Conhecendo subconsultas, seu arsenal de ferramentas em SQL fica ainda mais completo.

Você agora pode juntar muitas camadas de processamento de dados em uma só consulta e reduzir o número de passos até respostas a perguntas mais complexas.

Pratique os conhecimentos em subconsultas e responda às seguintes perguntas.

Submeta suas respostas em um arquivo com extensão .sql, e use comentários (com os caracteres -- ou /\* \*/) para escrever suas respostas por extenso e delimitar as questões.

1. Se você é do tempo da Blockbuster, deve lembrar que os filmes mais famosos ou mais novos eram oferecidos em maior quantidade para aluguel -- já que,

naturalmente, a demanda por eles era maior que a média. Vamos verificar essa premissa na nossa locadora usando o banco de dados sakila!

a. Escreva uma consulta que liste, em ordem decrescente, os filmes mais alugados na nossa locadora. Utilize a tabela rental para isso, relacionando com a tabela inventory.

SELECT i.film\_id, f.title, COUNT(r.rental\_id) as total\_locacoes

FROM rental AS r

INNER JOIN inventory AS i

ON r.inventory\_id = i.inventory\_id

INNER JOIN film AS f

ON i.film\_id = f.film\_id

GROUP BY i.film\_id

ORDER BY total\_locacoes DESC, film\_id ASC;

b. Escreva uma consulta que liste, em ordem decrescente, o número de unidades disponíveis de cada filme na locadora.

SELECT i.film\_id, f.title, COUNT(i.inventory\_id) as total\_estoque

FROM film AS f

INNER JOIN inventory AS i

ON i.film\_id = f.film\_id

GROUP BY i.film\_id

ORDER BY total\_estoque DESC, film\_id ASC;

c. Usando subconsultas, relacione as tabelas resultantes dos itens a) e b) e responda:

os títulos mais alugados de fato correspondem aos que têm maior número de itens disponíveis?

Sim.

Qual consulta você usou para chegar a esse resultado?

WITH locacoes AS(

SELECT i.film\_id, f.title, COUNT(r.rental\_id) as total\_locacoes

FROM rental AS r

INNER JOIN inventory AS i

ON r.inventory\_id = i.inventory\_id

INNER JOIN film AS f

ON i.film\_id = f.film\_id

GROUP BY i.film\_id

ORDER BY total\_locacoes DESC, film\_id ASC),

estoque AS(

SELECT i.film\_id, f.title, COUNT(i.inventory\_id) as total\_estoque

FROM film AS f

INNER JOIN inventory AS i

ON i.film\_id = f.film\_id

GROUP BY i.film\_id

ORDER BY total\_estoque DESC, film\_id ASC)

SELECT l.\*, e.total\_estoque

FROM locacoes AS l

INNER JOIN estoque AS e

ON l.film\_id = e.film\_id

GROUP BY l.film\_id

ORDER BY total\_locacoes DESC;

2. A view chamada sales\_by\_film\_category traz o resultado total de vendas em aluguéis por categoria dos filmes.

Escreva uma consulta que complemente essa tabela com as seguintes métricas para cada categoria:

Valor médio do aluguel dos filmes por categoria;

• Valor médio do custo de reposição dos filmes por categoria;

• Total de vendas por categoria na loja 1; Total de vendas por categoria na loja 2; (dica: use o comando CASE WHEN)

• Total de títulos disponíveis por categoria (dica: use a tabela inventory)

SELECT fc.category\_id,

c.name,

SUM(p.amount) AS valor\_total\_alugueis,

AVG(p.amount) AS valor\_medio\_alugueis,

AVG(f.replacement\_cost) AS valor\_medio\_custo\_reposicao,

SUM(CASE WHEN i.store\_id = 1 THEN p.amount ELSE 0 END) AS valor\_total\_alugueis\_loja\_1,

SUM(CASE WHEN i.store\_id = 2 THEN p.amount ELSE 0 END) AS valor\_total\_alugueis\_loja\_2,

COUNT(i.inventory\_id) AS total\_titulos

FROM category AS c

INNER JOIN film\_category AS fc

ON c.category\_id = fc.category\_id

INNER JOIN inventory AS i

ON fc.film\_id = i.film\_id

INNER JOIN film AS f

ON f.film\_id = i.film\_id

INNER JOIN rental AS r

ON i.inventory\_id = r.inventory\_id

INNER JOIN payment AS p

ON r.rental\_id = p.rental\_id

GROUP BY fc.category\_id;