1.1 今日目标

1. 了解MySQL常用的客户端软件
2. 理解整型的使用
3. 理解浮点型的使用
4. 理解字符型的使用
5. 理解枚举型的使用
6. 理解集合型的使用
7. 理解日期型的使用
8. 理解非空约束
9. 理解默认值约束
10. 理解自动增长
11. 理解主键约束
12. 理解唯一键约束
13. 知道备注
14. 知道SQL注释
15. 理解外键约束

1.2 数据类型——值类型

1.2.1 整型

类型	字节	范围
tinyint	1	-128~127
smallint	2	-32768~32767
mediumint	3	-8388608~8388607
int	4	-2^31^~2^31^-1
bigint	8	-2^63^~2^63^-1

1、无符号整数（unsigned）：无符号数没有负数，正数部分是有符号的两倍。

例题

```
mysql> create table stu(
    -> id smallint unsigned auto_increment primary key comment '主键',
    -> age tinyint unsigned not null comment '年龄',
    -> money bigint unsigned comment '存款'
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> desc stu;
+-------+---------------------+------+-----+---------+----------------+
| Field | Type                | Null | Key | Default | Extra          |
```

```
+-------+---------------------+------+-----+---------+----------------+
| id    | smallint(5) unsigned | NO   | PRI | NULL    | auto_increment |
| age   | tinyint(3) unsigned  | NO   |     | NULL    |                |
| money | bigint(20) unsigned  | YES  |     | NULL    |                |
+-------+---------------------+------+-----+---------+----------------+
3 rows in set, 3 warnings (0.00 sec)
```

2、整型支持显示宽度（最小的显示位数） 比如int(5)，如果数值的位数小于5位，前面加上前导0。比如输入12，显示00012；大于5位就不添加前导0。

脚下留心：必须结合zerofill才起作用

```
mysql> create table stu(
    -> id int(5),
    -> age int(5) zerofill   # 填充前导0
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> desc stu;
+-------+-------------------------+------+-----+---------+-------+
| Field | Type                    | Null | Key | Default | Extra |
+-------+-------------------------+------+-----+---------+-------+
| id    | int(5)                  | YES  |     | NULL    |       |
| age   | int(5) unsigned zerofill | YES  |     | NULL    |       |
+-------+-------------------------+------+-----+---------+-------+
2 rows in set (0.02 sec)

mysql> insert into stu values (1,11);
mysql> insert into stu values (1111111,2222222);
Query OK, 1 row affected (0.00 sec)

mysql> select * from stu;
+---------+---------+
| id      | age     |
+---------+---------+
|       1 |   00011 |
| 1111111 | 2222222 |    # 注意：age填充了前导0
+---------+---------+
2 rows in set (0.00 sec)
```

1.2.2 浮点型（保存近似值小数）

浮点型	占用字节	范围
float（单精度）	4	-3.4E+38~3.4E+38
double（双精度）	8	-1.8E+308~1.8E+308

1、浮点数声明：float(M,D) double(M,D)

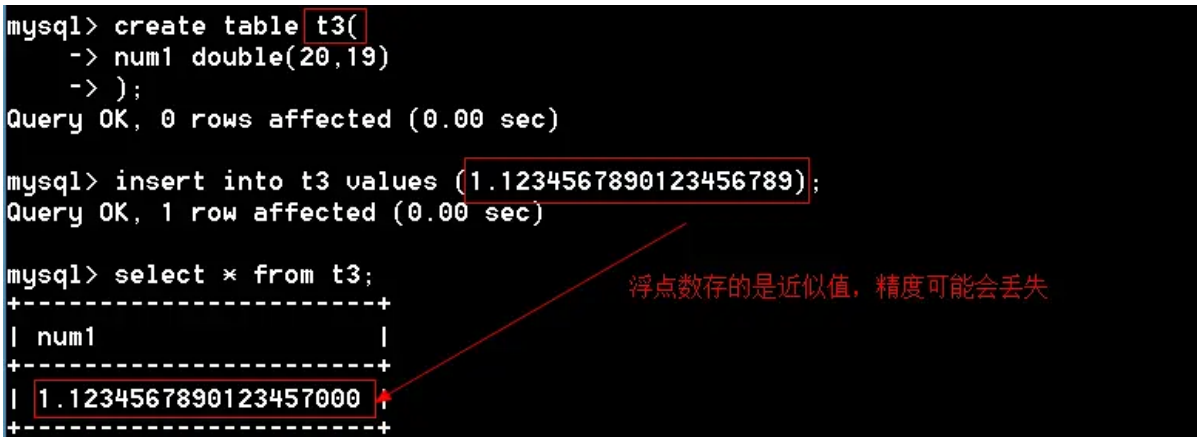
M：总位数

D：小数位数

例题；

```
mysql> create table t1(
    -> num1 float(5,2),    #总位数是5，小数位数是2，那么整数位数是3，
    -> num2 double(4,1)
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> insert into t1 values (1.23,1.23); #如果精度超出了允许的范围，会四舍五入
Query OK, 1 row affected (0.00 sec)

mysql> select * from t1;
+------+------+
| num1 | num2 |
+------+------+
| 1.23 |  1.2 |    #如果精度超出了允许的范围，会四舍五入
+------+------+
1 row in set (0.00 sec)
```

2、浮点的精度可能会丢失【精度指的是小数】



1.2.3 定点数

语法：decimal(M,D)

```
mysql> create table t4(
    -> num decimal(20,19)
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t4 values (1.1234567890123456789);
Query OK, 1 row affected (0.01 sec)

mysql> select * from t4;
+----------------------+
| num                  |
+----------------------+
| 1.1234567890123456789 |
+----------------------+
1 row in set (0.00 sec)
```

多学一招：

1、定点数是变长的，大致每9个数字用4个字节来存储。定点数之所以能保存精确的小数，因为整数和小数是分开存储的。占用的资源比浮点数要多。

2、定点数和浮点数都支持显示宽度和无符号数。

1.3 数据类型——字符型

数据类型	描述	长度
char(长度)	定长	最大255
varchar(长度)	变长	最大65535
tinytext	大段文本	2^8^-1=255
text	大段文本	2^16^-1=65535
mediumtext	大段文本	2^24^-1
longtext	大段文本	2^32^-1

1、char(10)和varchar(10)的区别?

答：相同点：它们最多只能保存10个字符；

不同点：char不回收多余的字符，varchar会回收多余的字符。

char效率高，浪费空间，varchar节省空间，效率比char低。

2、char的最大长度是255。

```
mysql> create table t5(
    -> name char(256)
    -> );
ERROR 1074 (42000): Column length too big for column 'name' (max = 255); use BLOB or TEXT instead
mysql>
```

最多只能到255

3、varchar理论长度是65535字节,实际根本达不到。具体长度与字符编码有关。

```
mysql> create table t6(
    -> name varchar(65535)
    -> )charset=utf8;
ERROR 1074 (42000): Column length too big for column 'name' (max = 21845); use BLOB or TEXT instead
```

```
mysql> create table t6(
    -> name varchar(65535)
    -> )charset=gbk;
ERROR 1074 (42000): Column length too big for column 'name' (max = 32767); use BLOB or TEXT instead
```

4、一个记录的总长度不能超过65535个字节。

5、大块文本（text）不计算在总长度中,一个大块文本只占用10个字节来保存文本的地址。

```
mysql> create table t7(
    -> name varchar(20),
    -> jieshao text);
```

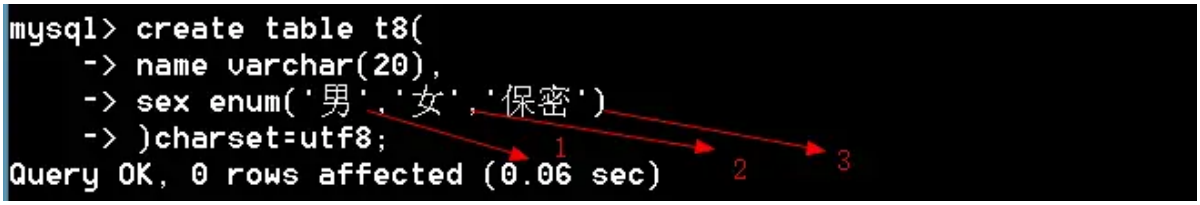
保存的是地址，占用10个字节左右

1.4 数据类型——枚举（enum）

1、从集合中选择一个数据（单选）

```
mysql> create table t8(
    -> name varchar(20),
    -> sex enum('男','女','保密')   # 枚举
    -> )charset=utf8;
Query OK, 0 rows affected (0.06 sec)

mysql> insert into t8 values ('tom','男');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t8 values ('berry','女');
Query OK, 1 row affected (0.05 sec)

mysql> insert into t8 values ('rose','未知');   # 报错，只能插入枚举值
ERROR 1265 (01000): Data truncated for column 'sex' at row 1
mysql> select * from t8;
+-------+------+
| name  | sex  |
+-------+------+
| tom   | 男     |
| berry | 女    |
+-------+------+
```

2、MySQL的枚举类型是通过整数来管理的，第一个值是1，第二个值是2，以此类推。




```
mysql> select sex+0 from t8;
+-------+
| sex+0 |
+-------+
|     1 |
|     2 |
+-------+
```

3、既然枚举在数据库内部存储的是整数，那么可以直接插入数字

```
mysql> insert into t8 values ('rose',3);  # 可以直接插入数字
Query OK, 1 row affected (0.00 sec)

mysql> select * from t8;
+-------+------+
| name  | sex  |
+-------+------+
| tom   | 男     |
| berry | 女     |
| rose  | 保密     |
+-------+------+
3 rows in set (0.00 sec)
```

枚举的优点：

1、 运行速度快（数字比字符串运算速度快）

2、 限制数据，保证数据完整性

3、 节省空间

思考：已知枚举占用2个字节，请问最多有多少个枚举值？

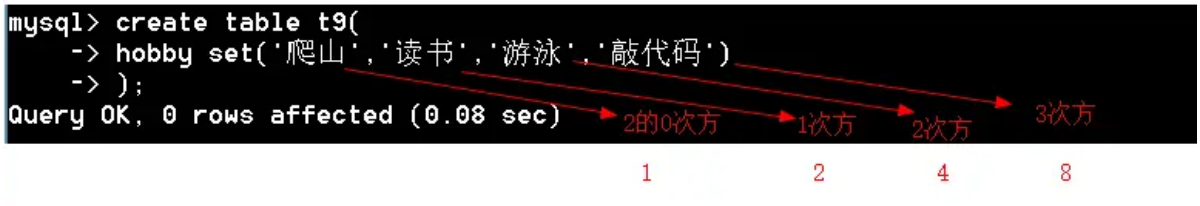
答：2个字节=16位，可以保存数字（0-65535），枚举是从1开始，所以枚举最多可以有65535个枚举值。

1.5 数据类型——集合（set）

从集合中选择一些数据（多选）

```
mysql> create table t9(
    -> hobby set('爬山','读书','游泳','敲代码')
    -> );
Query OK, 0 rows affected (0.08 sec)

mysql> insert into t9 values ('爬山');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t9 values ('爬山,游泳');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t9 values ('游泳,爬山');  # 插入顺序不一样，但是显示的顺序是一样的
Query OK, 1 row affected (0.02 sec)

mysql> insert into t9 values ('爬山,游泳,开车');  # 报错，插入集合中没有的选项会报错
ERROR 1265 (01000): Data truncated for column 'hobby' at row 1
```

每个集合的元素都分配一个固定的数字，分配的方式从左往右按2的0、1、2、...次方



思考：已知集合占用8个字节，最多可以表示几个选项？

答：8个字节=64位，一个位表示1个选项，最多可以表示64个选项。

1.6 数据类型——日期类型

数据类型	描述
datetime	日期时间，占用8个字节
date	日期 占用3个字节
time	时间 占用3个字节
timestamp	时间戳，占用4个字节
year	年份 占用1个字节

1、datetime 格式：年-月-日 小时:分钟:秒

```
mysql> create table t10(
    -> field datetime
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> insert into t10 values ('2025-10-12 10:12:36');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t10 values ('100-10-12 10:12:36');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t10 values ('10000-10-12 10:12:36');  #datetime保存范围是:
1~9999年
ERROR 1292 (22007): Incorrect datetime value: '10000-10-12 10:12:36' for column
'field' at row 1

mysql> select * from t10;
+---------------------+
| field               |
+---------------------+
| 2025-10-12 10:12:36 |
| 0100-10-12 10:12:36 |
+---------------------+
2 rows in set (0.00 sec)
```

2、date 日期格式

```
mysql> create table t11(
    -> field date
    -> );
Query OK, 0 rows affected (0.00 sec)
```

```
mysql> insert into t11 values ('2025-10-12');
Query OK, 1 row affected (0.00 sec)

mysql> select * from t11;
+------------+
| field      |
+------------+
| 2025-10-12 |
+------------+
```

3、timestamp：时间戳

timestamp类型和 datetime类型在表现上是一样的。他们的区别：
datetime是从1到9999，而timestamp从1970年~2038年，2038年01月19日11:14:07秒以后就超出timestamp范围了。

```
mysql> create table t12(
    -> field timestamp
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t12 values ('1975-5-5 12:12:12');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t12 values ('1969-5-5 12:12:12');   # 超出范围
ERROR 1292 (22007): Incorrect datetime value: '1969-5-5 12:12:12' for column
'field' at row 1
mysql> insert into t12 values ('2038-1-19 11:14:07');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t12 values ('2038-1-19 11:14:08');  # 超出范围
ERROR 1292 (22007): Incorrect datetime value: '2038-1-19 11:14:08' for column
'field' at row 1

mysql> select * from t12;
+---------------------+
| field               |
+---------------------+
| 1975-05-05 12:12:12 |
| 2038-01-19 11:14:07 |
+---------------------+
```

4、year

因为只占用1个字节，最多只能表示255个年份，范围是1901-2155之间的年份

```
mysql> create table t13(
    -> field year
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> insert into t13 values (2025);
Query OK, 1 row affected (0.00 sec)
```

```
mysql> insert into t13 values (1900);   # 超出范围
ERROR 1264 (22003): Out of range value for column 'field' at row 1
mysql> insert into t13 values (2155);
Query OK, 1 row affected (0.00 sec)

mysql> insert into t13 values (2156);   # 超出范围
ERROR 1264 (22003): Out of range value for column 'field' at row 1
```

5、time　表示时间或时间间隔，范围是-838:59:59~838:59:59

```
mysql> create table t14(
    -> field time
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t14 values ('12:12:12');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t14 values ('212:12:12');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t14 values ('838:59:59');
Query OK, 1 row affected (0.00 sec)

mysql> insert into t14 values ('839:00:00');  # 操作范围
ERROR 1292 (22007): Incorrect time value: '839:00:00' for column 'field' at row 1
mysql>
```

多学一招：time支持以天的方式插入

```
mysql> insert into t14 values ('10 10:10:10');
Query OK, 1 row affected (0.02 sec)

mysql> select * from t14;
+-----------+
| field     |
+-----------+
| 12:12:12  |
| 212:12:12 |
| 838:59:59 |
| 250:10:10 |
+-----------+
```

1.7 数据类型——boolean

MySQL不支持boolean类型，true和false在数据库中对应1和0。

```
mysql> create table t15(
    -> field boolean
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t15 values (true),(false);   # true和false在数据库中对应1和0
Query OK, 2 rows affected (0.00 sec)
```

```
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from t15;
+-------+
| field |
+-------+
|     1 |
|     0 |
+-------+
2 rows in set (0.00 sec)
```

1.8 关于数据类型的思考题

1. 手机号码一般使用什么数据类型存储?　　char
2. 电话号码使用什么数据类型　　varchar
3. 性别一般使用什么数据类型存储?　　char　enum
4. 学生年龄信息一般使用什么数据类型存储?　　tinyint
5. 照片信息一般使用什么数据类型存储?　　binary
6. 薪水一般使用什么数据类型存储?　　decimal

多学一招：一个字段到底选数字还是字符，取决于有没有计算的可能，如果没有计算的可能即使是数字也要用字符类型，比如手机号、QQ号，...

1.9 列属性——是否为空(null | not null)

null：可以为空

not null：不可以为空

思考题

1. 学员姓名允许为空吗?　　非空
2. 家庭地址允许为空吗?　　非空
3. 电子邮件信息允许为空吗?　　可以为空
4. 考试成绩允许为空吗?　　可以为空

1.10　列属性——默认值（default)

1、如果一个字段没有插入值，可以默认插入一个指定的值。

2、default关键字用来插入默认值

```
mysql> create table t16(
    -> id int unsigned,
    -> addr varchar(20) not null default '地址不详'
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> insert into t16 values (1,'北京'),(2,default);
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from t16;
```

```
+------+----------+
| id   | addr     |
+------+----------+
|    1 | 北京         |
|    2 | 地址不详        |
+------+----------+
2 rows in set (0.00 sec)
```

1.11 列属性——自动增长（auto_increment）

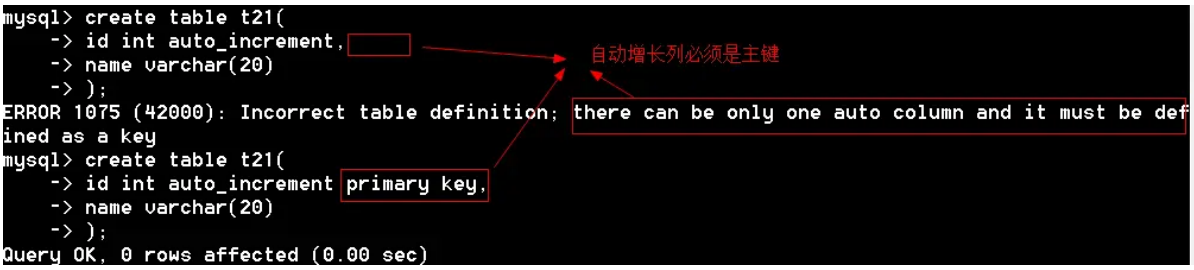
1、字段的值从1开始，每次递增1，特点就在字段中的数据不可能重复，适合为记录生成唯一的id

2、自动增长都是无符号整数。

3、在MySQL中，auto_increment必须是主键。但是主键不一定是自动增长的。

4、如果要给自动增长列插入数据，使用null关键字。

5、自动增长列上的数据被删除，默认情况下此记录的编号不再使用。



1.12 列属性——主键（primary key）

主键：唯一标识表中记录的一个或一组列

主键的特点：不能重复，不能为空

一个表只能有一个主键，主键可以有多个字段组成。

主键的作用：

1、 保证数据完整性

2、 加快查询速度

1.12.1 添加主键

方法一：创建表的时候添加主键

```
mysql> create table t17(
    -> id varchar(5) primary key,   # 创建主键
    -> name varchar(10) not null
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t17 values ('s2531','tom'),('s2532','berry');
Query OK, 2 rows affected (0.00 sec)
Records: 2  Duplicates: 0  Warnings: 0

mysql> select * from t17;
```

```
+-------+-------+
| id    | name  |
+-------+-------+
| s2531 | tom   |
| s2532 | berry |
+-------+-------+
2 rows in set (0.00 sec)

# 如果插入主键相同数据会报错
mysql> insert into t17 values ('s2531','tom');
ERROR 1062 (23000): Duplicate entry 's2531' for key 'PRIMARY'

# 主键不能插入null值
mysql> insert into t17 values (null,'tom');
ERROR 1048 (23000): Column 'id' cannot be null
```

方法二：创建表的时候添加主键

```
mysql> create table t18(
    -> id int,
    -> name varchar(10),
    -> primary key(id)
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> desc t18;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int(11)     | NO   | PRI | 0       |       |
| name  | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

方法三：更改表的时候添加主键

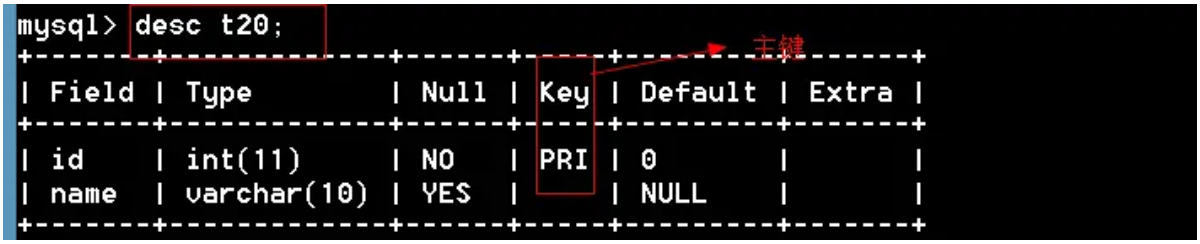
```
mysql> create table t20(
    -> id int,
    -> name varchar(10)
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> alter table t20 add primary key (id);   # 更改表添加主键
Query OK, 0 rows affected (0.08 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc t20;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int(11)     | NO   | PRI | 0       |       |
| name  | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```

1.12.2 创建组合键

```
mysql> create table t19(
    ->  classid int,
    ->  stuid int,
    ->  stuname varchar(10),
    ->  primary key(classid,stuid)
    -> );
Query OK, 0 rows affected (0.01 sec)

mysql> desc t19;
+---------+------------+------+-----+---------+-------+
| Field   | Type       | Null | Key | Default | Extra |
+---------+------------+------+-----+---------+-------+
| classid | int(11)    | NO   | PRI | 0       |       |
| stuid   | int(11)    | NO   | PRI | 0       |       |
| stuname | varchar(10) | YES  |     | NULL    |       |
+---------+------------+------+-----+---------+-------+
3 rows in set (0.00 sec)
```

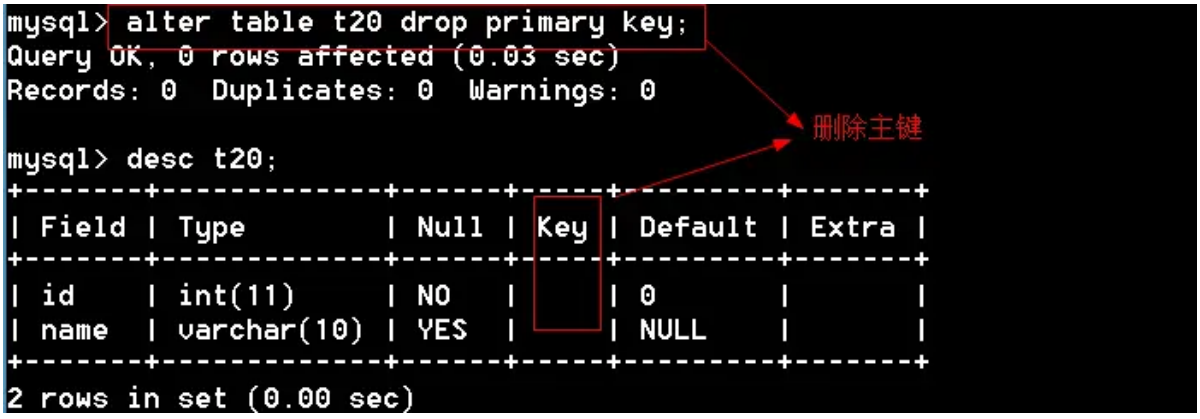
1.12.3 查看主键

```
mysql> desc t20;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int(11)     | NO   | PRI | 0       |       |
| name  | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
```



1.12.3 删除主键

```
mysql> alter table t20 drop primary key;
Query OK, 0 rows affected (0.03 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> desc t20;
+-------+-------------+------+-----+---------+-------+
| Field | Type        | Null | Key | Default | Extra |
+-------+-------------+------+-----+---------+-------+
| id    | int(11)     | NO   |     | 0       |       |
| name  | varchar(10) | YES  |     | NULL    |       |
+-------+-------------+------+-----+---------+-------+
2 rows in set (0.00 sec)
```



1.12.4 选择主键的原则

1、 最少性：尽量选择一个字段做主键

2、 稳定性：尽量选择更新少的列做主键

3、 尽量选择数字型的列做主键

1.12.5 主键思考题

1、在主键列输入的数值，允许为空吗？ 不可以

2、 一个表可以有多个主键吗？ 不可以

3、 在一个学校数据库中，如果一个学校内允许重名的学员，但是一个班级内不允许学员重名，可以组合班级和姓名两个字段一起来作为主键吗？ 可以

4、 标识列（自动增长列）允许为字符数据类型吗？　不可以

5、 表中没有合适的列作为主键怎么办？　添加自动增加列

6、 如果标识列A的初始值为1，增长量为1，则输入三行数据以后，再删除两行，下次再输入数据行的时候，标识值从多少开始？　从4开始

1.13　列属性——唯一键

特点：

1、不能重复，可以为空

2、一个表可以有多个唯一键

作用：

1、　保证数据不能重复。保证数据完整性

2、　加快数据访问

1.13.1 添加唯一键

方法一：创建表的时候添加唯一键

```
mysql> create table t22(
    -> id int primary key,
    -> name varchar(20) unique,    #通过unique添加唯一键
    -> addr varchar(100) unique
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into t22 values (1,'tom','上海');
Query OK, 1 row affected (0.05 sec)

mysql> insert into t22 values (2,'tom','北京');    #  name重复了，报错
ERROR 1062 (23000): Duplicate entry 'tom' for key 'name'
mysql> insert into t22 values (2,'berry','上海');   # addr重复了
ERROR 1062 (23000): Duplicate entry '上海' for key 'addr'
```

还有一种方法

```
mysql> create table t26(
    -> id int,
    -> name varchar(20),
    -> addr varchar(20),
    -> primary key(id),
    -> unique (name),     # 添加唯一键
    -> unique (addr)
    -> );
Query OK, 0 rows affected (0.06 sec)
```

方法二：修改表的时候添加唯一键

```
mysql> create table t23(
    -> id int primary key,
    -> name varchar(20)
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> alter table t23 add unique (name);     #  添加一个唯一键
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

一次添加多个唯一键

```
mysql> create table t24(
    -> id int primary key,
    -> name varchar(20),
    -> addr varchar(20)
    -> );
Query OK, 0 rows affected (0.06 sec)

mysql> alter table t24 add unique(name),add unique(addr);
Query OK, 0 rows affected (0.09 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

添加组合唯一键

```
mysql> create table t25(
    -> id int primary key,
    -> name varchar(20),
    -> addr varchar(20)
    -> );
Query OK, 0 rows affected (0.09 sec)

mysql> alter table t25 add unique(name,addr);
Query OK, 0 rows affected (0.01 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

1.13.2查看唯一键

```
mysql> show create table t26\G
*************************** 1. row ***************************
       Table: t26
Create Table: CREATE TABLE `t26` (
  `id` int(11) NOT NULL DEFAULT '0',
  `name` varchar(20) DEFAULT NULL,
  `addr` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`),      # 唯一键
  UNIQUE KEY `addr` (`addr`)       # 唯一键
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)

mysql> show create table t25\G
*************************** 1. row ***************************
```

```
       Table: t25
Create Table: CREATE TABLE `t25` (
  `id` int(11) NOT NULL,
  `name` varchar(20) DEFAULT NULL,
  `addr` varchar(20) DEFAULT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `name` (`name`,`addr`)     # 组合唯一键
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)
```

添加唯一键，给唯一键取名

```
mysql> create table t27(
    -> name varchar(20)
    -> );
Query OK, 0 rows affected (0.03 sec)

mysql> alter table t27 add unique UQ_name(name);
Query OK, 0 rows affected (0.00 sec)
Records: 0  Duplicates: 0  Warnings: 0

mysql> show create table t27\G
*************************** 1. row ***************************
       Table: t27
Create Table: CREATE TABLE `t27` (
  `name` varchar(20) DEFAULT NULL,
  UNIQUE KEY `UQ_name` (`name`)     # 唯一键的名字是UQ_name
) ENGINE=InnoDB DEFAULT CHARSET=utf8
1 row in set (0.00 sec)
```

1.13.3 删除唯一键

通过唯一键的名字来删除唯一键

```
语法：alter table 表名 drop index 唯一键名称
```

问题：主键和唯一键的区别?

1、主键不能重复，不能为空，唯一键不能重复，可以为空

2、主键只有一个，唯一键可以有多个。

1.14列属性——备注（comment）

为了程序员之间的相互交流

```
mysql> create table t28(
    -> id int primary key comment '主键',
    -> name varchar(20) comment '姓名'
    -> );
Query OK, 0 rows affected (0.00 sec)
```

备注

1.15　SQL注释

单行注释：--或#

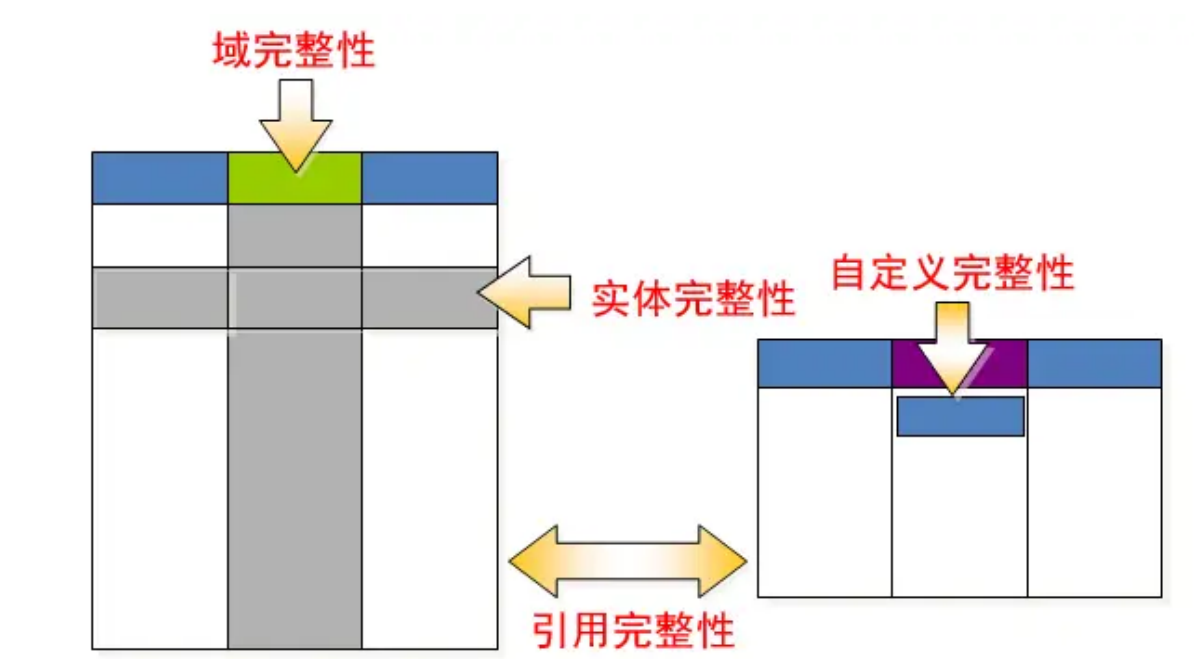
多行注释：/*　*/

```
mysql> create table t29(
    -> id int,     # 这是主键
    -> name varchar(20)   -- 这是姓名
    -> /*
    /*> 这是一段注释
    /*> */
    -> );
Query OK, 0 rows affected (0.00 sec)
```

单行注释

多行注释

1.16　数据完整性介绍



1.16.1 保证实体完整性

1、　主键约束

2、　唯一约束

3、　自动增长列

1.16.2 保证域完整性

1、 数据类型约束

2、 非空约束

3、 默认值约束

1.16.3 保证引用完整性

1、外键约束：从表中的公共字段是主表的外键

1.17 引用完整性

1.17.1 主表和从表

两个表建立关系（两个表只要有公共字段就有关系），一个表称为主表，一个表称为从表。

外键约束可以实现：

1、 主表中没有的从表中不允许插入

2、 从表中有的主表中不允许删除

3、 不能更改主表中的值而导致从表中的记录孤立存在。

4、 先删除从表，再删除主表

1.17.2 外键（foreign key）

1、 外键：从表中的公共字段，公共字段的名字可以不一样，但是数据类型必须一样。

2、 外键约束用来保证引用完整性

1.17.3 添加外键

方法一：创建表的时候添加外键

```
create table stuinfo(
    stuno char(4) primary key,
    name varchar(10) not null
);

create table stumarks(
    stuid char(4) primary key,
    score tinyint unsigned,
    foreign key (stuid) references stuinfo(stuno)
);
```

方法二：修改表的时候添加外键

```
mysql> create table stuinfo(
    ->  stuno char(4) primary key,
    ->  name varchar(10) not null
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> create table stumarks(
```

```
    ->  stuid char(4) primary key,
    ->  score tinyint unsigned
    -> );
Query OK, 0 rows affected (0.06 sec)

语法：  alter table 从表 add foreign key (从表的公共字段) references 主表(公共字段)

mysql> alter table stumarks add foreign key (stuid) references stuinfo(stuno);
Query OK, 0 rows affected (0.06 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

脚下留心：要创建外键必须是innodb引擎，myisam不支持外键约束

1.17.4 查看外键

1.17.5 删除外键

通过外键的名字删除外键

```
语法：alter table 表名 drop foreign key 外键名
```

例题

```
mysql> alter table stumarks drop foreign key stumarks_ibfk_1;
Query OK, 0 rows affected (0.02 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

。

1.18 外键操作

1、 严格操作（前面讲的是严格操作）

2、 置空操作（set null）：如果主表记录删除或更新，从表置空

3、 级联操作（cascade）：如果主表记录删除或更新，从表级联

一般来说：主表删除的时候，从表置空操作，主表更新的时候，从表级联操作。

```
语法：foreign key(外键) references 主表(关键字段)[主表删除是的动作][主表更新时候的动作]
```

例题

```
mysql> create table stuinfo(
```

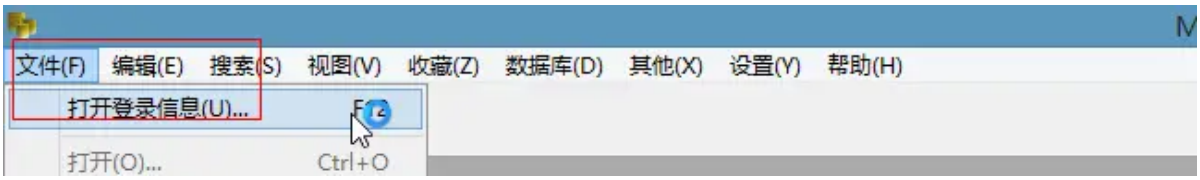
```
    -> stuno char(4) primary key,
    -> name varchar(10) not null
    -> );
Query OK, 0 rows affected (0.02 sec)

mysql> create table stumarks(
    -> stuid int auto_increment primary key,
    -> stuno char(4) ,
    -> score tinyint unsigned,
    ->  foreign key (stuno) references stuinfo(stuno) on delete set null on
update cascade
    -> );
Query OK, 0 rows affected (0.00 sec)

mysql> insert into stuinfo values ('s101','tom');
Query OK, 1 row affected (0.00 sec)

mysql> insert into stumarks values (null,'s101',88);
Query OK, 1 row affected (0.00 sec)

mysql> select * from stuinfo;
+-------+------+
| stuno | name |
+-------+------+
| s101  | tom  |
+-------+------+
1 row in set (0.00 sec)

mysql> update stuinfo set stuno='s102' where stuno='s101';   # 更新时级联
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0

mysql> select * from stumarks;
+-------+-------+-------+
| stuid | stuno | score |
+-------+-------+-------+
|     1 | s102  |    88 |
+-------+-------+-------+
1 row in set (0.00 sec)

mysql> delete from stuinfo where stuno='s102';    # 删除时置空
Query OK, 1 row affected (0.02 sec)

mysql> select * from stumarks;
+-------+-------+-------+
| stuid | stuno | score |
+-------+-------+-------+
|     1 | NULL  |    88 |
+-------+-------+-------+
1 row in set (0.00 sec)
```

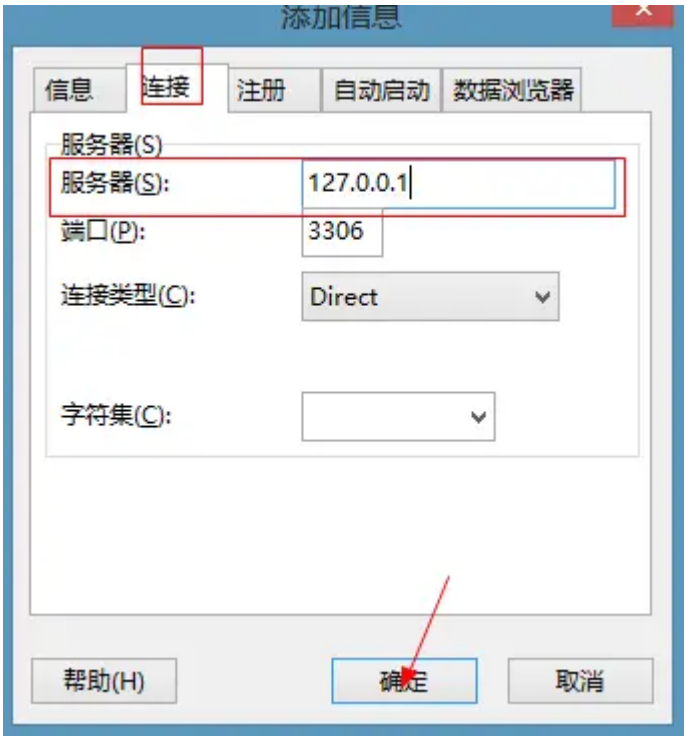
1.19客户端介绍

第一：命令行

第二：MySQL-Front和Navicat

MySQL-Front






数据库登录
信息
用户(U):
root
密码(P):
****
记住密码(S)
编辑(E)...
确定
取消