Musical Track Database

This application will read an iTunes export file in XML and produce a properly normalized database with this structure:

```
CREATE TABLE Artist (
    id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
    name    TEXT UNIQUE
);

CREATE TABLE Genre (
    id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
    name    TEXT UNIQUE
);

CREATE TABLE Album (
    id  INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE,
    artist_id  INTEGER,
    title   TEXT UNIQUE
);

CREATE TABLE Track (
    id  INTEGER NOT NULL PRIMARY KEY
        AUTOINCREMENT UNIQUE,
    title TEXT  UNIQUE,
    album_id  INTEGER,
    genre_id  INTEGER,
    len INTEGER, rating INTEGER, count INTEGER
);
```

If you run the program multiple times in testing or with different files, make sure to empty out the data before each run.

You can use this code as a starting point for your application: http://www.py4e.com/code3/tracks.zip. The ZIP file contains the **Library.xml** file to be used for this assignment. You can export your own tracks from iTunes and create a database, but for the database that you turn in for this assignment, only use the **Library.xml** data that is provided.

To grade this assignment, the program will run a query like this on your uploaded database and look for the data it expects to see:

If you run the program multiple times in testing or with different files, make sure to empty out the data before each run.

You can use this code as a starting point for your application: http://www.py4e.com/code3/tracks.zip. The ZIP file contains the **Library.xml** file to be used for this assignment. You can export your own tracks from iTunes and create a database, but for the database that you turn in for this assignment, only use the **Library.xml** data that is provided.

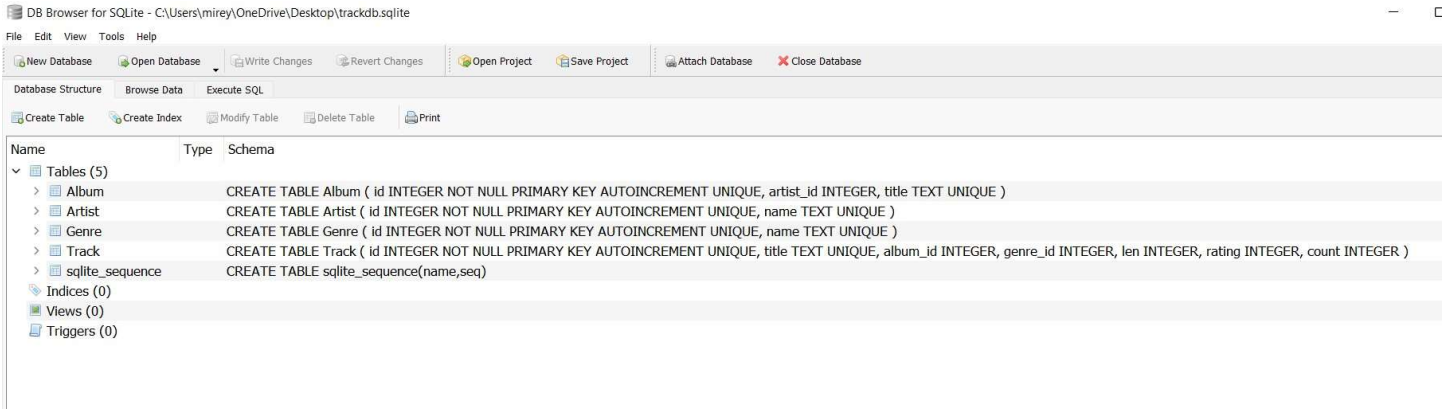
To grade this assignment, the program will run a query like this on your uploaded database and look for the data it expects to see:

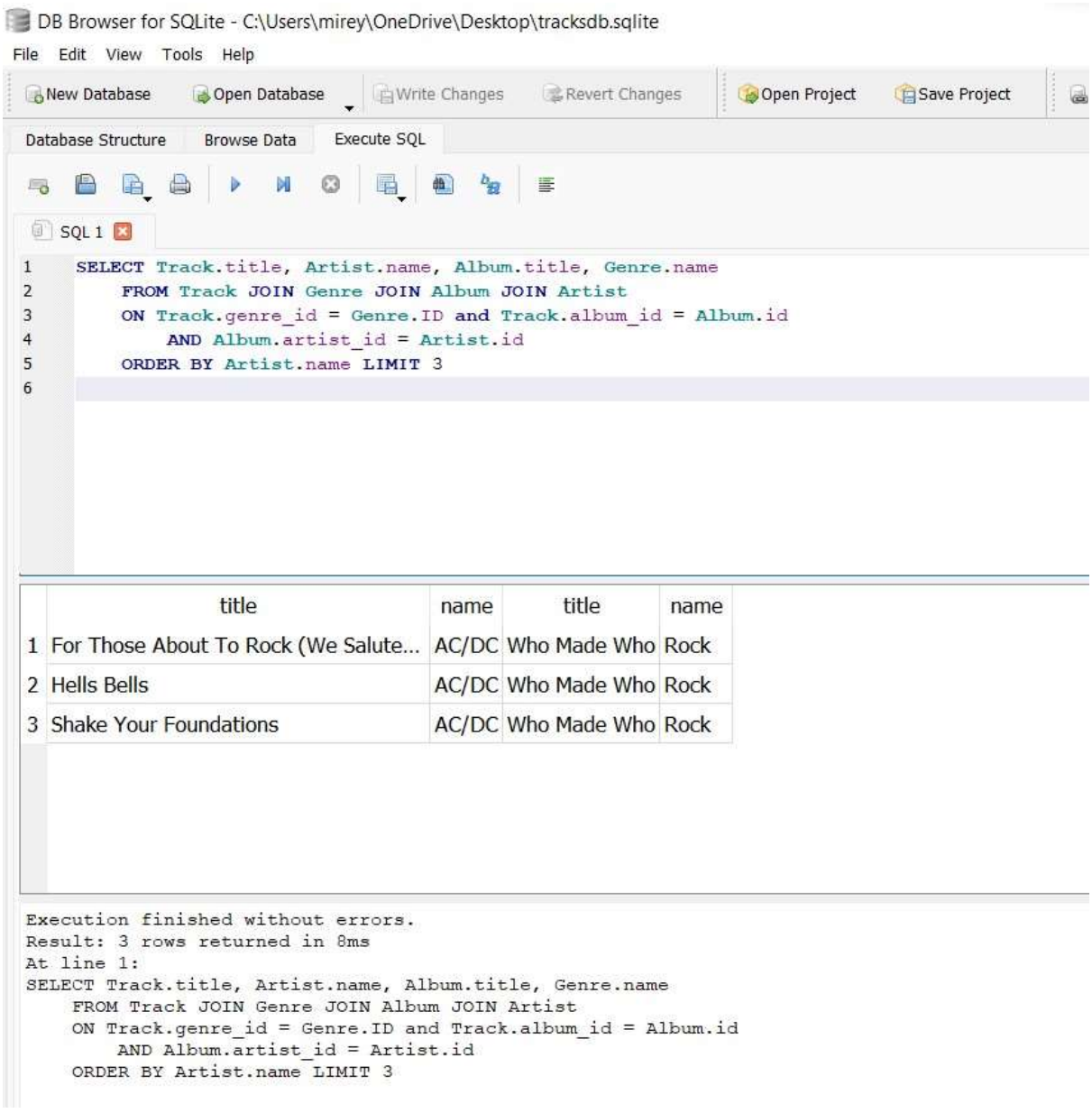
```
SELECT Track.title, Artist.name, Album.title, Genre.name
    FROM Track JOIN Genre JOIN Album JOIN Artist
    ON Track.genre_id = Genre.ID and Track.album_id = Album.id
        AND Album.artist_id = Artist.id
    ORDER BY Artist.name LIMIT 3
```

The expected result of the modified query on your database is: (shown here as a simple HTML table with titles)

Track	Artist	Album	Genre
Chase the Ace	AC/DC	Who Made Who	Rock
D.T.	AC/DC	Who Made Who	Rock
For Those About To Rock (We Salute You)	AC/DC	Who Made Who	Rock

---


DB Browser for SQLite - C:\Users\mirey\OneDrive\Desktop\trackdb.sqlite
File Edit View Tools Help
New Database
Open Database
Write Changes
Revert Changes
Open Project
Save Project
Attach Database
Close Database
Database Structure
Browse Data
Execute SQL
Create Table
Create Index
Modify Table
Delete Table
Print
Name
Type
Schema
Tables (5)
Album
CREATE TABLE Album (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, artist_id INTEGER, title TEXT UNIQUE)
Artist
CREATE TABLE Artist (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, name TEXT UNIQUE)
Genre
CREATE TABLE Genre (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, name TEXT UNIQUE)
Track
CREATE TABLE Track (id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT UNIQUE, title TEXT UNIQUE, album_id INTEGER, genre_id INTEGER, len INTEGER, rating INTEGER, count INTEGER)
sqlite_sequence
CREATE TABLE sqlite_sequence(name,seq)
Indices (0)
Views (0)
Triggers (0)


DB Browser for SQLite - C:\Users\mirey\OneDrive\Desktop\tracksdb.sqlite
File Edit View Tools Help
New Database
Open Database
Write Changes
Revert Changes
Open Project
Save Project
Database Structure
Browse Data
Execute SQL
SQL 1
1 SELECT Track.title, Artist.name, Album.title, Genre.name
2 FROM Track JOIN Genre JOIN Album JOIN Artist
3 ON Track.genre_id = Genre.ID and Track.album_id = Album.id
4 AND Album.artist_id = Artist.id
5 ORDER BY Artist.name LIMIT 3
6
title
name
title
name
1 For Those About To Rock (We Salute... AC/DC Who Made Who Rock
2 Hells Bells AC/DC Who Made Who Rock
3 Shake Your Foundations AC/DC Who Made Who Rock
Execution finished without errors.
Result: 3 rows returned in 8ms
At line 1:
SELECT Track.title, Artist.name, Album.title, Genre.name
FROM Track JOIN Genre JOIN Album JOIN Artist
ON Track.genre_id = Genre.ID and Track.album_id = Album.id
AND Album.artist_id = Artist.id
ORDER BY Artist.name LIMIT 3