数据库系统概论

第5章 数据库完整性

第5章 数据库完整性

- 第一节 概述
- 第二节 实体完整性
- 第三节 参照完整性
- 第四节 用户定义的完整性
- 第五节 完整性约束命名字句
- 第六节 域中的完整性限制(了解)
- 第七节 触发器

概述

- 数据库的完整性
  - 数据的正确性和相容性
- 数据的完整性和安全性是两个不同概念
  - 数据的完整性
    - 防止数据库中存在不符合语义的数据，也就是防止数据库中存在不正确的数据
    - 防范对象：不合语义的、不正确的数据
  - 数据的安全性
    - 保护数据库防止恶意的破坏和非法的存取
    - 防范对象：非法用户和非法操作

- 为维护数据库的完整性，DBMS必须
  - 提供定义完整性约束条件的机制
  - 提供完整性检查的方法
  - 违约处理

第5章 数据库完整性

- 第一节 概述
- **第二节 实体完整性**
- 第三节 参照完整性
- 第四节 用户定义的完整性
- 第五节 完整性约束命名字句
- 第六节 域中的完整性限制(了解)
- 第七节 触发器

实体完整性

- **实体完整性定义**
- 实体完整性检查和违约处理

实体完整性定义

- 关系模型的实体完整性
  - CREATE TABLE中用PRIMARY KEY定义
- 单属性构成的码有两种说明方法
  - 定义为列级约束条件
  - 定义为表级约束条件
- 对多个属性构成的码只有一种说明方法
  - 定义为表级约束条件

[例1] 将Student表中的Sno属性定义为码。

(1)在列级定义主码

```
CREATE TABLE Student
(Sno  CHAR(9)  PRIMARY KEY，
 Sname  CHAR(20) NOT NULL，
 Ssex  CHAR(2)，
 Sage  SMALLINT，
 Sdept  CHAR(20));
```

(2)在表级定义主码

```
CREATE TABLE Student
(Sno  CHAR(9)，
 Sname  CHAR(20) NOT NULL，
 Ssex  CHAR(2)，
 Sage  SMALLINT，
 Sdept  CHAR(20)
 PRIMARY KEY (Sno));
```

[例2] 将SC表中的Sno，Cno属性组定义为码。

```
CREATE TABLE SC
    (Sno   CHAR(9)  NOT NULL，
     Cno  CHAR(4)  NOT NULL，
     Grade    SMALLINT，
     PRIMARY KEY (Sno，Cno)     /*只能在表级定义主码*/
    );
```

实体完整性

- 实体完整性定义
- **实体完整性检查和违约处理**

实体完整性检查和违约处理

- 插入或对主码列进行更新操作时，RDBMS按照实体完整性规则自动进行检查。包括：
  - 1. 检查主码值是否唯一，如果不唯一则拒绝插入或修改
  - 2. 检查主码的各个属性是否为空，只要有一个为空就拒绝插入或修改

- 检查记录中主码值是否唯一的一种方法是进行**全表扫描**

待插入记录

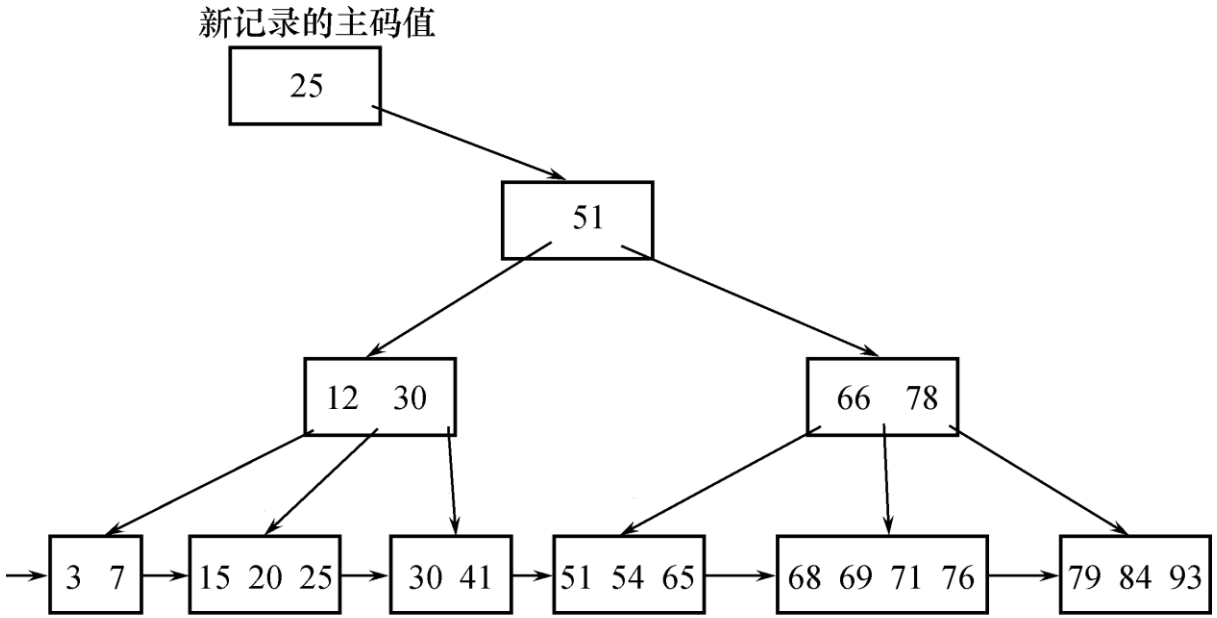
Keyi	F2i	F3i	F4i	F5i

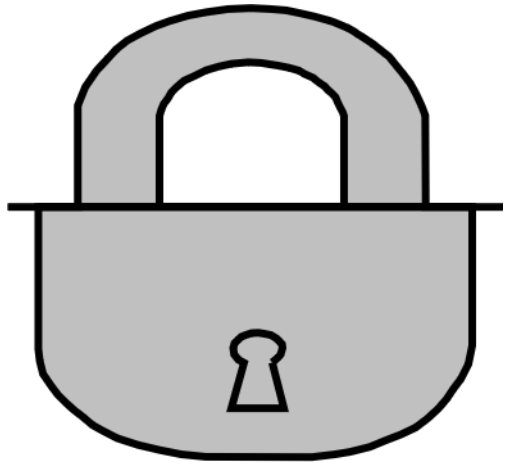
基本表

Key1	F21	F31	F41	F51
Key2	F22	F32	F42	F52
Key3	F23	F33	F43	F53
⋮				

索引

- 为了避免全表扫描，一般会在主码上建立索引





第5章 数据库完整性

- 第一节 概述
- 第二节 实体完整性
- **第三节 参照完整性**
- 第四节 用户定义的完整性
- 第五节 完整性约束命名字句
- 第六节 域中的完整性限制(了解)
- 第七节 触发器

参照完整性

- 参照完整性定义
- 参照完整性检查和违约处理

参照完整性定义

- 关系模型的参照完整性定义
  - 在CREATE TABLE中用FOREIGN KEY短语定义哪些列为外码
  - 用REFERENCES短语指明这些外码参照哪些表的主码

例如 关系SC中一个元组表示一个学生选修的某门课程的成绩，（Sno，Cno）是主码。

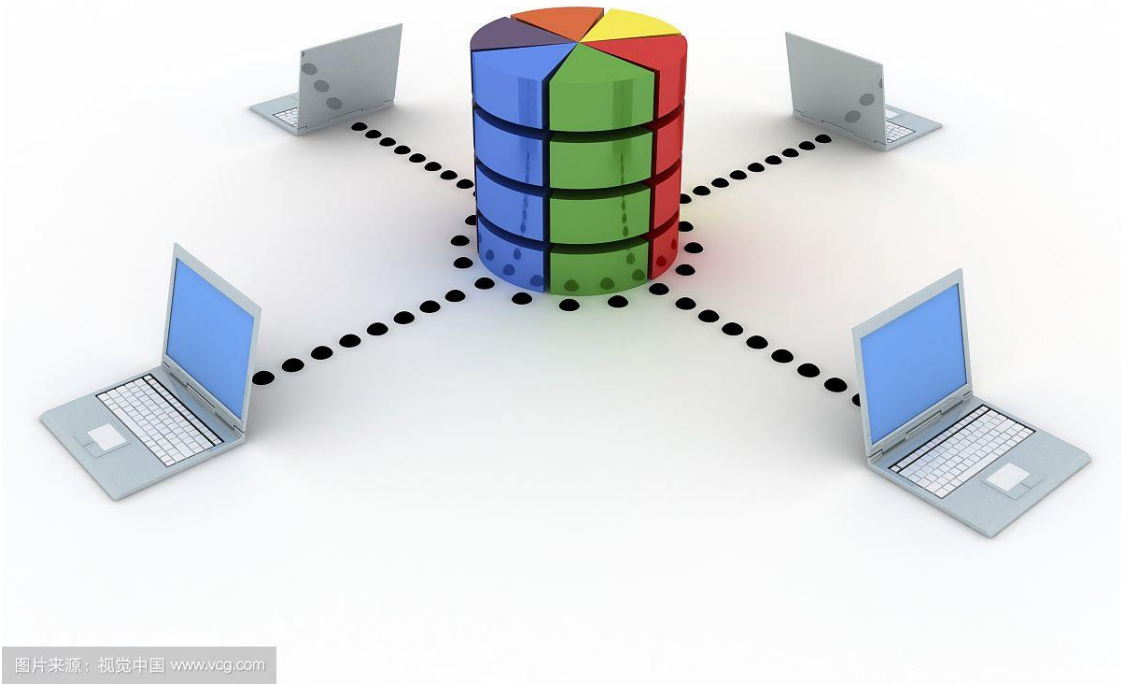
Sno，Cno分别参照引用Student表的主码和Course表的主码

[例3] 定义SC中的参照完整性。

```
CREATE TABLE SC
 ( Sno    CHAR(9)  NOT NULL，
   Cno     CHAR(4)  NOT NULL，
   Grade    SMALLINT，
   PRIMARY KEY (Sno， Cno)， /*在表级定义实体完整性*/
   FOREIGN KEY (Sno) REFERENCES  Student(Sno)，
   FOREIGN KEY (Cno) REFERENCES  Course(Cno)
 );
```

参照完整性

- 参照完整性定义
- **参照完整性检查和违约处理**



参照完整性检查和违约处理

可能破坏参照完整性的情况及违约处理

被参照表（例如Student）	参照表（例如SC）	违约处理
可能破坏参照完整性 ←	插入元组	拒绝
可能破坏参照完整性 ←	修改外码值	拒绝
删除元组 →	可能破坏参照完整性	拒绝/级连删除/设置为空值
修改主码值 →	可能破坏参照完整性	拒绝/级连修改/设置为空值

违约处理

❖参照完整性违约处理

- 1. 拒绝(NO ACTION)执行
  - 默认策略
- 2. 级联(CASCADE)操作
- 3. 设置为空值（SET-NULL）
  - 对于参照完整性，除了应该定义外码，还应定义外码列是否允许空值

[例4] 显式说明参照完整性的违约处理示例。

```
CREATE TABLE  SC
  (Sno   CHAR(9)  NOT NULL，
   Cno   CHAR(4)  NOT NULL，
   Grade  SMALLINT，
   PRIMARY KEY（Sno，Cno），
   FOREIGN KEY (Sno)  REFERENCES  Student(Sno)
        ON DELETE CASCADE     /*级联删除SC表中相应的元组*/
        ON UPDATE CASCADE，  /*级联更新SC表中相应的元组*/
   FOREIGN KEY (Cno)  REFERENCES  Course(Cno)
        ON DELETE NO ACTION
           /*当删除course 表中的元组造成了与SC表不一致时拒绝删除*/
        ON UPDATE CASCADE
           /*当更新course表中的cno时，  级联更新SC表中相应的元组*/
    );
```

第5章 数据库完整性

- 第一节 概述
- 第二节 实体完整性
- 第三节 参照完整性
- **第四节 用户定义的完整性**
- 第五节 完整性约束命名字句
- 第六节 域中的完整性限制(了解)
- 第七节 触发器

用户定义的完整性

- **属性上的约束条件的定义**
- 属性上的约束条件检查和违约处理
- 元组上的约束条件的定义
- 元组上的约束条件检查和违约处理

属性上的约束条件的定义

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

❖不允许取空值

[例5] 在定义SC表时，说明Sno、Cno、Grade属性不允许取空值。

```
CREATE TABLE SC
  Sno  CHAR(9)  NOT NULL，
  Cno  CHAR(4)  NOT NULL，
  Grade  SMALLINT NOT NULL，
  PRIMARY KEY (Sno， Cno)，  /* 如果在表级定义实体完整性，隐含了Sno，Cno不允许
                              取空值，则在列级不允许取空值的定义就不必写了 * /
）；
```

❖列值唯一

[例6] 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  NUMERIC(2)，
  Dname  CHAR(9)  UNIQUE，/*要求Dname列值唯一*/
  Location  CHAR(10)，
  PRIMARY KEY (Deptno)
 );
```

❖用CHECK短语指定列值应该满足的条件

[例7] Student表的Ssex只允许取“男”或“女”。

```
CREATE TABLE Student
    (Sno  CHAR(9) PRIMARY KEY，
     Sname CHAR(8) NOT NULL，
     Ssex  CHAR(2)  CHECK (Ssex  IN ('男', '女') )，/*性别属性Ssex只允许取'男'或'女' */
     Sage  SMALLINT，
     Sdept  CHAR(20)
    );
```

[例8]　SC表的Grade的值应该在0和100之间 。

```
CREATE TABLE SC
   (Sno  CHAR(9)  NOT NULL，
    Cno CHAR(4)  NOT NULL，
    Grade  SMALLINT CHECK(Grade >=0 AND Grade <=100)，
    PRIMARY KEY(sno , cno),
    FOREIGN (Sno) REFERENCES Student(sno),
    FOREIGN (Cno) REFERENCES Course(cno)
  );
```

用户定义的完整性

- 属性上的约束条件的定义
- **属性上的约束条件检查和违约处理**
- 元组上的约束条件的定义
- 元组上的约束条件检查和违约处理

属性上的约束条件检查和违约处理

- 插入元组或修改属性的值时，RDBMS检查属性上的约束条件是否被满足
- 如果不满足则操作被拒绝执行

用户定义的完整性

- 属性上的约束条件的定义
- 属性上的约束条件检查和违约处理
- **元组上的约束条件的定义**
- 元组上的约束条件检查和违约处理

元组上的约束条件的定义

- 在CREATE TABLE时可以用CHECK短语定义元组上的约束条件，即元组级的限制
- 同属性值限制相比，元组级的限制可以设置不同属性之间的取值的相互约束条件

[例9] 当学生的性别是男时，其名字不能以Ms.打头。

```
CREATE TABLE Student
 (Sno    CHAR(9)，
  Sname  CHAR(8) NOT NULL，
  Ssex    CHAR(2)，
  Sage   SMALLINT，
  Sdept  CHAR(20)，
  PRIMARY KEY (Sno)，
  CHECK (Ssex='女' OR Sname NOT LIKE 'Ms.%')
  /*定义了元组中Sname和 Ssex两个属性值之间的约束条件*/
);
```

✓性别是女性的元组都能通过该项检查，因为Ssex=‘女’成立；

✓当性别是男性时，要通过检查则名字一定不能以Ms.打头

用户定义的完整性

- 属性上的约束条件的定义
- 属性上的约束条件检查和违约处理
- 元组上的约束条件的定义
- **元组上的约束条件检查和违约处理**

元组上的约束条件检查和违约处理

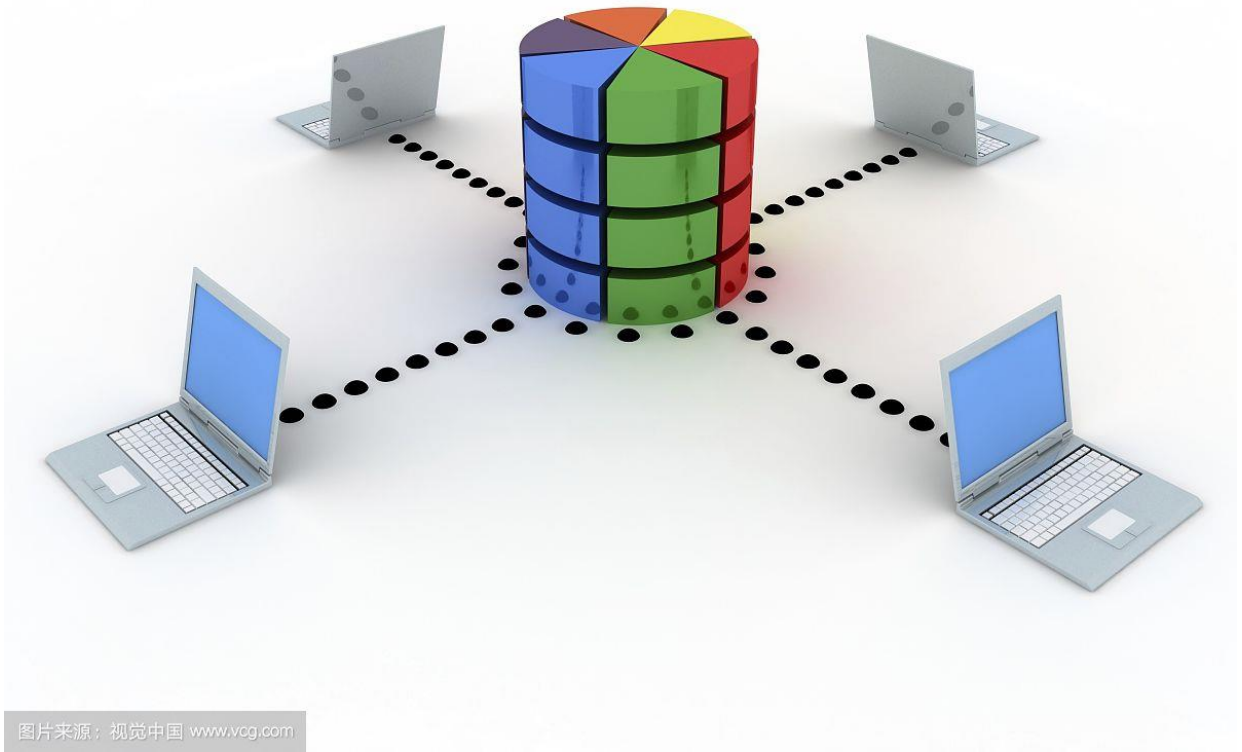
- 插入元组或修改属性的值时，RDBMS检查元组上的约束条件是否被满足
- 如果不满足则操作被拒绝执行

第5章 数据库完整性

- 第一节 概述
- 第二节 实体完整性
- 第三节 参照完整性
- 第四节 用户定义的完整性
- **第五节 完整性约束命名子句**
- 第六节 域中的完整性限制(了解)
- 第七节 触发器

❖完整性约束命名子句

❖修改表中的完整性限制



完整性约束命名子句

- CONSTRAINT 约束

CONSTRAINT <完整性约束条件名>

[PRIMARY KEY短语

|FOREIGN KEY短语

|CHECK短语]

[例10] 建立学生登记表Student，要求学号在90000~99999之间，姓名不能取空值，年龄小于30，性别只能是“男”或“女”。

```
CREATE TABLE Student
  (Sno  NUMERIC(6)
    CONSTRAINT C1 CHECK (Sno BETWEEN 90000 AND 99999)，
   Sname  CHAR(20)
    CONSTRAINT C2 NOT NULL，
   Sage  NUMERIC(3)
    CONSTRAINT C3 CHECK (Sage < 30)，
   Ssex  CHAR(2)
    CONSTRAINT C4 CHECK (Ssex IN ( '男'，'女'))，
    CONSTRAINT StudentKey PRIMARY KEY(Sno)
  );
```

- ✓ 在**Student**表上建立了**5**个约束条件，包括主码约束（命名为**StudentKey**）以及**C1**、**C2**、**C3**、**C4**四个列级约束。

[例11] 建立教师表Teacher，要求每个教师的应发工资 不低于3000元。(应发工资等于实发工资Sal和扣除项Deduct之和)

```
CREATE TABLE TEACHER
  (Eno  NUMERIC(4) PRIMARY KEY,
   Ename  CHAR(10) ,
   Job  char(8),
   Sal NUMERIC(7,2),
   Deduct NUMERIC(7,2),
   Deptno  NUMERIC(7,2),
   CONSTRAINT EMPFKEY FOREGIN KEY (Deptno) REFERENCES DEPT(Deptno)，
   CONSTRAINT C1 CHECK(Sal + Deduct >= 3000)
 );
```

✓ 在**Teacher**表上建立了外键约束。

完整性约束命名字句

- 完整性约束命名子句

- **修改表中的完整性限制**

修改表中的完整性限制

- 使用ALTER TABLE语句修改表中的完整性限制

[例12] 修改表Student中的约束条件，要求学号改为在900000~999999之间，年龄由小于30改为小于40 。

- 可以先删除原来的约束条件，再增加新的约束条件

```
ALTER TABLE Student
DROP CONSTRAINT C1;

ALTER TABLE Student
ADD CONSTRAINT C1 CHECK (Sno BETWEEN 900000 AND 999999),

ALTER TABLE Student
DROP CONSTRAINT C3;

ALTER TABLE Student
ADD CONSTRAINT C3 CHECK (Sage < 40);
```

吾日三省吾身。为人谋而不忠乎？与朋友交而不信乎？传不习乎？