**UNIVERSIDAD DE SANTIAGO DE CHILE**
**FACULTAD DE INGENIERÍA**
**DEPARTAMENTO DE INGENIERÍA INFORMÁTICA**


# Taller 1: Aplicación LPO para sugerir géneros musicales utilizando Python y Prolog

Integrantes: Daniel Frez
Matías Figueroa
Javier Sanhueza
John Serrano
Curso: Teoría de la Computación
Profesor: Daniel Vega
Ayudante: Matias Tobar

13 de Junio de 2022

# Tabla de contenidos

# 1. Introducción

Desde los tiempos de las antiguas civilicaciones, la musica ha ido acompañando al ser humano, evoluiconando junto a él. Dada la importancia de la música históricamente, y el crecimiento en el consumo de la música, se llegó a tener en la actualidad una gran variedad de canciones y géneros musicales de diversos orígenes y culturas. Esta gran variedad trajo consigo que las personas destinarán tiempo para buscar géneros musicales, canciones y artistas de su gusto. Sin embargo, este proceso puede ser tedioso. Para resolver esto y facilitar la búsqueda, este taller tiene como objetivo principal proponer una aplicación que sugiera géneros musicales y canciones, esto utilizando la base teórica de la **Lógica de Primer Orden** aplicando Prolog y Python con la librería PySwip. Para lograr sugerir géneros de manera precisa, se tienen como objetivos:

1. Obtener entradas que ayuden a filtrar las canciones y géneros en base a sus características, esto mediante preguntas claves sobre las preferencias del usuario, y así generar relaciones con las condiciones dadas para llegar a una sugerencia precisa.

2. Implementar una interfaz gráfica que permita al usuario ingresar sus preferencias y visualizar el resultado con la sugerencia de las correspondientes canciones y géneros musicales.

# 2. Desarrollo

## 2.1. Canciones y base de conocimiento

El programa es desarrollado en el lenguaje de programación **Python** version 3.10, utilizando la librería **PySwip** version 0.2.10, lo cual permite realizar consultas y cargar la base de conocimientos utilizando un intérprete de Prolog, similar a **Swi-Prolog**. En la base de conocimientos, se tienen hechos que representan la relación de una canción. Una canción de la base de conocimientos debe poseer un nombre, un género, un reconocimiento del artista de la canción (es decir, si es conocido o no tan conocido), una duración de la canción, la década de la canción, si el artista de la canción es un solista o una banda / grupo y una característica extra. Con lo anterior en mente, se llenó la base de conocimientos con muchos hechos que representan canciones, como se puede apreciar en la siguiente imagen:

```
% ------------------------------------------------------POP-------------------------------------------------
song("Maroon 5- She will be loved", pop, exponente, larga, dosmil, banda, artesania_diversos_elementos).
song("Michael Jackson - Beat it", pop, exponente, larga, ochentas, solista, artesania_diversos_elementos).
song("The Weeknd - Blinding Lights", pop, exponente, normal, moderna, solista, artesania_diversos_elementos).
song("Hunting high and low - Take on me", pop, exponente, normal, ochentas, banda, artesania_diversos_elementos).
song("The Chainsmokers - Closer", pop, exponente, larga, moderna, banda, artesania_diversos_elementos).
song("Christian Kuria - Losing You", pop, underground, larga, moderna, solista, artesania_diversos_elementos).
song("Michael Jackson - Billie Jean", pop, exponente, larga, ochentas, solista, artesania_diversos_elementos).
song("Frank Blunt - The Dirt Road", pop, underground, corta, moderna, solista, artesania_diversos_elementos).
song("BTS - Butter", pop, exponente, corta, moderna, banda, artesania_diversos_elementos).
song("Kairi Yagi (Vivy) - Sing My Pleasure", pop, underground, larga, moderna, solista, artesania_diversos_elementos).
song("Yellowcard - Way Way", pop, exponente, normal, moderna, banda, artesania_diversos_elementos).
% ------------------------------------------------------METAL-----------------------------------------------
song("Famous Last Words - One in the chamber", metal, underground, normal, moderna, banda, guitarras_fuertes_distorcionadas).
song("Beyond Borders - Absolute Zero", metal, underground, normal, moderna, banda, guitarras_fuertes_distorcionadas).
song("Pellek - Brokenbow", metal, exponente, normal, moderna, solista, guitarras_fuertes_distorcionadas).
song("DragonForce - Through The Fire And Flames", metal, exponente, dosmil, moderna, banda, guitarras_fuertes_distorcionadas).
song("Babymetal - Gimme Chocolate!!", metal, underground, normal, moderna, banda, guitarras_fuertes_distorcionadas).
song("Metallica - Battery", metal, exponente, larga, ochentas, banda, guitarras_fuertes_distorcionadas).
song("Iron Maiden - Run to the hills", metal, exponente, normal, ochentas, banda, guitarras_fuertes_distorcionadas).
song("Famous Last Words - Welcome To The Show", metal, underground, normal, moderna, banda, guitarras_fuertes_distorcionadas).
song("The Amity Affliction - Pittsburgh", metal, exponente, normal, moderna, banda, guitarras_fuertes_distorcionadas).
song("Machine Head - Be Still And Know", metal, exponente, larga, moderna, banda, guitarras_fuertes_distorcionadas).
song("Scorpions - Rock You Like a Hurricane", metal, exponente, larga, ochentas, banda, guitarras_fuertes_distorcionadas).
% ------------------------------------------------------DISCO-----------------------------------------------
song("Michael Jackson - Thriller", disco, exponente, larga, ochentas, solista, querer_bailar).
song("Bee Gees - Stayin Alive", disco, exponente, larga, setentas, banda, querer_bailar).
song("Earth, Wind & Fire, The Emotions - Boogie Wonderland", disco, exponente, larga, setentas, banda, querer_bailar).
song("Bob Sinclar, Pitbull, Dragonfly, Fatman Scoop - Rock the Boat", disco, underground, normal, moderna, banda, querer_bailar).
song("Boney M. - Rasputin", disco, exponente, media, setentas, banda, querer_bailar).
```

*Figura 1: Canciones en la base de conocimiento.*

Para llenar la base de conocimiento se utilizaron páginas web sobre música y la aplicación Spotify. De esta forma, se pudo obtener el género, duración y otras características más para cada canción.

Las opciones para las características de una canción de la base de conocimiento son las siguientes:

- El artista es Exponente (conocido) o Underground (no tan conocido).
- La duración de la canción es corta (menos de 3 minutos), normal (entre 3 y 4 minutos) o larga (más de 4 minutos)
- Las décadas son: moderna (desde el 2010 hasta la actualidad), 2000s, 90s, 80s, 70s o antigua (60s o antes)
- Un artista puede ser cantante / solista o puede ser un grupo o banda

Para las características extras, se tiene una opción para cada género. Se consideran 16 géneros para los cuales se tiene como mínimo 6 canciones, estos géneros junto a la característica extra que los representa son:

1. **Pop** ("Que tenga preferencia por la artesanía y diversos elementos")
2. **Metal** ("Que tenga guitarras fuertes y distorsionadas")
3. **Disco** ("Que provoque querer bailar")
4. **Rock** ("Que tenga una letra en tono medio y una guitarra eléctrica")
5. **Jazz** ("Que tenga sonidos de saxophone")
6. **Hip-Hop** ("Que contenga una letra hablada rápidamente")
7. **Electrónica** ("Que tenga sonidos electrónicos")
8. **Cumbia** ("Que tenga ritmos bastante movidos")
9. **Clásica** ("Que tenga sinfonía y tengan poca o nula improvisación")
10. **Country** ("Que emplee armonías vocales")
11. **Techno** ("Que tenga tempo rápido y sonidos distorsionados")
12. **Reggae** ("Que tenga sonidos de una batería rítmica")
13. **Salsa** ("Que use el patrón rítmico del son cubano")

14. **Romantica** ("Que tenga letras de amor")

15. **Rock n Roll** ("Que tenga un ritmo de música bailable y frenético")

16. **Blues** ("Que tenga tecnicas expresivas de la guitarra y harmonia")

Los hechos de la base de conocimiento son guardados en un archivo llamado **"base.pl"**. Esta base es cargada en el archivo principal **"taller1.py"** donde se utiliza el método **"prolog.consult("base.pl")"**. De esta forma, se pueden utilizar todos los hechos con el intérprete de prolog proporcionado por la librería importada.

## 2.2. Funciones e Interfaz Gráfica

Para el desarrollo del taller, se utilizan diferentes funciones, las cuales son:

- **reconocer_caracteristica:** Función que permite reconocer la característica extra de una música y entregar la característica de forma abreviada para poder buscarla en la base de conocimientos. Recibe un string correspondiente a una característica extra de una música y retorna un string de la característica en forma abreviada.

- **reconocer_tipo_artista:** Función que permite saber el reconocimiento de un artista, es decir, si es exponente o underground. Recibe un string correspondiente al reconocimiento del artista y retorna un string que permite buscar la característica en la base de conocimientos.

- **reconocer_decada:** Función que permite reconocer la época de una música. Recibe un string correspondiente a la época de una música y retorna un string que permite buscar la característica en la base de conocimientos.

- **reconocer_duracion:** Función que permite reconocer la duración de una música. Entrega un string que permite buscar la característica en la base de conocimientos. Recibe un string correspondiente a la duración de una música y retorna un string de la duración en forma abreviada.

- **reconocer_solista_banda:** Función que permite reconocer si el artista es un solista o una banda, entrega un string que es utilizado para buscar la característica en la base de conocimientos. Recibe un string que representa si el artista es un solista o una banda y retorna el mismo string, pero en minúsculas.

- **recivir:** Función principal utilizada al momento de utilizar el botón de "Buscar canciones" de la interfaz gráfica. Permite realizar el proceso de búsqueda en la base de conocimientos y escribir los resultados encontrados en la interfaz gráfica. No tiene entrada y debido a que solo escribe cosas en la interfaz gráfica, no entrega un valor de salida.

- **songQuery:** Función que permite buscar en la base de conocimientos las canciones que se acomoden a los parámetros entregados. Recibe cinco strings, donde el primero representa si el artista es un exponente o underground, el segundo la duración de la canción, el segundo la duración de la canción, el tercero la década de la canción, el cuarto si es que el artista es un solista o una banda, y el quinto la característica extra. Retorna una lista de todas aquellas canciones que cumplan con las características entregadas.

- **generoMasRepetido:** Función que permite determinar el género más repetido en la lista de respuestas. Recibe una lista de respuestas de músicas y retorna un string que representa el género más repetido en la lista de respuestas.

- **gustosSimilares:** Función que permite determinar las canciones que se acomoden a los gustos del usuario, buscando músicas con características similares a las entregadas. Recibe una lista de datos de músicas y retorna una lista que contiene las músicas que se acomoden a los gustos del usuario.

Utilizando la librería **tkinter** se desarrolla una interfaz gráfica (GUI), la cual permite que un usuario pueda seleccionar sus preferencias respondiendo preguntas respecto a las características anteriormente mencionadas de una canción. El programa se encarga de buscar cuales son las canciones que coinciden con sus preferencias (O alternativamente, que se parecen a sus preferencias) y mostrarlas a través de la interfaz gráfica.

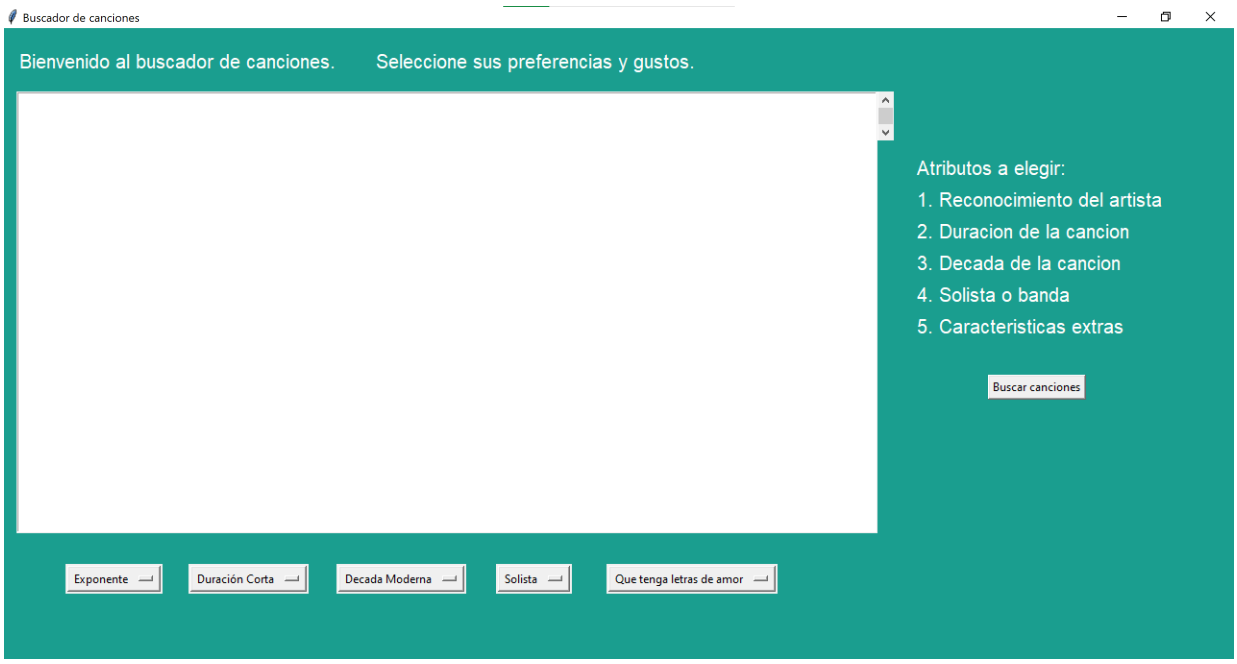


*Figura 2: Interfaz gráfica desarrollada.*

## 2.3. Explicación del código

Respecto al código de taller1.py, en el bloque principal se tienen cinco listas de opciones, las cuales son las opciones disponibles a elegir como respuestas a las cinco preguntas que aparecen en la interfaz gráfica. Se tiene además, una lista general, que contiene las cinco listas mencionadas anteriormente y una lista de entradas actualmente vacía, ya que ahí es donde se van a guardar las opciones seleccionadas por el usuario. Se invoca a Prolog utilizando el método "Prolog()" y se guarda en una variable del mismo nombre.

```
232
233 # BLOQUE PRINCIPAL
234 #------------------------------------------------------------------------
235
236 #Listas de opciones a elegir en la interfaz grafica
237 opciones1 = ["Exponente","Underground"] # Reconocimiento Artista
238 opciones2 = ["Duración Corta","Duración Normal","Duración Larga"] # Duracion
239 opciones3 = ["Decada Moderna","Los 2000s","Los 90s","Los 80s","Los 70s","Clasica"] # Decada
240 opciones4 = ["Solista","Banda"] # Solitario o Banda
241 opciones5 = ["Que tenga letras de amor" # Caracteristicas extras
242              ,"Que tenga guitarras fuertes y distorcionadas"
243              ,"Que tenga una preferencia por la artesania y diversos elementos"
244              ,"Que tenga sonidos electronicos"
245              ,"Que provoque querer bailar"
246              ,"Que contenga una letra hablada rapidamente"
247              ,"Que tenga una letra en tono medio y una guitarra electronica"
248              ,"Que tenga sonidos de saxophone"
249              ,"Que tenga sonidos de una bateria ritmica"
250              ,"Que tenga ritmos bastante movidos"
251              ,"Que tenga sinfonia y tengan poca o nula improvisacion"
252              ,"Que tenga tempo rapido y sonidos distorcionados"
253              ,"Que use el patron ritmico del son cubano"
254              ,"Que tenga un ritmo de musica bailable y frenetico"
255              ,"Que tenga tecnicas expresivas de la guitarra y harmonia"
256              ,"Que emplee armonias vocales"
257              ,"Ninguna de las anteriores"]
258 #Lista que contiene las opciones
259 lista_suprema = [opciones1, opciones2, opciones3, opciones4, opciones5]
260 lista_de_entradas = [] # Lista que contiene las entradas de los usuarios
```

*Figura 3: Bloque de listas a utilizar.*

Luego, se inicializa la interfaz gráfica, utilizando el método "Tk()" y se configuran distintos aspectos de esta. Se agrega un listbox, los distintos selectores y opciones a utilizar, textos y un botón. Se asigna la función recivir() al momento de presionar el botón. Esta función, como se describió anteriormente, permite realizar el proceso de búsqueda en la base de conocimientos y escribir los resultados encontrados en la interfaz gráfica. Para ello, guarda todas las opciones elegidas y las guarda en lista de salida. Luego por cada caracteristica va utilizando las funciones reconocer_tipo_artista(), reconocer_duracion(), reconocer_decada(), reconocer_solista_banda() y reconocer_caracteristica() para obtener los strings correspondientes para buscar en la base de conocimiento. Luego se utiliza la función songQuery() para realizar la búsqueda en la base de conocimientos y la lista retornada es guardada en la variable "resultados". Si resultados.[es]tá vacía, eso implica que no se encontró ninguna canción que tenga exactamente las mismas características solicitadas por el usuario y eso implica que el programa ahora debe buscar canciones con características similares, donde uno de los atributos será distinto al solicitado. En ese caso, se utiliza la función gustosSimilares() y se asigna una variable que sirve para mostrar un mensaje alternativo en la interfaz cuando no se encuentran canciones.

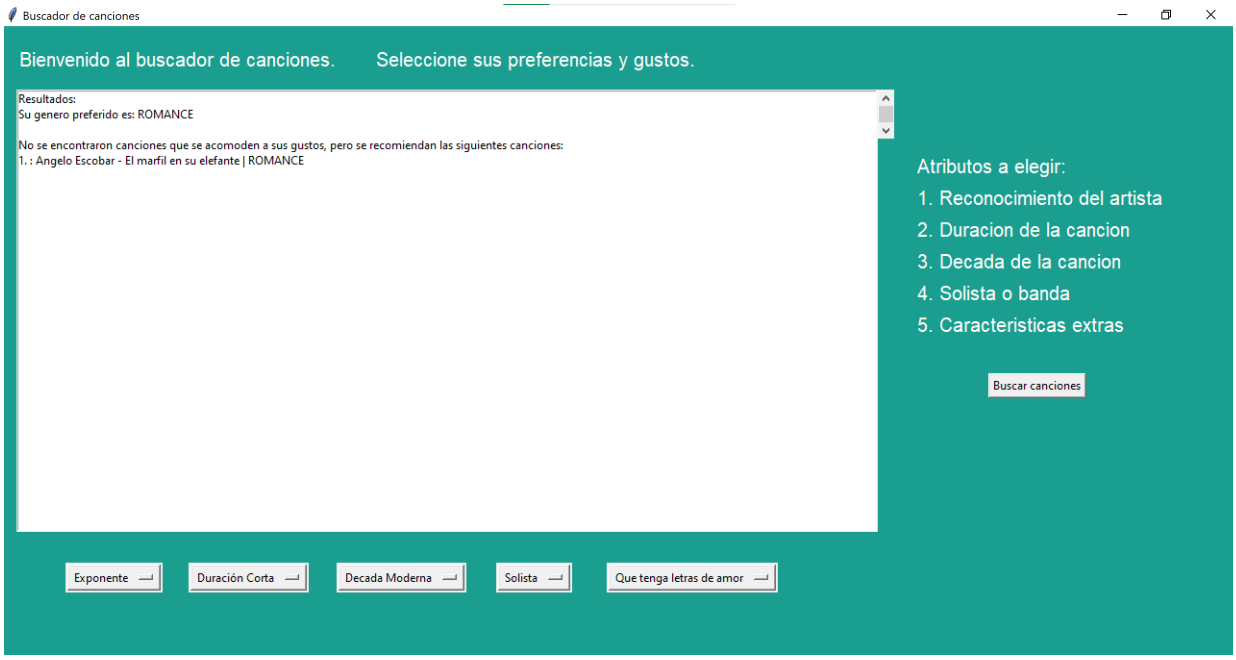


*Figura 4: Mensaje alternativo mostrado en la interfaz.*

Luego, ya sea porque se encontraron canciones con características exactas o similares, se determina el género preferido por el usuario utilizando la función generoMasRepetido() y este se muestra en la interfaz. Posteriormente se muestran todas las canciones encontradas junto con su respectivo género de cada una.

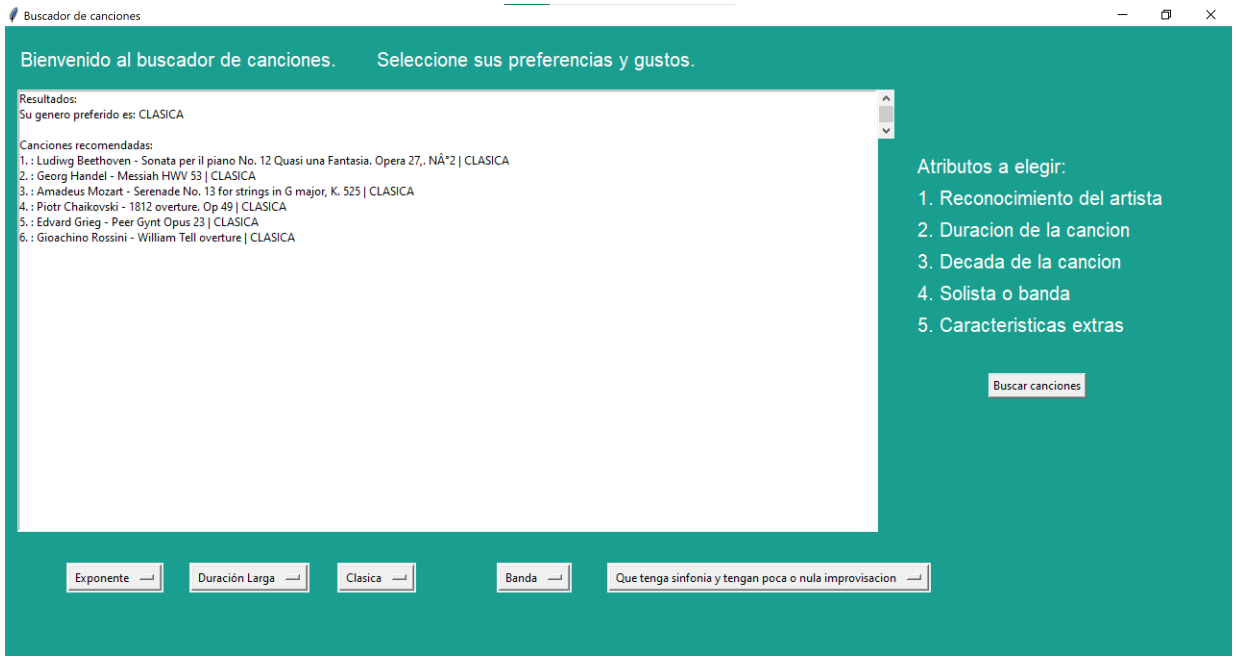


*Figura 5: Busqueda exitosa de canciones con las preferencias escogidas.*

El usuario puede volver a realizar la búsqueda de canciones con otras características, probando distintas combinaciones si así lo desea, o bien, puede cerrar la ventana, finalizando la ejecución de todo el programa.

# 3. Análisis de resultados

Con la funcionalidad del programa explicada en la sección anterior, ahora se puede analizar los resultados obtenidos y explicar los diferentes resultados que el programa les puede entregar al responder las preguntas encuestadas. Los resultados se dividen en casos dependiendo de ciertas condiciones, las cuales se describen a continuación:

## 3.1. Caso 1

Este caso es presentado cuando el usuario le entrega toda la información pedida por el programa obviando la respuesta "Ninguna de las anteriores" cuando se le pregunta de alguna preferencia en especial. El resultado de esta va depender de si es que el programa encuentra canciones que coincidan con todas las características entregadas o no en la base de conocimientos, pero estas tendrán algo en común, el cual es el mismo género musical entre las canciones que se le entregarán al usuario.

- **Coinciden las respuestas:** Cuando el programa encuentra que todas las características son iguales a una de las canciones almacenadas en la base de datos que se le está provisionando, este entregará el primer resultado que este encuentra señalándole al usuario cual es el género musical recomendado, el nombre de la canción encontrada y su género.

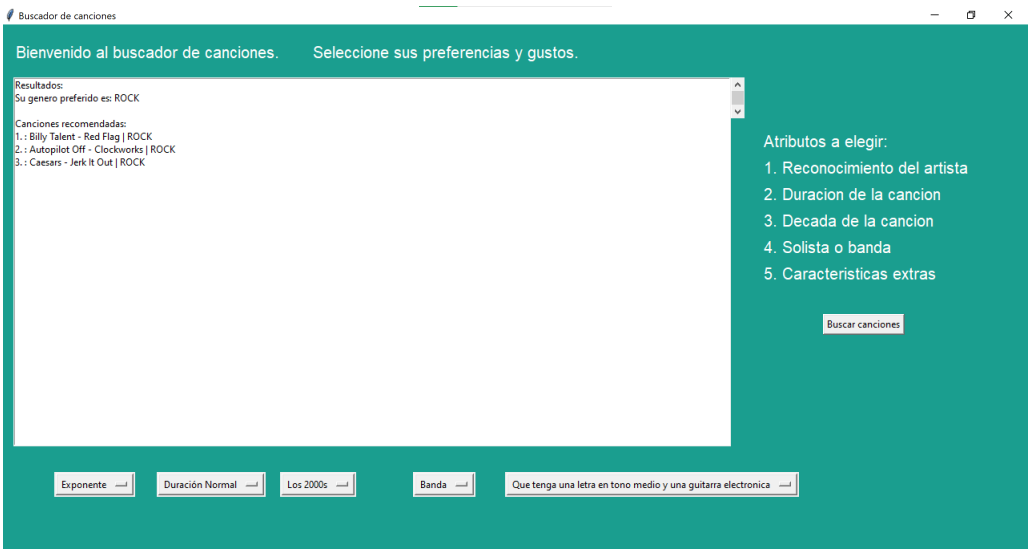


*Figura 6: Caso 1 - Coinciden las respuestas.*

- **No coinciden las respuestas:** Cuando las respuestas entregadas por el usuario en la encuesta, no coinciden en su totalidad a alguna de las canciones en la base de conocimientos entregada al programa, este obviara alguna parte de las respuestas que no coincide y le entregará al usuario canciones que coinciden con la mayoría de las especificaciones entregadas en la encuesta del programa, retornando así el género recomendado por el programa con los nombres de las canciones y su género.

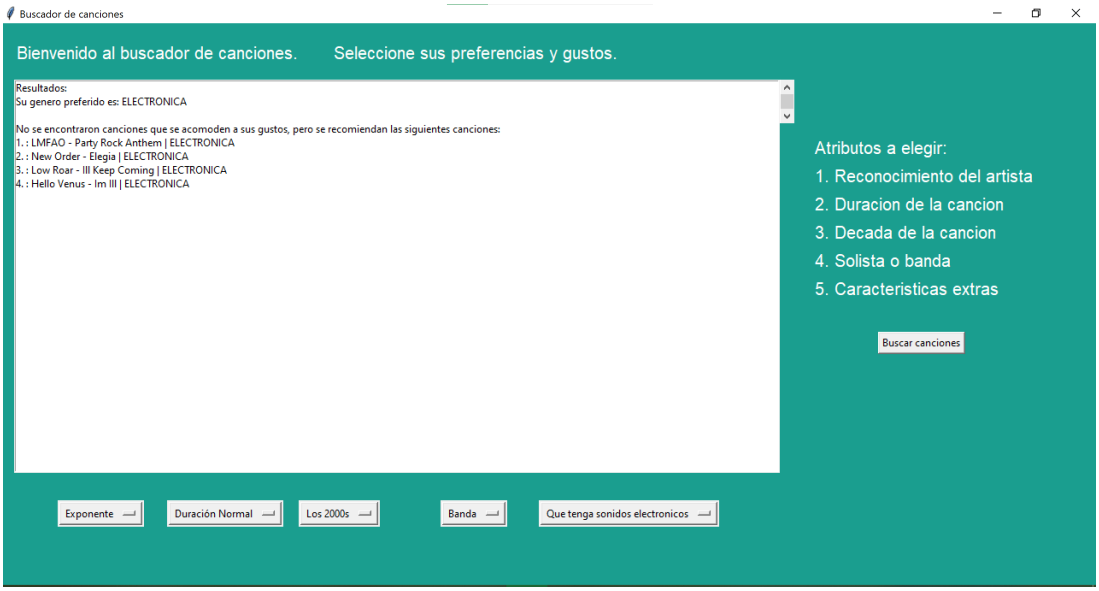


*Figura 7: Caso 1 - No coinciden las respuestas.*

## 3.2. Caso 2

Pasando al caso dos, este sucede cuando el usuario le entrega la mayoria de las respuesta pedidas en el formulario del programa, pero en el caso de la pregunta "Características Extra" responde con la respuesta "Ninguna de las anteriores " causando que este tenga una respuesta distinta a la del caso uno.

Cuando este caso ocurre el programa buscará todas las canciones almacenadas en la base de datos proporcionada, que tengan las mismas características entregadas por el usuario aun siendo que sean de distintos géneros musicales. Cuando el programa encuentra todas las canciones, este buscará cuál es el género más común entre las respuestas, para decidir cuál es el género que le recomendaran al usuario para luego entregar las canciones encontradas con su nombre y género.


*Figura 8: Caso 2 - El usuario elige "Ninguna de las anteriores".*

Las características extras son las que están encargadas de determinar el género de la canción, ya que estas tienen relación con distintos elementos de un género en específico, es por esto que cuando se elige la opción de "Ninguna de las anteriores", el programa no tendrá indicio de cual es el género que se prefiere y por lo tanto, solo deberá buscar canciones que coincidan con todas las demás características.

## 3.3. Relación con la teoría

La lógica es un método o razonamiento en el que las ideas se manifiestan o se desarrollan de forma coherente y sin que haya contradicciones entre ellas. Existen distintos tipos de lógica y en el caso de este taller se utiliza la Lógica de Primer Orden (LPO).

La Lógica de Primer Orden puede expresar cualquier cosa que pueda ser programada. La LPO tiene sentencias como lógica proposicional y, además, tiene términos, que representan objetos. Para construir términos se usan símbolos constantes, variables y funciones, cuantificadores y símbolos. En el caso del taller, el objeto de entrada son las canciones con sus diferentes características que son representadas como hechos de una base de conocimiento, y es analizada a través de las distintas consultas que realiza el programa. Además, prolog tiene sus raíces en la LPO, por lo que se podría decir que el programa está altamente ligado a la LPO. **(2. Autor Desconocido, 2018)**

# 4. Conclusión

Se concluye el cumplimiento de los objetivos planteados, ya que se logró obtener información por parte del usuario sobre sus gustos musicales, y utilizar estos datos para generar sugerencias precisas de canciones utilizando relaciones con la base de conocimientos.

También, se logró aplicar la lógica del programa a través de una interfaz gráfica creada con la librería tkinter en la que el usuario entrega la información requerida y visualiza las sugerencias provistas por el sistema.

Cabe destacar, que una de las principales limitaciones del programa es el hecho de que depende de una base de conocimiento. Esto implica, que al tener 16 géneros de música es muy difícil tener a lo menos una canción para cada combinación de atributos en la base de conocimientos. Sin embargo, para contrarrestar ese problema existe el caso alternativo, lo que de igual forma permite que el usuario obtenga un resultado asegurado. Otra limitación, es que las opciones a elegir son solamente opciones especificas y solo se puede elegir una por atributo, por lo que puede que no se ajusten a lo que el usuario desea o puede que el usuario no sepa que es lo que desea buscar.

El código creado aplica el paradigma de programación lógico a través del uso de los elementos de Prolog. Este programa se encarga primeramente de identificar y procesar los datos entregados por el usuario y, posteriormente, mostrar los resultados en la interfaz gráfica donde el usuario verá las canciones recomendadas según sea el caso. Una posible mejora para este proyecto sería conocer más detalles del usuario sobre sus gustos musicales, por ejemplo, saber si prefiere escuchar la voz de un hombre o una mujer, el idioma de la canción, los niveles de volumen, entre otros elementos. Además, las sugerencias podrían añadir por qué le podría gustar la canción mencionada y recomendarle un género musical similar a su preferencia.

Se espera que el conocimiento obtenido en este taller sirva como experiencia para el taller N°2 y futuros trabajos donde se deba aplicar la LPO o Prolog.

# 5. Bibliografía y Referencias

[**1**] Multiples Autores. (Septiembre 2013). “El lenguaje de programación PROLOG”. Libro online. Recuperado de: https://drive.google.com/file/d/1DgxI64oAB5do7AajCXCTphnGWHTCEmk/view?usp=sharing

[**2**] Autor Desconocido. (2018). ”First-Order Logic (FOL)”. Presentación Online. Recuperado de: https://homepage.cs.uri.edu/faculty/hamel/courses/2018/fall2018/csc501/lecture-notes/prolog-tutorial.pdf

[**3**] Standford University. (2019). ”First-Order Logic Part One”. Presentación Online. Recuperado de: http://web.stanford.edu/class/archive/cs/cs103/cs103.1202/lectures/04/Slides04.pdf

# 6. Anexos

README.md

You, hace 1 segundo | 1 author (You)

```
Taller 1 - Grupo 3
Autores: Mastias Figueroa, Daniel Frez, Javier Sanhueza y John Serrano
Teoria de la Computacion 2022-1

Instrucciones de Uso:
1. Primero, se debe tener instalado la extensión "PySwip". En el caso de no tenerla, se debe abrir la
consola de comandos (CMD) si se esta utilizando Windows o la bash de terminales si se esta utilizando
Linux. Se debe escribir "pip install pyswip" y de esa forma, se instalará PySwip.
2. Luego, sedebe tener la carpeta src del código, donde dentro de ella deben estar tanto el archivo "base.pl"
y el archivo "taller1.py"
3. Se ejecuta el archivo "taller1.py". Si PySwip esta instalado correctamente entonces se abrirá la interfaz
gáfica
4. El siguiente paso es elegir las preferencias de música y luego apretar el boton "buscar canciones". El
programa mostrará los resultados encontrados, indicando si se encontraron resultados exactos o bien,
resultados similares.
5. Repetir el paso 5 hasta que se desee cerrar la interfaz gráfica, finalizando la ejecución del programa.
```

*Figura 9: Archivo README.md que contiene las instrucciones de ejecución del programa.*