SQL PROJECT

MUSIC STORE ANALYSIS

PRESENTED BY - NAVANITA DAS

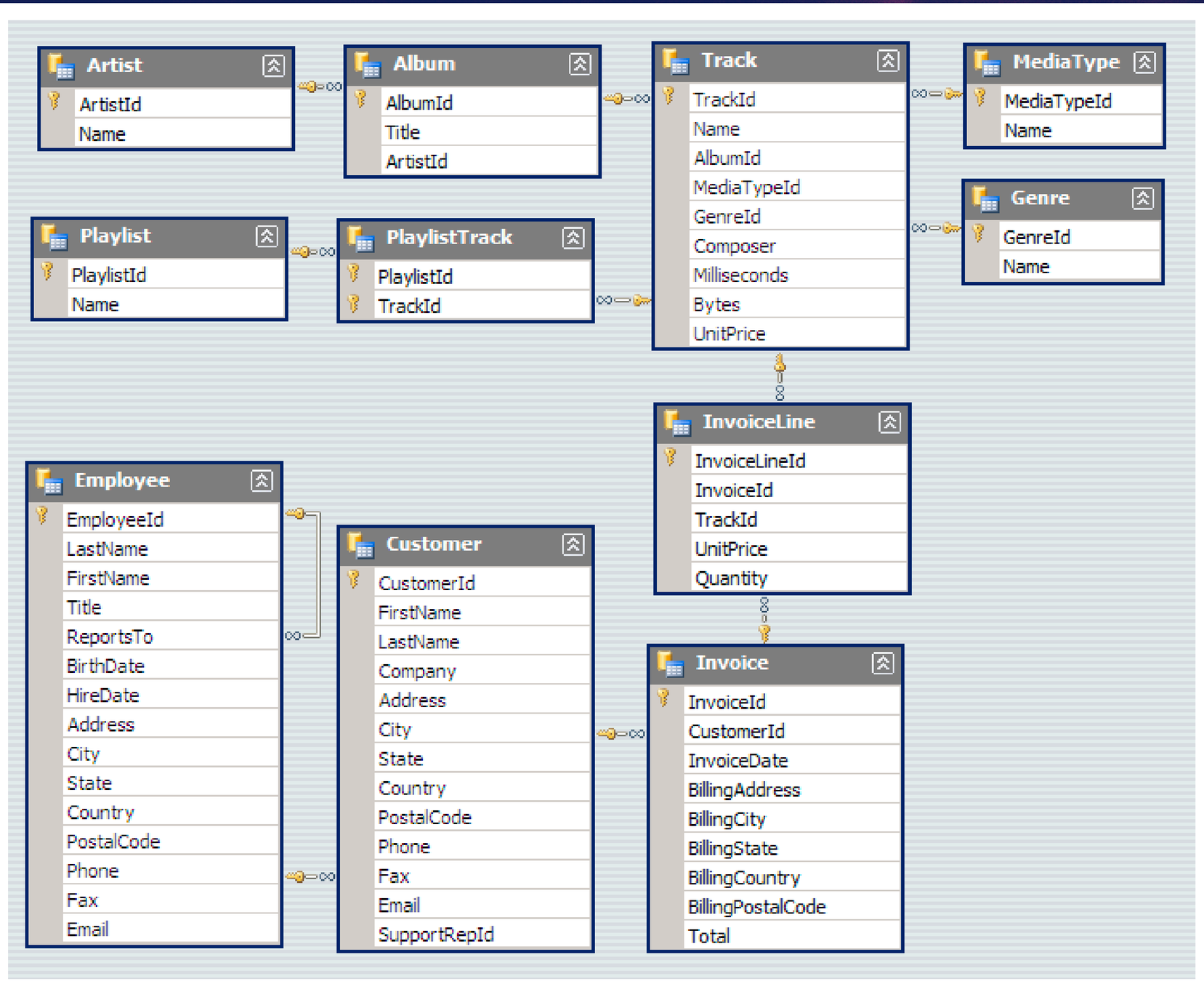
TOOLS
DATABASE

Postgre SQL

PgAdmin4

SCHEMA

MUSIC STORE DATABASE



QUERIES

01 Who is the senior most employee based on job title?

```
SELECT title, last_name, first_name
FROM employee
ORDER BY levels DESC
LIMIT 1
```

QUERIES

02 Which countries have the most Invoices?

```
SELECT COUNT(*) AS c, billing_country
FROM invoice
GROUP BY billing_country
ORDER BY c DESC
```

QUERIES

03 What are top 3 values of total invoice?

```
SELECT total
FROM invoice
ORDER BY total DESC
```

QUERIES

04 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.

QUERIES

04

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

QUERIES

05 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.

QUERIES

05

```
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;
```

QUERIES

06 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.

QUERIES

06

```
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;
```

QUERIES

07 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.

QUERIES

07

```
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;
```

QUERIES

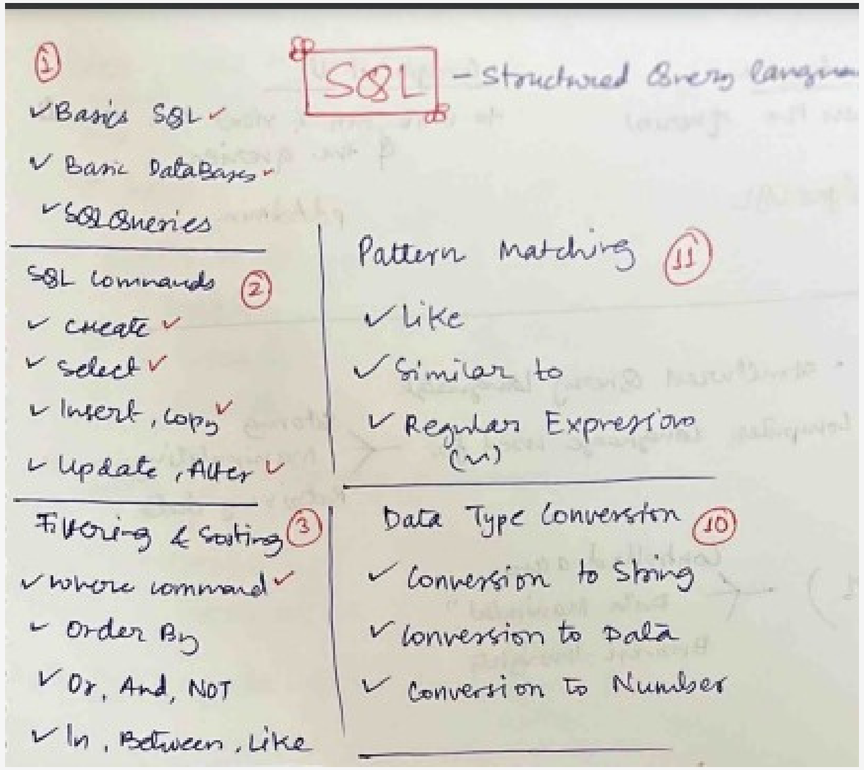
08 Return all the track names that have a song length longer than the average song length. Return the Name and Milliseconds for each track. Order by the song length with the longest songs listed first.

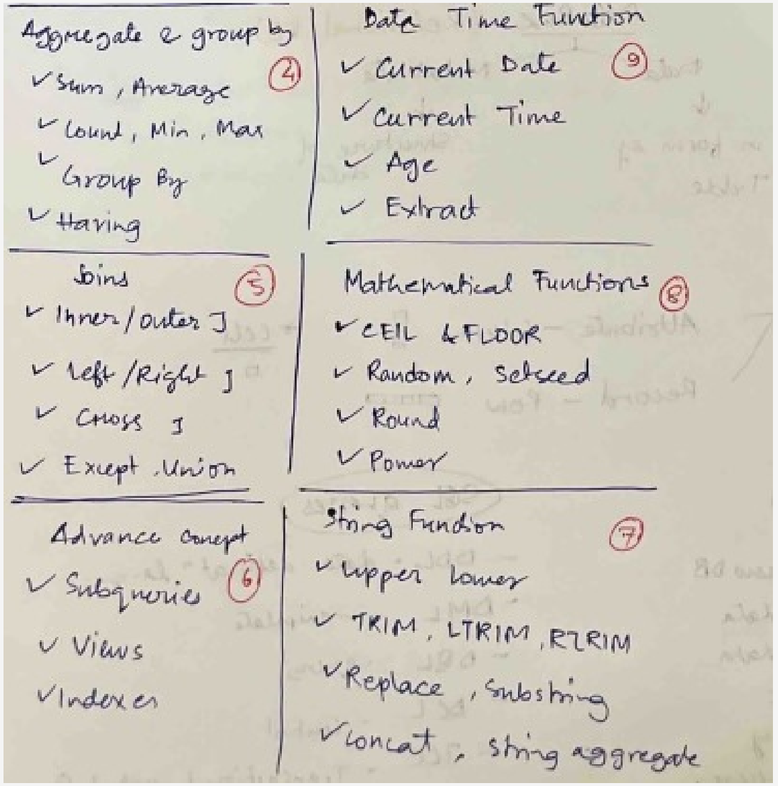
QUERIES

08

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

SQL


SQL - Structured Query Language
1
✓ Basics SQL ✓
✓ Basic DataBases ✓
✓ SQL Queries
SQL Commands 2
✓ create ✓
✓ select ✓
✓ Insert, copy ✓
✓ Update, Alter ✓
Filtering & Sorting 3
✓ where command ✓
✓ Order By
✓ Or, And, NOT
✓ In, Between, Like
Pattern Matching 11
✓ like
✓ Similar to
✓ Regular Expression (re)
Data Type Conversion 10
✓ Conversion to String
✓ Conversion to Date
✓ Conversion to Number


Aggregate & group by 4
✓ Sum, Average
✓ Count, Min, Max
✓ Group By
✓ Having
Joins 5
✓ Inner / Outer J
✓ Left / Right J
✓ Cross J
✓ Except, Union
Advance concept 6
✓ Subqueries
✓ Views
✓ Indexes
Data Time Function 9
✓ Current Date
✓ Current Time
✓ Age
✓ Extract
Mathematical Functions 8
✓ CEIL & FLOOR
✓ Random, Setseed
✓ Round
✓ Power
String Fundion 7
✓ Upper Lower
✓ TRIM, LTRIM, RTRIM
✓ Replace, Substring
✓ Concat, String aggregate

SQL PROJECT

NAVANITA DAS

THANK-YOU

PROJECT LINK

LI NKED IN LINK