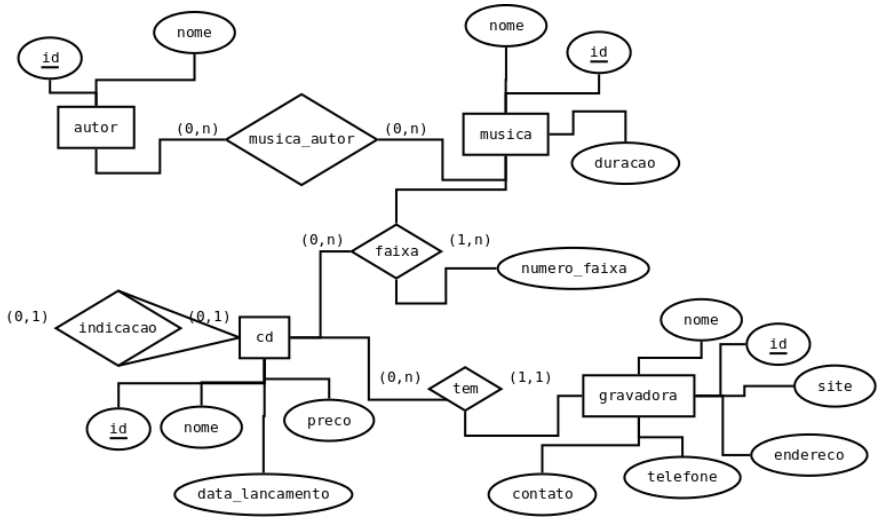
Banco de Dados **Nome:** _____
Prova - 1º Bimestre - Valor: 10 **Matrícula:** _____
Professor: Igor Avila Pereira

---

1. (2.0) Construa um DER para o gerenciamento de gravadoras de CD's onde:

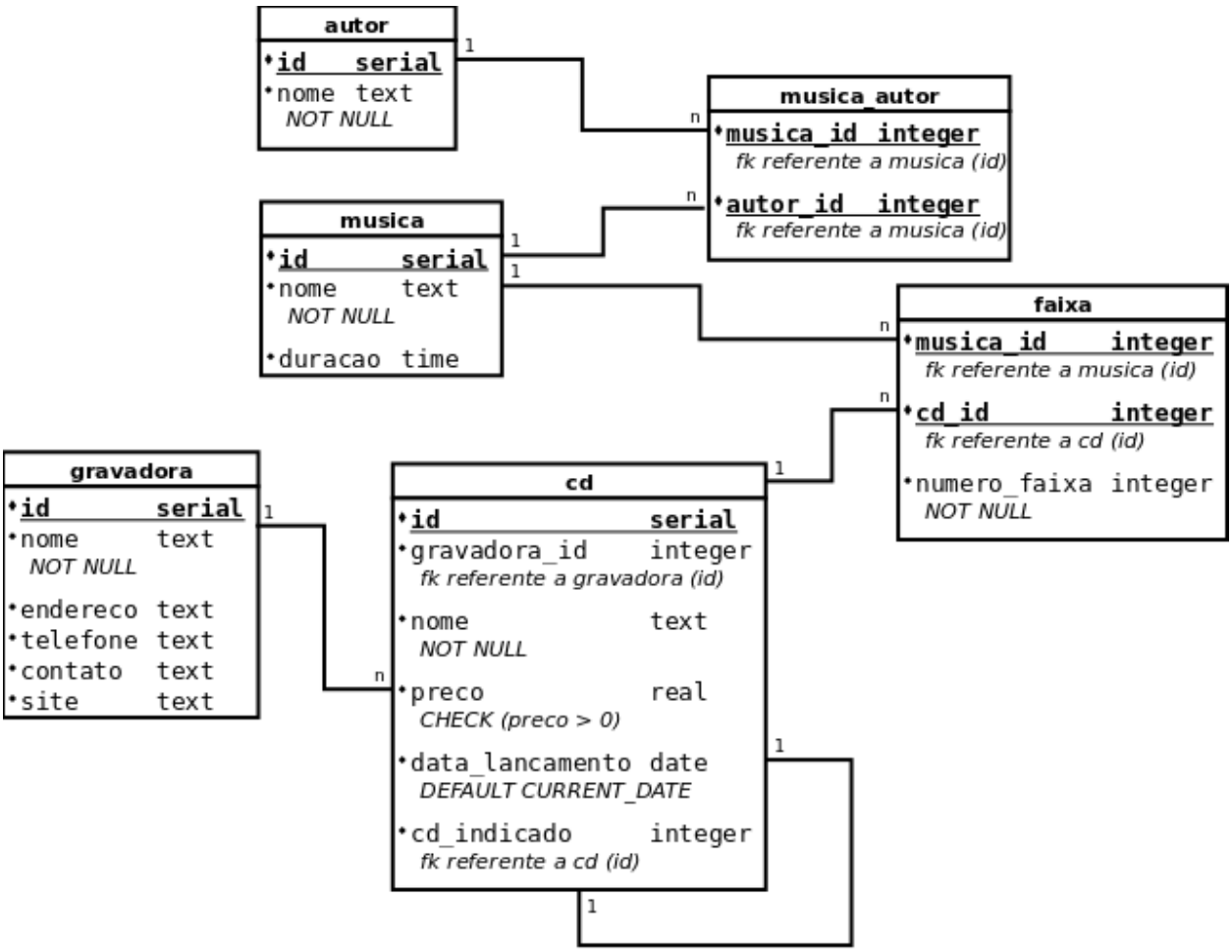
   - Um **autor** tem **id** e um **nome** (texto não-nulo);
   - Uma **música** tem **id**, **nome** (texto não-nulo) e **duração** (tempo);
   - Uma **gravadora** tem **id**, **nome** (texto não-nulo), **endereço** (texto), **telefone** (texto), **contato** (texto) e **site** (texto)
   - Um **CD** tem **id**, **nome**, **preço** (número real não-nulo e > 0), **data de lançamento** (data com valor padrão sendo a data atual)
   - Além disso:
     - Um **autor pode compor várias músicas e uma música pode ser composta por vários autores**;
     - **Um música pode estar em vários CD's e um CD pode ter várias músicas (faixas)**. Além disso, deve-se armazenar o **número da faixa** que cada música teve nos CD's que participou/integrou;
     - Músicas podem não ter nenhum autor;
     - **Uma gravadora produz vários CD's e um CD é produzido, exclusivamente, por uma única gravadora**;
     - **Para cada CD cadastrado na base dados é possível indicar um outro CD da base de dados, ou seja, cada CD pode recomendar um outro CD e, assim, sucessivamente**;
   - **Dica:** Use o DIA e cuidado com as cardinalidades n:n.

Gabarito:



2. (2.0) Construa o modelo relacional do exercício anterior.

   - **Dica:** Use o DIA, cuidado com as chaves primárias compostas e as tabelas intermediárias resultantes de relacionamentos n:n.



3. (2.0) Construa o *script.sql* de criação do B.D do modelo relacional desenvolvido no exercício anterior

   - **Dica:** CREATE DATABASE, CREATE TABLE, PRIMARY KEY, FOREIGN KEY, CHECK, NOT NULL e etc.

   Gabarito:

```
DROP DATABASE IF EXISTS gravadora;

CREATE DATABASE gravadora;

\c gravadora;
```

```
CREATE TABLE autor (
    id serial primary key,
    nome text not null
);

CREATE TABLE musica (
    id serial primary key,
    nome text not null,
    duracao time
);

CREATE TABLE musica_autor (
    musica_id integer references musica (id),
    autor_id integer references autor (id),
    primary key (musica_id, autor_id)
);

CREATE TABLE gravadora (
    id serial primary key,
    nome text not null,
    endereco text,
    telefone text,
    contato text,
    site text
);

CREATE TABLE cd (
    id serial primary key,
    gravadora_id integer references gravadora (id),
    nome text not null,
    preco real not null check (preco > 0),
    data_lancamento date DEFAULT CURRENT_DATE
);

CREATE TABLE faixa (
    musica_id integer references musica (id),
    cd_id integer references cd (id),
    numero_faixa integer not null,
    primary key (musica_id, cd_id)
);
```

4. (1.5) Além das instruções necessárias para criação do B.D, construa instruções SQL necessárias para **INSERIR** no B.D os seguintes dados:

   - **4 gravadoras**

- **4 autores**
  - Destes autores:
    - Pelo menos 1 com nome começando com R e terminando com O.
      - Ex: Renato Russo
- **4 cd's**
  - Destes cd's:
    - Pelo menos 2 cd's com datas de lançamento $\geq$ 1995 e $\leq$ 2000
  - **Dica:** Estas instruções SQL devem atribuir nome, preço, data de lançamento e alguma gravadora para estes cd's.
- **4 músicas**
  - Destas músicas:
    - Uma ou mais músicas **devem ter mais de um autor**;
    - Uma ou mais músicas devem **não ter autor**;
    - **Pelo menos 2 músicas devem ter sido composta por um mesmo autor**, ou em outras palavras, **um mesmo autor tem ser autor de mais de uma música cadastrada**;
  - **Dica:** Nas músicas com autor(es) é preciso criar também instruções SQL de inserção para tabelas intermediárias que possam surgir durante o mapeamento do DER para o Modelo Relacional (Ex: tabela **musica_autor**)

Gabarito:

```
INSERT INTO gravadora (nome) VALUES
('Universal'),
('Trama'),
('Sony'),
('Poligram');

INSERT INTO autor (nome) VALUES
('Peninha'),
('Renato Russo'),
('Michael Sullivan'),
('Massadas'),
('José Augusto');

INSERT INTO musica (nome) VALUES
('Alma Gemea'),
('País e Filhos'),
('Tarde De Domingo'),
('Que país é esse?'),
('Envolver');
```

```
INSERT INTO musica_autor (musica_id, autor_id) VALUES
(1,1),
(2,2),
(3,3),
(4,2),
(2,4);

INSERT INTO cd (nome, preco, data_lancamento, gravadora_id) VALUES
('Legião Urbana - CD', 10.0, '1990-05-05', 1),
('Tim maia', 23.0, '1995-04-06', 2),
('Fabio Jr.', 12.0, '2000-11-10', 1),
('Rita Lee.', 10.0, '2000-11-10', 1);
```

5. (0.5) Construa a instrução SQL que mostre os **nome** dos **autores** que tem o **nome começando com R e terminando com O**. Ex: Renato Russo, Roberto Gusmão e etc.

   - **Dica:** Funções de Manipulação de *Strings*

   Gabarito:

   ```
   SELECT nome FROM autor WHERE nome ILIKE 'R%O';
   ```

6. (0.5) Construa a instrução SQL que mostre somente as músicas que **NÃO** tem autores.

   - **Dica:** SUBSELECT ou EXCEPT

   Gabarito:

   ```
   SELECT musica.nome FROM musica
   WHERE id NOT IN (SELECT musica_id FROM musica_autor);
   ```

7. (0.5) Construa a instrução SQL que mostre o **nome** do CD e a **data de lançamento** dos CD's que foram lançados entre os anos de 1995 (inclusive) e 2000 (inclusive)

   - **Dica:** EXTRACT

   Gabarito:

   ```
   SELECT nome, data_lancamento FROM cd
   WHERE extract(year from data_lancamento) >= 1995
   AND extract(year from data_lancamento) <= 2000;
   ```

8. (0.5) Construa a instrução SQL que mostre para cada gravadora: o **nome**, o **preço médio** de seus CDs, o **maior preço** entre seus cd's, o **menor preço** entre seus cd's e **a quantidade** de CDs.

- **Dica:** Funções de agregação, Junção de Tabelas e GROUP BY

Gabarito:

```
SELECT gravadora.nome, avg(cd.preco),
max(cd.preco), min(cd.preco), count(*) FROM gravadora
INNER JOIN cd ON (gravadora.id = cd.gravadora_id)
GROUP BY gravadora.nome;
```

9. (0.5) Construa a instrução SQL que selecione o **nome** das músicas e a **quantidade de autores** das músicas que tem mais de um autor, ou seja, quantidade de autores ≥ 2

- **Dica:** Funções de agregação, Junção de Tabelas, GROUP BY e HAVING

Gabarito:

```
SELECT musica.nome, count(*) FROM autor
INNER JOIN musica_autor ON (autor.id = musica_autor.autor_id)
INNER JOIN musica ON (musica.id = musica_autor.musica_id)
GROUP BY musica.nome HAVING count(*) >= 2;
```