EXERCÍCIO 1

a. Explique o que é uma chave estrangeira

RESPOSTA: Chave estrangeira (foreign key) é o campo que estabelece o relacionamento entre duas tabelas.

---

b. Crie a tabela e, ao menos, uma avaliação para cada um dos filmes

RESPOSTA:

INSERT INTO Rating

VALUE (

‘004’,

‘Filme muito bom’,

9,

‘004’

);

---

c. Tente criar uma avaliação para um filme que não existe (ou seja, um id inválido). Anote e explique o resultado da query.

RESPOSTA:

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails

(`hamilton_ana_kempfer` . `Rating` , CONSTRAINT `Rating_ibfk_1` FOREIGN KEY (`movie_id`)

REFERENCES `Movie` (`id`))

Ele não permite a alteração na tabela e me informa que por causa da falta de uma informação indispensável que vem de outra tabela e lá ela não existe.

---

d. Altere a tabela de filmes para que ela não tenha mais uma coluna chamada rating.

RESPOSTA:

ALTER TABLE Movie DROP COLUMN rating;

---

e. Tente apagar um filme que possua avaliações. Anote e explique o resultado da query.

RESPOSTA:

Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails

(`hamilton_ana_kempfer` . `Rating` , CONSTRAINT `Rating_ibfk_1` FOREIGN KEY (`movie_id`)

REFERENCES `Movie` (`id`))

O erro me avisa que não pode deletar por que tem uma outra tabela que depende de informações contidas nessa tabel.

EXERCÍCIO 2

a. Explique, com as suas palavras, essa tabela

CREATE TABLE MovieCast (

movie_id VARCHAR(255),

actor_id VARCHAR(255),

FOREIGN KEY (movie_id) REFERENCES Movie(id),

FOREIGN KEY (actor_id) REFERENCES Actor(id)

);

RESPOSTA:

Essa tabela está sendo criada e usando como chave duas chaves estrangeiras e isso está escrito no FOREIGN KEY

---

b. Crie, ao menos, 6 relações nessa tabela

RESPOSTA:

INSERT INTO MovieCast

VALUE (

‘001’,

‘007’

);

---

c. Tente criar uma relação com um filme ou um ator inexistente. Anote e explique o resultado da query

RESPOSTA:

Error Code: 1452. Cannot add or update a child row: a foreign key constraint fails

(`hamilton_ana_kempfer`.`MovieCast`, CONSTRAINT `MovieCast_ibfk_1` FOREIGN KEY

(`movie_id`) REFERENCES `Movie` (`id`))

O erro avisa que não pode atualizar uma linha filha por que não foi achava a chave constritora.

---

d. Tente apagar um ator que possua uma relação nessa tabela. Anote e explique o resultado da query

RESPOSTA:

Error Code: 1451. Cannot delete or update a parent row: a foreign key constraint fails

(`hamilton_ana_kempfer`.`MovieCast`, CONSTRAINT `MovieCast_ibfk_2` FOREIGN KEY

(`actor_id`) REFERENCES `Actor` (`id`))

Não pode ser apagado por que é uma linha pai de outra tabela.

EXERCÍCIO 3

a. Explique, com suas palavras, a query acima. O que é o operador `ON` ?

SELECT * FROM Movie

INNER JOIN Rating ON Movie.id = Rating.movie_id;

RESPOSTA:

A query solicita que seja selecionado tudo da tabela MOvie que esteja relacionado com a tabela Rating onde o Movie.id for igual ao Rating.movie_id

---

b. Escreva uma query que retorne somente o nome, id e nota de avaliação dos filmes que já foram avaliados.

RESPOSTA:

SELECT Movie.id, title, rate FROM Movie

INNER JOIN Rating ON Movie.id = Rating.movie_id;

EXERCÍCIO 4

a. Escreva uma query que retorne todos os filmes e suas avaliações (com essa avaliação existindo ou não). A sua query deve retornar somente o nome, id, nota do filme e comentário

RESPOSTA:

SELECT Movie.id, title, rate, comment FROM Movie

INNER JOIN Rating ON Movie.id = Rating.movie_id;

---

b. Escreva uma query que retorne todas as relações de elenco, junto com as informações do filme. A sua query deve retornar o id do filme, título do filme e id do ator

RESPOSTA:

SELECT Movie.id, title, id FROM Movie

INNER JOIN MovieCast ON Movie.id = MovieCast.movie_id;

---

c. Escreva uma query que retorne a média das avaliações de todos os filmes agrupada em relação aos filmes (mesmo que ele não tenha sido avaliado ainda)

RESPOSTA:

SELECT AVG(rate) FROM Rating

INNER JOIN Movie ON Movie.id = Rating.movie_id;

EXERCÍCIO 5

a. Explique, com suas palavras essa query. Por que há a necessidade de dois `JOIN`?

SELECT * FROM Movie m

LEFT JOIN MovieCast mc ON m.id = mc.movie_id

JOIN Actor a ON a.id = mc.actor_id;

RESPOSTA:

É preciso usar dois JOIN's por que o código tem uma tabela intemediária

---

b. Altere a query para que ela retorne o id e o título do filme, e o id e o nome do ator. Coloque `alias` para facilitar o endentimento do retorno da query

RESPOSTA:

SELECT m.id as movie_id, m.title, a.id as actor_id, a.name FROM Movie m

LEFT JOIN MovieCast mc ON m.id = mc.movie_id

JOIN Actor a ON a.id = mc.actor_id;

---

*c. A query abaixo **deveria** ser a resposta do item b. Tente rodá-la. Anote e explique o resultado.

- Veja a query

  SELECT m.id as movie_id, m.title, a.id as actor_id, a.name FROM Movie m

  LEFT JOIN MovieCast mc ON m.id = mc.movie_id

  JOIN Actor a ON a.id = mc.actor_id;

RESPOSTA:

A query tornou uma tabela com o ID do filme e nome e o ID do ator e nome relacionado em todos os filmes que ele apareceu, isso aconteceu com todas as relações.

---

d. Desafio: Faça uma query que retorne todos os filmes com o nome de seus atores e as suas avaliações (nota e comentário)

RESPOSTA:

SELECT m.id as movie_id, m.title, a.id as actor_id, a.name, r.rate, r.comment

FROM Movie m

LEFT JOIN Rating r on r.movie_id = m.id

LEFT JOIN MovieCast mc ON m.id = mc.movie_id

JOIN Actor a ON a.id = mc.actor_id;

EXERCÍCIO 6

a. Que tipo de relação é essa?

RESPOSTA:

É uma relação M:N

---

b. Explicite a query que você usou para criar a tabela

RESPOSTA:

```
CREATE TABLE Movie_oscar(
id int PRIMARY KEY,
name VARCHAR(255) NOT NULL UNIQUE,
movie_id VARCHAR(255) NOT NULL UNIQUE,
date DATE NOT NULL,
FOREIGN KEY (movie_id)
REFERENCES Movie (id)
);
```

É uma tabela de criação que faz ligação e usa a informação de Id do filme que vem da tabela Movie.

---

c. Crie ao menos 2 óscar para cada um dos filmes

RESPOSTA:

```
INSERT INTO Movie_oscar
VALUE (
‘005’,
‘Oscar de melhor roteiro’,
‘001’,
‘2015-03-22’
);
```

---

d. Faça uma query que retorne todos os filmes e seus respectivos óscar

RESPOSTA:

```
SELECT * FROM Movie m
JOIN Movie_oscar mo ON m.id = mo.movie_id;
```