MUSIC STORE DATA ANALYSIS

LOADING TABLES:

Due to issues with the Table Data Import Wizard in MySQL Workbench, I had to import each table using SQL code. The following is a sample code that I used for importing the genre table. The same code structure can be used for importing the remaining tables.

```
LOAD DATA INFILE 'C:/ProgramData/MySQL/MySQL Server 8.0/Uploads/genre.csv'
INTO TABLE genre
CHARACTER SET utf8mb4
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\r \n'
IGNORE 1 ROWS;
```

DEFINING PRIMARY KEYS:

```
ALTER TABLE employee
ADD CONSTRAINT pk_employee PRIMARY KEY (employee_id);

ALTER TABLE customer
ADD CONSTRAINT pk_customer PRIMARY KEY (customer_id);

ALTER TABLE invoice
ADD CONSTRAINT pk_invoice PRIMARY KEY (invoice_id);

ALTER TABLE invoice_line
ADD CONSTRAINT pk_invoice_line PRIMARY KEY (invoice_line_id);

ALTER TABLE track
ADD CONSTRAINT pk_track PRIMARY KEY (track_id);

ALTER TABLE media_type
ADD CONSTRAINT pk_media_type PRIMARY KEY (media_type_id);

ALTER TABLE genre
```

```
ADD CONSTRAINT pk_genre PRIMARY KEY (genre_id);

ALTER TABLE playlist
ADD CONSTRAINT pk_playlist PRIMARY KEY (playlist_id);

ALTER TABLE playlist_track
ADD CONSTRAINT pk_playlist_track PRIMARY KEY (playlist_id, track_id);

ALTER TABLE artist
ADD CONSTRAINT pk_artist PRIMARY KEY (artist_id);

ALTER TABLE album
ADD CONSTRAINT pk_album PRIMARY KEY (album_id);
```

DEFINING CONSTRAINTS:

- Define relationships for employee table

```
ALTER TABLE customer
ADD CONSTRAINT fk_support_rep_id FOREIGN KEY (support_rep_id)
REFERENCES employee (employee_id);

ALTER TABLE employee
ADD CONSTRAINT fk_reports_to FOREIGN KEY (reports_to)
REFERENCES employee (employee_id);
```

- Define relationships for invoice table

```
ALTER TABLE invoice
ADD CONSTRAINT fk_customer_id FOREIGN KEY (customer_id)
REFERENCES customer (customer_id);
```

- Define relationships for invoice_line table

```
ALTER TABLE invoice_line
ADD CONSTRAINT fk_invoice_id FOREIGN KEY (invoice_id)
REFERENCES invoice (invoice_id);

ALTER TABLE invoice_line
ADD CONSTRAINT fk_track_id FOREIGN KEY (track_id)
REFERENCES track (track_id);
```

- Define relationships for playlist_track table

```
ALTER TABLE playlist_track
ADD CONSTRAINT fk_playlist_id FOREIGN KEY (playlist_id)
REFERENCES playlist (playlist_id);

ALTER TABLE playlist_track
ADD CONSTRAINT fk1_track_id FOREIGN KEY (track_id)
REFERENCES track (track_id);
```

- Define relationships for track table

```
ALTER TABLE track
ADD CONSTRAINT fk_album_id FOREIGN KEY (album_id)
REFERENCES album (album_id);

ALTER TABLE track
ADD CONSTRAINT fk_media_type_id FOREIGN KEY (media_type_id)
REFERENCES media_type (media_type_id);

ALTER TABLE track
ADD CONSTRAINT fk_genre_id FOREIGN KEY (genre_id)
REFERENCES genre (genre_id);
```

- Define relationships for album table

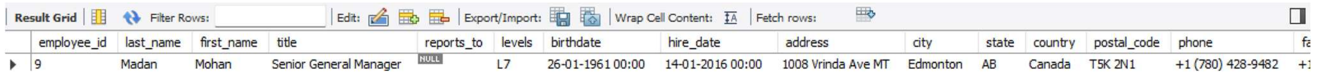
```
ALTER TABLE album
ADD CONSTRAINT fk_artist_id FOREIGN KEY (artist_id)
REFERENCES artist (artist_id);
```

(Question set 1 on the next page)

QUESTION SET 1 (EASY)

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

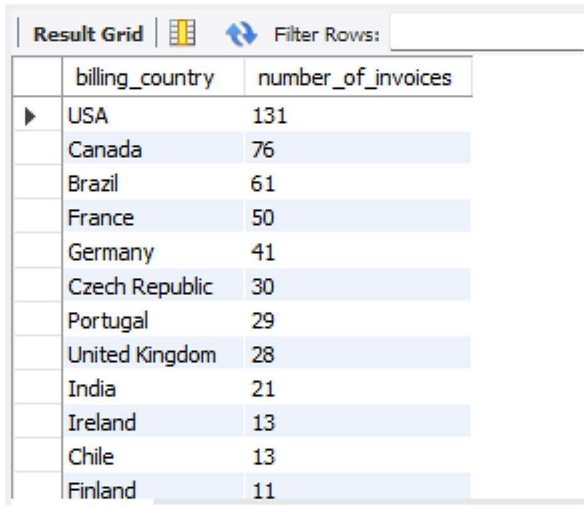
```
SELECT *
FROM employee
ORDER BY levels DESC
LIMIT 1;
```



employee_id	last_name	first_name	title	reports_to	levels	birthdate	hire_date	address	city	state	country	postal_code	phone	fa
9	Madan	Mohan	Senior General Manager	NULL	L7	26-01-1961 00:00	14-01-2016 00:00	1008 Vrinda Ave MT	Edmonton	AB	Canada	T5K 2N1	+1 (780) 428-9482	+1

2. Which countries have the highest number of invoices?

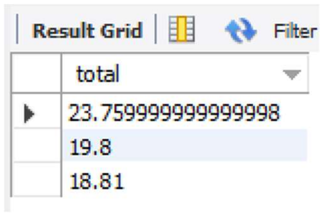
```
SELECT
  billing_country,
  COUNT(billing_country) AS number_of_invoices
FROM
  invoice
GROUP BY
  billing_country
ORDER BY
  number_of_invoices DESC;
```



billing_country	number_of_invoices
USA	131
Canada	76
Brazil	61
France	50
Germany	41
Czech Republic	30
Portugal	29
United Kingdom	28
India	21
Ireland	13
Chile	13
Finland	11

3. What are the top 3 unique values of total invoice?

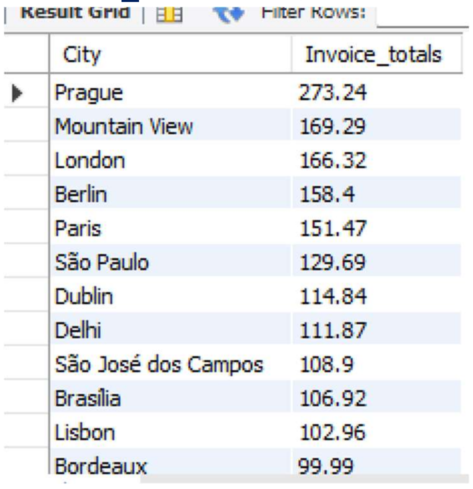
```
SELECT
  DISTINCT total
FROM
  invoice
ORDER BY
  total DESC
LIMIT 3;
```



total
23.759999999999998
19.8
18.81

4. Which city has generated the highest revenue?
   We're planning to host a promotional music festival in the city where we've made the most money. Write a query to return the city name along with the total invoice amount, for the city with the highest invoice total.

```
SELECT
  billing_city AS City,
  ROUND(SUM(total),2) AS Invoice_totals
FROM
  invoice
GROUP BY
  City
ORDER BY
  Invoice_totals DESC;
```

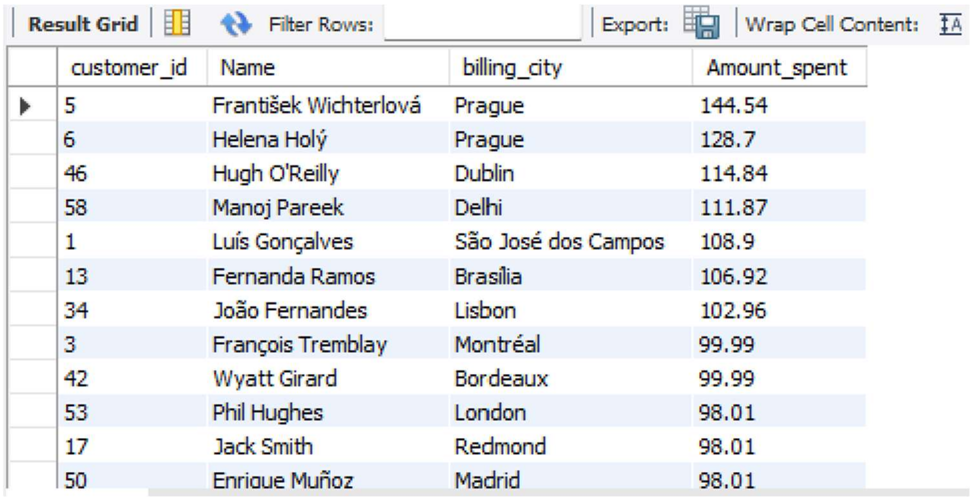


City	Invoice_totals
Prague	273.24
Mountain View	169.29
London	166.32
Berlin	158.4
Paris	151.47
São Paulo	129.69
Dublin	114.84
Delhi	111.87
São José dos Campos	108.9
Brasília	106.92
Lisbon	102.96
Bordeaux	99.99

5. Who is our best customer based on total spend?
   The customer who has spent the most is our best customer.
   Write a query to return the name of the customer along with their total spend.

```
SELECT
   customer.customer_id,
   CONCAT(customer.first_name," ",customer.last_name) AS Name,
   invoice.billing_city,
   ROUND(SUM(invoice.total),2) AS Amount_spent
FROM
   invoice
LEFT JOIN
   customer ON invoice.customer_id = customer.customer_id
GROUP BY
   customer.customer_id,
   Name,
   invoice.billing_city
```

```
ORDER BY
  Amount_spent DESC;
```


Result Grid | Filter Rows: | Export: | Wrap Cell Content:

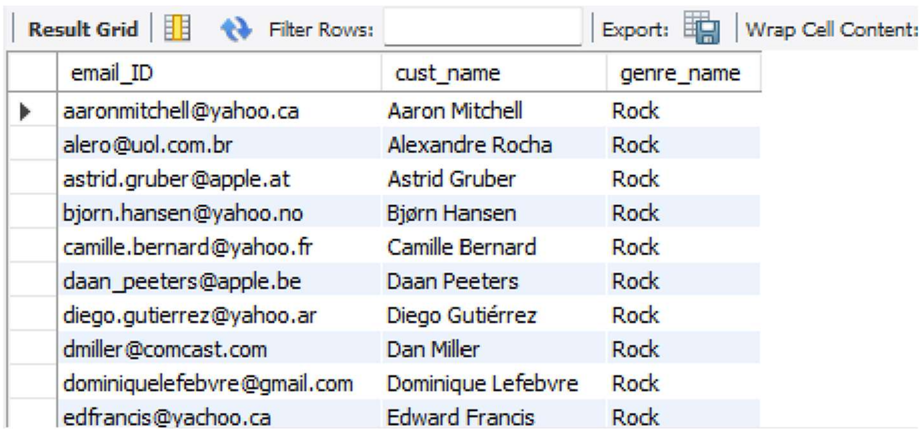
customer_id	Name	billing_city	Amount_spent
5	František Wichterlová	Prague	144.54
6	Helena Holý	Prague	128.7
46	Hugh O'Reilly	Dublin	114.84
58	Manoj Pareek	Delhi	111.87
1	Luís Gonçalves	São José dos Campos	108.9
13	Fernanda Ramos	Brasília	106.92
34	João Fernandes	Lisbon	102.96
3	François Tremblay	Montréal	99.99
42	Wyatt Girard	Bordeaux	99.99
53	Phil Hughes	London	98.01
17	Jack Smith	Redmond	98.01
50	Enrique Muñoz	Madrid	98.01

(Question set 2 on the next page)

QUESTION SET 2 (MODERATE)

1. Write a query to return the email, first name, last name, and genre of all Rock music listeners. Order the results alphabetically by email, starting with A.

```
SELECT
  DISTINCT customer.email AS email_ID,
  CONCAT(customer.first_name," ",customer.last_name) AS cust_name,
  genre.name AS genre_name
FROM
  customer
  LEFT JOIN invoice ON invoice.customer_id=customer.customer_id
  LEFT JOIN invoice_line ON invoice_line.invoice_id=invoice.invoice_id
  LEFT JOIN track ON track.track_id=invoice_line.track_id
  LEFT JOIN genre ON genre.genre_id=track.genre_id
WHERE
  genre.name="Rock"
ORDER BY
  email_ID;
```

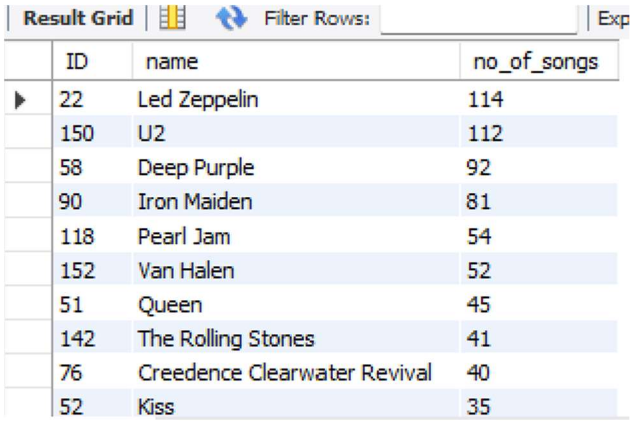


email_ID	cust_name	genre_name
aaronmitchell@yahoo.ca	Aaron Mitchell	Rock
alero@uol.com.br	Alexandre Rocha	Rock
astrid.gruber@apple.at	Astrid Gruber	Rock
bjorn.hansen@yahoo.no	Bjørn Hansen	Rock
camille.bernard@yahoo.fr	Camille Bernard	Rock
daan_peeters@apple.be	Daan Peeters	Rock
diego.gutierrez@yahoo.ar	Diego Gutiérrez	Rock
dmiller@comcast.com	Dan Miller	Rock
dominiquelefebvre@gmail.com	Dominique Lefebvre	Rock
edfrancis@yachoo.ca	Edward Francis	Rock

2. We plan on inviting the artists who have written the most Rock music in our dataset. So, write a query that returns the artist name and the total number of Rock tracks for the top 10 artists.

```
SELECT
  artist.artist_id AS ID,
  artist.name AS name,
  COUNT(album.artist_id) AS no_of_songs
FROM
  artist
  JOIN album ON album.artist_id=artist.artist_id
  JOIN track ON track.album_id=album.album_id
  JOIN genre ON genre.genre_id=track.genre_id
WHERE
  genre.name = “Rock”
GROUP BY
  ID,name
ORDER BY
```

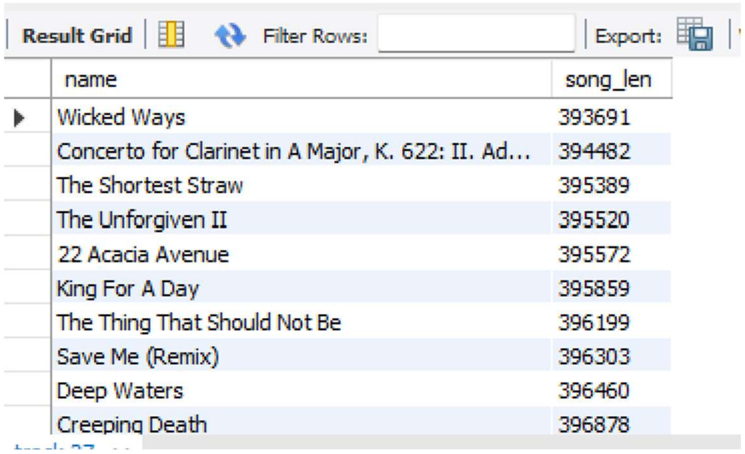
```
  no_of_songs DESC;
```


Result Grid | Filter Rows: | Exp

	ID	name	no_of_songs
▶	22	Led Zeppelin	114
	150	U2	112
	58	Deep Purple	92
	90	Iron Maiden	81
	118	Pearl Jam	54
	152	Van Halen	52
	51	Queen	45
	142	The Rolling Stones	41
	76	Creedence Clearwater Revival	40
	52	Kiss	35

3. Return all track names that are longer than the average song length.
   Write a query to return the track name and duration in milliseconds. Order the results by song length in descending order, with the longest songs listed first.

```
SELECT
  name,
  milliseconds AS song_len
FROM
  track
WHERE milliseconds > (SELECT AVG(milliseconds) as avg_len FROM track)
ORDER BY
  song_len;
```


Result Grid | Filter Rows: | Export:

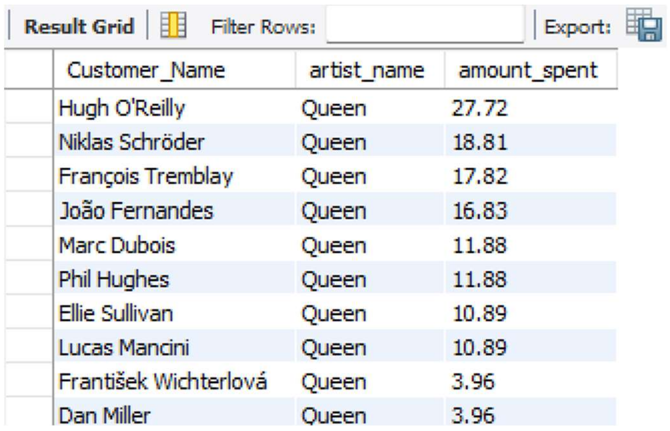
	name	song_len
▶	Wicked Ways	393691
	Concerto for Clarinet in A Major, K. 622: II. Ad...	394482
	The Shortest Straw	395389
	The Unforgiven II	395520
	22 Acacia Avenue	395572
	King For A Day	395859
	The Thing That Should Not Be	396199
	Save Me (Remix)	396303
	Deep Waters	396460
	Creeping Death	396878

(Question set 3 on the next page)

QUESTION SET 3 (ADVANCED)

1. Find the total amount spent by each customer on artists.
   Write a query that returns the customer name, artist name, and the total amount spent by each customer on each artist.

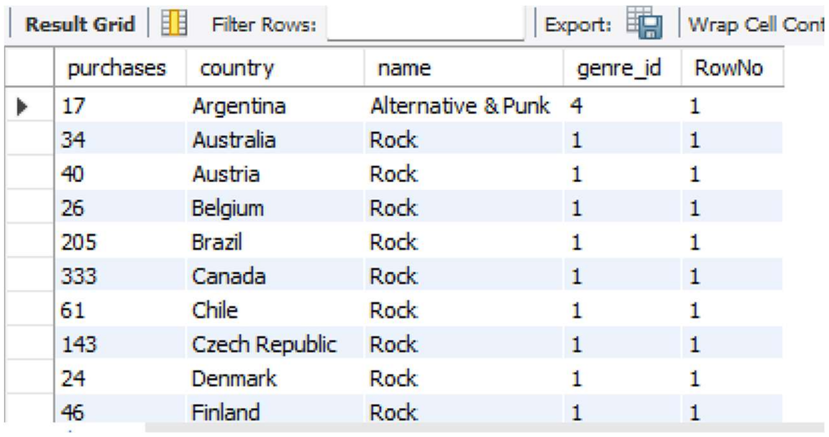
```
WITH highest_sell AS (
 SELECT
   artist.artist_id AS ID,
   artist.name AS artist_name,
   ROUND(SUM(invoice_line.unit_price*invoice_line.quantity),2) AS total_sales
 FROM
   artist
   JOIN album ON album.artist_id=artist.artist_id
   JOIN track ON track.album_id=album.album_id
   JOIN invoice_line ON invoice_line.track_id=track.track_id
 GROUP BY
   ID,artist_name
 ORDER BY
   total_sales DESC
   LIMIT 1;
)
SELECT
  CONCAT(customer.first_name," ",customer.last_name) AS Customer_Name,
  hs.artist_name,
  ROUND(SUM(invoice_line.unit_price*invoice_line.quantity),2) AS amount_spent
FROM
  customer AS customer
  JOIN invoice ON invoice.customer_id=customer.customer_id
  JOIN invoice_line ON invoice_line.invoice_id=invoice.invoice_id
  JOIN track ON track.track_id = invoice_line.track_id
  JOIN album ON album.album_id=track.album_id
  JOIN highest_sell hs ON hs.ID=album.artist_id
GROUP BY
  Customer_Name, hs.artist_name
ORDER BY
  amount_spent DESC;
```



Customer_Name	artist_name	amount_spent
Hugh O'Reilly	Queen	27.72
Niklas Schröder	Queen	18.81
François Tremblay	Queen	17.82
João Fernandes	Queen	16.83
Marc Dubois	Queen	11.88
Phil Hughes	Queen	11.88
Ellie Sullivan	Queen	10.89
Lucas Mancini	Queen	10.89
František Wichterlová	Queen	3.96
Dan Miller	Queen	3.96

2. Identify the most popular music genre for each country.
   The most popular genre is determined by the genre with the highest total purchase amount. Write a query that returns each country along with the top genre. For countries where multiple genres share the highest purchase amount, return all such genres.

```
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;
```

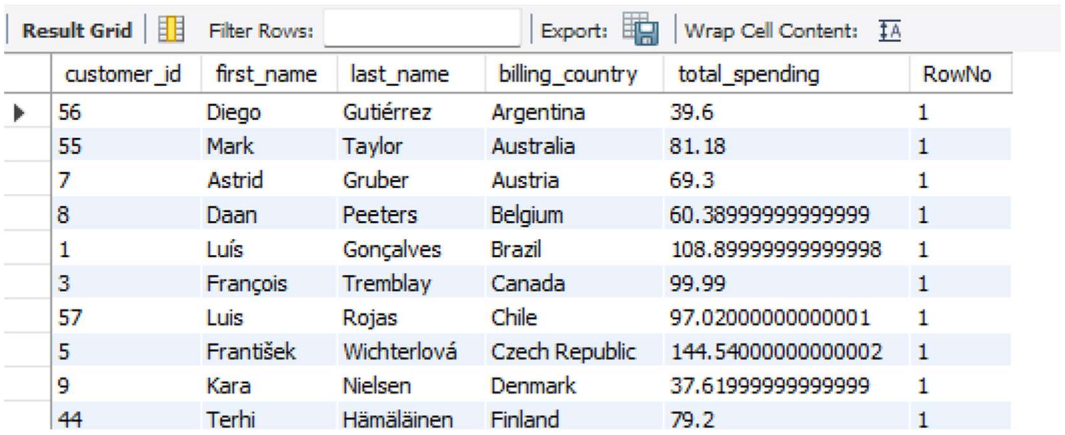

Result Grid | Filter Rows: | Export: | Wrap Cell Cont

purchases	country	name	genre_id	RowNo
17	Argentina	Alternative & Punk	4	1
34	Australia	Rock	1	1
40	Austria	Rock	1	1
26	Belgium	Rock	1	1
205	Brazil	Rock	1	1
333	Canada	Rock	1	1
61	Chile	Rock	1	1
143	Czech Republic	Rock	1	1
24	Denmark	Rock	1	1
46	Finland	Rock	1	1

3. Find the customer who has spent the most on music in each country.
   Write a query that returns the country, top customer, and the amount they spent. For countries where multiple customers share the highest spend, return all such customers.

```
WITH Customter_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 Customter_with_country WHERE RowNo <= 1;
```



Result Grid | Filter Rows: | Export: | Wrap Cell Content:

customer_id	first_name	last_name	billing_country	total_spending	RowNo
56	Diego	Gutiérrez	Argentina	39.6	1
55	Mark	Taylor	Australia	81.18	1
7	Astrid	Gruber	Austria	69.3	1
8	Daan	Peeters	Belgium	60.38999999999999	1
1	Luís	Gonçalves	Brazil	108.89999999999998	1
3	François	Tremblay	Canada	99.99	1
57	Luis	Rojas	Chile	97.02000000000001	1
5	František	Wichterlová	Czech Republic	144.54000000000002	1
9	Kara	Nielsen	Denmark	37.61999999999999	1
44	Terhi	Hämäläinen	Finland	79.2	1

(END)