

# 约束和分页

# 目 标

通过本章学习，您将可以：

- 描述约束
- 创建和维护约束
- 数据库分页

# 什么是约束

- 为了保证数据的一致性和完整性，SQL规范以约束的方式对表数据进行额外的条件限制。
- 约束是表级的强制规定
- 可以在创建表时规定约束（通过 CREATE TABLE 语句），或者在表创建之后也可以（通过 ALTER TABLE 语句）

# 约 束

- 有以下六种约束：
  - **NOT NULL** 非空约束，规定某个字段不能为空
  - **UNIQUE** 唯一约束，规定某个字段在整个表中是唯一的
  - **PRIMARY KEY** 主键 (非空且唯一)
  - **FOREIGN KEY** 外键
  - **CHECK** 检查约束
  - **DEFAULT** 默认值

注意：MySQL不支持check约束，但可以使用check约束，而没有任何效果；  
具体细节可以参阅W3Cschool手册

- 根据约束数据列的限制，约束可分为：
  - 单列约束：每个约束只约束一列
  - 多列约束：每个约束可约束多列数据
- 根据约束的作用范围，约束可分为：
  - 列级约束只能作用在一个列上，跟在列的定义后面
  - 表级约束可以作用在多个列上，不与列一起，而是单独定义

# NOT NULL 约束

- 非空约束用于确保当前列的值不为空值，非空约束只能出现在表对象的列上。
- Null类型特征：
  - 所有的类型的值都可以是null，包括int、float等数据类型
  - 空字符串""不等于null，0也不等于null

# NOT NULL 约束

保证列值不能为空：

EMPLOYEE_ID	LAST_NAME	EMAIL	PHONE_NUMBER	HIRE_DATE	JOB_ID	SALARY	DEPARTMENT_ID
100	King	SKING	515.123.4567	17-JUN-87	AD_PRES	24000	90
101	Kochhar	NKOCHHAR	515.123.4568	21-SEP-89	AD_VP	17000	90
102	De Haan	LDEHAAN	515.123.4569	13-JAN-93	AD_VP	17000	90
103	Hunold	AHUNOLD	590.423.4567	03-JAN-90	IT_PROG	9000	60
104	Ernst	BERNST	590.423.4568	21-MAY-91	IT_PROG	6000	60
178	Grant	KGRANT	011.44.1644.429263	24-MAY-99	SA_REP	7000	
200	Whalen	JWHALEN	515.123.4444	17-SEP-87	AD_ASST	4400	10

...

NOT NULL 约束

NOT NULL 约束

无NOT NULL 约束

# NOT NULL 约束

## ●创建 not null 约束:

```
CREATE TABLE emp(  
id INT(10) NOT NULL,  
NAME VARCHAR(20) NOT NULL DEFAULT 'abc',  
sex CHAR NULL  
);
```

## ●增加 not null 约束:

```
ALTER TABLE emp  
MODIFY sex VARCHAR(30) NOT NULL;
```



# NOT NULL 约束

- 取消 not null 约束:

```
ALTER TABLE emp  
MODIFY sex VARCHAR(30) NULL;
```

- 取消 not null 约束, 增加默认值:

```
ALTER TABLE emp  
MODIFY NAME VARCHAR(15) DEFAULT 'abc' NULL;
```

# UNIQUE 约束

EMPLOYEES

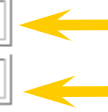
EMPLOYEE_ID	LAST_NAME	EMAIL
100	King	SKING
101	Kochhar	NKOCHHAR
102	De Haan	LDEHAAN
103	Hunold	AHUNOLD
104	Ernst	BERNST

...



INSERT INTO

208	Smith	JSMITH
209	Smith	JSMITH



允许  
不允许: 已经  
存在

唯一约束，允许出现多个空值：NULL。

# UNIQUE 约束

- 同一个表可以有多个唯一约束，多个列组合的约束。在创建唯一约束的时候，如果不给唯一约束名称，就默认和列名相同。
- MySQL会给唯一约束的列上默认创建一个唯一索引

```
CREATE TABLE USER(  
  id INT NOT NULL,  
  NAME VARCHAR(25),  
  PASSWORD VARCHAR(16),  
  #使用表级约束语法  
  CONSTRAINT uk_name_pwd UNIQUE (NAME,PASSWORD)  
);
```

- 表示用户名和密码组合不能重复

# UNIQUE 约束

- 添加唯一约束

```
ALTER TABLE USER  
ADD UNIQUE (NAME, PASSWORD);
```

```
ALTER TABLE USER  
ADD CONSTRAINT uk_name_pwd UNIQUE (NAME, PASSWORD);
```


```
ALTER TABLE USER  
MODIFY NAME VARCHAR(20) UNIQUE;
```

- 删除约束

```
ALTER TABLE USER  
DROP INDEX uk_name_pwd;
```

# PRIMARY KEY 约束

DEPARTMENTS

 PRIMARY KEY

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

...

不允许  
(空值)



INSERT INTO

	Public Accounting		1400
50	Finance	124	1500

不允许  
(50 已经存在)



# PRIMARY KEY 约束

- 主键约束相当于**唯一约束+非空约束**的组合，主键约束列不允许重复，也不允许出现空值
- 如果是多列组合的主键约束，那么这些列都不允许为空值，并且组合的值不允许重复。
- 每个表最多只允许一个主键，建立主键约束可以在列级别创建，也可以在表级别上创建。
- MySQL的主键名总是PRIMARY，当创建主键约束时，系统默认会在所在的列和列组合上建立对应的唯一索引。

# PRIMARY KEY 约束

```
CREATE TABLE emp4 (  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  NAME VARCHAR(20)  
);
```

列级模式

```
CREATE TABLE emp5 (  
  id INT NOT NULL AUTO_INCREMENT,  
  NAME VARCHAR(20),  
  pwd VARCHAR(15),  
  CONSTRAINT emp5_id_pk PRIMARY KEY(id)  
);
```

表级模式

```
CREATE TABLE emp6 (  
  id INT NOT NULL,  
  NAME VARCHAR(20),  
  pwd VARCHAR(15),  
  CONSTRAINT emp7_pk PRIMARY KEY(NAME, pwd)  
);
```

组合模式

# PRIMARY KEY 约束

- 删除主键约束

```
ALTER TABLE emp5  
DROP PRIMARY KEY;
```

- 添加主键约束

```
ALTER TABLE emp5  
ADD PRIMARY KEY (NAME, pwd);
```

- 修改主键约束

```
ALTER TABLE emp5  
MODIFY id INT PRIMARY KEY;
```



# FOREIGN KEY 约束

DEPARTMENTS

DEPARTMENT_ID	DEPARTMENT_NAME	MANAGER_ID	LOCATION_ID
10	Administration	200	1700
20	Marketing	201	1800
50	Shipping	124	1500
60	IT	103	1400
80	Sales	149	2500

PRIMARY  
KEY



...

EMPLOYEES

EMPLOYEE_ID	LAST_NAME	DEPARTMENT_ID
100	King	90
101	Kochhar	90
102	De Haan	90
103	Hunold	60
104	Ernst	60
107	Lorentz	60

FOREIGN  
KEY



...

INSERT INTO

200	Ford	9
201	Ford	60

不允许(9 不  
存在)  
允许



# FOREIGN KEY 约束

- 外键约束是保证一个或两个表之间的参照完整性，外键是构建于一个表的两个字段或是两个表的两个字段之间的参照关系。
- 从表的外键值必须在主表中能找到或者为空。当主表的记录被从表参照时，主表的记录将不允许删除，如果要删除数据，需要先删除从表中依赖该记录的数据，然后才可以删除主表的数据。
- 还有一种就是级联删除子表数据。
- 注意：外键约束的参照列，在主表中引用的只能是主键或唯一键约束的列
- 同一个表可以有多个外键约束

# FOREIGN KEY 约束

1. **cascade**: 在父表上 update / delete记录时，同步 update / delete掉子表的匹配记录
2. **No action**: 如果子表中有匹配的记录，则不允许对父表对应候选键进行update / delete操作
3. **Restrict**: 同 no action, 都是立即检查外键约束
4. **set null**: 在父表上 update / delete记录时，将子表上匹配记录的列设为 null要注意子表的外键列不能为 not null

# FOREIGN KEY 约束

- 创建外键约束:

```
CREATE TABLE dept(  
  dept_id INT AUTO_INCREMENT PRIMARY KEY,  
  dept_name VARCHAR(20)  
);
```

主表

```
CREATE TABLE emp(  
  emp_id INT AUTO_INCREMENT PRIMARY KEY,  
  last_name VARCHAR(15),  
  dept_id INT,  
); CONSTRAINT emp_dept_id_fk FOREIGN KEY(dept_id)  
REFERENCES dept(dept_id)
```

从表

# FOREIGN KEY 约束

- 创建多列外键组合，必须使用表级约束：

```
CREATE TABLE classes(  
  id INT,  
  NAME VARCHAR(20),  
  number INT,  
  PRIMARY KEY (NAME, number)  
);
```

主表

```
CREATE TABLE student(  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  classes_name VARCHAR(20),  
  classes_number INT,  
  FOREIGN KEY (classes_name, classes_number)  
  REFERENCES classes (NAME, number)  
);
```

从表

# FOREIGN KEY 约束

- 删除外键约束:

```
ALTER TABLE emp  
DROP FOREIGN KEY emp_dept_id_fk;
```

- 增加外键约束:

```
ALTER TABLE emp  
ADD [CONSTRAINT emp_dept_id_fk] FOREIGN KEY(dept_id)  
REFERENCES dept(dept_id);
```

# FOREIGN KEY 约束的关键字

- FOREIGN KEY: 在表级指定子表中的列
- REFERENCES: 标示在父表中的列
- **ON DELETE CASCADE(级联删除)**: 当父表中的列被删除时，子表中相对应的列也被删除
- **ON DELETE SET NULL(级联置空)**: 子表中相应的列置空

```
CREATE TABLE student(  
  id INT AUTO_INCREMENT PRIMARY KEY,  
  NAME VARCHAR(20),  
  classes_name VARCHAR(20),  
  classes_number INT,  
  /*表级别联合外键*/  
  FOREIGN KEY(classes_name, classes_number)  
  REFERENCES classes(NAME, number) ON DELETE CASCADE);
```

# CHECK 约束

- MySQL可以使用check约束，但check约束对数据验证没有任何作用，添加数据时，没有任何错误或警告

```
CREATE TABLE temp(  
  id INT AUTO_INCREMENT,  
  NAME VARCHAR(20),  
  age INT CHECK(age > 20),  
  PRIMARY KEY(id)  
);
```



# MySQL中使用limit实现分页

- 背景
  - 查询返回的记录太多了，查看起来很不方便，怎么样能够实现分页查询呢？
- 分页原理
  - 所谓分页显示，就是将数据库中的结果集，一段一段显示出来需要的条件

# MySQL中使用limit实现分页

- 怎么分段，当前在第几段（每页有几条，当前在第几页）
  - 前10条记录： `SELECT * FROM table LIMIT 0,10;`
  - 第11至20条记录： `SELECT * FROM table LIMIT 10,10;`
  - 第21至30条记录： `SELECT * FROM table LIMIT 20,10;`
- 公式：  
**( 当前页数-1 ) \* 每页条数 , 每页条数**  
`SELECT * FROM table LIMIT (PageNo - 1)*PageSize,PageSize;`
- 注意：
  - limit子句必须放在整个查询语句的最后！