MIT World Peace University

Database Management System

Assignment 5

Naman Soni Roll No. 10

Contents

1 Aim

Write suitable select commands to execute queries on the given data set.

2 Objectives

1. To get basic understanding of Aggregate Functions, Order By clause
2. To get basic understanding of Subquery or Inner query or Nested query and Select using subquery.
3. To understand the basic concept of Correlated Subquery.
4. To get familiar with the basic ALL, ANY, EXISTS, SOME functionality.
5. To understand basic TCL commands

3 Problem Statement

Create tables and solve given queries using Subqueries

4 Theory

4.1 Aggregate Functions

Aggregate functions are used to perform calculations on a set of values and return a single value. The following are the aggregate functions:

- COUNT() - Returns the number of rows that matches a specified criteria
- SUM() - Returns the sum of all the values in a column
- AVG() - Returns the average value of a numeric column
- MIN() - Returns the smallest value of the selected column
- MAX() - Returns the largest value of the selected column

4.2 Order By Clause

The ORDER BY clause is used to sort the result-set in ascending or descending order. The ORDER BY keyword sorts the records in ascending order by default. To sort the records in descending order, use the DESC keyword.

Syntax

```sql
SELECT column_name(s)
FROM table_name
ORDER BY column_name(s) ASC|DESC;
```

4.3 Group By Clause

The GROUP BY clause is often used with aggregate functions (COUNT, MAX, MIN, SUM, AVG) to group the result-set by one or more columns.

Syntax

```sql
SELECT column_name, aggregate_function(column_name)
FROM table_name
WHERE column_name operator value
GROUP BY column_name;
```

4.4 Subqueries

A subquery (sub-select) is a query within a query. The subquery is executed first, and the main query uses the subquery as a source of data.

Syntax

```sql
SELECT column_name(s)
FROM table_name
WHERE column_name operator
(SELECT STATEMENT);
```

4.5 Views

A view is a virtual table based on the result-set of an SQL statement.

Syntax

```sql
CREATE VIEW view_name AS
SELECT column_name(s)
FROM table_name
WHERE condition;
```

4.6 TCL Commands

TCL stands for Transaction Control Language. It is a set of SQL commands that are used to control the transaction. The following are the TCL commands:

- COMMIT - permanently saves all changes made by the transaction.
- ROLLBACK - cancels all changes made by the transaction
- SAVEPOINT - sets a savepoint within a transaction
- SET TRANSACTION - sets the transaction characteristics for the current transaction

5 Platform

Operating System: Arch Linux x86-64
IDEs or Text Editors Used: Draw.io for Drawing the ER diagram.

6 Input

Given Database from the Problem Statement for the Assignment for our batch. (A1 PA 20)

7 Creation and Insertion of Values in the Tables

```sql
-- Active: 1678946907415@@127.0.0.1@3306@dbms_lab
use dbms_lab;

show tables;

create table airline(name varchar(50) primary key);

create table airplane(reg_no int primary key, model_no int, capacity int, name
    varchar(50), foreign key(name) references airline(name));
create table flights(flight_no int primary key, place_from varchar(50), place_to
    varchar(50), departure_date date, departure_time time, arrival_date date,
    arrival_time time, reg_no int, foreign key(reg_no) references airplane(reg_no))
    ;

create table passenger(email varchar(50) primary key, first_name varchar(50),
    surname varchar(50));
create table flight_booking(email varchar(50), flight_no int, no_seats int,
    foreign key(email) references passenger(email), foreign key(flight_no)
    references flights(flight_no), primary key(email, flight_no));

describe airplane;

insert into airline values("Qatar Airways");
insert into airline values("Emirates");
insert into airline values("Air India");

insert into airplane values(111,007,180,"Qatar Airways");
insert into airplane values(112,007,169,"Qatar Airways");
insert into airplane values(113,008,200,"Qatar Airways");
insert into airplane values(221,017,150,"Emirates");
insert into airplane values(222,017,140,"Emirates");
insert into airplane values(223,018,175,"Emirates");
insert into airplane values(333,027,200,"Air India");
insert into airplane values(334,027,150,"Air India");
insert into airplane values(335,028,175,"Air India");

select * from airplane;

describe flights;
insert into flights values(12345,"Mumbai","London","2021-07-27","12:12:12","
    2021-07-28","23:59:56",111);
insert into flights values(67890,"Pune","Bangalore","2021-07-27","12:12:12","
    2021-07-27","16:59:56",221);
```

```sql
insert into flights values(23456,"London","Pune","2021-07-27","12:12:12","2021-07-28","22:59:56",333);

select * from flights;

describe passenger;

insert into passenger values("love@gmail.com","Love","Quinn");
insert into passenger values("joe@gmail.com","Joe","Goldberg");
insert into passenger values("beck@gmail.com","Gwen","Beck");

describe flight_booking;

insert into flight_booking values("love@gmail.com",12345,6);
insert into flight_booking values("joe@gmail.com",23456,2);
insert into flight_booking values("beck@gmail.com",67890,6);

select * from flight_booking;
select * from flights;

-- QUERIES

-- 1. Display the Passenger email ,Flight_no,Source and Destination Airport Names for all flights
-- booked

select b.email, b.flight_no, f.place_from, f.place_to from flight_booking as b inner join flights as f where b.flight_no = f.flight_no;

-- 2.
--  Display the flight and passenger details for the flights booked having Departure Date between
-- 23-08-2021 and 25-08-2021

select * from flights as f, passenger as p, flight_booking as b where b.email = p.email and b.flight_no = f.flight_no and departure_date between "2021-07-27" and "2021-07-28";

-- 3.
--  Display the top 5 airplanes that participated in Flights from Mumbai to London based on the
-- airplane capacity

select * from airplane as a, flights as f where a.reg_no = f.reg_no and f.place_from = "Mumbai" and f.place_to = "London" order by a.capacity desc limit 5;

-- 4.Display the passenger first names who have booked the no_of seats smaller than the average
-- number of seats booked by all passengers for the arrival airport:New Delhi

select * from passenger as p, flight_booking as b, flights as f where p.email = b.email and f.flight_no = b.flight_no and f.place_to = "New Delhi" and b.no_seats < all(select avg(no_seats) from flight_booking);


/*5.Display the surnames of passengers who have not booked a flight from Pune to Bangalore*/
select surname
```

```
from passenger
where email not in(
        select email
        from flight_booking
        where flight_no in (
                select flight_no
                from flights
                where place_from = 'Pune'
                    and place_to = 'Bangalore'
            )
    );

/*6. Display the Passenger details only if they have booked flights on 21st July
    2021. (Use Exists)*/
select *
from passenger
where exists (
        select email
        from flight_booking
        where flight_no in(
                select flight_no
                from flights
                where departure_date = '2021-07-27'
            )
    );
/*--7.Display the Flight-wise total time duration of flights if the duration is
    more than 8 hours (Hint : Date function,Aggregation,Grouping)*/

select flight_no, timediff(f.arrival_time, f.departure_time ) from flights as f
    where timediff(f.arrival_time, f.departure_time ) > "8:00:00" group by
    flight_no;

/*8.Display the Airplane-wise average seating capacity for any airline*/
select name,
    avg(capacity)
from airplane
group by name;

/*9.Display the total number of flights which are booked and travelling to London
    airport.*/
select count(b.flight_no) as total
from flight_booking b,
    flights f
where f.place_to = 'London';

/*10. Create a view having information about flight_no,airplane_no,capacity.*/
create view flightinfo as
select f.flight_no,
    a.reg_no,
    a.capacity
from flights f,
    airplane a
where a.reg_no = f.reg_no;

select * from flightinfo;
```

8 Tables

```
MariaDB [dbms_lab]> select * from passenger;
+----------------+------------+----------+
| email          | first_name | surname  |
+----------------+------------+----------+
| beck@gmail.com | Gwen       | Beck     |
| joe@gmail.com  | Joe        | Goldberg |
| love@gmail.com | Love       | Quinn    |
+----------------+------------+----------+
3 rows in set (0.001 sec)

MariaDB [dbms_lab]> select * from airplane;
+--------+----------+----------+---------------+
| reg_no | model_no | capacity | name          |
+--------+----------+----------+---------------+
|    111 |        7 |      180 | Qatar Airways |
|    112 |        7 |      169 | Qatar Airways |
|    113 |        8 |      200 | Qatar Airways |
|    221 |       17 |      150 | Emirates      |
|    222 |       17 |      140 | Emirates      |
|    223 |       18 |      175 | Emirates      |
|    333 |       27 |      200 | Air India     |
|    334 |       27 |      150 | Air India     |
|    335 |       28 |      175 | Air India     |
+--------+----------+----------+---------------+
9 rows in set (0.001 sec)

MariaDB [dbms_lab]> select * from airline;
+---------------+
| name          |
+---------------+
| Air India     |
| Emirates      |
| Qatar Airways |
+---------------+
3 rows in set (0.001 sec)

MariaDB [dbms_lab]> select * from flights;
+--
   ---------+------------+-----------+----------------+----------------+--------------+--------
| flight_no | place_from | place_to  | departure_date | departure_time |
   arrival_date | arrival_time | reg_no |
+--
   ---------+------------+-----------+----------------+----------------+--------------+--------
|     12345 | Mumbai     | London    | 2021-07-27     | 12:12:12       |
   2021-07-28   | 23:59:56     |    111 |
|     23456 | London     | Pune      | 2021-07-27     | 12:12:12       |
   2021-07-28   | 22:59:56     |    333 |
|     67890 | Pune       | Bangalore | 2021-07-27     | 12:12:12       |
   2021-07-27   | 16:59:56     |    221 |
+--
   ---------+------------+-----------+----------------+----------------+--------------+--------
3 rows in set (0.001 sec)

MariaDB [dbms_lab]> select * from flight_booking;
+----------------+-----------+----------+
| email          | flight_no | no_seats |
+----------------+-----------+----------+
| beck@gmail.com |     67890 |        6 |
| joe@gmail.com  |     23456 |        2 |
| love@gmail.com |     12345 |        6 |
+----------------+-----------+----------+
3 rows in set (0.001 sec)
```

9 Queries

```
MariaDB [dbms_lab]> -- QUERIES
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> -- 1. Display the Passenger email ,Flight_no,Source and Destination Airport Names for all flights
MariaDB [dbms_lab]> -- booked
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select b.email, b.flight_no, f.place_from, f.place_to from flight_booking as b inner join flights as f where b.flight_no = f.flight_no;
+----------------+-----------+------------+-----------+
| email          | flight_no | place_from | place_to  |
+----------------+-----------+------------+-----------+
| love@gmail.com |     12345 | Mumbai     | London    |
| joe@gmail.com  |     23456 | London     | Pune      |
| beck@gmail.com |     67890 | Pune       | Bangalore |
+----------------+-----------+------------+-----------+
3 rows in set (0.001 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> -- 2.
MariaDB [dbms_lab]> --  Display the flight and passenger details for the flights booked having Departure Date between
MariaDB [dbms_lab]> -- 23-08-2021 and 25-08-2021
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select * from flights as f, passenger as p, flight_booking as b where b.email = p.email and b.flight_no = f.flight_no and departure_date between "2021-07-27" and "2021-07-28";
+-----------+------------+-----------+----------------+----------------+--------------+--------
| flight_no | place_from | place_to  | departure_date | departure_time | arrival_date | arrival_time | reg_no | email          | first_name | surname  | email          | flight_no | no_seats |
+-----------+------------+-----------+----------------+----------------+--------------+--------
|     67890 | Pune       | Bangalore | 2021-07-27     | 12:12:12       | 2021-07-27   | 16:59:56     |    221 | beck@gmail.com | Gwen       | Beck     | beck@gmail.com |     67890 |        6 |
|     23456 | London     | Pune      | 2021-07-27     | 12:12:12       | 2021-07-28   | 22:59:56     |    333 | joe@gmail.com  | Joe        | Goldberg | joe@gmail.com  |     23456 |        2 |
|     12345 | Mumbai     | London    | 2021-07-27     | 12:12:12       | 2021-07-28   | 23:59:56     |    111 | love@gmail.com | Love       | Quinn    | love@gmail.com |     12345 |        6 |
+-----------+------------+-----------+----------------+----------------+--------------+--------
3 rows in set (0.004 sec)
```

```

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> -- 3.
MariaDB [dbms_lab]> --  Display the top 5 airplanes that participated in Flights from Mumbai to London based on the
MariaDB [dbms_lab]> -- airplane capacity
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select * from airplane as a, flights as f where a.reg_no = f.reg_no and f.place_from = "Mumbai" and f.place_to = "London" order by a.capacity desc limit 5;
+--------+----------+----------+---------------+-----------+------------+----------+-----------
| reg_no | model_no | capacity | name          | flight_no | place_from | place_to | departure_date | departure_time | arrival_date | arrival_time | reg_no |
+--------+----------+----------+---------------+-----------+------------+----------+-----------
|    111 |        7 |      180 | Qatar Airways |     12345 | Mumbai     | London   | 2021-07-27     | 12:12:12       | 2021-07-28   | 23:59:56     |    111 |
+--------+----------+----------+---------------+-----------+------------+----------+-----------
1 row in set (0.002 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> -- 4.Display the passenger first names who have booked the no_of seats smaller than the average
MariaDB [dbms_lab]> -- number of seats booked by all passengers for the arrival airport:New Delhi
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select * from passenger as p, flight_booking as b, flights as f where p.email = b.email and f.flight_no = b.flight_no and f.place_to = "New Delhi" and b.no_seats < all(select avg(no_seats) from flight_booking);
Empty set (0.002 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> /*5.Display the surnames of passengers who have not booked a flight from Pune to Bangalore*/
MariaDB [dbms_lab]> select surname
    -> from passenger
    -> where email not in(
    ->         select email
    ->         from flight_booking
    ->         where flight_no in (
    ->                 select flight_no
    ->                 from flights
    ->                 where place_from = 'Pune'
    ->                     and place_to = 'Bangalore'
    ->             )
    ->     );
+----------+
| surname  |
+----------+
| Goldberg |
| Quinn    |
+----------+
2 rows in set (0.003 sec)
```

```

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> /*6. Display the Passenger details only if they have booked
    flights on 21st July 2021. (Use Exists)*/
MariaDB [dbms_lab]> select *
    -> from passenger
    -> where exists (
    ->         select email
    ->         from flight_booking
    ->         where flight_no in(
    ->                 select flight_no
    ->                 from flights
    ->                 where departure_date = '2021-07-27'
    ->             )
    ->     );
+----------------+------------+----------+
| email          | first_name | surname  |
+----------------+------------+----------+
| beck@gmail.com | Gwen       | Beck     |
| joe@gmail.com  | Joe        | Goldberg |
| love@gmail.com | Love       | Quinn    |
+----------------+------------+----------+
3 rows in set (0.001 sec)

MariaDB [dbms_lab]> /*--7.Display the Flight-wise total time duration of flights
    if the duration is more than 8 hours (Hint : Date function,Aggregation,Grouping
    )*/
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select flight_no, timediff(f.arrival_time, f.departure_time )
    from flights as f where timediff(f.arrival_time, f.departure_time ) > "8:00:00"
     group by flight_no;
+-----------+---------------------------------------------+
| flight_no | timediff(f.arrival_time, f.departure_time ) |
+-----------+---------------------------------------------+
|     12345 | 11:47:44                                    |
|     23456 | 10:47:44                                    |
+-----------+---------------------------------------------+
2 rows in set (0.001 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> /*8.Display the Airplane-wise average seating capacity for any
     airline*/
MariaDB [dbms_lab]> select name,
    ->     avg(capacity)
    -> from airplane
    -> group by name;
+---------------+---------------+
| name          | avg(capacity) |
+---------------+---------------+
| Air India     |      175.0000 |
| Emirates      |      155.0000 |
| Qatar Airways |      183.0000 |
+---------------+---------------+
3 rows in set (0.001 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> /*9.Display the total number of flights which are booked and
    travelling to London airport.*/
MariaDB [dbms_lab]> select count(b.flight_no) as total
```

```
    -> from flight_booking b,
    ->     flights f
    -> where f.place_to = 'London';
+-------+
| total |
+-------+
|     3 |
+-------+
1 row in set (0.000 sec)

MariaDB [dbms_lab]>
MariaDB [dbms_lab]> /*10. Create a view having information about flight_no,
    airplane_no,capacity.*/
MariaDB [dbms_lab]> create view flightinfo as
    -> select f.flight_no,
    ->     a.reg_no,
    ->     a.capacity
    -> from flights f,
    ->     airplane a
    -> where a.reg_no = f.reg_no;
ERROR 1050 (42S01): Table 'flightinfo' already exists
MariaDB [dbms_lab]>
MariaDB [dbms_lab]> select * from flightinfo;
+-----------+--------+----------+
| flight_no | reg_no | capacity |
+-----------+--------+----------+
|     12345 |    111 |      180 |
|     67890 |    221 |      150 |
|     23456 |    333 |      200 |
+-----------+--------+----------+
3 rows in set (0.001 sec)
```

10 Conclusion

Thus, we have learned Subqueries commands thoroughly.

11 FAQ

1. *Explain following types of subqueries*

   - *Single-row subquery*
   - *Multiple-row subquery*
   - *Multiple-column subquery*

   The Given Subqueries can be explained as such:

   - *Single-row subquery* : A subquery that returns a single row is called a single-row subquery. A special case is the scalar subquery, which returns a single row with one column. Scalar subqueries are acceptable (and often very useful) in virtually any situation where you could use a literal value, a constant, or an expression.
   - *Multiple-row subquery* : A subquery that returns multiple rows is called a multiple-row subquery. These queries are commonly used to generate result sets that will be passed to a DML or SELECT statement for further processing. Both single-row and multiple-row subqueries will be evaluated once, before the parent query is run. Single- and multiple-row subqueries can be used in the WHERE and HAVING clauses of the parent query, but there are restrictions on the legal comparison operators
   - *Multiple-column subquery* : A subquery that returns multiple columns is called a multiple-column subquery. It is also called a correlated subquery. A correlated subquery has a more complex method of execution than single- and multiple-row subqueries and is potentially much more powerful. If a subquery references columns in the parent query, then its result will be dependent on the parent query. This makes it impossible to evaluate the subquery before evaluating the parent query.

2. *When subquery is used?*

   Subqueries are queires within queries.

   They are used for the following purposes:

   - *To find the value that is to be used in the outer query*
   - *To find the rows that are to be used in the outer query*
   - *To find the columns that are to be used in the outer query*

3. *Explain SQL SubQueries with ALL, ANY, EXISTS, SOME, With UPDATE*

   Sql subqueries can be used with the following operators, ALL, ANY, EXISTS, SOME, IN, NOT IN, =, <>, >, <, >=, <=, !=, IS NULL, IS NOT NULL, BETWEEN, NOT BETWEEN, LIKE, NOT LIKE, etc.

   They can be explained as follows:

   - *ALL* - Returns true if the subquery returns a value that is less than or equal to all the values in the subquery.

- *ANY* - Returns true if the subquery returns a value that is less than or equal to any of the values in the subquery.
- *EXISTS* - Returns true if the subquery returns any rows.
- *SOME* - Returns true if the subquery returns a value that is less than or equal to any of the values in the subquery.
- *IN* - Returns true if the subquery returns a value that is equal to any of the values in the subquery.
- *NOT IN* - Returns true if the subquery returns a value that is not equal to any of the values in the subquery

4. *How to get groupwise data from a table. What is use of Having Clause*

   Groupwise data can be obtained using the GROUP BY clause. The HAVING clause is used to filter the groups.

   An Example query would be

```
SELECT column_name, aggregate_function(column_name) from table_name having
    aggregate_function(column_name) operator value group by column_name;
```

5. *What is 'having' clause and when to use it?*

   The Having clause is used to filter the groups. It is used with the GROUP BY clause.

6. *How to display data from View. Are the views updatable? Explain*

   Data from a view can be displayed using the SELECT statement.

   Views are not updatable. They are read-only. This is because the view is not a physical table. It is a virtual table.