数据类型是一种限制能够存储在表中数据类别的方法。但是对于很多应用来说，它们提供的约束太粗糙。例如，一个人的年龄应该只接受正值。但是没有任何一种标准数据类型只接受正值。SQL允许我们在列和表上定义约束。约束让我们能够根据我们的愿望来控制表中的数据。如果一个用户试图在一个列中保存违反一个约束的数据，一个错误会被抛出。即便是这个值来自于默认值定义，这个规则也同样适用。

**检查约束**

检查约束是最普通的约束类型。它允许我们指定一个特定列中的值必须要满足一个布尔表达式。

create table t\_user(id int,name text,age int check(age>0));--列级约束

create table t\_user(id int,name text,age int constraint chk\_age check(age>0));--给与约束一个独立的名称

create table t\_user(id int,name text,age int,check(age>0 and age<150));--表级约束

create table t\_user(id int,name text,age int,constraint chk\_age check(age>0));--给与约束一个独立的名称

alter table t\_user add check(age>0);--添加一个约束

alter table t\_user add constraint chk\_age check(age>0);--添加一个约束

alter table t\_user rename constraint chk\_age to chk\_age2; --重命名约束

alter table t1 add check(age>0) not valid; --不检查以前的值，只检查后面的值。not valid只用于check和外键约束

alter table t1 validate constraint t1\_age\_check;--手动检查

alter table t\_user drop constraint chk\_age;--删除约束

注意：当约束表达式的计算结果为NULL时， 检查约束会被认为是满足条件的。 因为大多数表达式在含有NULL操作数的时候结果都是NULL， 所以这些约束不能阻止字段值为NULL。 要确保字段值不为NULL， 可以使用非空约束。

**非空约束**

非空约束只是声明一个字段不能是NULL。

create table t1(id int,name text,age int not null);

注意：not null没有表级约束，但可通过check约束实现表级定义。

create table t1(id int,name text,age int,check(age is not null));

create table t1(id int,name text,age int,check(name <> ''));

create table t1(id int,name text,age int not null check(age >0));--一个列可以有多于一个的约束

NOT NULL约束有一个相反的情况：NULL约束。

create table t1(id int null,name text null,age int null);

alter table t1 alter age set not null;--约束立即被检查，表中的数据必须符合约束

alter table t1 alter age drop not null;

**唯一约束**

唯一约束可以保证在一个字段或者一组字段中的数据相较于表中其他行的数据是唯一的。增加唯一约束会在约束中列出的列或列组上自动创建一个唯一B-tree索引。

create table t1(id int unique,name varchar,age int);--列级约束

create table t1(id int constraint uk\_t1\_id unique,name text);--为约束命名

create table t1(id int,name text,age int,unique(id)); --表级约束

create table t1(id int,name text,age int,constraint uk\_t1\_id unique(id));

create table t1(id int,name text,age int,constraint uk\_t1\_id\_name unique(id,name));--多个字段

create unique index idx\_id on t1(id);

alter table t1 add unique(id);

alter table t1 add constraint uk\_t1\_id unique(id);

alter table t1 add constraint uk\_t1\_id\_name unique(id,name);

alter table t1 drop constraint uk\_t1\_id;

注意：unique约束允许空值存在，唯一约束只针对非空值。

**主键约束**

主键约束表示可以用作表中行的唯一标识符。这要求那些值都是唯一的并且非空。

增加主键将自动在主键列中创建一个唯一B-tree索引。并且会强制这些列为NOT NULL。

一个表最多只能有一个主键，但可以有多个唯一非空约束。

create table t1(id int primary key,name text,age int);

create table t1(id int,name text,age int,primary key(id));

create table t1(id int constraint pk\_t7 primary key,name text,age int);

create table t1(id int,name text,age int,constraint pk\_t7 primary key(id,name));

alter table t1 add primary key(id);

alter table t1 add constraint pk\_t1\_id primary key(id);

alter table t1 add constraint pk\_t1\_id\_name primary key(id,name);

重建一个主键约束，并且在重建索引期间不阻塞更新：

create unique index concurrently uk\_t1\_id on t1(id);

alter table t1 drop constraint t1\_pkey,add constraint pk\_t1 primary key using index uk\_t1\_id;

alter table t1 drop constraint pk\_t1;

**外键约束**

外键约束指定一列中的值必须匹配出现在另一个表中某些行的值。这维持了两个关联表之间的引用完整性。

外键所引用的列必须是一个主键或者唯一键。这意味着被引用列总是拥有一个索引，因此在其上进行匹配的检查将会很高效。

包含外键的表称为引用表或子表。而外键引用的表称为被引用表或父表。

一个表可以有多个外键，这取决于它与其他表的关系。

create table class(id int primary key,name text);

create table student(id int,name text,age int,

class\_id int references class(id)

);

如果不指定列，则被引用表的主键将被用作被引用列。

create table student(id int,name text,age int,

class\_id int references class

);

create table student(id int,name text,age int,

class\_id int,

foreign key(class\_id) references class(id)

);

为外键约束命名

create table student(id int,name text,age int,

class\_id int constraint fk\_student references class(id)

);

外键也可以约束和引用一组列：

CREATE TABLE t1 (

a integer PRIMARY KEY,

b integer,

c integer,

FOREIGN KEY (b, c) REFERENCES other\_table (c1, c2)

多个外键

create table student(id int primary key,name text);

create table course(id int primary key, name text);

create table score(

student\_id int references student(id),

course\_id int references course,

scores int not null,

primary key(student\_id,course\_id)

);

alter table student add constraint fk\_student foreign key(class\_id) references class(id);

增加一个外键约束，并且不影响其他工作：

alter table student add constraint fk\_student foreign key(class\_id) references class(id) not valid;

alter table student validate constraint fk\_student;

insert into class values(1,'one'),(2,'two'),(3,'three'),(4,'four'),(5,'five'),(6,'six');

insert into student values(1,'zhangsan',22,1),(2,'lisi',18,3),(3,'wangwu',24,5);

在子表里插入数据，如果父表里没有，会报错。

同样，在父表里面随意删除数据也是不行的，因为主外键约束，主表也要对子表负责。

PostgreSQL 提供这几个主要选项来控制删除操作：DELETE RESTRICT, DELETE CASCADE 和 NO ACTION。

限制删除或者级联删除是两种最常见的选项。

RESTRICT阻止删除一个被引用的行。

NO ACTION表示在约束被检察时如果有任何引用行存在，则会抛出一个错误，这是我们没有指定任何东西时的默认行为（这两种选择的本质不同在于NO ACTION允许检查被推迟到事务的最后，而RESTRICT则不会）。

CASCADE指定当一个被引用行被删除后，引用它的行也应该被自动删除。

SET NULL和SET DEFAULT。这些将导致在被引用行被删除后，引用行中的引用列被置为空值或它们的默认值。

PostgreSQL 不允许删除任何被其它表引用的记录。直到 student 中的所有引用行被删除，PostgreSQL 才会删除 class表中的行。我们可以在定义外键的时候，使用 DELETE RESTRICT 来达到这一目标。

alter table student drop constraint student\_class\_id\_fkey;

alter table student add constraint fk\_student foreign key (class\_id) references class (id) on delete restrict;

如果要让 PostgreSQL 在删除某条记录的同时，将所有引用该条记录的关联记录也一起删除，我们可以使用 DELETE CASCADE：

alter table student drop constraint fk\_student;

alter table student add constraint fk\_student foreign key (class\_id) references class (id) on delete cascade;

如果要让 PostgreSQL 在删除某条记录的同时，将所有引用该条记录的关联记录置为null，我们可以使用 DELETE set null：

alter table student drop constraint fk\_student;

alter table student add constraint fk\_student foreign key (class\_id) references class (id) on delete set null;

提示，这些 DELETE 的行为同样适用于 UPDATE， 也就是说，有 ON UPDATE RESTRICT, ON UPDATE CASCADE 和 ON UPDATE NO ACTION 等行为。

**默认约束**

create table t\_user(id int,name text,age int default 12);

create table t\_user(id int,name text default 'abc',age int);

create table t\_user(id int,name text,age int,create\_time timestamp default now());

create table orders(

id int,

num int,

price numeric,

total\_price numeric GENERATED ALWAYS AS (num \* price) STORED

);

create table t1(id int serial primary key,name text,age int);

create table t3

(

id int generated always as identity (cache 100 START WITH 1 INCREMENT BY 1) primary key ,

name varchar(100)

);

alter table t\_user alter age set default 12;

alter table t\_user alter age drop default;--这实际上相当于把默认值设置为"NULL"。

如果没有明确声明默认值， 那么默认值是NULL。设置默认值的操作并不会影响表中现有的数据行， 它只会改变之后的INSERT命令的默认值。

**排他约束**

排他性约束机制特别适用于解决有关时间安排的问题。

增加排他约束将在约束声明所指定的类型上自动创建索引。

防止会议室预订冲突

CREATE TABLE room(use\_time tstzrange,exclude using gist(use\_time with &&));

ALTER TABLE room ADD CONSTRAINT ex\_room EXCLUDE USING gist (time\_slot WITH &&);

insert into room values('[2022-11-20 10:00:00,2022-11-20 11:00:00)');

insert into room values('[2022-11-20 10:30:00,2022-11-20 11:00:00)');--ERROR

防止圆重叠

CREATE TABLE circles ( c circle, EXCLUDE USING gist (c WITH &&) );

insert into circles values('(1,1),3');

insert into circles values('(2,2),3');