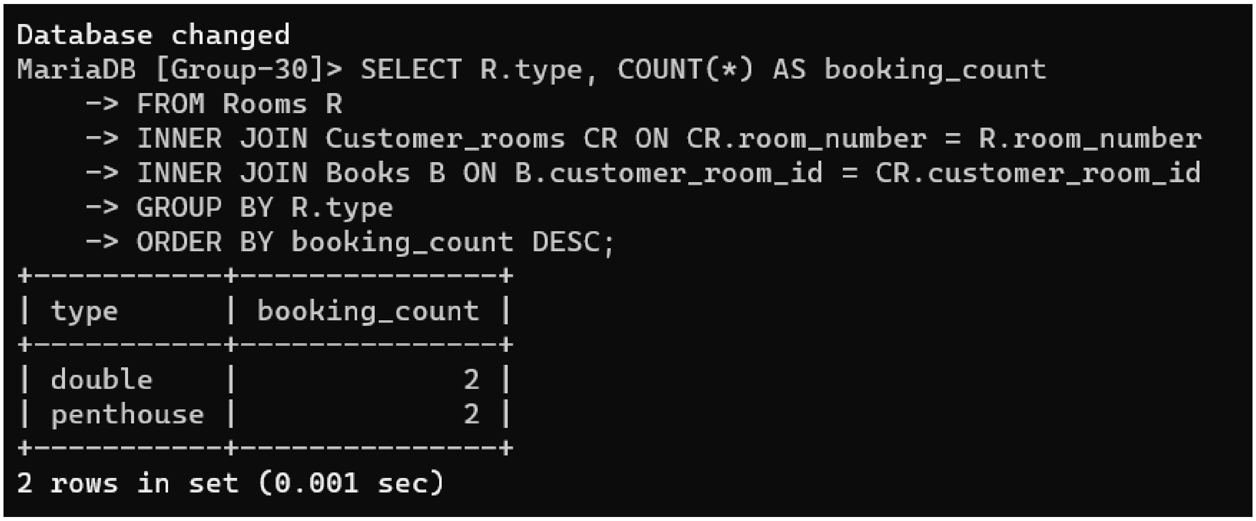
1. Group the customer rooms by type and count the number of times a certain type of room has been booked. This could be used as a way to see which rooms are preferred by the customers.

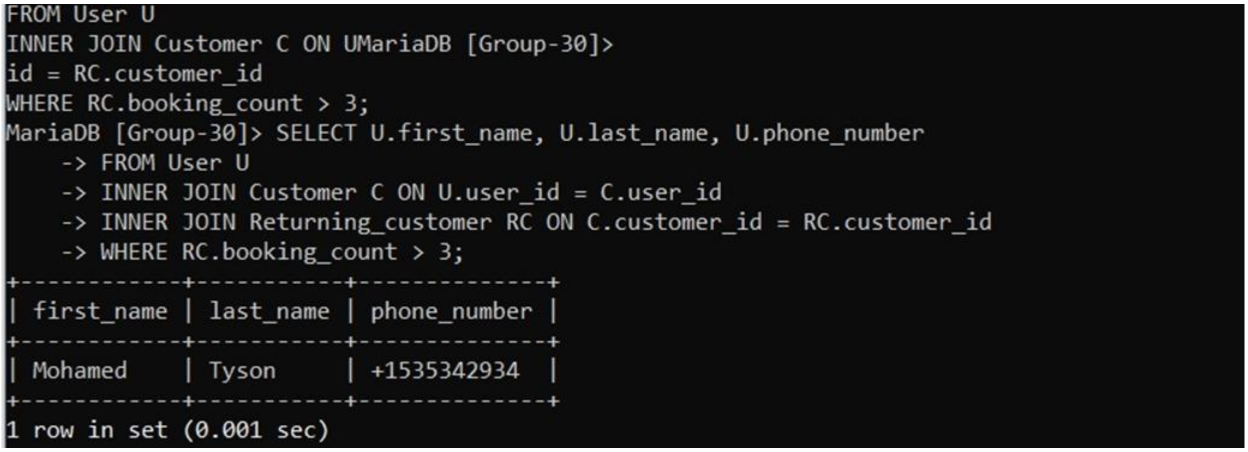
```
SELECT R.type, COUNT(*) AS booking_count
FROM Rooms R
INNER JOIN Customer_rooms CR ON CR.room_number = R.room_number
INNER JOIN Books B ON B.customer_room_id = CR.customer_room_id
GROUP BY R.type
HAVING Count(*) > 0
ORDER BY booking_count DESC;
```



```
Database changed
MariaDB [Group-30]> SELECT R.type, COUNT(*) AS booking_count
    -> FROM Rooms R
    -> INNER JOIN Customer_rooms CR ON CR.room_number = R.room_number
    -> INNER JOIN Books B ON B.customer_room_id = CR.customer_room_id
    -> GROUP BY R.type
    -> ORDER BY booking_count DESC;
+-----------+---------------+
| type      | booking_count |
+-----------+---------------+
| double    |             2 |
| penthouse |             2 |
+-----------+---------------+
2 rows in set (0.001 sec)
```

2. Fetch the first name, last name, and phone number of users who are registered customers and have made more than three bookings. This is done by joining the "User," "Customer," and "Returning_customer" tables and filtering for customers with a booking count greater than three.

```
SELECT U.first_name, U.last_name, U.phone_number
FROM User U
INNER JOIN Customer C ON U.user_id = C.user_id
INNER JOIN Returning_customer RC ON C.customer_id = RC.customer_id
WHERE RC.booking_count > 3;
```

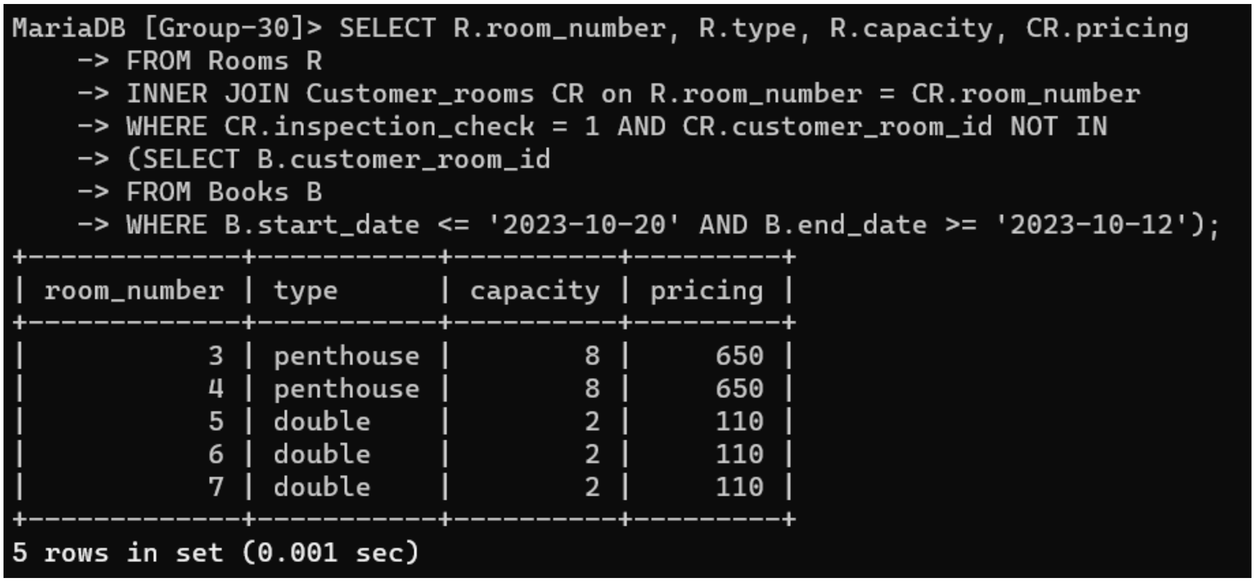


```
FROM User U
INNER JOIN Customer C ON UMariaDB [Group-30]>
id = RC.customer_id
WHERE RC.booking_count > 3;
MariaDB [Group-30]> SELECT U.first_name, U.last_name, U.phone_number
    -> FROM User U
    -> INNER JOIN Customer C ON U.user_id = C.user_id
    -> INNER JOIN Returning_customer RC ON C.customer_id = RC.customer_id
    -> WHERE RC.booking_count > 3;
+------------+-----------+--------------+
| first_name | last_name | phone_number |
+------------+-----------+--------------+
| Mohamed    | Tyson     | +1535342934  |
+------------+-----------+--------------+
1 row in set (0.001 sec)
```

3. Print information about the available rooms that a customer can book based on their check in - check out criteria and if the room has been inspected, meaning that it is usable. We can replace desired_check_in and desired_check_out with any dates (as in the example) and these will be used as variables that will be dynamically fetched from our web service.

```
SELECT R.room_number, R.type, R.capacity, CR.pricing
FROM Rooms R
INNER JOIN Customer_rooms CR on R.room_number = CR.room_number
WHERE CR.inspection_check = 1 AND CR.customer_room_id NOT IN
        (SELECT B.customer_room_id
         FROM Books B
         WHERE B.start_date <= :desired_check_in AND B.end_date >=
        :desired_check_out);
```

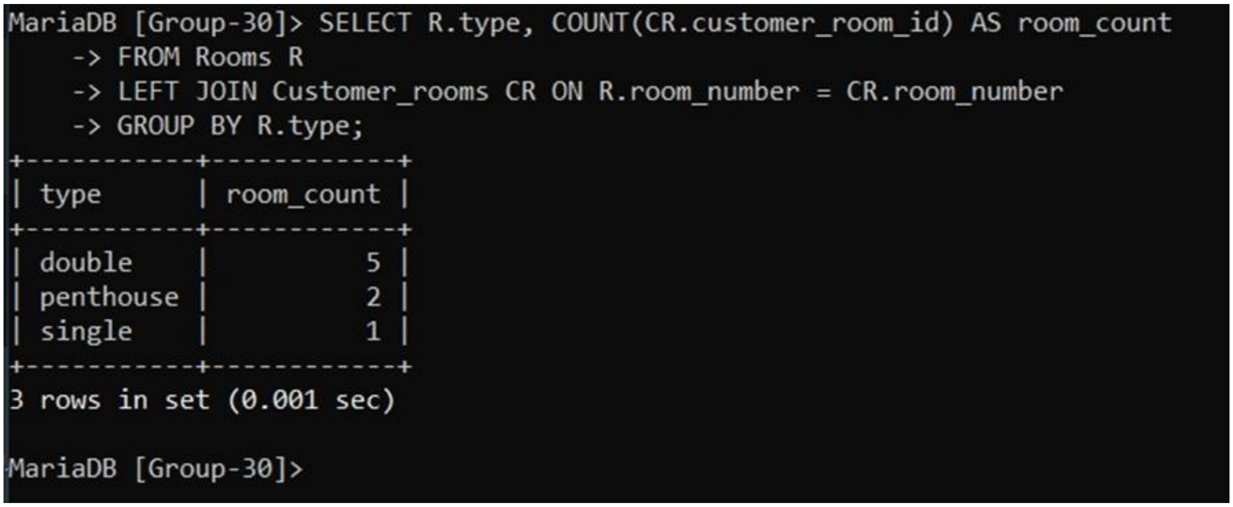
```
SELECT R.room_number, R.type, R.capacity, CR.pricing
FROM Rooms R
INNER JOIN Customer_rooms CR on R.room_number = CR.room_number
WHERE CR.inspection_check = 1 AND CR.customer_room_id NOT IN
        (SELECT B.customer_room_id
         FROM Books B
         WHERE B.start_date <= '2023-10-20' AND B.end_date >= '2023-10-12');
```



```
MariaDB [Group-30]> SELECT R.room_number, R.type, R.capacity, CR.pricing
    -> FROM Rooms R
    -> INNER JOIN Customer_rooms CR on R.room_number = CR.room_number
    -> WHERE CR.inspection_check = 1 AND CR.customer_room_id NOT IN
    -> (SELECT B.customer_room_id
    -> FROM Books B
    -> WHERE B.start_date <= '2023-10-20' AND B.end_date >= '2023-10-12');
+-------------+-----------+----------+---------+
| room_number | type      | capacity | pricing |
+-------------+-----------+----------+---------+
|           3 | penthouse |        8 |     650 |
|           4 | penthouse |        8 |     650 |
|           5 | double    |        2 |     110 |
|           6 | double    |        2 |     110 |
|           7 | double    |        2 |     110 |
+-------------+-----------+----------+---------+
5 rows in set (0.001 sec)
```

4. Retrieve the room types along with the count of customers assigned to each room type. Include room types even if they have no customers, using a left join between the "Rooms" and "Customer_rooms" tables. Group the results by room type.

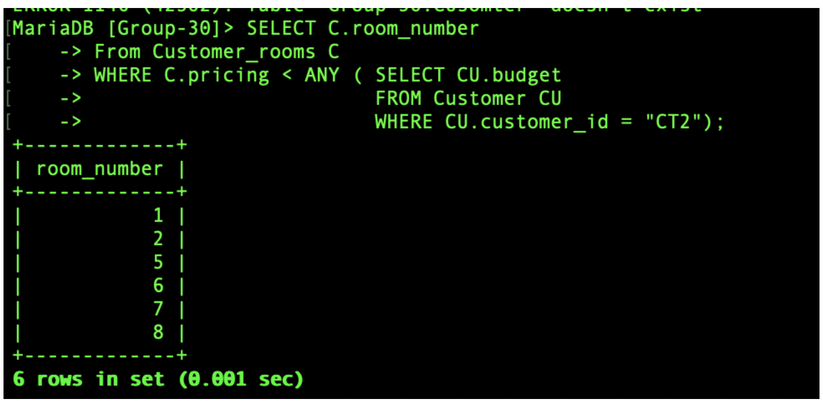
```
SELECT R.type, COUNT(CR.customer_room_id) AS room_count
FROM Rooms R
LEFT JOIN Customer_rooms CR ON R.room_number = CR.room_number
GROUP BY R.type;
```



```
MariaDB [Group-30]> SELECT R.type, COUNT(CR.customer_room_id) AS room_count
    -> FROM Rooms R
    -> LEFT JOIN Customer_rooms CR ON R.room_number = CR.room_number
    -> GROUP BY R.type;
+-----------+------------+
| type      | room_count |
+-----------+------------+
| double    |          5 |
| penthouse |          2 |
| single    |          1 |
+-----------+------------+
3 rows in set (0.001 sec)

MariaDB [Group-30]>
```

5. Retrieve the room numbers from the "Customer_rooms" table where the pricing is less than the budget of at least one customer with the customer ID "CT1" in the "Customer" table.

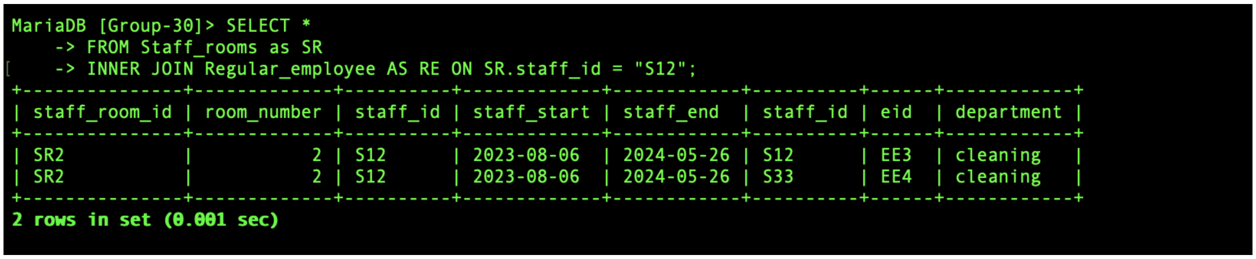
```
SELECT C.room_number
FROM Customer_rooms C
WHERE C.pricing < ANY ( SELECT CU.budget
                        FROM Customer CU
                        WHERE CU.customer_id = desired_customer_id);
SELECT C.room_number
FROM Customer_rooms C
WHERE C.pricing < ANY ( SELECT CU.budget
                        FROM Customer CU
                        WHERE CU.customer_id = “CT1”);
```



```
ERROR 1146 (42S02): Table 'Group-30.customer' doesn't exist
[MariaDB [Group-30]> SELECT C.room_number
[    -> From Customer_rooms C
[    -> WHERE C.pricing < ANY ( SELECT CU.budget
[    ->                         FROM Customer CU
[    ->                         WHERE CU.customer_id = "CT2");
+-------------+
| room_number |
+-------------+
|           1 |
|           2 |
|           5 |
|           6 |
|           7 |
|           8 |
+-------------+
6 rows in set (0.001 sec)
```

6. Fetch all columns from the "Staff_Rooms" table, joining it with the "Regular_Employee" table on the common column "staff_id."

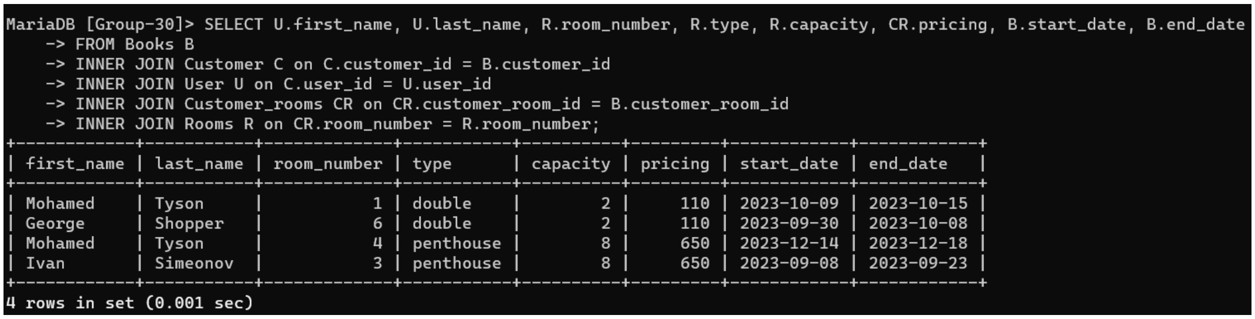
```
SELECT *
FROM Staff_rooms as SR
INNER JOIN Regular_employee AS RE ON SR.staff_id = desired_staff_id;

SELECT *
FROM Staff_rooms as SR
INNER JOIN Regular_employee AS RE ON SR.staff_id =  “S12”;
```



```
MariaDB [Group-30]> SELECT *
    -> FROM Staff_rooms as SR
[    -> INNER JOIN Regular_employee AS RE ON SR.staff_id = "S12";
+---------------+-------------+----------+-------------+------------+----------+------+------------+
| staff_room_id | room_number | staff_id | staff_start | staff_end  | staff_id | eid  | department |
+---------------+-------------+----------+-------------+------------+----------+------+------------+
| SR2           |           2 | S12      | 2023-08-06  | 2024-05-26 | S12      | EE3  | cleaning   |
| SR2           |           2 | S12      | 2023-08-06  | 2024-05-26 | S33      | EE4  | cleaning   |
+---------------+-------------+----------+-------------+------------+----------+------+------------+
2 rows in set (0.001 sec)
```

7. Using the relationship set Books we can get more information about the reservations which have been made and produce a summary. This can be used by admin to gather data for analysis and keep track of the reservations. - G

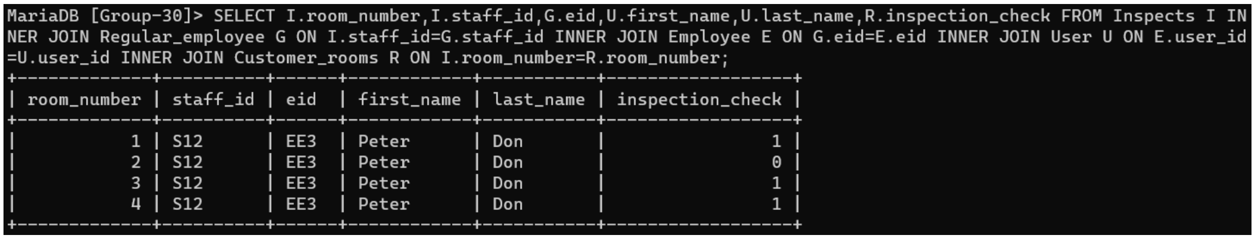
```
SELECT U.first_name, U.last_name, R.room_number, R.type, R.capacity, CR.pricing,
B.start_date, B.end_date
FROM Books B
INNER JOIN Customer C on C.customer_id = B.customer_id
INNER JOIN User U on C.user_id = U.user_id
INNER JOIN Customer_rooms CR on CR.customer_room_id = B.customer_room_id
INNER JOIN Rooms R on CR.room_number = R.room_number;
```



```
MariaDB [Group-30]> SELECT U.first_name, U.last_name, R.room_number, R.type, R.capacity, CR.pricing, B.start_date, B.end_date
    -> FROM Books B
    -> INNER JOIN Customer C on C.customer_id = B.customer_id
    -> INNER JOIN User U on C.user_id = U.user_id
    -> INNER JOIN Customer_rooms CR on CR.customer_room_id = B.customer_room_id
    -> INNER JOIN Rooms R on CR.room_number = R.room_number;
+------------+-----------+-------------+-----------+----------+---------+------------+------------+
| first_name | last_name | room_number | type      | capacity | pricing | start_date | end_date   |
+------------+-----------+-------------+-----------+----------+---------+------------+------------+
| Mohamed    | Tyson     |           1 | double    |        2 |     110 | 2023-10-09 | 2023-10-15 |
| George     | Shopper   |           6 | double    |        2 |     110 | 2023-09-30 | 2023-10-08 |
| Mohamed    | Tyson     |           4 | penthouse |        8 |     650 | 2023-12-14 | 2023-12-18 |
| Ivan       | Simeonov  |           3 | penthouse |        8 |     650 | 2023-09-08 | 2023-09-23 |
+------------+-----------+-------------+-----------+----------+---------+------------+------------+
4 rows in set (0.001 sec)
```

8. Get information about which employee has inspected or needs to inspect which room.

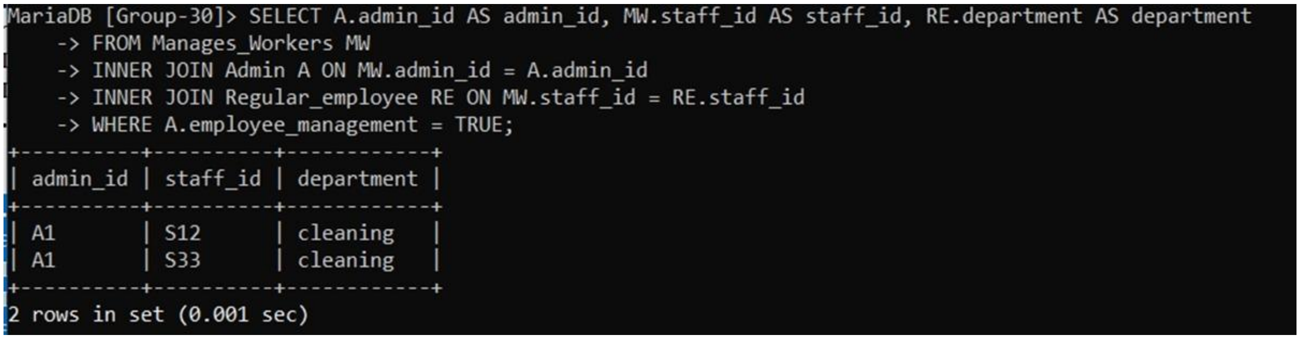
```
SELECT I.room_number,I.staff_id,G.eid,U.first_name,U.last_name,R.inspection_check
FROM Inspects I
INNER JOIN Regular_employee G ON I.staff_id=G.staff_id
INNER JOIN Employee E ON G.eid=E.eid
INNER JOIN User U ON E.user_id=U.user_id
INNER JOIN Customer_rooms R ON I.room_number=R.room_number;
```



```
MariaDB [Group-30]> SELECT I.room_number,I.staff_id,G.eid,U.first_name,U.last_name,R.inspection_check FROM Inspects I IN
NER JOIN Regular_employee G ON I.staff_id=G.staff_id INNER JOIN Employee E ON G.eid=E.eid INNER JOIN User U ON E.user_id
=U.user_id INNER JOIN Customer_rooms R ON I.room_number=R.room_number;
+-------------+----------+------+------------+-----------+------------------+
| room_number | staff_id | eid  | first_name | last_name | inspection_check |
+-------------+----------+------+------------+-----------+------------------+
|           1 | S12      | EE3  | Peter      | Don       |                1 |
|           2 | S12      | EE3  | Peter      | Don       |                0 |
|           3 | S12      | EE3  | Peter      | Don       |                1 |
|           4 | S12      | EE3  | Peter      | Don       |                1 |
+-------------+----------+------+------------+-----------+------------------+
```

9. Obtain a list of admin IDs, staff IDs, and departments for employees who manage workers. To do this, perform an inner join between the "Manages_Workers," "Admin," and "Regular_employee" tables, and only include rows where the "employee_management" flag in the "Admin" table is set to TRUE.

```
SELECT A.admin_id AS admin_id, MW.staff_id AS staff_id, RE.department AS
department
FROM Manages_Workers MW
INNER JOIN Admin A ON MW.admin_id = A.admin_id
INNER JOIN Regular_employee RE ON MW.staff_id = RE.staff_id
WHERE A.employee_management = TRUE;
```

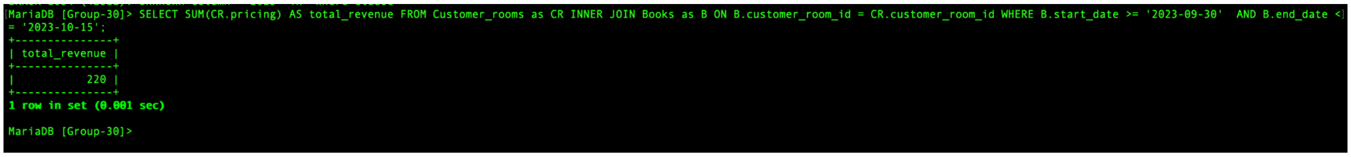


```
MariaDB [Group-30]> SELECT A.admin_id AS admin_id, MW.staff_id AS staff_id, RE.department AS department
    -> FROM Manages_Workers MW
    -> INNER JOIN Admin A ON MW.admin_id = A.admin_id
    -> INNER JOIN Regular_employee RE ON MW.staff_id = RE.staff_id
    -> WHERE A.employee_management = TRUE;
+----------+----------+------------+
| admin_id | staff_id | department |
+----------+----------+------------+
| A1       | S12      | cleaning   |
| A1       | S33      | cleaning   |
+----------+----------+------------+
2 rows in set (0.001 sec)
```

10. Calculate the total revenue by summing up the pricing of customer rooms that are associated with bookings made between specific dates, as specified in the "Customer_rooms" and "Books" tables.

```
SELECT SUM(CR.pricing) AS total_revenue
FROM Customer_rooms as CR
INNER JOIN Books as B ON B.customer_room_id = CR.customer_room_id
WHERE B.start_date >= desired_start_date
AND B.end_date <= desired_end_date;
```

```
SELECT SUM(CR.pricing) AS total_revenue
FROM Customer_rooms as CR
INNER JOIN Books as B ON B.customer_room_id = CR.customer_room_id
WHERE B.start_date >= ‘2023-09-30’
AND B.end_date <= ‘2023-10-15’ ;
```

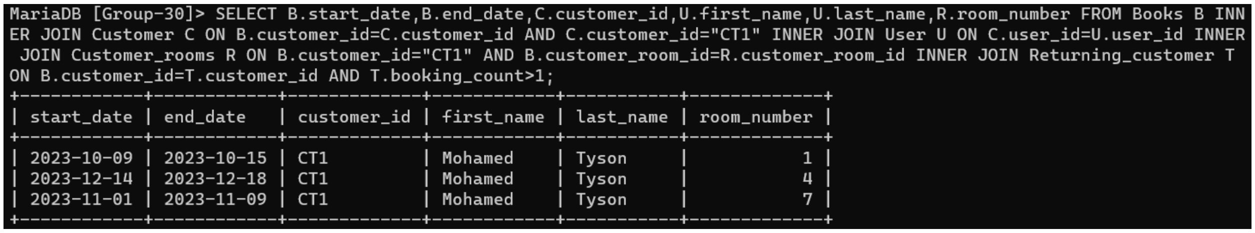


```
MariaDB [Group-30]> SELECT SUM(CR.pricing) AS total_revenue FROM Customer_rooms as CR INNER JOIN Books as B ON B.customer_room_id = CR.customer_room_id WHERE B.start_date >= '2023-09-30'  AND B.end_date <
= '2023-10-15';
+---------------+
| total_revenue |
+---------------+
|           220 |
+---------------+
1 row in set (0.001 sec)

MariaDB [Group-30]>
```

11. Print a booking history for a specific customer, that has booked more than once.

```
SELECT
B.start_date,B.end_date,C.customer_id,U.first_name,U.last_name,R.room_number
FROM Books B
INNER JOIN Customer C ON B.customer_id=C.customer_id AND
C.customer_id=:desired_customer_id
```

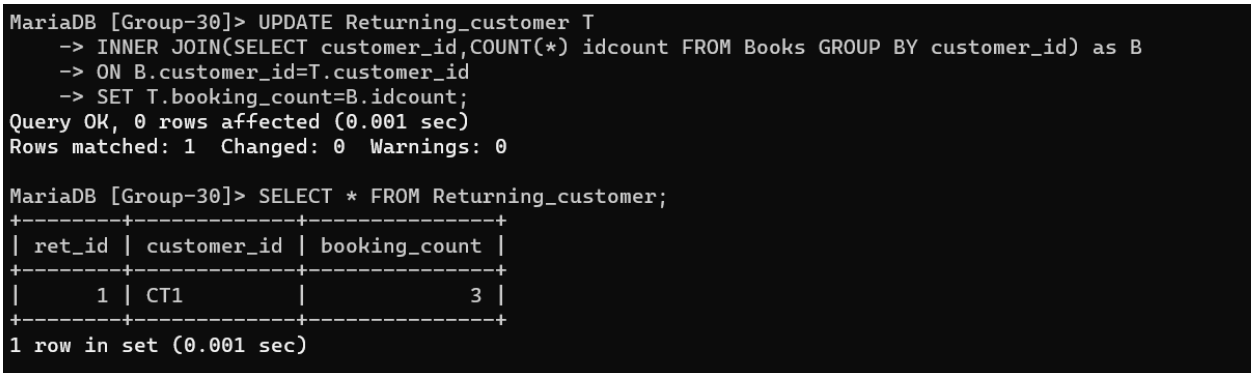
INNER JOIN User U ON C.user_id=U.user_id
INNER JOIN Customer_rooms R ON B.customer_id=:desired_customer_id AND B.customer_room_id=R.customer_room_id
INNER JOIN Returning_customer T ON B.customer_id=T.customer_id AND T.booking_count>1;



```
MariaDB [Group-30]> SELECT B.start_date,B.end_date,C.customer_id,U.first_name,U.last_name,R.room_number FROM Books B INN
ER JOIN Customer C ON B.customer_id=C.customer_id AND C.customer_id="CT1" INNER JOIN User U ON C.user_id=U.user_id INNER
 JOIN Customer_rooms R ON B.customer_id="CT1" AND B.customer_room_id=R.customer_room_id INNER JOIN Returning_customer T
ON B.customer_id=T.customer_id AND T.booking_count>1;
+------------+------------+-------------+------------+-----------+-------------+
| start_date | end_date   | customer_id | first_name | last_name | room_number |
+------------+------------+-------------+------------+-----------+-------------+
| 2023-10-09 | 2023-10-15 | CT1         | Mohamed    | Tyson     |           1 |
| 2023-12-14 | 2023-12-18 | CT1         | Mohamed    | Tyson     |           4 |
| 2023-11-01 | 2023-11-09 | CT1         | Mohamed    | Tyson     |           7 |
+------------+------------+-------------+------------+-----------+-------------+
```

12. Update the booking count for returning customer, depending on how many bookings they have done.

UPDATE Returning_customer T
INNER JOIN(SELECT customer_id,COUNT(*) idcount FROM Books GROUP BY customer_id) as B
ON B.customer_id=T.customer_id
SET T.booking_count=B.idcount;



```
MariaDB [Group-30]> UPDATE Returning_customer T
    -> INNER JOIN(SELECT customer_id,COUNT(*) idcount FROM Books GROUP BY customer_id) as B
    -> ON B.customer_id=T.customer_id
    -> SET T.booking_count=B.idcount;
Query OK, 0 rows affected (0.001 sec)
Rows matched: 1  Changed: 0  Warnings: 0

MariaDB [Group-30]> SELECT * FROM Returning_customer;
+--------+-------------+---------------+
| ret_id | customer_id | booking_count |
+--------+-------------+---------------+
|      1 | CT1         |             3 |
+--------+-------------+---------------+
1 row in set (0.001 sec)
```