1)

SELECT

CASE

WHEN HOUR(al.check\_in\_time) BETWEEN 8 AND 11 THEN 'Morning (8 AM to 12 PM)'

WHEN HOUR(al.check\_in\_time) BETWEEN 12 AND 16 THEN 'Afternoon (12 PM to 5 PM)'

WHEN HOUR(al.check\_in\_time) BETWEEN 17 AND 23 THEN 'Evening (5 PM to 11 PM)'

END AS `Time of day`,

COUNT(al.attendance\_id) AS `Total checkins`,

COUNT(CASE WHEN al.member\_id IS NOT NULL THEN 1 END) AS `Member Check-ins`,

COUNT(CASE WHEN al.member\_id IS NULL THEN 1 END) AS `Non-member check-ins`,

ROUND(

COUNT(al.attendance\_id) \* 100.0 / (SELECT COUNT(\*) FROM attendance\_log), 2

) AS `Check-in Percentage`

FROM

Attendance\_Log al

WHERE

HOUR(al.check\_in\_time) BETWEEN 8 AND 22

GROUP BY

`Time of day`

ORDER BY

FIELD(`Time of day`, 'Morning (8 AM to 12 PM)', 'Afternoon (12 PM to 5 PM)', 'Evening (5 PM to 11 PM)');

2)

SELECT

CASE

WHEN DAYOFWEEK(check\_in\_date) IN (1, 7) THEN 'Weekend'

ELSE 'Weekday'

END AS `Day Type`,

COUNT(attendance\_id) AS `Total Check-ins`,

SEC\_TO\_TIME(AVG(TIME\_TO\_SEC(check\_in\_time))) AS `Average Check-in Time`

FROM Attendance\_log

WHERE check\_in\_date IS NOT NULL

GROUP BY `Day Type`;

3)

SELECT

member\_id,

YEAR(check\_in\_date) AS year,

MONTHNAME(check\_in\_date) AS month,

COUNT(\*) AS total\_check\_ins

FROM Attendance\_log

WHERE member\_id

GROUP BY member\_id, year, month

ORDER BY year DESC, month DESC;

4)

WITH TagCounts AS (

SELECT

al.event\_id,

mt.tag\_id,

COUNT(mt.tag\_id) AS tag\_count

FROM Attendance\_log al

JOIN Member\_Tag mt ON al.member\_id = mt.member\_id

WHERE al.event\_id IS NOT NULL

GROUP BY al.event\_id, mt.tag\_id

),

MostPopularTag AS (

SELECT

event\_id,

tag\_id,

tag\_count

FROM TagCounts tc

WHERE tag\_count = (SELECT MAX(tag\_count)

FROM TagCounts

WHERE event\_id = tc.event\_id))

SELECT

e.event\_name as 'Event Name',

t.tag\_name as 'Tag',

mpt.tag\_count as 'Number of people with the tag'

FROM MostPopularTag mpt

JOIN Event e ON mpt.event\_id = e.event\_id

JOIN Tags t ON mpt.tag\_id = t.tag\_id

ORDER BY e.event\_name;

5)

SELECT

m.member\_id AS 'Member id',

CONCAT(m.first\_name, ' ', m.last\_name) AS 'Member Name',

COUNT(\*) AS 'Events Attended'

FROM

attendance\_log a

LEFT JOIN

member m ON a.member\_id = m.member\_id

WHERE

a.check\_in\_date BETWEEN '2024-01-01' AND '2024-12-31'

AND a.member\_id IS NOT NULL

GROUP BY

a.member\_id, m.first\_name, m.last\_name

ORDER BY

'Events Attended' DESC;

6)

SELECT

al.member\_id,

COUNT(al.attendance\_id) AS 'Total logs',

COUNT(CASE WHEN al.event\_id IS NULL THEN 1 END) AS 'Space access logs',

COUNT(CASE WHEN al.event\_id IS NOT NULL THEN 1 END) AS 'Event logs'

FROM

Attendance\_Log al

WHERE

al.member\_id IS NOT NULL

and check\_in\_date

GROUP BY

al.member\_id

ORDER BY

al.member\_id;

7)

SELECT

al.email AS `Email`,

al.attendee\_name AS `Attendee Name`,

e.event\_name AS `Event Name`,

et.event\_type\_description AS `Event Type`

FROM Attendance\_log al

LEFT JOIN Event e

ON al.event\_id = e.event\_id

LEFT JOIN Event\_Type et

ON e.event\_type\_id = et.event\_type\_id

WHERE al.member\_id IS NULL

and e.event\_id is not null

ORDER BY al.attendee\_name ASC;

8)

SELECT

m.member\_id,

CONCAT(m.first\_name, '', m.last\_name) AS 'Name',

COUNT(DISTINCT l.log\_in\_id) AS 'Number of sign ins', -- Count unique sign-ins

COUNT(DISTINCT b.bookings\_id) AS 'Events booked', -- Count unique events booked

COUNT(DISTINCT a.attendance\_id) AS 'Events attended', -- Count unique events attended

m.joining\_date AS 'Joining date', -- Use date\_of\_joining from members table

COUNT(DISTINCT mt.tag\_id) AS 'Amount of tags', -- Count unique tags for the member

COUNT(DISTINCT mi.interest\_id) AS 'Amount of interests', -- Count unique interests for the member

mem.membership\_type AS 'Membership' -- Membership type for the member

FROM

member m

LEFT JOIN log\_in l

ON m.member\_id = l.member\_id -- Joining for login info

LEFT JOIN bookings b

ON m.member\_id = b.member\_id -- Joining for booking info

LEFT JOIN attendance\_log a

ON m.member\_id = a.member\_id AND a.event\_id IS NOT NULL -- Joining for attendance with valid events

LEFT JOIN member\_tag mt

ON m.member\_id = mt.member\_id -- Joining for member tags

LEFT JOIN member\_interest mi

ON m.member\_id = mi.member\_id -- Joining for member interests

LEFT JOIN membership mem

ON m.membership\_id = mem.membership\_id

where m.member\_id

GROUP BY

m.member\_id, mem.membership\_type, m.joining\_date -- Group by member, membership type, and joining date

ORDER BY

m.member\_id;

9)

SELECT

m.membership\_id as 'Membership ID',

mem.membership\_type 'Membership Type',

COUNT(a.member\_id) AS 'Attendance count'

FROM attendance\_log a

JOIN

member m

ON a.member\_id = m.member\_id

JOIN

membership mem

On m.membership\_id = mem.membership\_id

GROUP BY m.membership\_id, `Membership Type`

ORDER BY 'Attendance count' DESC;

10)

SELECT

CEIL(AVG(follow\_up\_count)) AS 'Average follow-up count'

FROM

inquiries

WHERE

member\_id IS NOT NULL

AND user\_type\_id = 1;