Null | Key | Default  | Extra          |
+-------+-------------+------+-----+----------+----------------+
| id    | int(11)     | NO   | PRI | NULL     | auto_increment |
| name  | varchar(20) | NO   |     | NULL     |                |
| phone | varchar(20) | YES  |     | NULL     |                |
| add   | varchar(100)| YES  |     | 地址不详      |                |
| age   | int(11)     | YES  |     | NULL     |                |
+-------+-------------+------+-----+----------+----------------+
5 rows in set (0.00 sec)
```

例题二：在第一个位置上添加字段

```
mysql> alter table teacher add email varchar(30) first;
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc teacher;
+-------+-------------+------+-----+----------+----------------+
| Field | Type        | Null | Key | Default  | Extra          |
+-------+-------------+------+-----+----------+----------------+
| email | varchar(30) | YES  |     | NULL     |                |
| id    | int(11)     | NO   | PRI | NULL     | auto_increment |
| name  | varchar(20) | NO   |     | NULL     |                |
| phone | varchar(20) | YES  |     | NULL     |                |
| add   | varchar(100)| YES  |     | 地址不详      |                |
| age   | int(11)     | YES  |     | NULL     |                |
+-------+-------------+------+-----+----------+----------------+
```

例题三：在指定的字段后添加字段

```
mysql> alter table teacher add sex varchar(2) after name;
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc teacher;
+-------+-------------+------+-----+----------+----------------+
| Field | Type        | Null | Key | Default  | Extra          |
+-------+-------------+------+-----+----------+----------------+
| email | varchar(30) | YES  |     | NULL     |                |
| id    | int(11)     | NO   | PRI | NULL     | auto_increment |
| name  | varchar(20) | NO   |     | NULL     |                |
| sex   | varchar(2)  | YES  |     | NULL     |                |
| phone | varchar(20) | YES  |     | NULL     |                |
| add   | varchar(100)| YES  |     | 地址不详      |                |
| age   | int(11)     | YES  |     | NULL     |                |
+-------+-------------+------+-----+----------+----------------+
7 rows in set (0.00 sec)
```

2、删除字段：alter table 表 drop [column] 字段名

```
mysql> alter table teacher drop email;
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

3、修改字段(改名改类型)：alter table 表 change [column] 原字段名 新字段名 数据类型 ...

将字段sex改为xingbie，数据类型为int

```
mysql> alter table teacher change sex xingbie int;
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

4、修改字段（不改名）:alter table 表 modify 字段名 字段属性...

将性别的数据类型改为varchar(2)

```
mysql> alter table teacher modify xingbie varchar(2);
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

5、修改引擎：alter table 表名 engine=引擎名

```
mysql> alter table teacher engine=myisam;
Query OK, 0 rows affected (0.05 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

6、修改表名：alter table 表名 rename to 新表名

```
mysql> alter table teacher rename to stu;
Query OK, 0 rows affected (0.00 sec)

mysql> show tables;
+------------------+
| Tables_in_itcast |
+------------------+
| stu              |
+------------------+
1 row in set (0.00 sec)
```

1.10.5 复制表

```
语法一：create table 新表 select 字段 from 旧表
```

特点：不能复制父表的主键，能够复制父表的数据

```
mysql> create table stu1 select * from stu;
Query OK, 1 row affected (0.06 sec)
Records: 1  Duplicates: 0  Warnings: 0

mysql> select * from stu1;    # 查看数据复制到新表中
+----+------+------+-------+
| id | name | addr | score |
```

```
+----+------+------+-------+
|  1 | rose | 上海    |    88 |
+----+------+------+-------+
1 row in set (0.00 sec)

mysql> desc stu1;    #  主键没有复制
+-------+-------------+------+-----+----------+-------+
| Field | Type        | Null | Key | Default  | Extra |
+-------+-------------+------+-----+----------+-------+
| id    | int(11)     | NO   |     | 0        |       |
| name  | varchar(20) | NO   |     | NULL     |       |
| addr  | varchar(50) | YES  |     | 地址不详        |       |
| score | int(11)     | YES  |     | NULL     |       |
+-------+-------------+------+-----+----------+-------+
4 rows in set (0.00 sec)
```

语法二：create table 新表 like 旧表

特点：只能复制表结构，不能复制表数据

```
Query OK, 0 rows affected (0.00 sec)

mysql> select * from stu2;   # 数据没有复制
Empty set (0.01 sec)

mysql> desc stu2;   # 主键复制了
+-------+-------------+------+-----+----------+----------------+
| Field | Type        | Null | Key | Default  | Extra          |
+-------+-------------+------+-----+----------+----------------+
| id    | int(11)     | NO   | PRI | NULL     | auto_increment |
| name  | varchar(20) | NO   |     | NULL     |                |
| addr  | varchar(50) | YES  |     | 地址不详        |                |
| score | int(11)     | YES  |     | NULL     |                |
+-------+-------------+------+-----+----------+----------------+
4 rows in set (0.00 sec)
```

1.11 数据操作

创建测试表

```
mysql> create table stu(
    -> id int auto_increment primary key comment '主键',
    -> name varchar(20) not null,
    -> addr varchar(50) default '地址不详',
    -> score int comment '成绩'
    -> );
Query OK, 0 rows affected (0.01 sec)
```

1.11.1 插入数据

插入一条数据

```
语法：insert into 表名 (字段名, 字段名,…) values (值1, 值1,…)
```

例题一：插入数据

```
mysql> insert into stu (id,name,addr,score) values (1,'tom','上海',88);
Query OK, 1 row affected (0.11 sec)
```

例题二：插入的字段可以和表的字段顺序不一致。值的顺序必须和插入字段的顺序一致。

```
mysql> insert into stu (name,score,addr,id) values ('berry',77,'北京',2);
Query OK, 1 row affected (0.00 sec)
```

例题三：可以插入部分字段，但是，非空字段必须插入

```
mysql> insert into stu (id,name,addr) values (3,'ketty','上海');
```

例题四：自动增长字段不用插入，数据库会自动插入增长的数字

```
mysql> insert into stu (name,addr) values ('rose','北京');
Query OK, 1 row affected (0.00 sec)
```

例题五：自动增长列的值插入null即可

```
mysql> insert into stu (id,name,addr,score) values (null,'李白','上海',66);
Query OK, 1 row affected (0.00 sec)
```

例题六：插入值的顺序和个数与表字段的顺序和个数一致，插入的字段可以省略

```
mysql> insert into stu values (null,'杜甫','北京',null);
Query OK, 1 row affected (0.00 sec)
```

例题七：通过default关键字插入默认值

```
mysql> insert into stu values (null,'李清照',default,66);
```

```
脚下留心：
1、插入字段的顺序与值的顺序必须一致
```

插入多条数据

```
mysql> insert into stu values (null,'辛弃疾',default,66),(null,'岳飞','河南',77);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0
```

1.11.2 更新数据

语法：

```
update 表名 set 字段=值 [where 条件]
```

例题一：将1号学生的地址改成山东

```
mysql> update stu set addr='山东' where id=1
```

例题二：将ketty的成绩改为99

```
mysql> update stu set score=99 where name='ketty';
```

例题三：将berry地址改成上海，成绩改成66

```
mysql> update stu set addr='上海',score=66 where name='berry';
```

例题四：将上海的学生成绩改为60

```
mysql> update stu set score=60 where addr='上海';
```

例题五：条件可以省略，如果省略，更改所有数据（将所有数据的地址改为湖南，成绩改为70）

```
mysql> update stu set addr='湖南',score=70;
```

例题六：将2、3的学生成绩改为65

```
mysql> update stu set score=65 where id=2 or id=3;
```

1.11.3 删除数据

语法

```
delete from 表名 [where 条件]
```

例题一：删除学号是1号的学生

```
mysql> delete from stu where id=1;
```

例题二：删除成绩小于等于65分的

```
mysql> delete from stu where score<=65;
```

例题三：删除表中所有记录

```
mysql> delete from stu;
```

1.11.4 清空表

语法：

```
truncate table 表名
```

例题

```
mysql> truncate table stu;
Query OK, 0 rows affected (0.00 sec)
```

```
脚下留心：delete from 表和truncate table 表区别？
delete from 表：遍历表记录，一条一条的删除
truncate table：将原表销毁，再创建一个同结构的新表。就清空表而言，这种方法效率高。
```

1.11.5查询表

语法：

```
select 列名 from 表
```

例题：

```
mysql> select name,score from stu;
+------+-------+
| name | score |
+------+-------+
| rose |    88 |
+------+-------+
1 row in set (0.00 sec)

mysql> select id,name,addr,score from stu;
+----+------+------+-------+
| id | name | addr | score |
+----+------+------+-------+
|  1 | rose | 上海    |    88 |
+----+------+------+-------+
1 row in set (0.00 sec)

mysql> select * from stu;  # *表示所有字段
+----+------+------+-------+
| id | name | addr | score |
+----+------+------+-------+
|  1 | rose | 上海    |    88 |
+----+------+------+-------+
1 row in set (0.00 sec)
```

1.12 SQL分类

DDL（data definition language）数据库定义语言CREATE、ALTER、DROP、SHOW

DML（data manipulation language）数据操纵语言SELECT、UPDATE、INSERT、DELETE

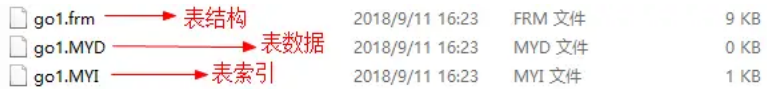
DCL（Data Control Language）数据库控制语言,是用来设置或更改数据库用户或角色权限的语句

1.13 数据表的文件介绍

一个数据库对应一个文件夹

一个表对应一个或多个文件

引擎是myisam，一个表对应三个文件



引擎是innodb,一个表对应一个表结构文件

所有的innodb引擎的数据统一的存放在data\ibdata1文件中。如果数据量很大，MySQL会自动的创建ibdata2，ibdata3，...，目的就是为了便于管理。

引擎是memory，数据存储在内存中，重启服务数据丢失，但是读取速度非常快。

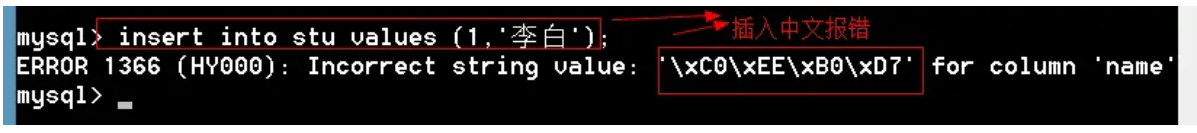
1.14 字符集

字符集：字符在保存和传输时对应的二进制编码集合。

创建测试数据库

```
mysql> create table stu(
    -> id int primary key,
    -> name varchar(20)
    -> );
Query OK, 0 rows affected (0.00 sec)
```

插入中文报错



分析原因：

客户端通过GBK发送的命令

但是，服务用utf8解释命令

```
mysql> show variables like 'character_set_%';
+--------------------------+----------------------------+
| Variable_name            | Value                      |
+--------------------------+----------------------------+
| character_set_client     | utf8                       |
| character_set_connection | utf8                       |
| character_set_database   | utf8                       |
| character_set_filesystem | binary                     |
| character_set_results    | utf8                       |
| character_set_server     | utf8                       |
| character_set_system     | utf8                       |
| character_sets_dir       | C:\MySQL\share\charsets\   |
+--------------------------+----------------------------+
```

通过utf8解释客户端的命令

以utf8字符返回到客户端

设置服务器，用gbk字符编码接受客户端发来的命令

```
mysql> set character_set_client=gbk;
Query OK, 0 rows affected (0.08 sec)
```

测试：插入中文，成功

```
mysql> insert into stu values (1,'李白');
Query OK, 1 row affected (0.00 sec)
```

查询数据，发现数据乱码

```
mysql> select * from stu;
+----+--------+
| id | name   |
+----+--------+
|  1 | 鏉庣櫧 |
+----+--------+
```

中文乱码

原因：以utf返回的结果，客户端用gbk来接受

解决：服务器用gbk返回数据

```
mysql> set character_set_results=gbk;
Query OK, 0 rows affected (0.00 sec)
```

再次测试，查询数据

```
mysql> select * from stu;
+----+------+
| id | name |
+----+------+
|  1 | 李白     |
+----+------+
```

总结：客户端编码、character_set_client、character_set_results三个编码的值一致即可操作中文。

多学一招：我们只要设置“set names 字符编码”，就可以更改character_set_client、character_set_results的值。

```
mysql> set names gbk;
Query OK, 0 rows affected (0.00 sec)

mysql> show variables like 'character_set_%';
+--------------------------+--------------------------+
| Variable_name            | Value                    |
+--------------------------+--------------------------+
| character_set_client     | gbk                      |
| character_set_connection | gbk                      |
| character_set_database   | utf8                     |
| character_set_filesystem | binary                   |
| character_set_results    | gbk                      |
| character_set_server     | utf8                     |
| character_set_system     | utf8                     |
| character_sets_dir       | C:\MySQL\share\charsets\ |
+--------------------------+--------------------------+
```