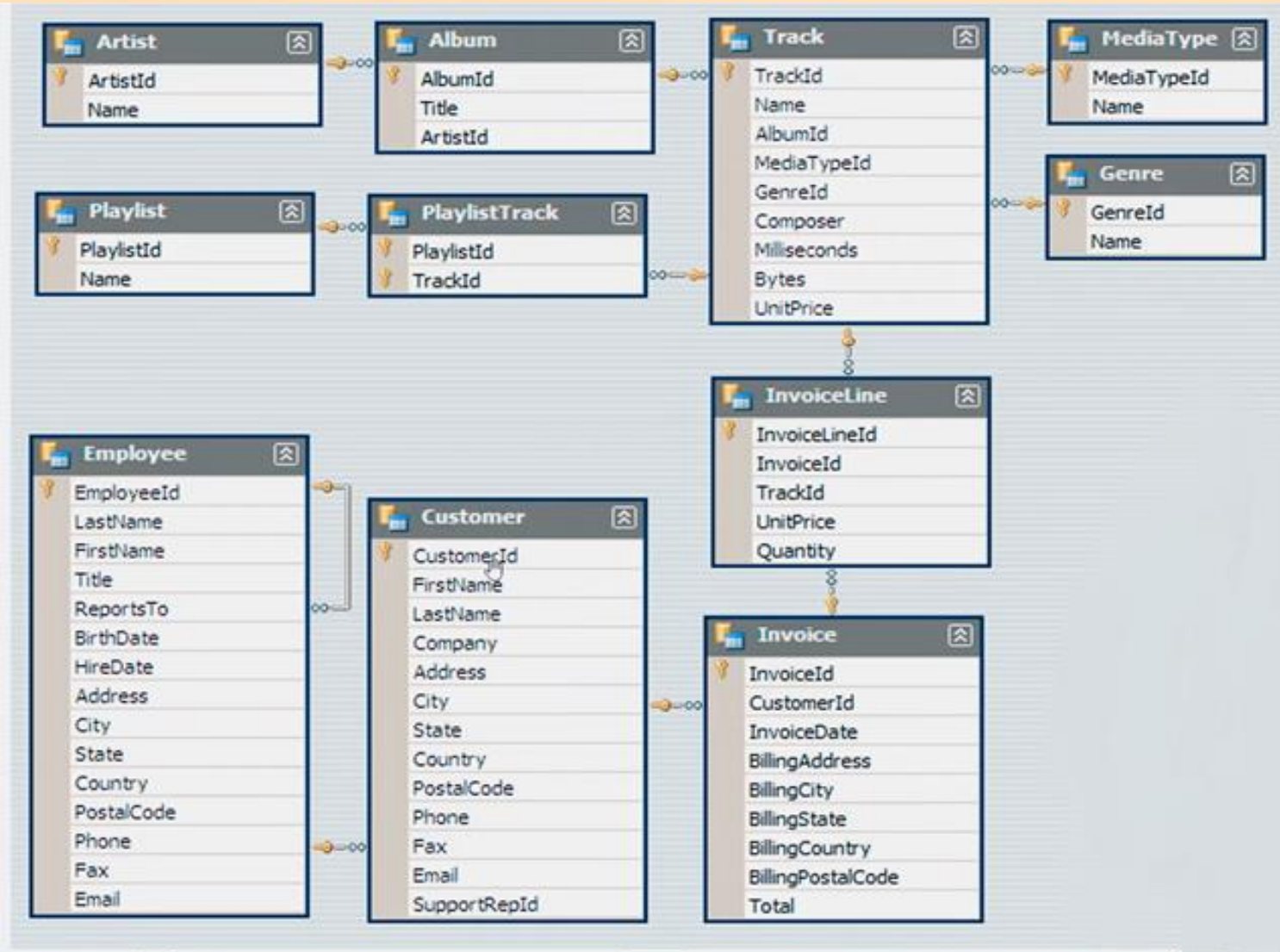


The background is a light orange and yellow gradient with various musical instruments and notes scattered around. In the top left, there's a violin. To its right is a trumpet. On the left side, there's a saxophone. In the bottom left, there's a xylophone with colorful keys. In the bottom right, there's a drum with two drumsticks. Several musical notes are floating around the instruments.

# Music Store Data Analysis Using SQL

# Schema of the Model





# QUESTION 1:

Who is the senior most employee based on job title?

## Query And Output:

The screenshot shows a database management tool interface. On the left is a sidebar with a tree view of databases. The 'music\_database' database is selected, showing its contents: Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrapper, Languages, Publications, Schemas, and Subscriptions. Below this, the 'postgres' database is also visible. The main area is divided into two tabs: 'Query' and 'Query History'. The 'Query' tab is active, displaying a SQL query:

```
1 SELECT title, last_name, first_name
2 FROM employee
3 ORDER BY levels DESC
4 LIMIT 1
5
```

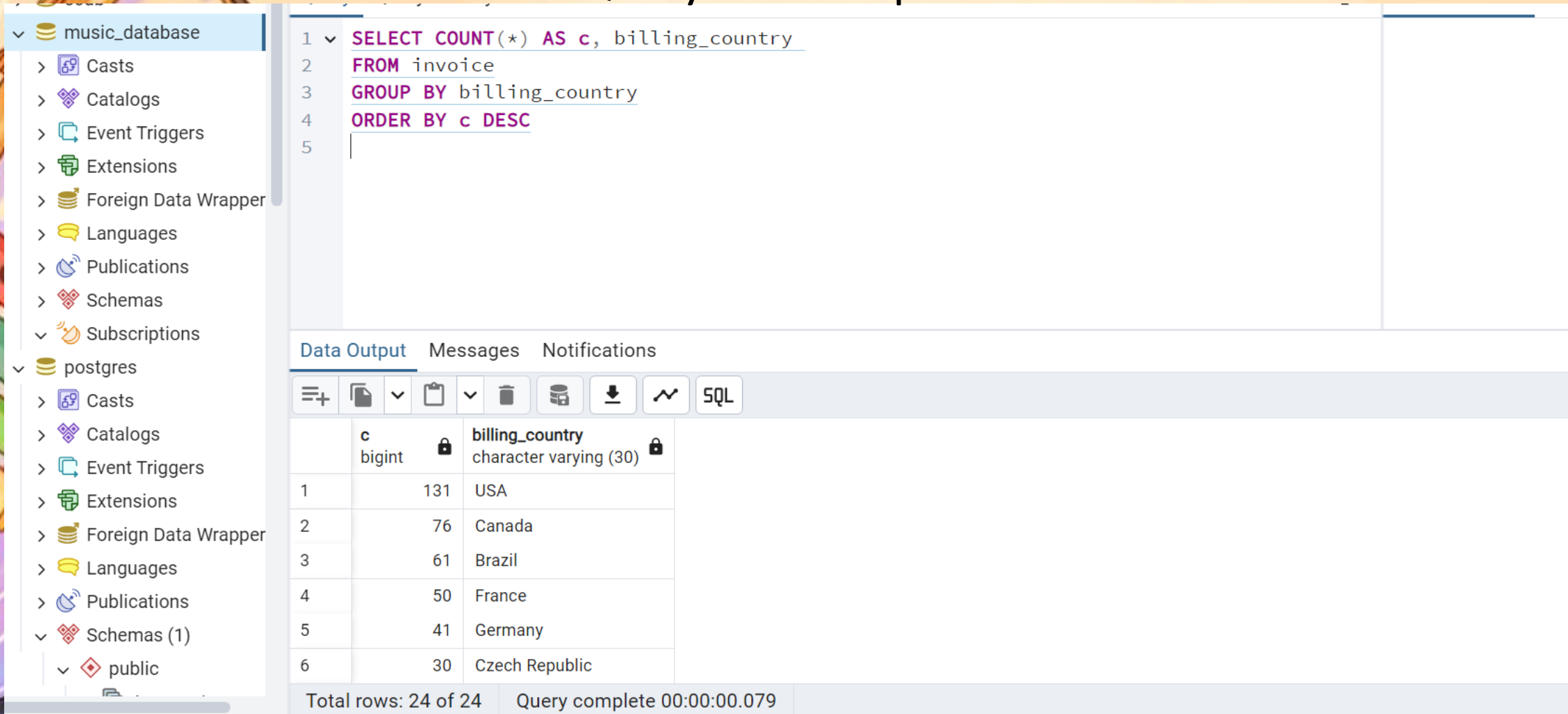
Below the query editor, there are three tabs: 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing the results of the query in a table format. The table has three columns: 'title', 'last\_name', and 'first\_name'. The first row of data shows 'Senior General Manager', 'Madan', and 'Mohan'.

	title character varying (50) 🔒	last_name character 🔒	first_name character 🔒
1	Senior General Manager	Madan	Mohan

## QUESTION 2:

Which countries have the most Invoices?

### Query And Output:



The screenshot shows a database client interface with a sidebar on the left and a main query editor area on the right. The sidebar lists the database structure, including 'music\_database' and 'postgres' schemas. The main area displays a SQL query to count invoices by country, ordered by count in descending order. Below the query editor, there is a 'Data Output' tab showing the results of the query as a table with 6 rows. The table has columns 'c' (count) and 'billing\_country'. The results show that the USA has the highest number of invoices (131), followed by Canada (76), Brazil (61), France (50), Germany (41), and the Czech Republic (30). The status bar at the bottom indicates 'Total rows: 24 of 24' and 'Query complete 00:00:00.079'.

```
1 SELECT COUNT(*) AS c, billing_country
2 FROM invoice
3 GROUP BY billing_country
4 ORDER BY c DESC
5
```

	c	billing_country
1	131	USA
2	76	Canada
3	61	Brazil
4	50	France
5	41	Germany
6	30	Czech Republic

Total rows: 24 of 24    Query complete 00:00:00.079

# QUESTION 3: What are top 3 values of total invoice?

## Query And Output:

The screenshot shows the PostgreSQL 16 interface. On the left, the 'Databases (4)' list includes 'ccdb', 'music\_database', and 'postgres'. The 'music\_database' is selected, showing its contents: 'Casts', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrapper', 'Languages', 'Publications', 'Schemas', and 'Subscriptions'. The 'postgres' database is also expanded, showing 'Casts', 'Catalogs', 'Event Triggers', 'Extensions', 'Foreign Data Wrapper', 'Languages', 'Publications', 'Schemas (1)', and 'public'.

The main query editor displays the following SQL query:

```
1 SELECT total
2 FROM invoice
3 ORDER BY total DESC
4
```

The 'Data Output' tab shows the results of the query. The output is a table with two columns: 'total' (double precision) and a lock icon. The results are sorted in descending order of 'total'.

	total	
	double precision	
9	17.82	
10	17.82	
11	17.82	
12	17.82	
13	17.82	
14	16.83	

The status bar at the bottom indicates 'Total rows: 614 of 614' and 'Query complete 00:00:00.086'.



## QUESTION 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

### Query And Output:

The screenshot shows a database client interface with a sidebar on the left listing databases and schemas. The main area displays a SQL query and its output.

**Query:**

```
1 SELECT billing_city, SUM(total) AS InvoiceTotal
2 FROM invoice
3 GROUP BY billing_city
4 ORDER BY InvoiceTotal DESC
5 LIMIT 1;
```

**Data Output:**

	billing_city character varying (30)	invoicetotal double precision
1	Prague	273.240000000000007

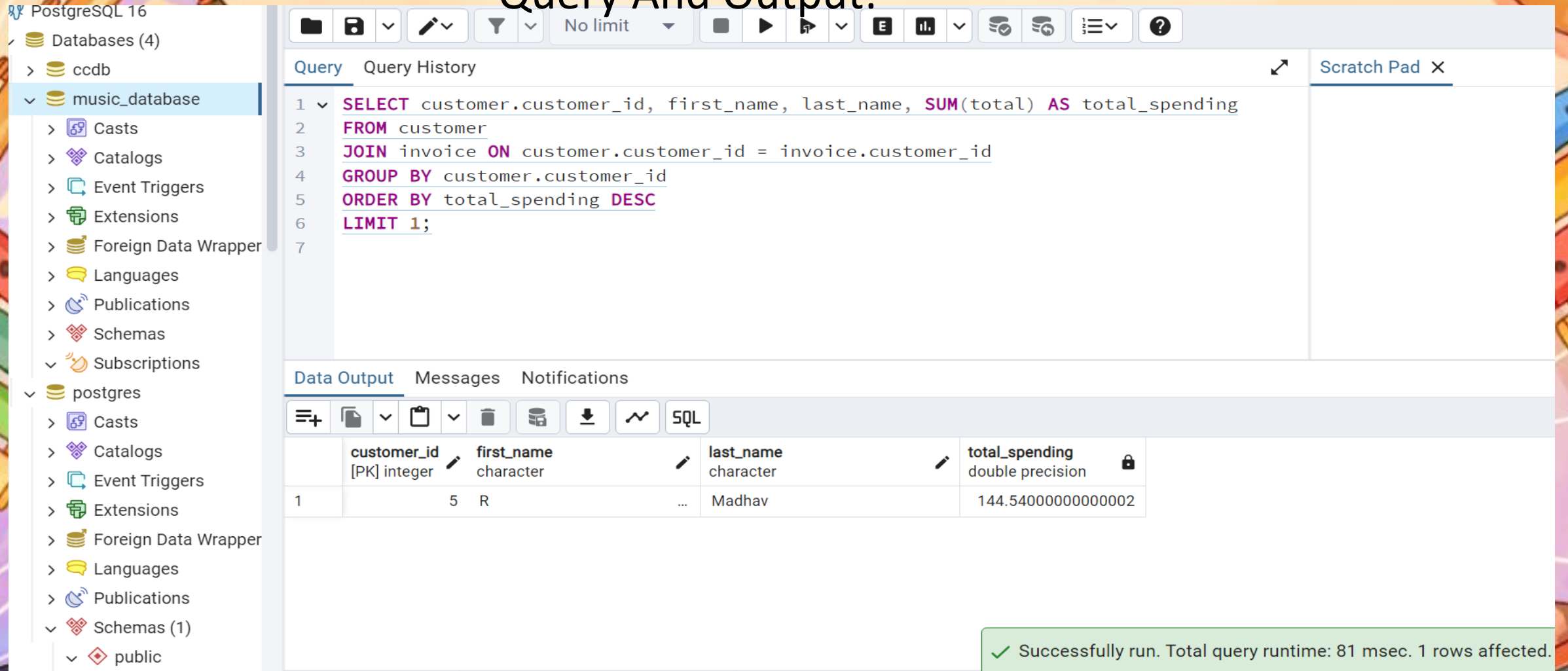
Total rows: 1 of 1    Query complete 00:00:00.109

## QUESTION 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.

### Query And Output:



The screenshot shows the PostgreSQL 16 interface. On the left, the 'Databases (4)' sidebar is expanded, showing 'ccdb' and 'music\_database'. The 'music\_database' is selected, and its contents are listed: Casts, Catalogs, Event Triggers, Extensions, Foreign Data Wrapper, Languages, Publications, Schemas, and Subscriptions. The main query editor displays the following SQL query:

```
1 SELECT customer.customer_id, first_name, last_name, SUM(total) AS total_spending
2 FROM customer
3 JOIN invoice ON customer.customer_id = invoice.customer_id
4 GROUP BY customer.customer_id
5 ORDER BY total_spending DESC
6 LIMIT 1;
7
```

Below the query editor, the 'Data Output' tab is active, showing a table with the following data:

	customer_id [PK] integer	first_name character	last_name character	total_spending double precision
1	5	R	Madhav	144.54000000000002

At the bottom right, a green status bar indicates: ✓ Successfully run. Total query runtime: 81 msec. 1 rows affected.

**QUESTION 6:** 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.

## Query And Output:

The screenshot shows a database management interface. On the left is a sidebar with a tree view of database objects: FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (11). The 'Tables (11)' folder is expanded, showing a list of tables: album, artist, customer, employee, genre, invoice, invoice\_line, media\_type, playlist, playlist\_track, and track. The main area is divided into two panes. The top pane, titled 'Query', contains an SQL query. The bottom pane, titled 'Data Output', shows the results of the query in a table format. The query is as follows:

```
1 SELECT DISTINCT email, first_name, last_name
2 FROM customer
3 JOIN invoice ON customer.customer_id = invoice.customer_id
4 JOIN invoice_line ON invoice.invoice_id = invoice_line.invoice_id
5 WHERE track_id IN(
6     SELECT track_id FROM track
7     JOIN genre ON track.genre_id = genre.genre_id
8     WHERE genre.name LIKE 'Rock'
9 )
10 ORDER BY email;
```

The 'Data Output' pane displays the following table:

	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 7:** 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.

## Query And Output:

The screenshot shows a database management interface. On the left is a sidebar with a tree view of database objects: FTS Dictionaries, FTS Parsers, FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (11). The 'Tables (11)' folder is expanded, showing tables: album, artist, customer, employee, genre, invoice, invoice\_line, media\_type, playlist, playlist\_track, and track. The main area is divided into three panes. The top pane, titled 'Query', contains an SQL query: 

```
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;
```

 The middle pane, titled 'Data Output', shows the results of the query in a table. The table has three columns: 'artist\_id' (character varying (50)), 'name' (character), and 'number\_of\_songs' (bigint). The results are: 1. Led Zeppelin (114 songs), 2. U2 (112 songs), 3. Deep Purple (92 songs), 4. Iron Maiden (81 songs), 5. Pearl Jam (54 songs), 6. Van Halen (52 songs). The bottom pane shows 'Total rows: 10 of 10' and 'Query complete 00:00:00.091'.

**Query**

```
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**

	artist_id [PK] character varying (50)	name character	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

Total rows: 10 of 10    Query complete 00:00:00.091

## QUESTION 8:

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.

### Query And Output:

The screenshot shows a database management interface. On the left is a sidebar with a tree view of database objects: FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (11). The 'Tables (11)' folder is expanded, showing a list of tables: album, artist, customer, employee, genre, invoice, invoice\_line, media\_type, playlist, playlist\_track, and track. The 'track' table is selected. The main area is divided into three tabs: 'Query', 'Query History', and 'Scratch Pad'. The 'Query' tab is active, displaying an SQL query:

```
1 SELECT name, milliseconds
2 FROM track
3 WHERE milliseconds > (
4     SELECT AVG(milliseconds) AS avg_track_length
5     FROM track )
6 ORDER BY milliseconds DESC;
```

Below the query editor are tabs for 'Data Output', 'Messages', and 'Notifications'. The 'Data Output' tab is active, showing a table of results. The table has two columns: 'name' (character varying (150)) and 'milliseconds' (integer). The results are ordered by milliseconds in descending order.

	name	milliseconds
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 Grass	2826502