Assignment 1: Data Extraction

Group 6: Juan Moreno Díez (1840886), Leticia Marcal Russo (0664618), Luc Lubbers (1558501)

Task 1: Describe the database (what are the relations, what is the domain, how many tables it has with a short description of the content of each table).

The Chinook database was used for the realization of the following tasks. It can be accessed and downloaded via this link:
https://www.sqlitetutorial.net/sqlite-sample-database/

The database includes 11 tables. The tables are: media_types, genres, playlists, playlist_track, tracks, artists, Invoices, invoice_items, albums, customers & employees. See also the database diagram in figure 1.

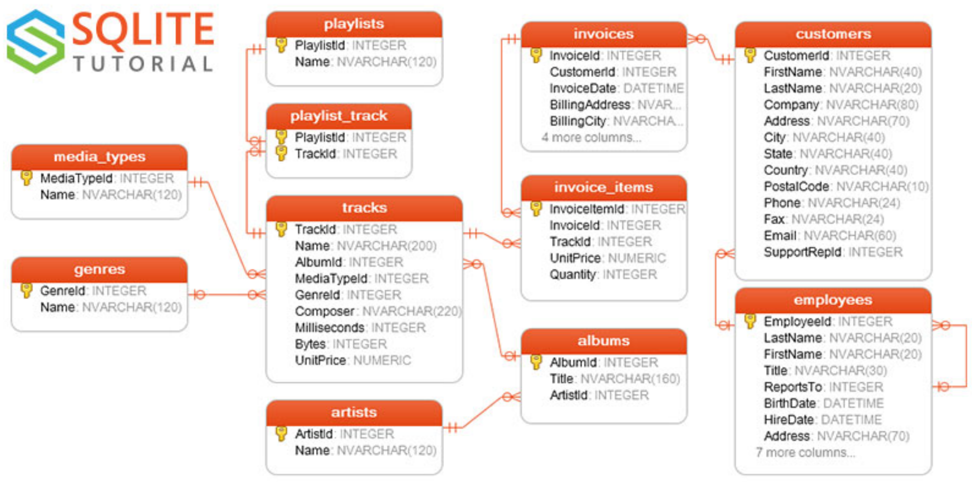

figure 1: Database diagram (source: Sqlite Tutorial).

Database Schema

media_types: (MediaTypeid, Name)

- Relations: 1 (media_types) to many (tracks)
- Description: table that stores the standard media types (MPEG audio and AAC audio)

Genres: (Genreid, Name)
This table is directly related to tracks.

- Relations: 1 (genres) to 1 (tracks)
- Description: table that stores the different musical genres (pop,rock, jazz, etc)

playlists: (PlaylistID, name)
This table is directly related to the playlist_track table.

- Relations: many(tracks) to many (playlists)

- Description: table that contains information about playlists, each playlist contains a list of tracks.

playlist_track: (PlaylistId, TrackId)
- Relations: helps relate tracks and playlists
- Description: table that is used to reflect the relation between tracks and playlists.

This table is directly related to the tracks table.

Tracks: (TrackId, Name, AlbumId, MediaTypeId, GenreId, Composer, Milliseconds, Bytes, UnitPrice)

- Relations: many(tracks) to many (playlists), 1(track) to 1(album)
- Description: table that contains information about music tracks, every track belongs to an album.

Artists: (ArtistsId, Name)
- Relations: 1(artists) to many(albums)
- Description: table that stores artists data

Invoices: (Invoiceid, Customerid, InvoiceDate, BillingAddress, BillingCity, BillingState, BillingCountry, BillingPostalCode, Total)
- Relations: 1(invoice) to many(invoice_items), many(invoices) to 1(customer)
- Description: this table stores invoice header data.

invoice_items: (InvoiceItemId, InvoiceId, TrackId, UnitPrice, Quantity)
- Relations: already written in invoices
- Description: this table stores invoice line items data.

Albums: (AlbumId, Title, ArtistsId)

Customers: (CustomerId, FirstName, LastName, Company, Address, City, State, Country, PostalCode, Phone, Fax, Email, SupportRepid)
- Relations:
- Description: table that contains customers information.

Employees: (EmployeeId, LastName, FirstName, Title, ReportsTo, BirthDate, HireDate, Address, City, State, Country, PostalCode, Phone, Fax, Email)
- Relations:
- Description: table that contains information about employees; the field ***ReportsTo*** specifies who reports whom.

Task 2: To solve upcoming queries:

Question 1

Q Write a query that contains an outer join (left is sufficient).

A

NL	**show the name of all artists and the albums ID and Title that belong to them**
RA	**$\pi_{\text{artists.name, albums.AlbumId, albums.Title}}$ (Albums ⟕ Artists)**

SQL	**SELECT artists.Name, albums.AlbumId, albums.Title
	FROM artists LEFT JOIN albums ON artists.ArtistId = albums.ArtistId
	ORDER BY artists.Name**

Question 2

Q Write a query that returns the number of missing values in a given attribute(look for an attribute that contains missing values).

A

NL	**show id and name of the tracks that have no composer information**
RA	**$\pi_{\text{trackid, name}}$ ($\sigma_{\text{composer = NULL}}$ (tracks))**

SQL	**SELECT TrackId
	FROM tracks
	WHERE Composer IS NULL**

Question 3

Q Write a query that contains (has no) constraint (e.g. return the names of the students who has no courses this period).

A

NL	**show names of the tracks where unit price is not 0.99**
RA	**π_{name}($\sigma_{\text{UnitPrice != 1.99}}$ (tracks))**

SQL	**SELECT Name
	FROM tracks
	WHERE NOT UnitPrice = 0.99**

Question 4

Q Write a query that contains(has only) constraint (e.g. return the names of the students who has only one course this period).

A

NL	**Show all tracks that have a length of less than 120000 milliseconds (2minutes).**
RA	**π_{name} ($\sigma_{\text{milliseconds <120000}}$(tracks))**
SQL	**SELECT Name**

	**FROM tracks
	WHERE Milliseconds < 120000;**

Question 5

Q Write a query that categorizes the records in one of the tables using a specific attribute that has a maximum of 5 distinct values and displays the number of records in each category.

A

NL	**Show the number of records of the tracks per mediatypeId.**
RA	$\pi_{\text{MediaTypeId},\ \mathcal{G}_{\text{count(MediaTypeId)}}}$ **(Tracks)**

SQL	**SELECT MediaTypeId, COUNT(MediaTypeId) as 'Sum'
	FROM tracks
	GROUP BY MediaTypeId;**

Question 6

Q Using a table that contains at least one numerical attribute, write a query that displays the records that contains the min and max values of a numerical attribute.

A

NL	**show the tracks with the highest storage value and the lowest one**
RA	$\pi_{\rho\ \text{LowestStorage}(\mathcal{G}_{\text{Min}}(\text{Bytes}),\ \rho\ \text{HighestStorage}\ (\mathcal{G}_{\text{max}}(\text{Bytes}))}$**(tracks)**

SQL	**SELECT MIN(Bytes) AS LowestStorage, MAX(Bytes) AS HightestStorage
	FROM tracks;**

Question 7

Q Write a query that requires self-join.

A

NL	**Get the information on which employee is reporting to which manager**
RA	τ **m.firstname(**$\pi_{\text{m.FirstName, m.LastName, e.FirstName, e.LastName}}(\text{employees e} \bowtie \text{employees m}))$

SQL	**SELECT m.firstname, m.lastname, e.firstname, e.lastname
	FROM employees e
	INNER JOIN employees m ON m.employeeid = e.reportsto
	ORDER BY m.firstname;**

Task 3: Python coding

The questions of task 3 are answered in a Jupyter Notebook attached in the assignment.

Question 1
Write python code that uses python connectors to extract all the tables from the database and save them to .csv files.

Question 2
Write the queries that you proposed in Task2 using python. You will need to read the tables that will be used in the queries into Pandas Dataframes and write the queries to extract the data from the Dataframes directly (do not use SQL on the original database).

Task 4: Relational Algebra and Boolean Operators

Question 1.a
Write of the definition of the division in relational algebra and explain its meaning in your own words.

Definition from: Database System Concepts - Seventh Edition (Abraham Silberschats, Henrey F. Korth, S. Sudarshan)
The division operator of relational algebra, "÷", is defined as follows. Let r(R) and s(S) be relations, and let S ⊆ R; that is, every attribute of schema S is also in schema R. Given a tuple t, let t[S] denote the projection of tuple t on the attributes in S. Then r ÷ s is a relation on schema R − S (that is, on the schema containing all attributes of schema R that are not in schema S). A tuplet is in r ÷ s if and only if both of two conditions hold:

- t is in ΠR−S(r)
- For every tuple ts in s, there is a tuple tr in r satisfying both of the following:
  a. tr[S] = ts[S]
  b. tr[R − S] = t

In our own words, if we have two tables called X(columns A and B) and Y (column B), when we apply the operator of division ("÷"), it should find the values of column A (from table X) that matches all values with B (column from table Y). The idea is that we are dividing AB by B (AB ÷ B = A).

Question 1.b
Express the operation in terms of the projection, join and difference.
R÷S = πA(R) - πA(πA(R) x S - R

Question 2
Write the operation (=) in terms of (~, ∧, ∨) i.e., write the expression X = Y using only (~, ∧, ∨). The truth table for X = Y is:

X	Y	X = Y
T	T	T
T	F	F
F	T	F
F	F	T

Answer: (X ∧ Y) ∨ ~(X ∧ Y)
Reasoning: we are searching for the **T** values in the output so we need to see which operations are the right ones for getting T on the output. In the first row we do not need to change anything to get T, so the first operation is going to be X ∧ Y. For the last column, we need to negate both X and Y in order to obtain T doing AND operation; so the second operation is ~(X ∧ Y). Finally both operations are combined with OR (∨)

Question 3

Prove that: $(A \vee B) \wedge (\sim A \vee C) \wedge (B \vee C) = (A \vee B) \wedge (\sim A \vee C)$

$= (A \vee B) \wedge (\sim A \vee C) \wedge [(B \vee C) \wedge \mathrm{T}]$

$= (A \vee B) \wedge (\sim A \vee C) \wedge [(B \vee C) \wedge (\sim A \vee \mathrm{A})]$

$= (A \vee B) \wedge (\sim A \vee C) \wedge [(B \vee C) \wedge \sim A] \vee [(B \vee C) \wedge A]$

$= (A \vee B) \wedge (\sim A \vee C) \wedge (\mathrm{A} \vee \sim A)$

$= (A \vee B) \wedge (\sim A \vee C) \wedge \mathrm{T}$

$= (A \vee B) \wedge (\sim A \vee C)$

A	B	C	$(A \vee B)$	$(\sim A \vee C)$	$(B \vee C)$	$(A \vee B) \wedge (\sim A \vee C) \wedge (B \vee C)$	$(A \vee B) \wedge (\sim A \vee C)$
0	0	0	0	1	0	0	0
0	0	1	0	1	1	0	0
0	1	0	1	1	1	1	1
0	1	1	1	1	1	1	1
1	0	0	1	0	0	0	0
1	0	1	1	1	1	1	1
1	1	0	1	0	1	0	0
1	1	1	1	1	1	1	1

By looking at the truth table, we can see that both last columns are the same so we have proven it.