Northeastern University
CS5200 – DBMS
Spring 2025, Derbinsky

HW1

This assignment focuses on using SQL to query the Chinook database. To receive credit, submit to Canvas **only** the following files:

- `p1.sql`: a text file containing **only** your SQL query for problem 1
- `p2.sql`: a text file containing **only** your SQL query for problem 2
- `p3.sql`: a text file containing **only** your SQL query for problem 3
- `p4.sql`: a text file containing **only** your SQL query for problem 4
- `p5.sql`: a text file containing **only** your SQL query for problem 5

Each deviation from these instructions will incur a 1 point penalty.

Querying Chinook

This assignment has five (5) problems worth 25 points total. For each problem, write an SQL query against the Chinook Database v1.4.5. Each query **must** run successfully using DB Browser. A description of the correct result set for each problem is provided – your query must reproduce this result exactly (including attribute names/order and row order/contents).

To help, you have been provided an `SQLiteDiff` utility to compare the output of your query versus a supplied answer in CSV format. To use this program, create a text file that contains **only** your SQL query for a particular problem. Then run the program, supplying first the path to the supplied CSV file to compare against, then the path to your SQL file:

```
$ java -jar SQLiteDiff.jar p1.csv p1.sql
```

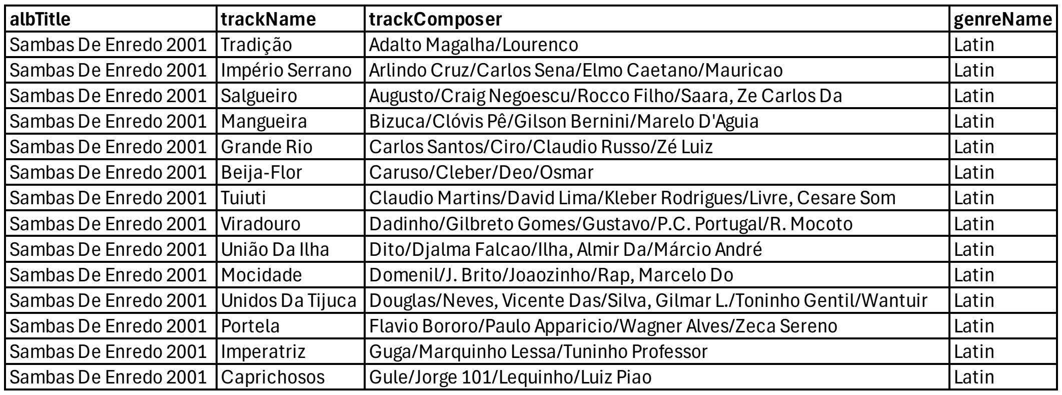
The program will either report success, or describe an issue. **If your query does not produce the correct output via this utility, you can earn at most 2 points for each problem.**

Note that each question warns against using numeric ids (i.e., internal foreign key values). If your solution uses such identifiers, or tries to "game the system" (i.e. write a query that produces the correct output but does not adhere to the spirit/constraints of the question), you will receive **no** credit.

Each question also indicates a required sorting of the resulting rows. Because a database management system may produce rows in an arbitrary order, it is always good practice to explicitly indicate sorting in your SQL. Thus, **if the result set has more than one row and your SQL does not fully specify row sorting order (according to the problem) you will lose 1 point, even if the output *happens* to match the answer**.

While an RDBMS will try to optimize your query, avoid queries that are wildly large/inefficient, such as those that unjustifiably include unneccessary joins or correlated subqueries

Problem 1 (5 points). Write a query to produce a very special list of tracks: the album's artist is "Various Artists", the tracks have a composer, and their genre is Latin. For each track, include the album title, the track name, the track composer, and the genre. You should sort the results by composer (alphabetical). Your query must not hardcode any numeric ids (e.g., `ArtistID`, `GenreId`), nor identify tracks by criteria other than that listed above.



albTitle	trackName	trackComposer	genreName
Sambas De Enredo 2001	Tradição	Adalto Magalha/Lourenco	Latin
Sambas De Enredo 2001	Império Serrano	Arlindo Cruz/Carlos Sena/Elmo Caetano/Mauricao	Latin
Sambas De Enredo 2001	Salgueiro	Augusto/Craig Negoescu/Rocco Filho/Saara, Ze Carlos Da	Latin
Sambas De Enredo 2001	Mangueira	Bizuca/Clóvis Pê/Gilson Bernini/Marelo D'Aguia	Latin
Sambas De Enredo 2001	Grande Rio	Carlos Santos/Ciro/Claudio Russo/Zé Luiz	Latin
Sambas De Enredo 2001	Beija-Flor	Caruso/Cleber/Deo/Osmar	Latin
Sambas De Enredo 2001	Tuiuti	Claudio Martins/David Lima/Kleber Rodrigues/Livre, Cesare Som	Latin
Sambas De Enredo 2001	Viradouro	Dadinho/Gilbreto Gomes/Gustavo/P.C. Portugal/R. Mocoto	Latin
Sambas De Enredo 2001	União Da Ilha	Dito/Djalma Falcao/Ilha, Almir Da/Márcio André	Latin
Sambas De Enredo 2001	Mocidade	Domenil/J. Brito/Joaozinho/Rap, Marcelo Do	Latin
Sambas De Enredo 2001	Unidos Da Tijuca	Douglas/Neves, Vicente Das/Silva, Gilmar L./Toninho Gentil/Wantuir	Latin
Sambas De Enredo 2001	Portela	Flavio Bororo/Paulo Apparicio/Wagner Alves/Zeca Sereno	Latin
Sambas De Enredo 2001	Imperatriz	Guga/Marquinho Lessa/Tuninho Professor	Latin
Sambas De Enredo 2001	Caprichosos	Gule/Jorge 101/Lequinho/Luiz Piao	Latin

Problem 2 (5 points). Write a query to produce a list of a subset of tracks on the "Heavy Metal Classic" playlist: those whose genre includes the word "Metal" but whose media type does NOT include the words "Protected" or "Purchased". For each track, include the track name, media type, genre, and unit price. You should sort the results by track length (longest first). Your query must not hardcode any numeric ids (e.g., `PlaylistID`, `GenreID`, `MediaTypeID`), nor identify tracks by criteria other than that listed above. To format the track price, you may find it useful to use the `PRINTF`[1] function.

trackName	trackType	genreName	trackPrice
Master Of Puppets	MPEG audio file	Metal	$0.99
The Four Horsemen	MPEG audio file	Metal	$0.99
Seek & Destroy	MPEG audio file	Metal	$0.99
Creeping Death	MPEG audio file	Metal	$0.99
Where Eagles Dare	MPEG audio file	Metal	$0.99
N.I.B.	MPEG audio file	Metal	$0.99
2 Minutes To Midnight	MPEG audio file	Metal	$0.99
Enter Sandman	MPEG audio file	Metal	$0.99
For Whom The Bell Tolls	MPEG audio file	Metal	$0.99
Wasted Years	MPEG audio file	Metal	$0.99
Killers	MPEG audio file	Heavy Metal	$0.99
Supernaut	MPEG audio file	Metal	$0.99
Looks That Kill	MPEG audio file	Metal	$0.99
Run to the Hills	MPEG audio file	Metal	$0.99
Live To Win	MPEG audio file	Metal	$0.99
Wrathchild	MPEG audio file	Heavy Metal	$0.99
Ace Of Spades	MPEG audio file	Metal	$0.99

[1]`https://www.sqlite.org/lang_corefunc.html` (and here's a pretty good primer if you've never used `printf` in another language: `https://www.codingunit.com/printf-format-specifiers-format-conversions-and-formatted-output`)

Problem 3 (5 points). Write a query that produces an alphabetical list of all the genres in all the tracks of the "Classical" playlist. Your query must not hardcode any numeric ids (e.g., `GenreId`, `PlayListId`), nor identify genres by criteria other than that listed above.

classyGenre
Classical
Opera
Soundtrack

Problem 4 (5 points). Write a query to produce an alphabetical list of all states supplied for invoice billing addresses. Your query must not hardcode any numeric ids (e.g. `InvoiceId`), nor identifying information about the billing states other than the criteria listed above.

billingState
AB (Canada)
AZ
BC (Canada)
CA
DF (Brazil)
Dublin (Ireland)
FL
IL
MA
MB (Canada)
NS (Canada)
NSW (Australia)
NT (Canada)
NV
NY
ON (Canada)
QC (Canada)
RJ (Brazil)
RM (Italy)
SP (Brazil)
TX
UT
VV (Netherlands)
WA
WI

Problem 5 (5 points). You are to write a report to assist Chinook Corp in congratulating its sales representative of the month, where the goal was to sell more than $20 in a single invoice (including *at least* one video track) to a USA billing address. Your query should return the invoice total; the first & last name of the customer; the first & last name of the corresponding sales representative, as well as their email address; and the first & last name, as well as email address, of the sales representative's boss. Your query must not hardcode any numeric ids (e.g., `InvoiceID`, `MediaTypeID`, `EmployeeID`), nor identify the invoice/customer/employees by criteria other than that listed above.

bigInvoice	customerFirst	customerLast	repFirst	repLast	repEmail	bossFirst	bossLast	bossEmail
23.86	Richard	Cunningham	Margaret	Park	margaret@chinookcorp.com	Nancy	Edwards	nancy@chinookcorp.com