Este é o CS50

Introdução à Ciência da Computação (CS50)

OpenCourseWare

Doar (https://cs50.harvard.edu/donate)

David J. Malan (https://cs.harvard.edu/malan/)
malan@harvard.edu
(https://www.facebook.com/dmalan) (https://github.com/dmalan)
(https://www.instagram.com/davidjmalan/) (https://www.linkedin.com/in/malan/)
(https://www.reddit.com/user/davidjmalan)
(https://www.threads.net/@davidjmalan) (https://twitter.com/davidjmalan)

Filmes

Problema a resolver

Você recebeu um arquivo chamado `movies.db`, um banco de dados SQLite que armazena dados do IMDb (https://www.imdb.com/) sobre filmes, os diretores e atores que os estrelaram e suas avaliações. Escreva consultas SQL para responder a perguntas sobre este banco de dados de filmes.

Demonstração

```
sqlite> SELECT * FROM stars LIMIT 5;
+----------+-----------+
| movie_id | person_id |
+----------+-----------+
| 11801    | 459029    |
| 11801    | 681726    |
| 11801    | 692612    |
| 11801    | 726256    |
| 11801    | 776458    |
+----------+-----------+
sqlite> .quit
$
```

Recorded with **asciinema**

Começando

Para resolver este problema, você usará um banco de dados fornecido pela equipe do CS50.

▼ **Baixe o código de distribuição.**

Faça login em cs50.dev (https://cs50.dev/) , clique na janela do terminal e execute `cd` o comando. Você verá que o prompt do terminal será semelhante ao seguinte:

```
$
```

Em seguida, execute

```
wget https://cdn.cs50.net/2024/fall/psets/7/movies.zip
```

para baixar um arquivo ZIP chamado `movies.zip` para o seu espaço de código.

Em seguida, execute

```
unzip movies.zip
```

para criar uma pasta chamada `movies`. Você não precisa mais do arquivo ZIP, então pode executar

```
rm movies.zip
```

e responda com "y" seguido de Enter quando solicitado para remover o arquivo ZIP que você baixou.

Agora digite

```
cd movies
```

Em seguida, pressione Enter para entrar (ou seja, abrir) esse diretório. Seu prompt agora deve ser semelhante ao abaixo.

```
movies/ $
```

Execute o programa `ls` isoladamente e você deverá ver 13 arquivos .sql, bem como outros arquivos `movies.db`.

Se você encontrar algum problema, siga esses mesmos passos novamente e veja se consegue determinar onde errou!

Especificação

Para cada um dos problemas a seguir, você deve escrever uma única consulta SQL que retorne os resultados especificados. Sua resposta deve ser uma única consulta SQL, embora você possa aninhar outras consultas dentro dela. Você **não deve** presumir nada sobre os `id` nomes de filmes ou pessoas em particular: suas consultas devem ser precisas mesmo que os nomes `id` de qualquer filme ou pessoa sejam diferentes. Por fim, cada consulta deve retornar apenas os dados necessários para responder à pergunta: se o problema pedir apenas os nomes dos filmes, por exemplo, sua consulta não deve retornar também o ano de lançamento de cada filme.

Você pode verificar os resultados de suas pesquisas diretamente no IMDb (https://www.imdb.com/), mas lembre-se de que as avaliações no site podem ser diferentes das do IMDb `movies.db`, já que mais votos podem ter sido computados desde que baixamos os dados!

1. Em `1.sql`, escreva uma consulta SQL para listar os títulos de todos os filmes lançados em 2008.
    - Sua consulta deve retornar uma tabela com uma única coluna para o título de cada filme.
2. Em `2.sql`, escreva uma consulta SQL para determinar o ano de nascimento de Emma Stone.
    - Sua consulta deve retornar uma tabela com uma única coluna e uma única linha (sem contar o cabeçalho) contendo o ano de nascimento de Emma Stone.
    - Você pode presumir que existe apenas uma pessoa no banco de dados com o nome Emma Stone.
3. Em `3.sql`, escreva uma consulta SQL para listar os títulos de todos os filmes com data de lançamento igual ou posterior a 2018, em ordem alfabética.
    - Sua consulta deve retornar uma tabela com uma única coluna para o título de cada filme.

    - Filmes lançados em 2018 devem ser incluídos, assim como filmes com datas de lançamento futuras.
4. Em `4.sql`, escreva uma consulta SQL para determinar o número de filmes com uma classificação de 10,0 no IMDb.
    - Sua consulta deve retornar uma tabela com uma única coluna e uma única linha (sem contar o cabeçalho) contendo o número de filmes com classificação 10.0.
5. Em `5.sql`, escreva uma consulta SQL para listar os títulos e os anos de lançamento de todos os filmes de Harry Potter, em ordem cronológica.
    - Sua consulta deve retornar uma tabela com duas colunas, uma para o título de cada filme e outra para o ano de lançamento de cada filme.
    - Você pode presumir que o título de todos os filmes de Harry Potter começará com as palavras "Harry Potter" e que, se o título de um filme começar com as palavras "Harry Potter", trata-se de um filme de Harry Potter.
6. Em `6.sql`, escreva uma consulta SQL para determinar a classificação média de todos os filmes lançados em 2012.
    - Sua consulta deve retornar uma tabela com uma única coluna e uma única linha (sem contar o cabeçalho) contendo a avaliação média.
7. Em `7.sql`, escreva uma consulta SQL para listar todos os filmes lançados em 2010 e suas respectivas classificações, em ordem decrescente de classificação. Para filmes com a mesma classificação, ordene-os alfabeticamente pelo título.
    - Sua consulta deve retornar uma tabela com duas colunas, uma para o título de cada filme e outra para a classificação de cada filme.
    - Filmes sem classificação indicativa não devem ser incluídos nos resultados.
8. Em `8.sql`, escreva uma consulta SQL para listar os nomes de todas as pessoas que atuaram em Toy Story.
    - Sua consulta deve retornar uma tabela com uma única coluna para o nome de cada pessoa.
    - Você pode presumir que existe apenas um filme no banco de dados com o título Toy Story.
9. Em `9.sql`, escreva uma consulta SQL para listar os nomes de todas as pessoas que atuaram em um filme lançado em 2004, ordenados por ano de nascimento.
    - Sua consulta deve retornar uma tabela com uma única coluna para o nome de cada pessoa.
    - Pessoas com o mesmo ano de nascimento podem ser listadas em qualquer ordem.
    - Não precisa se preocupar com as pessoas que não têm o ano de nascimento listado, contanto que aquelas que têm estejam listadas em ordem.
    - Se uma pessoa apareceu em mais de um filme em 2004, ela só deve aparecer uma vez nos seus resultados.
10. Em `10.sql`, escreva uma consulta SQL para listar os nomes de todas as pessoas que dirigiram um filme que recebeu uma classificação de pelo menos 9,0.

    - Sua consulta deve retornar uma tabela com uma única coluna para o nome de cada pessoa.
    - Se uma pessoa dirigiu mais de um filme que recebeu uma classificação de pelo menos 9,0, ela só deve aparecer uma vez nos seus resultados.
11. Em `11.sql`, escreva uma consulta SQL para listar os títulos dos cinco filmes mais bem avaliados (em ordem) estrelados por Chadwick Boseman, começando pelo de maior classificação.
    - Sua consulta deve retornar uma tabela com uma única coluna para o título de cada filme.
    - Você pode presumir que existe apenas uma pessoa no banco de dados com o nome Chadwick Boseman.
12. Em `12.sql`, escreva uma consulta SQL para listar os títulos de todos os filmes em que Bradley Cooper e Jennifer Lawrence atuaram juntos.
    - Sua consulta deve retornar uma tabela com uma única coluna para o título de cada filme.
    - Você pode presumir que existe apenas uma pessoa no banco de dados com o nome Bradley Cooper.
    - Você pode presumir que existe apenas uma pessoa no banco de dados com o nome Jennifer Lawrence.
13. Em `13.sql`, escreva uma consulta SQL para listar os nomes de todas as pessoas que atuaram em um filme no qual Kevin Bacon também atuou.
    - Sua consulta deve retornar uma tabela com uma única coluna para o nome de cada pessoa.
    - Pode haver várias pessoas chamadas Kevin Bacon no banco de dados. Certifique-se de selecionar apenas o Kevin Bacon nascido em 1958.
    - O próprio Kevin Bacon não deveria ser incluído na lista resultante.

Dicas

Clique nos botões abaixo para ler algumas dicas!

▼ **Compreenda o esquema de`movies.db`**

Sempre que você interagir com um novo banco de dados, é melhor primeiro entender seu *esquema* . Em uma janela de terminal, execute o comando `sqlite3 movies.db` para que você possa começar a executar consultas no banco de dados.

Primeiro, quando `sqlite3` o sistema solicitar que você forneça uma consulta, digite-a `.schema` e pressione Enter. Isso exibirá as `CREATE TABLE` instruções usadas para gerar cada uma das tabelas no banco de dados. Ao examinar essas instruções, você poderá identificar as colunas presentes em cada tabela.

Observe que a `movies` tabela possui uma `id` coluna que identifica cada filme de forma única, além de colunas para o título `title` do filme e o `year` ano de lançamento. A `people` tabela também possui uma `id` coluna para o nome da pessoa `name` e colunas para o `birth` ano de lançamento.

As avaliações dos filmes, por sua vez, são armazenadas na `ratings` tabela. A primeira coluna da tabela é `movie_id` uma chave estrangeira que referencia o nome `id` do filme na `movies` tabela. O restante da linha contém dados sobre a avaliação `rating` de cada filme e a nota que `votes` o filme recebeu no IMDb.

Por fim, as tabelas `film` `stars` e ` `directors` diretor` relacionam as pessoas aos filmes em que atuaram ou dirigiram. (Apenas os atores e diretores principais (https://www.imdb.com/interfaces/) `movie_id` estão incluídos.) Cada tabela possui apenas duas colunas: `film` e `person_id` `pessoa`, que fazem referência a um filme e a uma pessoa específicos, respectivamente.

O desafio que você tem pela frente é escrever consultas SQL para responder a uma variedade de perguntas diferentes, selecionando dados de uma ou mais dessas tabelas.

▼ **Forme suas perguntas de maneira consistente.**

Consulte o arquivo sqlstyle.guide (https://www.sqlstyle.guide/) para dicas sobre boas práticas de estilo em SQL, especialmente à medida que suas consultas se tornam mais complexas!

▼ **Liste os títulos de todos os filmes lançados em 2008.**

Lembre-se de que você pode selecionar uma (ou mais) colunas de um banco de dados usando o comando `SELECT`, conforme o exemplo abaixo.

```
SELECT column0, column1 FROM table;
```

onde `column0` é o título de uma coluna e `column1` é o título de outra.

E lembre-se de que você pode filtrar as linhas retornadas em uma consulta com a `WHERE` palavra-chave, seguida de uma condição. Você também pode usar as opções `=` `&` `>` `, `&`, `<` `&` e outros operadores (https://www.w3schools.com/sql/sql_operators.asp) .

```
SELECT column FROM table
WHERE condition;
```

Consulte esta referência de palavras-chave SQL (https://www.w3schools.com/sql/sql_ref_keywords.asp) para obter algumas dicas de sintaxe SQL que podem ser úteis!

▼ **Determine o ano de nascimento de Emma Stone.**

Lembre-se de que uma `WHERE` cláusula pode avaliar condições não apenas com números, mas também com strings.

▼ Liste os títulos de todos os filmes com data de lançamento a partir de 2018, em ordem alfabética.

Tente dividir essa busca em duas etapas. Primeiro, encontre os filmes com data de lançamento igual ou posterior a 2018. Em seguida, coloque os títulos desses filmes em ordem alfabética.

Para encontrar os filmes com data de lançamento igual ou posterior a 2018, lembre-se de que uma condição em SQL suporta o uso de muitos operadores de comparação (https://www.w3schools.com/sql/sql_operators.asp) comuns , incluindo `>=` "maior ou igual a". Verifique se sua consulta retorna o número correto de filmes, conforme descrito em Como Testar .

Por fim, classifique os resultados da consulta em ordem alfabética pelo título. Lembre-se de que `ORDER BY` é possível classificar os dados por uma coluna nos resultados, conforme o exemplo abaixo.

```
...
ORDER BY column;
```

▼ Determine o número de filmes com uma classificação de 10,0 no IMDb.

Observe que esta pergunta não pede filmes *individuais* com classificação 10,0, mas sim o *número* de filmes com essa classificação. Em outras palavras, você deve coletar ("agregar") os resultados da sua consulta em um único número (o número de linhas). Lembre-se de que o SQL oferece suporte a uma "função de agregação" chamada `COUNT` , que você pode usar em uma coluna, conforme o exemplo abaixo.

```
SELECT COUNT(column)
FROM table;
```

▼ Liste os títulos e os anos de lançamento de todos os filmes de Harry Potter, em ordem cronológica.

Para esta consulta, provavelmente você desejará usar a `LIKE` palavra-chave do SQL. Lembre-se de que o SQL `LIKE` pode usar os chamados "caracteres curinga", como o caractere curinga `%` (&), que corresponderá a qualquer caractere (ou sequência de caracteres).

```
SELECT column0, column1
FROM table
WHERE column1 LIKE pattern;
```

▼ Determine a classificação média de todos os filmes lançados em 2012.

Aqui está outro exemplo de uma consulta na qual você precisará agregar dados. Considere `AVG` a função de agregação do SQL, para calcular uma média.

Considere também que esta consulta utiliza dados armazenados em duas tabelas separadas: `table1` `ratings` e ` `movies` table2`. Lembre-se de que, desde que uma tabela tenha uma chave estrangeira que corresponda a uma coluna em outra tabela, você pode combinar duas tabelas usando a `JOIN` palavra-chave `UNION` do SQL. Para usar a `JOIN` palavra-chave `UNION`, você deve especificar a tabela que deseja unir e a coluna pela qual deseja fazê-lo.

```
SELECT column0
FROM table0
JOIN table1 ON table0.column1 = table1.column2
```

▼ Liste todos os filmes lançados em 2010 e suas respectivas classificações, em ordem decrescente de classificação.

Lembre-se de que `ORDER BY` nem sempre é necessário classificar em ordem crescente. Você pode especificar que seus resultados sejam classificados em ordem *decrescente* adicionando um ponto `DESC`.

```
...
ORDER BY column DESC;
```

▼ Liste os nomes de todas as pessoas que atuaram em Toy Story.

Quando você se depara com uma consulta mais complexa como esta, o melhor é dividi-la em partes menores. No final, sua consulta deve retornar uma lista de nomes, como a abaixo.

```
-- Select names
SELECT name
FROM people
WHERE ...
```

Mas qual a melhor maneira de descobrir os nomes de quem estrelou Toy Story? Considere que a `people` tabela sozinha não contém essa informação (mas a `stars` tabela pode conter!). Na verdade, a `stars` tabela combina duas colunas, `person_id` e `movie_id`: qualquer pessoa com um `person_id` que esteja associado a Toy Story `movie_id` estrelou Toy Story.

```
-- Select names
SELECT name
FROM people
WHERE ...

-- Select person IDs
SELECT person_id
FROM stars
WHERE movie_id = ...
```

O próximo passo natural, portanto, é descobrir a identificação do filme Toy Story.

```sql
-- Select names
SELECT name
FROM people
WHERE ...

-- Select person IDs
SELECT person_id
FROM stars
WHERE movie_id = ...

-- Find Toy Story's ID
SELECT id
FROM movies
WHERE title = 'Toy Story';
```

Claro, você escreveu três consultas *separadas* . Mas observe que algumas consultas (as duas primeiras) estariam completas se incluíssem os resultados da consulta imediatamente abaixo delas. O processo de criar uma consulta que depende dos resultados de uma subconsulta é chamado de "aninhamento" de consultas. É uma dica e tanto, mas aqui está uma maneira de aninhar as consultas acima!

```sql
-- Select names
SELECT name
FROM people
WHERE id IN
(
    -- Select person IDs
    SELECT person_id
    FROM stars
    WHERE movie_id = (

        -- Select Toy Story's ID
        SELECT id
        FROM movies
        WHERE title = 'Toy Story'
    )
);
```

▼ Liste os nomes de todas as pessoas que atuaram em um filme lançado em 2004, em ordem cronológica de nascimento.

Observe que esta consulta, assim como a anterior, exige que você utilize dados de várias tabelas. Lembre-se de que é possível "aninhar" consultas em SQL, o que permite dividir uma consulta maior em consultas menores. Talvez você possa escrever consultas para...

1. Encontre os IDs dos filmes lançados em 2004.
2. Descubra os IDs das pessoas que atuaram nesses filmes.
3. Encontre os nomes das pessoas com esses IDs.

Em seguida, tente aninhar essas consultas para chegar a uma única consulta que retorne todas as pessoas que atuaram em um filme lançado em 2004. Considere como você poderia então

ordenar os resultados da sua consulta.

▼ Liste os nomes de todas as pessoas que dirigiram um filme que recebeu uma classificação de pelo menos 9,0.

Observe que esta consulta, assim como a anterior, exige que você utilize dados de várias tabelas. Lembre-se de que é possível "aninhar" consultas em SQL, o que permite dividir uma consulta maior em consultas menores. Talvez você possa escrever consultas para...

1. Encontre os IDs dos filmes com classificação igual ou superior a 9,0.
2. Descubra a identidade das pessoas que dirigiram esses filmes.
3. Encontre os nomes das pessoas com esses IDs.

Em seguida, tente aninhar essas consultas para chegar a uma única consulta que retorne os nomes de todas as pessoas que dirigiram um filme que recebeu uma classificação de pelo menos 9,0.

▼ Liste os títulos dos cinco filmes mais bem avaliados (em ordem) estrelados por Chadwick Boseman, começando pelo mais bem avaliado.

Observe que esta consulta, assim como a anterior, exige que você utilize dados de várias tabelas. Lembre-se de que é possível "aninhar" consultas em SQL, o que permite dividir uma consulta maior em consultas menores. Talvez você possa escrever consultas para...

1. Descubra a identidade de Chadwick Boseman.
2. Encontre os IDs dos filmes associados ao ID de Chadwick Boseman.
3. Encontre os títulos dos filmes com esses IDs de filme.

Em seguida, tente aninhar essas consultas para chegar a uma única consulta que retorne os títulos dos filmes de Chadwick Boseman.

A partir daí, você precisará determinar as classificações desses títulos e ordená-los por classificação, em ordem decrescente. Considere como você poderia combinar uma tabela relevante (provavelmente `ratings`!) e ordenar os resultados por uma coluna relevante.

Por fim, estude a `LIMIT` (https://www.sqlitetutorial.net/sqlite-limit/)palavra-chave do SQL, que retornará o resultado principal.`n`linhas em uma consulta.

▼ Liste os títulos de todos os filmes em que Bradley Cooper e Jennifer Lawrence atuaram juntos.

Observe que esta consulta, assim como a anterior, exige que você utilize dados de várias tabelas. Lembre-se de que é possível "aninhar" consultas em SQL, o que permite dividir uma consulta maior em consultas menores. Talvez você possa escrever consultas para...

1. Descubra a identidade de Bradley Cooper.
2. Descubra a identidade de Jennifer Lawrence.

3. Encontre os IDs dos filmes associados ao ID de Bradley Cooper.
4. Encontre os IDs dos filmes associados ao ID de Jennifer Lawrence.
5. Encontre títulos de filmes a partir dos IDs de filmes associados a *Bradley* Cooper e Jennifer Lawrence.

Em seguida, tente aninhar essas consultas para chegar a uma única consulta que retorne os filmes em que Bradley Cooper e Jennifer Lawrence atuaram juntos.

Lembre-se de que você pode criar condições compostas em SQL usando `AND` OR `OR`.

▼ **Liste os nomes de todas as pessoas que atuaram em um filme em que Kevin Bacon também atuou.**

Observe que esta consulta, assim como a anterior, exige que você utilize dados de várias tabelas. Lembre-se de que é possível "aninhar" consultas em SQL, o que permite dividir uma consulta maior em consultas menores. Talvez você possa escrever consultas para...

1. Descubra a identidade de Kevin Bacon (o que nasceu em 1958!).
2. Encontre os IDs dos filmes associados ao ID de Kevin Bacon.
3. Encontre os IDs das pessoas associadas a esses IDs de filmes.
4. Encontre os nomes das pessoas com esses IDs.

Em seguida, tente aninhar essas consultas para chegar a uma única consulta que retorne os nomes de todas as pessoas que atuaram em um filme no qual Kevin Bacon também atuou.
Lembre-se de que você vai querer excluir o próprio Kevin Bacon dos resultados!

Passo a passo

Uso

Para testar suas consultas no VS Code, você pode consultar o banco de dados executando o seguinte comando:

```
$ cat filename.sql | sqlite3 movies.db
```

Onde `filename.sql` está o arquivo que contém sua consulta SQL?

Você também pode executar

```
$ cat filename.sql | sqlite3 movies.db > output.txt
```

Para redirecionar a saída da consulta para um arquivo de texto chamado `output.txt`. (Isso pode ser útil para verificar quantas linhas são retornadas pela sua consulta!)

Como testar

Embora `check50` exista uma solução para este problema, recomendamos que você teste seu código por conta própria para cada um dos itens a seguir. Você pode executar o comando `sqlite3 movies.db` para realizar consultas adicionais no banco de dados e garantir que o resultado esteja correto.

Se você estiver usando o `movies.db` banco de dados fornecido na distribuição deste conjunto de problemas, você deverá constatar que

- A execução `1.sql` resulta em uma tabela com 1 coluna e 10.494 linhas.
- A execução `2.sql` resulta em uma tabela com 1 coluna e 1 linha.
- A execução `3.sql` resulta em uma tabela com 1 coluna e 130.992 linhas.
- A execução `4.sql` resulta em uma tabela com 1 coluna e 1 linha.
- A execução `5.sql` resulta em uma tabela com 2 colunas e 14 linhas.
- A execução `6.sql` resulta em uma tabela com 1 coluna e 1 linha.
- A execução `7.sql` resulta em uma tabela com 2 colunas e 7.295 linhas.
- A execução `8.sql` resulta em uma tabela com 1 coluna e 10 linhas.
- A execução `9.sql` resulta em uma tabela com 1 coluna e 35.612 linhas.
- A execução `10.sql` resulta em uma tabela com 1 coluna e 4.726 linhas.
- A execução `11.sql` resulta em uma tabela com 1 coluna e 5 linhas.
- A execução `12.sql` resulta em uma tabela com 1 coluna e 4 linhas.
- A execução `13.sql` resulta em uma tabela com 1 coluna e 553 linhas.

Observe que a contagem de linhas não inclui as linhas de cabeçalho que mostram apenas os nomes das colunas.

Se a sua consulta retornar um número de linhas ligeiramente diferente do esperado, certifique-se de que está tratando corretamente os duplicados! Para consultas que solicitam uma lista de nomes, nenhuma pessoa deve ser listada duas vezes, mas duas pessoas diferentes com o mesmo nome devem ser listadas separadamente.

Correção

```
check50 cs50/problems/2025/x/movies
```

Como enviar

```
submit50 cs50/problems/2025/x/movies
```

Agradecimentos

Informações cedidas pelo IMDb (imdb.com (https://www.imdb.com)). Usadas com permissão.