完整性约束：

约束条件与数据类型的宽度一样，都是可选参数

作用：

用来保证数据的完整性和一致性，就是保证数据的正确

主要分为：

|  |  |
| --- | --- |
| 约束 | 作用 |
| PRIMARY KEY | 主键 |
| FOREIGN KEY | 外键(一定涉及到两个表) |
| NOT NULL | 非空 |
| UNIQUE KEY | 唯一 |
| AUTO\_INCREMENT | 自增 |
| DEFAULT | 默认 |

[案例]：

现在数据库有3个表要建立，分别是：请根据情况设计约束，并实现

student（sid，sname，sex)  （学号，姓名，性别）

|  |  |  |
| --- | --- | --- |
| sid | sname | sex |
| 202001 | 张三 | 男 |
| 202002 | 李四 | 男 |
| 202003 | 王五 | 男 |

course(cid,cname)  （课程号，课程名）

|  |  |
| --- | --- |
| cid | cname |
| C01 | 数据库 |
| C02 | C语言 |

grade(sid,cid,grade)

|  |  |  |
| --- | --- | --- |
| sid | cid | grade |
| 202001 | C01 | 80 |
| 202001 | C02 | 20 |
| 202002 | C02 | 30 |

1.主键约束

主键，也称主码，用于标识表中唯一的一条记录。**一张表中只能有一个主键（在一个表的创建语句中，只能出现一次主表关键字）**，并且主键值不能为空。

主键特点：1.非空 2.唯一

主键所在的列，一般多用于连接

主键约束是最常用的一种约束，设置主键约束的关键字为PRIMARY KEY。

主键又分为定义字段时直接设置和定义所有字段后设置主键约束。

定义字段时设置主键约束：

col\_name 数据类型 primary key

定义所有字段后设置主键约束

PRIMARY KEY (col\_name1，col\_name2)

2.自增约束

在向数据表中插入数据时，如果用户希望每条记录的“编号”自动生成，并且按顺序排列，可以为该字段设置自增约束。

设置自增约束的关键字为AUTO\_INCREMENT，语法形式如下：

col\_name 数据类型 AUTO\_INCREMENT

* 一张表中只能设置一个字段为自增约束，并且该字段必须为主键。
* 每增加一条记录，字段值自动增加1。
* 字段类型必须为整数型。

3.非空约束

设置非空约束的关键字为NOT NULL，作用是规定字段的值不能为空，用户在向数据表中插入数据时，如果设置非空约束的字段没有指定值，系统就会报错。

语法形式如下：

col\_name data\_type NOT NULL

4.唯一性约束

设置唯一性约束的关键字为UNIQUE，作用是规定该字段的值不能重复

定义字段时设置唯一性约束：

col\_name data\_type UNIQUE

定义所有字段后设置唯一性约束：

UNIQUE KEY(col\_name)

5.默认约束

设置默认约束的关键字为DEFAULT，当该字段为空时，自动填入默认值，语法形式如下：

col\_name data\_type DEFAULT value

6.外键约束

设置外键约束的主要作用是保证数据的完整性。 外键不是所属数据表的主键，但会对应着另外一张数据表的主键。

如果表A的主关键字是表B中的字段，则该字段称为表B的外键，表A称为主表，表B称为从表

设置外键约束的语法形式如下：

FOREIGN KEY(child\_col\_name) REFERENCES parent\_table\_name(parent\_col\_name)

[ON DELETE reference\_option]

[ON UPDATE reference\_option]

child\_col\_name表示从表中需要设置外键约束的字段名，parent\_table\_name表示主表名，parent\_col\_name表示主表中主键的字段名。

注意：

* 设置外键约束的字段参照的一定是另一个表的主键字段
* 设置外键约束的字段和关联的主键必须具有相同的数据类型。

级联外键的作用：

当主表中主键数据修改时，如果设置了级联更新或者级联删除，从表中数据会对应更新和删除

reference\_option:

RESTRICT | CASCADE | SET NULL | NO ACTION

1. CASCADE: 从父表中删除或更新对应的行，同时自动的删除或更新自表中匹配的行。ON DELETE CANSCADE和ON UPDATE CANSCADE都被InnoDB所支持。

2. SET NULL: 从父表中删除或更新对应的行，同时将子表中的外键列设为空。注意，这些在外键列没有被设为NOT NULL时才有效。ON DELETE SET NULL和ON UPDATE SET SET NULL都被InnoDB所支持。

3. NO ACTION: InnoDB拒绝删除或者更新父表。

4. RESTRICT: 拒绝删除或者更新父表。指定RESTRICT（或者NO ACTION）和忽略ON DELETE或者ON UPDATE选项的效果是一样的。

判断以上三个表每一个表中有哪些约束，并建立这三个表

建立的三个表：

CREATE DATABASE test;

USE test;

CREATE TABLE tb\_student(

sid INT(6) primary key auto\_increment, #设置自增约束 #在定义列时建立主键

sname VARCHAR(2) not null , #非空

sex SET('male','fmale') default 'male' #定义默认值

);

CREATE TABLE tb\_course(

cid VARCHAR(3),

cname VARCHAR(3) unique #列级唯一约束

);

CREATE TABLE tb\_course(

cid VARCHAR(3) primary key,

cname VARCHAR(3),

unique(cname) #表级的唯一约束

);

CREATE TABLE tb\_grade(

sid INT(6),

cid VARCHAR(3),

grade int(2),

#定义联合主键，需要在所有列定义完毕后去定义

primary key(sid,cid),

foreign key (cid) REFERENCES tb\_course(cid), #外键建立

foreign key (sid) REFERENCES tb\_student(sid) on update CASCADE #建立外键的同时级联修改

);

二 数据表的修改，删除和查看：

1.数据表查看

方法一：show create table 表名

方法二：desc 表名

2.数据表修改

MySQL提供了ALTER关键字来修改表结构

2.1 修改表名

ALTER TABLE old\_table\_name RENAME new\_table\_name;

[例2.1.1] 修改student的表名为stu

Alter table student rename stu;

2.2 修改字段数据类型/部分约束(可增加或删除 非空/自增约束）

修改字段数据类型的关键字为MODIFY，语法形式如下：

ALTER TABLE 数据表名 MODIFY [COLUMN] 字段名 新类型 [字段属性];

[例2.2.1] 将course表中的cname数据类型改为char(5)，并且该列非空

Alter table tb\_course modify cname char(5) not null;

2.3 修改字段名

修改数据表字段名称的关键字为CHANGE，语法形式如下：

ALTER TABLE 数据表名 CHANGE [COLUMN] 旧字段名 新字段名 字段类型 [字段属性];

* “旧字段名”指的是字段修改前的名称。
* “新字段名”指的是字段修改后的名称。
* “数据类型”表示新字段名的数据类型，不能为空，即使与旧字段的数据类型相同，也必须重新设置。

[例2.3.1] 将course表中的cname改名为cn，且数据类型转化为varchar(10)

Alter table course change cname cn varchar(10);

2.4 添加字段 （可增加 主键/外键/唯一约束)

常见添加字段的操作一般分为三种情况：在表的最后一列，在表的第一列或者在指定列之后。

在表的最后一列添加字段

# 语法格式1：

新增一个字段，并可指定其位置

ALTER TABLE 数据表名 ADD [COLUMN] 新字段名 字段类型 [FIRST | AFTER 字段名];

# 语法格式2：同时新增多个字段 ALTER TABLE 数据表名 ADD [COLUMN] (新字段名1 字段类型1, 新字段名2 字段类型2, ...);

* 在不指定位置的情况下，新增的字段默认添加到表的最后。
* 新增多个字段时不能指定字段的位置

[例2.4.1]在tb\_course表增加一个credit（学分)列，放在课程号和课程名之间

Alter table tb\_course add credit int after cid;

删除主键：

Alter table tb\_grade drop primary key;

[例2.4.2]增加grade表的主键。

Alter table tb\_grade add primary key(sid,cid);

[例2.4.3]三个表中存在几个外键关系？每个外键应当建立在哪个表之上？请在已有的表上增加外键。

foreign key (cid) REFERENCES tb\_course(cid), #外键建立

将普通外键修改为级联修改级联更新外键：

foreign key (sid) REFERENCES tb\_student(sid) on update CASCADE #建立外键的同时级联修改

1.5 删除字段

语法：

ALTER TABLE 数据表名 DROP [COLUMN] 字段名;

[例1.5.1] 删除course表中的cid，可以成功吗？

能成功！

alter table tb\_course drop cid

[例1.5.2]删除grade表上的外键

alter table tb\_grade drop FOREIGN key tb\_grade\_ibfk\_2 #外键名

实验：

1在上周sell数据库建立好的四个表的基础上，判断每个表或者表和表之间存在的约束，请把约束列填写完整，并通过编写sql代码将约束添加到表结构中。

顾客表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 | 约束 |
| Cid | int(10) | 顾客号 | 例如：20200101（固定长度） | 例：主键，自增 |
| Cname | varchar(10) | 顾客名 | 例如：张三 | 非空 |
| Cphone | char(11) | 电话 | 例如：18482112222 | 非空 |
| VIP | set('食','日','家') | 会员 | 目前会员只有”食品会员“，”日用品会员“，”家居会员"三种，可以同时拥有0-3个会员身份 |  |

建表语句

create table customer(

Cid int comment '顾客号' primary key auto\_increment,

Cname varchar(5) comment '顾客名' not null,

Cphone char(11) comment '电话' not null,

VIP set('食','日','家') comment '会员'

);

产品表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 | 约束 |
| Pid | varchar(5) | 产品号 | 例如：K1001，长度固定 | 主键 |
| Pname | varchar(10) | 产品名 | 例如果冻， 最多不超过10个字 | 非空 |
| Pprice | float(4,2) | 单价 | 例：20.12（精确到小数点后2位） | 非空 |

建表语句：

create table products(

Pid char(5) comment '产品号' primary key,

Pname varchar(10) comment '产品名' not null,

Pprice float(6,2) comment '单价 not null'

);

**订单表**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 | 约束 |
| Oid | char(12) | 订单编号 | 例如：D20200728001，定长 | 主键 |
| Cid | int | 顾客号 |  |  |
| Cnumber | Int | 订货数量 | 整数 | 非空 |
| Cotime | TIMESATMP | 订货日期 | 自动获取下单时间，格式为年月日时分秒 |  |
| Cdtime | date | 交货日期 | 记录交货时间，格式为年月日 |  |

建表语句：

create table sales(

Oid char(12) comment '订单编号' primary key,

sid int comment '顾客号',

Snumber int comment '订货数量' not null,

Sotime TIMESTAMP comment '订货日期',

Sdtime DATE comment '交货日期'

);

订单明细表：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| 属性名 | 数据类型 | 含义 | 存储示例 | 约束 |
| Oid | char(12) | 订单编号 |  | 外键级联 |
| Pid | char(5) | 产品号 |  | 外键级联 |
| Dnumber | Int | 订货数量 | 整数 |  |
| Dmoney | decimal(7,2) | 金额 | 精确到小数点后2位 | 级联外键 |

建表语句：

create table detailed(

Oid char(12) comment '订单编号',

Pid char(5) comment '产品号',

Dnumber int comment '订货数量',

Dmoney decimal(7,2) comment '金额',

-- primary key(Oid,Pid,Dnumber),#定义联合主键

foreign key (Oid) REFERENCES sales(Oid) on update CASCADE, #建立订单编号的外键并级联修改

foreign key (Pid) REFERENCES products(Pid) on update CASCADE, #建立产品号的外键并级联修改

-- foreign key (Dnumber) REFERENCES sales(Snumber) on update CASCADE # Snumber非主键，故不必要去级联外键

);

详细语句：

create database sell;

use sell;

create table customer(

Cid int comment '顾客号' primary key auto\_increment,

Cname varchar(5) comment '顾客名' not null,

Cphone char(11) comment '电话' not null,

VIP set('食','日','家') comment '会员'

);

create table products(

Pid char(5) comment '产品号' primary key,

Pname varchar(10) comment '产品名' not null,

Pprice float(6,2) comment '单价' not null

);

create table sales(

Oid char(12) comment '订单编号' primary key,

sid int comment '顾客号' not null auto\_increment,

Snumber int comment '订货数量',

Sotime TIMESTAMP comment '订货日期',

Sdtime DATE comment '交货日期'

);

drop table sales;

create table detailed(

Oid char(12) comment '订单编号',

Pid char(5) comment '产品号',

Dnumber int comment '订货数量',

Dmoney decimal(7,2) comment '金额',

-- primary key(Oid,Pid,Dnumber),#定义联合主键

foreign key (Oid) REFERENCES sales(Oid) on update CASCADE, #建立订单编号的外键并级联修改

foreign key (Pid) REFERENCES products(Pid) on update CASCADE, #建立产品号的外键并级联修改

foreign key (Dnumber) REFERENCES sales(Snumber) on update CASCADE

);

drop table detailed;

#删除sales表中外键

alter table sales drop FOREIGN key detailed\_ibfk\_3 #外键名

select now();