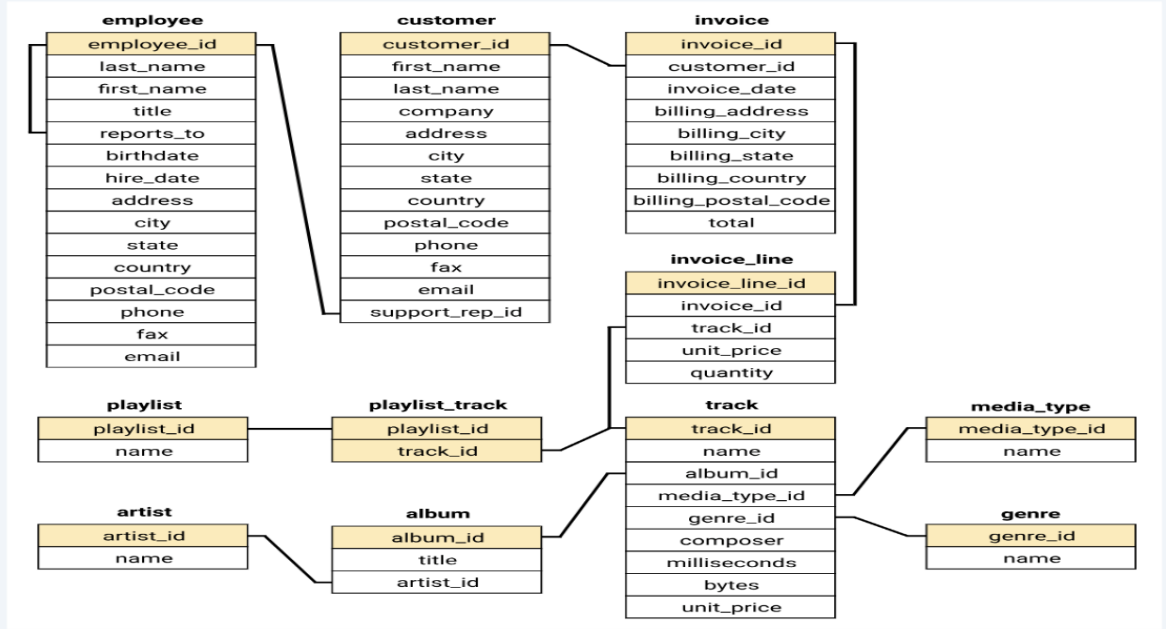
SQL PROJECT- MUSIC STORE DATA ANALYSIS

:Schema Diagram:



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

2.Which countries have the most Invoices?

3.What are top 3 values of total invoice?

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

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

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

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

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.

Create Table using SQL Query

```
ALTER TABLE assets
ALTER COLUMN asset_no TYPE INT
USING asset_no::INT;


CREATE TABLE employee(
employee_id VARCHAR(50) PRIMARY KEY,
last_name CHAR(50),
first_name CHAR(50),
title VARCHAR(50),
reports_to VARCHAR(30),
levels VARCHAR(10),
birthdate TIMESTAMP,
hire_date TIMESTAMP,
address VARCHAR(120),
city VARCHAR(50),
state VARCHAR(50),
country VARCHAR(30),
postal_code VARCHAR(30),
phone VARCHAR(30),
fax VARCHAR(30),
email VARCHAR(30));


CREATE TABLE customer(
customer_id VARCHAR(30) PRIMARY KEY,
first_name CHAR(30),
last_name CHAR(30),
company VARCHAR(30),
```

address VARCHAR(30),

city VARCHAR(30),

state VARCHAR(30),

country VARCHAR(30),

postal_code INT8,

phone INT,

fax INT,

email VARCHAR(30),

support_rep_id VARCHAR(30));

CREATE TABLE invoice(

invoice_id VARCHAR(30) PRIMARY KEY,

customer_id VARCHAR(30),

invoice_date TIMESTAMP,

billing_address VARCHAR(120),

billing_city VARCHAR(30),

billing_state VARCHAR(30),

billing_country VARCHAR(30),

billing_postal VARCHAR(30),

total FLOAT8);

CREATE TABLE invoice_line(

invoice_line_id VARCHAR(50) PRIMARY KEY,

invoice_id VARCHAR(30),

track_id VARCHAR(30),

unit_price VARCHAR(30),

quantity VARCHAR(30));

```
CREATE TABLE track(
track_id VARCHAR(50) PRIMARY KEY,
name VARCHAR(30),
album_id VARCHAR(30),
media_type_id VARCHAR(30),
genre_id VARCHAR(30),
composer VARCHAR(30),
milliseconds TIMESTAMP,
bytes INT8,
unit_price INT16);

CREATE TABLE playlist(
playlist_id VARCHAR(50) PRIMARY KEY,
name  VARCHAR(30));

CREATE TABLE playlist_track(
playlist_id VARCHAR(50) PRIMARY KEY,
track_id VARCHAR(50) PRIMARY KEY);

CREATE TABLE artist(
artist_id VARCHAR(50) PRIMARY KEY,
name  VARCHAR(30));

CREATE TABLE album(
album_id VARCHAR(50) PRIMARY KEY,
title  VARCHAR(30),
artist_id  VARCHAR(30));
```

```
CREATE TABLE media_type(
media_type_id VARCHAR(50) PRIMARY KEY,
name VARCHAR(30));

CREATE TABLE genre(
genre_id VARCHAR(50) PRIMARY KEY,
name VARCHAR(30));
```

:Questions Solved Using SQL Query:

```
/*  Question Set 1  */

/* Q1: Who is the senior most employee based on job title? */
SELECT title, last_name, first_name
FROM employee
ORDER BY levels DESC
LIMIT 1

/* Q2: Which countries have the most Invoices? */
SELECT COUNT(*) AS c, billing_country
FROM invoice
GROUP BY billing_country
ORDER BY c DESC

/* Q3: What are top 3 values of total invoice? */
SELECT total
FROM invoice
ORDER BY total DESC
```

/* Q4: 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 */

```
SELECT billing_city,SUM(total) AS InvoiceTotal

FROM invoice

GROUP BY billing_city

ORDER BY InvoiceTotal DESC

LIMIT 1;
```

/* Q5: 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.*/

```
SELECT customer.customer_id, first_name, last_name, SUM(total) AS total_spending

FROM customer

JOIN invoice ON customer.customer_id = invoice.customer_id

GROUP BY customer.customer_id

ORDER BY total_spending DESC

LIMIT 1;
```

/* Question Set 2 */

/* Q1: 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. */

/*Method 1 */

```
SELECT DISTINCT email,first_name, last_name

FROM customer

JOIN invoice ON customer.customer_id = invoice.customer_id

JOIN invoiceline ON invoice.invoice_id = invoiceline.invoice_id

WHERE track_id IN(

   SELECT track_id FROM track

   JOIN genre ON track.genre_id = genre.genre_id

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

/* Method 2 */

```
SELECT DISTINCT email AS Email,first_name AS FirstName, last_name AS LastName, genre.name AS Name
FROM customer
JOIN invoice ON invoice.customer_id = customer.customer_id
JOIN invoiceline ON invoiceline.invoice_id = invoice.invoice_id
JOIN track ON track.track_id = invoiceline.track_id
JOIN genre ON genre.genre_id = track.genre_id
WHERE genre.name LIKE 'Rock'
ORDER BY email;
```

/* Q2: 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. */

```
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;
```

/* Q3: 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. */

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