**交互式SQL**

**一、实验目的**

本次实验旨在通过实际操作，熟悉使用SQL语言对关系型数据库进行数据定义（DDL）和数据操作（DML），包括表的创建、修改和删除，索引的创建和删除，以及数据的插入、修改、删除和各类查询操作（单表查询、连接查询、嵌套查询、集合查询）。

**二、实验环境**

数据库管理系统：SQL Server

交互查询工具：SQL Server Management Studio(SSMS)

**三、实验数据库**

本次实验使用的数据库是自行设计的图书管理系统数据库。该数据库包含以下表：

users：存储系统用户账号、密码和身份。

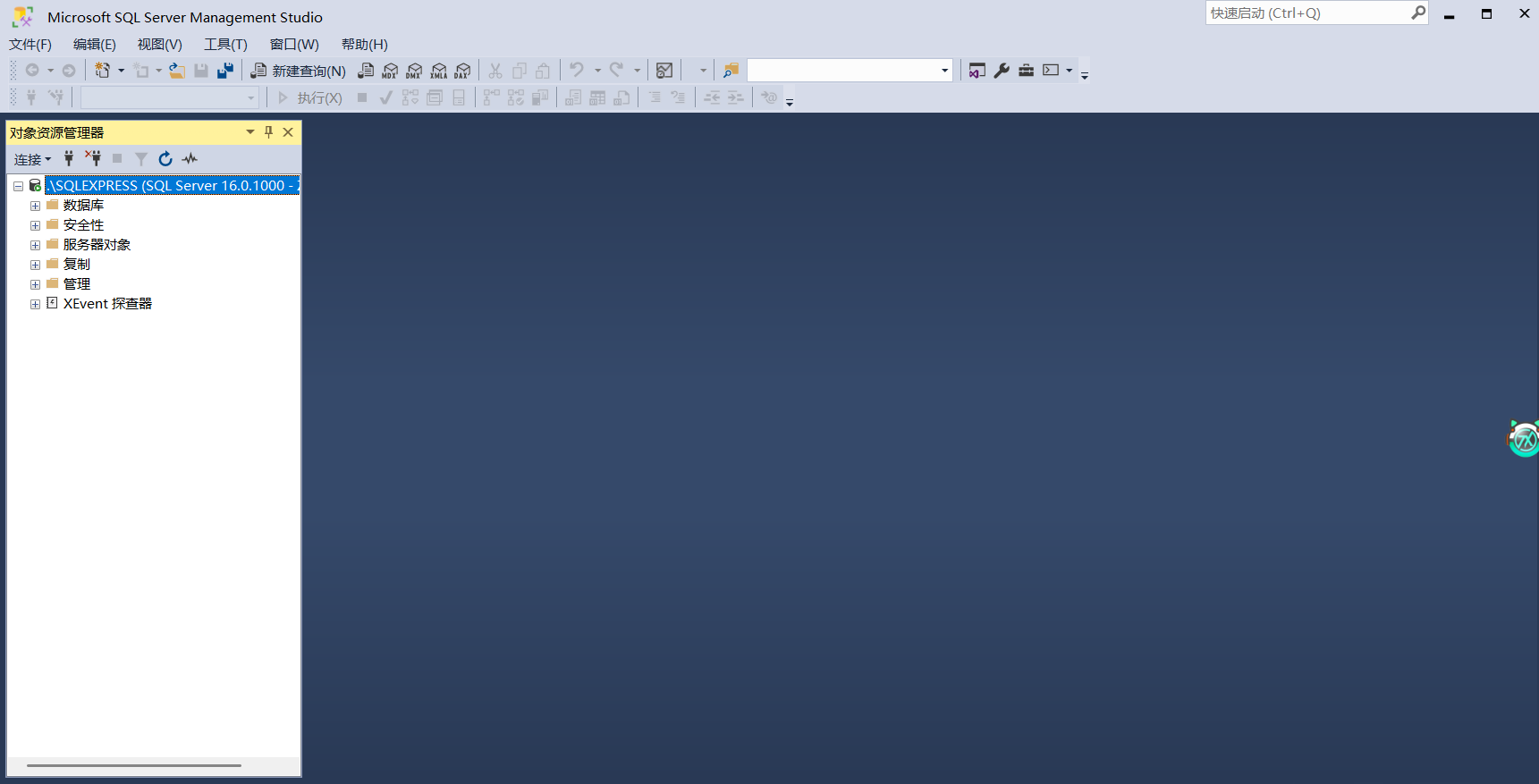
readers：存储读者详细信息，通过账号关联到users表。

books：存储图书信息。

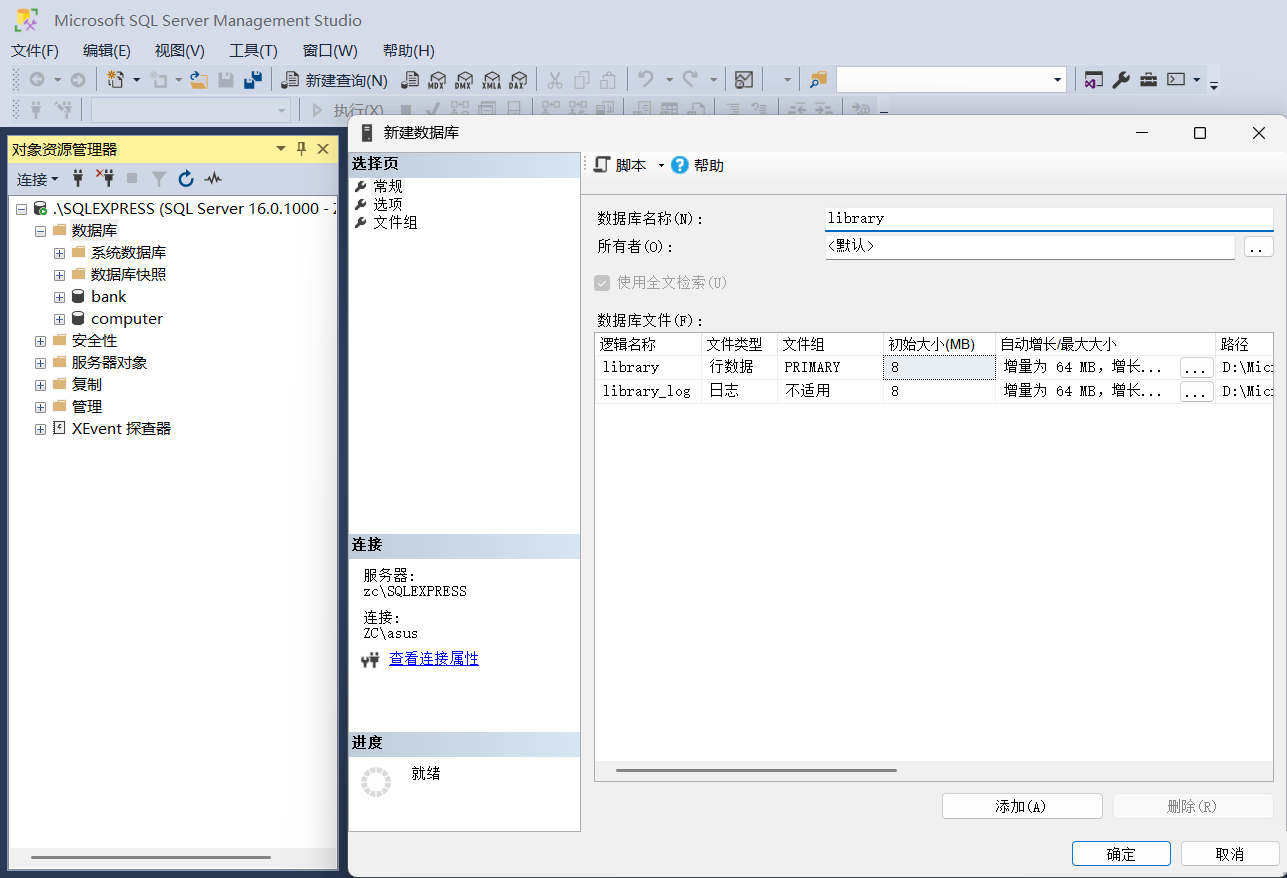
rb：存储读者的借阅记录，关联readers表和books表。

**四、实验步骤**

1.安装SQL Server：根据教程下载即可，打开如下图所示



2.创建数据库：在左边栏中点击数据库，右键，点击新建数据库，命名数据库为library，点击确定，即成功创建数据库。



3.基本表的创建：使用SQL语句创建users、readers、books、rb共4个基本表。

表的结构和列定义如下（省略了详细的约束和外键定义，这部分在修改表步骤中演示）：

-- 建立用户表：

CREATE TABLE users(

account VARCHAR(30) PRIMARY KEY,

passwd VARCHAR(30) NOT NULL,

id VARCHAR(10) CHECK (id IN ('管理员','读者'))

);

-- 建立读者表：

CREATE TABLE readers(

borrowid VARCHAR(30) PRIMARY KEY,

rname VARCHAR(30) NOT NULL,

sex VARCHAR(2) CHECK (sex IN ('男','女')),

job VARCHAR(10),

rCurNum INT,

rBorrowedNum INT,

dept VARCHAR(20),

phone VARCHAR(20),

account VARCHAR(30)

);

-- 建立图书表：

CREATE TABLE books(

isbn VARCHAR(30) PRIMARY KEY,

bname VARCHAR(30) NOT NULL,

pub VARCHAR(30) NOT NULL,

author VARCHAR(15) NOT NULL,

storeNum INT,

bCurNum INT,

available CHAR(2) CHECK (available IN ('是','否'))

);

-- 建立借阅关系表：

CREATE TABLE rb(

borrowid VARCHAR(30),

isbn VARCHAR(30),

startDate DATE,

duration INT,

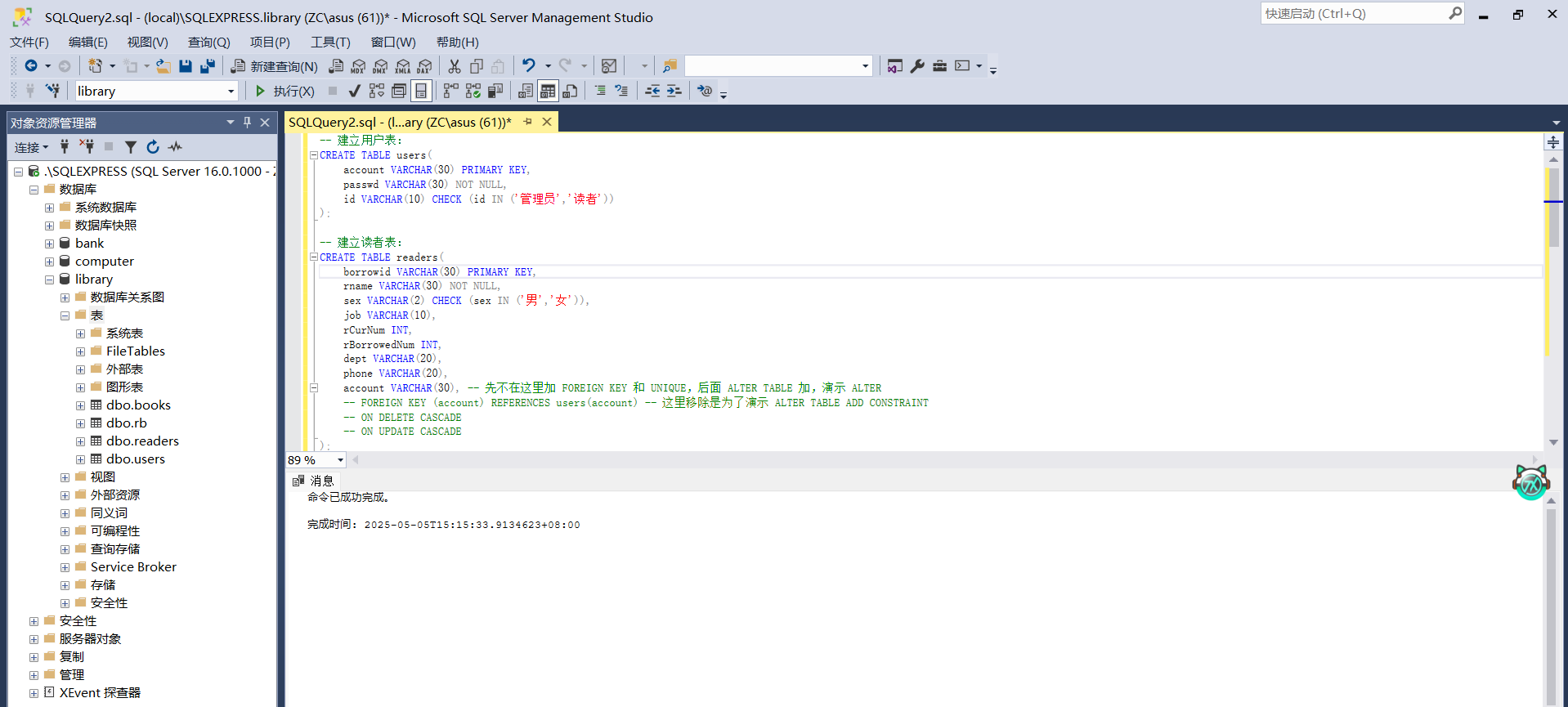
returnDate DATE,

fine INT,

PRIMARY KEY(borrowid, isbn)

);

如图可知四张表皆建立成功。



4.基本表的修改：通过ALTER TABLE语句修改了现有表的结构或添加约束。

示例操作包括为readers表添加了UNIQUE约束和FOREIGN KEY约束，为rb表添加了FOREIGN KEY约束，以及为books表添加了Price列。

-- 添加约束

ALTER TABLE readers ADD CONSTRAINT UQ\_readers\_account UNIQUE (account);

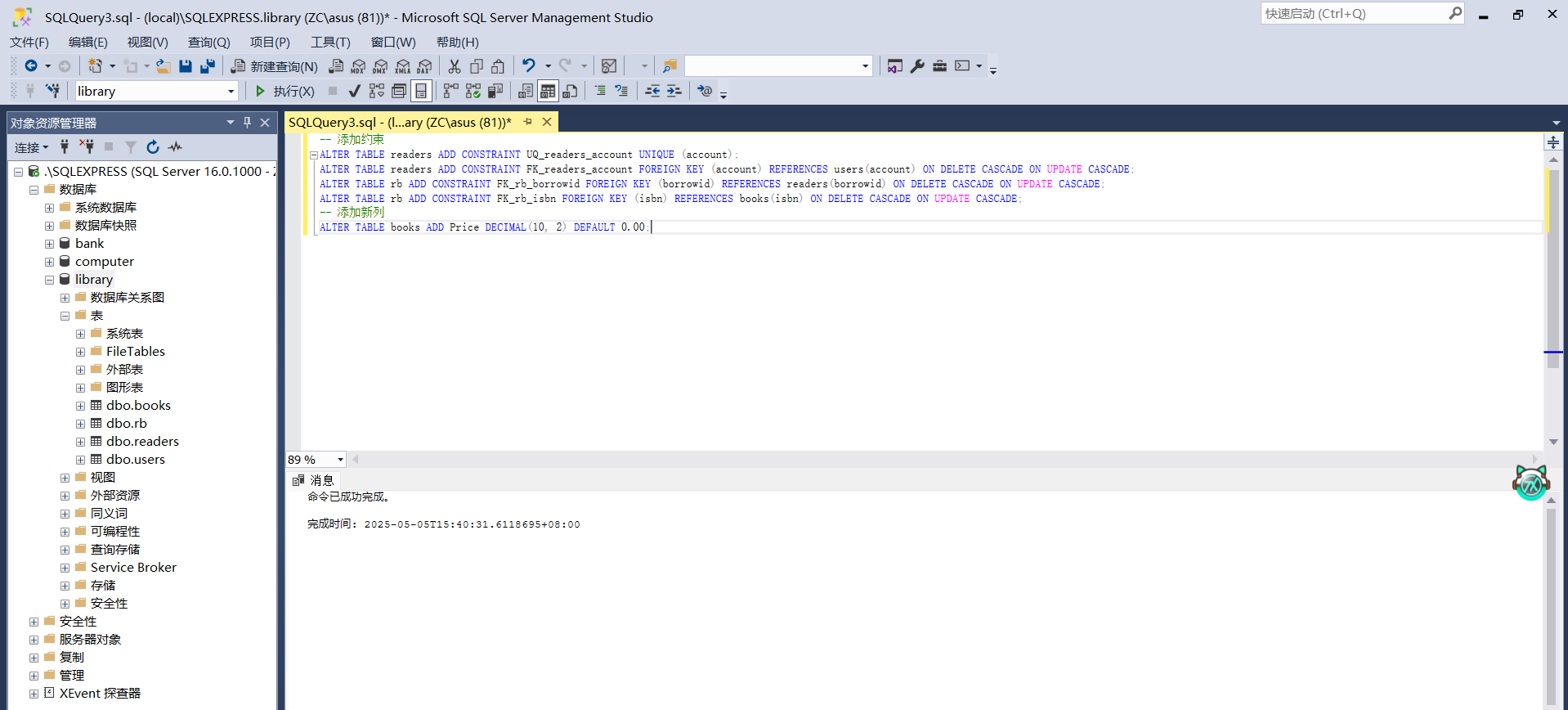
ALTER TABLE readers ADD CONSTRAINT FK\_readers\_account FOREIGN KEY (account) REFERENCES users(account) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE rb ADD CONSTRAINT FK\_rb\_borrowid FOREIGN KEY (borrowid) REFERENCES readers(borrowid) ON DELETE CASCADE ON UPDATE CASCADE;

ALTER TABLE rb ADD CONSTRAINT FK\_rb\_isbn FOREIGN KEY (isbn) REFERENCES books(isbn) ON DELETE CASCADE ON UPDATE CASCADE;

-- 添加新列

ALTER TABLE books ADD Price DECIMAL(10, 2) DEFAULT 0.00;



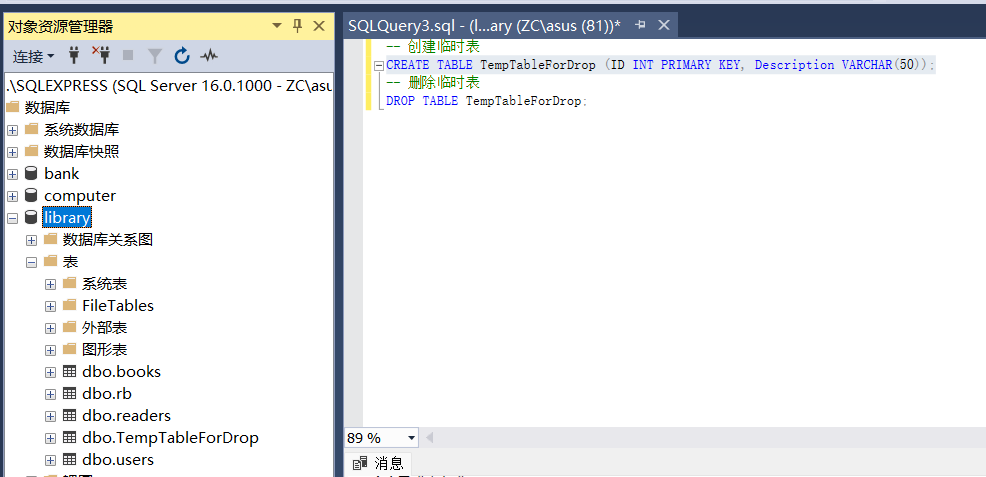
5.基本表的删除：为了演示DROP TABLE操作而不影响核心实验数据库结构，创建了一个临时表TempTableForDrop并随后将其删除。

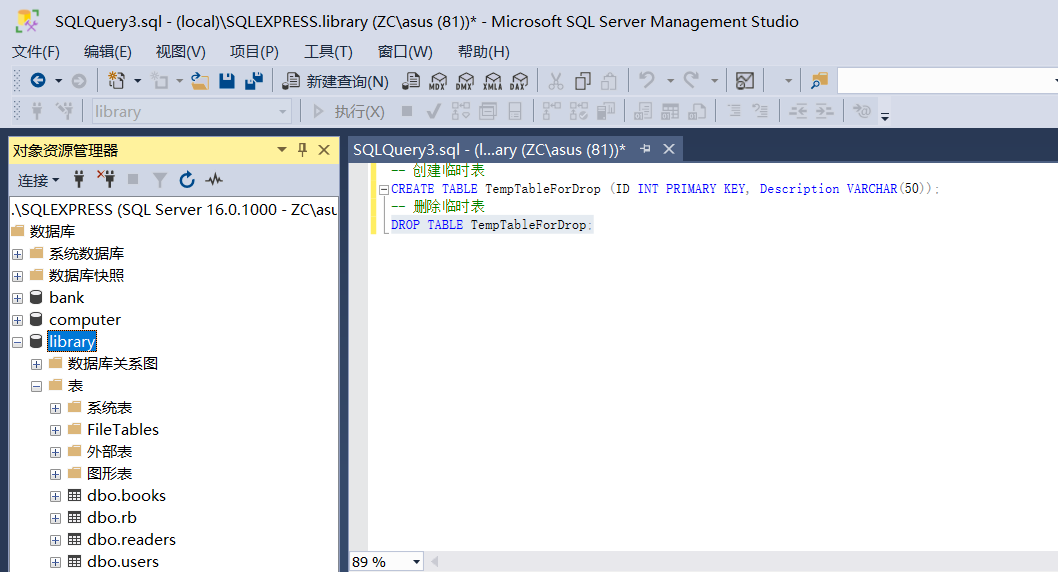
-- 创建临时表

CREATE TABLE TempTableForDrop (ID INT PRIMARY KEY, Description VARCHAR(50));

-- 删除临时表

DROP TABLE TempTableForDrop;

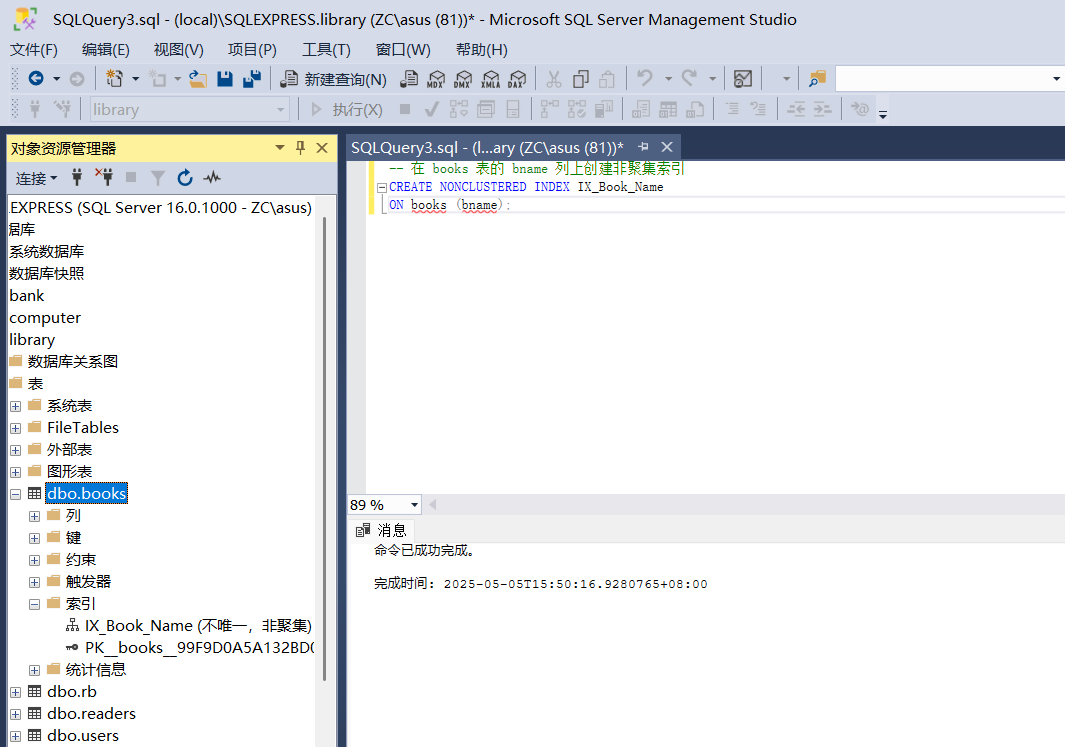




6.索引的创建：根据提供的数据库，在books表的bname列上创建非聚集索引，以加快按书名查找图书的速度。

-- 在 books 表的 bname 列上创建非聚集索引

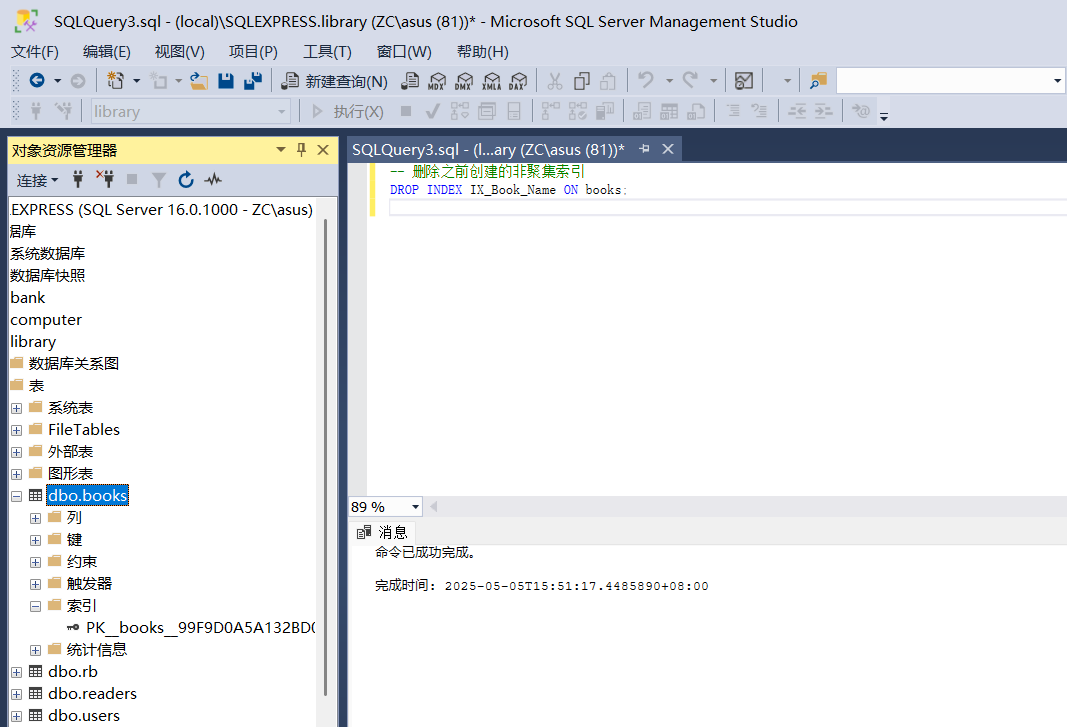
CREATE NONCLUSTERED INDEX IX\_Book\_Name ON books (bname);



7.索引的删除：删除步骤6中创建的非聚集索引IX\_Book\_Name。

-- 删除之前创建的非聚集索引

DROP INDEX IX\_Book\_Name ON books;



8.插入数据：向users、readers、books和rb表中插入了多条示例数据，包括不同身份的用户、不同类型的读者、不同种类的图书以及模拟的借阅记录。

-- 插入 users 数据

INSERT INTO users (account, passwd, id) VALUES

('admin01', 'adminpass', '管理员'),

('reader01\_acc', 'readerpass01', '读者'),

('reader02\_acc', 'readerpass02', '读者'),

('reader03\_acc', 'readerpass03', '读者')，

('reader04\_acc', 'readerpass04', '读者');

-- 插入 readers 数据

INSERT INTO readers (borrowid, rname, sex, job, rCurNum, rBorrowedNum, dept, phone, account) VALUES

('R001', '张小明', '男', '学生', 5, 0, '计算机系', '13800000001', 'reader01\_acc'),

('R002', '李美丽', '女', '教师', 10, 0, '物理系', '13800000002', 'reader02\_acc'),

('R003', '王大力', '男', '学生', 5, 0, '化学系', '13800000003', 'reader03\_acc')，

('R004', '赵四', '男', '职员', 5, 0, '图书馆', '13800000004', 'reader04\_acc');

-- 插入 books 数据

INSERT INTO books (isbn, bname, pub, author, storeNum, bCurNum, available, Price) VALUES

('978-7-121-12345-1', '数据库系统概论', '电子工业出版社', '王珊', 5, 5, '是', 65.50),

('978-7-111-67890-2', '数据结构与算法', '机械工业出版社', '严蔚敏', 8, 8, '是', 72.00),

('978-7-030-54321-3', '高等数学(上)', '科学出版社', '同济大学', 10, 10, '是', 48.00),

('978-7-560-98765-4', '编译原理', '清华大学出版社', '龙书作者', 3, 3, '是', 90.00);

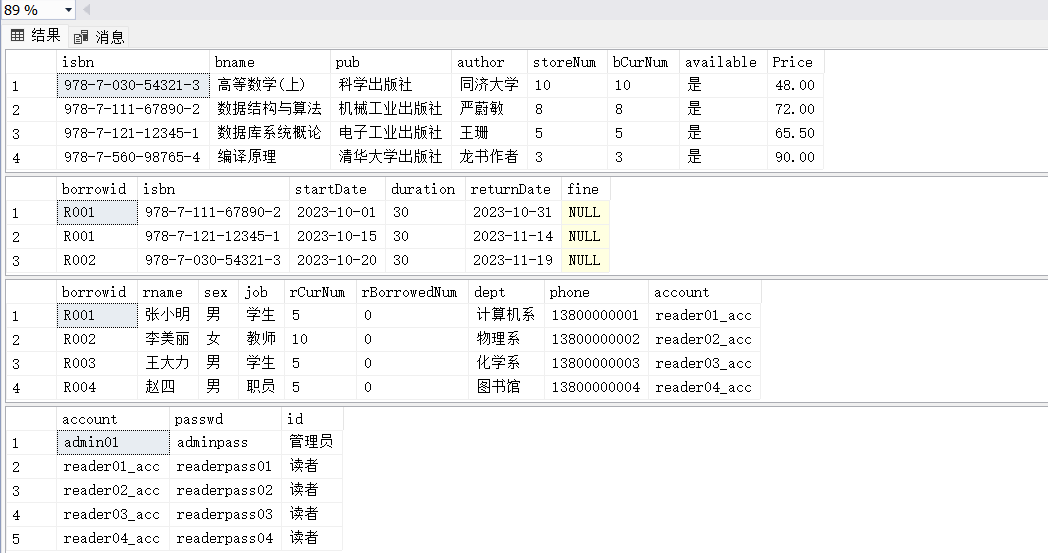
-- 插入 rb 数据

INSERT INTO rb (borrowid, isbn, startDate, duration, returnDate, fine) VALUES

('R001', '978-7-121-12345-1', '2023-10-15', 30, '2023-11-14', NULL),

('R001', '978-7-111-67890-2', '2023-10-01', 30, '2023-10-31', NULL),

('R002', '978-7-030-54321-3', '2023-10-20', 30, '2023-11-19', NULL);



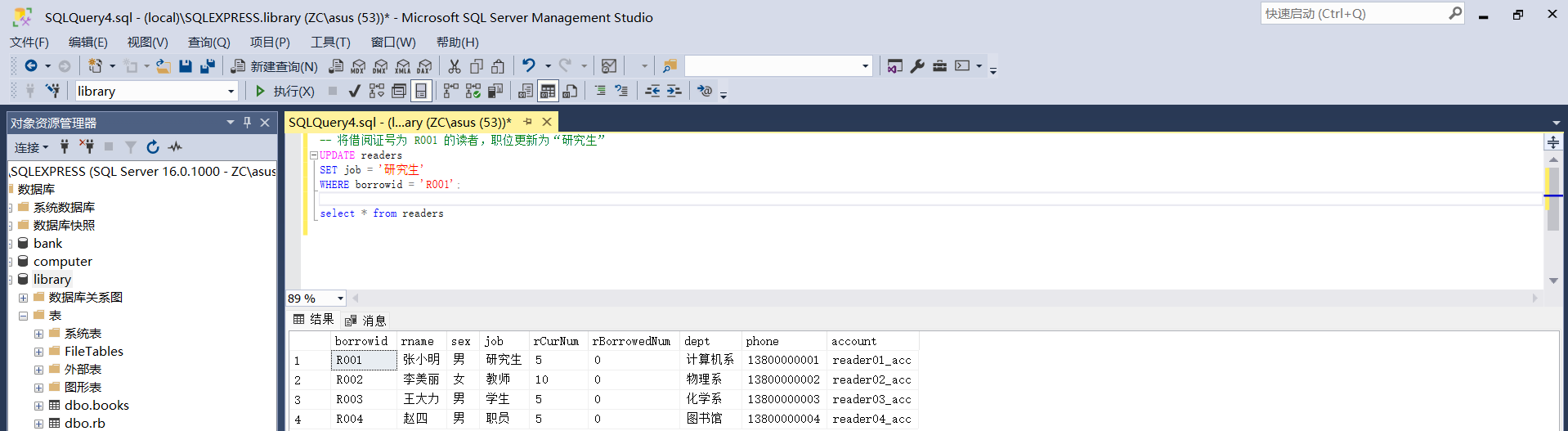
9.修改数据：修改readers表中某位读者的信息，更新了读者的职业。

-- 将借阅证号为 R001 的读者，职位更新为“研究生”

UPDATE readers

SET job = '研究生'

WHERE borrowid = 'R001';

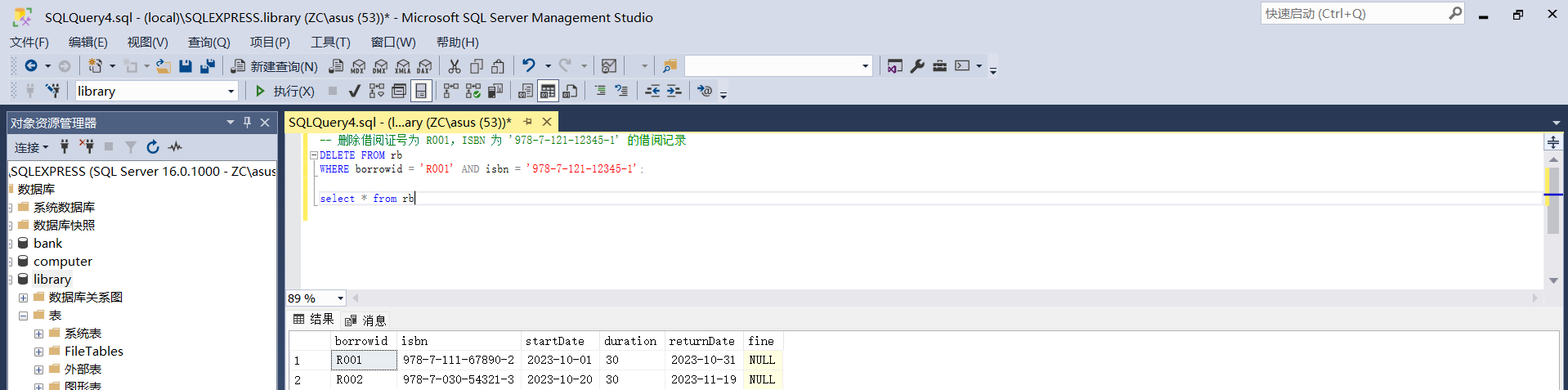


10.删除数据：删除了rb表中的一条借阅记录。

-- 删除借阅证号为 R001，ISBN 为 '978-7-121-12345-1' 的借阅记录

DELETE FROM rb

WHERE borrowid = 'R001' AND isbn = '978-7-121-12345-1';



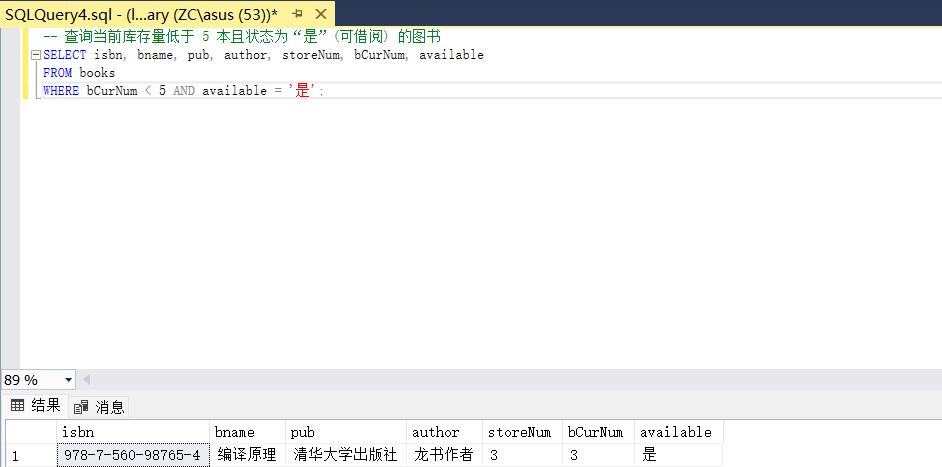
11.单表查询：

-- 查询当前库存量低于 5 本且状态为“是”(可借阅) 的图书

SELECT isbn, bname, pub, author, storeNum, bCurNum, available

FROM books

WHERE bCurNum < 5 AND available = '是';



12.连接查询：

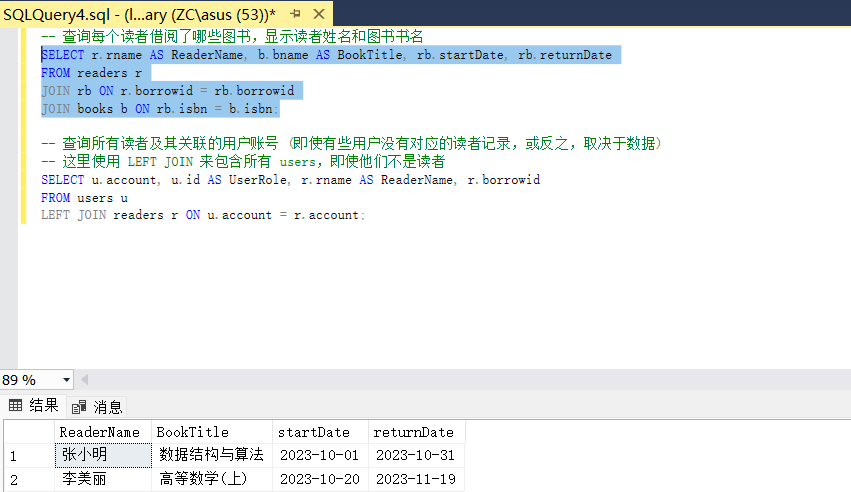
-- 查询每个读者借阅了哪些图书，显示读者姓名和图书书名

SELECT r.rname AS ReaderName, b.bname AS BookTitle, rb.startDate, rb.returnDate

FROM readers r

JOIN rb ON r.borrowid = rb.borrowid

JOIN books b ON rb.isbn = b.isbn;

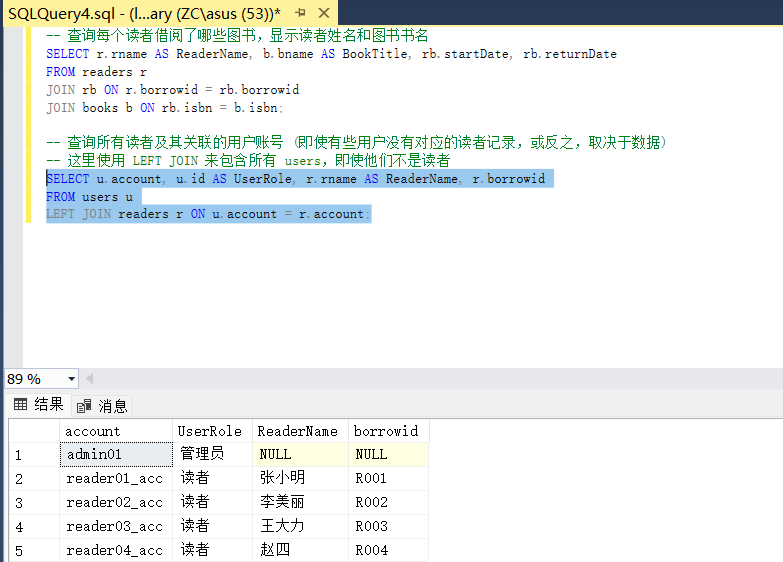


-- 查询所有读者及其关联的用户账号 (即使有些用户没有对应的读者记录)

SELECT u.account, u.id AS UserRole, r.rname AS ReaderName, r.borrowid

FROM users u

LEFT JOIN readers r ON u.account = r.account;



13.嵌套查询：

-- 查询借阅了“数据结构与算法”这本书的读者姓名 (使用 IN 子查询)

SELECT rname

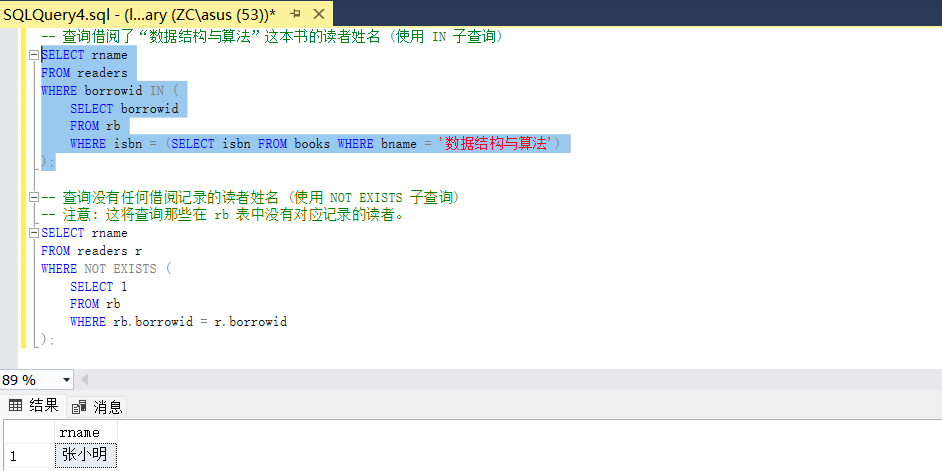
FROM readers

WHERE borrowid IN (

SELECT borrowid

FROM rb

WHERE isbn = (SELECT isbn FROM books WHERE bname = '数据结构与算法')

);

-- 查询没有任何借阅记录的读者姓名 (使用 NOT EXISTS 子查询)

SELECT rname

FROM readers r

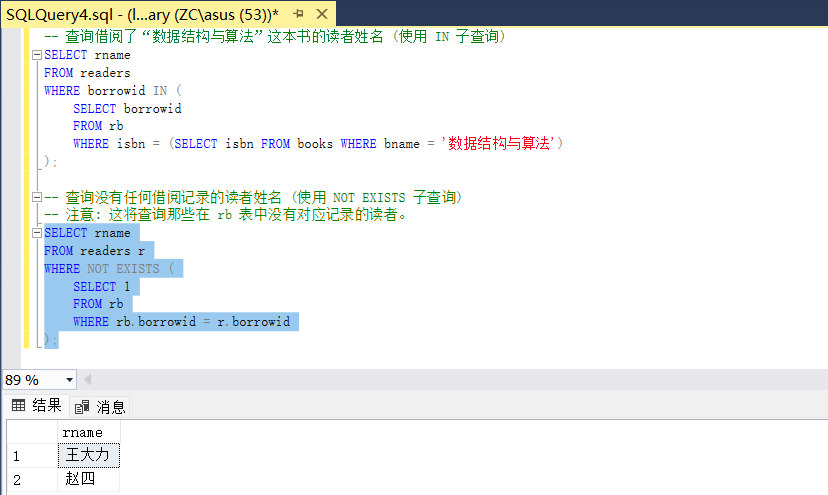
WHERE NOT EXISTS (

SELECT 1

FROM rb

WHERE rb.borrowid = r.borrowid

);

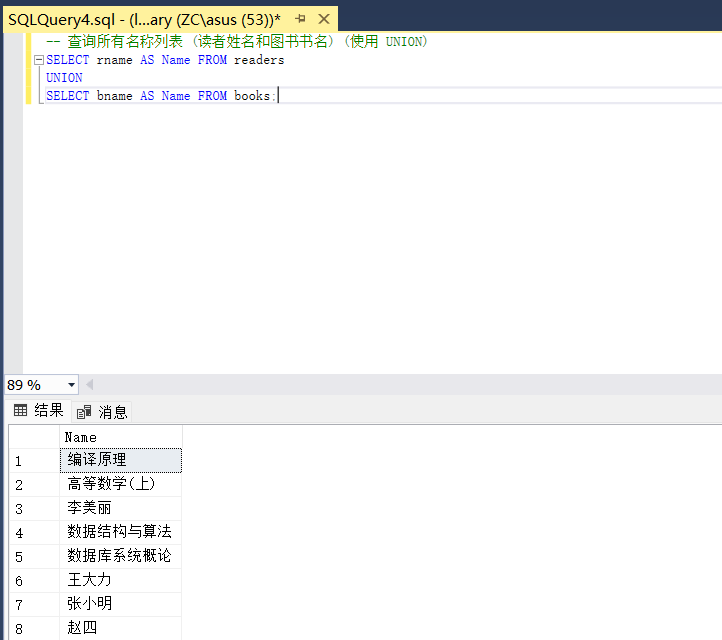


14.集合查询：  
-- 查询所有名称列表 (读者姓名和图书书名) (使用 UNION)

SELECT rname AS Name FROM readers

UNION

SELECT bname AS Name FROM books;



**四、出现的问题及解决方法**

1.问题描述

在“索引的创建”步骤中，一开始尝试在books表的isbn列上创建一个聚集索引，使用的SQL语句是：CREATE CLUSTERED INDEX IsbnIndex ON books(isbn);

执行此语句后，数据库系统（SQLServer）返回了以下错误信息：

消息1902，级别16，状态3，第1行

无法对表'books'创建多个聚集索引。请在创建新聚集索引前删除现有的聚集索引'PK\_\_books\_\_99F9D0A5A132BD00'。

2.问题原因分析

根据错误信息提示，无法创建聚集索引的原因是books表中已经存在一个聚集索引。这是因为在SQL Server中，一张表只能有一个聚集索引(Clustered Index)。

在本次实验的开始，创建books表的语句为：

CREATE TABLE books(

isbn VARCHAR(30) PRIMARY KEY,

...

);

在SQLServer中，当在创建表时指定了PRIMARYKEY约束，并且该表尚未有其他聚集索引时，数据库系统会默认为这个PRIMARYKEY约束所在的列（或列组合）自动创建一个聚集索引。错误信息中提到的'PK\_\_books\_\_99F9D0A5A132BD00'正是系统为isbn这个Primary Key自动生成的聚集索引的名称。

因此，再次尝试使用CREATE CLUSTERED INDEX在isbn列上（或者表中的任何其他列上）创建聚集索引，就会违反“一张表只能有一个聚集索引”的规则，导致创建失败。我们希望通过索引提高isbn列的检索速度的目的，实际上在定义PRIMARYKEY时就已经通过系统自动创建的聚集索引实现了。

3.解决方法

实验要求是演示索引的创建和删除，并不强制要求创建聚集索引。鉴于isbn列作为PrimaryKey已经有了聚集索引，为了完成实验要求且不违反数据库规则，我们决定调整实验方案，改为在books表的另一个常用查询列上创建非聚集索引(Non-ClusteredIndex)，然后删除它。

选择在bname（书名）列上创建非聚集索引，因为用户经常会按书名查找图书。非聚集索引不会改变数据的物理存储顺序，一个表可以拥有多个非聚集索引。

**五、实验源码**

-- 创建表

CREATE TABLE users(

account VARCHAR(30) PRIMARY KEY,

passwd VARCHAR(30) NOT NULL,

id VARCHAR(10) CHECK (id IN ('管理员','读者'))

);

CREATE TABLE readers(

borrowid VARCHAR(30) PRIMARY KEY,

rname VARCHAR(30) NOT NULL,

sex VARCHAR(2) CHECK (sex IN ('男','女')),

job VARCHAR(10),

rCurNum INT,

rBorrowedNum INT,

dept VARCHAR(20),

phone VARCHAR(20),

account VARCHAR(30)

);

CREATE TABLE books(

isbn VARCHAR(30) PRIMARY KEY,

bname VARCHAR(30) NOT NULL,

pub VARCHAR(30) NOT NULL,

author VARCHAR(15) NOT NULL,

storeNum INT,

bCurNum INT,

available CHAR(2) CHECK (available IN ('是','否'))

);

CREATE TABLE rb(

borrowid VARCHAR(30),

isbn VARCHAR(30),

startDate DATE,

duration INT,

returnDate DATE,

fine INT,

PRIMARY KEY(borrowid, isbn)

);

-- 修改表

ALTER TABLE readers ADD CONSTRAINT UQ\_readers\_account UNIQUE (account);

ALTER TABLE readers

ADD CONSTRAINT FK\_readers\_account FOREIGN KEY (account) REFERENCES users(account)

ON DELETE CASCADE

ON UPDATE CASCADE;

ALTER TABLE rb

ADD CONSTRAINT FK\_rb\_borrowid FOREIGN KEY (borrowid) REFERENCES readers(borrowid)

ON DELETE CASCADE

ON UPDATE CASCADE;

ALTER TABLE rb

ADD CONSTRAINT FK\_rb\_isbn FOREIGN KEY (isbn) REFERENCES books(isbn)

ON DELETE CASCADE

ON UPDATE CASCADE;

ALTER TABLE books ADD Price DECIMAL(10, 2) DEFAULT 0.00;

-- 删除表

CREATE TABLE TempTableForDrop (

ID INT PRIMARY KEY,

Description VARCHAR(50)

);

DROP TABLE TempTableForDrop;

-- 索引的创建

CREATE NONCLUSTERED INDEX IX\_Book\_Name

ON books (bname);

-- 索引的删除

DROP INDEX IX\_Book\_Name ON books;

-- 插入数据

INSERT INTO users (account, passwd, id) VALUES

('admin01', 'adminpass', '管理员'),

('reader01\_acc', 'readerpass01', '读者'),

('reader02\_acc', 'readerpass02', '读者'),

('reader03\_acc', 'readerpass03', '读者'),

('reader04\_acc', 'readerpass04', '读者');

INSERT INTO readers (borrowid, rname, sex, job, rCurNum, rBorrowedNum, dept, phone, account) VALUES

('R001', '张小明', '男', '学生', 5, 0, '计算机系', '13800000001', 'reader01\_acc'),

('R002', '李美丽', '女', '教师', 10, 0, '物理系', '13800000002', 'reader02\_acc'),

('R003', '王大力', '男', '学生', 5, 0, '化学系', '13800000003', 'reader03\_acc'),

('R004', '赵四', '男', '职员', 5, 0, '图书馆', '13800000004', 'reader04\_acc');

INSERT INTO books (isbn, bname, pub, author, storeNum, bCurNum, available, Price) VALUES

('978-7-121-12345-1', '数据库系统概论', '电子工业出版社', '王珊', 5, 5, '是', 65.50),

('978-7-111-67890-2', '数据结构与算法', '机械工业出版社', '严蔚敏', 8, 8, '是', 72.00),

('978-7-030-54321-3', '高等数学(上)', '科学出版社', '同济大学', 10, 10, '是', 48.00),

('978-7-560-98765-4', '编译原理', '清华大学出版社', '龙书作者', 3, 3, '是', 90.00),

('978-7-100-00000-1', '离散数学', '某出版社', '某作者', 2, 0, '否', 55.00),

('978-7-100-00000-2', '计算机导论', '某出版社', '某作者', 5, 5, '是', 60.00);

INSERT INTO rb (borrowid, isbn, startDate, duration, returnDate, fine) VALUES

('R001', '978-7-121-12345-1', DATEADD(day, -10, GETDATE()), 30, DATEADD(day, 20, GETDATE()), NULL),

('R001', '978-7-111-67890-2', DATEADD(day, -40, GETDATE()), 30, DATEADD(day, -10, GETDATE()), NULL), -- 逾期示例

('R002', '978-7-030-54321-3', DATEADD(day, -5, GETDATE()), 30, DATEADD(day, 25, GETDATE()), NULL);

-- 修改数据

UPDATE readers

SET job = '研究生'

WHERE borrowid = 'R001';

-- 删除数据

DELETE FROM rb

WHERE borrowid = 'R001' AND isbn = '978-7-121-12345-1';

-- 单表查询

SELECT isbn, bname, pub, author, storeNum, bCurNum, available

FROM books

WHERE bCurNum < 5 AND available = '是';

-- 连接查询 1

SELECT r.rname AS ReaderName, b.bname AS BookTitle, rb.startDate, rb.returnDate AS ExpectedReturnDate

FROM readers r

JOIN rb ON r.borrowid = rb.borrowid

JOIN books b ON rb.isbn = b.isbn;

-- 连接查询 2

SELECT u.account, u.id AS UserRole, r.rname AS ReaderName, r.borrowid

FROM users u

LEFT JOIN readers r ON u.account = r.account;

-- 嵌套查询 1

SELECT rname

FROM readers

WHERE borrowid IN (

SELECT borrowid

FROM rb

WHERE isbn = (SELECT isbn FROM books WHERE bname = '数据结构与算法')

);

-- 嵌套查询 2

SELECT rname

FROM readers r

WHERE NOT EXISTS (

SELECT 1

FROM rb

WHERE rb.borrowid = r.borrowid

);

-- 集合查询

SELECT rname AS Name FROM readers

UNION

SELECT bname AS Name FROM books;