

Projeto de Base de Dados

2024/25

2ª Fase

Plataforma de Músicas



Imagem de unsplash.com

Trabalho realizado pelo grupo G25 do turno prático 2:

Diogo Matias (Nº 70019)

Rafael Rodrigues (Nº 65771)

Rodrigo Santos (Nº 65745)

Apresentação do Tema

Os serviços de *streaming* estão a tornar-se cada vez mais populares, substituindo a compra física ou digital de artigos individuais. Ao pagar uma subscrição mensal, os utilizadores têm acesso a uma vasta biblioteca de conteúdos. No nosso caso, os artigos que pretendemos representar são músicas.

Grandes serviços de *streaming*, como o Spotify, revolucionaram a forma como consumimos e ouvimos música. Foi esta transformação que nos motivou a explorar e tentar replicar as funcionalidades base presentes nestes serviços. Entre estas funcionalidades, destacam-se a representação dos artistas presentes na plataforma, os seus álbuns e as suas músicas.

Além disso, estes serviços incluem uma componente significativa de registo do que cada utilizador ouve, permitindo recomendar artistas e músicas semelhantes. Por isso, também replicámos o histórico de reproduções das músicas que os utilizadores ouvem na plataforma.

Descrição

A plataforma tem vários artistas, cada um com um nome e país de origem. O nome do artista é único dentro da plataforma. O artista pode lançar vários álbuns e ter mais que uma música que criou.

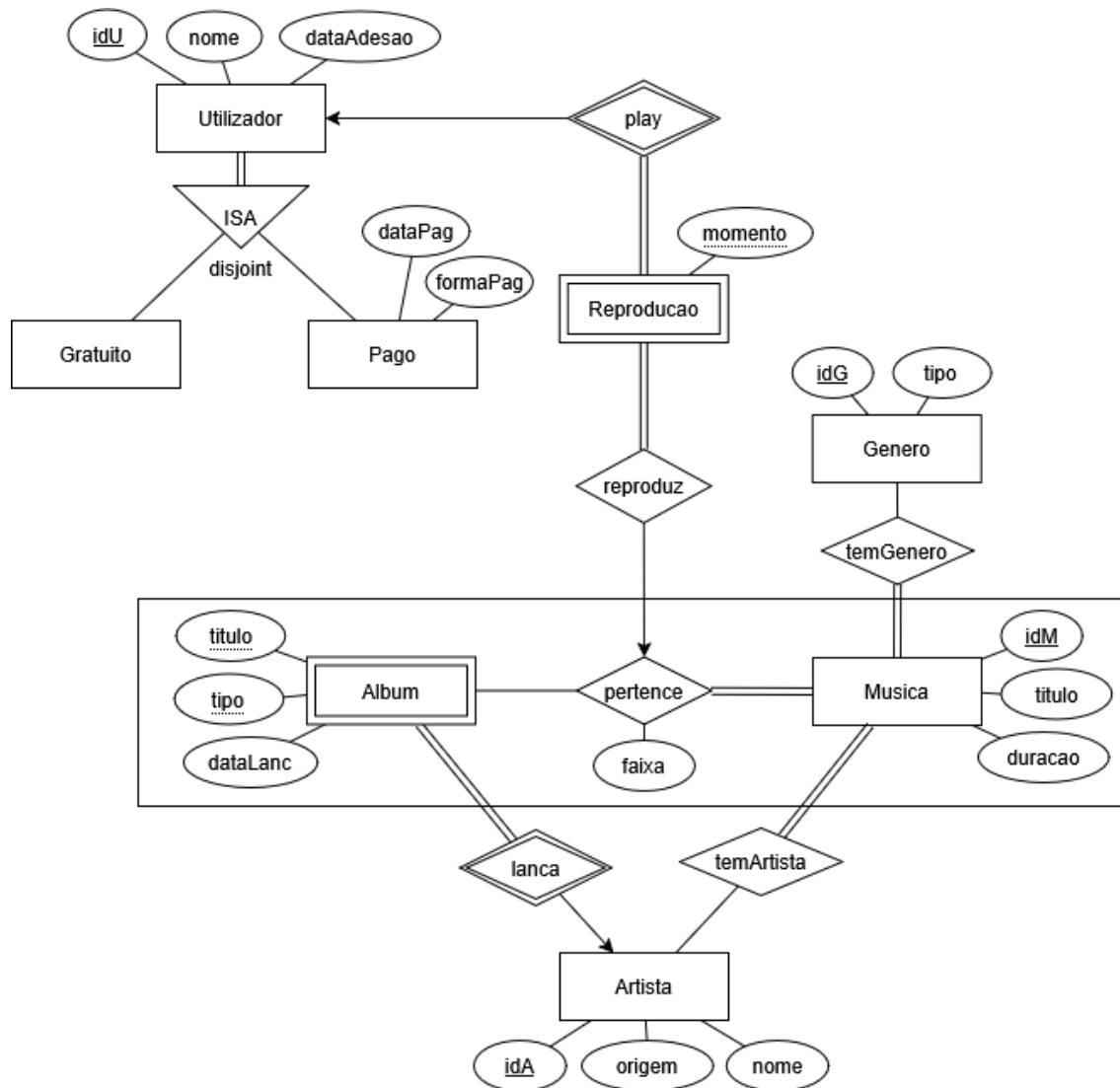
Cada álbum tem um título, data de lançamento e tipo, que pode ser: Single, EP ou LP. Um álbum só pode ter um único artista. Um álbum para existir não precisa de ter músicas, contudo, ele não será mostrado nem disponível para o utilizador. Um álbum pode ter muitas músicas, independentemente do tipo.

Cada música tem um título e duração, que está descrita em milissegundos. Cada música tem de ter um género musical, e no máximo só pode ter dois. Uma música tem de ter um álbum associado e pode pertencer a mais que um álbum. Além disso, uma música tem de ter no mínimo um artista.

Os utilizadores têm de estar inscritos na plataforma para poderem reproduzir música. De cada utilizador, queremos guardar o seu nome e data de adesão na plataforma. O utilizador poderá pagar, ou não, para usufruir do serviço, e para o utilizador que paga queremos guardar a data e o formato de pagamento. Cada utilizador tem associado um histórico das músicas que reproduziu.

O utilizador pode reproduzir várias músicas, mas nunca duas ao mesmo tempo. As músicas podem ser reproduzidas por qualquer utilizador e, como se trata de um serviço de *streaming*, vários utilizadores podem estar a ouvir a mesma música ao mesmo tempo.

Modelo ER



Opções Tomadas Sobre o Modelo ER

- **Agregação com Álbum e Música:** Quando guardamos uma reprodução que um utilizador faça, queremos saber que música é que reproduziu e a que álbum essa música pertence. Como uma música pode pertencer a vários álbuns, a melhor maneira de ter toda a informação que precisamos para uma reprodução é através de uma agregação de álbum com música.
- **Música:** Uma música depende de um artista para existir, mas como uma música pode ter mais que um artista a mesma terá de ser uma entidade forte. Mesmo assim, dentro da plataforma garantimos que uma música tem de ter um artista associado.

- **Especialização (ISA):** Na especialização do Utilizador temos dois tipos: pago, utilizador que paga pelo serviço; gratuito, utilizador que não paga pelo serviço. Esta especialização em utilizador é *disjoint* porque um utilizador que não pague não pode ser considerado um utilizador que paga, e vice-versa.
- **Reprodução:** A entidade fraca reprodução foi a que tivemos mais dificuldade em resolver. Queremos que esta entidade sirva de histórico das reproduções que os utilizadores fazem na plataforma. Para isso, é necessário garantir que um utilizador só possa ouvir uma música num dado momento. Daí a termos desenhado como uma entidade fraca que é dependente de um utilizador e que reproduz apenas uma música pertencente num álbum.

Alterações ao Esquema

- **Eliminação de críticos, críticas e classificar:** Foram eliminadas as entidades Crítico e Crítica e respetivas relações e mais a relação classificar, porque estávamos com tabelas a mais e quisemos reduzir a complexidade e concentrar a nossa atenção na implementação e teste das funcionalidades base.
- **Eliminação de pessoas:** Uma vez eliminada a entidade Crítico, deixou de fazer sentido fazer um ISA entre Crítico e Utilizador. Assim, Utilizador herdou os atributos da entidade Pessoa.
- **Alteração da relação reproduz:** Reproduz deixou de ser uma relação fraca, porque, devido a uma observação levantada pelo professor, um utilizador poderia estar a ouvir duas músicas diferentes ao mesmo tempo. Antigamente, a entidade Reprodução tinha todos os seus atributos como pertencendo à chave principal, levantando imensos problemas de integridade e lógicos. A solução encontrada vai de acordo à lógica descrita na descrição.
- **Alteração da cardinalidade entre álbum e música:** Álbum deixou de precisar de ter músicas para existir. A razão para esta alteração deve-se à lógica e ação de “querer” criar um álbum, mas ainda não ter músicas para adicionar.
- **Adição do tipo em álbum a chave discriminante:** Álbum passou de ter apenas o título como chave discriminante a ter título e tipo. Isto deve-se porque um artista pode ter um EP, ou até mesmo um Single, com o mesmo título que um Álbum. Necessitando então de mais um atributo para identificar.

5. Consultas Interessantes

Modelo Relacional

1. **Utilizador** (idU, nome, dataAdesão)
2. **Pago** (idU, dataPag, formaPag)
 - idU é chave estrangeira de Utilizador
3. **Gratuito** (idU)
 - idU é chave estrangeira de Utilizador
4. **Música** (idM, título, duração)
5. **Género** (idG, tipo)
6. **temGénero** (idG, idM)
 - idG é chave estrangeira de Género
 - idM é chave estrangeira de Música
7. **Artista** (idA, nome, origem)
8. **temArtista** (idA, idM)
 - idA é chave estrangeira de Artista
 - idM é chave estrangeira de Música
9. **Álbum** (idA, título, tipo, dataLanc)
 - idA é chave estrangeira de Artista
10. **pertence** (idM, idA, título, tipo, faixa)
 - idM é chave estrangeira de Música
 - idA, título e tipo são chaves estrangeiras de Álbum
11. **Reprodução** (idU, momento, idM, idA, título, tipo)
 - idU é chave estrangeira de Utilizador
 - idM, idA, título e tipo são chaves estrangeiras de pertence

SQL

Triggers de Integridade

trg_vw_inserir_musica

```
1 create or replace trigger trg_vw_inserir_musica
2 instead of insert on vw_inserir_musica
3 for each row
4 declare
5     v_idM number;
6     v_faixa number;
7 begin
8     -- inserir em músicas e obter o idM gerado
9     insert into musicas (titulo, duracao)
10    values (:new.titulo, :new.duracao)
11    returning idM into v_idM;
12
13    -- número de músicas no álbum
14    select count(*) into v_faixa
15    from albums join pertence using (idA, titulo, tipo)
16    where idA = :new.idA
17      and titulo = :new.album
18      and tipo = :new.tipo;
19
20    -- ligar música a álbum
21    insert into pertence
22    values (v_idM, :new.idA, :new.album, :new.tipo, v_faixa + 1);
23
24    -- ligar música a género
25    insert into temGenero
26    values (:new.idG, v_idM);
27
28    -- ligar música a artista
29    insert into temArtista
30    values (:new.idA, v_idM);
31 end;
32 /
```

Assim como a *view*

vw_inserir_musica descrita abaixo, este é um dos triggers mais importantes para manter a integridade da base de dados. Este *trigger* não só insere uma música na plataforma como a liga a todas as relações que estão dependentes – pertence, temGénero e temArtista – de forma a garantir que uma música pertence a um álbum, tem um género e foi criada por um artista.

Este *trigger* ao adicionar a música a um álbum, calcula o número da faixa que a música vai ter com base no número total de músicas no álbum.

trg_vw_remove_musica

```
1 create or replace trigger trg_vw_remove_musica
2 instead of delete on vw_inserir_musica
3 for each row
4 begin
5     delete from musicas where :old.idM = idM;
6 end;
7 /
```

Este *trigger* remove uma música da plataforma. Uma vez que no Oracle Apex iremos trabalhar na *view*, precisamos deste *trigger* para garantir que a música é removida na sua tabela.

trg_vw_update_musica

```
1 create or replace trigger trg_vw_update_musica
2 instead of update on vw_inserir_musica
3 for each row
4 declare
5     v_faixa number;
6 begin
7     update musicas
8     set titulo = :new.titulo, duracao = :new.duracao
9     where idM = :old.idM;
10
11     delete from temArtista where :old.idM = idM;
12     insert into temArtista
13     values (:new.idA, :old.idM);
14
15     -- número de músicas no álbum
16     select count(*) into v_faixa
17     from albums join pertence using (idA, titulo, tipo)
18     where idA = :new.idA
19     and titulo = :new.album
20     and tipo = :new.tipo;
21
22     delete from pertence where :old.idM = idM;
23     insert into pertence
24     values (:old.idM, :new.idA, :new.album, :new.tipo, v_faixa + 1);
25
26     delete from temGenero where :old.idM = idM;
27     insert into temGenero
28     values (:new.idG, :old.idM);
29 end;
30 /
```

Este *trigger* atualiza uma música na plataforma. Remove e insere a nova música em cada tabela, preservando o idM, o *ID* da música.

Com este *trigger* é possível mudar qualquer informação da música, desde o álbum a que pertence, o artista que a criou e até o género da mesma.

trg_data_pagamento

```
1 create or replace trigger trg_data_pagamento
2 before insert on pagos
3 for each row
4 declare
5     dAdesao date;
6 begin
7     select dataAdesao into dAdesao
8     from utilizadores
9     where idU = :new.idU;
10    if (:new.dataPag < dAdesao) then
11        Raise_Application_Error(-20019, 'Data de pagamento inválida.');
```

Este *trigger* valida que a data de pagamento inserida acontece depois da data de adesão, mantendo a integridade dos dados na plataforma.

trg_utilizador_default

```
1 create or replace trigger trg_utilizador_default
2 after insert on utilizadores
3 for each row
4 begin
5     insert into gratuitos (idU) values (:new.idU);
6 end;
7 /
```

Uma vez que o utilizador tem de ser pago ou gratuito, este *trigger*, na inserção de um novo utilizador, insere por *default* o utilizador como gratuito na plataforma.

trg_remove_utilizador_gratuito

```
1 create or replace trigger trg_remove_utilizador_gratuito
2 after insert on pagos
3 for each row
4 begin
5     delete from gratuitos where idU = :new.idU;
6 end;
7 /
```

Quando um utilizador gratuito passa a ser um utilizador pago, este *trigger* remove o utilizador da tabela de gratuito de forma a manter os utilizadores como sendo, ou gratuito ou pago.

trg_remove_utilizador_pago

```
1 create or replace trigger trg_remove_utilizador_pago
2 after insert on gratuitos
3 for each row
4 begin
5     delete from pagos where idU = :new.idU;
6 end;
7 /
```

Quando um utilizador pago passa a ser um utilizador gratuito, este *trigger* remove o utilizador da tabela de pago de forma a manter os utilizadores como sendo, ou gratuito ou pago.

trg_limite_generos_musica

```
1 create or replace trigger trg_limite_generos_musica
2 before insert on temGenero
3 for each row
4 declare
5     numTotal number;
6 begin
7     select count(*) into numTotal
8     from temGenero
9     where idM = :new.idM;
10    if (numTotal = 2) then
11        Raise_Application_Error(-20020, 'Música não pode ter mais que dois géneros.');
```

Este *trigger* limita uma música de ter no máximo dois géneros musicais como previsto na descrição da nossa plataforma.

Funções

fn_formatar_duracao(p_ms in number)

```
1 create or replace function fn_formatar_duracao(p_ms in number)
2   return varchar2
3 is
4   v_horas pls_integer;
5   v_minutos pls_integer;
6   v_segundos pls_integer;
7 begin
8   -- obter horas, minutos e segundos
9   v_horas := trunc(p_ms / 3600000);
10  v_minutos := trunc(mod(p_ms, 3600000) / 60000);
11  v_segundos := trunc(mod(p_ms, 60000) / 1000);
12
13  -- devolve string
14  return to_char(v_horas)
15         || ':' || lpad(to_char(v_minutos), 2, '0')
16         || ':' || lpad(to_char(v_segundos), 2, '0');
17 end fn_formatar_duracao;
18 /
```

Esta função formata a duração das músicas que se encontra em milissegundos para um formato de “h:mm:ss”.

Esta função foi feita com a ajuda do ChatGPT.

Views

vw_inserir_musica

```
1 create or replace view vw_inserir_musica as
2 select
3   a.idA,
4   m.idM,
5   m.titulo,
6   m.duracao,
7   alb.titulo as album,
8   alb.tipo,
9   g.idG
10 from
11   musicas m
12   join temArtista ta on m.idM = ta.idM
13   join artistas a on ta.idA = a.idA
14
15   join pertence p on m.idM = p.idM
16   join albuns alb on p.idA = alb.idA
17     and p.titulo = alb.titulo
18     and p.tipo = alb.tipo
19
20   join temGenero tg on m.idM = tg.idM
21   join generos g on tg.idG = g.idG
22 ;
```

Esta *view* é das *views* mais importantes e serve para a inserção de uma música na plataforma. Ao inserir uma música é preciso garantir que ela tem um artista, pertence a um álbum e tem um género. Com esta *view* e os *triggers* que dependem dela, é possível garantir que se coloca a informação necessária para inserir uma música na plataforma e atualizar todas as tabelas que advêm dessa ação.

vw_catalogo_musicas

```
1 create or replace view vw_catalogo_musicas as
2 select
3     a.idA,
4     m.idM,
5     nvl(r.numReproducoes, 0) as numReproducoes,
6     fn_formatar_duracao(m.duracao) as duracao,
7     g.idG
8 from
9     musicas m
10    join temArtista ta on m.idM = ta.idM
11    join artistas a on ta.idA = a.idA
12
13    join temGenero tg on m.idM = tg.idM
14    join generos g on tg.idG = g.idG
15
16    left join (
17        select idM, count(*) as numReproducoes
18        from reproducoes
19        group by idM
20    ) r on m.idM = r.idM
21 ;
```

Esta view foi criada para obter todas as informações de uma música assim como as reproduções totais que uma música tem. Com os *IDs* idA, idM e idG é possível chegar a todas as tabelas que têm mais informação. Parece incompleta por não ter esta informação já incluída, mas foi uma escolha de desenho que se reflete nas *views* seguintes.

vw_catalogo_albums

```
1 create or replace view vw_catalogo_albums as
2 with info_album as (
3     select
4         alb.idA,
5         alb.titulo,
6         alb.tipo,
7         count(*) as numMusicas,
8         sum(m.duracao) as duracaoTotal
9     from
10         albums alb
11        join pertence p on alb.idA = p.idA
12        and alb.titulo = p.titulo
13        and alb.tipo = p.tipo
14        join musicas m on p.idM = m.idM
15    group by alb.idA, alb.titulo, alb.tipo
16 )
17 select
18     idA,
19     titulo,
20     tipo,
21     dataLanc,
22     numMusicas,
23     fn_formatar_duracao(duracaoTotal) as duracaoTotal
24 from
25     info_album
26    join albums using (idA, titulo, tipo)
27    join artistas using (idA)
28 ;
```

Esta view foi criada para obter informações complementares de um álbum, como o número de músicas total que um álbum tem e a sua duração total.

vw_artistas

```
1 create or replace view vw_artistas as
2 with count_reproducoes_artista as (
3     select idA, count(*) as numReproducoes
4     from reproducoes
5     group by idA
6 ),
7 count_albums_artista as (
8     select idA, count(*) as numAlbums
9     from albums
10    group by idA
11 ),
12 count_musicas_artista as (
13     select idA, count(*) as numMusicas
14     from temArtista
15    group by idA
16 )
17 select
18     idA,
19     nome,
20     origem,
21     nvl(numReproducoes, 0) as numReproducoes,
22     nvl(numAlbums, 0) as numAlbums,
23     nvl(numMusicas, 0) as numMusicas
24 from
25     artistas
26 left join count_reproducoes_artista using (idA)
27 left join count_albums_artista using (idA)
28 left join count_musicas_artista using (idA)
29 ;
```

Esta view foi criada para obter informações complementares de um artista, como o número de reproduções totais, o número de álbuns total, e o número de músicas total.

vw_utilizadores

```
1 create or replace view vw_utilizadores as
2 select
3     idU,
4     nome,
5     dataAdesao,
6     'GRATUITO' as tipo,
7     null as dataPag,
8     null as formaPag
9 from
10    utilizadores
11 join gratuitos using (idU)
12 union all
13 select
14     idU,
15     nome,
16     dataAdesao,
17     'PAGO' as tipo,
18     dataPag,
19     formaPag
20 from
21    utilizadores
22 join pagos using (idU)
23 ;
```

Esta view foi criada para obter uma informação geral de todos os utilizadores que se encontram na plataforma. Uma vez que um utilizador ou é pago ou é gratuito, criámos esta view para unir os utilizadores de ambas essas tabelas – Pago e Gratuito – e obter todos numa só tabela.

Limitações e Opções Tomadas na Implementação da BD

Na implementação da base de dados, foi necessário fazer alguns compromissos para evitar complicações excessivas, considerando os nossos conhecimentos atuais e sendo esta a primeira vez que realizamos um projeto deste tipo.

Uma das principais limitações é que os álbuns só podem ter um artista associado. Com mais conhecimento em SQL, teríamos optado por uma solução em que um álbum poderia ter um artista principal e vários artistas secundários. Desta forma, poderíamos ter álbuns com múltiplos artistas, mantendo a dependência do álbum em relação ao artista principal.

Esta abordagem poderia também ser aplicada às músicas. Ou seja, uma música poderia ter um artista principal e vários artistas secundários. A decisão de implementar a solução atual limita a identificação do verdadeiro e principal criador de uma música. Por exemplo, se uma música tiver dois artistas, A (principal) e B, e o artista A for removido da plataforma, a música continuaria a existir com o artista B.

Outra limitação encontra-se no tipo do álbum. Devido à falta de planeamento prévio e gestão de tempo, optámos por fazer um *check* para garantir que o tipo do álbum seja exatamente “SINGLE”, “EP” ou “ALBUM”, em vez de criar uma tabela com código e tipo do álbum. Além disso, em teoria, existe uma limitação no número de músicas que um álbum pode ter dependente do tipo, mas deixámos essa opção em aberto para a pessoa que insere o álbum e as músicas nele.

Apesar destas limitações, tentámos preservar alguma complexidade na implementação da nossa base de dados, permitindo que uma música possa ter vários artistas, pertencer a vários álbuns e ter vários géneros musicais.

Uma opção tomada para a adição do tipo como chave discriminante em álbum foi devido a um problema que surgiu quando quisemos adicionar os últimos Singles e EPs antes do lançamento do álbum "Nenhuma Estrela" dos Terno Rei. Deparámo-nos com o facto de que tanto o álbum como o EP partilhavam o mesmo nome, pelo que decidimos elevar o tipo do álbum a uma chave que ajuda a definir.

Consultas Interessantes

Nenhuma Estrela de Terno Rei

```
1  with reproducoesMusica as (  
2      select idM, count(*) as numReproducoes  
3      from reproducoes  
4      group by idM  
5  )  
6  select  
7      a.nome as "Artista",  
8      p.faixa,  
9      m.titulo as "Título",  
10     r.numReproducoes as "Reproduções",  
11     fn_formatar_duracao(m.duracao) as "Duração",  
12     alb.titulo as "Álbum",  
13     alb.tipo  
14 from  
15     artistas a  
16     join albuns alb on a.idA = alb.idA  
17  
18     join pertence p on alb.idA = p.idA  
19         and alb.titulo = p.titulo  
20         and alb.tipo = p.tipo  
21  
22     join musicas m on p.idM = m.idM  
23  
24     left join reproducoesMusica r on m.idM = r.idM  
25 where  
26     a.nome = 'Terno Rei'  
27     and alb.titulo = 'Nenhuma Estrela'  
28     and alb.tipo = 'ALBUM'  
29 order by  
30     p.faixa
```

Nesta consulta, pretendemos obter as músicas pertencentes ao álbum "Nenhuma Estrela" do artista "Terno Rei", incluindo informações sobre as faixas, reproduções e duração das músicas.

1. **Definição temporária de uma relação:** Utilizamos uma tabela temporária chamada `reproducoesMusica` para contar o número de reproduções de cada música, agregando as reproduções por `idM`.
2. **Seleção de dados:** Seleccionamos as colunas nome do artista, faixa, título da música, número de reproduções, duração, título do álbum e tipo do álbum.
3. **Junções entre tabelas:** Fazemos junções entre as tabelas `artistas`, `albuns`, `pertence`, `musicas` e a tabela temporária `reproducoesMusica` para obter todas as informações necessárias.
4. **Filtragem por Artista e Álbum:** Filtramos os resultados para incluir apenas as informações do artista "Terno Rei" e do álbum "Nenhuma Estrela" do tipo "ALBUM".
5. **Ordenação por faixa:** Ordenamos os resultados pela faixa da música no álbum.

6. **Left Join para manter todas as músicas:** Utilizamos um *Left Join* para garantir que todas as músicas pertencentes ao álbum sejam mantidas na consulta, mesmo que não tenham nenhuma reprodução.

Top Dezembro de 2024

```
1 with reproducoesMusica as (  
2   select idM, count(*) as numReproducoes  
3   from reproducoes  
4   group by idM  
5 )  
6 select distinct  
7   a.nome as "Artista",  
8   m.titulo as "Música",  
9   alb.titulo as "Álbum",  
10  alb.tipo,  
11  rm.numReproducoes  
12 from  
13  reproducoes r  
14  join musicas m on r.idM = m.idM  
15  
16  join albums alb on r.idA = alb.idA  
17                  and r.titulo = alb.titulo  
18                  and r.tipo = alb.tipo  
19  
20  join artistas a on r.idA = a.idA  
21  
22  join reproducoesMusica rm on r.idM = rm.idM  
23 where  
24   extract(month from r.momento) = 12  
25   and extract(year from r.momento) = 2024  
26 order by  
27   rm.numReproducoes desc;
```

Nesta consulta, pretendemos obter as músicas que foram reproduzidas em dezembro de 2024, mostrando-as da mais reproduzida para a menos reproduzida.

1. **Definição temporária de uma relação:** Utilizamos uma tabela temporária chamada *reproducoesMusica* para contar o número de reproduções de cada música, agregando as reproduções por *idM*.
2. **Seleção de dados:** Seleccionamos as colunas nome do artista, título da música, título do álbum, tipo do álbum e número de reproduções.
3. **Junções entre tabelas:** Fazemos junções entre as tabelas reproduções, músicas, álbuns, artistas e a tabela temporária *reproducoesMusica* para obter todas as informações necessárias.
4. **Filtragem por data pretendida:** Filtramos os resultados para incluir apenas as reproduções que ocorreram em dezembro de 2024.

5. **Ordenação pelo número de reproduções:** Ordenamos os resultados pelo número de reproduções em ordem decrescente.
6. ***Distinct* para evitar duplicados:** Utilizamos o *select distinct* para garantir que não há linhas duplicadas na consulta.

Utilizadores que, no mínimo, reproduziram uma música de Capitão Fausto e de Travis Scott

```
1 with utilizadores_capitao_fausto as (  
2     select idu, count(*)  
3     from reproducoes join artistas using (idA)  
4     where nome = 'Capitão Fausto'  
5     group by idu  
6 ),  
7 utilizadores_travis_scott as (  
8     select idu, count(*)  
9     from reproducoes join artistas using (idA)  
10    where nome = 'Travis Scott'  
11    group by idu  
12 )  
13 select  
14     nome as "Utilizador"  
15 from  
16     utilizadores_capitao_fausto join utilizadores using (idU)  
17 intersect  
18 select  
19     nome as "Utilizador"  
20 from  
21     utilizadores_travis_scott join utilizadores using (idU)
```

Nesta consulta, pretendemos obter os utilizadores que, no mínimo, reproduziram uma música de "Capitão Fausto" e de "Travis Scott".

1. **Definição temporária de duas relações:**
 - utilizadores_capitao_fausto: seleciona os utilizadores que reproduziram músicas do artista "Capitão Fausto".
 - utilizadores_travis_scott: seleciona os utilizadores que reproduziram músicas do artista "Travis Scott".
2. **Seleção de dados:** Selecionamos a coluna nome dos utilizadores que reproduziram músicas de ambos os artistas.
3. **Operação *Intersect*:** Utilizamos a operação *intersect* para obter apenas os utilizadores que estão presentes em ambas as relações temporárias, ou seja, os utilizadores que reproduziram músicas de ambos os artistas pretendidos.

Breve Nota

Em todas as consultas interessantes, optámos por manter as colunas mostradas pela ordem da consulta e não exibir chaves relacionadas com tabelas externas. Apresentamos apenas a informação relevante para o utilizador final.

Foi por isso que utilizámos definições temporárias de relações em alguns dos casos. Estas definições temporárias permitem-nos obter os resultados desejados sem incluir as chaves necessárias para a execução das agregações, tornando a informação mais clara e direta para o utilizador.

Implementação das Funcionalidades no Oracle APEX

Inserir música e catálogo de músicas

The screenshot shows the Oracle APEX application interface for managing music. The interface includes a sidebar with navigation links, a top header, and a main content area with filters and a table of music entries.

Plataforma Log Out

Início \

Músicas

Gêneros

Gênero - Sem Gênero -

Artistas

Artista - Sem Artista -

Q Go Actions Create

	Ida	Tipo	Artista	Duração	Gênero	Reproduções	Faixa	Título	Álbum
	14	ALBUM	Capitão Fausto	0:05:31	Indie Pop	3	1	Muitos Mais Virão	Subida Infinita
	15	ALBUM	Capitão Fausto	0:03:31	Indie Pop	3	2	Andar à Solta	Subida Infinita
	16	ALBUM	Capitão Fausto	0:02:43	Indie Pop	2	3	Na Na Nada	Subida Infinita
	17	ALBUM	Capitão Fausto	0:04:01	Indie Pop	2	4	Nunca Nada Muda	Subida Infinita
	18	ALBUM	Capitão Fausto	0:00:22	Indie Pop	3	5	Fantasia	Subida Infinita
	19	ALBUM	Capitão Fausto	0:03:39	Indie Pop	4	6	Nada de Mal	Subida Infinita
	20	ALBUM	Capitão Fausto	0:03:00	Indie Pop	4	7	Há Sempre um Fardo	Subida Infinita

Neste *Report with Form on Table* de músicas temos as músicas inseridas na plataforma. Além disso temos várias funcionalidades implementadas, como:

- (1), a existência de uma página de entrada “Início” que possui as ligações para todas as outras páginas da aplicação;
- (2), listagem das músicas onde códigos referentes a chaves externas, como idA e idG, encontram-se substituídas por outros atributos de fácil compreensão;
- (3), valores derivados como o número de reproduções que cada música tem;
- (4), é possível adicionar novas músicas, atualizar uma música existente ou eliminar uma música;
- (5) é possível preencher valores de atributos correspondentes a relações sem se ter conhecimento de códigos, como selecionar o gênero de uma música ou selecionar um artista.

Artistas

	Nome	Origem	Reproduções	Álbuns	Músicas
	Terno Rei	Brasil	94	5	13
	Lô Borges	Brasil	0	0	1
	Travis Scott	Estados Unidos da Am...	29	2	18
	Paira	Brasil	0	0	1
	Capitão Fausto	Portugal	30	1	10

1 rows selected Total 5

Neste *Two Page Master Detail*, (9), de artistas temos os artistas inseridos na plataforma. Além da informação base de um artista, temos informação adicional do número de reproduções totais que um artista tem nas suas músicas e o número de álbuns e músicas totais. Temos também várias funcionalidades implementadas, mas quero dar destaca aos 3 *reports* interligados (7):

- Link em reproduções que irá para um *Report* de reproduções passando o código do artista;
- Link em álbuns que irá para um *Report* de álbuns passando o código do artista;
- Link em músicas que irá para um *Report with Form on Table* de músicas passando o código do artista.

Nesta página é possível criar, editar e eliminar artistas. Além disso, ao clicar no “lápis” é possível ver os álbuns associados a cada artista e adicionar, editar ou eliminá-los.

	Título	Tipo	Data lanc
<input checked="" type="checkbox"/>	Nada Igual	SINGLE	20-FEB-25
<input type="checkbox"/>	Nenhuma Estrela	ALBUM	15-APR-25
<input type="checkbox"/>	Nenhuma Estrela	EP	08-APR-25
<input type="checkbox"/>	Próxima Parada	SINGLE	11-MAR-25
<input type="checkbox"/>	Relógio	SINGLE	16-APR-25

1 rows selected Total 5

Reproduções

Plataforma

Log Out

Início

Músicas

Gêneros

Artistas

Utilizadores

Reproduções

Catálogo de álbuns

Consultas

Início \

Reproduções

Artistas

Artista Terno Rei

Detalhes do Artista

Número de reproduções: 94

1 - 1

Q Go Actions

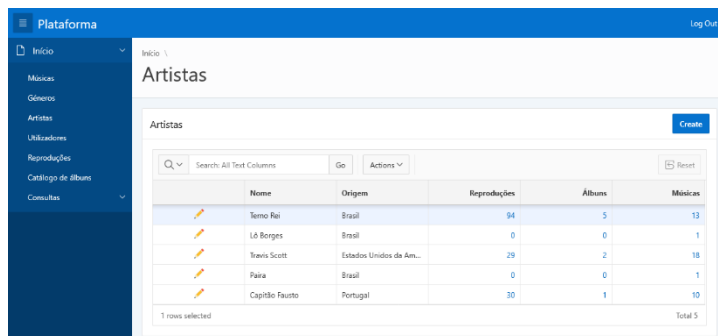
Momento	Tipo	Utilizador	Música	Artista	Álbum
24-MAY-25 09.00.00.000000 AM	SINGLE	Diogo	Nada Igual	Terno Rei	Nada Igual
24-MAY-25 09.02.00.000000 AM	SINGLE	Diogo	Viver de Amor	Terno Rei	Nada Igual
24-MAY-25 09.04.00.000000 AM	SINGLE	Diogo	Próxima Parada	Terno Rei	Próxima Parada
24-MAY-25 09.06.00.000000 AM	SINGLE	Diogo	Nada Igual	Terno Rei	Próxima Parada
24-MAY-25 09.08.00.000000 AM	SINGLE	Diogo	Viver de Amor	Terno Rei	Próxima Parada
24-MAY-25 09.10.00.000000 AM	EP	Diogo	Nenhuma Estrela	Terno Rei	Nenhuma Estrela
24-MAY-25 09.12.00.000000 AM	EP	Diogo	Próxima Parada	Terno Rei	Nenhuma Estrela

Neste *Report* de reproduções temos todas as reproduções de músicas feitas pelos utilizadores na plataforma. Esta página encontra-se interligada com a página de artistas, como descrito acima. É possível na “*select list*” filtrar as reproduções com base no artista que queremos e obter no detalhe condicional (8) o número de todas as reproduções correspondentes a esse mesmo artista.

Em todas as páginas implementadas no Oracle APEX existem links de navegação nas várias páginas (6) e na secção “Consultas Interessantes”, que pode ser visualizada em cima, temos implementados um *Report* para consulta interessante (10).

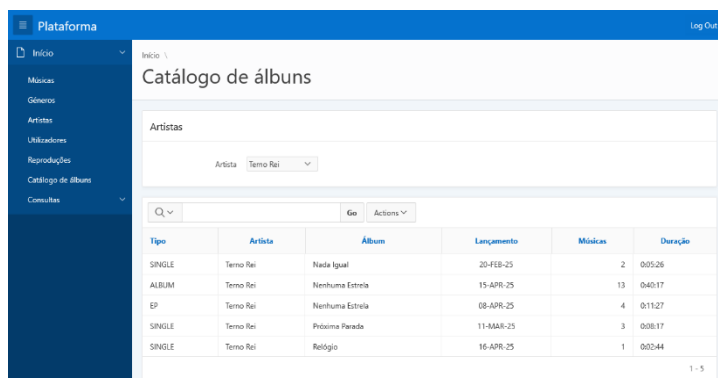
Pequeno Manual de Utilizador

Ver a discografia de Terno Rei



	Nome	Origem	Reproduções	Álbuns	Músicas
	Terno Rei	Brasil	94	5	13
	Lô Borges	Brasil	0	0	1
	Ireneis Scott	Estados Unidos da Am...	29	2	18
	Paiva	Brasil	0	0	1
	Capitão Fausto	Portugal	30	1	10

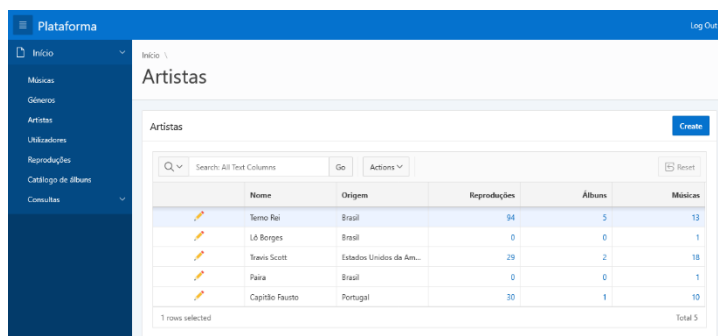
Para ver a discografia de Terno Rei basta clicar no número de álbuns que se encontra na coluna Álbuns da fila que tem o nome correspondente a Terno Rei.



Tipo	Artista	Álbum	Lançamento	Músicas	Duração
SINGLE	Terno Rei	Nada Igual	20-FEB-25	2	0:05:26
ALBUM	Terno Rei	Nenhuma Estrela	15-APR-25	13	0:40:17
EP	Terno Rei	Nenhuma Estrela	08-APR-25	4	0:11:27
SINGLE	Terno Rei	Próxima Parada	11-MAR-25	3	0:08:17
SINGLE	Terno Rei	Relógio	16-APR-25	1	0:02:44

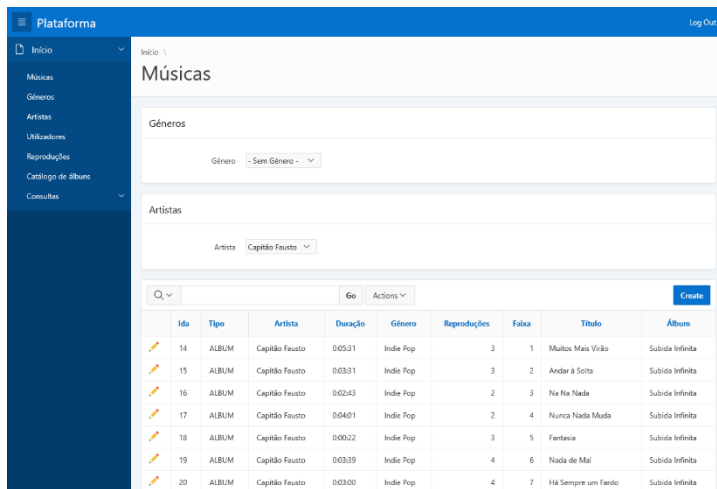
Ao clicar iremos ser levados à página que tem o catálogo dos álbuns. Já terá sido aplicado um filtro para mostrar apenas os álbuns do Terno Rei. Nesta página é possível ver os álbuns com o número de músicas e duração total que cada um tem.

Adicionar uma música ao álbum Subida Infinita dos Capitão Fausto

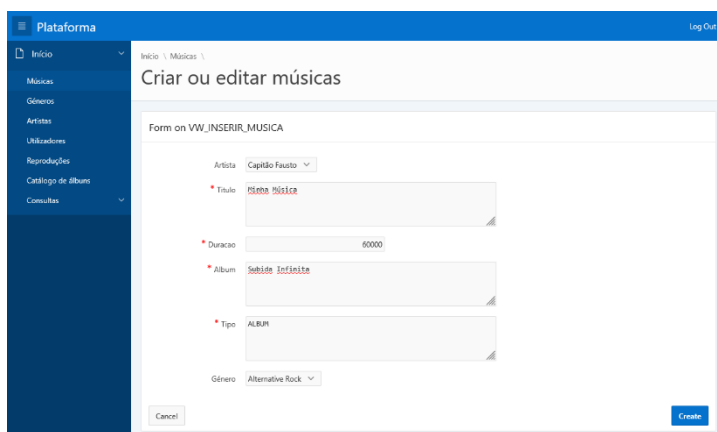


	Nome	Origem	Reproduções	Álbuns	Músicas
	Terno Rei	Brasil	94	5	13
	Lô Borges	Brasil	0	0	1
	Ireneis Scott	Estados Unidos da Am...	29	2	18
	Paiva	Brasil	0	0	1
	Capitão Fausto	Portugal	30	1	10

Vamos clicar no número de músicas correspondente ao Capitão Fausto para irmos para as músicas do artista inseridas na plataforma.

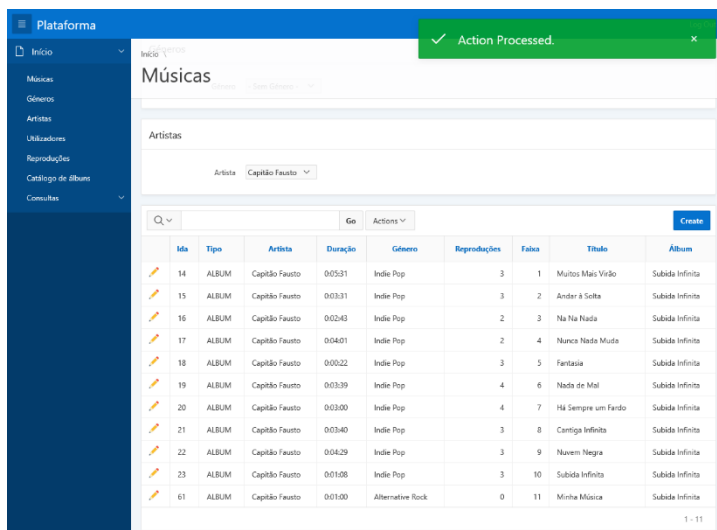


Nesta página vamos clicar no botão *Create* para adicionar uma música nova à plataforma



Nesta página vamos então preencher os espaços com a seguinte informação na imagem e clicar no botão *Create*.

Não esquecer que o álbum tem de existir e o artista estar associado ao álbum.



Como podemos ver, “Minha Música” foi inserida com sucesso na plataforma e encontra-se associada ao álbum, artista e género que definimos.

Agora vamos mudar o título da música que acabámos de criar. Para isso, clicamos no “lápiz” da música que queremos alterar.

Plataforma Log Out

Início \ Músicas \

Criar ou editar músicas

Form on VW_INSERT_MUSICA

Artista: Capitão Fausto

* Id: 3

* Título: A Colina Que Vejo

* Duração: 00000

* Album: Subida Infinita

* Tipo: ALBUM

* Idg: 1

Gênero: Alternative Rock

Cancel Delete Apply Changes

Vamos atualizar a música para ter o título como mostra na imagem. Depois clicamos no botão *Apply Changes*.

Plataforma ✓ Action Processed.

Início \ Músicas \

Músicas

Artistas

Artista: Capitão Fausto

Go Actions Cancel

	Id	Tipo	Artista	Duração	Gênero	Reproduções	Faixa	Título	Album
	14	ALBUM	Capitão Fausto	00531	Indie Pop	3	1	Muitos Mas Virão	Subida Infinita
	15	ALBUM	Capitão Fausto	00331	Indie Pop	3	2	Andar à Solta	Subida Infinita
	16	ALBUM	Capitão Fausto	00243	Indie Pop	2	3	Não Na Nada	Subida Infinita
	17	ALBUM	Capitão Fausto	00401	Indie Pop	2	4	Nunca Nada Muda	Subida Infinita
	18	ALBUM	Capitão Fausto	00022	Indie Pop	3	5	Fantasia	Subida Infinita
	19	ALBUM	Capitão Fausto	00339	Indie Pop	4	6	Nada de Mal	Subida Infinita
	20	ALBUM	Capitão Fausto	00300	Indie Pop	4	7	Há Sempre um Fim	Subida Infinita
	21	ALBUM	Capitão Fausto	00340	Indie Pop	3	8	Carteira Infinita	Subida Infinita
	22	ALBUM	Capitão Fausto	00429	Indie Pop	3	9	Nuvem Negra	Subida Infinita
	23	ALBUM	Capitão Fausto	00108	Indie Pop	3	10	Subida Infinita	Subida Infinita
	61	ALBUM	Capitão Fausto	00100	Alternative Rock	0	12	A Colina Que Vejo	Subida Infinita

1 - 11

Podemos ver que a nossa música foi atualizada para agora ter o título “A Colina Que Vejo”.

Versão APEX

A nossa aplicação APEX foi criada utilizando a versão do servidor local.

Utilização de Ferramentas IA

Na segunda fase do projeto, recorreremos a ferramentas de IA para auxiliar na pesquisa e resolução de problemas específicos encontrados durante a execução do projeto. Estas ferramentas foram utilizadas como apoio na realização das tarefas, nunca como substituto para a resolução de problemas que são da responsabilidade dos alunos implementar.

As ferramentas de IA foram aplicadas nas seguintes áreas:

- **Pesquisa de Problemas:** Para pesquisar e compreender melhor problemas técnicos específicos que surgiram durante a implementação do projeto.
- **Melhoria de Texto:** Para melhorar a fluidez e a coesão do texto escrito, aplicado nas seguintes secções do relatório: Apresentação do Tema; Limitações e Opções Tomadas na Implementação da BD; Consultas Interessantes; e Utilização de Ferramentas IA.

O uso de ferramentas de IA teve como objetivo principal melhorar a qualidade e a eficiência do trabalho realizado, sem substituir o esforço, a criatividade e os erros encontrados ao longo da implementação da base de dados.