Question Set 1 – Easy

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

```
>> SELECT * FROM employee
        ORDER BY levels
        LIMIT 1;
```

2. Which countries have the most Invoices?

```
>> SELECT billing_country, COUNT(*) AS invoice_count FROM invoice
        GROUP BY billing_country
        ORDER BY invoice_count DESC
        LIMIT 1;
```

3. Which city has the best customers? We would like to throw a promotional Music Festival in the city where we made the most money. Write a query that returns the one city that has the highest sum of invoice totals. Return both the city name and the sum of all invoice totals.

```
>> SELECT billing_city, ROUND(SUM(total)) AS total_revenue
        FROM invoice
        GROUP BY billing_city
        ORDER BY total_revenue DESC
        LIMIT 1;
```

4. 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 c.customer_id, CONCAT(c.first_name, " ", c.last_name) AS Full_Name,
c.Country , ROUND(SUM(i.total),2)  AS total_spent FROM customer AS c

        JOIN invoice AS i ON c.customer_id = i.customer_id
        GROUP BY c.customer_id
        ORDER BY total_spent DESC
        LIMIT 1;
```

5. Write a query to return the number of tracks in each genre.

```
>> SELECT g.genre_id, g.Name, COUNT(t.track_id) AS track_count FROM genre g
        LEFT JOIN track t ON g.genre_id = t.genre_id
        GROUP BY g.genre_id, g.Name;
```

6. Write a query to find all employees who do not report to anyone.

```
>> SELECT CONCAT(first_name, " ", last_name) AS Name, Title AS Position FROM employee
        WHERE reports_to
        IS NULL;
```

7. Write a query to find the names of all albums released by a particular artist, e.g., 'Queen'.

```
>> SELECT al.title AS Album_name, ar.name AS Artist FROM album al
        JOIN artist ar
        ON al.artist_id = ar.artist_id
        WHERE ar.name = 'Queen';
```

8. Write a query to find the first and last names of all employees who are IT staff.

```
>> SELECT CONCAT(first_name, ' ', last_name) AS Emp_Name, title AS Position FROM employee
         WHERE title like 'IT%';
```

9. Write a query to find the total number of albums for each artist.

```
>> SELECT ar.Name, COUNT(al.artist_id) AS Album_count FROM artist ar
         LEFT JOIN album al
```

```
ON ar.artist_id = al.artist_id
GROUP BY ar.artist_id, ar.name;
```

10. Write a query to list the names, position and birthday date of all employees who have a birthday in the current month.

```
>> SELECT CONCAT(first_name, ' ', last_name) AS Emp_Name, title AS Position, birthdate AS Born_Date FROM employee
        WHERE month(birthdate) = month(now());
```

QuestiON Set 2 – Moderate

1. **Write a query to find the number of invoices issued in each month of a specific year, e.g., 2020.**

```
>> SELECT monthname(invoice_date) AS month, COUNT(*) AS Invoice_count FROM invoice
        WHERE year(invoice_date) = 2020
        GROUP BY month;
```

2. **Write a query to list the names and total purchase amounts of customers who have spent more than $100.**

```
>> SELECT c.customer_id, CONCAT(c.first_name, " ", c.last_name) AS Full_Name , ROUND(SUM(i.total)) AS total_spent FROM
customer AS c
        JOIN invoice AS i ON c.customer_id = i.customer_id
        GROUP BY c.customer_id
        having total_spent >= 100
        ORDER BY total_spent DESC;
```

3. **Find the total revenue generated from each media type.**

```
>> SELECT m.name AS media_type, CONCAT("$", " ", SUM(il.quantity * il.unit_price)) AS total_revenue FROM invoice_line il
        INNER JOIN track t ON il.track_id = t.track_id
        INNER JOIN media_type m ON t.media_type_id = m.media_type_id
        GROUP BY m.media_type_id;
```

4. **Identify the top 5 best-selling albums by revenue.**

```
>> SELECT al.title AS Album_title, SUM(il.quantity * il.unit_price) AS total_revenue FROM album al
        INNER JOIN track t ON t.album_id = al.album_id
        INNER JOIN invoice_line il ON il.track_id = t.track_id
        GROUP BY al.album_id
        ORDER BY total_revenue DESC
        LIMIT 5;
```

5. **Find the average track length for each genre.**

```
>> SELECT g.name, avg(t.milliseconds) AS avg_length_milliseconds FROM track t
        JOIN genre g ON g.genre_id = t.genre_id
        GROUP BY g.name;
```

6. **List the customers who purchased tracks from more than ten genre.**

```
>> SELECT c.customer_id, c.first_name, c.last_name, COUNT(t.genre_id) AS genre_count FROM customer c
        JOIN invoice i ON i.customer_id = c.customer_id
        JOIN invoice_line il ON i.invoice_id = il.invoice_id
        JOIN track t ON t.track_id = il.track_id
        GROUP BY c.customer_id
        having COUNT(distinct t.genre_id) > 10;
```

7. **Identify the customers who made purchases in every playlist.**

```
>> SELECT c.customer_id, CONCAT(c.first_name, ' ', c.last_name) AS Name FROM customer c
        JOIN invoice i ON i.customer_id = c.customer_id
        JOIN invoice_line il ON il.invoice_id = i.invoice_id
        JOIN track t ON t.track_id = il.track_id
        JOIN playlist_track pt ON pt.track_id = t.track_id
        JOIN playlist p ON p.playlist_id = pt.playlist_id
```

```
GROUP BY c.customer_id
having COUNT(distinct p.playlist_id) = (SELECT COUNT(*) FROM playlist);
```

8. **Calculate the total revenue generated for each playlist.**

```
>> SELECT p.name, SUM(il.quantity * il.unit_price) AS Total_revenue FROM playlist p
        JOIN playlist_track pt ON pt.playlist_id = p.playlist_id
        JOIN track t ON t.track_id = pt.track_id
        JOIN invoice_line il ON il.track_id = t.track_id
        GROUP BY p.name;
```

9. **Find the tracks that are included in both the "Music" and "90's Music" playlists.**

```
>> SELECT t.track_id, t.name AS track_name FROM track t
        JOIN playlist_track pt ON pt.track_id = t.track_id
        JOIN playlist p ON p.playlist_id = pt.playlist_id
        WHERE p.name in ('music', "90's music")
        GROUP BY t.track_id
        having COUNT(distinct p.playlist_id) = 2;
```

10. **Find the total number of invoices issued each month in the year 2019.**

```
>> SELECT month(invoice_date) AS Month, COUNT(*) AS Total_Invoice FROM invoice
        WHERE year(invoice_date) = 2019
        GROUP BY month
        ORDER BY Total_Invoice DESC;
```

Question Set 3 – Advance

1. **List the customers who have spent more than the average total purchase amount. Calculate the average total purchase amount across all customers. Identify customers whose total spending exceeds this average. Return these customers' details, including their names and total purchase amounts.**

```
>> SELECT customer_id, first_name, last_name, total_spent
        FROM (
                SELECT c.customer_id, c.first_name, c.last_name, SUM(i.total) AS total_spent
                FROM customer c
                JOIN invoice i ON c.customer_id = i.customer_id
                GROUP BY c.customer_id, c.first_name, c.last_name
                ) AS customer_total_spent
        WHERE total_spent > (
                SELECT AVG(total_spent)
                FROM (
                        SELECT c.customer_id, SUM(i.total) AS total_spent
                        FROM customer c
                        JOIN invoice i ON c.customer_id = i.customer_id
                        GROUP BY c.customer_id
                        ) AS avg_total_spent
                );
```

2. **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 a.artist_id, a.name, SUM(il.unit_price * il.quantity) AS Total_Sale from artist a
                JOIN album al ON al.artist_id = a.artist_id
                JOIN track t ON t.album_id = al.album_id
                JOIN invoice_line il ON il.track_id = t.track_id
                GROUP BY 1, 2
                ORDER BY 3 DESC
                LIMIT 1
        )
```

```
select c.customer_id, c.first_name, c.last_name, bsa.name AS artist_name,
SUM(il.unit_price * il.quantity) AS Total_Spent from customer c
        JOIN invoice i ON i.customer_id = c.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 al ON al.album_id = t.album_id
        JOIN artist a ON a.artist_id = al.artist_id
        JOIN best_selling_artist bsa ON bsa.artist_id = a.artist_id
        GROUP BY 1,2,3,4
        ORDER BY 5 DESC;
```

3. **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 number 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 CountyGenreRank AS (
    SELECT c.Country, g.name AS genre, COUNT(*) AS genre_count,
        RANK() OVER (PARTITION BY c.Country ORDER BY COUNT(*) DESC) AS genre_rank
        FROM customer c
        JOIN invoice i ON c.customer_id = i.customer_id
        JOIN invoice_line il ON i.invoice_id = il.invoice_id
        JOIN track t ON il.track_id = t.track_id
        JOIN genre g ON t.genre_id = g.genre_id
        GROUP BY c. country, g.genre_id
        )
    SELECT country, genre, genre_count FROM CountyGenreRank
        WHERE genre_rank = 1;
```

4. **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 CountryTopCustomer AS (
    SELECT c.Country, c.first_name, c.last_name, SUM(il.unit_price * il.quantity) AS total_spent,
        RANK() OVER (PARTITION BY c.Country
        ORDER BY SUM(il.unit_price * il.quantity) DESC) AS customer_rank
        FROM customer c
        JOIN invoice i ON c.customer_id = i.customer_id
        JOIN invoice_line il ON i.invoice_id = il.invoice_id
        GROUP BY c.Country, c.customer_id
        )
    SELECT Country, first_name, last_name, total_spent
        FROM CountryTopCustomer
        WHERE customer_rank = 1;
```

5. **List the customers who have purchased more tracks than the average number of tracks per invoice.**

```
>> SELECT c.customer_id, c.first_name, c.last_name FROM customer c
    WHERE (SELECT COUNT(il.track_id)
            FROM invoice_line il
            WHERE il.invoice_id IN
            (SELECT i.invoice_id
                FROM invoice i
                WHERE i.customer_id = c.customer_id)
        ) > (SELECT AVG(track_count)
            FROM (SELECT COUNT(il.track_id) AS track_count
                    FROM invoice_line il
                    GROUP BY il.invoice_id
                    ) AS avg_tracks_per_invoice
        );
```