厦门大学计算机科学系本科生课程

《数据库系统原理》

第五章 数据库完整性
（2017版）

林子雨

厦门大学计算机科学系

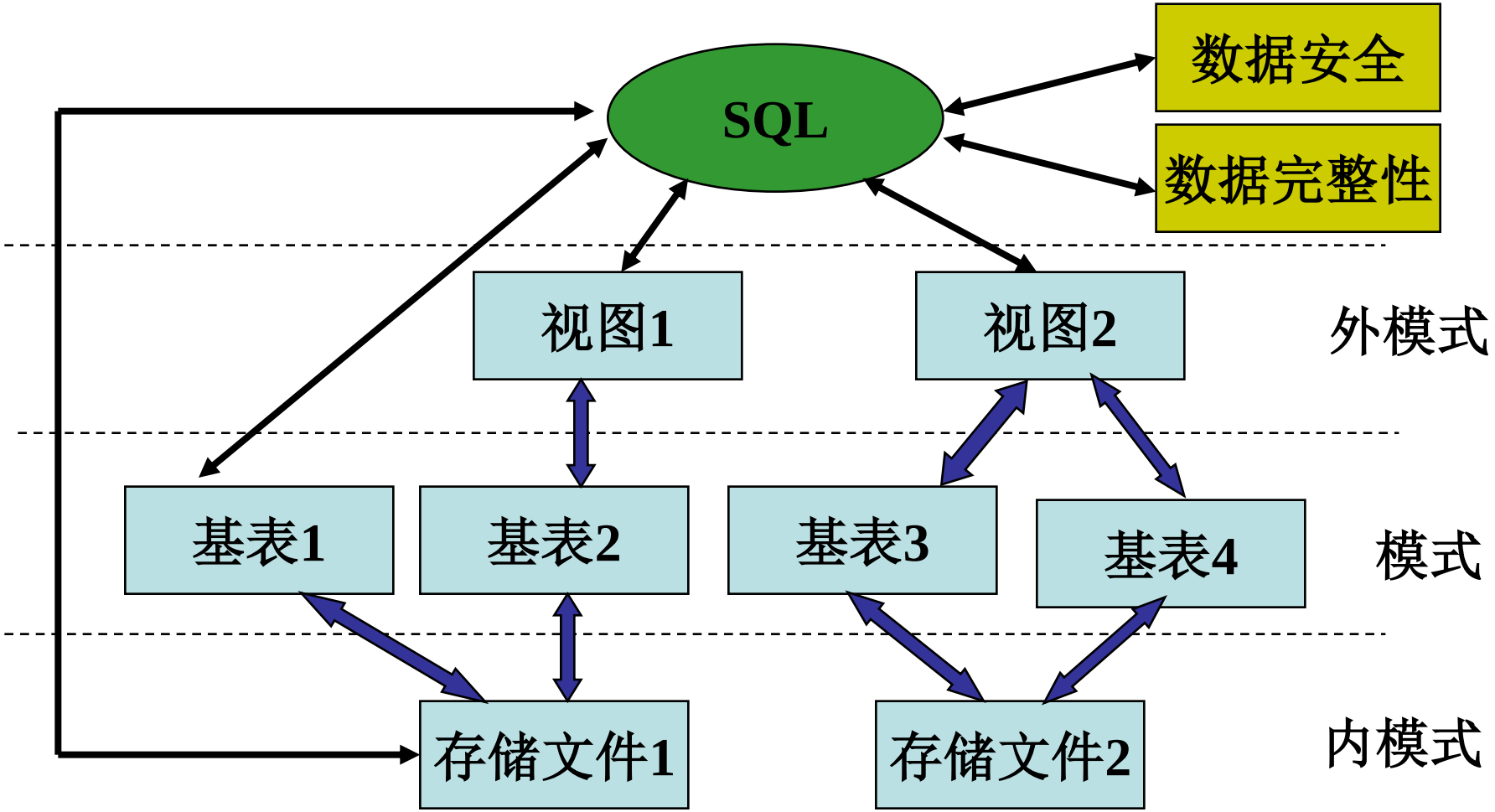
E-mail: ziyulin@xmu.edu.cn

主页：http://www.cs.xmu.edu.cn/linziyu

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

综合统一（操纵三级模式）



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：是否真实地反映现实世界

什么是完整性控制机制？

1. 完整性约束条件定义机制
2. 完整性检查机制
3. 违约处理

1、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
- 由**SQL**的**DDL**语句实现

2、完整性检查机制

- 检查用户发出的操作请求是否违背了完整性约束条件
- 在**INSERT、UPDATE、DELETE**语句执行时进行检查

3、违约处理

- 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。
- 拒绝(NO ACTION)、级连(CASCADE)

1.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

2.完整性约束条件分类

- **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

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.1 实体完整性定义

实体完整性：

实体完整性规则（**Entity Integrity**）

若属性**A**是基本关系**R**的主属性，则属性**A**不能取空值

5.1 实体完整性定义

- 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

5.1 实体完整性定义

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      CHAR(5)  primary key ,
       Sname     VARCHAR(10) ,
       Ssex      CHAR(2) ,
       Sage      INT,
       Sdept     CHAR(2));
```

5.1 实体完整性定义

例2：要在SC表中定义(Sno, Cno)为主码

```
CREATE TABLE SC
        (Sno      CHAR(5) ,
         Cno      CHAR(1) ,
         Grade    INT,
         primary key (Sno,Cno) );
```

5.1 实体完整性定义

- 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

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.2 参照完整性规则

参照完整性：

若属性（或属性组）***F***是基本关系***R***的外码，它与基本关系***S***的主码K_s相对应（基本关系***R***和***S***不一定是不同的关系），则对于***R***中每个元组在***F***上的值必须为：

- 或者取空值（***F***的每个属性值均为空值）
- 或者等于***S***中某个元组的主码值。

5.2 参照完整性规则

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执行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（续）

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（续）

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、在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、在被参照关系中删除元组时的问题（续）

- 违约反应：可有三种策略
  - 受限删除（**NO ACTION**）
  - 级联删除（**CASCADE**）
  - 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、在被参照关系中删除元组时的问题（续）

- **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

```
CREATE TABLE SC
      (Sno      CHAR(5)foreign key
     references Student(Sno),
       Cno      CHAR(1) ,
       Grade    INT
       );
```

2、在被参照关系中删除元组时的问题（续）

- **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

```
CREATE TABLE SC
       (Sno       CHAR(5)
foreign key references Student(Sno)
ON DELETE CASCADE,
        Cno      CHAR(1) ,
        Grade    INT
        );
```

2、在被参照关系中删除元组时的问题（续）

- **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。

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（续）

- **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（续）

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。

结论：参照完整性的执行

RDBMS在执行参照完整性时：

- 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

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.3 用户定义的完整性

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）

5.3 用户定义的完整性

约束命名

Constraint <约束名>

[**Primary key**… | **Foreign Key**… | **Check** …]

5.3 用户定义的完整性

修改约束

Alter Table <表名>

Add| Drop Constraint <约束名>

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.4 完整性约束命名子句

SQL还在CREATE TABLE 语句中提供了完整性约束命名子句CONSTRAINT，用来对完整性约束条件命名。从而可以灵活地增加、删除一个完整性约束条件。

1、完整性约束命名子句

CONSTRAINT <完整性约束条件名> [PRIMARY KEY 短语 | FOREIGN KEY 短语 | CHECK 短语]

2、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

讲课任务清单（1）

- 1.在student表中插入一条新记录(‘姚晨’,23)
- 2.把sc表的sno设置为外键（引用student表的sno），把sc表的cno设置为外键（引用course表的cno）
- 3. 再次在student表中插入一条新记录(‘李涛’,24)
- 4. 修改student表的一条记录，把学号95001修改为95020
- 5.修改sc表的一条记录，把学号95001修改为95020
- 6. 撤销course表的外键约束

讲课任务清单（2）

1．在数据库中建立一个test数据库，建立Student、Course和SC表
2、采用SQL语句，创建一个新表SC1，演示如何创建约束
3.创建命名的约束
4.修改Student表中的约束
5.约束的查询和删除
6.设置表SC的外键
7.演示删除和修改student表中的某个学号，对SC表的影响

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.6 触发器

通过触发器来定义复杂的完整性规则

- 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

5.6 触发器

CREATE TRIGGER <触发器名>
ON <表名>
FOR {**INSERT**| **DELETE**|**UPDATE**}
AS
<触发动作体>

删除
DROP TRIGGER <触发器名>

5.6 触发器

- 理解触发器里面的两个临时的表：Deleted , Inserted
- Deleted 与Inserted分别表示触发事件的表“旧的一条记录”和“新的一条记录”
- 一个数据库系统中有两个虚拟表用于存储在表中记录改动的信息

5.6 触发器

	虚拟表Inserted	虚拟表Deleted
在表记录新增时	存放新增的记录	不存储记录
修改时	存放用来更新的新记录	存放更新前的记录
删除时	不存储记录	存放被删除的记录

例. 学生成绩不低于60分,低于60分自动赋为60分

```
Create Trigger chggrade on SC  for insert
as
update SC set grade=60  where exists (select * from
inserted
where inserted.Sno=SC.Sno and inserted.Cno=SC.Cno
and inserted.grade < 60  )
```

课堂作业

计算机（sdept为'CS'）学生成绩不低于60分，低于60分自动赋为60分

课堂作业

计算机（sdept为'CS'）学生成绩不低于60分,低于60分自动赋为60分

```
create trigger T1
on sc
for insert
as
update sc set grade=60 where exists
(
select * from student
where sdept='CS'and student.sno=sc.sno and exists
(
select * from inserted
where inserted.sno=sc.sno and inserted.cno=sc.cno
and inserted.grade<60
))
```

例. 数学是必修课，所有学生都必须选，学生退学则将成绩置0

```
create Trigger selcou
on student
for insert
as
insert into SC(Sno,Cno)
select Sno, '2' from inserted
```

```
create Trigger sleft
on student for delete
as
update SC set Grade = 0
where Sno in
( select Sno from deleted)
```

课堂作业

（1）创建触发器，当更新student表中的学号时，也同时更新sc表中的学号

（2）创建一个触发器，当删除student表中的某条记录时，也同时删除sc表中的记录

（1）创建触发器，当更新student表中的学号时，也同时更新sc表中的学号

5.6 触发器

创建触发器，当更新student表中的学号时，也同时更新sc表中的学号

```
create trigger T2
on student
for update
as
if update(Sno)
begin
update sc set sc.sno=i.sno
from sc, inserted i,deleted d
 where sc.sno=d.sno
end
```

（2）创建一个触发器，当删除student表中的某条记录时，也同时删除sc表中的记录

5.6 触发器

创建一个触发器，当删除student表中的某条记录时，也同时删除sc表中的记录

```
create trigger T3
on student
for delete
as
delete sc from sc,deleted d
where sc.sno=d.sno
```

5.7 断言

- 使用断言（assertion）来指定更具一般性地约束
- 可以定义涉及多个表或聚集操作的比较复杂的完整性约束
- 任何对断言中所涉及关系的操作都会出发关系数据库管理系统对断言的检查
- 任何使得断言不为真的操作都会被拒绝执行

5.7 断言

- 语句格式：
- create assertion <断言名> <check子句>

5.7 断言

- 例：限制数据库课程最多60名学生选修

```
create assertion asse_sc_db_num
check (
60>=(
select count(*)  from course,sc
Where
Sc.cno=course.cno and course.cname='数据库'
))
```

5.7 断言

- 例：限制每一门课程最多60名学生选修

```
create assertion asse_sc_cnum1
Check(
60>=ALL(select count(*) from sc group by cno)
)
```

5.7 断言

- 例：限制每个学期每一门课最多60名学生选修

修改sc表，增加一个“学期(TERM)”属性

```
Create assertion asse_sc_cnum2
Check(
60>=ALL(select count(*) from sc group by
    cno,term)
)
```

第五章 数据库完整性

- 5.1 实体完整性
- 5.2 参照完整性
- 5.3 用户自定义完整性
- 5.4 完整性约束命名子句
- 5.6 触发器
- 5.7 小结

5.7 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

5.7 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

作业

- 第5版教材第173页　第6题

附录：本章常用SQL语句

```
alter table student alter column sno char(5) not null
alter table student add primary key (Sno)
alter table sc alter column sno char(5) not null
alter table sc add foreign key (Sno) references student(sno)
alter table course alter column Cno char(1) not null
alter table course add primary key (Cno)
alter table sc add foreign key (Cno) references course(Cno)
exec sp_helpconstraint course
sp_helpconstraint student
alter table sc drop constraint FK__SC__Sno__07020F21
alter table sc add constraint FK1 foreign key (Sno) references student(Sno)
alter table sc add constraint FK2 foreign key (Cno) references course(Cno)
alter table student add constraint PK1 primary key (Sno)
alter table student drop constraint PK1
```

附录：主讲教师

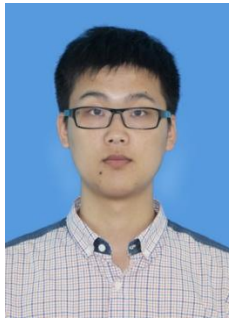
主讲教师：林子雨

单位：厦门大学计算机科学系
E-mail: ziyulin@xmu.edu.cn
个人网页：http://www.cs.xmu.edu.cn/linziyu
数据库实验室网站：http://dblab.xmu.edu.cn

扫一扫访问个人主页

林子雨，男，1978年出生，博士（毕业于北京大学），现为厦门大学计算机科学系助理教授（讲师），曾任厦门大学信息科学与技术学院院长助理、晋江市发展和改革局副局长。中国计算机学会数据库专业委员会委员，中国计算机学会信息系统专业委员会委员，荣获“2016中国大数据创新百人”称号。中国高校首个“数字教师”提出者和建设者，厦门大学数据库实验室负责人，厦门大学云计算与大数据研究中心主要建设者和骨干成员，2013年度厦门大学奖教金获得者。主要研究方向为数据库、数据仓库、数据挖掘、大数据、云计算和物联网，并以第一作者身份在《软件学报》《计算机学报》和《计算机研究与发展》等国家重点期刊以及国际学术会议上发表多篇学术论文。作为项目负责人主持的科研项目包括1项国家自然科学青年基金项目(No.61303004)、1项福建省自然科学青年基金项目(No.2013J05099)和1项中央高校基本科研业务费项目(No.2011121049)，同时，作为课题负责人完成了国家发改委城市信息化重大课题、国家物联网重大应用示范工程区域试点泉州市工作方案、2015泉州市互联网经济调研等课题。中国高校首个“数字教师”提出者和建设者，2009年至今，“数字教师”大平台累计向网络免费发布超过100万字高价值的研究和教学资料，累计网络访问量超过100万次。打造了中国高校大数据教学知名品牌，编著出版了中国高校第一本系统介绍大数据知识的专业教材《大数据技术原理与应用》，并成为京东、当当网等网店畅销书籍；建设了国内高校首个大数据课程公共服务平台，为教师教学和学生学习大数据课程提供全方位、一站式服务，年访问量超过50万次。具有丰富的政府和企业信息化培训经验，厦门大学管理学院EDP中心、浙江大学管理学院EDP中心、厦门大学继续教育学院、泉州市科技培训中心特邀培训讲师，曾给中国移动通信集团公司、福州马尾区政府、福建龙岩卷烟厂、福建省物联网科学研究院、石狮市物流协会、厦门市物流协会、浙江省中小企业家、四川泸州企业家、江苏沛县企业家等开展信息化培训，累计培训人数达3000人以上。

附录：课程助教



助教：魏亮

单位：厦门大学计算机科学系数据库实验室2016级硕士研究生
E-mail: 851189929@qq.com

助教：曾冠华

单位：厦门大学计算机科学系数据库实验室2016级硕士研究生
E-mail: 1393723173@qq.com

附录：班级网站

林子雨主讲《数据库系统原理》2017班级主页

http://dblab.xmu.edu.cn/post/7657/

扫一扫访问班级网站
支持手机浏览


Thank You!