Andrew Smith

Career Foundry Data Immersion

January 18, 2022

3.8: Performing Subqueries (Answers 3.8)

1)

SELECT AVG(total\_amount\_paid) AS average

FROM (SELECT A.customer\_id, A.first\_name, A.last\_name, C.city, D.country, SUM(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 C.city IN

(SELECT C.city

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

WHERE D.country IN

(SELECT D.country

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

GROUP BY D.country

ORDER BY COUNT(customer\_id) DESC

LIMIT 10)

GROUP BY D.country, C.city

ORDER BY COUNT(customer\_id) DESC

LIMIT 10)

GROUP BY A.customer\_id, A.first\_name, A.last\_name, C.city, D.country

ORDER BY sum(amount) DESC

LIMIT 5) AS total\_amount\_paid

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

2)

SELECT D.country,

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

COUNT(DISTINCT top\_5\_customers.customer\_id) AS top\_customer\_count

FROM country D

INNER JOIN city C on D.country\_id = C.country\_id

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

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

LEFT JOIN (SELECT A.customer\_id, A.first\_name, A.last\_name, C.city, D.country, SUM(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 C.city IN

(SELECT C.city

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

WHERE D.country IN

(SELECT D.country

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

GROUP BY D.country

ORDER BY COUNT(customer\_id) DESC

LIMIT 10)

GROUP BY D.country, C.city

ORDER BY COUNT(customer\_id) DESC

LIMIT 10)

GROUP BY A.customer\_id, A.first\_name, A.last\_name, C.city, D.country

ORDER BY sum(amount) DESC

LIMIT 5) AS top\_5\_customers

ON D.country = top\_5\_customers.country

GROUP BY D.country, top\_5\_customers

ORDER BY all\_customer\_count DESC

LIMIT 5

Graphical user interface, application

Description automatically generated

Table

Description automatically generated

3a) Yes, I think steps 1 and 2 could be done without subqueries but it would take much longer since multiple queries would have to be typed individually and then combined to make a super lengthy code. In addition, it would save time from manually updating queries since many databases change over time.

3b) I think subqueries are helpful when information is needed from many queries in multiple tables from larger datasets. I also think if the database is constantly being updated, then subqueries would save someone a lot of time from manually changing the commands or running separate queries.