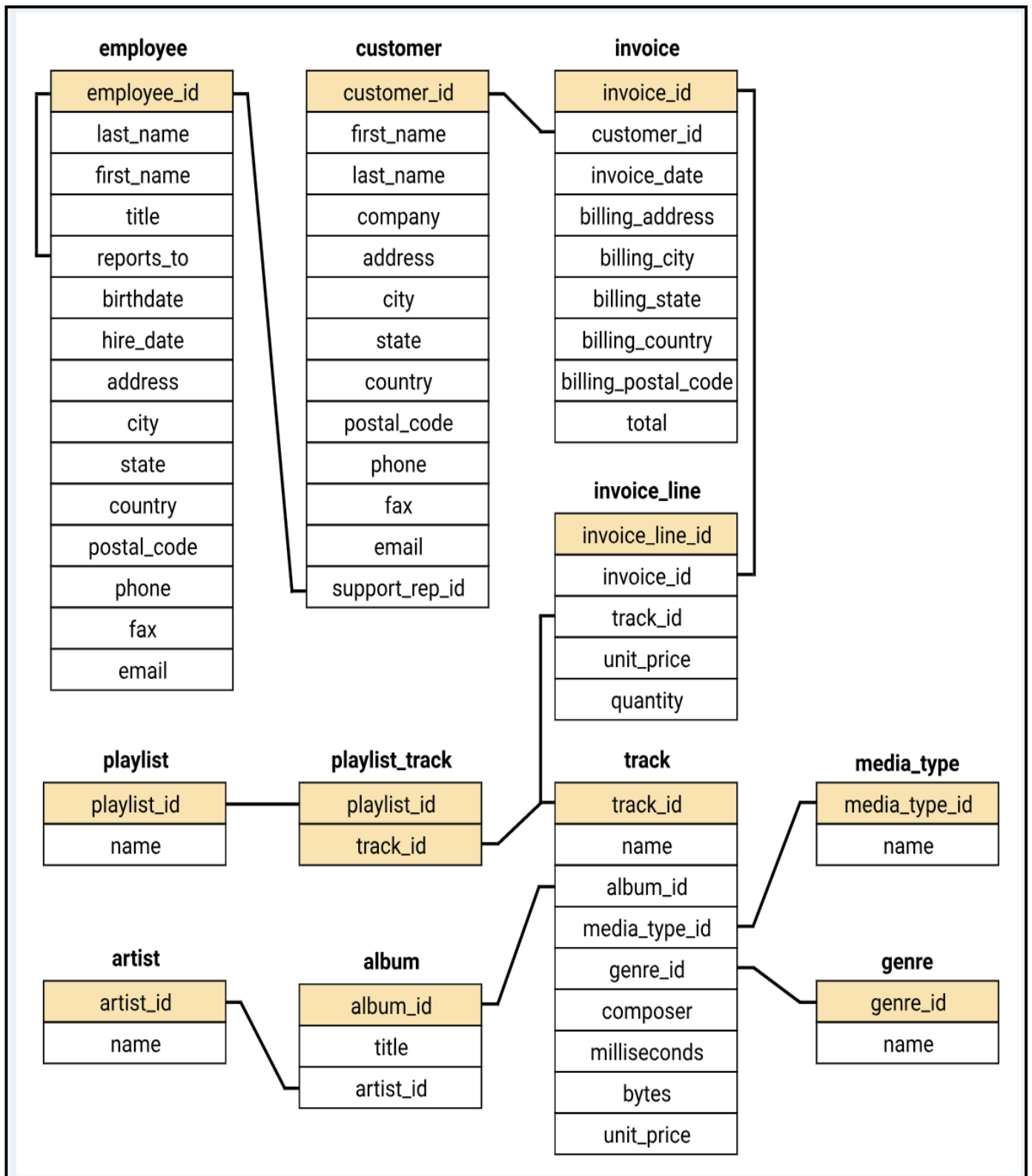


SQL PROJECT- MUSIC STORE DATA ANALYSIS

Schema Diagram









SQL PROJECT- MUSIC STORE DATA ANALYSIS

Query Query History

```
1  -- MUSIC STORE DATA ANALYSIS
2  -- SET 1 Easy Questions --
3
4  -- 1) Who is the senior most employee based on job title?
5  v select * from employee  -- first lets view the employee table
6  order by levels desc
7  limit 1
8
9
```


Data Output Messages Notifications





| |  last_name character |  first_name character |  title character varying (50) |  reports_to character varying (30) |  levels character varying (10) |
|---|----------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------|
| 1 | Madan | ... | Mohan | ... | Senior General Manager [null] L7 |

```
8
9  -- 2) Which countries have the most Invoices?
10 v select COUNT (*) as billing_count, billing_country
11 from invoice
12 group by billing_country
13 order by billing_count desc;
14
15
16
17
```

Data Output Messages Notifications



| | billing_count bigint  | billing_country character varying (30)  |
|---|-------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------|
| 1 | 131 | USA |
| 2 | 76 | Canada |
| 3 | 61 | Brazil |
| 4 | 50 | France |
| 5 | 41 | Germany |
| 6 | 30 | Czech Republic |
| 7 | 29 | Portugal |
| 8 | 28 | United Kingdom |
| 9 | 21 | India |

Total rows: 24 of 24 Query complete 00:00:00.270

Query

Query History

14

15

16

17

18

19

20

21

22

23

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

select total from invoice

order by total desc

limit 3;

Data Output

Messages

Notifications

+

📄

▼

📋

▼

🗑️

🔍

⬇️

📈

| | total |
|---|--------------------|
| | double precision |
| 1 | 23.759999999999998 |
| 2 | 19.8 |
| 3 | 19.8 |

4

5

6

7

8

9

-- 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

select SUM(total) as invoice_total,billing_city

from invoice

group by billing_city

order by invoice_total desc

limit 3;

-- hence, Prague has the maximum no. of customers purchasing the maximum no.of albums.

Data Output

Messages

Notifications

+

📄

▼

📋

▼

🗑️

🔍

⬇️

📈

| invoice_total | billing_city |
|---------------------|------------------------|
| double precision | character varying (30) |
| 273.240000000000007 | Prague |
| 169.29 | Mountain View |
| 166.32 | London |

y

Query History

--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

Select 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

ORDER BY total desc

Limit 1

Output

Messages

Notifications

📄

▼

📋

▼

🗑️

🔍

⬇️

📈

| customer_id | first_name | last_name | total |
|--------------|------------|-----------|---------------------|
| [PK] integer | character | character | double precision |
| 5 | R | Madhav | 144.540000000000002 |

-- SET 2 MODERATE QUESTIONS

```
--1) 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
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 track.genre_id = genre.genre_id
    WHERE genre.name LIKE 'Rock'
)
ORDER BY email
```

Data Output Messages Notifications

| email | first_name | last_name |
|--------------------------|------------|-----------|
| character varying (50) | character | character |
| aaronmitchell@yahoo.ca | Aaron | Mitchell |
| alero@uol.com.br | Alexandre | Rocha |
| astrid.gruber@apple.at | Astrid | Gruber |
| bjorn.hansen@yahoo.no | Bjørn | Hansen |
| camille.bernard@yahoo.fr | Camille | Bernard |

Total rows: 59 of 59 Query complete 00:00:00.217

Query Query History

```
--2) 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
```

Data Output Messages Notifications

| artist_id | name | number_of_songs |
|-----------------------------|-------------------------|-----------------|
| [PK] character varying (50) | character varying (120) | bigint |
| 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 |

Total rows: 10 of 10 Query complete 00:00:00.162

Query Query History

```
-- 3) 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, milliseconds from track
WHERE milliseconds >
(
    SELECT AVG(milliseconds) as avg_track_length
    From track
)
ORDER BY milliseconds desc;
```

Data Output Messages Notifications

| name | milliseconds |
|-----------------------------|--------------|
| character varying (150) | integer |
| Occupation / Precipice | 5286953 |
| Through a Looking Glass | 5088838 |
| Greetings from Earth, Pt. 1 | 2960293 |
| The Man With Nine Lives | 2956998 |
| Battlestar Galactica, Pt. 2 | 2956081 |
| Battlestar Galactica, Pt. 1 | 2952702 |
| Murder On the Rising Star | 2935894 |

Total rows: 494 of 494 Query complete 00:00:00.174

Query History

```

-- SET 3 ADVANCE QUESTIONS
-- 1) Find how much amount spent by each customer on artists? Write a query to return customer
--name, artist name and total spent.
WITH best_selling_artist AS (
    SELECT 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 album ON album.album_id = track.album_id
    JOIN artist ON artist.artist_id = album.artist_id
    GROUP BY 1
    ORDER BY 3 desc
    LIMIT 1
)
SELECT customer.customer_id, customer.first_name, customer.last_name, best_selling_artist.artist_
SUM(invoice_line.unit_price*invoice_line.quantity) AS amount_spent
FROM invoice
JOIN customer ON customer.customer_id = invoice.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 best_selling_artist ON best_selling_artist.artist_id = album.artist_id
GROUP BY customer.customer_id, customer.first_name, customer.last_name, best_selling_artist.artist_n
ORDER BY amount_spent DESC;

```

Activate

/*SET 3 ADVANCE QUESTIONS*/

```

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

```

Output Messages Notifications

| customer_id integer | first_name character | last_name character | artist_name character varying (120) | amount_spent double precision |
|------------------------|-------------------------|------------------------|----------------------------------------|----------------------------------|
| 46 | Hugh | O'Reilly | Queen | 27.719999999999985 |
| 38 | Niklas | Schröder | Queen | 18.81 |
| 3 | François | Tremblay | Queen | 17.82 |
| 34 | João | Fernandes | Queen | 16.830000000000002 |
| 53 | Phil | Hughes | Queen | 11.88 |
| 41 | Marc | Dubois | Queen | 11.88 |
| 47 | Lucas | Mancini | Queen | 10.89 |
| 33 | Ellie | Sullivan | Queen | 10.89 |
| 20 | Dan | Miller | Queen | 3.96 |
| 5 | R | Madhav | Queen | 3.96 |

Query Query History

```

110 --2) We want to find out the most popular music Genre for each country. We determine the most
111 --popular genre as the genre with the highest amount of purchases. Write a query that returns
112 --each country along with the top Genre. For countries where the maximum number of purchases is
113 --shared return all Genres
114
115 WITH popular_genre AS
116 (
117     SELECT COUNT (invoice_line.quantity) AS purchases, customer.country, genre.name, genre.genre_id,
118     ROW_NUMBER() OVER (PARTITION BY customer.country ORDER BY COUNT(invoice_line.quantity)DESC)
119     AS Row_No
120     FROM invoice_line
121     JOIN invoice on invoice.invoice_id = invoice_line.invoice_id
122     JOIN customer on customer.customer_id = invoice.customer_id
123     JOIN track on track.track_id = invoice_line.track_id
124     JOIN genre on genre.genre_id = track.genre_id
125     GROUP BY 2,3,4
126     ORDER BY 2 ASC, 1 DESC
127 )
128 SELECT * from popular_genre WHERE Row_No<=1;

```

Query History

```

10 --2) We want to find out the most popular music Genre for each country. We determine the most
11 --popular genre as the genre with the highest amount of purchases. Write a query that returns
12 --each country along with the top Genre. For countries where the maximum number of purchases is
13 --shared return all Genres
14
15 WITH popular_genre AS
16 (
17     SELECT COUNT (invoice_line.quantity) AS purchases, customer.country, genre.name, genre.genre_id

```

Data Output Messages Notifications

| purchases bigint | country character varying (50) | name character varying (120) | genre_id character varying (50) | row_no bigint |
|---------------------|-----------------------------------|---------------------------------|------------------------------------|------------------|
| 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 |

Total rows: 24 of 24 Query complete 00:00:00.153

Query History

```

153 --3) Write a query that determines the customer that has spent the most on music for each country.
154 --Write a query that returns the country along with the top customer and how much they spent.
155 --For countries where the top amount spent is shared, provide all customers who spent this amount
156
157 WITH Customer_with_country AS (
158     SELECT customer.customer_id,first_name,last_name,billing_country, SUM(total) AS amount_spent,
159     ROW_NUMBER() OVER (PARTITION BY billing_country ORDER BY SUM(total) DESC) AS Row_no
160     FROM invoice
161     JOIN customer on customer.customer_id = invoice.customer_id
162     GROUP BY 1,2,3,4
163     ORDER BY 4 ASC, 5 DESC
164 )
165 SELECT * FROM Customer_with_country WHERE Row_no<=1

```

Data Output Messages Notifications

| customer_id integer | first_name character | last_name character | billing_country character varying (30) | amount_spent double precision | row_no bigint |
|------------------------|-------------------------|------------------------|-------------------------------------------|----------------------------------|------------------|
| 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 | 60.389999999999999 | 1 |
| 1 | Luís | Gonçalves | Brazil | 108.899999999999998 | 1 |
| 3 | François | Tramblay | Canada | 00.00 | 1 |

Total rows: 24 of 24 Query complete 00:00:00.432