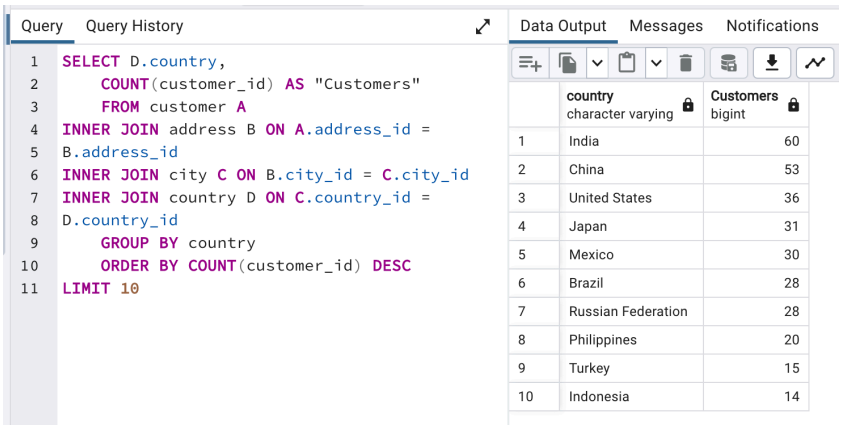
E. Christian Cotterman
Thursday, 1. June 2023

CF.Ex.3.7

Answers

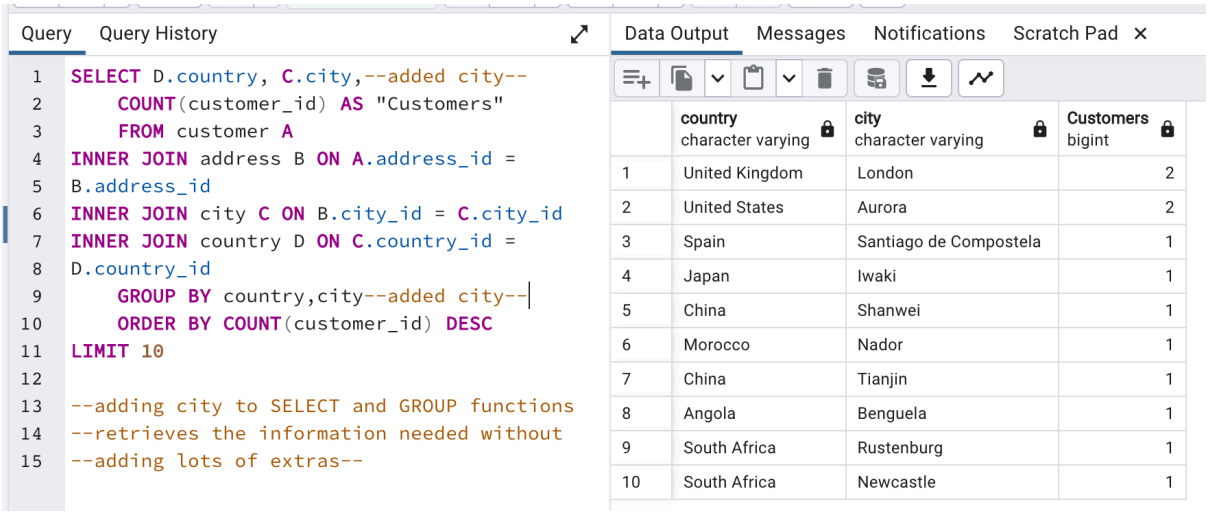
- 1. We need data from Customer, Address, City, Country (*in that reversed order*). We will use a **INNER JOIN** to get the overlapping information and **GROUP BY** and **ORDER BY** to get the highest number organised by country.
  INNER JOIN allows us to only combine and filter out the data that is contained in all 4 of those tables.



```
SELECT D.country,
    COUNT(customer_id) AS "Customers"
    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 country
    ORDER BY COUNT(customer_id) DESC
LIMIT 10
```

	country character varying	Customers bigint
1	India	60
2	China	53
3	United States	36
4	Japan	31
5	Mexico	30
6	Brazil	28
7	Russian Federation	28
8	Philippines	20
9	Turkey	15
10	Indonesia	14

- 2.Same query as above, but added **C.city** to the **SELECT** and **GROUP BY** commands to filter the information.



```
SELECT D.country, C.city,--added city--
    COUNT(customer_id) AS "Customers"
    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 country,city--added city--
    ORDER BY COUNT(customer_id) DESC
LIMIT 10

--adding city to SELECT and GROUP functions
--retrieves the information needed without
--adding lots of extras--
```

	country character varying	city character varying	Customers bigint
1	United Kingdom	London	2
2	United States	Aurora	2
3	Spain	Santiago de Compostela	1
4	Japan	Iwaki	1
5	China	Shanwei	1
6	Morocco	Nador	1
7	China	Tianjin	1
8	Angola	Benguela	1
9	South Africa	Rustenburg	1
10	South Africa	Newcastle	1

The KISS rule applies for me here, because writing or adding more information to the query can make it unwieldy and not as clean. This retains readability and keeps query times low. (*Added comments after it ran successfully.*)

- 3. Building upon the last two queries, we can add in the first_name and last_name in our **SELECT** function to ensure we know who our top customers are. We use the **SUM** function to get the total payments for the customers and later, we use the **GROUP** and **ORDER** functions to group our data and order it by amounts paid in descending value.

(*I personally don't like having E thrown in at the end like this. I didn't want to modify the query too much, and this is really an OCD thing for me. It is the same as having the code readable like HTML or CSS, which I have some small experience in. It's what I mentioned in the last exercise, where I want things to look readable, and still be functional.*)

Query | Query History

```
1  SELECT A.first_name AS "Name", A.last_name AS "Last Name", D.country, C.city, --added name queries
2      SUM(E.amount) AS "Total payments" --Sum of total payments per customer
3      FROM customer A
4  INNER JOIN address B ON A.address_id = B.address_id
5  INNER JOIN city C ON B.city_id = C.city_id
6  INNER JOIN country D ON C.country_id =D.country_id
7  INNER JOIN payment E ON A.customer_id = E.customer_id --I would prefer A,B,C,D,E
8      WHERE city IN ('Aurora','Acua','Citrus Heights','Iwaki','Ambattur','Shanwei','So
9  Leopoldo','Teboksary','Tianjin','Cianjur')
10     GROUP BY first_name,last_name,country,city --added first and last names--
11     ORDER BY "Total payments" DESC
12
```

Data Output | Messages | Notifications | Scratch Pad ×

	Name character varying	Last Name character varying	country character varying	city character varying	Total payments numeric
1	Arlene	Harvey	India	Ambattur	111.76
2	Kyle	Spurlock	China	Shanwei	109.71
3	Marlene	Welch	Japan	Iwaki	106.77
4	Glen	Talbert	Mexico	Acua	100.77
5	Clinton	Buford	United States	Aurora	98.76
6	Betty	White	United States	Citrus Heights	96.77
7	Dora	Medina	China	Tianjin	88.81
8	Norman	Currier	Indonesia	Cianjur	73.76
9	Juan	Fraley	Russian Federation	Teboksary	63.79
10	Scott	Shelley	United States	Aurora	60.82