```sql
CREATE DATABASE BookManagement;
USE BookManagement;

#Q1
CREATE TABLE USER (
    user_id VARCHAR(10) PRIMARY KEY,
    username VARCHAR(10),
    email VARCHAR(20),
    dob DATE,
    gender VARCHAR(5),
    phoneno VARCHAR(10),
    address VARCHAR(20)
);

CREATE TABLE BOOK (
    isbn VARCHAR(20) PRIMARY KEY,
    title VARCHAR(50),
    author VARCHAR(50),
    genre VARCHAR(25),
    publication_date DATE,
    publisher VARCHAR(50),
    price DECIMAL(10, 2)
);
```

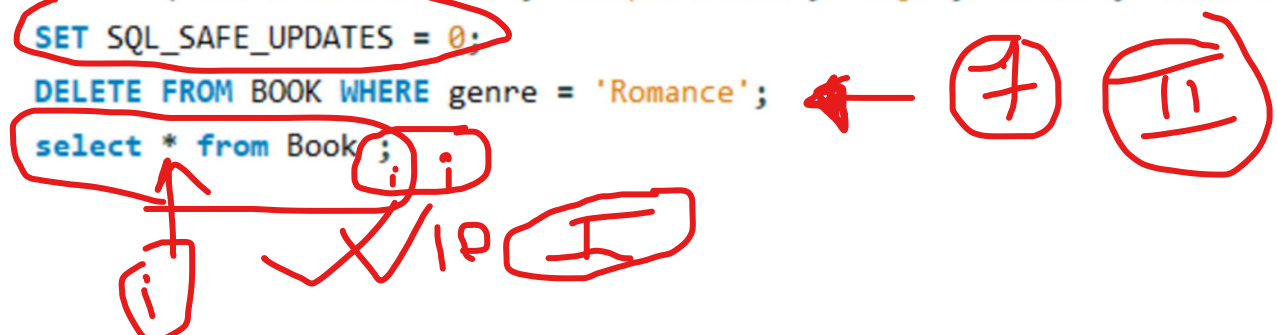
```sql
CREATE TABLE BORROWING (
    borrowing_id VARCHAR(10) PRIMARY KEY,
    user_id VARCHAR(10),
    isbn VARCHAR(20),
    borrow_date DATE,
    due_date DATE,
    return_date DATE,
    fine DECIMAL(10, 2),
    FOREIGN KEY (user_id) REFERENCES USER(user_id),
    FOREIGN KEY (isbn) REFERENCES BOOK(isbn)
);
```

```sql
SHOW TABLES;

DESCRIBE USER;
DESCRIBE BOOK;
DESCRIBE BORROWING;
```

```sql
#Q2
ALTER TABLE USER CHANGE COLUMN dob date_of_birth DATE;


#Q3
alter table book modify column PUBLISHER varchar(100),
                add column no_of_books int after PUBLISHER;


#Q4
SET SQL_SAFE_UPDATES = 0;
delete from user;


#Q5
Alter table user change column  date_of_birth dob date;
INSERT INTO USER (user_id, username, email, dob, gender, phoneno, address) VALUES
('UN001', 'Alice', 'alice@gmail.com', '1990-05-15', 'F', '9234567890', '123 Main St'),
('UN002', 'Harry', 'harry@yahoo.com', '1985-08-22', 'M', '9234566891', '60 Hobbson St'),
('UN003', 'Kate', 'kate@hotmail.com', '1970-03-18', 'F', '8234567898', '35 Willis St');
```

```sql
#Q6
INSERT INTO BOOK (isbn, title, author, genre, publication_date, publisher, price)
VALUES ('978-3-16-148410-0', 'Sample Book', 'Jane Doe', 'Fiction', '2023-01-01', 'Sample Publisher', 19.99);

INSERT INTO BORROWING (borrowing_id, user_id, isbn, borrow_date, due_date, return_date, fine)
VALUES ('BR001', 'UN001', '978-3-16-148410-0', '2024-08-01', '2024-08-15', NULL, 0.00);

UPDATE BORROWING SET return_date = '2023-02-27', fine = 25.00 WHERE borrowing_id = 'BR001';
select * from Borrowing ;
```

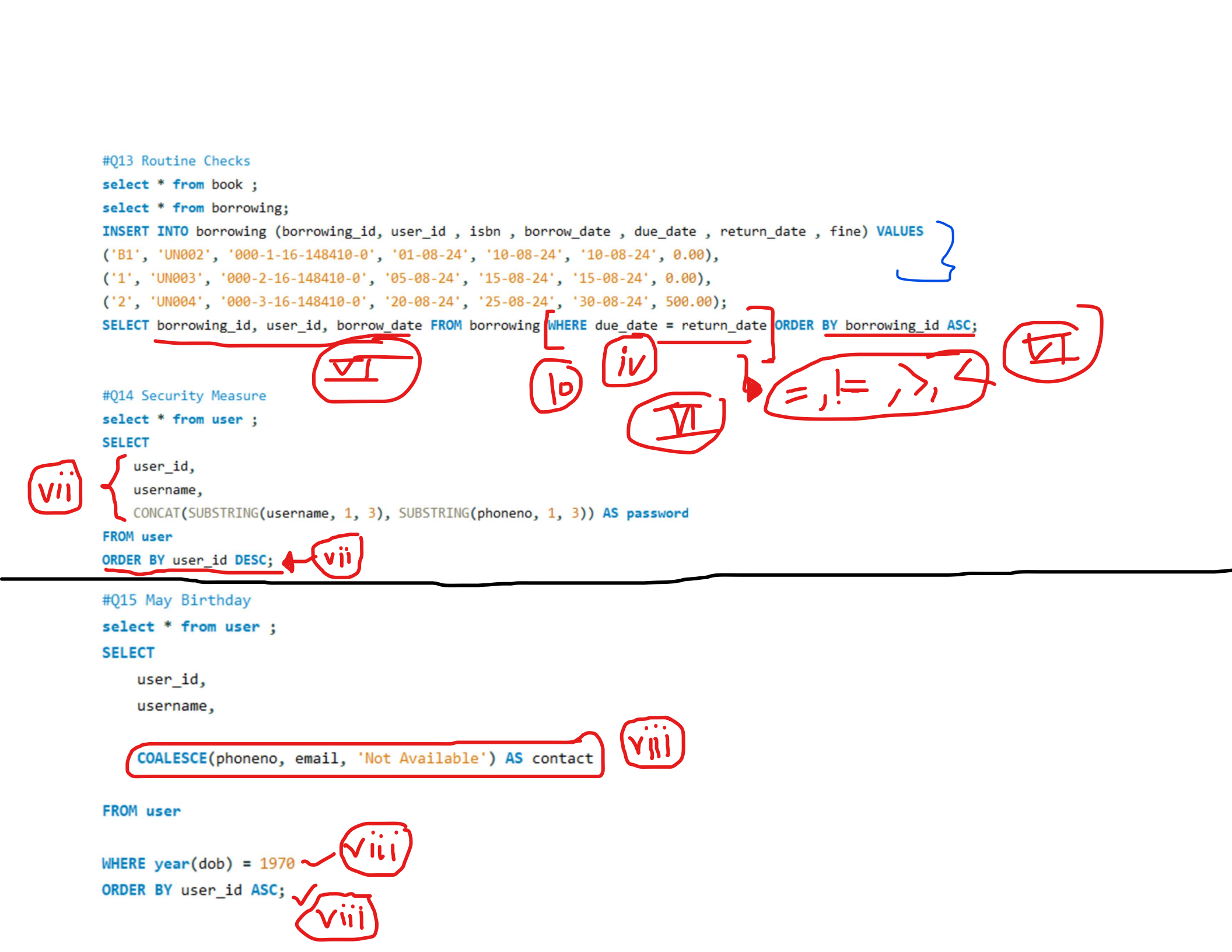
```sql
#Q7
INSERT INTO BOOK (isbn, title, author, genre, publication_date, publisher, price)
VALUES ('000-2-16-148410-0', 'Sample Book3', 'Surya', 'Psycho', '2024-08-30', 'Sample Publisher 4', 10.00),
       ('000-1-16-148410-0', 'Sample Book2', 'Teja', 'Crime', '2024-08-20', 'Sample Publisher 3', 1000.00);
SET SQL_SAFE_UPDATES = 0;
DELETE FROM BOOK WHERE genre = 'Romance';
select * from Book ;
```



```sql
#Q8
UPDATE Book SET title = case
    when isbn = '978-3-16-148410-0' then 'Potter'
    when isbn = '000-3-16-148410-0' then 'Harry'
    when isbn = '000-2-16-148410-0' then 'Ramayana'
    when isbn = '000-1-16-148410-0' then 'clayBook'
    END

    where title in ('Potter', 'PlayBook','Gita','Harry');


#Q9
UPDATE Book SET title = 'ClayBook' where isbn = '000-3-16-148410-0';
SELECT title, author, publication_date FROM BOOK ORDER BY title ASC;


#Q10
select * from User ;
INSERT INTO USER (user_id, username, email, dob, gender, phoneno, address)
VALUES ('UN004','Mike','mike@yahoo.com','2002-01-29', 'M' , '9304195487' , "36th st manhatten");

SELECT user_id, username, gender FROM user WHERE email LIKE '%yahoo%' ORDER BY username DESC;
```

```sql
#Q11 Book Event
SET SQL_SAFE_UPDATES = 0;
select * from book ;
INSERT INTO BOOK (isbn, title, author, genre, publication_date, publisher, price)
VALUES ('000-5-16-148410-0', 'Sample Book5', 'Mike', 'Fiction', '2024-08-18', 'Sample Publisher 5', 400),
       ('000-6-16-148410-0', 'Sample Book6', 'Nobitta', 'Comedy', '30-08-24', 'Sample Publisher 6', 447.00);

SELECT isbn, title, author, price FROM BOOK WHERE price BETWEEN 350 AND 450 ORDER BY price DESC;

#Q12 Upcoming Fair
select * from book ;
UPDATE Book SET

    title = case
    when isbn = '978-3-16-148410-0' then 'Potter_C'
    when isbn = '000-3-16-148410-0' then 'Harry_C'
    when isbn = '000-2-16-148410-0' then 'Ramayana_C'
    when isbn = '000-1-16-148410-0' then 'clayBook_C'
    END,

    author = case
    when isbn = '978-3-16-148410-0' then 'Jane Tesla'
    end,

    price = case
    when isbn = '978-3-16-148410-0' then 190.00
    end

    where title in ('Potter' , 'ClayBook' , 'Ramayana', 'clayBook');

select * from book ;
SELECT isbn, title, author, publication_date FROM BOOK WHERE genre = 'Fiction' OR genre = 'Mystery' ORDER BY isbn DESC;
```

```sql
#Q13 Routine Checks
select * from book ;
select * from borrowing;
INSERT INTO borrowing (borrowing_id, user_id , isbn , borrow_date , due_date , return_date , fine) VALUES
('B1', 'UN002', '000-1-16-148410-0', '01-08-24', '10-08-24', '10-08-24', 0.00),
('1', 'UN003', '000-2-16-148410-0', '05-08-24', '15-08-24', '15-08-24', 0.00),
('2', 'UN004', '000-3-16-148410-0', '20-08-24', '25-08-24', '30-08-24', 500.00);
SELECT borrowing_id, user_id, borrow_date FROM borrowing WHERE due_date = return_date ORDER BY borrowing_id ASC;


#Q14 Security Measure
select * from user ;
SELECT
    user_id,
    username,
    CONCAT(SUBSTRING(username, 1, 3), SUBSTRING(phoneno, 1, 3)) AS password
FROM user
ORDER BY user_id DESC;

#Q15 May Birthday
select * from user ;
SELECT
    user_id,
    username,

    COALESCE(phoneno, email, 'Not Available') AS contact

FROM user

WHERE year(dob) = 1970
ORDER BY user_id ASC;
```

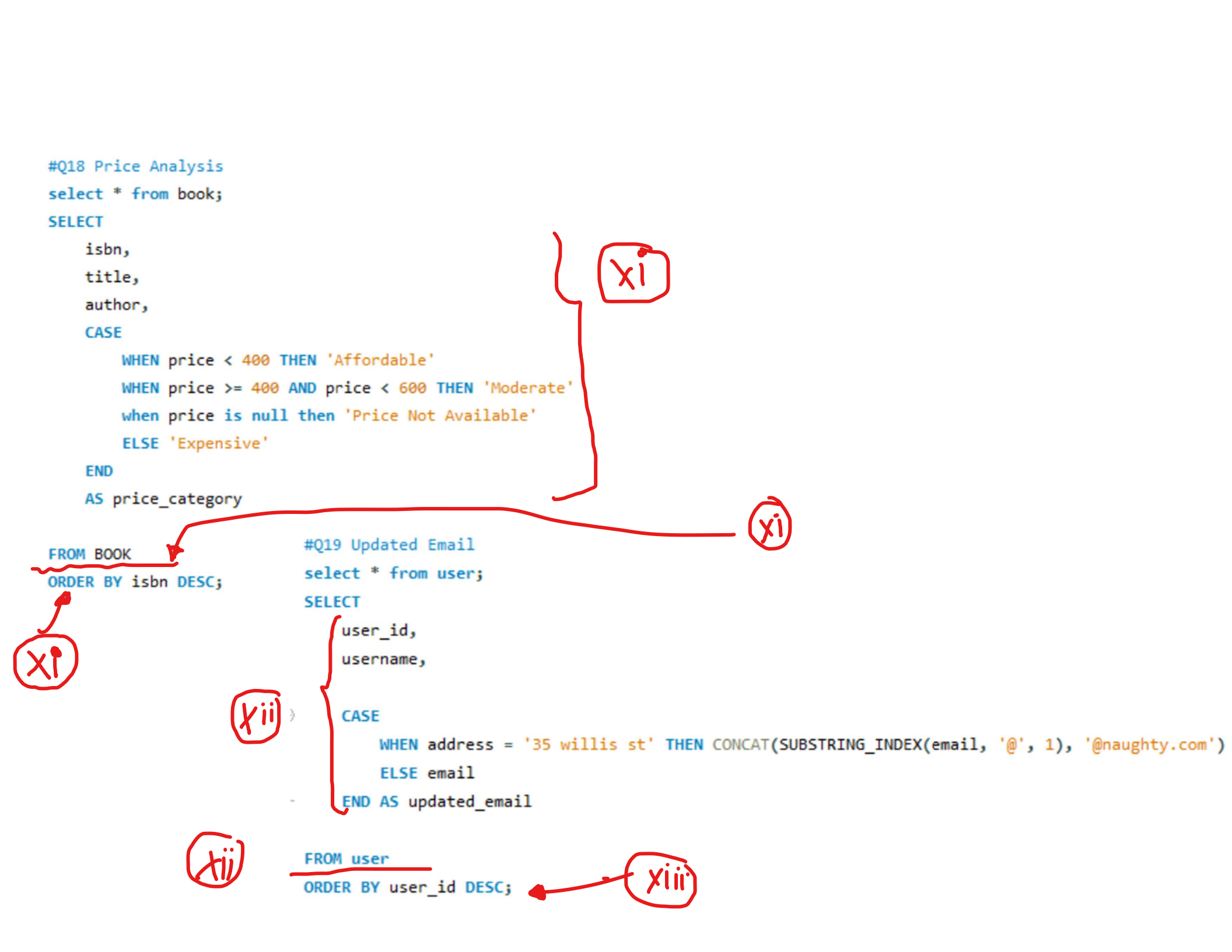


```sql
#Q16  Age Demographics
select * from user ;
SELECT
    user_id,
    username,
    email,

    ROUND(DATEDIFF(CURDATE(), dob) / 365.25) AS age_in_years

FROM user
WHERE ROUND(DATEDIFF(CURDATE(), dob) / 365.25) > 35
ORDER BY user_id DESC;
```

```sql
#Q17 Author Details
select * from Book ;
SELECT * ,
    CONCAT('Mr/Ms. ', COALESCE(author, genre, 'Not Available'), ' published a book on ', YEAR(publication_date)) AS book_info
FROM book
ORDER BY YEAR(publication_date) DESC;
```

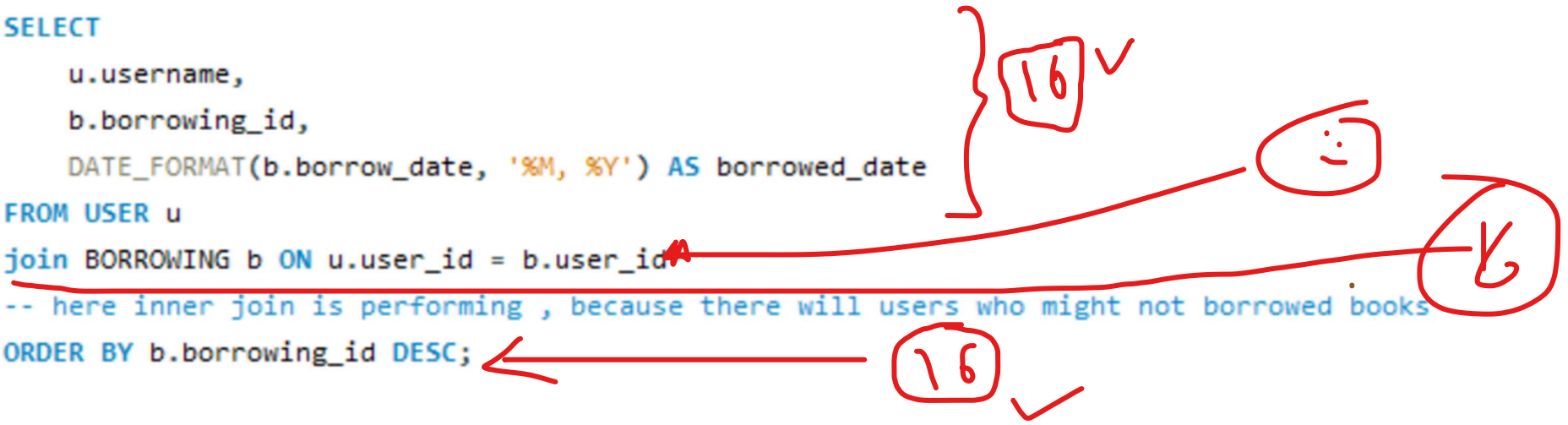
```sql
#Q18 Price Analysis
select * from book;
SELECT
    isbn,
    title,
    author,
    CASE
        WHEN price < 400 THEN 'Affordable'
        WHEN price >= 400 AND price < 600 THEN 'Moderate'
        when price is null then 'Price Not Available'
        ELSE 'Expensive'
    END
    AS price_category

FROM BOOK
ORDER BY isbn DESC;
```

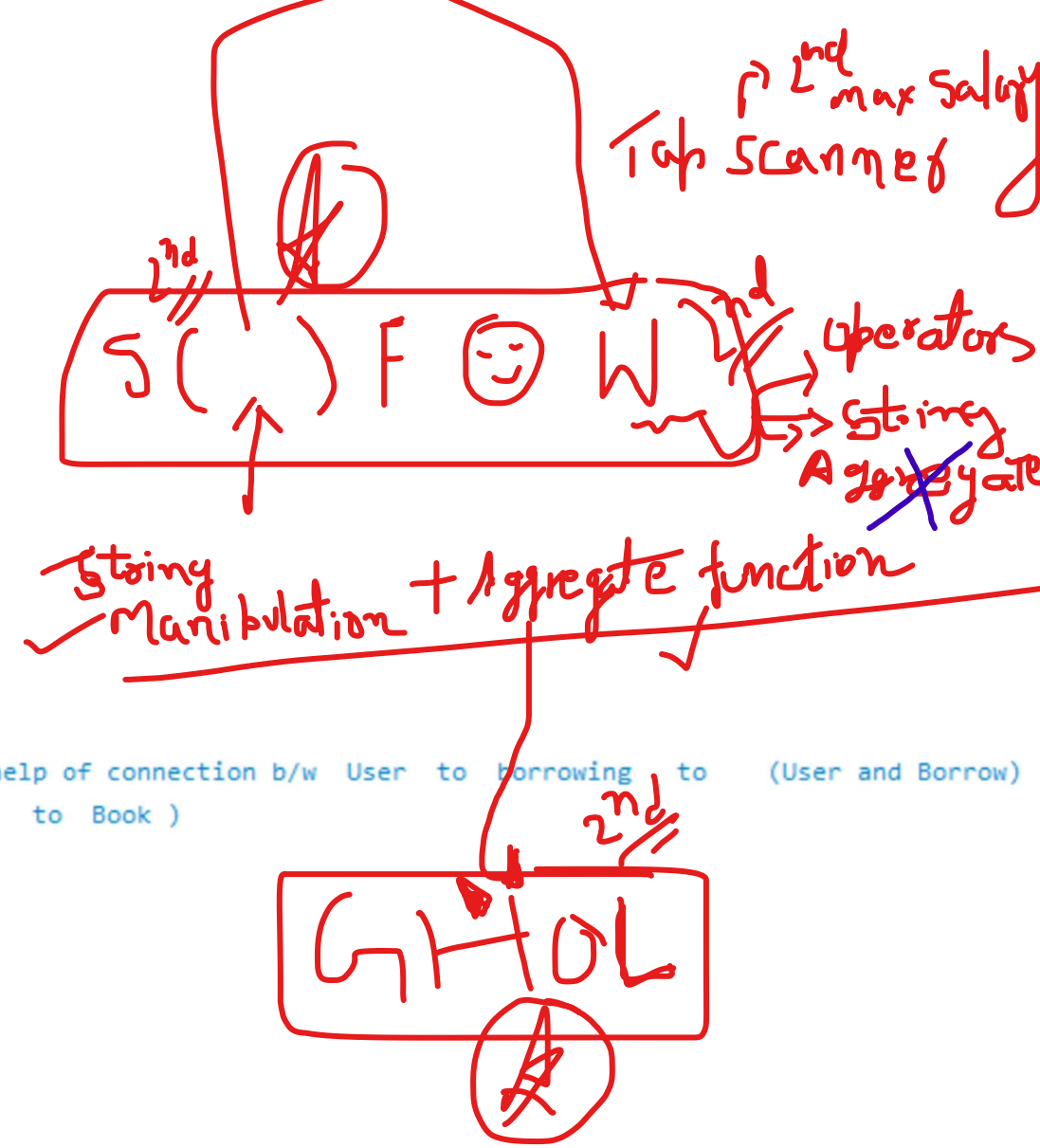
```sql
#Q19 Updated Email
select * from user;
SELECT
    user_id,
    username,

    CASE
        WHEN address = '35 willis st' THEN CONCAT(SUBSTRING_INDEX(email, '@', 1), '@naughty.com')
        ELSE email
    END AS updated_email

FROM user
ORDER BY user_id DESC;
```



```sql
#Q20 Lended Books
select * from borrowing;
SELECT
    user_id,
    COUNT(*) AS book_count

FROM BORROWING
WHERE YEAR(borrow_date) = 2024 -- trimming or filter clause
GROUP BY user_id               -- typically used with aggregate function. since user_id is repeated many times.
ORDER BY user_id ASC;

#Q21 The Chronicles of Authors and Their Tales
select * from book;
SELECT
    author,
    COUNT(*) AS total_books
FROM book
GROUP BY author
HAVING COUNT(*) >= 2 AND author is not Null
ORDER BY total_books DESC;
```

```sql
#Q22 Date Format
select * from user;
Alter table user modify column address varchar(100);
INSERT INTO USER (user_id, username, email, dob, gender, phoneno, address)
VALUES ('UN005','Ram','Ram@hoola.com','2002-01-17', 'M' , '7856148307' , "Bosten Main street"),
('UN006','Babita','Babita@play.com','2002-01-20', 'F' , '8561481234' , "Mumbai Thane Bank opp. street");

select * from borrowing;
-- alias  u.user  b.borrowing
SELECT
    u.username,
    b.borrowing_id,
    DATE_FORMAT(b.borrow_date, '%M, %Y') AS borrowed_date
FROM USER u
join BORROWING b ON u.user_id = b.user_id
-- here inner join is performing , because there will users who might not borrowed books
ORDER BY b.borrowing_id DESC;
```






```sql
#Q23 Exploring Users' Literary Journeys
select * from user;
select * from book;
select * from borrowing;

SELECT
    u.username,
    b.title,
    br.borrow_date
FROM User u
JOIN BORROWING br ON u.user_id = br.user_id     --  Taking help of connection b/w  User  to  borrowing  to    (User and Borrow)
JOIN BOOK b ON br.isbn = b.isbn                 --  (Borrow   to  Book )
ORDER BY br.borrow_date DESC;


#Q27 Maximum Fine Per User
select * from user;
select * from borrowing;
SELECT
    u.user_id,
    u.username,
    (SELECT MAX(fine) FROM BORROWING br WHERE br.user_id = u.user_id) AS max_fine  -- query for max fine for each user

FROM USER u
Order BY
    u.user_id DESC;  -- here we can also write " Order by max_fine desc "
```

```sql
#Q24 Active Borrowings with User and Book Details
select * from user;
select * from book;
select * from borrowing;

SELECT
    u.user_id,
    u.username,
    br.borrowing_id,
    br.isbn,
    b.title,
    br.borrow_date,
    br.due_date
FROM USER U
JOIN BORROWING br ON u.user_id = br.user_id     --  Taking help of connection b/w  User  to  borrowing   to    (User and Borrow)
JOIN BOOK b ON br.isbn = b.isbn                 --  (Borrow   to  Book )
WHERE br.return_date IS null
ORDER BY u.user_id DESC;
```

```sql
#Q25 Same Authors
UPDATE Book SET

    author = case
    when isbn = '000-1-16-148410-0' then 'Mike'
    when isbn = '000-2-16-148410-0' then 'Mike'
    when isbn = '000-3-16-148410-0' then 'Jane Tesla'
    when isbn = '978-3-16-148410-0' then 'Jane Tesla'
    END,

    genre = case
    when isbn = '000-1-16-148410-0' then 'Fiction'
    when isbn = '000-3-16-148410-0' then 'Romance'
    when isbn = '978-3-16-148410-0' then 'Romance'
    end,

    no_of_books = case
    when isbn = '000-1-16-148410-0' then 5
    end

    where isbn in ('000-1-16-148410-0', '000-3-16-148410-0','978-3-16-148410-0', '000-3-16-148410-0');
    -- always keep right hand side things
```

```sql
select * from book;

SELECT
    b1.title as title1,
    -- b2.title as title2,
    b1.author,
    b1.genre
FROM BOOK b1
JOIN
    BOOK b2 ON b1.author = b2.author -- same author , same genre , but title different
AND b1.genre = b2.genre
AND b1.title <> b2.title
ORDER BY author DESC;
```

```sql
#Q26 User Borrowing Summary
select * from user;
select * from borrowing;

SELECT
    u.user_id,
    u.username,
    COUNT(br.borrowing_id) AS Number_of_books

FROM
    USER u

LEFT JOIN
    BORROWING br ON u.user_id = br.user_id  -- ek user se two times same user id in borrowing  i.e two distinct borrowing_Id

GROUP BY -- group by is most present where " Aggregate fuction is used "
    u.user_id , u.username -- here it is mandatory to use one keyword which common in both tables (Borrowing , User) i.e user_id

ORDER BY
    u.user_id DESC;
```

```sql
#Q28 Top Users
select * from user;
select * from borrowing;
SELECT
    u.username,
    u.user_id
FROM USER u

Join BORROWING br ON u.user_id = br.user_id   -- It is Inner Join [ only the user who have borrowed at least one book ]
GROUP BY
    u.user_id, u.username

HAVING                  -- acts as "Where clause"   but " HAVING clause is used after grouping "
    COUNT(br.borrowing_id) = (SELECT MAX(borrow_count)
                              FROM (SELECT COUNT(borrowing_id) AS borrow_count -- counts the number of borrowing for each user
                                    FROM borrowing Group By user_id) As subquery);

/*
1st   count (br.borrowing_id)  --> counts the number of borrowing_id for each user (Not null) -> Think like a variable

2nd   SELECT COUNT(borrowing_id) AS borrow_count FROM borrowing Group By user_id  --> Because to remove duplicates User_id

3rd   select MAX(borrow_count) from  (....)  AS  subquery
*/
```

```sql
#Q29 User Count by Title and Author [  same as a specific user_id in borrowing table ]
select * from book;
select * from borrowing;
INSERT INTO borrowing (borrowing_id, user_id , isbn , borrow_date , due_date , return_date , fine) VALUES
('BR1', 'UN005', '978-3-16-148410-0', '01-08-05', '10-08-24', '10-08-24', 0.00),
('BR2', 'UN005', '978-3-16-148410-0', '05-08-14', '15-08-24', '15-08-24', 0.00),
('BR3', 'UN006', '978-3-16-148410-0', '20-08-10', '25-08-24', '30-08-24', 500.00);
SELECT
    b.title,
    b.author,
    br.user_id,

    COUNT(DISTINCT br.user_id) AS user_count
FROM
    BOOK b
Join
    BORROWING br ON b.isbn = br.isbn

WHERE -- we use where clause is used before grouping vvi.
    br.isbn IN ( SELECT br2.isbn FROM BORROWING br2 WHERE br2.user_id = 'UN001')

GROUP BY
    b.title, b.author , br.user_id ;
```