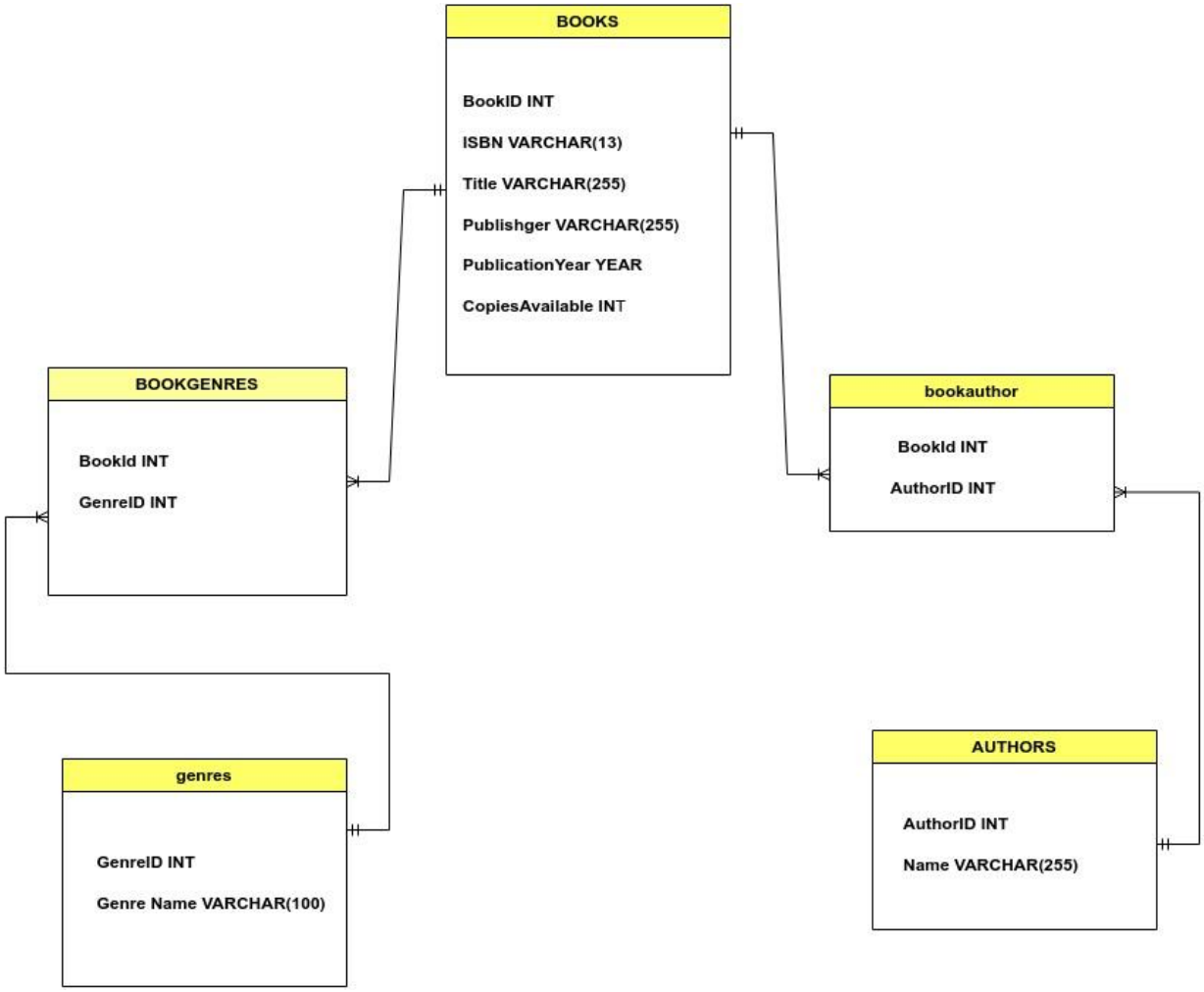
Name: Vaishnavi Vinod Ingole

Mail;:vaishnavingople54@gmail.com

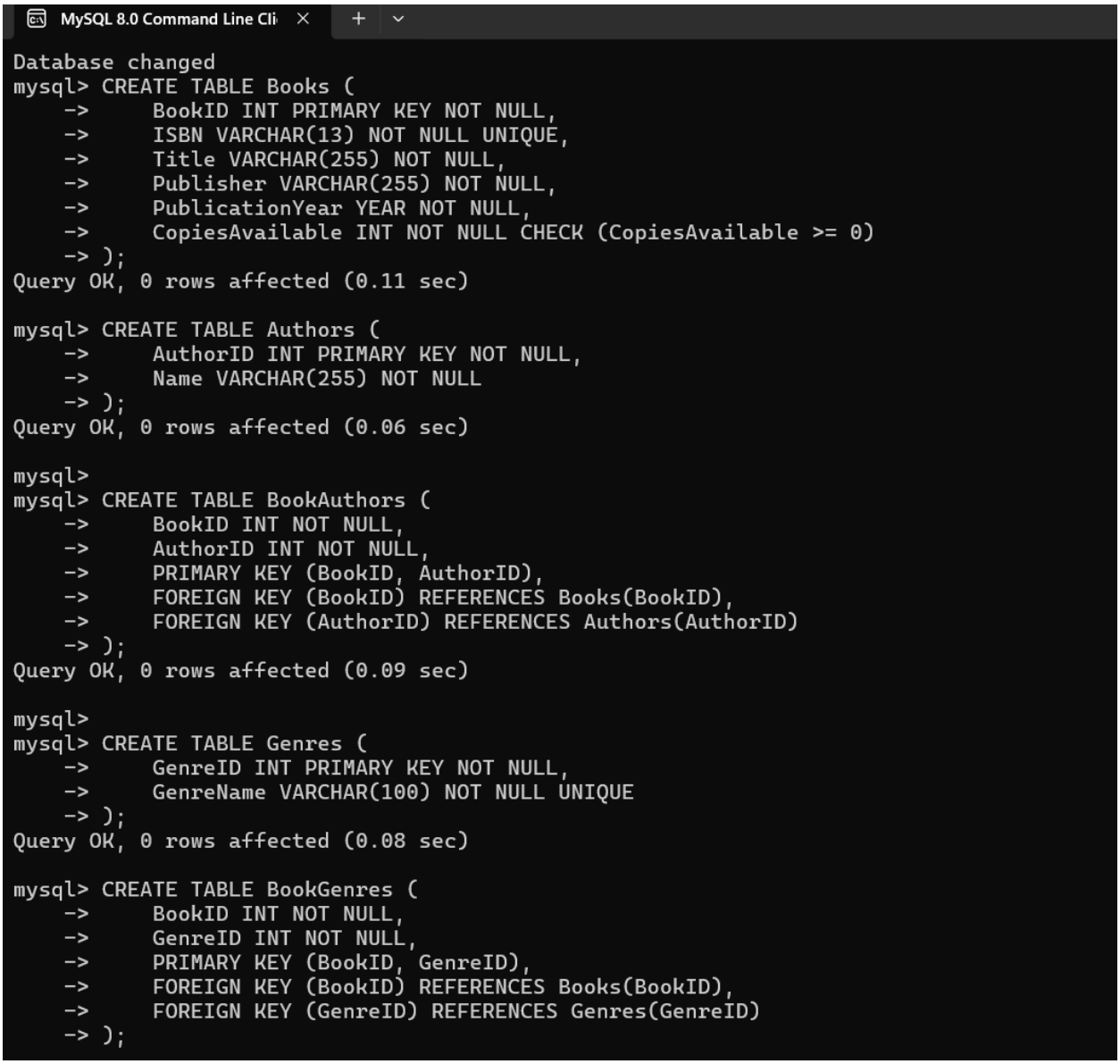
Assignment 1: Analyze a given business scenario and create an ER diagram that includes entities, relationships, attributes, and cardinality. Ensure that the diagram reflects proper normalization up to the third normal form.

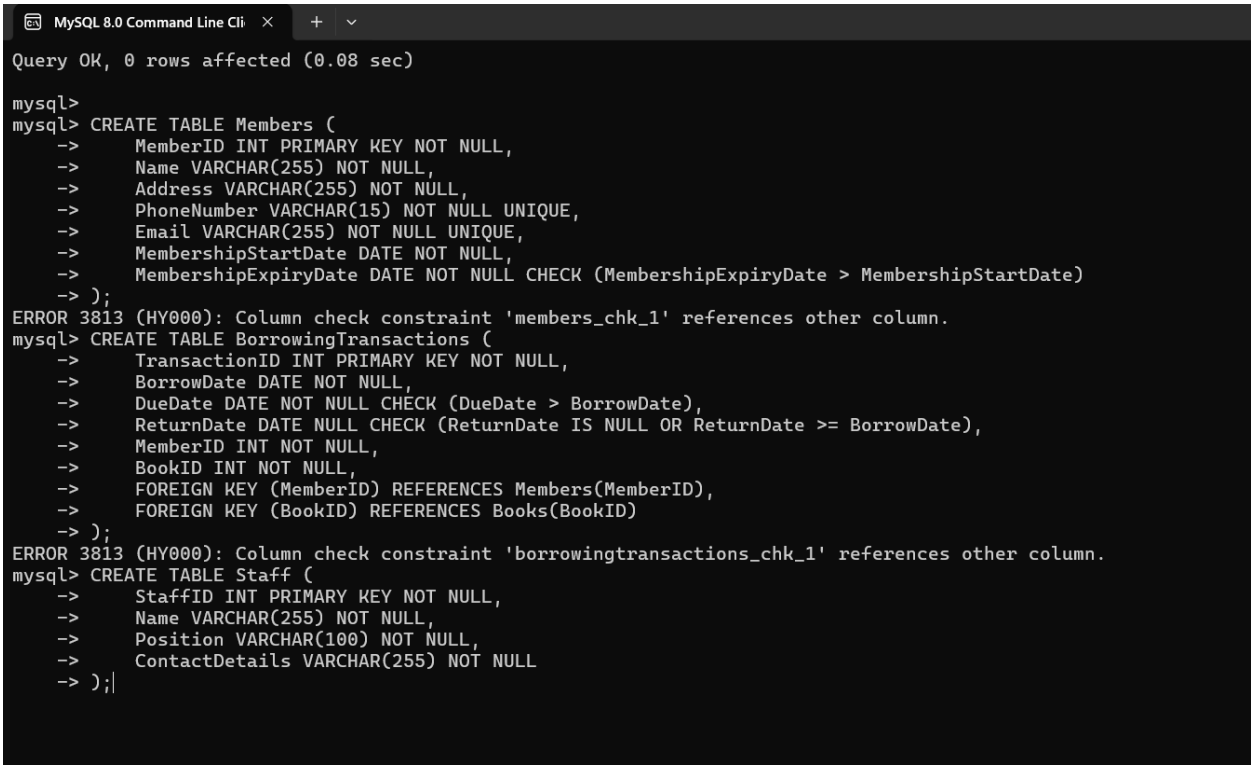
Solution:

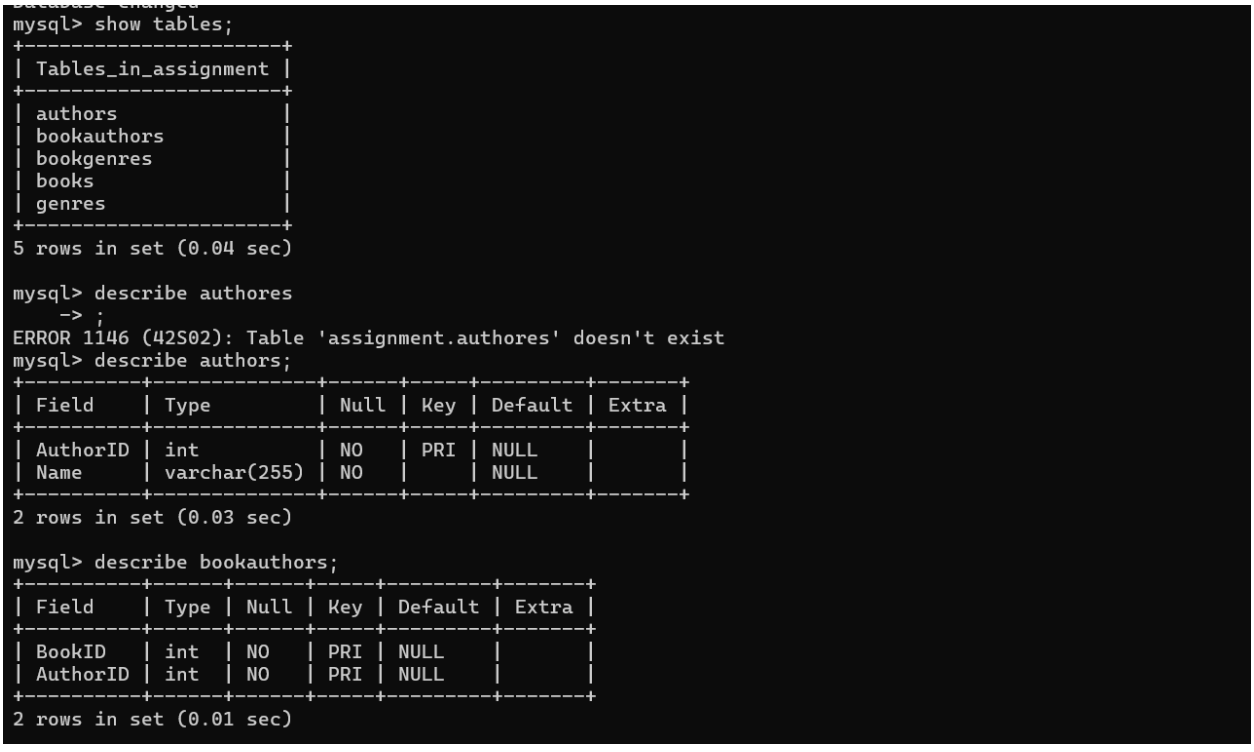


Assignment 2: Design a database schema for a library system, including tables, fields, and constraints like NOT NULL, UNIQUE, and CHECK. Include primary and foreign keys to establish relationships between tables.

Solution:




MySQL 8.0 Command Line Cli
Query OK, 0 rows affected (0.08 sec)
mysql>
mysql> CREATE TABLE Members (
-> MemberID INT PRIMARY KEY NOT NULL,
-> Name VARCHAR(255) NOT NULL,
-> Address VARCHAR(255) NOT NULL,
-> PhoneNumber VARCHAR(15) NOT NULL UNIQUE,
-> Email VARCHAR(255) NOT NULL UNIQUE,
-> MembershipStartDate DATE NOT NULL,
-> MembershipExpiryDate DATE NOT NULL CHECK (MembershipExpiryDate > MembershipStartDate)
->);
ERROR 3813 (HY000): Column check constraint 'members_chk_1' references other column.
mysql> CREATE TABLE BorrowingTransactions (
-> TransactionID INT PRIMARY KEY NOT NULL,
-> BorrowDate DATE NOT NULL,
-> DueDate DATE NOT NULL CHECK (DueDate > BorrowDate),
-> ReturnDate DATE NULL CHECK (ReturnDate IS NULL OR ReturnDate >= BorrowDate),
-> MemberID INT NOT NULL,
-> BookID INT NOT NULL,
-> FOREIGN KEY (MemberID) REFERENCES Members(MemberID),
-> FOREIGN KEY (BookID) REFERENCES Books(BookID)
->);
ERROR 3813 (HY000): Column check constraint 'borrowingtransactions_chk_1' references other column.
mysql> CREATE TABLE Staff (
-> StaffID INT PRIMARY KEY NOT NULL,
-> Name VARCHAR(255) NOT NULL,
-> Position VARCHAR(100) NOT NULL,
-> ContactDetails VARCHAR(255) NOT NULL
->);


mysql> show tables;
+----------------------+
| Tables_in_assignment |
+----------------------+
| authors |
| bookauthors |
| bookgenres |
| books |
| genres |
+----------------------+
5 rows in set (0.04 sec)
mysql> describe authores
-> ;
ERROR 1146 (42S02): Table 'assignment.authores' doesn't exist
mysql> describe authors;
+----------+--------------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| AuthorID | int | NO | PRI | NULL | |
| Name | varchar(255) | NO | | NULL | |
+----------+--------------+------+-----+---------+-------+
2 rows in set (0.03 sec)
mysql> describe bookauthors;
+----------+------+------+-----+---------+-------+
| Field | Type | Null | Key | Default | Extra |
+----------+------+------+-----+---------+-------+
| BookID | int | NO | PRI | NULL | |
| AuthorID | int | NO | PRI | NULL | |
+----------+------+------+-----+---------+-------+
2 rows in set (0.01 sec)

```
mysql> describe bookgeners;
ERROR 1146 (42S02): Table 'assignment.bookgeners' doesn't exist
mysql> describe bookgenres;
+---------+------+------+-----+---------+-------+
| Field   | Type | Null | Key | Default | Extra |
+---------+------+------+-----+---------+-------+
| BookID  | int  | NO   | PRI | NULL    |       |
| GenreID | int  | NO   | PRI | NULL    |       |
+---------+------+------+-----+---------+-------+
2 rows in set (0.01 sec)

mysql> describe books;
+-----------------+--------------+------+-----+---------+-------+
| Field           | Type         | Null | Key | Default | Extra |
+-----------------+--------------+------+-----+---------+-------+
| BookID          | int          | NO   | PRI | NULL    |       |
| ISBN            | varchar(13)  | NO   | UNI | NULL    |       |
| Title           | varchar(255) | NO   |     | NULL    |       |
| Publisher       | varchar(255) | NO   |     | NULL    |       |
| PublicationYear | year         | NO   |     | NULL    |       |
| CopiesAvailable | int          | NO   |     | NULL    |       |
+-----------------+--------------+------+-----+---------+-------+
6 rows in set (0.01 sec)

mysql> describe genres
    -> ;
+-----------+--------------+------+-----+---------+-------+
| Field     | Type         | Null | Key | Default | Extra |
+-----------+--------------+------+-----+---------+-------+
| GenreID   | int          | NO   | PRI | NULL    |       |
| GenreName | varchar(100) | NO   | UNI | NULL    |       |
+-----------+--------------+------+-----+---------+-------+
2 rows in set (0.01 sec)
```

Assignment 3: Explain the ACID properties of a transaction in your own words. Write SQL statements to simulate a transaction that includes locking and demonstrate different isolation levels to show concurrency control.

ACID Properties of a Transaction

magine a database as a big filing cabinet with folders for information. ACID properties make sure your filing stays neat and organized, even if things get a little messy.

- **All or Nothing (Atomicity):** This is like putting away all your papers in one folder at once. If you can't fit everything in, you take nothing out and leave the drawer how it was. No half-done filing!
- **Follow the Rules (Consistency):** This is like making sure everything in a folder goes together and makes sense. You wouldn't put socks in your tax folder, right? Transactions keep the data following the database's rules.
- **Don't Interrupt (Isolation):** Imagine two people trying to file at the same time. This property makes sure they don't see each other's changes until one finishes. No more mixed-up filing!
- **Remember Forever (Durability):** Once you file something important, you want to be sure it stays put. This property makes sure the changes from a transaction are saved permanently, even if the power goes out. No losing important documents!

SQL Statements to Simulate a Transaction with Locking and Isolation Levels

Let's consider a simple bank transaction where one account transfers money to another. We'll demonstrate the transaction, locking, and different isolation levels.

Setup Tables

```
CREATE TABLE Accounts (
    AccountID INT PRIMARY KEY,
    Balance DECIMAL(10, 2) NOT NULL CHECK (Balance >= 0)
);

INSERT INTO Accounts (AccountID, Balance) VALUES (1, 1000.00), (2, 500.00);

START TRANSACTION;

SELECT Balance FROM Accounts WHERE AccountID = 1 FOR UPDATE;

UPDATE Accounts SET Balance = Balance - 100 WHERE AccountID = 1;

SELECT Balance FROM Accounts WHERE AccountID = 2 FOR UPDATE;

UPDATE Accounts SET Balance = Balance + 100 WHERE AccountID = 2;
```

COMMIT;

Demonstrate Different Isolation Levels

Isolation levels control the visibility of changes made by one transaction to other concurrent transactions. Here are the four standard isolation levels:

READ UNCOMMITTED:

Lowest level, where transactions can read uncommitted changes made by other transactions (dirty reads).

READ COMMITTED:

Default level in many databases, where transactions can only read committed changes (no dirty reads).

REPEATABLE READ:

Ensures that if a transaction reads a row, it will see the same value throughout the transaction (prevents non-repeatable reads).

SERIALIZABLE:

Highest level, where transactions are completely isolated from each other, effectively serializing them.

Example: Setting and Testing Isolation Levels

SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED;

START TRANSACTION;

SELECT Balance FROM Accounts WHERE AccountID = 1;

COMMIT;

SET TRANSACTION ISOLATION LEVEL READ COMMITTED;

START TRANSACTION;

SELECT Balance FROM Accounts WHERE AccountID = 1;

COMMIT;

```
SET TRANSACTION ISOLATION LEVEL REPEATABLE READ;
START TRANSACTION;

SELECT Balance FROM Accounts WHERE AccountID = 1;
UPDATE Accounts SET Balance = Balance + 50 WHERE AccountID = 1;
SELECT Balance FROM Accounts WHERE AccountID = 1;
COMMIT;

SET TRANSACTION ISOLATION LEVEL SERIALIZABLE;
START TRANSACTION;

SELECT Balance FROM Accounts WHERE AccountID = 1;
COMMIT;
```

These SQL statements demonstrate how to manage transactions using locking and how different isolation levels impact transaction behavior and concurrency control.

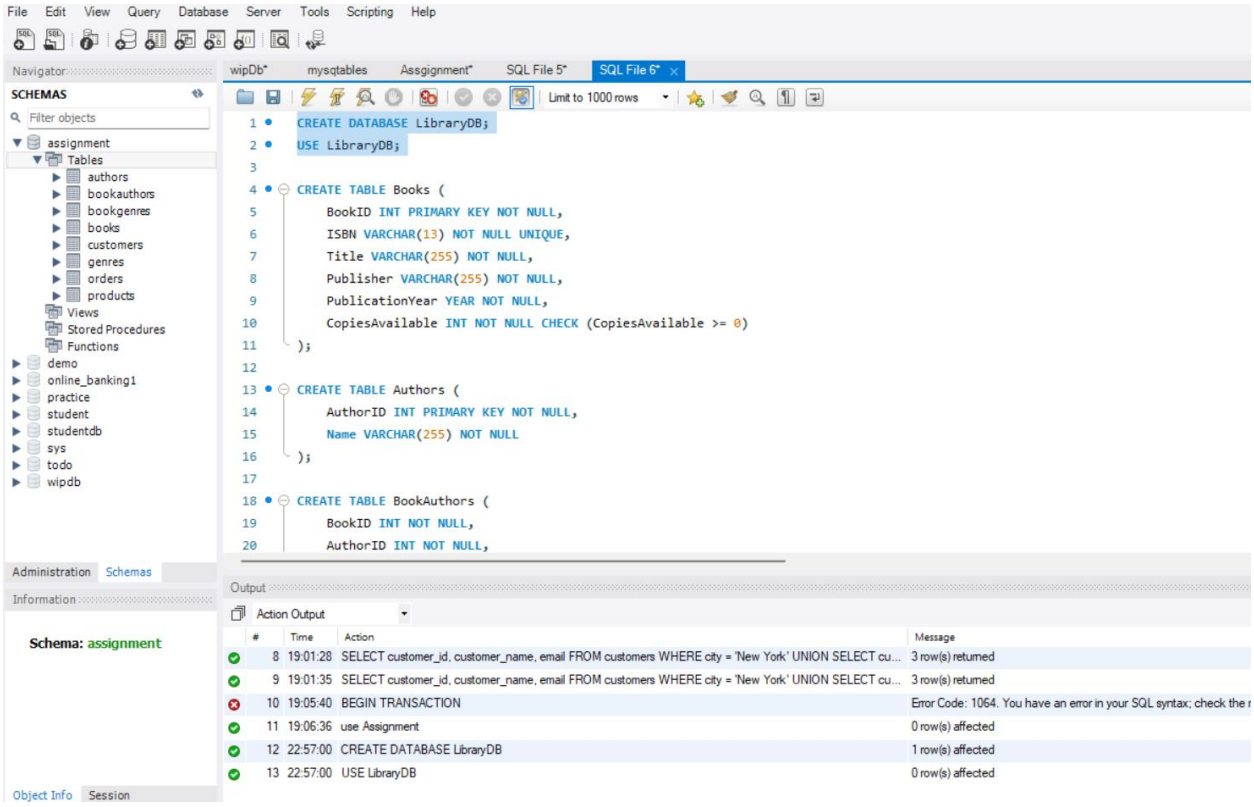
Assignment 4: Write SQL statements to CREATE a new database and tables that reflect the library schema you designed earlier. Use ALTER statements to modify the table structures and DROP statements to remove a redundant table.

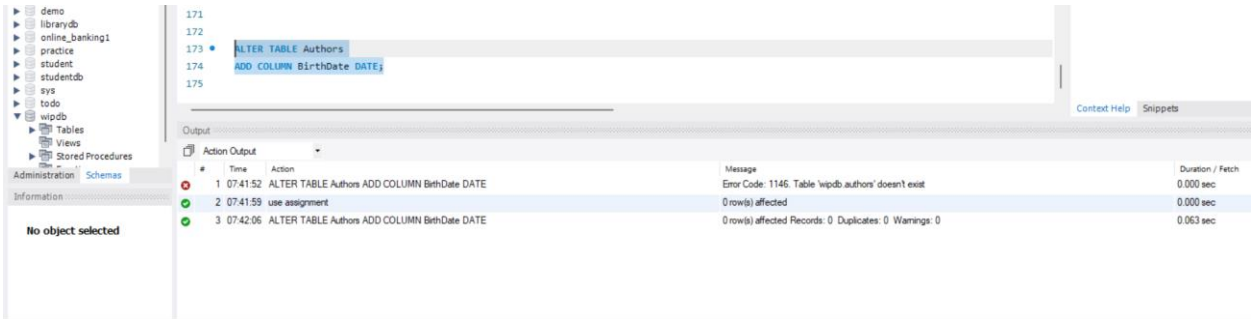
Solution:

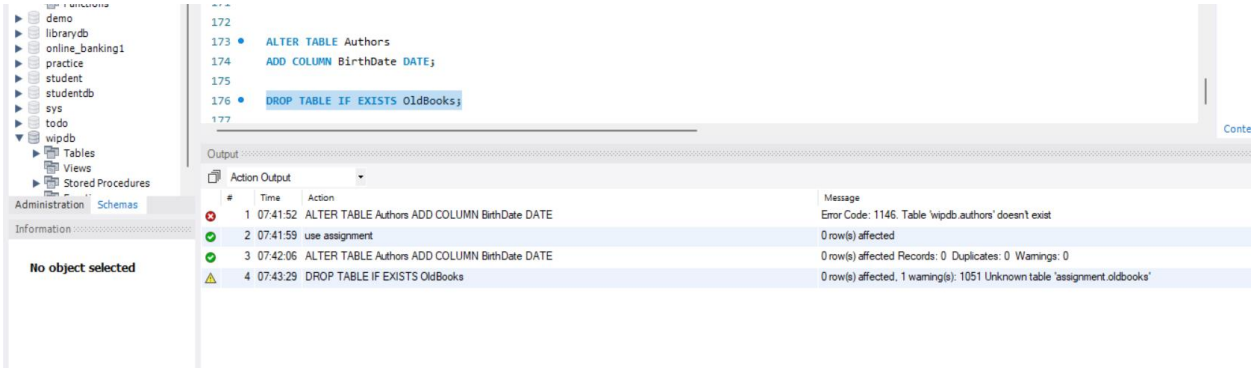
```
CREATE TABLE Books (
    BookID INT PRIMARY KEY NOT NULL,
```

```
    ISBN VARCHAR(13) NOT NULL UNIQUE,
    Title VARCHAR(255) NOT NULL,
    Publisher VARCHAR(255) NOT NULL,
    PublicationYear YEAR NOT NULL,
    CopiesAvailable INT NOT NULL CHECK (CopiesAvailable >= 0)
);

CREATE TABLE Authors (
    AuthorID INT PRIMARY KEY NOT NULL,
    Name VARCHAR(255) NOT NULL
);

CREATE TABLE BookAuthors (
    BookID INT NOT NULL,
    AuthorID INT NOT NULL,
    PRIMARY KEY (BookID, AuthorID),
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
    FOREIGN KEY (AuthorID) REFERENCES Authors(AuthorID)
);

CREATE TABLE Genres (
    GenreID INT PRIMARY KEY NOT NULL,
    GenreName VARCHAR(100) NOT NULL UNIQUE
);

CREATE TABLE BookGenres (
    BookID INT NOT NULL,
    GenreID INT NOT NULL,
    PRIMARY KEY (BookID, GenreID),
    FOREIGN KEY (BookID) REFERENCES Books(BookID),
```

```
    FOREIGN KEY (GenreID) REFERENCES Genres(GenreID)
);
ALTER TABLE Authors
ADD COLUMN BirthDate DATE;

DROP TABLE IF EXISTS OldBooks;
```







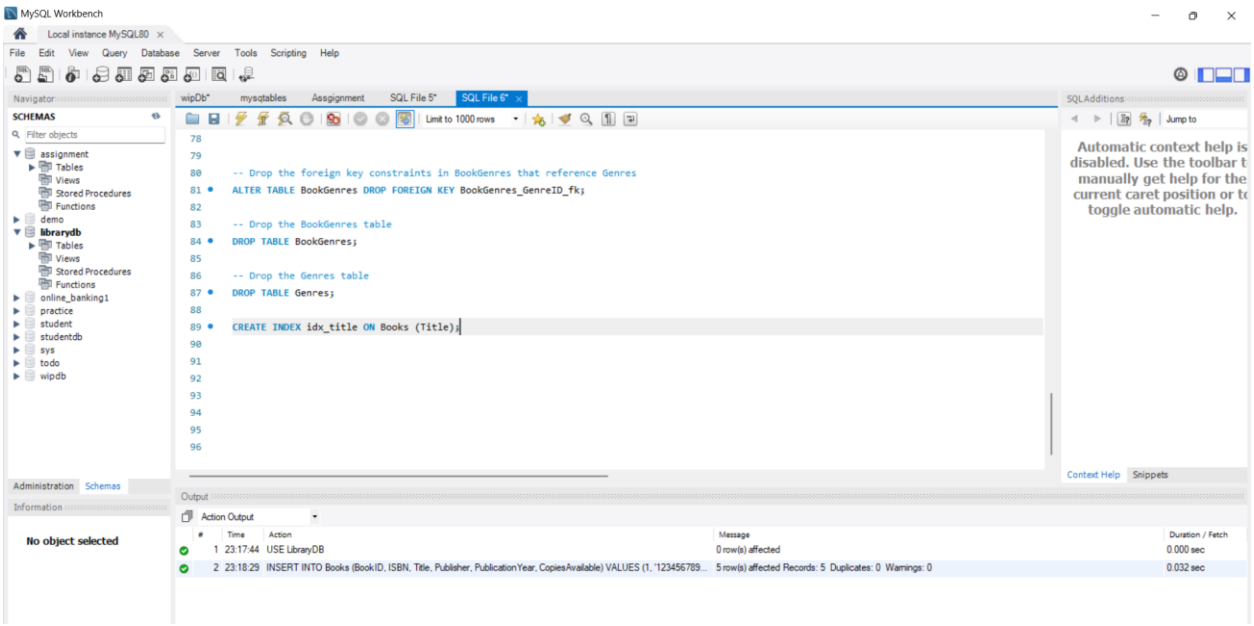
Assignment 5: Demonstrate the creation of an index on a table and discuss how it improves query performance. Use a DROP INDEX statement to remove the index and analyze the impact on query execution.

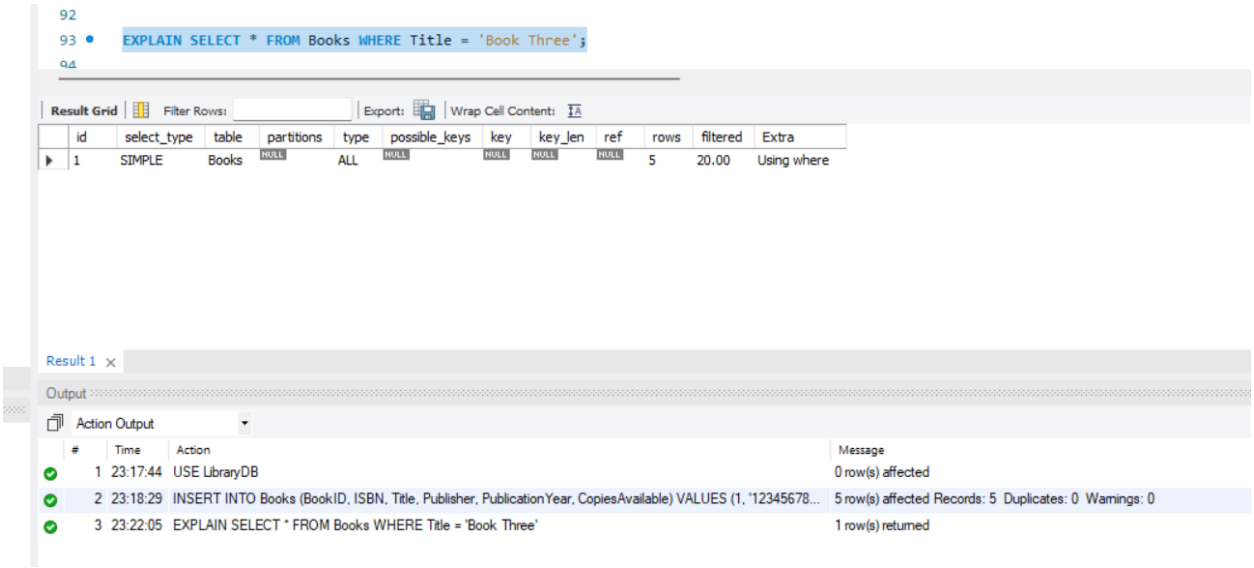
Solution:

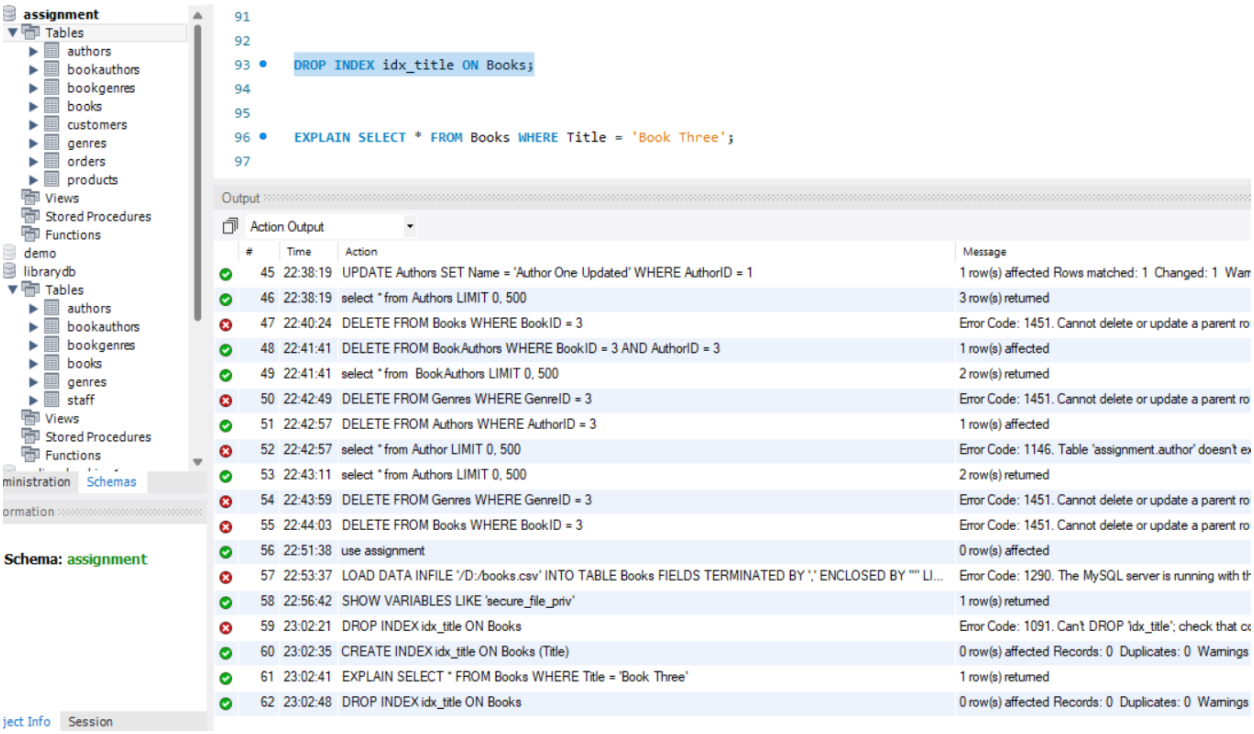
DROP INDEX idx_title ON Books;

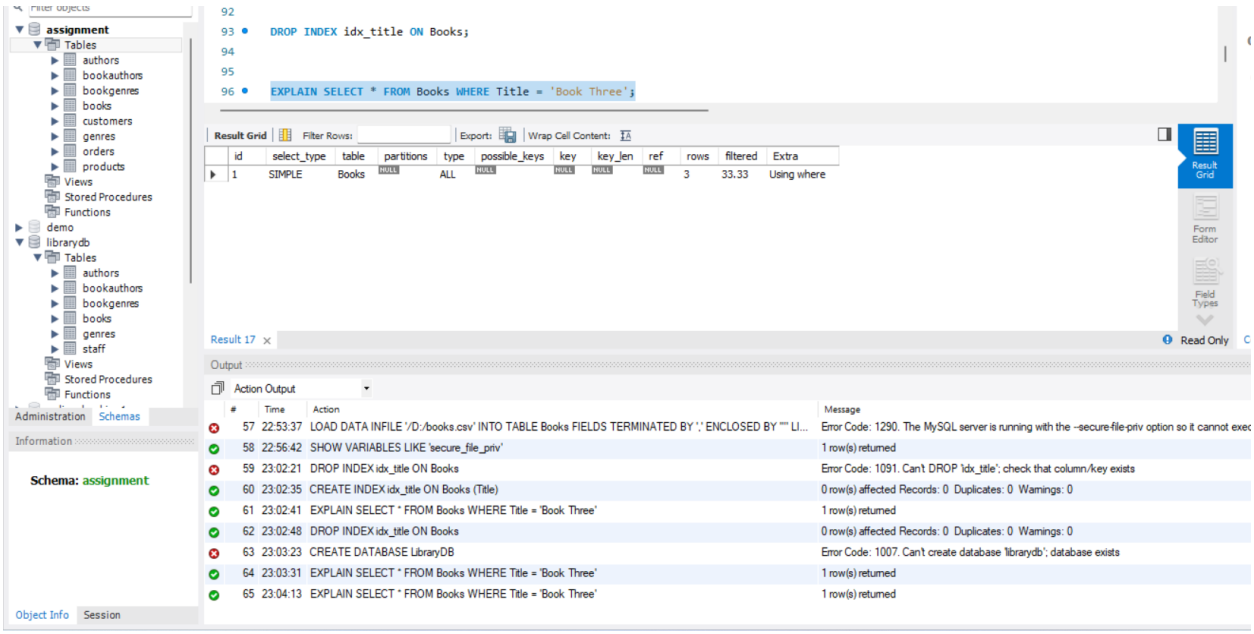
EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three';

CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass';


MySQL Workbench
Local instance MySQL80
File Edit View Query Database Server Tools Scripting Help
Navigator
SCHEMAS
Filter objects
assignment
Tables
Views
Stored Procedures
Functions
demo
librarydb
Tables
Views
Stored Procedures
Functions
online_banking1
practice
student
studentdb
sys
todo
wipdb
Administration Schemas
Information
No object selected
wipDb* mysqtables Assignment SQL File 5* SQL File 6*
Limit to 1000 rows
78
79
80 -- Drop the foreign key constraints in BookGenres that reference Genres
81 ALTER TABLE BookGenres DROP FOREIGN KEY BookGenres_GenreID_fk;
82
83 -- Drop the BookGenres table
84 DROP TABLE BookGenres;
85
86 -- Drop the Genres table
87 DROP TABLE Genres;
88
89 CREATE INDEX idx_title ON Books (Title);
90
91
92
93
94
95
96
SQLAdditions
Jump to
Automatic context help is disabled. Use the toolbar t manually get help for the current caret position or t toggle automatic help.
Context Help Snippets
Output
Action Output
Time Action Message Duration / Fetch
1 23:17:44 USE LibraryDB 0 row(s) affected 0.000 sec
2 23:18:29 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '123456789... 5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0 0.032 sec


92
93 EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three';
94
Result Grid Filter Rows: Export: Wrap Cell Content:
id select_type table partitions type possible_keys key key_len ref rows filtered Extra
1 SIMPLE Books NULL ALL NULL NULL NULL NULL 5 20.00 Using where
Result 1
Output
Action Output
Time Action Message
1 23:17:44 USE LibraryDB 0 row(s) affected
2 23:18:29 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '12345678... 5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0
3 23:22:05 EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three' 1 row(s) returned





Without Index:

Full table scan (type = ALL).

High number of rows examined (rows = total number of rows in the table).

With Index:

Index lookup (type = ref or range).

Low number of rows examined (rows = number of matching rows).

After Dropping Index:

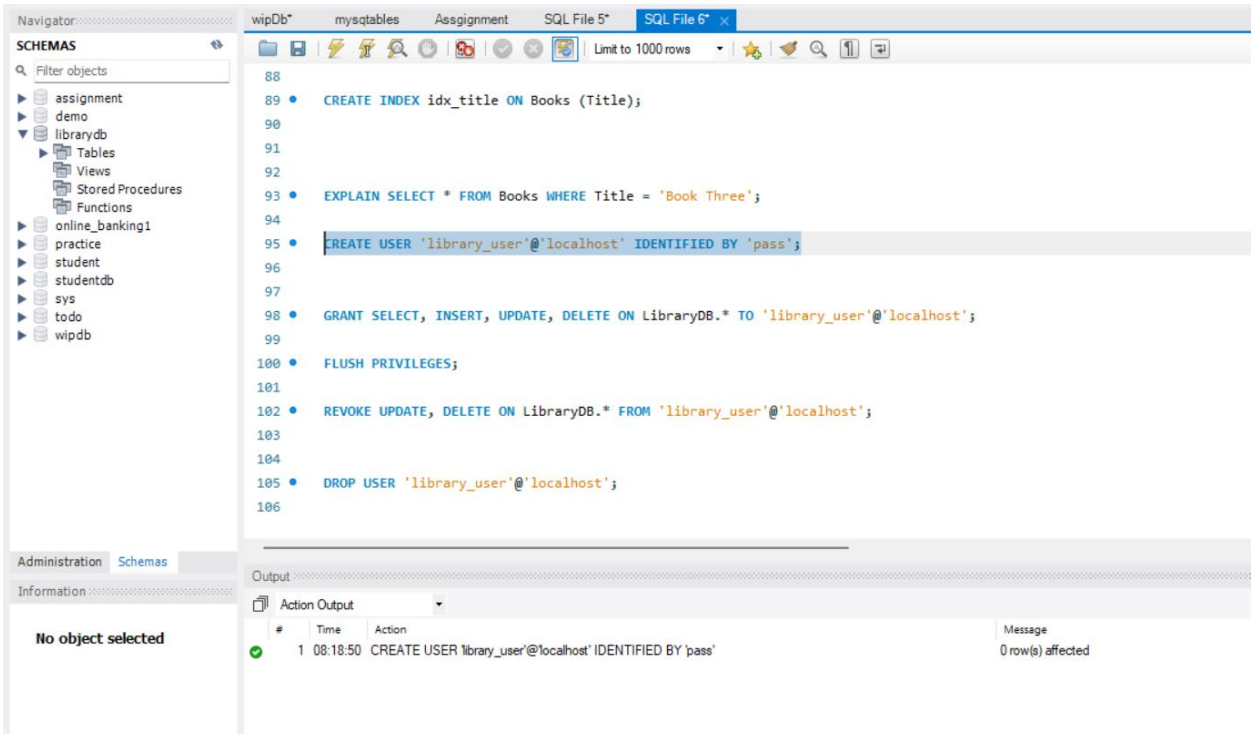
Reverts to full table scan (type = ALL).

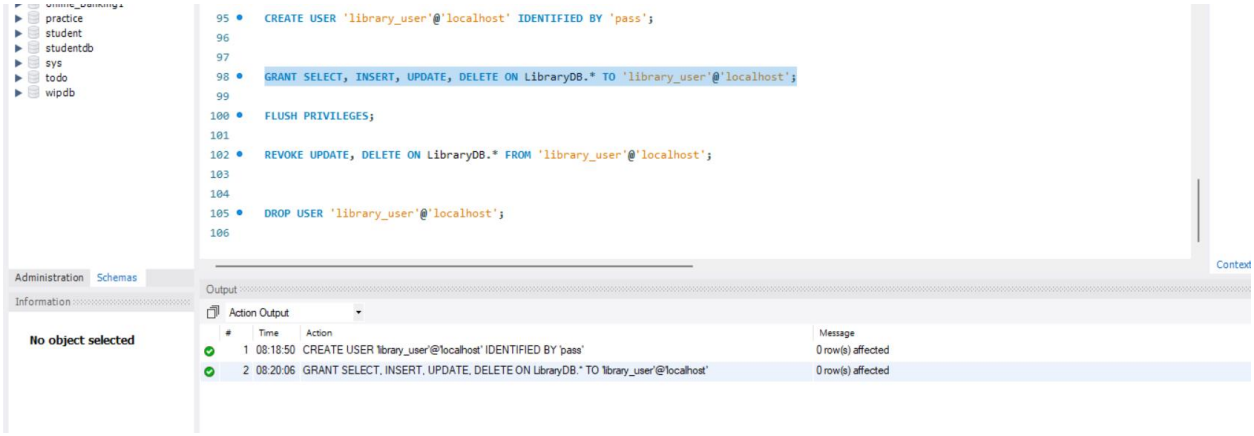
High number of rows examined again (rows = total number of rows in the table).

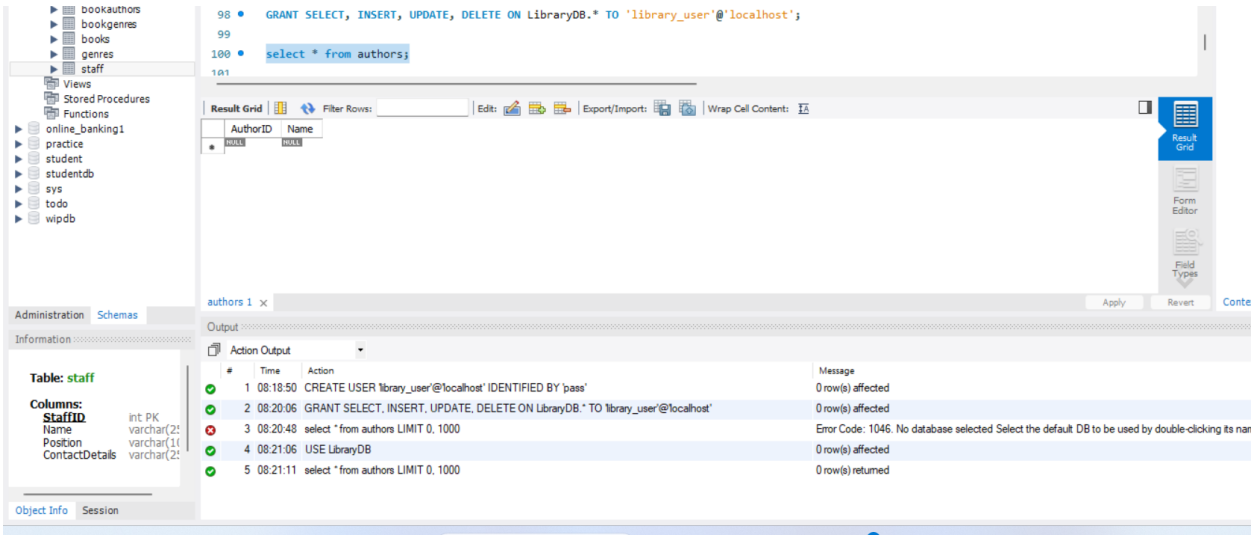
Assignment 6: Create a new database user with specific privileges using the CREATE USER and GRANT commands. Then, write a script to REVOKE certain privileges and DROP the user.

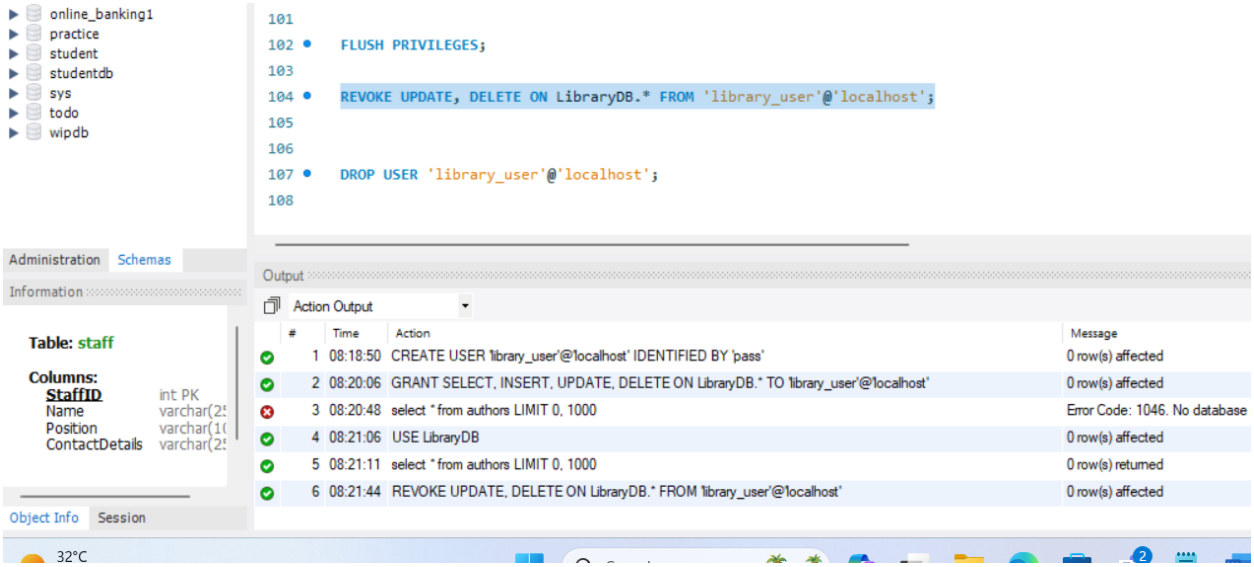
Solution:

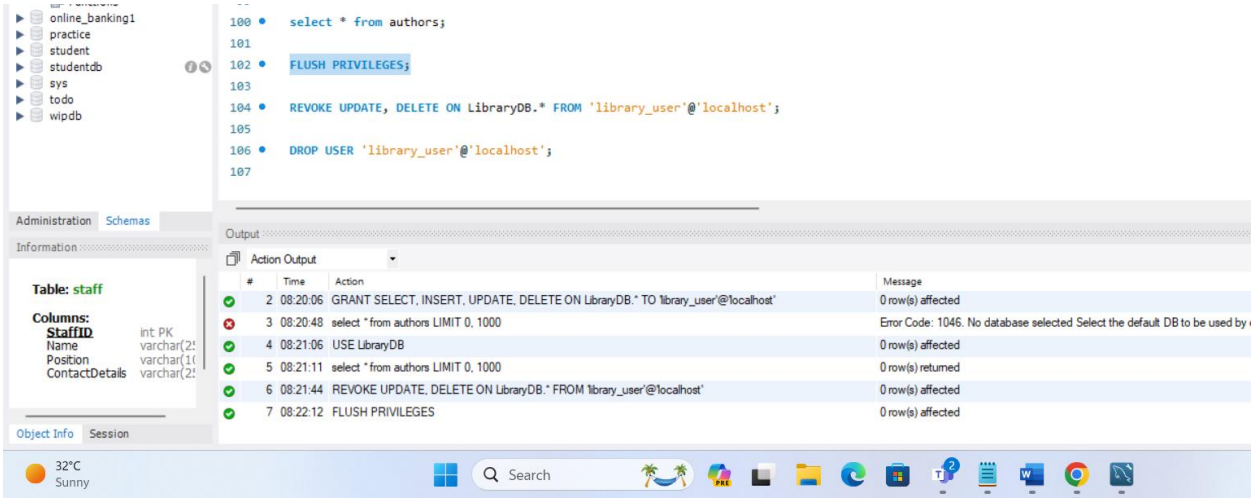
```
CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass';

GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';

select * from authors;

FLUSH PRIVILEGES;

REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';

DROP USER 'library_user'@'localhost';
```

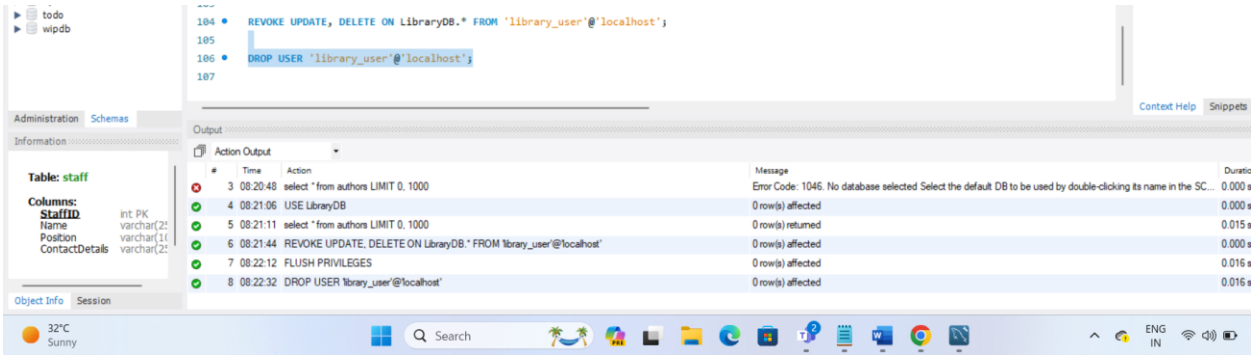

Navigator
SCHEMAS
Filter objects
assignment
demo
librarydb
Tables
Views
Stored Procedures
Functions
online_banking1
practice
student
studentdb
sys
todo
wipdb
Administration
Schemas
Information
No object selected
wipDb*
mysqtables
Assgignment
SQL File 5*
SQL File 6*
Limit to 1000 rows
88
89 CREATE INDEX idx_title ON Books (Title);
90
91
92
93 EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three';
94
95 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass';
96
97
98 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';
99
100 FLUSH PRIVILEGES;
101
102 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
103
104
105 DROP USER 'library_user'@'localhost';
106
Output
Action Output
Time Action Message
1 08:18:50 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass' 0 row(s) affected


practice
student
studentdb
sys
todo
wipdb
95 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass';
96
97
98 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';
99
100 FLUSH PRIVILEGES;
101
102 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
103
104
105 DROP USER 'library_user'@'localhost';
106
Context
Administration
Schemas
Information
No object selected
Output
Action Output
Time Action Message
1 08:18:50 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass' 0 row(s) affected
2 08:20:06 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost' 0 row(s) affected


bookauthors
bookgenres
books
genres
staff
Views
Stored Procedures
Functions
online_banking1
practice
student
studentdb
sys
todo
wipdb
98 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';
99
100 select * from authors;
101
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
AuthorID
Name
Result Grid
Form Editor
Field Types
authors 1
Apply
Revert
Administration
Schemas
Information
Table: staff
Columns:
StaffID int PK
Name varchar(2
Position varchar(1
ContactDetails varchar(2
Object Info
Session
Output
Action Output
Time Action Message
1 08:18:50 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass' 0 row(s) affected
2 08:20:06 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost' 0 row(s) affected
3 08:20:48 select * from authors LIMIT 0, 1000 Error Code: 1046. No database selected Select the default DB to be used by double-clicking its nam
4 08:21:06 USE LibraryDB 0 row(s) affected
5 08:21:11 select * from authors LIMIT 0, 1000 0 row(s) returned


online_banking1
practice
student
studentdb
sys
todo
wipdb
101
102 FLUSH PRIVILEGES;
103
104 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
105
106
107 DROP USER 'library_user'@'localhost';
108
Administration
Schemas
Information
Table: staff
Columns:
StaffID int PK
Name varchar(2
Position varchar(1
ContactDetails varchar(2
Object Info
Session
Output
Action Output
Time Action Message
1 08:18:50 CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass' 0 row(s) affected
2 08:20:06 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost' 0 row(s) affected
3 08:20:48 select * from authors LIMIT 0, 1000 Error Code: 1046. No database
4 08:21:06 USE LibraryDB 0 row(s) affected
5 08:21:11 select * from authors LIMIT 0, 1000 0 row(s) returned
6 08:21:44 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost' 0 row(s) affected
32°C


online_banking1
practice
student
studentdb
sys
todo
wipdb
100 select * from authors;
101
102 FLUSH PRIVILEGES;
103
104 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
105
106 DROP USER 'library_user'@'localhost';
107
Administration
Schemas
Information
Table: staff
Columns:
StaffID int PK
Name varchar(2
Position varchar(1
ContactDetails varchar(2
Object Info
Session
Output
Action Output
Time Action Message
2 08:20:06 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost' 0 row(s) affected
3 08:20:48 select * from authors LIMIT 0, 1000 Error Code: 1046. No database selected Select the default DB to be used by
4 08:21:06 USE LibraryDB 0 row(s) affected
5 08:21:11 select * from authors LIMIT 0, 1000 0 row(s) returned
6 08:21:44 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost' 0 row(s) affected
7 08:22:12 FLUSH PRIVILEGES 0 row(s) affected
32°C
Sunny
Search



Assignment 7: Prepare a series of SQL statements to INSERT new records into the library tables, UPDATE existing records with new information, and DELETE records based on specific criteria. Include BULK INSERT operations to load data from an external source.

Solution:

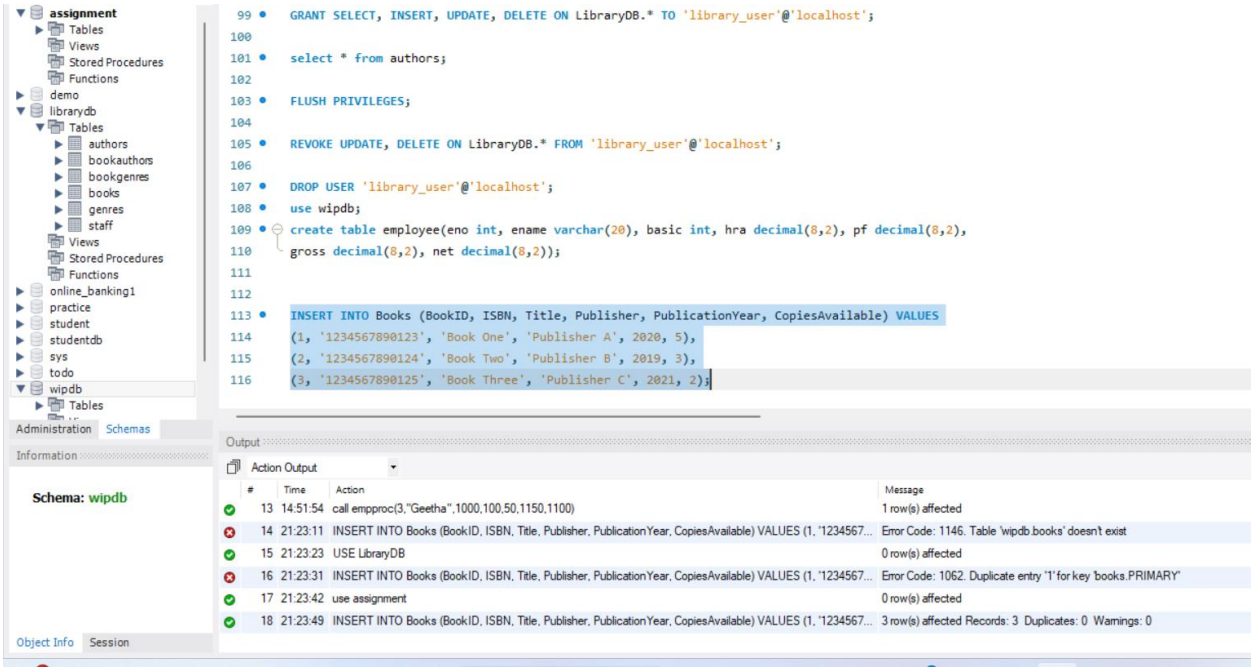
INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable)
VALUES

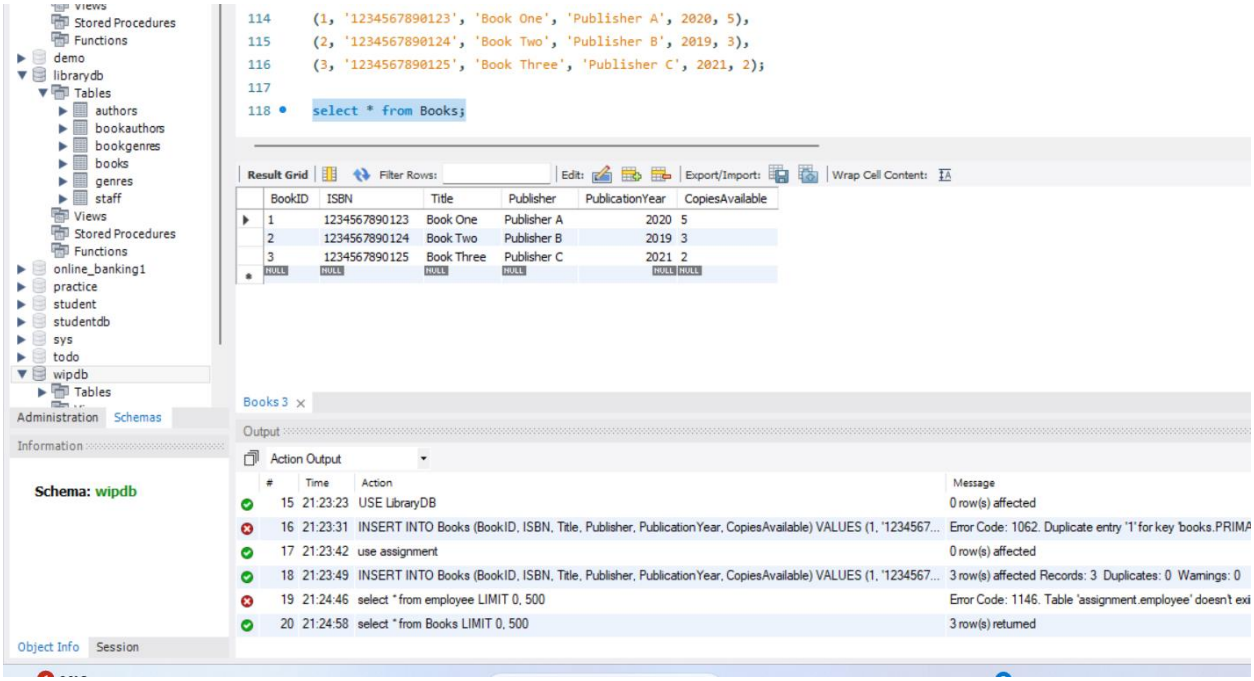
(1, '1234567890123', 'Book One', 'Publisher A', 2020, 5),

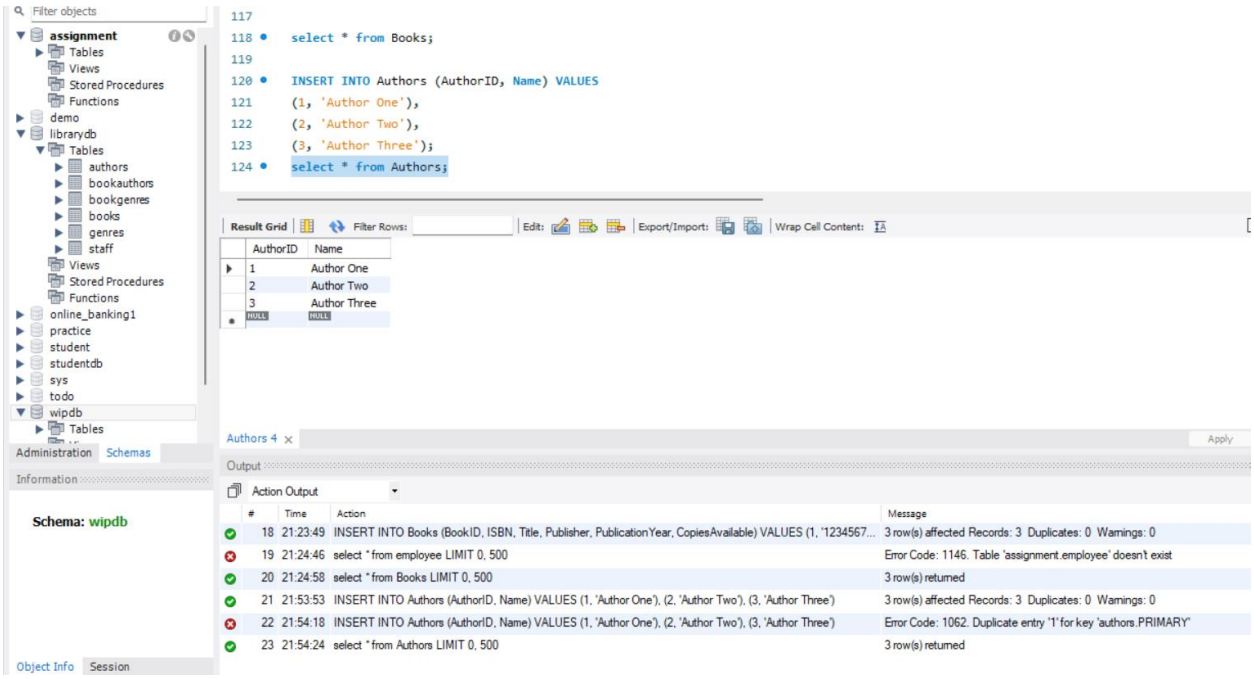
(2, '1234567890124', 'Book Two', 'Publisher B', 2019, 3),

(3, '1234567890125', 'Book Three', 'Publisher C', 2021, 2);

select * from Books;


assignment
Tables
Views
Stored Procedures
Functions
demo
librarydb
Tables
authors
bookauthors
bookgenres
books
genres
staff
Views
Stored Procedures
Functions
online_banking1
practice
student
studentdb
sys
todo
wipdb
Tables
Administration
Schemas
Information
Schema: wipdb
Object Info
Session
99 GRANT SELECT, INSERT, UPDATE, DELETE ON LibraryDB.* TO 'library_user'@'localhost';
100
101 select * from authors;
102
103 FLUSH PRIVILEGES;
104
105 REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
106
107 DROP USER 'library_user'@'localhost';
108 use wipdb;
109 create table employee(eno int, ename varchar(20), basic int, hra decimal(8,2), pf decimal(8,2),
110 gross decimal(8,2), net decimal(8,2));
111
112
113 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES
114 (1, '1234567890123', 'Book One', 'Publisher A', 2020, 5),
115 (2, '1234567890124', 'Book Two', 'Publisher B', 2019, 3),
116 (3, '1234567890125', 'Book Three', 'Publisher C', 2021, 2);
Output
Action Output
Time Action Message
13 14:51:54 call empproc(3,"Geetha",1000,100,50,1150,1100) 1 row(s) affected
14 21:23:11 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '1234567... Error Code: 1146. Table 'wipdb.books' doesn't exist
15 21:23:23 USE LibraryDB 0 row(s) affected
16 21:23:31 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '1234567... Error Code: 1062. Duplicate entry '1' for key 'books.PRIMARY'
17 21:23:42 use assignment 0 row(s) affected
18 21:23:49 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '1234567... 3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0


Stored Procedures
Functions
demo
librarydb
Tables
authors
bookauthors
bookgenres
books
genres
staff
Views
Stored Procedures
Functions
online_banking1
practice
student
studentdb
sys
todo
wipdb
Tables
Administration
Schemas
Information
Schema: wipdb
Object Info
Session
114 (1, '1234567890123', 'Book One', 'Publisher A', 2020, 5),
115 (2, '1234567890124', 'Book Two', 'Publisher B', 2019, 3),
116 (3, '1234567890125', 'Book Three', 'Publisher C', 2021, 2);
117
118 select * from Books;
Result Grid
Filter Rows:
Edit:
Export/Import:
Wrap Cell Content:
BookID ISBN Title Publisher PublicationYear CopiesAvailable
1 1234567890123 Book One Publisher A 2020 5
2 1234567890124 Book Two Publisher B 2019 3
3 1234567890125 Book Three Publisher C 2021 2
NULL NULL NULL NULL NULL NULL
Books 3
Output
Action Output
Time Action Message
15 21:23:23 USE LibraryDB 0 row(s) affected
16 21:23:31 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '1234567... Error Code: 1062. Duplicate entry '1' for key 'books.PRIMA
17 21:23:42 use assignment 0 row(s) affected
18 21:23:49 INSERT INTO Books (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable) VALUES (1, '1234567... 3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
19 21:24:46 select * from employee LIMIT 0, 500 Error Code: 1146. Table 'assignment.employee' doesn't exi
20 21:24:58 select * from Books LIMIT 0, 500 3 row(s) returned



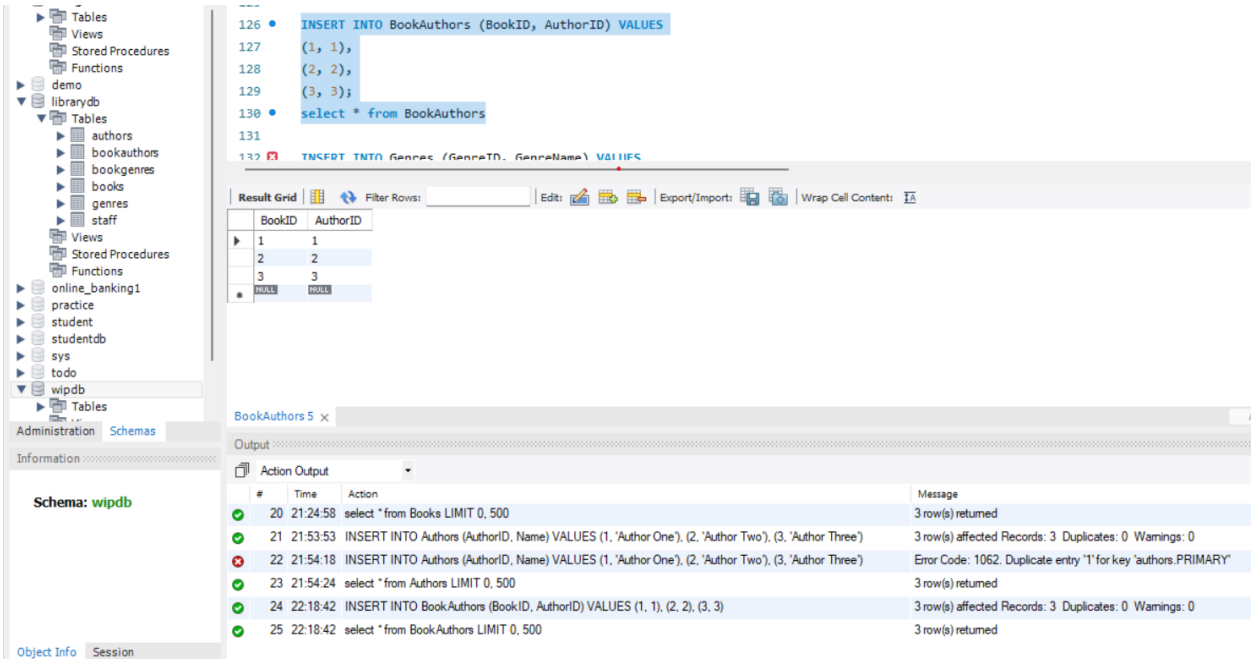
INSERT INTO BookAuthors (BookID, AuthorID) VALUES

(1, 1),

(2, 2),

(3, 3);

select * from BookAuthors



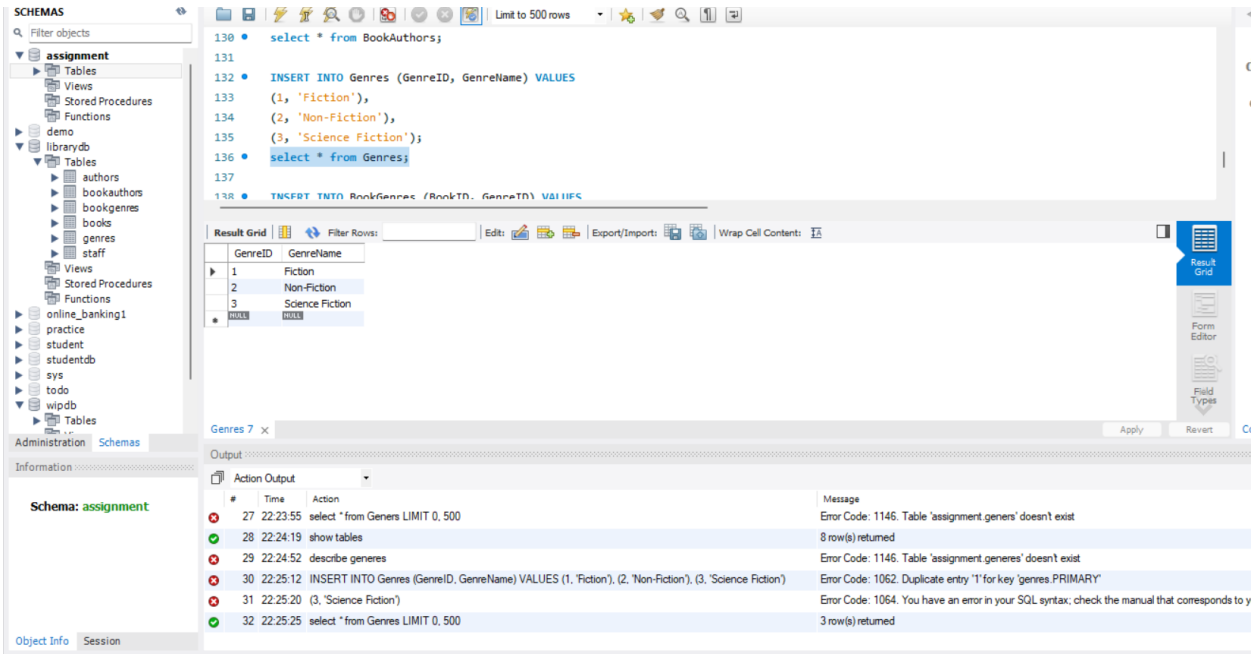
INSERT INTO Genres (GenreID, GenreName) VALUES

(1, 'Fiction'),

(2, 'Non-Fiction'),

(3, 'Science Fiction');

select * from Genres;



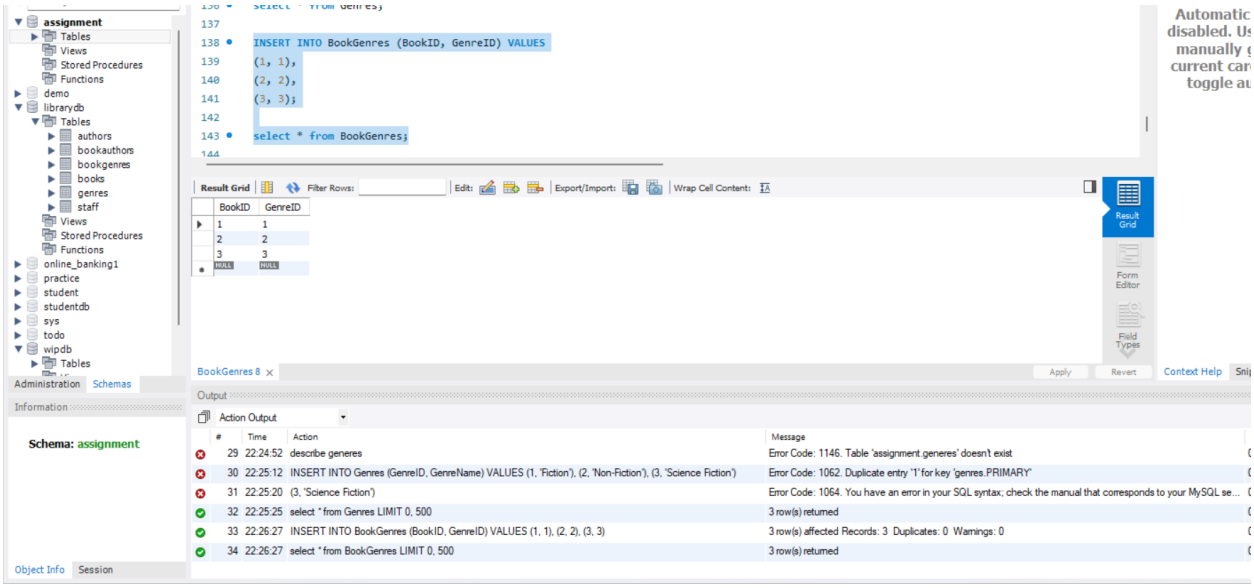
INSERT INTO BookGenres (BookID, GenreID) VALUES

(1, 1),

(2, 2),

(3, 3);

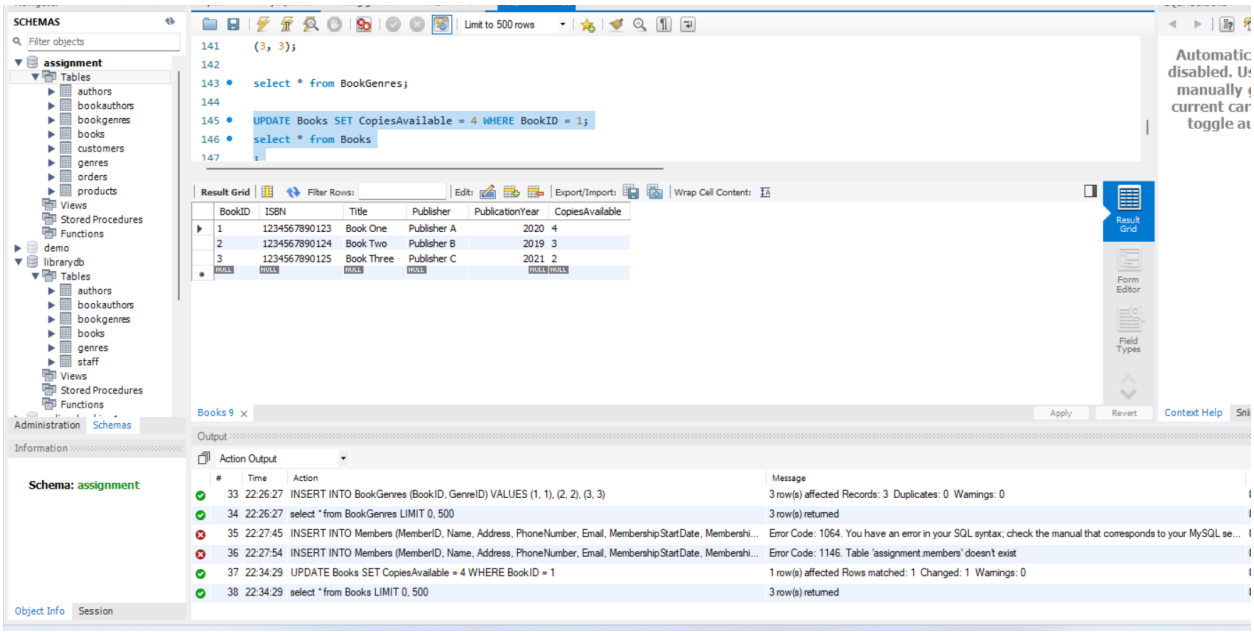
select * from BookGenres;

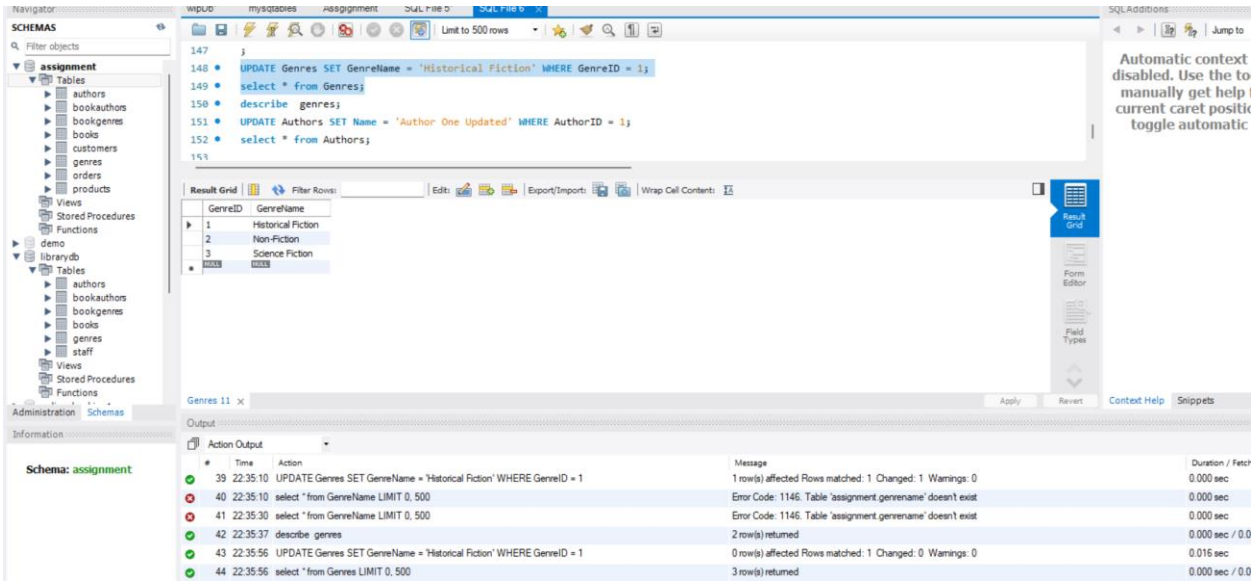


UPDATE Books SET CopiesAvailable = 4 WHERE BookID = 1;

select * from Books

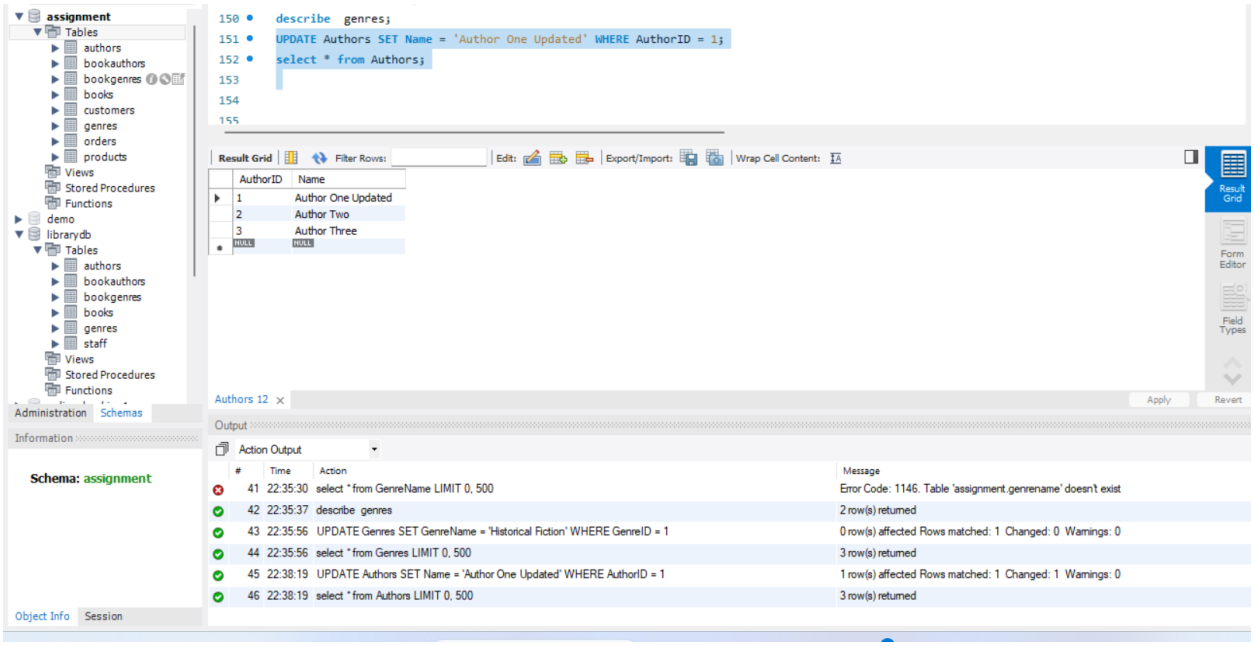
;





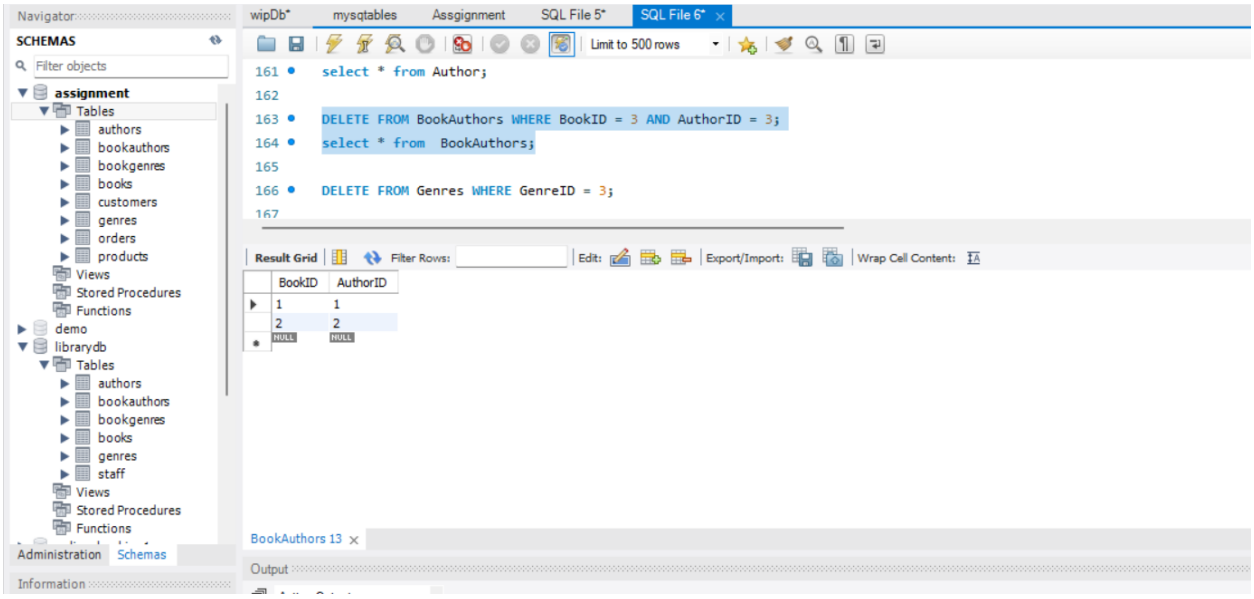
UPDATE Authors SET Name = 'Author One Updated' WHERE AuthorID = 1;

select * from Authors;



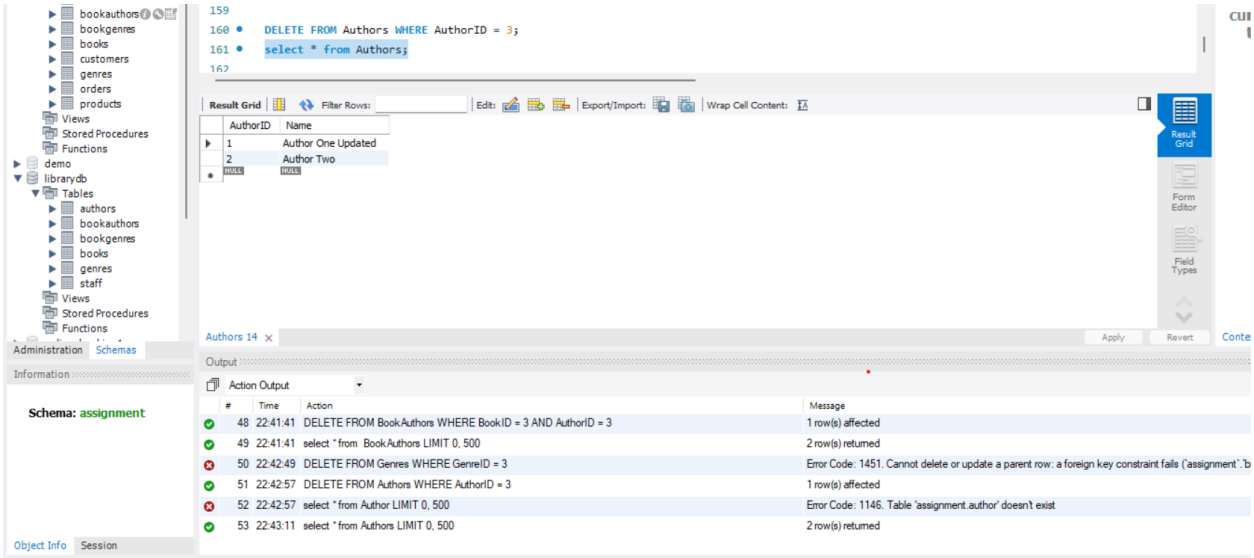
DELETE FROM BookAuthors WHERE BookID = 3 AND AuthorID = 3;

select * from BookAuthors;



DELETE FROM Authors WHERE AuthorID = 3;

select * from Authors;



LOAD DATA INFILE '/D:/books.csv'

INTO TABLE Books

FIELDS TERMINATED BY ','

ENCLOSED BY '"'

LINES TERMINATED BY '\n'

IGNORE 1 LINES

(BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable);