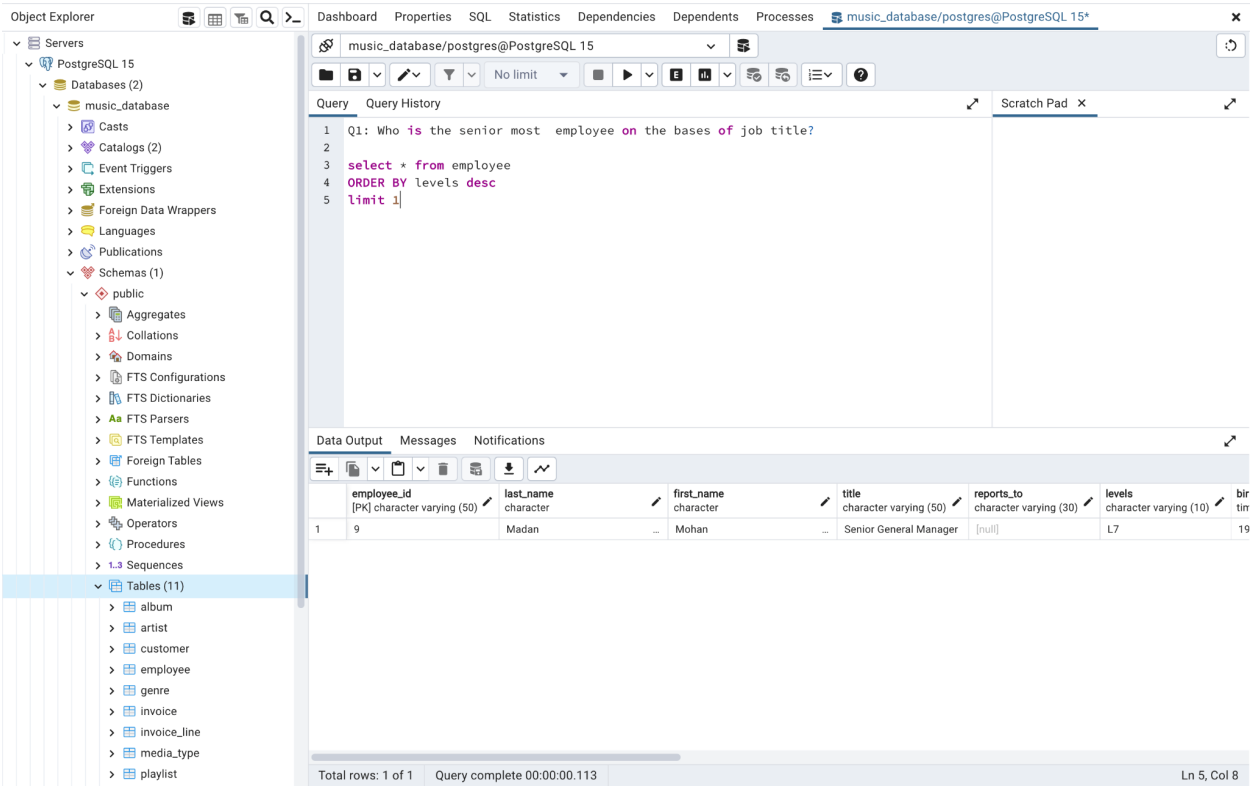
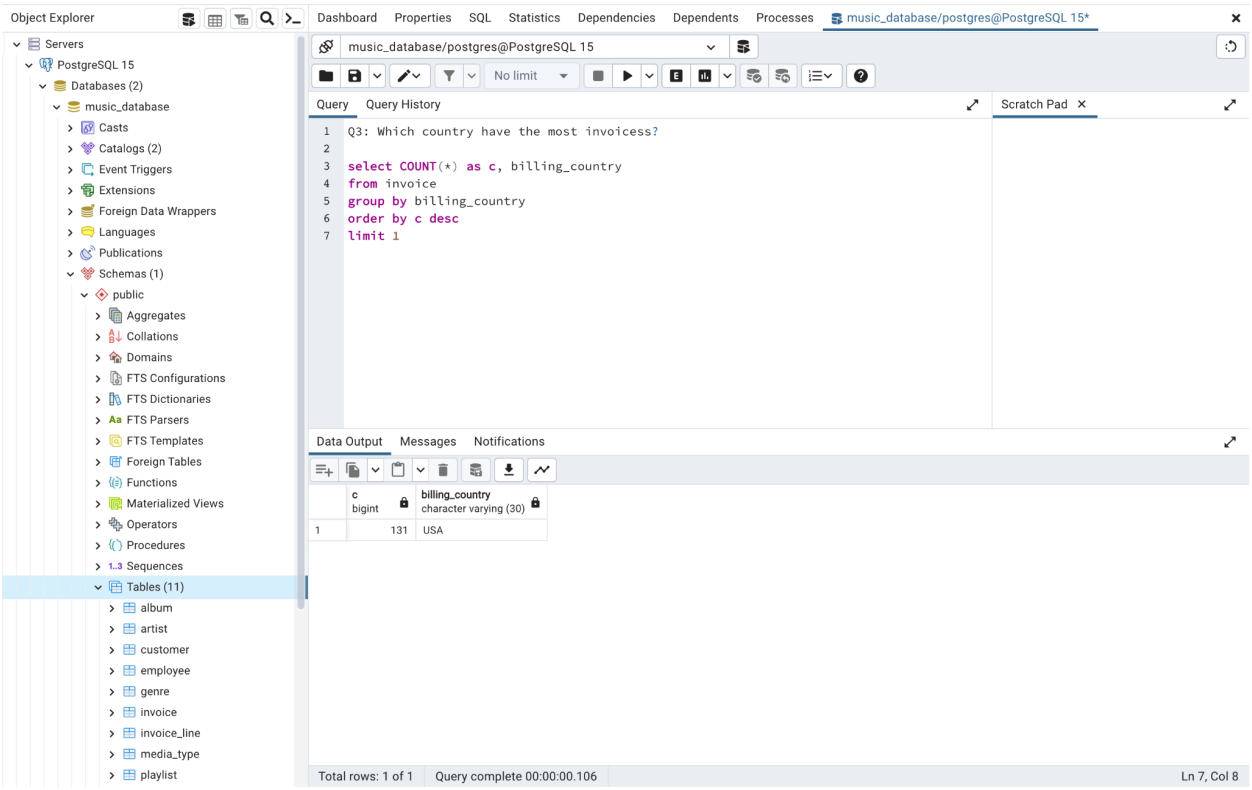
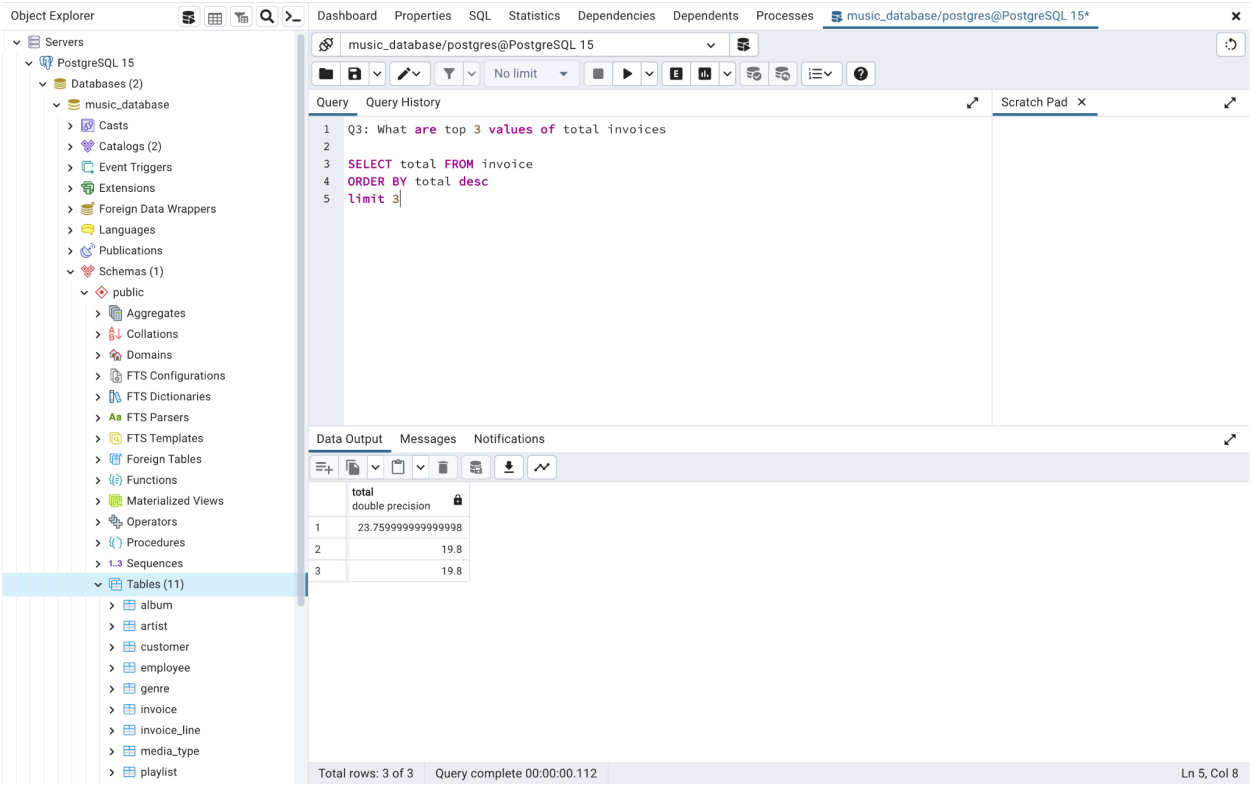
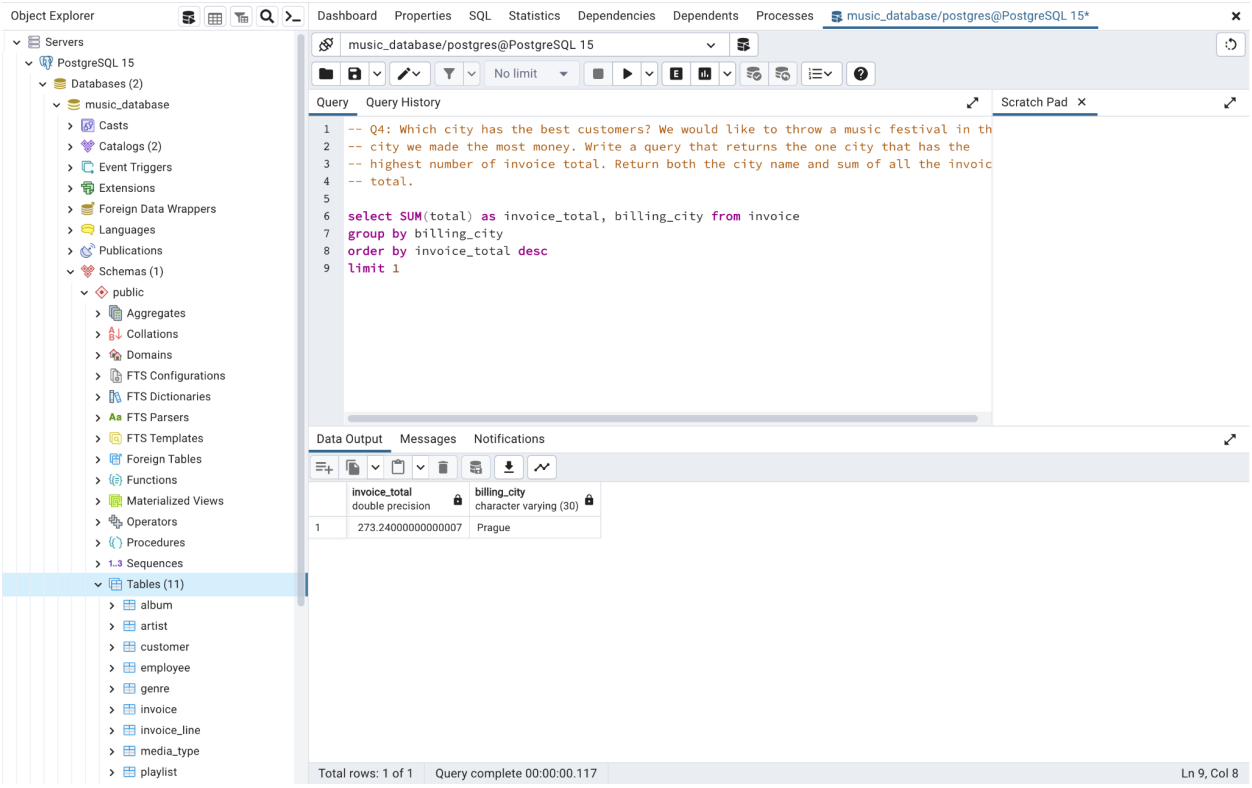
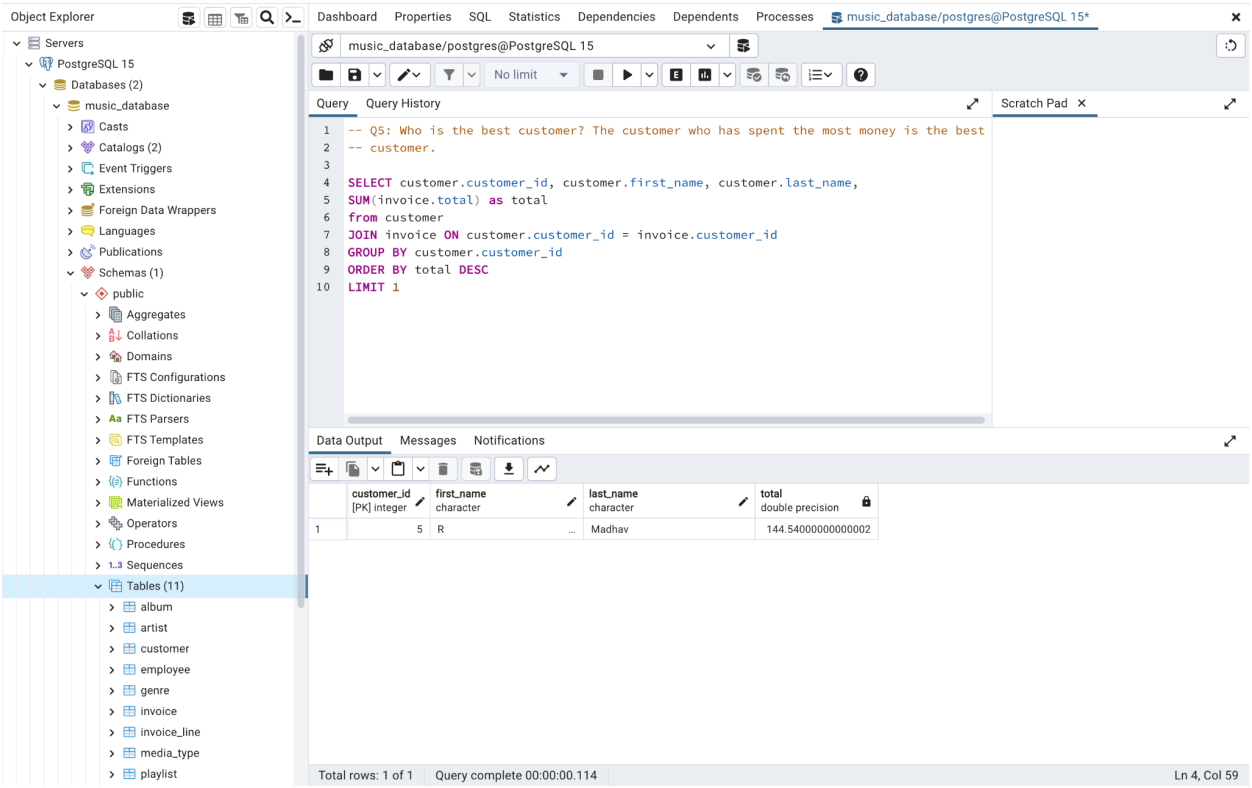
Music Store Data Analysis Project Using SQL

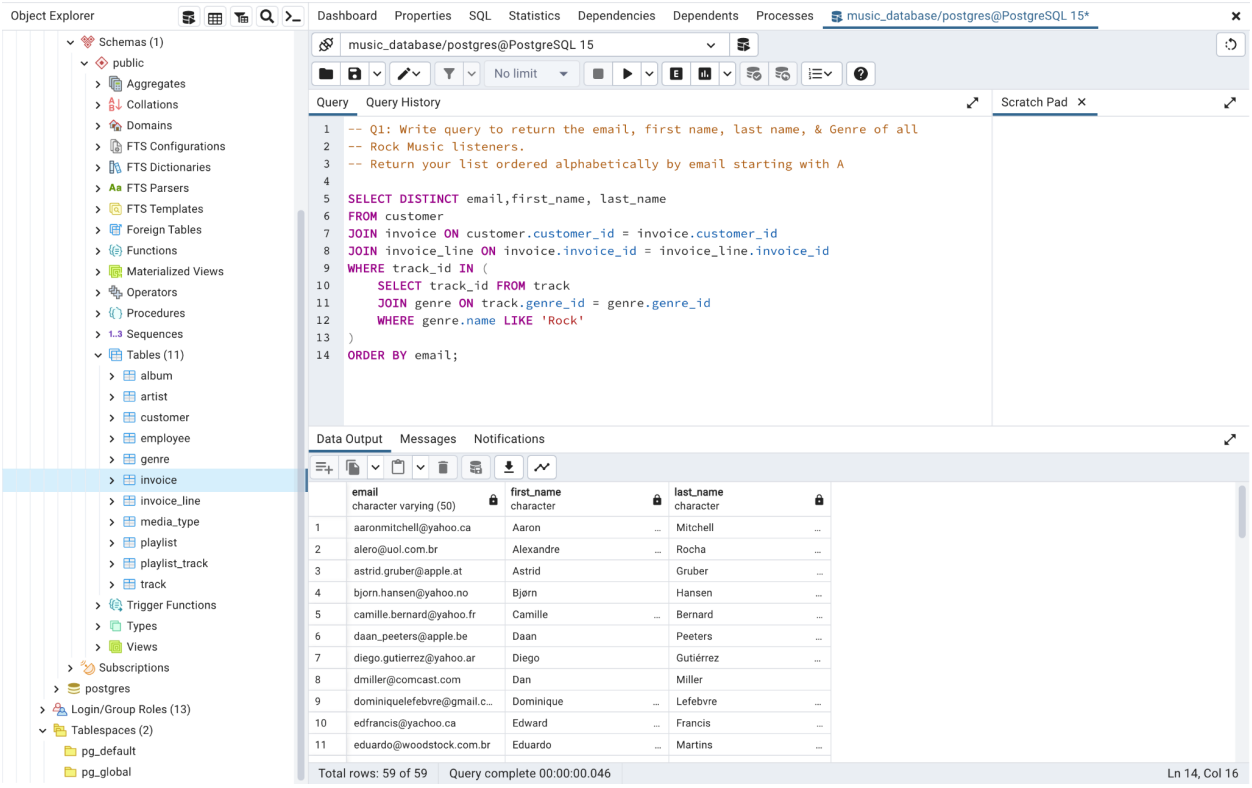


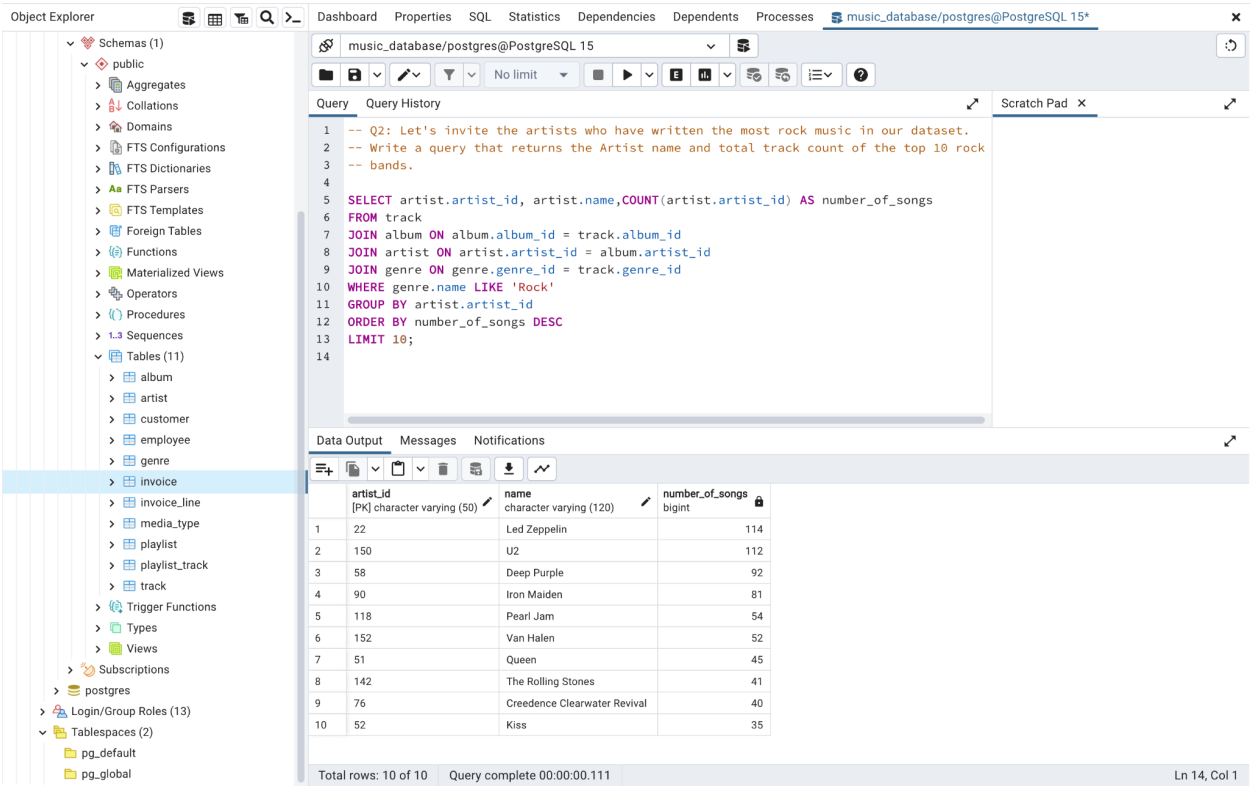


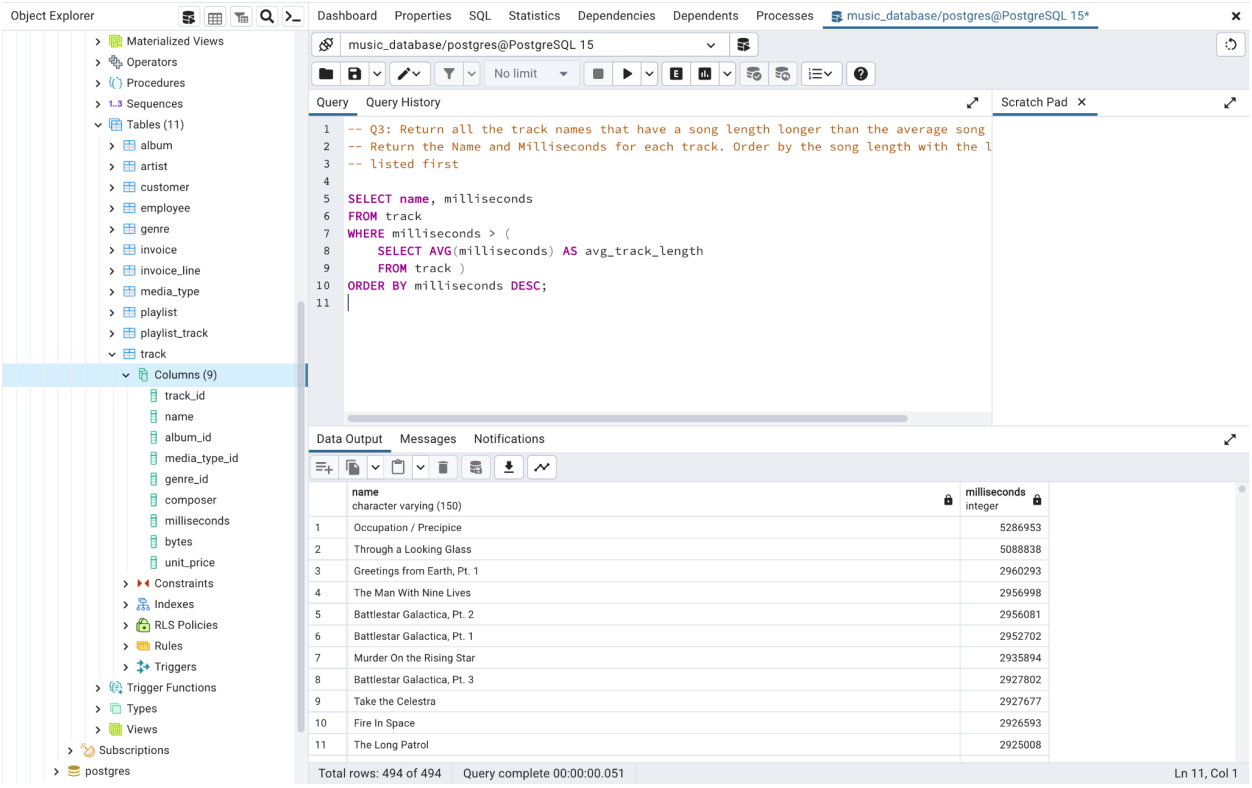

Object Explorer
Servers
PostgreSQL 15
Databases (2)
music_database
Casts
Catalogs (2)
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
1 Q3: What are top 3 values of total invoices
2
3 SELECT total FROM invoice
4 ORDER BY total desc
5 limit 3
Data Output
Messages
Notifications
total
double precision
1 23.759999999999998
2 19.8
3 19.8
Total rows: 3 of 3
Query complete 00:00:00.112
Ln 5, Col 8

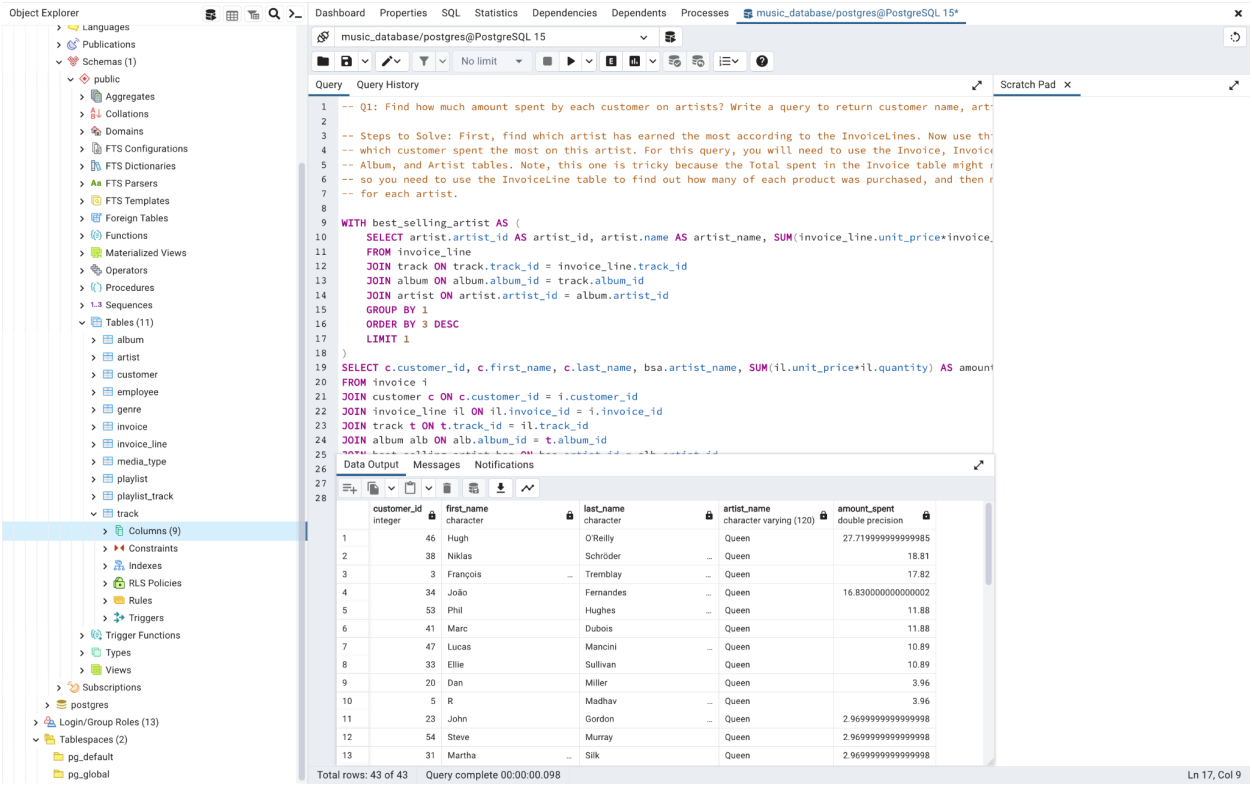

Object Explorer
Servers
PostgreSQL 15
Databases (2)
music_database
Casts
Catalogs (2)
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
1 -- Q4: Which city has the best customers? We would like to throw a music festival in th
2 -- city we made the most money. Write a query that returns the one city that has the
3 -- highest number of invoice total. Return both the city name and sum of all the invoic
4 -- total.
5
6 select SUM(total) as invoice_total, billing_city from invoice
7 group by billing_city
8 order by invoice_total desc
9 limit 1
Data Output
Messages
Notifications
invoice_total
double precision
billing_city
character varying (30)
1
273.24000000000007
Prague
Total rows: 1 of 1
Query complete 00:00:00.117
Ln 9, Col 8

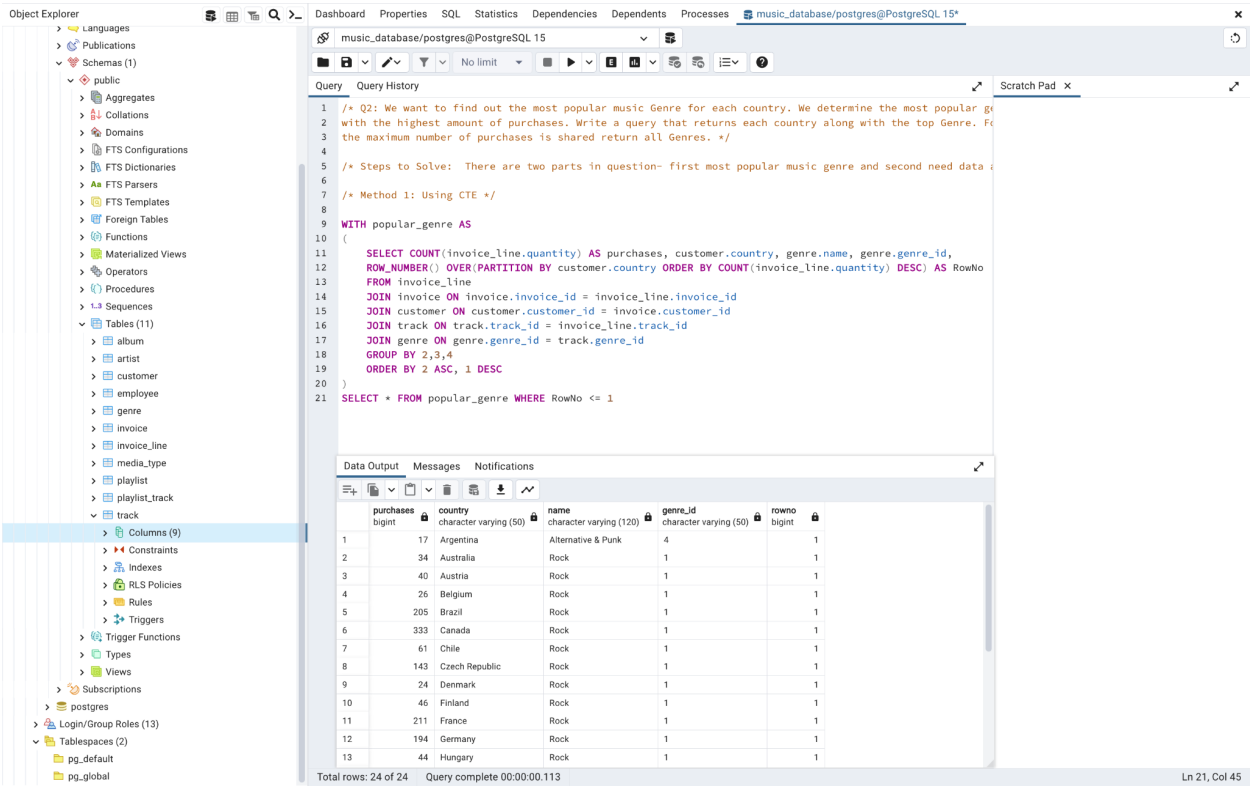

Object Explorer
Servers
PostgreSQL 15
Databases (2)
music_database
Casts
Catalogs (2)
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
1 -- Q5: Who is the best customer? The customer who has spent the most money is the best
2 -- customer.
3
4 SELECT customer.customer_id, customer.first_name, customer.last_name,
5 SUM(invoice.total) as total
6 from customer
7 JOIN invoice ON customer.customer_id = invoice.customer_id
8 GROUP BY customer.customer_id
9 ORDER BY total DESC
10 LIMIT 1
Data Output
Messages
Notifications
customer_id
[PK] integer
first_name
character
last_name
character
total
double precision
1
5
R
Madhav
144.54000000000002
Total rows: 1 of 1
Query complete 00:00:00.114
Ln 4, Col 59

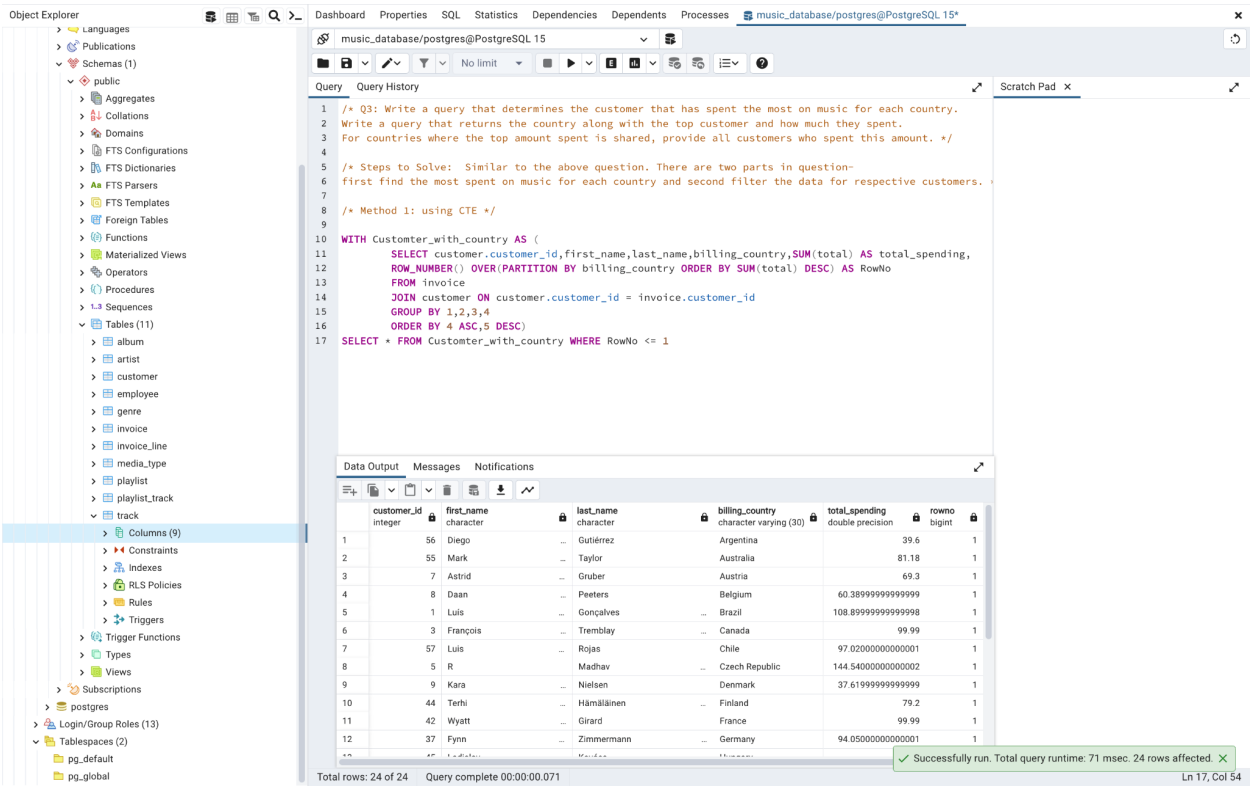

Object Explorer
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
-- Q1: Write query to return the email, first name, last name, & Genre of all
-- Rock Music listeners.
-- Return your list ordered alphabetically by email starting with A
SELECT DISTINCT email,first_name, last_name
FROM customer
JOIN invoice ON customer.customer_id = invoice.customer_id
JOIN invoice_line ON invoice.invoice_id = invoice_line.invoice_id
WHERE track_id IN (
SELECT track_id FROM track
JOIN genre ON track.genre_id = genre.genre_id
WHERE genre.name LIKE 'Rock'
)
ORDER BY email;
Data Output
Messages
Notifications
email character varying (50)
first_name character
last_name character
1 aaronmitchell@yahoo.ca Aaron Mitchell
2 alero@uol.com.br Alexandre Rocha
3 astrid.gruber@apple.at Astrid Gruber
4 bjorn.hansen@yahoo.no Bjørn Hansen
5 camille.bernard@yahoo.fr Camille Bernard
6 daan_peeters@apple.be Daan Peeters
7 diego.gutierrez@yahoo.ar Diego Gutiérrez
8 dmiller@comcast.com Dan Miller
9 dominiquelefebvre@gmail.c... Dominique Lefebvre
10 edfrancis@yachoo.ca Edward Francis
11 eduardo@woodstock.com.br Eduardo Martins
Total rows: 59 of 59
Query complete 00:00:00.046
Ln 14, Col 16
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
playlist_track
track
Trigger Functions
Types
Views
Subscriptions
postgres
Login/Group Roles (13)
Tablespaces (2)
pg_default
pg_global


Object Explorer
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
-- Q2: Let's invite the artists who have written the most rock music in our dataset.
-- Write a query that returns the Artist name and total track count of the top 10 rock
-- bands.
SELECT artist.artist_id, artist.name,COUNT(artist.artist_id) AS number_of_songs
FROM track
JOIN album ON album.album_id = track.album_id
JOIN artist ON artist.artist_id = album.artist_id
JOIN genre ON genre.genre_id = track.genre_id
WHERE genre.name LIKE 'Rock'
GROUP BY artist.artist_id
ORDER BY number_of_songs DESC
LIMIT 10;
Data Output
Messages
Notifications
artist_id [PK] character varying (50)
name character varying (120)
number_of_songs bigint
1 22 Led Zeppelin 114
2 150 U2 112
3 58 Deep Purple 92
4 90 Iron Maiden 81
5 118 Pearl Jam 54
6 152 Van Halen 52
7 51 Queen 45
8 142 The Rolling Stones 41
9 76 Creedence Clearwater Revival 40
10 52 Kiss 35
Total rows: 10 of 10
Query complete 00:00:00.111
Ln 14, Col 1
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
playlist_track
track
Trigger Functions
Types
Views
Subscriptions
postgres
Login/Group Roles (13)
Tablespaces (2)
pg_default
pg_global


Object Explorer
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
playlist_track
track
Columns (9)
track_id
name
album_id
media_type_id
genre_id
composer
milliseconds
bytes
unit_price
Constraints
Indexes
RLS Policies
Rules
Triggers
Trigger Functions
Types
Views
Subscriptions
postgres
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
1 -- Q3: Return all the track names that have a song length longer than the average song
2 -- Return the Name and Milliseconds for each track. Order by the song length with the l
3 -- listed first
4
5 SELECT name, milliseconds
6 FROM track
7 WHERE milliseconds > (
8 SELECT AVG(milliseconds) AS avg_track_length
9 FROM track)
10 ORDER BY milliseconds DESC;
11
Data Output
Messages
Notifications
name
character varying (150)
milliseconds
integer
1 Occupation / Precipice 5286953
2 Through a Looking Glass 5088838
3 Greetings from Earth, Pt. 1 2960293
4 The Man With Nine Lives 2956998
5 Battlestar Galactica, Pt. 2 2956081
6 Battlestar Galactica, Pt. 1 2952702
7 Murder On the Rising Star 2935894
8 Battlestar Galactica, Pt. 3 2927802
9 Take the Celestra 2927677
10 Fire In Space 2926593
11 The Long Patrol 2925008
Total rows: 494 of 494
Query complete 00:00:00.051
Ln 11, Col 1


Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query Query History
Scratch Pad
-- Q1: Find how much amount spent by each customer on artists? Write a query to return customer name, art
-- Steps to Solve: First, find which artist has earned the most according to the InvoiceLines. Now use th
-- which customer spent the most on this artist. For this query, you will need to use the Invoice, Invoic
-- Album, and Artist tables. Note, this one is tricky because the Total spent in the Invoice table might
-- so you need to use the InvoiceLine table to find out how many of each product was purchased, and then
-- for each artist.
WITH best_selling_artist AS (
SELECT artist.artist_id AS artist_id, artist.name AS artist_name, SUM(invoice_line.unit_price*invoice_
FROM invoice_line
JOIN track ON track.track_id = invoice_line.track_id
JOIN album ON album.album_id = track.album_id
JOIN artist ON artist.artist_id = album.artist_id
GROUP BY 1
ORDER BY 3 DESC
LIMIT 1
)
SELECT c.customer_id, c.first_name, c.last_name, bsa.artist_name, SUM(il.unit_price*il.quantity) AS amount
FROM invoice i
JOIN customer c ON c.customer_id = i.customer_id
JOIN invoice_line il ON il.invoice_id = i.invoice_id
JOIN track t ON t.track_id = il.track_id
JOIN album alb ON alb.album_id = t.album_id
Data Output Messages Notifications
customer_id integer
first_name character
last_name character
artist_name character varying (120)
amount_spent double precision
1 46 Hugh O'Reilly Queen 27.719999999999985
2 38 Niklas Schröder Queen 18.81
3 3 François Tremblay Queen 17.82
4 34 João Fernandes Queen 16.830000000000002
5 53 Phil Hughes Queen 11.88
6 41 Marc Dubois Queen 11.88
7 47 Lucas Mancini Queen 10.89
8 33 Ellie Sullivan Queen 10.89
9 20 Dan Miller Queen 3.96
10 5 R Madhav Queen 3.96
11 23 John Gordon Queen 2.9699999999999998
12 54 Steve Murray Queen 2.9699999999999998
13 31 Martha Silk Queen 2.9699999999999998
Total rows: 43 of 43
Query complete 00:00:00.098
Ln 17, Col 9


Object Explorer
Dashboard Properties SQL Statistics Dependencies Dependents Processes music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query Query History
Scratch Pad
/* Q2: We want to find out the most popular music Genre for each country. We determine the most popular g
with the highest amount of purchases. Write a query that returns each country along with the top Genre. F
the maximum number of purchases is shared return all Genres. */
/* Steps to Solve: There are two parts in question- first most popular music genre and second need data
/* Method 1: Using CTE */
WITH popular_genre AS
(
SELECT COUNT(invoice_line.quantity) AS purchases, customer.country, genre.name, genre.genre_id,
ROW_NUMBER() OVER(PARTITION BY customer.country ORDER BY COUNT(invoice_line.quantity) DESC) AS RowNo
FROM invoice_line
JOIN invoice ON invoice.invoice_id = invoice_line.invoice_id
JOIN customer ON customer.customer_id = invoice.customer_id
JOIN track ON track.track_id = invoice_line.track_id
JOIN genre ON genre.genre_id = track.genre_id
GROUP BY 2,3,4
ORDER BY 2 ASC, 1 DESC
)
SELECT * FROM popular_genre WHERE RowNo <= 1
Data Output Messages Notifications
purchases bigint
country character varying (50)
name character varying (120)
genre_id character varying (50)
rowno bigint
1 17 Argentina Alternative & Punk 4 1
2 34 Australia Rock 1 1
3 40 Austria Rock 1 1
4 26 Belgium Rock 1 1
5 205 Brazil Rock 1 1
6 333 Canada Rock 1 1
7 61 Chile Rock 1 1
8 143 Czech Republic Rock 1 1
9 24 Denmark Rock 1 1
10 46 Finland Rock 1 1
11 211 France Rock 1 1
12 194 Germany Rock 1 1
13 44 Hungary Rock 1 1
Total rows: 24 of 24
Query complete 00:00:00.113
Ln 21, Col 45


Object Explorer
Languages
Publications
Schemas (1)
public
Aggregates
Collations
Domains
FTS Configurations
FTS Dictionaries
FTS Parsers
FTS Templates
Foreign Tables
Functions
Materialized Views
Operators
Procedures
Sequences
Tables (11)
album
artist
customer
employee
genre
invoice
invoice_line
media_type
playlist
playlist_track
track
Columns (9)
Constraints
Indexes
RLS Policies
Rules
Triggers
Trigger Functions
Types
Views
Subscriptions
postgres
Login/Group Roles (13)
Tablespaces (2)
pg_default
pg_global
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
music_database/postgres@PostgreSQL 15*
music_database/postgres@PostgreSQL 15
No limit
Query
Query History
Scratch Pad
1 /* Q3: Write a query that determines the customer that has spent the most on music for each country.
2 Write a query that returns the country along with the top customer and how much they spent.
3 For countries where the top amount spent is shared, provide all customers who spent this amount. */
4
5 /* Steps to Solve: Similar to the above question. There are two parts in question-
6 first find the most spent on music for each country and second filter the data for respective customers.
7
8 /* Method 1: using CTE */
9
10 WITH Customter_with_country AS (
11 SELECT customer.customer_id,first_name,last_name,billing_country,SUM(total) AS total_spending,
12 ROW_NUMBER() OVER(PARTITION BY billing_country ORDER BY SUM(total) DESC) AS RowNo
13 FROM invoice
14 JOIN customer ON customer.customer_id = invoice.customer_id
15 GROUP BY 1,2,3,4
16 ORDER BY 4 ASC,5 DESC)
17 SELECT * FROM Customter_with_country WHERE RowNo <= 1
Data Output
Messages
Notifications
customer_id integer
first_name character
last_name character
billing_country character varying (30)
total_spending double precision
rowno bigint
1 56 Diego Gutiérrez Argentina 39.6 1
2 55 Mark Taylor Australia 81.18 1
3 7 Astrid Gruber Austria 69.3 1
4 8 Daan Peeters Belgium 60.38999999999999 1
5 1 Luís Gonçalves Brazil 108.89999999999998 1
6 3 François Tremblay Canada 99.99 1
7 57 Luis Rojas Chile 97.02000000000001 1
8 5 R Madhav Czech Republic 144.54000000000002 1
9 9 Kara Nielsen Denmark 37.61999999999999 1
10 44 Terhi Hämäläinen Finland 79.2 1
11 42 Wyatt Girard France 99.99 1
12 37 Fynn Zimmermann Germany 94.05000000000001 1
Successfully run. Total query runtime: 71 msec. 24 rows affected.
Total rows: 24 of 24
Query complete 00:00:00.071
Ln 17, Col 54