Question 1:

You have three tables:

Customers table:

https://drive.google.com/file/d/17oRPW3jA2mCsTOXKKt6Xx7wSgD78twq8/view?usp=sharing

Orders table:

https://drive.google.com/file/d/1TWBB9rqGRnWOJsDcwkUE2HkJNEafawQm/view?usp=sharing

Payments table:

https://drive.google.com/file/d/1vakf\_7F9cb7ZKOeCofyI370Paiswg3JX/view?usp=sharing

A.Which payment\_type has the fastest average delivery day where delivery day can be found by using order\_delivered\_customer\_date and order\_purchase\_timestamp columns? You should be careful about choosing these dates that shouldn’t be null.

(Hint: The result should show only one payment\_type with the average\_delivery\_day info)

SELECT

payment\_type,

AVG(DATE\_DIFF(order\_delivered\_customer\_date,order\_purchase\_timestamp,DAY)) AS AVG\_DELIVERY\_DAY

FROM bootcamp2022-372013.final.ORDERS

FULL JOIN bootcamp2022-372013.final.PAYMENTS AS PAY

USING(order\_id)

WHERE order\_delivered\_customer\_date IS NOT NULL AND order\_purchase\_timestamp IS NOT NULL

GROUP BY payment\_type

B.What is the maximum payment\_value of each state where order status is neither unavailable nor canceled? Additionally, the max\_payment\_revenue column should have 1 decimal point.

(Hint: The result table will only have state and max\_payment\_revenue columns)

SELECT

customer\_state,

ROUND(MAX(payment\_value),1) AS max\_payment\_revenue

FROM bootcamp2022-372013.final.CUSTOMERS AS cust

FULL JOIN bootcamp2022-372013.final.ORDERS AS orders

USING(customer\_id)

FULL JOIN bootcamp2022-372013.final.PAYMENTS AS payments

USING(order\_id)

WHERE order\_status != 'unavailable' AND order\_status != 'canceled'

GROUP BY customer\_state;

C.In the payments table, how many distinct order\_ids that exist more than once?

WITH NUM\_CITY AS(

SELECT DISTINCT

order\_id,

COUNT(order\_id) OVER(PARTITION BY order\_id ORDER BY order\_id) AS NUM\_OF\_ORDER

FROM bootcamp2022-372013.final.PAYMENTS

)

SELECT

order\_id,

NUM\_OF\_ORDER

FROM NUM\_CITY

WHERE NUM\_OF\_ORDER>1

D.For each row, find the payment\_value of previous row and leading row in the partition of order\_id and order of payment\_sequential ascendingly.(Hint: The result table should have all columns in the payments table and two new columns:previous\_payment\_value and leading\_payment\_value.Also, we don’t want null values for these two new fields)

WITH VALUES AS (

SELECT

\*,

LAG(payment\_value) OVER(PARTITION BY order\_id ORDER BY payment\_sequential ASC) AS PREVIOUS\_PAYMENT\_VALUE,

LEAD(payment\_value) OVER(PARTITION BY order\_id ORDER BY payment\_sequential ASC) AS LEADING\_PAYMENT\_VALUE

FROM bootcamp2022-372013.final.PAYMENTS

)

SELECT

\*

FROM VALUES

WHERE PREVIOUS\_PAYMENT\_VALUE IS NOT NULL AND LEADING\_PAYMENT\_VALUE IS NOT NULL

E.Get all the columns in the customers table and add three new columns:

\* order\_id

\* order\_purchase\_date (Hint: convert the timestamp to the date value)

\* payment\_value

\* payment\_value\_int (Hint: payment\_value but in the integer format. You should use a function

to convert it type)

SELECT

C.\*,

O.order\_id,

DATE(O.order\_purchase\_timestamp) AS order\_purchase\_date,

P.payment\_value,

CAST(payment\_value AS INTEGER) AS payment\_value\_int

FROM bootcamp2022-372013.final.CUSTOMERS AS C

LEFT JOIN bootcamp2022-372013.final.ORDERS O

USING (customer\_id)

LEFT JOIN bootcamp2022-372013.final.PAYMENTS AS P

USING (order\_id)

F.Get all columns from the payments table excluding the payment\_type column, and create payment\_method column instead of that. This new column should have this format:

If payment\_type is “not\_defined”, then payment\_method will be 0

If payment\_type is “credit\_card”, then payment\_method will be 1

If payment\_type is “voucher”, then payment\_method will be 2

If payment\_type is “debit\_card”, then payment\_method will be 3

If payment\_type is “boleto”, then payment\_method will be 4

SELECT

\* EXCEPT(payment\_type),

CASE

WHEN payment\_type = 'not\_defined' THEN 0

WHEN payment\_type = 'credit\_card' THEN 1

WHEN payment\_type = 'voucher' THEN 2

WHEN payment\_type = 'debit\_card' THEN 3

WHEN payment\_type = 'boleto' THEN 4

ELSE NULL

END AS payment\_method

FROM bootcamp2022-372013.final.PAYMENTS

**Question 2:**

You have one table:

Netflix\_movies: https://drive.google.com/file/d/1VRdGqa3KmgDj\_BEX4\_OBEtDa-JRmE1LH/view? usp=sharing

A.How many times each word is used in the table, that you get when you split the movie\_name by each space?  
(Hint: The result table only have 2 columns: words\_used\_in\_movie\_names, number\_of\_occurence.  
Ex. You can imagine that asks for how many times you see Rome word in this table)

WITH UNNEST\_WORDS AS (

SELECT

movie\_name,

WORDS

FROM bootcamp2022-372013.final.netflix,UNNEST(REGEXP\_EXTRACT\_ALL(movie\_name,r"(\w+)")) WORDS

)

SELECT

WORDS,

COUNT(\*) AS count

FROM UNNEST\_WORDS

GROUP BY WORDS

ORDER BY count DESC

B.If a movie\_name has “**Vol. 1**”, then change it to “Part 1”; if has “**Vol. 2**”, then change it to “Part. 2”. Else movie\_name. After these changes, name show this column as “movie\_name” instead of showing the original movie\_name column.

SELECT

REGEXP\_REPLACE(REGEXP\_REPLACE(movie\_name, 'Vol. 1', 'Part 1'), 'Vol. 2', 'Part 2') AS movie\_name

FROM bootcamp2022-372013.final.netflix

C.In this table, we have some movie\_names that occur multiple times. Create a new column in the result table, call this column as “**previous\_year**” and show the previous year of each movie if they exist multiple times in that table. In the result table, there shouldn’t be any null previous\_year values.

WITH year\_table AS(

SELECT

movie\_name,

lag(year) over(partition by movie\_name order by year) as previous\_year

FROM bootcamp2022-372013.final.netflix

)

SELECT

movie\_name,

previous\_year

FROM year\_table

WHERE previous\_year IS NOT NULL

ORDER BY movie\_name