Advanced SQL practise questions.

Problem 1:

I have below table, of ICC cricket matches, I need to get team name, number of matches played, number of wins and number of loses on country level.

Team_1	Team_2	Winner
India	SL	India
SL	Aus	Aus
SA	Eng	Eng
Eng	NZ	NZ
Aus	India	India

Solution:

```
select * from icc_world_cup;
select players, count(1) as num_matches_played, sum(fl_won) as num_matche
s_won, count(1) - sum(fl_won) as num_matches_lost
from(
select team_1 as players, case when Winner = Team_1 then 1 else 0 end as fl_
won from icc_world_cup
union all
select Team_2 as players, case when Winner = Team_2 then 1 else 0 end as fl
_won from icc_world_cup) a
group by 1;
```

players	num_matches_played	num_matches_won	num_matches_lost
India	2	2	0
SL	2	0	2
SA	1	0	1
Eng	2	1	1
Aus	2	1	1
NZ	1	1	0

Problem 2:

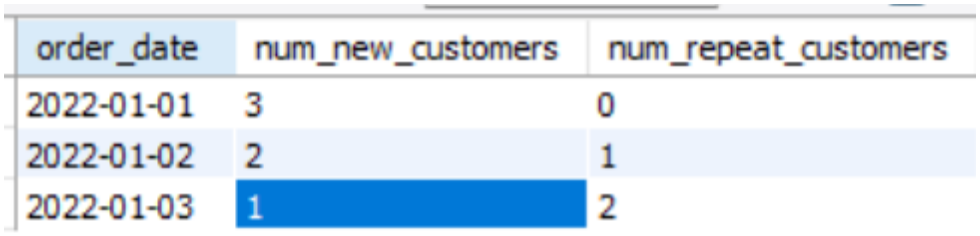
Find number of new customers and repeat customers from customers orders table. Output should be order_date, #new_customers, #repeat_customers.

order_id	customer_id	order_date	order_amount
1	100	2022-01-01	2000
2	200	2022-01-01	2500
3	300	2022-01-01	2100
4	100	2022-01-02	2000
5	400	2022-01-02	2200
6	500	2022-01-02	2700
7	100	2022-01-03	3000
8	400	2022-01-03	1000

solution:

```
with new as(
select customer_id, order_date,
min(order_date) over(partition by customer_id order by order_date) as first_order
from customer_orders)
select order_date,
count(case when order_date = first_order_date then customer_id end) as num_ne
```

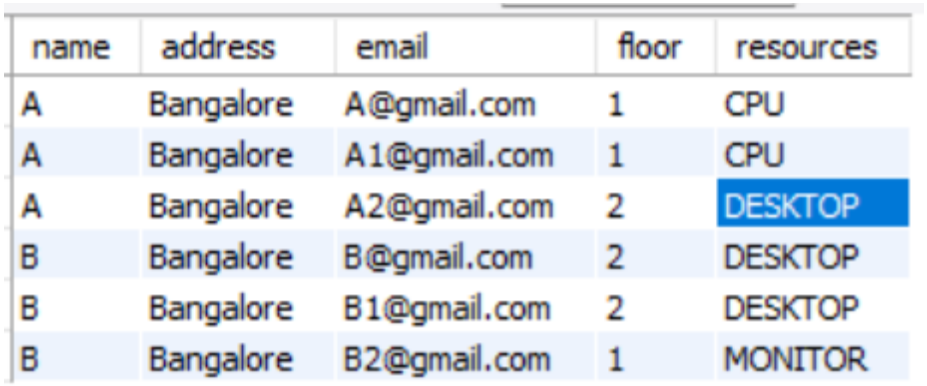
```
count(case when order_date != first_order_date then customer_id end) as num_re
from new group by 1
```



order_date	num_new_customers	num_repeat_customers
2022-01-01	3	0
2022-01-02	2	1
2022-01-03	1	2

Problem 3:

In a company, employee can enter only once using their mail. But their is a loophole, they can give another mail id and enter. we want to know for each employee, number of times entered, floor that they entered maximum and all names of the floor entered.



name	address	email	floor	resources
A	Bangalore	A@gmail.com	1	CPU
A	Bangalore	A1@gmail.com	1	CPU
A	Bangalore	A2@gmail.com	2	DESKTOP
B	Bangalore	B@gmail.com	2	DESKTOP
B	Bangalore	B1@gmail.com	2	DESKTOP
B	Bangalore	B2@gmail.com	1	MONITOR

solution:

```
with a as (
select name, floor,
rank() over(partition by floor order by count(1) desc) as rnk
from entries
group by 1,2)
```

```
select a.name, a.floor as max_visited,
count(a.name) as num_times_visited,
group_concat(distinct resources separator ',') as floors_visited
from  a
join entries b on a.name = b.name
where a.rnk = 1
group by 1,2;
```

name	max_visited	num_times_visited	floors_visted
A	1	3	CPU,DESKTOP
B	2	3	DESKTOP,MONITOR

Problem 4:

write a query to provide the date for nth occurence of sunday in future from given date.

Solution:

```
SET @given_date = '2025-03-30';
SET @n = 2;

-- Find the next Sunday
SET @days_to_sunday = (6 - WEEKDAY(@given_date)) % 6;
-- Calculate the nth Sunday from the given date
SELECT @given_date AS given_date,
@days_to_sunday,
WEEKDAY(@given_date) AS weekday_of_given_date,
DATE_ADD(@given_date, INTERVAL @days_to_sunday + (@n - 1) * 7 DAY) AS
nth_sunday_date;
```

given_date	@days_to_sunday	weekday_of_given_date	nth_sunday_date
2025-03-30	0	6	2025-04-06

Problem 5:

There are 2 tables, 1 has person and score, table 2 has person and friend. Get output as person, no of friends and total friends score(sum).

PersonID	Name	Score
1	Alice	88
2	Bob	11
3	Devis	27
4	Tara	45
5	John	63

pid	fid
1	3
2	1
2	3
3	5
4	2
4	3

Solution:

```
select a.pid, b.Name, num_friends, total_Score from
(
select pid, count(fid) as num_friends, sum(score) as total_Score
from (
select b.*, score
from friend b
join  person a  on a.PersonID = b.fid) a
group by 1
having sum(score) > 100) a
join person b
on a.pid = b.PersonID;
```

pid	Name	num_friends	total_Score
2	Bob	2	115
4	Tara	3	101

Problem 6:

Find cancellation requests for unbanned users on daily basis

id	client_id	driver_id	city_id	status	request_at
2	2	11	1	cancelled_by_driver	2013-10-01
3	3	12	6	completed	2013-10-01
4	4	13	6	cancelled_by_client	2013-10-01
5	1	10	1	completed	2013-10-02
6	2	11	6	completed	2013-10-02
7	3	12	6	completed	2013-10-02
8	2	12	12	completed	2013-10-03
9	3	10	12	completed	2013-10-03
10	4	13	12	cancelled_by_driver	2013-10-03

users_id	banned	role
1	No	client
2	Yes	client
3	No	client
4	No	client
10	No	driver
11	No	driver
12	No	driver
13	No	driver

solution:

```
select request_at,
round(COALESCE(sum(case when status = 'completed' then 0 else 1 end) , nu
ll) 100/  count(id),2) as num_orders
from(
select a. from Trips a
inner join Users1 b on a.client_id = b.users_id
inner join Users1 c on a.driver_id = c.users_id
where b.banned = "No" and c.banned = "No") a
group by 1
```

request_at	cancellation_rate
2013-10-01	33.33
2013-10-02	0.00
2013-10-03	50.00

Problem 7:

We have a group and members along with thier score. Find rank of member within each group. If there is a ties, then rank to min_id

player_id	group_id
15	1
25	1
30	1
45	1
10	2
35	2
50	2
20	3
40	3

match_id	first_player	second_player	first_score	second_score
1	15	45	3	0
2	30	25	1	2
3	30	15	2	0
4	40	20	5	2
5	35	50	1	1

solution:

```
with cte as
(select first_player as player, first_score as score
from matches
union all
select second_player as player, second_score as score
from matches) ,
```

```
cte2 as
(
select b.group_id, a.player,  sum(score) as score
from cte a
join players b
on a.player = b.player_id
group by 1,2)
select *,
rank() over(partition by group_id order by score desc, player) as rnk
from cte2
```

group_id	player	score	rnk
1	15	3	1
1	30	3	2
1	25	2	3
1	45	0	4
2	35	1	1
2	50	1	2
3	40	5	1
3	20	2	2

Problem 8:

Market analysis: for each seller find if the brand sold on second day is their favourite brand (yes/no).

Output should be like seller_id, is_2n_favourite_brand (yes/no)

user_id	join_date	favorite_brand
1	2019-01-01	Lenovo
2	2019-02-09	Samsung
3	2019-01-19	LG
4	2019-05-21	HP

item_id	item_brand
1	Samsung
2	Lenovo
3	LG
4	HP

order_id	order_date	item_id	buyer_id	seller_id
1	2019-08-01	4	1	2
2	2019-08-02	2	1	3
3	2019-08-03	3	2	3
4	2019-08-04	1	4	2
5	2019-08-04	1	3	4
6	2019-08-05	2	2	4

solution:

```
with cte as(
select seller_id, item_id
from(
select a.order_date, a.seller_id, a.item_id, b.item_brand,
rank() over(partition by seller_id order by order_date) as rnk
from orders a
join items b on a.item_id = b.item_id) a
where rnk = 2)
select user_id as seller_id, case when c.item_brand = a.favorite_brand then 'y
es' else 'no' end as 2nd_favorite_prdt_sold
from users2 a
left join cte b on a.user_id = b.seller_id
left join items c on b.item_id = c.item_id
```

seller_id	2nd_favorite_prdt_sold
1	no
2	yes
3	yes
4	no

Problem 9:

For a user, we have task date and status. get streak wise start and end date for status.

date_value	state
2019-01-01	success
2019-01-02	success
2019-01-03	success
2019-01-04	fail
2019-01-05	fail
2019-01-06	success

solution:

```
select min(date_value) as start_date, max(date_value) as end_date, state
from
(
SELECT
date_value, state,
ROW_NUMBER() OVER (PARTITION BY state ORDER BY date_value),
DATE_SUB(date_value, INTERVAL ROW_NUMBER() OVER (PARTITION BY state
ORDER BY date_value) DAY) AS group_id
FROM tasks) a
```

```
group by group_id, state
order by start_date
```

start_date	end_date	state
2019-01-01	2019-01-03	success
2019-01-04	2019-01-05	fail
2019-01-06	2019-01-06	success

Problem 10:

User purchase platform. we have user and their purchase (desktop/mobile/both) on date level. Find the split. output should be date, platform(desktop/mobile/both), total_amount, total users.

user_id	spend_date	platform	amount
1	2019-07-01	mobile	100
1	2019-07-01	desktop	100
2	2019-07-01	mobile	100
2	2019-07-02	mobile	100
3	2019-07-01	desktop	100
3	2019-07-02	desktop	100

solution:

```
with all_spend as(
select spend_date,  user_id, max(platform) as platform, count(distinct user_id)
as num_users, sum(amount) as total_amount
from spending
```

```
group by 1,2
having count(platform) = 1

union all

select spend_date, user_id, 'both'  as platform, count(distinct user_id) as num
_users, sum(amount) as total_amount
from spending
group by 1,2
having count(platform) = 2

union all

select  distinct spend_date, null user_id, 'both'  as platform, 0 as num_users, 0
as total_amount
from spending
group by 1,2)
select spend_date, platform, count(distinct user_id) as num_users, sum(total_a
mount) as total_amount
from all_spend
group by 1,2;
```

spend_date	platform	num_users	total_amount
2019-07-01	both	1	200
2019-07-01	desktop	1	100
2019-07-01	mobile	1	100
2019-07-02	both	0	0
2019-07-02	desktop	1	100
2019-07-02	mobile	1	100

Problem 11:

given a unclear period split and average daily sales. I need YOY total sales.

product_id	period_start	period_end	average_daily_sales
1	2019-01-25	2019-02-28	100
2	2018-12-01	2020-01-01	10
3	2019-12-01	2020-01-31	1

solution:

```
with recursive cte as (
select min(period_start) as dates, max(period_end) as max_date from sales
union all
select date_add(dates, interval 1 day) as dates, max_date from cte
where dates <max_date)
select year(dates) as year, product_id, sum(average_daily_sales) as total_amo
unt from cte
inner join sales on dates between period_start and period_end
group by 1,2
order by 1,2;
```

year	product_id	total_amount
2018	2	310
2019	1	3500
2019	2	3650
2019	3	31
2020	2	10
2020	3	31

Problem 12:

get frequently brought pairs in one order, that will help in recommendation.

order_id	customer_id	product_id
1	1	1
1	1	2
1	1	3
2	2	1
2	2	2

id	name
1	A
2	B
3	C
4	D
5	E

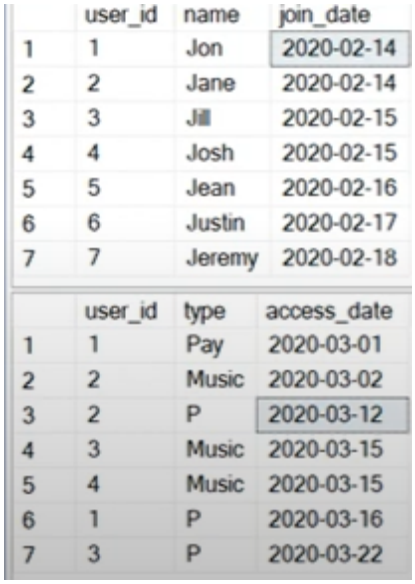
solution:

```
select pair, count(*) as num
from(
SELECT
a.order_id,
CONCAT(c.name, '-', d.name) AS pair
FROM orders1 a
INNER JOIN orders1 b
ON a.order_id = b.order_id
inner join products c on a.product_id = c.id
inner join products d on b.product_id = d.id
WHERE a.product_id != b.product_id
AND a.product_id > b.product_id) a
group by 1;
```

pair	num
B-A	2
D-A	1
C-A	1
D-B	1
C-B	1

Problem 13:

prime subsciprtion rate by product action. find percent who subscribed to prime out of access music within first 30 days of signing up.



	user_id	name	join_date
1	1	Jon	2020-02-14
2	2	Jane	2020-02-14
3	3	Jill	2020-02-15
4	4	Josh	2020-02-15
5	5	Jean	2020-02-16
6	6	Justin	2020-02-17
7	7	Jeremy	2020-02-18

	user_id	type	access_date
1	1	Pay	2020-03-01
2	2	Music	2020-03-02
3	2	P	2020-03-12
4	3	Music	2020-03-15
5	4	Music	2020-03-15
6	1	P	2020-03-16
7	3	P	2020-03-22

solution:

```
with denominator as (
select user_id, join_date from users3 where user_id in (select user_id from eve
nts1 where type = 'Music')),
numerator as(
select user_id, access_date from events1 where type = 'P' and
user_id in (select user_id from denominator)),
sets as(
select a.*, b.access_date ,DATEDIFF(access_date, join_date) AS date_differenc
e
from denominator a
left join numerator b on a.user_id = b.user_id)
select round(count(case when date_difference < 31 then access_date end)/co
```

```
unt(join_date),2)*100 as music_subs_rate
from sets;
```

music_subs_rate
33.00