

SQL Techniques for Music Store Data Analysis

Overview of SQL for Music Data Analysis

Introduction to SQL

SQL (Structured Query Language) is a powerful tool for querying and analyzing databases. It provides a standardized way to interact with data stored in relational databases.

Importance of SQL in data analysis

SQL plays a crucial role in data analysis by allowing users to extract and manipulate relevant information from databases efficiently. Its versatility and simplicity make it a preferred choice for data professionals.

Overview of music store data analysis

Music store data analysis involves extracting insights from data related to customer purchases, inventory management, sales trends, and more. SQL techniques facilitate querying and interpreting this data effectively.

Data Retrieval Techniques in SQL

Basic SELECT statements

The SELECT statement is fundamental in SQL for retrieving data from a database table. It allows users to specify which columns to retrieve and which table to query.

Filtering data with WHERE clause

The WHERE clause enables users to filter data based on specific conditions in SQL queries. It helps narrow down the results to meet the criteria specified by the user.

Sorting data using ORDER BY clause

The ORDER BY clause is used to sort query results in ascending or descending order based on specified columns. It provides control over the presentation of data retrieved from the database.

Aggregation and Grouping in SQL

Using aggregate functions (SUM, COUNT, AVG)

Aggregate functions such as SUM, COUNT, and AVG are essential for performing calculations on sets of data. They help summarize information and provide valuable insights during data analysis.

Grouping data with GROUP BY clause

The GROUP BY clause is used to group rows sharing a common value into summary rows. It allows for the categorization of data based on specified columns, enabling further analysis on grouped data.

Filtering grouped data with HAVING clause

The HAVING clause is employed with GROUP BY to filter grouped data based on specified conditions. It allows users to apply filtering criteria to aggregated data, refining the results further.

Advanced SQL Techniques for Data Analysis

Subqueries for complex queries

Subqueries are nested queries within a main query and are useful for executing complex queries. They enable users to break down complex problems into smaller, manageable parts.

Joins for combining data from multiple tables

Joins are used to combine related data from different tables based on common columns. They facilitate the retrieval of data from multiple sources to perform comprehensive analysis.

Using window functions for advanced analytics

Window functions provide a way to perform calculations across a set of rows related to the current row. This advanced SQL technique allows for tasks like ranking, moving averages, and cumulative sums during data analysis.

The project is divided into three categories of problems - **easy, intermediate, and advanced** - each addressing different aspects of the data.

Easy Problems and Solutions:

Q)Who is the senior most employee based on job title?

```
SELECT*FROM employee
```

```
ORDER BY levels DESC
```

```
LIMIT 1
```

	employee_id [PK] character varying (50)	last_name character	first_name character	title character varying (50)	reports_to character varying (30)	levels character varying (
1	9	Madan	Mohan	Senior General Manager	[null]	L7

Result: This query retrieves the senior-most employee by ordering the employees based on their levels in descending order and selecting the top entry.

Q. Which countries have the most Invoices?

```
SELECT COUNT(*) AS c, billing_country
```

```
FROM invoice
```

```
GROUP BY billing_country
```

```
ORDER BY c DESC
```


	c 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

Q. What are top 3 values of total invoice?

```
SELECT total FROM invoice
```

```
ORDER BY total DESC
```

LIMIT 3

total	
double precision	
23.759999999999998	
	19.8
	19.8

Q. 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 total_invoice ,billing_city
```

```
from invoice
```

```
GROUP BY billing_city
```

```
ORDER BY total_invoice DESC
```

total_invoice double precision	billing_city character varying (30)
273.24000000000007	Prague
169.29	Mountain View
166.32	London
158.4	Berlin
151.47	Paris
129.69	São Paulo

*** Here **Prague** city is the best customer.

Q. 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;
```

customer_id [PK] integer	first_name character	last_name character	total double precision
5	R	Madhav	144.54000000000002

**** R Madhav is the best customer.

Q. 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 ASC

```

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
daan_peeters@apple.be	Daan	Peeters
diego.gutierrez@yahoo.ar	Diego	Gutiérrez
dwiller@comcast.com	Don	Miller

Q.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 no_of_song
FROM track
JOIN album ON track.album_id=album.album_id
JOIN artist ON artist.artist_id=album.artist_id
JOIN genre ON genre.genre_id=track.genre_id
WHERE genre.name='Rock'
GROUP BY artist.artist_id
ORDER BY no_of_song DESC
LIMIT 10

```


artist_id [PK] character varying (50)	name character varying (120)	no_of_song 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

Q. 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

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

Q.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 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 album alb ON alb.album_id=t.album_id

JOIN best_selling_artist bsa ON bsa.artist_id=alb.artist_id

GROUP BY 1,2,3,4

ORDER BY 5 DESC

)

customer_id	first_name	last_name	artist_name	amount_spent
integer	character	character	character varying (120)	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
22	Ellie	Sullivan	Queen	10.89

Q) 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 purchase, 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

purchase bigint	country character varying (50)	name character varying (120)	genre_id character varying (50)	rowno 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

Q) 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 this amount.

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 1,2,3,4

ORDER BY 4 ASC,5 DESC)

SELECT* FROM customer_with_country WHERE RowNo<=1

Data Output Messages Notifications							
	customer_id integer	first_name character	last_name character	billing_country character varying (30)	total_spending double precision	rowno bigint	
1	56	Diego	Gutiérrez	Argentina	39.6	1	
2	55	Mark	Taylor	Australia	81.18	1	
3	7	Astrid	Gruber	Austria	69.3	1	
4	8	Daan	Peeters	Belgium	60.389999999999999	1	
5	1	Luis	Gonçalves	Brazil	108.899999999999998	1	
6	3	François	Tremblay	Canada	99.99	1	
7	57	Luis	Rojas	Chile	97.020000000000001	1	
8	5	R	Madhav	Czech Republic	144.540000000000002	1	