

Databases & SQL for Analysts







3.9: Common Table Expressions

ANSWERS

Step 1

CTE allows us to create temporary tables to use, after creating them we just choose columns from those tables. Using CTE to query the first question which is “average top 5 customers amount”, first step we need to know the customers details. The query as below:

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,
city,country)AS(
SELECT a.customer_id,
       a.first_name,
       a.last_name,
       c.city_id,
       c.city,
       d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id)
SELECT * FROM customer_detail
```

		 first_name character varying (45)	 last_name character varying (45)	 city_id integer	 city character varying (50)	 country character varying (50)	
1	24	Jared	Ely	419	Purwakarta	Indonesia	
2	1	Mary	Smith	463	Sasebo	Japan	
3	2	Patricia	Johnson	449	San Bernardino	United States	
4	3	Linda	Williams	38	Athenai	Greece	
5	4	Barbara	Jones	349	Myingyan	Myanmar	
6	5	Elizabeth	Brown	361	Nantou	Taiwan	
7	6	Jennifer	Davis	295	Laredo	United States	
8	7	Maria	Miller	280	Kragujevac	Yugoslavia	
9	8	Susan	Wilson	200	Hamilton	New Zealand	
10	9	Margaret	Moore	329	Masqat	Oman	
11	10	Dorothy	Taylor	162	Esfahan	Iran	
12	11	Lisa	Anderson	440	Sagamihara	Japan	
13	12	Nancy	Thomas	582	Yamuna Nagar	India	
14	13	Karen	Jackson	384	Osmaniye	Turkey	
15	14	Betty	White	120	Citrus Heights	United States	

Next, we need to find out the top10 countries that holds the most customer numbers,

Which allow us to create the second CTE called "top10_countries", there is a connection between these two CTES that I used the first CTE's result to query the second CTE. This is impressive.

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,
city,country)AS(
SELECT a.customer_id,
      a.first_name,
      a.last_name,
      c.city_id,
      c.city,
      d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
      count(*) AS customer_number
  FROM customer_detail
 GROUP BY country
 ORDER BY customer_number DESC
 LIMIT 10)
SELECT * FROM top10_countries
```

	country character varying (50)	customer_number 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

Following by next, we use the same step to query the top10 cities that in these top 10 countries.

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,
city,country)AS(
SELECT a.customer_id,
      a.first_name,
      a.last_name,
      c.city_id,
      c.city,
      d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
      count(*) AS customer_number
  FROM customer_detail
 GROUP BY country
 ORDER BY customer_number DESC
 LIMIT 10),
top10_cities(city)AS(
SELECT city,
      COUNT(1)AS customer_number
  FROM customer_detail
 WHERE country IN (SELECT country FROM top10_countries)
 GROUP BY city
 ORDER BY customer_number DESC
 LIMIT 10
)
SELECT* FROM top10_cities
```

	city character varying (50) 🔒	customer_number bigint 🔒
1	Aurora	2
2	Tokat	1
3	Tarsus	1
4	Atlixco	1
5	Emeishan	1
6	Pontianak	1
7	Shimoga	1
8	Aparecida de Goinia	1
9	Zalantun	1
10	Taguig	1

Then it's the time to get the top5 customers total payment where those 5 customers are in the top 10 cities.

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,city,country)AS(
SELECT a.customer_id,
       a.first_name,
       a.last_name,
       c.city_id,
       c.city,
       d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
       count(*) AS customer_number
  FROM customer_detail
 GROUP BY country
 ORDER BY customer_number DESC
 LIMIT 10),
top10_cities(city)AS(
SELECT city,
       COUNT(1)AS customer_number
  FROM customer_detail
 WHERE country IN (SELECT country FROM top10_countries)
 GROUP BY city
 ORDER BY customer_number DESC
 LIMIT 10
),
top5_customers(customer_id,first_name,last_name,city,|country,total_amount)
AS(
SELECT c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country,
       SUM(amount)AS total_pay
  FROM customer_detail AS c
 INNER JOIN payment AS p
    ON c.customer_id=p.customer_id
 WHERE c.city IN(SELECT city FROM top10_cities)
    AND c.country IN (SELECT country FROM top10_countries)
 GROUP BY
       c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country
 ORDER BY total_pay DESC
 LIMIT 5)
SELECT * FROM top5_customers
```

	customer_id integer	first_name character varying (45)	last_name character varying (45)	city character varying (50)	country character varying (50)	total_amount numeric
1	566	Casey	Mena	Tokat	Turkey	130.68
2	84	Sara	Perry	Atlixco	Mexico	128.7
3	506	Leslie	Seward	Pontianak	Indonesia	123.72
4	389	Alan	Kahn	Emeishan	China	119.75
5	537	Clinton	Buford	Aurora	United States	98.76

Finally, get the average payment from this top 5 customers.

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,city,country)AS(
SELECT a.customer_id,
      a.first_name,
      a.last_name,
      c.city_id,
      c.city,
      d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
      count(*) AS customer_number
  FROM customer_detail
 GROUP BY country
 ORDER BY customer_number DESC
 LIMIT 10),
top10_cities(city)AS(
SELECT city,
      COUNT(1)AS customer_number
  FROM customer_detail
 WHERE country IN (SELECT country FROM top10_countries)
 GROUP BY city
 ORDER BY customer_number DESC
 LIMIT 10
),
top5_customers(customer_id,first_name,last_name,city,country,total_amount)
AS(
SELECT c.customer_id,
      c.first_name,
      c.last_name,
      c.city,
      c.country,
      SUM(amount)AS total_pay
  FROM customer_detail AS c
 INNER JOIN payment AS p
    ON c.customer_id=p.customer_id
 WHERE c.city IN(SELECT city FROM top10_cities)
      AND c.country IN (SELECT country FROM top10_countries)
 GROUP BY
      c.customer_id,
      c.first_name,
      c.last_name,
      c.city,
      c.country
 ORDER BY total_pay DESC
 LIMIT 5)

SELECT AVG(total_amount)AS avg_top5_payment
  FROM top5_customers
```

	avg_top5_payment numeric	
1	120.322	

The second question to find how many top 5 are based in top 10 cities, the first four steps are the same as question 1.

```
WITH
customer_detail(customer_id,first_name,last_name,city_id,city,country)AS
(
SELECT a.customer_id,
       a.first_name,
       a.last_name,
       c.city_id,
       c.city,
       d.country
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
       count(*) AS customer_number
  FROM customer_detail
 GROUP BY country
 ORDER BY customer_number DESC
 LIMIT 10),
top10_cities(city)AS(
SELECT city,
       COUNT(1)AS customer_number
  FROM customer_detail
 WHERE country IN (SELECT country FROM top10_countries)
 GROUP BY city
 ORDER BY customer_number DESC
 LIMIT 10
),
top5_customers(customer_id,first_name,last_name,city,country,total_amount)AS(
SELECT c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country,
       SUM(amount)AS total_pay
  FROM customer_detail AS c
 INNER JOIN payment AS p
    ON c.customer_id=p.customer_id
 WHERE c.city IN(SELECT city FROM top10_cities)
       AND c.country IN (SELECT country FROM top10_countries)
 GROUP BY
       c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country
 ORDER BY total_pay DESC
 LIMIT 5)
```

Next, we use those CTES as left join, then the main query left join these CTES:

```
SELECT d.country,
       COUNT(DISTINCT a.customer_id) AS total_customer_count,
       COUNT(DISTINCT(d.country))AS top_customer_count
  FROM customer AS a
 INNER JOIN address AS b
    ON a.address_id=b.address_id
 INNER JOIN city AS c
    ON b.city_id=c.city_id
 INNER JOIN country AS d
    ON c.country_id=d.country_id
 LEFT JOIN top5_customers AS e
    ON d.country=e.country
 GROUP BY d.country
 ORDER BY 2 DESC
 LIMIT 5
```

From above query we can see the left join equal our outer query to join with top5_customers table, which we have already got from the first four steps without any further query or CTE. The final query as shows below:

```

WITH
customer_detail(customer_id,first_name,last_name,city_id,city,country)AS
(
SELECT a.customer_id,
       a.first_name,
       a.last_name,
       c.city_id,
       c.city,
       d.country
FROM   customer AS a
INNER JOIN address AS b
ON     a.address_id=b.address_id
INNER JOIN city AS c
ON     b.city_id=c.city_id
INNER JOIN country AS d
ON     c.country_id=d.country_id),
top10_countries(country)AS(
SELECT country,
       count(*) AS customer_number
FROM   customer_detail
GROUP BY country
ORDER BY customer_number DESC
LIMIT 10),
top10_cities(city)AS(
SELECT city,
       count(1)AS customer_number
FROM   customer_detail
WHERE  country IN (SELECT country FROM top10_countries)
GROUP BY city
ORDER BY customer_number DESC
LIMIT 10
),
top5_customers(customer_id,first_name,last_name,city,country,total_amount)AS(
SELECT c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country,
       sum(amount)AS total_pay
FROM   customer_detail AS c
INNER JOIN payment AS p
ON     c.customer_id=p.customer_id
WHERE  c.city IN(SELECT city FROM top10_cities)
AND    c.country IN (SELECT country FROM top10_countries)
GROUP BY
       c.customer_id,
       c.first_name,
       c.last_name,
       c.city,
       c.country
ORDER BY total_pay DESC
LIMIT 5)

SELECT d.country,
       count(DISTINCT a.customer_id) AS total_customer_count,
       count(DISTINCT(d.country))AS top_customer_count
FROM   customer AS a
INNER JOIN address AS b
ON     a.address_id=b.address_id
INNER JOIN city AS c
ON     b.city_id=c.city_id
INNER JOIN country AS d
ON     c.country_id=d.country_id
LEFT JOIN top5_customers AS e
ON     d.country=e.country
GROUP BY d.country
ORDER BY total_customer_count DESC
LIMIT 5

```

	country character varying (50) 🔒	total_customer_count bigint 🔒	top_customer_count bigint 🔒
1	India	60	1
2	China	53	1
3	United States	36	1
4	Japan	31	1
5	Mexico	30	1

Step 2:

Cost of average top 5 payment

Subquery:

	QUERY PLAN text
1	Aggregate (cost=65.71..65.72 rows=1 width=32)

CTE:

	QUERY PLAN text	
1	Aggregate (cost=119.41..119.42 rows=1 width=32)	🔒

The subquery costs less than CTE for the average payment because in the where condition we put ten cities' name in instead of writing another query, that's why subquery has less rows compare with CTE. CTE allows us to query the result in one time instead of putting cities' name.

Cost of top 5 customers number based on top 10 cities

Subquery:

	QUERY PLAN text
1	Limit (cost=181.20..181.21 rows=5 width=82)

CTE:

	QUERY PLAN text
1	Limit (cost=221.18..221.20 rows=5 width=25)

Subquery still costs less than CTE, it has the same situation as average top 5 payment.

For myself, I definitely choose CTE to write complex and long queries as these twos. Comparing with subquery, CTE here provides easily readable and clearly step by step. With every CTE table, we basically know what it indicates for. The most important thing is with CTE, I can clearly to know what should do by the next step till I totally finish the whole query.

Step 3:

At first, I was stucked by how to connect with each CTE I wrote. I didn't realize the first CTE can be used by the second CTE and so on, till I tried many times. From this case, I totally understand why CTE called a temporary table to allow you query data. The first CTE table would fit the final answer we need to query, like what columns will be selected from this table is important, because the following CTE tables will use them oftenly. We can not query without unknown columns, because CTE is already a table, we cannot choose any columns from a table that doesn't exist. I was surprised by CTE that it helps us to query the data in a long-detailed and clearly steps.