MUSIC STORE SQL ANALYSIS

BUSINESS INSIGHTS USING SQL QUERIES

PREPARED BY

SAKSHAM

DATASET OVERVIEW

THIS PROJECT USES A MUSIC STORE DATABASE CONTAINING TRANSACTIONAL AND REFERENCE DATA RELATED TO CUSTOMERS, EMPLOYEES, INVOICES, TRACKS, AND ARTISTS.

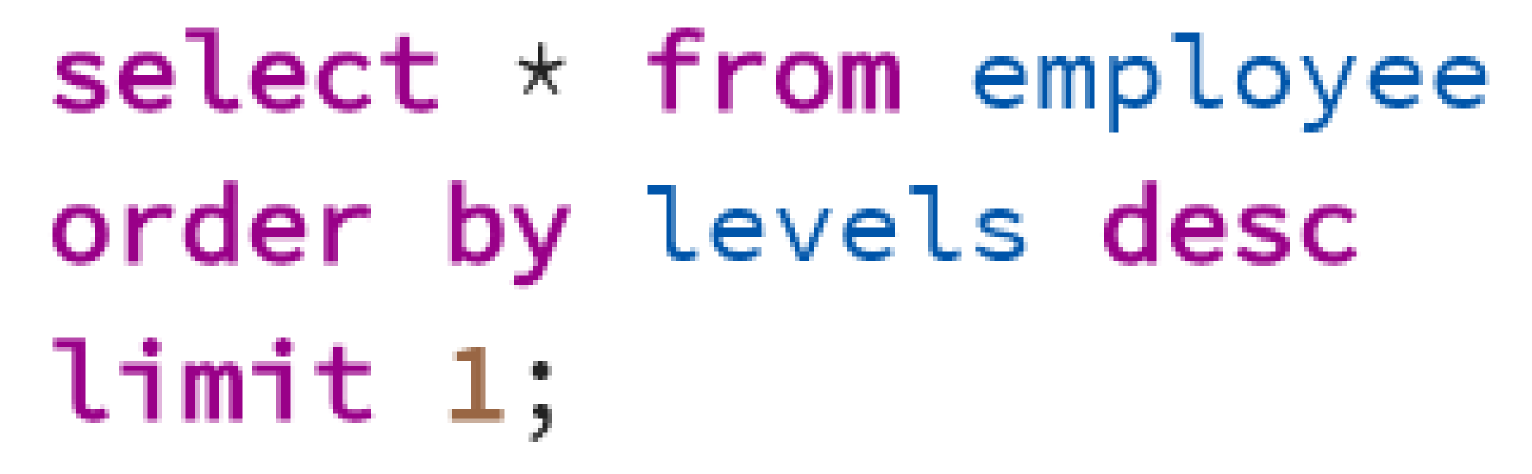
KEY TABLES

- EMPLOYEE
- CUSTOMER
- INVOICE
- INVOICE_LINE
- TRACK
- ALBUM
- ARTIST
- GENRE

OBJECTIVE

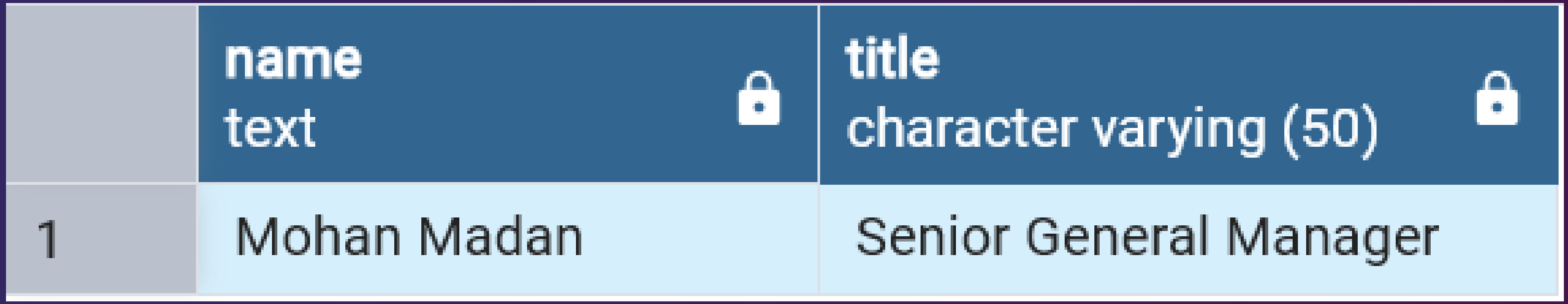
TO ANALYZE SALES, CUSTOMER BEHAVIOR, AND MUSIC PREFERENCES USING SQL.

IDENTIFY THE MOST SENIOR EMPLOYEE IN THE ORGANIZATION.



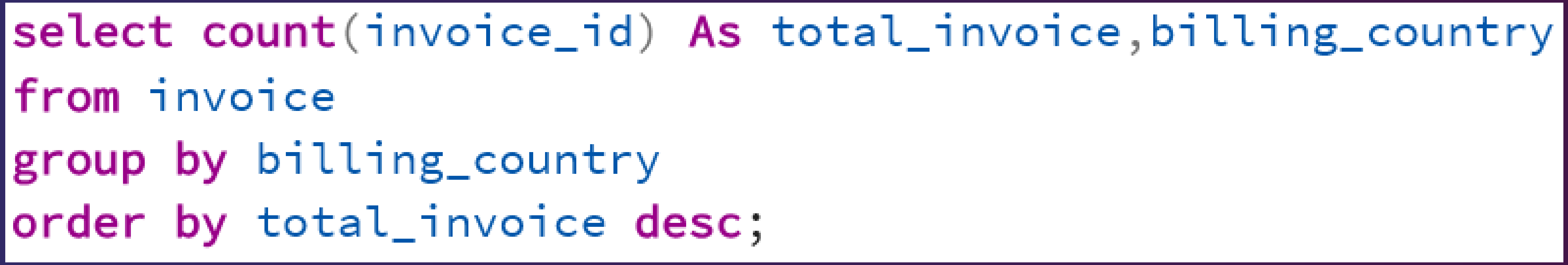
```sql
select * from employee
order by levels desc
limit 1;
```

Output



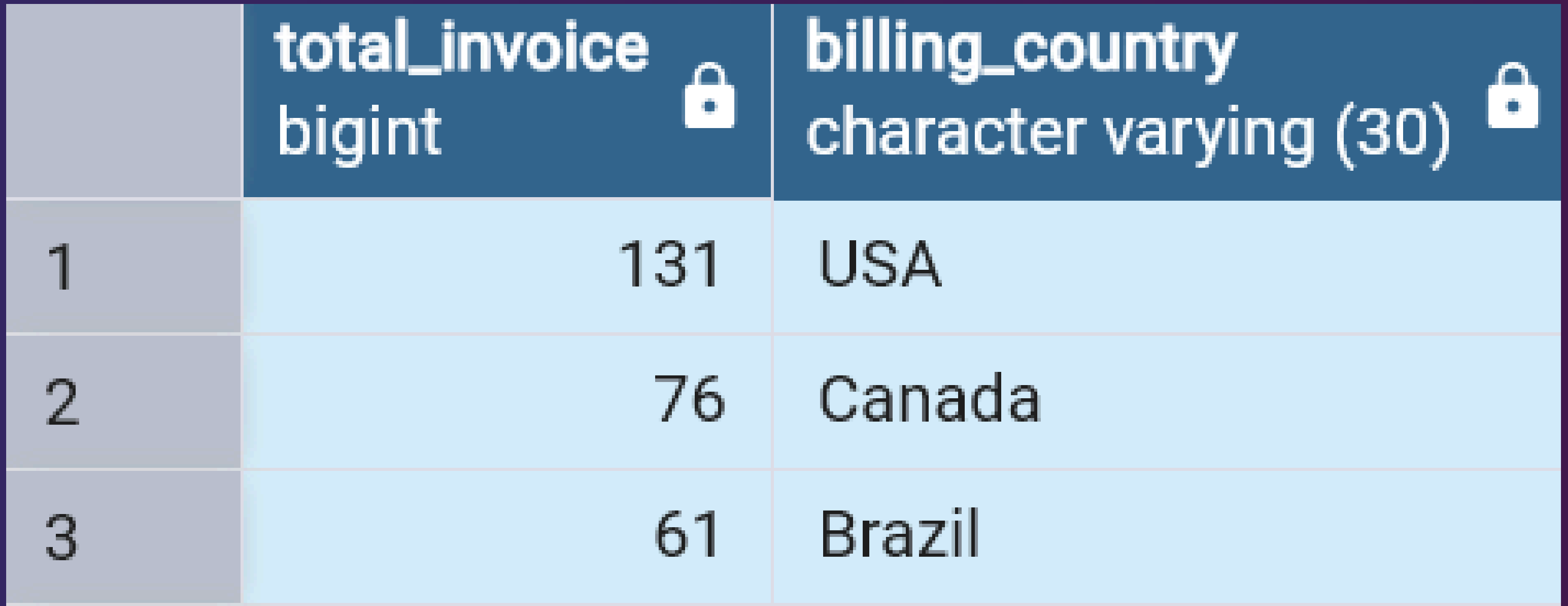
	name text	title character varying (50)
1	Mohan Madan	Senior General Manager

FIND THE NUMBER OF INVOICES GENERATED IN EACH BILLING COUNTRY.



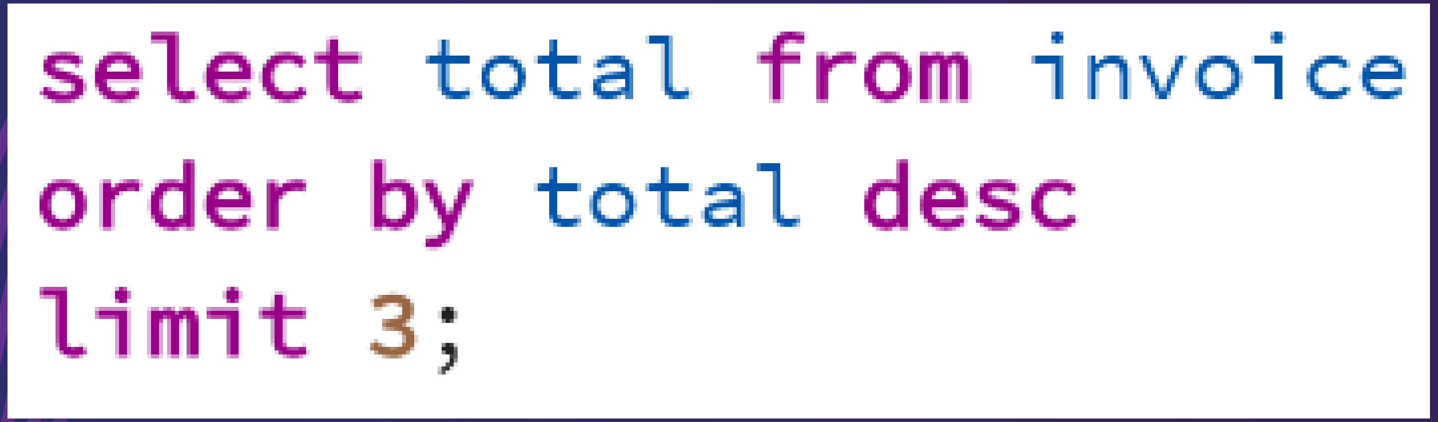
```
select count(invoice_id) As total_invoice,billing_country
from invoice
group by billing_country
order by total_invoice desc;
```

Output



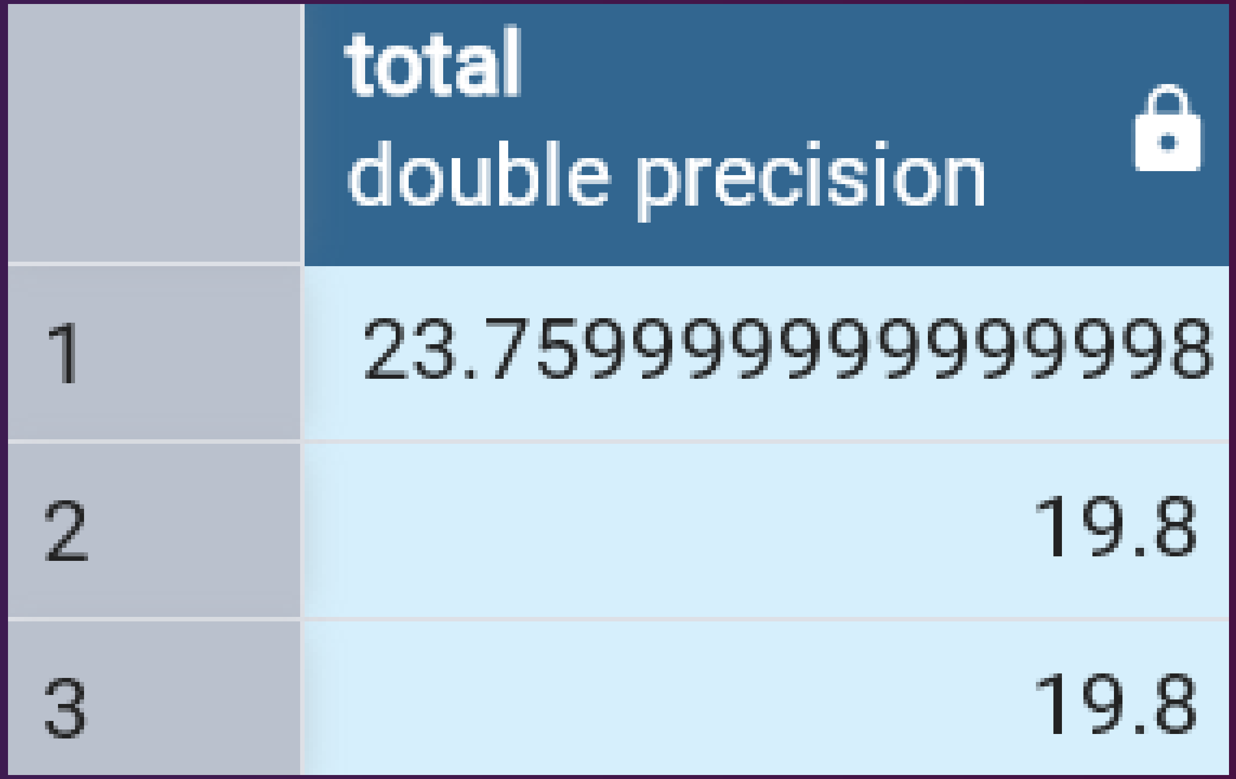
	total_invoice bigint	billing_country character varying (30)
1	131	USA
2	76	Canada
3	61	Brazil

IDENTIFY THE HIGHEST INVOICE TOTALS IN THE DATABASE



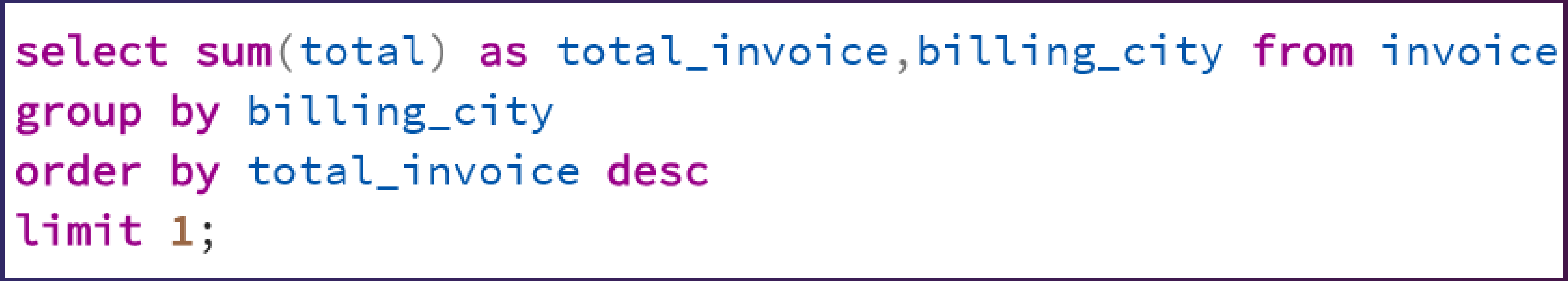
```
select total from invoice
order by total desc
limit 3;
```

Output



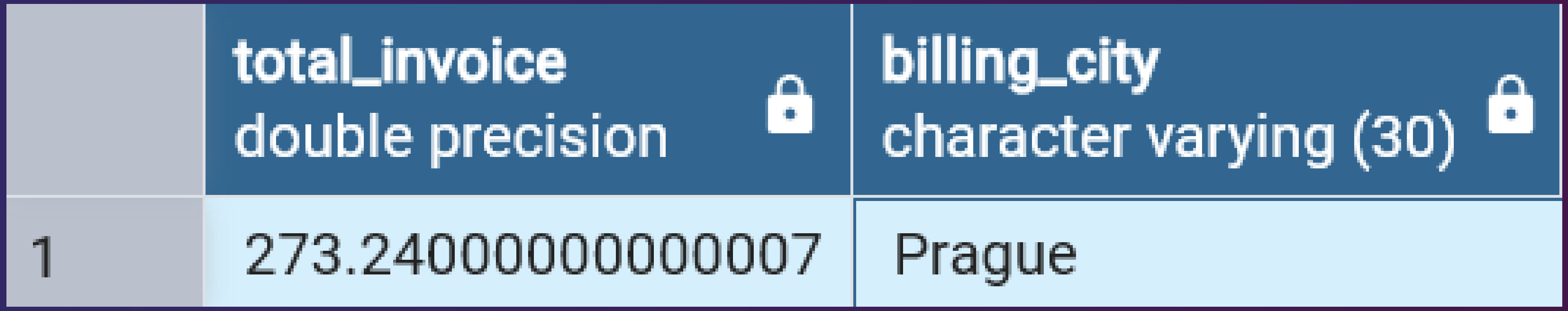
	total double precision
1	23.759999999999998
2	19.8
3	19.8

DETERMINE WHICH CITY GENERATED THE HIGHEST TOTAL INVOICE REVENUE.



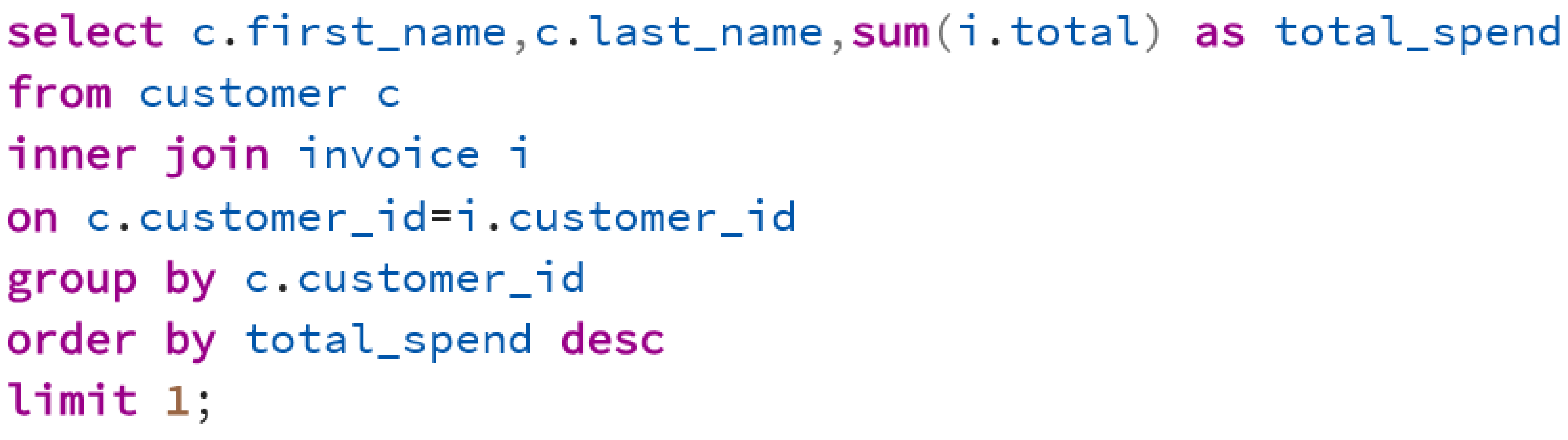
```
select sum(total) as total_invoice,billing_city from invoice
group by billing_city
order by total_invoice desc
limit 1;
```

Output



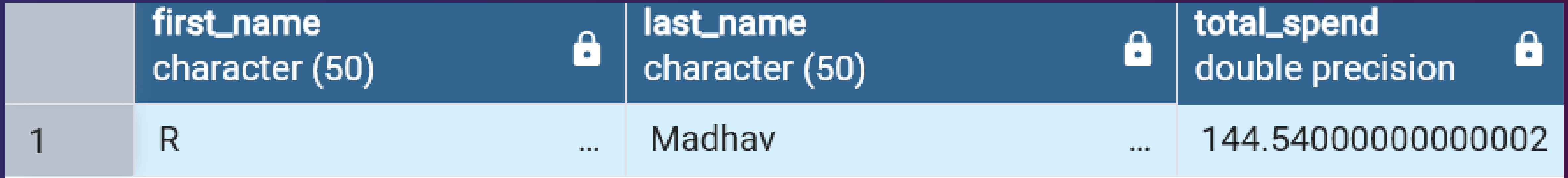
	total_invoice double precision	billing_city character varying (30)
1	273.24000000000007	Prague

IDENTIFY THE CUSTOMER WHO SPENT THE MOST MONEY



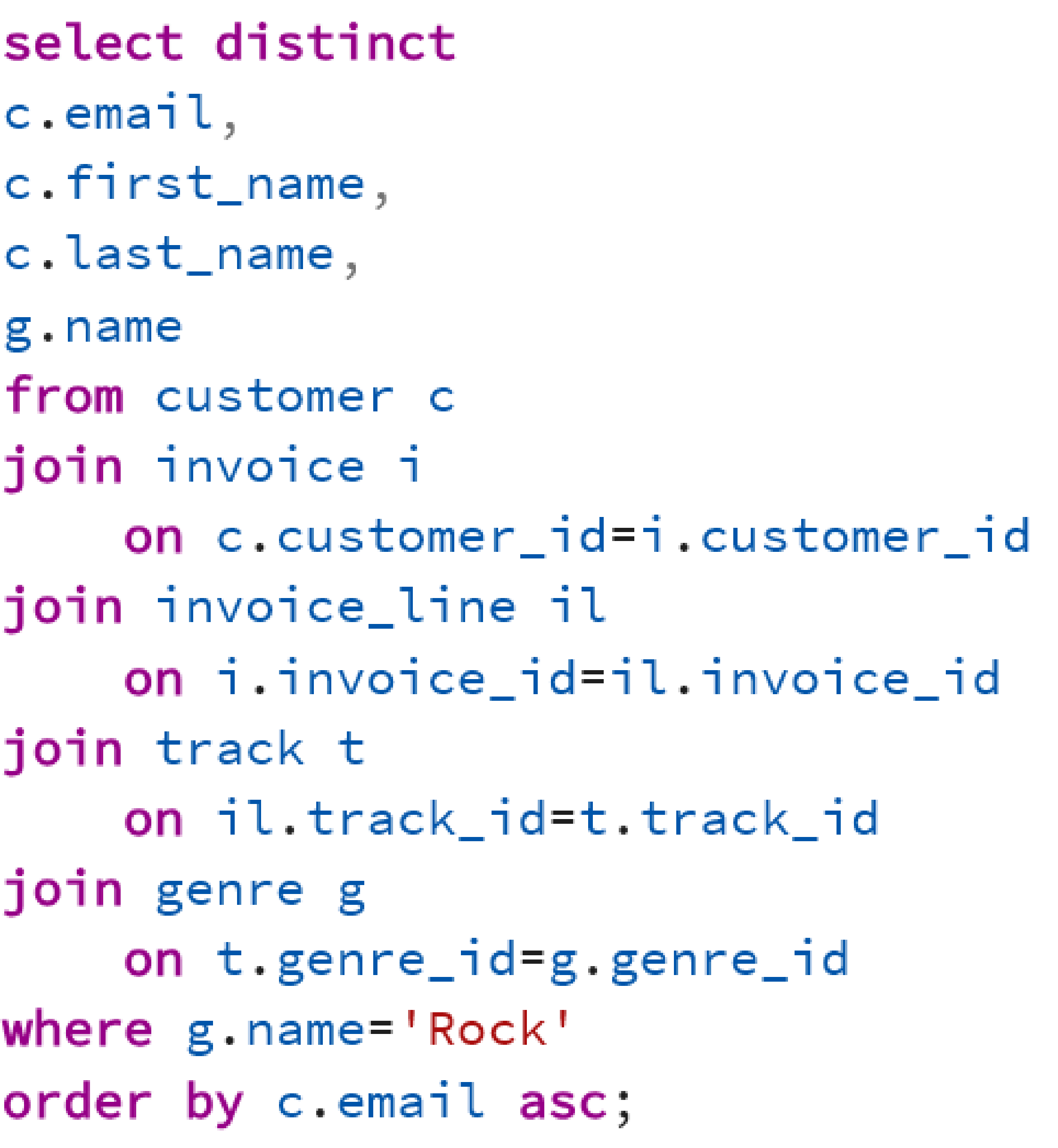
```sql
select c.first_name,c.last_name,sum(i.total) as total_spend
from customer c
inner join invoice i
on c.customer_id=i.customer_id
group by c.customer_id
order by total_spend desc
limit 1;
```

Output



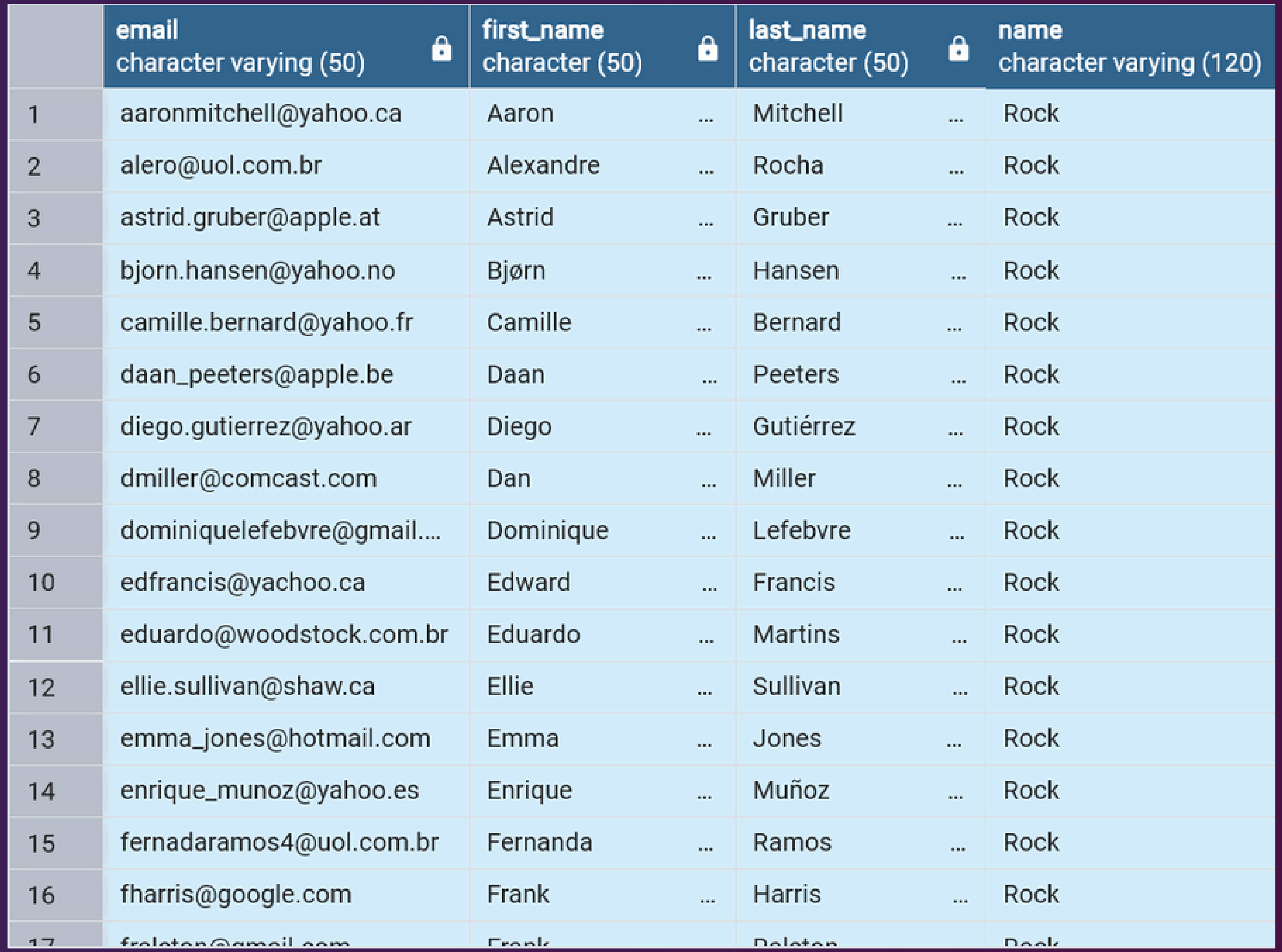
	first_name character (50)	last_name character (50)	total_spend double precision
1	R	Madhav	144.54000000000002

FIND CUSTOMERS WHO PURCHASED TRACKS FROM THE ROCK GENRE



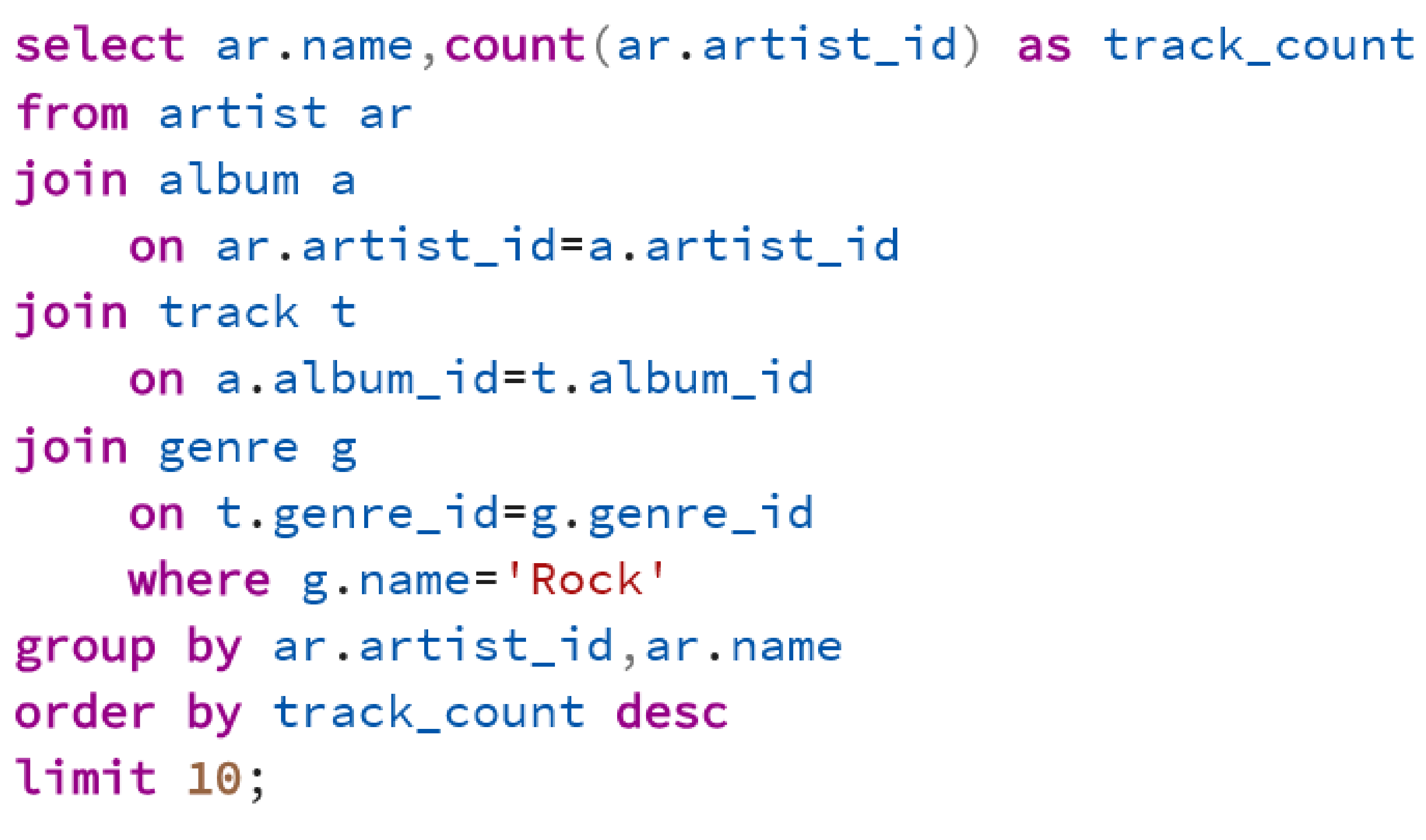
```
select distinct
c.email,
c.first_name,
c.last_name,
g.name
from customer c
join invoice i
    on c.customer_id=i.customer_id
join invoice_line il
    on i.invoice_id=il.invoice_id
join track t
    on il.track_id=t.track_id
join genre g
    on t.genre_id=g.genre_id
where g.name='Rock'
order by c.email asc;
```

Output



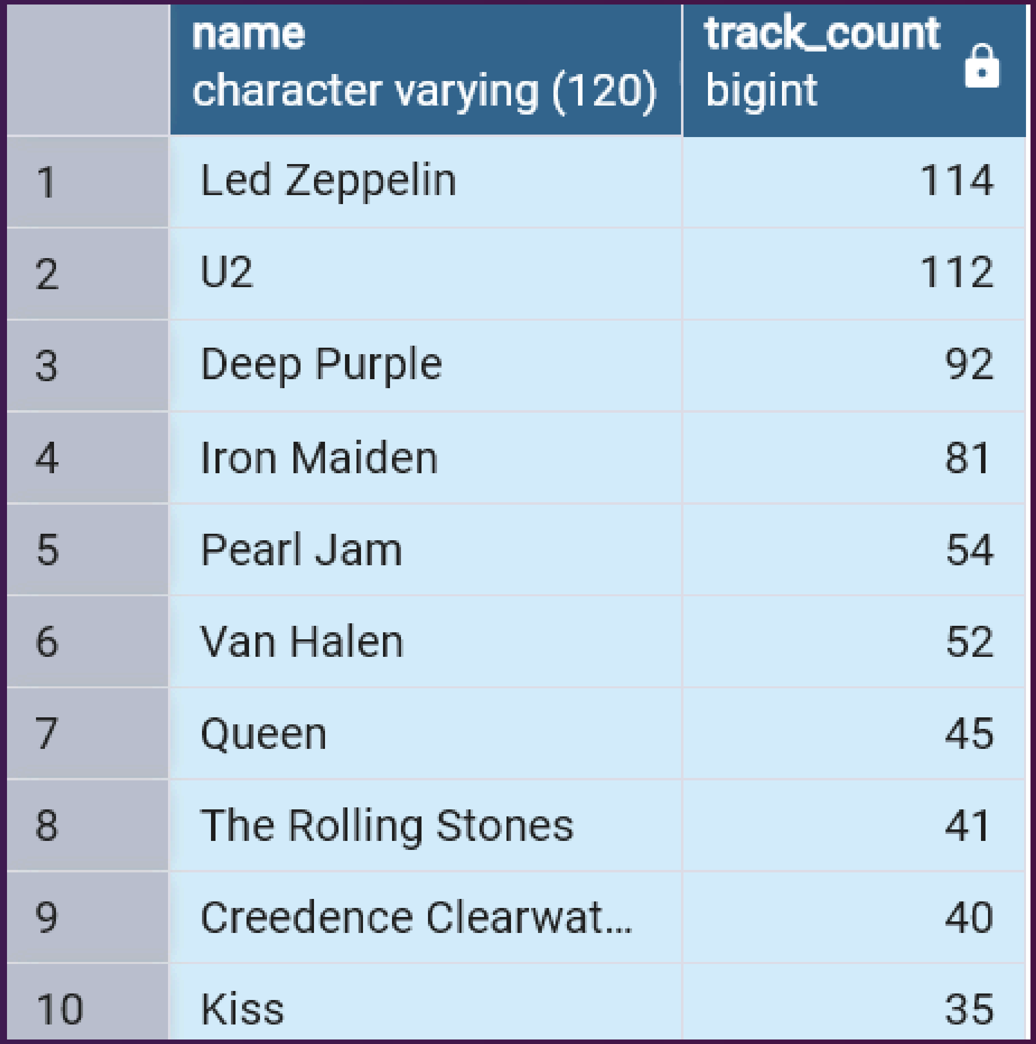
	email character varying (50)	first_name character (50)	last_name character (50)	name character varying (120)
1	aaronmitchell@yahoo.ca	Aaron	Mitchell	Rock
2	alero@uol.com.br	Alexandre	Rocha	Rock
3	astrid.gruber@apple.at	Astrid	Gruber	Rock
4	bjorn.hansen@yahoo.no	Bjørn	Hansen	Rock
5	camille.bernard@yahoo.fr	Camille	Bernard	Rock
6	daan_peeters@apple.be	Daan	Peeters	Rock
7	diego.gutierrez@yahoo.ar	Diego	Gutiérrez	Rock
8	dmiller@comcast.com	Dan	Miller	Rock
9	dominiquelefebvre@gmail....	Dominique	Lefebvre	Rock
10	edfrancis@yachoo.ca	Edward	Francis	Rock
11	eduardo@woodstock.com.br	Eduardo	Martins	Rock
12	ellie.sullivan@shaw.ca	Ellie	Sullivan	Rock
13	emma_jones@hotmail.com	Emma	Jones	Rock
14	enrique_munoz@yahoo.es	Enrique	Muñoz	Rock
15	fernadaramos4@uol.com.br	Fernanda	Ramos	Rock
16	fharris@google.com	Frank	Harris	Rock
17	fralston@gmail.com	Frank	Ralston	Rock

IDENTIFY TOP 10 ARTISTS WHO HAVE THE HIGHEST NUMBER OF ROCK TRACKS



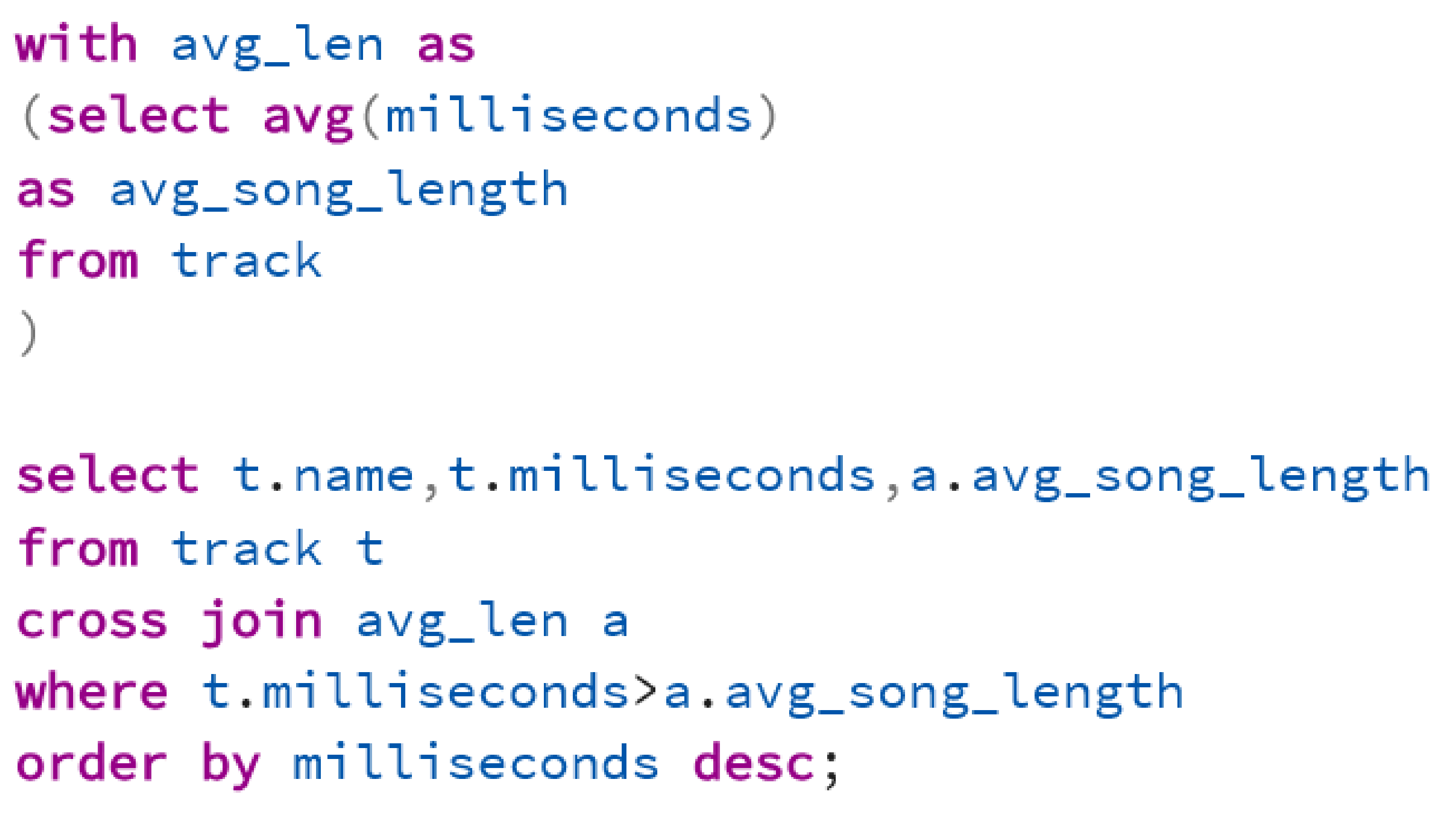
```sql
select ar.name,count(ar.artist_id) as track_count
from artist ar
join album a
    on ar.artist_id=a.artist_id
join track t
    on a.album_id=t.album_id
join genre g
    on t.genre_id=g.genre_id
    where g.name='Rock'
group by ar.artist_id,ar.name
order by track_count desc
limit 10;
```

Output



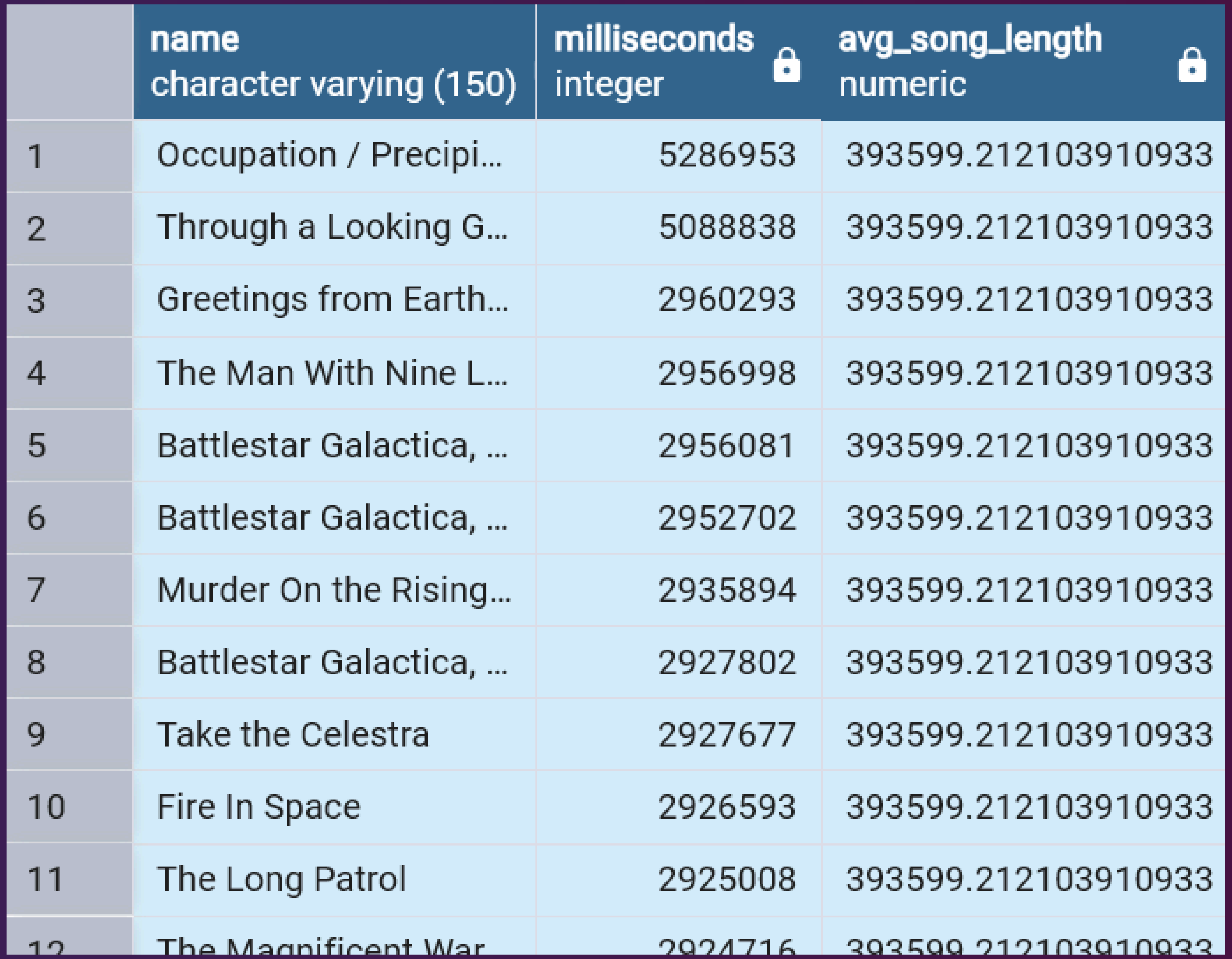
	name character varying (120)	track_count bigint
1	Led Zeppelin	114
2	U2	112
3	Deep Purple	92
4	Iron Maiden	81
5	Pearl Jam	54
6	Van Halen	52
7	Queen	45
8	The Rolling Stones	41
9	Creedence Clearwat...	40
10	Kiss	35

FIND TRACKS THAT ARE LONGER THAN THE AVERAGE TRACK LENGTH



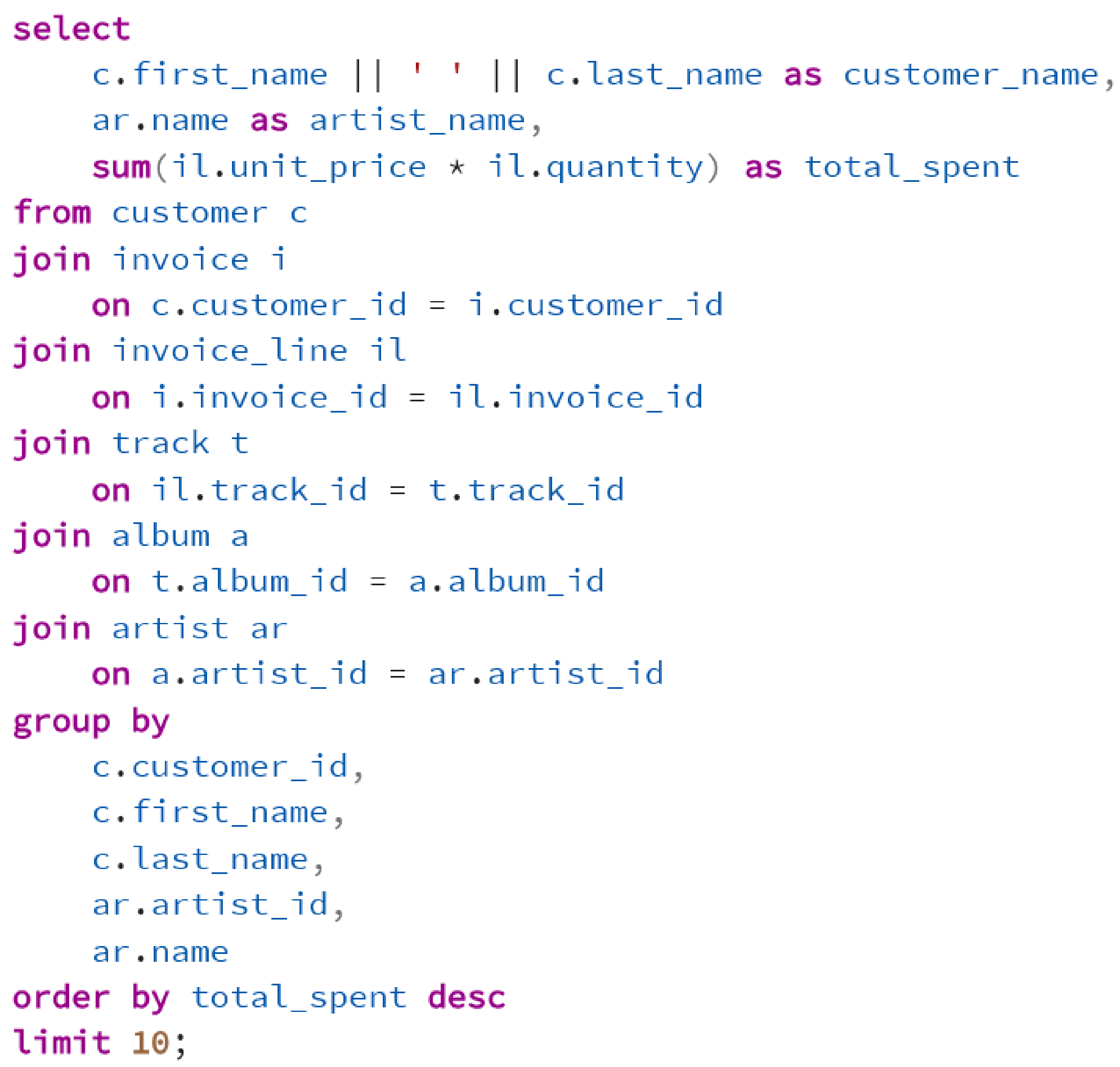
```
with avg_len as
(select avg(milliseconds)
as avg_song_length
from track
)

select t.name,t.milliseconds,a.avg_song_length
from track t
cross join avg_len a
where t.milliseconds>a.avg_song_length
order by milliseconds desc;
```

Output



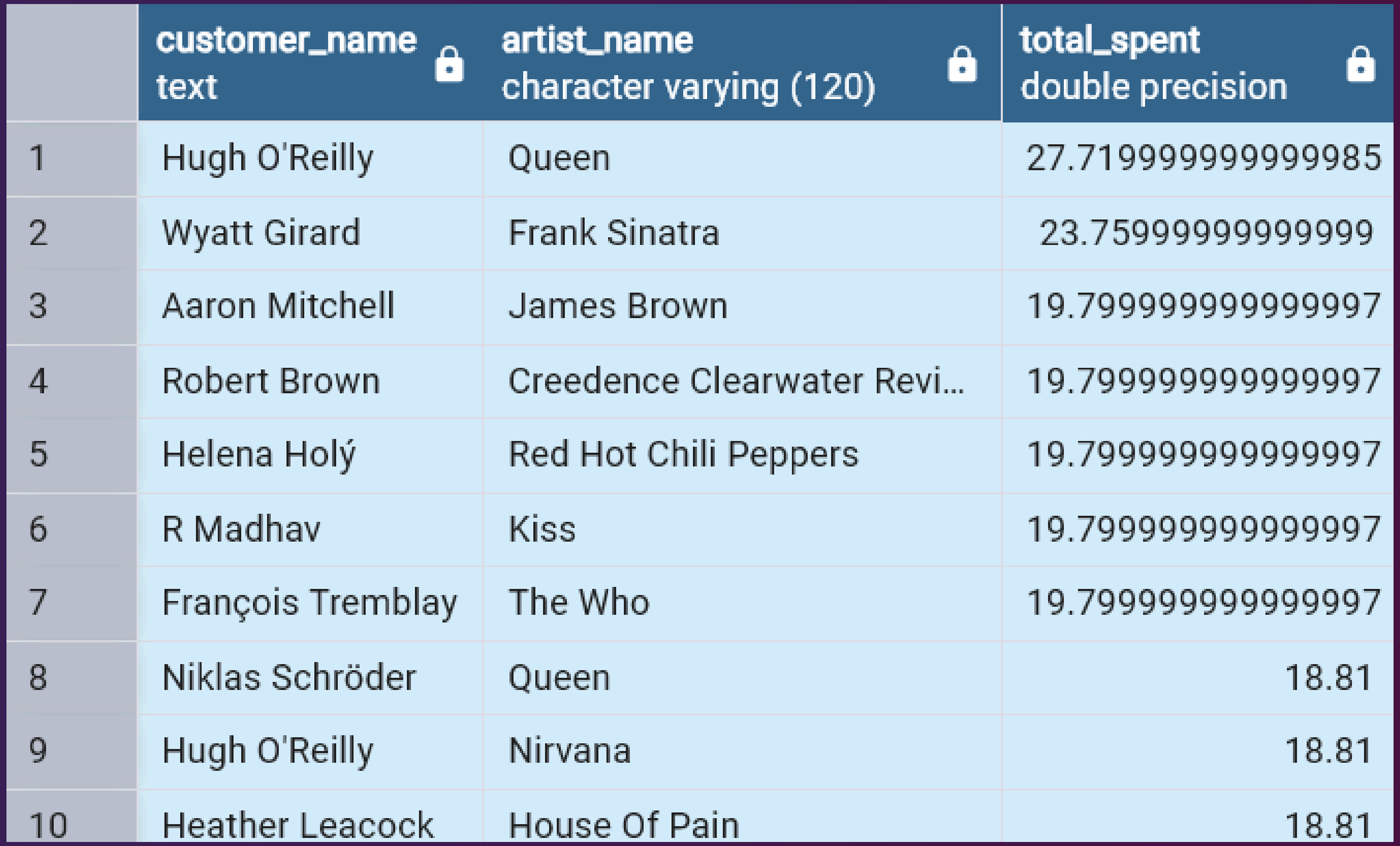
	name character varying (150)	milliseconds integer	avg_song_length numeric
1	Occupation / Precipi...	5286953	393599.212103910933
2	Through a Looking G...	5088838	393599.212103910933
3	Greetings from Earth...	2960293	393599.212103910933
4	The Man With Nine L...	2956998	393599.212103910933
5	Battlestar Galactica, ...	2956081	393599.212103910933
6	Battlestar Galactica, ...	2952702	393599.212103910933
7	Murder On the Rising...	2935894	393599.212103910933
8	Battlestar Galactica, ...	2927802	393599.212103910933
9	Take the Celestra	2927677	393599.212103910933
10	Fire In Space	2926593	393599.212103910933
11	The Long Patrol	2925008	393599.212103910933
12	The Magnificent War...	2924716	393599.212103910933

CALCULATE HOW MUCH EACH CUSTOMER HAS SPENT ON DIFFERENT ARTISTS



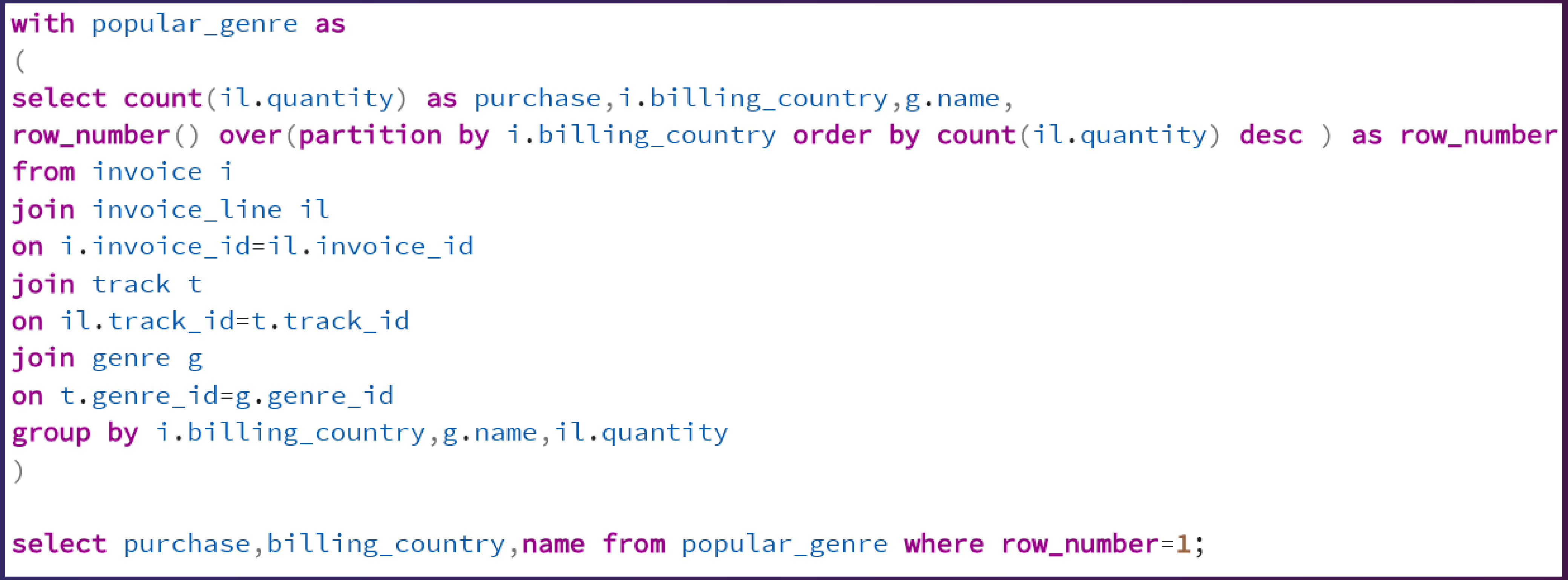
```
select
    c.first_name || ' ' || c.last_name as customer_name,
    ar.name as artist_name,
    sum(il.unit_price * il.quantity) as total_spent
from customer c
join invoice i
    on c.customer_id = i.customer_id
join invoice_line il
    on i.invoice_id = il.invoice_id
join track t
    on il.track_id = t.track_id
join album a
    on t.album_id = a.album_id
join artist ar
    on a.artist_id = ar.artist_id
group by
    c.customer_id,
    c.first_name,
    c.last_name,
    ar.artist_id,
    ar.name
order by total_spent desc
limit 10;
```

Output



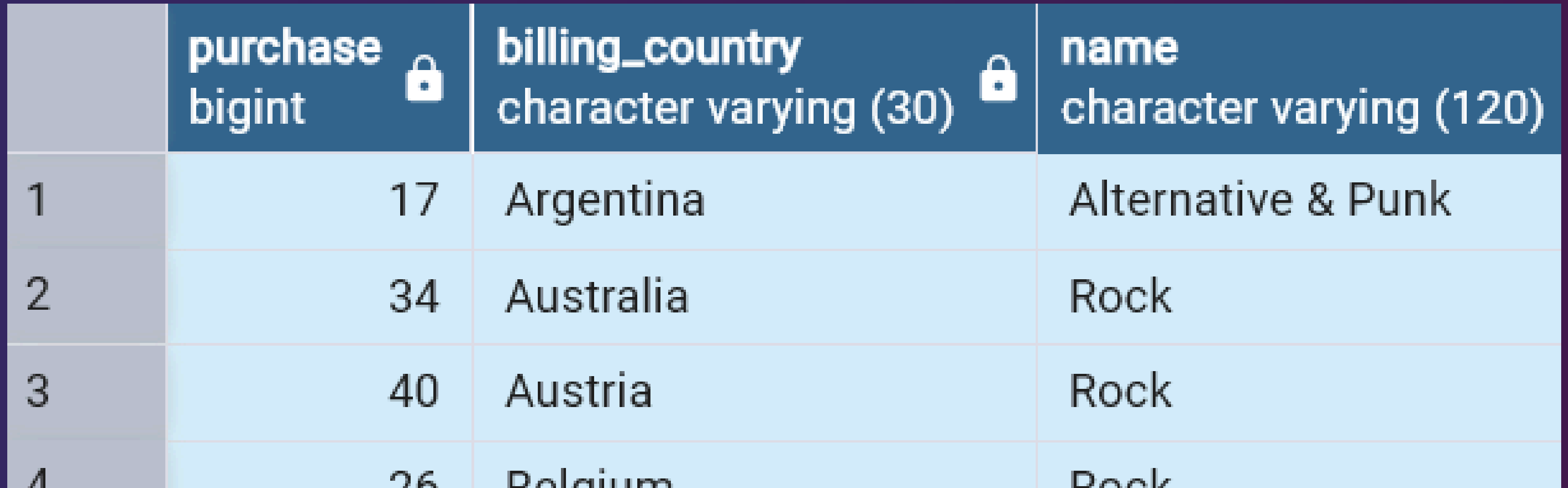
	customer_name	artist_name	total_spent
	text	character varying (120)	double precision
1	Hugh O'Reilly	Queen	27.719999999999985
2	Wyatt Girard	Frank Sinatra	23.75999999999999
3	Aaron Mitchell	James Brown	19.799999999999997
4	Robert Brown	Creedence Clearwater Revi...	19.799999999999997
5	Helena Holý	Red Hot Chili Peppers	19.799999999999997
6	R Madhav	Kiss	19.799999999999997
7	François Tremblay	The Who	19.799999999999997
8	Niklas Schröder	Queen	18.81
9	Hugh O'Reilly	Nirvana	18.81
10	Heather Leacock	House Of Pain	18.81

DETERMINE THE MOST POPULAR MUSIC GENRE IN EACH COUNTRY



```
with popular_genre as
(
select count(il.quantity) as purchase,i.billing_country,g.name,
row_number() over(partition by i.billing_country order by count(il.quantity) desc ) as row_number
from invoice i
join invoice_line il
on i.invoice_id=il.invoice_id
join track t
on il.track_id=t.track_id
join genre g
on t.genre_id=g.genre_id
group by i.billing_country,g.name,il.quantity
)

select purchase,billing_country,name from popular_genre where row_number=1;
```

Output



	purchase bigint	billing_country character varying (30)	name character varying (120)
1	17	Argentina	Alternative & Punk
2	34	Australia	Rock
3	40	Austria	Rock
4	26	Belgium	Rock

CONCLUSION

SUMMARY

- SQL queries were used to analyze employee hierarchy, sales trends, customer behavior, and music popularity
- Concepts such as joins, aggregation, CTEs, and window functions were applied
- Insights derived can support business and marketing decisions

SKILLS DEMONSTRATED

- SQL joins and filtering
- Aggregation and grouping
- CTEs and window functions
- Analytical thinking