Database Implementation

Design Report

Mountither Al Rashid 102486181

Contents

[Overview 3](#_Toc55312276)

[About the database 3](#_Toc55312277)

[Main Use Cases 4](#_Toc55312278)

[Script Used for Design 5](#_Toc55312279)

[Creation 5](#_Toc55312280)

[Data Types 8](#_Toc55312281)

[Information about the Music Store 9](#_Toc55312282)

[Customers 9](#_Toc55312283)

[Album 10](#_Toc55312284)

[Track 11](#_Toc55312285)

[Artist 12](#_Toc55312286)

[Orders 13](#_Toc55312288)

[Use Case Queries 14](#_Toc55312289)

[Alternative Storage of the Data 17](#_Toc55312290)

[Conclusion 17](#_Toc55312291)

[References 18](#_Toc55312292)

# Overview

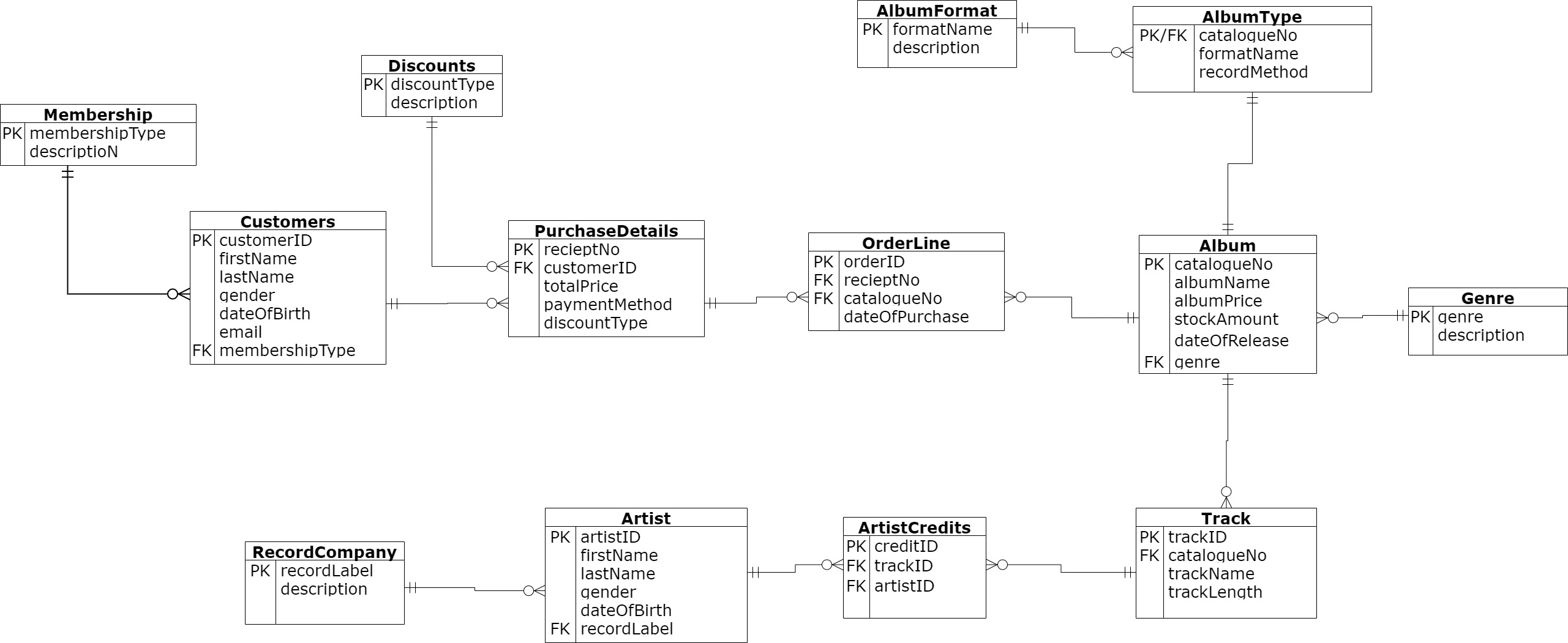


Figure 1 - UML diagram representing the entities and relations in the Music Database.

## About the database

A music store has many stakeholders involved. The processes of purchasing a physical album, the characteristics of that album and the industry revolving around the music are recorded. This music store records the customers details and allocates a membership status, while discounts are applied to some purchases depending on the circumstance. Each customer that purchases an album receives a receipt number that categorises the payment method, discount type (if applicable), the total price and the corresponding customer id. The aim of the orderline entity is to unify the receipt number and each album catalogue id through the unique attribute, order id. The catalogue id uniquely identifies an album. Each order id has a date of purchase. The album entity consists of a name, retail price, stock amount in inventory, date of release and the main genre. Each album contains many tracks, each track includes a name and length (in minutes). The artist is connected to the track through the artist credits entity. The artist credits entity includes the artist credited to the track with a unique id (creditID). The purpose for this junction entity is to resolve the many feature (artist) that occur on a track. The artist is also associated with a record company. For further observations refer to Figure 1 above. This information is recorded and stored for many use cases, discussed in the following section.

## Main Use Cases

The music store would find the most value in the customer information stored. The various attributes involved for the customer needs to be maintained consistently within the system. This eases retrieval and efficiency for the employees. A Relational Database is fitting in the situation since the structure is implemented with logical and the relations are broken up into relevant factions.

From a financial perspective, the music store can compute the revenue relative to the customer, the cyclical nature of business and the attributes related to the music. Such information is valuable to the business. Aggregations or other deductions can improve timing of promotions. The effectiveness of discount campaigns and membership types can be analysed easily in this Relational database.

The demographics are also analysed in this context. The type of album can contribute to the typical customers preference. This also applies to the genre and the artist. The retrieved information conveys a resolve to a query raised by the business.

# Script Used for Design

Each ‘create table’ statement is executed individually in a sequential manner.

## Creation

create table Membership(

    membershipType varchar(20) not null,

    description varchar(100),

    PRIMARY KEY (membershipType)

);

create table Customers(

    customerID int(10) not null AUTO\_INCREMENT,

    firstName varchar(30),

    lastName varchar(30),

    gender varchar(6) not null check(gender in ('Male', 'Female', 'Other')),

    dateOfBirth date,

    email varchar(30),

    membershipType varchar(20),

    PRIMARY key (customerID),

    FOREIGN key (membershipType) REFERENCES Membership (membershipType)

);

create table Discounts(

    discountType varchar(20) not null,

    description varchar(100),

    PRIMARY KEY (discountType)

);

create table PurchaseDetails(

    recieptNo int(10) not null AUTO\_INCREMENT,

    customerID int(10) not null,

    totalPrice decimal(10,2) not null,

    paymentMethod varchar(20),

    discountType varchar(20),

    PRIMARY KEY (recieptNo),

    FOREIGN KEY (discountType) REFERENCES Discounts (discountType),

    FOREIGN KEY (customerID) REFERENCES Customers (customerID)

)

-- catalogue number refers to the album id. Similar to barcode

create table OrderLine(

    orderID int(10) not null AUTO\_INCREMENT,

    recieptNo int(10) not null,

    catalogueNo int(10) not null,

    dateOfPurchase datetime,

    PRIMARY KEY (orderID),

    FOREIGN KEY (recieptNo) REFERENCES PurchaseDetails (recieptNo)

)

create table Album (

    catalogueNo int(10) not null AUTO\_INCREMENT,

    albumName varchar(40),

    albumPrice  decimal(10,2),

    stockAmount int(10),

    dateOfRelease date,

    genre varchar(10),

    PRIMARY KEY (catalogueNo)

);

ALTER TABLE OrderLine ADD FOREIGN KEY (catalogueNo) REFERENCES Album(catalogueNo);

create table Genre (

    genre varchar(10) not null,

    description varchar(100),

    PRIMARY KEY (genre)

);

ALTER TABLE Album ADD FOREIGN KEY (genre) REFERENCES Album(genre);

create table AlbumType(

    catalogueNo int(10) not null,

    formatName varchar(20),

    recordMethod varchar(20),

    PRIMARY KEY (catalogueNo),

    FOREIGN KEY (catalogueNo) REFERENCES Album(catalogueNo)

);

create table AlbumFormat (

    formatName varchar(20) not null,

    description varchar(100),

    PRIMARY KEY (formatName)

);

ALTER TABLE AlbumType ADD FOREIGN KEY (formatName) REFERENCES AlbumFormat(formatName);

create table Track (

    trackID int(10) not null AUTO\_INCREMENT,

    catalogueNo int(10),

    trackName VARCHAR(20),

    trackLength decimal,

    PRIMARY KEY (trackID),

    FOREIGN KEY (catalogueNo) REFERENCES Album(catalogueNo)

);

create table ArtistCredits (

    creditID int(10) not null AUTO\_INCREMENT,

    trackID int(10) not null,

    artistID int(10) not null,

    PRIMARY KEY (creditID),

    FOREIGN KEY (trackID) REFERENCES Track (trackID)

);

create table Artist(

    artistID int(10) not null AUTO\_INCREMENT,

    firstName varchar(50),

    lastName varchar(50),

    gender varchar(6) not null check(gender in ('Male', 'Female', 'Other')),

    dateOfBirth date,

    recordLabel varchar(30),

    PRIMARY KEY (artistID)

);

ALTER TABLE ArtistCredits ADD FOREIGN KEY (artistID) REFERENCES Artist(artistID);

create table RecordCompany (

    recordLabel varchar(30) not null,

    description varchar(100),

    PRIMARY KEY (recordLabel)

);

ALTER TABLE Artist ADD FOREIGN KEY (recordLabel) REFERENCES RecordCompany(recordLabel);

# Data Types

Various data types are used in this database.

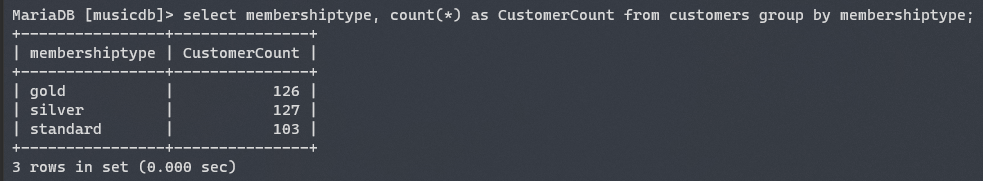
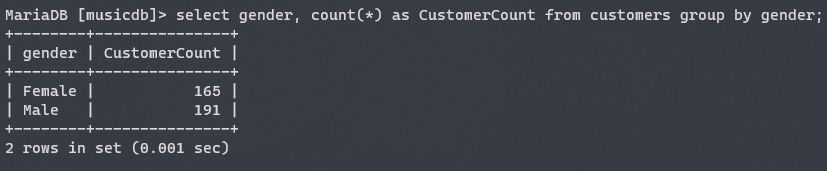
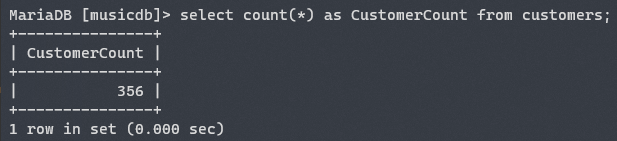
* Identification based attributes are integers that are auto incremented in the time being. This can change if actual identification is introduced. The auto increment function is practically row number with a ‘not null’.
* Checks are used for gender to constraint the selections.
* Currency based attributes are expressed through the decimal data type with 2 decimal points.
* Descriptions are attributed with varchar and size of 100
* Date of birth for the human entities are given the date datatype with the format ‘yyyy-mm-dd’
* Other dates such as purchase dates and release dates include the time using the ‘datetime’ datatype.
* Not nulls are utilised for all unique identification and necessary attributes that are required to be recorded for database integrity.
* Nulls are observed in the discount type attribute for the purchase details entity. Not all purchases have a discount.

# Information about the Music Store

The following section provides descriptive information of the music store via MySQL scripts.

## Customers

The count of customers, the proportion in gender and membership type between them is shown in the below figure. Also, the occurrences of email Domain names from their email is computed.



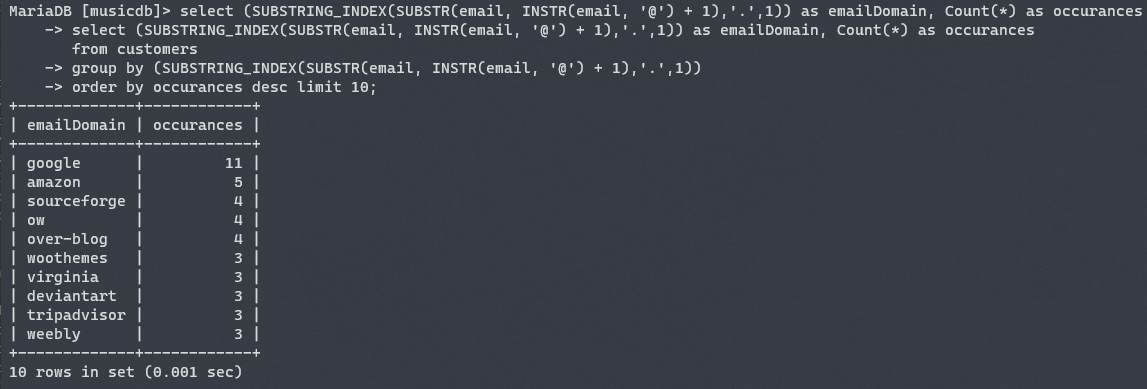
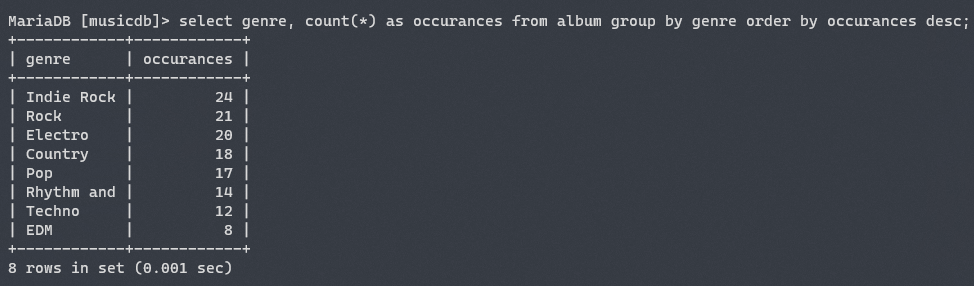
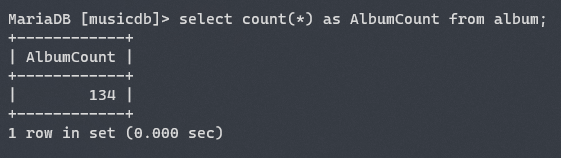
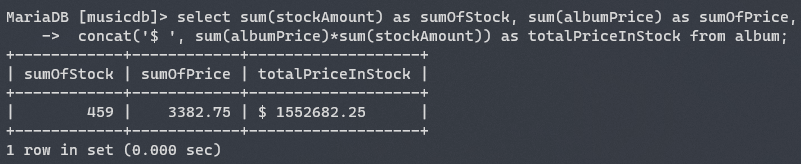


Figure 2 – Information extracted directly from the customer entity

## Album

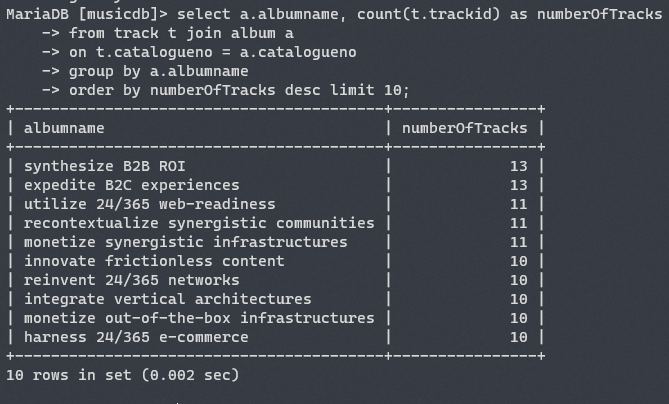
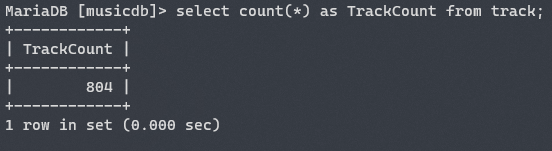
The count of albums is shown in figure 3. Also, the number of genres appearing in albums and the total retail price of all albums in the current inventory are shown below.

Figure 3 - Information about the albums

## Track

The figure below shows the count of tracks in database, number of tracks in the top 10 albums and the average track length is computed in minutes.



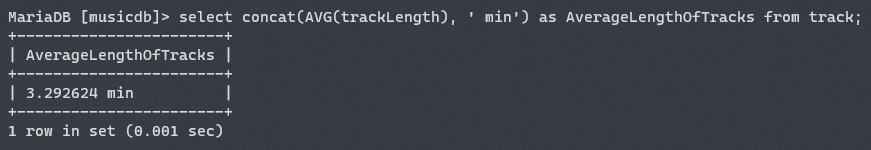
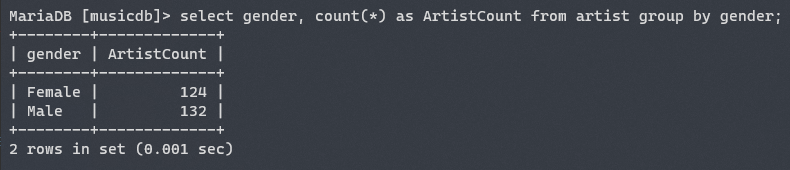
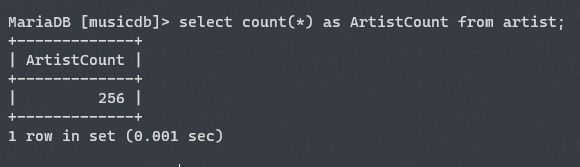


Figure 4 - Information about the tracks in database

## Artist

The number of artists credited in the albums sold in the music store, the number of artists associated with a record label, and the gender proportion are shown in Figure 5.



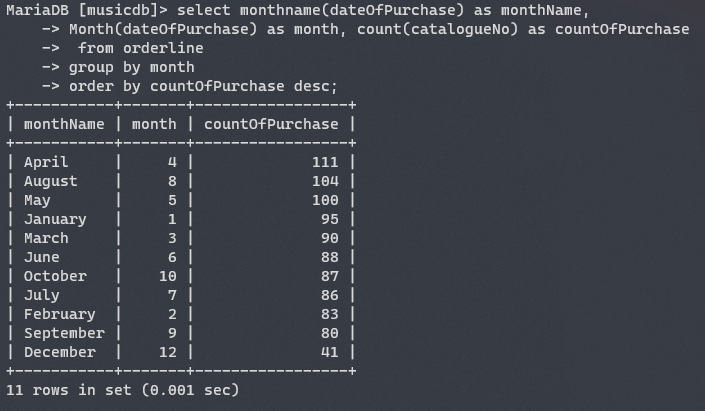
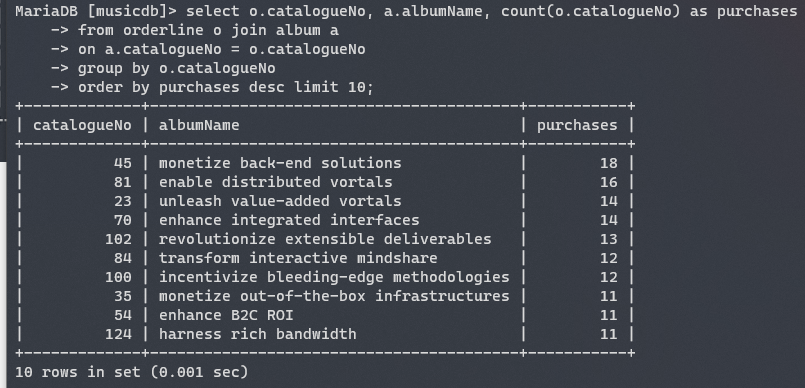
## 

Figure 5 - information about the artist

## Orders

The following figure show the amount of purchases for the top 10 albums and their names (top query ) and the number of purchases for each month in the music store.

Figure 6 - queries concerning orders in music store



## Use Case Queries

The following are queries that include join statements. Descriptions are given in the figure caption. Each query attempts to respond to the relevant matter.

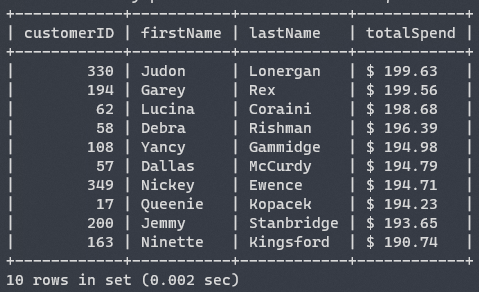
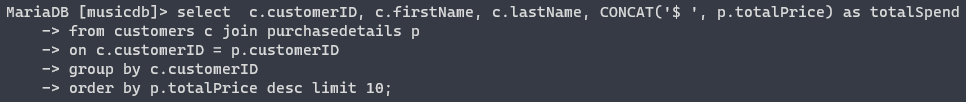
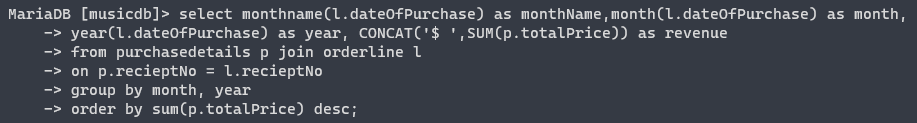


Figure 7 – This query seeks the top 10 customers that spent the most at the music store.



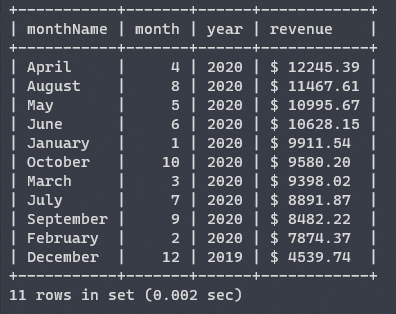
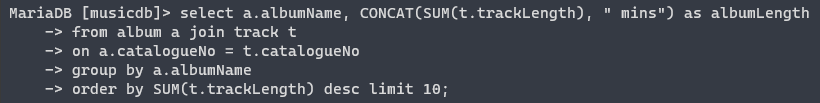


Figure 8 – This query retrieves the revenue earnings monthly at the music store.

Figure 9 - The top 10 albums with the longest duration of play time in minutes.



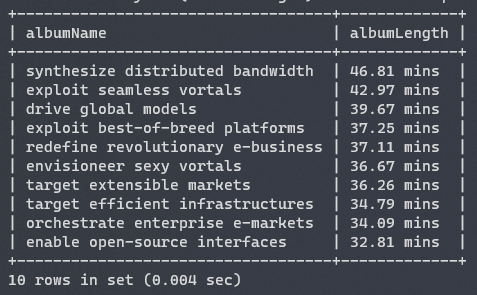
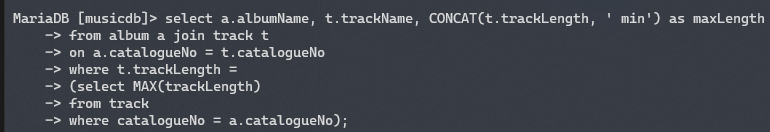


Figure 10 - Extraction of the longest song in each album.



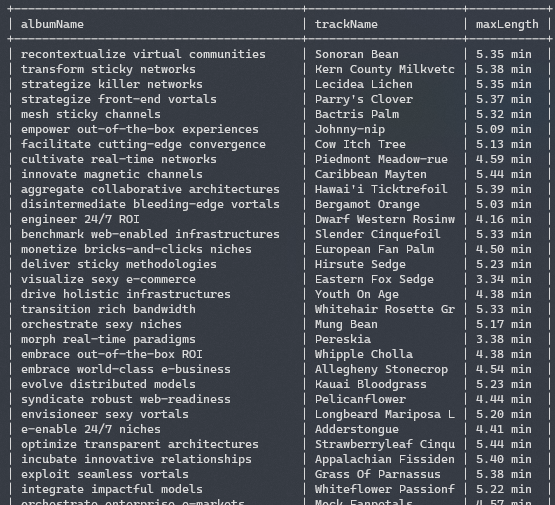
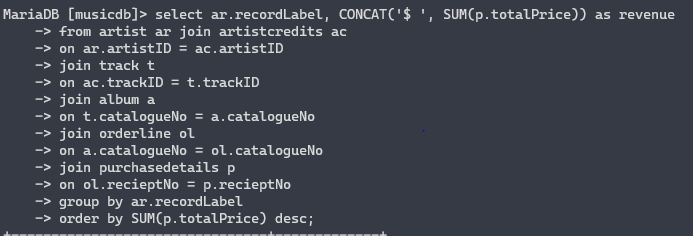
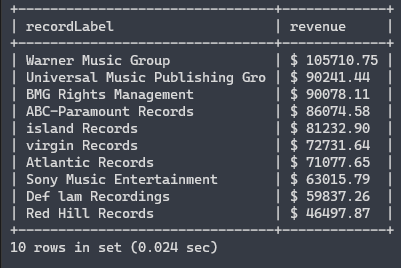


Figure 11 - Revenue gained from the music store for each record label.





# Alternative Storage of the Data

A NoSQL database can used to store the music database. The difference is that a schema will not be enforced. This is observed in the datatype, ‘null/not null’ and PK/FK allocation for each attribute. In NoSQL, this structure is not required. In the event of employing NoSQL database for the music store, it is suited that database operates on large volumes of unstructured data. Flexibility and leniency in the collection of data can improve elaborate analysis and fast retrieval of data can lead to efficient computation.

However, this music store collects the information from centralised system that records customer, order and album related data. It would be effective to use a Relational Database in this case. This can be attributed to the size of the business and maintenance of the inputted data. This type of business is rare because most music is internet-based, hence the data will not be in excess. The maturity of Relational Databases can impose strictness to the characteristics of the customer, orders, artists, albums and its features. This will develop effective and direct analysis of the permanent database.

# Conclusion

Throughout the design report, the design and use cases of the music database was discussed. Such use cases were elaborated through the implementation of SQL queries. The process and interaction of the entities within the music database was explored. In addition, the reasoning behind the selection of data types in the schemas was provided. An alternative method was proposed and the justification of selecting Relational database to convey the businesses data were also conveyed.

# References

Chackfield, M., 2014. *12 Album Tracks Released On 12 Different Formats.* [Online]   
Available at: https://www.shortlist.com/news/12-album-tracks-released-on-12-different-formats  
[Accessed 20 October 2020].

Gibbs, T., 2019. *Five Common Data Stores and When to Use Them.* [Online]   
Available at: https://shopify.engineering/five-common-data-stores-usage  
[Accessed 20 October 2020].

Mcdonald, H., 2019. *Catalog Numbers for CDs.* [Online]   
Available at: https://www.thebalancecareers.com/catalog-number-2460354  
[Accessed 20 October 2020].

Mockaroo, 2020. *Realistic Data Generator.* [Online]   
Available at: https://mockaroo.com/  
[Accessed 1 November 2020].

Statista, 2018. *Music album consumption U.S. 2018, by genre.* [Online]   
Available at: https://www.statista.com/statistics/310746/share-music-album-sales-us-genre/  
[Accessed 20 October 2020].