WebDev - Módulo 4
Roteiro de aula

Aula 9 - Hard: Tudo junto e misturado

Tópicos da aula:

- Inner joins
- Left join
- Subquery

Objetivos da aula:

1. Explicar a técnica de união de tabelas;
2. Identificar características do INNER JOIN;;
3. Produzir consultas utilizando INNER JOIN e outras mais complexas.
4. Compreender o LEFT JOIN, outros tipos de JOIN e casos diversos de união de tabelas;
5. Operar consultas utilizando LEFT JOIN;
6. Descrever o conceito de subqueries, incluindo-o como alternativa para o HAVING;
7. Operar consultas complexas utilizando subconsultas.

Atividades da Aula

➔ **Atividade: Juntando as peças**

- Ainda considerando o nosso banco SAKILA, vamos ter que completar alguns relatórios. Para isso, vocês deverão construir as consultas para responder às seguintes perguntas:
  - Nome e email dos clientes com o valor que gastaram em aluguéis?
  - No relatório de faturamento geral, qual o nome de cada um dos funcionários?
  - Os nomes dos atores e os filmes que participaram?
  - Nomes do cliente e funcionário envolvidos em cada aluguel?”

➔ **Atividade: Aquecendo os motores**

- Utilizando o Sakila e com a operação de **left join**, realize as consultas que correspondam a:
  - Os endereços cadastrados que estão vinculados apenas aos clientes;
  - As línguas que não possuem nenhum filme cadastrado na base;
  - Atores que não participaram de nenhum filme;

➔ **Atividade: Subconsultas de hoje**

- Utilizando o Sakila, sem usar o HAVING, façam as consultas que correspondam a:
  - Quais atores atuaram em mais de 15 filmes?
  - Quais clientes fizeram 29 pedidos?
  - Numa nova campanha publicitária, o público alvo são os clientes que gastaram entre 70 e 100 dólares. Quais os nomes desses clientes e quanto eles gastaram?

Momento Aprendizagem em Pares

➔ Esse momento é dedicado para vocês desenvolverem suas demandas e entregas para o curso.

➔ Utilize esse tempo da maneira que preferir, mas atente-se às aulas que você deve realizar as entregas.

- **Dica:** Nas Propostas dos projetos, vocês encontram uma sugestão de organização para a realização das atividades.
- **Lembre-se:** O momento de Aprendizagem em Pares é justamente para fazer trocas e aprender em comunidade, aproveite seus colegas e se desenvolvam juntos!

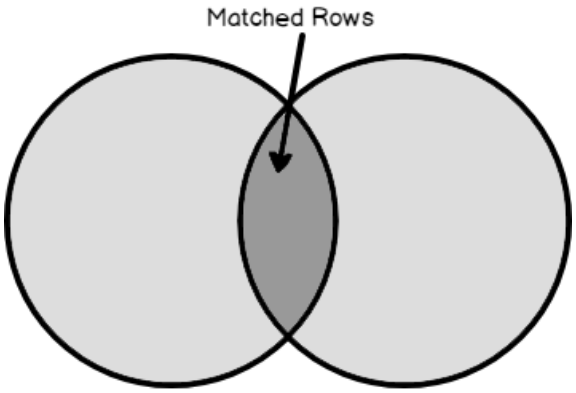
➔ **Entregas:**

- Projeto individual: aula 5 - HARD
- Projeto em grupo: aula 10 - HARD
- Apresentação do projeto: aula 10 - HARD

Revisão da aula

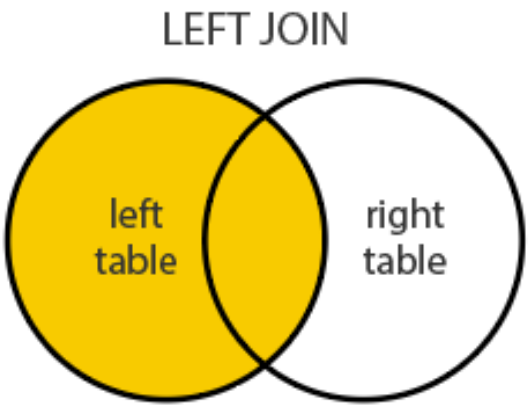
➔ No SQL, você usa **JOIN** para combinar dados de duas tabelas com base em uma coluna com valores correspondentes. Por exemplo, se você tiver uma tabela com as informações do cliente (ID do cliente, nome, sobrenome etc.) você pode unir essas duas tabelas com base nas colunas que armazenam os números de ID do cliente. Isso fornecerá detalhes do pedido e do cliente para cada cliente.

➔ A cláusula **Inner Join** no SQL Server **cria uma nova tabela (não física) combinando linhas que possuem valores correspondentes em duas ou mais tabelas**. Essa junção é baseada em um relacionamento lógico (ou um campo comum) entre as tabelas e é usada para recuperar dados que aparecem em ambas as tabelas.
Por exemplo: Suponha que temos duas tabelas, Tabela A e Tabela B, que gostaríamos de unir usando SQL Inner Join. O resultado dessa junção será um novo conjunto de resultados que retorna linhas correspondentes em ambas as tabelas. A parte da interseção em cinza escuro abaixo mostra os dados recuperados usando Inner Join no SQL Server. As palavras “matches rows” da figura são traduzidas como “corresponde às linhas”.



SQL Inner Join

➔ Um **LEFT JOIN** é usado em uma instrução **SELECT** para fornecer acesso a colunas em duas ou mais tabelas. Ele retornará dados de coluna para todas as linhas na primeira tabela de seu **JOIN** (também conhecida como tabela 'esquerda') e apenas dados de coluna para linhas correspondentes na segunda tabela (também conhecida como tabela 'direita').



➔ A cláusula **HAVING** no MySQL também é usada para filtrar os dados, assim como a cláusula where. HAVING no MySQL **filtra as linhas do conjunto de resultados intermediário** que é construído usando as cláusulas FROM, WHERE ou GROUP BY em uma instrução SELECT.

➔ HAVING no MySQL é normalmente usada com uma cláusula **GROUP BY**. Isso significa que a cláusula **HAVING** é usada em combinação com uma cláusula GROUP BY para restringir o número de grupos a serem retornados satisfazendo a condição especificada usando a HAVING.

Para ajudar

Links interessantes:

- Aprendendo a fazer JOINs: INNER, LEFT, RIGHT, FULL, CROSS E SELF: https://consultabd.wordpress.com/2020/12/06/aprendendo_joins/
- Criando uma Subconsulta para Recuperar Dados de Mais de uma Tabela: https://www.ibm.com/docs/pt-br/qmf/11.2?topic=ddfmtuss-creating-subquery-retrieve-data-from-more-than-one-table