第12章 数据库完整性

什么是数据库的完整性

- 数据的正确性和相容性
- 防止不合语义的数据进入数据库。

例：学生的年龄必须是整数，取值范围为14--29；

学生的性别只能是男或女；

学生的学号一定是唯一的；

学生所在的系必须是学校开设的系；

- 完整性：否真实地反映现实世界

完整性控制机制

1.完整性约束条件定义机制

2.完整性控制机制

3.违约反应

完整性约束条件定义

- 完整性约束条件：数据模型的组成部分约束数据库中数据的语义
- DBMS应提供定义数据库完整性约束条件，并把它们作为模式的一部分存入数据库中

完整性控制机制

- 检查用户发出的操作请求是否违背了完整性约束条件

违约反应

- 如果发现用户的操作请求使数据违背了完整性约束条件，则采取一定的动作来保证数据的完整性。

完整性约束条件（续）

完整性约束条件作用的对象

- 列：对属性的取值类型、范围、精度等的约束条件
- 元组：对元组中各个属性列间的联系的约束
- 关系：对若干元组间、关系集合上以及关系之间的联系的约束

完整性约束条件（续）

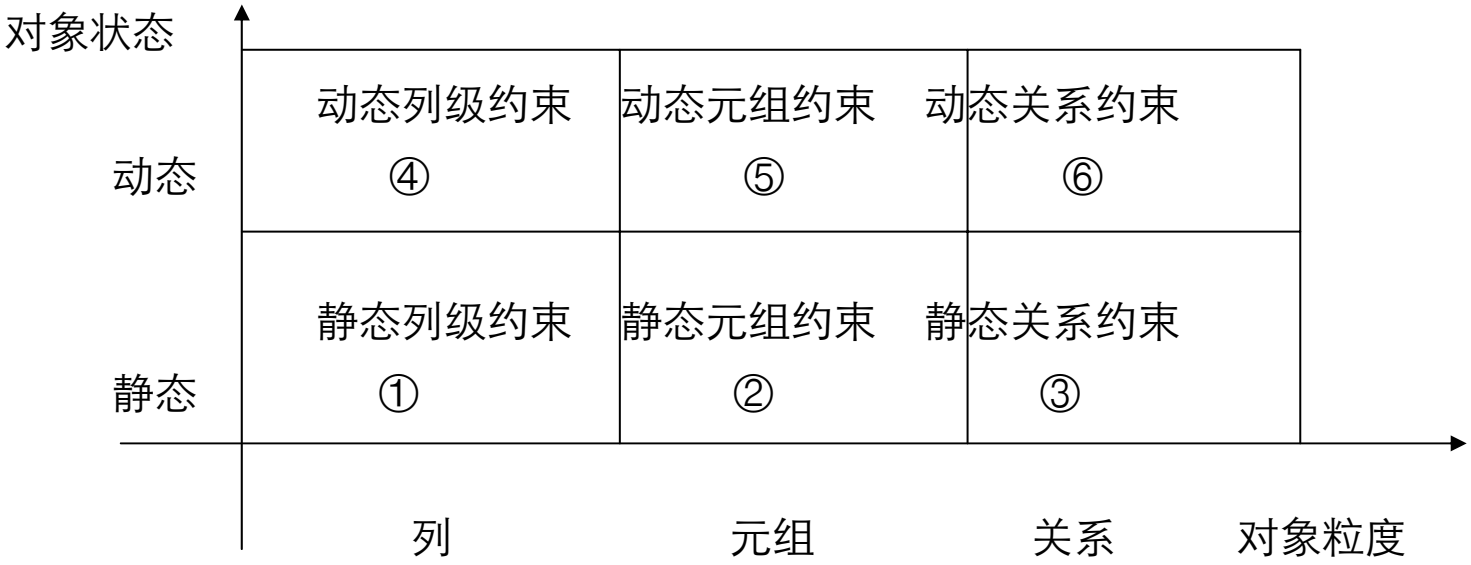
- 静态
  - 对静态对象的约束是反映数据库状态合理性的约束
- 动态
  - 对动态对象的约束是反映数据库状态变迁的约束

完整性约束条件分类

六类完整性约束条件

- 静态列级约束
- 静态元组约束
- 静态关系约束
- 动态列级约束
- 动态元组约束
- 动态关系约束

完整性约束条件（续）


对象状态
动态
静态
动态列级约束
④
动态元组约束
⑤
动态关系约束
⑥
静态列级约束
①
静态元组约束
②
静态关系约束
③
列
元组
关系
对象粒度

完整性约束条件（续）

1. 静态列级约束

- 静态列级约束：列的取值域的说明
- 最常见、最简单、最容易实现的一类完整性约束

完整性约束条件（续）

- 五类静态列级约束

1) 数据类型约束：数据的类型、长度、单位、精度等

例：学生姓名的数据类型为字符型，长度为8

2) 对数据格式的约束

例：

学号：前两位表示入学年份，后四位为顺序编号

日期：YY.MM.DD。

完整性约束条件（续）

3) 取值范围或取值集合的约束

例：规定成绩的取值范围为0-100

年龄的取值范围为14-29

性别的取值集合为[男,女]

4) 对空值的约束

空值：未定义或未知的值

空值：与零值和空格不同

有的列允许空值，有的则不允许，如成绩可为空值

5) 其他约束

例：关于列的排序说明，组合列等

2. 静态元组约束

- 规定元组的各个列之间的约束关系

  例：订货关系中发货量<=订货量

  教师关系中教授的工资>=700元

- 静态元组约束只局限在元组上

3. 静态关系约束

关系的各个元组之间或若干关系之间存在的各种联系或约束

常见静态关系约束:

1) 实体完整性约束
2) 参照完整性约束
3) 函数依赖约束
4) 统计约束

- 关系字段间存在的函数依赖

例：在学生－课程－教师关系

SJT(S,J,T) 的函数依赖：

((S,J）→T，T→J)

主码： (S, J)

统计约束

- 定义某个字段值一个关系多个元组的统计值之间的约束关系

  例：职工平均工资的2倍<=部门经理的工资<= 职工平均工资的5倍

  职工平均工资值: 统计值

4. 动态列级约束

动态列级约束是修改列定义或列值时应满足的约束条件。

1) 修改列定义时的约束

例：将原来允许空值的列改为不允许空值时:
该列目前已存在空值，则拒绝这种修改

2) 修改列值时的约束

修改列值时新旧值之间要满足的约束条件

例：职工工资调整 >= 原来工资
年龄只能增长

5. 动态元组约束

修改元组值: 各个字段之间要满足的约束条件

例: 职工工资调整不得低于其原来工资 + 工龄*1.5

6. 动态关系约束

关系变化前后状态：限制条件

例：事务一致性、原子性等约束条件

完整性约束条件小结

粒度 \ 状态	列 级	元 组 级	关 系 级
静 态	列定义 · 类型 · 格式 · 值域 · 空值	元组值应满足的条件	实体完整性约束 参照完整性约束 函数依赖约束 统计约束
动 态	改变列定义或列值	元组新旧值之间应满足的约束条件	关系新旧状态间应满足的约束条件

一、DBMS的完整性控制机制

二、关系系统三类完整性的实现

三、参照完整性的实现

一、DBMS的完整性控制机制

1. 定义功能

一个完善的完整性控制机制应该允许用户定义各类完整性约束条件。

2. 检查功能

- 立即执行的约束(Immediate constraints)

  语句执行完后立即检查是否违背完整性约束

- 延迟执行的约束(Deferred constrainsts)

  完整性检查延迟到整个事务执行结束后进行

DBMS的完整性控制机制（续）

例：银行数据库中“借贷总金额应平衡”的约束就应该是延迟执行的约束

- 从账号A转一笔钱到账号B为一个事务，从账号A转出去钱后账就不平了，必须等转入账号B后账才能重新平衡，这时才能进行完整性检查。

3. 违约反应

- 拒绝该操作
- 其他处理方法

DBMS的完整性控制机制（续）

完整性规则五元组表示:

(D，O，A，C，P)

- D（Data） 约束作用的数据对象;
- O（Operation） 触发完整性检查的数据库操作
  当用户发出什么操作请求时需要检查该完整性规则
  是立即检查还是延迟检查;
- A（Assertion） 数据对象必须满足的断言或语义约束这是规则的主体;
- C（Condition） 选择A作用的数据对象值的谓词;
- P（Procedure） 违反完整性规则时触发的过程。

DBMS的完整性控制机制（续）

例1：在“学号不能为空”的约束中

D 约束作用的对象为Sno属性

O 插入或修改Student 元组时

A Sno不能为空

C 无（A可作用于所有记录的Sno属性）

P 拒绝执行该操作

DBMS的完整性控制机制（续）

例2：在“教授工资不得低于1000元”的约束中

D 约束作用的对象为工资Sal属性

O 插入或修改职工元组时

A Sal不能小于1000

C 职称=′教授′

(A仅作用于职称=‘教授’的记录)

P 拒绝执行该操作

二、关系系统三类完整性的实现

- 关系数据库系统都提供了定义和检查实体完整性、参照完整性和用户定义的完整性的功能
- 违反实体完整性规则和用户定义的完整性规则的操作：

  一般是拒绝执行
- 违反参照完整性的操作：
  - 拒绝执行
  - 接受这个操作，同时执行一些附加的操作，以保证数据库的状态正确

三、参照完整性的实现

例:职工 – 部门数据库包含职工表EMP和部门表DEPT

1 DEPT关系的主码为部门号Deptno

2 EMP关系的主码为职工号Empno, 外码为部门号Deptno

称DEPT为被参照关系或目标关系，EMP为参照关系

RDBMS实现参照完整性时需要考虑以下4方面:

1. 外码是否可以接受空值的问题

- 外码是否能够取空值：依赖于应用环境的语义
- 实现参照完整性：

  系统提供定义外码的机制

  定义外码列是否允许空值的机制

1. 外码是否可以接受空值的问题

例1：在职工 – 部门数据库中，

EMP关系包含有外码Deptno

某元组的这一列若为空值，表示这个职工尚未分配到任何具体的部门工作

和应用环境的语义是相符

1. 外码是否可以接受空值的问题

例2：学生－选课数据库

Student关系为被参照关系，其主码为Sno。

SC为参照关系，外码为Sno。

若SC的Sno为空值：表明尚不存在的某个学生，或者某个不知学号的学生，选修了某门课程，其成绩记录在Grade中

与学校的应用环境是不相符的，因此SC的Sno列不能取空值。

2.被参照关系中删除元组时的问题

出现违约操作的情形：

删除被参照关系的某个元组（student）

而参照关系有若干元组(SC)的外码值与被删除的被参照关系的主码值相同

2.被参照关系中删除元组时的问题

- 违约反应：可有三种策略
  - 级联删除（CASCADES）
  - 受限删除（RESTRICTED）
  - 置空值删除（NULLIFIES）

  这三种处理方法，哪一种是正确的，要依应用环境的语义来定

2.被参照关系中删除元组时的问题

- 级联删除

  将参照关系中外码值与被参照关系中要删除元组主码值相对应的元组一起删除

- 受限删除

  当参照关系中没有任何元组的外码值与要删除的被参照关系的元组的主码值相对应时，系统才执行删除操作，否则拒绝此删除操作

2.被参照关系中删除元组时的问题

- 置空值删除

  删除被参照关系的元组，并将参照关系中与被参照关系中被删除元组主码值相等的外码值置为空值。

2.被参照关系中删除元组时的问题

例：要删除Student关系中Sno=950001的元组，而SC关系中有4个元组的Sno都等于950001。

- 级联删除：将SC关系中所有4个Sno=950001的元组一起删除。如果参照关系同时又是另一个关系的被参照关系，则这种删除操作会继续级联下去
- 受限删除：系统将拒绝执行此删除操作。
- 置空值删除：将SC关系中所有Sno=950001的元组的Sno值置为空值。
- 在学生选课数据库中，显然第一种方法和第二种方法都是对的。第三种方法不符合应用环境语义。

3.在参照关系中插入元组时的问题

- 出现违约操作的情形
  - 需要在参照关系中插入元组，而被参照关系不存在相应的元组
- 违约反应
  - 受限插入
  - 递归插入

3.在参照关系中插入元组时的问题

- 受限插入
  - 仅当被参照关系中存在相应的元组，其主码值与参照关系插入元组的外码值相同时，系统才执行插入操作，否则拒绝此操作。
- 递归插入
  - 首先向被参照关系中插入相应的元组，其主码值等于参照关系插入元组的外码值，然后向参照关系插入元组。

3.在参照关系中插入元组时的问题

例：向SC关系插入（99001，1，90）元组，而Student关系中尚没有Sno=99001的学生

- 受限插入：系统将拒绝向SC关系插入（99001，1，90）元组
- 递归插入：系统将首先向Student关系插入Sno=99001的元组，然后向SC关系插入（99001，1，90）元组。

4. 修改被参照关系中主码的问题

- 两种策略

(1)不允许修改主码

(2)允许修改主码

允许修改主码策略

- 违约操作
  - 要修改被参照关系中某些元组的主码值，而参照关系中有些元组的外码值正好等于被参照关系要修改的主码值
  - 要修改参照关系中某些元组的主码值，而被参照关系中没有任何元组的外码值等于被参照关系修改后的主码值
- 违约反应

  修改的关系是被参照关系：与删除类似
  - 级联修改
  - 受限修改
  - 置空值修改

允许修改主码策略

- 级联修改
  - 修改被参照关系中主码值同时，用相同的方法修改参照关系中相应的外码值。
- 受限修改
  - 拒绝此修改操作。只当参照关系中没有任何元组的外码值等于被参照关系中某个元组的主码值时，这个元组的主码值才能被修改。
- 置空值修改
  - 修改被参照关系中主码值，同时将参照关系中相应的外码值置为空值。

允许修改主码等略

例：将Student关系中Sno=950001的元组中Sno值改为960123。而SC关系中有 4个元组的Sno=950001

- 级联修改：将SC关系中4个Sno=950001元组中的Sno值也改为960123。如果参照关系同时又是另一个关系的被参照关系，则这种修改操作会继续级联下去。
- 受限修改：只有SC中没有任何元组的Sno=950001时，才能修改Student表中Sno=950001的元组的Sno值改为960123。
- 置空值修改：将Student表中Sno=950001的元组的Sno值改为960123。而将S表中所有Sno=950001的元组的Sno值置为空值。
- 在学生选课数据库中只有第一种方法是正确的。

允许修改主码策略

- 违约反应 (2)
- 修改的关系是参照关系：与插入类似
  - 受限插入
  - 递归插入

参照完整性的实现

RDBMS在实现参照完整性时:

- 需要向用户提供定义主码、外码的机制
- 向用户提供按照自己的应用要求选择处理依赖关系中对应的元组的方法

12.3 Oracle的完整性

一、Oracle中的实体完整性

二、Oracle中的参照完整性

三、Oracle中用户定义的完整性

一、ORACLE中的实体完整性

- ORACLE在CREATE TABLE语句中提供了PRIMARY KEY子句，供用户在建表时指定关系的主码列。
  - 在列级使用PRIMARY KEY子句
  - 在表级使用PRIMARY KEY子句

ORACLE中的实体完整性（续）

例1：在学生选课数据库中，要定义Student表的Sno属性为主码

```
CREATE TABLE Student
      (Sno   NUMBER(8),
       Sname VARCHAR(20),
       Sage  NUMBER(20),
       CONSTRAINT PK_SNO PRIMARY KEY (Sno));
```

或:

```
CREATE TABLE Student
      (Sno   NUMBER(8) PRIMARY KEY ,
       Sname VARCHAR(20),
       Sage  NUMBER(20));
```

ORACLE中的实体完整性（续）

例2：要在SC表中定义(Sno, Cno)为主码

```
CREATE TABLE SC
    (Sno   NUMBER(8),
    Cno   NUMBER(2),
    Grade NUMBER(2),
    CONSTRAINT PK_SC PRIMARY KEY (Sno, Cno));
```

ORACLE中的实体完整性（续）

- 用户程序对主码列进行更新操作时，系统自动进行完整性检查
- 违约操作
  - 使主属性值为空值的操作
  - 使主码值在表中不唯一的操作
- 违约反应
  - 系统拒绝此操作，从而保证了实体完整性

二、ORACLE中的参照完整性

定义参照完整性

- FOREIGN KEY子句：定义外码列
- REFERENCES子句：外码相应于哪个表的主码
- ON DELETE CASCADE子语：

  在删除被参照关系的元组时，同时删除参照关系中 外码值等于被参照关系的元组中主码值的元组

ORACLE中的参照完整性（续）

例1：建立表EMP表

```
CREATE TABLE EMP
    (Empno NUMBER(4),
     Ename VARCHAR(10),
     Job VERCHAR2(9),
     Mgr NUMBER(4),
     Sal NUMBER(7,2),
     Deptno NUMBER(2),
     CONSTRAINT FK_DEPTNO
          FOREIGN KEY (Deptno)
          REFERENCES DEPT(Deptno));
```

ORACLE中的参照完整性（续）

或：

```
CREATE TABLE EMP
    (Empno NUMBER(4),
     Ename VARCHAR(10),
     Job VERCHAR2(9),
     Mgr NUMBER(4),
     Sal NUMBER(7,2),
     Deptno NUMBER(2)  CONSTRAINT FK_DEPTNO
          FOREIGN KEY REFERENCES DEPT(Deptno));
```

ORACLE中的参照完整性（续）

- 这时EMP表中外码为Deptno，它相应于DEPT表中的主码Deptno。
- 当要修改DEPT表中的DEPTNO值时，先要检查EMP表中有无元组的Deptno值与之对应
  - 若没有，系统接受这个修改操作
  - 否则，系统拒绝此操作

ORACLE中的参照完整性（续）

- 当要删除DEPT表中某个元组时，系统要检查EMP表，若找到相应元组即将其随之删除。
- 当要插入EMP表中某个元组时，系统要检查DEPT表，先要检查DEPT表中有无元组的Deptno值与之对应
  - 若没有，系统拒绝此插入操作
  - 否则，系统接受此操作

三、ORACLE中用户定义的完整性

- ORACLE中定义用户完整性的两类方法
  - 用CREATE TABLE语句在建表时定义用户完整性约束
  - 通过触发器来定义用户的完整性规则

ORACLE中用户定义的完整性（续）

1. 用CREATE TABLE语句在建表时定义用户完整性约束

可定义三类完整性约束

- 列值非空（NOT NULL短语）
- 列值唯一（UNIQUE短语）
- 检查列值是否满足一个布尔表达式（CHECK短语）

ORACLE中用户定义的完整性（续）

例1：建立部门表DEPT，要求部门名称Dname列取值唯一，部门编号Deptno列为主码

```
CREATE TABLE DEPT
(Deptno NUMBER,
Dname VARCHAR(9) CONSTRAINT U1 UNIQUE,
Loc     VARCHAR(10),
CONSTRAINT PK_DEPT PRIMARY KEY (Deptno));
```

其中 CONSTRAINT U1 UNIQUE 表示约束名为U1，该约束要求Dname列值唯一。

ORACLE中用户定义的完整性（续）

例2： 建立学生登记表Student，要求学号在900000至999999之间，年龄<29，性别只能是‘男’或‘女’，姓名非空

```
CREATE TABLE Student
    (Sno    NUMBER(5)
            CONSTRAINT C1 CHECK
            (Sno BETWEEN 10000 AND 99999),
     Sname  VARCHAR(20) CONSTRAINT C2 NOT NULL,
     Sage   NUMBER(3) CONSTRAINT C3  CHECK (Sage < 29),
     Ssex   VARCHAR(2)
            CONSTRAINT C4 CHECK (Ssex IN ('男', '女'));
```

ORACLE中用户定义的完整性（续）

例3： 建立职工表EMP，要求每个职工的应发工资不得超过3000元。 应发工资实际上就是实发工资列Sal与扣除项Deduct之和。

```
CREATE TABLE EMP
  (Eno    NUMBER(4)
   Ename  VARCHAR(10),
   Job    VARCHAR(8),
   Sal    NUMBER(7,2),
   Deduct NUMBER(7,2)
   Deptno NUMBER(2),
   CONSTRAINTS C1 CHECK (Sal + Deduct <=3000));
```

2. 通过触发器来定义用户的完整性规则

- 定义其它的完整性约束时，需要用数据库触发器（Trigger）来实现。
- 数据库触发器：一类靠事务驱动的特殊过程
- 一旦由某个用户定义，任何用户对该数据的增、删、改操作均由服务器自动激活相应的触发子，在核心层进行集中的完整性控制
- 定义数据库触发器的语句

  CREATE [OR REPLACE] TRIGGER

ORACLE中用户定义的完整性（续）

例4： 为教师表Teacher定义完整性规则

“教授的工资不得低于800元，如果低于800元，自动改为800元”

ORACLE中用户定义的完整性（续）


```
CREATE TRIGGER UPDATE_SAL
   BEFORE INSERT OR UPDATE OF Sal, Pos ON Teacher
   FOR EACH ROW
   WHEN (:new.Pos='教授')
   BEGIN
        IF :new.sal<800
           THEN
            :new.Sal:=800;
        END IF;
   END;
```

Oracle完整性小结

- ORACLE提供定义完整性约束条件

  CREATE TABLE语句

  CREATE TRIGGER语句

  可以定义很复杂的完整性约束条件

- ORACLE自动执行相应的完整性检查

  对于违反完整性约束条件的操作:

  拒绝执行或者执行事先定义的操作

12.1 完整性约束条件

12.2 完整性控制

12.3 Oracle的完整性

12.4 小结

12.4 小结

- 数据库的完整性是为了保证数据库中存储的数据是正确的，所谓正确的是指符合现实世界语义的。

- DBMS完整性实现的机制
  - 完整性约束定义机制
  - 完整性检查机制
  - 违背完整性约束条件时DBMS应采取的动作

小结

- 完整性机制的实施会极大地影响系统性能
- 不同的数据库产品对完整性的支持策略和支持程度是不同的
  - 许多数据库管理系统对完整性机制的支持比对安全性的支持要晚得多也弱得多
  - 数据库厂商对完整性的支持越来越好，不仅在能保证实体完整性和参照完整性而且能在DBMS核心定义、检查和保证用户定义的完整性约束条件