Assume you're given tables with information on Snapchat users, including their ages and time spent sending and opening snaps.

Write a query to obtain a breakdown of the time spent sending vs. opening snaps as a percentage of total time spent on these activities grouped by age group. Round the percentage to 2 decimal places in the output.

with cte as

(SELECT a.user\_id, ab.age\_bucket, sum(time\_spent) as opening\_count

FROM activities a join age\_breakdown ab

on a.user\_id = ab.user\_id

where a.activity\_type = 'open'

group by a.user\_id, ab.age\_bucket),

cte2 as (

SELECT a.user\_id, sum(time\_spent) as sending\_count

FROM activities a join age\_breakdown ab

on a.user\_id = ab.user\_id

where a.activity\_type = 'send'

group by a.user\_id)

select cte.age\_bucket,

ROUND(SUM(cte2.sending\_count)

/ (SUM(cte.opening\_count) + SUM(cte2.sending\_count)) \* 100.0, 2) AS send\_perc,

ROUND(SUM(cte.opening\_count)

/ (SUM(cte.opening\_count) + SUM(cte2.sending\_count)) \* 100.0, 2) AS open\_prec

from cte join cte2

on cte.user\_id = cte2.user\_id

group by cte.age\_bucket;

//////////////////////////////////////////////////////////////////////

**--Q4 Which city has the best customers? We would like to throw a promotional Music**

**--Festival in the city** **we made the most money. Write a query that returns one city that**

**--has the highest sum of invoice totals. Return both the city name & sum of all invoice**

**--totals**

SELECT C1.CITY,

SUM(I1.TOTAL) AS TOTAL

FROM CUSTOMER C1

JOIN INVOICE I1 ON C1.CUSTOMER\_ID = I1.CUSTOMER\_ID

GROUP BY CITY

ORDER BY TOTAL DESC

LIMIT 1;

**--Q5 Who is the best customer? The customer who has spent the most money will be**

**--declared the best customer. Write a query that returns the person who has spent the most money**

SELECT C1.CUSTOMER\_ID,

SUM(I1.TOTAL) AS TOTAL\_SPENT

FROM CUSTOMER C1

JOIN INVOICE I1 ON C1.CUSTOMER\_ID = I1.CUSTOMER\_ID

GROUP BY C1.CUSTOMER\_ID

ORDER BY TOTAL\_SPENT DESC

LIMIT 1

**-Q6 Write query to return the email, first name, last name, & Genre of all Rock Music**

**--listeners. Return your list ordered alphabetically by email starting with A**

SELECT DISTINCT C1.EMAIL,

C1.FIRST\_NAME,

C1.LAST\_NAME,

G1.NAME AS GENRE\_NAME

FROM CUSTOMER C1

JOIN INVOICE I1 ON C1.CUSTOMER\_ID = I1.CUSTOMER\_ID

JOIN INVOICE\_LINE I2 ON I1.INVOICE\_ID = I2.INVOICE\_ID

JOIN TRACK T1 ON I2.TRACK\_ID = T1.TRACK\_ID

JOIN GENRE G1 ON T1.GENRE\_ID = G1.GENRE\_ID

WHERE G1.NAME = 'Rock'

ORDER BY EMAIL;

**--Q7 Let's invite the artists who have written the most rock music in our dataset. Write a**

**--query that returns the Artist name and total track count of the top 10 rock bannds.**

SELECT ARTIST.NAME,

COUNT(ALBUM.ARTIST\_ID) AS TOTAL\_SONGS

FROM ARTIST

JOIN ALBUM ON ARTIST.ARTIST\_ID = ALBUM.ARTIST\_ID

JOIN TRACK ON ALBUM.ALBUM\_ID = TRACK.ALBUM\_ID

WHERE TRACK\_ID IN

(SELECT TRACK\_ID

FROM TRACK

JOIN GENRE ON TRACK.GENRE\_ID = GENRE.GENRE\_ID

WHERE GENRE.NAME = 'Rock')

GROUP BY ARTIST.NAME

ORDER BY TOTAL\_SONGS DESC

LIMIT 10;

**--Q8 Return all the track names that have a song length longer than the average song length.**

**--Return the Name and Milliseconds for each track. Order by the song length with the**

**--longest songs listed first**

SELECT NAME,

MILLISECONDS

FROM TRACK

WHERE MILLISECONDS >

(SELECT AVG(MILLISECONDS)

FROM TRACK)

ORDER BY MILLISECONDS DESC;

SELECT \*

FROM TRACK;

**--Q9 Find how much amount spent by each customer on artists? Write a query to return**

**--customer name, artist name and total spent**

SELECT C1.FIRST\_NAME AS CUSTOMER\_NAME,

ARTIST.NAME AS ARTIST\_NAME,

SUM((I1.UNIT\_PRICE \* I1.QUANTITY))AS TOTAL\_SPENT

FROM INVOICE\_LINE I1

JOIN INVOICE ON I1.INVOICE\_ID = INVOICE.INVOICE\_ID

JOIN CUSTOMER C1 ON INVOICE.CUSTOMER\_ID = C1.CUSTOMER\_ID

JOIN TRACK ON I1.TRACK\_ID = TRACK.TRACK\_ID

JOIN ALBUM ON TRACK.ALBUM\_ID = ALBUM.ALBUM\_ID

JOIN ARTIST ON ALBUM.ARTIST\_ID = ARTIST.ARTIST\_ID

GROUP BY 1, 2

ORDER BY 3 DESC;

**--Q10 We want to find out the most popular music Genre for each country. We determine the**

**--most popular genre as the genre with the highest amount of purchases. Write a query**

**--that returns each country along with the top Genre. For countries where the maximum**

**--number of purchases is shared return all Genres**

SELECT GENRE\_NAME,

MAX(TOTAL\_PURCHASE)

FROM

(SELECT INVOICE.BILLING\_COUNTRY AS CON,

GENRE.NAME AS GENRE\_NAME,

COUNT(INVOICE\_LINE.QUANTITY) AS TOTAL\_PURCHASE

FROM INVOICE\_LINE

JOIN INVOICE ON INVOICE.INVOICE\_ID = INVOICE\_LINE.INVOICE\_ID

JOIN TRACK ON TRACK.TRACK\_ID = INVOICE\_LINE.TRACK\_ID

JOIN GENRE ON GENRE.GENRE\_ID = TRACK.GENRE\_ID

GROUP BY 1,2

ORDER BY 3 DESC)T1

GROUP BY 1

ORDER BY MAX(TOTAL\_PURCHASE)DESC;

SELECT \*

FROM INVOICE\_LINE;

**--Q11 Write a query that determines the customer that has spent the most on music for each**

**--country. Write a query that returns the country along with the top customer and how**

**--much they spent. For countries where the top amount spent is shared, provide all**

**--customers who spent this amount**

SELECT TEMP\_1.COUNTRY,

TEMP\_1.FIRST\_NAME,

TEMP\_1.TOTAL\_SPENT

FROM

(SELECT \*,

ROW\_NUMBER () OVER(PARTITION BY COUNTRY

ORDER BY TOTAL\_SPENT DESC) AS ROWNO

FROM

(SELECT C1.COUNTRY,

C1.FIRST\_NAME,

SUM(I1.TOTAL) AS TOTAL\_SPENT

FROM CUSTOMER C1

LEFT JOIN INVOICE I1 ON C1.CUSTOMER\_ID = I1.CUSTOMER\_ID

GROUP BY 1,

2

ORDER BY TOTAL\_SPENT DESC)TEMP)TEMP\_1

WHERE TEMP\_1.ROWNO = 1

ORDER BY COUNTRY,

FIRST\_NAME

/\* WINDWOS FUNTIONS \*/

***-- Write an SQL query that retrieves data from the 'Student' table. For each student, calculate both the maximum and average score within their respective departments.***

Select \* , Max(Score)

over (Partition by dep\_name) As Dep\_Max\_Score,

Round(Avg(Score))

over (Partition by dep\_name) As Dep\_Avg\_Score

from Student;

***-- Write a Sql Query to fetch Serial Numbers and Department Ranks for Students.***

SELECT

Student\_name AS "Student Name",

Score AS "Student Score",

ROW\_NUMBER() OVER (ORDER BY Student\_name) AS "Serial Number",

DENSE\_RANK() OVER (PARTITION BY dep\_name ORDER BY Score) AS "Department Rank"

FROM Student\_Score

ORDER BY "Department Rank", "Serial Number";

***-- Write an SQL query that retrieves the time difference between 2 trains within the same 'train\_ID' group.***

Select train\_ID, Station, time as “station Time”,

lead [time] over (Partition by train\_ID order by time) - time As time\_to\_next\_train

from train\_schedule;