Jeremy Obach
CareerFoundry DA Immersion
Task 3.9

1a. Query:

```
WITH top_5_customer_cte (customer_id, first_name, last_name, city, country, amount_paid) AS
	(SELECT A.customer_id,
		A.first_name,
		A.last_name,
		C.city,
		D.country,
		SUM(E.amount) AS 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,
			first_name,
			last_name,
			city,
			country
	ORDER BY amount_paid DESC
	LIMIT 5)
```

```
SELECT customer_id,
        first_name,
        last_name,
        city,
        country,
        amount_paid
FROM top_5_customer_cte
GROUP BY 1,2,3,4,5,6
```

Output:

"customer_id"	"first_name"	"last_name"	"city"	"country"	"amount_paid"
84	"Sara" "	Perry"	"Atlixco"	"Mexico"	128.70
330	"Scott"	"Shelley"	"Aurora"	"United States"	60.82
367	"Adam"	"Gooch"	"Adoni"	"India"	97.80
537	"Clinton"	"Buford"	"Aurora"	"United States"	98.76
564	"Bob"	"Pfeiffer"	"Xintai"	"China"	82.78

1b. Query:

```
WITH top_5_customer_CTE (customer_id, first_name, last_name, country, city,
total_amount_paid) AS (


        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,
                    D. country,
                    C. city
            ORDER BY total_amount_paid DESC
            LIMIT 5)

SELECT D. country,
        COUNT (A.customer_id) AS total_customer_count,
        COUNT (top_5_customer_CTE) 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 top_5_customer_cte ON A.customer_id = top_5_customer_cte.customer_id
GROUP BY D.country
ORDER BY top_customer_count DESC
LIMIT 5;
```

Output:

"country"	"total_customer_count"	"top_customer_count"
"United States"	36	2
"Mexico"	30	1
"India"	60	1
"China"	53	1
"Faroe Islands"	1	0

1c. First, I copied an entire query from my 3.8 Answers sheet into PG Admin's query tool. Then I wrote the WITH…AS main statement required for Subqueries , careful to match the number of categories to the number selected in the CTE. Then I include the entire statement after AS, and

closed parentheses. Now, with the CTE set up, I can select FROM it, and I tried using numbers in lieu of row names for the GROUP BY.

That's how the first one went at least, it was the more straightforward adaptation of the two. For the second one I ran into some issues at first. I basically had to rewrite a bunch of the second part, with the list of inner joins and left join at the end, but really I could have pulled it from the front of the second query. It required a little more reordering and renaming, but I ultimately was able to get the same result as the subquery method.

2a. Pre-trial performance estimation
I'm going to venture a guess that sub-querying will be faster because CTE must make a table first and then reference it, versus a subquery which must run the entire process every time. Maybe on initial running at least, and then CTE's have the edge on repeat inquiries.

2b. Query Plan Comparison
3.8.1
"QUERY PLAN"
"GroupAggregate (cost=27.47..27.60 rows=5 width=49)"

3.8.2
"QUERY PLAN"
"Limit (cost=100.47..100.48 rows=5 width=25)"

3.9.1
"QUERY PLAN"
"Group (cost=27.42..27.51 rows=5 width=67)"

3.9.2
"QUERY PLAN"
"Limit (cost=100.47..100.48 rows=5 width=25)"

Copied the entire EXPLAIN outputs at first, will try to interpret the first lines instead. Appears that the two versions across 3.8 and 3.9 have more in common with each other than the two subqueries and two CTE's, which makes sense as they're essentially doing the same thing in a different format. The queries finding how many customers in each country/ how many top 5 customers in each country is almost 4 times more costly than the other queries only trying to identify the top 5 customers. For the latter pair, the CTE has the slight cost edge, 27.42 vs 27.47, but also has more width at 67 vs 49. For the former pair, the CTE and subquery have apparently identical costs. Makes me think I might not be reading the QUERY PLAN correctly. Anyways, let's run the actual queries and compare run speed.

2c. Actual Query run time comparison

3.8.1
88 msec / 44 msec / 49 msec

3.8.2
83 msec / 69 msec / 67 msec

3.9.1
62 msec / 60 msec/ 63 msec

3.9.2
78 msec / 69 msec / 72 msec

These results are surprising, especially the seemingly wide variance between different runs for the subquery method queries. The CTEs were more consistent and had a slightly smaller spread, in the case of the top 5 customers, a much smaller spread. It all seems a bit random to me, I wonder how many factors go into the speed – computer processing demand at the time, the length of time between runs, etc. Based on this data it's difficult to give CTE a definitive performance edge, so any benefits toward it would have to be based on readability.

3. Challenges of replaceing subqueries with CTE's.

I touched on this on the step by step method explanation but it seems to vary based on the query how difficult or easy the adapatation from a subquery to a CTE will be. The first set, finding the top 5 customers, was relatively straightforward and only took a few steps to move the subquery into a CTE and draw from it accordingly. The second set, finding the number of top 5 customers in each country, was more challenging as my initial intuition as to what ought to go where was off-base, and led me to a reoccurring error about including the D.country column in GROUP BY even though it appeared to be there. Basically, I needed to create the CTE with the same main categories as the subquery, and save the new points of interest, namely country, and customer count aggregate functions for the SELECT function AFTER the entire CTE is dictated. It's straightforward in hindsight but I had to walk away from it for a day and come back to it to make sense of it. I'm still working toward comfortability and a deeper understanding with these adaptations.

SPOTLIGHT ON AI:

1. Rewriting subqueries into CTE's with ChatGPT was really easy, much easier than converting them by hand. I just used this prompt: "Rewrite this SQL query using a CTE instead of subquery" and then copied the prompt from 3.8.1. Converting it first by hand definitely aided my understanding of the process though. Worked just as easily and seamlessly for 3.8.2. What a tool for the tool belt.

2. The outputs for my conversions and chatGPT's suggestions were not identical and I'm not positive who's in error but I'm inclined to suspect chatGPT's results because both of my queries for the top 5 countries yielded 2 for the US, and 1 for Mexico, China, and India. ChatGPT's query yielded 2 for Mexico, 1 for Turkey(?),the US and India. Not sure where the variance would be introduced to be honest, the prompts looked materially the same – to the point that I started writing this response under the assumption that they yielded identical results before really looking at the outputs carefully.
3. This exercise absolutely would have saved me time and has definitely aided in my learning, but there's still some lingering distrust/skepticism which I think is appropriate given the subtle divergence in results from the top 5 customers query. Definitely a tool I would use selectively in the future, being careful not to trust the results immediately and to acknowledge that like many things, it's going to be garbage in/ garbage out so I should be extra cautious with my inputs.