SUBQUERIES

WHAT ARE SUBQUERIES

Subqueries are queries nested within other queries.

```
SELECT id, start_time FROM screenings
WHERE film_id IN
  (SELECT id FROM films
  WHERE length_min > 120)
;
```

```
### subqueries
    def -  queries nested within other queries - can be in WHERE caluse or FROM
            used in SELECT,INSERT,UPDATE or DELETE query
            NON-correlated and Correlated
```

NON-CORRELATED SUBQUERY

The inner query can run independently of the outer query.

```
SELECT id, start_time FROM screenings
WHERE film_id IN
  (SELECT id FROM films
  WHERE length_min > 120)
;
```

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

which are then used by the outer query.

part1

```
USE cinema_booking_system;

SELECT id, start_time FROM screenings
WHERE film_id IN
  (SELECT id FROM films
   WHERE length_min > 120);

SELECT id FROM films
WHERE length_min > 120;


SELECT * FROM customers;
SELECT * FROM bookings;
```

id	start_time
10	2017-10-02 18:00:00
14	2017-10-02 19:30:00
15	2017-10-02 20:00:00
19	2017-10-03 19:00:00
23	2017-10-04 18:30:00
29	2017-10-04 19:30:00
34	2017-10-06 15:00:00
41	2017-10-07 13:30:00

we can see it's returned id and start time

b, customers that made a booking for screening_id =1

```
SELECT first_name, last_name, email FROM customers
WHERE id IN
(SELECT customer_id FROM bookings
WHERE screening_id = 1);
```

first_name	last_name	email
John	Smith	smithy@gmail.com
Cherry	Wang	cherrywa@gmail.com
Simon	Davis	sdavis@gmail.com

part2

no of seats reserved for booking_id

```
SELECT * FROM reserved_seat;

SELECT booking_id, COUNT(seat_id) FROM reserved_seat
GROUP BY booking_id;
```

booking_id	COUNT(seat_id)
1	3
2	2
3	2
4	2
5	1
6	2
7	2

find the max number of seats reserved by a part booking_id

since we are creating a direct table by

SELECT booking_id, COUNT(seat_id) AS no_seats FROM reserved_seat GROUP BY booking_id

SO, we have to provide a name too, say b, then just SELECT MAX(no_seats) FROM b

```
SELECT MAX(no_seats) FROM
(SELECT booking_id, COUNT(seat_id) AS no_seats FROM reserved_seat
GROUP BY booking_id) b;
```

100% 10:8

Result Grid | Filter Rows: Search | Export:

MAX(no_seats)
6

WE can choose multiple columns from this direct table also

```
SELECT AVG(no_seats), MAX(no_seats) FROM
(SELECT booking_id, COUNT(seat_id) AS no_seats FROM reserved_seat
GROUP BY booking_id) b;

SELECT booking_id, COUNT(seat_id) AS no_seats FROM reserved_seat
GROUP BY booking_id;
```

100% 23:9

Result Grid | Filter Rows: Search | Export:

AVG(no_seats)	MAX(no_seats)
1.8122	6

CORRELATED SUBQUERY

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

```
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.

the inner query is running multiple times

```
SELECT screening_id, customer_id,
(SELECT COUNT(seat_id)
FROM reserved_seat WHERE booking_id = b.id)
FROM bookings b
ORDER BY screening_id;
```

00% 43:10

Result Grid | Filter Rows: Search | Export:

screening_id	customer_id	(SELECT COUNT(seat_id)
1	4	2
1	12	2
1	16	4
2	2	3
2	6	2
2	18	2
3	20	1
4	8	2
4	12	2
4	67	2

now, running the inner query will give error

```
SELECT COUNT(seat_id)
FROM reserved_seat WHERE booking_id = b.id;
```

as booking as b was declared in the outer query

192 13:59:06 SELECT COUNT(seat_id) FROM reserved_seat WHERE booking_id = b.id LIMIT 0,... Error Code: 1054. Unknown column 'b.id' in 'where clause'

EX. a. non correlated query

SELECT name, length_min FROM films

WHERE length_min >

(SELECT AVG(length_min) as average FROM films);

```
1  USE cinema_booking_system;
2
3  -- 1. Select the film name and length for all films with a length greater than the average film le
4
5  SELECT name, length_min FROM films
6  WHERE length_min >
7  (SELECT AVG(length_min) FROM films);
8
```

100% 36:7

Result Grid | Filter Rows: Search | Export:

name	length_min
Blade Runner 2049	153
Geostorm	121
Jigsaw	110
Murder on the Orient Express	135
Breathe	117
Blade Runner	127

verification that it is NCQ

```
8
9  SELECT AVG(length_min) FROM films;
10
```

100% 25:9

Result Grid | Filter Rows: Search | Export:

AVG(length_min)
115.0833

b. -- 2. Select the maximum number and the minimum number of screenings for a particular film.

AGAIN Ncq

SELECT film_id, COUNT(id) FROM screenings

GROUP BY film_id

```
SELECT film_id, COUNT(id) FROM screenings
GROUP BY film_id;
```

film_id	COUNT(id)
1	24
2	25
3	20
4	18
5	26
6	19
7	22
8	14
9	18
10	6
11	6
12	15

gives each film id has how many screenings

- ➔ **now just select max and min from this table**

```
SELECT MAX(id), MIN(id) FROM
(SELECT film_id, COUNT(id) AS id FROM screenings
GROUP BY film_id) a;
```

MAX(id)	MIN(id)
26	6

c. -- 3. Select each film name and the number of screenings for that film.

SELECT name

(SELECT COUNT(id) FROM screenings

WHERE film_id=f.id)

FROM films f;

- ➔ **we could have use group by screening but films and screenings are two diff table so we cant**

```
19  SELECT name,
20  (SELECT COUNT(id) FROM screenings
21  WHERE film_id = f.id
22  )
23  FROM films f;
24
25
26
```

100% 1:26

Result Grid Filter Rows: Search Export:

name	(SELECT COUNT(id) FROM screenings
Blade Runner	6
Blade Runner 2049	24
Breathe	6
Dunkirk	25
Geostorm	20
Jigsaw	26
Murder on the Orient Express	14
Paddington 2	18
The Death of Stalin	19
The Lego Ninjago Movie	22
Thor: Ragnarok	18