Data assignment

Problem Statement 1:

Data Definitions:
1. Id - Primary Key for Table
2. UserId - Unique Id for every users
3. Total - Total amount spent on each order
4. Created - Timestamp for order created

Q1. We need SQL query to identify weekly Retention for Users from their first created order Week in the following format.
(Kindly mention if any assumptions are made)
Ans: GitHub Link
Assumptions made:
1. Retention is assumed to be the placement of order by a user in a given week. In other words, retention is defined as follows – if a given user places an order in a given week, then he/she is said to have been 'retained' .
2. The solution assumes that as displayed in the question, the analysis is to be done till the week 10 of user account creation. Further analysis simply require the addition of a small amount of code.

Code:
```
WITH firstcreated AS (
SELECT
                userid,
       MIN(created) AS CreatedDate,
       WEEK(MIN(created)) AS CreatedWeek
FROM q1
GROUP BY userid
), user_week_orders AS (
SELECT
                q1.userid AS UserId,
       fc.CreatedDate,
       -- Create the Start of the week. Formula considers Monday as start of week
       CAST(DATE_ADD(fc.CreatedDate, INTERVAL(-WEEKDAY(fc.CreatedDate)) DAY) AS
DATE)
                        AS CreatedDateWeekStart,
       WEEK(q1.created) AS OrderWeek,
       fc.CreatedWeek AS CreatedWeek,
       (WEEK(q1.created) - fc.CreatedWeek) AS ActiveWeek
FROM q1
INNER JOIN firstcreated fc
USING(userid)
)
-- SELECT * FROM user_week_orders;
SELECT
                CreatedDateWeekStart AS 'Week Start Date',
       SUM(CASE WHEN ActiveWeek = 0 THEN 1 ELSE 0 END) AS 'Week 0',
       SUM(CASE WHEN ActiveWeek = 1 THEN 1 ELSE 0 END) AS 'Week 1',
       SUM(CASE WHEN ActiveWeek = 2 THEN 1 ELSE 0 END) AS 'Week 2',
       SUM(CASE WHEN ActiveWeek = 3 THEN 1 ELSE 0 END) AS 'Week 3',
       SUM(CASE WHEN ActiveWeek = 4 THEN 1 ELSE 0 END) AS 'Week 4',
       SUM(CASE WHEN ActiveWeek = 5 THEN 1 ELSE 0 END) AS 'Week 5',
       SUM(CASE WHEN ActiveWeek = 6 THEN 1 ELSE 0 END) AS 'Week 6',
       SUM(CASE WHEN ActiveWeek = 7 THEN 1 ELSE 0 END) AS 'Week 7',
       SUM(CASE WHEN ActiveWeek = 8 THEN 1 ELSE 0 END) AS 'Week 8',
       SUM(CASE WHEN ActiveWeek = 9 THEN 1 ELSE 0 END) AS 'Week 9',
       SUM(CASE WHEN ActiveWeek = 10 THEN 1 ELSE 0 END) AS 'Week 10'
FROM user_week_orders
GROUP BY CreatedDateWeekStart;
```

Problem Statement 2:

Q1. Find number of employees inside the Office at current time
Ans: GitHub Link

To accomplish the task, find the count of logins till date and logouts till date and calculate their difference.

```
SELECT
                (SELECT COUNT(*) FROM giventable WHERE Action = "In")
        -
                (SELECT COUNT(*) FROM giventable WHERE Action = "Out");
```

Q2. Find number of employees inside the Office at "2021-05-01 19:05:00"
Ans: GitHub Link

To accomplish the task, find the count of logins and logouts in the till datetime range and calculate their difference.

```
SELECT
                (SELECT COUNT(*) FROM giventable WHERE Action = "In"
        AND Created <= "2021-05-01 19:05:00")
        -
                (SELECT COUNT(*) FROM giventable WHERE Action = "Out"
        AND Created <= "2021-05-01 19:05:00");
```

Q3. Measure amount of hours spent by each employee inside the office since the day they started (Account for current shift if she/he is working).
Ans: GitHub Link

The task can be accomplishment by creating CTEs (Common Table Expressions) in specific way.

Step 1: Create 2 CTEs, rank the "in time" (Created column with Action "In") and "out time" (Created column with Action "Out") for each employee as shown below.
Step 2: Join the 2 CTEs based on the Employee id and Ranks (aliases are used below)
Step 3: Compute the total difference between the "in time" and "out time" of each Employee as shown below using TIMESTAMPDIFF() function.

```
WITH Emp_In AS (
SELECT
         "Employee id" AS Emp_id,
         Created AS in_time,
        ROW_NUMBER() OVER (PARTITION BY Emp_id ORDER BY Created) AS in_number
FROM giventable
WHERE Action = "In"
), Emp_out AS (
SELECT
        "Employee id" AS Emp_id,
        Created AS out_time,
        ROW_NUMBER() OVER (PARTITION BY Emp_id ORDER BY Created) AS out_number
FROM giventable
WHERE Action = "Out"
)
SELECT i.Emp_id, SUM(TIMESTAMPDIFF(HOUR, i.in_time, COALESCE(o.out_time, NOW())))
FROM Emp_In i
LEFT JOIN Emp_out o
ON i.Emp_id = o.Emp_id AND i.in_number = o.out_number
GROUP BY i.Emp_id;
```

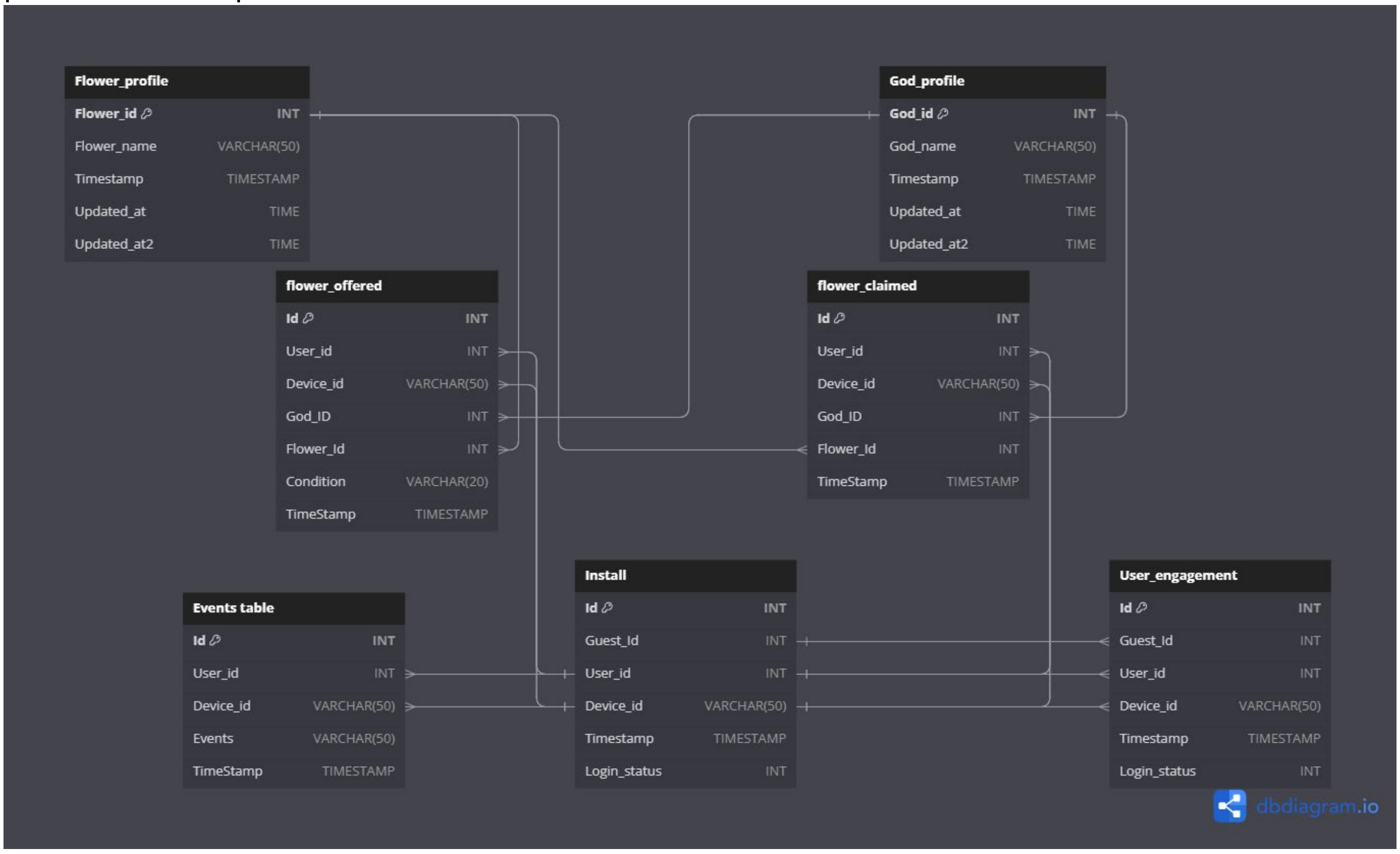
Q4. Measure amount of hours spent by each employee inside the office between "2021-04-01 14:00:00" and "2021-04-02 10:00:00"
Ans: GitHub Link

Same steps as in Problem 2 Question 3.
Just introduce the time period constraints specified while ranking as shown below.

```
WITH Emp_In AS (
SELECT
        "Employee id" AS Emp_id,
        Created AS in_time,
        ROW_NUMBER() OVER (PARTITION BY Emp_id ORDER BY Created) AS in_number
FROM giventable
WHERE Action = "In" AND Created BETWEEN "2021-04-01 14:00:00" AND "2021-04-02
10:00:00"
), Emp_out AS (
SELECT
        "Employee id" AS Emp_id,
        Created AS out_time,
        ROW_NUMBER() OVER (PARTITION BY Emp_id ORDER BY Created) AS out_number
FROM giventable
WHERE Action = "Out"
AND (Created BETWEEN (SELECT MIN(in_time) FROM Emp_In) AND "2021-04-02 10:00:00")
)
SELECT i.Emp_id, SUM(TIMESTAMPDIFF(HOUR, i.in_time, COALESCE(o.out_time, NOW())))
FROM Emp_In i
LEFT JOIN Emp_out o
ON i.Emp_id = o.Emp_id AND i.in_number = o.out_number
GROUP BY i.Emp_id;
```

Problem Statement 3:

This is the E-R Diagram, I could construct based on tables (and their respective columns) present in the spreadsheet.



Q1. DAU who clicked on Flower Icon on Mandir Home %
Ans: GitHub Link

Considering that all events are recorded in user_engagement table, a simple join and count can be used to perform the task.

```
SELECT
    DATE(ue.timestamp) AS Date,
    COUNT(ue.device_id) / COUNT(*) * 100 AS 'DAU mandir home %'
FROM user_engagement ue
LEFT JOIN event_table et
ON ue.device_id = et.device_id
WHERE et.events = 'mandir_home'
GROUP BY Date;
```

Q2. DAU who Offered at least 1 Flower %
And: GitHub Link

Considering that all flower offerings events are recorded in user_engagement table, a simple join and count can be used to perform the task.

```
SELECT
    DATE(ue.timestamp) AS Date,
    COUNT(ue.device_id) / COUNT(*) * 100 AS 'DAU flower offered %'
FROM user_engagement ue
```

```
LEFT JOIN flower_offered fo
ON ue.device_id = fo.device_id
GROUP BY Date;
```

Q3. DAU who Claimed at least 1 Flower %
Ans: GitHub Link

Considering that all flower claiming events are recorded in user_engagement table, a simple join and count can be used to perform the task.

```
SELECT
    DATE(ue.timestamp) AS Date,
    COUNT(ue.device_id) / COUNT(*) * 100 AS 'DAU flower claimed %'
FROM user_engagement ue
LEFT JOIN flower_claimed fc
ON ue.device_id = fc.device_id
WHERE ue.login_status=1
GROUP BY Date;
```

Q4. DAU who clicked on Locked Flowers %
Ans: GitHub Link

Considering that all flower offerings events are recorded in user_engagement table, a simple join and count can be used to perform the task.

```
SELECT
    DATE(ue.timestamp) AS Date,
    COUNT(ue.device_id) / COUNT(*) * 100 AS 'DAU locked flower click %'
FROM user_engagement ue
LEFT JOIN flower_offered fo
ON ue.device_id = fo.device_id
WHERE fo.Condition = 'Locked'
GROUP BY Date(ue.timestamp);
```