## 1. Marketing SQL Code:

For Marketing KPIs use both of the tables generated from these two codes, joining them on Power BI.

### Step 1.1:

WITH temp AS

(

SELECT

DISTINCT user\_cookie\_id as user\_cookie\_id

FROM `Prism\_Main.sessions`

),

temp2 AS

(

SELECT

user\_cookie\_id,

SUM(transaction\_revenue) AS total\_revenue,

'Registered' AS registration\_status

FROM `Prism\_Main.transactions`

GROUP BY user\_cookie\_id

)

SELECT

temp.user\_cookie\_id,

COALESCE(temp2.total\_revenue, 0) AS total\_revenue,

CASE

WHEN temp2.registration\_status IS NULL THEN 'Not Registered'

ELSE temp2.registration\_status

END AS registration\_status

FROM temp

LEFT JOIN temp2

ON temp.user\_cookie\_id = temp2.user\_cookie\_id

### Step 1.2:

SELECT

date,

session\_id,

user\_cookie\_id,

CASE

WHEN traffic\_source = 'lm.facebook.com' OR traffic\_source = 'l.facebook.com' OR traffic\_source = 'facebook.com' OR traffic\_source = 'm.facebook.com' THEN 'facebook'

WHEN traffic\_source = 'system\_mail' THEN 'email'

WHEN traffic\_source = 'youtube.com' THEN 'youtube'

WHEN traffic\_source = 'instagram.com' OR traffic\_source = 'l.instagram.com' THEN 'instagram'

WHEN traffic\_source = 'tiktok.com' THEN 'tiktok'

WHEN traffic\_source = 'twitter.com' THEN 'twitter'

WHEN traffic\_source = '(direct)' THEN 'directly'

ELSE traffic\_source

END AS traffic\_source

FROM `Prism\_Main.sessions`

## 2. Customer SQL Code:

### Step 2.1:

Saved the result in a table called prism\_test.Indigo-SC-W3-Basis

WITH temp2 AS

(

SELECT date, user\_cookie\_id, SUM(transaction\_revenue) AS total\_revenue, COUNT(transaction\_revenue) AS no\_transactions

FROM `Prism\_Main.transactions`

GROUP BY date, user\_cookie\_id

),

temp3 AS

(

SELECT date, user\_cookie\_id, COUNT(DISTINCT session\_id) AS total\_visits

FROM `Prism\_Main.funnelevents`

GROUP BY date, user\_cookie\_id

),

temp4 AS

(

SELECT date, user\_cookie\_id, COUNT(DISTINCT session\_id) AS total\_purchase

FROM `Prism\_Main.funnelevents`

WHERE event\_name = 'purchase'

GROUP BY date, user\_cookie\_id

)

SELECT temp2.date, temp2.user\_cookie\_id, temp2.total\_revenue, temp2.no\_transactions, temp3.total\_visits, temp4.total\_purchase

FROM temp2

LEFT JOIN temp3

ON temp2.user\_cookie\_id = temp3.user\_cookie\_id AND temp2.date = temp3.date

LEFT JOIN temp4

ON temp3.user\_cookie\_id = temp4.user\_cookie\_id AND temp3.date = temp4.date

### Step 2.2:

Saved the result in a table called prism\_test.Indigo-SC-W3-Seg

WITH temp AS

(

SELECT format\_date('%Y', date) AS year, user\_cookie\_id, SUM(total\_revenue) as total\_revenue, SUM(no\_transactions) AS no\_transactions, SUM(total\_visits) AS total\_visits, SUM(total\_purchase) AS total\_purchases

FROM `prism\_test.Indigo-SC-W3-Basis`

GROUP BY year, user\_cookie\_id

),

temp2 AS

(

SELECT user\_cookie\_id, date, MAX(date) OVER (PARTITION BY user\_cookie\_id, format\_date('%Y', date)) AS last\_purchase\_of\_year

FROM `prism\_test.Indigo-SC-W3-Basis`

GROUP BY user\_cookie\_id, date

),

temp3 AS

(

SELECT user\_cookie\_id, format\_date('%Y', date) AS year, last\_purchase\_of\_year

FROM temp2

GROUP BY year, user\_cookie\_id, last\_purchase\_of\_year

ORDER BY user\_cookie\_id

)

SELECT temp.\*, temp3. last\_purchase\_of\_year

FROM temp

JOIN temp3

ON temp.user\_cookie\_id = temp3.user\_cookie\_id AND temp.year = temp3.year

ORDER BY user\_cookie\_id

### Step 2.3:

Saved the result in a table called prism\_test.Indigo-SC-W3-Final

with t1 as (select \*,

sum(no\_transactions) over (partition by user\_cookie\_id order by last\_purchase\_of\_year) as running\_transactions,

sum(total\_revenue) over (partition by user\_cookie\_id order by last\_purchase\_of\_year) as running\_revenue,

from `prism\_test.Indigo-SC-W3-Seg`),

t2 as (select \*,

NTILE(5) OVER (PARTITION BY year ORDER BY last\_purchase\_of\_year ASC) AS R,

ntile(100) over (partition by year order by running\_transactions) as f\_precentile,

ntile(100) over (partition by year order by running\_revenue) as m\_precentile

from t1),

## giving f and m scores based on percentile

t3 as (

select \*,

case

when f\_precentile >= 95 then 5

when f\_precentile >= 80 then 4

when f\_precentile >= 70 then 3

when f\_precentile >= 40 then 2

else 1

end as F,

case

when m\_precentile >= 95 then 5

when m\_precentile >= 80 then 4

when m\_precentile >= 70 then 3

when m\_precentile >= 40 then 2

else 1

end as M

from t2),

## concatinating the rfm scores to one value and converting it into an integer

t4 as (

SELECT \*,

CAST(CONCAT(R,F,M) AS INT64) AS rfm

FROM t3),

## applying the segments

t5 AS (

select \*,

case

when rfm in (555, 554, 544, 545, 454, 455, 445) then "champions"

when rfm in (543, 444, 435, 355, 354, 345, 344, 335) then "loyal customer"

when rfm in (553, 551, 552, 541, 542, 533, 532, 531, 452, 451, 442, 441, 431, 453, 433, 432, 423, 353, 352, 351, 342, 341, 333, 323) then "potential loyalist"

when rfm in (512, 511, 422, 421, 412, 411, 311) then "new customer"

when rfm in (525, 524, 523, 522, 521, 515, 514, 513, 425, 424, 413, 414, 415, 315, 314, 313) then "promising"

when rfm in (535, 534, 443, 434, 343, 334, 325, 324) then "need attention"

when rfm in (331, 321, 312, 221, 213) then "about to sleep"

when rfm in (255, 254, 245, 244, 253, 252, 243, 242, 235, 234, 225, 224, 153, 152, 145, 143, 142, 135, 134, 133, 125, 124) then "at risk"

when rfm in (155, 154, 144, 214, 215, 115, 114, 113) then "can't lose them"

when rfm in (332, 322, 231, 241, 251, 233, 232, 223, 222, 132, 123, 122, 212, 211) then "hibernating"

when rfm in (111, 112, 121, 131, 141, 151) then "lost"

end as segment

from t4)

SELECT \*

FROM t5

ORDER BY user\_cookie\_id, year