Atividade - Consultas 2

Para a atividade deve-se executar o arquivo pagila-insert-data.sql disponibilizado no postgresql.

Essa é uma base de dados que contém dados sobre uma rede de lojas de locação de filmes.

Para facilitar a criação das sqls de seleção, o diagrama da base pode ser visto em: <https://dataedo.com/samples/html/Pagila/doc/Pagila_10/modules/Paglia_database_diagram_103/module.html>.

TODAS AS SQLS SERÃO TESTADAS, ATENÇÃO!

1. Crie uma sql de consulta que traga os seguintes dados: idioma(language.name), quantidade de filmes (count(film\_id)) para cada língua. VALOR(10 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT language.name, count(film.film\_id) FROM public.language  JOIN public.film ON language.language\_id = film.language\_id  GROUP BY language.name; |

1. Crie uma sql de consulta que traga os seguintes dados: nome da categoria(category.name), quantidade de filmes (count(film\_id)) para cada categoria.VALOR(10 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT category.name, count(film.film\_id) FROM public.category  JOIN public.film\_category ON category.category\_id = film\_category.category\_id  JOIN public.film ON film\_category.film\_id = film.film\_id  GROUP BY category.name; |

1. Crie uma sql de consulta que traga os seguintes dados: nome do filme(film.title), quantidade de aluguéis (count(r.rental\_id)), soma da quantidade de dinheiro proveniente de aluguéis desse filme (sum(payment.amount)). Ordene pelo nome do filme (film.title) VALOR(20 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT film.title, count(rental.rental\_id), sum(payment.amount) FROM public.film  JOIN public.inventory ON film.film\_id = inventory.film\_id  JOIN public.rental ON inventory.inventory\_id = rental.inventory\_id  JOIN public.payment ON rental.rental\_id = payment.rental\_id  GROUP BY film.title  ORDER BY film.title; |

1. Crie uma sql de consulta que traga os seguintes dados: primeiro nome do ator (actor.first\_name), sobrenome do ator (actor.last\_name), quantidade de filmes (count(film\_id)). Ordene (Ascendente) a busca pelo primeiro nome prioritariamente e o sobrenome secundariamente (a ideia é trazer o número de filmes por ator - (recomendação: agrupar pelo id do ator) VALOR(20 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT actor.actor\_id, actor.first\_name, actor.last\_name, count(film\_actor.film\_id)  FROM public.actor  JOIN public.film\_actor ON actor.actor\_id = film\_actor.actor\_id  GROUP BY actor.actor\_id, actor.first\_name, actor.last\_name  ORDER BY actor.first\_name ASC, actor.last\_name ASC; |

1. Crie uma sql de consulta que traga os seguintes dados: primeiro nome do ator (actor.first\_name), sobrenome do ator (actor.last\_name), nome da categoria(category.name), quantidade de filmes (count(film\_id)). Ordene (Ascendente) a busca pelo primeiro nome prioritariamente e o sobrenome secundariamente (a ideia é trazer o número de filmes por categoria de cada ator - (recomendação: agrupar pelo id do ator) VALOR(20 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT actor.actor\_id, actor.first\_name, actor.last\_name, category.name, count(film.film\_id) FROM public.actor  JOIN public.film\_actor ON actor.actor\_id = film\_actor.actor\_id  JOIN public.film ON film\_actor.film\_id = film.film\_id  JOIN public.film\_category ON film.film\_id = film\_category.film\_id  JOIN public.category ON film\_category.category\_id = category.category\_id  GROUP BY actor.actor\_id, actor.first\_name, actor.last\_name, category.name  ORDER BY actor.first\_name ASC, actor.last\_name ASC; |

1. Crie uma sql de consulta que traga os seguintes dados: primeiro nome do ator (actor.first\_name), sobrenome do ator (actor.last\_name), nome da categoria(category.name), quantidade de filmes (count(film\_id)), mas apenas de categorias que tenham mais do que 3 filmes no total. Ordene (Ascendente) a busca pelo primeiro nome prioritariamente e o sobrenome secundariamente (a ideia é trazer o número de filmes por categoria de cada ator - (recomendação: agrupar pelo id do ator) VALOR(20 pontos)

coloque abaixo a sql

|  |
| --- |
| SELECT actor.actor\_id, actor.first\_name, actor.last\_name, category.name, count(film.film\_id) FROM public.actor  JOIN public.film\_actor ON actor.actor\_id = film\_actor.actor\_id  JOIN public.film ON film\_actor.film\_id = film.film\_id  JOIN public.film\_category ON film.film\_id = film\_category.film\_id  JOIN public.category ON film\_category.category\_id = category.category\_id  GROUP BY actor.actor\_id, actor.first\_name, actor.last\_name, category.name  HAVING count(film.film\_id) > 3  ORDER BY actor.first\_name ASC, actor.last\_name ASC; |