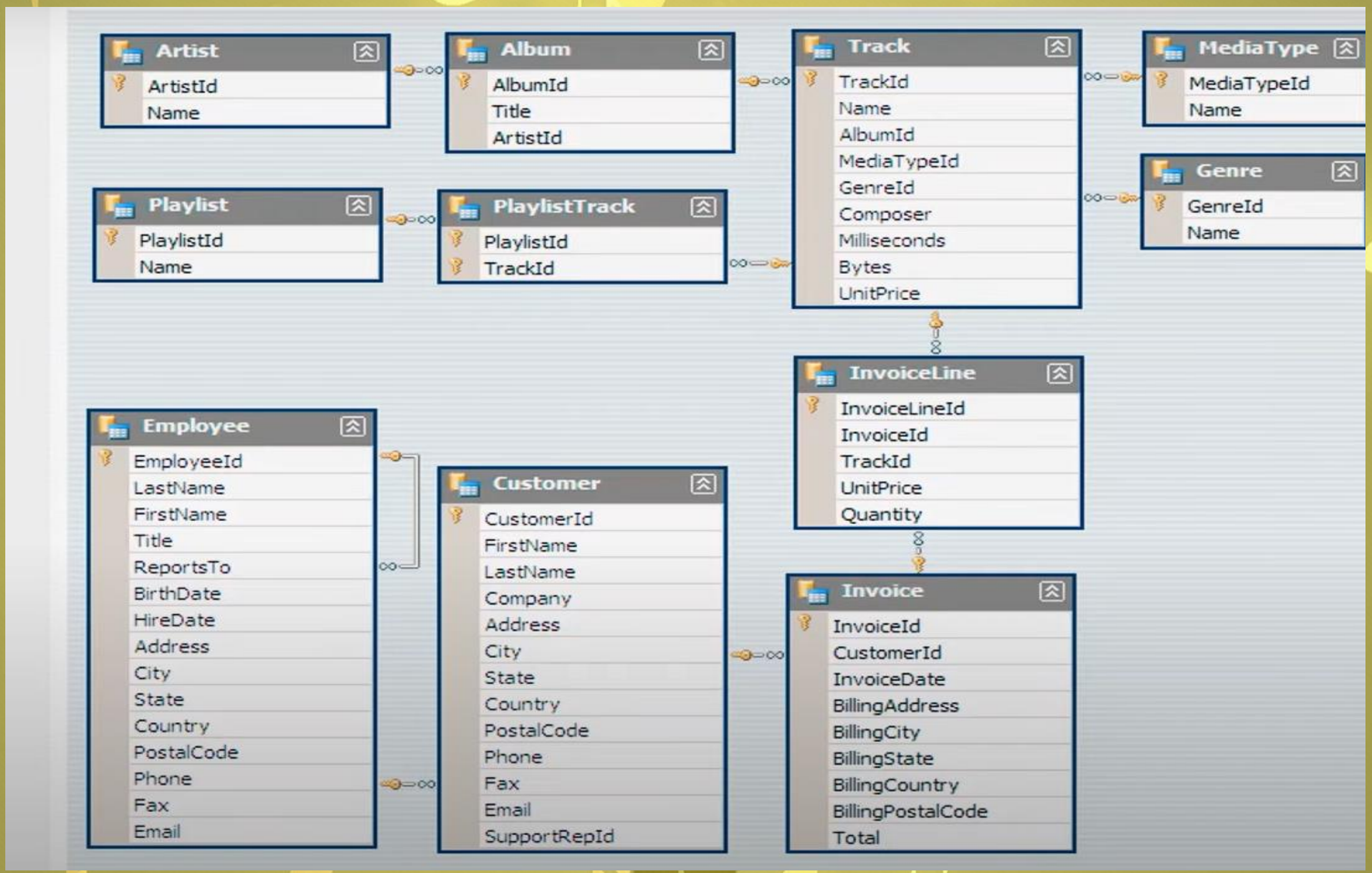

DIGITAL MUSIC
STORE
ANALYSIS

Objective of Project

- Examine the Data using SQL to help the Store understand its business growth and business strategy.
- Design and implement a relational database for a music store,
- Utilizing SQL to efficiently manage and query data related to artists, album, tracks, customers, sales.
- The goal is to create a robust and user-friendly database system that supports various functionalities within a music retail environment.

ENTITY RELATIONSHIP DIAGRAM



Q1: Who is the senior most employee based on the job title?

```sql
select * from employee
order by levels desc
limit 10

//Q2: Which countries have the most invoices //
```

Data Output | Messages | Notifications

	employee_id [PK] character varying (50)	last_name character	first_name character	title character varying (50)	reports_to character varying (30)	levels character varying (10)	birthdate timestamp without time zone	hire_date timestamp without tin
1	9	Madan ...	Mohan	Senior General Manager	[null]	L7	1961-01-26 00:00:00	2016-01-14 00:00:00
2	1	Adams ...	Andrew	General Manager	9	L6	1962-02-18 00:00:00	2016-08-14 00:00:00
3	2	Edwards ...	Nancy	Sales Manager	1	L4	1958-12-08 00:00:00	2016-05-01 00:00:00
4	6	Mitchell ...	Michael	IT Manager	1	L3	1973-07-01 00:00:00	2016-10-17 00:00:00
5	7	King	Robert	IT Staff	6	L2	1970-05-29 00:00:00	2017-01-02 00:00:00
6	8	Callahan ...	Laura	IT Staff	6	L2	1968-01-09 00:00:00	2017-03-04 00:00:00
7	5	Johnson ...	Steve	Sales Support Agent	2	L1	1965-03-03 00:00:00	2017-10-17 00:00:00
8	3	Peacock ...	Jane	Sales Support Agent	2	L1	1973-08-29 00:00:00	2017-04-01 00:00:00
9	4	Park ...	Margaret	Sales Support Agent	2	L1	1947-09-19 00:00:00	2017-05-03 00:00:00

Q2: Which countries have the most invoices

```
select * from invoice;

select count(*) as c, billing_country
from invoice
group by billing_country
order by c desc
```

Data Output | Messages | Notifications

	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
7	29	Portugal
8	28	United Kingdom
9	21	India
10	13	Chile
11	13	Ireland
12	11	Spain
13	11	Finland

```
join invoice i on c.customer_id = i.customer_id
```

Q3: What are top 3 values of total invoices?

```
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

Q: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 invoices totals.
Return both city name & sum of all invoices totals

```sql
select sum(total) as invoice_total, billing_city
from invoice
group by billing_city
order by invoice_total desc
```

Data Output | Messages | Notifications

	invoice_total double precision	billing_city character varying (30)
1	273.24000000000007	Prague
2	169.29	Mountain View
3	166.32	London
4	158.4	Berlin
5	151.47	Paris
6	129.69	São Paulo
7	114.83999999999997	Dublin
8	111.86999999999999	Delhi
9	108.89999999999998	São José dos Campos
10	106.91999999999999	Brasília
11	102.96000000000001	Lisbon
12	99.99	Bordeaux
13	99.99	Montréal

```sql
    where genre.name Like 'Rock'
)
```

otal rows: 53 of 53 | Query complete 00:00:00.121

Q5: 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 money.

```sql
select * from customer
select c.customer_id, c.first_name, c.last_name,sum(i.total) as total
from customer c
join invoice i on c.customer_id = i.customer_id
group by c.customer_id
order by total desc
limit 1
```

Data Output | Messages | Notifications

	invoice_total double precision	billing_city character varying (30)
1	273.24000000000007	Prague
2	169.29	Mountain View
3	166.32	London
4	158.4	Berlin
5	151.47	Paris
6	129.69	São Paulo
7	114.83999999999997	Dublin
8	111.86999999999999	Delhi
9	108.89999999999998	São José dos Campos

Total rows: 53 of 53 | Query complete 00:00:00.121

Q6: Write query to return the email, first name, last name, & Genre of all Rock Music Listener. Return your list ordered alphabetically by email starting with A?

```
select distinct email, first_name, last_name
from customer c
join invoice i on c.customer_id = i.customer_id
join invoice_line il on i.invoice_id = il.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 character varying (50)	first_name character	last_name character
1	aaronmitchell@yahoo.ca	Aaron	Mitchell
2	alero@uol.com.br	Alexandre	Rocha
3	astrid.gruber@apple.at	Astrid	Gruber
4	bjorn.hansen@yahoo.no	Bjørn	Hansen
5	camille.bernard@yahoo.fr	Camille	Bernard
6	daan_peeters@apple.be	Daan	Peeters
7	diego.gutierrez@yahoo.ar	Diego	Gutiérrez
8	dmiller@comcast.com	Dan	Miller
9	dominiquelefebvre@gmail.c...	Dominique	Lefebvre
10	edfrancis@yachoo.ca	Edward	Francis

```
from track
where milliseconds >(
```

al rows: 59 of 59 | Query complete 00:00:00.143

Q7: 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 tract count of the top 10 rock bands.

```sql
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.album_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

	email character varying (50)	first_name character	last_name character
1	aaronmitchell@yahoo.ca	Aaron	Mitchell
2	alero@uol.com.br	Alexandre	Rocha
3	astrid.gruber@apple.at	Astrid	Gruber
4	bjorn.hansen@yahoo.no	Bjørn	Hansen
5	camille.bernard@yahoo.fr	Camille	Bernard
6	daan_peeters@apple.be	Daan	Peeters
7	diego.gutierrez@yahoo.ar	Diego	Gutiérrez
8	dmiller@comcast.com	Dan	Miller

Total rows: 59 of 59 | Query complete 00:00:00.143

--Q8: Return all the tracks 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 song listed first

```
72  select name, milliseconds
73  from track
74  where milliseconds >(
75          select avg(milliseconds) as avg_track_length
76          from track)
77  order by milliseconds desc;
78
```

Data Output | Messages | Notifications

	name character varying (150)	milliseconds integer
1	Occupation / Precipice	5286953
2	Through a Looking Glass	5088838
3	Greetings from Earth, Pt. 1	2960293
4	The Man With Nine Lives	2956998
5	Battlestar Galactica, Pt. 2	2956081
6	Battlestar Galactica, Pt. 1	2952702
7	Murder On the Rising Star	2935894
8	Battlestar Galactica, Pt. 3	2927802
9	Take the Celestra	2927677
10	Fire In Space	2926593

Q9: Find how much amount spent by each customer on artists? Write a query to return customer name,
-- artist name and total spent?

```
-- 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;
```

Data Output | Messages | Notifications

	customer_id integer	first_name character	last_name character	artist_name character varying (120)	amount_spent double precision
1	46	Hugh	O'Reilly	Queen	27.719999999999985
2	38	Niklas	Schröder	Queen	18.81
3	3	François	Tremblay	Queen	17.82
4	34	João	Fernandes	Queen	16.830000000000002
5	53	Phil	Hughes	Queen	11.88
6	41	Marc	Dubois	Queen	11.88
7	47	Lucas	Mancini	Queen	10.89
8	33	Ellie	Sullivan	Queen	10.89
9	20	Dan	Miller	Queen	3.96

Total rows: 43 of 43 | Query complete 00:00:00.079 | Ln 105, Col