MySQL中的数据完整性约束(重点)

# 完整性约束

已经建好的表，字段的数据类型决定了**所能存储的数据类型**，但是表中所存储的**数据是否合法**并没有检查。

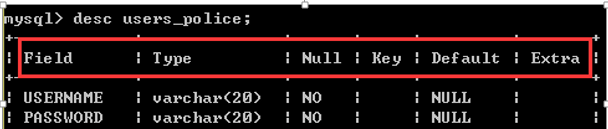
数据的完整性是指**数据的准确性和一致性**，而数据的**完整性检查**就是指**检查数据的准确性和一致性**。

MySQL数据库管理系统除了支持**标准的SQL的完整性约束**外，还进行了相应扩展。MySQL数据库扩展增添了**AUTO\_INCREMENT**约束，MySQL中独有的一个扩展约束。

这篇笔记主要是讲述了**六个约束**的设置方法：**主键、外键、唯一键、非空、默认值、自增**。

# 表的结构属性：

## 表的结构：desc tbName;



主要有：**Field、Type、Null、key、Default、Extra。**

分别为：**字段名称、数据类型**、能否为空、键、默认值、其他。

从表的结构可以看出，约束一共是4栏，分为**4类约束**：

* 1. **非空约束**：**NOT NULL**
  2. **默认值约束**：**DEFAULT**
  3. **KEY约束**：包括**UNIQUE [KEY]**(UK)、**PRIMARY** KEY(PK)、**FOREIGN** KEY(FK)。
  4. **其他**：MySQL单独提供自增约束：AUTO\_INCREMENT

完整性约束的关键字及含义：（6个）

A. **NOT NULL** ，约束该字段的值不能为空；

B. **DEFAULT** ，设置字段的默认值；

C. **UNIQUE（KEY ，UK）**，约束所有记录中该字段的值是唯一的，不重复的；

D. **PRIMARY KEY(PK)**，约束该字段为表的主键，可以作为该表记录的唯一标识；设置主键约束时，同时设置主键字段的值是唯一的（UNIQUE）和非空的（NOT NULL）。主键可以是单个字段，也可以是多个字段。

E. **FROEIGN KEY(FK)**，约束字段为表的外键；（查询时显示MUL）

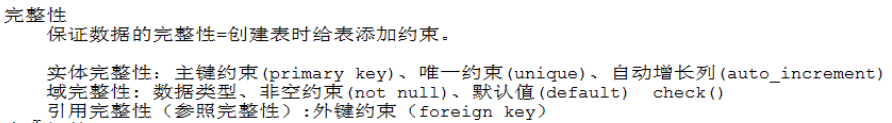
F. **AUTO\_INCREMENT**，约束字段的值为自动增加，这是MySQL自带的，在**Extra列**列出。

# 数据完整性

**确保数据的完整性**就是**在创建表的时候，给表添加约束**。

数据完整性的作用：保证用户输入的数据保存到数据库中是正确的。

**完整性的分类**： ①体完整性；②完整性；③用完整性



# 实体完整性

## 实体：表中的一行记录，代表一个实体（Entity），即就是一行。

## 实体完整性的作用：所有记录的该行数据不重复。

## 约束类型：

## A. 唯一键约束（Unique）（在KEY列显示为UNI）

1. 创建表时，设置**唯一约束**：

create table tbName(fieldName type **UNIQUE**,...);

create table tbName(field1 type,field2 type,....,primary key(field),**unique(field2)**)

②创建表后，修改某字段为非空约束：

alter table tbName **modify** fieldName type **UNIQUE**;

alter table tbName **change** fieldName fieldName type **UNIQUE**;

### 主键约束（Primary Key）（在KEY列显示为PRI）

一般情况下，每一个表都要有一个且仅有一个**主键**。主键可以为**单字段主键**，也可以为**多字段主键（也叫联合主键）**。

**单字段主键**：只有一个字段作为主键。

**多字段主键（联合主键）**：两个或两个以上的字段一起作为**主键**。

举个简单的例子：1班和2班都有两个叫赵西元的学生，且学号都是11和12；

则1班11号、1班12号、2班11号、2班12号都不是同一个人，即不是同一条记录。只有当两者都一样时，才认为是同一条记录。

主键约束的特点：**①数据唯一UNIQUE；②且不能为空（NOT NULL）。**

**主键**的建立方法：

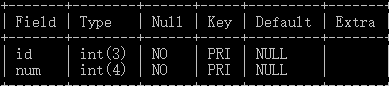
方法1和方法2都是**创建表格时**，设置主键约束。

方法1：create table students(id int(10) **PRIMARY KEY** ,name varchar(10));

在字段类型后紧加上primary key即可，只适用于**设置单字段主键**。

方法2：create table students(id int(10),name varchar(10),...，**[constraint conName]** **PRIMARY KEY(id,name,...)**); 这种方法既可以定义单字段主键，也可以用于定义**多字段主键（联合主键）**。

**create table** stud2(**classID** int(10),**studID** int(10),name varchar(10),**primary key(classID,studID)**);



方法3是对建立好的表格**增添约束**。

第3种方法：先简单的创建表格，再利用**add constraint** 或者**alter+modify**进行修改即可。

**alter table** students **modify** id int(10）**PRIMARY KEY;**

**或者alter table** students **add constraint** pk\_stu\_id **PRIMARY KEY(id);**

**constraint :约束。**

**注：**pk\_stu\_id 为约束的名字，随意起个名字即可。

### 自动增长约束AUTO\_INCREMENT（auto\_increment）

AUTO\_INCREMENT是MySQL唯一扩展的完整性约束。在一个表中，只能有一个字段使用自增约束且该字段必须是primary key主键，该字段的数据类型必须为**整数**类型。如果不是主键PK约束，设置AUTO\_INCREMENT会报出错误：

ERROR 1075 (42000): Incorrect table definition; there can be only one auto column and it must be defined as a key

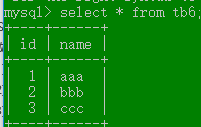
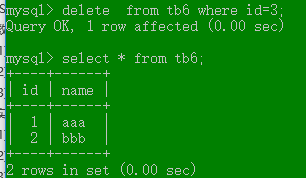
一般在创建表格时，主键与自增一起设置：

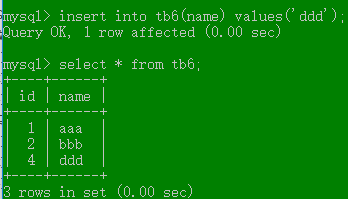
**create table** tbName(**id int primary key auto\_increment**,name varchar(10),...);

修改为主键自增：

**alter table** tbName **modify** id int primary key auto\_increment;

注意：即使删除了id号是最后一个的记录，再添加新的记录时，id号将会按照原来的数值加1，不会是被删除记录的id号。

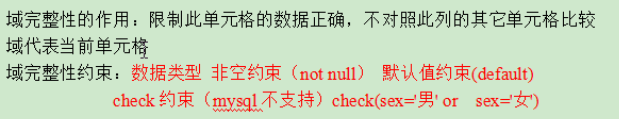
 



# 域完整性

**域完整性**的作用：只限制该单元格的数据，不对此列的其他单元格进行比较。

**域：代表的就是当前单元格。**



域完整性约束：

## 数据类型：数据类型就是一种域完整性约束，字符串型的数据不能插入到数值型的单元格中；

数据类型主要是这三大类：



## 非空约束：NOT NULL。

* + - 1. 创建表时，设置**非空约束**：（**在NULL列显示为NO**）

**create table** tbName(fieldName type **NOT NULL**,...);

②创建表后，修改某字段为非空约束：

**alter table** tbName **modify** fieldName type **NOT NULL**;

**alter table** tbName **change** fieldName fieldName type **NOT NULL**;

## 默认值约束：default。

如果想要某个字段设置为**默认值**，可以直接**不向该字段插入值或者插入DEFAULT**；但是，不能插入NULL。若插入NULL，则该字段就会变成了**NULL**。

**只有主键自增的时候，可以向主键插入NULL，此时主键会自增，不会是NULL，因为主键本身不允许NULL，这是MySQL支持的，其他数据库不一定支持。**

创建表时设置默认值约束：

create table tbName(id int,name varchar(5) **default value**);

修改表的字段默认值约束：

**alter table** tbName modify name varchar(5) **default newValue**;

示例：

**create table** university(id int ***primary key auto\_increment***,name varchar(10) **default '北邮'**);

**insert into** university **values(null,default),(null,null),(null,'清华')**;



**alter table** university modify name varchar(10) **default '北大'**;#默认值为北大

## check约束：MySQL不支持。

check（sex=‘男’or sex=‘女’）

# 引用完整性（参照完整性）（外键）(外键在KEY列中显示MUL)

注意：必须有一列是**主键列**。

**外键的作用**：在**表1**中设置一个属性为外键，作为外键的这个属性与**表2中的主键**相关联，在表1中插入数据时，作为外键的这个属性值必须在表格2中存在，否则插入不成功。报错**ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails。**

打个比方：假如表1是成绩表，包括学号、各科成绩等，表2是参加考试名单，包括学号、姓名等信息，只有在表2中有这个学生的学号，才可以在表1中插入成绩。即只有学生参加了考试，才可以有成绩；不可能学生没有考试，就会有成绩之说法。

## 第一种方法：在创建的时候添加外键约束。

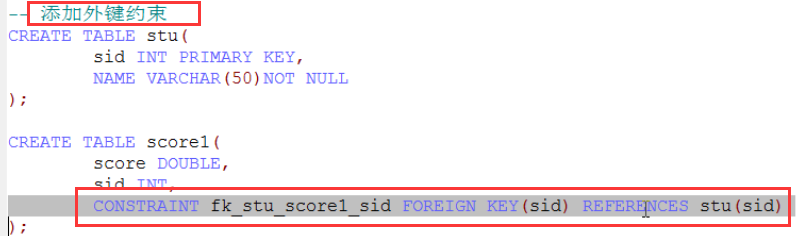
**添加外键约束**：

创建考试名单表：学号和姓名(学号为主键)

**create table** nameList(id int **primary key**,name varchar(10));

创建成绩表：学号为外键

**create table** scores(id int,score int,**[constraint cons\_fk] foreign key(id) references nameList(id)**);



## 第二种方法：先创建一个简单的表格，再单独添加外键约束。

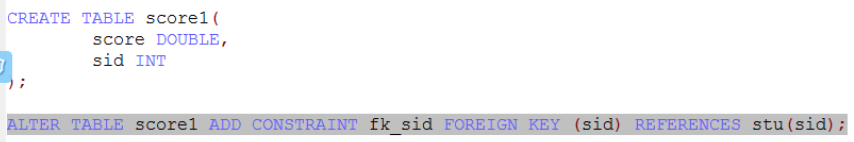
创建考试名单表：学号和姓名(学号为主键)

**create table** nameList(id int **primary key**,name varchar(10));

创建成绩表：create table scores(id int,score int);

添加外键约束：

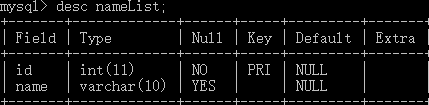
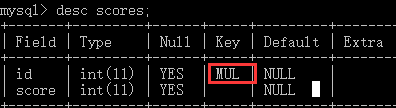
**alter table** scores **add constraint** cons\_fk **foreign key(id) references nameList(id)**);



示例：

**create table** nameList(id int **primary key**,name varchar(10));

**create table** scores(id int,score int,**[constraint cons\_fk] foreign key(id) references nameList(id)**);

**insert into** nameList values(02,'Jane'),(10,'Mike'),(26,'KangKang'),(08,'Maria'); #Query OK

**insert into** scores values(02,90),(08,99);#Query OK

**insert into** scores values(26,90),(11,99);#ERROR 1452 (23000): Cannot add or update a child row: a foreign key constraint fails

因为在关联的nameList表中没有id为11的学生，所以插入错误。

**insert into** nameList values(11,'Jerre');# Query OK

**insert into** scores values(26,90),(11,99); # Query OK

