数据库
安全性与完整性

1 数据库安全性

2 数据库完整性

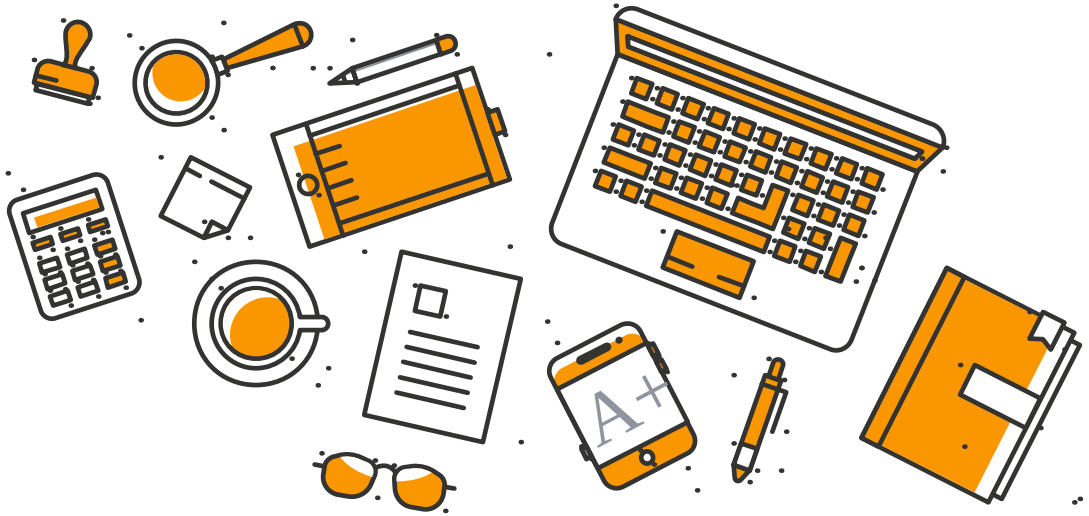
数据库完整性

- 为维护数据库的完整性，数据库管理系统提供如下功能：
  - 完整性约束条件定义机制
    - 完整性约束条件也称为完整性规则，是数据库中的数据必须满足的语义约束条件；
    - 由 SQL 的 DDL 实现，作为模式的一部分存入数据库中。
  - 完整性检查方法
    - 检查数据是否满足已定义的完整性约束条件称为完整性检查；
    - 一般在 insert 、 delete 、 update 执行后开始检查，或事务提交时进行检查。
  - 违约处理措施
    - 若发现用户操作违背了完整性约束条件，应采取一定的措施，如拒绝操作等。
- 商用 DBMS 都支持完整性控制。定义数据库模式时，都可以很明确地对完整性约束加以说明。

数据库完整性

1. 数据库完整性概述

2.SQL Server 完整性



1. 数据库完整性概述

- 完整性约束条件作用的对象可以是关系、元组、列三种：
  - 列约束：在定义属性的同时定义该属性应满足的约束条件，主要是定义属性的数据类型、取值范围和精度、默认值、是否允许空值、是否唯一、单一属性主码等约束条件；
  - 元组约束：定义元组中属性间的联系的约束条件；
  - 关系约束：定义若干元组间、关系集合上以及关系之间的联系的约束条件，亦称为表约束；在定义属性之后单独定义。
- 完整性约束，其状态可是静态的，也可是动态的。

1. 数据库完整性概述

- 用户可以为完整性约束命名，命名格式如下：

  [CONSTRAINT <*constraintName*>]
  PRIMARY KEY (<*constraintExpr*>)
  [CONSTRAINT <*constraintName*>]
  FOREIGN KEY (<*constraintExpr*>)
  REFERENCE <*refTable*>(<*constraintExpr*>)
  [CONSTRAINT <*constraintName*>]
  CHECK (<*constraintExpr*>)

- 用户命名有两点好处：
  - 一是便于理解约束的含义；
  - 二是修改约束方便，不必查询数据字典。
- 使用 ALTER TABLE 语句修改基本表中的完整性约束。
  - 要修改约束，必须先删除约束，然后加入新的约束。

2. SQL Server 完整性：实体完整性

- 实体完整性要求基本表的主码值唯一且不允许为空值。
- 在 SQL 中：
  - 实体完整性定义使用 CREATE TABLE 语句中的 PRIMARY KEY 短语实现；或使用 ALTER TABLE 语句中的 ADD PRIMARY KEY 短语实现；
  - 对单属性构成的主码可定义为列约束，也可定义为元组约束；
  - 对多个属性构成的主码，只能定义为元组约束（也有的书将其划归关系约束，即表约束）。

2. SQL Server 完整性：实体完整性

■实体完整性定义

[例 1] 在班级表 Class 中将 classNo 定义为主码。

```
CREATE TABLE Class (
    classNo      char(6)                    NOT NULL,    -- 班级号，列约束
    className    varchar(30)  UNIQUE        NOT NULL,    -- 班级名，列约束
    institute    varchar(30)                NOT NULL,    -- 所属学院，列约束
    grade        smallint   DEFAULT  0      NOT NULL,    -- 年级，列约束
    classNum     tinyint                     NULL,       -- 班级人数，列约束
    CONSTRAINT ClassPK PRIMARY KEY (classNo)             --  元组约束
)
```

- 本例定义 classNo 为主码，使用 CONSTRAINT 短语为该约束命名为 ClassPK ；
- 该主码定义为元组约束 (也有将其划归关系约束的) 。

2. SQL Server 完整性：实体完整性

■ 该例还可按下面的方式定义：

```
CREATE TABLE Class (
        classNo  char(6)    NOT NULL PRIMARY KEY,   -- 班级号，列约束
        ...
)
```

- 将主码 classNo 定义为列约束，且由系统取约束名称。

■ 可为约束取名，如：

```
CREATE TABLE Class (
      classNo    char(6)          NOT NULL
            CONSTRAINT ClassPK PRIMARY KEY,      -- 班级号，列约束
      ...
)
```

- 将主码 classNo 定义为列约束，且约束取名为 ClassPK 。

2. SQL Server 完整性：实体完整性

■实体完整性的检查和违约处理

- 当插入或对主码列进行更新操作时，数据库管理系统按照实体完整性规则自动进行检查，包括：
  - ➢检查主码值是否唯一，如果不唯一则拒绝插入或修改；
  - ➢检查主码的各个属性是否为空，只要有一个为空则拒绝插入或修改。

2. SQL Server 完整性：参照完整性

- 参照完整性为若干个基本表中的相应元组建立联系。
- 在 SQL 中：
  - 参照完整性定义使用 CREATE TABLE 语句中的 FOREIGN KEY 和 REFERENCES 短语来实现；或通过使用 ALTER TABLE 语句中的 ADD FOREIGN KEY 短语来实现；
  - FOREIGN KEY 指出定义哪些列为外码；
  - REFERENCES 短语指明这些外码参照哪些关系；
  - 给出 FOREIGN KEY 定义的关系称为参照关系；
  - 由 REFERENCES 指明的基本表称为被参照关系。

2. SQL Server 完整性：参照完整性

[例 3] 在学生成绩表 Score 中分别将 studentNo, courseNo, termNo 定义为外码。

```
CREATE TABLE Score (
    studentNo   char(7)                    NOT NULL,      -- 学号，列约束
    courseNo    char(3)                    NOT NULL,      -- 课程号，列约束
    termNo      char(3)                    NOT NULL,      -- 学期号，列约束
    score   numeric(5, 1)   DEFAULT  0   NOT NULL
          CHECK ( score BETWEEN 0.0 AND 100.0 ),     -- 成绩，列约束
    /*  主码由 3 个属性构成，必须作为元组约束进行定义  */
    CONSTRAINT ScorePK PRIMARY KEY (studentNo, courseNo, termNo)
    /*  外码约束只能定义为表约束，studentNo 是外码，被参照表是 Student */
    CONSTRAINT ScoreFK1 FOREIGN KEY (studentNo)
          REFERENCES Student (studentNo),                    --  表约束
    /*  外码约束只能定义为表约束，  courseNo 是外码，被参照表是 Course */
    CONSTRAINT ScoreFK2 FOREIGN KEY (courseNo)
          REFERENCES Course (courseNo),                      --  表约束
    /*  外码约束只能定义为表约束，  termNo 是外码，被参照表是 Term */
    CONSTRAINT ScoreFK1 FOREIGN KEY (studentNo)
          REFERENCES Term (studentNo)                        --  表约束
)
```

2. SQL Server 完整性：参照完整性

- 本例中， Score 为参照表， Student 、 Course 和 Term 为被参照表。
- Score 中 studentNo 列参照 Student 的 studentNo 列，其含义为：
  - Score 中 studentNo 列的取值必须是 Student 中 studentNo 列的某个属性值；
  - 即不存在一个未注册的学生选修了课程。
- Score 中 courseNo 列参照 Course 的 courseNo 列，其含义为：
  - Score 中 courseNo 列的取值必须是 Course 中 courseNo 列的某个属性值；
  - 即不存在学生选修了一门不存在的课程。

[例 3] 在学生成绩表Score中分别将studentNo, courseNo, termNo定义为外码

```
CREATE TABLE Score (
    studentNo   char(7)                 NOT NULL,     --学号，列约束
    courseNo    char(3)                 NOT NULL,     --课程号，列约束
    termNo      char(3)                 NOT NULL,     --学期号，列约束
    score   numeric(5, 1)   DEFAULT  0   NOT NULL
          CHECK ( score BETWEEN 0.0 AND 100.0 ),     --成绩，列约束
    /* 主码由3个属性构成，必须作为元组约束进行定义 */
    CONSTRAINT ScorePK PRIMARY KEY (studentNo, courseNo, termNo)
    /* 外码约束只能定义为表约束，studentNo是外码，被参照表是Student */
    CONSTRAINT ScoreFK1 FOREIGN KEY (studentNo)
          REFERENCES Student (studentNo),                -- 表约束
    /* 外码约束只能定义为表约束， courseNo是外码，被参照表是Course */
    CONSTRAINT ScoreFK2 FOREIGN KEY (courseNo)
          REFERENCES Course (courseNo),                  -- 表约束
    /* 外码约束只能定义为表约束， termNo是外码，被参照表是Term */
    CONSTRAINT ScoreFK1 FOREIGN KEY (studentNo)
          REFERENCES Term (studentNo)                    -- 表约束
)
```

2. SQL Server 完整性：参照完整性

学生表（被参照关系）

id	name	…
1	张三	
2	李四	
3	王五	

选课表（参照关系）

id	coursei	grade
1	11	100
2	12	90
2	11	80

- 在实现参照完整性时，
  - 如果外码是主码的一
  - 本例的外码皆不允许

修改，插入

■ 参照完整性的检查和违约处理

- 违约处理的策略如下：
  - 拒绝 (NO ACTION) 执行，是系统的默认策略：
    - ✓当在被参照关系中删除元组时，仅当参照关系中没有任何元组的外码值与被参照关系中要删除元组的主码值相同时，系统才执行删除操作，否则拒绝此操作。
    - ✓参照关系中可以随意删除元组。

2. SQL Server 完整性：参照完整性

- 级联 (CASCADE) 操作。当删除或修改被参照关系的某些元组造成了与参照关系的不一致时，则自动级联删除或修改参照关系中所有不一致的元组。

学生表（被参照关系）

id	name	…
1	张三	
2	李四	
3	王五	

选课表（参照关系）

id	coursei	grade
1	11	100
2	12	90
2	11	80

- 级联 (CASCADE) 操作必须在定义外码时给出显式定义。
- 设置为空值 (SET NULL)：
  - ✓ 对于参照完整性，除了定义外码，还应定义外码列是否允许空值。
  - ✓ 如果外码是主码的一部分，则外码不允许为空值。
- 置空值删除 (NULLIFIES)：
  - ✓ 删除被参照关系的元组，并将参照关系中相应元组的外码值置空值。

2. SQL Server 完整性：参照完整性

[例 4] 在学生成绩表 Score 中分别将 studentNo, courseNo, termNo 定义为外码，且 studentNo 外码定义为级联删除和修改操作， courseNo 外码定义为级联修改操作。

```
CREATE TABLE Score (
 studentNo   char(7)                          NOT NULL,     -- 学号，列约束
 courseNo    char(3)                          NOT NULL,     -- 课程号，列约束
 termNo      char(3)                          NOT NULL,     -- 学期号，列约束
 score   numeric(5, 1)   DEFAULT  0    NOT NULL,     -- 成绩，列约束
    /*  主码由 3 个属性构成，必须作为元组约束进行定义   */
    CONSTRAINT ScorePK PRIMARY KEY (studentNo, courseNo, termNo)
 /*  外码约束只能定义为表约束 ，  studentNo 是外码，  被参照表是 Student */
 CONSTRAINT ScoreFK1 FOREIGN KEY (studentNo)
        REFERENCES Student (studentNo)
        ON DELETE CASCADE     /*  级联删除参照关系 Score 中相应的元组   */
        ON UPDATE CASCADE,    /*  级联修改参照关系 Score 中相应的元组   */
```

2. SQL Server 完整性：参照完整性

```
/* 外码约束只能定义为表约束， courseNo 是外码，被参照表是 Course */
CONSTRAINT ScoreFK2 FOREIGN KEY (courseNo)
    REFERENCES Course (courseNo)
    ON DELETE NO ACTION          -- 该定义为默认值，可以不定义
    -- 当修改被参照表 Course 中的 courseNo 时，级联修改参照关系 Score 中相应的元组
    ON UPDATE CASCADE
)
```

2. SQL Server 完整性：自定义完整性

- 用户自定义完整性就是定义某一具体应用中数据必须满足的语义要求，由RDBMS 提供，而不必由应用程序承担。
- 用户自定义完整性包括属性上的约束和元组上的约束两种。
- 属性上的约束——列约束
  - 属性上的约束包括：数据类型、列值非空、列值唯一、设置默认值、满足CHECK (<*predicate*>) 定义等；
  - 属性上的约束：当向基本表中插入或修改属性值时，系统检查是否满足约束条件，若不满足，则拒绝相应的操作。

2. SQL Server 完整性：自定义完整性

[例 5] 创建学生表 Stud，属性及要求为：学号 studNo 为 5 位字符，且第 1 位为字母 D, M 或 U，其他 4 位为数字，主码，不允许为空值；姓名 studName 为 12 位字符，不允许为空值，且值必须唯一；性别 sex 为 2 位字符，允许为空值，但值只能取 '男' /'女'；年龄 age 为整型，允许为空值，缺省值为 16，取值范围 (0, 60)；民族 nation 为变长 20 位字符，允许为空值，缺省值为 '汉族'。

```
CREATE TABLE Stud (
  -- 学号，列约束：不允许为空值；第 1 位为字母 D, M 或 U，其他 4 位为数字，约束名为 sNoCK
  studNo      char(5)                NOT NULL
      CONSTRAINT sNoCK CHECK (studNo LIKE '[D,M,U][0-9][0-9][0-9][0-9]'),
  -- 姓名，列约束：不允许为空值；取值必须唯一
  studName   char(12)   UNIQUE        NOT NULL,
  -- 性别，列约束：允许为空值，仅取男或女两个值
  sex            char(2)              NULL   CHECK ( sex IN ( '男','女') ),
  -- 年龄，列约束：允许为空值，默认值为 16；取值范围 (0, 60)，约束名为 ageCK
  age            tinyint    DEFAULT  16    NULL
      CONSTRAINT ageCK CHECK ( age > 0 AND age < 60 ),
  nation   varchar(20)  DEFAULT '汉族'  NULL,   -- 民族，允许空值，默认值为汉族
  CONSTRAINT StudPK PRIMARY KEY (studNo)    --  定义主码，元组约束
)
```

2. SQL Server 完整性：自定义完整性

■ 元组上的约束——元组约束

- 元组上的约束可以设置不同属性之间的取值的相互约束条件；
- 用短语 CHECK (<*predicate*>) 引出的约束；
- 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，若不满足，则拒绝相应的操作。

[例 6] 在学生表 Stud 中定义：如果是男同学，则其姓名不能以刘开头。

```
CREATE TABLE Stud (
……
/* 性别，允许为空值，仅取男或女两个值 */
sex        char(2)        NULL  CHECK ( sex IN ('男','女') ),        -- 列约束
……                                                                  -- 以下是元组约束
CONSTRAINT SexCK CHECK (sex='女' OR studName NOT LIKE '刘%'),
……
)
```

2. SQL Server 完整性：修改完整性约束

■ 删除约束：

ALTER TABLE <*tableName*>

DROP CONSTRAINT <*constraintName*>

■ 添加约束：

ALTER TABLE <*tableName*>

ADD CONSTRAINT <*constraintName*>

< CHECK | UNIQUE | PRIMARY KEY | FOREIGN KEY >

(<*constraintExpr*>)

■ 其中

- <*tableName*> 为欲修改约束所在的基本表的名称；
- <*constraintName*> 为欲修改的约束的名称。

2. SQL Server 完整性：修改完整性约束

[例 8] 在例 27 的基础上，修改基本表 Stud 中的约束条件，要求学号改为在 15001 ~ 25999 之间，年龄由 (0, 60) 之间修改为 [15, 50] 之间。

- 首先，删除已经存在的约束：

```
ALTER TABLE Stud
    DROP CONSTRAINT sNoCK
ALTER TABLE Stud
    DROP CONSTRAINT ageCK
```

- 然后，添加修改后的约束：

```
ALTER TABLE Stud
    ADD CONSTRAINT sNoCK
        CHECK ( studNo BETWEEN '15001' AND '25999' )
ALTER TABLE Stud
    ADD CONSTRAINT ageCK
        CHECK ( age >= 15 AND age <= 50 )
```

- 对于复杂的完整性约束要求，必须通过触发器来实现。

小结

本小节主要讲述了数据库完整性的问题。

包括数据完整性概述和 SQL 的完整性机制实现等。