

|  |
| --- |
| PROJECT ON EXCEL **CHINOOK ANALYSIS** |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Developed by**  **Members:**   |  |  |  | | --- | --- | --- | | **No.** | **Student Name** | **Student ID** | | 1 | Nguyễn Văn Khang |  | | 2 | Lê Thảo Vy |  | | 3 | Trịnh Đình Hiếu |  | | 4 | Trần Trương Minh Thắng |  |  * **Class No.: DT2210L** * **Start Date:** * **End Date:** * **Name of the Coordinator:** * **Date of Submission:** |

|  |
| --- |
| I. GET THE DATA |

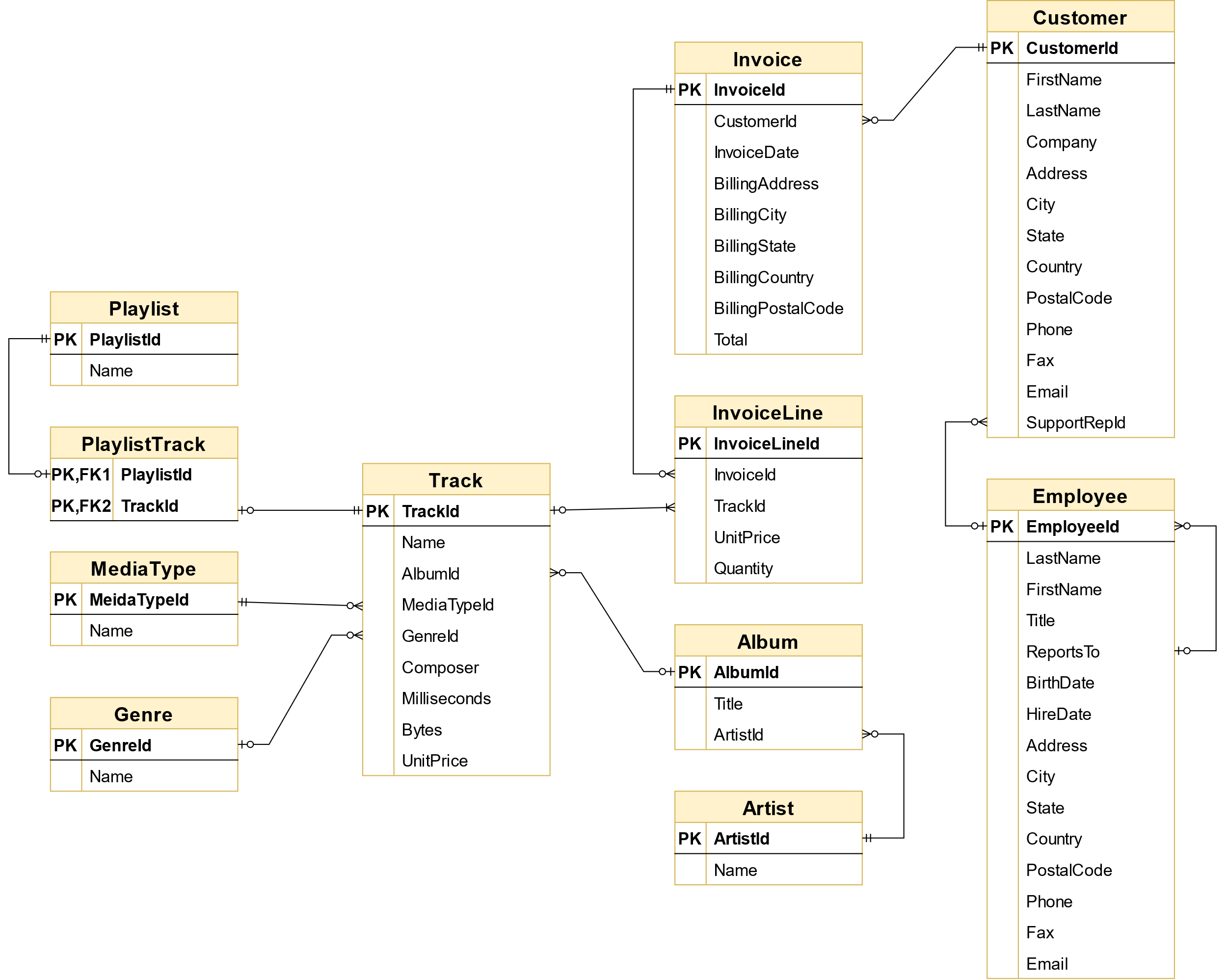
The Chinook data model represents a digital media store, including tables for artists, albums, media tracks, invoices, and customers.

* Media-related data was created using real data from an Apple iTunes library.
* Customer and employee information was created using fictitious names and addresses that can be located on Google maps, and other well formatted data (phone, fax, email, etc.)
* Sales information was auto generated using random data for a four-year period.

The Chinook sample database includes:

* 11 tables
* A variety of indexes, primary and foreign key constraints
* Over 15,000 rows of data

For details, here's the entity relationship diagram of the Chinook data model.



|  |
| --- |
| II. OVERVIEW OF THE DATA |

SELECT Name, Composer FROM track LIMIT 10;

|  |  |
| --- | --- |
| Name | Composer |
| For Those About To Rock (We Salute You) | Angus Young, Malcolm Young, Brian Johnson |
| Balls to the Wall |  |
| Fast As a Shark | F. Baltes, S. Kaufman, U. Dirkscneider & W. Hoffman |
| Restless and Wild | F. Baltes, R.A. Smith-Diesel, S. Kaufman, U. Dirkscneider & W. Hoffman |
| Princess of the Dawn | Deaffy & R.A. Smith-Diesel |
| Put The Finger On You | Angus Young, Malcolm Young, Brian Johnson |
| Let's Get It Up | Angus Young, Malcolm Young, Brian Johnson |
| Inject The Venom | Angus Young, Malcolm Young, Brian Johnson |
| Snowballed | Angus Young, Malcolm Young, Brian Johnson |
| Evil Walks | Angus Young, Malcolm Young, Brian Johnson |

SELECT FristName, LastName, Country, SupportRepId FROM customer LIMIT 10;

|  |  |  |  |
| --- | --- | --- | --- |
| FirstName | LastName | Country | SupportRepId |
| Luís | Gonçalves | Brazil | 3 |
| Leonie | Köhler | Germany | 5 |
| François | Tremblay | Canada | 3 |
| Bjørn | Hansen | Norway | 4 |
| František | Wichterlová | Czech Republic | 4 |

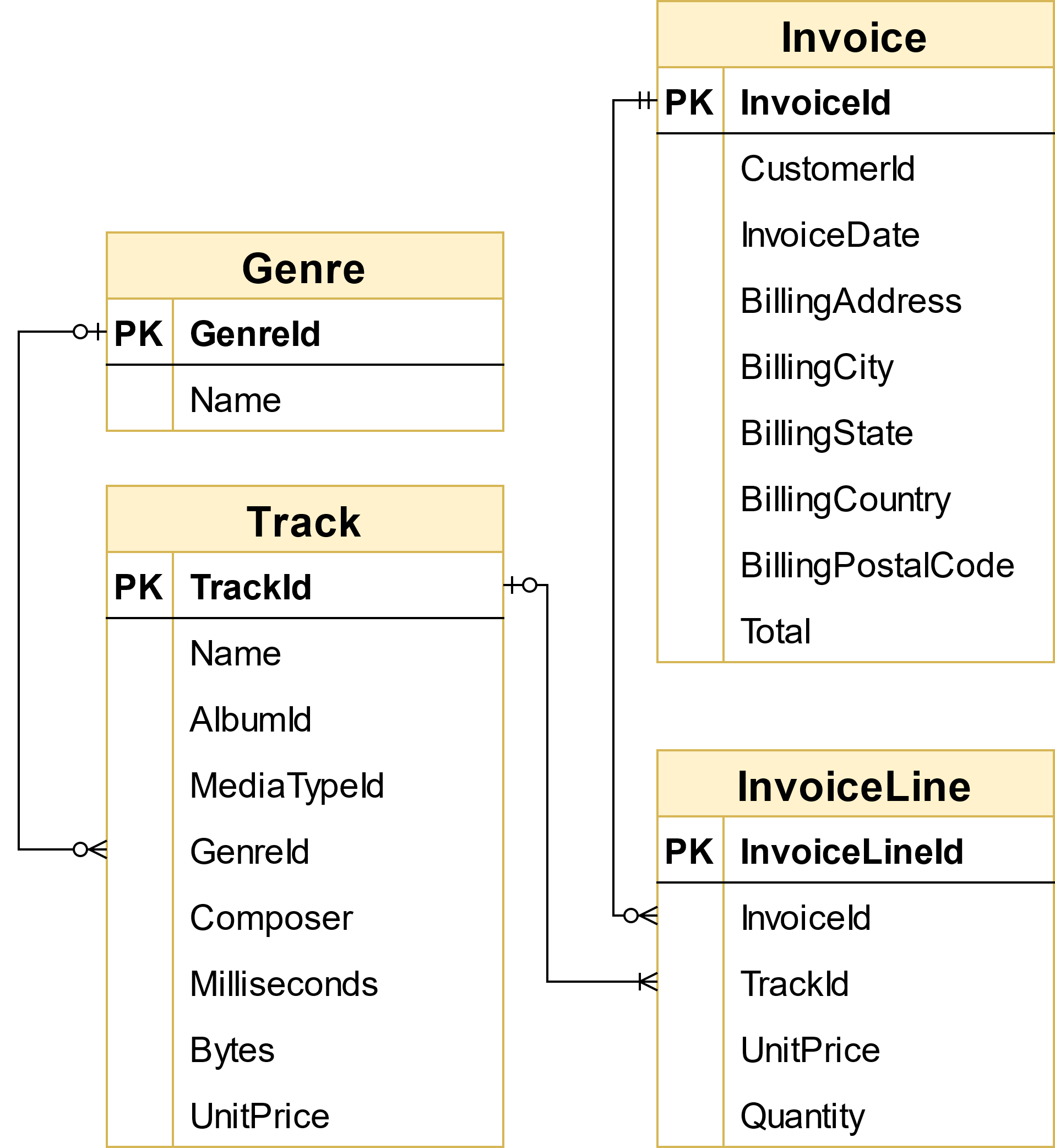
|  |
| --- |
| III. ANALYZE THE DATA |

### 1. Selecting Albums to Purchase

#### 1.1 Which genres sell the most tracks in the USA

The record label specializes in artists from the USA, and they have given Chinook some money to advertise the new albums in the USA, so we're interested in finding out which genres sell the best in the USA.

You'll need to write a query to find out which genres sell the most tracks in the USA, write up a summary of your findings, and make a recommendation for the three artists whose albums we should purchase for the store.



- We can join 4 tables: Genre, Track, Invoice, InvoiceLine because we need information about genres, track, sell or invoice and invoice line.

- To find out which genre sell the most tracks – we need to sum all invoiceline.quantity (total\_sold) and group by genre.

- We will sort descending the total\_sold value.

- This is our query:

SELECT genre.Name AS genre, SUM(invoiceline.Quantity) AS total\_sold

FROM genre

JOIN track ON genre.Genreid = track.Genreid

JOIN invoiceline ON track.TrackId = invoiceline.TrackId

JOIN invoice ON invoiceline.InvoiceId = invoice.InvoiceId

WHERE invoice.BillingCountry = 'USA'

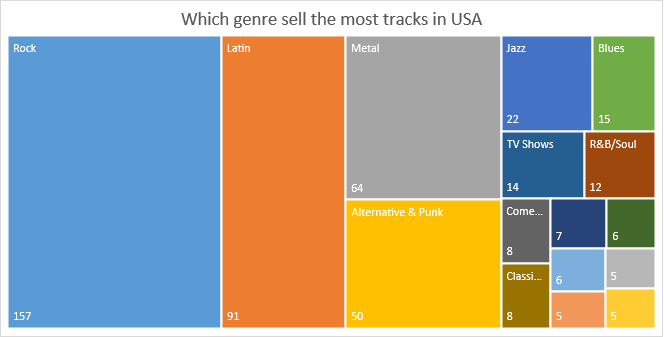
GROUP BY genre.GenreId

ORDER BY total\_sold DESC

LIMIT 8;

We get the table result below:

|  |  |
| --- | --- |
| Genre | Total\_Sold |
| Rock | 157 |
| Latin | 91 |
| Metal | 64 |
| Alternative & Punk | 50 |
| Jazz | 22 |
| Blues | 15 |
| TV Shows | 14 |
| R&B/Soul | 12 |

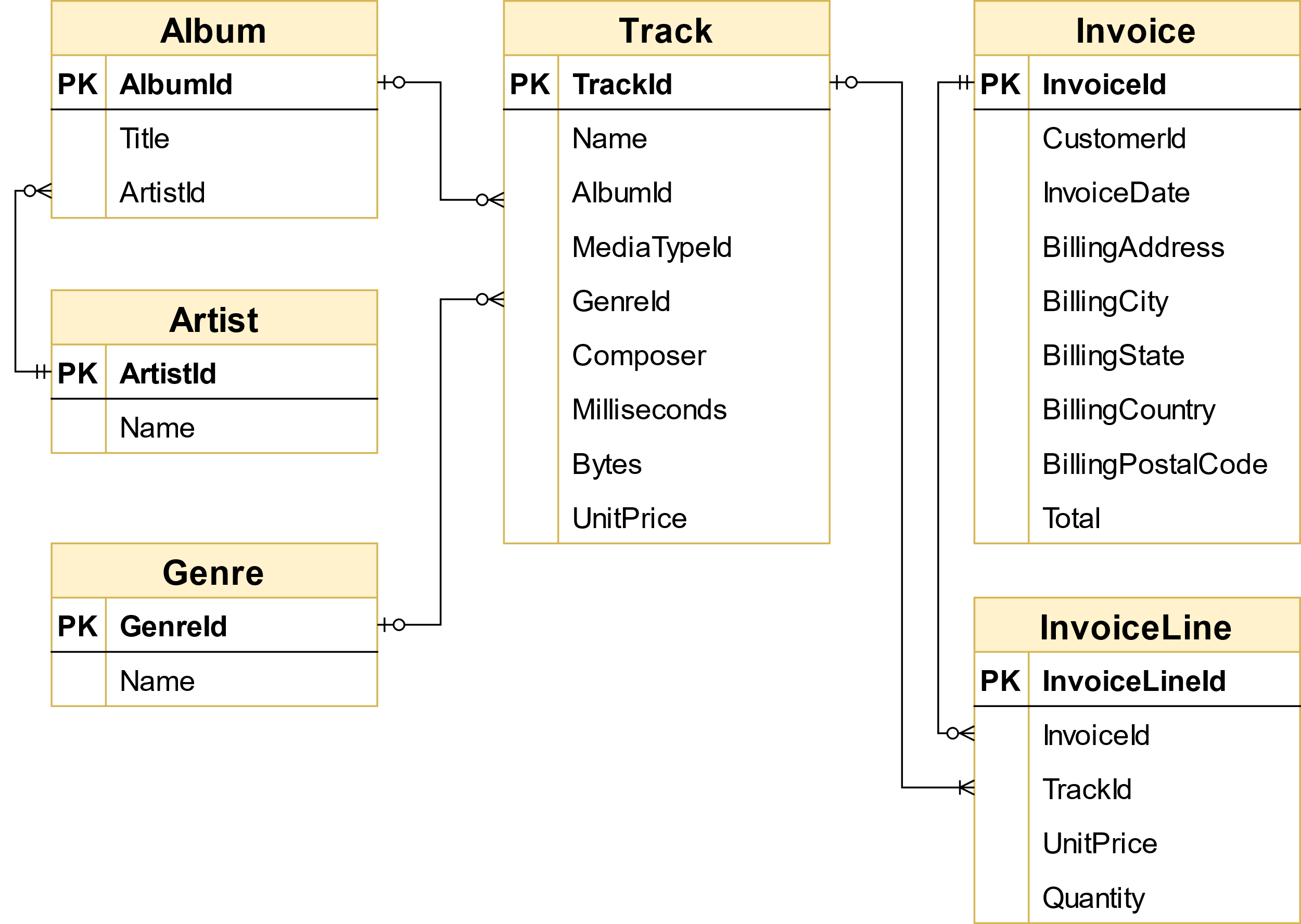


Base on above table, we see top 3 genres have most sells is Rock, Latin and Metal.

#### 1.2 Recommendation for the three artists whose albums we should purchase for the store

- To find artists whose albums we should purchase, we will recommend base on which album has the most track sold.

- Therefore, we need 4 tables above and 2 more tables: albums and artist.



- We still use SUM(invoiceline.quantity) to count total sold tracks but in this case we will group by album. We also narrow down with Rock and USA. We have this query:

SELECT artist.Name AS Artist,

SUM(invoiceline.Quantity) AS Total\_sold

FROM artist

JOIN album ON artist.ArtistId = album.ArtistId

JOIN track ON album.AlbumId = track.AlbumId

JOIN genre ON genre.GenreId = track.GenreId

JOIN invoiceline ON track.TrackId = invoiceline.TrackId

JOIN invoice ON invoiceline.InvoiceId = invoice.InvoiceId

WHERE

invoice.BillingCountry = 'USA' AND genre.name = 'Rock'

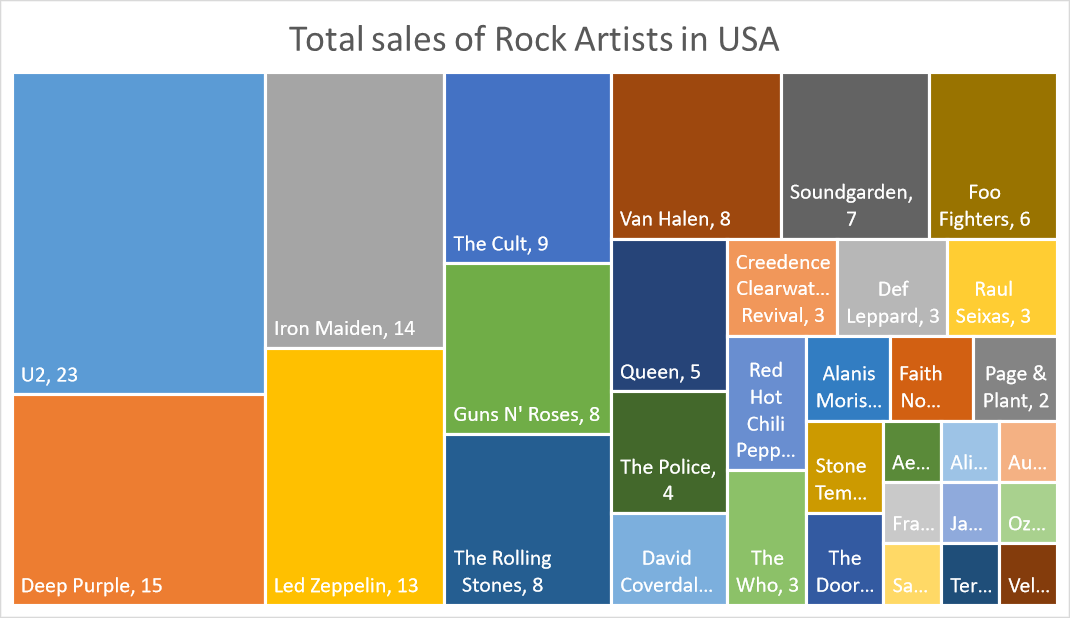
GROUP BY artist.Name

ORDER BY total\_sold DESC

LIMIT 3;

They are 3 artists in Rock you should purchase in US:

|  |  |
| --- | --- |
| Artist | Total\_Sold |
| U2 | 23 |
| Deep Purple | 15 |
| Iron Maiden | 14 |

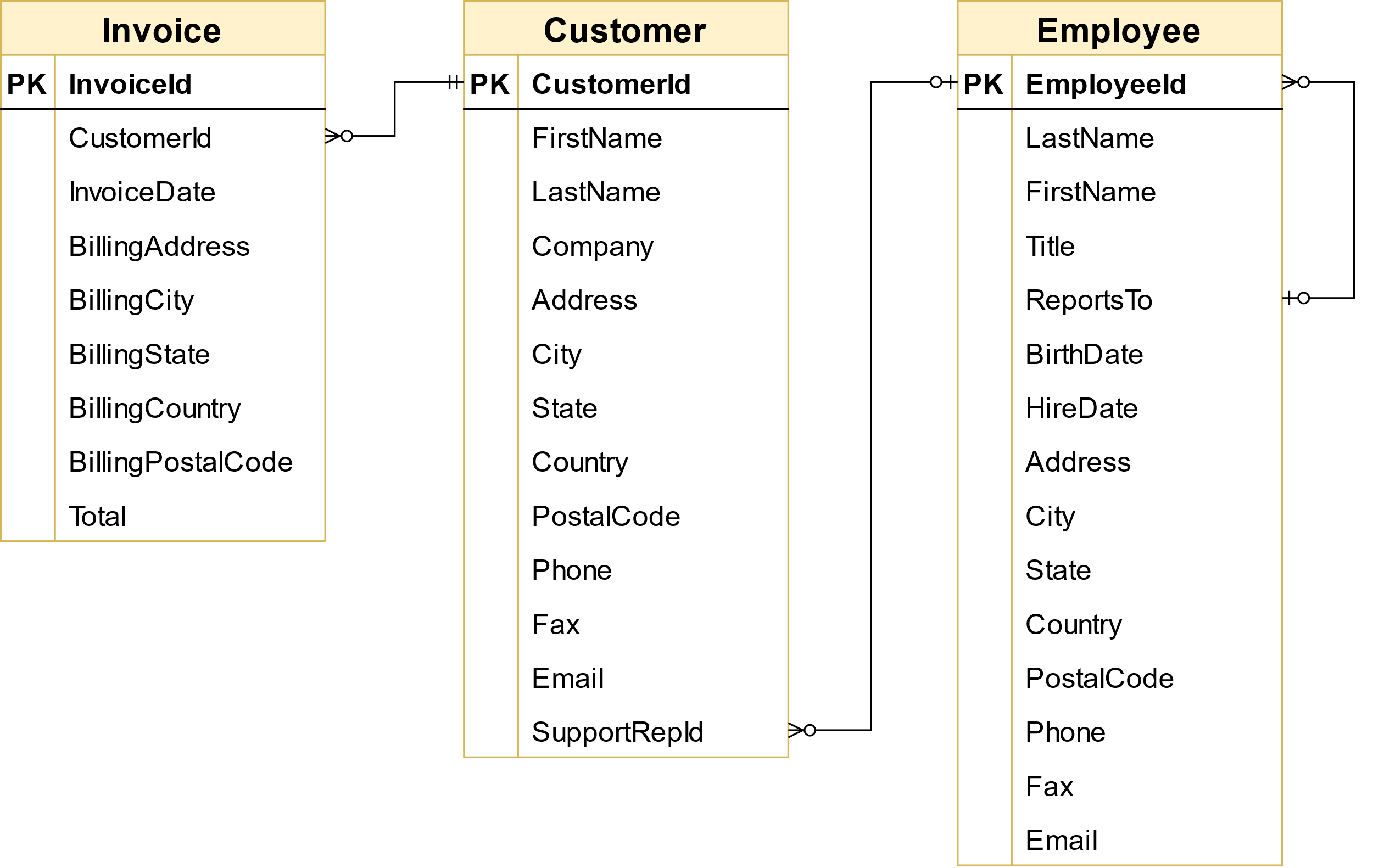


### 2. Analyzing Sales Agent performance

- To evaluate employee performance, we should know:

* Name of employee,
* HireDate
* All invoices with her/his name - COUNT(\*) AS Number\_of\_sales
* Number\_of\_customers - COUNT(Distinct c.customerid)
* Total\_revenue – SUM(invoice.Total)
* Revenue/customer (Total\_revenue / Number\_of\_invoice)

- We can get this information with 3 tables:



SELECT e.EmployeeId,

CONCAT(e.firstname,' ',e.lastname) AS Sales\_agent\_name,

CAST(e.HireDate AS DATE) AS HireDate,

COUNT(\*) AS Number\_of\_sales,

COUNT(DISTINCT c.customerid) AS Number\_of\_customers,

SUM(i.total) AS Total\_revenue,

ROUND(SUM(i.total) / COUNT(\*),2) AS Revenue\_per\_invoice

FROM invoice i

JOIN customer c ON c.customerid = i.customerid

JOIN employee e ON e.employeeid = c.supportrepid

GROUP BY employeeid

ORDER BY Total\_revenue;

We get this result:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Employee Id | Sale agent name | Hire Date | No of sales | No of customers | Total Revenue | Revenue/invoice |
| 5 | Steve Johnson | 2003-10-17 | 126 | 18 | 720.16 | 5.72 |
| 4 | Margaret Park | 2003-05-03 | 140 | 20 | 775.40 | 5.54 |
| 3 | Jane Peacock | 2002-04-01 | 146 | 21 | 833.04 | 5.71 |

Comment: If you look at revenue and number of customers, Jane Peacock does the best - but keep in mind this is a senior employee

Comment:

* Jane Peacock is the first sale agent of store and she has the highest revenue as well as number of customers and total invoices.
* Although Steve Johnson is the newest sale agent, but he has the highest Revenue/invoice. He is the employee with the best performance.

### 3. Analyzing Sales by Country

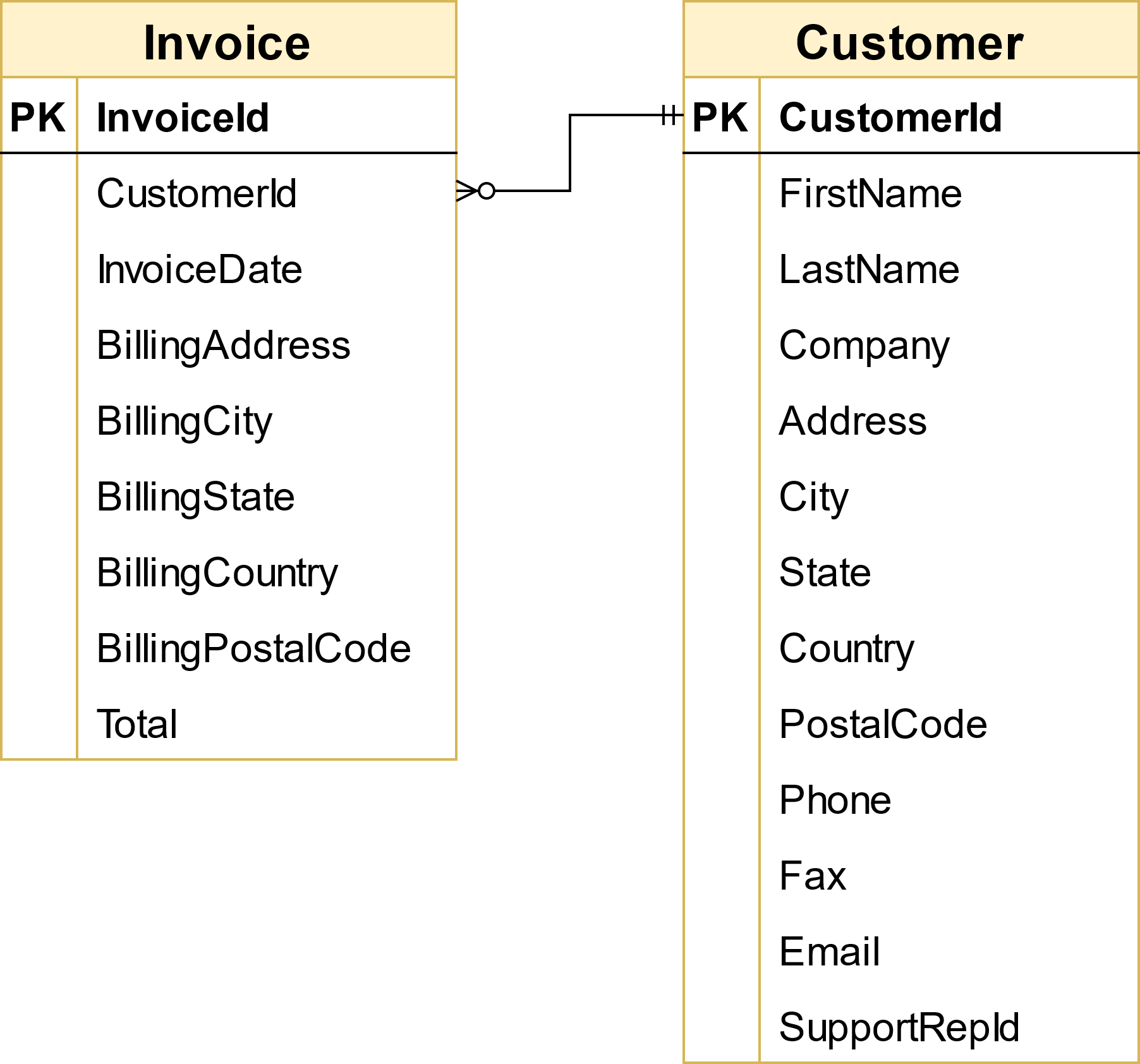
- Analyze the sales data for customers from each different country

- In particular, we need to calculate data, for each country, on:

* total number of customers
* total value of sales
* average value of sales per customer
* average order value

- Because there are several countries with only one customer, we will group these customers as "Other" in our analysis. Let’s find out how many customers for each country and classify which countries is “Other”.

- We need information from 2 tables: customer and invoice



SELECT Country,

COUNT(DISTINCT c.customerid) AS Number\_of\_customers,

SUM(i.Total) AS Value\_of\_sales,

COUNT(DISTINCT i.invoiceid) AS Number\_of\_orders,

(CASE

WHEN COUNT(DISTINCT c.customerid) = 1 THEN 'Other'

ELSE Country

END) AS Country\_sort

FROM customer c

JOIN invoice i ON c.customerid = i.customerid

GROUP BY country

We get this result:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Country | No\_of\_customers | Value\_of\_sales | No\_of\_orders | Country\_sort |
| Argentina | 1 | 37.62 | 7 | Other |
| Australia | 1 | 37.62 | 7 | Other |
| Austria | 1 | 42.62 | 7 | Other |
| Belgium | 1 | 37.62 | 7 | Other |
| Brazil | 5 | 190.10 | 35 | Brazil |
| Canada | 8 | 303.96 | 56 | Canada |
| Chile | 1 | 46.62 | 7 | Other |
| Czech Republic | 2 | 90.24 | 14 | Czech Republic |
| Denmark | 1 | 37.62 | 7 | Other |
| Finland | 1 | 41.62 | 7 | Other |
| France | 5 | 195.10 | 35 | France |
| Germany | 4 | 156.48 | 28 | Germany |
| Hungary | 1 | 45.62 | 7 | Other |
| India | 2 | 75.26 | 13 | India |
| Ireland | 1 | 45.62 | 7 | Other |
| Italy | 1 | 37.62 | 7 | Other |
| Netherlands | 1 | 40.62 | 7 | Other |
| Norway | 1 | 39.62 | 7 | Other |
| Poland | 1 | 37.62 | 7 | Other |
| Portugal | 2 | 77.24 | 14 | Portugal |
| Spain | 1 | 37.62 | 7 | Other |
| Sweden | 1 | 38.62 | 7 | Other |
| United Kingdom | 3 | 112.86 | 21 | United Kingdom |
| USA | 13 | 523.06 | 91 | USA |

Now, everything is clear and easy to understand, let’s group by Country\_sort and calculate 4 creteria that we want to find above, we use CTE in this case:

WITH country\_stat AS

(SELECT Country,

COUNT(DISTINCT c.customerid) AS Number\_of\_customers,

SUM(i.Total) AS Value\_of\_sales,

COUNT(DISTINCT i.invoiceid) AS Number\_of\_orders,

(CASE

WHEN COUNT(DISTINCT c.customerid) = 1 THEN 'Other'

ELSE Country

END) AS Country\_sort

FROM customer c

JOIN invoice i ON c.customerid = i.customerid

GROUP BY country)

SELECT Country\_sort,

SUM(Number\_of\_customers) AS Total\_number\_of\_customers,

SUM(Value\_of\_sales) AS Total\_value\_of\_sales,

ROUND(SUM(Value\_of\_sales) / SUM(Number\_of\_customers), 2) AS AVG\_value\_of\_sales\_per\_customer,

ROUND(SUM(Value\_of\_sales) / SUM(Number\_of\_orders), 2) AS AVG\_order\_value

FROM Country\_stat

GROUP BY Country\_sort

ORDER BY Total\_value\_of\_sales DESC;

- We get this result:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Country\_sort | Total no of customers | Total value of sales | AVG value of sales per customer | AVG order value |
| Other | 15 | 604.30 | 40.29 | 5.76 |
| USA | 13 | 523.06 | 40.24 | 5.75 |
| Canada | 8 | 303.96 | 38.00 | 5.43 |
| France | 5 | 195.10 | 39.02 | 5.57 |
| Brazil | 5 | 190.10 | 38.02 | 5.43 |
| Germany | 4 | 156.48 | 39.12 | 5.59 |
| United Kingdom | 3 | 112.86 | 37.62 | 5.37 |
| Czech Republic | 2 | 90.24 | 45.12 | 6.45 |
| Portugal | 2 | 77.24 | 38.62 | 5.52 |
| India | 2 | 75.26 | 37.63 | 5.79 |

- Comment:

* USA has the highest values in total customers, total sales
* Czech has the highest AVG order value and AVG value of sale/customer but only have 2 customers

### 4. Albums vs Individual Tracks

- The Chinook store is setup in a way that allows customer to make purchases in one of the two ways:

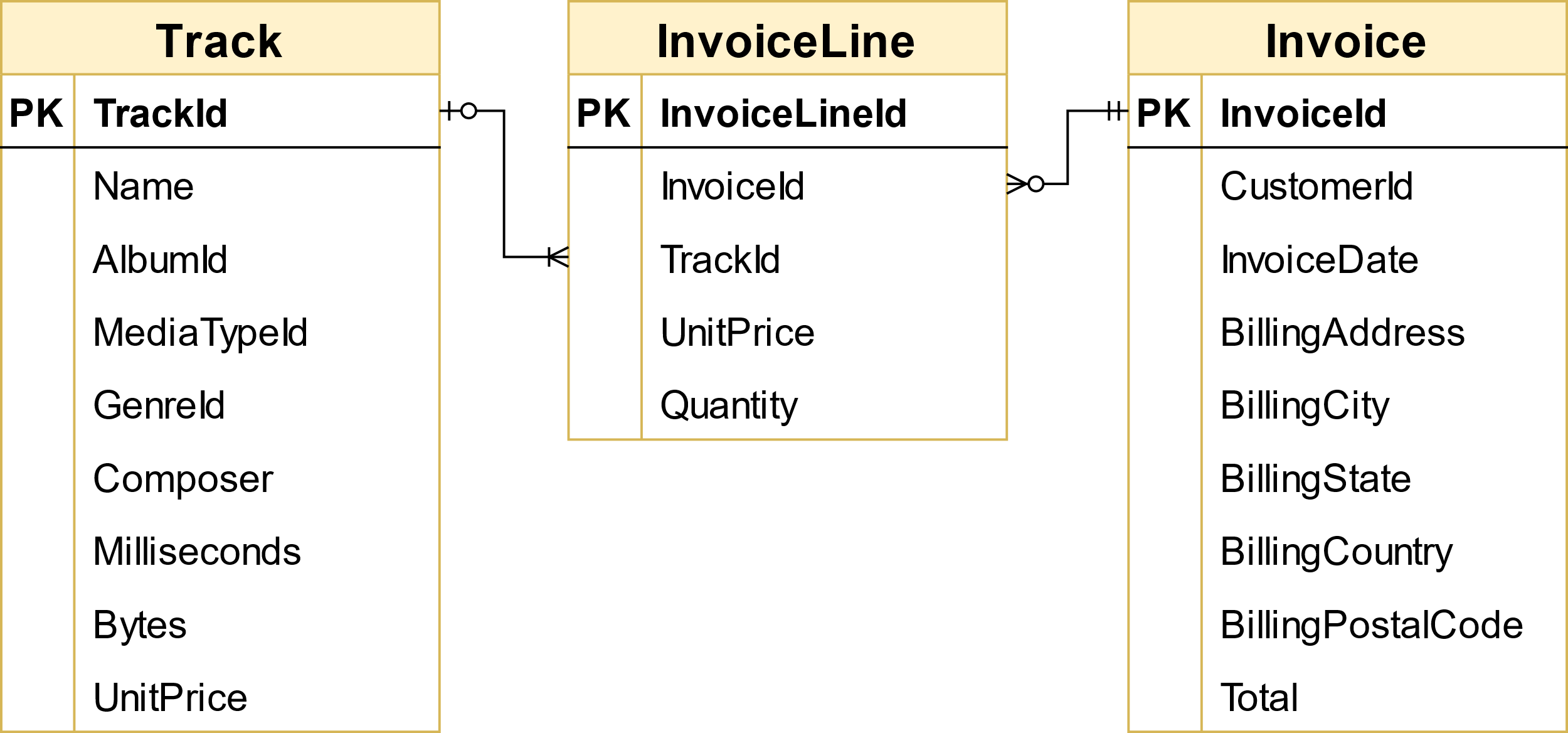
* purchase a whole album (if customers purchase a whole album, then they can not add individual tracks to that same purchase - unless they do that by choosing each track manually)
* purchase a collection of one or more individual tracks.

- Management is currently considering changing their purchasing strategy to save money. The strategy they are considering is to purchase only the most popular tracks from each album from record companies, instead of purchasing every track from an album.

- We must find out what percentage of purchases are individual tracks vs whole albums, so that management can use this data to understand the effect this decision might have on overall revenue

- Let’s get the list of tracks from an invoice and comparing it to the list of tracks from an album. We can find the album to compare the purchase to by looking up the album that one of the purchased tracks belongs to. It doesn't matter which track we pick, since if it's an album purchase, that album will be the same for all tracks.

- We will need 3 tables: invoice, invoiceline and track.



- Firstly, we need to know 2 things:

* How many tracks were selected for each invoice?
* How many tracks does each album have?

- This is a query to find out:

SELECT

i.invoiceid,

t.albumid,

COUNT(DISTINCT t.trackid) AS invoice\_track\_count,

(

SELECT COUNT(\*)

FROM track t2

WHERE t2.albumid = t.albumid

) AS album\_track\_count

FROM invoice i

JOIN invoiceline ii ON i.invoiceid = ii.invoiceid

JOIN track t ON ii.trackid = t.trackid

GROUP BY i.invoiceid, t.albumid

LIMIT 10;

- We get this result:

|  |  |  |  |
| --- | --- | --- | --- |
| Invoice Id | Album Id | Invoice\_track\_count | Album\_track\_count |
| 1 | 2 | 1 | 1 |
| 1 | 3 | 1 | 3 |
| 2 | 1 | 4 | 10 |
| 3 | 4 | 2 | 8 |
| 3 | 5 | 4 | 15 |
| 4 | 6 | 2 | 13 |
| 4 | 7 | 2 | 12 |
| 4 | 8 | 2 | 14 |
| 4 | 9 | 2 | 8 |
| 4 | 10 | 1 | 14 |

- Now we will divide into 2 cases:

1. album\_track\_count = invoice\_track\_count AND invoice\_track\_count > 1. This case we will be counted buying whole album.
2. album\_track\_count <> invoice\_track\_count OR invoice\_track\_count = 1. This case we will be counted buying individual tracks.

- We have this query to find out the percentage of buying whole album vs buying individual tracks:

SELECT

SUM(CASE

WHEN album\_track\_count = invoice\_track\_count

AND invoice\_track\_count > 1 THEN 1

ELSE 0

END) AS whole\_albums\_count,

SUM(CASE

WHEN album\_track\_count <> invoice\_track\_count

OR invoice\_track\_count = 1 THEN 1

ELSE 0

END) AS individual\_tracks\_count,

COUNT(\*) AS total\_invoices\_count,

(SUM(CASE

WHEN album\_track\_count = invoice\_track\_count

AND invoice\_track\_count > 1 THEN 1

ELSE 0

END) / COUNT(\*)) \* 100 AS whole\_albums\_percentage,

(SUM(CASE

WHEN album\_track\_count <> invoice\_track\_count

OR invoice\_track\_count = 1 THEN 1

ELSE 0

END) / COUNT(\*)) \* 100 AS individual\_tracks\_percentage

FROM (SELECT

i.invoiceid,

t.albumid,

COUNT(DISTINCT t.trackid) AS invoice\_track\_count,

(SELECT COUNT(\*)

FROM track t2

WHERE t2.albumid = t.albumid) AS album\_track\_count

FROM invoice i

JOIN invoiceline ii ON i.invoiceid = ii.invoiceid

JOIN track t ON ii.trackid = t.trackid

GROUP BY i.invoiceid, t.albumid

) AS invoice\_album\_track\_counts;

- We have this result:

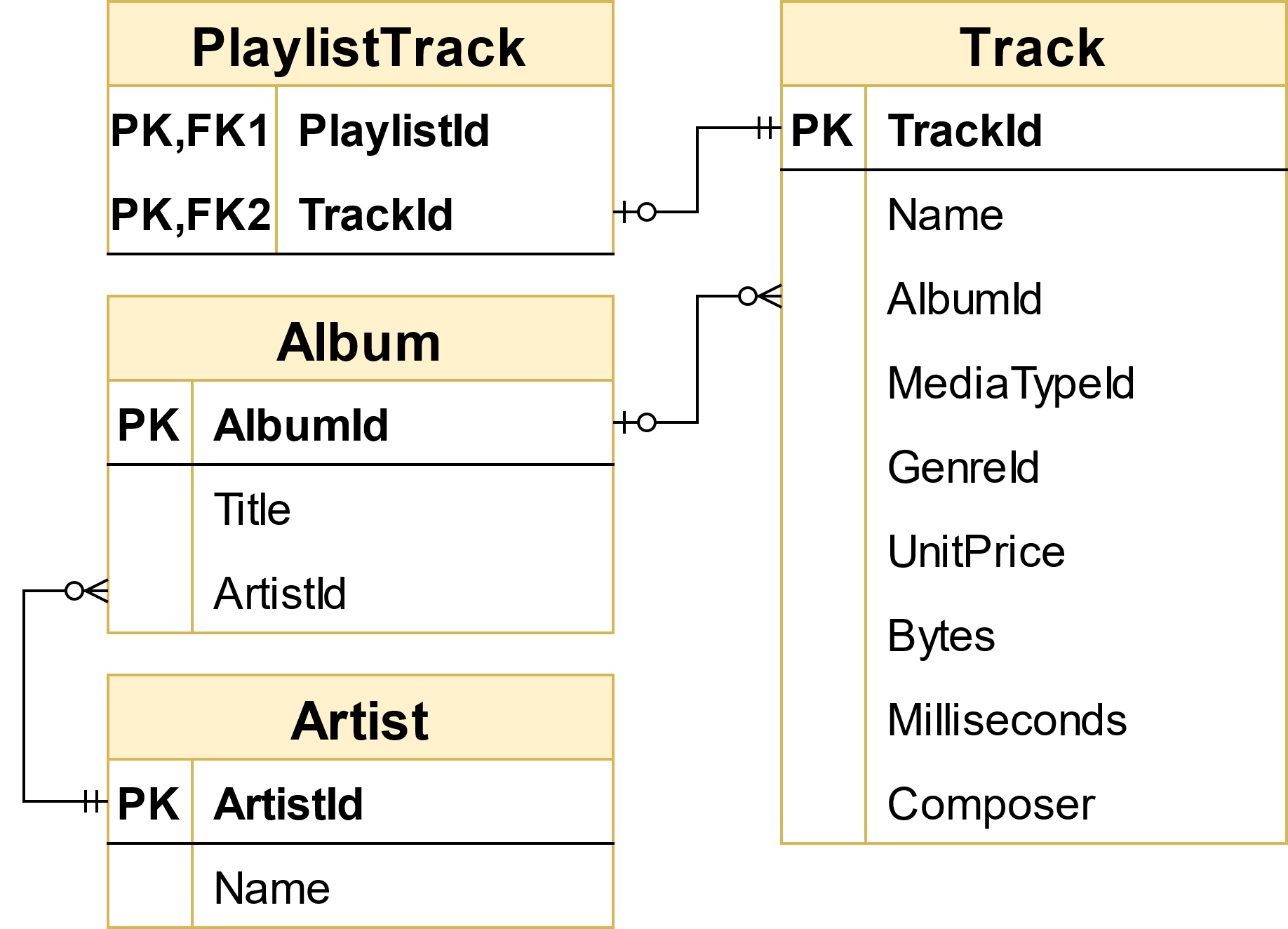
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| whole\_albums\_count | individual\_tracks\_count | total\_invoices\_count | whole\_albums\_percentage | individual\_tracks\_percentage |
| 0 | 1303 | 1303 | 0.0000 | 100.0000 |

- Comment: Store should change strategy to purchase only the most popular tracks from each album from record companies, instead of purchasing every track from an album to save money.

### 5. Which artist is used in the most playlists?

- To answer this question, we need a table has information about each track in which playlist? and who played it?

- We need 4 tables: Track, Album, PlaylistTrack and Artist to find out that:



SELECT ar.Name, COUNT(DISTINCT pt.playlistid) AS Playlist\_count

FROM playlisttrack pt

JOIN track t ON pt.trackid = t.trackid

JOIN album al ON t.albumid = al.albumid

JOIN artist ar ON al.artistid = ar.artistid

GROUP BY ar.Name

ORDER BY Playlist\_count DESC

LIMIT 1;

- We will get this result:

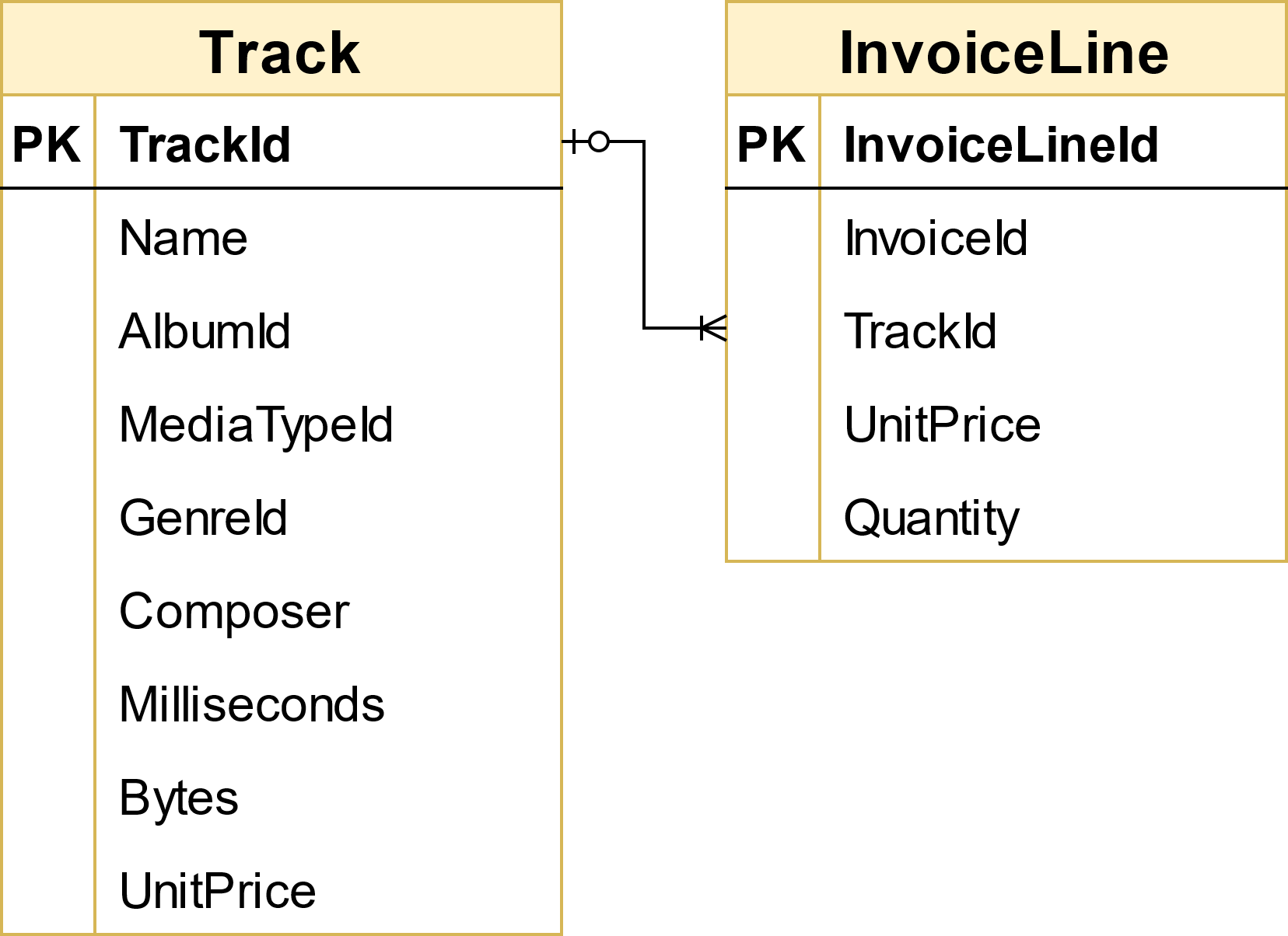
|  |  |
| --- | --- |
| Name | Playlist\_count |
| Eugene Ormandy | 7 |

Eugene Ormany was a Hungarian-born American conductor and violinist, best known for his association with the Philadelphia Orchestra, as its music director

### 6. How many tracks have been purchased vs not purchased?

- Firstly, we need to check whether a track appears in any invoice.

- To do this, we need 2 tables: Track and InvoiceLine



- We use this query to find out:

WITH Sort\_purchased AS

(SELECT TrackId, Name,

(CASE

WHEN EXISTS (SELECT trackid

FROM invoiceline il

WHERE t.trackid = il.trackid)

THEN 'Purchased'

ELSE 'Not purchased'

END) AS Purchased\_or\_not

FROM track t

GROUP BY trackid)

SELECT Purchased\_or\_not,

COUNT(trackid) AS Number\_of\_tracks,

ROUND(100\*COUNT(trackid)/(SELECT COUNT(\*) FROM track),2) AS Percent

FROM Sort\_purchased

GROUP BY Purchased\_or\_not;

- We got this result:

|  |  |  |
| --- | --- | --- |
| Purchased\_or\_not | Number\_of\_tracks | Percentage |
| Purchased | 1984 | 56.64 |
| Not purchased | 1519 | 43.36 |

- Comment: In total 3503 tracks in track table:

+ 1984 tracks have been purchased

+ 1519 tracks have never been bought by anyone

### 7. Is the range of tracks in the store reflective of their sales popularity?

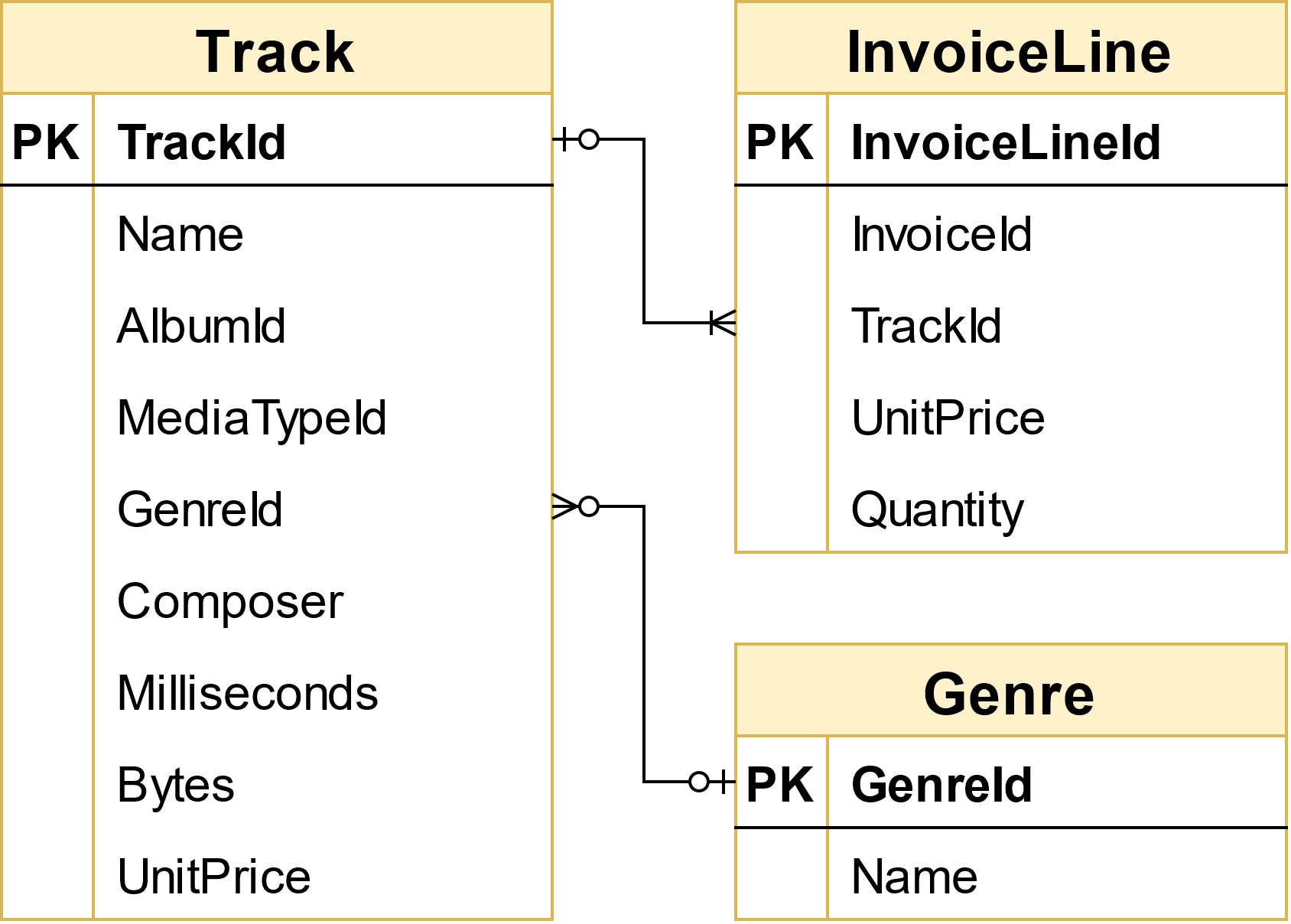
- To answer this question, we will choose Genre to represent range of tracks.

- Find out:

* How many tracks are sold for each genre?
* How many tracks in each genre?

- Compare 2 values to check whether tracks in the store reflective of their sales popularity

- We will need 3 tables: track, invoiceline and genre



- We use this query:

WITH

-- CTE 1

Total\_tracks\_instore AS (

SELECT g.GenreId, g.Name,

COUNT(t.trackid) AS Total\_tracks\_instore

FROM genre g

JOIN track t ON g.GenreId = t.GenreId

GROUP BY GenreId),

-- CTE 2

Total\_tracks\_sold AS (

SELECT g.GenreId, g.Name,

COUNT(DISTINCT il.trackid) AS Total\_tracks\_sold

FROM genre g

LEFT JOIN track t ON g.GenreId = t.GenreId

JOIN invoiceline il ON t.TrackId = il.TrackId

GROUP BY GenreId)

SELECT i.GenreId, i.Name,

i.Total\_tracks\_instore,

(CASE

WHEN s.Total\_tracks\_sold > 0 THEN s.Total\_tracks\_sold

ELSE 0

END) AS Total\_tracks\_sold,

(CASE

WHEN s.Total\_tracks\_sold != 0 THEN ROUND(Total\_tracks\_sold /i.Total\_tracks\_instore \* 100,2)

ELSE 0 / i.Total\_tracks\_instore \* 100

END) AS Proportion\_tracks\_sold

FROM Total\_tracks\_instore i

LEFT JOIN Total\_tracks\_sold s ON i.GenreId = s.GenreId

GROUP BY i.GenreId

- We got this result

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| GenreId | Name | Total\_track\_instore | Total\_tracks\_sold | Proportion\_tracks\_sold |
| 1 | Rock | 1297 | 745 | 57.4402 |
| 7 | Latin | 579 | 340 | 58.7219 |
| 3 | Metal | 374 | 231 | 61.7647 |
| 4 | Alternative & Punk | 332 | 203 | 61.1446 |
| 2 | Jazz | 130 | 68 | 52.3077 |
| 6 | Blues | 81 | 53 | 65.4321 |
| 19 | TV Shows | 93 | 43 | 46.2366 |
| 14 | R&B/Soul | 61 | 37 | 60.6557 |
| 24 | Classical | 74 | 36 | 48.6486 |
| 8 | Reggae | 58 | 28 | 48.2759 |
| 21 | Drama | 64 | 27 | 42.1875 |
| 9 | Pop | 48 | 26 | 54.1667 |
| 20 | Sci Fi & Fantasy | 26 | 20 | 76.9231 |
| 10 | Soundtrack | 43 | 19 | 44.1860 |
| 17 | Hip Hop/Rap | 35 | 15 | 42.8571 |
| 11 | Bossa Nova | 15 | 14 | 93.3333 |
| 23 | Alternative | 40 | 14 | 35.0000 |
| 16 | World | 28 | 13 | 46.4286 |
| 13 | Heavy Metal | 28 | 12 | 42.8571 |
| 15 | Electronica/Dance | 30 | 11 | 36.6667 |
| 12 | Easy Listening | 24 | 10 | 41.6667 |
| 22 | Comedy | 17 | 8 | 47.0588 |
| 5 | Rock And Roll | 12 | 6 | 50.0000 |
| 18 | Science Fiction | 13 | 5 | 38.4615 |
| 25 | Opera | 1 | 0 | 0.0000 |

- We can see percentage of sold tracks by genres are around 50%

- Only 2 cases in genre Bossa Nova (15) and SciFi & Fantasy (26) has proportion is over 70%, however, their number of total tracks is not much.

- Especial, nobody buy Opera track – and it is only has 1 track.

🡪 Range of tracks in store is not reflective sales popularity

### 8. Do protected vs. non-protected media types influence popularity?

- MediaType table has 5 types. Use this query to see:

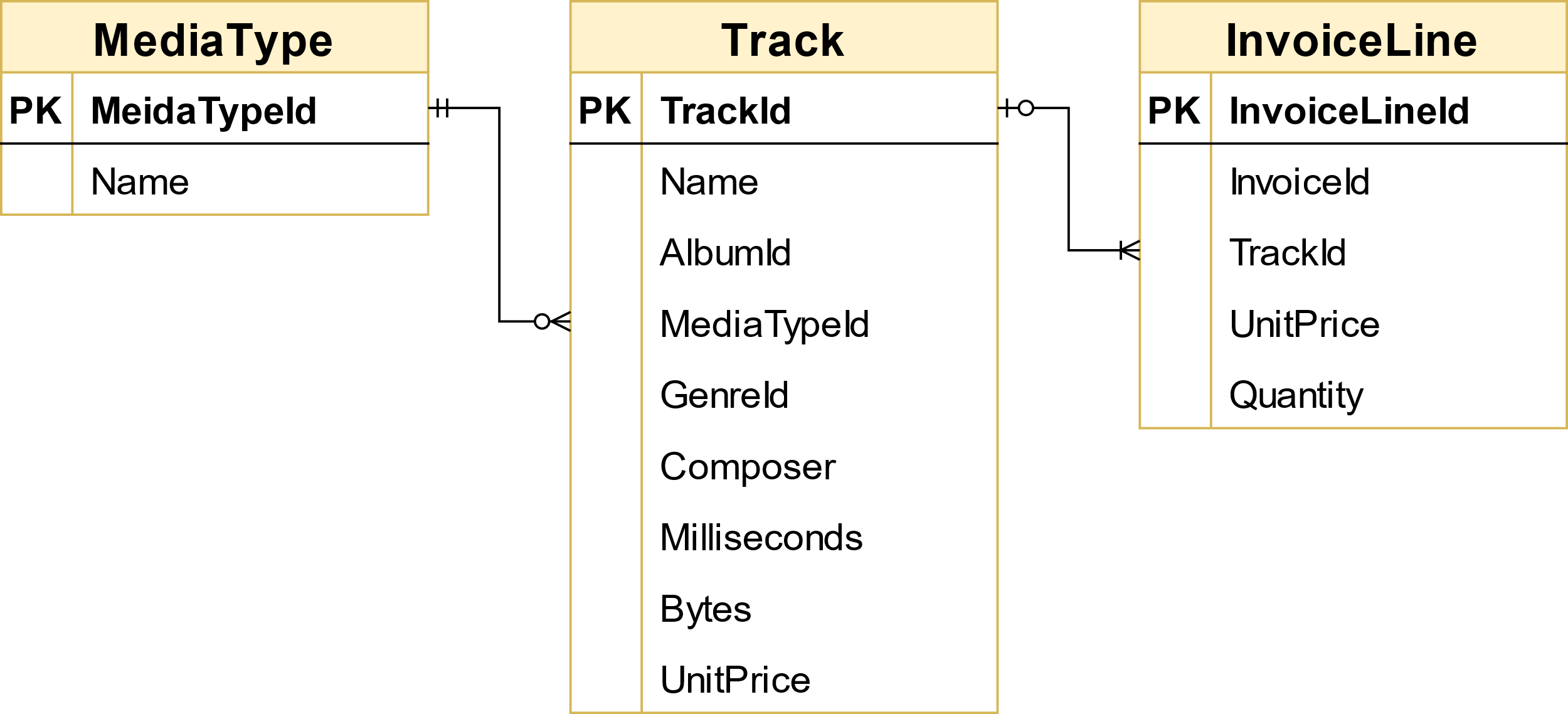
SELECT \* FROM chinook.mediatype;

|  |  |
| --- | --- |
| MediaTypeId | Name |
| 1 | MPEG audio file |
| 2 | Protected AAC audio file |
| 3 | Protected MPEG-4 video file |
| 4 | Purchased AAC audio file |
| 5 | AAC audio file |

- We will separate into 2 types:

* If Name has “Protected”, it means protected media type
* If Name don’t have “Protected”, it means non-protected type

- We will use 3 tables: MediaType, Track, InvoiceLine



- We use this query to find out:

SELECT(CASE

WHEN mt.Name LIKE '%protect%' THEN 'Protected'

ELSE 'Not protected'

END) AS Mediatype\_sort,

COUNT(il.trackid) AS Total\_tracks\_sold,

ROUND(COUNT(il.trackid) / (SELECT COUNT(\*)

FROM invoiceline) \* 100, 2) AS Sold\_tracks\_share

FROM mediatype mt

LEFT JOIN track t ON mt.MediaTypeId = t.MediaTypeId

LEFT JOIN invoiceline il ON t.TrackId = il.TrackId

GROUP BY Mediatype\_sort

ORDER BY Sold\_tracks\_share;

- We got this result:

|  |  |  |
| --- | --- | --- |
| Mediatype\_sort | Total\_tracks\_sold | Sold\_tracks\_share |
| Protected | 257 | 11.47 |
| Not protected | 1983 | 88.53 |

- It is proved that customers tend not to care about whether their track’s format are protected or not. More than 4/5 clients still purchase not protected ones.