Jeremy Obach

CareerFoundry DA Immersion

Task 3.8

1a.

SELECT total\_amount\_paid.first\_name,

total\_amount\_paid.last\_name,

AVG(total\_amount\_paid.amount) AS average

FROM

(SELECT A.customer\_id,

A.first\_name,

A.last\_name,

C.city,

D.country,

SUM(E.amount) AS amount

FROM customer A

INNER JOIN address B ON A.address\_id = B.address\_id

INNER JOIN city C ON B.city\_id = C.city\_id

INNER JOIN country D ON C.country\_id = D.country\_id

INNER JOIN payment E on A.customer\_id = E.customer\_id

WHERE country IN( 'India',

'China',

'United States',

'Japan',

'Mexico',

'Brazil',

'Russian Federation',

'Philippines',

'Turkey',

'Indonesia')

AND city IN ('Aurora',

'Atlixco',

'Xintai',

'Adoni',

'Dhule (Dhulia)'

'Kurashiki'

'Pingxiang'

'Sivas'

'Celaya'

'So Leopoldo')

GROUP BY A.customer\_id,

first\_name,

last\_name,

city,

country

ORDER BY SUM(E.amount)DESC

LIMIT 5) AS total\_amount\_paid -- subquery renamed total\_amount\_paid

GROUP BY total\_amount\_paid.customer\_id,

total\_amount\_paid.first\_name,

total\_amount\_paid.last\_name

1b.

First\_name last\_name average

"Sara" "Perry" 128.7000000000000000

"Scott" "Shelley" 60.8200000000000000

"Adam" "Gooch" 97.8000000000000000

"Clinton" "Buford" 98.7600000000000000

"Bob" "Pfeiffer" 82.7800000000000000

2a.

SELECT D.country,

COUNT(A.customer\_id) AS all\_customer\_count,

COUNT(top\_5\_customer) AS top\_customer\_count

FROM customer A

INNER JOIN address B ON A.address\_id = B.address\_id

INNER JOIN city C ON B.city\_id = C.city\_id

INNER JOIN country D ON C.country\_id = D.country\_id

LEFT JOIN

(SELECT A.customer\_id,

A.first\_name,

A.last\_name,

C.city,

D.country,

SUM(E.amount) AS total\_amount\_paid

FROM customer A

INNER JOIN address B ON A.address\_id = B.address\_id

INNER JOIN city C ON B.city\_id = C.city\_id

INNER JOIN country D ON C.country\_id = D.country\_id

INNER JOIN payment E on A.customer\_id = E.customer\_id

WHERE country IN( 'India',

'China',

'United States',

'Japan',

'Mexico',

'Brazil',

'Russian Federation',

'Philippines',

'Turkey',

'Indonesia')

AND city IN ('Aurora',

'Atlixco',

'Xintai',

'Adoni',

'Dhule (Dhulia)'

'Kurashiki'

'Pingxiang'

'Sivas'

'Celaya'

'So Leopoldo')

GROUP BY A.customer\_id,

A.first\_name,

A.last\_name,

C.city,

D.country

ORDER BY total\_amount\_paid DESC

LIMIT 5)top\_5\_customer ON A.customer\_id = top\_5\_customer.customer\_id

GROUP BY D.country

ORDER BY top\_customer\_count DESC

LIMIT 5;

2b.

"United States" 36 2

"Mexico" 30 1

"India" 60 1

"China" 53 1

"Faroe Islands" 1 0

3a.

Step 1 could have been achieved without subqueries by linking the payment table to the customer table, but it would be step by step process to determine the top cities and countries along the way. In that sense I can see the value of the subquery essentially saving the progress of a previous query, but the tradeoff in procssing speed or readability could hamper those advantages depending on the situation. Likewise, the second question would still require multiple joins to link the payment table to the customer table to the address table and so on, but where the subquery came in handy is that it already captured our desired parameters for only the top 5 customers from our last task.

Therefore, subqueries are useful in providing a “shorthand” for a certain condition, like our top 5 customers, that is dynamic by nature, and likely to change. I see it as a tool in our tool belt – appropriate for specific conditions, and overkill or just not suitable for others.