浙江大學
ZHEJIANG UNIVERSITY

实验 3　SQL 数据完整性

2024-2025 春夏学期 数据库系统

课程实验报告

姓名　王浩雄

学号　3230106032

年级　2023 级

专业　混合班（计算机科学与技术）

班级　混合 2303 班

2025 年 3 月 12 日

实验 3　SQL 数据完整性

1　实验综述

1.1　实验目的

1. 熟悉通过 SQL 进行数据完整性控制的方法。

1.2　实验内容

1. 定义若干表，其中包括 primary key, foreign key 和 check 的定义。
2. 表中插入数据，考察 primary key 如何控制实体完整性。
3. 删除被引用表中的行，考察 foreign key 中 on delete 子句如何控制参照完整性。
4. 修改被引用表中的行的 primary key，考察 foreign key 中 on update 子句如何控制参照完整性。
5. 修改或插入表中数据，考察 check 子句如何控制校验完整性。
6. 定义一个 trigger，并通过修改表中数据考察触发器如何起作用。
7. 完成实验报告。

2　实验环境

- 操作系统：
  Windows 11 Pro 24H2（64 位操作系统，基于 x64 的处理器）
- DBMS 版本：
  SQL Server Developer（64-bit）v16.0.1135.2
  SQL Server Management Studio v20.2.30.0

3　定义表

使用下述 SQL 代码，建立大学（University）数据库所涉及的多种表，其中包括 primary key, foreign key 和 check 的约束。

1. 部门表——departments

```sql
CREATE TABLE departments (
department_id INT NOT NULL,
department_name NVARCHAR(50) NOT NULL,
PRIMARY KEY (department_id),
);
```

2. 学生表——students

```sql
CREATE TABLE students (
student_id INT NOT NULL,
student_name NVARCHAR(50) NOT NULL,
gender NVARCHAR(10) CHECK (gender IN ('Male', 'Female', 'Other')
    ),
birth_date DATE,
phone_number NVARCHAR(15),
department_id INT,
PRIMARY KEY (student_id),
FOREIGN KEY (department_id) REFERENCES departments(department_id
    ) ON DELETE SET NULL ON UPDATE CASCADE
);
```

3. 教师表——teachers

```sql
CREATE TABLE teachers (
teacher_id INT NOT NULL,
teacher_name NVARCHAR(50) NOT NULL,
gender NVARCHAR(10) CHECK (gender IN ('Male', 'Female', 'Other')
    ),
birth_date DATE,
phone_number NVARCHAR(15),
department_id INT,
PRIMARY KEY (teacher_id),
FOREIGN KEY (department_id) REFERENCES departments(department_id
    ) ON DELETE SET NULL ON UPDATE CASCADE
);
```

4. 开课信息表——courses

```
CREATE TABLE courses (
course_id INT NOT NULL,
course_name NVARCHAR(100) NOT NULL,
credits FLOAT(1) NOT NULL,
department_id INT,
teacher_id INT,
PRIMARY KEY (course_id),
FOREIGN KEY (department_id) REFERENCES departments(department_id
    ) ON DELETE SET NULL,
FOREIGN KEY (teacher_id) REFERENCES teachers(teacher_id) ON
    DELETE SET NULL
);
```

5. 学生选课表——student_courses

```
CREATE TABLE student_courses (
student_id INT NOT NULL,
course_id INT NOT NULL,
FOREIGN KEY (student_id) REFERENCES students(student_id) ON
    DELETE CASCADE,
FOREIGN KEY (course_id) REFERENCES courses(course_id) ON DELETE
    CASCADE
);
```

6. 学生成绩表——grades

```
CREATE TABLE grades (
grade_id INT NOT NULL,
student_id INT NOT NULL,
course_id INT NOT NULL,
grade INT CHECK (grade BETWEEN 0 AND 100),
PRIMARY KEY (grade_id),
FOREIGN KEY (student_id) REFERENCES students(student_id) ON
    DELETE CASCADE,
FOREIGN KEY (course_id) REFERENCES courses(course_id) ON DELETE
    CASCADE
);
```

7. 导学关系表——student_advisor

```sql
CREATE TABLE student_advisor (
relationship_id INT NOT NULL,
student_id INT NOT NULL,
advisor_id INT NOT NULL,
PRIMARY KEY (relationship_id),
FOREIGN KEY (student_id) REFERENCES students(student_id) ON
    DELETE CASCADE,
FOREIGN KEY (advisor_id) REFERENCES teachers(teacher_id) ON
    DELETE CASCADE
);
```

建立的数据库关系如下图所示：

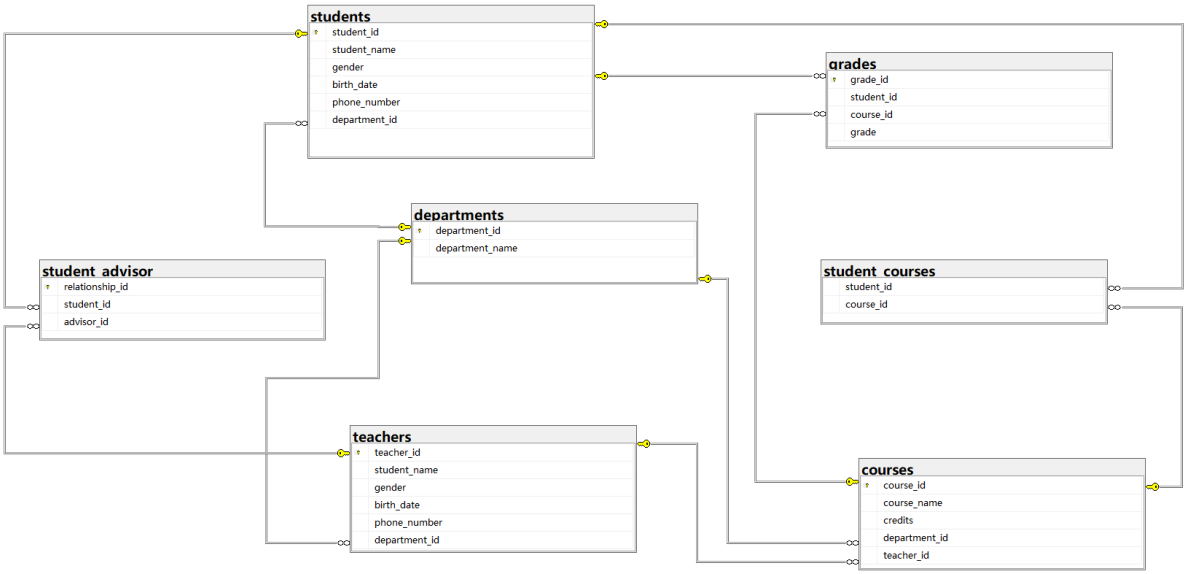


图 1: 数据库关系图

在上述表的建立语句中，我使用了如下特性，以确保数据库的信息完整与维护便利：

1. 为外键设置级联删除操作，允许在被引用对象被删除时，自动为引用对象进行删除/置空相关项的操作。

```sql
FOREIGN KEY (student_id) REFERENCES students(student_id) ON
    DELETE CASCADE,      // 级联删除
FOREIGN KEY (student_id) REFERENCES students(student_id) ON
    DELETE NULL          // 级联置空
```

2. 为外键设置级联更新操作，允许在被引用对象更新时，自动为引用对象进行相关属性的更新（本数据库模型没有使用级联更新的实际需求，使用级联更新是出于对该功能的学习目的）。

```
FOREIGN KEY (department_id) REFERENCES departments(department_id
    ) ON DELETE SET NULL ON UPDATE CASCADE
```

3. 使用 CHECK 子句设置规则检查，限制 gender 字段、grade 的填写内容。

```
gender NVARCHAR(10) CHECK (gender IN ('Male', 'Female', 'Other')
    ),
grade INT CHECK (grade BETWEEN 0 AND 100)
```

4　数据操作

4.1　插入数据

由于主键具有唯一性约束和非空约束，因此在插入的新数据，主键必须唯一且非空。

执行下述 SQL 操作，试图插入一条主键为空的数据：

```
INSERT INTO departments(department_id,department_name)
VALUES (null,'计算机科学与技术学院');
```

运行结果：

执行下述 SQL 操作，试图插入两条主键重复的数据：

```
INSERT INTO departments(department_id,department_name)
VALUES (1,'计算机科学与技术学院');
INSERT INTO departments(department_id,department_name)
VALUES (1,'控制科学与工程学院');
```

运行结果：

4.2　删除数据

执行下述 SQL 语句，删除 departments 表中的一条记录。按照 ON DELETE SET NULL 规则，参照表 students 中涉及数据的 department_id 字段将被置为 NULL。

```
DELETE FROM departments
WHERE department_id=1;
```

运行结果（students 表）：

	student_id	student_name	gender	birth_date	phone_number	department_id
1	1	王浩雄	male	2004-11-02	15703396384	1
2	2	张小明	female	2005-08-20	18299651234	1
3	3	李小华	male	2004-08-11	19513489474	2

（运行前）

	student_id	student_name	gender	birth_date	phone_number	department_id
1	1	王浩雄	male	2004-11-02	15703396384	NULL
2	2	张小明	female	2005-08-20	18299651234	NULL
3	3	李小华	male	2004-08-11	19513489474	2

（运行后）

执行下述 SQL 语句，删除 courses 表中的一条记录。按照 ON DELETE CASCADE 规则，参照表 student_courses 中涉及的数据将被整行删除。

```
DELETE FROM courses
WHERE course_id=1;
```

运行结果（student_courses 表）：

	student_id	course_id
1	1	1
2	1	2
3	2	1
4	2	3
5	3	3

（运行前）

	student_id	course_id
1	1	2
2	2	3
3	3	3

（运行后）

4.3　修改数据

执行下述 SQL 语句，修改 departments 表中一条记录的 department_id。按照 ON UPDATE CASCADE 规则，参照表 students 中涉及数据的 department_id 字段将被置为新值。

```
UPDATE departments
SET department_id=5
WHERE department_id=1;
```

运行结果（students 表）：

	student_id	student_name	gender	birth_date	phone_number	department_id
1	1	王浩雄	male	2004-11-02	15703396384	1
2	2	张小明	female	2005-08-20	18299651234	1
3	3	李小华	male	2004-08-11	19513489474	2

（运行前）

	student_id	student_name	gender	birth_date	phone_number	department_id
1	1	王浩雄	male	2004-11-02	15703396384	5
2	2	张小明	female	2005-08-20	18299651234	5
3	3	李小华	male	2004-08-11	19513489474	2

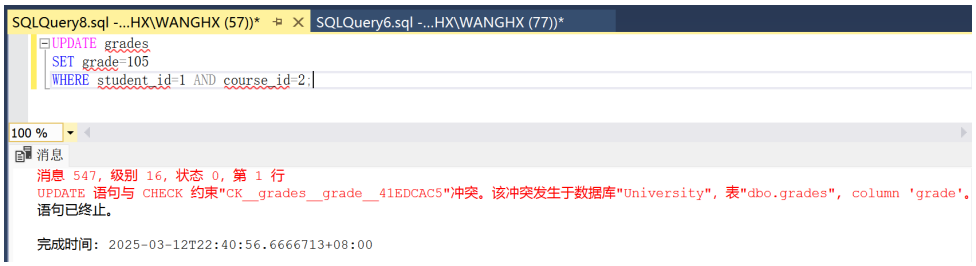
（运行后）

4.4　CHECK 子句校验完整性

执行下述 SQL 语句，修改 grades 表中一条记录的 grade。按照 CHECK (grade BETWEEN 0 AND 100) 规则，下述的修改将出现错误。

```
UPDATE grades
SET grade=105
WHERE student_id=1 AND course_id=2;
```

运行结果：



5　触发器

5.1　定义触发器

使用下述 SQL 代码，为 students 表建立一个日志表 student_audit_log；并定义一个触发器，用于在 students 表发生数据更新、删除、插入时，将相应的行为记录在日志表中。

1. 定义日志表

```sql
CREATE TABLE student_audit_log (
log_id INT IDENTITY(1, 1) PRIMARY KEY,
operation_type NVARCHAR(10) NOT NULL, -- 操作类型: INSERT /
   UPDATE / DELETE
student_id INT NOT NULL,              -- 被操作的学生 ID
student_name NVARCHAR(50),            -- 学生姓名
operation_time DATETIME DEFAULT GETDATE() -- 操作时间
);
```

2. 定义插入操作的触发器

```sql
CREATE TRIGGER trg_students_insert
ON students
AFTER INSERT
AS
BEGIN
        INSERT INTO student_audit_log (operation_type,
           student_id, student_name)
        SELECT 'INSERT', inserted.student_id, inserted.
           student_name
        FROM inserted;
END;
```

3. 定义更新操作的触发器

```sql
CREATE TRIGGER trg_students_update
ON students
AFTER UPDATE
AS
BEGIN
        INSERT INTO student_audit_log (operation_type,
           student_id, student_name)
        SELECT 'UPDATE', inserted.student_id, inserted.
           student_name
        FROM inserted;
END;
```

4. 定义删除操作的触发器

```sql
CREATE TRIGGER trg_students_delete
ON students
AFTER DELETE
AS
BEGIN
        INSERT INTO student_audit_log (operation_type,
           student_id, student_name)
        SELECT 'DELETE', deleted.student_id, deleted.
           student_name
        FROM deleted;
END;
```

5.2　测试触发器的作用

执行下述 SQL 操作，随后观察日志表 student_audit_log 的记录内容。

```sql
INSERT INTO students (student_id, student_name, gender, birth_date,
   phone_number)
VALUES (4, '李大三', 'Male', '2000-01-01', '14522367542');

UPDATE students
SET student_name = '张小三'
WHERE student_id = 4;

DELETE FROM students
WHERE student_id = 4;
```

运行结果如下，表明触发器正确触发和工作。

	log_id	operation_type	student_id	student_name	operation_time
1	1	INSERT	4	李大三	2025-03-12 23:00:59.977
2	2	UPDATE	4	张小三	2025-03-12 23:00:59.987
3	3	DELETE	4	张小三	2025-03-12 23:00:59.993