# 数据库介绍

## 数据库概述

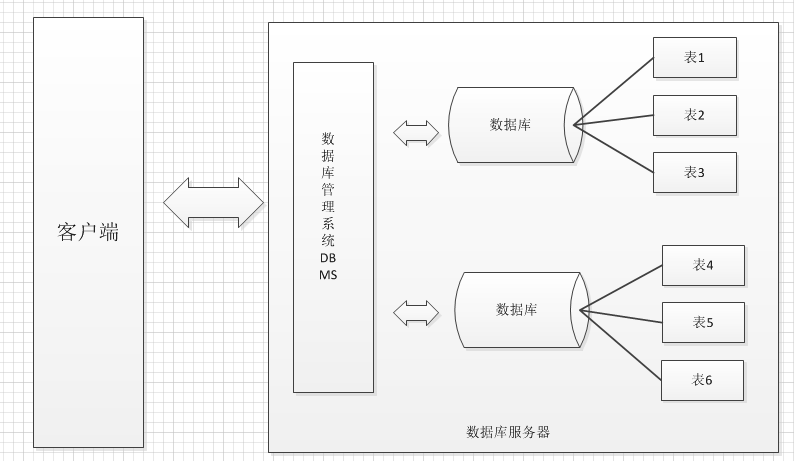
* 什么是数据库

数据库就是存储数据的仓库，其本质是一个文件系统，数据按照特定的格式将数据存储起来，用户可以对数据库中的数据进行增加，修改，删除及查询操作。

* 什么是数据库管理系统

数据库管理系统（DataBase Management System，DBMS）：指一种操作和管理数据库的大型软件，用于建立、使用和维护数据库，对数据库进行统一管理和控制，以保证数据库的安全性和完整性。用户通过数据库管理系统访问数据库中表内的数据。

* 数据库与数据库管理系统的关系



## 数据库表

数据库中以表为组织单位存储数据。

表类似我们的Java类，每个字段都有对应的数据类型。

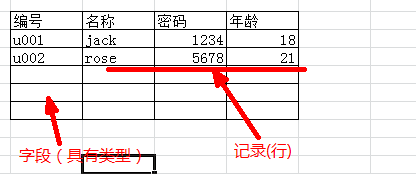


那么用我们熟悉的java程序来与关系型数据对比，就会发现以下对应关系。

类----------表

类中属性----------表中字段

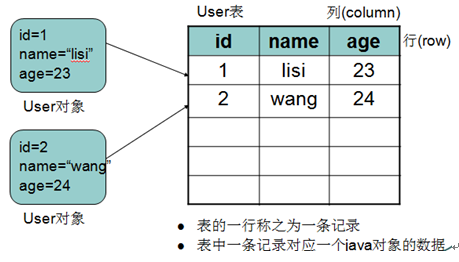
对象----------记录



## 表数据

根据表字段所规定的数据类型，我们可以向其中填入一条条的数据，而表中的每条数据类似类的实例对象。表中的一行一行的信息我们称之为记录。

* 表记录与java类对象的对应关系



## 常见数据库

* 常见的数据库管理系统

MYSQL ：开源免费的数据库，小型的数据库.已经被Oracle收购了.MySQL6.x版本也开始收费。

Oracle ：收费的大型数据库，Oracle公司的产品。Oracle收购SUN公司，收购MYSQL。

DB2 ：IBM公司的数据库产品,收费的。常应用在银行系统中.

SQLServer：MicroSoft 公司收费的中型的数据库。C#、.net等语言常使用。

SyBase ：已经淡出历史舞台。提供了一个非常专业数据建模的工具PowerDesigner。

SQLite : 嵌入式的小型数据库，应用在手机端。

常用数据库：**MYSQL**，Oracle．

这里使用MySQL数据库。MySQL中可以有多个数据库，数据库是真正存储数据的地方。

# MySql数据库

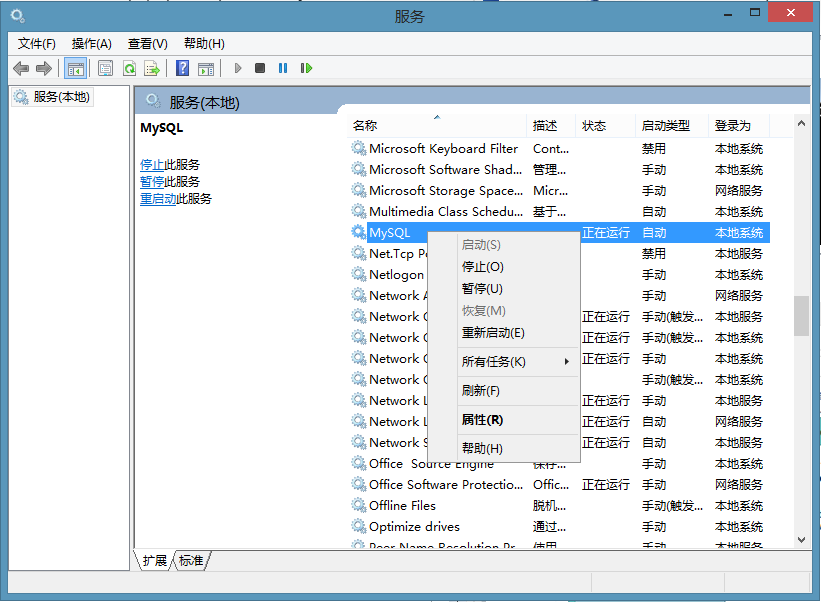
## MySql安装

* 安装

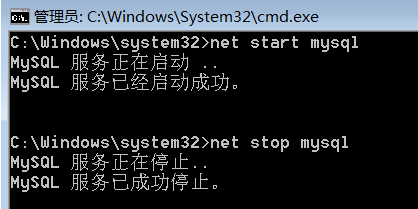
参考MySQL安装图解.doc



安装后，MySQL会以windows服务的方式为我们提供数据存储功能。开启和关闭服务的操作：右键点击我的电脑→管理→服务→可以找到MySQL服务开启或停止。



也可以在DOS窗口，通过命令完成MySQL服务的启动和停止（必须以管理员身份运行cmd命令窗口）

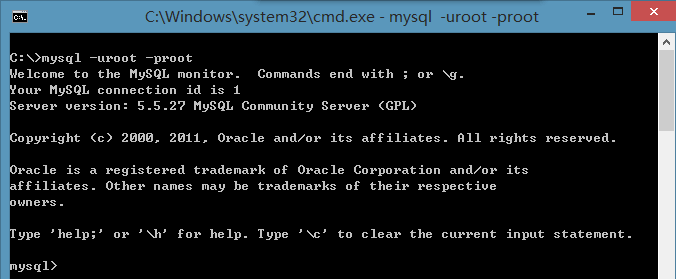


## 登录MySQL数据库

MySQL是一个需要账户名密码登录的数据库，登陆后使用，它提供了一个默认的root账号，使用安装时设置的密码即可登录。

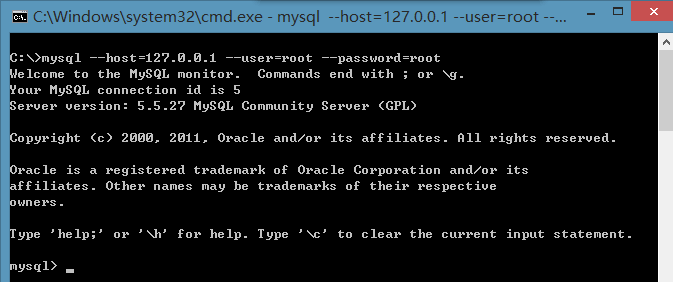
格式1：cmd> mysql –u用户名 –p密码

例如：mysql -uroot –proot



格式2：cmd> mysql --host=ip地址 --user=用户名 --password=密码

例如：mysql --host=127.0.0.1 --user=root --password=root



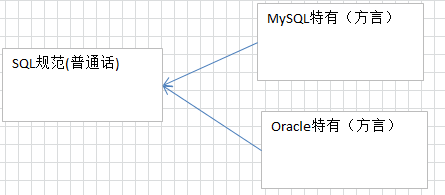
# SQL语句

## SQL概述

### SQL语句介绍

数据库是不认识JAVA语言的，但是我们同样要与数据库交互，这时需要使用到数据库认识的语言SQL语句，它是数据库的代码。

结构化查询语言(Structured Query Language)简称SQL，是关系型数据库管理系统都需要遵循的规范。不同的数据库生产厂商都支持SQL语句，但都有特有内容。

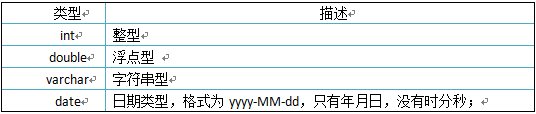


### SQL语句分类

* SQL分类：
  + 数据定义语言：简称DDL(Data Definition Language)，用来定义数据库对象：数据库，表，列等。关键字：create，alter，drop等
  + 数据操作语言：简称DML(Data Manipulation Language)，用来对数据库中表的记录进行更新。关键字：insert，delete，update等
  + 数据控制语言：简称DCL(Data Control Language)，用来定义数据库的访问权限和安全级别，及创建用户。
  + 数据查询语言：简称DQL(Data Query Language)，用来查询数据库中表的记录。关键字：select，from，where等

### SQL通用语法

* SQL语句可以单行或多行书写，以分号结尾
* 可使用空格和缩进来增强语句的可读性
* MySQL数据库的SQL语句不区分大小写，关键字建议使用大写
  + 例如：SELECT \* FROM user。
* 同样可以使用/\*\*/的方式完成注释
* MySQL中的我们常使用的数据类型如下



详细的数据类型如下(不建议详细阅读！)

|  |  |  |
| --- | --- | --- |
| **分类** | **类型名称** | **说明** |
| **整数类型** | tinyInt | 很小的整数 |
| smallint | 小的整数 |
| mediumint | 中等大小的整数 |
| int(integer) | 普通大小的整数 |
| **小数类型** | float | 单精度浮点数 |
| double | 双精度浮点数 |
| decimal（m,d） | 压缩严格的定点数 |
| **日期类型** | year | YYYY 1901~2155 |
| time | HH:MM:SS -838:59:59~838:59:59 |
| date | YYYY-MM-DD 1000-01-01~9999-12-3 |
| datetime | YYYY-MM-DD HH:MM:SS  1000-01-01 00:00:00~ 9999-12-31 23:59:59 |
| timestamp | YYYY-MM-DD HH:MM:SS  1970~01~01 00:00:01 UTC~2038-01-19 03:14:07UTC |
| **文本、二进制类型** | CHAR(M) | M为0~255之间的整数 |
| VARCHAR(M) | M为0~65535之间的整数 |
| TINYBLOB | 允许长度0~255字节 |
| BLOB | 允许长度0~65535字节 |
| MEDIUMBLOB | 允许长度0~167772150字节 |
| LONGBLOB | 允许长度0~4294967295字节 |
| TINYTEXT | 允许长度0~255字节 |
| TEXT | 允许长度0~65535字节 |
| MEDIUMTEXT | 允许长度0~167772150字节 |
| LONGTEXT | 允许长度0~4294967295字节 |
| VARBINARY(M) | 允许长度0~M个字节的变长字节字符串 |
| BINARY(M) | 允许长度0~M个字节的定长字节字符串 |

## DDL之数据库操作：database

### 创建数据库

格式:

\* create database 数据库名;

\* create database 数据库名 character set 字符集;

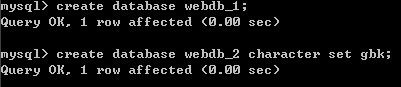
例如：

#创建数据库 数据库中数据的编码采用的是安装数据库时指定的默认编码 utf8

CREATE DATABASE webdb\_1;

#创建数据库 并指定数据库中数据的编码

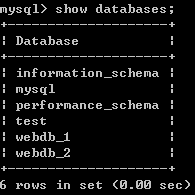
CREATE DATABASE webdb\_2 CHARACTER SET utf8;



### 查看数据库

查看数据库MySQL服务器中的所有的数据库:

show databases;

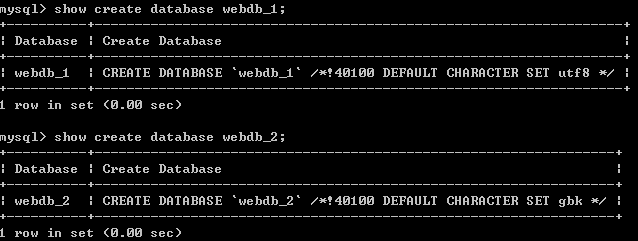


查看某个数据库的定义的信息:

show create database 数据库名;

例如：

show create database webdb\_1;



### 删除数据库

drop database 数据库名称;

例如：

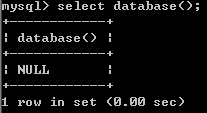
drop database webdb\_2;



### 使用数据库

* 查看正在使用的数据库:

select database();



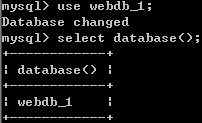
* 其他的数据库操作命令

切换数据库：

use 数据库名;

例如：

use webdb\_1;



## DDL之表操作：table

### 创建表

* 格式：

create table 表名(

字段名 类型(长度) [约束],

字段名 类型(长度) [约束],

...

);

类型：

varchar(n) 字符串

int 整形

double 浮点

date 时间

timestamp 时间戳

约束：(详情查看第四章)

primary key 主键，被主键修饰字段中的数据，不能重复、不能为null。

例如：

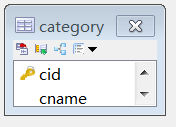
###创建分类表

CREATE TABLE category (

cid INT primary key, #分类ID

cname VARCHAR(100) #分类名称

);



### 查看表

* 查看数据库中的所有表：

格式：show tables;

* 查看表结构：

格式：desc 表名;

例如：desc category;

### 删除表

* 格式：drop table 表名;

例如：drop table category;

### 修改表结构格式：

* alter table 表名 add 列名 类型(长度) [约束];

作用：修改表添加列.

例如：

#1，为分类表添加一个新的字段为 分类描述 varchar(20)

ALTER TABLE category ADD `desc` VARCHAR(20);

* alter table 表名 modify 列名 类型(长度) 约束;

作用：修改表修改列的类型长度及约束.

例如：

#2, 为分类表的描述字段进行修改，类型varchar(50) 添加约束 not null

ALTER TABLE category MODIFY `desc` VARCHAR(50) NOT NULL;

* alter table 表名 change 旧列名 新列名 类型(长度) 约束;

作用：修改表修改列名.

例如：

#3, 为分类表的分类名称字段进行更换 更换为 description varchar(30)

ALTER TABLE category CHANGE `desc` description VARCHAR(30);

* alter table 表名 drop 列名;

作用：修改表删除列.

例如：

#4, 删除分类表中description这列

ALTER TABLE category DROP description;

* rename table 表名 to 新表名;

作用：修改表名

例如：

#5, 为分类表category 改名成 category2

RENAME TABLE category TO category2;

* alter table 表名 character set 字符集(了解);

作用：修改表的字符集

例如：

#6, 为分类表 category 的编码表进行修改，修改成 gbk

ALTER TABLE category CHARACTER SET gbk;

## DML数据操作语言

### 插入表记录：insert

* 语法：

-- 向表中插入某些字段

insert into 表 (字段1,字段2,字段3..) values (值1,值2,值3..);

-- 向表中插入所有字段,字段的顺序为创建表时的顺序

insert into 表 values (值1,值2,值3..);

* 注意：
  + 值与字段必须对应，个数相同，类型相同
  + 值的数据大小必须在字段的长度范围内
  + 除了数值类型外，其它的字段类型的值必须使用引号引起。（建议单引号）
  + 如果要插入空值，可以不写字段，或者插入 null。
* 例如：

INSERT INTO category(cid,cname) VALUES('c001','电器');

INSERT INTO category(cid,cname) VALUES('c002','服饰');

INSERT INTO category(cid,cname) VALUES('c003','化妆品');

INSERT INTO category(cid,cname) VALUES('c004','书籍');

INSERT INTO category(cid) VALUES('c005');

INSERT INTO category(cname,cid) VALUES('耗材','c006');

### 更新表记录：update

用来修改指定条件的数据，将满足条件的记录指定列修改为指定值

* 语法：

--更新所有记录的指定字段

update 表名 set 字段名=值,字段名=值,...;

--更新符合条件记录的指定字段

update 表名 set 字段名=值,字段名=值,... where 条件;

* 注意：
  + - 列名的类型与修改的值要一致.
    - 修改值得时候不能超过最大长度.
    - 除了数值类型外，其它的字段类型的值必须使用引号引起

### 删除记录：delete

* 语法：

delete from 表名 [where 条件];

或者

truncate table 表名;

* 面试题：

删除表中所有记录使用delete from 表名; 还是用truncate table 表名;

删除方式：delete 一条一条删除，不清空auto\_increment记录数。

truncate 直接将表删除，重新建表，auto\_increment将置为零，从新开始。

## DOS操作数据乱码解决

我们在dos命令行操作中文时，会报错

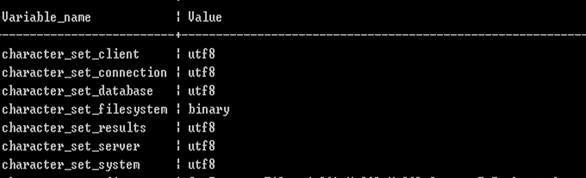
insert into category(cid,cname) values(‘c010’,’中文’);

ERROR 1366 (HY000): Incorrect string value: '\xB7\xFE\xD7\xB0' for column 'cname' at row 1

错误原因:因为mysql的客户端设置编码是utf8,而系统的cmd窗口编码是gbk

1）查看MySQL内部设置的编码

show variables like 'character%'; 查看所有mysql的编码

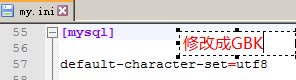


2）需要修改client、connection、results的编码一致（GBK编码）

解决方案1：在cmd命令窗口中输入命令，此操作当前窗口有效，为临时方案。

set names gbk;

解决方案2：安装目录下修改my.ini文件，重启服务所有地方生效。



# SQL约束

## 主键约束

PRIMARY KEY 约束唯一标识数据库表中的每条记录。

主键必须包含唯一的值。

主键列不能包含 NULL 值。

每个表都应该有一个主键，并且每个表只能有一个主键。

### 添加主键约束

* 方式一：创建表时，在字段描述处，声明指定字段为主键：

CREATE TABLE Persons

(

Id\_P int PRIMARY KEY,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 方式二：创建表时，在constraint约束区域，声明指定字段为主键：
  + 格式：[constraint 名称] primary key (字段列表)
  + 关键字constraint可以省略，如果需要为主键命名，constraint不能省略，主键名称一般没用。
  + 字段列表需要使用小括号括住，如果有多字段需要使用逗号分隔。声明两个以上字段为主键，我们称为联合主键。

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255),

CONSTRAINT pk\_PersonID PRIMARY KEY (FirstName,LastName)

)

或

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255),

PRIMARY KEY (FirstName,LastName)

)

* 方式三：创建表之后，通过修改表结构，声明指定字段为主键：

ALTER TABLE Persons ADD [CONSTRAINT 名称] PRIMARY KEY (字段列表)

CREATE TABLE Persons

(

FirstName varchar(255),

LastName varchar(255),

Address varchar(255),

City varchar(255)

)

ALTER TABLE Persons ADD PRIMARY KEY (FirstName,LastName)

### 删除主键约束

如需撤销 PRIMARY KEY 约束，请使用下面的 SQL：

ALTER TABLE Persons DROP PRIMARY KEY

## 自动增长列

我们通常希望在每次插入新记录时，数据库自动生成字段的值。

我们可以在表中使用 auto\_increment（自动增长列）关键字，自动增长列类型必须是整形，自动增长列必须为键(一般是主键)。

* 下列 SQL 语句把 "Persons" 表中的 "P\_Id" 列定义为 auto\_increment 主键

CREATE TABLE Persons

(

P\_Id int PRIMARY KEY AUTO\_INCREMENT,

LastName varchar(255),

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 向persons添加数据时，可以不为P\_Id字段设置值，也可以设置成null，数据库将自动维护主键值：

INSERT INTO Persons (FirstName,LastName) VALUES ('Bill','Gates')

INSERT INTO Persons (P\_Id,FirstName,LastName) VALUES (NULL,'Bill','Gates')

* 扩展：默认AUTO\_INCREMENT 的开始值是 1，如果希望修改起始值，请使用下列 SQL 语法：

ALTER TABLE Persons AUTO\_INCREMENT=100

## 非空约束

NOT NULL 约束强制列不接受 NULL 值。

NOT NULL 约束强制字段始终包含值。这意味着，如果不向字段添加值，就无法插入新记录或者更新记录。

* 下面的 SQL 语句强制 "Id\_P" 列和 "LastName" 列不接受 NULL 值：

CREATE TABLE Persons

(

Id\_P int NOT NULL,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

## 唯一约束

UNIQUE 约束唯一标识数据库表中的每条记录。

UNIQUE 和 PRIMARY KEY 约束均为列或列集合提供了唯一性的保证。

PRIMARY KEY 拥有自动定义的 UNIQUE 约束。

请注意，每个表可以有多个 UNIQUE 约束，但是每个表只能有一个 PRIMARY KEY 约束。

### 添加唯一约束

与主键添加方式相同，共有3种，

* 方式1：创建表时，在字段描述处，声明唯一：

CREATE TABLE Persons

(

Id\_P int UNIQUE,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255)

)

* 方式2：创建表时，在约束区域，声明唯一：

CREATE TABLE Persons

(

Id\_P int,

LastName varchar(255) NOT NULL,

FirstName varchar(255),

Address varchar(255),

City varchar(255),

CONSTRAINT 名称UNIQUE (Id\_P)

)

* 方式3：创建表后，修改表结构，声明字段唯一：

ALTER TABLE Persons ADD [CONSTRAINT 名称] UNIQUE (Id\_P)

### 删除唯一约束

* 如需撤销 UNIQUE 约束，请使用下面的 SQL：

ALTER TABLE Persons DROP INDEX 名称

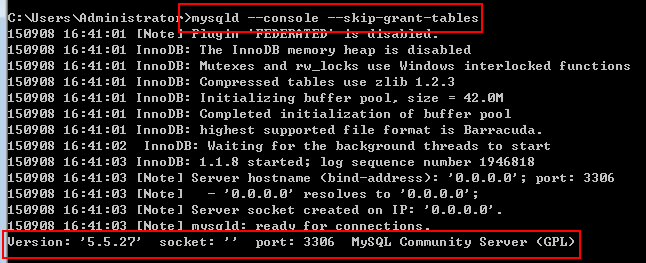
* 如果添加唯一约束时，没有设置约束名称，默认是当前字段的字段名。

## 外键约束

FOREIGN KEY 表示外键约束，将在多表中学习。

# MySQL数据库密码重置(扩展)

1. 停止mysql服务器 运行输入services.msc 停止mysql服务
2. 在cmd下,输入mysqld --console --skip-grant-tables 启动服务器,出现一下页面,不要关闭该窗口



1. 新打开cmd,输入mysql -uroot 不需要密码

use mysql;

update user set password=password('abc') WHERE User='root';

1. 关闭两个cmd窗口