**DBA1—day02**

**字段约束条件：控制表结构**

**约束条件**

**• Null 允许为空,默认设置**

**• NOT NULL 不允许为空**

**• Key 索引类型**

**• Default 设置默认值,缺省为 NULL**

**mysql> create table t10(**

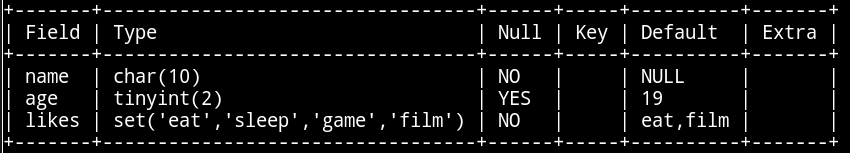
**-> name char(10) not null,**

**-> age tinyint(2) default 19,**

**-> likes set("eat","sleep","game","film") not null default "eat,film"**

**-> );**

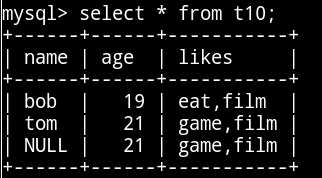
**mysql> desc t10;**

****

**mysql> insert into t10(name)values("bob");**

**mysql> insert into t10 values("tom",21,"game,film");**

**mysql> insert into t10 values("NULL",21,"game,film");**

****

**修改表结构**

**语法结构**

**\_ALTER TABLE 表名 执行动作**

**可执行动作：**

**Add 添加字段**

**Modify 修改字段类型**

**Change 修改字段名**

**Drop 删除字段**

**Rename 修改表名**

**添加新字段：**

**基本用法**

**– ALTER TABLE 表名**

**– ADD 字段名 类型 ( 宽度 ) 约束条件 ;**

**可加 AFTER 字段名 ;**

**或者 FIRST**

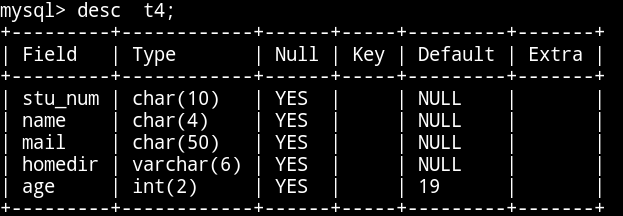
**mysql> alter table teadb.t4 add age int(2) default 19; //添加一个新字段**

**mysql> alter table t4 add mail char(50) after name;**

**//在name表字段之后添加新字段mail**

**mysql> alter table t4 add stu\_num char(10) first;**

**//在首行添加一个新字段**

****

**修改字段类型：**

**基本用法**

**– ALTER TABLE 表名**

**– modify 字段名 类型 ( 宽度 ) 约束条件 ;**

**可加 AFTER 字段名 ;**

**或者 FIRST;**

**mysql> alter table t4 modify name varchar(15); //修改字段类型**

**mysql> alter table t4 modify age int(2) default 19 after name;**

**//修改字段位置**

**修改字段名**

**基本用法**

**– ALTER TABLE 表名**

**– change 源字段名 新字段名 类型 ( 宽度 ) 约束条件**

**mysql> alter table t4 change stu\_num stu\_id char(10);**

**删除字段**

**基本用法**

**– ALTER TABLE 表名**

**– drop 字段名 ;**

**mysql> alter table t5 drop start;**

**修改表名**

**基本用法**

**– ALTER TABLE 表名**

**– Rename 新表名;**

**mysql> alter table t10 rename stuinfo; //修改表名**

**MySQL键值**

**INDEX : 普通索引**

**UNIQUE : 唯一索引**

**FULLTEXT : 全文索引**

**PRIMARY KEY : 主键**

**FOREIGN KEY : 外键**

**索引**

**– 索引是对记录集的多个字段进行排序的方法。**

**– 类似于书的目录**

**– 索引类型包括 :Btree 、 B+tree 、 hash**

**（二叉树）**

**索引优缺点**

**• 索引优点**

**– 通过创建唯一性索引,可以保证数据库表中每一行数**

**据的唯一性**

**– 可以加快数据的检索速度**

**• 索引缺点**

**– 当对表中的数据进行增加、删除和修改的时候,索引**

**也要动态的维护,降低了数据的维护速度**

**– 索引需要占物理空间**

**INDEX 普通索引**

**– 一个表中可以有多个 INDEX 字段**

**– 字段的值允许有重复,切可以赋 NULL 值**

**– 经常把做查询条件的字段设置为 INDEX 字段**

**– INDEX 字段的 KEY 标志是 MUL**

**mysql> show index from t4; //查看表是否有索引**

**建表的时候指定索引字段**

**– INDEX( 字段 1), INDEX( 字段 2) .. ..**

**创建索引：**

**mysql> create table t21(**

**-> name char(10),**

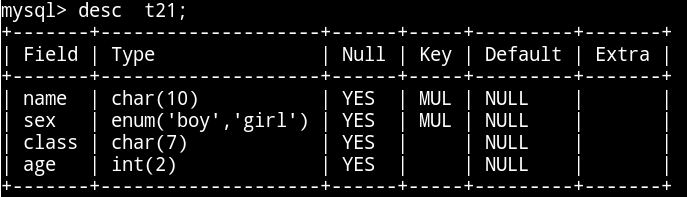
**-> sex enum("boy","girl"),**

**-> class char(7),**

**-> age int(2),**

**-> index(name),index(sex)**

**-> );**

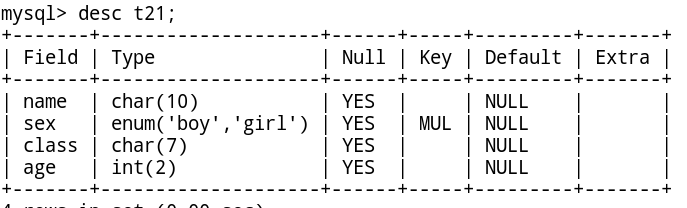


**查看索引的详细信息**

**mysql> show index from t21\G;**

**删除索引：**

**mysql> drop index name on t21;**



**在已有的表中加入索引：（设置index索引）**

**mysql> create index name on t4(name);**

**primary key 主键**

**• 注意事项**

**– 一个表中只能有一个 primary key 字段**

**– 对应的字段值不允许有重复,且不允许赋 NULL 值**

**– 如果有多个字段都作为 PRIMARY KEY ,称为复合主**

**键,必须一起创建。**

**– 主键字段的 KEY 标志是 PRI**

**– 通常与 AUTO\_INCREMENT 连用**

**– 经常把表中能够唯一标识记录的字段设置为主键字段**

**[ 记录编号字段 ]**

**建表的时候指定主键字段**

**– PRIMARY KEY( 字段名 )**

1. **格式一**

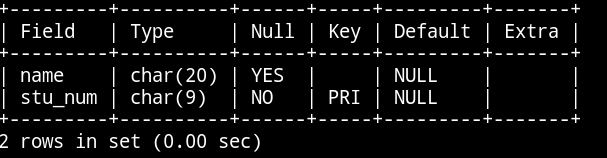
**mysql> create table t22(**

**-> name char(20),**

**-> stu\_num char(9),**

**-> primary key(stu\_num)**

**-> );**



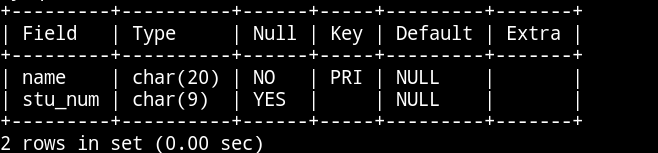
1. **格式二**

**mysql> create table t23(**

**-> name char(20) primary key,**

**-> stu\_num char(9)**

**-> );**



在已有的表中设置PRIMARY KEY字段

– ALTER TABLE 表名 ADD PRIMARY KEY( 字段名 );

移除表中的PRIMARY KEY字段

– ALTER TABLE 表名 DROP PRIMARY KEY;

**复合主键（表中多个字段一起做主键）**

mysql> create table t24(

-> cip char(15),

-> sport smallint(2),

-> status enum("deny","allow"),

-> primary key(cip,sport)

-> );

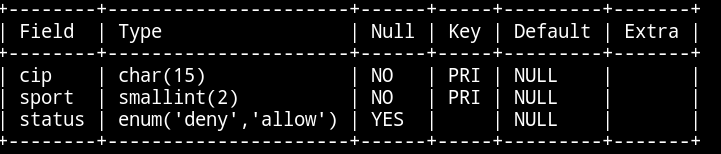
mysql> insert into t24 values("192.168.4.1",21,"deny");

mysql> insert into t24 values("192.168.4.1",22,"allow");

mysql> insert into t24 values("192.168.4.2",21,"deny");

mysql> insert into t24 values("192.168.4.1",21,"deny");

mysql> select \* from t24;



删除主键

mysql> alter table t24 drop primary key;

**primary key 与 AUTO\_INCAREMENT连用**

要求：主键不能重复,也不能赋空值

需求：在不给值的时候用自加1的方式赋值

mysql> create table t26(

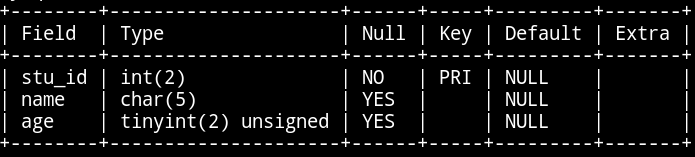
-> stu\_id int(2) primary key auto\_increment,

-> name char(5),

-> age tinyint(2) unsigned

-> );

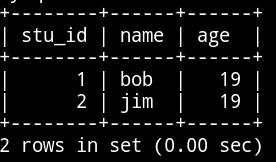
mysql> desc t26;



mysql> insert into t26(name,age)values("bob",19);

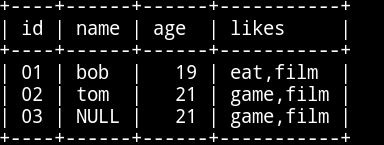
mysql> insert into t26(name,age)values("jim",19);

mysql> select \* from t26;



//当把表记录删除，重新写入表记录时，会从之前记录的数值开始计数

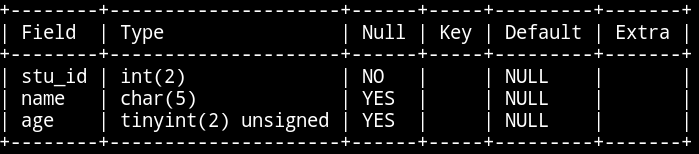
mysql> alter table stuinfo add id int(2) zerofill primary key auto\_increment first; //在原有的表中加入一列且数值自增长(有仅有一个主键)



此时删除主键须先把自赠长删除

mysql> alter table t26 modify stu\_id int(2) not null; //删除自增长

mysql> alter table t26 drop primary key; //再删除主键



**foreign key 外键**

外键概述：

– 让当前表字段的值在另一个表中字段值的范围内选择

使用外键的条件

– 表的存储引擎必须是 innodb

– 字段类型要一致

– 被参照字段必须要是索引类型的一种 (primary key)

基本用法

– FOREIGN KEY( 表 A 的字段名 )

References 表 B( 字段名 )

ON UPDATE CASCADE //同步更新功能

ON DELETE CASCADE //同步删除

表A——ygtab

mysql> create table ygtab(

-> yg\_id int(2) primary key auto\_increment,

-> name char(10)

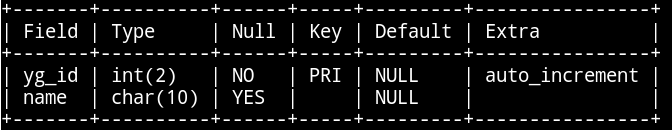
-> )engine=innodb;

mysql> insert into ygtab(name)values("bob");

mysql> insert into ygtab(name)values("jerry");

mysql> insert into ygtab(name)values("tom");

mysql> desc ygtab;



表B——gztab

mysql> create table gztab(

-> gz\_id int(2),

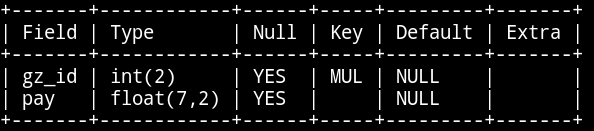
-> pay float(7,2),

-> foreign key(gz\_id) references ygtab(yg\_id)

-> on update cascade on delete cascade

-> )engine=innodb;

mysql> desc gztab;



mysql> show create table gztab; //查看是否有外键

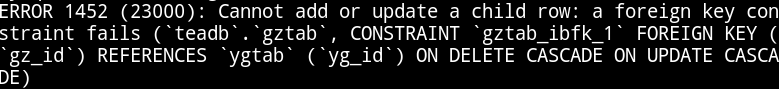
给员工工资表中插入信息，会受到员工信息表的约束

mysql> insert into gztab values(1,15600);

mysql> insert into gztab values (2,16000);

以上两条会插入成功

mysql> insert into gztab values (4,20000);



//会有以上报错因为在员工信息表中没有编号为4的信息

测试同步更新功能

在员工工资表中将员工编号2改为8；工资表中同步更改则为成功

mysql> update ygtab set yg\_id=8 where yg\_id=2;

分别查看员工信息表与工资表

mysql> select \* from ygtab;

mysql> select \* from gztab;

测试同步删除功能

在员工工资表中将员工编号为8的表信息删除，工资表中同步删除则为成功

mysql> delete from ygtab where yg\_id=8;

分别查看员工信息表与工资表

mysql> select \* from ygtab;

mysql> select \* from gztab;

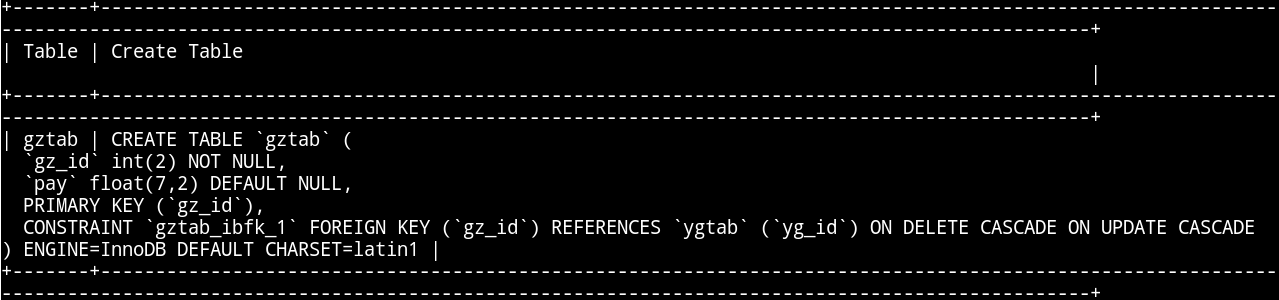
防止工资发重复，解决办法：给工资表的员工编号列加上主键值防止重复，防止空值

mysql> alter table gztab add primary key(gz\_id);

生产环境中在删除员工表记录时，须慎重！！！

删除外键：

mysql> show create table gztab; //查询约束名



mysql> alter table gztab drop foreign key gztab\_ibfk\_1; //删除外键