**CS 2300 – Homework Assignment 3**

Due date: Thursday, November 2nd, by 9:30 am

This is a group assignment. You must type out your answers and submit one PDF file (questions 1 and 2) and one SQL file (question 3) through Canvas, with each named Homework 3 - Group##.pdf and Homework 3 - Group##.sql, replacing ## with your group’s number.

Turn in a printed copy of both your answers and your SQL code. Use a monospaced font (e.g. Courier New, Consolas, Ubuntu) when printing the SQL code.

1. (*10 points*) Convert the following ER diagram into relational schemas  
e.g. R(id:int, name:str, …)

Make sure to denote foreign keys.

Movie

Theater

SHOWS

N

M

showtime

id

title

runtime

id

name

address

ticketprice

actor

street

zipcode

2. (*20 points*) Write a relational algebra expression that will create a relation of each actor’s latest film and their age when they starred in it - i.e. a relation that creates the following schema:  
  
 ActorLatestMovie(actorname:str, ageduringmovie:int, movietitle:str, movieyear:int)

Use the following relations in your expression.

Actor(id:int, name:str, birthyear:int)

Movie(id:int, title:str, year:int)

STARS\_IN(aid:int, mid:int)

Assume there exists some actors who haven’t starred in any movies. They ought to appear in the relation.

Note: When using the generalized projection, an operation on attributes with a NULL value results in NULL. Example:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | R | |  |  |  |
|  | x:int | y:int |  | x-y:int |  |
|  | 3 | 4 |  | -1 |  |
|  | 5 | NULL |  | NULL |  |

*Please separate out your answers using assignment statements. Example:*

*Page left blank for Problem 2 answer.*

3. (*40 pts*) Write one .sql file that creates a simple database, detailed below. Construct your SQL statements in accordance with the assumptions at the bottom of this page. Submit your .sql file through Canvas.

Indicate which database system (e.g. MySQL, PostgreSQL, SQLite, SQL\*Plus) you have used to test your code as a comment in the SQL code (e.g. -- Tested on MySQL). It is expected that your SQL file can be directly used to build a database in that system.

(1) Create 6 tables as shown in Figure 1 below.

1. You can rename invalid attribute names appropriately - e.g. rename cat# to catno or catalog\_number or whatever.
2. You need to create constraints on the tables. For example, create primary keys, foreign keys, and domain / not null constraints. See the assumptions on the next page.

(2) Insert values into the tables as shown in Figure 2 below.

(3) Create SELECT queries to show the contents of each table.

(4) Implement the following SQL queries.

1. Find the song titles from songs included on rereleases that were released both after 2008 and on the ‘Ultra Records’ label.
2. For each rerelease, list its associated release’s title and the rerelease’s year and the number of songs it contains.
3. Compute the runtime (in seconds) of each rerelease and include its catalog number and associated release title.
4. For each remixed song, display it’s title, duration, (remixing) artist name, and the title and (original) artist name who it remixed.

(5) Drop the tables in an order so that no referential integrity constraints are violated (and thus no CASCADEs are necessary).

**Assumptions:**

* Song durations are in units of seconds and must be > 0
* Years cannot be before 1900
* Unless otherwise stated, assume anything that is a string will not exceed 80 characters in length
* UPCs may not be longer than 12 characters
* Rerelease.medium can only be one of the following values: CD, Web, LP, 45, Tape
* Label.labbr will not be longer than 5 characters
* Any column in the relations in Figure 2 that does not have any NULL values (empty cells) should be assumed to have a NOT NULL constraint. Likewise, any column with a least one null value should be assumed to be nullable.

|  |  |
| --- | --- |
| AlbumTrack(rid:int, cat#:str, sid:int,  trackno:int) | Label(lid:int, lname:str,  labbr:str) |
| Song(sid:int, stitle:str, duration:int,  remixof:int, artist:int) | Release(rid:int, rtitle:str,  year:int, aid:int) |
| Rerelease(catno:str, rid:int, upc:str,   label:int, year:int, medium:str) | Artist(aid:int, aname:str) |
| **Figure 1:** Sample Relational Model | |

|  |
| --- |
|  |
| **Figure 2**: Sample Database State for Problem 3.2. |