Sistema de Música XPO

Mateus Medeiros dos Santos, Nikolas Eduardo Andreski Rodrigues

Sistemas de Informação
Universidade da Região de Joinville (UNIVILLE) – Joinville, SC – Brazil
mateus.medeiros@univille.br, nikolas.rodrigues@univille.br

1. Introdução

Nosso sistema imitará um software de músicas que terá planos e poderá fazer login para salvar os dados e playlists como músicas e álbuns curtidos(as), além de baixar os mesmos.

2. Requisitos Funcionais

As histórias abaixo apresentarão as funcionalidades do software como, criar um cadastro e fazer login, pesquisar músicas/artistas, salvar e baixar músicas, e criar playlists.

2.1. História de Usuário 01

Como um usuário do Sistema de música, eu quero criar o meu Cadastro. Preciso fornecer um email e senha para manter os dados salvos. Após isso tenho que selecionar uma assinatura, tendo gratuita ou paga. E para completar o perfil devo informar o meu nome completo, foto, curtir as músicas que gosto para o aplicativo buscar outras baseadas no meu interesse e configurar o tema que desejo, sendo claro ou escuro.

A Figura 01 é o UML da história de usuário 01. A entidade Usuário relaciona diretamente com a entidade Perfil sendo necessárias as informações como nome, foto, curtir algumas músicas e configurar o tema.

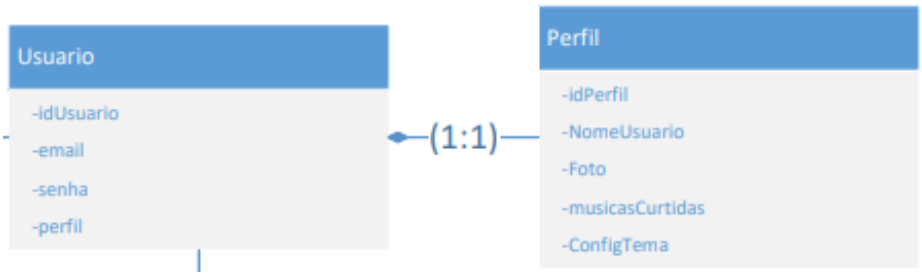

Figura 1. UML das entidades da História de Usuário 01.

A Figura 02 é o MER da história de usuário 01. A entidade Usuario contêm as chaves estrangeiras da entidade Perfil. A Entidade Usuario vai conter as informações de login e senha. Faz relação de 1:1 com a entidade Perfil onde tem as informações pessoais e compartilha o idPerfil com a entidade Usuario.

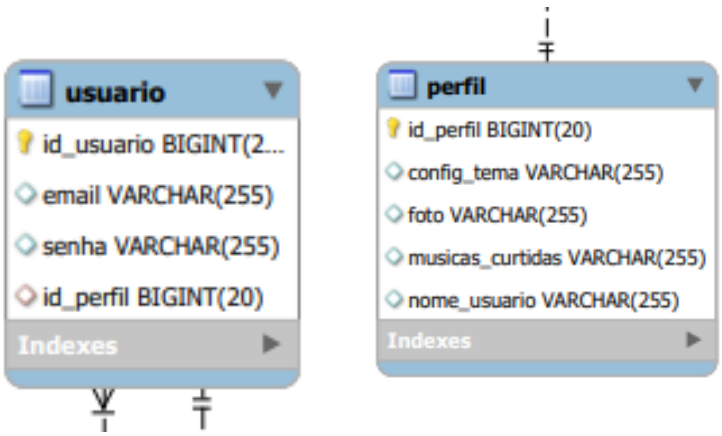

Figura 2. Modelo Entidade Relacionamento da História de Usuário 01.

2.2. História de Usuário 02

Como um usuário do Sistema de música, eu quero salvar um álbum. Para isso preciso pesquisar o artista que desejo e através das músicas eu escolho o álbum desejado e salvo no meu perfil.

A figura 03 é o UML da história de usuário 02. A entidade Artista relaciona diretamente com a entidade Musica onde pertence a um Álbum. São necessárias as informações do artista como o nome e do álbum como nome ou data de lançamento.

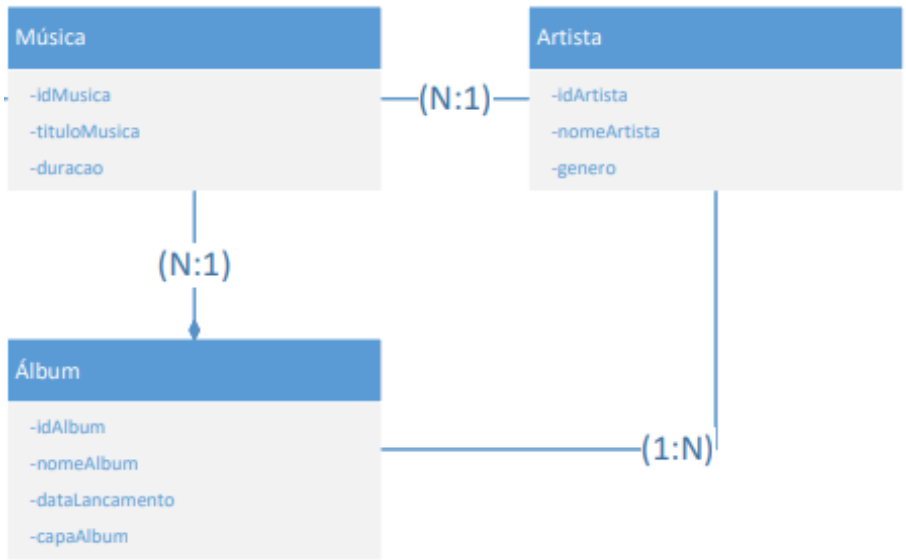

Figura 3. UML das entidades da História de Usuário 02.

A figura 04 é o MER da história de usuário 02. A entidade Artista contém as chaves estrangeiras da entidade Álbum. A entidade Artista vai conter as informações como nome, id e gênero de música. Faz relação 1:N com a entidade Álbum onde tem as informações de id do álbum, nome, data de lançamento e capa do álbum.

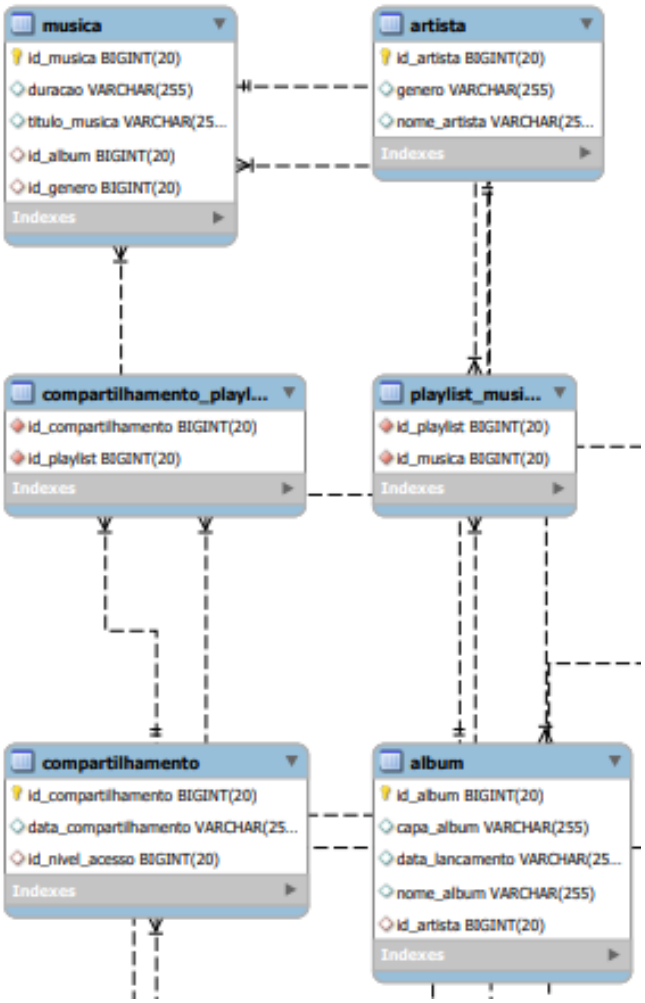

Figura 4. MER das entidades da História de Usuário 02.

2.3. História de Usuário 03

Como um usuário do Sistema de música, eu quero compartilhar uma playlists com amigos. Para isso preciso escolher a playlist e compartilhar com o usuário que desejo digitando o email dela.

A figura 05 é o UML da história de usuário 03. A entidade Playlist relaciona diretamente com a entidade Usuario. A entidade Usuario irá relacionar com a entidade Compartilhamento que irá relacionar com a entidade Nivel de Acesso onde vai ser dada as permissões necessárias para cada usuário que irá receber a playlist compartilhada. São necessárias as informações do nome da playlist, o email dos usuários que irão receber o compartilhamento, além do nível de acesso que cada usuário irá ter para a playlist que será compartilhada.

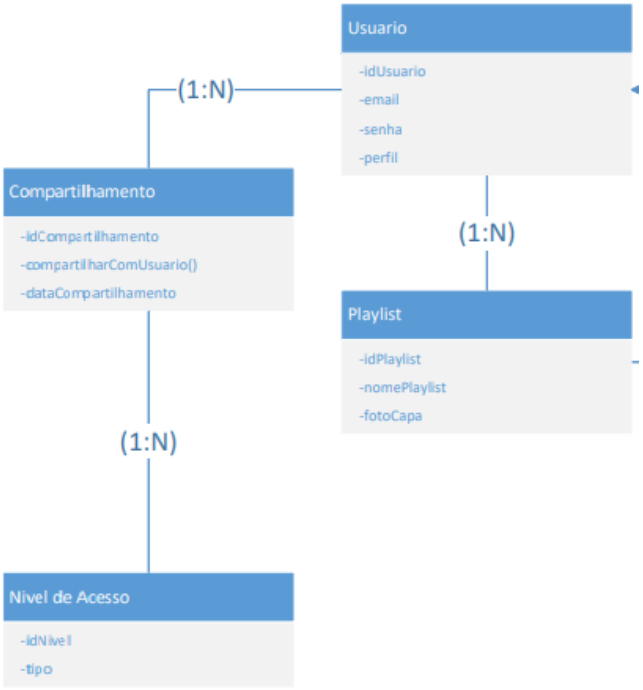

Figura 5. UML das entidades da História de Usuário 02.

A figura 06 é o MER da história de usuário 03. A entidade Usuario contém as chaves estrangeiras da entidade Compartilhamento. A entidade Usuario vai fornecer o email de quem vai receber a playlist. A entidade nível de acesso faz relação 1:N com a entidade Compartilhamento onde tem as informações do idCompartilhamento.

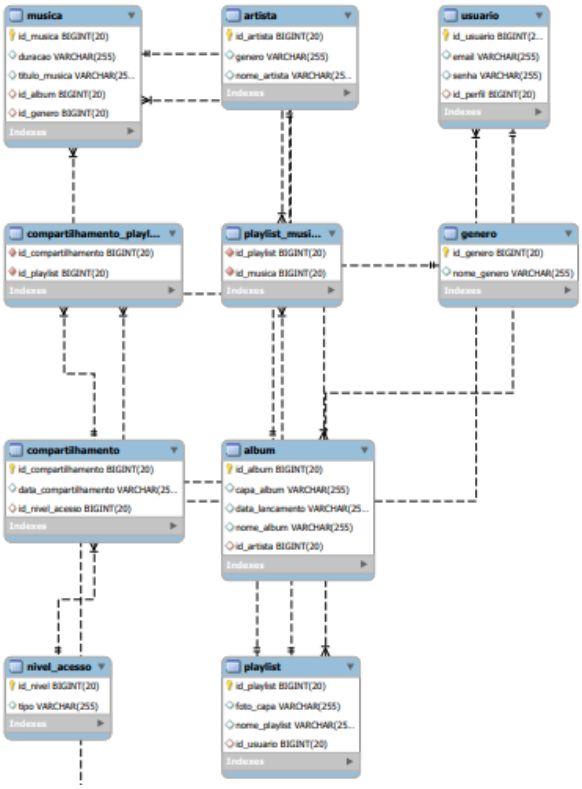

Figura 6. MER das entidades da História de Usuário 03.

2.4. História de Usuário 04

Como um usuário do Sistema de música, eu quero criar uma playlist. Para isso irei adicionar a primeira música em uma nova playlist, dar o nome para ela e adicionar outras músicas.

A figura 07 é o UML da história de usuário 04. A entidade Usuario vai relacionar com a entidade Playlist que vai relacionar com a entidade Musica, onde vai receber o título da música, nome do artista, e vai informar o nome da playlist.

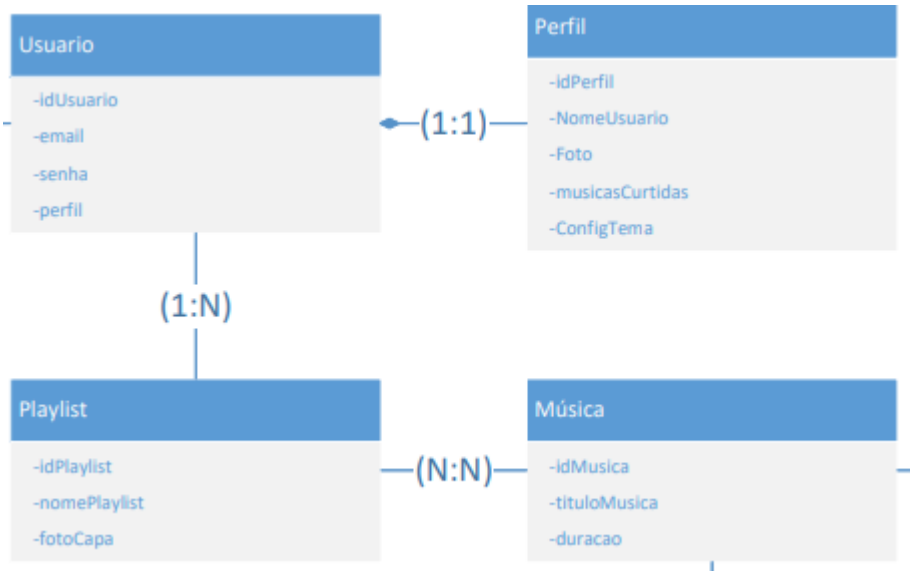

Figura 7. UML das entidades da História de Usuário 04.

A figura 08 é o MER da história de usuário 04. A entidade Usuario contém a chave estrangeira da entidade Playlist que contém a chave estrangeira da entidade Musica. A entidade Usuario vai fornecer o titulo da música e o nome da playlist que deverá ser atribuída. A entidade Usuario faz relacionamento com a entidade Playlist de 1:N e que faz relacionamento com a entidade Musica de N:N que fornece as informações de título da música, duração e id da música.

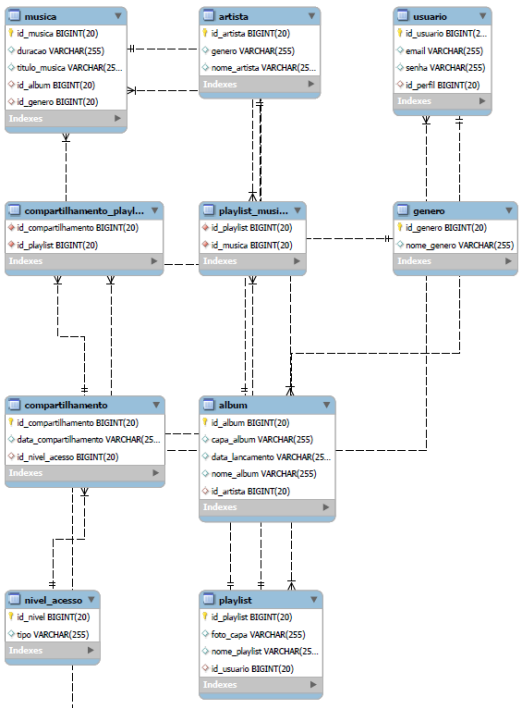

Figura 8. MER das entidades da História de Usuário 04.

2.5. História de Usuário 05

Como um usuário do Sistema de música, eu quero encontrar artistas de um certo gênero. Para isso irei pesquisar na parte de gênero o gênero que desejo e olhar os artistas que pertencem ao tal.

A figura 09 é o UML da história de usuário 05. A entidade Artista vai relacionar com a entidade Genero, onde vai receber o gênero e vai informar o nome dos artistas relacionados.

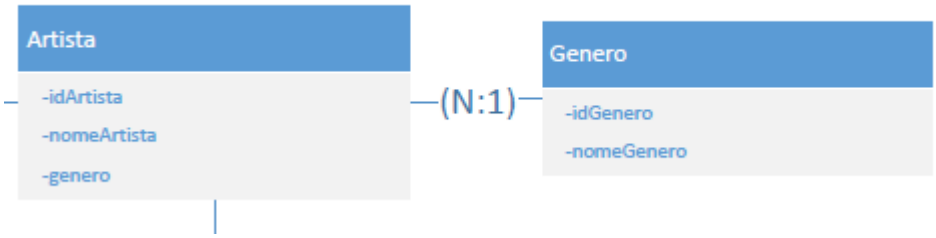

Figura 9. UML das entidades da História de Usuário 05.

A figura 10 é o MER da história de usuário 05. A entidade Artista contém a chave estrangeira da entidade Genero. A entidade Genero vai fornecer o nome dos artistas relacionado com o gênero solicitado. É um relacionamento de N:1 onde o mesmo gênero pode pertencer a vários artistas, mas o artista pode ter somente um gênero.

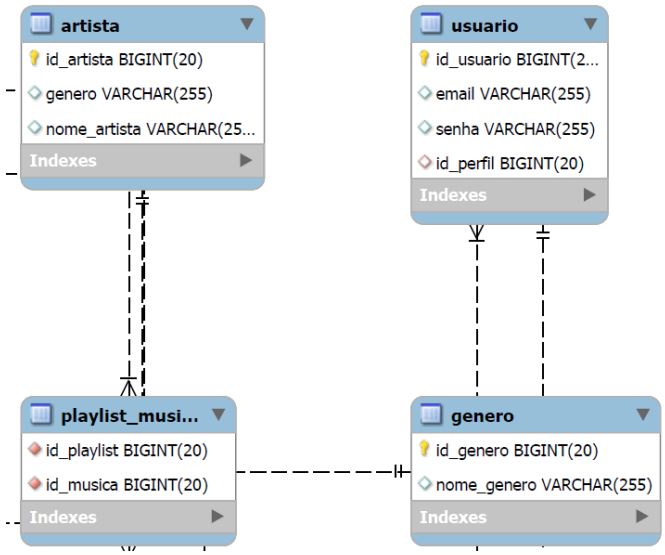

Figura 10. MER das entidades da História de Usuário 05.

3. Codificação

Apresentar as entidades e como realizou os relacionamentos. Apresentar o Diagrama complete em forma de figura XY.

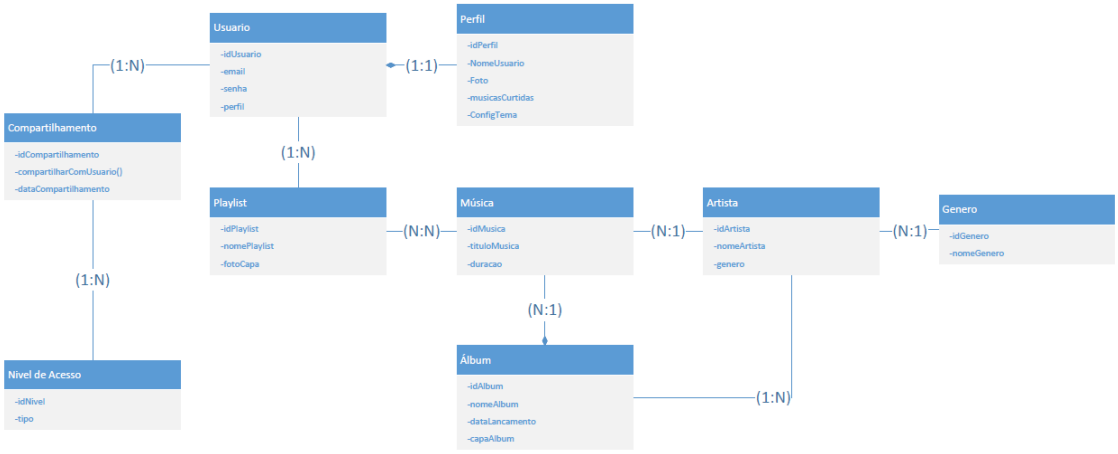

Figura 11. Diagrama de classe do Sistema de música Sorriso Brilhante.

3.1. Entidade Album

A entidade Album representa o relacionamento com o artista e as músicas.

```
1: @Entity
2: public class Album {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @OneToMany(mappedBy = "album")
6: private List<Musica> musicas;
7: @ManyToOne
8: @JoinColumn(name = "idArtista")
9: private Artista artista;
10: }
```

Figura 12. Código da entidade Album

A entidade Artista representa o relacionamento com o Álbum.

```
1: @Entity
2: public class Artista {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @ManyToMany(mappedBy = "artista")
6: private List<Album> albuns;
10: }
```

Figura 13. Código da entidade Artista

A entidade Compartilhamento representa o relacionamento com o a playlist a ser compartilhada e os níveis de acesso a serem concedidos.

```
1: @Entity
2: public class Compartilhamento {
3:@Id
4:@GeneratedValue(strategy = GenerationType.IDENTITY)
5: @ManyToOne
6: @JoinColumn(name = "idNivelAcesso")
7: private NivelAcesso nivelAcesso;
8: @ManyToMany
9: @JoinTable(name = "compartilhamento_playlist",
10: joinColumns = @JoinColumn(name = "idCompartilhamento"),
11: inverseJoinColumns = @JoinColumn(name = "idPlaylist"))
12: private List<Playlist> playlists;
13: }
```

Figura 14. Código da entidade Compartilhamento

A entidade Genero representa o relacionamento com as músicas.

```
1: @Entity
2: public class Genero {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @OneToMany(mappedBy = "genero")
6: private List<Musica> musicas;
7: }
```

Figura 15. Código da entidade Genero

A entidade Musica representa o relacionamento com a playlist, o álbum e o gênero.

```
1: @Entity
2: public class Musica {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @ManyToMany(mappedBy = "musicas")
6: private List<Playlist> playlists;
7: @ManyToOne
8: @JoinColumn(name = "idAlbum")
9: private Album album;
10: @ManyToOne
11: @JoinColumn(name = "idGenero")
12: private Genero genero;
13: }
```

Figura 16. Código da entidade Genero

A entidade NivelAcesso representa o relacionamento direto com o compartilhamento.

```
1: @Entity
2: public class NivelAcesso {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @OneToMany(mappedBy = "nivelAcesso")
6: private List<Compartilhamento> compartilhamentos;
7: }
```

Figura 17. Código da entidade NivelAcesso

A entidade Perfil representa o relacionamento direto com o usuário.

```
1: @Entity
2: public class Perfil {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @OneToMany(mappedBy = "perfil")
6: private Usuario usuario;
7: }
```

Figura 18. Código da entidade Perfil

A entidade Playlist representa o relacionamento com o usuário e com as músicas.

```
1: @Entity
2: public class Playlist {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @ManyToOne
6: @JoinColumn(name = "idUsuario")
7: private Usuario usuario;
8: @ManyToMany
9: @JoinTable(name = "playlist_musica",
10: joinColumns = @JoinColumn(name = "idPlaylist"),
11: inverseJoinColumns = @JoinColumn(name = "idMusica"))
12: private List<Musica> musicas;
13: }
```

Figura 19. Código da entidade Playlist

A entidade Usuario representa o relacionamento com o perfil e as playlists.

```
1: @Entity
2: public class Usuario {
3: @Id
4: @GeneratedValue(strategy = GenerationType.IDENTITY)
5: @OneToOne(cascade = CascadeType.ALL)
6: @JoinColumn(name = "idPerfil")
7: private Perfil perfil;
8: @OneToMany(mappedBy = "usuario", cascade =
9: CascadeType.ALL)
10: private List<Playlist> playlists;
11: }
```

Figura 20. Código da entidade Usuario

4. Banco de dados

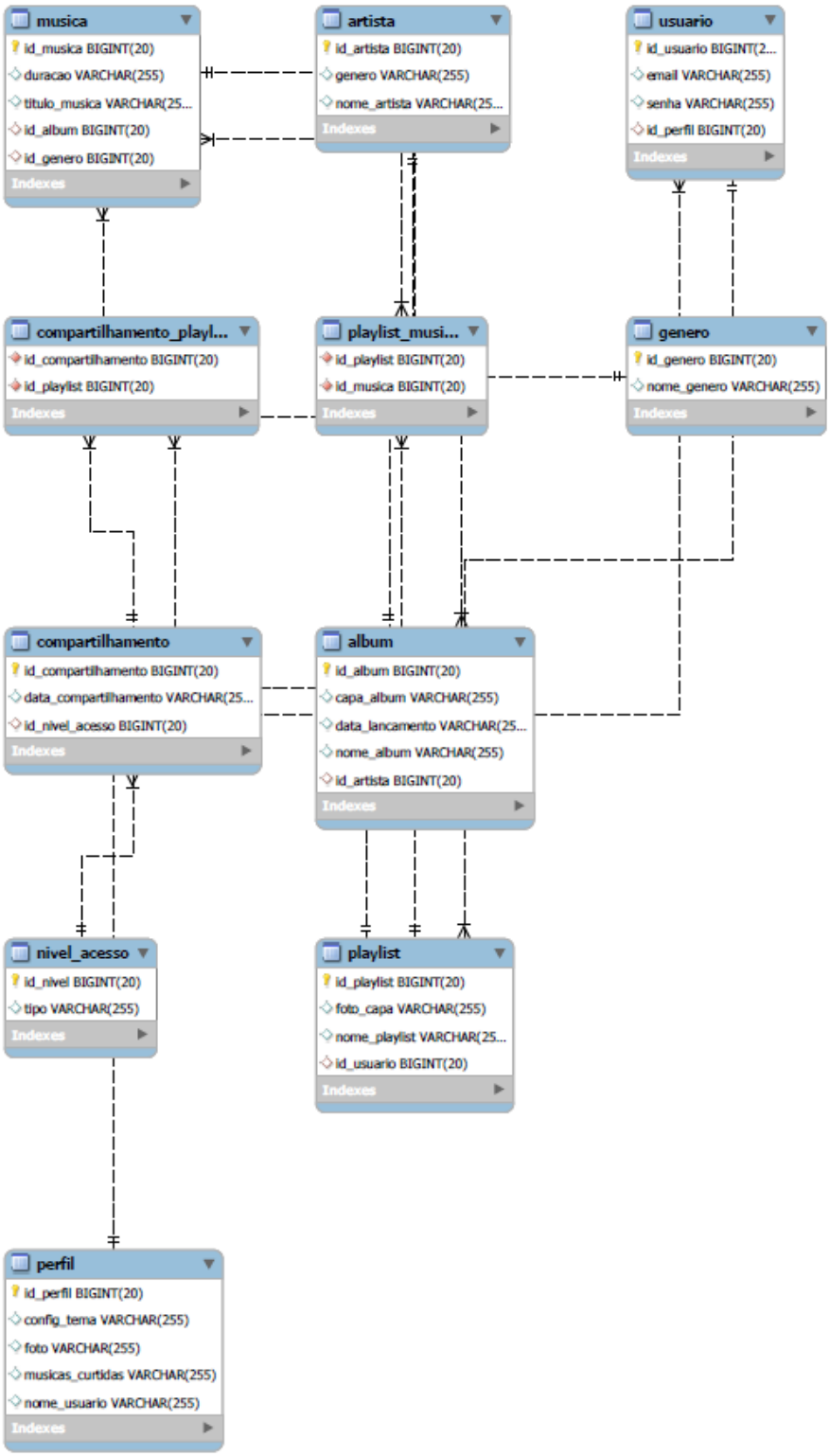

Figura 21. Modelo Entidade Relacionamento do Sistema de Música Sorriso Brilhante

4. Conclusão

É um código funcional com potencial, mas ainda há erros. Acredito que com dedicação e futuros updates o software poderá ser tão bom quanto os mais famosos reprodutores de músicas.

Referências

Boulic, R. and Renault, O. (1991) "3D Hierarchies for Animation", In: New Trends in Animation and Visualization, Edited by Nadia Magnenat-Thalmann and Daniel Thalmann, John Wiley & Sons ltd., England.

Dyer, S., Martin, J. and Zulauf, J. (1995) "Motion Capture White Paper", http://reality.sgi.com/employees/jam_sb/mocap/MoCapWP_v2.0.html, December.

Holton, M. and Alexander, S. (1995) "Soft Cellular Modeling: A Technique for the Simulation of Non-rigid Materials", Computer Graphics: Developments in Virtual Environments, R. A. Earnshaw and J. A. Vince, England, Academic Press Ltd., p. 449-460.

Knuth, D. E. (1984), The TeXbook, Addison Wesley, 15th edition.

Smith, A. and Jones, B. (1999). On the complexity of computing. In *Advances in Computer Science*, pages 555–566. Publishing Press.