Queries to create tables

CREATE TABLE userinfo

(user\_id INT PRIMARY KEY,

first\_name VARCHAR (50) NOT NULL,

last\_name VARCHAR (50) NOT NULL,

email VARCHAR (320),

contact VARCHAR(50),

gender VARCHAR(10) NOT NULL,

age INT NOT NULL,

birthdate DATE NOT NULL,

lastlogin TIMESTAMP NOT NULL,

registration\_date DATE NOT NULL);

CREATE TABLE geographic

(City\_id INT PRIMARY KEY,

City\_name VARCHAR(50) NOT NULL,

State VARCHAR(50) NOT NULL,

Country VARCHAR(50),

Latitude DECIMAL (9,4) NOT NULL,

Longitude DECIMAL (9,4) NOT NULL,

Delivery\_zone VARCHAR(100));

CREATE TABLE restaurant

(restaurant\_id INT PRIMARY KEY,

city\_id INT NOT NULL,

restaurant\_name VARCHAR(100) NOT NULL,

cuisine\_type VARCHAR(50) NOT NULL,

address VARCHAR(300),

average\_rating DECIMAL(3,1) NOT NULL,

opening\_hours VARCHAR(100),

delivery\_time TIME NOT NULL,

FOREIGN KEY (city\_id) REFERENCES geographic (city\_id));

CREATE TABLE orderinfo

(order\_id INT PRIMARY KEY,

user\_id INT NOT NULL,

restaurant\_id INT NOT NULL,

address VARCHAR(300),

order\_status varchar(50),

FOREIGN KEY (user\_id) REFERENCES userinfo(user\_id),

FOREIGN KEY (restaurant\_id) REFERENCES restaurant(restaurant\_id));

DROP TABLE IF EXISTS payment;

CREATE TABLE payment

(transaction\_id INT PRIMARY KEY,

order\_id INT NOT NULL,

payment\_datetime TIMESTAMP NOT NULL,

amount NUMERIC(10,2) NOT NULL,

method VARCHAR(50) NOT NULL,

transaction\_status VARCHAR(50) NOT NULL,

FOREIGN KEY (order\_id) REFERENCES orderinfo(order\_id));

Queries to copy tables

COPY userinfo(user\_id, first\_name, last\_name, email, contact, gender,

age, birthdate, lastlogin, registration\_date)

FROM 'D:\improve\1 courses\sql\archive\Zomato\_db\userinfo.csv'

DELIMITER ','

CSV HEADER;

COPY geographic( City\_id, City\_name, State, Country, Latitude, Longitude, Delivery\_zone)

FROM 'D:\improve\1 courses\sql\archive\Zomato\_db\geographic.csv'

DELIMITER ','

CSV HEADER;

COPY restaurant (restaurant\_id, city\_id, restaurant\_name,

cuisine\_type, address, average\_rating, opening\_hours, delivery\_time)

FROM 'D:\improve\1 courses\sql\archive\Zomato\_db\restaurant.csv'

DELIMITER ','

CSV HEADER;

COPY orderinfo(order\_id, user\_id, restaurant\_id, address, order\_status)

FROM 'D:\improve\1 courses\sql\archive\Zomato\_db\orderinfo.csv'

DELIMITER ','

CSV HEADER;

COPY payment (transaction\_id, order\_id, payment\_datetime, amount, method, transaction\_status)

FROM 'D:\improve\1 courses\sql\archive\Zomato\_db\payment.csv'

DELIMITER ','

CSV HEADER;

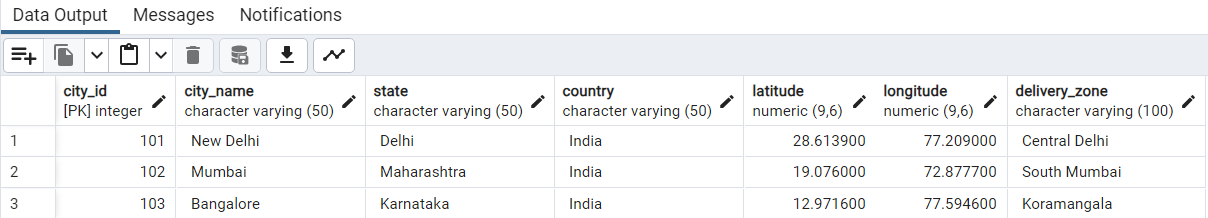
Queries to see tables with data

SELECT \* FROM userinfo;

SELECT \* FROM orderinfo;

SELECT \* FROM restaurant;

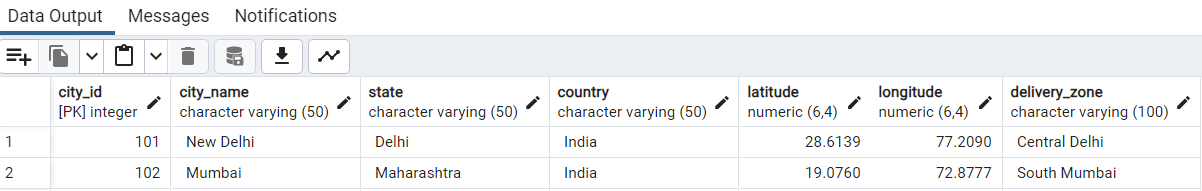
SELECT \* FROM payment;  
  
  
Queries to Answer Questions

SELECT \* FROM geographic;  
  
  
ALTER TABLE geographic

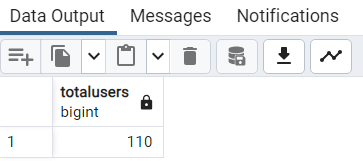
ALTER COLUMN latitude TYPE NUMERIC (6,4),

ALTER COLUMN longitude TYPE NUMERIC(6,4);

SELECT \* FROM geographic;

  
  
  
-- Que 1 Retrieve the total number of users registered on Zomato.

SELECT count(u.user\_id) AS totalusers FROM userinfo u;



-- Que 2 list of users who have placed orders using the 'cash' payment method.

SELECT u.user\_id, u.first\_name, u.last\_name

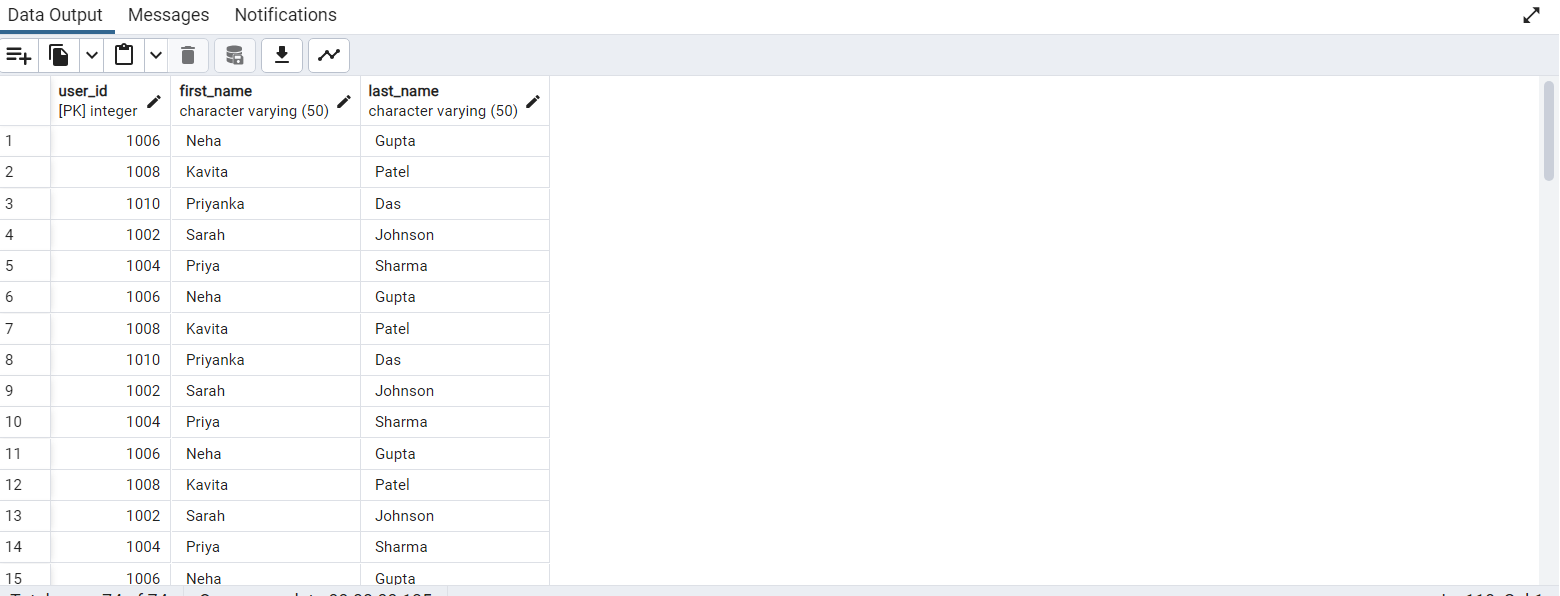
FROM userinfo u

JOIN orderinfo o ON u.user\_id = o.user\_id

JOIN payment p ON o.order\_id = p.order\_id

WHERE TRIM(LOWER(p.method)) = 'cash'; -- trim and lower is used to convert

--cash into lower case and remove leading or trailing spaces



-- Que 3 Find the number of restaurants in each city.

SELECT g.city\_name, COUNT(DISTINCT r.restaurant\_id) AS restaurant\_count

FROM geographic g

LEFT JOIN restaurant r ON g.City\_id = r.city\_id

GROUP BY g.city\_name

ORDER BY restaurant\_count DESC;

-- Que 4 List all the unique cuisine types available on Zomato.

SELECT DISTINCT r.cuisine\_type FROM restaurant r;

-- Que 5 find total no of cuisine types available on Zomato.

SELECT COUNT(DISTINCT cuisine\_type) AS total\_cuisine\_types

FROM restaurant;

-- Que 6 Retrieve the top-rated restaurant in each city.

SELECT r.restaurant\_id, r.restaurant\_name,cuisine\_type, average\_rating FROM restaurant r

ORDER BY average\_rating DESC;

----- OR-----------------------------------

WITH RankedRestaurants AS (

SELECT

r.restaurant\_id,

r.restaurant\_name,

r.cuisine\_type,

r.average\_rating,

g.City\_name,

ROW\_NUMBER() OVER (PARTITION BY g.City\_name ORDER BY r.average\_rating DESC) AS rn

FROM

restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

)

SELECT

restaurant\_id,

restaurant\_name,

cuisine\_type,

average\_rating,

City\_name

FROM

RankedRestaurants

WHERE rn = 1

ORDER BY average\_rating DESC;

-------------------------------------------------------OR----------------------------------------

WITH RankedRestaurants AS (

SELECT

r.restaurant\_id,

r.restaurant\_name,

r.cuisine\_type,

r.average\_rating,

g.City\_name,

ROW\_NUMBER() OVER (PARTITION BY g.City\_name ORDER BY CAST(r.average\_rating AS DECIMAL(3, 1)) DESC) AS rn

FROM

restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

)

SELECT

restaurant\_id,

restaurant\_name,

cuisine\_type,

average\_rating,

City\_name

FROM

RankedRestaurants

WHERE rn = 1

ORDER BY average\_rating DESC;

------------------------------OR------------------------------------------------------

SELECT r.restaurant\_id, r.restaurant\_name, r.cuisine\_type, r.average\_rating, g.City\_name

FROM restaurant r

JOIN (

SELECT city\_id, MAX(average\_rating) AS max\_rating

FROM restaurant

GROUP BY city\_id

) max\_ratings

ON r.city\_id = max\_ratings.city\_id AND r.average\_rating = max\_ratings.max\_rating

JOIN geographic g ON r.city\_id = g.City\_id

ORDER BY r.average\_rating DESC;

-------------------------------OR---------------------------------------------

SELECT r.restaurant\_id, r.restaurant\_name, r.cuisine\_type, r.average\_rating, g.City\_name

FROM restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

ORDER BY r.average\_rating DESC;

------------------------------------OR-------------------------------------------

WITH RankedRestaurants AS (

SELECT

r.restaurant\_id,

r.restaurant\_name,

r.cuisine\_type,

r.average\_rating,

g.City\_name,

ROW\_NUMBER() OVER (PARTITION BY g.City\_name ORDER BY r.average\_rating DESC) AS rn

FROM

restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

)

SELECT

restaurant\_id,

restaurant\_name,

cuisine\_type,

average\_rating,

City\_name

FROM

RankedRestaurants

WHERE rn = 1

ORDER BY average\_rating DESC;-- Sort by average\_rating

-------------------------------------or--------------------------------------------------

WITH RankedRestaurants AS (

SELECT

r.restaurant\_id,

r.restaurant\_name,

r.cuisine\_type,

r.average\_rating,

g.City\_name,

ROW\_NUMBER() OVER (PARTITION BY g.City\_name ORDER BY r.average\_rating DESC) AS rn

FROM

restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

)

SELECT

restaurant\_id,

restaurant\_name,

cuisine\_type,

average\_rating,

City\_name

FROM

RankedRestaurants

WHERE rn = 1

ORDER BY average\_rating DESC; -- Sort by average\_rating

-- Que 7 Find the average delivery time for restaurants in each city.

SELECT g.city\_name, AVG(r.delivery\_time) AS avg\_delivery\_time

FROM restaurant r

JOIN geographic g ON r.city\_id = g.city\_id

GROUP BY city\_name

ORDER BY avg\_delivery\_time DESC

----------------------------------------------------------OR------------------------

SELECT g.City\_name, ---BETTER WAY TO SOLVE THE QUESTION

AVG(EXTRACT(EPOCH FROM r.delivery\_time)::INT) AS average\_delivery\_time\_seconds --To extract the delivery time in seconds

FROM restaurant r

JOIN geographic g ON r.city\_id = g.City\_id

GROUP BY g.City\_name

ORDER BY average\_delivery\_time\_seconds DESC;

-- QUE 8 Count the total number of Delivered orders.

SELECT COUNT(\*) AS total\_completed\_orders

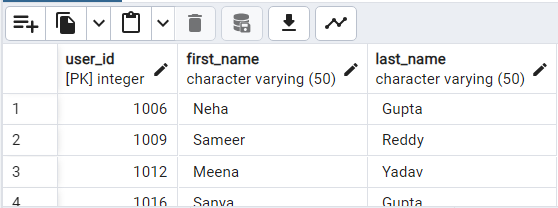
FROM orderinfo

WHERE TRIM(LOWER(order\_status)) = 'delivered'; --

-- Que 9 List all the users who have not logged in for the last 30 days.

SELECT u.user\_id, u.first\_name, u.last\_name FROM userinfo u

WHERE lastlogin <= NOW()- INTERVAL '30 days'; --- current time - 30 days



-- Que 10 Find the most common gender among Zomato users.

SELECT TRIM(LOWER(u.gender))AS user\_gender, COUNT(u.gender) AS gendertotal FROM userinfo u

GROUP BY user\_gender

ORDER BY gendertotal DESC

LIMIT 1;

--------------------------------------or-------------------------------------

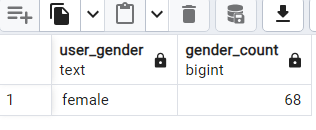
SELECT TRIM(LOWER(gender)) AS user\_gender, COUNT(\*) AS gender\_count

FROM userinfo

GROUP BY user\_gender

ORDER BY gender\_count DESC

LIMIT 1;



-- QUE 11 Retrieve the restaurants that are currently open (based on current time).

SELECT \* FROM restaurant

UPDATE restaurant ----- because opening\_hours data was not consistent some places it had extra spaces

SET opening\_hours = REPLACE(opening\_hours, ' - ', ' - ')

WHERE opening\_hours LIKE '% - %'

AND opening\_hours NOT LIKE '% - %';

---------------------------------------ANS TO QUE

SELECT restaurant\_id, restaurant\_name, opening\_hours, delivery\_time

FROM restaurant

WHERE opening\_hours IS NOT NULL

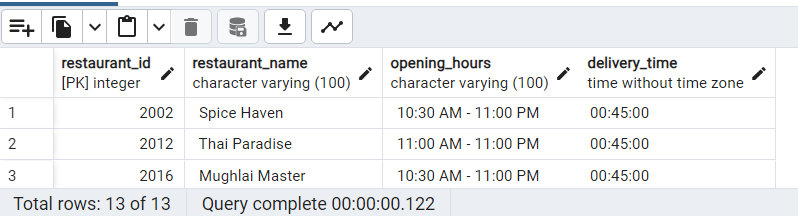
AND delivery\_time IS NOT NULL

AND CAST(CURRENT\_TIME AS TIME) BETWEEN

CAST(TRIM(BOTH ' ' FROM SUBSTRING(opening\_hours, 1, POSITION(' - ' IN opening\_hours) - 1)) AS TIME)

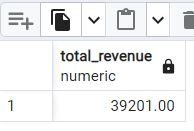
AND

CAST(TRIM(BOTH ' ' FROM SUBSTRING(opening\_hours, POSITION(' - ' IN opening\_hours) + 2)) AS TIME);



-- Que 12 Calculate the total revenue generated by Zomato.

SELECT SUM(amount) AS total\_revenue FROM payment;



-- Que 13 Find the top 5 users who have placed the most orders.

SELECT u.user\_id, u.first\_name, u.last\_name, COUNT(o.order\_id) AS order\_count

FROM userinfo u

JOIN orderinfo o ON u.user\_id = o.user\_id

GROUP BY u.user\_id, u.first\_name, u.last\_name

ORDER BY order\_count DESC

LIMIT 5;

----------or--------------

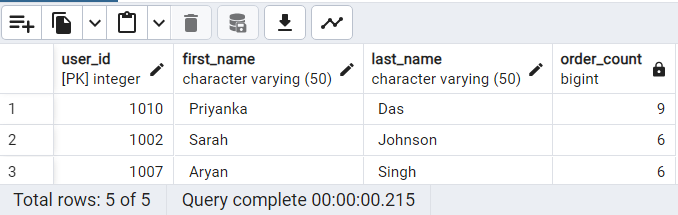
SELECT u.user\_id, u.first\_name, u.last\_name, COUNT(o.user\_id) AS total\_orders FROM orderinfo o

JOIN userinfo u ON u.user\_id = o.user\_id

Group BY u.user\_id

ORDER BY total\_orders DESC

LIMIT 5;



-- Que 14 List the restaurants that offer Indian cuisine and have an average rating above 4.

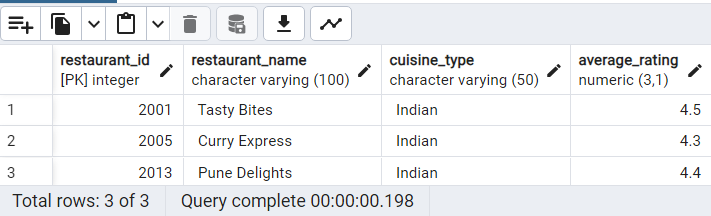
SELECT r.restaurant\_id, r.restaurant\_name, r.cuisine\_type, r.average\_rating

FROM restaurant r

WHERE TRIM(BOTH ' ' FROM LOWER(r.cuisine\_type)) = 'indian'

AND average\_rating > 4;

--BOTH keyword removes all leading and trailing spaces as well as any spaces within the text itself.

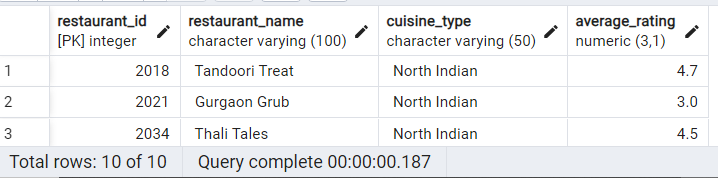


-- Que 15 List the restaurants that offer North Indian cuisine

SELECT r.restaurant\_id, r.restaurant\_name, r.cuisine\_type, r.average\_rating

FROM restaurant r

WHERE TRIM(BOTH ' ' FROM LOWER(r.cuisine\_type)) = 'north indian';



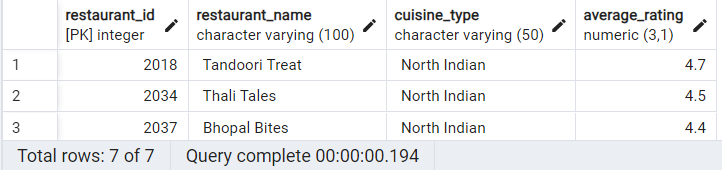
-- Que 16 List the restaurants that offer North Indian cuisine and have an average rating above 4.

SELECT r.restaurant\_id, r.restaurant\_name, r.cuisine\_type, r.average\_rating

FROM restaurant r

WHERE TRIM(BOTH ' ' FROM LOWER(r.cuisine\_type)) = 'north indian'

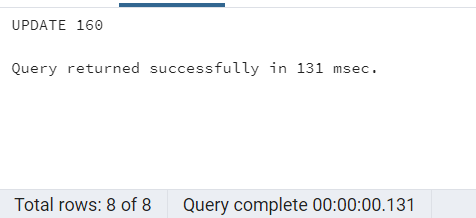
AND average\_rating > 4;



-- Que 17 Calculate the total revenue generated by each payment method.

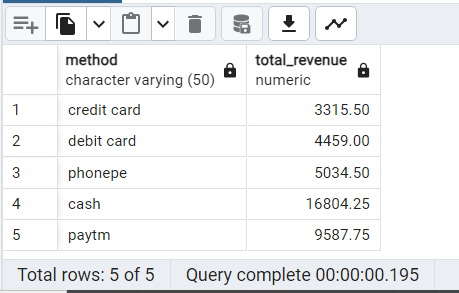
UPDATE payment

SET method = TRIM(LOWER(method)); -- method column data was not standardized



SELECT p.method, SUM(p.amount) AS total\_revenue FROM payment p

GROUP BY p.method;



-- Que 18 Identify the user who has placed the largest order by amount.

SELECT u.user\_id, u.first\_name, u.last\_name, MAX(p.amount) AS largest\_order\_amount

FROM userinfo u

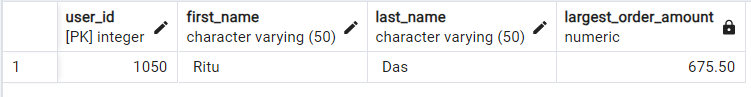
JOIN orderinfo o ON u.user\_id = o.user\_id

JOIN payment p ON o.order\_id = p.order\_id

GROUP BY u.user\_id, u.first\_name, u.last\_name

ORDER BY largest\_order\_amount DESC

LIMIT 1;



------------------------OR--------

SELECT u.user\_id, u.first\_name, u.last\_name, MAX(p.amount) AS largest\_order\_amount

FROM payment p

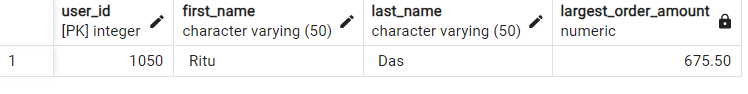
JOIN orderinfo o ON o.order\_id = p.order\_id

Join userinfo u ON u.user\_id = o.user\_id

GROUP BY u.user\_id

ORDER BY largest\_order\_amount DESC

LIMIT 1;



-- Que 19 find top 5 users who spent highest on zomato

SELECT u.user\_id, u.first\_name, u.last\_name, SUM(p.amount) AS total\_amount,

COUNT(o.user\_id) AS total\_orders FROM payment p

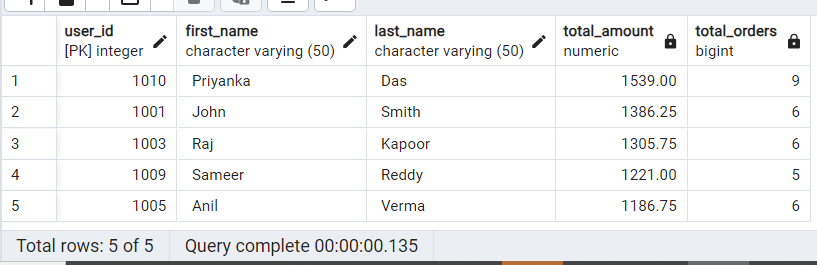
JOIN orderinfo o ON o.order\_id = p.order\_id

Join userinfo u ON u.user\_id = o.user\_id

GROUP BY u.user\_id

ORDER BY total\_amount DESC

LIMIT 5;



-------OR--------------------------------

SELECT u.user\_id, u.first\_name, u.last\_name, SUM(p.amount) AS total\_spending

FROM userinfo u

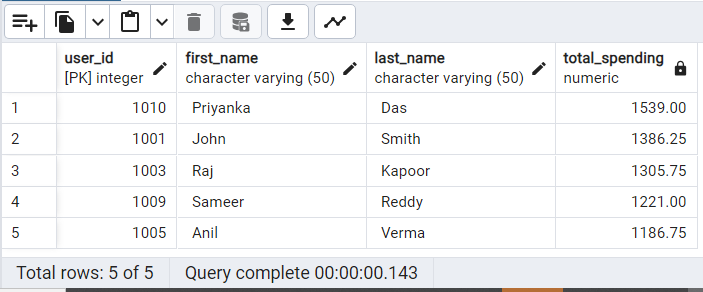
JOIN orderinfo o ON u.user\_id = o.user\_id

JOIN payment p ON o.order\_id = p.order\_id

GROUP BY u.user\_id, u.first\_name, u.last\_name

ORDER BY total\_spending DESC

LIMIT 5;



-- QUE 20 Find the city with the highest number of restaurants offering seafood cuisine.

SELECT g.city\_name, COUNT(r.restaurant\_id) AS seafood\_restaurant\_count

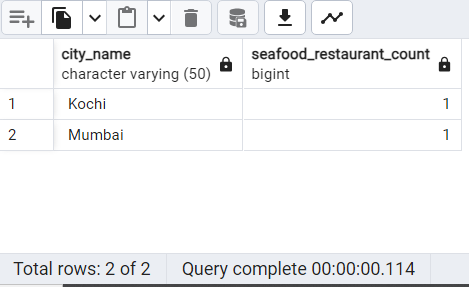
FROM geographic g

JOIN restaurant r ON r.city\_id = g.city\_id

WHERE TRIM(BOTH ' ' FROM LOWER (r.cuisine\_type)) = 'seafood'

GROUP BY g.city\_name

ORDER BY seafood\_restaurant\_count DESC;



------------------------or----------------------------------

SELECT g.City\_name, COUNT(r.restaurant\_id) AS seafood\_restaurant\_count

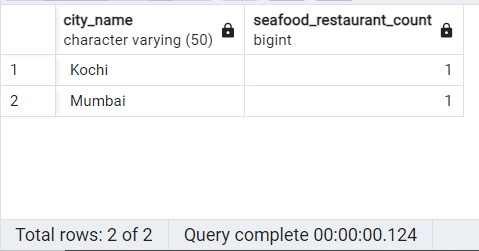
FROM geographic g

LEFT JOIN restaurant r ON g.City\_id = r.city\_id

WHERE TRIM(BOTH ' ' FROM LOWER (r.cuisine\_type)) LIKE '%seafood%'

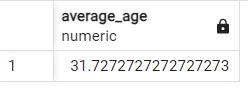
GROUP BY g.City\_name

ORDER BY seafood\_restaurant\_count DESC;



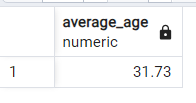
--Que 21 Calculate the average age of Zomato users.

SELECT AVG(u.age) AS average\_age FROM userinfo u



----------or--------------------------------------

SELECT ROUND(AVG(u.age),2) AS average\_age FROM userinfo u



-- QUE 22 List the cities with the highest and lowest average delivery times.

SELECT g.City\_name, AVG(EXTRACT(EPOCH FROM delivery\_time)) AS average\_delivery\_time\_seconds

FROM geographic g

LEFT JOIN restaurant r ON g.City\_id = r.city\_id

WHERE r.delivery\_time IS NOT NULL

GROUP BY g.City\_name

ORDER BY average\_delivery\_time\_seconds DESC

LIMIT 1;



SELECT g.City\_name, AVG(EXTRACT(EPOCH FROM delivery\_time)) AS average\_delivery\_time\_seconds

FROM geographic g

LEFT JOIN restaurant r ON g.City\_id = r.city\_id

WHERE r.delivery\_time IS NOT NULL

GROUP BY g.City\_name

ORDER BY average\_delivery\_time\_seconds ASC

LIMIT 1;

