Music Store Data Analysis — Questions & SQL Answers

David Otewa

September 2025

About

Beginner-friendly SQL practice on an online music store ("Chinook"-style) schema. Tables commonly used: `artist`, `album`, `track`, `genre`, `media_type`, `customer`, `employee`, `invoice`, `invoice_line`.

1 Question Set 1 — Easy

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

Assume senior-most ⇒ has no manager.

```sql
SELECT firstname, lastname, title
FROM employee
WHERE reportsto IS NULL;
```

2. Which countries have the most invoices?

```sql
SELECT billingcountry, COUNT(*) AS invoice_count
FROM invoice
GROUP BY billingcountry
ORDER BY invoice_count DESC;
-- Top one only: add LIMIT 1
```

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

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

4. Which city has the best customers (highest total revenue)?

Return city and sum of invoice totals.

```sql
SELECT billingcity, SUM(total) AS total_revenue
FROM invoice
GROUP BY billingcity
ORDER BY total_revenue DESC
LIMIT 1;
```

5. Who is the best customer (highest spend)?

```sql
SELECT
  c.customerid,
  c.firstname || ' ' || c.lastname AS customer,
  SUM(i.total) AS amount_spent
FROM customer c
JOIN invoice i ON i.customerid = c.customerid
GROUP BY c.customerid, customer
ORDER BY amount_spent DESC
LIMIT 1;
```

2 Question Set 2 — Moderate

1. Emails, first/last names, and Genre of all Rock listeners (ordered by email)

```sql
SELECT DISTINCT
  c.email,
  c.firstname,
  c.lastname,
  g.name AS genre
FROM customer c
JOIN invoice i ON i.customerid = c.customerid
JOIN invoice_line il ON il.invoiceid = i.invoiceid
JOIN track t ON t.trackid = il.trackid
JOIN genre g ON g.genreid = t.genreid
WHERE g.name = 'Rock'
ORDER BY c.email ASC;
```

2. Top 10 artists by number of Rock tracks

```sql
SELECT ar.name AS artist, COUNT(*) AS track_count
FROM artist ar
JOIN album al ON al.artistid = ar.artistid
JOIN track t ON t.albumid = al.albumid
JOIN genre g ON g.genreid = t.genreid
WHERE g.name = 'Rock'
GROUP BY ar.artistid, ar.name
ORDER BY track_count DESC
LIMIT 10;
```

3. Tracks longer than the average song length

Return name and milliseconds, longest first.

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

3 Question Set 3 — Advance

1. Amount each customer spent on each artist

```
SELECT
  (c.firstname || ' ' || c.lastname) AS customer,
  ar.name AS artist,
  SUM(il.unitprice * il.quantity) AS total_spent
FROM invoice_line il
JOIN invoice i ON i.invoiceid = il.invoiceid
JOIN customer c ON c.customerid = i.customerid
JOIN track t ON t.trackid = il.trackid
JOIN album al ON al.albumid = t.albumid
JOIN artist ar ON ar.artistid = al.artistid
GROUP BY customer, ar.name
ORDER BY total_spent DESC;
```

2. Most popular genre per country (by purchases; ties included)

```
WITH genre_counts AS (
  SELECT
    i.billingcountry AS country,
    g.name AS genre,
    COUNT(*) AS purchases
  FROM invoice_line il
  JOIN invoice i ON i.invoiceid = il.invoiceid
  JOIN track t ON t.trackid = il.trackid
  JOIN genre g ON g.genreid = t.genreid
  GROUP BY i.billingcountry, g.name
),
ranked AS (
  SELECT *,
         DENSE_RANK() OVER (PARTITION BY country ORDER BY purchases DESC) AS rnk
  FROM genre_counts
)
SELECT country, genre, purchases
FROM ranked
WHERE rnk = 1
ORDER BY country, genre;
```

3. Top-spending customer per country (ties included)

```
WITH customer_spend AS (
  SELECT
    i.billingcountry AS country,
    c.customerid,
    (c.firstname || ' ' || c.lastname) AS customer,
    SUM(i.total) AS total_spent
  FROM invoice i
  JOIN customer c ON c.customerid = i.customerid
  GROUP BY i.billingcountry, c.customerid, customer
),
ranked AS (
  SELECT *,
         DENSE_RANK() OVER (PARTITION BY country ORDER BY total_spent DESC) AS rnk
```

```
  FROM customer_spend
)
SELECT country, customer, total_spent
FROM ranked
WHERE rnk = 1
ORDER BY country, customer;
```

Notes: Queries are written for PostgreSQL. If your dates are in `DD-MM-YYYY`, consider running `SET datestyle TO 'ISO, DMY';` for parsing during loads/queries.