Subqueries

What are subqueries?

Subqueries are queries nested within another queries.

Eg:

Select id, start-time from screenings where film-id in
(select id from films where length-min >120);

outer query ↙ ↑ inner query

Subqueries can be used with select, insert, update or delete query.

The nested queries can be in the where clause or after the from ~~statement~~ clause in select statement.

These are of two types — (a) non-correlated (b) correlated

(a) Non-correlated subqueries —

The inner query can run indepentdly of the outer query.

Here, the inner query runs first and produces a result set, which is then used by the outer query.

The inner query is run only once.

(b) Correlated Subqueries —

The inner query can't run independently of the outer query.

Eg:

Select screening-id, customer-id (select ~~*~~ count (seat-id) from reserved-seat where booking-id = B.id) from bookings B

The inner query runs for every row in the outer query.

→ In non-correlated queries we can put the subquery with the where clause and also in the from clause.

Eg: select id, start-time from screenings where film-i
in (select id from films where length-min>120);

Here, subquery is used ~~with~~ the where clause.

* select avg(no-seats), max(no-seats) from (select booking-id, count(seat-id) as no-seats from reserved-seat groupby booking-id) b;
↳ derived-table name.

When we use subquery inside the from clause a derived table which is temporary in nature gets generated thus we put a name for it at the end.

Exercise-8

① Select the film name and length for all films with a length greater than the average film length.
select name, length-min from films where length-min > (select avg(length-min) from films);

② Select the maximum number of and the minimum number of screenings for a particular film.
select max(id), min(id) from (select film-id, count(id) as id from screenings group-by film-id) a;

③ Select each film name and the no. of screenings for that film

Using joins → select f.name, count(s.id) from films f join screenings s on f.id = s.film-id group by f.name;

Using Co-related subqueries:
Select name, (select count(id) from screenings where film-id = f.id) from films f;

MySQL functions -
String functions and Date functions

What are MySQL functions?

Functions are stored programs which can be passed parameters and return a value.

→ String functions :- Joining together data from multiple columns and returning it in a single column

* Concat ( ) function

Eg: select Concat(col1, col2)as new_column from table;

* Substring ( ) function - To extract a substring from a given string.

Syntax: Substring (string, start, length) → Used to take out substring

Eg: select substring ("Example", 3, 3) ; ↲ amp

length parameter is optional, if we donot include it it will start from starting index and will go till the last

* upper ( ) and lower ( ) function

Syntax: select upper (column1) as new_column_name from table;
select lower (column1) as new_column_name from table;

→ Date functions :-

* Date ( ) function: To extract date from date, datetime object.

Select date ('2017-06-14' 07:45:32'); ↲ 2017-06-14

select * from screenings where date (start_time) between '2017-06-14' and '2017-06-20';

* Month( ) function: It returns the month from a date

Select month('2018-06-14' 07:45:32) ; ↲ 06

* Year ( ) function: It returns the year from a date.

Select year ('2018-06-14'); ↲ 2018