ASSIGNMENT 2 (DQL/DML)

Write the following SQL Queries Based on the library database:

1. Get all books with titles containing 'The' followed by any characters.

Showing rows 0 - 17 (18 total, Query took 0.0006 seconds.)

```
SELECT* from books WHERE title LIKE '%The%';
```

[Edit inline] [Edit] [Create PHP code]

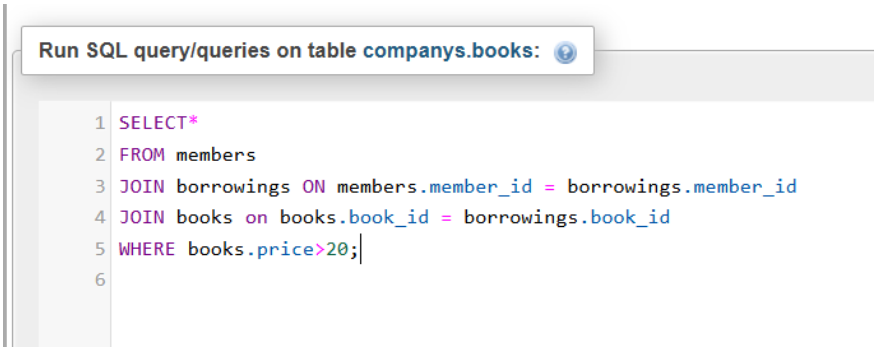
Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

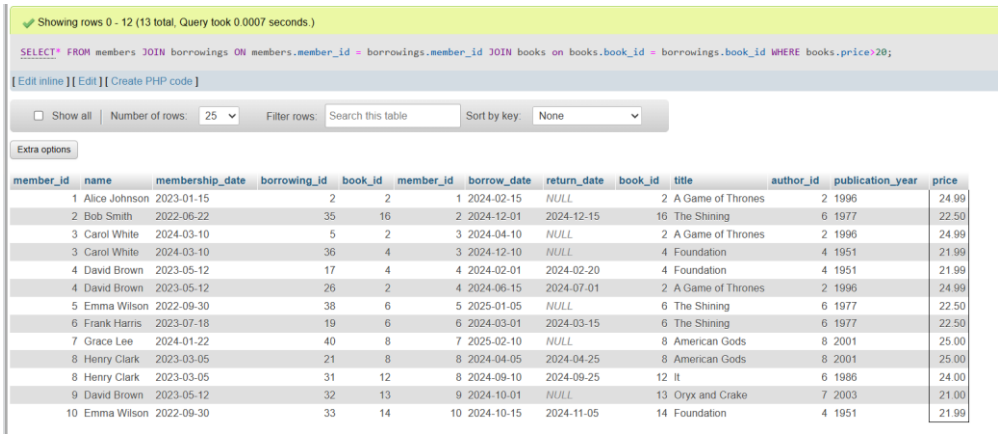
Extra options

	book_id	title	author_id	publication_year	price
Edit Copy Delete	1	Harry Potter and the Philosopher's Stone	1	1997	19.99
Edit Copy Delete	3	The Hobbit	3	1937	14.99
Edit Copy Delete	5	Murder on the Orient Express	5	1934	18.99
Edit Copy Delete	6	The Shining	6	1977	22.50
Edit Copy Delete	7	The Handmaid's Tale	7	1985	20.00
Edit Copy Delete	9	The Silmarillion	3	1977	15.50
Edit Copy Delete	11	And Then There Were None	5	1939	17.50
Edit Copy Delete	15	Murder on the Orient Express	5	1934	18.99
Edit Copy Delete	16	The Shining	6	1977	22.50
Edit Copy Delete	17	The Handmaid's Tale	7	1985	20.00
Edit Copy Delete	19	The Silmarillion	3	1977	15.50
Edit Copy Delete	21	And Then There Were None	5	1939	17.50
Edit Copy Delete	25	The Catcher in the Rye	10	1951	14.99
Edit Copy Delete	28	The Alchemist	13	1988	18.50
Edit Copy Delete	31	The Hunger Games	16	2008	21.99
Edit Copy Delete	33	The Moon is a Harsh Mistress	17	1966	19.99

Console

2. List members who have borrowed books with a price higher than $20





Showing rows 0 - 12 (13 total, Query took 0.0007 seconds.)

```
SELECT* FROM members JOIN borrowings ON members.member_id = borrowings.member_id JOIN books on books.book_id = borrowings.book_id WHERE books.price>20;
```

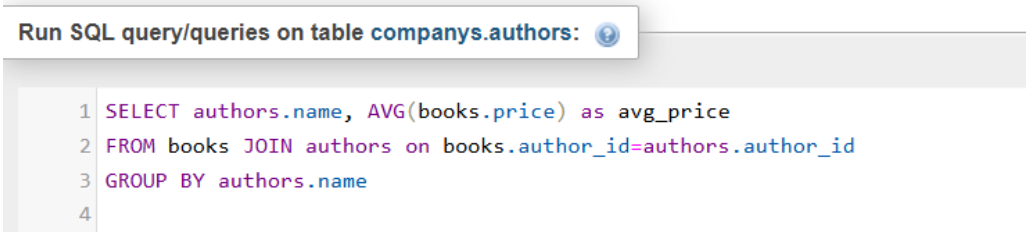
[Edit inline] [Edit] [Create PHP code]

Show all | Number of rows: 25 | Filter rows: Search this table | Sort by key: None

Extra options

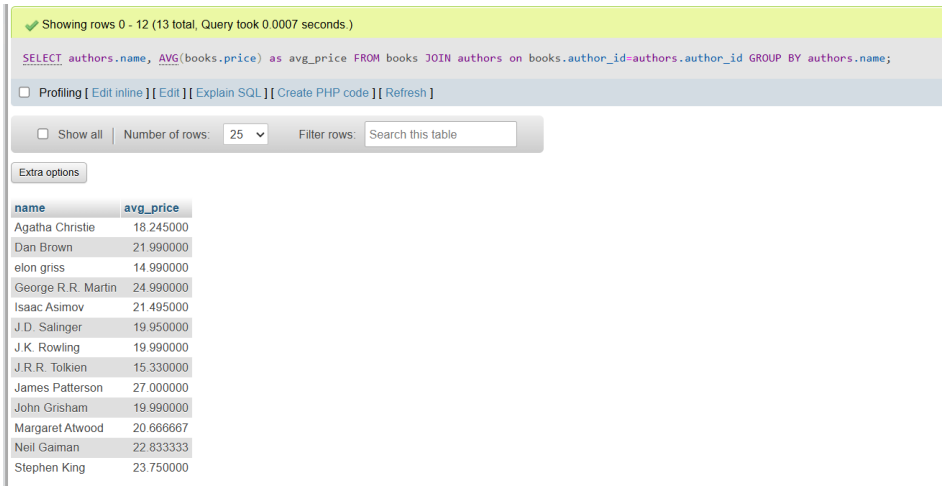
member_id	name	membership_date	borrowing_id	book_id	member_id	borrow_date	return_date	book_id	title	author_id	publication_year	price
1	Alice Johnson	2023-01-15	2	2	1	2024-02-15	NULL	2	A Game of Thrones	2	1996	24.99
2	Bob Smith	2022-06-22	35	16	2	2024-12-01	2024-12-15	16	The Shining	6	1977	22.50
3	Carol White	2024-03-10	5	2	3	2024-04-10	NULL	2	A Game of Thrones	2	1996	24.99
3	Carol White	2024-03-10	36	4	3	2024-12-10	NULL	4	Foundation	4	1951	21.99
4	David Brown	2023-05-12	17	4	4	2024-02-01	2024-02-20	4	Foundation	4	1951	21.99
4	David Brown	2023-05-12	26	2	4	2024-06-15	2024-07-01	2	A Game of Thrones	2	1996	24.99
5	Emma Wilson	2022-09-30	38	6	5	2025-01-05	NULL	6	The Shining	6	1977	22.50
6	Frank Harris	2023-07-18	19	6	6	2024-03-01	2024-03-15	6	The Shining	6	1977	22.50
7	Grace Lee	2024-01-22	40	8	7	2025-02-10	NULL	8	American Gods	8	2001	25.00
8	Henry Clark	2023-03-05	21	8	8	2024-04-05	2024-04-25	8	American Gods	8	2001	25.00
8	Henry Clark	2023-03-05	31	12	8	2024-09-10	2024-09-25	12	It	6	1986	24.00
9	David Brown	2023-05-12	32	13	9	2024-10-01	NULL	13	Oryx and Crake	7	2003	21.00
10	Emma Wilson	2022-09-30	33	14	10	2024-10-15	2024-11-05	14	Foundation	4	1951	21.99

3. Find the average price of books for each author, display the author name



Run SQL query/queries on table companys.authors:

```
1 SELECT authors.name, AVG(books.price) as avg_price
2 FROM books JOIN authors on books.author_id=authors.author_id
3 GROUP BY authors.name
4
```



Showing rows 0 - 12 (13 total, Query took 0.0007 seconds.)

```
SELECT authors.name, AVG(books.price) as avg_price FROM books JOIN authors on books.author_id=authors.author_id GROUP BY authors.name;
```

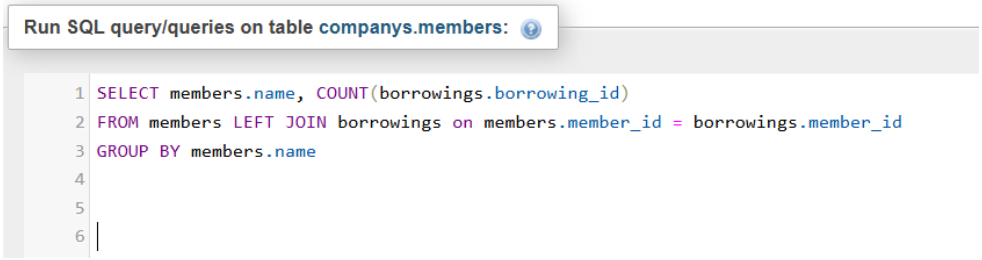
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]

Show all | Number of rows: 25 | Filter rows: Search this table

Extra options

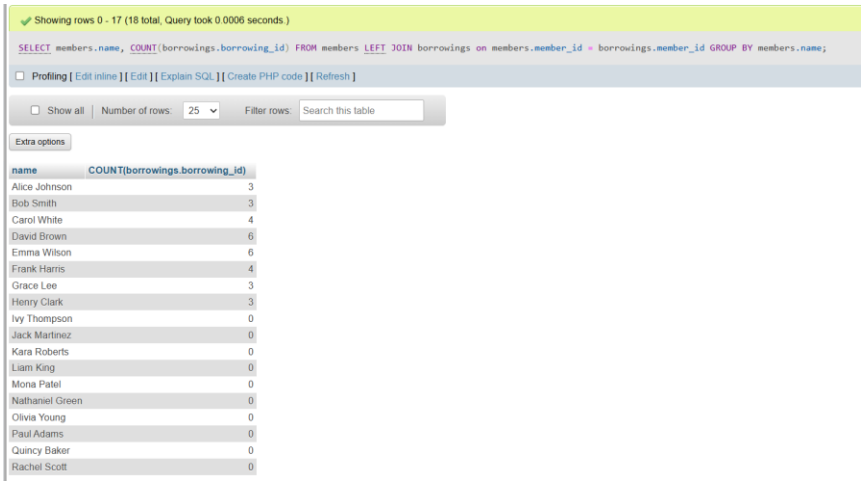
name	avg_price
Agatha Christie	18.245000
Dan Brown	21.990000
elon griss	14.990000
George R.R. Martin	24.990000
Isaac Asimov	21.495000
J.D. Salinger	19.950000
J.K. Rowling	19.990000
J.R.R. Tolkien	15.330000
James Patterson	27.000000
John Grisham	19.990000
Margaret Atwood	20.666667
Neil Gaiman	22.833333
Stephen King	23.750000

4. The number of books each member has borrowed even if they do not borrow any book.

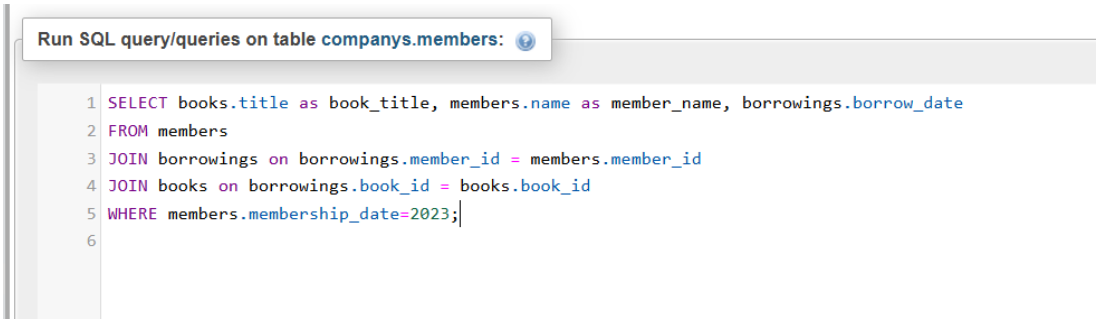


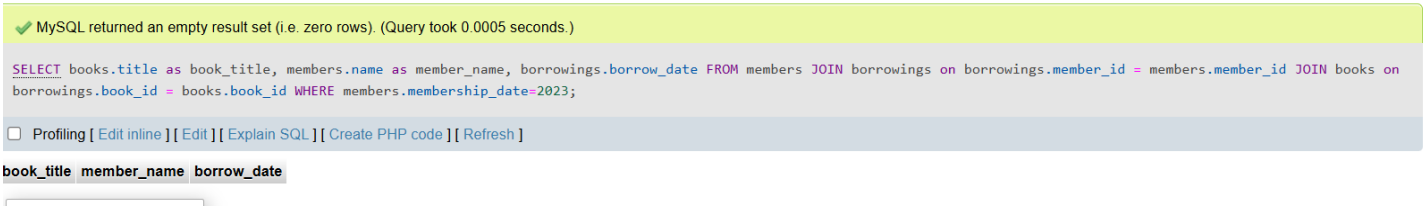
Run SQL query/queries on table companys.members:

```
1 SELECT members.name, COUNT(borrowings.borrowing_id)
2 FROM members LEFT JOIN borrowings on members.member_id = borrowings.member_id
3 GROUP BY members.name
4
5
6
```

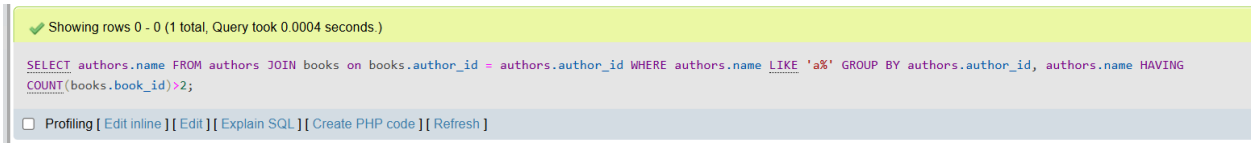


5. Find all books borrowed by members who joined in 2023, display book title, member name and borrow date





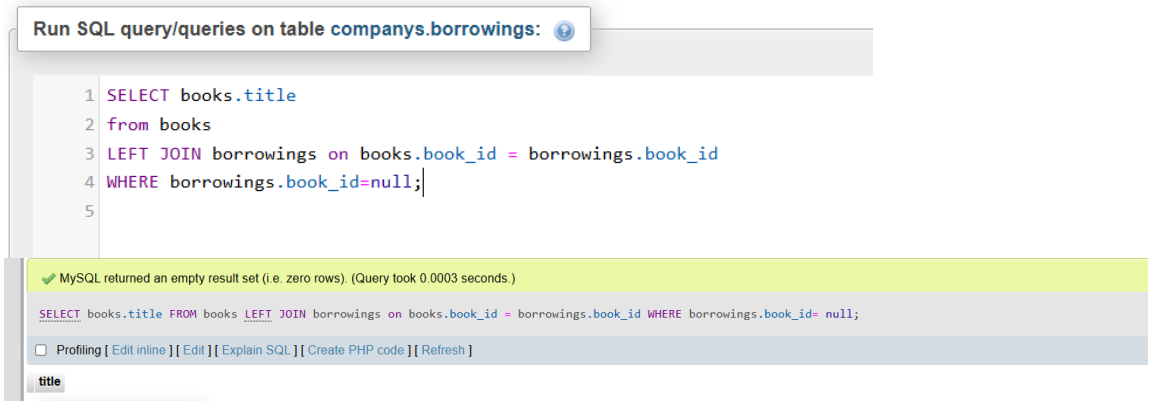
6. List authors who have written more than 2 books if their names starts with letter a



Run SQL query/queries on table companys.authors:

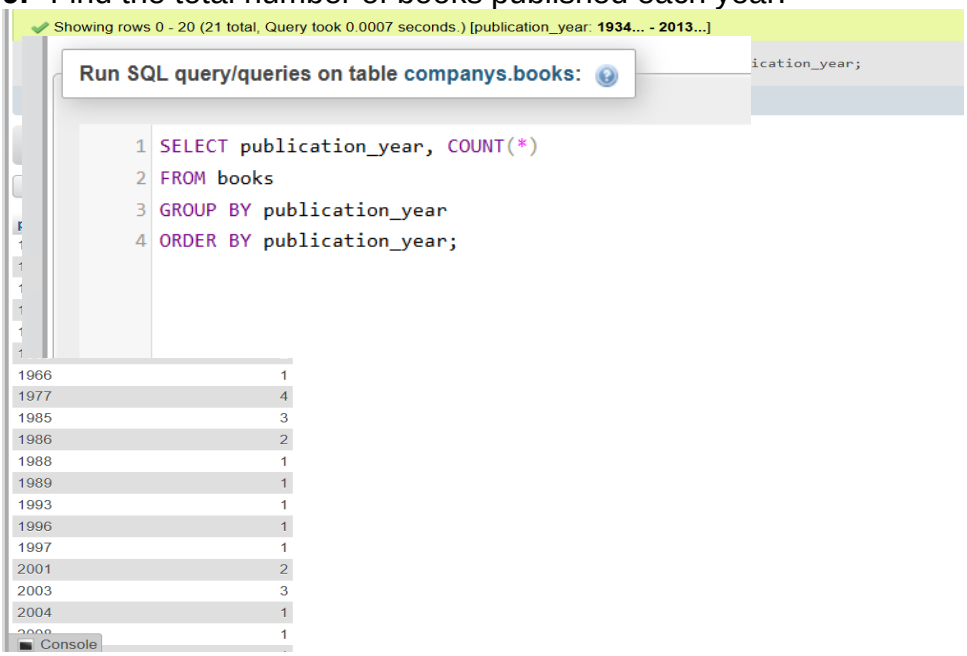
```
SELECT authors.name
FROM authors
JOIN books on books.author_id = authors.author_id
WHERE authors.name LIKE 'a%'
GROUP BY authors.author_id, authors.name
HAVING COUNT(books.book_id)>2
```

7. List all books that have never been borrowed.

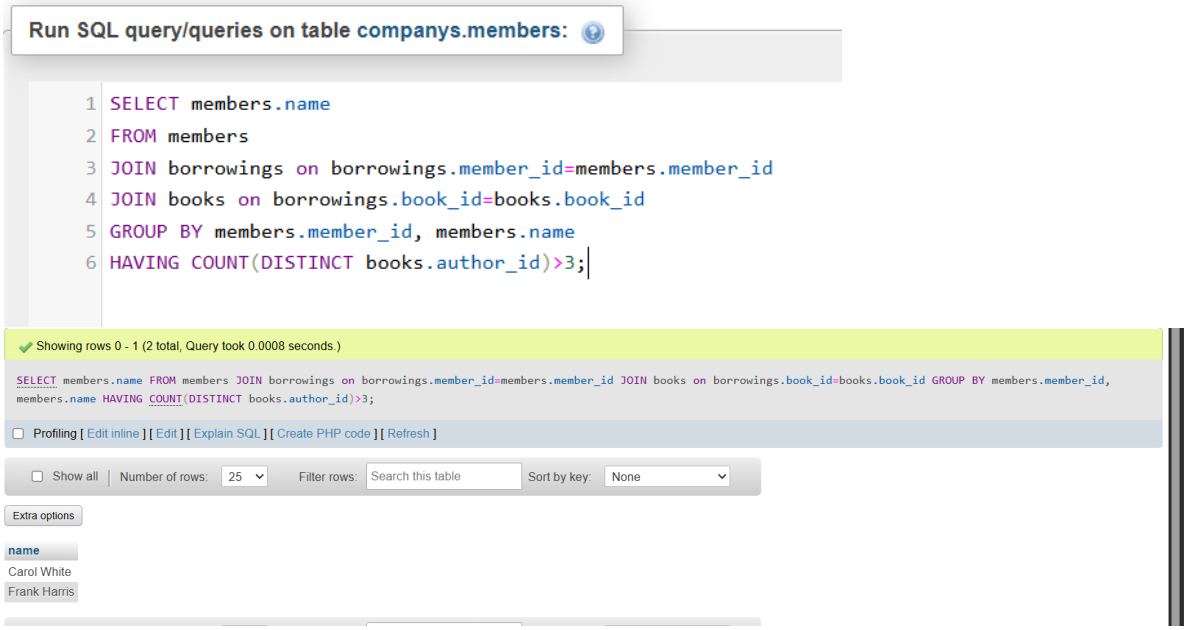


8. List all authors who have written books in more than 3 different publication years.

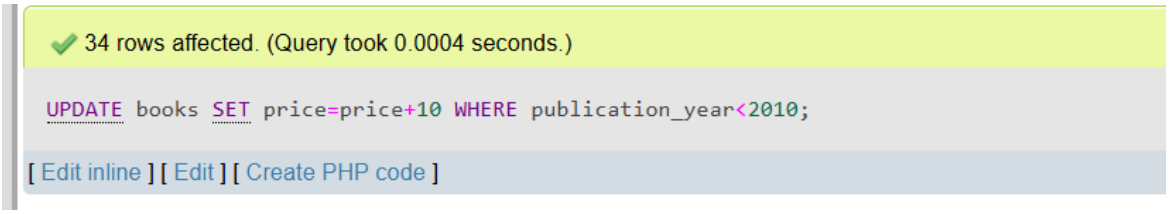
9. Find the total number of books published each year.



10. Find the members who have borrowed books from at least three different authors.



11. Increase the price of all books published before 2010 by 10.



Run SQL query/queries on table companys.books:

```
UPDATE books
SET price=price+10
WHERE publication_year<2010;
```

12. Change the return date of all borrowings that are still not returned (NULL) to today's date.

Run SQL query/queries on table companys.borrowings:

```
UPDATE borrowings
SET return_date=CURRENT_DATE
WHERE return_date= null;
```

0 rows affected. (Query took 0.0005 seconds.)

```
UPDATE borrowings SET return_date=CURRENT_DATE WHERE return_date= null;
```

[Edit inline] [Edit] [Create PHP code]

13. Update all members who joined before 2020 to have "(Old Member)" added to their name.

Run SQL query/queries on table companys.members:

```
UPDATE members
SET name = CONCAT(name, ' (old memeber)' )
WHERE membership_date<'2020-01-01' ;
```

0 rows affected. (Query took 0.0004 seconds.)

```
UPDATE members SET name = CONCAT(name, ' (old memeber)' ) WHERE membership_date<'2020-01-01';
```

[Edit inline] [Edit] [Create PHP code]

14. Increase prices of books written by authors who have published more than 3 books by 15%.

Run SQL query/queries on table companys.books:

```
UPDATE books
SET price = price*1.15
WHERE author_id IN(
    SELECT author_id
    FROM books
    GROUP BY author_id
    HAVING COUNT(book_id)>3
    );
```

17 rows affected. (Query took 0.0003 seconds.)

```
UPDATE books SET price = price*1.15 WHERE author_id IN( SELECT author_id FROM books GROUP BY author_id HAVING COUNT(book_id)>3 );
```

[Edit inline] [Edit] [Create PHP code]

15. Set the price of all books written by authors born before 1960 to 50.

Run SQL query/queries on table companys.books:

```
UPDATE books bo
JOIN authors au on bo.author_id=au.author_id
SET bo.price=50
WHERE au.birth_year<1960;
```

30 rows affected. (Query took 0.0004 seconds.)

```
UPDATE books bo JOIN authors au on bo.author_id=au.author_id SET bo.price=50 WHERE au.birth_year<1960;
```

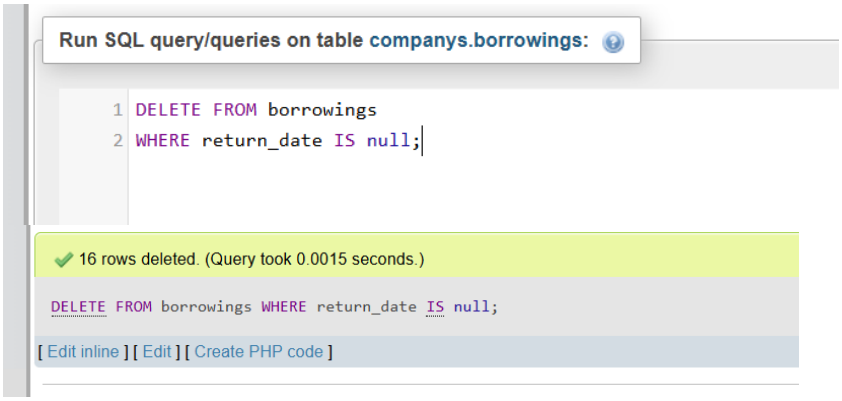
[Edit inline] [Edit] [Create PHP code]

16. Give a 20% discount on all books that cost more than 100.

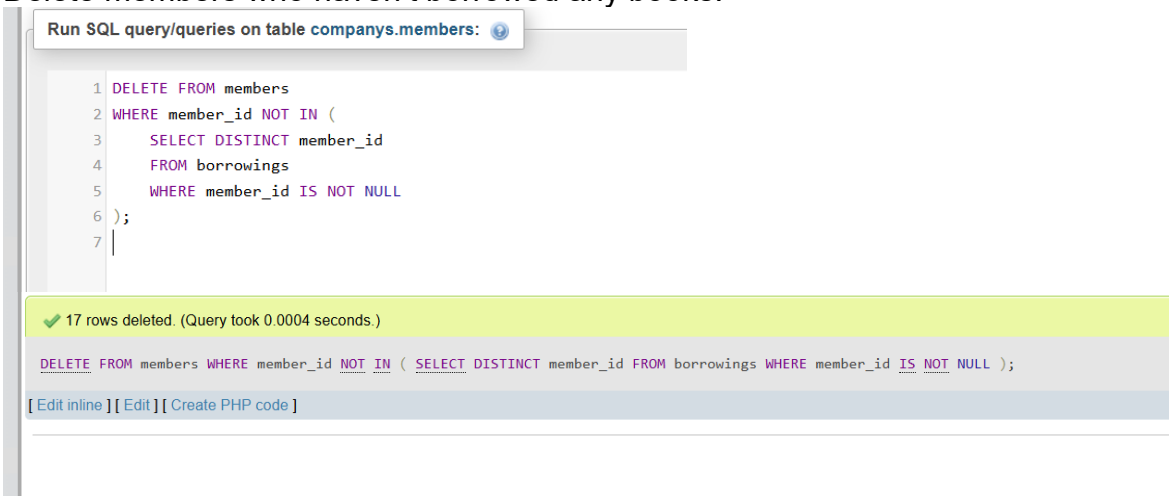
Run SQL query/queries on table companys.books:

```
UPDATE books
SET price=price*0.8
WHERE price>100;
```

17. Delete all borrow records where the book has not been returned.



18. Delete members who haven't borrowed any books.



19. Delete authors whose books have never been borrowed.

20. Delete all members who borrowed only books priced below $10.

Run SQL query/queries on table companys.members:

```
DELETE FROM members
WHERE member_id IN (
    SELECT m.member_id
    FROM members m
    WHERE NOT EXISTS (
        SELECT*
        FROM borrowings b
        JOIN books bo ON b.book_id = bo.book_id
        WHERE b.member_id = m.member_id
        AND bo.price >= 10
    )
    AND EXISTS (
        SELECT*
        FROM borrowings b
        WHERE b.member_id = m.member_id ));
```

0 rows deleted. (Query took 0.0005 seconds.)

```
DELETE FROM members WHERE member_id IN ( SELECT m.member_id FROM members m WHERE NOT EXISTS ( SELECT* FROM borrowings b JOIN books bo ON b.book_id = bo.book_id WHERE
b.member_id = m.member_id AND bo.price >= 10 ) AND EXISTS ( SELECT* FROM borrowings b WHERE b.member_id = m.member_id ));
```

[Edit inline] [Edit] [Create PHP code]