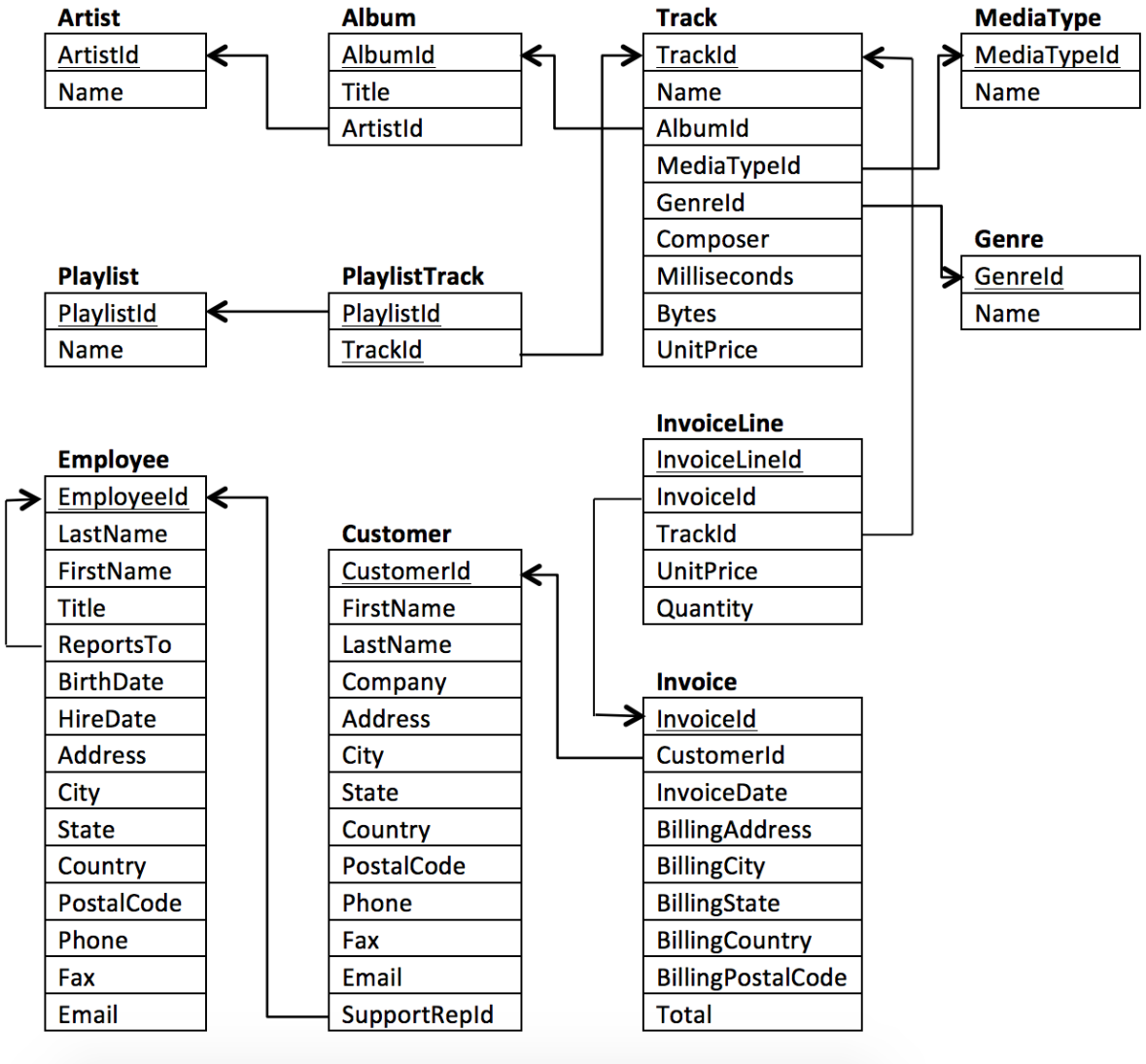
Music store database analysis using SQL

Gordon Ma

2024-01-15

Introduction

In this project, we use SQL comments to analyze the Chinook database, which is a public database containing the relevant information from a music store. The database can be accessed from the website. The following diagram displays the structure of this database.



There are 11 tables in the Chinook sample database:

- `employees` table stores employee data such as employee ID, last name, first name, etc. It also has a field named `ReportsTo` to specify who reports to whom.
- `customers` table stores customer data.
- `invoices` & `invoice_items` tables: these two tables store invoice data. The `invoices` table stores invoice header data and the `invoice_items` table stores the invoice line items data.
- `artists'` table stores artists' data. It is a simple table that contains only the artist's ID and name.
- `aalbum`table stores data about a list of tracks. Each album belongs to one artist. However, one artist may have multiple albums.
- `media_types` table stores media types such as MPEG audio and AAC audio files.
- `genres` table stores music types such as rock, jazz, metal, etc.
- `tracks` table stores the data of songs. Each track belongs to one album.
- `playlists` & `playlist_track` tables: `playlists` table store data about playlists. Each playlist contains a list of tracks. Each track may belong to multiple playlists. The `playlist_track` table is used to reflect this relationship.

```r
#We first load the required packages in R
library(RSQLite)
```

```
## Warning: package 'RSQLite' was built under R version 4.2.3
```

```r
#Connect to your SQLite database file and list all the tables
db_conn <- dbConnect(SQLite(), dbname = "chinook.db")
dbListTables(db_conn)
```

```
##  [1] "albums"          "artists"         "customers"       "employees"
##  [5] "genres"          "invoice_items"   "invoices"        "media_types"
##  [9] "playlist_track"  "playlists"       "sqlite_sequence" "sqlite_stat1"
## [13] "tracks"
```

```r
#Query to read one of the table
query <- "SELECT * FROM playlists;"
result <- dbSendQuery(db_conn, query)
dbFetch(result)
```

```
##    PlaylistId                     Name
## 1           1                    Music
## 2           2                   Movies
## 3           3                 TV Shows
## 4           4                Audiobooks
## 5           5                90's Music
## 6           6                Audiobooks
## 7           7                    Movies
## 8           8                    Music
## 9           9               Music Videos
## 10         10                 TV Shows
## 11         11            Brazilian Music
## 12         12                 Classical
## 13         13  Classical 101 - Deep Cuts
## 14         14 Classical 101 - Next Steps
## 15         15 Classical 101 - The Basics
## 16         16                    Grunge
## 17         17        Heavy Metal Classic
```

```
## 18         18                On-The-Go 1
```

As a music store manager, we would like to know which employees sold the most and made the highest profits.

```
query <- "SELECT m.FirstName ||' '|| m.LastName AS ManagerName, E.FirstName ||' '|| E.LastName AS Employ
  FROM employees AS E
  JOIN Customers AS C ON E.EmployeeId = C.SupportRepId
  JOIN invoices AS I ON I.CustomerId = C.CustomerId
  JOIN invoice_items AS II on I.InvoiceId = II.InvoiceId
  JOIN Employees m ON e.ReportsTo = m.EmployeeID
  GROUP BY E.EmployeeId
  ORDER BY profits DESC;
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##     ManagerName  EmployeeName               Title total_sold profits
## 1 Nancy Edwards  Jane Peacock Sales Support Agent        796 7427.06
## 2 Nancy Edwards Margaret Park Sales Support Agent        760 6931.40
## 3 Nancy Edwards Steve Johnson Sales Support Agent        684 6490.16
```

How many users per country? And how much did users spend on each country?

```
query <- "SELECT C.COUNTRY, COUNT(DISTINCT C.CUSTOMERID) AS total_user_count, SUM(I.total) AS total_pro
  FROM customers AS C
  JOIN invoices AS I
  GROUP BY 1
  ORDER BY 3 DESC;
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
(users <- dbFetch(result))
```

```
##           Country total_user_count total_profits
## 1             USA               13       30271.8
## 2          Canada                8       18628.8
## 3          France                5       11643.0
## 4          Brazil                5       11643.0
## 5         Germany                4        9314.4
## 6  United Kingdom                3        6985.8
## 7        Portugal                2        4657.2
## 8           India                2        4657.2
## 9  Czech Republic                2        4657.2
## 10         Sweden                1        2328.6
## 11          Spain                1        2328.6
## 12         Poland                1        2328.6
## 13         Norway                1        2328.6
## 14    Netherlands                1        2328.6
## 15          Italy                1        2328.6
```

```
## 16        Ireland                1        2328.6
## 17        Hungary                1        2328.6
## 18        Finland                1        2328.6
## 19        Denmark                1        2328.6
## 20          Chile                1        2328.6
## 21        Belgium                1        2328.6
## 22        Austria                1        2328.6
## 23      Australia                1        2328.6
## 24      Argentina                1        2328.6
```

Returns the list of customers ordered by name, whose sales representatives are in Canada.

```
query <- "SELECT customerid,
       firstname,
       lastname
  FROM customers
 WHERE supportrepid IN (
           SELECT employeeid
             FROM employees
            WHERE country = 'Canada'
            )
            ORDER BY firstname, lastname;
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##    CustomerId FirstName     LastName
## 1          32     Aaron     Mitchell
## 2          11 Alexandre        Rocha
## 3           7    Astrid       Gruber
## 4           4     Bjørn       Hansen
## 5          39   Camille      Bernard
## 6           8      Daan      Peeters
## 7          20       Dan       Miller
## 8          56     Diego    Gutiérrez
## 9          40 Dominique     Lefebvre
## 10         10   Eduardo      Martins
## 11         30    Edward      Francis
## 12         33     Ellie     Sullivan
## 13         52      Emma        Jones
## 14         50   Enrique        Muñoz
## 15         13  Fernanda        Ramos
## 16         16     Frank       Harris
## 17         24     Frank      Ralston
## 18          5 František  Wichterlová
## 19          3  François     Tremblay
## 20         37      Fynn   Zimmermann
## 21         36    Hannah    Schneider
## 22         22   Heather      Leacock
## 23          6    Helena         Holý
## 24         46      Hugh     O'Reilly
```

```
## 25         43  Isabelle      Mercier
## 26         17      Jack        Smith
## 27         15  Jennifer     Peterson
## 28         51    Joakim    Johansson
## 29         48  Johannes Van der Berg
## 30         23      John       Gordon
## 31         34      João    Fernandes
## 32         28     Julia      Barnett
## 33          9      Kara      Nielsen
## 34         21     Kathy        Chase
## 35         45  Ladislav       Kovács
## 36          2    Leonie       Köhler
## 37         47     Lucas      Mancini
## 38         57      Luis        Rojas
## 39          1      Luís    Gonçalves
## 40         35  Madalena      Sampaio
## 41         58     Manoj       Pareek
## 42         41      Marc       Dubois
## 43         14      Mark      Philips
## 44         55      Mark       Taylor
## 45         31    Martha         Silk
## 46         18  Michelle       Brooks
## 47         38    Niklas     Schröder
## 48         27   Patrick         Gray
## 49         53      Phil       Hughes
## 50         59      Puja   Srivastava
## 51         26   Richard   Cunningham
## 52         29    Robert        Brown
## 53         12   Roberto      Almeida
## 54         49 Stanisław       Wójcik
## 55         54     Steve       Murray
## 56         44     Terhi   Hämäläinen
## 57         19       Tim        Goyer
## 58         25    Victor      Stevens
## 59         42     Wyatt       Girard
```

Which city has the best customers? We would like to throw a promotional Music Festival in the city that made the most money.

```
query <- "SELECT BILLINGCITY AS City,
  SUM(TOTAL) AS profits
  FROM invoices
  GROUP BY 1
  ORDER BY 2 DESC;
          "
result <- dbSendQuery(db_conn, query)
```

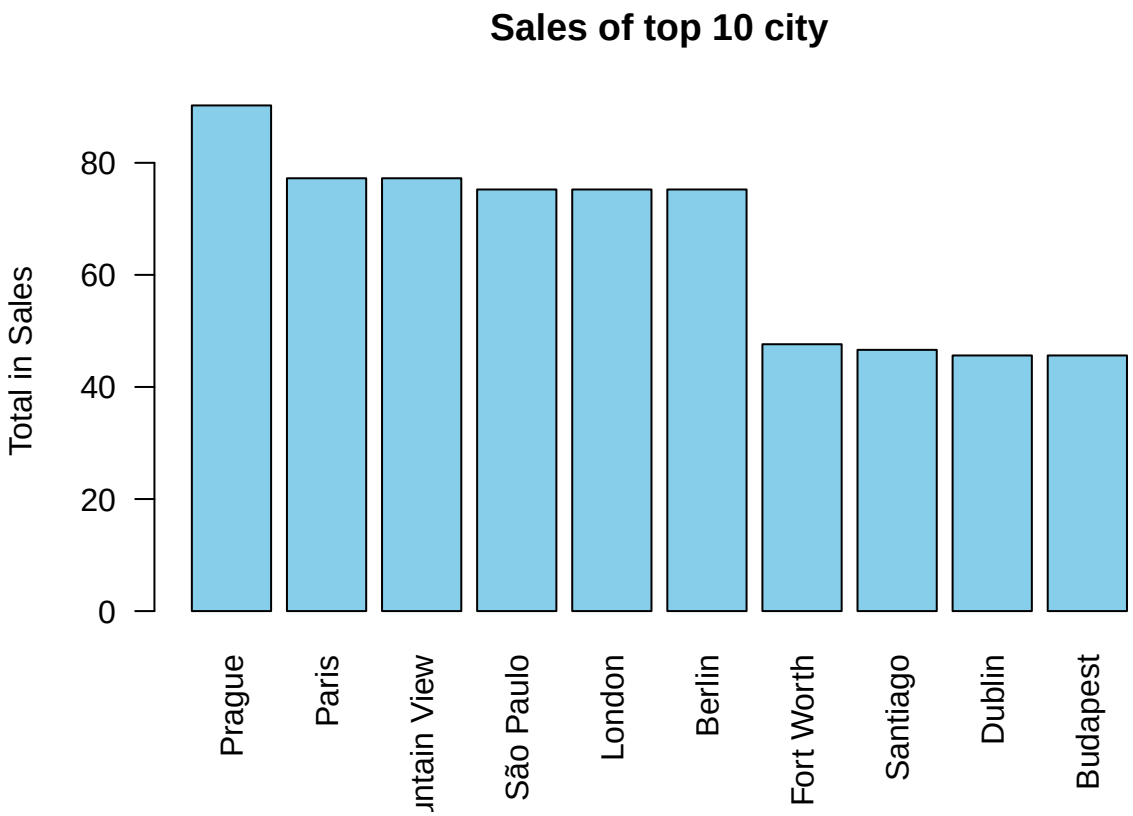
```
## Warning: Closing open result set, pending rows
```

```
(city_sales <- dbFetch(result))
```

```
##                   City profits
## 1               Prague   90.24
## 2                Paris   77.24
```

```
## 3        Mountain View   77.24
## 4            São Paulo   75.24
## 5               London   75.24
## 6               Berlin   75.24
## 7           Fort Worth   47.62
## 8             Santiago   46.62
## 9               Dublin   45.62
## 10             Budapest   45.62
## 11      Salt Lake City   43.62
## 12           Frankfurt   43.62
## 13             Chicago   43.62
## 14              Vienne   42.62
## 15             Madison   42.62
## 16            Helsinki   41.62
## 17               Dijon   40.62
## 18           Amsterdam   40.62
## 19 São José dos Campos   39.62
## 20             Redmond   39.62
## 21                Oslo   39.62
## 22             Orlando   39.62
## 23            Montréal   39.62
## 24              Lisbon   39.62
## 25            Bordeaux   39.62
## 26           Vancouver   38.62
## 27           Stockholm   38.62
## 28               Delhi   38.62
## 29           Cupertino   38.62
## 30         Yellowknife   37.62
## 31            Winnipeg   37.62
## 32              Warsaw   37.62
## 33              Tucson   37.62
## 34             Toronto   37.62
## 35           Stuttgart   37.62
## 36              Sidney   37.62
## 37                Rome   37.62
## 38      Rio de Janeiro   37.62
## 39                Reno   37.62
## 40               Porto   37.62
## 41              Ottawa   37.62
## 42            New York   37.62
## 43              Madrid   37.62
## 44                Lyon   37.62
## 45             Halifax   37.62
## 46            Edmonton   37.62
## 47          Edinburgh    37.62
## 48          Copenhagen   37.62
## 49        Buenos Aires   37.62
## 50            Brussels   37.62
## 51            Brasília   37.62
## 52              Boston   37.62
## 53           Bangalore   36.64
```

```
#Visualization of each city's profits
barplot(city_sales$profits[1:10], names.arg = city_sales$City[1:10], col = "skyblue", main = "Sales of
```



We decide to host a Music Festival in Prague. We would like to find out what genres of music do people like.

```
query <- "SELECT G.Name, SUM(TOTAL) as profits
  FROM invoices AS I
  JOIN invoice_items AS II ON I.InvoiceId = II.InvoiceId
  JOIN tracks AS T ON T.TrackId = II.TrackId
  JOIN genres AS G ON T.GenreId = G.GenreId
  WHERE I.billingCity = 'Prague'
  GROUP BY G.Name
  ORDER BY profits DESC;
          "
result <- dbSendQuery(db_conn, query)
```

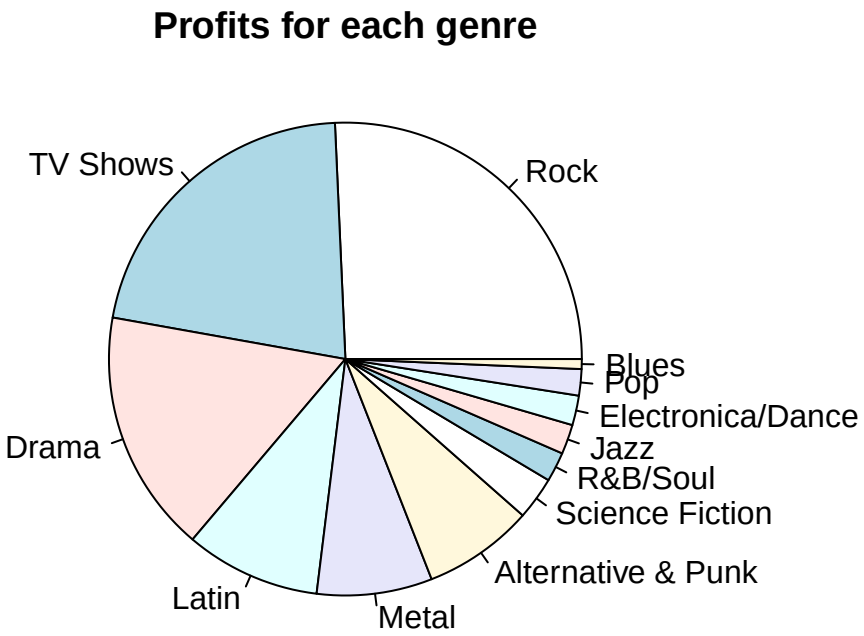
```
## Warning: Closing open result set, pending rows
```

```
(genres_profits <- dbFetch(result) )
```

```
##                  Name profits
## 1                Rock  226.02
## 2            TV Shows  188.88
## 3               Drama  146.16
## 4               Latin   81.27
## 5               Metal   69.36
## 6  Alternative & Punk   66.45
## 7     Science Fiction   25.86
```

```
## 8            R&B/Soul   17.82
## 9                Jazz   17.82
## 10  Electronica/Dance   17.82
## 11                Pop   15.84
## 12              Blues    5.94
```

```
#Create a pie chart to visualize the profits of each generes
pie(genres_profits$profits, genres_profits$Name, radius = 1.0, main = "Profits for each genre")
```



So Rock music is the most popular in Prague. We would like to invite artists who is known for Rock music. Return a table of rock artists with at least 20 tracks of rock music, and ordered them by total profits made.

```
query <- "SELECT AR.Name, COUNT(T.Name) AS Num_of_RockMusic, SUM(I.Total) AS Profits_made
  FROM tracks AS T
  JOIN genres AS G ON T.GENREID = G.GENREID
  JOIN albums AS AL ON AL.ALBUMID = T.ALBUMID
  JOIN artists AS AR ON AR.ARTISTID = AL.ARTISTID
  JOIN invoice_items AS II ON II.TrackId = T.TrackId
  JOIN invoices AS I ON I.InvoiceId = II.InvoiceId
  WHERE G.Name = 'Rock'
  GROUP BY AR.Name HAVING Num_of_RockMusic >= 20
  ORDER BY Profits_made DESC
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##                            Name Num_of_RockMusic Profits_made
## 1                            U2               91       773.65
## 2                  Led Zeppelin               87       620.73
## 3                   Deep Purple               44       550.44
## 4                   Iron Maiden               54       473.22
## 5                     Van Halen               29       336.82
## 6                     Pearl Jam               26       335.61
## 7                         Queen               37       256.41
## 8  Creedence Clearwater Revival               37       215.82
## 9                          Kiss               31       211.86
## 10               Guns N' Roses               26       142.56
```

We will invite the band U2 since they had the highest number of rock music and made the most profits. Let us find out which are their biggest fan and also invite them.

```
query <- "SELECT C.firstname, C.lastname,SUM(II.Quantity * II.UnitPrice) AS money_spend
  FROM customers AS C
  JOIN invoices AS I ON C.CustomerId = I.CustomerId
  JOIN invoice_items AS II ON I.InvoiceId = II.InvoiceId
  JOIN tracks AS T ON T.TrackId = II.TrackId
  JOIN albums AS AL ON AL.AlbumId = T.AlbumId
  JOIN artists AS AR ON AR.artistid = AL.artistid
  WHERE AR.Name = 'U2'
  GROUP BY C.customerId
  ORDER BY money_spend DESC
  LIMIT 10;
           "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##    FirstName    LastName money_spend
## 1     Astrid      Gruber        8.91
## 2   Madalena     Sampaio        8.91
## 3    Richard  Cunningham        7.92
## 4    Eduardo     Martins        5.94
## 5     Robert       Brown        5.94
## 6       Hugh    O'Reilly        5.94
## 7  Stanisław      Wójcik        5.94
## 8     Helena        Holý        4.95
## 9       Mark      Taylor        4.95
## 10 František Wichterlová        3.96
```

Let us find out which album of U2 sold the most?

```
query <- "SELECT AL.title, SUM(Quantity) as total_sold
          FROM albums AS AL
          JOIN tracks AS T ON T.AlbumId = AL.AlbumId
```

```
          JOIN invoice_items as II ON II.trackid = T.trackid
          WHERE artistid IN (
          SELECT artistid
          FROM artists
          WHERE name = 'U2'
          )
          GROUP BY al.title
          ORDER BY total_sold DESC
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##                                                                Title total_sold
## 1                                                    Rattle And Hum         17
## 2  Instant Karma: The Amnesty International Campaign to Save Darfur         16
## 3                                                              War         11
## 4                                          The Best Of 1980-1990         11
## 5                                              B-Sides 1980-1990         11
## 6                                                              Pop         10
## 7                                  How To Dismantle An Atomic Bomb         10
## 8                                                         Zooropa          9
## 9                                  All That You Can't Leave Behind          6
## 10                                                    Achtung Baby          6
```

To find out what are other songs do U2's fans like to listen to, we want to find which playlist contain U2's tracks?

```
query <- "SELECT P.Name AS name_of_the_playlist, COUNT(DISTINCT T.Name) as total_U2_song
          FROM playlists as P
          JOIN playlist_track as PT ON P.PlaylistId = PT.PlaylistId
          JOIN tracks AS T ON T.trackid = PT.trackid
          WHERE AlbumId IN (
            SELECT AlbumId
            FROM artists AS AR
            JOIN albums AS AL ON AR.artistid = AL.artistid
            WHERE AR.name = 'U2'
          )
          GROUP BY P.PlaylistId
          ORDER BY total_U2_song DESC;
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##   name_of_the_playlist total_U2_song
## 1                Music           124
## 2                Music           124
## 3           90's Music            62
```

```
#Explore the 90's Music playlist
query <- "SELECT T.*
```

```
          FROM tracks AS T
          JOIN playlist_track as PT ON T.trackid = PT.trackid
          JOIN playlists as PL on PL.playlistid = PT.playlistid
          WHERE PL.name = '90's Music'
          LIMIT 10;
          "
result <- dbSendQuery(db_conn, query)
```

```
## Warning: Closing open result set, pending rows
```

```
dbFetch(result)
```

```
##    TrackId                     Name AlbumId MediaTypeId GenreId
## 1        3          Fast As a Shark       3           2       1
## 2        4        Restless and Wild       3           2       1
## 3        5     Princess of the Dawn       3           2       1
## 4       23            Walk On Water       5           1       1
## 5       24      Love In An Elevator       5           1       1
## 6       25                 Rag Doll       5           1       1
## 7       26            What It Takes       5           1       1
## 8       27 Dude (Looks Like A Lady)       5           1       1
## 9       28        Janie's Got A Gun       5           1       1
## 10      29                   Cryin'       5           1       1
##                                                                Composer
## 1                     F. Baltes, S. Kaufman, U. Dirkscneider & W. Hoffman
## 2  F. Baltes, R.A. Smith-Diesel, S. Kaufman, U. Dirkscneider & W. Hoffman
## 3                                                 Deaffy & R.A. Smith-Diesel
## 4                        Steven Tyler, Joe Perry, Jack Blades, Tommy Shaw
## 5                                                   Steven Tyler, Joe Perry
## 6                     Steven Tyler, Joe Perry, Jim Vallance, Holly Knight
## 7                                  Steven Tyler, Joe Perry, Desmond Child
## 8                                  Steven Tyler, Joe Perry, Desmond Child
## 9                                               Steven Tyler, Tom Hamilton
## 10                                 Steven Tyler, Joe Perry, Taylor Rhodes
##    Milliseconds    Bytes UnitPrice
## 1        230619  3990994      0.99
## 2        252051  4331779      0.99
## 3        375418  6290521      0.99
## 4        295680  9719579      0.99
## 5        321828 10552051      0.99
## 6        264698  8675345      0.99
## 7        310622 10144730      0.99
## 8        264855  8679940      0.99
## 9        330736 10869391      0.99
## 10       309263 10056995      0.99
```