Pontificia Universidad Católica de Chile
Escuela de Ingeniería
Departamento de Ciencia de la Computación

IIC2115 – Programación como Herramienta para la Ingeniería (II/2017)

Tarea 3

Objetivos

- Aplicar conocimientos para la correcta modelación de una base de datos relacional.
- Realizar consultas a la base de datos construida, usando SQL.
- Utilizar Python para realizar consultas y modificaciones a una base de datos.
- **Lenguaje a utilizar:** Python 3.6
- **Lugar:** GitHub
- **Fecha:** 03 de junio
- **Hora:** 23:59
- **Desarrollo en parejas o individual**
- Las inscripciones para las parejas son hasta el martes 22 de mayo a las 23:59 y solo se aceptan las inscritas por este form: `https://goo.gl/forms/USD9VnQtVBhArFxb2`

Recomendaciones

Lee la tarea completa antes de comenzar a trabajar. Fíjate en los sustantivos y verbos relacionados con el problema. Al terminar, escribe en la parte final de tu informe, un comentario, donde detalles tus opiniones y percepción de la tarea. Estos comentarios son útiles para mejorar el curso durante el mismo semestre, por lo que se sugiere realizarlos a conciencia. No habrá penalizaciones en casos donde el comentario sea negativo.

1 Introducción

Como cuerpo de ayudantes, nos quedamos sin ideas para tareas y actividades. Hans no nos paga lo suficiente para pensar en esto y solo nos dijo que estaban viendo SQL. Por esa razón, deberán implementar un programa para interactuar con una base de datos de música. Lamentablemente, Bastian juntó toda la base de datos mientras hacía `PUSH` y Hernán se encontraba renspondiendo e-mails y derivando alumnos al formulario de contacto cuando ocurrió aquel horrible accidente, así que no pudo arreglar el error de Bastian a tiempo. Es aquí donde entras tú para reparar tal desastre.

2 Descripción

En esta tarea, tendrás que modelar un diagrama de relaciones que permita una interacción eficiente con una base de datos de música. Para esto, deberás procesar los datos entregados, implementar las tablas de entidades y relaciones necesarias y generar un menú por consola que permita: agregar, modificar, consultar y eliminar dicha información. Además, deberás implementar un sistema de usuarios quienes pondrán guardar canciones que escucharon y hacer consultas sobre ellas. En particular, deberás entregar al menos 3 archivos .py que realicen diferentes funcionalidades:

- Poblar base de datos
- Consultas en python
- Implementar sistema por consola.

3 Archivos csv entregados

Para esta tarea se entregan 2 archivos .csv cuyo separador es ",".

El archivo [tracks.csv] contiene las siguientes columnas:

- Artist: Nombre y apellido del artista.
- Album: Nombre del álbum de la canción.
- Track: Nombre de la canción.
- Duration: Duración de la canción. Esta en formato `mm:ss`

El archivo [genres.csv] contiene las siguientes columnas:

- Artist: Nombre y apellido del artista.
- Genre: Lista con géneros musicales.

Para abrir el archivo [genres.csv], deben usar encoding "UTF-8" y así no generar error.

Instrucciones

En esta tarea deberás entregar al menos 3 archivos ".py". A continuación se explicará qué se espera que haga cada archivo:

Poblar base de datos

Como podrás haber notado tras inspeccionar los archivos .csv entregados, no existen usuarios registrados en el sistema o canciones escuchadas por cada usuario. Por lo tanto, deberás crearlos tú con un nombre de usuario y una contraseña. Los nombres de usuario son únicos y deben tener **más de tres caracteres**. Las

contraseñas deben tener un **mínimo de 6 caracteres sin espacios**.

En este archivo .py, deberás crear los archivos .csv necesarios con al menos 5 usuarios y para cada usuario, 4 canciones escuchadas por cada uno. Las canciones escuchadas deben incluir el tiempo(mes, día, hora y minuto) en el cual fue escuchada. Se espera que este archivo sea el primero en ejecutarse para que se creen los archivos CSV necesarios.

Para trabajar los CSV deben utilizar las herramientas vistas en clases. La única librería externa que pueden utilizar en esta parte es CSV, cuya documentación pueden encontrar aquí:
https://docs.python.org/3.6/library/csv.html

Consultas en python

En esta parte, se solicita que implementes con las herramientas vistas en clases las siguientes dos consultas:

- Consulta 1: implementar la función `generos_populares(limite)`. Esta función retorna un diccionario cuya `key` es el nombre de cada usuario y la `value` es una lista de tuplas con los géneros más repetidos entre los artistas escuchados (es decir, cuando un usuario escucha la canción *A*, si el artista que la canta posee los géneros *1, 2 y 3*, se cuentan esos 3 géneros en esta consulta), y la cantidad de veces que se repite dicho género. Debe retornar solamente aquellos géneros que se hayan escuchado más de `limite` veces. El retorno será de la forma:

```
{
 'Hernán': [(género_1, 6), (género_2, 5), (género_3, 5), (género_4, 4)],
 'Daniela': [(género_1, 7), (género_2, 3), (género_6, 3), (género_7, 3)],
 .
 .
 .
}
```

- Consulta 2: implementar la función `tiempo_gastado(minutos)`. Esta función retorna una lista de usuarios que han invertido más tiempo escuchando música que lo indicado en `minutos`.

En esta parte deberás utilizar cualquier herramienta de Python que hayas aprendido en el curso, exceptuando la materia de SQL, para procesar la información y contestar las 2 consultas.

Implementar sistema por consola

Tu sofisticado sistema debe soportar a la perfección la **creación**, **consulta**, **actualización** y **destrucción** tanto de usuarios como de canciones mediante el uso del lenguaje **SQL**. Por esta razón, debes construir tu sistema basándote en un modelo adecuado para soportar cada una de las operaciones antes mencionadas. Se espera entonces que tu modelo incluya al menos **tres tablas distintas**, todas las **restricciones de integridad** necesarias y que toda entidad posea un **id único**, de manera que las comparaciones para realizar *joins* entre tablas se hagan solamente con respecto a los ids. El modelo debe incluír la misma información

entregada en los .csv[1].

Esta parte de la tarea consiste en la elaboración de un menú por consola que permitirá realizar toda interacción con la base de datos. Los requisitos mínimos con los que debe cumplir tu sofisticado sistema son:

- Tener un sistema de registro de nuevos usuarios. La consola debe contar con una opción para registrar nuevos usuarios (solicitando todos los datos necesarios) o ingresar al sistema con solo dar el nombre de usuario y contraseña.
- Permitir la adición de nuevas canciones, artistas, géneros musicales y álbumes. En resumen, agregar entradas a las tablas de la base de datos.
- Permitir al usuario "escuchar" una canción, lo cual agrega esa canción a su lista de canciones escuchadas. Si el usuario solicita escuchar una canción ya escuchada, esta se agrega por segunda vez.
- Permitir la modificación de datos de una canción.
- Eliminar personas de forma permanente de un sistema. En este caso, se debe eliminar toda información que exista de esa persona en la base de datos, eso implica eliminar también sus canciones guardadas.

Las consultas mínimas que debe soportar su sistema[2] son:

1. Todas las consultas necesarias para cumplir con los requisitos anteriores (*e.g.* la(s) consulta(s) para cumplir con el requisito de adición de nueva información).
2. Géneros más repetidos entre los artistas más escuchados. Esta consulta debe ser implementada **totalmente** en SQL.
3. Tiempo invertido. El usuario puede solicitar el total de tiempo invertido para las últimas K canciones escuchadas. Si K es mayor al número de canciones escuchadas, se acota a solicitar el tiempo invertido en todas las canciones escuchadas. Esta consulta debe ser implementada **totalmente** en SQL.
4. Búsqueda de amiguitos: el usuario puede solicitar sus k amigos más cercanos, según su puntaje de compatibilidad. Mientras mayor es este valor, más amigo es del usuario. El puntaje de compatibilidad se calcula como:
    - Canciones en común: por cada canción en común entre dos usuarios, el puntaje aumenta en 1.5 puntos.
    - Artistas en común: por cada artista en común entre dos usuarios, el puntaje aumenta en 1.0 puntos.
    - Géneros en común: por cada género en común entre dos usuarios, el puntaje aumenta en 0.5 puntos.

Ejemplo 1: si una canción coincide, el puntaje es:

$$PC = 1.5 + 1 + 0.5 \times cantidad\_generos\_coincidentes$$

[1]La duración de las canciones las pueden guardar en el formato que quieran (en cantidad de segundos, cantidad de minutos, etc)

[2]Que su sistema soporte las consultas se refiere a que provea la posibilidad de ejecutarlas mediante una interacción por cosola

Ejemplo 2: si un artista coincide, el puntaje es:

$$PC = 1 + 0.5 \times cantidad\_generos\_coincidentes$$

En las consultas que diga "**totalmente** en SQL" está **prohibido** el uso de herramientas de Python. En las que no esté especificada esa restricción, se pueden usar herramientas de Python, pero **debe** haber uso de SQL también.

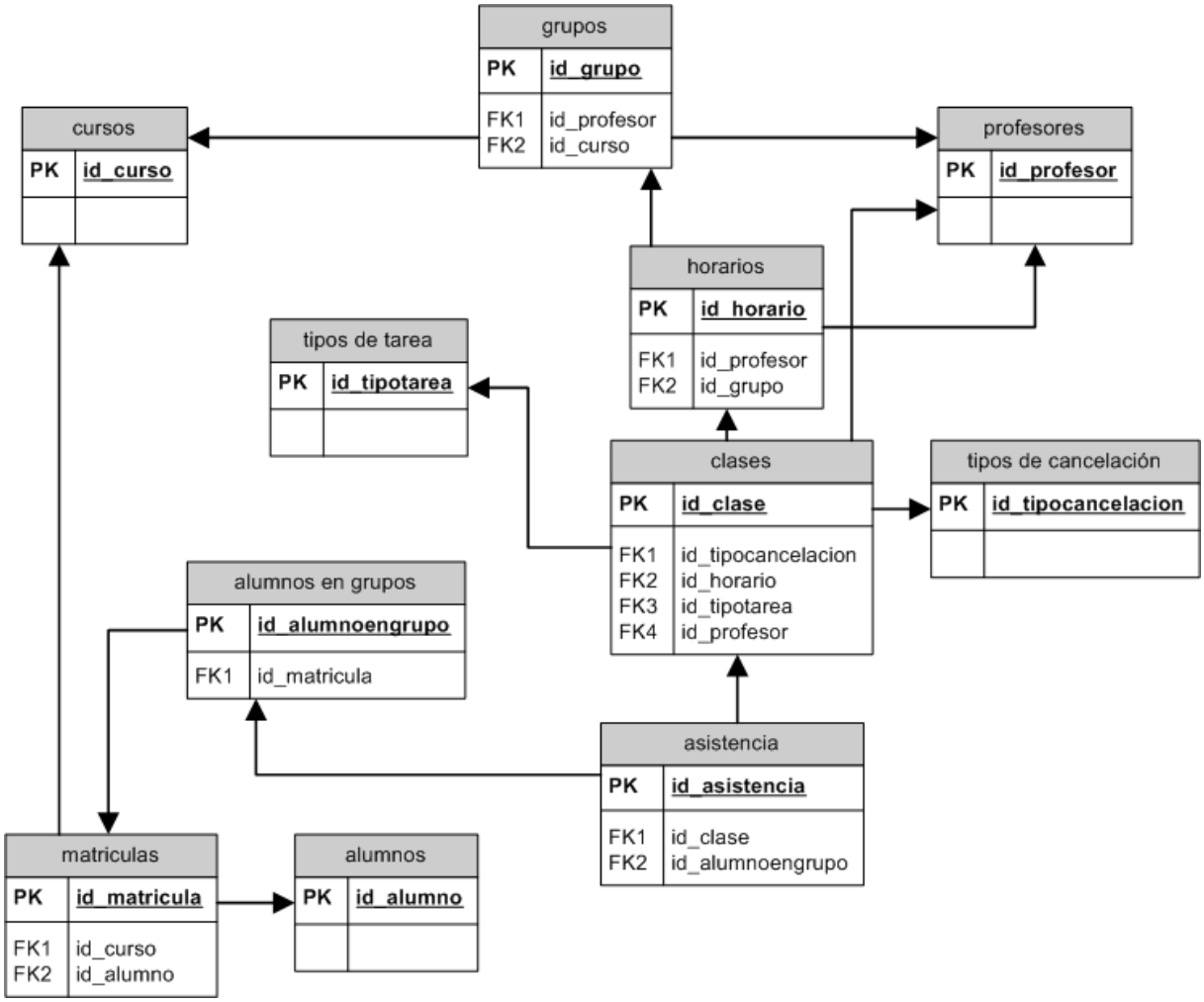
Se espera que el manejo de inputs se base en números para seleccionar las opciones en consola (excepto cuando se pida algún dato de una tabla, por ejemplo, un artista o canción), por ejemplo:

```
Hola Hans! Qué consultas deseas realizar hoy:
    1) ...
    2) ...
    3) ...
Ingrese opción: 1
Ingrese la canción sobre la cual desea consultar: Believe
```

Por lo tanto, al ingresar como input un `1`, se realiza la primera consulta. **Su consola debe manejar correctamente inputs inesperados**, es decir, el programa no debe caerse ante el ingreso de opciones que no existen.

Informe

Para esta parte deberás entregar un manual de usuario y un diagrama de como el siguiente:



El diagrama debe especificar:

- Cada tabla creada para su base de datos.
- Cada atributo de las tablas.
- Destacar los atributos utilizados para conectar la información de una tabla con otra.
- Flechas que indiquen las relaciones de una tabla, es decir, dado una fila de una tabla, a que otra tabla se puede acceder.

Recomendamos fuertemente utilizar la siguiente página: https://www.draw.io/ la cual puede exportar el diagrama en `PDF` y agregarlo al LaTeX. Puedes, por supuesto, utilizar otras páginas o herramientas para este diagrama, según lo estimes conveniente.

Para el manual de usuario, deberás mostrar cada menú de tu consola e indicar el input esperado (si es un integer, un string, una fecha, etc). Debe tener en cuenta que no conocemos su interfaz y por ello, este manual de usuario es el único medio para que sepamos como utilizar su sofisticado sistema.

¡OJO! No se aceptará el uso de *pantallazos* para mostrar aspectos de su consola. Estos deben estar mostrados con alguna herramienta proporcionada por LaTeX. Se recomienda investigar `listing` y/o `begin{verbatim}`.

Para su corrección **debe** subir el archivo PDF junto a todos los archivos necesarios para la compilación del pdf, es decir, los *.tex*, cualquier imagen, pdf, etc.

Consideraciones Importantes

- Para revisar la parte 3 (su sistema), este será corregida únicamente por consola. Verifiquen que cada consulta o funcionalidad implementada en su sistema posea una forma de acceso desde la consola (menu). En otro caso **no será corregida**. En otras palabras, en caso de que una consulta o funcionalidad no esté implementada, el ayudante no revisará el código para ver si al menos estaba programada y en esta misma línea, mandar a recorregir indicando: "hice la funcionalidad pero no está en el menú por X motivo" no será aceptado.
- Se evaluará uso de LaTeXcomo corresponde. Por lo tanto, para el manual de usuario, debe recurrir a los comandos de LaTeXpara mostrar lo que quiere y no pegar imágenes.
- No se evalúa informe que no sea subido con todos los archivos necesarios para compilar nuevamente.

Política de Integridad Académica

Los alumnos de la Escuela de Ingeniería deben mantener un comportamiento acorde al Código de Honor de la Universidad:

"Como miembro de la comunidad de la Pontificia Universidad Católica de Chile me comprometo a respetar los principios y normativas que la rigen. Asimismo, prometo actuar con rectitud y honestidad en las relaciones con los demás integrantes de la comunidad y en la realización de todo trabajo, particularmente en aquellas actividades vinculadas a la docencia, el aprendizaje y la creación, difusión y transferencia del conocimiento. Además, velaré por la integridad de las personas y cuidaré los bienes de la Universidad."

En particular, se espera que mantengan altos estándares de honestidad académica. Cualquier acto deshonesto o fraude académico está prohibido; los alumnos que incurran en este tipo de acciones se exponen a un procedimiento sumario. Ejemplos de actos deshonestos son la copia, el uso de material o equipos no permitidos en las evaluaciones, el plagio, o la falsificación de identidad, entre otros. Específicamente, para los cursos del Departamento de Ciencia de la Computación, rige obligatoriamente la siguiente política de integridad académica en relación a copia y plagio: Todo trabajo presentado por un alumno (grupo) para los efectos de la evaluación de un curso debe ser hecho individualmente por el alumno (grupo), sin apoyo en material de terceros. Si un alumno (grupo) copia un trabajo, se le calificará con nota 1.0 en dicha evaluación y dependiendo de la gravedad de sus acciones podrá tener un 1.0 en todo ese ítem de evaluaciones o un 1.1 en el curso. Además, los antecedentes serán enviados a la Dirección de Docencia de la Escuela de Ingeniería para evaluar posteriores sanciones en conjunto con la Universidad, las que pueden incluir un procedimiento sumario. Por "copia" o "plagio" se entiende incluir en el trabajo presentado como propio,

partes desarrolladas por otra persona. Está permitido usar material disponible públicamente, por ejemplo, libros o contenidos tomados de Internet, siempre y cuando se incluya la cita correspondiente.