Além da sintaxe INNER JOIN, estudamos diversas outras opções de relacionamentos entre tabelas SQL, que a partir de agora te permitem explorar as mais diversas bases de dados. Não haverá um dia no seu trabalho com SQL em que você não irá fazer um relacionamento!

Antes de caminharmos para o próximo módulo, estude o conteúdo e responda às perguntas:

1. A gestão da locadora decidiu que no mês de julho de 2005 haverá um prêmio para o funcionário que mais registrou vendas, e pediu sua ajuda para determinar a premiação.
   1. Usando a tabela *staff* como *driving table*, escreva uma consulta SQL que traga todas as vendas realizadas por cada funcionário no mês de julho de 2005. Não é permitido utilizar o filtro WHERE!

*SELECT*

*staff.first\_name,*

*staff.last\_name,*

*COUNT(payment.customer\_id) as "contagem vendas"*

*FROM sakila.staff*

*INNER JOIN sakila.payment ON payment.staff\_id = staff.staff\_id*

*AND payment.payment\_date BETWEEN '2005-07-01' AND '2005-07-31'*

*GROUP BY*

*staff.first\_name,*

*staff.last\_name;*

* 1. Qual cláusula de relacionamento você utilizou no item a), e por quê? Quais seriam as consequências de utilizar outro tipo de relacionamento?

R- INNER JOIN porque traz somente os registros com a chave em comum.

O OUTER JOIN traria a união de tudo ou a diferença entre as tabelas.

* 1. Altere essa consulta para trazer o total em pagamentos processados por cada funcionário.

*SELECT*

*staff.first\_name,*

*staff.last\_name,*

*sum(payment.amount) as "total"*

*FROM sakila.staff*

*INNER JOIN sakila.payment ON payment.staff\_id = staff.staff\_id*

*AND payment.payment\_date BETWEEN '2005-07-01' AND '2005-07-31'*

*GROUP BY*

*staff.first\_name,*

*staff.last\_name;*

* 1. Por fim, responda: qual funcionário deve ganhar o prêmio? Qual foi o valor total de vendas no mês?

*R – Mike, o total de vendas foi $ 25500,07.*

1. Na tarefa do módulo anterior, você descobriu que alguns aluguéis ainda não foram devolvidos. Precisamos buscar esses itens! Escreva uma consulta que retorne a lista de emails das pessoas que estão com aluguéis pendentes, e o número de itens a serem retornados.

*SELECT DISTINCT c.email,*

*COUNT(r.rental\_id) AS total\_itens\_retornar*

*FROM sakila.rental AS r*

*INNER JOIN sakila.inventory AS i ON r.inventory\_id = i.inventory\_id*

*INNER JOIN sakila.film AS f ON f.film\_id = i.film\_id*

*INNER JOIN sakila.customer AS c ON r.customer\_id = c.customer\_id*

*WHERE r.return\_date IS NULL*

*GROUP BY c.email*

*ORDER BY total\_itens\_retornar DESC;*

Bônus opcional: utilize a função GROUP\_CONCAT, que estudamos no módulo 5, para retornar também uma lista com o nome dos filmes que devem ser devolvidos.

*SELECT GROUP\_CONCAT(f.title) AS titulos\_filmes,*

*c.email,*

*COUNT(r.rental\_id) AS total\_itens\_retornar*

*FROM sakila.rental AS r*

*INNER JOIN sakila.inventory AS i ON r.inventory\_id = i.inventory\_id*

*INNER JOIN sakila.film AS f ON f.film\_id = i.film\_id*

*INNER JOIN sakila.customer AS c ON r.customer\_id = c.customer\_id*

*WHERE r.return\_date IS NULL*

*GROUP BY c.email*

*ORDER BY total\_itens\_retornar DESC;*