INF-239 Bases de Datos
Tarea 1: SQL Server

Profesores: Ricardo Salas - Rodrigo Olavarría
Ayudante de Cátedra: Beatriz Vásquez
Ayudantes de Tareas: Débora Alayo - Sebastián Castro
Salvador Fuentes - Daniela Sánchez - Isidora Ubilla

20 de marzo, 2023

1. Contextualización

Spot-Usm es un prestigioso proyecto de una empresa reconocida de streaming musical. Para ello, la compañía se ha encargado de recopilar información sobre los artistas y canciones más taquilleras de los últimos años para ser incorporados a la aplicación, en este contexto, se les solicita realizar una aplicación que simule la interacción de un usuario con Spot-Usm, permitiéndole obtener información según lo solicitado, marcar canciones como favoritas, entre otras características que se detallarán en los siguientes items. Para esto, deberá utilizar una conexión a SQL Server Express.

2. Descripción del problema

Para el desarrollo de esta tarea se le solicita que simule la interacción de un usuario con Spot-Usm. Para eso se le entregará un dataset **song.csv** con la siguiente información:

- **position** - Posición actual de la canción. [int]
- **artist_name** - Nombre del artista que produjo la canción. [str]
- **song_name** - Nombre de la canción. [str]
- **days** - Número de días desde el lanzamiento de la canción. [int]
- **top_10** - Número de veces que estuvo en el top 10. [int]
- **peak_position** - Posición más alta, alcanzada dentro del top 10. [int]
- **peak_position_time** - Número de veces en posición más alta, alcanzada dentro del top 10, considerando solo los 3 primeros puestos. [str]
- **peak_streams** - Total de reproducciones en el top 10, en la posición más alta que alcanzo. [int]
- **total_streams** - Total de reproducciones totales. [int]

Con la información de dicho csv, debe crear la tabla **repositorio_musica** con todas las columnas y datos que presenta, ademas debe agregar:

- **id** - PK, Identificador de la canción [int]

También debe crear la tabla **reproduccion**, que contiene las siguientes columnas:

- **id** - PK, Identificador de la canción [int]
- **song_name** - Nombre de la canción. [str]
- **artist_name** - Nombre del artista que produjo la canción. [str]
- **fecha_reproduccion** - Fecha en la que se reprodujo la canción por primera vez. [date]
- **cant_reproducciones** - Cantidad de veces que usuario ha reproducido la canción [int]
- **favorito** - Indica si es favorito o no la canción. [bit]

Por último se deben crear las tablas **lista_favoritos** que contienen las siguientes columnas:

- **id** - PK, Identificador de la canción [int]
- **song_name** - Nombre de la canción. [str]
- **artist_name** - Nombre del artista que produjo la canción. [str]
- **fecha_agregada** - Fecha en la que se agrego la canción a la lista de favoritos. [date]

3. To Do List

- Se solicita establecer una conexión entre python y su base de datos, para poder trabajar a través de queries desde el archivo .py
- Se solicita que realice una aplicación que simule la interacción entre un usuario y Spot-USM, que se debe presentar por medio de consola a través de un archivo python.
- Debe cargar el archivo csv en la base de datos con los criterios de la tabla repositorio_musica (hint: el id debe ser autoincremental).
- Crear la tabla Reproducción (para un solo usuario), que guarde la información de la primera vez que fue reproducida una canción (Fecha), la cantidad de veces que el usuario ha reproducido esa canción, y si es favorita o no.
- Se debe crear una lista de favoritos (para un solo usuario), en la cual el usuario podrá agregar y quitar canciones.
- Su aplicación debe mostrar un menú de navegación que permita realizar las siguientes operaciones:
  1. Mostrar Reproducción: Mostrar todas las canciones que actualmente están en la tabla Reproducción, permitiendo ordenar por fecha o por cantidad de veces reproducida. Se debe solicitar al usuario el orden deseado al momento de seleccionar esta opción.
  2. Mostrar las canciones favoritas del usuario.
  3. Hacer favorita una canción
  4. Sacar de favoritas a una canción.
  5. Reproducir una canción: Simular la reproducción de una canción (basta con mostrar por consola un mensaje). Al momento de seleccionar esta opción se debe solicitar el nombre de la canción a escuchar.
  6. Buscar una canción en la tabla Reproducción: mostrar la información de la canción elegida, disponible en la tabla Reproducción (nombre del artista, cantidad de veces reproducida, fecha de la primera reproducción). Se debe solicitar el nombre de la canción para realizar la búsqueda.
  7. Mostrar todas las canciones que el usuario haya escuchado por primera vez en los últimos X días (con X variable). Se debe solicitar la fecha al momento de seleccionar esta opción.

8. Buscar por nombre de canción y por artista: Se debe poder ingresar el nombre de una canción y mostrar al artista (si existe), Si varios artistas han escrito canciones con ese nombre, debe mostrarlos todos. También al ingresar un artista, se deben mostrar todas sus canciones.
9. Top 15 artistas con mayor cantidad total de veces en que sus canciones han estado en el top 10: Esto es, la suma de las veces en que sus canciones han estado en el top, por ejemplo, si un artista determinado tiene solo 2 canciones y cada canción ha llegado 8 y 9 veces al top 10 respectivamente, la cantidad total de veces que dicho artista llegó al top 10 con sus canciones es 17.
10. Peak position de un artista: Dado un artista, se debe mostrar la posición más alta obtenida entre todas sus canciones (deben fijarse en la columna peak position) sin utilizar una query que retorne más de un valor para luego seleccionar solo el primer resultado.
11. Promedio de streams totales: dado un artista, se debe retornar el promedio de los streams considerando todas sus canciones

- Se debe hacer una función (SQL, no de python) que se utilice en cualquiera de los puntos pedidos en el to-do list (y que sea útil).
- Se debe crear una **view** a elección, debe ser útil (y utilizada) acorde al contexto de la tarea.
- Su aplicación debe ser capaz de gestionar eventuales errores al momento de interactuar con el menú (errores en el ingreso de datos por consola).
- **ACLARACIONES:** no se darán puntos extra por realizar una interfaz gráfica, toda interacción debe realizarse por medio de la consola de Python. Además, si bien se trata de una plataforma de música no se deben incorporar pistas de audio a su programa, si se le solicita reproducir una canción mostrar por consola un mensaje es suficiente.

4. Especificaciones y reglas

El desarrollo de esta tarea debe cumplir las siguientes especificaciones, de lo contrario podría existir un descuento en la nota final:

- **Toda interacción con la base de datos debe ser mediante Querys en Python, inclusive el cargar los datos a la tabla repositorio_musica.**
- **El código debe ser realizado en Python utilizando la librería pyodbc** .
- **Debe** realizarse en parejas, no se aceptarán tareas individuales.
- En el foro de consultas podrán buscar pareja quienes no tengan, esto es exclusiva responsabilidad del estudiante.
- En caso de problemas con su pareja deberán contactar al profesor explicando su situación, (hacerlo con anticipación).
- La tarea debe ser entregada como un archivo .ZIP comprimido de la forma T1_ROL1_ROL2, este debe contener los archivos **.py** necesarios para el funcionamiento de su programa, además de un archivo README.txt el cual debe contener nombre, rol de los alumnos y las instrucciones para la correcta ejecución de su programa.
- Solo un alumno debe realizar la entrega.
- La entrega será vía AULA y el plazo máximo de entrega es hasta el **sábado 15 de abril de 2023 a las 23:55**.
- Las entregas con hasta 5 min. de atraso no tendrán descuento, posterior a las 00:00 existirá un descuento de 10 puntos por cada hora o fracción.
- Las funciones deberán ir comentadas, explicando clara y brevemente lo que realiza, los parámetros que recibe y el return en caso de ser necesario.
- Las copias serán evaluadas con nota 0 y se informarán a las respectivas autoridades.
- En caso que falle la ejecución de algún comando, no se asignará puntaje a éste.

- Las consultas sobre la tarea se deben hacer en las instancias de laboratorios. Consultas puntuales que no se hayan hecho en los laboratorios podrán hacerse a través de un foro creado para eso. Se responderán consultas hechas hasta 48 horas antes de la entrega, es decir, hasta el **jueves 13 de abril de 2023 a las 23:55**.
- Los laboratorios son instancia donde se explica y resuelven dudas sobre el enunciado de la tarea y además se hace una introducción a SQL.
- Ante cualquier duda sobre algo que no aparezca especificado en la tarea y no afecte las reglas ni la dificultad de esta, puede asumir lo que estime conveniente **pero debe ser especificado en el README.txt, en caso de que no se haga esto habrá descuento**.
- Esta evaluación debe ser defendida a través del servidor de Discord (el link se compartirá por aula los días previos a las defensas). Quienes no defiendan tendrán nota 0 en su tarea 1.
- Existe la posibilidad de que a su defensa, asista su profesor y realice preguntas.
- Cada grupo tendrá un horario definido para su defensa, en caso de atraso contarán con un tiempo menor para presentar su trabajo.
- La información respecto a la defensa será eventualmente publicada en Aula, esto considera el detalle sobre los descuentos. Es su obligación estar atento a esta información y cumplir con lo establecido allí.
- Durante las defensas se les preguntará sobre el código entregado, debiendo responder correctamente para poder optar al puntaje completo. Se le pedirá que explique su razonamiento detrás de cada consulta SQL para verificar su comprensión de la tarea.
- El incumplimiento de cualquier punto expuesto aquí, podría implicar un descuento en la nota final de la tarea 1.