交互式SQL实验报告

实验目的

熟悉通过SQL对数据库进行操作。

实验环境

- 实验工具：SQL Server，及其提供的SQL Server Management Studio 20。
- 数据库设计：LibraryDB，图书馆管理数据库，设计如下，
    - 实体1：**Book(book_id, title, isbn, publish_year, publisher_id, category, price, stock)**

字段名	类型	主键	说明
book_id	INT	√	图书ID
title	VARCHAR(100)		书名
isbn	VARCHAR(20)		ISBN编号
publish_year	INT		出版年份
publisher_id	INT	外键 → Publisher(publisher_id)	出版社ID
category	VARCHAR(50)		分类（如文学、计算机）
price	DECIMAL(6,2)		定价
stock	INT		馆藏数量

    - 实体2：**Author(author_id, name, birth_date, nationality)**

字段名	类型	主键	说明
author_id	INT	√	作者ID
name	VARCHAR(100)		作者姓名
birth_date	DATE		出生日期
nationality	VARCHAR(50)		国籍

- 实体3：**Member(member_id, name, email, phone, join_date, membership_type, status)**

字段名	类型	主键	说明
member_id	INT	√	会员ID
name	VARCHAR(100)		姓名
email	VARCHAR(100)		邮箱
phone	VARCHAR(20)		电话
join_date	DATE		加入日期
membership_type	VARCHAR(20)		会员类型（普通/高级）
status	VARCHAR(20)		状态（活跃/冻结）

- 实体4：**Publisher(publisher_id, name, address, contact_email)**

字段名	类型	主键	说明
publisher_id	INT	√	出版社ID
name	VARCHAR(100)		名称
address	VARCHAR(200)		地址
contact_email	VARCHAR(100)		联系邮箱

- 实体5：**Librarian(librarian_id, name, email, hire_date, role)**

字段名	类型	主键	说明
librarian_id	INT	√	管理员ID
name	VARCHAR(100)		姓名
email	VARCHAR(100)		邮箱
hire_date	DATE		入职日期
role	VARCHAR(50)		职位（馆员/系统管理员）

- 联系1：**BookAuthor(book_id, author_id)**

字段名	类型	主键	外键
book_id	INT	√	→ Book(book_id)
author_id	INT	√	→ Author(author_id)

- 联系2： **BorrowRecord(borrow_id, member_id, book_id, librarian_id, borrow_date, return_date, due_date, status)**

字段名	类型	主键	外键/说明
borrow_id	INT	√	借阅ID
member_id	INT		→ Member(member_id)
book_id	INT		→ Book(book_id)
librarian_id	INT		→ Librarian(librarian_id)（办理人）
borrow_date	DATE		借书日期
return_date	DATE		归还日期
due_date	DATE		应归还日期
status	VARCHAR(20)		借阅状态（借出/归还/逾期）

实验内容与完成情况

实验内容

数据定义部分

1. 根据数据库设计创建基本表，并加以修改/删除；
2. 进行索引的创建和删除操作。

数据操作部分

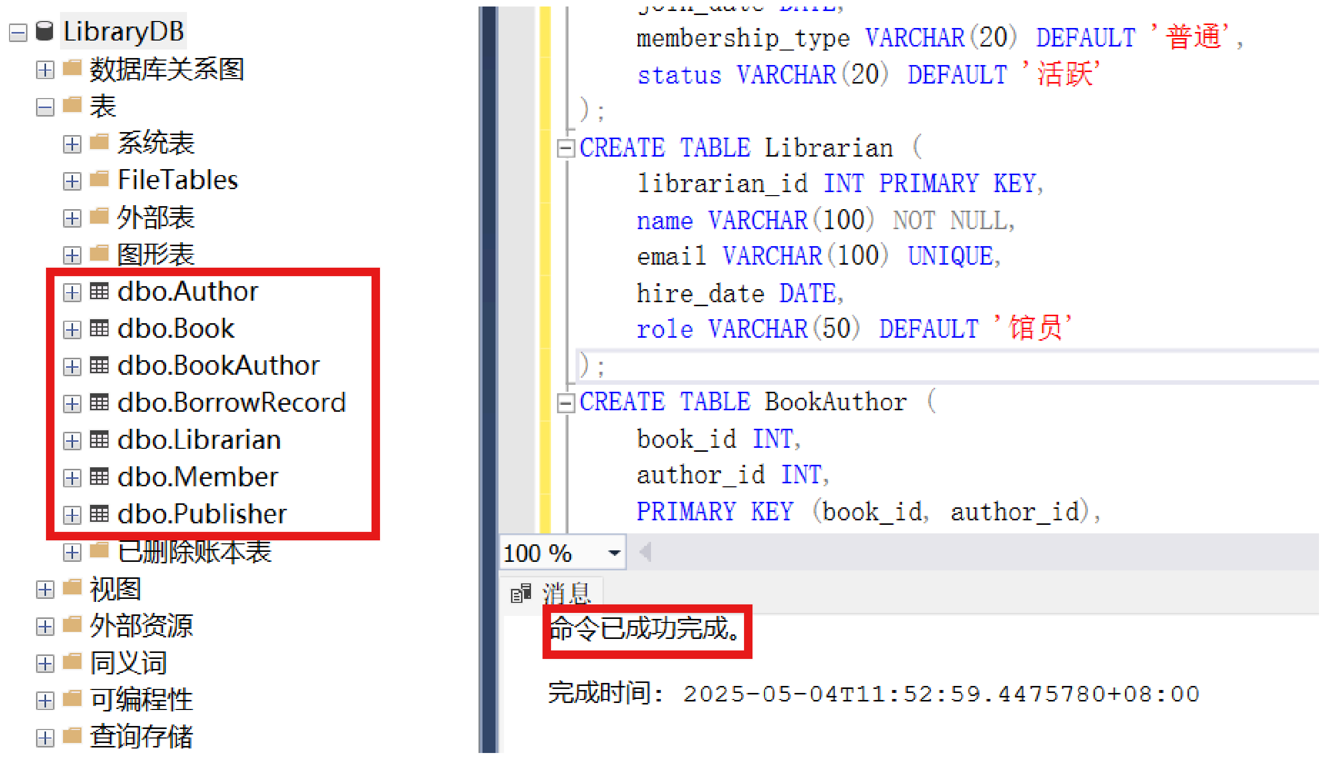
1. 插入、修改、删除数据库数据；
2. 进行各类查询操作，要求至少进行单表查询和集合查询各1次、连接查询和嵌套查询各1次。

实验完成情况

1. 创建基本表：
   - `SQL` 语句：

```sql
CREATE TABLE Publisher (
    publisher_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    address VARCHAR(200),
    contact_email VARCHAR(100)
);
CREATE TABLE Book (
    book_id INT PRIMARY KEY ,
    title VARCHAR(100) NOT NULL,
    isbn VARCHAR(20) UNIQUE,
    publish_year INT,
    publisher_id INT,
    category VARCHAR(50),
    price DECIMAL(6,2),
    stock INT DEFAULT 0,
    FOREIGN KEY (publisher_id) REFERENCES Publisher(publisher_id)
);
CREATE TABLE Author (
    author_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    birth_date DATE,
    nationality VARCHAR(50)
);
CREATE TABLE Member (
    member_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE,
    phone VARCHAR(20),
    join_date DATE,
    membership_type VARCHAR(20) DEFAULT '普通',
    status VARCHAR(20) DEFAULT '活跃'
);
CREATE TABLE Librarian (
    librarian_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE,
    hire_date DATE,
    role VARCHAR(50) DEFAULT '馆员'
);
CREATE TABLE BookAuthor (
    book_id INT,
    author_id INT,
    PRIMARY KEY (book_id, author_id),
    FOREIGN KEY (book_id) REFERENCES Book(book_id) ON DELETE CASCADE,
    FOREIGN KEY (author_id) REFERENCES Author(author_id) ON DELETE CASCADE
);
CREATE TABLE BorrowRecord (
    borrow_id INT PRIMARY KEY ,
    member_id INT,
    book_id INT,
    librarian_id INT,
    borrow_date DATE,
    due_date DATE,
    return_date DATE,
    status VARCHAR(20) DEFAULT '借出',
    FOREIGN KEY (member_id) REFERENCES Member(member_id),
    FOREIGN KEY (book_id) REFERENCES Book(book_id),
    FOREIGN KEY (librarian_id) REFERENCES Librarian(librarian_id)
);
```

- 建表语句执行图示：



```
    membership_type VARCHAR(20) DEFAULT '普通',
    status VARCHAR(20) DEFAULT '活跃'
);
CREATE TABLE Librarian (
    librarian_id INT PRIMARY KEY,
    name VARCHAR(100) NOT NULL,
    email VARCHAR(100) UNIQUE,
    hire_date DATE,
    role VARCHAR(50) DEFAULT '馆员'
);
CREATE TABLE BookAuthor (
    book_id INT,
    author_id INT,
    PRIMARY KEY (book_id, author_id),
```

```
命令已成功完成。

完成时间: 2025-05-04T11:52:59.4475780+08:00
```

2. 修改/删除表

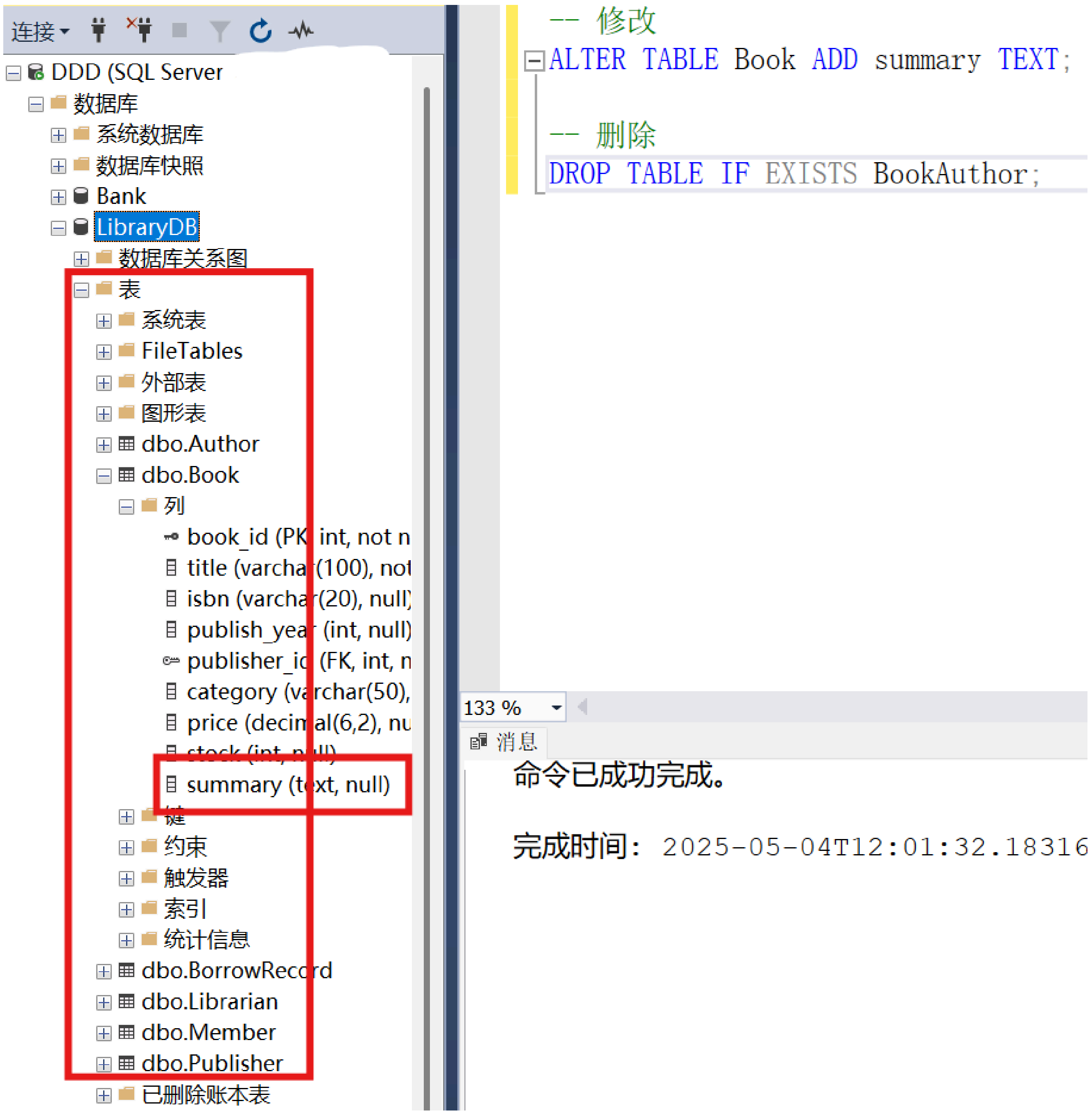
- 修改：为 Book 实体增加 Summary 属性：

```
ALTER TABLE Book ADD COLUMN summary TEXT;
```

- 删除：删除 BookAuthor 表（后续使用时已重新添加）

```
DROP TABLE IF EXISTS BookAuthor;
```

- 执行图示：



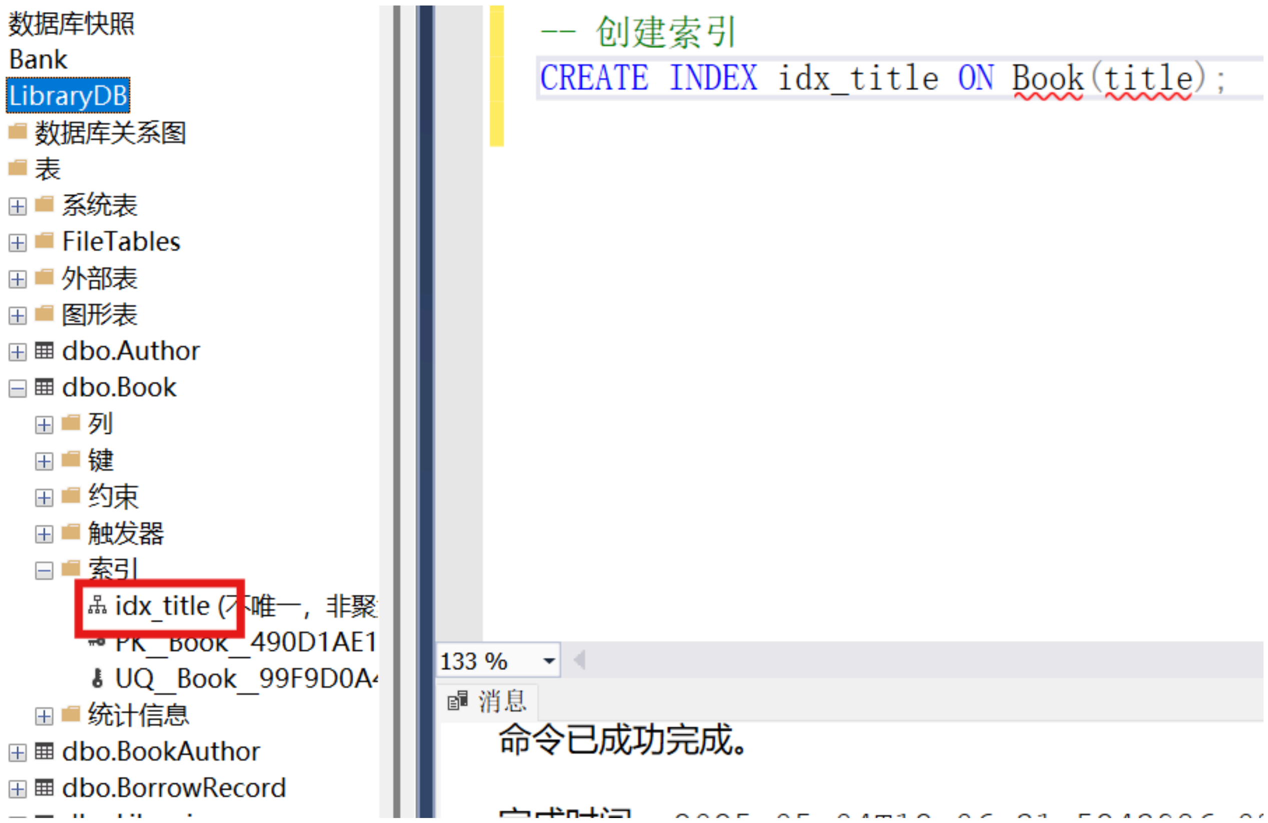
```sql
-- 修改
ALTER TABLE Book ADD summary TEXT;

-- 删除
DROP TABLE IF EXISTS BookAuthor;
```

```
命令已成功完成。

完成时间： 2025-05-04T12:01:32.18316
```

3. 创建/删除索引
   - 创建
     - SQL 语句

       ```sql
       CREATE INDEX idx_title ON Book(title);
       ```

     - 执行图示：



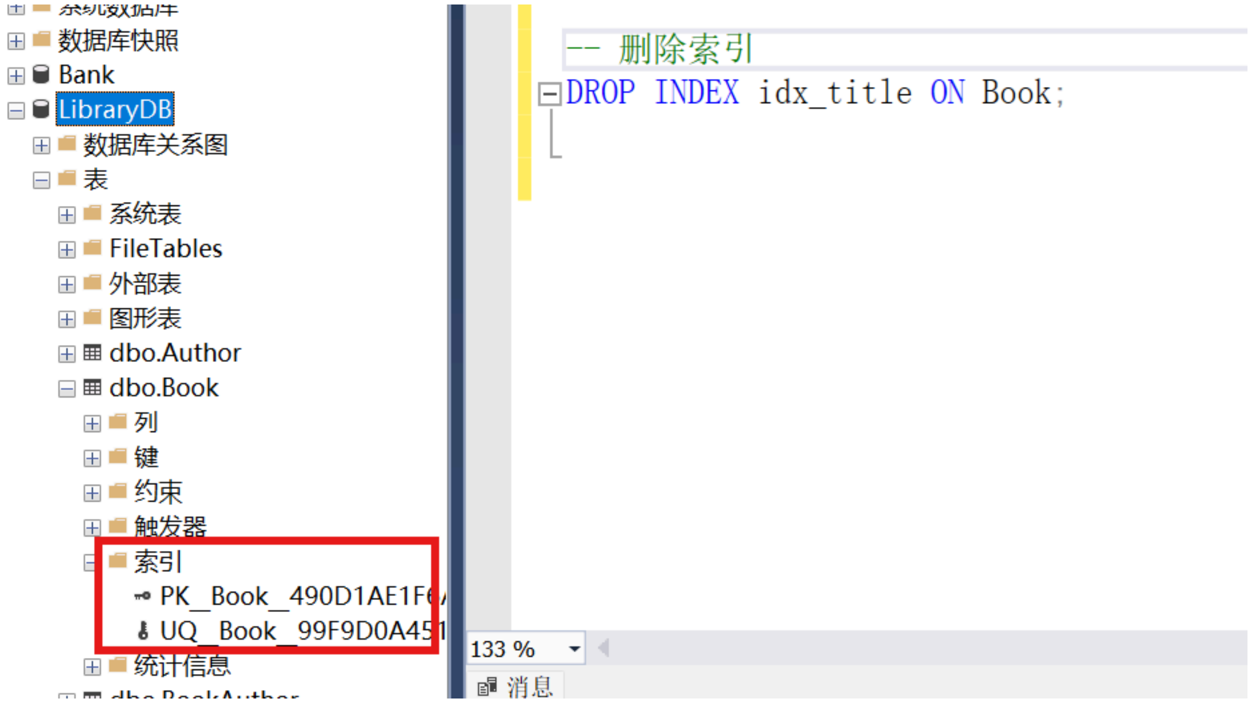
```
-- 创建索引
CREATE INDEX idx_title ON Book(title);
```

```
命令已成功完成。
```

- 删除
  - SQL 语句

    ```
    DROP INDEX idx_title ON Book;
    ```

  - 执行图示：



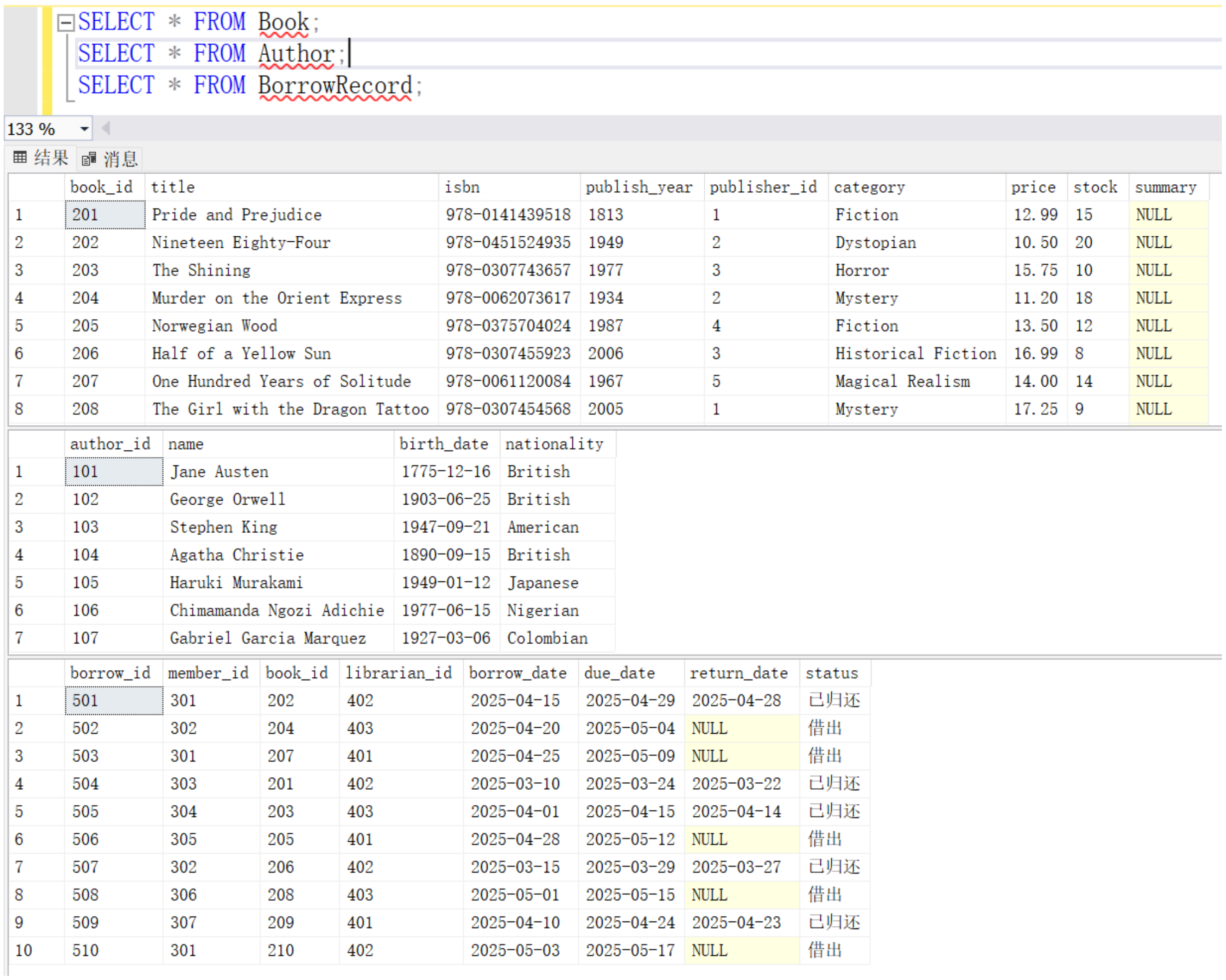
```
-- 删除索引
DROP INDEX idx_title ON Book;
```

4. 插入数据

- SQL 语句：

```sql
INSERT INTO Publisher (publisher_id, name, address, contact_email) VALUES
(1, 'Penguin Random House', '375 Hudson Street, New York, NY 10014, USA', 'contact@peng
(2, 'HarperCollins', '195 Broadway, New York, NY 10007, USA', 'inquiries@harpercollins.
(3, 'Simon & Schuster', '1230 Avenue of the Americas, New York, NY 10020, USA', 'custom
(4, 'Macmillan Publishers', '120 Broadway, New York, NY 10271, USA', 'info@macmillan.co
(5, 'Hachette Book Group', '1290 Avenue of the Americas, New York, NY 10104, USA', 'hbg

INSERT INTO Author (author_id, name, birth_date, nationality) VALUES
(101, 'Jane Austen', '1775-12-16', 'British'),
(102, 'George Orwell', '1903-06-25', 'British'),
(103, 'Stephen King', '1947-09-21', 'American'),
(104, 'Agatha Christie', '1890-09-15', 'British'),
(105, 'Haruki Murakami', '1949-01-12', 'Japanese'),
(106, 'Chimamanda Ngozi Adichie', '1977-06-15', 'Nigerian'),
(107, 'Gabriel Garcia Marquez', '1927-03-06', 'Colombian');

INSERT INTO Book (book_id, title, isbn, publish_year, publisher_id, category, price, st
(201, 'Pride and Prejudice', '978-0141439518', 1813, 1, 'Fiction', 12.99, 15),
(202, 'Nineteen Eighty-Four', '978-0451524935', 1949, 2, 'Dystopian', 10.50, 20),
(203, 'The Shining', '978-0307743657', 1977, 3, 'Horror', 15.75, 10),
(204, 'Murder on the Orient Express', '978-0062073617', 1934, 2, 'Mystery', 11.20, 18),
(205, 'Norwegian Wood', '978-0375704024', 1987, 4, 'Fiction', 13.50, 12),
(206, 'Half of a Yellow Sun', '978-0307455923', 2006, 3, 'Historical Fiction', 16.99, 8
(207, 'One Hundred Years of Solitude', '978-0061120084', 1967, 5, 'Magical Realism', 14
(208, 'The Girl with the Dragon Tattoo', '978-0307454568', 2005, 1, 'Mystery', 17.25, 9
(209, 'To Kill a Mockingbird', '978-0061120091', 1960, 2, 'Fiction', 11.99, 22),
(210, 'The Lord of the Rings', '978-0618260274', 1954, 4, 'Fantasy', 25.00, 7);

INSERT INTO BookAuthor (book_id, author_id) VALUES
(201, 101),
(202, 102),
(203, 103),
(204, 104),
(205, 105),
(206, 106),
(207, 107),
(208, 105),
(209, 101),
(210, 102),
(201, 107);

INSERT INTO Member (member_id, name, email, phone, join_date, membership_type, status)
(301, 'Alice Smith', 'alice.smith@email.com', '123-456-7890', '2024-01-15', '高级', '活
(302, 'Bob Johnson', 'bob.johnson@email.com', '987-654-3210', '2023-11-20', '普通', '活
(303, 'Charlie Brown', 'charlie.brown@email.com', '555-123-4567', '2024-03-01', '普通',
(304, 'Diana Lee', 'diana.lee@email.com', '111-222-3333', '2024-05-01', '高级', '活跃')
(305, 'Ethan Williams', 'ethan.williams@email.com', '444-555-6666', '2023-09-10', '普通
(306, 'Fiona Green', 'fiona.green@email.com', '777-888-9999', '2024-02-28', '普通', '活
(307, 'George Harris', 'george.harris@email.com', '222-333-4444', '2023-12-05', '高级',

INSERT INTO Librarian (librarian_id, name, email, hire_date, role) VALUES
(401, 'Linda Davis', 'linda.davis@library.com', '2022-08-10', '馆长'),
(402, 'Michael Clark', 'michael.clark@library.com', '2023-04-01', '助理馆员'),
(403, 'Susan Baker', 'susan.baker@library.com', '2023-10-15', '馆员');

INSERT INTO BorrowRecord (borrow_id, member_id, book_id, librarian_id, borrow_date, due
(501, 301, 202, 402, '2025-04-15', '2025-04-29', '2025-04-28', '已归还'),
(502, 302, 204, 403, '2025-04-20', '2025-05-04', NULL, '借出'),
(503, 301, 207, 401, '2025-04-25', '2025-05-09', NULL, '借出'),
(504, 303, 201, 402, '2025-03-10', '2025-03-24', '2025-03-22', '已归还'),
(505, 304, 203, 403, '2025-04-01', '2025-04-15', '2025-04-14', '已归还'),
(506, 305, 205, 401, '2025-04-28', '2025-05-12', NULL, '借出'),
(507, 302, 206, 402, '2025-03-15', '2025-03-29', '2025-03-27', '已归还'),
(508, 306, 208, 403, '2025-05-01', '2025-05-15', NULL, '借出'),
(509, 307, 209, 401, '2025-04-10', '2025-04-24', '2025-04-23', '已归还'),
(510, 301, 210, 402, '2025-05-03', '2025-05-17', NULL, '借出');
```

- 执行图示（使用查询语句验证）：



```
SELECT * FROM Book;
SELECT * FROM Author;
SELECT * FROM BorrowRecord;
```

133 %

结果 消息

	book_id	title	isbn	publish_year	publisher_id	category	price	stock	summary
1	201	Pride and Prejudice	978-0141439518	1813	1	Fiction	12.99	15	NULL
2	202	Nineteen Eighty-Four	978-0451524935	1949	2	Dystopian	10.50	20	NULL
3	203	The Shining	978-0307743657	1977	3	Horror	15.75	10	NULL
4	204	Murder on the Orient Express	978-0062073617	1934	2	Mystery	11.20	18	NULL
5	205	Norwegian Wood	978-0375704024	1987	4	Fiction	13.50	12	NULL
6	206	Half of a Yellow Sun	978-0307455923	2006	3	Historical Fiction	16.99	8	NULL
7	207	One Hundred Years of Solitude	978-0061120084	1967	5	Magical Realism	14.00	14	NULL
8	208	The Girl with the Dragon Tattoo	978-0307454568	2005	1	Mystery	17.25	9	NULL

	author_id	name	birth_date	nationality
1	101	Jane Austen	1775-12-16	British
2	102	George Orwell	1903-06-25	British
3	103	Stephen King	1947-09-21	American
4	104	Agatha Christie	1890-09-15	British
5	105	Haruki Murakami	1949-01-12	Japanese
6	106	Chimamanda Ngozi Adichie	1977-06-15	Nigerian
7	107	Gabriel Garcia Marquez	1927-03-06	Colombian

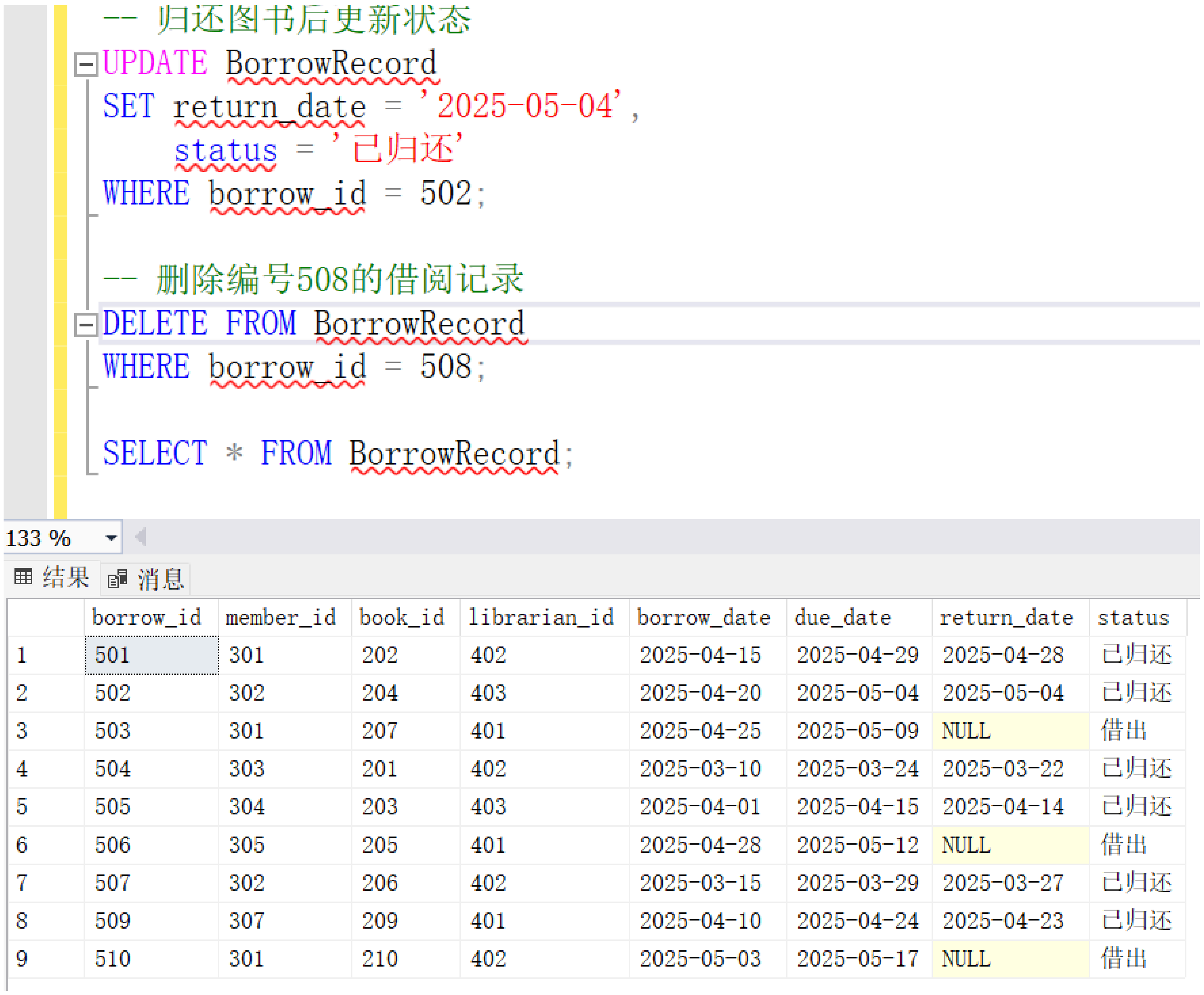
	borrow_id	member_id	book_id	librarian_id	borrow_date	due_date	return_date	status
1	501	301	202	402	2025-04-15	2025-04-29	2025-04-28	已归还
2	502	302	204	403	2025-04-20	2025-05-04	NULL	借出
3	503	301	207	401	2025-04-25	2025-05-09	NULL	借出
4	504	303	201	402	2025-03-10	2025-03-24	2025-03-22	已归还
5	505	304	203	403	2025-04-01	2025-04-15	2025-04-14	已归还
6	506	305	205	401	2025-04-28	2025-05-12	NULL	借出
7	507	302	206	402	2025-03-15	2025-03-29	2025-03-27	已归还
8	508	306	208	403	2025-05-01	2025-05-15	NULL	借出
9	509	307	209	401	2025-04-10	2025-04-24	2025-04-23	已归还
10	510	301	210	402	2025-05-03	2025-05-17	NULL	借出

5. 修改/删除数据：

- SQL 语句：

```sql
-- 归还图书后更新状态
UPDATE BorrowRecord
SET return_date = '2025-05-04',
    status = '已归还'
WHERE borrow_id = 502;

-- 删除编号508的借阅记录
DELETE FROM BorrowRecord
WHERE borrow_id = 508;
```

- 执行图示：



```
-- 归还图书后更新状态
UPDATE BorrowRecord
SET return_date = '2025-05-04',
    status = '已归还'
WHERE borrow_id = 502;

-- 删除编号508的借阅记录
DELETE FROM BorrowRecord
WHERE borrow_id = 508;

SELECT * FROM BorrowRecord;
```

133 %

结果 消息

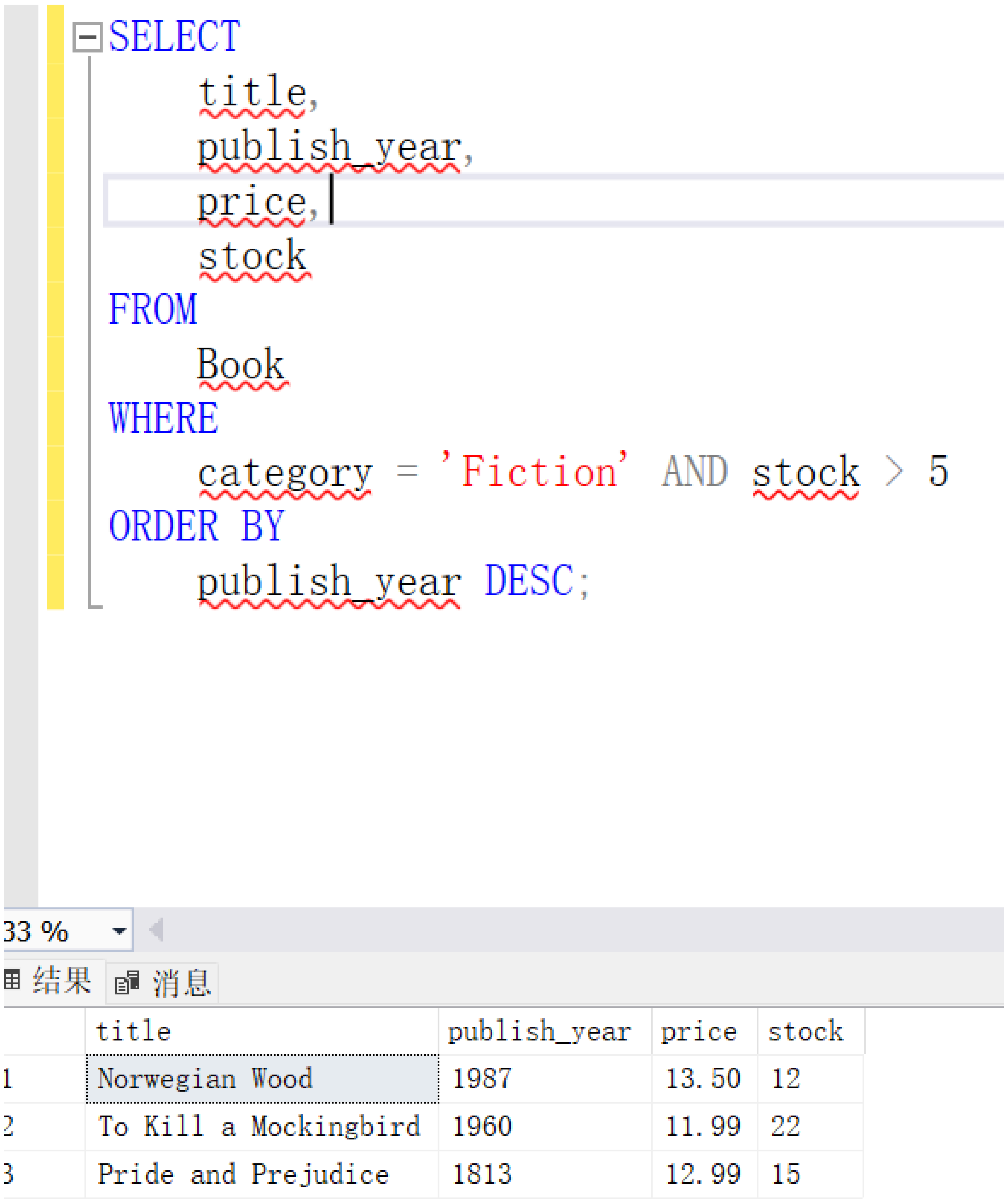
	borrow_id	member_id	book_id	librarian_id	borrow_date	due_date	return_date	status
1	501	301	202	402	2025-04-15	2025-04-29	2025-04-28	已归还
2	502	302	204	403	2025-04-20	2025-05-04	2025-05-04	已归还
3	503	301	207	401	2025-04-25	2025-05-09	NULL	借出
4	504	303	201	402	2025-03-10	2025-03-24	2025-03-22	已归还
5	505	304	203	403	2025-04-01	2025-04-15	2025-04-14	已归还
6	506	305	205	401	2025-04-28	2025-05-12	NULL	借出
7	507	302	206	402	2025-03-15	2025-03-29	2025-03-27	已归还
8	509	307	209	401	2025-04-10	2025-04-24	2025-04-23	已归还
9	510	301	210	402	2025-05-03	2025-05-17	NULL	借出

6. 查询：

- 单表查询：
  - SQL 语句：

```
-- 查询所有库存量大于5的小说的名字、出版年份、价格、库存量，以出版年份降序排列
SELECT
    title,
    publish_year,
    price,
    stock
FROM
    Book
WHERE
    category = 'Fiction' AND stock > 5
ORDER BY
    publish_year DESC;
```

  - 执行图示：



```
SELECT
      title,
      publish_year,
      price,
      stock
FROM
      Book
WHERE
      category = 'Fiction' AND stock > 5
ORDER BY
      publish_year DESC;
```

33 %

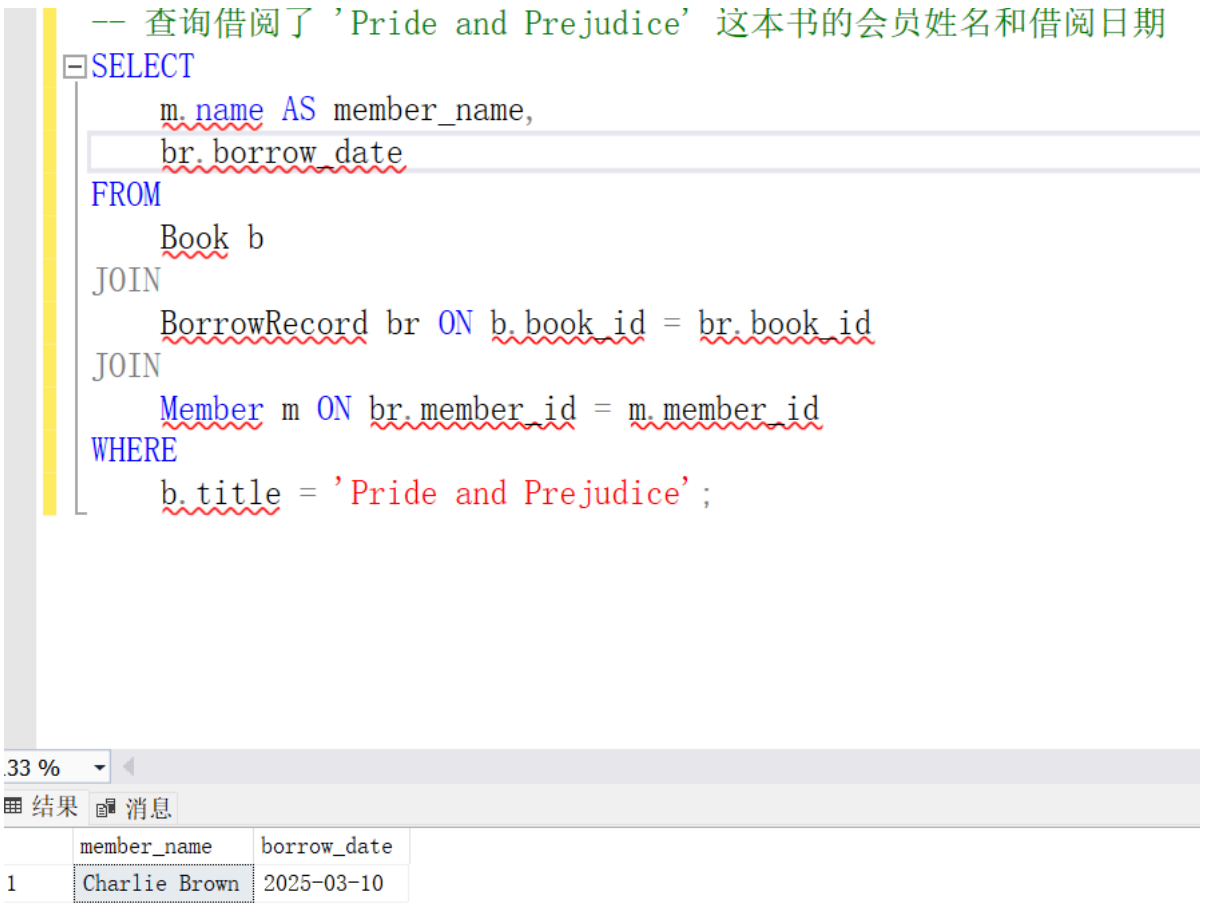
结果 消息

```
   title                  publish_year  price  stock
1  Norwegian Wood         1987          13.50  12
2  To Kill a Mockingbird  1960          11.99  22
3  Pride and Prejudice    1813          12.99  15
```

- 连接查询
  a. 查询借阅了 'Pride and Prejudice' 这本书的会员姓名和借阅日期
    - SQL 语句：

```
SELECT
    m.name AS member_name,
    br.borrow_date
FROM
    Book b
JOIN
    BorrowRecord br ON b.book_id = br.book_id
JOIN
    Member m ON br.member_id = m.member_id
WHERE
    b.title = 'Pride and Prejudice';
```

- 执行图示：



```
-- 查询借阅了 'Pride and Prejudice' 这本书的会员姓名和借阅日期
SELECT
    m.name AS member_name,
    br.borrow_date
FROM
    Book b
JOIN
    BorrowRecord br ON b.book_id = br.book_id
JOIN
    Member m ON br.member_id = m.member_id
WHERE
    b.title = 'Pride and Prejudice';
```

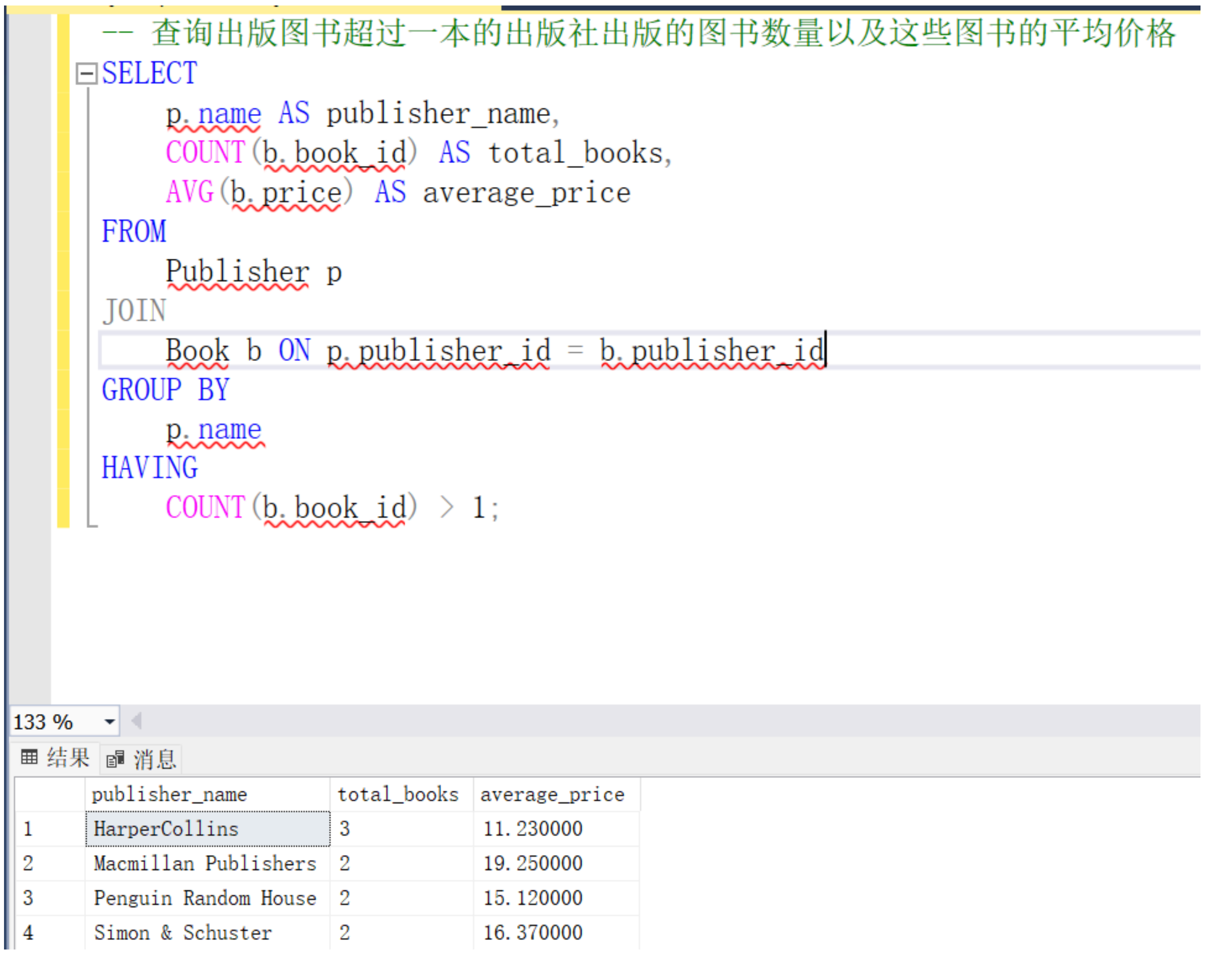
```
   member_name    borrow_date
1  Charlie Brown  2025-03-10
```

b. 查询出版图书超过一本的出版社出版的图书数量以及这些图书的平均价格

- SQL 语句

```
SELECT
    p.name AS publisher_name,
    COUNT(b.book_id) AS total_books,
    AVG(b.price) AS average_price
FROM
    Publisher p
JOIN
    Book b ON p.publisher_id = b.publisher_id
GROUP BY
    p.name
HAVING
    COUNT(b.book_id) > 1;
```

- 执行图示：



```
-- 查询出版图书超过一本的出版社出版的图书数量以及这些图书的平均价格
SELECT
    p.name AS publisher_name,
    COUNT(b.book_id) AS total_books,
    AVG(b.price) AS average_price
FROM
    Publisher p
JOIN
    Book b ON p.publisher_id = b.publisher_id
GROUP BY
    p.name
HAVING
    COUNT(b.book_id) > 1;
```

133 %

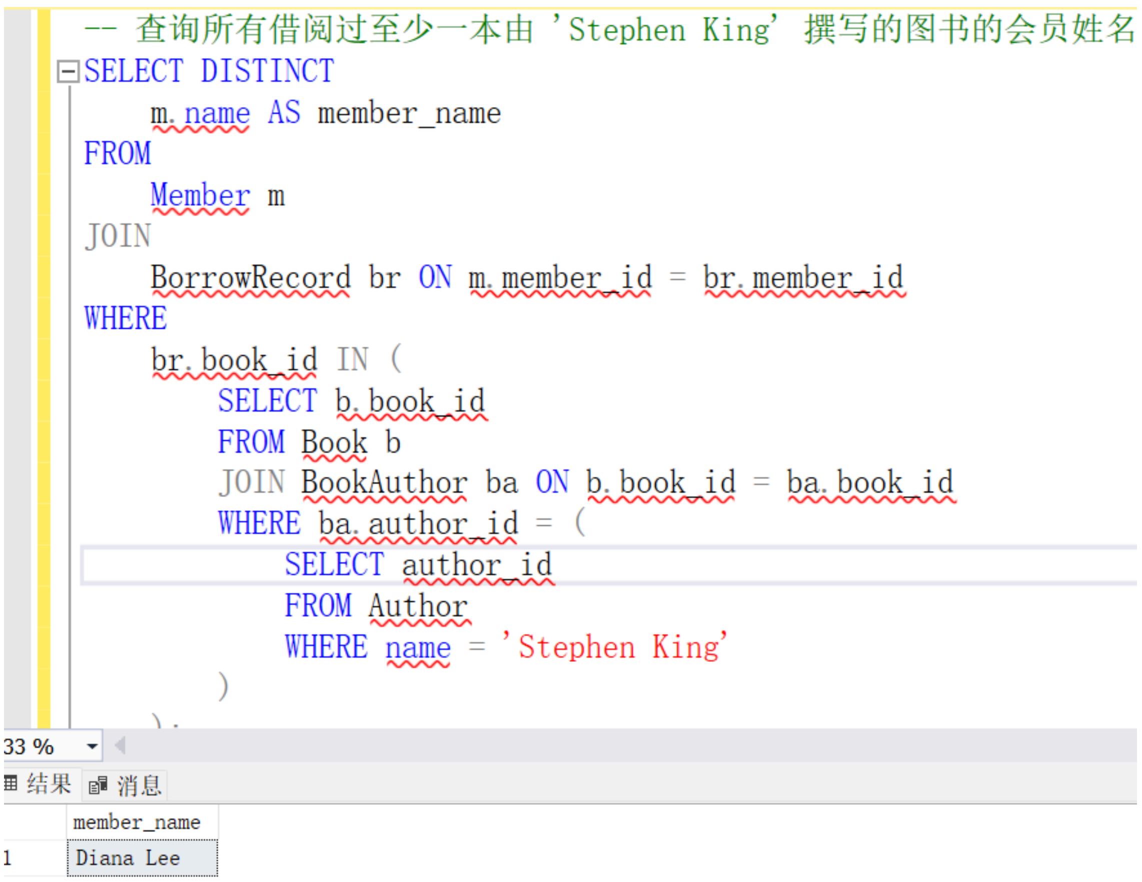
结果 消息

	publisher_name	total_books	average_price
1	HarperCollins	3	11.230000
2	Macmillan Publishers	2	19.250000
3	Penguin Random House	2	15.120000
4	Simon & Schuster	2	16.370000

- 嵌套查询
  a. 查询所有借阅过至少一本由 'Stephen King' 撰写的图书的会员姓名
    - SQL 语句：

```sql
SELECT DISTINCT
    m.name AS member_name
FROM
    Member m
JOIN
    BorrowRecord br ON m.member_id = br.member_id
WHERE
    br.book_id IN (
        SELECT b.book_id
        FROM Book b
        JOIN BookAuthor ba ON b.book_id = ba.book_id
        WHERE ba.author_id = (
            SELECT author_id
            FROM Author
            WHERE name = 'Stephen King'
        )
    );
```

- 执行图示：



```sql
-- 查询所有借阅过至少一本由 'Stephen King' 撰写的图书的会员姓名
SELECT DISTINCT
    m.name AS member_name
FROM
    Member m
JOIN
    BorrowRecord br ON m.member_id = br.member_id
WHERE
    br.book_id IN (
        SELECT b.book_id
        FROM Book b
        JOIN BookAuthor ba ON b.book_id = ba.book_id
        WHERE ba.author_id = (
            SELECT author_id
            FROM Author
            WHERE name = 'Stephen King'
        )
    );
```

33 %

结果 消息

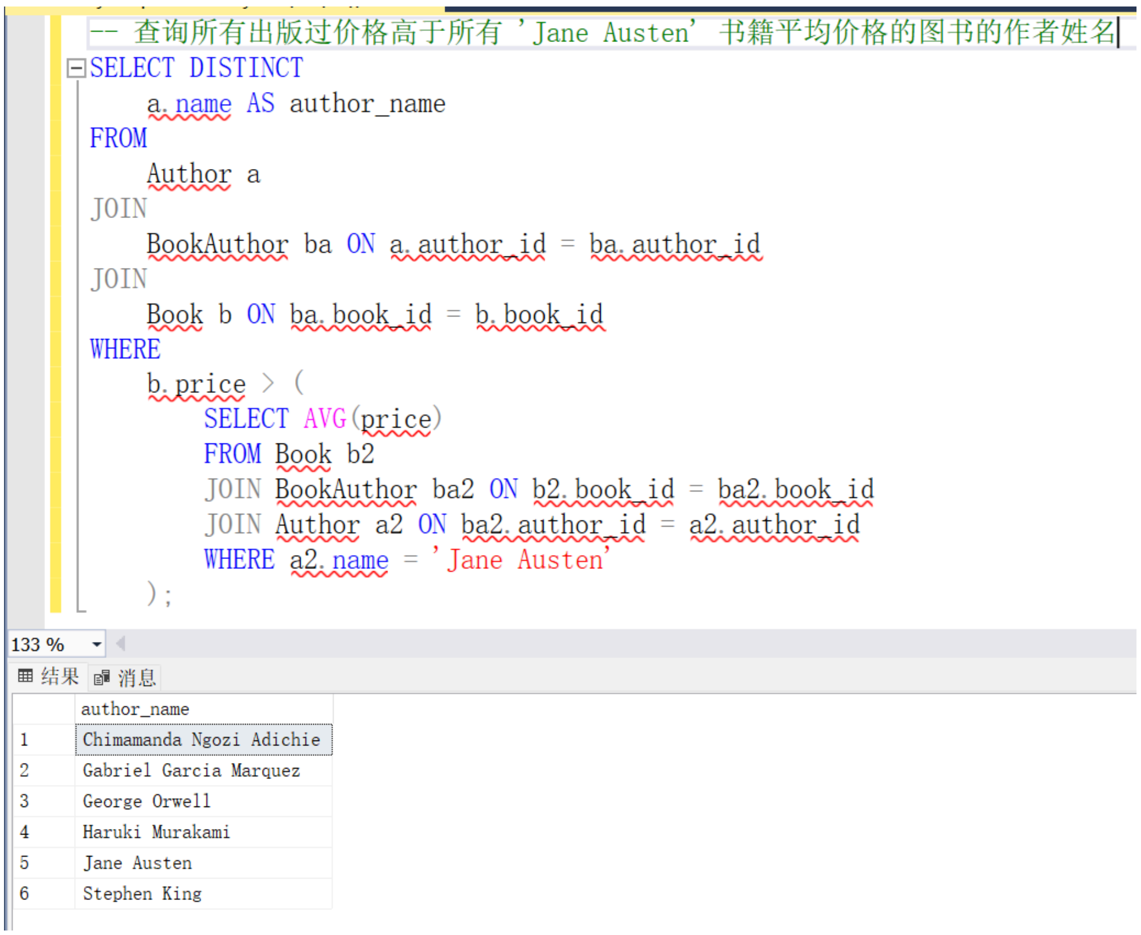
```
   member_name
1  Diana Lee
```

b. 查询所有出版过价格高于所有 'Jane Austen' 书籍平均价格的图书的作者姓名

- SQL 语句：

```sql
SELECT DISTINCT
    a.name AS author_name
FROM
    Author a
JOIN
    BookAuthor ba ON a.author_id = ba.author_id
JOIN
    Book b ON ba.book_id = b.book_id
WHERE
    b.price > (
        SELECT AVG(price)
        FROM Book b2
        JOIN BookAuthor ba2 ON b2.book_id = ba2.book_id
        JOIN Author a2 ON ba2.author_id = a2.author_id
        WHERE a2.name = 'Jane Austen'
    );
```

- 执行图示：



```sql
-- 查询所有出版过价格高于所有 'Jane Austen' 书籍平均价格的图书的作者姓名
SELECT DISTINCT
    a.name AS author_name
FROM
    Author a
JOIN
    BookAuthor ba ON a.author_id = ba.author_id
JOIN
    Book b ON ba.book_id = b.book_id
WHERE
    b.price > (
        SELECT AVG(price)
        FROM Book b2
        JOIN BookAuthor ba2 ON b2.book_id = ba2.book_id
        JOIN Author a2 ON ba2.author_id = a2.author_id
        WHERE a2.name = 'Jane Austen'
    );
```

133 %

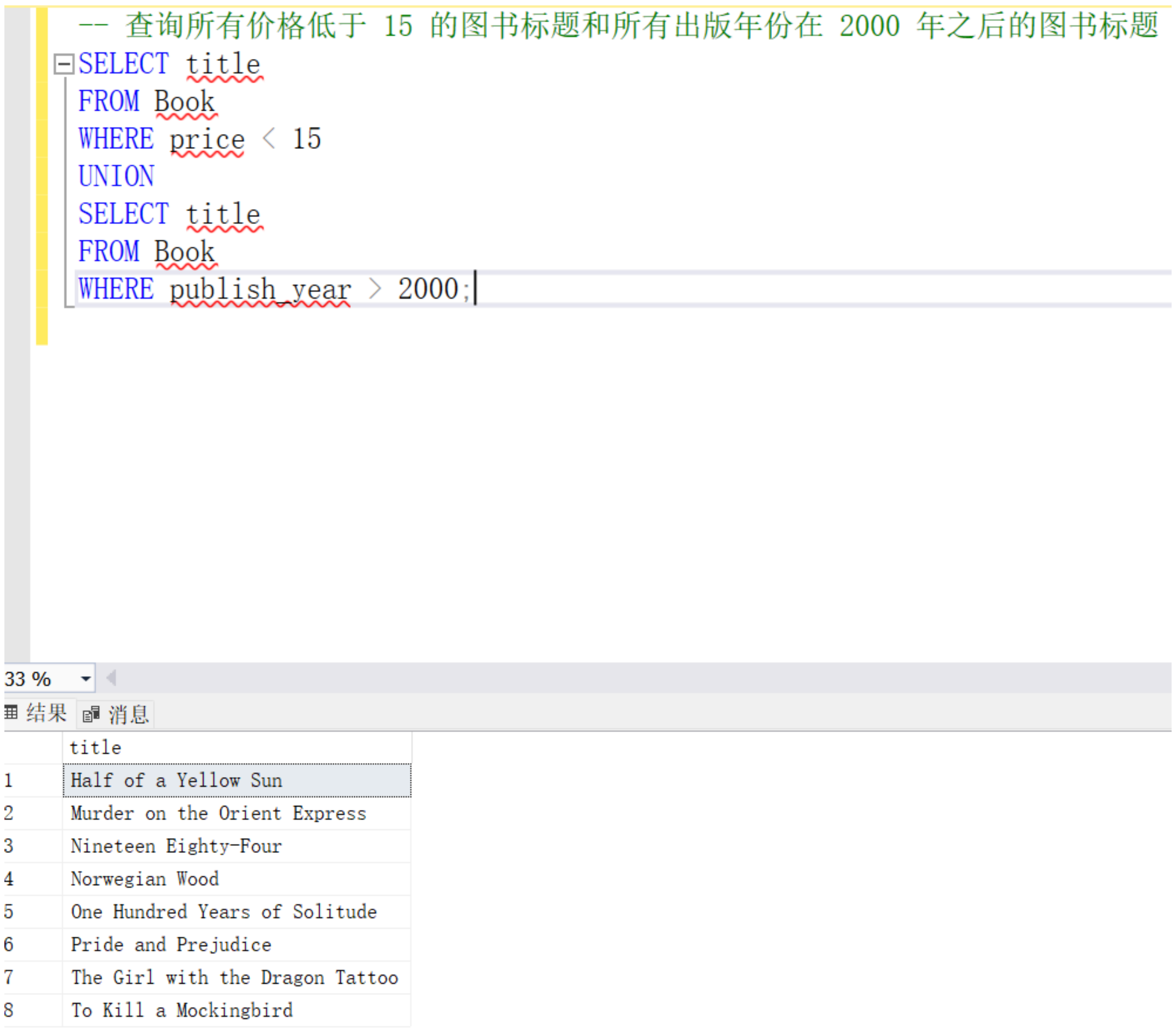
结果 消息

	author_name
1	Chimamanda Ngozi Adichie
2	Gabriel Garcia Marquez
3	George Orwell
4	Haruki Murakami
5	Jane Austen
6	Stephen King

- 联合查询：
  - SQL 语句：

```sql
SELECT title
FROM Book
WHERE price < 15
UNION
SELECT title
FROM Book
WHERE publish_year > 2000;
```

- 执行图示：



```sql
-- 查询所有价格低于 15 的图书标题和所有出版年份在 2000 年之后的图书标题
SELECT title
FROM Book
WHERE price < 15
UNION
SELECT title
FROM Book
WHERE publish_year > 2000;
```

	title
1	Half of a Yellow Sun
2	Murder on the Orient Express
3	Nineteen Eighty-Four
4	Norwegian Wood
5	One Hundred Years of Solitude
6	Pride and Prejudice
7	The Girl with the Dragon Tattoo
8	To Kill a Mockingbird

问题与解决方法

Q：插入数据时，想要在已插入的数据之后再添加数据，但是显示无法添加。

A：分析后发现新的语句与原数据主键有冲突，因此选择修改新的语句中的对应主键，完成数据添加。

Q：查询时无法通过书名直接找到作者。

A：利用连接操作，通过相同 `id` 查找。

实验总结

通过使用SQL对数据库进行各类操作，我真正通过实践实现了上课所学的知识，加深了对于数据库这门课程中SQL语句部分的理解与运用能力。