**# QUESTÃO 1**

SELECT

f.release\_year, f.language\_id, f.original\_language\_id, f.rental\_duration, f.rental\_rate, f.length, f.replacement\_cost, f.rating, f.special\_features,

fa.actor\_id, fa.last\_update,

fc.category\_id, fc.film\_id,

ft.title, ft.description

FROM film f

INNER JOIN film\_actor fa on fa.film\_id=f.film\_id

INNER JOIN film\_category fc on fc.film\_id=f.film\_id

INNER JOIN inventory iv on iv.film\_id=f.film\_id

INNER JOIN film\_text ft on iv.film\_id=ft.film\_id;

**ou**

SELECT

f.release\_year, f.language\_id, f.original\_language\_id, f.rental\_duration, f.rental\_rate, f.length, f.replacement\_cost, f.rating, f.special\_features,

fa.actor\_id, fa.last\_update,

fc.category\_id, fc.film\_id,

ft.title, ft.description

FROM film f

INNER JOIN film\_actor fa on fa.film\_id=f.film\_id

INNER JOIN film\_category fc on fc.film\_id=f.film\_id

INNER JOIN film\_text ft on f.film\_id=ft.film\_id;

**#QUESTÃO 2**

SELECT

c.first\_name as'nome', c.last\_name as 'sobrenome',c.email as 'email', ci.city as'cidade do endereço',

(CASE WHEN c.store\_id=1 THEN 'Lethbridge'

ELSE 'Woodridge' END) as'cidade da loja de cadastro'

FROM payment p

INNER JOIN customer c on c.customer\_id=p.customer\_id

INNER JOIN address ad on ad.address\_id=c.address\_id

INNER JOIN city ci on ci.city\_id=ad.city\_id

GROUP BY c.customer\_id HAVING COUNT(p.customer\_id)>40;

**# ou**

SELECT

c.first\_name as'nome', c.last\_name as 'sobrenome',c.email as 'email', ci.city as'cidade do endereço',

(CASE WHEN c.store\_id=1 THEN 'Lethbridge'

ELSE 'Woodridge' END) as'cidade da loja de cadastro'

FROM rental r

INNER JOIN customer c on c.customer\_id=r.customer\_id

INNER JOIN address ad on ad.address\_id=c.address\_id

INNER JOIN city ci on ci.city\_id=ad.city\_id

GROUP BY c.customer\_id HAVING COUNT(r.customer\_id)>40

**# QUESTÃO 3**

SELECT \* FROM sakila.inventory i

INNER JOIN rental r on r.inventory\_id=i.inventory\_id

INNER JOIN film f on f.film\_id=i.film\_id

NOTA: a ideia da tabela inventory é’ juntar’ todos os filmes de cada loja. assim o consultor pode usar ela como ferramenta para separar os filmes de cada loja, ou seja, do total de filmes quantos e quais filmes estão em cada loja, e do total de aluguéis quantos estão em cada loja. assim também dá pra saber o valor que cada loja arrecadou. (deu até vontade de fazer essas consultas kkkkk) se bem que usar só o store\_id já resolveria, um caminho mais fácil. Não entendi essa questão 3 achei fácil kkk, devo ter feito e entendido errado

**#QUESTÃO 4**

SELECT SUM(f.rental\_rate) as'soma dos alugueis ainda não entregues'

FROM rental r

INNER JOIN inventory i on i.inventory\_id=r.inventory\_id

INNER JOIN film f on f.film\_id=i.film\_id

where r.return\_date is null;

o valor é de 515,17 reais

**#ou para verificar aluguel por aluguel**

SELECT r.rental\_id, r.return\_date, f.rental\_rate as' alugueis que ainda não entregues'

FROM rental r

INNER JOIN inventory i on i.inventory\_id=r.inventory\_id

INNER JOIN film f on f.film\_id=i.film\_id

where r.return\_date is null

group by 1,2;

nota: ao verificar manualmente na tabela não achei nenhum return date null, porém ele me retornou um valor ao consultar o primeiro select e no segundo select me retornou uma lista de datas, estranho…