**Appendix**

**References**

ChatGPT. (2024, August 11). Name for the business. OpenAI. Prompt: “Here is the python code to create my SQLite database: conn = sqlite3.connect("SoundShift.db") cursor = conn.cursor() cursor.execute(''' CREATE TABLE Album ( AlbumID INTEGER PRIMARY KEY UNIQUE NOT NULL UNIQUE, Name TEXT NOT NULL, ReleaseDate DATE, CONSTRAINT Name CHECK (LENGTH(Name) <= 50) )''') cursor.execute(''' CREATE TABLE Song ( SongID INTEGER PRIMARY KEY UNIQUE NOT NULL UNIQUE, Name TEXT NOT NULL, Length INT, AlbumID INTEGER, ReleaseDate DATE, FOREIGN KEY (AlbumID) REFERENCES Album(AlbumID), CONSTRAINT Length CHECK (Length > 0), CONSTRAINT Name CHECK (LENGTH(Name) <= 50) ) ''') cursor.execute(''' CREATE TABLE BankDetails ( BankDetailsID INTEGER PRIMARY KEY UNIQUE NOT NULL UNIQUE, CardNumber INTEGER NOT NULL CHECK (LENGTH(CardNumber) <= 19), CardHolderName TEXT NOT NULL CHECK (LENGTH(CardHolderName) <= 50), ExpirationDate DATE NOT NULL CHECK (ExpirationDate > DATE('now')), CVV INTEGER NOT NULL CHECK (LENGTH(CVV) <= 4) ) ''') cursor.execute(''' CREATE TABLE Customer ( CustomerID INTEGER PRIMARY KEY UNIQUE NOT NULL, Username TEXT NOT NULL CHECK (LENGTH(UserName) <= 20), BankDetailsID INTEGER, Email Text NOT NULL CHECK (Email LIKE '%@%'), Password TEXT NOT NULL CHECK (LENGTH(Password) = 64), CONSTRAINT BankDetailsID FOREIGN KEY (BankDetailsID) REFERENCES BankDetails(BankDetailsID) )''') cursor.execute(''' CREATE TABLE RecentlyPlayedSongs ( BridgeID INTEGER PRIMARY KEY UNIQUE NOT NULL, SongID INTEGER, CustomerID INTEGER, DateListenedTo DATE, FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID), FOREIGN KEY (SongID) REFERENCES Song(SongID) )''') cursor.execute(''' CREATE TABLE Artist ( ArtistID INTEGER PRIMARY KEY UNIQUE NOT NULL, StageName TEXT NOT NULL CHECK (LENGTH(StageName) <= 50), FirstName TEXT CHECK (LENGTH(FirstName) <= 50), LastName TEXT CHECK (LENGTH(LastName) <= 50) )''') cursor.execute(''' CREATE TABLE Genre ( GenreID INTEGER PRIMARY KEY UNIQUE NOT NULL, Name TEXT NOT NULL CHECK (LENGTH(Name) <= 50) )''') cursor.execute(''' CREATE TABLE SongArtistBridge ( BridgeID INTEGER PRIMARY KEY UNIQUE NOT NULL, ArtistID INTEGER, SongID INTEGER, CONSTRAINT SongID FOREIGN KEY (SongID) REFERENCES Song(SongID), CONSTRAINT ArtistID FOREIGN KEY (ArtistID) REFERENCES Artist(ArtistID) )''') cursor.execute(''' CREATE TABLE SubscriptionInvoice ( InvoiceID INTEGER PRIMARY KEY UNIQUE NOT NULL, BankDetailsID INTEGER, CustomerID INTEGER, SaleDate DATE NOT NULL, AmountCharged NUMBER NOT NULL CHECK (AmountCharged > 0 AND ROUND(AmountCharged, 2) = AmountCharged), SubscriptionLengthBought INTEGER NOT NULL CHECK (SubscriptionLengthBought > 0), CONSTRAINT CustomerID FOREIGN KEY (CustomerID) REFERENCES Customer(CustomerID), CONSTRAINT BankDetailsID FOREIGN KEY (BankDetailsID) REFERENCES BankDetails(BankDetailsID) )''') cursor.execute(''' CREATE TABLE SongGenreBridge ( BridgeID INTEGER PRIMARY KEY UNIQUE NOT NULL, GenreID INTEGER, SongID INTEGER, CONSTRAINT SongID FOREIGN KEY (SongID) REFERENCES Song(SongID), CONSTRAINT GenreID FOREIGN KEY (GenreID) REFERENCES Genre(GenreID) )''') conn.commit() cursor.close() conn.close() Write SQL code to add a bunch of records to the database. Have at least 100 songs, 10 customers, 20 artists, each artist having multiple songs, each customer having their own banking details, each customer having at least 1-2 subscriptions bought, each having played at least 5 songs recently, and 8 albums, each having at least 10 unique songs, all the appropriate bridging table data should be created, and have at least 5 genres. And connect every single song with a genre, and at least 20 with a second genre”

Emn178. (n.d.). *Sha256*. Online Tools. <https://emn178.github.io/online-tools/sha256.html>

Ian. (2020, May 8). *How to generate a random number within a specified range in SQLite*. Database.Guide. <https://database.guide/how-to-generate-a-random-number-within-a-specified-range-in-sqlite/>