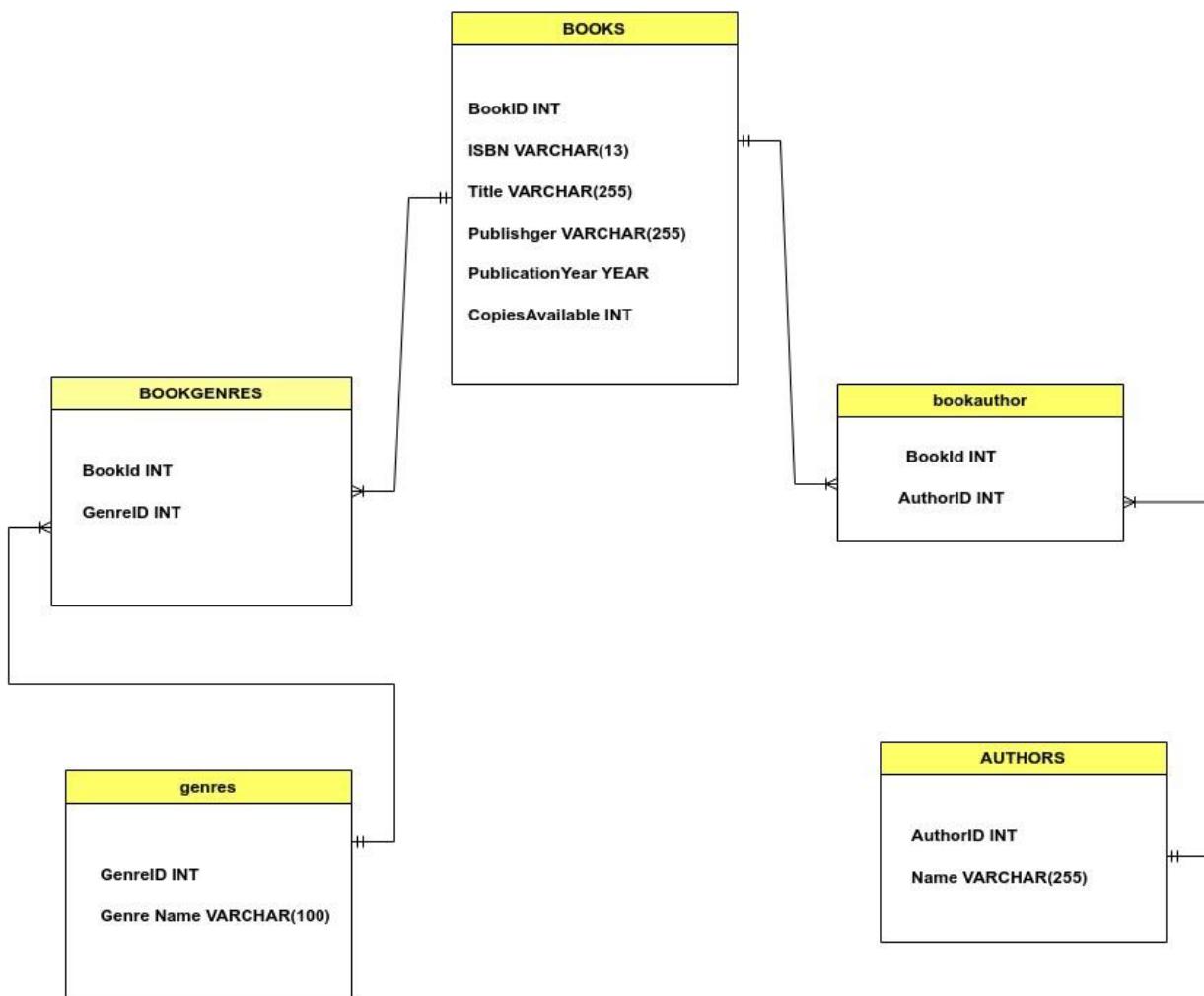


Name: Vaishnavi Vinod Ingole

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  X  +  ▾
Database changed
mysql> 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)
-> );
Query OK, 0 rows affected (0.11 sec)

mysql> CREATE TABLE Authors (
->     AuthorID INT PRIMARY KEY NOT NULL,
->     Name VARCHAR(255) NOT NULL
-> );
Query OK, 0 rows affected (0.06 sec)

mysql>
mysql> 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)
-> );
Query OK, 0 rows affected (0.09 sec)

mysql>
mysql> CREATE TABLE Genres (
->     GenreID INT PRIMARY KEY NOT NULL,
->     GenreName VARCHAR(100) NOT NULL UNIQUE
-> );
Query OK, 0 rows affected (0.08 sec)

mysql> 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)
-> );
```

```
MySQL 8.0 Command Line Cli × + ▾
Query OK, 0 rows affected (0.08 sec)

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
    -> );

```

```
Database changed
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 bookgenres;
ERROR 1146 (42S02): Table 'assignment.bookgenres' 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

Imagine 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;

The screenshot shows the SQL Server Management Studio interface. In the Object Explorer, under the 'assignment' schema, there are several tables: authors, bookauthors, bookgenres, books, customers, genres, orders, products, and views. The 'Books' table is defined with columns BookID (primary key), ISBN, Title, Publisher, PublicationYear, and CopiesAvailable. The 'Authors' table has AuthorID (primary key) and Name. The 'BookAuthors' table links Books and Authors. The 'OldBooks' table is listed as a 'Script' object. The 'Script' tab shows the T-SQL code for creating the database, selecting schema, and all three tables. The 'Output' tab displays the execution log with actions like SELECT statements, BEGIN TRANSACTION, and CREATE DATABASE, along with their times and results.

```
File Edit View Query Database Server Tools Scripting Help
wipDb* mysqltables Assignment* SQL File 5* SQL File 6*
Navigator: assignment Tables
SCHEMAS
assignment
Tables
Books
Authors
BookAuthors
BookGenres
Books
Customers
Genres
Orders
Products
Views
Stored Procedures
Functions
demo
online_banking1
practice
student
studentdb
sys
todo
wipdb
Administration Schemas
Information
Schema: assignment
Object Info Session
Action Output
# Time Action Message
8 19:01:28 SELECT customer_id, customer_name, email FROM customers WHERE city = 'New York' UNION SELECT cu... 3 row(s) returned
9 19:01:35 SELECT customer_id, customer_name, email FROM customers WHERE city = 'New York' UNION SELECT cu... 3 row(s) returned
10 19:05:40 BEGIN TRANSACTION Error Code: 1064. You have an error in your SQL syntax; check the ...
11 19:06:36 use Assignment 0 row(s) affected
12 22:57:00 CREATE DATABASE LibraryDB 1 row(s) affected
13 22:57:00 USE LibraryDB 0 row(s) affected
```

The screenshot shows the SQL Server Management Studio interface. The 'Tables' node under 'assignment' contains the 'Authors' table. The 'Script' tab shows the T-SQL command to add a 'BirthDate' column to the 'Authors' table. The 'Output' tab shows the execution log. It includes the creation of the database, switching to it, and attempting to alter the 'Authors' table. The third entry in the log shows an error: 'Error Code: 1146. Table 'wipdb.authors' doesn't exist'. The 'Messages' tab shows the error message and its duration.

```
File Edit View Query Database Server Tools Scripting Help
wipDb* mysqltables Assignment* SQL File 5* SQL File 6*
Tables
Books
Authors
BookAuthors
BookGenres
Books
Customers
Genres
Orders
Products
Views
Stored Procedures
Functions
demo
online_banking1
practice
student
studentdb
sys
todo
wipdb
Administration Schemas
Information
No object selected
Action Output
# Time Action Message Duration / Fetch
1 07:41:52 ALTER TABLE Authors ADD COLUMN BirthDate DATE Error Code: 1146. Table 'wipdb.authors' doesn't exist 0.000 sec
2 07:41:59 use assignment 0 row(s) affected 0.000 sec
3 07:42:06 ALTER TABLE Authors ADD COLUMN BirthDate DATE 0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0 0.063 sec
```

```

    172
173 • ALTER TABLE Authors
174   ADD COLUMN BirthDate DATE;
175
176 • DROP TABLE IF EXISTS OldBooks;
177

```

Output:

#	Time	Action	Message
1	07:41:52	ALTER TABLE Authors ADD COLUMN BirthDate DATE	Error Code: 1146. Table 'wipdb.authors' doesn't exist 0 row(s) affected
2	07:41:59	use assignment	0 row(s) affected
3	07:42:06	ALTER TABLE Authors ADD COLUMN BirthDate DATE	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
4	07:43:29	DROP TABLE IF EXISTS OldBooks	0 row(s) affected, 1 warning(s): 1051 Unknown table 'assignment.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

File Edit View Query Database Server Tools Scripting Help

Navigator: Local instance MySQL80 ×

SCHEMAS: Filter objects

- wipDb* mytables Assignment SQL File 5* SQL File 6*
- Tables Views Stored Procedures Functions
- assignment
- demo
- librarydb
- Tables Views Stored Procedures Functions
- online_banking1
- practice
- student
- studentdb
- sys
- todo
- wplib

SQL Additions: Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Context Help Snippets

Administration Schemas Information No object selected

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)	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, '123456789...', 5 row(s) affected Records: 5 Duplicates: 0 Warnings: 0)	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

assignment

```

91
92
93 •  DROP INDEX idx_title ON Books;
94
95
96 •  EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three';
97

```

Output

#	Time	Action	Message
45	22:38:19	UPDATE Authors SET Name = 'Author One Updated' WHERE AuthorID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
46	22:38:19	select * from Authors LIMIT 0, 500	3 row(s) returned
47	22:40:24	DELETE FROM Books WHERE BookID = 3	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails
48	22:41:41	DELETE FROM BookAuthors WHERE BookID = 3 AND AuthorID = 3	1 row(s) affected
49	22:41:41	select * from BookAuthors LIMIT 0, 500	2 row(s) returned
50	22:42:49	DELETE FROM Genres WHERE GenreID = 3	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails
51	22:42:57	DELETE FROM Authors WHERE AuthorID = 3	1 row(s) affected
52	22:42:57	select * from Author LIMIT 0, 500	Error Code: 1146. Table 'assignment.author' doesn't exist
53	22:43:11	select * from Authors LIMIT 0, 500	2 row(s) returned
54	22:43:59	DELETE FROM Genres WHERE GenreID = 3	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails
55	22:44:03	DELETE FROM Books WHERE BookID = 3	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails
56	22:51:38	use assignment	0 row(s) affected
57	22:53:37	LOAD DATA INFILE '/D:/books.csv' INTO TABLE Books FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n'	Error Code: 1290. The MySQL server is running with the -secure-file-priv option so it cannot execute this command
58	22:56:42	SHOW VARIABLES LIKE 'secure_file_priv'	1 row(s) returned
59	23:02:21	DROP INDEX idx_title ON Books	Error Code: 1091. Can't DROP 'idx_title'; check that column/key exists
60	23:02:35	CREATE INDEX idx_title ON Books (Title)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
61	23:02:41	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned
62	23:02:48	DROP INDEX idx_title ON Books	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
63	23:03:23	CREATE DATABASE LibraryDB	Error Code: 1007. Can't create database 'librarydb'; database exists
64	23:03:31	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned
65	23:04:13	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned

Object Info Session

assignment

```

92
93 •  DROP INDEX idx_title ON Books;
94
95
96 •  EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three';

```

Result Grid

id	select_type	table	partitions	type	possible_keys	key	key_len	ref	rows	filtered	Extra
1	SIMPLE	Books	NULL	ALL	NULL	NULL	NULL	NULL	3	33.33	Using where

Result 17 ×

Output

#	Time	Action	Message
57	22:53:37	LOAD DATA INFILE '/D:/books.csv' INTO TABLE Books FIELDS TERMINATED BY ',' ENCLOSED BY '\"' LINES TERMINATED BY '\n'	Error Code: 1290. The MySQL server is running with the -secure-file-priv option so it cannot execute this command
58	22:56:42	SHOW VARIABLES LIKE 'secure_file_priv'	1 row(s) returned
59	23:02:21	DROP INDEX idx_title ON Books	Error Code: 1091. Can't DROP 'idx_title'; check that column/key exists
60	23:02:35	CREATE INDEX idx_title ON Books (Title)	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
61	23:02:41	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned
62	23:02:48	DROP INDEX idx_title ON Books	0 row(s) affected Records: 0 Duplicates: 0 Warnings: 0
63	23:03:23	CREATE DATABASE LibraryDB	Error Code: 1007. Can't create database 'librarydb'; database exists
64	23:03:31	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned
65	23:04:13	EXPLAIN SELECT * FROM Books WHERE Title = 'Book Three'	1 row(s) returned

Object Info Session

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 wipDb* mysqltables Assignment SQL File 5* SQL File 6* Limit to 1000 rows

SCHEMAS

Filter objects

- assignment
- demo
- librarydb
 - Tables
 - Views
 - Stored Procedures
 - Functions
- online_banking1
- practice
- student
- studentdb
- sys
- todo
- wipdb

```

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

```

Administration Schemas

Information

No object selected

Output

#	Time	Action	Message
1	08:18:50	CREATE USER 'library_user'@'localhost' IDENTIFIED BY 'pass'	0 row(s) affected

Administration Schemas

Information

No object selected

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

Schemas

```

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: | Result Grid | Form Editor | Field Types | Apply | Revert | Continue

AuthorID	Name
NULL	NULL

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 name
4	08:21:06	USE LibraryDB	0 row(s) affected
5	08:21:11	select * from authors LIMIT 0, 1000	0 row(s) returned

Object Info Session

Schemas

```

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

```

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 name
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

Object Info Session

Schemas

```

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

```

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 double-clicking its name
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

Object Info Session

Schemas

```

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

```

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 double-clicking its name
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

Object Info Session

32°C Sunny

The screenshot shows a MySQL Workbench interface. On the left, there's a tree view of databases: 'todo' and 'wipdb'. Below it, under 'Information', is a table named 'staff' with columns: StaffID (int PK), Name (varchar(255)), Position (varchar(11)), and ContactDetails (varchar(255)). On the right, a session history window displays the following SQL commands:

```

104 • REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';
105
106 • DROP USER 'library_user'@'localhost';
107

```

Below the session history is an 'Output' pane titled 'Action Output' with a table of log entries:

#	Time	Action	Message	Duration
①	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 name in the SC...	0.000 s
②	4 08:21:06	USE LibraryDB	0 row(s) affected	0.000 s
③	5 08:21:11	select * from authors LIMIT 0, 1000	0 row(s) returned	0.015 s
④	6 08:21:44	REVOKE UPDATE, DELETE ON LibraryDB.* FROM 'library_user'@'localhost';	0 row(s) affected	0.000 s
⑤	7 08:22:12	FLUSH PRIVILEGES	0 row(s) affected	0.016 s
⑥	8 08:22:32	DROP USER 'library_user'@'localhost'	0 row(s) affected	0.016 s

The system tray at the bottom shows icons for battery, signal, and network.

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;

Schemas

```

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

Action Output

#	Time	Action	Message
13	14:51:54	call emproc(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, '1234567890123', 'Book One', 'Publisher A', 2020, 5),	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, '1234567890124', 'Book Two', 'Publisher B', 2019, 3),	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, '1234567890125', 'Book Three', 'Publisher C', 2021, 2);	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0

Object Info Session

Schemas

```

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

Books 3 x

Result Grid

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
*	HULL	HULL	HULL	HULL	HULL

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, '1234567890123', 'Book One', 'Publisher A', 2020, 5),	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, '1234567890124', 'Book Two', 'Publisher B', 2019, 3),	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 exist
20	21:24:58	select * from Books LIMIT 0, 500	3 row(s) returned

Object Info Session

Filter objects

```

117 •   select * from Books;
118 •   INSERT INTO Authors (AuthorID, Name) VALUES
119
120     (1, 'Author One'),
121     (2, 'Author Two'),
122     (3, 'Author Three');
123 •   select * from Authors;

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

AuthorID	Name
1	Author One
2	Author Two
3	Author Three
NULL	NULL

Authors 4 x

Action Output

#	Time	Action	Message
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 exist
20	21:24:58	select * from Books LIMIT 0, 500	3 row(s) returned
21	21:53:53	INSERT INTO Authors (AuthorID, Name) VALUES (1, 'Author One'), (2, 'Author Two'), (3, 'Author Three')	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
22	21:54:18	INSERT INTO Authors (AuthorID, Name) VALUES (1, 'Author One'), (2, 'Author Two'), (3, 'Author Three')	Error Code: 1062. Duplicate entry '1' for key 'authors.PRIMARY'
23	21:54:24	select * from Authors LIMIT 0, 500	3 row(s) returned

Object Info Session

INSERT INTO BookAuthors (BookID, AuthorID) VALUES

(1, 1),

(2, 2),

(3, 3);

select * from BookAuthors

Filter objects

```

126 •   INSERT INTO BookAuthors (BookID, AuthorID) VALUES
127     (1, 1),
128     (2, 2),
129     (3, 3);
130 •   select * from BookAuthors
131
132 •   INSERT INTO Genres (GenreID, GenreName) VALUES

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: |

BookID	AuthorID
1	1
2	2
3	3
NULL	NULL

BookAuthors 5 x

Action Output

#	Time	Action	Message
20	21:24:58	select * from Books LIMIT 0, 500	3 row(s) returned
21	21:53:53	INSERT INTO Authors (AuthorID, Name) VALUES (1, 'Author One'), (2, 'Author Two'), (3, 'Author Three')	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
22	21:54:18	INSERT INTO Authors (AuthorID, Name) VALUES (1, 'Author One'), (2, 'Author Two'), (3, 'Author Three')	Error Code: 1062. Duplicate entry '1' for key 'authors.PRIMARY'
23	21:54:24	select * from Authors LIMIT 0, 500	3 row(s) returned
24	22:18:42	INSERT INTO BookAuthors (BookID, AuthorID) VALUES (1, 1), (2, 2), (3, 3)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
25	22:18:42	select * from BookAuthors LIMIT 0, 500	3 row(s) returned

Object Info Session

```
INSERT INTO Genres (GenreID, GenreName) VALUES
```

```
(1, 'Fiction'),
```

```
(2, 'Non-Fiction'),
```

```
(3, 'Science Fiction');
```

```
select * from Genres;
```

The screenshot shows the MySQL Workbench interface. On the left, the 'SCHEMAS' tree view is open, showing the 'assignment' schema with its tables: BookAuthors, Genres, and BookGenres. The 'Tables' node also lists authors, bookauthors, books, and staff. In the center, the SQL editor window contains the following code:

```
130 • select * from BookAuthors;
131
132 • INSERT INTO Genres (GenreID, GenreName) VALUES
133     (1, 'Fiction'),
134     (2, 'Non-Fiction'),
135     (3, 'Science Fiction');
136 • select * from Genres;
137
138 • TRUNCATE TABLE BookGenres (BookID, GenreID) VALUES
```

The 'Result Grid' tab shows the results of the 'select * from Genres;' query:

GenreID	GenreName
1	Fiction
2	Non-Fiction
3	Science Fiction
*	NULL

Below the grid, the 'Action Output' pane displays the history of actions taken:

#	Time	Action	Message
27	22:23:55	select * from Genres LIMIT 0, 500	Error Code: 1146. Table 'assignment.genres' doesn't exist 8 row(s) returned
28	22:24:19	show tables	Error Code: 1146. Table 'assignment.genres' doesn't exist
29	22:24:52	describe genres	Error Code: 1062. Duplicate entry '1' for key 'genres.PRIMARY'
30	22:25:12	INSERT INTO Genres (GenreID, GenreName) VALUES (1, 'Fiction'), (2, 'Non-Fiction'), (3, 'Science Fiction')	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server version for the right syntax to use near 'Genres (GenreID, GenreName) VALUES (1, 'Fiction'), (2, 'Non-Fiction'), (3, 'Science Fiction')' at line 1
31	22:25:20	(3, 'Science Fiction')	3 row(s) returned
32	22:25:25	select * from Genres LIMIT 0, 500	3 row(s) returned

```
INSERT INTO BookGenres (BookID, GenreID) VALUES
```

```
(1, 1),
```

```
(2, 2),
```

```
(3, 3);
```

```
select * from BookGenres;
```

Automatic disabled. Use manually current connection to toggle at

```

137
138 • INSERT INTO BookGenres (BookID, GenreID) VALUES
139 (1, 1),
140 (2, 2),
141 (3, 3);
142
143 • select * from BookGenres;
144

```

BookID	GenreID
1	1
2	2
3	3
ALL	ALL

Action Output

#	Time	Action	Message
29	22:24:52	describe genres	Error Code: 1146. Table 'assignment.genres' doesn't exist
30	22:25:12	INSERT INTO Genres (GenreID, GerveName) VALUES (1, 'Fiction'), (2, 'Non-Fiction'), (3, 'Science Fiction')	Error Code: 1062. Duplicate entry '1' for key 'genres.PRIMARY'
31	22:25:20	(3, 'Science Fiction')	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server...
32	22:25:25	select * from Genres LIMIT 0, 500	3 row(s) returned
33	22:26:27	INSERT INTO BookGenres (BookID, GenreID) VALUES (1, 1), (2, 2), (3, 3)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
34	22:26:27	select * from BookGenres LIMIT 0, 500	3 row(s) returned

UPDATE Books SET CopiesAvailable = 4 WHERE BookID = 1;

select * from Books

Automatic disabled. Use manually current connection to toggle at

```

141 (3, 3);
142
143 • select * from BookGenres;
144
145 • UPDATE Books SET CopiesAvailable = 4 WHERE BookID = 1;
146 • select * from Books
147 ;

```

BookID	ISBN	Title	Publisher	PublicationYear	CopiesAvailable
1	1234567890123	Book One	Publisher A	2020	4
2	1234567890124	Book Two	Publisher B	2019	3
3	1234567890125	Book Three	Publisher C	2021	2
ALL	ALL	ALL	ALL	ALL	ALL

Action Output

#	Time	Action	Message
33	22:26:27	INSERT INTO BookGenres (BookID, GenreID) VALUES (1, 1), (2, 2), (3, 3)	3 row(s) affected Records: 3 Duplicates: 0 Warnings: 0
34	22:26:27	select * from BookGenres LIMIT 0, 500	3 row(s) returned
35	22:27:45	INSERT INTO Members (MemberID, Name, Address, PhoneNumber, Email, MembershipStartDate, MembershipEndDat...	Error Code: 1064. You have an error in your SQL syntax; check the manual that corresponds to your MySQL server...
36	22:27:54	INSERT INTO Members (MemberID, Name, Address, PhoneNumber, Email, MembershipStartDate, MembershipEndDat...	Error Code: 1146. Table 'assignment.members' doesn't exist
37	22:34:29	UPDATE Books SET CopiesAvailable = 4 WHERE BookID = 1	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0
38	22:34:29	select * from Books LIMIT 0, 500	3 row(s) returned

Navigator wip01 mysqld8 Assignment SQL File 0 | Limit to 500 rows | Jump to

Schemas

Filter objects

assignment

- Tables
 - authors
 - bookauthors
 - bookgenres
 - books
 - customers
 - genres
 - orders
 - products
- Views
- Stored Procedures
- Functions

demo

librarydb

- Tables
 - authors
 - bookauthors
 - bookgenres
 - books
 - genres
 - staff
- Views
- Stored Procedures
- Functions

Administration Schemas

Information

Schema: assignment

```

147 ;
148 • UPDATE Genres SET GenreName = 'Historical Fiction' WHERE GenreID = 1;
149 • select * from Genres;
150 • describe genres;
151 • UPDATE Authors SET Name = 'Author One Updated' WHERE AuthorID = 1;
152 • select * from Authors;
153

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types

GenreID	GenreName
1	Historical Fiction
2	Non-Fiction
3	Science Fiction
4	HULL

Genres 11 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
39	22:35:10	UPDATE Genres SET GenreName = 'Historical Fiction' WHERE GenreID = 1;	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	0.000 sec
40	22:35:10	select * from GenreName LIMIT 0, 500	Error Code: 1146 Table 'assignment.genrename' doesn't exist	0.000 sec
41	22:35:10	select * from GenreName LIMIT 0, 500	Error Code: 1146 Table 'assignment.genrename' doesn't exist	0.000 sec
42	22:35:37	describe genres	2 row(s) returned	0.000 sec / 0.0
43	22:35:56	UPDATE Genres SET GenreName = 'Historical Fiction' WHERE GenreID = 1;	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	0.016 sec
44	22:35:56	select * from Genres LIMIT 0, 500	3 row(s) returned	0.000 sec / 0.0

Apply Revert Context Help Snippets

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

select * from Authors;

Navigator wip01 mysqld8 Assignment SQL File 0 | Limit to 500 rows | Jump to

Schemas

Filter objects

assignment

- Tables
 - authors
 - bookauthors
 - bookgenres
 - books
 - customers
 - genres
 - orders
 - products
- Views
- Stored Procedures
- Functions

demo

librarydb

- Tables
 - authors
 - bookauthors
 - bookgenres
 - books
 - genres
 - staff
- Views
- Stored Procedures
- Functions

Administration Schemas

Information

Schema: assignment

```

150 • describe genres;
151 • UPDATE Authors SET Name = 'Author One Updated' WHERE AuthorID = 1;
152 • select * from Authors;
153
154
155

```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content: | Result Grid | Form Editor | Field Types

AuthorID	Name
1	Author One Updated
2	Author Two
3	Author Three
4	HULL

Authors 12 x

Output

Action Output

#	Time	Action	Message	Duration / Fetch
41	22:35:30	select * from GenreName LIMIT 0, 500	Error Code: 1146 Table 'assignment.genrename' doesn't exist	2 row(s) returned
42	22:35:37	describe genres	0 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	3 row(s) returned
43	22:35:56	UPDATE Genres SET GenreName = 'Historical Fiction' WHERE GenreID = 1;	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	3 row(s) returned
44	22:35:56	select * from Genres LIMIT 0, 500	1 row(s) affected Rows matched: 1 Changed: 0 Warnings: 0	3 row(s) returned
45	22:38:19	UPDATE Authors SET Name = 'Author One Updated' WHERE AuthorID = 1;	1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0	3 row(s) returned
46	22:38:19	select * from Authors LIMIT 0, 500	3 row(s) returned	3 row(s) returned

Apply Revert

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

select * from BookAuthors;

Navigator: mysqldb* mysqltables Assignment SQL File 5* SQL File 6*

SCHEMAS

- assignment
 - Tables: authors, bookauthors, bookgenres, books, customers, genres, orders, products
 - Views
 - Stored Procedures
 - Functions
- demo
- librarydb
 - Tables: authors, bookauthors, bookgenres, books, genres, staff
 - Views
 - Stored Procedures
 - Functions

Administration Schemas

Information

```

161 • select * from Author;
162
163 • DELETE FROM BookAuthors WHERE BookID = 3 AND AuthorID = 3;
164 • select * from BookAuthors;
165
166 • DELETE FROM Genres WHERE GenreID = 3;
167

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content:

BookID	AuthorID
1	1
2	2
HULL	HULL

BookAuthors 13 x

Output:

DELETE FROM Authors WHERE AuthorID = 3;

select * from Authors;

Navigator: mysqldb* mysqltables Assignment SQL File 5* SQL File 6*

SCHEMAS

- assignment
 - Tables: bookauthors, bookgenres, books, customers, genres, orders, products
 - Views
 - Stored Procedures
 - Functions
- demo
- librarydb
 - Tables: authors, bookauthors, bookgenres, books, genres, staff
 - Views
 - Stored Procedures
 - Functions

Administration Schemas

Information

Schema: assignment

Object Info Session

```

159
160 • DELETE FROM Authors WHERE AuthorID = 3;
161 • select * from Authors;
162

```

Result Grid | Filter Rows: Edit: Export/Import: Wrap Cell Content:

AuthorID	Name
1	Author One Updated
2	Author Two
HULL	HULL

Authors 14 x

Output:

Action Output

#	Time	Action	Message
48	22:41:41	DELETE FROM BookAuthors WHERE BookID = 3 AND AuthorID = 3	1 row(s) affected
49	22:41:41	select * from BookAuthors LIMIT 0, 500	2 row(s) returned
50	22:42:49	DELETE FROM Genres WHERE GenreID = 3	Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails ('assignment'.`books`)
51	22:42:57	DELETE FROM Authors WHERE AuthorID = 3	1 row(s) affected
52	22:42:57	select * from Author LIMIT 0, 500	Error Code: 1146. Table 'assignment.author' doesn't exist
53	22:43:11	select * from Authors LIMIT 0, 500	2 row(s) returned

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);

The screenshot shows the MySQL Workbench interface. The Navigator pane on the left lists databases: 'assignment' (selected), 'demo', and 'librarydb'. Under 'assignment', it shows tables: authors, bookauthors, bookgenres, books, customers, genres, orders, products. Under 'librarydb', it shows tables: authors, bookauthors, bookgenres, books, genres, staff. The SQL Editor pane on the right contains the following SQL code:

```
1 • use assignment;
2
3 • LOAD DATA INFILE '/D:/books.csv'
4   INTO TABLE Books
5   FIELDS TERMINATED BY ','
6   ENCLOSED BY '"'
7   LINES TERMINATED BY '\n'
8   IGNORE 1 LINES
9   (BookID, ISBN, Title, Publisher, PublicationYear, CopiesAvailable);
```