MySQL数据库(3)[1]

约束

创建表时添加约束

约束是作用于表中字段上的规则，用于限制存储在表中的数据。

目的：保证数据库中数据的正确、有效性和完整性。

约束是为了保证进入数据库的数据都是有效的、可靠的，会对列的值进行一些约束，确保存进去的数据都是有效的。

> 查看约束 `show create table 表名称;`

分类：

约束	描述	关键字
主键约束	主键是一行数据的唯一标识，要求非空且唯一	PRIMARY KEY (primary key)
唯一约束	保证该字段的所有数据都是唯一、不重复的	UNIQUE (unique)
默认约束	保存数据时，如果未指定该字段的值，则采用默认值	DEFAULT (default)
非空约束	限制该字段的数据不能为null	NOT NULL (not null)
检查约束(8.0.16版本之后)	保证字段值满足某一个条件	CHECK (check)
外键约束	用来让两张表的数据之间建立连接，保证数据的一致性和完整性	FOREIGN KEY (foreign key)

> 约束是作用于表中字段上的，可以在创建表/修改表的时候添加约束。

主键约束（PK）

主键约束最显著的特征是主键列中的值是不允许重复的，通过主键约束可强制表的实体完整性。当创建或更改表时可通过定义 PRIMARY KEY 约束来创建主键。一个表只能有一个 PRIMARY KEY 约束，且 PRIMARY KEY 约束中的列不能接受NULL值。

```
# 设置该字段为主键，主键约束名称
ALTER TABLE 表名 ADD CONSTRAINT 主键约束自定义名称 PRIMARY KEY(字段名);
alter table 表名 add constraint 主键约束自定义名称 primary key(字段名);
```

设置主键约束的几种方式：

```
-- 1.创建表的时候指定主键约束
CREATE TABLE `table_name` (
    # 刚开始的时候设置
    `id` int PRIMARY KEY,-- 设置主键
    `name` varchar(20) ,
);
CREATE TABLE `table_name` (
    `id` int,
    `name` varchar(20) ,
    # 设置完字段之后设置
    PRIMARY KEY (`id`) -- 设置主键
);
CREATE TABLE `table_name` (
    `id` int,
    `name` varchar(20),
    # 设置完字段之后设置并且自定义约束名称
    constraint pk primary key(id) -- 设置主键
);
-- 2.修改某一列为主键
ALTER TABLE 表名称 ADD [CONSTRAINT] PRIMARY KEY(字段名);
alter table 表名称 add [constraint] primary key(字段名);
-- 修改列类型的方式改主键约束
ALTER TABLE 表名称 MODIFY [COLUMN] 字段名 类型 PRIMARY KEY;
alter table 表名称 modify [column] 字段名 类型 primary key;
-- 修改列名称和类型的方式改主键约束
ALTER TABLE 表名称 CHANGE [COLUMN] 字段名 字段名 属性 PRIMARY KEY;
alter table 表名 change [column] 字段名 字段名 属性 primary key;
```

删除主键约束

```sql
# mysql 8.0.22 之后的版本
ALTER TABLE 表名称 DROP PRIMARY KEY;
alter table 表名称 drop primary key;
# mysql 8.0.22 之前的版本
ALTER TABLE 表名称 DROP INDEX PRIMARY KEY;
alter table 表名称 drop index primary key;
```

自增长列（标识列）

数据库提供了自增长列，自增长列是int类型的，其值是由数据库自动维护的，是永远都不会重复的，因此自增长是最适合作为主键列的。

`AUTO_INCREMENT` 关键字来标识自增长列，在MySQL数据库中自增长列必须是主键列或唯一约束。

特点：

1. 标识列必须和一个Key搭配（Key指主键、唯一、外键....）
2. 一个表最多有一个标识列
3. 标识列的类型只能是数值型
4. 标识列可以通过 SET auto_increment_increment = 3; 设置步长（全局，退出数据库重新进入会恢复默认值），可以通过插入行时手动插入标识列值设置起始值。

```sql
CREATE TABLE goods(
    -- 直接设置自增长
    no INT PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(10)
);
-- 修改为自增长列
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型 AUTO_INCREMENT;
alter table 表名 modify [column] 列名 列类型 auto_increment;

ALTER TABLE 表名 CHANGE 列名 列名 列类型 AUTO_INCREMENT;
alter table 表名 change 列名 列名 列类型 auto_increment;
-- 删除自增长列
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型;
alter table 表名 modify [column] 列名 列类型;
```

联合主键

联合主键（Composite Primary Key）是指在数据库表中，由多个列共同组成的主键，用来唯一标识表中的每一行数据。它的作用类似于单一列的主键，但不是由单个列组成，而是由多个列组合而成。联合主键可以确保表中的每一行都具有唯一性，并且每个列组合的值都不会重复。在创建表时，可以在列定义中指定多个列作为联合主键。

联合主键在以下情况非常有用：

1. 当单个列无法唯一标识表中的每一行，但多个列组合在一起可以唯一标识每一行数据时。
2. 提高查询性能：联合主键可以更有效地支持涉及多个列的查询，避免创建额外的索引。
3. 在具有多个外键的关联表中，可以使用联合主键来确保外键引用的准确性。

> 联合主键要求每个列组合的值都是唯一的

```
CREATE TABLE 表名 (
    列名1 数据类型,
    列名2 数据类型,
    列名3 数据类型,
    PRIMARY KEY (列名1, 列名2, 列名3)
);
-- 修改列的时候创建
ALTER TABLE 表名 ADD [CONSTRAINT] PRIMARY KEY (列名1, 列名2, 列名3);
alter table 表名 add [constraint] primary key(列名1, 列名2, 列名3);
-- 删除
ALTER TABLE 表名称 DROP PRIMARY KEY;
alter table 表名称 drop primary key;
```

唯一约束

对于非主键列中的值也要求唯一性时，就需要唯一约束

> 唯一约束允许有多个 NULL 值

```
-- 创建表时
CREATE TABLE `table_name` (
    `id` int,
```

```
    `name` varchar(20) UNIQUE # 唯一约束
);
CREATE TABLE `table_name` (
    `id` int,
    `name` varchar(20),
    CONSTRAINT uq UNIQUE(name) #唯一约束
);
-- 修改表
ALTER TABLE 表名 ADD UNIQUE(列名称);
alter table 表名 add unique(列名称);

ALTER TABLE 表名 ADD CONSTRAINT [约束名称] UNIQUE(列名称);
alter table 表名 add constraint [约束名称] unique(列名称);

ALTER TABLE 表名 CHANGE [COLUMN] 列名 列名 类型 UNIQUE;
alter table 表名 change [column] 列名 列名 类型 unique;

ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型 UNIQUE;
alter table 表名 modify [column] 列名 列类型 unique;
-- 删除唯一约束
# 如果没有设置约束名称，名称默认是字段名
ALTER TABLE 表名称 DROP INDEX 设置唯一时的名称;
alter table 表名称 drop index 设置唯一时的名称;
```

默认约束

为列中的值设置默认值， `DEFAULT 设置的值`

> 如果已经设置了值，默认值null就无效了

```
-- 创建表时
CREATE TABLE `table_name` (
    `id` int DEFAULT value,
    `name` varchar(20) unique,
);
-- 修改表
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型 DEFAULT 默认值;
alter table 表名 modify [column] 列名 列类型 default 默认值;

ALTER TABLE 表名 CHANGE 列名 列名 列类型 DEFAULT 默认值;
alter table 表名 change 列名 列名 列类型 default 默认值;
-- 删除
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型;
```

```
alter table 表名 modify [column] 列名 列类型;

# 8.0.23以上的版本
ALTER TABLE 表名 ALTER COLUMN 列名 DROP DEFAULT;
alter table 表名 alter column 列名 drop default;

# 8.0.23以前的版本
ALTER TABLE 表名 ALTER COLUMN 列名 SET DEFAULT NULL;
alter table 表名 alter column 列名 set default null;
```

非空约束

NOT NULL：非空，用于保证该字段的值不能为空。

> 修改列的约束确保现有数据满足非空约束条件，否则可能导致操作失败。

```
-- 创建表时
CREATE TABLE `table_name` (
    `id` int NOT NULL, # 非空约束
    `name` varchar(20),
);
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型 NOT NULL;
alter table 表名 modify [column] 列名 列类型 not null;

ALTER TABLE 表名 CHANGE 列名 列名 列类型 NOT NULL;
alter table 表名 change 列名 列名 列类型 not null;
-- 删除
ALTER TABLE 表名 MODIFY [COLUMN] 列名 列类型 [null];
alter table 表名 modify [column] 列名 列类型 [null];
```

检查约束

从 MySQL 8.0.19 版本开始， MySQL 支持了检查约束。检查约束允许你在表定义中声明条件，并确保符合该条件的数据才能插入或更新到相应的列中。

```
CREATE TABLE employees (
    id INT AUTO_INCREMENT PRIMARY KEY,
    name VARCHAR(50),
    age INT,
    email VARCHAR(100),
```

```
    CHECK (age ⩾ 18) -- 添加检查约束：年龄必须大于等于18
);
-- 修改表的时候
ALTER TABLE 表名 CHANGE 列名 列名 列类型 CHECK(约束条件);
alter table 表名 change 列名 列名 列类型 check(约束条件);

ALTER TABLE 表名 MODIFY 列名 列类型 CHECK(约束条件);
alter table 表名 modify 列名 列类型 check(约束条件);

ALTER TABLE 表名 ADD CONSTRAINT 列名 CHECK(约束条件);
alter table 表名 add constraint 列名 check(约束条件);

-- 删除检查约束
ALTER TABLE 表名 DROP CONSTRAINT 约束名;
alter table 表名 drop constraint 约束名;
```

外键约束（FK）

用来让两张表的数据之间建立连接，从而保证数据的一致性和完整性。表中列的值来自于另外一张表的主键或唯一键的列称为外键 FK ，将被引用值的表称为主表或父表，将引用值的表称为从表或子表。

- 外键列类型需要与引用列类型一致
- 外键列的值必须是主表中引用列的值或者 NULL
- 一个表可以有多个外键列
- 从表列可以随便删除
- 删除主表数据时，会先检查从表中有没有对此数据的关联，如果有就不能直接删除
    - 在设置外键的时候后面添加 `on delete cascade` / `on update cascade` 在删除/更新主表时，级联删除/更新外键列的数据
    - 在设置外键的时候后面添加 `on delete set null` / `on update set null` 在删除/更新主表时，外键列的值会变成null

```
-- 创建表时
CREATE TABLE `table_name` (
    `id` int NOT NULL,
    `name` varchar(20),
    `rid` int,
CONSTRAINT fk_a_b FOREIGN KEY(rid) REFERENCES b(id)
```

```
);

ALTER TABLE 从表表名 ADD CONSTRAINT 约束名称 FOREIGN KEY (从表字段) REFERENCES 主表表名(主表字段);
alter table 从表表名 add constraint 约束名称 foreign key (从表字段) references 主表表名(主表字段);

-- 删除
# 设置外键时的名称
ALTER TABLE 表名称 DROP FOREIGN KEY 索引名;
alter table 表名称 drop foreign key 索引名;
```

聚合函数

`MYSQL` 中内置了 5 种聚合函数，分别是：`SUM` 、 `MAX` 、 `MIN` 、 `AVG` 、 `COUNT` 。

- `sum` ： 求和
  ```
  select sum(列) from table_name [其他子句];
  ```
- `max` : 求最大值
  ```
  select max(列) from table_name [其他子句];
  ```
- `min` : 求最小值
  ```
  select min(列) from table_name [其他子句];
  ```
- `avg` ： 求平均值
  ```
  select avg(列) from table_name [其他子句];
  ```
- `count` : 求数量
  ```
  select count(列) from table_name [其他子句];
  ```

group by

`group by` 是对数据进行分组，分组时，表中有相同值的分为一组。分组后可以进行聚合查询。

`group by` 分组后的查询中， select 的列不能出现除了 group by 分组条件以及聚合函数外的其他列。

```
select 列1, 列2, (聚合函数) from table_name group by 列1, 列2;
```

having

`having` 是对 `group by` 分组后的结果集进行筛选。

```
select 列1, 列2, (聚合函数) from table_name group by 列1, 列2 having 分组后条件;
```

1. 第三天MySQL数据库学习笔记 ↩