```
use music_database

select * from album2
select * from artist
select * from customer
select * from employee
select * from genre
select * from invoice
select * from invoice_line
select * from media_type
select * from playlist
select * from playlist_track
select * from track
-----------------------------------------------------------------------------------------------------------
ALTER TABLE customer
ADD CONSTRAINT PK_CustomerID PRIMARY KEY (customer_id);

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

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

ALTER TABLE album2
ADD CONSTRAINT FK_ArtistID FOREIGN KEY (artist_id) REFERENCES artist(artist_id);

ALTER TABLE genre
ADD CONSTRAINT PK_GenreID PRIMARY KEY (genre_id);

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

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

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

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

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

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

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

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

ALTER TABLE playlist_track
```

```
ADD CONSTRAINT FK_Track2tID FOREIGN KEY (track_id) REFERENCES track(track_id);

ALTER TABLE track
ADD CONSTRAINT FK_Track3ID FOREIGN KEY (media_type_id) REFERENCES
media_type(media_type_id);

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

ALTER TABLE track
ADD CONSTRAINT FK_album2ID FOREIGN KEY (album_id) REFERENCES album2(album_id);

ALTER TABLE album2
ADD CONSTRAINT FK_artist2ID FOREIGN KEY (artist_id) REFERENCES artist(artist_id);
```

----> SET-1

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

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

2--> Which countries have the most invoices?

```
SELECT COUNT(*) as c, billing_country
from invoice
group by billing_country
order by c desc
```

3--> What are top 3 values of total invoice?

```
select top 3 total from invoice
order by total desc
```

4--> Which city has the most customers? We would ike 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.

```
SELECT SUM(total) as invoice_total, billing_city
from invoice
group by billing_city
order by invoice_total desc
```

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.

```
ALTER TABLE customer
ADD CONSTRAINT PK_CustomerID PRIMARY KEY (customer_id);

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

```
SELECT TOP 1 customer.customer_id, 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,customer.first_name,customer.last_name
ORDER BY total DESC
```

==========

---> SET 2

1--> Write query to return the email, firstname, lastname, & genre of all 'Rock' Music listners.
--Return you list ordered alphabetically by email starting with A.
--{select * from genre<-->select * from track<-->select * from invoice_line<-->select * from
invoice<-->select * from customer}

```
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 genre.genre_id = track.genre_id
  WHERE genre.name LIKE 'Rock'
)
ORDER BY email;
```

2--> Let's invite the artist 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 TOP 10 artist.artist_id, artist.name, COUNT(artist.artist_id) AS num_of_songs
FROM track
JOIN album2 ON album2.album_id = track.album_id
JOIN artist ON artist.artist_id = album2.artist_id
JOIN genre ON genre.genre_id = track.genre_id
WHERE genre.name LIKE 'Rock'
GROUP BY artist.artist_id, artist.name
ORDER BY num_of_songs
```

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

```
SELECT name, milliseconds
FROM track
WHERE milliseconds > (
    SELECT AVG(milliseconds) AS avg_song_length
  FROM track)
ORDER BY milliseconds DESC;
```

==========

--SET-3

1--> Find how much amount spent by each customer on artists? Writw a query to return customer name, artist name and total spent.

```
WITH best_selling_artists AS(
    SELECT TOP 1 artist.artist_id AS artist_id, artist.name AS artist_name,
  SUM(invoice_line.unit_price*invoice_line.quantity) AS total_sales
  FROM invoice_line
  JOIN track ON track.track_id = invoice_line.track_id
  JOIN album2 ON album2.album_id = track.album_id
  JOIN artist ON artist.artist_id = album2.artist_id
  GROUP BY artist.artist_id, artist.name
  ORDER BY total_sales DESC
)
SELECT c.customer_id, c.first_name, c.last_name, bsa.artist_name,
SUM(il.unit_price*il.quantity) AS amount_spent
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 album2 alb ON alb.album_id = t.album_id
JOIN best_selling_artists bsa ON bsa.artist_id = alb.artist_id
GROUP BY c.customer_id, c.first_name, c.last_name, bsa.artist_name
ORDER BY amount_spent DESC;

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.
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 customer.country, genre.name, genre.genre_id
--  ORDER BY customer.country ASC, purchases DESC
)
SELECT * FROM popular_genre WHERE RowNo <= 1
---------------------------------------------------------------------------------------------

 WITH sales_per_country AS(
     SELECT COUNT(*) AS purchase_per_genre, customer.country, genre.name, genre.genre_id
   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 customer.country, genre.name, genre.genre_id
   --ORDER BY customer.country
),
max_genre_per_country AS (SELECT MAX(purchase_per_genre) AS max_genre_number, country
    FROM sales_per_country
```

```
 GROUP BY country)
 --ORDER BY country)

SELECT sales_per_country.*
FROM sales_per_country
JOIN max_genre_per_country ON sales_per_country.country = max_genre_per_country.country
WHERE sales_per_country.purchase_per_genre = max_genre_per_country.max_genre_number
```

==========

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 his amount.

```
WITH customer_with_country AS (
    SELECT customer.customer_id, first_name, last_name, billing_country, SUM(total) AS total_spending
  FROM invoice
  JOIN customer ON customer.customer_id = invoice.customer_id
  GROUP BY customer.customer_id, first_name, last_name, billing_country),

  country_max_spending AS(
    SELECT billing_country, MAX(total_spending) AS max_spending
  FROM customer_with_country
  GROUP BY billing_country)

SELECT cc.billing_country, cc.total_spending, cc.first_name, cc.last_name, cc.customer_id
FROM customer_with_country cc
JOIN country_max_spending ms
ON cc.billing_country = ms.billing_country
WHERE cc.total_spending = ms.max_spending
ORDER BY cc.billing_country
```

```
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 customer.customer_id, first_name, last_name, billing_country
)
SELECT * FROM Customer_with_country WHERE RowNo <= 1
```