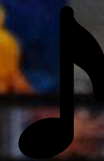




MUSIC STORE DATA ANALYSIS

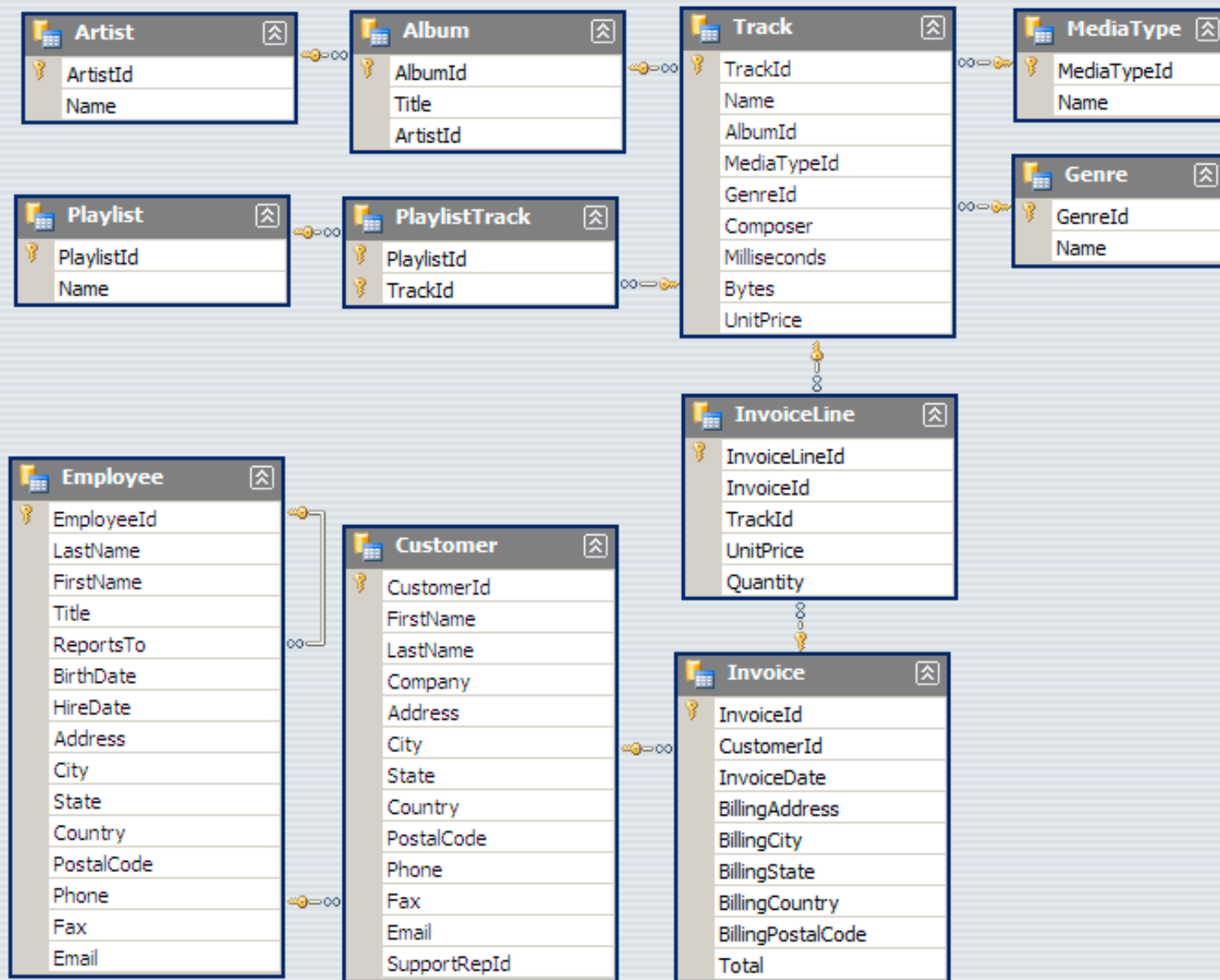
SQL PROJECT (CASE STUDY)



BY:- AMIT PAWAR

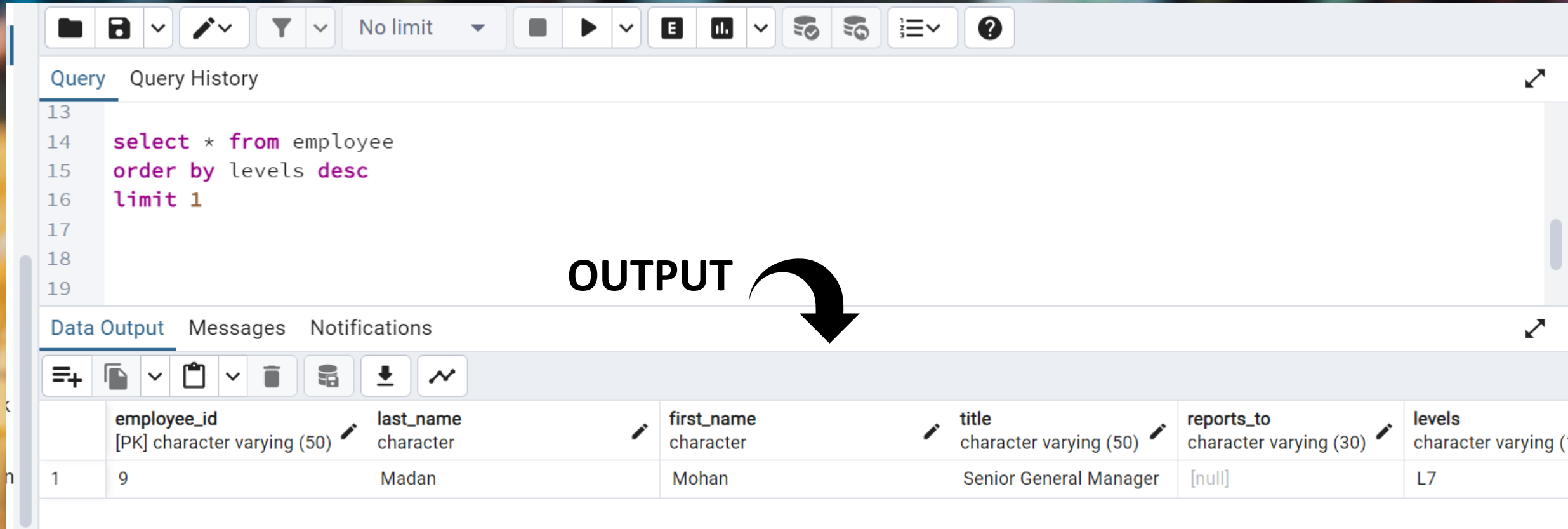


SCHEMA DIGRAM



Question Set 1 – Easy

1) Who is the senior most employee based on job title?



The screenshot shows a database query tool interface. The top toolbar contains icons for file operations, filters, and execution. The 'Query' tab is active, displaying a SQL query. The 'Data Output' tab is also visible, showing the results of the query in a table format. A large black arrow points from the word 'OUTPUT' to the 'Data Output' tab.

```
13
14 select * from employee
15 order by levels desc
16 limit 1
17
18
19
```

OUTPUT

	employee_id [PK] character varying (50)	last_name character	first_name character	title character varying (50)	reports_to character varying (30)	levels character varying (
1	9	Madan	Mohan	Senior General Manager	[null]	L7

Question Set 1 – Easy

2) Which countries have the most Invoices?

The screenshot shows a SQL query editor interface. At the top, there is a toolbar with icons for file operations, editing, and execution. Below the toolbar, there are tabs for 'Query' and 'Query History'. The 'Query' tab is active, showing a SQL query in a text area. The query is as follows:

```
#2)|
select count(billing_country), billing_country from invoice
group by billing_country
order by billing_country desc
limit 5
```

Below the query editor, there is a section labeled 'OUTPUT' with a curved arrow pointing to the results. The results are displayed in a table with the following columns: 'count' (bigint) and 'billing_country' (character varying (30)). The table contains five rows of data:

	count bigint	billing_country character varying (30)
1	131	USA
2	28	United Kingdom
3	10	Sweden
4	11	Spain
5	29	Portugal

Question Set 1 – Easy

3) What are top 3 values of total invoice?

The screenshot shows a SQL query editor interface. The query is as follows:

```
#3)
select total from invoice
order by total desc
limit 3
```

The word **OUTPUT** is written in large black letters, with a curved arrow pointing from it to the 'Data Output' tab. The 'Data Output' tab is active, showing a table with the results of the query. The table has two columns: 'total' and 'double precision'. The first row shows a total of 23.759999999999998. The second and third rows both show a total of 19.8.

	total	double precision
1	23.759999999999998	
2		19.8
3		19.8

Question Set 1 – Easy

4) Which city has the best customers? We would like to throw a promotional Music Festival in the city we made the most money. Write a query that returns one city that has the highest sum of invoice totals. Return both the city name & sum of all invoice totals

The screenshot shows a database management interface with a left sidebar containing a tree view of database objects (tables, triggers, types, views, subscriptions, postgres, casts, catalogs, event triggers, extensions, foreign data wrapper). The main area is divided into two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query that calculates the total invoice amount for each billing city, ordered by total invoice amount in descending order. The query is as follows:

```
#4)
select sum(total) as total_invoice, billing_city from invoice
group by billing_city
order by total_invoice desc
```

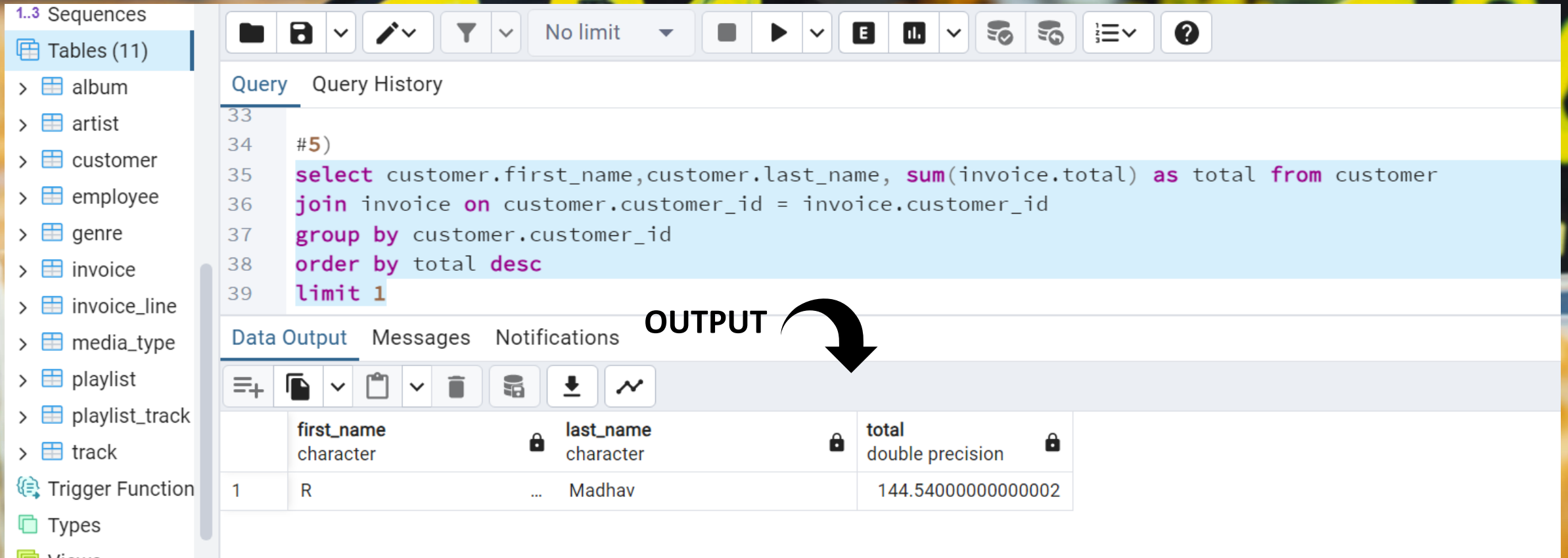
Below the query editor, there is a section labeled 'OUTPUT' with a large black arrow pointing to the 'Data Output' tab. The 'Data Output' tab is active, showing a table with the results of the query. The table has two columns: 'total_invoice' (double precision) and 'billing_city' (character varying (30)). The data is as follows:

	total_invoice double precision	billing_city character varying (30)
1	273.240000000000007	Prague
2	169.29	Mountain View
3	166.32	London
4	158.4	Berlin
5	151.47	Paris
6	129.69	São Paulo
7	114.839999999999997	Dublin
8	111.869999999999999	Delhi
9	108.899999999999998	São José dos Campos

At the bottom of the interface, there is a status bar showing 'Total rows: 53 of 53' and 'Query complete 00:00:00.057'. The bottom right corner indicates 'Ln 34, Col 1'.

Question Set 1 – Easy

1. 5) Who is the best customer? The customer who has spent the most money will be declared the best customer. Write a query that returns the person who has spent the most money



The screenshot shows a database management interface. On the left, a sidebar lists tables: album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, and track. The main area displays a SQL query in a text editor. The query is as follows:

```
#5)
select customer.first_name, customer.last_name, sum(invoice.total) as total from customer
join invoice on customer.customer_id = invoice.customer_id
group by customer.customer_id
order by total desc
limit 1
```

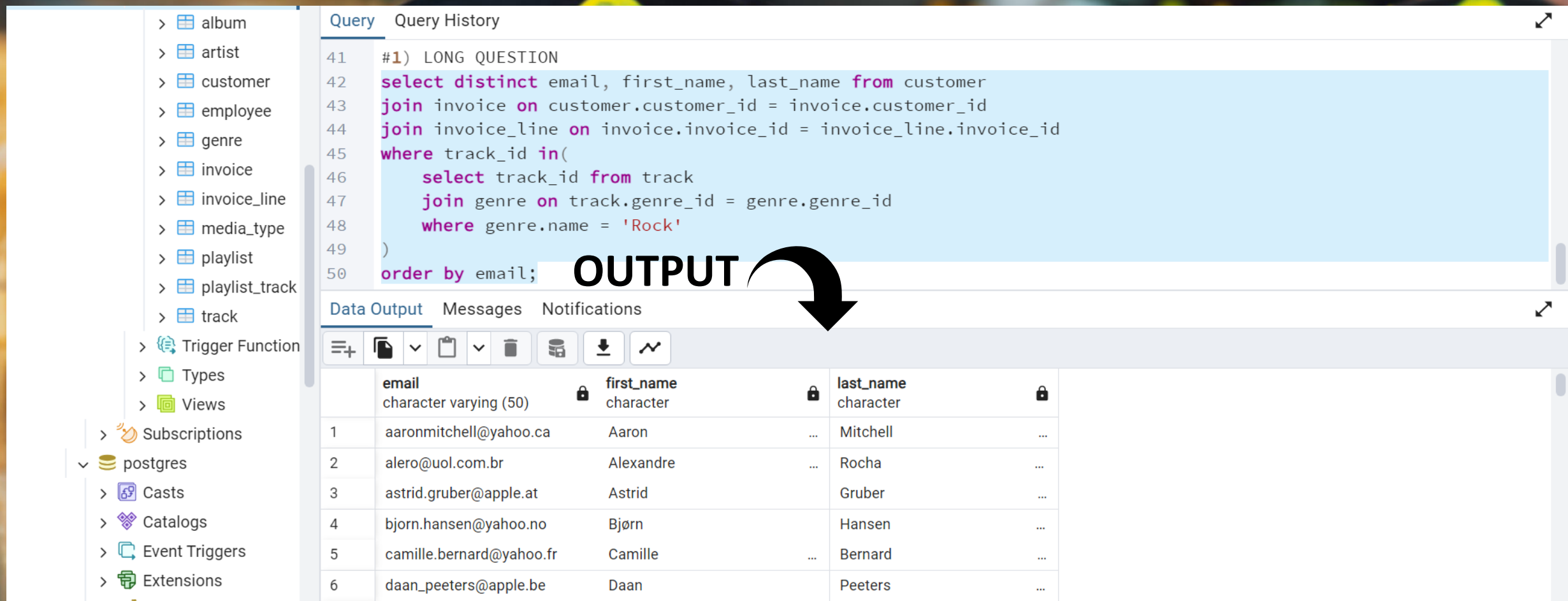
Below the query editor, the 'Data Output' tab is active, showing the results of the query. The output is a table with three columns: first_name, last_name, and total. The first row shows the results for the customer with the highest total spending.

	first_name character	last_name character	total double precision
1	R	Madhav	144.540000000000002

An arrow labeled 'OUTPUT' points from the query editor to the data output table.

Question Set 2 – Moderate

1) 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



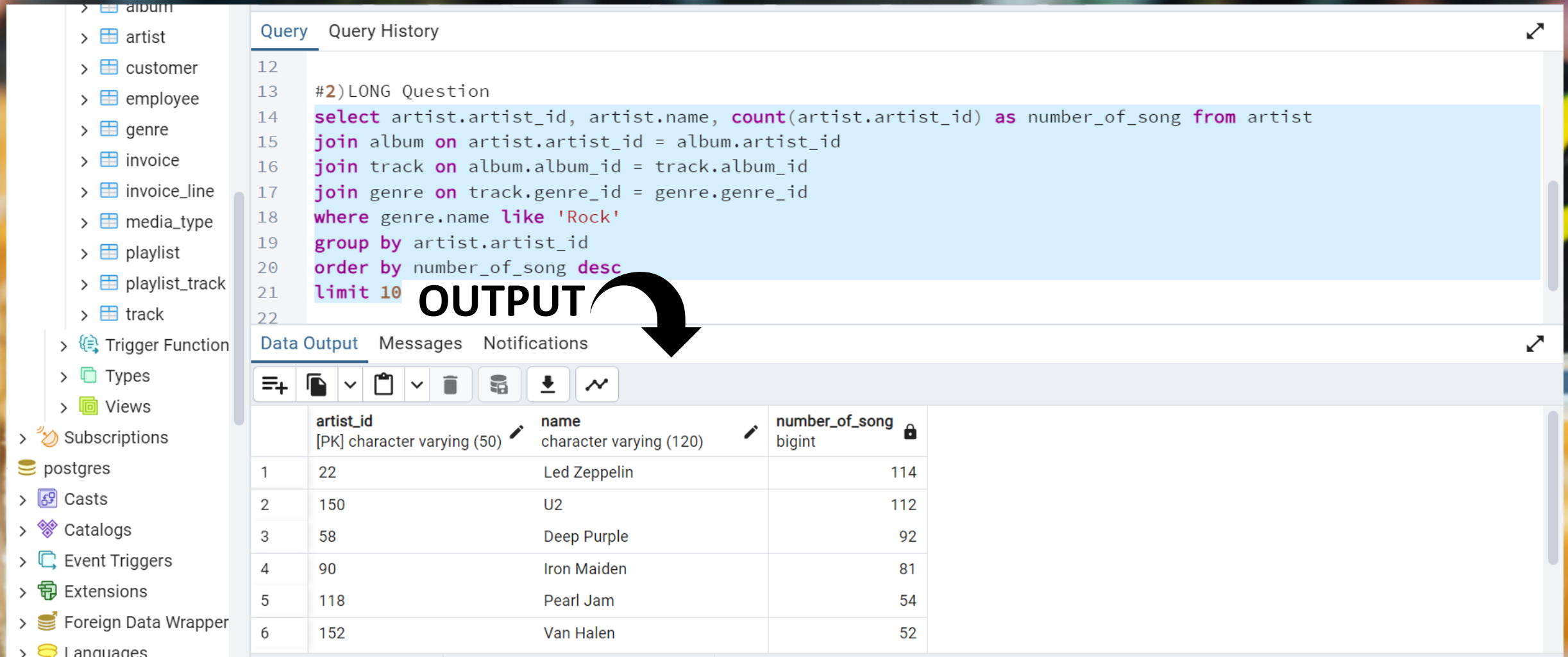
The screenshot displays a database management interface. On the left is a sidebar with a tree view of database objects: album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track, Trigger Function, Types, Views, Subscriptions, and postgres (expanded). The main area is divided into two tabs: 'Query' and 'Query History'. The 'Query' tab contains a SQL query (lines 41-50) that selects distinct email, first_name, and last_name from the customer table, joined with invoice and invoice_line tables, and filtered by track_id in a subquery that selects track_id from the track table where the genre is 'Rock'. The results are ordered by email. A large black arrow labeled 'OUTPUT' points from the query to the 'Data Output' tab below. The 'Data Output' tab shows a table with 6 rows of results, including email, first_name, and last_name.

```
41 #1) LONG QUESTION
42 select distinct email, first_name, last_name from customer
43 join invoice on customer.customer_id = invoice.customer_id
44 join invoice_line on invoice.invoice_id = invoice_line.invoice_id
45 where track_id in(
46     select track_id from track
47     join genre on track.genre_id = genre.genre_id
48     where genre.name = 'Rock'
49 )
50 order by email;
```

	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

Question Set 2 – Moderate

2) 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



The screenshot shows a database management interface with a left sidebar containing a tree view of database objects: album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, track, Trigger Function, Types, Views, Subscriptions, postgres, Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrapper, and Languages.

The main area displays a SQL query in the 'Query' tab, with 'Query History' also visible. The query is as follows:

```
12
13 #2) LONG Question
14 select artist.artist_id, artist.name, count(artist.artist_id) as number_of_song from artist
15 join album on artist.artist_id = album.artist_id
16 join track on album.album_id = track.album_id
17 join genre on track.genre_id = genre.genre_id
18 where genre.name like 'Rock'
19 group by artist.artist_id
20 order by number_of_song desc
21 limit 10
22
```

A large black arrow points from the query to the 'Data Output' tab, which is labeled 'OUTPUT' in large black text. The 'Data Output' tab shows a table with the following data:

	artist_id [PK] character varying (50)	name character varying (120)	number_of_song 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


Question Set 2 – Moderate

1. 3) Return all the track names that have a song length longer than the average song length. Return the Name and Milliseconds for each track. Order by the song length with the longest songs listed first

- > employee
- > genre
- > invoice
- > invoice_line
- > media_type
- > playlist
- > playlist_track
- > track
- > Trigger Function
- > Types
- > Views
- Subscriptions
- postgres
- Casts
- Catalogs
- Event Triggers
- Extensions
- Foreign Data Wrapper
- Languages

```
24
25 #3) Long Question
26 select track.track_id, track.name, track.milliseconds from track
27 where milliseconds > (
28     select avg(milliseconds) as average_song_length from track
29 )
30 order by milliseconds desc
31
32
33
```

OUTPUT

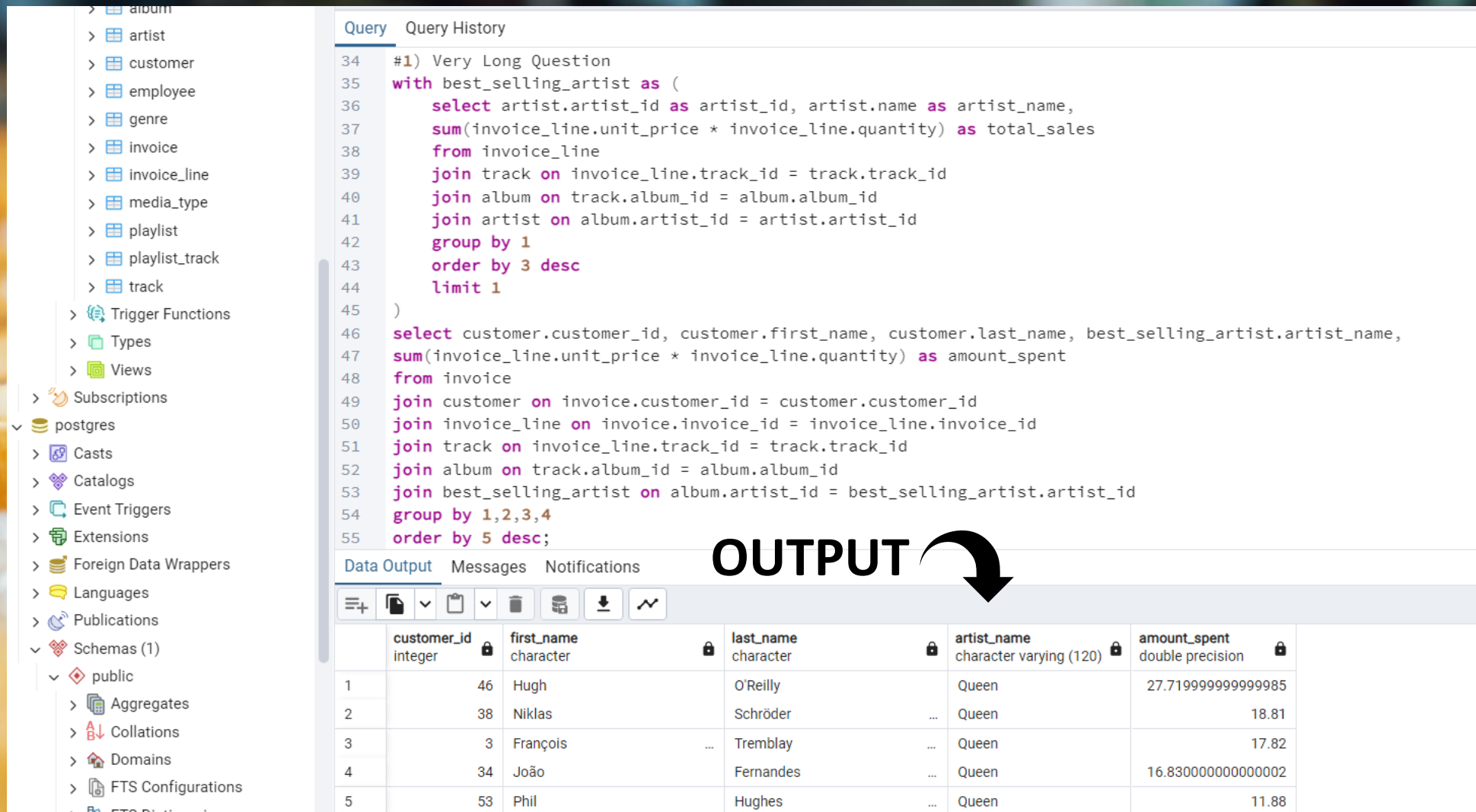


Data Output Messages Notifications

	track_id [PK] integer	name character varying (150)	milliseconds integer
5	3227	Battlestar Galactica, Pt. 2	2956081
6	3226	Battlestar Galactica, Pt. 1	2952702
7	3243	Murder On the Rising Star	2935894
8	3228	Battlestar Galactica, Pt. 3	2927802
9	3248	Take the Celestra	2927677
10	3239	Fire In Space	2926593

Question Set 3 – Advance

1) Find how much amount spent by each customer on artists? Write a query to return customer name, artist name and total spent



The screenshot displays a PostgreSQL database interface. On the left, a sidebar shows the database schema with tables like album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, and track. The main area shows a SQL query in the 'Query' tab, which is a complex join query to find the total amount spent by each customer on artists. The query is as follows:

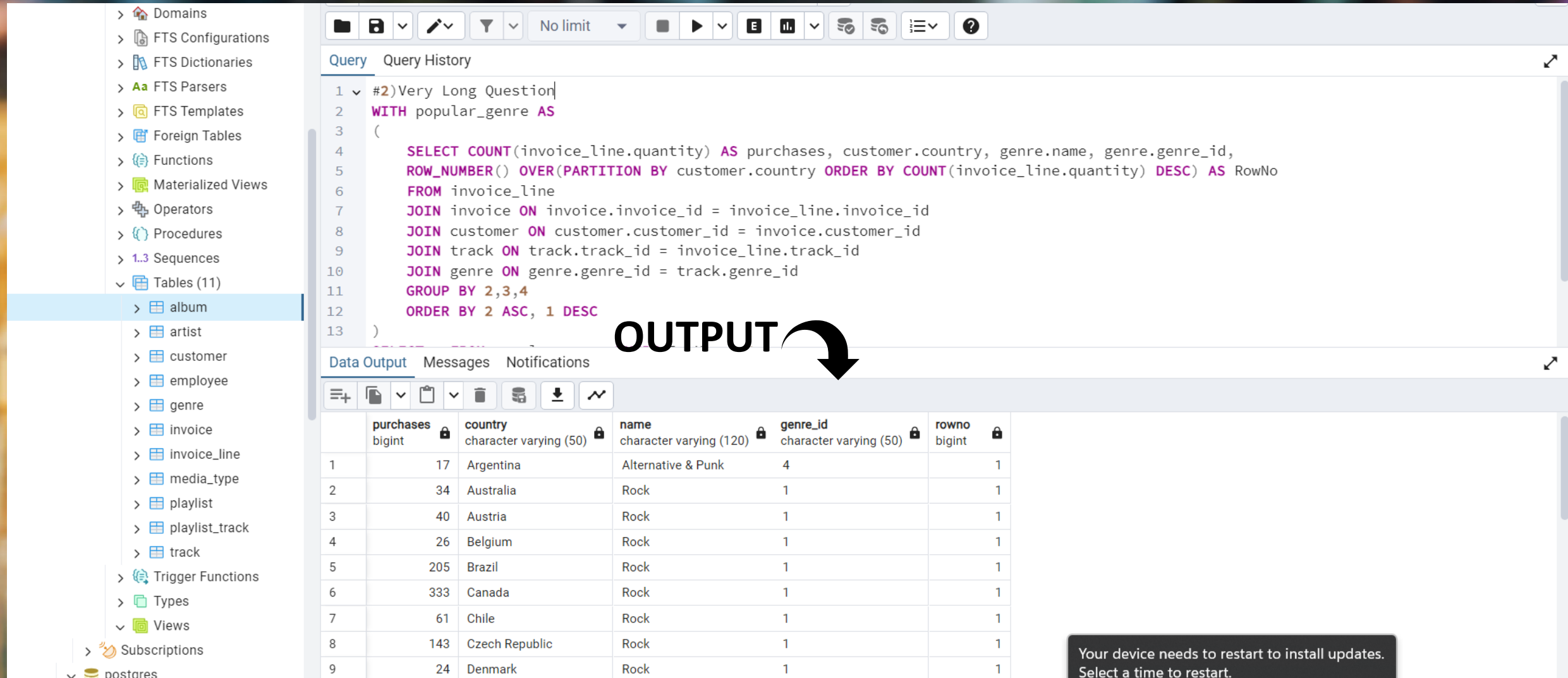
```
34 #1) Very Long Question
35 with best_selling_artist as (
36     select artist.artist_id as artist_id, artist.name as artist_name,
37     sum(invoice_line.unit_price * invoice_line.quantity) as total_sales
38     from invoice_line
39     join track on invoice_line.track_id = track.track_id
40     join album on track.album_id = album.album_id
41     join artist on album.artist_id = artist.artist_id
42     group by 1
43     order by 3 desc
44     limit 1
45 )
46 select customer.customer_id, customer.first_name, customer.last_name, best_selling_artist.artist_name,
47 sum(invoice_line.unit_price * invoice_line.quantity) as amount_spent
48 from invoice
49 join customer on invoice.customer_id = customer.customer_id
50 join invoice_line on invoice.invoice_id = invoice_line.invoice_id
51 join track on invoice_line.track_id = track.track_id
52 join album on track.album_id = album.album_id
53 join best_selling_artist on album.artist_id = best_selling_artist.artist_id
54 group by 1,2,3,4
55 order by 5 desc;
```

Below the query, the 'Data Output' tab shows the results of the query. The output is a table with 7 columns: customer_id, first_name, last_name, artist_name, and amount_spent. The results are ordered by the total amount spent in descending order.

	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

Question Set 3 – Advance

2) We want to find out the most popular music Genre for each country. We determine the most popular genre as the genre with the highest amount of purchases. Write a query that returns each country along with the top Genre. For countries where the maximum number of purchases is shared return all Genres



The screenshot shows a database management interface with a left sidebar containing a tree view of database objects. The 'Tables (11)' folder is expanded, showing tables like album, artist, customer, employee, genre, invoice, invoice_line, media_type, playlist, playlist_track, and track. The 'genre' table is selected.

The main area displays a SQL query in the 'Query' tab. The query is as follows:

```
1 #2) Very Long Question
2 WITH popular_genre AS
3 (
4     SELECT COUNT(invoice_line.quantity) AS purchases, customer.country, genre.name, genre.genre_id,
5           ROW_NUMBER() OVER(PARTITION BY customer.country ORDER BY COUNT(invoice_line.quantity) DESC) AS RowNo
6     FROM invoice_line
7     JOIN invoice ON invoice.invoice_id = invoice_line.invoice_id
8     JOIN customer ON customer.customer_id = invoice.customer_id
9     JOIN track ON track.track_id = invoice_line.track_id
10    JOIN genre ON genre.genre_id = track.genre_id
11    GROUP BY 2,3,4
12    ORDER BY 2 ASC, 1 DESC
13 )
```

The 'Data Output' tab shows the results of the query. The output is a table with 9 rows and 6 columns: purchases (bigint), country (character varying (50)), name (character varying (120)), genre_id (character varying (50)), rowno (bigint), and an unnamed column. The data is as follows:

	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

A large black arrow labeled 'OUTPUT' points from the query to the data output table.

A notification at the bottom right states: 'Your device needs to restart to install updates. Select a time to restart.'

Question Set 3 – Advance

3) Write a query that determines the customer that has spent the most on music for each country. Write a query that returns the country along with the top customer and how much they spent. For countries where the top amount spent is shared, provide all customers who spent this amount

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

Query Query History

#3)Very Long Question

WITH Customer_with_country AS (
SELECT customer.customer_id,first_name,last_name,billing_country,SUM(total) AS total_spending,
ROW_NUMBER() OVER(PARTITION BY billing_country ORDER BY SUM(total) DESC) AS RowNo
FROM invoice
JOIN customer ON customer.customer_id = invoice.customer_id
GROUP BY 1,2,3,4
ORDER BY 4 ASC,5 DESC)
SELECT * FROM Customer_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	27	Evan	Zimmerman	Germany	84.95000000000001	1

OUTPUT

THE
-
O
I
D
O
Y
I
S
O
R
Q
D
X

THANK

YOU