关系数据模型

笔记本： database_theory
创建时间： 2021/6/26 15:25 **更新时间：** 2021/7/2 10:52
作者： 134exetj717

关系数据模型是当前最流行的。为什么？因此我们这个世界彼此间就是以联系构造出来的。因此，我们需要学懂关系数据模型。

- 最典型的，什么是表？
  - 描述数据本身、数据之间的联系。也成为关系。
  - 列：字段、属性、数据项、成员；
  - 行：元组、记录。
- 关系模式：对关系的描述，由关系名和各个列构成。
- 关系实例：记录集或元组集。
- 有表，有数据，有数据操作，那么就一定有约束，约束有哪些呢？
  - 域约束（列的取值范围）、主键约束、唯一约束、外键约束、一般性约束。
- 其中，外键约束比较特殊，因为它涉及到了两张表，此时，主表到从表、从表到主表的行动策略有哪些呢？
  - 级联约束：CASCADE
  - 空值：SET NULL
  - 默认值：SET DEFAULT
  - 禁止删除：NO ACTION
- 除了以上四种约束以外，其实我们还有更一般的约束，分别是什么呢？
  - 检查约束：单个表的检查，表中的某一列是否在取值范围之内，或者某几列之间是否满足指定的条件
  - 断言：多个表的检查
- 了解了数据库中各个约束以后，我们如何进行完整性约束及其设定呢？
  - DEFAULT的使用

```
create default 默认名 as '默认值'       //创建默认
```

```
sp_bindefault '默认名','表名.列名'       //绑定方法
```

```
sp_unbindefault '表名.列名'     //取消绑定的方法
```

```
drop default 默认名          //删除默认，（注意：应保证该默认已从所有绑定的列上摘除，否则删除不会成功）
```

  - RULE的设定

```
create rule 规则名 as 规则     //创建规则
```

```
sp_bindrule 规则名,'表名.列名'     //绑定方法
```

```
sp_unbindrule '表名.列名'            //解除绑定
```

```
drop rule 规则名                 //删除规则
```

  - 检查约束的设定

```
例子:
```

```
constraint 约束名
check (pub_id in ('234','3344') or pub_id like '43[0-9]'

其中 in 表示 是否为 234 或者3344，like 表示 是否形如 43+两个0-9的数字
```

- 主键约束的设定

```
例子:

pub_id char(4) primary key              //生成随机名称的主键约束
或者
constraint 约束名
primary key nonclustered (主键)        //生成给定名称的主键约束，且是聚簇索引
```

- 唯一约束的设定

```
例子：

constraint 约束名（一般后面可以加上 _const 格式）
unique clustered （属性）                                    //非聚簇的唯一
```

- 外键约束的设定

```
例子：

constraint 约束名
foreign key （属性）                  //显示指出所定义的外键，无实际意义
references 另一张表名（属性）          //表示参照该表
on delete cascade
on update cascade                     //表示级联
```

- 触发器的定义
  - inserted：’ 新 ‘数据会放在其中，
  - deleted：’ 老 ‘数据会放在其中，我们只需要调用即可

```
例子：

create 触发器名
on 表名
after delete / update   或者用   for insert,update            //for 、 after 都可以
as
   if   @@rowcount = 0 return          //@@rowcount   是系统变量，表示表中有几行数据被删
除了
   ...（系列操作）
   /*
      RAISERROR ('错'，16，1）          //报错
      ROLLBACK TRANSACTION          //退回事务
      RETURN
   */

drop trigger 触发器名              //表示删除触发器
```

- 很容易我们发现，以上全部都是针对表的操作或者约束，那么我们别忘了，数据库还有很重要的视图，那么它的操作又是怎样的呢？
    - 基本概念
        - 视图也是一张表，但其数据不存储于视图中，而是由视图定义从表中查询出来，故有时称其为虚表。
        - 而创建视图基于的表，称作基表。
    - 视图的创建

```
create view 视图名
as select .... from（后边相当于是查询）

drop view 视图名
```

    - 视图的修改
        - 需要说明的是，对视图的更新，其实就是对基表的更新，即增删改查基表中的对应内容

```
例子：

update 视图名
set phone = '88846486'
where ...
```

    - 视图的插入

```
例子：
insert into 视图名
values (...)
```

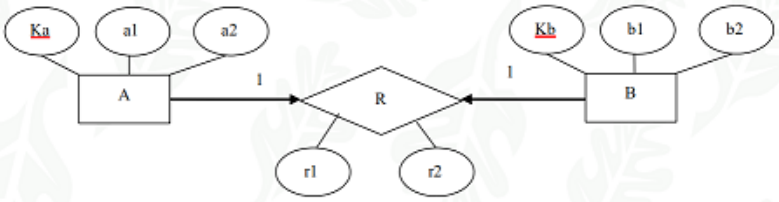
    - 视图的删除

```
例子：

delete 视图名
where ...
```

    - 值得注意的是：
        - ！！！！！以上增删改都有可能导致视图未改变，但是基表却改变了，因为我们改变的内容来源于基表中，但是此时视图的定义并不包含修改的内容
        - 解决办法：我们可以加上 with check option ，这样我们在修改视图的过程中，系统就会先判断修改的内容是否属于视图，如果不属于，那么就不能完成修改
- 讲完了视图的各类操作，我们此时就会思考，我们之前创建的实体联系模型，是一种图画，但我们的关系模型却是表，那么我们如何实现转换呢？
    - 实体型映射为表
    - 联系型的转换
        - 联系型映射为关系（即表）
        - 将联系所包含的属性移动到具有键约束的实体所映射到的表中

1）1:1联系



转换结果为：

A (Ka,a1,a2), Ka为主键

B (Kb,b1,b2), Kb为主键

R (Ka, Kb, r1, r2), Ka, Kb均为外键

由于有两个键约束，故Ka, Kb都能单独作为R的主键。如果指定Ka为主键，则Kb为候选键；否则，如指定Kb为主键，则Ka为候选键。

- 其中，我们需要注意的是，m：n的联系型只能映射为关系（即表）

- 以上内容中，我们主要学习了一些对数据的约束，但是，我们还有对数据的操作。关系代数与关系运算是两个与关系模型相关的查询语言。关系运算简单，那么关系代数具体是怎样的呢?
  - 关系代数表达式：由关系代数操作符和操作数组合形成的查询表达式。
  - 基本的代数操作符有如下：

8. 关系代数的基本操作符?

selection(选择) ： σ

projection(投影): π

union（并）

intersection（交）

difference（差）

cross-product（积）

3. Renaming

为解决R×S中产生的名字冲突，而引入“改名(ρ)”操作符。

表达式：ρ（R(F)，E）

说明：

①对任意代数表达式E (Expression)，ρ（R(F)，E）返回一个新的关系实例R，其元组与E相同，模式与E也相同，但某些列被改名。

②F的形式为：旧名→新名，或位置→新名。

③改名有两个作用：一是有名字冲突时可改名；二是作为表可临时保存表达式E的结果，表名为R。

-
- 将字段1中的姓名改为sid1，将字段5中的姓名改为sid2

示例：ρ（D(1→sid1, 5→sid2)，S1×E1）

4. Joins

概念：关系与关系的连接。可定义为R×S后跟选择。

种类：条件连接、等连接、自然连接、外连接。

(1) Condition Joins(条件连接)

概念：$R \bowtie_c S = \sigma_c(R \times S)$

说明：条件c会用到R和S的列，如R.name，R.i(位置)。

示例：$S1 \bowtie_{S1.sid<E1.sid} E1$

查询结果：

(sid)	sname	age	grade	(sid)	cid	score
8	何大明	19	2	66	108	80
11	李　峰	20	3	66	108	80
35	陈　胜	21	4	66	108	80

-

(2)等连接 (Equijoin)

概念：是条件连接的特例，即连接条件由等式组成，如R.name1=S.name2。

说明：由于列相等，因此结果中有重复的列，等连接定义中将此重复列在结果中去掉。

示例：$S1 \bowtie_{S1.sid=E1.sid} E1$

查询结果：

sid	sname	age	grade	cid	score
8	何大明	19	2	101	91

相应的SQL查询描述：

```
SELECT S1.sid, sname, age, grade, cid, score
FROM S1, E1
WHERE S1.Sid = E1.Sid
```

(3) Natural Join(自然连接)

概念：是等连接的特例，即：等式中所涉及的列名相同，这时可隐去连接条件，即为：$R \bowtie S$。

示例：$S1 \bowtie E1$

查询结果：同上。

这种关联才是最自然的。

相应的SQL查询描述：

```
SELECT S1.sid, sname, age, grade, cid, score
FROM S1 NATURAL JOIN E1
```

或：

```
SELECT S1.sid, sname, age, grade, cid, score
FROM S1 , E1
WHERE S1.Sid = E1.Sid
```

- **(4) 外连接 (Outer Joins)**

  概念：涉及有空值的自然连接，是自然连接的特例。

  说明：自然连接是寻找两表中相同列值相等的对应行。外连接除要寻找相同列值相等的对应行之外，还要列出一张表在另一张表中没有相同列值相等的对应行。因无对应，结果中某些列的值显示为NULL值。

  外连接的种类：

  ① 左外连接 ⟕ (LEFT OUTER JOIN)
  ② 右外连接 ⟖ (RIGHT OUTER JOIN)
  ③ 全外连接 ⟗ (FULL OUTER JOIN)

  种类的理解：因须对应，故要有参照点。以外连接操作符左边的表为参照，即为左外连接；如参照右表，则为右外连接；如既参照左表亦参照右表，则为全外连接。

  说明：与外连接对应，前面三种连接为**内连接** (Inner Join)。

  ```
  SELECT E1.sid, cid, sname
  FROM S1 NATURAL RIGHT OUTER JOIN E1
  ```

  示例及结果：

  S1

sid	sname	age	grade
8	何大明	19	2
11	李 峰	20	3
35	陈 胜	21	4

  E1

sid	cid	score
8	101	91
35	106	84
66	119	88
66	101	99

  结果：

sid	cid	sname
8	101	何大明
35	106	陈 胜
66	119	null
66	101	null

  右外自然连接示意图

- 除法操作，