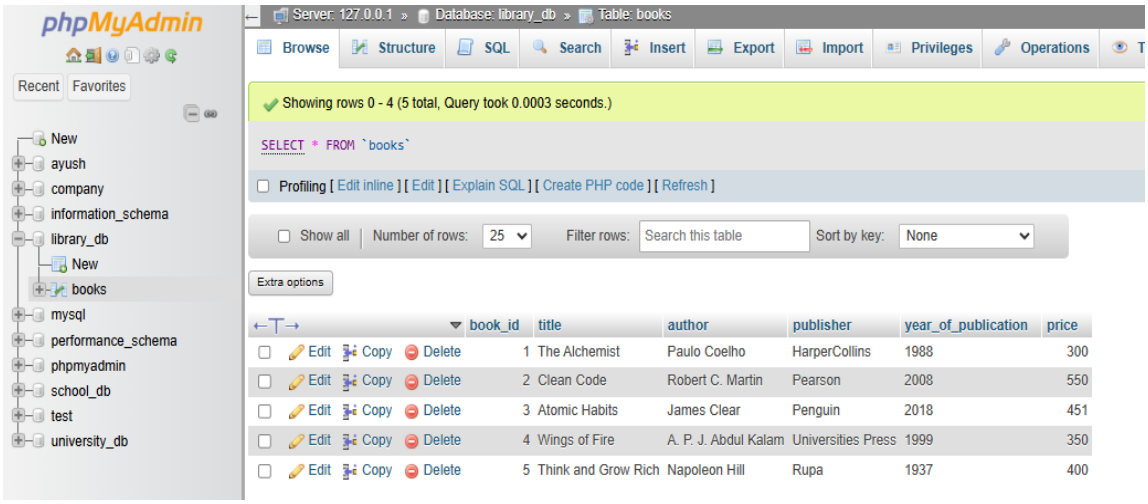
EXTRA LAB PRACTISE FOR DATABASE CONCEPTS

1. Introduction to SQL

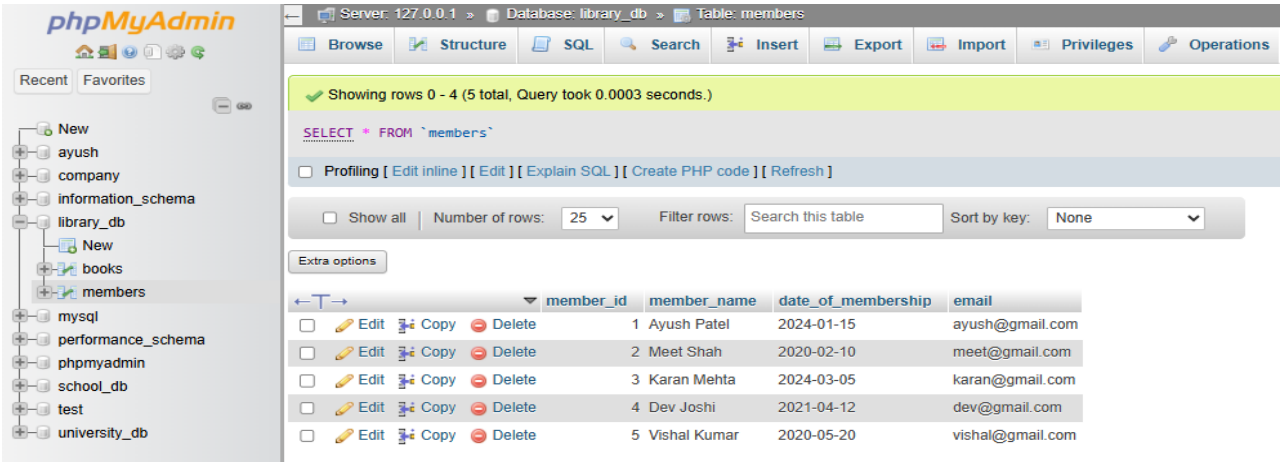
Lab 3 : Create a database called library_db and a table books with columns: book_id, title, author, publisher, year_of_publication, and price. Insert five records into the table.

- CREATE DATABASE library_db
- CREATE TABLE books (book_id int PRIMARY KEY AUTO_INCREMENT, title varchar(20),author varchar(20),publisher varchar(20),year_of_publication varchar(20), price int)
- INSERT INTO books (title,author,publisher,year_of_publication,price)VALUES('The Alchemist', 'Paulo Coelho', 'HarperCollins', 1988, 299.50),('Clean Code', 'Robert C. Martin', 'Pearson', 2008, 550.00),('Atomic Habits', 'James Clear', 'Penguin', 2018, 450.75),('Wings of Fire', 'A. P. J. Abdul Kalam', 'Universities Press', 1999, 350.00),('Think and Grow Rich', 'Napoleon Hill', 'Rupa', 1937, 399.99);



Lab 4 : Create a table members in library_db with columns: member_id, member_name, date_of_membership, and email. Insert five records into this table.

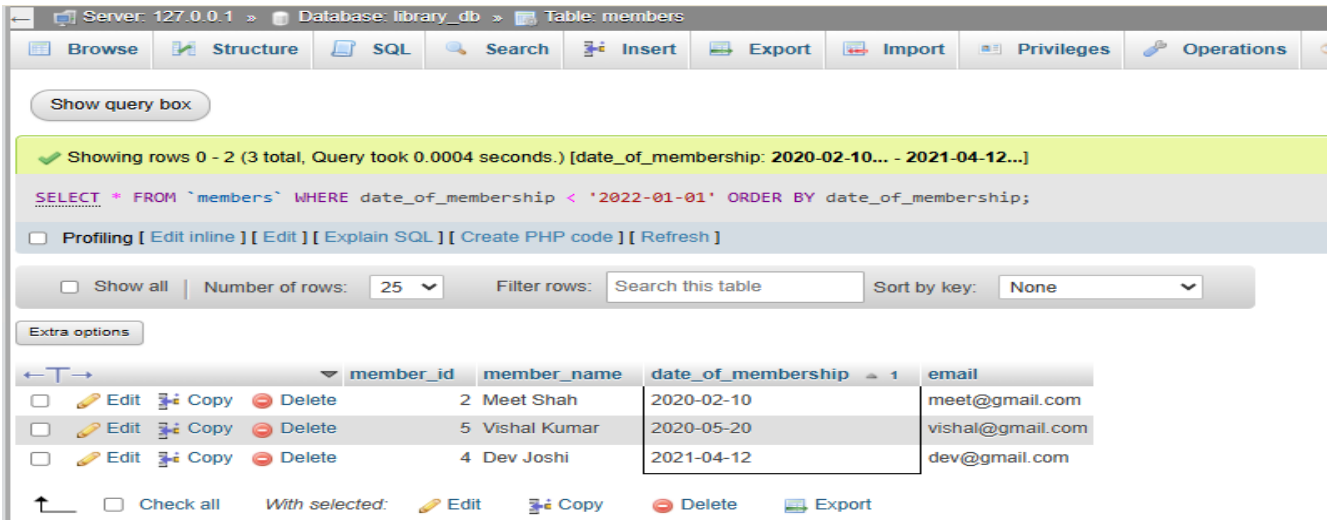
- CREATE TABLE members (member_id INT PRIMARY KEY,member_name VARCHAR(20),date_of_membership DATE,email VARCHAR(20));
- INSERT INTO members (member_name,date_of_membership,email)VALUES
  ('Ayush Patel', '2024-01-15', 'ayush@gmail.com'),
  ('Meet Shah', '2020-02-10', 'meet@gmail.com'),
  ('Karan Mehta', '2024-03-05', 'karan@gmail.com'),
  ('Dev Joshi', '2021-04-12', 'dev@gmail.com'),
  ('Vishal Kumar', '2020-05-20', 'vishal@gmail.com');



2. SQL Syntax

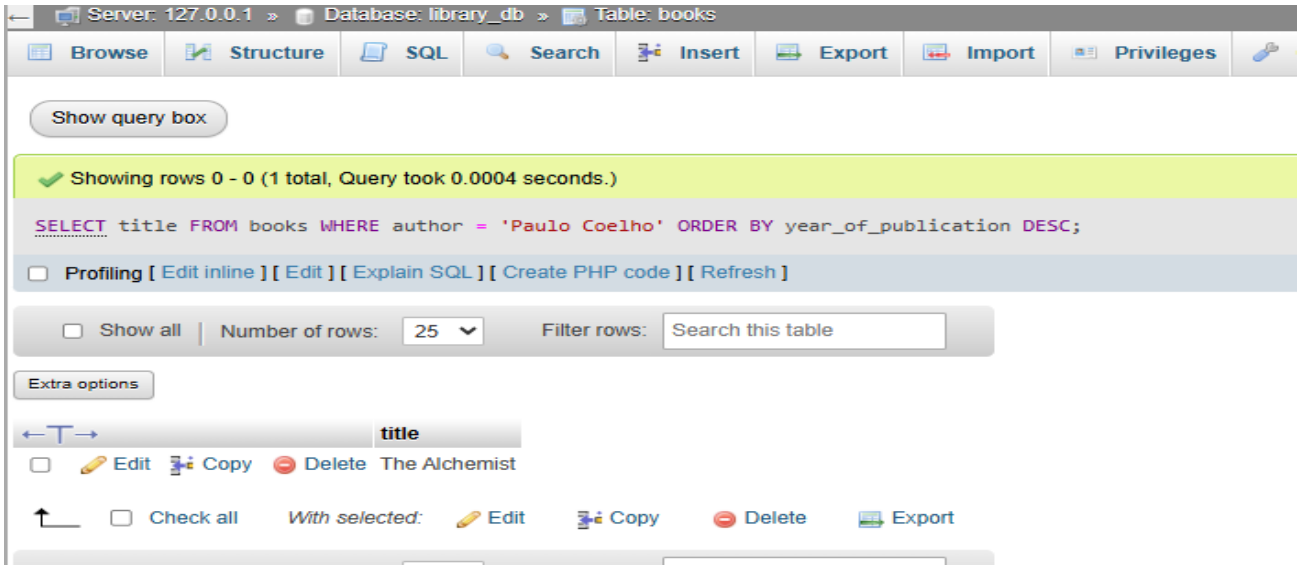
Lab 3 : Retrieve all members who joined the library before 2022. Use appropriate SQL syntax with WHERE and ORDER BY.

- SELECT * FROM `members` WHERE date_of_membership < '2022-01-01' ORDER BY date_of_membership



Lab 4 : Write SQL queries to display the titles of books published by a specific author. Sort the results by year_of_publication in descending order.

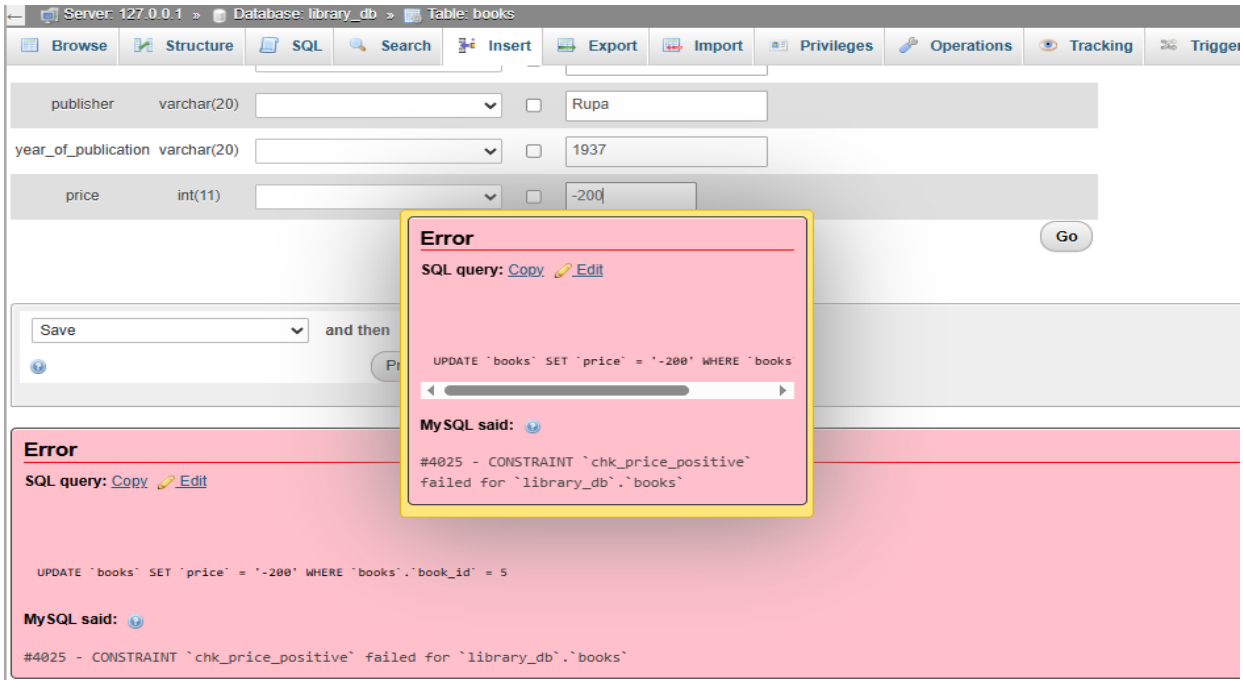
- SELECT title FROM books WHERE author = 'Paulo Coelho' ORDER BY year_of_publication DESC



3. SQL Constraints

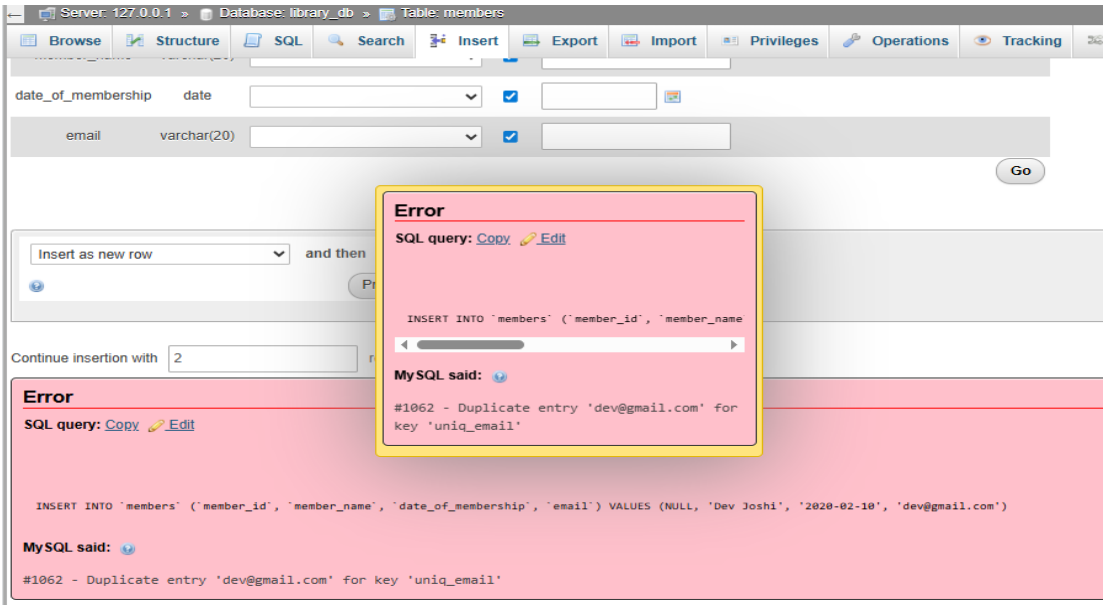
Lab 3 : Add a CHECK constraint to ensure that the price of books in the books table is greater than 0.

- ALTER TABLE books ADD CONSTRAINT check_price_positive CHECK (price > 0)



Lab 4 : : Modify the members table to add a UNIQUE constraint on the email column, ensuring that each member has a unique email address.

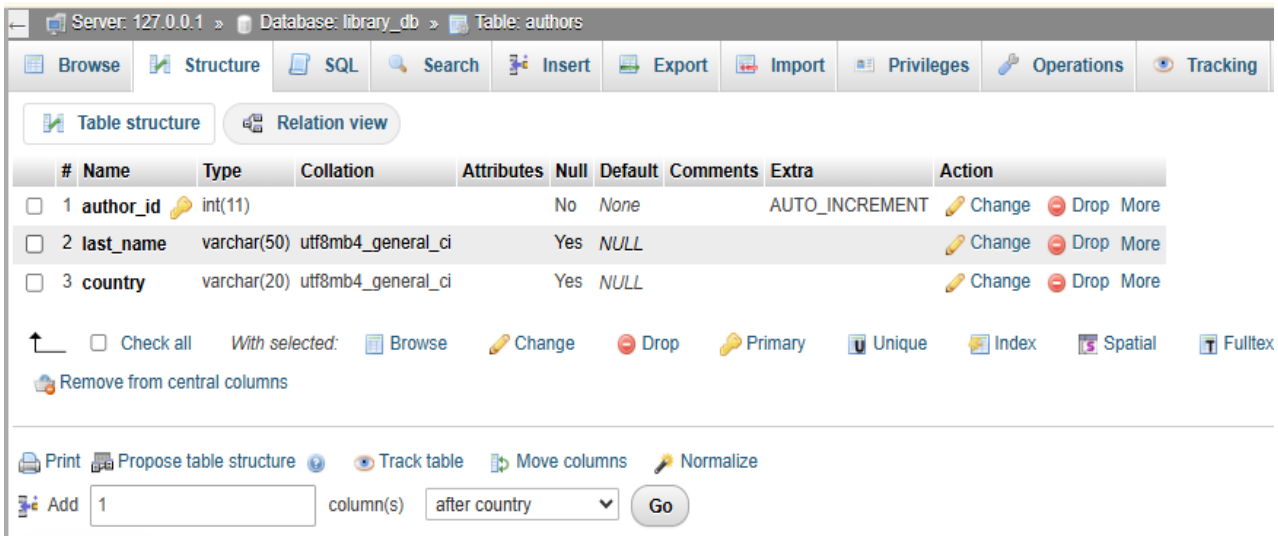
- ALTER TABLE members ADD CONSTRAINT uniq_email UNIQUE(email)



4. Main SQL Commands and Sub-commands (DDL)

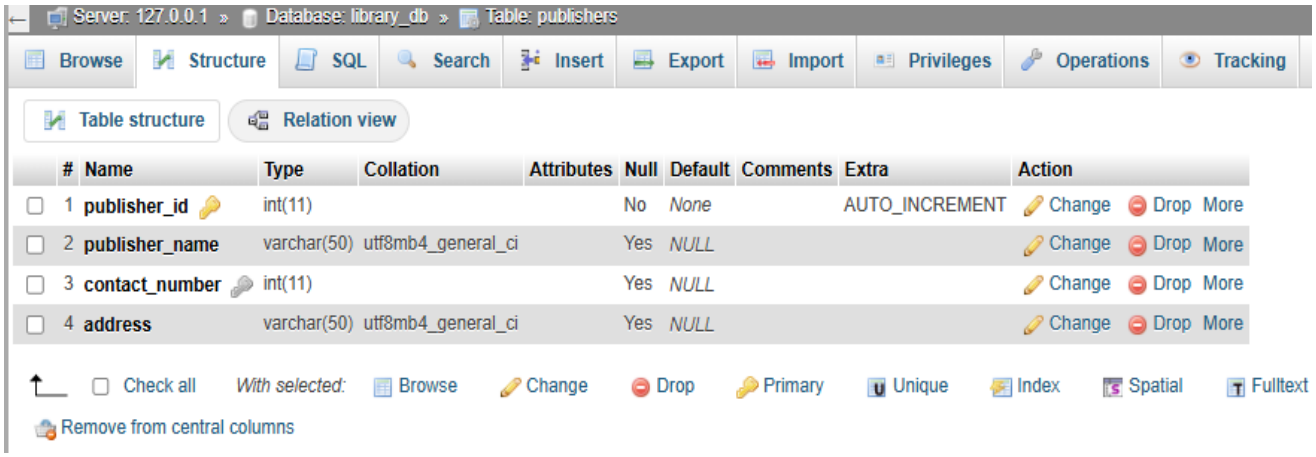
Lab 3 : Create a table authors with the following columns: author_id, first_name, last_name, and country. Set author_id as the primary key.

- CREATE TABLE authors (author_id int PRIMARY KEY AUTO_INCREMENT, last_name varchar(50),country varchar(20))



Lab 4 : Create a table publishers with columns: publisher_id, publisher_name, contact_number, and address. Set publisher_id as the primary key and contact_number as unique.

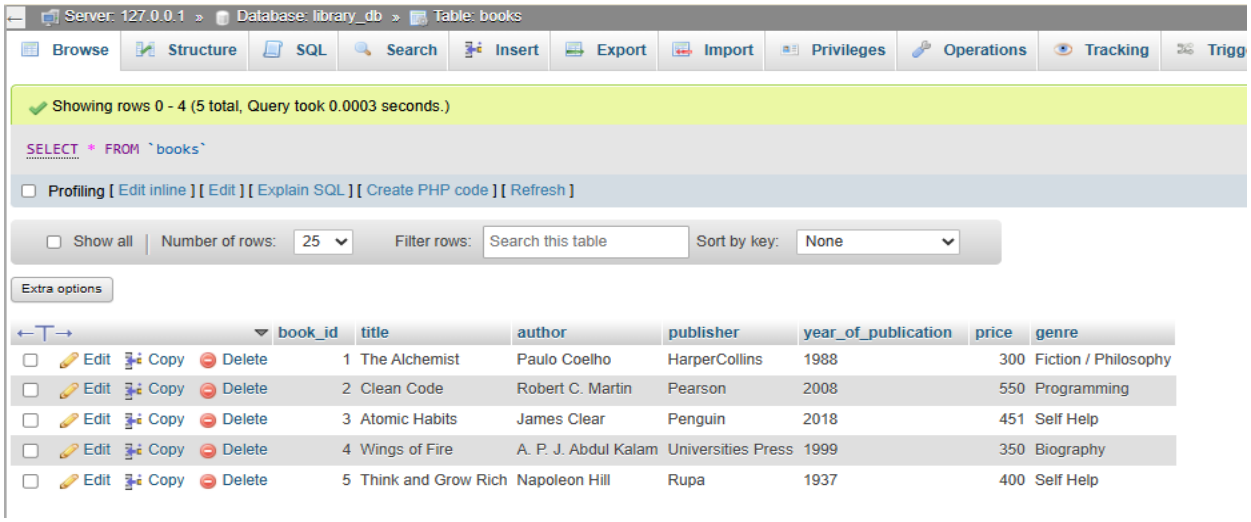
- CREATE TABLE publishers (publisher_id int PRIMARY KEY AUTO_INCREMENT,publisher_name varchar(50),contact_number int UNIQUE KEY, address varchar(50))



5. ALTER Command

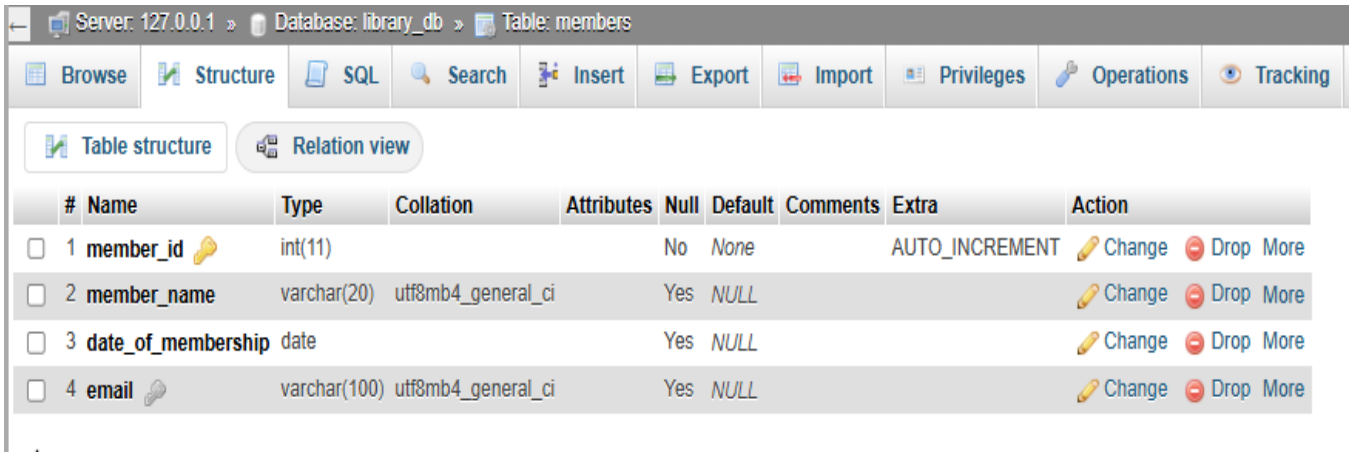
Lab 3 : Add a new column genre to the books table. Update the genre for all existing records.

- ALTER TABLE books ADD COLUMN gener varchar(50)
- UPDATE books SET genre = 'Fiction / Philosophy' WHERE book_id = 1;
- UPDATE books SET genre = 'Programming' WHERE book_id = 2;
- UPDATE books SET genre = 'Self Help' WHERE book_id = 3;
- UPDATE books SET genre = 'Biography' WHERE book_id = 4;
- UPDATE books SET genre = 'Self Help' WHERE book_id = 5;



Lab 4 : Modify the members table to increase the length of the email column to 100 characters.

- ALTER TABLE members MODIFY email varchar(100)



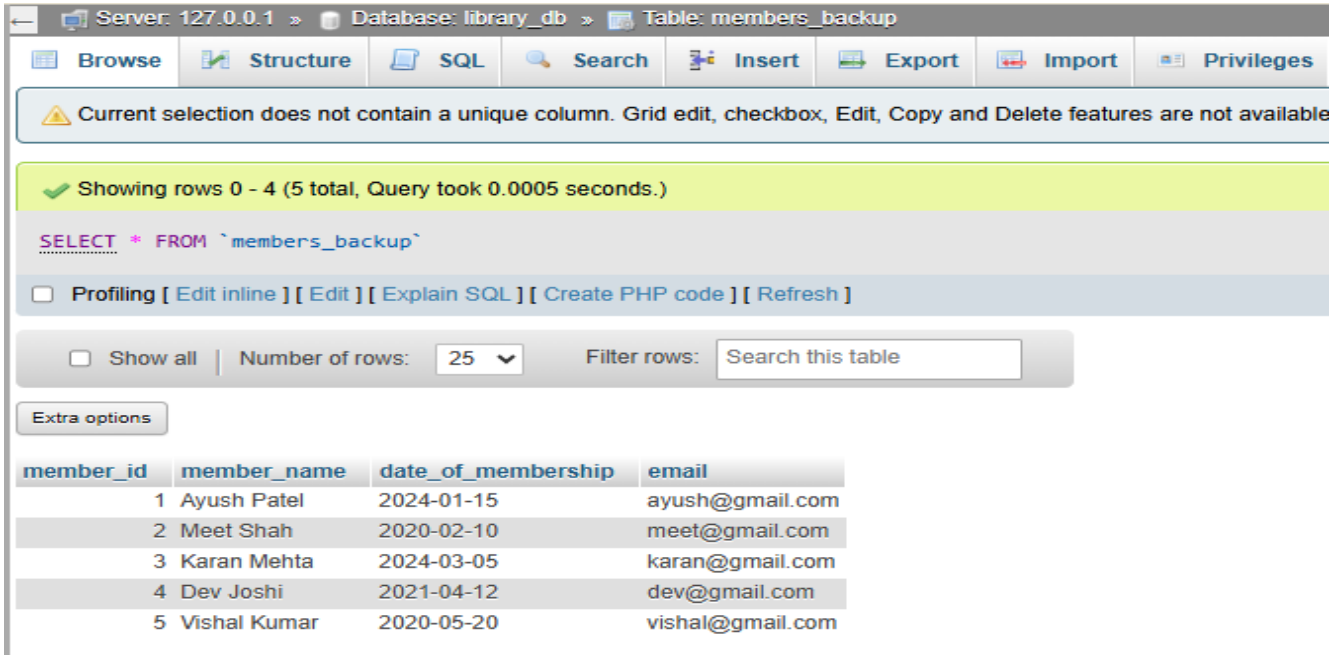
6. DROP Command

Lab 3 : Drop the publishers table from the database after verifying its structure.

- DROP TABLE publishers

Lab 4 : Create a backup of the members table and then drop the original members table.

- CREATE TABLE members_backup AS SELECT * FROM members
- DROP TABLE members



7. Data Manipulation Language (DML)

Lab 4 : Insert three new authors into the authors table, then update the last name of one of the authors.

- INSERT INTO authors (first_name,last_name,country) VALUES ('Paulo', 'Coelho', 'Brazil');
  INSERT INTO authors (first_name,last_name,country) VALUES ('Robert', 'Martin', 'USA');
  INSERT INTO authors (first_name,last_name,country) VALUES ('James', 'Clear', 'USA');
- UPDATE authors SET last_name = 'C.martin' WHERE author_id = 2

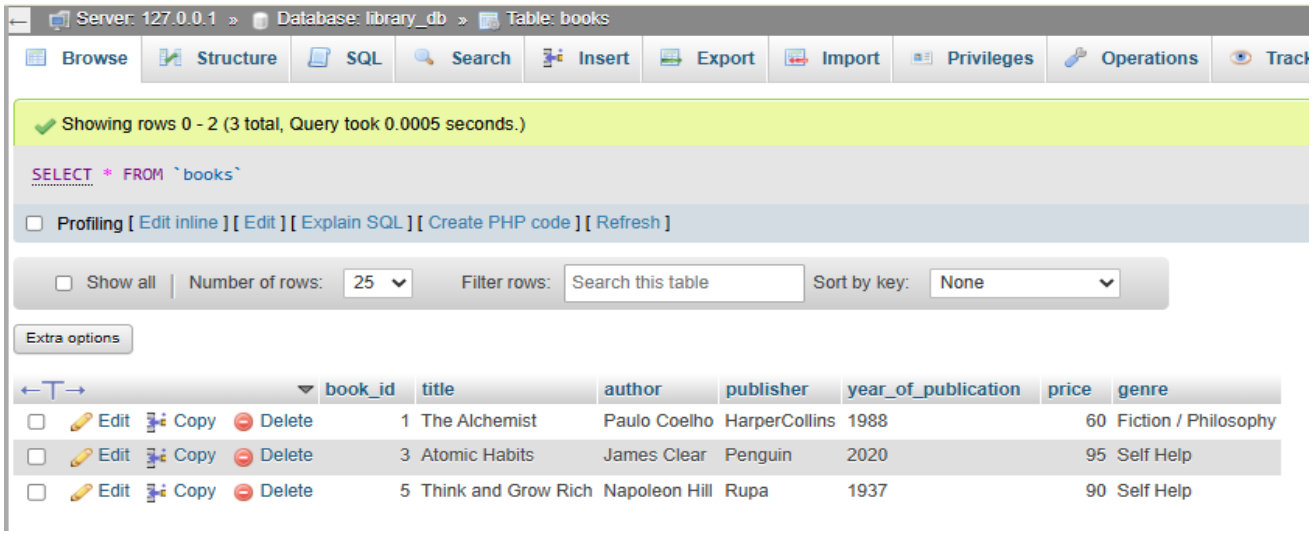
Lab 5 : Delete a book from the books table where the price is higher than $100.

- DELETE FROM books WHERE price>100

8. UPDATE Command

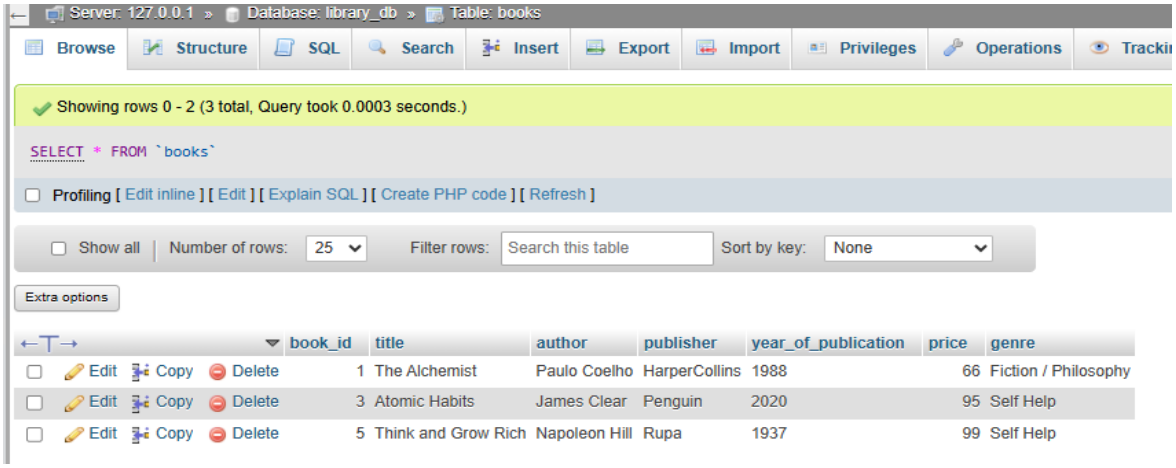
Lab 3 : Update the year_of_publication of a book with a specific book_id.

- UPDATE books SET year_of_publication = 2020 WHERE book_id = 3



Lab 4 : Increase the price of all books published before 2015 by 10%.

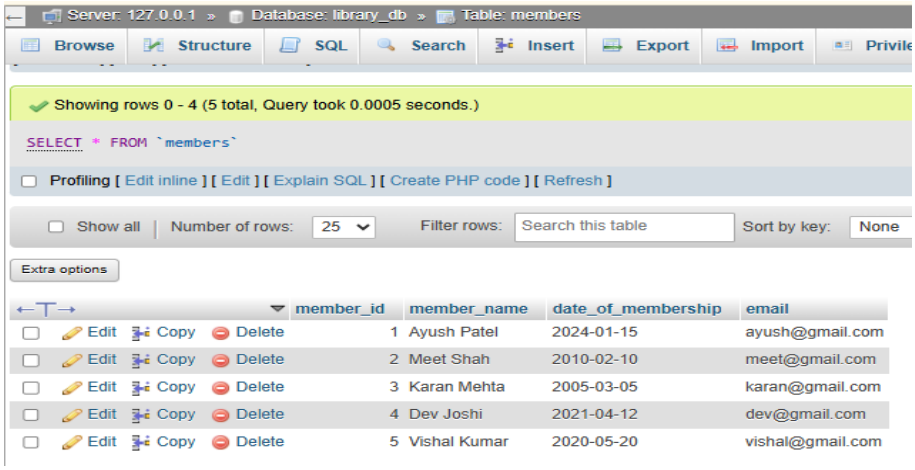
- UPDATE books SET price = price * 1.10 WHERE year_of_publication < 2015

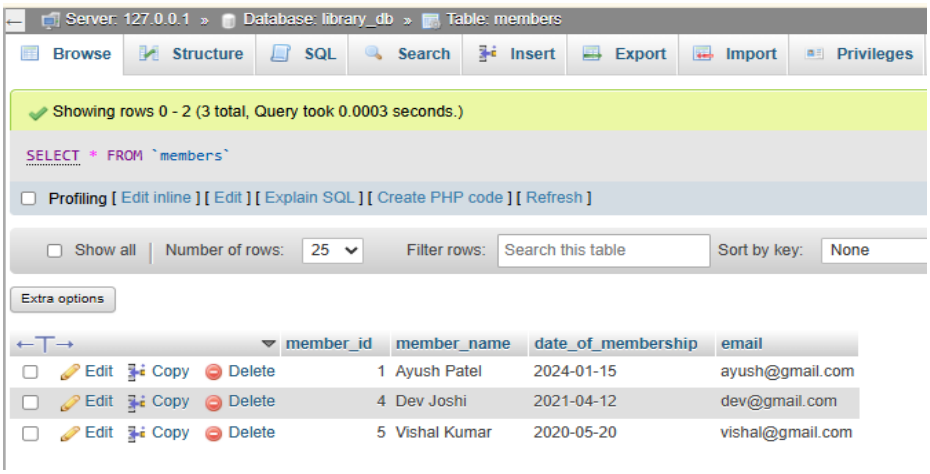


9. DELETE Command

Lab 3 : Remove all members who joined before 2020 from the members table.

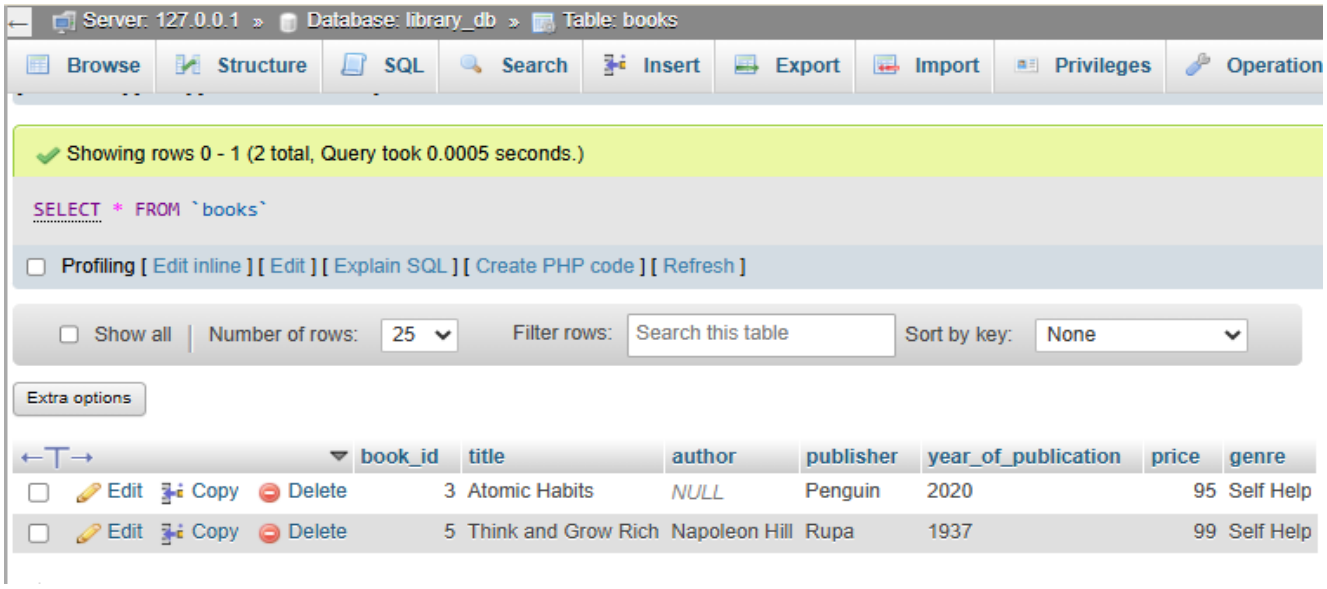
- DELETE FROM members WHERE date_of_membership < '2020-01-01'

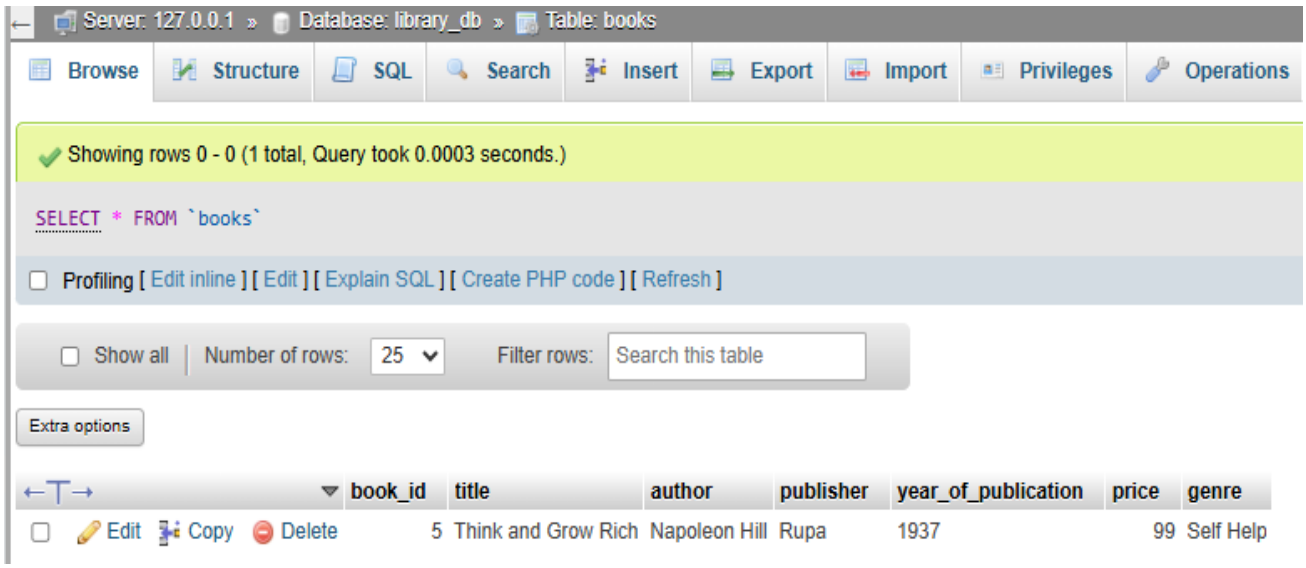




Lab 4 : Delete all books that have a NULL value in the author column.

- DELETE FROM books WHERE author IS null

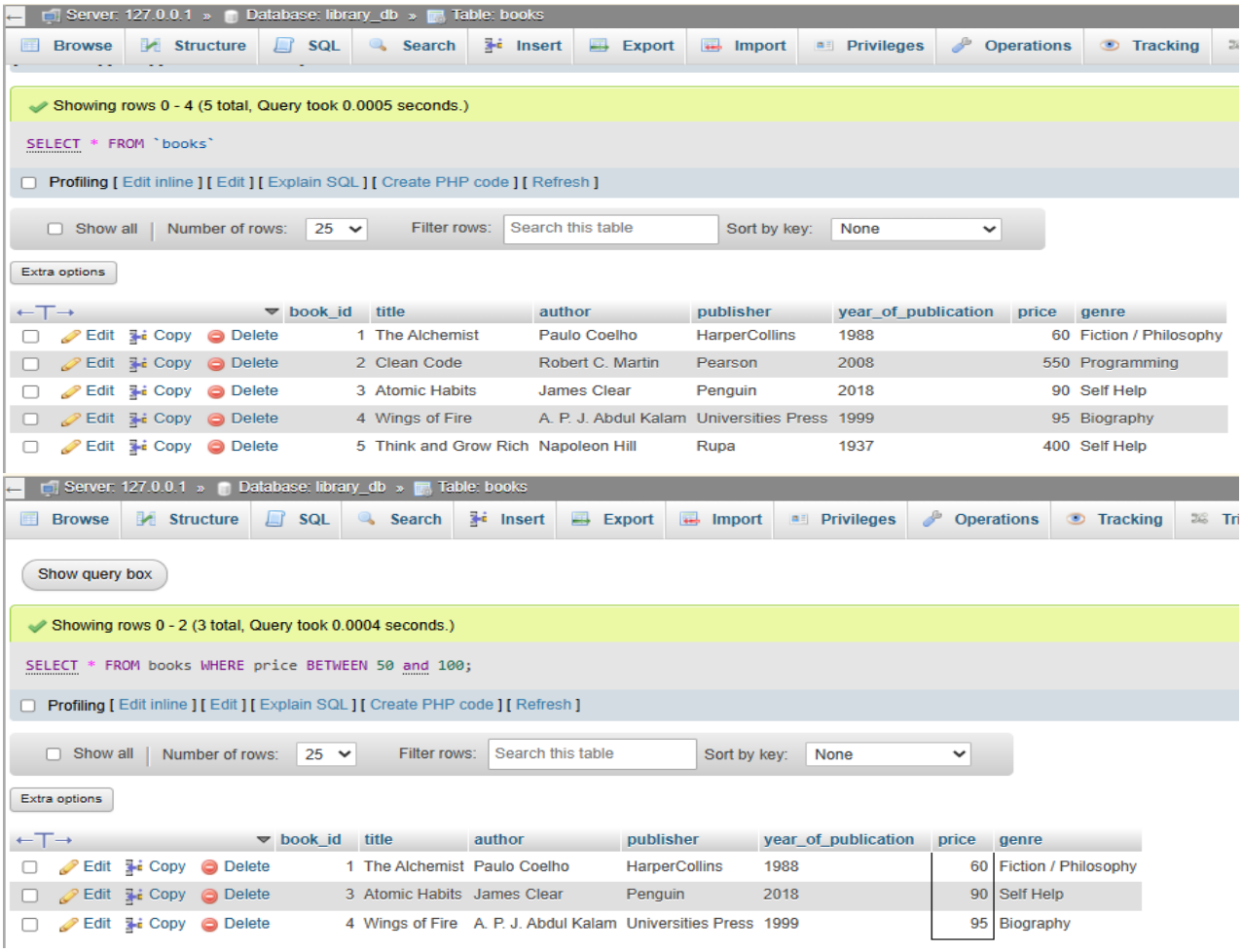




10. Data Query Language (DQL)

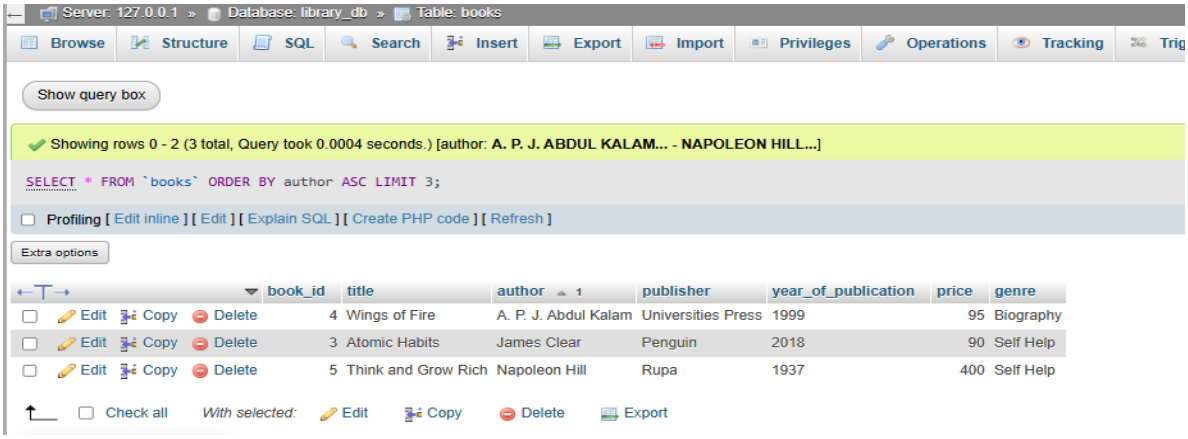
Lab 4 : Write a query to retrieve all books with price between $50 and $100.

- SELECT * FROM books WHERE price BETWEEN 50 and 100



Lab 5 : Retrieve the list of books sorted by author in ascending order and limit the results to the top 3 entries.

- SELECT * FROM `books` ORDER BY author ASC LIMIT 3;

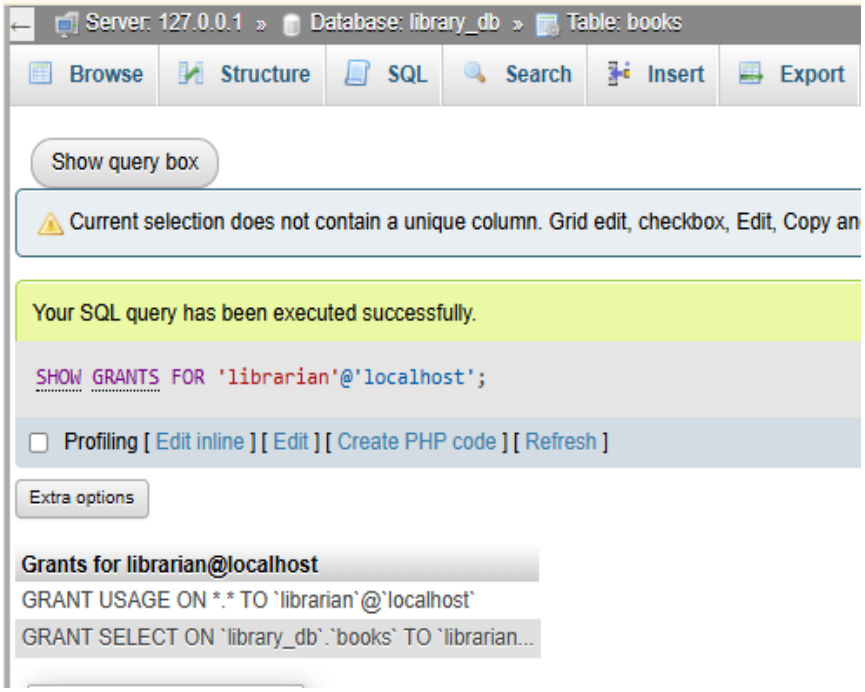


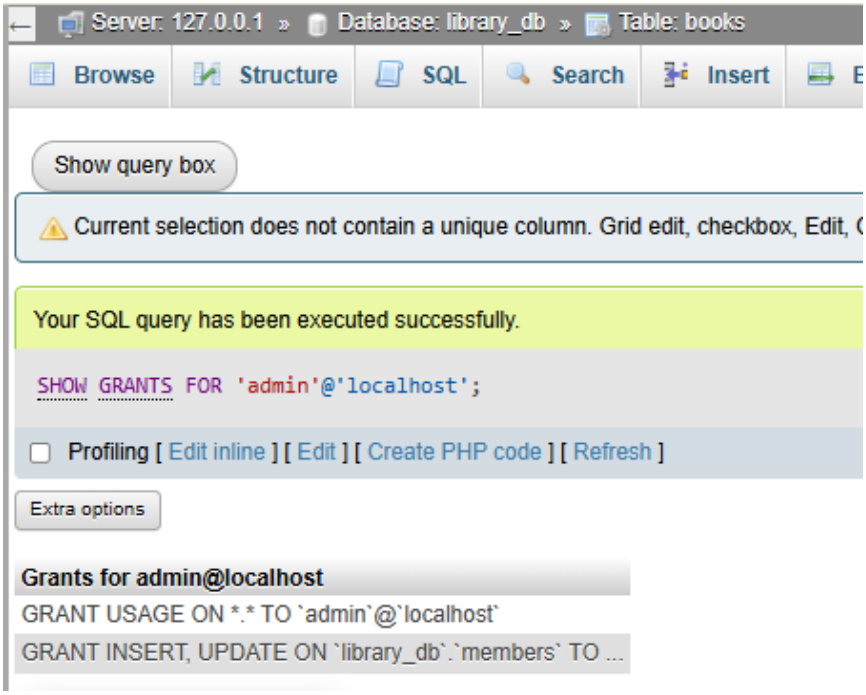
11. Data Control Language (DCL)

Lab 3 : Grant SELECT permission to a user named librarian on the books table.

Lab 4 : Grant INSERT and UPDATE permissions to the user admin on the members table.

- GRANT SELECT ON library_db.books TO 'librarian'@'localhost';
- GRANT INSERT, UPDATE ON library_db.members TO 'admin'@'localhost';



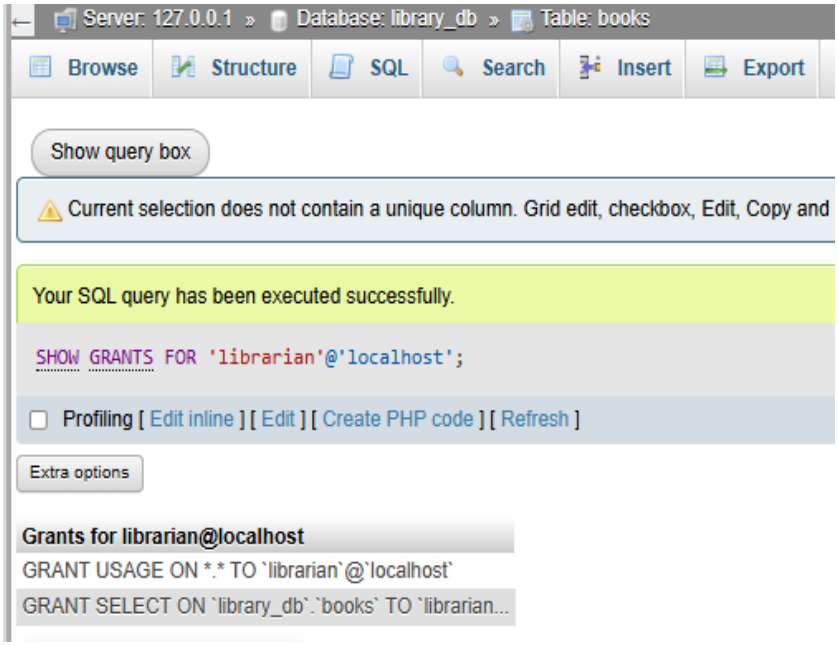


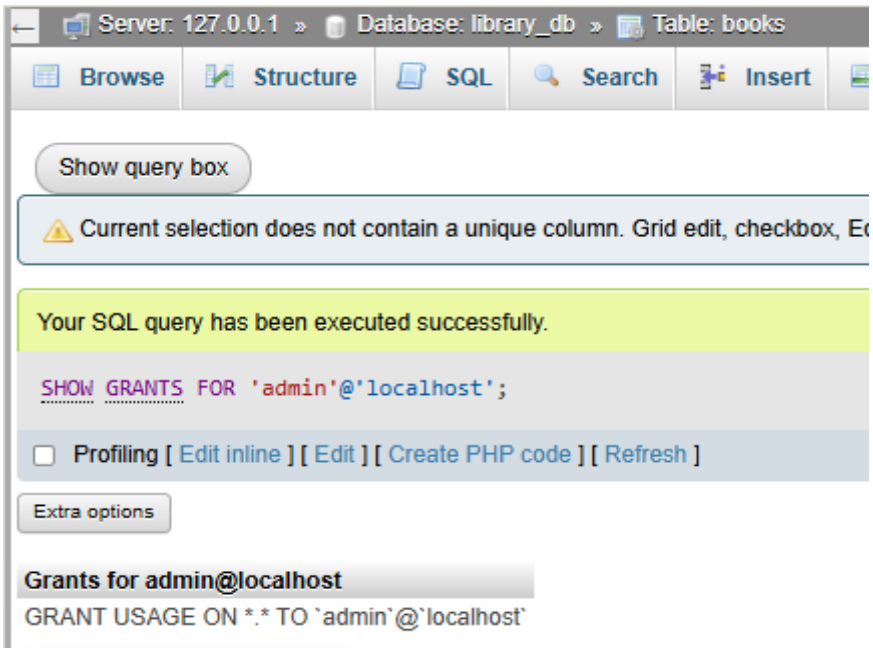
12. REVOKE Command

Lab 3 : Revoke the INSERT privilege from the user librarian on the books table.

Lab 4 : Revoke all permissions from user admin on the members table.

- REVOKE INSERT ON library_db.books FROM 'librarian'@'localhost';
- REVOKE ALL PRIVILEGES ON library_db.members FROM 'admin'@'localhost';

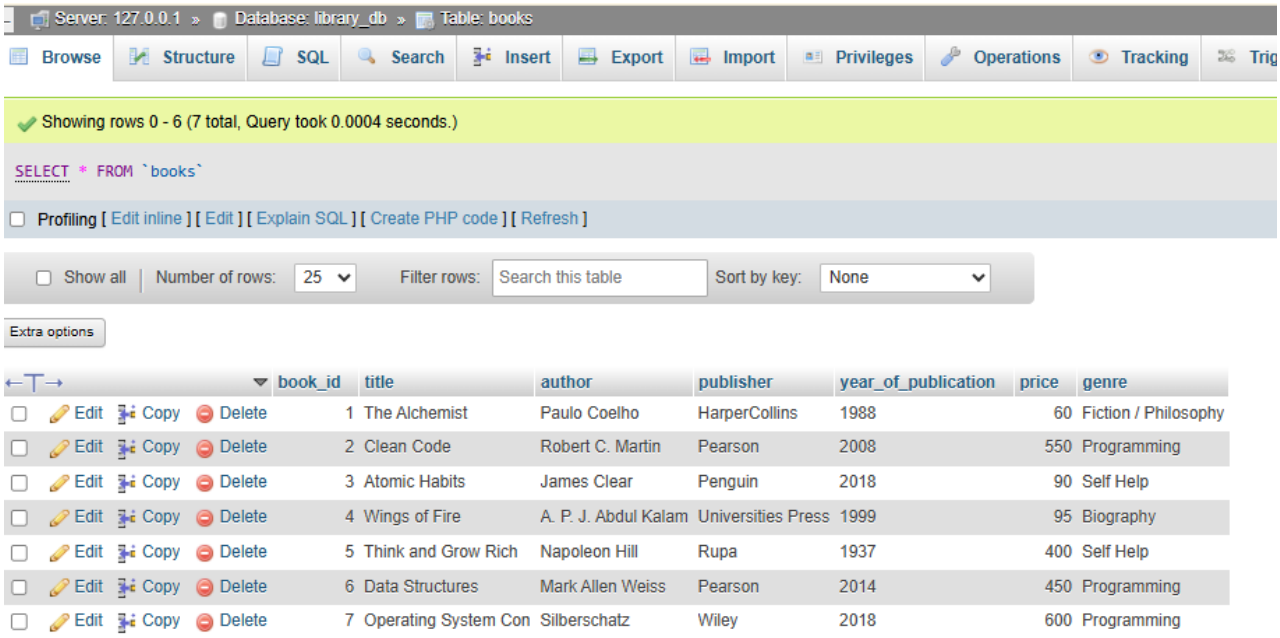




13. Transaction Control Language (TCL)

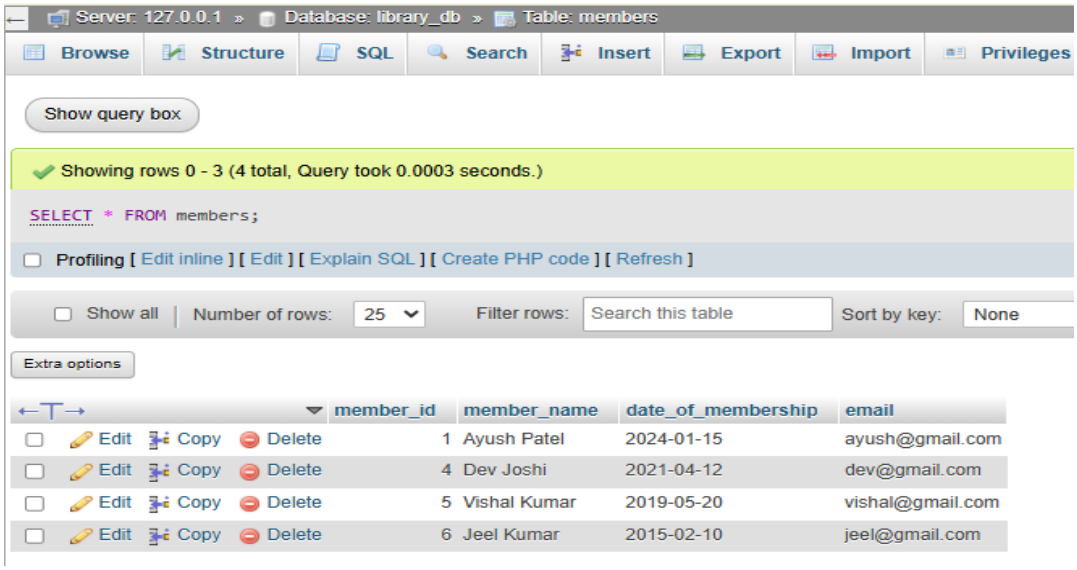
Lab 3 : Use COMMIT after inserting multiple records into the books table, then make another insertion and perform a ROLLBACK.

- START TRANSACTION;
  INSERT INTO books
  (book_id, title, author, publisher, year_of_publication, price, genre)
  VALUES
  (6, 'Data Structures', 'Mark Allen Weiss', 'Pearson', 2014, 450, 'Programming');
  INSERT INTO books
  (book_id, title, author, publisher, year_of_publication, price, genre)
  VALUES
  (7, 'Operating System Concepts', 'Silberschatz', 'Wiley', 2018, 600, 'Programming');
  COMMIT;
  START TRANSACTION;
  INSERT INTO books
  (book_id, title, author, publisher, year_of_publication, price, genre)
  VALUES
  (8, 'Machine Learning', 'Tom Mitchell', 'McGraw Hill', 2017, 700, 'AI');
  ROLLBACK;



Lab 4 : Set a SAVEPOINT before making updates to the members table, perform some updates, and then roll back to the SAVEPOINT.

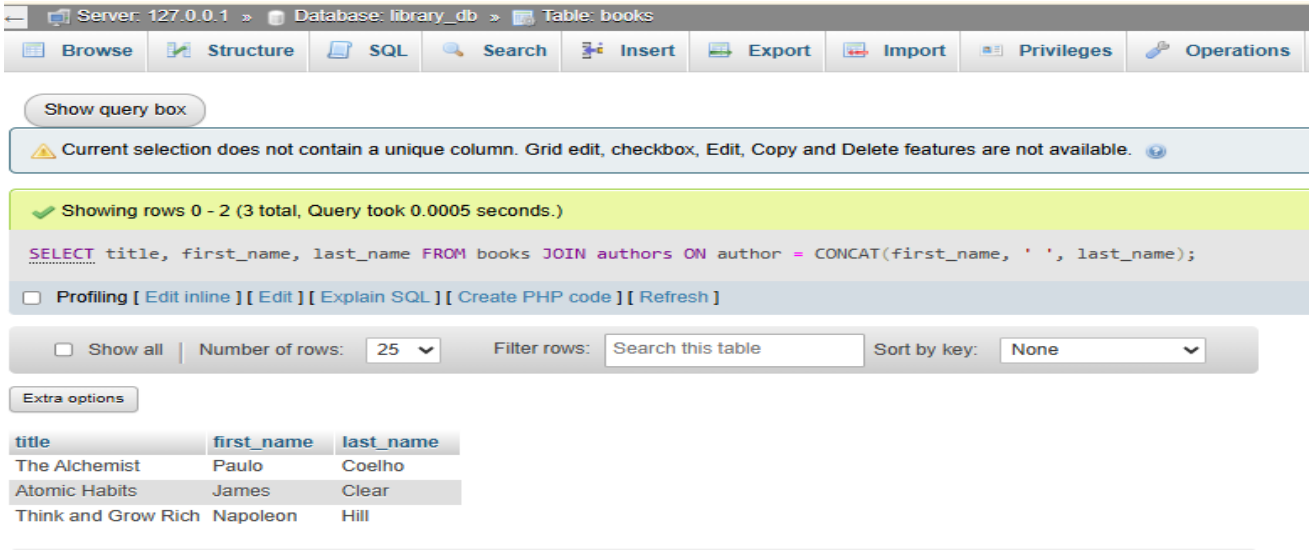
- START TRANSACTION;
  SAVEPOINT before_update;
  UPDATE members
  SET email = 'updated_ayush@gmail.com'
  WHERE member_id = 1;
  UPDATE members
  SET member_name = 'Dev Joshi Patel'
  WHERE member_id = 4;
  ROLLBACK TO before_update;
  COMMIT;



14. SQL Joins

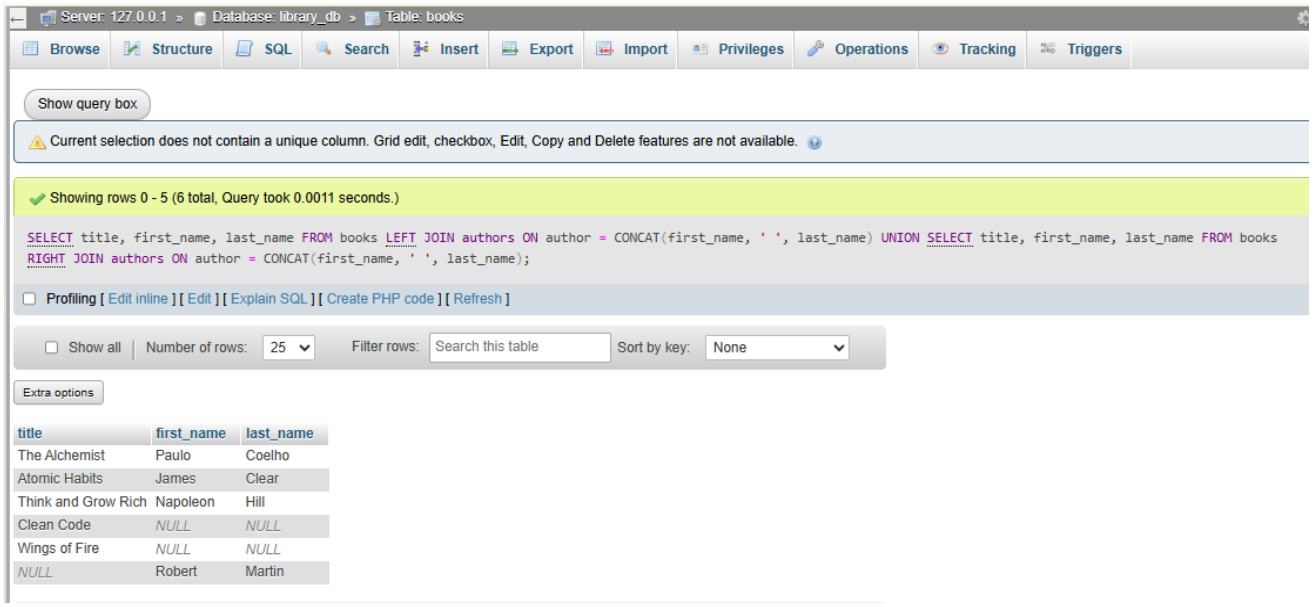
Lab 3 : Perform an INNER JOIN between books and authors tables to display the title of books and their respective authors' names.

- SELECT title, first_name, last_name FROM books JOIN authors ON author = CONCAT(first_name, ' ', last_name);



Lab 4 : Use a FULL OUTER JOIN to retrieve all records from the books and authors tables, including those with no matching entries in the other table.

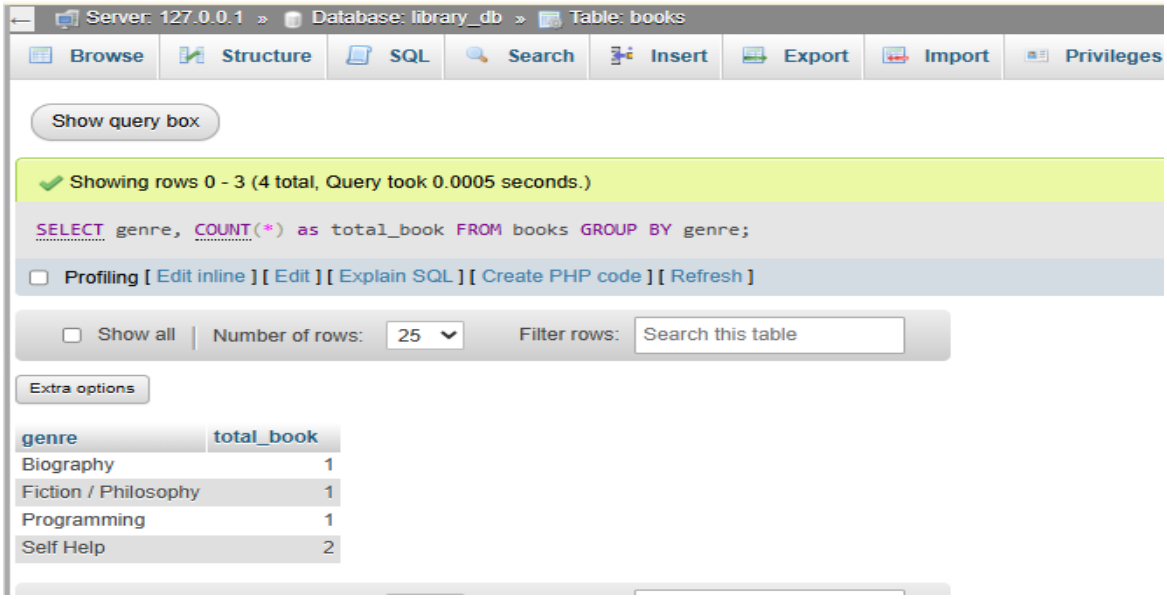
- SELECT title, first_name, last_name FROM books LEFT JOIN authors ON author = CONCAT(first_name, ' ', last_name) UNION SELECT title, first_name, last_name FROM books RIGHT JOIN authors ON author = CONCAT(first_name, ' ', last_name);



15. SQL Group By

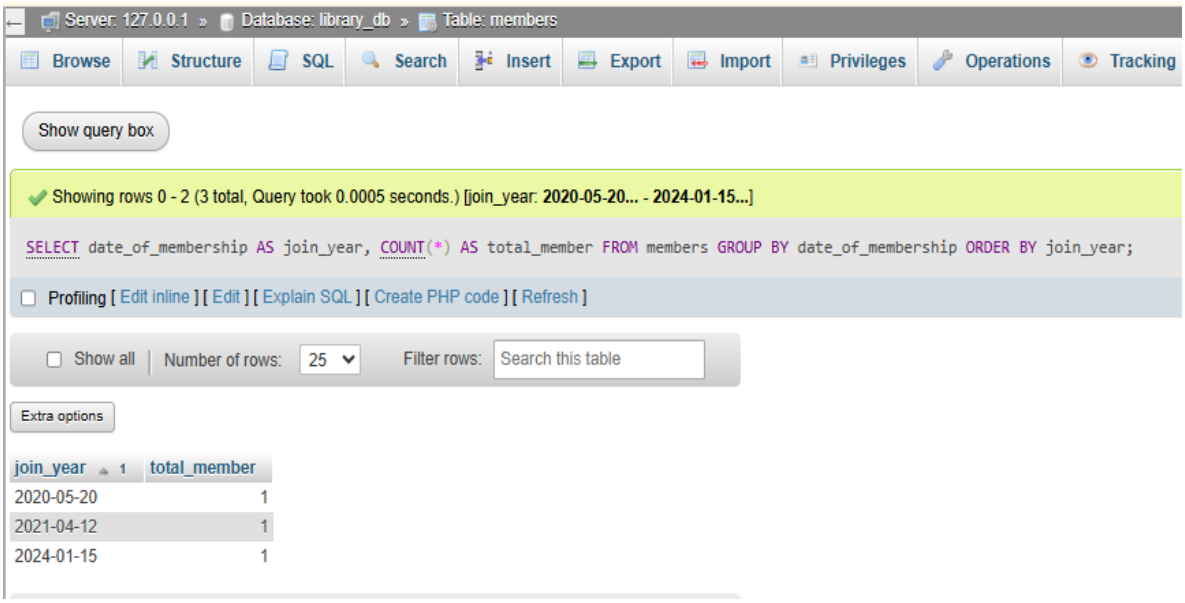
Lab 3 : Group books by genre and display the total number of books in each genre.

- SELECT genre, COUNT(*) as total_book FROM books GROUP BY genre;



Lab 4 : Group members by the year they joined and find the number of members who joined each year.

- SELECT date_of_membership AS join_year, COUNT(*) AS total_member FROM members GROUP BY date_of_membership ORDER BY join_year;



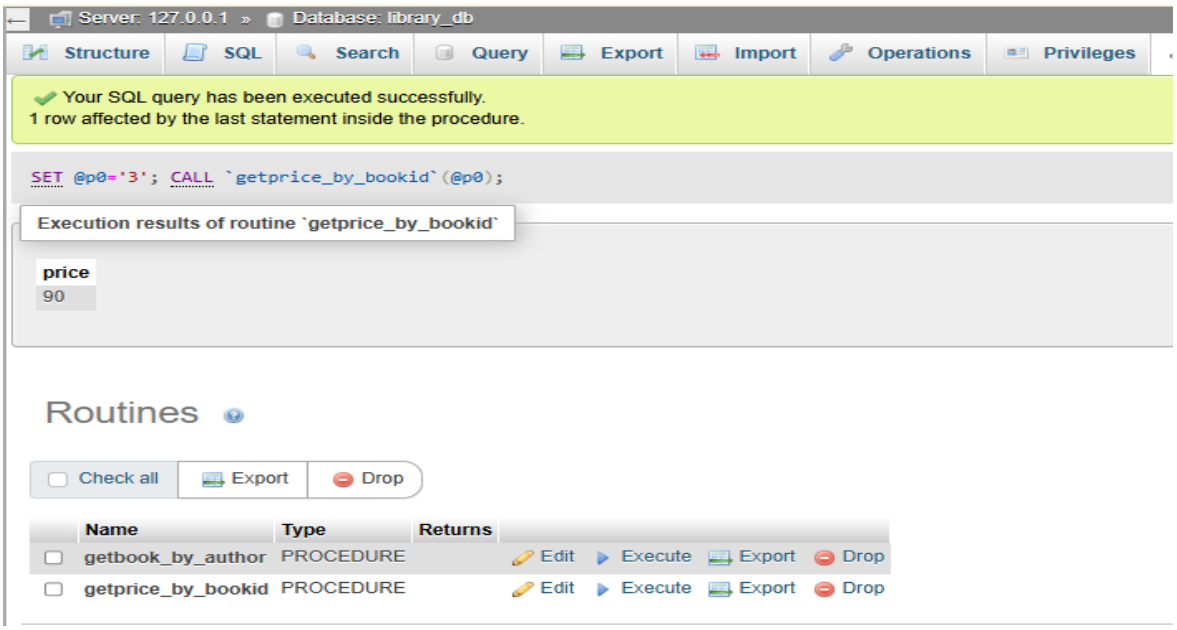
16. SQL Stored Procedure

Lab 3 : Write a stored procedure to retrieve all books by a particular author.

- DELIMITER //
  CREATE PROCEDURE getbook_by_author (IN author_name varchar(50))
  BEGIN
  SELECT * FROM books WHERE author = author_name;
  END //
  DELIMITER ;

Lab 4 : Write a stored procedure that takes book_id as an argument and returns the price of the book.

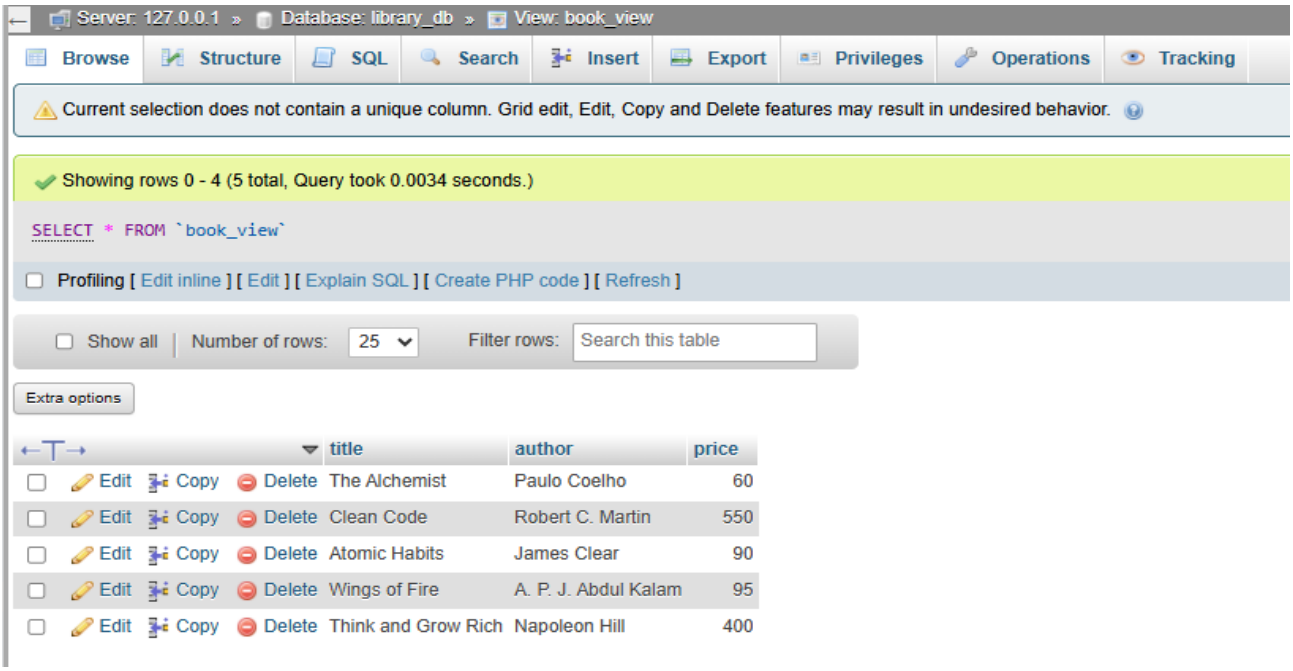
- DELIMITER //
  CREATE PROCEDURE getprice_by_bookid (IN b_id int)
  BEGIN
  SELECT price FROM books WHERE book_id = b_id;
  END//
  DELIMITER ;



17. SQL View

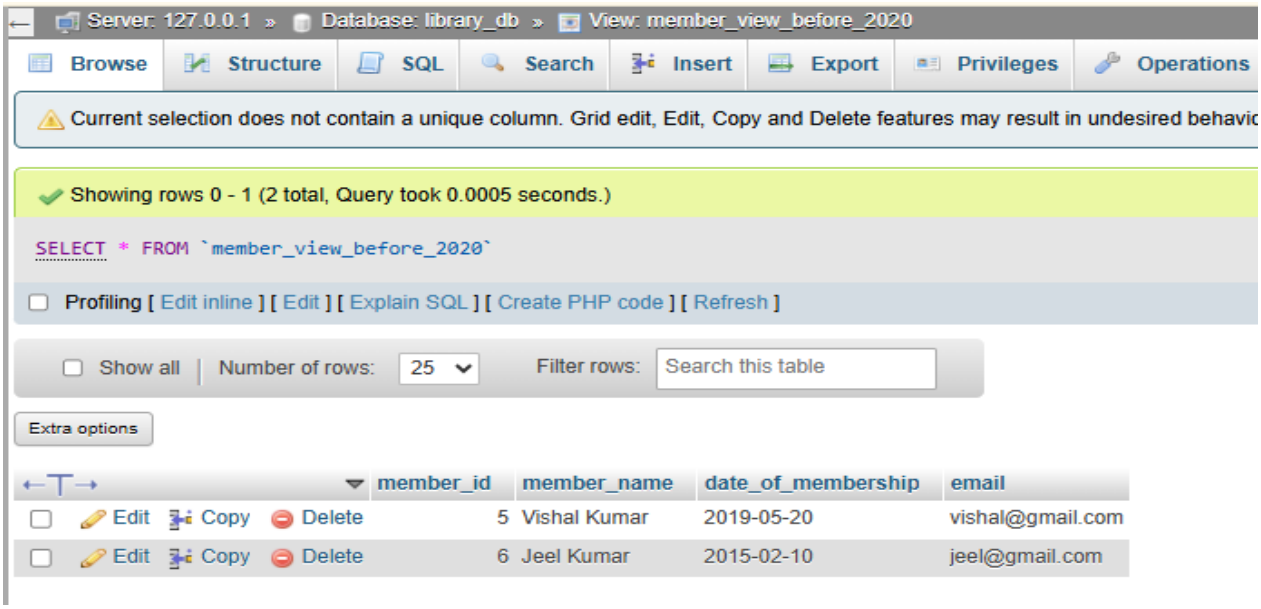
Lab 3 : Create a view to show only the title, author, and price of books from the books table.

- CREATE VIEW book_view AS SELECT title,author,price FROM books



Lab 4 : Create a view to display members who joined before 2020.

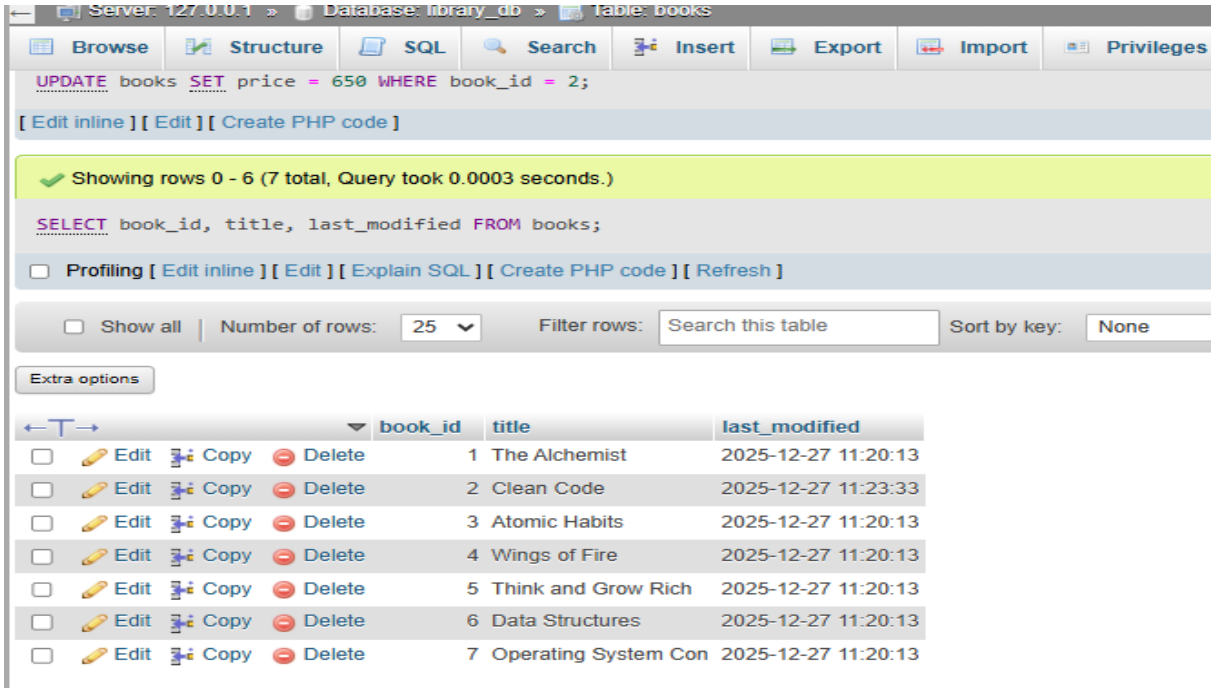
- CREATE VIEW member_view_before_2020 AS SELECT * FROM members WHERE date_of_membership < '2020-01-01'



18. SQL Trigger

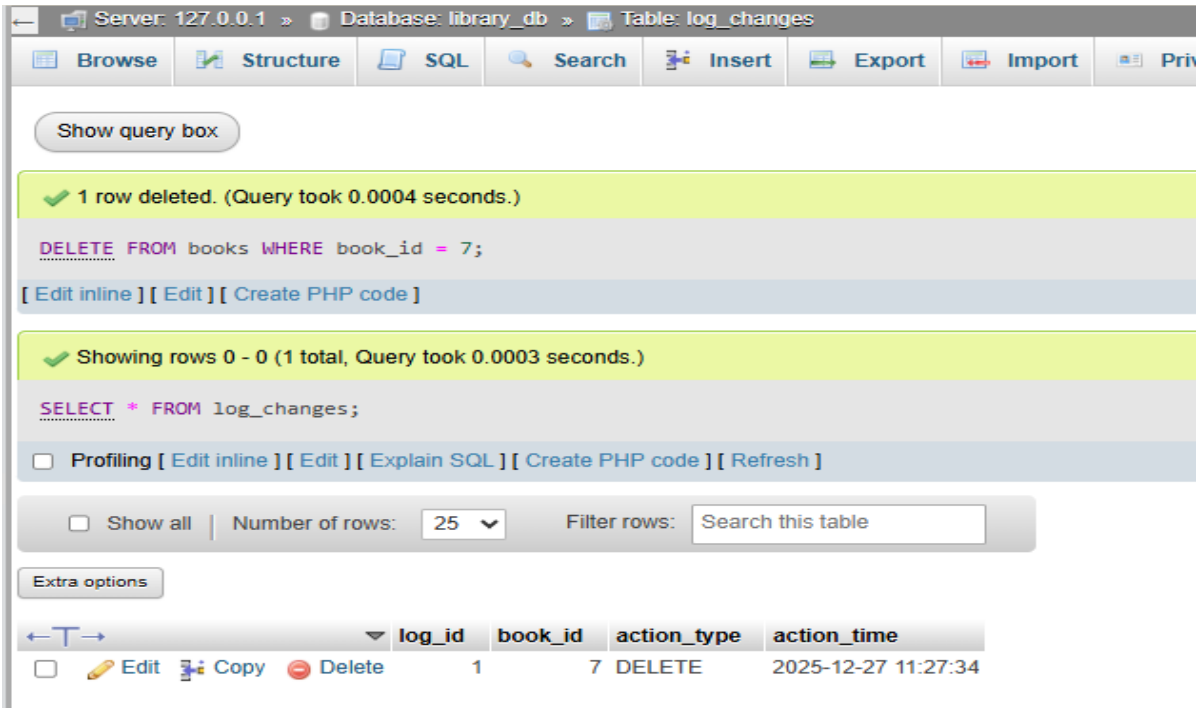
Lab 3 : Create a trigger to automatically update the last_modified timestamp of the books table whenever a record is updated.

- DELIMITER $$
  CREATE TRIGGER trg_update_books_timestamp
  BEFORE UPDATE ON books
  FOR EACH ROW
  BEGIN
  SET NEW.last_modified = CURRENT_TIMESTAMP;
  END$$
  DELIMITER ;



Lab 4 : Create a trigger that inserts a log entry into a log_changes table whenever a DELETE operation is performed on the books table.

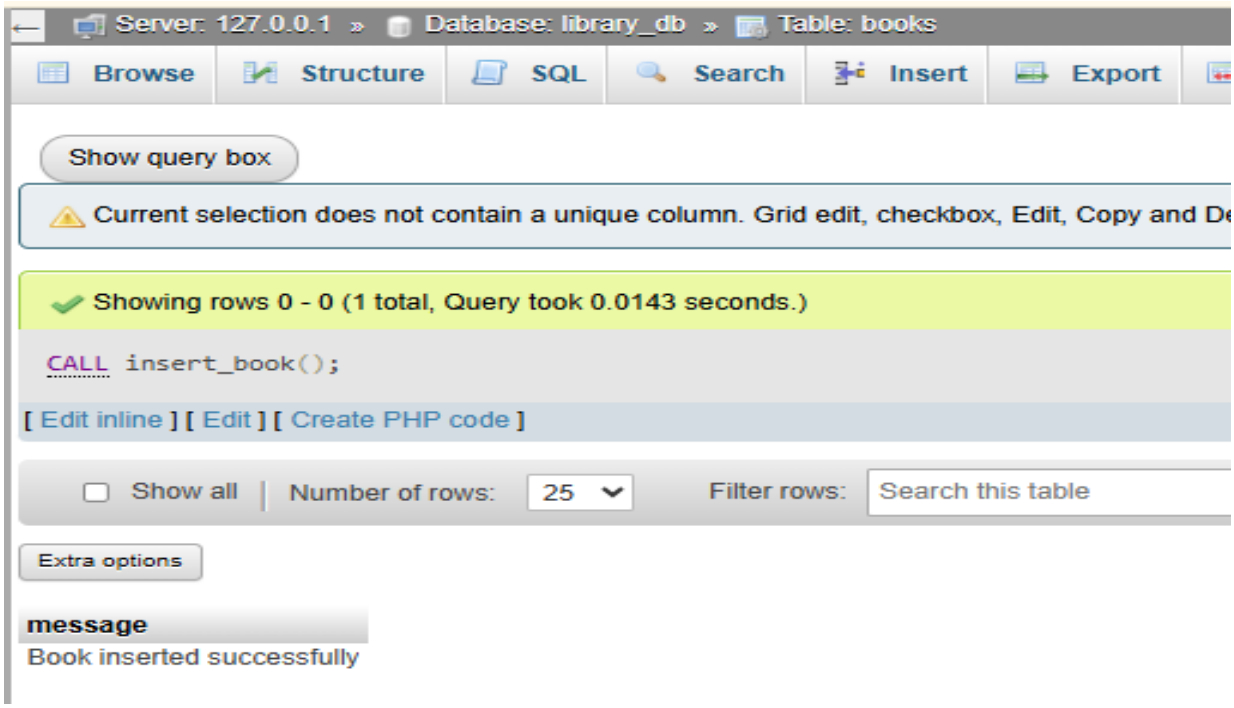
- DELIMITER $$
  CREATE TRIGGER trg_books_delete_log
  AFTER DELETE ON books
  FOR EACH ROW
  BEGIN
    INSERT INTO log_changes (book_id, action_type)
   VALUES (OLD.book_id, 'DELETE');
  END$$
  DELIMITER ;



19. Introduction to PL/SQL

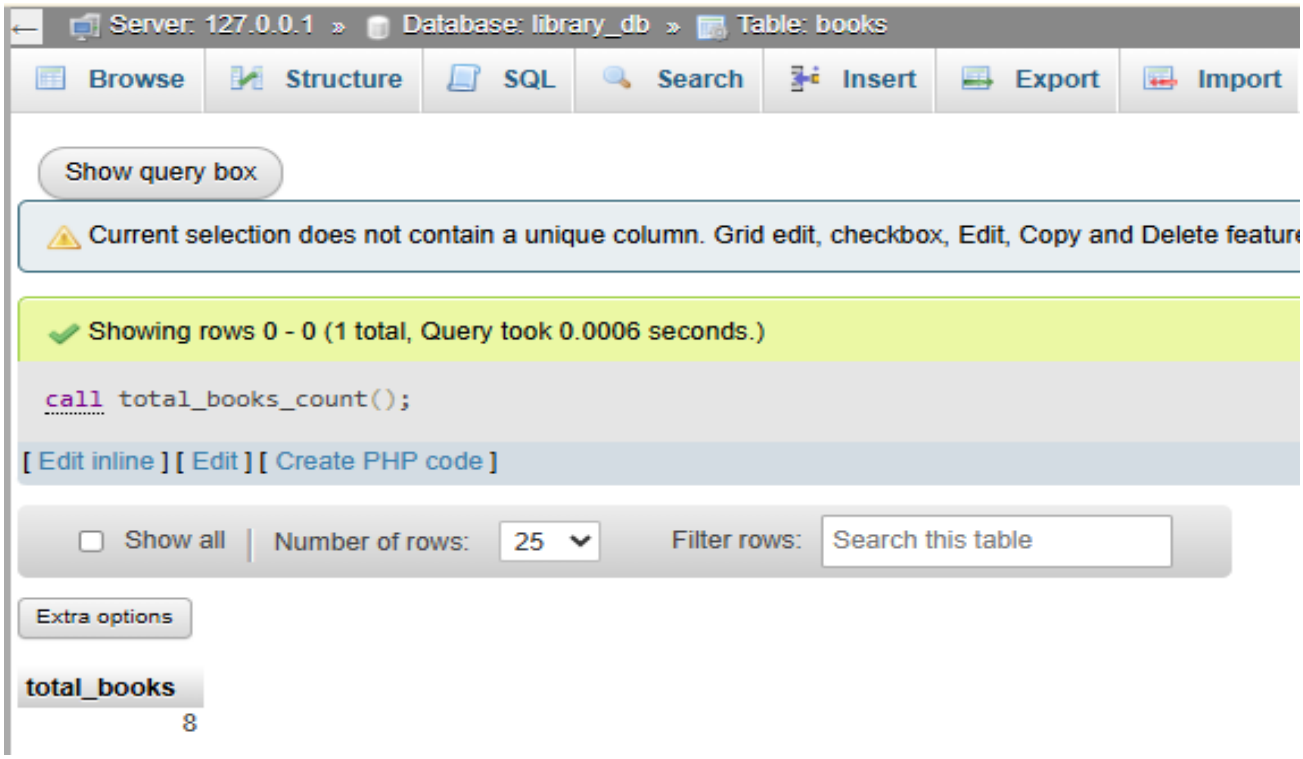
Lab 3 : Write a PL/SQL block to insert a new book into the books table and display a confirmation message.

- DELIMITER $$
  CREATE PROCEDURE insert_book()
  BEGIN
   INSERT INTO books
    (book_id, title, author, publisher, year_of_publication, price, genre)
     VALUES (10, 'MySQL Basics', 'John Smith', 'Pearson', 2023, 450, 'Database');
     SELECT 'Book inserted successfully' AS message;
  END$$
  DELIMITER ;



Lab 4 : Write a PL/SQL block to display the total number of books in the books table.

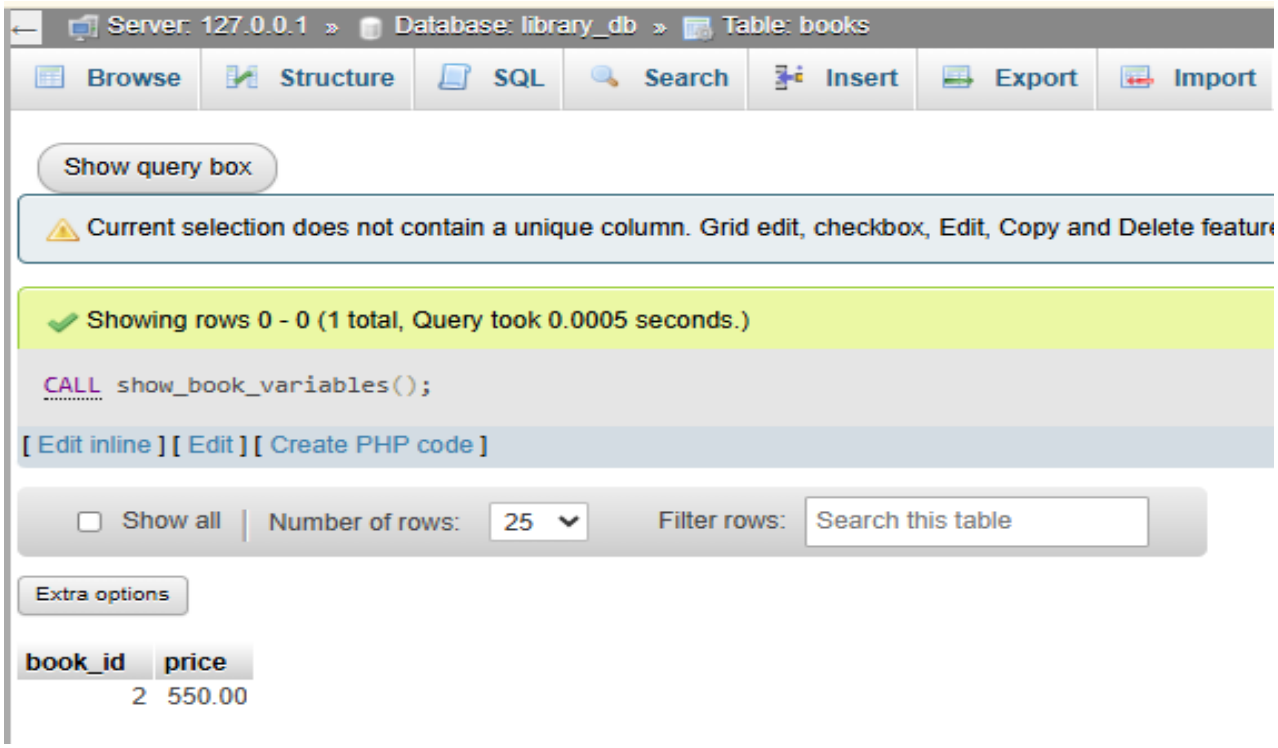
- ```
  DELIMITER $$
  CREATE PROCEDURE total_books_count()
  BEGIN
     DECLARE total INT;
     SELECT COUNT(*) INTO total FROM books;
     SELECT total AS total_books;
  END$$
  DELIMITER ;
  ```


  ```

20. PL/SQL Syntax

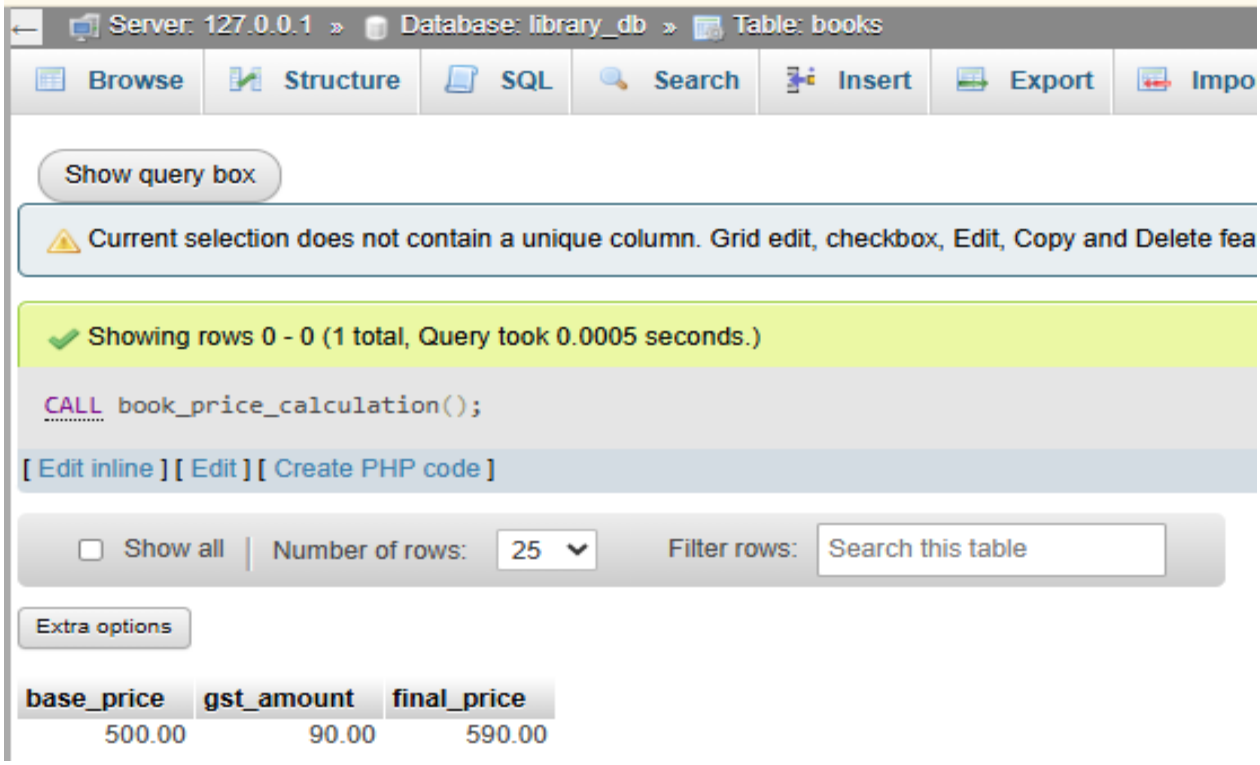
Lab 3 : Write a PL/SQL block to declare variables for book_id and price, assign values, and display the results.

- ```
  DELIMITER $$
  CREATE PROCEDURE show_book_variables()
  BEGIN
     DECLARE v_book_id INT;
     DECLARE v_price DECIMAL(10,2);
     SET v_book_id = 2;
     SET v_price = 550.00;
     SELECT v_book_id AS book_id, v_price AS price;
  END$$
  DELIMITER ;
  ```


  ```

Lab 4 : Write a PL/SQL block using constants and perform arithmetic operations on book prices.

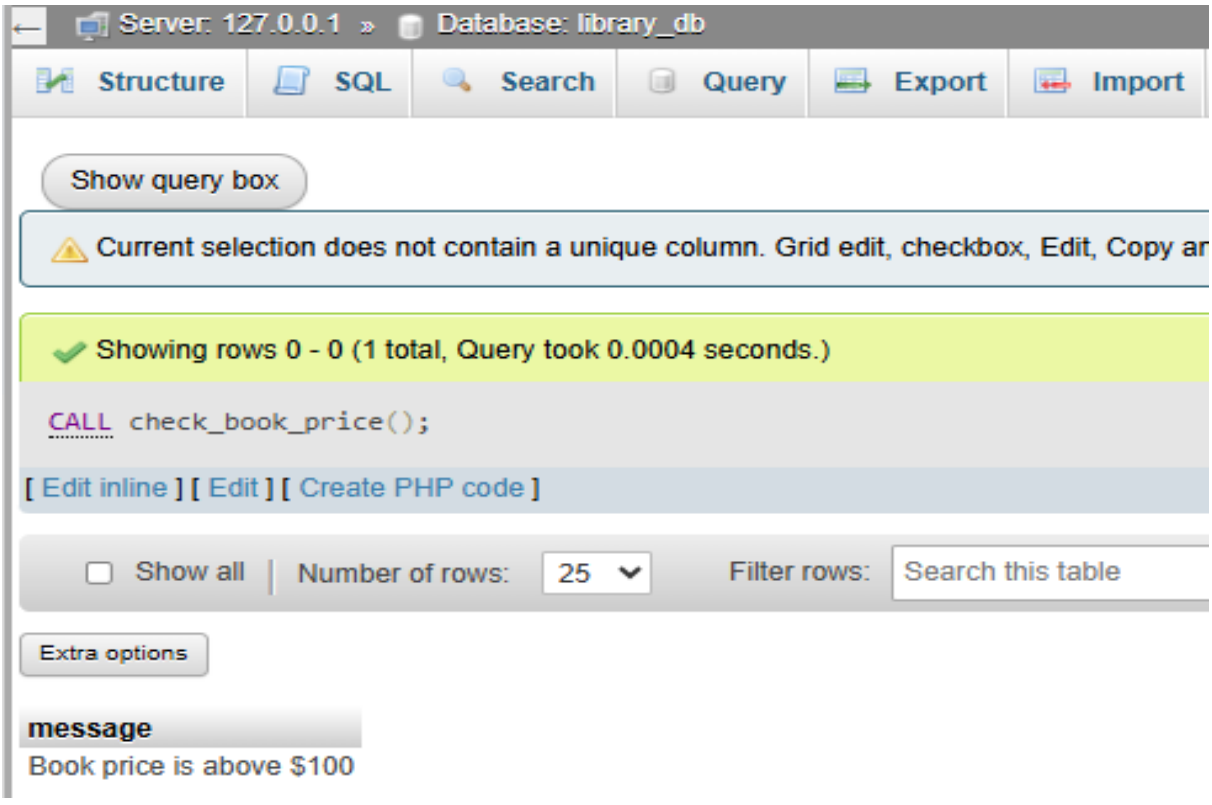
- ```
  DELIMITER $$
  CREATE PROCEDURE book_price_calculation()
  BEGIN
     DECLARE base_price DECIMAL(10,2) DEFAULT 500;
     DECLARE gst_rate DECIMAL(10,2) DEFAULT 18;
     DECLARE gst_amount DECIMAL(10,2);
     DECLARE final_price DECIMAL(10,2);
     SET gst_amount = (base_price * gst_rate) / 100;
     SET final_price = base_price + gst_amount;
     SELECT
        base_price AS base_price,
        gst_amount AS gst_amount,
        final_price AS final_price;
  END$$
  DELIMITER ;
  ```


  ```

21. PL/SQL Control Structures

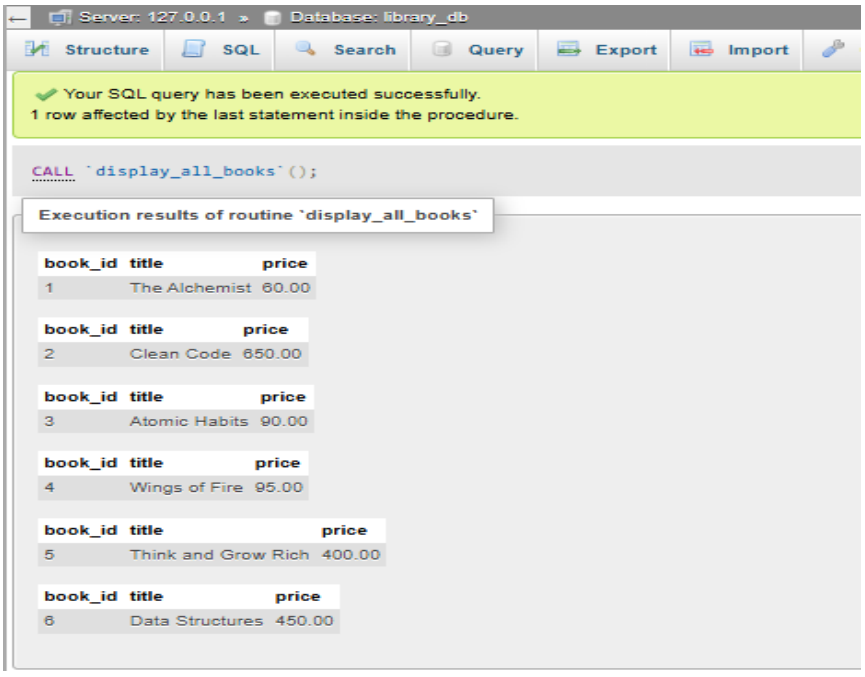
Lab 3 : Write a PL/SQL block using IF-THEN-ELSE to check if a book's price is above $100 and print a message accordingly.

- ```
  DELIMITER $$
  CREATE PROCEDURE check_book_price()
  BEGIN
    DECLARE v_price DECIMAL(10,2);
    SET v_price = 150;
    IF v_price > 100 THEN
      SELECT 'Book price is above $100' AS message;
    ELSE
      SELECT 'Book price is $100 or below' AS message;
    END IF;
  END$$
  DELIMITER ;
  ```


  ```

Lab 4 : Use a FOR LOOP in PL/SQL to display the details of all books one by one.

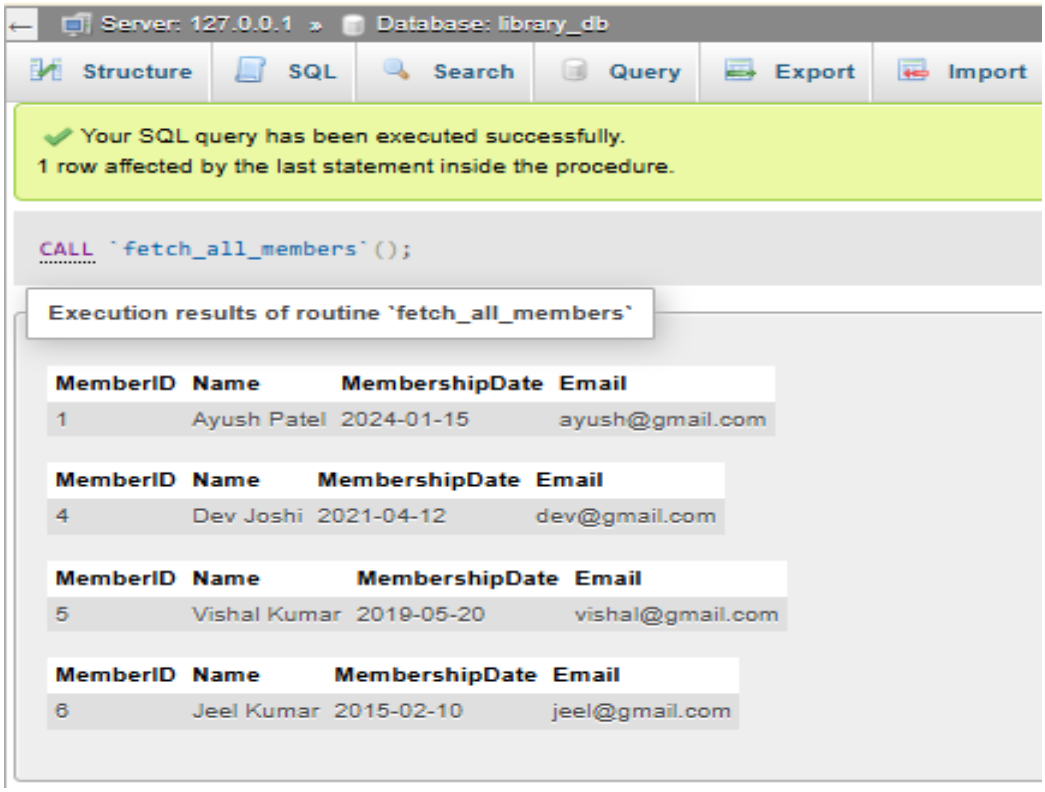
- ```
  DELIMITER $$
  CREATE PROCEDURE display_all_books()
  BEGIN
    DECLARE done INT DEFAULT 0;
    DECLARE v_id INT;
    DECLARE v_title VARCHAR(100);
    DECLARE v_price DECIMAL(10,2);
    DECLARE book_cursor CURSOR FOR
      SELECT book_id, title, price FROM books;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    OPEN book_cursor;
    read_loop: LOOP
      FETCH book_cursor INTO v_id, v_title, v_price;
      IF done = 1 THEN
        LEAVE read_loop;
      END IF;
      SELECT v_id AS book_id, v_title AS title, v_price AS price;
    END LOOP;
    CLOSE book_cursor;
  END$$
  DELIMITER ;
  ```


  ```

22. SQL Cursors

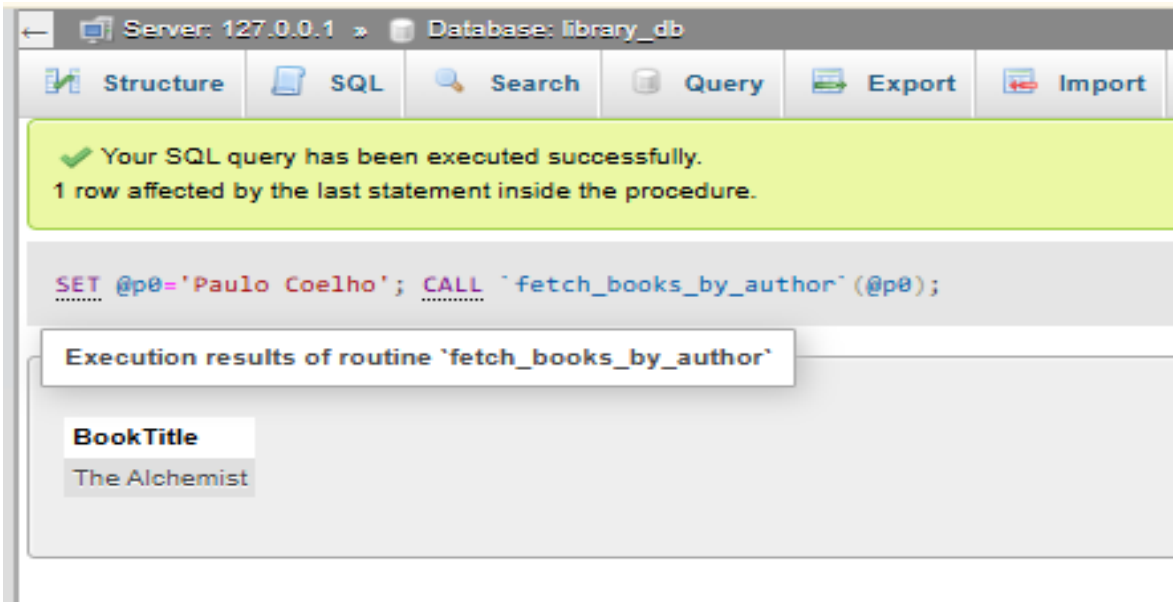
Lab 3 : Write a PL/SQL block using an explicit cursor to fetch and display all records from the members table.

- ```
  DELIMITER //
  CREATE PROCEDURE fetch_all_members()
  BEGIN
    DECLARE v_member_id INT;
    DECLARE v_member_name VARCHAR(100);
    DECLARE v_date_of_membership DATE;
    DECLARE v_email VARCHAR(100);
    DECLARE done INT DEFAULT 0;
    DECLARE member_cursor CURSOR FOR
      SELECT member_id, member_name, date_of_membership, email FROM
  members;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    OPEN member_cursor;
    read_loop: LOOP
      FETCH member_cursor INTO v_member_id, v_member_name,
  v_date_of_membership, v_email;
      IF done = 1 THEN
        LEAVE read_loop;
      END IF;
      SELECT v_member_id AS MemberID,
          v_member_name AS Name,
           v_date_of_membership AS MembershipDate,
           v_email AS Email;
    END LOOP;
    CLOSE member_cursor;
  END;
  //
  DELIMITER ;
  ```
  ```



Lab 4 : Create a cursor to retrieve books by a particular author and display their titles.

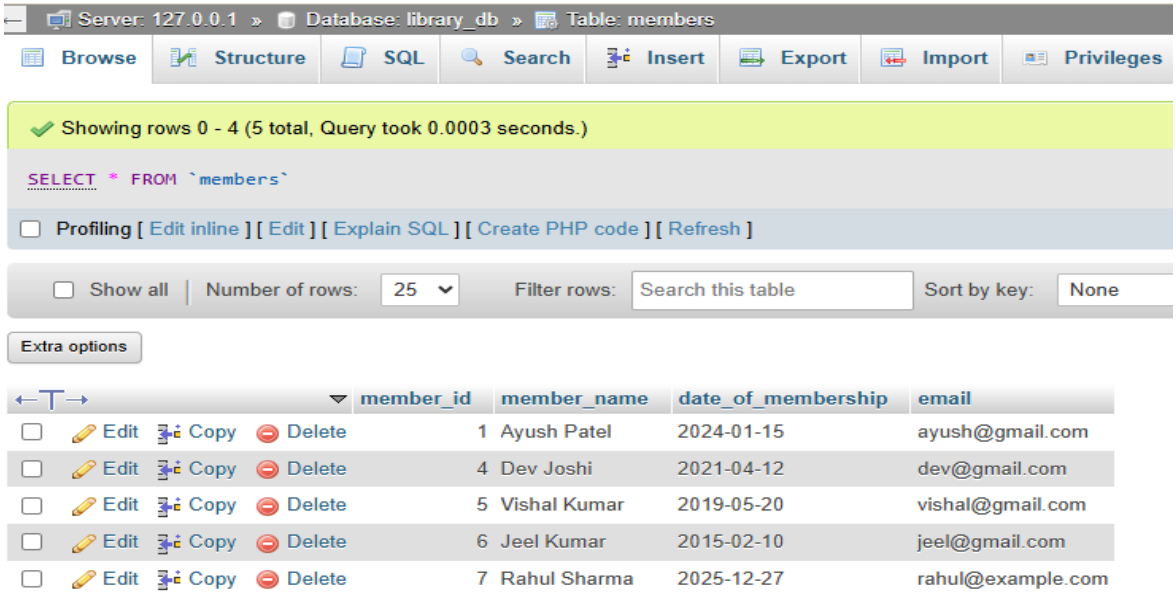
- ```
  DELIMITER //
  CREATE PROCEDURE fetch_books_by_author(IN author_name VARCHAR(100))
  BEGIN
    DECLARE v_title VARCHAR(255);
    DECLARE done INT DEFAULT 0;
    DECLARE books_cursor CURSOR FOR
      SELECT title FROM books WHERE author = author_name;
    DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = 1;
    OPEN books_cursor;
    read_loop: LOOP
      FETCH books_cursor INTO v_title;
      IF done = 1 THEN
        LEAVE read_loop;
      END IF;
      SELECT v_title AS BookTitle;
    END LOOP;
    CLOSE books_cursor;
  END;
  //
  DELIMITER ;
  ```
  ```



23. Rollback and Commit Savepoint

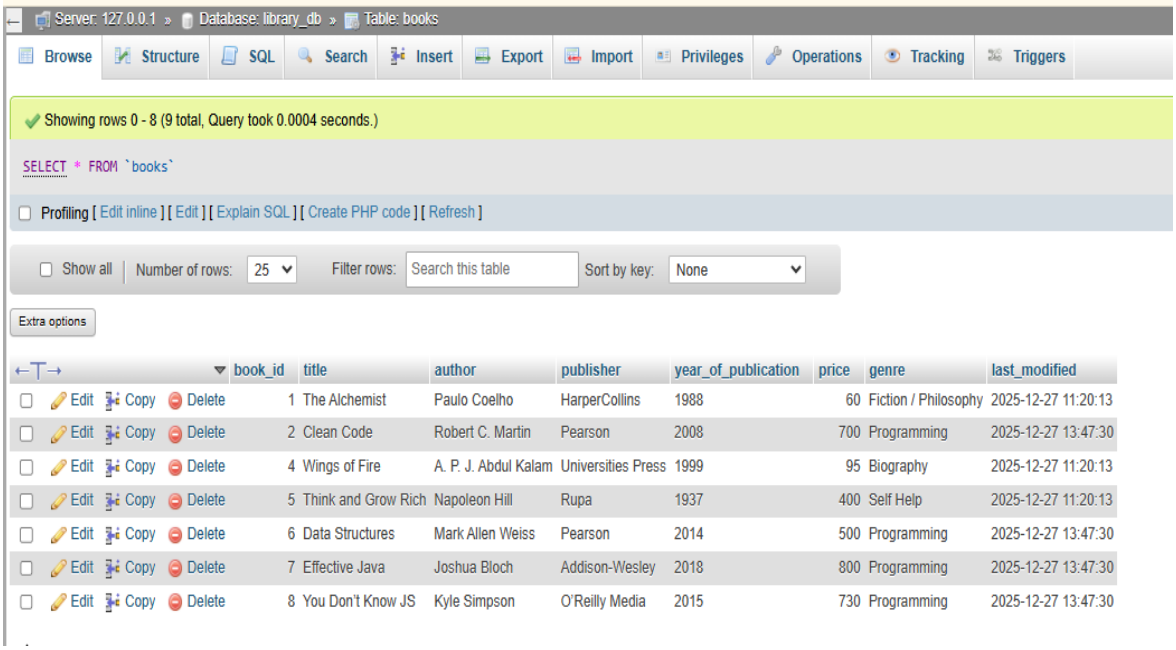
Lab 3 : Perform a transaction that includes inserting a new member, setting a SAVEPOINT, and rolling back to the savepoint after making updates.

- START TRANSACTION;
  INSERT INTO members (member_name,date_of_membership,email)
  VALUES ('Rahul Sharma', '2025-12-27', 'rahul@example.com');
  SAVEPOINT before_update;
  UPDATE members
  SET email = 'rahul_new@example.com'
  WHERE member_id = 7;
  ROLLBACK TO SAVEPOINT before_update;
  COMMIT;



Lab 4 : Use COMMIT after successfully inserting multiple books into the books table, then use ROLLBACK to undo a set of changes made after a savepoint.

- START TRANSACTION;
  INSERT INTO books (title, author, publisher, year_of_publication, price, genre)
  VALUES
  ('Effective Java', 'Joshua Bloch', 'Addison-Wesley', 2018, 750, 'Programming'),
  ('You Don’t Know JS', 'Kyle Simpson', 'O’Reilly Media', 2015, 680, 'Programming');
  COMMIT;
  START TRANSACTION;
  SAVEPOINT sp_books;
  UPDATE books
  SET price = price + 50
  WHERE genre = 'Programming';
  DELETE FROM books
  WHERE title = 'Atomic Habits';
  ROLLBACK TO sp_books;
  COMMIT;



book_id	title	author	publisher	year_of_publication	price	genre	last_modified
1	The Alchemist	Paulo Coelho	HarperCollins	1988	60	Fiction / Philosophy	2025-12-27 11:20:13
2	Clean Code	Robert C. Martin	Pearson	2008	700	Programming	2025-12-27 13:47:30
4	Wings of Fire	A. P. J. Abdul Kalam	Universities Press	1999	95	Biography	2025-12-27 11:20:13
5	Think and Grow Rich	Napoleon Hill	Rupa	1937	400	Self Help	2025-12-27 11:20:13
6	Data Structures	Mark Allen Weiss	Pearson	2014	500	Programming	2025-12-27 13:47:30
7	Effective Java	Joshua Bloch	Addison-Wesley	2018	800	Programming	2025-12-27 13:47:30
8	You Don't Know JS	Kyle Simpson	O'Reilly Media	2015	730	Programming	2025-12-27 13:47:30